Predicting the Price Movement of Bitcoin

50.0038 Computational Data Science Project Report
Group 17 StockUpz
Github Repository: https://github.com/vann-lim/50038-Final-Project.git

Vannara Lim 1006088
Koh Jin Rui Vinny 1006036
Mahima Sharma 1006106
Liew Min Hui 1006166
Teh Hui Yi 1006383

April 2024

1 Introduction+Problem Statement

Cryptocurrencies, particularly Bitcoin, have emerged as a significant asset class, attracting widespread attention from investors and traders worldwide. The dynamic nature of cryptocurrency markets, characterized by high volatility and rapid price fluctuations, presents both opportunities and challenges for market participants. In this context, the ability to accurately predict the direction of Bitcoin's price movement is of paramount importance for making informed investment decisions and managing risk effectively.

To address this need, our study focuses on developing a predictive model specifically tailored for forecasting the next day's price movement of Bitcoin. Leveraging machine learning techniques, statistical analysis, and a diverse set of market data, we aim to improve the accuracy of predictions and provide valuable insights for investors and stakeholders in the cryptocurrency space.

The primary objective of our research is to develop a robust predictive model capable of forecasting whether the price of Bitcoin will increase or decrease in the next trading day. This binary classification task is particularly challenging due to the inherent volatility and complexity of cryptocurrency markets, which are influenced by a myriad of factors including market sentiment, macroeconomic indicators, and technological developments.

Specifically, the problem can be formulated as follows:

Given historical data of Bitcoin prices, along with relevant features such as trading volume, volatility, sentiment analysis of news and social media, and technical indicators (e.g., moving averages, relative strength index), our task is to train a machine learning model that accurately predicts the direction of Bitcoin's price movement for the next day.

Throughout our study, we explore various machine learning algorithms, including LSTM, SVM, random forest, adaboost, MLP, logistic regression, and neural networks, to leverage different aspects of market data and sentiment analysis for prediction. Additionally, we employed feature stacking to enhance the predictive power of our model.

The performance of our model is evaluated using the F1 score, a widely used metric in binary classification tasks, to ensure robustness and reliability. By developing an accurate and reliable predictive model, our research aims to provide actionable insights and empower stakeholders in navigating the dynamic landscape of cryptocurrency markets.

2 Datasets

Note: All other visualisations can be found in the Visualisation of Datasets folder.

2.1 BTC-USD Dataset

BTC-USD Dataset Website
The BTC-USD dataset has been sourced from Yahoo Finance and is a time-series dataset. It comprises observations collected over successive time intervals. The dataset encompasses columns including Date, Open, High, Low, Close*, Adj Close** and Volume, each serving distinct roles in the temporal representation of financial data. This dataset will be heavily used in all of our analysis.

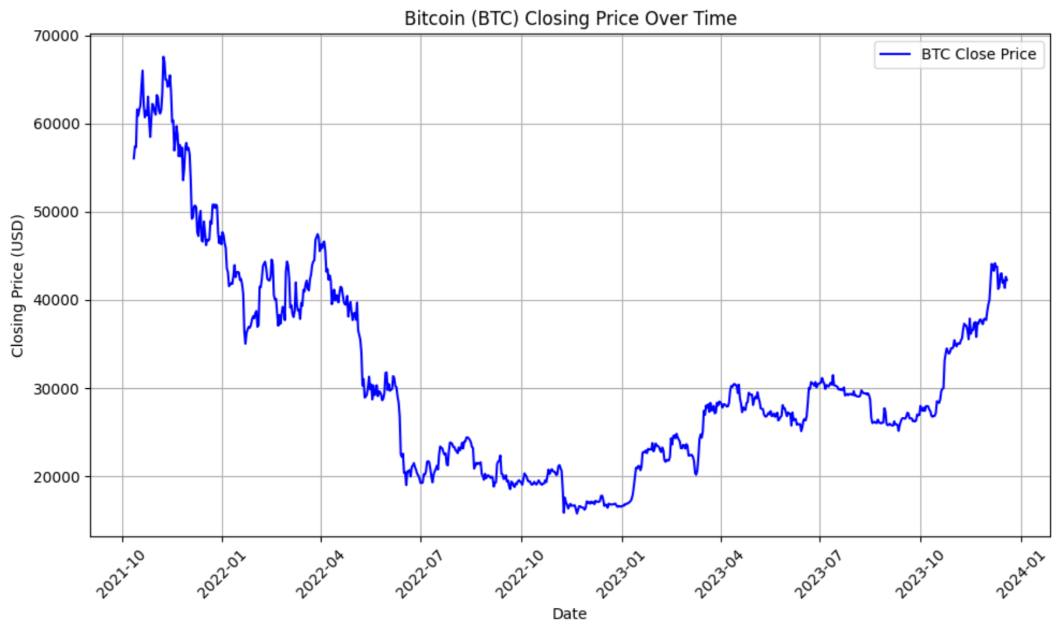


Figure 1: BTC Closing Price Over Month

2.2 S&P 500 Index Dataset

S&P 500 Index Dataset Website The dataset is taken from Yahoo Finance using yfinance python library. The dataset is a time-series dataset, reflecting data points gathered across consecutive time periods. It features key columns such as Date, Open, High, Low, Close*, Adj Close** and Volume, with each column fulfilling unique roles in delineating temporal trends within financial data.

We decided to look into S&P 500 data as there has been several speculation and hypothesis that there is a relationship between S&P 500 and Bitcoin prices. There has been reports in 2022 even up to recently.

2.3 US Interest Rate Dataset

Interest Rate Dataset Website
This dataset represents the Federal funds effective rate recorded over a specified timeframe, sourced from the Board of Governors of the Federal Reserve System. This dataset is structured in a time-series format, where each entry corresponds to a daily observation of the Federal funds effective rate. The rate is expressed as a percentage per annum, offering insights into the prevailing interest rate dynamics within the financial markets.

US Interest Rates is a federal-controlled financial indicator. Therefore, we selected this indicator to look at as a proxy to consider the connection between the cryptocurrency market and the financial market. In researching for this project, there are various articles speculating the effect of change in US Interest Rates on the Bitcoin market as seen here.

