Predict Stock Prices using Deep Learning

Machine Learning Engineer Nanodegree

Capstone Project Proposal

Definition

In this MLND capstone project, I will be attempting to predict future stock prices for selective stock symbols. I will take historical stock price data for specific Stock symbols from yahoo finance and use deep learning techniques to train a model. This model will then be used to predict prices for next couple of days.

As a reference, I will be using Machine Learning techniques taught in Udacity's Machine Learning NanoDegree and Machine Learning for Trading course.

Python language and Keras[6] library will be used for application development.

Domain Background

Predicting stock prices has long been a subject of interest for traders, mathematicians, statisticians and lately technologists. This has generated a lot of interest as one will be able to reap enormous profits if one is able to predict stock prices and stock market movements reflect, to some extent, the economic state of the country also. This has also been the field of research and interest as theoretical background of financial theory does not support stock market prediction.

Theoretically, there are two major hypotheses rejecting the predictability of future stock prices.

1. **Efficient Market Hypothesis**
2. **Random Walk Theory**

Efficient Market Hypothesis[1]

The efficient market hypothesis is an investment theory, proposed by Eugene Fama, which suggests that stock prices reflect all currently available information and so it is not possible to predict stock prices. This means that stock prices adjust quickly to any recently released information and past stock price history (which is publicly available) is also already reflected in the stock prices making it impossible to predict future prices. There are three versions of the efficient market hypothesis (EMH), all based on varying assumptions of price efficiency. The **weak** form of EMH claims that the prices of publicly-traded assets already reflect all available information, and past prices are of little value in predicting future trends. The **semi-strong** version of EMH holds that while prices are efficient, they react instantaneously to new information, while the **strong** version of EMH maintains that asset prices reflect not just public knowledge, but private insider information as well.

Random Walk Theory[2]

The **random walk theory hypothesis** is a financial theory which states that stock market prices evolve as per the random walk and thus cannot be predicted. The Random Walk Theory term was popularized by the

1973 book, *A Random Walk Down Wall Street*, by Burton Malkiel, and this was used earlier in Eugene Fama's 1965 article "Random Walks In Stock Market Prices".

There are both proponents and opponents to these hypotheses.

> Proponents believe that it is pointless to search for undervalued stocks or to try to predict trends in the market through either fundamental or technical analysis. They conclude that, because of the randomness of the market, investors could do better by investing in a low-cost, passive portfolio. A study done by Morningstar Inc found out that 25% of the top-performing active managers are able to consistently outperform their passive manager counterparts.
>
> Opponents believe that the market is predictable to some degree. Martin Weber, a leading researcher in behavioral finance, has performed many tests and studies on finding trends in the stock market. In one of his key studies, he found that stocks with high price increases in the first five years tended to become under-performers in the following five years. He cited this as a key contributor and contradictor to the random walk hypothesis. Other opponents give examples of investors such as Warren Buffett who has consistently beaten the market over long periods of time. Some give reference to events such as the 1987 stock market crash, when the Dow Jones Industrial Average (DJIA) fell by over 20% in a single day, as evidence that stock prices can seriously deviate from their fair values.

Prediction methodologies fall into three broad categories - Fundamental analysis, Technical analysis and Technological methods.

Fundamental analysis
In FA, basis of analysis is company's past performance. Fundamental analysis is built on the belief that human society needs capital to make progress and if a company operates well, it should be rewarded with additional capital which results in a surge in stock price. Fundamental analysis is widely used by fund managers as it is the most reasonable, objective and made from publicly available information like financial statement analysis. Following indicators related to past performance are used for FA.

Technical analysis
Technical analysis is not based on company's fundamentals. In TA, the future price of a stock is calculated solely on the basis of the (potential) trends of the past price (a form of time series analysis). Numerous patterns are employed such as the head and shoulders or cup and saucer. Alongside the patterns, statistical techniques are used such as the exponential moving average (EMA). Candle stick patterns are believed to be first developed by Japanese rice merchants, and nowadays widely used by technical analysts.

Technological methods

With the advent of the digital computer and recent developments in very high computing capacity, stock market prediction has evolved to a great extent. The algorithmic trading techniques such as Artificial Neural Networks (ANNs) and Genetic Algorithms are used to predict stock prices.

