**Abstract**

In the face of today's complex economy and the diversification of personal financial management needs, we have developed an intelligent financial advisor system. The system uses the ExtraTreesRegressor model trained on the SCFP2022 dataset to accurately predict the user's risk tolerance and provide personalized portfolio recommendations. The prediction has an R² of 0.92, an MSE of 0.0090, an RMSE of 0.0950, and an F1 score of 0.7394 on the validation dataset, and the model performs well. And reflecting on the research results, personal assets and income, that is, financial stability, have the greatest impact on personal risk tolerance. It also integrates a variety of financial analysis tools, including convex optimization methods for personalized portfolio allocation and Monte Carlo simulation methods for forward-looking performance evaluation under different economic scenarios. Supplementary modules developed by team member, including market overview, individual stock analysis, and expenditure planning, jointly enhance the functionality of the system.

The intelligent advisory platform lowers the cost of giving the general public individualized investment advice by increasing the efficiency and convenience of financial services.

**Table of Contents**

[1.0 Introduction 4](#_Toc205128879)

[2.0 Problem Statement 5](#_Toc205128880)

[2.1 Lack of Future Planning and Scenario Simulation 5](#_Toc205128881)

[2.2 Lack of Quantitative Risk Assessment Tools 5](#_Toc205128882)

[2.3 Absence of Risk-based Intelligent Asset Allocation 5](#_Toc205128883)

[2.4 Need for cost-effective financial analysis 6](#_Toc205128884)

[2.5 The risk of financial fraud has increased 6](#_Toc205128885)

[3.0 Objectives and Outcome 7](#_Toc205128886)

[3.1 Portfolio allocation based on usre risk tolerance 7](#_Toc205128887)

[3.2 Portfolio Historical Performance 7](#_Toc205128888)

[3.3 Simulator Tool 7](#_Toc205128889)

[3.4 Other modules contributed by team members 8](#_Toc205128890)

[4.0 Literature Review 9](#_Toc205128891)

[4.1 AI Financial Advisor 9](#_Toc205128892)

[4.2 Development history of AI financial advisors 9](#_Toc205128893)

[4.3 Application of AI and ML in finance 10](#_Toc205128894)

[4.4 Portfolio optimization 10](#_Toc205128895)

[4.5 Challenges of existing AI financial advisory platforms 11](#_Toc205128896)

[4.6 Limitations of Traditional Financial Advisory 11](#_Toc205128897)

[5.0 Project Methodology 12](#_Toc205128898)

[5.1 Calculate Risk Tolerance through data sets 12](#_Toc205128899)

[5.2 Data Processing - Missing Value Handling 12](#_Toc205128900)

[5.3 Data Processing - One-Hot Encoding 12](#_Toc205128901)

[5.4 Feature Engineering 13](#_Toc205128902)

[5.5 From basic linear models to advanced integrated models 15](#_Toc205128903)

[5.5 Portfolio Recommendation 16](#_Toc205128904)

[6.0 Design and Development 21](#_Toc205128905)

[6.1 System Architecture and Design 21](#_Toc205128906)

[6.2 Database Design and Normalization 23](#_Toc205128907)

[7.0 Implementation 25](#_Toc205128908)

[7.1 UserPortrait Module 25](#_Toc205128909)

[7.1.1 Backend Implementation 26](#_Toc205128910)

[7.2 Risk Tolerance predicting Implementation 30](#_Toc205128911)

[7.2.1 Data processing 30](#_Toc205128912)

[7.2.3 model training 33](#_Toc205128913)

[7.3 Simulation Module 35](#_Toc205128914)

[8.0 Collection of data 41](#_Toc205128915)

[8.1 SCFP2022 Dataset 41](#_Toc205128916)

[8.2 Market Data 41](#_Toc205128917)

[8.3 Market Data Processing 42](#_Toc205128918)

[9.0 Analysis of result and Demonstration 43](#_Toc205128919)

[9.1 Model Evaluation and Data Visualization 43](#_Toc205128920)

[9.2 Result Reflection 46](#_Toc205128921)

[9.3 User Portrait and Asset Allocation Demonstration 47](#_Toc205128922)

[9.4 Multi-Portfolio Backest Demonstration 51](#_Toc205128923)

[9.5 Simulator Demonstration 53](#_Toc205128924)

[10.0 Conclusion 57](#_Toc205128925)

[Reference 58](#_Toc205128926)

1. Introduction

How to guarantee the maintenance and appreciation of assets while making sufficient retirement preparations has emerged as a significant subject of public concern in the current complicated global economic climate. The 2024 Global Retirement Index highlights that with one in five investors believing that even with $1 million in savings, they would still be unable to afford retirement [1]. Traditional financial consulting services now in use mostly rely on qualified experts to offer customized guidance. Even if the impact is substantial, some advisors charge flat fees ranging from $1,000 to $10,000 annually, depending on the complexity of services provided [2]. So, average investors find it difficult to afford due to the high cost and poor service penetration rate.

However, even though the available financial robot advisors have partially automated their services, they still struggle to offer thorough, dynamic, and customized scenario planning assistance when faced with complex scenarios like interest rate fluctuations, inflation shifts, and future economic uncertainty. And also, Investors have increased decision-making risks as a result of this circumstance as they frequently lack awareness of their own circumstances and enough data support when creating long-term strategies [3].

Based on the real-world issues, this project suggests an intelligent financial consulting system that uses financial and artificial intelligence technologies to offer users affordable, encrypted, and customized investment management and financial planning services.

The system fully analyzes the user's asset status and financial information, scientifically quantifies the user's risk tolerance to achieve dynamic asset allocation, and simulates the performance of investment portfolios under different economic scenarios. Provide market trends, AI-driven personalized consulting, and modules for news and educational resources.

Our system overcomes the limitations of conventional financial advising services by helping investors better understand market movements and achieve a balance between risk and return through scenario simulations and automated asset allocation recommendations, sustaining steady growth in a volatile economic climate. Additionally, it is anticipated to encourage the general people to become more familiar with investing and financial management and to reduce the bar for financial services.

2.0 Problem Statement

2.1 Lack of Future Planning and Scenario Simulation

People sometimes find it difficult to project their future financial condition effectively when making long-term investment or retirement plans, especially when considering different economic scenarios. They are often not equipped to analyze and model "what-if" possibilities, such as shifts in inflation, interest rates, and economic development. The lack of extensive data support and scenario modeling capabilities prevents users from making well-informed, long-term investment decisions. Users are left with uncertainty when they cannot foresee and prepare for varying economic situations, which hinders their capacity to develop practical financial well-being plans [3].

2.2 Lack of Quantitative Risk Assessment Tools

When making financial decisions, the majority of investors frequently lack awareness of their own risk tolerance and instead depend on gut feeling, past performance, or basic market data. Furthermore, a lack of personalization in many risk analysis tools may cause users to adopt investment strategies that are not in line with their true risk tolerance, which could lead to excessively conservative or risky investment outcomes and ultimately hinder the achievement of their long-term financial objectives. According to a 2024 report from the FINRA Investor Education Foundation, nearly a third (30%) of U.S. adults say they are not willing to take any financial risk, while 46% are willing to take average risk and 24% are willing to take above-average or fairly high risk. This suggests that many people (including investors) do not really understand how much risk they can take, what risks are, and how to deal with them [4].

2.3 Absence of Risk-based Intelligent Asset Allocation

Even if investors clearly understand their risk preferences, whether conservative, aggressive or average, they often find it difficult to independently build the optimal asset portfolio based on their risk tolerance and financial goals. Most existing investment platforms do not have the function of dynamically analyzing users' risk profiles in real time and automatically generating customized asset allocations. Research by the National Bureau of Economic Research (NBER) found that active selection patterns for different default funds show that in the absence of participation friction, most investors prefer to hold stocks, and the proportion of stocks in retirement wealth decreases with age [5]. This shows that due to the lack of a scientific decision-making process, although investors have clear risk preferences, their portfolio allocations often differ significantly from their actual preferences. Users are forced to select asset categories and determine weight allocations on their own, which may result in asset allocations that do not meet their actual needs, ultimately limiting the performance and efficiency of the portfolio.

2.4 Need for cost-effective financial analysis

As personal finance and investment become more popular, the demand for professional financial analysis skills among ordinary investors is also growing. Because they must quickly and effectively interpret and analyze massive amounts of market data in order to make wise investment decisions. However, due to the lack of necessary professional skills, most ordinary investors find it difficult to independently evaluate and analyze complex market information. Although traditional financial advisory services can provide in-depth and personalized guidance, the cost of hiring a financial advisor varies greatly, ranging from US$1,000 to US$10,000 per year, depending on the scope and complexity of the service [2]. The high service fees and relatively rigid service methods have deterred many investors.

2.5 The risk of financial fraud has increased

The rapid development of financial technology has brought about new trends in digitalization and intelligence, but it has also made the financial ecosystem more complex and fragile [6].

Numerous financial fraud phenomena are still emerging as a result of the widespread use of robo-advisors and online financial services, posing unavoidable security hazards to investors. Users' private financial information can be readily leaked or used unlawfully during transmission or storage if there are no strong data protection mechanisms in place. Users' trust in financial goods and platforms has been steadily declining as a result of the information security problems this has produced. The creation of a sound and secure financial system is more important as investors are concerned about a number of legal and regulatory issues in addition to potential financial losses from data breaches.

3.0 Objectives and Outcome

The goal of this project is to develop An Intelligent Financial Advisor System that provides personalized investment advice, dynamic asset allocation, educational resources, and AI-driven news sentiment and personalized consulting. The method is intended to increase users' portfolios' potential for growth and long-term financial stability. Given the growing demand for post-retirement financial health planning in today's aging society, this is especially crucial.

3.1 Portfolio allocation based on usre risk tolerance

Based on the user's fundamental demographic and financial data, as well as an evaluation of their risk tolerance, the system will generate a user picture. In order to address the dearth of suitable risk assessment tools, the prediction model will use this data to deliver individualized quantitative risk evaluations. The system will employ mean-variance optimization to lower the portfolio's volatility while maintaining consistency with the user's risk tolerance, taking into account the stocks chosen and the user's risk tolerance. The historical cumulative returns of the optimized portfolio will be displayed through a simulation of the portfolio's performance based on historical data. Through the data that the system generates, users may also ask the AI in the lower right corner for more financial suggestions.

3.2 Portfolio Historical Performance

This feature provides multi-portfolio backtesting, allowing users to simulate the performance of multiple portfolios with different asset allocations over a specific time period in the past, and optionally incorporate withdrawal and rebalancing strategies. Finally, a comprehensive assessment of the risk and return of different portfolios based on history is provided (final balance, CAGR, standard deviation, maximum drawdown, Sharpe ratio and Sortino ratio).

Users can simulate the historical performance of their portfolios using different withdrawal amounts, dividend reinvestment strategies and rebalancing plans, providing valuable insights into potential future results. Finally, a line chart helps users easily see which of the multiple portfolios performed best over a specific time period in the past. It helps users evaluate the risk and return of their portfolios.

3.3 Simulator Tool

Visualizing the historical performance of a portfolio shows how the same assets would have performed historically with different weights. The simulator tool will allow users to simulate various future financial scenarios by adjusting economic variables such as interest rates and inflation. By testing a portfolio under different user-defined economic conditions (optimistic, pessimistic, or baseline scenarios), users can better prepare for potential future outcomes. The tool will provide valuable insights into how a portfolio would perform under different economic environments and will be able to make more informed decisions to manage future economic uncertainty.

3.4 Other modules contributed by team members

My team members also provide market overview features, AI-based personalized advisors, single stock performance evaluation tools, visual spending planners, daily financial news with sentiment analysis, and educational resources to improve financial literacy. The system will also implement six key security measures (recommended by the SANS Institute) to protect user data. Provide users with personalized AI recommendations and tools to track income, expenses, and market performance, while ensuring data security through strong measures such as authentication, error handling, and access control.

4.0 Literature Review

4.1 AI Financial Advisor

AI Financial Advisor (Robo-Advisor) [7] is an important bridge to connect with user-end and financial products. This platform can analyze user financial status investment goals, risk tolerance level and relevant data, by algorithm, ML and AI tech. And provide asset allocations and investment suggestions in an automatic way. Compared to the early stage of financial advisor service, they were mainly relying on human experience and specialized knowledge. However, with the development of technology, current AI financial advisor can offer online financial advice, and the service of investment management. They can reduce manual intervention and improve service efficiency.

4.2 Development history of AI financial advisors

The first robo-advisers were launched during the 2008 financial crisis, the primary functions are giving portfolio advises and managing investment automatically. In 2010, Jon Stein founded “Betterment” in his 30 years old, which becoming the representative platform in the early stage of robo-advisors development. At that time, this kind of platform using online interface, simplify the traditional financial management process [8]. And the functions are simple and basic, but set a solid foundation of AI financial advisor, with the purpose of optimizing asset allocation, and financial management of clients.

In 2013, because of the technology progress and the user acceptance growth, the functions of robo-advisors gradually enriched, began to provide more personalized services [9]. The number of subscribers and assets on popular robo-advisor platforms like Wealthsimple and Betterment has increased significantly, and robo-advisors are expanding quickly at this moment.

From the technological explosion of AI and LLM beginning in 2022 to the present. The primary financial organization and wealth management company also recommend their own robo-adviser platforms. It expands its offerings beyond traditional wealth management by targeting specific markets and offering specialist services. This is a stage in the development of all-around way in robo-advisors and integration with other industries.

Starting from now, the development of AI financial advisors are expected to have a significant progress in the future. There is a survey claim that the size of the Robo Advisory Market was estimated at USD 6.77 billion in 2023 and is expected to increase at a compound annual growth rate (CAGR) of 29.7% from 2024 to 2032, from USD 8.78 billion to USD 70.31 billion [10].

4.3 Application of AI and ML in finance

Traditional financial prediction approach, always rely on statistic model such as regression analytics. This kind of way have short aspect facing higher dimension, and nonlinear data. But deep learning can recognize the data inherent in the process of raw data to the results. So this strong data processing ability can solve this problem effectively, and getting a broader using in nowadays financial field. The common machinery model, including, Decision tree, Random forest, and Neural networks.