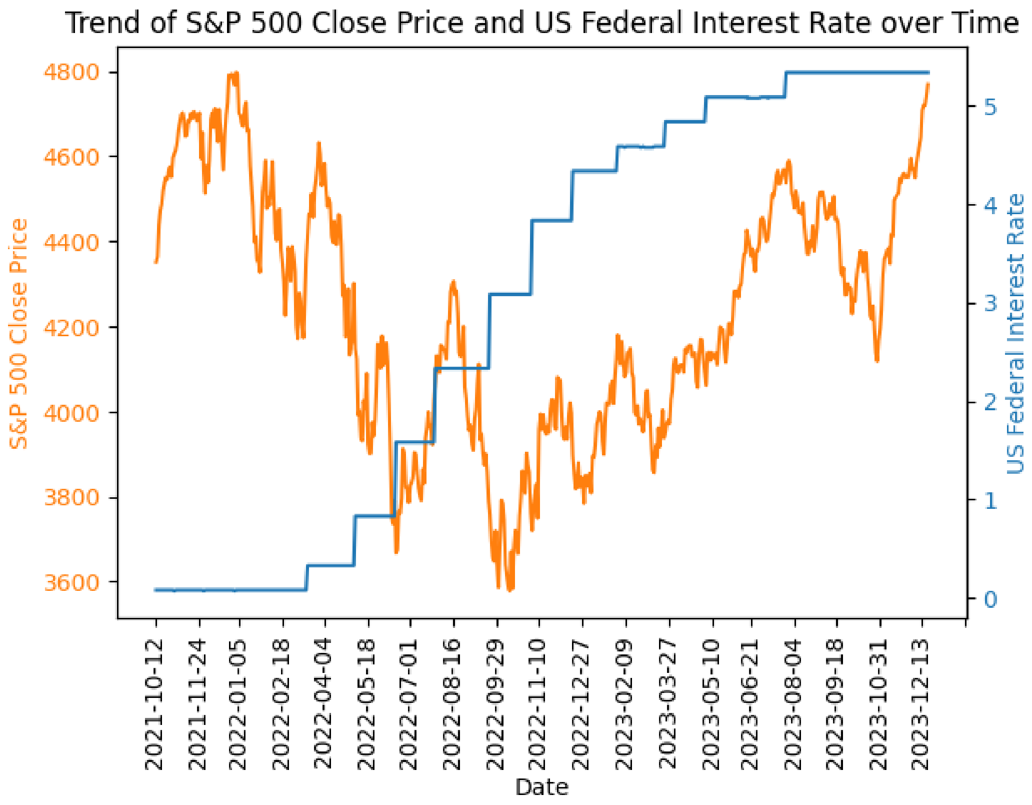


Figure 2: Trend of S&P 500 and Interest Rate Over Date

2.4 Cryptocurrency News Article Dataset

Cryptocurrency News Article Dataset Website
This dataset contains scraped cryptocurrency news article from over a year (2021-10-12 / 2023-12-19). It is structured formatting: title, text, source, subject, and sentiment analysis. The data is scraped from the following websites:

- cryptonews.com
- cryptopotato.com
- cointelegraph.com

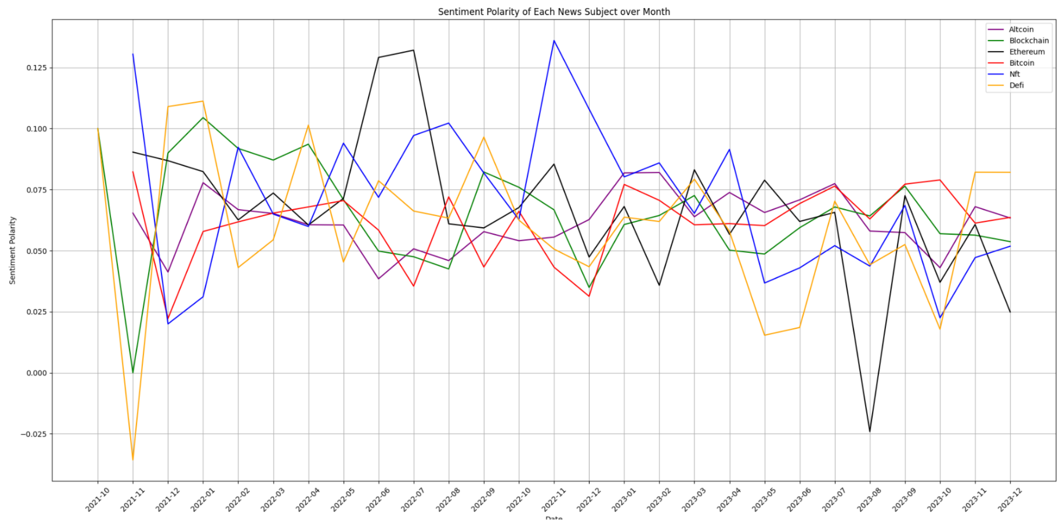


Figure 3: Sentiment Polarity of Each News Subject Over Month

The purpose of including this dataset is to understand whether news about cryptocurrency may influence the general outlook of the Bitcoin as there are studies that have established a connection between these too.

2.5 Cryptocurrency Tweets Dataset

Cryptocurrency Tweet Dataset Website
This dataset consist of a collation of scraped tweets from 50 identified cryptocurrency influencers on Twitter over 8 months. This data was originally cleaned by Mendeley to obtain clean tweet texts and the general tweet sentiment. Along with the tweet, some metadata like number of likes, retweets, reply counts are also included. Studies have shown that discussion on social media platforms like Twitter play a significant role in influencing sentiments not only in the cryptocurrency market but also the financial markets. Thus, we believe that including this dataset is highly relevant to our project.

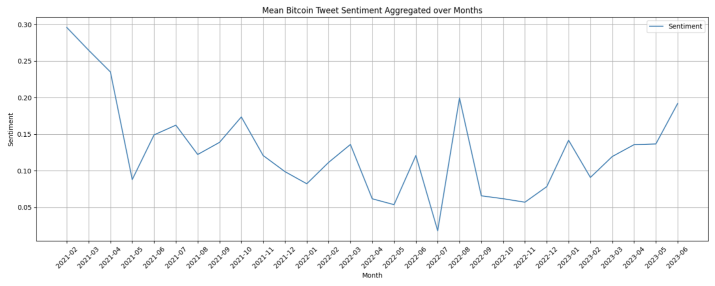


Figure 4: Mean Bitcoin Tweet Sentiment Aggregated Over Months

3 Approach

The approach to devise the solution is to first explore the impact of different factors on the price of Bitcoin. To do this, we trialed and tested different combinations of datasets (e.g News, Tweets, BTC-USD, S&P500, and Interest rates).

