**UE15CS356**

**MACHINE LEARNING**

**LABORATORY**

**A Project**

**On**

**Stock Market Prediction**

|  |  |
| --- | --- |
| Harshita Singh | 01FB15ECS123 |
| Kanika Priyadarshi | 01FB15ECS136 |
| Kopparam Hithyshi | 01FB15ECS151 |
| Likhita Navali | 01FB15ECS156 |

**Problem Statement**

A stock market is the aggregation of buyers and sellers of stocks, which represent ownership claims on businesses.

In this project, we have tried to predict the stock prices of four companies:

* Silicon Laboratories (SLAB)
* National Research Corporation Class B (NRCIB)
* Tesla, Inc. (TSLA)
* Neovasc Inc. (NVCN)

**OBJECTIVE:** To predict the stock prices of the above four companies to recommend investment in the company with the highest and most stable stock price.

**Dataset Details**

We have used four datasets, one for each of the four companies. The data sets have been taken from Yahoo! Finance, which provides financial news, data and commentary including stock quotes, press releases, financial reports, and original content.

All the companies are listed on the NASDAQ Stock Market.

The four datasets are:

* TSLA.csv
* SLAB.csv
* NRCIB.csv
* NVCN.csv

We will also be using real-time tweets from reputed media houses like Times Of India, etc.

**Total Attributes: 10**

**Attributes List:**

* **Date** - It is the date for which the stock prices has been listed.
* **Open** - The opening price is the price at which a security first trades upon the opening of an exchange on a given trading day.
* **High** - The day's highest price of a security that has changed hands between a buyer and a seller.
* **Low** - The day's lowest price of a security that has changed hands between a buyer and a seller.
* **Close** - It is the price of the last transaction of a particular stock completed during a day's trading session on an exchange.
* **Adj Close** - Adjusted Close is a stock's closing price on any given day of trading that has been amended to include any distributions and corporate actions that occurred at any time prior to the next day's open.
* **Volume** - It is the number of shares of stock traded during a particular time period, normally measured in average daily trading volume.
* **Profit -** Quarterly profits of the company
* **Tweets -** Ratio of percentage of positive tweets over percentage of negative tweets
* **Timestamp -** Calculated based on the date column.

Features extracted from tweets will be the attributes for the Twitter Real-Time Dataset.

**ML Techniques: Survey**

ML technique chosen by us during the initial phase of this project was Multilayer Perceptron (MLP).

**Multilayer Perceptron (MLP):** It is a class of feedforward artificial neural network. An MLP consists of at least three layers of nodes. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLP utilizes a supervised learning technique called backpropagation for training. Its multiple layers and non-linear activation distinguish MLP from a linear perceptron. It can distinguish data that is not linearly separable.

The ML technique chosen by for the final implementation is K-Nearest Neighbours.

**K-Nearest Neighbours (KNN):** K-nearest neighbor technique is a machine learning algorithm that is considered as simple to implement (Aha et al. 1991). The stock prediction problem can be mapped into a similarity based classification.

In this model, we applied k-nearest neighbor algorithm in order to predict stock prices for a sample of one major company listed on the NASDAQ stock exchange to assist investors, management, decision makers, and users in making correct and informed investments decisions. According to the results, the kNN algorithm is robust with small error ratio; consequently the results were rational and also reasonable. In addition, depending on the actual stock prices data; the prediction results were close and almost parallel to actual stock prices.

**Design Document**

**Architectural Design**

For initial implementations, we used MLP to predict the stock prices. The design of this technique was:

Our neural network will have one input layer, one output layer and three hidden layers. This would be our basic structure, which may be altered later according to our needs.

* Each unit is perceptron.
* There will be no connections within a layer.
* No direct connections between input and output layers.
* Fully connected between layers.
* Number of output units need not equal number of input units.
* Include bias as an extra weight.

Backpropagation will be implemented. It is divided into two phases:

* Forward pass phase - Computes functional signal, feed-forward propagation of input pattern signals through network.
* Backward pass phase - Computes error signal, propagates the error backwards through network starting at output units (where the error is the difference between actual and desired output values).

