Machine Learning based Quality Analysis on Inicial Coin Offering (ICO)

R.A.S.I. Cooray
Department of Computing and Information Systems
Faculty of Applied Sciences
Wayamba University of Sri Lanka

W.H.C. Wickramaarachchi
Department of Computing and Information Systems
Faculty of Applied Sciences
Wayamba University of Sri Lanka

ABSTRACT

Today society is a place where people are looking for improvement. Therefore, people are eager to increase their income in simple ways. Crypto currency is an upcoming topic in the world. Therefore, the number of investments on crypto currency are keep increasing. One of the primary ways to earn Crypto is by Initial Coin Offering (ICO). Crypto currency ICO is an easy and modern way to make money. But nowadays, many people are fooled by so many fake ICOs that they are risking their wealth for nothing. This is becoming a major issue and there are so many ongoing researches that are trying to address this problem. The aim of my research is to address this situation in a unique approach. A classifier algorithm will be developed using machine learning techniques and the accuracy of the proposed algorithm will be calculated against some existing approaches. Also, a novel database will be released to the research community as well.

Keywords

ICO, Cryptocurrency, Blockchain, Bitcoin, Factors

1. INTRODUCTION

Crypto ICOs (Initial Coin Offerings) are a set of value propositions that are introduced by companies to find the capital required before they start a project. This is considered to be a good investment. The coin invented by Satoshi Nakamoto, a major part of the Crypto field. It was sold for free in 2008 [8], but at present, we can buy a modern vehicle (Such as Lambogini) using a single Bitcoin [2]. For example, Etherium is just one of those successful crypto ICOs. Firstly, ICO was in 2014, valued at $0.3 per one coin(share). But, today the Etherium is worth about $250. This is just one example. Therefore, this crypto ICO’s investment is a good investment opportunity for investors. However, some fraudulent crypto ICOs, stealing of investor’s money is one of the reasons for the decline in ICO investments. This research looks at the accuracy of the projects that are currently underway or it looks at what similar ICO projects were in the past. This gives investors who do not know about the crypto ICOs, but who are willing to invest in it, this is the opportunity to avoid fake crypto ICO project and invest in good ICO projects. It also intends to encourage investors in a developing country such as Sri Lanka to adapt to the latest investments in the world..

2. LITERATURE REVIEW

2.1 Blockchain technology

All ICO and cryptocurrency-related activities are done through Blockchain technology. Here Blockchain technology removes the middleman between the transactions. In this technology, customers will be given two keys (called Private Key) which are mandatory for completing the transaction. It also has a hashing process to protect the identity of the customer [1].

2.2 Cryptocurrencies

The Bitcoin (BTC) could be the first cryptocurrency in history. The coin was developed by Satoshi Nakamoto in 2008. The opening price of Bitcoin on 1BTC=USD0.06 in 2010. But in 2017 it was 1BTC=USD19, 000. The Bitcoin has shown rapid growth during this period [17]. At the same time, investors tended to invest in cryptocurrency. Today, the cryptocurrency has a market cap of around $ 276 000 000 000 (from Coin Market Cap). So, in the future, all the fiat currency in the world will be cryptocurrency.

2.3 Initial coin offering (ICO)

Initial Coin Offering (ICO) is a company or group of people selling tokens online for a project and raising capital [4]. The first token sale (ICO) was held by Mastercoin in July 2013. Etherium (ETHmarket rank 02) raised money with a token sale in 2014. It was raising $ 2.3 million in its first 12 hours. This system of investment has now spread around the world. Therefore, many fraudulent and unsuccessful projects are still operating in this environment. So, the probability of success of the project is as follows.

ICO consists of three basic forms.

2.3.1 Private sale

Token sale arranged before presale or crowd sale. Not publicly announced and not everyone can participate. Investments are very high. Investors are usually institutional investors or a pool of investors.

2.3.2 Pre ICO

Token sale arranged before the crowd sale. Announced publicly and anyone can participate but the minimum number of investments is significantly higher than during the crowd sale.

2.3.3 Crowd sale

Main sale of an ICO’s tokens. Announced publicly and anyone can participate with even a minimum amount of investments.

The main factors behind the success of ICO areteam,whitepaper,country,purpose,social media,ICO price,pre ICO price,number of tokens,total supply ratio,campaign duration and platform.

2.3.4 Whitepaper

The most important of these is the whitepaper covering all aspects of the project [6]. 89% of all the successful project in the field presented some type of whitepaper [5].

2.3.5 Team

The other important point is the project team [3]. The investors can tell by studying the project team whether an ICO fake or not. Information about the group can be obtained by studying their social media (Linkedin, Twitter).

2.3.6 Pre ICO

Pre ICO has little impact on the ICO. In 100 successful projects, only 33 projects have pre ICO. This is the reason for it [5].

2.3.7 Social media

Social media has a special place here. Project managers should have the ability to quickly distribute whatever happens within the ICO period to investors. Also, a successful project requires a considerable amount of telegram and twitter user.

2.3.8 Platform

It is imperative to be aware of the platform on which the ICO project is running. In crypto history, there are several types of platforms. Most of them run their projects based on Etherium platform [5].

2.3.9 ICO duration

In successful projects, limit their ICO launch period for about one month.

2.3.10 Country

Several countries are launching ICO projects. The USA occupies a prominent place among them. Countries like Singapore, UK, Russia, and Estonia are also on this list [5].

3. LIBRARIES AND TOOLS

3.1 Pandas

Pandas is an open supply Python package that's most generally used for knowledge science/data analysis and machine learning tasks. It's engineered on prime of another package named Numpy, which provides support for multi-dimensional arrays. Jointly of the foremost widespread knowledge haggle packages, Pandas works well with several alternative knowledge science modules within the Python system, and is often enclosed in each Python distribution, from those who go together with your OS to industrial merchandiser distributions like ActiveState's Active Python.

