A MAJOR PROJECT REPORT ON

PREDICTING BITCOIN PRICES USING DEEP LEARNING

SUBMITTED IN PARTIAL FULFILLMENT FOR THE AWARD OF DEGREE OF

BACHELOR OF TECHNOLOGY IN ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted By:

ESHA MAHENDRA (9916102024)

HARSHITA MADAN (9916102069)

SHIVANGI GUPTA (9916102126)

Under the Guidance Of

Dr. Sajaivir Singh

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING, JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA (U.P.) Dec, 2019

CERTIFICATE

This is to certify that the major project report entitled, "PREDICTING BITCOIN PRICES USING DEEP LEARNING" submitted by Esha Mahendra, Shivangi Gupta, Harshita Madan in partial fulfillment of the requirements for the award of Bachelor of Technology Degree in **Electronics and Communication Engineering** of the Jaypee Institute of Information Technology, Noida is an authentic work carried out by them under my supervision and guidance. The matter embodied in this report is original and has not been submitted for the award of any other degree.

Signature of Supervisor:
Name of the Supervisor: Dr. Sajaivir Singh
ECE Department, JIIT, Sec-128, Noida-201304
Dated:

DECLARATION

We hereby declare that this written submission represents our own ideas in our own words and where others' ideas or words have been included, have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission.

Place: Noida
Date:

Name: Esha Mahendra
Enrollment: 9916102024

Name: Harshita Madan
Enrollment: 9916102069

Name: Shivangi Gupta
Enrollment: 9916102126

ABSTRACT

The rapid development of digital currencies during the last decade is one of the most controversial and ambiguous innovations in the modern global economy. Significant fluctuations in the exchange rate of cryptocurrencies and their high volatility, as well as the lack of legal regulation of their transactions in most countries resulted in significant risks associated with investment into crypto assets. This has led to heated discussions about their place and role in the modern economy. Therefore, the issue of developing appropriate methods and models for predicting prices for bitcoin is relevant both for the scientific community and for financial analysts, investors and traders. Methodological approaches to forecasting prices for financial assets depend on an analyst's understanding of the causal relationships in the pricing process.

ACKNOWLEDGEMENT

It is our privilege to express our sincerest regards to our project coordinator, Dr.Sajaivir Singh for his valuable inputs, guidance, encouragement, whole-hearted cooperation and constructive criticism throughout the duration of our project.

We deeply express our sincere thanks to our Head of Department for encouraging and allowing us to present the project on the topic "PREDICTING BITCOIN PRICES USING DEEP LEARNING" at our department premises for the partial fulfillment of the requirements leading to the award of B-Tech degree. We take this opportunity to thank all our lecturers who have directly or indirectly helped our project.

We pay our respects and love to our parents and all other family members and friends for their love and encouragement throughout our career. Last but not the least we express our thanks to our friends for their cooperation and support.

Signature:

Names:

Esha Mahendra (9916102024)
Harshita Madan (9916102069)
Shivangi Gupta (9916102126)

Table of Contents

Contents	**Page No.**
Abstract	*i*
Acknowledgement	*ii*
Table of Contents	*iii*
List of Figures	*iv*
CHAPTER 1: INTRODUCTION	**1**
1.1 Introduction to Deep learning	1
1.2 Introduction to Cryptocurrency and Bitcoin	2
1.3 Project Goal	3
CHAPTER 2: LITERATURE SURVEY	**4**
2.1 Forecasting cryptocurrency prices time series using machine learning approach	4
2.2 Predictive Analysis of Cryptocurrency Price Using Deep Learning	4
2.3 Predicting Bitcoin Prices using Deep Learning	4
CHAPTER 3: ALGORITHMS USED	**6**
3.1 Long Short Term Memory (LSTM)	6
3.2 Gated Recurrent unit (GRU)	8
CHAPTER 4: DETAILED EXPLANATION	**11**
4.1 Softwares and libraries used	11
4.2 Work Flow	12
CHAPTER 5: RESULTS	**14**
CHAPTER 6: CONCLUSION AND FUTURE SCOPE	**17**
References	
Appendices	

LIST OF FIGURES

S. No.	Title	Page No.
1.	Difference between Machine Learning and Deep Learning	1
2.	RNN Algorithm	6
3.	LSTM Algorithm	8
4.	Comparison between LSTM and GRU Algorithm	10
5.	Work Flow	12
6.	Dataset	13
7.	LSTM Model	14
8.	GRU Model	14
9.	Prediction using LSTM	15
10.	Prediction using GRU	15

CHAPTER 1: INTRODUCTION

1.1 Introduction to Deep Learning

Deep learning is part of a broader family of machine learning methods based on artificial neural networks.[Fig[1]] Learning can be supervised, semi-supervised or unsupervised.

Deep learning architectures such as deep neural networks, deep belief networks, recurrent neural networks and convolutional neural networks have been applied to fields including computer vision, speech recognition, natural language processing, audio recognition, social network filtering, machine translation, bioinformatics, drug design, medical image analysis, material inspection and board game programs, where they have produced results comparable to and in some cases superior to human experts.

Deep learning is a class of machine learning algorithms that uses multiple layers to progressively extract higher level features from the raw input. For example, in image processing, lower layers may identify edges, while higher layers may identify the concepts relevant to a human such as digits or letters or faces.[1]

