# AI PROJECT

NAME – VEDANT RAMAKANT SHITALE
ROLL NO – 51
REG NO – 11810958
SECTION – K18SP
Git hub link-
https://github.com/vedantshitale/python/blob/master/Stock%20prediction%20us

# INDEX

# Abstract:

In Stock Market Prediction, the aim is to predict the future value of the financial stocks of a company. The recent trend
in stock market prediction technologies is the use of machine learning which makes predictions based on the values of current stock market indices by training on their previous values.

# INTRODUCTION:

Predicting how the stock market will perform is one of the most difficult things to do. There are so many factors involved in the prediction – physical factors vs. physiological, rational and irrational behavior, etc. All these aspects combine to make share prices volatile and very difficult to predict with a high degree of accuracy.

Can we use machine learning as a game changer in this domain? Using features like the latest announcements about an organization, their quarterly revenue results, etc., machine learning techniques have the potential to unearth patterns and insights we didn't see before, and these can be used to make unerringly accurate predictions.

Basically AI uses mathematical logics for the calculation of

Future stock this is basically starts with simple algorithms like averaging and linear regression, and then move on to advanced techniques like Auto ARIMA and LSTM.

Literal Review: -

Basically this process of predicting the future stock is effective we can use many methods for finding the future stock prices using machine learning, linear regration fundamental analysis but the best of all is through machine learning but why? This because machine learning uses some major library and major logics with that it also learns from the mistake. Basically the prediction is based on LSTM stands for long and short term memory it is one of effective method. The process is so efficient that it will give you 90% of effective predictions. Now a days many big companies like Goldman sack is highering data-science and machine learning experts to predict stocks .

Methodology: - The methodology used for the predictions as mentioned are Lstm, linear regreation are used for the stock predictions. Basically LSTM is part of deep learning it has feedback connections

It can not only process single data but also entire sequence of data so it remembers data / old data on that basis it predicts the future data. It works with machine learning but how?

1. 1st we take csv file online and the we will preprocess it

2. Then we take the data for comparing through this csv file which contain the data of stocks of some years from it we will train data we will make x and y axis on that basis we will divide the data as the data will come in form of rows and matrix we will convert it into a 2d matrix for converting it to a gram basically 2d graph using min max and lstm we will take a range of 2 years data for further prediction then the rows and columns will be made on that basis then the library find the rms(root mean square) from data of the matrix for getting the exaction location of further stock prices and mark them on graph.
3. We have basically used numpy, pandas and matplotlib to covert data to proper dataframes, handle data and to make linear graph so by seeing that graph the viewer can get idea of future stocks

# Program (with lines explained): -

```
import math
import pandas_datareader as web
import numpy as np
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from tensorflow.python.keras import Sequential
from tensorflow.python.keras.layers import Dense,LSTM
import matplotlib.pyplot as plt
plt.style.use('fivethirtyeight')
a=input("enter the company name")
b=input("enter the starting dates from where you want to take date to pred
ict")
c=input("enter the ending date")
#Get the stock quote
df = web.DataReader(a, data_source='yahoo', start=b, end=c)
#Show teh data
df
#Get the number of rows and columns in the data set
df.shape

#Visualize the closing price history
plt.figure(figsize=(16,8))
plt.title('Close Price History')
plt.plot(df['Close'])
plt.xlabel('Date', fontsize=18)
plt.ylabel('Close Price USD ($)', fontsize=18)
plt.show()
#Create a new dataframe with only the 'Close column
data = df.filter(['Close'])
#Convert the dataframe to a numpy array
dataset = data.values
#Get the number of rows to train the model on
training_data_len = math.ceil( len(dataset) * .8 )

training_data_len

#Scale the data
scaler = MinMaxScaler(feature_range=(0,1))
scaled_data = scaler.fit_transform(dataset)
```

