**Stock Market Trend Prediction Using K-Nearest Neighbor(KNN) Algorithm**

In this paper author is evaluating performance of KNN(K-Nearest Neighbor) supervised machine learning algorithm. In the finance world stock trading is one of the most important activities. Stock market prediction is an act of trying to determine the future value of a stock other financial instrument traded on a financial exchange. The programming language is used to predict the stock market using machine learning is Python. In this paper we propose a Machine Learning (ML) approach that will be trained from the available stocks data and gain intelligence and then uses the acquired knowledge for an accurate prediction. In this context this study uses a machine learning technique called K-Nearest Neighbor to predict stock prices for the large and small capitalizations and in the three different markets, employing prices with both daily and up-to-the-minute frequencies.

Predicting the Stock Market has been the bane and goal of investors since its existence. Everyday billions of dollars are traded on the exchange, and behind each dollar is an investor hoping to profit in one way or another. Entire companies rise and fall daily based on the behaviour of the market. Should an investor be able to accurately predict market movements, it offers a tantalizing promises of wealth and influence. It is no wonder then that the Stock Market and its associated challenges find their way into the public imagination every time it misbehaves. The 2008 financial crisis was no different, as evidenced by the flood of films and documentaries based on the crash. If there was a common theme among those productions, it was that few people knew how the market worked or reacted. Perhaps a better understanding of stock market prediction might help in the case of similar events in the future.

Despite its prevalence, Stock Market prediction remains a secretive and empirical art. Few people, if any, are willing to share what successful strategies they have. A chief goal of this project is to add to the academic understanding of stock market prediction. The hope is that with a greater understanding of how the market moves, investors will be better equipped to prevent another financial crisis. The project will evaluate some existing strategies from a rigorous scientific perspective and provide a quantitative evaluation of new strategies.

There are several data mining algorithms that can be used for prediction purposes in the field of finance. Some examples would be the naive Bayes classifier, the k nearest neighbour (KNN) algorithm and the classification and the regression tree algorithm (Wu et al. 2007). All the mentioned algorithms could fill the purpose of the paper but it will center around the kNN algorithm as a method of predicting stock market movements as well as the MA formula. The movements will be detected by looking at a large amount of historical data and finding patterns to establish a well estimated forecast. This specific algorithm was chosen as it is a simple but a very effective algorithm to implement when looking at large amounts of data (Berson et al. 1999).The KNN algorithm simply states: "Objects that are ’near’ to each other will have similar prediction values as well. Thus if you know the prediction value of one of the objects you can predict it for its nearest neighbours" (Berson et al. 1999). As a comparison with the KNN algorithm, the MA formula was chosen. The MA formula has its simplicity as a common factor with the KNN algorithm, but it is a statistical method used frequently by traders (Interactive Data Corp, 2014).

There are existing techniques when it comes to stock prediction, some of them are multispectral prediction, distortion controlled prediction and lempel-ziv based prediction. These are based on the fact that the data representation is more compact by removing redundancy while the essential information is kept in format that is accessible (Azhar et al. 1994). Due to the scope of the project the techniques that were the most suitable to work with were the KNN algorithm and the MA formula despite the existing techniques listed above.

To conduct experiment author has used Yahoo Finance stock Dataset and below is some example records of that dataset which contains request signatures. I have also used same dataset and this dataset is available inside ‘dataset’ folder.

Dataset example

['High', 'Low', 'Open', 'Close', 'Volume', 'Adj Close']

Above list are the columns of Yahoo finance

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **2017-01-05** | 116.860001 | 115.809998 | 115.919998 | 116.610001 | 22193600.0 | 111.393303 |
| **2017-01-09** | 119.430000 | 117.940002 | 117.949997 | 118.989998 | 33561900.0 | 113.666824 |

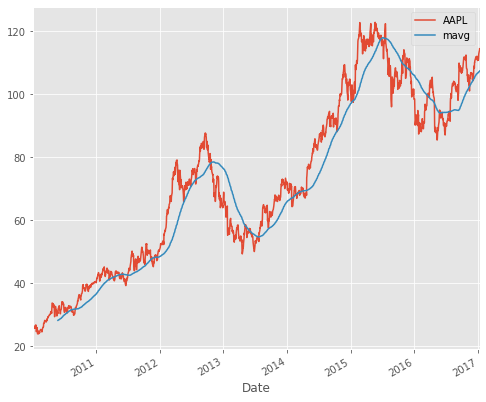
Above two records are the APPLE stock form the Yahoo Finance Dataset. For the rest of analysis, we will use the Closing Price which remarks the final price in which the stocks are traded by the end of the day.

we analyse stocks using two key measurements: Rolling Mean and Return Rate

**Rolling Mean:**

Rolling mean/Moving Average (MA) smooths out price data by creating a constantly updated [average price](https://www.investopedia.com/terms/a/averageprice.asp). This is useful to cut down “noise” in our price chart. Furthermore, this Moving Average could act as “Resistance” meaning from the downtrend and uptrend of stocks you could expect it will follow the trend and less likely to deviate outside its resistance point.

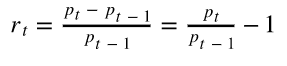
The Moving Average makes the line smooth and showcase the increasing or decreasing trend of stocks price.

