University for the Creative Arts

BERLIN SCHOOL OF BUSINESS & INNOVATION

Essay / Assignment Title: Leveraging Random Forest, Regression, and KNN for Predicting Cryptocurrency Prices

Programme title: MSc Data Analytics

Name: VICTORY CHIEMEKA OKEZIE

Year: 2024

CONTENTS

Statement of compliance with academic ethics and the avoidance of plagiarism

I honestly declare that this dissertation is entirely my own work and none of its part has been copied from printed or electronic sources, translated from foreign sources and reproduced from essays of other researchers or students. Wherever I have been based on ideas or other people texts I clearly declare it through the good use of references following academic ethics.

(In the case that is proved that part of the essay does not constitute an original work, but a copy of an already published essay or from another source, the student will be expelled permanently from the postgraduate program).

Name and Surname (Capital letters): VICTORY CHIEMEKA OKEZIE

.....

Date: .....27...../....12.... /....2024....

INTRODUCTION

This report is a guide-along analysis about the necessary steps that the writer took to leverage random forest, regression and KNN for predicting crypto currency prices. It will also explain how the writer implemented and compared the different machine learning models for bitcoin price predictions. It will explain how the implementation of feature selection technique e.g. Filter methods, wrapper methods and embedded methods was able to select the best data points features that should be used for the bitcoin prediction analysis, while also explaining the results of the different feature selection methods. The link to the author full python code is Victory Okezie (December 2024). *Predicting-Crypto.* Available at*:* vbarbas03/Predicting-Crypto: Predicting Crypto Prices (Accessed: 27 December 2024).

For the bitcoin price prediction, this report developed a robust price prediction model by gathering, exploring, preparing and analyzing relevant daily bitcoin market segment data. Such as, high, low and volumes of bitcoins for up to three years. Although, bitcoin crypto currency is a relevantly staple currency, lack of standardized data sources from different exchange rates could provide data in varying formats or with inconsistency in reported data points metrics. This report will also discuss about the inherent volatility of bitcoin prices and the limitations of historical data predicting future trends. This report will suggest future work on how to explore advanced feature engineering and whether we should consider external factors like market sentiment or news analysis for a more comprehensive model.

CHAPTER ONE: DATA EXPLORATION

I imported two dataset Bitcoin and Ethereum that was provided by Usman Akhtar (September 2024). *Cryptocurrency-Dataset*. Available at: https://github.com/usmanakhtar/Cryptocurrency-Dataset/. (Accessed: 23 December 2024). I will use machine learning model for crypto currency price prediction on the Bitcoin dataset. The main relevant features to use to predict crypto currency prices are the open price, close price, high price, low price and finally volume of it during the market. The open and close price help identify the direction of bitcoin price movement during a trading period, the high and low price determines the bitcoin price range during a specific time range, and the volume represents the total trading activity in a given period (high volume often shows the strength of the price movement). See Figure 1 and 1.2 Below

```
[4]: df
```

[4]:

	Date	Open	High	Low	Close	Volume	Currency
0	2019-06-18	9128.269531	9149.763672	8988.606445	9062.045898	952850.0	USD
1	2019-06-19	9068.174805	9277.677734	9051.094727	9271.459961	131077.0	USD
2	2019-06-20	9271.567383	9573.689453	9209.416992	9519.200195	83052.0	USD
3	2019-06-21	9526.833984	10130.935547	9526.833984	10127.998047	76227.0	USD
4	2019-06-22	10151.890625	11171.013672	10083.189453	10719.981445	84485.0	USD
...	...	...	...	...	...	...	...
1146	2022-08-18	23331.542969	23567.285156	23152.455078	23222.242188	4546110.0	USD
1147	2022-08-19	23219.097656	23219.097656	20898.304688	20902.404297	13856579.0	USD
1148	2022-08-20	20899.923828	21344.845703	20864.435547	21153.019531	7139073.0	USD
1149	2022-08-21	21153.412109	21695.794922	21125.320312	21561.177734	6657571.0	USD
1150	2022-08-23	21337.318359	21412.892578	21280.458984	21303.986328	6955056.0	USD

