PART-A

Selection of five stocks, traded on Bombay Stock Exchange (BSE) or National Stock Exchange (NSE) and download of prices from yahoo finance

```
import pandas as pd
import numpy as np
from pandas_datareader import data as pdr import
yfinance as yf
import datetime as dt
import   matplotlib.pyplot   as   plt
import pandas_datareader as web from
scipy import stats
from pandas_datareader import data as pdr import
yfinance as yf
```

```
tickers=['^BSESN','GILLETTE.BO','NESCO.BO','SONATSOFTW.BO','TATAELXSI.
BO','TATAMOTORS.BO']
```

```
data = pdr.get_data_yahoo(tickers, start="2017-01-01",end="2022-01- 01")['Adj Close']
returns = data.pct_change(1).dropna()
returns.head(10)
```

```
Symbols          ^BSESN  GILLETTE.BO  NESCO.BO  SONATSOFTW.BO
TATAELXSI.BO  ¥
Date

2017-01-03       0.001797      -0.005411    0.032280          0.026430        -
0.006449
2017-01-04      -0.000379      -0.000237    0.039474         -0.007428
0.012234
2017-01-05       0.009203      -0.002260    0.008322          0.038663
0.010289
2017-01-06      -0.004428       0.000427   -0.001945         -0.054275        -
0.028424
2017-01-09      -0.001221      -0.000818   -0.001972         -0.010157        -
0.006138
2017-01-10       0.006473       0.001079   -0.001000          0.015392
0.019106
2017-01-11       0.008954      -0.002086   -0.005553          0.007327        -
0.008222
2017-01-12       0.003933      -0.000059   -0.013565         -0.007524
0.002716
2017-01-13    -0.000334        0.000154   0.003013             0.003791       -
0.003207
2017-01-16     0.001840       -0.002458   0.000412            -0.009315
0.011870
```

```
Symbols        TATAMOTORS.BO
Date
2017-01-03         -0.012314
2017-01-04          0.010909
2017-01-05          0.031757
2017-01-06         -0.008268
2017-01-09          0.005123
2017-01-10          0.029879
2017-01-11          0.006986
2017-01-12         -0.001060
2017-01-13         -0.007138
2017-01-16          0.023122
```

data

```
Symbols             ^BSESN    GILLETTE.BO      NESCO.BO    SONATSOFTW.BO     ¥
Date
2017-01-02     26595.449219   3896.873047   378.418243      160.056503
2017-01-03     26643.240234   3875.785400   390.633759      164.286743
2017-01-04     26633.130859   3874.868652   406.053619      163.066467
2017-01-05     26878.240234   3866.113281   409.432831      169.371109
2017-01-06     26759.230469   3867.762695   408.636536      160.178482
...                     ...           ...          ...             ...
2021-12-27     57420.238281   5214.617676   571.096313      838.112183
2021-12-28     57897.480469   5196.808594   578.457581      834.271423
2021-12-29     57806.488281   5172.240723   578.159180      845.696472
2021-12-30     57794.320312   5191.233887   575.274353      846.863281
2021-12-31     58253.820312   5241.010254   578.955017      851.481934

Symbols        TATAELXSI.BO     TATAMOTORS.BO 487.250000
Date
2017-01-02       663.228088
2017-01-03       658.950684      481.250000
2017-01-04       667.011963      486.500000
2017-01-05       673.874634      501.950012
2017-01-06       654.720459      497.799988
...                     ...             ...
2021-12-27      5417.303223      471.299988
2021-12-28      5501.772949      480.100006
2021-12-29      5678.522949      475.799988
2021-12-30      5783.239258      470.350006
2021-12-31      5838.059570      482.350006

[1236 rows x 6 columns]
```

```
mean_daily_ret     = data.pct_change(1).mean()
mean_daily_ret
```