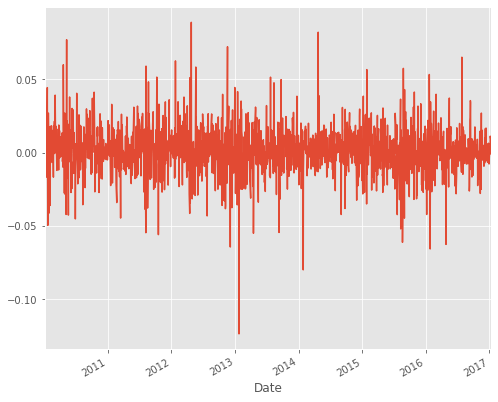


**Return Deviation — to determine risk and return**

*Expected****Return****measures the mean, or expected value, of the probability distribution of investment****returns****. The expected****return****of a portfolio is calculated by multiplying the weight of each asset by its expected****return****and adding the values for each investment — Investopedia.*

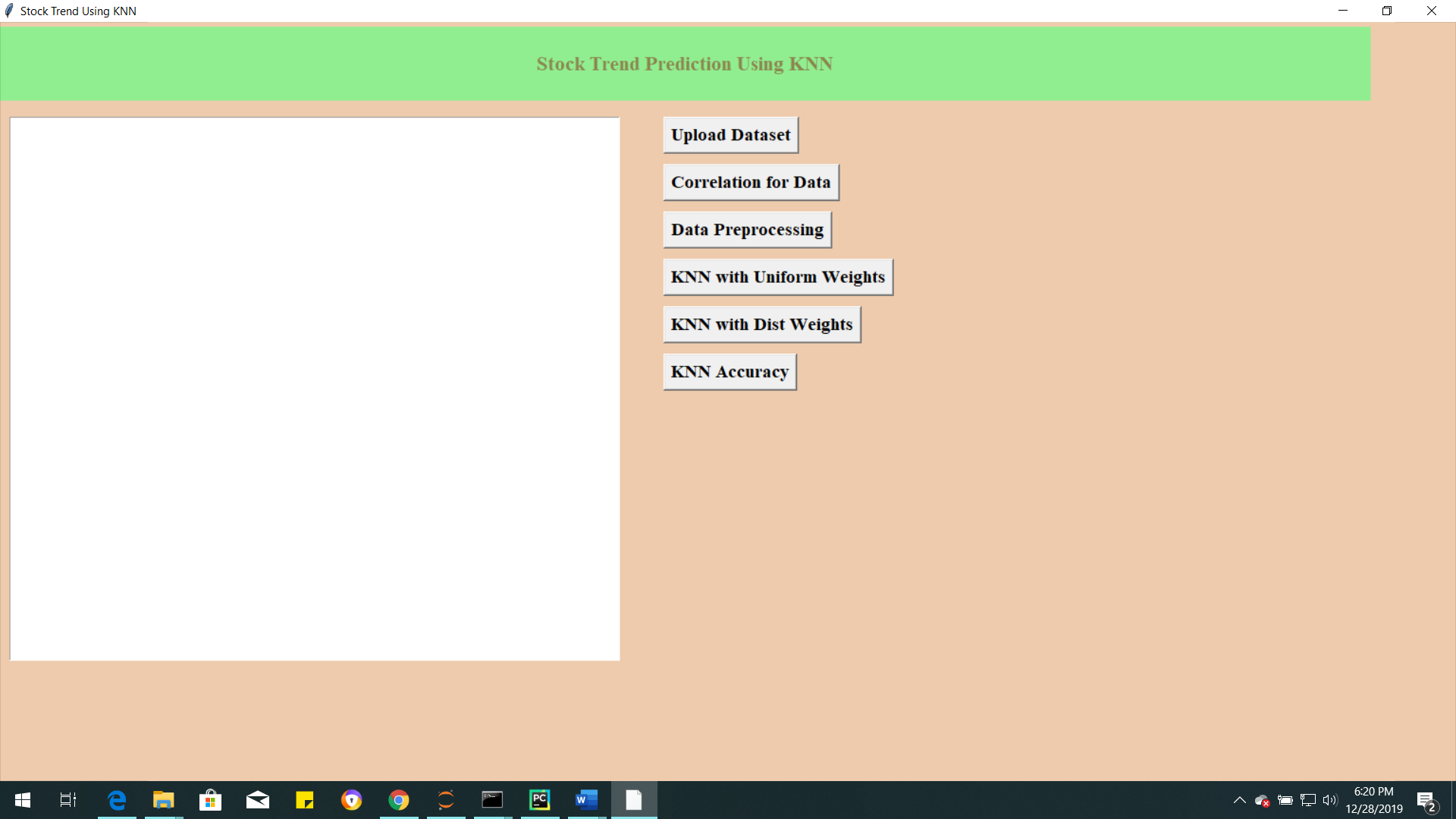
Following is the formula you could refer to:



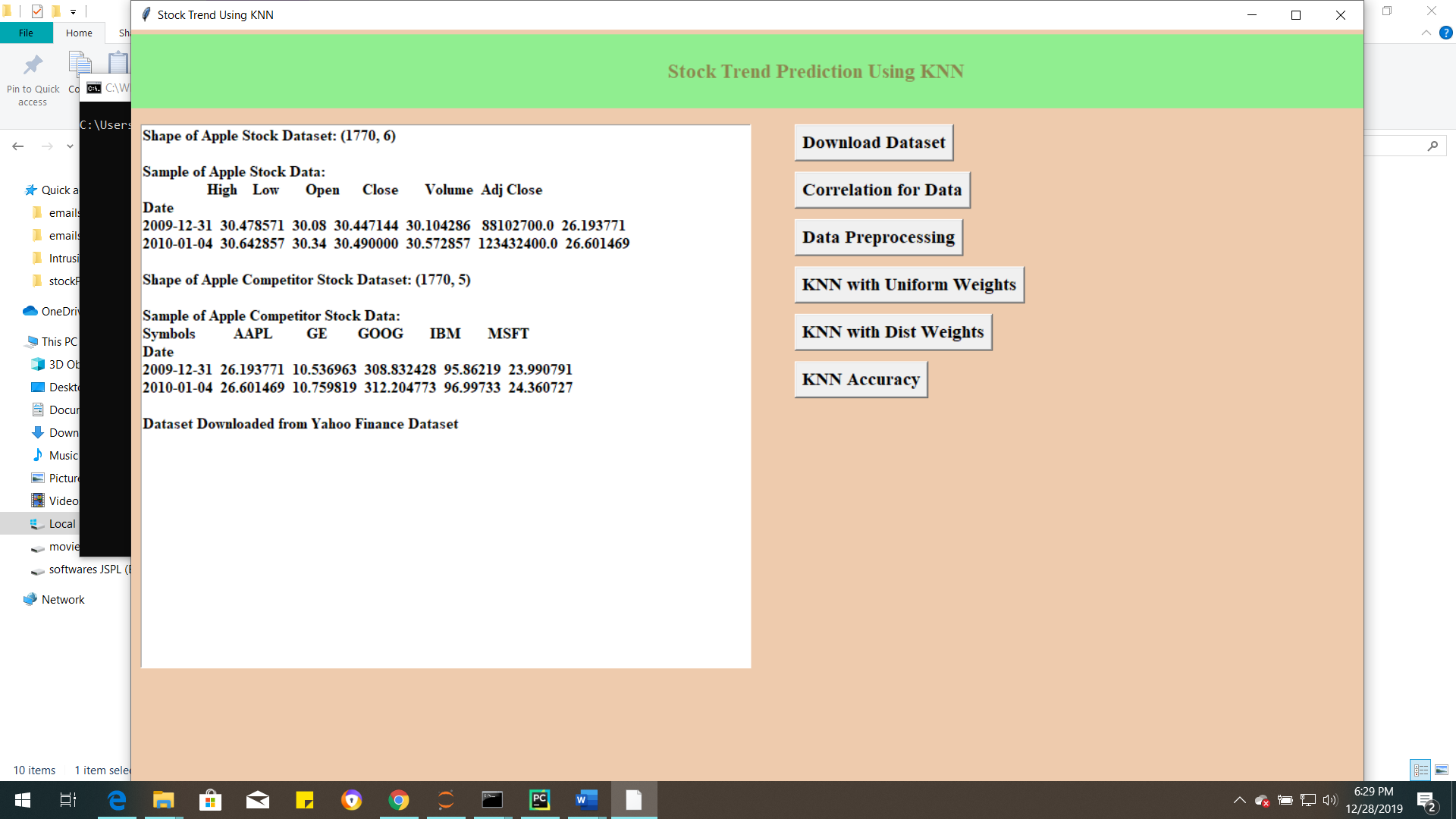


Before running code execute below two commands

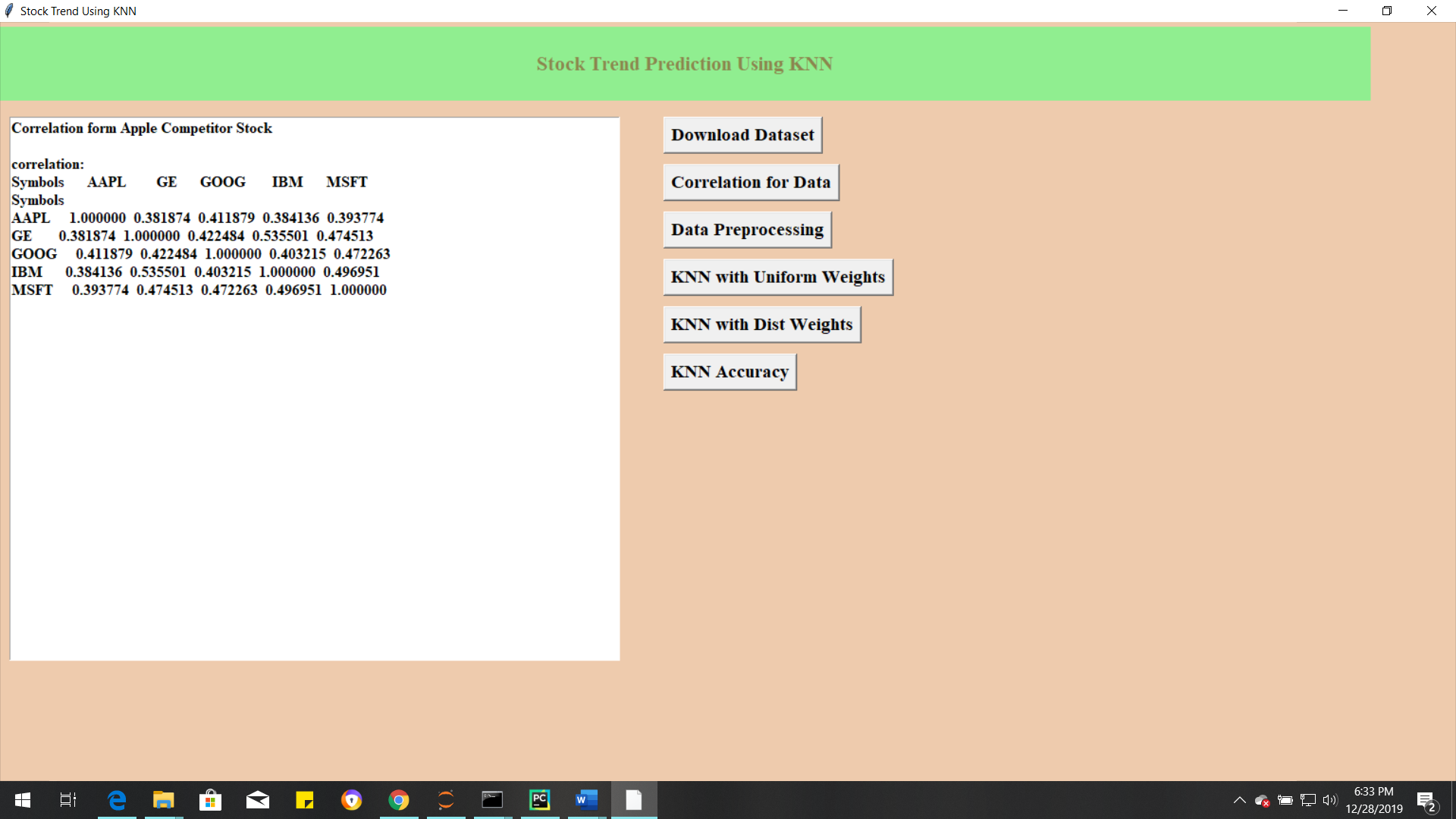
Screen shots

Double click on ‘run.bat’ file to get below screen

In above screen click on ‘Download