A Project Work on

“STOCK MARKET PREDICTION”

under the guidance of

**Mr. Subhojit Mondal**

**Mr. Shubham Bagade**

**Abstract :**

Predicting stock market prices is a complex task that traditionally involves extensive human-computer interaction. Due to the correlated nature of stock prices, conventional batch processing methods cannot be utilized efficiently for stock market analysis.

We use an learning algorithm that utilizes a kind of *recurrent neural network (RNN)* called *Long Short Term Memory (LSTM)*, where the weights are adjusted for individual data points using stochastic gradient descent. This will provide more accurate results when compared to existing stock price prediction algorithms. The network is trained and evaluated for accuracy with various sizes of data, and the results are tabulated.

**Key Words:** *stock prediction, Long Short Term Memory, Recurrent Neural Network, online learning, stochastic gradient descent.*

**Introduction :**

The stock market is a vast array of investors and traders who buy and sell stock, pushing the price up or down. The prices of stocks are governed by the principles of demand and supply, and the ultimate goal of buying shares is to make money by buying stocks in companies whose perceived value (i.e., share price) is expected to rise. Stock markets are closely linked with the world of economics —the rise and fall of share prices can be traced back to some Key Performance Indicators (KPI's).

The five most commonly used KPI's are the –

* opening stock price (`Open')
* end-of-day price (`Close')
* intraday low price (`Low')
* intra-day peak price (`High')
* total volume of stocks traded during the day (`Volume')

Economics and stock prices are mainly reliant upon subjective perceptions about the stock market. It is near impossible to predict stock prices to the T, owing to the volatility of factors that play a major role in the movement of prices. However, it is possible to make an educated estimate of prices.

Stock prices never vary in isolation: the movement of one tends to have an avalanche effect on several other stocks as well. This aspect of stock price movement can be used as an important tool to predict the prices of many stocks at once.

1. **Info Gathering :**

The primary requisites in any Machine learning projects are datasets, there are two datasets that we utilized in this project: Stock data of four companies – Apple, Facebook, Tesla & Microsoft and individual data of Microsoft.

The above-mentioned datasets are downloaded from the following links:

**Stock dataset** – https://raw.githubusercontent.com/pierpaolo28/Data-Visualization/master/Dash/stock\_data.csv

**MSFT dataset** –

<https://github.com/matplotlib/sample_data/blob/master/msft.csv>

1. **Planning :**

**TASK** –

In this machine learning project, we will be talking about predicting the returns on stocks. We will analyse daily stocks status and predict for the next day.

**EXPLORATORY DATA ANALYSIS & FEATURE EXTRACTION–**

There are two datasets that we utilized in this project: Stock data and MSFT data.

Following are the steps:

a. Data Collection.

b. Data Preparation.

c. Data Visualization.

d. Feature Engineering.

e. Model Creation.

f. Dashboard Creation

**MODELLING –**

We have used Supervised learning.

Model Methods:

Attempted to predict the average next day’s closing price with four different regression models:

• Linear regression, Decision Tree regression, Support vector regression, and Random forest regression.

• We utilized Scikit-Learn to implement these four models.

**IMPLEMENTATION –**

We attempted to predict the average next day prices of the stocks with four different regression models; linear regression, support vector regression, decision tree regressor and random forest regression. We utilized Scikit-Learn to implement these four models.

1. **Linear Regression:**

Linear regression works well as an initial baseline measurement. Given an input vector x ∈ RK+1, where x contains the multiple days proportion of the topics and current day closing price, linear regression finds a θ such that θ T x is the predicted next day’s closing price. To find the θ, we choose the θ that minimizes the least-square cost function

J() =2

where ;

n is the number of training examples

y (i) is the true label for examples i,

x (i) is the data for example i.

1. **Support Vector Regression**:

Support vector regression is another good model that might be able to capture non-linear trends in the data with certain kernels. We aim to find a w such that w T x + b is a prediction for the next day’s closing price. This w is found by minimizing:

)

where n is the number of training examples.

1. **Decision Tree Regression:**

Decision tree regression is also another approach that can capture non-linear trends in a data set. However, because decision trees are prone to high variance that is dependent on random splits, random forests are often used as they aggregate the results from several decision trees. To train our model, we utilized a 100 decision trees with the splitting criterion

Where;

y (L) is the average currency exchange rate in the left node.

y (R) is the average currency exchange rate in the right node.

