Predicting and Enhancing the Fluctuations of Cryptocurrency Value by Using Machine Learning Algorithms

Saurav Singh , Bala Prabhu Anush Reddy Katakam, Siddhesh Shelke, Akula Phanidhar, Bharath Sai Lingampalli, Sahana V Kallur, Praveen Dommalapati , K. Naga Venkata Maha Lakshmi, Parmati Preetham

Abstract

Neural network-based machine learning has applications in a wide range of industries, including translation, finance, distribution, the medical industry, and cognition. This paper demonstrates a recurrent neural network learning model based on LSTM that forecasts future Bitcoin prices by analysing past prices. By learning the previous prices for 30 days, this model predicts the next day's price and shows the real and anticipated Bitcoin prices for 81 days. The training data set is divided at a rate of 1:9 from the regularised data set for modelling. The latter set is once more divided into verification and training data.

This study's machine learning requires the use of the Keras framework and Neural Network library. Using the training data, one must optimise the procedure in order to get the model's weight. In this study, batch size for the fit function is 11 and epochs are 30. The loss declines more monotonously and eventually converges to a more regular value when learning is processed more frequently. In other words, it denotes that there is no overfitting. As a result of the experiment, machine learning not only suggests that weights converge towards a specific one after examining graphs of error rates and weight change rates, but also that as learning progresses, the neural network's processing effectiveness improves.

1. Introduction

• A peer-to-peer cryptocurrency technology that runs decentralised and bank-free. The source is open-source. Although nobody in the world has control over cryptocurrencies, everyone can engage in trading.

• Currently, there are more than $230 billion worth of publicly traded cryptocurrencies on the market.

• By purchasing cryptocurrencies at a discount and then selling them at a profit when their value rises, investors can benefit.

• The value of cryptocurrencies fluctuates exactly like the stock market.

• As a result, if we can foresee the ups and downs of bitcoin value, we may quickly develop and make money.

Therefore, the purpose of my work in this paper is to use machine learning techniques to discover how the value of cryptocurrencies changes on a daily basis.

1.1 Background

The enormous amount of data is categorised using machine learning algorithms into little pieces. We try to arrange the tweet's extreme where
Either way, it is certain or unfavourable. If the tweet has any potential of
both positive and negative aspects, with the former being more pronounced
choosing emotion as the last name was an estimation. Different machines
The features from the data can be extracted using learning techniques.

1.2 Motivation

David Chaum, a pioneer in cryptologic design, created the Blind Signature technology, which transmits encoded signals securing digital signatures and led to the creation of E-cash. That is the main cryptocurrency used for commerce. The first cryptocurrency developed using block chain technology was Bit Coin in 2009. Following that, block chain technology has enhanced the majority of cryptocurrencies. In 2015, Ethereum emerged as a mature currency with services and applications in addition to the blockchain.

1.3 Challenge

The following are the challenges of this Paper:

- Developing a Scientifically sound with cost effective machine learning Paper for predicting raise and fall of Cryptocurrency.

- Predicting the updating the market of Cryptocurrency to the users frequently.
- Promoting our services and product online.
- Providing product support to clients.

- Make sure of the data given to users is accurate and useful.

- Getting feedbacks from users and clients frequently.
- Updating the Paper keeping the feedback of users and clients

2 .Planning and Requirements Specification

2.1. System Planning

Literature Review

Bitcoin is a new technology hence currently there are few price prediction models available. 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 produces three linear models. These three models are linearly combined to predict the price of Bitcoin.

According to the author is analysing 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.

In , instead of directly forecasting the future price of the stock, the authors predict 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, sell or hold the stock.

Feasibility study

The feasibility of the Paper is analyzed in this phase and business proposal is put forth with a very general plan for the Paper and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- SOCIAL FEASIBILITY

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

2.2 Requirements

2.2.1 User Requirements

- tweets picked up from Twitter API provided by Twitter itself. Due to the presence of Twitter API, there are many techniques available for sentimental analysis of data on Social media. In this Paper a set of available libraries has been used. The approach to extract sentiment from tweets.

- 1.Starting with downloading the sentimental dictionary

- 2. Then download the twitter testing data sets and add them as an input to the program.

2.2.2 Non-functional requirements

In systems engineering and requirements engineering, a non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. They are contrasted with functional requirements that define specific behavior or functions. **Non-functional requirements** add tremendous value to business analysis. It is commonly misunderstood by a lot of people. It is important for business stakeholders, and Clients to clearly explain the requirements and their expectations in measurable terms. If the non-functional requirements are not measurable then they should be revised or rewritten to gain better clarity. For example, User stories help in mitigating the gap between developers and the user community in Agile Methodology.

Usability:
Prioritize the important functions of the system based on usage patterns.

Frequently used functions should be tested for usability, as should complex and critical functions. Be sure to create a requirement for this.

Reliability:

Reliability defines the trust in the system that is developed after using it for a period of time. It defines the likeability of the software to work without failure for a given time period.

The number of bugs in the code, hardware failures, and problems can reduce the reliability of the software.

Your goal should be a long MTBF (mean time between failures). It is defined as the average period of time the system runs before failing.

