Predicting the Change of Close Price from Open Price of the Top 5 Cryptocurrencies in the Market

James Zhao

2024-05-02

Contents

List of Figures

List of Tables

Abstract

Predicting how close the price changes from the open price of cryptocurrencies is critical to all investors for selecting the appropriate investment strategy to make profits from the non-stationary cryptocurrency market. In this report, we discuss the process of predicting the price difference in the cryptocurrency market. Data are collected from the real market which includes historical cryptocurrency prices for all tokens, but we only focus on analyzing the top 5 cryptocurrencies by market capitalization, which represents the total value of the tokens that have been mined, in the current market. We detail the procedures of analysis and compare the prediction accuracy of three different methods. Based on our chosen performance indicator (RMSE),random forest, elastic net, and bagged forest possess the best predictive accuracy. We discuss our findings, and the limitations of the study, in the following paper.

Section 1: Introduction

In this report, the primary goal is to make predictions of the difference between the close and open prices of the top 5 cryptocurrencies (ranked by their market capitalization which represents the total value that has been mined) using three methods, comparing their performance with respect to the chosen metrics RMSE and determining the best prediction model/algorithm which can be used in the real market. It is worthwhile to perform the prediction task because the cryptocurrency market reveals signs of continuing growth in values and potential opportunities that leads to the urgent need for an accurate prediction model which contributes to investors making profits.

In brief, the goal of this report is to predict the the amount of change in close price from the open price of selected cryptocurrencies using selected features in the data. The data set we use in this report is named "Every Cryptocurrency Daily Market Price" (we will just refer to it as data in the rest of the report). We will compare the performance of three prediction methods (regression models, KNN algorithm, and tree and forest models) and determine the best prediction model/algorithm according to the Root-Mean-Square Error (RMSE). About the data set, there are 942297 observations and 13 columns in the raw data, which contains all historical cryptocurrency information and prices of the top 5 cryptocurrencies. Moreover, we have information on the name of the cryptocurrency, the date of trading in the format of yyyy-mm-dd, the open price of the cryptocurrency each day in US dollars, the highest trading price of the cryptocurrency each day in US dollars, the lowest trading price of cryptocurrency each day in US dollars, the close price of the cryptocurrency each day in US dollars, and the volume of trade each day for each cryptocurrency. Although these may not include information on all columns, they are accessible through the link: https://www.kaggle.com/datasets/jessevent/all-crypto-currencies?resource=download.

