**CORIZO JANUARY COHORT 2024 PROJECT- 2**

**Name- Sarvesh Tikekar**

**Title- Stock Market analysis using Data Science and Machine Learning Models**

**Introduction**

In this project, we have built a stock market analysis system using machine learning with Python, using a KNN Classifier and Regressor.

The parameters used for building the system are:

1. Date of the stock
2. Open Price of the stock
3. Lowest price of the stock on that date
4. Highest price of the stock on that date
5. Closing price of the stock on that date

Some other parameters are- Adjusted Closing price and Volume of the stocks.

**GETTING THE DATA/ DATASET FOR THE SYSTEM**

The dataset used for the analysis is given below:



**DATA ANALYSIS AND VISUALISATION**

1. First, we found out the total no. of rows and columns in the dataset using the **shape()** function. The output was: (1009, 7)
2. Next step, was to find the total number of rows, having missing values, which was found out using the **isna().sum()** function.
3. After implementing the above function, we came to know that there are no missing values in the dataset, so we proceed with the next upcoming steps.
4. We then, tried to find out the co-relation between the closing price of stock and rest of the parameters, using the **corr()** function.
5. Also we used the **describe()** function, so it gives an overview about the dataset, i.e. mean, mode, median, etc.
6. Then, for visualisation, we plotted a graph of the ‘Close’ column to find out the trend in the Closing price of the stock on any particular date.

**PREPROCESSING THE DATA**

1. Further, we created 2 columns in the dataset, which are ‘Open - Close’

(Difference between Opening and Closing Price of the stock on the same date) and ‘High-Low’ (Difference between Highest and Lowest Price of the stock on the same date).

1. From the above point, we can observe that the values for the ‘Open - Close’ can be of two types:
2. Positive value (+ve) which means Closing price of the stock has increased.
3. Negative value (-ve) which means Closing price of the stock has decreased.
4. Based on this observation, we have classified the stocks on each day, into different parts, with the help of 2 types of signals:
5. +1 signal 🡪 If predicted close price of next date is greater than closing price of current date, it’s a signal to buy the stocks on the current date.
6. -1 signal 🡪 If predicted close price of next date is lesser than closing price of current date, it’s a signal to sell the stocks on the current date.
7. This signal data was then inserted into our original dataset as well.

**PREPARING THE MACHINE-LEARNING (ML) MODEL**

1. To prepare our model, we split the data first into two parts:
2. The Training data, used to train our ML model.
3. The Testing data, used to test our ML model.
4. For this splitting, we used the **train\_test\_split()** method from **sklearn** module. The inputs given are the 2 dataset parts we formed while pre-processing the data.
5. For the test-size, we kept is as 0.30 or 30% so to increase the accuracy of prediction of the model, with a random state as 50 so whenever we train the model the result would be the same i.e. same values in train and test datasets.

**TRAINING THE MACHINE LEARNING MODEL**

1. The Machine Learning model was built using a **KNN Classifier** as well as a **KNN Regressor.**
2. The K-Nearest Neighbours (KNN) classifier is a simple, instance-based learning algorithm that is widely used for classification tasks. It assumes that similar instances will exist in close proximity to each other.

**How It Works:**

* 1. **Training Phase:** KNN does not have an explicit training phase since it simply stores all the training data.
  2. **Prediction Phase:** To classify a new data point, the algorithm:
     + Calculates the distance (e.g., Euclidean distance) between the new data point and all the training data points.
     + Identifies the K nearest neighbours to the new data point.
     + Determines the majority class among the K nearest neighbours.
     + Assigns the new data point to the majority class.

1. The K-Nearest Neighbours (KNN) regressor is an extension of the KNN algorithm for regression tasks. Instead of predicting a class label, it predicts a continuous value.

**How It Works:**

* 1. **Training Phase:** Similar to the KNN classifier, there is no explicit training phase.
  2. **Prediction Phase:** To predict the value for a new data point, the algorithm:
* Calculates the distance between the new data point and all the training data points.
* Identifies the K nearest neighbours to the new data point.
* Computes the average (or sometimes weighted average) of the values of the K nearest neighbours.
* Assigns this average value as the prediction for the new data point.

1. For optimal operations, the number of neighbours for the classifier were chosen as 25 for the Classifier and 20 for the Regressor.
2. A KN-classifier “Knn” was created, and using a function “**GridSearchCV**”, we prepared our machine learning model. **GridSearchCV** is basically considering all the combinations of the candidates in the set, to find the best parameters. The parameters given were the KN-classifier we initialised above, along with the number of neighbours.
3. Additionally, the cross validation (cv) value was set to 5, to optimally assess the performance of each combination of parameters.
4. Before evaluation, we fitted our ML model, with the X-type training data as well as Y-type training data.

**EVALUATING THE MODEL**

1. First, we passed the Y-type train values (The signal data for the stocks) and the actual values of the X-type train data (data containing difference between high and low and open and close values of stock), into an **accuracy\_score()** function, and the score was in the range of 60-65 inclusive.(i.e. 60-65% accuracy)
2. Additionally, we passed the Y-type test values (The signal data for the stocks) and the actual values of the X-type test data (data containing difference between high and low and open and close values of stock), into an **accuracy\_score()** function, and the score was in the range of 50-55 inclusive.(i.e. 50-55% accuracy)

**MAKING PREDICTIONS**

1. We again predicted the signal values using the **predict()** function, on our model and compared it with our actual values. The accuracy was the same that we achieved above.
2. Now, using the KNN regressor, we also predicted the close value for a particular date. The method used to create the Regressor was the same compared to the KNN classifier, but instead of signal data, it returned us the predicted closing value for the stock at any particular date.

**GITHUB LINK FOR THIS PROJECT**

**https://github.com/T-SarvesH/DataScience-Internship-Project/blob/master/StockMarketPrediction.ipynb**