## Package ‘matman’

December 13, 2021

**Type** Package  
**Title** Material Management  
**Version** 1.1.3  
**Date** 2021-12-13  
**Maintainer** Leon Binder <leon.binder@th-deg.de>  
**Description** A set of functions, classes and methods for performing ABC and ABC/XYZ analyses, identifying overperforming, underperforming and constantly performing items, and plotting, analyzing as well as predicting the temporal development of items.  
**License** GPL-3  
**Depends** R (>= 3.5.0), shiny  
**Imports** methods, graphics, stats, utils, data.table, dplyr, tidyr, tidyselect, plotly, DT, shinydashboard, shinyWidgets, forecast, parsedate, lubridate  
**Encoding** UTF-8  
**LazyData** true  
**LazyLoad** true  
**RoxygenNote** 7.1.2  
**NeedsCompilation** no  
**Author** Leon Binder [cre, aut], Bernhard Bauer [aut], Michael Scholz [aut]  
**Repository** CRAN  
**Date/Publication** 2021-12-13 09:30:02 UTC

### R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>matman-package</td>
<td>2</td>
</tr>
<tr>
<td>ABCXYZComparison-class</td>
<td>3</td>
</tr>
<tr>
<td>ABCXYZData-class</td>
<td>4</td>
</tr>
<tr>
<td>aggregateData</td>
<td>5</td>
</tr>
</tbody>
</table>

1
Description

A set of functions, classes and methods for performing ABC and ABC/XYZ analyses, identifying overperforming, underperforming and constantly performing items, and plotting, analyzing as well as predicting the temporal development of items.

Details

Package: matman
Type: Package
Version: 1.1.4
Date: 2021-11-15
License: GPL-3
Depends: R (>= 3.5.0), stats
Description

This S4 class represents the result of a comparison of two ABC/XYZ analysis results.

Slots

data (data.frame) The comparison result as data.frame.
type (character) The type of the analysis that has been performed. This is either ‘abc’ or ‘abcxyz’.
valueDiff (numeric) The difference between the value of an item in ABC/XYZ analysis A and the value of the same item in ABC/XYZ analysis B that is required to consider the item in the comparison.
xyzCoefficientDiff (numeric) The difference between the xyz coefficient of an item in ABC/XYZ analysis A and the xyz coefficient of the same item in ABC/XYZ analysis B that is required to consider the item in the comparison.
unequalABC (logical) If TRUE only items are returned, where the ABC-Classes are different. If FALSE only items are returned, where the ABC-Classes are equal. If NA, no further restriction takes place based on the column ABC.
unequalXYZ (logical) If TRUE only items are returned, where the XYZ-Classes are different. If FALSE only items are returned, where the XYZ-Classes are equal. If NA, no further restriction takes place based on the column XYZ.

Objects from the Class

Objects can be created by calling the function compare function. This S4 class represents the result of a comparison of two ABC/XYZ analysis results.

Author(s)

Leon Binder <leon.binder@th-deg.de>
Bernhard Bauer <bernhard.bauer@th-deg.de>
Michael Scholz <michael.scholz@th-deg.de>
Examples

data("Amount")
data1 = Amount[sample(1:nrow(Amount), 1000),]
data2 = Amount[sample(1:nrow(Amount), 1000),]
abcxyzData1 = computeABCXYZAnalysis(data1, value = "value", item = "item", timestamp = "date",
temporalAggregation = "day", XY = 0.5, YZ = 1)
abcxyzData2 = computeABCXYZAnalysis(data2, value = "value", item = "item", timestamp = "date",
temporalAggregation = "day", XY = 0.5, YZ = 1)
comparison = compare(abcxyzData1, abcxyzData2)
comparison

ABCXYZData-class  Class ABCXYZData

Description

This S4 class represents the result of an ABC/XYZ analysis.

Slots

data (data.frame) The result table of an ABC/XYZ analysis.
type (character) The type of the analysis that has been performed. This is either 'abc' or 'abcxyz'.
value (character) The name of the value column in the result table.
item (character) Vector of the names of the item columns in the result table.

Objects from the Class

Objects can be created by calling the function computeABCXYZ. This S4 class represents the result of an ABC/XYZ analysis.

Author(s)

Leon Binder <leon.binder@th-deg.de>
Bernhard Bauer <bernhard.bauer@th-deg.de>
Michael Scholz <michael.scholz@th-deg.de>

Examples

data("Amount")
abcResult = computeABCXYZAnalysis(data = Amount,
   value = "value",
   item = "item",
   timestamp = "date")
abcResult
aggregateData

Performs a temporal aggregation of a data frame

Description

Aggregates a data frame based on a timestamp column to days, weeks, months, quarters, years or total.

