Ahmed Ibrahim
Week05_Answers

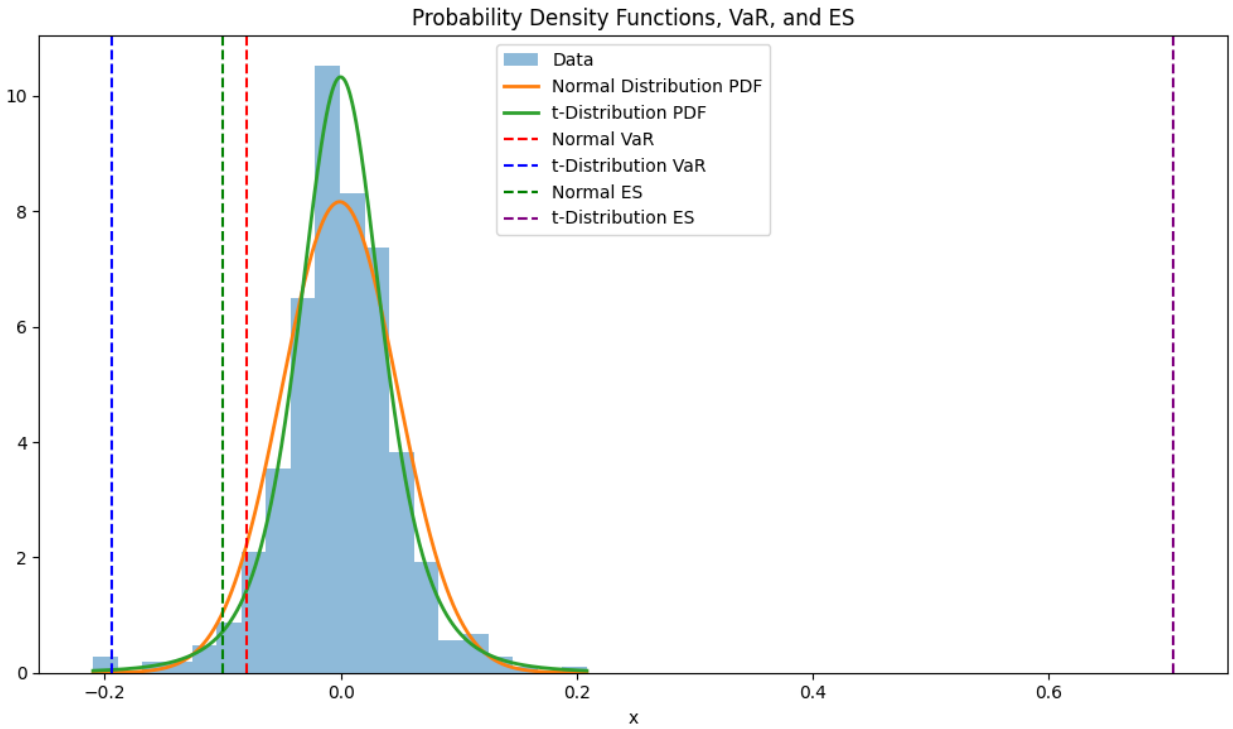
Problem 1:

Use the data in problem1.csv. Fit a Normal Distribution and a Generalized T distribution to this data. Calculate the VaR and ES for both fitted distributions.

Overlay the graphs the distribution PDFs, VaR, and ES values. What do you notice? Explain the differences.

I fit a normal distribution and a generalized t distribution by importing stats.norm and stats.t packages. Using the provided data, I calculated the value at risk(VaR) and expected shortfall for both distributions. Here are the results below with the distribution graphs.

```
VaR (Normal Distribution): -0.0795
ES (Normal Distribution): -0.0999
VaR (t-Distribution): -0.1943
ES (t-Distribution): 0.7056
```



From the graph above, we can notice that the generalized T distribution graph has a more negative VaR than the normal distribution which suggests that there is a higher change of larger loss with t distribution. The ES value for the generalized t distribution is positive and much higher than normal distribution,

indicating that the expected loss by VaR is greater than zero, compared to the negative ES for the normal distribution.

Problem 2:

In your main repository, create a Library for risk management. Create modules, classes, packages, etc as you see fit. Include all the functionality we have discussed so far in class. Make sure it includes

1. Covariance estimation techniques.
2. Non PSD fixes for correlation matrices
3. Simulation Methods
4. VaR calculation methods (all discussed)
5. ES calculation

Create a test suite and show that each function performs as expected.

