

# **BITCOIN PRICE ANALYSIS**

Submitted in partial fulfillment of the requirements for the  
award of  
Bachelor of Engineering degree in Computer Science and Engineering  
by

**TATAVARTHI VAMSI KRISHNA (Roll No. 17SCS0458)**

**VANKAYALA SRI VENKATA NAGA SAI VAMSI (Roll No. 17SCS0483)**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
SCHOOL OF COMPUTING**

# **SATHYABAMA**

**INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade “A” by NAAC JEPPIAAR NAGAR, RAJIV GANDHI  
SALAI, CHENNAI – 600 119**

**MARCH – 2021**



# **SATHYABAMA**



INSTITUTE OF SCIENCE AND TECHNOLOGY

**(DEEMED TO BE UNIVERSITY)**

Accredited with "A" grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

[www.sathyabama.ac.in](http://www.sathyabama.ac.in)

## **DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

### **BONAFIDE CERTIFICATE**

This is to certify that this project report is the bonafide work of **TATAVARTHY VAMSI KRISHNA (Roll No. 17SCS0458)** and **VANKAYALA SRI VENKATA NAGA SAI VAMSI (Roll No. 17SCS0483)** who carried out the project entitled **"BITCOIN PRICE ANALYSIS"** under my supervision from **November 2020 to March 2021**.

**Internal Guide**

**Mrs. L.K. JOSHILA GRACE, M.E., PhD**

**Head of the Department**

**Dr. S. Vigneshwari, M.E., Ph.D.,**

**Dr. L. Lakshmanan, M.E., Ph.D.,**

Submitted for Viva voce Examination held on \_\_\_\_\_

---

**Internal Examiner**

**External Examiner**

## **DECLARATION**

**I TATAVARTHY VAMSI KRISHNA, VANKAYALA SRI VENKATA NAGA SAI VAMSI** hereby declare that the Project Report entitled “**BITCOIN PRICE ANALYSIS**” is done by me under the guidance of **Mrs.L.K.JOSHILA GRACE, M.E.**, Department of Computer Science and Engineering at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

**DATE:**

**PLACE: CHENNAI**

**SIGNATURE OF THE CANDIDATE**

## ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph.D., Dean, School of Computing,**

I convey my thanks to **Dr.T.Sasikala M.E., Ph.D., Dean, School of Computing** and **Dr.L.Lakshmanan M.E., Ph.D., and Dr.S.Vigneshwari M.E., Ph.D.,** Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Mrs.L.K.JOSHILA GRACE, M.E., Assistant Professor,** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

## **ABSTRACT**

This work aims to enhance and tried to analyze and predict the price of a Bitcoin by taking some parameters into consideration. After a huge research taking all the parameters which affect the bitcoin value and identified daily changes in the bitcoin market. In this work all the data consists of different features over the past few year's daily records. This project is started by gaining all the information that all are needed to predict the bitcoin price. All the information was collected from the past few years and implemented the data into this project. In this work Support Vector Machine (SVM) algorithm is used as it gives much more accuracy better than previous algorithms. This study predicts sign of change in the price of bitcoin to the investors so that they can invest in this easily and also for the new-comers to this market or business.

## TABLE OF CONTENTS

|                        |             |
|------------------------|-------------|
| <b>ABSTRACT</b>        | <b>v</b>    |
| <b>LIST OF FIGURES</b> | <b>viii</b> |

| <b>CHAPTER No.</b> | <b>TITLE</b>                            | <b>PAGE No.</b> |
|--------------------|-----------------------------------------|-----------------|
| <b>1.</b>          | <b>INTRODUCTION</b>                     | <b>1</b>        |
|                    | 1.1 WHAT IS BIT COIN                    | 1               |
|                    | 1.2 OVERVIEW                            | 2               |
|                    | 1.3 APPLICATIONS                        | 5               |
| <b>2.</b>          | <b>LITERATURE SURVEY</b>                | <b>6</b>        |
|                    | 2.1 LITERATUREREVIEW                    | 6               |
| <b>3.</b>          | <b>AIM AND SCOPE</b>                    | <b>10</b>       |
|                    | 3.1 AIM OF THE PROJECT                  | 10              |
|                    | 3.1.1OBJECTIVES                         | 10              |
|                    | 3.2 SCOPE                               | 10              |
|                    | 3.2.1 ADVANTAGES                        | 11              |
|                    | 3.3 BIT COIN ANALYSIS                   | 11              |
| <b>4.</b>          | <b>SOFTWARE DEVELOPMENT METHODOLOGY</b> | <b>12</b>       |
|                    | 4.1 DESCRIPTION OF DIAGRAM              | 12              |
|                    | 4.2 SOFTWARE ENVIRONMENT                | 15              |
|                    | 4.3 ALGORITHM USED                      | 19              |
|                    | 4.4 MODULES USED                        | 28              |
|                    | 4.5 ARCHITECTURAL DESIGN                | 32              |

|           |                                   |           |
|-----------|-----------------------------------|-----------|
| <b>5.</b> | <b>RESULTS AND DISCUSSION</b>     | <b>33</b> |
| <b>6.</b> | <b>CONCLUSION AND FUTURE WORK</b> | <b>37</b> |
|           | <b>REFERENCES</b>                 | <b>38</b> |
|           | <b>APPENDIX</b>                   |           |
|           | A. SOURCE CODE                    | 40        |
|           | B. SCREEN SHOT                    | 43        |
|           | C. PALAGRISM REPORT               | 47        |

## LIST OF FIGURES

| Figure-number | name                           | Page number |
|---------------|--------------------------------|-------------|
| 1.1           | Bit coin                       | 1           |
| 1.2           | Bitcoin graph                  | 3           |
| 4.1.1         | System design                  | 12          |
| 4.3.1         | Our labeled data               | 20          |
| 4.3.2         | Hyper plane is simply a line   | 21          |
| 4.3.3         | A more complex data set        | 22          |
| 4.3.4         | Before division                | 23          |
| 4.3.5         | After division                 | 23          |
| 4.3.6         | Everything is neatly separated | 24          |
| 4.3.7         | Low regularization value       | 26          |
| 4.3.8         | High regularization value      | 26          |
| 4.3.9         | High gamma                     | 27          |
| 4.3.10        | Low gamma                      | 27          |
| 4.3.11        | Good margin                    | 28          |
| 4.3.12        | Bad margin                     | 28          |
| 4.5           | ARCHITECTURALDESIGN            | 32          |

|     |                         |    |
|-----|-------------------------|----|
| 5.1 | Data set                | 34 |
| 5.2 | Trained data set        | 35 |
| 5.3 | Output                  | 35 |
| 5.4 | Accuracy                | 36 |
| 6.1 | Importing libraries     | 43 |
| 6.2 | Performing operation    | 43 |
| 6.3 | Taking variables        | 44 |
| 6.4 | Importing SVM           | 44 |
| 6.5 | Predicting x variable   | 45 |
| 6.6 | Compare with y variable | 45 |
| 6.7 | Result                  | 46 |

# CHAPTER 1

## INTRODUCTION

### 1.1 What Is Bitcoin?

Bitcoin is an open-source, peer-to-peer digital currency. Among many other things, what makes Bitcoin unique is that it is the world's first completely decentralized digital-payments system. This may sound complicated, but the underlying concepts are not difficult to understand. Crypto-currencies, such as Bitcoin, are one of the most controversial and complex technological innovations in today's financial system. From the past few years bitcoin is one of the trending in the market more than the stocks. The value of a bitcoin (or BTC) has grown and fluctuated greatly, from pennies in its early days to more than \$260 at its peak in April 2013. The current market capitalization of the bitcoin economy is estimated to be more than \$1 billion.



***Fig 1.1 Bitcoin***

## **1.2 Overview:**

A cryptocurrency is a medium of exchange (currency) that is digital in form, non-governmental, and relies on cryptography for its security (Investopedia, 2018). It is the latter component of the definition that gives cryptocurrency its name. The vast majority of currency in the world comes in the form of exchangeable units that are issued and have their value controlled by governments. Nations will either allow their currency to float freely on the market, where the value derives in part from supply and demand, and is backed by the ability of that state to raise revenues. These are considered to be the strongest currencies. Other currencies have values that are fixed by the state – as a medium of exchange those official values may or may not reflect the ability of the state to raise funds, but the value is enforceable by law regardless.

The rapid development of digital currencies during the last decade is one of the most controversial and ambiguous innovations in the modern global economy. Rising technology changes the structure of economies, financial markets and payment methods. The world's financial markets have become more digital than ever before and cashless society is around the corner. Today's technology enables people to create their own money (digital cryptocurrency) and the functions of the central banks, as lenders of last resorts, are discussed and questioned. Bitcoin, as a financial phenomenon, as



**Fig 1.2 Bitcoin Graph**

well as other cryptocurrencies, are in fact data treated like money. Users (called “miners”) send and receive these cryptocurrencies (data) electronically from their computers in peer-to-peer network systems to pay for things, if other parties are willing to accept such payments. Market capitalization and the number of miners of 2957 crypto-currencies reached \$221 billion (Bitcoin \$147) and 42 million in 2019.

The price of Bitcoin has drastically increased from \$0.0008 to \$10,168 per single coin from being launched in January 2009 to February 2020. Hence, first and foremost, the Bitcoin and other cryptocurrencies have become extremely popular due to increasing number of their users and their huge gains. On the other hand, Bitcoin’s and other cryptocurrencies’ prices-series, similar to other financial assets-series, exhibit chaotic fluctuations. Because of asymmetric information problems in financial markets, increasing economic-political uncertainties and changing behaviors of miners may make the prices of cryptocurrencies not easily predictable for investors. Cryptocurrencies’ forecasting difficulties may well be higher than those of other conventional assets; although they are so popular for investors, very little is known about them, about how they work and how they are created (mined), since they are

not physical currencies. Accordingly, accurately forecasting their prices may minimize potential losses-risks for users.

This market is characterized by high volatility, no closed trading periods, relatively smaller capitalization, and high market data availability . The financial feasibility of the cryptocurrency market in relation to other markets has been documented and the algorithms upon which the cryptocurrencies operate have been validated in other fields as well . The cryptocurrency market seems to behave independently from the other financial markets , but there is a strongly influenced by Asian economies . Part of the appeal behind this market is that the technology used for mining cryptocurrency provides feasible alternative to more traditional markets such as gold . These characteristics have attracted a considerable amount of capital, however up to now there are few studies that have attempted to create profitable trading strategies in the cryptocurrency market. Another point of interest in the cryptocurrency market is the large-scale of available public sentiment data, particularly from social networks. This data can presumably be used to infer future human behavior, and therefore could be used to develop advantageous trading strategies as has been shown in recent attempts to detect speculative bubbles in the cryptocurrency market using sentiment analysis.

The complexity of the task can be attributed to the multiple factors and uncertainties that interact in the markets including economic and political conditions, as well as human behavior. Being able to consistently predict the market price movements is quite difficult, but not impossible. According to academic research, movements in the market prices are not random, but behave in a highly non-linear and dynamic way. Previous studies have also shown that it is not necessary to be able to foretell the exact value of the future price in order to make profit in financial predictions. In reality, predicting the market direction as compared to its value can result in higher profits

### **1.3 Applications:**

Lower Transaction Costs Because there is no third-party intermediary, Bitcoin transactions are substantially cheaper and quicker than traditional payment networks. And because transactions are cheaper, Bitcoin makes micropayments and other innovations possible. Additionally, Bitcoin holds much promise as a way to lower transaction costs for small businesses and global remittances, alleviate global poverty by improving access to capital, protect individuals against capital controls and censorship, ensure financial privacy for oppressed groups, and spur innovation (within and on top of the Bitcoin protocol). On the other hand, Bitcoin's decentralized nature also presents opportunities for crime. The challenge, then, is to develop processes that diminish the opportunities for criminality while maintaining the benefits that Bitcoin can provide.