Create a requirement that data created in the system will be retained for a number of years without the data being changed by the system.

It's a good idea to also include requirements that make it easier to monitor system performance.

Performance:
What should system response times be, as measured from any point, under what circumstances?

Are there specific peak times when the load on the system will be unusually high?

Think of stress periods, for example, at the end of the month or in conjunction with payroll disbursement.

Supportability:
The system needs to be **cost-effective to maintain**.

Maintainability requirements may cover diverse levels of documentation, such as system documentation, as well as test documentation, e.g. which test cases and test plans will accompany the system.

2.3 System Requirements

2.3.1 Hardware Requirements:

- Processor - Pentium –IV
- RAM - 1 GB (min)
- Hard Disk - 20 GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - SVGA

2.3.2 Software Requirements:

- Operating System - Microsoft Windows
- Coding Language - Python
- Platform - Python 3.5
- Database server - MySQL

3. System Design

Systems design is the process or art of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. One could see it as the application of systems theory to product development. There is some overlap and synergy with the disciplines of systems analysis, systems architecture and systems engineering.

Figure 1 : Architecture diagram:

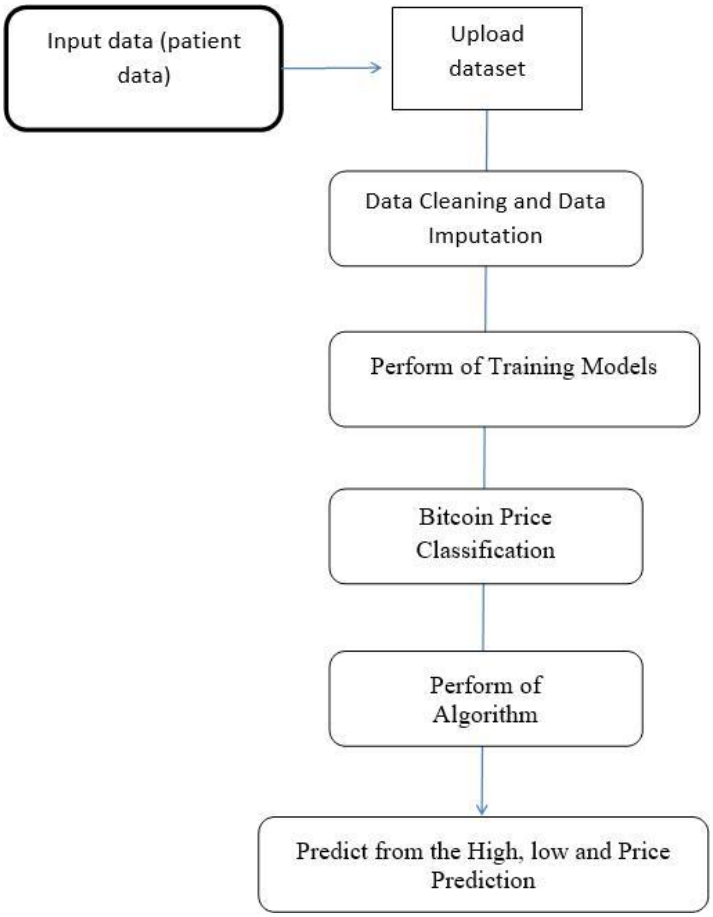


Figure 2: Activity Diagram:

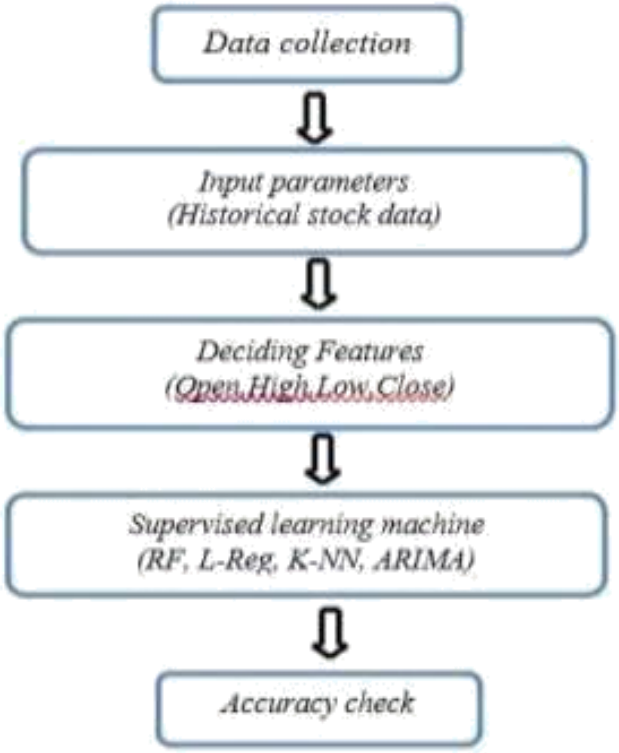


Fig-2: Flow Chart of Algorithm

Figure 3:

Sequence Diagram:

