yield-curve-analysis

September 30, 2023

[1]:
```
import yfinance as yf
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
```

[3]:
```
datasets =  ['^IRX', '^FVX', '^TNX', '^TYX']

for dataset in datasets :
    Ticker = yf.Ticker(dataset)
    data = Ticker.history(start='2023-08-01', end='2023-09-28')
    filename = f"{dataset}_data.csv"
    data.to_csv(filename)
    print(f"Download data for {dataset} and saved as {filename}")
```

```
Download data for ^IRX and saved as ^IRX_data.csv
Download data for ^FVX and saved as ^FVX_data.csv
Download data for ^TNX and saved as ^TNX_data.csv
Download data for ^TYX and saved as ^TYX_data.csv
```

[6]:
```
Ticker  ='^IRX', '^FVX', '^TNX', '^TYX'
start_date = '2023-08-01'
end_date = '2023-09-28'
data = yf.download(Ticker, start=start_date, end=end_date, period='1d')['Adj␣
 ↪Close']/100
```

```
[**********************100%%**********************]  4 of 4 completed
```

[7]:
```
data
```

[7]:

Date	^FVX	^IRX	^TNX	^TYX
2023-08-01	0.04249	0.05260	0.04051	0.04105
2023-08-02	0.04241	0.05260	0.04078	0.04165
2023-08-03	0.04301	0.05258	0.04189	0.04304
2023-08-04	0.04163	0.05253	0.04060	0.04214
2023-08-07	0.04151	0.05260	0.04078	0.04258
2023-08-08	0.04113	0.05275	0.04026	0.04204
2023-08-09	0.04128	0.05280	0.04012	0.04179

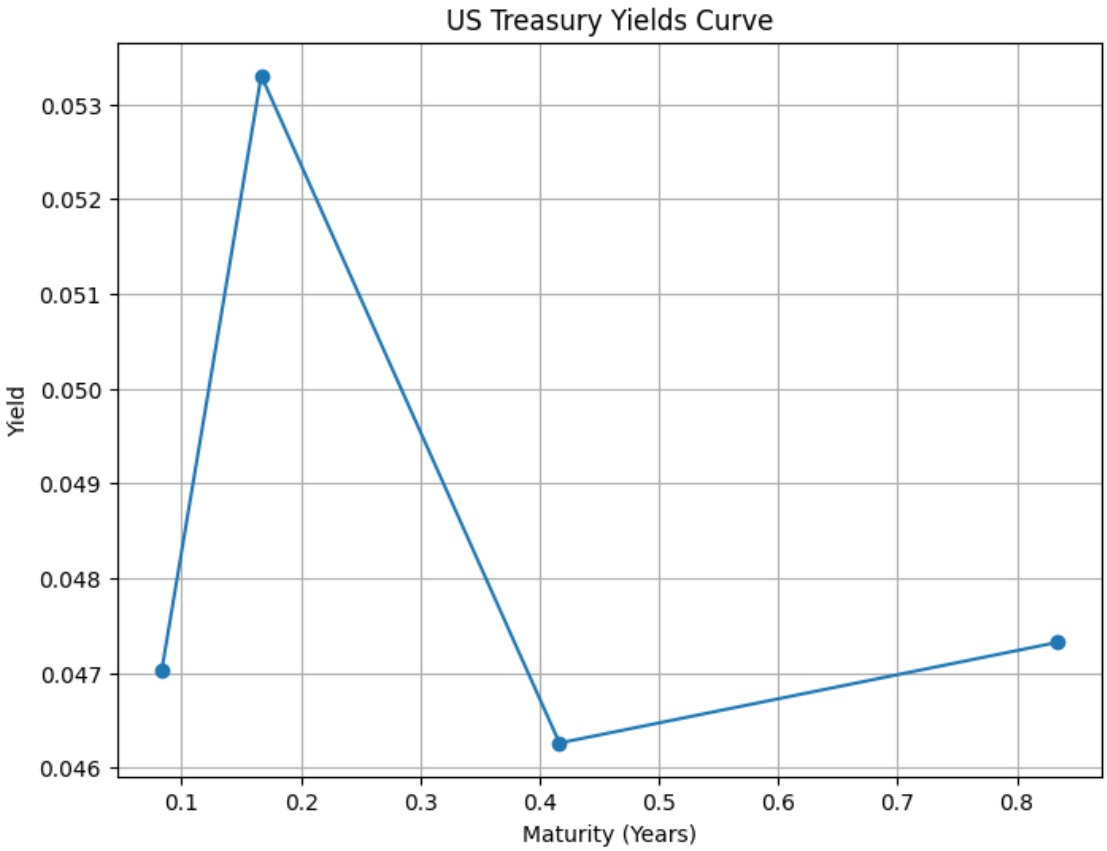
```
2023-08-10  0.04202  0.05268  0.04080  0.04241
2023-08-11  0.04308  0.05265  0.04168  0.04272
2023-08-14  0.04356  0.05270  0.04184  0.04281
2023-08-15  0.04380  0.05280  0.04221  0.04319
2023-08-16  0.04408  0.05285  0.04258  0.04361
2023-08-17  0.04439  0.05278  0.04308  0.04412
2023-08-18  0.04382  0.05278  0.04251  0.04379
2023-08-21  0.04459  0.05280  0.04342  0.04456
2023-08-22  0.04479  0.05295  0.04328  0.04410
2023-08-23  0.04361  0.05295  0.04198  0.04284
2023-08-24  0.04404  0.05303  0.04235  0.04301
2023-08-25  0.04433  0.05315  0.04239  0.04294
2023-08-28  0.04412  0.05323  0.04212  0.04291
2023-08-29  0.04277  0.05320  0.04122  0.04237
2023-08-30  0.04273  0.05315  0.04118  0.04228
2023-08-31  0.04244  0.05298  0.04093  0.04204
2023-09-01  0.04291  0.05268  0.04173  0.04285
2023-09-05  0.04384  0.05278  0.04268  0.04376
2023-09-06  0.04438  0.05300  0.04290  0.04359
2023-09-07  0.04380  0.05283  0.04260  0.04352
2023-09-08  0.04396  0.05293  0.04258  0.04332
2023-09-11  0.04414  0.05293  0.04288  0.04377
2023-09-12  0.04410  0.05305  0.04264  0.04346
2023-09-13  0.04391  0.05305  0.04249  0.04337
2023-09-14  0.04419  0.05295  0.04288  0.04385
2023-09-15  0.04453  0.05298  0.04322  0.04411
2023-09-18  0.04462  0.05300  0.04319  0.04396
2023-09-19  0.04521  0.05310  0.04365  0.04428
2023-09-20  0.04515  0.05315  0.04349  0.04401
2023-09-21  0.04616  0.05305  0.04480  0.04552
2023-09-22  0.04569  0.05305  0.04438  0.04521
2023-09-25  0.04620  0.05313  0.04542  0.04658
2023-09-26  0.04625  0.05325  0.04558  0.04696
2023-09-27  0.04703  0.05330  0.04626  0.04732
```

