
Research article

A portfolio recommendation system based on machine learning and big data analytics

Man-Fai Leung^{1,*}, Abdullah Jawaid², Sai-Wang Ip², Chun-Hei Kwok² and Shing Yan²

¹ School of Computing and Information Science, Faculty of Science and Engineering, Anglia Ruskin University, Cambridge, UK

² School of Science and Technology, Hong Kong Metropolitan University, Hong Kong, China

* **Correspondence:** Email address: man-fai.leung@aru.ac.uk

Abstract: This research paper introduces a portfolio recommendation system that utilizes machine learning and big data analytics to offer a profitable stock portfolio and stock analytics via a web application. The system's effectiveness was evaluated through backtesting and user evaluation studies, which consisted of two parts: user evaluation and performance evaluation. The findings indicate that the development of a machine learning-based portfolio recommendation system and big data analytics can effectively meet the expectations of the majority of users and enhance users' financial knowledge. This study contributes to the growing body of research on utilizing advanced technologies for portfolio recommendation and highlights the potential of machine learning and big data analytics in the financial industry.

Keywords: reinforcement learning; stock market; trading; portfolio; finance

1. Introduction

Asset allocation Leung & Wang (2021) is one of the most prevalent investment strategies that require a certain level of financial knowledge to build a sound stock portfolio. However, many people fail to generate capital gains when investing in the stock market due to various factors. For instance, they may lack prior experience with stock trading, making it challenging for them to make informed decisions. In addition, a higher level of financial literacy is often desired, but many individuals may not have the necessary knowledge to make suitable investment decisions based on their limited understanding.

In general, there are two methods of investment: online and offline. The offline method tends to be slower and more expensive, as investors need to communicate with a broker about their investments,

and the broker may charge a commission. In contrast, online trading brokers offer more convenience, as investors can make investments from anywhere, and the trading commissions are usually lower. Online platforms also provide access to investment advice, news, corporate information, and stock quotes. However, the abundance of information can be both an advantage and a disadvantage. Some investors may find it challenging to understand the vast amount of data and different investment strategies, leading to a loss of confidence in using the online method. Another issue is that when market conditions fluctuate greatly, such as before or after market hours, online platforms may encounter errors or problems due to the large volume of user input data.

Currently, there are various smart trading systems available, such as SoFi Active Invest and Fidelity. These systems invest in different types of stocks based on user preferences such as risk tolerance, income, and experience. However, investors often have limited control over their investments in these systems. For instance, SoFi Auto Invest is a smart trading system that employs AI technology to execute trades automatically. An investment portfolio recommendation system for individual e-commerce users has been proposed Li & Yu (2017). In the proposed system, the Value at risk (VaR) method is used to measure the risk level of securities and risk preference of investors. Overall, the state-of-the-art recommendation systems do not account for the low market liquidity and hence can lead to unwise investment decisions. A recommendation framework was proposed to build an investment portfolio, which results in the highest return and the lowest risk along with a statistical measure of the number of days required for the amount to be completely funded Ren & Malik (2019). Based on each firm's technology portfolio and each patent's co-classification information, Lee & Sohn (2021) proposed a novel recommendation system for firms seeking new convergence opportunities through representations of convergence items and firms.

In addition to smart trading systems, popular financial data and information platforms such as Yahoo Finance and Futu provide professional stock information. However, their information is generally not suitable for beginners and requires professional analytical skills to fully comprehend. Most of these trading systems involve stock price prediction, and there are various methods available in the literature for predicting stock prices, such as neurodynamic and metaheuristic approaches Leung & Wang (2022); Yuen et al. (2021).

This work aims to develop a machine-learning-based portfolio recommendation system that not only provides users with a profitable portfolio but also allows them to gain financial knowledge and make better financial decisions in the long term. This work also aims to address the problem of novice investors who lack the knowledge and experience to make informed investment decisions and often end up making suboptimal choices. By developing a portfolio recommendation system that utilizes machine learning algorithms to optimize investment portfolios, a simple and effective way to invest money while also learning about the stock market will be provided to users. Overall, the motivation behind this paper is to bridge the gap between novice investors and the complex world of finance by providing a user-friendly and informative system that helps users make better financial decisions. These are the contributions of this work:

- A portfolio recommendation system that utilizes machine learning techniques to generate profitable portfolios for users is proposed. The system considers various factors to generate a customized portfolio.
- A user market survey is included to evaluate the effectiveness and usability of the proposed system. The survey results suggest that the system is easy to use and provides useful recommenda-

tions that match users' preferences.

The remainder of this paper is structured as follows: Section 2 discusses the use of existing portfolio optimization platforms or investment-related applications. The lack of real portfolio optimization is described in detail, and other solutions are discussed and critiqued. Section 3 outlines the design of our application and its major components. Section 4 presents the results of the implementation of the portfolio optimization application. Finally, Section 5 concludes with a discussion of the merits and limitations of the application, as well as possible future work.