Button’ download the Apple Stock and competitors data from Yahoo Finance Dataset

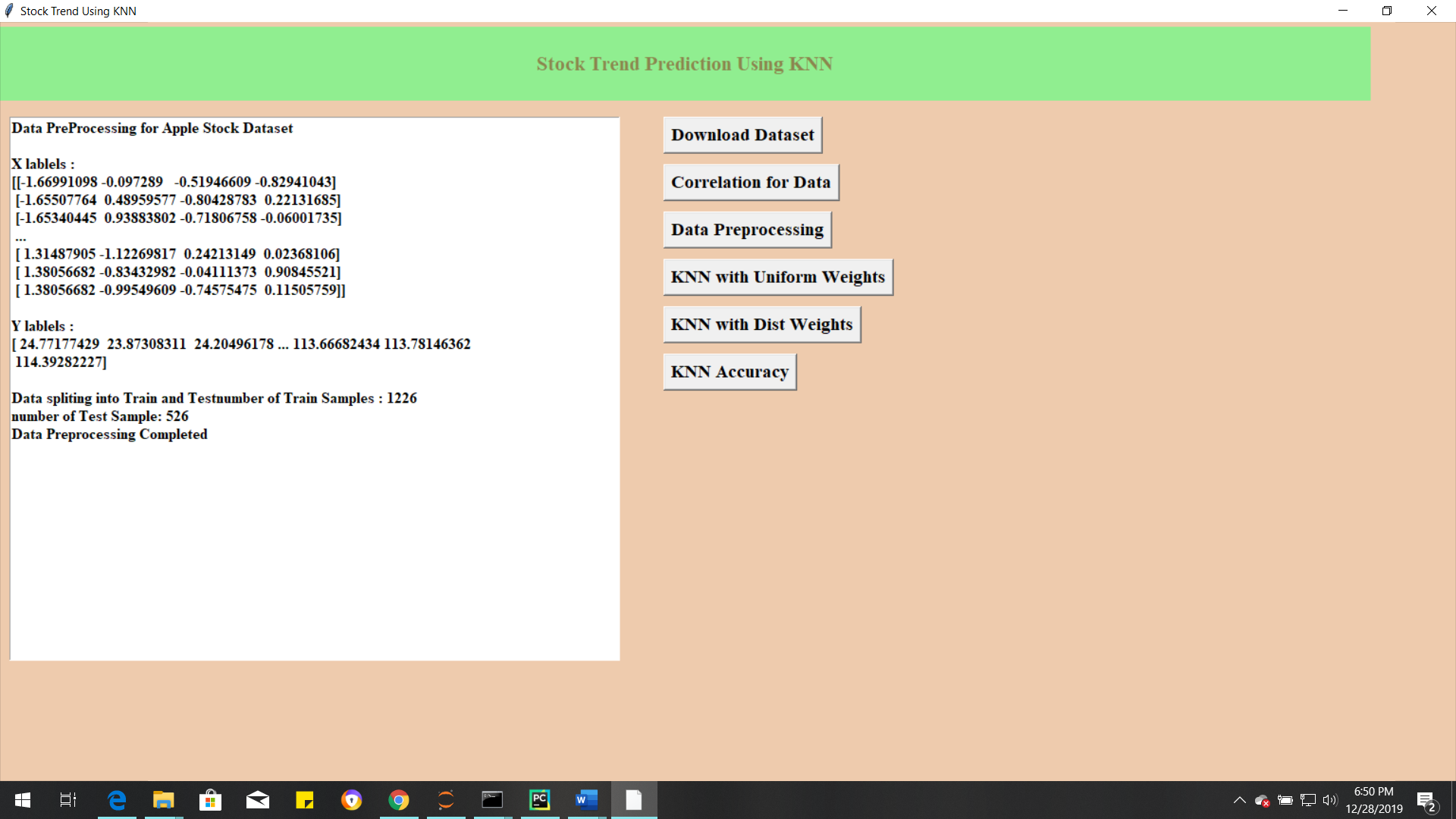


In above screen I am Downloading of Apple Stock and Apple competitor Stock Data from Yahoo Finance Dataset.