3.2 Numpy

NumPy, which stands for Numerical Python, may be a library consisting of three-dimensional array objects and a set of routines for process those arrays. Exploitation NumPy, mathematical and logical operations on arrays are often performed [11]. NumPy may be a Python package. It stands for 'Numerical Python'.

3.3 Min-max scaler

Transform features by scaling each feature to a given range. This estimator scales and translates each feature individually such that it is in the given range on the training set, e.g. between zero and one [12].

3.4 Mean absolute error

The mean absolute error (MAE) is that the simplest regression error metric to know. We'll calculate the residual for each datum, taking solely absolutely the worth of every so negative and positive residual don't get rid of. We tend to then take the type of these residuals [13].

3.5 Mean squared error

The Mean square Error (MSE) or Mean square Deviation (MSD) of associate figures measures the type of error squares i.e. the typical square distinction between the calculable worth and true value. It's a risky operation, equivalent to the arithmetic mean of the square error loss. It's continuously non – negative and values about to zero square measure higher. The MSE is that the moment of the error (about the origin) and therefore incorporates each the variance of the figurer and its bias [15].

3.6 Keras

Keras could be a powerful and easy-to-use free open supply Python library for developing and evaluating deep learning models. It wraps the economical numerical computation libraries Theano and TensorFlow and permits you to outline and train neural network models in precisely many lines of code [14].

3.7 Pyplot

Pyplot is a Matplotlib module that provides a MATLAB-like interface. every pyplot perform makes some modification to a figure: e.g., creates a figure, creates a plotting space in a very figure, plots some lines in a very plotting space, decorates the plot with labels, etc.

3.8 Neural network

A neural network works equally to the human brain's neural network. A "neuron" during a neural network could be a function that collects and classifies data in keeping with a selected design. The network bears a powerful likeness to applied math strategies like curve fitting and multivariate analysis. A neural network contains layers of interconnected nodes. every node could be a perceptron and is comparable to multiple rectilinear regression. The perceptron feeds the signal created by a multiple rectilinear regression into AN activation perform which will be nonlinear.

In a multi-layered perceptron (MLP), perceptron area unit organized in interconnected layers. The input layer collects input patterns. The output layer has classifications or output signals to that input patterns could map. For example, the patterns could comprise an inventory of quantities for technical indicators a few securities; potential outputs may be "buy," "hold" or "sell."

Hidden layers fine-tune the input weightings until the neural network's margin of error is minimal. It is hypothesized that hidden layers extrapolate salient features in the input data that have predictive power regarding the outputs. This describes feature extraction, which accomplishes a utility similar to statistical techniques such as principal component analysis [16].

3.9 Feed forward neural network

A feedforward neural network may be a biologically impressed classification rule. It contains (possibly large) a range of straightforward neuron-like process units, organized in layers. Each unit during a layer is connected with all the units within the previous layer. These associations don't seem to be all equal: every connection might have a special strength or weight. The weights on these connections inscribe the data of a network. Typically, the units during a neural network also are known as nodes. Data enters at the inputs and passes through the network, layer by layer, till it arrives at the outputs. throughout the traditional operation, that's once it acts as a classifier, there's no feedback between layers. This can be why they're known as feed forward neural networks [7].

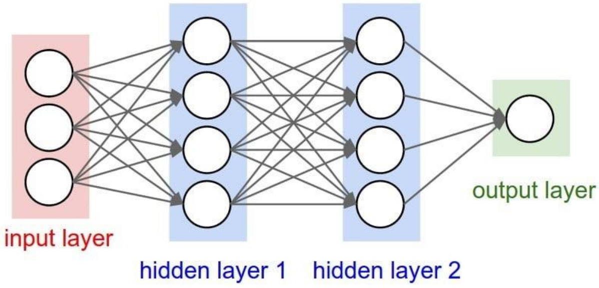


Fig 1: Neural network layered architecture

3.10 Optimizer

During the training process optimizers tie together the loss function and model parameters by updating the model in response to the output of the loss function. Optimizer's shape and meld the model into its most accurate possible form.

3.11 Adam

The adaptive moment estimate combines the power ofRMSProp (root-mean-square prop) and momentum-based GD. In Adam Optimization, the power of momentum GDs and the adaptive learning rate provided by RMSProp make Adam Optimization a powerful way to keep track of updates history. It also introduces two new hyper-parameters, beta 1 and beta 2, which are usually kept between 0.9 and 0.99 but can be changed depending on your usage.

4. METHODOLOGY

4.1 Data set

The dataset that is used here is built by mine. At that time, that relevant data was found using five websites [9][10][18][19][20]. The reason for using five websites was to verify the accuracy of the data. And also, the amount of available Twitter, Facebook, LinkedIn followers in here are those found from each social media.

4.2 Model

Some information of the taken data collection was not necessary to form the mode. So, those unnecessary columns were deleted as the first step. For example, the columns loke 'LinkedIn followers' and 'Platform' always take the same values. So, they were deleted because they were not necessary to form the model. Fig. 2 shows how the data collection wasinserted.

There are two variable such as data and target as shown in the fig. 3. Here, the variable named 'data' contains the columns numbered from one to eleven in the data collection. They are Team_members, Advisors, ICO, Duration, Softcap, Hardcap, Twitter, Telegram, Total supply and Available. In addition to that, the variable named 'target' contains the column rating, expected as the output column.

After that, all the values in this data collection should be taken into values between 0 and 1. The reason for it is, some of the values here are very larger numbers and some of them are very smaller numbers. For example, the values of column ICO are represented as decimal numbers and the values of the column Total_supply is represented as very larger numbers. When doing calculations, it is very difficult because those smaller numbers turn into more smaller ones and those larger numbers turn into more larger ones. This becomes a problemin receiving the accuracy. Here, the min-max scaler is used to take all those numbers into a value between 0 and 1.