Since Bitcoin facilitates direct transactions without a third party, it removes costly charges that accompany credit card transactions. The Founders Fund, the venture capital fund headed by Peter Thiel of PayPal and Facebook fame, recently invested \$3 million in the payment-processing company BitPay because of the service's ability to lower the costs of doing online commerce across borders. In fact, small businesses have already started to accept bitcoin as a way to avoid the costs of doing business with credit card companies Others have adopted the currency for its speed and efficiency in facilitating transactions. Bitcoin will likely continue to lower transaction costs for businesses that accept it as more people adopt the currency.

## CHAPTER 2

### LITERATURE SURVEY

#### 2.1 Literature Review:

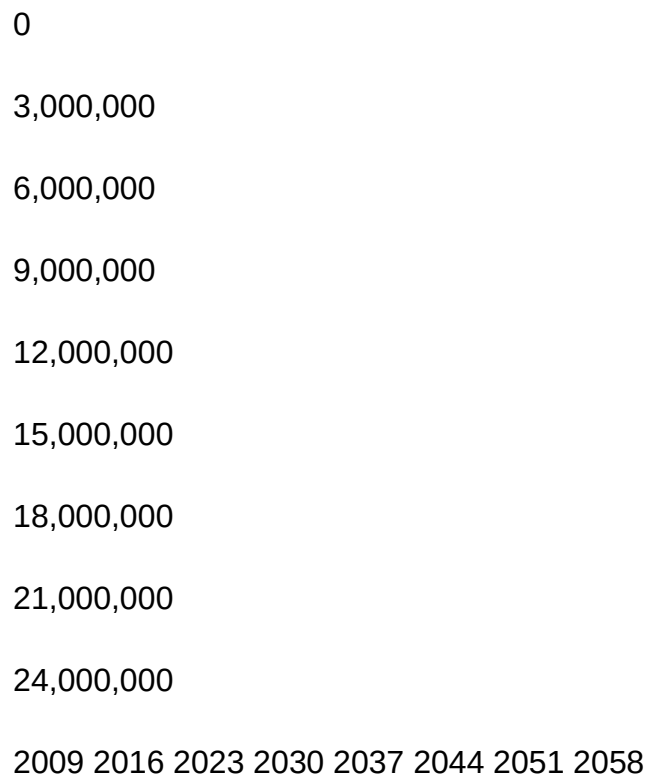
Bitcoin is currently a new technology and the world most expensive cryptocurrency thus there are some price prediction models available. Amjad et al. utilized the historical time series price data for price forecast and exchanging methodology [3] and Garcia et al. also appeared that the increments in opinion polarization and trade volume precede rising of Bitcoin prices [3]. Chen and Lazer [5] determined investment methodologies by observing and classifying the twitter feeds. Go et al. train the classifiers utilizing the dataset clarified by distant supervision and approve the classification performance [1]. Go et al. refer to the powerful paper by Pang et al. [8] where those analysts have set a standard for machine-learning based opinion analysis. Their approach is credited as one of the primary attempts at applying machine learning strategies to the issue of opinion analysis [10]. Some recent works focused on high-frequency trading and applying deep-learning techniques such as RNN for the prediction on time series data that have been tested dense, feed-forward networks as function model [10]. McNally [9] predicts the Bitcoin pricing process using machine learning techniques, such as recurrent neural networks (RNNs) and long short-term memory (LSTM) and compare results with those obtained using auto regressive integrated moving average (ARIMA) models. From [9] a comparison between multi-layer perceptron MLP and non-linear auto regressive exogenous (NARX) model is made. They conclude that MLP can also be used for stock market prediction even though it does not outperform the NARX model in price prediction. The authors made use of MATLAB's neural network toolbox to build and evaluate the performance of the network. Another paper [5] deals with daily time series data 10-minute and 10-second time-interval data. They have created three time-series data sets for 30 60 and 120 minutes followed by performing GLM/random forest on the datasets which produce three linear models. These

three models are linearly combined to predict the price of bitcoin. According to [6] the author is analyzing what has been done to predict the U.S. stock market. the conclusion of his work is the mean square error of the prediction network was as large as the standard deviation of the excess return. However, the author is providing evidence that several basic financial and economic factors have predictive power for the market excess return. instead of directly forecasting the future price of the stock the authors predict the trend of the stock. The trend can be considered as a pattern. They perform both short-term predictions day or week predictions and also long-term predictions months they found that the latter produced better results with 79% accuracy. Another interesting approach the paper reflects is the performance evaluation criteria of the network. Based on the predicted output the performance evaluation algorithm decides to either buy and sell or hold the stock. In this paper, we explored some of the relevant methods.

Artificial Neural Network ANN models are one of the most popular forecasting tools that has been widely used in various industries, such as engineering, finance and healthcare, due to its capability in performing complex tasks, such as classification, pattern recognitions and predictions, with great accuracies. ANN success in these areas is largely attributed to some of its key defining features as these models are nonlinear models and are powerful universal functional approximations that can approximate any continuous functions to a desired level of accuracy [1]. As such, ANN models have been seen as a viable alternative to the traditional statistical models for forecasting problems [18-27]. In one study, a literature review of comparative studies between ANN and traditional statistical models has found that out of 96 studies, only 18% of the cases concluded that traditional methods have outperformed ANN while ANN models have either performed well or outperformed in 72% of the cases [11]. Hence, it is unsurprising that neural networks have been Several authors have attempted to describe Bitcoin as a currency, stock, or asset. Yermack (2013) argues that Bitcoin appears to behave more similar to a speculative store of value rather than a currency. Dwyer (2015), on the other hand, describes Bitcoin as an electronic currency that can be used to trade and

store in a personal balance sheet. Dwyer's argument is supported by Polasik et al. (2015), who adds that Bitcoin can operate as a medium of exchange alongside other payment technologies. An increasing number of researchers have focused on the existence of a fundamental value of Bitcoin, and some have studied whether or not it is a bubble. Garcia et al. (2014) finds that Bitcoin is a financial bubble because of the difference between the exchange rate and fundamental value of Bitcoin. He argues for a fundamental value given the cost of mining. Similarly, Hayes (2015, 2018) proposed a specific cost of production model for valuating Bitcoin. Additionally, Cheah and Fry (2015) conclude that Bitcoin is a speculative bubble and that the fundamental value of Bitcoin is zero. Unlike earlier studies, Corbet et al. (2017) found that there is no clear evidence of a bubble in Bitcoin. While these authors discuss if Bitcoin is a bubble or not, Bouri et al. (2017b) found that Bitcoin could be used as an effective diversifier and, in some periods, also display safe-haven and hedge properties. Some studies have been dedicated to determining the factors that drive the price of Bitcoin. Bouoiyour and Selmi (2015) argue that long-term fundamentals are likely to be major contributors to Bitcoin price variations. Among others, they also found technical factors to be a positive driver of Bitcoin prices (Bouoiyour and Selmi 2015; Ciaian et al. 2016; Garcia et al. 2014; Georgoula et al. 2015; Hayes 2015; Kristoufek 2015). Specifically, Georgoula et al. (2015) and Hayes (2015) found the technical factor Hashrate to be a significant positive price driver. Bouoiyour and Selmi (2016), Garcia et al. (2014), Kristoufek (2015), Kjærland et al. (2018), and Sovbetov (2018) have all used Hashrate as a variable in their respective models. Other scholars also argue for the significance of fundamental factors such as exchange-trade, equity market indices, currency exchange rates, commodity prices, and transaction volume (Balcilar et al. 2017; Bouri et al. 2018a; Bouoiyour and Selmi 2016; Bouoiyour et al. 2016; Ciaian et al. 2016; Dyhrberg 2016; Kristoufek 2013; Yermack 2013). In contrast to Bouoiyour and Selmi (2015), Polasik et al. (2015) states that an increase in the transaction volume will lead to higher prices and that global economic factors do not seem to be an important driver. Ciaian et al. (2016) also found that supply and demand factors have strong impacts on price and that standard economic currency models can partly explain price fluctuations. Kristoufek (2013, 2015) analyzed the frequency of online searches on Bitcoin,

found them to be a good proxy for interest and popularity, and discovered that the relationship between the price of Bitcoin and online popularity is bidirectional. Ciaian et al. (2016) also found a positive relationship between Wikipedia searches and Bitcoin. Others argue along the same lines as Kristoufek in that it is primarily popularity and investor attractiveness that drive price movements (Bouoiyour et al. 2016, Mai et al. 2015).



## **CHAPTER 3**

### **AIM AND SCOPE**

#### **3.1 AIM OF THE PROJECT:**

The main aim of the project is to predict the price of a bitcoin on a specific date. Then train the data using machine learning algorithms and split them in the ratio 80:20 by using SVM algorithm. Where the 80% is trained and remaining 20% is tested based on the trained data. Then by using all the data we can predict the data for upcoming days.

##### **3.1.1 OBJECTIVES:**

The proposed model “Bitcoin Price Analysis” is based on the new algorithm which is not been used in the past which makes the whole process efficient and better than the previous versions. So that this algorithm helps us to increase the accuracy of the problem rather than the previous versions so that it will be very helpful for the investors and business man to invest in Bitcoin.

#### **3.2 SCOPE:**

This model will help us to increase the accuracy of the program which results in giving the better output. In this algorithm the null values will be taken as mean or median values where this is not present in previous versions. This surely helpful for investors for investing in Bitcoin.

### **3.2.1ADVANTAGES:**

- Time saving as it is operated online.
- All the data can be collected from online itself.
- It can hold up on a huge number of data.
- Easy to understand.

### **3.3 BITCOIN ANALYSIS:**

This study aims to forecast the movements of Bitcoin prices at high degree of accuracy. To this end, machine learning (henceforth, ML) algorithms are applied, which do not require strict assumptions like traditional methods, (e.g., regression analysis, discriminant analysis, cluster analysis, etc.). While traditional models use whole data to investigate causal relations, ML algorithms normally split the data set into training and testing sets. Hence, ML allows computers to “learn” and make predictions. Although both methods try to increase the accuracy by minimizing some loss functions, ML does so use nonlinear algorithms (Butner et al., 2019; Markakis et al., 2018). This does not mean that ML algorithms always outperform traditional models. However, ML algorithms, specially developed to address specific problems, may provide better forecasts for large data sets. All these make ML algorithms very popular for the scholars to apply. Many studies empirically compare traditional models and ML algorithms concerning their forecasting performances. In some studies, traditional models outperform ML algorithms, while in others the latter outperform the former. So we use SVM algorithm to the data and analyses and predict the data.

