Advanced Machine Learning

Project: Data analysis

Topic: Stocks Selection Using Neural Network Based on Financial Ratios

Author: Thanasit Chiaravanont

1. Abstract

This project explores the application of Neural Networks to stock selection using financial ratios, leveraging data from the S&P 500 index over a 13-year period (2010–2023). The dataset, sourced from Bloomberg, includes 15 quarterly financial ratios, such as Price-to-Earnings (P/E) ratio, Profit Margin, Gross Margin, Dividend per Share, Debt-to-Equity, Return on Assets (ROA), and Beta, among others, alongside stock prices to calculate returns. Following preprocessing to handle missing values and scaling independent variables, the analysis employed three types of target variables: Buy-recommendation, Sell-recommendation, and 3-month stock return prediction.

A total of nine Neural Network models were developed, varying in architecture and complexity. Classification models (Models 1–6) aimed to predict Buy or Sell actions using binary target variables with a 5% threshold for stock returns. Regression models (Models 7–9) focused on predicting continuous stock returns. The classification models demonstrated moderate performance, with precision values as high as 0.8 in some models. However, they suffered from low recall, indicating challenges with false positive predictions, which carry higher financial risks. Despite incorporating advanced techniques such as adjustable learning rates, batch normalization, and regularization, the return prediction models exhibited minimal predictive power, with R-squared values below 0.004.

The results highlight the limitations of the models in accurately capturing stock patterns, particularly in regression tasks, but suggest potential utility in classification tasks with proper risk management strategies. Enhancements to model recall and careful definition of cut-loss points are critical for mitigating investment losses and improving the robustness of stock selection strategies. This study underscores the complexities of using machine learning for financial prediction and provides a foundation for future exploration in this domain.

2. Data

a. Data structure and cleaning

I decided to focus on stocks included in the S&P 500 index over the past 13 years, from 2010 to 2023. For this analysis, I rely on Bloomberg, widely regarded as the most reliable financial data provider globally. The study examines 15 financial ratios on a quarterly basis, including the Price-to-Earnings (P/E) ratio, profit margin, gross margin, dividend per share, debt-to-assets ratio, debt-to-equity ratio, price-to-book value (PBV) ratio, return on assets (ROA), return on equity (ROE), weighted average cost of capital (WACC), put-option volume, call-option volume, interest coverage ratio, beta, and implied volatility. Additionally, the stock price is analyzed to calculate stock returns using the natural logarithm.

$$r_{i,t} = \ln\left(\frac{P_{i,t}}{P_{i,t-1}}\right)$$

Given:

$r_{i,t} = the\ stock\ return\ of\ stock\ i\ in\ time\ t$
$P_{i,t} = the\ stock\ price\ of\ stock\ i\ in\ time\ t$
$i = stock\ in\ S\&P\ 500$
$t = time$

The dataset contains 26,730 rows, with 57,756 missing values out of a total of 400,950 data points. The variables with the highest number of missing values are Gross Margin (8,298 instances), Interest Coverage Ratio (8,254 instances), Cost of Capital (7,632 instances), and Beta (7,631 instances). Since this dataset is treated as panel data, the time-series effect will be disregarded. Filling in the missing values could shift the data distribution, and I cannot confirm whether all missing values are random. Removing entire columns with a high percentage of missing values is also not ideal, as all 15 factors are proven to have predictive power, supported by financial theory (beyond the scope of this analysis). Ultimately, I decided to drop all rows with missing values, reducing the dataset to only 50% of its original size for the analysis.

PE_Ratio	382
Profit_Margin	3939
Gross_Margin	8298
Dividend_per_Share	3939
Debt_to_Assets	3939
Debt_to_Equity	4083
Price_to_Book	400
Return_on_Assets	3942
Return_on_Equity	4129
Cost_of_Capital	7632
Volume_Put	108
Volume_Call	54
Int_Cov_Ratio	8254
Beta	7631
Implied_Vol	1026

Figure 1 Number of missing values of each financial ratio

b. Data preparation

From an investment perspective, profit can be generated from both buying and selling actions. In this analysis, I classified the target variable into stocks to BUY and stocks to SELL, using a cut-off threshold of 5% (or -5% for a SELL action) in price return over the next 3-month period. A binary variable (0 or 1) is the most appropriate type for this target variable, where a value of 1 indicates a stock return exceeding 5% (or below -5%), and a value of 0 otherwise.

After transforming the target variable into a binary format, I observed that the data is slightly imbalanced, with 30% of instances classified as 1 and 70% as 0. Additionally, it is important to highlight the imbalance in the cost of misclassification. A false positive is more costly than a false negative because a false positive prediction could result in a direct loss of investment.

For another type of target variable, a continuous value is used to predict the actual 3-month return. In the final model, the raw 3-month stock return is directly employed as the target variable.