Usage

aggregateData(
  data,
  value = NULL,
  item,
  timestamp,
  temporalAggregation = c("day", "week", "month", "quarter", "year", "total"),
  fiscal = 1,
  aggregationFun = sum
)

Arguments

data  Data frame or matrix on which the ABC analysis is performed.
value Name(s) of the column variable(s) that contains the values for the ABC and XYZ analysis.
item Names of the columns including the item names or identifiers (e.g., product name, EAN).
timestamp Name of the column including the timestamp. This column should be in POSIX or Date-format.
temporalAggregation Temporal aggregation mode for the XYZ-analysis. Possible modes are 'day', 'week', 'month', 'quarter', 'year', and 'total'. Total only aggregates by item whereas the other modes aggregate by item an temporal unit.
fiscal consider the start of the business year. Default is set to 1 (January)
aggregationFun Function for aggregating the value column. Default is sum.

Value

Returns a data frame with the aggregated data with the columns of item, timestamp and sum, which is the sum of the value column.

Author(s)

Leon Binder <leon.binder@th-deg.de>
Bernhard Bauer <bernhard.bauer@th-deg.de>
Michael Scholz <michael.scholz@th-deg.de>
See Also

expandData

Examples

data('Amount')
aggregatedData = aggregateData(data = Amount,
   value = "value",
   item = "item",
   timestamp = "date",
   temporalAggregation = "quarter")

<table>
<thead>
<tr>
<th>Amount</th>
<th>Amount data</th>
</tr>
</thead>
</table>

Description

A dataset containing 23 items and their amounts over 3 years of data.

Usage

Amount

Format

A data frame with 10,000 rows and 9 variables:

date  Date in format yyyy-mm-dd
week  Date in format yyyy-'W'ww
month Date in format yyyy-mm
quarter Date in format yyyy-'Q'q
year  Date in format yyyy
item  Item ID
itemgroup  Item group ID
amount Item amount
value  Item value

Source

anonymized real data
**Description**

Compares two S4 objects.

**Usage**

```r
compare(object1, object2, ...)
```

**Arguments**

- `object1`  
  First S4 object.
- `object2`  
  Second S4 object.
- `...`  
  Further comparison parameters.

**Value**

Comparison result.

**Author(s)**

- Leon Binder <leon.binder@th-deg.de>
- Bernhard Bauer <bernhard.bauer@th-deg.de>
- Michael Scholz <michael.scholz@th-deg.de>

**See Also**

`compare`

**Examples**

```r
data("Amount")
data1 = Amount[sample(1:nrow(Amount), 1000),]
data2 = Amount[sample(1:nrow(Amount), 1000),]
abcxyzData1 = computeABCXYZAnalysis(data1, value = "value", item = "item", timestamp = "date",
temporalAggregation = "day", XY = 0.5, YZ = 1)
abcxyzData2 = computeABCXYZAnalysis(data2, value = "value", item = "item", timestamp = "date",
temporalAggregation = "day", XY = 0.5, YZ = 1)
comparison = compare(abcxyzData1, abcxyzData2)
```
compare, ABCXYZData, ABCXYZData-method

Compares the results of two ABC/XYZ analyses

Description

Compares the class assignments of two ABC- or two ABC/XYZ analyses.

Usage

```r
## S4 method for signature 'ABCXYZData,ABCXYZData'
compare(
  object1,
  object2,
  valueDiff = NA,
  xyzCoefficientDiff = NA,
  unequalABC = NA,
  unequalXYZ = NA
)
```

Arguments

- `object1`: Object of class `ABCXYZData`.
- `object2`: Object of class `ABCXYZData`.
- `valueDiff`: Only items with a difference of the column value larger than `valueDiff` between the first and second ABC-XYZ-Analysis are returned. In the comparison data.frame a new column is added for the difference in the value columns.
- `xyzCoefficientDiff`: Only items with a difference of the column `xyzCoefficient` larger than the `xyzCoefficientDiff` between the first and second ABC-XYZ-Analysis are returned. In the comparison data.frame a new column is added for the difference in the `xyzCoefficient` columns.
- `unequalABC`: If `TRUE` only items are returned, where the ABC-Classes are different. If `FALSE` only items are returned, where the ABC-Classes are equal. If `NA`, no further restriction takes place based on the column ABC.
- `unequalXYZ`: If `TRUE` only items are returned, where the XYZ-Classes are different. If `FALSE` only items are returned, where the XYZ-Classes are equal. If `NA`, no further restriction takes place based on the column XYZ.

