Kanwarpartap Singh Brar

Impact of EFFR Changes on Sectoral Stock Market Performance

Part 1 (Question Formulation)/Introduction:

The question I will be diving into is:

"How do changes in the Effective Federal Funds Rate (EFFR) affect the stock market performance of various economic sectors?"

The Effective Federal Funds Rate (EFFR) is the interest rate that banks pay to borrow funds from each other overnight. By looking into this question, I aim to analyze how changes in the EFFR impact different economic sectors such as the technology, healthcare, financial, and consumer discretionary sectors. I will be analyzing data over a period of approximately 4 years (Jan 2020 - August 2024).

I choose to use the EFFR as the chosest rate to look at because it has one der influences on financial markets. The EFFR has direct influences on other interest rates such as mortgage rates, auto loan rates, borrowing rates, etc, and this is important because it influences the interest rate banks charge their customers. In turn, the rates banks charge their customers have a broader impact on economic activity and financial markets because they directly affects liquidity and expectations of economic growth or a recession.

The different sectors I will be looking at are the tech, healthcare, financial, and consumer discretionary sectors (this is the sector that deals with retail/luxury goods people buy when they have more disposable income). I get my data from the yahoo finance sector ETFs which are funds that represent different sectors. I will go into more detail about where I get my data from in the 'Data Attribution & Reference' section of this report. I will only be looking at the ***'Close'*** price (in $USD) of each sector at the first of every month from Jan 2020 to August 2024. ***Close*** will be the main metric to evaluate stock performance because it is a convention that is commonly used in financial analysis to track changes in stock price. I can go into further detail as to the specifics of why it is one of the better

metrics to track stock performance in future works. In this report, we will only be looking at the closing price of each ETF to analyze how each sector performs over the 4 years.

4 ETFs:

1. XLK: Tech Sector
2. XLV: Healthcare Sector
3. XLF: Financial Sector
4. XLY: Consumer Discretionary Sector

In conclusion, I will be gathering data from the 4 ETFs from Yahoo Finance and data on the EFFR from The FED. Then I will merge these datasets into one on which I will perform my analysis for this project.

<u>Part 2 (Data Collection & Pre-Processing):</u>

I collected data from two sources, Yahoo Finance & FRED (Federal Reserve Economic Data). I wasn't able to download the ETF data directly from Yahoo Finance without paying, so I installed the yfinance python library and obtained the data that way. Below is my code for the datasets, I will also attach the datasets as a CSV file with my final submission of this project.

```
import yfinance as yf
#XLK data (tech sector)

XLK_Tech = yf.download('XLK', start = "2020-01-01", end = "2024-09-01")
XLK_monthly = XLK_Tech.resample('MS').first() #MS is Month Start so getting data from first of the month

XLK_monthly.to_csv("XLK_Tech_Data.csv") #Download tech sector data

#XLV data (Health sector)

XLV_Healthcare = yf.download('XLV', start = "2020-01-01", end = "2024-09-01" )
XLV_monthly = XLV_Healthcare.resample('MS').first()

XLV_monthly.to_csv("XLV_Healthcare_Data.csv") #Download healthcare sector data

#XLF data (Financial sector)

XLF_Finance = yf.download('XLF',start = "2020-01-01", end = "2024-09-01" )
XLF_monthly = XLF_Finance.resample('MS').first()

XLF_monthly.to_csv("XLF_Finance_Data.csv") #Download financial sector data

#XLY data (Consumer discretionary data)
#consumer industreis like retail, media, etc

XLY_Consumer = yf.download('XLY', start= "2020-01-01", end = "2024-09-01" )
XLY_monthly = XLY_Consumer.resample('MS').first()

XLY_monthly.to_csv("XLY_Consumer_Data.csv") #Download consumer spending data
```

Below is my code for organizing and cleaning the data so that it is organized for analysis:

```
@author: shawnbrar
"""

import pandas as pd

#Read 5 csv files

XLK_Tech_Data = pd.read_csv('/Users/shawnbrar/Desktop/DS_221/XLK_Tech_Data.csv', skiprows = [1,2])
#Have to skip importing rows 1 and 2 b/c they contain unnecessary data that arent part of the actual data
#print(XLK_Tech_Data.head())

XLV_Healthcare_Data = pd.read_csv('/Users/shawnbrar/Desktop/DS_221/XLV_Healthcare_Data.csv', skiprows = [1,2])
#print(XLV_Healthcare_Data.head())

XLF_Finance_Data = pd.read_csv('/Users/shawnbrar/Desktop/DS_221/XLF_Finance_Data.csv', skiprows = [1,2])
#print(XLF_Finance_Data.head())

XLY_Consumer_Data = pd.read_csv('/Users/shawnbrar/Desktop/DS_221/XLY_Consumer_Data.csv', skiprows = [1,2])
#print(XLY_Consumer_Data.head())

FFER_Data = pd.read_csv('/Users/shawnbrar/Desktop/DS_221/FFER_Dataset.csv')
#print(FFER_Data.head())
FFER_Data.rename(columns = {'FEDFUNDS':'EFFR','DATE':'Date'}, inplace = True) #Rename column Date so all datasets can be merged on one columr
FFER_Data['Date'] = pd.to_datetime(FFER_Data['Date']).dt.date #make sure all datasets have same date format

###ORGANIZING/CLEANING STOCK DATA

##Organize the XLK Data

XLK_Tech_Data.rename(columns = {'Price':'Date'}, inplace = True)
#rename Price column to Date b/c data was imported incorrectly
#use inplace = True so I dont have to make another variable of the data


#Fix the date so it alligns with the FFER_Data and standardized as a simple data format YY-MM-DD
XLK_Tech_Data['Date'] = pd.to_datetime(XLK_Tech_Data['Date']).dt.date


#Drop all unnecesary columns, we only want to look at the stock Close column to analyze stock performance over time
drop_unnecesary_columns = ['Adj Close', 'High', 'Low', 'Open', 'Volume']
XLK_Tech_Data = XLK_Tech_Data.drop(columns = drop_unnecesary_columns)

#Rename Close column to Close(Tech) to make it easier to identify which industry
XLK_Tech_Data.rename(columns = {'Close':'Close(Tech)'}, inplace = True)
#print(XLK_Tech_Data.head())


#Now I repeat the same process for the XLY, XLF, and XLV stock data
```

```
#Now I repeat the same process for the XLY, XLF, and XLV stock data

XLV_Healthcare_Data.rename(columns = {'Price':'Date'}, inplace = True)
XLV_Healthcare_Data['Date'] = pd.to_datetime(XLV_Healthcare_Data['Date']).dt.date
XLV_Healthcare_Data = XLV_Healthcare_Data.drop(columns = drop_unnecesary_columns)
XLV_Healthcare_Data.rename(columns = {'Close':'Close(Healthcare)'}, inplace = True)
#print(XLV_Healthcare_Data.head())

XLF_Finance_Data.rename(columns = {'Price':'Date'}, inplace = True)
XLF_Finance_Data['Date'] = pd.to_datetime(XLF_Finance_Data['Date']).dt.date
XLF_Finance_Data = XLF_Finance_Data.drop(columns = drop_unnecesary_columns)
XLF_Finance_Data.rename(columns = {'Close':'Close(Finance)'}, inplace = True)
#print(XLF_Finance_Data.head())

XLY_Consumer_Data.rename(columns = {'Price':'Date'}, inplace = True)
XLY_Consumer_Data['Date'] = pd.to_datetime(XLY_Consumer_Data['Date']).dt.date
XLY_Consumer_Data = XLY_Consumer_Data.drop(columns = drop_unnecesary_columns)
XLY_Consumer_Data.rename(columns = {'Close':'Close(Consumer)'}, inplace = True)
#print(XLY_Consumer_Data.head())

###MERGE all the datasets into one dataset (merge on 'Date')

merged_datasets = pd.merge(FFER_Data, XLK_Tech_Data, on='Date')
merged_datasets = pd.merge(merged_datasets, XLV_Healthcare_Data, on = 'Date')
merged_datasets = pd.merge(merged_datasets, XLF_Finance_Data, on = 'Date')
merged_datasets = pd.merge(merged_datasets, XLY_Consumer_Data, on ='Date')
print(merged_datasets.head())

#saving my dataset as a file
#Merged_Stock_EFFR_Data = merged_datasets
#Merged_Stock_EFFR_Data.to_csv('/Users/shawnbrar/Desktop/DS_221/Merged_Stock_EFFR_Data.csv')
```

