GROUP WORK PROJECT # 3
GROUP NUMBER: 5657

MScFE 622: Stochastic Modeling

FULL LEGAL NAME	LOCATION (COUNTRY)	EMAIL ADDRESS	MARK X FOR ANY NON-CONTRIBUTING MEMBER
BHARAT SWAMI	INDIA	bharatswami1299@gmail.com	
WYCLIFFE KIPKOECH CHERUIYOT	KENYA	cherukipkoech@gmail.com	
CHRIS ENNY OFIKWU	NIGERIA	c.ofikwu@outlook.com	

Statement of integrity: By typing the names of all group members in the text boxes below, you confirm that the assignment submitted is original work produced by the group (excluding any non-contributing members identified with an "X" above).	
Team member 1	**BHARAT SWAMI**
Team member 2	**WYCLIFFE CHERUIYOT**
Team member 3	**CHRISTOPHER ENNY OFIKWU**

Portfolio Selection Problem

The multi-armed bandit problems are a collection of problems in which we are given the choice to choose options which will give us reward from specific probability distributions with some mean as a result, and these probability distributions are unknown to us. By choosing multiple times from the options we are supposed to learn and choose the best options for better cumulative reward in the end. It is up to us to exploit the options which gave us better rewards previously or explore other options to learn and choose better options in future.
Now, the Portfolio Section Problem is an example of multi-armed bandit problems. In a portfolio selection problem we need to choose the assets from baskets of assets to maximize our return from the portfolio. We also need to change the number of assets at each time to make more profit and make our self-financing model assumption valid.

Model 1: Sequential Portfolio Selection Problem Pseudocode

Pseudocode[1]

$Parameters\colon\ \delta,\ N$
$Receive\ historical\ returns\ H_{i,t}\ of\ each\ asset\ i\ for\ t = 1, 2,\ldots., \delta\ ;$
$Filter\ to\ select\ a\ basket\ of\ K\ assets;$
$for\ t = 1,.., N\ do$
$\quad Choose\ portfolio\ \omega_t = (\omega_{1,t}, \omega_{2,t},\ldots, \omega_{K,t})^T;$
$\quad Observe\ R_t = (R_{1,t}, R_{2,t},\ldots, R_{K,t})\ and\ receive\ reward\ \omega_t R_t$
end

In the above pseudocode have two parameters $\delta\ and\ N$ **,** which are the number of times we are exploring the historical data and total number of historical data. Then we are calculating the historical return from the data. This historical return is the natural log of price ratio for each asset for every time step. Then we are filtering the K asset from the basket. And, finally we run the for loop for N time steps to calculate the weight row vector at each time steps t. The weight row vector is of dimension K X 1, this vector contains the weights of each asset at a particular time step. With weight vectors we also observe the rewards for each step, which is the cross product of matrices weight vector and return vector.

Data

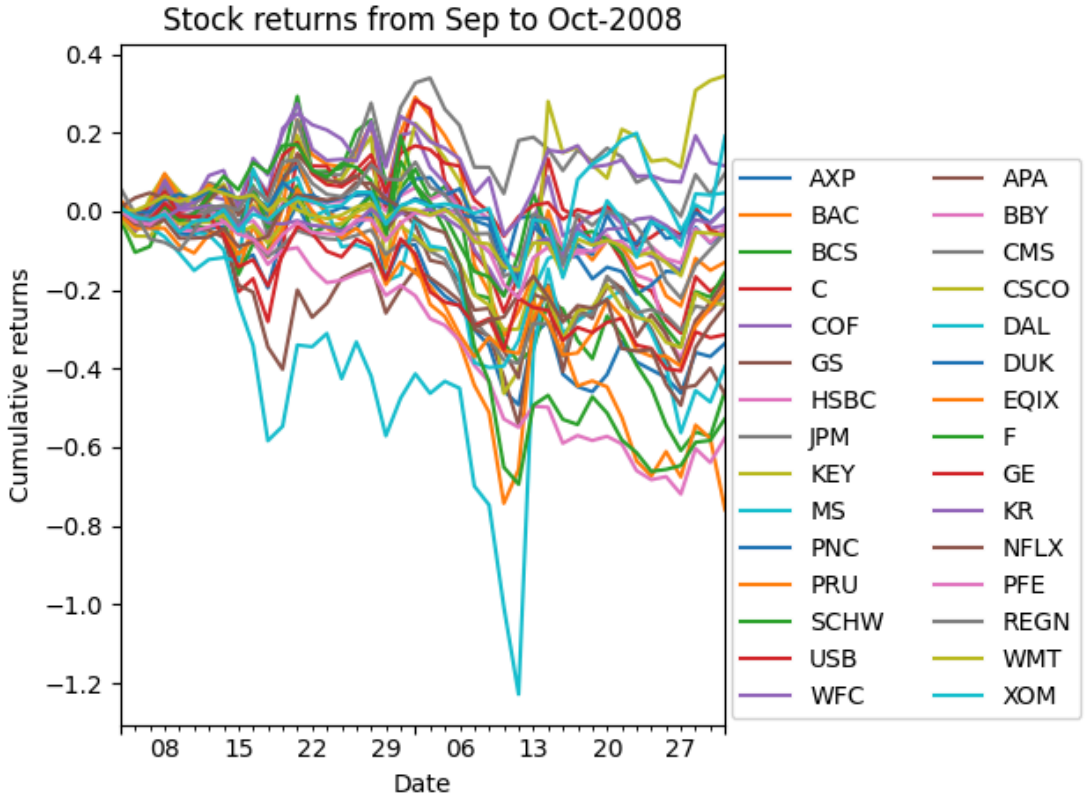
As mentioned in the assignment we need the daily returns of 30 assets, in which 15 are financial and other 15 are non-financial data. We are going to use 2-months data from 1st September 2008 to 30th October 2008 for our further calculations if not mentioned otherwise.