All independent variables were scaled using a normal distribution to ensure that the size of any variable does not affect the output. Additionally, I included historical 3-month and 6-month returns as features for predicting the target variable, as supported by the price momentum theory.

3. Methods and Model building

To achieve my goal, I will use a Neural Network model to perform the data analysis. The target variables are categorized into three types: BUY-recommendation, SELL-recommendation, and Return prediction. For this analysis, I developed a total of 9 models, each tailored to address specific aspects of these target variables. The table below provides a detailed breakdown of the model construction.

The first six models focus on binary classification tasks, with models 1 to 3 dedicated to BUY-recommendations and models 4 to 6 addressing SELL-recommendations. Models 7 to 9, on the other hand, are designed to predict continuous values for stock returns over a 3-month period.

Each model incorporates specific architectural and functional differences. The layers and nodes vary across models, with models employing either 1 or 4 layers and node sizes ranging from 32 to 256. The binary classification models (1-6) use the sigmoid activation function, while the return prediction models (7-9) omit an activation function for the output layer, as they predict continuous values. All models utilize the adam optimizer, with the learning rate adjusted in specific cases for better performance.

For evaluation, the binary classification models are assessed using accuracy, precision, and recall, while the return prediction models use MAE, RMSE, and R-squared metrics to measure performance.

Table 1

Model	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Class	Binary	Binary	Binary	Binary	Binary	Binary	Cont.	Cont.	Cont.
Type	Buy	Buy	Buy	Sell	Sell	Sell	Return	Return	Return
Hidden Layers	1	4	4	1	4	4	1	4	4
Nodes	32	32 64 128 64	64 128 256 128	32	32 64 128 64	64 128 256 128	32	32 64 128 64	64 128 256 128
Matrices	Accuracy Precision Recall	Accuracy Precision Recall	Accuracy Precision Recall	Accuracy Precision Recall	Accuracy Precision Recall	Accuracy Precision Recall	Mae RMSE R-sq	Mae RMSE R-sq	Mae RMSE R-sq
Activation function	Sigmoid	Sigmoid	Sigmoid	Sigmoid	Sigmoid	Sigmoid	-	-	-
Optimizer	adam	adam	adam	adam	adam	adam	adam	adam	adam
Learning Rate			Adj			Adj			Adj
Data Augmentation	L2 Dropout	L2 Dropout	L2 Dropout Batch normalize	L2 Dropout	L2 Dropout	L2 Dropout Batch normalize	L2 Dropout	L2 Dropout	L2 Dropout Batch normalize

Regularization techniques, such as L2 regularization, dropout, and batch normalization, are employed across the models to mitigate overfitting and enhance generalization.

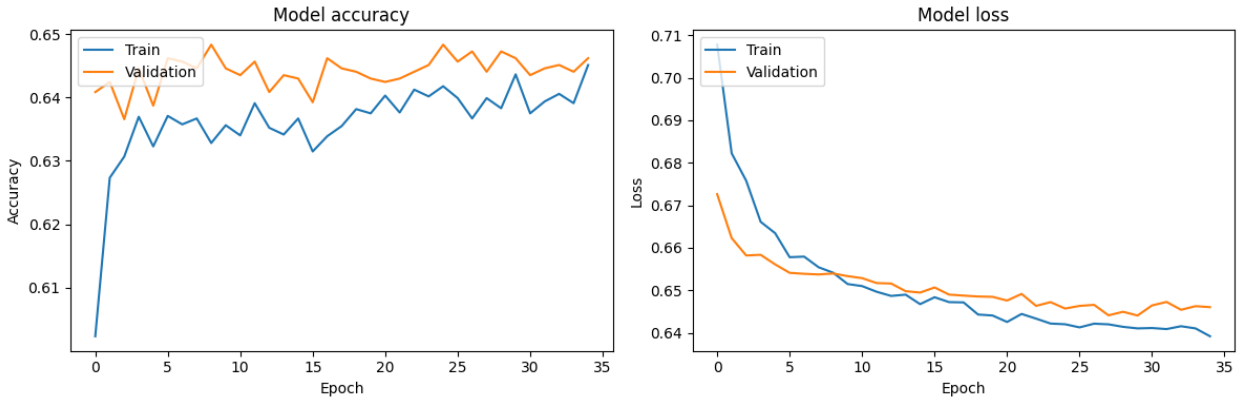
Additionally, data augmentation techniques are applied where necessary to improve model robustness.

