LiteCrypto

March 30, 2022

By Marwan Khalil

1 IBM Time Series and Survival Analysis Capstone Project

1.1 Introduction

Litecoin is a peer-to-peer Internet currency that enables instant, near-zero cost payments to anyone in the world. Litecoin is an open source, global payment network that is fully decentralized without any central authorities. Mathematics secures the network and empowers individuals to control their own finances. Litecoin features faster transaction confirmation times and improved storage efficiency than the leading math-based currency. With substantial industry support, trade volume and liquidity, Litecoin is a proven medium of commerce complementary to Bitcoin.

This notebook demonstrates the prediction of the litecoin price by the neural network model. We are using both long short term memory (LSTM) and recurrent neural network (RNN) algorithms, to find the one that suits the problem better.

1.2 Dataset

Litecoin cryptocurrency data were retrieved from Yahoo Finance

- Date: date of observation
- Open: Opening price on the given day
- High: Highest price on the given day
- Close: Closing price on the given day
- Adjusted Close: Is the closing price after adjustments for all applicable splits and dividend distributions
- Volume: Volume of transactions on the given date

[1]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

[2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import keras
from sklearn.preprocessing import MinMaxScaler

%matplotlib inline
```

[3]:
```
Litecoin = pd.read_csv('/content/drive/MyDrive/LTC-USD.csv')
```

1.3 Data preparation

[4]:
```
Litecoin.tail()
```

[4]:
```
             Date        Open        High  ...       Close   Adj Close  
Volume
1822  2021-05-02  276.960419  277.483459  ...  269.104370  269.104370
3.118501e+09
1823  2021-05-03  269.008301  299.300537  ...  294.704010  294.704010
5.172505e+09
1824  2021-05-04  294.774261  326.888672  ...  306.234497  306.234497
1.154125e+10
1825  2021-05-05  305.177399  359.500153  ...  356.037079  356.037079
1.318354e+10
1826  2021-05-06  357.893585  362.496399  ...  344.669098  344.669098
9.519699e+09

[5 rows x 7 columns]
```

[5]:
```
Litecoin.isnull().values.any()
```

[5]: True

[6]:
```
Litecoin.isnull().sum().sum()
```

[6]: 24

[7]:
```
Litecoin[Litecoin.isnull().any(axis=1)]
```

[7]:
```
             Date  Open  High  Low  Close  Adj Close  Volume
1442  2020-04-17   NaN   NaN  NaN    NaN        NaN     NaN
1617  2020-10-09   NaN   NaN  NaN    NaN        NaN     NaN
1620  2020-10-12   NaN   NaN  NaN    NaN        NaN     NaN
1621  2020-10-13   NaN   NaN  NaN    NaN        NaN     NaN
```

[8]:
```
Litecoin['Open'] = Litecoin['Open'].interpolate()
Litecoin['High'] = Litecoin['High'].interpolate()
Litecoin['Low'] = Litecoin['Low'].interpolate()
Litecoin['Close'] = Litecoin['Close'].interpolate()
Litecoin['Adj Close'] = Litecoin['Adj Close'].interpolate()
Litecoin['Volume'] = Litecoin['Volume'].interpolate()
```

[9]:
```
Litecoin.isnull().sum().sum()
```

[9]: 0

1.3.1 Dividing the dataset into training and testing splits

[10]:
```
lite_train = Litecoin[Litecoin['Date']< '2021-01-01'].copy()
lite_train
```

[10]:
```
            Date        Open        High  ...       Close   Adj Close
Volume
0     2016-05-06    3.716130    3.822110  ...    3.822110    3.822110
1.899060e+06
1     2016-05-07    3.819520    4.003440  ...    3.953450    3.953450
3.278610e+06
2     2016-05-08    3.936410    3.944110  ...    3.938140    3.938140
1.783970e+06
3     2016-05-09    3.937630    4.083380  ...    4.056430    4.056430
3.120940e+06
4     2016-05-10    4.057970    4.115630  ...    3.819150    3.819150
3.688200e+06
…          …          …          …  …          …          …
…
1696  2020-12-27  129.456619  138.319717  ...  127.516968  127.516968
1.410331e+10
1697  2020-12-28  127.588303  136.185074  ...  130.050339  130.050339
1.024873e+10
1698  2020-12-29  130.033264  130.608582  ...  129.040802  129.040802
9.160551e+09
1699  2020-12-30  129.061859  132.450119  ...  129.466080  129.466080
8.127317e+09
1700  2020-12-31  129.480286  130.166245  ...  124.690323  124.690323
6.274573e+09

[1701 rows x 7 columns]
```

[11]:
```
lite_test = Litecoin[Litecoin['Date'] > '2021-01-01'].copy()
lite_test
```

[11]:
```
            Date        Open        High  ...       Close   Adj Close
Volume
1702  2021-01-02  126.272964  140.372574  ...  136.944885  136.944885
1.053207e+10
1703  2021-01-03  136.949402  163.898636  ...  160.190582  160.190582
1.538566e+10
1704  2021-01-04  160.271164  173.027817  ...  154.807327  154.807327
1.365979e+10
1705  2021-01-05  154.897552  162.850189  ...  158.594772  158.594772
1.019282e+10
1706  2021-01-06  158.665970  169.657455  ...  169.016922  169.016922
1.074388e+10
…          …          …          …  …          …          …
```

```
…
1822  2021-05-02  276.960419  277.483459  …  269.104370  269.104370
3.118501e+09
1823  2021-05-03  269.008301  299.300537  …  294.704010  294.704010
5.172505e+09
1824  2021-05-04  294.774261  326.888672  …  306.234497  306.234497
1.154125e+10
1825  2021-05-05  305.177399  359.500153  …  356.037079  356.037079
1.318354e+10
1826  2021-05-06  357.893585  362.496399  …  344.669098  344.669098
9.519699e+09