2. Related supporting technologies

To make an investment choice, algorithms and machine learning techniques may be used to uncover hidden patterns in historical stock price data Lai et al. (2020); Leung et al., (2021). In order to retrieve the data, yfinance can be used Finance Y (2020). It is a module in Python. In addition, a Python data manipulation and analysis library, Pandas, can be used to manage and analyze large amounts of historical stock price data. Furthermore, NumPy can be used together with Pandas to provide efficient functions while processing data McKinney (2012). In addition, various libraries such as Scikit-learn Pedregosa, F. et al., (2011), Keras Gulli & Pal (2017) and TensorFlow Shukla & Fricklas (2018) can be used to train machine learning and deep learning models. Some examples include training the stock prediction model using Long-Short Term Memory (LSTM) and convolution neural network (CNN) Wu et al., (2021). It should be noted that while both these models exhibited high levels of accuracy, LSTM was found to be the more accurate model Sen & Mehtab (2020). However, in terms of execution speed, the CNN model was found to be the faster, and LSTM models tend to perform better on time series data. Although most algorithms do not work well in long sequence time-series forecasting, LSTM is effective in overcoming the problem Du et al., (2021). Furthermore, flairNLP provides pre-trained models that can be used to perform sentiment analysis on sentences collected from online forums and social media. For the portfolio optimization model, it can be developed using reinforcement learning by utilizing libraries in OpenAI Gym and Stable Baselines Raffin, A. et al. (2021). A custom OpenAI Gym environment can be developed to train the agent to find the optimal portfolio weights. Additionally, Stable Baselines provides a range of reinforcement learning algorithms that we can select from for various use cases and evaluate their performance and accuracy, such as advantage actor-critic (A2C), proximal policy optimization (PPO) and others. To build a web application, we can use the Python Flask framework. It serves as a backend API server for communicating with the frontend client and can manage authentication, databases and cron jobs. ReactJS can also be utilized as a frontend UI library, enabling the creation of an interactive UI to enhance the user experience.

3. Methodology

The system of our application is based on client-server architecture, as a client/server system can minimize the development time of an application Oluwatosin, H. S. (2014). We utilize python (Flask) as our API server on the server side, which includes processing client requests and cron jobs. Furthermore, because of hierarchical data and big datasets, we utilize MongoDB as our database as there are a few relationships between datasets. To render the UI on the client side, we use the React.js Framework. Figure 1 shows the hierarchical structure of the proposed system, and Figure 2 shows the flow Diagram

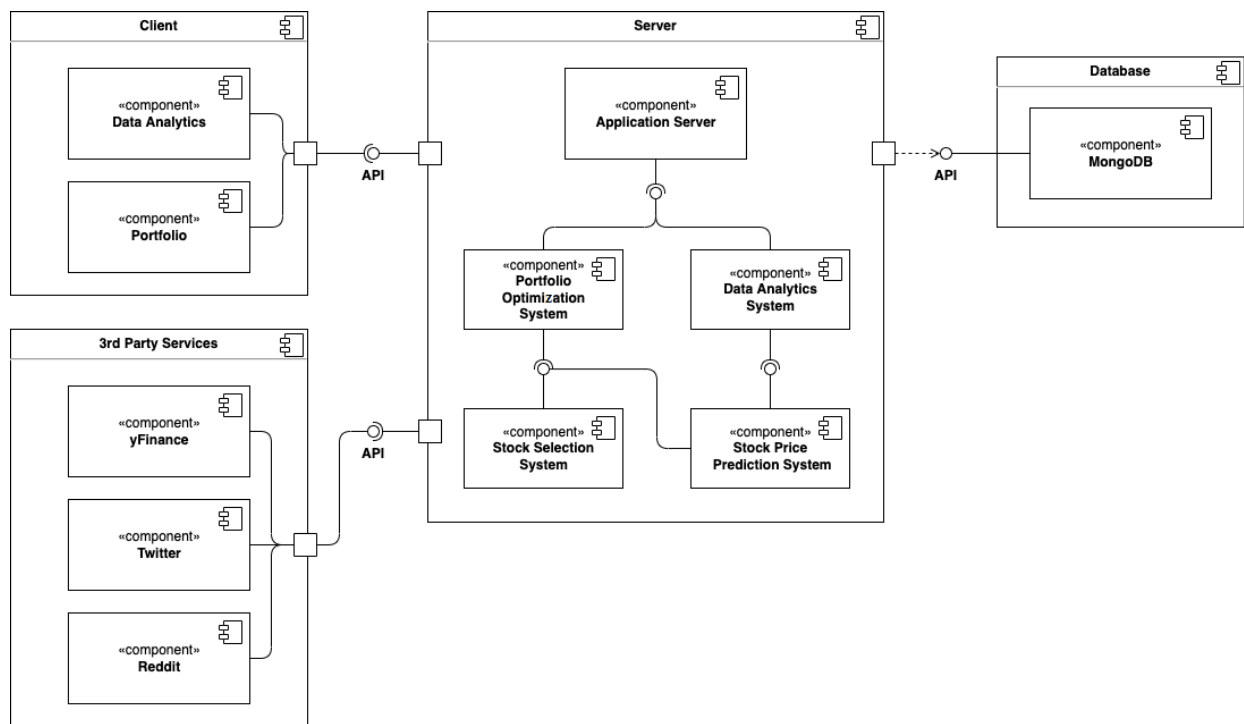


Figure 1. Hierarchical structure of the proposed system

of the proposed system. Stock analytics is another feature of our system. Users can get stock analyses that are simple to grasp with our data analytics. Beginners may perceive the risk as low, whereas professionals may see the precise percentage on the analytics. We obtain stock data from yFinance and analyze each stock in a scheduled job to accomplish our solution. In addition, there are statistics such as maximum drawdown, peg ratio, Sortino ratio, and so on. They'll sort into categories that are simple to comprehend.