Value

An `ABCXYZComparison` object.
computeABCXYZAnalysis

Author(s)
Leon Binder <leon.binder@th-deg.de>
Bernhard Bauer <bernhard.bauer@th-deg.de>
Michael Scholz <michael.scholz@th-deg.de>

See Also

ABCXYZComparison

Examples

data("Amount")
data1 = Amount[sample(1:nrow(Amount), 1000),]
data2 = Amount[sample(1:nrow(Amount), 1000),]
abcxyzData1 = computeABCXYZAnalysis(data1, value = "value", item = "item", timestamp = "date",
temporalAggregation = "day", XY = 0.5, YZ = 1)
abcxyzData2 = computeABCXYZAnalysis(data2, value = "value", item = "item", timestamp = "date",
temporalAggregation = "day", XY = 0.5, YZ = 1)
comparison = compare(abcxyzData1, abcxyzData2)

computeABCXYZAnalysis  Performs an ABC/XYZ analysis

Description

Divides a given data frame into 3 classes, A, B, C, according to the value of one column (e.g., revenue).

Usage

computeABCXYZAnalysis(
  data,
  value,
  item,
  timestamp,
  temporalAggregation = c("day", "week", "month", "quarter", "year"),
  AB = 80,
  BC = 95,
  XY = NA,
  YZ = NA,
  ignoreZeros = FALSE
)
computeABCXYZAnalysis

Arguments

data Data frame or matrix on which the ABC analysis is performed.
value Name of the column variable that contains the value for the ABCXYZ analysis.
item Names of the columns including the item names or identifiers (e.g., product name, EAN).
timestamp Name of the column including the timestamp. This column should be in POSIX or date-format.
temporalAggregation Temporal aggregation for the XYZ-analysis (i.e., "day", "week", "month", "quarter", "year").
AB Threshold (in percent) between category A and B.
BC Threshold (in percent) between category B and C.
XY Threshold (in percent) between category X and Y.
YZ Threshold (in percent) between category Y and Z.
ignoreZeros Whether zero values should be ignored in XYZ-analysis.

Value

Returns an ABCXYZData object. Only positive values are displayed

Author(s)

Leon Binder <leon.binder@th-deg.de>
Bernhard Bauer <bernhard.bauer@th-deg.de>
Michael Scholz <michael.scholz@th-deg.de>

See Also

ABCXYZData summary

Examples

# ABC Analysis
data("Amount")
abcResult = computeABCXYZAnalysis(data = Amount,
  value = "value",
  item = "item",
  timestamp = "date")

# ABC/XYZ Analysis
data("Amount")
abcxyzResult = computeABCXYZAnalysis(data = Amount,
  value = "value",
  item = "item",
  timestamp = "date",
  temporalAggregation = "week",
  XY = 0.3, YZ = 0.5)
computeConstants

Select constant items

Description

Selects items with a constant value for a specified time period.

Usage

computeConstants(
  data, value, group, timestamp,
  timestampFormat = c("day", "week", "month", "quater", "year"),
  currentTime, thresholdTime = 7,
  use_latest = FALSE
)

Arguments

data      Dataframe containing item stock data.
value     Name of the column variable containing the stock values.
group     Name(s) of the column(s) that are used to group stock data. These columns are usually the item ID or item name to group stock data by items.
timestamp Name of the column including the timestamp. This column should be in Date, POSIX, YY-mm, YYYY-`W`ww, YYYY-mm, YYYY-`Q`q or YYYY format.
timestampFormat Declares in which format the timestamp comes in (i.e., "day", "week", "month", "quarter", "year")
currentTime Qualifying date for the value variable. Date must exist in data and have the same format as timestamp-variable.
thresholdTime Time for which the value shouldn’t exceed the threshold value. Number declares the time in the format of timestampFormat.
use_latest  boolean value. If TRUE data will expand and dates with no existing values will be filled up with the latest known values.

Value

Returns a data frame listing all constant items, the date since when the stock is constant and the value of the stock since this time.
Author(s)

Leon Binder <leon.binder@th-deg.de>
Bernhard Bauer <bernhard.bauer@th-deg.de>
Michael Scholz <michael.scholz@th-deg.de>

See Also

computeUnderperformer computeOverperformer

Examples

data("Stocks")
constants = computeConstants(data=Stocks,
   value = "stock",
   group = "item",
   timestamp = "date",
   timestampFormat = "day",
   currentTime = "2019-07-27",
   thresholdTime = 7,
   use_latest = FALSE)

computeOverperformer Select overperforming items

Description

Selects items with a value higher than a given threshold for a specified time period.

