Bitcoin Price Prediction Using News Sentiment and Technical Indicators

Abstract:

In this study, we aim to predict Bitcoin price movements by combining sentiment analysis on cryptocurrency-related news headlines and established technical indicators. Our methodology involves gathering news headlines, performing sentiment analysis, combining the sentiment with numerical indicators, and training machine learning models to predict price movements.

Data Collection:

News Articles Collection:

- News articles related to Bitcoin were collected by crawling bitcoin.com using a Selenium-based script. The script extracted article title, link, category, date, and description from multiple pages of the website. The data was saved in a JSON file for further processing.

Data Processing:

Sentiment Analysis:
The entire process of sentiment analysis and post-processing the sentiment data has been broken down into two separate scripts.

- **a.) Sentiment Analysis Script:**
- - This script performs sentiment analysis on news data and saves the results to a JSON file.
  It uses the Transformers library's pre-trained 'ProsusAI/finbert' model to analyze the sentiment of news titles.
  Sentiment labels are categorized as negative, neutral, or positive, with corresponding numerical values -1, 0, and 1.
  The sentiment scores and labels for each news title are stored in a JSON file organized by date.
  The script reads settings and paths from a configuration file config.ini, including paths for input and output files, and a testing mode flag.

- **b.) Sentiment Data Processing Script:**
- This script loads the raw sentiment data from a JSON file, applies multipliers to sentiment scores, calculates the arithmetic mean for each day's data, and saves the processed data back to a JSON file.
  It also generates a CSV file that contains the date and the corresponding arithmetic mean of the sentiment data. If a particular date is missing from the sentiment data, a default sentiment mean value (0.0) is assigned in the CSV file.
  This script also reads settings and paths from the same configuration file config.ini.
  (iii) Final Dataset Formation:
  The daily sentiment scores were combined with the numeric attributes to form a unified dataset. Each entry represents a day and includes numeric indicators and the sentiment score.

Machine Learning Algorithm Application:

- **a.) Data Loading and Preprocessing:**
- The data files, BTC-USD.csv and sentiment_means.csv, are loaded to form the foundation for the modeling process. The 'Close' column is shifted to create the target variable, where a rise in price is denoted as 1, and a fall is denoted as 0. The feature extraction process involves using the differences in closing prices and sentiment means as features for the model. The datasets are meticulously handled for missing values and are merged for a comprehensive and cohesive dataset for training and testing.

- **b.) Exploratory Data Analysis:**
- A graphical representation delineates the training and test splits over the time series data, offering a visual insight into the dataset division. The distribution of the target variable is explored to understand the balance and spread of the upward and downward movements in the dataset.

- **c.) Model Building:**
- Time Series Split is employed for generating the training and test sets, ensuring chronological order and time-based splitting for optimal time series modeling. StandardScaler is used in the normalization process, providing standardized features for the XGBoost model, which is discussed with its relevant parameters.

- **d.) Hyperparameter Tuning:**
- A defined space dictionary contains the chosen hyperparameters for tuning. The objective function is constructed to calculate the model's accuracy with various hyperparameter sets, returning the loss as a negative accuracy. Hyperopt's fmin function is instrumental in finding

the most effective set of hyperparameters for the XGBoost model, enhancing the model's predictive capabilities.

- **e.) Model Evaluation:**
- The evaluation of the model is conducted using a confusion matrix, with the normalized confusion matrix plot visualized for clearer insight into the model's performance. The accuracy score, a vital performance metric, is computed within the objective function during each iteration of hyperparameter tuning. All test set predictions and their corresponding true labels are aggregated for a comprehensive evaluation.

Results

Hyperparameter Tuning:
The Hyperopt results reveal various combinations of parameters for the XGBoost model. These parameters include colsample_bylevel, colsample_bytree, gamma, learning_rate, max_depth, n_estimators, reg_alpha, reg_lambda, and subsample. The loss values resulting from these different combinations hover around -0.529, with a few exceptions showing better or worse performance. The normalized_loss values also range diversely, providing a comprehensive insight into the robustness of the hyperparameter tuning process.

Analyze the patterns within the results to determine the most successful parameter combinations for this task. As an example, the configuration with colsample_bylevel=0.5, colsample_bytree=0.7, learning_rate=3.0, max_depth=3, and n_estimators=500 showed a lower loss of -0.509 and a normalized_loss of 1.0, signaling the most favorable combination among the tested configurations.

Confusion Matrix:

The confusion matrix results present the following outcomes for the predictions:

- True Negatives (0,0): 68329
- False Positives (0,1): 12098
- False Negatives (1,0): 58766
- True Positives (1,1): 11207

The model, as evidenced by these values, has a higher tendency to correctly predict negatives, i.e., a downward movement in Bitcoin prices. The True Negative count is significantly higher compared to True Positives, indicating a potential bias in predictions towards the negative class.

Conclusion:

In conclusion, the experiment's outcomes demonstrate varying performances based on different hyperparameter combinations in the XGBoost model. The parameter tuning stage, executed with Hyperopt, provided a broad array of results, offering insights into the optimal settings for the model. The specific parameter set that gave the most favorable outcomes can be further exploited for more accurate and efficient future predictions.

However, the confusion matrix results highlight an area for improvement in the model’s ability to correctly predict positive classes, i.e., an upward movement in Bitcoin prices. Despite a considerable count of True Negatives, the model fails to achieve a balanced performance in predicting True Positives, leading to a larger number of False Negatives.

This imbalance signals the necessity for further refinement in the model, possibly by exploring additional feature engineering, employing different algorithms, or fine-tuning the model architecture and parameters. Furthermore, the incorporation of other evaluation metrics, such as precision, recall, and the F1-score, could provide a more comprehensive understanding of the model's performance and areas for enhancement.

The endeavor to enhance the model’s performance should be continuous, employing the insights gained from these results to ensure more balanced and accurate predictions for Bitcoin price movements in future analyses.