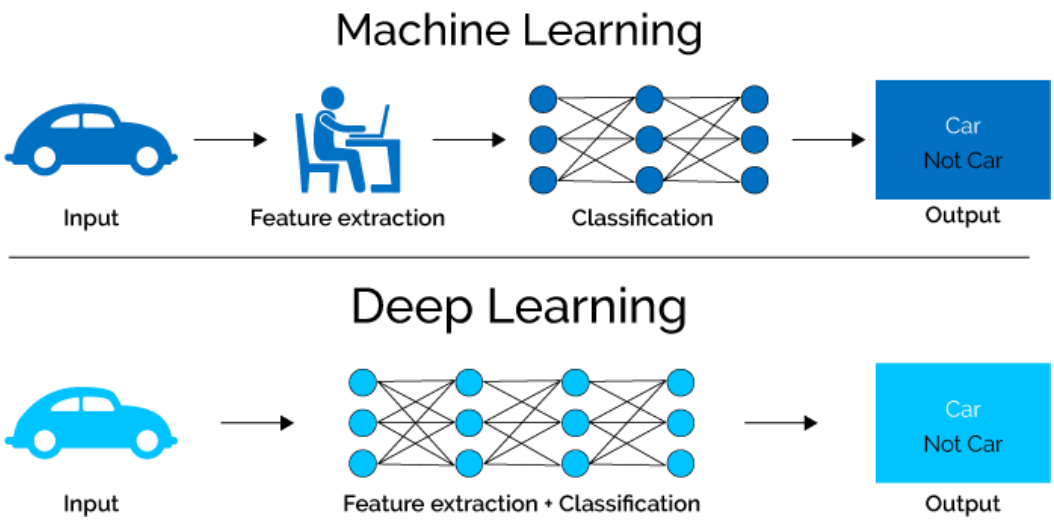

Fig. 1 Difference between Deep learning and Machine learning[1]

1.2 Introduction to Cryptocurrency and Bitcoin

Cryptocurrency is an internet-based medium of exchange which uses cryptographical functions to conduct financial transactions. Cryptocurrencies leverage blockchain technology to gain decentralization, transparency, and immutability. The most important feature of a cryptocurrency is that it is not controlled by any central authority: the decentralized nature of the blockchain makes cryptocurrencies theoretically immune to the old ways of government control and interference. Cryptocurrencies can be sent directly between two parties via the use of private and public keys. These transfers can be done with minimal processing fees, allowing users to avoid the steep fees charged by traditional financial institutions.[8]

Bitcoin is a digital payment currency that utilizes cryptocurrency and peer-to-peer (P2P) technology to create and manage monetary transactions as opposed to a central authority. The open source Bitcoin P2P network creates the bitcoins and manages all the bitcoin transactions. Often referred to as cash for the Internet, Bitcoin is one of several popular digital payment currencies along with Litecoin, Peercoin and Namecoin. Bitcoin is considered the biggest cryptocurrency. It was first introduced in 2009 and is the most widely-traded cryptocurrency. Bitcoin as an implementation of the cryptocurrency concept was described by Wei Dai in 1998 on the cypherpunks mailing list. Dai suggested a new form of money that uses cryptography to control its creation and transactions, rather than a central authority. In 2009, the Bitcoin specification and proof of concept was published in a cryptography mailing list by Satoshi Nakamoto. Payments are made via a Bitcoin wallet application that resides on a user's computer or mobile device, and a person only needs to enter the recipient's Bitcoin address information and payment amount before pressing send to complete payment.

New bitcoins are created by a competitive and decentralized process called mining. Bitcoin miners are processing transactions and securing the network using specialized hardware and are collecting new bitcoins in exchange. The Bitcoin protocol ensures new bitcoins are created at a fixed rate, making the process of bitcoin mining a very competitive business. While attackers are going after Bitcoin-related sites, there is an important distinction between the security of the Bitcoin network and the Bitcoin exchanges.

1.3 Project Goal

Time series forecasting or prediction is a well-known problem. Much research has been done for predicting markets such as the stock marke. Cryptocurrencies can be considered a form of virtual currency intended to serve as a medium of exchange and presents an interesting topic since it can be treated as a time series prediction problem. This problem still remains in nascent stages. Consequently, there is high volatility in the market, and this offers opportunities for further research on the prediction of cryptocurrency price. Moreover, cryptocurrencies such as the Bitcoin are increasingly adopted across the world. Because of the open nature of the cryptocurrency, it operates on a decentralized, peer-to-peer, and trustless system in which all transactions are passed to an open ledger known as the blockchain. Such transparency is unknown in the world of classical financial markets.

The soul purpose of the project to give a jest idea of how deep learning can be used to detect, predict bitcoin prices at a much faster rate and thus complexity. The current work is focused mainly in two directions: Predicting the prices of bitcoin and understanding the pattern of cryptocurrency.[1]

In this project, we will be using the technique of deep learning for prediction of bitcoin prices of the state-wise crime dataset. The bitcoin data is extracted from the official website Kaggle.com. It consists of the information like timestamp, open price, close price, high price, low price and volume. Before training of the model data preprocessing will be done following this feature selection and scaling will be done so that accuracy obtain will be high. The Long short term memory (LSTM) and Gated Recurrent Unit (GRU) algorithms will be tested for prediction of bitcoin prices and one with better accuracy will be used for training.[11]

As soon as the algorithms were processed, accuracy of different algorithms were measured & the algorithm with the most accuracy is used for the prediction. Visualization of dataset is done in terms of graphical representation is observed.

CHAPTER 2: LITERATURE SURVEY

2.1 Forecasting cryptocurrency prices time series using machine learning approach[2]

In this research, it describes the construction of the short-term forecasting model of cryptocurrencies prices using machine learning approach. The modified model of Binary Auto Regressive Tree (BART) is adapted from the standard models of regression trees and the data of the time series. BART combines the classic algorithm classification and regression trees (C&RT) and autoregressive models ARIMA. Using the BART model, they have made a short-term forecast (from 5 to 30 days) for the 3 most capitalized cryptocurrencies: Bitcoin, Ethereum and Ripple. We found that the proposed approach was more accurate than the ARIMA-ARFIMA models in forecasting cryptocurrencies time series both in the periods of slow rising (falling) and in the periods of transition dynamics (change of trend).

2.2 Predictive Analysis of Cryptocurrency Price Using Deep Learning[3]

This paper proposes a novel method to predict cryptocurrency price by considering various factors such as market cap, volume, circulating supply, and maximum supply based on deep learning techniques such as the recurrent neural network (RNN) and the long short-term memory (LSTM), which are effective learning models for training data, with the LSTM being better at recognizing longer-term associations. The proposed approach is implemented in Python and validated for benchmark datasets. The results verify the applicability of the proposed approach for the accurate prediction of cryptocurrency price.

