**Portfolio Analysis and Visualization**

*A Project Report Submitted*

*to*

**MANIPAL ACADEMY OF HIGHER EDUCATION**

*For Partial Fulfillment of the Requirement for the*

*Award of the Degree*

*Of*

**Bachelor of Technology**

*in*

**Computer and Communication Engineering**

*by*

**Urja Shah, Kadali Praharshini, Utkarsh Shukla**

**210953XXX, 210953XXX, 210953XXXX**

****

**November 2023**

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| **Sl No.** | **CONTENTS** |
| 1. | Introduction |
| 2. | Literature Survey |
| 3. | Methodology |
| 4. | Results |
| 5. | Conclusion |
| 6. | References |

**INTRODUCTION**

In portfolio management and risk assessment, advanced analytics help uncover valuable insights from financial data to improve investment portfolios and reduce risks.

The project typically kicks off by gathering and refining various data sets covering historical market trends, asset performances, economic indicators, and sometimes external factors like geopolitical events. Data mining techniques are then used to uncover patterns, and correlations. Predictive modeling is a key step to predict future market behavior, asset prices, and portfolio performance. This involves using methods like learning algorithms, regression analysis, and time series forecasting to make educated predictions.

Assessing risk is crucial. Models are built to measure and handle different risks such as market, credit, and operational risks. These models help gauge the potential impact of negative events on the portfolio and develop strategies to protect against such risks. The project results often include practical insights for optimizing portfolios, deciding on asset allocations, spotting undervalued or overvalued assets, and constructing portfolios that aim for specific risk and return profiles.

Throughout the process, it's vital to thoroughly check the accuracy and reliability of the models to ensure their usefulness in the real world. Also, using effective visualization methods is important to clearly communicate findings and recommendations to stakeholders and decision-makers.

Our project utilizes historical stock data fetched from Yahoo Finance to perform portfolio optimization and visualization. It allows users to create custom investment portfolios by specifying stock weights and compares these portfolios to randomly generated ones on a risk-return plot. The script calculates portfolio returns, risks, and visualizes the efficient frontier to help users make informed investment decisions. Additionally, it includes a segment to calculate the mean absolute error between predicted and actual stock returns, aiding in assessing the accuracy of return predictions.

Decision Tree Classifier is used to analyze a stock portfolio's historical data. It evaluates daily returns of specific stocks, predicts future returns, and suggests an optimal portfolio based on user-defined weights and the classifier's predictions. The decision tree visually represents the criteria for stock selection, aiming to assist in making informed investment decisions.

Clustering performs a financial analysis by creating portfolios comprising various stocks like Apple, Google, Netflix, Amazon, and Tesla. It then groups these portfolios into clusters based on their expected returns and risk (measured by standard deviation). Using KMeans clustering, it categorizes portfolios with similar risk-return profiles and visualizes these clusters for better understanding.

Overall, this project aims to use data-driven insights to make smarter investment choices, maximize returns, and effectively manage risks in a constantly changing financial environment.

**LITERATURE SURVEY**

**Meng Yuam et.al.,[1]**

This paper is to propose a minimax portfolio selection model with an interval expected return rate. The paper aims to describe the return and risk of a portfolio more accurately and provide a better strategy than Markowitz's model. Classification and regression are the two data mining techniques used for non-linear to solve problems in financial analysis. A new method is proposed to determine the interval of expected return for a portfolio. The model assumes that returns from all securities are normally distributed, which is only a hypothesis and has not been empirically proven.

**GouRong Hu et.al.,[2]**

This paper is to explore the application of data mining technology in portfolio optimization. It discusses how data mining can be used to analyse and manage a large amount of portfolio optimization data, providing new ideas and solutions for portfolio optimization problems. Data mining techniques involve a combination of data extraction, sorting, classification, clustering, prediction, and visualization to analyse and extract valuable insights from large datasets. The key advantages involve enhanced portfolio performance, integration with investment, and support for enterprise business.