Hyper parameters that were turned were minimum leaf samples and minimum samples for a split to reduce over-fitting.

1. **Random Forest Regression :**
2. The ramdom forest means data about data estimator. It fits
3. a number decision tress on various sub samples of the given
4. data. It control over-fitting. It improve the predictive
5. accuracy.
6. Algorithm:
7. Step 1: From the dataset pick N random records.
8. Step 2: Based on N records, build a decision tree.
9. Step 3a: From your algorithm, choose the number of trees
10. and repeat steps 1 and 2.
11. Step 3b: In case of a regression problem, for a new record,
12. each tree in the forest predicts a value for Y (output).
13. The ramdom forest means data about data estimator. It fits
14. a number decision tress on various sub samples of the given
15. data. It control over-fitting. It improve the predictive
16. accuracy.
17. Algorithm:
18. Step 1: From the dataset pick N random records.
19. Step 2: Based on N records, build a decision tree.
20. Step 3a: From your algorithm, choose the number of trees
21. and repeat steps 1 and 2.
22. Step 3b: In case of a regression problem, for a new record,
23. each tree in the forest predicts a value for Y (output).
24. The ramdom forest means data about data estimator. It fits
25. a number decision tress on various sub samples of the given
26. data. It control over-fitting. It improve the predictive
27. accuracy.
28. Algorithm:
29. Step 1: From the dataset pick N random records.
30. Step 2: Based on N records, build a decision tree.
31. Step 3a: From your algorithm, choose the number of trees
32. and repeat steps 1 and 2.
33. Step 3b: In case of a regression problem, for a new record,
34. each tree in the forest predicts a value for Y (output).

The random forest means data about data estimator. It fits a number decision tress on various sub samples of the given data. It control over-fitting. It improve the predictive accuracy.

Algorithm:

Step 1: From the dataset pick N random records.

Step 2: Based on N records, build a decision tree.

Step 3a: From our algorithm, choose the number of trees and repeat steps 1 and 2.

Step 3b: In case of a regression problem, for a new record, each tree in the forest predicts a value for Y (output).

The ramdom forest means data about data estimator. It fits

a number decision tress on various sub samples of the given

data. It control over-fitting. It improve the predictive

accuracy.

Algorithm:

Step 1: From the dataset pick N random records.

Step 2: Based on N records, build a decision tree.

Step 3a: From your algorithm, choose the number of trees

and repeat steps 1 and 2.

Step 3b: In case of a regression problem, for a new record,

each tree in the forest predicts a value for Y (output).

1. **PROPOSED SYSTEM :**

We propose an learning algorithm for predicting the end-of-day price of a given stock with the help of Long Short Term Memory (LSTM), a type of Recurrent Neural Network (RNN).

* **LSTM – an overview**

LSTM's are a special subset of RNN’s that can capture context-specific temporal dependencies for long periods of time. Each LSTM neuron is a memory cell that can store other information i.e., it maintains its own cell state. While neurons in normal RNN’s merely take in their previous hidden state and the current input to output a new hidden state, an LSTM neuron also takes in its old cell state and outputs its new cell state.

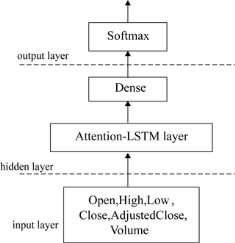
An LSTM memory cell has the following three components, or gates:

1. **Forget gate**: the forget gate decides when specific portions of the cell state are to be replaced with more recent information. It outputs values close to 1 for parts of the cell state that should be retained, and zero for values that should be neglected.

**(2)** **Input gate** : based on the input (i.e., previous output o(t-1), input x(t), and previous cell state c(t-1)), this section of the network learns the conditions under which any information should be stored (or updated) in the cell state

**(3) Output gate**: depending on the input and cell state, this portion decides what information is propagated forward (i.e., output o(t) and cell state c(t) to the next node in the network.

Thus, LSTM networks are ideal for exploring how variation in one stock's price can affect the prices of several other stocks over a long period of time.



