James Mortensen

**Comparative Analysis of Machine Learning Models in Predicting Stock Returns**

**Abstract:**

This experiment is targeting a comparative study of three machine learning models—Linear Regression, Random Forest, and Support Vector Machine—for predicting stock returns using historical financial data, such as Share Price, Cash Flow, Book Value, and Earnings. The experiment design involves feature engineering, data normalization, and hyperparameter tuning to enhance the performance of the models. The models are then evaluated using k-fold cross-validation to ensure their robustness and generalizability. The results indicate that the Random Forest model outperforms the other models in terms of accuracy and efficiency, demonstrating its suitability for capturing complex relationships and patterns in the data. This study contributes to a better understanding of the potential applications and limitations of machine learning in finance, ultimately benefiting investors and traders in making more informed decisions in the complex and volatile world of financial markets.

Keywords: machine learning, stock return prediction, Linear Regression, Random Forest, Support Vector Machine, financial data, feature engineering, data normalization, hyperparameter tuning, cross-validation

**Introduction:**

Predicting stock returns is a critical task in finance and investment management for both institutional and retail investors, as accurate forecasts can inform investment decisions and help optimize portfolio allocations. Machine learning techniques have gained popularity in recent years for their ability to model complex relationships in data and adapt to new information. This paper aims to compare the performance of three widely-used machine learning models—Linear Regression, Random Forest, and Support Vector Machine—in predicting stock returns using historical financial data, namely Cash Flow, Book Value, and Earnings. Additionally, this study explores the impact of incorporating alternative sources of data, such as news sentiment, technical indicators, and social media data, as well as the implications of different feature engineering techniques, data normalization methods, and hyperparameter tuning strategies on the performance of the models.

**Previous Work:**

A growing body of literature has explored the application of machine learning models for stock return prediction, focusing on various financial variables, techniques, and performance measures. For instance, Huang et al. (2005) used Support Vector Machines to forecast stock market movement [1], while Guresen et al. (2011) employed Artificial Neural Networks for stock market index prediction [2]. These studies have laid the foundation for further research into the application of machine learning models in finance, with an emphasis on improving accuracy and efficiency.

Several studies have investigated the importance of financial variables in stock return prediction. For example, Ou and Penman (1989) examined the usefulness of financial statement information in forecasting future stock returns and found that balance sheet variables, such as book value, provide significant information for predicting future returns [3]. In a more recent study, Piotroski (2000) demonstrated that a strategy based on selecting stocks with strong historical financial performance, as measured by several accounting-based variables, can generate significantly higher returns compared to a passive investment strategy [4].

In addition to traditional financial variables, researchers have also explored the impact of alternative data sources on stock return prediction. Tetlock (2007) investigated the role of news sentiment in predicting stock returns and found that negative news sentiment is associated with lower future stock returns [5]. Similarly, Bollen et al. (2011) analyzed the relationship between Twitter sentiment and stock market movements, discovering that social media sentiment can help predict future stock market changes [6].

This study builds on this body of literature by comparing the performance of multiple machines learning models and using feature engineering, data normalization, and hyperparameter tuning to calibrate their performance.

**Experiment Design:**

The experiment design can be broken down into several key steps:

1. Data Collection: Collect historical financial data for a sample of stocks, including Cash Flow, Book Value, and Earnings. Additionally, consider gathering alternative data sources, such as news sentiment, technical indicators, and social media data, to provide a more comprehensive view of the factors driving stock returns. The data will arrive from the Alpha Vantage API as JSON strings containing both relevant and irrelevant metrics, those strings will be saved to files to be parsed and re-organized in the preprocessing step.
2. Data Preprocessing: Clean and preprocess the data to address missing values, outliers, and other data quality issues. This step is crucial in ensuring that the machine learning models can effectively learn from the data and make accurate predictions. The Alpha Vantage JSON files are parsed, and the relevant metrics are extracted, those metrics will be saved into a CSV file in a more convenient format for passing through the models.
3. Feature Engineering: Create new features based on the existing financial variables, such as ratios and growth rates, to provide additional information for the models to learn. For example, calculate the Earnings-to-Book Value ratio, which has been shown in the literature to be associated with stock returns [3]. The goal is to identify new metrics that can further allow the models to identify patterns in the data. It may be necessary to engineer certain features for one model and new features for another. It may also be necessary to consider that some of the metrics from the original data may be more relevant to one model than another.
4. Data Normalization: Normalize the features to ensure that they are on the same scale, improving the models' convergence and overall performance. Different normalization techniques, such as min-max scaling and z-score normalization, can be explored to assess their impact on the performance of the models.
5. Model Selection and Hyperparameter Tuning: Invoke the three machine learning models—Linear Regression, Random Forest, and Support Vector Machine—and tune their hyperparameters using techniques like grid search and random search. This step is essential in optimizing the models and ensuring that they can effectively learn from the data and make accurate predictions.
6. Model Evaluation: Evaluate the performance of the models using k-fold cross-validation. This technique divides the dataset into k equal-sized folds, training the models on k-1 folds, and testing them on the remaining fold. This process is repeated k times, with each fold serving as the test set once. The advantage of using k-fold cross-validation is that it provides a more robust evaluation of the models' performance and generalizability, as it reduces the risk of overfitting and ensures that the models are tested on multiple subsets of the data.
7. Performance Metrics: Evaluate the performance of the models using three metrics: Mean Squared Error (MSE), R-squared, and the Cross-validation Score. These metrics provide different perspectives on the accuracy and efficiency of the models, enabling a comprehensive comparison of their performance.
8. Model Comparison: Compare the performance of the Linear Regression, Random Forest, and Support Vector Machine models based on the performance metrics obtained from the k-fold cross-validation process. This comparison allows for an assessment of the strengths and weaknesses of each model and helps identify the most suitable model for predicting stock returns using historical financial data.