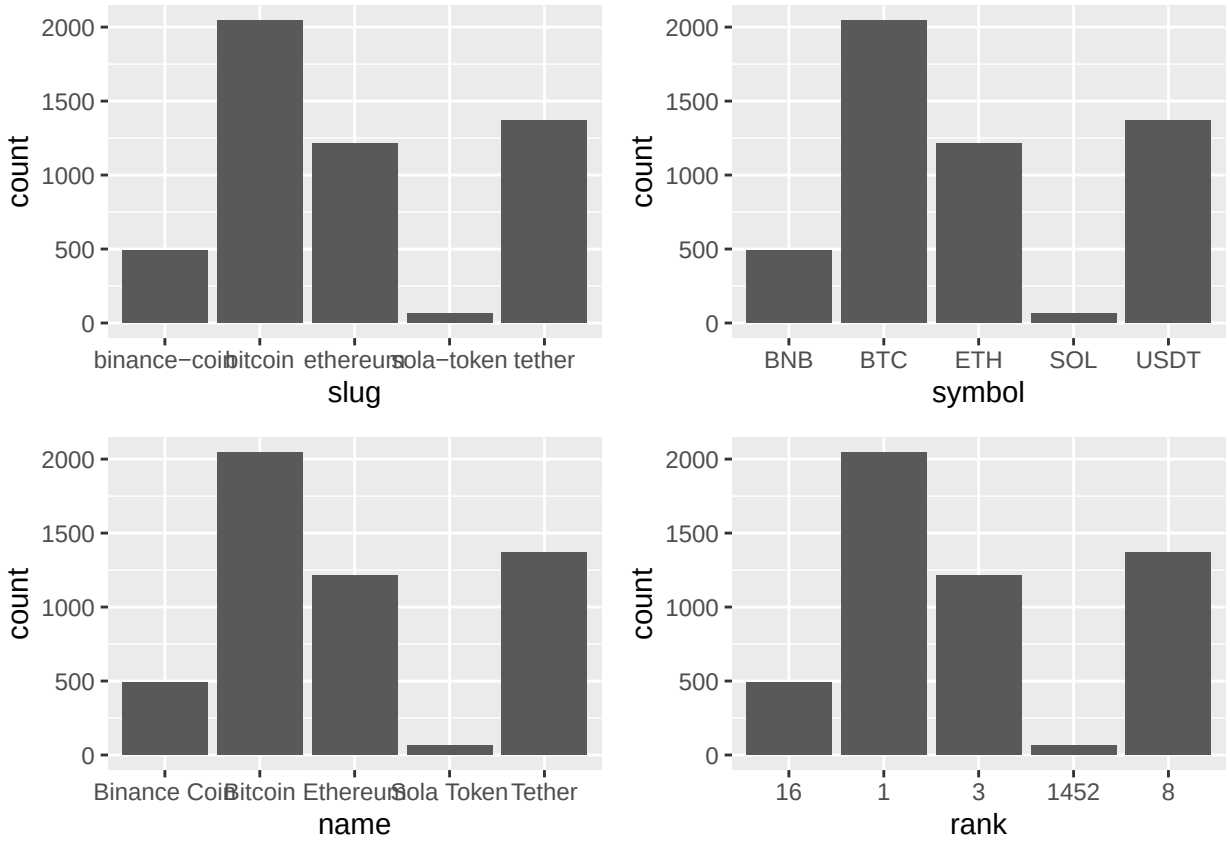
Section 2: Data Cleaning and EDA

Before our analysis, it is important to take a look at the raw data set and make sure it is ready for analysis. Firstly, we will take a look at the data set in general:

Table 1: Raw Data Dimension and Missingness

Missingness	Rows	Columns
0	942297	13

Table 1 provides a general description of the data. We can see that the data set has 942297 observations and 13 columns, and there is no missing value in the data set. According to our research question, we will focus on only the top 5 cryptocurrencies in this report, so we will subset a new data set from this raw data set and include only the rows of these cryptocurrencies. We wouldn't consider using the column of "date" either because we are working on prediction instead of forecast. Besides, since our response variable is the



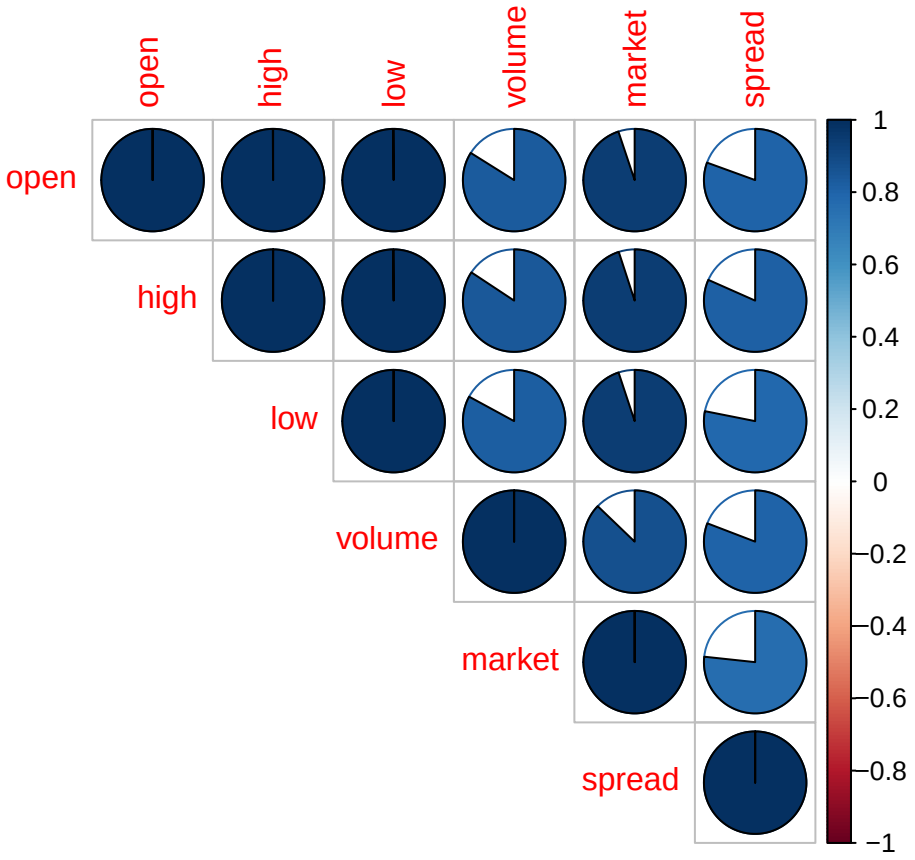
1: Figure 2.1 - Distributions of four categorical variables in data

difference between the close and open price of the cryptocurrency (more specifically close price - open price), we will create another column using the close price subtracting the open price.

