
toad

Release 0.0.59

Feb 07, 2020

Contents

1	Installation	1
2	Tutorial	3
3	Contents	5
3.1	toad package	5
3.2	Submodules	5
3.3	Module contents	22
4	Indices and tables	23
	Python Module Index	25
	Index	27

CHAPTER 1

Installation

via pip

```
pip install toad
```

via source code

```
python setup.py install
```


CHAPTER 2

Tutorial

A [basic tutorial](#) is provided.

3.1 toad package

3.2 Submodules

3.2.1 toad.detector module

Command line tools for detecting csv data

Team: ESC

Examples

```
python detector.py -i xxx.csv -o report.csv
```

```
toad.detector.getTopValues (series, top=5, reverse=False)
```

Get top/bottom n values

Parameters

- **series** (*Series*) – data series
- **top** (*number*) – number of top/bottom n values
- **reverse** (*bool*) – it will return bottom n values if True is given

Returns Series of top/bottom n values and percentage. ['value:percent', None]

Return type Series

```
toad.detector.getDescribe (series, percentiles=[0.25, 0.5, 0.75])
```

Get describe of series

Parameters

- **series** (*Series*) – data series

- **percentiles** – the percentiles to include in the output

Returns the describe of data include mean, std, min, max and percentiles

Return type Series

`toad.detector.countBlank(series, blanks=[None])`

Count number and percentage of blank values in series

Parameters

- **series** (*Series*) – data series
- **blanks** (*list*) – list of blank values

Returns number of blanks str: the percentage of blank values

Return type number

`toad.detector.isNumeric(series)`

Check if the series's type is numeric

Parameters **series** (*Series*) – data series

Returns bool

`toad.detector.detect(dataframe)`

Detect data

Parameters **dataframe** (*DataFrame*) – data that will be detected

Returns report of detecting

Return type DataFrame

3.2.2 toad.merge module

`toad.merge.ChiMerge()`

Chi-Merge

Parameters

- **feature** (*array-like*) – feature to be merged
- **target** (*array-like*) – a array of target classes
- **n_bins** (*int*) – n bins will be merged into
- **min_samples** (*number*) – min sample in each group, if float, it will be the percentage of samples
- **min_threshold** (*number*) – min threshold of chi-square

Returns array of split points

Return type array

`toad.merge.DTMerge()`

Merge continue

Parameters

- **feature** (*array-like*) –
- **target** (*array-like*) – target will be used to fit decision tree
- **nan** (*number*) – value will be used to fill nan

- **n_bins** (*int*) – n groups that will be merged into
- **min_samples** (*int*) – min number of samples in each leaf nodes

Returns array of split points

Return type array

`toad.merge.KMeansMerge()`
Merge by KMeans

Parameters

- **feature** (*array-like*) –
- **target** (*array-like*) – target will be used to fit kmeans model
- **nan** (*number*) – value will be used to fill nan
- **n_bins** (*int*) – n groups that will be merged into
- **random_state** (*int*) – random state will be used for kmeans model

Returns split points of feature

Return type array

`toad.merge.QuantileMerge()`
Merge by quantile

Parameters

- **feature** (*array-like*) –
- **nan** (*number*) – value will be used to fill nan
- **n_bins** (*int*) – n groups that will be merged into
- **q** (*array-like*) – list of percentage split points

Returns split points of feature

Return type array

`toad.merge.StepMerge()`
Merge by step

Parameters

- **feature** (*array-like*) –
- **nan** (*number*) – value will be used to fill nan
- **n_bins** (*int*) – n groups that will be merged into
- **clip_v** (*number | tuple*) – min/max value of clipping
- **clip_std** (*number | tuple*) – min/max std of clipping
- **clip_q** (*number | tuple*) – min/max quantile of clipping

Returns split points of feature

Return type array

`toad.merge.merge`
merge feature into groups

Parameters

- **feature** (*array-like*) –
- **target** (*array-like*) –
- **method** (*str*) – ‘dt’, ‘chi’, ‘quantile’, ‘step’, ‘kmeans’ - the strategy to be used to merge feature
- **return_splits** (*bool*) – if needs to return splits
- **n_bins** (*int*) – n groups that will be merged into

Returns a array of merged label with the same size of feature array: list of split points

Return type array

3.2.3 toad.metrics module

`toad.metrics.KS(score, target)`
calculate ks value

Parameters

- **score** (*array-like*) – list of score or probability that the model predict
- **target** (*array-like*) – list of real target

Returns the max KS value

Return type float

`toad.metrics.KS_bucket(score, target, bucket=10, method='quantile', **kwargs)`
calculate ks value by bucket

