Prediction of Closing Price of Various Stocks

Name:	**Aayush Verma**
Registration No./Roll No.:	20003
Institute/University Name:	IISER Bhopal
Program/Stream:	EECS
Problem Release date:	January 12, 2023
Date of Submission:	April 16, 2023

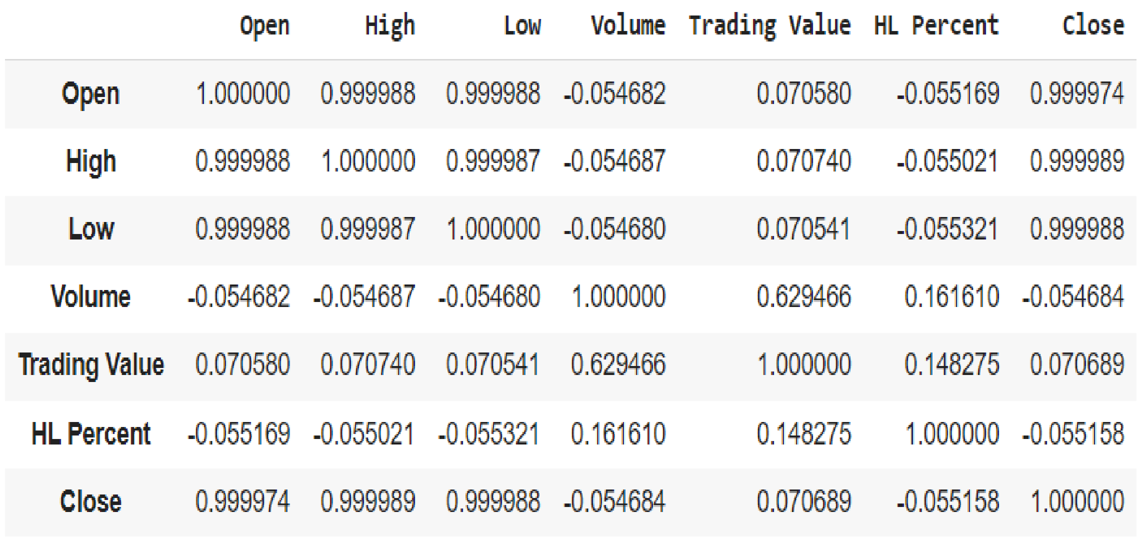
1 Introduction

Stock price prediction is a challenging task that has attracted a lot of attention in recent years. With the advent of machine learning and deep learning algorithms, it is now possible to develop predictive models that can analyze historical stock data and forecast future prices with reasonable accuracy. In this project, we aim to develop a stock price prediction models using machine learning techniques and evaluate its performance on real-world stock market data.

The data we have comprises the Open, Low, High, Close, and Volume of several companies from various industries. Data from one day is represented by each row. Our goal is to develop a model that accurately predicts the closing price to minimise our losses and maximise our profits. We will try to analyse the dataset and apply various algorithms to predict the closing price.

2 Data Visualisation and Feature Selection

The first task will be to clean the data provided to make it suitable for Supervised ML methods that have been mentioned below. First we checked for any NaN values and remove them.



	Open	High	Low	Volume	Trading Value	HL Percent	Close
Open	1.000000	0.999988	0.999988	-0.054682	0.070580	-0.055169	0.999974
High	0.999988	1.000000	0.999987	-0.054687	0.070740	-0.055021	0.999989
Low	0.999988	0.999987	1.000000	-0.054680	0.070541	-0.055321	0.999988
Volume	-0.054682	-0.054687	-0.054680	1.000000	0.629466	0.161610	-0.054684
Trading Value	0.070580	0.070740	0.070541	0.629466	1.000000	0.148275	0.070689
HL Percent	-0.055169	-0.055021	-0.055321	0.161610	0.148275	1.000000	-0.055158
Close	0.999974	0.999989	0.999988	-0.054684	0.070689	-0.055158	1.000000

Figure 1: Correlation Matrix

We have tried adding two new columns ”Trading Volume”=Open $\times$ Volume and ”HL Percentage”=(HighLow)/High. Here trading volume indicates strong interest and liquidity in the market, while low trading volume indicates the opposite and the high-low percentage or H-L percent is a measure of volatility in the stock market. It is the difference between a stock's highest and lowest prices for a given period, divided by the closing price for the same period. Both of them are useful indicator of the market's price movement level, so we tried them as well by creating a copy of our data set by adding two of the columns. Though after applying some algorithms, it was found that there was not much change after adding the two data sets, which can be explained by the correlation matrix.

