Project_Part1_Nimish_Agarwal

June 26, 2019

[13]:

```
#       CODE FOR "PORTFOLIO OPTIMISATION" - SUBMITTED BY NIMISH AGARWAL
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import pandas_datareader as dr
from datetime import datetime
pd.options.display.float_format = '{:,.2f}'.format
```

[ ]:

```
# Web Scapping Stocks Data
companies = ['goog','mo','dal','fb','vedl']
#,'amzn','aapl','t','aa','axp','DB','AEM','APD','AMBA','NVS','ANF','LULU']
start = datetime(2016, 1, 1)
end = datetime(2019, 5, 1)

df = dr.data.get_data_yahoo(companies, start, end, interval='m')

# Getting Returns
data = df[['Adj Close']]
log_data = np.log(data)
df = log_data['Adj Close'].diff()
ndata = 100*df
ndata = ndata.dropna()

# Saving Scapped Data
ndata.to_csv('stocks.csv')
```

[14]:

```
df = pd.read_csv('stocks.csv')

mean_return = df.mean(axis=0)  # Stocks Return
std_dev = df.std(axis=0)  # Stocks Risk (Standard Deviation)
cov_matrix = np.matrix(df.cov())  # Stocks Covariance Matrix
corr_matrix = df.corr()  # Stocks Correlation Matrix
```

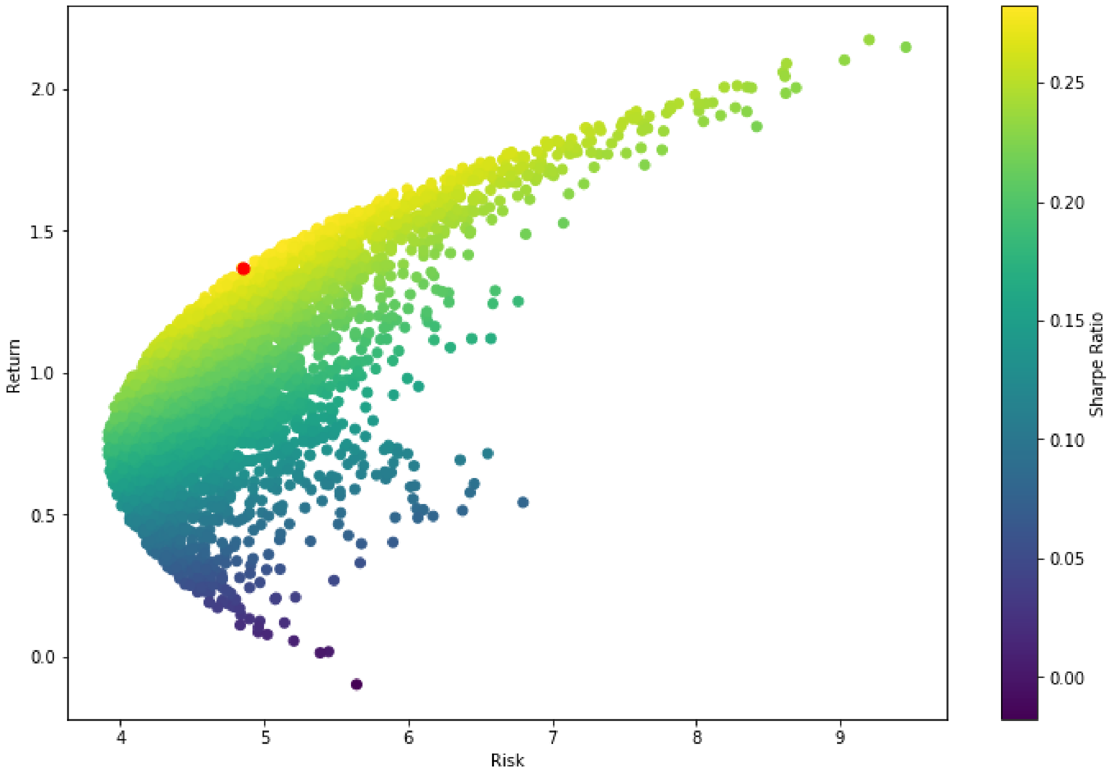
[15]:

```
import random  # For generating weights

ports = 5000  # Count of Total Portfolios
(m, n) = df.shape
```

```
portfolios = []  # Set of Portfolios
all_weights = []  # Set of Portfolio Weights

for i in range(1, ports):
#    Generating Weights
    w = [np.sqrt(random.random()*random.random())*(random.random()*50)
         for i in range(1,n)]
    s = sum(w)
    weight = [ i/s for i in w ]
    all_weights.append(weight)
#     Portfolio Properties
    portfolio_return = np.dot(weight, mean_return)
    variance = np.matmul(np.matmul(weight, cov_matrix), np.transpose(weight))
    portfolio_std_dev = np.sqrt(variance[0,0])
    sharpe_ratio = portfolio_return / portfolio_std_dev  # Assuming Rf=0
#     Add Portfolio to the list
    portfolios.append((portfolio_return, portfolio_std_dev, sharpe_ratio))

portfolios = pd.DataFrame(portfolios,
                          columns=['Return', 'Std. Dev.', 'Sharpe Ratio'])
```

[18]:

```
# Plot Portfolios
plt.figure(figsize=(12,8))
plt.scatter(portfolios.iloc[:,1], portfolios.iloc[:,0],
            c=portfolios.iloc[:,2], cmap='viridis')
plt.colorbar(label='Sharpe Ratio')
plt.xlabel('Risk')
plt.ylabel('Return')

# Find Point with Maximum sharpe Ratio
idx = np.argmax(portfolios.iloc[:,2])
plt.scatter(portfolios.iloc[idx, 1], portfolios.iloc[idx, 0], c='red', s=50) #␣
 ↪Plotting the point (red dot)

plt.show()
```



[19]:

```
# Return Weights with Max Sharpe Ratio
opt_weight = all_weights[idx]
opt_weight
```

[19]:

```
[0.18022122798542684,
 0.14687938901646788,
 0.3669573288202598,
 0.004281352760713035,
 0.3016607014171324]
```