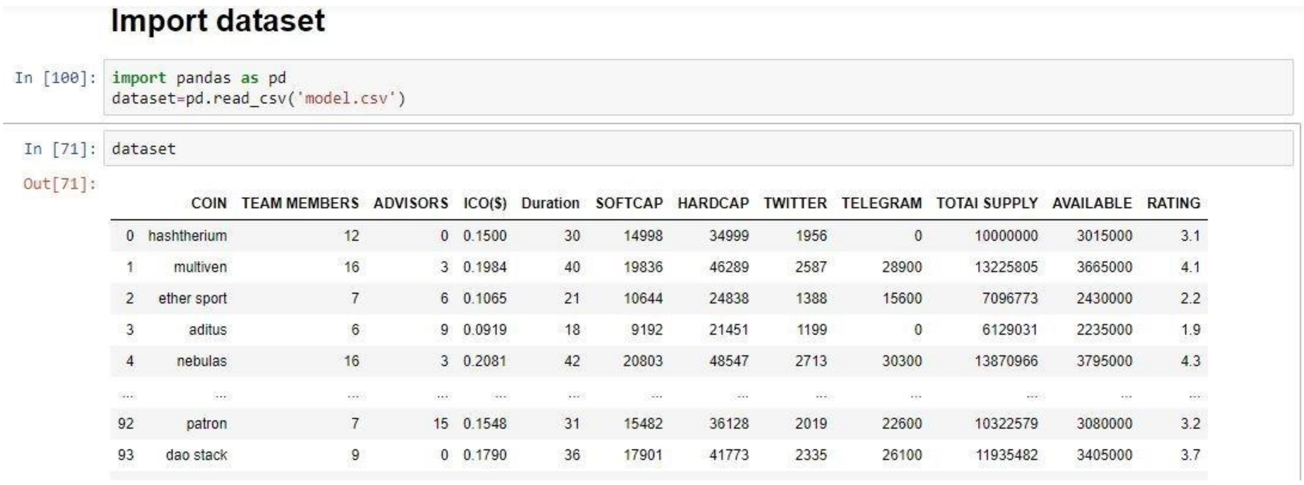
After that, this data collection is divided into two parts. They are named as train data collection and test data collection. When taken as percentages, 85% is given to train data and 15% is given to test data. In this case, train data is being used to train the model and the test is used to test the trained model. Here, the process of training the data collection is done by using the neural network.

The activity done by a neural network is, training the data collection by doing calculation until the parameters in hidden layers come to an optimal value, after giving the input and output. Here, the hidden layers mean the parameters formed by the neural network with the expectation to receive the output after giving an input.Two hidden layers are used here. Then, the model is being trained until a minimal cost (into an optimal value). An optimize is used for this purpose.

The activity done by using an optimize is searching for the instance where a minimum lost is reported. This instance comes after the model has gone through the neural network 90 times. Then, we can have the accuracy (r2) value here. That means we can have the neural network model after this process.

4.3 Model testing

This model can be tested using test data. In addition, the value receive as the output here is always between 0 and 1. The reason for it is, All the numbers are converted into value between 0 and 1 using a min-max scaler, in a step mentioned above. So, the value shown as the output now is not equal to the value represented in column 'rate' in the data collection. So, the value we find now as the output should be converted again into the accurate value with the aid of an inverse transform.



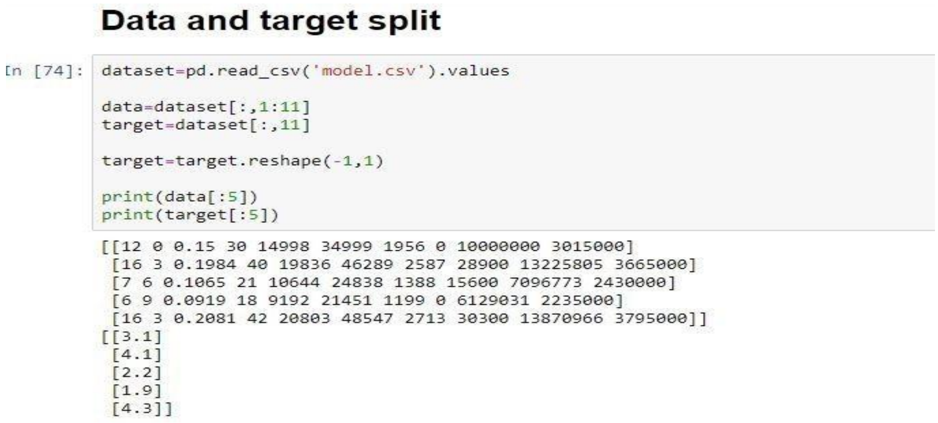
Import dataset

```
In [100]: import pandas as pd
          dataset=pd.read_csv('model.csv')
```

```
In [71]: dataset
Out[71]:
```

	COIN	TEAM MEMBERS	ADVISORS	ICO($)	Duration	SOFTCAP	HARDCAP	TWITTER	TELEGRAM	TOTAl SUPPLY	AVAILABLE	RATING
0	hashtherium	12	0	0.1500	30	14998	34999	1956	0	10000000	3015000	3.1
1	multiven	16	3	0.1984	40	19836	46289	2587	28900	13225805	3665000	4.1
2	ether sport	7	6	0.1065	21	10644	24838	1388	15600	7096773	2430000	2.2
3	aditus	6	9	0.0919	18	9192	21451	1199	0	6129031	2235000	1.9
4	nebulas	16	3	0.2081	42	20803	48547	2713	30300	13870966	3795000	4.3
...	...	...	...	...	...	...	...	...	...	...	...	...
92	patron	7	15	0.1548	31	15482	36128	2019	22600	10322579	3080000	3.2
93	dao stack	9	0	0.1790	36	17901	41773	2335	26100	11935482	3405000	3.7

