iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 9 **Issue:** XII **Month of publication:** December 2021

DOI: https://doi.org/10.22214/ijraset.2021.39101

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 9 Issue XII Dec 2021 - Available at www.ijraset.com

Profitable Algorithmic Trading Strategy

Prince Nathan S[1], Onkar Saudagar[2], Rutika Shinde3
[1]*PG.Student, Department of Data Science and Analytics, National Institute of Electronics and Information Technology, Chennai, India*
[2]*Student, Pune Institute of Computer Technology, Department of Information Technology Engineering, Pune, India*
[3]*Student, Sinhgad Institute of Technology and Science, Department of Information Technology, Pune, India*

Abstract: Cryptocurrency has drastically increased its growth in recent years and Bitcoin (BTC) is a very popular type of currency among all the other types of cryptocurrencies which is been used in most of the sectors nowadays for trading, transactions, bookings, etc. In this paper, we aim to predict the change in bitcoin prices by using machine learning techniques on data from Investing.com. We interpret the output and accuracy rate using various machine learning models. To see whether to buy or sell the bitcoin we created exploratory data analysis from a year of data set and predict the next 5 days change using machine learning models like logistic Regression, Logistic Regression with PCA (Principal Component Analysis), and Neural network.

Keywords: Data Science, Machine Learning, Regression, PCA, Neural Network, Data Analysis

I. INTRODUCTION

In the modern era, the cryptocurrency created an impact on the financial sector. It is one of the greatest revolutions that the future might see after the introduction of Artificial intelligence. Blockchain technology kept a rigid foundation for the greater growth of cryptocurrency. Bitcoin has become one of the buzzwords in the market that has gained traction ever since this virtual currency touched an all-time high. The topic of cryptocurrencies has become extremely popular recently, and as cryptocurrencies have become more popular in the mainstream, some of their prices have greatly increased as well. Blockchain is the technology that enables the existence of cryptocurrency (among other things). Bitcoin is the name of the best-known cryptocurrency, the one for which blockchain technology was invented. A cryptocurrency is a medium of exchange, such as the US dollar, but is digital and uses encryption techniques to control the creation of monetary units and to verify the transfer of funds. In this project, we decided to focus purely on Bitcoin, as it was the cryptocurrency with the most data and the highest volume of trading aim of our internship project is to check whether we can buy or sell the bitcoin and also predict for 5 days change of the bitcoin pricing using the dataset from investing.com. One year dataset helps to predict the 5 days the change so that the decision regarding the buy or sell can be made. Using the data set we create three models for examining the price data for the bitcoin. From the bitcoin pricing, dataset Exploratory data analysis is done and also the machine models ie. Logistic Regression, Logistic Regression with PCA (Principal component analysis), and Neural network (LSTM MODEL).

II. DATASET AND FEATURES

We performed data analysis on the Bitcoin data to create a profitable algorithmic trading strategy.
The Data that was collected ranges between the period of April 2021 and October 2021.
Trading days – 184 Days

A. Main Dataset Used Is

Bitcoin historical dataset from in.investing.com website
Original format of the dataset: CSV
Columns in the dataset with a brief description:

1) *Date:* Format: DD-MM-YY
2) *Price:* Price of the stock
3) *Open:* Open Price of stock i.e. The price at which stock opened
4) *High:* The highest price the stock touched
5) *Low:* The lowest price the stock touched
6) *Volume:* The number of shares that changed hands during a given day
7) Change percentage

©IJRASET: All Rights are Reserved

III. EXPLORATORY DATA ANALYSIS

In statistics, exploratory data analysis is an approach to analyzing data sets to summarize their main characteristics, often using statistical graphics and other data visualization methods. A statistical model can be used or not, but primarily EDA is for seeing what the data can tell us beyond the formal modeling or hypothesis testing task.

EDA refers to the exploration (or explanation of the data) of the given data for future usage. It is a good practice that we need to understand our data and gather as many insights out of the data.