As we take a glance at the distributions of our four categorical variables in Figure 2.1, we notice that, for each categorical variable, there are exactly 5 sub-categories and also equal number of rows for each sub-category. This suggests that these four categorical variables give us the same information. Hence, we would only keep one of the four variables in our data set.

Additionally, based on the column description of the data set, we know that the close ratio is computed using all of the high, low, open, and close prices. However, we will not know what the close price is when implementing real-world predictions. Thus, we will exclude this column from our data set too.

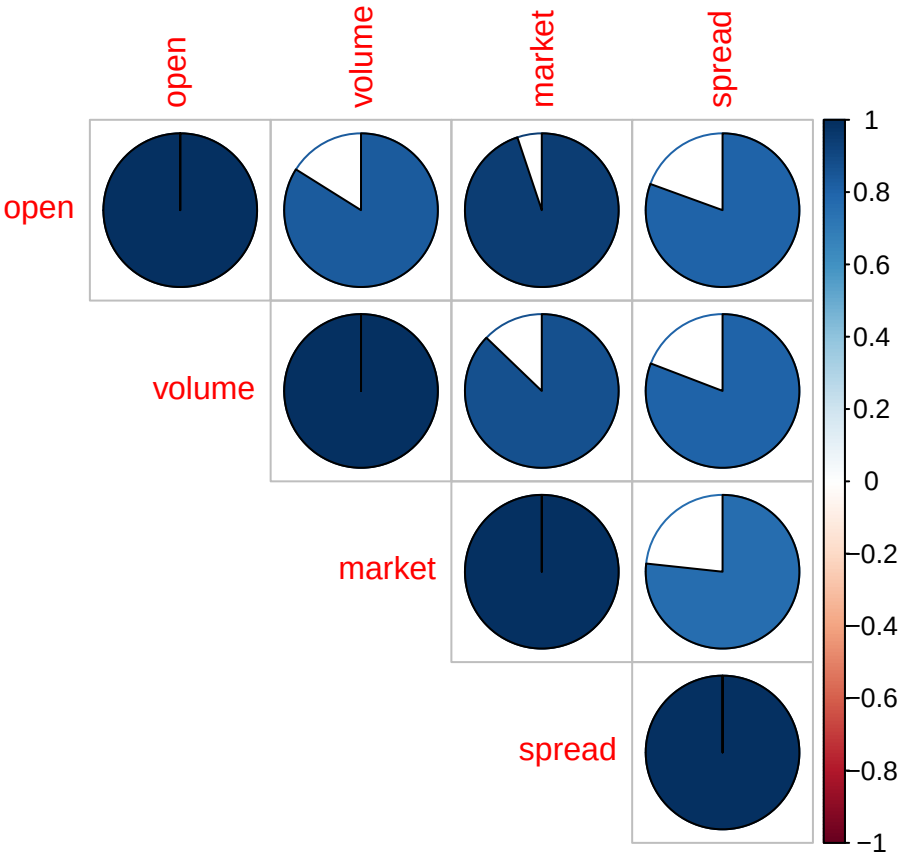
Furthermore, we explore the correlation between all numerical variables except our response variable to check whether multicollinearity occurs:



2: Figure 2.2 - Correlation of possible features

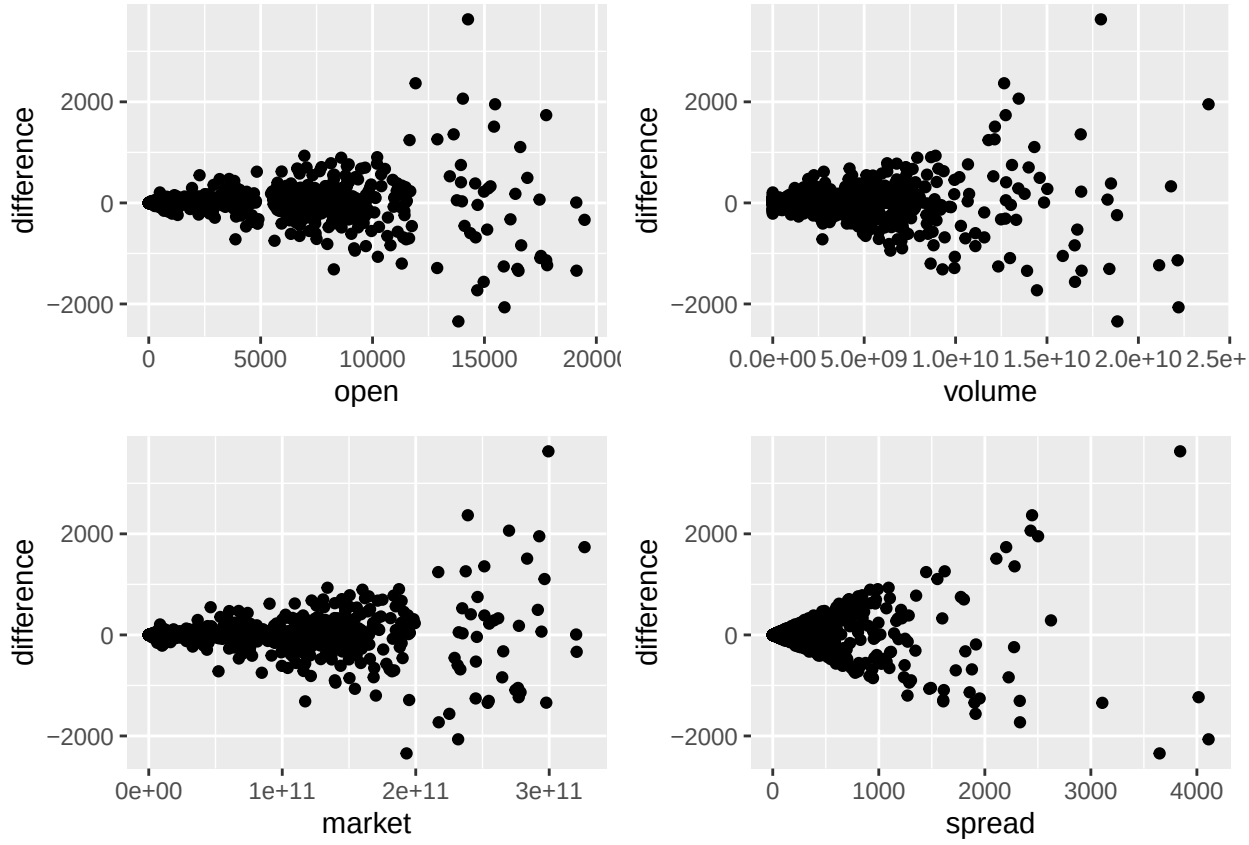
According to Figure 2.2, we can tell that the open price, the high price, and the low price are perfectly correlated, which is a severe problem for regression analysis. In fact, based on the column description of the data, we know that the price spread is calculated by using the highest price minus the lowest price. Since we are able to obtain similar information from one column than two columns that are perfectly correlated with other features, we can simply remove those two problematic columns from our data. Then, we will plot the correlation of our numerical variables again to check whether we are still struggling with a high correlation issue.

Figure 2.3 suggests that we no longer have any columns that are perfectly correlated, but we are left with some columns with high correlation for which we will pay more attention. Nevertheless, we are still one step from performing and assessing prediction. Because we intended to train regression models, it is necessary to



3: Figure 2.3 - Correlation check after removing problematic colums

check the shape assumptions between our features and the response variable:



4: Figure 2.4 - Shape assumption for regression

Based on Figure 2.4, we observe some fan-shaped patterns between the numerical features and the response variable. This may result from the high correlation problem between our features. Now we are ready to start training prediction models/algorithms and assess their performances. Our final data dimensions are provided below in Table 2.

Table 2: Dimension of Final Data Set

Rows	Columns
5182	6

Section 3: Ridge Regression, Lasso, and Elastic Net.

Section 3.1: Introduction

The first prediction method we use in this report is elastic net because, theoretically, the high correlation between our features would result in large estimates of coefficient variance, which would be problematic for further analysis.

