If you do not have these libraries, you can install them via pip.

```
pip install yfinance
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple
Collecting yfinance
  Downloading yfinance-0.1.74-py2.py3-none-any.whl (27 kB)
Requirement already satisfied: multitasking>=0.0.7 in /usr/local/lib/python3.7/dist-packages (fro
Requirement already satisfied: pandas>=0.24.0 in /usr/local/lib/python3.7/dist-packages (from yfi
Requirement already satisfied: numpy>=1.15 in /usr/local/lib/python3.7/dist-packages (from yfinan
Collecting lxml>=4.5.1
  Downloading lxml-4.9.1-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.manylinux_2_24_x86_
     |████████████████████████████████| 6.4 MB 7.5 MB/s
Collecting requests>=2.26
  Downloading requests-2.28.1-py3-none-any.whl (62 kB)
     |████████████████████████████████| 62 kB 1.5 MB/s
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (from panda
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (from python-da
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.7/dist-packages (from reque
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: charset-normalizer<3,>=2 in /usr/local/lib/python3.7/dist-packages
Installing collected packages: requests, lxml, yfinance
  Attempting uninstall: requests
    Found existing installation: requests 2.23.0
    Uninstalling requests-2.23.0:
      Successfully uninstalled requests-2.23.0
  Attempting uninstall: lxml
    Found existing installation: lxml 4.2.6
    Uninstalling lxml-4.2.6:
      Successfully uninstalled lxml-4.2.6
ERROR: pip's dependency resolver does not currently take into account all the packages that are i
google-colab 1.0.0 requires requests~=2.23.0, but you have requests 2.28.1 which is incompatible.
datascience 0.10.6 requires folium==0.2.1, but you have folium 0.8.3 which is incompatible.
Successfully installed lxml-4.9.1 requests-2.28.1 yfinance-0.1.74
```

We need to load the following libraries:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px



from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score,mean_squared_log_error,mean_absolute_error


from keras.models import Sequential
from keras.layers import Dropout,Dense,BatchNormalization ,LSTM ,GRU
```

```
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.utils import plot_model

import yfinance as yf
import datetime as time
```

get data as yahoo finance

```
btc_data=yf.download("BTC-USD",start='2014-09-17',stop=time.date.today())
```

```
[*********************100%***********************]  1 of 1 completed
```

```
min_date = str(btc_data.index.min())
max_date = str(btc_data.index.max())
print("We have collected Bitcoin stock price data from %s to %s (today)" %( min_date , max_date))
```

```
We have collected Bitcoin stock price data from 2014-09-17 00:00:00 to 2022-07-16 00:00:00 (today
```

see the data shape

```
print("our data have %d rows and %d columns(Features)"%(btc_data.shape[0],btc_data.shape[1]))
```

```
our data have 2859 rows and 6 columns(Features)
```

```
btc_data.head()
```

	Open	High	Low	Close	Adj Close	Volume
Date						
2014-09-17	465.864014	468.174011	452.421997	457.334015	457.334015	21056800
2014-09-18	456.859985	456.859985	413.104004	424.440002	424.440002	34483200
2014-09-19	424.102997	427.834991	384.532013	394.795990	394.795990	37919700
2014-09-20	394.673004	423.295990	389.882996	408.903992	408.903992	36863600
2014-09-21	408.084991	412.425995	393.181000	398.821014	398.821014	26580100

```
btc_data.describe()
```

	Open	High	Low	Close	Adj Close	Volume
count	2859.000000	2859.000000	2859.000000	2859.000000	2859.000000	2.859000e+03
mean	12458.528546	12782.213041	12094.551533	12463.873693	12463.873693	1.547102e+10
std	16566.050623	16993.655647	16069.524839	16561.479049	16561.479049	1.990822e+10

```
btc_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 2853 entries, 2014-09-17 to 2022-07-09
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Open       2853 non-null   float64
 1   High       2853 non-null   float64
 2   Low        2853 non-null   float64
 3   Close      2853 non-null   float64
 4   Adj Close  2853 non-null   float64
 5   Volume     2853 non-null   int64
dtypes: float64(5), int64(1)
memory usage: 156.0 KB
```