## CHAPTER 4

### SOFTWARE DEVELOPMENT AND METHODOLOGY

#### 4.1 DESCRIPTION OF DIAGRAM

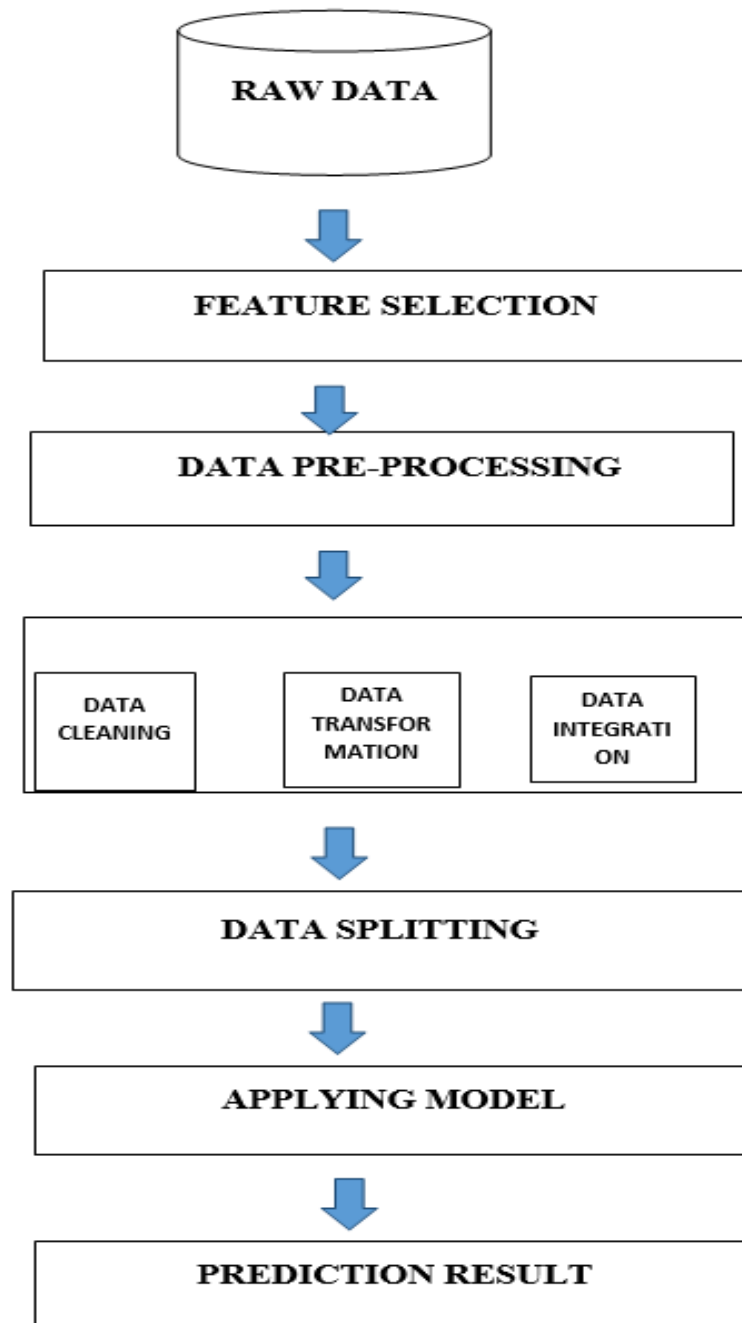


Fig 4.1.1 System Design

As by observing the above fig we can understand that it is a step by step process where one can say that it is a sequential process.

### ***Raw Data -***

As in the first step it is raw data where all the data is collected from the various resources. In this work, the data is collected from the last two years with parameters such as (date, price, closing price, opening price at investing.com, coin-desk.com, with the size of 5kb). There are many more websites to collect the data but these are the data collected to this project.

### ***Feature Selection -***

And the next step is about feature selection where as only the needed parameters are considered and taken into consideration so that it will be efficient for the program. As taking the raw data it consists of date, price, starting time, ending time etc., by taking all set of columns decreases the accuracy of the program. So, without effecting the program main things such as date and price are taken into consideration so that the accuracy will not be affected.

### ***Data Pre-processing -***

When we talk about data, we usually think of some large datasets with huge number of rows and columns. While that is a likely scenario, it is not always the case — data could be in so many different forms: Structured Tables, Images, Audio files, Videos etc.

Machines don't understand free text, image or video data as it is, they understand 1s and 0s. So it probably won't be good enough if we put on a slideshow of all our images and expect our machine learning model to get trained just by that. In any Machine Learning process, Data Pre-processing is that step in which the data gets transformed, or Encoded, to bring it to such a

state that now the machine can easily parse it. In other words, the features of the data can now be easily interpreted by the algorithm.

In this process all the missing cases are combined together when and wherever possible. In this regression models the data set is splitted into a ratio of 80:20 where the 80% is trained data set and remaining 20%is validation set. So by using machine learning algorithm we perform the training for 80%of the data set and then the 20% data is validated on the trained data. It has further more steps where one by one is done in a sequential way Data Cleaning, Data Transformation ,Data Integration

### ***Data Cleaning-***

In this method all the inaccurated data is corrected for all the record set..

Data cleaning is a technique that can remove un necessary data such as null values.., null valued data causes model may in accurate and miss leading of model prediction

```
print df['price'].isnull()
```

Above code shows the null values of the particular column

Filling the null values can be done by as follows

```
df['price'].median()
```

```
df['price'].fillna(median, inplace=True)
```

### ***Data Transformation -***

In this part data is to be transformed such as data like prices in dollars and some other price in rupees and some in the form of euros etc., . So all the data is transformed based on the convenient to the need.

### ***Data Integration -***

In this data is taken from the year 20018-2019 from website investing.com and another set of data from coin-desk.com from the year 2019-2020.And then the data is to be integrated into a single file. So that data integration is used in that way.

### ***Data Splitting -***

It splits the data into the two parts. which one part is for testing and another for training data with ratio 80%:20%

### ***Applying Model –***

Now the data is collected and all the modelling techniques are applied to the data. Support Vector Machine algorithm is applied to the data-set where 80% data is trained and remaining 20% data is tested for getting the resultant output.

### ***Prediction Result -***

Now the data is tested for 20% and checked with the trained data values. So by checking it can be able to predict the accuracy of the program.

## **4.2 SOFTWARE ENVIRONMENT-**

### **PYTHON**

Python is a high level and effective general use programming language. It supports multi-paradigms. Python has a large standard library which provide tools suited to perform various tasks. Python is a simple, less-clustered

language with extensive features and libraries. Different programming abilities are utilized for performing the experiment in our work. In this thesis, the following python libraries were used.

Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of Python's other implementations. Python and CPython are managed by the non-profit Python Software Foundation.

It is used for:

1. web development (server-side),
2. software development,
3. mathematics,
4. system scripting.

## **DATASET**

A Dataset is the basic data container in PyMVPA. It serves as the primary form of data storage, but also as a common container for results returned by most algorithms. The dataset assumes that the first axis of the data is to be used to define individual samples, usually presented in tabular form. Each column represents a particular variable. Each row corresponds to a given member of the dataset in question. It lists values for each of the variables, such as height and weight of an object

To train the model to predict the Bitcoin prices we need to have training data. Collect the data. The data might have some gaps.

Read the CSV files. The code is given below.

```
data = read.csv("../input/data.csv")
```

Remove the rows and columns that you will not use or which are not required.

Now, split the dataset into two parts. One becomes a training data set and another is test data set. Here we have taken 80% of the data set for training the model and the rest of the 20% are for test set.

```
n_train_rows = int(dataset.shape[0]*.8)-1
```

```
train = dataset.iloc[:n_train_rows, :]
```

```
test = dataset.iloc[n_train_rows:, :]
```

We are using LSTM algorithm here, so we need to organize our data in blocks. Our data is organized in 1 minute intervals. So we will be using 50 blocks to predict.

Now prepare the training data.

```
x_train = []
```

```
y_train = []
```

```
for i in range(steps, training_set_scaled.shape[0]-steps):
```

```
    x_train.append(training_set_scaled[i-steps:i, :])
```

```
    y_train.append(training_set_scaled[i, :])
```

```
x_train, y_train = np.array(x_train), np.array(y_train)
```

```
print(x_train.shape)
```

Create a variable named `prediction_days`. This will be an array. Set this equal to the last 30 days of the original data.

Following the above code, prepare test data.

Now plot the graph of test data and training data.

Now test the model accuracy.

Print the predicted values and actual values. Do this for the remaining next 30 days.

## **JUPYTER**

Jupyter Notebook is built off of IPython, an interactive way of running Python code in the terminal using the REPL model (Read-Eval-Print-Loop). The IPython Kernel runs the computations and communicates with the Jupyter Notebook front-end interface. It also allows Jupyter Notebook to support multiple languages.

Jupyter is a free, open-source, interactive web tool known as a computational notebook, which researchers can use to combine software code, computational output, explanatory text and multimedia resources in a single document.

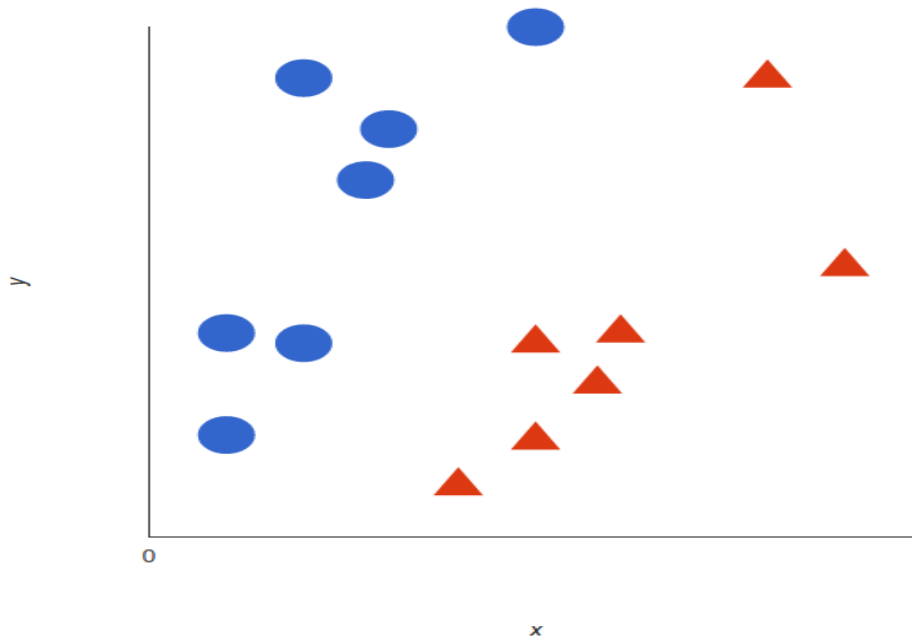
### **4.3 ALGORITHMS USED-**

#### **SVM (SUPPORT VECTOR MACHINE)-**

A support vector machine (SVM) is a supervised machine learning model that uses classification algorithms for two-group classification problems. After giving an SVM model sets of labeled training data for each category, they're able to categorize new text.

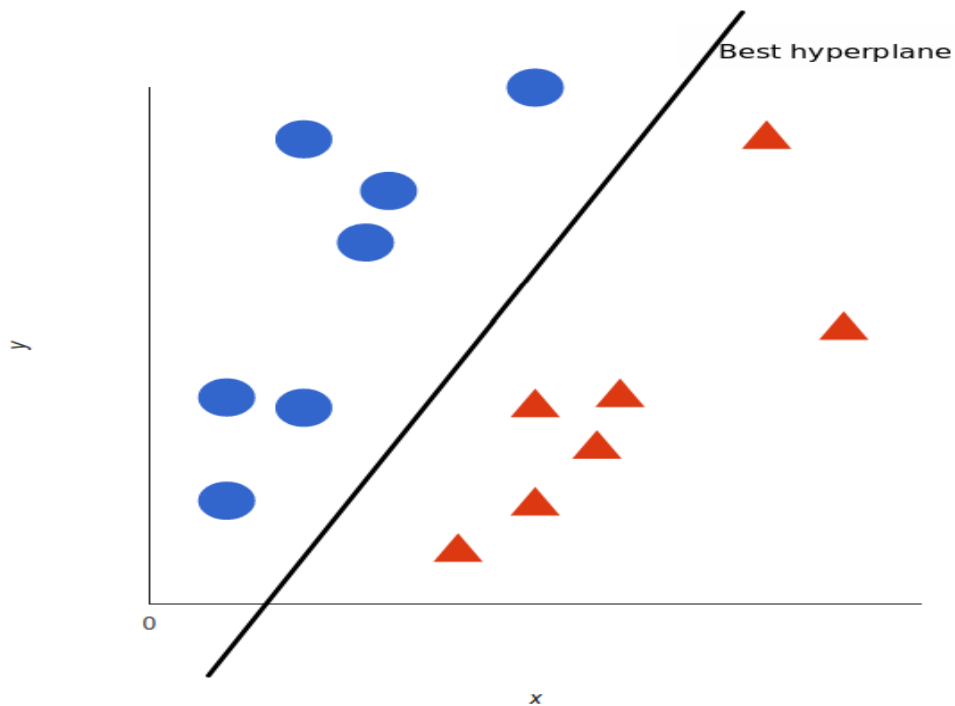
The basics of Support Vector Machines and how it works are best understood with a simple example. Let's imagine we have two tags: red and blue, and our data has two features:  $x$  and  $y$ . We want a classifier that, given a pair of  $(x,y)$  coordinates, outputs if it's either red or blue. We plot our already labeled

training data on a plane.