* Decision three is a simple and efficient classification and regression method, the common way in financial field is to optimize the investment portfolio and to rank the bond [11].
* Random forest is a kind of the ensemble learning method based on decision trees. Prabhu et al. observed that random forest is significantly better than the traditional linear model in predicting asset return [12].
* Neural network, especially deep learning mode, they can capture the comprehensive, no linear relationship and higher dimension data. And well-known application is LSTM, it has outstanding performance in processing long-term and time series data [13].

By combining machine learning models and time series analysis, the accuracy of future portfolio performance predictions can be significantly improved, providing users with more reliable decision support.

4.4 Portfolio optimization

This is one of core function in AI financial advisor system, it can adjust asset weight allocation to improve investment returns and effectively reduce risk. In order to help investors achieve solid wealth growth in a complex market environment.

The Modern Portfolio Theory (MPT) proposed by Markowitz provides an important theoretical basis in asset allocation. The MPT refers to an investment theory that allows investors to assemble an asset portfolio that maximizes expected return for a given level of risk [14]. This means that MPT can offer the best asset allocation scheme in the situation of when the user gives an "Efficient Frontier".

However, MPT has gradually revealed its limitations in practical applications. First, it assumes that the future market is in a stable state and that asset correlation remains constant, which is often not the case in real dynamic financial markets [14]. In addition, one of the main challenges of MPT is that the level of risk that investors can bear is a subjective choice, which is closely related to decision-making psychology and behavioral science [15]. However, the static nature and incompleteness of MPT limit its flexibility in a rapidly changing market environment.

4.5 Challenges of existing AI financial advisory platforms

Who is the development of AI technology, the AI-driven financial consulting platform such as Betterment, Wealthfront and SigFig I already achieve an outstanding progress in the field of wealth management area. This platform using algorithm-driven way to allocate assets automatically and optimize tax affairs and financial portfolio, offering low cost, but high efficient service in wealth management.

* **Betterment:** offers financial advisor guidance based on user risk preference. Betterment brokerage accounts create and manage customized portfolios using computer algorithms. The core function including Transparent fee structure, No minimum deposit requirement, Expert-built curated portfolios [16].
* **Wealthfront:** One of Wealthfront's biggest characteristics is how well it harvests tax losses. Auto-rebalancing, risk parity, US direct indexing, and intelligent beta tools are other noteworthy features. But compared to some of the other leading robo-advisors, the online brokerage does have a $500 minimum [17].
* **SigFig:** Low management costs in comparison to traditional financial advisers are one of the finest features of investing with a robo-advisor, and SigFig consumers will be able to benefit from this feature [18].

4.6 Limitations of Traditional Financial Advisory

Traditional financial consulting services mainly rely on experienced financial advisors to provide one-on-one customized services, but this model is usually accompanied by a management fee of 0.5% to 1.5%, which will continue to increase over time, significantly reducing the long-term returns of users [19]. At the same time, by the end of 2022, the proportion of people aged 65 and above in China will reach 14.9%, and more and more elderly people are facing the challenge of how to effectively manage pensions and asset appreciation [20], and traditional face-to-face financial consulting has obvious shortcomings in financial planning for this group due to its high cost and service threshold.

5.0 Project Methodology

5.1 Calculate Risk Tolerance through data sets

In order to use the model to predict user Risk Tolerance, a suitable dataset is very important. I used the SCFP2022 dataset, which is a comprehensive household financial survey of the Federal Reserve [21]. It contains the user's financial status and demographic information, including age, marital status, income, assets, education level, etc. With this dataset, I can calculate Risk Tolerance by calculating ratio of risky assets to total assets (RiskFree represents total value of risk-free assets):

RT = Risky / (Risky + RiskFree)

Where Risky is the total value of the user's risky assets, including NMMF (non-money market funds), STOCKS (stocks) and BOND (bonds); RiskFree is the total value of risk-free assets, including LIQ (liquid assets), CDS (time deposits), SAVBND (savings bonds) and CASHLI (cash).

**Risk Tolerance:** The system evaluates the user's Risk Tolerance based on the basic information entered by the user. Risk tolerance is quantified on a scale from 1 to 5, and different financial goals are prioritized. This helps identify investor groups most similar to the user, enabling more targeted recommendations. Based on the evaluation results, the system generates multiple labels for users (e.g., "Conservative Investor," "Aggressive Growth Investor"), which are directly used for subsequent portfolio recommendations. For example, compared with older investors, younger investors may have a higher risk tolerance. Next, I will introduce how my project extracts features based on ML models, and Feature Engineering to get hidden relationships between attributes, thereby gaining a deeper understanding of the user's financial behavior.

5.2 Data Processing - Missing Value Handling

It is necessary to exclude rows from the dataset that include missing values (NaN), positive infinity (inf), or negative infinity (-inf). The performance and prediction accuracy of the model may be impacted by these kinds of incorrect data as they might lead to mistakes throughout the model training and validation process.

5.3 Data Processing - One-Hot Encoding

Each category in the categorical variable is converted into a separate binary variable (0 or 1) [22]. This method guarantees that the machine learning model can treat these categories as distinct and independent without assuming any order or sequence (which would happen if numerical values like 0, 1, or 2 were used).

After One-Hot Encoding, the new columns EDCL\_2, EDCL\_3, and EDCL\_4 represent different education level categories. For instance, if the education level of each sample is EDCL\_2, the EDCL\_2 column will be labeled as True (1) and the other columns will be False (0).

By using one-hot encoding, each category in a categorical variable is effectively converted into a separate binary feature, thus avoiding any bias that might arise from treating the categorical variable as an ordinal variable [22].

5.4 Feature Engineering

5.4.1 Feature selection

Two approaches are combined to select the features that can best predict risk tolerance (RT).

ExtraTreesRegressor model (this Model performs best, see 9.1 Model Evaluation) can calculate each feature's contribution to lowering uncertainty (like variance) while dividing the data, this tree-based ensemble learning approach assesses the significance of these features [23]. I first screened which attributes had the best chance of predicting RT using ExtraTreesRegressor. I did not, however, depend only on this automated method because doing so might leave out important aspects of real-world applications.

According to the study "AI in Context: Harnessing Domain Knowledge for Smarter Machine Learning", it highlights how incorporating domain knowledge significantly enhances feature selection and engineering, thereby improving model performance [24]. I combined the expertise in the financial field with the specific requirements of the project itself. Because machine learning models cannot identify factors that require experience and domain knowledge to understand. To ensure that these features are not only statistically valid, but also reasonable in the actual financial context.

* Age: As people approach retirement age, their risk preferences usually change [25].
* Income and assets: Income level and asset status directly affect a person's financial stability, which in turn affects their risk preferences [26].
* Education background and occupation: These two factors reflect a person's level of financial knowledge and income stability, which in turn affects financial decisions [27].
* Marital status and number of children: Increased family responsibilities usually also affect people's attitude towards risk [25].

Doing so not only improves the accuracy of the model, but also enhances the interpretability of the model [24].

5.4.2 K-fold Cross Validation

K-fold Cross Validation is a cross-validation method designed to evaluate the generalization ability of a model. The basic principle is to divide the dataset into K equal subsets, use K-1 subsets for training each time, and the remaining 1 subset for validation [28]. This process is repeated K times, and a different subset is selected as the validation set each time. In the end, the evaluation result of the model is the average of the K validation results, which can reduce the deviation of the model on a specific data partition and improve the stability and reliability of the model.

5.4.3 Model evaluation index

In order to fully evaluate the capability of the model, I also used followed regression and classification evaluation indicators.

Mean Square Error (MSE) estimates the average of the sum of squares of the differences between the predicted values () ​​and the actual values (). The MSE smaller, the model prediction performance will be better. And denotes that sample number [29][30].

Root Mean Square Error (RMSE) is the square root of the MSE. It is in the same units as the actual data, making it easier to interpret. The smaller the RMSE value, the smaller the prediction error of the model [31].

Mean Absolute Error (MAE) calculates the average of the absolute differences between the predicted and actual values. MAE is less sensitive to outliers than MSE [30].

In classification problems, the F1 Score is the harmonic mean of precision and recall, where precision is the percentage of actually positive samples that are correctly predicted as positive [32] and recall is the percentage of actually positive samples that are correctly predicted as positive [30]. The higher the F1 Score value, the better the model balances precision and recall.

These indicators can help measure the performance of the model on the training set and validation set.

5.4.4 Parameter search

I used RandomizedSearchCV to randomly search the hyperparameters of the ExtraTreesRegressor model, and optimized key parameters such as n\_estimators, max\_depth, min\_samples\_split, min\_samples\_leaf, and bootstrap [33]. When evaluating the model, we not only focused on the R² score, but also combined regression indicators such as MSE, RMSE, and MAE to ensure that the model can maintain good predictive performance and stability on new data.

5.5 From basic linear models to advanced integrated models

In the intelligent financial advisor system, selecting a suitable ML model is a key step to improve the accuracy of user risk tolerance prediction and optimize investment portfolio recommendations. This project uses a variety of methods from basic linear models to advanced integrated models to comprehensively see the performance of different algorithms on prediction tasks, which is used to subsequently select the most suitable model for the function of predicting Risk Tolerance.

* **Linear Regression:** A popular and practical statistical learning technique, linear regression is a very simple and common way to supervised learning. One effective method for forecasting a quantitative response is linear regression [34].
* **Random forest:** The values of a random vector sampled independently and with the same distribution for every tree in the forest determine the values of each tree in a random forest, which is a mixture of tree predictors [35].
* **Extremely randomized Trees:** A regressor with more trees (ExtraTreesRegressor). An extremely randomized tree growing algorithm that combines the attribute randomization of Random Subspace with a totally random selection of the cut-point [23].
* MLP: it is the most common neural network [36] MLP is used to explore potential nonlinear relationships in user financial behavior. Its multi-layer structure can capture deep interactions between input features. It is an important model for this project.
* XGBoost: An efficient regression model based on gradient boosting framework. It is a novel sparsity-aware algorithm for sparse data and weighted quantile sketch for approximate tree learning [37]. It is an important candidate model for this project.

5.5 Portfolio Recommendation

Dynamic asset allocation is the core function of the intelligent financial advisory system, which aims to generate personalized asset allocation recommendations for users based on their risk tolerance and market data. Let's explain step by step my methodology for personalizing Portfolio recommendations and using Convex Optimization.

5.5.1 Data preparation and revenue calculation

The system calculates the expected returns for each asset in the portfolio. Historical price data is loaded, cleaned, and resampled to a monthly frequency. Given a time series of historical asset prices , where denotes the asset and denotes the time index (monthly), the monthly return ​is computed using the formula [38] [39]:

This transformation converts absolute prices into relative returns, enabling comparison across assets and time. The result is a matrix . The monthly expected return for each asset is calculated as the arithmetic mean of its historical monthly returns and then annualized by multiplying by 12.

5.5.2 Covariance Matrix

As we already get the return matrix  before, the empirical covariance matrix can now be calculated. Among this, is the mean return vector (each asset’s average return) and ⊤ denotes the transpose [39] [40].

The covariance matrix is then annualized by multiplying by 12.

5.5.3 Portfolio Volatility Calculation

Portfolio volatility is computed as the square root of the weighted sum of variances and covariances [39] [40]:

where is the weight vector, and is the covariance matrix. The optimization process aims to minimize portfolio volatility while ensuring that the sum of asset weights equals 1 [40].

5.5.4 Portfolio Optimization Objective Function

Now, we have found the weight vector (optimized weights from the convex problem) , but this vector is the weight combination that minimizes the overall risk of the portfolio without considering personal risk preferences [39]. To this end, we introduced a Risk Tolerance parameter, which we use mathematical symbols to represent as , and its value range is . This is predicted by my machine learning model based on the user's basic information.

So, the final weight vector is adjusted as:

where represents an equally weighted portfolio. The final weights are normalized so that their sum equals 1.

5.6 Efficient Frontier and Capital Market Line

It is built by resolving optimization issues for different target returns. It denotes the collection of investments that yields the highest expected return at a specific risk level or the lowest risk at a specific expected return, creating the portfolio with the least amount of volatility for each return [39]. The Capital Market Line (CML) connects the risk-free asset to the tangent portfolio with the highest Sharpe ratio, defining the best risk-return tradeoff.

The efficient frontier is calculated by:

In financial modeling, the 10-year Treasury yield is often used as an approximation of the Risk-Free Rate (). The US 10-year Treasury bond interest rate is around 4% [42], in the project we set ≈0.04. The Sharpe ratio is maximized to identify the tangent portfolio:

Where: is the portfolio return, and is the portfolio volatility.

The Capital Market Line connects:

Point : representing 100% investment in the risk-free asset.

Point : the tangent portfolio with maximum Sharpe ratio.

Where: is the expected return of the tangent portfolio, is the volatility of the tangent portfolio [30].

5.7 Portfolio Simulation and Scenario Adjustment

This section explains the methodologies used in the portfolio simulation system, which allows for dynamic adjustments of portfolio parameters, different simulation models, and scenario-based analyses.

5.7.1. Cash Flow Application (Adjusting Portfolio Balance)

In real-world financial planning, portfolio balances are often adjusted over time by adding or withdrawing cash. These changes can either be fixed amounts or percentages based on the user’s preferences.

* **Fixed Withdrawal:** User can input fixed withdrawal amount, it can be adjusted for inflation. If inflation is applied, this amount increases each year by the inflation rate:

The balance is then reduced by the total withdrawal over the specified frequency (monthly, quarterly, annually).

* **Fixed Contribution:** Similarly, a fixed contribution can be added each year, again adjusted for inflation if needed.
* **Percentage Withdrawal:** Similar to Fixed Withdrawal, but the specific amount of money withdrawn becomes a percentage of the portfolio. The user can input the percentage of withdraw. And it is processed in coordination with the withdrawal cycle.

5.7.2 Scenario Adjustment (Optimistic and Pessimistic Scenarios)

Different market conditions (such as optimistic and pessimistic outlooks) require adjusting the parameters used in simulations.

Optimistic Scenario: Increases mean returns and decreases volatility.

