TEXAS A&M UNIVERSITY
J. Mike Walker ’66 Department of Mechanical Engineering

Project 2:

Forecasting Cryptocurrency Prices with Social Media Sentiment Analysis

Authors: Alvaro Guerra, Tyler Hveem, Will Grubbs, Ryan Li, Brae Barnes
Instructor: Dr. Wei Chen
Team 13
MEEN 423 500 Spring

Abstract:

The project explored Bitcoin price forecasting by applying historical market data and Twitter sentiment analysis. Due to the high volatility of cryptocurrency and the influence of crowd sentiment, the group combined traditional technical indicators. By analyzing BTC price and Twitter dataset the team was able to create techniques to anticipate price movements, and simulate real-world trading. BTC historical price data along with Twitter sentiment data was analyzed using XGBoost, Random Forest, and LSTM models. The goal of the model was to be able to take in both datasets to train with and then make predictions of future closing prices using only the Twitter sentiment data and its own BTC price predictions. Model training was done using BTC price data along with Twitter sentiment data spanning from around the end of 2014 to the end of 2017, and model testing was done using Twitter data from 2018 and 2019. The Random Forest model was selected for its stability and strong performance, achieving an R-squared value of 0.7008. This cryptocurrency forecasting project also revealed findings that sentiment analysis improves the model's ability to capture short-term price movements, but decreases accuracy during periods of extreme volatility.

Methodology:

The first component of creating this ML model was the data collection. In order to have a wide range of dates to train the model on, four years of data was collected. There are two main data sets, the bitcoin (BTC) closing price for each day from 2015-2019 [1], and tweets talking about BTC from 2014-2019 [2]. The tweet per day data set was massive, with some days having up to 100,000 tweets and the entire data file taking up around 3 gigabytes of storage. In order to trim this file to a more manageable size and make feature creation easier, a separate csv summary file was created. In order to create this tweet summary file, the Valence Aware Dictionary and Sentiment Reasoner (VADER) was used to create a numerical sentiment score of each tweet. Based on that sentiment score, each tweet was classified as positive, negative, or neutral. The number of each category of tweet as well as the average numerical sentiment score was saved to the new DailySentimentSummary.csv file. This compact summary file was then used to create features for ML training in tandem with the BTC closing price data.

The daily closing price from the Bitcoin data as well as the raw average sentiment for each day using VADER is shown below in **Figure 1**. This is only a small cross section of the total five year data.

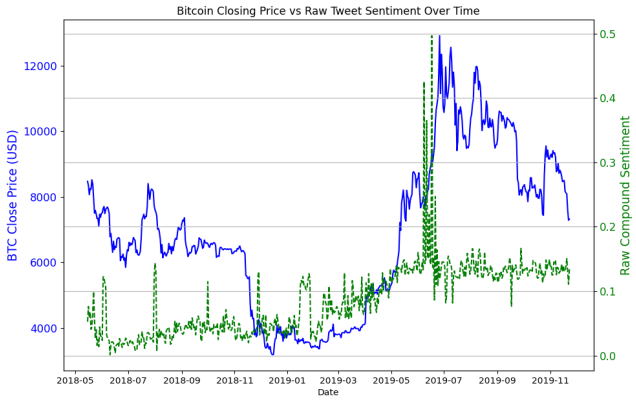

Figure 1: BTC Closing Price and Raw Sentiment Scores per Day

The team focused on creating three different models to test and see which models would perform the best. These models were a Random Forest, XGBoost, and Long Short-Term Memory Neural Network (LSTM). Random Forest and XGBoost were both tested because these are tree based models that the team thought would be able to handle the tabular features that the team planned on using for the model training. They also would both be able to forecast using time-dependent financial data that can be very volatile and noisy. The team was worried that there wouldn't be enough data for the XGBoost to perform well, which is why both of these were tested even though they have some similarities. The LSTM model was tested because this model is built for a time series such as this and so the team figured that this model would be able to perform well when forecasting future prices based on past trends. The models were trained using a combination of the BTC price data and Twitter sentiment data from around the end of 2014 to the end of 2017, and then they were evaluated based on how well they were able to predict future price data in 2018 and 2019 using only the Twitter sentiment data and their own previous forecasting. The models were evaluated using mean squared error and R^2 values.

