# Crypto market analysis prediction in distributed systems

Asahi Cantu Moreno
University of Stavanger, Norway
a.cantumoreno@stud.uis.no

Daniel Urdal
University of Stavanger, Norway
d.urdal@stud.uis.no

## ABSTRACT

The creation of cryptocurrency and its massive adoption throughout the last three years has led to the emission of many different exchange and tools to trade it via Internet. Unlike common stock trading, cryptocurrency works nonstop all the year with simple and advanced interfaces, allowing any user with a minimum amount of money to enter the market and change the currency on real time due to its high liquidity and asset availability. Because of the early state of the technology and the socioeconomical-political factors affecting the regulation of the cryptocurrency markets as well as the speed at which transactions are made has led to a high risk-high volatility market, making very hard to analyze or predict prices that can lead to profits for a chosen currency. This academic report contains a deep analysis on crypto market and aims to generate a prediction model by analyzing historical data, the news and events that can potentially affect the market and its impact on the prices by using distributed systems Hadoop® and Apache Spark® for data storage and interaction. Machine Learning models are implemented as well for a big dataset. Further analysis, results and potential for future work is presented in this report.

## KEYWORDS

Cryptocurrency, Machine Learning, Hadoop, Spark, Neural Networks, Distributed Systems

## 1. INTRODUCTION

It was not long ago that bitcoin and the concept of Blockchain appeared for the first time on the Web as a white paper, October 31st 2008 to be more precise "Bitcoin: A Peer-to-Peer Electronic Cash System" [1] Said the document in its heading. Little could the world know by that time how disruptive and massively adopted such technology could be, and the importance of its role nowadays. A decade later, Bitcoin has consolidated as a strong and reliable asset in which people is trusting to invest and transfer money. The term "Cryptocurrency" was born. Little after its emission and open source code publication, several teams and individuals started readapting its code and creating different electronic assets, slowly with its adoption a philosophy more people started trading on their private markets. It would not take too long to foresee the creation of a community and a market craving to trade different assets. As a result of this crescent demand Cryptocurrency market exchanges were created. From the financial perspective, cyptocurrency markets are nowadays allowing people who did not have access to credits or bank accounts before to enter into a different economic world were markets do not sleep, moreover assets can be instantly traded, changed and transferred. Years ago, this market consolidated and matured, data is produced massively, regulations or innovations, even political situations have made the market very peculiar and volatile, something unprecedented in the financial trading.

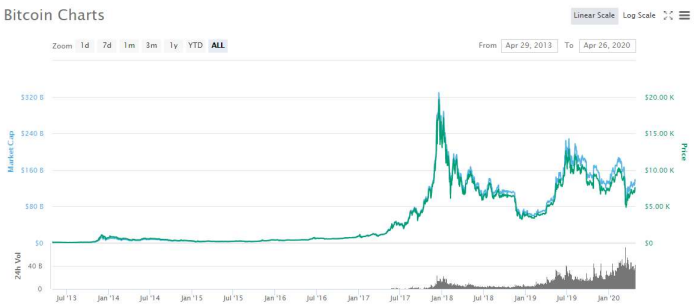



**Figure 1. Bitcoin price chart (from coinmarketcap.com [2])**

Bitcoin price volatility generated a high-risk market, but also a lot of interest and market capitalization has been set to the cryptocurrency assets. On April 2020, the total market capitalization of all cryptocurrencies is up to $219,165,492,833.00 USD [3] which translates in a growing market accessible to anyone with sufficient resources around the world to trade. As the time goes by more resources and assets will be added resulting in an expected increase of profit depending on the investment, but daily fluctuations are visible as well for any asset.

In addition to this, the impact of social media and published news about recent technological progress and Information Technology improvements are expected to be a key player in the decision different traders made to trade cryptocurrency assets. There might be a direct correlation in what is published on social media, news and the price fluctuation of the market in short term basis. It is therefore the aim of this report to explore, discover and predict potential market prices through distributed systems to show the potential of creating a scalable product trained to predict short term prices according to past fluctuations and media publications.

The system used, algorithms, data set and brief explanation of the challenges in addition to obtain results will be explained in detail though this report. Provided results can be also used as a basis for future work development and algorithm improvements.

## 2. BACKGROUND AND MOTIVATION

The growing importance of cryptocurrency in a global market where facing a potential economic crisis in which fiat money might not be anymore the common exchange medium between different parties. The technological development and amazing availability of growing data to explore are the key factors that led to research on cryptocurrency and try to analyze through machine learning and distributed systems the potential price fluctuations.
Another point to consider: Social media plays nowadays an important role in our daily life, more and more politicians use these platforms for instant communications, researchers announce publications and press uses them as a main platform to publish breaking news. All in all, we cannot take out any more social media from the impact of our economic and political situations. It is therefore important to consider this as a key factor in the market decisions made on daily basis and the price fluctuations of the assets.

The main idea is to use a big cryptocurrency dataset and a trained model of cryptocurrency related news to predict short term future prices for all the provided asset by using Machine learning algorithms.

### 2.1 Cryptocurrency

Wikipedia [4] defines cryptocurrency as "a digital asset designed to work as a medium of exchange that uses strong cryptography to secure financial transactions, control the creation of additional units, and verify the transfer of assets." Cryptocurrencies use decentralized control as opposed to centralized digital currency and central banking systems, which means no central entity has full control or power over the asset, but the community maintains it through systems known as "nodes" and each transaction is recorded in a public ledger known as the "blockchain" public to anyone and resilient to changes. It is therefore virtually impossible to commit fraud if more than 50% of the computational resources belong to the community and not the entity intending to hack or modify the blockchain.