Pessimistic Scenario: Decreases mean returns and increases volatility.

5.7.3. Simulation Models

We mainly use Monte Carlo Simulation to simulate the portfolio's performance over time [43]. And there are various types of simulations to model the performance. Three main types of models are used: Statistical, Historical, and Parameterized in Monte Carlo Simulation.

* **Historical Simulation:** Historical simulation is a forecasting method that uses actual past returns of assets to simulate future portfolio performance. The process involves loading historical data, calculating asset returns, and applying these returns to simulate portfolio growth over multiple periods. Cash flows (withdrawals or contributions) and rebalancing adjustments are applied throughout the simulation. First monthly returns are calculated as [38]:

Portfolio returns are simulated by applying random historical returns to the portfolio's assets. The weighted total of the returns on each individual asset makes up the portfolio return:

* **Statistical** **Simulation (Normal Model)**

The Normal Model assumes that asset returns follow a normal distribution [44]. In the annual normal simulation, the returns of assets are generated based on the mean and volatility parameters provided, and portfolio value is recalculated yearly.

Multivariate Normal Distribution: For each year, we generate a vector of asset returns from a multivariate normal distribution:

The balance is updated each year by applying the cash flow adjustments.

* **Statistical Simulation (GARCH Model)**

The GARCH model is used to simulate volatility clustering and time-varying volatility, which is a more advanced model for modeling asset returns [45] [46]. The GARCH(1,1) model allows for modeling of volatility over time, considering past volatility and returns.

The GARCH(1,1) model for asset returns is given by:

is the time-varying volatility modeled as:

,, and are parameters estimated from data, is the residual (shock) at time

* **The Parameterized Simulation**: involves defining parameters for the return distribution (such as mean 𝜇 and volatility 𝜎) and using these to generate future returns based on either a lognormal or normal distribution [47]. This model can simulate periodic returns as well.

Lognormal Model:

Normal Model:

Below are the parameters involved and how each affects the results.

* Annualized mean (μ) as the central value of the expected return of an asset, which is the center value of the predicted return of an asset. A higher mean will result in a greater level of simulated returns.
* Annualized volatility (σ) indicates the degree of dispersion of the asset return data, which establishes the fluctuation range of the randomly produced returns. The greater the volatility, the more evident the generated data's uncertainty and the higher the risk level.
* The correlation coefficient and covariance matrix between assets describe the joint movement relationship between assets, and change the portfolio risk structure by affecting the multivariate normal distribution generation process; close correlation may lead to risk concentration.
* The normal distribution model is appropriate for either yearly or periodic simulations and produces random returns based on annualized parameters [44]; the accuracy and continuity of return generation within the cycle are directly impacted by parameter conversion.
* The GARCH model setup incorporates volatility aggregation, or the time-dependent fluctuations of asset returns [46]; the dynamically adjusted volatility is especially sensitive to risk shifts and simulations of severe events.

6.0 Design and Development

6.1 System Architecture and Design

The system adopts a front-end and back-end separation architecture. The server integrates Node.js, Express, and Flask to provide back-end framework support, and uses Postman for API testing and debugging. The database layer currently uses Amazon Web Services RDS to store user data, and then financial stock and market data use local csv files. The front-end is developed using the React framework, combined with the UI library (Ant), to provide an intuitive and responsive user interface. And all user data and sensitive information are processed through encryption technology to ensure the security of the system.

This complete project was completed by me and my team member LEE, Ka Chun Caius (20069386D). In Figure 1, my personal contribution is detailed with attributes and methods, while the function written by Caius is only the class name.

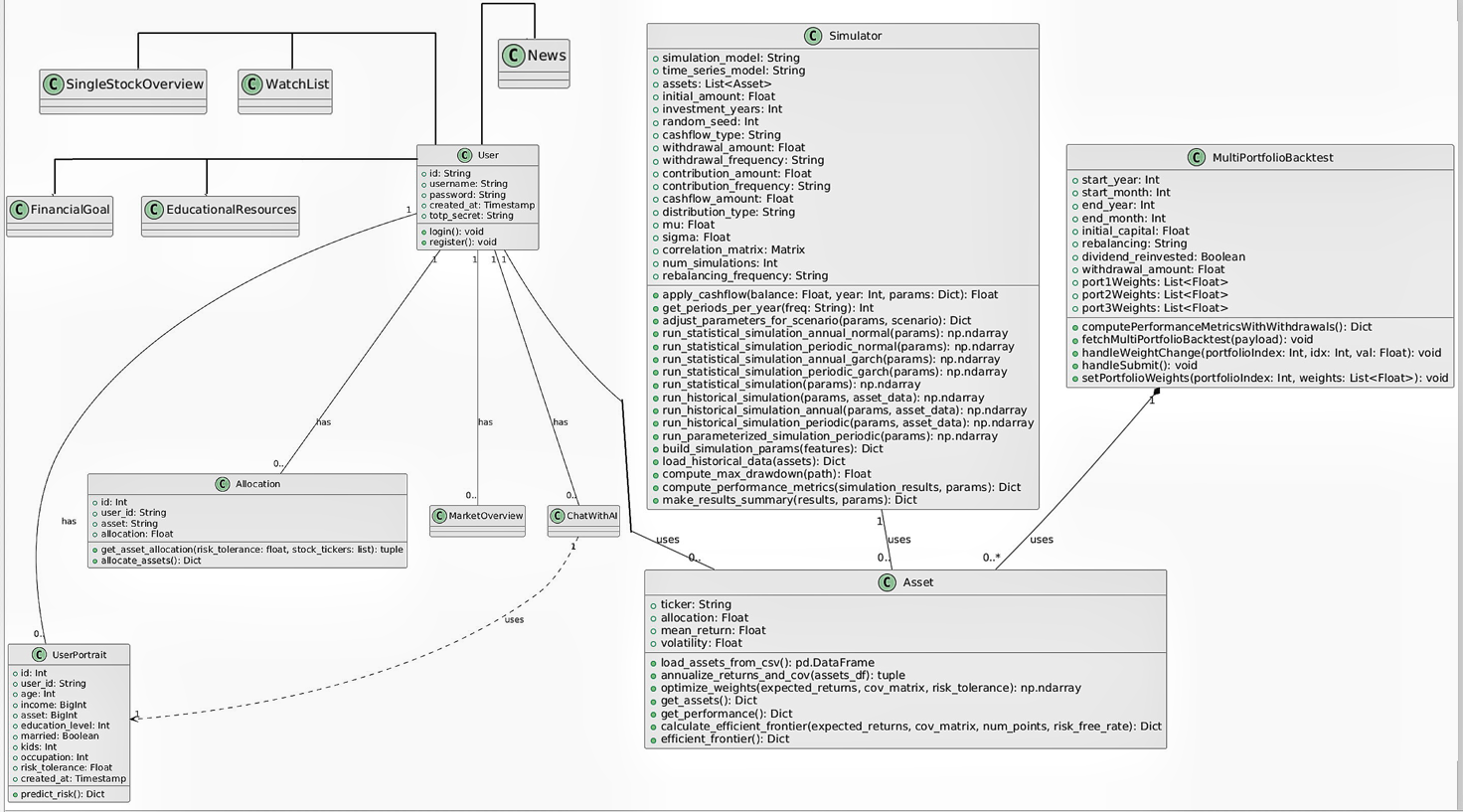


Figure 6.1: Class Diagram of the System Architecture

In this project, my main contribution is concentrated on three core components, namely UserPortfolio, Simulator and MultiPortfolioBacktest. These components are closely connected, processing data through the powerful technology of Machine Learning model and providing customized financial planning solutions for users.

* The UserPortfolio component is used to manage the user's basic information, investment preferences, risk tolerance, etc. The data entered by the user will drive the system's calculations for other modules, and combined with the user's investment goals, provide data for subsequent ChatWithAI. This module not only helps users manage their investment portfolios, but also generates customized investment recommendations based on the user's risk preferences. In this module, the user's ID is generated through an encryption mechanism and stored in combination with SessionKeyId. After the user logs in, the user ID in the token is extracted and used to access the user's personal investment data through a decryption mechanism.
* The Simulator component is the core simulation module in the system. It simulates the user's investment portfolio by inputting market data, financial goals, and other relevant parameters. This module simulates the performance of investment portfolios under different investment scenarios through historical data, market trends, and user preferences, and provides users with practical investment plans. The simulation results will be presented to users in the form of charts or reports to help them make decisions.
* The MultiPortfolioBacktest component focuses on backtesting analysis of multiple investment portfolios. Users can enter the configuration of multiple investment portfolios and view their performance in historical data. This module uses statistical methods to analyze backtest results, generate performance indicators, and provide users with historical trend references.

In addition to these core modules, I also participated in data transmission and integration. By extracting and inserting user data from the database and passing information between modules, the system can generate accurate investment advice and investment questions and answers. For example, ChatWithAI will fetch user information generated by the UserPortfolio module, so the user information in UserPortfolio will directly affect the answer results of ChatWithAI.

In addition, the project also connects the parts of LEE Ka Chun Caius. LEE is responsible for designing the Market Overview, SpendingPlanner, News, EducationalResources, and ChatWithAI. These modules provide users with support for market trends, consumption planning, news information, and educational resources, helping users make better investment decisions in the Portfolio module. At the same time, they interact with users through ChatWithAI and guide them in using FinancialGoal and Simulator for long-term financial planning.

The entire system interacts with users through ChatAdvisor, providing users with personalized question-and-answer suggestions, while helping them better utilize portfolio backtesting and simulation functions to form a complete financial planning tool.

6.2 Database Design and Normalization

6.2.1 Database Design

The main objective of database architecture in this project is to store and manage user data, including personal information, investment portfolios, financial goals, consumption records, and other information, in order to guarantee the system's effectiveness, consistency, and dependability. The whole data architecture was designed using a relational database management system (RDBMS), and the database administration platform was Amazon RDS for MySQL, a cloud database. In order to minimize redundancy, enhance data consistency, and guarantee the scalability of the data model, the database design process adhered to the data normalization concept. There are several tables in our database. The specific information is shown in the ER Diagram below:

图示

描述已自动生成

Figure 6.2: ER Diagram of the database

The users table stores the user's basic account information; the user\_portrait table stores detailed data related to the user's finances, including assets, risk tolerance, age, etc.; the allocations table stores user asset configuration data and allocation ratios; the goals table records the user's financial goals; the expenses\_income table stores the user's income and expenditure records; the user\_watchlist table stores the user's stock watchlist; These tables are associated through foreign keys to form a complete data relationship model, which provides data support for various modules of the system (such as user module, investment portfolio module, etc.).

6.2.2 Database Normalization Process

Database normalization is a design technique that aims to optimize data structure and eliminate data redundancy and inconsistency. The database design of this project follows the principle of normalization.

* According to the first normal form (1NF), a data table's columns must all have atomic values; that is, no field may include set-type data or duplicate values [48]. So, the data we kept in the fields of every table in this project's database architecture is atomic.
* The second normal form (2NF), every non-primary key field must rely entirely on the table's main key rather than only a portion of it [48]. Every non-primary key field in our tables depends entirely on the primary key. Fields like age, salary, and asset, for instance, in the user\_portrait table are entirely dependent on user\_id rather than a portion of the primary key.
* The third normal form (3NF), all non-primary key fields must directly rely on the main key and not be able to transmit the reliance through other non-primary key fields [48]. In order for the database design to adhere to the third normal form (3NF), we made sure that every non-primary key field was directly reliant on the main key during the design process. For instance, in the user\_portrait database, user\_id is the direct determinant of columns like age, income, and asset, but not of other variables (such employment, risk tolerance, etc.).

6.2.3 Data table design and foreign key constraints

We create associations between tables using foreign key restrictions to guarantee data integrity and consistency. For instance, the user\_id column of the users table is referenced by the foreign keys user\_id of the user\_portrait, allocations, objectives, and expenses\_income tables. Data redundancy and isolated data production are avoided in this architecture.

At the same time, we further improve query efficiency through reasonable index design. The users table's id field is automatically indexed as the primary key, and other tables' user\_id fields are similarly indexed as foreign keys. This makes it easier to find user-related information, prevents complete table scans, and enhances system efficiency.

7.0 Implementation

In the local development environment, I first initialized the Node.js project using npm and defined RESTful API routes based on the Express framework. The entire system backend adopts a modular structure, integrating functional modules such as user authentication, data processing, Flask microservice calls, security middleware, and database connections.

In the Flask application, we structured the application by organizing Python functions and performance analysis blueprints into separate modules. The machine learning model is loaded from the pickle file and used in the /predict\_risk endpoint to provide risk prediction services. A dedicated function is used to load historical asset data from a CSV file (stock\_data\_with\_sector.csv). The front end is developed using React, using a componentized design to communicate with the backend APIs.

I used my team members' security module as a basis for my logic implemented. The back end creates and safely maintains temporary key pairs and SessionKeyId, while the system front end encrypts sensitive data using elliptic curve cryptography and the AES method.

The decryptRequestMiddleware on the backend decrypts the original data based on the SessionKeyId for use in my asset allocation, risk prediction, efficient frontier computing and other businesses; after the business is processed, the encryptResponseMiddleware uses the same SessionKeyId to encrypt the response data and safely transmit it back to the frontend. (Contributed by my team members)

For the database, we use AWS RDS to store user and system data. The front-end is developed using React, which has a componentized design and is easier to expand and maintain the subsequent system functions.

7.1 UserPortrait Module

To provide users individualized financial planning and investment portfolio advice, this module creates a safe and effective user portrait system. The system gathers user preferences and basic data, integrates the back-end machine learning model, forecasts user risk tolerance, and creates customized asset allocation strategies. Additionally, my team member makes use of JWT-based authentication, SessionKeyId session management, and data encryption. To preserve user privacy, all sensitive user data will be encrypted both during transmission and storage.

The module uses a single API interface to communicate with the front-end. To store, query, and update user pictures and asset allocation data, the back-end logic processes the data and communicates with the MySQL database.

7.1.1 Backend Implementation

The back-end system uses Flask to develop the RESTful API interface, and combines Node.js/Express to handle user authentication and encryption processing to receive registration and login requests supplied by the front-end.