Parameters

- **score** (*array-like*) – list of score or probability that the model predict
- **target** (*array-like*) – list of real target
- **bucket** (*int*) – n groups that will bin into
- **method** (*str*) – method to bin score. *quantile* (default), *step*

Returns DataFrame

`toad.metrics.KS_by_col(df, by='feature', score='score', target='target')`

`toad.metrics.SSE(y_pred, y)`
sum of squares due to error

`toad.metrics.MSE(y_pred, y)`
mean of squares due to error

`toad.metrics.AIC(y_pred, y, k, llf=None)`
Akaike Information Criterion

Parameters

- **y_pred** (*array-like*) –
- **y** (*array-like*) –
- **k** (*int*) – number of featuers
- **llf** (*float*) – result of log-likelihood function

`toad.metrics.BIC(y_pred, y, k, llf=None)`
Bayesian Information Criterion

Parameters

- **y_pred** (*array-like*) –
- **y** (*array-like*) –
- **k** (*int*) – number of features
- **llf** (*float*) – result of log-likelihood function

`toad.metrics.F1(score, target, split='best', return_split=False)`
calculate f1 value

Parameters

- **score** (*array-like*) –
- **target** (*array-like*) –

Returns best f1 score float: best splitter

Return type float

`toad.metrics.AUC(score, target)`
AUC Score

Parameters

- **score** (*array-like*) – list of score or probability that the model predict
- **target** (*array-like*) – list of real target

Returns auc score

Return type float

`toad.metrics.PSI(test, base, combiner=None, return_frame=False)`
calculate PSI

Parameters

- **test** (*array-like*) – data to test PSI
- **base** (*array-like*) – base data for calculate PSI
- **combiner** (*Combiner/list/dict*) – combiner to combine data
- **return_frame** (*bool*) – if need to return frame of proportion

Returns float|Series

`toad.metrics.matrix(y_pred, y, splits=None)`
confusion matrix of target

Parameters

- **y_pred** (*array-like*) –
- **y** (*array-like*) –
- **splits** (*float/list*) – split points of y_pred

Returns confusion matrix with true labels in rows and predicted labels in columns

Return type DataFrame

3.2.4 toad.plot module

`toad.plot.badrates_plot` (*frame*, *x=None*, *target='target'*, *by=None*, *freq=None*, *format=None*, *return_counts=False*, *return_proportion=False*, *return_frame=False*)

plot for badrate

Parameters

- **frame** (*DataFrame*) –
- **x** (*str*) – column in frame that will be used as x axis
- **target** (*str*) – target column in frame
- **by** (*str*) – column in frame that will be calculated badrate by it
- **freq** (*str*) – offset aliases string by pandas <http://pandas.pydata.org/pandas-docs/stable/timeseries.html#offset-aliases>
- **format** (*str*) – format string for time
- **return_counts** (*bool*) – if need return counts plot
- **return_frame** (*bool*) – if need return frame

Returns badrate plot Axes: counts plot Axes: proportion plot Dataframe: grouping detail data

Return type Axes

`toad.plot.corr_plot` (*frame*, *figure_size=(20, 15)*)

plot for correlation

Parameters **frame** (*DataFrame*) – frame to draw plot

Returns Axes

`toad.plot.proportion_plot` (*x=None*, *keys=None*)

plot for comparing proportion in different dataset

Parameters

- **x** (*Series|list*) – series or list of series data for plot
- **keys** (*str|list*) – keys for each data

Returns Axes

`toad.plot.roc_plot` (*score*, *target*)

plot for roc

Parameters

- **score** (*array-like*) – predicted score
- **target** (*array-like*) – true target

Returns Axes

`toad.plot.bin_plot` (*frame*, *x=None*, *target='target'*, *iv=True*)

plot for bins

Parameters

- **frame** (*DataFrame*) –
- **x** (*str*) – column in frame that will be used as x axis
- **target** (*str*) – target column in frame

- **iv** (*bool*) – if need to show iv in plot

Returns bins' proportion and badrate plot

Return type Axes

3.2.5 toad.scorecard module

class toad.scorecard.**ScoreCard** (*pdo=60, rate=2, base_odds=35, base_score=750, card=None, combiner={}, transer=None, **kwargs*)

Bases: sklearn.base.BaseEstimator

coef_
coef of LR model

generate_card (*card=None*)

Parameters **card** (*dict | str | IOBase*) – dict of card or io to read json

fit (*X, y*)