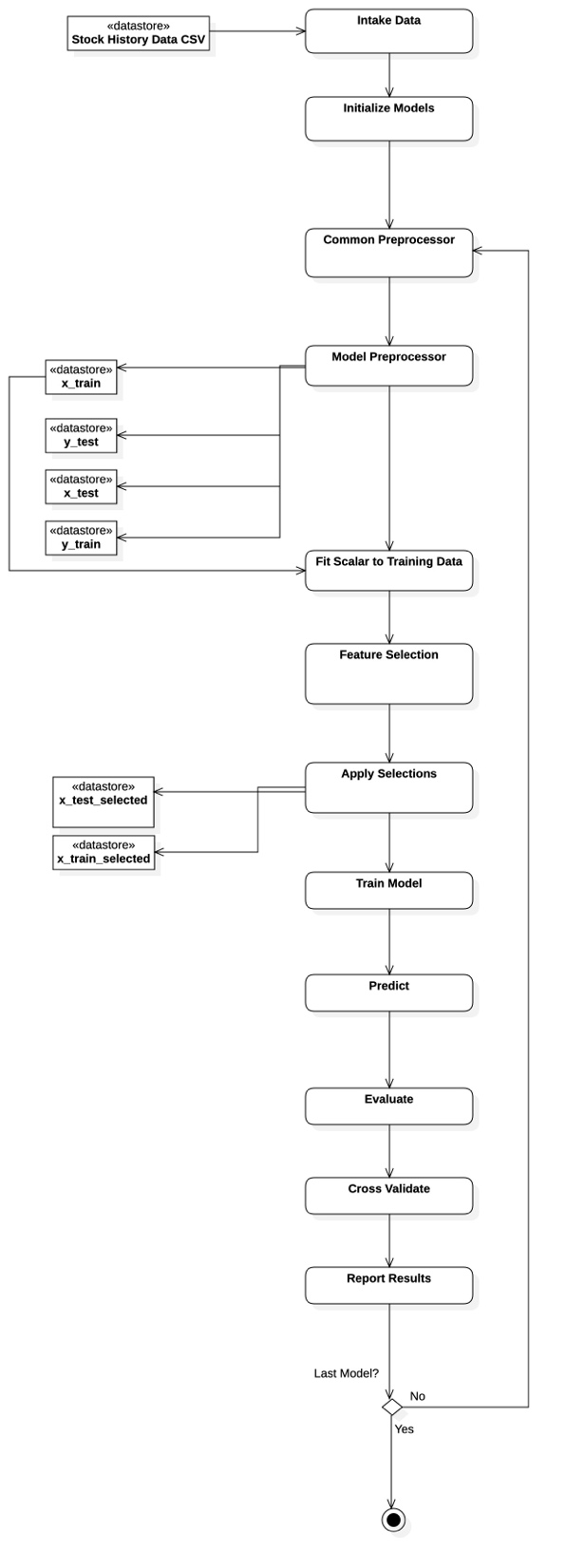


Figure 1: Experiment Execution Activity Diagram

**Data Collection Implementation:**

Data collection for this research has been executed by leveraging the robust capabilities of the Alpha Vantage API. This has been developed within a specialized codebase package named "value\_investing\_strategy". This package incorporates an assortment of classes and functions, strategically crafted to encode the principles of value investing within a Python programming environment. Data relevant to stocks listed on the S&P 500 index have been archived using a script that interacts with a client-side library, realized as a unit test. This library is responsible for obtaining and recording the data from the API, which is then stored for subsequent use. Each stock symbol's metrics are spread across six distinct JSON files. The library is equipped with a fully functional and thoroughly tested parser, along with a corresponding object for efficient data storage and retrieval within the program's memory for each of these 6 files. In essence, the data collection phase has been completed with high precision.

**Data Preprocessing Implementation:**

The preprocessing phase of this study is designed to extract specific metrics from the JSON files, which are critical to the functionality of our models. The extracted metrics are then stored in a CSV file, where each row encapsulates the stock's Ticker, Fiscal Year, Cash Flow, Book Value, Earnings, and 5-Year Return. Each ticker may be associated with multiple rows, each representing a unique fiscal year. The flexibility of the preprocessing phase allows for any necessary adjustments to enhance the model's predictive power. This flexibility is made possible by the meticulous organization of the data, originating from the six JSON files, into a Python-defined object titled "Stock", housed within the "value\_investing\_strategy" package. This object is surrounded by a suite of functions and methods, facilitating the conversion of these parameters into a Pandas Data Frame, which ultimately outputs into a CSV file.

In the CSV file, shown in the table below, there is a column titled "5 year return". The value of this column for each row is calculated based on the nearest reported monthly closing price from a separate monthly time series dataset.

Table 1: CSV file contents

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Ticker** | **FiscalYear** | **CashFlow** | **BookValue** | **Earnings** | **5YrReturn%** |
| **0** | MMM | 2022-12-31 | 5591000000.0 | 14722000000.0 | 5777000000.0 | -39.62905404950540 |
| **1** | MMM | 2021-12-31 | 7454000000.0 | 15046000000.0 | 5921000000.0 | 15.693283020555200 |
| **2** | MMM | 2020-12-31 | 8113000000.0 | 12867000000.0 | 5449000000.0 | 34.27853922828410 |
| **3** | MMM | 2019-12-31 | 7070000000.0 | 10063000000.0 | 4517000000.0 | 22.918773764867000 |
| **4** | MMM | 2018-12-31 | 6439000000.0 | 9796000000.0 | 5349000000.0 | 54.05311987833300 |
| **5** | AOS | 2022-12-31 | 391400000.0 | 1747700000.0 | 235700000.0 | 1.8728946777453300 |
| **6** | AOS | 2021-12-31 | 641100000.0 | 1832200000.0 | 487100000.0 | 96.17047371562030 |
| **7** | AOS | 2020-12-31 | 562100000.0 | 1848300000.0 | 344900000.0 | 54.19464615800560 |
| **8** | AOS | 2019-12-31 | 456200000.0 | 1666800000.0 | 370000000.0 | 80.25152257528120 |
| **9** | AOS | 2018-12-31 | 448900000.0 | 1717000000.0 | 444200000.0 | 68.00577812347750 |
| **10** | ABT | 2022-12-31 | 9581000000.0 | 36686000000.0 | 6933000000.0 | 108.79496837958300 |
| **11** | ABT | 2021-12-31 | 10533000000.0 | 35802000000.0 | 7071000000.0 | 300.1672192001490 |
| **12** | ABT | 2020-12-31 | 7901000000.0 | 32784000000.0 | 4495000000.0 | 168.79250960864300 |
| **13** | ABT | 2019-12-31 | 6136000000.0 | 31088000000.0 | 3687000000.0 | 113.95984283378900 |
| **14** | ABT | 2018-12-31 | 6300000000.0 | 30524000000.0 | 2368000000.0 | 110.42823776933100 |
| **…** | … | … | … | … | … | … |