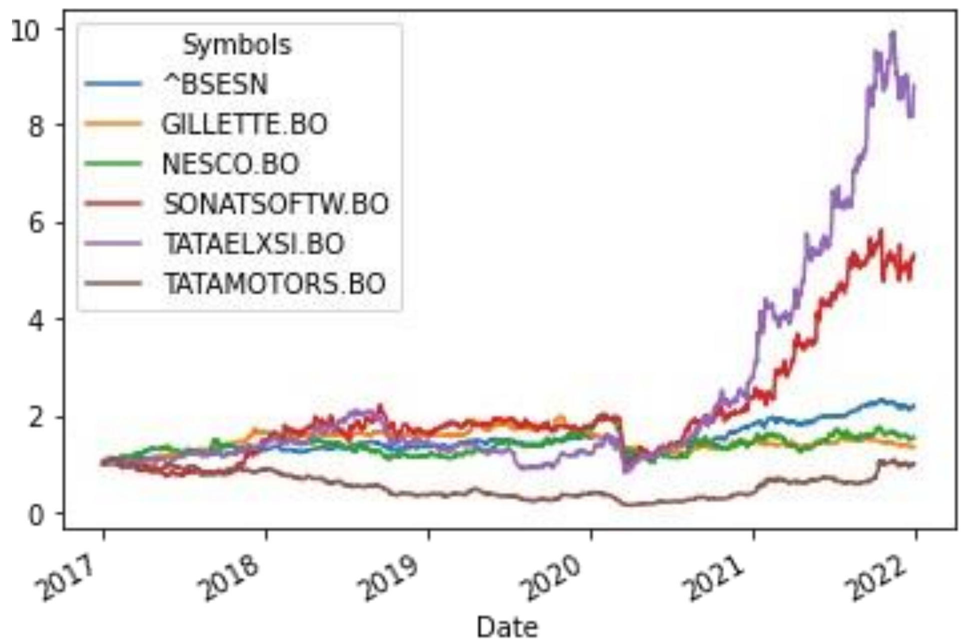
```
Symbols
^BSESN           0.000704
GILLETTE.BO      0.000305
NESCO.BO         0.000555
SONATSOFTW.BO    0.001678
TATAELXSI.BO     0.002045
TATAMOTORS.BO    0.000447
dtype: float64
```

```
data.pct_change(1).corr()
```

```
Symbols            ^BSESN  GILLETTE.BO  NESCO.BO  SONATSOFTW.BO
TATAELXSI.BO  ¥
Symbols

^BSESN          1.000000     0.348634    0.378006       0.297889
0.393798
GILLETTE.BO     0.348634     1.000000    0.164589       0.191244
0.264459
NESCO.BO        0.378006     0.164589    1.000000       0.150330
0.240774
SONATSOFTW.BO   0.297889     0.191244    0.150330       1.000000
0.269108
TATAELXSI.BO    0.393798     0.264459    0.240774       0.269108
1.000000
TATAMOTORS.BO   0.510698     0.237534    0.218542       0.183235
0.278511

Symbols         TATAMOTORS.BO
Symbols
^BSESN               0.510698
GILLETTE.BO          0.237534
NESCO.BO             0.218542
SONATSOFTW.BO        0.183235
TATAELXSI.BO         0.278511
TATAMOTORS.BO        1.000000
```

```
data_normed = data/data.iloc[0]
data_normed.plot()
```

```
<AxesSubplot:xlabel='Date'>
```




```
data_daily_ret = data.pct_change(1) data_daily_ret
```

```
Symbols          ^BSESN  GILLETTE.BO  NESCO.BO  SONATSOFTW.BO
TATAELXSI.BO  ¥
Date

2017-01-02          NaN          NaN       NaN            NaN
NaN
2017-01-03     0.001797    -0.005411  0.032280       0.026430      -
0.006449
2017-01-04    -0.000379    -0.000237  0.039474      -0.007428
0.012234
2017-01-05     0.009203    -0.002260  0.008322       0.038663
0.010289
2017-01-06    -0.004428     0.000427 -0.001945      -0.054275      -
0.028424
...                 ...          ...       ...            ...            .
..
2021-12-27     0.005180    -0.016332  0.001395       0.025826      -
0.010297
2021-12-28     0.008311    -0.003415  0.012890      -0.004583
0.015593
2021-12-29    -0.001572    -0.004727 -0.000516       0.013695
0.032126
2021-12-30    -0.000210     0.003672 -0.004990       0.001380
0.018441
2021-12-31     0.007951     0.009589  0.006398       0.005454
0.009479
```