Now click on ‘Correlation Data’ Button to find the correlation between Apple and Competitor Stock market Dataset. show the trend in the technology industry rather than show how competing stocks affect each other.

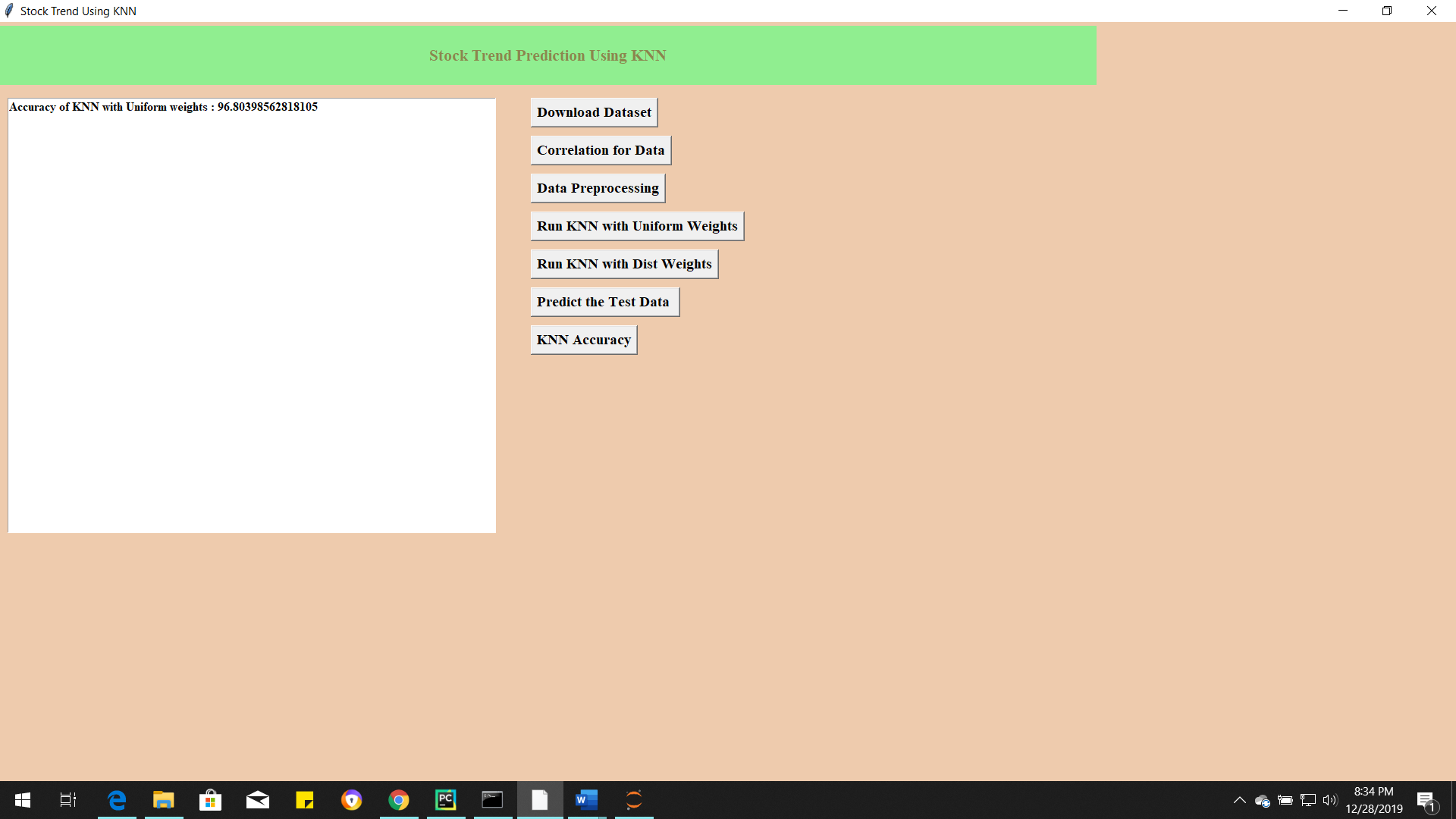
Now click on ‘Data PreProcessing button to drop missing values, split labels split train and test

