PHENIKAA UNIVERSITY

FALCULTY OF ELECTRONIC ENGINEERING

MACHINE LEARNING MID-TERM PROJECT REPORT

Project: Predicting Cryptocurrency Direction using Classification Algorithms with 5 Classes

1. Duong Doan Tung

2. Nguyen Trong Huy Hoang

3. Le Hoang Nam

HA NOI, 03/2023

TABLE OF CONTENTS

WORKLOAD DISTRIBUTION

Full name	Task
Duong Doan Tung	Improve code, report, learn algorithms (Fibonacci retracement levels, pivot points, and Sentiment analysis.
Nguyen Trong Huy Hoang	Collect and process data, learn algorithms (moving average, RSI, Bollinger Bands), power point.
Le Hoang Nam	Collect and process data, learn algorithms MACD, OBV, ATR), calculate: Accuracy, Recall, Precision, F1 score.

Project include:

1. Project report
2. Presentation file and video
3. Python files, Jupyter Notebook
4. Dataset sample

Full project material can be found here:

https://drive.google.com/drive/folders/1N1mTDexQZVoSI7srQ6srAZk_G0kJADpr?usp=share_link

Project repository: https://github.com/dtungpka/CPC5

Special thanks to our teacher Dr. Huy Minh Le for his help in this project

Section 1: ABSTRACT

The use of classification algorithms for predicting cryptocurrency prices has garnered significant interest in recent years. In this study, we focus exclusively on using the K-Nearest Neighbors (KNN) and Logistic regression algorithm to predict the direction of cryptocurrency prices with five classes: small increase, large increase, small decrease, large decrease, and no change.

We used a dataset containing historical cryptocurrency price data and calculate some technical indicators from it. We evaluated the performance of the algorithm using accuracy, precision, recall, and F1 score metrics to determine the performance of KNN and Logistic regression in predicting cryptocurrency price direction with five classes.

Our results demonstrate that the Logistic regression algorithm is a viable tool for predicting cryptocurrency price direction with multiple categories. In average, the Logistic regression algorithm achieved an accuracy of 0.869, precision of 0.673, recall of 0.155, and F1 score of 0.252.

The findings of this study could potentially benefit investors and traders in making more informed and nuanced decisions within the cryptocurrency market. This study provides valuable insights into the use of classification algorithms for predicting cryptocurrency prices, specifically highlighting the efficiency of the KNN and logistic regression algorithm for this purpose.

Section 2: PROJECT OVERVIEW

1. Inspiration

The advent of cryptocurrencies has introduced a new phase of digital financial transactions. The popularity and volatility of cryptocurrency prices have led to increased interest in predicting their direction. Accurate price predictions can potentially benefit investors and traders by providing insights into market trends and facilitating informed decision-making.

Despite the rapid growth of the cryptocurrency market, predicting cryptocurrency prices, or stock price in general, remains a challenging task. The complex and dynamic nature of this market presents unique challenges for traditional financial analysis techniques. Therefore, there is a growing need to explore alternative methods for predicting cryptocurrency prices. Because of that, we decided to start this project: **Cryptocurrency Price Classification with 5 classes (CPC5)**

The inspiration for this study stems from the need to explore new approaches for predicting cryptocurrency prices. By focusing on the KNN algorithm and the Logistic regression algorithm, we aim to contribute to the growing body of research on machine learning techniques for predicting cryptocurrency prices. The findings of this study have the potential to benefit investors and traders in making more informed and nuanced decisions within the cryptocurrency market.

2. Basic Idea

To accomplish our objective of predicting cryptocurrency price direction using the above algorithm, we utilized a dataset consisting of historical cryptocurrency price data and technical indicators. We first obtained the price data of a cryptocurrency from Binance Public Data, one of the most reliable, biggest sources for cryptocurrency data.

After obtaining the price data, we calculated several technical indicators such as the Relative Strength Index (RSI), Moving Average Convergence Divergence (MACD), Boilinger Bands, Average Directional Movement Index (ADX), and others. These technical indicators provide valuable information on market trends, momentum, and potential price movements.