[125 rows x 7 columns]
```

[12]:
```
train_lt = lite_train.drop(['Date', 'Adj Close'], axis = 1)
train_lt.head()
```

[12]:
```
      Open     High      Low    Close     Volume
0  3.71613  3.82211  3.70600  3.82211  1899060.0
1  3.81952  4.00344  3.81952  3.95345  3278610.0
2  3.93641  3.94411  3.88620  3.93814  1783970.0
3  3.93763  4.08338  3.89863  4.05643  3120940.0
4  4.05797  4.11563  3.79838  3.81915  3688200.0
```

[13]:
```
scaler = MinMaxScaler(feature_range= (0,1))
train_lt = scaler.fit_transform(train_lt)
train_lt
```

[13]:
```
array([[6.38518125e-04, 7.11245300e-04, 1.13793655e-03, 8.27293862e-04,
         8.77429683e-05],
        [9.29236766e-04, 1.19904783e-03, 1.47950707e-03, 1.19746651e-03,
         1.85565038e-04],
        [1.25791557e-03, 1.03944200e-03, 1.68014064e-03, 1.15431634e-03,
         7.95820883e-05],
        …,
        [3.55825146e-01, 3.41784215e-01, 3.54627329e-01, 3.53747456e-01,
         6.49515689e-01],
        [3.53093688e-01, 3.46738201e-01, 3.61418847e-01, 3.54946073e-01,
         5.76250431e-01],
        [3.54270247e-01, 3.40594268e-01, 3.60191637e-01, 3.41485934e-01,
         4.44874784e-01]])
```

[14]:
```
X_train = []
y_train = []

train_lt.shape
```

[14]: (1701, 5)

[15]:
```
for i in range(60, train_lt.shape[0]):
    X_train.append(train_lt[i-60:i])
    y_train.append(train_lt[i,0])
```

[16]:
```
X_train, y_train = np.array(X_train), np.array(y_train)
```

[17]:
```
X_train.shape
```

[17]: (1641, 60, 5)

[18]:
```
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Dense, LSTM, Dropout
```

1.4 LSTM model

[19]:
```
regressor = Sequential()
regressor.add(LSTM(units = 50, activation = 'relu', return_sequences = True,␣
↪input_shape = (X_train.shape[1], 5)))
regressor.add(Dropout(0.2))
```

```
WARNING:tensorflow:Layer lstm will not use cuDNN kernel since it doesn't meet
the cuDNN kernel criteria. It will use generic GPU kernel as fallback when
running on GPU
```

[20]:
```
regressor.add(LSTM(units = 60, activation = 'relu', return_sequences = True))
regressor.add(Dropout(0.3))

regressor.add(LSTM(units = 80, activation = 'relu', return_sequences = True))
regressor.add(Dropout(0.4))

regressor.add(LSTM(units = 120, activation = 'relu'))
regressor.add(Dropout(0.5))

regressor.add(Dense(units =1))
```

```
WARNING:tensorflow:Layer lstm_1 will not use cuDNN kernel since it doesn't meet
the cuDNN kernel criteria. It will use generic GPU kernel as fallback when
running on GPU
WARNING:tensorflow:Layer lstm_2 will not use cuDNN kernel since it doesn't meet
the cuDNN kernel criteria. It will use generic GPU kernel as fallback when
running on GPU
WARNING:tensorflow:Layer lstm_3 will not use cuDNN kernel since it doesn't meet
the cuDNN kernel criteria. It will use generic GPU kernel as fallback when
running on GPU
```

[21]:
```
regressor.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 60, 50)            11200
_________________________________________________________________
dropout (Dropout)            (None, 60, 50)            0
_________________________________________________________________
lstm_1 (LSTM)                (None, 60, 60)            26640
_________________________________________________________________
dropout_1 (Dropout)          (None, 60, 60)            0
_________________________________________________________________
lstm_2 (LSTM)                (None, 60, 80)            45120
_________________________________________________________________
dropout_2 (Dropout)          (None, 60, 80)            0
_________________________________________________________________
lstm_3 (LSTM)                (None, 120)               96480
_________________________________________________________________
dropout_3 (Dropout)          (None, 120)               0
_________________________________________________________________
dense (Dense)                (None, 1)                 121
=================================================================
Total params: 179,561
Trainable params: 179,561
Non-trainable params: 0
_________________________________________________________________
```

[22]:

```
from keras import optimizers

optimizer = optimizers.Adam(clipvalue=0.5)

regressor.compile(optimizer = optimizer, loss = 'mean_squared_error')
```

[23]:

```
regressor.fit(X_train, y_train, epochs = 20, batch_size =32)
```

```
Epoch 1/20
52/52 [==============================] - 28s 454ms/step - loss: 0.0213
Epoch 2/20
52/52 [==============================] - 24s 462ms/step - loss: 0.0058
Epoch 3/20
52/52 [==============================] - 24s 455ms/step - loss: 0.0067
Epoch 4/20
52/52 [==============================] - 23s 451ms/step - loss: 0.0040
Epoch 5/20
52/52 [==============================] - 24s 456ms/step - loss: 0.0031
Epoch 6/20
52/52 [==============================] - 24s 456ms/step - loss: 0.0033
Epoch 7/20
```

```
52/52 [==============================] - 23s 450ms/step - loss: 0.0032
Epoch 8/20
52/52 [==============================] - 24s 457ms/step - loss: 0.0027
Epoch 9/20
52/52 [==============================] - 24s 452ms/step - loss: 0.0022
Epoch 10/20
52/52 [==============================] - 23s 449ms/step - loss: 0.0022
Epoch 11/20
52/52 [==============================] - 23s 450ms/step - loss: 0.0022
Epoch 12/20
52/52 [==============================] - 23s 450ms/step - loss: 0.0021
Epoch 13/20
52/52 [==============================] - 23s 448ms/step - loss: 0.0019
Epoch 14/20
52/52 [==============================] - 23s 451ms/step - loss: 0.0022
Epoch 15/20
52/52 [==============================] - 24s 453ms/step - loss: 0.0021
Epoch 16/20
52/52 [==============================] - 23s 449ms/step - loss: 0.0026
Epoch 17/20
52/52 [==============================] - 24s 453ms/step - loss: 0.0021
Epoch 18/20
52/52 [==============================] - 23s 451ms/step - loss: 0.0021
Epoch 19/20
52/52 [==============================] - 23s 443ms/step - loss: 0.0021
Epoch 20/20
52/52 [==============================] - 24s 452ms/step - loss: 0.0019
```

[23]:
```
<tensorflow.python.keras.callbacks.History at 0x7f88401ed710>
```

[24]:
```
past_60_days = lite_train.tail(60)
lt= past_60_days.append(lite_test, ignore_index = True)
lt = lt.drop(['Date', 'Adj Close'], axis = 1)
lt.head()
```

[24]:

	Open	High	Low	Close	Volume
0	55.587322	56.710835	53.205944	53.817482	3.019890e+09
1	53.817410	54.272400	51.648445	53.819622	2.580301e+09
2	53.819469	55.007088	51.606560	54.499104	3.050534e+09
3	54.501873	59.223793	54.501873	58.678497	3.250515e+09
4	58.676193	63.280354	58.465351	63.131111	4.083503e+09

[25]:
```
inputs = scaler.transform(lt)
inputs
```

[25]:
```
array([[0.14649327, 0.14298916, 0.1500784 , 0.14173577, 0.21408949],
       [0.14151651, 0.13642943, 0.14539204, 0.1417418 , 0.18291881],
       [0.1415223 , 0.13840584, 0.14526601, 0.14365687, 0.21626243],
```