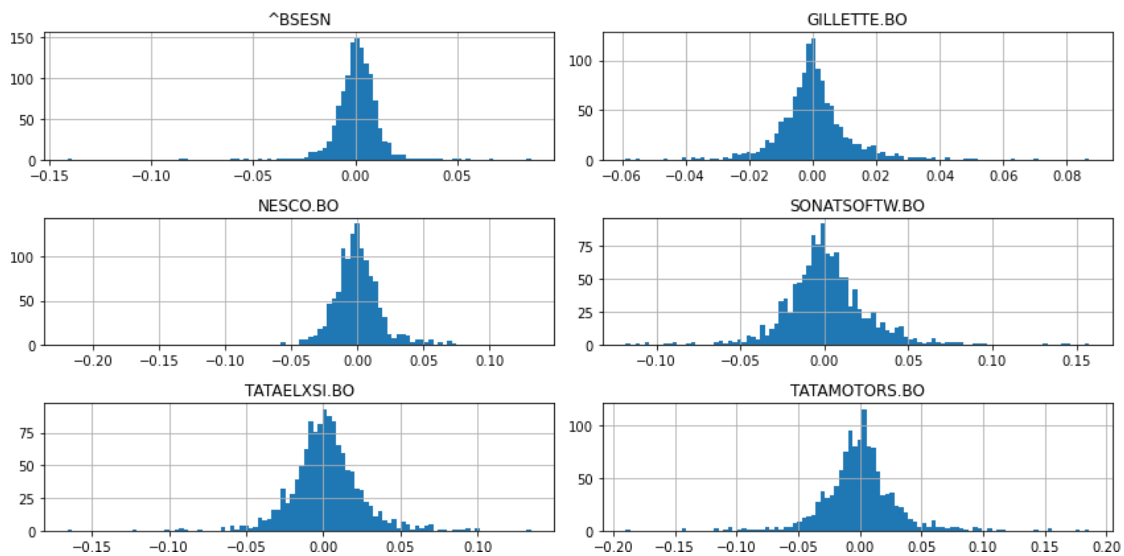
```
Symbols          TATAMOTORS.BO
Date
2017-01-02                 NaN
2017-01-03           -0.012314
2017-01-04            0.010909
2017-01-05            0.031757
2017-01-06           -0.008268
...                        ...
2021-12-27            0.008021
2021-12-28            0.018672
2021-12-29           -0.008957
2021-12-30           -0.011454
2021-12-31            0.025513

[1236 rows x 6 columns]
```

```
log_ret = np.log(data/data.shift(1)) log_ret
```

```
Symbols         ^BSESN  GILLETTE.BO  NESCO.BO  SONATSOFTW.BO
TATAELXSI.BO  ¥
Date

2017-01-02           NaN          NaN       NaN            NaN
NaN
2017-01-03      0.001795    -0.005426  0.031770       0.026086      -
0.006470
2017-01-04     -0.000380    -0.000237  0.038715      -0.007455
0.012159
2017-01-05      0.009161    -0.002262  0.008288       0.037934
0.010236
2017-01-06     -0.004438     0.000427 -0.001947      -0.055804      -
0.028836
...                  ...          ...       ...            ...            .
..
2021-12-27      0.005167    -0.016467  0.001394       0.025498      -
0.010350
2021-12-28      0.008277    -0.003421  0.012807      -0.004593
0.015472
2021-12-29     -0.001573    -0.004739 -0.000516       0.013602
0.031621
2021-12-30     -0.000211     0.003665 -0.005002       0.001379
0.018273
2021-12-31      0.007919     0.009543  0.006378       0.005439
0.009435

Symbols        TATAMOTORS.BO
Date
2017-01-02               NaN
2017-01-03         -0.012390
```

```
2017-01-04              0.010850
2017-01-05              0.031264
2017-01-06             -0.008302
...                          ...
2021-12-27              0.007989
2021-12-28              0.018500
2021-12-29             -0.008997
2021-12-30             -0.011520
2021-12-31              0.025193

[1236 rows x 6 columns]
```

```
log_ret.hist(bins=100,figsize=(12,6))
plt.tight_layout()
```




```
log_ret.describe().transpose()
```

```
                    count        mean         std         min          25%
50%  ¥
Symbols

^BSESN              1225.0     0.000602    0.011774   -0.141017    -0.004297
0.000898
GILLETTE.BO         1235.0     0.000240    0.011402   -0.058816    -0.005079   -
0.000237
NESCO.BO            1235.0     0.000344    0.020516   -0.219048    -0.010031   -
0.000595
SONATSOFTW.BO       1235.0     0.001353    0.025360   -0.118081    -0.011613   -
0.000221
TATAELXSI.BO        1235.0     0.001761    0.023751   -0.165787    -0.010036
0.001283
```

```
TATAMOTORS.BO      1235.0    -0.000008    0.030136    -0.189675    -0.013630
0.000358
                          75%         max
Symbols
^BSESN              0.006319     0.085947
GILLETTE.BO         0.004869     0.086985
NESCO.BO            0.009272     0.131324
SONATSOFTW.BO       0.012632     0.157527
TATAELXSI.BO        0.013274     0.134614