**Siyamak Goudarzi et.al.,[3]**

The paper discusses a model for clustering and optimizing portfolios in the Tehran Stock Exchange using data mining algorithms. The goal of the research is to develop a portfolio that considers the behaviour of investors in risk-taking and aims to increase output for investors. This paper mentions two data mining algorithms used in the research: K-means clustering and Self-Organizing Maps (SOM). These algorithms are used to optimize the formation of investment portfolios by reducing risk and increasing output. The paper identifies several potential research gaps, including the need for longer time intervals, alternative clustering and optimization algorithms, and ranking methods.

**Nishant Dudhwala et.al.,[4]**

This paper is to propose a ARMA (autoregressive-moving- average) algorithm is used. This paper aims to predict the output of an input which is nonlinear in nature. It proposes to create a system which can store the data of shares in a database and then suggest predictions of the same with respect to the indices of the market. It also uses neural networks which will determine the stock price during a particular incident. The authors believe the next step of progress is not just being able to buy or sell stocks but also predict values of the shares in the market.

**K. Senthamarai Kannan et.al.,[5]**

This paper builds an algorithm to predict the stock market trends on 5 of the most important indices which predict the market, Typical Price (TP), Bollinger Bands, Relative Strength Index(RSI), CMI, Moving Average(MA). These five indices if increase or decrease, if applied to various formulae can predict the price of the stock by the closing time. Some of these formulae include a binomial test as well as a Time Series Analysis to find the price of the stock. These algorithms are founded which help investors discover hidden patterns from historic data that have helped them give predictable capability in investment decisions. .

**Jaydip Sen et.al.,[6]**

This paper proposes building a portfolio of capital assets using the predicted prices to achieve the optimization between its return and risk. It depends on the model long-and-short-term memory (LSTM) which was build after 5 months of portfolio construction. It carries an carried out an analysis of the time series of the historical prices of the top five stocks from the nine different sectors of the Indian stock market and then builds an optimum portfolio. The predicted and the actual returns of each portfolio are found to be high, indicating the high precision of the LSTM model.

**Salman Zafar et.al.,[7]**

This paper focuses on providing investment portfolio guidelines in the market, utilizing data mining techniques to extract meaningful patterns from business data. The study emphasizes attracting foreign investment and improving stock market performance. It employs data mining methods for comparative analysis and decision support in portfolio selection. The research aims to offer investors valuable guidance in selecting optimal portfolios based on comprehensive data analysis of share prices across various sectors. While the paper doesn't specify drawbacks or research gaps, it cautions against over-concentration in specific sectors or securities. The importance of diversification in investment strategy is underscored, along with the risk associated with over-concentration. Overall, the paper provides a framework for informed investment decisions in the market.

**Yilin Ma et.al.,[8]**

The paper addresses the critical field of portfolio optimization, aiming to enhance investors' profitability and stability. Employing three deep neural networks (DNNs) - DMLP, LSTM, and CNN - the study combines modern portfolio theory with deep learning technology. These DNNs predict future stock returns and quantify their associated risks. The portfolio optimization models integrate these predictive returns and semi-absolute deviation of predictive errors. Experimental results utilizing component stocks of China's Securities 100 Index showcase the superiority of the DMLP-based model, particularly at different desired portfolio returns. Notably, higher desired returns further enhance this model's performance. It underscores the ongoing need for improved portfolio optimization models to achieve higher returns at equivalent risk levels. This research provides valuable insights into the potential of DNNs in revolutionizing portfolio optimization, highlighting the continuous quest for advancements in this critical investment management field.

**Monica Tiria et.al.,[9]**

The paper aims to develop a system that assists traders in optimizing their stock market portfolio for short or medium-term investments. The system employs sentiment analysis, characteristics of investments, and the trader's confidence level to classify and quantify risk. Additionally, it utilizes an ontology to describe market status and expectations based on future events, and to determine relationships and correlations between news articles. The system also assesses the impact of an equal percentage of positive and negative news articles on stock price movements. Algorithms used are Text mining, ontology, sentiment analysis, trust models, risk management. The proposed multi-agent system employs various techniques, including sentiment analysis, ontology, trust models, and risk management, to select the optimal mix of investments that minimizes risk and maximizes gains in a stock portfolio. A prototype was developed and validated in the research. The paper doesn't address potential implementation challenges or practical considerations for deploying the multi-agent system in real-world trading environments, specially for long-term investing.