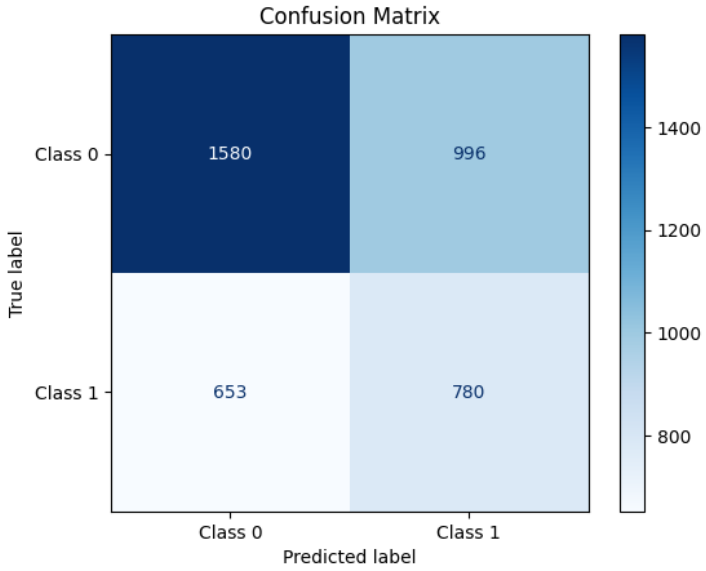
This comprehensive model architecture ensures a robust approach to analyzing financial data and provides actionable insights into investment recommendations and return predictions.

4. Results

a. Buy-Recommendation model

For the first Buy-recommendation model, a simple architecture was implemented with one hidden layer consisting of 32 nodes. The model performed decently, although it began overfitting after 35 epochs. The training accuracy reached 0.635, and the ROC-AUC score was 0.61, indicating performance slightly better than a random model. However, the testing performance was less satisfactory. The testing accuracy was 0.589, and the testing precision was 0.544. Most notably, the testing recall was only 0.439, which is quite low. This indicates that the model struggled with false positive predictions, resulting in a significant number of misclassifications.





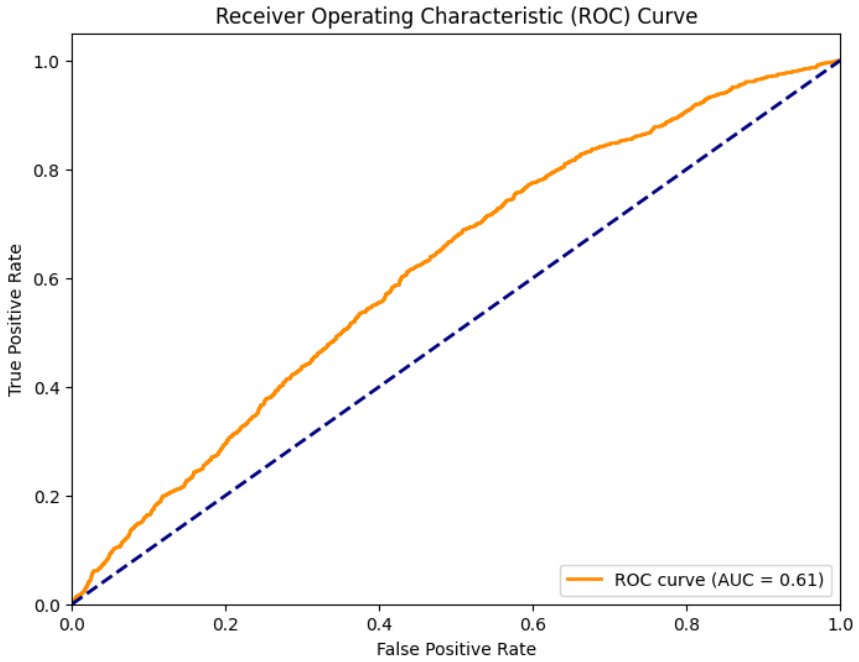
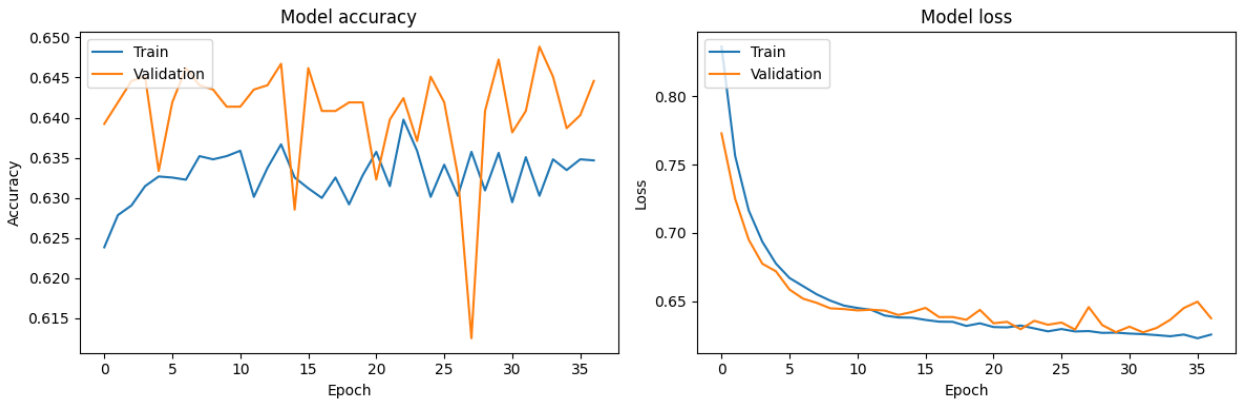