This code reads all the datasets and cleans them by organizing the columns/headers, renaming/deleting columns when necessary, as well as making sure the ‘Date’ columns are all in the same timestamp format (YY-MM-DD) so they can be merged together. In the end, I have merged all the data sets into one which I will also attach with the final submission of this project.

The following code checks to see if the data is clean:

```
#Check to see if data is clean


print("\n")

print("Cheking to see if the data contains any null values: ")
print(merged_datasets.isnull().any())

print("\n")
print('Shape of data: ')
print(merged_datasets.shape)

print("\n")
print('Data Types: ')
merged_datasets['Date'] = pd.to_datetime(merged_datasets['Date'], errors='coerce')
#python was having trouble with the '-' in between the dates so by using coerce pandas will ignore them and convert the dtype to datetime instea
print(merged_datasets.dtypes)
```

```
In [66]: runcell(0, '/Users/shawnbrar/Desktop/DS_221/DS_Midterm_Project.py')
         Date  EFFR  ...  Close(Finance)  Close(Consumer)
0  2020-01-01  1.55  ...       31.080000       126.910004
1  2020-02-01  1.58  ...       30.170000       126.209999
2  2020-03-01  0.65  ...       27.950001       119.809998
3  2020-04-01  0.05  ...       19.549999        93.699997
4  2020-05-01  0.05  ...       22.059999       112.239998

[5 rows x 6 columns]


Cheking to see if the data contains any null values:
Date                 False
EFFR                 False
Close(Tech)          False
Close(Healthcare)    False
Close(Finance)       False
Close(Consumer)      False
dtype: bool


Shape of data:
(56, 6)


Data Types:
Date                 datetime64[ns]
EFFR                        float64
Close(Tech)                 float64
Close(Healthcare)           float64
Close(Finance)              float64
Close(Consumer)             float64
dtype: object
```

Here we can see that the dataset does not contain any null values and has 56 records from Jan 2020 Aug 2024. It also has 6 columns, Date, EFFR, Close (Finance), Close (Healthcare), Close (Technology), and Close (Consumer). All of

the ETF stock columns and the EFFR are float64 data types meaning they have decimals. The Date column is the datetime64 data type, it was an object because the time was initially stored in a string, but I was able to convert it into datetime64 so I don't run into any possible problems when doing analysis.

Part 3 (Exploratory Data Analysis):

Summary Statistics:

```
###Part 3 (EDA)
## 1. Summary Statisitcs


summary_statistics = merged_datasets.describe()
#gets mean, median, sd, min, max, quartiles
numeric_columns = merged_datasets.select_dtypes(include = 'float64') #selecting all columns except 'Date'


var = numeric_columns.var()  #getting variance of columns
summary_statistics.loc['var'] = var
summary_statistics = summary_statistics.drop(columns = 'Date') #drop 'Date' as it is unnecessary

print("\n")
print("Summmary Statistics: ")
print(summary_statistics)
```

```
              FFER  Close(Tech)  Close(Healthcare)  Close(Finance)
count  56.000000    56.000000          56.000000       56.000000
mean    2.296429   147.794644         125.314465       33.746964
min     0.050000    76.540001          85.209999       19.549999
25%     0.087500   126.577503         115.122501       31.155000
50%     1.380000   143.740005         129.489998       34.495001
75%     5.090000   166.602501         134.047504       37.857499
max     5.330000   227.940002         151.160004       43.119999
std     2.332670    34.350815          14.957157        5.582706
var     5.441351  1179.978510         223.716532       31.166603

       Close(Consumer)
count         56.000000
mean         160.477857
min           93.699997
25%          145.609997
50%          164.284996
75%          177.379997
max          210.309998
std           23.563091
var          555.219269
```

By looking at our summary statistics we can see quite a few things and learn about the data. Firstly, over the 56 months the average EFFR was approximately 2.3%, and the average closing price (which indicates stock performance) over the period was approximately $147.79, $125.31, $33.75, and $160.48 for the tech, healthcare, finance, and consumer sectors respectively. The lowest EFFR during this period was .05% and the highest was 5.33%. We can also see the different quartiles and see how each sector performed during that period. For example, the third quartile tells us that 75% of the time the EFFR was below 5.09%. The standard deviations tell us how volatile each sector is, the higher the std the more volatile the sector because that would mean it has a higher +- and fluctuation around the mean. We see that the tech sector has the largest std at approximately +-34.35 around the mean, telling us it is the most volatile sector of the 4. On the other hand, the financial sector has the smallest std at approximately +- 5.58, making it the least volatile economic sector of the 4.

