fireTS Documentation

Release 1.0.1

Jinyu Xie

Nov 10, 2020

Contents

CHAPTER 1

fireTS

1.1 core module

class core.**GeneralAutoRegressor**(*base_estimator*, *auto_order*, *exog_order*, *exog_delay=None*, *pred_step=1*, ***base_params*)

Bases: *core.TimeSeriesRegressor*, sklearn.base.RegressorMixin

The general auto regression model can be written in the following form:

$$\begin{aligned} y(t+k) = \\ f(y(t), ..., y(t-p+1), \\ x_1(t-d_1), ..., x_1(t-d_1-q_1+1), \\ ..., x_m(t-d_1), ..., x_m(t-d_m-q_m+1)) + e(t) \end{aligned} \tag{1.1}$$

Parameters

- **base_estimator** (*object*) – an estimator object that implements the scikit-learn API (fit, and predict). The estimator will be used to fit the function f in equation (1.1).
- **auto_order** (*int*) – the autoregression order p in equation (1.1).
- **exog_order** (*list*) – the exogenous input order, a list of integers representing the order for each exogenous input, i.e. $[q_1, q_2, ..., q_m]$ in equation (1.1).
- **exog_delay** (*list*) – the delays of the exogenous inputs, a list of integers representing the delay of each exogenous input, i.e. $[d_1, d_2, ..., d_m]$ in equation (1.1). By default, all the delays are set to 0.
- **pred_step** (*int*) – the prediction step k in equation (1.1). By default, it is set to 1.
- **base_params** (*dict*) – other keyword arguments for base_estimator.

fit(*X*, *y*, ***params*)

Create lag features and fit the base_estimator.

Parameters

- **X** (*array-like*) – exogenous input time series, shape = (n_samples, n_exog_inputs)
- **y** (*array-like*) – target time series to predict, shape = (n_samples)

grid_search(*X*, *y*, *para_grid*, ***params*)
Perform grid search on the base_estimator. The function first generates the lag features and predicting targets, and then calls `GridSearchCV` in scikit-learn package.

Parameters

- **X** (*array-like*) – exogenous input time series, shape = (n_samples, n_exog_inputs)
- **y** (*array-like*) – target time series to predict, shape = (n_samples)
- **para_grid** (*dict*) – use the same format in `GridSearchCV` in scikit-learn package.
- **params** (*dict*) – other keyword arguments that can be passed into `GridSearchCV` in scikit-learn package.

class core.**TimeSeriesRegressor**(*base_estimator*, ***base_params*)
Bases: `sklearn.base.BaseEstimator`, `sklearn.base.RegressorMixin`

TimeSeriesRegressor creates a time series model based on a general-purpose regression model defined in base_estimator. base_estimator must be a model which implements the scikit-learn APIs.

set_params(***params*)
Set the parameters of this estimator.

The method works on simple estimators as well as on nested objects (such as pipelines). The latter have parameters of the form `<component>__<parameter>` so that it's possible to update each component of a nested object.

****params** [dict] Estimator parameters.

self [object] Estimator instance.

1.2 models module

class models.**DirectAutoRegressor**(*base_estimator*, *auto_order*, *exog_order*, *pred_step*, *exog_delay=None*, ***base_params*)
Bases: `fireTS.core.GeneralAutoRegressor`

This model performs autoregression with exogenous inputs on the k-step ahead output directly. The model equation is written as follows.

$$\begin{aligned} y(t+k) = \\ f(y(t), ..., y(t-p+1), \\ x_1(t-d_1), ..., x_1(t-d_1-q_1+1), \\ ..., x_m(t-d_1), ..., x_m(t-d_m-q_m+1)) + e(t) \end{aligned} \tag{1.2}$$

Parameters

- **base_estimator** (*object*) – an estimator object that implements the scikit-learn API (fit, and predict). The estimator will be used to fit the function f in equation (1.2).
- **auto_order** (*int*) – the autoregression order p in equation (1.2).

- **exog_order** (*list*) – the exogenous input order, a list of integers representing the order for each exogenous input, i.e. $[q_1, q_2, ..., q_m]$ in equation (1.2).
- **pred_step** (*int*) – the prediction step k in equation (1.1). By default, it is set to 1.
- **exog_delay** (*list*) – the delays of the exogenous inputs, a list of integers representing the delay of each exogenous input, i.e. $[d_1, d_2, ..., d_m]$ in equation (1.2). By default, all the delays are set to 0.
- **base_params** (*dict*) – other keyword arguments for base_estimator.

predict (*X*, *y*)

Produce multi-step prediction of y. The multi-step prediction is done directly. No future X inputs are used in the prediction. The prediction equation is as follows:

$$\hat{y}(t+k) = f(y(t), ..., y(t-p+1), x_1(t-d_1), ..., x_1(t-d_1-q_1+1) ..., x_m(t-d_m), ..., x_m(t-d_m-q_m+1))$$

Parameters

- **X** (*array-like*) – exogenous input time series, shape = (n_samples, n_exog_inputs)
- **y** (*array-like*) – target time series to predict, shape = (n_samples)
- **step** (*int*) – prediction step.