Section 3.2: Method

In least-square linear regression, we get coefficient estimates by minimizing the residual sum of square (RSS), which is the square difference between the true response value and the estimated response value. However, given the fact that multicollinearity exists, we employ another regression model that chooses the coefficient estimated by minimizing $RSS + \lambda \sum ((1-\alpha)\hat{\beta_j}^2 + \alpha|\hat{\beta_j}|)$, where the latter term is the penalty term we are trying to minimize as well. When choosing coefficient estimates of elastic net model, there are two tuning parameters called lambda and alpha. Lambda controls the amount of shrinkage we want on coefficient estimates. Besides, since we know the penalty term of ridge is $\lambda \sum \hat{\beta_j}^2$ and that of lasso is $\lambda \sum |\hat{\beta_j}|$, elastic net is just lasso when alpha is one and it is ridge when alpha equals zero. Hence, alpha balances between the coefficient estimate shrinkage of ridge and selection property of lasso in elastic net.

In general, we will create one sequence of possible values for each tuning parameter, lambda and alpha, and find the value of the two parameters that generates the lowest test RMSE using 10-fold cross validation. For each possible value of alpha in the sequence, we run 10-fold cross validation for each possible value of lambda and compute the test RMSE for each lambda. Then we find the lambda with the lowest test RMSE and store the result in a data frame in the row with the corresponding alpha value. After we run through all possible values of alpha, we find the lowest test RMSE in the data frame and output the information in the row. Thus, these are our final tuning parameter values and test RMSE of the model.

Section 3.3: Result

After training the elastic net model, as well as ridge regression and lasso that are trained simultaneously, we examine their predicting performances using our chosen metrics RMSE. Furthermore, to emphasize how elastic net solves high correlation between features and improves predictive accuracy, we fit linear regression and compare its RMSE value with the results of elastic net.

Table 3: Prediction Accuracy Comparison of Regression Models

Model	Tuning_Parameter	Test_RMSE
Elastic Net	$\alpha = 0.45$, $\lambda = 0.5$	145.7330
LSLR		148.4929

According to Table 3, we get alpha being 0.45 and lambda being 0.5, which suggests that, in terms of minimizing RMSE, the elastic net outperforms ridge regression and lasso since alpha doesn't take the value of either 0 or 1. To re-emphasize, elastic net becomes lasso when alpha is 1 and it is ridge when alpha equals 0. Lambda equals 0.5 prevents us from getting exploded coefficient estimates. Since elastic net not only shrinks coefficient estimates but also performs selection on features to suggest a better fit and easier-interpreted model, we compare the coefficient estimates between least-square linear regression and elastic net:

Table 4: Coefficient estimates

	LSLR	Elastic.Net
Intercept	5.7243683	6.442173
nameBitcoin	8.3331998	5.922820
nameEthereum	-48.0744279	-43.609252
nameSola Token	2.0250205	0.000000
nameTether	11.0783614	8.105885
open	-124.3327742	-109.424612
volume	-31.4473074	-27.482976
market	149.5105593	132.954483

	LSLR	Elastic.Net
spread	0.6562987	-1.305064

From Table 4, compared with coefficient estimates of LSLR, we find that the coefficient estimate of the subcategory Sola cryptocurrency of feature name is being reduced to zero as a result of elastic net performing selection, but coefficient estimates of other features don't change much. Moreover, since we have already scaled all numerical features before we train models, the coefficient estimates also serve as an indicator of feature importance. Based on the coefficient table, we observe that market capitalization has the highest feature importance, which means that it contributes toward the prediction generated from elastic net the most.

Referring back to Table 3, our test RMSE of Elastic Net is 145.733, which indicates that, on average, our prediction of the difference between the close and open price differs from the true price difference by 145.733 US dollars.

In sum, elastic net works well when there are highly correlated features, yielding lower variances of coefficient estimates than LSLR. It is also very efficient since, when we take values of alpha from 0 to 1 in sequence to run elastic net, we also run ridge and lasso simultaneously. In addition, elastic net outperforms ridge as it performs feature selections and lasso as it does some selection but less than lasso intended to do. However, the computational complexity could still be problematic when we have a large data set because there are two tuning parameters to monitor.

