Forecasting Bitcoin Price Direction from Twitter Sentiment and Market Data

By Rishi Nambiar

Domain: Cryptocurrency, Virtual Markets, Finance.

Over the past decade, cryptocurrency markets have emerged as a highly volatile and sentiment-driven exchange. Bitcoin, the most prominent cryptocurrency, stands out; it is widely discussed and traded as a digital asset. Both traditional market forces and frequently changing narratives online, particularly on the social media platform X (formerly known as Twitter), influence Bitcoin's price movements. My project aims to predict whether the hourly price of Bitcoin will increase or decrease based on a sentiment analysis of tweets and related market data. This combination of social sentiments and financial data provides a path for AI to uncover and transcribe hidden patterns. With the democratization of investment platforms and increased access to them, combined with the rise of crypto influencers, it is becoming increasingly important to acknowledge how public perception affects market trends. This project is very interesting to me because I'm passionate about both finance, my future major, and AI in general. Predicting price direction using machine learning techniques combines these two interests in a unique and fascinating manner.

The dataset I used for this project is the Bitcoin 17.7 million Tweets and price dataset available on Kaggle, published by Jaime Badiola, and last updated 6 years ago. It combines market data for Bitcoin with sentiment metrics scores from X, formerly known as Twitter. The data spans from August 1, 2017, to January 21, 2019, at an hourly frequency with 12,936 rows and 16 columns. Variables include Date timestamp, market features such as Open, High, Low, Close, and trading volumes, and multiple sentiment metrics, including

Compound Score, Count Negatives, Count Positives, Count Neutrals, Sent Positives, and Sent Negatives. The sentiment features were directed using VADER, a lexicon-based algorithm created for short and informal texts (like tweets). Vader assigns each word a value score, adjusting for context like intensifiers and punctuation, then producing three proportion scores: positive, negative, and neutral. It also assigns a compound score ranging from -1 (most negative) to +1 (most positive). In this dataset, Compound_Score summarizes the general sentiment of all tweets given per hour, while Sent_Positives and Sent_Negatives emphasize the magnitude of sentiment in either direction. The count columns show tweet volume by respective sentiment category. These features allow the model to use both tone and intensity of social reactions alongside price data. I began by loading the dataset and investigating its structure using .info() and .head() to inspect the first few rows. Example rows revealed how the price data and sentiment score were aligned hourly. For example, an hour with a Compound_Score of .41 and positive price movements, versus one with a score of -.32 and a downward movement. I then visualized missing values using a heatmap, which showed that several sentiment and price columns had small gaps (around 4.5% missing). I then used a correlation heatmap to demonstrate the strong relationship among market features (Open, High, Low, Close) and weaker but still moderate correlations between other sentiment scores and market variables.

Once the dataset was loaded (through the file df_Final.csv), preprocessing began to ensure it was ready for modeling. All numeric columns were converted to float form. Timestamps were standardized into a datetime format, and the dataset was chronologically ordered for greater visualization and accuracy. Next, creation of Price Change or Close-Open, and Percent Change ((Close-Open)/Open*100). They both showed the absolute

and relative magnitude of hourly price movements. These features could aid understanding more than raw price values because they showed the volatility and direction between hours. I then used scatter plots to visualize the relationship between these new features and sentiment scores. To prepare for modeling, categorical labels in Direction were encoded, and all feature variables were standardized. The dataset was then split into training and testing sets, with 80% used for training and 20% saved for evaluation. This preserved the original balance in both data sets.

With the dataset being fully prepared, I split it into training and testing sets using an 80/20 split via stratification to keep the same class balance in both sets. The model was then trained with two machine learning models: a Random Forest Classifier and an XGBoost Classifier.

For the Random Forest, I used 100 decision trees (n_estimators=100) with a fixed random state to ensure results for reproducibility. The model was then trained on features created earlier, and predictions were made for the test set. Next, for the XGBoost model, I set the objective to binary classification (binary: logistic) and used the logloss evaluation metric. Similar to the Random Forest, the model was trained on the same training set and tested on the same set as well. I also created a feature importance plot to visualize which variables had the greatest impact on both sets of predictions. Both models were evaluated and built, and printed directly after training.

For the Random Forest model, the test accuracy was 56.31%. This could be seen through a confusion matrix, which showed that the model correctly classified a majority of the examples in the larger class; it struggled more with the minority class. The XGBoost model achieved a higher test accuracy of 74.15%. Its confusion matrix revealed a more

balanced performance between classes. To further investigate, a feature importance plot was created, which revealed that both market features (Close, High) and sentiment features (Compound_Score, Total Volume of Tweets) were significant factors in the model's decision-making. Finally, classification reports were generated for both models, providing precision, recall, and F1-scores for each class. This confirmed that XGBoost had a stronger overall performance, especially in balancing precision and recall across both classes.

For the next part of the project, I plan to add more features in order to give the models more useful information. This could include creating new variables that show recent trends, such as averages of price and sentiment over a greater period, such as a few hours. I also want to adjust the model settings (hyperparameters) to see if that affects accuracy.
When evaluating the models, I will look at more than just accuracy by checking out other measures like precision-recall to better understand how well the models predict both classes. Lastly, I could try to combine the two models in order to see if their predictions together outperform either by themselves.

References:

Data set: https://www.kaggle.com/datasets/jaimebadiola/bitcoin-tweets-and-price

Thesis Paper:

https://drive.google.com/file/d/1we5kywWQqQRYX0Hy8edsKKN7PdaaFuzY/view

alongside the Dataset

Vader Sentiment: https://github.com/cjhutto/vaderSentiment