From these combinations, the best performing models are selected to be consolidated into the final model. In each of the subsection, we dive deeper into the study of each combination.

3.1 Tweets + BTC Prices

Note: The implementation of models using Tweets dataset can be found in the Exploration: News Tweets folder.

For this combination, we are investigating the effect of tweets and various tweet metadata on the price movement of bitcoin. The biggest consideration for this combination study is to relate textual data along with some normalised numerical data to price movement.

3.1.1 Data Processing

To start, data pre-processing is done to aggregate and organise the data to be used. Some primary preprocessing is done to BTC-USD dataset. All numerical datas are first normalised by applying the MinMax() normalisation function from scikit-learn. A new "Output" column is created in the BTC-USD dataset. The calculation for this column entails

$$Day\ 1\ Output = Closing\ Price\ of\ Day\ 2 - Closing\ Price\ of\ Day\ 1$$

There are also some dis-junctions between the Tweet dataset and the BTC-USD dataset that requires some data transformation. Firstly, each row in Tweet dataset is a unique tweet and there are multiple tweets recorded in one day. On the other hand, each row in the BTC-USD is a unique day. Additionally, the start and end dates for both datasets are different.

Keeping this dis-junction in mind, date format standardisation is done for both dataset, and an inner-join on 'Date' is performed on the transformed BTC-USD dataset with Tweets dataset. Each row in Tweets will be accompanied with BTC-USD data as well as the output. After this, rows containing Nan or Null values are removed and only the following columns are kept:

- clean_text
- normalized_vol
- importance_coefficient_normalized
- output

3.1.2 Model

Rationale for using LSTM

We focused on applying long-short-term-model (LSTM) for this section. There are various reasons why LSTM is a main consideration, but we took heavy inspiration from a study of Predicting stock market index using LSTM by Bhandari et al., 2022. LSTM networks offer a robust framework for investigating the relationship between tweets and Bitcoin price movements for several reasons. Firstly, they excel in handling sequential data, making them apt at capturing the temporal dynamics and dependencies inherent in both tweets and price fluctuations. Traditional neural networks often struggle with long-term dependencies, but LSTMs address this challenge by incorporating memory cells with gating mechanisms, enabling them to retain information over extended periods. This is particularly crucial when analyzing how specific tweets impact Bitcoin prices over time.

Additionally, LSTM networks accommodate variable-length sequences, allowing for the flexible processing of tweets of varying lengths without compromising performance. Their ability to automatically extract relevant features from input data, such as tweet metadata, eliminates the need for manual feature engineering and enhances the model's predictive capabilities. Furthermore, LSTMs are proficient in capturing complex relationships within datasets, which is essential given the diverse content of tweets and the multifaceted factors influencing Bitcoin prices. Leveraging historical Bitcoin price data alongside tweet metadata, LSTM networks can effectively forecast future price movements, leveraging their proven effectiveness in time-series prediction tasks.

Implementation

We created two LSTM models: 1) Using only Tweets as features 2) Using Tweets, Importance Coefficient, and Bitcoin volume traded as features. We will focus on the latter to explain the implementation.

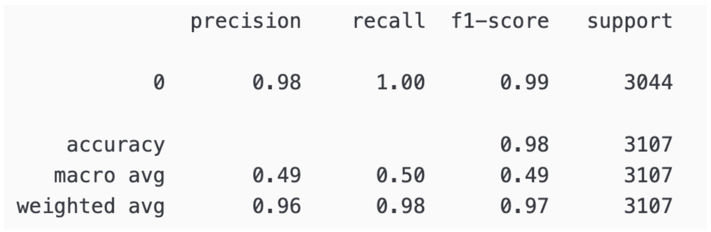
In order to embed the text for LSTM, a word index is built for all the tweets in the dataset and then all tweets are tokenized. All vectors are then padded, standardized using the StandardScaler(), after which they are stacked together with the other numerical data like Importance Coefficient and Bitcoin volume into a single numpy array to be used as input.

The LSTM is basically a sequence model with two LSTM layers and a Dense output layer. The LSTM layers have dropout and recurrent dropout to prevent overfitting, and L2 regularization is used to penalize large weights. The model is compiled with the Adam optimizer, binary cross entropy loss (since it's a binary classification task), and accuracy as the metric.

After defining the architecture, the model is trained for 10 epochs with a batch size of 10, and the validation data is the test set. The dataset trained is split but is not randomised as the data is time sensitive.

3.1.3 Results

We report on the test result from the train-test fit training. Surprisingly, both variations of the model produced the same performance scores. The result is show below:



```
              precision    recall  f1-score   support

           0       0.98      1.00      0.99      3044

    accuracy                           0.98      3107
   macro avg       0.49      0.50      0.49      3107
weighted avg       0.96      0.98      0.97      3107
```

Figure 5: Tweets LSTM Train-Test Results

3.2 News + BTC Prices

Note: The implementation of models using News dataset can be found in the Exploration: News & Tweets folder.

In this combination of datasets, we investigated the relationship between sentiments from news articles and Bitcoin prices. Since the focus for this section is to try different models to look for the best performing model, we focused on the sentiment analysis instead of textual input:

- Sentiment polarity: emotional orientation or attitude expressed in the text.
- Sentiment subjectivity: degree to which the sentiment expressed in the text reflects personal opinions, beliefs, or feelings rather than factual information.

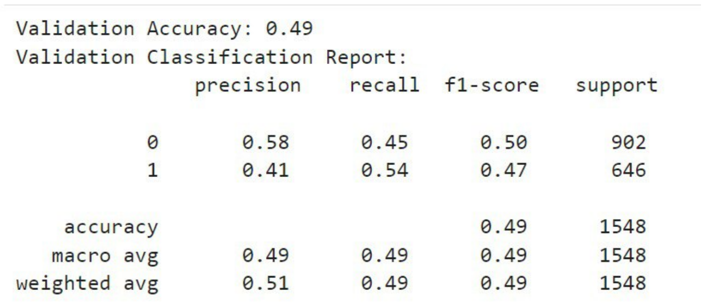
3.2.1 Data Processing

Data processing is done in a same fashion as the steps outline in the previous section on Tweets. However we found there is a class imbalance where the number '0' and '1' outputs in the dataset is skewed. This can reduce the accuracy and generalisation of our models. As such, we introduced the under-sampling and oversampling pipeline to our model. Resampling was performed using the Synthetic Minority Over-sampling Technique (SMOTE) to generate synthetic samples for the minority class.