Figure 2

b. Buy-Recommendation model – 2

The second Buy-recommendation model increased its complexity by adding three additional hidden layers, with the largest layer consisting of 128 nodes. The model performed decent, although it began overfitting after 37 epochs. The training accuracy reached 0.645, which is only 1% higher than the baseline model. The ROC-AUC score improved slightly to 0.65, indicating overall performance that was marginally better than the previous model.

However, the testing performance was less satisfactory. The testing accuracy dropped to 0.547, which is lower than the baseline model. On the other hand, the testing precision improved significantly to 0.8, which is impressive and suggests the model is much better at identifying true positives. Despite this improvement, the testing recall remained low at 0.428, indicating the model still struggled with false positive predictions, leading to a substantial number of misclassifications.



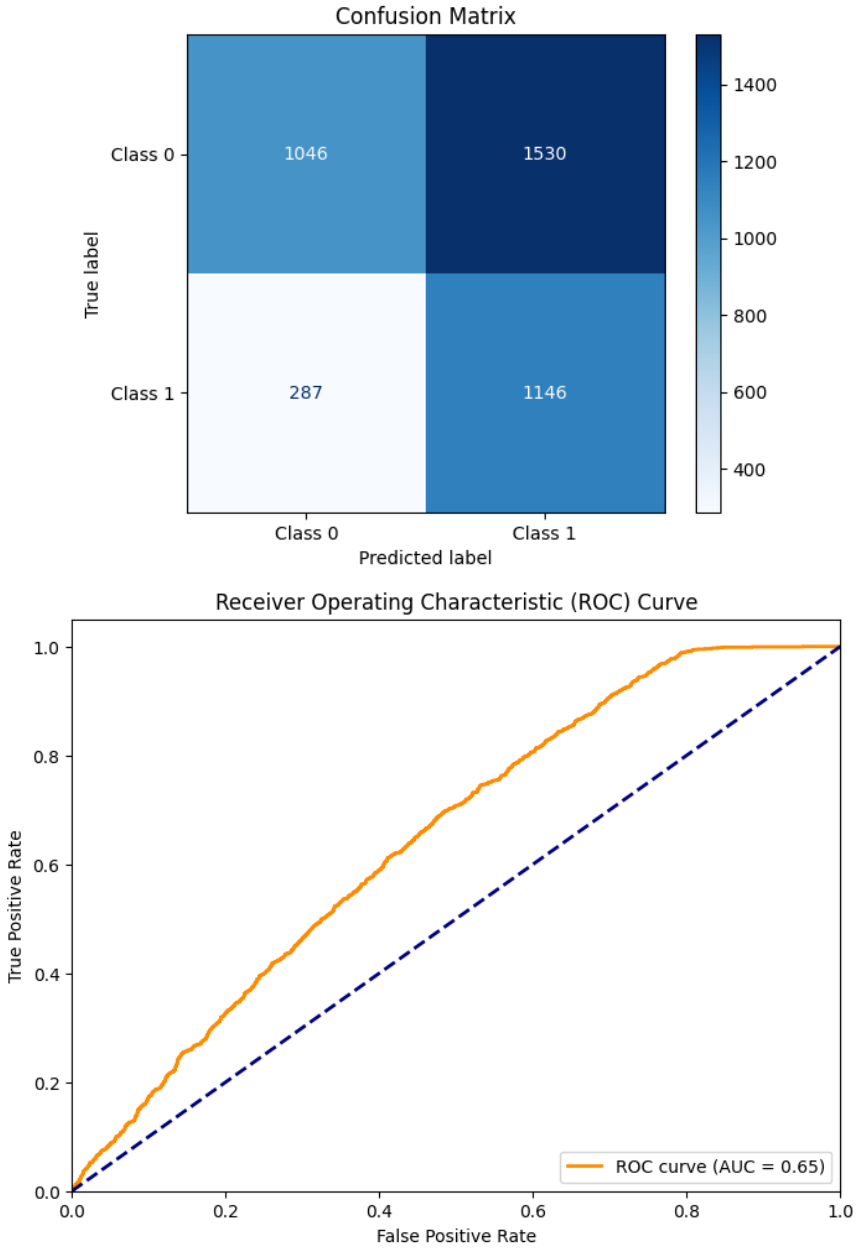


