CS 483 HW 3

April 6, 2022

[1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn import metrics
%matplotlib inline
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import matplotlib
plt.style.use('bmh')

#read the file
#Perform analysis on Apple stock first
df = pd.read_csv('C://Users//huyvu//Downloads//AAPL.csv')

# The timeframe I decided to use is from 3/29/2021 to 9/27/2021 which is 6␣
 ↪months
#The reason I decided to use 6-month timeframe is because we only look 3 weeks␣
 ↪into the future
#Stocks are changes based on news and events happening at the present
#Since we only look into a short future, I think datasets limit to 6 weeks is␣
 ↪enough and it would limit
#some events that affected stock prices drop such as covid or when the company␣
 ↪was just established

#print the head
df.head()
```

[1]:

```
         Date        Open        High         Low       Close   Adj Close  \
0  2021-03-29  121.650002  122.580002  120.730003  121.389999  121.002861
1  2021-03-30  120.110001  120.400002  118.860001  119.900002  119.517616
2  2021-03-31  121.650002  123.519997  121.150002  122.150002  121.760445
3  2021-04-01  123.660004  124.180000  122.489998  123.000000  122.607727
4  2021-04-05  123.870003  126.160004  123.070000  125.900002  125.498489

     Volume
0  80819200
1  85671900
```

```
2   118323800
3    75089100
4    88651200
```

[4]:

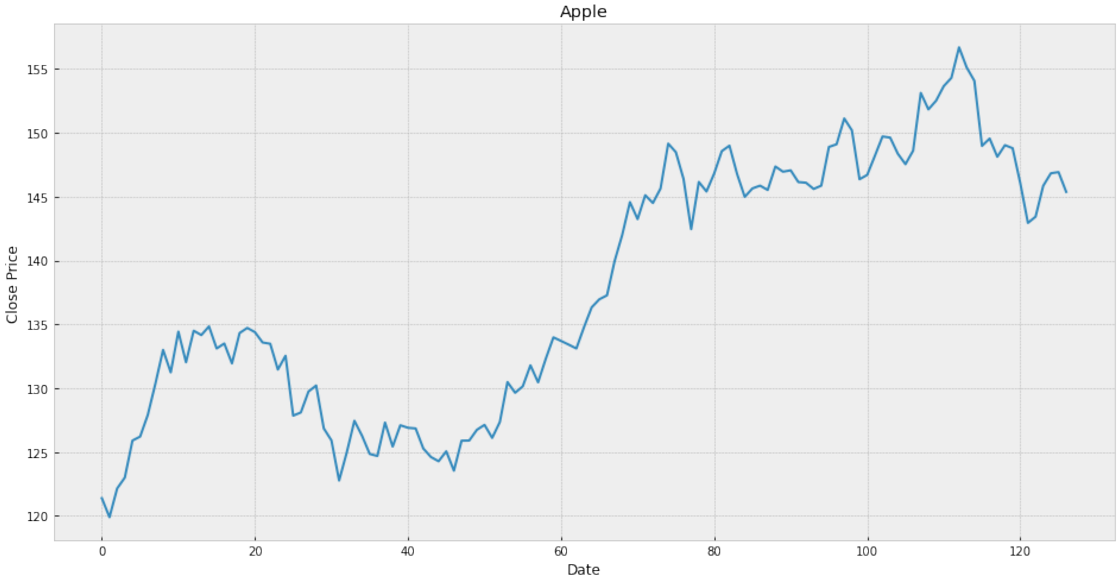
```
#shape of dataset
df.shape
```

[4]:

```
(127, 7)
```

[2]:

```
plt.figure(figsize=(16,8))
plt.title('Apple')
plt.xlabel('Date')
plt.ylabel('Close Price')
plt.plot(df['Close'])
plt.show()
```



[9]:

```
#I'm going to just focus on the Close Price
df = df[['Close']]
df.head(5)
```

[9]:

```
        Close
0  121.389999
1  119.900002
2  122.150002
3  123.000000
4  125.900002
```

[10]:

```
#Create a variable to predict 'x' days to the future
#for our project, 3 weeks = 21 days
future_days = 21
df['Prediction'] = df[['Close']].shift(-future_days)
df.tail(5)
#at this point, prediction is shown NA
```

[10]:

```
          Close  Prediction
122  143.429993         NaN
123  145.850006         NaN
124  146.830002         NaN
125  146.919998         NaN
126  145.369995         NaN
```

[11]:

```
#Create feature data set (X) and convert it to numpy array
#without the shifted days
X = np.array(df.drop(['Prediction'],1))[:-future_days]
print(X)
```

```
[[121.389999]
 [119.900002]
 [122.150002]
 [123.      ]
 [125.900002]
 [126.209999]
 [127.900002]
 [130.360001]
 [133.      ]
 [131.240005]
 [134.429993]
 [132.029999]
 [134.5     ]
 [134.160004]
 [134.839996]
 [133.110001]
 [133.5     ]
 [131.940002]
 [134.320007]
 [134.720001]
 [134.389999]
 [133.580002]
 [133.479996]
 [131.460007]
 [132.539993]
 [127.849998]
 [128.100006]
 [129.740005]
 [130.210007]
```