#### 2.1.1 Cryptocurrency dataset

The dataset used for this project is a Binance exchange dataset extracted from the portal Kaggle.com [5]. The dataset is titled "Binance Full History" and contains 1 minute candlestick data for a total of 786 cryptocurrency pairs traded on the exchange Web portal Binance.com [6][1].
The features of this dataset are:

- Data types: python-pandas parquet files [2]
- Data size: 10GB
- Number of files: 764

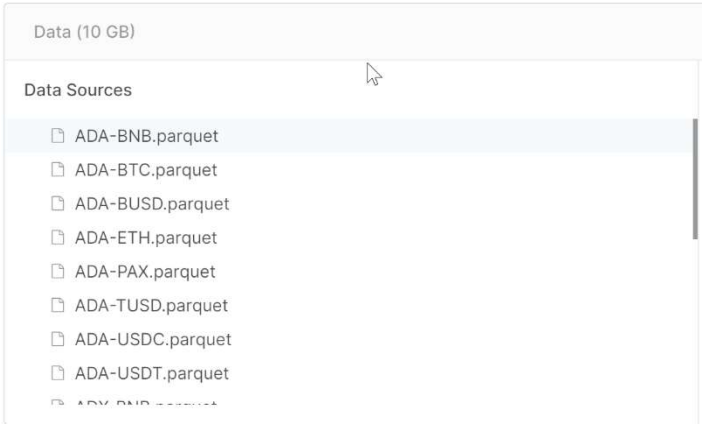



**Figure 2. Cryptocurrency data set. Parquet files [5]**

#### 2.1.2 Cryptocurrency dataset features and details

A minute candlestick represents the maximum and minimum price at which any given asset was sold. It also contains information about the open price (price at the beginning of the time) and close price (end of time).

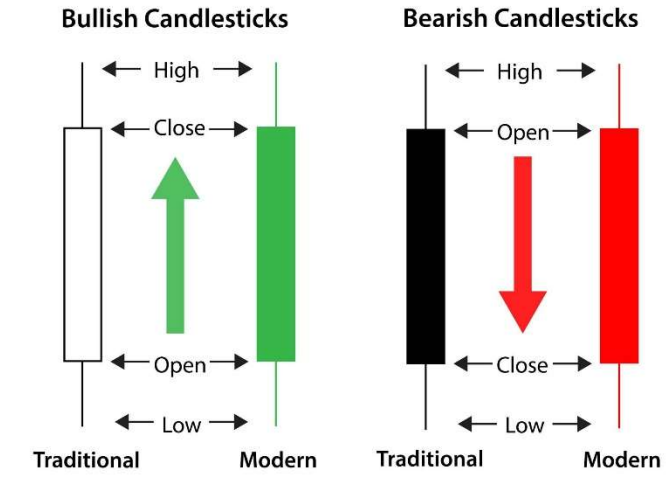



**Figure 3. Candlestick representation**

A series of candlestick represent the market fluctuation in each period.

### 2.2 News dataset

The news dataset was extracted from the web portal "CoinTelegraph.com"[3]. The dataset was created using python scripts due to the lack of availability of a full set.
The features of this dataset are:

[1] Binance is cryptocurrency exchange that provides a platform for trading various cryptocurrencies. As of January 2018, Binance was the largest cryptocurrency [14] exchange in the world in terms of trading volume.

[2] pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python [15]. Parquet files are compressed binary files specially built to reduce storage size of big data files and improve loading performance.

[3] The most recent news about crypto industry at Cointelegraph. Latest news about bitcoin, Ethereum, blockchain, mining, cryptocurrency prices and more [16].

- Data types: 2 .csv files total size is 128 MB.
  - More than 20,000 different news from 2013 until present for human-written news about cryptocurrency and related subjects.
  - Header file. Contains relevant metadata information about the news (such as title, thumbnail, creation date, etc.).
  - Content file. Contains the body of the news and other metadata like the number of views and times it has been shared as well as image references for it.

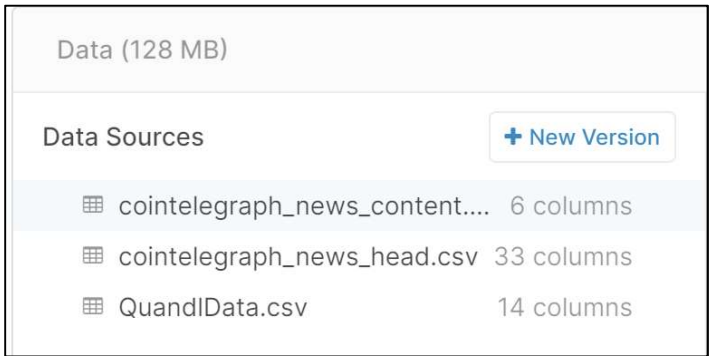



**Figure 4. Cryptocurrency news set [7]**

## 3. SETUP

The system used for this for this project consisted of four virtual machines (VMs) running Ubuntu Server 16.04. Apache Hadoop version 3.2.1 was installed on each machine, and they were configured to run as a Hadoop cluster using the YARN resource manager. They were configured such that one VM acted as a master node, while the other three acted as slave nodes.
Apache Spark was installed using the 3.0.0 preview version, as this was the only version supporting Hadoop 3.2.1. In order to get Spark running on the VMs, Python version 3.8.1 was compiled from source and installed alongside the default Python 2.7.12.

## 4. PREPROCESSING

The intention is to combine both datasets, transform them whereas necessary to reduce size, unnecessary features and use them with ML training algorithms that enable an accurate cryptocurrency market price prediction.

### 4.1 Cryptocurrency dataset preprocessing

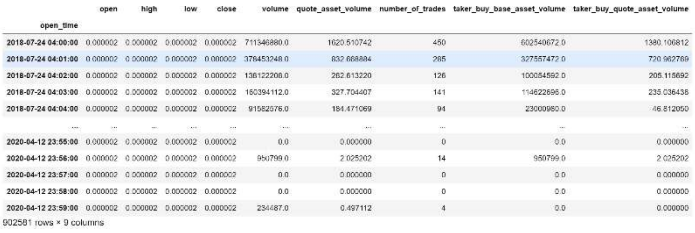
Because this dataset was machine-generated little had to be done to process it. It was important, however, to examine the contained information and extract just the necessary one in real time. Because of data size, and the technical resources limitations it was not possible to extract and store the required information, but rather open the file, select the main features and use them in real time for machine learning algorithm to train and generate a model, and the close it. Uncompressed data reached 70 GB of space, whereas compressed dataset just 9 GB.