***Fig 4.3.1 Our labeled data***

A support vector machine takes these data points and outputs the hyperplane (which in two dimensions it's simply a line) that best separates the tags. This line is the decision boundary: anything that falls to one side of it we will classify as blue, and anything that falls to the other as red.

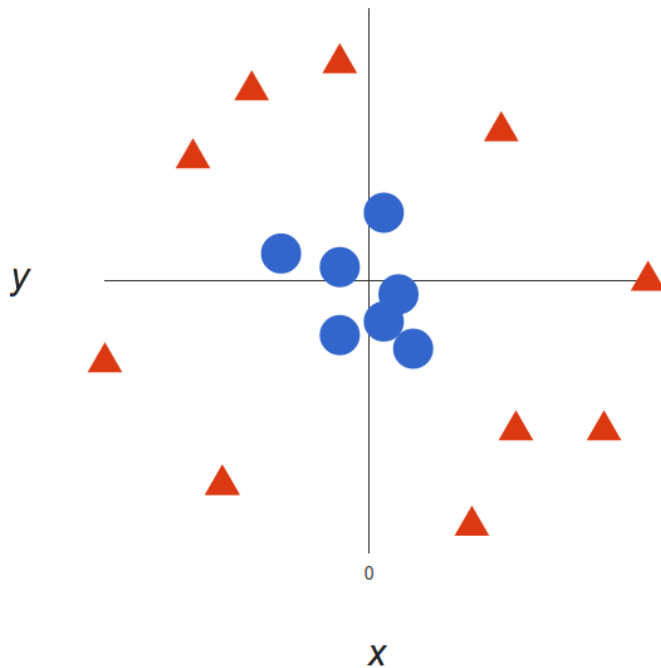


***Fig 4.3.2hyper plane is simply a line***

But, what exactly is the best hyperplane? For SVM, it's the one that maximizes the margins from both tags. In other words: the hyperplane (remember it's a line in this case) whose distance to the nearest element of each tag is the largest.

### **Nonlinear data-**

Now this example was easy, since clearly the data was linearly separable — we could draw a straight line to separate red and blue. Sadly, usually things aren't that simple.

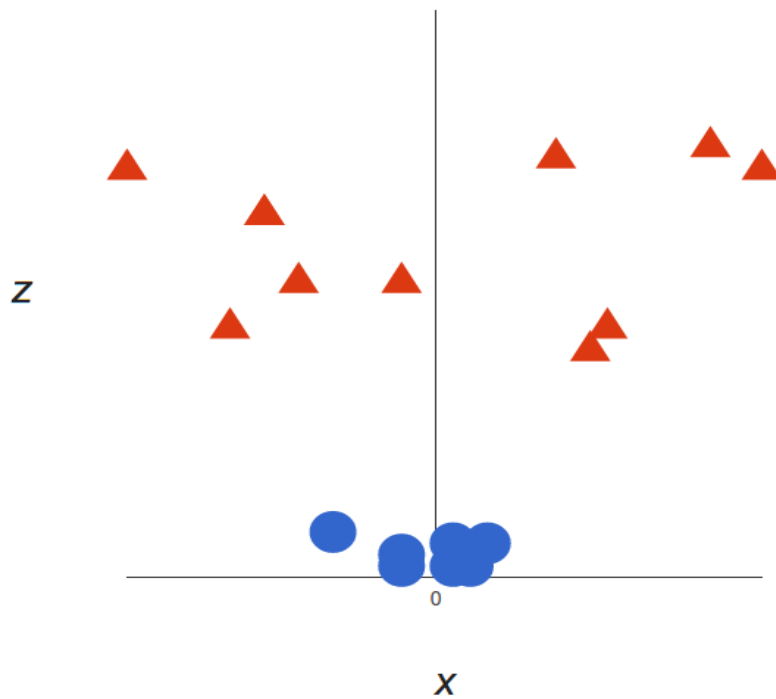


***Fig 4.3.3 A more complex dataset***

It's pretty clear that there's not a linear decision boundary (a single straight line that separates both tags). However, the vectors are very clearly segregated and it looks as though it should be easy to separate them.

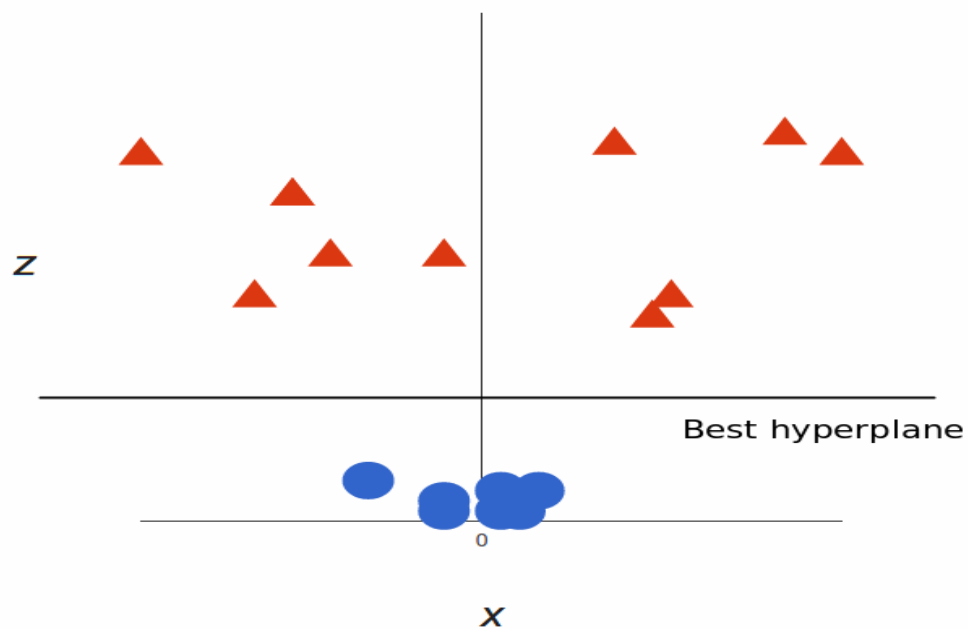
So here's what we'll do: we will add a third dimension. Up until now we had two dimensions: x and y. We create a new z dimension, and we rule that it be calculated a certain way that is convenient for us:  $z = x^2 + y^2$  (you'll notice that's the equation for a circle).

This will give us a three-dimensional space. Taking a slice of that space, it looks like this:



**Fig 4.3.4 Before Division**

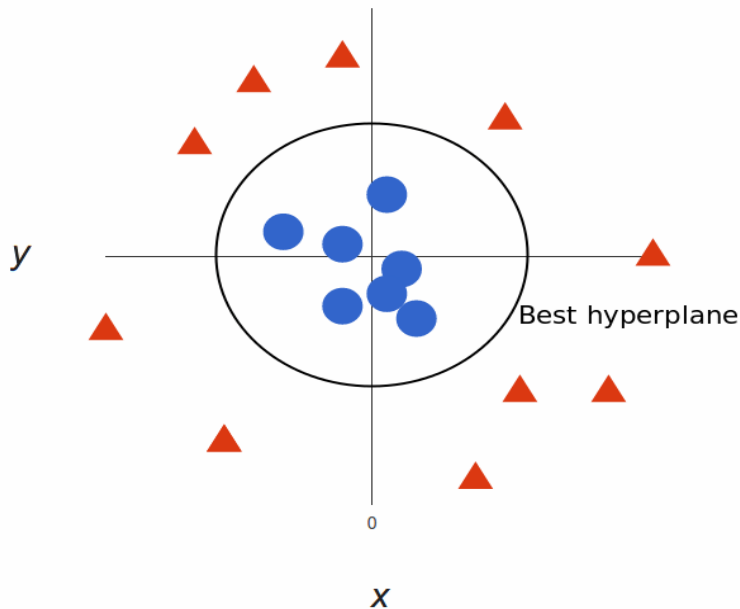
*From a different perspective, the data is now in two linearly separated groups*



**Fig 4.3.5 After Division**

That's great! Note that since we are in three dimensions now, the hyperplane is a plane parallel to the  $x$  axis at a certain  $z$  (let's say  $z = 1$ ).

What's left is mapping it back to two dimensions:



**Fig 4.3.6 Everything is neatly separated**

And there we go! Our decision boundary is a circumference of radius 1, which separates both tags using SVM. Check out this 3d visualization to see another example of the same effect:

And in this SVM algorithm we also have kernels

Kernel-

The learning of the hyperplane in linear SVM is done by transforming the problem using some linear algebra. This is where the kernel plays role.

For linear kernel the equation for prediction for a new input using the dot product between the input ( $x$ ) and each support vector ( $x_i$ ) is calculated as follows:

$$f(x) = B(0) + \sum(a_i * (x, x_i))$$

This is an equation that involves calculating the inner products of a new input vector (x) with all support vectors in training data. The coefficients B0 and ai (for each input) must be estimated from the training data by the learning algorithm.

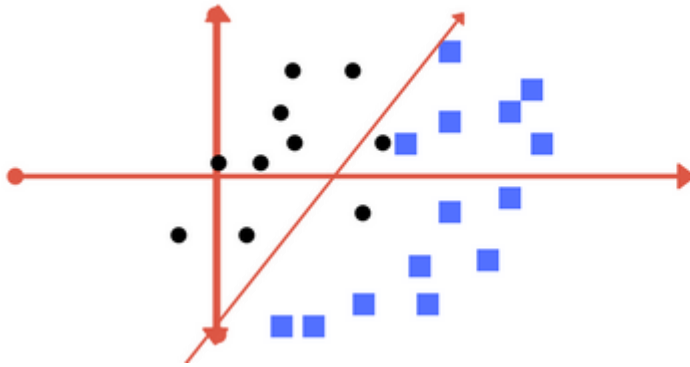
The polynomial kernel can be written as  $K(x, x_i) = 1 + \sum(x * x_i)^d$  and exponential as  $K(x, x_i) = \exp(-\gamma * \sum((x - x_i)^2))$ . *Polynomial and exponential kernels calculate separation line in higher dimension. This is called kernel trick*

### **Regularization**

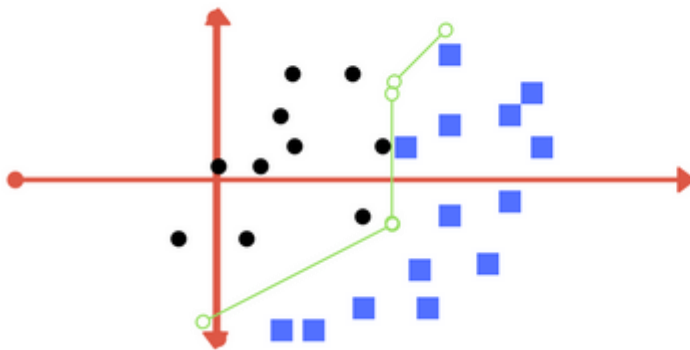
The Regularization parameter (often termed as C parameter in python's sklearn library) tells the SVM optimization how much you want to avoid misclassifying each training example.

For large values of C, the optimization will choose a smaller-margin hyperplane if that hyperplane does a better job of getting all the training points classified correctly. Conversely, a very small value of C will cause the optimizer to look for a larger-margin separating hyperplane, even if that hyperplane misclassifies more points.

The images below (same as image 1 and image 2 in section 2) are example of two different regularization parameter. Left one has some misclassification due to lower regularization value. Higher value leads to results like right one.