3.1. Server & deployment

To start with, the Flask server and web application will be deployed to Heroku, a user-friendly cloud platform with robust Python and JavaScript support. The Flask server will also manage API calls from clients, such as data requests and user authentication. The hosted server is linked to MongoDB for data storage. Figures 3 and 4 depict the cloud deployment and client-server architecture.

3.2. Portfolio optimization

Moreover, after users register their account on our web application, they are required to answer a set of questions that we have set up to map our stock selection system. Using Euclidean distance, we find the most suitable stocks according to the user's preferences and return the result to the user for further filtering, such as removing some unwanted stocks. Then, the result is passed to the portfolio optimization system. The portfolio optimization system is based on reinforcement learning, which requires creating an environment in which an agent can perform actions and learn iteratively. Figure 5 shows the flow of the reinforcement learning model applied in portfolio optimization. One of the most significant functions in the process is the step function, which involves taking actions in the given

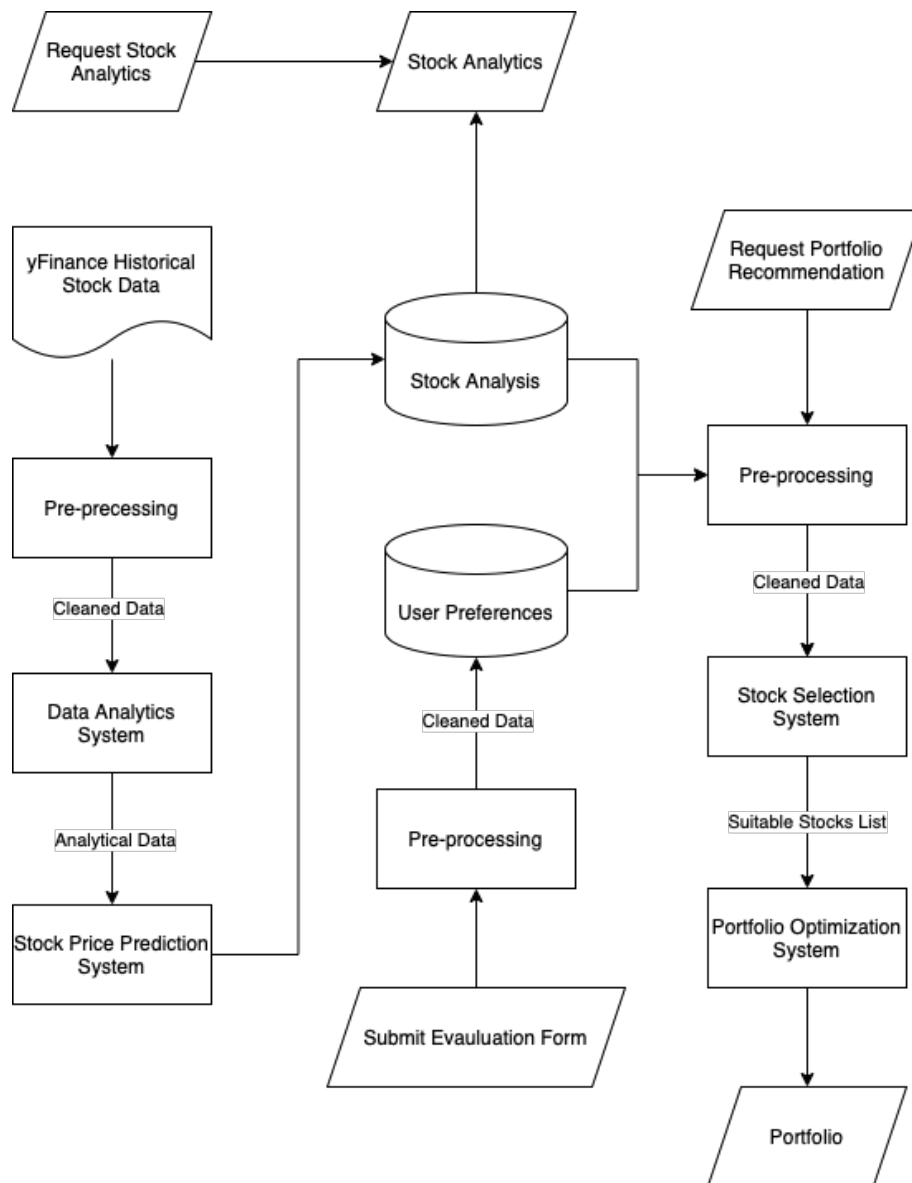


Figure 2. Flow Diagram of the proposed system

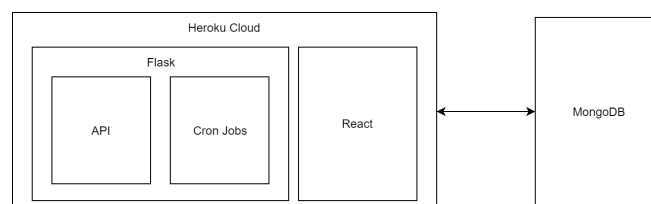


Figure 3. Cloud Deployment

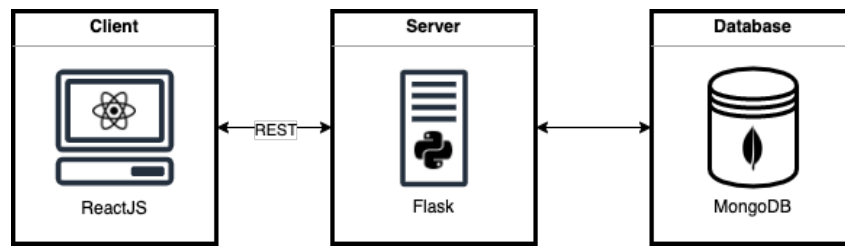


Figure 4. Client-Server architecture

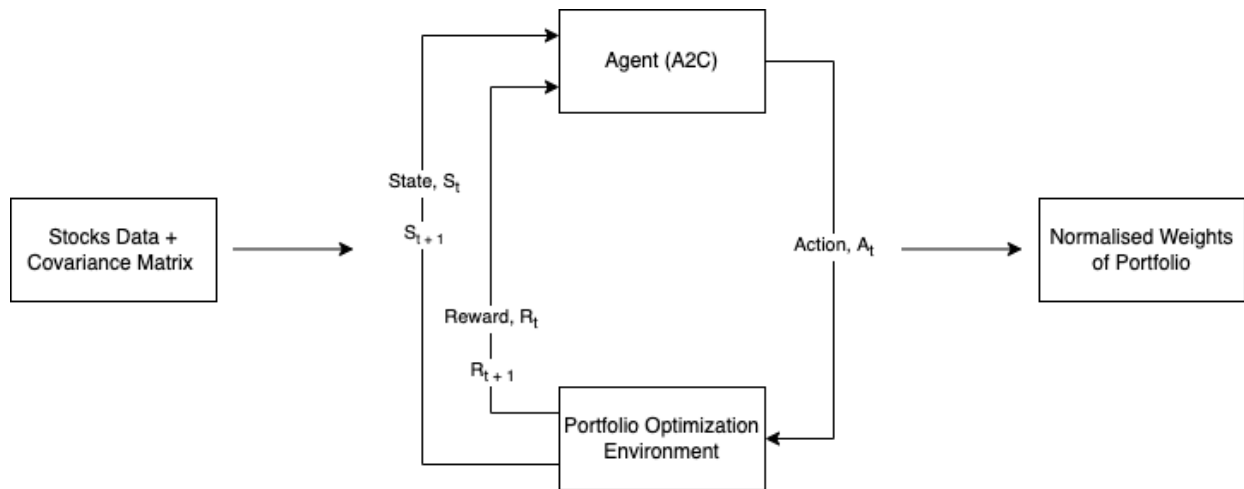


Figure 5. Flow of the Reinforcement learning model

environment and trying to learn by maximizing the reward. The portfolio optimization environment starts with an observation state, which is a dictionary object comprising two 2d arrays, one of which is the covariance matrix, and the other is a list of technical indicators. The agent then performs an action on the observation state, and the action weights are supplied to the step function in a 1d array with values ranging from -1 to 1, as shown above. The function begins by determining whether this is the final iteration; if so, the function returns the final state. Otherwise, the main process begins; the first step is to locate all tickers that need to be bought and sold; values greater than 1 must be bought, while values less than 0 must be sold. The weights to be purchased are then processed by a softmax function, resulting in a sum of the weights equal to 1. Then, using the sell method, all previous stocks are sold, and the money is added to the money_available variable, followed by the buy method, which is used to purchase the selected stocks. The buy method algorithm is quite unique; first, we split the money among the stocks based on their weight; for example, if the money available is \$10,000 and the weight is 0.3, the money available to buy the