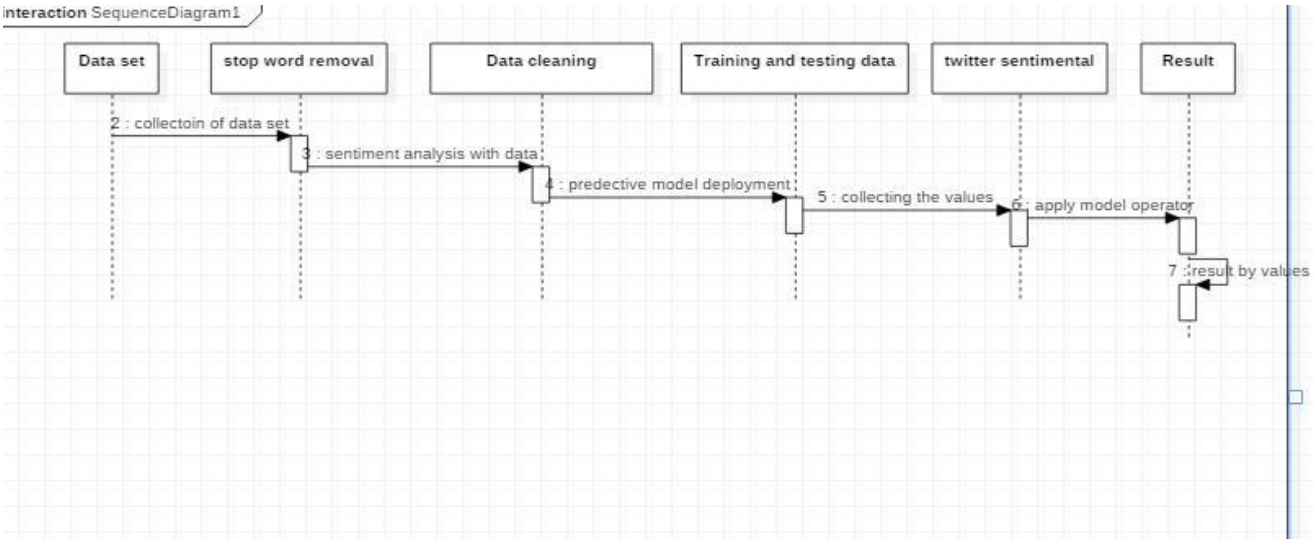


Figure 3:

Use case diagram:

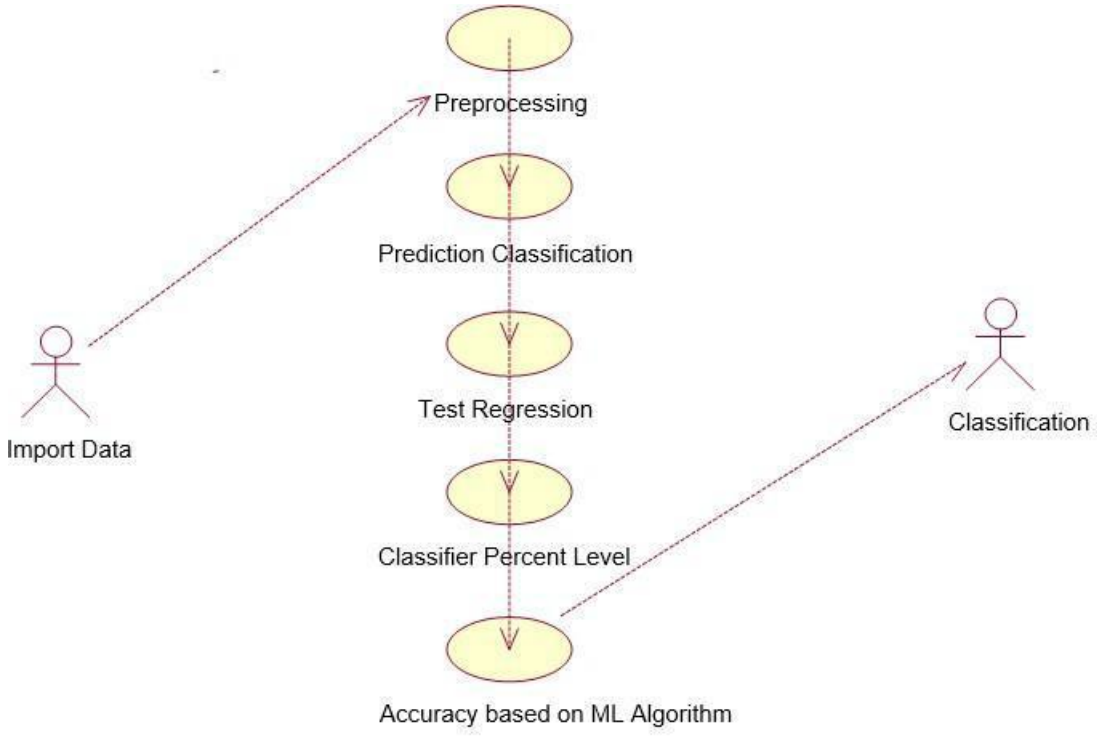
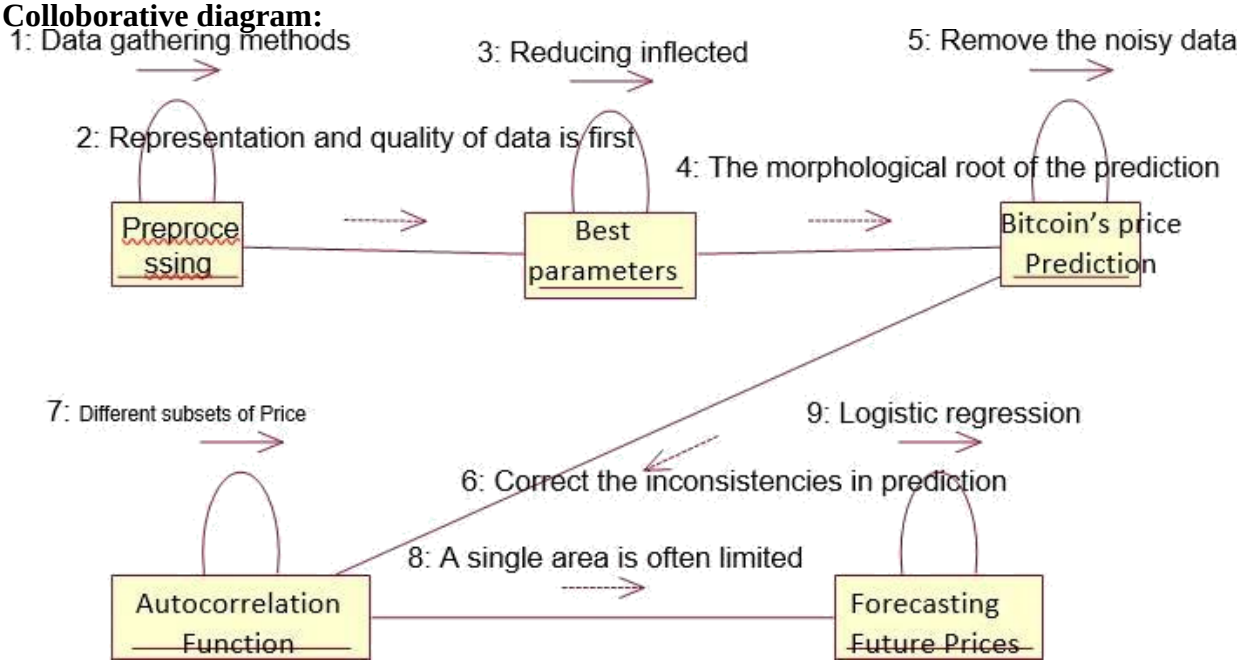


