In [1]:

```
import math
import numpy as n
import csv
import matplotlib.pyplot as plt
import pandas as pd
import statistics



def function(string,string2,value_1,value_2):
    S = n.zeros(shape=(10,60))

    with open(string) as csv_file:
        csv_reader = csv.reader(csv_file, delimiter=',')
        line_count = 0
        for row in csv_reader:
            if line_count == 61:
                break
            if line_count == 0:
                line_count += 1
            else:
                f = -1
                for value in row:
                    if f==-1:
                        f= f + 1
                        continue
                    if value=='':
                        continue
                    S[f][line_count-1] = float(value)
                    f=f+1

                line_count = line_count + 1

    m = n.array([[0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0]])
    u = n.array([[1,1,1,1,1,1,1,1,1,1]])
    C = n.zeros(shape=(10,10))

    for i in range(0,10):
        for j in range(0,60):
            if j+1<60:
                S[i][j] = (S[i][j+1]-S[i][j])/(S[i][j])
            else:
                S[i][j] = S[i][j-1]

            S[i][j]*=12



    f = 0
    for arr in S:
        m[0][f] = arr.mean()
        f+=1

    for i in range(0,10):
        for j in range(0,10):
            value = 0
            for k in range(0,60):
                value+= (S[i][k]-m[0][i])*(S[j][k]-m[0][j])
            C[i][j] = value/60
```