**Susan George et.al.,[10]**

This paper employs network-based data mining to analyze stock market dynamics, identifying pivotal players and systemic risks. Using daily time series data, it constructs a network, emphasizing both structural aspects and influential players. Portfolio analysis identifies critical market stability sectors. The paper highlights a shift in financial risk analysis, post-2008 crisis, towards understanding systemic risks through network analysis. The example of Bear Stearns illustrates the potential cascade of failures due to its pivotal position. Key player identification and portfolio analysis inform risk management. It emphasizes the crucial need to understand interdependencies for addressing systemic risk in financial markets.

**METHODOLOGY**

Algorithm 1: Frontier Efficient

1. **Importing Libraries and Data Retrieval:**
   * pandas: For data manipulation and analysis.
   * numpy: For numerical operations.
   * yfinance: For fetching historical stock data.
   * It defines a list of stocks and retrieves historical stock data from Yahoo Finance for these stocks, focusing on adjusted closing prices.
2. **Calculating Returns and Statistics:**
   * Daily returns are computed from the adjusted closing prices.
   * Mean returns and the covariance matrix are calculated from the return data, providing essential inputs for portfolio analysis.
3. **User-Specified Portfolios:**
   * The code asks the user to specify the number of portfolios they want to create.
   * For each portfolio, users input weights for each stock.
   * It normalizes these weights to ensure they sum up to 1, then calculates the portfolio's return and standard deviation. These details are stored.
4. **Random Portfolios:**
   * The code generates a specified number of random portfolios (100 in this case).
   * Similar to user-specified portfolios, it computes returns and standard deviations for these random portfolios and stores the results.
5. **Efficient Frontier Visualization:**
   * Using Matplotlib, the code creates a plot illustrating the efficient frontier.
   * User-specified portfolios are depicted as green circles, while random portfolios appear as blue crosses.
   * The best and worst user-specified portfolios are highlighted with red circles, with labels displaying their weights and the number of stocks included.
   * The best random portfolio is marked with a red cross and labelled similarly.
6. **Error Calculation (Assuming Predicted Returns):**
   * The code assumes the existence of a variable named **predicted\_returns**.
   * It computes absolute errors between the predicted returns and the mean historical returns.
   * Finally, it calculates the mean absolute error to evaluate the accuracy of the predictions.

Algorithm 2: Decision Tree

1. **Stock Data Retrieval and Processing:**
   * Define a list of stock symbols and ask the user for the weight of each stock.
   * Use Yahoo Finance (yfinance) to get historical stock data within a specified date range.
   * Calculate daily returns based on the adjusted closing prices and organize the data into a table.
2. **Classification and Decision Tree Creation:**
   * Set a threshold to categorize returns as positive or negative.
   * Prepare the data for the classifier by separating features (X) and the target (y).
   * Create and teach a Decision Tree Classifier using a method called entropy.
3. **Visualizing the Decision Tree:**
   * Use Matplotlib to draw and display the Decision Tree made by the classifier.
   * The tree shows labels indicating stocks and decision points.
4. **Portfolio Optimization:**
   * Initially, find the best stocks based on user-given weights and the Decision Tree predictions indicating positive outcomes.
   * Then, factor in risk assessment by calculating volatility (how much returns vary) for each stock. Add this as a feature to the dataset (X).
   * Re-educate the Decision Tree with this updated data.
   * Reassess the best stocks considering both positive predictions and user weights, along with volatility, aiming for better risk-adjusted returns.
5. **Printing the Best Portfolio:**
   * Show the final recommended stocks based on Decision Tree predictions and risk-adjusted returns.
6. **Risk Assessment:**
   * The code contains comments hinting at different aspects of the Decision Tree model, such as splitting conditions, entropy, samples, class values, decision paths, tree depth, and threshold values. These describe properties of the trained Decision Tree. Risk assessment by is done by considering volatility as an extra factor in choosing stocks. This aims to improve stock selection based on both the Decision Tree predictions and the user-provided weights, potentially leading to better risk-adjusted returns.