#### 4.1.1 Dataset description and feature selection

Each parquet file in the dataset contains a corresponding pandas Dataframe as shown in Figure 5.

| COLUMN | DESCRIPTION | SELECTED? |
|---|---|---|
| open_time | Date with minute scale | X |
| open | Price at the beginning | X |
| high | Highest price in a minute | X |
| low | Lowest price in a minute | X |
| close | Price at the end | X |
| volume | Total asset traded | |
| quote_asset_volume | Total currency exchanged | |
| number_of_trades | Total TRADES | |
| taker_buy_base_asset_volume | Total amount of traded assets | |
| taker_buy_quote_asset_volume | Total currency exchanged | |

**Tableau 1. Cryptocurrency features**



| open_time | open | high | low | close | volume | quote_asset_volume | number_of_trades | taker_buy_base_asset_volume | taker_buy_quote_asset_volume |
|---|---|---|---|---|---|---|---|---|---|
| 2018-07-24 04:00:00 | 0.000002 | 0.000002 | 0.000002 | 0.000002 | 711546880.0 | 1620.510742 | 450 | 602540672.0 | 1380.106812 |
| 2018-07-24 04:01:00 | 0.000002 | 0.000002 | 0.000002 | 0.000002 | 376453248.0 | 832.668884 | 285 | 327557472.0 | 720.962769 |
| 2018-07-24 04:02:00 | 0.000002 | 0.000002 | 0.000002 | 0.000002 | 136122208.0 | 262.613220 | 126 | 100054592.0 | 205.115692 |
| 2018-07-24 04:03:00 | 0.000002 | 0.000002 | 0.000002 | 0.000002 | 160394112.0 | 327.704407 | 141 | 114622696.0 | 235.036438 |
| 2018-07-24 04:04:00 | 0.000002 | 0.000002 | 0.000002 | 0.000002 | 91582976.0 | 184.471069 | 94 | 23009960.0 | 46.812050 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 2020-04-12 23:55:00 | 0.000002 | 0.000002 | 0.000002 | 0.000002 | 0.0 | 0.000000 | 0 | 0.0 | 0.000000 |
| 2020-04-12 23:56:00 | 0.000002 | 0.000002 | 0.000002 | 0.000002 | 950799.0 | 2.025202 | 14 | 950799.0 | 2.025202 |
| 2020-04-12 23:57:00 | 0.000002 | 0.000002 | 0.000002 | 0.000002 | 0.0 | 0.000000 | 0 | 0.0 | 0.000000 |
| 2020-04-12 23:58:00 | 0.000002 | 0.000002 | 0.000002 | 0.000002 | 0.0 | 0.000000 | 0 | 0.0 | 0.000000 |
| 2020-04-12 23:59:00 | 0.000002 | 0.000002 | 0.000002 | 0.000002 | 234487.0 | 0.497112 | 4 | 0.0 | 0.000000 |

902581 rows × 9 columns

**Figure 5. Example of parquet file HOT-ETH parquet**

From Tableau 1 features marked with 'X' where selected to be processed and to train the Machine Learning model.

### 4.2 Cryptocurrency news data preprocessing

This data set was non-existent, it had to be generated by creating a python script to bring the information from the web portal "coinmarketcap.com". The script uses web scraping methodology and extracts the news set according to the given parameters.
A total of 24519 had to be gathered and processed to generate the dataset in two steps.
The first step consists on getting the header information from the web portal, whereas the second brings the content of a specific news article.
The script was generated and run in a Hadoop cluster to better control and preserve information in case of data loss.
A part of the algorithm is shown, full code description and implementation can be seen in [8].

```
try:
    winsound.Beep(500, 100)
    pbar.set_description(f"Fetching data for page {page} ({total_items} items)")
    r = fetch_data(page,API_KEY)
    data = json.loads(r.text)
    if not 'posts' in data.keys():
        break
    total_items += len(data['posts'])
    pbar.set_description(f"Fetching data for page {page} ({total_items} items)")
    for row in data['posts']:
        for key in list(row.keys()):
            val = row[key]
            if type(val) == dict:
                for s_key in val.keys():
                    s_val = val[s_key]
                    s_key = key + '_' + s_key
                    row.update({s_key:s_val})
                row.pop(key)
        if df is None:
            df = pd.DataFrame(row,index = [0])
        else :
            df = df.append(row,ignore_index = True)
except:
    info = sys.exc_info()[0]
    print(info[0],info[1],info[2])
    winsound.Beep(1500, 100)
finally:
    df.to_csv(f'{file_name}{start_index}-{total_pages}.csv')
```

**Figure 6. Code implementation for news heading extraction. Data is extracted into a dictionary and later saved on a pandas Dataframes.**

```
file_name = 'cointelegraph_news1-1000'
df = pd.read_csv (f"{file_name}.csv",index_col=[0])
start_index = get_arg(1,0,'int')
end_index =  get_arg(2, len(df),'int')
step = get_arg(3,10,'int')
file_mode = get_arg(4, 'w','str')
print('Getting data:')
cols = ['id','header','date','total_views','total_shares','content']
pbar = tqdm.tqdm(range(start_index,end_index,step))
f_name = f'{file_name}_content.csv'
if file_mode == 'w':
    content_df = pd.DataFrame(columns=cols)
    content_df.to_csv(f_name, mode=file_mode, header=True,index=False)
    file_mode = 'a'
for row_idx in pbar:
        winsound.Beep(500, 100)
        idxs =list( range(row_idx,row_idx + step))
        vals = [(df.loc[x]['id'],df.loc[x]['url']) for x in idxs]
        results = gather_results([run_item(parse,val) for val in vals])
        #     #content_df.loc[len(content_df)] = result
        content_df = pd.DataFrame(results, columns=cols)
        content_df.to_csv(f_name, mode=file_mode, header=False,index=False)
        #     pbar.set_description(f"Fetching data for item {row_idx}-{id}")
```

**Figure 7. Code implementation for news article extraction, data is directly saved in a pandas dataframe and updated in a .csv file. Http request interactions are intensive and took close to 3 days of continuous execution to retrieve the full dataset.**

Once created the dataset, it was properly cleaned and processed to be used for natural language processing just one file (cointelegraph_news_content.csv) was used to train the model for being the one with the article content.