stock at index 0 is \$3000. If a stock is too expensive, for example, a share of index 0 costs \$3500, the weight is set to 0, and the softmax function is used again. The allocation for each stock is then completed by determining how many shares can be purchased with the available money. In rare cases, money is left over after the previous technique and cannot be distributed to any stock. In this scenario, we sort the weights from largest to smallest and use an infinite loop to buy each stock until the money available is less than the lowest stock or there are no more stocks to buy. The allocation result is then used to determine today's and tomorrow's portfolio values using the current iteration price and the next iteration price, respectively. After that, the return

is computed, and the reward is set to the portfolio price for the next day since this could indicate the agent's performance during this iteration. Finally, the next observation is loaded, and the process is repeated until completed.

Furthermore, the stock price prediction system consists of LSTM. The difference between the start and end stock data is the length of the dataset. Ten years of historical data for each stock are used to train a model (i.e., each stock has its own model). The dataset is divided into training data and test data, with training data accounting for 80% of the total. In the dataset, the model will look back 60 days. Figures 6 and 7 show the flow diagrams of the LSTM model for training and prediction.

4. Experimental results

4.1. User evaluation

The objective of the user evaluation study focuses on non-functional requirements such as processing speed, usefulness, and usability, all of which are related to the system's performance. User evaluation is also essential to understand how users feel about our web app after interacting with our prototype. A summative evaluation method is used. The user survey is distributed at the end of the development process, and the results ensure that the work meets the needs of various consumers.

According to the results of our user evaluation, as shown in Figure 8, the majority of users agree that the portfolio generation procedure is simple. Furthermore, as shown in Figure 9, more than half of the users feel that the portfolio generated by our system matches their desired risk and reward potential. As a result, the majority of users decided to follow the system's advice.

As shown in Figure 10, 85% of users think the system can achieve good performance in portfolio generation. In addition, users thought the system had clear instructions (67%) and was easy to use (82%). Based on the user evaluation, 66% of users believed that the recommendations generated by the system matched their preferences, as seen in Figure 11. Lastly, over three-quarters of users thought that we had a good UI/UX design.

