Package ‘tidybins’

June 12, 2024

Type Package

Title Make Tidy Bins

Version 0.1.1

Maintainer Harrison Tietze <Harrison4192@gmail.com>

Description Multiple ways to bin numeric columns with a tidy output. Wraps a variety of existing binning methods into one function, and includes a new method for binning by equal value, which is useful for sales data. Provides a function to automatically summarize the properties of the binned columns.

Encoding UTF-8

RoxygenNote 7.3.1


BugReports https://github.com/Harrison4192/tidybins/issues

Imports magrittr, dplyr, stringr, tidyselect, purrr, janitor, tibble, rlang, lubridate, stats, scales, ggplot2, rlist, OneR, strex, ClusterR, framecleaner, xgboost, badger, autostats

Suggests knitr, rmarkdown, arulesCBA, embed, woeBinning, recipes

VignetteBuilder knitr

License GPL (>= 3)

NeedsCompilation no

Author Harrison Tietze [aut, cre]

Repository CRAN

Date/Publication 2024-06-12 04:50:02 UTC

Contents

 add_clusters ......................................................... 2
 bin_cols ............................................................ 3
 bin_equal_value .................................................... 4
add_clusters

Description

Wraps `KMeans_rcpp` to create a column that is a cluster formed from select columns in the data frame. Clusters names are specified by capital letters.

Usage

```
add_clusters(.data, ..., n_clusters = 4, cluster_name = "cluster")
```

Arguments

- `.data` dataframe
- `...` columns to cluster (tidyselect)
- `n_clusters` integer
- `cluster_name` column name

Value

data frame

Examples

```
iris %>%
tibble::as_tibble() %>%
add_clusters(Sepal.Width, Sepal.Length, n_clusters = 3, cluster_name = "Sepal_Cluster") -> iris1

iris1

iris1 %>%
numeric_summary(original_col = Sepal.Width, bucket_col = Sepal_Cluster)
```
**Description**

Make bins in a tidy fashion. Adds a column to your data frame containing the integer codes of the specified bins of a certain column. Specifying multiple columns is only intended for supervised binning, so multiple columns can be simultaneously binned optimally with respect to a target variable.

**Usage**

```r
bin_cols(
  .data, 
  col, 
  n_bins = 10, 
  bin_type = "frequency", 
  ..., 
  target = NULL, 
  pretty_labels = FALSE, 
  seed = 1, 
  method = "mdlp"
)
```

**Arguments**

- `.data` a data frame
- `col` a column, vector of columns, or tidyselect
- `n_bins` number of bins
- `bin_type` method to make bins
- `...` params to be passed to selected binning method
- `target` unquoted column for supervised binning
- `pretty_labels` logical. If T returns interval label rather than integer rank
- `seed` seed for stochastic binning (xgboost)
- `method` method for bin mdlp

**Details**

Description of the arguments for `bin_type`

- **frequency (fr)** creates bins of equal content via quantiles. Wraps `bin` with method "content". Similar to `ntile`
- **width (wi)** create bins of equal numeric width. Wraps `bin` with method "length"
- **kmeans (km)** create bins using 1-dimensional kmeans. Wraps `bin` with method "clusters"
**bin_equal_value**

- **value (va)**: each bin has equal sum of values
- **xgboost (xg)**: column is binned by best predictor of a target column using `step_discretize_xgb`
- **cart (ca)**: if the col does not have enough distinct values, xgboost will fail and automatically revert to `step_discretize_cart`
- **woe (wo)**: column is binned by weight of evidence. Requires binary target
- **logreg (lr)**: column is binned by logistic regression. Requires binary target.
- **mdlp** uses the `discretizeDF.supervised` algorithm with a variety of methods.

**Value**

a data frame

**Examples**

```r
iris %>%
  bin_cols(Sepal.Width, n_bins = 5, pretty_labels = TRUE) %>%
  bin_cols(Petal.Width, n_bins = 3, bin_type = c("width", "kmeans")) %>%
  bin_cols(Sepal.Width, bin_type = "xgboost", target = Species, seed = 1) -> iris1

#binned columns are named by original name + method abbreviation + number bins created.
#Sometimes the actual number of bins is less than n_bins if the col lacks enough variance.
iris1 %>%
  print(width = Inf)
iris1 %>%
  bin_summary() %>%
  print(width = Inf)
```

**Description**

Bins a numeric column such that each bin contains 10. Intended for positive numeric vectors that make sense to sum, such as sales. Negative and NAs get treated as 0. The function never puts two rows with the same value into different bins. Accessed by the "value" method of the `bin_cols` function.

**Usage**

`bin_equal_value(mdb, col, n_bins = 10)`

**Arguments**

- **mdb**: dataframe
- **col**: a numeric vector
- **n_bins**: number of bins
### bin_summary

**Value**

an integer vector

---

**Description**

Returns a summary of all bins created by `bin_cols` in a data frame. Takes no arguments other than the data frame but relies on regular expressions based of the `bin_cols` output in order to identify the corresponding columns.

**Usage**

```r
bin_summary(mdb, ...)
```

**Arguments**

- `mdb`: dataframe output from `bin_cols`
- `...`: optional tidyselect specification for specific cols

**Value**

a tibble

**Examples**

```r
iris %>%
  bin_cols(Sepal.Width) %>%
  bin_summary()
```

---

### drop_original_cols

**Drop Original Cols**

**Description**

Drops the original column from the dataframe once bins are made. Throws an error if the same column has multiple bin cols.

**Usage**

```r
drop_original_cols(.data, ..., restore_names = FALSE)
```
Arguments

.data dataframe output from bin_cols
... tidyselect. default chooses all cols created from binning
.restore_names Logical, default FALSE. rename the binned cols with the original column names?

Value
dataframe

Examples

iris %>%
  bin_cols(Sepal.Length) %>%
  bin_cols(Sepal.Width, pretty_labels = TRUE) -> iris1

iris1

iris1 %>%
  drop_original_cols(restore_names = TRUE)

iris1 %>%
  drop_original_cols(restore_names = FALSE)

Description

The five number summary of a numeric vector you would get from `summary` but returned with a tidy output.

Usage

five_number_summary(x)

Arguments

x a numeric vector

Value

a tibble

Examples

iris$Petal.Width %>%
five_number_summary()
Description

This function summarizes an arbitrary bin column, with respect to its original column. Can be used to summarize bins created from any package, or any arbitrary categorical column paired with a numeric column.

Usage

`numeric_summary(mdb, original_col, bucket_col)`

Arguments

- `mdb` a data frame
- `original_col` original numeric column
- `bucket_col` columns of bins

Value

a tibble

Examples

```
iris %>%
numeric_summary(original_col = Sepal.Length, bucket_col = Species)
```
Index

add_clusters, 2

bin, 3
bin_cols, 3
bin_equal_value, 4
bin_summary, 5

discretizeDF.supervised, 4
drop_original_cols, 5

tfive_number_summary, 6
KMeans_rcpp, 2

ntile, 3
numeric_summary, 7

step_discretize_cart, 4
step_discretize_xgb, 4