Oracle Machine Learning for R 2.0

October 30, 2023

R topics documented:

OREbase-package .................................. 4
OREdm-package .................................. 6
OREdplyr-package ................................. 7
OREds-package .................................. 8
OREeda-package .................................. 9
OREembed-package ................................ 10
OREgraphics-package ............................... 11
OREpredict-package ............................... 12
OREstats-package .................................. 13
ORExml-package .................................. 14
arrows-methods .................................. 15
as.ore ............................................. 16
binom.test-methods ............................... 17
boxplot.ore ....................................... 18
cdplot.ore ....................................... 19
chisq.test-methods ............................... 19
clusterhists ..................................... 20
desc ............................................... 22
distinct ........................................... 22
fitdistr-methods ................................. 23
group_by ......................................... 24
hist.ore .......................................... 25
identify.ore ..................................... 26
is.ore ............................................ 27
join ............................................... 28
ks.test-methods .................................. 29
lines.ore ......................................... 30
n ..................................................... 31
nth .................................................. 31
n_distinct ......................................... 32
ore-boxplot.stats .................................. 32
ore-class ......................................... 33
ore-coplot ....................................... 34
ore-data-image-class .............................. 34
ore-matplot ..................................... 35
ore-polygon ....................................... 36
ore-polypath ..................................... 37
<table>
<thead>
<tr>
<th>R topics documented:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ore-rug</td>
<td>37</td>
</tr>
<tr>
<td>ore-segments</td>
<td>38</td>
</tr>
<tr>
<td>ore-smoothScatter</td>
<td>39</td>
</tr>
<tr>
<td>ore-symbols</td>
<td>39</td>
</tr>
<tr>
<td>ore-xspline</td>
<td>40</td>
</tr>
<tr>
<td>ore-xy.coords</td>
<td>41</td>
</tr>
<tr>
<td>ore.attach</td>
<td>41</td>
</tr>
<tr>
<td>ore.character-class</td>
<td>43</td>
</tr>
<tr>
<td>ore.connect</td>
<td>44</td>
</tr>
<tr>
<td>ore.const</td>
<td>47</td>
</tr>
<tr>
<td>ore.corr</td>
<td>48</td>
</tr>
<tr>
<td>ore.create</td>
<td>50</td>
</tr>
<tr>
<td>ore.crosstab</td>
<td>52</td>
</tr>
<tr>
<td>ore.datastore</td>
<td>53</td>
</tr>
<tr>
<td>ore.datastoreSummary</td>
<td>55</td>
</tr>
<tr>
<td>ore.datetime-class</td>
<td>57</td>
</tr>
<tr>
<td>ore.delete</td>
<td>60</td>
</tr>
<tr>
<td>ore.doEval</td>
<td>61</td>
</tr>
<tr>
<td>ore.esm</td>
<td>67</td>
</tr>
<tr>
<td>ore.exec</td>
<td>70</td>
</tr>
<tr>
<td>ore.exists</td>
<td>71</td>
</tr>
<tr>
<td>ore.factor-class</td>
<td>72</td>
</tr>
<tr>
<td>ore.frame-class</td>
<td>73</td>
</tr>
<tr>
<td>ore.frame-factanal</td>
<td>76</td>
</tr>
<tr>
<td>ore.frame-model.frame</td>
<td>77</td>
</tr>
<tr>
<td>ore.frame-prcomp</td>
<td>79</td>
</tr>
<tr>
<td>ore.frame-princomp</td>
<td>80</td>
</tr>
<tr>
<td>ore.freq</td>
<td>82</td>
</tr>
<tr>
<td>ore.get</td>
<td>83</td>
</tr>
<tr>
<td>ore.getXlevels</td>
<td>84</td>
</tr>
<tr>
<td>ore.grant</td>
<td>85</td>
</tr>
<tr>
<td>ore.hash</td>
<td>87</td>
</tr>
<tr>
<td>ore.hiveOptions</td>
<td>88</td>
</tr>
<tr>
<td>ore.impalaOptions</td>
<td>89</td>
</tr>
<tr>
<td>ore.itemsets-class</td>
<td>90</td>
</tr>
<tr>
<td>ore.lazyLoad</td>
<td>91</td>
</tr>
<tr>
<td>ore.list-class</td>
<td>93</td>
</tr>
<tr>
<td>ore.load</td>
<td>94</td>
</tr>
<tr>
<td>ore.logical-class</td>
<td>95</td>
</tr>
<tr>
<td>ore.ls</td>
<td>96</td>
</tr>
<tr>
<td>ore.make.names</td>
<td>97</td>
</tr>
<tr>
<td>ore.matrix-class</td>
<td>98</td>
</tr>
<tr>
<td>ore.move</td>
<td>100</td>
</tr>
<tr>
<td>ore.number-class</td>
<td>102</td>
</tr>
<tr>
<td>ore.number-multivar</td>
<td>104</td>
</tr>
<tr>
<td>ore.number-univar</td>
<td>105</td>
</tr>
<tr>
<td>ore.object-class</td>
<td>107</td>
</tr>
<tr>
<td>ore.odmAI</td>
<td>108</td>
</tr>
<tr>
<td>ore.odmAssocRules</td>
<td>109</td>
</tr>
<tr>
<td>ore.odmDT</td>
<td>112</td>
</tr>
<tr>
<td>ore.odmEM</td>
<td>115</td>
</tr>
<tr>
<td>ore.odmESA</td>
<td>119</td>
</tr>
</tbody>
</table>
R topics documented:

ore.odmESM .............................................................. 122
ore.odmGLM .............................................................. 123
ore.odmKMeans ......................................................... 128
ore.odmNB ............................................................... 133
ore.odmNMF ............................................................. 136
ore.odmNN ............................................................... 138
ore.odmOC ............................................................... 140
ore.odmRAlg ............................................................ 144
ore.odmRF ............................................................... 148
ore.odmSVD ............................................................. 150
ore.odmSVM ............................................................. 153
ore.odmXGB ............................................................ 157
ore.options ............................................................. 160
ore.predict ............................................................. 161
ore.predict-glm ......................................................... 162
ore.predict-kmeans .................................................... 163
ore.predict-lm .......................................................... 164
ore.predict-matrix ..................................................... 165
ore.predict-multinom .................................................. 167
ore.predict-nnet ......................................................... 168
ore.predict-ore.model ................................................ 169
ore.predict-pcomp ..................................................... 170
ore.predict-princomp .................................................. 171
ore.predict-rpart ....................................................... 172
ore.pull ................................................................. 173
ore.rank ............................................................... 175
ore.raw-class .......................................................... 177
ore.recode ............................................................. 178
ore.rm ................................................................. 179
ore.roll ............................................................... 180
ore.rules-class .......................................................... 181
ore.save .............................................................. 183
ore.scriptCreate ....................................................... 185
ore.showHiveOptions .................................................. 187
ore.showImpalaOptions ................................................. 188
ore.sort ............................................................... 188
ore.summary .......................................................... 190
ore.sync .............................................................. 193
ore.toXML ............................................................. 194
ore.univariate .......................................................... 195
ore.vector-aggregate .................................................. 197
ore.vector-ave .......................................................... 198
ore.vector-class ........................................................ 199
ore.year .............................................................. 201
OREDistributions ........................................................ 202
OREShowDoc ............................................................ 203
pairs.ore.frame .......................................................... 204
partitions .............................................................. 204
plot.ore.vector .......................................................... 206
points.ore ............................................................. 207
prop.test-methods ...................................................... 207
ranking ................................................................. 208
### Description

The **OREbase** package implements the Oracle R Enterprise transparency layer for R’s **base** package.

### Details

<table>
<thead>
<tr>
<th>Package</th>
<th>OREbase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Package</td>
</tr>
<tr>
<td>Version</td>
<td>2.0</td>
</tr>
<tr>
<td>Date</td>
<td>2023-10-16</td>
</tr>
<tr>
<td>Depends</td>
<td>methods, utils, OREcommon</td>
</tr>
<tr>
<td>Suggests</td>
<td>DBI, ROracle (&gt;= 1.1-12)</td>
</tr>
<tr>
<td>License</td>
<td>file LICENSE</td>
</tr>
</tbody>
</table>


**LazyLoad**: yes
Index:

OREbase-package Oracle R Enterprise Base Functions
as.ore Oracle R Enterprise Coercion Functions
is.ore Oracle R Enterprise Type Checking Functions
ore-class Class ore
ore.attach Oracle R Enterprise Environment Management Functions
ore.character-class Class ore.character
ore.connect Oracle R Enterprise Connection Functions
ore.const Oracle R Enterprise Constants
ore.create Database Table Creation or Destruction Functions
ore.datetime-class Class ore.datetime
ore.exec SQL Query Execution Function
ore.exists Oracle R Enterprise Object Existence Checking Function
ore.factor-class Class ore.factor
ore.frame-class Class ore.frame
ore.get Oracle R Enterprise Object Retrieval Function
ore.grant Oracle R Enterprise Privilege Grant and Revoke Functions
ore.hash Oracle R Enterprise ore.vector Hash Function
ore.logical-class Class ore.logical
ore.ls Oracle R Enterprise Object Listing Function
ore.make.names Oracle R Enterprise Valid Column Name Generator
ore.matrix-class Class ore.matrix
ore.number-class Class ore.number
ore.options Oracle R Enterprise Global Options
ore.pull Oracle R Enterprise Data Exchange Functions
ore.raw-class Class ore.raw
ore.recode Oracle R Enterprise ore.vector Value Recode Function
ore.rm Oracle R Enterprise Object Removal Function
ore.sync Oracle R Enterprise Object Synchronization Function
ore.vector-class Class ore.vector
ore.year Oracle R Enterprise Date/Time Functions

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.character, ore.datetime, ore.factor, ore.frame, ore.logical, ore.matrix, ore.number, ore.raw, ore.vector
Examples

```r
# Transparency layer function overloads of the base package
intersect(ls("package:OREbase"), ls("package:base"))

# Transparency layer function overloads of the methods package
intersect(ls("package:OREbase"), ls("package:methods"))
```

Description

The **OREdm** package provides access to the in-database data mining functionality of Oracle Database. The Oracle R Enterprise Data Mining package contains several data mining and data analysis algorithms for classification, regression, clustering, attribute importance, and anomaly detection.

Details

Package: OREdm
Type: Package
Version: 2.0
Date: 2023-10-16
Depends: OREbase (>= 2.0)
Imports: utils, methods, lattice, stats, ROracle, OREstats
Suggests: arules
License: file LICENSE
LazyLoad: yes
https://community.oracle.com/community/developer/english/business_intelligence/data_warehousing/r

Index:

- OREdm-package Oracle R Enterprise data mining package
- clusterhist Cluster Histogram Data from Clustering Models
- ore.itemsets-class Class ore.itemsets
- ore.odmAI In-Database Attribute Importance Ranking Models
- ore.odmAssocRules In-Database Association Rules Models
- ore.odmDT In-Database Decision Tree Models
- ore.odmEM In-Database Expectation Maximization Models
- ore.odmESA In-Database Explicit Semantic Analysis Models
- ore.odmGLM In-Database Generalized Linear Models
- ore.odmKMeans In-Database Hierarchical K-Means Models
- ore.odmNB In-Database Naive Bayes Models
- ore.odmNMF In-Database Non-Negative Matrix Factorization Models
- ore.odmOC In-Database Orthogonal Partitioning Cluster Models
- ore.odmRAlg Extensible R Algorithm Models
- ore.odmSVD In-Database Singular Value Decomposition Models
- ore.odmSVM In-Database Support Vector Machine Models
- ore.rules-class Class ore.rules
The functions above are used to build data mining models in Oracle Database. Each function implicitly uses the Oracle R Enterprise database connection and takes a formula object to specify the predictors (terms) and target (response).

To use this package with Oracle Database 11g, Oracle Data Mining must be enabled. To use this package with Oracle Database 12c, the Oracle Advanced Analytics option must be enabled.

Author(s)
Oracle
Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References
Oracle R Enterprise
Oracle Data Mining Concepts 11g Release 2 (11.2) http://download.oracle.com/docs/cd/E11882_01/datamine.112/e16808/toc.htm

See Also
ore.connect

The OREdplyr package contains functions for data manipulation.

Details
Package: OREdplyr
Type: Package
Version: 2.0
Date: 2023-10-25
Depends: OREbase
Suggests: dplyr, magrittr
Imports: methods
License: file LICENSE
LazyLoad: yes
 https://community.oracle.com/community/developer/english/business_intelligence/data_warehousing/r
desc  Descending Order
distinct  Select Distinct Rows
group_by  Group Rows by One or More Columns
join  Join Two ore.frame Objects
n  The number of rows in the current group
n_distinct  Number of Unique Values
nth  Get the nth Value
ranking  Various Ranking Functions
sample  Sample Rows
select  Select, Rename, Arrange, Filter, Mutate, or Transmute
slice  Select Rows by Positions
summarise  Summarise Columns by Aggregate Functions
tally  Counts or Tallies Rows by Group
top_n  Select Top n Rows

Author(s)
Oracle <oracle-r-enterprise@oracle.com>

References
Oracle R Enterprise

See Also
desc, distinct, group_by, join, n, n_distinct, nth, ranking, sample, select,
slice, summarise, tally, top_n

OREds-package  Oracle R Enterprise Datastore Functions

Description
The OREds package contains functions for managing R objects stored within the Oracle database.

Details
Package: OREds
Type: Package
Version: 2.0
Date: 2023-10-16
Depends: OREbase
Suggests: rpart
Imports: methods, OREcommon, ROracle (>= 1.1-12), utils
License: file LICENSE
Collate: persist.R
LazyLoad: yes
https://community.oracle.com/community/developer/english/business_intelligence/data_warehousing/r
OREeda-package

Index:

OREds-package Oracle R Enterprise Embedded R Functions
ore.datastore Oracle R Enterprise Datastore Listing Function
ore.datastoreSummary Oracle R Enterprise Datastore Summarizing Function
ore.delete Oracle R Enterprise Datastore Removal Function
ore.lazyLoad Oracle R Enterprise Datastore Lazy Loading Function
ore.load Oracle R Enterprise Datastore Loading Function
ore.move Oracle R Enterprise Datastore Moving Function
ore.save Oracle R Enterprise Datastore Saving Function

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.save, ore.load, ore.delete, ore.datastore, ore.datastoreSummary

OREeda-package Oracle R Enterprise Exploratory Data Analysis Functions

Description

The OREeda package contains functions that allow different types of analysis of numeric columns in the input object of type ore.frame with flexible aggregation, ranking, cross-tabulation, and sorting functionality.

Details

Package: OREeda
Type: Package
Version: 2.0
Date: 2023-10-17
Depends: OREbase (>= 2.0)
Imports: methods, stats, DBI, OREbase, OREembed, OREstats
Suggests: ROracle (>= 1.1-12), OREgraphics
License: file LICENSE
LazyLoad: yes
  https://community.oracle.com/community/developer/english/business_intelligence/data_warehousing/r

Index:

ore.corr Oracle R Enterprise Correlation Analysis
OREembed-package

<table>
<thead>
<tr>
<th>ore.crosstab</th>
<th>Oracle R Enterprise Crosstabulations</th>
</tr>
</thead>
<tbody>
<tr>
<td>ore.esm</td>
<td>Oracle R Enterprise Time Series Exponential Smoothing Models</td>
</tr>
<tr>
<td>ore.freq</td>
<td>Oracle R Enterprise Frequency Analysis</td>
</tr>
<tr>
<td>ore.rank</td>
<td>Oracle R Enterprise Data Ranking</td>
</tr>
<tr>
<td>ore.sort</td>
<td>Oracle R Enterprise Data Sorting</td>
</tr>
<tr>
<td>ore.summary</td>
<td>Oracle R Enterprise Data Summary</td>
</tr>
<tr>
<td>ore.univariate</td>
<td>Oracle R Enterprise Univariate Summaries</td>
</tr>
</tbody>
</table>

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.corr, ore.crosstab, ore.freq, ore.rank, ore.sort, ore.summary, ore.univariate

OREembed-package  Oracle R Enterprise Embedded R Functions

Description

The OREembed package contains functions for using the Oracle R Enterprise embedded R functionality.

Details

<table>
<thead>
<tr>
<th>Package:</th>
<th>OREembed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Package</td>
</tr>
<tr>
<td>Version:</td>
<td>2.0</td>
</tr>
<tr>
<td>Date:</td>
<td>2023-10-17</td>
</tr>
<tr>
<td>Depends:</td>
<td>OREbase</td>
</tr>
<tr>
<td>Suggests:</td>
<td>rpart</td>
</tr>
<tr>
<td>Imports:</td>
<td>methods, OREcommon, OREDs, ROracle (&gt;= 1.1-12), grDevices, grid, png, utils</td>
</tr>
<tr>
<td>License:</td>
<td>file LICENSE</td>
</tr>
<tr>
<td>LazyLoad:</td>
<td>yes</td>
</tr>
</tbody>
</table>

Index:

<table>
<thead>
<tr>
<th>OREembed-package</th>
<th>Oracle R Enterprise Embedded R Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ore.doEval</td>
<td>Oracle R Enterprise Embedded R Script Execution Functions</td>
</tr>
<tr>
<td>ore.list-class</td>
<td>Class ore.list</td>
</tr>
<tr>
<td>ore.object-class</td>
<td>Class ore.object</td>
</tr>
</tbody>
</table>
The `OREgraphics` package implements the Oracle R Enterprise transparency layer for R's `graphics` package.

**Details**

- **Package:** OREgraphics
- **Type:** Package
- **Version:** 2.0
- **Date:** 2023-10-17
- **Depends:** methods, grDevices, graphics, OREbase (>= 2.0)
- **Imports:** stats, OREstats
- **License:** file LICENSE
- **LazyLoad:** yes
- **URL:**
  - https://community.oracle.com/community/developer/english/business_intelligence/data_warehousing/r

**Index:**

- `ore-arrows` 'arrows' methods for Oracle R Enterprise
- `ore-boxplot` 'boxplot' methods for Oracle R Enterprise
- `ore-boxplot.stats` 'boxplot.stats' methods for Oracle R Enterprise
- `ore-cdplot` 'cdplot' methods for Oracle R Enterprise
- `ore-coplot` 'coplot' and 'co.intervals' methods for Oracle R Enterprise
- `ore-hist` 'hist' methods for Oracle R Enterprise
- `ore-identify` 'identify' methods for Oracle R Enterprise
- `ore-matplot` 'matplot', 'matlines', and 'matpoints' methods for Oracle R Enterprise
- `ore-pairs` 'pairs' methods for Oracle R Enterprise
OREpredict-package

ore-plot
'plot' methods for Oracle R Enterprise
ore-points
'points' methods for Oracle R Enterprise
ore-polygon
'polygon' methods for Oracle R Enterprise
ore-polypath
'polypath' methods for Oracle R Enterprise
ore-rug
'rug' methods for Oracle R Enterprise
ore-segments
'segments' methods for Oracle R Enterprise
ore-smoothScatter
'smoothScatter' methods for Oracle R Enterprise
ore-sunflowerplot
'sunflowerplot' methods for Oracle R Enterprise
ore-symbols
'symbols' methods for Oracle R Enterprise
ore-text
'text' methods for Oracle R Enterprise
ore-xspline
'xspline' methods for Oracle R Enterprise
ore-xy.coords
'xy.coords' methods for Oracle R Enterprise

Author(s)
Oracle <oracle-r-enterprise@oracle.com>

References
Oracle R Enterprise

Description
The OREpredict package contains functions for generating model predictions using standard R models.

Details
Package: OREpredict
Type: Package
Version: 2.0
Date: 2023-10-17
Depends: utils, methods, stats, OREbase (>= 2.0), OREmodels
Imports: OREstats
Suggests: MASS, rpart, nnet
License: file LICENSE
LazyLoad: yes
                                  https://community.oracle.com/community/developer/english/business_intelligence/data_warehousing/r

Index:
ore.predict Generic for model predictions in Oracle R Enterprise
ore.predict-lm Method for lm models
ore.predict-glm Method for glm models
ore.predict-matrix Method for matrix objects
ore.predict-kmeans Method for kmeans models
ore.predict-prcomp Method for prcomp models
ore.predict-princomp Method for princomp models
ore.predict-rpart Method for rpart models
ore.predict-nnet Method for nnet models
ore.predict-multinom Method for multinom models
ore.predict-ore.lm Method for ore.lm models

Author(s)
Oracle <oracle-r-enterprise@oracle.com>

References
Oracle R Enterprise

OREstats-package  Oracle R Enterprise Statistical Functions

Description
The OREstats package implements the Oracle R Enterprise transparency layer for R's stats and MASS packages.

Details
Package: OREstats
Type: Package
Version: 2.0
Date: 2023-10-17
Depends: methods, stats, MASS, OREbase (>= 2.0)
Imports: OREembed
License: file LICENSE
LazyLoad: yes

https://community.oracle.com/community/developer/english/business_intelligence/data_warehousing/r

Index:
ore.factor-reorder Reorder Levels of an \code{ore.factor} Object
ore.frame-factanal Oracle R Enterprise Factor Analysis
ore.frame-model.frame Oracle R Enterprise Modeling Framework
ore.frame-princomp Oracle R Enterprise Principal Components Analysis
ore.frame-prcomp Oracle R Enterprise Principal Components Analysis
ore.frame-svd Oracle R Enterprise Singular Value Decomposition
ore.getXlevels Factor Levels for an Oracle R Enterprise Model Matrix
ore.number-Distributions Oracle R Enterprise Distribution Functions
ore.number-fitdistr Oracle R Enterprise Maximum Likelihood Estimation of Univariate Distributions
ore.number-multivar Oracle R Enterprise Multivariate Statistics
ORExml-package

ore.number-univar  Oracle R Enterprise Univariate Statistics
ore.roll  Oracle R Enterprise Rolling Window Aggregates
ore.tblmatrix-svd  Oracle R Enterprise Singular Value Decomposition
ore.vecmatrix-svd  Oracle R Enterprise Singular Value Decomposition
ore.vector-aggregate  Oracle R Enterprise Univariate Aggregation by a Set of Grouping Variables
ore.vector-ave  Oracle R Enterprise Univariate Operations within a Set of Grouping Variables
ore.vector-binom.test  Oracle R Enterprise Exact Binomial Test
ore.vector-chisq.test  Oracle R Enterprise Pearson's Chi-squared Test for Count Data
ore.vector-ks.test  Oracle R Enterprise Kolmogorov-Smirnov Tests
ore.vector-prop.test  Oracle R Enterprise Test of Equal or Given Proportions
ore.vector-t.test  Oracle R Enterprise Student's t-Test
ore.vector-var.test  Oracle R Enterprise F Test to Compare Two Variances
ore.vector-wilcox.test  Oracle R Enterprise Wilcoxon Rank Sum and Signed Rank Tests

Author(s)
Oracle <oracle-r-enterprise@oracle.com>

References
Oracle R Enterprise

Examples

# Transparency layer function overloads of the stats package
intersect(ls("package:OREstats"), ls("package:stats"))

ORExml-package  Oracle R Enterprise XML String Generation Functions

Description

The ORExml package contains functions for converting R objects to XML.

Details

Package: ORExml
Type: Package
Version: 2.0
Date: 2023-10-25
Depends: OREbase
Imports: methods
License: file LICENSE
Collate: ORExml.R
LazyLoad: yes
Index:

ORExml-package Oracle R Enterprise Embedded R Functions
ore.toXML Oracle R Enterprise XML String Generation

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.toXML

arrows-methods Add Arrows to a Plot

Description

Draws arrows between pairs of points.

Usage

```r
## S4 method for signature 'ore.vector'
arrows(x0, y0, x1 = x0, y1 = y0, length = 0.25,
angle = 30, code = 2, col = par("fg"), lty = par("lty"),
lwd = par("lwd"), ...)
```

Arguments

- `x0, y0, x1, y1` ore.number objects.
- `length, angle, code, col, lty, lwd, ...`
  See description in `arrows`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

arrows
as.ore  Oracle R Enterprise Coercion Functions

Description

Coerces objects to ore class types.

Usage

as.ore(x, ...)
as.ore.vector(x, mode = "any")
as.ore.logical(x, ...)
as.ore.integer(x, ...)
as.ore.numeric(x, ...)
as.ore.character(x, ...)
as.ore.factor(x, ...)
as.ore.date(x, ...)
as.ore.datetime(x, ...)
as.ore.difftime(x, ...)
as.ore.frame(x, ...)
as.ore.matrix(x, ...)

Arguments

x     An object.
mode  A character string specifying an atomic mode or "any". The supported atomic
       modes are "logical", "integer", "numeric" (synonym "double"),
       and "character".
...   Additional arguments.

Details

The as.ore function coerces in-memory R objects to ore objects. It is similar to the ore.push function, but whereas ore.push may not create an ore object if no mapping exists, the as.ore function will throw an error if it cannot create an ore object. See ore.push for more information on the mapping from in-memory R objects to ore objects.

Value

An ore object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore, ore.push, is.ore, as.vector
Examples

```r
df <- data.frame(A = 1:26, B = letters)
dim(df)
class(df)
oreDF <- as.ore(df)
class(oreDF)
dim(oreDF)
head(oreDF)

vec <- 1:20
class(vec)
oreVec <- as.ore(vec)
class(oreVec)
```

binom.test-methods  
**Oracle R Enterprise Exact Binomial Test**

Description

Performs an exact test of a simple null hypothesis about the probability of success in a Bernoulli experiment.

Usage

```r
## S4 method for signature 'ore.vector'
binom.test(x, n, p = 0.5,
    alternative = c("two.sided", "less", "greater"),
    conf.level = 0.95)
```

Arguments

- **x**: An `ore.vector` object with exactly two distinct values.
- **n**: Argument not supported.
- **p**: A numeric value specifying the hypothesized probability of success.
- **alternative**: A character string specifying the alternative hypothesis and must be one of "two.sided", "greater" or "less".
- **conf.level**: A numeric value specifying the confidence level for the returned confidence interval.

Details

The `binom.test` method in OREstats differs from the `binom.test` function in the `stats` package in that the `ore.vector` object represents the "raw" successes and failures instead of the aggregate counts of those successes and failures.

Value

Returns an `binom.test` `htest` object.
Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References


Oracle R Enterprise

See Also

*binom.test*

---

**boxplot.ore**

*Create Box Plots*

**Description**

Creates Box-and-Whisker Plots

**Usage**

```r
## S3 method for class 'ore'
boxplot(x, ..., range = 1.5, width = NULL, varwidth = FALSE,
notch = FALSE, outline = TRUE, names, plot = TRUE,
border = par("fg"), col = NULL, log = "",
pars = list(boxwex = 0.8, staplewex = 0.5, outwex = 0.5),
horizontal = FALSE, add = FALSE, at = NULL)
```

**Arguments**

- `x` An *ore.number* object, a list of *ore.number* objects or a formula, e.g. `y ~ grp`, where `y` is an *ore.number* object and `grp` is an *ore.factor* object.
- `...` range, width, varwidth, notch, outline, names, plot, border, col, log, pars, horizontal
  - See description in *boxplot*.

**Author(s)**

Oracle <oracle-r-enterprise@oracle.com>

**References**

Oracle R Enterprise

**See Also**

*boxplot*
cdplot.ore

Conditional Density Plots

Description
Computes and plots conditional densities.

Usage
## S3 method for class 'ore'
cdplot(x, y, plot = TRUE, tol.ylab = 0.05, ylevels = NULL,
bw = "nrd0", n = 512, from = NULL, to = NULL, col = NULL,
border = 1, main = "", xlab = NULL, ylab = NULL,
yaxlabels = NULL, xlim = NULL, ylim = c(0, 1), ...)

Arguments
x An ore.number object.
y An ore.factor object.
plot, tol.ylab, ylevels, bw, n, from, to, col, border, main, xlab, ylab, yaxlabels, xlim, ylim
See description in cdplot.

Author(s)
Oracle <oracle-r-enterprise@oracle.com>

References
Oracle R Enterprise

See Also
cdplot

chisq.test-methods

Oracle R Enterprise Pearson’s Chi-squared Test for Count Data

Description
Performs chi-squared contingency table tests and goodness-of-fit tests.

Usage
## S4 method for signature 'ore.vector'
chisq.test(x, y = NULL, correct = TRUE,
p = rep(1/length(x), length(x)), rescale.p = FALSE,
simulate.p.value = FALSE, B = 2000)
Arguments

x  An `ore.vector` object.
y  An `ore.vector` object or a NULL value.
correct  A logical value indicating whether to apply continuity correction when computing the test statistic for 2 by 2 tables: one half is subtracted from all \(|O - E|\) differences. No correction is done if `simulate.p.value = TRUE`.
p  If argument `y = NULL`, a vector of probabilities of the same length as argument `x`.
rescale.p  A logical value indicating if `p` should be rescaled (if necessary) to sum to 1.
simulate.p.value  A logical value indicating whether to compute p-values by Monte Carlo simulation.
B  An integer value specifying the number of replicates to use in the Monte Carlo test.

Value

Returns an `chisq.test` `htest` object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References


Oracle R Enterprise

See Also

`chisq.test`

---

**clusterhists**  
Cluster Histogram Data from Clustering Models

Description

The generic function `clusterhists` returns cluster histogram data, which may be used to generate histograms.
clusterhists

Usage

clusterhists(object,...)

## S3 method for class 'ore.odmKMeans'
clusterhists(object,...)
## S3 method for class 'ore.odmOC'
clusterhists(object,...)
## S3 method for class 'ore.odmEM'
clusterhists(object,...)

Arguments

object  Object for which cluster histogram data is desired.
...
  Additional arguments affecting the result.

Details

The function clusterhists provides the histogram data for each variable for each cluster. This function generates a data.frame with histogram data for each cluster and variable combination in the model. Numerical variables are binned. A label is produced combining the range of the bin for numerical variables. The label for categorical variables contains the value.

Value

The function clusterhists returns a data.frame with columns:

cluster.id  The numeric identifier associated with a cluster.
variable  The character name of the variable.
bin.id  The numeric identifier of a bin for a variable in the range of 1 to N where N is the number of bins generated if the variable is numeric, or the number of categories if the variable is categorical.
lower.bound  For a numeric variable, the lower bound of the bin.
upper.bound  For a numeric variable, the upper bound of the bin.
label  For a numeric variable, the concatenation of lower bound and upper bound. For a categorical variable, the category string.
count  The number of rows with this value range in the cluster for this variable, or the probability of the values in this value range appearing in the cluster for this variable.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User’s Guide
**distinct**

See Also

ore.odmKMeans, ore.odmOC ore.odmEM

Examples

```r
x <- rbind(matrix(rnorm(100, sd = 0.3), ncol = 2),
            matrix(rnorm(100, mean = 2, sd = 0.3), ncol = 2))
colnames(x) <- c("x", "y")
X <- ore.push(data.frame(x))

km.mod <- NULL
km.mod <- ore.odmKMeans(~., X, num.centers=2)
clusterhists(km.mod)
```

---

**desc**

**Descending Order**

Description

Sorts an ore.number, ore.factor, or ore.character object in descending order.

Usage

desc(x)

Arguments

- **x**
  
The ore.vector object to transform.

Examples

```r
desc(ore.push(1:10))
desc(ore.push(factor(letters)))
```

---

**distinct**

**Select Distinct Rows**

Description

Retains unique rows from an input ore.frame object over the specified columns.

Usage

distinct(.data, ...)

distinct_(.data, ..., .dots)
Arguments

.data An ore.frame object.
... Columns to determine the uniqueness.
.dots Used to work around non-standard evaluation. See distinct for details.

Examples

df <- data.frame(
  x = sample(10, 100, rep = TRUE),
  y = sample(10, 100, rep = TRUE)
)
DF <- ore.push(df)
nrow(DF)
nrow(distinct(DF))
arrange(distinct(DF, x), x)
arrange(distinct(DF, y), y)

# Use distinct on computed variables
arrange(distinct(DF, diff = abs(x - y)), diff)

fitdistr-methods   Oracle R Enterprise Maximum Likelihood Estimation of Univariate Distributions

Description

Maximum-likelihood fitting of univariate distributions, allowing parameters to be held fixed if desired.

Usage

## S4 method for signature 'ore.number'
fitdistr(x, densfun, start, ...)

Arguments

x An ore.number object.
densfun A character string specifying the density function; one of "beta", "cauchy", "chi-squared", "exponential", "f", "gamma", "geometric", "log-normal", "lognormal", "logistic", "negative binomial", "normal", "Poisson", "t" and "weibull".
start A named list giving the parameters to be optimized with initial values. This can be omitted for some of the named distributions and must be for others.
... Additional parameters, either for 'densfun' or for 'optim'.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>
group_by

Group Rows by One or More Columns

Description

group_by Groups an ore.frame object over the specified columns. Functions summarise,
top_n, count can perform on such an ore.frame object group-wise. ungroup Drops the
grouping from the input ore.frame object. group_size Lists the number of rows in each
group. groups Shows the names of the grouping columns. n_groups Gives the number of
groups.

Usage

  group_by(.data, ..., add = FALSE)
  group_by_(.data, ..., .dots, add = FALSE)

  groups(x)
  ungroup(x)

  group_size(x)
  n_groups(x)

Arguments

  .data       An ore.frame object.
  x           A grouped ore.frame object.
  ...         Columns to group by.
  add         By default, add = FALSE, group_by overrides the existing groups. Use add = TRUE to add new groups.
  .dots       Used to work around non-standard evaluation. See group_by_ for details.

See Also

summarise
Examples

```r
MTCARS <- ore.push(mtcars)
by_cyl <- group_by(MTCARS, cyl)
arrange(summarise(by_cyl, mean(disp), mean(hp)), cyl)

# summarise drops one layer of grouping
by_vs_am <- group_by(MTCARS, vs, am)
by_vs <- summarise(by_vs_am, n = n())
arrange(by_vs, vs, am)
arrange(summarise(by_vs, n = sum(n())), vs)

# remove grouping
summarise(ungroup(by_vs), n = sum(n()))

# group by expressions with mutate
arrange(group_size(group_by(mutate(MTCARS, vsam = vs + am), vsam)), vsam)

# rename the grouping column
groups(rename(group_by(MTCARS, vs), vs2 = vs))

# add more grouping columns
groups(group_by(by_cyl, vs, am))
groups(group_by(by_cyl, vs, am, add = TRUE))

# Duplicate groups are dropped
groups(group_by(by_cyl, cyl, cyl))

library(magrittr)
by_cyl_gear_carb <- MTCARS %>% group_by(cyl, gear, carb)
n_groups(by_cyl_gear_carb)
arrange(group_size(by_cyl_gear_carb), cyl, gear, carb)
by_cyl <- MTCARS %>% group_by(cyl)

# number of groups
n_groups(by_cyl)

# size of each group
arrange(group_size(by_cyl), cyl)
```

---

### hist.ore

#### Histograms

**Description**

Computes histograms.

**Usage**

```r
## S3 method for class 'ore'
hist(x, breaks = "Sturges", freq = NULL,
     probability = !freq, density = NULL,
     angle = 45, col = NULL, border = NULL,
     main = paste("Histogram of" , xname),
```
identify.ore

```r
xlim = range(breaks), ylim = NULL,
xlab = xname, ylab, axes = TRUE,
plot = TRUE, labels = FALSE,
warn.unused = TRUE, ...
```

Arguments

- **x**
  - An `ore.number` object.
- **breaks**
  - Either a single number giving the number of cells for the histogram or one of the following character strings: "Sturges", "Scott", "FD", or "Freedman-Diaconis". See Details in `hist`.
- freq, probability, density, angle, col, border, main, xlim, ylim, xlab, ylab, axes, plot, labels, warn.unused, ...
  - See description in `hist`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

- hist

---

**identify.ore**

### Identify Points in a Scatter Plot

**Description**

Interactively marks positions of a graphic.

**Usage**

```r
## S3 method for class 'ore'
identify(x, y = NULL, labels = seq_along(x), pos = FALSE,
n = length(x), plot = TRUE, atpen = FALSE,
offset = 0.5, tolerance = 0.25, ...)
```

Arguments

- **x, y**
  - `ore.number` objects.
- labels, pos, n, plot, atpen, offset, tolerance, ...
  - See description in `identify`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>
is.ore

References

Oracle R Enterprise

See Also

identify

is.ore

Oracle R Enterprise Type Checking Functions

Description

Returns logical value TRUE if the \( x \) argument is an object of the type named in the function; otherwise returns logical value FALSE.

Usage

\[
\begin{align*}
is.ore(x) \\
is.ore.vector(x) \\
is.ore.logical(x) \\
is.ore.integer(x) \\
is.ore.numeric(x) \\
is.ore.character(x) \\
is.ore.factor(x) \\
is.ore.date(x) \\
is.ore.datetime(x) \\
is.ore.difftime(x) \\
is.ore.frame(x) \\
is.ore.matrix(x)
\end{align*}
\]

Arguments

\( x \quad \text{An object.} \)

Value

The logical value TRUE if the \( x \) argument is an \texttt{ore} object of the specified type, or the logical value FALSE otherwise.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

\texttt{ore,as.ore,is.vector}
Examples

df <- data.frame(A = 1:26, B = letters)
is.ore(df)

oreDF <- as.ore(df)
is.ore(oreDF)

join

Join Two ore.frame Objects by the Specified Columns

Description

inner_join Returns all combination of rows from x and y over matched columns. left_join Returns rows from inner_join plus rows from x that do not match with y. For unmatched rows of x, NA is returned. right_join Returns rows from inner_join plus rows from y that do not match with x. For unmatched rows of y, NA is returned. full_join Returns the union of rows from left_join and right_join.

Usage

inner_join(x, y, by = NULL)
left_join(x, y, by = NULL)
right_join(x, y, by = NULL)
full_join(x, y, by = NULL)

Arguments

x, y The ore.frame objects to join.
by A character vector of column names to join by. The default is NULL. The join function does a natural join, using all columns with common names across the two tables.

To join by different columns on x and y, use a named vector string. For example, by = c("a" = "b") will match column a in x with column b in y.

Examples

MTCARS <- ore.push(mtcars)
M1 <- filter(select(MTCARS, mpg, cyl, carb), carb < 6L)
M2 <- filter(select(MTCARS, cyl, hp, carb), carb > 2L)

names(inner_join(M1, M2))
nrow(inner_join(M1, M2))
nrow(left_join(M1, M2))
nrow(right_join(M1, M2))
nrow(full_join(M1, M2))

names(M2) <- c("cyl", "hp", "carb2")
names(inner_join(M1, M2, by = c("cyl", carb="carb2")))
nrow(inner_join(M1, M2, by = c("cyl", carb="carb2")))
ks.test-methods

nrow(left_join(M1, M2, by = c("cyl", carb="carb2")))
nrow(right_join(M1, M2, by = c("cyl", carb="carb2")))
nrow(full_join(M1, M2, by = c("cyl", carb="carb2")))

Oracle R Enterprise Kolmogorov-Smirnov Tests

Description

Performs a one- or two-sample Kolmogorov-Smirnov test.

Usage

# one sample test
## S4 method for signature 'ore.number,character'
ks.test(x, y, ..., alternative = c("two.sided", "less", "greater"),
        exact = NULL)

# two sample test
## S4 method for signature 'ore.number,ore.number'
ks.test(x, y, ..., alternative = c("two.sided", "less", "greater"),
        exact = NULL)

# two sample test; use x as a grouping variable
## S4 method for signature 'ore.vector,ore.number'
ks.test(x, y, ..., alternative = c("two.sided", "less", "greater"),
        exact = NULL)

Arguments

x An ore.number object or an ore.factor object with exactly two levels.
y Either an ore.number object or one of the following character strings specifying a cumulative distribution function: "pexp", "pnorm", "ppois", "punif", "pweibull".
... Parameters of the distribution specified (as a character string) by argument y.
alternative A character string specifying the alternative hypothesis and must be one of "two.sided" (default), "less", or "greater".
exact Argument not supported.

Value

Returns an ks.test htest object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>
References


Durbin, J. (1973) Distribution theory for tests based on the sample distribution function. SIAM.


Oracle R Enterprise

See Also

ks.test

lines.ore

Add Connected Line Segments to a Plot

Description

Draws line segments.

Usage

## S3 method for class 'ore'
lines(x, y = NULL, type = "l", ...)

Arguments

x, y ore.number objects.

type,... See description in lines.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

lines
\[ n \] \hspace{1cm} The number of rows in the current group

**Description**

This function can only be used from within `summarise`, `mutate` and `filter` to get the number of rows in the group.

**Usage**

\( n() \)

**Examples**

```r
MTCARS <- ore.push(mtcars)
mtcars_cyl <- group_by(MTCARS, cyl)
arrange(summarise(mtcars_cyl, n = n()), cyl)
mutate(mtcars_cyl, n = n())
filter(mtcars_cyl, n() <= 10L)
```

\[ nth \] \hspace{1cm} Get the First, Last or nth Value from an Ordered `ore.vector` Object

**Description**

Obtains the value at the specified position in the order.

**Usage**

\( nth(x, n, \text{order\_by} = \text{NULL}, \text{default} = \text{NULL}) \)

\( first(x, \text{order\_by} = \text{NULL}, \text{default} = \text{NULL}) \)

\( last(x, \text{order\_by} = \text{NULL}, \text{default} = \text{NULL}) \)

**Arguments**

- \( x \) An ordered `ore.vector` object.
- \( n \) An integer specifying the position of a value in the order.
- \( \text{order\_by} \) An `ore.vector` or column used to determine the order.
- \( \text{default} \) not used.

**Value**

A scalar `ore.number`, `ore.character`, or `ore.factor` value.
Examples

```r
X <- ore.push(1:10)
Y <- ore.push(10:1)

nth(X, 2)
nth(X, 2, Y)

last(X)
last(X, order_by = Y)
```

---

### n_distinct

**Number of Unique Values in an ore.vector Object**

#### Description

Counts the number of unique values in an `ore.vector` object.

#### Usage

```r
n_distinct(x, na_rm = FALSE)
```

#### Arguments

- `x`: An `ore.vector` object.
- `na_rm`: Specifies whether to include NA value.

#### Examples

```r
x <- sample(1:10, 1e5, rep = TRUE)
X <- ore.push(x)
length(unique(X))
n_distinct(X)
```

---

### ore-boxplot.stats

**Box Plot Statistics**

#### Description

Calculates statistics necessary for producing box plots.

#### Usage

```r
## S4 method for signature 'ore.vector'
boxplot.stats(x, coef = 1.5, do.conf = TRUE, do.out = TRUE)
```
ore-class

Arguments

- **x**: An ore.number object.
- **coef**: A numeric value that specifying the maximum relative distance the whiskers can extend out from the box.
- **do.conf, do.out**: Logical values indicating if the `conf` or `out` elements of the list output are NULL values.

Value

Returns a list conforming to the output of `boxplot.stats`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`boxplot.stats`

ore-class  

Class ore

Description

The ore virtual class that sits at the top of the Oracle R Enterprise class hierarchy.

Note

See the corresponding R documentation for the functions listed above.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.character, ore.datetime, ore.factor, ore.frame, ore.logical, ore.matrix, ore.number, ore.vector

Examples

`showClass("ore")`
ore-coplot  Conditioning Plots

Description

Creates conditioning plots.

Usage

```r
## S4 method for signature 'ANY,ore.frame'
coplot(formula, data, given.values,
       panel = points, rows, columns, show.given = TRUE,
       col = par("fg"), pch = par("pch"),
       bar.bg = c(num = gray(0.8), fac = gray(0.95)),
       xlab, ylab, subscripts = FALSE,
       axlabels = function(f) abbreviate(levels(f)),
       number = 6, overlap = 0.5, xlim, ylim, ...)
```

Arguments

- `formula`: A formula describing the form of conditioning plot.
- `data`: An `ore.frame` object.
- `given.values`, `panel`, `rows`, `columns`, `show.given`, `col`, `pch`, `bar.bg`, `xlab`, `ylab`, `subscripts`, `axlabels`, `number`, `overlap`, `xlim`, `ylim`, `...`:
  See the descriptions of these arguments in `coplot`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

coplot

ore-data-image-class  Class ore-data-image

Description

A class containing R object(s) and PNG images returned from Embedded R Execution when graphical output is captured.

List Methods

- `x$dat`: R object returned from embedded R execution.
- `x$img`: A list of PNG images returned from embedded R execution.
ore-matplot

Plot Columns of Matrices

Description

Plots the columns of one matrix against the columns of another.

Usage

```r
## S4 method for signature 'ore'
matplot(x, y, type = "p", lty = 1:5, lwd = 1,
        lend = par("lend"), pch = NULL, col = 1:6, cex = NULL,
        bg = NA, xlab = NULL, ylab = NULL, xlim = NULL, ylim = NULL,
        ..., add = FALSE, verbose = getOption("verbose"))

## S4 method for signature 'ore'
matlines(x, y, type = "l", lty = 1:5, lwd = 1,
         pch = NULL, col = 1:6, ...)  

## S4 method for signature 'ore'
matpoints(x, y, type = "p", lty = 1:5, lwd = 1,
          pch = NULL, col = 1:6, ...)
```

Arguments

- `x, y` 
  ore objects.
- `type, lty, lwd, lend, pch, col, cex, bg, xlab, ylab, xlim, ylim, ...` 
  See description in `matplot`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>
**Description**

Draws polygons.

**Usage**

```r
## S4 method for signature 'ore'
polygon(x, y = NULL, density = NULL, angle = 45,
        border = NULL, col = NA, lty = par("lty"), ..., 
        fillOddEven = FALSE)
```

**Arguments**

- `x, y` *ore.number* objects.
- `density, angle, border, col, lty, ..., fillOddEven`

  See description in `polygon`.

**Author(s)**

Oracle <oracle-r-enterprise@oracle.com>

**References**

*Oracle R Enterprise*

**See Also**

`polygon`
Description

Draws paths.

Usage

```r
## S4 method for signature 'ore'
polypath(x, y = NULL, border = NULL, col = NA,
        lty = par("lty"), rule = "winding", ...)
```

Arguments

- `x,y` ore.number objects.
- `border,col,lty,rule,...`

See description in `polypath`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`polypath`

---

Description

Draws rug ticks.

Usage

```r
## S4 method for signature 'ore.vector'
rug(x, ticksize = 0.03, side = 1, lwd = 0.5,
     col = par("fg"), quiet = getOption("warn") < 0, ...)
```

Arguments

- `x` ore.number objects.
- `ticksize,side,lwd,col,quiet,...`

See description in `rug`. 
ore-segments

Add Line Segments to a Plot

Description

Draws line segments between pairs of points.

Usage

```r
## S4 method for signature 'ore.vector'
segments(x0, y0, x1 = x0, y1 = y0,
          col = par("fg"), lty = par("lty"), lwd = par("lwd"), ...)
```

Arguments

- `x0, y0, x1, y1` `ore.number` objects.
- `col, lty, lwd, ...`
  - `See description in segments.`

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`segments`
ore-smoothScatter  Scatterplots with Smoothed Densities Color Representation

Description

Creates a color kernel density smoothed representation of the scatterplot.

Usage

## S4 method for signature 'ore'
smoothScatter(x, y = NULL, nbin = 128, bandwidth,
colramp = colorRampPalette(c("white", blues9)),
nrpoints = 100, ret.selection = FALSE,
pch = ".", cex = 1, col = "black",
transformation = function(x) x^.25,
postPlotHook = box, xlab = NULL, ylab = NULL,
xlim, ylim, xaxs = par("xaxs"), yaxs = par("yaxs"),
...)

Arguments

x,y  
ore.number  objects.
nbin,bandwidth,colramp,nrpoints,ret.selection,pch,cex,col,transformation,postPlotHook,xlab,ylab,xlim,ylim,xaxs,yaxs,...

See description in smoothScatter.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

smoothScatter

ore-symbols  Draw Symbols (Circles, Squares, Stars, Thermometers, Boxplots)

Description

Draws symbols on a plot.

Usage

## S4 method for signature 'ore.vector'
symbols(x, y = NULL, circles, squares,
rectangles, stars, thermometers, boxplots, inches = TRUE,
add = FALSE, fg = par("col"), bg = NA,
xlab = NULL, ylab = NULL, main = NULL,
xlim = NULL, ylim = NULL,...)
ore-xspline

Arguments

- `x, y` 
  `ore.number` objects.
- `shape, open, repEnds, draw, border, col, ...`

See description in `symbols`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

symbols

---

Draw and X-spline

Description

Draws an X-spline, which is a curve drawn relative to control points.

Usage

```r
## S4 method for signature 'ore'
x spline(x, y = NULL, shape = 0, open = TRUE,
   repEnds = TRUE, draw = TRUE, border = par("fg"), col = NA,
   ...)
```

Arguments

- `x, y` 
  `ore.number` objects.
- `shape, open, repEnds, draw, border, col, ...
  See description in `xspline`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

xspline
**ore-xy.coords**

*Extract Plotting Structures*

**Description**

Obtains x and y coordinates for plotting.

**Usage**

```r
## S4 method for signature 'ore'
xy.coords(x, y = NULL, xlab = NULL, ylab = NULL,
          log = NULL, recycle = FALSE, ...)
```

**Arguments**

- `x, y` : `ore.number` objects.
- `xlab, ylab, log, recycle`
  See description in `xy.coords`.
- `...` : Additional arguments for future extension.

**Author(s)**

Oracle <oracle-r-enterprise@oracle.com>

**References**

Oracle R Enterprise

**See Also**

- `xy.coords`

---

**ore.attach**

*Oracle R Enterprise Environment Management Functions*

**Description**

Attaches or detaches the R `environment` of the named schema to or from the R `search` path in the Oracle R Enterprise session. When the R `environment` is attached to the R search path, the corresponding `ore.frame` objects, representing database tables and views, in the schema can be accessed directly by name.

**Usage**

```r
ore.attach(schema, pos = 2, warn.conflicts = TRUE)
ore.detach(schema)
```
ore.attach

Arguments

schema  A character string specifying the database schema name.
pos    An integer value larger than 1 specifying the position in the R search path.
warn.conflicts  A logical value indicating whether or not warnings should be printed for conflicts, objects that share the same name, that result from attaching the R environment.

Details

If argument schema is unspecified, the default schema - the one specified at connection time for the Oracle R Enterprise session - is used.

By default the R environment for a schema in the Oracle R Enterprise session is attached in position 2 in the R search path, immediately after the user’s workspace and before all previously attached packages and environment. The pos argument can be used to attach the R environment for a schema in the Oracle R Enterprise session at a different location in the search path, but it cannot be attached at pos = 1.

The functions ore.sync and ore.rm add and remove ore objects from this R environment for a schema in the Oracle R Enterprise session.

Value

For the ore.attach function, the environment is returned invisibly with a "name" attribute.

For the ore.detach function, the return value is an invisible NULL value.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.connect, ore.exists, ore.get, ore.ls, ore.rm, ore.sync, search

Examples

if (!interactive())
{
  search()
  ore.attach()
  search()
  ore.detach()
  search()
  ore.attach("rquser", 3)
  search()
  ore.detach("rquser")
  search()
}
The `ore.character` class represents character data columns in Oracle R Enterprise.