```
[0.14344113, 0.14974936, 0.15397773, 0.1554362 , 0.23044278],
[0.15517875, 0.16066206, 0.16590344, 0.16798558, 0.28950892],
[0.16770538, 0.16367967, 0.16327626, 0.15674719, 0.30634571],
[0.15649317, 0.1567142 , 0.1671548 , 0.16213434, 0.18150171],
[0.16186768, 0.1566163 , 0.16402882, 0.15736792, 0.19791029],
[0.1571169 , 0.15091579, 0.16329215, 0.15419284, 0.16090385],
[0.15394455, 0.15331251, 0.16512002, 0.15779391, 0.18295357],
[0.1575361 , 0.15304731, 0.16542201, 0.16019056, 0.17770213],
[0.15992727, 0.16803526, 0.17145945, 0.1745415 , 0.30662866],
[0.17424602, 0.16753728, 0.17631918, 0.16979795, 0.20416013],
[0.16951353, 0.1629931 , 0.1742356 , 0.16558657, 0.18188798],
[0.16531194, 0.1841119 , 0.17705348, 0.19108301, 0.36086261],
[0.19686444, 0.19604877, 0.20432538, 0.20383948, 0.4098845 ],
[0.20346877, 0.1960831 , 0.19949234, 0.197759  , 0.43396585],
[0.19741151, 0.21428184, 0.20601586, 0.22027468, 0.46907436],
[0.21987255, 0.21956164, 0.2314975 , 0.22280981, 0.44418734],
[0.22240188, 0.22717096, 0.23265752, 0.23359584, 0.46551633],
[0.23316278, 0.22620308, 0.22963422, 0.22315099, 0.44561746],
[0.22274248, 0.23298386, 0.23426513, 0.24100605, 0.51041855],
[0.24053728, 0.24216155, 0.25125489, 0.24147408, 0.54936384],
[0.24101741, 0.2336487 , 0.23029017, 0.22082838, 0.42776623],
[0.2204112 , 0.21390612, 0.18624361, 0.18978505, 0.63571418],
[0.18950649, 0.18805655, 0.1884316 , 0.18487576, 0.41654152],
[0.18457858, 0.19174178, 0.19625617, 0.194935  , 0.33920818],
[0.19461073, 0.20774766, 0.20582406, 0.21337347, 0.37678909],
[0.21291833, 0.22750832, 0.22187403, 0.23687794, 0.48301587],
[0.23644277, 0.23866642, 0.23465341, 0.2308497 , 0.69615511],
[0.23042809, 0.23282501, 0.24091499, 0.24049084, 0.54689075],
[0.24009782, 0.23556354, 0.24817274, 0.24117418, 0.42641642],
[0.24073741, 0.23147117, 0.22699574, 0.2155775 , 0.38598829],
[0.21515919, 0.21685419, 0.22815502, 0.22431929, 0.32029588],
[0.22389993, 0.21919086, 0.23251174, 0.22554984, 0.24492881],
[0.22510785, 0.22373527, 0.23651598, 0.22574354, 0.30573444],
[0.22530742, 0.21832152, 0.21859277, 0.20636569, 0.29995892],
[0.20596723, 0.20052877, 0.20849032, 0.2082324 , 0.30058267],
[0.20786189, 0.19911749, 0.21105496, 0.20095684, 0.2131825 ],
[0.20058912, 0.19179911, 0.20130605, 0.19322807, 0.26670161],
[0.19286482, 0.19896861, 0.20686471, 0.20624798, 0.21031673],
[0.20589627, 0.21610218, 0.21763835, 0.22116804, 0.27259983],
[0.22084494, 0.21337289, 0.22897437, 0.22218977, 0.24529074],
[0.2217867 , 0.21830629, 0.23280448, 0.21947707, 0.26426491],
[0.21910747, 0.24019788, 0.22778754, 0.25157622, 0.43622918],
[0.25112856, 0.28428539, 0.26422059, 0.27460247, 0.79164855],
[0.27415677, 0.29239915, 0.28952579, 0.29834756, 0.71587745],
[0.29771029, 0.3233483 , 0.30798577, 0.32984183, 0.69906613],
[0.32899869, 0.32060433, 0.32329189, 0.31323306, 0.61345974],
[0.31262099, 0.3079061 , 0.29568863, 0.28600533, 0.70222127],
```

