FULL LEGAL NAME	LOCATION (COUNTRY)	EMAIL ADDRESS	MARK X FOR ANY NON-CONTRIBUTING MEMBER
Arkam Hayat	India	arkamhyatt786@gmail.com	
Regulavalasa Krishna Vamsi	India	krishnavamsi8262@gmail.com	
Xinjie Wang	Germany	ljbg1996wang@gmail.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	**Arkam Hayat**
Team member 2	**Regulavalasa Krishna Vamsi**
Team member 3	**Xinjie Wang**

Use the box below to explain any attempts to reach out to a non-contributing member. Type (N/A) if all members contributed.
Note: You may be required to provide proof of your outreach to non-contributing members upon request.

Step 1

We have read the paper, "Risk-aware Multi-armed Bandit Problem with Application to Portfolio Selection" developed by Huo et al. [1]. The research paper introduces a sequential algorithm for portfolio selection that achieves maximum reward with minimum risk. The algorithm is combined with coherent risk measures and the theory of graphs.

Step 2

The following is a framework of MAB problem in portfolio selection:

Arms: Each asset within a selection of K assets is considered an arm.

Actions: A portfolio selection $\omega_t = (\omega_{1,t}, \ldots, \omega_{K,t})$ corresponds to pulling several arms simultaneously using different weights.

Rewards: The reward per stage is the weighted sum of returns, $\omega_t R_t$, where $R_t = (R_{1,t}, \ldots, R_{K,t})$.

Exploration **vs**. Exploitation: The algorithm balances exploring asset combinations and exploiting historically successful ones.

Learning: The algorithm adapts portfolio weights by learning the expected returns of assets over time.

To better illustrate, we provide a pseudocode in Algorithm 1 below:

Algorithm 1: Sequential Portfolio Selection

Procedure SequentialPortfolioSelection(δ, N):

1. **historical returns** ← GetHistoricalReturns(δ)
2. **K** ← SelectAssets(historical returns)
3. **cumulative reward** ← 0
4. For **t** ← 1 to **N** do:
    - **ωt** ← ChoosePortfolio(K, historical returns, t)
    - **Rt** ← ObserveReturns(K)
    - **reward** ← CalculateReward(ωt, Rt)
    - **cumulative reward** ← cumulative reward + reward
    - UpdateHistoricalData(historical returns, Rt)
5. Return **cumulative reward**

Procedure GetHistoricalReturns(δ):

- Return matrix of historical returns for all assets over **δ** periods

Procedure SelectAssets(historical returns):

- Return **K** (Number of selected assets)

Procedure ChoosePortfolio(K, historical returns, t):

- Return **ωt** (Vector of weights summing to 1)

Procedure ObserveReturns(K):

- Return **Rt** (Vector of **K** returns)

Procedure CalculateReward(ωt, Rt):

- $$\text{Return} = \sum_{i=1}^{K} \omega_{i,t} R_{i,t}$$

Procedure UpdateHistoricalData(historical returns, Rt):

- Update historical returns with new data **Rt**

Step 3

We have been able to proceed with the collection and processing of data according to plan. Member A downloaded daily returns data from 15 financial institutions while Member B downloaded the same data from 15 non-financial institutions in the months of September to October, 2008. Member C then aggregated data in a suitable Python time series data structure, from which the daily returns for all 30 series were computed.

Ticker	Company Name	Industry	Sector
PM	Philip Morris International Inc.	Tobacco	Consumer Staples
WFC	Wells Fargo & Co.	Banks	Financials
BAC	Bank of America Corp.	Banks	Financials
C	Citigroup Inc.	Banks	Financials
GS	Goldman Sachs Group Inc.	Capital Markets	Financials
USB	U.S. Bancorp	Banks	Financials
MS	Morgan Stanley	Capital Markets	Financials
KEY	KeyCorp	Banks	Financials
PNC	PNC Financial Services Group Inc.	Banks	Financials
COF	Capital One Financial Corp.	Consumer Finance	Financials
AXP	American Express Co.	Consumer Finance	Financials
PRU	Prudential Financial Inc.	Insurance	Financials
SCHW	Charles Schwab Corp.	Capital Markets	Financials
BBT	BB&T Corp. (now part of Truist Financial)	Banks	Financials
STI	SunTrust Banks Inc. (now part of Truist Financial)	Banks	Financials
KR	The Kroger Co.	Food & Staples Retailing	Consumer Staples
PFE	Pfizer Inc.	Pharmaceuticals	Health Care
XOM	Exxon Mobil Corp.	Oil, Gas & Consumable Fuels	Energy
WMT	Walmart Inc.	Food & Staples Retailing	Consumer Staples
DAL	Delta Air Lines Inc.	Airlines	Industrials
CSCO	Cisco Systems Inc.	Communications Equipment	Information Technology
HCP	HCP Inc. (now Healthpeak Properties Inc.)	Health Care REITs	Real Estate
EQIX	Equinix Inc.	Specialized REITs	Real Estate
DUK	Duke Energy Corp.	Electric Utilities	Utilities
NFLX	Netflix Inc.	Entertainment	Communication Services
GE	General Electric Co.	Industrial Conglomerates	Industrials
APA	APA Corp. (formerly Apache Corp.)	Oil, Gas & Consumable Fuels	Energy
F	Ford Motor Co.	Automobiles	Consumer Discretionary
REGN	Regeneron Pharmaceuticals Inc.	Biotechnology	Health Care
CMS	CMS Energy Corp.	Multi-Utilities	Utilities