Parameters

- **X** (*2D DataFrame*) –
- **Y** (*array-like*) –

set_card (*card*)
set card dict

set_combiner (*combiner*)
set combiner

set_score (*map*)
set score map by dict

predict (*X, **kwargs*)
predict score :param X: X to predict :type X: 2D array-like :param return_sub: if need to return sub score, default *False* :type return_sub: bool

Returns predicted score DataFrame: sub score for each feature

Return type array-like

proba_to_score (*prob*)
covert probability to score

combine (*X*)

bin_to_score (*bins, return_sub=False*)
predict score from bins

woe_to_score (*woe, weight=None*)
calculate score by woe

set_model (*model*)
set logistic regression model

generate_map (*transer, model*)
calculate score map by woe

export (*to_frame=False, to_json=None, to_csv=None, decimal=2*)
generate a scorecard object

Parameters

- **to_frame** (*bool*) – return DataFrame of card
- **to_json** (*str* | *IOBase*) – io to write json file
- **to_csv** (*filepath* | *IOBase*) – file to write csv

Returns dict

testing_frame (***kwargs*)
get testing frame with score

Returns testing frame with score

Return type DataFrame

3.2.6 toad.selection module

class toad.selection.**StatsModel** (*estimator='ols', criterion='aic', intercept=False*)
Bases: object

get_estimator (*name*)

stats (*X, y*)

get_criterion (*pre, y, k*)

t_value (*pre, y, X, coef*)

p_value (*t, n*)

loglikelihood (*pre, y, k*)

toad.selection.**stepwise** (*frame, target='target', estimator='ols', direction='both', criterion='aic',
p_enter=0.01, p_remove=0.01, p_value_enter=0.2, intercept=False,
max_iter=None, return_drop=False, exclude=None*)

stepwise to select features

Parameters

- **frame** (*DataFrame*) – dataframe that will be use to select
- **target** (*str*) – target name in frame
- **estimator** (*str*) – model to use for stats
- **direction** (*str*) – direction of stepwise, support 'forward', 'backward' and 'both', suggest 'both'
- **criterion** (*str*) – criterion to statistic model, support 'aic', 'bic'
- **p_enter** (*float*) – threshold that will be used in 'forward' and 'both' to keep features
- **p_remove** (*float*) – threshold that will be used in 'backward' to remove features
- **intercept** (*bool*) – if have intercept
- **p_value_enter** (*float*) – threshold that will be used in 'both' to remove features
- **max_iter** (*int*) – maximum number of iterate
- **return_drop** (*bool*) – if need to return features' name who has been dropped
- **exclude** (*array-like*) – list of feature names that will not be dropped

Returns selected dataframe array: list of feature names that has been dropped

Return type DataFrame

`toad.selection.drop_empty` (*frame*, *threshold*=0.9, *nan*=None, *return_drop*=False, *exclude*=None)
drop columns by empty

Parameters

- **frame** (*DataFrame*) – dataframe that will be used
- **threshold** (*number*) – drop the features whose empty num is greater than threshold. if threshold is float, it will be use as percentage
- **nan** (*any*) – values will be look like empty
- **return_drop** (*bool*) – if need to return features' name who has been dropped
- **exclude** (*array-like*) – list of feature names that will not be dropped

Returns selected dataframe array: list of feature names that has been dropped

Return type DataFrame

`toad.selection.drop_var` (*frame*, *threshold*=0, *return_drop*=False, *exclude*=None)
drop columns by variance

Parameters

- **frame** (*DataFrame*) – dataframe that will be used
- **threshold** (*float*) – drop features whose variance is less than threshold
- **return_drop** (*bool*) – if need to return features' name who has been dropped
- **exclude** (*array-like*) – list of feature names that will not be dropped

Returns selected dataframe array: list of feature names that has been dropped

Return type DataFrame

`toad.selection.drop_corr` (*frame*, *target*=None, *threshold*=0.7, *by*='IV', *return_drop*=False, *exclude*=None)
drop columns by correlation

Parameters

- **frame** (*DataFrame*) – dataframe that will be used
- **target** (*str*) – target name in dataframe
- **threshold** (*float*) – drop features that has the smallest weight in each groups whose correlation is greater than threshold
- **by** (*array-like*) – weight of features that will be used to drop the features
- **return_drop** (*bool*) – if need to return features' name who has been dropped
- **exclude** (*array-like*) – list of feature names that will not be dropped

Returns selected dataframe array: list of feature names that has been dropped

Return type DataFrame

`toad.selection.drop_iv` (*frame*, *target*='target', *threshold*=0.02, *return_drop*=False, *return_iv*=False, *exclude*=None)
drop columns by IV

