

Project definition

Project overview

In order to better understand stock market behaviour and to improve profitability of investments, hedge funds, investment firms and individual investors have been using financial models to predict future behaviour of stock prices. In recent times, these financial models have become easier to implement due to the abundance of data available about stock prices as well as information available on implementing financial models using python. Financial data relevant to stock prices are readily available through a number of websites using an API's to retrieve the data. Stock price data can be classified as time series data as it is sequential in nature. To make predictions or forecasts using time series data two commonly used approaches exist: classical machine learning models and deep learning models. Classical machine learning models have three main disadvantages namely:

1. They are unable to recognize complex patterns in data
2. Missing data values affect the performance of the model
3. They only work well for short term forecasts and struggle to capture long term trends

Deep learning, however, addresses these disadvantages. A particular deep learning method that is commonly used for time series forecasting and predictions is Long Short-Term Memory (LSTM). This method has become particularly useful in stock price predictions in recent years due to its ability to capture and model the uncertainty of stock prices both in the short and the long term.

Problem statement

This project aims to build a stock prediction model by using stock price data and technical indicators thereof to predict future values of stock prices. For this project an API called *yfinance* is used to extract the stock price data since 2010 to the current date. Each API call extracts the open price, highest price, lowest price, volume of stocks traded and adjusted closing price for a prescribed date period and for a particular stock. The data received from the API is very clean and well structured. In stock trading technical indicators are often used to predict a long term or short term trend. One such technical indicator is the concept of moving averages. A moving average of 5 days, 50 days and 200 days is used in this project to show movement of momentum in stock prices.

More specifically, the opening price, the highest daily trading price, the closing price, the volume of stocks traded as well as the adjusted closing price of a stock is used to predict the future adjusted closing price. The results of this model need to be presented in the form of a functioning web application. The web application should include a page to analyse the stock price data of a selected stock, a training interface page to train a new model with preselected parameters and a page to display the results of an existing pre-trained model. This project is limited to the analysis of three stocks only; Microsoft, S&P500 and Nasdaq. This project will thus explore the implementation of an LSTM model to predict the future prices of stocks using Keras.

Metrics

Since this project aims to predict the numeric value of the adjusted closing price of a stock the metric selected for determining the performance of the model is the root mean squares error (RMSE). The root mean square error is the standard deviation of the prediction error, in other words the standard deviation of the difference between the predicted values and the actual values.

$$RMSE = \sqrt{\frac{1}{n} \sum_{j=1}^n (y_j - \hat{y}_j)^2}$$

Analysis

Data exploration

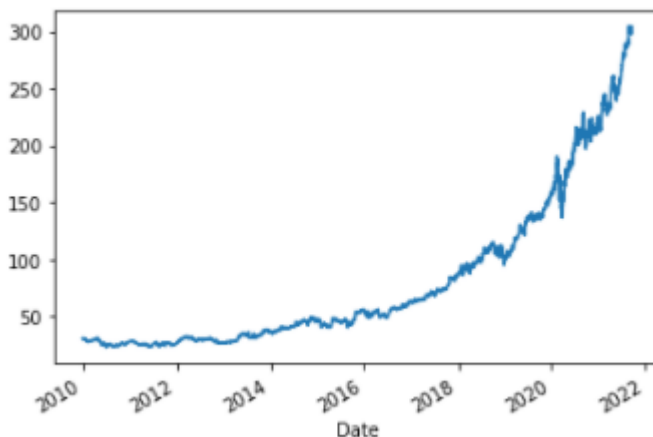
The data is received using an API call to *yfinance* and returned as shown in the following image:

	Open	High	Low	Close	Volume	AdjClose
Date						
2009-12-31	30.980000	30.990000	30.480000	30.480000	31929700	23.630186
2010-01-04	30.620001	31.100000	30.590000	30.950001	38409100	23.994564
2010-01-05	30.850000	31.100000	30.639999	30.959999	49749600	24.002319

The data received by the API is clean and already converted into the correct types and thus no further transformations are needed to make the data available to the pre-processing phase.