* **Stock Prediction Algorithm**

**Input :** Historical stock price data

**Output :** Prediction for stock prices based on stock price variation

1. Start
2. Stock data is taken and stored in a numpy array of 3 dimensions (N,W,F) where :

* N is number of training sequence
* W is sequence length
* F is the number of features of each sequence

1. A network structure is built with [l,a,b,l] dimensions, where there is *l* input layer, *a* neurons in the next layer, *b* neurons in the subsequent layer, and a single layer with a linear activation function.
2. Train the constructed network on the data.
3. Use the output of the last layer as predicted of the next time step.
4. Repeat steps 4 and 5 until optimal convergence is reached.
5. Obtain predictions by providing test data as input to the network.
6. Evaluate accuracy by comparing predictions made with actual data.
7. End

* **Terminologies used**

Given below is a brief summary of the various terminologies relating to our proposed stock prediction system:

1. **Training set :** subsection of the original data that is used to train the neural network model for predicting the output values.
2. **Test set :** part of the original data that is used to make predictions of the output value, which are then compared with the actual values to evaluate the performance of the model.
3. **Validation set :** portion of the original data that is used to tune the parameters of the neural network model.

4. **Activation function:** in a neural network, the activation function of a node defines the output of that node as a weighted sum of inputs.

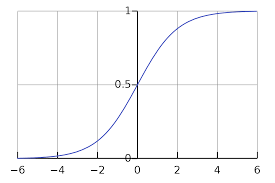
*Activation fuction = ∑(Input \* weights) + Bias*

Here, the sigmoid and ReLU (Rectified Linear Unit) activation functions were tested to optimize the prediction model.

1. **Sigmoid** – has the following formula

*y = 1/(1+e-x)*

and graphical represenation



1. **ReLU** – has the following formula

*y = max(0,x)*

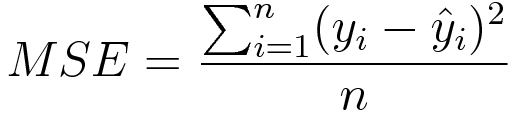
**5.Batch size :** number of samples that must be processed by the model before updating the weights of the parameters.

6. **Epoch :** a complete pass through the given dataset by the training algorithm.

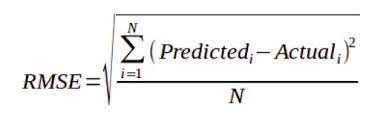
7. **Dropout:** a technique where randomly selected neurons are ignored during training i.e., they are “dropped out” randomly. Thus, their contribution to the activation of downstream neurons is temporally removed on the forward pass, and any weight updates are not applied to the neuron on the backward pass.

8. **Loss function :** a function, defined on a data point, prediction and label, that measures a penalty such as square loss.

1. **Cost function:** a sum of loss functions over the training set. An example is the Mean Squared Error (MSE), which is mathematically explained as follows:



1. **Root Mean Square Error (RMSE):** measure of the difference between values predicted by a model and the values actually observed. It is calculated by taking the summation of the squares of the differences between the predicted value and actual value, and dividing it by the number of samples. It is mathematically expressed as follows: In general, smaller the RMSE value, greater the accuracy of the predictions made.



* **Obtaining dataset and preprocessing**

The obtained data contained five features:

1. Date: of the observation

2. Opening price: of the stock

3. High: highest intra-day price reached by the stock

4. Low: lowest intra-day price reached by the stock

5. Volume: number of shares or contracts bought and sold in the market during the day

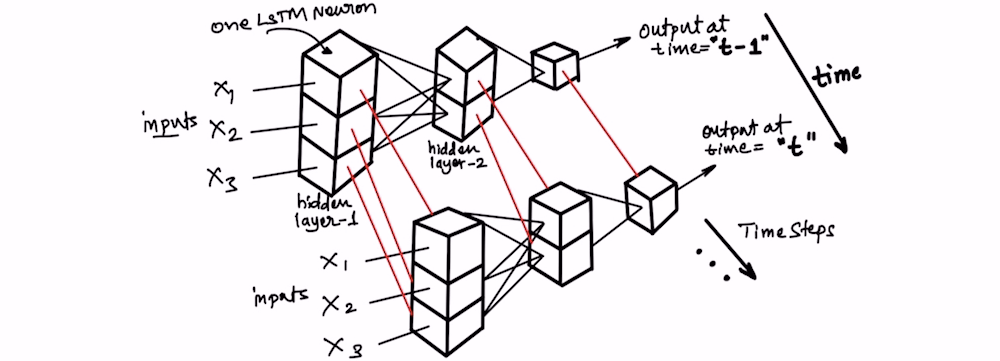
6. OpenInt i.e., Open Interest: how many futures contracts are currently outstanding in the market

The above data was then transformed into a format suitable for use with our prediction model by performing the following steps:

1. Transformation of time-series data into input-output components for supervised learning

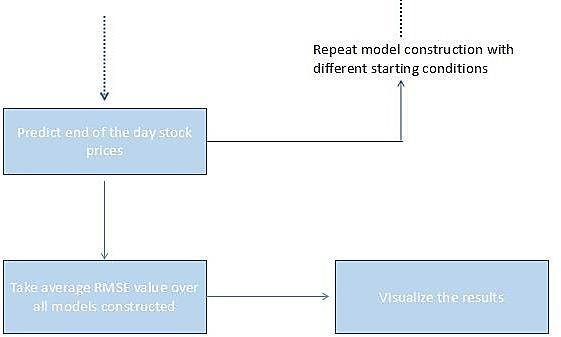
2. Scaling the data to the [-1, +1] range

* **Construction of prediction model**

****

The input data is split into training and test datasets; our LSTM model will be fit on the training dataset, and the accuracy of the fit will be evaluated on the test dataset. The LSTM network is constructed with one input layer having five neurons, 'n' hidden layers (with 'm' LSTM memory cells per layer), and one output layer (with one neuron). After fitting the model on the training dataset, hyper-parameter tuning is done using the validation set to choose the optimal values of parameters such as the number of hidden layers 'n', number of neurons 'm' per hidden layer, batch size, etc.

* **Prediction and Accuracy**

****

Once the LSTM model is fit to the training data, it can be used to predict the end-of-day stock price of an arbitrary stock.

This prediction can be performed in two ways:

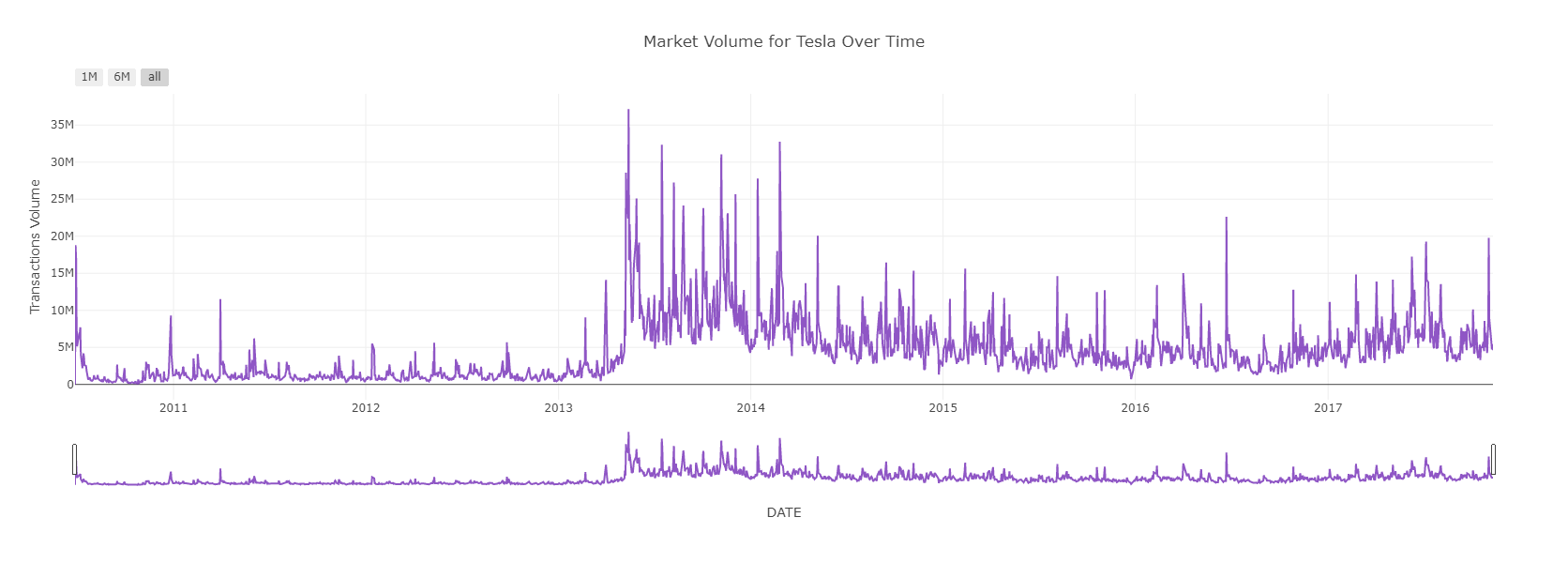
1. Static – a simple, less accurate method where the model is fit on all the training data. Each new time step is then predicted one at a time from test data.
2. Dynamic – a complex, more accurate approach where the model is refit for each time step of the test data as new observations are made available.