Data Visualization/Data Distribution/Correlation Analysis:

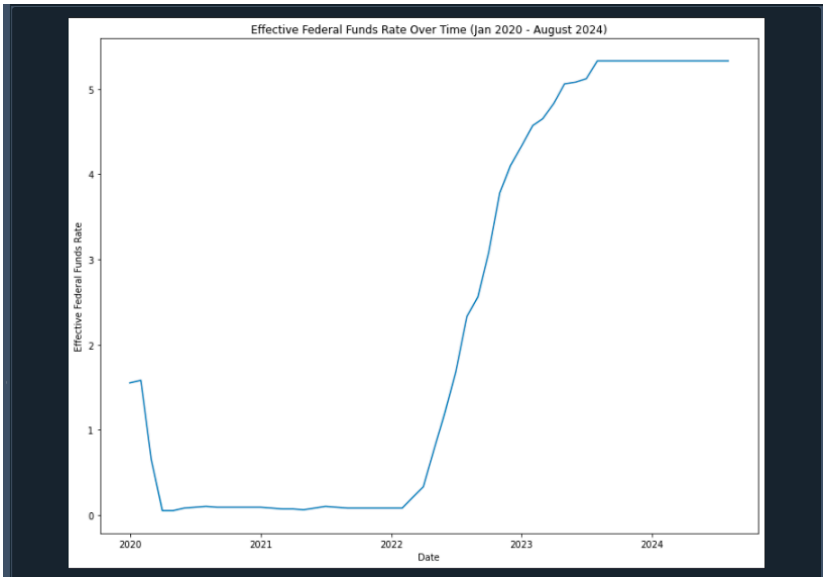
Below is the code for the data visualization and the correlation analysis:

```
## 2. Data Visualization/Correlation Analysis

#Line graph of FFER over time (see how the interst rate changed over 56 months)

x = merged_datasets.Date
y = merged_datasets.EFFR
plt.figure(figsize = (13,10))
plt.title('Effective Federal Funds Rate Over Time (Jan 2020 - August 2024)')
plt.xlabel('Date')
plt.ylabel('Effective Federal Funds Rate')
plt.plot(x,y)
plt.show()

#Scatterplot relationship between EFFR & Tech Sector

effr_x = merged_datasets['EFFR']
y_tech = merged_datasets['Close(Tech)']
plt.figure(figsize = (13,10))
plt.xlabel("Effective Federal Funds Rate", fontsize = 17)
plt.ylabel("Closing Price (Technology Sector)", fontsize = 17)
plt.title('Relationship Between EFFR and Technology Sector Closing Prices', fontsize = 20, pad = 5, )
plt.scatter(effr_x,y_tech, color = 'royalblue', s = 120)
plt.grid(linewidth = 0.5)
plt.show()

effr_tech_correlation = merged_datasets[['EFFR','Close(Tech)']].corr()
print("\n")
print('Correlation beween EFFR & Close(Tech): ')
print(effr_tech_correlation)

#Scatterplot relationship between EFFR & Health sector


y_healthcare = merged_datasets['Close(Healthcare)']
plt.figure(figsize = (13,10))
plt.xlabel("Effective Federal Funds Rate", fontsize = 17)
plt.ylabel("Closing Price (Healthcare Sector)", fontsize = 17)
plt.title('Relationship Between EFFR and Healthcare Sector Closing Prices', fontsize = 20, pad = 5, )
plt.scatter(effr_x,y_healthcare, color = 'royalblue', s = 120)
plt.grid(linewidth = 0.5)
plt.show()

effr_health_correlation = merged_datasets[['EFFR','Close(Healthcare)']].corr()
print("\n")
print('Correlation beween EFFR & Close(Healthcare): ')
print(effr_health_correlation)
```

```
#Scatterplot relationship between EFFR & Finance sector
y_finance = merged_datasets['Close(Finance)']
plt.figure(figsize = (13,10))
plt.xlabel("Effective Federal Funds Rate", fontsize = 17)
plt.ylabel("Closing Price (Finance Sector)", fontsize = 17)
plt.title('Relationship Between EFFR and Finance Sector Closing Prices', fontsize = 20, pad = 5, )
plt.scatter(effr_x,y_finance, color = 'royalblue', s = 120)
plt.grid(linewidth = 0.5)
plt.show()

effr_finance_correlation = merged_datasets[['EFFR','Close(Finance)']].corr()
print("\n")
print('Correlation beween EFFR & Close(Finance): ')
print(effr_finance_correlation)


#Scatterplot relationship between EFFR & consumer discretionary

y_consumer = merged_datasets['Close(Consumer)']
plt.figure(figsize = (13,10))
plt.xlabel("Effective Federal Funds Rate", fontsize = 17)
plt.ylabel("Closing Price (Consumer Discretionary Sector)", fontsize = 17)
plt.title('Relationship Between EFFR and Consumer Sector Closing Prices', fontsize = 20, pad = 5, )
plt.scatter(effr_x,y_consumer, color = 'royalblue', s = 120)
plt.grid(linewidth = 0.5)
plt.show()

effr_consumer_correlation = merged_datasets[['EFFR','Close(Consumer)']].corr()
print("\n")
print('Correlation beween EFFR & Close(Consumer): ')
print(effr_consumer_correlation)

##Box plot distribution over sectors
plt.boxplot([merged_datasets['Close(Tech)'],merged_datasets['Close(Healthcare)'],merged_datasets['Close(Finance)'],merged_datasets['Close(Consumer)']],
            labels = ['Tech','Healthcare', 'Finance', 'Consumer'])
plt.title('Distributions Of Closing Prices By Sector')
plt.ylabel('Closing Price ($ USD)')
plt.show()
```