Parameters

- **frame** (*DataFrame*) – dataframe that will be used
- **target** (*str*) – target name in dataframe

- **threshold** (*float*) – drop the features whose IV is less than threshold
- **return_drop** (*bool*) – if need to return features' name who has been dropped
- **return_iv** (*bool*) – if need to return features' IV
- **exclude** (*array-like*) – list of feature names that will not be dropped

Returns selected dataframe array: list of feature names that has been dropped Series: list of features' IV

Return type DataFrame

`toad.selection.drop_vif(frame, threshold=3, return_drop=False, exclude=None)`
variance inflation factor

Parameters

- **frame** (*DataFrame*) –
- **threshold** (*float*) – drop features until all vif is less than threshold
- **return_drop** (*bool*) – if need to return features' name who has been dropped
- **exclude** (*array-like*) – list of feature names that will not be dropped

Returns selected dataframe array: list of feature names that has been dropped

Return type DataFrame

`toad.selection.select(frame, target='target', empty=0.9, iv=0.02, corr=0.7, return_drop=False, exclude=None)`
select features by rate of empty, iv and correlation

Parameters

- **frame** (*DataFrame*) –
- **target** (*str*) – target's name in dataframe
- **empty** (*number*) – drop the features which empty num is greater than threshold. if threshold is float, it will be use as percentage
- **iv** (*float*) – drop the features whose IV is less than threshold
- **corr** (*float*) – drop features that has the smallest IV in each groups which correlation is greater than threshold
- **return_drop** (*bool*) – if need to return features' name who has been dropped
- **exclude** (*array-like*) – list of feature name that will not be dropped

Returns selected dataframe dict: list of dropped feature names in each step

Return type DataFrame

3.2.7 toad.stats module

`toad.stats.gini(target)`
get gini index of a feature

Parameters **target** (*array-like*) – list of target that will be calculate gini

Returns gini value

Return type number

`toad.stats.gini_cond`

get conditional gini index of a feature

Parameters

- **feature** (*array-like*) –
- **target** (*array-like*) –

Returns conditional gini value. If feature is continuous, it will return the best gini value when the feature bins into two groups

Return type number

`toad.stats.entropy(target)`

get information entropy of a feature

Parameters **target** (*array-like*) –

Returns information entropy

Return type number

`toad.stats.entropy_cond`

get conditional entropy of a feature

Parameters

- **feature** (*array-like*) –
- **target** (*array-like*) –

Returns conditional information entropy. If feature is continuous, it will return the best entropy when the feature bins into two groups

Return type number

`toad.stats.probability(target, mask=None)`

get probability of target by mask

`toad.stats.WOE(y_prob, n_prob)`

get WOE of a group

Parameters

- **y_prob** – the probability of grouped y in total y
- **n_prob** – the probability of grouped n in total n

Returns woe value

Return type number

`toad.stats.IV`

get the IV of a feature

Parameters

- **feature** (*array-like*) –
- **target** (*array-like*) –
- **n_bins** (*int*) – n groups that the feature will bin into
- **method** (*str*) – the strategy to be used to merge feature, default is 'dt'
- **()** (***kwargs*) – other options for merge function

`toad.stats.badrate(target)`
calculate badrate

Parameters `target` (*array-like*) – target array which *I* is bad

Returns float

`toad.stats.VIF(frame)`
calculate vif

Parameters `frame` (*ndarray|DataFrame*) –

Returns Series

`toad.stats.column_quality(feature, target, name='feature', iv_only=False, **kwargs)`
calculate quality of a feature

Parameters

- **feature** (*array-like*) –
- **target** (*array-like*) –
- **name** (*str*) – feature's name that will be setted in the returned Series
- **iv_only** (*bool*) – if only calculate IV

Returns a list of quality with the feature's name

Return type Series

`toad.stats.quality(dataframe, target='target', iv_only=False, **kwargs)`
get quality of features in data

Parameters

- **dataframe** (*DataFrame*) – dataframe that will be calculate quality
- **target** (*str*) – the target's name in dataframe
- **iv_only** (*bool*) – if only calculate IV

Returns quality of features with the features' name as row name

Return type DataFrame

3.2.8 toad.transform module

class `toad.transform.Transformer`

Bases: `sklearn.base.TransformerMixin`, `toad.utils.mixin.SaveMixin`

Base class for transformers

fit()

fit method, see details in *fit_* method

transform (*X*, **args*, ***kwargs*)

transform method, see details in *transform_* method

export (***kwargs*)

fit_transform (*X*, *y=None*, ***fit_params*)

Fit to data, then transform it.