Section 4: KNN

Section 4.1: Introduction

Other than regression models like LSLR and elastic net that are bounded by shapes, K nearest neighbor (KNN) is another useful and more flexible algorithm for prediction since it is not restricted by shape assumptions. Particularly, KNN doesn't predict using parameters of a mathematical equation and can be more flexible to portrait the patterns of training data.

Section 4.2: Method

Similar to Section 3, we still apply 10-fold cross validation to obtain the test RMSE and compare the results. The main difference is the algorithm we use. For KNN, the first step is to convert all categorical features into binary variables, which allows us to compute distance measures (we use Gower's distance in this report) between rows using chosen distance metrics. After the conversion, we run 10-fold cross validation on KNN and get the training and validation data in each loop. How KNN predicts is that we compute the distance measure to determine how similar a row in the validation data is to all rows in the training data. Then we pull out the K rows from the training data that are closest to the row in the validation data by distance measure. Lastly, we take the average value of the response variable for these K selected rows as our prediction for the row in the validation data. Since we are given a large data set and it would be computationally expensive to run for a range of K values and observe each individual's performance, we determine our K to be the square root of the number of rows, which in this case is 72.

Section 4.3: Result

After implementing KNN for prediction and evaluating its predictive accuracy using the chosen metrics RMSE, our results are exhibited in the following table.

Table 5: KNN Prediction

Method	K	Test_RMSE
K Nearest Neighbor	72	149.5483

According to Table 5, we notice that when K=72, our test RMSE is 149.5483. This indicates that our predicted value of the difference between the close and open price of cryptocurrency differs, on average, by 149.5483 US dollars from the true price difference.

To sum up, it is acknowledged that KNN has more flexibility in adapting patterns in training data and is not restricted by shapes and lines like regression models. However, it is doubtful whether K=72 is the optimal value for K that generates the lowest RMSE. But in the condition with a large data set, it would be very computationally expensive to run for all possible values of K and determine the optimal K value. Regarding prediction performance, however, our result suggests that KNN performs a little worse than elastic net since it generates a higher RMSE value, which means the prediction of KNN deviates more from the truth on average than that of elastic net. This may possibly be caused by not choosing the optimal K, which would be too computationally expensive to simulate and find with large data.

Section 5: Regression Tree, Bagged Forest, and Random Forest

Section 5.1: Introduction

Finally, the tree and forest model is another powerful tool for prediction tasks but has more interpretability than other prediction models in this report. Since our response variable is numerical, we limit our choice to using only regression tree, bagged forest, and random forest. Compared to the regression model and KNN algorithm which involves prediction using mathematical equations and averaging values of the neighbors, tree and forest models treat data as clusters defined by features, and rows of data in different clusters tends to have different response values. The hierarchical structure of the regression tree reveals splitting rules and values we use for each split, which illustrates feature importance by presenting how often we use a feature for splits.

Section 5.2: Method

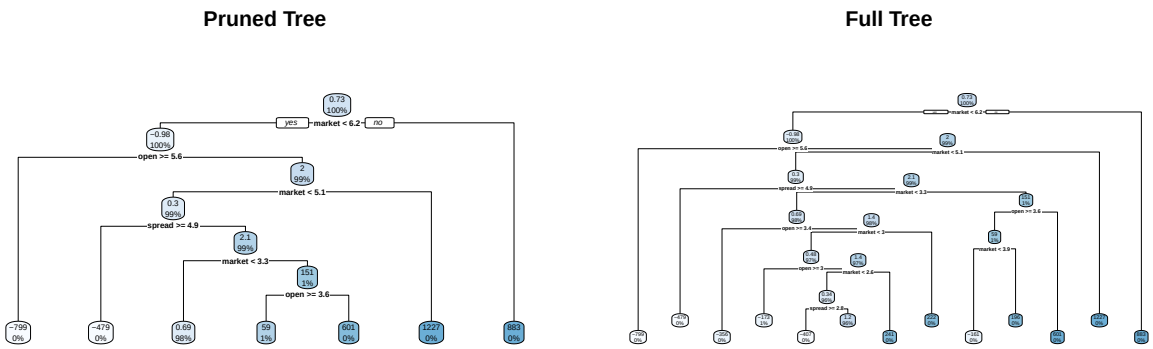
For the regression tree, we start from a root node that contains all rows in the data. We then find a splitting rule to split data in the root node into two leaves, and we decide the splitting rule depends on the training RSS for each possible split of every feature considered. We determine the split that gives us the smallest RSS. This is how we make one split from the root node, and we grow a regression tree using a process called recursive binary split, which means that all splits depend on the last split; and if we no longer have a split that has lower RSS, like what we did for one split from root node, than last split, we stop. This is how we grow regression tree to the biggest size in our belief. Then, we perform a process called pruning, which cuts leaves from the full tree we grew and checks if it improves test RMSE by applying 10-fold cross validation. As a result, we determine the pruned tree that has the lowest RMSE as our final model and output its test RMSE to compare the predicting accuracy with other models/algorithms.