TATAMOTORS.BO       0.012556     0.185860
```

```
log_ret.mean()*252
```

```
Symbols
^BSESN               0.151709
GILLETTE.BO          0.060468
NESCO.BO             0.086767
SONATSOFTW.BO        0.341057
TATAELXSI.BO         0.443813
TATAMOTORS.BO       -0.002062
dtype: float64
```

```
log_ret.cov()
```

```
Symbols                 ^BSESN  GILLETTE.BO  NESCO.BO  SONATSOFTW.BO
TATAELXSI.BO  ¥
Symbols

^BSESN             0.000139         0.000047     0.000096              0.000093
0.000112
GILLETTE.BO        0.000047         0.000130     0.000041              0.000057
0.000073
NESCO.BO           0.000096         0.000041     0.000421              0.000084
0.000122
SONATSOFTW.BO      0.000093         0.000057     0.000084              0.000643
0.000168
TATAELXSI.BO       0.000112         0.000073     0.000122              0.000168
0.000564
TATAMOTORS.BO      0.000181         0.000083     0.000141              0.000148
0.000204

Symbols            TATAMOTORS.BO
Symbols
^BSESN                  0.000181
GILLETTE.BO             0.000083
NESCO.BO                0.000141
SONATSOFTW.BO           0.000148
TATAELXSI.BO            0.000204
TATAMOTORS.BO           0.000908
```

```
log_ret.cov()*252
```

```
Symbols             ^BSESN  GILLETTE.BO  NESCO.BO  SONATSOFTW.BO
TATAELXSI.BO  ¥
Symbols

^BSESN          0.034936        0.011939    0.024245            0.023463
0.028247
GILLETTE.BO     0.011939        0.032762    0.010238            0.014351
0.018456
NESCO.BO        0.024245        0.010238    0.106065            0.021163
0.030734
SONATSOFTW.BO   0.023463        0.014351    0.021163            0.162068
0.042306
TATAELXSI.BO    0.028247        0.018456    0.030734            0.042306
0.142158
TATAMOTORS.BO   0.045562        0.020889    0.035495            0.037293
0.051381

Symbols         TATAMOTORS.BO
Symbols
^BSESN                0.045562
GILLETTE.BO             0.020889
NESCO.BO                0.035495
SONATSOFTW.BO           0.037293
TATAELXSI.BO            0.051381
TATAMOTORS.BO           0.228855
```

```
np.random.seed()
print('Stocks :',data.columns)
weights = np.array(np.random.random(6)) print('¥nCreating
Random Weights :',weights) weights = weights /
np.sum(weights) print('¥nRebalance to sum to 1.0
:',weights) exp_ret = np.sum(log_ret.mean() * weights)
*252 print('¥nExpected Portfolio Return :',exp_ret)
exp_vol = np.sqrt(np.dot(weights.T, np.dot(log_ret.cov() * 252, weights)))
print('¥nExpected Volatility :',exp_vol) sr =
exp_ret/exp_vol
print('¥nSharpe Ratio :',sr)
```

```
Stocks : Index(['^BSESN', 'GILLETTE.BO', 'NESCO.BO', 'SONATSOFTW.BO', 'TATAELXSI.BO',
        'TATAMOTORS.BO'],
       dtype='object', name='Symbols')

Creating Random Weights : [0.3403384  0.30099014 0.34250491 0.72191529
0.59763824 0.98853346]

Rebalance to sum to 1.0 : [0.10338597 0.09143299 0.1040441  0.21929913
0.18154699 0.30029081]
```

