stockDL Documentation

Release v0.0.3

Ashish Papanai

Mar 02, 2021

CONTENTS:

CHAPTER

ONE

STOCKDL

1.1 stockDL package

1.1.1 Submodules

1.1.2 stockDL.calculations module

This module comprises all functions to calculate the net yield and gross yield.

class stockDL.calculations.**Calculations**
Bases: object

gross_portfolio(*df*, *w*)

gross_yield(*df*, *v*)

net_yield(*df*, *v*)

separate_ones(*u*)

1.1.3 stockDL.data module

This module helps in data collection from Yahoo Finance API with the help of yfinace library. The stock data is loaded from its unique stock symbol, also called ticker. After the data is loaded, we drop the axis which isn't being used by us in the library. After dropping the unnecessary axis we calculate and store the first days of each trading month in a list.

class stockDL.data.**Data_Loader**(*ticker*)
Bases: object

date_calculations()

end_date
Stores the first day of each month from the starting date of stock to the day of executing the library

load_data()

1.1.4 stockDL.main module

This Module handles the generation of results in the CLI, based on the ticker passed as the argument to the object of the Main class. The execution begins by retrieving the financial data from yfinace library which is handled in the data module of stockDL. After the data is collected the first trading day of each month is stored in a list by the date_calculation() function in the data module. The data is then preprocessed by the preprocessor module which creates a window to run the predictions on. It also comprises min-max scaling which reduces sudden highs and lows in the data which would have resulted in anomalies. The training module trains the model created in the model module on the data retrieved by the preprocessing module. The plots module helps in plotting the necessary graphs for better visualisation for EDA. The result module uses the calculation and market modules to run the necessary calculations on the predictions, and generate net and the gross yield on the predictions obtained.

class stockDL.main.**Main**(*ticker*)

Bases: object

data_preprocessor
This variable stores the monthly data for analysis by the model

df_monthly
This instance of train module will help in preventing the training to occur more than once thus reducing the probability of overfitting.

plots
Generates a comparison plot of the 4 methos used

result
Converts the pandas dataframe to JSON for better utilisation by web developers

results
Stores the final result in a pandas dataframe

ticker
this module helps in loading the data by calling an instance of data.py module and also helps in data preprocessing.

train
Trains the data on the defined models

1.1.5 stockDL.market module

This module stores the stock market variables related to the stock ticker, This module must be run after training the model by calling the train_model() function in the train module.

class stockDL.market.**Market**(*ticker*)

Bases: object

v_lstm
Stores the sign of the lstm predictions as +ve, -ve or zero which implies profit, loss and no profit no loss in the trade.

v_mix
Stores the sign of the mix model predictions as +ve, -ve or zero which implies profit, loss and no profit no loss in the trade.

w_lstm
This stores the predictions of the lstm model and reshapes it.

w_mix
This stores the predictions of the mix model (conv1D + LSTM) and reshapes it.

1.1.6 stockDL.models module

This module stores the brain of the library which is the Deep Learning model. The two Deep Learning strategies are defined in their respective methods. This module requires to run the preprocessing module so that the model gets the data to work on.

class stockDL.models.**Models**(*ticker*)
Bases: object

LSTM_Model(*window*, *features*)

Mix_LSTM_Model(*window*, *features*)

1.1.7 stockDL.plots module

This module helps in plotting the various details related to the model and the predictions, This module requires the processed data from the pre-processing module, training data from the train module, Market information from the market module, and the calculations from the calculations module.

class stockDL.plots.**Plots**(*ticker*)
Bases: object

comparison_plots()

in_out()

plot_predictions()

plot_training_data(*model_selected*, *metric='loss'*, *val_metric='val_loss'*)

1.1.8 stockDL.preprocessing module

This module makes the data ready for predictions and calculations, This module requires the data module to get the data from the Yahoo finance API. This module has 3 main components: 1. Convert the daily stock data to monthly data for analysis based on the opening prices for the month. 2. Creating an analysis window for predictions. [6 years is the window size in stockDL] 3. Making data RNN ready.

class stockDL.preprocessing.**data_preprocessing**(*ticker*)
Bases: object

create_window(*data*, *window_size*)

data_scaling(*dfm*)

monthly_df(*df*)

1.1.9 stockDL.results module

This is the final module responsible for the result calculation and parsing the dataframe to JSON. This module requires data from preprocessing, calculations, market and all their dependencies.

class stockDL.results.**Results**(*ticker*)
Bases: object

result_calculations()

1.1.10 stockDL.train module

This module handles the training of the two-deep learning strategies used in this library. It requires preprocessing and models modules and their dependencies.

class stockDL.train.**Training**(*ticker*)

Bases: object

learning_rate_reduction

Uncomment and add tensorboard to callbacks to run TensorBoard for visualisation self.tensorboard = keras.callbacks.TensorBoard(

log_dir='TensorBoard_Logs', histogram_freq=1, embeddings_freq=1

)

models

Prevents false minima by reducing the learning rates on plateaus.

train_model()

Stores the history of the LSTM model.

1.1.11 Module contents

Version of module

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

S

INDEX

T

V

W