To ensure the optimal performance of our model, we employed a feature selection process to identify the most relevant technical indicators. This process involves filtering out irrelevant features and identifying the ones that provide the most significant predictive power for our model.

Once we obtained the most suitable features, we proceeded to calculate the true labels for our model. To achieve this, we used a clustering algorithm, such as k-means, to group the data into several clusters. We then assigned a label to each cluster, representing the expected direction of price movement for that cluster.

By utilizing this approach, we created a dataset with true labels that accurately reflect the price direction of the cryptocurrency. This dataset was then used to train and evaluate our models, with the aim of accurately predicting the direction of cryptocurrency prices.

Our ultimate goal is to create a trading bot that can leverage our model's predictive power to make profitable trades. The trading bot will use the predicted price direction to determine when to enter or exit trades.

To increase the trading bot's profitability, we will incorporate a modified version of the Martingale strategy. The Martingale strategy is a popular money management technique that involves doubling the trading volume after each loss, with the aim of recouping all losses in a single winning trade.

However, the Martingale strategy has its limitations, and in its purest form, it can lead to significant losses. To address this, we will modify the Martingale strategy to incorporate risk management measures that limit losses and maximize profits.

By combining our prediction models with a modified Martingale strategy, we aim to create a robust and profitable trading bot that can navigate the volatile cryptocurrency market. This study represents a crucial step towards realizing this goal and has the potential to revolutionize the cryptocurrency trading landscape.

3. Final-term development direction

In the final-term project, we want to improve performance of the model, implement a new model for predicting high and low price. Also we will make a real-time trading bot, that use all of our model, strategy combined to start trading online.

Section 3: PROJECT DETAILS

1. The dataset

1.1. Acquiring the dataset

To procure the comprehensive dataset required for our study, we developed a script, *binance_data.py*, which utilizes the Binance API to download historical cryptocurrency price data. The script is available in our GitHub repository.

To ensure the data downloaded is relevant and aligned with our study's goals, we specified the time range in the details.json file. This file provides the script with the necessary information to download data for the specified time range.

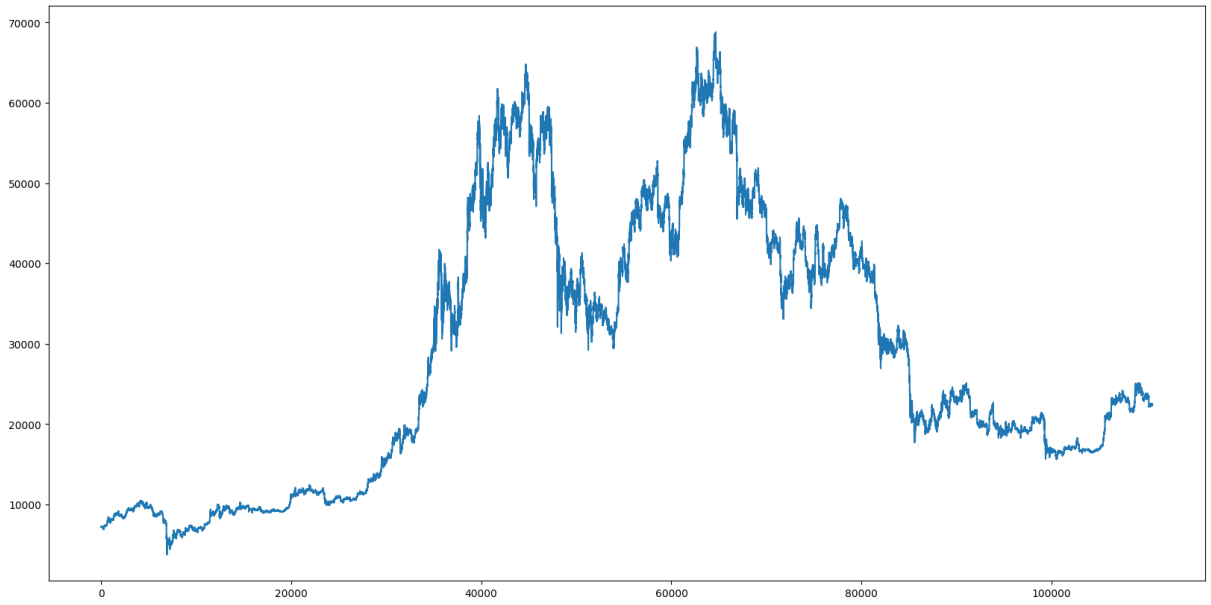

Figure 1. BTCUSDT in 15 minute interval

Figure 1 above shows the BTCUSDT price in 15 minute interval, from 31/12/2019 to 06/03/2023, with total of 110607 data points.