4.2. Performance evaluation study

The aim of this study is to determine whether the trained model can generate a profitable portfolio. As the performances of various buying strategies (e.g., buy-and-hold and daily rebalance) differ significantly, and several factors affect those strategies (such as investment timeframes and associated fees with each trade), our main objective is to offer users the most efficient buying strategy that maximizes profits.

The cumulative return utilizing our methods is greater than the S&P 500, as seen in Figure 12. By backtesting with the previous year's stock data, it can be shown that if we trade for a year, our portfolio can gain 20 to 25 percent utilizing a daily rebalancing strategy. We can also gain up to 20% by using the buy-and-hold strategy. The S&P 500, on the other hand, only produces a 10% profit every year. We've profited up to twice as much as the S&P 500. Despite the fact that the daily rebalance strategy offers a better return, some investors find it too difficult to trade on a daily basis, as they must rebalance every day to receive the most recent portfolio. As a result, the system opts for the buy-and-hold strategy. Finally, as shown in Figure 13, the average user of our service makes slightly more profit from our portfolio than the S&P 500, ranging from 2.5 to 5%. Overall, the portfolio recommendation system's

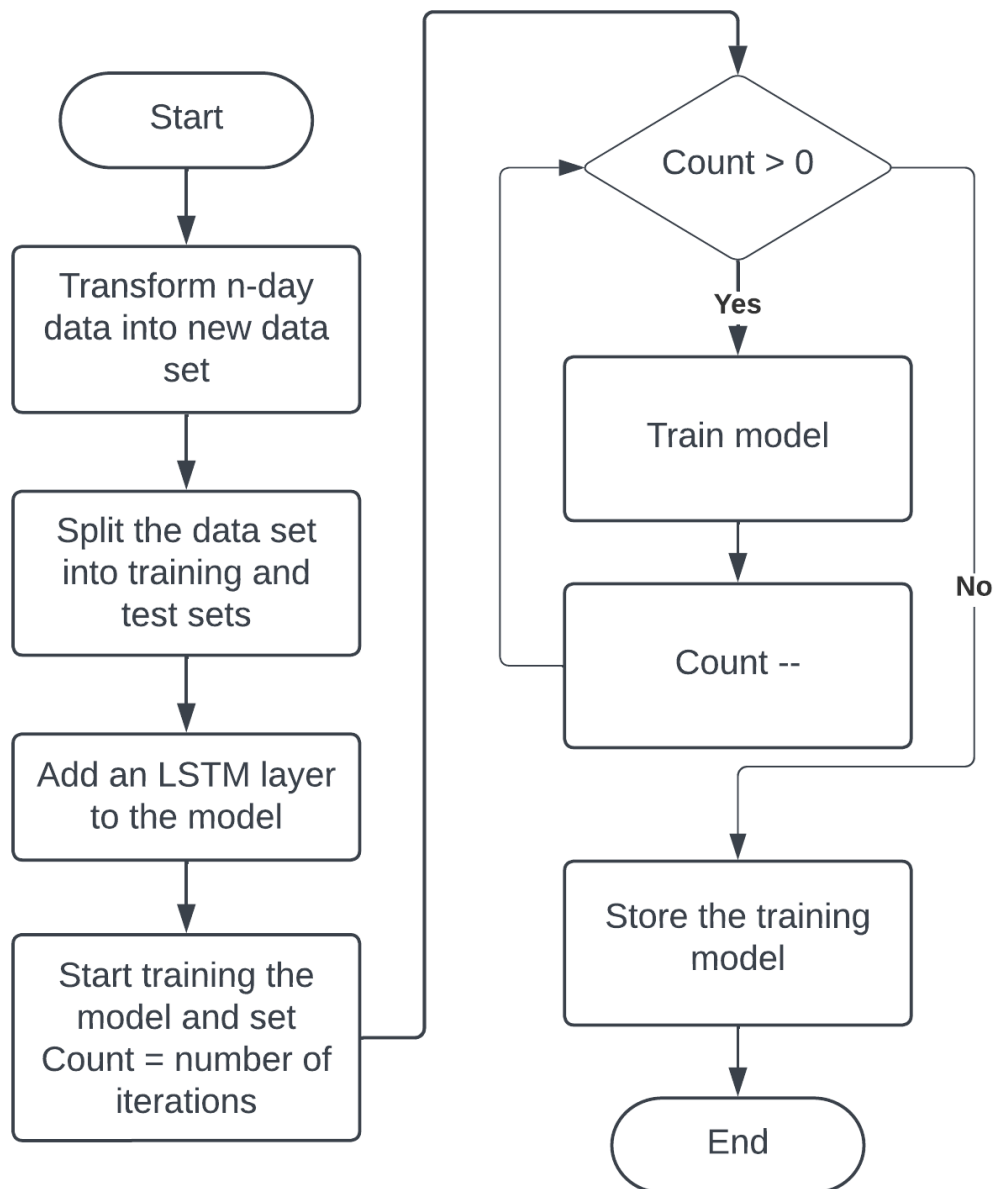


Figure 6. LSTM model training flow diagram

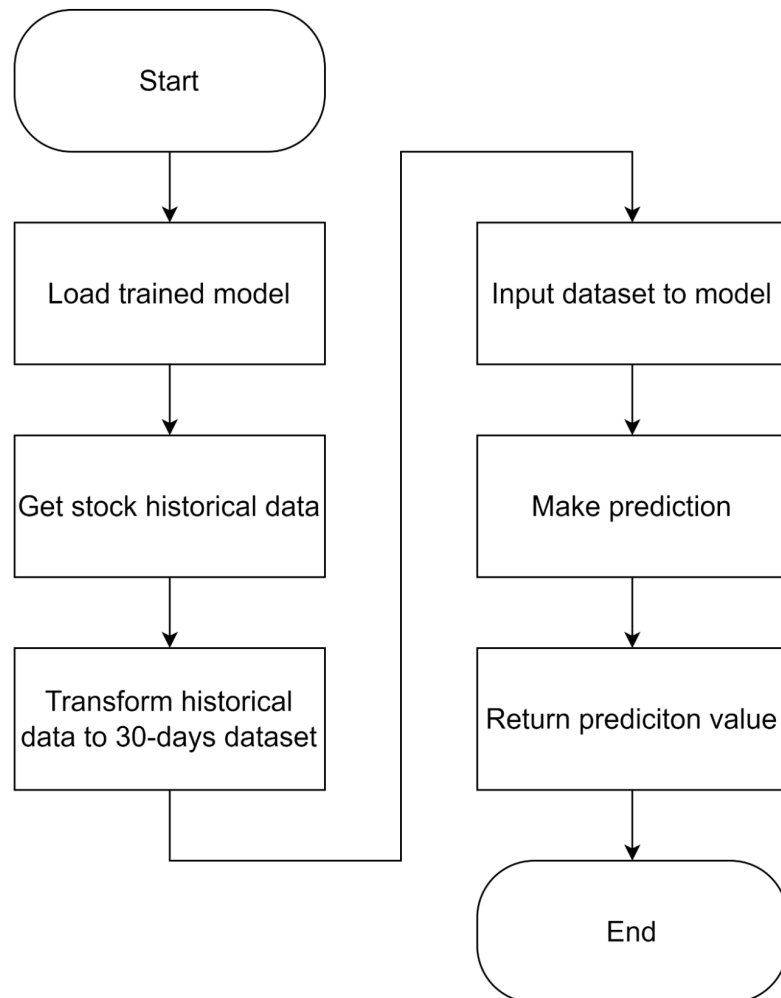


Figure 7. LSTM model prediction flow diagram

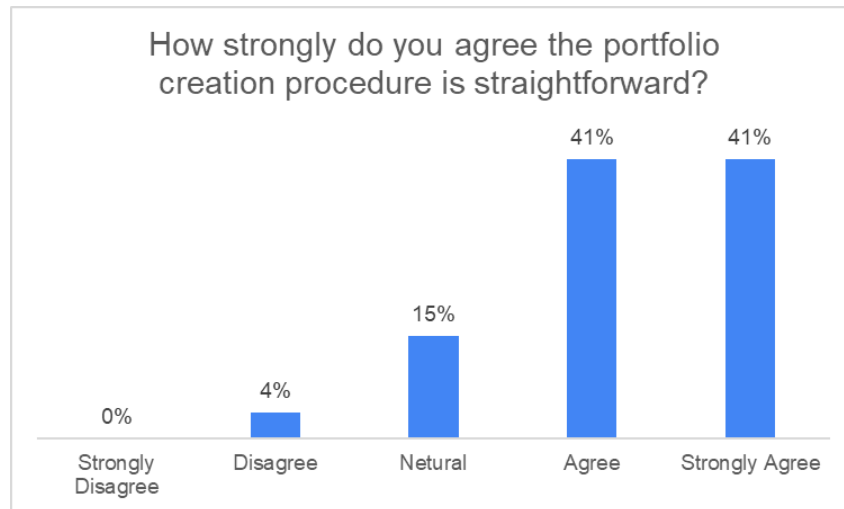


Figure 8. User evaluation question 1

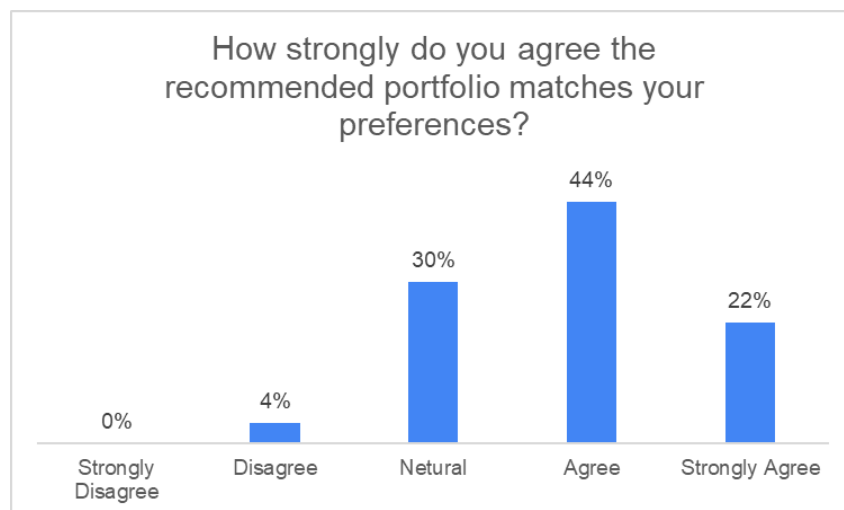


Figure 9. User evaluation question 2

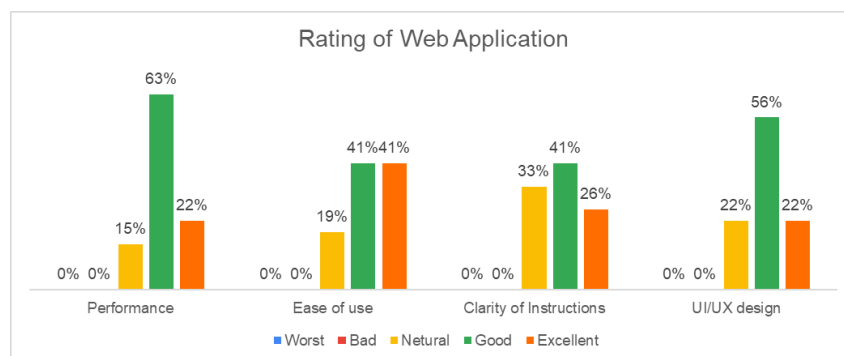


Figure 10. User evaluation question 3

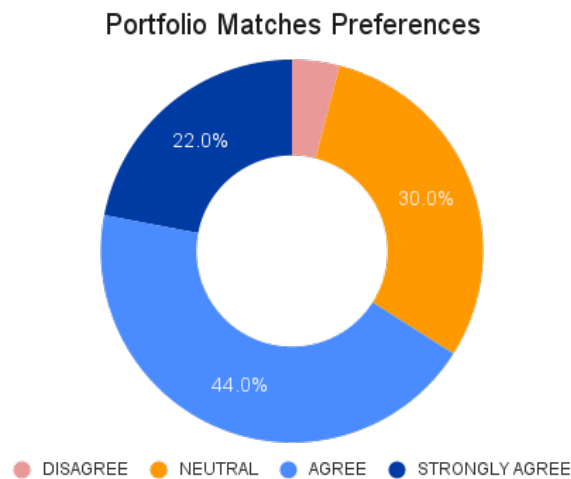