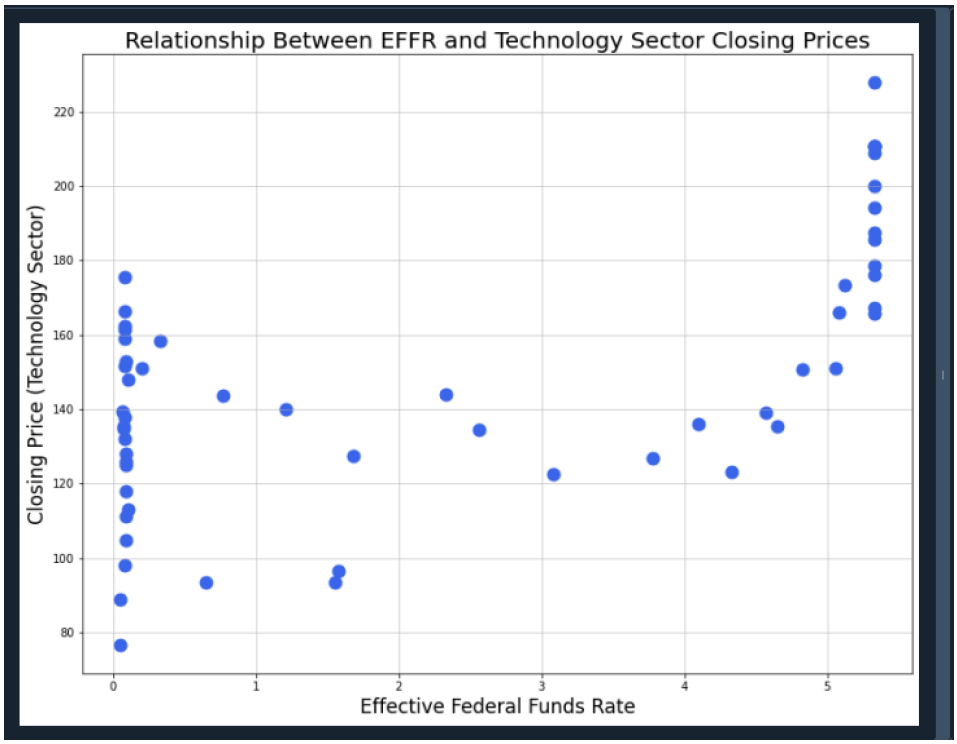
In this section, I will walk through 6 different graphs that help visualize and understand the data. The 5 graphs will be a line graph of EFFR over time, the scatterplot relationship between EFFR & Tech Sector, the scatterplot relationship between EFFR & Health sector, the scatterplot relationship between EFFR & Finance sector, the scatterplot relationship between EFFR & the consumer discretionary, and a box plot that shows a distribution over the sectors.

1. *Line graph of EFFR over time (see how the interest rate changed over 56 months):*



Here we see that the EFFR has a sharp decline in early 2020 and the interest rates go close to 0%. This was during the start of the Covid-19 lockdowns, where the rates had to be dropped so the economy could be stimulated. Then once we started coming out of the pandemic around 2022, interest rates were raised to help combat inflation and an active economy. Then around 2023 and the current day, we are in a period where the EFFR is stabilized and The FED is assessing their next steps on what to do based on different macroeconomic factors.

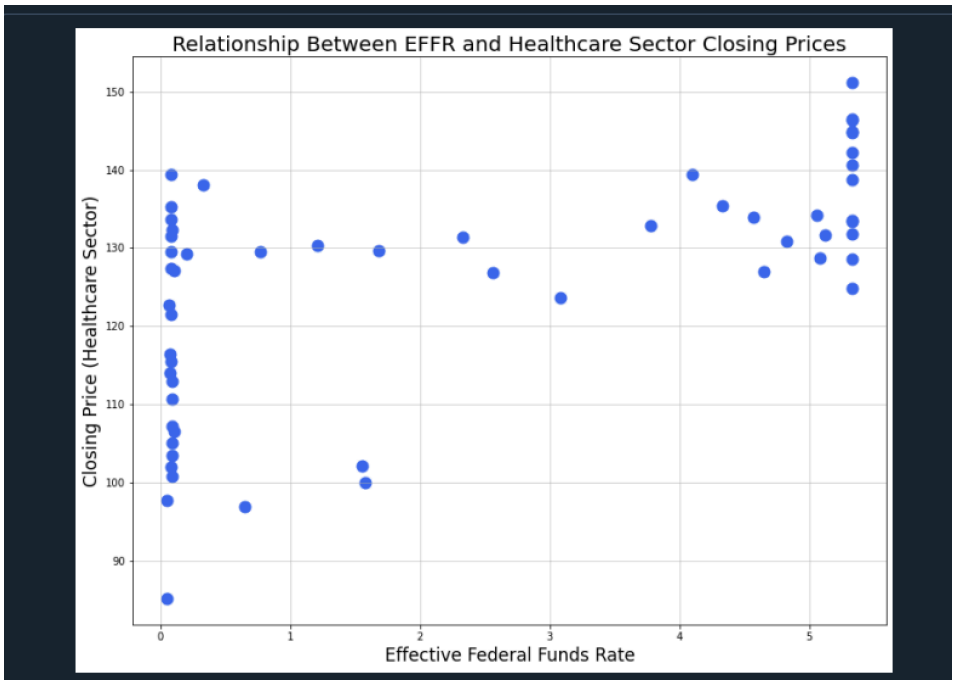
2. *Scatterplot relationship between EFFR & the Tech Sector:*




```
Correlation beween EFFR & Close(Tech):
                EFFR  Close(Tech)
EFFR         1.00000      0.62311
Close(Tech)  0.62311      1.00000
```

Here we see a moderate positive correlation between EFFR and Close(Tech) at approximately r = .62311. Meaning, that as the EFFR rises, the stock performance of the tech sector generally tends to rise as well. This could be due to the tech sector started booming during economic growth, which is usually when interest rates tend to rise.

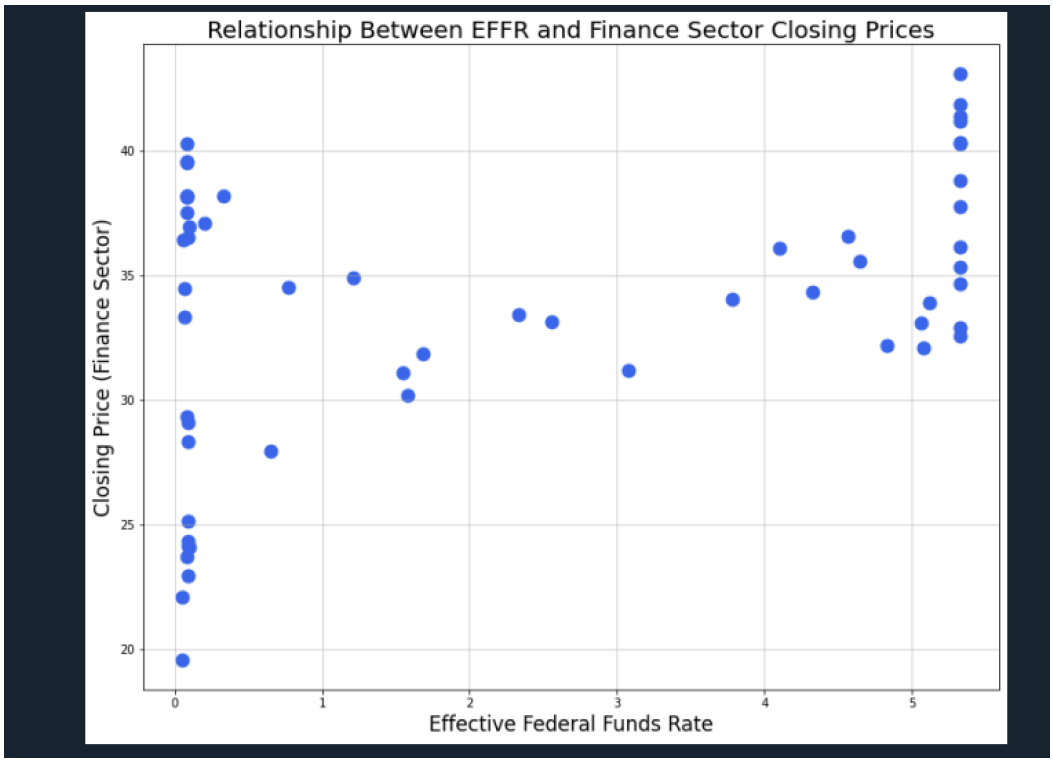
3. *Scatterplot relationship between EFFR & Close(Healthcare):*




```
Correlation beween EFFR & Close(Healthcare):
                        EFFR  Close(Healthcare)
EFFR                1.000000           0.611765
Close(Healthcare)   0.611765           1.000000
```