Algorithm 3: KMeans (Clustering)

1. **Stock Data Retrieval:**
   * The code fetches historical stock data for Apple (AAPL), Google (GOOGL), Netflix (NFLX), Amazon (AMZN), and Tesla (TSLA) from Yahoo Finance.
2. **Calculating Returns:**
   * Daily returns for each stock are calculated by finding the percentage change in their adjusted closing prices.
   * Any resulting NaN values, often occurring due to return calculations, are removed.
3. **Expected Returns and Standard Deviations:**
   * The code computes the average returns and standard deviations for each stock based on their daily returns.
   * These calculated values are organized into a DataFrame (risk\_return\_df) to portray the risk-return profile of each individual stock.
4. **Portfolio Generation:**
   * It generates 100 sets of random weights, which dictate how funds are allocated among the selected stocks within a portfolio.
   * The user is prompted to input the desired number of clusters for K-means clustering.
5. **Portfolio Metrics Calculation:**
   * For every portfolio i.e, determined by its set of weights, the code calculates the expected return and standard deviation using the given weights, expected returns, and covariance matrix of returns.
6. **K-Means Clustering:**
   * Utilizing the calculated portfolio metrics using expected return and standard deviation, the code performs K-means clustering.
   * The number of clusters is determined by the user input, and each portfolio is categorized into a cluster based on its risk-return characteristics.
7. **Visualization:**
   * The code generates a scatter plot to visualize the clusters, assigning different colours to each cluster.
   * The x-axis represents standard deviation, while the y-axis displays expected return.
   * This visual representation helps in identifying clusters with diverse risk-return profiles.
8. **Cluster Information:**
   * Finally, the code displays the portfolios included within each cluster, demonstrating which portfolios belong to specific risk-return groups. This methodology aids investors or analysts in comprehending how various combinations of stock portfolios can be grouped based on their risk and return characteristics. It offers insights into diversified investment strategies by identifying and categorizing portfolios with different risk-return profiles.

Algorithm 4: Monte Carlo

1. **Data Collection:**

* Stock Data Retrieval: Historical stock data for specified symbols is acquired using a specialized PriceHistory client or loaded from a stored CSV file if previously obtained.

2. **Data Preparation:**

* Data Cleaning and Transformation: The raw stock data undergoes thorough processing to ensure accuracy and consistency for subsequent analysis.
* Returns Calculation: Logarithmic returns are computed based on closing prices to assess the historical performance of the stocks.

3. **Portfolio Generation:**

* Random Portfolios: Multiple portfolios are created with diverse weight distributions among chosen stocks to simulate various investment scenarios.
* User Input: Users have the option to input their own portfolio weight allocations for comparison and evaluation purposes.

4. **Metrics Calculation:**

* Portfolio Metrics: Essential metrics like expected returns, volatility, and the Sharpe Ratio are computed for each portfolio, providing insights into their performance and risk profiles.

5. **Portfolio Evaluation and Optimization:**

* Portfolio Evaluation: Comparative analysis is conducted among different portfolios (randomly generated, minimum volatility, and Sharpe Ratio-optimized) based on their computed metrics to assess strengths and weaknesses.
* Portfolio Optimization: Utilization of optimization techniques, such as the scipy.optimize module, to optimize portfolio allocation by maximizing the Sharpe Ratio while considering constraints like maintaining a total weight sum of 1.

6. **Visualization and Analysis:**

* Scatter Plot: Visualization of simulated portfolios on a scatter plot illustrates the relationship between returns and volatility, with color gradients indicating the Sharpe Ratio.
* Comparison: Comparative analysis presenting the user's portfolio alongside randomly generated ones and optimized portfolios aids in decision-making and strategy assessment.