Personal motivation

Finance field has quite intrigued me since last couple of years. Seeing conflicting views about predicting stock prices, I myself wanted to check whether it is really possible to predict prices. During my MLND, I saw quite interesting implementations of Machine Learning techniques and wanted to utilize Deep Learning techniques to predict stock prices. Further motivation was to be able to apply the algorithm in real time to see if it will really result in stock market profits.

Problem Statement

The problem to be solved in this project is to predict the future stock prices for next couple (5, 10, 20) of days.

The dataset of the selected ticker (particular stock symbol) is divided into training and test datasets. After developing the model using training dataset, stock price will be predicted for the period starting where the training dataset period ends. The predicted values can be compared against the testing dataset values for evaluation. The closeness of predicted values to actual values can be measured using Kera's evaluate method or using RMSE method.

Datasets and Inputs

Data for the project is taken from Yahoo finance website. Historical data of following stock tickers is taken – S&P500, AAPL, GOOG, MSFT, XOM etc.

The historical data will be collected for following events – Open, Close, High, Low, Adj Close, Volume. Here, we will use only Adj Close[5] price of the stocks. Adj Close price is similar to daily close price, only difference is that it takes care of past splits, reverse splits, dividends, rights offerings also.

Final dataset will contain Adj Close column with Date as the index. Dataset will be cleaned for any "null" values and it will be made sure that Adj Close data only have float values. Pandas dataframe dropna function will be used to drop any rows containing undefined data.

After data cleanup, it will be normalized so that its values are scaled between -1 and 1. It will help in convergence of machine learning algorithm and it will also help in bringing different stocks to the same scale.

Solution Statement

In this project, attempt will be made to predict the prices using Deep Learning techniques of Recurrent Neural Networks. Recurrent Neural Network model LSTM (Long short term Memory) will be used to predict the stock prices.

Benchmark Model

Multiple benchmark models will be used for comparison:

1) As we are using existing data and we have test data available for which actual values are already available. We can test our model generated predicted values with these actual values and can see how accurate our model is.
2) Online website - https://www.stock-forecasting.com/Content/Data/Test.aspx

Evaluation Metrics

Predicted prices will be compared to Actual historical prices.

- Keras model.evaluate method will be used to find the training and test errors.
- RMSE (Root mean Square Error) will be calculated for predicted and actual values.

Project Design

Project will be implemented using programming language Python and neural network API Keras will be used for deep learning.

Input Data

- Data will be downloaded from finance.yahoo.com[8] for selected tickers (S&P500, APPL. GOOG, MSFT, XOM etc.).
- Data will be read using pandas dataframe.
- Data will be indexed using date value.
- Only Adj Close share price will be used for analysis.
- Data will be cleaned so that is does not have any null and NA(undefined) values.
- For training and testing purpose, data will be split into training and test datasets.

Ref code:

Reading data

```
import pandas as pd
data_XOM = pd.read_csv(XOM_file_path,index_col ='Date')
data_XOM.dropna()
```

sklearn minmaxscaler can be used for normalizing data

```
from sklearn.preprocessing import MinMaxScaler
minmaxScaler = MinMaxScaler((-1,1))
```

train\test split

```
train_test_split = int(np.ceil(2*len(y)/float(3)))   # set the split point

# partition the training set
X_train = X[:train_test_split,:]
y_train = y[:train_test_split]

# keep the last chunk for testing
X_test = X[train_test_split:,:]
y_test = y[train_test_split:]
```

Data Modelling

- Keras API will be used for model training and prediction testing.
- Keras sequential model[7] will used for training.
- LSTM layer will be added to this sequential model.
- Depending upon model performance, model LSTM layers will be added.
- Dropout layer will be added(if required) to handle data overfitting. Overfitting happens when training error keeps on decreasing but testing error starts rising after reaching bottom.
- Multiple Dropout layer will be added after each LSTM layer. (if required)
- Final linear layer with single output will be added at the end of model.
- Different Model Optimizers[7] (provided by Keras API) will be used to optimize the model.
- Model will be compiled with appropriate parameters values.

- Model will fit the training data – different parameter values for the fit function[7] will be provided to better fit the model.

  Ref code :

```
model = Sequential()
model.add(LSTM(50,input_shape = (window_size,1)))
model.add(Dense(1))

optimizer = keras.optimizers.RMSprop(lr=0.001, rho=0.9, epsilon=1e-08, decay=0.0)
model.compile(loss='mean_squared_error', optimizer=optimizer)
model.fit(X_train, y_train, epochs=10, batch_size=50, verbose=1)
```

Prediction

- After model fitting, prediction will be made for training and testing data to calculate training and testing errors. Keras Predict[7] and evaluate functions will be used to predict and evaluate[7].
- Actual values of testing dataset will be used to check the accuracy of Predicted values.