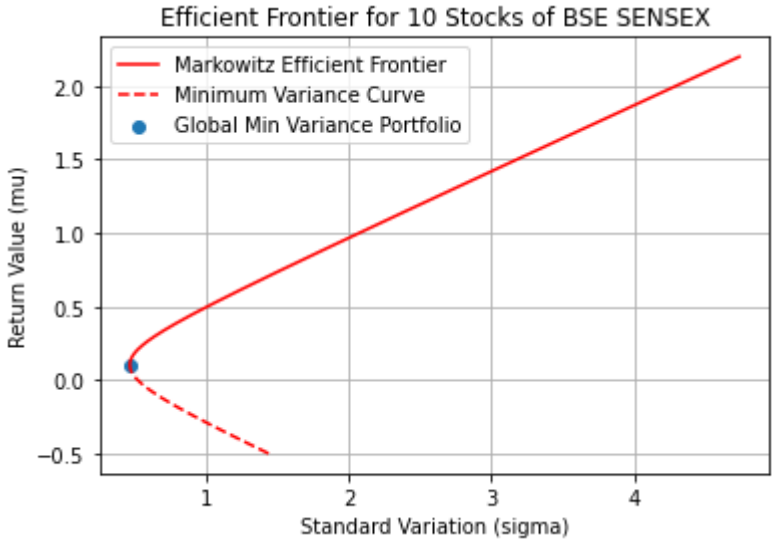
```
    C_inv = n.linalg.inv(C)
    mu_arr = n.linspace(-0.50,2.20,401)
    a = []
    b = []
    SD1_arr = []
    SD2_arr = []

    q_1 = (m.dot(C_inv)).dot(m.T)
    q_2 = (u.dot(C_inv)).dot(m.T)
    q_3 = (m.dot(C_inv)).dot(u.T)
    q_4 = (u.dot(C_inv)).dot(u.T)

    M = n.array([[q_1[0][0],q_2[0][0]],[q_3[0][0],q_4[0][0]]])

    M_inv = n.linalg.inv(M)


    Global_min_mu = q_2/q_4

    for mu in mu_arr:
        lamda = 2*M_inv.dot((n.array([[mu,1]])).T)
        w = (lamda[0][0]* (m.dot(C_inv)) + (lamda[1][0]*(u.dot(C_inv))))/2
        var = w.dot(C.dot(w.T))
        if mu<Global_min_mu[0][0]:
            a.append(mu)
            SD1_arr.append(math.sqrt(var[0][0]))
        else :
            b.append(mu)
            SD2_arr.append(math.sqrt(var[0][0]))


    Global_min_mu = q_2/q_4
    lamda = 2*M_inv.dot((n.array([[Global_min_mu[0][0],1]])).T)
    lam1 = lamda[0][0]
    lam2 = lamda[1][0]
    w = lam1*(m.dot(C_inv)) + lam2*(u.dot(C_inv))
    w = w/2
    Global_min_var = w.dot(C.dot(w.T))

    plt.scatter(math.sqrt(Global_min_var[0][0]),Global_min_mu[0][0],label="Globa
l Min Variance Portfolio")

    plt.plot(SD2_arr, b,c='r',label = "Markowitz Efficient Frontier")
    plt.plot(SD1_arr, a,c='r',label = "Minimum Variance Curve",linestyle='dashe
d')
    plt.xlabel("Standard Variation (sigma)")
    plt.ylabel("Return Value (mu)")
    plt.title("Efficient Frontier for "+string2)
    plt.grid(True)
    plt.legend()
    plt.show()


    rf = 0.05

    num = (m-rf*u).dot(C_inv)
    denom = (m-rf*u).dot(C_inv.dot(u.T))
    w = num/denom
```

```
    mu = w.dot(m.T)
    var = w.dot(C.dot(w.T))

    print("")
    print("Risk Free return = 0.05")
    print("The Return on Market portfolio :",round((mu[0][0]),4))
    print("The Risk on market portfolio :",round(math.sqrt(var[0][0]),4),"(",rou
nd(100*math.sqrt(var[0][0]),4),"% )")




    print("Risk Free Rate :",rf)
    point1 = (0,rf)
    point2 = (math.sqrt(var[0][0]),mu[0][0])

    x_values = [point1[0],point2[0],4]

    y_values = [point1[1], point2[1],rf+4.10*(point2[1]-rf)/point2[0]]


    plt.plot(x_values, y_values,label="CML")
    plt.plot(SD2_arr, b,c='r',label = "Markowitz Efficient Frontier")
    plt.plot(SD1_arr, a,c='r',label = "Minimum Variance Curve",linestyle='dashe
d')
    plt.scatter([math.sqrt(var[0][0])],[mu[0][0]],c='g',label="Market Portfolio"
)
    plt.xlabel("Standard Variation (sigma)")
    plt.ylabel("Return Value (mu)")

    plt.title("Markowitz efficient Frontier and CML for "+string2)
    plt.grid(True)
    plt.legend()
    plt.show()

    if string=="nse_non_index_data1.csv" or string =="bse_non_index_data1.csv":
        return

    beta = n.linspace(-2,2,401)
    mu_v = rf + (value_1 - rf)*beta;

    plt.plot(beta, mu_v)

    plt.xlabel("Beta Coefficient (beta)")
    plt.ylabel("Return Value (mu)")
    plt.title("Security Market Line for BSE")
    if string=="nsedata1.csv":
        plt.title("Security Market Line for NSE")

    plt.grid(True)
    plt.show()


bse_mean = 0
bse_var = 0

nse_mean = 0
nse_var = 0

bse_arr = []
nse_arr = []
```