Returns k-step prediction time series, shape = (n_samples). The i th value of the output is the k-step prediction of the i th value of the input `y`. The first `pred_step + max(auto_order - 1, max(exog_order + exog_delay) - 1)` values of the output is `np.nan`.

score (*X*, *y*, *method='r2'*, *verbose=False*)

Produce multi-step prediction of y, and compute the metrics against y. Nan is ignored when computing the metrics.

Parameters

- **X** (*array-like*) – exogenous input time series, shape = (n_samples, n_exog_inputs)
- **y** (*array-like*) – target time series to predict, shape = (n_samples)
- **method** (*string*) – could be "r2" (R Square) or "mse" (Mean Square Error).

Returns prediction metric. Nan is ignored when computing the metrics.

class models.**NARX** (*base_estimator*, *auto_order*, *exog_order*, *exog_delay=None*, ***base_params*)

Bases: `fireTS.core.GeneralAutoRegressor`

NARX stands for Nonlinear AutoRegressive eXogenous model. The model equation is written as follows.

$$y(t+1) = f(y(t), ..., y(t-p+1), x_1(t-d_1), ..., x_1(t-d_1-q_1+1), ..., x_m(t-d_1), ..., x_m(t-d_m-q_m+1)) + e(t) \tag{1.3}$$

Parameters

- **base_estimator** (*object*) – an estimator object that implements the scikit-learn API (fit, and predict). The estimator will be used to fit the function f in equation (1.3).
- **auto_order** (*int*) – the autoregression order p in equation (1.3).
- **exog_order** (*list*) – the exogenous input order, a list of integers representing the order for each exogenous input, i.e. $[q_1, q_2, ..., q_m]$ in equation (1.3).
- **exog_delay** (*list*) – the delays of the exogenous inputs, a list of integers representing the delay of each exogenous input, i.e. $[d_1, d_2, ..., d_m]$ in equation (1.3). By default, all the delays are set to 0.
- **base_params** (*dict*) – other keyword arguments for base_estimator.

forecast (*X*, *y*, *step=1*, *X_future=None*)

Forecast y multiple step ahead given the exogenous input history X, output history y and the future exogenous input X_future. X_future is assumed to be all zeros if not specified.

Parameters

- **X** (*array-like*) – exogenous input time series, shape = (n_samples, n_exog_inputs)
- **y** (*array-like*) – target time series to predict, shape = (n_samples)
- **step** (*int*) – prediction step.
- **X_futrue** (*array-like*) – future exogenous input time series, shape = (step - 1, n_exog_inputs)

Returns multi-step forecasted time series, shape = (step).

predict (*X*, *y*, *step=1*)

Produce multi-step prediction of y. The multi-step prediction is done recursively by using the future inputs in X. The prediction equation is as follows:

$$\hat{y}(t+k) = f(\hat{y}(t+k-1), ..., \hat{y}(t+k-p),$$

$$x_1(t+k-1-d_1), ..., x_1(t+k-d_1-q_1)$$

$$..., x_m(t+k-1-d_m), ..., x_m(t+k-d_m-q_m))$$

Parameters

- **X** (*array-like*) – exogenous input time series, shape = (n_samples, n_exog_inputs)
- **y** (*array-like*) – target time series to predict, shape = (n_samples)
- **step** (*int*) – prediction step.

Returns k-step prediction time series, shape = (n_samples). The i th value of the output is the k-step prediction of the i th value of the input `y`. The first `step + max(auto_order - 1, max(exog_order + exog_delay) - 1)` values of the output is `np.nan`.

score (*X*, *y*, *step=1*, *method='r2'*)

Produce multi-step prediction of y, and compute the metrics against y. Nan is ignored when computing the metrics.

Parameters

- **X** (*array-like*) – exogenous input time series, shape = (n_samples, n_exog_inputs)

- **y** (*array-like*) – target time series to predict, shape = (n_samples)
- **step** (*int*) – prediction step.
- **method** (*string*) – could be "r2" (R Square) or "mse" (Mean Square Error).

Returns prediction metric. Nan is ignored when computing the metrics.

1.3 utils module

class utils.**InputLagFeatureProcessor**(*data*, *order*, *delay*)
Bases: object

generate_lag_features()

update()

class utils.**MetaLagFeatureProcessor**(*X*, *y*, *auto_order*, *exog_order*, *exog_delay*)
Bases: object

generate_lag_features()

update(*data_new*)

class utils.**OutputLagFeatureProcessor**(*data*, *order*)
Bases: object

generate_lag_features()

update(*data_new*)

utils.**shift**(*darray*, *k*, *axis=0*)
Utility function to shift a numpy array

darray: a numpy array the array to be shifted.

k: integer number of shift

axis: non-negative integer axis to perform shift operation

shifted numpy array, fill the unknown values with nan

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

c

m

u

Index