Figure 4:

Colloborative diagram:



4. Implementation of System

EXISTING SYSTEM:

Block chain technology is a system that stores online transactional information on the block, which must be approved to be connected with an existing chain. It means the parties to a transaction exchange value with one another. Bitcoin based on Blockchain skill was invented by Nakamoto Satoshi in 2009. It realized the idea of Bit Gold and B-money and made up for faults like double expenditure. Bitcoin is saved as a type of a wallet file which is given its own address, and the transactions of Bitcoin is accomplished on the basis of the address. Block is a bundle of transactional information of Bitcoin every 10 minute. That is, Blockchain is a kind of trading book which covers trading record. The graph shows market price of Bitcoin from 2009 to 2019.

DISADVANTEGS:

- Machine Learning, a method solving a problem by using methods like clustering, classification, forecasting, and so on, ought to be defined as a learning model in advance which the human beings' brains process problems.

- Recurrent Neural Network generates following data prediction through learning the context, that is, the relationship among data in estimation problem dealing with such time series data as monthly sales, price index, unemployment rare, exchange rate, and stock price

PROPOSED SYSTEM:

There is memory cell in the center and there are 3 units which fix the prices of input, output, and forget gate and 2 units which receive external inputs around the cell. Memory realizes its memory by returning to itself after particular time from the present state. The external input of memory is received by*gate g* and its output is recorded in memory cell. The external output of memory goes out of *gate h*. This structure which overcomes the limit of RNN processing short term memory can have more precise prediction if opening and closing the gates can be controlled appropriately, which extracts the information of longer context. A lot of studies have recently proposed that RNN performs well in forecasting time series financial data. They predict stock markets by implementing RNN which uses embedded memory for dynamic system application. As a result, forecasts on Bit coin prices are getting visible.

ADVANTAGES OF PROPOSED SYSTEM:

- Machine learning based on Neural Network has integrated usages in a variety of fields such as translation, finance, distribution, and medical world as well as cognition.

- This study shows Recurrent Neural Network Learning Model on the basis of LSTM, which analyzes the previous prices of a cryptocurrency, Bitcoin and predicts the next one.

5.Results And Discussion

TESTING AND DEBUGGING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal

code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

1. Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.

- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.
-

2. Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

3. Acceptance Testing

User Acceptance Testing is a critical phase of any Paper and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

In this Paper, a system is developed to give the analysis to the user for women safety and security in Indian cities. When we execute this program, we will get various outputs in different times which will be helpful for women who are in danger. If user executes program for multiple times and results are the normal of the back-to-back yields. On the off chance that the impartial message on twitter is fundamentally very high, implies that individuals have a lower enthusiasm for the theme and are not willing to have both good or bad side on it. The final results are based on the data given on twitter which will bring change in the individuals because human mind will always change according to the situations. So, this system can give best analysis and gives the update based on data.