Tickers of assets we using for our calculations -

- Fifteen financial assets (GS, USB, MS, KEY, PNC, JPM, WFC, BAC, C, COF, AXP, PRU, SCHW, BBT, STI)
- Fifteen non-financial assets (CSCO, HCP, EQIX, DUK, KR, PFE, XOM, WMT, DAL, NFLX, GE, APA, F, REGN, CMS)

Remarks-

- Out of 15 financial institutions two of them no Longer exist due to merges that is BBT and STI which were replaced with HSBC and BCS
- Out of 15 of the non-financial institutions, one did not exist in the given period, that is HCP which was replaced with BBY.

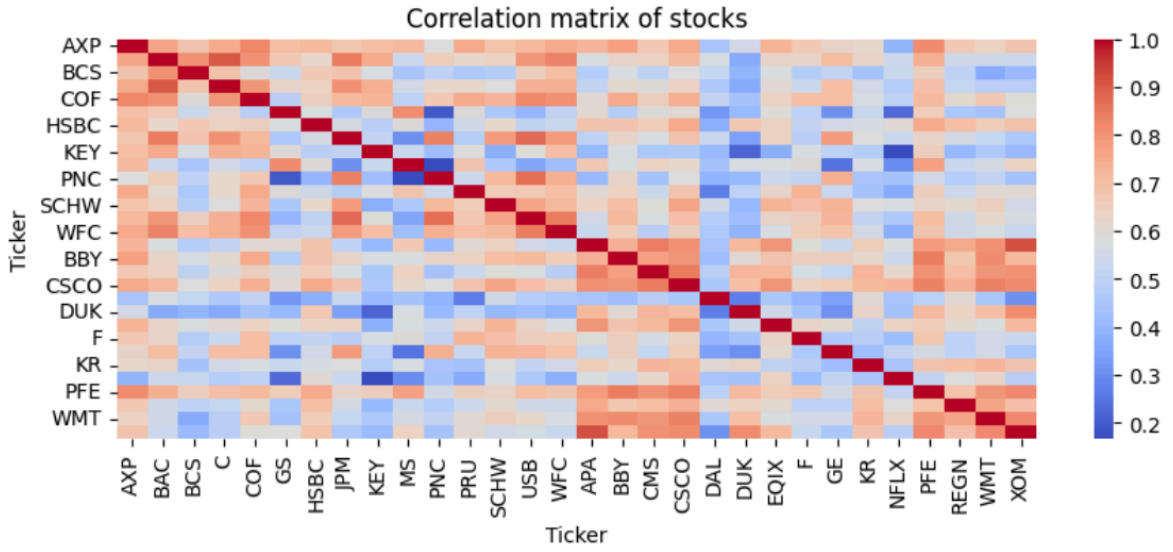


Graph 1: showing the cumulative returns for 30 assets from 1st September 2008 to 30th October 2008

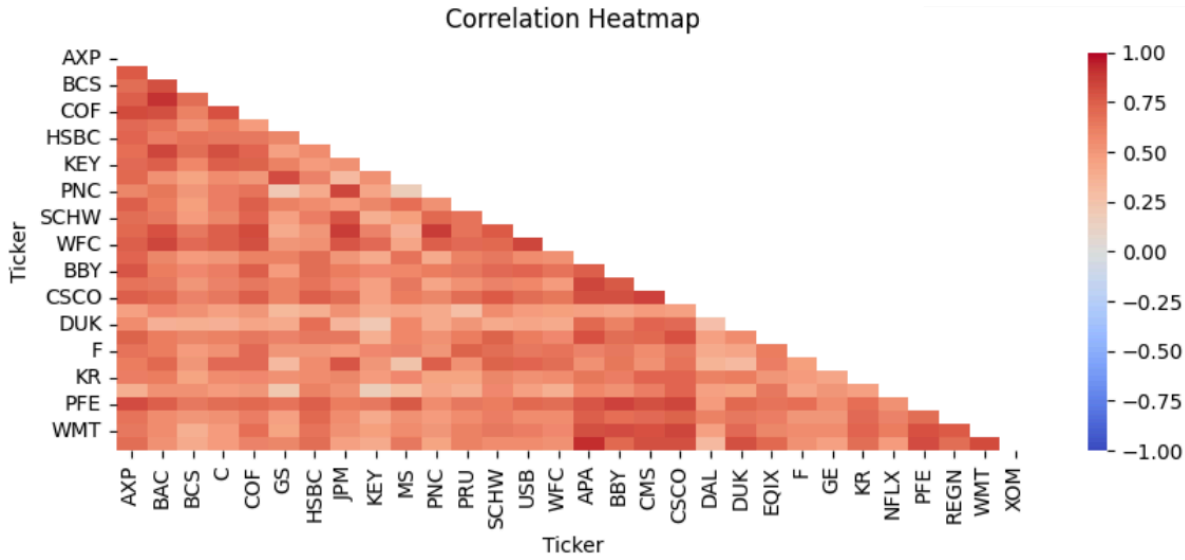
From the stock return visualization diagram Graph 1 above, it seems most of the stocks performed poorly, recording negative returns within the period September and October 2008, this could be attributed to the 2008 world economic recession.