The backend API interfaces related to UserPortrait include:

* **Risk prediction interface (/predict\_risk):** Use the features entered by the user (such as age, income, assets, education, etc.) to call the trained machine learning model to predict the user's risk tolerance.

When the user submits feature data such as age, income, assets, education, etc. through a POST request, the backend first uses the Flask framework to receive the request data in JSON format. Subsequently, the system calls NumPy's methods np.array and .reshape(1, -1) to convert the incoming data into a two-dimensional array to meet the input requirements of the pre-trained machine learning model. Next, the system calls the model loaded through pickle to pass the data into the prediction function, and the value returned by the model represents the user's risk tolerance. After the prediction is completed, the system uses Flask's jsonify method to package the result into JSON format and return it to the request end. In the subsequent data update process, this prediction result will be used to update the user portrait data and provide a reference for subsequent asset allocation. In addition, the try-except block is used throughout the process to capture possible exceptions to ensure that the exception information can be fed back in JSON format.

* **Asset allocation interface (/allocate):** extracts the user's risk tolerance and selected stock data from the user profile table, and calculates the best asset allocation plan.

After the user sends a POST request containing risk tolerance and a list of stock codes, the backend first uses an input validation mechanism to check whether the risk\_tolerance data and a non-empty stock\_tickers array are passed in, as well as the existence of the stock code in the historical data (assets\_monthly in Pandas DataFrame) to ensure data integrity. The system then calls the data extraction method in Pandas to filter out the price information of the user's selected stocks from the historical asset data, and then uses the .pct\_change() method to calculate the monthly yield of each stock. In the next step, the annualized expected return (mean) and covariance matrix are calculated by calling the custom function annualize\_returns\_and\_cov on the monthly yield data.

Based on these data, the system uses the minimize function in the SciPy library and specifies the SLSQP algorithm as the solver to calculate the stock allocation combination that achieves the minimum volatility under the constraints (the sum of all weights is equal to 1, and each weight is between 0 and 1). The optimal weights calculated are adjusted by the user's risk preference (risk\_tolerance) and the normalization method is used again to ensure that the sum of the weights is 1. Finally, the data returned to the client includes the allocation ratio of each stock and the cumulative return time series of the investment portfolio calculated by weight and monthly return.

* **Performance backtesting interface (/get\_performance):** The performance backtesting interface is used to evaluate the performance of the user's historical investment portfolio. By matching and calculating the previous asset allocation data and historical market data, a time series that intuitively displays the cumulative returns of the investment portfolio is generated.

This interface will be automatically executed after receiving a POST request containing the stock code and the corresponding configuration weight. Then the backend will verify the data to ensure that the number of stock codes and weight data is consistent, and then extract the corresponding stock historical price data through Pandas, and use the .pct\_change() method to calculate the monthly return for each period. The dot multiplication operation (np.dot) in NumPy is used to combine the returns of each stock with the corresponding weights to obtain the returns of the entire investment portfolio in each period. Next, the historical portfolio value time series is calculated by compounding the period returns and multiplying them by the initial capital or 100 to convert them into percentages. Finally, these data frames organized by Pandas with dates as indexes and corresponding market values ​​as values ​​will be returned to the client in JSON format through the jsonify method.

* **Efficient frontier data interface (/efficient\_frontier):** This interface is designed to show users the optimal return possibilities under different risk levels, that is, the efficient frontier data of asset allocation, so as to help users make more informed investment decisions.

After the user submits the selected stock code, risk-free rate (default value is 0.04) and the desired number of data points through a POST request. The backend will first extract the historical price data of the corresponding stock, and then call the custom function annualize\_returns\_and\_cov to calculate the annualized expected return and covariance matrix of each stock. Then, SciPy's minimize function is used to adopt the SLSQP algorithm to first solve the minimum volatility combination that satisfies the sum of all weights to 1 and each weight is in the range of 0 to 1. Then the system will construct a "preset target return interval" to represent the possible annualized return target range of the combination. This interval is composed of the minimum and maximum annualized returns of all stocks, and then the NumPy np.linspace method is used to generate a set of equally spaced target return values ​​between these two endpoints (num\_points defaults to 50). The system will then use the target return value as a constraint to optimize the optimal weight combination that meets the target return, thereby constructing a series of risk-return points and forming an effective frontier curve. In addition, the system also defines the Sharpe ratio function and uses the SLSQP optimization algorithm to solve the tangent investment portfolio, which corresponds to the configuration with the highest return-risk ratio; based on the tangent portfolio and the risk-free rate, the data points of the capital market line (CML) are further calculated to show the incremental return that the investment portfolio can achieve above the risk-free rate. The final calculation results are returned to the client in JSON format.

7.1.2 Interaction between the backend and the database

The database mainly involves two tables: user\_portrait and allocations.

The user's basic financial information and risk tolerance data are stored in the user\_portrait table in the database. It stores user portrait information, including basic user information, investment preferences, risk tolerance, etc. The user\_id of this table is a random value generated by encryption and is associated with the users table as a foreign key. Every time a user registers or logs in, the system manages the user portrait data through the following process:

* When the user submits basic information through the API, the backend first queries the database based on the user's unique identifier (such as user\_id) to see if the corresponding user record already exists. If it does, it means that the user has registered before. At this time, the system will use the database's UPDATE operation to update the latest financial data (such as age, income, assets, education level, marital status, number of children, and occupation, etc.) to the existing record; if it does not exist, a new user portrait record will be created through the INSERT operation.
* After completing the update or insertion of basic information, the system will call the backend machine learning model to predict the user's risk tolerance. After obtaining the risk tolerance value, the UPDATE operation is used to update the prediction result to the corresponding user portrait record. This ensures the accurate storage of basic data and ensures that the risk assessment results are always consistent with the latest information submitted by the user.

User asset allocation data is stored in the allocations table in the database, which records the allocation ratio of each user in each stock or asset category.

* When a user initiates an asset allocation request, the backend calculates the optimal asset allocation ratio through a specific algorithm based on the user's current risk tolerance and the selected stock code. To prevent data redundancy, the system first queries the existing configuration data in the database based on the user ID, and uses the DELETE operation to delete all related records previously stored.
* After ensuring that the old data has been cleared, the backend inserts the new asset allocation data into the allocations table one by one. Each new record contains the user ID, the corresponding stock code, and the calculated allocation ratio, so as to ensure that the latest asset allocation information is always stored in the database.

7.1.3 Front-end Implementation

The React framework and Material-UI (or Ant Design) are used to construct the front end, to build a modern user interface. The entire front-end adopts a component-based design, effectively splitting the user input, data display, and chart visualization modules.

* **InvestorForm:** This component is used to collect basic information entered by users, including age, income, assets, education level, marital status, number of children, occupation, and risk preferences of users. The data entered by users will be passed to the backend in JSON format for risk tolerance assessment. The component uses controls such as Grid, Slider, and FormControl in Material-UI to implement dynamic responsive layout and provide real-time data feedback.
* **AllocationChart:** This component displays the asset allocation data returned by the backend, and uses PieChart in Recharts to draw a pie chart to intuitively display the allocation ratio of each asset category. After receiving the data, the component processes the data and converts the ratio of each asset allocation into a percentage, so that users can understand the optimized configuration of the current investment portfolio at a glance.
* **PerformanceChart:** Displays to users the investment portfolio's past performance and uses Recharts' LineChart to illustrate the cumulative return curve. The net value curve of the investment portfolio over time is shown in an understandable manner utilizing charts that structure date labels and numerical scales to assist readers in comprehending the investment portfolio's historical performance.
* **EfficientFrontierChart:** To help users understand the expected returns under different risk levels, this component displays the effective frontier data and the CML of asset allocation. Through Recharts' ComposedChart, the efficient frontier curve, ineffective area, and scatter information of the tangent portfolio are displayed simultaneously. It helps users intuitively understand the risk and return relationship faced by asset allocation, which helps to make more scientific investment decisions.

In addition, after the user registers or logs in, the backend returns a signed JWT Token. The frontend stores the token in localStorage or global state, and includes it in the request header in subsequent API calls to ensure that all data interactions are authenticated. And use HTTP clients such as Axios to uniformly call the backend interfaces (/predict\_risk, /allocate, /get\_performance, and /efficient\_frontier).

7.2 Risk Tolerance predicting Implementation

7.2.1 Data processing

In the data preprocessing part, first for rows containing missing values ​​(NaN), positive infinity (inf), and negative infinity (-inf) were removed to eliminate errors in model training and validation.

In order to realize the function of predicting user risk tolerance, I decided to use machine learning to learn data and predict results.

During the implementation process, I first loaded the SCFP2022 dataset from the Excel file, which contains the user's financial information and asset distribution data. Calculate the user's risk tolerance (RT).

Risk Tolerance = Risky assets / (Risky assets + Risk Free assets)

Then for the categorical variables 'EDCL(education level)', 'MARRIED', 'OCCAT1(occupation)', 'YESFINRISK(willing to take financial risks)', there are different categories that have no inherent order or numerical relationship. A person's education level, marital status, or occupation category should not be considered as a numerical variable with magnitude or ranking meaning. Using numerical values ​​such as 0, 1, or 2 to represent these categories may introduce bias to the model because it means that there is a natural order or distance between these categories that does not exist.

In order to resolve this issue, I used one-hot encoding. One-hot encoding is a method that converts each categorical variable into a series of binary variables (0 or 1) [22]. Every category is represented by a distinct binary feature, where all other values are "0" (showing the absence of the category) and only one value is "1" (representing the presence of the category). It guarantees that every category is handled as a separate feature without any implied order or relationship between them. It enables the model to accurately analyze categorical data without assuming the wrong things about how categories relate to one another.

The final dataset consists of continuous variables (e.g., age, income, assets) and one-hot encoded categorical variables. The target variable, risk tolerance (RT), is separated from the feature set, ensuring that the model can focus on predicting this outcome.

7.2.2 feature selection

The feature selection process is a critical part of building a predictive model because it ensures that only the most relevant variables are included in the final model.

In the final model, I selected the following features as input: AGE (age), EDCL (education level), MARRIED (marital status), KIDS (number of children), OCCAT1 (occupation category), INCOME (income), YESFINRISK (willingness to take financial risks), ASSET (assets), and RT (risk tolerance).

The selection of these features is not based solely on the feature importance scores automatically calculated by the model, but is supported by domain expertise and existing literature. Although the results of automatic evaluation using ExtraTreesRegressor (I use this model because it performs best, the overall model performance will be shown in “9.1 Model Evaluation”) on the raw data in the For feature importance demonstration BEFORE one-hot encoding stage show that some variables rank high, such as MORTPAY and PENACCTWD, but I did not finally choose them because following reason.

Categorical variables like EDCL, MARRIED, OCCAT1, and YESFINRISK remain as a single value or category in the absence of One-Hot encoding. When this is done, the contribution of categorical variables is "compressed" or understated when determining feature importance since the distinctions between them cannot be completely expressed.

However, continuous variables such as MORTPAY and PENACCTWD may be considered by the model to have strong "explanatory power" in explaining the target variable (RT) due to their large numerical changes or significant distribution differences in the original data. Below image shows that MORTPAY and PENACCTWD on the top 2 importance features to RT, and this is the process before one-hot encoding.

图片包含 图表

描述已自动生成

Figure 7.2: Top Features before One-hot Encoding

However, the feature relevance of these variables will vary dramatically following data preprocessing, particularly One-Hot encoding. The total effect of the category factors on the target variable RT is more accurate and comprehensive since they are divided into dummy variables, each of which only contains a portion of the information.

Conversely, without preprocessing, continuous variables like MORTPAY and PENACCTWD appear significant, but their data could not be crucial in complicated risk tolerance prediction since they might only represent one economic activity (loan repayment, for example). We can see the below top 10 feature importance image, MORTPAY and PENACCTWD is no longer in the top 3, after we did one-hot encoding.

图表, 条形图

描述已自动生成

Figure 7.2: Top Features After One-hot Encoding

Then, I separated the target variable RT from the other features in the dataset. The target variable (Y) represents risk tolerance, while the other columns serve as independent predictor variables (X). This means that the model is trained only on the input features without access to the target value. The dataset is then divided in an 8:2 ratio into train and validation sets. (Not randomly and with a fixed seed for reproducibility)

7.2.3 model training

During this process, I tested multiple regression algorithms (linear regression, decision trees, and ExtraTreesRegressor and so on) to select the best model.

To further improve the model performance, I tuned the hyperparameters of ExtraTreesRegressor using RandomizedSearchCV. The tuning process is guided by the R² metric and uses a cross-validation strategy (with 10 folds) to ensure robust evaluation.

The final ExtraTreesRegressor model is trained on the training data using the best hyperparameters that were found. The model's performance is assessed using metrics like , MSE, RMSE, and MAE on both training and validation sets.

7.2.2 User Risk Tolerance Prediction

When the front-end user submits basic financial and demographic data, the data is sent to the /predict\_risk endpoint in JSON format. The back-end converts the received feature list into a NumPy array and resizes it to the required input dimensions of the model.

The pre-trained model (loaded from the pickle file by deserialization) predicts the processed feature data and outputs a risk tolerance value. This value is mapped to a fixed range (1 to 5) for easy display on the front-end and subsequent asset allocation.

The predicted risk tolerance is returned to the front-end in JSON format, and the result is also saved for subsequent asset allocation decisions.

7.2.3 Dynamic Asset Allocation

Dynamic asset allocation is implemented through the /allocate endpoint. This part of the system uses the risk tolerance score, it is obtained from the prediction module, and the list of stock symbols selected by the user to calculate the optimal asset allocation.

The system first verifies the validity of the received risk tolerance parameters and the list of stock symbols, and checks whether the requested stock exists in the preloaded asset data (assets\_monthly).

Using the preloaded CSV data, the system extracts the historical monthly closing prices of the selected stocks and calculates the monthly percentage change of the assets. In addition, the auxiliary function annualize\_returns\_and\_cov is used to annualize the returns and covariance matrices of each stock.

