**Step 1 : Pull Data from Moneycontrol**

Objective: To get NAV data for all pension funds

Technology: Python, Selenium, HighCharts, JavaScript

Python Program: moneyControlNAVCode.py

Moneycontrol tracks [all pension schemes](https://www.moneycontrol.com/personal-finance/nps-national-pension-scheme)

NAV data for each fund is stored in a chart which is in HighCharts format

The python program emulates a browser, opens each of those fund pages, clicks the entire history of the fund, gets the chart data for it’s entire history in HighCharts format, converts HighCharts format to Python readable list using a JavaScript converter & stores the data. This is repeated for each fund page automatically.

Once NAV is collected datewise, returns are calculated as % change in NAV on a daily basis.

**Step 2 : Pre-Process Data for Analysis**

Objective: To get clean daily returns data for all pension funds

Technology: Python

Python Program: moneyControlNAVCode.py , copulaGenerator.py

The returns data can be observed to have a number of spikes that interferes with further analysis. These spikes are removed using a multiple of quantile method. 5Th & the 95th percentile values is calculated on returns for each fund. Values outside of 4 times these percentiles are removed from the data. This process is repeated two times to remove residual outliers.

Schemes with data history of less than 500 data points are removed. This corresponds to schemes with less than 2 year history.

Only tier 1 schemes data is retained for further analysis.

Data for Saturday & Sundays are removed.

Only Gsec, CorpBonds & Equity asset classes data are retained. Alternative asset classes are not going to be considered in the analysis.

**Step 3 : Calculation of Key Statistics for Simulation**

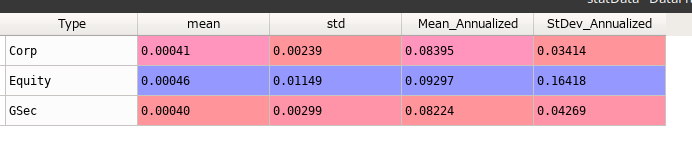
Objective: To calculated expected returns, volatility & correlation among chosen asset classes

Technology: Python

Python Program: copulaGenerator.py

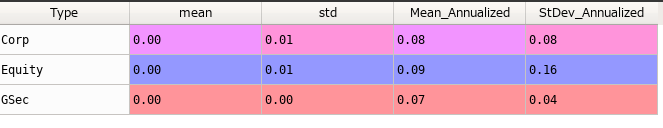
Daily returns across all funds are averaged out (equal contribution to each tier-i fund is assumed)

Daily mean & standard deviation is calculated from this data by asset class. These are annualized by using the number of working days in 2019 where NAVs were reported. The output looks like below:

It can be observed that corporate bonds have a lower volatility than GSec. This is because corporate bonds have lower liquidity and are infrequently traded. So NAVs don’t vary as much. To overcome this problem, a standard deviation of 8% is forced onto the corporate bonds asset class.

Additionally. GSec & corporate bonds have performed well in the past due to falling interest rate scenario. Those with new contribution will not benefit out of this. So an expected return of 6.5% and 8% are enforced onto GSec & Corp Bonds respectively going by the current prevailing rates in the market.

Final values look like below:



The calculated correlation matrix is also not positive-semi definite & has negative eigenvalues. This means that there are certain combinations of the asset class that could potentially make the portfolio variance negative. This is avoided by applying a Cholesky decomposition. The two matrix output is listed below:

|  |  |
| --- | --- |
|  |  |

**Step 4 : Simulate Economic Scenarios for Asset Returns**

Objective: To simulate the returns of asset classes & calculate the portfolio aggregate value for Rs.1 salary today.

Technology: Python

Python Program: copulaGenerator.py

Assumptions:

Mean-Variance portfolio theory holds good (asset returns are normally distributed)

Constancy of parameters

Salary growth rate is assumed to be 8% p.a compounded monthly

Retirement age is compulsorily 60. Min age of entry is 18.

1000 correlated sets of normal random variables are generated using Gaussian Copula from the mean & covariance matrix for the asset classes for all months from now to retirement. For eg, if the person is 35 years old with 25 pending years (300 months) to retirement, this will be a **300\*3\*1000** matrix of random variables.

Compounded value of Rs.1 growing at 8% p.a compounded monthly invested per month in each asset class are calculated for 1000 simulations. This is then summed up for each simulation & asset class (summarizing & removing the monthly dimension).

This process is repeated for each ages between 18 and 59. The output will be a 1000\*3 (simulations\*asset class) matrix of accumulated values for each age group.

**Step 5 : Efficient Frontier for Portfolio**

Objective: To get the mean variance efficient frontier for the asset classes

Technology: Python

Python Program: optimalPortfolio.py

An efficient frontier is defined as a portfolio with highest return for a given level of risk or the highest risk for a given level of return.

This is computed based on mean & covariance matrix calculated above.

Mean values are rounded off the nearest 1 decimal in percentage (Eg: 8.1%) in this calculation to avoid too many frontier points.

**Step 6 : The Website**

Objective: To provide a user friendly & customized interface for individuals to plan their NPS portfolio

Technology: Python, Django, Bootstrap, ChartJS, HTML, JavaScript

Python Program: views.py

Assumptions: Annuity rate of 8.2%, salary growth rate of 8%. No decrements such as mortality or withdrawal.

The website requests 6 inputs from users :-

Age: Current age of the individual

Monthly Salary: Individual’s current monthly salary

Contribution Percent: Monthly salary contribution rate into NPS

Equity Percent: % of portfolio the individual intends to hold in equities

GSec Percent: % of portfolio the individual intends to hold in gsec

CBond Percent: % of portfolio the individual intends to hold in Cbonds (100% - Equity – GSec)

For each of the 1000 scenarios, using the portfolio allocation, the final value of the overall portfolio is calculated for the level of contribution selected by the user. This will give the final portfolio value in a range of 1000 economic scenarios for the user. The following metrics are then calculated:

**Portfolio Mean & StDev:** mean & stdev from the chosen portfolio allocation using mean & covar matrix

**Salary@Retirement:** Current monthly pay increased @ 8% p.a compounded monthly.

**Annuity@Retirement: (**50 percentile value of the portfolio series/annuity purchase price)/12

**Replacement %:** Salary@Retirement/Annuity@Retirement

**Shortfall at 50% replacement:** Additional value of portfolio required at retirement to buy an annuity which is equal to 50% of final salary. PV of this is also calculated using the expected GSec yield. This can be viewed as the equivalent other asset accumulation outside of NPS that’s needed.

**Tvar/Expected Shortfall**: Mean of the final portfolio – mean of values below 50 percentile mark of portfolio. PV of this is also calculated using the expected GSec yield.

**Replacement Estimates/VaR**: Computes the value at risk estimates of replacement ratios by selected percentile at retirement. Stressed (5%), Pessimistic (25%) , Likely (50%), Optimistic (75%), Overtly Optimistic (95%)

**Efficient Frontier Curve**: Shows where the chosen allocation is in the efficient frontier curve.