**STOCK PREDICTION**

**Introduction**

This code was created for the sole purpose of the IC Course at the University of St. Gallen and shall not be used as any real guideline for investing purposes.

The Code is comprised of the following stages:

1. Extraction of S&P 500 Tickers from Wikipedia
2. Collection of historical data of those companies using QUANDL
3. Statistical Prediction of future movements using an LSTM model
4. Adding further data and analysis
   1. Collection of gold, dollar, etc. prices for dashboard
   2. Sentiment analysis and summarizer of news articles

This project aims to collect and analyze stock-relevant data in order to create a dashboard for private day traders or short-term investors.

**Extraction of S&P500 tickers**

This first step is the first part of the file S&P historische Daten abrufen.py. It utilized the pandas package and its read\_html functions to scrape the S&P 500 list from Wikipedia and saves it into a CSV file called SP500.csv.

**Collecting historical data with QUANDL**

In order to collect our database of all available data from the S&P500 companies we use the api from QUANDL. Therefore, an account had to be created with the API key that is hard-coded into the code in the file S&P historische Daten abrufen.py. With the created function get\_data\_from\_quandl() the previously read S&P500 list is read from a previously saved pickle and used in order to create a loop to go through all tickers and read the financial data from each company and save it into a separate csv file.

In a second stage the created function compile\_data() takes all the separate files that have been created for each ticker and creates one CSV file with the date on the y axis, the ticker on the x-axis and only using the for us relevant data point adjusted close (ignoring other figures such as volume, close, open, etc)

The function visualize\_data() was merely used for our sake to see if the extracted data is complete and also to see the correlation between the single data points, however this was not used for the final product.

The functions extract\_featuresets(ticker) & do\_ml(ticker) were previous attempts for the data processing/prediction, however are not used in the final product since the Machine Learning results did not suffice for our purpose.

**Prediction with LSTM model**

The prediction is found in the file Prediction for all 500 tickers as loop.py & Prediction single ticker.py. It begins by reading all separate tickers from the combined S&P500 historical data CSV file. With this list it creates a loop through all tickers that one by one creates the predictions and creates a CSV file with the results.

For this purpose the data first has to be prepared by removing all rows that have no values for the selected ticker on a certain date. With the MinMaxScaler() the data is further normalized. Splitting the data into 70% training data and 30% testing data we can already move on to the LSTM model. A Sequential deep learning model with 3 Layers was used. The first layer being the LSTM layer and two dense layers. After the prediction was created the Mean Squared error of the results is calculated and later collected in one overview CSV, which can give you an overview of the quality of the prediction for each ticker.

Result of this phase is that you have separate CSV files with prediction data for each ticker (unformatted results, which require some manual formatting for later use)

**Further data and analysis**

For the dashboard further data was needed, therefore there is a code that extracts the gold, oil etc. prices (see Additional Data extraction.py) and some news information was required. Since we only used this data for two example companies the code was not included to run over all tickers and also requires some manual aspects.

The extraction of oil and gold prices is once again done through QUANDL and can easily be increased if other data is required.

The News code is focused on the analysis and does not include the extraction of the news articles. (this was done manually for the chosen dates and companies) The code in Sentiment Analysis and Summarizer.py uses for one the VaderSentiment analysis and as a second part the Gensim Summarizer.

The SentimentAnalysis is used to find potential negative or positive press that can impact the stock price. The Summarizer could potentially be used on the dashboard in order to not only give the title, but also a small extract of the article that is created automatically. However this would till require some finetuning.

REQUIREMENTS (Python packages)

* gensim
* vaderSentiment
* pandas
* numpy
* sklearn
* matplotlib
* bs4
* quandl
* datetime
* csv
* os
* pandas\_datareader
* pickle
* requests
* collections
* keras
* tensorflow
* os
* importlib
* math