d:\2022-2023\ML\CPC5\Training.ipynb > X_data_df (110607, 10)

	index	open	high	low	close	volume	close_time	quote_volu...	count	taker_buy_vol...	taker_buy_qu...
0	0	7224.17	7229	7208.62	7215.26	1187.906	1577752199999	8574712.64865	1575	601.806	4344205.47527
1	1	7215	7233.36	7214.68	7231.68	849.478	1577753099999	6138599.81583	1691	489.493	3536729.75568
2	2	7231.68	7250	7227.86	7246.97	812.426	1577753999999	5880106.53795	1519	468.28	3389344.4409
3	3	7247.35	7264.99	7239.67	7254.54	1023.073	1577754899999	7420736.49665	1914	563.82	4089698.74031
4	4	7254.1	7258.96	7237.53	7243.3	774.87	1577755799999	5614987.42656	1391	281.101	2036913.36513
5	5	7243.38	7250	7239.59	7246.52	427.683	1577756699999	3098689.7136	661	222.975	1615419.64139
6	6	7246.8	7262	7246.13	7260.23	464.359	1577757599999	3368810.59152	797	272.737	1978350.43282
7	7	7260	7260.34	7253.01	7258.44	483.454	1577758499999	3508589.99622	793	240.066	1742091.55862
8	8	7258.46	7258.46	7236.68	7240	839.146	1577759399999	6078935.71451	1234	323.015	2339748.9902
9	9	7239.65	7240.04	7219	7230	1012.255	1577760299999	7320168.5464	2017	459.847	3324863.12562
10	10	7229.66	7242.3	7229.66	7242.06	638.084	1577761199999	4618480.04337	807	428.017	3097826.24172
11	11	7242.05	7245	7236.35	7242.99	351.646	1577762099999	2546734.39436	528	228.487	1654799.50103
12	12	7242.99	7245	7222.05	7227.73	772.186	1577762999999	5583418.11493	1187	336.075	2430156.47495
13	13	7227.74	7237	7214.31	7233.09	715.062	1577763899999	5167622.38296	1049	337.396	2438616.56969
14	14	7233.14	7239.28	7221	7232.69	544.625	1577764799999	3937441.5827	920	267.271	1932436.80808
15	15	7232.69	7247.25	7223.24	7244.99	750.762	1577765699999	5432229.24991	1119	464.002	3357635.02089
16	16	7245	7257	7240.65	7256.96	469.108	1577766599999	3399328.43082	1028	278.826	2020722.10996

Figure 2. Data structure

Each line of the dataset contains 10 values, but we only use 5 of them: open, high, low, close, volume.

This dataset is what made our project diffirent from other online stock/cryptocurrency price dataset and model. Instead of using 1 day period, we

use 15-minute time period, which made the dataset much bigger, while also increase our bot trading speed.[1]

1.2. Calculate the true labels

To classify cryptocurrency price movements into 5 different labels: "Large decrease", "Small decrease", "No change", "Small increase", "Large increase", we first calculate the average price each row by add the opening price (open) and closing price (close) for each row in the dataset and divide by 2.

$$\mu_t = \frac{O_t + C_t}{2}$$

With O_t is the open price, C_t is the close price

Then, the percentage change in price can be calculated as follows:

$$P_t = \frac{\mu_t - \mu_{t-1}}{\mu_{t-1}}$$

We can then visualize it:

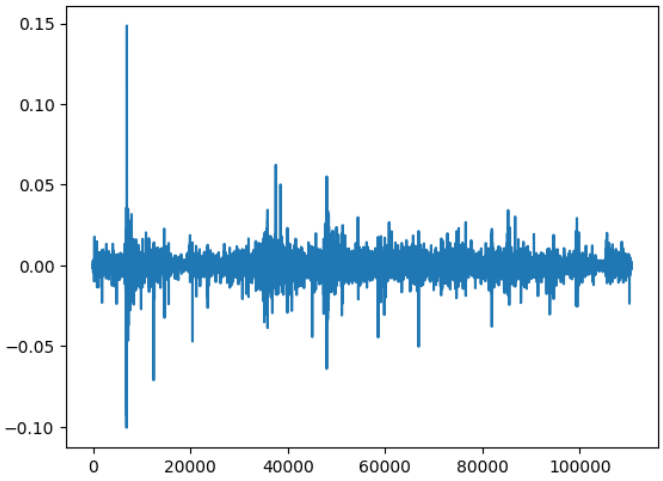

Figure 3. Percentage change in price

Based on this data, we used k-means clustering algorithm to cluster the data into 5 groups based on their similarity.[2]

[1] Our intention is to make the bot to trade every 15 minutes.
[2] The k-means we implement in this project is from sklearn. Note that this is diffirent from our KNN implementation, which is not from sklearn library. This is due to lack of time in the mid-term, and we will use numpy/cuda entirely instead of sklearn in the final-term report.

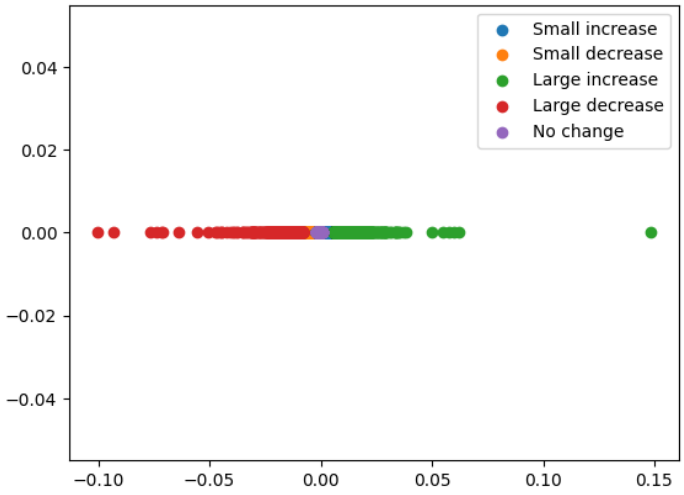

Figure 4. K-mean clustering

2. Feature Engineering

Our next step is to select, calculate technical indicators, and choose what to use in our model.

2.1. Technical Indicators

Exclude the first 5 value we already have from the dataset: open, close, low, high, volume; we need to calculate the remaining technical indicators. For better understanding, let $p = 96$ is the time period for calculating some of the indicators.[1]

2.1.1. Simple Moving Average (SMA)

The Simple Moving Average (SMA) is a commonly used technical analysis indicator that helps to identify trends in the price of an asset, it can be calculated by:

$$SMA_t = \frac{\sum_{i=t-p}^{t} C_i}{p}$$

2.1.2. Exponential Moving Average (EMA)

The Exponentially Moving Average (EMA) is a quantitative or statistical measure used to model or describe a time series.

$$EMA_t = \alpha * C_t + (1 - \alpha) * EMA_{t-1}$$

With

$$\alpha = 2/(p + 1)$$

[1] We choose value 96 because the dataset is 15-min period, so 96*15 = 1440 minutes which is 1 day.

2.1.3. Relative Strength Index (RSI)

The Relative Strength Index (RSI) is a momentum oscillator that measures the speed and change of price movements.

$$RSI_t = 100 - \frac{100}{1 - \frac{AG}{AL}}$$

Where AG, AL is the average gain, average loss:

$$AG = \begin{cases} abs(SMA_t) & if \quad C_t - C_{t-1} > 0 \\ 0 & \quad C_t - C_{t-1} \leq 0 \end{cases}$$

$$AL = \begin{cases} abs(SMA_t) & if \quad C_t - C_{t-1} \leq 0 \\ 0 & \quad C_t - C_{t-1} > 0 \end{cases}$$

2.1.4. Bollinger Bands

Bollinger Bands are volatility bands placed above and below a moving average. Volatility is based on the standard deviation, which changes as volatility increases and decreases.