3.2.2 Model 1: Logistic Regression

We chose a logistic regression model initially due to its ability to handle binary classification tasks. From the result score of 0.49, we hypothesised that using sentiments as features may not be a good indicator of price movement. Nevertheless, we continued to use sentiments for our next model as trial.

Results



```
Validation Accuracy: 0.49
Validation Classification Report:
              precision    recall  f1-score   support

           0       0.58      0.45      0.50       902
           1       0.41      0.54      0.47       646

    accuracy                           0.49      1548
   macro avg       0.49      0.49      0.49      1548
weighted avg       0.51      0.49      0.49      1548
```

Figure 6: Logistic Regression Train-Test Results

3.2.3 Model 2: Neural Network

Next, in our model exploration, we attempted a simple neural network. After some preliminary research, there were varying implementations that could work for a binary classification task on sequential data like stock market. We went ahead with this simple implementation as a way to test how well our chosen datasets would train on such a model architecture.

Architecture

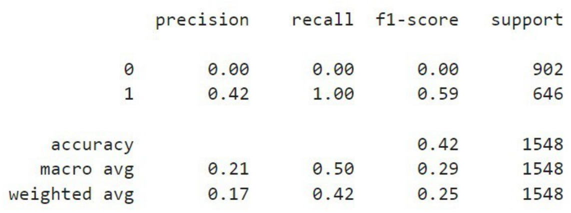
The neural network consists of multiple fully connected dense layers with activation functions to introduce non-linearity. Here are some main features of the neural network

- **Dropout Layer**: Introduced a dropout layer with a dropout rate of 0.6 after the first hidden layer to regularize by randomly setting a fraction of input units to zero during training to prevent overfitting.
- **Loss Function and Optimizer**: The binary cross-entropy loss function (BCELoss) is used to compute the loss between predicted and true labels. The Adam optimizer (Adam) is employed to update the model parameters based on the computed gradients, with a learning rate of 0.01.
- **Activation Function (ReLU)**: The Rectified Linear Unit (ReLU) activation function introduces non-linearity to the network by applying the element-wise rectification operation.

The model is trained over 15 epochs. In each iteration, the optimiser's gradients are zeroed, and forward and backward passes are performed to compute the loss and update the model parameters.

Results

The train-test score is 0.42, which shows that the model does not perform the best. This re-enforces the hypothesis that sentiment may not be a good indicator of Bitcoin price movement.

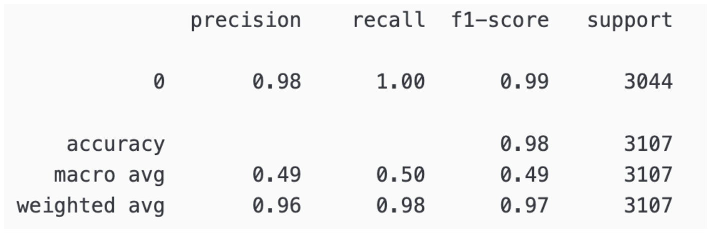


```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00       902
           1       0.42      1.00      0.59       646

    accuracy                           0.42      1548
   macro avg       0.21      0.50      0.29      1548
weighted avg       0.17      0.42      0.25      1548
```

Figure 7: Neural Network Train-Test Results

3.2.4 Model 3: LSTM

For this implementation, we passed the News textual data into the LSTM model as mentioned in the tweets section after discovering sentiments are not good features for prediction. We saw that the results of the train-test using the News dataset for the LSTM model is the same as that of Tweets.



```
              precision    recall  f1-score   support

           0       0.98      1.00      0.99      3044

    accuracy                           0.98      3107
   macro avg       0.49      0.50      0.49      3107
weighted avg       0.96      0.98      0.97      3107
```

Figure 8: News LSTM Train-Test Results

3.3 S&P500 Index + Interest Rate + BTC Prices

Note: The implementation of models using SP 500 and Interest Rate dataset can be found in the Exploration: SP Interest rates folder.

For this combination, we are investigating the effect of S&P 500 index and US Federal interest rate can influence the price movement of bitcoin. The models that we used are LSTM and Support Vector Machines (SVM).

3.3.1 Data Processing

The data was loaded from Yahoo Finance S&P 500, BTC-USD, and US Federal Interest Rate. Missing S&P 500 data was forward-filled with the last available trading day's values for days when the market is closed to ensure continuity and consistency in the dataset, as it prevents gaps in the time series data caused by market closures. The BTC-USD Closing price column and S&P 500 Closing value column were selected from individual dataframes containing S&P 500 and BTC-USD respectively. BTC-USD Closing price, S&P 500 Closing value, and US Federal Interest Rate data were then merged on the 'Date' column. A new column, 'BTC-USD Next Day Close', was created to capture the next day's closing price of BTC-USD by shifting the 'Close' column one row forward. Missing values in the last row of the 'BTC-USD Next Day Close' column were replaced with its next day closing price. A 'Price Change Indicator' column was generated to classify price changes as 'increase' if the next day's closing price was higher than the current day's, or 'decrease' otherwise. Column names were renamed to enhance clarity and consistency. Min-max normalization was applied to standardize the scale of selected columns.

The chosen models were Support Vector Machines (SVM) and Long Short-Term Memory (LSTM). SVM is favored for its ability to handle complex relationships and non-linear decision boundaries, making it suitable for analyzing financial data. On the other hand, LSTM excels in capturing temporal patterns and dependencies in time series data, making it well-suited for stock price prediction tasks. Each model's implementation and results are described below.

3.3.2 Model 1: SVM

In our approach to training Support Vector Machines (SVM) with polynomial (Poly) and radial basis function (RBF) kernels, we first employed a train-test split methodology to partition the dataset. This involves allocating 80% of the dataset for training purposes, allowing the model to learn patterns and relationships within the majority of the data, while reserving the remaining 20% for testing to evaluate the model's performance on unseen data. Subsequently, we utilized grid search coupled with cross-validation to optimize the SVM's hyperparameters. Specifically, we employed 10-fold cross-validation, dividing the dataset into 10 equal subsets (folds) and iteratively trained the model on 9 folds while using the remaining fold for validation. By performing grid search within each fold, we systematically explored various combinations of hyperparameters, ensuring that the SVM model is robustly optimized for performance across different subsets of the data.