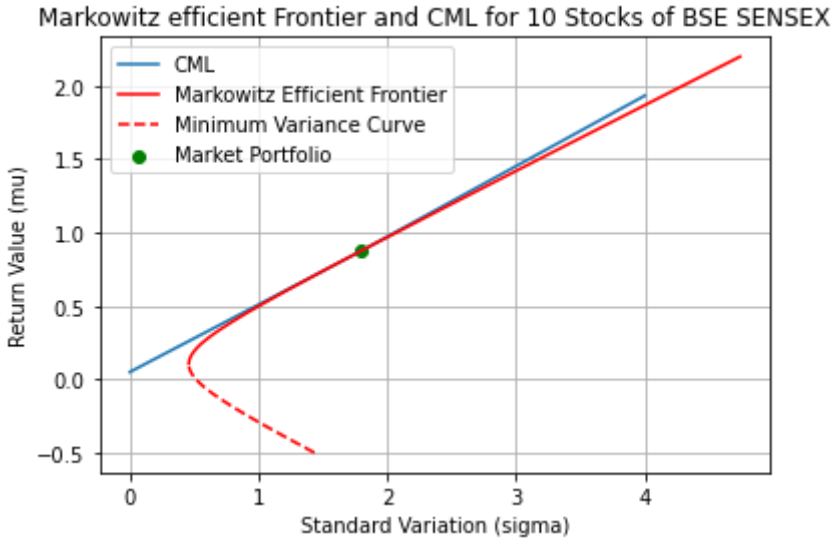
```
with open('bse_index.csv') as csv_file:
        csv_reader = csv.reader(csv_file, delimiter=',')
        line_count = 0
        for row in csv_reader:
            if line_count == 61:
                break
            if line_count == 0:
                line_count += 1
            else:
                bse_arr.append(float(row[1]))
                line_count += 1

bse_ret = []
for i in range(1,len(bse_arr)):
    bse_ret.append(12*(bse_arr[i]-bse_arr[i-1])/bse_arr[i-1])


bse_mean = n.mean(bse_ret)
bse_var = n.var(bse_ret)

with open('nse_index.csv') as csv_file:
        csv_reader = csv.reader(csv_file, delimiter=',')
        line_count = 0
        for row in csv_reader:
            if line_count == 61:
                break
            if line_count == 0:
                line_count += 1
            else:
                nse_arr.append(float(row[1]))
                line_count += 1

nse_ret = []
for i in range(1,len(nse_arr)):
    nse_ret.append(12*(nse_arr[i]-nse_arr[i-1])/nse_arr[i-1])


nse_mean = n.mean(nse_ret)
nse_var = n.var(nse_ret)

print("Expected return for NIFTY is:",nse_mean)
print("Variance for NIFTY :",nse_var)

print("")
print("Expected return for SENSEX is:",bse_mean)
print("Variance for SENSEX :",bse_var)

print("")
print("For 10 Stocks of BSE SENSEX")
function("bsedata1.csv","10 Stocks of BSE SENSEX",bse_mean,bse_var)

print("")
print("For 10 Stocks of NSE NIFTY")
function("nsedata1.csv","10 Stocks of NSE NIFTY",nse_mean,nse_var)

print("")
print("For 10 Stocks not included in BSE SENSEX")
function("nse_non_index_data1.csv","10 Stocks not in BSE SENSEX",bse_mean,bse_va
r)
```

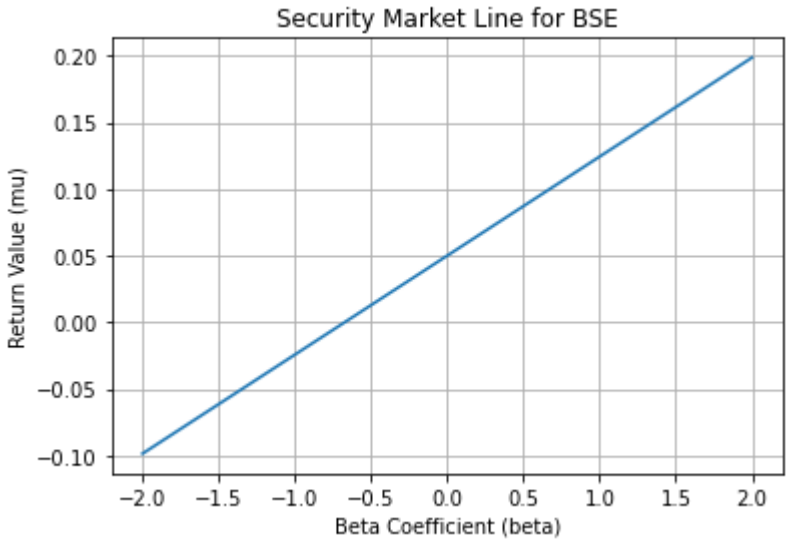
```
print("")
print("For 10 Stocks not included in NSE NIFTY")
function("bse_non_index_data1.csv","10 Stocks not in NSE NIFTY",nse_mean,nse_var
)
```

```
Expected return for NIFTY is: 0.1274085826169013
Variance for NIFTY : 0.22001584461429835

Expected return for SENSEX is: 0.12437255087912862
Variance for SENSEX : 0.21809497993913696

For 10 Stocks of BSE SENSEX
```