| | id | header | date | total_views | total_shares | content |
|---|---|---|---|---|---|---|
| 0 | 42577 | Cryptocurrency News From Japan: March 29 - Ap... | 2020-04-05t00:00:00+01:00 | 1497 | 139 | This week's headlines from Japan included the ... |
| 1 | 42575 | Retail Bought $3.7K Bitcoin Price Dip on Reco... | 2020-04-04t22:50:00+01:00 | 6807 | 102 | Although the Coronavirus pandemic has led many... |
| 2 | 42574 | Bitcoin's Hedging Performance in the Wake of ... | 2020-04-04t22:41:00+01:00 | 2626 | 136 | The recent coronavirus outbreak has far-reachi... |
| 3 | 42573 | SEC Alleges Minority Communities Targeted By ... | 2020-04-04t21:58:00+01:00 | 5375 | 96 | The United States Securities and Exchange Comm... |
| 4 | 42572 | April Fools, Celebrity Scams, & Manipulated M... | 2020-04-04t20:53:00+01:00 | 1801 | 83 | Bitcoin seems to be settling happily above $6,... |
| 5 | 42571 | Staking, Consensus and the Pursuit of Decentr... | 2020-04-04t19:59:00+01:00 | 1078 | 52 | Oh, the wonders of decentralized consensus — t... |
| 6 | 42570 | Crypto Community Largely Approves of Binance ... | 2020-04-04t18:30:00+01:00 | 2026 | 138 | Despite markets all over the world facing ever... |
| 7 | 42569 | Billionaire Optimistic On Bitcoin as a 'Fligh... | 2020-04-04t18:00:00+01:00 | 6211 | 197 | In an interview with Morgan Creek Digital foun... |
| 8 | 42568 | Blockchain Experts Weigh in on Russia's Contr... | 2020-04-04t19:30:00+01:00 | 3394 | 118 | Experts suggest that blockchain technology cou... |
| 9 | 42566 | Voyager Onboards 40,000 Circle Invest Customers | 2020-04-04t16:59:00+01:00 | 2091 | 65 | Crypto trade service firm, Voyager Digital Can... |
| 10 | 42562 | Talking Digital Future: Smart Cities | 2020-04-04t16:01:00+01:00 | 1130 | 72 | My journey into smart cities and their future ... |
| 11 | 42561 | IBM Praises CEO For Playing a Significant Rol... | 2020-04-04t15:39:00+01:00 | 4822 | 114 | In a letter to the shareholders, IBM has ackno... |
| 12 | 42557 | Bitcoin Price Struggling to Break $7K — Here'... | 2020-04-04t15:00:00+01:00 | 11413 | 80 | The price of Bitcoin (BTC) has seen a relative... |

**Figure 8. Cryptocurrency news content example**

## 5. ALGORITHM IMPLEMENTATIONS

Unlike other Machine Learning algorithm applications to systems and problem, stock trading belongs to a branch of algorithm implementation: sequence prediction problems, which tend to prediction of current and next values depends on the history and past records, events do not occur independently. For these situations RNN[4] algorithms are applied to the cryptocurrency candlestick dataset [9].

On the other hand, for the news feed dataset it was decided to implement NLP algorithms and clustering for unclassified text, since the dataset is not labeled. The intention of such classification was to better understand the topics for the news, to isolate, classify them and try to find a pattern to categorize the news. Subsequently a sentiment analysis algorithm can be performed to generate a label that determines whether a specific article is positive or negative, and how impactful it is for the trading in during the next hours or days. The assumption is that by this results forecasts can be more accurate as they can take articles in real time and merge them with the candlestick data to adapt the training model into a more realistic prediction.

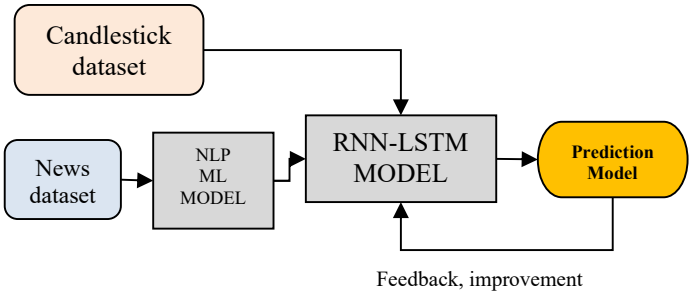



**Figure 9. ML Training model prototype**

[4] Recurrent Neural Networks, is a class of artificial neural networks where connections between nodes form a directed graph along a temporal sequence [17]

### 5.1 Cryptocurrency analysis algorithm

Memory-based networks such as Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM) networks allow for predictions to be remembered and forgotten; this can be useful when predicting extreme changes in a short space of time via the remembering of previous examples. Compared to other models, RNN and LSTM are suited towards sequential data such as stock prices, whereas other models must take a non-sequential, sample-independent interpretation of the data, considering only a single input and output sample at a time [10]

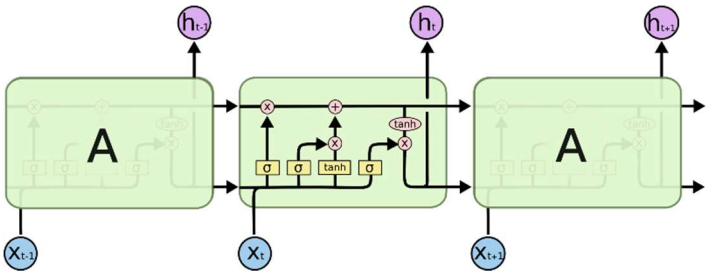



**Figure 10. Example of an RNN-LSTM model**

To implement RNN-LSTM approach the python TensorFlow[5] deep learning model package was used and adjusted for the best-case scenario in which the loss function and consequent prediction were as accurate as possible without much overfitting. Several examples were observed, the model was retrained for all the cryptocurrency assets using spark functionality with Hadoop ass a file system [11].

### 5.2 Text and technical sentiment analysis algorithm

News data set was intended to be classified and processed by using Natural Language processing tools and algorithms to find patterns in the whole dataset. Normal processes for NLP processing require (among others):

- String tokenization
- Stop words removal
- Lexicon normalization
  - Lemmatization method was applied
  - Word2Vet approach implemented as well

Several NLP and sentiment analysis algorithms were used to classify the model and try obtaining the news topics:

- K-Means Clustering
  - Treating with an unlabeled dataset for cryptocurrency news required a machine learning model to try to process and generate corresponding clusters in order to identify the main topics
- Sentiment analysis
  - Common sentiment analysis was performed over the full dataset. This approach aimed to find and tag an approximate positive, neutral or negative sentiment to each document.
- Pre-Trained data models
  - The existence of complex machine learning models based on big scientific documents were used to try to identify and label each article
- Text summarization
  - Extractive summarization approach was performed to try to identify the main topics

## 6. CODE IMPLEMENTATIONS

Most of the algorithms were implemented thinking on a distributed system approach in which data intensive processes and redundancy systems could be able to handle the provided information. The configuration employed is listed below:

- Hadoop Cluster (Ubuntu System)
  - 1 Name Node (Master)
  - 3 Data Nodes (Slaves)
- Apache Spark as the processing interface with PySpark
  - Spark Dataframes
  - RDD's and distributed processes
- Jupyter notebook for python scripting
- Tensorflow as the machine learning package
  - Intended to use SparkML
- NLP Packages and tools:
  - NLTK
  - Spacy
  - VaderSentiment
  - Textblob
  - Textacy
  - Ge Nasim

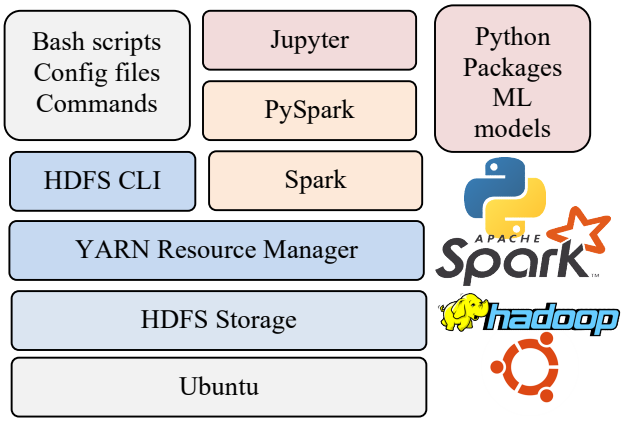



**Figure 11. Distributed system architecture**

[5] TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library and is also used for machine learning applications such as neural networks. Package can be downloaded from https://www.tensorflow.org

### 6.2 Cryptocurrency dataset code implementation

The code for the cryptocurrency candlestick dataset was divided in three different python files

- Settings file [crypto_params.py]
  - Contains a predefined set of variables, resulted from the training and tuning of the RNN-LSTM model. The most optimistic approach ad loss function implemented can be visualized here
- ML algorithm [crypto_utils.py]
  - Contains a code snippet for the ML training model and the configuration of its layers. The way it is called and used by TensorFlow
- Dataset reading and training in Spark [sprk.ipynb]
  - Is a Jupyter notebook that contains the main functionality and coordination to call the settings and utils files, respectively. In addition, it sets the environment to open a spark environment and connect to the Hadoop YARN resource manager, which allows the system to read the parquet files in a distributed manner.
  - Once the files are called, the ML model is called to be trained based on the obtained dataset and the result is saved in a spark json format. This allows spark to save back the trained model in the distributed system, eventually as the data keeps growing the model will be fed in an iterative process to readjust the prediction

```
from tensorflow.keras.layers import LSTM
############################################################
###  DEFINE TRAINING DATA FRAME PARAMETERS
############################################################
# Number of iterations
N_STEPS = 100
# Window of 'n' next days to load as values to predict
WINDOW_OFFSET = 90
# test ratio size, 0.2 is 20%
TEST_SIZE = 0.2
# Select default feature columns to train the neural network
FEATURE_COLS = ['high','low','open','close', 'volume']
############################################################
###  DEFINE DEFAULT NEURAL NETWORK PARAMETERS
############################################################
N_LAYERS = 3
# RNN=LSTM BEST SUITABLE FOR TIME SERIES PREDICTION
CELL = LSTM
# NUMBER OF "NEURONS TO CREATE"
N_UNITS = 256
DROPOUT = 0.4
############################################################
###  DEFINE THE TRAINING PARAMETERS
############################################################
# Loss type function, default is mean square error
LOSS = "mse"
OPTIMIZER = "rmsprop"
BATCH_SIZE = 64
EPOCHS = 300
RESULTS_PATH = "results"
LOG_PATH = "logs"
DATA_PATH = "data"
```

**Figure 12. Fragment from file crypto_params.py**

```
def train_models(epochs, pfiles):
    models = []
    for parquet_file in tqdm.tqdm(pfiles):
        currency = os.path.splitext(parquet_file)[0]
        df = pq.read_table(\
            f"{hdp_master}/Data/Binance_Parquet/{parquet_file}")
        df = df.to_pandas()
        data = cu.load_data(currency,df,'hour',\
             N_STEPS, window_offset=WINDOW_OFFSET, \
             test_size=TEST_SIZE, feature_cols=FEATURE_COLS,log =False)

        model = cu.create_model(N_STEPS, loss=LOSS, \
            units=N_UNITS, cell=CELL, \
                n_layers=N_LAYERS,dropout=DROPOUT, optimizer=OPTIMIZER)
        checkpoint, tensorboard = create_tensorboard(currency)
        trained_model = model.fit(data["X_train"], data["y_train"],
                            batch_size=BATCH_SIZE,
                            epochs=epochs,
                            validation_data=(data["X_test"], data["y_test"]),
                            callbacks=[checkpoint, tensorboard],
                            verbose=3)

        json_model = trained_model.model.to_json()
        cu.save_model(trained_model.model,data, RESULTS_PATH,currency)
        df_model = spark.read.json(session.parallelize([json_model]))

        model_dict =
        {
            'currency':currency,
            'data':data,
            'model':trained_model.model
        }
        models.append(model_dict)
    return models
```

**Figure 13. Fragment from file crypto_utils.py**

```
def create_model(input_length, units=256,\
    cell=tf.keras.layers.LSTM,\
    n_layers=2, dropout=0.3,\
    loss="mean_absolute_error", optimizer="rmsprop"):
    model = tf.keras.Sequential()
    for i in range(n_layers):
        if i == 0: # Create input layer
            i_s = (None, input_length)
            c = cell(units, return_sequences=True, input_shape=i_s)
            model.add(c)
        elif i == n_layers - 1: # Create output layer
            model.add(cell(units, return_sequences=False))
        else: # Create hidden layers
            model.add(cell(units, return_sequences=True))
        model.add(tf.keras.layers.Dropout(dropout))
    model.add(tf.keras.layers.Dense(1, activation="linear"))
    model.compile(loss=loss, metrics=[loss], optimizer=optimizer)
    return model
```

**Figure 14. Fragment from file spr.ipynb**

### 6.3 Newsfeed dataset code implementation

The newsfeed dataset code required a special treatment. It consisted of two stages:

#### 6.3.1 Stage 1. Newsfeed dataset extraction

Since the whole data was not available on the web, a python script had to be developed in order to store such the information from the known sources. The implementation of such script is very important due to the lack of the information on the web. The script sets the basis for anyone else to use it and implement the code for any other newsfeed to assemble different data repositories. Both dataset and code implementation have been published and made open for anyone to use it and improve it if required. The algorithm was developed in two scripts:

- Newsfeed headings extraction
- Newsfeed articles extraction

```
01: try:
02:         winsound.Beep(500, 100)
03:         msg = f"Fetching data for page {page} \
04:              ({total_items} items)"
05:         pbar.set_description(msg)
06:         r = fetch_data(page,API_KEY)
07:         data = json.loads(r.text)
08:         if not 'posts' in data.keys():
09:             break
10:         total_items += len(data['posts'])
11:         msg = f"Fetching data for page {page} \
12:              ({total_items} items)"
13:         pbar.set_description(msg)
14:         for row in data['posts']:
15:             for key in list(row.keys()):
16:                 val = row[key]
17:                 if type(val) == dict:
18:                     for s_key in val.keys():
19:                         s_val = val[s_key]
20:                         s_key = key + '_' + s_key
21:                         row.update({s_key:s_val})
22:                     row.pop(key)
23:             if df is None:
24:                 df = pd.DataFrame(row,index = [0])
25:             else :
26:                 df = df.append(row,ignore_index = True)
27:     except:
28:         info = sys.exc_info()[0]
29:         print(info[0],info[1],info[2])
30:         winsound.Beep(1500, 100)
31:     finally:
32:         f_name=f'{file_name}{start_index}-
{total_pages}.csv'
33:         df.to_csv(f_name)
```

Figure 15. Code snipped for news headings extraction

```
01: file_name = 'cointelegraph_news1-1000'
02: df = pd.read_csv (f"{file_name}.csv",index_col=[0])
03: start_index = get_arg(1,0,'int')
04: end_index =  get_arg(2, len(df),'int')
05: step = get_arg(3,10,'int')
06: file_mode = get_arg(4, 'w','str')
07: print('Getting data:')
08: cols = [
09:     'id','header','date',
10:     'total_views',
11:     'total_shares','content']
12: pbar = tqdm.tqdm(range(start_index,end_index,step))
13: f_name = f'{file_name}_content.csv'
14: if file_mode == 'w':
15:     content_df = pd.DataFrame(columns=cols)
16:     content_df.to_csv(f_name, mode=file_mode,\
17:          header=True,index=False)
18:     file_mode = 'a'
19: for row_idx in pbar:
20:         winsound.Beep(500, 100)
21:         idxs =list( range(row_idx,row_idx + step))
22:         vals = [(df.loc[x]['id'],df.loc[x]['url']) fo
r x in idxs]
23:         results = gather_results([run_item(parse,val)
 for val in vals])
24:         #     #content_df.loc[len(content_df)] = resu
lt
25:         content_df = pd.DataFrame(results, columns=co
ls)
26:         content_df.to_csv(f_name, mode=file_mode, hea
der=False,index=False)
27:         #     pbar.set_description(f"Fetching data fo
r item {row_idx}-{id}")
28: winsound.Beep(2000, 500)
```

Figure 16. Code snipped for articles extraction

#### 6.3.2 Stage 2. Newsfeed exploration and classification

Several attempts to classify the dataset and train it to obtain relevant information that lead to decide whether an article is positive, or negative were performed. As stated in **section 5.2,** different approaches and analysis were tried to obtain a proper classification. Sentiment analysis, clustering, text summarization and deep learning algorithms were applied to the dataset, the classification results were neutral and did not provide insightful information for the whole dataset to be used and applied as an extra feature for the candlestick dataset in the RNN-LTSM algorithm.

The code snippets shown below will demonstrate the use of such algorithms and the way to try to obtain useful information to tag the dataset. It was observed, however, that such attempts were not enough, and further text analysis was required. This is a limitation will be addressed in **section 8.**

```
01: def getSentimentAnalysis(analyzer,text):
02:     tb_text_sentiment = tb.TextBlob(text).sentiment
03:     vd_text_sentiment = analyser.polarity_scores(te
xt)
04:     vd_text_sentiment['polarity'] = tb_text_sentime
nt.polarity
05:     vd_text_sentiment['subjectivity'] = tb_text_sen
timent.subjectivity
06:     return vd_text_sentiment
07: analyser = SentimentIntensityAnalyzer()
08: df = pd.read_csv("\cointelegraph_news_content.csv")
09: df.dropna(inplace=True)
10: df.reset_index(drop=True, inplace=True)
11: nlp = spacy.load('en_core_web_lg')
12: extra_cols =['neg', 'neu', 'pos', 'compound', 'pola
rity', 'subjectivity']
13: for extra_col in  extra_cols:
14:     df[extra_col] = 0.0
15: items = len(df)
16: for i in  tqdm.tqdm(range(items)):
17:     data = getSentimentAnalysis(analyser,df['conten
t'][i])
18:     for key in data.keys():
19:         df[key][i]=data[key]
```

**Figure 17. Code snippet for sentiment analysis attempt**

```
01: from sklearn.feature_extraction.text import
TfidfVectorizer
02: def vectorize(text, maxx_features):
03:     vectorizer = TfidfVectorizer(max_feature
s=maxx_features)
04:     X = vectorizer.fit_transform(text)
05:     return X
06:
07: text = df['processed_content'].values
08: X = vectorize(text, 2 ** 12)
09: X.shape
```

**Figure 18. Code snippet for text vectorization**

```
01: from sklearn.decomposition import PCA
02: pca = PCA(n_components=0.95, random_state=42)
03: X_reduced= pca.fit_transform(X.toarray())
04: X_reduced.shape
05: from sklearn.cluster import KMeans
06: from sklearn import metrics
07: from scipy.spatial.distance import cdist
08: # run kmeans with many different k
09: distortions = []
10: K = range(2, 50)
11: for k in K:
12:     k_means = KMeans(n_clusters=k, \
13:         random_state=42).fit(X_reduced)
14:     k_means.fit(X_reduced)
15:     distortions.append(sum(np.min(cdist(X_reduced,\
16:         k_means.cluster_centers_,\
17:             'euclidean'), axis=1)) / X.shape[0])
```

**Figure 19. Code snippet for KNN clustering classification**

## 7. MODEL TRAINING AND TUNING

Several iterations were required to get proper adjustments to the LSTM model for candlestick data and KNN for the NLP processing. The iterations involved observing the loss function through the deep learning model using TensorBoard and analyzing its precision, overfitting and underfitting on both models.