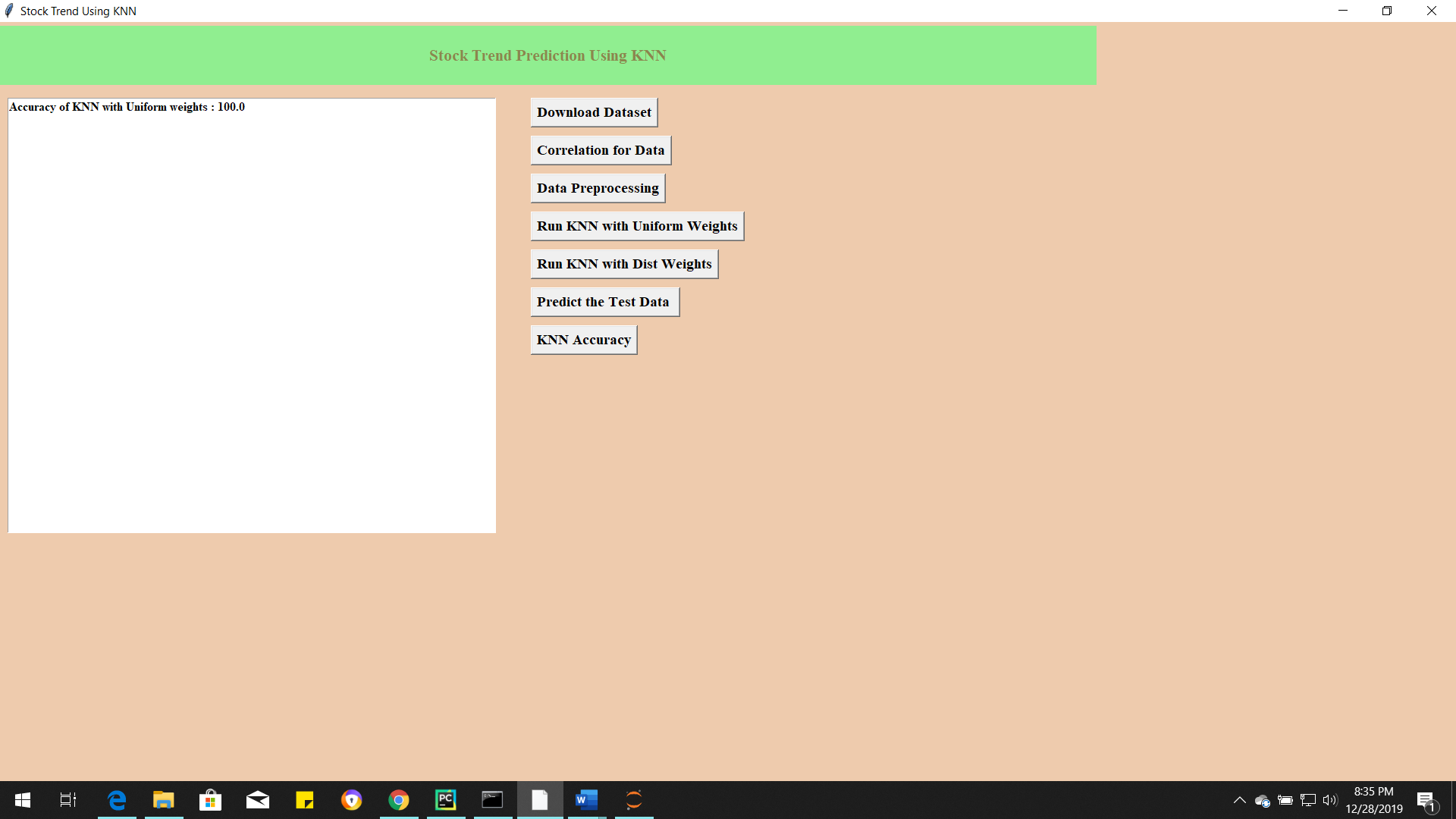


After pre-processing all missing values are dropped, Separating the label here, Scalling of X, find Data Series of late X and early X (train) for model generation and evaluation, Separate label and identify it as y and Separation of training and testing of model.

In above screen we can see dataset contains total 1752 records and 1226 used for training and 526 used for testing.

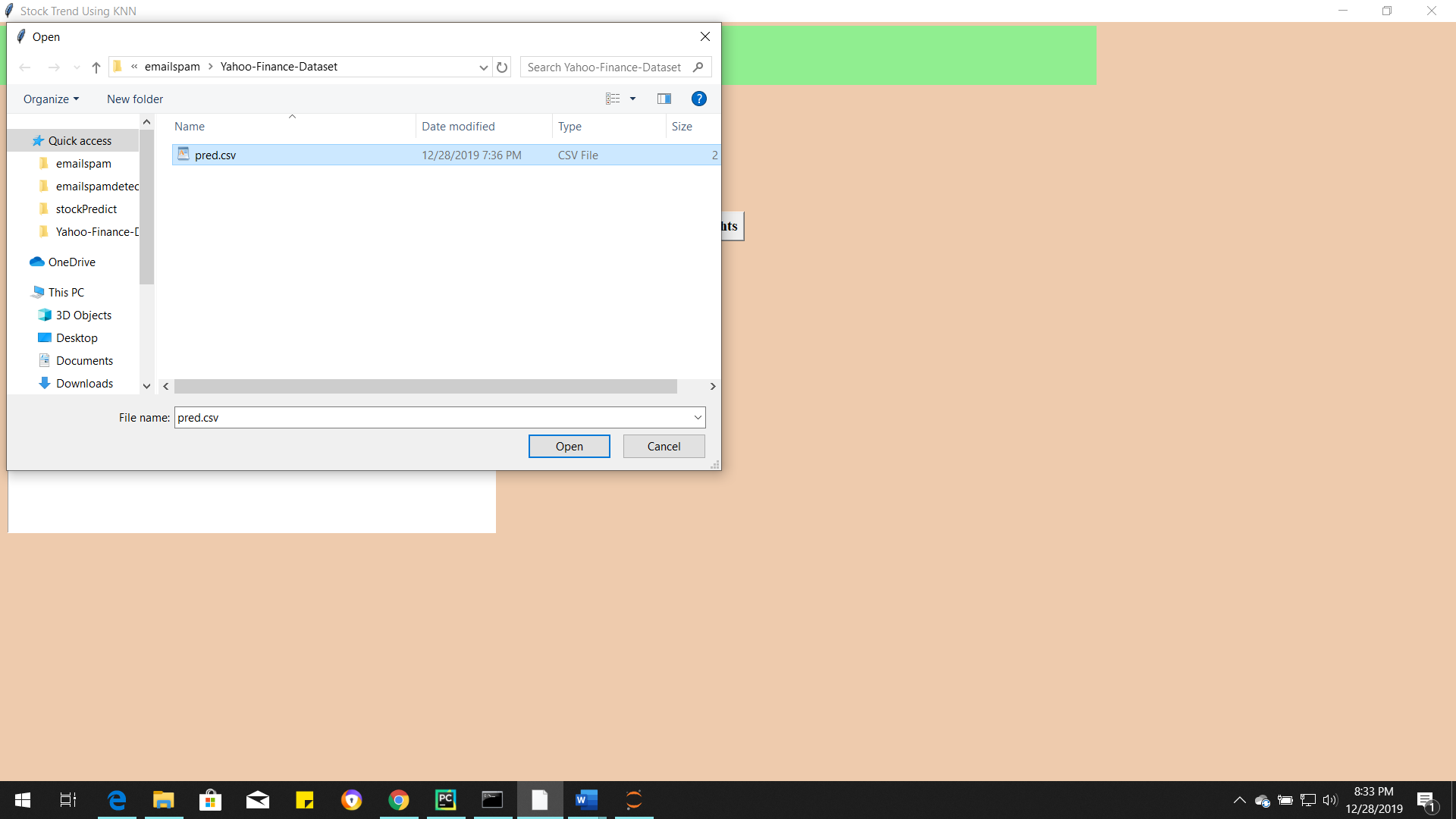
Now click on ‘Run KNN with Uniform Weights’ to generate KNN model with uniform weights and calculate its model accuracy

