Financial Timeseries Test

August 21, 2021

1 Introduction to the Task

We would like to build a model that can predict the daily closing price of US stocks. Because a company's historical price data alone does not contain all information about what the future close prices might be, please be sure to include contextual information about the company, such as industry/sector, in your analysis.

For this task, please use the following datasets: - stocks dataset https://hr-projects-assets-prod.s3.amazonaws.com/3maoh37pfp9/8c4c02a9a49365719dee849a8bfeb64e/Stocks2.zip - stocks dataset cont https://hr-projects-assets-prod.s3.amazonaws.com/3maoh37pfp9/193bb5ff397532fa61b2a6417a125e90
- etfs https://hr-projects-assets-prod.s3.amazonaws.com/3maoh37pfp9/874ba49094e7d05cef0d82fc2fd8ef1a/ETFs.z
- company info https://hr-projects-assets-prod.s3.amazonaws.com/3maoh37pfp9/4cfaa8631b61675dfa033b316ad3b

You may limit your analysis to any date range you choose. If you choose to work with a subset companies in the given dataset, please justify how you selected which companies to include.

[1]:
```
# If you'd like to install packages that aren't installed by default, list them␣
↪here.
# This will ensure your notebook has all the dependencies and works everywhere

import sys
!{sys.executable} -m pip install xgboost sklearn
```

```
Requirement already satisfied: xgboost in /opt/conda/lib/python3.7/site-packages
(1.4.2)
Requirement already satisfied: sklearn in /opt/conda/lib/python3.7/site-packages
(0.0)
Requirement already satisfied: numpy in /opt/conda/lib/python3.7/site-packages
(from xgboost) (1.19.2)
Requirement already satisfied: scipy in /opt/conda/lib/python3.7/site-packages
(from xgboost) (1.3.3)
Requirement already satisfied: scikit-learn in /opt/conda/lib/python3.7/site-
packages (from sklearn) (0.21.3)
Requirement already satisfied: joblib>=0.11 in /opt/conda/lib/python3.7/site-
packages (from scikit-learn->sklearn) (0.17.0)
```

[2]:
```
import pandas as pd
import seaborn as sns
```

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.metrics import mean_squared_error
```

2 Step 1: Load, merge, and explore the data

What does the data look like? What are some general characteristics? Are there any anomalies that might impact the later modeling steps?

[3]:
```
companies = pd.read_csv("https://hr-projects-assets-prod.s3.amazonaws.com/
↪3maoh37pfp9/4cfaa8631b61675dfa033b316ad3bbe5/companies.csv")
companies.head()
```

[3]:
```
  ticker                  company name                     short name  \
0      A        Agilent Technologies Inc.                       Agilent
1     AA             Alcoa Corporation                           Alcoa
2   AABA                   Altaba Inc.                          Altaba
3    AAC             AAC Holdings Inc.                             AAC
4   AADR  AdvisorShares Dorsey Wright ADR  AdvisorShares Dorsey Wright

                          industry  \
0  Medical Diagnostics & Research
1                  Metals & Mining
2                 Asset Management
3            Health Care Providers
4                              NaN

                                         description  \
0  Agilent Technologies Inc is engaged in life sc…
1  Alcoa Corp is an integrated aluminum company. …
2  Altaba Inc is an independent, non-diversified,…
3  AAC Holdings Inc provides inpatient and outpat…
4  The investment seeks long-term capital appreci…