First we calculate the standard deviation:

$$\sigma = \sqrt{\frac{\left(\sum_{i=t-p}^{t} C_i - SMA_t\right)^2}{p-1}}$$

Calculate the upper Bollinger bands:

$$BB_{up} = SMA_t + 2\sigma$$

Calculate the lower Bollinger bands:

$$BB_{lower} = SMA_t - 2\sigma$$

2.1.5. Moving Average Convergence Divergence (MACD)

Calculate the short term exponential moving average (SEMA):

$$SEMA = \alpha * C_t + (1 - \alpha) * SEMA_{t-1}$$

With

$$\alpha = 2/(p/2 + 1)$$

Then the moving average convergence divergence:

$$MACD = EMA - SEMA$$

2.1.6. On Balance Volume (OBV)

Let V_t be the volume of the current period.

$$OBV_t = \begin{cases} OBV_{t-1} + V_t \quad if\ C_t > C_{t-1} \\ OBV_{t-1} - V_t\ if\ C_t < C_{t-1} \\ OBV_t\ if\ C_t = C_{t-1} \end{cases}$$

2.1.7. Average true range (ATR)

Let H_t be the current high price of the current period, L_t be the current low price of the current period.

$$TR_t = H_t - L_t$$

Then:

$$ATR_t = \frac{(ATR_{t-1} * (p-1)) + TR_t}{p}$$

2.1.8. Fibonacci Retracement

Fibonacci retracement is a popular technical analysis tool used to identify potential levels of support and resistance in a financial market. It is based on the idea that prices will often retrace a predictable portion of a move, after which they will continue to move in the original direction. The key Fibonacci retracement levels are 23.6%, 38.2%, 50%, 61.8%, and 78.6%. These levels are calculated by taking the difference between the high and low of a price move, multiplying it by the Fibonacci ratios and then adding or subtracting the result from the starting price of the move.

2.2. Feature Selection

To optimize the performance of our classification algorithms, we need to select the most relevant features from our dataset. Feature selection is an essential step in the preprocessing stage, as it helps to reduce the dimensionality of the dataset, remove irrelevant or redundant features, and improve the accuracy and efficiency of the algorithms.

In our project, we experimented with various combinations of features[1], where each feature corresponds to a unique technical indicator. Our dataset consists of 16 features, with a minimum of 8 features used in the model. As a result, the total number of possible combinations amounts to 6475.[2]

[1] Right now, in mid-term report, we find the best combination by randomly choose the combination, but we planned to use Correlation-based Feature Selection (CFS) algorithm in the final-term report.
[2] Bollinger band upper, lower act as one feature, the same with fibonacci retracement level.

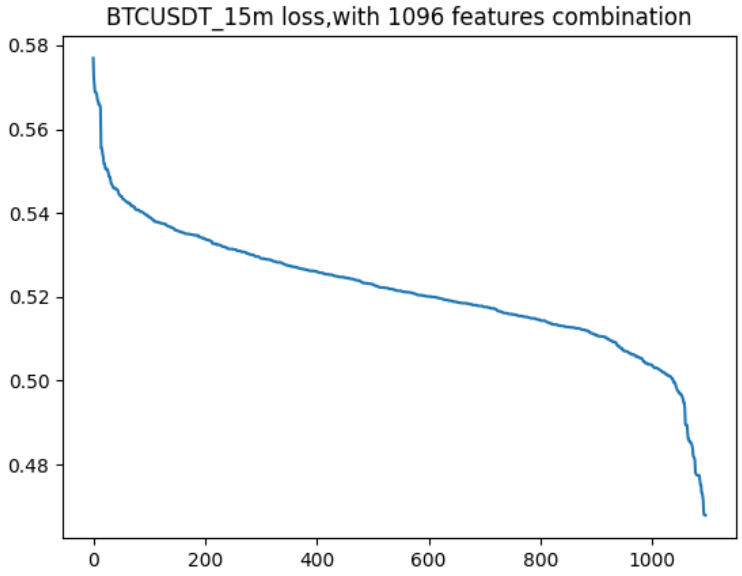

Figure 5.BTCUSDT loss

After choosing 1096 different combinations, we find that combination of *['VOLUME', 'EMA', 'BOILINGER_UP', 'BOILINGER_DOWN', 'OBV', 'ADX', 'FIBO0', 'FIBO1', 'FIBO2', 'FIBO3']* yield the best result, with average of 0.83 accuracy in KNN model.