Data visualization

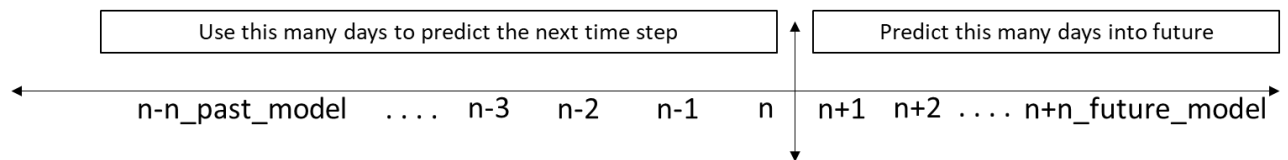
As an example of how the data looks see below the opening price of Microsoft plotted from 2010 to 2021.



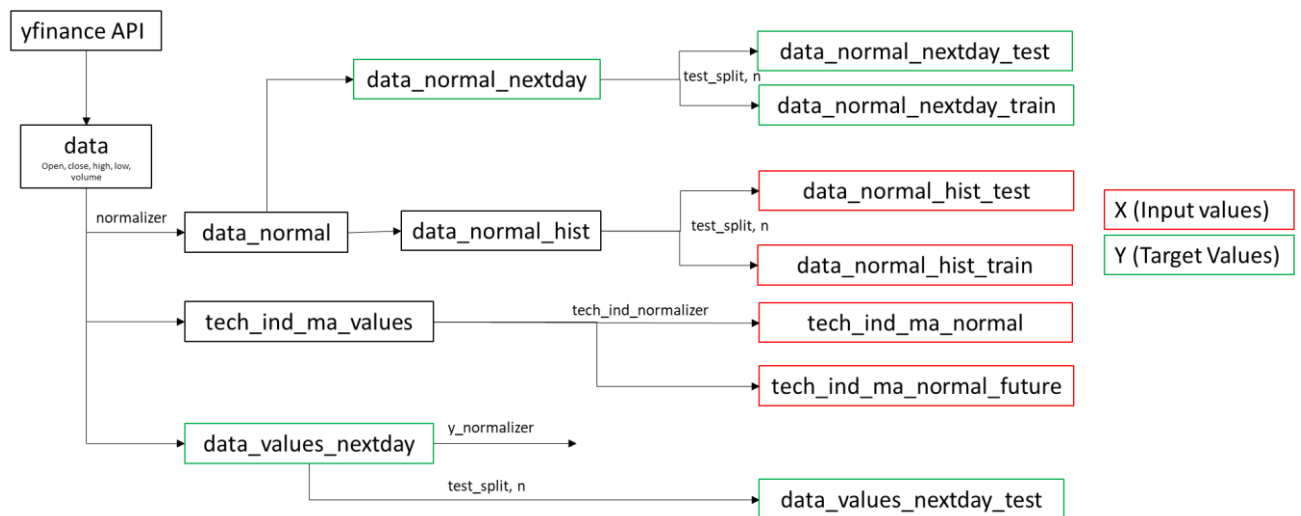
Methodology

Data Pre-processing

The data is received from an API call to *yfinance*. Before this data can be used in a machine learning model it needs to be scaled and since this is time series data it needs to be windowed. The data scaling was done using Sklearn's pre-processing module. The windowing of the data be best described using the below image. As shown in the image, a certain number of days is used to predict the next days stock price. As an example: use the last 50days of stock prices to predict the stock price for tomorrow. This way the prediction for tomorrow is using the behaviour of the stock price from the past 50days to predict a more accurate stock price for 1 day into the future.