Table 1: List of Companies with Tickers, Industries, and Sectors

skimpy summary

Data Summary

dataframe	Values
Number of rows	42
Number of columns	30

Data Types

Column Type	Count
float64	30

number

column_name	NA	NA %	mean	sd	p0	p25	p50	p75	p100	hist
JPM	0	0	0.002304	0.07825	-0.1501	-0.05021	-0.001627	0.05311	0.1675	
WFC	0	0	0.002759	0.06838	-0.1458	-0.04174	-0.003483	0.0493	0.1287	
BAC	0	0	-0.003077	0.09945	-0.2623	-0.06507	-0.005298	0.06069	0.2256	
C	0	0	-0.004175	0.09679	-0.1844	-0.06328	-0.02104	0.03696	0.2402	
GS	0	0	-0.01109	0.07877	-0.1392	-0.06449	-0.01817	0.01869	0.25	
USB	0	0	-0.001354	0.04824	-0.0953	-0.03345	-0.005627	0.03221	0.1029	
MS	0	0	-0.009401	0.1771	-0.2589	-0.08661	-0.02325	0.03292	0.8698	
KEY	0	0	0.008212	0.137	-0.3333	-0.04955	0.006848	0.05738	0.5425	
PNC	0	0	-0.001294	0.05703	-0.1166	-0.04409	-0.001155	0.02926	0.1362	
COF	0	0	-0.0008501	0.07922	-0.1934	-0.05119	-0.01244	0.03177	0.1653	
AXP	0	0	-0.007992	0.07009	-0.1759	-0.04902	-0.01166	0.02796	0.1793	
PRU	0	0	-0.01807	0.1049	-0.2315	-0.09321	-0.03141	0.02683	0.3825	
SCHW	0	0	-0.003709	0.0739	-0.1204	-0.05498	-0.01426	0.04294	0.1965	
BBW	0	0	-0.007584	0.05899	-0.1487	-0.04037	-0.01476	0.03756	0.1258	
HSBC	0	0	-0.004854	0.04563	-0.1431	-0.03223	-0.0005516	0.01999	0.09423	
KR	0	0	0.0001493	0.02994	-0.04661	-0.02108	-0.001667	0.01909	0.07044	
PFE	0	0	-0.0009608	0.0386	-0.08523	-0.03032	-0.002676	0.02659	0.1017	
XOM	0	0	0.001122	0.06158	-0.1395	-0.02002	-0.001527	0.02524	0.1719	
WMT	0	0	-0.001445	0.03517	-0.08064	-0.01518	-0.004374	0.01399	0.1107	
DAL	0	0	0.004547	0.0866	-0.2113	-0.04328	-0.001388	0.03488	0.2348	
CSCO	0	0	-0.005642	0.05023	-0.1063	-0.03771	-0.01163	0.01696	0.138	
VRSN	0	0	-0.006987	0.06689	-0.1198	-0.05352	-0.01451	0.02547	0.1401	
EQIX	0	0	-0.004616	0.05516	-0.1057	-0.04384	-0.01723	0.04436	0.1306	
DUK	0	0	0.0001865	0.04324	-0.1051	-0.02386	-0.00527	0.0122	0.1331	
NFLX	0	0	-0.005761	0.05349	-0.1578	-0.03203	-0.001343	0.02091	0.1466	
GE	0	0	-0.007555	0.05252	-0.09592	-0.04422	-0.01094	0.01554	0.131	
APA	0	0	-0.004196	0.07632	-0.1824	-0.04823	-0.01168	0.04538	0.2132	
F	0	0	-0.01255	0.08506	-0.218	-0.0467	-0.01561	0.03009	0.247	
REGN	0	0	-0.001337	0.05869	-0.1112	-0.04118	-0.008323	0.0407	0.1429	
CMS	0	0	-0.005097	0.04485	-0.083	-0.03937	-0.008138	0.01337	0.1069	

End

Figure 2: Combined Data

Step 4 :

We performed correlation analysis and hierarchical clustering on the daily returns data. We built a hierarchical structure using agglomerative hierarchical clustering with Ward's method. The measure of dissimilarity used was the correlation distance.

The dendrogram is represented in how clusters are formed, and there is an optimal stopping point at four clusters. This gave us

Clusters 1 & 2: Outlier clusters of 5 securities

Cluster 3: Banking and Investment sector

Cluster 4: Non-financial multi-sector securities

This method works effectively in forming clusters of highly similar patterns of asset return hence much better portfolio analysis.