**Algorithm:**

**Step 1:** Initialize weights at random, choose a learning rate η.

Until network is trained:

**Step 2:** Do forward pass through network (with fixed weights) to produce output(s)

* + i.e., in Forward Direction, layer by layer:
    - Inputs applied
    - Multiplied by weights
    - Summed
    - ‘Squashed’ by sigmoid activation function
    - Output passed to each neuron in next layer
  + Repeat above until network output(s) produced

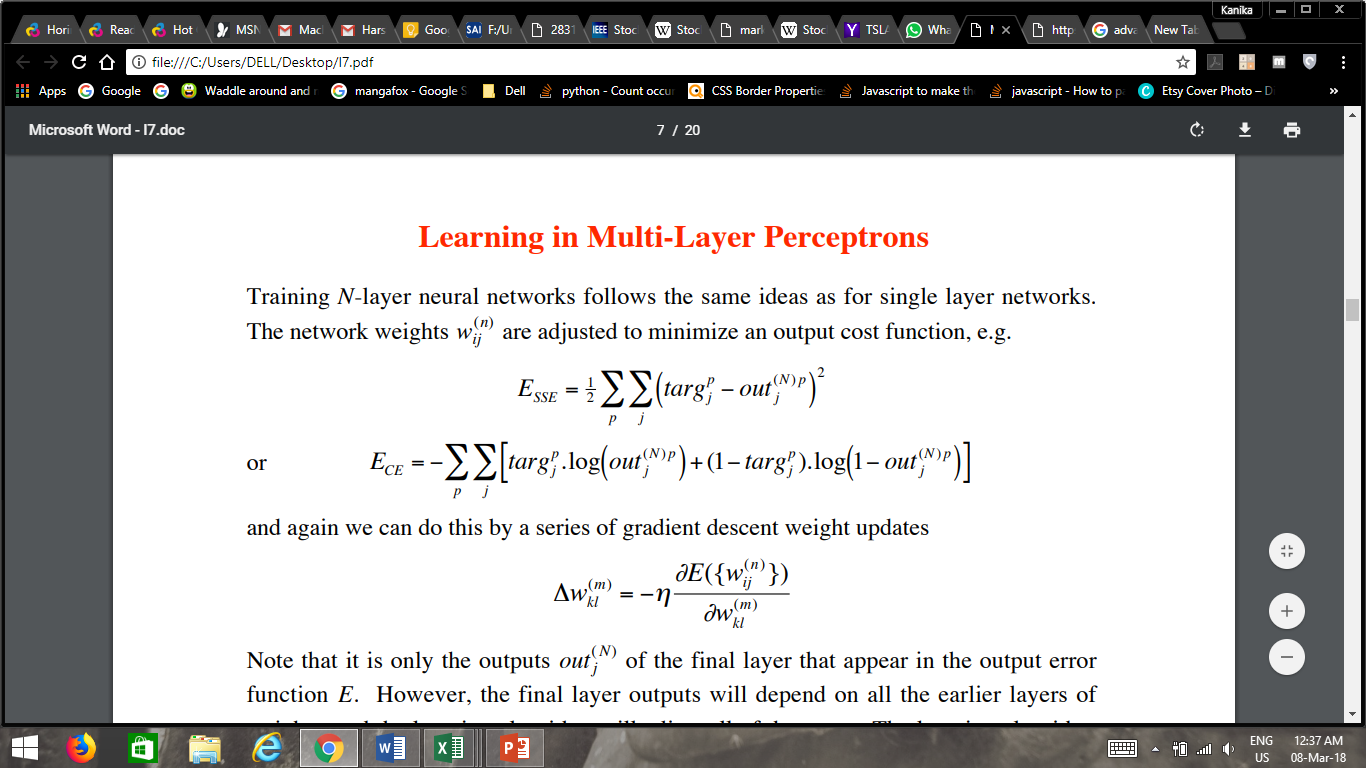
• Compute error (delta or local gradient) for each output unit δ *k*

• Layer-by-layer, compute error (delta or local gradient) for each hidden unit δ *j*  by back propagating errors (as shown previously)