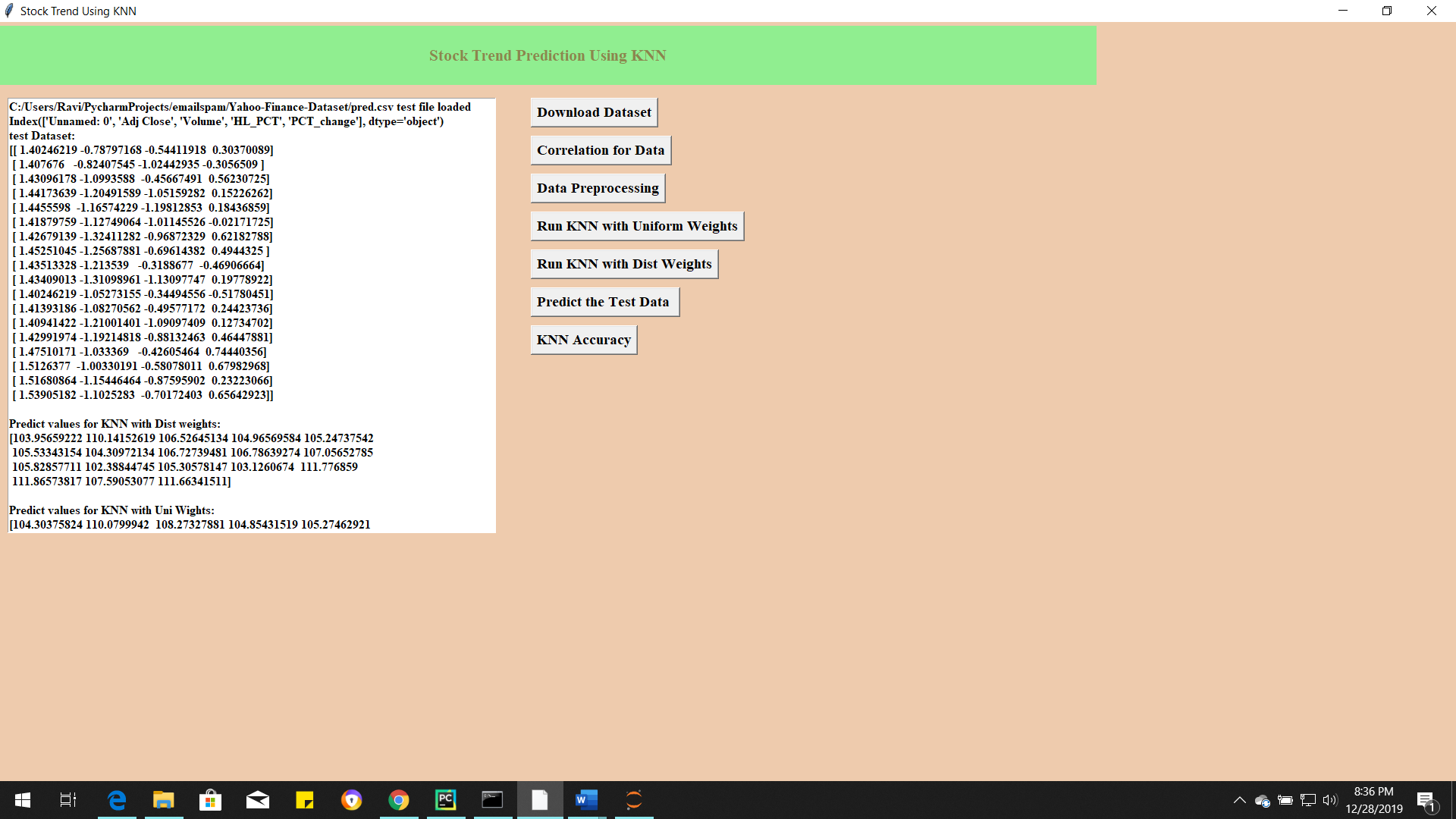


In above screen we can see with KNN with uniform weights got 96.8% accuracy, now click on ‘Run KNN with distance weights’ to calculate accuracy