IV. STEPS TO CREATING AN EXPLORATORY DATA ANALYSIS

A. The float values are in strings so we need to convert it to float & perform numeric operations onto it , The numeric data which is of string type is converted to float type (Label Encoded) and also K n % will be removed:

```
In [5]: # Remove K and % from last two columns to perform numeric operations onto it

        # The numeric data which is of string type is converted to float type and also K n % will be removed/
        data1 = df.replace('[^\d.]','', regex=True).astype(float)
```

```
In [6]: data1
```

Out[6]:

	Price	Open	High	Low	Vol.	Change %
Date						
2021-10-03	47878.9	47665.4	48097.0	47123.0	36.99	0.44
2021-10-02	47666.9	48147.2	48306.9	47451.7	39.82	1.00
2021-10-01	48146.0	43824.4	48435.2	43292.9	94.66	9.86
2021-09-30	43823.3	41534.5	44101.2	41416.7	64.32	5.50
2021-09-29	41536.8	41023.1	42571.2	40815.0	48.21	1.25
...	...	...	...	...	...	...
2021-04-07	55948.7	57996.3	58627.7	55489.3	110.69	3.53
2021-04-06	57996.3	59169.0	59487.0	57403.3	77.15	1.69
2021-04-05	58993.4	58202.3	59205.1	56842.7	54.13	1.36
2021-04-04	58199.9	57059.7	58464.8	56470.6	57.21	2.00
2021-04-03	57059.9	58976.8	59770.5	56906.7	68.74	3.25

184 rows × 6 columns

B. Assigning New Column "Buy" Based on Price with Values 0 and 1:

```
In [9]: data1.loc[data1['Price'] < data1['Open'] , 'Buy'] = 0
        data1.loc[data1['Price'] > data1['Open'] , 'Buy'] = 1
```

```
In [10]: data1
```

Out[10]:

	Price	Open	High	Low	Vol.	Change %	Buy
Date							
2021-10-03	47878.9	47665.4	48097.0	47123.0	36.99	0.44	1.0
2021-10-02	47666.9	48147.2	48306.9	47451.7	39.82	1.00	0.0
2021-10-01	48146.0	43824.4	48435.2	43292.9	94.66	9.86	1.0
2021-09-30	43823.3	41534.5	44101.2	41416.7	64.32	5.50	1.0
2021-09-29	41536.8	41023.1	42571.2	40815.0	48.21	1.25	1.0
...	...	...	...	...	...	...	...
2021-04-07	55948.7	57996.3	58627.7	55489.3	110.69	3.53	0.0
2021-04-06	57996.3	59169.0	59487.0	57403.3	77.15	1.69	0.0
2021-04-05	58993.4	58202.3	59205.1	56842.7	54.13	1.36	1.0
2021-04-04	58199.9	57059.7	58464.8	56470.6	57.21	2.00	1.0
2021-04-03	57059.9	58976.8	59770.5	56906.7	68.74	3.25	0.0

184 rows × 7 columns

C. Finding the Null Values Present in the Given Series of Objects

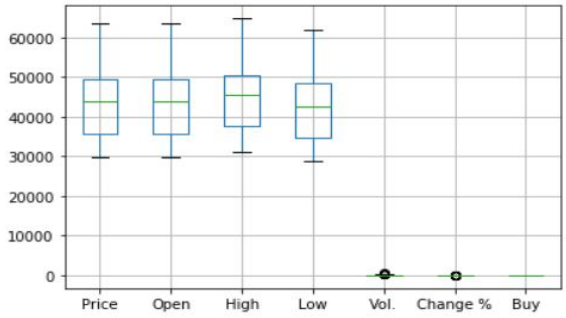
```
In [11]: data1.isnull().sum()

Out[11]: Price      0
         Open       0
         High       0
         Low        0
         Vol.       0
         Change %   0
         Buy        0
         dtype: int64
```

D. Boxplots are a measure of how well distributed the data in a data set is. Here is the example below

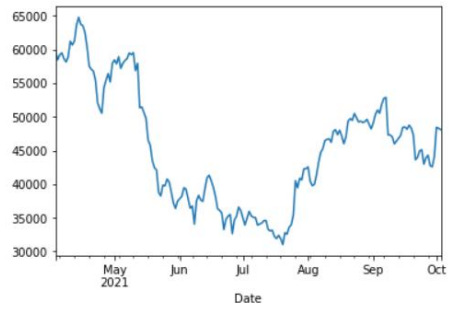
```
In [12]: data1.boxplot()

Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0x1e94aaf41c8>
```



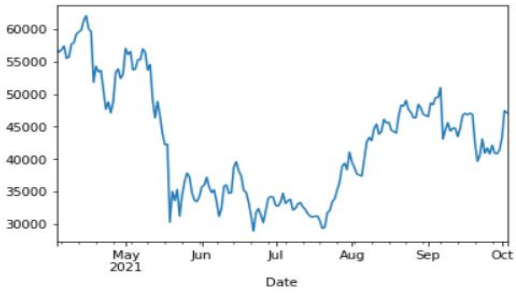
E. Ax Object Plotting on Column "High":

```
In [13]: ax = data1['High'].plot(style=['-'])
```