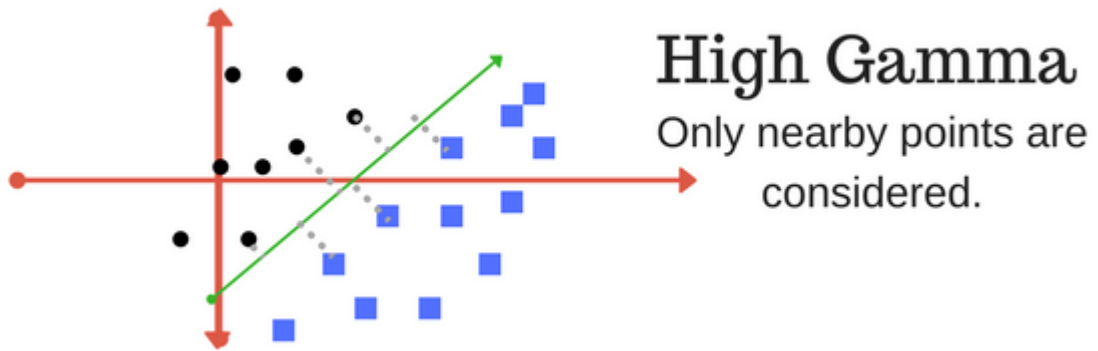
**4.3.7 Low regularization value**



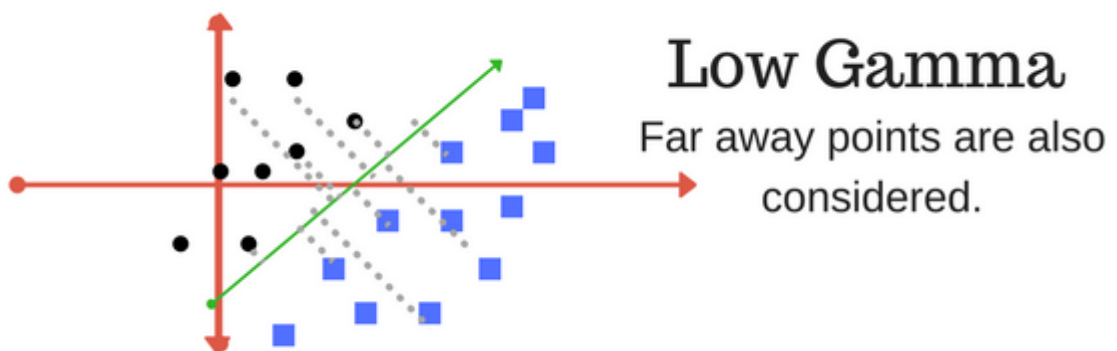
**4.3.8 High regularization value**

### **Gamma**

The gamma parameter defines how far the influence of a single training example reaches, with low values meaning 'far' and high values meaning 'close'. In other words, with low gamma, points far away from plausible separation line are considered in calculation for the separation line. Whereas high gamma means the points close to plausible line are considered in calculation.



**4.3.9 High Gamma**



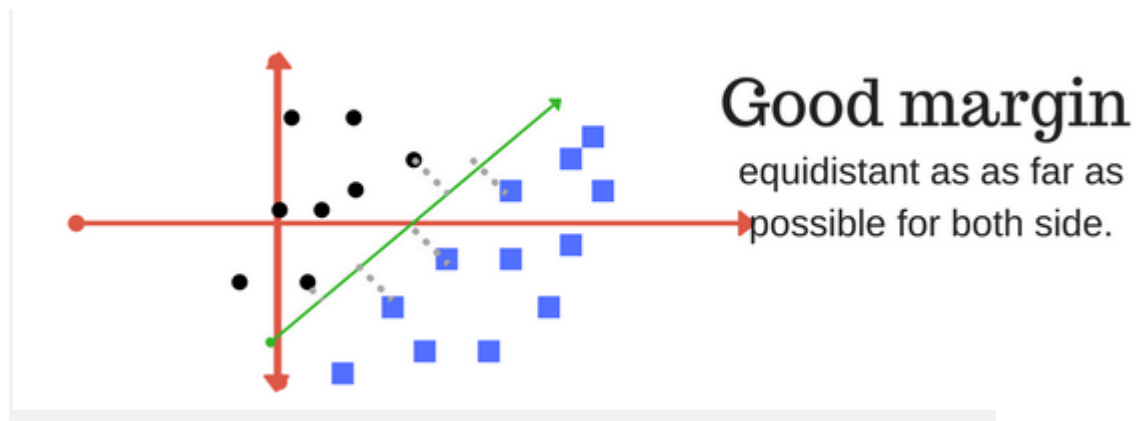
**4.3.10 Low Gamma**

### **Margin**

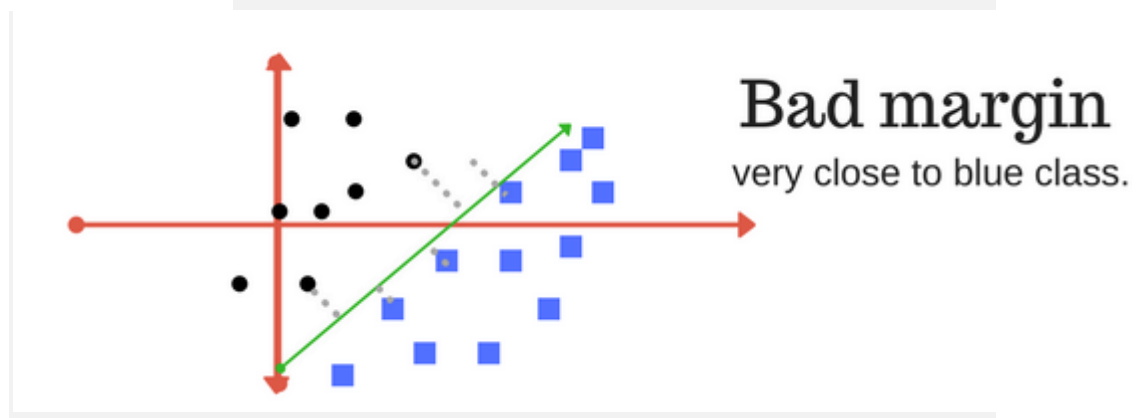
And finally last but very important characteristic of SVM classifier. SVM to core tries to achieve a good margin.

*A margin is a separation of line to the closest class points.*

A *good margin* is one where this separation is larger for both the classes. Images below gives to visual example of good and bad margin. A good margin allows the points to be in their respective classes without crossing to other class.



*Fig 4.3.11 Good Margin*



*Fig 4.3.12 Bad Margin*

#### 4.4 MODULES USED

##### ***NumPy-***

NumPy is a Python library used for working with arrays.

It also has functions for working in domain of linear algebra, fourier transform, and matrices.

NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely.

NumPy stands for Numerical Python.

In Python we have lists that serve the purpose of arrays, but they are slow to process.

NumPy aims to provide an array object that is up to 50x faster than traditional Python lists.

The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy.

### ***Pandas-***

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. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis/manipulation tool available in any language. It is already well on its way toward this goal.

pandas is well suited for many different kinds of data:

- Tabular data with heterogeneously-typed columns, as in an SQL table or Excel spreadsheet
- Ordered and un ordered (not necessarily fixed-frequency) time series data.
- Arbitrary matrix data (homogeneously typed or heterogeneous) with row and column labels
- Any other form of observational / statistical data sets. The data need not be labeled at all to be placed into a pandas data structure

The two primary data structures of pandas, Series (1-dimensional) and Data Frame (2-dimensional), handle the vast majority of typical use cases in finance, statistics, social science, and many areas of engineering. For R users, Data Frame provides everything that R's data. Frame provides and much more. pandas is built on top of NumPy and is intended to integrate well within a scientific computing environment with many other 3rd party libraries.

Here are just a few of the things that pandas does well:

- Easy handling of missing data (represented as NaN) in floating point as well as non-floating-point data
- Size mutability: columns can be inserted and deleted from Data Frame and higher dimensional objects
- Automatic and explicit data alignment: objects can be explicitly aligned to a set of labels, or the user can simply ignore the labels and let Series, Data Frame, etc. automatically align the data for you in computations
- Powerful, flexible group by functionality to perform split-apply-combine operations on data sets, for both aggregating and transforming data
- Make it easy to convert ragged, differently-indexed data in other Python and NumPy data structures into Data Frame objects
- Intelligent label-based slicing, fancy indexing, and sub setting of large data sets
- Intuitive merging and joining data sets.

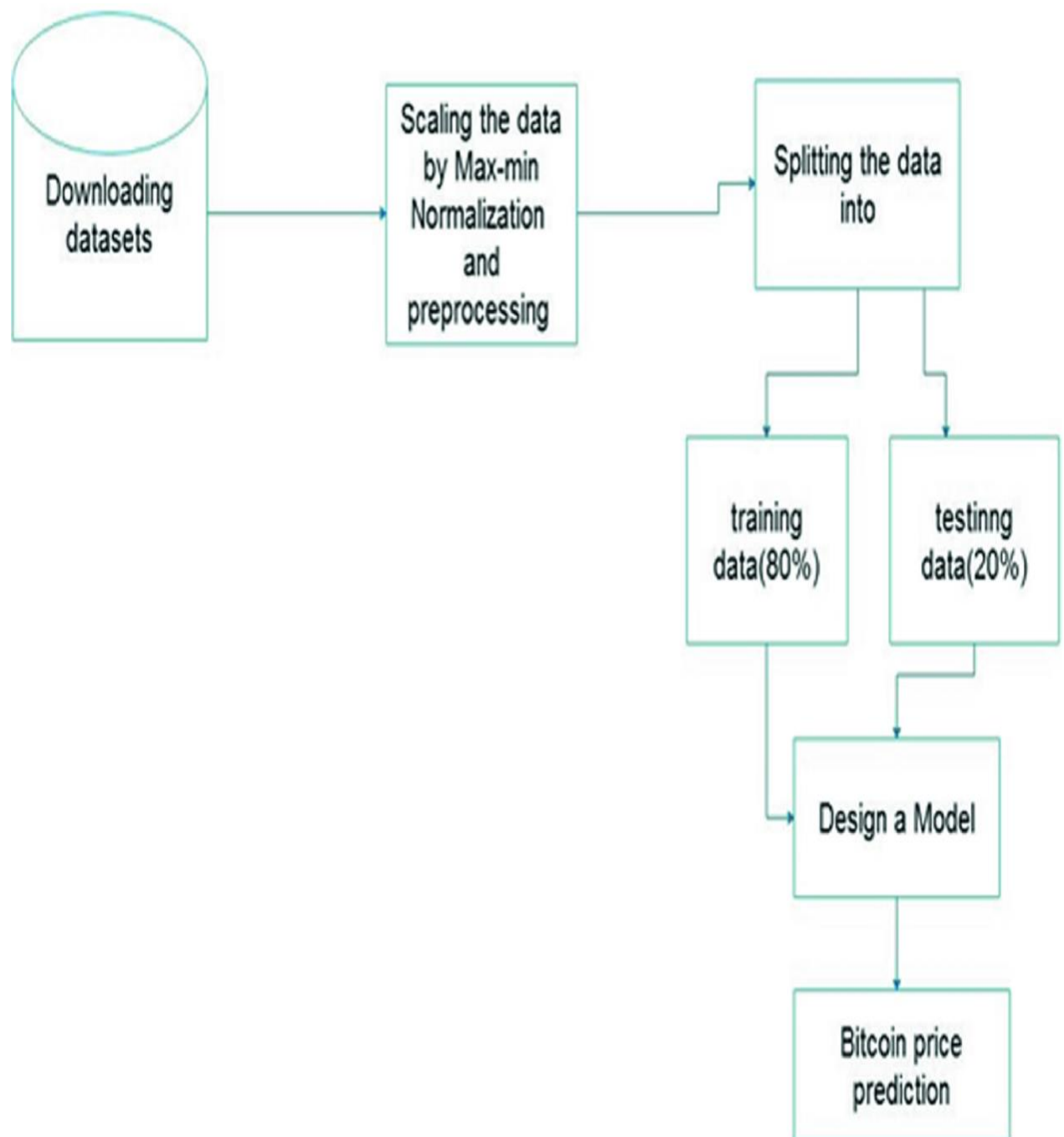
### **Scikit-learn-**

Scikit-learn (formerly `skits.learn` and also known as `sklearn`) is a free software machine learning library for the Python programming language.[3] It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, *k*-means and DBSCAN, and is designed to inter operate with the Python numerical and scientific libraries NumPy and SciPy.