                                  website      logo                     ceo  \
0                  http://www.agilent.com     A.png      Michael R. McMullen
1                    http://www.alcoa.com    AA.png  Roy Christopher Harvey
2                   http://www.altaba.com  AABA.png      Thomas J. Mcinerney
3  http://www.americanaddictioncenters.org       NaN    Michael T. Cartwright
4              http://www.advisorshares.com  AADR.png                     NaN

              exchange    market cap              sector  \
```

```
0  New York Stock Exchange  2.421807e+10         Healthcare
1  New York Stock Exchange  5.374967e+09    Basic Materials
2     Nasdaq Global Select  4.122368e+10 Financial Services
3  New York Stock Exchange  6.372010e+07         Healthcare
4               NYSE Arca  1.031612e+08                NaN

              tag 1                   tag 2                          tag 3
0          Healthcare  Diagnostics & Research  Medical Diagnostics & Research
1     Basic Materials                Aluminum                 Metals & Mining
2  Financial Services        Asset Management                             NaN
3          Healthcare            Medical Care           Health Care Providers
4                 NaN                     NaN                             NaN
```

[4]:
```
#Show all categories for "exchange" to see which one is related to NASDAQ
companies["exchange"].unique()
```

[4]:
```
array(['New York Stock Exchange', 'Nasdaq Global Select', 'NYSE Arca',
       'NYSE American', 'NASDAQ Global Market', 'NASDAQ Capital Market',
       'Cboe Global Markets EDGX', 'Investors Exchange', 'OTC Pink'],
      dtype=object)
```

[5]:
```
# To see which NASDAQ category has the maximum count
companies["exchange"].value_counts()
```

[5]:
```
New York Stock Exchange     2467
Nasdaq Global Select        1432
NASDAQ Capital Market        793
NYSE Arca                    745
NASDAQ Global Market         621
NYSE American                273
Cboe Global Markets EDGX      35
OTC Pink                       1
Investors Exchange             1
Name: exchange, dtype: int64
```

[6]:
```
# Filter just the companies that belong to "Nasdaq Global Select"
nas_df = companies[companies["exchange"] == "Nasdaq Global Select"]
nas_df.head()
```

[6]:
```
   ticker                 company name           short name  \
2    AABA                  Altaba Inc.               Altaba
5     AAL  American Airlines Group Inc.   American Airlines
10   AAON                    AAON Inc.                 AAON
12   AAPL                   Apple Inc.                Apple
15   AAWW  Atlas Air Worldwide Holdings  Atlas Air Worldwide

                      industry  \
```

```
2              Asset Management
5                      Airlines
10           Building Materials
12            Computer Hardware
15   Transportation & Logistics

                                          description  \
2   Altaba Inc is an independent, non-diversified,…
5   American Airlines Group Inc operates over 6,00…
10  AAON Inc is a heating, ventilation and air con…
12  Apple Inc is designs, manufactures and markets…
15  Atlas Air Worldwide Holdings Inc is engaged in…

                    website      logo                  ceo  \
2     http://www.altaba.com  AABA.png   Thomas J. Mcinerney
5         http://www.aa.com   AAL.png     W. Douglas Parker
10      http://www.aaon.com  AAON.png  Norman H. Asbjornson
12     http://www.apple.com  AAPL.png       Timothy D. Cook
15  http://www.atlasair.com  AAWW.png      William J. Flynn

                exchange    market cap              sector  \
2   Nasdaq Global Select  4.122368e+10  Financial Services
5   Nasdaq Global Select  1.694019e+10         Industrials
10  Nasdaq Global Select  1.961880e+09     Basic Materials
12  Nasdaq Global Select  8.074917e+11          Technology
15  Nasdaq Global Select  1.395183e+09         Industrials

                 tag 1                    tag 2                       tag 3
2   Financial Services         Asset Management                         NaN
5          Industrials                 Airlines                         NaN
10     Basic Materials       Building Materials                         NaN
12          Technology      Consumer Electronics           Computer Hardware
15         Industrials   Airports & Air Services  Transportation & Logistics
```

[7]:

```python
# Read the dataset from the company Altaba
df = pd.read_csv('aaba.us.txt', delimiter = ",")
```

[8]:

```python
df[["Date","Close"]]
```

[8]:

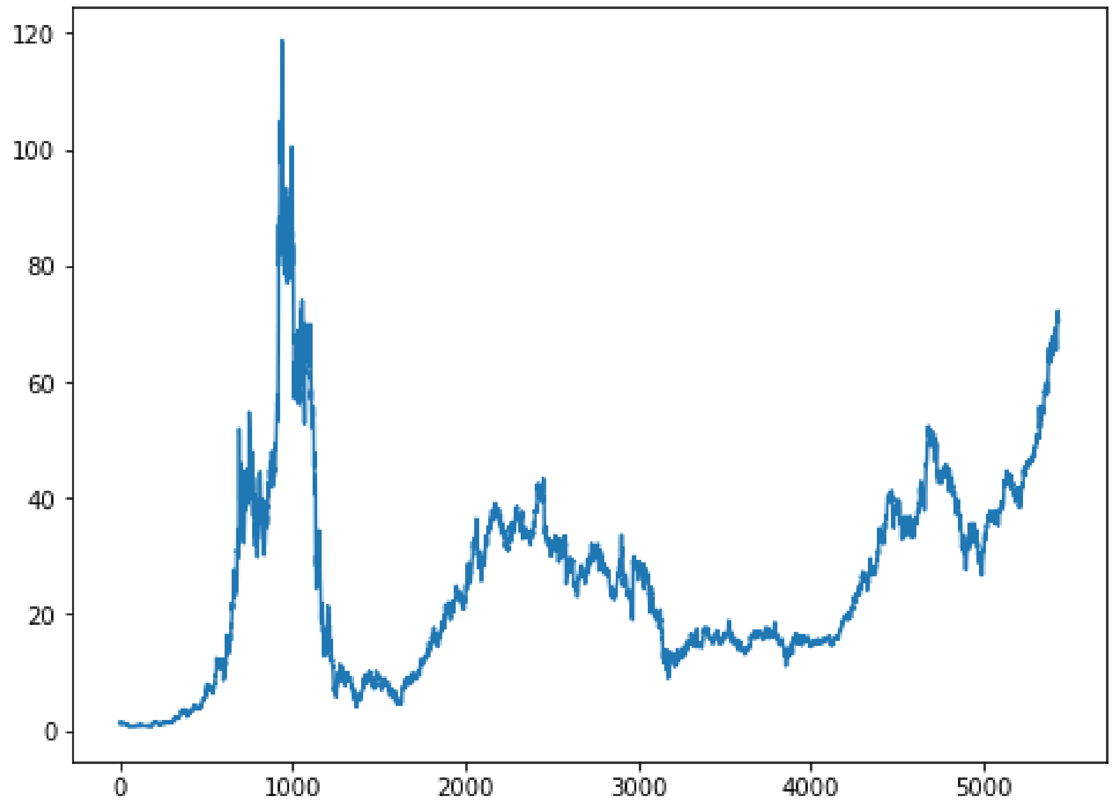
	Date	Close
0	1996-04-12	1.38
1	1996-04-15	1.34
2	1996-04-16	1.20
3	1996-04-17	1.12
4	1996-04-18	1.22
…	…	…
5429	2017-11-06	71.71

```
5430  2017-11-07  72.22
5431  2017-11-08  71.16
5432  2017-11-09  70.19
5433  2017-11-10  70.56

[5434 rows x 2 columns]
```

[9]:
```
# Plot the "Close" information
plt.figure(figsize=(8, 6))
plt.plot(df["Close"])
```

[9]:
```
[<matplotlib.lines.Line2D at 0x7f826b706ed0>]
```



3 Step 2: Build a Model

Given the data explored in the previous step, predict tomorrow's closing price for each company in the NASDAQ index. If it helps simplify the analysis, you may build a model that predicts closing price(s) for a subset of companies in the NASDAQ index.

[10]:
```
# This is a function that transforms a series into a