6. Conclusion and Future WORK

Conclusion

Machine learning algorithms such as supervised learning methods are very useful in solving real world problem. In this research, the main goal is to ascertain how the direction and accuracy of price of Bitcoin can be predicted by using data mining method. The dataset related to ten cryptocurrencies are selected to form a new dataset which is utilized to achieve the goal. The nine cryptocurrencies show strong correlation with Bitcoin as shown in the figure.

Future work

The research will aim at developing the best forecasting cryptocurrency program from applying various experiments of Artificial Neural Network learning models, adding and technically analyzing data from much more cryptocurrency exchanges.

There are many ways to improve the existing methodology by applying different classification methods of Deep learning. The data related to Bitcoin play an important role. Using data from different exchanges, like Blockchain, and average of closing price collected from different exchanges can provide a different insight in the model used. Unsupervisedmethods can provide a different perspective about the data, and open new avenues to explore in this area. In addition, the author from his point of view recommended that one should gain knowledge regarding the technology that is behind Bitcoin for further investigation.

References:

[1] Archana C, Savita K, Raj K. Machine Learning Classification Techniques: A Comparative Study.International Journal on Advanced Computer Theory and Engineering (IJACTE) (2013);2(4):21-25.

[2] Pavlyshenko BM.Machine-Learning Models for Sales Time Series Forecasting. Data.(2019) Mar; 4(1):15. DOI:10.3390/data4010015.

[3] Nakamoto S. Bitcoin: A Peer-to-Peer Electronic Cash System. Available from: https://bitcoin.org/bitcoin.pdf

[4] BLOCKCHAIN.[Internet]. Available from:

https://www.blockchain.com/en/charts/market-price?timespan=all

[5] Kim JS.A Survey of Cryptocurrencies based on Blockchain: Journal of the Korea Society of Computer and Information.(2019)Feb;24(2):67-74. DOI: http://dx.doi.org/10.9708/jksci.2019.24.02.067

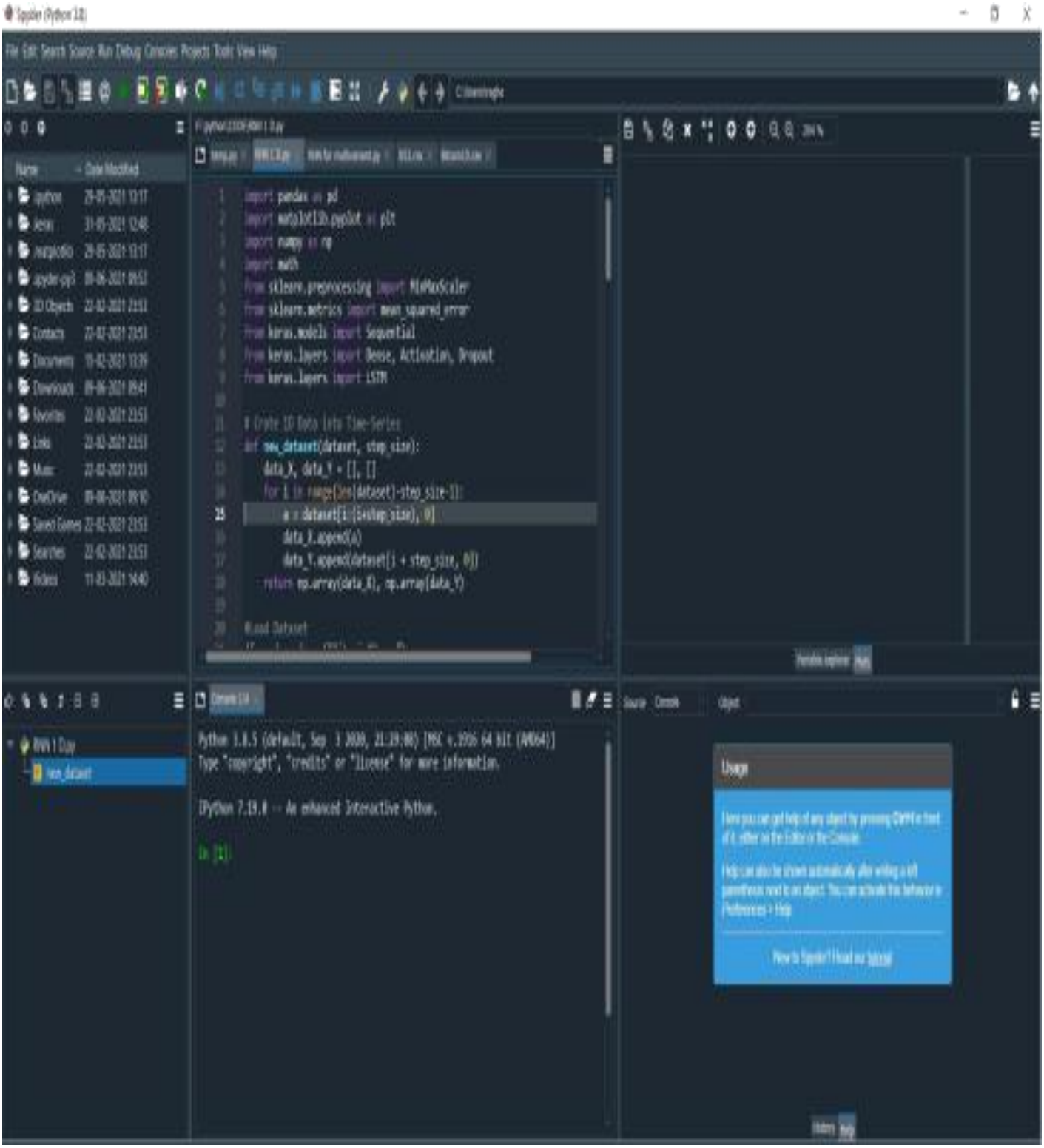
[6] Cann O. Artificial Intelligence & Robotics Top List of Technologies in Need of Better Governance. [Internet]. Available from:https://www.weforum.org/press/ (2016)/11/artificial-intelligence-robotics-top-list-of-technologies-in-need-of-better-governance/