Correlation Matrix

The correlation matrix measures the relationship between different stocks in the market, light colors indicate weaker correlation while the dark colors measure strong correlation, for instance XOM and WMT are weakly correlated and they are good for diversification of investment.



Graph 2: Correlation matrix for our 30 assets

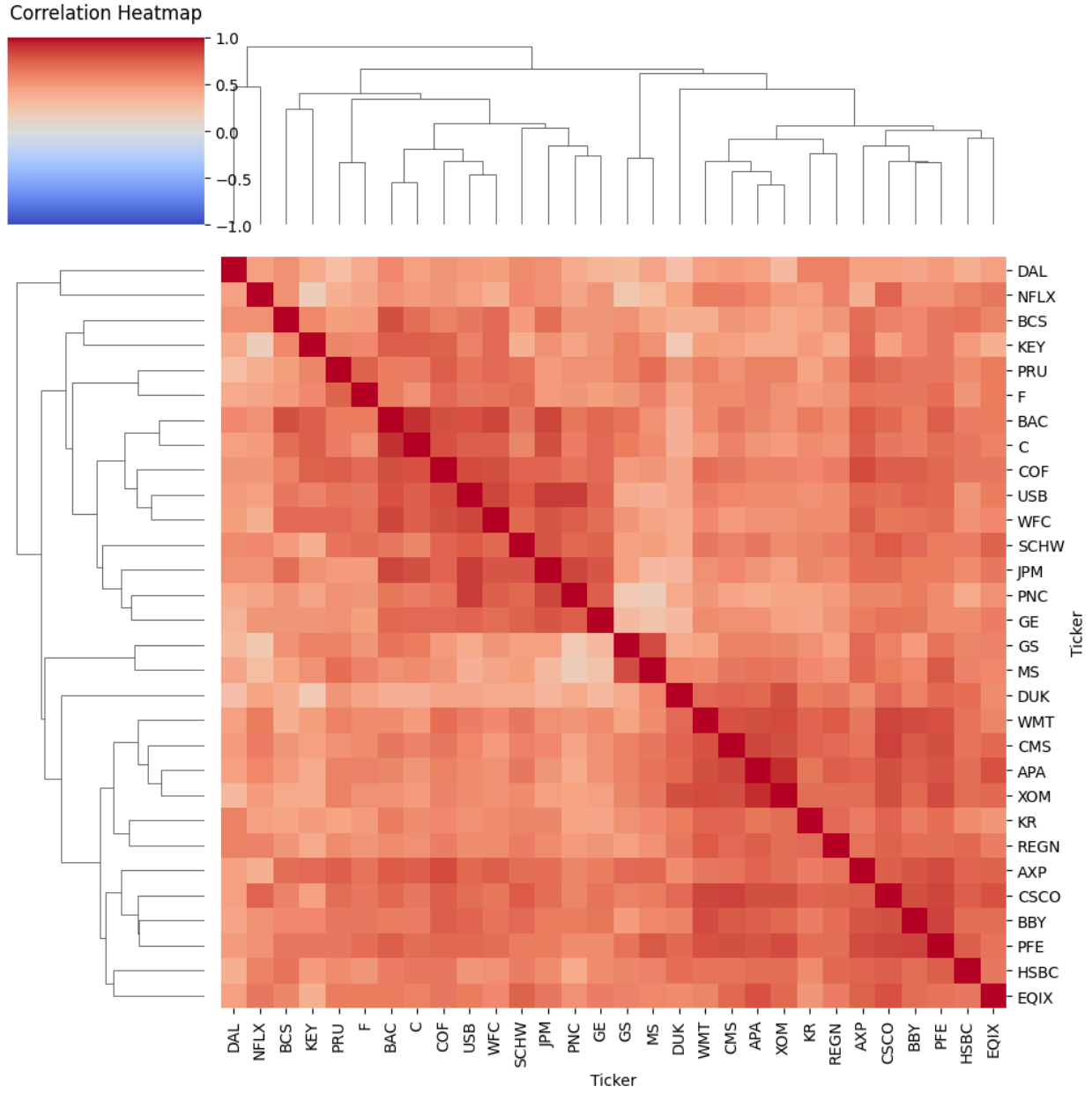


Graph 3: Correlation Heatmap for our 30 assets

Correlation heatmap uses colour gradient to show the relationship between stocks where red colour indicates a positive correlation while blue colour indicates a negative correlation. The heatmap above clearly indicates that all the financial and non-financial stocks were positively correlated, therefore moving in one direction.

Hierarchical Clustering

In Hierarchical clustering, stocks are classified or sorted according to their similarities or dissimilarities. The dendrogram represents the hierarchical structures of the clusters based on their similarity in correlation. Those stocks with almost similar correlation are clustered together. Dendrograms that merge earlier are considered to be more similar to one another as compared to those that merge later or those with long dendrograms.



Graph 4: Dendrogram for our assets

From the 30 stocks on dendrogen, it clearly indicates that there are at least ten clusters, out of which two are broader, that is they merge later in the dendrogram. These are considered as portfolios.