For a multi-asset portfolio, the system calls the optimization function optimize\_weights. This function first initializes uniform weights for all assets, and then defines an optimization function that targets portfolio volatility based on the covariance matrix. Under the constraints that the total weight is 1 and each weight is between 0 and 1, the SLSQP algorithm is used to solve the optimal weights. Then, according to the user's risk tolerance, the optimal weights are adjusted to ensure that the final asset allocation reflects both the optimal distribution revealed by the market historical data and the user's personalized risk preference.

Finally, the system returns the calculated configuration weights and the cumulative returns of the portfolio based on compound returns in JSON format for front-end display and further performance evaluation.

7.2.4 Portfolio Historical Performance

This part is implemented through the interface /get\_performance. First, based on the user's existing asset configuration, the system extracts the historical price data of the corresponding assets from the preloaded assets\_monthly data and calculates the monthly return. Using compute\_performance\_metrics, the calculation process comprehensively evaluates the performance of the portfolio over a specified period of time based on the portfolio's historical returns and initial capital. The final performance data, including detailed time series, is formatted as JSON and returned to the front end, providing users with an intuitive cumulative return line chart and a detailed portfolio performance report.

7.3 Simulation Module

The simulator simulates the potential performance of investment portfolios in various market scenarios through a variety of different financial models and user-defined parameters. This module can not only generate future return forecasts based on historical data and statistical models, but also support users to directly define expected returns and volatility in a parameterized manner. In addition, the simulation module has built-in scenario adjustment and cash flow application functions, making the simulation results closer to reality.

7.3.1. Simulation model

The system supports a variety of simulation methods to adapt to different investment scenarios and user needs. It mainly includes the following three types of models:

1. **Historical simulation:** Use asset return data from past historical stock price data to simulate.

* The system extracts historical data for each asset from the preloaded CSV data and calculates the monthly return for each stock.
* Using the run\_historical\_simulation\_annual and run\_historical\_simulation\_periodic functions, the system randomly extracts a historical return sequence for each asset, calculates the compound return of each asset within a year or a period, and then generates the overall return of the portfolio according to the asset allocation weight.

1. **Statistical simulation:** Statistical simulation generates portfolio returns based on predefined statistical models. This module supports normal distribution models and GARCH statistical models. Statistical simulation first obtains the annualized mean, volatility, and correlation coefficient of each asset from historical data or user input, and calculates the covariance matrix based on these parameters.

* For the normal distribution model, the system uses np.random.multivariate\_normal(means, cov\_matrix) in the annual simulation function run\_statistical\_simulation\_annual\_normal to generate a set of random asset returns, and then weighted sums them according to the allocation ratio of each asset to obtain the annual return of the overall portfolio; if periodic simulation (such as monthly or quarterly) is used, the number of periods in a year is determined by get\_periods\_per\_year, and the annualized mean and annualized volatility are converted to periods, and then norm(mu, vol).rvs() is called to randomly generate return data for each period, and asset rebalancing and cash flow adjustments are performed after each period to cumulatively calculate the overall return of the portfolio.
* For the GARCH model, run\_statistical\_simulation\_annual\_garch or run\_statistical\_simulation\_periodic\_garch is called, and arch\_model is used to construct the GARCH model to simulate the asset return sequence with time-dependent volatility.

Regardless of the statistical model used, the system will repeatedly randomly generate simulation paths, record the value changes of the investment portfolio in each cycle, and finally output multiple simulation curves for subsequent performance evaluation.

1. **Parametric simulation:** completely relies on key financial parameters provided by the user, such as expected annualized return (μ) and annualized volatility (σ). It does not rely on historical data or statistical assumptions. This method is flexible and suitable for situations where users have specific expectations for the market.

* After the user enters the parameters, the system will directly call run\_parameterized\_simulation\_annual or run\_parameterized\_simulation\_periodic, skipping the sampling process of historical data.
* According to the μ and σ set by the user, a random annual return value is generated using lognorm or norm (for example, a random value is generated by lognorm(sigma, scale=np.exp(mu)).rvs(), and then minus 1 to get the actual return rate), then the initial balance of the portfolio is updated and cash flow operations are applied;
* In periodic parameterized simulation, the annualized parameters are converted to periodic parameters (such as monthly mean and volatility), random return data is generated using the corresponding distribution for each period, and assets are rebalanced and cash flows are adjusted at the end of each period.

7.3.2. Scenario Adjustment

The scenario adjustment component is used to dynamically adjust the simulation input parameters according to the market scenario selected by the user (such as baseline, optimistic, and pessimistic). Use the adjust\_parameters\_for\_scenario function to modify the key parameters of the model according to the scenario selected by the user:

* In the optimistic scenario, the average return (μ) may be adjusted upward and the volatility (σ) may be appropriately reduced;
* In the pessimistic scenario, the average return (μ) is adjusted downward and the volatility (σ) is increased;
* The baseline scenario keeps the parameters unchanged.

7.3.3. Cash flow application

The cash flow application module processes the regular cash flow in and out of the investment portfolio through the apply\_cashflow function. The following cash flow operations are considered in the simulation:

* Fixed withdrawal: simulates investors to withdraw a fixed amount of money regularly (such as monthly or annually), which is achieved by directly deducting from the account balance;
* Fixed contribution: used to handle the situation of regular injection of funds into the investment portfolio;
* Percentage withdrawal: withdrawal is made based on a certain percentage of the current portfolio value, reflecting the dynamic risk management needs.

The function supports inflation adjustment internally to ensure that when the cash flow operation set by the user needs to consider price changes, the amount can be adjusted correctly, thereby affecting the performance of the final investment portfolio.

7.3.4 Interaction between the Simulation Module frontend and backend

Backend process:

1. The backend API endpoint /api/simulator receives the POST request submitted by the frontend, which contains all the parameters required for the user's simulation - such as initial investment, investment period, selected simulation model (historical, statistical or parametric), cash flow type, rebalancing frequency, and selected scenario parameters. Through the build\_simulation\_params function, the system parses and verifies the data passed in by the frontend, and builds a complete parameter dictionary for subsequent simulation.
2. According to the constructed parameters and the scenario selected by the user, the backend uses the adjust\_parameters\_for\_scenario function to adjust the input parameters, and then calls the corresponding function according to the selected simulation model:

* Historical simulation: call run\_historical\_simulation (annual or periodic) and perform simulation using the loaded historical data;
* Statistical simulation: call run\_statistical\_simulation to trigger the normal distribution or GARCH model respectively;
* Parametric simulation: call the corresponding parameterized function to run the simulation.

During the simulation, each simulation path takes into account the impact of cash flow to ensure that the output portfolio value path truly reflects the inflow and outflow of investors' funds.

1. After the simulation is completed, all simulation results are integrated through make\_results\_summary to calculate various performance indicators. The backend organizes all results according to different scenarios and constructs them into structured JSON objects and returns them to the frontend.

Front-end process:

1. Users input various simulation parameters (initial investment, investment period, selected simulation model, cash flow settings, and asset allocation data) through the React-based InvestorForm component. When the user clicks the "Run Simulation" button, the front-end first verifies the input data to ensure that the investment amount is positive and the sum of the asset allocation weights is 100%.
2. The front-end uses Axios to send the encapsulated JSON request to the back-end /api/simulator endpoint, and attaches a JWT Token for authentication. Once the simulation results returned by the back-end are received, the front-end will parse the JSON data and update the internal state (such as predictionData), allowing users to view the performance charts and performance indicators of the portfolio over time under different scenarios on the results page.
3. Front-end components such as PredictionChart, tables, and other information display components will visualize the data returned by the back-end and display the simulation results under different scenarios (benchmark, optimistic, pessimistic). Users can not only see the curve of the portfolio value over time, but also intuitively compare the differences between various indicators under different scenarios.

7.4 Implementation of Multi-Portfolio Backtesting Project

The functionality uses the multi\_portfolio\_backtest function to handle requests to the /multi\_portfolio\_backtest endpoint, which accepts POST requests.

* Input data is sent in JSON format, in the request body of the POST request. The data includes parameters such as start and end year, initial capital, rebalancing frequency, dividend reinvestment flag, withdrawal amount, and portfolio configuration (weights of different assets in each portfolio).
* The backend performs validation checks on the input data. If no portfolio is provided, the backend responds with an error message. The backend also returns an error if the date range is invalid or there is no data for the specified period.
* Once the input data is validated, the backend gets the asset data for the specified date range from a preloaded DataFrame (assets\_monthly) that contains historical prices for various assets.
* For each portfolio configuration, the backend extracts the weights and symbols, calculates the monthly returns, and calculates the performance of the portfolio using the compute\_performance\_metrics\_with\_withdrawals function. Performance metrics include final balance, compound annual growth rate (CAGR), standard deviation, maximum drawdown, Sharpe ratio, and Sortino ratio.

The backend returns a JSON response containing two key components, performance\_summary: a summary of the portfolio performance. and growth\_series: the growth trajectory of each portfolio over time. The backend also includes an error handling mechanism that catches exceptions and returns appropriate error messages if there are problems during processing.

The frontend uses React's useState hook to manage multiple states, such as input parameters and backtest results. The user interface provides form fields that allow users to enter the start and end years, initial capital, withdrawal amount, and portfolio weights. Users can enter the start year, initial capital, rebalancing frequency, and adjust portfolio weights.

A table is also provided to visualize the portfolio weights, allowing users to adjust the weight of each asset in each portfolio.

When the user clicks the "Run Backtest" button, the frontend builds the payload of the backtest request. This payload includes the data entered by the user, including portfolio weights, start/end dates, and other parameters.

The frontend sends the data to the /api/multi\_portfolio\_backtest endpoint of the backend via a POST request. HTTP requests are handled using the axios library.

Once the backend processes the request and returns the result, the frontend updates the status to display the performance summary and growth series data. The performance summary is displayed in a table format, where each row represents the performance indicator of a specific portfolio. The growth series is displayed in the form of a line chart, using the Line component in react-chartjs-2, showing the growth of each portfolio over time.

8.0 Collection of data

8.1 SCFP2022 Dataset

I first collected user data from the SCFP2022 dataset, which is a comprehensive dataset containing user financial status and demographic information [49], covering key analyzable data such as user asset allocation, income, Asset, age, marital status, and education level.

So I decided to use this dataset to calculate the user risk tolerance index. Calculate the user's risk-free assets, in the fields LIQ, CDS, SAVBND, CASHLI, which represent Liquid Assets, Certificates of Deposit, Savings Bonds and Cash Holdings, respectively.

And risky assets, fields NMMF, STOCKS, BOND, where NMMF represents Total value of directly held pooled investment funds held by household [49], as part of calculating the total value of risky assets. And the risk tolerance index (RT) is then computed as Risky Assets / (Risky Assets+Risk-Free Assets).

After calculating these measures, rigorous data cleaning is performed to remove any records containing missing or invalid values (such as NaN or infinite values). For subsequent modeling and analysis, only a subset of essential variables—such as AGE, EDCL (education level), MARRIED, KIDS, OCCAT1 (occupation category), INCOME, YESFINRISK (indicates prior financial risk exposure), ASSET, and RT—is retained. In addition, before applying transformations (like one-hot encoding), an ExtraTreesRegressor is used to evaluate feature importance, ensuring that the most influential predictors of risk tolerance are highlighted.

8.2 Market Data

The market data I use is stored in a CSV file (stock\_data\_with\_sector.csv), which contains information such as stock prices, industry classifications, and stock codes. I found this S&P500 Daily Update dataset from Kaggle [50], but I processed it and added the industry of each stock. The reason I added the industry is to prepare for the model of stock simulation later, even if this content is not used in the current project.

* The first few lines contain metadata such as industry classifications and stock codes, which are used to establish mapping relationships later.
* The main body of the data records daily stock price information and has a date field, which is used to construct time series data of asset prices.

The reason why none of my modules use yfinance to obtain stock data in real time is that its real-time data interface has call frequency, data request and traffic restrictions, which makes it difficult to meet the needs of large-scale and high-frequency data updates, and may lead to a decrease in data continuity and accuracy, thereby affecting historical backtesting and long-term analysis; the use of pre-processed CSV files not only ensures the consistency and stability of data sources, versions and formats, but also loads data at one time when the system is initialized, avoiding network delays and interface instability, thereby improving overall operating efficiency and result reproducibility.

8.3 Market Data Processing

Now that we have the CSV file of market data, we need to convert and clean the stock data.

1. First, use the pandas.read\_csv() method to read the CSV data stored in the file system. Then extract the industry information (the row) and the stock code mapping relationship (ticker mapping) from the first few rows to facilitate the subsequent conversion of wide format data to long format. Extract the main part of the data (starting from row 4) and convert the date field to date and time format.
2. Use the pd.melt method to convert the data from wide format to long format, so that each row represents a record corresponding to "date-stock" and comes with price information (such as closing price, opening price, highest price, trading volume, etc.). Reorganize the long format data into a multidimensional data table through the pivot\_table method to construct a data set containing date, stock code, industry and corresponding price indicators. Convert the price data to numeric type and remove missing data (such as null values ​​or values ​​that cannot be converted). Forward fill and backward fill the price data to ensure the continuity of the time series.
3. Use the resample('M') method to convert daily data into monthly price data, and select the last data point of each month to ensure the smoothness and consistency of time series data. Finally, use the pct\_change() function to calculate the monthly yield, and calculate the mean and covariance based on the monthly data, and then annualize (multiply by 12).

The purpose of the entire market data processing process is to provide an accurate and stable historical data basis for subsequent yield calculations, covariance matrix construction, and portfolio optimization, thereby supporting various financial analysis tasks such as risk assessment, asset allocation, and portfolio optimization.

9.0 Analysis of result and Demonstration

9.1 Model Evaluation and Data Visualization

After reading the raw data from SCFP2022.xlsx [21], we calculated the risky assets (Risky) and risk-free assets (RiskFree). Based on the previous content, we finally retained the 9 core fields of AGE, EDCL, MARRIED, KIDS, OCCAT1, INCOME, YESFINRISK, ASSET, and RT through feature importance or business logic, and then divided the 9 retained fields by type:

* Category: EDCL, MARRIED, OCCAT1, YESFINRISK (count the frequency of each category through countplot).
* Discrete numeric: KIDS (using histogram histplot, but without KDE).
* Continuous numeric: AGE, INCOME, ASSET, RT (using histogram + KDE curve).

