HW 3

January 26, 2024

[1]:
```python
import pandas as pd
import matplotlib.pyplot as plt
```

0.1 Clean the Apple Data to get a quarterly series of EPS.

[2]:
```python
 y = pd.read_csv("AAPL_quarterly_financials.csv")
```

[3]:
```python
y.index = y.name
```

[4]:
```python
y = pd.DataFrame(y.loc["BasicEPS", :]).iloc[2:,:]
```

[5]:
```python
y.index = pd.to_datetime(y.index)
```

[6]:
```python
# CHECK IF NAS ARE NO DIVIDEND PERIOD
y = y.sort_index().fillna(0.)
```

0.2 6 search terms to nowcast earnings. Add in 3 terms that will not Nowcast earnings. Pull in the gtrends data

[7]:
```python
from pytrends.request import TrendReq
```

[8]:
```python
# Create pytrends object
pytrends = TrendReq(hl='en-US', tz=360)



# Set up the keywords and the timeframe
keywords = [ "MacBook", "FaceTime","iOS", "Apple Watch","Apple TV",
             "Design", "iTunes","AirPods","Star Wars","BasketBall","Candy"]
start_date = '2004-01-01'
end_date = '2024-01-01'

# Create an empty DataFrame to store the results
df = pd.DataFrame()

# Iterate through keywords and fetch data
for keyword in keywords:
    pytrends.build_payload([keyword], cat=0, timeframe=f'{start_date}␣
 ↪{end_date}', geo='', gprop='')
```

```
    interest_over_time_df = pytrends.interest_over_time()
    df[keyword] = interest_over_time_df[keyword]
```

[9]:
```
X = df.resample("Q").mean()
X
```

[9]:
```
              MacBook   FaceTime        iOS  Apple Watch   Apple TV  \
date
2004-03-31   0.000000   0.333333   1.000000     0.000000   2.000000
2004-06-30   0.000000   0.000000   1.666667     0.000000   1.333333
2004-09-30   0.000000   0.000000   1.333333     0.000000   2.000000
2004-12-31   0.000000   0.666667   1.000000     0.000000   2.000000
2005-03-31   0.000000   0.000000   1.000000     0.000000   2.000000
…                 …         …         …          …         …
2023-03-31  73.666667  22.333333  13.666667    67.000000  59.000000
2023-06-30  69.000000  21.000000  14.000000    66.666667  62.000000
2023-09-30  79.333333  20.333333  16.666667    74.333333  71.666667
2023-12-31  81.333333  23.666667  13.666667    83.000000  66.333333
2024-03-31  81.000000  26.000000  14.000000    73.000000  71.000000

               Design     iTunes    AirPods  Star Wars  BasketBall      Candy
date
2004-03-31  98.000000  11.666667   0.000000  12.000000   53.333333  29.333333
2004-06-30  91.666667  13.333333   0.000000  10.666667   26.333333  28.000000
2004-09-30  87.333333  14.666667   0.000000  15.333333   23.333333  28.333333
2004-12-31  79.000000  18.000000   0.000000  18.333333   34.333333  36.000000
2005-03-31  86.666667  21.666667   0.000000  19.000000   56.333333  43.666667
…                 …         …         …         …          …          …
2023-03-31  65.333333   6.000000  55.333333   6.666667   69.666667  31.333333
2023-06-30  67.333333   5.000000  48.000000   8.666667   25.000000  32.000000
2023-09-30  63.666667   5.666667  52.666667   6.000000   21.333333  30.000000
2023-12-31  66.000000   5.000000  62.666667   6.666667   38.000000  41.333333
2024-03-31  63.000000   5.000000  54.000000   6.000000   52.000000  36.000000

[81 rows x 11 columns]
```

[10]:
```
# ALIGN DATA
temp = pd.concat([y, X],axis = 1).dropna()
y = temp[["BasicEPS"]].copy()
X = temp.iloc[:,1:].copy()
```

0.3 Normalize all the X data

[11]:
```
from sklearn.preprocessing import StandardScaler
```

[12]:
```
scaler = StandardScaler()
```

[13]:
```
X_scaled = scaler.fit_transform(X)
```

0.4 Plot a bar chart

0.4.1 A. With Lambda = 0

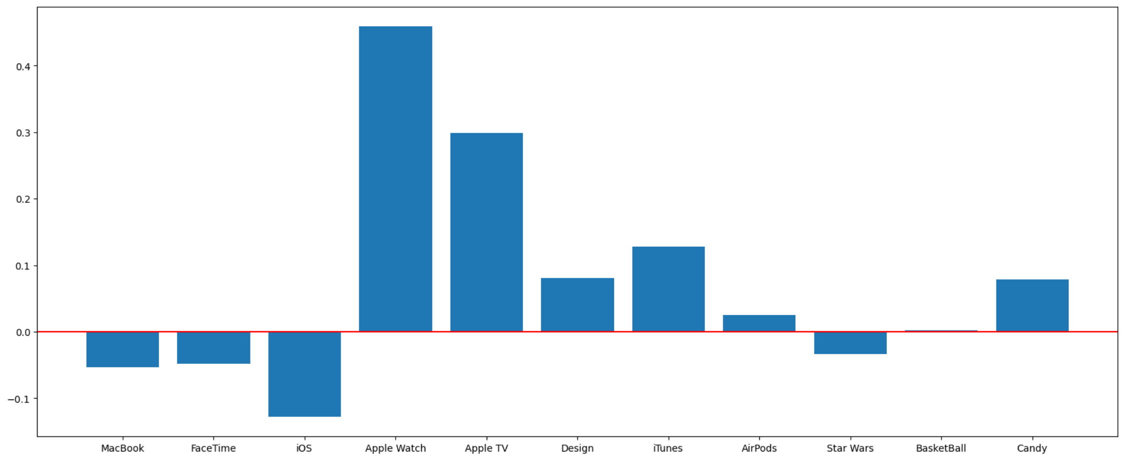
[14]:
```
from sklearn.linear_model import Lasso
```