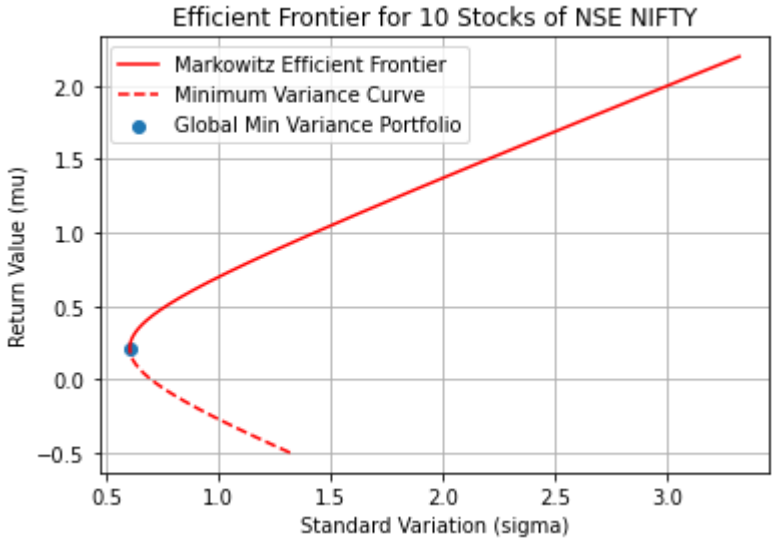



```
Risk Free return = 0.05
The Return on Market portfolio : 0.8766
The Risk on market portfolio : 1.7989 ( 179.8895 % )
Risk Free Rate : 0.05
```

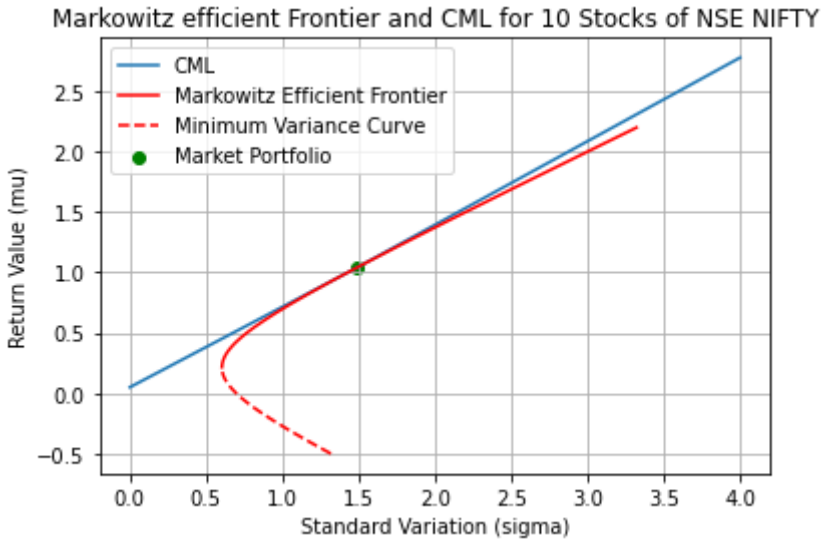


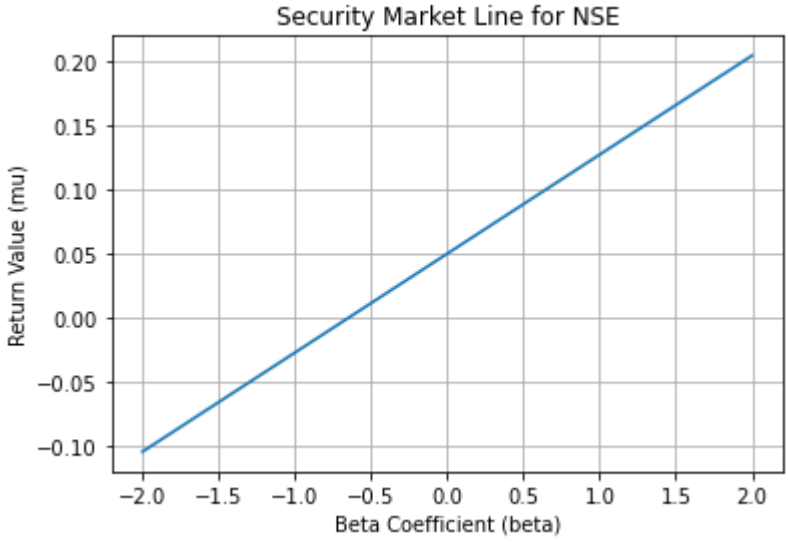


For 10 Stocks of NSE NIFTY

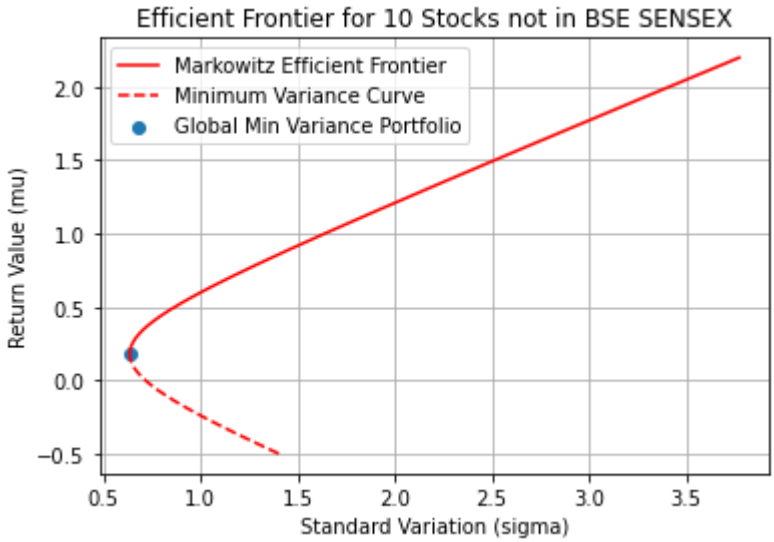


Risk Free return = 0.05
The Return on Market portfolio : 1.0399
The Risk on market portfolio : 1.4871 (148.7138 %)
Risk Free Rate : 0.05

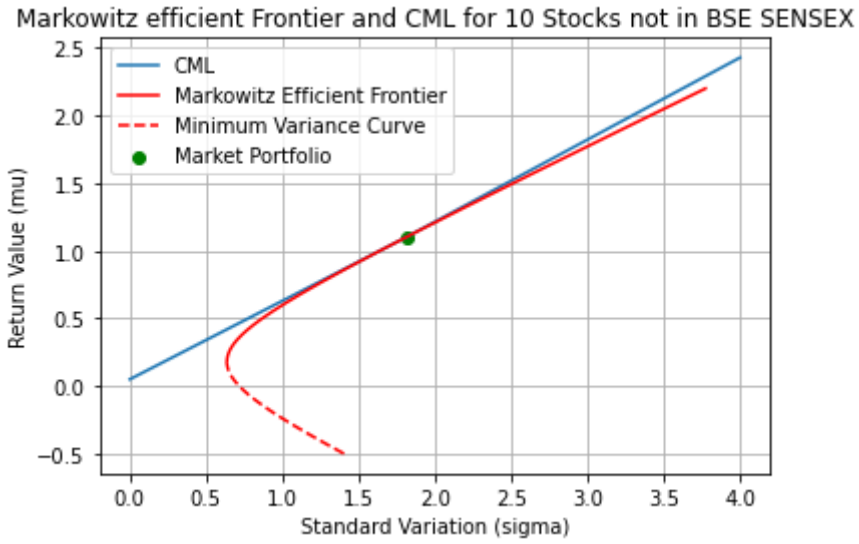




For 10 Stocks not included in BSE SENSEX



Risk Free return = 0.05
The Return on Market portfolio : 1.101
The Risk on market portfolio : 1.8116 (181.1558 %)
Risk Free Rate : 0.05



For 10 Stocks not included in NSE NIFTY