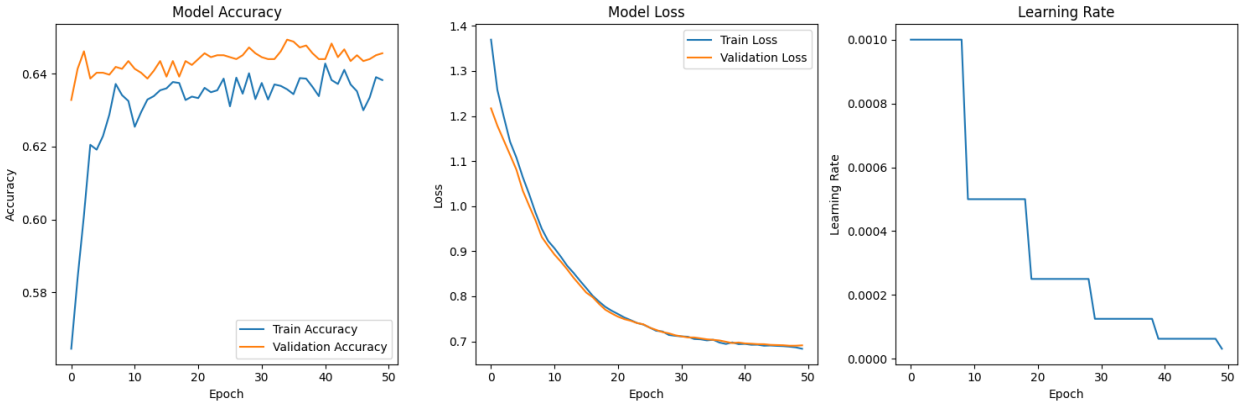
Figure 3

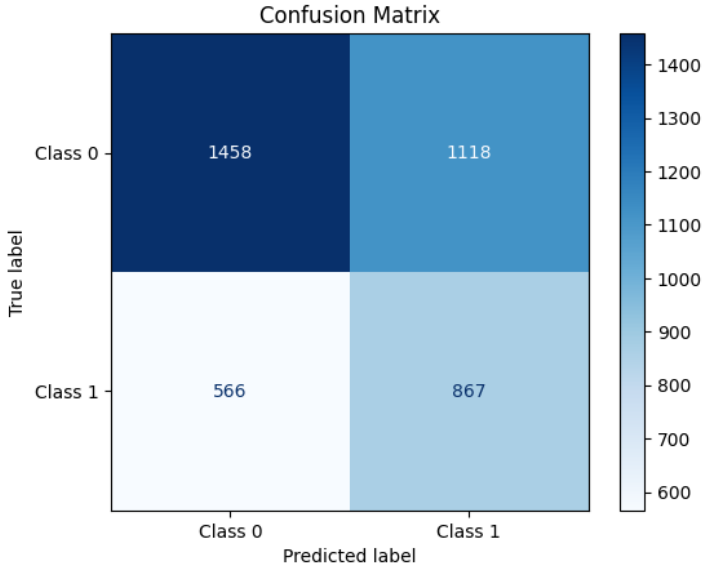
c. Buy-Recommendation model – 3

The third Buy-recommendation model increased its complexity even more by adding an adjustable learning rate, add batch normalization to avoid model overfitting, and increase the number of nodes to 256 nodes in the most complex layers. The model performed reasonably well, although it began overfitting after 50 epochs. The training accuracy drop to 0.635, which is equal to the baseline model. The ROC-AUC score dropped slightly to 0.635, indicating overall performance that was the same as baseline model.

However, the testing performance was less satisfactory. The testing accuracy increased to 0.58, which is equivalent the baseline model. The testing precision dropped to 0.6, which is acceptable. Despite this improvement, the testing recall remained low at 0.437, indicating the

model still struggled with false positive predictions, leading to a substantial number of misclassifications.