Results

This table below is the test result from the train-test fit training as well as the test result on the back-testing data. The results are reported below:

Model	Number of Features	Selected Features	Best Hyperparameters	k-fold	F1 Score (from k-fold cross validation)
SVM Poly Kernel	3	['BTC-USD Open','BTC-USD High','S&P500 Adj Close']	['svm_C:0.1','svm_coef0:2.0','svm_degree: 4']	10	0.5115
SVM Poly Kernel	8	['BTC-USD Open','BTC-USD High',‘S&P500 Close',‘S&P500 Open',‘S&P500 High',‘S&P500 Low', 'S&P500Adj Close', ‘Interest Rate']	['svm_C:1','svm_coef0:0.0','svm_degree: 4']	10	0.509
SVM Poly Kernel	5	['BTC-USD Open','BTC-USD High',‘S&P500 Close', 'S&P500Adj Close', ‘Interest Rate']	['svm_C:10','svm_coef0:2.0','svm_degree: 4']	10	0.4978
SVM RBF Kernel	4	['BTC-USD Open','BTC-USD High', 'S&P500Adj Close', ‘Interest Rate']	['svm_C:10','svm_coef0:2.0','svm_degree: 4']	10	0.5238
SVM RBF Kernel	5	['BTC-USD Open','BTC-USD High', ‘S&P500 Close', 'S&P500Adj Close', ‘Interest Rate']	['svm_C:10','svm_coef0:2.0','svm_degree: 4']	10	0.5183
SVM RBF Kernel	3	['BTC-USD Open','BTC-USD High', ‘Interest Rate']	['svm_C:0.1','svm_coef0:2.0','svm_degree: 4']	10	0.5024

Table 1: SVM Training F1 Score

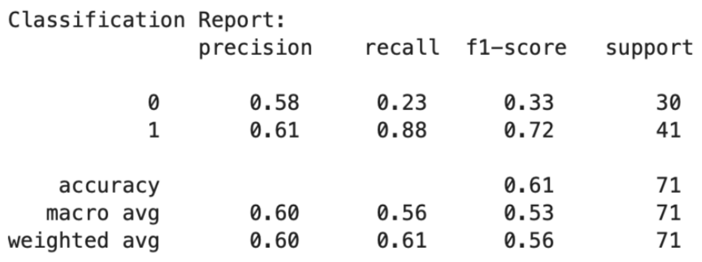
3.3.3 Model 2: LSTM

For this model, for the train-test split, 90% of the data was allocated for training, while the remaining 10% was used for testing. Before running through the LSTM model, data sequences were created for the train dataset and test dataset. This was done separately to prevent data leakage. Rolling windows of data sequences were also created. A short window frame was decided as the model was making short-term predictions. After trialing some window frames, a 10-day window was found to give the best eventual F1 score results. The train dataset of rolling windows was further randomly split into train (80% of all data) and validation (10% of all data) datasets, so that validation data can be used during model fitting.

The LSTM model consists of 2 LSTM layers and 2 dropout layers. Dropout regularisation was used to prevent overfitting. The loss function used in compiling the model was binary cross entropy, for this binary classification task. After prediction, the ROC-curve was used to find the best threshold that would be applied to the predicted probabilities, to obtain binary predictions.

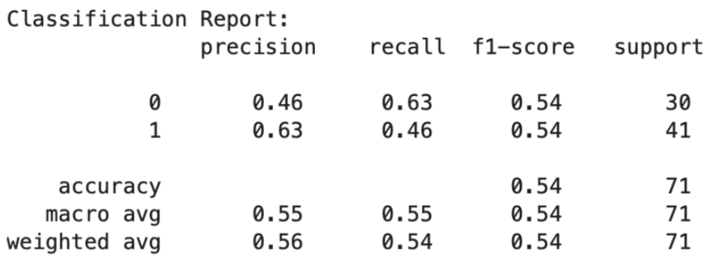
Results

In Iteration 1, adam optimizer and tanh activation function was used. To improve the model further, in Iteration 2, Grid Search was used to find the best combination of optimizer and activation function for the model. Along with this, stratified K-Fold was used to ensure that each fold (subset of the data) maintains the same class distribution as the original dataset to avoid a biased evaluation due to imbalanced class distributions. Based on the best parameters (according to the best F1 score), the model was refitted and tested. The best parameters found were adam optimizer and sigmoid activation function.



```
Classification Report:
              precision    recall  f1-score   support

           0       0.58      0.23      0.33        30
           1       0.61      0.88      0.72        41

    accuracy                           0.61        71
   macro avg       0.60      0.56      0.53        71
weighted avg       0.60      0.61      0.56        71
```

Figure 9: Iteration 1 Result



```
Classification Report:
              precision    recall  f1-score   support

           0       0.46      0.63      0.54        30
           1       0.63      0.46      0.54        41

    accuracy                           0.54        71
   macro avg       0.55      0.55      0.54        71
weighted avg       0.56      0.54      0.54        71
```

Figure 10: Iteration 2 Result

The results of train-test are tabulated below.

Model	Number of Features	Selected Features	Hyperparameters	Train-Test F1 Score
LSTM	3	['BTC-USD Close', 'S&P500 Close', 'Interest Rate']	{'activation': 'tanh', 'optimizer': 'adam'}	0.36
LSTM	3	['BTC-USD Close', 'S&P500 Close', 'Interest Rate']	{'activation': 'sigmoid', 'optimizer': 'adam'}	0.64

Table 2: LSTM Training F1 Score

Both SVM and LSTM methods were evaluated after hyperparameter tuning, with the highest F1 score achieved for SVM being 0.52 and for LSTM being 0.61. Being slightly above 0.5, these evaluation results demonstrate a moderate effectiveness of both approaches in predicting Bitcoin price trends. From this, it can be deduced that interest rate and S&P 500 index might not closely influence the Bitcoin price movement.

3.4 BTC Prices

Note: All other visualisations can be found in the Visualisation of Datasets folder.

In this dataset, we will focus on exploring the effects of BTC Open, High, Low, Close, Volume and the below technical indicators on the bitcoin price movement:

1. RSI (14 day) which indicates speed and magnitude of price change
2. MACD (26 and 12 day moving average convergence divergence) which signals strength of trend or trend reversals
3. 20 Day Bollinger bands which signal volatility
4. On Balance Volume (OBV) which measures positive and negative volume flow and analyzes the trading direction

We will also be investigating 2 other non-technical indicators :

1. High-Low (current day)
2. Close-Open (current day)

3.4.1 Data Preprocessing

In the data pre-processing step, we first created columns for the above technical indicators using TA-Lib. For each of the feature columns, we normalize it using MinMaxScalar before feeding it into the model. We also created an output column consisting of 1s and 0s where 1 represents that the next day closing price is higher than today's closing price.