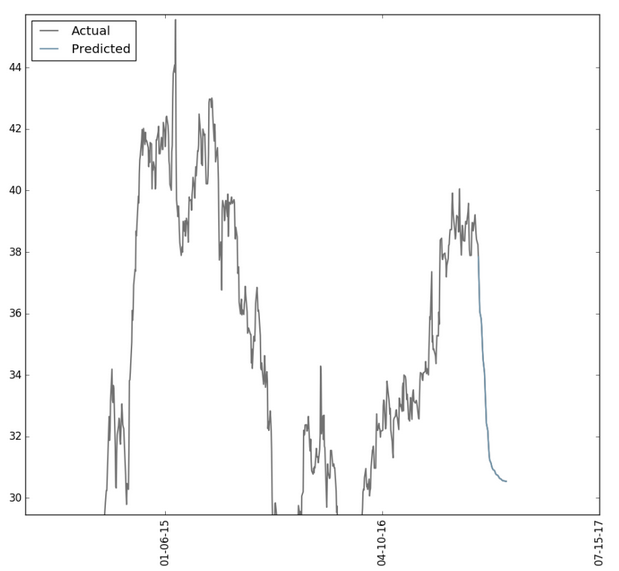
**Step 4:** Next, update all the weights Δ*wij.* By gradient descent, and go

back to Step 2.

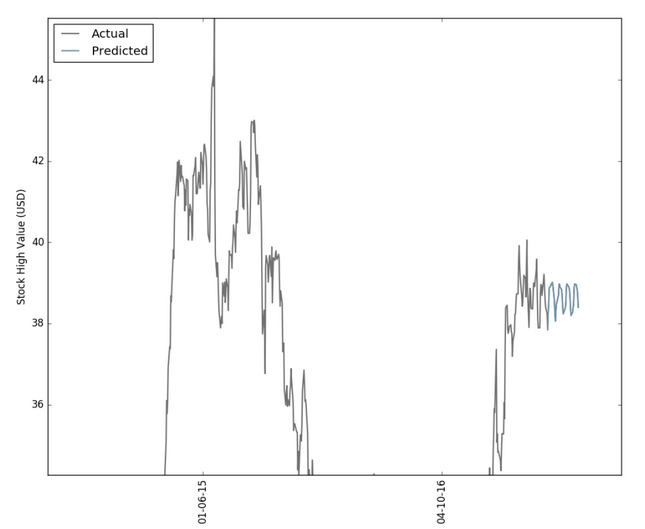
* The overall MLP learning algorithm, involving forward pass and backpropagation of error (until the network training completion), is known as the Generalised Delta Rule (GDR), or more commonly, the Back Propagation (BP) algorithm.



After more research, we realised that our MLP model was not predicting the stocks with the desired accuracy. Therefore we started to consider other ML techniques in our quest to get the desired accuracy. After intensive research, we realised that K-Nearest Neighbours (KNN) gave us the accuracy that we wanted.



MLP

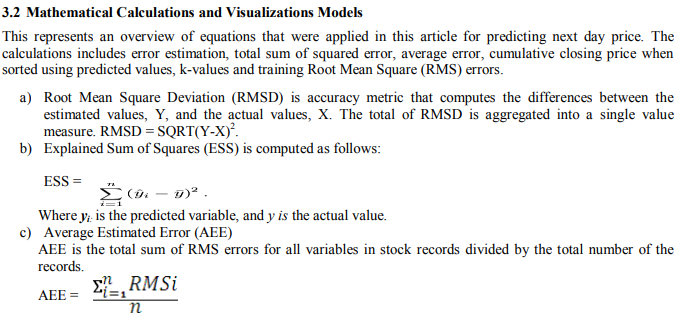


KNN

Thus, we chose KNN as the technique to use for our final implementation of our predictor model.