**Character Data Methods**

- `casefold(x, upper = FALSE)`: Returns an `ore.character` object containing the upper case or lower case form of argument `x`.
- `chartr(old, new, x)`: Returns an `ore.character` object containing a character translation version of argument `x`.
- `grepl(pattern, x)`: Returns an `ore.logical` object indicating whether or not the specified pattern argument is found in the elements of argument `x`.
- `gsub(pattern, replacement, x)`: Returns an `ore.character` object containing a version of argument `x` where all matches to argument pattern are replaced with argument replacement.
- `nchar(x, type = "chars")`: Returns an `ore.integer` object containing the number of characters or bytes for the elements in argument `x`.
- `paste(..., sep = " ")`: Returns an `ore.character` object containing a concatenation of the values in arguments `...`.
- `sub(pattern, replacement, x)`: Returns an `ore.character` object containing a version of argument `x` where the first match to argument pattern is replaced with argument replacement.
- `substr(x, start, stop)`: Returns an `ore.character` object containing a the specified substrings for the elements of argument `x` that begin at the location of argument `start` and end at the location of argument `stop`.
- `substring(text, first, last = 1000000L)`: Returns an `ore.character` object containing the specified substrings for the elements of argument `text` that begin at the location of argument `first` and end at the location of argument `last`.
- `tolower(x)`: Returns an `ore.character` object containing the lower case form of argument `x`.
- `toupper(x)`: Returns an `ore.character` object containing the upper case form of argument `x`.

**Note**

See the corresponding R documentation for the functions listed above.

**Author(s)**

Oracle <oracle-r-enterprise@oracle.com>

**References**

Oracle R Enterprise
ore.connect

Oracle R Enterprise Connection Functions

Description

Establishes or terminates a connection to an Oracle R Enterprise, Apache Hive or Apache Impala server.

Usage

ore.is.connected(type = c("ORACLE", "HIVE", "IMPALA"))
ore.connect(user = "", sid = "", host = "localhost", password = "", port = NULL, service_name = NULL, conn_string = NULL, all = FALSE, type = c("ORACLE", "HIVE", "IMPALA"), tzone = Sys.getenv("TZ"), schema = NULL, ...)
ore.disconnect()

Arguments

user If argument type = "ORACLE" (the default), then a character string specifying the Oracle Database user name. If argument type = "HIVE" or type = "IMPALA", then a character string specifying the Apache Hive or Apache Impala user name.

sid When argument type = "ORACLE" (the default), a character string specifying the Oracle Database SID or Oracle Wallet connection string. If type = "HIVE" or type = "IMPALA", then this argument is ignored.

host If argument type = "ORACLE" (the default), then a character string specifying the host name of the Oracle Database server. If type = "HIVE" or type = "IMPALA", then a character string specifying the host name of the Apache Hive or Apache Impala.

password If argument type = "ORACLE" (the default), then a character string specifying the Oracle Database user password. If type = "HIVE" or type = "IMPALA", then the connection attempt uses the Apache Hive or Apache Impala user name.

port If argument type = "ORACLE" (the default), then a number specifying the Oracle Database port (usually 1521). If type = "HIVE" or type = "IMPALA", then a number specifying the Apache Hive or Apache Impala port.

Examples

x <- ore.push("MiXeD cAsE 123")
class(x)
chartr("iXs", "why", x)
chartr("a-cX", "D-Fw", x)
tolower(x)
toupper(x)
ore.connect

service_name When argument type = "ORACLE" (the default), a character string specifying the service name to be used in the connection identifier for the Oracle Database instance. If type = "HIVE" or type = "IMPALA", then this argument is ignored.

conn_string When argument type = "ORACLE" (the default), a character string specifying the connection string used to connect to the Oracle Database instance. Usually used for a database connection with Oracle Wallet. If type = "HIVE" or type = "IMPALA", then this parameter is ignored.

all A logical value indicating whether to call functions ore.sync and ore.attach using their default arguments. This results in synchronizing and attaching all tables and views in the schema for the Oracle R Enterprise connection. The execution time for ore.sync grows linearly with the number of visible tables and views.

type A character string specifying the database type: either "ORACLE" (default) to connect to an Oracle Database instance or "HIVE" to connect to an Apache Hive server or type = "IMPALA" to connect to an Apache Impala server.

tzone A character string specifying the session time zone for both the R and database sessions. The default value of tzone is the value of the system environment variable TZ. If the value of tzone is either NA or an empty string, then UTC is used for both the R and database sessions. For type = "HIVE" or type = "IMPALA", this parameter is ignored.

schema A character string specifying the schema. For type = "HIVE" or type = "IMPALA", this parameter specifies the target database.

Additional parameters for a Hive or Impala connection when running Hiveserver2 in different modes such as:

- Hive Specific Configuration parameters: Use the parameter hiveconf to specify the list of Apache Hive Server configuration parameters to set for Hive connection.
- HiveServer2 with Kerberos authentication: Use the parameter principal to specify the Kerberos server principal for the host where HiveServer2 is running.
- Hiveserver2 with SSL: In this mode, use parameters such as ssl="true", sslTrustStore to specify the path to the client’s truststore file and trustStorePassword to specify the password for the truststore.
- Impala with Kerberos authentication: In this mode, use parameters such as AuthMech="1" to indicate Kerberos authentication. Use KrbRealm="realm.example.com" and KrbHostFQDN="Kerberos_host_name" to specify the realm and Kerberos Host FQDN (Fully Qualified Domain Name). Use KrbServiceName="impala" to specify the Kerberos service name for Impala.
- Any other configuration of HiveServer2: There are many such configuration combinations in which HiveServer2 could be running. Check the Hive Documentation for the various modes and parameters needed to connect in those modes.

Details

Functions ore.connect and ore.disconnect are called for their side effects, namely establishing or terminating a connection to an Oracle R Enterprise, Apache Hive or Apache Impala server. The call to function ore.connect must precede all other calls to Oracle R Enterprise, Apache Hive Apache Impala functionality (except ore.is.connected). There can only be one active
Oracle R Enterprise, Apache Hive or Apache Impala connection. An Oracle R Enterprise, Apache Hive or Apache Impala session can optionally end with a call to function `ore.disconnect`. An Oracle R Enterprise, Apache Hive or Apache Impala session is implicitly terminated when the R session ends.

Both the R and the database session time zones are set during the execution of function `ore.connect`. The session time zones are set to the value of the argument `tzone`, whose default value is equal to the system environment variable `TZ`. If both `TZ` and `tzone` are not specified, the session time zones are set to UTC.

If you are using Oracle Wallet to store username and password, then use the `conn_string` argument to pass the connection string for the wallet mapping (for more information refer to chapter 3 "Configuring Clients to Use the External Password Store" of Oracle Database Security Guide). In addition, the `service_name` argument can be used to pass service name information. (Refer to the chapter 2 of Oracle Database Net Services Administrator’s Guide for more information on service names).

If you are connecting to Hiveserver2 with Kerberos authentication, then use the parameter `principal` to specify the Kerberos server principal for the host where Hiveserver2 is running (See the example below).

If you are connecting to Impala with Kerberos authentication, then you must use parameters specific to Impala. Do not use the Hive with Kerberos parameters. (See the example below).

If Apache Sentry is enabled on a cluster, then it is fully supported and compatible with Apache Hive. All the conditions and access restrictions imposed by Apache Sentry on Hive tables are applied when accessing it through the ORE transparency layer.

Calling `ore.connect` during an active Oracle R Enterprise connection of any type disconnects that session before starting a new session.

**Value**

A logical value in the case of function `ore.is.connected` indicating whether an active connection to an Oracle Database instance or Apache Hive or Apache Impala server exists; otherwise an invisible NULL value.

**Author(s)**

Oracle <oracle-r-enterprise@oracle.com>

**References**

Oracle R Enterprise

**See Also**

`ore.attach`, `ore.exists`, `ore.get`, `ore.ls`, `ore.rm`, `ore.sync`

**Examples**

```r
## Not run:
# pass the username password during connect
if (!is.ore.connected())
  ore.connect("rquser", "orcl", "localhost", "rquser")
ore.ls()
ore.disconnect()
```
# pass the connect string for wallet mode
if (!is.ore.connected())
    ore.connect(conn_string = "<wallet_connect_string>")
ore.ls()
ore.disconnect()

# Apache Hive Connection
if (!is.ore.connected(type="HIVE"))
    ore.connect(host="localhost", port=10000, user="user", password="password", schema="default")
ore.ls()
ore.disconnect()

# Apache Impala Connection
if (!is.ore.connected(type="IMPALA"))
    ore.connect(host="localhost", port=21500, user="user", password="password", schema="default")
ore.ls()
ore.disconnect()

# Apache Hive with Kerberos authentication
if (!is.ore.connected(type="HIVE"))
    ore.connect(user="user", password="password", host="localhost", port=10000, type="HIVE")
ore.ls()
ore.disconnect()

# Apache Impala with Kerberos authentication
if (!is.ore.connected(type="IMPALA"))
    ore.connect(host="localhost", port=21050, AuthMech="1", KrbRealm="DEV.EXAMPLE.COM", type="IMPALA")
ore.ls()
ore.disconnect()

## End(Not run)

ore.const  

**Oracle R Enterprise Constants**

**Description**

Holds mathematical constants for SQL calculations.

**Usage**

`ore.const`

**Format**

An `environment` object containing:

"1_PI" 1/pi
"1_SQRT2PI" 1/sqrt(pi)
"2PI" 2*pi
"2_PI" 2/pi
"2_SQRTPI" 2/sqrt(pi)
"E" exp(1)
ore.const
ore.const$PI
ore.const$"2_SQRTPI"

ore.corr

"LN10" log(10)
"LN2" log(2)
"LNSQRT2PI" log(sqrt(2*pi))
"LOG10E" log10(e)
"LOG2E" log2(e)
"PI" pi
"PI_2" pi/2
"PI_4" pi/4
"SQRT1_2" 1/sqrt(2)
"SQRT2" sqrt(2)

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Examples

ore.const
ore.const$PI
ore.const$"2_SQRTPI"

ore.corr

*Oracle R Enterprise Correlation Analysis*

Description

Performs correlation analysis across numeric columns in *ore.frame* objects. Supports partial correlations with a control column specification and allows the specification of aggregations prior to computing correlations.

Usage

ore.corr(data, var, stats = "pearson", group.by = NULL, freq = NULL,
with = NULL, weight = NULL, partial = NULL)

Arguments

data
An *ore.frame* object.

var
A comma-separated character string specifying the names of numeric columns within argument *data*.

stats
A character string specifying the correlation type; one of "pearson" (default), "spearman" or "kendall".

group.by
An optional character vector specifying the group by column names within argument *data*. 
freq

An optional character string specifying a numeric column within argument `data` to use as a frequency count. If a frequency value is less than 1 or missing, the observation is excluded from correlation calculation. If a frequency value is not an integer, it is truncated.

with

An optional character vector specifying the numeric columns in argument `data` to pair with columns specified in argument `var`. For example, with `var = c('x1', 'x2')` and `with = c('y1', 'y2', 'y3')`, function `ore.corr` will compute the following correlation pairs: `(x1, y1), (x1, y2), (x1, y3), (x2, y1), (x2, y2) and (x2, y3).

weight

An optional character string specifying a numeric column within argument `data` to use as analytic weights. Cannot be used with `stats = "kendall"`.

partial

An optional character vector specifying the numeric columns within argument `data` to use as control variables for partial correlations.

Value

When argument `group.by` is not specified, returns an `ore.frame` object.

When argument `group.by` is specified, returns a list of `ore.frame` objects.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

cor

Examples

# Copy iris data set to the database
IRIS <- ore.push(iris)

# Pearson's correlation
x <- cor(iris[,1:3])
y <- ore.corr(IRIS, var = "Sepal.Length, Sepal.Width, Petal.Length")

# Kendall's tau
x <- cor(iris[,1:3], method = "kendall")
y <- ore.corr(IRIS, var = "Sepal.Length, Sepal.Width, Petal.Length",
              stats = "kendall")

# Partial correlation
y <- ore.corr(IRIS, var = "Sepal.Length, Sepal.Width, Petal.Length",
              partial = "Petal.Width")

# Group by partial correlation
y <- ore.corr(IRIS, var = "Sepal.Length, Sepal.Width, Petal.Length",
              partial = "Petal.Width", group.by = "Species")
ore.create  Database Table Creation or Destruction Functions

Description

Create an Oracle Database table from a `data.frame` or `ore.frame` object, or a view from an `ore.frame` object.
Drop a database table or view.

Usage

```r
ore.create(x, table = NULL, view = NULL, append = FALSE, ...)
ore.drop(table = NULL, view = NULL, model = NULL, silent = FALSE)
```

Arguments

- **x**: A `ore.frame` object. Can also be a `data.frame` if argument `table` is used.
- **table**: A character string specifying the name of the table. This argument cannot be used with argument `view` or `model`.
- **view**: A character string specifying the name of the view. This argument cannot be used with argument `table` or `model`.
- **model**: A character string specifying the name of the model. This argument cannot be used with argument `table` or `view`.
- **silent**: A boolean (TRUE or FALSE) indicating whether to suppress warning message in case object does not exists.
- **append**: A boolean value currently supported in Hive and Impala only, if set to TRUE, `data.frame/ore.frame` can be appended to an existing table. Default value is FALSE.
- **...**: Additional parameters for a Hive connection when using Hiveserver2 in different modes such as:
  - To Overwrite the current data in table: Use the parameter `overwrite` to specify whether to overwrite the data in the table supported in Hive and Impala.
  - type of connection: Use parameter `type` to specify the connection type. A character string specifying to create an INTERNAL/EXTERNAL table. Defaults to internal table

Details

Exactly one of arguments `table` or `view` must be specified, depending on whether a database table or view is desired for creation or dropping.

When a new object is created using function `ore.create`, it is automatically included into the list of objects that can be accessed via `ore.get` and `ore.ls` functions. It will also be added to the list of attached `ore.frame` objects if `ore.attach` was already called. Also for function `ore.create`, only an `ore.frame` object can be used with the `view` argument.

Similarly, when an object is removed using function `ore.drop`, it will no longer show up in output of `ore.ls` function. It will also be removed from a list of attached `ore.frame` objects.
If the input object contains column names that do not match the naming convention of the backend server they will be modified by `ore.make.names` function.

Function `ore.create` does not preserve `row.names` neither in the database table nor in the newly created `ore.frame` object.

Value

Both functions `ore.create` and `ore.drop` return an invisible `NULL` value.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore.ls`, `ore.exists`, `ore.sync`, `ore.rm`, `ore.attach`, `ore.connect`

Examples

```r
if (!interactive())
{
  ore.drop(table = "TEN_LETTERS", silent = TRUE)
  ore.drop(table = "IRIS_TABLE", silent = TRUE)
  ore.drop(view = "IRIS_VIEW", silent = TRUE)

  ore.create(data.frame(x = 1:10, y = letters[1:10]),
             table = "TEN_LETTERS")
  ore.exists("TEN_LETTERS")

  ore.create(iris, table = "IRIS_TABLE")
  ore.create(head(ore.get("IRIS_TABLE"), 10), view = "IRIS_VIEW")
  ore.exists("IRIS_VIEW")

  ore.drop(table = "TEN_LETTERS")
  ore.drop(table = "IRIS_TABLE")
  ore.drop(view = "IRIS_VIEW")

  X <- ore.push(iris)
  em1 <- OREdm::ore.odmEM(~., X, odm.settings = list(model_name = "emmod"))
  ore.drop(model = "emmod")
  ore.drop(model = "emmod")
  ore.drop(model = "emmod", silent = TRUE)
}
```
Description

Expands on function xtabs by supporting multiple columns with optional aggregations, weighting, and ordering options. Building cross-tabulation is a pre-requisite to using the function ore.freq.

Usage

ore.crosstab(expr, data, ..., group.by = NULL, order = NULL, weights = NULL, where = NULL, strata = NULL)

Arguments

- **expr**: A formula object defining a cross-tabulation. Syntax: `[CS]~CS[<WC>]/[GC][^[SC]][<OC>`
  where `CS` is `[+CSET]` `[<CRANGE>` and `CRANGE` is `<FROM COLUMN>-<TO COLUMN>`.
- **data**: An ore.frame object.
- **...**: Additional arguments.
- **group.by**: An optional character vector specifying the group by column names within argument data.
- **order**: An optional character string specifying an ordering for the cross-tabulation; one of "NAME" (ascending by name), "-NAME" (descending by name), "DATA" (ascending by data), "-DATA" (descending by data), "FREQ" (ascending by frequency), "-FREQ" (descending by frequency), "INTERNAL".
- **weights**: An optional character string specifying a numeric column within argument data to use as analytic weights.
- **where**: An optional character vector specifying arbitrary partitions of argument data.
- **strata**: An optional character string specifying a column within argument data to use as stratification variable.

Value

When argument where is not specified, returns an ore.frame object.

When argument where is specified, returns a list of ore.frame objects.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.freq, xtabs
Examples

```r
IRIS <- ore.push(iris)

table(iris$Species, iris$Petal.Length)
ore.crosstab(Species ~ Petal.Length, data = IRIS)

# 1 way table
ore.crosstab(~ Species, data = IRIS)

# 2 2-way tables
ore.crosstab(Species ~ Petal.Length + Sepal.Length, data = IRIS)

# Order rows of cross-tabulation by asc frequency counts
ore.crosstab(Species ~ Petal.Length | FREQ, data = IRIS)

# Order by descending frequency counts
ore.crosstab(Species ~ Petal.Length | -FREQ, data = IRIS)

# As many cross-tabs as distinct values in Species
ore.crosstab(Petal.Length ~ Sepal.Length / Species, data = IRIS)

# Syntactic simplification
ore.crosstab(Sepal.Length ~ Petal.Width ~ Species, data = IRIS)

# Illustration of the use of group.by
ore.crosstab(Species ~ Petal.Length, data = IRIS)

# Compare with the following where Petal.Length values are hidden
ore.crosstab(~ Species, group.by = "Petal.Length", data = IRIS)

# Use of derived columns
IRIS$PetalBins <- ifelse(IRIS$Petal.Length < 2, "SMALL",
ifelse(IRIS$Petal.Length < 5, "MEDIUM", "LARGE"))
ore.crosstab(Species ~ PetalBins, data = IRIS)
```

ore.datastore

**Oracle R Enterprise Datastore Listing Function**

Description

Lists existing datastores in the user’s Oracle Database schema.

Usage

```r
ore.datastore(name, pattern, type = c("user", "private", "all",
  "grantable", "grant", "granted"))
```

Arguments

- **name**: An optional character string specifying the datastore name; cannot be used with argument `pattern`.
- **pattern**: An optional regular expression character string specifying the matching datastore names; cannot be used with argument `name`. 
type

An optional scalar character string specifying the type of datastore to list. The valid values are 'user' (default), 'private', 'all', 'grantable', 'grant', or 'granted'. 'user' lists the datastores created by current session user. 'private' lists these datastores the read privileges for which cannot be granted by the current session user to other users. 'all' lists all the datastores to which the current session user has read access. 'grantable' lists these datastores the read privilege for which can be granted by the current session user to other users. 'grant' lists these datastores the read privilege for which has been granted by the current session user to other users. 'granted' lists these datastores the read privilege for which has been granted by other users to the current session user.

Details

Function ore.datastore lists high-level information about datastores in the user’s Oracle Database schema. This datastore listing may be optionally filtered by either the name or pattern argument.

Value

A data.frame object is returned with columns datastore.name, object.count, size, creation.date, and description when argument type is 'user', 'private', or 'grantable'; these columns specify the datastore name, the number of objects in the named datastore, the size of the datastore in bytes, the datastore creation date, and the datastore comment, respectively. The output data.frame has an extra column owner specifying the owner of the datastores when argument type is 'all' or 'granted'. When argument type is 'grant', the output data.frame contains columns datastore.name and grantee specifying the datastore name and the name of the user to which the read privilege of the named datastore has been granted by the current session user. The rows of the data.frame object are alphabetically sorted according to column owner, if present, and datastore.name.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.save, ore.load, ore.move, ore.delete, ore.datastoreSummary, ore.grant, ore.revoke

Examples

if (!interactive())
{
  if (any(sapply(c("x", "y", "z"), exists)))
    stop("object x, y, or z exists")

  x <- stats::runif(20)
  y <- list(a = 1, b = TRUE, c = "oops")
  z <- ore.push(x)
ore.save(list=ls(), name="rqds_1")
ore.save(x, y, name="rqds_2")
ore.save(z, name="rqds_3", grantable=TRUE)

# list overall information about the datastores in user's schema
ds <- ore.datastore()
## Not run:
ds
## End(Not run)

# list overall information about the datastores available in the user's schema
ds <- ore.datastore(type="all")
## Not run:
ds
## End(Not run)

# list overall information about the datastore the read privilege for which
# can be granted
ds <- ore.datastore(type="grantable")
## Not run:
ds
## End(Not run)

# list overall information about the datastore with name 'rqds_1'
ds <- ore.datastore(name="rqds_1")
## Not run:
ds
## End(Not run)

# list overall information about the datastores whose name starts
# with 'rqds_'
ds <- ore.datastore(pattern="^rqds_")
## Not run:
ds
## End(Not run)

sapply(c("rqds_1", "rqds_2", "rqds_3"), ore.delete)
rm(x, y, z)
}

ore.datastoreSummary

*Oracle R Enterprise Datastore Summarizing Function*
Description

Summarizes the contents of a datastore in the user’s Oracle Database schema.

Usage

ore.datastoreSummary(name, owner)

Arguments

name

The character string specifying the datastore name.

owner

An optional character string specifying the owner of the datastore to summarize. If the owner of the datastore is not the current user, the read privilege for the datastore must be granted to the current user by `ore.grant`; otherwise `ore.load` errors out. If the `owner` argument not specified, the default value is the current user.

Details

Summarizes the datastore specified by argument `name` and `owner`. If the named datastore does not exist or the current user does not have read privilege for this datastore, the function throws an error.

Value

A `data.frame` object is returned with columns `object.name`, `class`, `size`, `length`, `row.count`, and `col.count`; these columns specify the object name, the object’s class, the object’s size in bytes, the length of object, and the number of rows and columns of object, respectively. The rows of the `data.frame` object are alphabetically sorted by column `object.name`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore.save`, `ore.load`, `ore.move`, `ore.delete`, `ore.datastore`, `ore.grant`, `ore.revoke`

Examples

```r
## Not run:
if (any(sapply(c("x", "y", "z"), exists)))
  stop("object x, y, or z exists")

x <- stats::runif(20)
y <- list(a = 1, b = TRUE, c = "oops")
z <- ore.push(x)

ore.save(list=ls(), name="rq$ds_1")

# show detailed information about the datastore rq$ds_1
```
ore.datetime-class

ore.datastoreSummary(name="rq$ds_1")

ore.delete(name="rq$ds_1")
rm(x, y, z)

## End(Not run)

ore.datetime-class  Class ore.datetime

Description

The ore.date and ore.datetime, and ore.difftime classes represent DATE, TIMESTAMP, and INTERVAL DAY TO SECOND data columns in Oracle R Enterprise.

Details

ore.datetime objects, unlike POSIXt objects, do not support a time zone setting. Instead all ore.datetime use the system time zone, Sys.timezone(), if is available or GMT if this information is not available.

Date/Time Arithmetic

The following arithmetic operations are supported for date/time data types:

<table>
<thead>
<tr>
<th>Arith Op</th>
<th>Signature</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;+&quot;</td>
<td>ore.date, ore.difftime</td>
<td>ore.date</td>
</tr>
<tr>
<td>&quot;+&quot;</td>
<td>ore.difftime, ore.date</td>
<td>ore.date</td>
</tr>
<tr>
<td>&quot;+&quot;</td>
<td>ore.date, ore.number</td>
<td>ore.date</td>
</tr>
<tr>
<td>&quot;+&quot;</td>
<td>ore.number, ore.date</td>
<td>ore.date</td>
</tr>
<tr>
<td>&quot;+&quot;</td>
<td>ore.datetime, ore.difftime</td>
<td>ore.datetime</td>
</tr>
<tr>
<td>&quot;-&quot;</td>
<td>ore.date, ore.date</td>
<td>ore.difftime</td>
</tr>
<tr>
<td>&quot;-&quot;</td>
<td>ore.date, ore.difftime</td>
<td>ore.difftime</td>
</tr>
<tr>
<td>&quot;-&quot;</td>
<td>ore.datetime, ore.date</td>
<td>ore.difftime</td>
</tr>
<tr>
<td>&quot;-&quot;</td>
<td>ore.difftime, ore.number</td>
<td>ore.date</td>
</tr>
<tr>
<td>&quot;-&quot;</td>
<td>ore.difftime, ore.difftime</td>
<td>ore.difftime</td>
</tr>
<tr>
<td>&quot;-&quot;</td>
<td>ore.datetime, ore.difftime</td>
<td>ore.difftime</td>
</tr>
<tr>
<td>&quot;*&quot;</td>
<td>ore.difftime, ore.difftime</td>
<td>ore.difftime</td>
</tr>
<tr>
<td>&quot;*&quot;</td>
<td>ore.difftime, ore.number</td>
<td>ore.difftime</td>
</tr>
<tr>
<td>&quot;*&quot;</td>
<td>ore.difftime, ore.difftime</td>
<td>ore.difftime</td>
</tr>
<tr>
<td>&quot;/&quot;</td>
<td>ore.difftime, ore.difftime</td>
<td>ore.difftime</td>
</tr>
</tbody>
</table>

In addition to binary arithmetic operators, date/time data support:

diff(x, lag = 1, differences = 1): Returns an ore.difftime object containing a suit-
ably lagged and iterated differences of argument \( x \).

- **lag**: An integer indicating which lag to use.
- **differences**: An integer indicating the order of the difference.

\[
\text{trunc}(x, \text{units} = \text{"DD"}, \ldots): \text{Returns a ore.datetime object containing a truncated version of argument } x.
\]

- **x**: An ore.datetime object.
- **units**: A character string specifying the level of truncation, one of

<table>
<thead>
<tr>
<th>units</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CC, SCC</td>
<td>Century</td>
</tr>
<tr>
<td>D</td>
<td>Day of week (1-7).</td>
</tr>
<tr>
<td>DAY</td>
<td>Name of day.</td>
</tr>
<tr>
<td>DD</td>
<td>Day of month (1-31).</td>
</tr>
<tr>
<td>DDD</td>
<td>Day of year (1-366).</td>
</tr>
<tr>
<td>DY</td>
<td>Abbreviated name of day.</td>
</tr>
<tr>
<td>HH, HH12</td>
<td>Hour of day (1-12).</td>
</tr>
<tr>
<td>HH24</td>
<td>Hour of day (0-23).</td>
</tr>
<tr>
<td>IW</td>
<td>Week of year (1-52 or 1-53) based on the ISO standard.</td>
</tr>
<tr>
<td>IYY, IY, I</td>
<td>Last 3, 2, or 1 digit(s) of ISO year.</td>
</tr>
<tr>
<td>IYYYY</td>
<td>4-digit year based on the ISO standard.</td>
</tr>
<tr>
<td>J</td>
<td>Julian day; the number of days since January 1, 4712 BC.</td>
</tr>
<tr>
<td>MI</td>
<td>Minute (0-59).</td>
</tr>
<tr>
<td>MM</td>
<td>Month (01-12; January = 01).</td>
</tr>
<tr>
<td>MON</td>
<td>Abbreviated name of month.</td>
</tr>
<tr>
<td>MONTH</td>
<td>Name of month.</td>
</tr>
<tr>
<td>Q</td>
<td>Quarter of year (1, 2, 3, 4; January - March = 1).</td>
</tr>
<tr>
<td>RM</td>
<td>Roman numeral month (I-XII; January = 1).</td>
</tr>
<tr>
<td>WW</td>
<td>Week of year (1-53) where week 1 starts on the first day of the year and continues to the seventh day of the year.</td>
</tr>
<tr>
<td>W</td>
<td>Week of month (1-5) where week 1 starts on the first day of the month and ends on the seventh.</td>
</tr>
<tr>
<td>Y, YYYY</td>
<td>Year with comma in this position.</td>
</tr>
<tr>
<td>YEAR, SYEAR</td>
<td>Year, spelled out; S prefixes BC dates with a minus sign (-).</td>
</tr>
<tr>
<td>YYYY, SYYYY</td>
<td>4-digit year; S prefixes BC dates with a minus sign.</td>
</tr>
<tr>
<td>YYY, YY, Y</td>
<td>Last 3, 2, or 1 digit(s) of year.</td>
</tr>
</tbody>
</table>

**Character Coersion**

\[
\text{as.character}(x, \text{format} = \text{"YYYY-MM-DD"}, \ldots) \text{ and as.ore.character}(x, \text{format} = \text{"YYYY-MM-DD"}, \ldots) \text{ for ore.date; as.character}(x, \text{format} = \text{"YYYY-MM-DD HH24:MI:SS"}, \ldots) \text{ and as.ore.character}(x, \text{format} = \text{"YYYY-MM-DD HH24:MI:SS"}, \ldots) \text{ for ore.datetime:}
\]

- Returns an ore.character object containing information from argument \( x \) in a form specified by the format argument.

\[
\begin{align*}
\text{format} & \quad \text{Description} \\
-./...;.."text" & \quad \text{Punctuation and quoted text is reproduced in the result.} \\
A.D, A.D. & \quad \text{AD indicator with or without periods.} \\
A.M, A.M. & \quad \text{Meridian indicator with or without periods.} \\
B.C, B.C. & \quad \text{BC indicator with or without periods.} \\
CC, SCC & \quad \text{Century.} \\
D & \quad \text{Day of week (1-7).} \\
DAY & \quad \text{Name of day.} \\
DD & \quad \text{Day of month (1-31).}
\end{align*}
\]
DDD Day of year (1-366).
DL Returns a value in the long date format.
DS Returns a value in the short date format.
DY Abbreviated name of day.
E Abbreviated era name.
EE Full era name.
FF [1..9] Fractional seconds.
FM Returns a value with no leading or trailing blanks.
FX Requires exact matching between the character data and the format model.
HH, HH12 Hour of day (1-12).
HH24 Hour of day (0-23).
IW Week of year (1-52 or 1-53) based on the ISO standard.
IYY, IY, I Last 3, 2, or 1 digit(s) of ISO year.
IYYY 4-digit year based on the ISO standard.
J Julian day; the number of days since January 1, 4712 BC.
MI Minute (0-59).
MM Month (01-12; January = 01).
MON Abbreviated name of month.
MONTH Name of month.
PM, P.M. Meridian indicator with or without periods.
Q Quarter of year (1, 2, 3, 4; January - March = 1).
RM Roman numeral month (I-XII; January = I).
RR Lets you store 20th century dates in the 21st century using only two digits.
RRRR Round year.
SS Second (0-59).
SSSS Seconds past midnight (0-86399).
TS Returns a value in the short time format.
TZD Daylight saving information.
TZH Time zone hour.
TZM Time zone minute.
TZR Time zone region information.
WW Week of year (1-53) where week 1 starts on the first day of the year and continues to the seventh.
W Week of month (1-5) where week 1 starts on the first day of the month and ends on the seventh.
X Local radix character.
Y, YYYY Year with comma in this position.
YEAR, SYEAR Year, spelled out; § prefixes BC dates with a minus sign (-).
YYYY, SYYYY 4-digit year; § prefixes BC dates with a minus sign.
YYY, YY, Y Last 3, 2, or 1 digit(s) of year.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.year, ore.character, ore.factor, ore.frame, ore.logical, ore.matrix, ore.number, ore.vector.
Examples

```r
DATE <- ore.push(seq(as.Date("1999/1/1"), as.Date("2002/1/1"), by="3 months"))
DATE
as.character(DATE, format = "YYYY/MM/DD")
```

**ore.delete**  
*Oracle R Enterprise Datastore Removal Function*

**Description**

Deletes one or more datastores from the user’s Oracle Database schema or specific R objects from within a datastore.

**Usage**

```r
ore.delete(name, list = character(0), pattern)
```

**Arguments**

- `name`  
  A character string specifying the name of the datastore to delete or modify; cannot be used with argument `pattern`.

- `list`  
  An optional character vector containing the names of the objects to delete from the datastore specified by `name`. If this argument is not specified, then the entire datastore is deleted.

- `pattern`  
  Optional. Cannot be used with either argument `name` or `list`. Can be either a single unnamed regular expression character string or a named vector of them. If it is a single regular expression character string, then the argument specifies the names of datastores to delete. If it is a vector of regular expression character strings, then each element of the vector specifies the names of objects to delete from the datastore specified by the name of the element. If an element is `NA`, then the entire datastore is deleted. If the name of an element is `NA`, then an error is thrown.

**Details**

- If argument `list` is missing, and argument `name` is used, then function `ore.delete` deletes the datastore specified in the `name` argument from the user’s Oracle Database schema.
- If arguments `name` and `list` are used, then function `ore.delete` deletes the specified R objects from the datastore supplied in the `name` argument.
- If argument `pattern` is a single character string, then function `ore.delete` deletes the datastores in the user’s Oracle Database schema whose names match the regular expression specified in the `pattern` argument.
- If argument `pattern` is a named vector of character strings, then for each element in the list, function `ore.delete` deletes from the datastore specified by the name of the element those R objects whose names match the regular expression of the element.

**Author(s)**

Oracle <oracle-r-enterprise@oracle.com>
ore.doEval

References

Oracle R Enterprise

See Also

ore.save, ore.load, ore.move, ore.datastore, ore.datastoreSummary

Examples

if (any(sapply(c("rqobj1", "rqobj2", "y", "z"), exists)))
  stop("object rqobj1, rqobj2, y, or z exists")

rqobj1 <- matrix(c(TRUE, FALSE, TRUE, TRUE, TRUE, FALSE), nrow = 2, ncol = 3)
raqobj2 <- stats::runif(20)
y <- list(a = 1, b = TRUE, c = "oops")
z <- ore.push(rqobj2)

ore.save(rqobj1, rqobj2, y, z, name="rq$ds_1")
ore.save(y, name="rq$ds_2a")
ore.save(z, name="rq$ds_2b")
ore.save(rqobj1, rqobj2, y, name="rq$ds_3")
ore.save(y, z, name="rq$ds_4")

# delete R object y, z from datastore 'rq$ds_1' from user's schema
ore.delete(name="rq$ds_1", list=c("y", "z"))
# delete the datastore with name 'rq$ds_1' from user's schema
ore.delete(name="rq$ds_1")
# delete datastores with names starting with 'rq$ds_2' from user's schema
sort(ore.delete(pattern="^rq$ds_2"))

# delete R objects whose names contain 'rqobj' from datastore 'rq$ds_3' from user's schema
ore.delete(pattern=c("rq$ds_3"="rqobj"))
# delete the contents of datastore 'rq$ds_4' from user's schema, but not the datastore itself
ore.delete(pattern=c("rq$ds_4"=".*"))
# delete datastores 'rq$ds_3' and 'rq$ds_4' from user's schema
ore.delete(pattern=c("rq$ds_3"=NA, "rq$ds_4"=NA))
# delete all datastores
## Not run:
ore.delete(pattern=".*")
## End(Not run)
rm(rqobj1, rqobj2, y, z)

ore.doEval

Oracle R Enterprise Embedded R Script Execution Functions

Description

Runs a function within the Oracle database under various conditions.
Usage

ore.doEval(FUN, ..., FUN.VALUE = NULL, FUN.NAME = NULL, FUN.OWNER = NULL)
ore.groupApply(X, INDEX, FUN, ..., FUN.VALUE = NULL,
  FUN.NAME = NULL, FUN.OWNER = NULL,
  parallel = getOption("ore.parallel", NULL))
ore.indexApply(times, FUN, ..., FUN.VALUE = NULL,
  FUN.NAME = NULL, FUN.OWNER = NULL,
  parallel = getOption("ore.parallel", NULL))
ore.rowApply(X, FUN, ..., FUN.VALUE = NULL,
  FUN.NAME = NULL, FUN.OWNER = NULL, rows = 1,
  parallel = getOption("ore.parallel", NULL))
ore.tableApply(X, FUN, ..., FUN.VALUE = NULL,
  FUN.NAME = NULL, FUN.OWNER = NULL)

Arguments

**X**

An ore.frame object.

**INDEX**

A ore.vector or ore.frame object containing ore.factor objects or columns, each of which is the same length as argument X. It is used to partition the data in X before sending it to function FUN. The counterpart supported R types are logical, integer, numeric, character, factor.

**times**

The number of times to execute the function.

**FUN**

The function to be applied. For functions ore.groupApply, ore.rowApply, and ore.tableApply the first argument to the FUN argument must represent a data.frame object. For function ore.indexApply, the first argument to FUN must represent the index number. For function ore.doEval, no arguments are required for FUN. The function specified by FUN cannot recursively call embedded R APIs. Cannot be used with argument FUN.NAME.

**...**

Additional arguments to FUN.

Arguments that start with ore. are special control arguments. They are not passed to the function specified by FUN or FUN.NAME arguments, but instead control what happens before or after the execution of the closure. The following control arguments are supported:

1. ore.drop - controls the object type for the input data. If TRUE, a one column data.frame will be converted to a vector. The default value is TRUE.
2. ore.na.omit - controls the handling of missing values in the input data. If TRUE, rows or vector elements, depending on the ore.drop setting, containing missing values will be removed from the input data. If all the rows in a chunk contain missing values, the input data for that chunk will be an empty data.frame or vector. The default value is FALSE.
3. ore.connect - controls whether to automatically connect to Oracle R Enterprise inside the closure. This is equivalent to doing an ore.connect call with the same credentials as the client session. The default value is FALSE.
4. ore.graphics - controls whether to start a graphical driver and look for images. The default value is TRUE.
5. ore.png.* - if ore.graphics is TRUE, additional parameters for the png graphics device driver. The naming convention for these arguments is to add an ore.png. prefix to the arguments of the png function.
example, if ore.png.height is supplied, argument height will be passed to the png function. If not set, the standard default values for the png function are used.

6. ore.envAsEmptyenv - controls whether referenced environments in an object should be replaced with an empty environment during serialization. Some types of input parameters and returned objects, such as list, formula, are serialized before being saved to the database. If TRUE, the referenced environment in the object will be replaced with an empty environment whose parent is .GlobalEnv, and therefore, the objects in the original referenced environment will not be serialized. In some situations, this could significantly reduce the size of serialized objects. If FALSE, all the objects in the referenced environment will be serialized, and could be unserialized and recovered later. The default value is regulated by the global option ore.envAsEmptyenv.

7. ore.characterAsFactor - controls the type that character and factor columns of the in-database data set referred to by X are treated as. If TRUE, all character and factor columns are treated as factor type. If FALSE, all character and factor columns are treated as character type. For functions ore.groupApply and ore.rowApply, each partition will be only aware of factor levels in itself and will not be aware of levels in other partitions. The default value is FALSE.

FUN.VALUE A data.frame or ore.frame to use as a template for the return value.

The attribute ora.type can be applied to a data.frame column to specify that the corresponding output column of a ore.frame uses a CLOB or BLOB type.

FUN.NAME A character string specifying the name of a serialized R script, which contains a single R function definition, within the Oracle R Enterprise in-database R script archive. Cannot be used with FUN.

Oracle R Enterprise comes with a number of predefined graphical scripts. All predefined scripts have a reserved name that start with RQG$ followed by a function name from the graphics package that the script wraps. Depending on the function it either takes the first, the first and second or all columns of the input data.frame. Thus, predefined scripts can only be used with ore.tableApply, ore.groupApply, or ore.rowApply. Each function also has a ... so that it can pass any parameter to the function that it wraps. Here is a list of predefined graphical scripts:

1. RQG$plot1d - a wrapper for plot. Works on the first column of the input data.frame object.
2. RQG$plot2d - a wrapper for plot. Works on the first two columns of the input data.frame object.
3. RQG$hist - a wrapper for hist. Works on the first column of the input data.frame object.
4. RQG$boxplot - a wrapper for boxplot. Works on the first column of the input data.frame object.
5. RQG$smoothScatter - a wrapper for smoothScatter. Works on the first two columns of the input data.frame object.
6. RQG$cdplot - a wrapper for cdplot. Works on the first two columns of the input data.frame object.
7. RQG$pairs - a wrapper for pairs. Works on all columns of the input data.frame object.
8. RQ$matplot - a wrapper for matplot. Works on all columns of the input data.frame object.

Oracle R Enterprise also comes with a number of predefined R and package version scripts. These scripts start with RQS followed by an R function name that the script wraps and can only be used with ore.doEval. Here is a list of these predefined scripts:

2. RQ$getRversion - a wrapper for getRversion. Takes no argument and returns R version number.
3. RQ$installed.packages - a wrapper for installed.packages. Takes no argument and returns package name, version number, and package installation location of installed packages.

FUN.OWNER
An optional character string specifying the owner of the FUN.NAME R script. The user that creates an R script with ore.scriptCreate is the owner of that script. The RQSYS schema is the owner of the global and pre-defined R scripts. When FUN.OWNER is not specified or is NULL, then Oracle R Enterprise looks for the owner in the following order: user of the current session, RQSYS. Argument FUN.OWNER is only used with argument FUN.NAME.

rows
The maximum number of rows in each chunk.

parallel
A preferred degree of parallelism to use in the embedded R job; either a positive integer greater than or equal to 2 for a specific degree of parallelism, a value of FALSE or 1 for no parallelism, a value of TRUE for the data argument’s default parallelism, or NULL for the database default for the operation. The default value is regulated by the global option ore.parallel.

Details
Function ore.doEval executes a function, either FUN or FUN.NAME, within an R process running inside the Oracle database.

Function ore.groupApply partitions an in-database data set by a (potentially derived) column and executes a function on those partitions within R processes running inside the Oracle database. Each partition must fit wholly within a single R process.

Function ore.indexApply executes a function index number of times inside the Oracle database.

Function ore.rowApply partitions an in-database data set into row chunks and executes a function on those partitions within R processes running inside the Oracle database. Each partition must fit wholly within a single R process.

Function ore.tableApply executes a function on an in-database data set.

Either argument FUN or FUN.NAME must be supplied. For security reasons, use of argument FUN requires ‘RQADMIN’ Oracle database privileges. Because creation of the R script represented by argument FUN.NAME has to be published by someone with ‘RQADMIN’ credentials, it can be used by anyone authorized to use Oracle R Enterprise.

Argument FUN.OWNER can be used with argument FUN.NAME to uniquely specify an R function defined in the R script repository.

The parallel argument regulates the use of a ‘/*+parallel */’, ‘/*+parallel(DOP) */’, or a ‘/*+no_parallel */’ hint being added to the underlying SQL query. Consult Oracle database documentation for more information.
The function to be applied specified via argument FUN or FUN.NAME is automatically connected to Oracle R Enterprise with the same credentials as the client session invoking it. Only an equivalent of ore.connect is invoked. Functions such as ore.sync, ore.attach, and ore.get should be called explicitly.

**Value**

If argument FUN.VALUE is supplied, an ore.frame object that conforms to the FUN.VALUE template is returned.

If argument FUN.VALUE is not supplied, then functions ore.doEval and ore.tableApply return an ore.object while functions ore.groupApply, ore.indexApply, and ore.rowApply return an ore.list.

**Author(s)**

Oracle <oracle-r-enterprise@oracle.com>

**References**

Oracle R Enterprise

**See Also**

ore.scriptCreate, ore.options

**Examples**

```r
## ore.doEval
eval1 <- ore.doEval(function() "Hello, world")
eval2 <-
  ore.doEval(function()
    data.frame(x = "Hello, world", stringsAsFactors = FALSE))
eval3 <-
  ore.doEval(function()
    data.frame(x = "Hello, world", stringsAsFactors = FALSE),
    FUN.VALUE =
    data.frame(x = character(), stringsAsFactors = FALSE))
out.df <- data.frame(x = character(), y = raw(), stringsAsFactors = FALSE)
attr(out.df$x, "ora.type") <- "clob"
attr(out.df$y, "ora.type") <- "blob"
eval4 <-
  ore.doEval(function()
    res <- data.frame(x = "Hello, world", stringsAsFactors = FALSE)
    res$y[[1L]] <- charToRaw("Hello, world")
res),
    FUN.VALUE = out.df)
eval1
class(eval1) # ore.object
eval2
class(eval2) # ore.object
eval3
class(eval3) # ore.frame
eval4$x
rawToChar(ore.pull(eval4$y))
```

## copy data to the database
IRIS <- ore.push(iris)

## ore.groupApply
grpAp1 <-
  ore.groupApply(IRIS, IRIS$Species,
  function(df)
    if(nrow(df) == 0)
      NULL
    else
      summary(lm(Sepal.Length ~ ., data = df[1:4]),
        parallel = TRUE)
  grpAp2 <-
  ore.groupApply(IRIS, IRIS$Species,
  function(df) {
    if (nrow(df) == 0) {
      species <- character()
      cf <- numeric()
      names(cf) <- character()
    } else {
      species <- as.character(df$Species[1])
      cf <- coef(lm(Sepal.Length ~ .,
        data = df[1:4]))
    }
    data.frame(Species = species,
      CoefName = names(cf),
      CoefValue = unname(cf),
      stringsAsFactors = FALSE)
  },
  FUN.VALUE =
    data.frame(Species = character(),
      CoefName = character(),
      CoefValue = numeric(),
      stringsAsFactors = FALSE),
  parallel = TRUE)

class(grpAp1) # ore.list
class(grpAp2) # ore.frame

## ore.indexApply
ore.indexApply(5, function(i) i)
if (interactive())
  ore.indexApply(5, function(i) summary(rnorm(100)), parallel = TRUE)

## ore.rowApply
# create a classification tree for iris data
library(rpart)
irisRpart <- rpart(Species ~ ., data = iris)

irisPred <-
  ore.rowApply(IRIS,
  function(df, model) {
    library(rpart)
    cbind(df, PRED = predict(model, df, type = "class"))
  }, model = irisRpart,
  FUN.VALUE =
    cbind(iris[integer()],], PRED = character()),
  rows = 50, parallel = TRUE)
## ore.tableApply

```r
ore.tableApply(IRIS, function(df) summary(df))
```

---

**ore.esm**

*Oracle R Enterprise Time Series Exponential Smoothing Models*

### Description

Creates exponential smoothing models on ordered `ore.vector` data.

### Usage

```r
ore.esm(x,  
  interval = NULL,  
  model = "simple",  
  accumulate = "NONE",  
  setmissing = "PREV",  
  optim.start = c(alpha=0.3, beta=0.1),  
  optim.control = list())
```

```r
## S3 method for class 'ore.esm'
fitted(object, start = NULL, end = NULL, ...)  
## S3 method for class 'ore.esm'
predict(object, n.ahead = 12L, ...)  
## S3 method for class 'ore.esm'
forecast.ore.esm(object, h = 12L, ...)
```

### Arguments

- `x` An ordered `ore.vector` of time series data or transactional data. The ordering column could be either integers from 1 to the length of the time series or of type `ore.datetime`.
- `interval` The interval of the time series, or the time interval by which the transactional data is to be accumulated. If the ordering column of the argument `x` is of type `ore.datetime`, `interval` must be specified.
  - Possible values: "YEAR", "QTR", "MONTH", "WEEK", "DAY", "HOUR", "MINUTE", "SECOND"
- `model` The exponential smoothing model name. Possible values: "simple", "double"
- `accumulate` The method of accumulation.
  - Possible values:
    - "NONE" No accumulation occurs. In this case, the argument `x` is required to be equally spaced time series observations.
    - "TOTAL" Accumulation based on the sum of the observed values.
    - "AVERAGE" Accumulation based on the average of the observed values. The value could be abbreviated to "AVG".
    - "MINIMUM" Accumulation based on the minimum of the observed values. The value could be abbreviated to "MIN".
    - "MAXIMUM" Accumulation based on the maximum of the observed values. The value could be abbreviated to "MAX".
    - "MEDIAN" Accumulation based on the median of the observed values. The value could be abbreviated to "MED".
    - "STDDEV" Accumulation based on the standard deviation of the observed values. The value could be abbreviated to "STD".
    - "N" Accumulation based on the number of nonmissing observations.
    - "NOBS" Accumulation based on the number of observations.
    - "NMISS" Accumulation based on the number of missing observations.
setmissing  The method of treating missing values.
Possible values:

"AVERAGE"  Missing values are set to the average of the accumulated values. The value could be abbreviated to "AVG".
"MINIMUM"  Missing values are set to the minimum of the accumulated values. The value could be abbreviated to "MIN".
"MAXIMUM"  Missing values are set to the maximum of the accumulated values. The value could be abbreviated to "MAX".
"MEDIAN"  Missing values are set to the median of the accumulated values. The value could be abbreviated to "MED".
"FIRST"  Missing values are set to the first accumulated nonmissing value.
"LAST"  Missing values are set to the last accumulated nonmissing value.
"PREVIOUS"  Missing values are set to the previous accumulated nonmissing value. The value could be abbreviated to "PREV".
"NEXT"  Missing values are set to the next accumulated nonmissing value.

optim.start  A vector with named components alpha and beta containing the starting values for the optimizer. The starting values should be in the range of 0 to 1. Ignored in the simple model case.

optim.control  Optional list with additional control parameters passed to optim in the double model case. Ignored in the simple model case.

object  An object of type ore.esm.

start  A positive integer that specifies the beginning index of the output of the fitted values. If a value is specified, it should be less than or equal to the argument end. The default is NULL, which indicates start=1.

end  A positive integer that specifies the ending index of the output of the fitted values. If a value is specified, it should be greater than or equal to the argument start. The default is NULL, which indicates the value is equal to the length of the training dataset.

n.ahead  The number of time periods to forecast.

h  The number of time periods to forecast.

...  Additional arguments.

Details

The function ore.esm implements exponential smoothing models for in-database time series observations. The function can work with either time series data, whose observations are evenly spaced by a fixed interval, or transactional data, whose observations are not equally spaced. For transactional data, specify an aggregation method with the argument accumulate and a time interval with the argument interval. To handle missing values, specify a value with the argument setmissing. The function preprocesses the data using the specified aggregation method, time intervals, and handling of missing values.

The fitted method provides the fitted times series values aligned with the training dataset. The arguments start and end specify the index range of the output fitted values. This function currently does not support a model built with an accumulation method specified.

The predict method predicts the time series by using the exponential smoothing model built by ore.esm. If the forecast package is loaded, the forecast method can be called through generic.

Value

For ore.esm, returns an object of class "ore.esm". Some of its components are as follows:
ore.esm

smoothing.param
The estimated smoothing parameters.

model
The model type.

model.param
The input arguments.

For fitted.ore.esm, returns an ore.vector of fitted values of the model in the range that is specified by the arguments start and end.

For predict.ore.esm or forecast.ore.esm, returns a data.frame of the predicted time series.