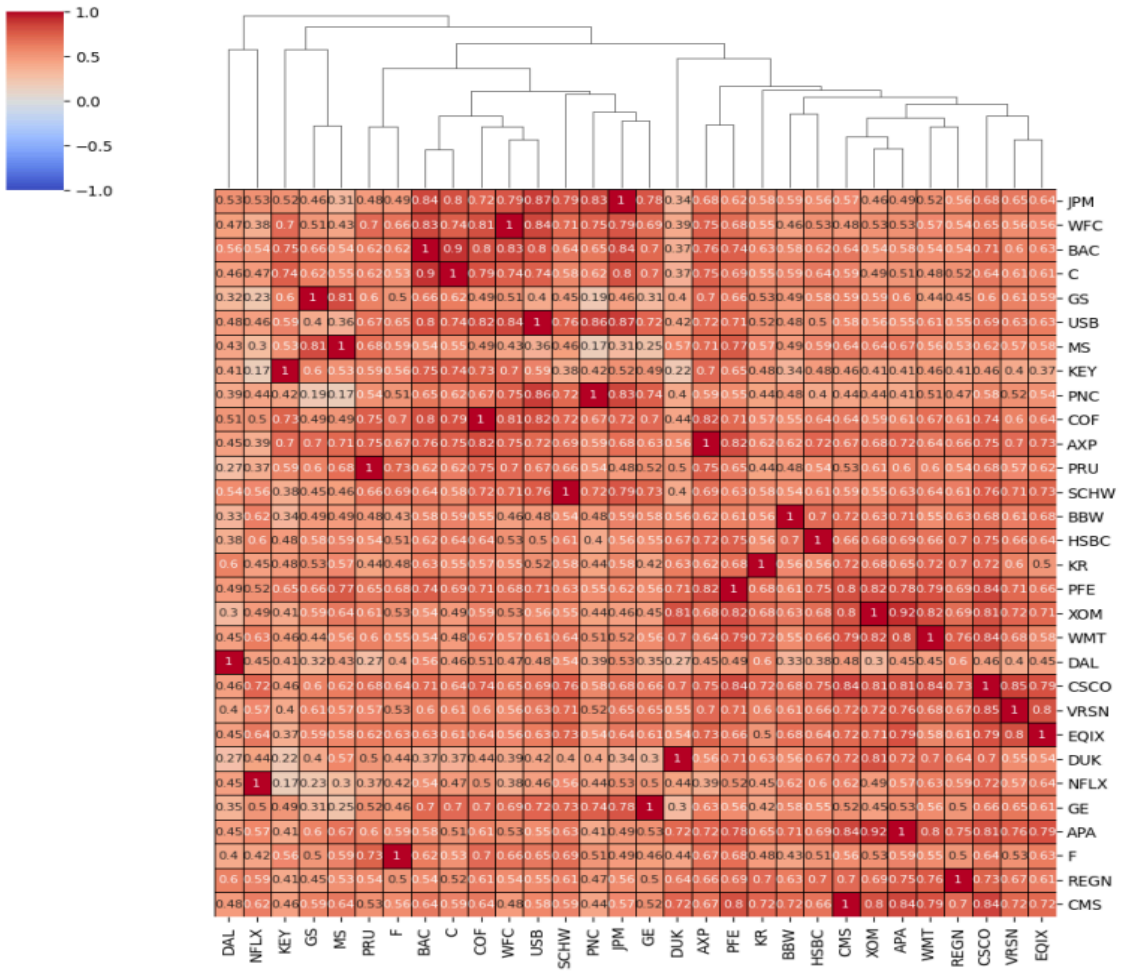


Figure 3: Hierarchical Clustering Dendrogram

The final clustering dendrogram shows that securities belonging to the same sector are highly correlated in terms of both returns and correlation portraits. Thus, this observation gives the optimum number of securities that should be held in one portfolio so that good diversification can be achieved. These findings are important to guide our asset selection process for the multi-armed bandit algorithm to create a diverse and balanced portfolio that reflects the underlying correlation structure of the market.

Step 5 & 6

Given that our correlation and cluster analysis have been carried out in the previous section, let us now proceed with the multi-armed bandit algorithm implementation. We will first start with the Upper Confidence Bound (UCB) algorithm, which will give us at least some level of exploration on our actions since we're still quite uncertain about our action value estimates.

For a standard ε-greedy choice of an action, one has to explore non-greedy actions without being

particularly selective. To this end, we follow the next optimal choice given as follows:

$$A_t = \arg\max_a \left(Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}} \right)$$

Here, the action selection is updated by using the new term that reflects the "unexploration" of the action w.r.t. the steps completed. The parameter > 0 controls the extent of exploration. If = 0 (i.e., the action has never been tried), then act as a maximizing action, so that the ratio would not become infinite.
We will apply this process, which focuses on the exploration of unknown actions, in our application of stock-picking strategies. Let's now proceed with the implementation of the UCB algorithm using the daily return data of the cluster portfolio that we have selected. Before that, we first describe the UCB algorithm using the following pseudocode:

Algorithm 2: Upper Confidence Bound (UCB) Algorithm for Stock Picking

Input:

- k: Number of stocks
- T: Total number of trading days
- R[i, t]: Daily return of stock i on day t

Initialize:

- N[i] = 0 for all i ∈ {1, 2, ..., k} ▷ Number of times stock i has been selected
- Rtotal[i] = 0 for all i ∈ {1, 2, ..., k} ▷ Total return obtained from stock i

For each trading day t = 1, 2, ..., T do:

1. **Calculate UCB values**:
    - For each stock i ∈ {1, 2, ..., k} do:
        - If N[i] = 0:
            - UCB[i] = ∞ ▷ Ensure unselected stocks are considered
        - Else:
            - $\text{UCB}[i] = \frac{R_{\text{total}}[i]}{N[i]} + c\sqrt{\frac{\ln t}{N[i]}}$ ▷ Modified UCB formula
2. **Select the stock with the highest UCB value**:
    - S = arg max UCB[i] ▷ Choose the stock with the maximum UCB value
3. **Observe the daily return of the selected stock**:
    - r = R[S, t] ▷ Observe the daily return of stock S
4. **Update the counts and total returns**:
    - N[S] = N[S] + 1 ▷ Increment the count for the selected stock
    - Rtotal[S] = Rtotal[S] + r ▷ Add the observed return to the total return

End For

Return: The stock with the highest average return.