Fits transformer to *X* and *y* with optional parameters *fit_params* and returns a transformed version of *X*.

Parameters

- **X** (*numpy array of shape [n_samples, n_features]*) – Training set.
- **y** (*numpy array of shape [n_samples]*) – Target values.
- ****fit_params** (*dict*) – Additional fit parameters.

Returns **X_new** – Transformed array.

Return type *numpy array of shape [n_samples, n_features_new]*

load (*rules, update=False, **kwargs*)

class `toad.transform.WOETransformer`

Bases: `toad.transform.Transformer`

WOE transformer

fit_ (*X, y*)

fit WOE transformer

Parameters

- **X** (*DataFrame/array-like*) –
- **y** (*str/array-like*) –
- **select_dtypes** (*str/numpy.dtypes*) – ‘object’, ‘number’ etc. only selected dtypes will be transform

transform_ (*rule, X, default='min'*)

transform function for single feature

Parameters

- **X** (*array-like*) –
- **default** (*str*) – ‘min’ (default), ‘max’ - the strategy to be used for unknown group

Returns *array-like*

export (***kwargs*)

fit ()

fit method, see details in *fit_* method

fit_transform (*X, y=None, **fit_params*)

Fit to data, then transform it.

Fits transformer to X and y with optional parameters *fit_params* and returns a transformed version of X.

Parameters

- **X** (*numpy array of shape [n_samples, n_features]*) – Training set.
- **y** (*numpy array of shape [n_samples]*) – Target values.
- ****fit_params** (*dict*) – Additional fit parameters.

Returns **X_new** – Transformed array.

Return type *numpy array of shape [n_samples, n_features_new]*

load (*rules, update=False, **kwargs*)

transform (*X, *args, **kwargs*)

transform method, see details in *transform_* method

class toad.transform.Combiner

Bases: *toad.transform.Transformer*

Combiner for merge data

fit_(*X*, *y=None*, *method='chi'*, *empty_separate=False*, ***kwargs*)
fit combiner

Parameters

- **X** (*DataFrame* / *array-like*) – features to be combined
- **y** (*str* / *array-like*) – target data or name of target in *X*
- **method** (*str*) – the strategy to be used to merge *X*, same as *.merge*, default is *chi*
- **n_bins** (*int*) – counts of bins will be combined
- **empty_separate** (*bool*) – if need to combine empty values into a separate group

transform_(*rule*, *X*, *labels=False*, ***kwargs*)
transform *X* by combiner

Parameters

- **X** (*DataFrame* / *array-like*) – features to be transformed
- **labels** (*bool*) – if need to use labels for resulting bins, *False* by default

Returns *array-like*

set_rules(*map*, *reset=False*)
set rules for combiner

Parameters

- **map** (*dict* / *array-like*) – map of splits
- **reset** (*bool*) – if need to reset combiner

Returns *self*

export (***kwargs*)

fit ()
fit method, see details in *fit_* method

fit_transform(*X*, *y=None*, ***fit_params*)
Fit to data, then transform it.

Fits transformer to *X* and *y* with optional parameters *fit_params* and returns a transformed version of *X*.

Parameters

- **X** (*numpy array of shape [n_samples, n_features]*) – Training set.
- **y** (*numpy array of shape [n_samples]*) – Target values.
- ****fit_params** (*dict*) – Additional fit parameters.

Returns *X_new* – Transformed array.

Return type *numpy array of shape [n_samples, n_features_new]*

load(*rules*, *update=False*, ***kwargs*)

transform(*X*, **args*, ***kwargs*)
transform method, see details in *transform_* method

```

class toad.transform.GBDTTransformer
    Bases: toad.transform.Transformer

    GBDT transformer

    fit_(X, y, **kwargs)
        fit GBDT transformer

        Parameters
            • X (DataFrame/array-like) –
            • y (str/array-like) –
            • select_dtypes (str/numpy.dtypes) – ‘object’, ‘number’ etc. only selected
              dtypes will be transform,

    transform_(rules, X)
        transform woe

        Parameters X (DataFrame/array-like) –
        Returns array-like

    export (**kwargs)

    fit()
        fit method, see details in fit_ method

    fit_transform(X, y=None, **fit_params)
        Fit to data, then transform it.

        Fits transformer to X and y with optional parameters fit_params and returns a transformed version of X.

        Parameters
            • X (numpy array of shape [n_samples, n_features]) – Training set.
            • y (numpy array of shape [n_samples]) – Target values.
            • **fit_params (dict) – Additional fit parameters.

        Returns X_new – Transformed array.

        Return type numpy array of shape [n_samples, n_features_new]

    load(rules, update=False, **kwargs)

    transform(X, *args, **kwargs)
        transform method, see details in transform_ method