2.3 Predicting Bitcoin Prices using Deep Learning[4]

The goal for this research is to show how a trained machine model can predict the price of a cryptocurrency if we give the right amount of data and computational power. It displays a graph with the predicted values. The most popular technology is the kind of technological solution that could help mankind predict future events. With vast amount of data being generated and recorded on a daily basis, we have finally come close to an era where predictions can be accurate and be generated based on concrete factual data. Furthermore, with the rise of the crypto digital era more heads have turned towards the digital market for investments. This gives us the opportunity to create a model capable of

predicting crypto currencies primarily Bitcoin. This can be accomplished by using a series of machine learning techniques and methodologies. Building algorithms and models to predict prices and future events has been given significant amount of attention in the past decade. With user data being collected through various forms of paths, there has never been an abundance in raw data like there is now. Any model capable of predicting a future event whether it be to find out what the next big trend is or to predict the next behavior of a anything possess a great potential in today's world.

CHAPTER 3: ALGORITHMS USED

3.1 Long Short term Memory (LSTM)

A simple machine learning model or an Artificial Neural Network may learn to predict the stock prices based on a number of features: the volume of the stock, the opening value etc. While the price of the stock depends on these features, it is also largely dependent on the stock values in the previous days. In the conventional feed-forward neural networks, all test cases are considered to be independent. That is when fitting the model for a particular day, there is no consideration for the stock prices on the previous days.[9]

This dependency on time is achieved via Recurrent Neural Networks. Fig[2] A typical RNN looks like:

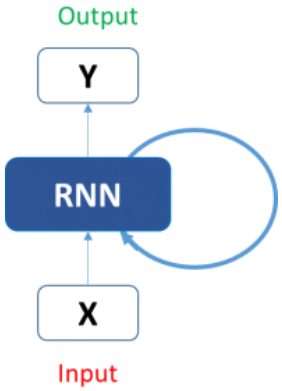

Fig 2. RNN Algorithm

Recurrent Neural Networks work just fine when we are dealing with short-term dependencies. The reason behind this is the problem of Vanishing Gradient**.** We know that for a conventional feed-forward neural network, the weight updating that is applied on a particular layer is a multiple of the learning rate, the error term from the previous layer and the input to that layer. Thus, the error term for a particular layer is somewhere a product of all previous layers' errors.

To solve the problem of Vanishing and Exploding Gradients in a deep Recurrent Neural Network, many variations were developed. One of the most famous of them is the Long Short Term Memory Network (LSTM). In concept, an LSTM recurrent unit tries to "remember" all the past knowledge that the network is seen so far and to "forget" irrelevant data. This is done by introducing different activation function layers called "gates" for different purposes. Each LSTM recurrent unit also maintains a vector called the Internal Cell State which conceptually describes the information that was chosen to be

retained by the previous LSTM recurrent unit. A Long Short Term Memory Network consists of four different gates for different purposes as described below:-

1. **Forget Gate (f):** It determines to what extent to forget the previous data.
2. **Input Gate (i):** It determines the extent of information to be written onto the Internal Cell State.
3. **Input Modulation Gate (g):** It is often considered as a sub-part of the input gate and many literatures on LSTM's do not even mention it and assume it inside the Input gate. It is used to modulate the information that the Input gate will write onto the Internal State Cell by adding non-linearity to the information and making the information Zero-mean. This is done to reduce the learning time as Zero-mean input has faster convergence. Although this gate's actions are less important than the others and is often treated as a finesse-providing concept, it is good practice to include this gate into the structure of the LSTM unit.
4. **Output Gate (o):** It determines what output(next Hidden State) to generate from the current Internal Cell State.

Working of an LSTM recurrent unit: Fig[3]

1. Take input the current input, the previous hidden state and the previous internal cell state.
2. Calculate the values of the four different gates by following the below steps:-

- For each gate, calculate the parameterized vectors for the current input and the previous hidden state by element-wise multiplication with the concerned vector with the respective weights for each gate.
- Apply the respective activation function for each gate element-wise on the parameterized vectors. Below given is the list of the gates with the activation function to be applied for the gate.

3. Calculate the current internal cell state by first calculating the element-wise multiplication vector of the input gate and the input modulation gate, then calculate the element-wise multiplication vector of the forget gate and the previous internal cell state and then adding the two vectors.
4. Calculate the current hidden state by first taking the element-wise hyperbolic tangent of the current internal cell state vector and then performing element wise multiplication with the output gate.