Then, after one-hot encoding the 'category' variable, we can find that the original category column is split into multiple binary (0 or 1) columns, such as EDCL\_1 and EDCL\_2, and each new column indicates whether the record belongs to a certain category.

图表, 条形图, 直方图

描述已自动生成

Figure 9.1 (1) Feature Distribution

In figure 9.1 (1), we can easily find that, from the distribution of AGE, we can observe that the age of the sample group is concentrated in the middle-aged, young or elderly groups. MARRIED\_1 is significantly higher than MARRIED\_2, and the proportion of married people is higher. The income and asset distribution tends to be normal after manual logarithmic transformation.

The overall risk tendency is mostly concentrated between 0 and 0.1. For this result, I went back to refer to the data set and the US Consumer Finance Survey [21], and found that only a few families hold a large amount of risky assets, while the vast majority of families rely mainly on low-risk investments (such as cash and savings), and do not even touch risky investments such as stocks and funds.

Below is overall model performance, the trained ExtraTreesRegressor achieves a highest R² score.

图表, 条形图

描述已自动生成

Figure 9.1 (2) Overall model R² score

For the model evaluation results, we can see in the below table.

| **Metric** | **Training Value** | **Validation Value** |
| --- | --- | --- |
| **R²** | 0.9873 | 0.9275 |
| **MSE** | 0.0016 | 0.0090 |
| **RMSE** | 0.0399 | 0.0950 |
| **MAE** | 0.0179 | 0.0439 |
| **F1 Score** | 0.8907 | 0.7394 |

Table 9.1 The model evaluation results

The training and validation set R² are very high, indicating that the model fits the training data very well. The MSE and RMSE are significantly lower for the training set compared to the validation set. This indicates that while the model performs well on the training set, its performance degrades on the unseen validation set.

I initially discretized the target variable RT in the risk tolerance regression task and separated it into 5 levels so that I could use the F1 score to assess the model's performance in the classification job. With a strong balance of precision and recall, the model performs well in classification on the training set, as indicated by the training set's F1 score of 0.8907. Although the model performs well on the training set, its generalization ability is poor on unknown data, as indicated by the validation set's F1 score of 0.7394.

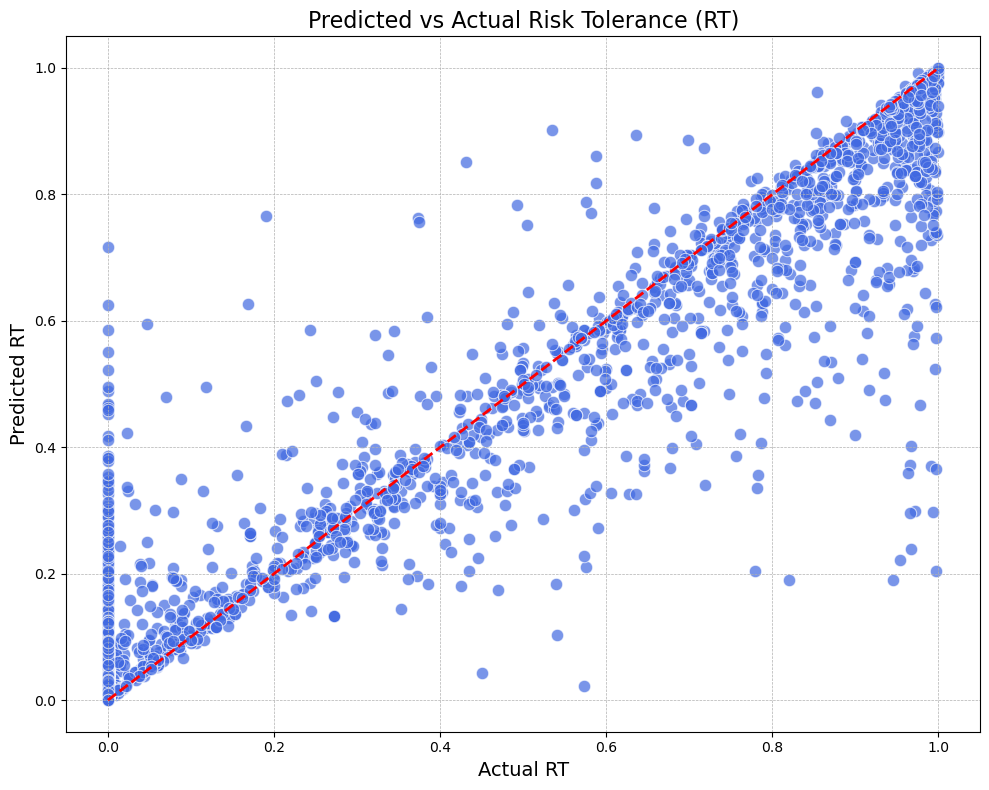


Figure 9.1 (3) Predicted vs Actual RT

From the 9.1 (3) Predicted vs Actual RT plot, we can see that the model’s predictions are close to actual values. The ExtraTreesRegressor model performs well on the validation set and is able to capture the pattern of the target variable well.

图表, 条形图

描述已自动生成

Figure 9.1 (4) feature importance

The model’s feature importance analysis in Figure 9.1 (4) shows that ASSET and INCOME are the most important features, ranking first and second. This is also in line with intuition, indicating that financial resources have a significant impact on risk tolerance (RT). Age (AGE) and EDCL\_4 are also at the top, indicating that an individual’s life stage and economic classification also have a certain effect on risk tolerance.

9.2 Result Reflection

Through the results of this analysis, I found that financial stability plays a dominant role in determining individual risk tolerance (RT), that is, assets and income. Because financial stability is like a safety net for them to absorb potential losses.

From a policy perspective, increasing asset accumulation and income stability, especially among low-income groups, can encourage wider participation in economic activities, thereby promoting economic growth and innovation.

And according to a 2024 report from FINRA’s Investor Education Foundation, having greater risk knowledge may make consumers more willing to take financial risks, or conversely, consumers who are more willing to take financial risks are also more likely to understand the risks. And our analysis results also confirm this view. In our result, the model considers age and YESFINRISK as influencing factors. Even though it can still be seen that their influence is less than tangible financial resources. EDCL\_4 indicates a high level of education, while EDCL\_2 indicates a lower level of education. Through the results, we can observe that individuals with high education levels usually have more resources and information to understand and manage risks. Therefore, they tend to have a higher risk tolerance. On the contrary, individuals with low education levels may face greater uncertainty and economic pressure, which may limit their risk tolerance. This observation shows that education not only affects individuals' knowledge and skills, but also indirectly affects their financial decisions and risk preferences by improving their economic status and cognitive abilities. Therefore, by improving the education level, it may help to improve the risk tolerance of the low-educated groups, thereby enhancing their sense of economic participation and innovation.

We can observe that, the assessment of personal risk tolerance is affected by multiple factors, but the impact of financial resources (assets and income) and education level is particularly significant. Improving personal asset accumulation and income stability, as well as improving education level, can not only enhance the ability of individuals to cope with financial risks, but also promote the stability and innovative development of the overall social economy.

Therefore, our project pays more attention to economically disadvantaged groups, and by providing users with effective asset growth plans and investment education resources, more people can have a stronger sense of financial security and risk management capabilities.

9.3 User Portrait and Asset Allocation Demonstration

Click User Portrait on the left sidebar to enter our function of optimizing user portfolios based on user portraits. Clicking it will enter the interface as shown in the picture below. This interface allows users to enter investor characteristics. I use sliders to improve the user experience. Users can easily scroll the mouse to enter age, current assets, income, education level, marital status, number of children, occupation type, and financial risk they are willing to take. Each slider has an icon and value prompt for users to make intuitive choices.

After ensuring that all information is filled in correctly, click the "Calculate Risk Tolerance" button to calculate the user's Risk Tolerance based on the user information.

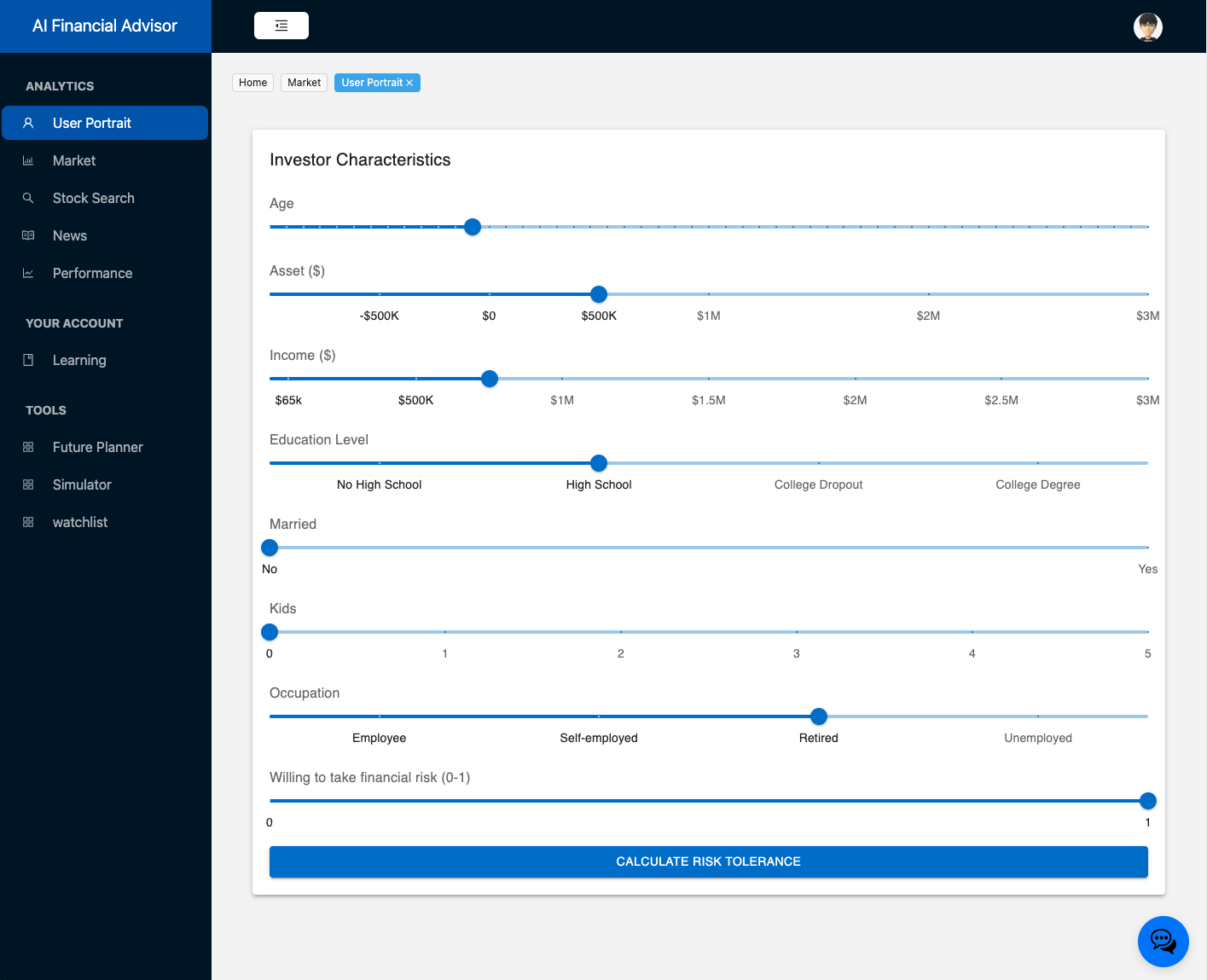


Figure 9.3 (1) Investor Characteristics

Click “Calculate Risk Tolerance” and get Risk Tolerance result:

图形用户界面, 应用程序

描述已自动生成

Figure 9.3 (2) Calculate Risk Tolerance

After system calculate risk tolerance, users can Select Stocks for Portfolio allocate:

背景图案

低可信度描述已自动生成

Figure 9.3 (3) Select Stocks

Click “Allocate Assets”. The system will dynamically generate a recommended asset allocation based on the risk tolerance score and selected assets. The page will display a pie chart showing the proportion of each asset in the portfolio. Different colors represent different assets, and the corresponding stock code and percentage will be marked in the chart. And the results can be divided into three parts, generally. 4-5 = Aggressive Investor; 2-3 = Balanced Investor; 0-1 = Conservative Investor.

If the stock code needs to be modified, the user can return to the previous step and recalculate.