```

3.2.9 toad.utils module

toad.utils.func module

```

class toad.utils.func.Parallel
    Bases: object

    apply(func, args=(), kwargs={})

    join()

toad.utils.func.np_count(arr, value, default=None)

toad.utils.func.has_nan(arr)

```

```

toad.utils.func.np_unique (arr, **kwargs)
toad.utils.func.to_ndarray (s, dtype=None)
toad.utils.func.fillna (feature, by=-1)
toad.utils.func.bin_by_splits (feature, splits)
    Bin feature by split points
toad.utils.func.feature_splits (feature, target)
    find possibility spilt points
toad.utils.func.iter_df (dataframe, feature, target, splits)
    iterate dataframe by split points

Returns iterator (df, splitter)

toad.utils.func.inter_feature (feature, splits)
toad.utils.func.is_continuous (series)
toad.utils.func.split_target (frame, target)
toad.utils.func.unpack_tuple (x)
toad.utils.func.generate_str (size=6, chars='ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789')
toad.utils.func.save_json (contents, file, indent=4)
    save json file

```

Parameters

- **contents** (*dict*) – contents to save
- **file** (*str* / *IOBase*) – file to save

```

toad.utils.func.read_json (file)
    read json file
toad.utils.func.clip (series, value=None, std=None, quantile=None)
    clip series

```

Parameters

- **series** (*array-like*) – series need to be clipped
- **value** (*number* | *tuple*) – min/max value of clipping
- **std** (*number* | *tuple*) – min/max std of clipping
- **quantile** (*number* | *tuple*) – min/max quantile of clipping

```

toad.utils.func.diff_time (base, target, format=None, time='day')
toad.utils.func.diff_time_frame (base, frame, format=None)
toad.utils.func.bin_to_number (reg=None)

```

Returns func(string) -> number

Return type function

```

toad.utils.func.generate_target (size, rate=0.5, weight=None, reverse=False)
    generate target for reject inference

```

Parameters

- **size** (*int*) – size of target

- **rate** (*float*) – rate of '1' in target
- **weight** (*array-like*) – weight of '1' to generate target
- **reverse** (*bool*) – if need reverse weight

Returns array

```
toad.utils.func.get_dummies (dataframe, exclude=None, binary_drop=False, **kwargs)
    get dummies
```

toad.utils.decorator module

```
class toad.utils.decorator.Decorator (*args, is_class=False, **kwargs)
    Bases: object
    base decorater class
    is_class = False
    setup (*args, **kwargs)
    call (*args, **kwargs)
    wrapper (*args, **kwargs)

class toad.utils.decorator.frame_exclude (*args, is_class=False, **kwargs)
    Bases: toad.utils.decorator.Decorator
    decorator for exclude columns
    wrapper (X, *args, exclude=None, **kwargs)

class toad.utils.decorator.select_dtypes (*args, is_class=False, **kwargs)
    Bases: toad.utils.decorator.Decorator
    decorator for select frame by dtypes
    wrapper (X, *args, select_dtypes=None, **kwargs)

class toad.utils.decorator.save_to_json (*args, is_class=False, **kwargs)
    Bases: toad.utils.decorator.Decorator
    support save result to json file
    wrapper (*args, to_json=None, **kwargs)

class toad.utils.decorator.load_from_json (*args, is_class=False, **kwargs)
    Bases: toad.utils.decorator.Decorator
    support load data from json file
    require_first = False
    wrapper (*args, from_json=None, **kwargs)

class toad.utils.decorator.support_dataframe (*args, is_class=False, **kwargs)
    Bases: toad.utils.decorator.Decorator
    decorator for supporting dataframe
    require_target = True
    target = 'target'
    wrapper (frame, *args, **kwargs)
```

```
class toad.utils.decorator.proxy_docstring(*args, is_class=False, **kwargs)
    Bases: toad.utils.decorator.Decorator

    method_name = None
```

3.3 Module contents

CHAPTER 4

Indices and tables

- `genindex`
- `modindex`
- `search`

t

- `toad`, [22](#)
- `toad.detector`, [5](#)
- `toad.merge`, [6](#)
- `toad.metrics`, [8](#)
- `toad.plot`, [10](#)
- `toad.scorecard`, [11](#)
- `toad.selection`, [12](#)
- `toad.stats`, [14](#)
- `toad.transform`, [16](#)
- `toad.utils.decorator`, [21](#)
- `toad.utils.func`, [19](#)