Expected Portfolio Return : 0.18498803346586526 Expected

Volatility : 0.24773000893469002

Sharpe Ratio : 0.7467324377105817 num_ports =

20000

all_weights = np.zeros((num_ports,len(data.columns))) ret_arr =
np.zeros(num_ports)
vol_arr = np.zeros(num_ports)
sharpe_arr = np.zeros(num_ports)

```
for ind in range(num_ports):
    weights = np.array(np.random.random(6)) weights =
    weights / np.sum(weights) all_weights[ind,:] =
    weights
    ret_arr[ind] = np.sum((log_ret.mean() * weights) *252) vol_arr[ind] =
    np.sqrt(np.dot(weights.T, np.dot(log_ret.cov() *
252, weights)))
    sharpe_arr[ind] = ret_arr[ind]/vol_arr[ind]
```

```
variance = df_log_ret['^BSESN'].var() beta_val =
np.array(cov['^BSESN']/variance) for stocks in
list(df.columns):
    print('Beta for ',stocks,' is:
',round(beta_val[list(df.columns).index(stocks)],5))
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call
last)
Input In [17], in <cell line: 1>()
----> 1 variance = df_log_ret['^BSESN'].var()
      2 beta_val = np.array(cov['^BSESN']/variance)
      3 for stocks in list(df.columns):

NameError: name 'df_log_ret' is not defined sharpe_arr.max()
```

sharpe_arr.argmax()

all_weights[sharpe_arr.argmax(),:]

max_sr_ret = ret_arr[sharpe_arr.argmax()]
max_sr_vol = vol_arr[sharpe_arr.argmax()]

```
plt.figure(figsize=(12,8))
plt.scatter(vol_arr,ret_arr,c=sharpe_arr,cmap='magma')
plt.colorbar(label='Sharpe Ratio') plt.xlabel('Volatility')
plt.ylabel('Return') plt.scatter(max_sr_vol,max_sr_ret,c='red',s=50,edgecolors='black')

def get_ret_vol_sr(weights): weights =
    np.array(weights)
    ret = np.sum(log_ret.mean() * weights) * 252
    vol = np.sqrt(np.dot(weights.T, np.dot(log_ret.cov() * 252, weights)))
    sr = ret/vol
    return np.array([ret,vol,sr]) from

scipy.optimize import minimize

def neg_sharpe(weights):
    return  get_ret_vol_sr(weights)[2] * -1

def check_sum(weights):
    return np.sum(weights) - 1

cons = ({'type':'eq','fun': check_sum})

bounds = ((0, 1), (0, 1), (0, 1), (0, 1),(0, 1),(0,1))

init_guess = [0.25,0.25,0.25,0.25,0.25,0.25]

opt_results = minimize(neg_sharpe,init_guess,method='SLSQP',bounds=bounds,constraint
s=cons)

opt_results

opt_results.x

get_ret_vol_sr(opt_results.x) frontier_y =

np.linspace(0,0.3,100)

def minimize_volatility(weights):
    return  get_ret_vol_sr(weights)[1]

frontier_volatility = []

for possible_return in frontier_y:
    cons = ({'type':'eq','fun': check_sum},
            {'type':'eq','fun': lambda w: get_ret_vol_sr(w)[0] - possible_return})

    result = minimize(minimize_volatility,init_guess,method='SLSQP',bounds=bounds,c
```

```
onstraints=cons)

     frontier_volatility.append(result['fun'])

plt.figure(figsize=(12,8))
plt.scatter(vol_arr,ret_arr,c=sharpe_arr,cmap='magma')
plt.colorbar(label='Sharpe Ratio') plt.xlabel('Volatility')
plt.ylabel('Return') plt.plot(frontier_volatility,frontier_y,'g--
',linewidth=3)
```