We split the dataset into k folds where 80 percent of the fold was used for training and the next 20 percent was used for testing. We experimented with both k=5 and k=10 folds. K-fold blocking time series split (where the folds do not overlap) is used instead of the regular k-fold which may not be as suitable for time series data. This is because models trained on future data should not be used to predict past data.

3.4.2 Model

We considered 3 types of models namely Support vector machines (SVM), Random Forest (RF) and AdaBoost. Our initial approach involves selecting features manually based on the correlation coefficients, however the results were poor and the combinations tried were non-exhaustive and time consuming. Therefore, we came up with a new approach below to automate the feature selection and hyper parameter search for each model:

1. Select n best features using SelectKBest with mutual info regression since we expect the dependencies to be non-linearly correlated due to the complexity of bitcoin price movement.
2. Pass these n features into the model, conduct grid search with cross-validation, select the best hyper parameters based on aggregated F1 score over k-folds
3. For the best estimator of each n, print its hyper parameters, features selected as well as scores
4. Repeat for different n (3 to max number of feature columns in dataset)

5. Save the weights of the best model

After experimenting with all models and k-folds, the weights of each of the best models are saved and loaded to test on the back testing dataset.

3.4.3 Result

We report on the test result from the train-test fit training as well as the test result on the back-testing data. The results are reported below:

Overall, the top 3 model performances based on backtested dataset were the SVM-poly kernel for k-fold=5 and 10 (F1 score of 0.7 and 0.6 respectively) and random forest for k-fold=10 (F1 score of 0.58)

Model	Number of Features	Selected Features	Best Hyperparameters	k-fold	F1 Score (from k-fold cross validation)
SVM-poly kernel	14	['Open', 'High', 'Close', 'Adj Close', 'Volume', 'RSI_14', 'macd', 'macd_signal', 'BB_middle', 'BB_lower', 'Close-Open', 'High-Low', 'OBV', 'BB_height']	{'svm__C': 1, 'svm__coef0': 1.0, 'svm__degree': 4}	5	0.5922
SVM-poly kernel	12	['Open', 'High', 'Close', 'Adj Close', 'Volume', 'RSI_14', 'macd', 'macd_signal', 'BB_lower', 'HighLow', 'OBV', 'BB_height']	{'svm__C': 0.1, 'svm__coef0': 0.0, 'svm__degree': 4}	10	0.4337
SVM-rbf kernel	4	['Volume', 'macd', 'macd_signal', 'OBV']	{"svm__C': 10, 'svm__gamma': 100}	5	0.5296
SVMrbf kernel	15	['Open', 'High', 'Close', 'Adj Close', 'Volume', 'RSI_14', 'macd', 'macd_signal', 'BB_upper', 'BB_middle', 'BB_lower', 'CloseOpen', 'High-Low', 'OBV', 'BB_height']	{'svm__C': 10, 'svm__gamma': 10}	10	0.48205
Random Forest	12	['Open', 'High', 'Close', 'Adj Close', 'Volume', 'RSI_14', 'macd', 'macd_signal', 'BB_lower', 'HighLow', 'OBV', 'BB_height']	{'RF__max$_{d}epth'$: $None,'RF\_\_min\_samples\_leaf'$: $1,'RF\_\_min\_samples\_split'$: $5,'RF\_\_n_{e}stimators'$: 50}	5	0.3887
Random Forest	15	['Open', 'High', 'Close', 'Adj Close', 'Volume', 'RSI_14', 'macd', 'macd_signal', 'BB_upper', 'BB_middle', 'BB_lower', 'CloseOpen', 'High-Low', 'OBV', 'BB_height']	{'RF__max$_{d}epth'$: $10,'RF\_\_min\_samples\_leaf'$: $2,'RF\_\_min\_samples\_split'$: $5,'RF\_\_n_{e}stimators'$: 50}	10	0.4804
Adaboost	15	['Open', 'High', 'Close', 'Adj Close', 'Volume', 'RSI_14', 'macd', 'macd_signal', 'BB_upper', 'BB_middle', 'BB_lower', 'CloseOpen', 'High-Low', 'OBV', 'BB_height']	{'adaboost__learning$_{r}ate'$: $1.0,'adaboost\_\_n\_estimators'$: 100}	5	0.4695
Adaboost	15	['Open', 'High', 'Close', 'Adj Close', 'Volume', 'RSI_14', 'macd', 'macd_signal', 'BB_upper', 'BB_middle', 'BB_lower', 'CloseOpen', 'High-Low', 'OBV', 'BB_height']	{'adaboost__learning$_{r}ate'$: $0.5,'adaboost\_\_n_{e}stimators'$: 100}	10	0.5532

Table 3: BTC Model F1 Training Results

4 Results and Discussion

With our methodology of constructing smaller models trained on subsets of the data, our objective was to evaluate model performance and explore strategies for integrating model outputs into a final optimized model. Through the utilization of ensemble learning techniques, we sought to select the best performing models as parts to form an ensemble final model. This provides additional robustness to noise, mitigating potential distortions in the final model predictions, and enhancing overall generalization capabilities.

To select the best model, a new backtesting dataset is created. This dataset consist of 30 datapoints of:

- News from Kaggle
- Tweets manually taken from Twitter website under the bitcoin hashtag with dates matching the News date
- Bitcoin data, US Interest Rates, S&P 500 from Yahoo! Finance

This method was preferred due to the constraints in data which made it difficult to find datasets with coinciding dates for both the tweets and news before our train dataset time period. All model variations were run on this backtesting data. The best performing models for each combination are summarised in the table below:

Explore	Model	Selected Features	F1 Score	Backtest
Tweets	LSTM	['News','Coefficient value','Bitcoin value']	0.97	0.57
News-BTC	LSTM	['News']	0.97	0.57
BTC-USD	SVM-Poly Kernel	['Open', 'High', 'Close', 'Adj Close', 'Volume', 'RSI_14', 'macd', 'macd_signal', 'BB_middle', 'BB_lower', 'Close-Open', 'HighLow', 'OBV', 'BB_height']	0.5922	0.70
S&P 500 and Interest rates	SVM-RBF Kernel	['BTCUSD Open', 'BTCUSD High', 'S&P500 Adj Close', 'Interest Rate']	0.51	0.57

Table 4: Best Performing Models for Each Combination with F1 and Backtest Score

As seen, the performance of the model reduces significantly when it runs on the back-testing dataset. A possible reason for this is because text tokenisation does not incorporate word morphological analysis for potentially unknown words which hinders the model's ability to parse new textual data. On the other hand, for numerical inputs, there are possible reasons for the drop in performance. Firstly, Bitcoin prices are relatively volatile. Secondly, the since our backtesting data dates 2020, the Covid-19 global outbreak played a major role in destabilising the global economy. Our models are not trained on data for dates where major global events took place.