Author(s)
Oracle <oracle-r-enterprise@oracle.com>

References
Exponential Smoothing method http://en.wikipedia.org/wiki/Exponential_smoothing

Oracle R Enterprise

Examples

# case 1
N <- 5000
ts0 <- ore.push(data.frame(ID=1:N, VAL=seq(1,5,length.out=N)^2+rnorm(N,sd=0.5)))
rownames(ts0) <- ts0$ID
x <- ts0$VAL
esm.mod <- ore.esm(x, model = "double")
esm.predict <- predict(esm.mod, 30)
esm.fitted <- fitted(esm.mod, start=4000, end=5000)
plot(ts0[4000:5000,], pch='.
lines(ts0[4000:5000, 1], esm.fitted, col="blue")
lines(esm.predict, col="red", lwd=2)

# case 2
ts01 <- data.frame(ID=seq(as.POSIXct("2008/6/13"), as.POSIXct("2011/6/16"),
length.out=4000), VAL=rnorm(4000, 10))
ts02 <- data.frame(ID=seq(as.POSIXct("2011/7/19"), as.POSIXct("2012/11/20"),
length.out=1500), VAL=rnorm(1500, 10))
ts03 <- data.frame(ID=seq(as.POSIXct("2012/12/09"), as.POSIXct("2013/9/25"),
length.out=1000), VAL=rnorm(1000, 10))
ts1 = ore.push(rbind(ts01, ts02, ts03))
rownames(ts1) <- ts1$ID
x <- ts1$VAL
esm.mod <- ore.esm(x, "DAY", accumulate = "AVG", model="simple",
setmissing="PREV")
esm.predict <- predict(esm.mod)

# case 3
x <- ore.push(BJsales)
esm.mod <- ore.esm(x, model="double")
esm.predict <- predict(esm.mod)
esm.fitted <- fitted(esm.mod)
library(OREgraphics)
plot(x)
lines(esm.fitted, col="blue")
lines(esm.predict, col="red", lwd=4)

---

ore.exec **SQL Query Execution Function**

Description

Executes the specified SQL query in the user’s Oracle Database schema.

Usage

```r
ore.exec(qry)
```

Arguments

- **qry**: A character string specifying the SQL statement.

Details

The `ore.exec` function is intended for use with data definition language (DDL) type statements, such as CREATE TABLE, that have no return value.

If the `ore.exec` function call creates a table in the user’s Oracle Database schema, a corresponding `ore.frame` object is not created unless the user invokes function `ore.sync`. To avoid refreshing all `ore.frame` objects, the name of the table can be explicitly provided.

Value

Returns an invisible `NULL` value.

Author(s)

- Oracle `<oracle-r-enterprise@oracle.com>`

References

- Oracle R Enterprise

See Also

- `ore.ls`, `ore.sync`, `dbGetQuery`
Examples

```r
if (!interactive())
{
  ore.exec("create table TEST_CREATE as select * from dual")
  ore.exists("TEST_CREATE")  # table not found
  ore.sync(table="TEST_CREATE")  # sync metadata of new table only
  ore.exists("TEST_CREATE")  # table TEST_CREATE is present
  ore.drop(table="TEST_CREATE")  # clean up
  ore.exists("TEST_CREATE")
}
```

ore.exists  

*Oracle R Enterprise Object Existence Checking Function*

**Description**

Determines whether a named `ore.frame` object, representing a database table or view, exists within the R environment for a schema in the Oracle R Enterprise session.

**Usage**

```r
ore.exists(name, schema)
```

**Arguments**

- `name`  
  A character string specifying the name of the `ore.frame` object.

- `schema`  
  A character string specifying the database schema name.

**Details**

If argument `schema` is unspecified, the default schema - the one specified at connection time for the Oracle R Enterprise session - is used.

**Value**

A logical value indicating whether an `ore.frame` object of the given name is found within the specified schema.

**Author(s)**

Oracle <oracle-r-enterprise@oracle.com>

**References**

*Oracle R Enterprise*

**See Also**

`ore.attach, ore.connect, ore.get, ore.ls, ore.rm, ore.sync`
ore.factor-class

Examples

    if (!interactive())
    {
        ore.exists("IRIS_TABLE") # FALSE
        ore.create(iris, table = "IRIS_TABLE")
        ore.exists("IRIS_TABLE") # TRUE
        ore.exists("IRIS_TABLE", "RQUSER") # TRUE
        ore.drop(table = "IRIS_TABLE")
    }

Description

The ore.factor class represents factor data columns in Oracle R Enterprise.

Details

The levels of ore.factor objects are not managed as a lookup table the same way that factor objects are. Instead the levels for ore.factor objects are equivalent to `sort(unique(as.character(x)))`.

Factor Data Methods

- `interaction(..., sep = ".")`: Returns an ore.factor object containing the combination of the values in arguments ... .
- `levels(x)`: Returns a character vector containing the factor levels.
- `nlevels(x)`: Returns an integer value containing the number of factor levels.
- `summary(object, maxsum = 100, ...)`: Returns a tabular summary of the ore.factor object.

Note

See the corresponding R documentation for the functions listed above.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore, ore.datetime, ore.character, ore.frame, ore.logical, ore.matrix, ore.number, ore.vector
Examples

```r
sdiv <- ore.push(state.division)
class(sdiv)  # ore.factor
levels(sdiv)
nlevels(sdiv)
summary(sdiv)

sdiv2 <- ore.pull(sdiv)
class(sdiv2)  # factor
```

Description

The `ore.frame` class represents data sets in Oracle R Enterprise.

Accessors

In the code snippets below, argument `x` is an `ore.frame` object.

- `nrow(x), ncol(x)`: Returns the number of rows and columns, respectively.
- `NROW(x), NCOL(x)`: Same as `nrow(x)` and `ncol(x)`, respectively.
- `dim(x)`: Returns an `integer` vector defined as `c(nrow(x), ncol(x))`.
- `length(x)`: Returns the number of columns.
- `colnames(x), names(x)`: Returns a `character` vector containing the column names.
  - `colnames(x) <- value, names(x) <- value`: Replaces the column names in argument `x` with the names in argument `value`.
- `rownames(x), row.names(x)`: Returns an `ore.character` object containing the row names. When the row names are made of multiple components, they will be separated with the value specified in the `ore.sep` option, which by default is set to "|".
  - `rownames(x) <- value, row.names(x) <- value`: Replaces the row names in argument `x` with the names in argument `value`. The `value` argument must be either `NULL` to remove row names, an `ore.vector` object for single component names, or an `ore.frame` object for multiple component names.
- `dimnames(x)`: Returns a `list` object defined as `list(NULL, colnames(x))`.

Subsetting

In the code snippets below, argument `x` is an `ore.frame` object. The vector selecting columns can be logical, numeric or character. The behavior is similar to that of `data.frame` with a few exceptions or rather extensions. When doing character subsetting on an `ore.frame` with complex rownames only the first component is used for subsetting. Subsetting on all components is not yet supported.

- `x[i, j, drop=TRUE]`: Returns a new `ore.frame` object made of the selected rows and columns.
  - For single column selection, the `drop` argument specifies whether or not to return an `ore.vector` object.
- `x[j] <- value, x[, j] <- value`: Replaces the specified columns in `x` with `value`. When `value` is `NULL`, the specified columns are removed from `x`. 
ore.frame-class

`x[[i]], x$name`: Returns the selected column as an `ore.vector` object.

`x[[i]] <- value, x$name <- value`: Replaces the specified column in `x` with `value`. When `value` is `NULL`, the column is removed from `x`.

`head(x, n = 6L)`:
- If argument `n` is non-negative, returns the first `n` rows of argument `x`.
- If argument `n` is negative, returns all but the last `abs(n)` rows of argument `x`.

`tail(x, n = 6L)`:
- If argument `n` is non-negative, returns the last `n` rows of argument `x`.
- If `n` is negative, returns all but the first `abs(n)` rows of argument `x`.

`subset(x, subset, select, drop = FALSE)`:
- Returns a new `ore.frame` object using:
  - `subset` A logical expression indicating rows to keep, where missing values are taken as a logical `FALSE` value.
  - `select` An expression indicating columns to keep.
  - `drop` Argument passed on to `[` indexing operator.

`is.na(x)`:
- Returns an `ore.frame` object containing `logical` data columns that indicate which cells contain missing values.

`is.finite(x)`:
- Returns an `ore.frame` object containing `logical` data columns that indicate which cells contain finite numbers.

`is.infinite(x)`:
- Returns an `ore.frame` object containing `logical` data columns that indicate which cells contain infinite values.

`is.nan(x)`:
- Returns an `ore.frame` object containing `logical` data columns that indicate which cells contain a not-a-number value.

**Splitting and Combining**

In the code snippets below, argument `x` is an `ore.frame` object.

`split(x, f, drop = FALSE)`:
- Splits argument `x` into a `list` object, according to argument `f`, dropping elements corresponding to unrepresented levels if `drop` is `TRUE`.

`cbind(...)`:
- Returns a new `ore.frame` object by combining the columns of the `ore.frame` objects in `...`.

`rbind(...)`:
- Returns a new `ore.frame` object by combining the rows of the `ore.frame` objects in `...`.

`merge(x, y, ...)`:
- Merges two `ore.frame` objects `x` and `y`, with arguments in `...` being the same as those allowed by the base `merge` function. It is allowed for either arguments `x` or `y` to be a `data.frame` object. If both inputs are ordered `ore.frame` objects, the result will also be an ordered `ore.frame` object with a complex key made from `x` and `y` keys. Otherwise, the result is unordered.

**Looping**

In the code snippets below, argument `x` is an `ore.frame` object.

`by(data, INDICES, FUN, ..., simplify = TRUE)`:
- Apply argument `FUN` to each partitioning of argument `data`, an `ore.frame` object, specified by the `factor` (or list of factor objects) argument `INDICES`.

**Utilities**

In the code snippets below, argument `x` is an `ore.frame` object.

`unique(x)`:
- Returns a new `ore.frame` object that contains only the distinct rows in argument `x`. 
Evaluation

eval(expr, envir, enclos = parent.frame()): Converts the ore.frame object specified in argument envir to an environment using function as.env with argument enclos as its parent, and then evaluates argument expr within that environment.

with(data, expr, ...): Equivalent to expression eval(quote(expr), data, ...).

within(data, expr, ...): Similar to function with, except assignments made during evaluation are taken as assignments into argument data, i.e., new symbols have their value appended to argument data, and assigning new values to existing symbols results in replacement.

transform(~_data~, ...): Similar to function within, except assignments are specified as independent optional arguments instead as depended evaluations within a single expression.

Matrix Methods

colMeans(x, na.rm = FALSE): Returns an ore.number object containing the column means of argument x.

colSums(x, na.rm = FALSE): Returns an ore.number object containing the column sums of argument x.

rowMeans(x, na.rm = FALSE): Returns an ore.number object containing the row means of argument x.

rowSums(x, na.rm = FALSE): Returns an ore.number object containing the row sums of argument x.

scale(x, center = TRUE, scale = TRUE): Returns an ore.frame object containing the possibly centered and scaled version of argument x.

max.col(m, ties.method = c("first", "last")): Returns an ore.integer object containing the column number of the maximum value for each row of argument m.

Group Generics

ore.frame objects have support for S4 group generic functionality:

Arith "+", "-", "+", "/", "/%", "/%", "/%/%", "/%/%/, "/%"/

Compare "==", ">", "<", ">=", ">=", ">="

Logic "&", "|"

Ops "Arith", "Compare", "Logic"


Math2 "round", "signif"

Summary "max", "min", "range", "prod", "sum", "any", "all"

See S4groupGeneric for more details.

Logical Methods

!x: Returns an ore.frame object containing the logical negation (NOT) of argument x.

xor(x, y): Returns an ore.frame object containing the exclusive or combination of arguments x and y.
Coercion

In the code snippets below, argument x is an ore.frame object.

- `as.env(x, enclos = parent.frame())`: Returns an environment object containing an ore.vector for each column in argument x.
- `as.list(x)`: Returns a list object containing an ore.vector for each column in argument x.
- `as.matrix(x)`: Returns an ore.matrix object.

Note

See the corresponding R documentation for the functions listed above.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore, ore.character, ore.datetime, ore.factor, ore.logical, ore.matrix, ore.number, ore.vector

Examples

IRIS <- ore.push(iris)
head(IRIS)
summary(IRIS)
colMeans(IRIS[1:4])
Arguments

x
A formula or \texttt{ore.frame} object containing numeric columns.
factors
The number of factors in the analysis.
data
An optional \texttt{ore.frame} to use when argument \texttt{x} is a formula.
covmat, n.obs
Ignored when supplying \texttt{ore.frame} input data.
subset
An optional subset expression.
na.action
The manner in which to handle NA values, either \texttt{na.omit} or \texttt{na.pass}.
start
Optional starting values.
scores
The type of scores to produce; only settings of "none" or "regression" are supported.
rotation
The type of rotation to use.
control
A list of control values.
...
Additional optional arguments.

Value

A \texttt{factanal} object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

\texttt{factanal, ore.frame-prcomp, ore.frame-princomp}

Examples

\begin{verbatim}
LONGLEY <- ore.push(longley)

factanal(LONGLEY, 3)
factanal(~ GNP + Unemployed + Population + Employed, 1, data = LONGLEY)
\end{verbatim}

Description

Model functions related to variable preparation of \texttt{ore.frame} data in statistical modeling.
Usage

```r
## S4 method for signature 'ore'
complete.cases(...)
## S4 method for signature 'formula,ore.frame'
get_all_vars(formula, data = NULL, ...)
## S4 method for signature 'formula'
model.frame(formula, data = NULL, subset = NULL,
    na.action = getOption("na.action", "na.omit"),
    drop.unused.levels = FALSE, xlev = NULL, ...)
## S4 method for signature 'formula'
model.matrix(object, data = environment(object),
    contrasts.arg = NULL, xlev = NULL,
    na.action = getOption("na.action", "na.omit"), ...)
## S4 method for signature 'ore.frame'
na.omit(object, ...)
```

Arguments

- `formula`: A model `formula` or a `terms` object.
- `object`: For the `model.matrix` method, a model `formula` or a `terms` object. For the `na.omit` method, an `ore.frame` object.
- `data`: An `ore.frame` object.
- `subset`: An `expression` object for row selection.
- `na.action`: The manner in which NA values are handled, either `na.omit` or `na.pass`.
- `drop.unused.levels`: This argument is not supported.
- `xlev`: A named `list` of `character` vectors specifying the `levels` for each `ore.factor` variable.
- `contrasts.arg`: A named `list` of `contrasts`, defined as numeric matrices or character strings specifying contrast functions, for the `ore.factor` variables.
- `...`: Additional arguments.

Value

For the `complete.cases` method, returns an `ore.logical` object.

For the `get_all_vars`, `model.frame`, and `na.omit` methods, returns an `ore.frame` object.

For the `model.matrix` method, returns an `ore.matrix` object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise
Description

Principal components analysis of \texttt{ore.frame} data.

Usage

\begin{verbatim}
## S4 method for signature 'ANY'
prcomp(x, ...)

# prcomp(formula, data = NULL, subset, na.action, ...)
# prcomp(x, retx = TRUE, center = TRUE, scale. = FALSE,
#       tol = NULL, ...)
\end{verbatim}

Arguments

\begin{itemize}
  \item \code{x} See Details section below.
  \item \code{...} Additional arguments.
\end{itemize}

Details

This is a wrapper method around the \texttt{prcomp} function in the \texttt{stats} package to support the analysis of \texttt{ore.frame} objects. As with the original function, it can be used through two different signatures: \texttt{prcomp(formula, data = NULL, subset, na.action, ...)} and \texttt{prcomp(x, retx = TRUE, center = TRUE, scale. = FALSE, tol = NULL, ...)}. Descriptions of the arguments for these two function signatures are the following:

\begin{description}
  \item[formula] A formula with no response variable that refers only to numeric variables.
  \item[data] An \texttt{ore.frame} object that contains the variables specified in the \code{formula} argument.
  \item[subset] An optional subset expression.
  \item[na.action] The manner in which to handle NA values, either \code{na.omit} or \code{na.pass}.
  \item[x] An \texttt{ore.frame} object that contains only numeric data.
  \item[retx] A logical value indicating whether the rotated variables should be returned.
\end{description}
The `center` argument to be used internally by the `scale` function.

`scale`. The `scale` argument to be used internally by the `scale` function.

tol See the description of the `tol` argument in `prcomp`.

Value

A `prcomp` object.

Note

The function `biplot` currently does not work on the `prcomp` object returned by this ORE wrapper method.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`prcomp`, `ore.frame-factanal`, `ore.frame-princomp`

Examples

```r
USARRESTS <- ore.push(USArrests)

prcomp(USARRESTS)
prcomp(USARRESTS, scale. = TRUE)

# formula interface
prcomp(~ Murder + Assault + UrbanPop, data = USARRESTS, scale. = TRUE)
```

ore.frame-princomp  Oracle R Enterprise Principal Components Analysis

Description

Performs principal components analysis of `ore.frame` data.

Usage

```r
## S4 method for signature 'ANY'
princomp(x, ...)

# princomp(formula, data, subset, na.action, cor = FALSE,
# scores = TRUE, ...)
# princomp(x, subset, na.action, cor = FALSE, scores = TRUE, ...)
```
Arguments

x  See Details section below.
...

Additional arguments.

Details

This is a wrapper method around the princomp function in the stats package to support the analysis of ore.frame objects. As with the original function, it can be used through two different signatures:

princomp(formula, data, subset, na.action, cor = FALSE, scores = TRUE, ...)

and princomp(x, subset, na.action, cor = FALSE, scores = TRUE, ...). Descriptions of the arguments for these two function signatures are the following:

- **formula**  A formula with no response variable that refers only to numeric variables.
- **data**  An ore.frame object that contains the variables specified in the formula argument.
- **subset**  An optional subset expression.
- **na.action**  The manner in which to handle NA values, either na.omit or na.pass.
- **x**  An ore.frame object that contains only numeric data.
- **cor**  A logical value that indicates whether the principal components should be based on the correlation matrix (cor = TRUE) or the covariance matrix (cor = FALSE).
- **scores**  A logical value that indicates whether the principal component scores should be included in the output.

Value

A princomp object.

Note

The function biplot currently does not work on the princomp object returned by this ORE wrapper method.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

princomp, ore.frame-factanal, ore.frame-prcomp

Examples

USARRESTS <- ore.push(USArrests)

princomp(USARRESTS)
princomp(USARRESTS, cor = TRUE)

# formula interface
princomp(~ Murder + Assault + UrbanPop, data = USARRESTS, cor = TRUE)
Description

Operates on output from the function `ore.crosstab` and automatically determines techniques that are relevant for the table. For 1 way cross tables, goodness of fit tests for equal proportions or specific null proportions, confidence limits and tests are equivalence are supported. For 2 way cross tables, various statistics that describe relationships between columns involved in the cross tabulation are supported. This includes chi-square tests, cox-const-mantel-haenszel statistics, measures of association, strength of association, risk differences, odds ratio and relative risk for 2x2 tables and tests for trend. N-way tables specifications are build as N, 2 way cross tables.

Usage

```r
ore.freq(x, stats = NULL, params = NULL, persist = FALSE,
         skip.failed = FALSE, skip.missing = FALSE, use.ext = FALSE,
         use.r = FALSE, use.sql = FALSE)
```

Arguments

- **x**: Output from function `ore.crosstab`.
- **params**: A character string specifying the control parameters for the methods in argument `stats`. Possible setting include: Scoring Options: "SCORE=TABLE|RANK|RIDIT|MODRIDIT" Alpha criterion: "ALPHA=<Number>" Weights: "WEIGHTS=<Number>"
- **persist**: For internal use only.
- **skip.failed**: A logical value indicating whether to return with an error if a particular statistical test fails or to skip over the in-applicable test.
- **skip.missing**: A logical value indicating whether to skip cells with missing values in argument `x`.
- **use.ext, use.r, use.sql**: For internal use only.

Value

Returns an `ore.frame` object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>
ore.get

References

Oracle R Enterprise

See Also

ore.crosstab

Examples

## Not run:
IRIS <- ore.push(iris)
x <- ore.crosstab(Species ~ Petal.Length + Sepal.Length, data = IRIS)
ore.freq(x, stats = "SCORR, GAMMA, STUTC")

## End(Not run)

ore.get

Oracle R Enterprise Object Retrieval Function

Description

Retrieves the specified ore.frame object, representing a database table or view, from the R environment for a schema in the Oracle R Enterprise session.

Usage

ore.get(name, schema)

Arguments

name A character string specifying the name of the ore.frame object.
schema A character string specifying the database schema name.

Details

If argument schema is unspecified, a search is conducted across the attached R environments for the schemas in the Oracle R Enterprise session, according to the search list order, and returns the first object instance found.

Value

If found within argument schema, returns an ore.frame object. If not found, an error is produced.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise
ore.getXlevels

Factor Levels for an Oracle R Enterprise Model Matrix

See Also
ore.attach, ore.connect, ore.exists, ore.ls, ore.rm, ore.sync

Examples

if (!interactive())
{
  ore.create(data.frame(x=3, y="A", z=TRUE), table = "TABLE_GET")
  # search in the default schema
  ore.get("TABLE_GET")
  # search in the specified schema
  ore.get("TABLE_GET", "RQUSER")
  ore.drop(table = "TABLE_GET")
}

ore.getXlevels(Terms, m)
ore.getXnlevels(Terms, m)

Arguments

Terms       A terms or formula object.
m          An ore.frame or data.frame object that was generated by a model.frame function call.

Details

The ore.getXlevels function is the Oracle R Enterprise equivalent to the .getXlevels function in the stats package.

Value

For ore.getXlevels, a named list containing the factor levels for the categorical variables derived in the Terms argument.

For ore.getXnlevels, a table containing the number of factor levels in each of the categorical variables derived in the Terms argument.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>
ore.grant

References

Oracle R Enterprise

See Also

.getXlevels, ore.frame-model.frame

Examples

X <- ore.push(data.frame(V1 = -1:2,
                         V2 = 1:4,
                         V3 = rep(c("a", "b"), 2),
                         V4 = rep(c("A", "B"), c(2, 2))))
trms <- terms(V1 ~ log(V2) * V3 * V4)
mf <- model.frame(trms, data = X)
ore.getXlevels(trms, mf)
ore.getXnlevels(trms, mf)

ore.grant

Oracle R Enterprise Privilege Grant and Revoke Functions

Description

Grants or revokes read privilege for an R script or datastore.

Usage

ore.grant(name, type = c("datastore", "rqscript"), user = NULL)
ore.revoke(name, type = c("datastore", "rqscript"), user = NULL)

Arguments

name
  A character string that specifies the name of an R script in the R script repository
  or the name of a datastore. The current user must be the owner of the R script
  or datastore.

type
  A scalar character string specifying either ‘datastore’ or ‘rqscript’ to grant or
  revoke the read privilege. This argument must be specified.

user
  An optional character string specifying the user that read privilege of the named
  R script or datastore is granted to or revoked from. The default value NULL
  indicates the privilege is granted to or revoked from public.

Details

Functions ore.grant and ore.revoke require the user to have the ‘RQADMIN’ Oracle Database
role.

Value

Functions ore.grant and ore.revoke return an invisible NULL value if they succeed in priv-ilege grant or revoke; otherwise they produce an error.
Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.scriptCreate, ore.scriptDrop, ore.scriptLoad, ore.scriptList, ore.load, ore.datastore, ore.datastoreSummary

Examples

```r
if (!interactive())
{
  library(OREds)
  library(OREembed)
  # create an R script for the current user
  ore.scriptCreate("MYLM",
    function(data, formula, ...) lm(formula, data, ...))
  IRIS <- ore.push(iris)
  ore.tableApply(IRIS[1:4], FUN.NAME = "MYLM",
    formula = Sepal.Length ~ .)

  # create a global R script available to any user
  ore.scriptCreate("GLBGLM",
    function(data, formula, ...) glm(formula=formula, data=data, ...),
    global = TRUE)
  ore.tableApply(IRIS[1:4], FUN.NAME = "GLBGLM",
    formula = Sepal.Length ~ .)

  # list R scripts
  ore.scriptList()
  ore.scriptList(pattern="LM", type="all")

  # load an R script to an R function object
  ore.scriptLoad(name="MYLM")
  ore.scriptLoad(name="GLBGLM", newname="MYGLM")
  MYLM(iris, formula = Sepal.Length ~ .)
  MYGLM(iris, formula = Sepal.Length ~ .)

  # grant and revoke R script read privilege to and from public
  ore.grant(name = "MYLM", type = "rqscript")
  ore.scriptList(type="grant")
  ore.revoke(name = "MYLM", type = "rqscript")
  ore.scriptList(type="grant")

  # drop an R script
  ore.scriptDrop("MYLM")
  ore.scriptDrop("GLBGLM", global=TRUE)

  ore.scriptList(type="all")

  # create grantable datastores
```
ore.save(iris, name="ds_1", grantable=TRUE)
ore.save(mtcars, name="ds_2", grantable=TRUE)

# grant the read privilege of one datastore to every user
ore.grant(name="ds_1", type="datastore", user=NULL)

# show all the datastores
ore.datastore(type="all")[-5L]

# show the grantable datastores
ore.datastore(type="grantable")[-4L]

# show the datastore the read privilege for which was granted to other users
ore.datastore(type="grant")

# revoke the granted privilege
ore.revoke(name="ds_1", type="datastore", user=NULL)

ore.delete(name="ds_1")
ore.delete(name="ds_2")

---

**ore.hash**

*Oracle R Enterprise ore.vector Hash Function*

**Description**

Hashes the values in an `ore.vector` object.

**Usage**

```r
ore.hash(x, ...)  
## S4 method for signature 'ore.vector'
ore.hash(x, size = 4294967296,
        seed = round(runif(1, min = 0.0, max = 4294967295.5)), ...)
```

**Arguments**

- `x`: An `ore.vector` object.
- `size`: A numeric value within \([1, 4294967296]\) representing the total number of hash buckets.
- `seed`: A numeric value within \([0, 4294967295]\) representing the seed value for the hash function.
- `...`: For future expansion.

**Details**

For the Oracle Database, the SQL snippet `ORA_HASH(<SQL expr>, size - 1, seed) + 1` is used to calculate the hash value.

**Value**

Returns an `ore.numeric` object containing the hash values for the original vector.
ore.hiveOptions

Sets HIVE options, namely, field delimiters for the HIVE tables and the current database name.

Usage

ore.hiveOptions(delim = "\001", dbname, storedAs='textfile',
       exeEngine = 'mr')

Arguments

delim A character string specifying the field delimiter for the HIVE tables created by
       ORCH-HIVE.
dbname A character string specifying the HIVE database name for the current session.
storedAs A character string specifying the HIVE table storage type for the current session.
exeEngine A character string specifying the HIVE execution engine for the current session.

Details

Use the delim argument to configure the separator character between fields (columns) in the text
files corresponding to the HIVE tables. Use the dbname argument to set the database name for
ORCH-HIVE to work on. The database must pre-exist otherwise an error is thrown. This is an
environment-level change and once the arguments are set, they are respected by all the tables created
using ORCH-HIVE in the R session.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

See Also

ore.showHiveOptions
ore.impalaOptions

Examples

```r
if(ore.is.connected(type="HIVE"))
{
  if (!interactive())
    ore.hiveOptions(delim = ",")
}
```

ore.impalaOptions  
**Set IMPALA Options**

Description

Sets IMPALA options, namely, field delimiters for the IMPALA tables and the current database name.

Usage

```r
ore.impalaOptions(delim = "\001", dbname, storedAs='textfile',
                   exeEngine = 'mr')
```

Arguments

- **delim**: A character string specifying the field delimiter for the IMPALA tables created by ORCH-HIVE.
- **dbname**: A character string specifying the IMPALA database name for the current session.
- **storedAs**: A character string specifying the IMPALA table storage type for the current session.
- **exeEngine**: A character string specifying the IMPALA execution engine for the current session.

Details

Use the `delim` argument to configure the separator character between fields (columns) in the text files corresponding to the IMPALA tables. Use the `dbname` argument to set the database name for ORCH-HIVE to work on. The database must pre-exist otherwise an error is thrown. This is an environment-level change and once the arguments are set, they are respected by all the tables created using ORCH-HIVE in the R session.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

See Also

`ore.showImpalaOptions`
ore.itemsets-class

Class ore.itemsets

Description

The ore.itemsets class represents a set of itemsets of an ore.odmAssocRules object in an Oracle database. An ore.odmAssocRules object represents an Association model created by Oracle Data Mining.

Subsetting

In the code snippets below, argument x is an ore.itemsets object.

subset(x, min.support=NULL, max.itemset.length=NULL, min.itemset.length=NULL, itemset.id=NULL, items=NULL)

Returns a new ore.itemsets object using:

min.support The minimum support of the itemsets to return.
max.itemset.length The maximum length of the itemsets to return.
min.itemsets.length The minimum length of the itemsets to return.
itemset.id The itemset ID (numeric type) of the itemsets to return.
items A list of items. This method returns all itemsets that contain one or more of the specified items. Examples of items lists are items=list("apple","orange") and items=list(age=31,occupation="engineer").

Utilities

In the code snippets below, argument x is an ore.itemsets object.

ore.pull(x): Returns the in-memory counterpart of the ore.itemsets object, which is an itemsets object in the arules package.

Details

The ore.itemsets class represents a set of itemsets in an Oracle Database. The set of itemsets is the result of an Association Rules model.

The column names of an ore.itemsets object are "ITEMSET_ID", "NUMBER_OF_ITEMS", "ITEMS", and "SUPPORT".

You can pull itemsets into memory in a local R session from an Oracle Database instance by using ore.pull. The in-memory object is of class itemsets, which is defined in the arules package.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>
ore.lazyLoad

References

Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User’s Guide

See Also

ore.rules, ore.odmAssocRules

Examples

id <- c(1, 1, 1, 2, 2, 2, 3, 3, 3, 3)
item <- c("b", "d", "e", "a", "b", "c", "e", "b", "c", "d", "e")
data.ore <- ore.push(data.frame(ID = id, ITEM = item))

ar.mod <- ore.odmAssocRules(~., data.ore, case.id.column = "ID",
  item.id.column = "ITEM", min.support = 0.6, min.confidence = 0.6,
  max.rule.length = 3)

itemsets <- itemsets(ar.mod)

sub.itemsets1 <- subset(itemsets, items=list("b", "c"))
sub.itemsets2 <- subset(sub.itemsets1, min.itemset.length=2)

library(arules, warn.conflicts=FALSE)
items.arules <- ore.pull(itemsets)
inspect(items.arules[1:2])
items.arules1 <- ore.pull(sub.itemsets1)
items.arules2 <- ore.pull(sub.itemsets2)

ore.lazyLoad

**Oracle R Enterprise Datastore Lazy Loading Function**

Description

Lazy (just in time) loads R objects from the named Oracle R Enterprise datastore in the user’s Oracle Database schema.

Usage

```
ore.lazyLoad(name, list = character(0), envir = parent.frame())
```

Arguments

- `name` A character string specifying the name of the datastore.
- `list` An optional character vector containing the names of the objects within the datastore to lazy load. If this argument is not specified, then all of the objects within the datastore will be lazy loaded.
- `envir` The environment into which the objects are to be lazy loaded.
ore.lazyLoad

Details

Unlike the ore.load function, the ore.lazyLoad function does not immediately retrieve the specified objects from an Oracle R Enterprise datastore. Instead objects are retrieved upon first reference to the object. See examples below for implications of lazy loading.

Value

Function ore.lazyLoad returns a character vector specifying the names of objects that are lazy loaded from the datastore.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.save, ore.delete, ore.datastore, ore.datastoreSummary, ore.load

Examples

```r
# save object to new datastore
x <- 1:10
ore.save(x, name = "rq$ds_1")

# enable lazy loading of object
e <- new.env()
ore.lazyLoad(name = "rq$ds_1", envir = e)
bindingIsActive("x", e)

# load object
e$x
bindingIsActive("x", e)

# re-enable lazy loading
ore.lazyLoad(name = "rq$ds_1", envir = e)
bindingIsActive("x", e)

# overwrite object in datastore
x <- letters
ore.save(x, name = "rq$ds_1", overwrite = TRUE)

# load updated object
e$x
bindingIsActive("x", e)

# clean up
rm(x, e)
ore.delete(name = "rq$ds_1")
```
Description

An `ore` class for representing and manipulating `list` data stored in an Oracle database.

List Methods

- `x[[i]]`, `x$name`: Returns the selected element as an `ore.object` object.
- `length(x)`: Returns an `integer` value representing the number of elements.
- `names(x)`: Returns a `character` vector containing the element names.

Note

See the corresponding R documentation for the functions listed above.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore`, `ore.object`

Examples

```r
x <- list(A = 1, B = "value", C = TRUE)
X <- ore.push(x)
class(X) # ore.list
X$A
X[[2]]
X["C"]
length(X)
names(X)
X

y <- ore.pull(X)
class(y) # list
```
ore.load  

Oracle R Enterprise Datastore Loading Function

Description

Loads R objects from a datastore in the user’s Oracle Database schema.

Usage

ore.load(name, list = character(0), owner, envir = parent.frame())

Arguments

name  
A character string specifying the name of the datastore.

list  
An optional character vector containing the names of the R objects within the datastore to load. If this argument is not specified, then all of the R objects within the datastore will be loaded.

owner  
An optional character string specifying the owner of the datastore to load. If the owner of the datastore is not the current user, the read privilege for the datastore must be granted to the current user by ore.grant; otherwise ore.load errors out. If the owner argument not specified, the default value is the current user.

envir  
The environment into which the R objects are to be loaded.

Details

Functions ore.save and ore.load operate together in a similar manner to functions save and load from the base package. Whereas the save and load functions save and load R objects to and from an external file, the ore.save and ore.load functions save and load R objects from a datastore.

Value

Function ore.load returns a character vector specifying the names of the R objects that are loaded from the datastore to environment envir.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.save, ore.move, ore.delete, ore.datastore, ore.datastoreSummary, ore.lazyLoad, ore.grant, ore.revoke
Examples

```r
if (any(sapply(c("x", "y", "z", "e"), exists)))
  stop("object x, y, z, or e exists")

x <- stats::runif(20)
y <- list(a = 1, b = TRUE, c = "oops")
z <- ore.push(x)

# save all objects in the current workspace environment to
# a datastore with name 'rq$ds_1' in the user's schema
ore.save(list = ls(), name = "rq$ds_1")
rm(x, y, z)

# load all objects from datastore rq$ds_1 to current workspace
ore.load(name = "rq$ds_1")
ls()

# load x and y from datastore rq$ds_1 to environment e
e <- new.env()
ore.load(name = "rq$ds_1", list = c("x", "y"), envir = e)
lse(envir = e)

# clean up
ore.delete(name = "rq$ds_1")
rm(x, y, z, e)
```

ore.logical-class

Class ore.logical

Description

The ore.logical class represents logical data columns in Oracle R Enterprise.

Group Generics

ore.logical objects have support for S4 group generic functionality:

Logic "&", "|

See S4groupGeneric for more details.

Logical Methods

!x: Returns an ore.logical object containing the logical negation (NOT) of argument x.

ifelse(x, yes, no): For each element of argument x, returns the corresponding element in argument yes if TRUE, otherwise returns the element in argument no. Arguments yes and no may be length one vectors (scalar values) or ore.vector objects.

xor(x, y): Returns an ore.logical object containing the exclusive or combination of arguments x and y.

Note

See the corresponding R documentation for the functions listed above.
ore.ls

Oracle R Enterprise Object Listing Function

Description

Returns a vector of character strings giving the names of the ore.frame objects, representing database tables and views, available in the R environment for a schema in the Oracle R Enterprise session.

Usage

ore.ls(schema, all.names = FALSE, pattern)

Arguments

- **schema**: A character string specifying the database schema name.
- **all.names**: A logical value indicating whether to include objects with a leading . character.
- **pattern**: An optional regular expression whereby only matching names are returned.

Details

If argument `schema` is unspecified, the default schema - the one specified at connection time for the Oracle R Enterprise session - is used.

Value

A character vector.
ore.make.names

Author(s)
Oracle <oracle-r-enterprise@oracle.com>

References
Oracle R Enterprise

See Also
ore.attach, ore.connect, ore.exists, ore.get, ore.rm, ore.sync

Examples
if (!interactive())
{
  ore.ls()
  ore.ls("RUSER")
  ore.ls("RUSER", all.names = TRUE)
  ore.ls("RUSER", pattern = "IR")
}

ore.make.names  Oracle R Enterprise Valid Column Name Generator

Description
Creates valid column names for ore.frame objects.

Usage
ore.make.names(names)

Arguments
names  A character vector specifying unvalidated names.

Details
Creates distinct strings from argument names that are at most 30 bytes long using an iterative algorithm involving the call
make.unique(abbreviate(names, minlength, strict = TRUE))
where minlength <= 30.

Value
A character vector with the same length as argument names containing valid column names for an ore.frame object.

Author(s)
Oracle <oracle-r-enterprise@oracle.com>
ore.matrix-class

References
Oracle R Enterprise

See Also
make.names, abbreviate

Examples

xnames <- c("col1", "Col.2", "COL_3", "col 4", "coll",
"L_A_S_T C.O.L.U.M.N abcdefghijklmnopqrstuvwxyz")
ore.make.names(xnames)

ore.matrix-class  Class ore.matrix

Description
The ore.matrix class represents numeric matrices in Oracle R Enterprise.

Accessors
In the code snippets below, argument x is an ore.frame object.

nrow(x), ncol(x): Returns the number of rows and columns, respectively.
NROW(x), NCOL(x): Same as nrow(x) and ncol(x), respectively.
dim(x): Returns an integer vector defined as c(nrow(x), ncol(x)).
rownames(x), colnames(x): Returns the row and column names, respectively.
rownames(x) <- value, colnames(x) <- value: Replaces the row and column names, respectively.
dimnames(x): Returns a list object defined as list(rownames(x), colnames(x)).
dimnames(x) <- value: Replaces the row and column names.

Subsetting
In the code snippets below, argument x is an ore.matrix object.

x[i,j,drop=TRUE]: Returns a new ore.matrix object made of the selected rows and columns. For single column selection, the drop argument specifies whether or not to return an ore.vector object.

Group Generics
ore.matrix objects have support for S4 group generic functionality:

Arith "+", "-", "+", "/", "/", "/", "/", "%\%", "/", "%\%/","/"
Math "abs", "sign", "sqrt", "ceiling", "floor", "trunc", "cummax", "cummin",
"cumprod", "cumsum", "log", "log10", "log2", "log1p", "acos", "acosh",
"asin", "asinh", "atan", "atanh", "exp", "expm1", "cos", "cosh", "sin",
"sinh", "tan", "tanh", "gamma", "lgamma", "digamma", "trigamma"
Summary "max", "min", "range", "prod", "sum", "any", "all"

See S4groupGeneric for more details.
Matrix Operations

\texttt{t(x)}: Returns the matrix transpose of argument \( x \).

\texttt{crossprod(x, y = NULL), tcrossprod(x, y = NULL)}: Returns the matrix crossproduct of the \( x \) and \( y \) arguments.

\( x \%\%\% y \): Returns the matrix product of the \( x \) and \( y \) arguments.

\texttt{colMeans(x, na.rm = FALSE)}: Returns an \texttt{ore.number} object containing the column means of argument \( x \).

\texttt{colSums(x, na.rm = FALSE)}: Returns an \texttt{ore.number} object containing the column sums of argument \( x \).

\texttt{rowMeans(x, na.rm = FALSE)}: Returns an \texttt{ore.number} object containing the row means of argument \( x \).

\texttt{rowSums(x, na.rm = FALSE)}: Returns an \texttt{ore.number} object containing the row sums of argument \( x \).

\texttt{scale(x, center = TRUE, scale = TRUE)}: Returns an \texttt{ore.matrix} object containing the possibly centered and scaled version of argument \( x \).

\texttt{max.col(m, ties.method = c("first", "last"))}: Returns an \texttt{ore.integer} object containing the column number of the maximum value for each row of argument \( m \).

Note

See the corresponding \texttt{R} documentation for the functions listed above.

Author(s)

Oracle \(<\text{oracle-r-enterprise@oracle.com}>\)

References

Oracle R Enterprise

See Also

\texttt{ore, ore.character, ore.datetime, ore.factor, ore.frame, ore.logical, ore.number, ore.vector}

Examples

\texttt{IRIS\_MATRIX <- ore.push(as.matrix(iris[1:4]))}
\texttt{crossprod(IRIS\_MATRIX)}
ore.move

Oracle R Enterprise Datastore Moving Function

Description
Renames a datastore in the user’s Oracle Database schema, moves or renames R objects within a datastore, or both moves and renames R objects within a datastore.

Usage
```r
ore.move(name = stop("parameter 'name' must be specified"),
          newname = NULL,
          object.names = NULL,
          object.newnames = NULL,
          overwrite = FALSE)
```

Arguments
- **name**: A character string specifying the datastore to be renamed or modified.
- **newname**: A character string specifying the newname of the datastore or the name of the datastore that is the destination for objects moved from the source datastore. Can be left unspecified if objects are only being renamed and not moved out of the datastore specified by `name`. Cannot be an empty string.
- **object.names**: A character vector specifying the R objects to be moved or renamed, or both.
- **object.newnames**: A character vector specifying the newnames of the R objects specified by `object.names`. If not `NULL`, must be the same length as `object.names`. Cannot contain an empty string.
- **overwrite**: If `TRUE` and renaming a datastore, then if the datastore specified by `newname` already exists, the datastore specified by `newname` and its contents are overwritten by the contents of the datastore specified by `name`. If `TRUE` and moving or renaming datastore R objects, or both, then if the destination datastore already has objects whose names are in `object.newnames`, those objects are overwritten. If `FALSE`, then an attempt to rename a datastore or object to an existing datastore or object, or an attempt to move an R object to a datastore where an object of the same name already exists will cause an error. Default value is `FALSE`.

Details
This function performs one of three actions depending on the input. If `object.names` or `object.newnames` is a character vector of length zero, the function treats it as `NULL`.

1) Rename a datastore in the user’s Oracle Database schema. Neither `name` nor `newname` may be `NULL`, and `newname` may not be an empty string. `old.obj.name` and `object.newnames` must both be `NULL`. The datastore specified by `name` will be renamed to the value specified by `newname`.

2) Rename objects within a datastore. `name` must not be `NULL`. `newname` must be either `NULL` or the same as `name`. `object.names` and `object.newnames` must have distinct values, and neither can be `NULL`.
3) Move objects from one datastore to another. name and newname must have distinct values, and neither can be NULL. object.names cannot be NULL. If object.newnames is not NULL and if the names in it are distinct from object.names, the objects are renamed in the datastore specified by newname. An error occurs if there is not a one-to-one mapping of names in object.names to object.newnames.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.save, ore.load, ore.delete, ore.datastore ore.datastoreSummary

Examples

if (any(sapply(c("a1", "a2", "b1", "b2", "w1", "w2", "x1", "x2", "z1", "z2"), exists)))
  stop("object a1, a2, b1, b2, w1, w2, x1, x2, z1, or z2 exists")

if (nrow(ore.datastore(pattern="^rq\$ds_\[123\]$")) > 0)
  stop("datastore 'rq$ds_1', 'rq$ds_2', or 'rq$ds_3' exists")

a1 <- data.frame(a=c("foo", "bar"), b=c("xx", "yy"))
a2 <- data.frame(a=1:10, b=11:20)
b1 <- ore.push(a1)
b2 <- ore.push(a2)
w1 <- stats::runif(20)
w2 <- stats::runif(50)
x1 <- list(a = 1, b = TRUE, c = "oops")
x2 <- list(a = w1, b = w2, c = "data", d=200)
z1 <- ore.push(x1)
z2 <- ore.push(w1)

ore.save(a1, a2, w1, w2, x1, x2, name="rq$ds_1")
ore.save(b1, b2, name="rq$ds_2")
ore.save(z1, z2, name="rq$ds_3")

#rename a datastore
ore.move(name="rq$ds_1", newname="rq$ds_1_renamed")
ds <- ore.datastore()
## Not run:
ds
## End(Not run)

#rename objects within a datastore
ore.move(name="rq$ds_1_renamed", object.names=c("x1", "x2"),
          object.newnames=c("y1", "y2"))
ore.datastoreSummary(name="rq$ds_1_renamed")

#move objects from one datastore to another
ore.move(name="rq$ds_1_renamed", newname="rq$ds_2", object.names=c("y1", "y2"))
ore.datastoreSummary(name="rq$ds_1_renamed")
ore.datastoreSummary(name="rq$ds_2")

#move objects from one datastore to another, and renaming them
ore.move(name="rq$ds_2", newname="rq$ds_1_renamed", object.names=c("y1", "y2"),
        object.newnames=c("z1", "z2"))
ore.datastoreSummary(name="rq$ds_1_renamed")
ore.datastoreSummary(name="rq$ds_2")

#rename a datastore, overwriting the existing datastore
ore.move(name="rq$ds_1_renamed", newname="rq$ds_2", overwrite = TRUE)
ore.datastoreSummary(name="rq$ds_2")

#rename objects within a datastore, overwriting existing objects of the same name
ore.move(name="rq$ds_2", object.names=c("w1", "w2"), object.newnames=c("z1", "z2"),
         overwrite = TRUE)
ore.datastoreSummary(name="rq$ds_2")

#transfering objects to another datastore, overwriting existing objects of the same name
ore.move(name="rq$ds_2", newname="rq$ds_3", object.names=c("z1", "z2"),
         overwrite=TRUE)
ore.datastoreSummary(name="rq$ds_2")
ore.datastoreSummary(name="rq$ds_3")

#rename objects while transfering them to another datastore, overwriting existing objects
ore.move(name="rq$ds_3", newname="rq$ds_2", object.names=c("z1", "z2"),
         object.newnames=c("a1", "a2"), overwrite=TRUE)
ore.datastoreSummary(name="rq$ds_2")
ore.datastoreSummary(name="rq$ds_3")

ore.delete(name="rq$ds_2")
ore.delete(name="rq$ds_3")
rm(a1, a2, b1, b2, w1, w2, x1, x2, z1, z2)

---

ore.number-class  Class ore.number

Description

The ore.number subclasses ore.integer and ore.numeric represent integer and numeric data columns in Oracle R Enterprise.

Group Generics

ore.number objects have support for S4 group generic functionality:

Arith `+` `*` `-` `^` `%%` `%/%` `/`
Compare `==` `>` `<` `!=` `<>` `<=` `>=`
Logic `&` `|`
Ops "Arith", "Compare", "Logic"

Math2 "round", "signif"

Summary "max", "min", "range", "prod", "sum", "any", "all"

See S4groupGeneric for more details.

**Numerical Data Methods**

`cut(x, breaks, labels = NULL, include.lowest = FALSE, right = TRUE, dig.lab = 3L, ...)`: Returns an ore.vector object containing a categorical representation of `x`-based interval ranges. When `labels` is `FALSE`, the output is an ore.integer object; otherwise returns an ore.factor object.

- **breaks** Either a numeric vector of two or more distinct cut points or an integer value greater than or equal to 2 that specifies the number of intervals to create.
- **labels** Either a character vector containing the labels for the resulting categories, the logical value `FALSE` to specify an integer coding for the intervals, or `NULL` to construct labels using `(a,b]` or `[a,b)` interval notation.
- **include.lowest** A logical value indicating if the lowest (when argument `right` is `TRUE`) or highest (when argument `right` is `FALSE`) interval should be closed, i.e. of the form `[a,b]`.
- **right** A logical value indicating if the half-open intervals should be closed on the right or left.
- **dig.lab** When argument `labels` is `NULL`, the number of digits to use in the formatting of the break numbers in the labels.

`diff(x, lag = 1, differences = 1)`: Returns suitably lagged and iterated differences of argument `x`.

- **lag** An integer indicating which lag to use.
- **differences** An integer indicating the order of the difference.

`is.finite(x)`: Returns an ore.logical object indicating which values in argument `x` contain finite numbers.

`is.infinite(x)`: Returns an ore.logical object indicating which values in argument `x` contain infinite numbers.

`is.nan(x)`: Returns an ore.logical object indicating which values in argument `x` contain not-a-number values.

`scale(x, center = TRUE, scale = TRUE)`: Returns an ore.number object containing the possibly centered and scaled version of argument `x`.

`tabulate(bin, nbins = max(1, bin, na.rm = TRUE))`: Returns an integer vector containing counts of the number of times each integer occurs in argument `bin`.

`zapsmall(x, digits = getOption("digits"))`: Returns an ore.number object where values close to 0 are replaced with 0.

**Note**

See the corresponding R documentation for the functions listed above.
ore.number-multivar

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore, ore.character, ore.datetime, ore.factor, ore.frame, ore.logical, ore.matrix, ore.vector

Examples

v <- c(2, 4, 4, 7, 7)
V <- ore.push(v)
V
log(V)
tabulate(V)

descriptor

ore.number-multivar

Oracle R Enterprise Multivariate Statistics

Description

Performs multivariate numerical aggregation methods for ore.frame objects based on function in R’s stats package.

Usage

## S4 method for signature 'ore.frame'
cor(x, y = NULL, use = "everything",
    method = c("pearson", "kendall", "spearman"))

## S4 method for signature 'ore.frame'
cov(x, y = NULL, use = "everything",
    method = c("pearson", "kendall", "spearman"))

Arguments

x, y
Either argument x is an ore.frame object containing numeric columns and argument y is NULL or arguments x and y are ore.number objects.

use
A method of computation when missing values are present. One of "everything", "all.obs", "complete.obs", or "na.or.complete".

method
For cor method where x and y are ore.number objects, can be one of "pearson", "kendall", "spearman"; otherwise must be "pearson".
Details

These statistics are calculated using a custom analytic based on the crossproduct of the centered (and in the case of *cor* scaled) data matrix.

Unlike the *cor* and *cov* functions in the *stats* package, `use = "pairwise.complete.obs"` and `method %in% c("kendall","spearman")` are not supported.

Value

A matrix of dimension `ncol(x) by ncol(x)`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

*cor*, *cov*, *ore.number-univar*, *ore.vector-aggregate*

Examples

```r
LONGLEY <- ore.push(longley)
all.equal(cor(LONGLEY), cor(longley))
all.equal(cov(LONGLEY), cov(longley))
```

Description

Performs univariate numerical aggregation methods for *ore.number* vectors based on the corresponding function in R’s *stats* package.

Usage

```r
## S4 method for signature 'ore.number'
fivenum(x, na.rm = TRUE)
## S4 method for signature 'ore.number'
IQR(x, na.rm = FALSE, type = 7)
## S4 method for signature 'ore.number'
mad(x, center = median(x), constant = 1.4826,
    na.rm = FALSE, low = FALSE, high = FALSE)
## S4 method for signature 'ore.vector'
median(x, na.rm = FALSE)
## S4 method for signature 'ore.vector'
quantile(x, probs = seq(0, 1, 0.25),
    na.rm = FALSE, names = TRUE, type = 7, ...)
```
sd(x, na.rm = FALSE)
## S4 method for signature 'ore.number'
var(x, y = NULL, na.rm = FALSE, use)

Arguments

x        An ore.vector object for median and quantile; an ore.number object otherwise.
na.rm    A logical value indicating whether NA values should be ignored during the calculations.
type     The type of quantile algorithm to use according to Types section of quantile. Only 1 and 7 are supported.
center   The center value in median absolute deviation.
constant The scale factor in median absolute deviation.
low      A logical value indicating whether a 'lo-median' should be calculated in median absolute deviation.
high     A logical value indicating whether a 'hi-median' should be calculated in median absolute deviation.
probs    A numeric vector of probabilities with values in [0, 1].
names    A logical value indicating whether vector element names should be assigned to the quantile values.
y, use   Argument not supported.
...      Additional arguments.