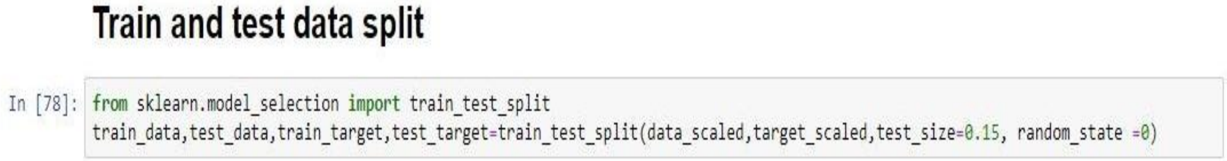
Fig 2: Import data set



Data and target split

```
In [74]: dataset=pd.read_csv('model.csv').values

         data=dataset[:,1:11]
         target=dataset[:,11]

         target=target.reshape(-1,1)

         print(data[:5])
         print(target[:5])

[[12 0 0.15 30 14998 34999 1956 0 10000000 3015000]
 [16 3 0.1984 40 19836 46289 2587 28900 13225805 3665000]
 [7 6 0.1065 21 10644 24838 1388 15600 7096773 2430000]
 [6 9 0.0919 18 9192 21451 1199 0 6129031 2235000]
 [16 3 0.2081 42 20803 48547 2713 30300 13870966 3795000]]
[[3.1]
 [4.1]
 [2.2]
 [1.9]
 [4.3]]
```

Fig 3:Data and target split



Train and test data split

```
In [78]: from sklearn.model_selection import train_test_split
         train_data,test_data,train_target,test_target=train_test_split(data_scaled,target_scaled,test_size=0.15, random_state =0)
```

Fig 4:Train and test data split

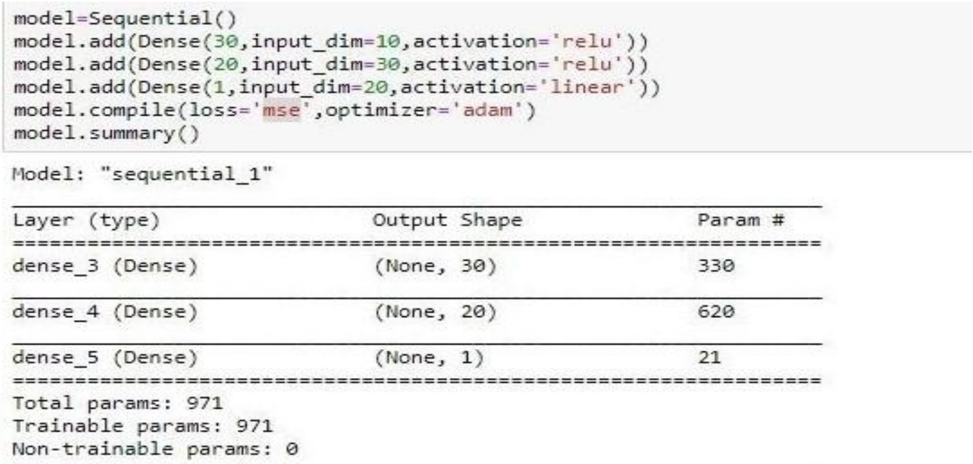
In addition, when a user is inserting data into the model by using an ICO preferred by him/her, all the values that have been input are converted into values between 0 and 1. Then the model predicts. It gives us the output rating and the names of former ICO relevant to the specific rating.

5. RESULTS

5.1 Model training

Data such as Team member, advisors, ICO, duration, soft cap, hard cap, twitter, telegram, total supply, available are given as input to the neural network to train the model. Since 85% of the total database is used for the training process, all the data taken for the training is entered as above. In order to build the relationship between the given input and output values, the optimal values must come from the parameters in the hidden layers.

Two hidden layers are used. 30 parameters are used in the first hidden layer and parameters are used in the second hidden layer. (fig. 5) Thus the relevant parameter size is determined by the parameter size in the hidden layer when the model has maximum accuracy. Therefore, the accuracy should be found by changing the parameter count from time to time.An activation function must be used. It decides which of the neurons in the hidden layer should be activated or not. Therefore, Relu is used here as the activation function. The reason it was used was because of all the values (+) in the data set. Linear is used as the activation function in the output. Because the usually output is used, hence linear was used. As seen in the methodology above, the loss and epchos obtained by optimization (epchos is the number of times the data set is insert into the neural network, where it is equal to 90 times).



```
model=Sequential()
model.add(Dense(30,input_dim=10,activation='relu'))
model.add(Dense(20,input_dim=30,activation='relu'))
model.add(Dense(1,input_dim=20,activation='linear'))
model.compile(loss='mse',optimizer='adam')
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_3 (Dense)              (None, 30)                330
_________________________________________________________________
dense_4 (Dense)              (None, 20)                620
_________________________________________________________________
dense_5 (Dense)              (None, 1)                 21
=================================================================
Total params: 971
Trainable params: 971
Non-trainable params: 0
```