3. Model Selection

3.1. K-Nearest Neighbors (KNN)

The K-Nearest Neighbor algorithm is a simple algorithm consisting of four main steps. Firstly, the algorithm calculates the distance between the given point (x) and all the points in class A and B. Secondly, the algorithm identifies the smallest distance from point x to a point in classes A and B. Thirdly, the algorithm checks which class the nearest point belongs to, either class A or B. Finally, the algorithm assigns the point x to the same class as the nearest point.

To calculate the distance between two vectors, we used the euclidean distance formula:

$$d_i = \sqrt{\sum_{i=1}^{k} (x_i - y_i)^2}$$

With $k = 5$ is the number of neighbors.

We use this formula to calculate the distance from x to all data points, in this case, distance from every vector in train set to test set.

After that, we sort and take d_k points of distance in each point in test set, and predict that point to the class having the most points out of *k* point.

3.2. Logistic regression

In addition to the KNN model, we also implement a Multi-class logistic regression model to compare the results. This model can be found in *Training logistic regression.ipynb* file. We compared and evaluated the perfomance of logistic regression model to the KNN model using the same features combination and training/testing set.

Logistic regression is a popular classification algorithm used to predict the probability of an event occurring based on input variables. In our case, we used logistic regression to classify the price movement of cryptocurrencies into 5 said classes.

The model can be represented by the following steps:

First we define the cost function:

$$J(\theta) = -\left[\frac{1}{m}\sum_{i=1}^{m} -y^{(i)} \log h(x^{(i)} - \left(1 - y^{(i)}\right) \log\left(1 - h_\theta\left(x^{(i)}\right)\right)\right]$$

With $h_\theta(x)$ function is the sigmoid function:

$$h_\theta(x) = \frac{1}{1 + e^{-x}}$$

We use conjugate gradient to find the minimum.

The partial derrivative or J(θ):

$$\frac{\delta}{\delta\theta_J} = \frac{1}{m}\sum_{i=1}^{m}\left[\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right]x_j^{(i)}\right.$$

Then we learn the θ using conjugate gradient to find the optimize value.

This then applied for every *k* class in our dataset (one-vs-all classification).

4. Result

After we trained our model, we evaluated the result using 4 popular metrics:

1. Accuracy: measures the proportion of correctly classified instances out of all instances.

2. Recall (also known as sensitivity): measures the proportion of true positive instances (i.e., instances that belong to the positive class) that are correctly classified.
3. Precision: measures the proportion of true positive instances out of all instances that are classified as positive (i.e., the instances that the model predicted to belong to the positive class).
4. F1 score: is the harmonic mean of precision and recall. It is a single number that represents the balance between precision and recall.

The below data are trained on our best combination of *['VOLUME', 'EMA', 'BOILINGER_UP', 'BOILINGER_DOWN', 'OBV', 'ADX', 'FIBO0', 'FIBO1', 'FIBO2', 'FIBO3']*, with the data displayed is the test set, total of 44165 data points.

4.1. KNN model

KNN Model								
Label	TP	TN	FP	FN	Accuracy	Precision	Recall	F1 score
SI	855	5495	1328	1155	0.719	0.392	0.135	0.2
SD	388	6950	688	807	0.831	0.361	0.053	0.092
LI	80	8435	157	161	0.964	0.338	0.009	0.018
LD	36	8644	69	84	0.983	0.343	0.004	0.008
NC	3737	2071	1495	1530	0.658	0.714	0.643	0.677
				Average	**0.831**	**0.577**	**0.139**	**0.224**

Label meaning:

Shorthand	Full label name
SI	Small increase
SD	Small decrease
LI	Large increase
LD	Large decrease
NC	No change

4.2. Logistic regression

Logistic regression Model								
	TP	TN	FP	FN	Accuracy	Precision	Recall	F1 score
SI	0	5965	0	643	0.903	0	0	0
SD	283	4672	431	1222	0.75	0.396	0.057	0.1
LI	4146	623	1698	141	0.722	0.709	0.869	0.781
LD	7	6453	31	117	0.978	0.184	0.001	0.002
NC	9	6556	3	40	0.993	0.75	0.001	0.003
				Average	**0.869**	**0.673**	**0.155**	**0.252**

Section 3: Discussion

1. Interpretation of Results

The KNN model achieved an accuracy of 0.831, indicating that it correctly classified 83.1% of the instances. The precision of the model was 0.577, which means that of all the instances classified as positive, only 57.7% were positive. The recall, or true positive rate, was 0.139, indicating that only 13.9% of all positive instances were correctly classified. The F1 score, a measure of the trade-off between precision and recall, was 0.224.

On the other hand, the Logistic regression model achieved higher performance, with an accuracy of 0.869. This indicates that it correctly classified 86.9% of the instances. The precision of the model was 0.673, meaning that of all the instances classified as positive, 67.3% were actually positive. The recall was 0.155, indicating that only 15.5% of all positive instances were correctly classified. The F1 score was 0.252, indicating that the Logistic regression model outperformed the KNN model in terms of the trade-off between precision and recall.

2. Limitations and Future Work

2.1. Limitations

Despite achieving promising results in this study, there are still several limitations:

Firstly, the technical indicators used in this study are only a subset of the available indicators. Future studies should explore the use of other indicators that may provide additional insights into the market behavior.

Secondly, the trading strategy used in this study was based on the martingale strategy, which can be risky and may not be suitable for all investors. Future studies should explore other trading strategies that can minimize risks and maximize returns.

2.2. Future Work

As described in the basic idea and Final-term development direction, our ultimate goal is to create a trading bot that can leverage our model's predictive power to make profitable trades. So we will continue to further improve the accuracy and applicability of the predictive models and start designing the trading bot.

Section 4: Conclusion

In conclusion, we have demonstrated the effectiveness of using technical indicators in predicting cryptocurrency prices and creating a trading bot. Our study shows that feature selection is crucial in optimizing the performance of classification algorithms, and that the K-nearest neighbors and logistic regression models are promising approaches for predicting cryptocurrency prices.

However, it should be noted that the choice of technical indicators and feature selection is not exhaustive, and there may be other indicators or features that could improve the performance of the model.

Overall, our study highlights the potential of using machine learning techniques for cryptocurrency trading and price prediction. We hope our findings will inspire further research and contribute to the development of more sophisticated trading bots in the future.

References

Binance Public Data. (n.d.). Retrieved from https://github.com/binance/binance-public-data

Bollinger Bands in Technical Analysis. (n.d.). Retrieved from https://www.elearnmarkets.com/blog/bollinger-bands-in-technical-analysis/

Market data Endpoints. (n.d.). Retrieved from https://binance-docs.github.io/apidocs/spot/en/#market-data-endpoints

MinhHuyLe, D. (n.d.). *Chapter 2-2 Logistic regression.*

MinhHuyLe, D. (n.d.). *Chapter 2-4 KNN.*

sklearn.cluster.KMeans. (n.d.). Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

Stock Closing Price Prediction using Machine Learning Techniques. (n.d.). Retrieved from https://www.sciencedirect.com/science/article/pii/S1877050920307924

Stock Prediction and Automated Trading System. (không ngày tháng). Được truy lục từ https://www.researchgate.net/publication/301143650_Stock_Prediction_and_Automated_Trading_System

Stock Price Prediction using Machine Learning Algorithms. (n.d.). Retrieved from https://www.ijraset.com/research-paper/stock-price-prediction-using-machine-learning

Study of Market Indicators used for Technical Analysis. (n.d.). Retrieved from https://www.researchgate.net/publication/360497413_Study_of_Market_Indicators_used_for_Technical_AnalysisStudy of Market