Details

Each of these aggregations are performed within a SQL engine of an Oracle RDBMS. The Oracle SQL aggregate functions involved in these calculations include: AVG, MAX, MEDIAN, MIN, PERCENTILE_CONT, PERCENTILE_DISC, STDDEV_SAMP, and VAR_SAMP.

The fivenum method for ore.number uses combinations of results from Oracle SQL PERCENTILE_DISC to compute Tukey’s five number summary of minimum, lower-hinge, median, upper-hinge, and maximum.

For IQR and quantile methods, the type argument serves as a toggle between Oracle SQL PERCENTILE_CONT when type = 7 and Oracle SQL PERCENTILE_DISC when type = 1.

For mad method when the center argument is missing, a two-pass algorithm is used where the first pass calculates the median, represented by center, and the second pass produces constant * median(abs(x - center)).

For sd and var a two-pass algorithm is used where first the pass calculates the mean and the second pass produces STDDEV_SAMP(x - mean) and VAR_SAMP(x - mean).

Value

For IQR, mad, median, sd, and var, returns a numeric vector of length 1.

For fivenum, returns a numeric vector of length 5.

For quantile, returns a numeric vector containing the same number of values as elements in argument probs.
ore.object-class

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

fivenum, IQR, mad, median, quantile, sd, var, ore.number-multivar, ore.vector-aggregate

Examples

# Sample data
X <- data.frame(V1 = rnorm(100))
oreX <- ore.push(X)

# Test for equality with in-memory data
all.equal(fivenum(oreX$V1), fivenum(X$V1))
all.equal(IQR(oreX$V1), IQR(X$V1))
all.equal(mad(oreX$V1), mad(X$V1))
all.equal(median(oreX$V1), median(X$V1))
all.equal(quantile(oreX$V1), quantile(X$V1))
all.equal(quantile(oreX$V1, type = 1), quantile(X$V1, type = 1))
all.equal(sd(oreX$V1), sd(X$V1))
all.equal(var(oreX$V1), var(X$V1))
ore.odmAI

In-Database Attribute Importance Ranking Models

Description

Computes the relative importance of predictor variables for predicting a response variable using Oracle Data Mining.

Usage

```r
ore.odmAI (formula, data, auto.data.prep = TRUE, na.action = na.pass, odm.settings = NULL)
```

Arguments

- `formula`: An object of class `formula` (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
- `data`: An `ore.frame` object used for computing relative importance.
- `auto.data.prep`: A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build.
- `na.action`: A function to use for handling missing values, either `na.pass` to allow missing values or `na.omit` to remove rows with missing values. The default value is `na.pass`.
- `odm.settings`: Same as `odm.settings` in `ore.odmKMeans`.

Details

Attribute Importance uses a Minimum Description Length (MDL) based algorithm that ranks the relative importance of predictor variables in predicting a specified response (target) variable. Use this function to gain insight into the relevance of variables to guide manual variable selection or reduction, with the goal to reduce predictive model build time and/or improve model accuracy.

The `formula` specification has the form `response ~ terms` where `response` is the numeric or character response vector and `terms` is a series of terms, i.e., column names, to include in the analysis. Multiple terms are specified using `+` between column names. Use `response ~ .` if all columns in `data` should be used for model building. Functions can be applied to `response` and `terms` to realize transformations. To exclude columns, use `-` before each column name to exclude.

Value

An object of class `ore.odmAI` including the following components:

- `importance`: A `data.frame` of terms (predictor variables) as row names with their corresponding importance value and rank.
- `formula`: The `formula` provided when computing the relative importance.
- `call`: The matched call.
ore.odmAssocRules

Author(s)
Oracle
Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References
Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User’s Guide

See Also
ore.odmSVM, ore.odmGLM, ore.odmNB, ore.odmDT, partitions

Examples
IRIS <- ore.push(iris)
ore.odmAI(Species ~ ., IRIS)

 ore.odmAssocRules  In-Database Association Rules Models

Description
Creates an association model using Oracle Data Mining.

Usage
ore.odmAssocRules(formula, 
data, 
case.id.column, 
item.id.column = NULL, 
item.value.column = NULL, 
min.support = 0.1, 
min.confidence = 0.1, 
max.rule.length = 4, 
na.action = na.pass, 
odm.settings = NULL)

## S3 method for class 'ore.odmAssocRules'
itemsets(object, ...)

Arguments
formula  An object of class formula (or one that can be coerced to that class): a sym-
           bolic description of the model to be fitted. The details of the model specification
           are given under ‘Details’.
data  An ore.frame object used for model building.
case.id.column  The name of a column in data that contains unique case identifiers.
ore.odmAssocRules

item.id.column
The name of a column in data that contains item IDs. If NULL, the model treats data as a single-record case relational table, where each row is considered a transaction and column values of that row are converted to items for that transaction; if specified, the model treats data as a transactional or multi-record case table where each row corresponds to an item in the transaction, and the model ignores any columns in data other than the item ID and the item value. The default value is NULL.

item.value.column
The name of a column in data that contains the value of the item. The default value is NULL.

min.support
A value that specifies the minimum support for the model.

min.confidence
A value that specifies the minimum confidence for the model.

max.rule.length
A value that specifies the maximum rule length for the model.

na.action
A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.

odm.settings
Same as odm.settings in ore.odmKMeans.

object
An object of type ore.odmAssocRules.

Additional arguments.

Details

This function implements the Apriori algorithm (Agrawal and Srikant 1994) to find frequent itemsets and generate association models. It finds the co-occurrence of items in large volumes of "transactional" data such as in the case of market basket analysis, as well as "relational" data. The rule is a statement that the appearance of a set of items in a transactional record implies the existence of another set of items. The groups of items used to form rules must pass a minimum threshold according to how frequently they occur (the support) and how often the consequent follows the antecedent (the confidence). Association models generate all rules that have support and confidence greater than user-specified thresholds. The Apriori algorithm is efficient, and scales well with respect to the number of transactions, number of items, and number of itemsets and rules produced.

The formula specification has the form ~ terms where terms is a series of terms, for example, the column names, to include in the analysis. Multiple terms are specified using + between column names. Use ~ . if all columns in data should be used for model building. To exclude columns, use - before each column name to exclude. Functions can be applied to terms to realize transformations.

Value

The function ore.odmAssocRules returns an object of class "ore.odmAssocRules" that has the following components:

name
The name of the in-database model.

settings
A data.frame of settings used to build the model.

attributes
A named vector of the types of input item values.

inputType
The type of input data table. It is "trans", "tranWithValue", or "relational" for a multi-record case table, a multi-record case table with the values specified, or a single-record case table, respectively.
ore.odmAssocRules

formula  A formula specified by users.
call     The matched call.

The method itemsets returns an object of class ore.itemsets that describes the property of each itemset. An ore.itemsets object has the following columns:

ITEMSET_ID  The numerical identifier associated with each itemset.
NUMBER_OF_ITEMS  The number of items in the itemset.
ITEMS  The items in the itemset.
SUPPORT  The support of the itemset.

The method rules returns an object of class ore.rules that describes the property of each rule. For details, see rules.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User’s Guide

See Also

ore.rules, ore.itemsets, rules.partitions

Examples

# Transactional data.
id <- c(1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3)
item <- c("b", "d", "e", "a", "b", "c", "e", "b", "c", "d", "e")
data.ore <- ore.push(data.frame(ID = id, ITEM = item))

# Build model with specifications.
ar.mod1 <- ore.odmAssocRules(~., data.ore, case.id.column = "ID",
item.id.column = "ITEM", min.support = 0.6, min.confidence = 0.6,
max.rule.length = 3)

# Generate itemsets and rules of the model.
itemsets <- itemsets(ar.mod1)
rules <- rules(ar.mod1)

# subsetting
sub.itemsets <- subset(itemsets, min.support=0.7, items=list("b"))
sub.rules <- subset(rules, min.confidence=0.7, lhs=list("b", "c"))

# Convert the rules to the rules object in arules package.
# The package arules is required.
library(arules, warn.conflicts=FALSE)
rules.arules <- ore.pull(rules)
In-Database Decision Tree Models

Description

Fits a classification tree using Oracle Data Mining.

Usage

```r
ore.odmDT(formula, data, auto.data.prep = TRUE,
```
ore.odmDT

cost.matrix = NULL,
impurity.metric = "gini",
max.depth = 7,
min.rec.split = 20,
min.pct.split = 0.1,
min.rec.node = 10,
min.pct.node = 0.05,
na.action = na.pass,
ode.m.settings = NULL)

## S3 method for class 'ore.odmDT'
predict(object,
    newdata,
supplemental.cols = NULL,
type = c("class","raw"),
na.action = na.pass,
bestN = NULL,
topNattrs = FALSE,...)

Arguments

formula An object of class formula (or one that can be coerced to that class): a sym-
    bolic description of the model to be fitted. The details of model specification are
given under 'Details'.
data An ore.frame object used for model building.
auto.data.prep A logical value that specifies whether Oracle Data Mining should invoke auto-
    matic data preparation for the build.
cost.matrix An optional numerical square matrix that specifies the costs for incorrectly pre-
    dicting the target values. Specifying the row and column names of the matrix
is required. The values of the names are possible target values: the row names
represent actual target values and the column names represent predicted target
values. The vectors of the row and column names must be the same. In general,
the diagonal entries of the matrix are zeros. The default value is NULL.
impurity.metric Tree impurity metric "gini" or "entropy". The default value is "gini".
max.depth The maximum depth of the tree, from root to leaf inclusive. A value in the range
of 2 to 20. The default value is 7.
min.rec.split The minimum number of cases required in a node for a further split to be possi-
    ble. The value must be positive. The default value is 20.
min.pct.split The minimum number of cases required in a node for a further split to be possible.
The value is expressed as a percentage of all rows in the training data, and
must be in the range of 0 to 20. The default value is 0.1 (0.1 percent).
min.rec.node The optional minimum number of cases required in a child node. The value
must be positive. The default value is 10.
min.pct.node The optional minimum number of cases required in a child node. The value is
expressed as a percentage of the rows in the training data, and must be in the
range of 0 to 10. The default value is 0.05 (0.05 percent).
na.action A function to use for handling missing values, either `na.pass` to allow missing values or `na.omit` to remove rows with missing values. The default value is `na.pass`.

odm.settings Same as `odm.settings` in `ore.odmKMeans`.

object An object of type `ore.odmDT`.

newdata The data used for scoring.

supplemental.cols The columns from `newdata` to include as the columns in the `ore.frame` prediction result.

type If set to "raw", provides probability for each class returned. If set to "class", the class with the maximum probability is returned. The default value is `c("class","raw")`.

bestN A positive integer that restricts the returned target classes to the specified number of those that have the highest probability.

topN.attrs boolean, positive integer, False (default). Returns the top N most influence attributes of the predicted target value for regression if `topN.attrs` is not False. Returns the top N most influence attributes of the highest probability class for classification if `topN.attrs` is not False. N is equal to the specified positive integer or 5 if `topN attrs` is True.

... Additional arguments affecting the predictions produced.

Details

The Decision Tree algorithm can be used for both binary and multiclass classification problems. The tree structure, created in the model build, is used for a series of simple tests. Each test is based on a single predictor. It is a membership test: either IN or NOT IN a list of values (categorical predictor); or LESS THAN or EQUAL TO some value (numeric predictor).

The formula specification has the form `response ~ terms` where `response` is the numeric or character response vector and `terms` is a series of terms, for example, the column names, to include in the model. Multiple terms are specified using `+` between column names. Use `response ~ .` if all columns in `data` should be used for model building. Functions can be applied to `response` and `terms` to realize transformations. To exclude columns, use `-` before each column name to exclude.

The function `predict` computes predictions based on the input data and model. Results are specified in the section on Value.

Value

The function `ore.odmDT` returns an object of class `ore.odmDT`, which includes the following components:

name The name of the in-database model.

settings A `data.frame` of settings used to build the model.

attributes A `data.frame` of attributes used to build the model. The columns include: name, type (numerical or categorical), data type, data length (size), precision and scale for numeric data, and whether the variable is the target.

distributions The target class distributions at each tree node.

nodes The node summary information. See `summary.ore.odmDT`.

formula The `formula` used for the model fitted.
The function `predict` returns an `ore.frame` with columns according to the `type` and `supplemental.cols` parameters. If `type` is "class", the result includes the most likely target class and its probability. If `type` is "raw", the result includes one column for each target class and the column values reflect the probability for that class. Both can be specified together. If `supplemental.cols` are specified, the named columns are included in the result.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User’s Guide

See Also

`ore.odmSVM`, `ore.odmGLM`, `ore.odmNB`, `partitions`

Examples

```r
m <- mtcars
m$gear <- as.factor(m$gear)
m$cyl <- as.factor(m$cyl)
m$vs <- as.factor(m$vs)
m$ID <- 1:nrow(m)
MTCARS <- ore.push(m)
row.names(MTCARS) <- MTCARS
dt.mod <- ore.odmDT(gear ~ ., MTCARS)
summary(dt.mod)
dt.res <- predict (dt.mod, MTCARS, "gear")
with(dt.res, table(gear,PREDICTION)) # generate confusion matrix
```

Description

Fits an Expectation Maximization model using Oracle Data Mining in Oracle Database 12.2 or later.
ore.odmEM(formula, data, num.centers = NULL, auto.data.prep = TRUE, na.action = na.pass, odm.settings = NULL)

## S3 method for class 'ore.odmEM'
histogram(x, data=NULL, cluster.id="all",...)

## S3 method for class 'ore.odmEM'
predict(object, newdata, supplemental.cols = NULL, type = c("class","raw"), na.action = na.pass, topN = NULL, topN.attrs = FALSE,...)

## S3 method for class 'ore.odmEM'
print(x,...)

Arguments

formula An object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of the model specification are given under 'Details'.
data An ore.frame object used for model building. This argument is not used in the histogram method.
num.centers The maximum number of clusters for a clustering model. A value greater than or equal to 1. The default value NULL and the number of clusters is determined by system.

auto.data.prep A logical value that specifies whether to perform automatic data preparation. The default value is TRUE.
a.action A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.

odm.settings Same as odm.settings in ore.odmKMeans.
cluster.id A numerical cluster ID that specifies a particular cluster shown in the histograms, or "all" to show all the clusters. The default value is "all".

object An object of type ore.odmEM.
x An object of type ore.odmEM.
newdata Data used for scoring.
supplemental.cols The columns to include in the output.
ore.odmEM

**type**
If set to "raw", provides the probability for each class returned. If set to "class", the class with the maximum probability is returned.

**topN**
A positive integer that restricts the set of predicted clusters to those that have one of the specified number of highest probability values.

**topN.attrs**
boolean, positive integer, False (default). Returns the top N most influence attributes of the predicted target value for regression if topNattrs is not False. Returns the top N most influence attributes of the highest probability class for classification if topNattrs is not False. N is equal to the specified positive integer or 5 if topNattrs is True.

... Additional arguments affecting the predictions produced.

**Details**

The Expectation Maximization algorithm is a popular probability density estimation technique. Oracle Data Mining uses EM to implement a distribution-based clustering algorithm (EM-clustering).

The formula specification has the form \( \sim \) terms where terms are the column names to include in the model. Multiple terms are specified using + between column names. Use \( \sim \) if all columns in data should be used for model building. Functions can be applied to terms to realize transformations. To exclude columns, use - before each column name to exclude.

The function histogram generates a grid of histograms with rows corresponding to clusters and columns corresponding to variables, if cluster.id="all" (default). It provides a quick way to visualize and assess differences among clusters for individual variables. Only leaf clusters (that is, those with no child cluster nodes in the hierarchy) are displayed with cluster.id="all". If cluster.id is set to the numeric ID of an individual cluster, then only the histograms of variables associated with that cluster are displayed. Both leaf and non-leaf clusters can be displayed using cluster.id.

For specific values associated with the histograms, see clusterhists.ore.odmEM.

**Value**

The function ore.odmEM produces an object of class "ore.odmEM" that includes the following components:

**name**
The name of the in-database model.

**settings**
A data.frame of settings used to build the model.

**attributes**
A data.frame of attributes used to build the model. The columns include: name, type (numerical or categorical), data type, data length (size), and precision and scale for numeric data.

**clusters**
A data.frame describing general per-cluster information.

**leaf.cluster.count**
A data.frame containing leaf clusters with support.

**taxonomy**
A data.frame describing the parent/child cluster relationship.

**centers**
A data.frame containing per cluster-attribute center (centroid) information.

**centers2**
A data.frame containing center information of leaf clusters.

**formula**
The formula used for the model fitted.

**call**
The matched call.
The function `predict` returns an `ore.frame` with columns according to the `type` and `supplemental.cols` parameters. If `type` is "class", the result includes the most likely cluster ID and its probability. If `type` is "raw", the result includes one column for each cluster ID, and the column values reflect the probability for that cluster ID. Both can be specified together. If `supplemental.cols` are specified, the named columns are included in the result.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User’s Guide

See Also

`clusterhists.ore.odmEM`, `rules.ore.odmEM`, `ore.odmKMeans`, `ore.odmOC`, `partitions`

Examples

```r
x <- rbind(matrix(rnorm(100, sd = 0.3), ncol = 2),
            matrix(rnorm(100, mean = 1, sd = 0.3), ncol = 2))
colnames(x) <- c("x", "y")

X <- ore.push(cbind(data.frame(x), part = as.integer(x[, 2L]*100)%%2L))

em.mod <- ore.odmEM(~. -part, X, num.centers = 3L)

summary(em.mod)
rules(em.mod)
clusterhists(em.mod)
histogram(em.mod)

e.m.res1 <- predict(em.mod,X,type="class",supplemental.cols=c("x","y"))
head(e.m.res1,3)
e.m.res1.local <- ore.pull(e.m.res1)
plot(data.frame(x=e.m.res1.local$x,
               y=e.m.res1.local$y),
     col=e.m.res1.local$CLUSTER_ID)
points(em.mod$centers2, col = rownames(em.mod$centers2), pch = 8, cex=2)

head(predict(em.mod,X))
head(predict(em.mod,X,type=c("class","raw"),supplemental.cols=c("x","y")),3)
head(predict(em.mod,X,type="raw",supplemental.cols=c("x","y")),3)

em.pmod <- ore.odmEM(~. , X, num.centers = 3L,
                     odm.settings = list(odms_partition_columns = "part"))
partitions(em.pmod)
summary(em.pmod)
rules(em.pmod)
clusterhists(em.pmod)
histogram(em.pmod, part = "DM$$_P1")
head(predict(em.pmod,X))
```
ORE.ODMESA

In-Database Explicit Semantic Analysis Models

Description

Creates an Explicit Semantic Analysis model for feature extraction using Oracle Data Mining in Oracle Database 12.2 or later.

Usage

ore.odmESA(formula,
data,
auto.data.prep = TRUE,
na.action = na.pass,
odm.settings = NULL,
ctx.settings = NULL)

## S3 method for class 'ore.odmESA'
features(object,...)

## S3 method for class 'ore.odmESA'
feature_compare(object,
newdata,
compare.cols = NULL,
supplemental.cols = NULL)

## S3 method for class 'ore.odmESA'
predict(object,
newdata,
supplemental.cols = NULL,
type = c("class","raw"),
na.action = na.pass,
topN = NULL,...)

Arguments

- **formula**: An object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. For details of the model specification, see ‘Details’.
- **data**: An ore.frame object used for model building.
- **auto.data.prep**: A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build.
- **na.action**: A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.
odm.settings  Same as odm.settings in ore.odmKMeans. In addition, to perform text mining, parameter CASE_ID_COLUMN_NAME must specify the name of the column containing unique identifiers. Parameters ODMS_TEXT_POLICY_NAME, ODMS_TEXT_MIN_DOCUMENTS, and ODMS_TEXT_MAX_FEATURES can be used to specify the name of a valid Oracle text policy, the minimal number of documents in which echo token occurs, and the maximum number of distinct features for text mining, respectively.

ctx.settings  Same as ctx.settings in ore.odmKMeans.

object  An object of type ore.odmESA.

newdata  Data used for scoring.

compare.cols  Columns used to compare the features. All the columns are used by default.

supplemental.cols  Columns to include in the output.

type  If set to "raw", provides the probability for each feature returned. If set to "class", the feature with the maximum probability is returned.

topN  A positive integer that restricts the set of features to those that have one of the specified number of highest probability values.

...  Additional arguments affecting the predictions produced.

Details

Explicit Semantic Analysis (ESA) is a feature extraction algorithm. ESA does not discover latent features but instead uses explicit features based on an existing knowledge base.

Explicit knowledge often exists in text form. Multiple knowledge bases are available as collections of text documents. These knowledge bases can be generic, for example, Wikipedia, or domain-specific. Data preparation transforms the text into vectors that capture attribute-concept associations.

Value

An object of class "ore.odmESA" includes the following components:

name  The name of the in-database model.

settings  A data.frame of settings used to build the model.

attributes  A data.frame of attributes used to build the model.

formula  The formula used for the model fitted.

call  The matched call.

The function features returns an ore.frame that describes each feature extracted. The columns are:

FEATURE_ID  The numeric identifier associated with the feature.

ATTRIBUTE_NAME  The name of the attribute.

ATTRIBUTE_VALUE  The value of the attribute.

COEFFICIENT  The coefficient associated with the attribute in a particular feature.

The function feature_compare returns an ore.frame containing a SIMILARITY column that measures relatedness and supplementary columns if specified.
ore.odmESA

Author(s)
Oracle
Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References
Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User’s Guide

See Also
ore.odmSVD, ore.odmNMF, partitions

Examples

title <- c('Aids in Africa: Planning for a long war',
'Mars rover maneuvers for rim shot',
'Mars express confirms presence of water at Mars south pole',
'NASA announces major Mars rover finding',
'Drug access, Asia threat in focus at AIDS summit',
'NASA Mars Odyssey THEMIS image: typical crater',
'Road blocks for Aids')

# TEXT contents in character column
df <- data.frame(CUST_ID = seq(length(title)), TITLE = title)
ESA_TEXT <- ore.push(df)

# TEXT contains in clob column
attr(df$TITLE, "ora.type") <- "clob"
ESA_TEXT_CLOB <- ore.push(df)

# create text policy (CTXSYS.CTX_DDL privilege is required)
ore.exec("Begin ctx_ddl.create_policy('ESA_TXTPOL'); End;")

# specify TEXT POLICY_NAME, MIN_DOCUMENTS, MAX_FEATURES and
# ESA algorithm settings in odm.settings
esa.mod <- ore.odmESA(~ TITLE, data = ESA_TEXT_CLOB,
odm.settings = list(case_id_column_name = "CUST_ID",
ODMS_TEXT_POLICY_NAME = "ESA_TXTPOL",
ODMS_TEXT_MIN_DOCUMENTS = 1,
ODMS_TEXT_MAX_FEATURES = 3,
ESAS_MIN_ITEMS = 1,
ESAS_VALUE_THRESHOLD = 0.0001,
ESAS_TOPN_FEATURES = 3))

class(esa.mod)
summary(esa.mod)
settings(esa.mod)
features(esa.mod)
predict(esa.mod, ESA_TEXT, type = "class", supplemental.cols = "TITLE")

# use ctx.settings to specify a character column as TEXT and
# the same above settings as well as TOKEN_TYPE
esa.mod2 <- ore.odmESA(~ TITLE, data = ESA_TEXT,
odm.settings = list(case_id_column_name = "CUST_ID", ESAS_MIN_ITEMS = 1),
Description

Fits a ESM model using Oracle Data Mining.

Usage

ore.odmESM(formula,  
data,  
auto.data.prep = TRUE,  
na.action = na.pass,  
odm.settings = NULL,  
ctx.settings = NULL)

Arguments

- **formula**: An object of class `formula` (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
- **data**: An `ore.frame` object used for model building.
- **auto.data.prep**: A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build. The default value is `TRUE`.
- **na.action**: A function to use for handling missing values, either `na.pass` to allow missing values or `na.omit` to remove rows with missing values. The default value is `na.pass`.
- **odm.settings**: Same as `odm.settings` in `ore.odmESM`.
- **ctx.settings**: Text mining settings.

Details

Exponential Smoothing Models (ESM) is a useful technique for extracting meaningful and interpretable features.

The `formula` specification has the form `attribute ~ terms` where `attribute` is the attribute to be used for model build and `terms` is a series of terms, for example, the column names, to include in the model. Multiple terms are specified using `+` between column names. Use `response ~ .` if all columns in `data` should be used for model building. To exclude columns, use `−` before each column name to exclude.
ore.odmGLM

In-Database Generalized Linear Models

Description

Fits a linear or logistic regression model using Oracle Data Mining. Supports ridge estimation of the coefficients.
ore.odmGLM

Usage

ore.odmGLM(formula, data, weights = NULL, type = c("normal", "logistic"), na.treatment = c("delete.row", "mean.or.mode"), reference = NULL, ridge = FALSE, ridge.value = NULL, ridge.vif = FALSE, auto.data.prep = FALSE, odm.settings = NULL, ctx.settings = NULL)

## S3 method for class 'ore.odmGLM'
residuals(object, type = c("deviance", "pearson", "response"), ...)

## S3 method for class 'ore.odmGLM'
fitted(object, ...)

## S3 method for class 'ore.odmGLM'
predict(object, newdata, supplemental.cols = NULL, confint = FALSE, level = 0.95, na.action = na.pass, topN.attrs = FALSE, ...)

## S3 method for class 'ore.odmGLM'
confint(object, parm, level = 0.95, ...)

## S3 method for class 'ore.odmGLM'
deviance(object, ...)

## S3 method for class 'ore.odmGLM'
extractAIC(fit, scale = 0, k = 2, ...)

## S3 method for class 'ore.odmGLM'
logLik(object, ...)

## S3 method for class 'ore.odmGLM'
nobs(object, ...)

Arguments

formula An object of class formula that describes the model to be fit. (See the ‘Details’ section for more information.)
data An ore.frame object containing the variables for the model fit.
weights An optional character string representing the column name in the data argument to use as analytical weights in the model fit.
type For ore.odmGLM, the type of generalized linear model; either "normal" (Gaussian) or "logistic" (binomial). For the residuals method, the type of residuals; either "deviance", "pearson", or "response".
na.treatment The missing value treatment; either "delete.row" (delete entire row) or...
"mean.or.mode" (replace missing values with the mean in numeric predictors and the mode in categorical predictors).

**reference**
An optional response variable category to use as the reference value (non-case/failure code) in a logistic regression model. By default, reference is taken to be the category with the highest prevalence.

**ridge**
A logical indicator for whether to use ridge estimation for the coefficients.

**ridge.value**
When the ridge argument is TRUE, an optional positive value for the ridge parameter. If ridge estimation is used and the argument ridge.value is NULL, the ridge parameter is determined by the algorithm.

**ridge.vif**
(Linear regression only) Optional logical indicator for whether to produce Variance Inflation Factor (VIF) statistics for the ridge estimates. VIFs can only be produced if enough Oracle database system resources are available.

**auto.data.prep**
A logical indicator for optional automatic data preparation.

**object, fit**
Object of type ore.odmGLM.

**newdata**
Data used for scoring.

**supplemental.cols**
A character vector containing additional columns from argument newdata to include in the prediction result.

**confint**
A logical indicator for whether to produce confidence intervals for the predicted values.

**level**
A numeric value within [0, 1] to use for the confidence level.

**na.action**
A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.

**topNattrs**
boolean, positive integer, False (default). Returns the top N most influence attributes of the predicted target value for regression if topNattrs is not False. Returns the top N most influence attributes of the highest probability class for classification if topNattrs is not False. N is equal to the specified positive integer or 5 if topNattrs is True.

**odm.settings**
Same as odm.settings in ore.odmKMeans.

**ctx.settings**
Same as ctx.settings in ore.odmKMeans.

**parm**
An optional character vector that specifies which coefficients to include in the set of confidence intervals.

**scale**
An optional numeric scale parameter.

**k**
An optional numeric weight of the equivalent degrees of freedom.

**...**
Additional arguments.

---

**Details**

ODM generalized linear models (GLM) implements Gaussian linear regression for continuous response variables and logistic regression for binary response variables using the canonical link and variance functions. The ODM GLM implementation contains two features for robust model fitting: ridge estimation of the coefficients (argument ridge) and automatic data preparation (argument auto.data.prep). When ridge estimation is enabled, a user can supply a specific ridge parameter in argument ridge.value or have an automated procedure determine a suitable value. In the case of exact multi-collinearity, if the user disables ridge estimation, a database exception is raised.
to inform the user that the data is problematic and either explanatory variables could be removed from the model or ridge estimation could be used. The automated data preparation includes normalization for numerical values, encoding as a series of indicators for categorical values, and missing data replacement.

In addition to the classical weighted least squares estimation for linear regression and iteratively reweighted least squares estimation for logistic regression, both solved through Cholesky decomposition and matrix inversion, ODM GLM provides a conjugate gradient-based optimization algorithm that does not require matrix inversion and is very well suited to high-dimensional data similar to the approach in [Komarek, 2004]. The choice of algorithm is handled internally and is transparent to the user.

Feature selection and generation can be enabled and configured by the settings in `odm.settings`. The feature selection criteria can be AIC, SBIC, RIC, or alpha-investing. The maximum number of features can be specified. Features can be pruned in the final model. Feature generation is only possible when feature selection is enabled. The feature generation method can be either quadratic or cubic. Note that feature selection and ridge regression are mutually exclusive.

The `formula` argument in function `ore.odmGLM` uses the form `response ~ terms`, where `response` is the numeric or binary category response vector and `terms` is a combination of explanatory variable terms to include in the model. Explanatory terms are separated by the `+` operator. The `.terms` wildcard is supported with `response ~ .` representing the use of all but the response column in `data` as linear terms in the model. The `:` operator can be used in combination with `~` to exclude variables from the model. Interaction effects created using the operators `.:.*` and `:` are not supported.

**Value**

The function `ore.odmGLM` returns an object of class `ore.odmGLM`, which includes the following components:

- **coefficients** A named vector of coefficients.
- **residuals** An `ore.frame` containing three types of residuals: "deviance", "pearson", and "response".
- **fitted.values** An `ore.vector` containing the fitted values.
- **rank** The numeric rank of the fitted model.
- **type** The type of model fit.
- **deviance** Minus twice the maximized log-likelihood, up to a constant.
- **aic** The same version of Akaike’s `An Information Criterion` as used by `glm`.
- **null.deviance** The deviance for the null (intercept only) model.
- **prior.weights** The weights initially supplied or 1.
- **df.residual** The residual degrees of freedom.
- **df.null** The residual degrees of freedom for the null model.
- **y** An `ore.vector` containing the response variable.
- **converged** The indicator for whether the model converged.
- **model** An `ore.frame` containing the model frame.
- **na.treatment** An indicator for how missing values were treated.
- **na.action** The number of rows with missing values that were removed.
ore.odmGLM

- **call**: The matched call.
- **formula**: The `formula` supplied.
- **terms**: The `terms` object used.
- **data**: The `data` argument.
- **nonreference**: For logistic regression, the response values that represents success.
- **ridge**: The `ridge` argument.
- **auto.data.prep**: The `auto.data.prep` argument.
- **fit.name**: The internal name for the in-database model.
- **fit.details**: The model fit details.

The `residuals` method returns an `ore.vector` containing the type of residuals specified in the `type` argument.

The `predict` method returns an `ore.frame` containing the predictions along with the columns specified by the `supplemental.cols` argument.

**Author(s)**

Oracle <oracle-r-enterprise@oracle.com>

**References**


**See Also**

`summary.ore.odmGLM`, `residuals`, `fitted`, `predict`, `confint`, `deviance`, `extractAIC`, `logLik`, `nobs`, `ore.odmSVM`, `ore.odmNB`, `ore.odmDT`, `partitions`
Examples

library(OREstats)
# Linear regression using the longley data set
LONGLEY <- ore.push(longley)
longfit1 <- ore.odmGLM(Employed ~ ., data = LONGLEY)
summ.mod1 <- summary(longfit1)

# Ridge regression using the longley data set
longfit2 <- ore.odmGLM(Employed ~ ., data = LONGLEY, ridge = TRUE,
                      ridge.vif = TRUE)
summ.mod2 <- summary(longfit2)

# Logistic regression using the infert data set
INFERT <- ore.push(infert)
infit1 <- ore.odmGLM(case ~ age+parity+education+spontaneous+induced,
                     data = INFERT, type = "logistic")

infit1

# Changing the reference value to 1
infit2 <- ore.odmGLM(case ~ age+parity+education+spontaneous+induced,
                      data = INFERT, type = "logistic", reference = 1)
infit2

# Feature selection
IRIS <- ore.push(iris)
irisfit1 <- ore.odmGLM(Sepal.Length ~ ., data = IRIS,
                        odm.settings = list(GLMS_FTR_SELECTION = "GLMS_FTR_SELECTION_ENABLE",
                                            GLMS_MAX_FEATURES = 5))
settings(irisfit1)
iris.sum1 <- summary(irisfit1)

# Feature extraction
irisfit2 <- ore.odmGLM(Sepal.Length ~ ., data = IRIS,
                        odm.settings = list(GLMS_FTR_SELECTION = "GLMS_FTR_SELECTION_ENABLE",
                                            GLMS_FTR_GENERATION = "GLMS_FTR_GENERATION_ENABLE",
                                            GLMS_MAX_FEATURES = 10))
settings(irisfit2)
iris.sum2 <- summary(irisfit2)

ore.odmKMeans

**In-Database Hierarchical K-Means Models**

**Description**

Fits a hierarchical k-Means model using Oracle Data Mining.

**Usage**

```r
ore.odmKMeans(formula, 
data, 
auto.data.prep = TRUE, 
num.centers = 10, 
block.growth = 2, 
conv.tolerance = 0.01,
```

```r
```
ore.odmKMeans

distance.function = "euclidean",
iterations = 3,
min.pct.attr.support = 0.1,
um.bins = 10,
split.criterion = "variance",
na.action = na.pass,
odm.settings = NULL,
ctx.settings = NULL)

## S3 method for class 'ore.odmKMeans'
histogram(x,
data=NULL,
cluster.id="all",...)

## S3 method for class 'ore.odmKMeans'
predict(object,
newdata,
 supplemental.cols = NULL,
type = c("class","raw"),
na.action = na.pass,
topN = NULL,
topN.attrs = FALSE,...)

## S3 method for class 'ore.odmKMeans'
print(x,...)

Arguments

formula An object of class formula (or one that can be coerced to that class): a sym-
bolic description of the model to be fitted. The details of the model specification
are given under 'Details'.
data An ore.frame object used for model building. This argument is not used in
the histogram method.
auto.data.prep A logical value that specifies whether to perform automatic data preparation.
The default value is TRUE.
num.centers The number of clusters for a clustering model. A value greater than or equal to
1. The default value is 10.
block.growth The numeric growth factor for memory to hold cluster data. The value must be
greater than 1 and less than or equal to 5. The default value is 2.
conv.tolerance The convergence tolerance setting. The value must be greater than 0 and less
than or equal to 0.5. The default value is 0.01.
distance.function A distance function, which can be "cosine", "euclidean", or "fast.cosine". The
default value is "euclidean".
iterations The number of iterations. A value in the range of 1 to 20. The default value is 3.
min.pct.attr.support The minimum percent required for attributes (variables) to appear in rules. A
value in the range 0 to 1. The default value is 0.1.
um.bins The number of histogram bins. A value greater than 0. The default value is 10.
split.criterion
A logical value that specifies whether to split clusters by "variance" or "size" for split criteria. The default value is "variance".

na.action
A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.

odm.settings
A list to specify Oracle Data Mining parameter settings. This argument is applicable to building a model in Database 12.2 or later. Each list element’s name and value refer to the parameter setting name and value, respectively. The setting value must be numeric or string. Refer to Oracle Data Mining User’s Guide for each algorithm’s valid settings.

The name of the created Oracle Database mining model is system-determined by default while the parameter MODEL_NAME can be specified to explicitly set the mining model name. Note that ORE does not manage the life cycle of an explicitly-named database mining model.

When parameter ODMS_PARTITION_COLUMNS is set to the names of the partition columns, then a partition model with sub-model in each partition is created from the input data.

ore.odmGLM, ore.odmESA, ore.odmKMeans, ore.odmNMF, ore.odmSVD, and ore.odmSVM models can be used to perform text mining. Parameters ODMS_TEXT_POLICY_NAME, ODMS_TEXT_MIN_DOCUMENTS, and ODMS_TEXT_MAX_FEATURES can specify the name of a valid Oracle text policy, the minimal number of documents in which echo token occurs, and the maximum number of distinct features for text mining, respectively. Refer to Mining Unstructured Text in Oracle Data Mining User’s Guide for details.

ctx.settings
A list to specify Oracle Text attribute-specific settings. This argument is applicable to building models in Database 12.2 or later. The name of each list element refers to the text column while the list value is a scalar string specifying the attribute-specific text transformation. The valid entries in the string include TEXT, POLICY_NAME, TOKEN_TYPE, and MAX_FEATURES. See Mining Unstructured Text in Oracle Data Mining User’s Guide for details.

cluster.id
A numerical cluster ID that specifies a particular cluster shown in the histograms, or "all" to show all the clusters. The default value is "all".

object
An object of type ore.odmKMeans.

x
An object of type ore.odmKMeans.

newdata
Data used for scoring.

supplemental.cols
The columns to include in the output.

type
If set to "raw", provides probability for each class returned. If set to "class", the class with the maximum probability is returned.

topN
A positive integer that restricts the set of predicted clusters to those that have one of the specified number of highest probability values.

topN.attrs
boolean, positive integer, False (default). Returns the top N most influence attributes of the predicted target value for regression if topN_atts is not False. Returns the top N most influence attributes of the highest probability class for classification if topN_atts is not False. N is equal to the specified positive integer or 5 if topN_atts is True.

...
Additional arguments affecting the predictions produced.
Details

The k-Means algorithm uses a distance-based similarity measure and tessellates the data space creating hierarchies. It handles large data volumes through summarization and supports sparse data. It is especially useful when the dataset has a moderate number of numerical attributes and there is a predetermined number of clusters.

The formula specification has the form \(~ terms\) where terms are the column names to include in the model. Multiple terms are specified using + between column names. Use \(~\) if all columns in data should be used for model building. Functions can be applied to terms to realize transformations. To exclude columns, use \(\sim\) before each column name to exclude.

The function \(\text{histogram}\) generates a grid of histograms with rows corresponding to clusters and columns corresponding to variables, if \(\text{cluster.id} = \"all\"\) (default). It provides a quick way to visualize and assess differences among clusters for individual variables. That is, Only leaf clusters (for example, those with no child cluster nodes in the hierarchy) are displayed with \(\text{cluster.id} = \"all\"\). If \text{cluster.id} is set to the numeric ID of an individual cluster, then only the histograms of variables associated with that cluster are displayed. Both leaf and non-leaf clusters can be displayed using \text{cluster.id}.

For specific values associated with the histograms, see \(\text{clusterhists.ore.odmKMeans}\).

Value

The function \(\text{ore.odmKmeans}\) produces an object of class \"ore.odmKMeans\" that includes the following components:

- **name**: The name of the in-database model.
- **settings**: A \text{data.frame} of settings used to build the model.
- **attributes**: A \text{data.frame} of attributes used to build the model. The columns include: name, type (numerical or categorical), data type, data length (size), and precision and scale for numeric data.
- **clusters**: A \text{data.frame} describing general per-cluster information.
- **leaf.cluster.count**: A \text{data.frame} containing leaf clusters with support.
- **taxonomy**: A \text{data.frame} describing the parent/child cluster relationship.
- **centers**: A \text{data.frame} containing per cluster-attribute center (centroid) information.
- **centers2**: A \text{data.frame} containing center information of leaf clusters.
- **formula**: The formula used for the model fitted.
- **call**: The matched call.

The function \(\text{predict}\) returns an \text{ore.frame} with columns according to the type and supplemental.cols parameters. If type is \"class\", the result includes the most likely cluster ID and its probability. If type is \"raw\", the result includes one column for each cluster ID, and the column values reflect the probability for that cluster ID. Both can be specified together. If supplemental.cols are specified, the named columns are included in the result.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>
References

Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User’s Guide

See Also

clusterhists.ore.odmKMeans, rules.ore.odmKMeans, ore.odmOC, ore.odmEM, partitions

Examples

x <- rbind(matrix(rnorm(100, sd = 0.3), ncol = 2),
           matrix(rnorm(100, mean = 1, sd = 0.3), ncol = 2))
colnames(x) <- c("x", "y")

X <- ore.push (data.frame(x))
km.mod1 <- NULL
km.mod1 <- ore.odmKMeans(~., X, num.centers=2)
km.mod1
summary(km.mod1)
rules(km.mod1)
clusterhists(km.mod1)
histogram(km.mod1)

km.res1 <- predict(km.mod1,X,type="class",supplemental.cols=c("x","y"))
head(km.res1,3)
km.res1.local <- ore.pull(km.res1)
plot(data.frame(x=km.res1.local$x,
               y=km.res1.local$y),
     col=km.res1.local$CLUSTER_ID)
points(km.mod1$centers2, col = rownames(km.mod1$centers2), pch = 8, cex=2)

head(predict(km.mod1,X))
head(predict(km.mod1,X,type=c("class","raw"),supplemental.cols=c("x","y")),3)
head(predict(km.mod1,X,type="raw",supplemental.cols=c("x","y")),3)

# Text mining with ore.odmKMeans
title <- c('Aids in Africa: Planning for a long war',
            'Mars rover maneuvers for rim shot',
            'Mars express confirms presence of water at Mars south pole',
            'NASA announces major Mars rover finding',
            'Drug access, Asia threat in focus at AIDS summit',
            'NASA Mars Odyssey THEMIS image: typical crater',
            'Road blocks for Aids')
response <- c('Aids', 'Mars', 'Mars', 'Mars', 'Aids', 'Mars', 'Aids')

# TEXT contents in character column
KM_TEXT <- ore.push(data.frame(CUST_ID = seq(length(title)),
                              RESPONSE = response, TITLE = title))

# create text policy (CTXSYS.CTX_DDL privilege is required)
ore.exec("Begin ctx_ddl.create_policy('ESA_TXTPOL'); End;")

# specify POLICY_NAME, MIN_DOCUMENTS, MAX_FEATURES and
ore.odmNB

In-Database Naive Bayes Models

Description
Fits a Naive Bayes classifier using Oracle Data Mining.

Usage
ore.odmNB(formula, 
data, 
  auto.data.prep = TRUE, 
  class.priors = NULL, 
  na.action = na.pass, 
  odm.settings = NULL)

  ## S3 method for class 'ore.odmNB'
predict(object, 
  newdata, 
  supplemental.cols = NULL, 
  type = c("class","raw"), 
  na.action = na.pass, 
  bestN = NULL, 
  topN.attrs = FALSE,...)

Arguments

formula An object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.

data An ore.frame object used for model building.

auto.data.prep A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build. The default value is TRUE.

class.priors An optional named numerical vector that specifies the priors for the target classes. The names of the vector are the possible target classes. The default value is NULL.
na.action  A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.

odm.settings  Same as odm.settings in ore.odmKMeans.

object  An object of type ore.odmNB.

newdata  The data used for scoring.

supplemental.cols  The columns from newdata to include as columns in the ore.frame prediction result.

type  If set to "raw", provides probability for each class returned. If set to "class", the class with the maximum probability is returned. The default value is c("class","raw").

bestN  A positive integer that restricts the returned target classes to the specified number of those that have the highest probability.

topN.attrs  boolean, positive integer, False (default). Returns the top N most influence attributes of the predicted target value for regression if topN.attrs is not False. Returns the top N most influence attributes of the highest probability class for classification if topN.attrs is not False. N is equal to the specified positive integer or 5 if topN.attrs is True.

...  Additional arguments affecting the predictions produced.

Details

The Naive Bayes (NB) algorithm for classification makes predictions using Bayes’ Theorem assuming that each attribute is conditionally independent of the others given a particular value of the target (Duda, Hart and Stork 2000). NB provides a flexible general classifier for fast model building and scoring that can be used for both binary and multiclass classification problems.

The formula specification has the form response ~ terms where response is the numeric or character response vector and terms is a series of terms, for example, the column names, to include in the model. Multiple terms are specified using + between column names. Use response ~ . if all columns in data should be used for model building. To exclude columns, use - before each column name to exclude.

The function predict computes predictions based on the input data and model. Results are specified in the section on Value.

Value

The ore.odmNB function returns an object of class ore.odmNB, which includes the following components:

name  The name of the in-database model.

settings  A data.frame of settings used to build the model.

attributes  A data.frame of variables (attributes) used to build the model. The columns include: name, type (numerical or categorical), data type, data length (size), precision and scale for numeric data, and whether the variable is the target.

apriori  The class distribution for the dependent variable.

tables  A list of tables, one for each predictor variable with conditional probabilities.

levels  A vector of unique target class values.

formula  The formula used for the model fitted.
The matched call.

The function predict returns an ore.frame with columns according to the type and supplemental.cols parameters. If type is "class", the result includes the most likely target class and its probability. If type is "raw", the result includes one column for each target class and the column values reflect the probability for that class. Both can be specified together. If supplemental.cols are specified, the named columns are included in the result.

Author(s)
Oracle
Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References
Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User's Guide

See Also
ore.odmSVM, ore.odmGLM, ore.odmDT, partitions

Examples

```r
# mtcars example
m <- mtcars
m$gear <- as.factor(m$gear)
m$cyl <- as.factor(m$cyl)
m$vs <- as.factor(m$vs)
m$ID <- 1:nrow(m)
MTCARS <- ore.push(m)
row.names(MTCARS) <- MTCARS

nb.mod <- ore.odmNB(gear ~ ., MTCARS)
summary(nb.mod)
nb.res <- predict (nb.mod, MTCARS,"gear")
with(nb.res, table(gear,PREDICTION)) # generate confusion matrix

# iris example
IRIS <- ore.push(iris)
species.priors <- c(setosa=0.33, versicolor=0.33, virginica=0.34)

nb.mod <- ore.odmNB(Species ~ ., IRIS, class.priors=species.priors)
summary(nb.mod)
nb.res <- predict (nb.mod, IRIS, "Species")
with(nb.res, table(Species, PREDICTION)) # generate confusion matrix
```
ore.odmNMF  In-Database Non-Negative Matrix Factorization Models

Description

Creates a Non-Negative Matrix Factorization model for feature extraction using Oracle Data Mining.

Usage

```
ore.odmNMF(formula, data, 
            auto.data.prep = TRUE, 
            num.features = NULL, 
            conv.tolerance = NULL, 
            num.iter = NULL, 
            rand.seed = NULL, 
            allow.negative.scores = FALSE, 
            na.action = na.pass, 
            odm.settings = NULL, 
            ctx.settings = NULL)
```

```
features(object,...)
## S3 method for class 'ore.odmNMF'
features(object,...)
```

```
feature_compare(object, 
                 newdata, 
                 compare.cols = NULL, 
                 supplemental.cols = NULL)
```

```
predict(object, 
         newdata, 
         supplemental.cols = NULL, 
         type = c("class", "raw"), 
         na.action = na.pass, 
         topN = NULL, 
         topN.attrs = FALSE,...)
```

Arguments

- **formula**: An object of class `formula` (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.

- **data**: An `ore.frame` object used for model building.

- **auto.data.prep**: A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build.
num.features An integer that specifies the number of features to be extracted. The default is estimated by the algorithm.

conv.tolerance A positive fractional number less than or equal to 0.5 that specifies the convergence tolerance. The default value is 0.05.

num.iter An integer number less than or equal to 500 that specifies the maximum allowed number of iterations. The default value is 50.

rand.seed A number that specifies the random seed. The default value is -1.

allow.negative.scores A logical value that specifies whether negative scores are allowed in scoring. The default value is FALSE.

na.action A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.

odm.settings Same as odm.settings in ore.odmKMeans.

ctx.settings Same as ctx.settings in ore.odmKMeans.

object An object of type ore.odmNMF.

newdata Data used for scoring.

compare.cols Columns used to compare the features. All the columns are included by default.

supplemental.cols Columns used to include in the output.

type If set to "raw", provides the probability for each feature returned. If set to "class", the feature with the maximum probability is returned.

topN A positive integer that restricts the set of features to those that have one of the specified number of highest probability values.

topN.attrs boolean, positive integer, False (default). Returns the top N most influence attributes of the predicted target value for regression if topN_attrs is not False. Returns the top N most influence attributes of the highest probability class for classification if topN_attrs is not False. N is equal to the specified positive integer or 5 if topN_attrs is True.

... Additional arguments affecting the predictions produced.

Details

Non-Negative Matrix Factorization (NMF) is a feature extraction algorithm. Each feature extracted by NMF is a linear combination of the original attribution set. Each feature has a set of non-negative coefficients, which are a measure of the weight of each attribute on the feature. If the argument allow.negative.scores is TRUE, negative coefficients are allowed.

Value

An object of class "ore.odmNMF" includes the following components:

name The name of the in-database model.

settings A data.frame of settings used to build the model.

attributes A data.frame of attributes used to build the model.

formula The formula used for the model fitted.

call The matched call.
The function `features` returns a `data.frame` to describe each feature extracted. The columns are:

- **FEATURE_ID**: The numeric identifier associated with the feature.
- **ATTRIBUTE_NAME**: The name of the attribute.
- **ATTRIBUTE_VALUE**: The value of the attribute.
- **COEFFICIENT**: The coefficient associated with the attribute in a particular feature.

The function `feature_compare` returns an `ore.frame` containing the `SIMILARITY` column and supplementary columns if specified.

**Author(s)**

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

**References**

- Oracle R Enterprise
- Oracle Data Mining Concepts
- Oracle Data Mining User’s Guide

**See Also**

`ore.odmSVD`, `ore.odmESA`, `partitions`

**Examples**

```r
library(OREstats)
training.set <- ore.push(npk[1:18, c("N","P","K")])
scoring.set <- ore.push(npk[19:24, c("N","P","K")])

nmf.mod <- ore.odmNMF(~., training.set, num.features = 3)
features(nmf.mod)
summary(nmf.mod)
predict(nmf.mod, scoring.set)
```

**ore.odmNN**

*In-Database Neural Network Models*

**Description**

Fits a neural network model using Oracle Data Mining.
ore.odmNN

Usage

ore.odmNN(formula, data, type, auto.data.prep = TRUE, na.action = na.pass, odm.settings = NULL, ctx.settings = NULL)

Arguments

formula An object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
data An ore.frame object used for model building.
type For ore.odmNN, the type of NN model: "classification" or "regression".
auto.data.prep A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build. The default value is TRUE.
na.action A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.
odm.settings Same as odm.settings in ore.odmNN.
ctx.settings Text mining settings.

details

An Neural Network (NN) algorithm is a non-linear predictive algorithm that learn through training. It attempt to emulate the processing of a biological brain. It could be used for classification or regression.

The formula specification has the form attribute ~ terms where attribute is the attribute to be used for model build and terms is a series of terms, for example, the column names, to include in the model. Multiple terms are specified using + between column names. Use response ~ . if all columns in data should be used for model building. To exclude columns, use ~ before each column name to exclude.