Portfolio Number	Assets	Remarks
Portfolio 1	WMT, CMS, APA, XOM, KR, REGN, AXP, CSCO, BBY, PFE, HSBC EQIX	These stocks are made of non-financial stocks except HSBC and AXP.
Portfolio 2	GE, C, JPM, SCHW, WFC, USB, COF, C, BAC, F, PRU, KEY, BCS	Portfolio 2 is made up of assets which are mainly financial stocks: These classes of stocks are mainly in the financial category.
Portfolio 3	HSBC, EQIX	
Portfolio 4	PFE, BBY, CSCO, AXP	
Portfolio 5	KR, REGN	Both non-financial assets
Portfolio 6	WMT, CMS, APA, XOM	
Portfolio 7	GE, PNC, JPM, SCHW	
Portfolio 8	WFC, USB, COF, C, BAC	
Portfolio 9	F and PRO	
Portfolio 10	KEY and BCS	

From portfolio 3 to portfolio 10, are stocks that merge earlier in the dendrogram, clearly indicating they have a strong relationship. In this portfolio, we have.

Upper Confidence Bound (UCB) Action

The purpose of upper confidence bound is to balance between exploration and exploitation in uncertainty about our action value estimates.
The best way is to select optimal action according to the following equation,

$$A_t = argmax_a[Q_t(a) + c\sqrt{\frac{ln\, t}{N_t(a)}}]$$

Steps followed when implementing Upper Confidence Bound (UCB) algorithm in multi-armed bandit

i) Initializing parameters, this involves the number of assets or arms (N) and the total number of trials (T). We also need to state exploration parameter c and initialize the estimated value of each armed-bandit. Where c controls the degree of exploration.
ii) Create the main loop and for each arm in order to calculate the upper confidence bound and select the one with the highest UCB, observe your rewards for a given period of time there after we update the estimated value of the selected asset.
iii) Repeat the same process until it converges or the maximum number simulation is reached.
iv) Decision is made based on the highest estimated value of a given arm.

Pseudocode

- ➔ Initialize parameters:
  - ◆ N: #assets
  - ◆ T: #Trails
  - ◆ c: Choose based on problem
  - ◆ Initialize for each asset $q_{valuw}[i]\ to\ Zero,\ i\ \ from\ 1\ to\ N$
  - ◆ Initialize visit count for each asset $N_{valuw}[i]\ to\ Zero,\ i\ \ from\ 1\ to\ N$
- ➔ Loop for t = 1 to T:
  - ◆ For each arm i, calculate the upper confidence bound

$$UCB[i]: UCB[i] = q_{value}[i] + c * \sqrt{\frac{ln(t)}{n_{visits}[i]}}$$

  - ◆ Select the arm with the highest UCB as the action to take:

$$action = argmax(UCB)$$

  - ◆ Observe the reward (daily return) for the selected asset at time t
  - ◆ Update the estimated value of the selected asset:

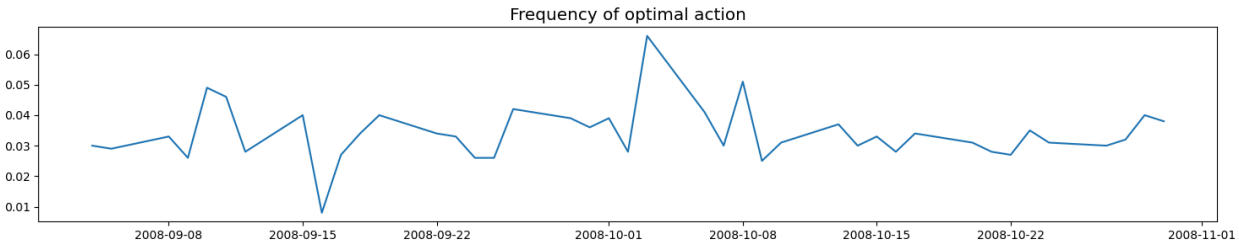
$$q_{value}[action] = q_{value}[action] + (\frac{1}{N_{visits}[action]+1})(reward - q_{value}[action])$$

  - ◆ Increment the visit count for the selected asset:

$$n_{visits}[action] = n_{visits}[action] + 1$$

- ➔ End loop
- ➔ After T trials, the portfolio consists of assets with the highest estimated values.

Below are results from above running above algorithm (in graphical format)



Graph 5: Frequency of optimal action

Graph 6: Comparison of (a) Max returns and average rewards (b) Average returns and average rewards from UCB algorithm.

Graph 7: Cumulative return bandit and cumulative average returns

From the above graphs we can see the optimal frequency fluctuate a lot for UCB condition algorithm. Other Graphs are showing the Average returns and rewards from iterating over the code with our chosen data of 30 assets.

Epsilon-Greedy Algorithm

Pseudocode

➔ Starting Point:

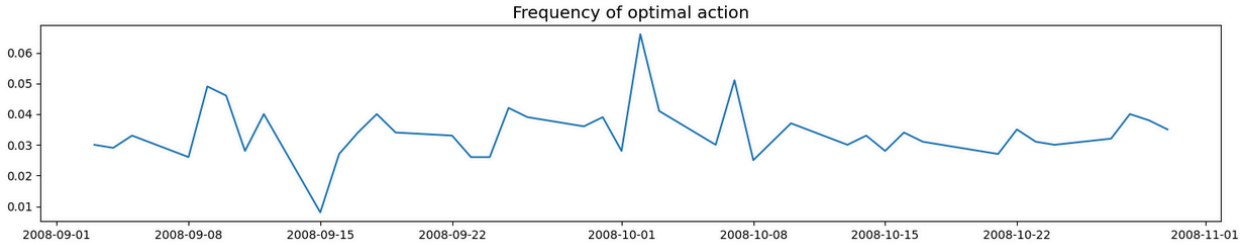
- q_{value}: An array that shows each action's estimated value.
- $eps : \xi$, The parameter that controls the probability of exploration in the exploration-exploitation trade-off.