On the other hand, forest models employ multiple trees for prediction, and it has each tree grows on multiple bootstrap samples to their biggest size. Since we are growing 1000 trees for the forest models in this report, we need 1000 training data, which are so-called bootstrap samples, created by rows in the data with replacement. It means that we grow trees on different subsets of the data. Then we will obtain out-of-bag (OOB) prediction using the rows from the data that are not in the bootstrap samples of each tree model, and then use these predictions to calculate the estimated RMSE. To compare model performances on the same scale, we still have to run 10-fold cross validation of these forest models.

Moreover, despite the similar process to build forest models, the main difference between bagged forest and random forest is the number of features considered during each split of every tree that grew across the forest. In this case, bagged forest considers all five features but random forest only considers 2 random features. In contrast with bagged forest, the random forest model not only grows forest faster because it considers fewer features for each split of trees but also embraces more diversity which serves as another factor for its better prediction.

Section 5.3: Result

Let's first look at how we pruned the regression tree with respect to test RMSE.



5: Figure 5.3.1 - Pruned Tree versus Full Tree

According to Figure 5.3.1, we compare the full regression tree and the pruned regression tree. We notice that there are a total of 12 splits in the full tree, whereas retaining only 6 splits in the pruned tree. Furthermore, we observe that the most frequently used splitting rule is using the values of feature market capitalization, which aligns with our finding in Section 3.3 regarding the feature importance for the elastic net. Compared with the feature importance of elastic net indicated by the scaled coefficient estimates, the tree plot illustrates the feature importance more obviously because it utilizes this feature most frequently in splitting. We will also examine the feature importance of bagged forest and random forest models later in this section and see if we get the same result.

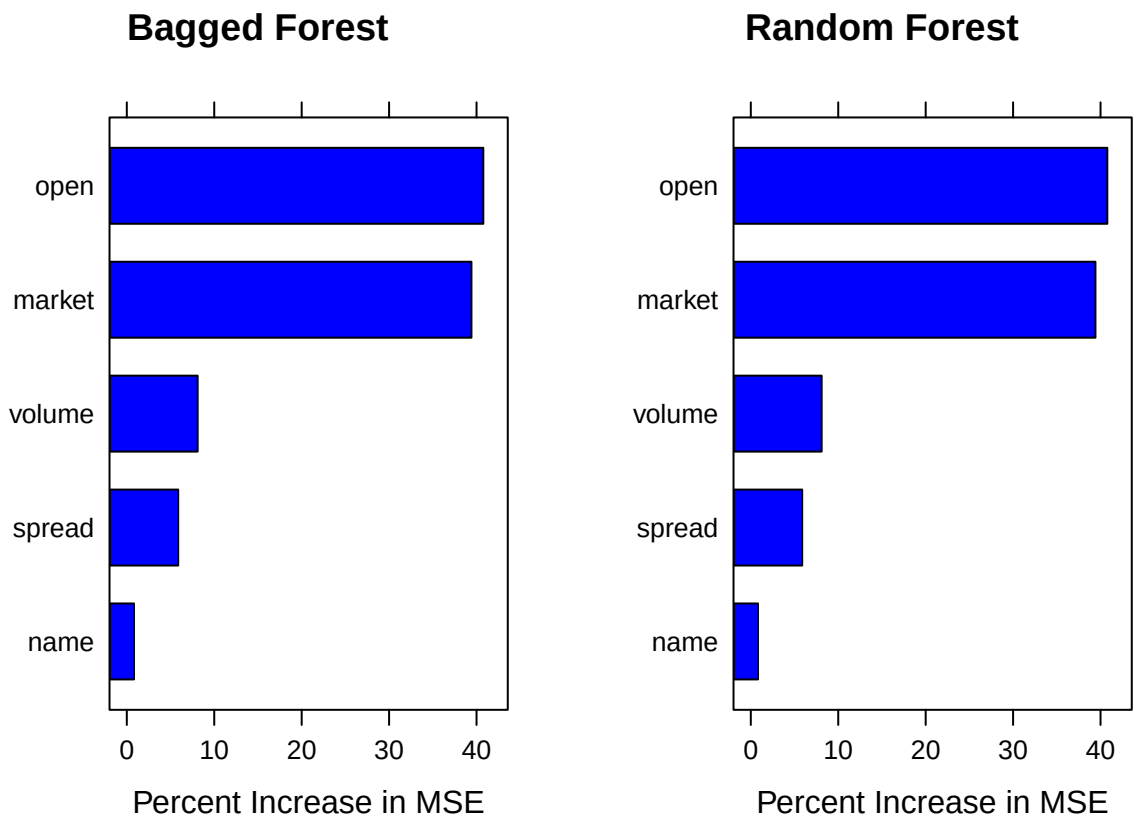
Additionally, with a fundamental understanding of our pruned tree, we run 10-fold cross validation on the pruned tree, bagged forest, and random forest, and compare their predictive ability using RMSE.