Value

The ore.odmNN function returns an object of class ore.odmNN, which includes the following components:

name The name of the in-database model.
settings A data.frame of settings used to build the model.
attributes A data.frame of variables (attributes) used to build the model. The columns include: name, type (numerical or categorical), data type, data length (size), precision and scale for numeric data, and whether the variable is the target.
formula The formula used for the model fitted.
nlayers Number of layers of the fitted model
nnodes Number of nodes in each layer
weight Weights of fitted model
call The matched call.
ore.odmOC

Author(s)
Oracle
Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References
Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User's Guide

See Also
ore.odmSVM, ore.odmGLM, ore.odmDT.partitions

Examples
INFERT <- ore.push(infert)
fit <- ore.odmNN(case~., data=INFERT, type='regression')
fit
summary(fit)
head(weights(fit),3)
ans <- predict(fit, INFERT)
head(ans,3)

ore.odmOC

In-Database Orthogonal Partitioning Cluster Models

Description
Creates an Orthogonal Partitioning Cluster (O-Cluster) model using Oracle Data Mining.

Usage
ore.odmOC(formula, 
data, 
num.centers = 10, 
max.buffer = 50000, 
sensitivity = 0.5, 
na.action = na.pass, 
odm.settings = NULL)

## S3 method for class 'ore.odmOC'
histogram(x, 
data=NULL, 
cluster.id="all",...)
## S3 method for class 'ore.odmOC'
predict(object,
    newdata,
    supplemental.cols = NULL,
    type = c("class", "raw"),
    na.action = na.pass,
    topN = NULL,
    topN.attrs = FALSE,...)

## S3 method for class 'ore.odmOC'
print(x,...)

### Arguments

**formula**  
An object of class `formula` (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.

**data**  
An ordered `ore.frame` object used for model building. The number of ordering columns must be 1. This argument in the method `histogram` is unused.

**num.centers**  
A number greater than or equal to 1 that specifies the number of clusters for the clustering model. The default value is 10.

**max.buffer**  
A number greater than or equal to 1 that specifies the maximum buffer size. The default value is 50,000.

**sensitivity**  
A fraction that specifies the peak density required for separating a new cluster. The fraction is related to the global uniform density. The value must be in the range 0 to 1. The default value is 0.5.

**na.action**  
A function to use for handling missing values, either `na.pass` to allow missing values or `na.omit` to remove rows with missing values. The default value is `na.pass`.

**odm.settings**  
Same as `odm.settings` in `ore.odmKMeans`.

**cluster.id**  
A numerical cluster ID that specifies a particular cluster shown in the histograms, or "all" to show all the clusters. The default value is "all".

**object**  
An object of type `ore.odmOC`.

**x**  
An object of type `ore.odmOC`.

**newdata**  
The data used for scoring.

**supplemental.cols**  
The columns to include in the output.

**type**  
If set to "raw", provides the probability for each class returned. If set to "class", the class with the maximum probability is returned.

**topN**  
A positive integer that restricts the set of predicted clusters to those that have one of the specified number of highest probability values.

**topN.attrs**  
boolean, positive integer, False (default). Returns the top N most influence attributes of the predicted target value for regression if `topN.attrs` is not False. Returns the top N most influence attributes of the highest probability class for classification if `topN.attrs` is not False. N is equal to the specified positive integer or 5 if `topN.attrs` is True.

**...**  
Additional arguments affecting the predictions produced.
Details

The O-Cluster algorithm creates a hierarchical grid-based clustering model, that is, it creates axis-parallel (orthogonal) partitions in the input attribute space. The algorithm operates recursively. The resulting hierarchical structure represents an irregular grid that tessellates the attribute space into clusters. The resulting clusters define dense areas in the attribute space.

The clusters are described by intervals along the attribute axes and the corresponding centroids and histograms. A parameter called sensitivity defines a baseline density level. Only areas with peak density above this baseline level can be identified as clusters.

The k-Means algorithm tessellates the space even when natural clusters may not exist. For example, if there is a region of uniform density, k-Means tessellates it into n clusters (where n is specified by the user). O-Cluster separates areas of high density by placing cutting planes through areas of low density. O-Cluster needs multi-modal histograms (peaks and valleys). If an area has projections with uniform or monotonically changing density, O-Cluster does not partition it.

The clusters discovered by O-Cluster are used to generate a Bayesian probability model that is then used during scoring (predict) for assigning data points to clusters. The generated probability model is a mixture model where the mixture components are represented by a product of independent normal distributions for numerical attributes and multinomial distributions for categorical attributes.

Keep the following in mind if you choose to prepare the data for O-Cluster:
1. O-Cluster does not necessarily use all the input data when it builds a model. It reads the data in batches (the default batch size is 50000). It will only read another batch if it believes, based on statistical tests, that there may still exist clusters that it has not yet uncovered.
2. Because O-Cluster may stop the model build before it reads all of the data, it is highly recommended that the data be randomized.
3. Binary attributes should be declared as categorical. O-Cluster maps categorical data to numerical values.
4. The use of Oracle Data Mining equi-width binning transformation with automated estimation of the required number of bins is highly recommended.
5. The presence of outliers can significantly impact clustering algorithms. Use a clipping transformation before binning or normalizing. Outliers with equi-width binning can prevent O-Cluster from detecting clusters. As a result, the whole population appears to fall within a single cluster.

The formula specification has the form ~ terms where terms are the column names to include in the model. Multiple terms are specified using + between column names. Use ~. if all columns in data should be used for model building. To exclude columns, use − before each column name to exclude.

Value

The function ore.odmOC returns an object of class ore.odmOC, which includes the following components:

- name
  The name of the in-database model.

- settings
  A data.frame of settings used to build the model.

- attributes
  A data.frame of attributes used to build the model.

- clusters
  A data.frame describing general per-cluster information.

- leaf.cluster.counts
  A data.frame containing leaf clusters with support.

- taxonomy
  A data.frame describing parent-child cluster relationship.

- centers
  A data.frame describing per cluster-attribute centroid information.

- centers2
  A data.frame containing center information of leaf clusters.

- histogram
  A data.frame describing per cluster-attribute histogram information.
rule A data.frame describing the clustering rules.
formula The formula used for the model fitted.
call The matched call.

The function predict returns an ore.frame with columns according to the type and supplemental.cols parameters. If type is "class", the result includes the most likely cluster ID and its probability. If type is "raw", the result includes one column for each cluster ID, and the column values reflect the probability for that cluster ID. Both can be specified together. If supplemental.cols are specified, the named columns are included in the result.

Author(s)
Oracle
Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References
B.L. Milenova and M.M. Campos, Clustering Large Databases with Numeric and Nominal Values Using Orthogonal Projection, Proceeding of the 29th VLDB Conference, Berlin, Germany (2003).
Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User's Guide

See Also
ore.odmKMeans, ore.odmOC, ore.odmEM, partitions

Examples

# a 2-dimensional example
x <- rbind(matrix(rnorm(100, mean = 4, sd = 0.3), ncol = 2),
            matrix(rnorm(100, mean = 2, sd = 0.3), ncol = 2))
colnames(x) <- c("x", "y")
X <- ore.push (data.frame(ID=1:100,x))
rownames(X) <- X$ID
oc.mod <- ore.odmOC(~., X, num.centers=2)

summary(oc.mod)
clusterhists(oc.mod)
histogram(oc.mod)
r <- rules(oc.mod)

p <- predict(oc.mod,X)
p <- predict(oc.mod,X,type=c("class","raw"))
p <- predict(oc.mod,X,type=c("class","raw"),supplemental.cols=c("x","y"))
p <- predict(oc.mod,X,type="class")
p <- predict(oc.mod,X,type="class",supplemental.cols=c("x","y"))
p <- predict(oc.mod,X,type="raw")
p <- predict(oc.mod,X,type="raw",supplemental.cols=c("x","y"))
ore.odmRAlg

Extensible R Algorithm Models

Description

Creates an Extensible R Algorithm model using Oracle Data Mining in Oracle Database 12.2 or later.

Usage

```
ore.odmRAlg(data,
            mining.function = c("classification", "regression", "clustering",
                                "feature_extraction", "attribute_importance",
                                "association"),
            formula = NULL,
            build.function,
            build.parameter = NULL,
            score.function = NULL,
            detail.function = NULL,
            detail.value = NULL,
            odm.settings = NULL)
```

```
## S3 method for class 'ore.odmRAlg'
predict(object,
         newdata,
         supplemental.cols = NULL,
         type = "class",
         na.action = na.pass,...)
```

```
## S3 method for class 'ore.odmRAlg'
summary(object,...)
```

```
## S3 method for class 'summary.ore.odmRAlg'
print(x,...)
```

Arguments

data An ore.frame object used for model building.

mining.function A scalar string to specify the type of mining function. The valid values are classification, regression, clustering, feature_extraction, attribute_importance, association.

formula An R formula or a string representing a formula in characters. This formula can be named or take the default name 'formula'. This name is used to match the argument name of the build.function to pass the specified formula. If the formula is NULL, the R build.function shall not take a formula.

build.function The name of a registered R script used to build the model. The R script uses the first argument for input data, optionally the second argument for weight numeric vector when parameter odms_row_weight_column_name is specified in
ore.odmRAlg

odm.settings, and matches the rest of the arguments by name with the values from build.parameter. The R script returns an R object representing the fit model. For clustering, the return R object must contain an attribute dm$nclus to specify the number of clusters. For feature extraction, attribute dm$nfeat shall be used to specify the number of features in the object returned.

**build.parameter**
A list containing build function parameters excluding input data and weight vector if applicable. The list element names must match with the name of build.function script input parameter names. Only scalar numeric and string are valid parameter values.

**score.function**
The name of a registered R script used to score the model. The script takes two arguments for the model and new data respectively and returns a data.frame containing prediction results. For regression, the results are predicted values. In classification, clustering, and feature exaction, the results are probabilities for each class, cluster, and feature respectively. Rows of the results match with rows of input data. For classification, the column names of the results match with the class label names. For clustering and feature extraction, the columns are arranged in the order of the cluster and feature ids.

**detail.function**
The name of a registered R script used to obtain model details and return them in a data.frame.

**detail.value**
A data.frame object used to specify the data types of the return data.frame from detail.function.

**odm.settings**
Same as odm.settings in ore.odmKMeans.

**object**
An object of type ore.odmRAlg.

**newdata**
Data used for scoring.

**supplemental.cols**
Columns to include in the output.

**type**
If set to "raw", provides the probability for each class, cluster, or feature returned. If set to "class", the one with the maximum probability is returned. The default is "class".

**na.action**
A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.

**...**
Additional arguments affecting the predictions produced.

**x**
An object of type summary.ore.odmRAlg.

Details
Extensible R Algorithm builds, scores, and views R model using the registered R scripts. It supports classification, regression, clustering, feature_extraction, attribute_importance, and association mining function.

Value
An object of class "ore.odmRAlg" includes the following components:

**name**
The name of the in-database model.
mining.function
  The type of the data mining function for the model.

details
  An ore.frame returned by the R detail.function script.

settings
  A data.frame of settings used to build the model.

attributes
  A data.frame of attributes used to build the model.

formula
  The formula used for the model fitted.

An object of class "summary.ore.odmRAlg" includes the following components:

name
  The name of the model object.

call
  The function call used to build the model with the given arguments.

settings
  A data.frame of settings used to build the model.

details
  An ore.frame returned by the R detail.function script.

The predict method executes the score.function specified for the model build and returns an ore.frame containing the predictions along with the columns specified by the supplemental.cols argument. Function predict is applicable to only classification, regression, clustering, and feature_extraction models.

Author(s)
  Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References
  Oracle R Enterprise
  Oracle Data Mining Extensible R Algorithm Model Setting

See Also
  ore.scriptCreate ore.scriptDrop ore.scriptList partitions settings

Examples

library(OREembed)
digits <- getOption("digits")
options(digits = 5L)

IRIS <- ore.push(iris)

# Regression with glm
ore.scriptCreate("glm_build", function(data, form, family) {  
  glm(formula = form, data = data, family = family))

ore.scriptCreate("glm_score", function(mod, data) {  
  res <- predict(mod, newdata = data); data.frame(res))

ore.scriptCreate("glm_detail", function(mod) {  
  data.frame(name=names(mod$coefficients), coef=mod$coefficients))

ore.scriptList(name = "glm_build")
ore.scriptList(name = "glm_score")
ore.scriptList(name = "glm_detail")

ralg.glm <- ore.odmRAlg(IRIS, mining.function = "regression",
    formula = c(form="Sepal.Length ~ ."),
    build.function = "glm_build", build.parameter = list(family="gaussian"),
    score.function = "glm_score",
    detail.function = "glm_detail", detail.value = data.frame(name="a", coef=1))

summary(ralg.glm)
predict(ralg.glm, newdata = head(IRIS), supplemental.cols = "Sepal.Length")

ore.scriptDrop(name = "glm_build")
ore.scriptDrop(name = "glm_score")
ore.scriptDrop(name = "glm_detail")

# Classification with nnet
ore.scriptCreate("nnet_build", function(dat, form, sz){
    require(nnet);
    set.seed(1234);
    nnet(formula = formula(form), data=dat,
         size=sz, linout=TRUE, trace=FALSE);
}, overwrite=TRUE)

ore.scriptCreate("nnet_score", function(mod, data) {
    require(nnet);
    res <- data.frame(predict(mod, newdata = data));
    names(res) <- sort(mod$lev); res})

ore.scriptCreate("nnet_detail", function(mod)
    data.frame(conn=mod$conn, wts=mod$wts),
    overwrite=TRUE)

ralg.nnet <- ore.odmRAlg(IRIS, mining.function = "classification",
    formula = c(form="Species ~ ."),
    build.function = "nnet_build", build.parameter = list(sz=2),
    score.function = "nnet_score",
    detail.function = "nnet_detail", detail.value = data.frame(conn=1, wts =1))

summary(ralg.nnet)
predict(ralg.nnet, newdata = head(IRIS), supplemental.cols = "Species")

ore.scriptDrop(name = "nnet_build")
ore.scriptDrop(name = "nnet_score")
ore.scriptDrop(name = "nnet_detail")

# Feature extraction with pca
ore.scriptCreate("pca_build", function(dat){
    mod <- prcomp(dat, retx = FALSE)
    attr(mod, "dm$nfeat") <- ncol(mod$rotation)
}, overwrite=TRUE)

ore.scriptCreate("pca_score", function(mod, data) {
    res <- predict(mod, data)
    as.data.frame(res), overwrite=TRUE)
ore.scriptCreate("pca_detail", function(mod) {
  rotation_t <- t(mod$rotation)
  data.frame(id = seq_along(rownames(rotation_t)), rotation_t),
  overwrite=TRUE)

X <- IRIS[, -5L]
ralg.pca <- ore.odmRAlg(X, mining.function = "feature_extraction",
  formula = NULL,
  build.function = "pca_build",
  score.function = "pca_score",
  detail.function = "pca_detail",
  detail.value = data.frame(Feature.ID=1, ore.pull(head(X,1L))))

summary(ralg.pca)
head(cbind(X, Pred = predict(ralg.pca, newdata = X)))

ore.scriptDrop(name = "pca_build")
ore.scriptDrop(name = "pca_score")
ore.scriptDrop(name = "pca_detail")

options(digits = digits)

ore.odmRF

In-Database Random Forest Models

Description

Fits a random forest model using Oracle Data Mining.

Usage

ore.odmRF(formula, 
data, 
  auto.data.prep = TRUE, 
  na.action = na.pass, 
  odm.settings = NULL, 
  ctx.settings = NULL)

Arguments

formula An object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
data An ore.frame object used for model building.
auto.data.prep A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build. The default value is TRUE.
na.action A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.
odm.settings Same as odm.settings in ore.odmRF.
ctx.settings Text mining settings.
Details

Random Forest is a popular ensemble learning technique for classification and regression. By combining the ideas of bagging and random selection of variables, the algorithm produces a collection of decision trees with controlled variance, while avoiding overfitting - a common problem for decision trees.

The formula specification has the form attribute ~ terms where attribute is the attribute to be used for model build and terms is a series of terms, for example, the column names, to include in the model. Multiple terms are specified using + between column names. Use response ~ . if all columns in data should be used for model building. To exclude columns, use - before each column name to exclude.

Value

The ore.odmRF function returns an object of class ore.odmRF, which includes the following components:

- name: The name of the in-database model.
- settings: A data.frame of settings used to build the model.
- attributes: A data.frame of variables (attributes) used to build the model. The columns include: name, type (numerical or categorical), data type, data length (size), precision and scale for numeric data, and whether the variable is the target.
- formula: The formula used for the model fitted.
- importance: Attribute importance of the fitted model
- call: The matched call.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

- Oracle R Enterprise
- Oracle Data Mining Concepts
- Oracle Data Mining User’s Guide

See Also

- ore.odmSVM, ore.odmGLM, ore.odmDT, partitions

Examples

```r
IRIS <- ore.push(iris)
mod <- ore.odmRF(Species~., IRIS)
mod
summary(mod)
importance(mod)
ans <- predict(mod, IRIS)
head(ans,3)
```
Description

Creates a Singular Value Decomposition (SVD) model for feature extraction using Oracle Data Mining in Oracle Database 12.2 or later.

Usage

ore.odmSVD(formula, data, auto.data.prep = TRUE, na.action = na.pass, odm.settings = NULL, ctx.settings = NULL)

## S3 method for class 'ore.odmSVD'
features(object, ...)

## S3 method for class 'ore.odmSVD'
feature_compare(object, newdata, compare.cols = NULL, supplemental.cols = NULL)

## S3 method for class 'ore.odmSVD'
predict(object, newdata, supplemental.cols = NULL, type = c("class", "raw"), na.action = na.pass, topN = NULL, topN.attrs = FALSE, ...)

u(object)

## S3 method for class 'ore.odmSVD'
u(object)

v(object)

## S3 method for class 'ore.odmSVD'
v(object)

d(object)

## S3 method for class 'ore.odmSVD'
d(object)

Arguments

formula An object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. For details of the model specification,
ore.odmSVD

see 'Details'.

data  An \texttt{ore.frame} object used for model building.

\texttt{auto.data.prep}  A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build.

\texttt{na.action}  A function to use for handling missing values, either \texttt{na.pass} to allow missing values or \texttt{na.omit} to remove rows with missing values. The default value is \texttt{na.pass}.

\texttt{odm.settings}  Same as \texttt{odm.settings} in \texttt{ore.odmKMeans}.

\texttt{ctx.settings}  Same as \texttt{ctx.settings} in \texttt{ore.odmKMeans}.

\texttt{object}  An object of type \texttt{ore.odmSVD}.

\texttt{newdata}  The data used for scoring.

\texttt{compare.cols}  Columns used to compare the features. All the columns are included by default.

\texttt{supplemental.cols}  Columns to include in the output.

\texttt{type}  If set to "raw", provides the probability for each feature returned. If set to "class", the feature with the maximum probability is returned.

\texttt{topN}  A positive integer that restricts the set of features to those that have one of the specified number of highest probability values.

\texttt{topN.attrs}  boolean, positive integer, False (default). Returns the top N most influence attributes of the predicted target value for regression if \texttt{topN.attrs} is not False. Returns the top N most influence attributes of the highest probability class for classification if \texttt{topN.attrs} is not False. \texttt{N} is equal to the specified positive integer or 5 if \texttt{topN.attrs} is True.

...  Additional arguments affecting the predictions produced.

\textbf{Details}

Singular Value Decomposition (SVD) is a feature extraction algorithm. SVD is orthogonal linear transformations that capture the underlying variance of the data by decomposing a rectangular matrix into three matrices: U, D, and V. Matrix D is a diagonal matrix and its singular values reflect the amount of data variance captured by the bases.

\textbf{Value}

An object of class "ore.odmSVD" includes the following components:

\texttt{name}  The name of the in-database model.

\texttt{settings}  A \texttt{data.frame} of settings used to build the model.

\texttt{attributes}  A \texttt{data.frame} of attributes used to build the model.

\texttt{formula}  The \texttt{formula} used for the model fitted.

\texttt{call}  The matched call.

The \texttt{d} function returns an \texttt{ore.frame} with the singular values of the input data. Column FEATURE_ID represents the singular value ID. The \texttt{v} function returns an \texttt{ore.frame} whose columns contain the right singular vectors. The column names match with FEATURE_ID values. The \texttt{u} function returns an \texttt{ore.frame} whose columns contain the left singular vectors. \texttt{CASE_ID_COLUMN_NAME} and \texttt{SVDS_U_MATRIX_OUTPUT} must be specified in argument \texttt{odm.settings} to produce this result.
The `features` function returns an `ore.frame` the same as the `v` function does except in an (vertical) unpivot format. The columns are:

- **FEATURE_ID**: The numeric identifier associated with the singular value.
- **ATTRIBUTE_NAME**: The name of the attribute.
- **ATTRIBUTE_VALUE**: The value of the attribute.
- **VALUE**: The coefficient value associated with the attribute in the new space.

The `feature_compare` function returns an `ore.frame` containing a SIMILARITY column and supplementary columns if specified.

**Author(s)**

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

**References**

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User’s Guide

**See Also**

`ore.odmNMF`, `ore.odmESA`, `partitions`

**Examples**

```r
IRIS <- ore.push(cbind(Id = seq_along(iris[[1L]]), iris))
svd.mod <- ore.odmSVD(~. -Id, IRIS)
summary(svd.mod)
d(svd.mod)
v(svd.mod)
head(predict(svd.mod, IRIS, supplemental.cols = "Id"))

svd.pmod <- ore.odmSVD(~. -Id, IRIS,
                        odm.settings = list(odms_partition_columns = "Species"))
summary(svd.pmod)
d(svd.pmod)
v(svd.pmod)
head(predict(svd.pmod, IRIS, supplemental.cols = "Id"))
```
In-Database Support Vector Machine Models

Description

Fits a support vector machine using Oracle Data Mining to be used for regression, classification, or anomaly detection.

Usage

```
ore.odmSVM(formula, data, type, auto.data.prep = TRUE, class.priors = NULL, active.learning = TRUE, complexity.factor = "system.determined", conv.tolerance = 0.0001, epsilon = "system.determined", cache.size = 50000000, kernel.function = "system.determined", std.dev = "system.determined", outlier.rate = 0.1, na.action = na.pass, odm.settings = NULL, ctx.settings = NULL)
```

```
## S3 method for class 'ore.odmSVM'
predict(object, newdata, supplemental.cols = NULL, type = c("class","raw"), na.action = na.pass, bestN = NULL, topN.attrs = FALSE,...)
```

```
## S3 method for class 'ore.odmSVM'
coef(object,...)
```

```
## S3 method for class 'ore.odmSVM'
print(x,...)
```

Arguments

- **formula**: An object of class `formula` (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.

- **data**: An `ore.frame` object used for model building.
type  
For `ore.odmSVM`, the type of SVM model: "classification", "regression", or "anomaly.detection". For `predict`, if set to "raw", provides probability for each class returned. If set to "class", the class with the maximum probability is returned.

bestN  
A positive integer that restricts the returned target classes to the specified number of those that have the highest probability.

topN.attrs  
boolean, positive integer, False (default). Returns the top N most influence attributes of the predicted target value for regression if topN_attrs is not False. Returns the top N most influence attributes of the highest probability class for classification if topN_attrs is not False. N is equal to the specified positive integer or 5 if topN_attrs is True.

auto.data.prep  
A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build. The default value is TRUE.

class.priors  
An optional named numerical vector that specifies the priors for the target classes. The names of the vector are the possible target classes. The default value is NULL.

active.learning  
A logical value that specifies whether to use active learning. The default value is TRUE.

complexity.factor  
The complexity factor. The value must be a positive number or "system.determined". The default value is "system.determined".

conv.tolerance  
The convergence tolerance. The value must be positive. The default value is 0.001.

epsilon  
The regularization setting for regression, similar to the complexity factor. Epsilon specifies the allowable residuals, or noise, in the data. The value must be a positive number or "system.determined". The default is "system.determined".

cache.size  
If kernel.function is "gaussian", the setting for the kernel cache size (bytes). The default value is 50,000,000.

kernel.function  
The setting for the kernel function. It can be "gaussian", "linear", or "system.determined". The default value is "system.determined".

std.dev  
If the kernel.function is "gaussian", the setting for the standard deviation for the SVM Gaussian kernel. The Value must be a positive number or "system.determined". The default value is "system.determined".

outlier.rate  
For an anomaly detection model, the setting for desired rate of outliers in the training data. A value in the range of 0 to 1. The default value is 0.1.

na.action  
A function to use for handling missing values, either `na.pass` to allow missing values or `na.omit` to remove rows with missing values. The default value is `na.pass`.

odm.settings  
Same as `odm.settings` in `ore.odmKMeans`.

ctx.settings  
Same as `ctx.settings` in `ore.odmKMeans`.

object  
An object of type `ore.odmSVM`.

x  
An object of type `ore.odmSVM`.

newdata  
The data used for prediction.
supplemental.cols

The columns from `newdata` to be included as the columns in the `ore.frame` prediction result.

... Additional arguments affecting the predictions produced.

Details

This function invokes the Oracle Data Mining SVM implementation that supports classification, regression, and anomaly detection (one-class classification) with linear or Gaussian kernels and an automatic and efficient estimation of the complexity factor (C) and standard deviation (sigma).

Support Vector Machines (SVMs) for regression utilizes an epsilon-insensitive loss function and works particularly well for high-dimensional noisy data. The implementation also supports "active learning" which forces the SVM algorithm to restrict learning to the most informative training examples and not to attempt to use the entire body of data. In most cases, the resulting models have predictive accuracy comparable to that of a standard (exact) SVM model. Active learning provides a significant improvement in both linear and Gaussian SVM models, whether for classification, regression, or anomaly detection. However, active learning is especially advantageous when using the Gaussian kernel, because nonlinear models can otherwise grow to be very large and can place considerable demands on memory and other system resources.

The SVM algorithm operates natively on numeric attributes. The function automatically "explodes" categorical data into a set of binary attributes, one per category value. For example, a character column for marital status with values married or single would be transformed to two numeric attributes: married and single. The new attributes could have the value 1 (true) or 0 (false). When there are missing values in columns, SVM interprets them as missing at random. The algorithm automatically replaces missing categorical values with the mode and missing numerical values with the mean. SVM requires the normalization of numeric input. Normalization places the values of numeric attributes on the same scale and prevents attributes with a large original scale from biasing the solution. Normalization also minimizes the likelihood of overflows and underflows. Furthermore, normalization brings the numerical attributes to the same scale (0,1) as the exploded categorical data. The SVM algorithm automatically handles missing value treatment and the transformation of categorical data, but normalization and outlier detection must be handled manually.

The formula specification has the form `response ~ terms` where `response` is the numeric or character response vector and `terms` is a series of terms, for example, the column names, to include in the model. Multiple terms are specified using `+` between column names. Use `response ~ .` if all columns in `data` should be used for model building. Functions can be applied to `response` and `terms` to realize transformations. To exclude columns, use `-` before each column name to exclude.

The function `predict` computes predictions based on the input data and model.

Value

The function `ore.odmSVM` returns an object of class `ore.odmSVM` including the following components:

- **name**: The name of the in-database model.
- **settings**: A `data.frame` of settings used to build the model.
- **attributes**: A `data.frame` of predictor columns used to build the model. The columns include: name, type (numerical or categorical), data type, data length (size), precision and scale for numeric data, and whether the variable is the target.
- **fit.values**: An `ore.frame` of the actual column and predicted column. For regression, the columns are ‘ACTUAL’ and ‘PREDICTED’. For classification, the columns
are 'ACTUAL', 'PREDICTED', 'PROBABILITY'. For anomaly detection, the columns are 'PREDICTED' and 'PROBABILITY'.

residuals For regression models, an ore.numeric vector containing the residual values (PREDICTED - ACTUAL).

formula The formula used for the symbolic description of the model fitted.

call The matched call.

If the model built uses a linear kernel, the following is additionally returned:

coefficients The coefficients of the SVM model, one for each predictor variable. If auto.data.prep is set to TRUE, these coefficients will be in the transformed space (after automatic outlier-aware normalization is applied).

The function predict returns an ore.frame with columns according to the type and supplemental.cols parameters. For a classification model, if type is "class", the result includes the most likely target class and its probability. If type is "raw", the result includes one column for each target class and the column values reflect the probability for that class. Both can be specified together. For a regression model, a column for the prediction is included in the result with name "PREDICTION". If supplemental.cols are specified, the named columns are included in the result. For an anomaly detection model, the result includes a prediction and its probability. If the prediction is 1, the case is considered typical. If the prediction is 0, the case is considered anomalous. This behavior reflects the fact that the model is trained with normal data.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References


Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User's Guide

See Also

ore.odmGLM, ore.odmNB, ore.odmDT, partitions

Examples

x <- seq(0.1, 5, by = 0.02)
y <- log(x) + rnorm(x, sd = 0.2)
dat <- ore.push(data.frame(x=x, y=y))
ore.odmXGB

# Regression
svm.mod <- ore.odmSVM(y~x,dat,"regression",kernel.function="linear")
summary(svm.mod)
coef(svm.mod)
svm.res <- predict(svm.mod,dat,supplemental.cols="x")
head(svm.res,6)

m <- mtcars
m$gear <- as.factor(m$gear)
m$cyl <- as.factor(m$cyl)
m$vs <- as.factor(m$vs)
m$ID <- 1:nrow(m)
MTCARS <- ore.push(m)

# Classification
gears.priors <- c("3" = 0.2, "4" = 0.4, "5" = 0.4)
svm.mod <- ore.odmSVM(gear ~ .-ID, MTCARS,"classification",
  class.priors=gears.priors)
summary(svm.mod)
coef(svm.mod)
svm.res <- predict (svm.mod, MTCARS,"gear")
with(svm.res, table(gear,PREDICTION)) # generate confusion matrix

# Anomaly Detection
svm.mod <- ore.odmSVM(~ .-ID, MTCARS,"anomaly.detection")
summary(svm.mod)
svm.res <- predict (svm.mod, MTCARS, "ID")
head(svm.res)
table(svm.res$PREDICTION)

ore.odmXGB

In-Database XGBoost Models

Description

Fits a XGBoost model using Oracle Data Mining to be used for regression or classification.

Usage

ore.odmXGB(formula, 
data, 
type, 
auto.data.prep = TRUE, 
na.action = na.pass, 
odm.settings = NULL, 
ctx.settings = NULL)

## S3 method for class 'ore.odmXGB'
predict(object, 
newdata, 
supplemental.cols = NULL, 
type = c("class","raw"), 
na.action = na.pass,...)
## S3 method for class 'ore.odmXGB'
importance(object,...)

## S3 method for class 'ore.odmXGB'
print(x,...)

### Arguments

- **formula**
  - An object of class `formula` (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.

- **data**
  - An `ore.frame` object used for model building.

- **type**
  - For `ore.odmXGB`, the type of XGB model: "classification" or "regression". For `predict`, if set to "raw", provides probability for each class returned. If set to "class", the class with the maximum probability is returned.

- **auto.data.prep**
  - A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build. The default value is `TRUE`.

- **na.action**
  - A function to use for handling missing values, either `na.pass` to allow missing values or `na.omit` to remove rows with missing values. The default value is `na.pass`.

- **odm.settings**
  - Same as `odm.settings` in `ore.odmXGB`.

- **ctx.settings**
  - Same as `ctx.settings` in `ore.odmXGB`.

- **object**
  - An object of type `ore.odmXGB`.

- **x**
  - An object of type `ore.odmXGB`.

- **newdata**
  - The data used for prediction.

- **supplemental.cols**
  - The columns from `newdata` to be included as the columns in the `ore.frame` prediction result.

- **...**
  - Additional arguments affecting the predictions produced.

### Details

This function invokes the Oracle Data Mining XGBoost implementation that supports classification and regression models.

XGBoost is a highly-efficient, scalable gradient tree boosting machine learning algorithm for regression and classification. The XGBoost algorithm prepares training data, builds and persists a model, and applies the model for prediction. It can be used as a stand-alone predictor or be incorporated into real-world production pipelines for a wide range of problems such as ad click-through rate prediction, hazard risk prediction, web text classification, and so on.

The `formula` specification has the form `response ~ terms` where `response` is the numeric or character response vector and `terms` is a series of terms, for example, the column names, to include in the model. Multiple terms are specified using `+` between column names. Use `response ~ .` if all columns in `data` should be used for model building. Functions can be applied to `response` and `terms` to realize transformations. To exclude columns, use `~` before each column name to exclude.

The function `predict` computes predictions based on the input data and model.
ore.odmXGB

Value

The function ore.odmXGB returns an object of class ore.odmXGB including the following components:

- **name**: The name of the in-database model.
- **settings**: A data.frame of settings used to build the model.
- **attributes**: A data.frame of predictor columns used to build the model. The columns include: name, type (numerical or categorical), data type, data length (size), precision and scale for numeric data, and whether the variable is the target.
- **fit.values**: An ore.frame of the actual column and predicted column. For regression, the columns are 'ACTUAL' and 'PREDICTED'. For classification, the columns are 'ACTUAL', 'PREDICTED', 'PROBABILITY'.
- **residuals**: For regression models, an ore.numeric vector containing the residual values (PREDICTED - ACTUAL).
- **formula**: The formula used for the symbolic description of the model fitted.
- **call**: The matched call.
- **importance**: The attribute importance of the XGB model, one for each predictor variable.

The function **predict** returns an ore.frame with columns according to the type and supplemental.cols parameters. For a classification model, if type is "class", the result includes the most likely target class and its probability. If type is "raw", the result includes one column for each target class and the column values reflect the probability for that class. Both can be specified together. For a regression model, a column for the prediction is included in the result with name "PREDICTION". If supplemental.cols are specified, the named columns are included in the result.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User's Guide

See Also

ore.odmGLM, ore.odmSVM, ore.odmDT, partitions

Examples

```r
x <- seq(0.1, 5, by = 0.02)
y <- log(x) + rnorm(x, sd = 0.2)
dat <- ore.push(data.frame(x=x, y=y))

# Regression
xgb.mod <- ore.odmXGB(y~x,dat,"regression")
summary(xgb.mod)
importance(xgb.mod)
xgb.res <- predict(xgb.mod,dat,supplemental.cols="x")
```
head(xgb.res,6)

m <- mtcars
m$gear <- as.factor(m$gear)
m$cyl <- as.factor(m$cyl)
m$vs <- as.factor(m$vs)
m$ID <- 1:nrow(m)
MTCARS <- ore.push(m)

# Classification
xgb.mod <- ore.odmXGB(gear ~ .-ID, MTCARS,"classification")
summary(xgb.mod)
importance(xgb.mod)
xgb.res <- predict(xgb.mod, MTCARS,"gear")
with(xgb.res, table(gear,PREDICTION))  # generate confusion matrix

ore.options

Oracle R Enterprise Global Options

Description
The global options that affect Oracle R Enterprise operations.

Options for Reporting
ore.trace: A logical value indicating whether iterative Oracle R Enterprise functions should
print output at each iteration. The default value for this option is FALSE.

Options for Row Ordering
ore.sep: A character string specifying the separator to use between multiple column row names
of an ore.frame. The default value for this option is "|".
ore.warn.order: A logical value indicating whether a warning should be issued when pulling
an ore.frame that lacks row names or an ore.vector that lacks element names into
memory. The default value for this option is TRUE.

Options for Server Execution
ore.parallel: A preferred degree of parallelism to use in the embedded R job; either a positive
integer greater than or equal to 2 for a specific degree of parallelism, a value of FALSE or 1
for no parallelism, a value of TRUE for the database’s default for parallelism, or NULL for the
database default for the operation. The default value for this option is NULL.

Options for Subsetting
ore.na.extract: A logical value used during logical subscripting of an ore.frame or ore.vector
object. When TRUE rows or elements with an NA logical subscript produces rows or elements
with NA values. When FALSE an NA logical subscript is interpreted as a FALSE value, result-
ing in the removal of the corresponding row or element. The default value for this option is
FALSE, whereas a value of TRUE would mimic how R treats missing value logical subscript-
ing of data.frame and vector objects.
Options for Serialization

ore.envAsEmptyenv: A logical value indicating whether referenced environments in an object should be replaced with an empty environment during serialization to an Oracle Database. When TRUE, the referenced environment in the object will be replaced with an empty environment whose parent is .GlobalEnv, and therefore, the objects in the original referenced environment will not be serialized. In some situations, this could significantly reduce the size of serialized objects. When FALSE, all the objects in the referenced environment will be serialized, and could be unserialized and loaded into memory. The default value for this option is FALSE.

Serialization is used across Oracle R Enterprise. It is used in ore.save before saving objects to a datastore. It is used in the method ore.push for a list, saving a serialized list object to the database. It is used as well in the embedded R functions for serializing parameters of list type and serializing some objects returned by an embedded R function.

Author(s)
Oracle <oracle-r-enterprise@oracle.com>

References
Oracle R Enterprise

See Also
options, ore.save, ore.datastore, ore.doEval

Examples
options()

ore.predict

Oracle R Enterprise Predictions Using R Models

Description
Generic for model predictions in Oracle R Enterprise.

Usage
ore.predict(object, newdata, ...)

Arguments

object

A model object.

newdata

An ore.frame object.

...

Optional arguments for implemented methods.

Value

Returns an object of an ore subclass.
ore.predict-glm

Oracle R Enterprise Predictions Using glm Models

Description

Oracle R Enterprise method for generating predictions using glm models.

Usage

```r
## S4 method for signature 'glm'
o.re.predict(object, newdata, type = c("link", "response"),
              se.fit = FALSE, dispersion = NULL, na.action = na.pass,
              ...)```

Arguments

- **object**: A glm model object.
- **newdata**: An ore.frame object.
- **type**: A character string specifying the type of prediction to make; either "link" (scale of the link function) or "response" (scale of the response variable).
- **se.fit**: A logical indicating whether to return the standard errors for the predictions.
- **dispersion**: The dispersion parameter to use when calculating the standard errors for the predictions.
- **na.action**: The manner in which NA values are handled, either na.omit or na.pass.
- **...**: Optional arguments.

Value

- When argument `se.fit` is FALSE, returns an ore.numeric object containing the predictions in the specified type.
- When argument `se.fit` is TRUE, returns an ore.frame object with two columns: "PRED" and "SE.PRED".

Note

Use of date/time terms in this method will result in an error.
ore.predict-kmeans

Author(s)
Oracle <oracle-r-enterprise@oracle.com>

References
Oracle R Enterprise

See Also
ore.predict, ore.predict-lm, predict.glm.

Examples

```r
infertModel <-
  glm(case ~ age + parity + education + spontaneous + induced,
      data = infert, family = binomial())
INFERT <- ore.push(infert)
INFERTpred <- ore.predict(infertModel, INFERT, type = "response",
                          se.fit = TRUE)
INFERT <- cbind(INFERT, INFERTpred)
head(INFERT)
```

ore.predict-kmeans  Oracle R Enterprise Predictions Using kmeans Models

Description
Oracle R Enterprise method for generating predictions using kmeans Models.

Usage
```r
## S4 method for signature 'kmeans'
ore.predict(object, newdata, type = c("classes", "distances"),
            na.action = na.pass, ...)
```

Arguments
- **object**  A kmeans model object.
- **newdata**  An ore.frame object.
- **type**  A character string specifying the type of prediction to make; either "classes" (cluster id) or "distances" (Euclidean distance from cluster centers).
- **na.action**  The manner in which NA values are handled, either na.omit or na.pass.
- **...**  Optional arguments.

Value
If argument type is "classes", returns an ore.integer object of cluster classifications.
If argument type is "distances", returns an ore.frame object with one column for each cluster.
ore.predict-lm

Oracle R Enterprise Predictions Using lm Models

Description

Oracle R Enterprise method for generating predictions using lm models.

Usage

```r
## S4 method for signature 'lm'
ore.predict(object, newdata, se.fit = FALSE, scale = NULL, df = Inf,
            interval = c("none", "confidence", "prediction"),
            level = 0.95, na.action = na.pass, pred.var = NULL,
            ...)  
```

Arguments

- `object`: An lm model object.
- `newdata`: An ore.frame object.
- `se.fit`: A logical indicating whether to return the standard errors for the predictions.
- `scale`: The scale parameter for standard error of the predictions.
- `df`: The degrees of freedom for the predictions when argument `scale` is not NULL.
- `interval`: The type of interval to return, either "none", "confidence", or "prediction".
- `level`: The level for argument `interval`.
- `na.action`: The manner in which NA values are handled, either na.omit or na.pass.
- `pred.var`: When argument `interval` is "prediction", the variance for a single observation.

Examples

```r
irisClusters <- kmeans(as.matrix(iris[1:4]), centers = 3)
IRIS <- ore.push(iris)
IRIS$CLUSTER <- ore.predict(irisClusters, IRIS)
IRIS <- cbind(IRIS, ore.predict(irisClusters, IRIS, type = "distances"))
head(IRIS)
table(IRIS$CLUSTER, IRIS$Species)
```
weights | When argument interval is "prediction" and argument pred.val is
NULL and object$weights is not NULL, the variance weights for the pre-
dictions as either an ore.numeric object or a one-sided model formula
referring to data within argument newdata.

... | Optional arguments.

Value

When argument se.fit is FALSE and argument interval is "none", returns an ore.numeric
object containing the predictions.

Otherwise returns an ore.frame object with up to four columns: "PRED", "SE.PRED" (when
argument se.fit is TRUE), "LOWER.CONF" and "UPPER.CONF" (when argument interval
is "confidence"), and "LOWER.PRED" and "UPPER.PRED" (when argument interval is
"prediction").

Note

Use of date/time terms in this method will result in an error.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.predict, ore.predict-glm, predict.lm.

Examples

irisModel <- lm(Sepal.Length ~ ., data = iris)
IRIS <- ore.push(iris)
IRISpred <- ore.predict(irisModel, IRIS, se.fit = TRUE,
interval = "prediction")
IRIS <- cbind(IRIS, IRISpred)
head(IRIS)
ore.predict-method

Usage

```r
## S4 method for signature 'matrix'
ore.predict(object, newdata, type = c("classes", "distances"),
    method = "euclidean", p = 2, na.action = na.pass, ...)
```

Arguments

- `object`: A `matrix` object with no more than 1000 rows.
- `newdata`: An `ore.frame` object.
- `type`: A character string specifying the type of prediction to make; either "classes" (row id) or "distances".
- `method`: A character string specifying the distance measure to use; either "euclidean", "maximum", "manhattan", "canberra", or "minkowski". See function `dist` for further explanations.
- `p`: The power of the Minkowski distance when argument `method` is "minkowski".
- `na.action`: The manner in which `NA` values are handled, either `na.omit` or `na.pass`.
- `...`: Optional arguments.

Value

If argument `type` is "classes", returns an `ore.integer` object of row number references to argument `object`.

If argument `type` is "distances", returns an `ore.frame` object with one column for each row in argument `object`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore.predict`, `ore.predict-kmeans`, `dist`.

Examples

```r
groups <- cutree(hclust(dist(iris[1:4], "manhattan")), 3)
centers <- do.call(rbind, lapply(split(iris[1:4], groups), colMeans))
rownames(cents) <- sprintf("DISTANCE%d", 1:3)
IRIS <- ore.push(iris)
IRIS$CLUSTER <- ore.predict(cents, IRIS, method = "manhattan")
IRIS <- cbind(IRIS, ore.predict(cents, IRIS, type = "distances",
    method = "manhattan"))
head(IRIS)
table(IRIS$CLUSTER, IRIS$Species)
```
Description

Oracle R Enterprise method for generating predictions using \texttt{multinom} models.

Usage

\begin{verbatim}
## S4 method for signature 'multinom'
ore.predict(object, newdata, type = c("class", "probs"),
    na.action = na.pass, ...)
\end{verbatim}

Arguments

- \texttt{object} \hspace{1em} An \texttt{multinom} model object.
- \texttt{newdata} \hspace{1em} An \texttt{ore.frame} object.
- \texttt{type} \hspace{1em} A character string specifying the type of prediction to make; either "class" or "probs".
- \texttt{na.action} \hspace{1em} The manner in which \texttt{NA} values are handled, either \texttt{na.omit} or \texttt{na.pass}.
- \texttt{...} \hspace{1em} Optional arguments.

Value

Returns an object of an \texttt{ore} subclass.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

\texttt{ore.predict}, \texttt{predict.multinom}

Examples

\begin{verbatim}
library(nnet)
model <- multinom(Species ~ ., data = iris)
IRIS <- ore.push(iris)
IRIS <- cbind(IRIS, ore.predict(model, IRIS, type = "probs"))
head(IRIS)
\end{verbatim}
Description

Oracle R Enterprise method for generating predictions using `nnet` models.

Usage

```r
## S4 method for signature 'nnet.formula'
ore.predict(object, newdata, type = c("raw", "class"),
            na.action = na.pass, ...)
```

Arguments

- `object` An `nnet` model object.
- `newdata` An `ore.frame` object.
- `type` A character string specifying the type of prediction to make; either "raw" or "class".
- `na.action` The manner in which NA values are handled, either `na.omit` or `na.pass`.
- `...` Optional arguments.

Value

Returns an object of an `ore` subclass.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore.predict`, `predict.nnet`

Examples

```r
library(nnet)
model <- nnet(Species ~ ., data = iris, size = 2, rang = 0.1,
              decay = 5e-4, maxit = 200, trace = FALSE)
IRIS <- ore.push(iris)
IRIS <- cbind(IRIS, ore.predict(model, IRIS))
```
Description

Oracle R Enterprise method for generating predictions using ore.model models.

Usage

```r
## S4 method for signature 'ore.model'
ore.predict(object, newdata, ...)
```

Arguments

- `object` An ore.model model object.
- `newdata` An ore.frame object.
- `...` Optional arguments for implemented methods.

Value

Returns an object of an ore subclass.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.predict

Examples

```r
library(OREmodels)
IRIS <- ore.push(iris)
IRISModel <- ore.lm(Sepal.Length ~ ., data = IRIS)
IRIS$PRED <- ore.predict(IRISModel, IRIS)
head(IRIS)
```
ore.predict-prcomp  *Oracle R Enterprise Predictions Using prcomp Models*

Description

Oracle R Enterprise method for generating predictions using prcomp models.

Usage

```r
### S4 method for signature 'prcomp'
ore.predict(object, newdata, ...)
```

Arguments

- `object`: A `prcomp` object.
- `newdata`: An `ore.frame` object.
- `...`: Optional arguments.

Value

Returns an `ore.frame` object containing the rotated columns of `newdata`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore.predict.prcomp`

Examples

```r
irisModel <- prcomp(~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width, data = iris)
IRIS <- ore.push(iris)
IRIS <- cbind(IRIS, ore.predict(irisModel, IRIS))
```
Description

Oracle R Enterprise method for generating predictions using `princomp` models.

Usage

```r
## S4 method for signature 'princomp'
ore.predict(object, newdata, ...)
```

Arguments

- `object`: A `princomp` object.
- `newdata`: An `ore.frame` object.
- `...`: Optional arguments.

Value

Returns an `ore.frame` object containing the rotated columns of `newdata`.

Author(s)

Oracle &lt;oracle-r-enterprise@oracle.com&gt;

References

Oracle R Enterprise

See Also

`ore.predict.princomp`

Examples

```r
irisModel <- princomp(~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width, data = iris)
IRIS <- ore.push(iris)
IRIS <- cbind(IRIS, ore.predict(irisModel, IRIS))
```
### ore.predict-rpart

**Oracle R Enterprise Predictions Using rpart Models**

**Description**

Oracle R Enterprise method for generating predictions using `rpart` models.

**Usage**

```r
## S4 method for signature 'rpart'
ore.predict(object, newdata, type = c("vector", "prob", "class", "matrix"), na.action = na.pass, ...)
```

**Arguments**

- `object`: An `rpart` model object.
- `newdata`: An `ore.frame` object.
- `type`: A character string specifying the type of prediction to make; either "vector", "prob", "class", or "matrix".
- `na.action`: The manner in which NA values are handled, either `na.omit` or `na.pass`.
- `...`: Optional arguments.

**Value**

Returns an object of an `ore` subclass.

**Author(s)**

Oracle <oracle-r-enterprise@oracle.com>

**References**

Oracle R Enterprise

**See Also**

`ore.predict`, `predict.rpart`.

**Examples**

```r
library(rpart)
model <- rpart(Kyphosis ~ ., data = kyphosis)
KYPHOSIS <- ore.push(kyphosis)
KYPHOSIS <- cbind(KYPHOSIS, ore.predict(model, KYPHOSIS))
head(KYPHOSIS)
```
**Description**

Pulls or pushes data between an R session and an Oracle R Enterprise schema.

**Usage**

```r
ore.pull(x, ...)  
ore.push(x, ...)  
```

**Arguments**

- `x` For function `ore.pull`, `ore` objects. For function `ore.push`, vector, `data.frame`, `matrix`, or `list` objects.
- `...` For future expansion.

**Details**

Functions `ore.push` and `ore.pull` place data in or retrieve data from an Oracle R Enterprise schema. The supported in-memory R object to `ore` object mappings are as follows:

- logical maps to `ore.logical`
- integer maps to `ore.integer`
- numeric maps to `ore.numeric`
- character maps to `ore.character`
- factor maps to `ore.factor`
- raw maps to `ore.raw`
- Date maps to `ore.date`
- POSIXt maps to `ore.datetime`
- difftime maps to `ore.difftime`
- `data.frame` with columns of type logical, integer, numeric, character, factor, list of raw, Date, POSIXct, difftime maps to `ore.frame` with the appropriately typed columns

In addition to the mappings above, for function `ore.push` if the input is a `list` object, `ore.push` is applied recursively to each element. Meanwhile, the list elements are serialized before saving to the database. The argument `envAsEmptyenv` in `...` is a logical value indicating whether referenced environments in the list element objects to be saved should be replaced with an empty environment during serialization. When `TRUE`, the referenced environment will be replaced with an empty environment whose parent is `.GlobalEnv`, and therefore, the contents in the referenced environment will not be serialized and saved to the database. In some situations, this could significantly reduce the size of the saved objects. When `FALSE`, the contents in the referenced environment will be serialized and saved, and could be unserialized and loaded into memory when calling `ore.pull`. The default value is regulated by the global option `ore.envAsEmptyenv`.

For in-memory R types listed in the table above, the `ore.push` function creates objects that maintain vector element or data set row ordering of the original data object. For all other data types the behavior of `ore.push` is unspecified.
Attribute `ora.type` can be used to specify mapping R types `character` and `factor` to CLOB and R type `raw` to BLOB. The `ora.type` takes the string value "clob" and "blob" respectively for these two cases.

**Value**

Function `ore.pull` returns an in-memory R object containing the appropriate in-memory data. Function `ore.push` returns an `ore` object of the appropriate type.

**Examples**

```r
vec <- 1:10
oreVec <- ore.push(vec)
class(oreVec)
vec2 <- ore.pull(oreVec)
class(vec2)
oreVec3 <- ore.push(oreVec)
class(oreVec3)
vec
oreVec
vec2
oreVec3