```
[0.28529604, 0.29718441, 0.28747305, 0.31045514, 0.79563108],
[0.30983909, 0.30487305, 0.2796529 , 0.27701924, 0.87124618],
[0.27670427, 0.29154399, 0.28969261, 0.30449425, 0.71674438],
[0.3037451 , 0.33483643, 0.31671037, 0.34794937, 0.82806316],
[0.3475187 , 0.35288522, 0.36650153, 0.35492668, 0.89908984],
[0.3542037 , 0.36252822, 0.36037494, 0.34945264, 1.        ],
[0.34895025, 0.35678574, 0.37233213, 0.35659277, 0.72667697],
[0.35582515, 0.34178421, 0.35462733, 0.35374746, 0.64951569],
[0.35309369, 0.3467382 , 0.36141885, 0.35494607, 0.57625043],
[0.35427025, 0.34059427, 0.36019164, 0.34148593, 0.44487478],
[0.34525169, 0.36805069, 0.36216889, 0.37602456, 0.74676813],
[0.37527239, 0.43133902, 0.39841501, 0.44154094, 1.09093002],
[0.44085002, 0.45589776, 0.42213733, 0.42636861, 0.96855046],
[0.42574015, 0.42851855, 0.43349927, 0.43704326, 0.72271243],
[0.43633643, 0.44683103, 0.45772934, 0.46641737, 0.76178775],
[0.46558615, 0.47700341, 0.47985585, 0.46810569, 0.92882711],
[0.4671982 , 0.47940637, 0.44628766, 0.47843221, 1.04270883],
[0.4773553 , 0.47220606, 0.48390563, 0.49028104, 0.7896474 ],
[0.48923102, 0.49020382, 0.48279929, 0.4723302 , 0.9057733 ],
[0.4712847 , 0.45069777, 0.33587964, 0.38252765, 1.27590256],
[0.3812633 , 0.38511641, 0.37791439, 0.36388002, 0.84368684],
[0.36320818, 0.38824104, 0.37170237, 0.40555173, 0.59712817],
[0.4046011 , 0.41161571, 0.42381878, 0.41869474, 0.58646798],
[0.41790622, 0.40771584, 0.38774368, 0.39657843, 0.59676486],
[0.39562687, 0.39949368, 0.41123925, 0.396657  , 0.50934742],
[0.39572993, 0.38540654, 0.40241561, 0.39148628, 0.43241347],
[0.39069615, 0.40612453, 0.40765691, 0.41886246, 0.47080172],
[0.41814813, 0.43678833, 0.4427902 , 0.42264857, 0.66045186],
[0.42140578, 0.41046316, 0.4085032 , 0.41262623, 0.63827567],
[0.41181169, 0.39379938, 0.36981325, 0.35682715, 0.61707176],
[0.35593923, 0.36978026, 0.3620284 , 0.37694498, 0.60626336],
[0.37597055, 0.37334136, 0.39652379, 0.37829652, 0.38148631],
[0.37749181, 0.37418375, 0.39461922, 0.38879889, 0.35252318],
[0.3878945 , 0.38628366, 0.40223637, 0.37743185, 0.44985875],
[0.37663198, 0.36581356, 0.37773331, 0.36985327, 0.41812916],
[0.36909204, 0.35322821, 0.3474069 , 0.33657946, 0.44533701],
[0.33588338, 0.34481981, 0.35567789, 0.35811848, 0.47926764],
[0.37884098, 0.38038728, 0.38764869, 0.37057848, 0.6295436 ],
[0.36978883, 0.35602799, 0.37788422, 0.36603362, 0.44897463],
[0.36534265, 0.35280066, 0.36986979, 0.35524126, 0.32183505],
[0.35453545, 0.35663817, 0.37016303, 0.36195228, 0.39754939],
[0.36122293, 0.38193271, 0.38702029, 0.39170092, 0.53547076],
[0.39081448, 0.40980089, 0.41775575, 0.42863661, 0.60884415],
[0.42787415, 0.4157097 , 0.41849696, 0.39914502, 0.55386753],
[0.39833314, 0.41153066, 0.42422616, 0.42649514, 0.52180209],
[0.42561791, 0.43008085, 0.44286509, 0.42892608, 0.59018835],
[0.42791087, 0.41312682, 0.42827121, 0.41603272, 0.48815397],
```

```
[0.41508307, 0.44308768, 0.43567367, 0.4613502 , 0.62847939],
[0.4603723 , 0.48445806, 0.48551641, 0.5033319 , 0.72012662],
[0.50191328, 0.51280806, 0.50828666, 0.50350078, 0.90905968],
[0.50212498, 0.50829374, 0.52122219, 0.51004408, 0.65791431],
[0.50866088, 0.52688793, 0.52594261, 0.54593877, 0.76855854],
[0.54478542, 0.60481471, 0.57046321, 0.62766168, 1.17459175],
[0.62647441, 0.60702249, 0.62216222, 0.59441172, 0.91485314],
[0.59284205, 0.58128689, 0.5680609 , 0.57538942, 0.95598979],
[0.57396164, 0.58508829, 0.5956668 , 0.58346579, 0.89811334],
[0.58228079, 0.62792048, 0.60453387, 0.65794832, 0.97798162],
[0.65645838, 0.63121074, 0.6567382 , 0.62947098, 0.55935724],
[0.62819926, 0.63918612, 0.65482416, 0.65710979, 0.50702694],
[0.65563073, 0.65208865, 0.64184042, 0.63158574, 0.5820211 ],
[0.62994355, 0.61762928, 0.65490533, 0.63049069, 0.49168137],
[0.62911178, 0.60172198, 0.53919471, 0.57658945, 0.67521958],
[0.57540404, 0.5509722 , 0.47046245, 0.4887337 , 0.77602707],
[0.48779578, 0.49931025, 0.50086537, 0.5012576 , 0.51087567],
[0.50022044, 0.5412182 , 0.51965039, 0.49427732, 0.51957184],
[0.49387435, 0.48023822, 0.48118487, 0.47031026, 0.55842973],
[0.4694193 , 0.47446524, 0.49520797, 0.47510807, 0.35386394],
[0.47411725, 0.45487111, 0.4557643 , 0.45489069, 0.42150519],
[0.45409945, 0.46484878, 0.48334583, 0.48395076, 0.32806582],
[0.48285511, 0.48737958, 0.50510981, 0.49137602, 0.37203042],
[0.49062901, 0.51839267, 0.52366532, 0.5214014 , 0.45321002],
[0.52010845, 0.51293954, 0.52353966, 0.50041249, 0.38914233],
[0.49924906, 0.48300023, 0.49947565, 0.49953023, 0.32335294],
[0.49831092, 0.48632797, 0.51773461, 0.50577669, 0.26937768],
[0.50473424, 0.50631743, 0.53887787, 0.52769621, 0.25106607],
[0.52661217, 0.5139815 , 0.53179513, 0.53098977, 0.31203816],
[0.52989862, 0.54123646, 0.55993223, 0.56399408, 0.38344814],
[0.56280175, 0.54834318, 0.56944864, 0.55519775, 0.44640001],
[0.55382675, 0.53697853, 0.56630246, 0.557265  , 0.34195661],
[0.55614348, 0.59198415, 0.58861951, 0.61244725, 0.57287555],
[0.61123034, 0.60747665, 0.63272249, 0.62865008, 0.45224706],
[0.627092  , 0.60332347, 0.63779175, 0.596852  , 0.32171029],
[0.59537915, 0.58679602, 0.58801383, 0.55730525, 0.4386208 ],
[0.55591162, 0.54154252, 0.5682732 , 0.55667397, 0.37364394],
[0.55555439, 0.54537068, 0.57680061, 0.57084306, 0.30925683],
[0.56952648, 0.55023699, 0.59207729, 0.55525538, 0.26370229],
[0.55396585, 0.54283197, 0.58260152, 0.55673904, 0.19905984],
[0.55546789, 0.54647438, 0.59230098, 0.55601387, 0.17928647],
[0.55485052, 0.5361198 , 0.57219782, 0.54367288, 0.19557323],
[0.54252347, 0.52550087, 0.5515006 , 0.51732079, 0.2208567 ],
[0.51620219, 0.50144201, 0.53969593, 0.51821179, 0.2160422 ],
[0.51696132, 0.52129265, 0.51908168, 0.49076306, 0.28722345],
[0.48953814, 0.47230466, 0.50040551, 0.47781585, 0.24312202],
[0.47670416, 0.4865956 , 0.50975131, 0.50943154, 0.22046277],
```

