FUNDAMENTALS OF MANAGEMENT MG-301

STOCK MARKET PREDICTION USING MACHINE LEARNING

INNOVATIVE MID-TERM PROJECT

BY **ADITYA SINGH** 2K19/EP/005

AND **ARKAJYOTI CHAKRABORTY** 2K19/EP/022

TABLE OF CONTENTS

DEFINITION

The stock market is known for being volatile, dynamic, and nonlinear. Accurate stock price prediction is extremely challenging because of multiple (macro and micro) factors, such as politics, global economic conditions, unexpected events, a company's financial performance, and so on.

But, all of this also means that there's a lot of data to find patterns in. So, financial analysts, researchers, and data scientists keep exploring analytics techniques to detect stock market trends. This gave rise to the concept of algorithmic trading, which uses automated, pre-programmed trading strategies to execute orders.

The ability to successfully and consistently predict the stock market is, obviously, a gold mine which technologists have been working towards for many years. Thanks to recent rapid developments in deep learning algorithms, more individuals and companies are able to rely on stock market forecasting from artificial intelligence, as the technology has begun to predict better than the pros.

Project Overview

This project seeks to utilize Deep Learning models, Long-Short Term Memory (LSTM) Neural Network algorithm, to predict stock prices. For data with time frames recurrent neural networks (RNNs) come in handy but recent research has shown that LSTM networks are the most popular and useful variants of RNNs. I will use Keras to build a LSTM to predict stock

prices using historical closing price and trading volume and visualize both the predicted price values over time and the optimal parameters for the model.

Problem Statement

The challenge of this project is to accurately predict the future closing value of a given stock across a given period of time in the future. For this project I will use a Long Short Term Memory network – usually just called “LSTMs” to predict the closing price of the 1 S&P 500 using a dataset of past prices.

GOALS

1. Explore stock prices.
2. Implement a basic model using linear regression.
3. Implement LSTM using keras library.
4. Compare the results and submit the report.

Metrics

For this project measure of performance will be using the Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) calculated as the difference between predicted and actual values of the target stock at adjusted close price and the delta between the performance of the benchmark model (Linear Regression) and our primary model (Deep Learning).

ANALYSIS

Data Exploration

The data is presented in a couple of formats to suit different individual's needs or computational limitations. We have included files containing 5 years of stock data (in the all_stocks_5yr.csv).
The all_stocks_5yr.csv contains the same data, presented in a merged .csv file.

All the files have the following columns: Date - in format: yy-mm-dd

Open - price of the stock at market open
High - Highest price reached in the day
Low Close - Lowest price reached in the day
Volume - Number of shares traded
Name - the stock's ticker name

Date	Open	High	Low	Close	Volume
30-Jun-17	943.99	945.00	929.61	929.68	2287662
29-Jun-17	951.35	951.66	929.60	937.82	3206674
28-Jun-17	950.66	963.24	936.16	961.01	2745568

What matters is the opening price of the stock and closing prices of the stock. If at the end of the day we have higher closing prices than the opening prices that we have some profit otherwise we saw losses. Also volume of share is important as a rising market should see rising volume, i.e, increasing price and decreasing volume show lack of interest, and this is a warning of a potential reversal. A price drop (or rise) on large volume is a stronger signal that something in the stock has fundamentally changed.

Visualization

To visualize the data We have used matplotlib library. We plotted the Closing stock price of 7 the data with the no of items(no of days) available. Following is the snapshot of the plotted data :

Through this data we can see a continuous growth in Infosys. The major fall in the prices might be because of the Global Financial Crisis of 2008-2009.