```
print(btc_data.isna().sum())
```

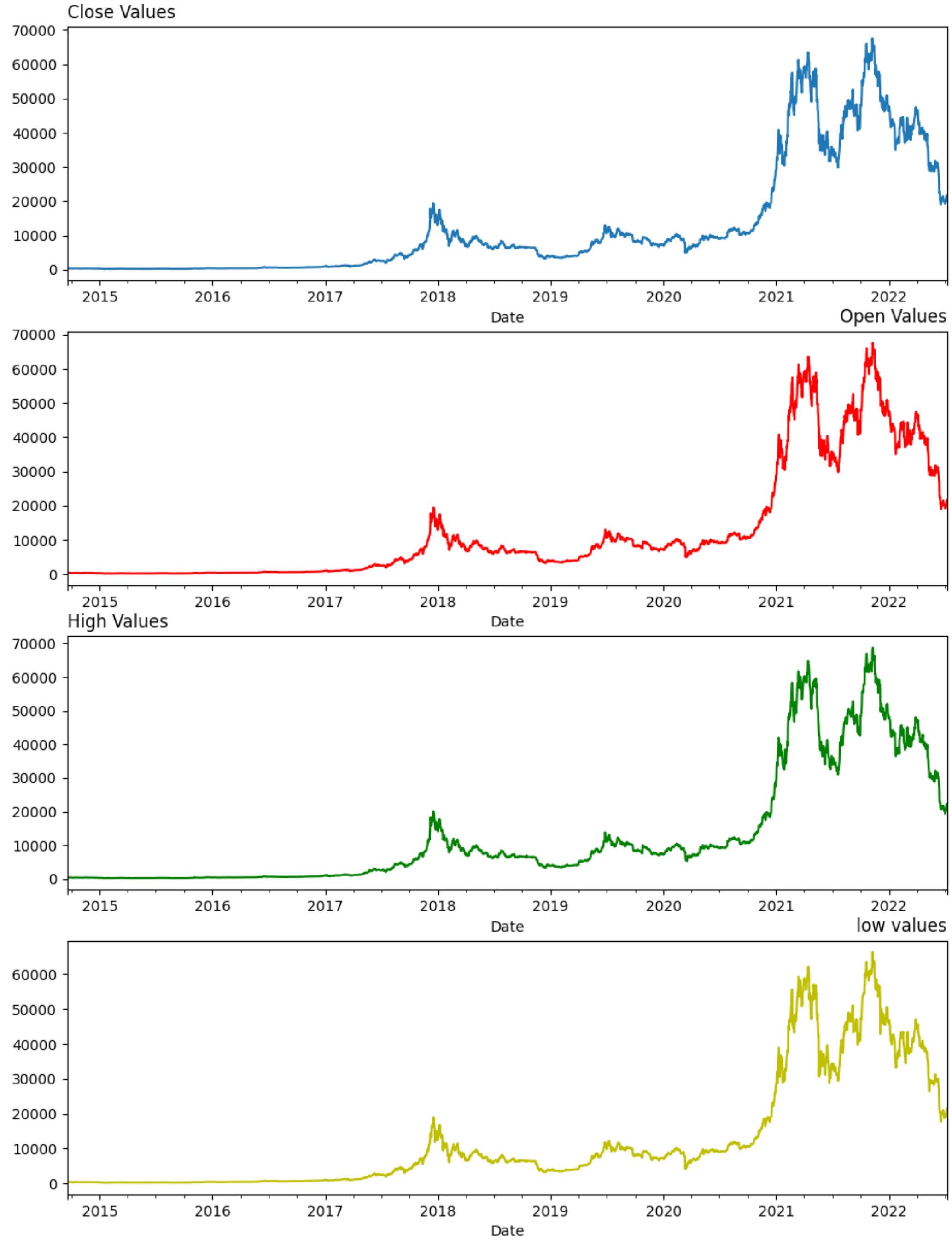
```
Open         0
High         0
Low          0
Close        0
Adj Close    0
Volume       0
dtype: int64
```

```
print("The dataset contains %d duplicate data"%(btc_data.duplicated().sum()))
```

```
The dataset contains 0 duplicate data
```

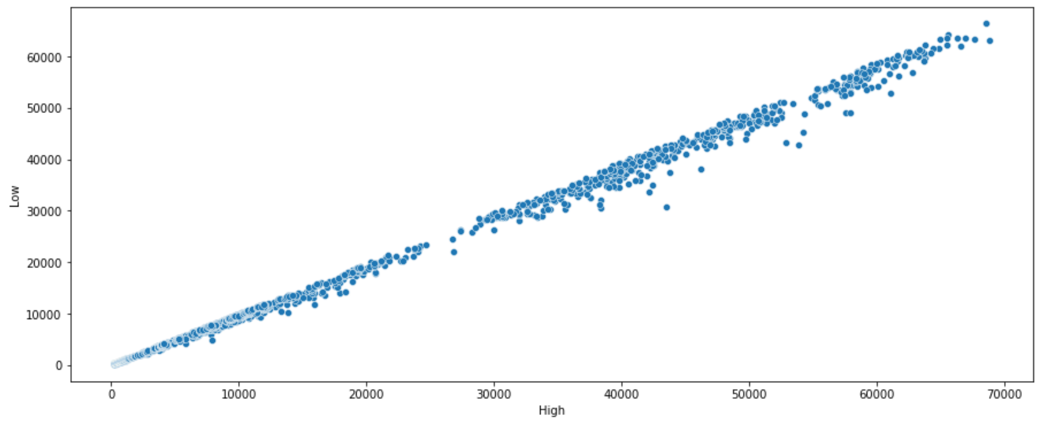
```
plt.figure(figsize=(11,15),dpi=100)
plt.subplot(411)
btc_data.Close.plot()
plt.title("Close Values",loc="left")
plt.subplot(412)
btc_data.Open.plot(c="r")
plt.title("Open Values",loc="right")
plt.subplot(413)
btc_data.High.plot(c="g")
plt.title("High Values",loc="left")
plt.subplot(414)
btc_data.Low.plot(c="y")
plt.title("low values",loc="right")
```