def transformSeriesToDataset(series, NumberOfElements):
    dataset = None
    outputDataset = None
    for counter in range (len(series)-NumberOfElements-1):
        sample = np.array([series[counter:counter+NumberOfElements]])
        output = np.array([series[counter+NumberOfElements]])
        if dataset is None:
            dataset = sample
        else:
            dataset = np.append(dataset,sample,axis = 0)
        if outputDataset is None:
            outputDataset = output
        else:
            outputDataset = np.append(outputDataset,output)
    return dataset, outputDataset
```

[11]:
```
# Create a series variable with the "Close" field

series = df['Close'].to_numpy()
X, Y = transformSeriesToDataset(series, NumberOfElements = 10)
```

[12]:
```
# Split the information in train and test

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.4)
```

[13]:
```
# Creation of a lineal model and fitting
lin_model = LinearRegression()
lin_model.fit(X_train, Y_train)
```

[13]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

[14]:
```
y_train_predict = lin_model.predict(X_train)
MSE = mean_squared_error(Y_train,y_train_predict)
print("Entrenamiento: MSE ="+str(MSE))

y_test_predict = lin_model.predict(X_test)
MSE = (mean_squared_error(Y_test, y_test_predict))
print("Pruebas: MSE =    "+str(MSE))

df_predictions = pd.DataFrame({'Real_Value':Y_test, 'Prediction':
 ↪y_test_predict, 'Diff':Y_test-y_test_predict})
df_predictions = df_predictions.reset_index(drop = True)
df_predictions.head(10)
```

```
Entrenamiento: MSE =1.1831113909006064
Pruebas: MSE =    1.1620926887716074
```

[14]:

	Real_Value	Prediction	Diff
0	29.64	29.333274	0.306726
1	17.31	17.472378	-0.162378
2	16.09	16.186748	-0.096748
3	8.52	8.898262	-0.378262
4	21.41	21.117678	0.292322
5	42.99	45.463895	-2.473895
6	4.33	4.472247	-0.142247
7	15.73	15.858349	-0.128349
8	11.42	10.767273	0.652727
9	23.72	23.612139	0.107861

4 Step 3: Evaluate the model on test data

How does the model perform on data that it hasn't seen? When predictions are poor, why does the model fail to predict the close price?

[15]:
```
# We will do the test in the American Airlines dataset, which belongs to the␣
↪"Nasdaq Global Select" exchange
df2 = pd.read_csv('aal.us.txt', delimiter = ",")
df2[["Date","Close"]].head(15)
```

[15]:

	Date	Close
0	2013-12-10	24.064
1	2013-12-11	25.139
2	2013-12-12	24.616
3	2013-12-13	25.369
4	2013-12-16	25.739
5	2013-12-17	25.245
6	2013-12-18	25.369
7	2013-12-19	25.265
8	2013-12-20	25.466
9	2013-12-23	25.324
10	2013-12-24	25.389
11	2013-12-26	25.275
12	2013-12-27	24.123
13	2013-12-30	23.967
14	2013-12-31	24.422

[17]:
```
# Transformation of the series into a dataframe
series2 = df2['Close'].to_numpy()
X, Y = transformSeriesToDataset(series2, NumberOfElements = 10)
```

[18]:
```python
# Prediction
y_predict = lin_model.predict(X)
```

[20]:
```python
# Calculation of the mean squared error
MSE = mean_squared_error(Y, y_predict)
print("Test data evaluation: MSE ="+str(MSE))
```