Usage

computeOverperformer(
   data,
   value,
   group,
   timestamp,
   timestampFormat = c("day", "week", "month", "quater", "year"),
   currentTime,
   thresholdValue = 0,
   thresholdTime = 90,
   use_latest = FALSE
)
computeOverperformer

Arguments

- **data**
  - Dataframe containing item stock data.
- **value**
  - Name of the column variable containing the stock values.
- **group**
  - Name(s) of the column(s) that are used to group stock data. These columns are usually the item ID or item name to group stock data by items.
- **timestamp**
  - Name of the column including the timestamp. This column should be in Date, POSIX, YY-mm, YYYY-’W’ww, YYYY-mm, YYYY-’Q’q or YYYY format.
- **timestampFormat**
  - Declares in which format the timestamp comes in (i.e., "day", "week", "month", "quarter", "year")
- **currentTime**
  - Qualifying date for the value variable. Date must exist in data and have the same format as timestamp-variable.
- **thresholdValue**
  - Name of the column variable containing the items’ stock threshold value or the threshold value used in this analysis for all items.
- **thresholdTime**
  - Time for which the value shouldn’t exceed the threshold value. Number declares the time in the format of timestampFormat.
- **use_latest**
  - boolean value. If TRUE data will expand and dates with noexisting values will be filled up with the latest known values.

Value

Returns a data frame listing all overperforming items, the date their stock was the last time under the threshold (lastunder), the duration in days since the stock is over the threshold (toolowindays), the average difference between the stock and the threshold (meandiff) and the count of switched between over- and underperformance (moves).

Author(s)

Leon Binder <leon.binder@th-deg.de>
Bernhard Bauer <bernhard.bauer@th-deg.de>
Michael Scholz <michael.scholz@th-deg.de>

See Also

computeUnderperformer computeConstants

Examples

```r
data("Stocks")
overperformer = computeOverperformer(data = Stocks,
  value = "stock",
  group = "item",
  timestamp = "date",
  timestampFormat = "day",
  currentTime = "2019-07-27",
  thresholdValue = "reorderlevel",
  thresholdTime = 0,
  use_latest = FALSE)
```
computeUnderperformer  *Select underperforming items*

**Description**

Selects items with a value lower than a given threshold for a specified time period.

**Usage**

```r
computeUnderperformer(
  data, value, group, timestamp,
  timestampFormat = c("day", "week", "month", "quater", "year"),
  currentTime, thresholdValue = 0,
  thresholdTime = 90,
  use_latest = FALSE
)
```

**Arguments**

- **data**  
  Dataframe containing item stock data.
- **value**  
  Name of the column variable containing the stock values.
- **group**  
  Name(s) of the column(s) that are used to group stock data. These columns are usually the item ID or item name to group stock data by items.
- **timestamp**  
  Name of the column including the timestamp. This column should be in Date, POSIX, YY-mm, YYYY-'W'ww, YYYY-mm, YYYY-'Q'q or YYYY format.
- **timestampFormat**  
  Declares in which format the timestamp comes in (i.e., "day", "week", "month", "quarter", "year")
- **currentTime**  
  Qualifying date for the value variable. Date must exist in data and have the same format as timestamp-variable.
- **thresholdValue**  
  Name of the column variable containing the items’ stock threshold value or the threshold value used in this analysis for all items.
- **thresholdTime**  
  Time for which the value shouldn’t exceed the threshold value. Number declares the time in the format of timestampFormat
- **use_latest**  
  boolean value. If TRUE data will expand and dates with no existing values will be filled up with the latest known values

**Value**

Returns a data frame listing all underperforming items, the date their stock was the last time over the threshold (lastover), the duration in days since the stock is under the threshold (tooldownays), the average difference between the stock and the threshold (meandiff) and the count of switched between over- and underperformance (moves).
**detectTimeVariations**

**Author(s)**
Leon Binder <leon.binder@th-deg.de>
Bernhard Bauer <bernhard.bauer@th-deg.de>
Michael Scholz <michael.scholz@th-deg.de>

**See Also**
computeOverperformer computeConstants

**Examples**

```r
data("Stocks")
underperformer = computeUnderperformer(data=Stocks,
  value = "stock",
  group = "item",
  timestamp = "date",
  timestampFormat = "day",
  currentTime = "2019-07-27",
  thresholdValue = "minstock",
  thresholdTime = 90,
  use_latest = FALSE)
```

---

detectTimeVariations  *Detects items whose value (stock, demand, etc.) has changed over time*

**Description**

Detects items whose value (stock, demand, etc.) has changed over time in contrast to other items. This analysis is based on the Macnaughton-Smith et al. clustering algorithm.

**Usage**

```r
detectTimeVariations(
  data,
  value,
  item,
  timestamp,
  temporalAggregation = c("day", "week", "month", "quarter", "year"),
  aggregationFun = sum,
  preProcess = NA,
  recentTimePeriods = 5
)
```
**Arguments**

- **data**
  Data frame that will be expanded.

- **value**
  Name of the column variable that contains the value for the ABC and XYZ analysis.

- **item**
  Name of the column including the item names or identifiers (e.g., product name, EAN)

- **timestamp**
  Name of the column including the timestamp. This column should be in POSIX or Date-format.