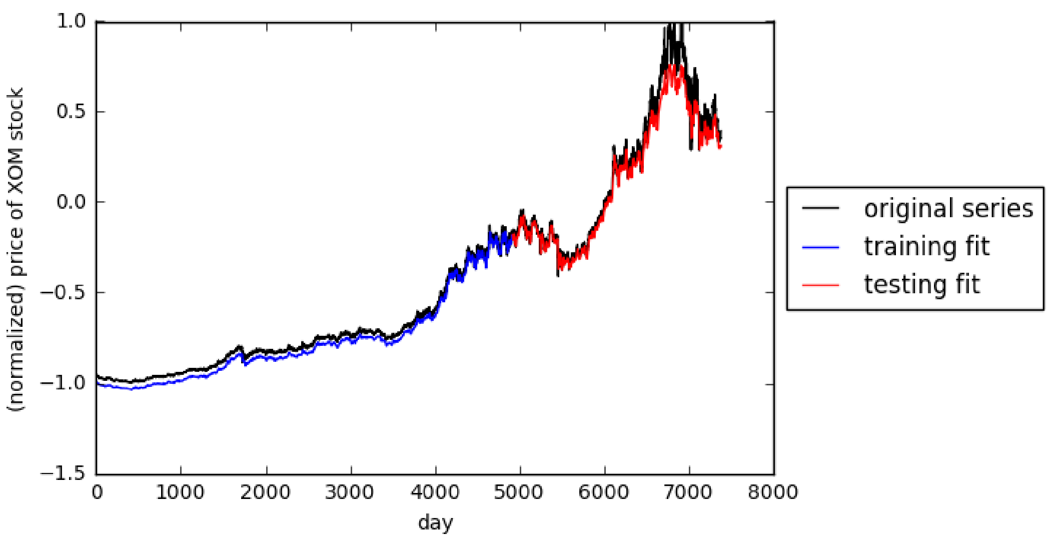
  Ref code:

```
train_predict = model.predict(X_train)

test_predict = model.predict(X_test)

training_error = model.evaluate(X_train, y_train, verbose=0)

testing_error = model.evaluate(X_test, y_test, verbose=0)
```

Visualization

Actual training and testing data will be plotted alongside the predicted data using matplotlib library functions. Visualization should be able to show the closeness of predicted values to actual values.

Fig below shows data for stock ticker-XOM. It plots Adj Close price for about 7300 data points. Blue part shows the training data. Red part shows the testing data. Black data shows the actual data. Closeness of Blue\Red points to the black point lines will show the degree of fit.



Appendix

Artificial Neural Networks[3]

Artificial neural networks are computing systems inspired by the biological neural networks constituting human brains. An ANN is a collection of connected units (logical units) called artificial neurons. In a neural network, thousands or millions of neurons, are organized in layers. Each layer may perform different kinds of transformations on their inputs. ANN learn by forward propagation (matching inputs to outputs) and backpropagation (matching outputs to inputs).

RNN (Recurrent Neural Network)[4]

A **recurrent neural network** (**RNN**) is a class of artificial neural network in which neurons have directed cycle connections. Unlike feedforward neural networks(where neurons are connected in only one direction from input to output), RNNs have backward connections also. RNNs can use their internal memory to process arbitrary sequences of inputs.

LSTM (Long Short Term Memory) Network[4]

Long short-term memory (LSTM) is specific type of RNN that attempts to remember and forget previously identified patterns. LSTMs have "forget" gates to discard unwanted patterns. LSTMs try to remove the vanishing or exploding gradient problems and this helps LSTM learn tasks that require memories of events that happened thousands or even millions of discrete time steps earlier.

References

1) https://en.wikipedia.org/wiki/Stock_market_prediction
2) https://en.wikipedia.org/wiki/Random_walk_hypothesis
3) https://en.wikipedia.org/wiki/Artificial_neural_network
4) https://en.wikipedia.org/wiki/Recurrent_neural_network
5) http://www.investopedia.com/terms/a/adjusted_closing_price.asp
6) https://keras.io/
7) https://keras.io/models/model/
8) https://finance.yahoo.com/