CSC521 Final Project(Andrew Tillmann)

Description the Problem

The problem here is in the area of finical application. In the stock market when an entity is trading from a brokerage house they are able to use leverage. With this leverage it is possible to lose more than the equity of the account. Trading of an asset can only occur during its trading session. Furthermore, when the price closes at one session the next session does not open up at the last close price. It will open up at the price the buyers and sellers agree at the opening of the market session to be. The next sessions opening price may be similar to the previous session close but one is not based off the other. At the end of the session the brokerage house needs to be able to fully cover all of its trading position and still have equity left over. For this reason the brokerage house also passes this equity requirement on to its clients.

Since each of the assets held by the brokerage house's clients variates with each trading session the problem at hand is to calculate the overall exposure of the brokerage house. The broker needs to be prepared for bad cases scenarios. Since the data variates seemingly random the use of a Monte Carlo algorithm would be ideal for the task.

My Solution

For my implementation of the solution is broken down into four functions: get_data(stockString), simulate_position(nshares,stochString,days), simulate_client(data,client,days), VAR_client(data,client, N), VAR_brokerageHouse(data,N).

The get_data function needs the stock ticker as a string value (stockString) as an input. With this it then checks to see if it is in a stock dictionary. If the key is not found the information is downloaded from the YStock function from the nlib.py file. It downloads the past 250 trading sessions. Thus, as the program run through it only has to download each stocks data only once. Furthermore, this value is held in the ram of the computer so that these values can be speedy accessed later. The return of the function get_data is a Boolean indicating if the data is all there or not. Later functions will skip a stock in its calculation if the function get_data returns false.

The simulate_position function needs the number of shares(nshares), the stock ticker as a string value (stockString), and lastly the chosen days as a vector. It calls the The function get_data is called to make sure the stock's price data is in the stock dictionary. Furthermore, the simulate_position function will not run if the function get_data returns false. The overall goal of the function is to looks up the cumulative profit or loss that would have been return if the position of shares would have been held on the given days. Simulate_position then returns this cumulative profit or loss value.

The simulate_client function needs the data given from the input pickle file, the clients key and the vector of chosen days. The variable data contains the stock ticker as a string and the number of stocks that the given client has. Variable data and the chosen days are feed into the function

simulate_position the sum of all the returns for all of the clients stocks for the chosen day is returned.

The VAR_client function needs the data from the input pickle file, client and the number of times it should be simulated(N). The function then runs simulate_client function N times and each return value is recorded in a vector. During each simulation a vector of days is created that is feed into simulate_client. Thus, the simulation simulate_client will use the same days on a given simulation but different days on different simulation. This is since in a clients portfolio it helps to reduce the volatility by diversification. In the end the function will return the value located at the percentage position that was specified as input by the user when the main file was called.

The VAR_brokerageHouse needs the data from the input pickle file and the number of times it should be simulated(N). This function then runs simulate_client function N times for all clients then finds the net value of all clients positions. During each simulation a vector of days is created that is feed into simulate_client. Thus, the simulation simulate_client will use the same days on a given simulation(N) but different days on different simulation(N). Just like how the clients portfolio volatility was reduced by diversification the brokerage house volatility is reduced by diversification.

The main loop will first take the inputs from the user in the format {filename numberOfDays precentage}. The filename will open a pickle file which will be stored into variable data. A for loop will loop over all the clients in the data and print client,VAR_client(data,client). Once this for loop is complete another print is called print 'Brokerage House ',VAR_brokerageHouse(data) after this the main program is finished.

Algorithms Used

random.choice() from nlib from random:

This random is from the python standard library which uses the Mersenne Twister as the core pseudo-random number generators.

YStock() from nlib:

This function connects to the Yahoo finance servers gets the data needed for a given stock. This is put into a dictionary where the user can access the values for the given keys.

range():

This function will create a list of given values of the algorithmic progression that is specified.

sort():

This function will take the given list object and sort the function low to high.

reverse():

This function will transform the given list object and switch the values of the list in reverse order.

