In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense,LSTM,Dropout
```

In [2]:
```
data = pd.read_csv('data.csv')
data.head()
```

Out[2]:

	Date	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
0	08-10-2018	208.00	222.25	206.85	216.00	215.15	4642146	10062.83
1	05-10-2018	217.00	218.60	205.90	210.25	209.20	3519515	7407.06
2	04-10-2018	223.50	227.80	216.15	217.25	218.20	1728786	3815.79
3	03-10-2018	230.00	237.50	225.75	226.45	227.60	1708590	3960.27
4	01-10-2018	234.55	234.60	221.05	230.30	230.90	1534749	3486.05

In [3]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1235 entries, 0 to 1234
Data columns (total 8 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Date                  1235 non-null   object 
 1   Open                  1235 non-null   float64
 2   High                  1235 non-null   float64
 3   Low                   1235 non-null   float64
 4   Last                  1235 non-null   float64
 5   Close                 1235 non-null   float64
 6   Total Trade Quantity  1235 non-null   int64  
 7   Turnover (Lacs)       1235 non-null   float64
dtypes: float64(6), int64(1), object(1)
memory usage: 77.3+ KB
```

In [4]:
```
data["Close"]=pd.to_numeric(data.Close,errors='coerce')
data = data.dropna()
trainData = data.iloc[:,4:5].values
```

In [5]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1235 entries, 0 to 1234
Data columns (total 8 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Date                  1235 non-null   object 
 1   Open                  1235 non-null   float64
 2   High                  1235 non-null   float64
 3   Low                   1235 non-null   float64
 4   Last                  1235 non-null   float64
 5   Close                 1235 non-null   float64
 6   Total Trade Quantity  1235 non-null   int64  
 7   Turnover (Lacs)       1235 non-null   float64
dtypes: float64(6), int64(1), object(1)
memory usage: 77.3+ KB
```

In [6]:
```
sc = MinMaxScaler(feature_range=(0,1))
trainData = sc.fit_transform(trainData)
trainData.shape
```

Out[6]:
```
(1235, 1)
```

In [7]:
```
X_train = []
y_train = []

for i in range (60,1100):
    X_train.append(trainData[i-60:i,0])
    y_train.append(trainData[i,0])

X_train,y_train = np.array(X_train),np.array(y_train)
```

In [8]:
```
X_train = np.reshape(X_train,(X_train.shape[0],X_train.shape[1],1))
X_train.shape
```

Out[8]:
```
(1040, 60, 1)
```

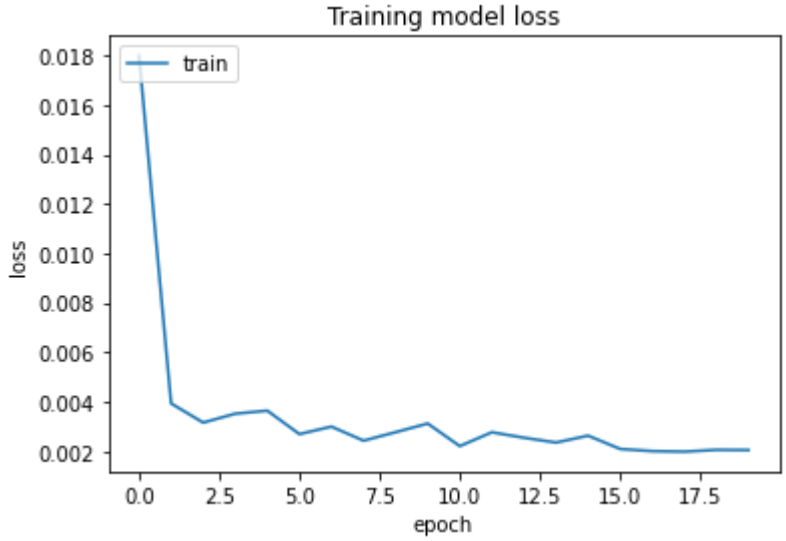
In [9]:
```
model = Sequential()

model.add(LSTM(units=100, return_sequences = True, input_shape =(X_train.shape[1],1)))
model.add(Dropout(0.2))

model.add(LSTM(units=100, return_sequences = True))
model.add(Dropout(0.2))

model.add(LSTM(units=100, return_sequences = True))
model.add(Dropout(0.2))

model.add(LSTM(units=100, return_sequences = False))
model.add(Dropout(0.2))