Figure 11. User evaluation - performance matches preferences result

performance is satisfactory, and it can outperform the S&P 500.

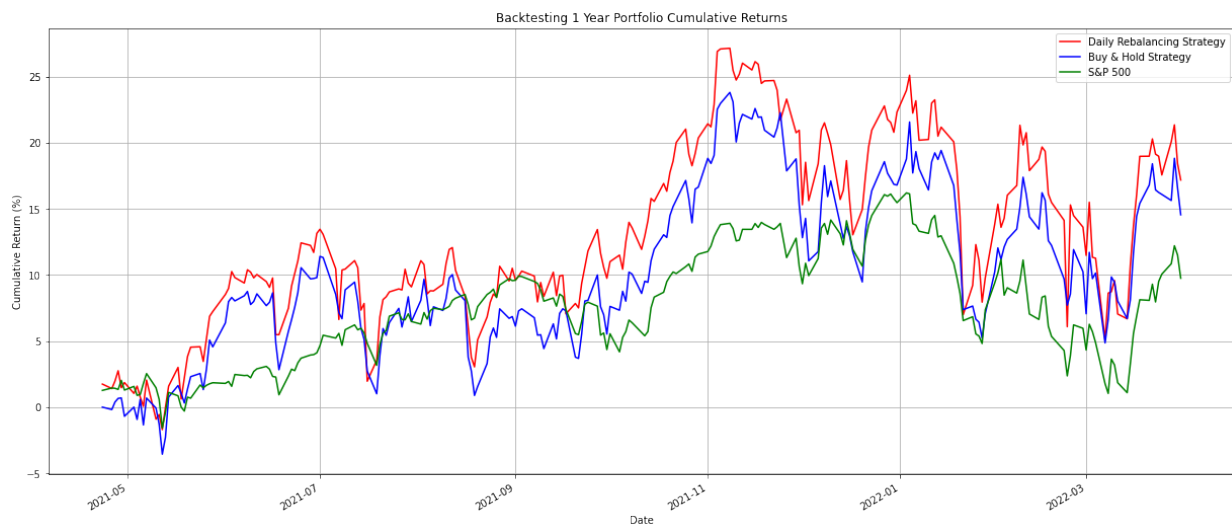


Figure 12. Comparing cumulative returns with different strategies

5. Conclusion

This paper presents a machine learning-based portfolio recommendation system that not only provides users with a profitable portfolio but also helps them gain financial knowledge and make better financial decisions in the long term. Beginners and users with a low understanding of financial knowledge can learn more about the stock market by using the proposed system. The experimental results demonstrate that the system performs well on the system-generated portfolio that matches users' preferences, as well as different buying strategies. Future work could focus on improving the performance of the portfolio optimization system.