F. ax object plotting on column on "Low"

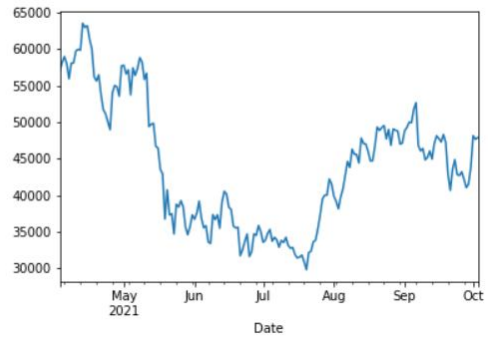
```
In [14]: ax = data1['Low'].plot(style=['-'])
```



©IJRASET: All Rights are Reserved

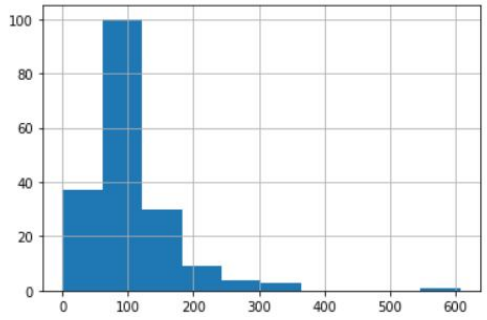
G. ax object plotting on column "Price":

```
In [15]: ax = data1['Price'].plot(style=['-'])
```



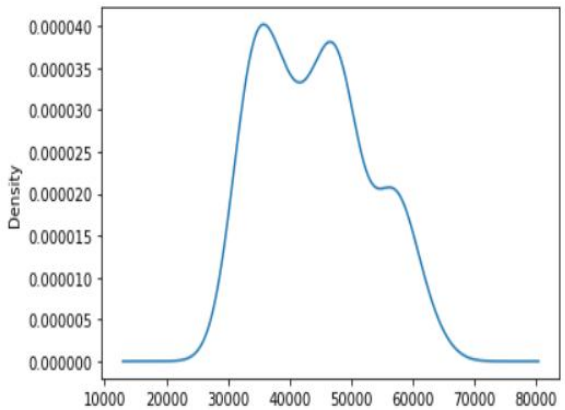
H. A histogram is a representation of the distribution of data. Here is an example of Vol. column :

```
In [16]: data1['Vol.'].hist()
Out[16]: <matplotlib.axes._subplots.AxesSubplot at 0x1e94b149508>
```



I. Kernel density estimation (KDE) is a non-parametric way to estimate the probability density function (PDF) of a random variable. This function uses Gaussian kernels and includes automatic bandwidth determination:

```
In [17]: # Kernel Distribution
         ax = data1['Price'].plot(kind= 'kde')
```



J. *Exponential smoothing is a time series forecasting method for univariate data that can be extended to support data with a systematic trend or seasonal component. Here we are doing it on "Price" column:*

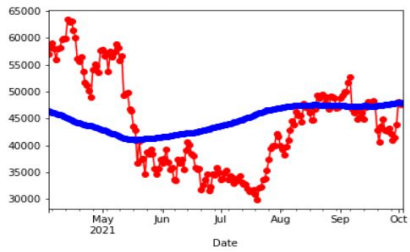
```
In [18]: # exponential smoothing

         from statsmodels.tsa.holtwinters import SimpleExpSmoothing

         model = SimpleExpSmoothing(data1['Price']).fit(smoothing_level=.01, optimized = False)

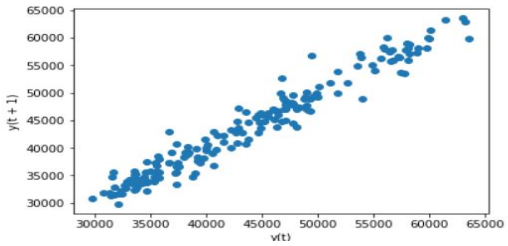
         data1['Price'].plot(marker='o', color='red')
         model.fittedvalues.plot(marker='o', color='blue')

         C:\Users\omkar\anacondanew\lib\site-packages\statsmodels\tsa\base\tsa_model.py:165: ValueW
         so inferred frequency -1D will be used.
           % freq, ValueWarning)
Out[18]: <matplotlib.axes._subplots.AxesSubplot at 0x1e94b3beac8>
```