```
Text(1.0, 1.0, 'low values')
```

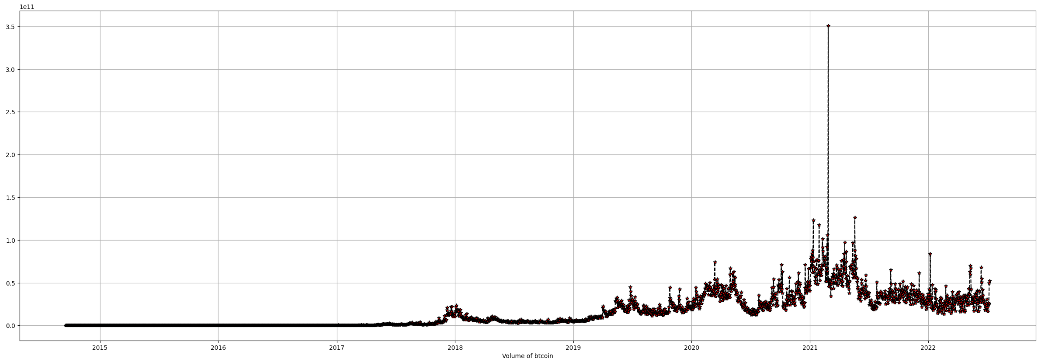


```
plt.figure(figsize=(15,6))
sns.scatterplot(x= btc_data.High, y= btc_data.Low)
```

<matplotlib.axes._subplots.AxesSubplot at 0x7fe509da3e50>

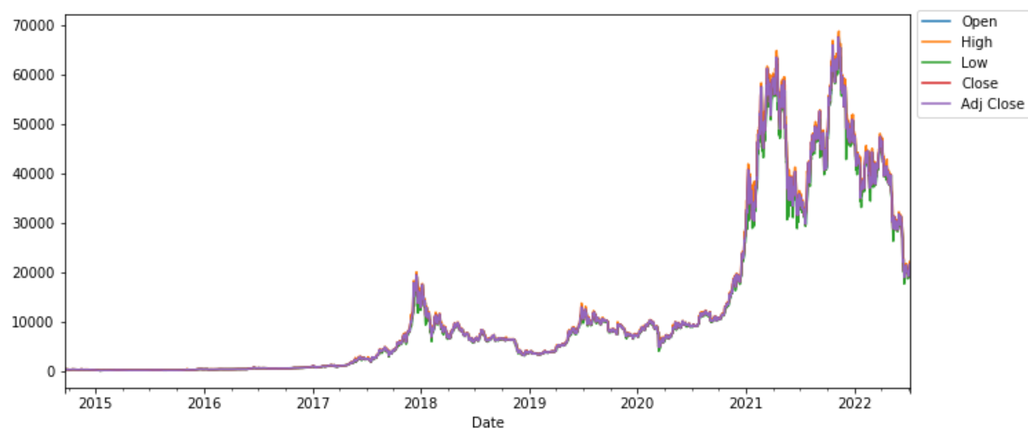



```
plt.figure(figsize = (30,10),dpi = 100)
plt.plot(btc_data["Volume"],linestyle = "--" , color = "k",marker = "*",markerfacecolor = "red")
plt.xlabel("Volume of btcoin")
plt.grid()
```



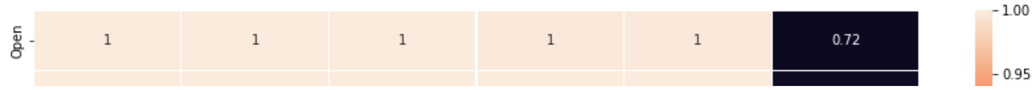

```
btc_data.drop("Volume" , axis = 1).plot(figsize = (11,5))
plt.legend(bbox_to_anchor=(1,1.03))
```

```
<matplotlib.legend.Legend at 0x7fe509ca4fd0>
```



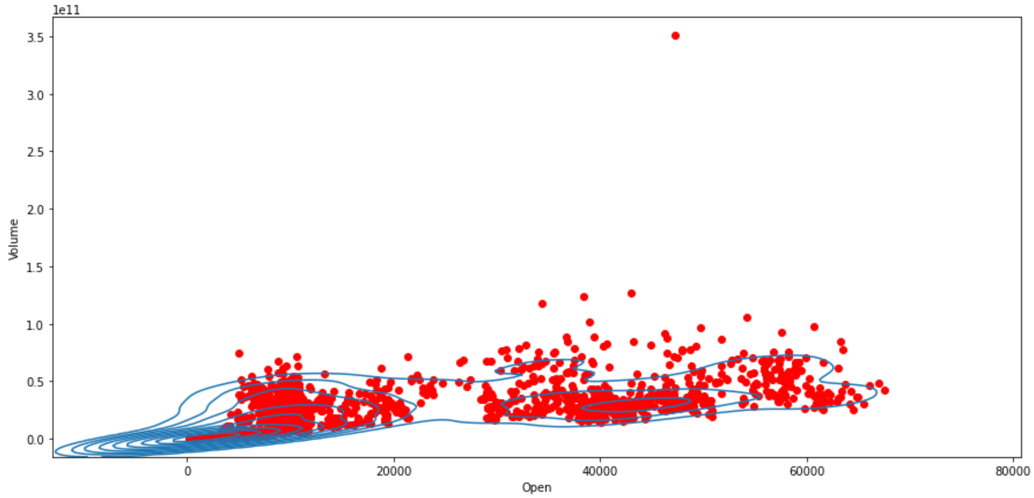