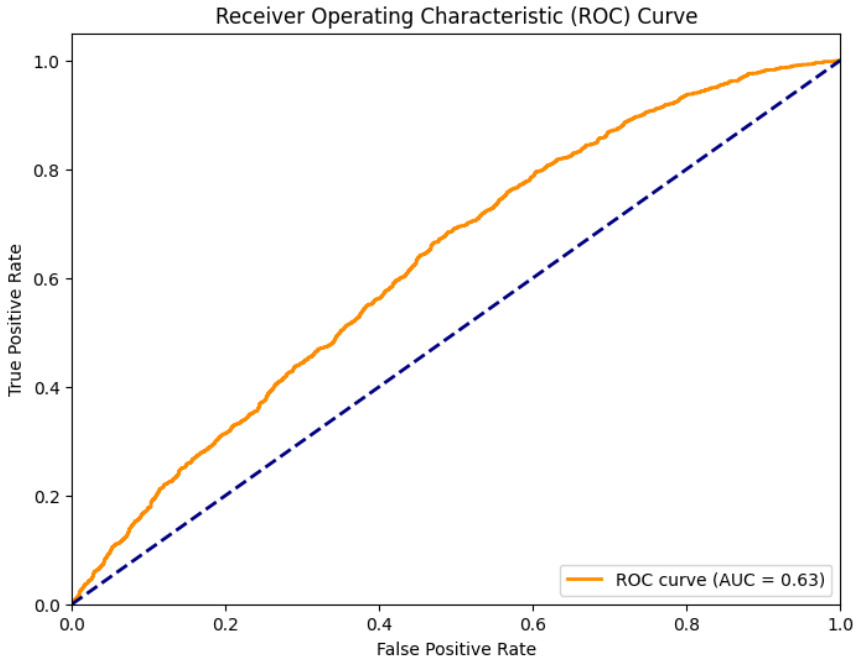
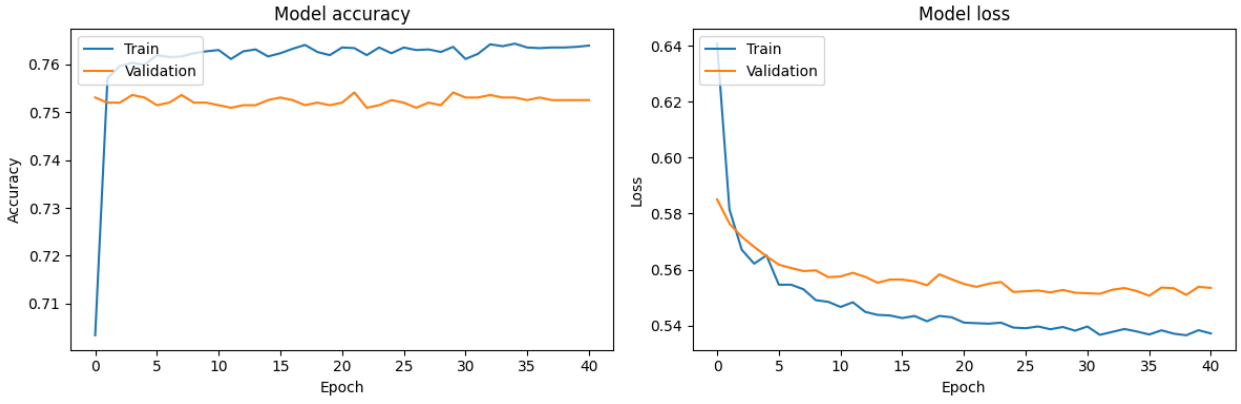


Figure 4

d. Sell-Recommendation model

For the first Sell-recommendation model, a simple architecture was implemented with one hidden layer consisting of 32 nodes. The model performed decently, although it began overfitting after 41 epochs. The training accuracy reached 0.755, and the ROC-AUC score was 0.58, indicating performance slightly better than a random model.

However, the testing performance was less satisfactory. The testing accuracy was 0.6, and the testing precision was 0.526. Most notably, the testing recall was only 0.306, which is very low. This indicates that the model struggled with false positive predictions, resulting in a significant number of misclassifications. This is a sign of overfitting.