图形用户界面, 文本

中度可信度描述已自动生成

Figure 9.3 (4.1) The Relationship between Assets Allocation and Risk Tolerance

图表, 饼图

描述已自动生成

Figure 9.3 (4.2) The Relationship between Assets Allocation and Risk Tolerance

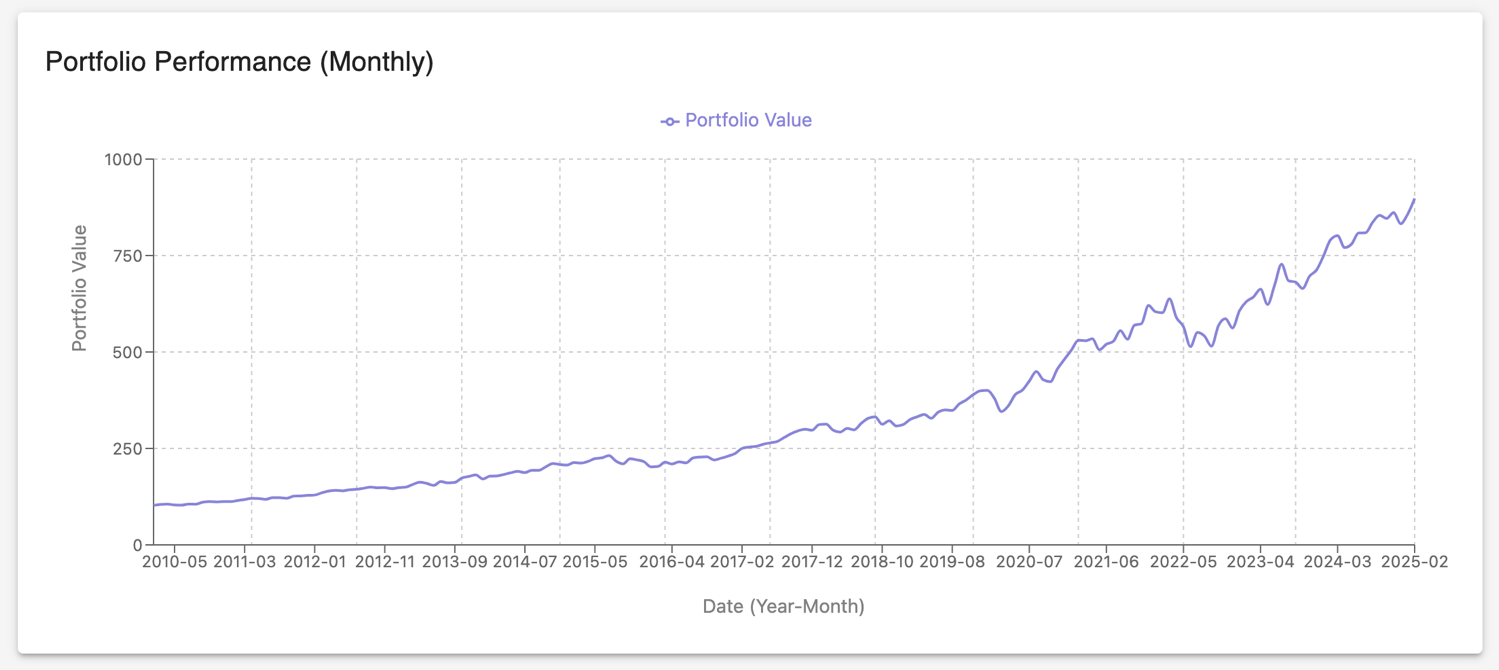
The system will then automatically display a line chart showing the cumulative value of the portfolio over time. The horizontal axis is the date (year-month), and the vertical axis shows the cumulative return of the portfolio. Users can hover their mouse over the chart to view the specific value.  


Figure 9.3 (5) result and performance of the portfolio

If the number of stocks selected by the user is >1, the *web*page will automatically calculate an efficient frontier graph, showing the efficient frontier curve, ineffective frontier, capital market line, and tangent portfolio positioning.

This graph intuitively shows the expected return of each portfolio at different risk levels, helping users identify the best risk-return balance point, and can also be used as a reference to help users develop better investment strategies.

图表, 折线图

描述已自动生成

Figure 9.3 (6) Efficient Frontie of the portfolio

9.4 Multi-Portfolio Backest Demonstration

At the top of the page users will see a card called "Parameters". This section allows the user to configure the overall settings for the backtest.

First is the Ticker (comma separated). The user can enter the tickers (ticker symbols) that they would like to include in the portfolio. For example: AAPL, GOOGL, NVDA, MSFT, TSLA. After entering or modifying the ticker symbol, click the "Update Ticker Symbol" button. This will refresh the available assets in the weight table below.

The Start Year/Start Month defines when the user's backtest will start, and the system will start fetching historical data from this point in time.

The End Year/End Month defines when the backtest will end.

Enter the starting capital amount for each portfolio (e.g., 10,000). For this feature we assume that all portfolios have the same initial capital.

The user can also select how often the portfolios should be rebalanced. The available options are "None", "Monthly", "Quarterly", and "Yearly". For example, selecting "Monthly" will automatically adjust the weights of each portfolio to its original distribution at the beginning of each month. This works in conjunction with the Withdrawal Amount to set a fixed amount that users withdraw from each portfolio on a regular basis (according to the selected rebalancing schedule).

Below the parameters card is the Portfolio Weights table. Users can specify the asset allocation percentage for each portfolio.

图形用户界面

描述已自动生成

Figure 9.4 (1) Performance Summary and Portfolio Growth of Multi-Portfolio Demo

After the user clicks the "Run Backtest" button, the system will calculate the portfolio performance indicators and growth series. After the backtest is successful, the user will see the Performance Summary and the portfolio growth chart.

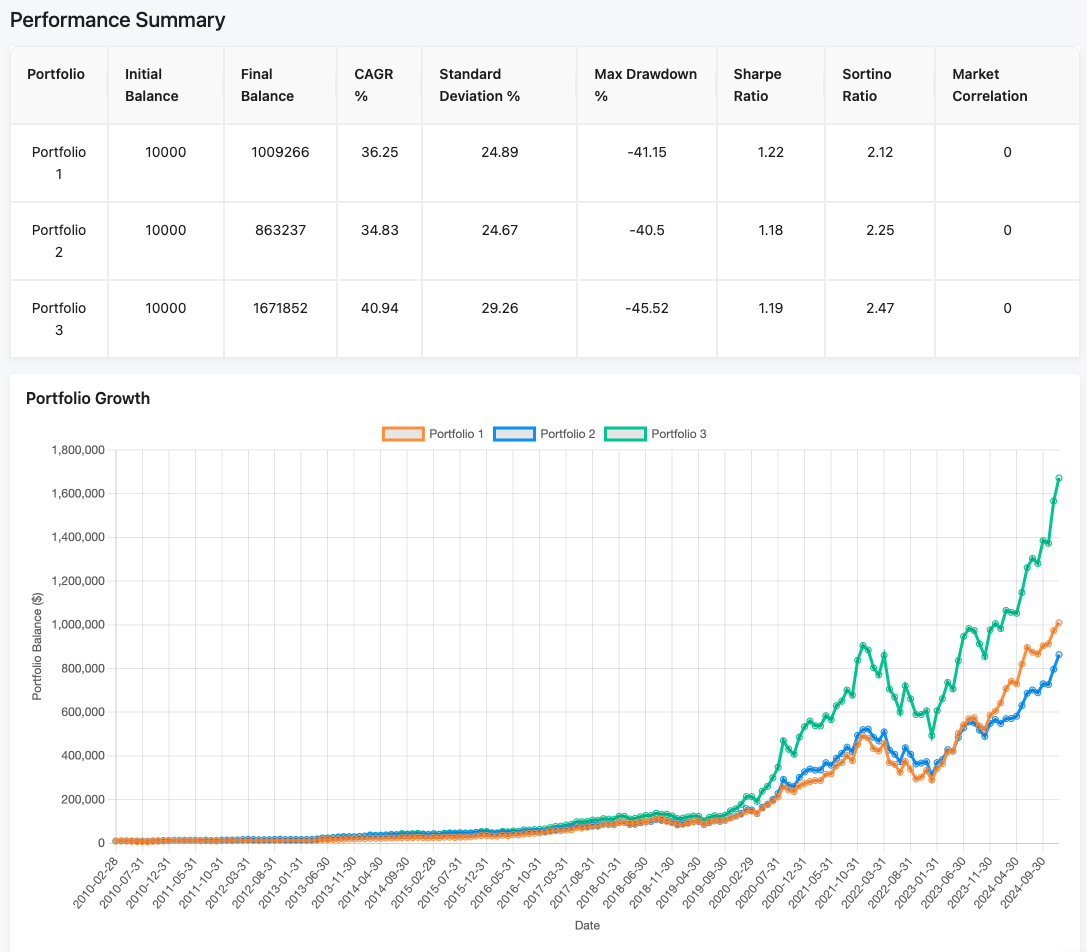


Figure 9.4 (2) Multi-Portfolio Backest Demo

9.5 Simulator Demonstration

This simulator provides Monte Carlo simulation functions for multiple investment portfolios, including:

* Historical (based on historical data simulation)
* Parameterized (based on parameterized distribution, such as normal distribution or lognormal distribution)
* Statistical (based on statistical models, such as Normal/GARCH models, etc.)

At the same time, the simulator supports parameter adjustment for three scenarios: baseline, optimistic, and pessimistic, and can output rich performance indicators based on different asset weight configurations.

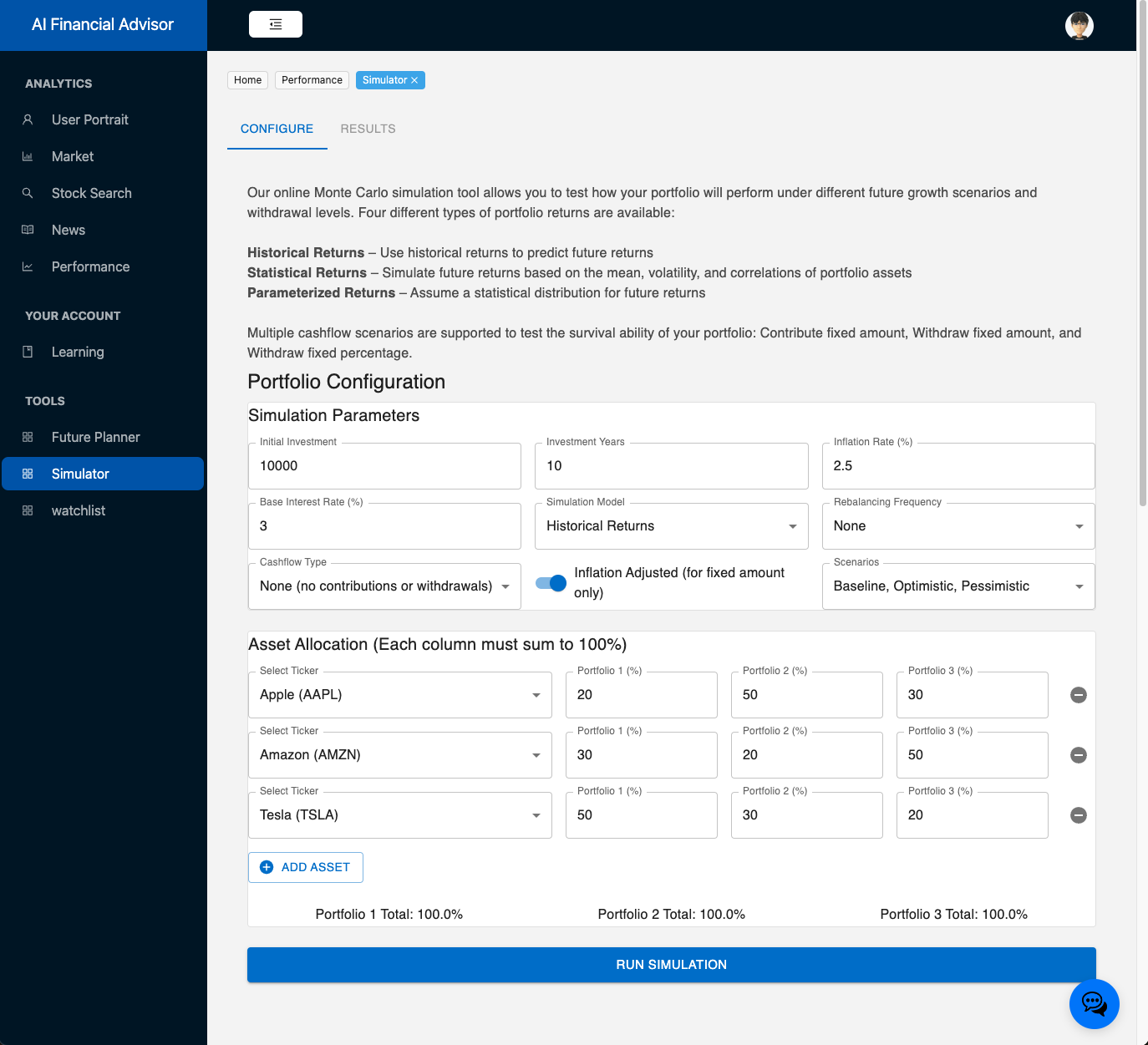


Figure 9.5 (1) Simulator Configure page Demo

After opening the page, the user will see an interface with two main tabs: Configure and Results (disabled by default, unlocked after configuring and running the simulation).

Under the Configure tab, an investor configuration form will be displayed on the page, allowing the user to fill in the initial amount, investment period, annual cash flow settings, asset weight distribution, expected rate of return and other parameters, and then click "Run Simulation" to initiate a backend request and obtain simulation results with one click. Among them:

* Simulation Model can choose historical, parameterized, statistical, etc.
* For Time Series Model, if statistical is selected, normal or garch can be selected.
* Cashflow settings can choose whether to withdraw/add regularly, whether to adjust with inflation, etc.

After filling in or selecting the above parameters, click Run Simulation. The front end will automatically switch to the "Results" tab and allow users to click into "Results" to view the results.