Fig 5:Neural network model

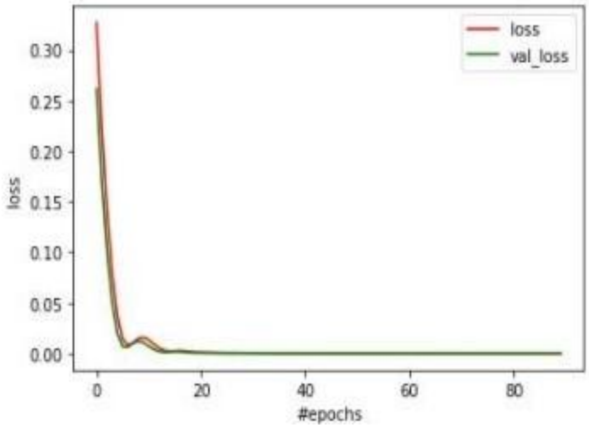


Fig 6:Model training chart

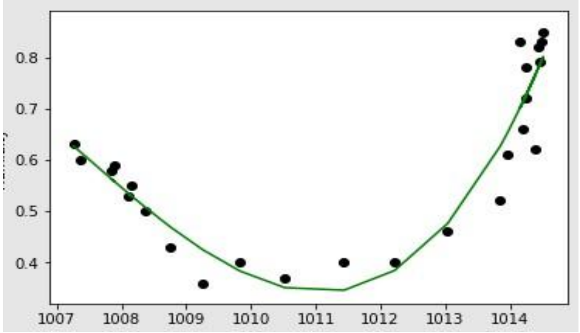


Fig 7: Over flitting

The train data is inserted 90 times into the neural network model and the minimum loss is obtained evenly. The above graph (fig. 6) shows us that the loss is not over flit.

Over flitting is in fig. 7 when the data set is trained and the loss decreases and starts increasing again without reaching a minimum value. Then a minimum loss cannot be obtained. But that problem does not come here because the loss has come to an approximate value and is uniform.

Even if there is an over flit, you can look at the values in the graph and add the epoches value to the epoches value of the model training part when the loss value is added through those values to prevent over flit. That is why the epoches value is equal to 90.

After the above process we get the neural network model. The parameter size used is as follows.

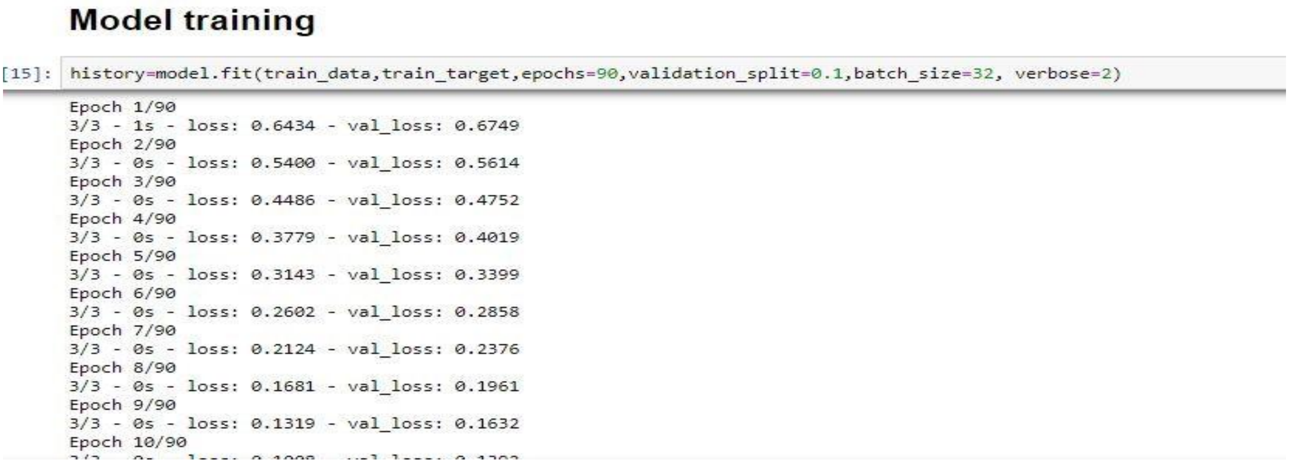
- Input layer 10
- Hidden layer1 30
- Hidden layer2 20
- Output layer 1

5.2 Model testing

The accuracy is checked using the amount of data used to test (15% of the data set). Where r2 is the value. This r2 value is obtained based on the test data, the output value given as input and the actual output value of the test data set. Here r2 is 0.90117 (90.117%). This is a very good accuracy. Also, the MSE (mean squared error) value is 0.03331 which is a very small value.

The actual target is the current rating of the test data set. Predicted targets are the rating value that comes when sending test data through a trained model. Now the training process is over.

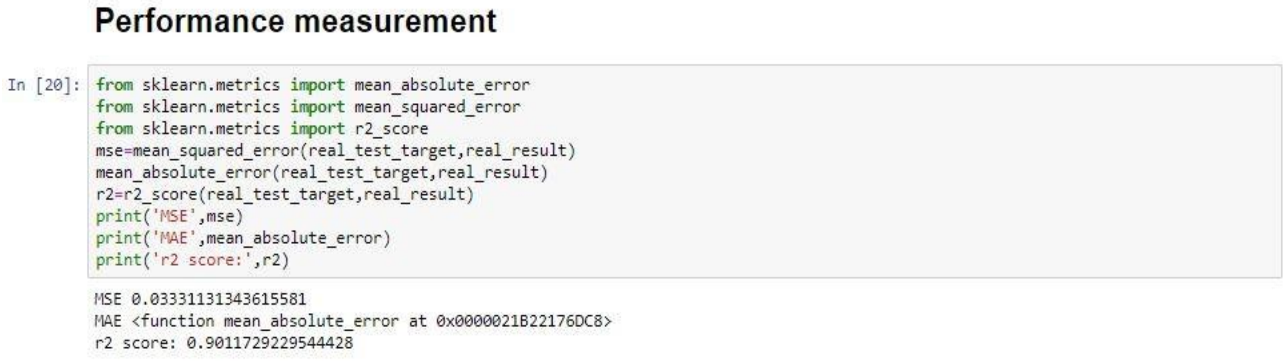
Fig. 10 shows the actual rating values of the test data and the values derived from the model. They are very similar to each other because the above accuracy was very good. Also, the rating values of this fig. 10 are not the actual rating values. That is, since all the values were initially transformed into a 0 - 1 range, here are the values that are in the 0 - 1 range of the actual rating value. Therefore, the inverse-transform is used to bring them back to their original values like fig. 11 illustrates.