```
[0.50832386, 0.492258  , 0.52893604, 0.51098668, 0.17070409],
[0.50989026, 0.49549071, 0.53661458, 0.51154485, 0.15331607],
[0.51042804, 0.52170141, 0.54204521, 0.53816856, 0.23777877],
[0.53701482, 0.52878255, 0.56514557, 0.54438974, 0.25555517],
[0.54325188, 0.52652695, 0.56014678, 0.5466924 , 0.24220151],
[0.54560097, 0.5431636 , 0.57956016, 0.56533186, 0.27449798],
[0.56396033, 0.56720419, 0.5993153 , 0.59102432, 0.29034031],
[0.58956552, 0.57752536, 0.58318052, 0.54570344, 0.28503644],
[0.54443518, 0.54518896, 0.57832966, 0.56513567, 0.20323623],
[0.5639439 , 0.59678839, 0.59456091, 0.61312265, 0.33958067],
[0.61333954, 0.64821472, 0.64482462, 0.65989692, 0.65709897],
[0.65818014, 0.64453947, 0.62855609, 0.60848529, 0.61765059],
[0.60836811, 0.60473188, 0.64618146, 0.62984715, 0.32133153],
[0.62854478, 0.60638222, 0.65227452, 0.61535857, 0.22241665],
[0.6136246 , 0.67823264, 0.6525493 , 0.71050265, 0.47409324],
[0.71261715, 0.70200888, 0.73141893, 0.704906  , 0.4811759 ],
[0.70208583, 0.68608956, 0.71544635, 0.6791453 , 0.37012239],
[0.67828671, 0.72551647, 0.72617502, 0.74252364, 0.48982369],
[0.74161072, 0.75087262, 0.76256057, 0.77850636, 0.58247674],
[0.77565866, 0.77269558, 0.79389849, 0.79779061, 0.45878275],
[0.79517164, 0.84311817, 0.79478717, 0.8654953 , 0.84028001],
[0.86062051, 0.89169112, 0.89450231, 0.84899955, 0.93978509],
[0.84681536, 0.81711978, 0.70712082, 0.75976494, 0.93097046],
[0.75964225, 0.7473061 , 0.7467693 , 0.72825602, 0.66541815],
[0.724874  , 0.72374023, 0.69910505, 0.72549995, 0.6697706 ],
[0.72427676, 0.73374163, 0.74728008, 0.71948051, 0.5458261 ],
[0.71581731, 0.76876439, 0.71603926, 0.70333866, 0.69932115],
[0.70188559, 0.68086954, 0.62795887, 0.67030786, 0.65605098],
[0.66913579, 0.63998377, 0.65580788, 0.6237389 , 0.32386208],
[0.62263895, 0.62316578, 0.62873717, 0.62034858, 0.26255724],
[0.62001068, 0.65747208, 0.65769349, 0.68735133, 0.34992839],
[0.68634576, 0.69690256, 0.7241451 , 0.72186682, 0.32652841],
[0.72068615, 0.70683287, 0.7377223 , 0.71863372, 0.30029214],
[0.71727267, 0.69671654, 0.74008522, 0.71040422, 0.2521001 ],
[0.7084816 , 0.72549578, 0.75090154, 0.75432048, 0.276417  ],
[0.75260564, 0.73887758, 0.79225466, 0.77029096, 0.25606146],
[0.76896436, 0.73689774, 0.78835607, 0.74850687, 0.22108189],
[0.74660408, 0.79558867, 0.79856602, 0.82065767, 0.3667286 ],
[0.81905446, 0.86980453, 0.81883396, 0.85315555, 0.81832817],
[0.84830667, 0.95753388, 0.8970251 , 0.99352065, 0.93478044],
[0.99653743, 0.96559419, 0.97632831, 0.96148079, 0.67498237]])
```

[26]:
```
X_test = []
y_test = []
for i in range (60, inputs.shape[0]):
    X_test.append(inputs[i-60:i])
    y_test.append(inputs[i, 0])
```

[27]:
```
X_test, y_test = np.array(X_test), np.array(y_test)
X_test.shape, y_test.shape
```

[27]:
```
((125, 60, 5), (125,))
```

[28]:
```
y_pred = regressor.predict(X_test)
y_pred, y_test
```

[28]:
```
(array([[0.25960687],
        [0.25841773],
        [0.26391304],
        [0.27821484],
        [0.29683056],
        [0.31620568],
        [0.33584392],
        [0.35537764],
        [0.37175015],
        [0.384363  ],
        [0.39348674],
        [0.3954489 ],
        [0.38958508],
        [0.37829712],
        [0.36403194],
        [0.3487959 ],
        [0.33404443],
        [0.32170013],
        [0.3149298 ],
        [0.31488097],
        [0.31956702],
        [0.32570446],
        [0.32975867],
        [0.33057946],
        [0.32920912],
        [0.32647035],
        [0.3228251 ],
        [0.31880963],
        [0.31632724],
        [0.31484786],
        [0.3129187 ],
        [0.3104181 ],
        [0.3089343 ],
        [0.31081933],
        [0.316398  ],
        [0.32447952],
        [0.33436936],
        [0.34368506],
        [0.35249925],
```