1151 rows × 7 columns

Figure 1: Bitcoin Dataset

[5]: df2

[5]:

	date	Open	High	Low	Close	price	Currency
0	3/10/2016	11.20	11.85	11.07	11.75	4	USD
1	3/11/2016	11.75	11.95	11.75	11.95	179	USD
2	3/12/2016	11.95	13.45	11.95	12.92	833	USD
3	3/13/2016	12.92	15.07	12.92	15.07	1295	USD
4	3/14/2016	15.07	15.07	11.40	12.50	92183	USD
...	...	...	...	...	...	...	...
2353	8/19/2022	1846.52	1846.97	1607.60	1609.48	1594321	USD
2354	8/20/2022	1609.01	1654.84	1525.51	1575.60	1007240	USD
2355	8/21/2022	1575.61	1644.88	1563.92	1618.25	852071	USD
2356	8/22/2022	1618.21	1627.13	1531.91	1626.75	1044290	USD
2357	8/23/2022	1626.70	1636.32	1615.55	1623.24	1053674	USD

2358 rows × 7 columns

Figure 1.2: Ethereum Dataset

I continued to explore the dataset in my jupyter notebook file and I noticed that the Bitcoin and Ethereum dataset are both time-series dataset, the Bitcoin dataset has 1151 rows that have 7 columns while the Ethereum dataset have 2358 rows and 7 columns. Both datasets have columns with numerical values (e.g. high, low, open) that are essential for successfully predicting the crypto currency prices. Additionally, the dataset has an order that was likely influenced from the previous data (e.g. the close price of one day is related to the close price of the previous day). The only categorical values in both dataset is currency, while date is a datetime feature.

Figure 2: Bitcoin Price Data Over Time Series

I then visualize the price data over time series to identify trends, seasonality and potential outliers. See Figure 2 and 2.1. From the visualization, I noticed that Bitcoin close price had a spike increase at the middle of 2021 up to the year 2022 While Ethereum main price increase is at 2022. The seasonality of both datasets does not have any recurring pattern. Also, there are no outliers in Bitcoin price dataset but however there are 142 rows with outliers in the Ethereum price dataset. I didn't clean or remove those outliers because the outliers could be a genuine observation that reflect real occurrences in Ethereum market, removing them may hinder progress. See Figure 3.

