Warsaw University of Technology

FACULTY OF
MATHEMATICS AND INFORMATION SCIENCE

Project of an application based on machine learning for stock market prediction
Code

Vladislav Sorokin

student record book number 295462

Dzianis Harbatsenka

student record book number 295452

Nikita Zakharov

student record book number 295464

thesis supervisor
prof. PW dr hab. inż., Jerzy Balicki

Warsaw 2021

Version 1.0

Warsaw, 24.11.20

DECLARATION

We declare that this piece of work which is the basis for recognition of achieving learning outcomes in the Group Project course was completed on our own.

Table of contents

SVR

Function to train SVR

```
function [dates, y, fxpr, Bestmse] = train_svr(dtname,
maxItr,file_name,isAutoHyper,kernelFunctionName,MaxEvalNu
m,epsilon,loopMaxKernelScale, loopMaxBoxConstraint)
input data table name, number of max iterations, file
name of the model, auto training, max number of
evaluations, epsilon value, iteration number for my
optimization of kernel scale, iteration number for my
optimization of box constraint.
Return is required values to plot the graph.
```

Function to predict price using SVR

```
function [datespr, ypr, rst,errors,mse,Mape] =
predict_svr(dtname,model,daysToPredict)
```

Input data table name, model name, how many days to predict.

LSTM

parameters.py

Below the global parameters are defined for the program. It includes the settings of LSTM network, testing procedure and other useful parameters.

stock_prediction.py

The below script servers as a help for the main function. The first function in it

```
def load_data(ticker, n_steps=50, scale=True, shuffle=True, lookup_step=1,
              test_size=0.2, feature_columns=['High', 'Low', 'Open',
'Close', 'Volume']):
```

The function load_data downloads the dataset from WIG_20 and transforms the data into the form comprehensive for the network. The result of the function and the way the data will look like depends on the parameters provided. The specific feature columns can be set. The dataset can be shuffled or not.

```
def create_model(input_length, units=256, cell=LSTM, n_layers=2,
dropout=0.3,
                 loss="mean_absolute_error", optimizer="rmsprop"):
```

The function create_model defines and initializes the network with the predefined settings.

start.py

The main script is provided below. The function InitUT initializes the GUI of the application. Then two functions for each button click event are provided (Train and Test).

```
def button1_clicked(self):
```

Triggers the training procedure to start. The model is initialized, the data is loaded and then the network starts to get fitted.

```
def button2_clicked(self):
```

Triggers the prediction of the model after training process is finished. At this stage the network evaluates the predicted values.

```
def plot_graph(self, model, data)
```

The function plots the graph to see the comparison between real and predicted values.

CNN

Function to initialize CNN

```
model_initialization <- function(timeseires_length)
```

Function to train CNN

```
model_training <- function(model, X, Y, cb)
```

Function to predict with CNN

```
model_prediction <- function(model, X)
```

Function to save a model

```
model_save <- function(model, name)
```

Function to load existing model

```
model_load <- function(name)
```

Function to prepare data for train, test and prediction

```
prepare_data <- function(data_frame, timeseires_length,
t_num, p_num = 5)
```

Function to make prediction based on backpropagation

```
real_prediction <- function(model,last_ts,pnum = 5)
```

Function to form timeseries out of input data

```
form_timeseriese <- function(open, close = NULL, steps)
```

Function to modify a timeseries by pop and push a value

```
append_timeseries <- function(ts, value)
```

Function to divide data into train, test and prediction sets

```
divide_data <- function(data, tnum, ts_length, pnum = 5)
```

Function to load data from a chosen csv

```
choose_data <- function(name)
```

Function to obtain existing models by name

```
get_models_names <- function(name)
```

Function to obtain names of data sets

```
get_data_names <- function()
```

Function to plot raw data set

```
plot_raw_data <- function(data_frame)
```

Function to plot test data

```
plot_test <- function(actual, test, ts_length)
```

Function to plot predicted data

```
plot_prediction <- function(actual, test)
```