PART-B

Calculation of portfolio mean, portfolio standard deviation and Sharpe Ratio for 100 portfolios and associated risk and return (portfolio mean) for each of the 100 portfolios

```
import pandas as pd
import numpy as np
from numpy import random
np.random.seed(101)
from dateutil.relativedelta import relativedelta

df = pd.DataFrame(random.rand(3000,6),columns=['^BSESN','GILLETTE.BO','NESC
O.BO','SONATSOFTW.BO','TATAELXSI.BO','TATAMOTORS.BO'])
df

mean_list = list(df.mean()) std_list =
list(df.std()) col_list =
list(df.columns)
for i in range(len(mean_list)):
     print(col_list[i],' has mean ',mean_list[i],' and standard deviation
',std_list[i])

#Risk free rate is 0.00
for i in range(len(col_list)):
     new_mean = (df.iloc[:,i].pct_change().mean())/100 sr =
     new_mean/std_list[i]
     print('Sharpe ratio for ',col_list[i],' is: ',sr)

log_ret = np.log(df/df.shift(1)).dropna()
log_ret.describe().transpose()

#Plotting of the statistics
ports = 20000

ret_arr = np.zeros(ports)
vol_arr = np.zeros(ports)
var_covar = log_ret.cov()*252 #Construct var_covar matrix
sharpe_arr = np.zeros(ports)
```

```
for vals in range(ports):

     weights = random.random(len(col_list)) weights =
     weights/np.sum(weights)
     ret_arr[vals] = np.sum(log_ret.mean() * weights)*252 #Yearly return
     vol_arr[vals] =
np.sqrt(np.dot(weights,np.dot(var_covar,weights.T)))
     sharpe_arr[vals] = ret_arr[vals]/vol_arr[vals]

import matplotlib.pyplot as plt
%matplotlib inline

plt.figure(figsize=(12,8))
plt.scatter(vol_arr,ret_arr,c=sharpe_arr,cmap='magma')
plt.colorbar(label='Sharpe Ratio') plt.xlabel('Volatility')
plt.ylabel('Return')

# Add red dot for max SR
plt.scatter(vol_arr[sharpe_arr.argmax()],ret_arr[sharpe_arr.argmax()],
c='Blue',s=50,edgecolors='green')
```

#Optimizing the Sharpe Ratio and finding the weights corresponding to Maximum Sharpe Ratio

```
from scipy.optimize import minimize

def ret_vol_sr(weights,rf):

     ret = np.sum((log_ret.mean()*weights))*252
     vol = np.sqrt(np.dot(weights,np.dot(var_covar,weights.T))) sharpe_arr = (ret -
     rf)/vol
     return np.array([ret,vol,sharpe_arr],dtype=object)


def neg_sharpe(weights): rf =

     rsk
     return ret_vol_sr(weights,rf)[2]*-1

def check_weight(weights):
     return (np.sum(weights)-1)

cons = ({'type':'eq','fun':check_weight}) bounds =

((0,1),(0,1),(0,1),(0,1),(0,1),(0,1))

init_guess =
```

```
np.array([0.166,0.166,0.166,0.166,0.166,0.17],dtype=object)

risk = np.sort(np.array([0.03,0.053,0.62,0.058,0.082,0.059,0.263,0.149,0.262,
0.076],dtype=object))

# Getting input from user for risk level #
Considering the risk level is upto 8 risk =
risk[:8]
sharpe_w = [] weights_w
= []

for rsk in risk:

    opt_results = minimize(neg_sharpe, init_guess, method='SLSQP', bounds=bounds,
constraints=cons)
    sharpe = ret_vol_sr(opt_results.x,rsk)[2]
    sharpe_w.append(sharpe)
    weights_w.append(opt_results.x)

temp = {}
for rsk in range(len(risk)):

    temp[risk[rsk]] = [sharpe_w[rsk]]+list(weights_w[rsk]) df1 =

pd.DataFrame(temp)
df1 = df1.transpose()
df1.index.name = 'Risk Rate'
df1.columns = ['Sharpe','w1','w2','w3','w4','w5','w6'] df1

df1.head(1)

#Get Beta Values
#Formula = cov(X,Y)/var(^NSEI(market-standard))

variance = df_log_ret['^BSESN'].var() beta_val =
np.array(cov['^BSESN']/variance) for stocks in
list(df.columns):
    print('Beta for ',stocks,' is:
',round(beta_val[list(df.columns).index(stocks)],6))

    #Beta for Market standards is always 1.0
```