Model training

```
[15]: history=model.fit(train_data,train_target,epochs=90,validation_split=0.1,batch_size=32, verbose=2)

Epoch 1/90
3/3 - 1s - loss: 0.6434 - val_loss: 0.6749
Epoch 2/90
3/3 - 0s - loss: 0.5400 - val_loss: 0.5614
Epoch 3/90
3/3 - 0s - loss: 0.4486 - val_loss: 0.4752
Epoch 4/90
3/3 - 0s - loss: 0.3779 - val_loss: 0.4019
Epoch 5/90
3/3 - 0s - loss: 0.3143 - val_loss: 0.3399
Epoch 6/90
3/3 - 0s - loss: 0.2602 - val_loss: 0.2858
Epoch 7/90
3/3 - 0s - loss: 0.2124 - val_loss: 0.2376
Epoch 8/90
3/3 - 0s - loss: 0.1681 - val_loss: 0.1961
Epoch 9/90
3/3 - 0s - loss: 0.1319 - val_loss: 0.1632
Epoch 10/90
```

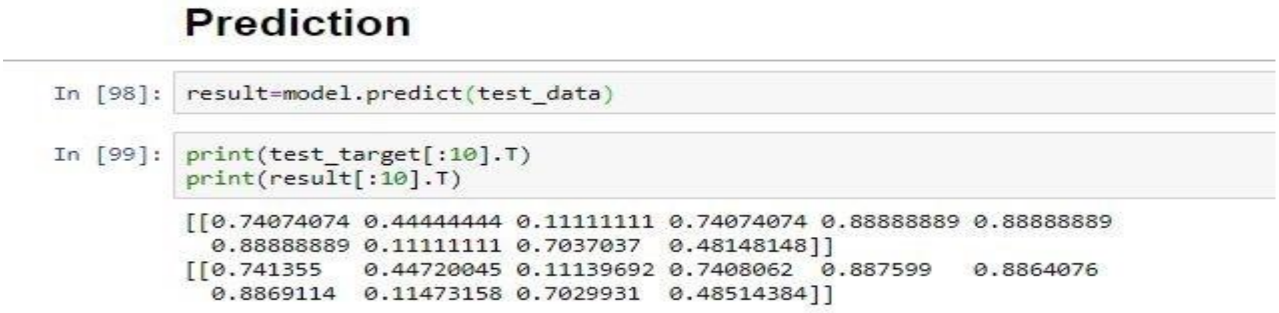
Fig 8: Model training



Performance measurement

```
In [20]: from sklearn.metrics import mean_absolute_error
         from sklearn.metrics import mean_squared_error
         from sklearn.metrics import r2_score
         mse=mean_squared_error(real_test_target,real_result)
         mean_absolute_error(real_test_target,real_result)
         r2=r2_score(real_test_target,real_result)
         print('MSE',mse)
         print('MAE',mean_absolute_error)
         print('r2 score:',r2)

         MSE 0.03331131343615581
         MAE <function mean_absolute_error at 0x0000021B22176DC8>
         r2 score: 0.9011729229544428
```

Fig 9:Performance measurement



Prediction

```
In [98]: result=model.predict(test_data)

In [99]: print(test_target[:10].T)
         print(result[:10].T)

         [[0.74074074 0.44444444 0.11111111 0.74074074 0.88888889 0.88888889
           0.88888889 0.11111111 0.7037037  0.48148148]]
         [[0.741355   0.44720045 0.11139692 0.7408062  0.887599   0.8864076
           0.8869114  0.11473158 0.7029931  0.48514384]]
```

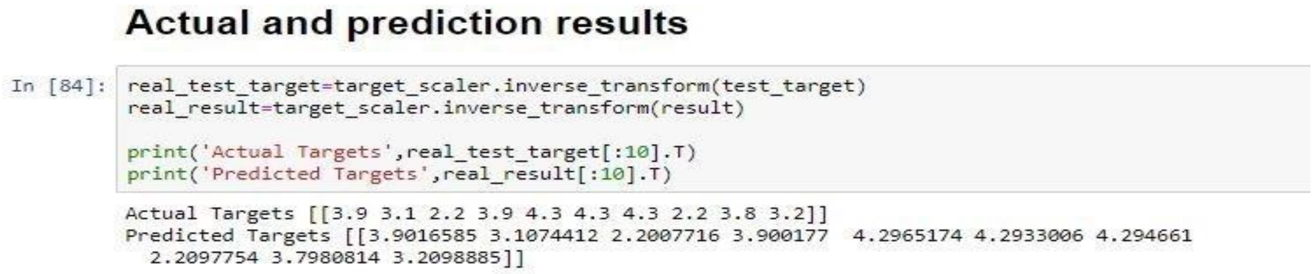
Fig 10: Prediction part

5.3 User input

Now we have the process of selecting the ICO that matches the new ICO from the existing ICOs after entering the data for the new ICO. Here all the input values of the user and those input values must be in the range of 0-1 in order to get an output predicted by the model. This can be done using the MIN-MAX scaler. The value obtained from the MIN-MAX scaler is obtained using the following formula.

Min-max scaler value = (data – min)/(max – min) --------(1)

Data - The value to be entered
Min - The minimum value of the column to which the value is entered
Max - The maximum value of the column to which the value is entered
Similarly, the output value for the above data is between 0-1. It should also be converted to real value as follows.