```
plt.figure(figsize = (15,5))
c_df = btc_data.corr()
sns.heatmap(c_df ,annot =True , linewidths =0.1 )
```

<matplotlib.axes._subplots.AxesSubplot at 0x7fe509a45fd0>



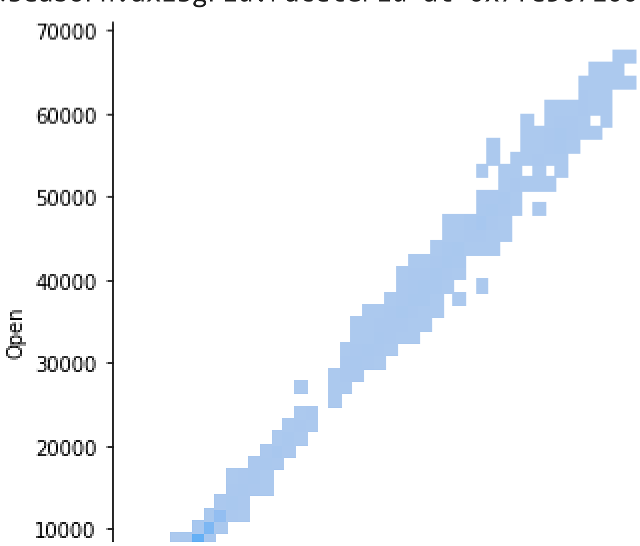

```
plt.figure(figsize = (15,7))
sns.kdeplot(x ="Open" ,y ="Volume" ,data =btc_data)
plt.scatter(btc_data["Open"], btc_data["Volume"],color = "red")
```

<matplotlib.collections.PathCollection at 0x7fe507056250>

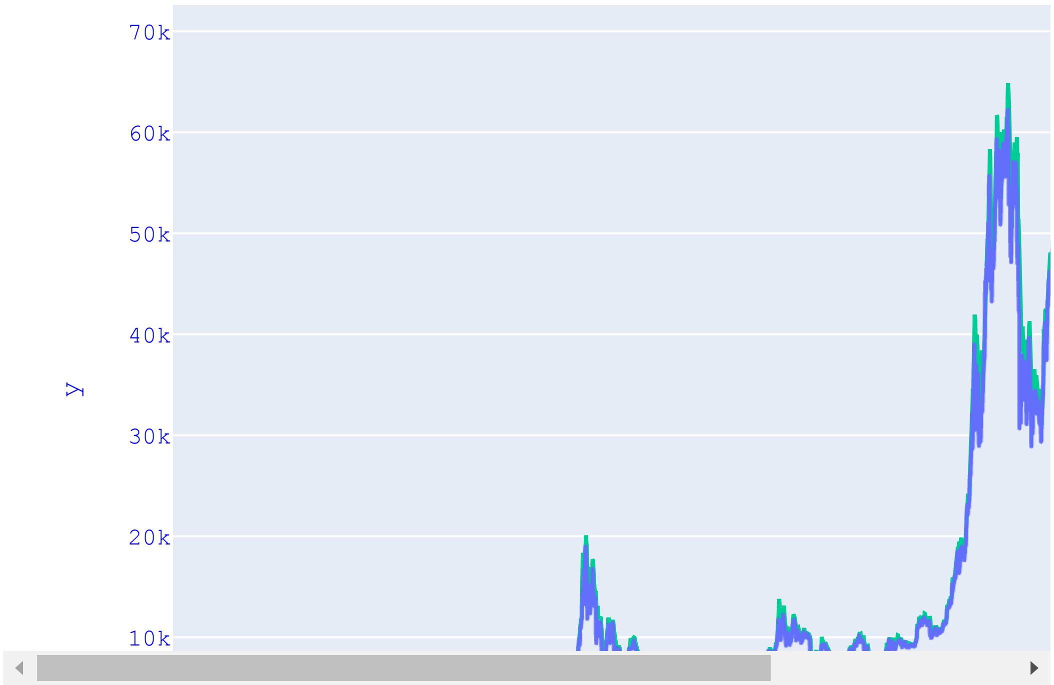



```
sns.displot(data = btc_data ,x =btc_data["Close"], y =btc_data["Open"])
```

<seaborn.axisgrid.FacetGrid at 0x7fe507100a90>



```
fig = px.line(x= btc_data.index.values , y= btc_data["Low"])
fig.add_bar(x= btc_data.index.values , y= btc_data["High"])
fig.add_scatter(x= btc_data.index.values , y= btc_data["High"])
fig.add_bar(x= btc_data.index.values , y= btc_data["Low"])
fig.update_layout(font_family="Courier New",
    font_color="blue",
    title_font_family="Times New Roman",
    title_font_color="red",
    legend_title_font_color="green")
fig.update_xaxes(title_font_family="Arial")
```



train test split

```
x = btc_data.drop("Close",axis=1)
y = btc_data.Close
x_full_train,x_test,y_full_train,y_test = train_test_split(x,y,test_size=0.2)
```

validation data