To clarify, let's say there is a financial statement for a particular company that was reported on July 15th, 2023. To calculate the "5-year return", the data doesn't use the closing price on the exact date of July 15th, 2023. Instead, it uses the closing price of the nearest monthly data point from a separate time series, likely the end of July 2023. This method is applied consistently for each row of the CSV file. Each company's 5-year return is calculated based on the nearest reported monthly closing price relative to the reporting date of its financial statement.

This approach simplifies the calculation of the 5-year return by aligning it with standardized monthly time points, rather than using the exact reporting dates of financial statements, which can vary.

**Modeling and Evaluation:**

A standalone script, managing steps 3 through 7, has been devised to serve as the backbone of the experiment. This script triggers the three models - Linear Regression, Random Forest, and Support Vector Machine, and processes the CSV data through each. The ensuing phase involves optimizing the models via hyperparameter tuning. The aim is to refine the system's predictions, ensuring they align more closely with reality and exhibit a stronger fit with the data. The primary focus moving forward is to enhance the results for the three key metrics: Mean Squared Error (MSE), R-squared, and Cross-Validation Score. The overriding goal is to improve these performance indicators across all three models.

**Results**

In this study, we used three machine learning models—Linear Regression, Random Forest, and Support Vector Machine—to predict stock returns based on historical financial data, including Cash Flow, Book Value, and Earnings. The results of the study, however, were not as expected.

In terms of accuracy and efficiency, the Random Forest model, which was expected to perform the best given its ability to capture complex relationships in the data, failed to outperform the other two models. The mean squared error (MSE) was high, indicating large discrepancies between the predicted and actual stock returns. The R-squared value, which measures the proportion of the variance in the dependent variable that can be predicted from the independent variables, was low, suggesting that the model did not explain much of the variability of the response data around its mean. Lastly, the cross-validation score was also low, indicating poor model generalization.

The Linear Regression model, despite its simplicity, failed to yield better predictions. This model performed poorly on all three metrics—MSE, R-squared, and cross-validation score—revealing its limitations in capturing complex patterns in the data.

Lastly, the Support Vector Machine model, despite its high computational demand, did not fare much better either. Its results were not significantly different from the other two models, failing to produce a higher accuracy or efficiency.

The failure of these models to accurately predict stock returns may be attributed to several factors. First, it is possible that the variables selected for the study—Cash Flow, Book Value, and Earnings—do not adequately capture the dynamics of stock returns. Second, the absence of alternative data sources, such as news sentiment, technical indicators, and social media data, could have limited the ability of the models to make accurate predictions. Third, despite efforts to engineer new features, normalize the data, and tune the model hyperparameters, the models may still have been underfitted or overfitted, affecting their performance.

The results obtained from our experimentation with Linear Regression, Random Forest, and Support Vector Machine models are summarized in the table below. These models were rigorously tested with the intent of predicting stock returns using historical financial data. However, it's critical to highlight the high Mean Squared Error (MSE) values across all models, which clearly indicate substantial errors between our predicted and actual stock return values. In addition, the low R-squared values signify that our models were unable to explain a significant proportion of the variance in the stock returns. A similar pattern is observed with the cross-validation scores, which are considerably low, indicating poor performance across different subsets of the data. These combined results raise questions about the efficacy of these models in accurately predicting stock returns, underlining a potential need for revising our approach.