Here we also see a moderate positive correlation between EFFR & Close(Healthcare) at approximately r = .6118. Similar to the tech sector, it seems the healthcare sector has performed somewhat well with higher interest rates. However, because the correlation isn't strong, it tells us that there are also other factors besides the EFFR that influence the performance of the healthcare sector.

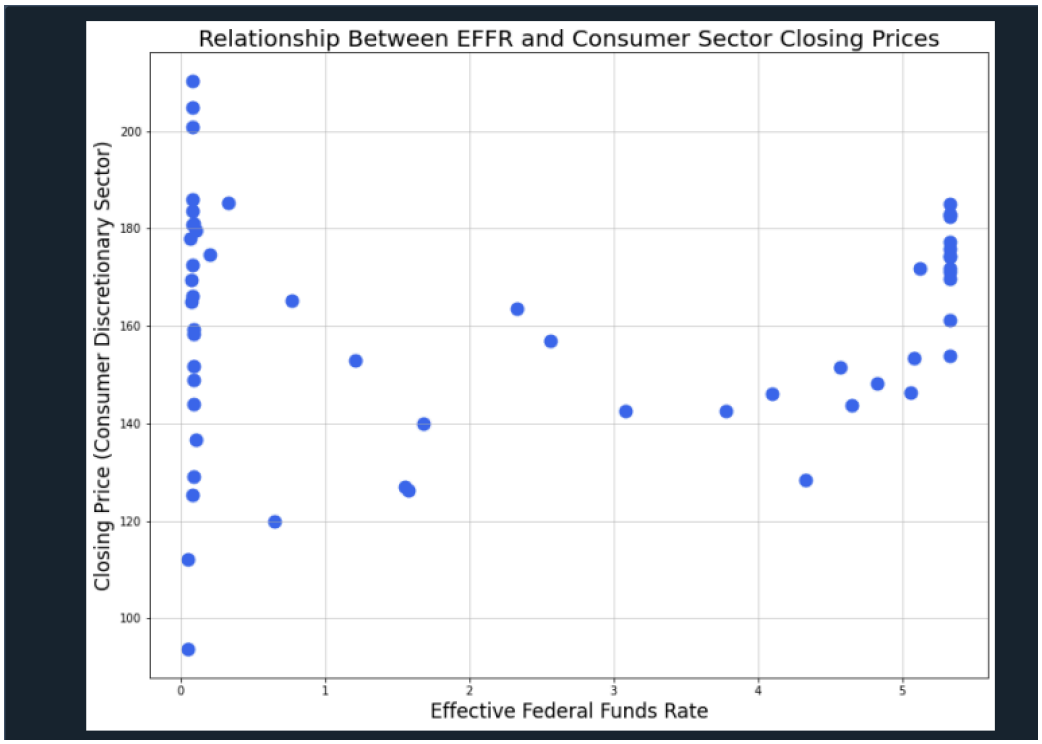
4. *Scatterplot relationship between EFFR & Close(Finance):*




```
Correlation beween EFFR & Close(Finance):
                    EFFR  Close(Finance)
EFFR            1.000000        0.415164
Close(Finance)  0.415164        1.000000
```

Here we see that the two variables have a weaker positive correlation as the correlation is approximately r = .4152. This tells us that there is a positive relationship, however, EFFR does not have a major factor in the performance of the financial sector and is not as consistent/strong as it was with the tech and healthcare sectors.

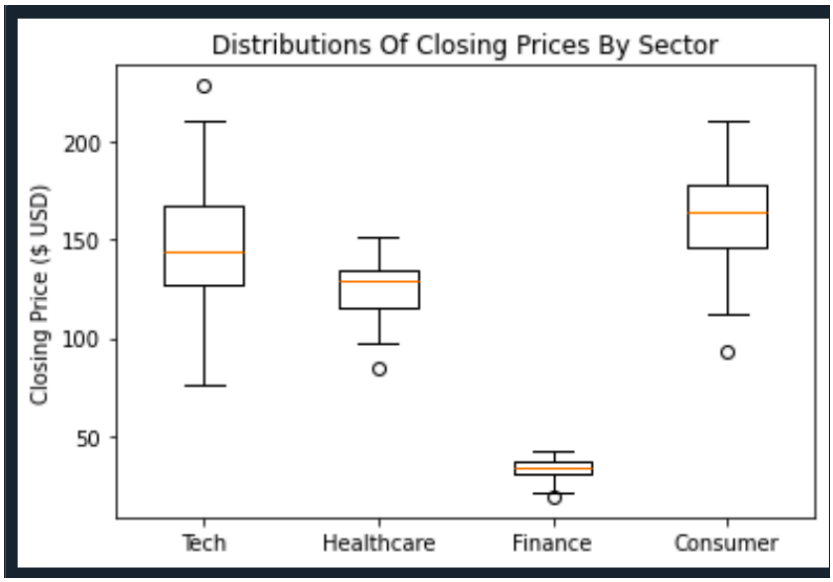
5. *Scatterplot relationship between EFFR & Close(Consumer):*




```
Correlation beween EFFR & Close(Consumer):
                     EFFR  Close(Consumer)
EFFR             1.000000         0.078275
Close(Consumer)  0.078275         1.000000
```