```
x_train,x_val,y_train,y_val=train_test_split(x_full_train,y_full_train,test_size=0.2)
```

```
print("shape of x train  is :",x_train.shape)
print("shape of y train is  :", y_train.shape)

print("shape of x test is  :", x_test.shape)
print("shape of y test is  :", y_test.shape)


print("shape of x val is  :", x_val.shape)
print("shape of y val is  :", y_val.shape)
```

```
shape of x train  is : (1825, 5)
shape of y train is  : (1825,)
shape of x test is  : (571, 5)
shape of y test is  : (571,)
shape of x val is  : (457, 5)
shape of y val is  : (457,)
```

Construction of neural network(ann)

```
ann_model = Sequential()
ann_model.add(Dense(45,activation="relu"))
ann_model.add(Dropout(0.3))
ann_model.add(Dense(60,activation="linear"))
ann_model.add(BatchNormalization())
ann_model.add(Dense(40,activation="relu"))
ann_model.add(Dense(1,activation="linear"))
```

compile NN

```
ann_model.compile(optimizer="adam",loss="mean_squared_error")
```

Training neural network with data

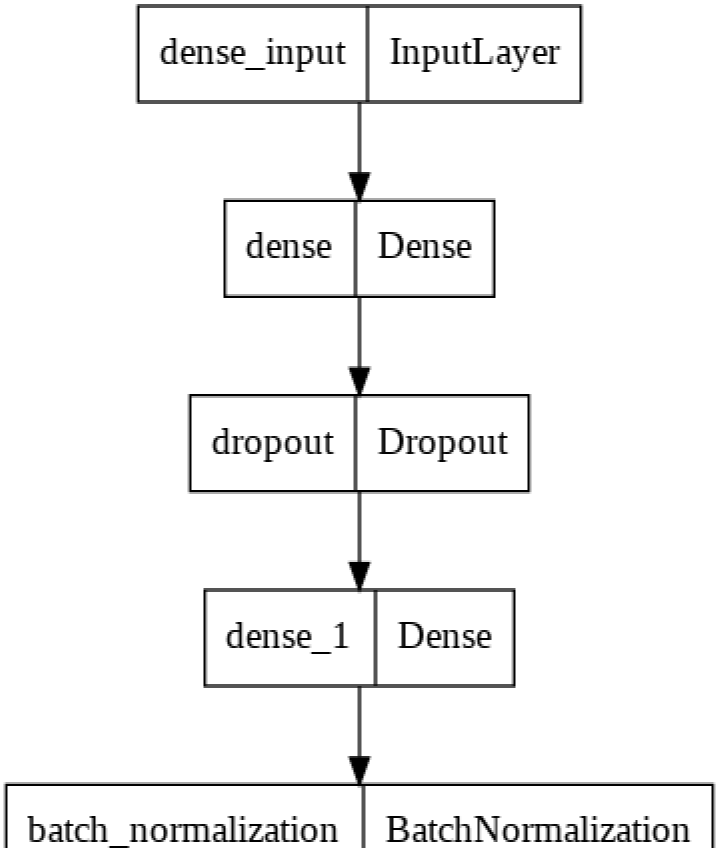
```
history = ann_model.fit(x_train,y_train,epochs=100,batch_size=32,validation_data=(x_val,y_val))
```

```
Epoch 1/100
58/58 [==============================] - 1s 6ms/step - loss: 442671040.0000 - val_loss: 4276334
Epoch 2/100
58/58 [==============================] - 0s 3ms/step - loss: 441755296.0000 - val_loss: 4249021
Epoch 3/100
58/58 [==============================] - 0s 4ms/step - loss: 439496416.0000 - val_loss: 4212468
Epoch 4/100
58/58 [==============================] - 0s 3ms/step - loss: 435445536.0000 - val_loss: 4157959
Epoch 5/100
58/58 [==============================] - 0s 3ms/step - loss: 428975136.0000 - val_loss: 4083477
Epoch 6/100
58/58 [==============================] - 0s 3ms/step - loss: 420185568.0000 - val_loss: 3979600
Epoch 7/100
58/58 [==============================] - 0s 3ms/step - loss: 408053184.0000 - val_loss: 3835580
Epoch 8/100
58/58 [==============================] - 0s 4ms/step - loss: 392740480.0000 - val_loss: 3676774
Epoch 9/100
```