Scikit-learn is largely written in Python, and uses `numpy` extensively for high-performance linear algebra and array operations. Furthermore, some core algorithms are written in Cpython to improve performance. Support vector machines are implemented by a Cpython wrapper around LIBSVM; logistic regression and linear support vector machines by a similar wrapper around LIBLINEAR. In such cases, extending these methods with Python may not be possible.

Scikit-learn integrates well with many other Python libraries, such as `matplotlib` and `plotly` for plotting, `numpy` for array vectorization, `pandas` data frames, `scipy`, and many more.

#### 4.5.ARCHITECTURAL DESIGN:



**Fig 4.5 ARCHITECTURAL DESIGN**



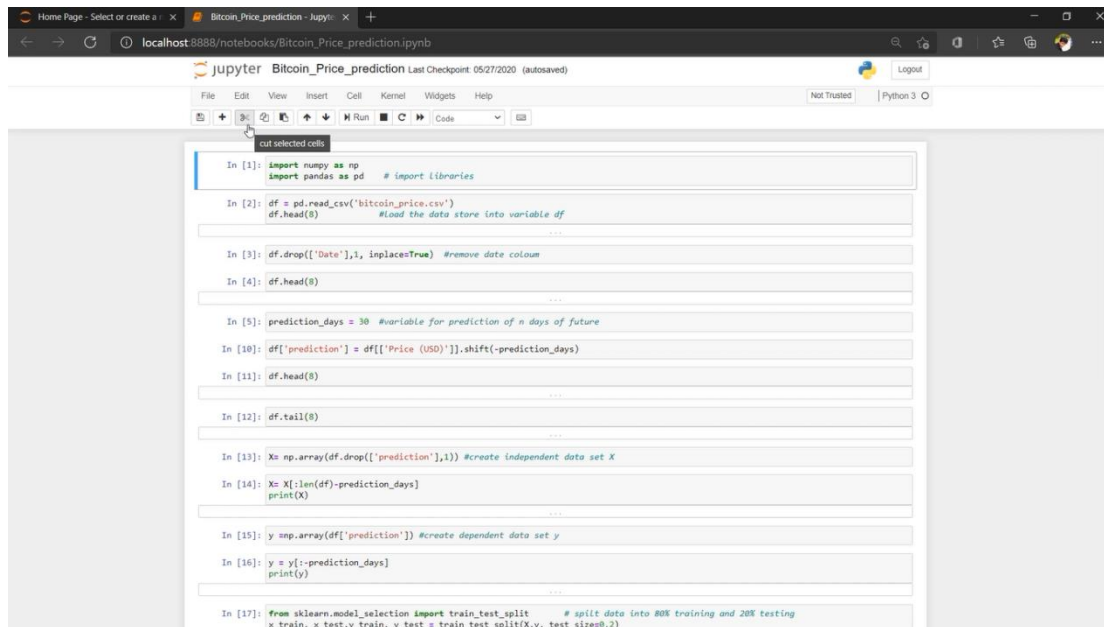
## CHAPTER 5

### RESULT AND DISCUSSION

This chapter presents the results that are obtained from the experiment. In this work, have developed and compared various algorithms based on machine learning. more specially work tested in random forest , Convolutional Neural Networks and combined both and also regression type but that algorithms are much more effective in other algorithms ,even in sequence data analysis but not in bit-coin prediction, why because when work did with random forest ,first it will convert in decision tree it will take high time to train data, in the same if work did in support vector machine (SVM) in this algorithm the main advantage is keras function .By using this function it is easy to calculate or manipulating the number functions, even it will take less time to train data when compared to other function and another advantage is time complexity is low when compared to Convolutional neural network and random forest, next about space complexity even in this category Support Vector Machine is very good. Finally if go to the accuracy, when work use support vector machine algorithm it was getting high when compared to other any algorithms so, in every case and support vector machine is very good and user friendly and even easy to understand.

Support vector machine is the classifier that maximizes the margin. The goal of a classifier in our example below is to find a line or  $(n-1)$  dimension hyper-plane that separates the two classes present in the  $n$ -dimensional space. The margin is calculated as the perpendicular distance from the line to only the closest points. Only these points are relevant in defining the line and in the construction of the classifier. These points are called the support vectors. They support or define the hyperplane. Supervised machine learning model that uses classification algorithms for two-group classification problems. After

giving an SVM model sets of labelled training data for each category, they're able to categorize new text. So you're working on a text classification problem. The support vector uses a mathematical function, often called a kernel function which is a math function that matches the new data to the best image from the training data in order to predict the unknown picture's label



```
In [1]: import numpy as np
import pandas as pd # import libraries

In [2]: df = pd.read_csv('bitcoin_price.csv')
df.head(8)

In [3]: df.drop(['Date'],1, inplace=True) #remove date column

In [4]: df.head(8)

In [5]: prediction_days = 30 #variable for prediction of n days of future

In [10]: df['prediction'] = df[['Price (USD)']].shift(-prediction_days)

In [11]: df.head(8)

In [12]: df.tail(8)

In [13]: X = np.array(df.drop(['prediction'],1)) #create independent data set X

In [14]: X = X[:len(df)-prediction_days]
print(X)

In [15]: y = np.array(df['prediction']) #create dependent data set y

In [16]: y = y[:len(df)-prediction_days]
print(y)

In [17]: from sklearn.model_selection import train_test_split # split data into 80% training and 20% testing
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

**Fig 5.1 Dataset**

So by observing the above figure5.1 at the starting of the program all the libraries are imported and dataset is collected. Then variables are taken and trained.

```

In [13]: from sklearn.model_selection import train_test_split # split data into 80% training and 20% testing
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

In [14]: prediction_days_array = np.array(df.drop(['prediction'],1)[-prediction_days:])

In [15]: from sklearn.svm import SVR

In [16]: svr_rbf = SVR(kernel='rbf', C=1e3, gamma=0.00001) #create and train Support vector Machine

In [17]: svr_rbf.fit(x_train, y_train)

Out[17]: SVR(C=1000.0, cache_size=200, coef0=0.0, degree=3, epsilon=0.1, gamma=1e-05,
kernel='rbf', max_iter=1, shrinking=True, tol=0.001, verbose=False)

In [18]: svr_rbf_confidence = svr_rbf.score(x_test, y_test)
print("svr_rbf accuracy: ", svr_rbf_confidence) #testing accuracy of model # the model shows 62% accuracy
svr_rbf accuracy: 0.6524554001875631

In [19]: svm_prediction = svr_rbf.predict(x_test)
print(svm_prediction) #print prediction values
[ 3851.74271928  8215.32452119  6314.26709113  6594.23321526
  6634.6438668  10252.89206775  8041.04612915  8435.73577355
  9133.81405866  8014.51773973  6447.07965124  9143.27199274
  6342.18007917  3853.2391926  4599.71153648  8270.50040046
  4481.67811816  8720.7350803  8421.99929024  6973.19340951
  3886.72285352  8296.34728944  6434.18428802  6290.9214239
  8635.014277  3866.44428037  8234.10032173  8849.80050327
  6471.51759161  6222.18304668  4195.83348054  6683.5260666
  3937.18008175  7798.85941906  6396.9134772  9983.03759811
  10415.83877441  8264.55126147  8290.21409922  6798.68217196
  6469.23975726  6486.36049022  8282.80525948  10501.527912
  6321.07209197  7801.59175428  7567.91430081  4392.6382783
  3852.12828776  8919.39710958  9356.33262835  4488.74684565
  7545.56547293  8541.9017956  8952.42107004  8699.40712446]

```

**Fig 5.2 Trained Dataset**

By observing the above fig5.2 svm is imported and all the modules are taken such as sklearn and training and testing is performed on the data set. At the end of the program is the value which is obtained by training.

```

In [21]: svm_prediction = svr_rbf.predict(prediction_days_array)
print(svm_prediction) #print the bitcoin price prediction
[7551.24236283  8806.05254923  6965.21404655  6688.96402331  6826.28751927
  6855.41740884  7602.53974696  7554.6733365  8418.0101107  7638.65630534
  8062.17933011  8184.28339406  7667.12067311  8435.28437521  7941.52581769
  7531.00850725  8044.96238739  8035.20564931  8046.65746905  8157.03477312
  7770.96920631  8389.52975258  7553.13638495  8065.70638543  8215.6697241
  7961.69013944  8162.55638994  8075.05674041  8128.18207586  8236.58216786]

In [ ]:

In [ ]:

In [ ]:

```

**Fig 5.3 Output**

So above shown fig 5.3 is the output in which Bitcoin price is predicted for the next 30 days.

```
7793.636818 8973.079277 9469.078423 9073.81753 9037.341927
9223.385534 8958.532234 9112.866151 9392.635447 9958.733998
10062.71591 9957.110928 9650.018603 9176.18639 8980.624505
9416.590183 9940.956082 9848.185361 9583.062439 9879.71204
9933.976225 9899.775994 9834.557819]
```

```
In [13]: from sklearn.model_selection import train_test_split # split data into 80% training and 20% testing
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

In [14]: prediction_days_array = np.array(df.drop(['prediction'],1))[-prediction_days:]

In [15]: from sklearn.svm import SVR

In [16]: svr_rbf = SVR(kernel='rbf', C=1e3, gamma=0.00001) #create and train Support vector Machine

In [17]: svr_rbf.fit(x_train, y_train)

Out[17]: SVR(C=1000.0, cache_size=200, coef0=0.0, degree=3, epsilon=0.1, gamma=1e-05,
kernel='rbf', max_iter=1, shrinking=True, tol=0.001, verbose=False)

In [18]: svr_rbf_confidence = svr_rbf.score(x_test, y_test)
print("svr_rbf accuracy: ", svr_rbf_confidence) #testing accuracy of model # the model shows 62% accuracy
svr_rbf accuracy: 0.6524554001875631

In [25]: svm_prediction = svr_rbf.predict(x_test)
print(svm_prediction) #print prediction values

In [26]: print(y_test)