```
[0.36278874],
[0.3773867 ],
[0.3932778 ],
[0.40919703],
[0.4301499 ],
[0.45358604],
[0.47464517],
[0.48968926],
[0.49999833],
[0.50446653],
[0.5050118 ],
[0.505032  ],
[0.5060692 ],
[0.5089161 ],
[0.51151174],
[0.5097476 ],
[0.5034816 ],
[0.49334618],
[0.47986758],
[0.46449578],
[0.44871902],
[0.434816  ],
[0.42585513],
[0.4228222 ],
[0.42416444],
[0.42775604],
[0.43243513],
[0.43812048],
[0.44600868],
[0.45689708],
[0.468436  ],
[0.482392  ],
[0.4979027 ],
[0.51143765],
[0.52086425],
[0.5237389 ],
[0.52032983],
[0.51243967],
[0.50175714],
[0.49036902],
[0.47992697],
[0.47131813],
[0.46448272],
[0.45969763],
[0.45556182],
[0.45151365],
[0.44750535],
```

```
       [0.4437994 ],
       [0.4420432 ],
       [0.443643  ],
       [0.4484115 ],
       [0.4561778 ],
       [0.46683708],
       [0.47865433],
       [0.48813483],
       [0.49648345],
       [0.5099887 ],
       [0.5278459 ],
       [0.5421676 ],
       [0.5486271 ],
       [0.5521928 ],
       [0.5579968 ],
       [0.56564206],
       [0.5767001 ],
       [0.5932346 ],
       [0.6131123 ],
       [0.6405065 ],
       [0.675848  ],
       [0.704404  ],
       [0.7141733 ],
       [0.70825845],
       [0.6927945 ],
       [0.67436236],
       [0.6553622 ],
       [0.6354195 ],
       [0.6148242 ],
       [0.59598017],
       [0.58219653],
       [0.57537717],
       [0.57518816],
       [0.58167124],
       [0.5941396 ],
       [0.60957247],
       [0.6285907 ],
       [0.6580169 ],
       [0.70218045]], dtype=float32),
array([0.34525169, 0.37527239, 0.44085002, 0.42574015, 0.43633643,
       0.46558615, 0.4671982 , 0.4773553 , 0.48923102, 0.4712847 ,
       0.3812633 , 0.36320818, 0.4046011 , 0.41790622, 0.39562687,
       0.39572993, 0.39069615, 0.41814813, 0.42140578, 0.41181169,
       0.35593923, 0.37597055, 0.37749181, 0.3878945 , 0.37663198,
       0.36909204, 0.33588338, 0.37884098, 0.36978883, 0.36534265,
       0.35453545, 0.36122293, 0.39081448, 0.42787415, 0.39833314,
       0.42561791, 0.42791087, 0.41508307, 0.4603723 , 0.50191328,
```

```
         0.50212498, 0.50866088, 0.54478542, 0.62647441, 0.59284205,
         0.57396164, 0.58228079, 0.65645838, 0.62819926, 0.65563073,
         0.62994355, 0.62911178, 0.57540404, 0.48779578, 0.50022044,
         0.49387435, 0.4694193 , 0.47411725, 0.45409945, 0.48285511,
         0.49062901, 0.52010845, 0.49924906, 0.49831092, 0.50473424,
         0.52661217, 0.52989862, 0.56280175, 0.55382675, 0.55614348,
         0.61123034, 0.627092  , 0.59537915, 0.55591162, 0.55555439,
         0.56952648, 0.55396585, 0.55546789, 0.55485052, 0.54252347,
         0.51620219, 0.51696132, 0.48953814, 0.47670416, 0.50832386,
         0.50989026, 0.51042804, 0.53701482, 0.54325188, 0.54560097,
         0.56396033, 0.58956552, 0.54443518, 0.5639439 , 0.61333954,
         0.65818014, 0.60836811, 0.62854478, 0.6136246 , 0.71261715,
         0.70208583, 0.67828671, 0.74161072, 0.77565866, 0.79517164,
         0.86062051, 0.84681536, 0.75964225, 0.724874  , 0.72427676,
         0.71581731, 0.70188559, 0.66913579, 0.62263895, 0.62001068,
         0.68634576, 0.72068615, 0.71727267, 0.7084816 , 0.75260564,
         0.76896436, 0.74660408, 0.81905446, 0.84830667, 0.99653743]))
```

[29]:
```
scaler.scale_
```

[29]:
```
array([2.81186421e-03, 2.69013692e-03, 3.00890173e-03, 2.81843036e-03,
       7.09086803e-11])
```

[30]:
```
scale = 1/2.81186421e-03
scale
```

[30]:
```
355.6359501442639
```

[31]:
```
y_test = y_test*scale
y_pred = y_pred*scale
```

[32]:
```
y_pred
```

[32]:
```
array([[ 92.32554 ],
       [ 91.90263 ],
       [ 93.856964],
       [ 98.9432  ],
       [105.56362 ],
       [112.45411 ],
       [119.43817 ],
       [126.38507 ],
       [132.20772 ],
       [136.6933  ],
       [139.93803 ],
       [140.63585 ],
       [138.55046 ],
       [134.53606 ],
```

```
[129.46284 ],
[124.04436 ],
[118.79821 ],
[114.408134],
[112.00037 ],
[111.982994],
[113.64952 ],
[115.832214],
[117.27404 ],
[117.56594 ],
[117.0786  ],
[116.10459 ],
[114.80821 ],
[113.380165],
[112.497345],
[111.97122 ],
[111.28514 ],
[110.395836],
[109.86815 ],
[110.53853 ],
[112.52251 ],
[115.39658 ],
[118.913765],
[122.22677 ],
[125.361404],
[129.02072 ],
[134.21228 ],
[139.86372 ],
[145.52518 ],
[152.97678 ],
[161.31151 ],
[168.80089 ],
[174.15111 ],
[177.81738 ],
[179.40643 ],
[179.60036 ],
[179.60754 ],
[179.9764  ],
[180.98886 ],
[181.91197 ],
[181.28458 ],
[179.05617 ],
[175.45164 ],
[170.65816 ],
[165.1914  ],
[159.58061 ],
[154.6362  ],
```