```
58/58 [==============================] - 0s 4ms/step - loss: 374175296.0000 - val_loss: 3507485
Epoch 10/100
58/58 [==============================] - 0s 3ms/step - loss: 352534944.0000 - val_loss: 3301571
Epoch 11/100
58/58 [==============================] - 0s 3ms/step - loss: 329665088.0000 - val_loss: 3063310
Epoch 12/100
58/58 [==============================] - 0s 3ms/step - loss: 306539168.0000 - val_loss: 2828935
Epoch 13/100
58/58 [==============================] - 0s 3ms/step - loss: 281743616.0000 - val_loss: 2575172
Epoch 14/100
58/58 [==============================] - 0s 4ms/step - loss: 255142672.0000 - val_loss: 2345363
Epoch 15/100
58/58 [==============================] - 0s 4ms/step - loss: 232776608.0000 - val_loss: 2134723
Epoch 16/100
58/58 [==============================] - 0s 3ms/step - loss: 208674768.0000 - val_loss: 1936673
Epoch 17/100
58/58 [==============================] - 0s 3ms/step - loss: 193430976.0000 - val_loss: 1755641
Epoch 18/100
58/58 [==============================] - 0s 3ms/step - loss: 172498960.0000 - val_loss: 1617776
Epoch 19/100
58/58 [==============================] - 0s 3ms/step - loss: 170636352.0000 - val_loss: 1484571
Epoch 20/100
58/58 [==============================] - 0s 3ms/step - loss: 158757120.0000 - val_loss: 1410756
Epoch 21/100
58/58 [==============================] - 0s 3ms/step - loss: 154279616.0000 - val_loss: 1324417
Epoch 22/100
58/58 [==============================] - 0s 3ms/step - loss: 148687840.0000 - val_loss: 1244343
Epoch 23/100
58/58 [==============================] - 0s 3ms/step - loss: 141801488.0000 - val_loss: 1201362
Epoch 24/100
58/58 [==============================] - 0s 3ms/step - loss: 140499424.0000 - val_loss: 1178114
Epoch 25/100
58/58 [==============================] - 0s 4ms/step - loss: 146311408.0000 - val_loss: 1165189
Epoch 26/100
58/58 [==============================] - 0s 3ms/step - loss: 143358864.0000 - val_loss: 1144772
Epoch 27/100
58/58 [==============================] - 0s 3ms/step - loss: 142638624.0000 - val_loss: 1124139
Epoch 28/100
58/58 [==============================] - 0s 3ms/step - loss: 134273888.0000 - val_loss: 1114108
Epoch 29/100
```

Schematic drawing of the model

```
plot_model(ann_model)
```

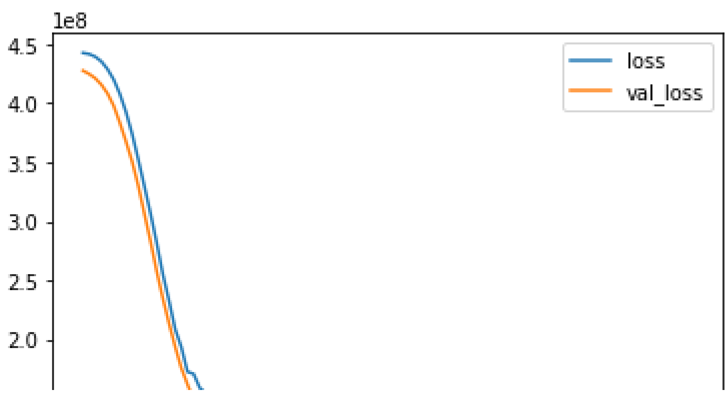


Drawing a graph of the process of changing the error rate of the model in each epoch of training:

1. The error rate on the training data has gradually decreased
2. The error rate on the validation data has gradually decreased

```
pd.DataFrame(history.history).plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe5003a34d0>
```



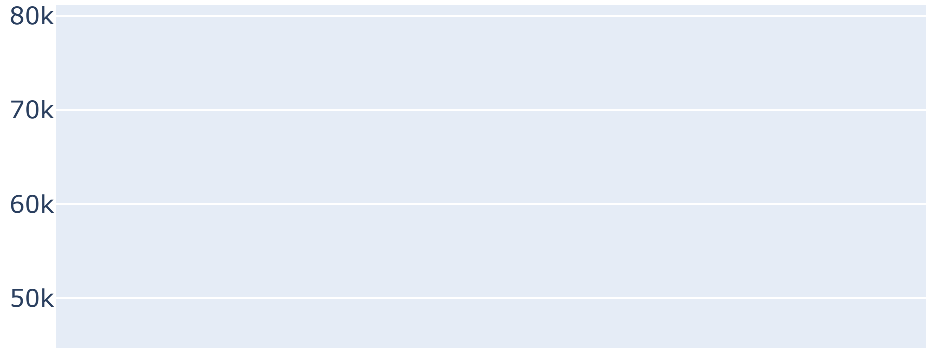
predict:

```
y_pred = ann_model.predict(x_test)
y_=pd.DataFrame(data=y_pred,index=y_test.index,columns=["y_pred"])
y_["y_test"]=y_test.values
```

```
y_.sort_index(inplace =True)
```

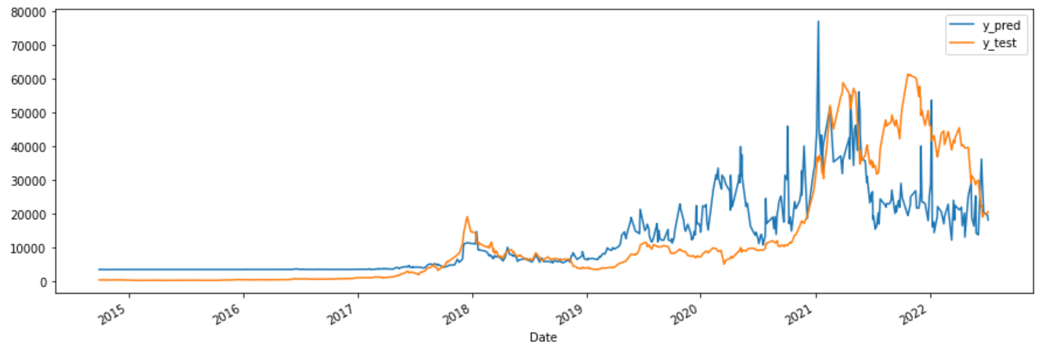
```
fig = px.bar(x= y_.index.values , y= y_["y_test"])
fig.add_scatter(x= y_.index.values , y= y_["y_test"])
```