As we can see Open, High, Low, and Close are highly correlated. Also, since these values are so close, we will obtain high accuracy. We have defined some new metrics and evaluated the models to distinguish between wrong and correct predictions better.

New StandardScaler class instances are used to standardize data by removing the mean and scaling to unit variance. we used the fit transform method on each scalar object with the respective input (fo1, fo3, f1, f2, and f3). This standardizes the data for each data set in place and stores back in their respective variables.

3 Methods

We applied Linear Regression, Ridge Regression, and Lasso Regression with the help of *sklearn*. Two linear regression analyses were performed on different sets of data, and the corresponding errors were calculated using the previously defined functions. The $error_list$ function was used to calculate and print the error metrics (RMSE, MAPE, MBE, and R2 Score) for the Linear Regression model using $f1$ features and $f2$ target data.

Since Linear Regression did not have many parameters, hyper parameter was done for Ridge and Lasso Regression only. Regularization techniques like Ridge and Lasso regression helps to avoid overfitting by adding a penalty term to the regression objective function. The penalty term controls the complexity of the model by shrinking the coefficients of the predictors towards zero. Thus, we used them in order to make sure there are no problems of overfitting and multicollinearity which might occur in Linear Regression. For these 3 techniques we tried both datasets. Though,it was found that there was not much difference in the results.

We have created a GridSearchCV object for ridge and lasso regression which is used to find the optimal values for the hyperparameters of the ridge regression. This object takes the Ridge model, the parameters dictionary, and cv=10 which is the number of cross-validation folds.

We then applied KNeighbors Regressor. It performs a loop over a range of values for the number of neighbors (K) in a K-Nearest Neighbors (KNN) regression model. For each value of K, the code trains a KNN regression model on the training data (fo_{train} and $f2_{train}$), makes predictions on the test data (fo_{test}), calculates the root mean squared error (RMSE) between the actual and predicted values using the *math.sqrt* and $mean_squared_error$ functions from the *math* and *sklearn metrics* modules, respectively, and appends the RMSE to a list called *rms*. After the loop, the code creates a line plot using *matplotlib.pyplot* to visualize the relationship between the number of neighbors and the RMSE.

First, the code was used on multiple values of K, and then the graph of K versus mean squared error was plotted. However, it was observed that the plot was not smooth because the Volume had extremely high values, which was affecting the results. Therefore, the data was scaled, and the algorithms were reapplied. There was a slight difference in the previous algorithms. As for KNeighbors Regressor, the following graph was obtained:

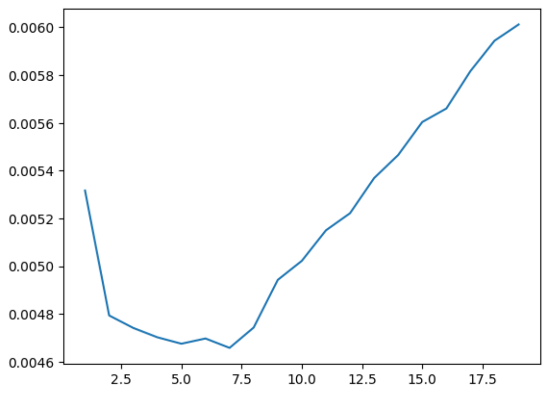


Figure 2: Number of Neighbors vs. Mean Squared Error

4 Evaluation Criteria

To check the performance of our models, we have used four evaluating criterion.