### 7.1 Cryptocurrency dataset training and tuning

The final adjustments for the training of the neural network can be seen in [12]. The neural network implementation, training and performance for every cryptocurrency trained on the spark environment can be visualized as well in the provided citation. Each candlestick dataset is preprocessed in a pandas Dataframe. The loss function and window offset (period in which the neural network predicts the values based on past data) were carefully chosen to approximate the prediction as much as possible for all the sets. Primarily, it was observed that on average most of the models adapt properly at 300 epochs[6] with "Mean Square Error" as loss function and 256 neurons on 3 layers (one hidden). The model can mutate, and it is let to the reader to try to find a better tuning mechanism. It is important to highlight that for the given model the dataset was aggregated to present hourly data instead of minute. The reason for this was for the expensive of the process and the amount of resources a training model takes for a single currency. In more powerful environments the model can mutate to observe minute predictions.

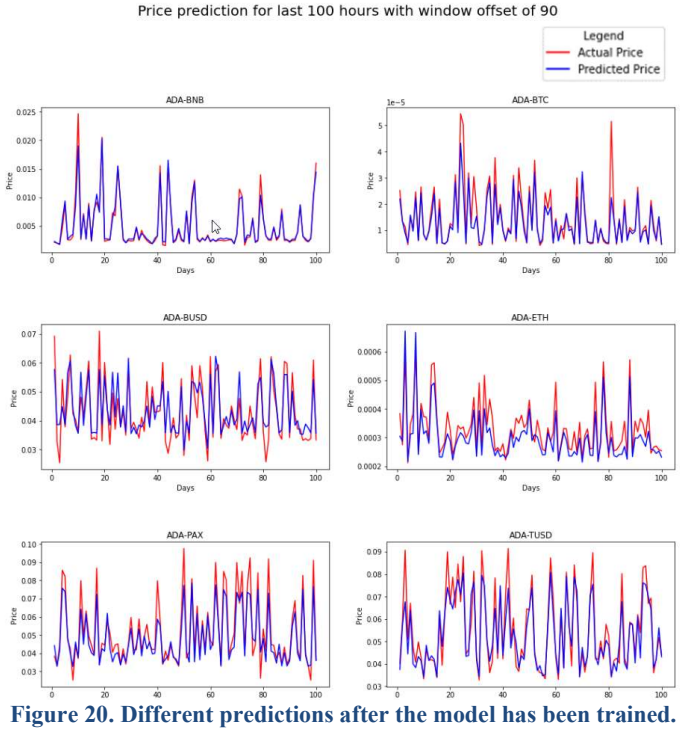



**Figure 20. Different predictions after the model has been trained.**

[6] The number of epochs is a hyperparameter that defines the number times that the learning algorithm will work through the entire training dataset [19]

### 7.2 Newsfeed dataset training and tuning for

Trying to tune and adjust the newsfeed dataset was even more complicated. The KNN algorithm was used under the premise that as the number of clusters grow for the model to be trained, more differences can be spotted after text preprocessing. The distances between the clusters should increase and get to a state enough to assert and properly distinguish the topics for the articles. On the other hand, a normal sentiment analysis from previously pretrained models were made just to visualize how much the algorithms could predict the positive or negative impact on specific articles. It was however not possible to segregate the clusters to a distance considerable enough to define topics on the news. This is a limitation addressed in **section 8.**

```
# run kmeans with many different k
distortions = []
#X_reduced =  df["processed_content"]
K = range(2, 50)
for k in tqdm.tqdm(K):
    k_means = KMeans(n_clusters=k, random_state=42).fit(X_reduced)
    k_means.fit(X_reduced)
    distortions.append(\
            sum(np.min(cdist(X_reduced, \
                k_means.cluster_centers_,\
                    'euclidean'), axis=1))\
                        / X.shape[0])
#%%
X_line = [K[0], K[-1]]
Y_line = [distortions[0], distortions[-1]]

# Plot the elbow
plt.plot(K, distortions, 'b-')
plt.plot(X_line, Y_line, 'r')
plt.xlabel('k')
plt.ylabel('Distortion')
plt.show()
```

**Figure 21. Code snippet to determine the best number of clusters**

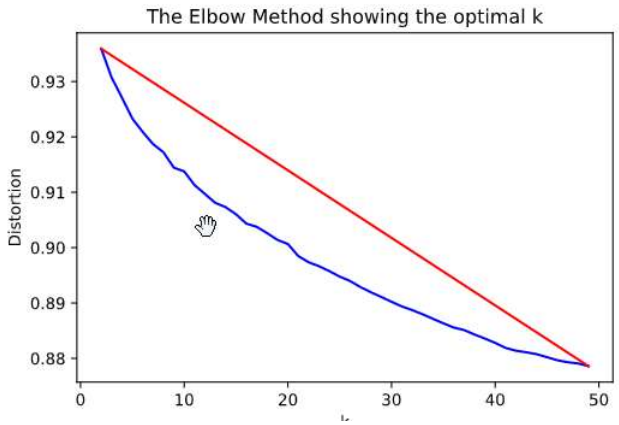



**Figure 22. Results plot to determine proper number of clusters**

After executing the code from **Figure 21,** the results to determine a suitable distance between clusters was performed. It was not possible to find a proper distinction between the news, the best fit for KNN cluster classification was 2, which is not suitable for the problem to solve. As the number of clusters increase, it is evident the classification algorithm is not able to distribute properly information among different groups, and therefore different approaches must be performed.

## 8. LIMITATIONS

Several limitations were encountered during the project. In some cases, the limitations could be partially avoided by modifying our use case, but in other cases intended implementations had to be dropped and noted as possible future work.

### 8.1 Limitations. Environment resources

One issue was regarding storage space. The three Virtual machines configured as slave nodes, which would be the machines to hold the data, each had 40 GB of storage. In addition to needing some storage for the operating system and applications needed to run the cluster, the Hadoop Distributed File System (HDFS) would store data across all nodes with redundancy, giving us a limited amount of storage. The Binance dataset consisted of 9.36 GB of data in parquet files, and in order to convert this data into a plaintext format for use in Machine Learning, it would require up to 72GB on each data node.

This limitation was addressed by performing a parquet file decompression mechanism from Spark, work with the model in memory and saving the trained model once the process was finished. This was a resource intensive mechanism and took more than 6 days to train the whole dataset.