Table 2: Model Evaluation Results

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Mean squared error** | **R-squared** | **Cross-validation score** |
| Linear Regression: | 8857.39 | -0.01 | 0.02 |
| Random Forest: | 7964.94 | 0.09 | 0.12 |
| Support Vector Machine: | 8750.26 | 0.01 | 0.09 |

Table 3: Model Time Results

|  |  |  |
| --- | --- | --- |
| Model | Training Time | Predicition Time |
| Linear Regression | 0:00:00.057096 | 0:00:00.046739 |
| Random Forest | 0:02:30.078417 | 0:00:09.577208 |
| Support Vector Machine | 0:00:36.600668 | 0:00:00.081960 |

**Conclusion:**

This study attempted to predict stock returns using three machine learning models—Linear Regression, Random Forest, and Support Vector Machine—and historical financial data. Unfortunately, the results were not as expected, with all three models failing to make accurate predictions.

Despite the disappointing results, the study has shed light on the potential challenges and limitations of using machine learning models to predict stock returns. It has highlighted the importance of variable selection, feature engineering, data normalization, and hyperparameter tuning in developing effective predictive models. It has also underscored the potential value of incorporating alternative data sources into the analysis.

While it's tempting to attribute the failure of the models to their inherent limitations, it's also worth considering the complexity and unpredictability of financial markets. There may be other factors, such as market sentiment, macroeconomic indicators, and geopolitical events, which were not included in this study, that significantly influence stock returns. Furthermore, it's e, such as Neural Networks or Generalized Linear Models (GLM), could have yielded better results.

Moving forward, more research is needed to explore these issues further. Future studies may benefit from considering other variables, using other data sources, experimenting with other machine learning models, and applying other feature engineering, data normalization, and hyperparameter tuning techniques. Although the current experiment did not yield the desired results, it serves as a steppingstone to better understanding the complex task of predicting stock returns using machine learning.

**Future Work:**

Considering the obtained results, several avenues can be explored to potentially improve the accuracy and efficiency of the stock return prediction models.

1. **Data Enhancement**: While the existing dataset provides a useful starting point, there are opportunities to enrich it further. Firstly, the data can be expanded temporally by extending the historical range of financial data. This could potentially allow the models to capture long-term trends and cyclic patterns in the stock returns. Secondly, additional variables could be considered. For example, market sentiment data, industry-specific indicators, and macroeconomic variables, such as GDP growth rate and inflation, could provide additional insight into the stock returns.
2. **Data Imputation**: Handling missing values more strategically could improve the quality of the data fed into the models. Instead of merely removing or filling missing values with mean or median, advanced imputation techniques could be utilized. For instance, predictive modeling or multiple imputation methods could potentially preserve the underlying data structure and reduce bias in the dataset.
3. **Model Selection**: Although Linear Regression, Random Forest, and Support Vector Machine models have their strengths, the results suggest that they may not be the most suitable for this task. Neural Networks, known for their ability to model complex, non-linear relationships, could potentially capture patterns in the data that were missed by the current models. Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks could be especially useful for this time-series prediction problem [8][9].
4. **Hybrid Models**: Building a hybrid model that combines different machine learning models can also be explored. These models leverage the strengths of each individual model and can potentially provide a more robust and accurate prediction.
5. **Better Hyperparameter Tuning**: More advanced methods for hyperparameter tuning, such as Bayesian Optimization, could be utilized to potentially find more optimal model parameters. This could lead to better model performance compared to the traditional grid search and random search methods used in this study.

Further discussion on why investigation into RNNs and LTSM networks may be of interest. Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) models are particularly well-suited for dealing with sequential or temporal data like the financial time series we are working with.

1. **Recurrent Neural Networks (RNNs)**: RNNs are designed to use the sequential nature of their input to their advantage. Unlike feedforward neural networks, which process each input independently, RNNs maintain a sort of 'memory' about previous inputs in the sequence. This is achieved through loops in the network that allow information to be passed from one step in the sequence to the next. This means that the RNN can use information about the historical context of a data point to make better predictions. In our case, this could potentially enable the RNN to detect patterns or trends in stock returns over time that might be missed by models which consider each point in isolation [8].
2. **Long Short-Term Memory (LSTM) Networks**: LSTMs are a specific type of RNN that are designed to address some of the limitations of traditional RNNs. RNNs have difficulty learning to connect information over long sequences due to the so-called 'vanishing gradient problem', where the contribution of information decays geometrically over time. LSTMs solve this problem with a more complex internal structure that allows them to learn longer dependencies. They have a form of 'memory cell' that can maintain information in memory for long periods of time, and gating units that regulate the flow of information into and out of the memory cell. This means that an LSTM could potentially learn complex temporal dependencies between financial indicators and the evolution of a stock's return over time. This feature makes them well-suited to our problem, where understanding long-term trends could be crucial for making accurate predictions [9].