- r2-score: R squared in regression acts as an evaluation metric to evaluate the scatter of the data points around the fitted regression line.
- RMSE: Root mean square is the measures the average difference between values predicted by a model and the actual values. It provides an estimation of how well the model is able to predict the target value (accuracy).
- MAPE: Mean Absolute Percentage Error is the mean of all absolute percentage errors between the predicted and actual values.
- MBE: Mean Bias Error is the mean of the difference between the predicted values and the actual values.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(A_i - P_i)^2}{n}}$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{|A_i - P_i|}{A_i}$$

$$MBE = \frac{1}{n}\sum_{i=1}^{n}|(A_i - P_i)|$$

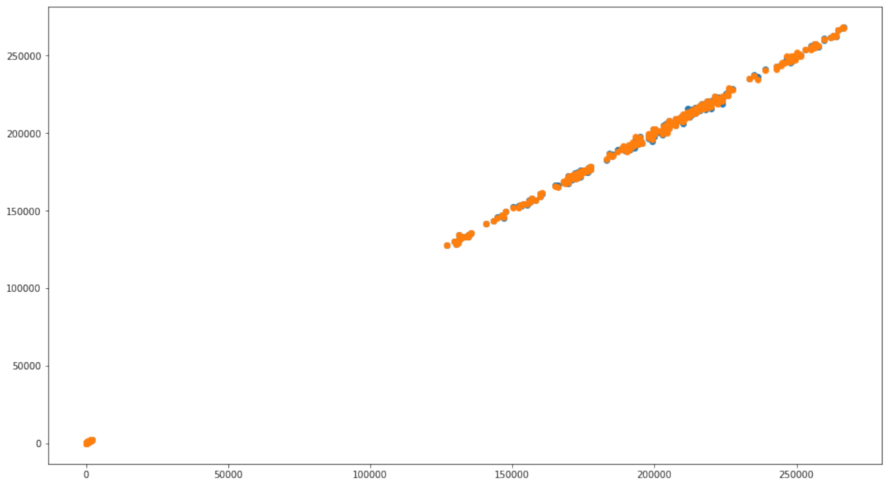
5 Analysis of Results

The following table shows best results for each model. Here we have represented four types of error scores [r2, RMSE, MAPE and MBE] to evaluated the models on two data sets having 4 and 6 features (with Trading Volume and HL Percentage) respectively.

Table 2: Performance Of Different Regressors

Model	Number of features	r2-score	RMSE	MAPE	MBE
Linear	4	0.999991285	0.002985876	0.00019927	0.00027001
Linear	6	0.999991285	0.002985848	0.0002032	0.00027043
Ridge	4	0.999991285	0.002985861	0.0002	0.00027008
Ridge	6	0.999991285	0.002985834	0.00020171	0.00027026
Lasso	4	0.999991285	0.002985861	0.0002	0.00027008
Lasso	6	0.999991285	0.002985834	0.00020171	0.00027026
Random Forest	4	0.999974232	0.005134319	0.00024235	0.0004292
K-neighbours	6	0.999978918	0.004643972	0.00095669	0.00047497

- Here, we can observe that the that Linear Regression with 4 features performs the best gives lowest error overall.
- r2-score is not a good evaluation criteria as it gives almost similar values in all the models.
- Also MAPE is more important in our evaluations as compared to other metrics because, MAPE considers the percentage difference. A $2 error will have more significance for stock with price $20 than a $200 stock.
- To visualize the Linear Regression predicted output we plotted the Test vs Predicted data on a scatter plot.



The test data is represented by blue dots, while predicted closing prices are indicated by orange points. We can clearly observe that the test data is predicted very accurately and the predicted data values are almost overlapping the test data. Thus our linear regression model is trained very well. The code for this project is uploaded on the GitHub.

6 Discussions and Conclusion

Stock market closing price prediction is using various analytical techniques and statistical models to forecast the future price of a stock at the end of a trading day. The main objective of our study is to forecast the closing stock price with improved efficiency using machine learning and deep learning models to minimize the risks for investors and policy-makers.

While working on the data set, we included two extra features, namely HL Percent and Trading Volume, using the information from the given features. But the selection of these features did not significantly improve the result of the Learning models. Scaling down all the feature values using hyper-parameter tuning greatly helped to increase the efficiency of the models (specially the volume feature which had too much variation compared to other features).

Linear Regression has the best outcome among all the learning models that were used in the study and we were able to predict the test data with very high accuracy and low error. However, it is crucial to consider a wide range of factors and parameters, in addition to past price data, to enhance the accuracy of machine learning-based prediction models for stock prices. They may include market sentiment, global economic indicators, regulatory changes, and technological advancements related to global market. A more refined data set with defined classes for each of the 88 stocks would have given better possibilities to under and predict the closing prices.

7 References

1. https://scikit-learn.org/stable/
2. https://finance.yahoo.com/