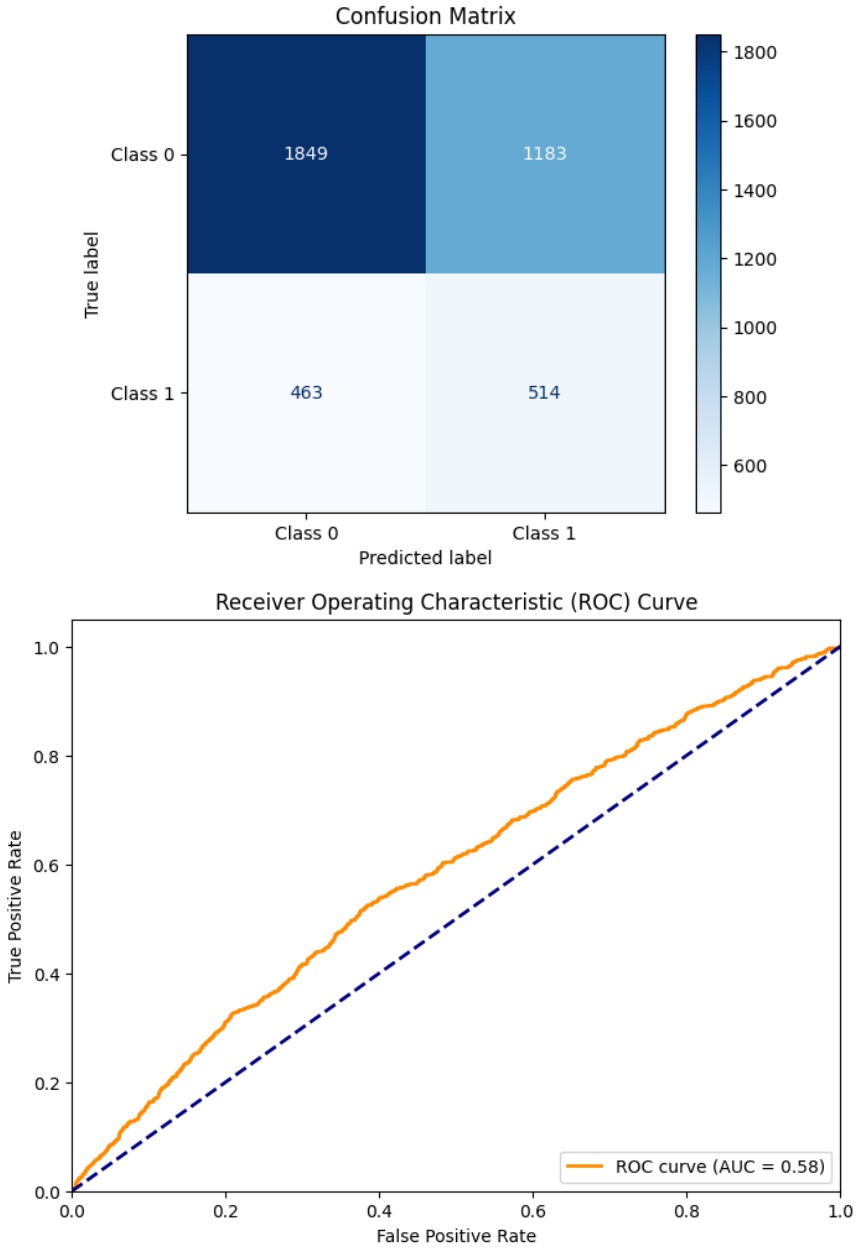


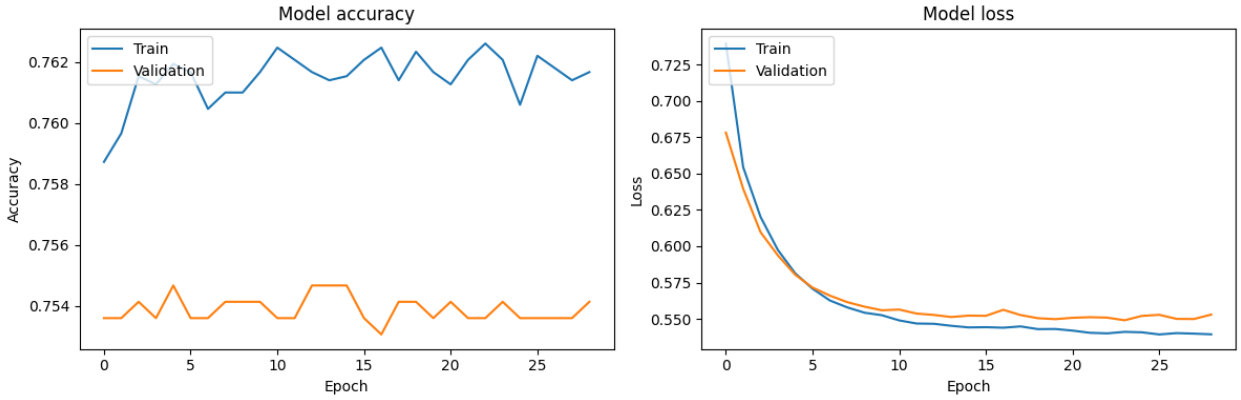
Figure 5

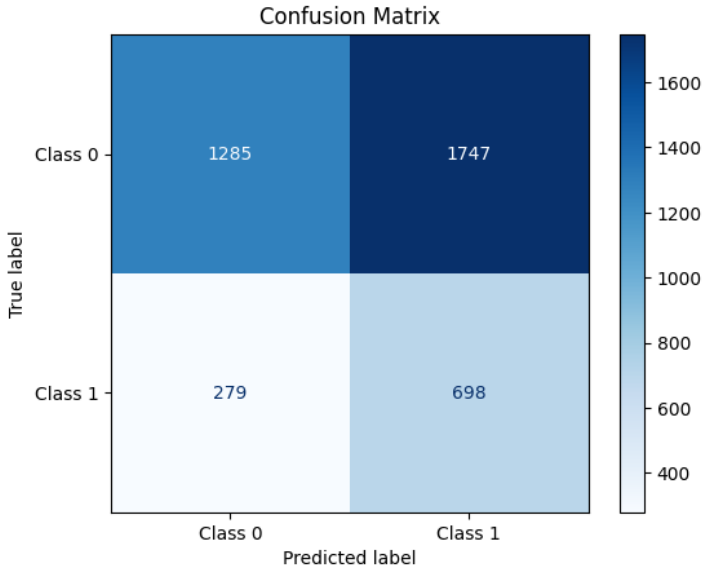
e. Sell-Recommendation model – 2

The second Sell-recommendation model increased its complexity by adding three additional hidden layers, with the largest layer consisting of 128 nodes. The model performed decent, although it began overfitting after 29 epochs. The training accuracy reached 0.76, which is identical as baseline model. The ROC-AUC score improved slightly to 0.6, indicating overall performance that was the same as the baseline model.