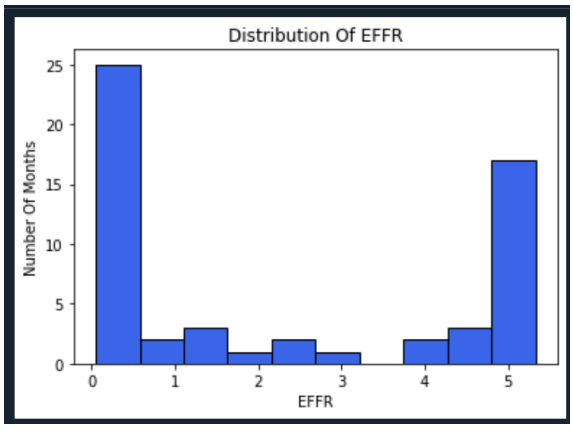
Here we see that the two variables have a very weak positive correlation as r is approximately = .0783. This tells us that the consumer sector closing prices are spread out all over the place with no predictable pattern. This tells us that the EFFR does not seem to be a primary factor that affects the performance of the consumer market. However, we do see that when the EFFR was close to 0, and money was cheaper to borrow, that was when the consumer sector peaked at its highest which is something of note.

6. *Box plot distribution over sectors:*



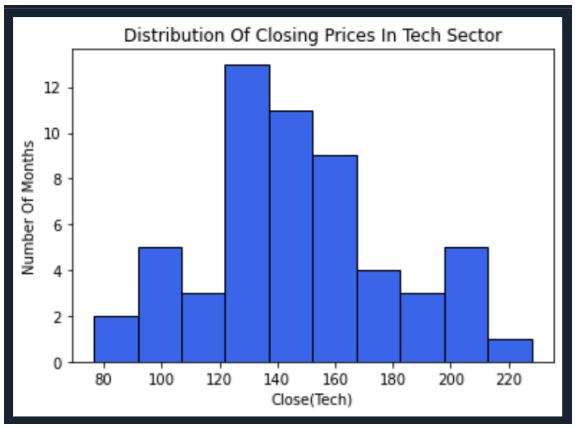
With the box plot, we can see the distribution of the data for each sector. In the tech sector we see that it has the highest range out of all the other sectors, which could indicate that it is a more volatile sector and has more variability in its performance. In the healthcare sector we see that the range is quite small and it seems to be fairly stable. Similar to the healthcare sector, the finance sector has very low variability, the least out of all 4 sectors, suggesting that it is a more stable sector when it comes to its performance. The consumer sector similar to tech is wider than the other two sectors, suggesting higher volatility in its performance.

7. *Histogram for EFFR:*



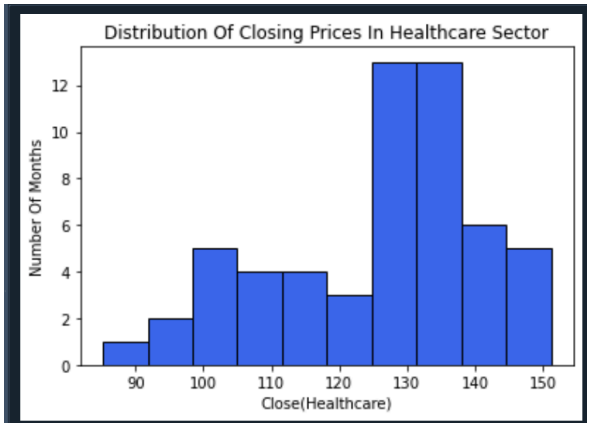
Here we can see the distribution of the EFFR. The distribution can be looked at as skewed to the right, as the majority of the data points are on the left, indicating that over this time period interest rates were closer to 0%. The split between low and high indicates that over this period, the EFFR changed rapidly and did not remain in the intermediate stage (1% - 4%) for long.

8. *Histogram for Close(Tech):*



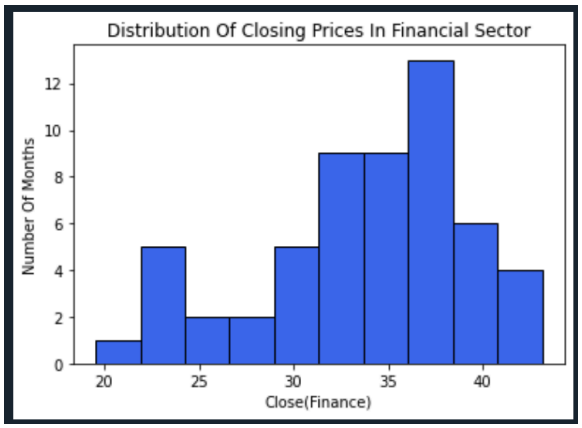
Here we can see the distribution of the closing prices in the tech sector. The distribution seems to be slightly skewed to the right, very close to being symmetric.

9. *Histogram for Close(Healthcare):*



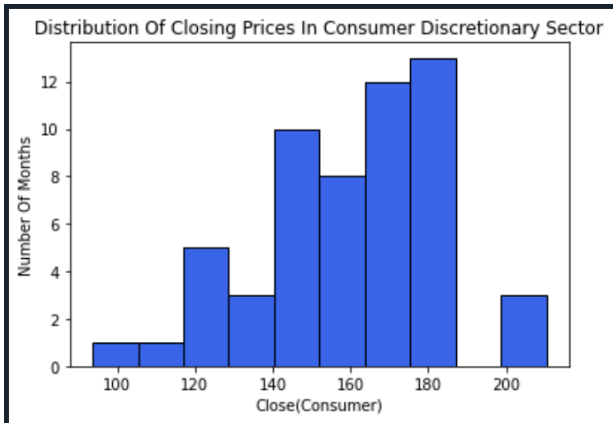
We can see that the distribution of the closing prices in the healthcare sector seems to be skewed to the left, indicating that it remained performing at a higher level according to having a higher price for longer.

10. Histogram for Close(Finance):



We can see that the distribution of the closing prices in the financial sector seems to be slightly skewed to the left.

11. Histogram for Close(Consumer):



Similar to the finance and healthcare sector, we see that the distribution of the closing pieces of the consumer sector is also skewed to the left. It would also be nice to note that the performance of the consumer sector was highest in 22 months, which is about how long the pandemic and lockdowns lasted.