Given the superior performance of the LSTM models for the sentiment analysis on the stock market and SVM from BTC-price dataset giving the best F1 scores, we decided to use their model outputs, as inputs to our final model.

4.1 Models Consolidation

We have devised two ways to combine our models together:

4.1.1 Approach 1: Stacking

Note: The implementation of models in the Stacked Approach can be found in the Stacked Approach folder.

For stacking approach, we tried to use the output of individual models (from tweet and news sentiment and BTC-price models) as the inputs for the final model. We want to test if combining outputs from individual models would give better results.

4.1.2 Data Preprocessing

Firstly, using the saved weights from the individual models, we ran the train dataset through the models to derive each models output. We combined all the 3 model's output into a single data frame. We repeated this for the back testing dataset as well. For both dataframes, we also compute the output column of either 0 or 1. Then, we split the dataset into k folds where 80 percent of the fold is used for training and the next 20 percent is used for testing. This time, we experimented with k-fold=5 with blocking time series split.

4.1.3 Model

A similar model training approach as the BTC-price model is conducted with SVM, Random Forest and Adaboost using SelectKBest and GridSearchCV and weights of each of the best models are saved and loaded to test on the back testing dataset, producing the below results:

Model	Number of Features	Selected Features	Best Hyperparameters	k-fold	F1 Score (from k-fold cross validation)	Backtest
SVM-poly kernel	3	['predictions_BTC_only', 'prediction_Tweet_only', 'prediction_News_only']	{śvm__C': 10, 'svm__coef0': 2.0, 'svm__degree': 3}	5	1	0.42
SVM-rbf kernel	3	['predictions_BTC_only', 'prediction_Tweet_only', 'prediction_News_only']	{śvm__C': 10, 'svm__gamma': 100}	5	1	0.42
RF	3	['predictions_BTC_only', 'prediction_Tweet_only', 'prediction_News_only']	{'RF__max_depth': None, 'RF__min$_s amples_l eaf'$: $1,' RF\_\_min_s amples_s split'$: $2,' RF\_\_n\_estimators'$: 50}	5	1	0.42
Adaboost	3	['predictions_BTC_only', 'prediction_Tweet_only', 'prediction_News_only']	{ádaboost__learning_rate': 0.1, 'adaboost__n_estimators': 50}	5	1	0.42

Table 5: Approach 1 Scores

4.1.4 Approach 2: Combining into One Big Model

Note: The implementation of models in the Combined Approach can be found in the Combined Approach folder.

In the second approach, we want to test if using raw inputs from all datasets would be better since stacking did not produce ideal results.

4.1.5 Data Preprocessing

We first joined all the raw datasets together, used TA-Lib to compute technical indicators (same as in BTC-price dataset) as well as compute the output column and normalized the feature columns .

We split the dataset into k folds where 80 percent of the fold is used for training and the next 20 percent is used for testing. This time, we experimented with k-fold=5 and 10 with blocking time series split.

4.1.6 Model

A similar model training approach as the BTC-price model is conducted with SVM ,Random Forest and Adaboost using SelectKBest and GridSearchCV and weights of each of the best models are saved and loaded to test on the back testing dataset.

4.1.7 Result

The best backtest F1 scores we achieved were 0.57 for SVM-poly kernel, 0.57 for adaboost and 0.53 for random forest.

Model	Number of Features	Selected Features	Best Hyperparameters	k-fold	F1 Score (from k-fold cross validation)
SVM Poly Kernel	21	['Open', 'High', 'Close', 'Adj Close', 'Volume', 'macd', 'macd_signal', 'BB_middle', 'BB_lower', 'HighLow', 'OBV', 'BB_height', 'sentiment_polarity', 'sentiment_subjectivity', 'Interest Rate', 'SP_Open', 'SP_High', 'SP_Low', 'SP_Close', 'SP_AdjClose', 'SP_Volume']	{'svm__C': 1, 'svm__coef0': 0.0, 'svm__degree': 4}	10	0.48047
SVM-poly kernel	24	['Open', 'High', 'Close', 'Adj Close', 'Volume', 'RSI_14', 'macd', 'macd_signal', 'BB_upper', 'BB_middle', 'BB_lower', 'CloseOpen', 'HighLow', 'OBV', 'BB_height', 'sentiment_polarity', 'sentiment_subjectivity', 'Interest Rate', 'SP_Open', 'SP_High', 'SP_Low', 'SP_Close', 'SP_AdjClose', 'SP_Volume']	{'svm_C': 0.1, 'svm__coef0': 0.0, 'svm__degree': 3}	5	0.5482
SVM-rbf-kernel	5	['Open', 'High', 'Close', 'Adj Close', 'Volume', 'RSI_14', 'macd', 'macd_signal', 'BB_upper', 'BB_middle', 'BB_lower', 'Close-Open', 'High-Low', 'OBV', 'BB_height', 'sentiment_polarity', 'sentiment_subjectivity', 'Interest Rate', 'SP_Open', 'SP_High', 'SP_Low', 'SP_Close', 'SP_AdjClose', 'SP_Volume']	{'svm__C': 10, 'svm__gamma': 100}	10	0.6066
SVM-rbf-kernel	18	['Open', 'High', 'Close', 'Adj Close', 'Volume', 'macd', 'macd_signal', 'BB_middle', 'BB_lower', 'HighLow', 'OBV', 'BB_height', 'sentiment_subjectivity', 'Interest Rate', 'SP_High', 'SP_Close', 'SP_AdjClose', 'SP_Volume']	{'svm__C': 10, 'svm__gamma': 10}	10	0.5678