```
Test data evaluation: MSE =0.8404990422436212
```

[22]:
```python
# The comparison between the real value and the prediction
df2_predictions = pd.DataFrame({'Real_Value':Y, 'Prediction':y_predict, 'Diff':
 ↪Y-y_predict})
df2_predictions = df2_predictions.reset_index(drop = True)
df2_predictions.tail()
```

[22]:

	Real_Value	Prediction	Diff
973	47.346	47.120376	0.225624
974	47.406	47.283819	0.122181
975	46.358	47.519122	-1.161122
976	46.269	46.375744	-0.106744
977	45.670	46.360802	-0.690802

---

[58]:
```python
lin_model.predict(X[-2:])
```

[58]:
```
array([46.37574405, 46.36080185])
```

[65]:
```python
# Generates a df to predict the next one
last_last_array=np.array(df2_predictions["Prediction"][-11:-1])
last_array=np.array(df2_predictions["Prediction"][-10:])
last_df = last_last_array,last_array
```

[68]:
```python
last_df
```

[68]:
```
(array([50.83182221, 48.34180402, 47.51436499, 47.33626258, 46.71770821,
        47.94578569, 47.12037621, 47.28381897, 47.51912232, 46.37574405]),
 array([48.34180402, 47.51436499, 47.33626258, 46.71770821, 47.94578569,
        47.12037621, 47.28381897, 47.51912232, 46.37574405, 46.36080185]))
```

[74]:
```python
# Next one prediction
next_value = lin_model.predict(last_df)
print("The next value is:",round(next_value[1],2))
```

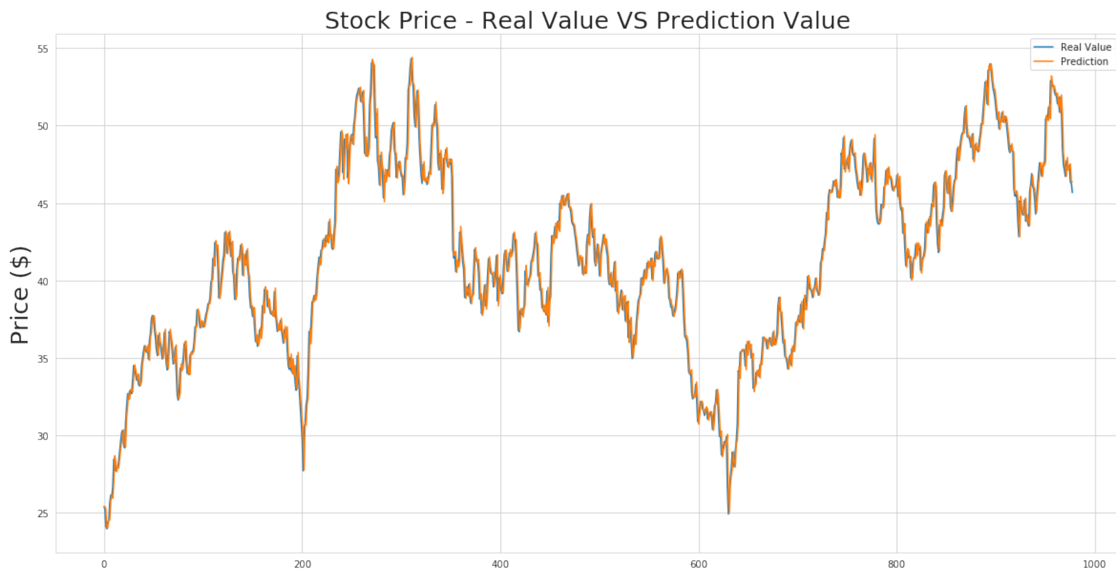
```
The next value is: 46.44
```

5 Step 4: Model Exploration / Explanation

Explain the model's predictions to someone who is unfamiliar with machine learning. Use charts/tables/etc to explain why they should (or perhaps should not?) rely on these predictions.

[23]:
```
plt.figure(figsize=(20,10))
sns.set_style("whitegrid")
ax = sns.lineplot(data = df2_predictions["Real_Value"], label = "Real Value")
sns.lineplot(data = df2_predictions["Prediction"], label = "Prediction")
ax.set_title("Stock Price - Real Value VS Prediction Value", fontsize=25)
ax.set_ylabel("Price ($)", fontsize=25)
```

[23]:
```
Text(0, 0.5, 'Price ($)')
```



This is the comparison of the real values vs the predicted values for the American Airlines stock "Close" price