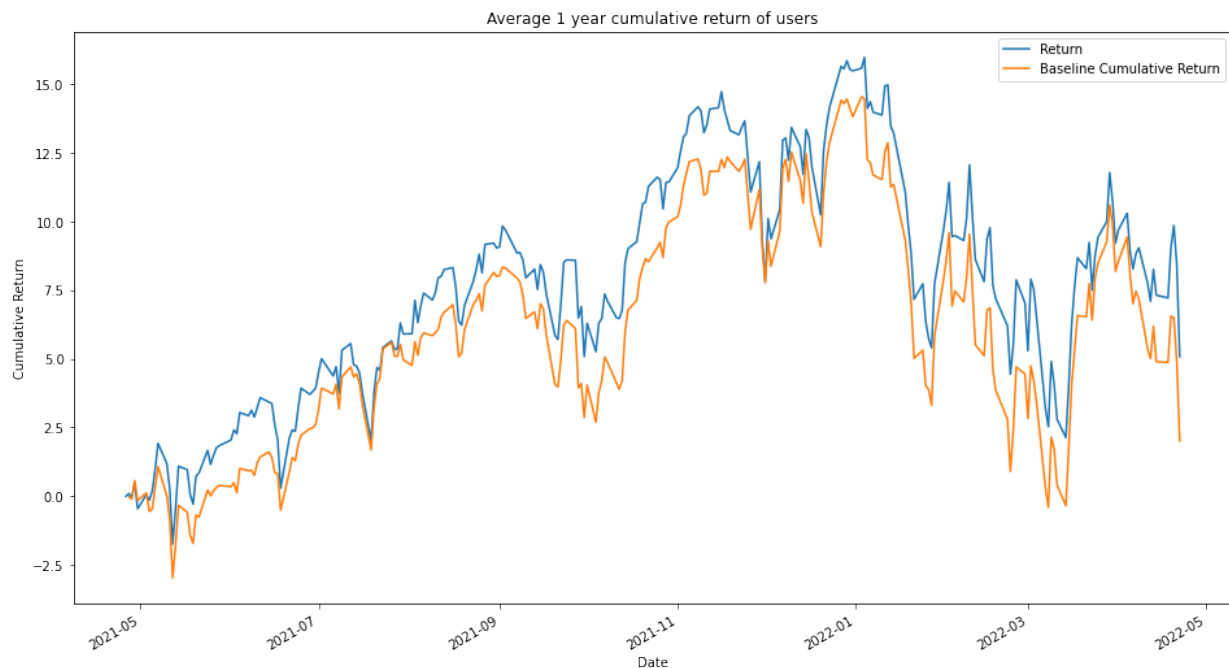


Figure 13. Comparing average returns of users with the S&P 500

Conflict of interest

All authors declare no conflicts of interest in this paper.

References

- Du B, Zhou Q, Guo J, et al. (2021) Deep learning with long short-term memory neural networks combining wavelet transform and principal component analysis for daily urban water demand forecasting. *Expert Syst Appl* 171: 114571. <https://doi.org/10.1016/j.eswa.2021.114571>
- Finance Y (2020) Yahoo Finance. Retrieved from [finance.yahoo.com](https://finance.yahoo.com/recent-quotes). Available from: <https://finance.yahoo.com/recent-quotes>.
- Gulli A, Pal S (2017) *Deep learning with Keras*. Packt Publishing Ltd.
- Lai ZR, Yang PY, Fang L, et al. (2020) Reweighted price relative tracking system for automatic portfolio optimization. *IEEE T Syst* 50: 4349-4361. <https://doi.org/10.1109/TSMC.2018.2852651>
- Lee J, Sohn SY (2021) Recommendation system for technology convergence opportunities based on self-supervised representation learning. *Scientometrics* 126: 1–25. <https://doi.org/10.1007/s11192-020-03731-y>
- Li X, Yu C (2017) An investment portfolio recommendation system for individual e-commerce users. *DEStech Transactions on Engineering and Technology Research*, 580-585.
- Leung MF, Wang J (2021) Minimax and biobjective portfolio selection based on collabo-

-
- rative neurodynamic optimization. *IEEE Trans Neural Netw Learn Syst* 32: 2825–2836. <https://doi.org/10.1109/TNNLS.2019.2957105>
- Leung MF, Wang J, Che H (2021) Another Two-Timescale Duplex Neurodynamic Approach to Portfolio Selection. In *2021 11th International Conference on Intelligent Control and Information Processing (ICICIP)* 2021: 387-391. <https://doi.org/10.1109/ICICIP53388.2021.9642204>
- Leung MF, Wang J (2022) Cardinality-constrained portfolio selection based on collaborative neurodynamic optimization. *Neural Networks* 145: 68-79. <https://doi.org/10.1016/j.neunet.2021.10.007>
- McKinney W (2012) Python for data analysis: Data wrangling with Pandas, NumPy, and IPython. O'Reilly Media, Inc.”.
- Oluwatosin, H. S. (2014). Client-server model. *IOSRJ Comput. Eng*, 16(1), 2278-8727.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *the Journal of machine Learning research*, 12, 2825-2830.
- Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M., & Dormann, N. (2021). Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*.
- Ren K, Malik A (2019) Investment recommendation system for low-liquidity online peer to peer lending (P2PL) marketplaces. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. 510–518. <https://doi.org/10.1145/3289600.3290959>
- Sen J, Mehtab S (2020) A Time Series Analysis-Based Stock Price Prediction Using Machine Learning and Deep Learning Models. *Int J Bus Forecas Market Intell* 6: 272. <https://doi.org/10.1504/IJBFMI.2020.115691>
- Shukla N, Fricklas K (2018) *Machine learning with TensorFlow*. Greenwich: Manning.
- Wu JMT, Li Z, Herencsar N, et al. (2021) A graph-based CNN-LSTM stock price prediction algorithm with leading indicators. *Multimedia Syst* 2021: 1-20. <https://doi.org/10.1007/s00530-021-00758-w>
- Yuen MC, Ng SC, Leung MF, et al. (2021) A metaheuristic-based framework for index tracking with practical constraints. *Complex Intell Syst* 8: 4571-4586. <https://doi.org/10.1007/s40747-021-00605-5>