Step 7 & 8

For comparison purposes, we also implemented the Epsilon-greedy algorithm. It is another effective alternative that balances between exploration and exploitation strategies. Both strategies can be applied together. The Epsilon-greedy algorithm balances exploration and exploitation by providing an epsilon probability of selecting the next action randomly rather than choosing the action that has the highest expected reward always.

How the Epsilon-greedy algorithm works:
- With probability $(1 - epsilon)$, the algorithm exploits existing knowledge by choosing the action with the highest estimated value (exploitation step).
- With probability $epsilon$, the algorithm explores by choosing a random action uniformly (exploration step).

The choice of $epsilon$ will determine the trade-off between exploration and exploitation. Larger $epsilon$ implies more exploration and more flexibility in a non-stationary environment, but lower $epsilon$ favours exploitation.

Algorithm 3: Epsilon-Greedy Algorithm for Portfolio Selection

1. **RunEpsilonGreedy(R, K, ε, α, N)**
2. **Input**:
   - `R[i, t]`: Matrix of asset returns
   - `K`: Number of assets
   - `ε`: Exploration probability
   - `α`: Step size
   - `N`: Number of trials
3. **Initialize**:
   - `Q ← zeros(K)` ▷ Values for each asset action
4. **For each trading day t = 1, 2, ..., T do**:
   - If `random() < ε` then:
     - `at ← random choice(K)` ▷ Exploration step
   - Else:
     - `at ← arg max Q[a]` ▷ Exploitation step
   - Observe reward `rt` for chosen asset `at`
   - `Q[at] ← Q[at] + α(rt - Q[at])`
5. **End For**
6. **Return Q**

Step 9

We compared and analyzed the outcomes of the results of the UCB and Epsilon-greedy algorithms with the conclusions given in Ho's paper. Our comparison of cumulative rewards, as presented in Figure 4, shows that the algorithm Epsilon-greedy algorithm performed slightly below UCB although both were strongly correlated with each other. The portfolio average performed much better: it had a lower level of volatility, and even higher overall returns.

Optimal Action Frequency, in Figure 5, shows how often the algorithms selected the best possible actions. Both UCB and Epsilon-greedy algorithms showed comparable average Optimal Action Frequency. They were equally effective in choosing the best actions even during short-term volatility.

From the results, the algorithm was slightly underperformed by the Epsilon-greedy algorithm. Still, both algorithms worked as expected. The algorithms generated comparable cumulative rewards. Even so, the average portfolio had the best performance.