Hypothesis Generation & Hypothesis Testing:

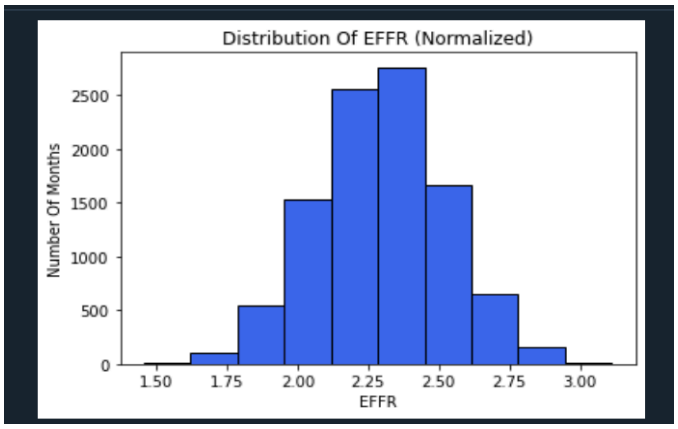
Hypothesis: Does a change in the EFFR impact consumer spending on luxury goods and services (the consumer discretionary sector)?

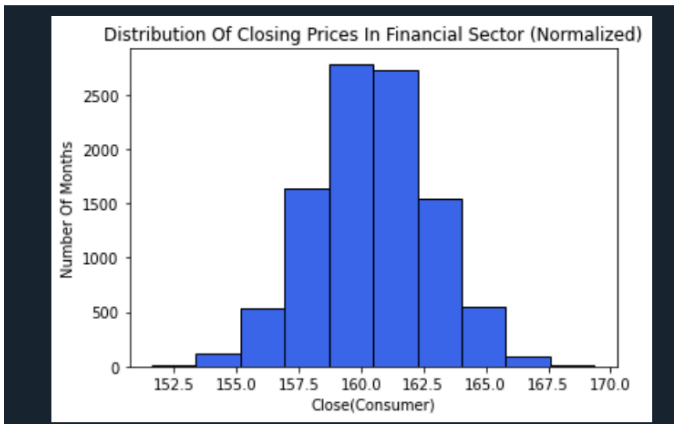
H0 (Null): Changes in the EFFR have no significant impact on consumer spending on luxury goods and services (consumer discretionary sector).

H1 (Alternative): Changes in the EFFR significantly impact consumer spending on luxury goods and services (consumer discretionary sector).

Code for hypothesis testing:

```
#Hypothesis Testing

#Take mean of 100 random samples from EFFR 10000 times and then append to EFFR_mean_size_100
#This allows for an approximation of a normal dist as the sample size gets larger

EFFR_mean_size_100 = []
for i in range (10000):
    sample_effr = np.random.choice(merged_datasets['EFFR'],100).mean()
    EFFR_mean_size_100.append(sample_effr)

#Now doing the same thing for Close(Consumer) to take advantage of the Central Limit Theorem

Consumer_mean_size_100 = []
for i in range(10000):
    sample_consumer = np.random.choice(merged_datasets['Close(Consumer)'],100).mean()
    Consumer_mean_size_100.append(sample_consumer)


#Histogram for EFFR after taking multiple random samples and averaging them to use the CLT
plt.hist(EFFR_mean_size_100, edgecolor = 'black', color = 'royalblue')
plt.xlabel('EFFR')
plt.ylabel('Number Of Months')
plt.title('Distribution Of EFFR (Normalized)')
plt.show()

#Histogram for Close(Consumer) after taking multiple random samples and averaging them to use the CLT

plt.hist(Consumer_mean_size_100, edgecolor = 'black', color = 'royalblue')
plt.xlabel('Close(Consumer)')
plt.ylabel('Number Of Months')
plt.title('Distribution Of Closing Prices In Financial Sector (Normalized)')
plt.show()

#Now performing independent ttest

#Have to split the EFFR into two groups: High Effr group & Low EFFR group
#spltiing at the mean

average_EFFR = merged_datasets['EFFR'].mean()
above_average_EFFR = merged_datasets[merged_datasets['EFFR'] > average_EFFR]['Close(Consumer)']
below_average_EFFR = merged_datasets[merged_datasets['EFFR'] <= average_EFFR]['Close(Consumer)']
#splitting dataset into two different groups
#gives us the closing price of the consumer discretionary stock when Effer is above/below average

#ttest

statistic, p_value = stats.ttest_ind(a = above_average_EFFR, b = below_average_EFFR, equal_var = False)
print("\n")
print('HYPOTHESIS TESTING')
print("t_stat: ", statistic, "p_value: ", p_value)

alpha = .05
if pvalue < .05:
    print("Failed to reject the null hypothesis")
else:
    print("Reject the null hypothesis, the result is statistically significant at the 5% significance level")
```






```
HYPOTHESIS TESTING
t_stat:  0.5546743958057629 p_value:  0.5816605510501098
Failed to reject the null hypothesis
```

Before I started the hypotheses test, I made sure to approximate a normal distribution by taking multiple random samples and averaging, this allowed me to leverage the Central Limit Theorem (CLT).

The test I chose is an independent ttest. I split the EFFR into two groups, one above the mean and one below the mean. This way I was able to test the difference in the closing price of the consumer stock over the two groups.

After running the ttest, the p-value was very high at approximately 0.582 and I failed to reject the null hypothesis. There is not enough evidence to conclude that a change in the EFFR impacts consumer spending on luxury goods and services (the consumer discretionary sector).

Part 4 (Machine Learning):