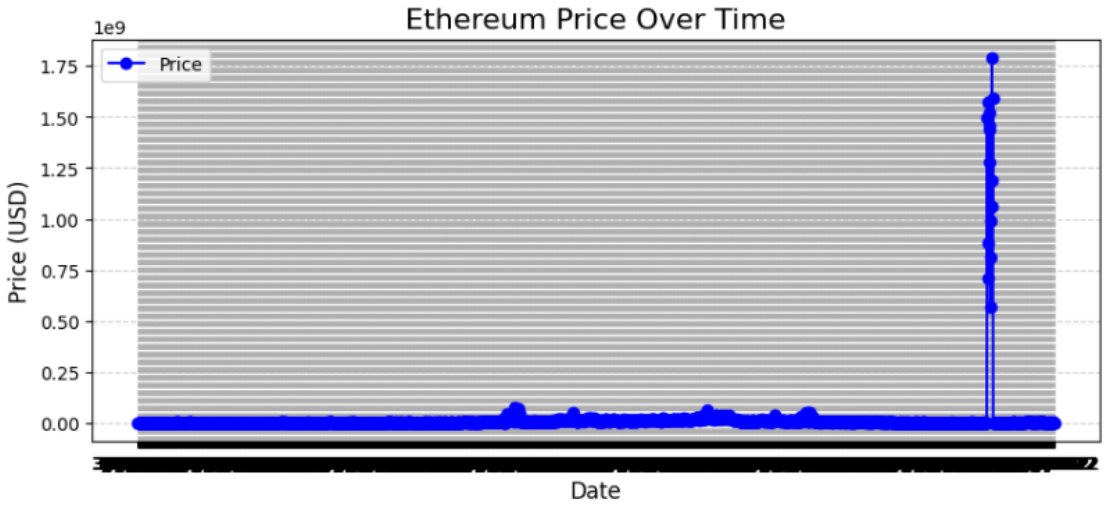

Figure 2.1: Ethereum Price Data Over Time Series

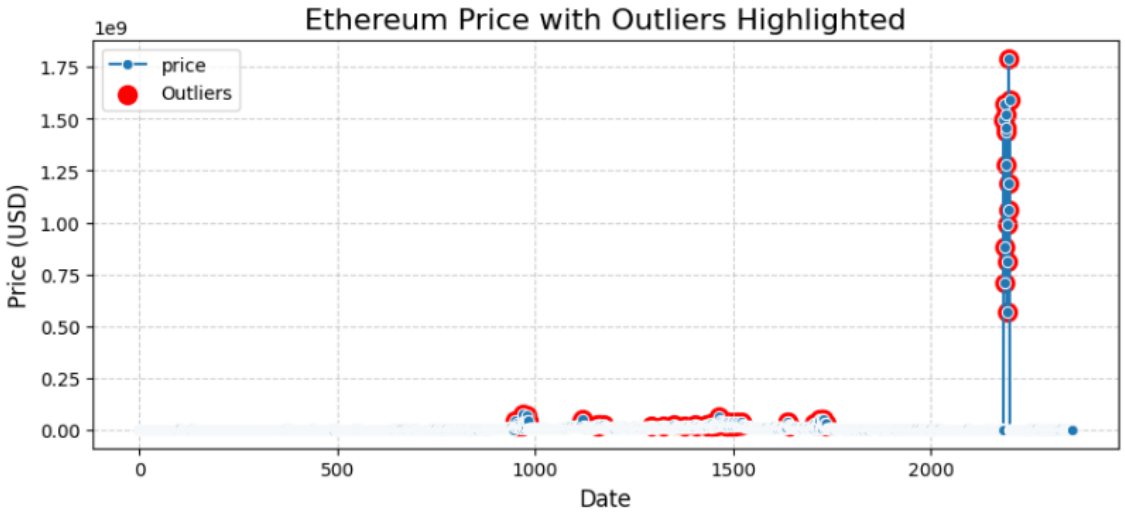

Figure 3: Ethereum Price with Outliers

The EDA (exploratory data analysis), which is the descriptive statistics for the numerical variables of Bitcoin and Ethereum dataset was found. The mean of Bitcoin close price is 2.65 while Ethereum is 8.48, all the EDA has been illustrated in my Jupyter notebook file. I analyzed frequency distribution of the currency column, which to me is the categorical feature because it is the only column without numerical values (date is a datetime feature), the result for Bitcoin is 1151 While Ethereum is 2358.

CHAPTER TWO: DATA PREPROCESSING

This is the chapter where I prepared the dataset and used feature selection to select the best future for the machine learning model. I discovered the target variable using shift(-1) to create a new column called future close, this is column with the future price for each role. While checking the new column for any missing values, I discovered that there was one empty row at the end of the column, this occurred because the column couldn't create any new instances from no previous data, I then subtracted the difference of the previous data and added it to the second to last column, which should give me a estimate of what the missing value should be, based on the previous data patterns.