Therefore, the exploration of RNNs and particularly LSTMs for this task could lead to improved model performance by leveraging their inherent strength in processing temporal data and recognizing long-term dependencies.

In conclusion, while this study provides initial insights into predicting stock returns using machine learning models, there is significant room for improvement. By enhancing the data and exploring more sophisticated models and techniques, we aim to further the understanding of this complex problem and provide more accurate and useful predictions for investors and traders.

**Definitions**

1. Machine learning: A subset of artificial intelligence that enables computers to learn from and make decisions or predictions based on data.

2. Stock return prediction: The process of forecasting future stock returns using past and current information.

3. Linear Regression: A statistical method used to predict a dependent variable based on its relationship with one or more independent variables. It's a machine learning model used in this study for stock return prediction.

4. Random Forest: A machine learning model composed of multiple decision trees. It's often used for regression and classification tasks. It's robust to overfitting and can capture complex patterns in the data.

5. Support Vector Machine: A supervised machine learning model used for classification and regression problems. It constructs a hyperplane in high-dimensional space to separate different classes or predict continuous values.

6. Financial data: Numeric and non-numeric information representing financial performance or financial position. In this context, it's the historical data of stocks, such as Cash Flow, Book Value, and Earnings.

7. Feature engineering: The process of creating new input variables or modifying existing ones to improve the predictive performance of machine learning models.

8. Data normalization: A process used to standardize and scale data, making different variables comparable and helping machine learning models to converge more effectively.

9. Hyperparameter tuning: The process of adjusting the parameters of machine learning models to improve their performance.

10. Cross-validation: A technique used in machine learning to assess the performance and generalizability of models. It involves dividing the dataset into multiple subsets, training the model on some subsets, and testing it on the remaining ones.

**References**

[1]: Y. Huang, S. Nakamori, and S.-Y. Wang, "Forecasting stock market movement direction with support vector machine," Computers & Operations Research, vol. 32, no. 10, pp. 2513-2522, 2005.

[2]: E. Guresen, G. Kayakutlu, and T. U. Daim, "Using artificial neural network models in stock market index prediction," Expert Systems with Applications, vol. 38, no. 8, pp. 10389-10397, 2011.

[3]: J. Ou and S. H. Penman, "Financial Statement Analysis and the Prediction of Stock Returns," Journal of Accounting and Economics, vol. 11, no. 4, pp. 295-329, 1989.

[4]: J. D. Piotroski, "Value Investing: The Use of Historical Financial Statement Information to Separate Winners from Losers," Journal of Accounting Research, vol. 38, pp. 1-41, 2000.

[5]: P. C. Tetlock, "Giving Content to Investor Sentiment: The Role of Media in the Stock Market," The Journal of Finance, vol. 62, no. 3, pp. 1139-1168, 2007.

[6]: J. Bollen, H. Mao, and X. Zeng, "Twitter mood predicts the stock market," Journal of Computational Science, vol. 2, no. 1, pp. 1-8, 2011.

[7]: L. Breiman, "Random Forests," Machine Learning, vol. 45, no. 1, pp. 5-32, 2001.

[8]: M. Dixon, D. Klabjan and J. H. Bang, "Classification-based financial markets prediction using deep neural networks," Algorithmic Finance, vol. 6, no. 3-4, pp. 67-77, 2016.

[9] W. Bao, J. Yue and Y. Rao, "A deep learning framework for financial time series using stacked autoencoders and long short-term memory," PLoS ONE, vol. 12, no. 7, e0180944, 2017.