For feature engineering, the raw tweet count and closing price series were first transformed into a consolidated set of time-series predictors, blending social media sentiment and cryptocurrency dynamics. The features that encapsulate the sentiments involve normalizing the VADER output into sentient ratios (positive and negative over total tweets for a given day). To reduce noise and capture short-term trends, features were incorporated with one-day lags of both compound sentiment score and closing price, along with three-day rolling averages of each. A simple momentum metric was also defined as the current day's close minus the previous day's close. More features were added to enrich the feature set: daily-percent change and seven-day price volatility reflected local variability, and binary trend up and trend down flags for big price and sentiment shifts. Lastly, five classic technical indicators were added: MACD (12-day/26-day EMA difference) to track trend momentum, RSI14 to flag overbought and oversold conditions, Bollinger to measure any compression in the market, ATR14 to gauge general volatility, and on-balance volume to aggregate buying and selling pressure.

The XGBoost and Random Forest models both used hyperparameter tuning for certain hyperparameters depending on the model. In each case, RandomizedSearchCV was used to find the best hyperparameters and the R^2 value was used to optimize. The Random Forest model optimized the hyperparameters n_estimators, max_depth, min_samples_split, min_samples_leaf, and max_features, using 20 random combinations and a 3 fold TimeSeriesSplit. The values tested for each were based on what team members had used for RF models in the past and had success with. For XGBoost, the hyperparameters that were tested were n_estimators, learning_rate, max_depth, subsample, colsample_bytree, and gamma. This model also used 20 random combinations and a 3 fold TimeSeriesSplit and optimized based on R^2 value. The LSTM model didn't use any hyperparameter tuning. The hyperparameters that were tested for the Random Forest model can be seen in **Table 1** below. The three models can be seen in the submitted code (**A.5, A.6, A.7**).

Table 1: Hyperparameter Tuning

Hyperparameter	Values Tested
n_estimators	**100**, 200, 300**,** 500
max_depth	5, **10**, 15, 20
min_samples_split	2, **5**, 10
min_samples_leaf	1, **2**, 4
max_features	sqrt, **log2**

Results and Discussion:

The final tuned Random Forest model (with hyperparameters n_estimators = 100, max_depth = 10, min_samples_split = 5, min_samples_leaf = 2, max_features = log2) yielded the highest out-of-sample R^2 value of 0.7008 with a Mean Squared Error of 2021834.19. This model and other models are outlined in **Table 2**. It is important to note that the LSTM model performs differently each time, so that is the best performance the group was able to get from it.

Table 2: Model Performance

Model	R^2	MSE
XGBoost	0.6946	2063875.95
Random Forest	**0.7008**	**2021834.19**
LSTM	0.6138	2692072.46

The predicted vs actual Bitcoin prices reveal that the Random Forest tracks short-term trends and general direction behavior well; however, larger deviations occur during higher volatility time periods. This suggests that further robustness techniques, such as uncertainty quantification, could be valuable. **Figure 2** illustrates the accuracy at which the model tracks the price of BTC.