I was able to select relevant features based on exploration change using the numerical columns. For example, the difference between close and open was used to create the price change because close and open columns are able to identify the direction of price movement during a trading period. While the difference between high and low was used to create the price spread because, the high and low price is able to determine the price range during a specific time range, the price spread which another name for price range. And finally, price return which is the percentage change between the close price of one period and the previous period. See Figure 3

	Date	Open	High	Low	Close	Volume	Currency	Price_Return	Price_Change	Price_Spread
0	2019-06-18	9128.269531	9149.763672	8988.606445	9062.045898	952850.0	USD	NaN	-0.054713	-0.522275
1	2019-06-19	9068.174805	9277.677734	9051.094727	9271.459961	131077.0	USD	0.380870	0.143737	-0.504758
2	2019-06-20	9271.567383	9573.689453	9209.416992	9519.200195	83052.0	USD	0.446709	0.176392	-0.467892
3	2019-06-21	9526.833984	10130.935547	9526.833984	10127.998047	76227.0	USD	1.125438	0.436710	-0.403679
4	2019-06-22	10151.890625	11171.013672	10083.189453	10719.981445	84485.0	USD	1.025097	0.412357	-0.274166
...	...	...	...	...	...	...	...	...	...	...
1146	2022-08-18	23331.542969	23567.285156	23152.455078	23222.242188	4546110.0	USD	-0.122132	-0.086432	-0.454356
1147	2022-08-19	23219.097656	23219.097656	20898.304688	20902.404297	13856579.0	USD	-1.861376	-1.711820	0.055954
1148	2022-08-20	20899.923828	21344.845703	20864.435547	21153.019531	7139073.0	USD	0.178183	0.180414	-0.436797
1149	2022-08-21	21153.412109	21695.794922	21125.320312	21561.177734	6657571.0	USD	0.311357	0.294304	-0.412683
1150	2022-08-23	21337.318359	21412.892578	21280.458984	21303.986328	6955056.0	USD	-0.257817	-0.030493	-0.529966

1151 rows × 10 columns

Figure 4: Relevant features based on exploration change

I then used standard scaler and min max scaler to scale the feature to a common range, in order to improve model performance. For standard scaler, I made a new variable called feature_column that has the numerical features of the price change, price spread and price return. This variable is going to be used to fit my standard scaler and also my min max scaler. See Figure 5 and Figure 5.1

```
[56]: scaler = MinMaxScaler()

[57]: df[numerical_columns] = scaler.fit_transform(df[numerical_columns])

[58]: print(df[numerical_columns].head())
```

```
   Price_Change  Price_Spread  Price_Return
0      0.476445      0.001094           NaN
1      0.495890      0.001873      0.425304
2      0.499090      0.003515      0.429079
3      0.524597      0.006373      0.467995
4      0.522211      0.012140      0.462242
```

Figure 5: Min Max Scaler

```
[164]: scaler = StandardScaler()

[165]: feature_columns = ['Price_Change', 'Price_Spread', 'Price_Return']
       df[feature_columns] = scaler.fit_transform(df[feature_columns])
       print(df[feature_columns].head())
```

```
   Price_Change  Price_Spread  Price_Return
0     -0.054713     -0.522275           NaN
1      0.143737     -0.504758      0.380870
2      0.176392     -0.467892      0.446709
3      0.436710     -0.403679      1.125438
4      0.412357     -0.274166      1.025097
```

Figure 5.1: Standard Scaler

CHAPTER THREE: FEATURE SELECTION TECHNIQUES

Feature selection is important because it identifies and select the most relevant features that contribute to the prediction of the target variable. I implemented three feature selection method, which are filter, wrapper and embedded method in my jupyter notebook. For filter methods, I implemented two types, the correlation matrix and mutual info, See Figure 6 and 6.2. They are both valuable tools for understanding the relationship between variables in a dataset. Correlation matrix shows linear relationship between two variables that typically ranges from -1 to 1. While mutual info detects nonlinear relationship between variable. High mutual information with the target variable indicates that the feature is relevant for prediction.