Algorithms & Technique

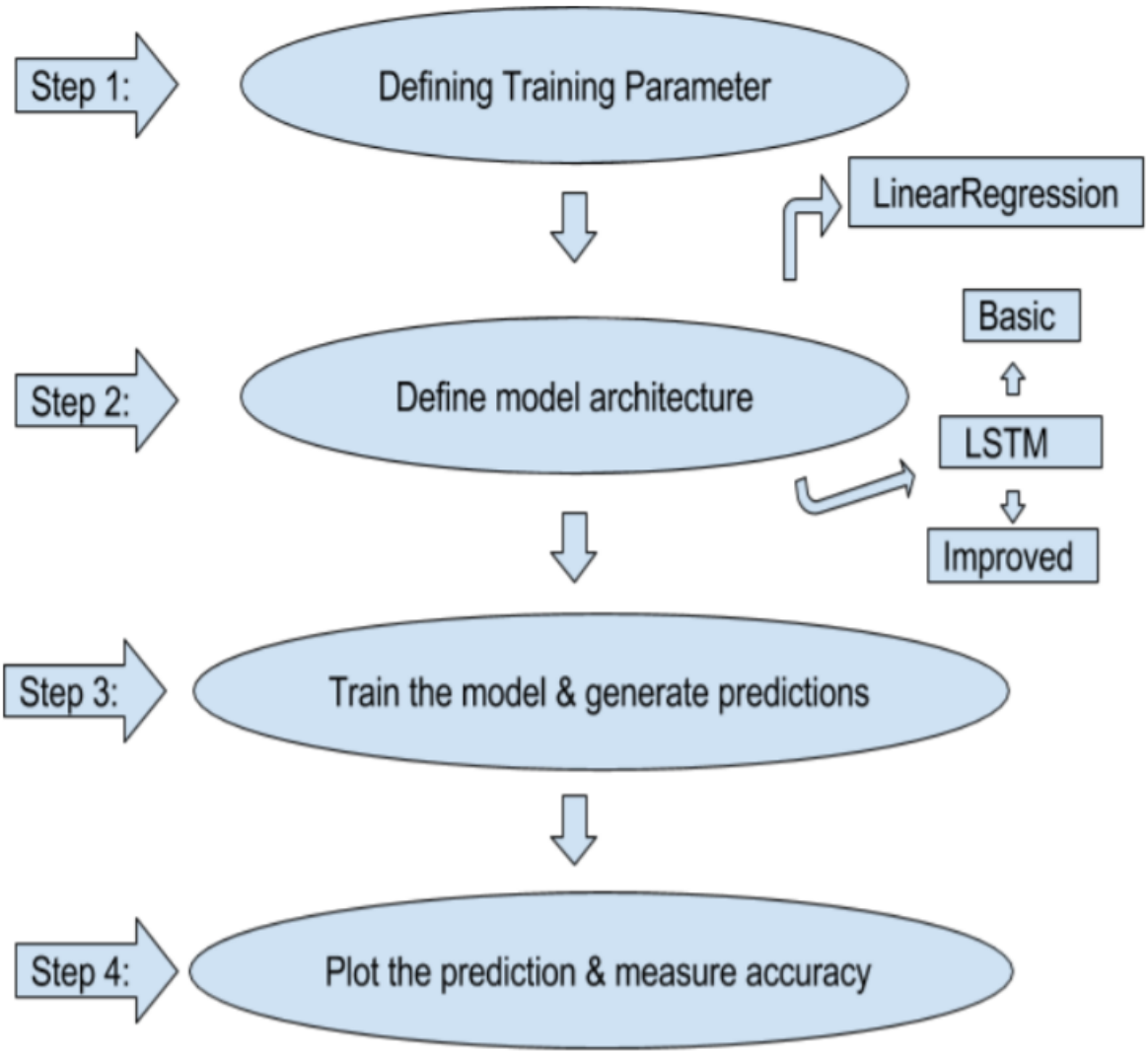
The goal of this project was to study time-series data and explore as many options as possible to accurately predict the Stock Price. We came to know about Recurrent Neural Nets (RNN) which are used specifically for sequence and pattern 8 learning. As they are networks with loops in them, allowing information to persist and thus ability to memorise the data accurately. But Recurrent Neural Nets have a vanishing Gradient descent problem which does not allow it to learn from past data as was expected. The remedy of this problem was solved in Long-Short Term Memory Networks , usually referred to as LSTMs. These are a special kind of RNN, capable of 9 learning long-term dependencies.

.

For this project We have used a Linear Regression model as its primary benchmark. This Linear Regression was used for error rate comparison MSE and RMSE utilizing the same dataset as the deep learning models.

IMPLEMENTATION

Once the data has been downloaded and preprocessed, the implementation process occurs consistently through all three models as follow:



Benchmark model :

Step 1 : Split into train and test model :

```
X_train, X_test, y_train, y_test, label_range= sd.train_test_split_linear_regression(stocks)
```

```
def train_test_split_linear_regression(stocks):

    feature = []
    label = []

    for index, row in stocks.iterrows():
        feature.append([(row['Item'])])
        label.append([(row['Close'])])

    feature_bounds = [min(feature), max(feature)]
    feature_bounds = [feature_bounds[0][0], feature_bounds[1][0]]
    label_bounds = [min(label), max(label)]
    label_bounds = [label_bounds[0][0], label_bounds[1][0]]

    feature_scaled, feature_range = scale_range(np.array(feature),
    label_scaled, label_range = scale_range(np.array(label), input_

    split = .315
    split = int(math.floor(len(stocks['Item']) * split))

    X_train = feature_scaled[:-split]
    X_test = feature_scaled[-split:]

    y_train = label_scaled[:-split]
    y_test = label_scaled[-split:]

    return X_train, X_test, y_train, y_test, label_range
```

Step 2: In this step model is built using scikit-learn linear_model library.

```
model = LinearRegressionModel.build_model(X_train,y_train)
```

```
def build_model(X, y):

    linear_mod = linear_model.LinearRegression()
    X = np.reshape(X, (X.shape[0], 1))
    y = np.reshape(y, (y.shape[0], 1))
    linear_mod.fit(X, y)

    return linear_mod
```

Step 3: Now it's time to predict the prices for given test datasets.

```
predictions = LinearRegressionModel.predict_prices(model,X_test, label_range)
```