In [27]: svm_prediction = svr_rbf.predict(prediction_days_array)
print(svm_prediction) #print the Bitcoin price prediction
```

**Fig 5.4 Accuracy**

So above fig gives the accuracy of the program which is 65% which is far better than all the other algorithms which are used previously for Bitcoin.

## **CHAPTER 6**

### **CONCLUSIONS AND FUTURE WORK**

This work, addresses short-term to mid-term BTC price forecasts using ML models. It is the primary study that takes into consideration all the worth indicators up to end of the year, 2019, and provides highly accurate end-of-day, short-term (7 days) and mid-term (30 and 90 days) BTC price forecasts using machine learning. Four sorts of ML models are used: ANN, SANN, Support Vector Machine and LSTM. The LSTM showed the best overall performance. This work results show that it's possible to forecast the particular BTC price with very low error rates, while it's much harder to forecast its rise and fall. The classification model performance scores presented are the simplest within the literature. Having said that, the classification models for Bitcoin got to be further studied. As further work, hourly BTC prices and technical indicators could also be utilized using ensemble models that combine different types of models that for creating forecast.

#### **FUTURE WORK-**

There are several research directions for future work. First, it would be interesting to consider other recent deep learning models, such as transformer networks and few-shot/one-shot learning, to build prediction models. But this model has disadvantages also because of the bitcoin price also influenced by political reasons also this can cause our model fails.

## REFERENCES

- [1] A. Go, L. Huang and R. Bhayani, "Twitter Sentiment Classification using Distant Supervision," Stanford Computer Science, 2009.
- [2] B. Pang, L. Lee and S. Vaithyanathan, "Thumbs up: sentiment classification using machine learning techniques," in ACL-02 conference on Empirical methods in natural language processing, Philadelphia, PA, USA, 2002.
- [3] D. Garcia and F. Schweitzer, "Social signals and algorithmic trading of Bitcoin," Royal Society Open Science, vol. 2, no. 9, 2015.
- [4] D. L. K. Chuen, Handbook of Digital Currency: Bitcoin, Innovation, Financial Instruments, and Big Data, Academic Press, 2015.
- [5] D. Shah and K. Zhang, "Bayesian regression and Bitcoin," in 52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton), 2015.
- [6] H. Jang and J. Lee, "An Empirical Study on Modelling and Prediction of Bitcoin Prices with Bayesian Neural Networks based on Block-chain Information," in IEEE Early Access Articles, 2017.
- [7] F. A. d. Oliveira, L. E. Zarate, M. d. A. Reis and C. N. Nobre, "The use of artificial neural networks in the analysis and prediction of stock prices," in IEEE International Conference on Systems, Man, and Cybernetics, 2011. [14] "Understanding LSTM Networks -- colah's blog," 27 August 2015. [Online]. Available
- [8] M. Amjad and D. Shah, "Trading Bitcoin and Online Time Series Prediction," in NIPS 2016 Time Series Workshop, 2017.5.
- [9] M. Daniela and A. BUTOI, "Data mining on Romanian stock market using neural networks for price prediction," Informatica Economica, vol. 17, no. 3, 2013.
- [10] M. Dixon, D. Klabjan and J. H. Bang, "Classification-based financial markets prediction using deep neural networks," ArXiv, 2017.

- [11] R. Chen and M. Lazer, "Sentiment Analysis of Twitter Feeds for the Prediction of Stock Market Movement," *Stanford Computer Science*, no. 229, 2011, p. 15.
- [12] S. McNally, J. Roche and S. Caton, "Predicting the price of Bitcoin using machine learning," in *26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP)*, 2018.
- [13] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.
- [14] Farokhmanesh, F., & Sadeghi, M. T. (2019). Deep Feature Selection using an Enhanced Sparse Group Lasso Algorithm. 2019 27th Iranian Conference on Electrical Engineering (ICEE).
- [15] Variable selection using the Lasso-Cox model with Bayesian regularization, Wenxin Lu ; Zhuliang Yu ; Zhenghui Gu ; Jinhong Huang ; Wei Gao ; Haiyu Zhou
- [16] —Robust Regression and Lasso Huan Xu ; Constantine Caramanis ; Shie Mannor IEEE 2018
- [17] D. Shah and K. Zhang, —Bayesian regression and Bitcoin, in *52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2015, pp. 409-415. [2]
- [18] Huisu Jang and Jaewook Lee, —An Empirical Study on Modelling and Prediction of Bitcoin Prices with Bayesian Neural Networks based on Blockchain Information, in *IEEE Early Access Articles*, 2017, vol. 99, pp. 1-1.
- [3] F.
- [19] Andrade de Oliveira, L. Enrique ZÃrate and M. de Azevedo Reis; C. Neri Nobre, —The use of artificial neural networks in the analysis and prediction of stock prices, in *IEEE International Conference on Systems, Man, and Cybernetics*, 2011, pp. 2151-2155.

## SOURCE CODE-

```
# -*- coding: utf-8 -*-
```

```
"""Bitcoin_Price_prediction.ipynb
```

Automatically generated by Colaboratory.

Original file is located at

<https://colab.research.google.com/drive/1yTqGolH7ta9JpqnfS3l6fT8QjNiSbaV>

q

```
.....
```

```
import numpy as np
```

```
import pandas as pd # import libraries
```

```
df = pd.read_csv('bitcoin_price.csv')
```

```
df.head(8) #load the data store into variable df
```

```
df.drop(['Date'],1, inplace=True) #remove date coloum
```

```
df.head(8)
```

```
prediction_days = 30 #variable for prediction of n days of future
```

```
df['prediction'] = df[['Price (USD)']].shift(-prediction_days)
```

```
df.head(8)
```

```
df.tail(8)
```

```
X= np.array(df.drop(['prediction'],1)) #create independent data set X
```

```
X= X[:len(df)-prediction_days]
```

```
print(X)
```

```
y =np.array(df['prediction']) #create dependent data set y
```

```
y = y[:-prediction_days]
```

```
print(y)
```

```
from sklearn.model_selection import train_test_split      # spilt data into 80%  
training and 20% testing
```

```
x_train, x_test,y_train, y_test = train_test_split(X,y, test_size=0.2)
```

```
prediction_days_array = np.array(df.drop(['prediction'],1))[:-prediction_days:]
```

```
print(prediction_days_array)
```

```
from sklearn.svm import SVR
```

```
svr_rbf = SVR(kernel='rbf', C=1e3, gamma=0.00001) #create and train Support  
vector Machine
```

```
svr_rbf.fit(x_train, y_train)
```

```
svr_rbf_confidence = svr_rbf.score(x_test, y_test)
```

```
print("svr_rbf accuracy: ", svr_rbf_confidence) #testing accuracy of model # the  
model shows 62% accuracy
```

```
svm_prediction = svr_rbf.predict(x_test)
```

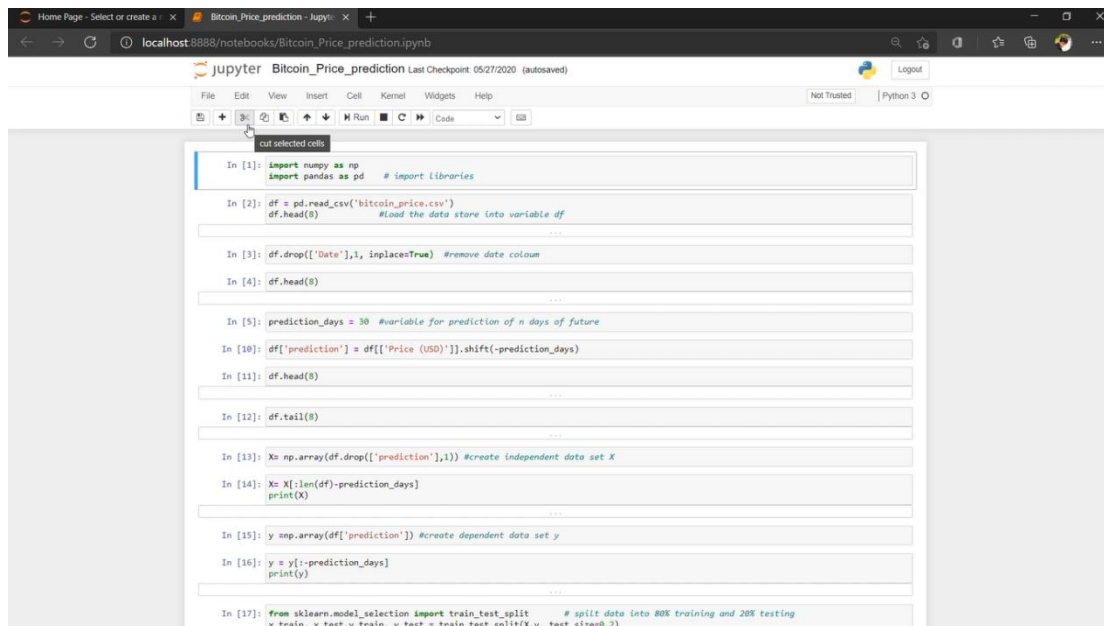
```
print(svm_prediction)    #print prediction values
```

```
print(y_test)
```

```
svm_prediction = svr_rbf.predict(prediction_days_array)
```

```
print(svm_prediction) #print the bitcoin price prediction
```

## SCREEN SHOTS-



The screenshot shows a Jupyter Notebook interface with the title "Bitcoin\_Price\_prediction". The notebook contains 17 code cells. The first cell imports numpy and pandas. The second cell reads a CSV file named "bitcoin\_price.csv" and displays the first 8 rows. The third cell drops the "Date" column. The fourth cell displays the first 8 rows of the resulting DataFrame. The fifth cell sets "prediction\_days" to 30. The sixth cell shifts the "Price (USD)" column by -prediction\_days. The seventh cell displays the first 8 rows. The eighth cell displays the last 8 rows. The ninth cell creates an independent data set X. The tenth cell prints X. The eleventh cell creates a dependent data set y. The twelfth cell prints y. The thirteenth cell splits the data into 80% training and 20% testing sets.

```
In [1]: import numpy as np
import pandas as pd # import libraries

In [2]: df = pd.read_csv('bitcoin_price.csv')
df.head(8)

Out[2]:
Date Price (USD)
0 4/19/2018 8253.86625
1 4/20/2018 8380.25124
2 4/21/2018 8930.43000
3 4/22/2018 9021.07000
4 4/23/2018 9009.82000
5 4/24/2018 9220.96750
6 4/25/2018 8246.81749
7 4/26/2018 9514.41375

In [3]: df.drop(['Date'],1, inplace=True) #remove date column

In [4]: df.head(8)

Out[4]:
Price (USD)
0 8253.86625
1 8380.25124
2 8930.43000
3 9021.07000
4 9009.82000
5 9220.96750
6 8246.81749
7 9514.41375

In [5]: prediction_days = 30 #variable for prediction of n days of future

In [10]: df['prediction'] = df[['Price (USD)']].shift(-prediction_days)

In [11]: df.head(8)

Out[11]:
Price (USD) prediction
0 8253.86625 8253.86625
1 8380.25124 8380.25124
2 8930.43000 8930.43000
3 9021.07000 9021.07000
4 9009.82000 9009.82000
5 9220.96750 9220.96750
6 8246.81749 8246.81749
7 9514.41375 9514.41375

In [12]: df.tail(8)

Out[12]:
Price (USD) prediction
1 8380.25124 8380.25124
2 8930.43000 8930.43000
3 9021.07000 9021.07000
4 9009.82000 9009.82000
5 9220.96750 9220.96750
6 8246.81749 8246.81749
7 9514.41375 9514.41375

In [13]: X= np.array(df.drop(['prediction'],1)) #create independent data set X

In [14]: X=X[:len(df)-prediction_days]
print(X)

Out[14]:
[[8253.86625 8380.25124 8930.43  9021.07  9009.82  9220.9675  8246.81749
  9514.41375]]

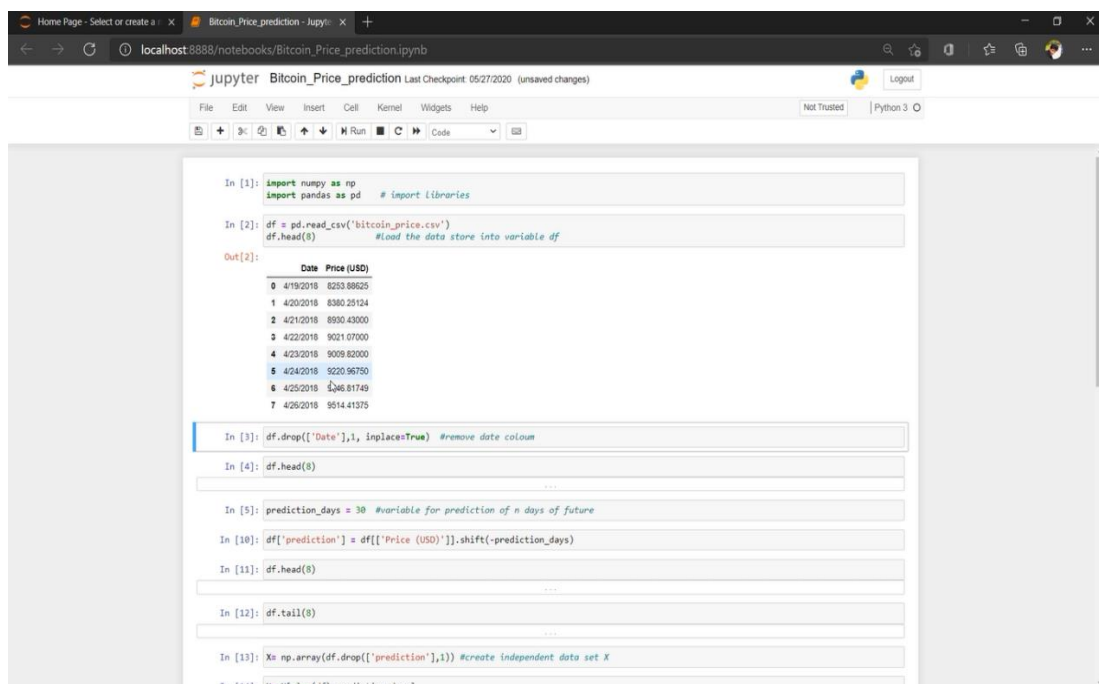
In [15]: y= np.array(df['prediction']) #create dependent data set y

In [16]: y=y[:len(df)-prediction_days]
print(y)

Out[16]:
[8253.86625 8380.25124 8930.43  9021.07  9009.82  9220.9675  8246.81749
  9514.41375]

In [17]: from sklearn.model_selection import train_test_split # split data into 80% training and 20% testing
x_train, x_test, y_train, y_test = train_test_split(X,y, test_size=0.2)
```

**Fig 6.1 Importing libraries**



The screenshot shows a Jupyter Notebook interface with the title "Bitcoin\_Price\_prediction". The notebook contains 17 code cells. The first cell imports numpy and pandas. The second cell reads a CSV file named "bitcoin\_price.csv" and displays the first 8 rows. The third cell drops the "Date" column. The fourth cell displays the first 8 rows of the resulting DataFrame. The fifth cell sets "prediction\_days" to 30. The sixth cell shifts the "Price (USD)" column by -prediction\_days. The seventh cell displays the first 8 rows. The eighth cell displays the last 8 rows. The ninth cell creates an independent data set X. The tenth cell prints X. The eleventh cell creates a dependent data set y. The twelfth cell prints y. The thirteenth cell splits the data into 80% training and 20% testing sets.