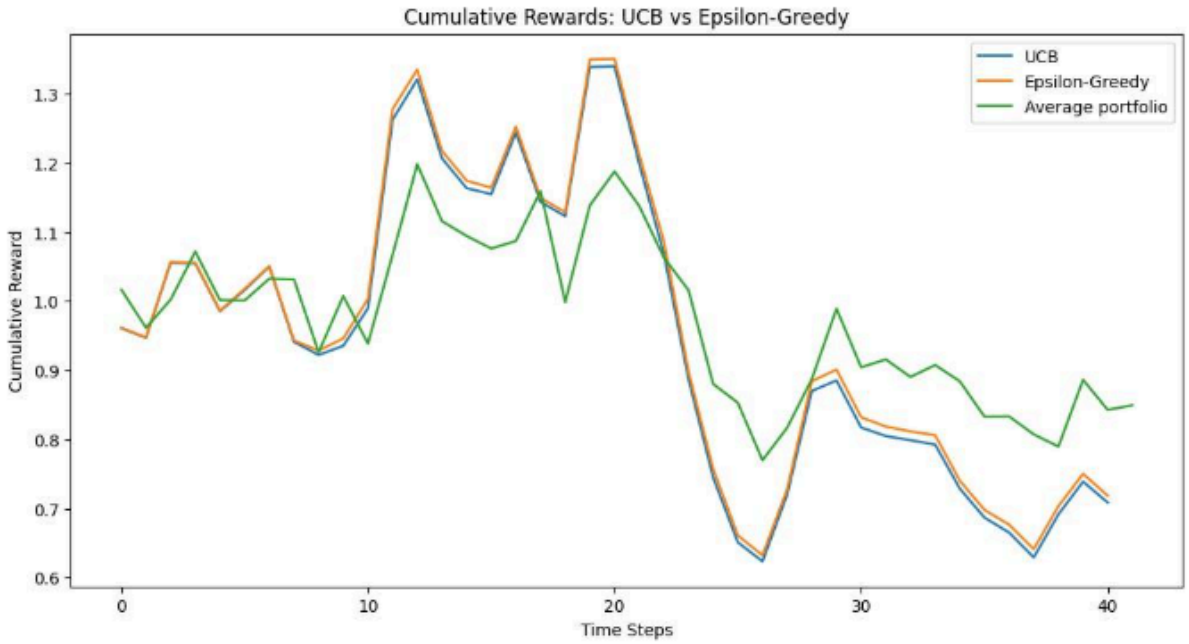


Figure 4: Comparison of Cumulative Rewards

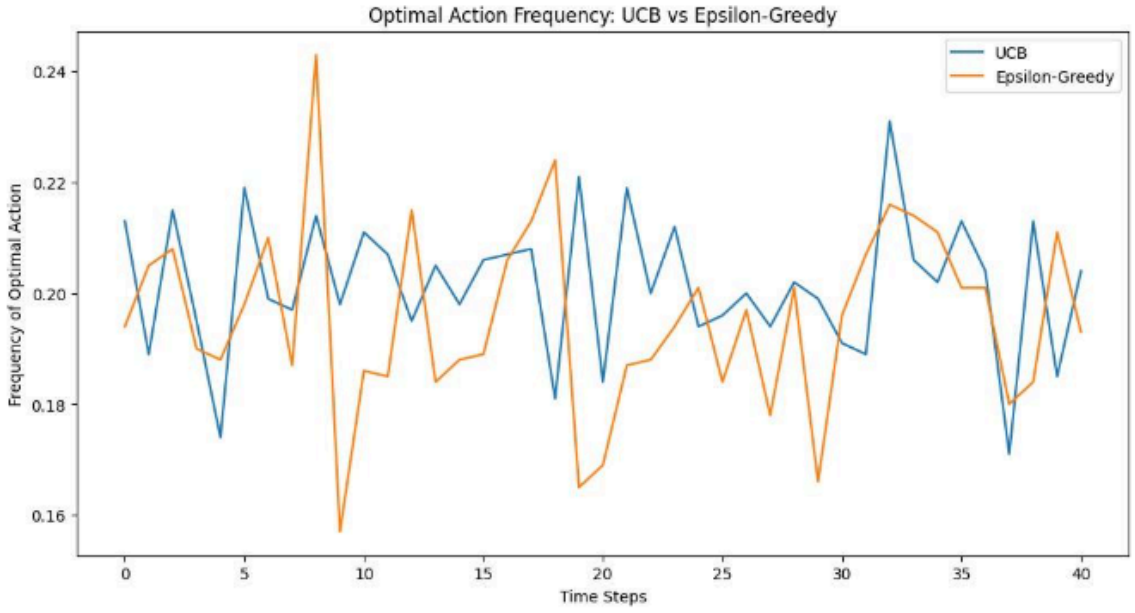


Figure 5: Comparison of Optimal Auction frequency

The last metric we considered is the comparison of the cumulative returns as shown in Figure 6.

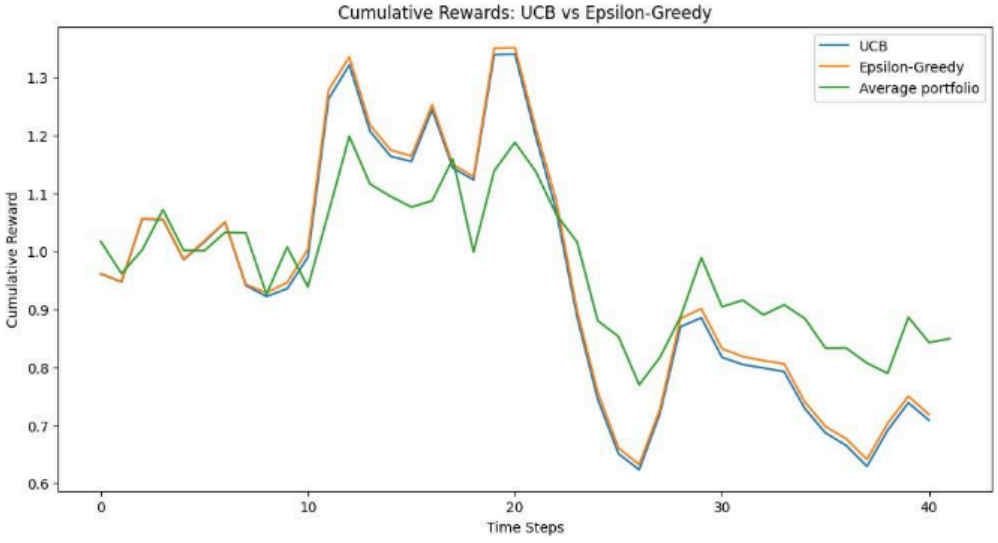


Figure 6: Comparison of Cumulative Return

On a graph it is obvious, that in all experiments with regard to the portfolio all algorithms finish with the negative results (Cumulative Return < 1). Two algorithms, which are showing better performance in comparison with the portfolio at the start, have the enormous draw-downs (> 60%). Portfolio shows the

smaller levels of the volatility, draw down and finishes better with the positive cumulative return at the moment of the stopping rule used.

Summary and Conclusion:

- September-October 2008. The period happens to coincide with the peak phase of the Global Financial Crisis, where the prices for stocks in the financials, banks, and in particular the real estate sector sank to very low levels. Price movements were highly extreme.

This obviously impacts the model performance along multiple dimensions.