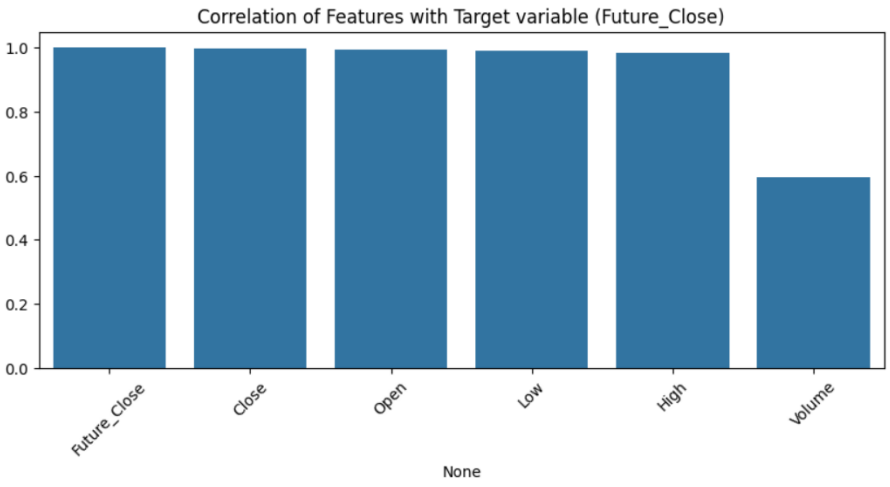

Figure 6: Correlation of Features with target variable

```
mi = mutual_info_regression(X, y)
mi_series = pd.Series(mi, index=X.columns)
print(mi_series)
```

```
Open      2.178671
High      2.243441
Low       2.256409
Close     2.464932
Volume    0.806197
dtype: float64
```

Figure 6.2: Mutual Information of Features with target variable

For wrapper methods, I implemented Recursive Feature Elimination (RFE) and Recursive Feature Elimination with Cross-Validation (RFECV). RFE is used remove less important features and determine the optimal subset of features for prediction. While RFECV is an advanced method of RFE because it automates the process and select the most relevant features by using cross validation. See Figure 7. And finally for embedded method, I implemented lasso regression. Lasso regression is used in creating simpler, more interpretable models by selecting only the most important features.

```
R2: 0.9950806874786295
Mean Squared Error: 1533477.428766134
```

```
print('Using RFE and RFECV open and close is the subset of the relevant feature for prediction')
```

```
Using RFE and RFECV open and close is the subset of the relevant feature for prediction
```

Figure 7: Using RFE and RFECV

The best method in this three is wrapper method because it has the lowest MSE and the highest R-squared. See table 1 below for detail comparison.

METHOD	USES	PROS	CONS	RESULT
Filter Method	Used for understanding the relationship between features to the target variable	- Independent of models - Works well with large dataset and categorical variables	- May select features irrelevant to the prediction	The correlation selected features that were already relevant but couldn't specify which was more valuable. Mutual info gave us which features varies in

				relevance.
Wrapper Method	Used to iteratively evaluates subset of features with the target variable	- Tailored to the specific model - Considered feature interaction - Leads to a better Performance	- Computing RFE for each feature is redundant. - May overfit on training data if cross validation is not used.	Had the lowest MSE and the highest R2. It had better feature optimization.
Embedded Method	Used to handle regularization-based models	- Combines feature selection and training - Avoids overfitting by regularization	- Requires a model with already built in important feature	MSE was low to moderate and R2 was high but this depends on the model used.

Table 1: Comparison of All Feature Selection Method

CHAPTER FOUR: MODEL TRAINING AND EVALUATION

Now that I know the relevant features using feature selection. In my jupyter notebook, I implemented three different models for Bitcoin price prediction. Those models are random forest, regression, and KNN. I made sure to split the data into training and testing sets and I trained each model on the training set. The result shows that linear regression was the best performing model because it has the lowest MSE and the highest R2 score, it MSE score is 152590 and R2 is 0.995104. Random forest is the second-best preforming model, it has an MSE of 19037838 and R2 score of 0.9938. While KNN was the least performing model with an MSE of 6221617 and R2 of 0.8004.

I can also verify each model performance using visualization. See Figure 8, 8.2 and 8.3. The visualization will show me which model has the best perfect fit to the prediction across the axis of the predicted values. From the MSE and R2 score it will then be safe to say that it will be linear regression model that will have the perfect fit.