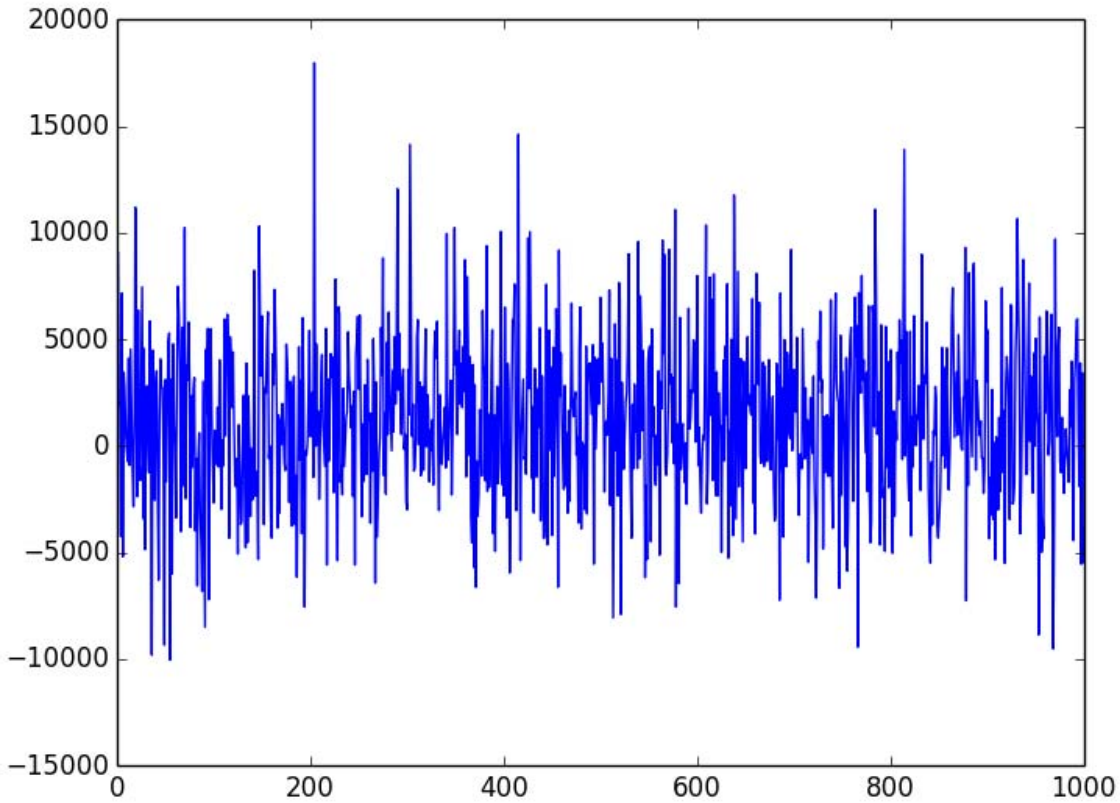
exp() from nlib:

Depending on the given case it will return cmath.exp(x) or math.exp(x) both from standard library.

Input and Output File Formats

The main loop will first take the inputs from the user in the format {filename numberOfDays percentage}. Thus, if this line is called from command line "python compute.py clients.pickle 7 99" the filename is client.pickle , numberOfDays is 7 and the percentage is 99. The output format is a loop that will print all the clients in the order they are found is the filename file. Alongside each client is the return value of VAR_client. After this loop finishes printing the client it will then print 'Brokerage House ' followed by the return of VAR_ brokerageHouse. No output file is created just the print statements.

Plots and Explanation of One Simulated Stock



BA Stock Simulation 6/1/2014 returns

The lines 12 , 46 and 55 need to be uncommented and then you will receive the first stock of the client VAR simulation. This will plot a bar graph of the of the given returns. The stock position that was chosen was the first clients on the data file and the first of the stocks listed with that client. In this case the stock had the ticker BA. What happen is the simulation above was that seven days were chosen from the past 250 trading days of returns on BA. Then a total return was calculated for holding those seven days. This is the y value in the given chart above. Each simulation was given its own bar. Since the simulation was run 1000 times there are 1000 bars showing all of the returns.