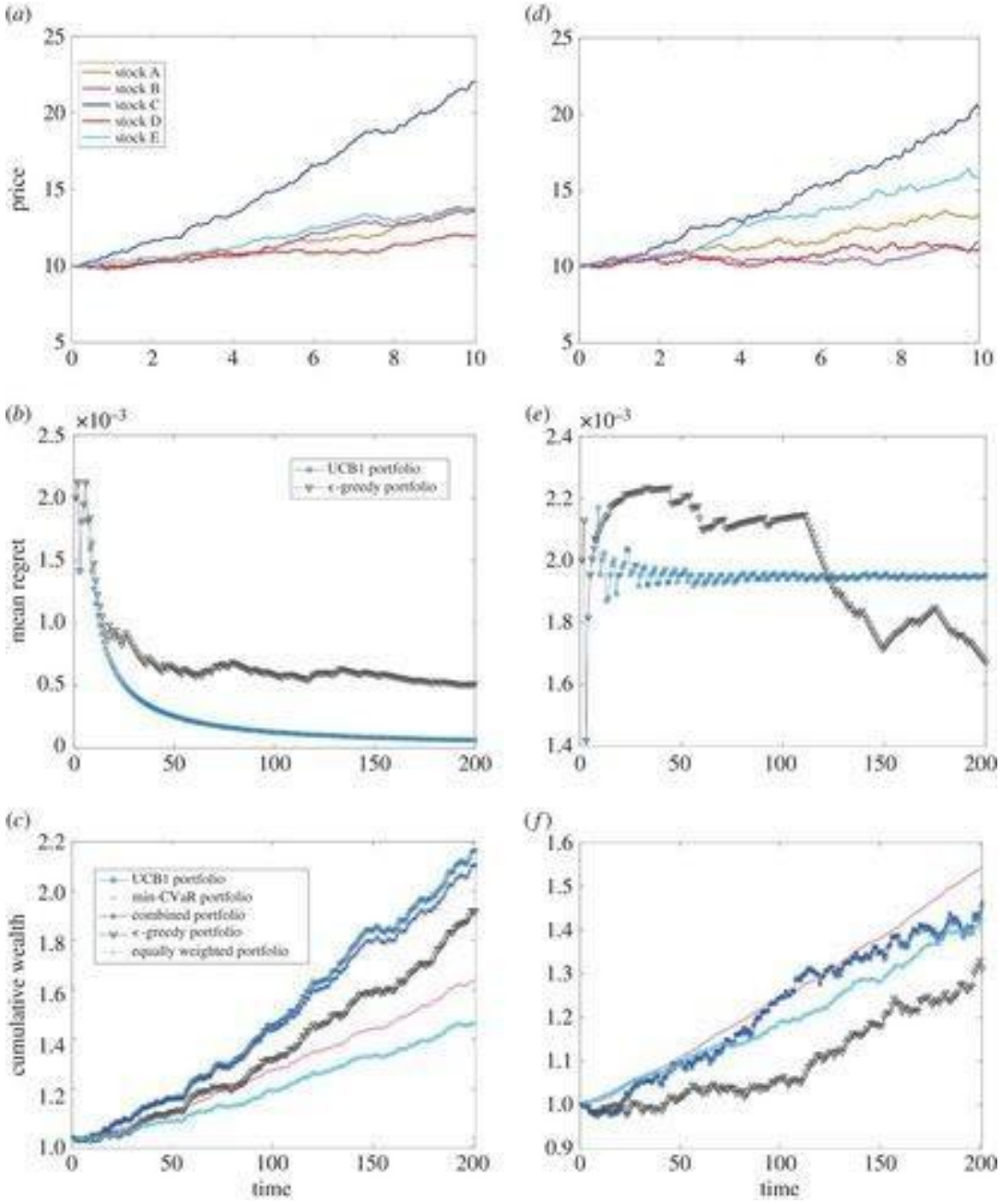
Financial markets turned increasingly volatile. Algorithms in our decisions such as UCB and Epsilon-greedy experienced higher variance.

Both balance exploration and exploitation in some way, differing over how that might be accomplished. Randomness introduced via epsilon-greedy was sometimes a few percentage points better than UCB, depending upon whether policy selection is actually being driven based on the confidence intervals of its actions.

The average using this much more diversified policy had much better average outcomes during this time, particularly considering such significant fluctuations were as often negative as positive during the period, stable for a reason: most of the portfolio can and will stabilize against major shock.

We also compare our results with Ho's paper in Figure 7 of the "Simulation Results" section.

Again, this analysis shows us how market conditions affect algorithmic performance and underlines that diversification is the secret to stable, long-run results.



The UCB approach tends to produce, more often than not, the highest cumulative wealth with portfolio selection but does not maintain consistent performance variation; on the other hand, a risk-aware portfolio performs lower in terms of the cumulative wealth but delivers consistently stable results with a very low variation. In balance, this would optimize maximization of returns while taking minimum risk. Evidently, UCB1 loses effectiveness in the very volatile markets where a risk-aware portfolio is able to outperform. The parameter λ is very critical and needs to be changed in accordance with market conditions so that the trade-off between risk and reward can be optimized.