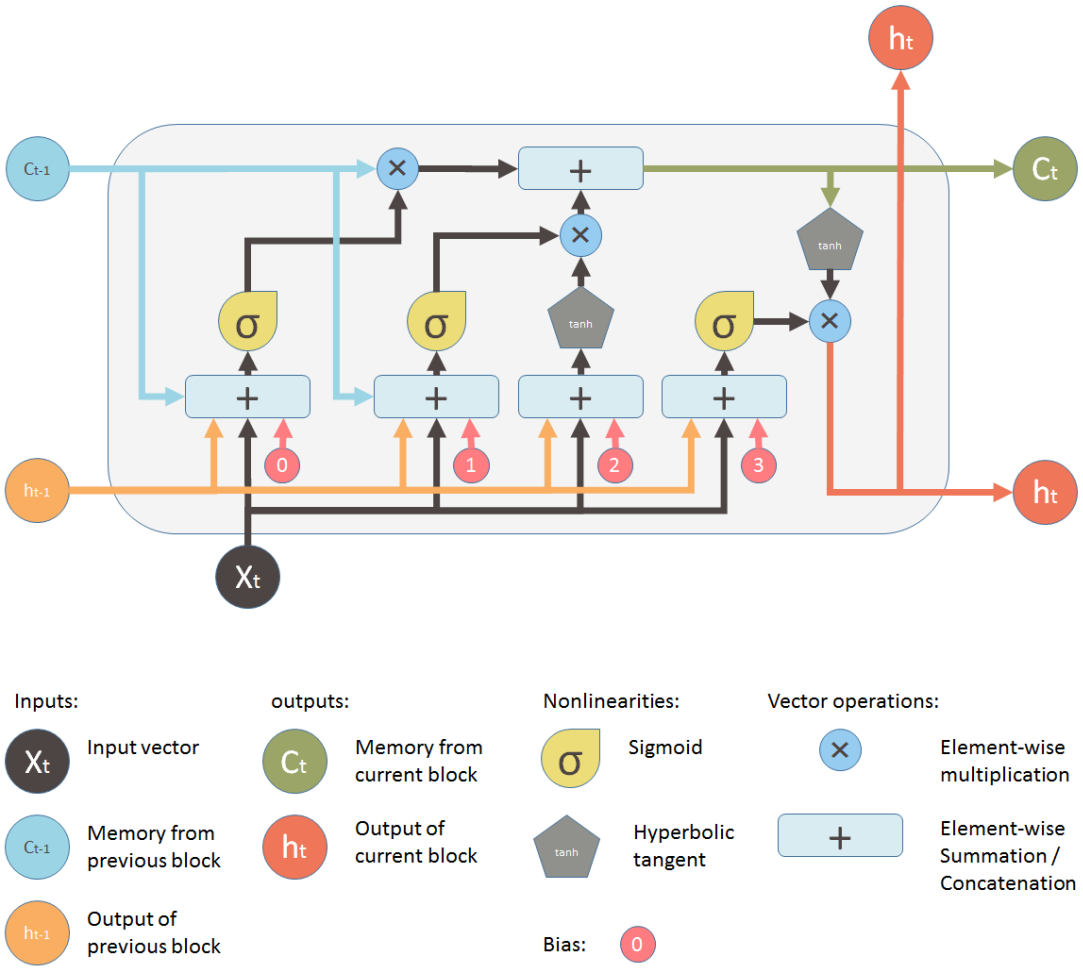

Fig 3. LSTM Block Diagram[13]

3.2 Gated Recurrent Unit (GRU)

A gated recurrent unit (GRU) is part of a specific model of recurrent neural network that intends to use connections through a sequence of nodes to perform machine learning tasks associated with memory and clustering, for instance, in speech recognition. To solve the Vanishing-Exploding gradients problem often encountered during the operation of a basic Recurrent Neural Network, many variations were developed. One of the lesser known but equally effective variations is the **Gated Recurrent Unit Network(GRU).**Unlike LSTM, it consists of only three gates and does not maintain an Internal Cell State. The information which is stored in the Internal Cell State in an LSTM recurrent unit is incorporated into the hidden state of the Gated Recurrent Unit. Fig[4] This collective information is passed onto the next Gated Recurrent Unit. The different gates of a GRU are as described below:-

Update Gate (z): It determines how much of the past knowledge needs to be passed along into the future. It is analogous to the Output Gate in an LSTM recurrent unit.[6]

Reset Gate (r): It determines how much of the past knowledge to forget. It is analogous to the combination of the Input Gate and the Forget Gate in an LSTM recurrent unit.

Current Memory Gate: It is often overlooked during a typical discussion on Gated Recurrent Unit Network. It is incorporated into the Reset Gate just like the Input Modulation Gate is a sub-part of the Input Gate and is used to introduce some non-linearity into the input and to also make the input Zero-mean. Another reason to make it a sub-part of the Reset gate is to reduce the effect that previous information has on the current information that is being passed into the future.[9]

The basic work-flow of a Gated Recurrent Unit Network is similar to that of a basic Recurrent Neural Network when illustrated, the main difference between the two is in the internal working within each recurrent unit as Gated Recurrent Unit networks consist of gates which modulate the current input and the previous hidden state.[12]

Working of a Gated Recurrent Unit:

Take input the current input and and the previous hidden state as vectors.

Calculate the values of the three different gates by following the steps given below:-

1. For each gate, calculate the parameterized currrent input and previous hidden state vectors by performing element-wise multiplication (hadmard product) between the concerned vector and the respective weights for each gate.
2. Apply the respective activation function for each gate element-wise on the parameterized vectors. Below given is the list of the gates with the activation function to be applied for the gate.
3. The process of calculating the Current Memory Gate is a little different. First, the Hadmard product of the Reset Gate and the previous hidden state vector is calculated. Then this vector is parameterized and then added to the parameterized current input vector.
4. To calculate the current hidden state, first a vector of ones and the same dimensions as that of the input is defined. First calculate the hadmard product of the update gate and the previous hidden state vector. Then generate a new vector by subtracting the update gate from ones and then calculate the hadmard product of the newly generated vector with the current memory gate. Finally add the two vectors to get the current hidden state vector.

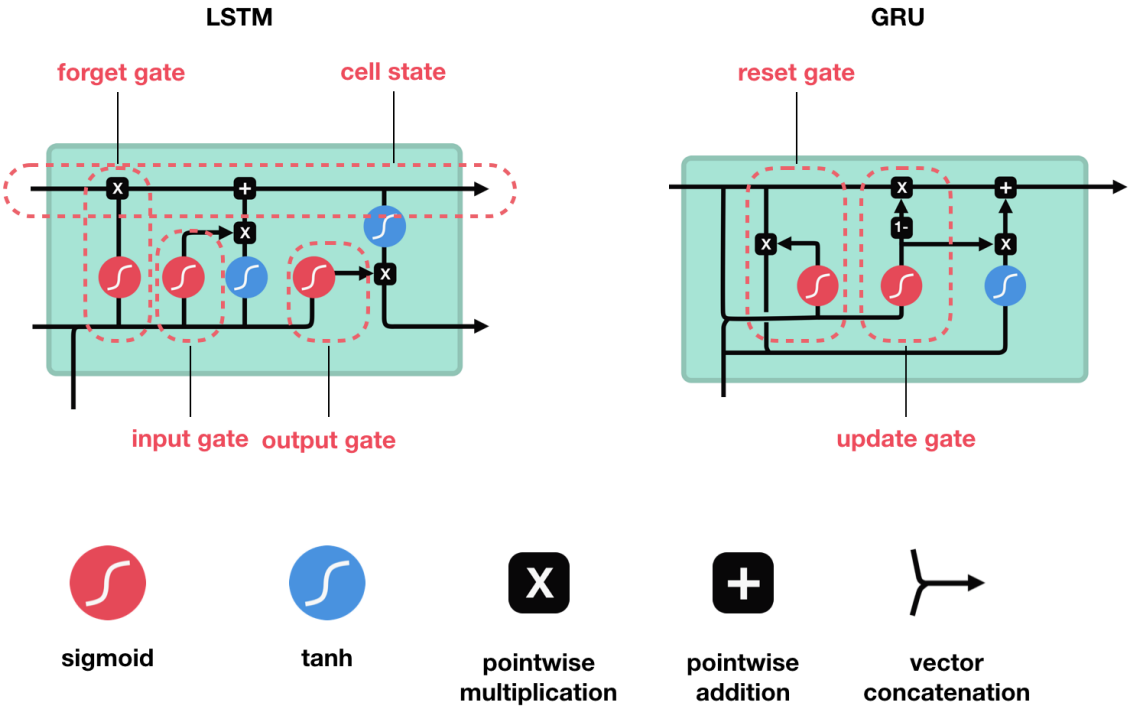

Fig 4. Comparison between LSTM and GRU Algorithms[14]