### 8.2 Limitations. Dataset availability

No cryptocurrency-related newsfeed dataset could be found for use in Machine Learning. Normally to process NLP corpora and perform sentiment analysis over text is now available through different libraries, however it was observed that no pre-trained model nor available dataset open to the public has been yet generated. The solution was to create Python scripts for collecting news data (explained in a previous section) and try to exploit the obtained data to obtain relevant results that allowed our dataset to be classified. Because the data was unlabeled, using it to train a Machine Learning model would require much more analysis and processing in order to classify the data. Accurate results would require to manually read and label more than 50% of the news catalog (an equivalent of 15,000 different news) and afterwards used them to train a

model and test it over the unlabeled data. This task was so much time consuming and neither of the NLP tools employed to explore the dataset provided relevant information that allowed the process to be automated or slightly simplified.

### 8.3 Limitations. Spark state of the art

The last big limitation encountered, was regarding the execution of Deep Learning algorithms in a distributed fashion. Although already existent SparkML for distributed Machine Learning training on clusters, there is no currently an algorithm available for deep learning process, something necessary required for the provided dataset. Fr this reason TensorFlow must be configured in the cluster to work with Spark, which also created a challenge with regards to distributing the resources when training Machine Learning models. Nevertheless, before implementing Tensorflow natively in python in the cluster an exhaustive research was made to try to address such limitation. It turns out there are already attempts to make deep learning-TensorFlow algorithms to work in a distributed manner under Spark clusters, SparkFlow [13] is a library that tries to implement deep learning on Spark, however, after several attempts to make it work and configure with the TensorFlow algorithms used for the project, several incompatibility issued arose. The final solution therefore was to keep the non-distributed pySpark-TensorFlow approach.

## 9 FUTURE WORK

Possible future work includes classifying the collected news data, addressing the limitation discussed in section 8.2. This would allow the training of a Machine Learning model able to perform sentiment analysis on extracted news, labeling data as positive or negative with regards to cryptocurrency market impact.

Combining historical price data and labeled news data, a more accurate model for predicting cryptocurrency prices could be realized. Implementing automatic collection of price and news data on the Hadoop cluster, with the help of real-time sentiment analysis, would allow for a continuously improving prediction model.

Possible further improvements could be made by also considering factors such as technological improvements and social adoption.

## 10 CONCLUSIONS

The idea of combining social media information with cryptocurrency trading remains unexplored for the lack of information available. This presents a potential to mature the project as a product with models to be trained on daily basis once new data is released. Although social media keeps nowadays a very important role for cryptocurrency trading decisions, there is no open source dataset nor code available to explore. The code developed for this project is now available on GitHub [8] and Kaggle [7]. It is therefore led for future work to exploit the findings on this project and tune the newsfeed dataset to properly classify it. Social media data such as Twitter would be interesting to be added to find the correlations and perform a Sentiment analysis for the trading decisions on daily basis. There would be, however, an inherent need to previously analyze such dataset. It is important to mention that beforehand the intention was to use any other social media dataset, no results were found, therefore the need to create one by itself. Also the state of the art for distributed systems is not mature enough to address this kind of problems since deep learning cannot be run on clusters yet for parallel computations and calculations, will be very interesting to retake the project and explore future functionalities once this features are released in future versions.

## REFERENCES

[1] S. Nakamoto, "Bitcoin.org," 31 October3 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf.

[2] CoinMarketCap, "Bitcoin price, charts, market cap and other metrics," CoinMarketCap, 2020. [Online]. Available: https://coinmarketcap.com/currencies/bitcoin/ . [Accessed 26 04 2020].

[3] CoinMarketCap, "Cryptocurrency Market Capitalizations," CoinMarketCap, 2020. [Online]. Available: https://coinmarketcap.com. [Accessed 23 04 2020].

[4] Wikiipedia, "Cryptocurrency," Wikipedia, 04 2020. [Online]. Available: https://en.wikipedia.org/wiki/Cryptocurrency. [Accessed 23 04 2020].

[5] J. J. Smit, "Binance Full History 1 minute candlesticks for all 786 cryptocurrency pairs," Kaggle, 04 2020. [Online]. Available: https://www.kaggle.com/jorijnsmit/binance-full-history. [Accessed 25 04 2020].

[6] Binance, "Bitcoin Exchange, Cryptocurrency Exchange," Binance, 04 2020. [Online]. Available: https://www.binance.com/en. [Accessed 23 04 2020].

[7] A. C. Moreno, "Cryptocurency CoinTelegraph Newsfeed," Kaggle, 04 2020. [Online]. Available: https://www.kaggle.com/asahicantu/. [Accessed 23 04 2020].

[8] A. Cantu and D. Urdal, "CoinTelegraph package," Github, 03 2020. [Online]. Available: https://github.com/asahicantu/DAT500/ tree/master/CoinTelegraph. [Accessed 23 04 2020].

[9] J. Brownlee, "When to Use MLP, CNN, and RNN Neural Networks," Machine Learning Mastery, 23 07 2018.

[Online]. Available: https://machinelearningmastery.com/when-to-use-mlp-cnn-and-rnn-neural-networks/. [Accessed 20 03 2020].

[10] K. Z. Y. a. D. F. Chen, "A LSTM-based method for stock returns prediction. A case study of China stock market," in *2015 IEEE International Conference IEEE, pp. 2823–2824*, 2015.

[11] Y. Ahmed, "Predicting stock prices using deep learning," 11 10 2019. [Online]. Available: https://towardsdatascience.com/getting-rich-quick-with-machine-learning-and-stock-market-predictions-696802da94fe. [Accessed 23 04 2020].

[12] A. Cantu, "Crypto Params," 20 03 2020. [Online]. Available: https://github.com/asahicantu/DAT500/blob/master/CryptoPrediction/crypto_params.py. [Accessed 23 04 2020].

[13] LifeOmic, "Github," 05 2019. [Online]. Available: https://github.com/lifeomic/sparkflow. [Accessed 23 04 2020].

[14] Wikipedia, "Binance-Wikipedia," Wikipedia, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Binance. [Accessed 23 04 2020].

[15] Pandas, "Pandas," Github, 2020. [Online]. Available: https://github.com/pandas-dev/pandas. [Accessed 23 04 2020].

[16] CoinTelegraph, "CoinTelegraph Bitcoin & Ethereum Blockchain news," CoinTelegraph, 04 2020. [Online]. Available: https://cointelegraph.com. [Accessed 23 04 2020].

[17] Wikipedia, "Recurrent Neural Network," Wikipedia, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Recurrent_neural_network. [Accessed 23 04 2020].

[18] C. Olah, "Understanding LSTM Networks," Github, 27 08 2015. [Online]. Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/. [Accessed 23 04 2020].