A

AIC() (in module *toad.metrics*), 8
 apply() (in module *toad.utils.func.Parallel* method), 19
 AUC() (in module *toad.metrics*), 9

B

badrate() (in module *toad.stats*), 15
 badrate_plot() (in module *toad.plot*), 10
 BIC() (in module *toad.metrics*), 8
 bin_by_splits() (in module *toad.utils.func*), 20
 bin_plot() (in module *toad.plot*), 10
 bin_to_number() (in module *toad.utils.func*), 20
 bin_to_score() (in module *toad.scorecard.ScoreCard* method), 11

C

call() (in module *toad.utils.decorator.Decorator* method), 21
 ChiMerge() (in module *toad.merge*), 6
 clip() (in module *toad.utils.func*), 20
 coef_ (in module *toad.scorecard.ScoreCard* attribute), 11
 column_quality() (in module *toad.stats*), 16
 combine() (in module *toad.scorecard.ScoreCard* method), 11
 Combiner (class in *toad.transform*), 17
 corr_plot() (in module *toad.plot*), 10
 countBlank() (in module *toad.detector*), 6

D

Decorator (class in *toad.utils.decorator*), 21
 detect() (in module *toad.detector*), 6
 diff_time() (in module *toad.utils.func*), 20
 diff_time_frame() (in module *toad.utils.func*), 20
 drop_corr() (in module *toad.selection*), 13
 drop_empty() (in module *toad.selection*), 12
 drop_iv() (in module *toad.selection*), 13
 drop_var() (in module *toad.selection*), 13
 drop_vif() (in module *toad.selection*), 14
 DTMerge() (in module *toad.merge*), 6

E

entropy() (in module *toad.stats*), 15

entropy_cond (in module *toad.stats*), 15
 export() (in module *toad.scorecard.ScoreCard* method), 11
 export() (in module *toad.transform.Combiner* method), 18
 export() (in module *toad.transform.GBDTTransformer* method), 19
 export() (in module *toad.transform.Transformer* method), 16
 export() (in module *toad.transform.WOETransformer* method), 17

F

F1() (in module *toad.metrics*), 9
 feature_splits() (in module *toad.utils.func*), 20
 fillna() (in module *toad.utils.func*), 20
 fit() (in module *toad.scorecard.ScoreCard* method), 11
 fit() (in module *toad.transform.Combiner* method), 18
 fit() (in module *toad.transform.GBDTTransformer* method), 19
 fit() (in module *toad.transform.Transformer* method), 16
 fit() (in module *toad.transform.WOETransformer* method), 17
 fit_() (in module *toad.transform.Combiner* method), 18
 fit_() (in module *toad.transform.GBDTTransformer* method), 19
 fit_() (in module *toad.transform.WOETransformer* method), 17
 fit_transform() (in module *toad.transform.Combiner* method), 18
 fit_transform() (in module *toad.transform.GBDTTransformer* method), 19
 fit_transform() (in module *toad.transform.Transformer* method), 16
 fit_transform() (in module *toad.transform.WOETransformer* method), 17
 frame_exclude (class in *toad.utils.decorator*), 21

G

GBDTTransformer (class in *toad.transform*), 18
 generate_card() (in module *toad.scorecard.ScoreCard* method), 11
 generate_map() (in module *toad.scorecard.ScoreCard* method), 11
 generate_str() (in module *toad.utils.func*), 20
 generate_target() (in module *toad.utils.func*), 20

`get_criterion()` (*toad.selection.StatsModel method*), 12
`get_dummies()` (*in module toad.utils.func*), 21
`get_estimator()` (*toad.selection.StatsModel method*), 12
`getDescribe()` (*in module toad.detector*), 5
`getTopValues()` (*in module toad.detector*), 5
`gini()` (*in module toad.stats*), 14
`gini_cond` (*in module toad.stats*), 14

H

`has_nan()` (*in module toad.utils.func*), 19

I

`inter_feature()` (*in module toad.utils.func*), 20
`is_class` (*toad.utils.decorator.Decorator attribute*), 21
`is_continuous()` (*in module toad.utils.func*), 20
`isNumeric()` (*in module toad.detector*), 6
`iter_df()` (*in module toad.utils.func*), 20
IV (*in module toad.stats*), 15

J

`join()` (*toad.utils.func.Parallel method*), 19

K

`KMeansMerge()` (*in module toad.merge*), 7
`KS()` (*in module toad.metrics*), 8
`KS_bucket()` (*in module toad.metrics*), 8
`KS_by_col()` (*in module toad.metrics*), 8