图表

描述已自动生成

Figure 9.5 (2) Simulator Results page Demo

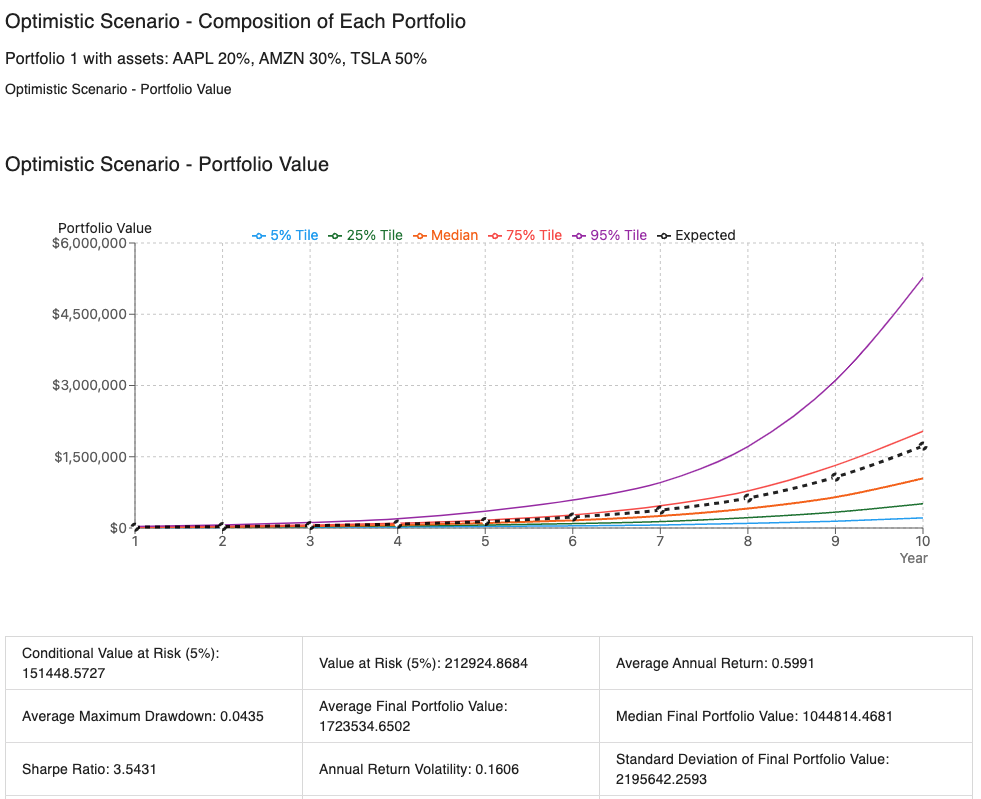


Figure 9.5 (3) Simulator Results page Demo

In the Results tab, the page will be grouped by Scenario or Portfolio: Baseline Scenario / Optimistic Scenario / Pessimistic Scenario. In each scenario, the corresponding portfolio will be listed, as well as the simulation result chart and performance indicators of the portfolio. Users will see:

* Portfolio composition (Composition): Displays each asset in each portfolio and its corresponding weight (such as AAPL 20%, AMZN 30%, TSLA 50%).
* Simulated trend chart (Portfolio Value): The simulated portfolio 5%/25%/50%/75%/95% quantiles and expected values ​​(Expected) are plotted together on the line chart to intuitively understand the potential asset value range in different years.
* Performance Metrics (performance indicators): including average terminal value, annualized rate of return, maximum drawdown, VaR, CVaR, Sharpe Ratio, etc., displayed in table form.

In this interface, users can compare the three scenarios of Baseline, Optimistic and Pessimistic, and view the performance of asset portfolios under different assumptions in the same report, helping to make a more comprehensive risk-return analysis.

10.0 Conclusion

This project effectively illustrated how financial technology and artificial intelligence are being integrated in the domains of portfolio management and financial consultancy.

Accurately determining consumers' risk tolerance with machine learning models, it achieves performance metrics such as a balanced F1 value, low error rate, and of 0.92 on the validation set. Additionally, it employs convex optimization for future planning and dynamic asset allocation to assist users in analyzing in various future economic contexts. My team members concentrate on additional components, such the spending planner, stock overview, and market overview. Improving users' understanding of markets and finances. To make the interface snappy and easy to use, the system is isolated from the front and back ends.

The intelligent financial advisor system not only lowers the threshold for the public to obtain high-quality financial advice, but also provides an effective and efficient solution to the ever-changing financial planning challenges, and paves the way for improving financial inclusion and accessibility.

Reference

1. D. Goodsell, “2024 global retirement index,” *Natixis Investment Managers,* 2024, [Online]. Available: <https://www.im.natixis.com/en-us/insights/investor-sentiment/2024/global-retirement-index>
2. J. Wade, “How much does a financial advisor cost?,” *Wealthtender*, [Online]. Available: <https://wealthtender.com/insights/financial-planning/how-much-does-a-financial-advisor-cost/>
3. E. Dunne, “Strange behavioural biases that will skew your investments,” *The Times & The Sunday Times*, Sep., 2024, [Online]. Available: <https://www.thetimes.com/world/ireland-world/article/strange-behavioural-biases-that-will-skew-your-investments-nv2lngzvj>
4. A. Fontes, O. Valdes, and G. Mottola, “How consumers think about investment risk”, *FINRA Investor Education Foundation,* 2024, [Online]. Available: <https://www.finrafoundation.org/sites/finrafoundation/files/2024-12/how-consumbers-think-about-investment-risk.pdf>.
5. T. Choukhmane and T. de Silva, *What drives investors’ portfolio choices? separating risk preferences from frictions*, May 2024. doi:10.3386/w32476
6. L. Duan, Y. Qu, and Y. Li, “Gas pedal or stabilizer: The mystery of financial development affecting financial fraud revisited,” *Modern Economy*, vol. 16, no. 01, pp. 81–113, 2025. doi:10.4236/me.2025.161004
7. D. Belanche, L. V. Casaló, and C. Flavián, “Artificial Intelligence in FinTech: understanding robo-advisors adoption among customers,” *Industrial Management & Data Systems*, vol. 119, no. 7, pp. 1411-1430, 2019.
8. J. E. Fisch, M. Laboure, and J. A. Turner, “The emergence of the robo-advisor,” *The Disruptive Impact of FinTech on Retirement Systems*, vol. 13, pp. 13-37, 2019.
9. M. Adam, J. Toutaoui, N. Pfeuffer, and O. Hinz, “Investment decisions with robo-advisors: The role of anthropomorphism and personalized anchors in recommendations,” in *Proc. 27th Eur. Conf. Inf. Syst. (ECIS)*, Stockholm & Uppsala, Sweden, Jun. 8-14, 2019. [Online]. Available: <https://aisel.aisnet.org/ecis2019_rp/33>.
10. Skyquest, “Robo advisory market trends, Size, share & forecast: 2032,” Robo Advisory Market Trends, Size, Share & Forecast | 2032, https://www.skyquestt.com/report/robo-advisory-market
11. C. S. Tucker and H. M. Kim, “Data-driven decision tree classification for product portfolio design optimization,” *Journal of Computing and Information Science in Engineering*, vol. 9, no. 4, Nov. 2009. doi:10.1115/1.3243634
12. P. P. Panda, M. K. Gharanchaei, X. Chen, and H. Lyu, “Application of deep learning for factor timing in asset management,” arXiv.org, https://arxiv.org/abs/2404.18017
13. X. Song, Y. Liu, L. Xue, J. Wang, J. Zhang, J. Wang, and Z. Cheng, “Time-series well performance prediction based on Long Short-Term Memory (LSTM) neural network model,” *Journal of Petroleum Science and Engineering*, vol. 186, p. 106682, 2020.
14. E. J. Elton, M. J. Gruber, S. J. Brown, and W. N. Goetzmann, *Modern Portfolio Theory and Investment Analysis*. John Wiley & Sons, 2009.
15. S. Yu, H. Wang, and C. Dong, "Learning Risk Preferences from Investment Portfolios Using Inverse Optimization," arXiv:2010.01687, Oct. 2020. [Online]. Available: https://doi.org/10.48550/arXiv.2010.01687
16. T. Campbell, “Betterment 2025 review: Pros, cons and alternatives,” *Business Insider,* [Online]. Available: <https://www.businessinsider.com/personal-finance/investing/betterment-review>.
17. T. Campbell, “Wealthfront 2025 review: Pros, cons and alternatives,” *Business Insider*, [Online]. Available:<https://www.businessinsider.com/personal-finance/investing/wealthfront-review>.
18. B. Baker, “Sigfig Review 2023,” *Bankrate*, [Online]. Available <https://www.bankrate.com/investing/roboadvisor-reviews/sigfig/#in-the-details>.
19. S. McKeown, “Hidden costs of traditional advisors and how to avoid them,” *Southern Pacific Asset Management*, [Online]. Available: <https://www.southernpacificam.com/blog/hidden-costs-of-traditional-advisors-and-how-to-avoid-them/>.
20. X. Wang and Z. Mao, "Research on the Impact of Digital Inclusive Finance on the Financial Vulnerability of Aging Families," Risks, vol. 11, no. 12, p. 209, Nov. 2023, doi: 10.3390/risks11120209.
21. Survey of Consumer Finances (SCF), “Changes in U.S. family finances from 2019 to 2022: Evidence from the survey of consumer finances,” *Federal Reserve Bulletin*, Oct. 2023. doi:10.17016/8799
22. J. A. Samuels, “One-Hot Encoding and Two-Hot Encoding: An Introduction,” ResearchGate, Jan. 2024. [Online]. Available: https://doi.org/10.13140/RG.2.2.21459.76327
23. P. Geurts, D. Ernst, and L. Wehenkel, “Extremely randomized trees,” *Machine Learning*, vol. 63, pp. 3–42, 2006.
24. T. Miller, I. Durlik, A. Łobodzińska, L. Dorobczyński, and R. Jasionowski, “AI in Context: Harnessing Domain Knowledge for Smarter Machine Learning,” Applied Sciences, vol. 14, no. 24, p. 11612, Dec. 2024, doi: 10.3390/app142411612.
25. “An evaluation of the association between marital status and Financial Risk Tolerance,” *Financial Planning Association*, [Online]. Available: <https://www.financialplanningassociation.org/learning/publications/journal/JUL22-evaluation-association-between-marital-status-and-financial-risk-tolerance-OPEN>.
26. Y. Bayar, H. F. Sezgin, Ö. F. Öztürk, and M. Ü. Şaşmaz, “Financial literacy and financial risk tolerance of individual investors: Multinomial logistic regression approach,” SAGE Open, vol. 10, no. 3, 2020. [Online]. Available: <https://doi.org/10.1177/2158244020945717>.
27. H. Doan and S. Yeasmin, The Impact of Demographic Factors on Financial Risk Tolerance: a replication study, Master’s thesis, Dept. of Business Studies, Uppsala Univ., Sweden, 2024. [Online]. Available: <https://www.diva-portal.org/smash/get/diva2:1891472/FULLTEXT01.pdf>.
28. P. Refaeilzadeh, L. Tang, and H. Liu, “Cross-validation,” in *Encyclopedia of Database Systems*, L. Liu and M. T. Özsu, Eds. Boston, MA, USA: Springer, 2009. [Online]. Available: <https://doi.org/10.1007/978-0-387-39940-9_565>.
29. T. Hodson, T. M. Over, and S. Foks, “Mean squared error, deconstructed,” J. Adv. Model. Earth Syst., vol. 13, no. 12, Dec. 2021. [Online]. Available: <https://doi.org/10.1029/2021MS002681>.
30. F. Sfravara, E. Barberi, G. Bongiovanni, M. Chillemi, and S. Brusca, “Development of a Predictive Model for Evaluation of the Influence of Various Parameters on the Performance of an Oscillating Water Column Device,” *Sensors*, vol. 24, no. 11, p. 3582, Jun. 2024. [Online]. Available: <https://doi.org/10.3390/s24113582>
31. T. Chai and R. R. Draxler, “Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature,” Geosci. Model Dev., vol. 7, pp. 1247–1250, 2014. [Online]. Available: <https://gmd.copernicus.org/articles/7/1247/2014/gmd-7-1247-2014.pdf>
32. D. M. W. Powers, “Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation,” *arXiv preprint* arXiv:2010.16061, Oct. 2020. [Online]. Available: <https://arxiv.org/abs/2010.16061>.
33. F. Pedregosa, G. Varoquaux, et al., “RandomizedSearchCV,” *Scikit-learn Documentation*, version 1.6.1. [Online]. Available: <https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html>
34. G. James, D. Witten, T. Hastie, R. Tibshirani, and J. Taylor, “Linear regression,” in *An Introduction to Statistical Learning: With Applications in Python*, Cham: Springer International Publishing, 2023, pp. 69–134.
35. L. Breiman, “Random forests,” *Machine Learning*, vol. 45, pp. 5–32, 2001.
36. H. Taud and J. F. Mas, “Multilayer perceptron (MLP),” in *Geomatic Approaches for Modeling Land Change Scenarios*, 2018, pp. 451–455.
37. T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Aug. 2016, pp. 785–794.
38. J. Hull, Options, Futures, and Other Derivatives, 9th ed., Pearson Education, 2015.
39. A. Avella, “Real-World Applications of Markowitz’s Portfolio Optimization: A Quantitative Study,” SSRN Electron. J., Sep. 2024. [Online]. Available: <https://doi.org/10.2139/ssrn.4971433>.
40. E. J. Elton, M. J. Gruber, S. J. Brown, and W. N. Goetzmann, Modern Portfolio Theory and Investment Analysis, 9th ed., Wiley, 2014.
41. H. Dong, Data Analytics in Finance, 1st ed. Boca Raton, FL, USA: Auerbach Publications, 2025. [Online]. Available: <https://doi.org/10.1201/9781003620198>.
42. U.S. Department of the Treasury, "Daily Treasury Yield Curve Rates," Apr. 10, 2025. [Online]. Available: <https://home.treasury.gov/policy-issues/financing-the-government/interest-rate-statistics>.
43. D. P. Kroese, T. Brereton, T. Taimre, and Z. I. Botev, “Why the monte Carlo method is so important today,” *WIREs Computational Statistics*, vol. 6, no. 6, pp. 386–392, Jun. 2014. doi:10.1002/wics.1314.
44. J. Wang, “Generating daily changes in market variables using a multivariate mixture of normal distributions,” in Proc. Winter Simulation Conf., vol. 1, Jan. 2001, pp. –, doi: 10.1109/WSC.2001.977286.
45. T. Bollerslev, “Generalized autoregressive conditional heteroskedasticity,” Journal of Econometrics, vol. 31, no. 3, pp. 307–327, 1986. doi: 10.1016/0304-4076(86)90063-1
46. F. J. Fabozzi, Encyclopedia of Financial Models, vol. 2, Hoboken, NJ, USA: Wiley, 2013.
47. K. Zucchi, “Lognormal and normal distribution,” *Investopedia*, Oct. 28, 2024, [Online]. Available:<https://www.investopedia.com/articles/investing/102014/lognormal-and-normal-distribution.asp>.
48. C. J. Date, An Introduction to Database Systems, 8th ed. Boston, MA, USA: Addison-Wesley, 2003.
49. J. F. Albert and N. Gómez-Fernández, “The impact of monetary policy shocks on net worth and consumption across races in the United States,” *Economic Systems*, vol. 48, no. 1, p. 101178, 2024.
50. Yash, “S&P500 Daily Update dataset,” *Kaggle*. [Online]. Available: <https://www.kaggle.com/datasets/yash16jr/s-and-p500-daily-update-dataset>.