Comparing Observations of Ho's Conclusions with Our Results :

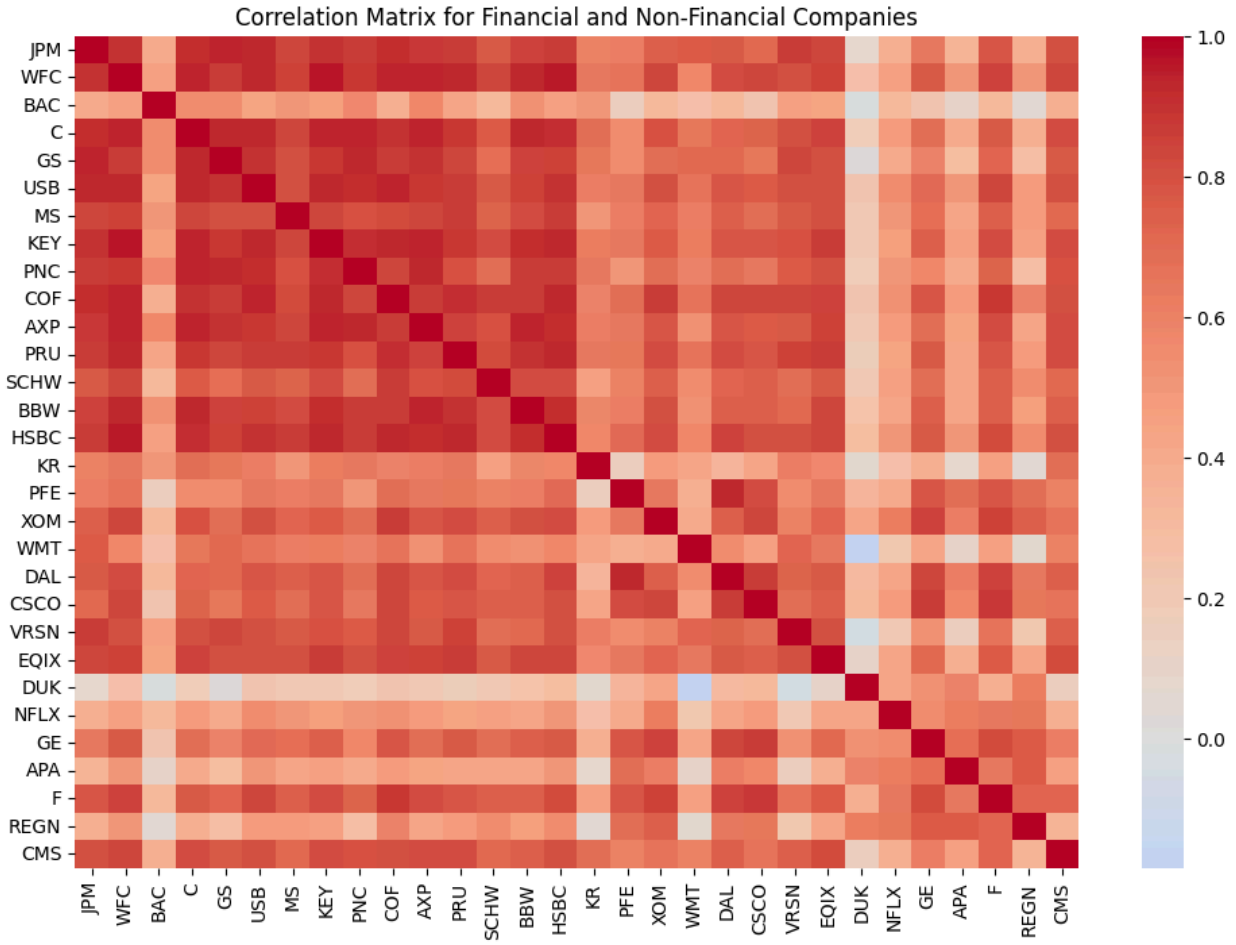
1. Unlike Ho's analysis, the strategy UCB produced the highest cumulative return; however, in our results, the average portfolio was more profitable.

2. The peculiar features of the period we had, one that witnessed the Global Financial Crisis, affected our model's performance to a good extent. This period being of extremely high volatility had made the UCB strategy not so effective, and hence, vindicated the conclusion of Ho that for periods of very high volatility UCB is not as effective.

These lessons teach us to take into account market conditions when we choose and apply different strategies for portfolios.

Step 10

We updated the 30 data series according to the list. Member C imported and structured the data of 15 financial companies, and Member A did the same for 15 non-financial companies. The selections were consistent with the previous ones. Member B then merged these series into a single data structure and computed the returns for the specified time period.



The result of correlation matrix for these for both financial and non-financial companies are given as:

```
all_returns.head()
```

Date	JPM	WFC	BAC	C	GS	USB	MS	KEY	PNC	COF	...	CSCO
2020-03-03 00:00:00+00:00	-0.051462	-0.055159	-0.010152	-0.037579	-0.055209	-0.028835	-0.035535	-0.037525	-0.044785	-0.044750	...	-0.006011
2020-03-04 00:00:00+00:00	0.071197	0.023063	-0.017949	0.035972	0.033673	0.026102	0.020264	0.024709	0.029611	0.018919	...	0.049304
2020-03-05 00:00:00+00:00	-0.041141	-0.050691	-0.067885	-0.057872	-0.047576	-0.047667	-0.015047	-0.049061	-0.042540	-0.058577	...	-0.044648
2020-03-06 00:00:00+00:00	-0.024338	-0.039955	-0.022409	-0.034809	-0.032793	-0.029881	-0.014360	-0.051680	-0.069462	-0.017610	...	-0.012973
2020-03-09 00:00:00+00:00	-0.091925	-0.147024	-0.151862	-0.161717	-0.112043	-0.103915	-0.056107	-0.135455	-0.182246	-0.103728	...	-0.056862