```
[151.4494  ],
[150.37077 ],
[150.84813 ],
[152.12543 ],
[153.78947 ],
[155.8114  ],
[158.61673 ],
[162.48903 ],
[166.59268 ],
[171.55594 ],
[177.0721  ],
[181.88562 ],
[185.23805 ],
[186.26039 ],
[185.048   ],
[182.24197 ],
[178.44289 ],
[174.39285 ],
[170.67929 ],
[167.61768 ],
[165.18675 ],
[163.485   ],
[162.01416 ],
[160.5745  ],
[159.149   ],
[157.83102 ],
[157.20645 ],
[157.7754  ],
[159.47125 ],
[162.23323 ],
[166.02405 ],
[170.22668 ],
[173.5983  ],
[176.56737 ],
[181.37033 ],
[187.721   ],
[192.8143  ],
[195.11151 ],
[196.37962 ],
[198.44373 ],
[201.16266 ],
[205.09529 ],
[210.97556 ],
[218.04477 ],
[227.78714 ],
[240.35585 ],
[250.51138 ],
```

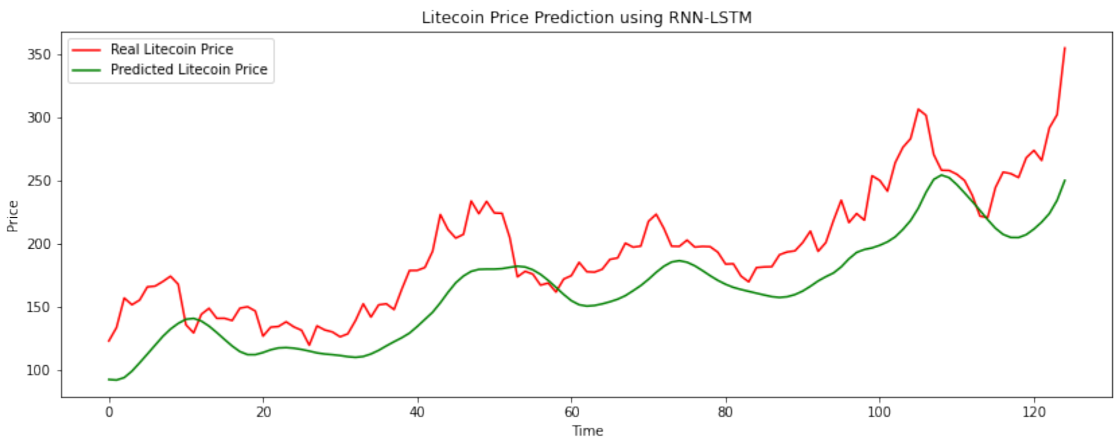
```
          [253.98572 ],
          [251.88217 ],
          [246.38263 ],
          [239.8275  ],
          [233.07036 ],
          [225.97801 ],
          [218.65358 ],
          [211.95198 ],
          [207.05002 ],
          [204.6248  ],
          [204.55759 ],
          [206.8632  ],
          [211.2974  ],
          [216.78589 ],
          [223.54945 ],
          [234.01448 ],
          [249.72061 ]], dtype=float32)
```

[33]:
```
y_test
```

[33]:
```
array([122.78391405, 133.46035205, 156.78211406, 151.40850206,
       155.17692006, 165.57917207, 166.15247507, 169.76470407,
       173.98813807, 167.60578307, 135.59093706, 129.16988605,
       143.89069506, 148.62247606, 140.69913706, 140.73578806,
       138.94559706, 148.70850606, 149.86704406, 146.45504206,
       126.58478705, 133.70864305, 134.24965805, 137.94922806,
       133.94387205, 131.26240005, 119.45220405, 134.72947105,
       131.51020205, 129.92898005, 126.08555005, 128.46386105,
       138.98768006, 152.16742906, 141.66158506, 151.36503006,
       152.18049006, 147.61846306, 163.72493907, 178.49840707,
       178.57369407, 180.89809607, 193.74527908, 222.79682309,
       210.83594709, 204.12139308, 207.07998108, 233.46019909,
       223.41024209, 233.16585709, 224.03057309, 223.73476609,
       204.63436308, 173.47771607, 177.89637207, 175.63947507,
       166.94237707, 168.61313807, 161.49408907, 171.72063607,
       174.48531507, 184.96926308, 177.55091307, 177.21727907,
       179.50164207, 187.28222108, 188.45099808, 200.15253608,
       196.96070308, 197.78461608, 217.37548209, 223.01645809,
       211.73823009, 197.70215808, 197.57511308, 202.54409208,
       197.01017208, 197.54435108, 197.32479308, 192.94085108,
       183.58005707, 183.85003107, 174.09736107, 169.53313607,
       180.77823807, 181.33530607, 181.52656007, 190.98177508,
       193.19990008, 194.03531808, 200.56456908, 209.67069409,
       193.62072208, 200.55872508, 218.12558909, 234.07251909,
       216.35756909, 223.53312109, 218.22696909, 253.4322761 ,
       249.6869601 , 241.2231391 , 263.74343311, 275.85210611,
       282.79162011, 306.06759112, 301.15798412, 270.15609211,
```

```
257.7912541 , 257.5788521 , 254.5703681 , 249.6157471 ,
237.9687441 , 221.43279409, 220.49808709, 244.0892281 ,
256.3019051 , 255.0879461 , 251.9615271 , 267.65362011,
273.47136911, 265.51925111, 291.28521112, 301.68834912,
354.40453514])
```

[35]:
```
plt.figure(figsize=(14,5))
plt.plot(y_test, color = 'red', label = 'Real Litecoin Price')
plt.plot(y_pred, color = 'green', label = 'Predicted Litecoin Price')
plt.title('Litecoin Price Prediction using RNN-LSTM')
plt.xlabel('Time')
plt.ylabel('Price')
plt.legend()
plt.show()
```



1.5 RNN model

[38]:
```
regressor_2 = Sequential()
regressor_2.add(SimpleRNN(units = 50, activation = 'relu', return_sequences =␣
↪True, input_shape = (X_train.shape[1], 5)))
regressor_2.add(Dropout(0.2))
```

[39]:
```
regressor_2.add(SimpleRNN(units = 60, activation = 'relu', return_sequences =␣
↪True))
regressor_2.add(Dropout(0.3))

regressor_2.add(SimpleRNN(units = 80, activation = 'relu', return_sequences =␣
↪True))
regressor_2.add(Dropout(0.4))