```
In [1]: import numpy as np
import pandas as pd # import libraries

In [2]: df = pd.read_csv('bitcoin_price.csv')
df.head(8)

Out[2]:
Date Price (USD)
0 4/19/2018 8253.86625
1 4/20/2018 8380.25124
2 4/21/2018 8930.43000
3 4/22/2018 9021.07000
4 4/23/2018 9009.82000
5 4/24/2018 9220.96750
6 4/25/2018 8246.81749
7 4/26/2018 9514.41375

In [3]: df.drop(['Date'],1, inplace=True) #remove date column

In [4]: df.head(8)

Out[4]:
Price (USD)
0 8253.86625
1 8380.25124
2 8930.43000
3 9021.07000
4 9009.82000
5 9220.96750
6 8246.81749
7 9514.41375

In [5]: prediction_days = 30 #variable for prediction of n days of future

In [10]: df['prediction'] = df[['Price (USD)']].shift(-prediction_days)

In [11]: df.head(8)

Out[11]:
Price (USD) prediction
0 8253.86625 8253.86625
1 8380.25124 8380.25124
2 8930.43000 8930.43000
3 9021.07000 9021.07000
4 9009.82000 9009.82000
5 9220.96750 9220.96750
6 8246.81749 8246.81749
7 9514.41375 9514.41375

In [12]: df.tail(8)

Out[12]:
Price (USD) prediction
1 8380.25124 8380.25124
2 8930.43000 8930.43000
3 9021.07000 9021.07000
4 9009.82000 9009.82000
5 9220.96750 9220.96750
6 8246.81749 8246.81749
7 9514.41375 9514.41375

In [13]: X= np.array(df.drop(['prediction'],1)) #create independent data set X

In [14]: X=X[:len(df)-prediction_days]
print(X)

Out[14]:
[[8253.86625 8380.25124 8930.43  9021.07  9009.82  9220.9675  8246.81749
  9514.41375]]

In [15]: y= np.array(df['prediction']) #create dependent data set y

In [16]: y=y[:len(df)-prediction_days]
print(y)

Out[16]:
[8253.86625 8380.25124 8930.43  9021.07  9009.82  9220.9675  8246.81749
  9514.41375]

In [17]: from sklearn.model_selection import train_test_split # split data into 80% training and 20% testing
x_train, x_test, y_train, y_test = train_test_split(X,y, test_size=0.2)
```

**Fig 6.2 Performing Operation**

```

In [9]: X = np.array(df.drop(['prediction'],1)) #create independent data set X

In [10]: X = X[:(df['prediction_days']-1)]
print(X)

[[ 8253.88625 ]
 [ 8380.25124 ]
 [ 8930.43 ]
 [ 9021.07 ]
 [ 9009.82 ]
 [ 9220.9675 ]
 [ 9746.81749 ]
 [ 9514.41375 ]
 [ 9303.01874 ]
 [ 9372.38375 ]
 [ 9536.64 ]
 [ 9472.4975 ]
 [ 9344.89125 ]
 [ 9134.9525 ]
 [ 9262.32625 ]
 [ 9790.94249 ]
 [ 9859.62875 ]
 [ 9954.95375 ]
 [ 9869.65 ]
 [ 9888.88888 ]

In [15]: y = np.array(df['prediction']) #create dependent data set y

In [16]: y = y[:(df['prediction_days']-1)]
print(y)

...

In [17]: from sklearn.model_selection import train_test_split # split data into 80% training and 20% testing
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

In [18]: prediction_days_array = np.array(df.drop(['prediction'],1))[-prediction_days:]

In [21]: from sklearn.svm import SVR

```

**Fig 6.3 Taking variables**

```

In [13]: from sklearn.model_selection import train_test_split # split data into 80% training and 20% testing
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

In [14]: prediction_days_array = np.array(df.drop(['prediction'],1))[-prediction_days:]

In [*]: from sklearn.svm import SVR

In [22]: svr_rbf = SVR(kernel='rbf', C=1e3, gamma=0.00001) #create and train Support vector Machine

In [23]: svr_rbf.fit(x_train, y_train)

...

In [24]: svr_rbf_confidence = svr_rbf.score(x_test, y_test)
print("svr_rbf accuracy: ", svr_rbf_confidence) #testing accuracy of model # the model shows 62% accuracy

...

In [25]: svm_prediction = svr_rbf.predict(x_test)
print(svm_prediction) #print prediction values

...

In [26]: print(y_test)

...

In [27]: svm_prediction = svr_rbf.predict(prediction_days_array)
print(svm_prediction) #print the bitcoin price prediction

...

In [ ]:

In [ ]:

In [ ]:

```

**Fig 6.4 Importing SVM**

```

In [13]: from sklearn.model_selection import train_test_split # split data into 80% training and 20% testing
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

In [14]: prediction_days_array = np.array(df.drop(['prediction'],1))[-prediction_days:]

In [15]: from sklearn.svm import SVR

In [16]: svr_rbf = SVR(kernel='rbf', C=1e3, gamma=0.00001) #create and train Support vector Machine

In [17]: svr_rbf.fit(x_train, y_train)

Out[17]: SVR(C=1000.0, cache_size=200, coef0=0.0, degree=3, epsilon=0.1, gamma=1e-05,
kernel='rbf', max_iter=1, shrinking=True, tol=0.001, verbose=False)

In [18]: svr_rbf_confidence = svr_rbf.score(x_test, y_test)
print("svr_rbf accuracy: ", svr_rbf_confidence) #testing accuracy of model # the model shows 62% accuracy
svr_rbf accuracy: 0.6524554001875631

In [19]: svm_prediction = svr_rbf.predict(x_test)
print(svm_prediction) #print prediction values
[ 3851.74271928  8215.3245219  6314.26709133  6594.23212526
  6634.6436668  10252.09206775  8041.04612915  8435.73577355
  9133.81405866  8014.51773973  6447.07965124  9143.27199274
  6342.18007917  3853.2391926  4599.71153648  8270.50040046
  4481.67811816  8720.7590863  8421.99920234  6973.1930951
  3886.72285352  8296.34728944  6434.18428802  6290.924239
  8635.014277  3866.44428037  8234.10032173  8849.00603027
  6471.51759161  6222.18304668  4195.8348054  6683.52606666
  3937.18008175  7788.85941906  6396.9134772  9983.03759811
  10415.83877441  8264.55126147  8290.21409022  6798.68217196
  6469.23975726  6486.36049022  8282.80552948  10501.527912
  6321.07209197  7801.59175428  7567.91430081  4392.6382783
  3852.12828776  8919.39710968  9356.33262835  4488.74684565
  7545.56547293  8541.9017956  8952.42107604  8699.40712446]

```

**Fig 6.5 Predicting x variables**

```

In [21]: svm_prediction = svr_rbf.predict(prediction_days_array)
print(svm_prediction) #print the bitcoin price prediction
[7551.24236283  8806.05254923  6965.21404655  6688.96402331  6826.28751927
  6855.41740884  7602.53974696  7554.673365  8418.0101107  7638.65630534
  8062.17933011  8184.28350406  7667.12067111  8435.28437521  7941.52583769
  7532.09850725  8044.96238739  8035.20584931  8046.65746805  8157.03473712
  7790.96920631  8389.52975258  7553.13638495  8065.70638543  8215.6697241
  7961.69013944  8162.55638994  8075.05074841  8128.16207586  8236.58216786]

```

**Fig 6.6 Comparing with y variable**

The screenshot shows a Jupyter Notebook titled "Bitcoin\_Price\_prediction" running on a local host. The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations, running code, and viewing output. The main area displays a large array of numerical data, likely representing Bitcoin prices over time. Below the data, there is a code cell with the following content:

```
In [21]: svm_prediction = svm_rbf.predict(prediction_days_array)
print(svm_prediction) #print the bitcoin price prediction

[7551.24236283 8806.05254923 6965.21404655 6688.96402331 6826.28751927
 6855.41740804 7602.53974696 7554.6733365 8418.0101107 7638.65630534
 8062.17933011 8184.28339406 7667.12067311 8435.28437521 7941.52581769
 7531.09510725 8044.86238739 8035.20564931 8046.65746805 8157.03477312
 7770.96920631 8389.52975258 7553.13638495 8065.70638543 8215.6697241
 7961.69013944 8162.55638994 8075.05874841 8128.18207586 8236.58216786]
```

Below the code cell, there are three empty input fields for the next steps in the notebook.

**Fig 6.7 Result**

## Plagiarism report

| Project Paper Presentation for plg |                                                                                                                                                                                                              |              |                |
|------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|----------------|
| ORIGINALITY REPORT                 |                                                                                                                                                                                                              |              |                |
| <b>8%</b>                          | <b>6%</b>                                                                                                                                                                                                    | <b>7%</b>    | <b>2%</b>      |
| SIMILARITY INDEX                   | INTERNET SOURCES                                                                                                                                                                                             | PUBLICATIONS | STUDENT PAPERS |
| PRIMARY SOURCES                    |                                                                                                                                                                                                              |              |                |
| <b>1</b>                           | <b>www.mdpi.com</b><br>Internet Source                                                                                                                                                                       |              | <b>2%</b>      |
| <b>2</b>                           | <b>iopscience.iop.org</b><br>Internet Source                                                                                                                                                                 |              | <b>1%</b>      |
| <b>3</b>                           | <b>www.ncbi.nlm.nih.gov</b><br>Internet Source                                                                                                                                                               |              | <b>1%</b>      |
| <b>4</b>                           | <b>www.tandfonline.com</b><br>Internet Source                                                                                                                                                                |              | <b>1%</b>      |
| <b>5</b>                           | <b>Suhwan Ji, Jongmin Kim, Hyeonseung Im. "A Comparative Study of Bitcoin Price Prediction Using Deep Learning", Mathematics, 2019</b><br>Publication                                                        |              | <b>1%</b>      |
| <b>6</b>                           | <b>"International Conference on Intelligent Data Communication Technologies and Internet of Things (ICICI) 2018", Springer Science and Business Media LLC, 2019</b><br>Publication                           |              | <b>1%</b>      |
| <b>7</b>                           | <b>export.arxiv.org</b><br>Internet Source                                                                                                                                                                   |              | <b>1%</b>      |
| <b>8</b>                           | <b>www.ijitee.org</b><br>Internet Source                                                                                                                                                                     |              | <b>&lt;1%</b>  |
| <b>9</b>                           | <b>Siddhi Velankar, Sakshi Valecha, Shreya Maji. "Bitcoin price prediction using machine learning", 2018 20th International Conference on Advanced Communication Technology (ICACT), 2018</b><br>Publication |              | <b>&lt;1%</b>  |