- **temporalAggregation**
  Temporal aggregation mode (i.e., "day", "week", "month", "quarter", "year").

- **aggregationFun**
  Function for aggregating the value column. Default is sum.

- **preProcess**
  A string vector that defines a pre-processing of the aggregated data before clustering. Available pre-processing methods are "center", "scale", "standardize", and "normalize". Default is NA (no pre-processing).

- **recentTimePeriods**
  Integer indicating the number of time periods that are used to define the recent item values. Default is 5.

**Value**

Returns a data frame showing to which cluster each item belongs based on all value and based on the recent values as well as whether the item has switched the cluster.

**References**


**Examples**

```r
data("Amount")
timeVariations = detectTimeVariations(data = Amount, 
  value = "amount", 
  item = "item", 
  timestamp = "date", 
  temporalAggregation = "week")
```

**Description**

Expands a temporal data frame and fills values for missing dates.
Usage

```r
expandData(
  data, 
  expand, 
  expandTo = c("all", "event"), 
  valueColumns, 
  latest_values = F, 
  valueLevels = NA, 
  timestamp, 
  timestampFormat = c("day", "week", "month", "quarter", "year"), 
  keepData = T 
)
```

Arguments

data: Data frame that will be expanded.
expand: Name of the variables that will be expanded.
expandTo: Defines whether values for the variables to be expanded will be filled for all dates or only those dates included in the data.
valueColumns: Name of the columns that are filled with specific values.
latest_values: If `True` missing values are filled with the latest known value until the next known value comes in.
valueLevels: Specific values that are used to fill the value columns. If `latest_values = TRUE` only values with no known values in the past of this values are specified with this specific values.
timestamp: Name of the column including the timestamp. This column should be in `Date`, `YY-mm`, `YYYY-'W'ww`, `YYYY-mm`, `YYYY-'Q'q` or `YYYY` format.
timestampFormat: Declares in which format the timestamp comes in (i.e., "day", "week", "month", "quarter", "year").
keepData: Defines whether variables that will not be expanded should be kept.

Value

Returns the expanded data frame.

Author(s)

Leon Binder <leon.binder@th-deg.de>
Bernhard Bauer <bernhard.bauer@th-deg.de>
Michael Scholz <michael.scholz@th-deg.de>

See Also

aggregateData
Examples

data("Amount")
expandedItems = expandData(Amount,
    expand = c("item", "itemgroup"),
    expandTo = "all",
    valueColumns = c("amount", "value"),
    latest_values = TRUE,
    valueLevels = c(0, 0),
    timestamp = "date",
    timestampFormat = "day")

---

Forecast-class Class Forecast

Description

This S4 class represents the result of forecast using function predictValue.

Slots

data (data.frame) Data frame including the predicted data and optionally the training data.
models (list) List of fitted ARIMA models.
value (character) Name of the value column.
item (character) Name of the item column.
items (character) IDs or Names of the items.

Objects from the Class

Objects can be created by calling the function predictValue. This S4 class represents the result of a forecast.

Author(s)

Leon Binder <leon.binder@th-deg.de>
Bernhard Bauer <bernhard.bauer@th-deg.de>
Michael Scholz <michael.scholz@th-deg.de>

Examples

data("Amount")
prediction = predictValue(data = Amount,
    value = "amount",
    item = "item",
    timestamp = "date",
    temporalAggregation = "week",
    timeUnitsAhead = 3)
prediction
**matmanDemo**

*Launches a demo app*

**Description**

Launches a shiny app that demonstrates how to use the functions provided by package **matman**.

**Usage**

matmanDemo()

**Author(s)**

Leon Binder <leon.binder@th-deg.de>
Bernhard Bauer <bernhard.bauer@th-deg.de>
Michael Scholz <michael.scholz@th-deg.de>

**Examples**

```r
## Not run: matmanDemo()
```

---

**plot,ABCXYZData,ANY-method**

*Plots the result of an ABC/XYZ analysis*

**Description**

Plots a graph that shows what percentage of items is responsible for what amount of value.

**Usage**

```r
## S4 method for signature 'ABCXYZData,ANY'
plot(
  x,
  plot_engine = c("graphics", "plotly"),
  title = "",
  xlab = "",
  ylab = "",
  top5lab = NA,
  color = list(itemColor = "blue", top5Color = "black", aColor = "green", bColor = "orange", cColor = "red"),
  item = NA,
  ...
)
```
Arguments

- **x**: Object of class `ABCXYZData`.
- **plot_engine**: Name of the plot engine ("graphics", "plotly").
- **title**: Plot title (e.g., 'ABC-Analysis').
- **xlab**: Label of x-axis (e.g., 'Percentage of Items').
- **ylab**: Label of y-axis (e.g., 'Percentage of cumulative Value').
- **top5lab**: Title of the rank of the top 5 items (e.g., 'Items with the highest Value').
- **color**: List of plot colors (i.e., itemColor, top5Color, aColor, bColor, cColor). Default is `list(itemColor = "blue", top5Color = "black", aColor = "green", bColor = "orange", cColor = "red")`.
- **item**: Name of a single column with an identifier, that is displayed in the top-5-ranking. Used if the `ABCXYZData` object has multiple item columns. If `NA` the first item column is displayed.
- **...**: Further optional parameters for function `graphics::plot` or function `plotly::plot_ly`.