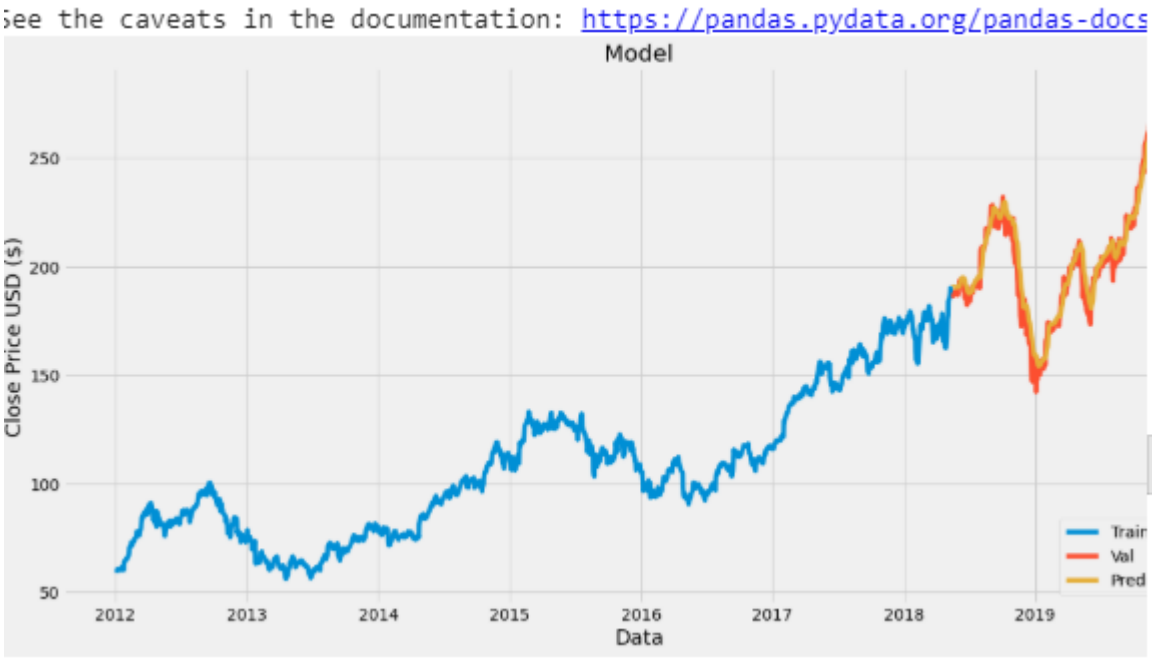
```
scaled_data
#Create the training data set
#Create the scaled training data set
train_data = scaled_data[0:training_data_len , :]
#Split the data into x_train and y_train data sets
x_train = []
y_train = []

for i in range(60, len(train_data)):
  x_train.append(train_data[i-60:i, 0])
  y_train.append(train_data[i, 0])
  if i<= 61:
    print(x_train)
    print(y_train)
    print()
#Convert the x_train and y_train to numpy arrays
x_train, y_train = np.array(x_train), np.array(y_train)
#Reshape the data
x_train = np.reshape(x_train, (x_train.shape[0], x_train.shape[1], 1))
x_train.shape
#Build the LSTM model
model = Sequential()
model.add(LSTM(50, return_sequences=True, input_shape= (x_train.shape[1],
1)))
model.add(LSTM(50, return_sequences= False))
model.add(Dense(25))
model.add(Dense(1))
#Compile the model
model.compile(optimizer='adam', loss='mean_squared_error')
#Train the model
model.fit(x_train, y_train, batch_size=1, epochs=1)
#create the testing data set
#create new array containing scaled valude from index 1540 to 2003
test_data = scaled_data[training_data_len - 60: ,:]
#create the data sets x_test and y_test
x_test = []
y_test = dataset[training_data_len:, :]
for i in range(60,len(test_data)):
  x_test.append(test_data[i-60:i,0])
  #convert the dta into numpy
x_test=np.array(x_test)
#reshape the data
x_test = np.reshape(x_test , (x_test.shape[0], x_test.shape[1], 1))
#get the models predicted values
```

```
predictions = model.predict(x_test)
predictions=scaler.inverse_transform(predictions)
#get the root mean swure  error (RMSE)
rmse = np.sqrt(np.mean(predictions-y_test)**2)
rmse
#plot data
train=data[:training_data_len]
valid=data[training_data_len:]
valid['Predictions'] = predictions
#Visualization
plt.figure(figsize=(16,8))
plt.title('Model')
plt.xlabel('Data', fontsize=18)
plt.ylabel('Close Price USD ($)', fontsize=18)
plt.plot(train['Close'])
plt.plot(valid[['Close','Predictions']])
plt.legend(['Train','Val','Predictions'], loc='lower right')
plt.show()
```

## Result:

```
enter the company nameaapl
enter the starting dates from where you want to take date to predict2012-(
enter the ending date2019-12-17
```




From this images the result or the execution we get the future prices of the stocks for showing the acuracy the data was compared with the stock prices as we see and using this machine learning we really get the prediction over 90% so it's a good no as you can see the from the blue line is the data from which the program learned to predict data the red line is the value of stock  and the yellow line is the prediction line and they are very close .So here are the predicted results before you.

Conclusion:

1. From this we can conclude the digital/technical prediction of the stock if efficiently carried out by AI and without mistakes.
2. We also can conclude from this that the prediction time taken by this device can be very fast and time saving which is one of the major factor