The accuracy of the prediction model can then be estimated robustly using the RMSE (Root Mean Squared Error) metric. This is due to the fact that neural networks in general (including LSTM) tend to give different results with different starting conditions on the same data.

1. **Results :**
2. Stock Data –

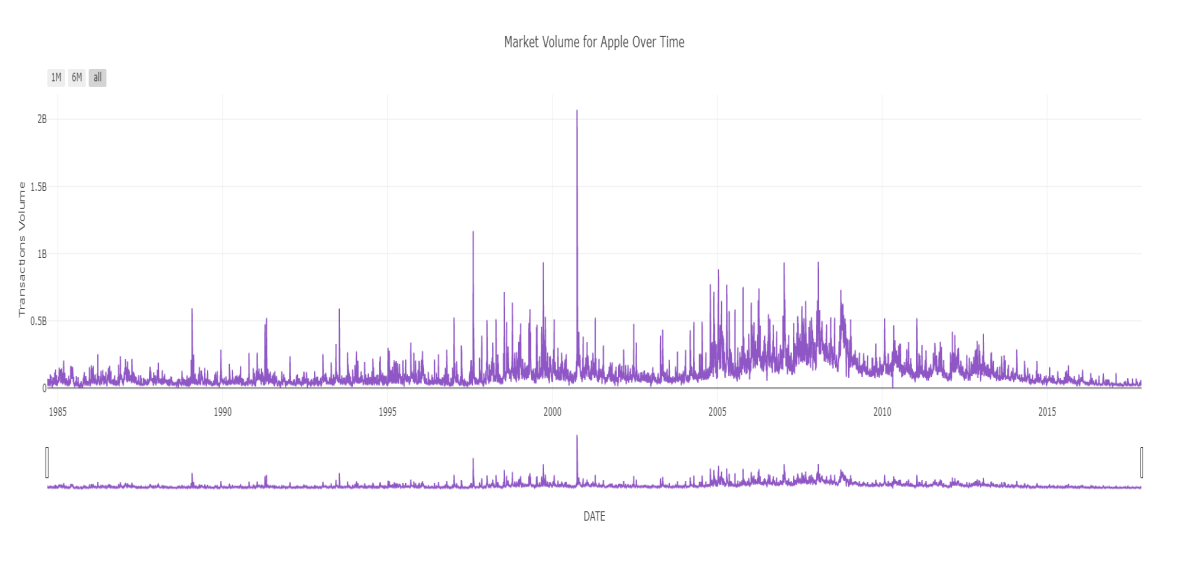
* Tesla