```
[126.849998]
[125.910004]
[122.769997]
[124.970001]
[127.449997]
[126.269997]
[124.849998]
[124.690002]
[127.309998]
[125.43    ]
[127.099998]
[126.900002]
[126.849998]
[125.279999]
[124.610001]
[124.279999]
[125.059998]
[123.540001]
[125.889999]
[125.900002]
[126.739998]
[127.129997]
[126.110001]
[127.349998]
[130.479996]
[129.639999]
[130.149994]
[131.789993]
[130.460007]
[132.300003]
[133.979996]
[133.699997]
[133.410004]
[133.110001]
[134.779999]
[136.330002]
[136.960007]
[137.270004]
[139.960007]
[142.020004]
[144.570007]
[143.240005]
[145.110001]
[144.5     ]
[145.639999]
[149.149994]
[148.479996]
[146.389999]
```

```
 [142.449997]
 [146.149994]
 [145.399994]
 [146.800003]
 [148.559998]
 [148.990005]
 [146.770004]
 [144.979996]
 [145.639999]
 [145.860001]
 [145.520004]
 [147.360001]
 [146.949997]
 [147.059998]
 [146.139999]
 [146.089996]
 [145.600006]
 [145.860001]
 [148.889999]
 [149.100006]
 [151.119995]
 [150.190002]
 [146.360001]
 [146.699997]
 [148.190002]
 [149.710007]
 [149.619995]
 [148.360001]
 [147.539993]]
```

[13]:

```
#Create the target data set (y) and convert it to numpy array
y=np.array(df['Prediction'])[:-future_days]
print(y)
```

```
[133.580002 133.479996 131.460007 132.539993 127.849998 128.100006
 129.740005 130.210007 126.849998 125.910004 122.769997 124.970001
 127.449997 126.269997 124.849998 124.690002 127.309998 125.43
 127.099998 126.900002 126.849998 125.279999 124.610001 124.279999
 125.059998 123.540001 125.889999 125.900002 126.739998 127.129997
 126.110001 127.349998 130.479996 129.639999 130.149994 131.789993
 130.460007 132.300003 133.979996 133.699997 133.410004 133.110001
 134.779999 136.330002 136.960007 137.270004 139.960007 142.020004
 144.570007 143.240005 145.110001 144.5      145.639999 149.149994
 148.479996 146.389999 142.449997 146.149994 145.399994 146.800003
 148.559998 148.990005 146.770004 144.979996 145.639999 145.860001
 145.520004 147.360001 146.949997 147.059998 146.139999 146.089996
 145.600006 145.860001 148.889999 149.100006 151.119995 150.190002
 146.360001 146.699997 148.190002 149.710007 149.619995 148.360001
 147.539993 148.600006 153.119995 151.830002 152.509995 153.649994
 154.300003 156.690002 155.110001 154.070007 148.970001 149.550003
 148.119995 149.029999 148.789993 146.059998 142.940002 143.429993
 145.850006 146.830002 146.919998 145.369995]
```

```
[14]: #Split data into 75% training and 25% testing
      X_train, X_test, y_train, y_test = train_test_split(X,y, test_size = 0.25)
```

```
[15]: #Create the linear regression model
      reg = LinearRegression().fit(X_train, y_train)
```

```
[16]: #Now we use the last x rows
      x_future = df.drop(['Prediction'],1)[:-future_days]
      x_future = x_future.tail(future_days)
      x_future = np.array(x_future)
      x_future
```

```
[16]: array([[145.639999],
             [145.860001],
             [145.520004],
             [147.360001],
             [146.949997],
             [147.059998],
             [146.139999],
             [146.089996],
             [145.600006],
             [145.860001],
             [148.889999],
             [149.100006],
             [151.119995],
             [150.190002],
             [146.360001],
             [146.699997],
             [148.190002],
             [149.710007],
             [149.619995],
             [148.360001],
             [147.539993]])
```

```
[18]: #Show the linear regression prediction
      reg_prediction = reg.predict(x_future)
      print(reg_prediction)
      #These are Apple Close prices predicted for 3 weeks after 9/27/2021
```

```
[146.79831572 146.96611635 146.70679275 148.11020115 147.79748159
 147.8813819  147.17967732 147.14153887 146.76781213 146.96611635
 149.27716612 149.43734333 150.9780358  150.26870857 147.34747795
 147.60680079 148.74326217 149.90260524 149.833951   148.87292435
148.24748522]
```

[25]:

```python
#Now we do the same steps for Nvidia
df = pd.read_csv('C://Users//huyvu//Downloads//NVDA.csv')
df.head()
```

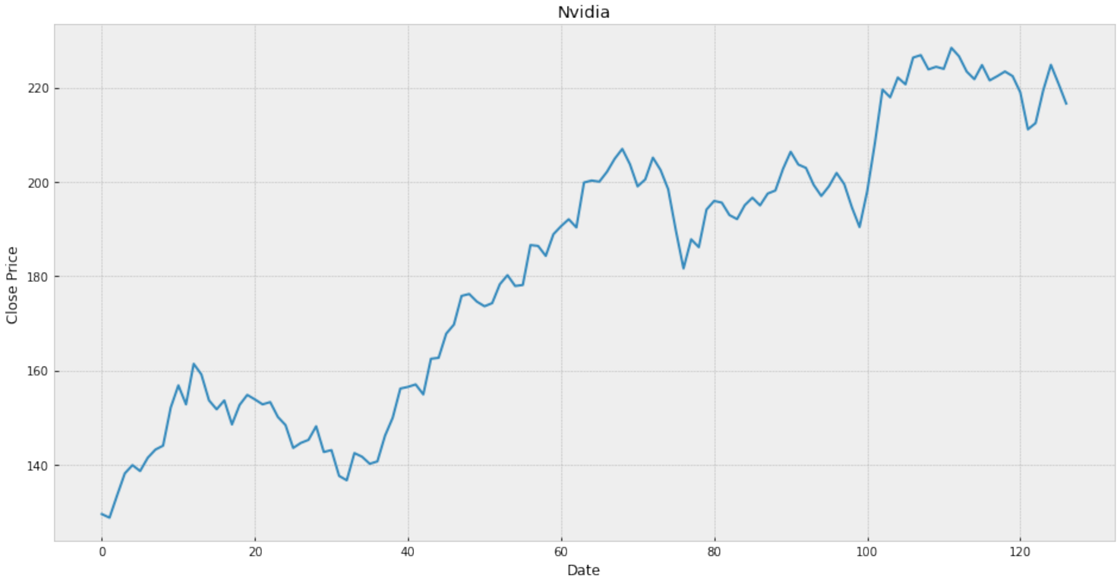
[25]:

```
         Date        Open        High         Low       Close   Adj Close  \
0  2021-03-29  128.202499  130.625000  127.000000  129.482498  129.430008
1  2021-03-30  128.419998  129.752502  127.050003  128.717499  128.665329
2  2021-03-31  130.154999  134.705002  129.824997  133.482498  133.428391
3  2021-04-01  135.722504  138.699997  135.112503  138.117493  138.061493
4  2021-04-05  138.675003  140.139999  137.330002  139.875000  139.818298

     Volume
0  27352000
1  20020400
2  31477600
3  30827600
4  25567200
```

[31]:

```python
plt.figure(figsize=(16,8))
plt.title('Nvidia')
plt.xlabel('Date')
plt.ylabel('Close Price')
plt.plot(df['Close'])
plt.show()
```



[27]:

```
df = df[['Close']]
df.head(5)
```

[27]:

```
        Close
0  129.482498
1  128.717499
2  133.482498
3  138.117493
4  139.875000
```

[28]:

```
future_days = 21
df['Prediction'] = df[['Close']].shift(-future_days)
df.tail(5)
```

[28]:

```
          Close  Prediction
122  212.460007         NaN
123  219.410004         NaN
124  224.820007         NaN
125  220.809998         NaN
126  216.600006         NaN
```

[29]:

```
#Create a variable to predict 'x' days to the future
#for our project, 3 weeks = 21 days
future_days = 21
df['Prediction'] = df[['Close']].shift(-future_days)
df.tail(5)
#at this point, prediction is shown NA
```

[29]:

```
          Close  Prediction
122  212.460007         NaN
123  219.410004         NaN
124  224.820007         NaN
125  220.809998         NaN
126  216.600006         NaN
```

[30]:

```
#Create feature data set (X) and convert it to numpy array
#without the shifted days
X = np.array(df.drop(['Prediction'],1))[:-future_days]

#Create the target data set (y) and convert it to numpy array
y=np.array(df['Prediction'])[:-future_days]
#Split data into 75% training and 25% testing
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size = 0.25)
#Create the linear regression model
reg = LinearRegression().fit(X_train, y_train)
#Now we use the last x rows
x_future = df.drop(['Prediction'],1)[:-future_days]
x_future = x_future.tail(future_days)
x_future = np.array(x_future)
#Show the linear regression prediction
reg_prediction = reg.predict(x_future)
print(reg_prediction)
#These are Nvida Close prices predicted for 3 weeks after 9/27/2021
```

```
[208.24545195 206.91875906 208.96171178 209.49075945 213.22669337
 216.18124027 213.97550541 213.39761153 210.47561683 208.54661308
 210.22330109 212.52671615 210.5895658  206.58504655 203.18282515
 209.35239349 217.63817694 226.93322174 225.59023485 229.00873805
 227.82853412]
```

[ ]:

```

```