**Architectural Design:**

The historical stock data and the test data is mapped into a set of vectors. Each vector represents N dimension for each stock features. Then, a similarity metric such as Euclidean distance is computed to take a decision. In this section, a description of kNN is provided. kNN is considered a lazy learning that does not build a model or function previously, but yields the closest k records of the training data set that have the highest similarity to the test (i.e. query record). Then, a majority vote is performed among the selected k records to determine the class label and then assigned it to the query record. The prediction of stock market closing price is computed using kNN as follows:



**Algorithm:**

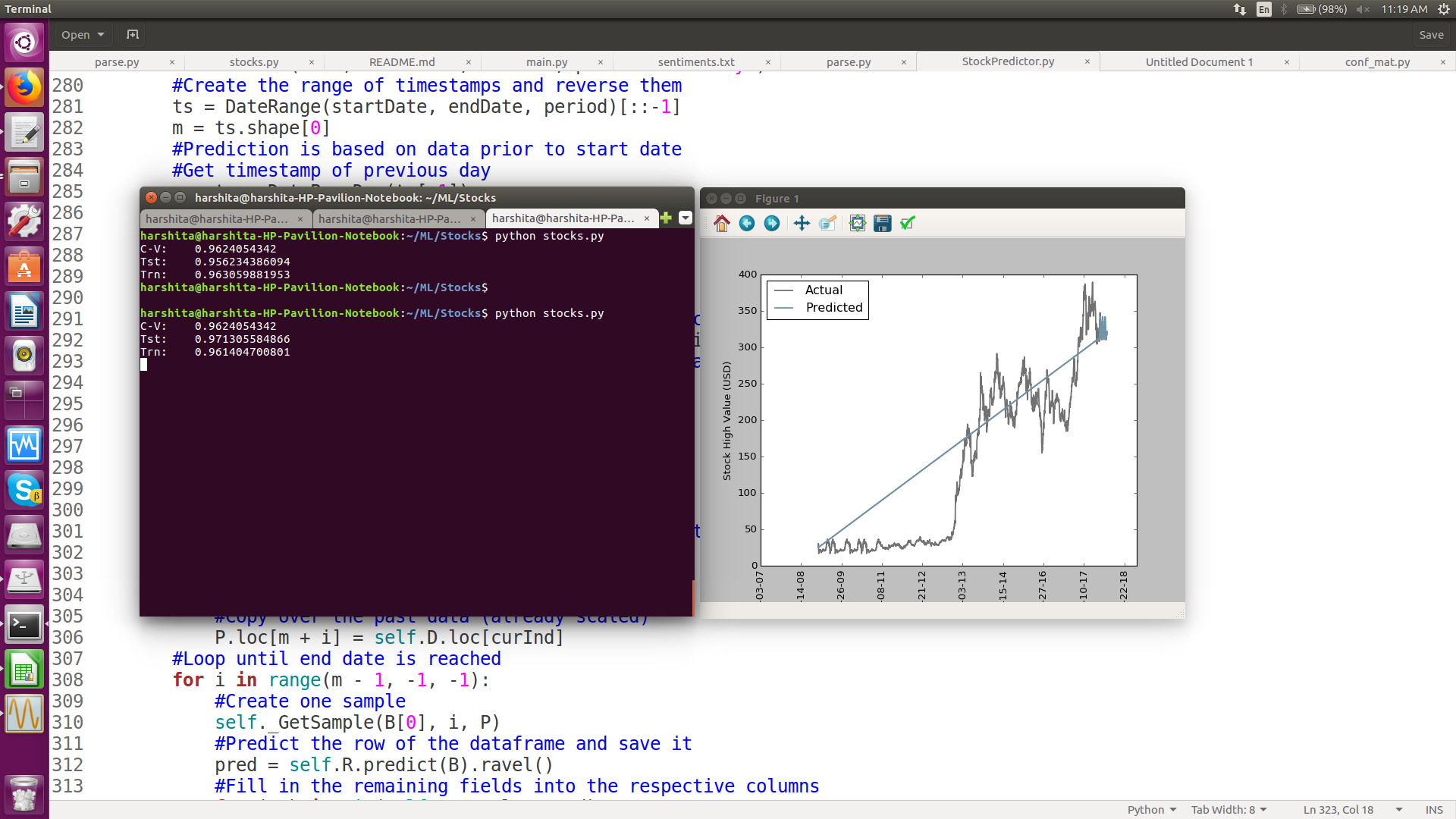
a) Determine the number of nearest neighbors, k.

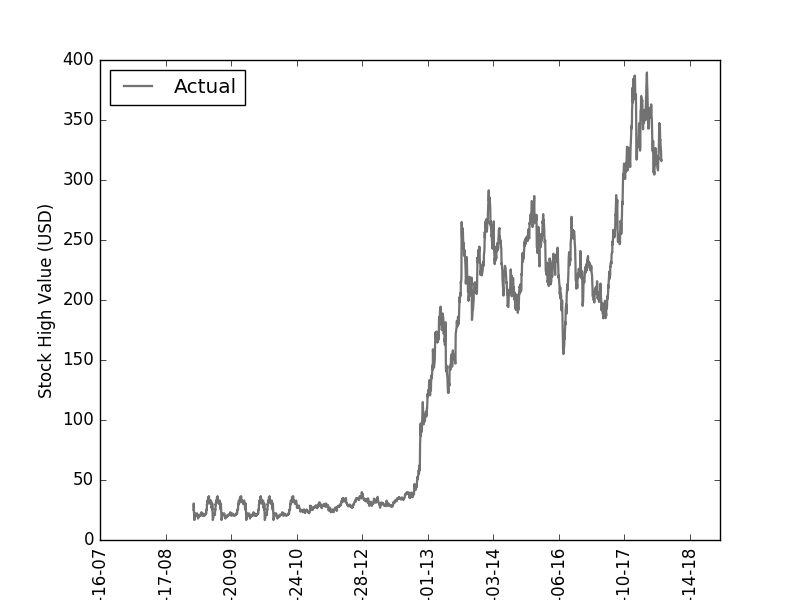
b) Compute the distance between the training samples and the query record.

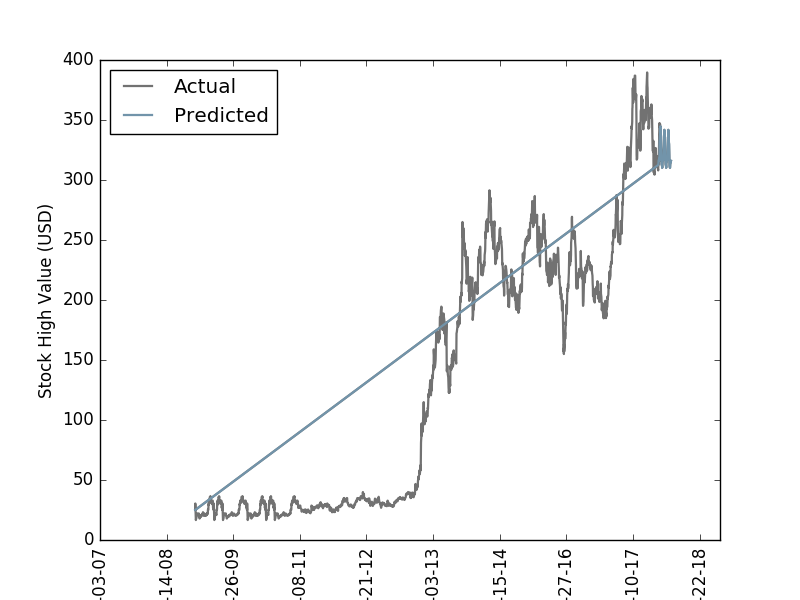
c) Sort all training records according to the distance values.

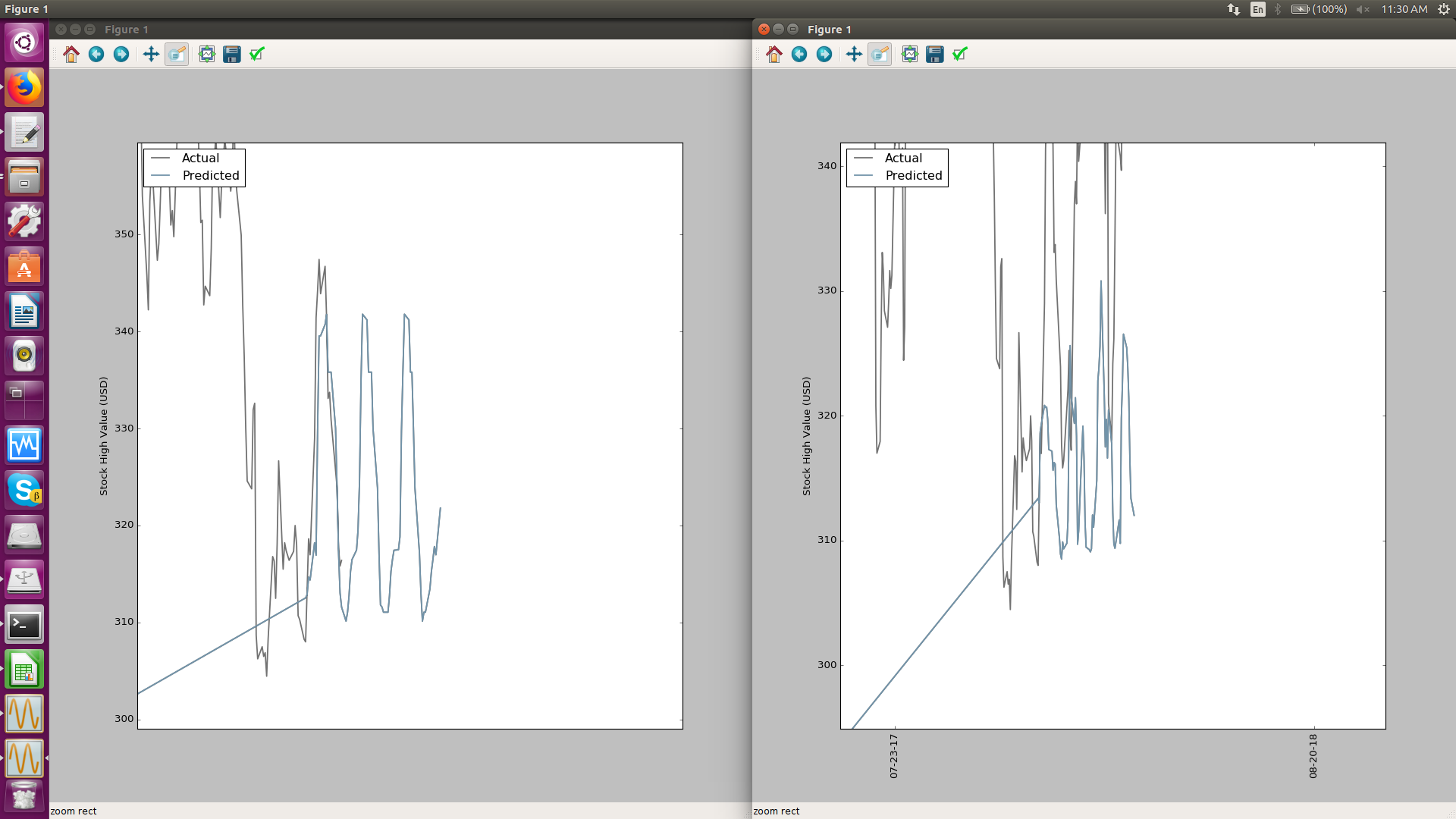
d) Use a majority vote for the class labels of k nearest neighbors, and assign it as a prediction value of the query record.

**Result Analysis**





****

****

Left: With tweets and profit columns. Right: Without tweets and profit column.

The original dataset (without profit and tweets column) gave lesser accuracy compared to the dataset with these columns.

**Conclusion**

The given model has been run only on one company (TSLA.csv). Since it gave us an accuracy of 95-96% , we can conclude that our model can predict the stock prices for investment with great confidence.

**REFERENCES**

<https://github.com/nicholastoddsmith/pythonml/blob/master/Stocks/stocks.py>

<http://www.ijbhtnet.com/journals/Vol_3_No_3_March_2013/4.pdf>

<http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsRegressor.html>

<https://en.wikipedia.org/wiki/Sentiment_analysis>

<https://nicholastsmith.wordpress.com/2016/11/04/stock-market-prediction-in-python-part-2/>