Actual and prediction results

```
In [84]: real_test_target=target_scaler.inverse_transform(test_target)
         real_result=target_scaler.inverse_transform(result)

         print('Actual Targets',real_test_target[:10].T)
         print('Predicted Targets',real_result[:10].T)

Actual Targets [[3.9 3.1 2.2 3.9 4.3 4.3 4.3 2.2 3.8 3.2]]
Predicted Targets [[3.9016585 3.1074412 2.2007716 3.900177  4.2965174 4.2933006 4.294661
  2.2097754 3.7980814 3.2098885]]
```

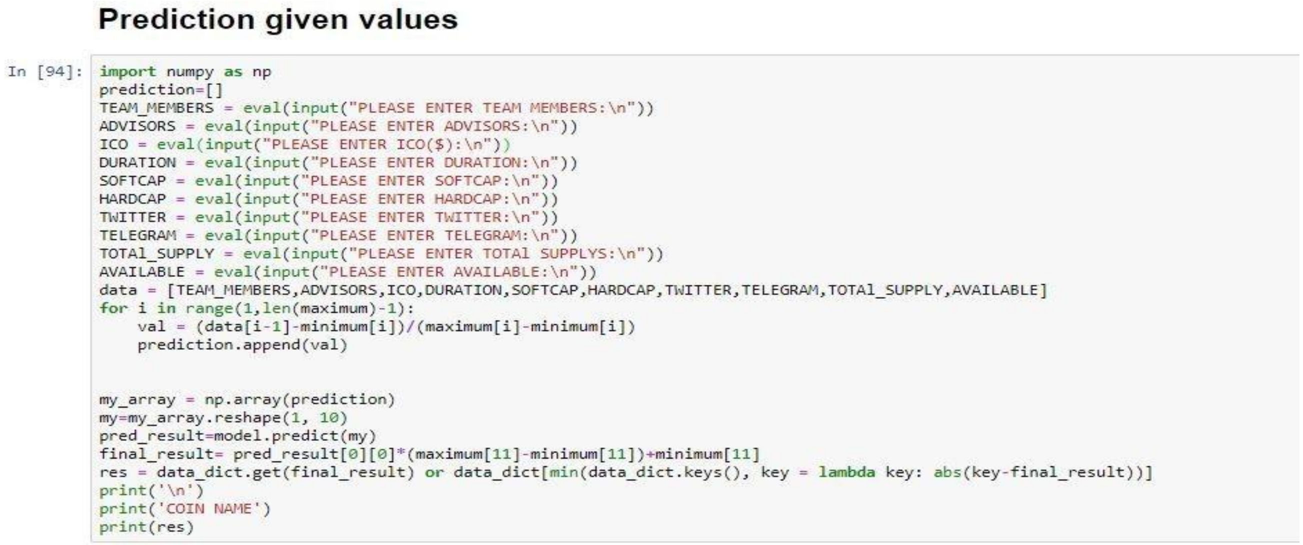
Fig 11:Actual and prediction results

Data= min_max scaler value(max-min)+ min-----(2)

Data- The value to be entered
Min- The minimum value of the column to which the value is entered
Max- The maximum value of the column to which the value is entered

All of the above calculations are shown in Fig. 12.

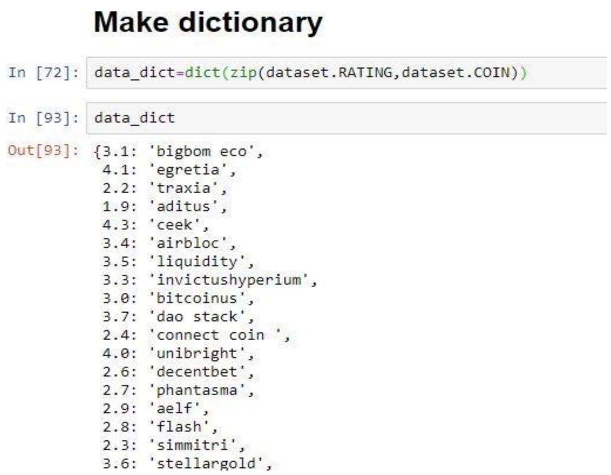
The main objective of this research is to provide an ICO that matches the rating value received from previous ICOs. Therefore, in order to get the ICO related to this rating, the rating column and the coin name column of the data set have been inserted into a dictionary as fig. 13 below.



Prediction given values

```
In [94]: import numpy as np
         prediction=[]
         TEAM_MEMBERS = eval(input("PLEASE ENTER TEAM MEMBERS:\n"))
         ADVISORS = eval(input("PLEASE ENTER ADVISORS:\n"))
         ICO = eval(input("PLEASE ENTER ICO($):\n"))
         DURATION = eval(input("PLEASE ENTER DURATION:\n"))
         SOFTCAP = eval(input("PLEASE ENTER SOFTCAP:\n"))
         HARDCAP = eval(input("PLEASE ENTER HARDCAP:\n"))
         TWITTER = eval(input("PLEASE ENTER TWITTER:\n"))
         TELEGRAM = eval(input("PLEASE ENTER TELEGRAM:\n"))
         TOTAl_SUPPLY = eval(input("PLEASE ENTER TOTAl SUPPLYS:\n"))
         AVAILABLE = eval(input("PLEASE ENTER AVAILABLE:\n"))
         data = [TEAM_MEMBERS,ADVISORS,ICO,DURATION,SOFTCAP,HARDCAP,TWITTER,TELEGRAM,TOTAl_SUPPLY,AVAILABLE]
         for i in range(1,len(maximum)-1):
             val = (data[i-1]-minimum[i])/(maximum[i]-minimum[i])
             prediction.append(val)


         my_array = np.array(prediction)
         my=my_array.reshape(1, 10)
         pred_result=model.predict(my)
         final_result= pred_result[0][0]*(maximum[11]-minimum[11])+minimum[11]
         res = data_dict.get(final_result) or data_dict[min(data_dict.keys(), key = lambda key: abs(key-final_result))]
         print('\n')
         print('COIN NAME')
         print(res)
```