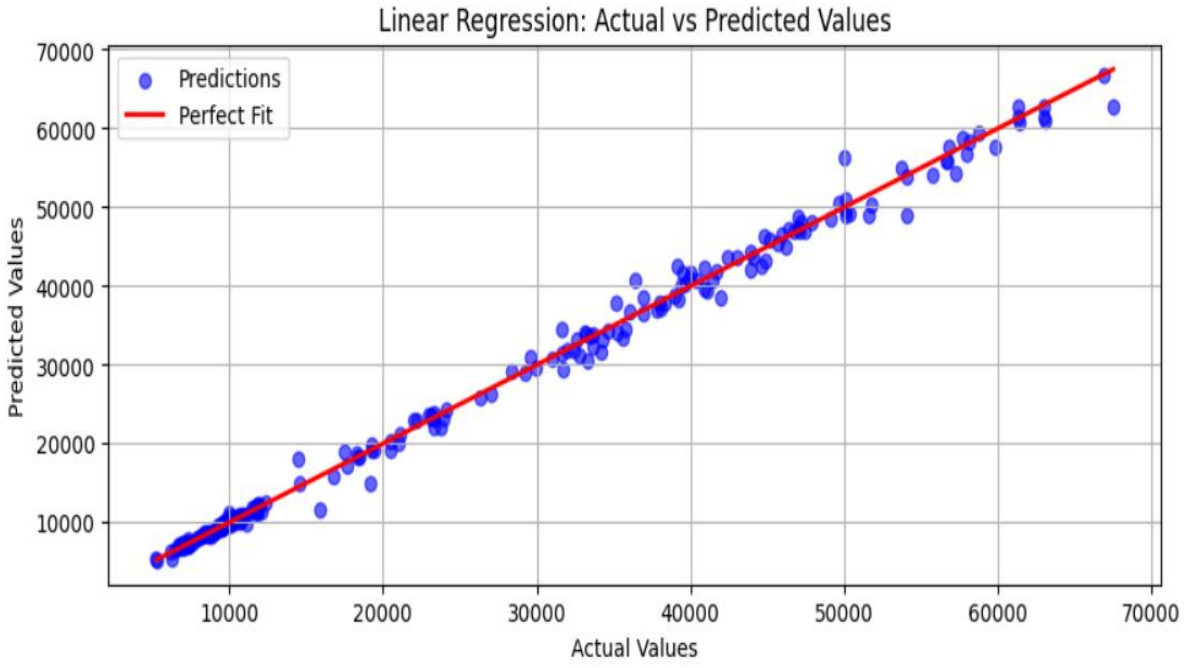

Figure 8: Linear Regression Model

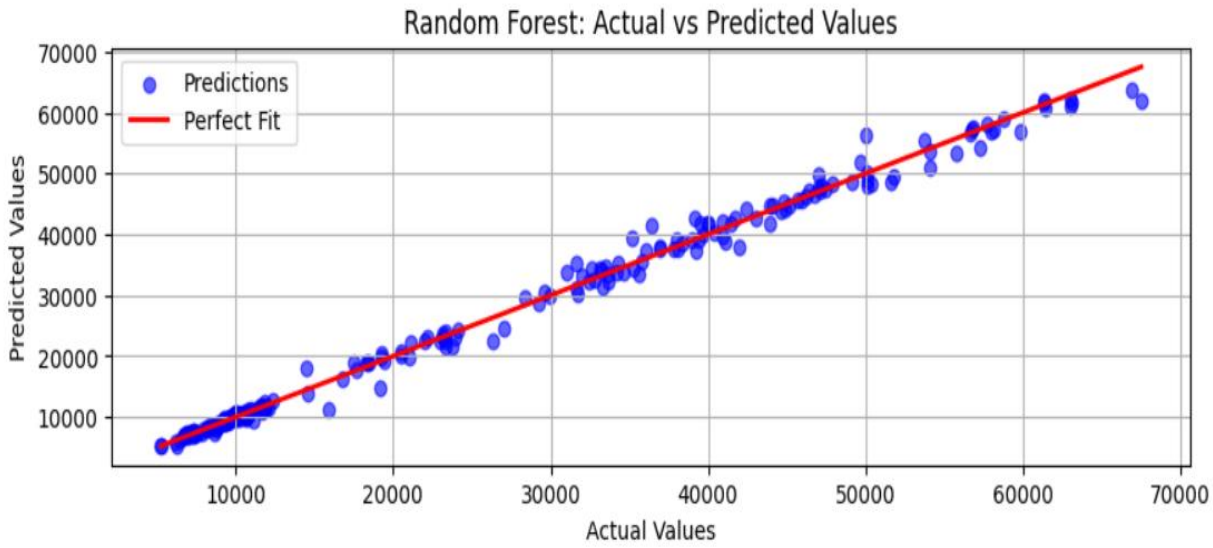

Figure 8.2: Random Forest Model

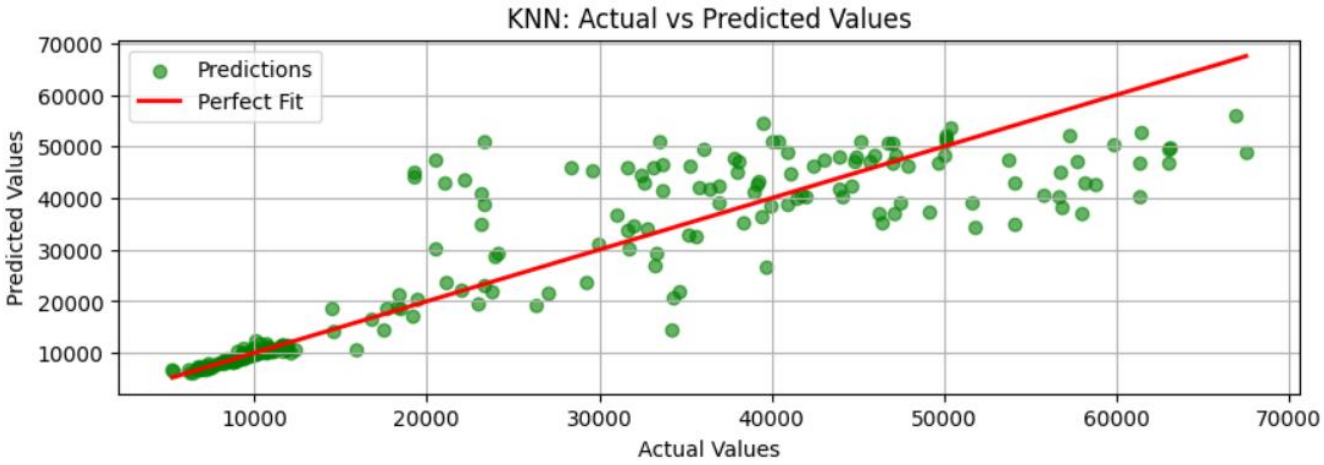

Figure 8.3: KNN Model

I also evaluated each model performance on the testing set using RMSE. RMSE measures the average difference between predicted and actual prices. Linear regression model has the lowest RMSE of 1235.28, while random forest has RMSE of 1379.76 and KNN with RMSE of 7887.72. Therefore, linear regression model is the best model for optimal price prediction.

CHAPTER FIVE: FINDINGS AND FUTURE WORK

It is possible that any model could outwork or be the best among the other models, like how linear regression was the best model for predicting bitcoin prices but this solely depends on the dataset and on what questions was being asked. Linear regression performed slightly better than random forest in terms of MSE and R-squared, despite its simplicity. Random forest ability to capture complex relationships and handle feature interaction contributed to its accuracy. KNN model exhibited significantly higher error and a lower R-squared score, indicating poor predictive performance compared to Random Forest and Linear Regression. Its sensitivity to feature scaling and the choice of k could likely contribute to these results.