➔ The optimal action function, or action selection:

- Based on the size of the q_{value} array, calculate the number of actions that are available ($n_{actions}$).
- Determine which action indices ($action_{hat}$) have the highest estimated value. Multiple indices might be returned if there are ties.
- Create a random number between 0 and 1, known as a "$randnum$."
- Select exploration if randnum is less than or equal to eps:
  - Choose at random one action out of all the actions that have a uniform probability.
- If not, pick exploitation:
  - Choose one of the tied actions at random if there are ties for the maximum estimated value.
  - If not, choose the course of action with the highest estimated value.

➔ Reward Update (reward_update function):

- Using the specified learning rate (α), update the estimated value of the chosen action based on the reward received.
- α determines the size of the update, which updates the anticipated value of the chosen action in the direction of the reward received.



Graph 8: Frequency of optimal action

From the figure above it clearly shows that the reinforcement learning algorithm did fluctuate for a number of days due to its exploration before it stabilized at approximately 0.03 on 8th october 2008 to the end of the period.

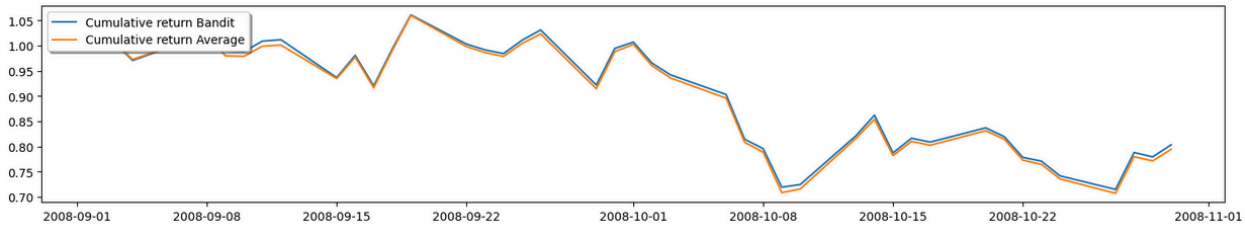
Graph 9: Comparison of (a) Max returns and average rewards (b) Average returns and average rewards from epsilon-greedy algorithm.

The max returns performed better than the average reward, even though it was not stable as compared to average reward. The average returns performed slightly better than average reward with their optimal rewards,all recording the average of negative returns.

Returns for Holding stocks for one year

Portfolio	Annualized Returns	Annualized Standard deviation
Armed Bandit Portfolio	-0.6112	0.8459
Equally Weighted Portfolio	-0.6336	0.8544

From the table of annualized returns, both armed bandit portfolio and equally weighted portfolio performed poorly by recording negative returns. But comparing the two portfolios, the armed bandit portfolio was better than the equally weighted portfolio in terms of returns and volatility, therefore not advisable to hold stock for one year.

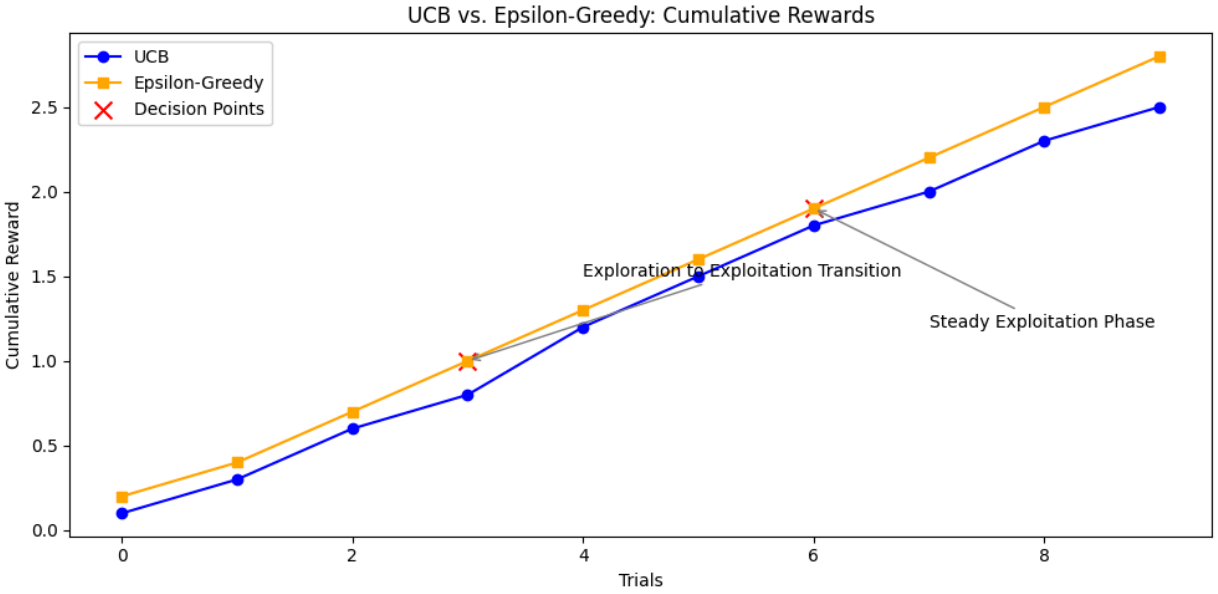


Graph 10: Cumulative average returns and rewards

From the figure above on the comparison of cumulative return bandit and the cumulative return average, it is clear that the cumulative return bandit performs better than the cumulative return average. Generally the cumulative returns for both techniques are falling and slightly volatile as the time goes.