CHAPTER 4: DETAILED EXPLANATION

4.1 SOFTWARES AND LIBRARIES USED

ANACONDA: It is a free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. Package versions are managed by the package management system conda. The Anaconda distribution includes data-science packages suitable for Windows, Linux, and MacOS.[13]

Anaconda provides the tools needed to easily:

- Collect data from files, databases, and data lakes
- Manage environments with Conda (all package dependencies are taken care of at the time of download)
- Share, collaborate on, and reproduce projects
- Deploy projects into production with the single click of a button

JUPYTER: Project Jupyter is a non-profit, open-source project, born out of the IPython Project in 2014 as it evolved to support interactive data science and scientific computing across all programming languages. Jupyter will always be 100% open-source software, free for all to use. The Jupyter Notebook App is a server-client application that allows editing and running notebook documents via a web browser. The Jupyter Notebook App can be executed on a local desktop requiring no internet access (as described in this document) or can be installed on a remote server and accessed through the internet. Code is executed line by line in it making it more efficient.

KERAS LIBRARY: Keras is our recommended library for deep learning in Python, especially for beginners. Its minimalistic, modular approach makes it a breeze to get deep neural networks up and running. Keras doesn't handle low-level computation. Instead, it uses another library to do it, called the "Backend. So Keras is high-level API wrapper for the low-level API, capable of running on top of TensorFlow, CNTK, or Theano. Keras High-Level API handles the way we make models, defining layers, or set up multiple input-output models. In this level, Keras also compiles our model with loss and optimizer functions, training process with fit function. Keras doesn't handle Low-Level API such as making the computational graph, making tensors or other variables because it has been handled by the "backend" engine.[7]

Advantages: Fast Deployment and Easy to understand

Large Community Support

Have multiple Backends

Cross-Platform and Easy Model Deployment

Multi GPUs Support

TENSORFLOW LIBRARY: It is a Python library for fast numerical computing created and released by Google. It is a foundation library that can be used to create Deep Learning models directly or by using wrapper libraries that simplify the process built on top of TensorFlow.

It can run on single CPU systems, GPUs as well as mobile devices and large scale distributed systems of hundreds of machines. TensorFlow can train and run deep neural networks for handwritten digit classification, image recognition, word embeddings, recurrent neural networks, sequence-to-sequence models for machine translation, natural language processing, and PDE (partial differential equation) based simulations.

4.2 WORK FLOW

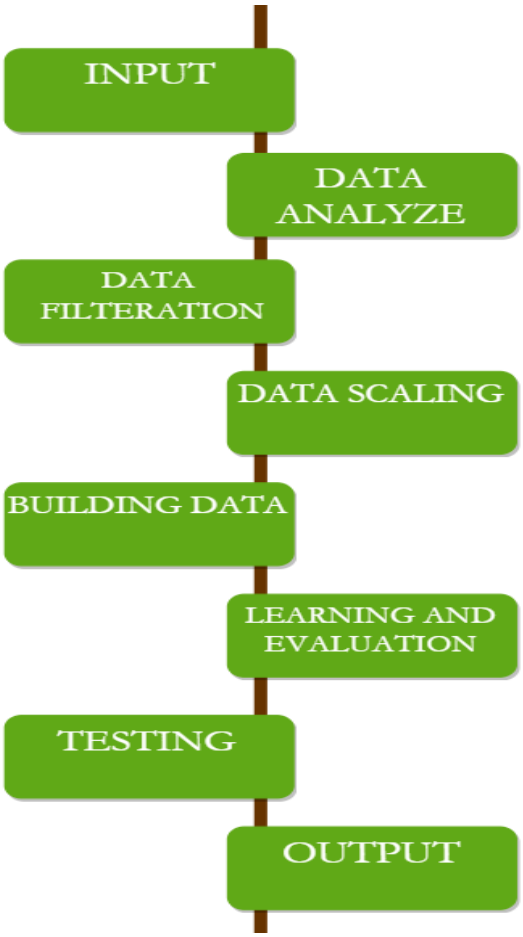

Fig 5. Flow Diagram

Data Analysis Phase: It analyze for the possible merging of data for improved model predictability.

Data Filtration Phase: This phase filters data to remove all empty/redundant values.

Data-Scaling Phase: Before data are passed to the model, the data are scaled according to model requirements. In this way, this phase reshapes data to make them more suitable for the model.

Keras with tensor flow is used as the backend library to make the model more accurate. The Keras sequential model consists of two layers named LSTM and dense layers. These layers process data in depth to analyze all kind of patterns formed in the dataset to make the model more precise.[5]

Data are trained using various LSTM units model and GRU.

Input values are passed to the model to give predicted values as the output. Then that output is compared with testing data to calculate accuracy and losses.

Various plots are taken.

