Iran University of Science and Technology
Computer Engineering Faculty

Natural Language Processing

Instructor
PhD. Sauleh Etemadi

Student
Sahand Nazarzadeh

Final Project
First Phase Report

Spring 2023

Dataset Description

The textual data used in this dataset was collected from articles related to Bitcoin from the website 'CryptoNews'. The numerical data, on the other hand, was obtained from the Binance Futures API using the BTC-USDT symbol and the daily time frame. The dataset is also available on Hugging Face and GitHub.

An Overview of Numerical and Textual Data Gathering Techniques

To begin with, the numerical portion of the dataset was collected using the REST API endpoint of Binance, a popular cryptocurrency exchange. In order to do this, the requests Python package was used to send an HTTP GET request to the Binance server. The response from the server, which contained candlestick data, was then received and stored in CSV files for further analysis.

After collecting the numerical data using the REST API endpoint of Binance, the next step was to gather the textual part of the dataset. This proved to be the most challenging part of the process, as it required interacting with the Bitcoin articles homepage to collect article titles and URLs.

To accomplish this, the author utilized the Selenium Python package, which allowed them to interact with the website and scrape the necessary data. Once the article titles and URLs were collected, the author used the requests package to send GET requests to the collected URLs in order to retrieve the content of the web pages.
The final step in the process involved using Beautiful Soup, a Python library for web scraping, to parse the paragraphs within the web pages and extract text. This text was then added to the dataset alongside the numerical data collected earlier.

Overall, the process of collecting the textual data required a combination of tools and techniques, including web scraping with Selenium, sending GET requests with requests, and parsing HTML with Beautiful Soup. However, by gathering both numerical and textual data, the resulting dataset can provide a more comprehensive analysis of Bitcoin and its related trends.

Data Files Structure

The numerical data can be found in the following location: 'data/clean/numerical/binance/[symbol]/[time frame].csv'. The data is stored in CSV format and contains the following headers: 'timestamp', 'open', 'high', 'low', 'close', 'volume', and 'trade'.

The textual data for this dataset is available in both raw and clean versions. The raw version can be found in the './data/raw/cryptonews/' directory and contains two files: 'urls.csv' and 'news.json'. The 'urls.csv' file contains the URLs of the web pages scraped during the Selenium data collection process. Each row in this file corresponds to a single web page. The 'news.json' file contains the important parts of each web page, such as the datetime, URL, title, and paragraphs. This file is generated during the web scraping process and is used to create the clean version of the textual data. The clean version of the textual data can be found in the 'data/clean/textual/cryptonews.json' file, which is a cleaned-up version of the 'news.json' file. As of now, only the data from cryptonews.com is available in this directory.

The combined dataset, which includes both the numerical and textual data with respect to their respective dates, can be found in the './data/dataset/[textual source]/[labeling method]/' directory. The dataset contains both the news contents and BTC price momentum labels. The files are stored in CSV format and contain the following headers: 'datetime', 'text', 'url', and 'label.

Preprocessing

1. Due to the uncommon nature of this type of data, the decision was made to not break paragraphs into individual sentences.
2. Again, due to the uncommon nature of this type of news, the decision was made to split paragraphs using only a space as the delimiter to collect the words.
3. To clean the data, punctuations, unicode characters, and stop-words were removed from the raw text.
   Punctuation marks and unicode characters are often irrelevant or unnecessary for language analysis and modeling. Therefore, they are typically removed from text during the cleaning process.
   Stop-words, on the other hand, are commonly used words that are often removed from text to reduce noise and improve the quality of analysis. Examples of stop-words include 'the', 'and', 'or', and 'is'.

Labeling Methods

In this study, two types of labeling methods were used: ROC and Color methods. Both methods are based on the future value of BTC as of the publishing date of the news. The future value of BTC is assigned using a look-ahead factor, which determines how many days forward should be considered when adding labels to each news article based on its publishing date.

In the Color Method, if the future value of BTC is greater than the value of BTC at the article's publishing date, the label is assigned as green or 1. On the other hand, if the future value is less than the value of BTC at the article's publishing date, the label is assigned as red or 0.

In the ROC Method, the same process is followed, but instead of a comparison, the future value of BTC is divided by the current value of BTC at the time of the article's publishing and then subtracted by one. This method is commonly used in finance to measure the rate of change in a stock or asset's value over time.

Code Execution

To execute this process, first set up a Python environment and install the necessary packages listed in the 'requirements.txt' file.
To collect the data, run the following command in Python: 'python main.py --crawl'.
To clean the data, use the following command in Python: 'python main.py --clean'. This will remove any unwanted characters, noise, or irrelevant information from the data.
To add labels to the data for analysis and modeling, use the following command in Python: 'python main.py --add-label --labeling-method [roc / color]'. The 'add-label' command adds labels to the data based on either the ROC or Color method, depending on the desired approach.
To create statistics based on the cleaned and labeled data, use the following command in Python: 'python main.py --analyse'

Statistics

A. Samples count 180345
B. Sentence count 180345
C. Word count 3776263
D. Unique word count 11884
E. Unique word count of each label

label	unique words count
green	8305
red	8890
overlapping	5311

F.

1) Unique overlapping frequently words

word	count
bitcoin	88477
crypto	43463
market	42969
price	40014
btc	27249
trading	23033
last	21764
us	20566
level	19808
also	17076

2) Unique non-overlapping frequently green words

word	count
dad	668
kiyosaki	665
thiel	530
ohanian	470
landscape	447
localbitcoins	440
depth	407
trustee	392
silver	333
nts	306

3) Unique non-overlapping frequently red words

word	count
ccharge	780
hive	695
andresen	604
blockfi	589
seventh	581
meta	480
ev	468
evs	468
cuban	459
transitioning	453

G. 10 most overlapping RNF word

word	RNF
hayes	259.64100628173765
blockstreams	185.1595379918022
pumping	159.40426391023573
rplr	132.95290133997827
analyses	121.8154855209225
wrong	119.03113156615856
faith	118.33504307746756
hiking	113.81046790097615
bigger	107.1976272584118
arthur	105.10936179233885

H.

1) 10 most TF-IDF green words

word	TF-IDF
dad	0.0002987608902865085
kiyosaki	0.0002974191497612697
thiel	0.0002370408261255232
ohanian	0.00021020601562074702
landscape	0.00019991933826058277
localbitcoins	0.0001967886103683588 7
depth	0.000182029464590732
trustee	0.0001753207619645379
silver	0.00014893319830150796
nts	0.00013685753357435868

2) 10 most TF-IDF red words

word	TF-IDF
ccharge	0.0002428322349515226
hive	0.00021636974780936952
andresen	0.00018803932039835854
blockfi	0.00018336946972621388
seventh	0.00018087888270107005
meta	0.0001494352215086293
evs	0.00014569934097091356
ev	0.00014569934097091356
cuban	0.00014289743056762676
transitioning	0.0001410294902987689

J. Words Histogram