Table 6: Regression Tree, Bagged Forest, and Random Forest Predictive Accuracy

Method	Number_of_Splits	Test_RMSE
Pruned Tree	6	150.1357
Bagged Forest		148.1710
Random Forest		144.8951

Table 6 suggests that the random forest model predicts the best with the lowest RMSE value among all three models. The random forest model's prediction of the difference between the close and open price of a cryptocurrency, on average, differs from the true price difference by 144.8951 US dollars.

Besides, we are interested in what feature(s) are important for forest models in a prediction task. Hence, we make two plots showing each model's feature importance.



6: Figure 5.3.2 - Feature Importance Plot of Forests

Based on Figure 5.3.2, it indicates that features open price and market capitalization are the two most important indicators in forest model prediction as they produce the highest percent increase in OOB Mean Square Error (RMSE is just taking the square root of it). Therefore, it confirms our conclusion regarding feature importance in this report, and we claim that feature market capitalization is the most important feature in the prediction of price difference among all methods except KNN in our report.

In brief, among regression tree, bagged forest, and random forest, we believe that the random forest model possesses the best predictive ability since it produces the lowest RMSE value. It indicates that its prediction deviates the least from the truth. Also, according to the pruned tree figure and feature importance plot,

we state that both features of market capitalization and open price are crucial in predicting the price difference of cryptocurrency. Although these models generally have high prediction accuracy as well as better interpretability, it is computationally expensive to train and predict for large data sets, especially for the forest models that fit 1000 trees at the same time.

Section 6: Conclusion

In conclusion, we attempted three different prediction methods in this report for the purpose of predicting how much the close price changes from the open price of the top 5 cryptocurrencies in the market. In specific, we applied the Elastic Net regression model, KNN predicting algorithm, Regression Tree, Bagged Forest model, and Random Forest Model and assessed each model's predictive accuracy using 10-fold cross validation to compute RMSE. In general, we prefer the model/algorithm that produces the lowest RMSE value since it suggests that the prediction of the model/algorithm is closest to the true price difference. Moreover, regarding feature importance analysis for all methods, we conclude that feature market capitalization has the highest feature importance among all methods except KNN and contributes the most towards predicting the price difference of the top 5 cryptocurrencies.

Table 7: Summary of All Methods' Predictive Accuracy

Method	Test.RMSE
Random Forest	144.8951
Elastic Net	145.7330
Bagged Forest	148.1710
KNN	149.5483
Pruned Tree	150.1357

Table 7 exhibits a summary of the prediction ability of all models/algorithms. Although we find that random forest has the lowest RMSE value among all methods, the top 2 models (random forest and elastic net) according to RMSE metrics are quite close to one another, even in the cryptocurrency market where prices are sensitive measures. Therefore, we would apply random forest and elastic net in practice for predicting the price difference between the close and open prices of the top 5 cryptocurrencies, but we would not consider bagged forest, KNN, and pruned tree in practice since their predictions deviate more from the truth than that of random forest and elastic net.

Work Cited

Every Cryptocurrency Daily Market Price, Version 17. Retrieved April 8, 2024 from https://www.kaggle.com/datasets/jessevent/all-crypto-currencies/data.