Comparing Results from Epsilon-Greedy Algorithm And UCB

Comparison in Graphical Form-



Graph 11 : epsilon-greedy algorithm. and UCB results

From the UCB vs Epsilon-Greedy cumulative rewards graph, it is clear that UCB algorithm exploration is slower in the beginning as compared to epsilon-greedy as indicated by lower cumulative rewards from 0 to approximately 4 trials where learning took place and thereafter the phase of steady exploitation is experience.

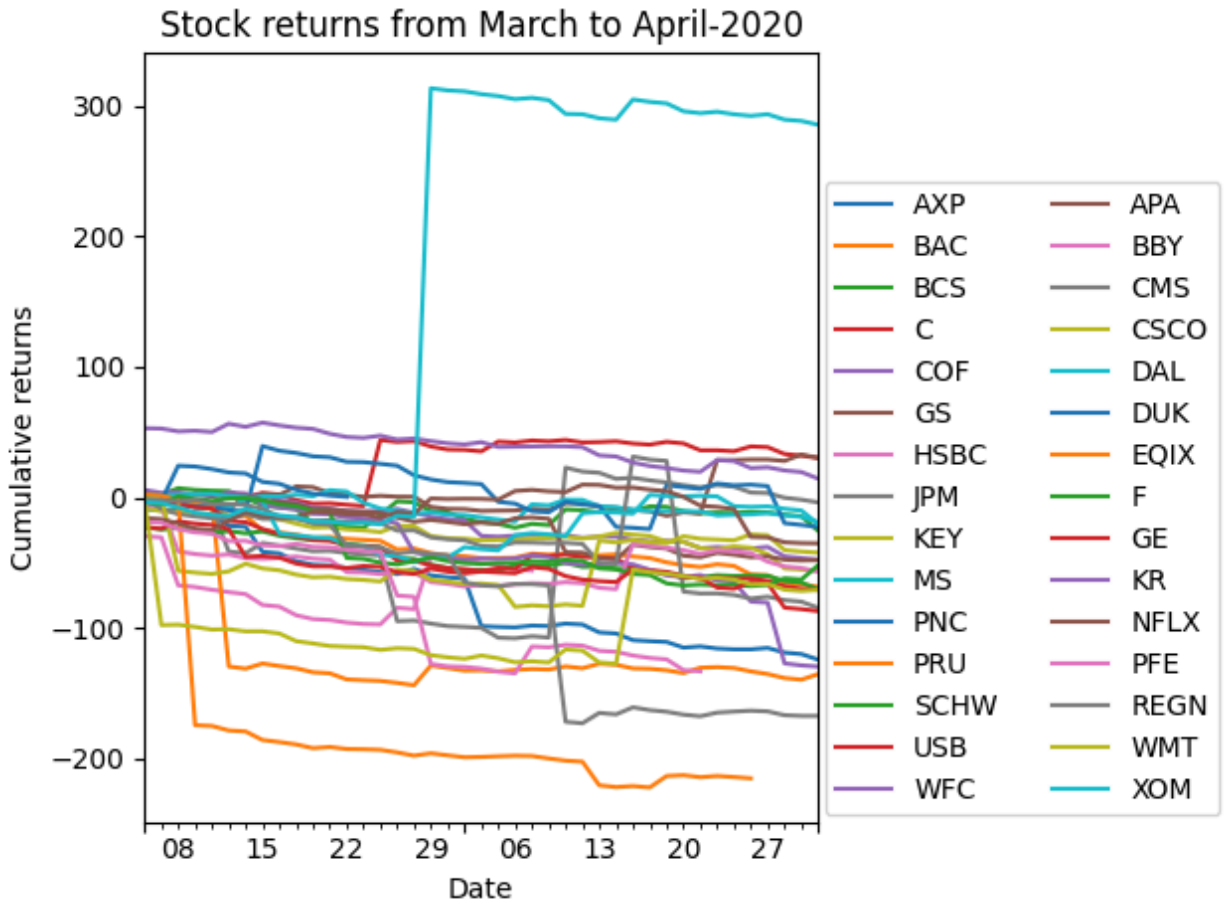
UCB is known to have good balancing in between exploration and exploitation where enough time is given to less explored options as it exploits arms with higher returns as learning continues.

On the other hand, epsilon greedy recorded steady improvement in cumulative rewards, this could be due to a good choice of the epsilon.

In general, the epsilon-greedy performs extremely well in terms of cumulative rewards as compared to UCB in this case.