[7] Lannquist A. [Internet]. Available from:https://www.weforum.org/agenda/ (2019)/04/this-new-form-of-currency-could-transform-the-way-we-see-money/

[8] [Internet]. Available from: https://www.pluralsight.com/guides/blockchain-architecture

[9] Gola Y. 3 Reasons Bitcoin Price Will Sprint Beyond (2019)-High $6,000. [Internet]. Available from: https://www.ccn.com/3-reasons-bitcoin-price-will-sprint-beyond-2019-high-6000/

[10] Goh B, John A. China wants to ban Bitcoin mining. [Internet]. Available from: https://www.reuters.com/article/us-china-cryptocurrency/china-wants-to-ban-bitcoin-mining.


import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import math
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error
from keras.models import Sequential
from keras.layers import Dense, Activation, Dropout
from keras.layers import LSTM
# Create 3D Data into Time-Series
def new_dataset(dataset, step_size):
data_X, data_Y = [], []
for i in range(len(dataset)-step_size-1):
a = dataset[i:(i+step_size), 0]
data_X.append(a)
data_Y.append(dataset[i + step_size, 0])
return np.array(data_X), np.array(data_Y)
Load Dataset
Python 3.8.5 (default, Sep 3 2020, 21:29:08) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.
IPython 7.19.0 -- An enhanced Interactive Python.
In [1]:
Usage

F:\python\CODE\RNN for multivariant.py

temp.py | RNN 1 D.py | RNN for multivariant.py | DS2.csv | Bitcoin1D.csv

```
1  import pandas as pd
2  from pandas import DataFrame
3  from pandas import concat
4  from math import sqrt
5  from numpy import concatenate
6  from sklearn.metrics import mean_squared_error
7  import matplotlib.pyplot as pyplot
8  import numpy as np
9  from sklearn.preprocessing import MinMaxScaler
10 from keras import Sequential
11 from keras.layers import LSTM, Dense, Dropout, Activation
12 from pandas import read_csv
13
14 # Load dataset by using Pandas library
15 dataset = read_csv('Bitcoin3D.csv', header=0, index_col=0)
16 print(dataset.head())
17 values = dataset.values
18
19 # Here was prepared column for visualizing
20 groups = [0, 1, 2, 3, 5, 6,7,8,9]
```

Console 1/A

```
mlir_graph_optimization_pass.cc:116] None of the MLIR optimization passes are
enabled (registered 2)
672/672 - 12s - loss: 0.0116 - val_loss: 0.0759
Epoch 2/10
672/672 - 3s - loss: 0.0118 - val_loss: 0.0692
Epoch 3/10
672/672 - 3s - loss: 0.0092 - val_loss: 0.0546
Epoch 4/10
672/672 - 3s - loss: 0.0082 - val_loss: 0.0568
Epoch 5/10
672/672 - 3s - loss: 0.0069 - val_loss: 0.0318
Epoch 6/10
672/672 - 3s - loss: 0.0061 - val_loss: 0.0189
Epoch 7/10
672/672 - 3s - loss: 0.0052 - val_loss: 0.0129
Epoch 8/10
672/672 - 3s - loss: 0.0047 - val_loss: 0.0114
Epoch 9/10
```