```
fig = px.bar(x= y_.index.values , y= y_["y_pred"])
fig.add_scatter(x= y_.index.values , y= y_["y_pred"])
```




```
y_.plot(figsize = (15,5))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe5002141d0>
```



Claculation model error using different metrics:

1. r2_score
2. mean_squared_log_error
3. mean_absolute_error

```
print("r2 score:" ,r2_score(y_["y_test"],y_["y_pred"]))
print("mean squared error: ", mean_squared_log_error(y_["y_test"],y_["y_pred"]))
print("mean absolute erroe:",mean_absolute_error(y_["y_test"],y_["y_pred"]))
```

```
r2 score: 0.5278549482791974
mean squared error:  1.6350380349087086
mean absolute erroe: 7082.549899446985
```

Create and fit the LSTM network

```
lstm_model = Sequential()
lstm_model.add(LSTM(128, return_sequences=True, input_shape= (x_train.shape[1], 1)))
```

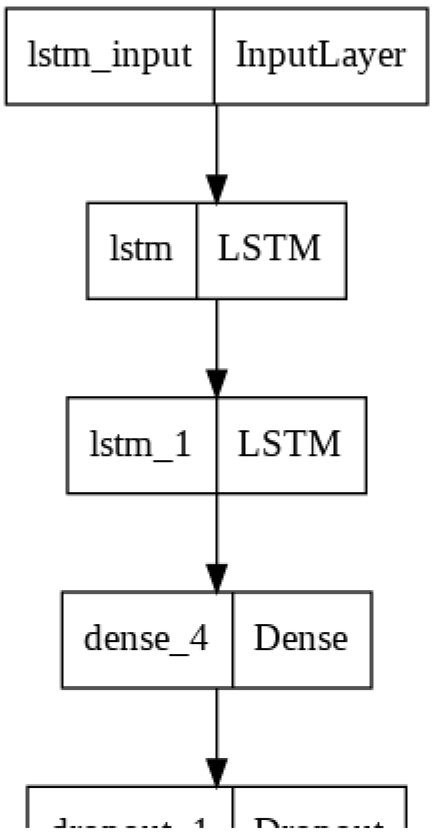
```
lstm_model.add(LSTM(15))
lstm_model.add(Dense(30))
lstm_model.add(Dropout(0.25))
lstm_model.add(Dense(1))

lstm_model.compile(loss='mean_squared_error', optimizer='adam')


history = lstm_model.fit(x_train,y_train,epochs=100,batch_size=32,validation_data=(x_val,y_val))
```

```
Epoch 1/100
58/58 [==============================] - 6s 43ms/step - loss: 442772960.0000 - val_loss: 428472
Epoch 2/100
58/58 [==============================] - 1s 12ms/step - loss: 442419392.0000 - val_loss: 428056
Epoch 3/100
58/58 [==============================] - 1s 12ms/step - loss: 441935360.0000 - val_loss: 427513
Epoch 4/100
58/58 [==============================] - 1s 12ms/step - loss: 441311136.0000 - val_loss: 426795
Epoch 5/100
58/58 [==============================] - 1s 12ms/step - loss: 440465152.0000 - val_loss: 425900
Epoch 6/100
58/58 [==============================] - 1s 12ms/step - loss: 439476224.0000 - val_loss: 424817
Epoch 7/100
58/58 [==============================] - 1s 13ms/step - loss: 438247232.0000 - val_loss: 423569
Epoch 8/100
58/58 [==============================] - 1s 13ms/step - loss: 436869184.0000 - val_loss: 422116
Epoch 9/100
58/58 [==============================] - 1s 13ms/step - loss: 435339072.0000 - val_loss: 420536
Epoch 10/100
58/58 [==============================] - 1s 13ms/step - loss: 433522912.0000 - val_loss: 418778
Epoch 11/100
58/58 [==============================] - 1s 13ms/step - loss: 431789440.0000 - val_loss: 416940
Epoch 12/100
58/58 [==============================] - 1s 13ms/step - loss: 429763584.0000 - val_loss: 415019
Epoch 13/100
58/58 [==============================] - 1s 12ms/step - loss: 427871872.0000 - val_loss: 413015
Epoch 14/100
58/58 [==============================] - 1s 12ms/step - loss: 425555200.0000 - val_loss: 410789
Epoch 15/100
58/58 [==============================] - 1s 12ms/step - loss: 423330400.0000 - val_loss: 408516
Epoch 16/100
58/58 [==============================] - 1s 12ms/step - loss: 420964000.0000 - val_loss: 406150
Epoch 17/100
58/58 [==============================] - 1s 12ms/step - loss: 418487040.0000 - val_loss: 403673
Epoch 18/100
58/58 [==============================] - 1s 12ms/step - loss: 415990400.0000 - val_loss: 401091
Epoch 19/100
58/58 [==============================] - 1s 12ms/step - loss: 412952256.0000 - val_loss: 398370
Epoch 20/100
58/58 [==============================] - 1s 12ms/step - loss: 410610656.0000 - val_loss: 395598
Epoch 21/100
58/58 [==============================] - 1s 13ms/step - loss: 407249600.0000 - val_loss: 392672
Epoch 22/100
58/58 [==============================] - 1s 12ms/step - loss: 404555296.0000 - val_loss: 389700
Epoch 23/100
58/58 [==============================] - 1s 13ms/step - loss: 401259712.0000 - val_loss: 386617
Epoch 24/100
58/58 [==============================] - 1s 13ms/step - loss: 398308896.0000 - val_loss: 383525
Epoch 25/100
```