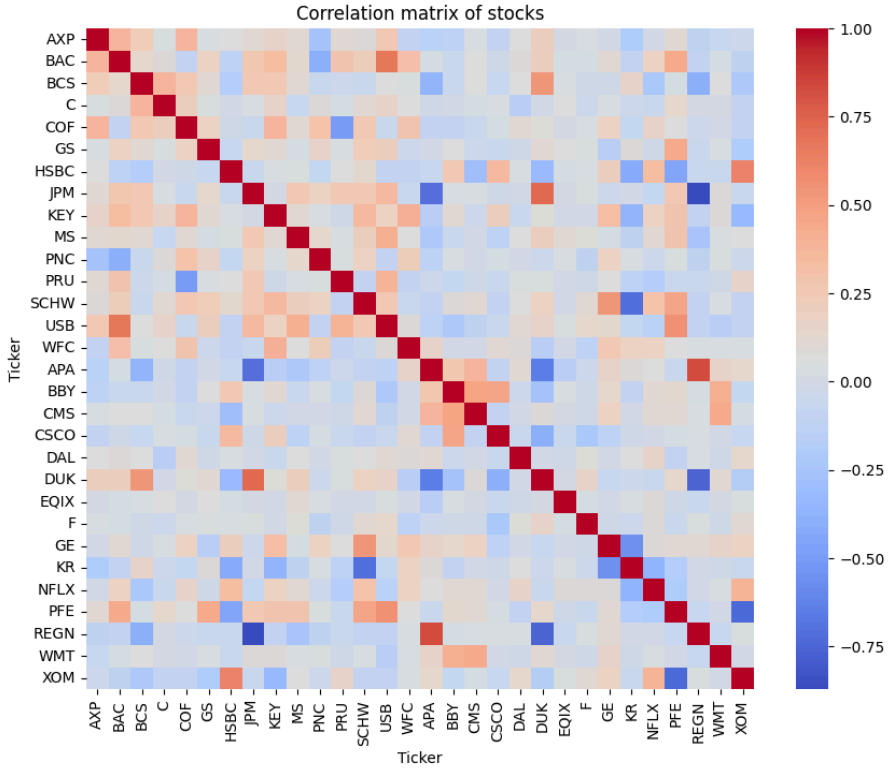
New Data (more recent data)

Now we are going to use a new time period for our data. The tickers or assets are the same as before in the same categories,i.e.15 Financial and 15 Non-Financial Assets. Now the time period is more recent, which is 1st March 2020 to 30th April 2020.
Below are the Graphical representation as before -