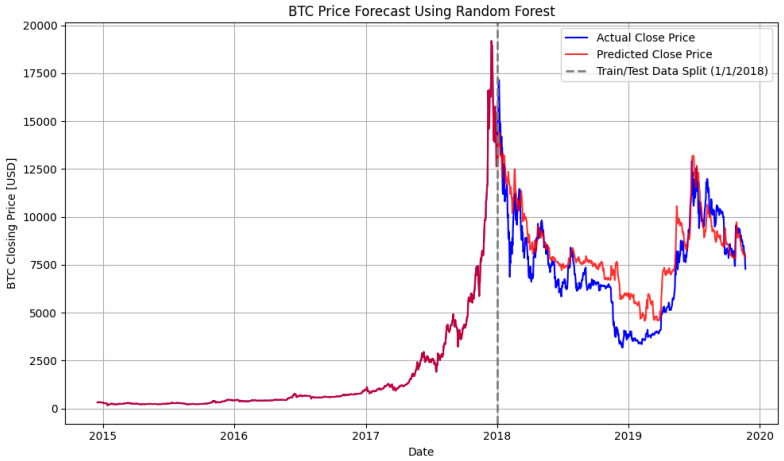

Figure 2: BTC price forecast with Random Forest Model

It can be observed that the model tracks the training data (pre-2018 cutoff) extremely closely, since it is fed features from the BTC price data, such as opening and closing price of the previous day. After the 2018 cutoff, the model predicts the behavior with just the features, knowing only the closing price at the training boundary. It can also be observed that the training data generally trends upwards. However, the model is still able to predict the general downward trend even after the cutoff, relying on just the closing price at the boundary, Twitter sentiment, and related features.

Overall, the group was pleasantly surprised by how well the model was able to perform with the test data, achieving a relatively high R^2 and following the actual price fairly closely when looking at the graph. The model is flatter than the actual price data in most situations, as stated before, but the group was expecting the model to keep increasing based on what it saw in the training data. However, the model was able to follow the trend very closely, showing the correlation between the sentiment data and the BTC price, as well as showing that the features chosen by the group were very beneficial to the model.

Conclusion:

In conclusion, the Random Forest model performed very well using Twitter Sentiment data along with BTC price data to train the model and then predicting BTC prices from 2018-2019. A limitation of this model is that it uses its own predictions for the BTC price data during testing, so if its predictions start to get off they will likely continue to get further away from the actual price. Also, the Random Forest model would struggle to predict prices outside the range of the training data, so if new highs or lows are hit, it wouldn't be able to predict them. Luckily, the testing data never reached a new high or low compared to the training data so the model didn't struggle with this.

In terms of future work, one thing would be to update the model to be able to predict tomorrow's price only using all previously available Twitter and price data. Right now the model was set up to predict a two year span to show the capabilities of the model for the sake of the project, so it would need to be adjusted to be able to predict one day in the future. Another thing would be to find a way to scrape and take in more present twitter data, as the data we found ended before 2020. Lastly, creating models for other cryptocurrencies that are currently available to trade. This model could ultimately be used by day traders to predict how Bitcoin prices are going to perform on a daily basis based on Twitter sentiment and make large amounts of money considering the volatility of Bitcoin.

References:

[1] https://www.kaggle.com/datasets/prasoonkottarathil/btcinusd
[2] https://www.kaggle.com/datasets/aisolutions353/btc-tweets-sentiment

Appendix: Include a brief description of all the files included in your zip (code and data files). Most will require no more than a sentence to describe, but important files may require a little more description.

A.1 LotsOfTweets.csv. This is the raw tweet data file from 2014-2019. **This file is 2.9 GB and it was going to take about 30 minutes to upload the zip file to Canvas with this csv file included, so we decided to leave it out. If you would like to see it, we can find another way to get it to you.**

A.2 DailySentimentSummary.csv. This is the compressed tweet data file that contains the sentiment scores and number of tweets per category.

A.3 BTC-Daily.csv. The daily prices of bitcoin from 2015-2019. The file contains a few different data points but the closing price of bitcoin for each day was the metric used.

A.4 DataReduction.py. This is the file used to compress RawTweets.csv down to DailySentimentSummary.csv using VADER. Uses **A.1** as an input.

A.5 Proj2_RF.ipynb. This is the Random Forest model that the team tested and ultimately used. Necessary inputs for the model are **A.2** and **A.3**.

A.6 Proj2_XGBoost.ipynb. This is the XGBoost model that the team tested. Necessary inputs for the model are **A.2** and **A.3**.

A.7 Proj2_LSTM.ipynb. This is the LSTM model that the team tested. Necessary inputs for the model are **A.2** and **A.3**.