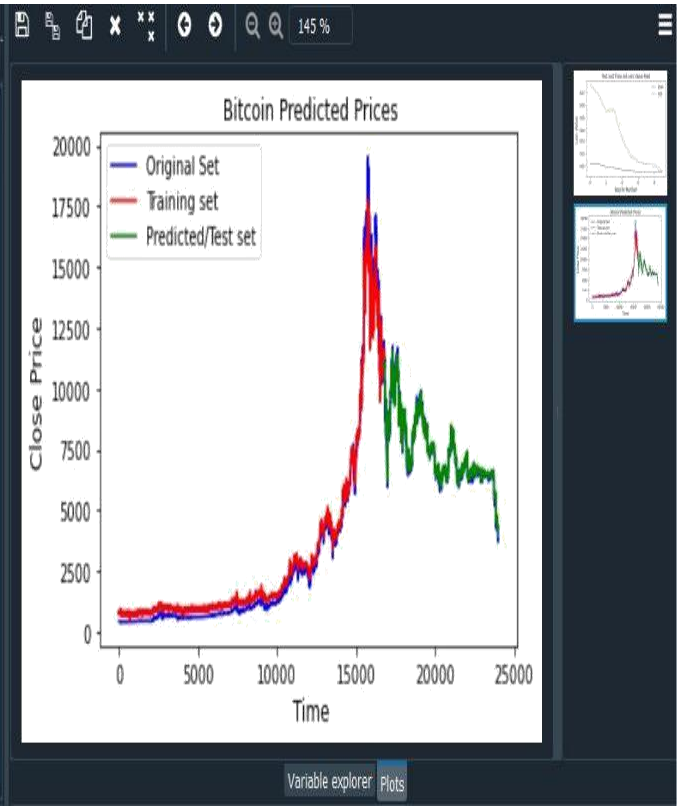
F:\python\CODE\RNN for multivariant.py

temp.py | RNN 1 D.py | RNN for multivariant.py | DS2.csv | Bitcoin1D.csv

```python
import pandas as pd
from pandas import DataFrame
from pandas import concat
from math import sqrt
from numpy import concatenate
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as pyplot
import numpy as np
from sklearn.preprocessing import MinMaxScaler
from keras import Sequential
from keras.layers import LSTM, Dense, Dropout, Activation
from pandas import read_csv

# Load dataset by using Pandas library
dataset = read_csv('Bitcoin3D.csv', header=0, index_col=0)
print(dataset.head())
values = dataset.values

# Here was prepared column for visualizing
groups = [0, 1, 2, 3, 5, 6,7,8,9]
```



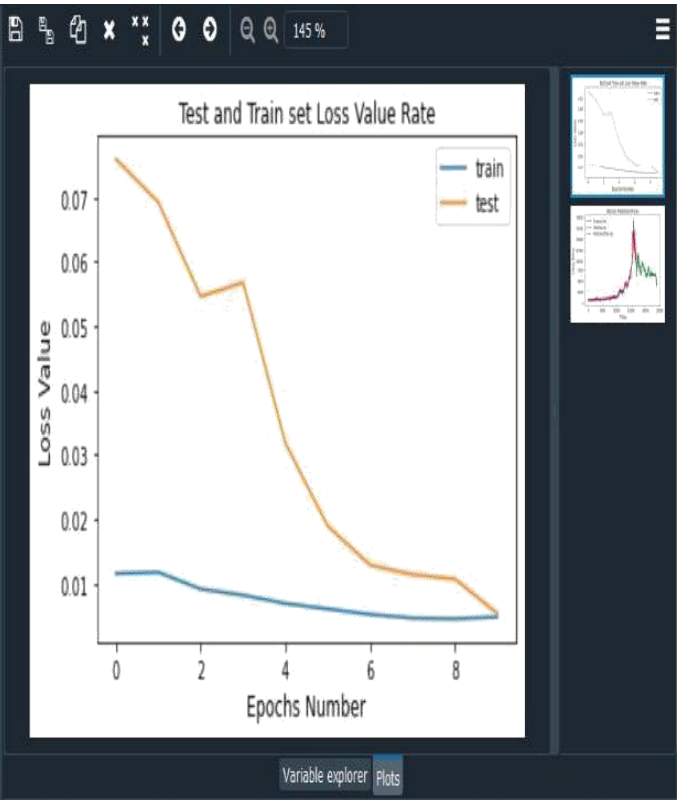
F:\python\CODE\RNN 1 D.py

temp.py | RNN 1 D.py | RNN for multivariant.py | DS2.csv | Bitcoin1D.csv

```python
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import math
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error
from keras.models import Sequential
from keras.layers import Dense, Activation, Dropout
from keras.layers import LSTM

# Crate 1D Data into Time-Series
def new_dataset(dataset, step_size):
    data_X, data_Y = [], []
    for i in range(len(dataset)-step_size-1):
        a = dataset[i:(i+step_size), 0]
        data_X.append(a)
        data_Y.append(dataset[i + step_size, 0])
    return np.array(data_X), np.array(data_Y)

#Load Dataset
```