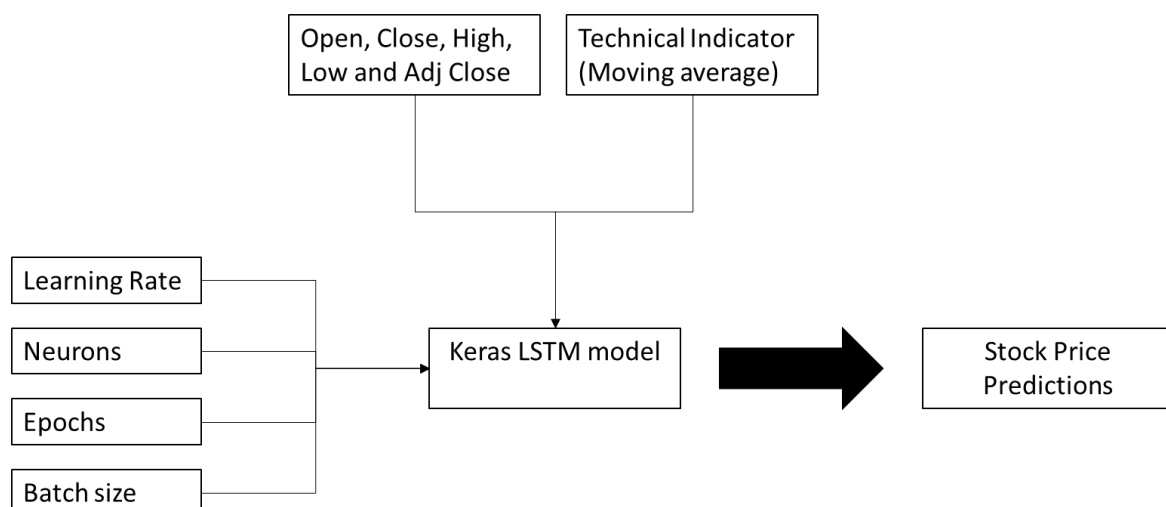


Further pre-processing is done to divide the data into inputs and targets for the machine learning model. In the image below it is shown how the data from the API call gets divided into the different variables to form the features of the machine learning model. All variables indicated in red are values used as inputs and all variables indicated in green are values used as targets for the machine learning model. Additionally, this schematic also indicates how the data is divided into test and train sets and shows how a technical indicator is incorporated into the model (tech_ind_ma_normal and tech_ind_ma_future)



Implementation

The model is created using Keras and generally takes two inputs: the moving average technical indicator and the opening price, closing price, highest price, lowest price, and adjusted closing price of the stock over the period defined. Additionally, the model utilizes 4 different parameters to be considered as these parameters affect the accuracy of predictions.



Refinement

Refinement of the model was done during the implementation of the model. At first a model was created using only the open price, closing price, highest price, lowest price, and adjusted closing price but it was found that prediction accuracy was not sufficient. This then led to the introduction of the moving average technical indicator to be included in the model to improve accuracy.

Results

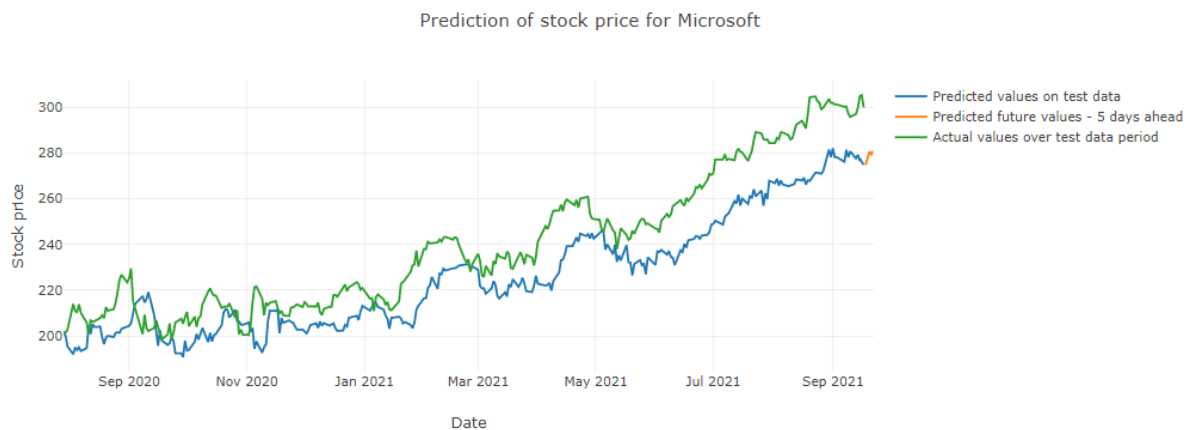
Model evaluation and validation

To determine the optimal values for the different input parameters to the Keras LSTM model cross validation was used. This process entailed keeping all other parameters constant while varying one specific parameter. For example: a constant value for the number of neurons and batch size while the number of epochs was varied. For each of these scenarios the mean squared error is calculated to be able to compare the different combinations to one another. The table below summarizes the results of the cross validation and highlights the parameters which resulted in the best mean square error. It shows that the best value for batch size is 5, number of epochs is 150 and number of neurons is 40.