Author(s)

Leon Binder <leon.binder@th-deg.de>
Bernhard Bauer <bernhard.bauer@th-deg.de>
Michael Scholz <michael.scholz@th-deg.de>

See Also

`computeABCXYZAnalysis` `ABCXYZData`

Examples

```r
data("Amount")
abcResult = computeABCXYZAnalysis(data = Amount,
     value = "value",
     item = "item",
     timestamp = "date")
plot(abcResult,
     plot_engine = "graphics",
     title = "ABC Analysis",
     xlab = "Items",
     ylab = "Demand")
```
**plotValueSeries**  
*Plots the development of the values*

**Description**

Plots a bar chart that shows the sum of the value column for a certain time interval.

**Usage**

```r
plotValueSeries(  
  data,  
  item,  
  item_id,  
  value,  
  timestamp,  
  temporalAggregation = c("day", "week", "month", "quarter", "year"),  
  expand = TRUE,  
  withTrendLine = TRUE,  
  windowLength = 5,  
  trendLineType = "s"
)
```

**Arguments**

- **data**
  Data frame or matrix on which the ABC analysis is performed.

- **item**
  Name of the column including the item name or identifier (e.g., product name, EAN).

- **item_id**
  Name of the item that will be displayed.

- **value**
  Name of the column variable that contains the values.

- **timestamp**
  Name of the column including the timestamp. This column should be in POSIX or date-format.

- **temporalAggregation**
  Temporal aggregation for the XYZ-analysis (i.e., "day", "week", "month", "quarter", "year").

- **expand**
  Indicator if the data should be expanded with time intervals that have no data.

- **withTrendLine**
  Indicator if a trend line should be displayed in the bar chart.

- **windowLength**
  Backwards window length.

- **trendLineType**
  If "s" the simple and if "w" the weighted moving average is calculated.

**Value**

A plotly bar chart, that shows the development of the value column.
predictValue

Author(s)
Leon Binder <leon.binder@th-deg.de>
Bernhard Bauer <bernhard.bauer@th-deg.de>
Michael Scholz <michael.scholz@th-deg.de>

Examples

```
data("Amount")
plotValueSeries(Amount,
    item = "item",
    item_id = "45186",
    value = "amount",
    timestamp = "date",
    temporalAggregation = "week",
    withTrendLine = TRUE,
    windowLength = 10,
    trendLineType = "w")
```

predictValue | Predicts the value for items |

Description

Predicts the value for items based on previous values. Previous values can be aggregated to value per day, week, month, quarter or year. An ARIMA model is estimated for each item based on the function forecast: auto.arima. The best model is selected and used for prediction. Note that only models without drift term will be considered in order to ensure consistent predictions.

Usage

```
predictValue(
    data,
    value,
    item,
    timestamp,
    temporalAggregation = c("day", "week", "month", "quarter", "year"),
    aggregationFun = sum,
    timeUnitsAhead = 1,
    digits = 3,
    expand = F,
    keepPreviousData = F,
    level = 0.95,
    ...
)
```
predictValue

Arguments

data  Data frame including previous values.
value Name of the column representing the item value.
item  Name of the column representing the item ID or the item name.
timestamp Name of the column including the timestamp. This column should be in POSIX or date-format.
temporalAggregation Temporal aggregation mode (i.e., "day", "week", "month", "quarter", "year").
aggregationFun Function for aggregating the value column. Default is sum.
timeUnitsAhead Integer indicating the number of time units (i.e., days, weeks, months, quarters or years) the should be predicted.
digits Integer indicating the number of significant digits used for the predicted values.
expand Logical indicating whether the data will be expanded after they are aggregated. Default is not (FALSE).
keepPreviousData Logical indicating whether the data from the given data frame will be added to the result or not. Default is not (FALSE).
level Numeric value representing the confidence level for the predictions. The default is 0.95 (i.e. lower level = 0.025 and upper level = 0.975).
... Further arguments for function forecast::auto.arima.

Value

Returns a Forecast object.