F:\python\CODE\RNN 1 D.py

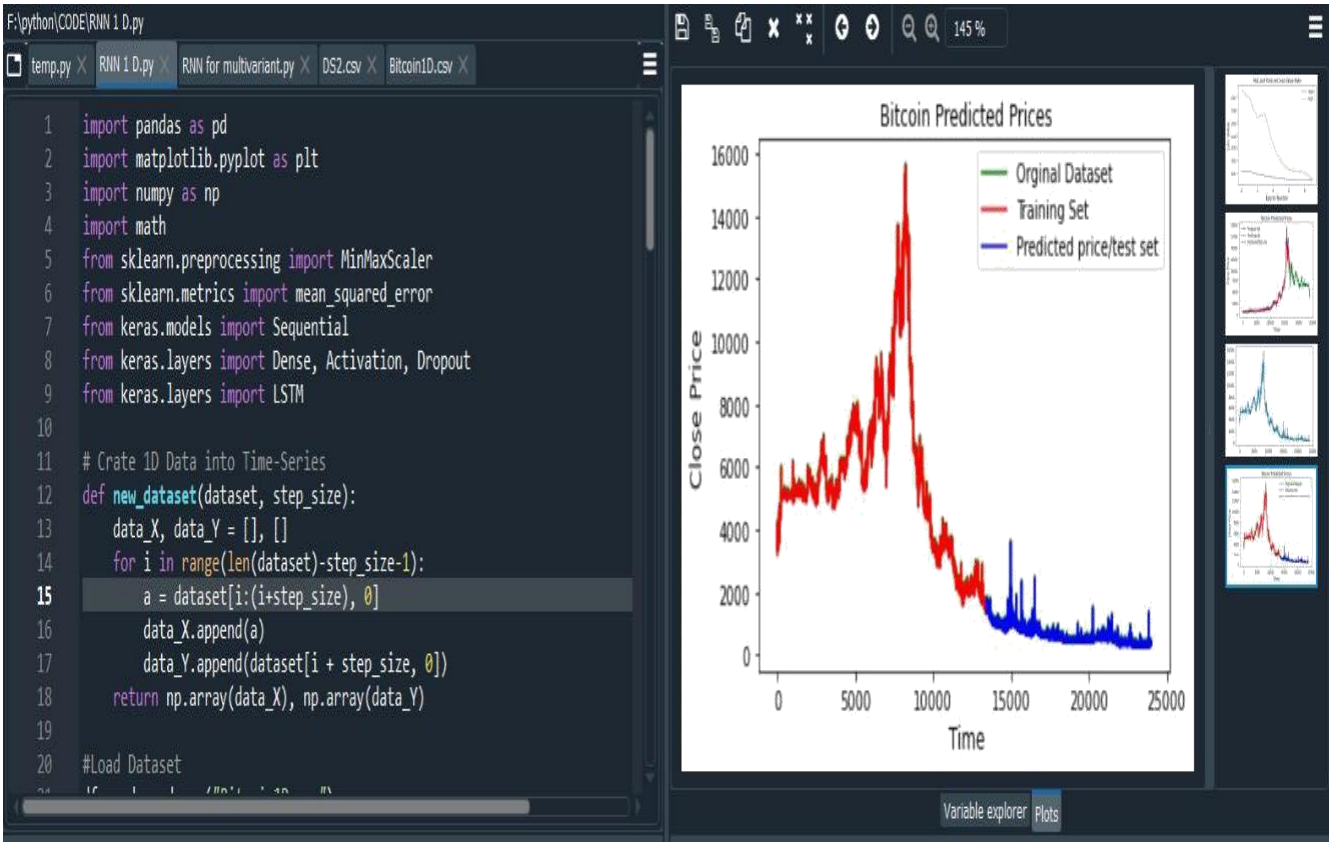
temp.py | RNN 1 D.py | RNN for multivariant.py | DS2.csv | Bitcoin1D.csv

```
1   import pandas as pd
2   import matplotlib.pyplot as plt
3   import numpy as np
4   import math
5   from sklearn.preprocessing import MinMaxScaler
6   from sklearn.metrics import mean_squared_error
7   from keras.models import Sequential
8   from keras.layers import Dense, Activation, Dropout
9   from keras.layers import LSTM
10
11  # Crate 1D Data into Time-Series
12  def new_dataset(dataset, step_size):
13      data_X, data_Y = [], []
14      for i in range(len(dataset)-step_size-1):
15          a = dataset[i:(i+step_size), 0]
16          data_X.append(a)
17          data_Y.append(dataset[i + step_size, 0])
18      return np.array(data_X), np.array(data_Y)
19
20  #Load Dataset
```

Console 1/A

```
Epoch 4/10
537/537 - 2s - loss: 2.1449e-04
Epoch 5/10
537/537 - 2s - loss: 2.0002e-04
Epoch 6/10
537/537 - 2s - loss: 1.7818e-04
Epoch 7/10
537/537 - 2s - loss: 1.6894e-04
Epoch 8/10
537/537 - 2s - loss: 1.5277e-04
Epoch 9/10
537/537 - 2s - loss: 1.4615e-04
Epoch 10/10
537/537 - 2s - loss: 1.3899e-04
Train predicted value: 127.00
Test predicted value: 97.50

In [4]:
```


F:\python\CODE\RNN 1 D.py
temp.py
RNN 1 D.py
RNN for multivariant.py
DS2.csv
Bitcoin1D.csv
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import math
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error
from keras.models import Sequential
from keras.layers import Dense, Activation, Dropout
from keras.layers import LSTM
# Crate 1D Data into Time-Series
def new_dataset(dataset, step_size):
data_X, data_Y = [], []
for i in range(len(dataset)-step_size-1):
a = dataset[i:(i+step_size), 0]
data_X.append(a)
data_Y.append(dataset[i + step_size, 0])
return np.array(data_X), np.array(data_Y)
#Load Dataset
145 %
Bitcoin Predicted Prices
Orginal Dataset
Training Set
Predicted price/test set
Close Price
Time
16000
14000
12000
10000
8000
6000
4000
2000
0
0
5000
10000
15000
20000
25000
Variable explorer
Plots