K. *Checking Randomness in the Price Column*

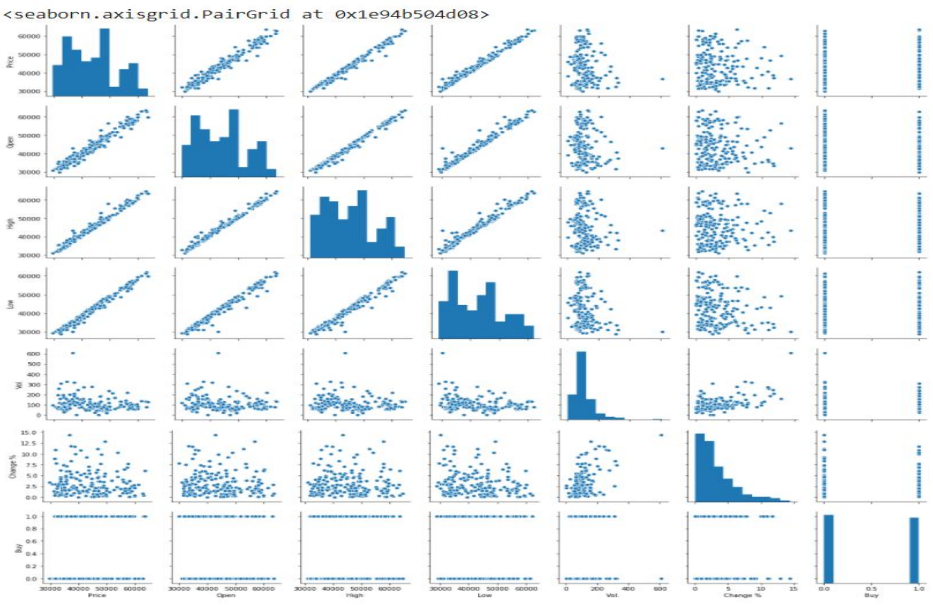
```
In [19]: # Randomness in the Price Column
         from pandas.plotting import lag_plot
         lag_plot(data1['Price'])
Out[19]: <matplotlib.axes._subplots.AxesSubplot at 0x1e94b1a6288>
```



L. *Creating a grid of Axes such that each numeric variable in data will by shared across the y-axes across a single row and the x-axes across a single column:*

```
[ ]  sns.pairplot(data=data1)

<seaborn.axisgrid.PairGrid at 0x1e94b504d08>
```



V. PRINCIPAL COMPONENT ANALYSIS.

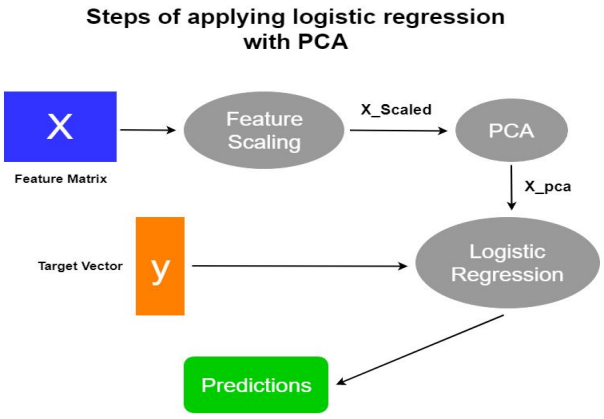
A. When we have a lot of variables in the dataset we can reduce them using PCA.
B. In our dataset we principal component analysis as the second model.
C. We applied Principal Component Analysis (PCA) to the data in order to figure out the correlations between our features and remove the noise in our data set.
D. First, we normalized our data by subtracting the mean of each feature and dividing by the standard deviation of each feature from each training point.

VI. LOGISTIC REGRESSION WITH PCA

A. PCA is an unsupervised machine learning algorithm that is used for dimensionality reduction.
B. If the variables are not measured on a similar scale, we need to do feature scaling before applying PCA for our data.
C. This is because PCA directions are highly sensitive to the scale of the data.
D. The most important part of PCA is selecting the best number of components for the given dataset.
E. PCA technique is basically used for dimensional reduction. We can import PCA from sklearn decomposition from this library we called PCA.