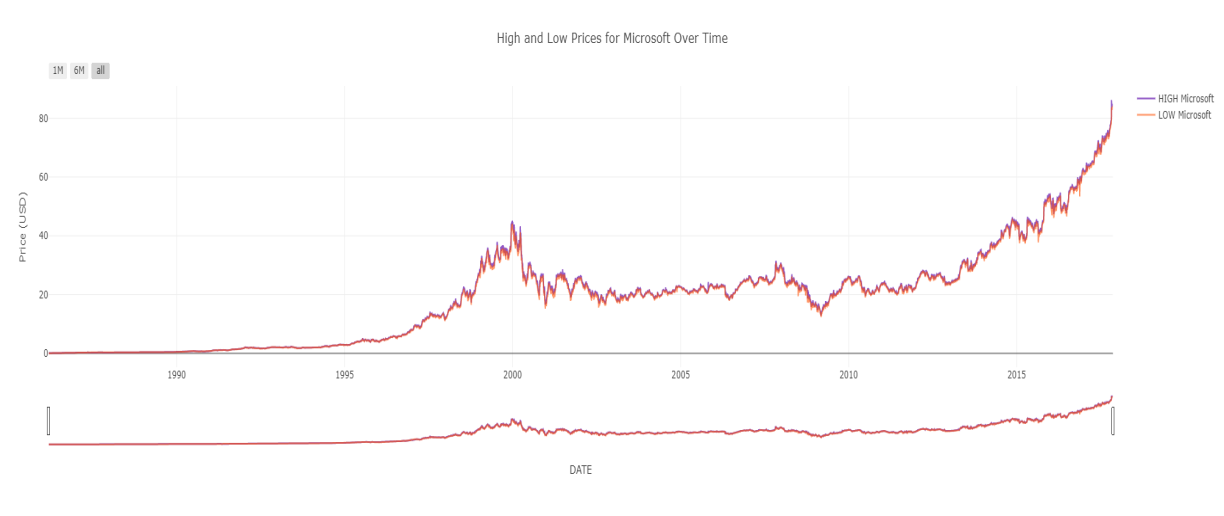


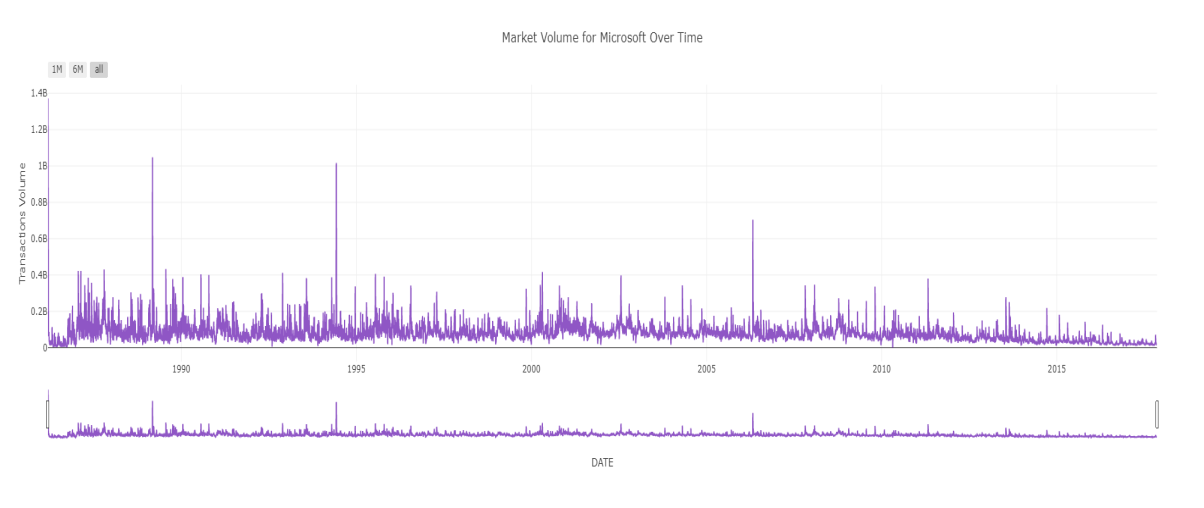
* Apple



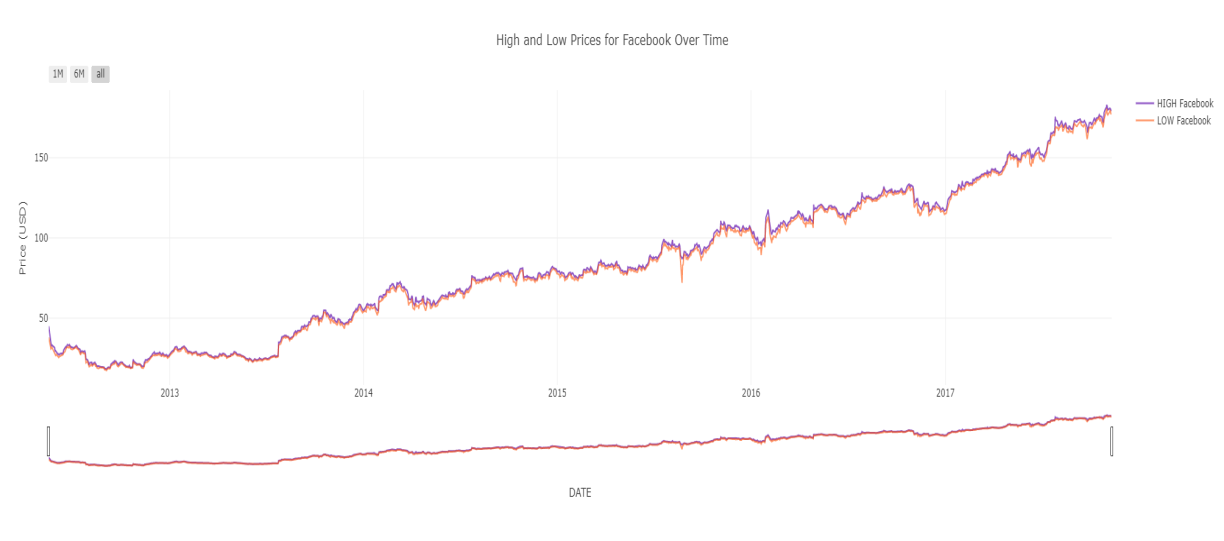


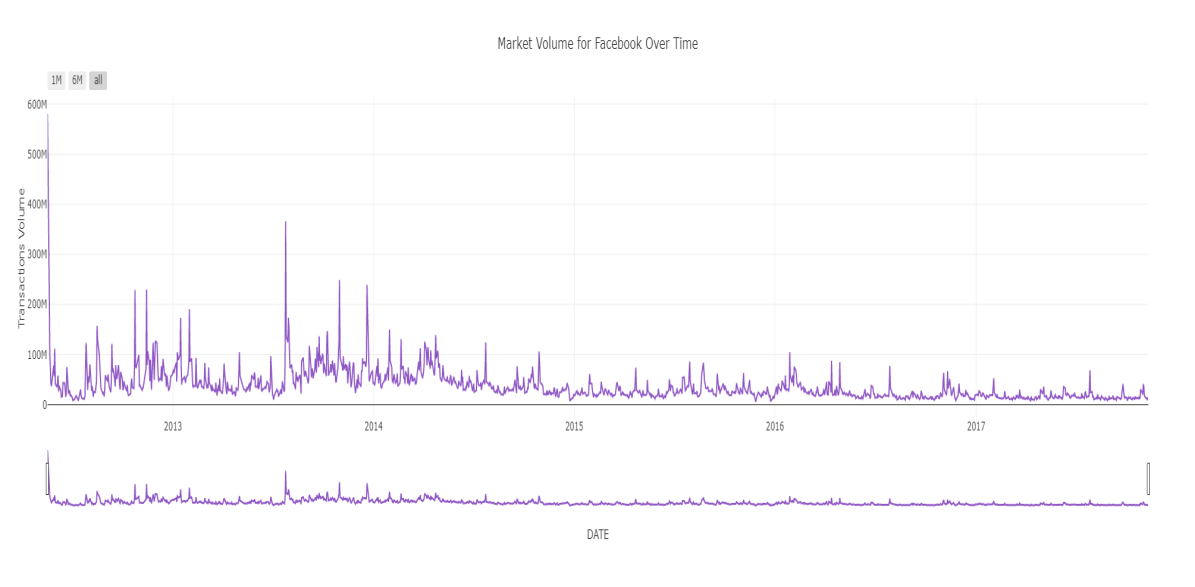