**RESULTS**

Algorithm 1: Frontier Efficient

1. **User-specified Portfolios:** Users provide the weight allocation for each stock in a portfolio. The code standardizes these allocations, calculates the portfolio's return and risk (standard deviation), and saves these measures. It identifies the portfolios with the highest and lowest returns among the user-specified ones.
2. **Random Portfolios:** Random portfolios are generated by assigning random weightings to the stocks. Similar to the user-input portfolios, it computes the returns and risks for these random combinations.
3. **Efficient Frontier Visualization:** The plot illustrates various portfolios on the risk-return spectrum. Portfolios created by user input are shown as green circles, while randomly generated ones are marked as blue crosses. It highlights the best and worst user-defined portfolios using red circles, providing details about their stock weightings and count.
4. **Error Calculation (Predicted Returns):** Given predicted returns and historical average returns, the code computes the absolute discrepancies between them. It then calculates the mean absolute error, serving as an indicator of how accurately the predictions align with historical data.

The visualization of the efficient frontier aids in evaluating portfolios based on risk and potential returns. This graphical representation allows for the identification of portfolios that offer higher returns considering a particular risk level and facilitates comparison with randomly generated portfolios.

The mean absolute error serves as a metric for assessing the accuracy of predicted returns by measuring their divergence from historical mean returns. In essence, the code offers users an avenue to explore, compare portfolios, and comprehend their risk-return dynamics, all while evaluating predicted returns against historical data.

Algorithm 2: Decision Tree

1. **Decision Tree Visualization:** The code creates a visual representation of a Decision Tree using a plotting function from the **sklearn** library. This tree helps in deciding whether to buy or not buy a stock based on historical returns and possibly other factors like volatility.
2. **Initial Portfolio Identification:** Initially, it selects the best stocks based on user-provided weights and the Decision Tree predictions. It looks for stocks where the model predicts positive returns ('Buy') and the user has allocated a positive weight.
3. **Adjusting for Risk in Portfolio Identification:** After calculating volatility for each stock (a measure of risk), it adds this information to the dataset and retrains the Decision Tree. It then refines the selection of stocks by considering not just positive predictions and user weights but also volatility. This aims to suggest stocks with potentially better risk-adjusted returns.
4. **Displaying the Results:** It displays the recommended stocks for investment based on initial predictions and the risk-adjusted considerations. Stocks meeting the criteria (predicted as 'Buy', positive weights, and lower volatility) are suggested for the portfolio.
5. **Interpreting the Results:** The output shows the selected stocks recommended for investment initially based on the Decision Tree's predictions and then considering risk. It may suggest a portfolio with stocks that not only show predicted positive returns but also demonstrate lower volatility, indicating reduced risk.

The result displays recommended stocks for investment based on the Decision Tree's predictions and adjustments made for risk assessment. The intention is to guide towards a selection that balances potential returns with a consideration for risk, aiming for a more informed investment strategy.

Algorithm 3: K-Means (Clustering)

1. **Stock Selection and Historical Data Retrieval:**
   * Choose a specific set of stocks for analysis.
   * Define the timeframe for historical stock data retrieval, including the start and end dates.
2. **Fetching Historical Stock Data:**
   * Utilize financial data retrieval tools like 'yfinance' to acquire historical stock data for the selected stocks within the defined timeframe.
3. **Calculation of Daily Returns:**
   * Compute the daily returns for each stock based on the acquired historical data.
   * Eliminate any instances of missing or incomplete data resulting from calculations.
4. **Generation of Random Portfolios:**
   * Determine the number of portfolios required for analysis.
   * Create random weight distributions for the chosen stocks within each portfolio, ensuring the sum of weights per portfolio totals 1.
5. **Computation of Portfolio Metrics:**
   * For each randomly generated portfolio:
     1. Calculate the expected return by weighing the individual stocks' expected returns within the portfolio.
     2. Determine the portfolio's standard deviation by considering the variance, which comprises weighted stock variances and covariances.
6. **Application of K-means Clustering:**
   * Consolidate the expected returns and standard deviations of the portfolios into a matrix format.
   * Employ K-means clustering to categorize the portfolios based on their risk-return profiles.
   * Determine the optimal number of clusters using techniques such as the elbow method or silhouette score, if required.
7. **Visualization of Clusters:**
   * Develop a scatter plot representing the portfolios, with the x-axis denoting standard deviation and the y-axis indicating expected return.
   * Use distinct colors or labels to visually highlight the clusters generated by K-means for clearer interpretation.
8. **Interpretation of Clusters:**
   * Analyze the resulting clusters to grasp their distinct risk-return characteristics.
   * Identify the constituent portfolios within each cluster and scrutinize their allocations across different stocks.
9. **Validation and Refinement Process:** Validate the obtained clusters and refine the analysis iteratively. Adjust parameters like the number of portfolios or selected stocks to enhance the clustering outcomes.