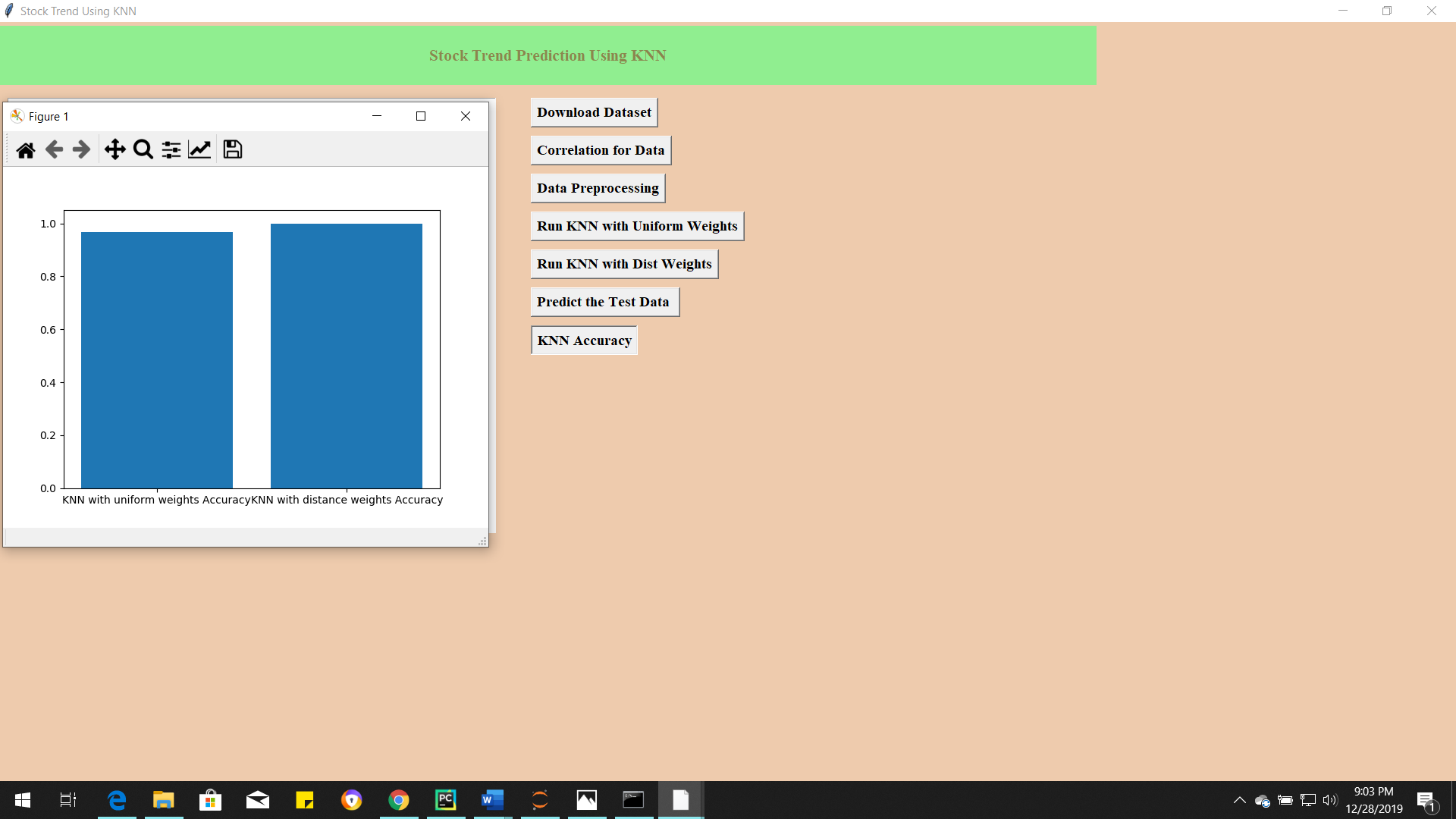
In above screen we got 100% accuracy, now we will click on ‘Predict Test Data’button to upload test data and to predict whether test data stock market for both models.

accuracy score (>0.95) for most of the models.



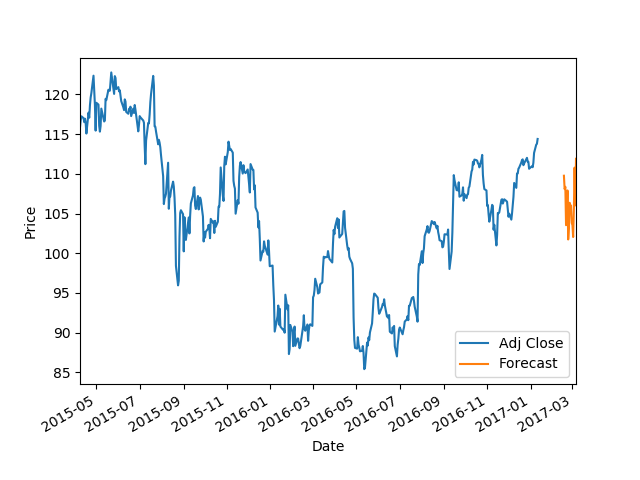


In above screen for each test data we got forecast values for Apple Stock for each test record. Now click on ‘KNN Accuracy’ button to save the predicted values for each model save in the local directory and Accuracy comparison to both the models



From above graph we can see that distance weights has little bit better better accuracy compare to Uniform weights, in above graph x-axis contains algorithm name and y-axis represents accuracy of that algorithms

# **Plotting the Prediction for KNN with Uniform Weights:**



# **Plotting the Prediction for KNN with Distance Weights:**