DATASET:

A8 1/7/2017 12:00:00 AM

	A	B	C	D	E	F	G	H	I	J
1	Timestamp	Open	High	Low	Close	Volume (B	Volume (C	Weighted Price		
2	########	966.34	1005	960.53	997.75	6850.59	6764742	987.47		
3	########	997.75	1032	990.01	1012.54	8167.38	8273577	1013		
4	########	1011.44	1039	999.99	1035.24	9089.66	9276500	1020.56		
5	########	1035.51	1139.89	1028.56	1114.92	21562.46	23469645	1088.45		
6	########	1114.38	1136.72	885.41	1004.74	36018.86	36211400	1005.35		
7	########	1004.73	1026.99	871	893.89	27916.7	25523261	914.26		
8	########	894.02	907.05	812.28	906.2	20401.11	17624310	863.89		
9	########	906.2	941.81	881.3	909.75	8937.49	8168170	913.92		
10	########	909.8	912.87	875	896.23	8716.18	7780059	892.6		
11	########	896.09	912.47	889.41	905.05	8535.52	7704271	902.61		
12	########	905.24	918.4	755	778.62	35893.77	29459969	820.75		
13	########	778.7	832.99	751.34	807.47	17400.14	13632251	783.46		
14	########	807.51	831.4	775	825.86	11409.52	9224730	808.51		
15	########	825.98	837.76	810	818.27	6614.72	5470215	826.98		
16	########	819.52	823.45	808	821.86	4231.46	3455366	816.59		
17	########	821.86	835	818.09	831.81	6166.04	5107031	828.25		
18	########	831.76	908.5	827	905.99	12264.17	10775498	878.62		
19	########	905.95	915.99	851.74	887.7	11181.9	9830118	879.11		
20	########	887.69	910	878.89	901.01	11094.6	9929647	895		
21	########	902.23	902.43	880	895.8	6618.63	5915865	893.82		
22	########	895.81	928	895	921.98	5865.63	5373391	916.08		
23	########	921.98	937.74	886.76	923.76	7166.67	6569177	916.63		
24	########	923.75	927.57	913.21	913.52	3514.74	3234454	920.25		

Fig 6. This is how our data looks[10]

CHAPTER 5: RESULTS

The LSTM model in our project has 3 LSTM Layers and one Dense Layer. The model has 75 neurons in the first hidden layer, 30 neurons in the 2nd and 3rd layers and 1 neuron in the output layer for price prediction. The model is fitted with 50 training epochs with a batch size of 8.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_48 (LSTM)               (None, None, 75)          23100
_________________________________________________________________
lstm_49 (LSTM)               (None, None, 30)          12720
_________________________________________________________________
lstm_50 (LSTM)               (None, 30)                7320
_________________________________________________________________
dense_72 (Dense)             (None, 1)                 31
=================================================================
Total params: 43,171
Trainable params: 43,171
Non-trainable params: 0
```

Fig 7: LSTM Model

GRU network is also created with the same parameters and configuration as LSTM.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
gru_42 (GRU)                 (None, None, 75)          17325
_________________________________________________________________
gru_43 (GRU)                 (None, None, 30)          9540
_________________________________________________________________
gru_44 (GRU)                 (None, 30)                5490
_________________________________________________________________
dense_74 (Dense)             (None, 1)                 31
=================================================================
Total params: 32,386
Trainable params: 32,386
Non-trainable params: 0
```

Fig 8: GRU Model

Here, we can observe that the both LSTM and GRU have the same architecture but the number of parameters in LSTM is 43,171 and that in GRU is 32,386 because of the difference in gates. Because of the fewer parameters, GRU is computationally more efficient than LSTM.

The prediction is done for 240 days starting from 12-04-2017 and the BTC price versus Time for the actual and the predicted data is plotted using matplotlib library for the two models as shown below:

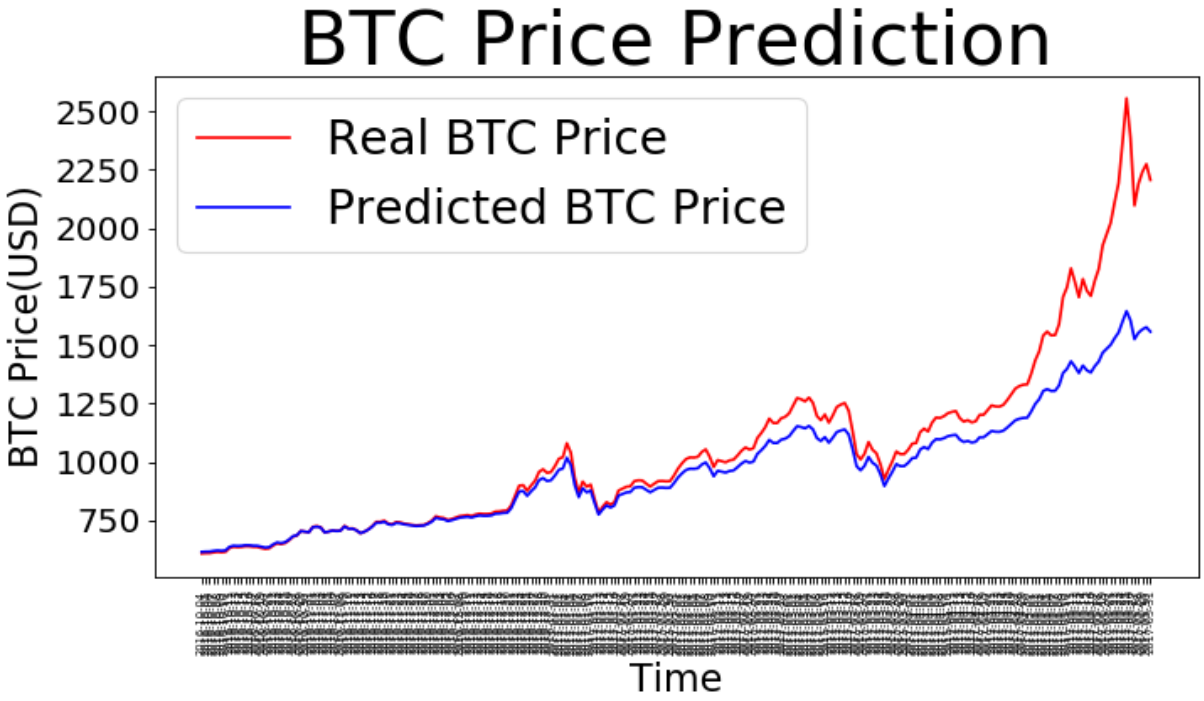

Fig 9: Prediction using LSTM

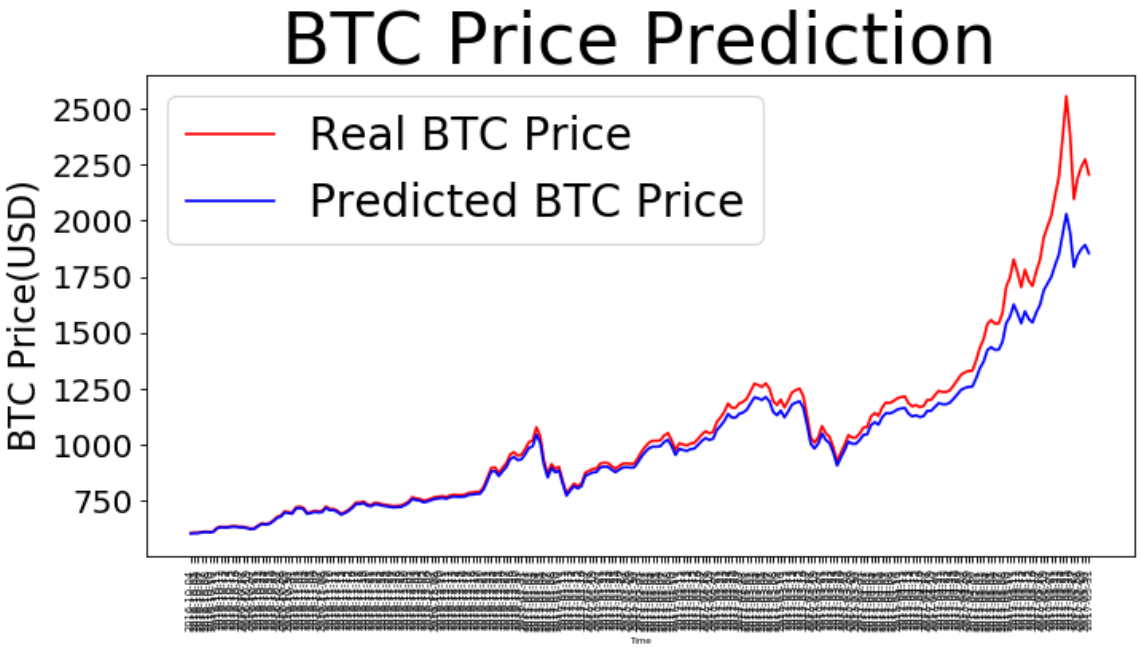

Fig 10: Prediction using GRU

The RMSE value for the LSTM model obtained is 0.022 and that of GRU is 0.014. We can see that in both the cases, both the models could not perform well for prediction for the higher BTC price, though GRU performed better than the LSTM model. For the lower price range both are successful in predicting the prices. Because of time series data and its non-linear nature prediction is difficult. GRU prediction is able to pick up the shape of observed data, but not quite there yet and there are rooms for improvement by means of optimizing and fine tuning the network architecture. The above two models weights and architecture have been save for future use.

CHAPTER 6: CONCLUSION AND FUTURE SCOPE

Predicting the future will always be on the top of the list of uses for machine learning algorithms. Here in this project we have attempted to predict the prices of Bitcoins using two deep learning methodologies. This work focuses on the development of project based learning in the field of Computer science engineering, by taking into account the problem definition, progression, student assessment and use of hands on activities based on use of deep learning algorithm to develop application which can predict bitcoin prices.[6]

From the above plots and accuracies, it can be seen that GRU is better in terms of prediction.

There is always room for improvement and, with the rate at which deep learning is growing, these improvements will surely be possible:

- Train the model on a larger data set to increase prediction accuracy.
- Design model with high number of neurons and run on a supercomputer or a cluster of system.
- Include more features to the feature map and integrate the model with other model which can learn based on customers' interest to a certain commodity.

The work in progress includes implementing Django API as a tool to fetch dataset and showing real time progress and bitcoin prices. Also, other features can be considered (although from our experiments with Bitcoin, more features have not always led to better results). Moreover, more algorithms like CNN, ARIMA etc will be taken into account for prediction in coming work progress.

REFERENCES

[1] M. Klobukov, G. Matthews,“Bitcoin Price Prediction Using Machine Learning”, Department of Computer Science, Drexel University, Philadelphia, Pennsylvania 19104, pp: 1-6, 2018

[2] V. Derbentsev, N. Datsenko, O. Stepanenko, and V. Bezkorovainyi-“ Forecasting cryptocurrency prices time series using machine learning approach” SHS Web of Conferences 65, 02001, 2019

[3] Y. Yao, J. Yi, S. Zhai, Y. Lin, T. K. G. Zhang, L. Yoonjae Lee-“ Predictive Analysis of Cryptocurrency Price Using Deep Learning” International Journal of Engineering & Technology, pp: 258-264, 2018

[4] S. Yogeshwaran, M. J. Kaur, P. Maheshwari- “Predicting Bitcoin Prices using Deep Learning” Department of Engineering Amity University Dubai, UAE- 9–11 April, 2019 - American University in Dubai, Dubai, UAE, IEEE Global Engineering Education Conference (EDUCON), pp: 1449-1454, April 2019

[5] T. Phaladisailoed, Faculty of Information Technology, “Machine Learning Models Comparison for Bitcoin Price Prediction”, Thanisa Numnonda Faculty of Information Technology, January 2019

[6] M. Thakker, B. Vaidhyanathan, “Bitcoin Stock Price Prediction” College of Computer & Information Science Northeastern University Boston, MA 02115, 32nd Conference on Neural Information Processing Systems, Montreal, Canada-2018

[7] F. Tschorsch, B. Scheuermann, “Bitcoin and beyond: A technical survey on decentralized digital currencies”, IEEE Communications Surveys & Tutorials Vol.18 No.3, pp: 2084-2123, 2016

[8] RC Phillips, D. Gorse, “Predicting cryptocurrency price bubbles using social media data and epidemic modeling”, IEEE Symposium Series on Computational Intelligence (SSCI), pp: 1-7, 2017

[9] https://medium.com/swlh/how-i-used-ml-to-predict-bitcoin-prices-82af7c655092

[10] https://www.kaggle.com/atuljhasg/bitcoin-price-prediction-using-rnn

[11] https://blog.quantinsti.com/rmm-lstm-gru-trading/?utm_campaign=News&utm_medium=Community&utm_source=DataCamp.com

[12] https://blog.floydhub.com/gru-with-pytorch/

[13] https://www.google.com/deeplearning_and_deeplearning/

[14] https://stackabuse.com/time-series-analysis-with-lstm-using-pythons-keras-library/