* Microsoft



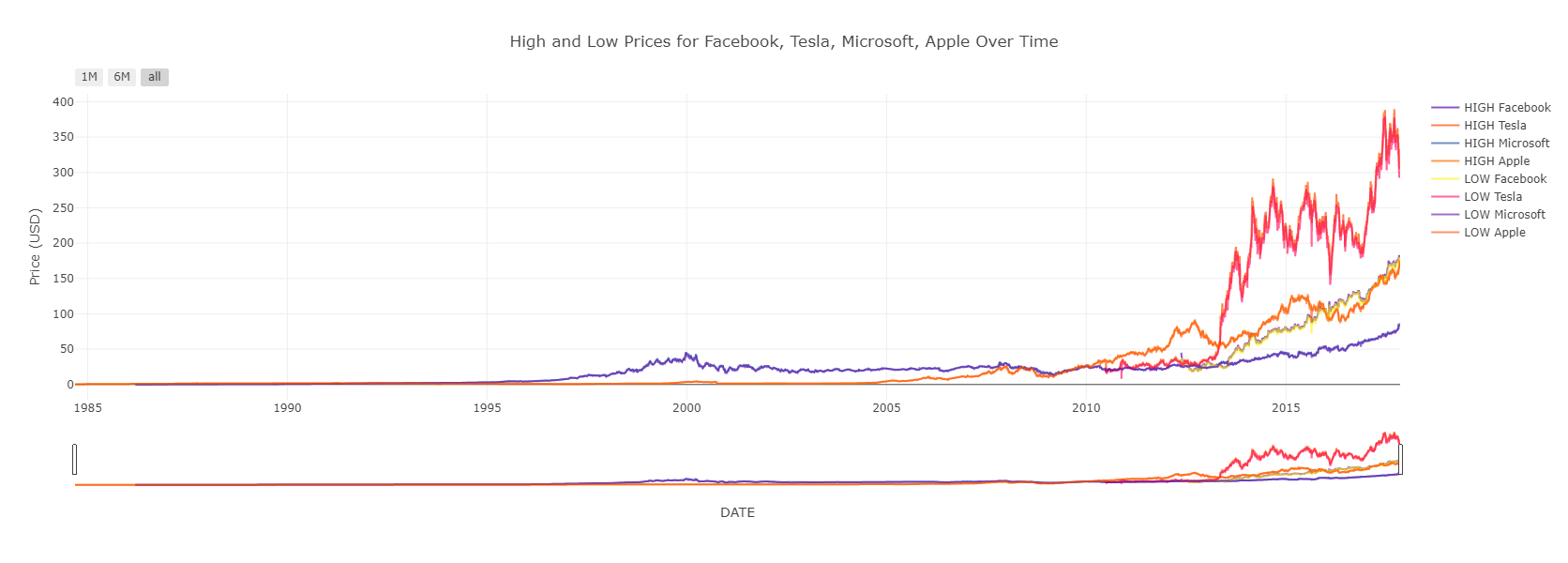


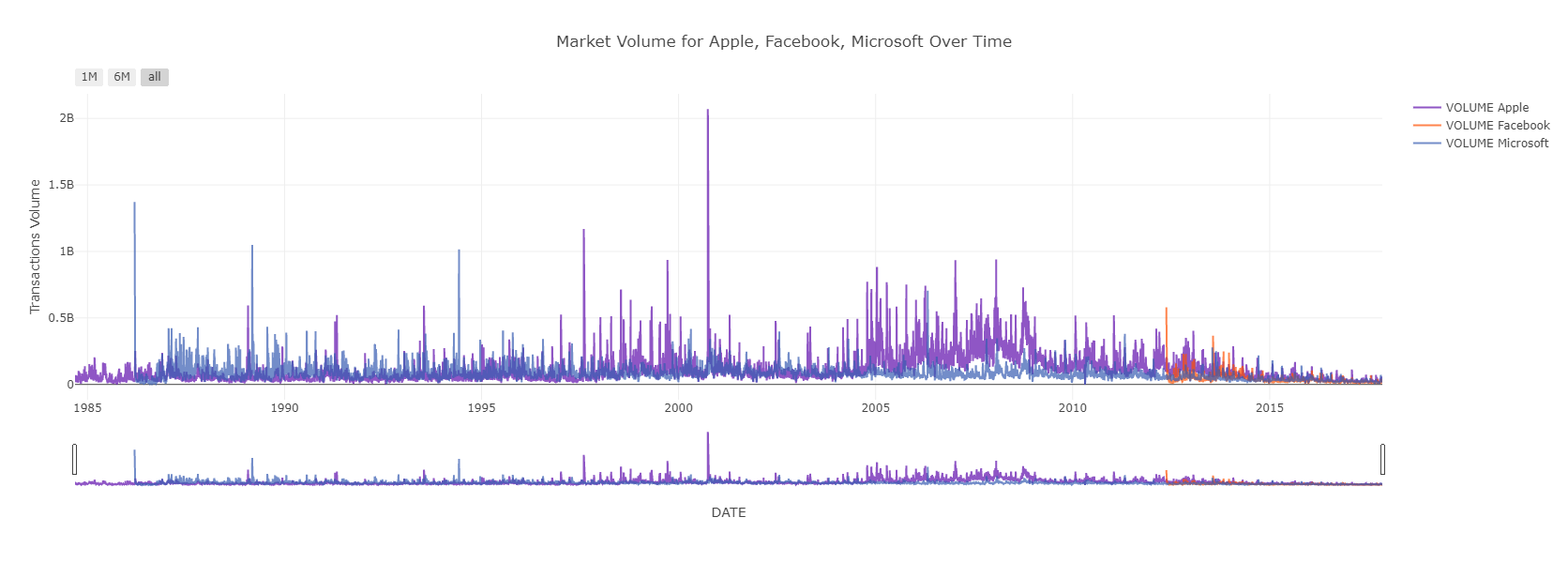
* Facebook





Comparison plot between all the four companies :





1. Microsoft Data