The model I have decided to use is a decision tree regressor model using the criterion *squared_error*. I am choosing to use this model because it will be easier to run my continuous variables with this model, as well as because it will have an easier time capturing the non-linear relationships of the variables. The purpose of these models is to determine the relationship between EFFR (economic indicator of interest rates), and the performance of specific economic sectors such as the tech and healthcare sectors (by looking at the close prices of their stocks). Below is my code and an explanation of its mean squared error and R2.

```
### Part 4 Machine Learning

#Decision Tree Regressor Model
#Predicting performance of the tech sector by predicting the closing stock price

#features and target variable
x = merged_datasets.iloc[:,[1]].values
y = merged_datasets.iloc[:,[2]].values

#split data so 20% is for testing and 80% is for training

x_train, x_test, y_train, y_test = train_test_split(x,y,test_size = 0.2, random_state = 42)

decision_tree_squared_error_model = DecisionTreeRegressor(criterion = 'squared_error', random_state = 42) #regression dt model uses squared_error as the criterion
decision_tree_squared_error_model.fit(x_train,y_train)
y_pred_decision_tree_squared_error_model = decision_tree_squared_error_model.predict(x_test)

decision_tree_squared_error_mse = mean_squared_error(y_test,y_pred_decision_tree_squared_error_model) #get mse


decision_tree_squared_error_R2_score = r2_score(y_test, y_pred_decision_tree_squared_error_model) #get R^2 of model


print('\n')
print("Predicted Array Of Tech Sector Performance: ", y_pred_decision_tree_squared_error_model)
print('\n')
print("Mean Squared Error Of Decision Tree Regression Model: ",decision_tree_squared_error_mse )
print("R2 Score of Decision Tree Model Using Squared Error: ", decision_tree_squared_error_R2_score)

#plotting decision tree

plot_tree(decision_tree_squared_error_model, feature_names=['EFFR'], filled = True)
plt.title("Decision Tree for Tech Sector Closing Price Prediction")
plt.figure(figsize = (25,20), dpi = 100)
plt.savefig('tech_sector_decision_tree.png',dpi = 100)
plt.show()
plt.close
```

```
#Predicting performance of the healthcare sector by predicting the closing stock price

x = merged_datasets.iloc[:,[1]].values
y_health = merged_datasets.iloc[:,[3]].values


#split data so 20% is for testing and 80% is for training

x_train, x_test, y_health_train, y_health_test = train_test_split(x,y_health,test_size = 0.2, random_state = 42)

decision_tree_squared_error_model = DecisionTreeRegressor(criterion = 'squared_error', random_state = 42) #regression dt model uses squared_error as the criterion
decision_tree_squared_error_model.fit(x_train,y_health_train)
y_pred_decision_tree_squared_error_model = decision_tree_squared_error_model.predict(x_test)

decision_tree_squared_error_mse = mean_squared_error(y_health_test,y_pred_decision_tree_squared_error_model) #get mse


decision_tree_squared_error_R2_score = r2_score(y_health_test, y_pred_decision_tree_squared_error_model) #get R^2 of model


print('\n')
print("Predicted Array Of Healthcare Sector Performance: ", y_pred_decision_tree_squared_error_model)
print('\n')
print("Mean Squared Error Of Decision Tree Regression Model: ",decision_tree_squared_error_mse )
print("R2 Score of Decision Tree Model Using Squared Error: ", decision_tree_squared_error_R2_score)


#plotting decision tree

plot_tree(decision_tree_squared_error_model, feature_names=['EFFR'], filled = True)
plt.title("Decision Tree for Healthcare Sector Closing Price Prediction")
plt.figure(figsize = (25,20), dpi = 100)
plt.savefig('healthcare_sector_decision_tree.png')
plt.show()
plt.close()
```

```
Predicted Array Of Tech Sector Performance:  [ 96.58999634 159.11428615 134.6000061
159.11428615 116.94000092
 190.91499939 136.1499939  130.53000259 190.91499939 116.94000092
 190.91499939  88.90000153]


Mean Squared Error Of Decision Tree Regression Model:  729.5696140790016
R2 Score of Decision Tree Model Using Squared Error:  0.6208129171584325
<Figure size 2500x2000 with 0 Axes>


Predicted Array Of Healthcare Sector Performance:  [100.04000092 131.19000135
126.80999756 131.19000135 105.48000031
 138.25400085 139.41999817 116.84500122 138.25400085 105.48000031
 138.25400085  97.83000183]


Mean Squared Error Of Decision Tree Regression Model:  195.96902800886383
R2 Score of Decision Tree Model Using Squared Error:  0.3920546653787361
<Figure size 2500x2000 with 0 Axes>
```

Results & Discussion:

When looking at the tech sector we can see that the mean squared error (mse) is approximately 729.59. The higher the mse is the more discrepancy there is in the data usually because it is a measure of the squared difference between the predicted values and the actual values. 729.59 suggests that there is some level of error in the model's predictions for the Tech(Close) variable. However in this context of the data, because the target variable varies so much (the tech sector is volatile), a mse of 729 is a reasonable amount of error. The R^2 is approximately 0.621, meaning that about 62% of the variance in the target variable (Close(Tech)) is explained by the model. This is a moderately good R^2 suggesting that most of the way the tech sector performs is captured by the relationship between the two variables, however the remaining 38% of variability is due to other economic factors.

Looking at the healthcare sector we see that the mse is approximately 195.97. This is a lower mse, indicating that the model's prediction for the healthcare sector closing prices is closer to the actual values. The R^2 is approximately 39%, indicating that 39% of the variance in the target variable (Close(Healthcare)) is explained by the model, the rest suggest that other factors might have a more significant impact on healthcare performance.

In conclusion, the differences in R^2 for both variables suggest that changes in EFFR explain more of the variability in the tech sector than in healthcare, implying that the tech sector is more sensitive to interest rate changes than the healthcare sector. I have attached pics of the decision tree for both models separately.

Conclusion:

Based on the results, we can conclude that changes in the Effective Federal Funds Rate (EFFR) have varying impacts on different economic sectors (tech, healthcare, finance, and consumer discretionary).

The hypothesis testing did not show any significant relationship between changes in the EFFR and consumer spending on luxury goods and services. This tells us that within the time (Jan 2020 - Aug 2024), EFFR fluctuations did not have a significant impact on consumer discretionary spending. My theory is that this is

due to Covid-19 and because of the lockdown and people losing their jobs, most of the spending went towards essential goods instead of luxury goods. This is a concept I will be willing to explore further in future works.

When it comes to the performance of the machine learning model of decision tree regressor, it was implied that the tech sector is more reactive to interest rate changes than the healthcare sector. This makes sense because the tech sector is influenced by capitalism and is more influenced by monetary policy shifts and its growth depends more on the ability to borrow money when compared to the healthcare sector.

In conclusion, the decision tree regressor machine learning model was able to how various economic sectors respond to fluctuations in the effective federal funds rate.

Data Attribution & References:

EFFR data was obtained from FRED (Federal Reserve Economic Data):
Name: Federal Funds Effective Rate
https://fred.stlouisfed.org/series/FEDFUNDS

Stock data for each sector was obtained from the Yahoo finance library called yfinance. From here I obtained the stock data for the stocks:
XLK: The Technology Select Sector SPDR Fund
XLV: The Health Care Select Sector SPDR Fund
XLF: The Financial Select Sector SPDR Fund
XLY: The Consumer Discretionary Select Sector SPDR Fund

I also did some research that helped me understand more about what stocks to choose and what interest rate to look at using the following resources:

Federal Reserve Bank of New York. "Effective Federal Funds Rate." *Federal Reserve Bank of New York*, https://www.newyorkfed.org/markets/reference-rates/effr.

Office of Financial Research. "Effective Federal Funds Rate." *Short-Term Funding Monitor*, https://www.financialresearch.gov/short-term-funding-monitor/datasets/fnyr-single/?mnemonic=FNYR-EFFR-A.

Chen, James. "Closing Price." *Investopedia*, 27 Jan. 2022, https://www.investopedia.com/terms/c/closingprice.asp.