Author(s)

Leon Binder <leon.binder@th-deg.de>
Bernhard Bauer <bernhard.bauer@th-deg.de>
Michael Scholz <michael.scholz@th-deg.de>

See Also

auto.arima Forecast

Examples

# Simple Example
data("Amount")
prediction = predictValue(data = Amount,
  value = "amount",
  item = "item",
  timestamp = "date",
  temporalAggregation = "week",
  timeUnitsAhead = 3)
prediction
# More Sophisticated Example

data("Amount")
prediction = predictValue(data = Amount,
  value = "amount",
  item = "item",
  timestamp = "date",
  temporalAggregation = "week",
  aggregationFun = mean,
  timeUnitsAhead = 5,
  digits = 4,
  keepPreviousData = TRUE,
  level = 0.9,
  trace = TRUE)
prediction

---

show,ABCXYZComparison-method

*Shows an ABCXYZComparison object*

**Description**

Shows an ABCXYZComparison object as a table consisting of the absolute and relative amount of each item, the cumulative relative amount and the ABC-class for both ABCXYZData objects. It furthermore shows the ABC comparison of the two objects. If XY and YZ parameters have been specified for computing the ABCXYZData object, the table also includes a column for the XYZ coefficient, the XYZ-class, the ABC/XYZ-class and the XYZ comparison.

**Usage**

```r
## S4 method for signature 'ABCXYZComparison'
show(object)
```

**Arguments**

- `object` The ABCXYZComparison object

**Author(s)**

Leon Binder <leon.binder@th-deg.de>

Bernhard Bauer <bernhard.bauer@th-deg.de>

Michael Scholz <michael.scholz@th-deg.de>

**See Also**

- ABCXYZComparison
- compare
Examples

data("Amount")
data1 = Amount[sample(1:nrow(Amount), 1000),]
data2 = Amount[sample(1:nrow(Amount), 1000),]
abcxyzData1 = computeABCXYZAnalysis(data1, value = "value", item = "item", timestamp = "date",
temporalAggregation = "day", XY = 0.5, YZ = 1)
abcxyzData2 = computeABCXYZAnalysis(data2, value = "value", item = "item", timestamp = "date",
temporalAggregation = "day", XY = 0.5, YZ = 1)
comparison = compare(abcxyzData1, abcxyzData2)
comparison

show,ABCXYZData-method

Shows an ABCXYZData object

Description

Shows the ABCXYZData object as a table consisting of the absolute and relative amount of each item, the cumulative relative amount and the ABC-class. If XY and YZ parameters have been specified for computing the ABCXYZData object, the table also includes a column for the XYZ coefficient, the XYZ-class and the ABC/XYZ-class.

Usage

## S4 method for signature 'ABCXYZData'
show(object)

Arguments

object The ABCXYZData object

Author(s)

Leon Binder <leon.binder@th-deg.de>
Bernhard Bauer <bernhard.bauer@th-deg.de>
Michael Scholz <michael.scholz@th-deg.de>

See Also

ABCXYZData computeABCXYZAnalysis

Examples

data("Amount")
abcResult = computeABCXYZAnalysis(data = Amount,
  value = "value",
  item = "item",
  timestamp = "date")
abcResult
show.Forecast-method  Shows a Forecast object

Description

Shows the predicted data of a Forecast object. If the Forecast object was created using keepPreviousData = TRUE, also the training data are shown.

Usage

```r
## S4 method for signature 'Forecast'
show(object)
```

Arguments

- `object` The Forecast object

Author(s)

Leon Binder <leon.binder@th-deg.de>
Bernhard Bauer <bernhard.bauer@th-deg.de>
Michael Scholz <michael.scholz@th-deg.de>

See Also

Forecast

Examples

```r
data("Amount")
prediction = predictValue(data = Amount,
value = "amount",
item = "item",
timestamp = "date",
temporalAggregation = "week",
timeUnitsAhead = 3)
prediction
```
**Description**

A dataset containing 10 items and their stocks over 3 years of data.

**Usage**

Stocks

**Format**

A data frame with 1,610 rows and 5 variables:

- **date**: Date in format yyyy-mm-dd
- **item**: Item ID
- **stock**: Item stock value
- **minstock**: Minimum stock per item
- **reorderlevel**: Stock threshold for triggering item reorders

**Source**

anonymized real data

---

**summary**

Summarizes an S4 object

**Description**

Summarizes an S4 object.

**Usage**

`summary(object, ...)`

**Arguments**

- **object**: S4 object.
- **...**: Optional parameters.

**Value**

Summary.
summary,ABCXYZComparison-method

Prints the summary of the comparison of two ABC/XYZ analyses

Description

Summarizes the differences between two ABCXYZData objects.

Usage

## S4 method for signature 'ABCXYZComparison'
summary(object, withMissing = FALSE)

Arguments

object Object of class ABCXYZComparison.
withMissing Logical indicating whether missing categories will be shown. Default is FALSE.

Value

A contingency table showing the differences.