regressor_2.add(SimpleRNN(units = 120, activation = 'relu'))
```

```
regressor_2.add(Dropout(0.5))

regressor_2.add(Dense(units =1))
```

[40]:
```
regressor_2.summary()
```

```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
simple_rnn_4 (SimpleRNN)     (None, 60, 50)            2800
_________________________________________________________________
dropout_7 (Dropout)          (None, 60, 50)            0
_________________________________________________________________
simple_rnn_5 (SimpleRNN)     (None, 60, 60)            6660
_________________________________________________________________
dropout_8 (Dropout)          (None, 60, 60)            0
_________________________________________________________________
simple_rnn_6 (SimpleRNN)     (None, 60, 80)            11280
_________________________________________________________________
dropout_9 (Dropout)          (None, 60, 80)            0
_________________________________________________________________
simple_rnn_7 (SimpleRNN)     (None, 120)               24120
_________________________________________________________________
dropout_10 (Dropout)         (None, 120)               0
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 121
=================================================================
Total params: 44,981
Trainable params: 44,981
Non-trainable params: 0
_________________________________________________________________
```

[41]:
```
regressor_2.compile(optimizer = optimizer, loss = 'mean_squared_error')
```

[42]:
```
regressor_2.fit(X_train, y_train, epochs = 20, batch_size =32)
```

```
Epoch 1/20
52/52 [==============================] - 13s 205ms/step - loss: 0.0906
Epoch 2/20
52/52 [==============================] - 11s 208ms/step - loss: 0.0059
Epoch 3/20
52/52 [==============================] - 11s 204ms/step - loss: 0.0039
Epoch 4/20
52/52 [==============================] - 11s 203ms/step - loss: 0.0043
Epoch 5/20
52/52 [==============================] - 11s 206ms/step - loss: 0.0057
Epoch 6/20
```

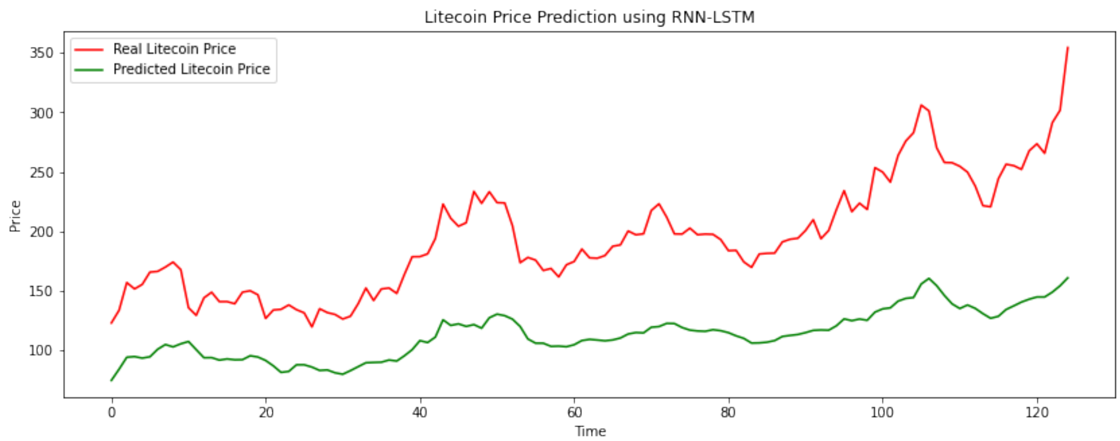
```
52/52 [==============================] - 11s 203ms/step - loss: 0.0033
Epoch 7/20
52/52 [==============================] - 10s 201ms/step - loss: 0.0039
Epoch 8/20
52/52 [==============================] - 10s 199ms/step - loss: 0.0039
Epoch 9/20
52/52 [==============================] - 11s 203ms/step - loss: 0.0031
Epoch 10/20
52/52 [==============================] - 11s 203ms/step - loss: 0.0029
Epoch 11/20
52/52 [==============================] - 11s 202ms/step - loss: 0.0027
Epoch 12/20
52/52 [==============================] - 10s 199ms/step - loss: 0.0027
Epoch 13/20
52/52 [==============================] - 10s 199ms/step - loss: 0.0018
Epoch 14/20
52/52 [==============================] - 10s 200ms/step - loss: 0.0024
Epoch 15/20
52/52 [==============================] - 10s 200ms/step - loss: 0.0016
Epoch 16/20
52/52 [==============================] - 10s 199ms/step - loss: 0.0016
Epoch 17/20
52/52 [==============================] - 11s 203ms/step - loss: 0.0019
Epoch 18/20
52/52 [==============================] - 10s 200ms/step - loss: 0.0022
Epoch 19/20
52/52 [==============================] - 10s 194ms/step - loss: 0.0024
Epoch 20/20
52/52 [==============================] - 10s 197ms/step - loss: 0.0020
```

[42]:
```
<tensorflow.python.keras.callbacks.History at 0x7f88044874d0>
```

[43]:
```python
y_pred = regressor_2.predict(X_test)
```

[45]:
```python
y_pred = y_pred*scale
```

[46]:
```python
plt.figure(figsize=(14,5))
plt.plot(y_test, color = 'red', label = 'Real Litecoin Price')
plt.plot(y_pred, color = 'green', label = 'Predicted Litecoin Price')
plt.title('Litecoin Price Prediction using RNN-LSTM')
plt.xlabel('Time')
plt.ylabel('Price')
plt.legend()
plt.show()
```



1.6 Conclusion

In this report, two models were created for forecasting the litecoin cryptocurrency price; a Recurent neural network model and a Long Short-Term Memory model. For comparison reasons, the algorithms used the same hyperparameters and both were trained on 20 epochs.

1.7 Results

By comparing the model summary for Simple RNN with the model summary for LSTM, it is evident that there are more trainable parameters for the LSTM (approximately 1:4 ratio for the LSTM), which explains why it took a longer time to train this model.

Overall the plots show that the LSTM model performs a lot better, compared to the RNN, as it clearly captures the upward trend of the price.

1.7.1 Next Steps

To improve the quality of forecasts over many time steps, we'd need to use more data and more sophisticated LSTM model structures. We could try training with more data and running more training epochs. An additional recommendation would be to change the hyperparameters to achieve even smaller losses. However, the loss of this simple LSTM model was only 0.019.