Feature selection played a role in the machine learning models. KNN performance degradation can be attributed with its use of irrelevant features, KNN based its model on all the features; relevant and irrelevant without scaling. Random forest has a in built feature selection which made it perform well in the analysis. While linear regression used feature selection to improve model interpretability. Linear regression reduces its multicollinearity by eliminating redundant features which leads to a staple prediction estimate. Below is the table comparing each machine learning models.

Aspect	Random Forest	Linear Regression	KNN Regressor
Mean Squared Error (MSE)	1903738	1525909	6221617
R-squared	0.99389	0.995104	0.8004
Performance	High accuracy with low error; captures nonlinear relationships.	Strong baseline performance; best R-squared and MSE but limited to linear assumptions.	Poor performance; sensitive to scaling and the choice of hyperparameters.
Impact of Feature	Handled redundant or	Sensitive to	Performance degraded

Selection	irrelevant features well due to built-in feature selection.	multicollinearity; benefits from careful feature selection.	with irrelevant features.
Strength	Robust to overfitting, handles nonlinear relationships, interpretable via feature importance.	Simple, interpretable, fast to train and deploy.	Nonparametric, captures local patterns well in smooth data.
Weaknesses	Computationally intensive, requires tuning for best results.	Struggles with nonlinear data and oversimplifies complex relationships.	High computational cost, sensitive to outliers and irrelevant features.

Table 2: Model Comparison

Some potential limitations include the fact that Bitcoin's price is highly volatile and influenced by factors such as market sentiment, regulatory decisions and global economic events, which may not be captured adequately by historical price data. Also, models trained on historical data might not generalize well to future trends due to unforeseen market shifts or events not represented in the training set.

For future work, it is best to implement advanced future engineering such as moving averages, Bollinger bands or volatility indices and also, is best to analyze dependencies using time-series techniques. We can also include data on market sentiment, news headlines and macroeconomic indicators to improve model performance. Finally, we can integrate model ensemble technique such as combining two models like random forest and linear regression for enhanced predictive accuracy.

CONCLUSION

Predicting Bitcoin prices is a complex challenge, driven by the cryptocurrency's volatility and unpredictable market forces. This report compared the performance of three machine learning models, Random Forest, Linear regression and KNN regressor to understand their strengths and weaknesses in Bitcoin price prediction. The results emphasized the crucial role of model selection, feature engineering and data preprocessing in achieving accurate predictions.

Among the models tested, linear regression delivered the highest R- squared value of 0.9951, suggesting strong accuracy. However, its linear assumptions limit its ability to capture the complex, nonlinear relationships within Bitcoin prices. Random forest, with an R-squared of 0.9939, outperformed linear regression in capturing nonlinearities and feature interactions, demonstrating greater flexibility and robustness to overfitting. In contrast, the KNN regressor performed poorly, showing a much higher MSE and a lower R-squared score due to its sensitivity to irrelevant features.

Feature selection played a significant role, with random forest effectively managing redundant features, while careful feature curation improved linear regression and KNN performance. This underscores the importance of feature engineering for optimizing model results.

Despite the model's promising performance, their reliance on historical data limits their predictive power, as external factors like market sentiment or regulatory events can significantly influence Bitcoin prices. Future work should focus on incorporating external data sources, advanced feature engineering, and deep learning techniques to enhance prediction accuracy and adaptability in this volatile market.

BIBLIOGRAPHY

References

Usman Akhtar (September 2024). *Cryptocurrency-Dataset.* Available at: https://github.com/usmanakhtar/Cryptocurrency-Dataset/. (Accessed: 23 December 2024).

Victory Okezie (December 2024). *Predicting-Crypto.* Available at: vbarbas03/Predicting-Crypto: Predicting Crypto Prices (Accessed: 27 December 2024).

List of Figures

List of Tables