IRIS <- ore.push(iris)
class(IRIS)
new.iris <- ore.pull(IRIS)
class(new.iris)
head(IRIS)
head(new.iris)

vraw <- raw(2000L)
oreRaw <- ore.push(vraw)
class(oreRaw)
new.vraw <- ore.pull(oreRaw)
class(new.vraw)
length(new.vraw)

vbraw <- raw(3000L)
attr(vbraw, "ora.type") <- "blob"
oreBRaw <- ore.push(vbraw)
class(oreBRaw)
new.vbraw <- ore.pull(oreBRaw)
class(new.vbraw)
length(new.vbraw)
```
attr(iris$Species, "ora.type") <- "clob"
iris$Species.raw <- lapply(iris$Species,
    function(x) charToRaw(as.character(x)))
attr(iris$Species.raw, "ora.type") <- "blob"
IRIS2 <- ore.push(iris)
class(IRIS2)
new.iris2 <- ore.pull(IRIS2)
class(new.iris2)

---

**ore.rank**  
*Oracle R Enterprise Data Ranking*

**Description**

Enables the investigation of the distribution of values along numeric columns in an *ore.frame* object. Highlights include: Allows ranking within groups, Partitions observations into groups based on rank tiles, Provides options for treatment of ties, Calculates cumulative percentages and percentiles, Calculates normal scores from ranks.

**Usage**

```r
ore.rank(data, var, desc = FALSE, groups = NULL, group.by = NULL,  
ties = c("mean", "high", "low", "dense", "condense"),  
score = c("none", "fraction", "nplus1", "blom", "tukey",  
          "vw", "percent", "savage", "waerden", "fn1", "n1"),  
fraction = FALSE, percent = FALSE, nplus1 = FALSE,  
savage = FALSE, blom = FALSE, tukey = FALSE, vw = FALSE)
```

**Arguments**

- **data**
  - An *ore.frame* object.
- **var**
  - A comma-separated character string specifying the names of numeric columns within argument *data*.
- **desc**
  - A logical value indicating whether to rank in ascending or descending order.
- **groups**
  - An optional numeric value specifying the number of partitions in the data. For percentiles, specify `groups = 100`. For deciles, specify `groups = 10`. For quartiles, specify `groups = 4`.
- **group.by**
  - An optional character vector specifying the group by column names within argument *data*.
- **ties**
  - A character string specifying how to handle ties; One of "low" (smallest rank within the tied group), "high" (largest rank within the tied group), "mean" (average rank within the tied group), and "dense"/"condense" (arbitrary unique rank within the tied group).
- **score**
  - A character string specifying a score; One of "none", "fraction", "nplus1", "blom", "tukey", "vw", "percent", or "savage", "waerden", "fn1", or "n1".
**ore.rank**

fraction  A logical value indicating whether to compute the ratio of ‘rank/#non-missing values’ for each column in argument var.

percent   A logical value indicating whether to compute the ratio ‘(rank * 100)/#non-missing values’ for each column in argument var.

nplus1    A logical value indicating whether to compute the ratio ‘rank/(#non-missing values + 1)’ for each column in argument var.

savage    Equivalent to score = "savage".

blom      Equivalent to score = "blom".

tukey     Equivalent to score = "tukey".

vw        Equivalent to score = "vw".

**Value**

Returns an ore.frame object.

**Author(s)**

Oracle <oracle-r-enterprise@oracle.com>

**References**

Oracle R Enterprise

**See Also**

ore.sort

**Examples**

```r
IRIS <- ore.push(iris)

# Rank 2 columns with column aliases and sort them in descending rank order
ore.rank(data = IRIS,
         var = "Petal.Length=Col1Rank, Sepal.Length=Col2Rank",
         desc = TRUE)

# Handling of ties
ore.rank(data = IRIS,
         var = "Petal.Length=PetallengthRanks, Sepal.Length=SepalRanks",
         ties = "low")

# Rank within each Species group
ore.rank(data = IRIS,
         var = "Petal.Length=PetallengthRanks, Sepal.Length=SepalRanks",
         group.by = "Species")

# Partition rows into 10 groups to get deciles
ore.rank(data = IRIS,
         var = "Petal.Length=PetallengthRanks, Sepal.Length=SepalRanks",
         groups = 10)

# Estimate the cumulative distribution function
ore.rank(data = IRIS,
         var = "Petal.Length=PetallengthRanks, Sepal.Length=SepalRanks",
```
# Calculate scores
ore.rank(data = IRIS,
    var = "Petal.Length=PetalLengthRanks, Sepal.Length=SepalRanks",
    score = "savage", groups = 100, group.by = "Species")

---

## Description

The `ore.raw` class represents raw data columns in Oracle R Enterprise.

## Raw Data Methods

**ore.pull** `signature(x="ore.raw")`: Returns an R `raw` object that contains the binary fetched from `x`

## Note

See the corresponding R documentation for the functions listed above.

## Author(s)

Oracle <oracle-r-enterprise@oracle.com>

## References

Oracle R Enterprise

## See Also

`ore, ore.datetime, ore.character, ore.factor, ore.frame, ore.logical, ore.matrix, ore.number, ore.vector`

## Examples

```r
val <- ore.push(charToRaw('ABCDE'))
class(val) # ore.raw

val2 <- ore.pull(val)
class(val2) # raw
```
Recodes the values in an `ore.vector` object.

**Usage**

```r
ore.recode(x, old, new, default = NULL)
```

**Arguments**

- `x`:
  - An `ore.vector` object.
- `old`:
  - An R vector specifying the old values in `x`.
- `new`:
  - Either an R vector of the same length as argument `old` or an R matrix with the same number of rows as argument `old` specifying the new values.
- `default`:
  - A single value to use for the non-matched elements in argument `old`. If `NULL`, non-matched elements are converted to `NA` values.

**Value**

If argument `new` is an R vector, returns an `ore.vector` object representing a recoded vector.

If argument `new` is a R matrix, returns an `ore.frame` object where each column of argument `new` generates a corresponding recoded column in the return value.

**Author(s)**

Oracle <oracle-r-enterprise@oracle.com>

**References**

Oracle R Enterprise

**See Also**

`ore.vector`, `ore.hash`

**Examples**

```r
vec <- ore.push(c("a", "b", NA, "a", "c", NA, "e"))
ore.recode(vec, c("a", "b", "c"), c("able", "baker", "charlie"))
ore.recode(vec, c("a", "b", "c"),
    cbind(lower = c("able", "baker", "charlie"),
    UPPER = c("ABLE", "BAKER", "CHARLIE")))
```
ore.rm

**Oracle R Enterprise Object Removal Function**

**Description**

Removes `ore.frame` objects, representing database tables and views, from the R environment for a schema in the Oracle R Enterprise session. Corresponding data within the database are not affected.

**Usage**

```r
ore.rm(list = character(0L), schema)
```

**Arguments**

- `list`: A character vector naming `ore.frame` objects to be removed from the attached schema.
- `schema`: A character string specifying the database schema name.

**Details**

Function `ore.rm` removes the specified objects in argument `list` from the specified argument `schema`. If argument `schema` is unspecified, the default schema - the one specified at connection time for the Oracle R Enterprise session - is used.

**Author(s)**

Oracle <oracle-r-enterprise@oracle.com>

**References**

Oracle R Enterprise

**See Also**

`ore.attach`, `ore.connect`, `ore.exists`, `ore.get`, `ore.ls`, `ore.sync`

**Examples**

```r
if (!interactive())
{
  ore.create(iris, table="IRIS_TABLE")
  ore.exists("IRIS_TABLE")
  ore.rm("IRIS_TABLE")
  ore.exists("IRIS_TABLE")
  ore.sync()
  ore.exists("IRIS_TABLE")

  ore.drop("IRIS_TABLE")  # clean up
}
```
Description

Performs rolling univariate aggregation methods for `ore.number` vectors.

Usage

```r
ore.rollmax(x, k, align = c("center", "left", "right"),
             na.rm = FALSE, ...)
ore.rollmin(x, k, align = c("center", "left", "right"),
             na.rm = FALSE, ...)
ore.rollsum(x, k, align = c("center", "left", "right"),
             na.rm = FALSE, ...)
ore.rollmean(x, k, align = c("center", "left", "right"),
              na.rm = FALSE, ...)
ore.rollsd(x, k, align = c("center", "left", "right"),
           na.rm = FALSE, ...)
ore.rollvar(x, k, align = c("center", "left", "right"),
            na.rm = FALSE, ...)
```

Arguments

- `x` An ordered `ore.vector` object for `ore.rollmax` and `ore.rollmin`; an ordered `ore.number` object otherwise.
- `k` A positive integer width for the rolling window.
- `align` A character string specifying the type of alignment; either "center", "left", or "right" justified.
- `na.rm` A logical value indicating whether NA values should be ignored during the calculations.
- `...` Additional arguments.

Details

These rolling window aggregate functions take an ordered `ore.vector` or `ore.number` object and create moving windows using one of the the following `align` types:

- "center" ceiling((k-1)/2) preceding rows, the current row, and floor((k-1)/2) following rows
- "left" the current row and the k-1 rows that follow
- "right" the current row and the k-1 preceding rows

These moving windows produce smaller widths at the beginning and ending of the series. For example if argument k is 5 and argument align is "center", then the first aggregate uses subset `x[1:3]`, the second aggregate subset `x[1:4]`, and the third aggregate subset `x[1:5]`. 
Value

For `ore.rollmax` and `ore.rollmin`, returns an `ore.vector` object of the same type as argument `x`. Otherwise returns an `ore.numeric` object. The output object always has the same length as argument `x`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

one.number-univar, ore.vector-aggregate

Examples

```r
NHTEMP <- ore.frame(year = 1912:1971, temp = as.vector(nhtemp))
rownames(NHTEMP) <- NHTEMP$year
NHTEMP$rollmean7 <- ore.rollmean(NHTEMP$temp, k = 7)
NHTEMP$rollsd7 <- ore.rollsd(NHTEMP$temp, k = 7)
head(NHTEMP, 10)
```

ore.rules-class

Class `ore.rules`

Description

The `ore.rules` class represents a set of rules of an `ore.odmAssocRules` object in an Oracle Database instance. An `ore.odmAssocRules` object represents an Association model created by Oracle Data Mining.

Subsetting

In the code snippets below, argument `x` is an `ore.rules` object.

```r
subset(x, topN=NULL, min.support=NULL, min.confidence=NULL, min.lift=NULL, max.rule.length=NULL, min.rule.length=NULL, rule.id=NULL, orderby=NULL, decreasing=TRUE, lhs=NULL, rhs=NULL)
```

Returns a new `ore.rules` object using:

- `topN` The top number of rules to return. If `topN` has been specified in the object `x`, the `subset` method does not work for this object, and will report an error.
- `min.support` The minimum support of the rules to return.
- `min.confidence` The minimum confidence of the rules to return.
- `min.lift` The minimum lift of the rules to return.
- `max.rule.length` The maximum length of the rules to return. The length of a rule is defined as the total number of the items in a rule, including both `lhs` and `rhs`.
- `min.rule.length` The minimum length of the rules to return.
- `rule.id` The rule ID (numeric type) of the rule to return. If a `rule.id` is specified, the other filtering parameters are ignored.
ore.rules-class

orderby A vector of column names to be ordered. The vector should be a subset of the set
["confidence", "support", "lift", "number_of_items"]). By default, the
rules are sorted by confidence in descending order and then by support in descending
order.

decreasing A vector of logical variables that indicates whether the corresponding column of
the orderby vector is sorted in descending order. The length of this vector should be
the same as the length of orderby, with an exception of length 1, which indicates that
all the columns in orderby are sorted in the same order.

lhs A list of items. This method returns all rules in which the antecedent contains one or
more of the specified items. For example, lhs=list("apple", "orange") or
lhs=list(age=31, occupation="engineer").

rhs A list of items. This method returns all rules in which the consequent contains one or
more of the specified items.

Utilities

In the code snippets below, argument x is an ore.rules object.

ore.pull(x): Returns the in-memory counterpart object of the ore.rules object, the rules
object defined in the arules package.

Details

The ore.rules class represents a set of rules in an Oracle Database. The ore.rules object is
the result of an Association Rules model.

The column names of the ore.rules object are "RULE_ID", "NUMBER_OF_ITEMS", "LHS",
"RHS", "SUPPORT", "CONFIDENCE", "LIFT".

You can pull rules into memory in a local R session from an Oracle Database instance by using
ore.pull. The in-memory object is of class rules, which is defined in the arules package.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User’s Guide

See Also

ore.itemsets, ore.odmAssocRules, rules

Examples

id <- c(1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3)
item <- c("b", "d", "e", "a", "b", "c", "e", "b", "c", "d", "e")
data.ore <- ore.push(data.frame(ID = id, ITEM = item))
ar.mod <- ore.odmAssocRules(~., data.ore, case.id.column = "ID",
item.id.column = "ITEM", min.support = 0.6, min.confidence = 0.6,
max.rule.length = 3)
rules <- rules(ar.mod)

sub.rules1 <- subset(rules, min.confidence=0.7, lhs=list("b", "c"))
sub.rules2 <- subset(sub.rules1, max.rule.length=2)

library(arules, warn.conflicts=FALSE)
rules.arules <- ore.pull(rules)
inspect(rules.arules[1:3])
rules.arules1 <- ore.pull(sub.rules1)
rules.arules2 <- ore.pull(sub.rules2)

---

**ore.save**

*Oracle R Enterprise Datastore Saving Function*

**Description**

Saves R objects into a datastore in the user’s Oracle Database schema.

**Usage**

```r
ore.save(..., list = character(0),
        name = stop("parameter 'name' must be specified"),
        grantable = FALSE,
        envir = parent.frame(), overwrite = FALSE, append = FALSE,
        description = character(0),
        envAsEmptyenv = getOption("ore.envAsEmptyenv", FALSE))
```

**Arguments**

- `...` The names of the R objects to be saved (as symbols or character strings).
- `list` A character vector containing the names of R objects to be saved.
- `name` A character string specifying the name of datastore in which to save the R objects.
- `grantable` A scalar logical value specifying whether to create a new datastore the read privilege for which can be granted to other users. Argument `grantable` is ignored when used with argument `overwrite` or `append`.
- `envir` An environment to search for the R objects to be saved.
- `overwrite` A logical value specifying whether to overwrite the datastore, if it already exists.
- `append` A logical value specifying whether to append the R objects to the datastore, if it already exists.
- `description` A character string containing no more than 2000 characters to be used as comments for the datastore.
- `envAsEmptyenv` A logical value indicating whether referenced environments in the R objects to be saved should be replaced with an empty environment during serialization. When `TRUE`, the referenced environment in the objects will be replaced with an empty environment whose parent is `.GlobalEnv`, and therefore, the contents in the original referenced environment will not be serialized and saved to the database. In some situations, this could significantly reduce the size of
the saved objects. When FALSE, the contents in the referenced environment will be serialized and saved, and could be unserialized and loaded into memory when calling ore.load. The default value is regulated by the global option ore.envAsEmptyenv.

Details

Function ore.save saves the R objects from argument ... or itemized in argument list for retrieval from environment envir to the datastore specified in argument name of the user’s Oracle Database schema. Nonexistent R objects referenced in argument list are ignored. If no R objects given in argument list exist in environment envir, the function will throw an error.

By default, R objects are saved to a new datastore with the specified name. Arguments overwrite and append can be used to save R objects to an existing datastore, but only one of them can be TRUE.

Comments can be added to datastore through argument description.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.load, ore.move, ore.delete, ore.datastore, ore.datastoreSummary, ore.options

Examples

if (any(sapply(c("x", "y", "z"), exists)))
  stop("object x, y, or z exists")

x <- stats::runif(20)
y <- list(a = 1, b = TRUE, c = "oops")
z <- ore.push(x)

# save all objects in the current workspace environment to
# a datastore with name 'rq$ds_1' in the user's schema
ore.save(list=ls(), name="rq$ds_1")

# overwrite existing datastore rq$ds_1 with object x, y in
# the current workspace environment
ore.save(x, y, name="rq$ds_1", overwrite=TRUE)

# add object z in the current workspace environment to
# the existing datastore rq$ds_1
ore.save(z, name="rq$ds_1", append=TRUE)

ore.delete(name="rq$ds_1")
rm(x, y, z)
ore.scriptCreate

Oracle R Enterprise Embedded R Script Creation, List, Load, Drop Functions

Description
Creates an R script, which contains a single function definition, in the Oracle Database R script repository, or lists, loads, or drops an R script from the repository.

Usage
ore.scriptCreate(name, FUN, global = FALSE, overwrite = FALSE)
ore.scriptList(name = NULL, pattern = NULL, type = c("user", "global", "system", "all", "grant", "granted"))
ore.scriptLoad(name = NULL, pattern = NULL, owner = NULL, newname = NULL, envir = parent.frame())
ore.scriptDrop(name = NULL, pattern = NULL, global = FALSE, silent = FALSE)

Arguments

name A character string that specifies the name of an R script in the R script repository; cannot be used with argument pattern.

FUN A function definition to be used with functions ore.doEval, ore.groupApply, ore.indexApply, ore.rowApply, or ore.tableApply. The function cannot recursively call these embedded R APIs.

pattern An optional regular expression character string specifying the matching R script names; cannot be used with argument name.

global An optional logical value indicating whether to create or drop a global R script. The default value is FALSE, which indicates the R script to create or drop is a private script. Every user has read access to a global R script while only the user who created the R script can access and drop a private R script. Functions ore.grant and ore.revoke can be used to grant or revoke the read privilege for a private R script to other users.

overwrite A logical value specifying whether to overwrite the named R script, if it already exists.

type A scalar character string specifying the type of R script to list. The valid value is 'user' (default), 'all', 'grant', 'granted', or 'global'. 'user' lists R scripts created by current session user. 'grant' lists R scripts which has been granted by the current session user to other users. 'granted' lists R scripts the read privilege for which has been granted by other users to the current session user. 'global' lists all user-created global R scripts. 'system' lists all system predefined R scripts. 'all' lists all R scripts to which the current session user has read access.

owner An optional character string specifying the user who created the named R script. Argument owner can be used along with argument name to specify which R script to load into an R environment. Without the owner argument, ore.scriptLoad finds and loads the R script that matches name in the following order: R script that the current session user created, global R script.
newname | An optional character string specifying a new name in the current session for the R function that is loaded by `ore.scriptLoad`. The `newname` value must be a valid R function name. By default, the name of the R function takes the name of the script. Cannot be used with argument `pattern`.

envir | An optional R environment variable that specifies the environment that contains the R function into which `ore.scriptLoad` loads the named R script.

silent | An optional logical value indicating whether to display an error message when `ore.scriptDrop` encounters an error in dropping the named R script.

Details

Functions `ore.scriptCreate`, `ore.scriptDrop` require the user to have the ‘RQADMIN’ Oracle Database role.

Value

Function `ore.scriptCreate` returns an invisible `NULL` value if it succeeds in creating an R script; otherwise it produces an error. Function `ore.scriptLoad` returns an invisible `NULL` value if argument `name` is used and it succeeds in loading an R script, or, if argument `pattern` is used, it returns a list of script names that have been successfully loaded; otherwise it produces an error or a warning message. Function `ore.scriptDrop` returns an invisible `NULL` value if argument `name` is used and it succeeds in dropping an R script, or if argument `pattern` is used, it returns a list of script names that have been successfully dropped; otherwise it produces an error. Function `ore.scriptList` returns a data frame that contains the columns `NAME` and `SCRIPT`, and optionally the columns `OWNER` and `GRANTEE`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore.doEval`

Examples

```r
if (!interactive())
{

# create an R script for the current user
ore.scriptCreate("MYLM",
    function(data, formula, ...) lm(formula, data, ...))
IRIS <- ore.push(iris)
ore.tableApply(IRIS[1:4], FUN.NAME = "MYLM",
    formula = Sepal.Length ~ .)

# create a global R script available to any user
ore.scriptCreate("GLBGLM",
    function(data, formula, ...) glm(formula=formula, data=data, ...),
```
ore.showHiveOptions

Show HIVE Options

Description

Shows the current value of all HIVE options, namely, field delimiters for the HIVE tables and the database name.

Usage

ore.showHiveOptions()

Details

This function displays the values of HIVE options set in the current environment. The HIVE options can be modified using ore.hiveOptions.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

See Also

ore.hiveOptions
ore.sort

**Oracle R Enterprise Data Sorting**

**Description**

Performs flexible sorting on `ore.frame` objects.

**Usage**

```
ore.sort(data, by, nls.sort = NULL, reverse = FALSE, stable = FALSE,
         unique.keys = FALSE, unique.data = FALSE,
         cache = getOption(".ore.persist", TRUE))
```
ore.sort

Arguments

data  An ore.frame object.
by     A comma-separated character string specifying the columns to use in sorting from argument data.
nls.sort   A character string specifying Oracle Database NLS_SORT options.
reverse   A logical vector indicating the use of ascending or descending sorts.
stable    A logical value indicating whether to maintain the relative order within sorted groups.
unique.keys A logical value indicating whether to maintain a single row for each distinct combination of sorting columns.
unique.data A logical value indicating whether to maintain a single row for each distinct combination of all the columns in argument data.
cache For internal use only.

Details

Typical use case for ore.sort is during data pre-processing to obtain top-k rows.

Value

Returns an ore.frame object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.rank

Examples

IRIS <- ore.push(iris)

# Sort all specified columns in descending order
ore.sort(data = IRIS, by = c("Petal.Length", "Sepal.Length"), reverse = TRUE)

# Sort one of the columns in ascending and another in descending order
ore.sort(data = IRIS, by = c("-Petal.Length", "Sepal.Length"))

# Retain just one row per unique value of Petal.Length
ore.sort(data = IRIS, by = "Petal.Length", unique.key = TRUE)

# Remove duplicate rows and rows with the same value for Petal.Length
ore.sort(data = IRIS, by = "Petal.Length", unique.key = TRUE,
         unique.data = TRUE)
ore.summary

Oracle R Enterprise Data Summary

Description

Generates descriptive statistics for ore.frame objects within flexible row aggregations.

Usage

ore.summary(data, var, stats = c("n", "mean", "min", "max"),
        class = NULL, types = NULL, ways = NULL, weight = NULL,
        order = NULL, maxid = NULL, minid = NULL, mu = 0,
        no.type = FALSE, no.freq = FALSE)

Arguments

data An ore.frame object of data.

var A vector of character strings specifying the names of numeric columns in argument data to which to apply all of the statistical calculations in argument stats, or a list of character string vectors. If the var argument is a list, then the length of the list must be either 1 or the same as the length of stats. If it's a list of length 1, it's equivalent to a vector of strings. If it's a list of length greater than 1, each element of the var list specifies the columns of data to which to apply the statistical calculation in the corresponding position in stats.

stats A vector of character strings specifying the statistical calculations for argument var. If the name of the vector element is specified, the name becomes the output column name.

The values of this argument can be one or more of the following:
"n" or "freq" (Count of non-missing values),
"count" or "cnt" (Count of all observations),
"nmiss" (Count of missing values),
"mean" or "avg" (Average of values),
"min" (Minimum of values)
"max" (Maximum of values),
"css" (Corrected sum of squares),
"uss" (Uncorrected sum of squares),
"cv" (Coefficient of variation),
"sum" (Sum of values),
"sumwgt" (Weighted sum of values),
"range" (Range of values),
"stddev" or "std" (Standard deviation of values),
"stderr" or "stdmean" (Standard error for the mean),
"variance" or "var" (Variance of values),
"kurtosis" or "kurt" (Kurtosis),
"skewness" or "skew" (Skewness),
"loccount<" or "loc<" (Number of observations whose values are less than the supplied mu).
"loccount>" or "loc>" (Number of observations whose values are greater than the supplied \( \mu \)),

"loccount!" or "loc!" (Number of observations whose values are not equal to the supplied \( \mu \)),

"loccount" or "loc" (Number of observations whose values are equal to the supplied \( \mu \)),

Percentiles Types: "p0", "p1", "p5", "p10", "p25" or "q1", "p50" or "q2" or "median", "p75" or "q3", "p90", "p95", "p99", "p100" (Percentile or quantile),

"qrange" or "iqr" (Interquartile range, Q3-Q1),

"mode" (Most frequently occurring value),

"lclm" (Two-sided left confidence limit with confidence level of the interval equal to 0.95),

"rclm" (Two-sided right confidence limit with confidence level of the interval equal to 0.95),

"clm" (Two-sided confidence interval with confidence level of the interval equal to 0.95),

"t" (Student’s t-test statistic),

"probt" or "prt" (Two-tailed p-value for student’s t-test)

class
A vector of character strings specifying the names of categorical columns within argument data. If not specified, the aggregation of the entire data is returned.

types
A list of character string vectors specifying the combinations of the column names in class within which the aggregations will be executed in the returning summary.

ways
A vector of integers with each value indicating the number of columns in class that are used to generate types. With one integer number, it generates types of all possible combinations with the specified number of columns in class. The types generated by ways will be combined with the types specified in types with redundancy removed automatically.

weight
An optional single character string specifying a numeric column within data to use as analytic weights. By default, the weight for each non-missing observation is 1. The statistics in stats that can take weight are "sum", "sumwgt", "mean", "css", "uss", "cv", "stddev", "variance", and "stderr". The weight argument is ignored when specified with other statistics.

order
A vector of character strings specifying the sorting criteria. The values of this argument can be one or more of the following:

"freq" or "-freq" (Ascending or descending sorts based on count statistics),

"type" or "-type" (Ascending or descending sorts based on type),

"class" or "-class" (Ascending or descending sorts based on the columns in class).

maxid
A named vector of character strings, each element of which specifies two columns in data. The name of an element specifies an over-column and the value of the element specifies an id-column. Each element results in an additional column in the returned ore.frame object. Each additional column contains the value from the id-column that corresponds to the observation that has the maximum value in the over-column.
ore.summary

minid A named vector of character strings, each element of which specifies two columns in data. The name of an element specifies an over-column and the value of the element specifies an id-column. Each element results in an additional column in the returned ore.frame object. Each additional column contains the value from the id-column that corresponds to the observation that has the minimum value in the over-column.

mu A single number or a vector of numbers whose elements correspond to each value in var, to supply additional numeric parameters for some statistics. The default value is 0. The statistics that use mu are "loccount<", "loccount>", "loccount", "loccount!", "t", and "probt". The mu argument is ignored when specified with other statistics.

no.type A logical value indicating whether to drop the TYPE column from the output.

no.freq A logical value indicating whether to drop the FREQ column from the output.

Details

The function ore.summary generates descriptive statistics for ore.frame objects within user specified aggregation sub-groups.

The argument class specifies the columns to be used to define aggregation sub-groups. The arguments types and ways define the sub-groups. If class is NULL, the function aggregates the entire data without sub-groups. If class is specified, but both types and ways are NULL, the function returns aggregations of all possible sub-groups by the columns in class. The number of sub-groups increases exponentially over the number of class columns. Oracle recommends using types and ways to specify the sub-groups of interest.

Value

Returns an ore.frame object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Examples

IRIS <- ore.push(iris)

ore.summary(IRIS, c("Sepal.Length", "Petal.Length"))

ore.summary(IRIS, c("Sepal.Length", "Petal.Length"), c("mean", "std", "p10"), class="Species")

ore.summary(IRIS, list(c("Sepal.Length", "Petal.Length"), "Sepal.Width"), c(avg="mean", "std"), class="Species")

ore.summary(IRIS, c("Sepal.Length", "Petal.Length"), c("mean", "std"), class="Species", weight="Sepal.Width")

ore.summary(IRIS, c("Sepal.Length", "Petal.Length"), c("mean", "std"), class=c("Species", "Petal.Width"),
ore.sync

Oracle R Enterprise Object Synchronization Function

Description

Synchronize ore.frame objects the represent database tables, views, and queries within the R environment for a schema in the Oracle R Enterprise session.

Usage

ore.sync(schema, table = NULL, use.keys = TRUE, query = NULL)

Arguments

schema A character string specifying the database schema name. Not supported with use of argument query.

table An optional character vector specifying the database table or view names to use for ore.frame object creation.

use.keys A logical value specifying whether primary keys, if they exist, should be used for constructing ordered ore.frame objects.

query An optional named character vector specifying queries to use for ore.frame object creation. The element names are used as the names of the ore.frame objects in the R environment for the default schema.

Details

Oracle R Enterprise creates proxy objects in R that correspond to the tables or views in the database schema. These proxy objects contain metadata used by Oracle R Enterprise internally to provide transparency layer functionality.

By default, function ore.sync uses the schema given at the time of ore.connect. The schema argument can be used to select a different schema for which the user has the appropriate access privileges.

When both the table and query arguments are NULL in an ore.sync function call, all tables and views whose name do not contain a $ or begin with SYS_ are selected. Selecting all of the tables in a database schema can be expensive and so the recommended practice is to set either the table or query argument to limit the the number of tables and views represented in the Oracle R Enterprise schema environment.

types=list("Species", c("Species", "Petal.Width")),
order=c("type", "-freq", "class")

ore.summary(IRIS, c("Sepal.Length", "Petal.Length"), c("mean", "std"),
class=c("Species", "Petal.Width"),
ways=1, order=c("type", "-freq", "class"))

ore.summary(IRIS, c("Sepal.Length", "Petal.Length"), c("mean", "prt"),
class="Species", mu=c(5.8, 3.7))

ore.summary(IRIS, c("Sepal.Length", "Petal.Length"), "mean",
class="Species",
maxid=c(Sepal.Length="Sepal.Width", Petal.Length="Petal.Width")
If argument `use.keys` is set to `FALSE` or if the table has no primary key, the corresponding `ore.frame` object will be unordered; otherwise it will be ordered. For many operations on `ore.frame` objects, for example `colMeans`, the row order is not important. However for operations that require a row ordering, for example `diff`, an error message is issued specifying that the ordering information is not available. A row ordering can be imposed on an `ore.frame` object by setting its row names to one or more of its columns.

**Author(s)**

Oracle <oracle-r-enterprise@oracle.com>

**References**

Oracle R Enterprise

**See Also**

`ore.attach`, `ore.connect`, `ore.exists`, `ore.get`, `ore.ls`, `ore.rm`, `row.names`

**Examples**

```r
if (!interactive()) {
  ore.sync()
  ore.sync("RQUSER", use.keys = FALSE)
  ore.exec("create table TABLE1 as select * from dual")
  ore.exec("create table TABLE2 as select * from dual")
  ore.sync(table = c("TABLE1", "TABLE2"))
  ore.sync("RQUSER", table = c("TABLE1", "TABLE2"))
  ore.sync(query = c("QUERY1" = "select 0 X, 1 Y from dual",
                     "QUERY2" = "select 1 X, 0 Y from dual"))
  ore.drop("TABLE1")
  ore.drop("TABLE2")
}
```

---

**ore.toXML**

*Oracle R Enterprise XML String Generation Function*

**Description**

Creates a string containing an XML representation for `vector`, `matrix`, `data.frame`, `list` or `ore` objects.

**Usage**

`ore.toXML(obj)`

**Arguments**

`obj`  
A `vector`, `matrix`, `data.frame`, `list` or `ore` object.
ore.univariate

Details
Generates XML strings intended for consumption by external tools, such as Business Intelligence (BI) web portals.

Value
A character string containing an XML representation of obj.

Author(s)
Oracle <oracle-r-enterprise@oracle.com>

References
Oracle R Enterprise

See Also
ore

Examples
ore.toXML(1:10)
irisXML <- ore.toXML(iris)
substring(irisXML, 1, 65)

---

ore.univariate  Oracle R Enterprise Univariate Summaries

Description
Performs distribution analysis of numeric columns in an object of type ore.frame. Reports all statistics from the function ore.summary plus signed-rank test and extreme values.

Usage
ore.univariate(data, var, class = NULL, stats = NULL, special = NULL, id = NULL, weight = NULL, extremes = FALSE, nextrval = 5, exclnpwgt = FALSE, freq = NULL, idout = FALSE, loccount = FALSE, mu0 = NULL, def = NULL)

Arguments
- data  An ore.frame object.
- var  A comma-separated character string specifying the names of numeric columns within argument data.
- class  A comma-separated character string specifying the names of categorical columns within argument data.
<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>stats</strong></td>
<td>A comma-separated character string specifying the statistical calculations for argument <code>var</code>. Currently the supported arguments are <code>moments</code> or <code>m</code>, <code>measures</code> or <code>s</code>, <code>quantiles</code> or <code>q</code>, <code>location</code> or <code>l</code>, <code>normality</code> or <code>n</code>, <code>loccount</code> or <code>lc</code>, <code>extremes</code> or <code>x</code>.</td>
</tr>
<tr>
<td><strong>special</strong></td>
<td>Argument not supported.</td>
</tr>
<tr>
<td><strong>id</strong></td>
<td>Argument not supported.</td>
</tr>
<tr>
<td><strong>weight</strong></td>
<td>An optional character string specifying a numeric column within argument <code>data</code> to use as analytic weights.</td>
</tr>
<tr>
<td><strong>extremes</strong></td>
<td>A logical value indicating whether to report extreme values for columns specified in argument <code>var</code>.</td>
</tr>
<tr>
<td><strong>nextrval</strong></td>
<td>An integer value specifying the depth of reporting for extreme values; default is 5.</td>
</tr>
<tr>
<td><strong>exclnpwgt</strong></td>
<td>Argument not supported.</td>
</tr>
<tr>
<td><strong>freq</strong></td>
<td>Argument not supported.</td>
</tr>
<tr>
<td><strong>idout</strong></td>
<td>Argument not supported.</td>
</tr>
<tr>
<td><strong>loccount</strong></td>
<td>Argument not supported.</td>
</tr>
<tr>
<td><strong>mu0</strong></td>
<td>Argument not supported.</td>
</tr>
<tr>
<td><strong>def</strong></td>
<td>Argument not supported.</td>
</tr>
</tbody>
</table>

**Value**

Returns an `ore.frame` object.

**Author(s)**

Oracle <oracle-r-enterprise@oracle.com>

**References**

Oracle R Enterprise

**See Also**

`ore.summary`

**Examples**

```r
IRIS <- ore.push(iris)

# Default univariate statistics
x <- ore.univariate(IRIS, var=c("Sepal.Length", "Sepal.Width"))

# Compute location statistics on Sepal.Length
x <- ore.univariate(IRIS, var="Sepal.Length", stats="location")

# Compute quantiles statistics on Sepal.Length and Sepal.Width
x <- ore.univariate(IRIS, var=c("Sepal.Length", "Sepal.Width"), stats="quantiles")
```
Description

Univariate aggregations based on a set of grouping variables.

Usage

```r
## S4 method for signature 'ore.vector'
aggregate(x, by, FUN, ..., simplify = TRUE)
```

Arguments

- `x`: An `ore.vector` object.
- `by`: Either an `ore.vector` object representing a single grouping variable or an `ore.frame` or list object containing a set of grouping variables.
- `FUN`: One of `all`, `any`, `fivenum`, `IQR`, `length`, `max`, `mean`, `median`, `min`, `prod`, `quantile`, `range`, `sd`, `sum`, `summary`, `var`.
- `...`: Optional additional arguments supplied to argument `FUN`.
- `simplify`: Argument not supported.

Details

Unlike the `aggregate` method for vectors in the `stats` package, the `FUN` argument is limited to the predefined list given above.

Value

An `ore.frame` object with (number of grouping variables plus number of aggregate values) columns, where the first set of columns provide the distinct combinations of `by` argument values and the last set of columns contains the `FUN` aggregate values.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`aggregate`, `ore.vector-ave`, `ore.number-univar`

Examples

```r
eG <- ore.push(ToothGrowth)
aggregate(eG$len, eG[2:3], mean)
aggregate(eG$len, eG[2:3], quantile)
```
Description

Performs univariate operations within a set of grouping variables.

Usage

```r
## S4 method for signature 'ore.vector'
ave(x, ..., FUN = mean)
```

Arguments

- `x`: An `ore.vector` object.
- `...`: A set of `ore.vector` and `ore.frame` objects representing grouping variables as well as optional arguments to `FUN`.
- `FUN`: One of `length`, `max`, `mean`, `median`, `min`, `rank`, `sd`, `sum`, `var`.

Details

Unlike the `ave` method for vectors in the `stats` package, the `FUN` argument is limited to the predefined list given above.

Value

An `ore.vector` object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ave`, `ore.vector-aggregate`, `ore.number-univar`

Examples

```r
TG <- ore.push(ToothGrowth)
ave(TG$len, TG[2:3], FUN = mean)
ave(TG$len, TG[2:3], FUN = rank)
```
ore.vector-class

Class ore.vector

Description

The `ore.vector` class represents data columns in Oracle R Enterprise.

Accessors

- `length(x)`: Returns the number of elements in argument `x`.
- `names(x)`: Returns an `ore.character` object containing the element names. When the element names are made of multiple components, they will be separated with the value specified in the `ore.sep` option, which by default is set to "|".
- `names(x) <- value`: Replaces the element names in argument `x` with the names in argument `value`. The `value` argument must be either `NULL` to remove element names, an `ore.vector` object for single component names, or a `ore.frame` object for multiple component names.

Subsetting

In the code snippets below, argument `x` is an `ore.vector` object.

- `x[i]`: Returns the `ore.logical` selected elements of argument `x` as an `ore.vector` object.
- `x[i] <- value`: Creates/replaces the `ore.logical` specified elements of argument `x` with `value`.
- `head(x, n = 6L)`: If argument `n` is non-negative, returns the first `n` elements of argument `x`. If `n` is negative, returns all but the last `abs(n)` elements of argument `x`.
- `tail(x, n = 6L)`: If argument `n` is non-negative, returns the last `n` elements of argument `x`. If `n` is negative, returns all but the first `abs(n)` elements of argument `x`.

Splitting and Combining

In the code snippets below, argument `x` is an `ore.vector` object.

- `split(x, f, drop = FALSE)`: Splits argument `x` into a `list` object, according to argument `f`, dropping elements corresponding to unrepresented levels if `drop` is `TRUE`.
- `c(...)`: Returns a new `ore.vector` object by combining the elements of the `ore.vector` objects in `...`.

Looping

In the code snippets below, argument `x` is an `ore.frame` object.

- `tapply(X, INDEX, FUN = NULL, ..., simplify = TRUE)`: Apply argument `FUN` to each partitioning of argument `data`, an `ore.vector` object, specified by the `list` of factor objects argument `INDEX`.
- `by(x, INDICES, FUN, ..., simplify = TRUE)`: Apply argument `FUN` to each partitioning of argument `data`, an `ore.vector` object, specified by the factor (or list of factor objects) argument `INDICES`. 
Utilities

In the code snippets below, argument \( x \) is an \texttt{ore.vector} object.

\( x \%in\% \text{table} \): Returns an \texttt{ore.logical} object indicating which values in argument \( x \) are present in argument \text{table}. To maximize performance, if \text{table} is a large \texttt{R} vector, then use \texttt{ore.push} to convert it to an \texttt{ore.vector} before using \( x \%in\% \text{table} \).

\texttt{is.na(x)}: Returns an \texttt{ore.logical} object indicating which values in argument \( x \) contain missing values.

\texttt{pmax(...,na.rm = FALSE)}: Parallel maxima of the \texttt{ore.vector} objects in the \(...\) argument. Removes NA values when \( \text{na.rm} = \text{TRUE} \).

\texttt{pmin(...,na.rm = FALSE)}: Parallel minima of the \texttt{ore.vector} objects in the \(...\) argument. Removes NA values when \( \text{na.rm} = \text{TRUE} \).

\texttt{rank(x,na.last = TRUE,ties.method = c("average","first","random","max","min"))}: Returns the ranks for values of argument \( x \).

\texttt{sort(x,decreasing = FALSE,na.last = NA,...)}: Returns a sorted version of argument \( x \).

\texttt{table(...,exclude,useNA = c("no","ifany","always"),dnn,deparse.level = 1)}: Tabulates the values of argument \( x \).

\texttt{unique(x)}: Returns the unique values of argument \( x \).

Group Generics

\texttt{ore.vector} objects have support for S4 group generic functionality:

Compare \:"==",">","<","!=","<=",">="\n
Summary \:"max","min","range","prod","sum","any","all"\n
See \texttt{S4groupGeneric} for more details.

Coercion

In the code snippets below, argument \( x \) is an \texttt{ore.vector} object.

\texttt{as.character(x)}: Returns an \texttt{ore.character} object coercion of argument \( x \).

\texttt{as.data.frame(x,row.names = NULL,optional = FALSE,...,nm = \text{deparse}\{(\text{substitute}\{x,parent.frame()\},width.cutoff = 30L)[1L]\},stringsAsFactors = \text{default.stringsAsFactors}())}: Returns a single column \texttt{ore.frame} object containing the values of argument \( x \).

\texttt{as.factor(x)}: Returns an \texttt{ore.factor} object coercion of argument \( x \).

\texttt{as.integer(x)}: Returns an \texttt{ore.integer} object coercion of argument \( x \).

\texttt{as.logical(x)}: Returns an \texttt{ore.factor} object coercion of argument \( x \).

\texttt{as.matrix(x,...)}: Returns a single column \texttt{ore.matrix} object containing the values of argument \( x \).

\texttt{as.numeric(x)}: Returns an \texttt{ore.numeric} object coercion of argument \( x \).

\texttt{as.vector(x,mode = "any")}: Returns an \texttt{ore.vector} object based on the values contained in argument \( x \). The vector will be coerced to the requested \( \text{mode} \), unless argument \( \text{mode} \) is "any", in which case the most appropriate type is chosen.

Note

See the corresponding \texttt{R} documentation for the functions listed above.
ore.year

Author(s)
Oracle <oracle-r-enterprise@oracle.com>

References
Oracle R Enterprise

See Also
ore, ore.character, ore.datetime, ore.factor, ore.frame, ore.logical, ore.matrix, ore.number

Examples
showClass("ore.vector")

ore.year Oracle R Enterprise Date/Time Functions

Description
Extracts date and time related information from date/time data types.

Usage
ore.year(x, ...)
ore.month(x, ...)
ore.mday(x, ...)
ore.hour(x, ...)
ore.minute(x, ...)
ore.second(x, ...)

Arguments
x For functions ore.year and ore.month, ore.date and ore.datetime objects. For function ore.mday, ore.date, ore.datetime, ore.difftime and objects. For functions ore.hour, ore.minute, and ore.second, ore.datetime and ore.difftime objects.
...
For future expansion.

Details
These functions return the following information:
ore.year Year with century.
ore.month Month (1-12; January = 1).
ore.mday Day of month (1-31).
ore.hour Hour of day (0-23).
ore.minute Minute (0-59).
ore.second Second (0-59).
Value

For functions `ore.year`, `ore.month`, `ore.mday`, `ore.hour`, and `ore.minute`, returns an `ore.integer` object.

For function `ore.second`, returns an `ore.numeric` object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore.date`, `ore.datetime`, `ore.difftime`

Examples

```r
DATE <- ore.push(seq(as.Date("1999/3/14"), as.Date("2002/2/16"), by="3 months"))
DATE
ore.year(DATE)
ore.month(DATE)
ore.mday(DATE)
```

Description

Density, cumulative distribution function, quantile function and random variate generation for many standard probability distributions.

Details

The functions for the density/mass function, cumulative distribution function, and quantile function are named in the form `dxxx`, `pxxx`, and `qxxx` respectively.

Refer to `dbeta` for information on the beta distribution.

Refer to `dbinom` for information on the binomial (including Bernoulli) distribution.

Refer to `dcauchy` for information on the Cauchy distribution.

Refer to `dchisq` for information on the chi-squared distribution.

Refer to `dexp` for information on the exponential distribution.

Refer to `df` for information on the F distribution.

Refer to `dgamma` for information on the gamma distribution.

Refer to `dgeom` for information on the geometric distribution. (This is also a special case of the negative binomial.)

Refer to `dlnorm` for information on the log-normal distribution.

Refer to `dlogis` for information on the logistic distribution.
Refer to `dnbinom` for information on the negative binomial distribution.
Refer to `dnorm` for information on the normal distribution.
Refer to `dpois` for information on the Poisson distribution.
Refer to `dsignrank` for information on the Wilcoxon Signed Rank statistic distribution.
Refer to `dt` for information on Student’s t distribution.
Refer to `dunif` for the uniform distribution.
Refer to `dweibull` for information on the Weibull distribution.

Author(s)
Oracle <oracle-r-enterprise@oracle.com>

References
Oracle R Enterprise

See Also
`ore.number`

Description
Utility function that displays the Oracle R Enterprise documentation welcome page in a web browser.

Usage
OREShowDoc()

Details
An Oracle R Enterprise welcome page will be displayed in a web browser containing references to Oracle R Enterprise Release Notes as well as Oracle R Enterprise User’s Guide.

Value
Returns an invisible character string specifying the path to the Oracle R Enterprise welcome page.

See Also
`RShowDoc`

Examples
```r
## Not run:
OREShowDoc()
## End(Not run)
```
partitions

## S3 method for class 'ore.frame'
pairs(x, labels, panel = points, ..., lower.panel = panel, upper.panel = panel, diag.panel = NULL, text.panel, label.pos, cex.labels = NULL, font.labels = 1, row1attop = TRUE, gap = 1)

Arguments

- **x**: An `ore.frame` object.
- **labels,panel,...,lower.panel,upper.panel,diag.panel,text.panel,label.pos,cex.labels**: See documentation in `pairs`.

Author(s)

- **Oracle** <oracle-r-enterprise@oracle.com>

References

- **Oracle R Enterprise**

See Also

- `pairs`

partitions

## Partitions and Settings of an Oracle Data Mining Model Object

Description

The `partition` function returns partitions names from a partitioned model. The `settings` function returns the Oracle Data Mining parameter settings used to build the model.
partitions

Usage

partitions(object)

## S3 method for class 'ore.odmPart'
partitions(object)

settings(object)

## S3 method for class 'ore.model'
settings(object)

Arguments

object An partitioned model object.

Value

The function `partitions` returns an `ore.frame` listing each partition of the specified model object with the associated partition column values of the model. The partition name is system-determined. The function returns NULL for a non-partitioned model.

The function `settings` returns a `data.frame` listing each Oracle Data Mining parameter setting name and value pair used to build the model.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User’s Guide

See Also

`ore.odmAssocRules`, `ore.odmKMeans`, `ore.odmOC`

Examples

```r
irisp <- iris
irisp[, "part"] <- as.numeric(rownames(iris))%%2
IRISP <- ore.push(irisp)

# build partition model with one partition column
svm.pmod <- ore.odmSVM(Sepal.Length ~ ., data = IRISP, type = "regression",
                       odm.settings = list(odms_partition_columns = "Species"))

settings(svm.pmod)
summary(svm.pmod)
partitions(svm.pmod)
head(predict(svm.pmod, IRISP, supplemental.cols = "part"))
```
```r
# build partition model with two partition columns
svm.pmod2 <- ore.odmSVM(Sepal.Length ~ ., data = IRISP, type = "regression",
                        odm.settings = list(odms_partition_columns = c("Species", "part"))

settings(svm.pmod2)
summary(svm.pmod2)
partitions(svm.pmod2)
head(predict(svm.pmod2, IRISP, supplemental.cols = "part"))
```

---

### plot.ore.vector

**Generic X-Y Plotting**

**Description**

Generic plotting function.

**Usage**

```r
## S3 method for class 'ore.vector'
plot(x, y = NULL, type = "p",
xlim = NULL, ylim = NULL, log = "",
main = NULL, sub = NULL, xlab = NULL, ylab = NULL,
ann = par("ann"), axes = TRUE, frame.plot = axes,
panel.first = NULL, panel.last = NULL, asp = NA, ...)
```

**Arguments**

- `x, y` : `ore.vector` objects.
- `type, xlim, ylim, log, main, sub, xlab, ylab, ann, axes, frame.plot, panel.first, panel.last`
  
  See description in `plot`.

**Author(s)**

Oracle <oracle-r-enterprise@oracle.com>

**References**

Oracle R Enterprise

**See Also**

plot
Add Points to a Plot

Description
Draws points.

Usage
```r
## S3 method for class 'ore'
points(x, y = NULL, type = "p", ...)
```

Arguments
- `x`, `y` : `ore.number` objects.
- `type`, `...`: See description in `points`.

Author(s)
Oracle <oracle-r-enterprise@oracle.com>

References
Oracle R Enterprise

See Also
- `points`

Oracle R Enterprise Test of Equal or Given Proportions

Description
Tests the null hypothesis that either the proportions in several groups are the same, or that they equal certain given values.

Usage
```r
## S4 method for signature 'ore.vector'
prop.test(x, n, p = NULL,
          alternative = c("two.sided", "less", "greater"),
          conf.level = 0.95, correct = TRUE)
```
Arguments

- **x**: An `ore.vector` object.
- **n**: Argument not supported.
- **p**: A vector of probabilities; one for each distinct value in argument `x`.
- **alternative**: A character string specifying the alternative hypothesis and must be one of "two.sided" (default), "greater" or "less".
- **conf.level**: A numeric value specifying the level of the confidence interval.
- **correct**: A logical value indicating whether Yates’ continuity correction should be applied where possible.

Value

Returns an `prop.test` object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References


Oracle R Enterprise

See Also

`prop.test`

---

**ranking**

Various Ranking Functions

---

Description

Six variations of ranking functions to rank the elements in an ordered `ore.vector` by its values. An `ore.character` is coerced to an `ore.factor`. The values of `ore.factor` are based upon factor levels. Use `desc` to reverse the direction.
**Usage**

- `row_number(x)`
- `ntile(x, n)`
- `min_rank(x)`
- `dense_rank(x)`
- `percent_rank(x)`
- `cume_dist(x)`

**Arguments**

- **x**: An ordered `ore.number`, `ore.character`, or `ore.factor` object to rank. Missing values are left as is.
- **n**: The number of groups to split into.

**Details**

- `row_number`: Equivalent to `rank(ties.method = "first")`.
- `min_rank`: Equivalent to `rank(ties.method = "min")`.
- `dense_rank`: Like `min_rank`, but with no gaps between ranks.
- `percent_rank`: A number between 0 and 1 computed by rescaling `min_rank` to [0, 1].
- `cume_dist`: A cumulative distribution function. The proportion of all values that are less than or equal to the current rank.
- `ntile`: A rough rank, which breaks the input vector into n buckets.

**Examples**

```r
X <- ore.push(c(5, 1, 3, 2, 2, NA))
row_number(X)
row_number(desc(X))
min_rank(X)
dense_rank(X)
percent_rank(X)
cume_dist(X)
ntile(X, 2)
ntile(ore.push(runif(100)), 10)
```
reorder  

Reorder Levels of an ore.factor Object

Description

Reorders the levels of an ore.factor based on the values of a second ore.vector.

Usage

```r
## S4 method for signature 'ore.factor'
reorder(x, X, FUN = mean, ..., order = is.ordered(x))
```

Arguments

- **x**  
  An ore.factor object.

- **X**  
  An ore.vector of the same length as argument x.

- **FUN**  
  A function whose first argument represents an ore.vector and returns a scalar, to be applied to each subset of argument X determined by the levels of argument x.

- **...**  
  Optional additional arguments supplied to argument FUN.

- **order**  
  Argument not implemented.

Value

Returns an ore.factor object with reordered factor levels.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.factor

Examples
rules

Predicate Rule Data for Clusters or Association Rules Models

Description

The generic `rules` function is used to return cluster predicate rule data from a clustering model or to return association rules from an association model.

Usage