APPENDICES

LSTM CODE:

```
import numpy as np
import pandas as pd
import math
from math import sqrt
from matplotlib import pyplot as plt
from sklearn.metrics import mean_squared_error
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
import keras
from sklearn.preprocessing import MinMaxScaler
df = pd.read_csv(r'C:\Users\user\Desktop\bitstampUSD_1-mindata_.csv')
df['date'] = pd.to_datetime(df['Timestamp'],unit='s').dt.date
group = df.groupby('date')
Real_Price = group['Weighted Price'].mean()
df.head()
prediction_days=240
df_train= Real_Price[:len(Real_Price)-prediction_days]
df_test= Real_Price[len(Real_Price)-prediction_days:]
training_set = df_train.values
training_set= training_set.astype('float32')
training_set = np.reshape(training_set, (len(training_set), 1))
sc = MinMaxScaler(feature_range=(0,1))
training_set = sc.fit_transform(training_set)
X_train = training_set[0:len(training_set)-1]
y_train = training_set[1:len(training_set)]
X_train = np.reshape(X_train, (len(X_train), 1, 1))
regressor = keras.Sequential()
regressor.add(keras.layers.LSTM(units = 75,
activation='sigmoid',return_sequences=True,input_shape = (None, 1)))
regressor.add(keras.layers.LSTM(units=30,return_sequences=True))
regressor.add(keras.layers.LSTM(units=30))
```

```
regressor.add(Dense(units = 1))
regressor.compile(optimizer = 'adam', loss = 'mean_squared_error')
regressor.summary()

regressor.fit(X_train, y_train, epochs = 50, batch_size=8, verbose=2)
testpredict= regressor.predict(X_train,batch_size=5)
score= regressor.evaluate(X_train,y_train,batch_size=5,verbose=2)
MSE=score
RMSE=math.sqrt(score)
print('Test RMSE:%.3f'%RMSE)
test_set = df_test.values
inputs = np.reshape(test_set, (len(test_set), 1))
inputs = sc.transform(inputs)
inputs = np.reshape(inputs, (len(inputs), 1, 1))

predicted_BTC_price = regressor.predict(inputs)
predicted_BTC_price = sc.inverse_transform(predicted_BTC_price)

plt.figure(figsize=(10,5), dpi=80, facecolor='w', edgecolor='k')
ax = plt.gca()
plt.plot(test_set, color = 'red', label = 'Real BTC Price')
plt.plot(predicted_BTC_price, color = 'blue', label = 'Predicted BTC Price')
plt.title('BTC Price Prediction', fontsize=40)
df_test = df_test.reset_index()
x=df_test.index
labels = df_test['date']
plt.xticks(x, labels, rotation = 'vertical')
for tick in ax.xaxis.get_major_ticks():
    tick.label1.set_fontsize(6)
for tick in ax.yaxis.get_major_ticks():
    tick.label1.set_fontsize(18)
plt.xlabel('Time', fontsize=20)
plt.ylabel('BTC Price(USD)', fontsize=20)
plt.legend(loc=2, prop={'size': 25})
plt.show()
```

```
#trainScore = sqrt(mean_squared_error(inputs,predicted_BTC_price))
#print('Train Score: %.2f RMSE' % (trainScore))
```

GRU CODE:

```
import numpy as np
import math
import pandas as pd
from math import sqrt
from matplotlib import pyplot as plt
from sklearn.metrics import mean_squared_error
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import GRU
import keras
from sklearn.preprocessing import MinMaxScaler
df = pd.read_csv(r'C:\Users\user\Desktop\bitstampUSD_1-mindata_.csv')
df['date'] = pd.to_datetime(df['Timestamp'],unit='s').dt.date
group = df.groupby('date')
Real_Price = group['Weighted Price'].mean()
df.head()
prediction_days=240
df_train= Real_Price[:len(Real_Price)-prediction_days]
df_test= Real_Price[len(Real_Price)-prediction_days:]
training_set = df_train.values
training_set = np.reshape(training_set, (len(training_set), 1))
sc = MinMaxScaler(feature_range=(0,1))
training_set = sc.fit_transform(training_set)
X_train = training_set[0:len(training_set)-1]
y_train = training_set[1:len(training_set)]
X_train = np.reshape(X_train, (len(X_train), 1, 1))
regressor = Sequential()
regressor.add(keras.layers.GRU(units = 75,
activation='relu',return_sequences=True,input_shape = (None, 1)))
regressor.add(keras.layers.GRU(units=30,return_sequences=True))
regressor.add(keras.layers.GRU(units=30))
```

```
regressor.add(Dense(units = 1))
regressor.compile(optimizer = 'adam', loss = 'mean_squared_error')
regressor.fit(X_train, y_train, epochs =50, batch_size=8, verbose=2)
regressor.summary()
testpredict= regressor.predict(X_train,batch_size=8)
score= regressor.evaluate(X_train,y_train,batch_size=8,verbose=2)
MSE=score
RMSE=math.sqrt(score)
print('Test RMSE:%.3f'%RMSE)

test_set = df_test.values
inputs = np.reshape(test_set, (len(test_set), 1))
inputs = sc.transform(inputs)
inputs = np.reshape(inputs, (len(inputs), 1, 1))
predicted_BTC_price = regressor.predict(inputs)
predicted_BTC_price = sc.inverse_transform(predicted_BTC_price)

plt.figure(figsize=(10,5), dpi=80, facecolor='w', edgecolor='k')
ax = plt.gca()
plt.plot(test_set, color = 'red', label = 'Real BTC Price')
plt.plot(predicted_BTC_price, color = 'blue', label = 'Predicted BTC Price')
plt.title('BTC Price Prediction', fontsize=40)
df_test = df_test.reset_index()
x=df_test.index
labels = df_test['date']
plt.xticks(x, labels, rotation = 'vertical')
for tick in ax.xaxis.get_major_ticks():
    tick.label1.set_fontsize(6)
for tick in ax.yaxis.get_major_ticks():
    tick.label1.set_fontsize(18)
plt.xlabel('Time', fontsize=5)
plt.ylabel('BTC Price(USD)', fontsize=20)
plt.legend(loc=2, prop={'size': 25})
plt.show()
```