Algorithm 4: Monte Carlo

1. **Portfolio Metrics:**
   * Expected Returns: Computed for each portfolio, representing the anticipated average returns based on historical data.
   * Volatility: Demonstrates the fluctuation degree or risk associated with individual portfolios, showcasing the standard deviation of returns.
   * Sharpe Ratio: An indicator of risk-adjusted returns, aiding in assessing portfolio efficiency by considering risk relative to returns.
2. **Comparative Analysis:**
   * Portfolio Evaluation: Comparative assessments among various portfolios, including randomly generated sets, minimum volatility, Sharpe Ratio-optimized, and user-input allocations.
   * Performance Ranking: Evaluation of the user-provided portfolio against randomly generated ones in terms of Sharpe Ratio, positioning it relative to other portfolios.
3. **Visual Representation:**
   * Scatter Plot Visuals: Depiction of portfolios on scatter plots, plotting returns against volatility and using color gradients to illustrate the Sharpe Ratio. This visualization aids in comprehending the trade-offs between risk and returns.
4. **Optimized Portfolio:**
   * Optimal Allocation: Identification of the portfolio with the highest Sharpe Ratio obtained through optimization techniques, emphasizing risk-adjusted returns within specified constraints.
5. **Decision Support:**
   * Insightful Guidance: Provision of insights into portfolios offering superior risk-adjusted returns, facilitating informed decision-making regarding portfolio allocation and strategy.
6. **User Engagement:**
   * Analysis of User-Provided Portfolios: Evaluation and comparison of portfolios submitted by users against randomly generated and optimized sets, enabling users to gauge the effectiveness of their chosen allocations.

**CONCLUSION**

The utilization of diverse analytical tools has played a pivotal role. This project aimed to understand portfolio analysis and visualization by employing frontier efficient, k-means, decision tree, and Monte Carlo algorithms.

These methodologies provided a spectrum of insights. Frontier efficient analysis laid the groundwork for risk-return trade-offs, identifying portfolios on the efficient frontier. K-means clustering delineated distinct clusters in the portfolio space, revealing insights into asset groupings and behaviors. Decision tree analysis structured decision-making and assessed probable outcomes and associated risks. Monte Carlo simulations offered aid to risk assessment.

The integration of these algorithms enriched analysis depth and visualization clarity. Visualizing results through scatter plots and decision trees achieved a multidimensional understanding of portfolio dynamics.

However, each tool had its strengths and limitations. Frontier efficient analysis demanded precise inputs and assumptions despite depicting clear trade-offs. K-means clustering's insights relied on initial grouping assumptions and might overlook complex interdependencies. Decision trees could become complex with large datasets and many variables. Monte Carlo simulations were sensitive to injected randomness and assumptions.

In conclusion, these algorithms significantly contributed to portfolio analysis and visualization, enhancing understanding of risk, return, and diversification for better investment decisions. However, no single algorithm provided a complete solution. Their combined utilization created a robust framework for portfolio optimization in financial markets' dynamic landscape.

The accuracy of the algorithms is compared on the basis of return, volatility and sharpe ratio.

**REFERENCES**

<https://www.investopedia.com/terms/e/efficientfrontier.asp#:~:text=The%20efficient%20frontier%20is%20the,for%20the%20level%20of%20risk>.

<https://realpython.com/k-means-clustering-python/>

<https://scikit-learn.org/stable/modules/tree.html>

<https://tradewithpython.com/portfolio-analysis-using-python>