Fig 12: Prediction given values

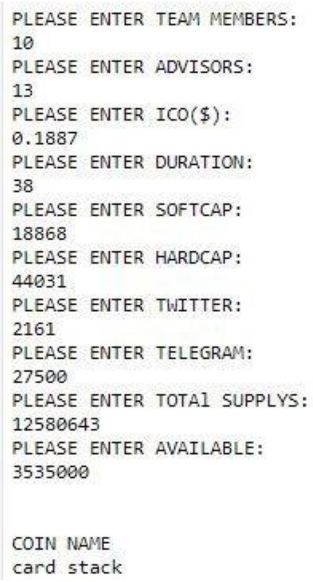


Make dictionary

```
In [72]: data_dict=dict(zip(dataset.RATING,dataset.COIN))

In [93]: data_dict

Out[93]: {3.1: 'bigbom eco',
          4.1: 'egretia',
          2.2: 'traxia',
          1.9: 'aditus',
          4.3: 'ceek',
          3.4: 'airbloc',
          3.5: 'liquidity',
          3.3: 'invictushyperium',
          3.0: 'bitcoinus',
          3.7: 'dao stack',
          2.4: 'connect coin ',
          4.0: 'unibright',
          2.6: 'decentbet',
          2.7: 'phantasma',
          2.9: 'aelf',
          2.8: 'flash',
          2.3: 'simmitri',
          3.6: 'stellargold',
```

Fig 13: Make dictionary

This built-in dictionary is called in the code shown in fig. 12 and we can get the old most suitable ICO name for the relevant rating.



```
PLEASE ENTER TEAM MEMBERS:
10
PLEASE ENTER ADVISORS:
13
PLEASE ENTER ICO($):
0.1887
PLEASE ENTER DURATION:
38
PLEASE ENTER SOFTCAP:
18868
PLEASE ENTER HARDCAP:
44031
PLEASE ENTER TWITTER:
2161
PLEASE ENTER TELEGRAM:
27500
PLEASE ENTER TOTAl SUPPLYS:
12580643
PLEASE ENTER AVAILABLE:
3535000


COIN NAME
card stack
```

Fig 14: Inputs and outputs

6. DISCUSSION AND CONCLUSION

The new dataset used to create this model. It had to be used because of problem with the accuracy of the data. Because the information about some ICO was differ from each site. That is because they contained contradictory information. Here a new dataset had to create to solve the above problem using five web sites. It took a lot of time. In conflicting information in these five websites, the information contained in the maximum number of websites out of those five websites was included in the dataset.

In addition to these factors, many other factors can be used for this. For example, politics, economics can be introduced. Factors such as hardcap, softcap, duration have been used, as data can be obtained with good data accuracy and a good mathematical background. The reason for using the feed forward neural network in the neural network is that even though several types of network are used, the best accuracy was obtained from this type of network.

The main objective of this research is to promote ICO investment in Sri Lanka. Therefore, in the future, hope to build a website using the research data and API. This will enable even those who are not familiar with ICO and crypto to get a good investment opportunity.

7. ACKNOWLEDGMENT

I wish to express my sincere appreciation to my supervisor, Mrs. W.H.C.Wickramaarachchi, who has the substance of a genius: she convincingly guided and encouraged me to be professional and do the right thing even when the road got tough. Without his persistent help, the goal of this project would not have been realized.

Besides my supervisor, I would like to thank the rest my thesis committee: Sr.Prof.N.G.J.Dias, Mrs W.H.C. Wickramaarachchi, Department Head, Dt. V.G. T.N. Vidanagama, Mr. T. Arudchelvam, for their encouragement, insightful comments, and hard questions.

Special Thanks to Mr. P.M.N.S.T.Dharmarathna(UOM - Electronics and Telecommunications Engineering) for being a part of the introducing member for blockchain and Cryptocurrencies.

Last but not the least I wish to acknowledge the support and great love of my family. They kept me going on and this work would not have been possible without their input.

8. REFERENCES

[1] Panin, A., 2019. A multiple case study on factors affecting ICO success.

[2] THALAENGKIJ, M.R., 2017. FACTORS DETERMINING SUCCESSFUL ICO (Doctoral dissertation, THAMMASAT UNIVERSITY).

[3] Burns, L. and Moro, A., 2018. What makes an ICO successful? An investigation of the role of ICO characteristics, team quality and market sentiment. An Investigation of the Role of ICO Characteristics, Team Quality and Market Sentiment (September 27, 2018).

[4] Dean, T., Daluwathumullagamage, D. J., & Marsden, A. (2020). Predictability of ICO success and returns. *Journal of Applied Business and Economics*, *22*(13), 20-36.

[5] https://www.academia.edu/38803810/Factors_Which_Affect_ICOs_Success.

[6] https://www.investopedia.com/tech/how-identify-cryptocurrency-and-ico-scams/.

[7] https://deepai.org/machine-learning-glossary-and-terms/feed-forward-neural-network.

[8] https://whitepaper.io/document/5/ethereum-whitepaper.

[9] https://icomarks.com/ico/tradeplace

[10] https://icoholder.com/en/icos/past

[11] https://www.tutorialspoint.com/numpy/index.htm

[12] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.minmax_scale.html

[13] https://www.dataquest.io/blog/understanding-regression-error-metrics/

[14] https://machinelearningmastery.com/tutorial-first-neural-network-python-keras/

[15] https://www.geeksforgeeks.org/python-mean-squared-error/

[16] https://www.investopedia.com/terms/n/neuralnetwork.asp

[17] https://en.wikipedia.org/wiki/History_of_bitcoin

[18] https://icodrops.com/

[19] https://icobench.com/

[20] https://coinmarketcap.com/