```
58/58 [==============================] - 1s 12ms/step - loss: 394723456.0000 - val_loss: 380132
Epoch 26/100
58/58 [==============================] - 1s 12ms/step - loss: 391166880.0000 - val_loss: 376838
Epoch 27/100
58/58 [==============================] - 1s 12ms/step - loss: 387758752.0000 - val_loss: 373504
Epoch 28/100
58/58 [==============================] - 1s 12ms/step - loss: 384403104.0000 - val_loss: 370252
Epoch 29/100
```

Schematic drawing of the model

```
plot_model(lstm_model)
```

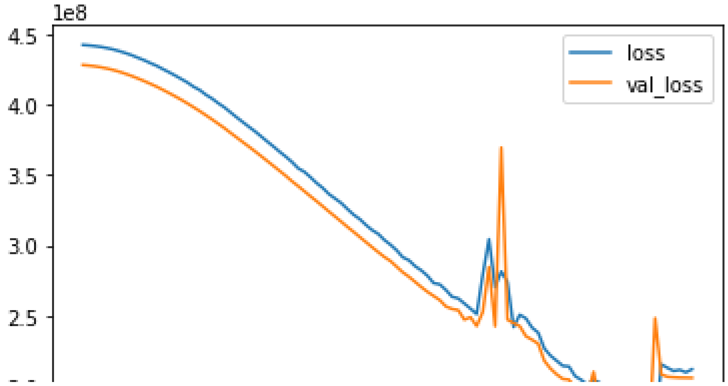


Drawing a graph of the process of changing the error rate of the model in each epoch of training:

1. The error rate on the training data has gradually decreased
2. The error rate on the validation data has gradually decreased

```
pd.DataFrame(history.history).plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe4ff5c7b90>
```

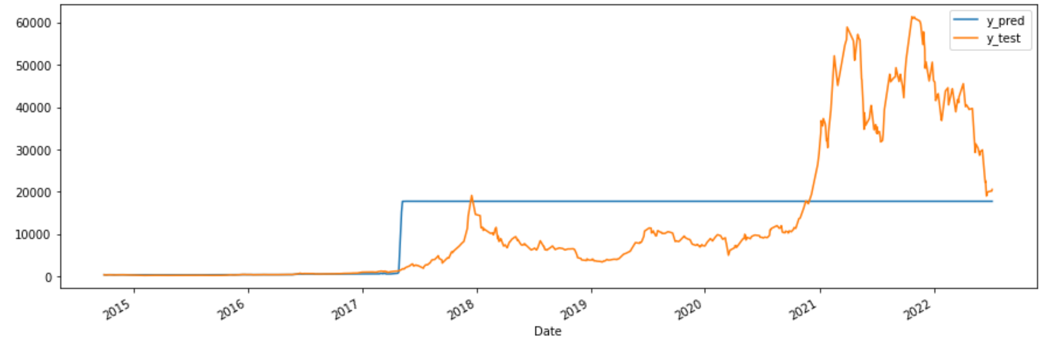


make predictions

```
y_pred = lstm_model.predict(x_test)
y_lstm=pd.DataFrame(data=y_pred,index=y_test.index,columns=["y_pred"])
y_lstm["y_test"]=y_test.values
y_lstm.sort_index(inplace =True)


fig = px.bar(x= y_lstm.index.values , y= y_lstm["y_pred"])
fig.add_scatter(x= y_lstm.index.values , y= y_lstm["y_pred"])

fig.add_bar(x= y_lstm.index.values , y= y_lstm["y_test"])
fig.add_scatter(x= y_lstm.index.values , y= y_lstm["y_test"])
```

```
y_lstm.plot(figsize = (15,5))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe50976e210>
```



Claculation model error using different metrics:

1. r2_score
2. mean_squared_log_error
3. mean_absolute_error

```
print("r2 score:" ,r2_score(y_lstm["y_test"],y_lstm["y_pred"]))
print("mean squared error: ", mean_squared_log_error(y_lstm["y_test"],y_lstm["y_pred"]))
print("mean absolute erroe:",mean_absolute_error(y_lstm["y_test"],y_lstm["y_pred"]))
```

```
r2 score: 0.25484859679733984
mean squared error:  0.7146281692506509
mean absolute erroe: 9501.05085110372
```