VII. STEPS IN LOGISTIC REGRESSION WITH PCA

A. Model 1:
Logistic Regression Model with PCA
1) Split the data into X and y
2) Split the data into training and test data set
3) Decide the number of PCA components based on the explained variance
4) Train the PCA model
5) Check the correlations between components
6) Apply the PCA model to the test data
7) Train the Logistic Regression model



VIII. REGRESSION WITHOUT PCA

A logistic regression model on the transformed dataset (the dataset obtained by applying PCA) is done. One of the weaknesses of PCA is that we won't know which variables are the top predictors. To know the top predictors we will have to build the Linear Regression model without PCA. As we don't use PCA, to reduce the number of variables will use RFE + manual.
One of the advantages of PCA is that we don't need to worry about multicollinearity in the data (highly correlated features). So on the second model where we don't use PCA, have to handle the multicollinearity, i.e. remove the highly correlated features using VIF (Variance Inflation Factor)

IX. SUMMARY

PCA is useful to remove multicollinearity. It acts as a data pre-processing step. PCA has many other use cases. It is just one of the dimensionality reduction techniques

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 9 Issue XII Dec 2021 - Available at www.ijraset.com

X. LSTM

LSTMs are widely used for sequence prediction problems and have provento be extremely effective. The reason they work so well is that LSTM can store past important information and forget the information that is not.

LSTM has three gates:

1) The input gate: The input gate adds information to the cell state,
2) The forget gate: It removes the information that is no longer requiredby the model,
3) The output gate: Output Gate at LSTM selects the information to beshown as output.

While Implementing any LSTM, we should always reshape our X train in 3D,add 1 the reason behind is the time step and the 1 is given to the LSTM.

Code:

```
X_train =X_train.reshape(X_train.shape[0],X_train.shape[1] , 1)
X_test = X_test.reshape(X_test.shape[0],X_test.shape[1] , 1)
```

Then import required modules for the stacked LSTM.

Code:

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers  import  Dense
from tensorflow.keras.layers import LSTM
```

We will be using a sequential model and adding the layers of the LSTM assaid, in the above sentence. The first layer should be the time step in 1.

Code:

```
model=Sequential()
model.add(LSTM(50,return_sequences=True,input_shape=(100,1)))
model.add(LSTM(50,return_sequences=True))
```

```
    model.add(LSTM(50))
    model.add(Dense(1))

    model.compile(loss='mean_squared_error',optimizer='adam')
```

Let's see the summary: The final part is to fit the X_train and the y_train.
Prediction: Predict both the X_train and the X_test, now let's scaler inverse transform because I want to see the root mean square performance.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

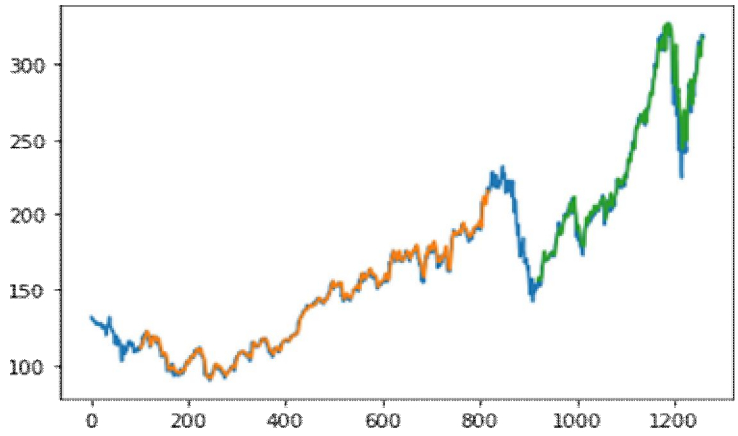
Code:

```
train_predict=model.predict(X_train)
test_predict=model.predict(X_test)
train_predict=scaler.inverse_transform(train_predict)
test_predict=scaler.inverse_transform(test_predict)
```

Code:

```
import math
from sklearn.metrics import mean_squared_error
math.sqrt(mean_squared_error(y_train,train_predict))
```

Here the time step is 100, Whatever the values in train predict and test predict. I got I am just plotting it don't forget we have to inverse the scalertransform.



1) Green indicates the Predicted Data
2) Blue indicates the Complete Data
3) Orange indicates the Train Data

XI. MODEL EVALUATION

1) *Logistic Regression*: We decided to transition into a weighted logistic regression model based on the sign of the price change so that our loss function would be accurately correlated to the gains we were making with our strategy of either investing at each time step or doing nothing. We made a prediction of the sign of the pricechange by applying the indicator function on outputs and using the absolute value of the outputs as the individual weights. This way the loss function that we are minimizing is calculated as below, and our logistic regression aims to minimize loss, which correlates to us maximizing our gains:

$$\mathcal{L}(\hat{y}, y) = \sum_i w_i(-(z_i \log(\hat{y}_i) + (1 - z_i) \log(1 - \hat{y}_i)))$$

$$w_i = |y_i|, z_i = \mathbb{1}[y_i > 0]$$

After running the weighted regression, we defined a few metrics; namely, the weighted accuracy WA and gains, G. The weighted accuracy is a weighted measurement to judge the accuracy of our algorithm in proportion to the gainsthat we're receiving. The gains correspond to the average price ratio increase we get each time step.

2) Principal Component Analysis

$$WA(\hat{y}, y) = \frac{\sum_i w_i \mathbb{1}[|\hat{y}_i - z_i| < 0.5]}{\sum_i w_i}$$

$$G(\hat{y}, y) = \sum_i y_i \mathbb{1}[\hat{y}_i > 0.5]$$

```
[ ]  # Use score method to get accuracy of model
     score = classifier.score(x_test, y_test)
     print(score)

     0.6756756756756757
```

```
     from sklearn.metrics import accuracy_score
     accuracy_score(y_test,y_pred)

     0.6756756756756757
```

In our second model, we applied Principal Component Analysis (PCA) to the data in order to figure out the correlations between our features and remove the noise in our data set. First, we normalized our data by subtracting the meanof each feature and dividing by the standard deviation of each feature from each training point. We then created the covariance matrix.
where m is the number of time steps we have in the training set, and xirepresented the features from the i'th training data point. To find the k principal components, we took the k eigenvectors corresponding to the klargest eigenvalues.

$$\frac{1}{m}\sum_{i=1}^{m} x_i x_i^\top$$

$$\frac{1}{m}\sum_{i=1}^{m} x_i x_i^\top$$

By looking at the most significant eigenvectors, we projected the relationships between our features and did some self-selection to remove either low variance features or highly correlated features. We decided on removing WAP, VR, and R as a result of our analysis.
Then, we projected each data point onto the principal components by taking the matrix product between the training points and the matrix of concatenatedeigenvectors, which got us our new feature set and once again ran weighted linear regression.
Given that vi is our ith principal component, our new features for our data would be as below for our k principal components and X is them by p matrixcontaining all our m training examples and their p features:
To find the optimal number of principal components to use, we utilized our development set in order to find the optimal k that maximized the gains forour development set.

$$[Xv_1 \quad Xv_2 \quad \dots \quad Xv_k]$$

```
[ ]  # Predict for One Observation (image)
     logisticRegr.predict(test_img[0].reshape(1,-1))

     array([1.])
```

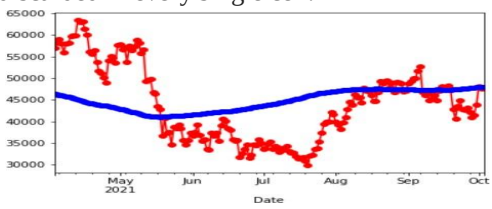
```
[ ]  # Predict for One Observation (image)
     logisticRegr.predict(test_img[0:10])

     array([1., 1., 0., 0., 1., 1., 1., 1., 0., 1.])
```

```
[ ]  logisticRegr.score(test_img, y_test)

     0.8243243243243243
```

3) LSTM: LSTM is a type of recurrent neural network but is better than traditional recurrent neural networks in terms of memory. Having a good hold over memorizing certain patterns LSTMs perform fairly better. As with every other NN, LSTM can have multiple hidden layers and as it passes through every layer,the relevant information is kept and all the irrelevant information gets discarded in every single cell.




```
[ ]  ### Calculate RMSE performance metrics
     import math
     from sklearn.metrics import mean_squared_error
     math.sqrt(mean_squared_error(y_train,train_predict))

     45788.90628632325

[ ]  ### Test Data RMSE
     math.sqrt(mean_squared_error(ytest,test_predict))