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

In [10]:
```
hist = model.fit(X_train, y_train, epochs = 20, batch_size = 32, verbose=2)
```

```
Epoch 1/20
33/33 - 12s - loss: 0.0180 - 12s/epoch - 357ms/step
Epoch 2/20
33/33 - 5s - loss: 0.0039 - 5s/epoch - 156ms/step
Epoch 3/20
33/33 - 5s - loss: 0.0032 - 5s/epoch - 150ms/step
Epoch 4/20
33/33 - 5s - loss: 0.0035 - 5s/epoch - 141ms/step
Epoch 5/20
33/33 - 5s - loss: 0.0036 - 5s/epoch - 155ms/step
Epoch 6/20
33/33 - 4s - loss: 0.0027 - 4s/epoch - 134ms/step
Epoch 7/20
33/33 - 4s - loss: 0.0030 - 4s/epoch - 124ms/step
Epoch 8/20
33/33 - 4s - loss: 0.0024 - 4s/epoch - 127ms/step
Epoch 9/20
33/33 - 4s - loss: 0.0028 - 4s/epoch - 133ms/step
Epoch 10/20
33/33 - 4s - loss: 0.0031 - 4s/epoch - 126ms/step
Epoch 11/20
33/33 - 4s - loss: 0.0022 - 4s/epoch - 124ms/step
Epoch 12/20
33/33 - 4s - loss: 0.0028 - 4s/epoch - 126ms/step
Epoch 13/20
33/33 - 4s - loss: 0.0025 - 4s/epoch - 125ms/step
Epoch 14/20
33/33 - 4s - loss: 0.0023 - 4s/epoch - 124ms/step
Epoch 15/20
33/33 - 4s - loss: 0.0026 - 4s/epoch - 127ms/step
Epoch 16/20
33/33 - 4s - loss: 0.0021 - 4s/epoch - 127ms/step
Epoch 17/20
33/33 - 4s - loss: 0.0020 - 4s/epoch - 127ms/step
Epoch 18/20
33/33 - 5s - loss: 0.0020 - 5s/epoch - 144ms/step
Epoch 19/20
33/33 - 4s - loss: 0.0021 - 4s/epoch - 132ms/step
Epoch 20/20
33/33 - 5s - loss: 0.0020 - 5s/epoch - 153ms/step
```

In [11]:

```
plt.plot(hist.history['loss'])
plt.title('Training model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train'], loc='upper left')
plt.show()
```



In [12]:

```
testData = pd.read_csv('data.csv')
testData["Close"]=pd.to_numeric(testData.Close,errors='coerce')
testData = testData.dropna()
```

```
testData = testData.iloc[:,4:5]
y_test = testData.iloc[60:,0:].values
#input array for the model
inputClosing = testData.iloc[:,0:].values
inputClosing_scaled = sc.transform(inputClosing)
inputClosing_scaled.shape
X_test = []
length = len(testData)
timestep = 60
for i in range(timestep,length):
    X_test.append(inputClosing_scaled[i-timestep:i,0])
X_test = np.array(X_test)
X_test = np.reshape(X_test,(X_test.shape[0],X_test.shape[1],1))
X_test.shape
```

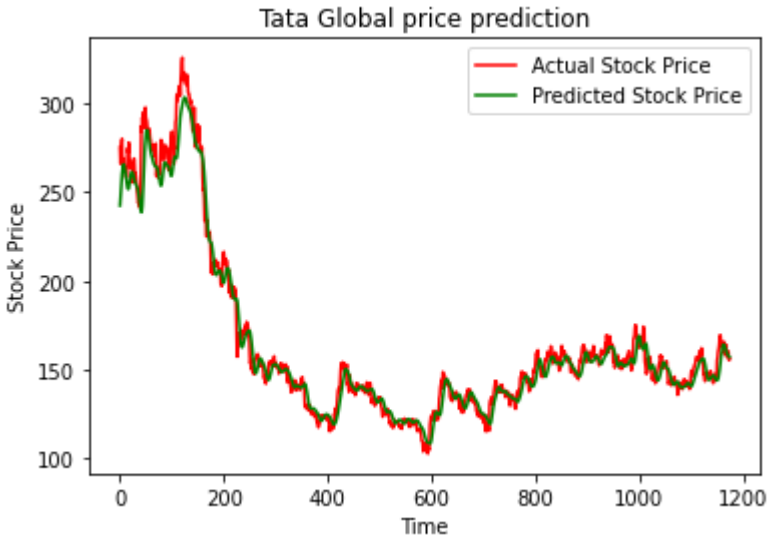
Out[12]:
```
(1175, 60, 1)
```

In [13]:
```
y_pred = model.predict(X_test)
y_pred
```

Out[13]:
```
37/37 [==============================] - 3s 45ms/step
array([[0.62653685],
       [0.64871967],
       [0.6709529 ],
       ...,
       [0.24676874],
       [0.2452808 ],
       [0.24392778]], dtype=float32)
```

In [14]:
```
predicted_price = sc.inverse_transform(y_pred)
```

In [15]:
```
plt.plot(y_test, color = 'red', label = 'Actual Stock Price')
plt.plot(predicted_price, color = 'green', label = 'Predicted Stock Price')
plt.title('Tata Global price prediction')
plt.xlabel('Time')
plt.ylabel('Stock Price')
plt.legend()
plt.show()
```



In [ ]: