## Summer Training (5th semester) Report

**Subject Code - ART 355**

# Risk analysis of portfolio


**Ankit Tayal (IIOT) 04319011721**
**Anshika Gupta (AIDS) 11419011921**
**Aman Gupta (AIML) 05019011621**
**Hariansh Gaur (AIML) 09619011621**
**Priyanshu Jha (AIDS) 08719011921**

**Submitted To:**
**Dr. Arti Singh**

# CONTENTS

# Acknowledgement

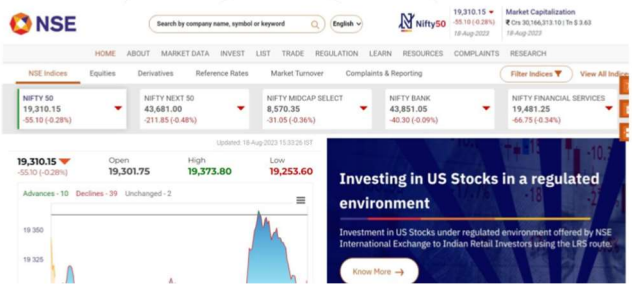
I would like to express my sincere gratitude to my professor ***Arti Singh***, classmates, and mentors for their invaluable guidance and support throughout the process of conducting the Risk Analysis of Portfolio. Your insights and encouragement have been instrumental in shaping this report.

Sincerely,

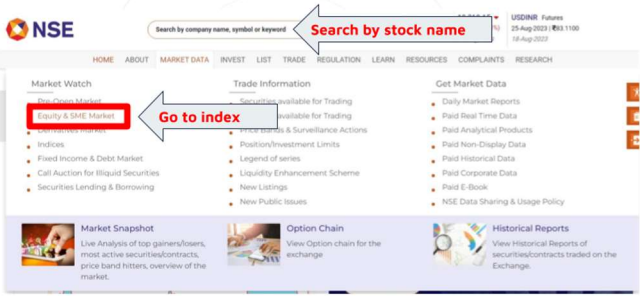
**Ankit Tayal (IIOT) 04319011721**
**Anshika Gupta (AIDS) 11419011921**
**Aman Gupta (AIML) 05019011621**
**Hariansh Gaur (AIML) 09619011621**
**Priyanshu Jha (AIDS) 08719011921**

# GRAPH

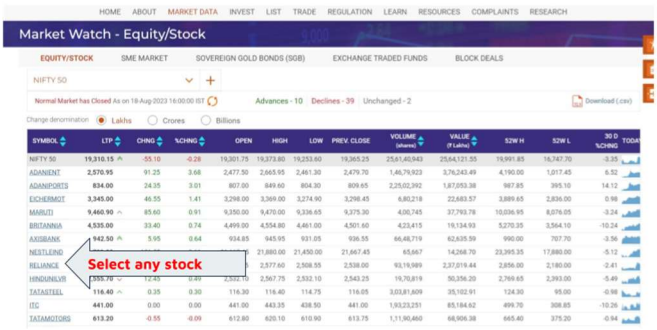
## NSE Website



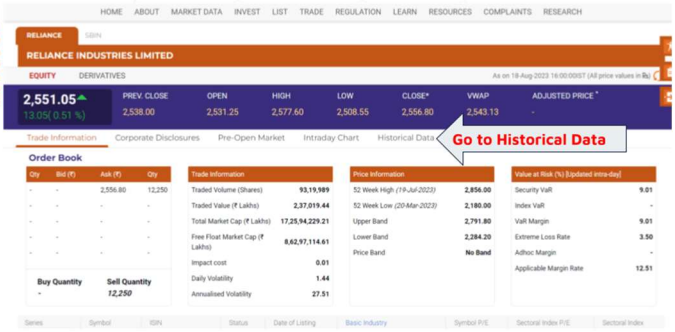

## Finding the stock



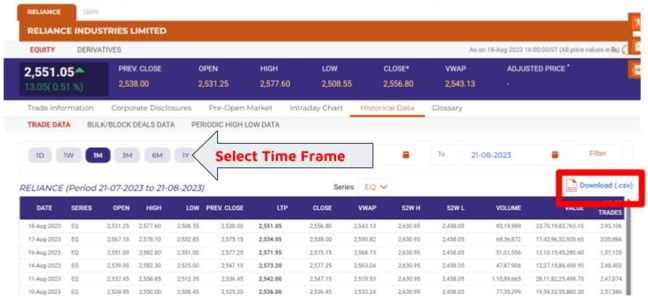

## NIFTY 50 LIST




## Stock Data

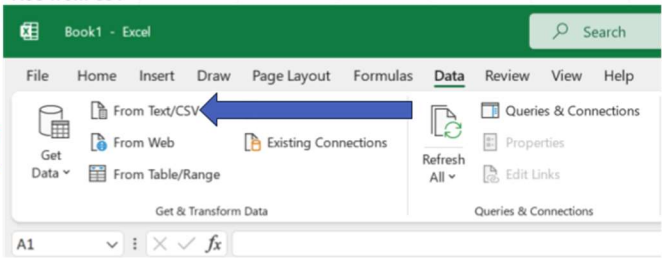



## Download CSV

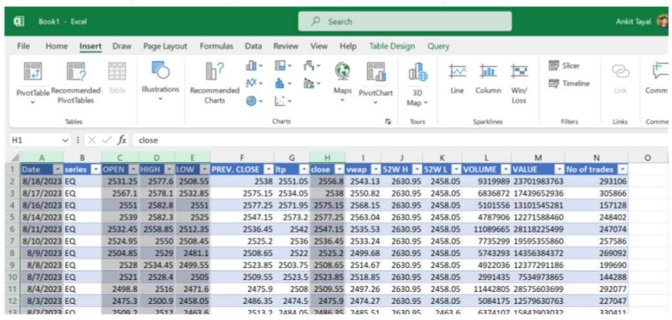



## Import CSV Data to excel

Create a new Workbook in ms excel
Go to data tab
Add from CSV



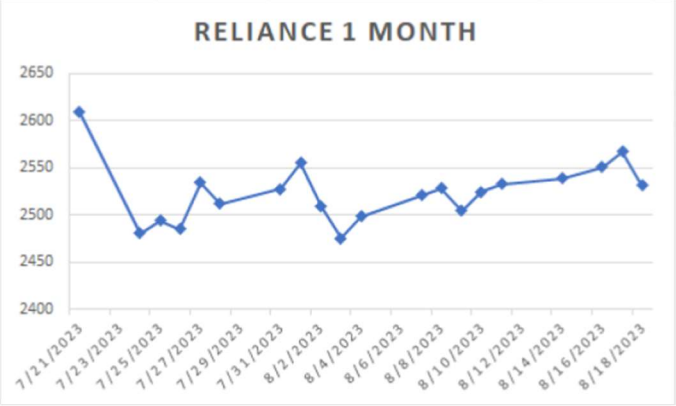

## Select Columns and Insert Chart



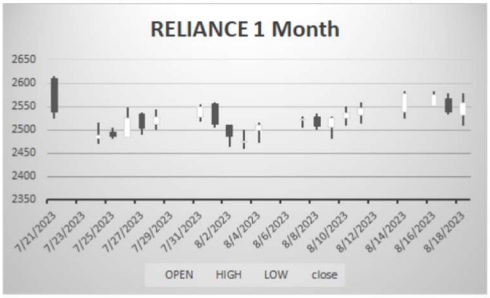
# CANDLESTICK

## Basic Line Chart

**Created using Date and Close price**

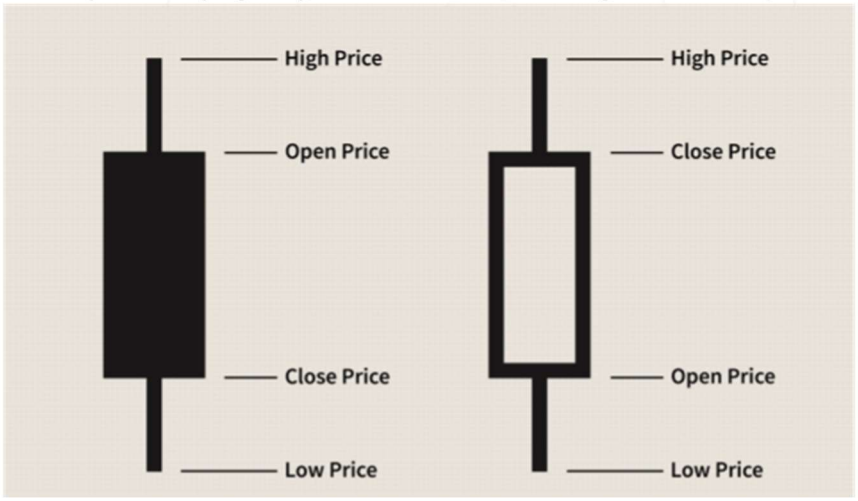



### Candlestick chart 1 month



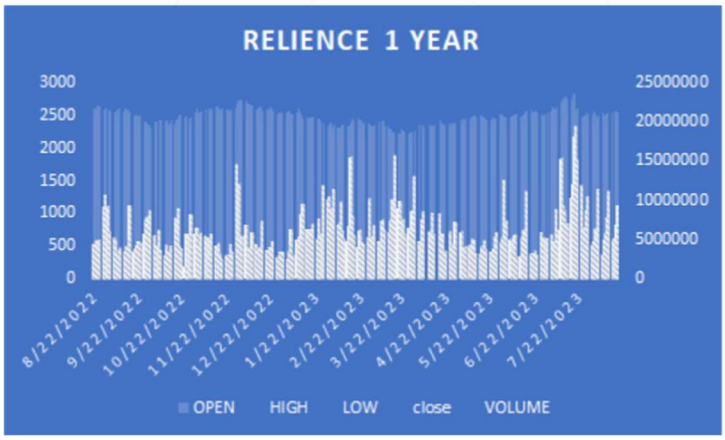

## Candlestick Components

When the real body is filled in or black (also red), it means the close was lower than the open. If the real body is white (or green), it means the close was higher than the open.



## Line chart of 1 year


## Volume in 1 Year




# PORTFOLIO OPTIMIZATION

## Definition of Portfolio Optimization

Portfolio optimization is the method of selecting the best portfolio, which gives back the most profitable rate of return for each unit of risk taken by the investors. A portfolio is the asset distribution or, in other words, the pool of investment options of an investor. The best portfolio for an investor depends upon various options like risk appetite, expected rate of return, other cost minimization, etc.

## How to Optimize a Portfolio

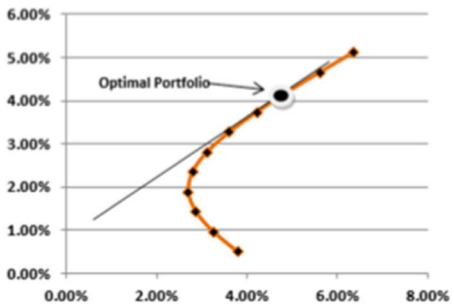
**Optimization of portfolio or selection of an optimized portfolio comprises a two-step process that differs according to the needs and requirements of each investor:**

**Step 1: Selection of Asset classes**: In the first step, portfolio managers choose asset classes for investing the client's or investor's funds. This selection depends upon the factors like the expected rate of return needed by the investor and the risk appetite of the investor. If the investor's need is a high rate of return and is ready to take high risk for that, then the portfolio manager can choose to invest in equities, options, futures, etc, and if the investor does not want to take much of a risk and is ready to accept average rate of return, then portfolio manager can invest the funds in bonds, gold, fixed interest securities, debentures, etc.

**Step 2: Selection of Assets within the class**: In the second step, the portfolio manager will choose how much bonds or equity will be included in the portfolio. What kinds of shares will form part of the equities in the portfolio, and how much money will be invested in real estate and fixed-interest securities? This choice is also based on the risk-return relationship chosen by the investor.

## Example of Portfolio Optimization

**There are two high-risk stocks (A and B) and one risk-free stock with a return of 1%. The efficient frontier is plotted based on the return of these stocks. Please refer to below the efficient frontier.**




Efficient Frontier (Stock A and B)

Now the X-axis represents the standard deviation of the stocks, and the y-axis depicts the returns of the stock. The risk-averse investor would move towards the left of the optimal portfolio tangential line, and investors wanting a high rate of return would move to the right towards the high-risk-return trade-off.

# Portfolio Optimization Methods

Below is a brief explanation of some of the main portfolio optimization methods:

**Mean-variance model**: Through this model, the portfolio manager is able to determine the optimal portfolio, which is the point at which the risk-return trade-off is highest in the graphical representation in the efficient frontier. It is based upon several assumptions relating to the market and investors.

**Mean semi-variance model**: Mean semi-variance model helps in measuring the potential downside risk of an optimized portfolio. The average standard deviation of values that falls below the mean is termed semi-variance.

**Conditional Value at Risk model (CVAR)**: CVAR model helps in measuring the tail risk associated with the investment portfolio. The technique is used for effective risk management in the portfolio optimization process.

**Mean absolute deviation model**: Mean absolute deviation model is used while selecting the large-scale optimized portfolio.

# Advantages

- Portfolio optimization helps in the maximization of the return on investment. This is done through the efficient frontier graph, which depicts the point where the risk-return trade-off for the portfolio is highest; that point gives back the optimal portfolio.
- Maximized return leads to increased satisfaction on investors' part.
- Portfolio optimization helps in the diversification of the portfolio. The portfolio manager chooses the diversified portfolio so that one underperforming asset does not impact the complete portfolio.
- For selecting the optimal portfolio, portfolio managers do a lot of market research which helps them in identifying market opportunities before other people, which ultimately benefits their clients or investors.

# Disadvantages

- Some assumptions taken by portfolio managers while choosing an optimal portfolio, like a frictionless market, are untrue in reality; there are frictions like transaction cost and other constraints that complicate the process of portfolio optimization.
- The assumptions can throw back incorrect results, which could result in the loss of money.
- There is always a risk of over-diversification of the assets, which would result in a marginal loss of expected return more than the marginal benefit of the reduced risk. This would ultimately result in erosion of the investor's benefit of the expected rate of return on the investment.

# What is the Portfolio Return Formula?

The portfolio return is the return obtained from the gain or loss realized by the investment portfolio which is a composite of several types of investments. Portfolios aim is to deliver a return on the basis of prespecified investment strategy to meet the investment objective, as well as the risk tolerance of the type of investors targeted by the portfolio.

**Portfolio Expected Return:**

Portfolio expected return is the sum of each product of individual asset's expected return with its associated weight.

$R_p = \sum (W_i * R_i)$

Where i = 1,2,3, . . . n

- **$W_i$:** Defines the associated weight to the asset i
- **$R_i$:** It is the asset's return

The weight attached to an asset = Market Value of an Asset / Market Value of Portfolio

# Portfolio Return Formula (contd.)

**Portfolio Variance:**

The variance of a Portfolio's return is a function of the individual assets and covariance between each of them. If we have two assets, A and B,

**Portfolio Variance = $W_A^2 * \sigma^2 * R_A + W_B^2 * \sigma^2 * R_B + 2 * W_A * W_B * Cov(R_A, R_B)$**

Portfolio variance is a measure of risk, more variance, more risk involve in it. Usually, an investor tries to reduce the risk by selecting negative covariance assets such as stocks and bonds.

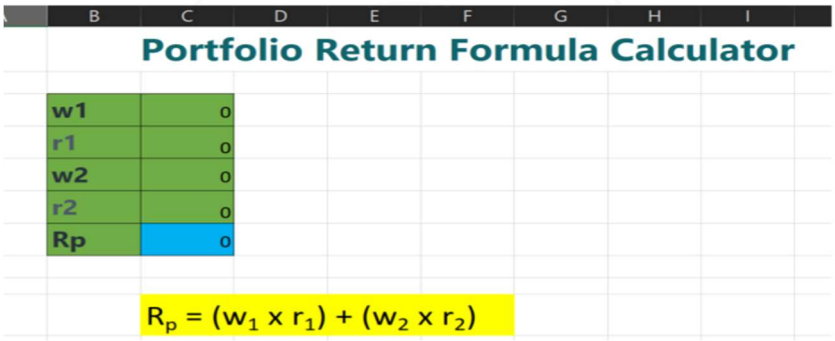
**Portfolio Standards Deviation:**

It is simply the square root of the portfolio variance.

**S.D = $\sqrt{[W_A^2 * \sigma^2 * R_A + W_B^2 * \sigma^2 * R_B + 2 * W_A * W_B * Cov(R_A, R_B)]}$**

And it is a measure of the riskiness of a portfolio.

## Portfolio Return Formula Calculator




# Portfolio Variance Formula

$$\text{Variance} = (w(1)^2 \times o(1)^2) + (w(2)^2 \times o(2)^2) + (2 \times (w(1) \times o(1) \times w(2) \times o(2) \times q(1,2)))$$

## Portfolio Variance Formula

Portfolio variance is a measure of dispersion of returns of a portfolio. It refers to the total returns of the portfolio over a particular period of time. The portfolio variance formula is used widely in the modern portfolio theory.

The portfolio variance formula is measured by the squaring the weights of the individual stocks in the portfolio and then multiplying it by the standard deviation of the individual assets in the portfolio and also squaring it. The numbers are then added by the covariance of the individual assets multiplied by two, also multiplied by the weights of each stock, also multiplying by a correlation between the different stocks present in the portfolio.

# Portfolio Variance Formula (contd.)

**Hence, the formula can be summarised as**

**Variance = (w(1)^2 * o(1)^2) + (w(2)^2 * o(2)^2) + (2 * (w(1)*o(1)*w(2)*o(2)*q(1,2)))**

Where the symbols stand for:-

- W (1): Weight of one stock in the portfolio squared.
- O (1): The standard deviation of one asset in the portfolio squared.
- W (2): Weight of second stock in the portfolio squared.
- O (2): The standard deviation of the second asset in the portfolio squared.
- Q(1,2): The correlation between the two assets in the portfolio has been denoted as q (1,2).

**Sample Standard Deviation Formula** $= \sqrt{\frac{\sum (X_i - X_m)^2}{(n-1)}}$

# Sample Standard Deviation

**In statistics, the standard deviation is a measure to find the dispersion of the data set values from the mean value of the data set. It measures the distance between that data point and the mean. So the higher the standard deviation, the higher the dispersion, and data points tend to be far from the mean**

The formula for sample standard deviation is given by:

**Sample Standard Deviation = $\sqrt{[\Sigma (X_i - X_m)^2 / (n - 1)]}$**

Where:

- $X_i$ – $i^{th}$ value of data set
- $X_m$ – Mean value of data set
- n – Total number of data points

# Portfolio Optimization Problem

**The Portfolio Optimization Problem is a critical financial challenge aiming to construct an investment portfolio that maximizes returns while managing risk, often utilizing mathematical models and algorithms.**

**r = portfolio return**
**V(o) = The current value of portfolio**

$$V(o) = x_1S_1(o) + x_2S_2(o) + \ldots\ldots\ldots + x_nS_n(o)$$

## Other Risk Measures:

**Mean absolute deviation** (MAD) is a statistical tool that measures the average absolute distance between each data value and the mean of a data set. It's similar to standard deviation, which also measures the spread, or variation, in data.

Steps to Solve

Apply the Mean Absolute Deviation Formula

$$MAD = \frac{\sum |x_i - \bar{x}|}{n}$$

Step One: Find the Mean

$$\bar{x} = \frac{\sum x_i}{n}$$

Step Two: Find the Absolute Value of the Difference $(\Delta)$ Between Each Number and the Mean

$$\Delta = |x_i - \bar{x}|$$

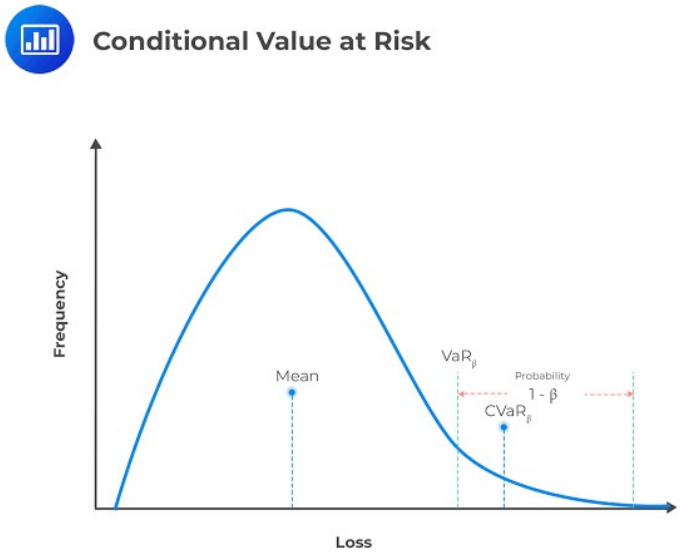
Step Three: Find the MAD

$$MAD = \frac{\text{sum of differences}}{n}$$

**Conditional Value at Risk (CVaR)** is a risk measure that quantifies the expected loss of an investment or portfolio in the event of extreme market conditions also known as:

- Expected Shortfall (ES)
- Tail Value at Risk (TVaR)

CVaR is an extended risk measure of value-at-risk (VaR). It quantifies the average loss over a specified time period of unlikely scenarios beyond the confidence level.
CVaR is calculated using historical data from similar trade.




# Modeling

The three key components of an optimization model are:
-The decision variables representing the actual decisions we are seeking.
-The constraints that specify the restrictions and interactions between the decision variables, thus defining the set of possible decisions.
-The objective function quantifies the criteria for choosing the best decision. The values of the decision variables that maximize or minimize the objective function is the "best" among the set of decision values defined by the constraints in the optimization model.

# Formulating the Optimization Problem

To maximize the return of your portfolio while considering the covariance matrix and the given investment amount of 1000 rupees, we can formulate this as an optimization

problem. The goal is to find the weights ( wR ),( wT ), and ( wL ) that maximize the expected return $\mu$ under the constraint that the weights sum up to 1.

**The objective function to maximize is the expected return of the portfolio:**

***Maximize μ = wR .Mean(R) + wT .Mean(T)+w_L.Mean(L)***

**Subject to the constraint:** ***wR + wT + wL = 1***

# Calculating the Covariance

To calculate the covariance matrix for the given data, you can follow these steps:

Step 1: Calculate the mean (average) returns for each stock over the given months.

Step 2: Calculate the deviation of each month's return from its mean for each stock.

Step 3: Multiply the deviations for each pair of stocks and take the average over the months to calculate the covariance between the stocks.

Step 4: Organize the covariance values into a matrix format.

# Sharpe Ratio

The Sharpe ratio is a measure of the risk-adjusted return of an investment or portfolio. The negative of the Sharpe ratio is used as an objective function to minimize, aiming to reduce the portfolio's risk (standard deviation) while maximizing the return.

$$Sharpe\ Ratio = \frac{R_p - R_f}{\sigma_p}$$

$R_p$ = Return of portfolio

$R_f$ = Risk-Free rate

$\sigma_p$ = Standard deviation of portfolio's excess return

# EXCEL

## SBIN 3 months

=SUM(R3:R67)

| | A | B | C | D | I | P | Q | R | S |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Quote-Equity-SBIN-EQ-22-05-2023-to-22-08-2023 | | | | | | | | |
| 2 | Count | Date | series | OPEN | close | Return/Da | R/D-M | SQ(R/D-M) | |
| 3 | 1 | 8/22/2023 | EQ | 572.5 | 568.5 | -0.0069869 | -0.005318579 | 0.000028287280164537000000 | |
| 4 | 2 | 8/21/2023 | EQ | 572.95 | 571.7 | -0.00218169 | -0.00051337 | 0.000000263549225842585000 | |
| 5 | 3 | 8/18/2023 | EQ | 570 | 572.95 | 0.005175439 | 0.006843759 | 0.000046837042544640200000 | |
| 6 | 4 | 8/17/2023 | EQ | 565.75 | 572.15 | 0.011312417 | 0.012980738 | 0.000168499557355515000000 | |
| 7 | 5 | 8/16/2023 | EQ | 558.8 | 565.35 | 0.011721546 | 0.013389867 | 0.000179288537224990000000 | |
| 8 | 6 | 8/14/2023 | EQ | 572.95 | 560.8 | -0.02120604 | -0.019537718 | 0.000381722429768232000000 | |
| 60 | 58 | 5/31/2023 | EQ | 583.8 | 579.85 | -0.00676602 | -0.005097695 | 0.000025986493991830600000 | |
| 61 | 59 | 5/30/2023 | EQ | 595 | 592.8 | -0.00369748 | -0.002029158 | 0.000004117483005787100000 | |
| 62 | 60 | 5/29/2023 | EQ | 590 | 595 | 0.008474576 | 0.010142897 | 0.000102878360800423000000 | |
| 63 | 61 | 5/26/2023 | EQ | 581.25 | 586 | 0.008172043 | 0.009840364 | 0.000096832759737565800000 | |
| 64 | 62 | 5/25/2023 | EQ | 582 | 581.25 | -0.00128866 | 0.000379661 | 0.000000144142472271614000 | |
| 65 | 63 | 5/24/2023 | EQ | 579.8 | 582.7 | 0.005001725 | 0.006670046 | 0.000044489507278784000000 | |
| 66 | 64 | 5/23/2023 | EQ | 578.9 | 581.6 | 0.004664018 | 0.006332339 | 0.000040098514113541000000 | |
| 67 | 65 | 5/22/2023 | EQ | 577.5 | 577.15 | -0.00060606 | 0.00106226 | 0.000001128396699073950000 | divide by n-1 |
| 68 | | | | | **-0.00166832** | **65** | **0.006818315861627370000000** | 0.000106536 | 0.010321637 |
| 69 | | | | | | | | | SD |
| 74 | | | | | w1+w2+w3=1 | | | | |
| 75 | | | | | w1,w2,w3>0 | | | | |
| 76 | | | | | sum(w1r1,w2r2,w3r3) | | | | |

## ITC 3 months

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 100 | Quote-Equity-ITC-EQ-22-05-2023-to-22-08-2023 | | | | | | | |
| 101 | count | Date | series | OPEN | close | Return/day | R/D-M | P2, sq |
| 102 | 1 | 8/22/2023 | EQ | 449 | 454.25 | 0.01169265 | 0.011846498 | 0.000140339511052209000000 |
| 103 | 2 | 8/21/2023 | EQ | 442 | 447.8 | 0.013122172 | 0.013276019 | 0.000176252692456503000000 |
| 104 | 3 | 8/18/2023 | EQ | 441 | 441.65 | 0.001473923 | 0.00162777 | 0.000002649636499649200000 |
| 105 | 4 | 8/17/2023 | EQ | 450 | 441 | -0.02 | -0.019846152 | 0.000393869768853148000000 |
| 106 | 5 | 8/16/2023 | EQ | 451 | 450.25 | -0.00166297 | -0.001509124 | 0.000002277454251730930000 |
| 107 | 6 | 8/14/2023 | EQ | 448.75 | 449.2 | 0.001002786 | 0.001156633 | 0.000001337799943791280000 |
| 162 | 61 | 5/26/2023 | EQ | 443.4 | 443.6 | 0.00045106 | 0.000604907 | 0.000000365913078741675000 |
| 163 | 62 | 5/25/2023 | EQ | 436.95 | 441.15 | 0.009612084 | 0.009765931 | 0.000095373413521254700000 |
| 164 | 63 | 5/24/2023 | EQ | 428.3 | 433.5 | 0.012141023 | 0.01229487 | 0.000151163832072204000000 |
| 165 | 64 | 5/23/2023 | EQ | 426 | 429.15 | 0.007394366 | 0.007548214 | 0.000056975530094450400000 |
| 166 | 65 | 5/22/2023 | EQ | 421.95 | 424.75 | 0.006635857 | 0.006789705 | 0.000046100091734322500000 divide by n-1 |
| 167 | | | | | -0.00015385 | | 0.008930606023026540000000 | 0.000139541 0.011812735 |
| 168 | | | | | | | | SD |

# RELIANCE 3 months

| | Date | series | count | OPEN | close | Return/day | R/D-M | P2, sq |
|---|---|---|---|---|---|---|---|---|
| 200 | Quote-Equity-RELIANCE-EQ-22-05-2023-to-22-08-2023 | | | | | | | |
| 201 | Date | series | count | OPEN | close | Return/day | R/D-M | P2, sq |
| 202 | 8/22/2023 | EQ | 1 | 2516.9 | 2519.4 | 0.000993285 | -0.000179776 | 0.000000032319423320372000 |
| 203 | 8/21/2023 | EQ | 2 | 2539.95 | 2520 | -0.00785449 | -0.009027547 | 0.000081496600356481800000 |
| 204 | 8/18/2023 | EQ | 3 | 2531.25 | 2556.8 | 0.010093827 | 0.008920766 | 0.000079580061266239100000 |
| 205 | 8/17/2023 | EQ | 4 | 2567.1 | 2538 | -0.01133575 | -0.01250881 | 0.000156470326047693000000 |
| 206 | 8/16/2023 | EQ | 5 | 2551 | 2575.15 | 0.009466876 | 0.008293814 | 0.000068787355770440200000 |
| 207 | 8/14/2023 | EQ | 6 | 2539 | 2577.25 | 0.015064986 | 0.013891925 | 0.000192985574307925000000 |
| 208 | 8/11/2023 | EQ | 7 | 2532.45 | 2547.15 | 0.005804656 | 0.004631594 | 0.000021451664308958700000 |
| 209 | 8/10/2023 | EQ | 8 | 2524.95 | 2536.45 | 0.004554546 | 0.003381484 | 0.000011434435443887300000 |
| 210 | 8/9/2023 | EQ | 9 | 2504.85 | 2525.2 | 0.008124239 | 0.006951178 | 0.000048318869318690200000 |
| 211 | 8/8/2023 | EQ | 10 | 2528 | 2508.65 | -0.00765427 | -0.008827334 | 0.000077921818118733200000 |
| 212 | 8/7/2023 | EQ | 11 | 2521 | 2523.85 | 0.001130504 | -4.25577E-05 | 0.000000001811154337028010 |
| 262 | 5/26/2023 | EQ | 61 | 2458 | 2506.5 | 0.019731489 | 0.018558428 | 0.000344415234544370000000 |
| 263 | 5/25/2023 | EQ | 62 | 2433.9 | 2439.95 | 0.002485723 | 0.001312661 | 0.000001723079099559530000 |
| 264 | 5/24/2023 | EQ | 63 | 2445.9 | 2440.25 | -0.00230999 | -0.00348305 | 0.000012131634312230900000 |
| 265 | 5/23/2023 | EQ | 64 | 2460.65 | 2454.55 | -0.00247902 | -0.003652081 | 0.000013337697080544800000 |
| 266 | 5/22/2023 | EQ | 65 | 2435 | 2455 | 0.008213552 | 0.007040491 | 0.000049568512592865400000 |
| 267 | | | | | | 0.001173061 | | 0.006362622897550820000000 |
| 268 | | | | | | | | |
| 269 | | | | | | | | |
| 270 | | | | | | | | |

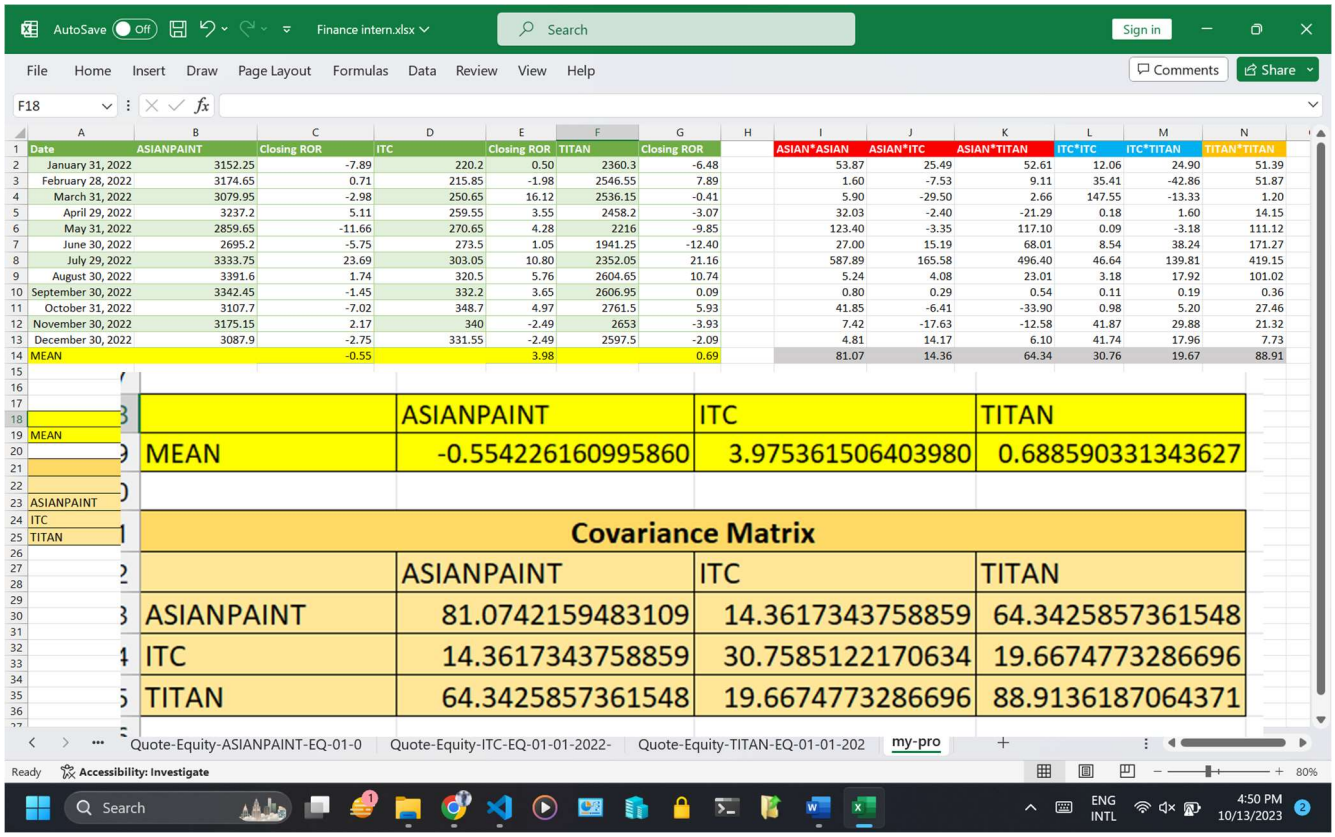
divide by n-1 | 0.000099415983 | 0.009970756 | SD

# SBIN One YEAR

J255 | Return Per Day is calculated as:

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Date | series | OPEN | HIGH | LOW | PREV. CLOSE | ltp | close | vwap | 52W H | 52W L | VOLUME | VALUE | No of trades | R/D (c-O/O | R/D-M | SQ(R/D-M) | | | |
| 2 | 9/22/2023 | EQ | 593.5 | 602 | 591.4 | 588.15 | 598.7 | 598.15 | 598.12 | 629.55 | 499.35 | 18784139 | 1.1235E+10 | 295166 | 0.007834878 | 0.009447792 | 8.92608E-05 | | | |
| 3 | 9/21/2023 | EQ | 600 | 606.6 | 584.65 | 601.1 | 587.9 | 588.15 | 593.29 | 629.55 | 499.35 | 23453425 | 1.3915E+10 | 425490 | -0.01975 | -0.01813709 | 0.000328954 | | | |
| 4 | 9/20/2023 | EQ | 600.85 | 606.8 | 599.4 | 603.95 | 601 | 601.1 | 602.78 | 629.55 | 499.35 | 21314600 | 1.2848E+10 | 351772 | 0.000416077 | 0.002028992 | 4.11681E-06 | | | |
| 5 | 9/18/2023 | EQ | 598.8 | 608.4 | 597.05 | 598.8 | 603 | 603.95 | 605.18 | 629.55 | 499.35 | 19305055 | 1.1683E+10 | 336027 | 0.008600534 | 0.010213449 | 0.000104315 | | | |
| 6 | 9/15/2023 | EQ | 599.35 | 600.7 | 595.2 | 597.3 | 598.1 | 598.8 | 598.23 | 629.55 | 499.35 | 11756389 | 7033062065 | 182595 | -0.000917661 | 0.000695254 | 4.83378E-07 | | | |
| 7 | 9/14/2023 | EQ | 598.45 | 603.7 | 595.2 | 596.45 | 596.6 | 597.3 | 598.77 | 629.55 | 499.35 | 13178911 | 7891138918 | 260528 | -0.001921631 | -0.00030872 | 9.53059E-08 | | | |
| 8 | 9/13/2023 | EQ | 588.4 | 598 | 583.45 | 588.35 | 596.4 | 596.45 | 592.24 | 629.55 | 499.35 | 19377172 | 1.1476E+10 | 279643 | 0.013681169 | 0.015294084 | 0.000233909 | | | |
| 9 | 9/12/2023 | EQ | 593.65 | 594.25 | 584.65 | 591.7 | 587.6 | 588.35 | 588.44 | 629.55 | 499.35 | 16093060 | 9469853288 | 254109 | -0.008927819 | -0.00731491 | 5.35078E-05 | | | |
| 10 | 9/11/2023 | EQ | 587 | 592.3 | 585.35 | 583.45 | 591.3 | 591.7 | 590.12 | 629.55 | 499.35 | 10946134 | 6459550843 | 271751 | 0.008006814 | 0.009619729 | 9.25392E-05 | | | |
| 11 | 9/8/2023 | EQ | 581 | 588.5 | 579.7 | 579.05 | 583.4 | 583.45 | 584.57 | 629.55 | 499.35 | 14032068 | 8202779360 | 205241 | 0.004216867 | 0.005829782 | 3.39864E-05 | | | |
| 12 | 9/7/2023 | EQ | 568 | 581 | 568 | 570.5 | 580.6 | 579.05 | 575.6 | 629.55 | 499.35 | 12390597 | 7132078108 | 180550 | 0.019454225 | 0.02106714 | 0.000443824 | | | |
| 13 | 9/6/2023 | EQ | 574.25 | 574.25 | 568 | 573 | 571 | 570.5 | 570.55 | 629.55 | 499.35 | 20609936 | 1.1759E+10 | 337013 | -0.006530257 | -0.00491734 | 2.41803E-05 | | | |
| 14 | 9/5/2023 | EQ | 575.6 | 578.25 | 571.35 | 575.55 | 573 | 573 | 574.05 | 629.55 | 499.35 | 14444729 | 8292019922 | 247672 | -0.004517026 | -0.00290411 | 8.43386E-06 | | | |
| 15 | 9/4/2023 | EQ | 572 | 576.35 | 568.45 | 569.55 | 575.6 | 575.55 | 572.87 | 629.55 | 499.35 | 12108655 | 6936698350 | 195524 | 0.006206294 | 0.007819208 | 6.114E-05 | | | |
| 16 | 9/1/2023 | EQ | 562.7 | 571.2 | 562.1 | 561.35 | 569.5 | 569.55 | 567.55 | 629.55 | 499.35 | 18121168 | 1.0285E+10 | 260408 | 0.012173449 | 0.013786364 | 0.000190064 | | | |
| 17 | 8/31/2023 | EQ | 569.3 | 570.3 | 559.4 | 567.4 | 562.2 | 561.35 | 564.43 | 629.55 | 499.35 | 27898476 | 1.5747E+10 | 326854 | -0.013964518 | -0.0123516 | 0.000152562 | | | |
| 18 | 8/30/2023 | EQ | 576.7 | 577.7 | 566.1 | 574.85 | 567.5 | 567.4 | 571.91 | 629.55 | 499.35 | 15281416 | 8739616796 | 245036 | -0.016126235 | -0.01451332 | 0.000210636 | | | |
| 19 | 8/29/2023 | EQ | 573.65 | 576.3 | 571.85 | 572.85 | 575.3 | 574.85 | 574.01 | 629.55 | 499.35 | 13144855 | 7545323890 | 232951 | 0.002091868 | 0.003704782 | 1.37254E-05 | | | |
| 242 | 10/4/2022 | EQ | 533 | 535.8 | 527.8 | 519.75 | 533 | 532.7 | 532.39 | 578.5 | 425 | 8745887 | 4656244525 | 128142 | -0.000562852 | 0.001050063 | 1.10263E-06 | | | |
| 243 | 10/3/2022 | EQ | 527.6 | 531.65 | 518.5 | 530.6 | 519 | 519.75 | 525.01 | 578.5 | 425 | 9351162 | 4909456239 | 141130 | -0.014878696 | -0.01326578 | 0.000175981 | | | |
| 244 | 9/30/2022 | EQ | 524 | 536.9 | 516.15 | 521.75 | 531.6 | 530.6 | 528.17 | 578.5 | 425 | 17466950 | 9225477677 | 199404 | 0.01259542 | 0.014208334 | 0.000201877 | | | |
| 245 | 9/29/2022 | EQ | 529 | 535.7 | 519.55 | 524.85 | 524 | 521.75 | 525.18 | 578.5 | 425 | 17310168 | 9090986646 | 215284 | -0.013705104 | -0.01209219 | 0.000146221 | | | |
| 246 | 9/28/2022 | EQ | 532.9 | 539.8 | 520.35 | 536.5 | 523.5 | 524.85 | 532.17 | 578.5 | 425 | 12233485 | 6510284800 | 183919 | -0.015106024 | -0.01349311 | 0.000182064 | | | |
| 247 | 9/27/2022 | EQ | 544.55 | 550.9 | 533.2 | 543.3 | 538 | 536.5 | 542.4 | 578.5 | 425 | 11066799 | 6002626270 | 154979 | -0.014782848 | -0.01316993 | 0.000173447 | | | |
| 248 | 9/26/2022 | EQ | 545.8 | 549 | 533 | 550.6 | 545.3 | 543.3 | 541.37 | 578.5 | 425 | 16158750 | 8747941723 | 274214 | -0.004580432 | -0.00296752 | 8.80616E-06 | | | |
| 249 | Total | | | | | | | | | | | | | | | -0.001612914 | 247 | 0.041756415 | | |
| 250 | | | | | | | | | | | | | | | | | S.D. | 0.013028489 | | |
| 251 | | | | | | | | | | | | | | | | | | | | |
| 252 | | | | | | | | | | | | | | | | | | | | |

| | |
|---|---|
| Return Per Day is calculated as: | (CLOSE - OPEN ) / OPEN |
| **Mean is calculated as:** | **AVERAGE of Return Per Day column** |
| that is =AVERAGE(O2:O248) | -0.0016129 |
| **A - Am is calculated as:** | **Return Per Day - Mean** |
| that is [(CLOSE-OPEN )/OPEN] - AVERAGE(O2:O248) | |
| **$(A - A_m)^2$ is calculated as:** | SQUARE of Return Per Day - Mean |
| that is {[(CLOSE-OPEN )/OPEN] - AVERAGE(O2:O248)} ^2 | |
| **Sum of $(A - A_m)^2$ is calculated as: =SUM(Q2:Q248)** | 0.0417564 |
| N-1 is calculated as: that is =COUNT(Q2:Q248) -1 | 246 |
| SUM / n-1 is calculated as: =SUM(Q2:Q248)/246 | 0.0001697 |
| Sample Standard Deviation is calculated using the formula below | |
| **Sample Standard Deviation = $\sqrt{[\Sigma (X_i - X_m)^2 / (n - 1)]}$** | |
| | that is =SQRT(SUM/n-1) |
| **Sample Standard Deviation** | 0.013028489 |



Finance intern.xlsx

| Date | ASIANPAINT | Closing ROR | ITC | Closing ROR | TITAN | Closing ROR | ASIAN*ASIAN | ASIAN*ITC | ASIAN*TITAN | ITC*ITC | ITC*TITAN | TITAN*TITAN |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| January 31, 2022 | 3152.25 | -7.89 | 220.2 | 0.50 | 2360.3 | -6.48 | 53.87 | 25.49 | 52.61 | 12.06 | 24.90 | 51.39 |
| February 28, 2022 | 3174.65 | 0.71 | 215.85 | -1.98 | 2546.55 | 7.89 | 1.60 | -7.53 | 9.11 | 35.41 | -42.86 | 51.87 |
| March 31, 2022 | 3079.95 | -2.98 | 250.65 | 16.12 | 2536.15 | -0.41 | 5.90 | -29.50 | 2.66 | 147.55 | -13.33 | 1.20 |
| April 29, 2022 | 3237.2 | 5.11 | 259.55 | 3.55 | 2458.2 | -3.07 | 32.03 | -2.40 | -21.29 | 0.18 | 1.60 | 14.15 |
| May 31, 2022 | 2859.65 | -11.66 | 270.65 | 4.28 | 2216 | -9.85 | 123.40 | -3.35 | 117.10 | 0.09 | -3.18 | 111.12 |
| June 30, 2022 | 2695.2 | -5.75 | 273.5 | 1.05 | 1941.25 | -12.40 | 27.00 | 15.19 | 68.01 | 8.54 | 38.24 | 171.27 |
| July 29, 2022 | 3333.75 | 23.69 | 303.05 | 10.80 | 2352.05 | 21.16 | 587.89 | 165.58 | 496.40 | 46.64 | 139.81 | 419.15 |
| August 30, 2022 | 3391.6 | 1.74 | 320.5 | 5.76 | 2604.65 | 10.74 | 5.24 | 4.08 | 23.01 | 3.18 | 17.92 | 101.02 |
| September 30, 2022 | 3342.45 | -1.45 | 332.2 | 3.65 | 2606.95 | 0.09 | 0.80 | 0.29 | 0.54 | 0.11 | 0.19 | 0.36 |
| October 31, 2022 | 3107.7 | -7.02 | 348.7 | 4.97 | 2761.5 | 5.93 | 41.85 | -6.41 | -33.90 | 0.98 | 5.20 | 27.46 |
| November 30, 2022 | 3175.15 | 2.17 | 340 | -2.49 | 2653 | -3.93 | 7.42 | -17.63 | -12.58 | 41.87 | 29.88 | 21.32 |
| December 30, 2022 | 3087.9 | -2.75 | 331.55 | -2.49 | 2597.5 | -2.09 | 4.81 | 14.17 | 6.10 | 41.74 | 17.96 | 7.73 |
| MEAN | | -0.55 | | 3.98 | | 0.69 | 81.07 | 14.36 | 64.34 | 30.76 | 19.67 | 88.91 |

| | ASIANPAINT | ITC | TITAN |
|---|---|---|---|
| MEAN | -0.554226160995860 | 3.975361506403980 | 0.688590331343627 |

**Covariance Matrix**

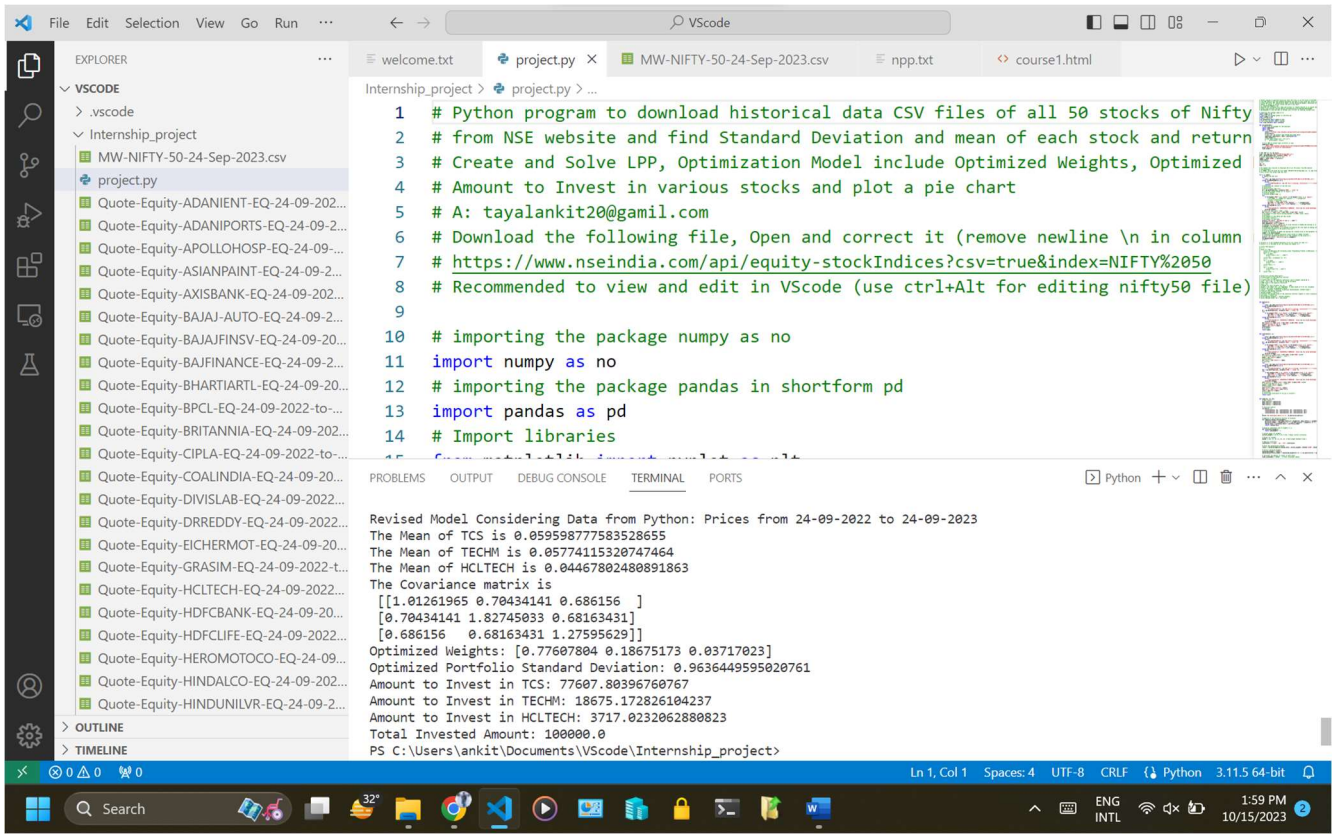
| | ASIANPAINT | ITC | TITAN |
|---|---|---|---|
| ASIANPAINT | 81.0742159483109 | 14.3617343758859 | 64.3425857361548 |
| ITC | 14.3617343758859 | 30.7585122170634 | 19.6674773286696 |
| TITAN | 64.3425857361548 | 19.6674773286696 | 88.9136187064371 |

Quote-Equity-ASIANPAINT-EQ-01-0 | Quote-Equity-ITC-EQ-01-01-2022- | Quote-Equity-TITAN-EQ-01-01-202 | my-pro

Ready | Accessibility: Investigate | 80%

4:50 PM 10/13/2023

# CODE




# CODE:

```
# Python program to download historical data CSV files of all 50 stocks of Nifty50 constituents of any timeframe
# from NSE website and find Standard Deviation and mean of each stock and return list
# Create and Solve LPP, Optimization Model include Optimized Weights, Optimized Portfolio Standard Deviation
# Amount to Invest in various stocks and plot a pie chart
# A: tayalankit20@gamil.com
# Download the following file, Open and correct it (remove newline \n in column name)
# https://www.nseindia.com/api/equity-stockIndices?csv=true&index=NIFTY%2050
# Recommended to view and edit in VScode (use ctrl+Alt for editing nifty50 file)

# importing the package numpy as no
import numpy as no
# importing the package pandas in shortform pd
import pandas as pd
# Import libraries
from matplotlib import pyplot as plt
# from scipy.optimize import linprog
from scipy.optimize import minimize

def nse_download():
    # importing the package for CSV download
    import webbrowser
    import time
    for i in symbol:
        nseurl = f'https://www.nseindia.com/api/historical/cm/equity?symbol={i}&series=[%22EQ%22]&from=24-09-2022&to=24-09-2023&csv=true'
        print(nseurl)
        # then call the default open method described above
        webbrowser.open(nseurl, new=2, autoraise=True)
        time.sleep(0.5)
        # with open("output.txt", "a") as f:
        #     print(nseurl, file=f)

    # Error M&M url dosent open correctly in loop
    webbrowser.open(
```

```
        'https://www.nseindia.com/api/historical/cm/equity?symbol=M%26M&series=[%22EQ%22]&from=24-09-2022&to=24-09-2023&csv=true', new=2,
autoraise=True)
    print("Waiting to finish all downloading")
    time.sleep(5)



# FOR LOOP for all 50 Stocks
csv0 = pd.read_csv('MW-NIFTY-50-24-Sep-2023.csv')
df0 = pd.DataFrame(csv0, columns=['SYMBOL '])
# Convert DataFrame column as a list
symbol = (df0['SYMBOL '].tolist())
symbol.pop(0)
# print(symbol)
n = 0
sd = []
mean = []

# Un-comment this function to download CSV of all 50 stocks from NSE website
# nse_download()
# if ERROR: move the Nifty 50 file named 'MW-NIFTY-50-24-Sep-2023.csv' to same folder as this file
# Also correct the column name new line

for i in symbol:
    # reading the CSV file
    try:
        csv1 = pd.read_csv(f'Quote-Equity-{i}-EQ-24-09-2022-to-24-09-2023.csv')
    except FileNotFoundError as e:
        print(
            f"FileNotFoundError: {i} CSV file is missing. **********---------********")
        continue
    # displaying the contents of the CSV file
    # print(csv1)
    # print(f'sucess open {i}')
    df = pd.DataFrame(csv1, columns=['OPEN ', 'close '])
    # print("the output datype is: of stock-", i)
    # print(df.dtypes)
    # print(df.dtypes['OPEN '])
    try:
        if df.dtypes['OPEN '] == 'object' or df.dtypes['close '] == 'object':
            # df['OPEN '] = dfOPEN str.replace(',', '').astype(int)
            # print("converting")
            # remove , comma from values
            df['OPEN '] = df['OPEN '].str.replace(',', '').astype(float)
            df['close '] = df['close '].str.replace(',', '').astype(float)
    except AttributeError as e:
        print(
            f'AttributeError: SUCESSFULLY HANDELED - Stock {i} has mixed datatypes in same column')
        print(df.dtypes)
    # return per day close - open / close
    df['result'] = ((df['close ']-df['OPEN '])/df['OPEN '])*100
    # Un-comment to show entire working dataset i.e OPEN, close, result
    # print(df)
    # Un-comment to show Retun per day column
    # print(df['result'])
    # Un-comment to show mean
    print(f'S.No {n+1} - The Mean of {i} is ', end="")
    print(df['result'].mean())
    mean.append(df['result'].mean())
    # Creating an array by making use of array function in NumPy and storing it in a variable called arrayname
    arrayname = no.array(df['result'])
    # Displaying the elements of arrayname followed by one line space by making use of \n
    # print('The elements of the given array are:')
    # print(arrayname)
    # using std function of NumPy and passing the created array as the parameter to that function to find the standard deviation value of all the
elements in the array and store it in a variable called stddev
    stddev = no.std(arrayname)
    # Displaying the standard deviation value stored in stddev variable
    # print('The standard deviation of all the elements of the array is:')
    print(f'S.No {n+1} - The S.D of {i} is {stddev}')
    sd.append(stddev)
    n += 1

# print(f'\n \n The standard deviation of all {n} stocks are {sd} \n')
# print(f'\n \n The mean of all {n} stocks are {mean} \n')

def meancal(i):
    try:
        csv1 = pd.read_csv(f'Quote-Equity-{i}-EQ-24-09-2022-to-24-09-2023.csv')
    except FileNotFoundError as e:
        print(
            f"FileNotFoundError: {i} CSV file is missing. **********---------********")
    df = pd.DataFrame(csv1, columns=['OPEN ', 'close '])
    try:
```

```
        if df.dtypes['OPEN '] == 'object' or df.dtypes['close '] == 'object':
            df['OPEN '] = df['OPEN '].str.replace(',', '').astype(float)
            df['close '] = df['close '].str.replace(',', '').astype(float)
    except AttributeError as e:
        print(
            f'AttributeError: SUCESSFULLY HANDELED - Stock {i} has mixed datatypes in same column')
        print(df.dtypes)
    df['result'] = ((df['close ']-df['OPEN '])/df['OPEN '])*100
    print(f'The Mean of {i} is ', end="")
    meanc = df['result'].mean()
    print(meanc)
    # print(df)
    return meanc


def covariance(i, j):
    try:
        csv1 = pd.read_csv(f'Quote-Equity-{i}-EQ-24-09-2022-to-24-09-2023.csv')
    except FileNotFoundError as e:
        print(
            f"FileNotFoundError: {i} CSV file is missing. **********---------********")
    df = pd.DataFrame(csv1, columns=['OPEN ', 'close '])
    try:
        if df.dtypes['OPEN '] == 'object' or df.dtypes['close '] == 'object':
            df['OPEN '] = df['OPEN '].str.replace(',', '').astype(float)
            df['close '] = df['close '].str.replace(',', '').astype(float)
    except AttributeError as e:
        print(
            f'AttributeError: SUCESSFULLY HANDELED - Stock {i} has mixed datatypes in same column')
        print(df.dtypes)
    df['result'] = ((df['close ']-df['OPEN '])/df['OPEN '])*100
    # print(f'The Mean of {i} is ', end="")
    meanc = df['result'].mean()
    # print(meanc)
    df['rorm'] = df['result'] - meanc
    # print(df)
    try:
        csv2 = pd.read_csv(f'Quote-Equity-{j}-EQ-24-09-2022-to-24-09-2023.csv')
    except FileNotFoundError as e:
        print(
            f"FileNotFoundError: {j} CSV file is missing. **********---------********")
    df1 = pd.DataFrame(csv2, columns=['OPEN ', 'close '])
    try:
        if df1.dtypes['OPEN '] == 'object' or df1.dtypes['close '] == 'object':
            df1['OPEN '] = df1['OPEN '].str.replace(',', '').astype(float)
            df1['close '] = df1['close '].str.replace(',', '').astype(float)
    except AttributeError as e:
        print(
            f'AttributeError: SUCESSFULLY HANDELED - Stock {j} has mixed datatypes in same column')
        print(df1.dtypes)
    df1['result'] = ((df1['close ']-df1['OPEN '])/df1['OPEN '])*100
    # print(f'The Mean of {j} is ', end="")
    meancc = df1['result'].mean()
    # print(meancc)
    df1['rorm'] = df1['result'] - meancc
    df1['mul'] = df['rorm'] * df1['rorm']
    covvf = df1['mul'].mean()
    # print(df1)
    # print(f'The covariance of {i},{j} is {covvf}')
    return covvf


def model(n1, n2, n3):
    # Mean returns
    mean_return_R = meancal(n1)
    mean_return_T = meancal(n2)
    mean_return_L = meancal(n3)

    # Covariance matrix
    cov_matrix = [
        [covariance(n1, n1), covariance(n1, n2), covariance(n1, n3)],
        [covariance(n1, n2), covariance(n2, n2), covariance(n2, n3)],
        [covariance(n1, n3), covariance(n2, n3), covariance(n3, n3)]
    ]
    print('The Covariance matrix is \n', no.matrix(cov_matrix))

    # Negative of the objective function to minimize
    def negative_sharpe_ratio(weights):
        portfolio_return = sum([mean_return_R * weights[0], mean_return_T * weights[1], mean_return_L * weights[2]])
        portfolio_stddev = (weights[0] ** 2 * cov_matrix[0][0] + weights[1] ** 2 * cov_matrix[1][1] + weights[2] ** 2 * cov_matrix[2][2] + 2 *
weights[0] * weights[1] * cov_matrix[0][1] + 2 * weights[0] * weights[2] * cov_matrix[0][2] + 2 * weights[1] * weights[2] * cov_matrix[1][2]) **
0.5
```

```
        sharpe_ratio = portfolio_return / portfolio_stddev
        return -sharpe_ratio

    # Equality constraint (sum of weights == 1)
    def constraint(weights):
        return sum(weights) - 1

    # Initial guess for weights
    initial_weights = [0.33, 0.33, 0.34]  # Equal initial allocation

    # Bounds for weights
    bounds = ((0, 1), (0, 1), (0, 1))  # Each weight between 0 and 1

    # Equality constraint
    constraints = {'type': 'eq', 'fun': constraint}

    # Solve the optimization problem
    result = minimize(negative_sharpe_ratio, initial_weights, method='SLSQP', bounds=bounds, constraints=constraints)

    # Extract optimized weights
    optimized_weights = result.x
    optimized_portfolio_stddev = (optimized_weights[0] ** 2 * cov_matrix[0][0] + optimized_weights[1] ** 2 * cov_matrix[1][1] +
optimized_weights[2] ** 2 * cov_matrix[2][2] + 2 * optimized_weights[0] * optimized_weights[2] * cov_matrix[0][2] + 2 * optimized_weights[0] *
optimized_weights[1] * cov_matrix[0][1] + 2 * optimized_weights[1] * optimized_weights[2] * cov_matrix[1][2]) ** 0.5

    # Calculate the amounts to invest in each stock
    total_investment = 100000    # Total investment amount
    amount_x = optimized_weights[0] * total_investment
    amount_y = optimized_weights[1] * total_investment
    amount_z = optimized_weights[2] * total_investment

    print("Optimized Weights:", optimized_weights)
    print("Optimized Portfolio Standard Deviation:", optimized_portfolio_stddev)
    print(f"Amount to Invest in {n1}:", amount_x)
    print(f"Amount to Invest in {n2}:", amount_y)
    print(f"Amount to Invest in {n3}:", amount_z)
    print("Total Invested Amount:", amount_x + amount_y + amount_z)

    # Creating dataset
    stocks = [n1, n2, n3]
    data = [amount_x, amount_y , amount_z]

    # Creating explode data
    explode = (0.2, 0.1, 0.0)
    # Creating color parameters
    colors = ( "orange", "cyan", "brown", "grey", "indigo", "beige")
    # Wedge properties
    wp = { 'linewidth' : 1, 'edgecolor' : "green" }
    # Creating autocpt arguments
    def func(pct, allvalues):
        absolute = int(pct / 100.*no.sum(allvalues))
        return "{:.1f}%\n(₹{:d})".format(pct, absolute)

    # Creating plot
    fig, ax = plt.subplots(figsize =(10, 7))
    wedges, texts, autotexts = ax.pie(data, autopct = lambda pct: func(pct, data), explode = explode, labels = stocks, shadow = True, colors =
colors, startangle = 90, wedgeprops = wp, textprops = dict(color ="magenta"))
    # Adding legend
    ax.legend(wedges, stocks, title ="Stocks", loc ="center left", bbox_to_anchor =(1, 0, 0.5, 1))
    plt.setp(autotexts, size = 8, weight ="bold")
    ax.set_title(f"Optimized Portfolio \nTotal Invested Amount ₹{amount_x + amount_y + amount_z}")
    # show plot
    plt.show()

print("\nInitial Model Considering Data from Excel: Prices from 1st January 2022 to 30 December 2022")

# Mean returns
mean_return_R = -0.55422616099586
mean_return_T = 3.97536150640398
mean_return_L = 0.688590331343627

# Covariance matrix
cov_matrix = [
    [81.0742159483109, 14.3617343758859, 64.3425857361548],
    [14.3617343758859, 30.7585122170634, 19.6674773286696],
    [64.3425857361548, 19.6674773286696, 88.9136187064371]
]

# Negative of the objective function to minimize
def negative_sharpe_ratio(weights):
    portfolio_return = sum([mean_return_R * weights[0], mean_return_T * weights[1], mean_return_L * weights[2]])
    portfolio_stddev = (weights[0] ** 2 * cov_matrix[0][0] + weights[1] ** 2 * cov_matrix[1][1] + weights[2] ** 2 * cov_matrix[2][2] + 2 *
weights[0] * weights[1] * cov_matrix[0][1] + 2 * weights[0] * weights[2] * cov_matrix[0][2] + 2 * weights[1] * weights[2] * cov_matrix[1][2]) **
0.5
```

# OUTPUT

```
    sharpe_ratio = portfolio_return / portfolio_stddev
    return -sharpe_ratio

# Equality constraint (sum of weights == 1)
def constraint(weights):
    return sum(weights) - 1

# Initial guess for weights
initial_weights = [0.33, 0.33, 0.34]  # Equal initial allocation

# Bounds for weights
bounds = ((0, 1), (0, 1), (0, 1))  # Each weight between 0 and 1

# Equality constraint
constraints = {'type': 'eq', 'fun': constraint}

# Solve the optimization problem
result = minimize(negative_sharpe_ratio, initial_weights, method='SLSQP', bounds=bounds, constraints=constraints)

# Extract optimized weights
optimized_weights = result.x
optimized_portfolio_stddev = (optimized_weights[0] ** 2 * cov_matrix[0][0] + optimized_weights[1] ** 2 * cov_matrix[1][1] + optimized_weights[2] **
2 * cov_matrix[2][2] + 2 * optimized_weights[0] * optimized_weights[2] * cov_matrix[0][2] + 2 * optimized_weights[0] * optimized_weights[1] *
cov_matrix[0][1] + 2 * optimized_weights[1] * optimized_weights[2] * cov_matrix[1][2]) ** 0.5

# Calculate the amounts to invest in each stock
total_investment = 100000     # Total investment amount
amount_x = optimized_weights[0] * total_investment
amount_y = optimized_weights[1] * total_investment
amount_z = optimized_weights[2] * total_investment

print("Optimized Weights:", optimized_weights)
print("Optimized Portfolio Standard Deviation:", optimized_portfolio_stddev)
print("Amount to Invest in ASIANPAINT:", '{:.11f}'.format(amount_x))
print("Amount to Invest in ITC:", amount_y)
print("Amount to Invest in TITAN:", amount_z)
print("Total Invested Amount:", amount_x + amount_y + amount_z)

print("\nRevised Model Considering Data from Python: Prices from 24-09-2022 to 24-09-2023")
n1 = 'TCS'
n2 = 'TECHM'
n3 = 'HCLTECH'
model(n1, n2, n3)

# Thank You !!
```

# OUTPUT:

S.No 1 - The Mean of INDUSINDBK is -0.06895956395912455
S.No 1 - The S.D of INDUSINDBK is 1.5815557409041228
S.No 2 - The Mean of MARUTI is -0.11495633369298155
S.No 2 - The S.D of MARUTI is 1.114034466299496
S.No 3 - The Mean of SBIN is -0.16129144002185583
S.No 3 - The S.D of SBIN is 1.300208881401146
S.No 4 - The Mean of M&M is -0.09194969145985053
S.No 4 - The S.D of M&M is 1.271324707579204
S.No 5 - The Mean of ASIANPAINT is -0.12309846072905688
S.No 5 - The S.D of ASIANPAINT is 1.117595476331528
S.No 6 - The Mean of TECHM is 0.0577411532074746 4
S.No 6 - The S.D of TECHM is 1.3518322127050477
S.No 7 - The Mean of BAJAJFINSV is -0.26194363930896597
S.No 7 - The S.D of BAJAJFINSV is 1.3626478118086243
S.No 8 - The Mean of LT is 0.018656430423613198
S.No 8 - The S.D of LT is 1.082295082301565
S.No 9 - The Mean of COALINDIA is 0.02001769308103163
S.No 9 - The S.D of COALINDIA is 1.320964954798953
S.No 10 - The Mean of HEROMOTOCO is -0.08713211080194098
S.No 10 - The S.D of HEROMOTOCO is 1.2097843448044021