     26545.355229003915
```

XII. CONCLUSION

Long Short-Term Memory (LSTM) has better predictions than the other models since it works well on time series datasets and the accuracy of it is better than other models trained. Bitcoin Being the most volatile cryptocurrency the predictions made are fairly correct and using the LSTM model would be a better choice to use and predict the future prices and can invest wisely into bitcoin by using the model we built. In this paper, we predict the price of Cryptocurrencies(Bitcoin) by using an LSTM, Logistic Regression, and Logistic Regression with PCA. We introduced a technique to adaptively learn the pattern of the market's reaction to any updated information. Moreover, alternate reaction functions can be tested to learn the pattern of the market's reaction to fresh data. These functions can themselves be stochastic in nature to better simulate market volatility. In addition to that, there are a lot of unexplored territories in the cross-disciplinary field of stochastic processes and neural networks that can be exploited in Cryptocurrency markets.

REFERENCES

[1] Kondor, I. Csabai, J. Szäle, M. Pósfai, and G. Vattay, ''Inferring the interplay between network structure and market effects in bitcoin,'' New J. Phys., vol. 16, no. 12, 2014, Art. no. 125003.

[2] S. Hochreiter and J. Schmidhuber, ''Long short-term memory,'' Neural Comput., vol. 9, no. 8, pp. 1735–1780, 1997.

[3] W. Yiying and Z. Yeze, ''Cryptocurrency price analysis with artificial intelligence,'' in Proc. 5th Int. Conf. Inf. Manage. (ICIM), Mar. 2019, pp. 97–101.

[4] J. Rebane, ''Seq2Seq RNNs and ARIMA models for cryptocurrency prediction: A comparative study,''

[5] C. J. Hutto and E. Gilbert, ''Vader: A parsimonious rule-based model for sentiment analysis of social media text,'' in Proc. ICWSM, 2014, pp. 1–16.

[6] Y. Peng, P. H. M. Albuquerque, J. M. Camboim de Sá, A. J. A. Padula, and M. R. Montenegro, ''The best of two worlds: Forecasting high frequency volatility for cryptocurrencies and traditional currencies with support vector regression,'' Expert Syst. Appl., vol. 97, pp. 177–192, May 2018.

[7] G. T. Friedlob and F. J. Plewa, Jr., Understanding Return on Investment. Hoboken, NJ, USA: Wiley, 1996.

[8] E. F. Fama, ''Random walks in stock market prices,'' Financial Anal. J., vol. 21, no. 5, pp. 55–59, Sep. 1965.