```r
rules(object, ...)  
## S3 method for class 'ore.odmAssocRules'
rules(object, ...)  
## S3 method for class 'ore.odmKMeans'
rules(object, ...)  
## S3 method for class 'ore.odmOC'
rules(object, ...)  
## S3 method for class 'ore.odmEM'
rules(object, ...)
```

Arguments

- `object`: An object for which cluster rules or association rules are desired.
- `...`: Additional arguments affecting the result.

Details

For clustering models (``ore.odmKMeans``,  ``ore.odmOC``), and ``ore.odmEM``, the function `rules` provides the predicate rule data that defines each cluster as determined by the algorithm.

For an association model (``ore.odmAssocRules``), the function `rules` provides the property of each association rule.

Value

For clustering model objects, the function `rules` returns a list with elements for each cluster. Each element contains a `data.frame` with the following columns (note that common data repeats across rows):

- `rhs.cluster.id`: The numeric identifier associated with the cluster.
- `rhs.support`: The right-hand side number of rows (support) for this cluster.
- `rhs.conf`: The right-hand side rule confidence associated with this cluster.
- `lhs.conf`: The left-hand side rule confidence associated with this cluster.
- `lhs.variable`: The left-hand side variable name.
- `lhs.var.support`: The left-hand side number of rows (support) for the variable and the combined predicate.
- `lhs.var.conf`: The left-hand side confidence for variable and combined predicate.
For numerical variables, the lower or upper bound predicate, for categorical variables, the specific category.

For association model objects, the function `rules` returns an object of class `ore.rules` that describes the property of each rule. An `ore.rules` object has the following columns:

- **RULE_ID**: The numerical identifier associated with each rule.
- **NUMBER_OF_ITEMS**: The number of items in the rule.
- **LHS**: The itemset of the antecedent.
- **RHS**: The itemset of the consequent.
- **SUPPORT**: The support of the rule.
- **CONFIDENCE**: The confidence of the rule.
- **LIFT**: The lift of the rule.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

- Oracle R Enterprise
- Oracle Data Mining Concepts
- Oracle Data Mining User’s Guide

See Also

- `ore.odmAssocRules`, `ore.odmKMeans`, `ore.odmOC`, `ore.odmEM`, `ore.rules-class`

Examples

```r
# clustering
x <- rbind(matrix(rnorm(100, sd = 0.3), ncol = 2),
           matrix(rnorm(100, mean = 2, sd = 0.3), ncol = 2))
colnames(x) <- c("x", "y")

X <- ore.push (data.frame(x))
km.mod <- NULL
km.mod <- ore.odmKMeans(~., X, num.centers=2)
rules.km <- rules(km.mod)

oc.mod <- NULL
oc.mod <- ore.odmOC(~., X, num.centers=2)
rules.oc <- rules(oc.mod)

# association rules
id <- c(1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3)
item <- c("b", "d", "e", "a", "b", "c", "e", "b", "c", "d", "e")
data.ore <- ore.push(data.frame(ID = id, ITEM = item))

ar.mod <- ore.odmAssocRules(~., data.ore, case.id.column = "ID",
                           item.id.column = "ITEM", min.support = 0.6, min.confidence = 0.6,
```

max.rule.length = 3)
rules.ar <- rules(ar.mod)

describe_sample

Sample Rows from an ore.frame Object

Description

Samples an ore.frame object by a fixed number of rows or a fraction, with optional arguments of a sample weight and whether or not to use replacement.

Usage

sample_n(tbl, size, replace = FALSE, weight = NULL)
sample_frac(tbl, size = 1, replace = FALSE, weight = NULL)

Arguments

tbl
An ore.frame object to sample.

size
For sample_n, the number of rows to select. For sample_frac, the fraction of rows to select. If tbl is grouped, then size applies to each group.

replace
Specify TRUE to sample with replacement or FALSE to sample without replacement. The default is FALSE.

weight
Sampling weights. This expression is evaluated in the context of the input ore.frame or an ore.number object. It must return a vector of non-negative numbers with the same length as the input. Weights are automatically normalized to sum to 1.

Examples

MTCARS <- ore.push(mtcars)
by_cyl <- group_by(MTCARS, cyl)

# Sample fixed number per group
sample_n(MTCARS, 10)
nrow(sample_n(MTCARS, 50, replace = TRUE))
sample_n(MTCARS, 10, weight = mpg)
sample_n(MTCARS, 10, weight = MTCARS[,"mpg"])

arrange(sample_n(by_cyl, 3), cyl, mpg)
arrange(summarise(sample_n(by_cyl, 10, replace = TRUE), n = n()), cyl)
arrange(summarise(sample_n(by_cyl, 3, weight = mpg/mean(mpg)), n = n()), cyl)
arrange(summarise(sample_n(by_cyl, 3, weight = by_cyl["mpg"]/mean(by_cyl["mpg"])), n

# Sample fixed fraction per group
nrow(sample_frac(MTCARS, 0.1))
nrow(sample_frac(MTCARS, 1.5, replace = TRUE))
nrow(sample_frac(MTCARS, 0.1, weight = 1/mpg))

arrange(summarise(sample_frac(by_cyl, 0.2), n = n()), cyl)
arrange(summarise(sample_frac(by_cyl, 1, replace = TRUE), n = n()), cyl)

select  

Select, Rename, Arrange, Filter, Mutate, or Transmute an 
ore.frame Object

Description

select Selects only the specified columns. rename Renames the specified columns and keeps 
all columns. arrange Orders rows by the specified columns. filter Filters rows by matching 
the specified condition. mutate Adds new columns. transmute Adds new columns and drops 
the existing columns.

Usage

select(.data, ...)  
select_(.data, ..., .dots)

rename(.data, ...) 
rename_(.data, ..., .dots)

arrange(.data, ...)  
arrange_(.data, ..., .dots)

filter(.data, ...)  
filter_(.data, ..., .dots)

mutate(.data, ...)  
mutate_(.data, ..., .dots)

transmute(.data, ...)  
transmute_(.data, ..., .dots)

Arguments

.data  An ore.frame object.
...
...  Comma separated list of unquoted expressions. See select, rename, arrange, 
filter, mutate, transmute for details.
.dots  Used by select_, rename_, arrange_, filter_, mutate, and transmute_ 
for standard evaluation. See select_, rename_, arrange_, filter_, 
mutate_, transmute_ for details.

Value

An ore.frame object.

Special functions

select does not support the special functions in select.

See Also

slice, slice_, summarise, summarise_
slice

Select Rows by Positions

slice works with ordered ore.frame object. It ignores the grouping of the input ore.frame.
Usage

slice(.data, ...)

slice_(.data, ..., .dots)

Arguments

.data An ordered ore.frame object.
...
.dots An integer vector specifying row positions. Used to work around non-standard evaluation. See slice_ for details.

See Also

Other single.table.verbs: arrange, arrange_, filter, filter_, mutate, mutate_, transmute, transmute_, rename, rename_, select, select_, summarise, summarise_

Examples

MTCARS <- ore.push(mtcars)
rownames(MTCARS)
slice(MTCARS, 1L)
slice(MTCARS, n())
slice(MTCARS, 25:n())

MTCARS <- arrange(MTCARS, hp)
slice(MTCARS, 1L)
slice(MTCARS, n())
slice(MTCARS, 25:n())

# grouping is ignored by slice
# use filter and row_number to obtain slices per group
by_cyl <- group_by(MTCARS, cyl)
filter(by_cyl, row_number(hp) < 3L) # slice(by_cyl, 1:2)

summarise

Summarise Columns by Aggregate Functions

Description

Aggregates the specified column values.

When an ore.frame object is grouped, the aggregate function is applied group-wise. The supported aggregate functions are min, mean, max, median, length, IQR, prod, sum, range, quantile, fivenum, summary, sd, var, all, and any. The resulting ore.frame drops one grouping of the input ore.frame.

Usage

summarise(.data, ...)

summarise_(.data, ..., .dots)
Arguments

.data  An ore.frame object.

...  Name-value pairs of aggregate functions such as min(), mean(), max(), and so on.

dots  Used to work around non-standard evaluation. See summarise_ for details.

Value

An ore.frame object.

See Also

arrange, arrange_, filter, filter_, mutate, mutate_, transmute, transmute_;
rename, rename_, select, select_; slice, slice_

Examples

MTCARS <- ore.push(mtcars)
summarise(MTCARS, mean(disp))
arrange(summarise(group_by(MTCARS, cyl), mean(disp)), cyl)
arrange(summarise(group_by(MTCARS, cyl), m = mean(disp), r = range(disp)), cyl)

library(magrittr)
by_cyl <- MTCARS %>% group_by(cyl)
by_cyl %>% summarise(a = n(), b = n() + 1) %>% arrange(cyl)

summary.ore.odmAssocRules

Summarize In-Database Association Models

Description

Methods for class ore.odmAssocRules or summary.ore.odmAssocRules objects.

Usage

## S3 method for class 'ore.odmAssocRules'
summary(object,...)

## S3 method for class 'summary.ore.odmAssocRules'
print(x,...)

Arguments

object  An object of type ore.odmAssocRules.

x  An object of type summary.ore.odmAssocRules.

...  Additional arguments affecting the predictions produced.
Details

The function `summary.ore.odmAssocRules` summarizes the results produced by the `ore.odmAssocRules` function.

Value

The function `summary.ore.odmAssocRules` returns an object of class `summary.ore.odmAssocRules`, which is a list with the following components:

- **call**: The function call used to build the model with the given arguments.
- **settings**: A `data.frame` of settings used to build the model.
- **attributes**: A `data.frame` of attributes used to build the model.
- **rules**: An object of type `ore.rules` describing the association rules from the model built.
- **itemsets**: An object of type `ore.itemsets` describing the item sets from the model built.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

- Oracle R Enterprise
- Oracle Data Mining Concepts
- Oracle Data Mining User’s Guide

See Also

- `ore.odmAssocRules`, `ore.rules`, `ore.itemsets`

Examples

```r
## For examples see \code{\link{ore.odmAssocRules}}.
```

summary.ore.odmDT  Summarize In-Database Decision Tree Models

Description

Methods for class `ore.odmDT` or `summary.ore.odmDT` objects.

Usage

```r
## S3 method for class 'ore.odmDT'
summary(object,...)

## S3 method for class 'summary.ore.odmDT'
print(x,...)
```
Arguments

object  An object of type `ore.odmDT`.

x       An object of type `summary.ore.odmDT`.

... Additional arguments affecting the predictions produced.

Details

The function `summary.ore.odmDT` summarizes the results produced by the `ore.odmDT` function.

Value

The function `summary.ore.odmDT` returns an object of class `summary.ore.odmDT`, which is a list with the following components:

- `call` The function call used to build the model with the given arguments.
- `n` The number of rows in the training data.
- `costs` A `data.frame` containing the cost matrix supplied at model build.
- `nodes` A `data.frame` with tree node details, including: parent node id, node id, number of rows assigned to that node, predicted value, split predicate, surrogate variables (if applicable), and full split predicates from current node to root node.
- `settings` A `data.frame` of settings used to build the model.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User’s Guide

See Also

`ore.odmDT`

Examples

```r
## For examples see \code{\link{ore.odmDT}}.
```
Summary In-Database Expectation Maximization Models

Description
Methods for class `ore.odmEM` or `summary.ore.odmEM` objects.

Usage

```r
## S3 method for class 'ore.odmEM'
summary(object, ...)

## S3 method for class 'summary.ore.odmEM'
print(x, ...)
```

Arguments

- `object` An object of type `ore.odmEM`.
- `x` An object of type `summary.ore.odmEM`.
- `...` Additional arguments affecting the summary produced.

Details

The function `summary.ore.odmEM` summarizes the results produced by the `ore.odmEM` function.

Value

The function `summary.ore.odmEM` returns an object of class `summary.ore.odmEM`, which is a list with the following components:

- `call` The function call used to build the model with the given arguments.
- `settings` A `data.frame` of settings used to build the model.
- `centers2` A `data.frame` that has the cluster ID values as rownames, and columns corresponding to variable means (if numerical) or mode (if categorical).
- `formula` The `formula` used for the symbolic description of the model fitted.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

- Oracle R Enterprise
- Oracle Data Mining Concepts
- Oracle Data Mining User’s Guide
**summary.ore.odmESA**

### See Also

*ore.odmEM*

### Examples

```r
## For examples see \code{\link{ore.odmEM}}.
```

---

**summary.ore.odmESA  Summarize In-Database Explicit Semantic Analysis Models**

### Description

Methods for class `ore.odmESA` or `summary.ore.odmESA` objects.

### Usage

```r
## S3 method for class 'ore.odmESA'
summary(object, ...)

## S3 method for class 'summary.ore.odmESA'
print(x, ...)
```

### Arguments

- **object**
  - An object of type `ore.odmESA`.
- **x**
  - An object of type `summary.ore.odmESA`.
- **...**
  - Additional arguments affecting the summary produced.

### Details

The function `summary.ore.odmESA` summarizes the results produced by `ore.odmESA`.

### Value

The function `summary.ore.odmESA` returns an object of class `summary.ore.odmESA`, which is a list with the following components:

- **name**
  - The name of the model object.
- **call**
  - The function call used to build the model with the given arguments.
- **settings**
  - A data.frame of settings used to build the model.
- **attributes**
  - A data.frame of attributes used to build the model.
- **features**
  - A data.frame describing the features extracted.
- **formula**
  - The formula used for the symbolic description of the model fitted.

### Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>
summary.ore.odmESM

References

Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User’s Guide

See Also

ore.odmESA

Examples

## For examples see \code{\link{ore.odmESA}}.

summary.ore.odmESM  Summarize In-Database ESM Models

Description

Methods for class ore.odmESM or summary.ore.odmESM objects.

Usage

## S3 method for class 'ore.odmESM'
summary(object,...)

## S3 method for class 'summary.ore.odmESM'
print(x,...)

Arguments

object     An object of type ore.odmESM.
x       An object of type summary.ore.odmESM.
...       Additional arguments affecting the predictions produced.

Details

The function \code{summary.ore.odmESM} summarize the results produced by \code{ore.odmESM}.

Value

The function \code{summary.ore.odmESM} returns an object of class summary.ore.odmESM, which is a list with the following components:

call       The function call used to build the model with the given arguments.
settings   A data.frame of settings used to build the model.
attributes A data.frame of attributes used to build the model.
importance The importance for different attributes
formula    The \code{formula} used for the symbolic description of the model fitted.
summary.ore.odmGLM

Author(s)
Oracle
Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References
Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User’s Guide

See Also
ore.odmESM

Examples
### For examples see \code{\link{ore.odmESM}}.

---

**summary.ore.odmGLM  Summarize In-Database Generalized Linear Models**

**Description**
Methods for class ore.odmGLM or summary.ore.odmGLM objects.

**Usage**
```r
## S3 method for class 'ore.odmGLM'
summary(object,...)

## S3 method for class 'summary.ore.odmGLM'
print(x, digits = max(3, getOption("digits") - 3),
     signif.stars = getOption("show.signif.stars"), ...)
```

**Arguments**
- **object**  An object of class ore.odmGLM.
- **x**  An object of class summary.ore.odmGLM as produced by the summary method.
- **digits**  The number of significant digits to use when printing.
- **signif.stars**  A logical value indicating whether to print ‘significance stars’ for each coefficient.
- **...**  Additional arguments.

**Details**
The function `print.summary.ore.odmGLM` follows the formatting conventions of `print.summary.glm`.
The function `summary.ore.odmGLM` returns an object of class `summary.ore.odmGLM`, which is a list with the following components:

- **call**: The matched call.
- **terms**: The `terms` object used.
- **type**: The type of model fit.
- **family**: A list containing the element family.
- **deviance**: Minus twice the maximized log-likelihood, up to a constant.
- **aic**: The same version of Akaike’s An Information Criterion as used by `glm`.
- **df.residual**: The residual degrees of freedom.
- **null.deviance**: The deviance for the null (intercept only) model.
- **df.null**: The residual degrees of freedom for the null model.
- **na.action**: The number of rows with missing values that were removed.
- **deviance.resid**: The deviance residuals.
- **coefficients**: The matrix of coefficients, standard errors, z-values and p-values.
- **aliased**: The named logical vector showing if the original coefficients are aliased (all elements are `FALSE`).
- **dispersion**: The estimated dispersion.
- **df**: A 3-vector of the rank of the model and the number of residual degrees of freedom, plus number of non-aliased coefficients.
- **nonreference**: For logistic regression, the response values that represents success.
- **ridge**: The `ridge` argument.
- **auto.data.prep**: The `auto.data.prep` argument.

The return value for linear regression models also include elements:

- **residuals**: The deviance residuals.
- **sigma**: The square root of the estimated variance of the random error.
- **r.squared**: The R-squared value.
- **adj.r.squared**: The adjusted R-squared value.
- **fstatistic**: A three element vector containing the value of the F-statistic with its numerator and denominator degrees of freedom.

**Author(s)**

Oracle <oracle-r-enterprise@oracle.com>

**References**

Oracle R Enterprise  
Oracle Data Mining Concepts  
Oracle Data Mining User’s Guide
**summary.ore.odmKMeans**

See Also

ore.odmGLM

Examples

```r
## For examples see \code{\link{ore.odmGLM}}.
```

---

**summary.ore.odmKMeans**  
*Summarize In-Database Hierarchical K-Means Models*

### Description

Methods for class `ore.odmKMeans` or `summary.ore.odmKMeans` objects.

### Usage

```r
## S3 method for class 'ore.odmKMeans'
summary(object,...)

## S3 method for class 'summary.ore.odmKMeans'
print(x,...)
```

### Arguments

- `object`  
  An object of type `ore.odmKMeans`.

- `x`  
  An object of type `summary.ore.odmKMeans`.

- `...`  
  Additional arguments affecting the predictions produced.

### Details

The function `summary.ore.odmKMeans` summarizes the results produced by the `ore.odmKMeans` function.

### Value

The function `summary.ore.odmKMeans` returns an object of class `summary.ore.odmKMeans`, which is a list with the following components:

- `call`  
  The function call used to build the model with the given arguments

- `settings`  
  A data.frame of settings used to build the model.

- `centers2`  
  A data.frame that has the cluster ID values as rownames, and columns corresponding to variable means (if numerical) or mode (if categorical).

- `formula`  
  The `formula` used for the symbolic description of the model fitted.

### Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>
summary.ore.odmNB

References

Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User's Guide

See Also

ore.odmKMeans

Examples

## For examples see \code{\link{ore.odmKMeans}}.

summary.ore.odmNB  Summarize In-Database Naive Bayes Models

Description

Methods for class ore.odmNB or summary.ore.odmNB objects.

Usage

## S3 method for class 'ore.odmNB'
summary(object,...)

## S3 method for class 'summary.ore.odmNB'
print(x,...)

Arguments

object      An object of type ore.odmNB.
x         An object of type summary.ore.odmNB.
...         Additional arguments affecting the predictions produced.

Details

The function summary.ore.odmNB summarize the results produced by ore.odmNB.

Value

The function summary.ore.odmNB returns an object of class summary.ore.odmNB, which
is a list with the following components:

call        The function call used to build the model with the given arguments.
settings    A data.frame of settings used to build the model.
attributes   A data.frame of attributes used to build the model.
apriori     The class distribution for the dependent variable.
tables      A list of table objects, one for each predictor variable with conditional prob-
            abilities.
levels      A vector of unique target class values.
formula    The formula used for the symbolic description of the model fitted.
Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User’s Guide

See Also

ore.odmNB

Examples

## For examples see \code{\link{ore.odmNB}}.

summary.ore.odmNMF  

**Summarize In-Database Non-Negative Matrix Factorization Models**

Description

Methods for class ore.odmNMF or summary.ore.odmNMF objects.

Usage

## S3 method for class 'ore.odmNMF'
summary(object,...)

## S3 method for class 'summary.ore.odmNMF'
print(x, ...)

Arguments

object  
An object of type ore.odmNMF.

x  
An object of type summary.ore.odmNMF.

...  
Additional arguments affecting the predictions produced.

Details

The function summary.ore.odmNMF summarizes the results produced by ore.odmNMF.
Value

The function `summary.ore.odmNN` returns an object of class `summary.ore.odmNN`, which is a list with the following components:

- **name**: The name of the model object.
- **call**: The function call used to build the model with the given arguments.
- **settings**: A `data.frame` of settings used to build the model.
- **attributes**: A `data.frame` of attributes used to build the model.
- **features**: A `data.frame` describing the features extracted.
- **formula**: The `formula` used for the symbolic description of the model fitted.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User’s Guide

See Also

`ore.odmNMF`

Examples

```r
## For examples see \code{\link{ore.odmNMF}}.
```

### Methods for class `ore.odmNN` or `summary.ore.odmNN` objects.

#### Usage

```r
## S3 method for class 'ore.odmNN'
summary(object,...)
```

```r
## S3 method for class 'summary.ore.odmNN'
print(x,...)
```

Arguments

- **object**: An object of type `ore.odmNN`.
- **x**: An object of type `summary.ore.odmNN`.
- **...**: Additional arguments affecting the predictions produced.
Details

The function `summary.ore.odmNN` summarize the results produced by `ore.odmNN`.

Value

The function `summary.ore.odmNN` returns an object of class `summary.ore.odmNN`, which is a list with the following components:

- **call**: The function call used to build the model with the given arguments.
- **settings**: A `data.frame` of settings used to build the model.
- **attributes**: A `data.frame` of attributes used to build the model.
- **importance**: The importance for different attributes
- **formula**: The `formula` used for the symbolic description of the model fitted.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User’s Guide

See Also

`ore.odmNN`

Examples

```r
## For examples see \code{\link{ore.odmNN}}.
```

---

**summary.ore.odmOC**  
**Summarize In-Database Orthogonal Partitioning Cluster Models**

Description

Methods for class `ore.odmOC` or `summary.ore.odmOC` objects.

Usage

```r
## S3 method for class 'ore.odmOC'
summary(object,...)

## S3 method for class 'summary.ore.odmOC'
print(x,...)
```
summary.ore.odmOC

Arguments

object       An object of type ore.odmOC.
x           An object of type summary.ore.odmOC.
...         Additional arguments affecting the predictions produced.

Details

The function summary.ore.odmOC summarizes the results produced by ore.odmOC.

Value

The function summary.ore.odmOC returns an object of class summary.ore.odmOC, which is a list with the following components:

call       The function call used to build the model with the given arguments.
settings   A data.frame of settings used to build the model.
attributes A data.frame of attributes used to build the model.
clusters   A data.frame with each row describing a cluster.
centers2   A data.frame that has the cluster ID values as rownames, and columns corresponding to variable means (if numerical) or mode (if categorical).
formula    The formula used for the symbolic description of the model fitted.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User’s Guide

See Also

ore.odmOC

Examples

```r
## For examples see \code{\link{ore.odmOC}}.
```
Summary for In-Database Neural Network Models

Description

Methods for class `ore.odmRF` or `summary.ore.odmRF` objects.

Usage

```r
## S3 method for class 'ore.odmRF'
summary(object, ...)

## S3 method for class 'summary.ore.odmRF'
print(x, ...)
```

Arguments

- `object` An object of type `ore.odmRF`.
- `x` An object of type `summary.ore.odmRF`.
- `...` Additional arguments affecting the predictions produced.

Details

The function `summary.ore.odmRF` summarize the results produced by `ore.odmRF`.

Value

The function `summary.ore.odmRF` returns an object of class `summary.ore.odmRF`, which is a list with the following components:

- `call` The function call used to build the model with the given arguments.
- `settings` A `data.frame` of settings used to build the model.
- `attributes` A `data.frame` of attributes used to build the model.
- `importance` The importance for different attributes
- `formula` The `formula` used for the symbolic description of the model fitted.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

- Oracle R Enterprise
- Oracle Data Mining Concepts
- Oracle Data Mining User’s Guide

See Also

- `ore.odmRF`
## Examples

```r
## For examples see \code{\link{ore.odmRF}}.
```

### summary.ore.odmSVD

**Summarize In-Database Singular Value Decomposition Models**

#### Description

Methods for class `ore.odmSVD` or `summary.ore.odmSVD` objects.

#### Usage

```r
## S3 method for class 'ore.odmSVD'
summary(object, ...)

## S3 method for class 'summary.ore.odmSVD'
print(x, ...)
```

#### Arguments

- `object`: An object of type `ore.odmSVD`.
- `x`: An object of type `summary.ore.odmSVD`.
- `...`: Additional arguments affecting the summary produced.

#### Details

The function `summary.ore.odmSVD` summarizes the results produced by `ore.odmSVD`.

#### Value

The function `summary.ore.odmSVD` returns an object of class `summary.ore.odmSVD`, which is a list with the following components:

- `name`: The name of the model object.
- `call`: The function call used to build the model with the given arguments.
- `settings`: A `data.frame` of settings used to build the model.
- `attributes`: A `data.frame` of attributes used to build the model.
- `d`: An `ore.frame` with the singular values of the input data. Column `FEATURE_ID` represents the singular value ID.
- `v`: An `ore.frame` whose columns contain the right singular vectors. The column names match with `FEATURE_ID` values.
- `u`: An `ore.frame` whose columns contain the left singular vectors.
- `features`: An `ore.frame` same as `v` but in unpivot format.
- `formula`: The formula used for the symbolic description of the model fitted.

#### Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>
summary.ore.odmSVM

References

Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User’s Guide

See Also

ore.odmSVD

Examples

## For examples see \code{\link{ore.odmSVD}}.

---

summary.ore.odmSVM  Summarize In-Database Support Vector Machine Models

Description

Methods for class ore.odmSVM or summary.ore.odmSVM objects.

Usage

## S3 method for class 'ore.odmSVM'
summary(object,...)

## S3 method for class 'summary.ore.odmSVM'
print(x,...)

Arguments

object       An object of type ore.odmSVM.
x            An object of type summary.ore.odmSVM.
...          Additional arguments affecting the predictions produced.

Details

The function print.summary.ore.odmSVM tries to be smart about formatting the coefficients, standard errors, and so on.

Value

The function summary.ore.odmSVM returns an object of class summary.ore.odmSVM, which is a list with the following components:

call      The function call used to build the model with the given arguments.
settings  A data.frame of settings used to build the model.
fit.values An link[OREbase]{ore.frame} of the actual column and predicted column. For regression, the columns are 'ACTUAL' and 'PREDICTED'. For classification, the columns are 'ACTUAL', 'PREDICTED', 'PROBABILITY'. For anomaly detection, the columns are 'PREDICTED' and 'PROBABILITY'.
residuals  For regression models, an \texttt{link[OREbase][ore.numeric]} vector containing the residual values (PREDICTED - ACTUAL).

coefficients The coefficients of the SVM model, one for each predictor variable. If \texttt{auto.data.prep} was set to \texttt{TRUE} during model build, coefficients will be in the transformed space (after automatic outlier-aware normalization is applied).

formula The \texttt{formula} used for the symbolic description of the model fitted.

Author(s)
Oracle
Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References
Oracle R Enterprise
Oracle Data Mining Concepts
Oracle Data Mining User’s Guide

See Also
\texttt{ore.odmSVM}

Examples

```r
## For examples see \code{\link{ore.odmSVM}}.
```

summary.ore.odmXGB  \textit{Summarize In-Database XGBoost Models}

Description
Methods for class \texttt{ore.odmXGB} or \texttt{summary.ore.odmXGB} objects.

Usage

```r
## S3 method for class 'ore.odmXGB'
summary(object,...)

## S3 method for class 'summary.ore.odmXGB'
print(x,...)
```

Arguments

object  An object of type \texttt{ore.odmXGB}.

x  An object of type \texttt{summary.ore.odmXGB}.

...  Additional arguments affecting the predictions produced.

Details
The function \texttt{print.summary.ore.odmXGB} tries to be smart about formatting the settings, importance, and so on.
Value

The function `summary.ore.odmXGB` returns an object of class `summary.ore.odmXGB`, which is a list with the following components:

- **call**: The function call used to build the model with the given arguments.
- **settings**: A `data.frame` of settings used to build the model.
- **fit.values**: An `ore.frame` of the actual column and predicted column. For regression, the columns are 'ACTUAL' and 'PREDICTED'. For classification, the columns are 'ACTUAL', 'PREDICTED', 'PROBABILITY'.
- **residuals**: For regression models, an `ore.numeric` vector containing the residual values (PREDICTED - ACTUAL).
- **importance**: The attribute importance of the XGB model, one for each predictor variable.
- **formula**: The `formula` used for the symbolic description of the model fitted.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

- Oracle R Enterprise
- Oracle Data Mining Concepts
- Oracle Data Mining User’s Guide

See Also

- `ore.odmXGB`

Examples

```r
## For examples see \code{\link{ore.odmXGB}}.
```

sunflowerplot.ore  Produce a Sunflower Scatter Plot

Description

Creates sunflower plots.

Usage

```r
## S3 method for class 'ore'
sunflowerplot(x, y = NULL, number, log = "", digits = 6,
  xlab = NULL, ylab = NULL, xlim = NULL, ylim = NULL,
  add = FALSE, rotate = FALSE, pch = 16, cex = 0.8,
  cex.fact = 1.5, col = par("col"), bg = NA, size = 1/8,
  seg.col = 2, seg.lwd = 1.5, ...)
```
svd-methods

Arguments

x, y  
ore.number objects.
number, log, digits, xlab, ylab, xlim, ylim, add, rotate, pch, cex, cexfact, col, bg, size, seg.col, seg.lwd,...
See description in sunflowerplot.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

sunflowerplot

svd-methods  Oracle R Enterprise Singular Value Decomposition

Description

Singular value decomposition of ore.frame and ore.tblmatrix data.

Usage

## S4 method for signature 'ore.frame'
svd(x, nu, nv, LINPACK = FALSE)
## S4 method for signature 'ore.tblmatrix'
svd(x, nu, nv, LINPACK = FALSE)

Arguments

x  
An ore.frame or ore.tblmatrix object of a numeric matrix.

nu  
This argument is not supported, as the function does not return the left singular vector matrix u. See details.

nv  
The number of right singular vectors to be computed. The value should be between 0 and ncol(x). If missing, the default value is either the number of rows or the number of columns, whichever is smaller.

LINPACK  
See the description of the LINPACK argument in svd.

Details

This function uses a parallel block update algorithm for singular value decomposition of tall and skinny numeric matrix. The degree of parallelism is regulated by the global option ore.parallel. The function returns the singular value vector d and the right singular vector matrix v. The function does not return the left singular vector matrix u, and thus the argument nu is defunct.
Value

Returns a list object containing the $c$ vector and $v$ matrix components of a singular value decomposition of argument $x$.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

svd

Examples

USARRESTS <- ore.push(USArrests)

svd(USARRESTS)

Description

Performs one and two sample t-tests on vectors of data.

Usage

# one or two sample test
## S4 method for signature 'vector'
t.test(x, y = NULL,
     alternative = c("two.sided", "less", "greater"),
     mu = 0, paired = FALSE, var.equal = FALSE,
     conf.level = 0.95, ...)

# two sample test
## S4 method for signature 'ore.factor'
t.test(x, y = NULL,
     alternative = c("two.sided", "less", "greater"),
     mu = 0, paired = FALSE, var.equal = FALSE,
     conf.level = 0.95, ...)

Arguments

x An ore.number object or an ore.factor object with exactly two levels.

y An optional ore.number object.

alternative A character string specifying the alternative hypothesis and must be one of "two.sided" (default), "greater" or "less".
mu A number specifying the true value of the mean (or difference in means if you are performing a two sample test).

paired A logical value indicating whether to compute a paired t-test.

var.equal A logical value indicating whether to treat the two variances as being equal. If TRUE then the pooled variance is used to estimate the variance otherwise the Welch (or Satterthwaite) approximation to the degrees of freedom is used.

conf.level A numeric value specifying the level of the confidence interval.

... Additional arguments to be passed to or from methods.

Value

Returns an \texttt{t.test} \texttt{htest} object.

Author(s)

Oracle \texttt{<oracle-r-enterprise@oracle.com>}

References

Oracle R Enterprise

See Also

t.test

tally Counts or Tallies Rows by Group

Description

tally is a convenient wrapper for \texttt{summarise} that will either call \texttt{n} or \texttt{sum(n)} depending on whether you're tallying for the first time, or re-tallying. \texttt{count} is similar, but it does the \texttt{group_by} for you. \texttt{count_} uses \texttt{vars} to specify the grouping column names.

Usage

tally(x, wt, sort = FALSE)

count(x, \ldots, wt = NULL, sort = FALSE)

count_(x, vars, wt = NULL, sort = FALSE)

Arguments

\begin{itemize}
\item \texttt{x} A grouped \texttt{ore.frame} object to tally or count.
\item \texttt{wt} If not specified, tally the number of rows. If specified, perform a weighted tally by summing over the weight values.
\item \texttt{sort} Whether to sort output in descending order of \texttt{n}.
\item \texttt{\ldots, vars} Columns to group by.
\end{itemize}
Examples

```
MTCARS <- ore.push(mtcars)
df <- arrange(tally(group_by(MTCARS, cyl), main.cyl), main.cyl)
tally(group_by(MTCARS, cyl), sort = TRUE)

# Multiple tallies progressively roll up the groups
cyl_by_gear <- tally(group_by(MTCARS, cyl, gear), sort = TRUE)
tally(cyl_by_gear, sort = TRUE)
tally(tally(cyl_by_gear))

cyl_by_gear <- tally(group_by(MTCARS, cyl, gear), wt = hp, sort = TRUE)
tally(cyl_by_gear, sort = TRUE)
tally(tally(cyl_by_gear))

cyl_by_gear <- count(MTCARS, cyl, gear, wt = MTCARS["hp"] + mpg, sort = TRUE)
tally(cyl_by_gear, sort = TRUE)
tally(tally(cyl_by_gear))

library(magrittr)
MTCARS %>% group_by(cyl) %>% tally(sort = TRUE)

# count is more succinct and also does the grouping
MTCARS %>% count(cyl) %>% arrange(cyl)
MTCARS %>% count(cyl, wt = hp) %>% arrange(cyl)
MTCARS %>% count("cyl", wt = hp, sort = TRUE)
```

---

text.ore  

Add Text to a Plot

Description

Draws text.

Usage

```
## S3 method for class 'ore'
text(x, y = NULL, labels = seq_along(x), adj = NULL,
pos = NULL, offset = 0.5, vfont = NULL, cex = 1, col = NULL,
font = NULL, ...)
```

Arguments

- **x, y**  
  ore.number objects.
- **labels, adj, pos, offset, vfont, cex, col, font,...**
  See description in text.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise
See Also

text

top_n

Select Top n Rows

Description

This is a convenient wrapper that uses filter and min_rank to select the top n entries in each group, ordered by wt.

Usage

top_n(x, n, wt)

Arguments

x  The ore.frame object to filter.

n  The number of rows to return. If x is grouped, this is the number of rows per group. May include more than n if there are ties.

wt  An optional argument to specify the column to use for ordering. If not specified, defaults to the last column in x.

Examples

MTCARS <- ore.push(mtcars)

# Find top 3 carbs with most hps
carbs <- group_by(MTCARS, carb)
hps <- tally(carbs, hp)
top_n(hps, 3, n)

library(magrittr)
MTCARS %>% group_by(carb) %>% tally(hp) %>% top_n(3)
MTCARS %>% group_by(carb) %>% top_n(1, hp)

var.test-methods  Oracle R Enterprise F Test to Compare Two Variances

Description

Performs an F test to compare the variances of two samples from normal populations.
Usage

# one or two sample test
## S4 method for signature 'vector'
var.test(x, y, ratio = 1,
          alternative = c("two.sided", "less", "greater"),
          conf.level = 0.95, ...)

# two sample test
## S4 method for signature 'ore.factor'
var.test(x, y, ratio = 1,
          alternative = c("two.sided", "less", "greater"),
          conf.level = 0.95, ...)

Arguments

x                   An ore.number object or an ore.factor object with exactly two levels.
y                   An optional ore.number object.
ratio               A numeric value specifying the hypothesized ratio of the population variances of arguments x and y.
alternative         A character string specifying the alternative hypothesis and must be one of "two.sided" (default), "greater" or "less".
conf.level          A numeric value specifying the level of the confidence interval.
...                  Additional arguments.

Value

Returns an var.test htest object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

var.test

Description

Performs one- and two-sample Wilcoxon tests on vectors of data; the latter is also known as the ‘Mann-Whitney’ test.
Usage

```r
## S4 method for signature 'vector'
wilcox.test(x, y = NULL,
    alternative = c("two.sided", "less", "greater"),
    mu = 0, paired = FALSE, exact = NULL, correct = FALSE,
    conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

- `x`: An `ore.number` object.
- `y`: An optional `ore.number` object.
- `alternative`: A character string specifying the alternative hypothesis and must be one of "two.sided" (default), "greater" or "less".
- `mu`: A number specifying an optional parameter used to form the null hypothesis.
- `paired`: A logical value indicating whether to compute a paired test.
- `exact`: Argument not supported.
- `correct`: Argument not supported.
- `conf.int`: Argument not supported.
- `conf.level`: Argument not supported.
- `...`: Additional arguments.

Value

Returns an `wilcox.test` `htest` object.

Author(s)

Oracle `<oracle-r-enterprise@oracle.com>`

References


Oracle R Enterprise

See Also

`wilcox.test`
Index

!,ore.frame-method
(ore.frame-class), 73
!,ore.number-method
(ore.logical-class), 95
* NA
  ore.frame-model.frame, 77
* OREdm
  ore.itemsets-class, 90
  ore.rules-class, 181
* ORE
  as.ore, 16
  clusterhists, 20
  is.ore, 27
  ore-class, 33
  ore-data-image-class, 34
  ore.attach, 41
  ore.character-class, 43
  ore.connect, 44
  ore.const, 47
  ore.create, 50
  ore.datastore, 53
  ore.datastoreSummary, 55
  ore.datetime-class, 57
  ore.delete, 60
  ore.doEval, 61
  ore.exec, 70
  ore.exists, 71
  ore.factor-class, 72
  ore.frame-class, 73
  ore.get, 83
  ore.grant, 85
  ore.hash, 87
  ore.hiveOptions, 88
  ore.impalaOptions, 89
  ore.itemsets-class, 90
  ore.lazyLoad, 91
  ore.list-class, 93
  ore.load, 94
  ore.logical-class, 95
  ore.ls, 96
  ore.make.names, 97
  ore.matrix-class, 98
  ore.move, 100
  ore.number-class, 102
  ore.object-class, 107
  ore.odmAI, 108
  ore.odmDT, 112
  ore.odmEM, 115
  ore.odmESA, 119
  ore.odmGLM, 123
  ore.odmKMeans, 128
  ore.odmNMF, 136
  ore.odmOC, 140
  ore.odmRAAlg, 144
  ore.odmSVD, 150
  ore.odmSVM, 153
  ore.odmXGB, 157
  ore.options, 160
  ore.pull, 173
  ore.raw-class, 177
  ore.recode, 178
  ore.rm, 179
  ore.rules-class, 181
  ore.save, 183
  ore.scriptCreate, 185
  ore.showHiveOptions, 187
  ore.showImpalaOptions, 188
  ore.sync, 193
  ore.toXML, 194
  ore.vector-class, 199
  ore.year, 201
  OREbase-package, 4
  OREdm-package, 6
  OREdplyr-package, 7
  OREds-package, 8
  OREembed-package, 10
  ORExml-package, 14
  partitions, 204
  rules, 211
  summary.ore.odmAssocRules,
    217
  summary.ore.odmDT, 218
  summary.ore.odmEM, 220
  summary.ore.odmESA, 221
  summary.ore.odmESM, 222
  summary.ore.odmGLM, 223

243
INDEX

summary.ore.odmKMeans, 225
summary.ore.odmNB, 226
summary.ore.odmNMF, 227
summary.ore.odmNN, 228
summary.ore.odmOC, 229
summary.ore.odmSVD, 232
summary.ore.odmSVM, 233
summary.ore.odmXGB, 234

*R model
ore.odmRAlg, 144

*anomalydetection
ore.odmSVM, 153

*aplot
lines.ore, 30
ore-coplot, 34
ore-matplot, 35
ore-polygon, 36
ore-polypath, 37
ore-rug, 37
ore-segments, 38
ore-symbols, 39
ore-xspline, 40
points.ore, 207
text.ore, 239

*array
ore-matplot, 35

*association rules
rules, 211
summary.ore.odmAssocRules, 217

*category
ore.crosstab, 52
ore.doEval, 61
ore.freq, 82
ore.summary, 190
ore-vector-aggregate, 197

*character
ore.make.names, 97

*chron
ore.year, 201

*classes
as.ore, 16
is.ore, 27
ore-class, 33
ore-data-image-class, 34
ore.character-class, 43
ore.datetime-class, 57
ore.factor-class, 72
ore.frame-class, 73
ore.itemsets-class, 90
ore.list-class, 93
ore.logical-class, 95
ore.matrix-class, 98
ore.number-class, 102
ore.object-class, 107
ore.raw-class, 177
ore.rules-class, 181
ore.vector-class, 199

*classification
ore.odmAI, 108
ore.odmDT, 112
ore.odmGLM, 123
ore.odmSVM, 153
ore.odmXGB, 157
summary.ore.odmDT, 218
summary.ore.odmGLM, 223
summary.ore.odmNB, 226
summary.ore.odmNN, 228
summary.ore.odmRF, 231
summary.ore.odmSVM, 233
summary.ore.odmXGB, 234

*classif
ore.odmESM, 122
ore.odmNB, 133
ore.odmNN, 138
ore.odmRF, 148

*cluster
clusterhists, 20
ore.odmEM, 115
ore.odmRMeans, 128
ore.odmOC, 140
ore.predict-kmeans, 163
ore.predict-matrix, 165
rules, 211
summary.ore.odmEM, 220
summary.ore.odmKMeans, 225
summary.ore.odmOC, 229

*database
as.ore, 16
clusterhists, 20
is.ore, 27
ore.attach, 41
ore.connect, 44
ore.create, 50
ore.datastore, 53
ore.datastoreSummary, 55
ore.delete, 60
ore.doEval, 61
ore.exec, 70
ore.exists, 71
ore.get, 83
ore.grant, 85
ore.hash, 87
ore.lazyLoad, 91
ore.load, 94
ore.ls, 96
ore.move, 100
ore.odmDT, 112
ore.odmEM, 115
ore.odmESA, 119
ore.odmGLM, 123
ore.odmKMeans, 128
ore.odmNMF, 136
ore.odmOC, 140
ore.odmRAlog, 144
ore.odmSVD, 150
ore.odmSVM, 153
ore.odmXGB, 157
ore.pull, 173
ore.recode, 178
ore.rm, 179
ore.save, 183
ore.scriptCreate, 185
ore.sync, 193
ore.year, 201
OREbase-package, 4
OREdm-package, 6
OREdplyr-package, 7
OREds-package, 8
OREembed-package, 10
ORExml-package, 14
partitions, 204
rules, 211
summary.ore.odmAssocRules, 217
summary.ore.odmDT, 218
summary.ore.odmEM, 220
summary.ore.odmESA, 221
summary.ore.odmESM, 222
summary.ore.odmGLM, 223
summary.ore.odmKMeans, 225
summary.ore.odmNB, 226
summary.ore.odmNMF, 227
summary.ore.odmNN, 228
summary.ore.odmOC, 229
summary.ore.odmRF, 231
summary.ore.odmSVD, 232
summary.ore.odmSVM, 233
summary.ore.odmXGB, 234
* datastore
ore.datastore, 53
ore.datastoreSummary, 55
ore.delete, 60
ore.lazyLoad, 91
ore.load, 94
ore.move, 100
ore.save, 183
* data
ore.attach, 41
ore.doEval, 61
ore.exists, 71
ore.get, 83
ore.pull, 173
* distribution
chisq.test-methods, 19
fitdistr-methods, 23
hist.ore, 25
OREDistributions, 202
* documentation
OREShowDoc, 203
* dplot
hist.ore, 25
ore-xy.coords, 41
* environment
ore.create, 50
ore.ls, 96
ore.options, 160
ore.rm, 179
ore.sync, 193
* error
ore.options, 160
* feature extraction
ore.odmESA, 119
ore.odmNMF, 136
ore.odmSVD, 150
summary.ore.odmESA, 221
summary.ore.odmSVD, 232
* graphics
arrows-methods, 15
t.ore.boxplot.stats, 32
* hplot
boxplot.ore, 18
cdplot.ore, 19
hist.ore, 25
ore-coplot, 34
ore-matplot, 35
ore-smoothScatter, 39
pairs.ore.frame, 204
plot.ore.vector, 206
sunflowerplot.ore, 235
* htest
binom.test-methods, 17
chisq.test-methods, 19
ks.test-methods, 29
prop.test-methods, 207
t.test-methods, 237
var.test-methods, 240
wilcox.test-methods, 241
*iplot
  identify.ore, 26
*iteration
  ore.doEval, 61
*logic
  ore.frame-model.frame, 77
*manip
  ore.sort, 188
*methods
  fitdistr-methods, 23
*model partitions
  partitions, 204
*model settings
  partitions, 204
*models
  ore.frame-model.frame, 77
  ore.odmGLM, 123
  ore.predict, 161
  ore.predict-glm, 162
  ore.predict-lm, 164
  ore.predict-multinom, 167
  ore.predict-nnet, 168
  ore.predict-ore.model, 169
  ore.predict-rpart, 172
  summary.ore.odmGLM, 223
*multivariate
  ore.corr, 48
  ore.frame-factanal, 76
  ore.frame-prcomp, 79
  ore.frame-princomp, 80
  ore.number-multivar, 104
  ore.predict-kmeans, 163
  ore.predict-matrix, 165
  ore.predict-prcomp, 170
  ore.predict-princomp, 171
*nonparametric
  sunflowerplot.ore, 235
*package
  OREbase-package, 4
  OREdm-package, 6
  OREdplyr-package, 7
  OREds-package, 8
  OREda-package, 9
  OREembed-package, 10
  OREgraphics-package, 11
  OREpredict-package, 12
  OREstats-package, 13
  ORExml-package, 14
*print
  ore.options, 160
*programming
  ore.doEval, 61
  ore.grant, 85
  ore.scriptCreate, 185
*regression
  ore.odmAI, 108
  ore.odmGLM, 123
  ore.odmSVM, 153
  ore.odmXGB, 157
  ore.predict-glm, 162
  ore.predict-lm, 164
  ore.predict-ore.model, 169
  summary.ore.odmGLM, 223
  summary.ore.odmNN, 228
  summary.ore.odmSVM, 233
  summary.ore.odmXGB, 234
*robust
  ore.number-univar, 105
  ore.roll, 180
*smooth
  ore.roll, 180
  sunflowerplot.ore, 235
*summary
  ore.summary, 190
*svd
  ore.odmSVD, 150
  svd-methods, 236
*sysdata
  ore.const, 47
*time series
  summary.ore.odmESM, 222
*univar
  ore.number-univar, 105
  ore.rank, 175
  ore.univariate, 195
  ore.vector-ave, 198
*utilities
  ore.getXlevels, 84
  ore.hash, 87
  ore.recode, 178
  ore.toXML, 194
  ore.year, 201
  reorder, 210
  .getXlevels, 84, 85
[,ore.frame,ANY,ANY-method
  (ore.frame-class), 73
[,ore.frame,ANY,missing-method
  (ore.frame-class), 73
[,ore.frame,character,missing-method
  (ore.frame-class), 73
[,ore.frame,logical,missing-method
  (ore.frame-class), 73
[,ore.frame,missing,ANY-method
INDEX

(ore.frame-class), 73
[,ore.frame,missing,missing-method (ore.frame-class), 73
[,ore.frame,numERIC,missing-method (ore.frame-class), 73
[,ore.frame,ore.character,missing-method, ore.vector,ore.logical,missing-method (ore.frame-class), 73
[,ore.frame,ore.logical,missing-method [,ore.frame,character,missing-method (ore.frame-class), 73
[,ore.frame,ore.number,missing-method (ore.frame-class), 73
[,ore.tblmatrix,ANY,ANY-method (ore.matrix-class), 98
[,ore.tblmatrix,ANY,missing-method (ore.matrix-class), 98
[,ore.tblmatrix,missing,ANY-method (ore.matrix-class), 98
[,ore.tblmatrix,missing,missing-method (ore.matrix-class), 98
[,ore.vecmatrix,ANY,ANY-method (ore.matrix-class), 98
[,ore.vector,ANY,ANY-method (ore.vector-class), 199
[,ore.vector,NULL,missing-method (ore.vector-class), 199
[,ore.vector,logical,missing-method (ore.vector-class), 199
[,ore.vector,numeric,missing-method (ore.vector-class), 199
[,ore.list,character,missing-method (ore.list-class), 93
[,ore.list,numERIC,missing-method (ore.list-class), 93
[,ore.tblmatrix,missing,missing-method [,ore.frame,character,missing-method (ore.frame-class), 73
[,ore.frame,logical,missing-method (ore.frame-class), 73
[<-,ore.frame,character,missing-method (ore.frame-class), 73
[<-,ore.frame,logical,missing-method (ore.frame-class), 73
[<-,ore.frame,numeric,missing-method (ore.frame-class), 73
[<-,ore.vector,character,missing-method (ore.vector-class), 199
[<-,ore.vector,logical,missing-method (ore.vector-class), 199
[<-,ore.vector,numeric,missing-method (ore.vector-class), 199
$,ore.data.image-method (ore.data.image-class), 34
$,ore.frame-method (ore.frame-class), 73
$,ore.list-method (ore.list-class), 93
$,<-,ore.frame-method (ore.frame-class), 73
%*,ANY,ore-method (ore.matrix-class), 98
%*,ore.number,ore.number-method (ore.matrix-class), 98
%*,ore.tblmatrix-method (ore.matrix-class), 98
%*,ore.tblmatrix,matrix-method (ore.matrix-class), 98
%*,ore.tblmatrix,number-method (ore.matrix-class), 98
%*,ore.tblmatrix,ore.number-method (ore.matrix-class), 98
%*,ore.tblmatrix,ore.tblmatrix-method (ore.matrix-class), 98
%*,ore.vector,ANY,ANY-method (ore.vector-class), 199
%*,ANY,ore.vector-method (ore.vector-class), 199
%*,ANY,ore.vector-method (ore.vector-class), 199
%*,ANY,ore.vector-method (ore.vector-class), 199
%*,ANY,ore.vector-method (ore.vector-class), 199
%*,ANY,ore.vector-method (ore.vector-class), 199
%*%,ANY,ore-method (ore.matrix-class), 98
%*%,ore.number,ore.number-method (ore.matrix-class), 98
%*%,ore.number,ore.tblmatrix-method (ore.matrix-class), 98
%*%,ore.tblmatrix,matrix-method (ore.matrix-class), 98
%*%,ore.tblmatrix,number-method (ore.matrix-class), 98
%*%,ore.tblmatrix,ore.number-method (ore.matrix-class), 98
%*%,ore.tblmatrix,ore.tblmatrix-method (ore.matrix-class), 98
%*%,ore.vector,ANY,ANY-method (ore.vector-class), 199
%*%,ANY,ore.vector-method (ore.vector-class), 199
%*%,ANY,ore.vector-method (ore.vector-class), 199
%*%,ANY,ore.vector-method (ore.vector-class), 199
(ore.vector-class), 199
%in%, ore.vector, POSIXxT-method
(ore.vector-class), 199
%in%, ore.vector, character-method
(ore.vector-class), 199
%in%, ore.vector, difftime-method
(ore.vector-class), 199
%in%, ore.vector, factor-method
(ore.vector-class), 199
%in%, ore.vector, numeric-method
(ore.vector-class), 199
%in%, ore.vector, ore.vector-method
(ore.vector-class), 199
%*, 99
%in%, 200
abbreviate, 98
aggregate, 197
aggregate, ore.vector-method
(ore.vector-aggregate), 197
all, 216
any, 216
Arith, Date, ore.date-method
(ore.datetime-class), 57
Arith, Date, ore.datetime-method
(ore.datetime-class), 57
Arith, Date, ore.difftime-method
(ore.datetime-class), 57
Arith, difftime, ore.date-method
(ore.datetime-class), 57
Arith, difftime, ore.datetime-method
(ore.datetime-class), 57
Arith, difftime, ore.difftime-method
(ore.datetime-class), 57
Arith, matrix, ore.matrix-method
(ore.matrix-class), 98
Arith, number, ore.date-method
(ore.datetime-class), 57
Arith, number, ore.datetime-method
(ore.datetime-class), 57
Arith, number, ore.difftime-method
(ore.datetime-class), 57
Arith, number, ore.frame-method
(ore.frame-class), 73
Arith, number, ore.matrix-method
(ore.matrix-class), 98
Arith, number, ore.number-method
(ore.number-class), 102
Arith, ore.date, Date-method
(ore.datetime-class), 57
Arith, ore.date, difftime-method
(ore.datetime-class), 57
Arith, ore.date, number-method
(ore.datetime-class), 57
Arith, ore.date, ore.date-method
(ore.datetime-class), 57
Arith, ore.date, ore.datetime-method
(ore.datetime-class), 57
Arith, ore.date, ore.difftime-method
(ore.datetime-class), 57
Arith, ore.date, ore.frame-method
(ore.frame-class), 73
Arith, ore.date, POSIXct-method
(ore.datetime-class), 57
Arith, ore.date, POSIXlt-method
(ore.datetime-class), 57
Arith, ore.datetime, Date-method
(ore.datetime-class), 57
Arith, ore.datetime, difftime-method
(ore.datetime-class), 57
Arith, ore.datetime, number-method
(ore.datetime-class), 57
Arith, ore.datetime, ore.date-method
(ore.datetime-class), 57
Arith, ore.datetime, ore.datetime-method
(ore.datetime-class), 57
Arith, ore.datetime, ore.difftime-method
(ore.datetime-class), 57
Arith, ore.datetime, ore.frame-method
(ore.frame-class), 73
Arith, ore.datetime, POSIXct-method
(ore.datetime-class), 57
Arith, ore.datetime, POSIXlt-method
(ore.datetime-class), 57
Arith, ore.difftime, Date-method
(ore.datetime-class), 57
Arith, ore.difftime, diffitime-method
(ore.datetime-class), 57
Arith, ore.difftime, number-method
(ore.datetime-class), 57
Arith, ore.difftime, ore.date-method
(ore.datetime-class), 57
Arith, ore.difftime, ore.datetime-method
(ore.datetime-class), 57
Arith, ore.difftime, ore.difftime-method
(ore.datetime-class), 57
Arith, ore.difftime, ore.frame-method
(ore.frame-class), 73
Arith, ore.difftime, ore.number-method
(ore.datetime-class), 57