[15]:
```
lasso = Lasso(alpha = 0)
lasso.fit(X_scaled, y)
```

```
/Users/raghavirajumohan/anaconda3/lib/python3.11/site-
packages/sklearn/base.py:1151: UserWarning: With alpha=0, this algorithm does
not converge well. You are advised to use the LinearRegression estimator
  return fit_method(estimator, *args, **kwargs)
/Users/raghavirajumohan/anaconda3/lib/python3.11/site-
packages/sklearn/linear_model/_coordinate_descent.py:628: UserWarning:
Coordinate descent with no regularization may lead to unexpected results and is
discouraged.
  model = cd_fast.enet_coordinate_descent(
/Users/raghavirajumohan/anaconda3/lib/python3.11/site-
packages/sklearn/linear_model/_coordinate_descent.py:628: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 8.377e-01, tolerance: 2.085e-03 Linear regression models with null weight
for the l1 regularization term are more efficiently fitted using one of the
solvers implemented in sklearn.linear_model.Ridge/RidgeCV instead.
  model = cd_fast.enet_coordinate_descent(
```

[15]:
```
Lasso(alpha=0)
```

[16]:
```
plt.figure(figsize = (20,8))
coefficients = lasso.coef_
plt.bar(range(len(coefficients)), coefficients,tick_label = X.columns)
plt.axhline(0.,color = "red")
plt.show()
```

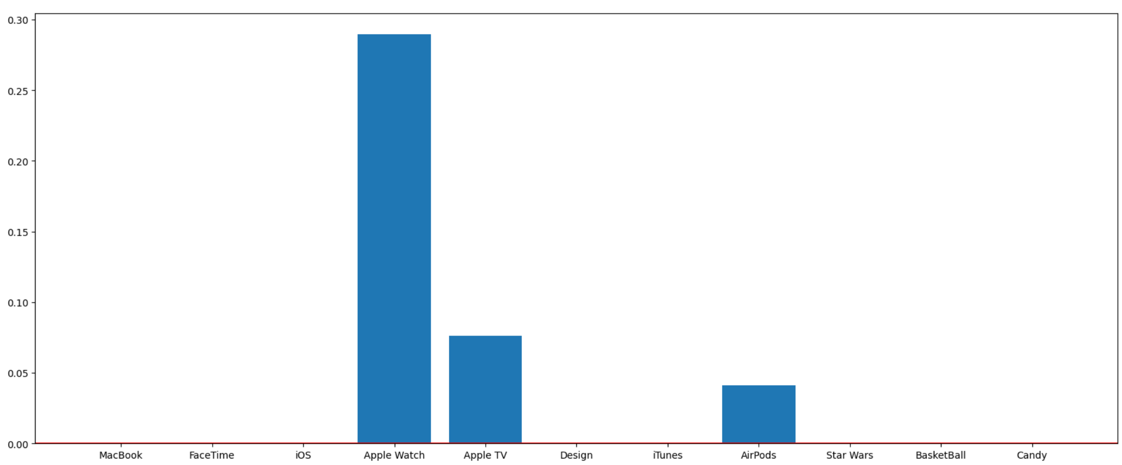


0.4.2 B. With Lambda = 0.1

[17]:
```python
lasso = Lasso(alpha = 0.1)
lasso.fit(X_scaled, y)
```

[17]: Lasso(alpha=0.1)

[18]:
```python
plt.figure(figsize = (20,8))
coefficients = lasso.coef_
plt.bar(range(len(coefficients)), coefficients,tick_label = X.columns)
plt.axhline(0.,color = "red")
plt.show()
```



0.5 Do these coefficient magnitudes make sense?

In Lasso regression, the regularization parameter lambda ( ) controls the amount of shrinkage applied to the coefficients. The role of lambda is to add a penalty term to the model's cost function, which encourages the model to prefer simpler models with fewer non-zero coefficients.

1. Case 1, $= 0$

- No regularization penalty is applied, the model behaves like a linear regression model. The inclusion of non-zero coefficients for most terms suggests that, according to this model, all terms are considered important for predicting Apple's earnings per share, with only "Basketball" considered to be irrelevant.

2. Case 2, $= 0.1$

- As lambda increases, the regularization penalty becomes stronger.
- Non-zero coefficients: Apple Watch, Apple TV and AirPods. So according to the Lasso analysis, only these search terms have a meaningful impact or association with Apple's earnings per share (EPS). The model has decided to retain these features as they contribute useful information in predicting the target variable.
- The fact that coefficients for terms like "MacBook," "FaceTime," "iOS," etc., are zero suggests that, according to the Lasso analysis, these search terms do not provide significant information or influence the prediction of Apple's EPS. The model has effectively disregarded these terms, considering them less relevant for the task at hand.