Author(s)

Leon Binder <leon.binder@th-deg.de>
Bernhard Bauer <bernhard.bauer@th-deg.de>
Michael Scholz <michael.scholz@th-deg.de>
summary.ABCXYZData-method

See Also

compare ABCXYZComparison

Examples

data("Amount")
data1 = Amount[sample(1:nrow(Amount), 1000),]
data2 = Amount[sample(1:nrow(Amount), 1000),]
abcxyzData1 = computeABCXYZAnalysis(data1, value = "value", item = "item", timestamp = "date",
temporalAggregation = "day", XY = 0.5, YZ = 1)
abcxyzData2 = computeABCXYZAnalysis(data2, value = "value", item = "item", timestamp = "date",
temporalAggregation = "day", XY = 0.5, YZ = 1)
comparison = compare(abcxyzData1, abcxyzData2)
summary(comparison)

summary.ABCXYZData-method

Prints the result summary of an ABC/XYZ analysis

Description

Summarizes the items count and value sum grouped by the different ABC- or ABC/XYZ-Classes.

Usage

## S4 method for signature 'ABCXYZData'
summary(object, withMissing = FALSE)

Arguments

object 
Object of class ABCXYZData.

withMissing 
Logical indicating whether missing categories will be shown. Default is FALSE.

Value

A data.table with the summarized results.

Author(s)

Leon Binder <leon.binder@th-deg.de>
Bernhard Bauer <bernhard.bauer@th-deg.de>
Michael Scholz <michael.scholz@th-deg.de>

See Also

computeABCXYZAnalysis ABCXYZData
# ABC Analysis

```r
data("Amount")
abcResult = computeABCXYZAnalysis(data = Amount,
    value = "value",
    item = "item",
    timestamp = "date")
summary(abcResult)
```

# ABC/XYZ Analysis

```r
data("Amount")
abcxyzResult = computeABCXYZAnalysis(data = Amount,
    value = "value",
    item = "item",
    timestamp = "date",
    temporalAggregation = "week",
    XY = 0.3, YZ = 0.5)
summary(abcxyzResult)
```

---

### summary,Forecast-method

> Prints the summary of a Forecast object

---

**Description**

Summarizes the fitted models estimated for predicting item values (e.g., demand, stock).

**Usage**

```r
## S4 method for signature 'Forecast'
summary(object)
```

**Arguments**

- `object` Object of class Forecast

**Value**

A data frame showing a summary of fitted models.

**Author(s)**

- Leon Binder <leon.binder@th-deg.de>
- Bernhard Bauer <bernhard.bauer@th-deg.de>
- Michael Scholz <michael.schoz@th-deg.de>

**See Also**

- `predictValue Forecast`
Examples

```r
data("Amount")
prediction = predictValue(data = Amount,
  value = "amount",
  item = "item",
  timestamp = "date",
  temporalAggregation = "week",
  timeUnitsAhead = 3)
summary(prediction)
```
Index

* ABC-Analysis
  matman-package, 2
* XYZ-Analysis
  matman-package, 2
* classes
  ABCXYZComparison-class, 3
  ABCXYZData-class, 4
  Forecast-class, 18
* datasets
  Amount, 6
  Stocks, 27
* methods
  compare, 7
  compare, ABCXYZData, ABCXYZData-method, 8
  plot, ABCXYZData, ANY-method, 19
  plotValueSeries, 21
  show, ABCXYZComparison-method, 24
  show, ABCXYZData-method, 25
  show, Forecast-method, 26
  summary, 27
  summary, ABCXYZComparison-method, 28
  summary, ABCXYZData-method, 29
  summary, Forecast-method, 30

ABCXYZComparison, 9, 24, 29
ABCXYZComparison-class, 3
ABCXYZData, 10, 20, 25, 29
ABCXYZData-class, 4
aggregateData, 5, 17
Amount, 6
auto.arima, 23

compare, 7, 7, 24, 29
compare, ABCXYZData, ABCXYZData-method, 8
compare, ABCXYZData-method
  (compare, ABCXYZData, ABCXYZData-method), 8

computeABCXYZAnalysis, 9, 20, 25, 29
computeConstants, 11, 13, 15
computeOverperformer, 12, 12, 15
computeUnderperformer, 12, 13, 14
detectTimeVariations, 15
expandData, 6, 16
Forecast, 23, 26, 30
Forecast-class, 18
matman (matman-package), 2
matman-package, 2
matmanDemo, 19
plot, ABCXYZData, ANY-method, 19
plotValueSeries, 21
predictValue, 22, 30
show, ABCXYZComparison-method, 24
show, ABCXYZData-method, 25
show, Forecast-method, 26
Stocks, 27
summary, 10, 27, 28
summary, ABCXYZComparison-method, 28
summary, ABCXYZData-method, 29
summary, Forecast-method, 30