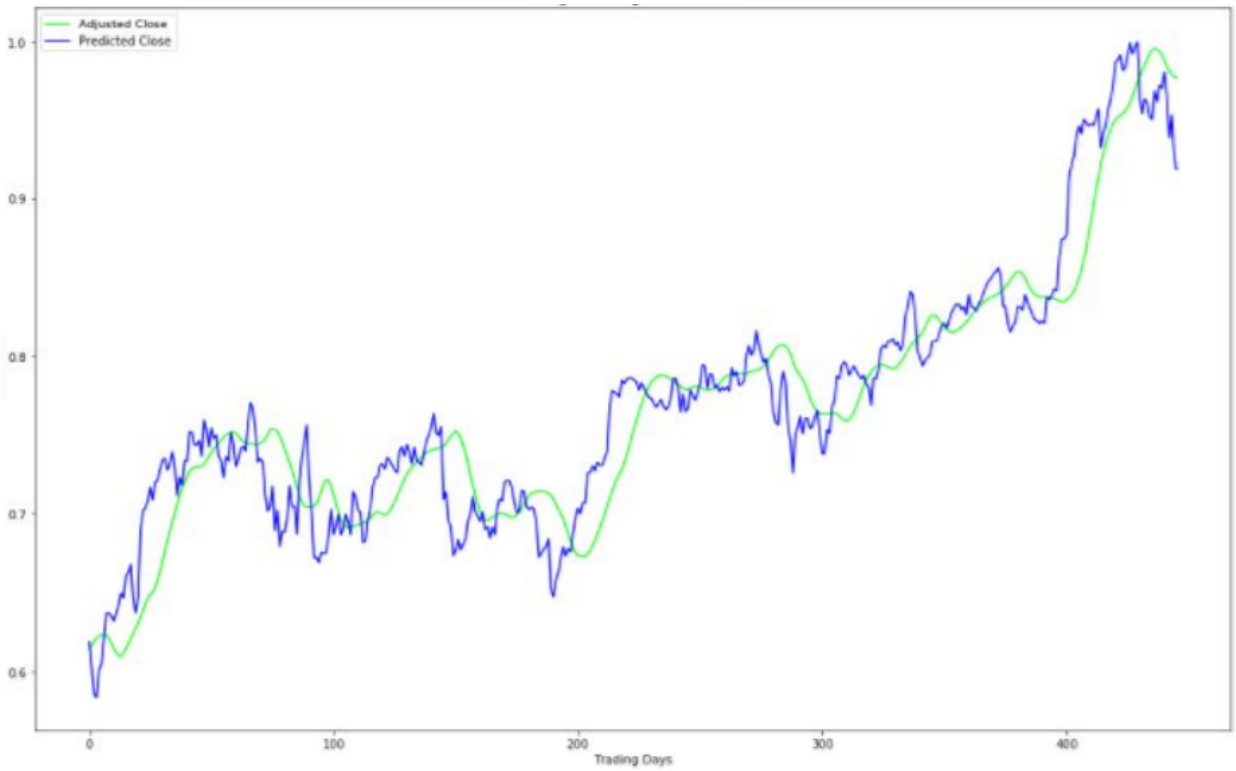
```
def predict_prices(model, x, label_range):

    x = np.reshape(x, (x.shape[0], 1))
    predicted_price = model.predict(x)
    predictions_rescaled, re_range =
        sd.scale_range(predicted_price,
            input_range=[-1.0, 1.0], target_range=label_range)

    return predictions_rescaled.flatten()
```

Refinement

For this project We worked on fine tuning parameters of LSTM to get better predictions. We did the improvement by testing and analysing each parameter and then selecting the final value for each of them. To improve LSTM i have done following:

- Increased the number of hidden nodes from 100 to 128.
- Added Dropout of 0.2 at each layer of LSTM
- Increased batch size from 1 to 512

- Increased epochs from 1 to 20
- Added verbose = 2
- Making predictions with the batch size Thus improved my mean squared error, for testing sets, from 0.01153170 MSE to 0.00093063 MSE.



Conclusion

To recap, the process undertaken in this project:

- Set Up Infrastructure
    - iPython Notebook
    - Incorporate required Libraries (Keras, Tensor flow, Pandas, Matplotlib, Sklearn, Numpy)
    - Git project organization
- Prepare Dataset

    - Incorporate data of Alphabet Inc company
    - Process the requested data into Pandas Dataframe
    - Develop function for normalizing data
    - Dataset used with a 80/20 split on training and test data across all models
- Develop Benchmark Model
    - Set up basic Linear Regression model with Scikit-Learn
    - Calibrate parameters
- Develop Basic LSTM Model
    - Set up basic LSTM model with Keras utilizing parameters from Benchmark Model
- Improve LSTM Model
    - Develop, document, and compare results using additional labels for the LSMT model 5. Document and Visualize Results
- Plot Actual, Benchmark Predicted Values, and LSTM Predicted Values per time series
- Analyze and describe results for the report.

We used a completely new algorithm, i.e, Long-Short Term Memory and also to explore real time series data sets. The final model really exceeded the expectations and has worked remarkably well. The major problem We faced during the implementation of the project was exploring the data. It was the toughest task. To convert data from raw format to preprocess data and then to split them into training and test data. All of these steps require a great deal of patience and a very precise approach. Also to work around a lot to successfully use the data for 2 models, i.e, Linear Regression and Long-Short Term Memory, as both of them have different input sizes.

END