L

`load()` (*toad.transform.Combiner method*), 18
`load()` (*toad.transform.GBDTTransformer method*), 19
`load()` (*toad.transform.Transformer method*), 17
`load()` (*toad.transform.WOETransformer method*), 17
`load_from_json` (*class in toad.utils.decorator*), 21
`loglikelihood()` (*toad.selection.StatsModel method*), 12

M

`matrix()` (*in module toad.metrics*), 9
`merge` (*in module toad.merge*), 7
`method_name` (*toad.utils.decorator.proxy_docstring attribute*), 22
`MSE()` (*in module toad.metrics*), 8

N

`np_count()` (*in module toad.utils.func*), 19
`np_unique()` (*in module toad.utils.func*), 19

P

`p_value()` (*toad.selection.StatsModel method*), 12
`Parallel` (*class in toad.utils.func*), 19

`predict()` (*toad.scorecard.ScoreCard method*), 11
`proba_to_score()` (*toad.scorecard.ScoreCard method*), 11
`probability()` (*in module toad.stats*), 15
`proportion_plot()` (*in module toad.plot*), 10
`proxy_docstring` (*class in toad.utils.decorator*), 21
`PSI()` (*in module toad.metrics*), 9

Q

`quality()` (*in module toad.stats*), 16
`QuantileMerge()` (*in module toad.merge*), 7

R

`read_json()` (*in module toad.utils.func*), 20
`require_first` (*toad.utils.decorator.load_from_json attribute*), 21
`require_target` (*toad.utils.decorator.support_dataframe attribute*), 21
`roc_plot()` (*in module toad.plot*), 10

S

`save_json()` (*in module toad.utils.func*), 20
`save_to_json` (*class in toad.utils.decorator*), 21
`ScoreCard` (*class in toad.scorecard*), 11
`select()` (*in module toad.selection*), 14
`select_dtypes` (*class in toad.utils.decorator*), 21
`set_card()` (*toad.scorecard.ScoreCard method*), 11
`set_combiner()` (*toad.scorecard.ScoreCard method*), 11
`set_model()` (*toad.scorecard.ScoreCard method*), 11
`set_rules()` (*toad.transform.Combiner method*), 18
`set_score()` (*toad.scorecard.ScoreCard method*), 11
`setup()` (*toad.utils.decorator.Decorator method*), 21
`split_target()` (*in module toad.utils.func*), 20
`SSE()` (*in module toad.metrics*), 8
`stats()` (*toad.selection.StatsModel method*), 12
`StatsModel` (*class in toad.selection*), 12
`StepMerge()` (*in module toad.merge*), 7
`stepwise()` (*in module toad.selection*), 12
`support_dataframe` (*class in toad.utils.decorator*), 21

T

`t_value()` (*toad.selection.StatsModel method*), 12
`target` (*toad.utils.decorator.support_dataframe attribute*), 21
`testing_frame()` (*toad.scorecard.ScoreCard method*), 12
`to_ndarray()` (*in module toad.utils.func*), 20
`toad` (*module*), 22
`toad.detector` (*module*), 5
`toad.merge` (*module*), 6
`toad.metrics` (*module*), 8
`toad.plot` (*module*), 10

toad.scorecard (*module*), 11
 toad.selection (*module*), 12
 toad.stats (*module*), 14
 toad.transform (*module*), 16
 toad.utils.decorator (*module*), 21
 toad.utils.func (*module*), 19
 transform() (*toad.transform.Combiner method*), 18
 transform() (*toad.transform.GBDTTransformer method*), 19
 transform() (*toad.transform.Transformer method*), 16
 transform() (*toad.transform.WOETransformer method*), 17
 transform_() (*toad.transform.Combiner method*), 18
 transform_() (*toad.transform.GBDTTransformer method*), 19
 transform_() (*toad.transform.WOETransformer method*), 17
 Transformer (*class in toad.transform*), 16

U

unpack_tuple() (*in module toad.utils.func*), 20

V

VIF() (*in module toad.stats*), 16

W

WOE() (*in module toad.stats*), 15
 woe_to_score() (*toad.scorecard.ScoreCard method*), 11
 WOETransformer (*class in toad.transform*), 17
 wrapper() (*toad.utils.decorator.Decorator method*), 21
 wrapper() (*toad.utils.decorator.frame_exclude method*), 21
 wrapper() (*toad.utils.decorator.load_from_json method*), 21
 wrapper() (*toad.utils.decorator.save_to_json method*), 21
 wrapper() (*toad.utils.decorator.select_dtypes method*), 21
 wrapper() (*toad.utils.decorator.support_dataframe method*), 21