5 rows × 30 columns

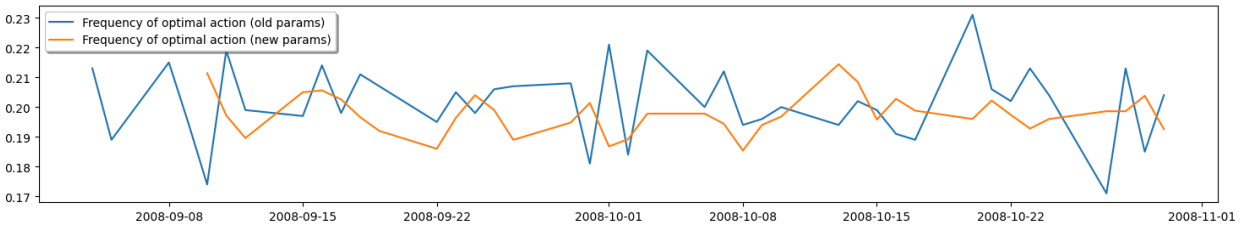
Step 11

1. Run UCB Algorithm Again

By using the parameters of UCB algorithm, we were now able to balance exploration vs. exploitation more effectively

- High EPSILON (0.1): Increased the rate by which understudied companies could be explored, thus, a smoother return curve that led to better long-run stability
- Low ALPHA (0.7): Reduced sensitivity by balancing historical returns with newer data and decreasing reactions by short-term volatility.
- Long Holding Period (HOLD=5): It reduced noise from short-term fluctuations and ensured consistent returns.
- Increased UCB Weight (UCB_WEIGHT=2.0): It encouraged exploration of less certain equities, which enhanced long-term stability.

These changes ensured stable performance, balancing exploration and exploitation, and returning consistent results over the long term.



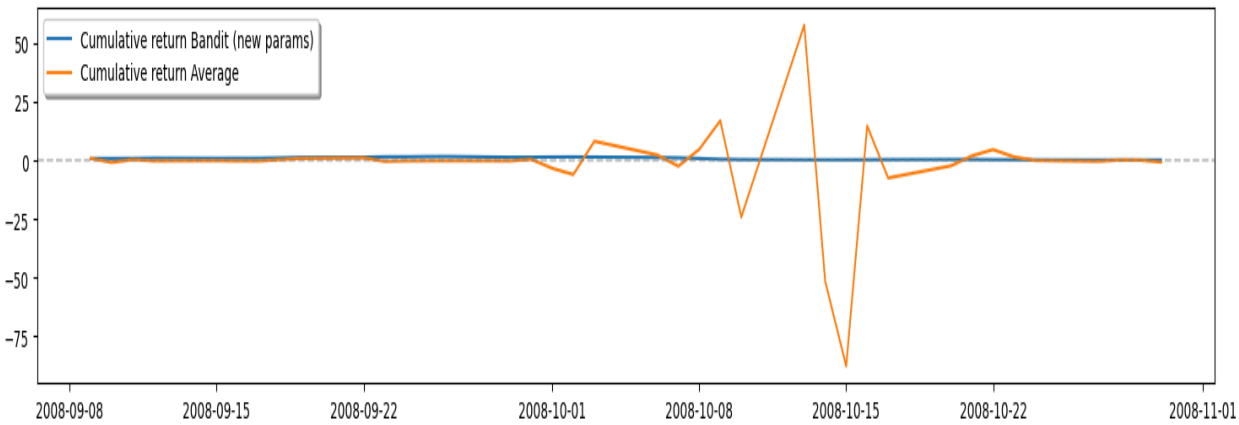
2. Repeat Epsilon-greedy Algorithm

By altering the parameters of the Epsilon-greedy algorithm, we see a drastic difference in performance and stability:

- Default Parameters (epsilon=0.9, alpha=0.9): High exploration and learning rates make the model very volatile, especially from the week of September to that of October. Exponential swings in returns were quite common because the model tried out so many new combinations of stocks.

- Epsilon 0.1, Alpha 0.7: Reduced exploration and learning with high return stocks led to a more stable cumulative return curve, as would be expected with less volatility and returning more consecutively, especially past October.
- Holding Period of HOLD=5: The extension of the holding period smoothed returns further and acted as a buffer to short-term variability.

- Range of Optimal Action Frequencies: It was balanced exploitation and exploration much better with adjusted parameters; thus, the range of optimal action frequencies increased.



Bibliography

[1] Huo, Xiaoguang & Fu, Feng. (2017). Risk-Aware Multi-Armed Bandit Problem with Application to Portfolio Selection. Royal Society Open Science. 4. 10.1098/rsos.171377.

[2] Yahoo Finance. (2024). Bitcoin USD (BTC-USD) Historical Data. (https://finance.yahoo.com/)

[3] UCB Algorithm. https://people.maths.bris.ac.uk/ maajg/teaching/stochopt/ucb.pdf