Table 6: Approach 2 Result pt.1

Model	Number of Features	Selected Features	Best Hyperparameters	k-fold	F1 Score (from k-fold cross validation)
Random Forest	14	['Open', 'Close', 'Adj Close', 'Volume', 'macd', 'macd_signal', 'BB_middle', 'BB_lower', 'High-Low', 'OBV', 'BB_height', 'sentiment_polarity', 'sentiment_subjectivity', 'SP_High']	{'RF__max_depth': 10, 'RF__min_samples_leaf': 4, 'RF__min_samples_split': 10, 'RF__n$_e stimators' : 100$}	10	0.6433
Random Forest	3	['Open', 'macd', 'BB_middle']	{'RF__max$_d epth' : 10,' RF\_\_min\_samples\_leaf' : 4,' RF\_\_min\_samples\_split' : 5,' RF\_\_n\_estimators' : 100$}	5	0.5871
AdaBoost	3	['Open', 'macd', 'BB_middle']	{'adaboost__learning$_r ate' : 0.5,' adaboost\_\_n\_estimators' : 50$}	5	0.5686
AdaBoost	8	['Open', 'Close', 'Adj Close', 'macd', 'BB_middle', 'HighLow', 'OBV', 'sentiment_subjectivity']	{''adaboost__learning_rate': 0.5, 'adaboost__n_estimators': 150}	10	0.5542

Table 7: Approach 2 Result pt.2

4.2 Results and Implications

Overall, we decided to evaluate which is the best model across individual models and aggregated models based on F1 score on the back tested set for a fair comparison to be used in our GUI. Thus , the best model we obtained was the SVM(poly kernel) model trained on the BTC-USD dataset with a back tested F1 score of 0.70 with features 'Open', 'High', 'Close', 'Adj Close', 'Volume', 'RSI_14', 'macd', 'macd_signal', 'BB_middle', 'BB_lower', 'Close-Open', 'High-Low', 'OBV', 'BB_height.

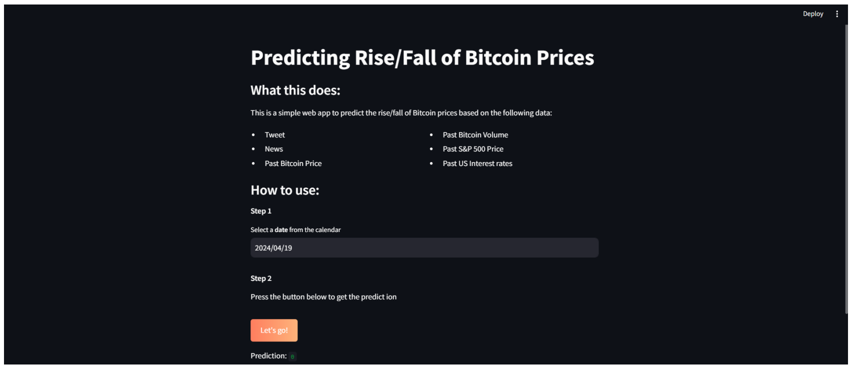
While the aggregated models and sentiment models did not work as well as we intended to based on the backtest dataset, it is important to note that this could be due to the lower quality data of the backtested dataset since we had to manually pick out tweets due to the time period constraints of the dataset, also leading to lower quantity of backtesting data. Additionally, the tweets manually picked were not from the most influential people unlike the tweet dataset used for training, thus causing the models to not perform as well. Given more time, expanding and improving the quality of our back testing dataset is an area of improvement for our group.

5 Graphical User Interface

The graphical user interface (GUI) was implemented using the Streamlit python module. Since the final model chosen does not involve textual data, users simply need to select a date and press the button. The result of the prediction can be interpreted:

- If prediction = 0 : The price of Bitcoin will not increase or fall the next day
- If prediction = 1: The price of Bitcoin will increase the next day

The below picture shows a screen capture of the GUI.



Predicting Rise/Fall of Bitcoin Prices

What this does:

This is a simple web app to predict the rise/fall of Bitcoin prices based on the following data:

- Tweet
- News
- Past Bitcoin Price
- Past Bitcoin Volume
- Past S&P 500 Price
- Past US Interest rates

How to use:

Step 1

Select a **date** from the calendar

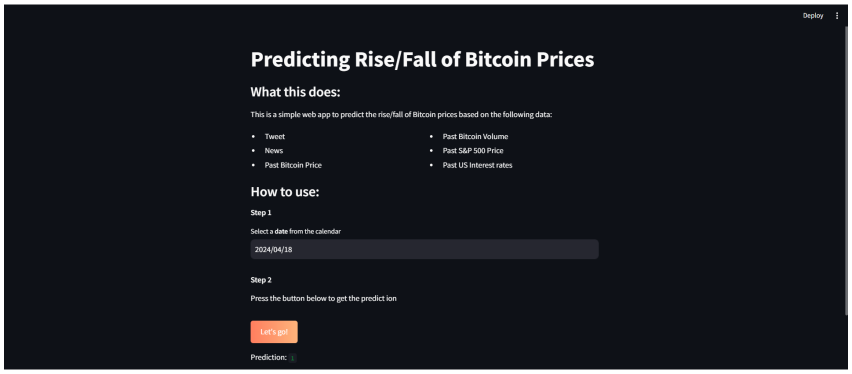
2024/04/19

Step 2

Press the button below to get the predict ion

Prediction: 0

Figure 11: GUI Prediction=0



Predicting Rise/Fall of Bitcoin Prices

What this does:

This is a simple web app to predict the rise/fall of Bitcoin prices based on the following data:

- Tweet
- News
- Past Bitcoin Price
- Past Bitcoin Volume
- Past S&P 500 Price
- Past US Interest rates

How to use:

Step 1

Select a **date** from the calendar

2024/04/18

Step 2

Press the button below to get the predict ion

Prediction: 1

Figure 12: GUI Prediction=1

6 Conclusions

Overall, this study allowed the team to comprehensively study and test out a variety of possible information and therefore features that can affect the price movement of crytocurrencies, in this case Bitcoin. While the results of the study reflects that financial data like US Interest Rates and S&P 500 price index has a more reliable influence on the price movement, we believe that further pursuit on experimentation with textual data can be fruitful.

Thus, a further iteration of this study could be to implement large language models to better parse textual data with better contextual and semantic understanding. Additionally, to expand further, perhaps more combinations of features and model variations could also provide a more comprehensive comparison for the best feature-model combination for usage.

We believe that possible implementations of the work from this project can be integrated in intelligent financial systems. Especially with the steady rise of cryptocurrency and decentralised financein the past decade, these systems can be proven helpful in taking proactive approach in flagging irregularities in the market at a large scale. At a personal scale, it can be integrated into personalised trading agents.