batch_size	mse_score (average)	no_epochs	mse_score (average)	neuron_num	mse_score (average)
5	214	25	3168	5	8464
10	414	50	2387	10	5352
15	697	75	2271	15	3191
20	1697	100	1037	20	1929
25	1610	125	975	25	1043
30	2855	150	508	30	500
35	2344			35	792
40	2876			40	358
				45	408

Justification

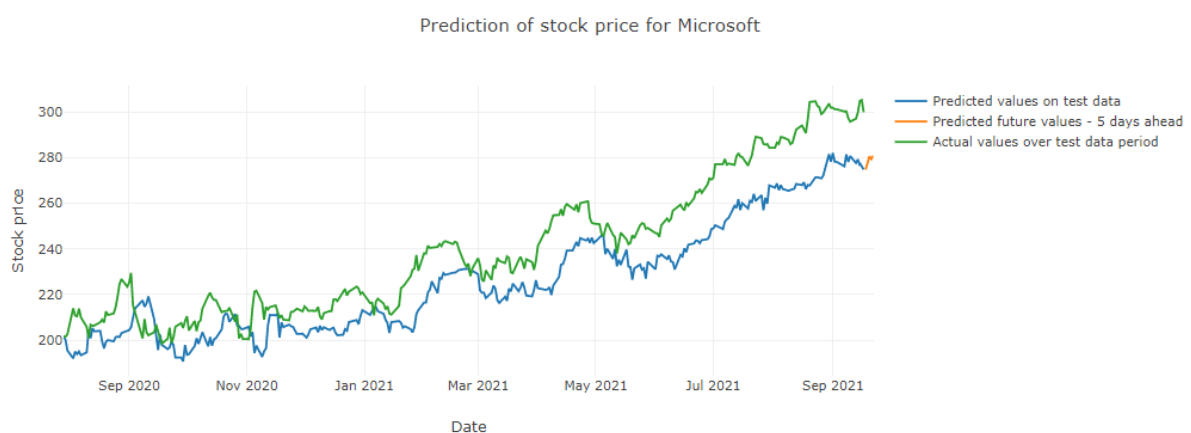
To test the model, the data set was split into a training and testing set using an 80% split. This means that 20% of the data is unseen by the model when testing the model. The testing data set was used to test the model performance on unseen data and the results of the stock Microsoft is shown below. The line indicated in blue are the predicted values and the green line indicated the actual stock price values. The orange line is the stock price predicted 5 days into the future.



Conclusion

Reflection

The web application functions as it should; providing a user with the option to analyse the stock price with various moving averages indicated on the same axis, view the results of an existing model that was pre-trained to provide a visualization of the model performance and then finally a page that allows a user to train their own model with a certain set of their own parameters. The model performance, as measured by the mean square error, is not sufficient to be able to accurately predict the future stock prices with a good level of certainty. On average, the mean square error of the model was $\pm 200\%$. This means that a true value differs from a predicted value by as much as 200%. It should however be noted that when the model results are plotted vs. the actual results the predicted values follow the correct trend overall, but it seems like there is a relatively constant offset between actual and predicted values. This is shown in the plot below.



Improvement

As mentioned in the reflection section, the model prediction performance is not as good as one would need to invest money in the stock market based on the models' predictions. The following can be done to improve the prediction model:

1. Incorporate more features into the input of the LSTM model. Moving average is one of many technical indicators that investment analysts use to determine trends in the stock market. To improve the performance of the model additional technical indicators can be used as additional features.
2. The web application can be improved by incorporating sentiment analysis on stock news for the three different stocks analysed in this project. Combining sentiment analysis of stocks with stock price predictions will provide a more reliable indicator of when to invest in a particular stock.