INDEX

Arith, ore.difftime, POSIXct-method
(ore.datetime-class), 57
Arith, ore.difftime, POSIXlt-method
(ore.datetime-class), 57
Arith, ore.frame, missing-method
(ore.frame-class), 73
Arith, ore.frame, number-method
(ore.frame-class), 73
Arith, ore.frame, ore.date-method
(ore.frame-class), 73
Arith, ore.frame, ore.datetime-method
(ore.frame-class), 73
Arith, ore.frame, ore.difftime-method
(ore.frame-class), 73
Arith, ore.frame, ore.frame-method
(ore.frame-class), 73
Arith, ore.frame, ore.number-method
(ore.frame-class), 73
Arith, ore.matrix, matrix-method
(ore.matrix-class), 98
Arith, ore.matrix, missing-method
(ore.matrix-class), 98
Arith, ore.matrix, number-method
(ore.matrix-class), 98
Arith, ore.number, missing-method
(ore.number-class), 102
Arith, ore.number, number-method
(ore.number-class), 102
Arith, ore.number, ore.date-method
(ore.datetime-class), 57
Arith, ore.number, ore.datetime-method
(ore.datetime-class), 57
Arith, ore.number, ore.difftime-method
(ore.datetime-class), 57
Arith, ore.number, ore.frame-method
(ore.frame-class), 73
Arith, ore.number, ore.matrix-method
(ore.matrix-class), 98
Arith, ore.number, ore.number-method
(ore.number-class), 102
Arith, ore.tblmatrix, ore.vecmatrix-method
(ore.matrix-class), 98
Arith, ore.vecmatrix, ore.tblmatrix-method
(ore.matrix-class), 98
Arith, POSIXct, ore.difftime-method
(ore.datetime-class), 57
Arith, POSIXlt, ore.date-method
(ore.datetime-class), 57
Arith, POSIXlt, ore.datetime-method
(ore.datetime-class), 57
Arith, POSIXlt, ore.difftime-method
(ore.datetime-class), 57
arrange, 214, 216, 217
arrange (select), 214
arrange_(select), 214
arrows, 15
arrows (arrows-methods), 15
arrows, ANY-method
arrows-methods, 15
as.character, 200
as.character, ore-method
(ore.vector-class), 199
as.character, ore.date-method
(ore.datetime-class), 57
as.character, ore.datetime-method
(ore.datetime-class), 57
as.character, ore.difftime-method
(ore.datetime-class), 57
as.data.frame, 200
as.data.frame, list-method
(ore.frame-class), 73
as.data.frame, ore.frame-method
(ore.frame-class), 73
as.data.frame, ore.vector-method
(ore.vector-class), 199
as.env (ore.frame-class), 73
as.env, ore.frame-method
(ore.frame-class), 73
as.factor, 200
as.factor, ore.date-method
(ore.datetime-class), 57
as.factor, ore.datetime-method
(ore.datetime-class), 57
as.factor, ore.difftime-method
(ore.datetime-class), 57
as.factor, ore.vector-method
as.integer, 200
as.integer, ore.method
(ore.vector-class), 199
as.logical, 200
as.list, 76
as.list, ore.frame-method
(ore.frame-class), 73
as.logical, ore-method
(ore.vector-class), 199
as.matrix, 76, 200
as.matrix, ore.frame-method
(ore.frame-class), 73
as.matrix, ore.matrix-method
(ore.matrix-class), 98
as.matrix, ore.vector-method
(ore.vector-class), 199
as.numeric, 200
as.numeric, ore-method
(ore.vector-class), 199
as.ore, 16, 27
as.ore, ANY-method (as.ore), 16
as.ore, data.frame-method
(as.ore), 16
as.ore, Date-method (as.ore), 16
as.ore, diffftime-method (as.ore), 16
as.ore, list-method
(ore.list-class), 93
as.ore, matrix-method (as.ore), 16
as.ore, POSIXt-method (as.ore), 16
as.ore, vector-method (as.ore), 16
as.ore.character (as.ore), 16
as.ore.character, ANY-method
(as.ore), 16
as.ore.character, ore.character-method
as.ore.logical, ore-method
(as.ore), 16
as.ore.character, ore.character-method
(ore.datatime-class), 57
as.ore.character, ore.datetime-method
(ore.datatime-class), 57
as.ore.character, ore.difftime-method
(ore.datatime-class), 57
as.ore.date (as.ore), 16
as.ore.date, ANY-method (as.ore), 16
as.ore.date, ore.date-method
(as.ore), 16
as.ore.datetime (as.ore), 16
as.ore.datetime, ANY-method
(as.ore), 16
as.ore.datetime, ore.datetime-method
(as.ore), 16
as.ore.difftime (as.ore), 16
as.ore.difftime, ANY-method
(as.ore), 16
as.ore.difftime, ore.difftime-method
(as.ore), 16
as.ore.factor (as.ore), 16
as.ore.factor, ANY-method
(as.ore), 16
as.ore.factor, ore-method
(as.ore), 16
as.ore.factor, ore.factor-method
(as.ore), 16
as.ore.frame (as.ore), 16
as.ore.frame, ANY-method (as.ore), 16
as.ore.frame, ore-method (as.ore), 16
as.ore.frame, ore.frame-method
(as.ore), 16
as.ore.integer (as.ore), 16
as.ore.integer, ANY-method
(as.ore), 16
as.ore.integer, ore-method
(as.ore), 16
as.ore.integer, ore.integer-method
(as.ore), 16
as.ore.list (ore.list-class), 93
as.ore.list, list-method
(ore.list-class), 93
as.ore.list, ore.list-method
(ore.list-class), 93
as.ore.logical (as.ore), 16
as.ore.logical, ANY-method
(as.ore), 16
as.ore.logical, ore-method
(as.ore), 16
as.ore.logical, ore.logical-method
(as.ore), 16
as.ore.matrix (as.ore), 16
as.ore.matrix, ANY-method
(as.ore), 16
as.ore.matrix, ore-method
(as.ore), 16
as.ore.matrix, ore.matrix-method
(as.ore), 16
as.ore.numeric (as.ore), 16
as.ore.numeric, ANY-method
(as.ore), 16
as.ore.numeric, ore-method
(as.ore), 16
as.ore.numeric, ore.numeric-method
(as.ore), 16
as.ore.object (ore.object-class), 107
as.ore.object, ore.object-method
(ore.object-class), 107
INDEX

as.ore.vector (as.ore), 16
as.vector, 16, 200
as.vector, ore-method
(ore.vector-class), 199
as.vector, ore.date-method
(ore.vector-class), 199
as.vector, ore.datetime-method
(ore.vector-class), 199
as.vector, ore.difftime-method
(ore.vector-class), 199
as.vector, ore.factor-method
(ore.vector-class), 199
as.vector, ore.vector-method
(ore.vector-class), 199
atan2, ore.matrix, ore.matrix-method
(ore.matrix-class), 98
atan2, ore.number, ore.number-method
(ore.number-class), 102
atan2, ore.tblmatrix, ore.vecmatrix-method
(ore.matrix-class), 98
atan2, ore.vecmatrix, ore.tblmatrix-method
(ore.matrix-class), 98
attach, ore.frame-method
(ore.frame-class), 73
ave, 198
ave, ore.vector-method
(ore.vector-ave), 198
backsolve, ANY-method
(ore.matrix-class), 98
besselI, ore.matrix-method
(ore.matrix-class), 98
besselI, ore.number-method
(ore.number-class), 102
besselJ, ore.matrix-method
(ore.matrix-class), 98
besselJ, ore.number-method
(ore.number-class), 102
besselK, ore.matrix-method
(ore.matrix-class), 98
besselK, ore.number-method
(ore.number-class), 102
bessely, ore.matrix-method
(ore.matrix-class), 98
bessely, ore.number-method
(ore.number-class), 102
binom.test, 17, 18
binom.test, ANY-method
(binom.test-methods), 17
binom.test, ore.vector-method
(binom.test-methods), 17
binom.test-methods, 17
boxplot, 18, 63
boxplot.formula (boxplot.ore), 18
boxplot.list (boxplot.ore), 18
boxplot.ore, 18
boxplot.stats, 33
boxplot.stats, ANY-method
(ore-boxplot.stats), 32
boxplot.stats, ore.vector-method
(ore-boxplot.stats), 32
by, 74, 199
by, ore.frame-method
(ore.frame-class), 73
by, ore.vector-method
(ore.vector-class), 199
c, 199
c, ore.vector-method
(ore.vector-class), 199
casefold, 43
casefold, ore.character-method
(ore.character-class), 43
casefold, ore.vector-method
(ore.character-class), 43
cbind, 74
cbind (ore.frame-class), 73
cbind, ore-method
(ore.frame-class), 73
cdplot, 19, 63
cdplot.formula (cdplot.ore), 19
cdplot.ore, 19
character, 43, 72, 73, 78, 93
chartr, 43
chartr, character, character, ore.character-method
(ore.character-class), 43
chartr, character, character, ore.vector-method
(ore.character-class), 43
chisq.test, 20
chisq.test, ANY-method
(chisq.test-methods), 19
chisq.test, ore.vector-method
(chisq.test-methods), 19
chisq.test-methods, 19
clusterhists, 20
clusterhists.ore.odmEM, 117, 118
clusterhists.ore.odmKMeans, 131, 132
cointervals, ANY-method
(ore-coplot), 34
cointervals, ore.vector-method
(ore-coplot), 34
coef (ore.odmSVM), 153
coe, ANY, ore.character-method
(ore.character-class), 199
coerce, ANY, ore.integer-method (ore.vector-class), 199
coerce, ANY, ore.logical-method (ore.vector-class), 199
coerce, ANY, ore.numeric-method (ore.vector-class), 199
coerce, Date, ore.date-method (ore.datetime-class), 57
coerce, Date, ore.datetime-method (ore.datetime-class), 57
colMeans, ore.frame-method (ore.frame-class), 73
colMeans, ore.tblmatrix-method (ore.matrix-class), 98
colMeans, ore.vecmatrix-method (ore.matrix-class), 98
colMeans<-, ore.frame-method (ore.frame-class), 73
colMeans<-, ore.tblmatrix-method (ore.matrix-class), 98
colMeans<-, ore.vecmatrix-method (ore.matrix-class), 98
colnames, ore.frame-method (ore.frame-class), 73
colnames, ore.tblmatrix-method (ore.matrix-class), 98
colnames, ore.vecmatrix-method (ore.matrix-class), 98
confint
confint (ore.odmGLM), 123
cor, 49, 105
cor, ore.frame-method (ore.number-multivar), 104
cor, ore.number-method (ore.number-multivar), 104
complete.cases, ore-method (ore.frame-model.frame), 77
complete.cases, ore-method
contrasts, 42
coplot, 34
coplot, ANY, ANY-method (ore-coplot), 34
coplot, ANY, ore.frame-method (ore-coplot), 34
colMeans, 75, 99
colMeans, ore.frame-method (ore.frame-class), 73
colMeans, ore.tblmatrix-method (ore.matrix-class), 98
colMeans, ore.vecmatrix-method (ore.matrix-class), 98
colnames, 73, 98
INDEX

count. 24
count (tally), 238
count_(tally), 238
cov, 105
cov, ore.frame-method
(ore.number-multivar), 104
cov, ore.number-method
(ore.number-multivar), 104
crossprod, 99
crossprod, ANY, ore-method
(ore.matrix-class), 98
crossprod, ore, ANY-method
(ore.matrix-class), 98
crossprod, ore, ore-method
(ore.matrix-class), 98
crossprod, ore.matrix, missing-method
(ore.matrix-class), 98
crossprod, ore.number, missing-method
(ore.matrix-class), 98
crossprod, ore.tblmatrix, missing-method
(ore.matrix-class), 98
cume_dist (ranking), 208
cut, 103
cut, ore.number-method
(ore.number-class), 102
d (ore.odmSVD), 150
data.frame, 21, 50, 62–64, 74, 108, 114,
117, 123, 131, 134, 139, 142, 143,
149, 155, 159, 211, 219, 233, 235
data.frame (ore.frame-class), 73
data.frame, ore-method
(ore.frame-class), 73
dbeta, 202
dbeta, ore.number-method
(OREDistributions), 202
dGetQuery, 70
dbinom, 202
dbinom, ore.number-method
(OREDistributions), 202
dcauchy, 202
dcauchy, ore.number-method
(OREDistributions), 202
dchisq, 202
dchisq, ore.number-method
(OREDistributions), 202
dense_rank (ranking), 208
desc, 8, 22, 208
deviance, 127
deviance (ore.odmGLM), 123
dexp, 202
dexp, ore.number-method
(OREDistributions), 202
df, 202
df, ore.number-method
(OREDistributions), 202
dgamma, 202
dgamma, ore.number-method
(OREDistributions), 202
dgeom, 202
dgeom, ore.number-method
(OREDistributions), 202
diff, 57, 103
diff, ore.date-method
(ore.datetime-class), 57
diff, ore.datetime-method
(ore.datetime-class), 57
diff, ore.difftime-method
(ore.datetime-class), 57
diff, ore.number-method
(ore.number-class), 102
dim, 73, 98
dim, ore.frame-method
(ore.frame-class), 73
dim, ore.tblmatrix-method
(ore.matrix-class), 98
dim, ore.vecmatrix-method
(ore.matrix-class), 98
dimnames, 73, 98
dimnames, ore.frame-method
(ore.frame-class), 73
dimnames, ore.tblmatrix-method
(ore.matrix-class), 98
dimnames, ore.vecmatrix-method
(ore.matrix-class), 98
dimnames<-, 73, 98
dimnames<-, ore.tblmatrix-method
(ore.matrix-class), 98
dimnames<-, ore.vecmatrix-method
(ore.matrix-class), 98
dist, 166
distinct, 8, 22
distinct_(distinct), 22
dnorm, 202
dnorm, ore.number-method
(OREDistributions), 202
dlogis, 202
dlogis, ore.number-method
(OREDistributions), 202
dnbinom, 203
dnbinom, ore.number-method
(OREDistributions), 202
dnorm, 203
dnorm, ore.number-method
(OREDistributions), 202
dpois, 203
dpois, ore.number-method
   (OREDistributions), 202
dsignrank, 203
dsignrank, ore.number-method
   (OREDistributions), 202
dt, 203
dt, ore.number-method
   (OREDistributions), 202
dunif, 203
dunif, ore.number-method
   (OREDistributions), 202
dweibull, 203
dweibull, ore.number-method
   (OREDistributions), 202
evironment, 41, 42, 47, 71, 75, 76, 83, 96, 179, 193
eval, 75
eval(ore.frame-class), 73
eval, expressionORlanguage, ore.frame-method
   (ore.frame-class), 73
expression, 78
extractAIC, 127
extractAIC(ore.odmGLM), 123
factanal, 77
factanal, ANY-method
   (ore.frame-factanal), 76
factor, 72
factorial, ore.number-method
   (ore.number-class), 102
feature_compare(ore.odmNMF), 136
feature_compare.ore.odmESA
   (ore.odmESA), 119
feature_compare.ore.odmSVD
   (ore.odmSVD), 150
features(ore.odmNMF), 136
features.ore.odmESA(ore.odmESA), 119
features.ore.odmSVD(ore.odmSVD), 150
filter, 31, 214, 216, 217, 240
filter(select), 214
filter_(214, 216, 217
filter_(select), 214
first(nth), 31
fitdistr, 24
fitdistr(fitdistr-methods), 23
fitdistr, ANY-method
   (fitdistr-methods), 23
fitdistr, number-method
   (fitdistr-methods), 23
fitdistr, ore.number-method
   (fitdistr-methods), 23
fitdistr-methods, 23
fitted, 127
fitted(ore.esm), 67
fitted.ore.odmGLM(ore.odmGLM), 123
fivenum, 107, 216
fivenum, ore.number-method
   (ore.number-univar), 105
forecast.ore.esm(ore.esm), 67
formula, 78, 108–110, 113, 114, 116, 117, 
   119, 120, 122–124, 126, 127, 129, 
   131, 133, 134, 136, 137, 139, 
   141–143, 146, 148–151, 153, 155, 
   156, 158, 159, 165, 220–222, 225, 
   226, 228–232, 234, 235
forwardsolve, ANY-method
   (ore.matrix-method), 98
full_join(join), 28
method, 210
get_all_vars, 79
get_all_vars, formula, ore.frame-method
   (ore.frame-model.frame), 77
get_all_vars, terms, ore.frame-method
   (ore.frame-model.frame), 77
getRversion, 64
glm, 126, 162, 224
grepl, 43
grepl, character, ore.character-method
   (ore.character-class), 43
grepl, character, ore.vector-method
   (ore.character-class), 43
group_by, 8, 24, 238
group_by_, 24
group_by_(group_by), 24
group_size(group_by), 24
groups(group_by), 24
gsub, 43
gsub, character, character, ore.character-method
   (ore.character-class), 43
gsub, character, character, ore.vector-method
   (ore.character-class), 43
head, 74, 199
head, ore.frame-method
   (ore.frame-class), 73
head, ore.vector-method
   (ore.vector-class), 199
hist, 26, 63
hist.ore, 25
histogram(ore.odmKMeans), 128
INDEX

is.nan, ore.frame-method
  (ore.frame-class), 73
is.nan, ore.number-method
  (ore.number-class), 102
is.nan, ore.numeric-method
  (ore.number-class), 102
is.ore, 16, 27
is.ore.list (ore.list-class), 93
is.ore.object (ore.object-class), 107
is.vector, 27
is.vector, ore.vector-method
  (ore.frame-class), 199
itemsets (ore.odmAssocRules), 109
join, 8, 28
kmeans, 163, 164
ks.test, 29, 30
ks.test, number, ore.number-method
  (ks.test-methods), 29
ks.test, ore.number, character-method
  (ks.test-methods), 29
ks.test, ore.number, number-method
  (ks.test-methods), 29
ks.test, ore.number, ore.number-method
  (ks.test-methods), 29
ks.test, ore.vector, ore.number-method
  (ks.test-methods), 29
ks.test-methods, 29
last (nth), 31
left_join (join), 28
length, 73, 93, 199, 216
length, ore.frame-method
  (ore.frame-class), 73
length, ore.list-method
  (ore.list-class), 93
length, ore.vector-method
  (ore.frame-class), 199
levels, 72, 78
levels, ore.factor-method
  (ore.factor-class), 72
l_factorial, ore.number-method
  (ore.frame-class), 102
lines, 30
lines.formula (lines.ore), 30
lines.ore, 30
l_m, 164
load, 94
log, ore.frame-method
  (ore.frame-class), 73

histogram.ore.odmEM (ore.odmEM), 115
histogram.ore.odmOC (ore.odmOC), 140
I, ore.vector-method
  (ore.vector-class), 199
identify, 26, 27
identify.ore, 26
ifelse, 95
ifelse, ore.number-method
  (ore.logical-class), 95
importance (ore.odmRF), 148
importance.ore.odmXGB
  (ore.odmXGB), 157
inner_join (join), 28
installed.packages, 64
integer, 72, 73, 93, 98, 102, 103
interaction, 72
interaction (ore.factor-class), 72
interaction, ore.frame-method
  (ore.frame-class), 73
interaction, ore.vector-method
  (ore.factor-class), 72
IQR, 107, 216
IQR, ore.number-method
  (ore.number-univar), 105
is.finite, 74, 103
is.finite, ore.frame-method
  (ore.frame-class), 73
is.finite, ore.number-method
  (ore.number-class), 102
is.finite, ore.numeric-method
  (ore.number-class), 102
is.infinite, 74, 103
is.infinite, ore.frame-method
  (ore.frame-class), 73
is.infinite, ore.number-method
  (ore.number-class), 102
is.infinite, ore.numeric-method
  (ore.number-class), 102
is.matrix, ore.matrix-method
  (ore.matrix-class), 98
is.na, 74, 200
is.na, ore.frame-method
  (ore.frame-class), 73
is.na, ore.numeric-method
  (ore.number-class), 102
is.na, ore.vector-method
  (ore.vector-class), 199
is.nan, 74, 103

kmeans, 163, 164
ks.test, 29, 30
ks.test, number, ore.number-method
  (ks.test-methods), 29
ks.test, ore.number, character-method
  (ks.test-methods), 29
ks.test, ore.number, number-method
  (ks.test-methods), 29
ks.test, ore.number, ore.number-method
  (ks.test-methods), 29
ks.test, ore.vector, ore.number-method
  (ks.test-methods), 29
ks.test-methods, 29
last (nth), 31
left_join (join), 28
length, 73, 93, 199, 216
length, ore.frame-method
  (ore.frame-class), 73
length, ore.list-method
  (ore.list-class), 93
length, ore.vector-method
  (ore.frame-class), 199
levels, 72, 78
levels, ore.factor-method
  (ore.factor-class), 72
l_factorial, ore.number-method
  (ore.frame-class), 102
lines, 30
lines.formula (lines.ore), 30
lines.ore, 30
l_m, 164
load, 94
log, ore.frame-method
  (ore.frame-class), 73

histogram.ore.odmEM (ore.odmEM), 115
histogram.ore.odmOC (ore.odmOC), 140
I, ore.vector-method
  (ore.vector-class), 199
identify, 26, 27
identify.ore, 26
ifelse, 95
ifelse, ore.number-method
  (ore.logical-class), 95
importance (ore.odmRF), 148
importance.ore.odmXGB
  (ore.odmXGB), 157
inner_join (join), 28
installed.packages, 64
integer, 72, 73, 93, 98, 102, 103
interaction, 72
interaction (ore.factor-class), 72
interaction, ore.frame-method
  (ore.frame-class), 73
interaction, ore.vector-method
  (ore.factor-class), 72
IQR, 107, 216
IQR, ore.number-method
  (ore.number-univar), 105
is.finite, 74, 103
is.finite, ore.frame-method
  (ore.frame-class), 73
is.finite, ore.number-method
  (ore.number-class), 102
is.finite, ore.numeric-method
  (ore.number-class), 102
is.infinite, 74, 103
is.infinite, ore.frame-method
  (ore.frame-class), 73
is.infinite, ore.number-method
  (ore.number-class), 102
is.infinite, ore.numeric-method
  (ore.number-class), 102
is.matrix, ore.matrix-method
  (ore.matrix-class), 98
is.na, 74, 200
is.na, ore.frame-method
  (ore.frame-class), 73
is.na, ore.numeric-method
  (ore.number-class), 102
is.na, ore.vector-method
  (ore.vector-class), 199
is.nan, 74, 103

kmeans, 163, 164
ks.test, 29, 30
ks.test, number, ore.number-method
  (ks.test-methods), 29
ks.test, ore.number, character-method
  (ks.test-methods), 29
ks.test, ore.number, number-method
  (ks.test-methods), 29
ks.test, ore.number, ore.number-method
  (ks.test-methods), 29
ks.test, ore.vector, ore.number-method
  (ks.test-methods), 29
ks.test-methods, 29
last (nth), 31
left_join (join), 28
length, 73, 93, 199, 216
length, ore.frame-method
  (ore.frame-class), 73
length, ore.list-method
  (ore.list-class), 93
length, ore.vector-method
  (ore.frame-class), 199
levels, 72, 78
levels, ore.factor-method
  (ore.factor-class), 72
l_factorial, ore.number-method
  (ore.frame-class), 102
lines, 30
lines.formula (lines.ore), 30
lines.ore, 30
l_m, 164
load, 94
log, ore.frame-method
  (ore.frame-class), 73
INDEX

log, ore.matrix-method
(ore.matrix-class), 98
log, ore.number-method
(ore.number-class), 102
logb, ore.frame-method
(ore.frame-class), 73
logb, ore.matrix-method
(ore.matrix-class), 98
logb, ore.number-method
(ore.number-class), 102
Logic, number, ore.frame-method
(ore.frame-class), 73
Logic, number, ore.number-method
(ore.logical-class), 95
Logic, ore.frame, ore.frame-method
(ore.frame-class), 73
Logic, ore.frame, ore.number-method
(ore.logical-class), 95
Logic, ore.number, ore.frame-method
(ore.frame-class), 73
Logic, ore.number, ore.number-method
(ore.logical-class), 95
logical, 74, 95
logLik, 127
logLik(ore.odmGLM), 123
mad, 107
mad, ore.number-method
(ore.number-univar), 105
make.names, 98
Math, ore.frame-method
(ore.frame-class), 73
Math, ore.matrix-method
(ore.matrix-class), 98
Math, ore.number-method
(ore.number-class), 102
matlines, ANY-method
(ore-matplot), 35
matlines, ore-method
(ore-matplot), 35
matplot, 35, 36, 64
matplot, ANY-method(ore-matplot), 35
matplot, ore-method(ore-matplot), 35
matpoints, ANY-method
(ore-matplot), 35
matpoints, ore-method
(ore-matplot), 35
matrix, 166
max, 216, 217
max.col, 75, 99
max.col, ore.frame-method
(ore.frame-class), 73
max.col, ore.tblmatrix-method
(ore.matrix-class), 98
max.col, ore.vecmatrix-method
(ore.matrix-class), 98
mean, 216, 217
mean, ore.matrix-method
(ore.matrix-class), 98
mean, ore.number-method
(ore.number-class), 102
median, 107, 216
median, ore.vector-method
(ore.number-univar), 105
merge, 74
merge, data.frame, ore.frame-method
(ore.frame-class), 73
merge, ore.frame, data.frame-method
(ore.frame-class), 73
merge, ore.frame, ore.frame-method
(ore.frame-class), 73
min, 216, 217
min_rank, 240
min_rank(ranking), 208
model.frame, 79
model.frame, formula-method
(ore.frame-model.frame), 77
model.frame, terms-method
(ore.frame-model.frame), 77
model.matrix, 79
model.matrix, formula-method
(ore.frame-model.frame), 77
model.matrix, terms-method
(ore.frame-model.frame), 77
multinom, 167
mutate, 31, 214, 216, 217
mutate(select), 214
mutate_(select), 214
n, 8, 31
n_distinct, 8, 32
n_groups(group_by), 24
na.action, 79
na.omit, ore.frame-method
(ore.frame-model.frame), 77
na.omit, ore.vector-method
(ore.frame-model.frame), 77
names, 73, 93, 199
INDEX

names, ore.frame-method  
(ore.frame-class), 73
names, ore.list-method  
(ore.list-class), 93
names, ore.vector-method  
(ore.vector-class), 199
names<-, ore.frame-method  
(ore.frame-class), 73
names<-, ore.vector-method  
(ore.vector-class), 199
nchar, 43
nchar, ore.character-method  
(ore.character-class), 43
nchar, ore.vector-method  
(ore.character-class), 43
NCOL, 73, 98
ncol, 73, 98
NCOL, ore.frame-method  
(ore.frame-class), 73
ncol, ore.frame-method  
(ore.frame-class), 73
NCOL, ore.matrix-method  
(ore.matrix-class), 98
ncol, ore.matrix-method  
(ore.matrix-class), 98
ncol, ore.tblmatrix-method  
(ore.matrix-class), 98
ncol, ore.vecmatrix-method  
(ore.matrix-class), 98
nlevels, 72
nlevels, ore.factor-method  
(ore.factor-class), 72
nnet, 168
nobs, 127
nobs (ore.odmGLM), 123
NROW, 73, 98
nrow, 73, 98
NROW, ore.frame-method  
(ore.frame-class), 73
nrow, ore.frame-method  
(ore.frame-class), 73
NROW, ore.matrix-method  
(ore.matrix-class), 98
nrow, ore.matrix-method  
(ore.matrix-class), 98
nrow, ore.tblmatrix-method  
(ore.matrix-class), 98
nrow, ore.vecmatrix-method  
(ore.matrix-class), 98
nth, 8, 31
ntile (ranking), 208
numeric, 102
options, 161
order (ore.vector-class), 199
order, ore.vector-method  
(ore.vector-class), 199
ore.boxplot.stats, 32
ore-class, 33
ore-coplot, 34
ore-data-image  
(ore-data-image-class), 34
ore-data-image-class, 34
ore-matlines (ore-matplot), 35
ore-matplot, 35
ore-matpoints (ore-matplot), 35
ore-polygon, 36
ore-polypath, 37
ore-rug, 37
ore-segments, 38
ore-smoothScatter, 39
ore-symbols, 39
ore-xspline, 40
ore-xy.coords, 41
ore.attach, 41, 45, 46, 51, 71, 84, 97, 179, 194
ore.character, 5, 33, 58, 59, 72, 73, 76, 96, 99, 104, 173, 177, 199–201
ore.character  
(ore.character-class), 43
ore.character-class, 43
ore.connect, 7, 42, 44, 51, 71, 84, 97, 179, 193, 194
ore.const, 47
ore.corr, 10, 48
ore.create, 50
ore.create, data.frame-method  
(ore.create), 50
ore.create, ore.frame-method  
(ore.create), 50
ore.create, ore.odmGLM  
(ore.create), 50
ore.crosstab, 10, 52, 82, 83
ore.datastore, 9, 53, 56, 61, 86, 92, 94, 101, 161, 184
ore.datastoreSummary, 9, 54, 55, 61, 86, 92, 94, 101, 161, 184
ore.date, 173, 201, 202
ore.date (ore.datetime-class), 57
ore.date-class  
(ore.datetime-class), 57
ore.datetime, 5, 33, 44, 72, 76, 96, 99, 104, 173, 177, 201, 202
ore.datetime  
(ore.datetime-class), 57
ore.datetime-class, 57
ore.delete, 9, 54, 56, 60, 92, 94, 101, 184
ore.detach (ore.attach), 41
ore.month, ore.date-method (ore.year), 201
ore.month, ore.datetime-method (ore.year), 201
ore.month, POSIXt-method (ore.year), 201
ore.move, 54, 56, 61, 94, 100, 184
ore.na.extract (ore.options), 160
ore.number, 5, 15, 18, 19, 23, 26, 29, 30, 33, 36–41, 44, 49, 72, 75, 76, 96, 99, 103, 104, 106, 177, 180, 201, 203, 207, 236, 237, 239, 241, 242
ore.number (ore.number-class), 102
ore.number-class, 102
ore.number-multivar, 104, 107
ore.number-univar, 105, 105, 181
ore.numeric, 87, 156, 159, 162, 165, 173, 181, 200, 202
ore.numeric (ore.number-class), 102
ore.numeric-class (ore.number-class), 102
ore.object, 11, 65, 93
ore.object (ore.object-class), 107
ore.object-class, 107
ore.odmAI, 108
ore.odmAssocRules, 91, 109, 182, 205, 212, 218
ore.odmDT, 109, 112, 123, 127, 135, 140, 149, 156, 159, 219
ore.odmEM, 22, 115, 132, 143, 212, 220, 221
ore.odmESA, 119, 138, 152, 221, 222
ore.odmESM, 122, 122, 222, 223
ore.odmGLM, 109, 115, 123, 123, 135, 140, 149, 156, 159, 225
ore.odmKMeans, 22, 108, 110, 114, 116, 118, 120, 125, 128, 134, 137, 141, 143, 145, 151, 154, 205, 212, 222, 225, 226
ore.odmNB, 109, 115, 127, 133, 156, 226, 227
ore.odmNMF, 121, 136, 152, 227, 228
ore.odmNN, 138, 139, 229
ore.odmOC, 22, 118, 132, 140, 143, 205, 212, 230
ore.odmRAlg, 144
ore.odmRF, 148, 148, 231
ore.odmSVD, 121, 138, 150, 232, 233
ore.odmSVM, 109, 115, 123, 127, 135, 140, 149, 153, 159, 234
ore.odmXGB, 157, 158, 235
ore.options, 65, 160, 174, 184
ore.parallel (ore.options), 160
ore.predict, 161, 163–172
ore.predict, ANY-method (ore.predict), 161
ore.predict, glm-method (ore.predict-glm), 162
ore.predict, kmeans-method (ore.predict-kmeans), 163
ore.predict, lm-method (ore.predict-lm), 164
ore.predict, matrix-method (ore.predict-matrix), 165
ore.predict, multinom-method (ore.predict-multinom), 167
ore.predict, nnet.formula-method (ore.predict-nnet), 168
ore.predict, ore.model-method (ore.predict-ore.model), 169
ore.predict, prcomp-method (ore.predict-prcomp), 170
ore.predict, princomp-method (ore.predict-princomp), 171
ore.predict, rpart-method (ore.predict-rpart), 172
ore.predict-glm, 162
ore.predict-kmeans, 163
ore.predict-lm, 164
ore.predict-matrix, 165
ore.predict-multinom, 167
ore.predict-nnet, 168
ore.predict-ore.model, 169
ore.predict-prcomp, 170
ore.predict-princomp, 171
ore.predict-rpart, 172
ore.pull, 173
ore.pull, ANY-method (ore.pull), 173
ore.pull, list-method (ore.pull), 173
ore.pull, ore.date-method (ore.pull), 173
ore.pull, ore.factor-method (ore.pull), 173
ore.pull, ore.frame-method (ore.pull), 173
ore.pull, ore.integer-method (ore.pull), 173
ore.pull, ore.itemsets-method (ore.itemsets-class), 90
ore.pull, ore.list-method
ore.univariate, 10, 195
ore.vecmatrix(ore.matrix-class), 98
ore.vecmatrix-class
   (ore.matrix-class), 98
ore.vector, 5, 17, 20, 22, 32, 33, 44, 59,
   62, 72–74, 76, 87, 88, 95, 96, 98, 99,
   104, 106, 126, 127, 160, 174, 177,
   178, 180, 181, 197–199, 206, 208,
   210
ore.vector(ore.vector-class), 199
ore.vector-aggregate, 105, 107, 181,
   197
ore.vector-ave, 198
ore.vector-class
ore.year,Date-method
ore.year,ore.date-method
ore.year
ore.year,POSIXt-method
OREbase (OREbase-package), 4
OREbase-package, 4
OREDistributions, 202
OREDm (OREDm-package), 6
OREDm-package, 6
OREDplyr (OREDplyr-package), 7
OREDplyr-package, 7
OREds (OREds-package), 8
OREds-package, 8
OREda (OREda-package), 9
OREda-package, 9
OREembed (OREembed-package), 10
OREembed-package, 10
OREgraphics
   (OREgraphics-package), 11
OREgraphics-package, 11
OREpredict (OREpredict-package), 12
OREpredict-package, 12
OREShowDoc, 203
OREStats (OREStats-package), 13
OREStats-package, 13
ORExml (ORExml-package), 14
ORExml-package, 14
packageVersion, 64
pairs, 63, 204
pairs.formula(pairs.ore.frame), 204
pairs.ore.frame, 204
partitions, 109, 111, 115, 118, 121, 123,
   127, 132, 135, 138, 140, 143, 146,
   149, 152, 156, 159, 204
paste, 43
paste(ore.character-class), 43
paste, ore.vector-method
   (ore.character-class), 43
pbeta, ore.number-method
   (OREDistributions), 202
pbinom, ore.number-method
   (OREDistributions), 202
pcauchy, ore.number-method
   (OREDistributions), 202
pchisq, ore.number-method
   (OREDistributions), 202
percent_rank (ranking), 208
pexp, ore.number-method
   (OREDistributions), 202
pf, ore.number-method
   (OREDistributions), 202
pgamma, ore.number-method
   (OREDistributions), 202
pgeometric, ore.number-method
   (OREDistributions), 202
pnorm, ore.number-method
   (OREDistributions), 202
plogis, ore.number-method
   (OREDistributions), 202
plot, 63, 206
plot (plot.ore.vector), 206
plot.ore.vector, 206
pmax, 200
pmax (ore.vector-class), 199
pmax, vector-method
   (ore.vector-class), 199
pmin, 200
pmin (ore.vector-class), 199
pmin, vector-method
   (ore.vector-class), 199
pnbinom, ore.number-method
   (OREDistributions), 202
png, 62, 63
pnorm, ore.number-method
   (OREDistributions), 202
points, 207
points.formula(points.ore), 207
points.ore, 207
polygon, 36
polygon, ANY-method(ore-polygon),
INDEX

36 polygon, ore-method (ore-polygon), 36
polypath, 37
polypath, ANY-method (ore-polypath), 37
polypath, ore-method (ore-polypath), 37
POSIXt, 57
ppois, ore.number-method (OREDistributions), 202
prcomp, 79, 80, 170
prcomp, ANY-method (ore.frame-prcomp), 79
predict, 127
predict (ore.esm), 67
predict (ore.odmGLM), 123
predict.glm, 163
predict.lm, 165
predict.multinom, 167
predict.nnet, 168
predict.ore.odmDT (ore.odmDT), 112
predict.ore.odmEM (ore.odmEM), 115
predict.ore.odmESA (ore.odmESA), 119
predict.ore.odmKMeans (ore.odmKMeans), 128
predict.ore.odmNB (ore.odmNB), 133
predict.ore.odmNMF (ore.odmNMF), 136
predict.ore.odmNN (ore.odmNN), 138
predict.ore.odmOC (ore.odmOC), 140
predict.ore.odmRAlg (ore.odmRAlg), 144
predict.ore.odmRF (ore.odmRF), 148
predict.ore.odmSVD (ore.odmSVD), 150
predict.ore.odmSVM (ore.odmSVM), 153
predict.ore.odmXGB (ore.odmXGB), 157
predict.rpart, 172
princomp, 81, 171
princomp, ANY-method (ore.frame-princomp), 80
print.ore.odmAssocRules (ore.odmAssocRules), 109
print.ore.odmEM (ore.odmEM), 115
print.ore.odmGLM (ore.odmGLM), 123
print.ore.odmKMeans (ore.odmKMeans), 128
print.ore.odmNN (ore.odmNN), 138
print.ore.odmOC (ore.odmOC), 140
print.ore.odmRAlg (ore.odmRAlg), 144
print.ore.odmRF (ore.odmRF), 148
print.ore.odmSVM (ore.odmSVM), 153
print.ore.odmXGB (ore.odmXGB), 157
print.summary.ore.odmAssocRules (summary.ore.odmAssocRules), 217
print.summary.ore.odmDT (summary.ore.odmDT), 218
print.summary.ore.odmEM (summary.ore.odmEM), 220
print.summary.ore.odmESA (summary.ore.odmESA), 221
print.summary.ore.odmESM (summary.ore.odmESM), 222
print.summary.ore.odmGLM (summary.ore.odmGLM), 223
print.summary.ore.odmKMeans (summary.ore.odmKMeans), 225
print.summary.ore.odmNB (summary.ore.odmNB), 226
print.summary.ore.odmNMF (summary.ore.odmNMF), 227
print.summary.ore.odmNN (summary.ore.odmNN), 228
print.summary.ore.odmOC (summary.ore.odmOC), 229
print.summary.ore.odmRAlg (ore.odmRAlg), 144
print.summary.ore.odmRF (summary.ore.odmRF), 231
print.summary.ore.odmSVD (summary.ore.odmSVD), 232
print.summary.ore.odmSVM (summary.ore.odmSVM), 233
print.summary.ore.odmXGB (summary.ore.odmXGB), 234
prod, 216
prop.test, 208
prop.test, ANY-method (prop.test-methods), 207
prop.test, ore.vector-method (prop.test-methods), 207
prop.test-methods, 207
psignrank, ore.number-method (OREDistributions), 202
pt, ore.number-method (OREDistributions), 202
INDEX

(ore.matrix-class), 98
rowSums, ore.vecmatrix-method (ore.matrix-class), 98
rpart, 172
RShowDoc, 203
rug, 37, 38
rug, ANY-method (ore-rug), 37
rug, ore.vector-method (ore-rug), 37
rules, 111, 182, 211
rules.ore.odmEM, 118
rules.ore.odmKMeans, 132
S4groupGeneric, 75, 95, 98, 103, 200
sample, 8, 213
take sample_frac (sample), 213
take sample_n (sample), 213
save, 94
take scale, 75, 80, 99, 103
take scale, ore.frame-method (ore.frame-class), 73
take scale, ore.number-method (ore.number-class), 102
take scale, ore.tblmatrix-method (ore.matrix-class), 98
take scale, ore.vecmatrix-method (ore.matrix-class), 98
sd, 107, 216
take sd, ore.number-method (ore.number-univar), 105
take search, 41, 42
take segments, 38
take segments, ANY-method (ore-segments), 38
take segments, ore.vector-method (ore-segments), 38
take select, 8, 214, 214, 216, 217
take select, 214, 216, 217
select, ANY-method (select), 214
take settings, 146
take settings (partitions), 204
take show, ore-data-image-method (ore-data-image-class), 34
take show, ore.frame-method (ore.frame-class), 73
take show, ore.list-method (ore.list-class), 93
take show, ore.matrix-method (ore.matrix-class), 98
take show, ore.object-method (ore.object-class), 107
take show, ore.vector-method (ore.vector-class), 199
slice, 8, 214, 215, 217
slice, 214, 216, 217
slice, (slice), 215
smoothScatter, 39, 63
smoothScatter, ANY-method (ore-smoothScatter), 39
smoothScatter, ore-method (ore-smoothScatter), 39
solve, ANY, ore-method (ore.matrix-class), 98
solve, ore, ANY-method (ore.matrix-class), 98
solve, ore, missing-method (ore.matrix-class), 98
solve, ore, ore-method (ore.matrix-class), 98
take sort, 200
take sort, ore.vector-method (ore.vector-class), 199
take split, 74, 199
split, ore.frame-method (ore.frame-class), 73
split, ore.vector-method (ore.vector-class), 199
take sub, 43
sub, character, character, ore.character-method (ore.character-class), 43
sub, character, character, ore.vector-method (ore.character-class), 43
take subset, 74
take subset, ore.frame-method (ore.frame-class), 73
take subset, ore.itemsets-method (ore.itemsets-class), 90
take subset, ore.rules-method (ore.rules-class), 181
substr, 43
substr, ore.character-method (ore.character-class), 43
substr, ore.vector-method (ore.character-class), 43
take substring, 43
substring, ore.character-method (ore.character-class), 43
substring, ore.vector-method (ore.character-class), 43
take sum, 216, 238
summarise, 8, 24, 31, 214, 216, 216, 238
summarise, 214, 216, 217
summarise, (summarise), 216
summary, 72, 216
summary, ore.date-method
INDEX

(ore.datetime-class), 57
summary, ore.datetime-method
(ore.datetime-class), 57
summary, ore.difftime-method
(ore.datetime-class), 57
summary, ore.factor-method
(ore.factor-class), 72
Summary, ore.frame-method
(ore.frame-class), 73
summary, ore.frame-method
(ore.frame-class), 73
Summary, ore.matrix-method
(ore.matrix-class), 98
summary, ore.number-method
(ore.number-class), 102
summary, ore.tblmatrix-method
(ore.matrix-class), 98
Summary, ore.vector-method
(ore.vector-class), 199
summary, ore.odmAssocRules, 217
summary, ore.odmDT, 114, 218
summary, ore.odmEM, 220
summary, ore.odmESA, 221
summary, ore.odmESM, 222
summary, ore.odmGLM, 127, 223
summary, ore.odmKMeans, 225
summary, ore.odmNB, 226
summary, ore.odmNMF, 227
summary, ore.odmNN, 228
summary, ore.odmOC, 229
summary, ore.odmRAlg
(ore.odmRAlg), 144
summary, ore.odmRF, 231
summary, ore.odmSVD, 232
summary, ore.odmSVM, 233
summary, ore.odmXGB, 234
sunflowerplot, 236
sunflowerplot.ore, 235
svd, 236, 237
svd(svd-methods), 236
svd, ore.frame-method
(svd-methods), 236
svd, ore.tblmatrix-method
(svd-methods), 236
svd, ore.vecmatrix-method
(svd-methods), 236
svd-methods, 236
symbols, 40
symbols, ANY-method(ore-symbols), 39
symbols, ore.vector-method
(ore-symbols), 39

t. 99
t, ore-method(ore.matrix-class), 98
t, ore.tblmatrix-method
(ore.matrix-class), 98
t, ore.vecmatrix-method
(ore.matrix-class), 98
t.test. 238
t.test (t.test-methods), 237
t.test, formula-method
(t.test-methods), 237
t.test, missing-method
(t.test-methods), 237
t.test, ore.factor-method
(t.test-methods), 237
t.test, ore.logical-method
(t.test-methods), 237
t.test, vector-method
(t.test-methods), 237
t.test-methods, 237
table, 200, 226
table (ore.vector-class), 199
table, ore.vector-method
(ore.vector-class), 199
tabulate, 103
tabulate, ore.matrix-method
(ore.matrix-class), 98
tabulate, ore.number-method
(ore.number-class), 102
tail, 74, 199
tail, ore.frame-method
(ore.frame-class), 73
tail, ore.vector-method
(ore.vector-class), 199
tapply, 199
tapply, ANY, ore.vector-method
(ore.vector-class), 199
tapply, ore.vector, list-method
(ore.vector-class), 199
tapply, ore.vector, ore.vector-method
(ore.vector-class), 199
tapply, ore.vector, vector-method
(ore.vector-class), 199
tcrossprod, 99
tcrossprod, ANY, ore-method
(ore.matrix-class), 98
tcrossprod, ore, ANY-method
(ore.matrix-class), 98
tcrossprod, ore, missing-method
(ore.matrix-class), 98
tcrossprod, ore, ore-method

INDEX

266

(ore.matrix-class), 98
terms, 78, 127, 224
terms, formula-method
(ore.frame-model.frame), 77
text, 239, 240
text.formula (text.ore), 239
text.ore, 239
tolower, 43
tolower, ore.character-method
(ore.character-class), 43
tolower, ore.vector-method
(ore.character-class), 43
top_n, 8, 24, 240
toupper, 43
toupper, ore.character-method
(ore.character-class), 43
toupper, ore.vector-method
(ore.character-class), 43
transform, 75
transform, ore.frame-method
(ore.frame-class), 73
transmute, 214, 216, 217
transmute (select), 214
transmute_ 214, 216, 217
transmute_ (select), 214
trunc, 58
trunc, ore.date-method
(ore.datetime-class), 57
trunc, ore.datetime-method
(ore.datetime-class), 57
u (ore.odmSVD), 150
ungroup (group_by), 24
unique, 74, 200
unique, ore.frame-method
(ore.frame-class), 73
unique, ore.vector-method
(ore.vector-class), 199
unlist, ore.frame-method
(ore.frame-class), 73
v (ore.odmSVD), 150
var, 107, 216
var, ore.frame-method
(ore.number-univar), 105
var, ore.number-method
(ore.number-univar), 105
var.test, 241
var.test, formula-method
(var.test-methods), 240
var.test, missing-method
(var.test-methods), 240
var.test, ore.factor-method
(var.test-methods), 240
var.test, ore.logical-method
(var.test-methods), 240
var.test, vector-method
(var.test-methods), 240
var.test-methods, 240
weight (ore.odmNN), 138
wilcox.test, 242
wilcox.test, ANY-method
(wilcox.test-methods), 241
wilcox.test, formula-method
(wilcox.test-methods), 241
wilcox.test, missing-method
(wilcox.test-methods), 241
wilcox.test, vector-method
(wilcox.test-methods), 241
wilcox.test-methods, 241
with, 75
with, ore.frame-method
(ore.frame-class), 73
within, 75
within, ore.frame-method
(ore.frame-class), 73
xor, 75, 95
xor, number, ore.frame-method
(ore.frame-class), 73
xor, number, ore.number-method
(ore.logical-class), 95
xor, ore.frame, number-method
(ore.frame-class), 73
xor, ore.frame, ore.frame-method
(ore.frame-class), 73
xor, ore.frame, ore.number-method
(ore.frame-class), 73
xor, ore.number, number-method
(ore.logical-class), 95
xor, ore.number, ore.frame-method
(ore.frame-class), 73
xor, ore.number, ore.number-method
(ore.logical-class), 95
xspline, 40
xspline, ANY-method (ore-xspline), 40
xspline, ore-method (ore-xspline), 40
xtabs, 52
xy.coords, 41
xy.coords (ore-xy.coords), 41
xy.coords, ANY-method
(ore-xy.coords), 41
xy.coords, ore-method
  (ore-xy.coords), 41

zapsmall, 103
zapsmall, ore.number-method
  (ore.number-class), 102