Using the Julia template provided in the repo, I created four python files(.py) named as 'covariance_estimate.py', 'non_psd_fixes.py', 'simulation_methods.py', 'var_allcalc.py' that includes all the functionality as described above. The functions are then tested by importing them in the existing jupyter notebook to for testing. The python files are also included in the library.

Problem 3:

Use your repository from #2.
Using Portfolio.csv and DailyPrices.csv. Assume the expected return on all stocks is 0. This file contains the stock holdings of 3 portfolios. You own each of these portfolios.

Fit a Generalized T model to each stock and calculate the VaR and ES of each portfolio as well as your total VaR and ES. Compare the results from this to your VaR form Problem 3 from Week 4.

To calculate daily returns for the given set of stock prices, I begin by applying the `pct_change()` function, which computes the daily percentage change between consecutive prices. After obtaining this percentage change, I further process the resulting DataFrame by subtracting the mean using `np.mean()`. This results in a new DataFrame called "returns," which now holds the daily returns for each stock from the original prices DataFrame.

The "portfolio_price" function accepts a collection of stock prices and a portfolio of stocks as input parameters. Its purpose is to calculate and return the present value of each stock within the portfolio. The function follows these steps:

- It initializes an empty list called "pv."
- Then, it iterates through each stock in the portfolio.
- For each stock, it retrieves its current price by using the ".iloc[-1]" method.
- Finally, it appends this current price to the "pv" list.
- In the end, the "pv" list contains the current value of each stock within the portfolio.

Subsequently, I establish four distinct portfolios, labeled as A, B, C, and total. These portfolios are constructed using data from a portfolio DataFrame, which encompasses details regarding each stock's

respective portfolio and holdings. To facilitate this process, I utilize the "tolist()" method to convert the columns of the DataFrame into lists.

The code computes the daily returns for each portfolio by employing the logarithmic returns of the individual stocks contained within each portfolio. To achieve this, it utilizes the `np.diff()` function, which calculates the disparity between the logarithmic return of each day and the return of the preceding day.

The "calculate_t_pValues" function is designed to work with a portfolio, its daily returns, and the current prices of stocks within the portfolio as input parameters. Here's how it operates:

- Initially, it applies the "t.fit()" function to each stock's returns. This function provides the degrees of freedom, location, and scale parameters for each stock.
- Subsequently, it computes the cumulative distribution function (CDF) for the returns of each stock by employing the "t.cdf()" function.
- The resulting data is organized into a DataFrame called "return_cdf," which contains the CDF values for every stock within the portfolio.

The functions perform several key tasks:

1. It computes simulated returns using the "t.ppf()" function, taking into account the cumulative distribution function (CDF) values, degrees of freedom, location, and scale parameters. These simulated returns are stored in an array called "simu_return," with each element representing a stock within the portfolio.

2. After obtaining the simulated returns, the function calculates the potential portfolio values. For each simulation, it multiplies each simulated return by the current price of the corresponding stock within the portfolio. This multiplication is achieved through matrix multiplication.

3. The resulting values are collected in a list called "pVals," with each entry representing a potential portfolio value for a specific simulation. These values are arranged in ascending order.

The VaR and ES are calculated using the risk library. The results are shown below:

```
Simulating with 23 PC Factors: 95.06 % total variance explained
Simulating with 23 PC Factors: 95.06 % total variance explained
portfolio A
VaR = 19610.758617957355
ES = 46752.40365483138
```

```
Simulating with 26 PC Factors: 95.42 % total variance explained
Simulating with 26 PC Factors: 95.42 % total variance explained
portfolio C
VaR = 25471.554651659215
ES = 63180.311902574664
```

```
Simulating with 62 PC Factors: 95.19 % total variance explained
Simulating with 62 PC Factors: 95.19 % total variance explained
portfolio total
VaR = 55247.26733953692
ES = 138350.1922901589
```

All the VAR values in this problem are significantly greater than the VAR values calculated in problem 3 from Week 4. Please find the VAR values from week 4 below:

Exponentially weighted covariance

Portfolio A VAR: $15189.50331
Portfolio B VAR: $7423.58090
Portfolio C VAR: $25128.04557
Total Portfolio VAR: $47576.11536

Fitted AR(1) model:

Portfolio A VAR: $611.37705
Portfolio B VAR: $344.45046
Portfolio C VAR: $776.32527
Total Portfolio VAR: $47576.11536