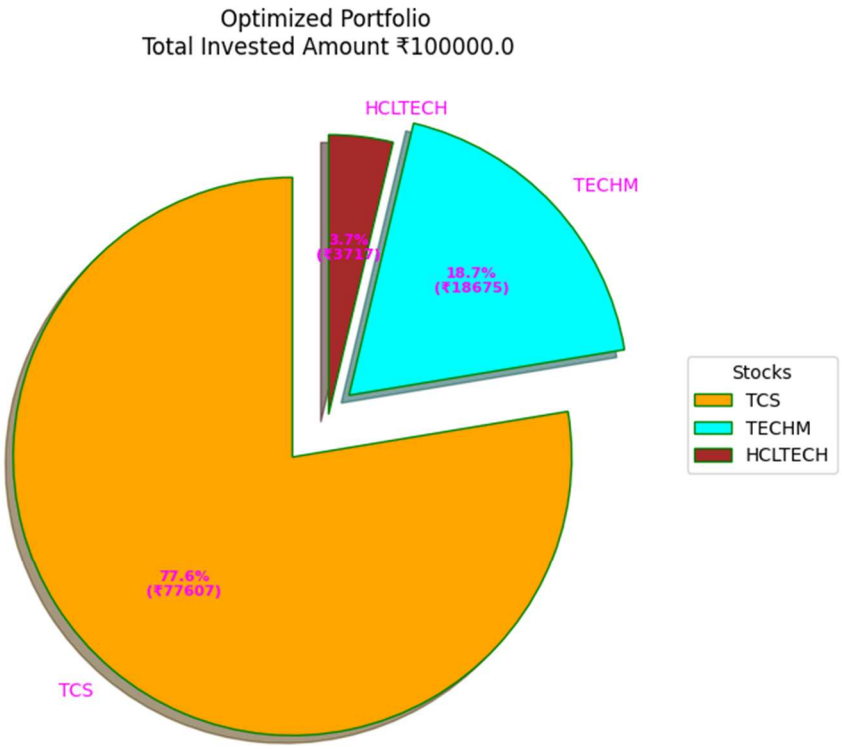
S.No 11 - The Mean of LTIM is 0.006657152842629899
S.No 11 - The S.D of LTIM is 1.474282120272349
S.No 12 - The Mean of TATACONSUM is -0.06132706905574073
S.No 12 - The S.D of TATACONSUM is 1.116517136251457
S.No 13 - The Mean of HDFCLIFE is -0.05491584709514896
S.No 13 - The S.D of HDFCLIFE is 1.5780873568080394
S.No 14 - The Mean of TCS is 0.059598777583528655
S.No 14 - The S.D of TCS is 1.0062900414555476
S.No 15 - The Mean of AXISBANK is -0.011083140645774429
S.No 15 - The S.D of AXISBANK is 1.1688959033979862
S.No 16 - The Mean of HINDUNILVR is -0.051059822127108044
S.No 16 - The S.D of HINDUNILVR is 1.0509886977083174
S.No 17 - The Mean of KOTAKBANK is -0.12309712663830982
S.No 17 - The S.D of KOTAKBANK is 0.9705245794718332
S.No 18 - The Mean of BRITANNIA is -0.08823256232720696
S.No 18 - The S.D of BRITANNIA is 1.0432871710800908
S.No 19 - The Mean of NESTLEIND is -0.03441168648209569
S.No 19 - The S.D of NESTLEIND is 0.9944453774507007
S.No 20 - The Mean of GRASIM is -0.022077627267998102
S.No 20 - The S.D of GRASIM is 1.0974020638476334
S.No 21 - The Mean of BAJFINANCE is -0.20502109886144665
S.No 21 - The S.D of BAJFINANCE is 1.385187663831959
S.No 22 - The Mean of HCLTECH is 0.04467802480891863
S.No 22 - The S.D of HCLTECH is 1.1295823534475251
S.No 23 - The Mean of RELIANCE is -0.0847728033639779
S.No 23 - The S.D of RELIANCE is 1.114918309149737
S.No 24 - The Mean of HINDALCO is -0.10430282811074802
S.No 24 - The S.D of HINDALCO is 1.6343424735216934
S.No 25 - The Mean of APOLLOHOSP is -0.10958848407633485
S.No 25 - The S.D of APOLLOHOSP is 1.3916312732316232
S.No 26 - The Mean of INFY is -0.021595395608729792
S.No 26 - The S.D of INFY is 1.01701603374312
S.No 27 - The Mean of TATASTEEL is -0.12714568394589806
S.No 27 - The S.D of TATASTEEL is 1.3745449279227555
S.No 28 - The Mean of NTPC is -0.02583267574897471
S.No 28 - The S.D of NTPC is 1.2386668294578487
S.No 29 - The Mean of JSWSTEEL is -0.07481238248142001
S.No 29 - The S.D of JSWSTEEL is 1.379034327511677
S.No 30 - The Mean of EICHERMOT is -0.22362893431494282
S.No 30 - The S.D of EICHERMOT is 1.4788376692587337
S.No 31 - The Mean of TITAN is -0.10766913136920721
S.No 31 - The S.D of TITAN is 1.1164087646679717
S.No 32 - The Mean of BHARTIARTL is -0.04459423108733848
S.No 32 - The S.D of BHARTIARTL is 1.1639448908939096
S.No 33 - The Mean of ADANIENT is -0.36010826824497566
S.No 33 - The S.D of ADANIENT is 4.591194961088048
AttributeError: SUCESSFULLY HANDELED - Stock ICICIBANK has mixed datatypes in same column
S.No 34 - The Mean of ICICIBANK is -0.07464514184630217
S.No 34 - The S.D of ICICIBANK is 0.9426625391376977
S.No 35 - The Mean of ONGC is 0.0355250549861437
S.No 35 - The S.D of ONGC is 1.2283945852382856
S.No 36 - The Mean of ADANIPORTS is -0.28300065101624844
S.No 36 - The S.D of ADANIPORTS is 2.7270417299901157
S.No 37 - The Mean of BPCL is -0.09875555350956891
S.No 37 - The S.D of BPCL is 1.2757458219320568
S.No 38 - The Mean of TATAMOTORS is -0.15945442122611994
S.No 38 - The S.D of TATAMOTORS is 1.3327169238516907

```
S.No 39 - The Mean of SUNPHARMA is 0.002678211302387599
S.No 39 - The S.D of SUNPHARMA is 0.9641162150845868
S.No 40 - The Mean of SBILIFE is -0.1026868357000643
S.No 40 - The S.D of SBILIFE is 1.289806277225226
S.No 41 - The Mean of ITC is -0.03564788001208404
S.No 41 - The S.D of ITC is 1.1433354173375034
S.No 42 - The Mean of DIVISLAB is -0.06940381406290821
S.No 42 - The S.D of DIVISLAB is 1.8468675013762685
S.No 43 - The Mean of POWERGRID is -0.06070444564908107
S.No 43 - The S.D of POWERGRID is 1.3629876558414737
S.No 44 - The Mean of ULTRACEMCO is -0.06465216079594001
S.No 44 - The S.D of ULTRACEMCO is 1.077783061532093
S.No 45 - The Mean of HDFCBANK is -0.020905464016428538
S.No 45 - The S.D of HDFCBANK is 0.9729978160857867
S.No 46 - The Mean of BAJAJ-AUTO is -0.008201220394535245
S.No 46 - The S.D of BAJAJ-AUTO is 1.1917621162646024
S.No 47 - The Mean of CIPLA is -0.12512424014662066
S.No 47 - The S.D of CIPLA is 1.1593905706421488
S.No 48 - The Mean of UPL is -0.24261480540817176
S.No 48 - The S.D of UPL is 1.3139245342786963
S.No 49 - The Mean of DRREDDY is -0.006644528474565617
S.No 49 - The S.D of DRREDDY is 1.017317085885501
S.No 50 - The Mean of WIPRO is -0.12534012646913895
S.No 50 - The S.D of WIPRO is 0.9722658972238264

Initial Model Considering Data from Excel: Prices from 1st January 2022 to 30 December 2022
Optimized Weights: [2.56395023e-16 1.00000000e+00 0.00000000e+00]
Optimized Portfolio Standard Deviation: 5.546035720860749
Amount to Invest in ASIANPAINT: 0.00000000003
Amount to Invest in ITC: 100000.0
Amount to Invest in TITAN: 0.0
Total Invested Amount: 100000.00000000003

Revised Model Considering Data from Python: Prices from 24-09-2022 to 24-09-2023
The Mean of TCS is 0.059598777583528655
The Mean of TECHM is 0.05774115320747464
The Mean of HCLTECH is 0.04467802480891863
The Covariance matrix is
 [[1.01261965 0.70434141 0.686156  ]
 [0.70434141 1.82745033 0.68163431]
 [0.686156   0.68163431 1.27595629]]
Optimized Weights: [0.77607804 0.18675173 0.03717023]
Optimized Portfolio Standard Deviation: 0.9636449595020761
Amount to Invest in TCS: 77607.80396760767
Amount to Invest in TECHM: 18675.172826104237
Amount to Invest in HCLTECH: 3717.0232062880823
Total Invested Amount: 100000.0
PS C:\Users\ankit\Documents\VScode\Internship_project>
```

# PIE CHART




## Solution Analysis *of Revised Model Considering Data from Python:*

Optimized Weights: [0.77607804 0.18675173 0.03717023]

Optimized Portfolio Standard Deviation: 0.9636449595020761

Amount to Invest in TCS: 77607.80396760767

Amount to Invest in TECHM: 18675.172826104237

Amount to Invest in HCLTECH: 3717.0232062880823

Total Invested Amount: 100000.0

## References:


- https://www.educba.com/sample-standard-deviation-formula/
- https://pyportfolioopt.readthedocs.io/en/latest/index.html#
- https://en.wikipedia.org/wiki/Portfolio_optimization
- https://smartasset.com/investing/guide-portfolio-optimization-strategies
- https://medium.com/@nusfintech.ml/ml-optimisation-for-portfolio-allocation-9da34e7fe6b1
- https://docs.scipy.org/doc/scipy/reference/optimize.html
- https://matplotlib.org/stable/index.html
- https://www.geeksforgeeks.org/plot-a-pie-chart-in-python-using-matplotlib/
- https://www.nseindia.com/market-data/live-equity-market?symbol=NIFTY%2050