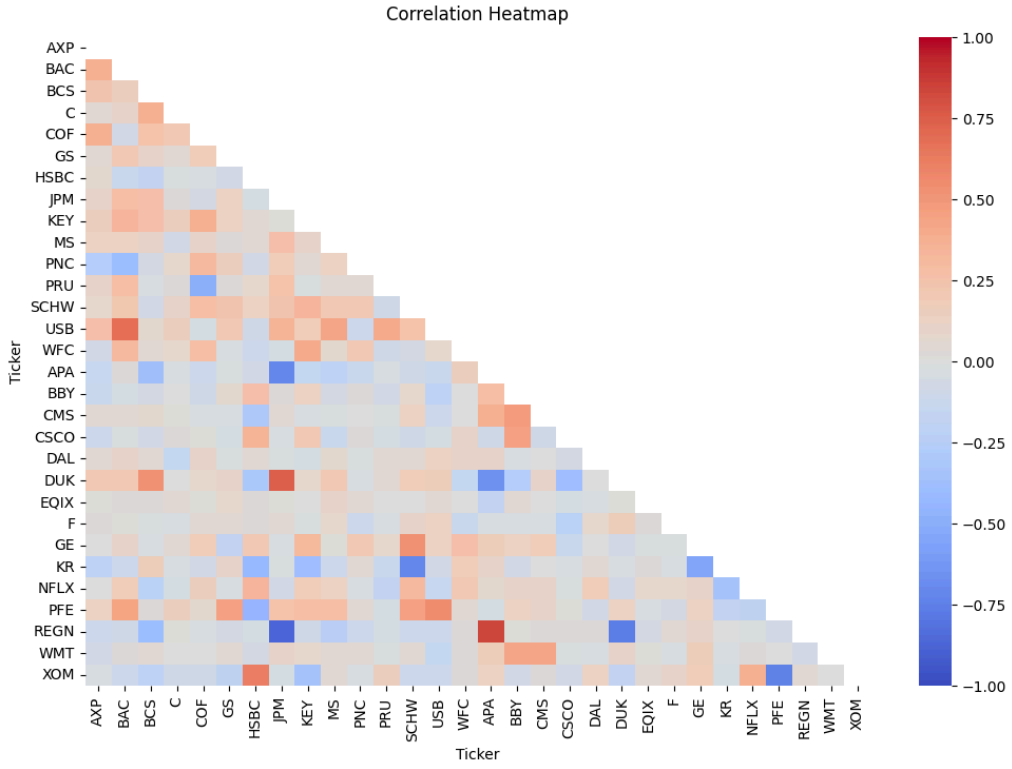


Graph 12: Cumulative returns from March 2020 to April 2020

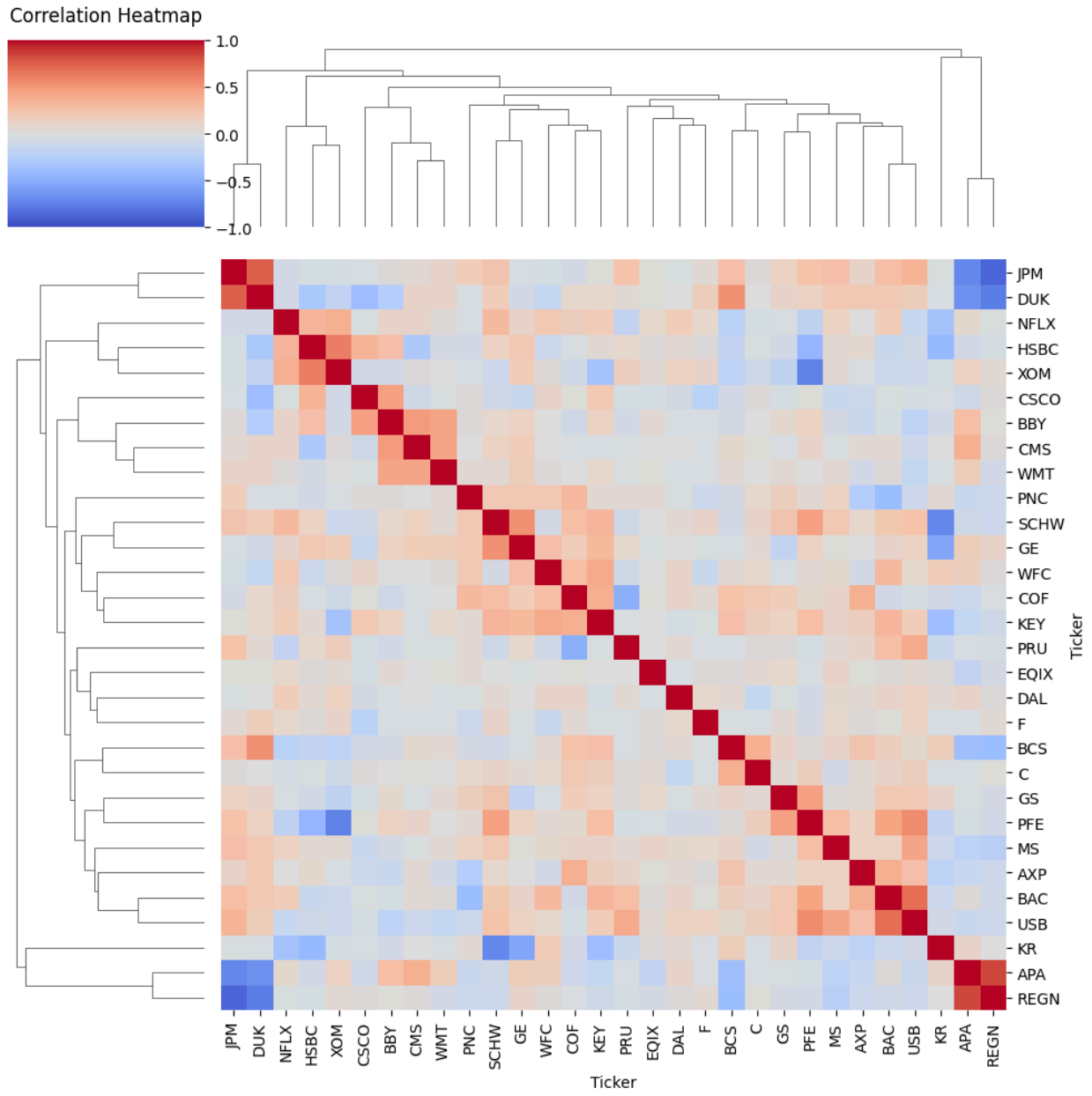
From the stock return visualization diagram Graph 12 above, it seems most of the stocks **again** performed poorly, recording negative returns within the period March 2020 and April 2020 with respect to time period from 1st September 2008 to 30th October 2008 from previous data.



Graph 13 : Correlation Matrix for data in timestamp March 2020 to April 202



Graph 13 : Correlation Heatmap for data in timestamp March 2020 to April 202



Graph 14: Dendrogram for our assets from time march 2020 to April 2020

From the above Correlation matrix and Heatmap we can say the correlations between the assets became weak in the recent time. Dendrogram also shows the change in the groups of assets in map form.

Rerunning the epsilon-greedy algorithm and UCB with different parameters on new data (more recent data)

Our group investigated multi-armed bandit algorithms, concentrating on two well-liked approaches: UCB and Epsilon-Greedy. These algorithms are essential for making decisions when there are many possibilities (arms) and little information available. We now explore how incorporating more recent data affects their performance. We acquired historical financial information for a group of businesses, covering the years prior to the present. Daily returns for both financial and non-financial enterprises are included in the dataset. By applying hierarchical classification techniques, these data sets were further categorized or divided into more manageable groupings.

Algorithm Implementation:

We used Python to implement the UCB and Epsilon-Greedy algorithms. The confidence parameter (c for UCB) and the exploration-exploitation trade-off (epsilon for Epsilon-Greedy) were adjusted. We changed the holding duration for each method to evaluate the effect of more recent data.

UCB Algorithm:

Utilizing data from the latest time frame, UCB is able to promptly adjust to evolving market circumstances. It quickly recognises attractive assets and modifies the balance between exploration and exploitation appropriately. The most recent information may cause a quick shift in the portfolio's makeup. UCB stabilizes as the holding duration increases, such as to six months. It emphasizes long-term patterns and is less dependent on cyclical variations. The portfolio emphasizes stability while progressively changing. Greedy-Epsilon Algorithm: Even with new data, Epsilon-Greedy keeps a balanced approach. It continues to prioritize assets with greater estimated worth while also exploring new ones. The portfolio effortlessly adjusts to capture both long-term stability and short-term rewards. Long-term consistency is maintained by Epsilon-Greedy. Without making any abrupt changes, it progressively incorporates new knowledge. A combination of familiarity and exploration may be seen in the portfolio composition.

Reference :

1. *Huo "Risk-aware multi-armed bandit problem with application to portfolio selection" 2017.*
2. *WQU Notes from 24/03 622 Stochastic Modeling Course*