[16]:
```
data.tail()
```

[16]:
```
              ^FVX     ^IRX     ^TNX     ^TYX
Date
2023-09-21  0.04616  0.05305  0.04480  0.04552
2023-09-22  0.04569  0.05305  0.04438  0.04521
2023-09-25  0.04620  0.05313  0.04542  0.04658
2023-09-26  0.04625  0.05325  0.04558  0.04696
2023-09-27  0.04703  0.05330  0.04626  0.04732
```

[8]:
```
maturities = [1/12, 2/12, 5/12, 10/12]
yields = data.iloc[-1].values
```

[9]:
```
plt.figure(figsize=(8, 6))
plt.plot(maturities, yields, marker='o', linestyle='-')
plt.title('US Treasury Yields Curve')
plt.xlabel('Maturity (Years)')
plt.ylabel('Yield')
plt.grid(True)
plt.show()
```



[10]:
```
! pip install scipy
```

```
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages
(1.11.2)
Requirement already satisfied: numpy<1.28.0,>=1.21.6 in
/usr/local/lib/python3.10/dist-packages (from scipy) (1.23.5)
```

[11]:
```
from scipy.optimize import curve_fit
```

[13]:
```
def nelson_siegel(maturity, beta0, beta1, beta2, tau):
```

```
    return beta0 + (beta1 * (1 - np.exp(-maturity/tau)) / (maturity/tau)) +␣
 ↪(beta2 * ((1 - np.exp(-maturity/ tau)) / (maturity/tau) - np.exp(-maturity/
 ↪tau)))

popt, _ = curve_fit(nelson_siegel, maturities, yields )

beta0, beta1, beta2, tau = popt
```

[15]:
```
fitted_yields = nelson_siegel(np.array(maturities), *popt)
plt.figure(figsize=(8,6))
plt.plot(maturities, yields, marker='o', linestyle='-', label='Actual Data')
plt.plot(maturities, fitted_yields, linestyle='--', label='Nelson-Siegel Model')
plt.title('U.S Treasury Yields Cuvre vs Fitted Nelson-Siegel Model')
plt.xlabel('Maturity (Years)')
plt.ylabel('Yield')
plt.legend()
plt.grid(True)
plt.show()
```