[9] B. G. Malkiel, A Random Walk Down Wall Street: Including a Life-Cycle Guide to Personal Investing. New York, NY, USA: W. W. Norton & Company, 1973.

[10] D. Vasan, M. Alazab, S. Wassan, B. Safaei, and Q. Zheng, ''Image-based malware classification using an ensemble of CNN architectures (IMCEC),'' Comput. Secur., vol. 92, May 2020, Art. no. 101748.

[11] D. Mungra, A. Agrawal, P. Sharma, S. Tanwar, and M. S. Obaidat, ''PRATIT: A CNN-based emotion recognition system using histogram equalization and data augmentation,'' Multimedia Tools Appl., vol. 79, nos. 3–4, pp. 2285–2307, Jan. 2020.

[12] A. C. Logue, ''The efficient market hypothesis and its critics,'' CFA Dig., vol. 33, no. 4, pp. 40–41, Nov. 2003.

[13] W. Brock, J. Lakonishok, and B. LeBARON, ''Simple technical trading rules and the stochastic properties of stock returns,'' J. Finance, vol. 47, no. 5, pp. 1731–1764, Dec. 1992.

[14] R. U. Khan, X. Zhang, M. Alazab, and R. Kumar, ''An improved convolutional neural network model for intrusion detection in networks,'' in Proc. Cybersecurity Cyberforensics Conf. (CCC), May 2019, pp. 74–77.

[15] Bitinfocharts. Accessed: May 1, 2020. [Online]. Available: https://bitinfocharts.com/

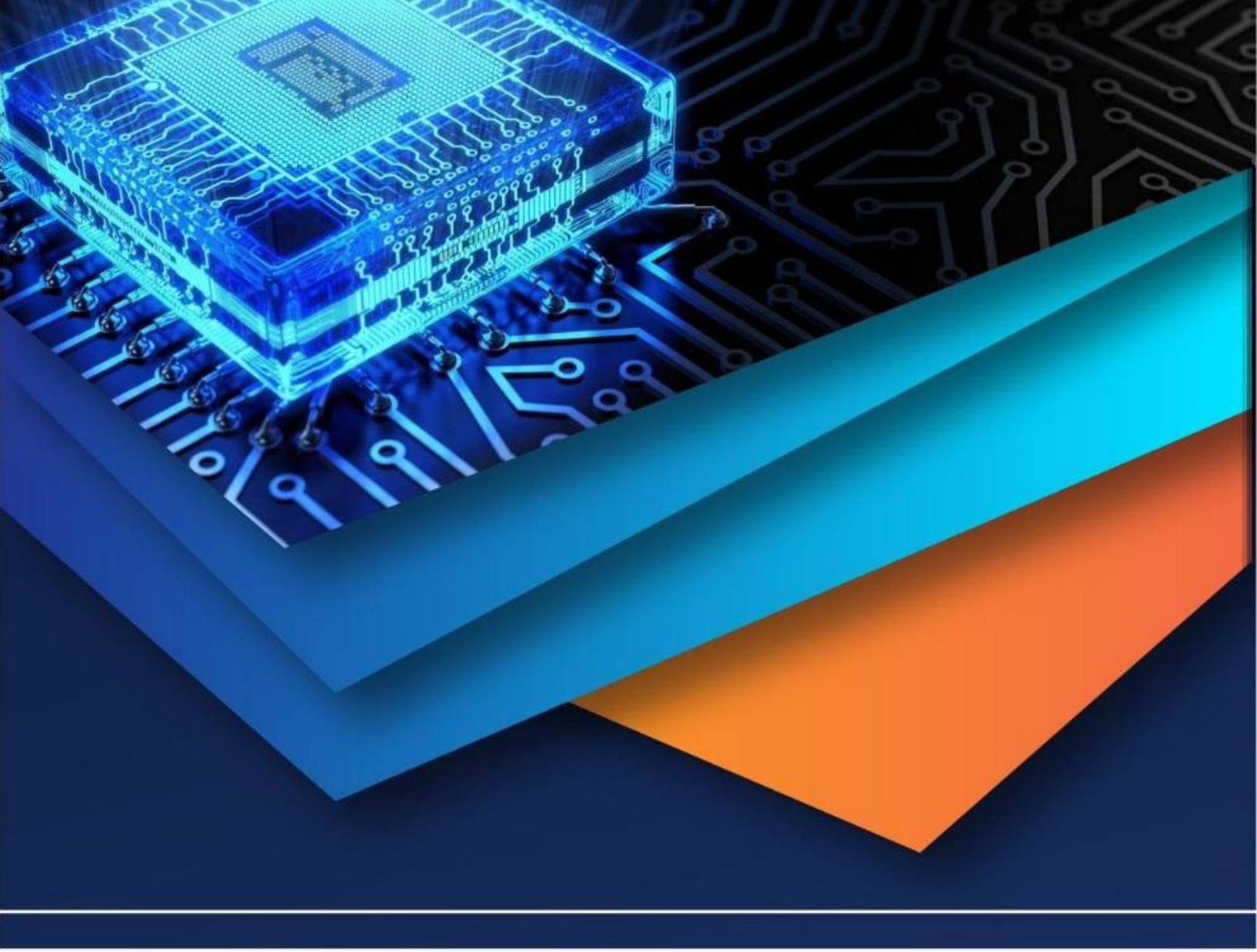
[16] D. Garcia, C. J. Tessone, P. Mavrodiev, and N. Perony, ''The digital traces of bubbles: Feedback cycles between socio-economic signals in the bitcoin economy,'' J. Roy. Soc. Interface, vol. 11, no. 99, Oct. 2014, Art. no. 20140623. [17] S. Rahman, J. N. Hemel, S. Junayed Ahmed Anta, H. A. Muhee, and J. Uddin, ''Sentiment analysis using R: An approach to correlate cryptocurrency price fluctuations with change in user sentiment using machine learning,'' in Proc. Joint 7th Int. Conf. Informat., Electron. Vis. (ICIEV) 2nd Int. Conf. Imag., Vis. Pattern Recognit. (icIVPR), Jun. 2018, pp. 492–497.

[17] L. Kristoufek, ''BitCoin meets Google trends and wikipedia: Quantifying the relationship between phenomena of the Internet era,'' Sci. Rep., vol. 3, no. 1, p. 3415, Dec. 2013.

[18] J. Patel, S. Shah, P. Thakkar, and K. Kotecha, ''Predicting stock market index using fusion of machine learning techniques,'' Expert Syst. Appl., vol. 42, no. 4, pp. 2162–2172, Mar. 2015.

[19] D. Vasan, M. Alazab, S. Wassan, H. Naeem, B. Safaei, and Q. Zheng, ''IMCFN: Image-based malware classification using fine-tuned convolutional neural network architecture,'' Comput. Netw., vol. 171, Apr. 2020, Art. no. 107138.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)