However, the testing performance was less satisfactory. The testing accuracy dropped to 0.49, which is lower than the baseline model. On the other hand, the testing precision improved significantly to 0.7, which is impressive and suggests the model is much better at identifying true positives. Despite this improvement, the testing recall is extremely low at 0.285, indicating

the model still struggled with false positive predictions, leading to a substantial number of misclassifications. This is a sign of overfitting.





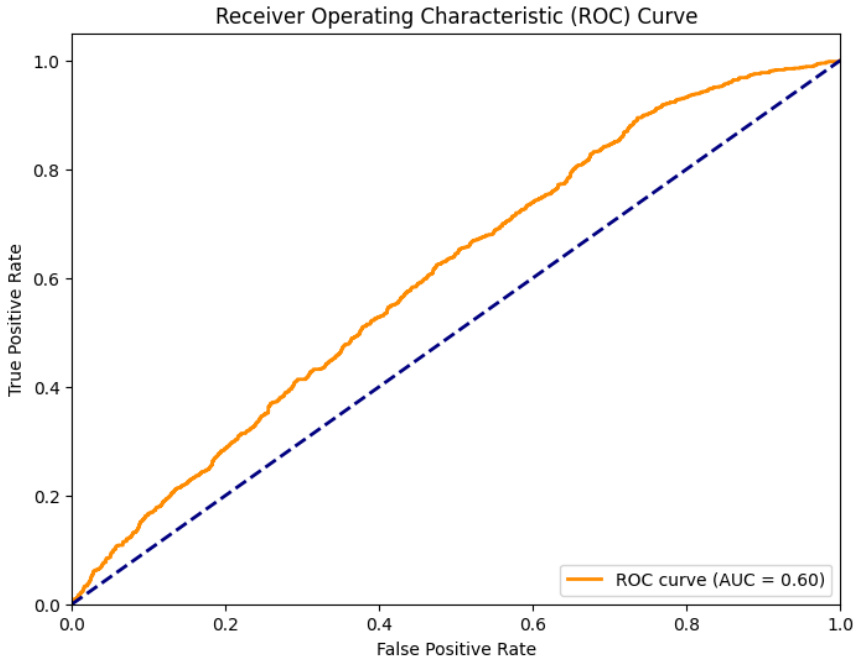
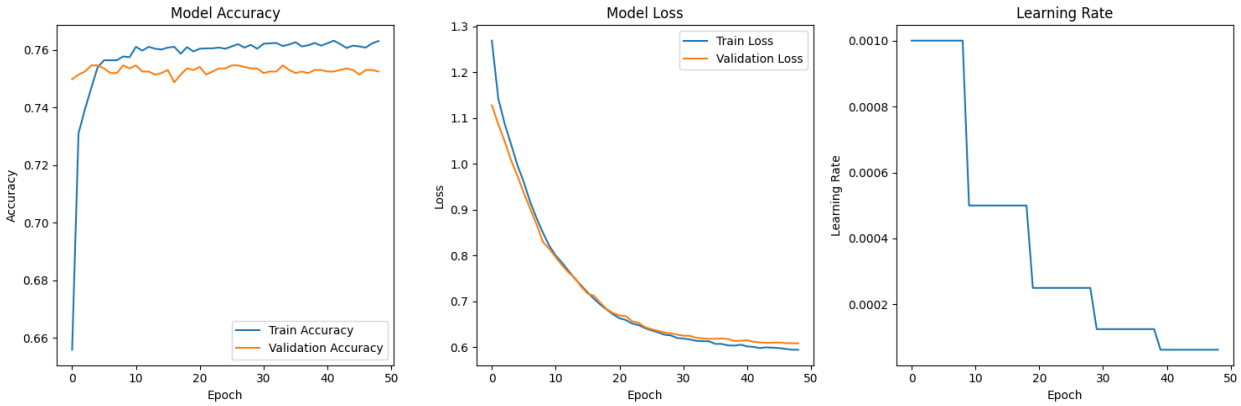


Figure 6

f. Sell-Recommendation model – 3

The third Buy-recommendation model increased its complexity even more by adding an adjustable learning rate, add batch normalization to avoid model overfitting, and increase the number of nodes to 256 nodes in the most complex layers. The model performed reasonably well, although it began overfitting after 49 epochs. The training accuracy stayed around 0.76, which is equal to the baseline model. The ROC-AUC score dropped slightly to 0.59, indicating overall performance that was the same as baseline model.

However, the testing performance was less satisfactory. The testing accuracy stayed at 0.58, which is a little bit better than the baseline model. The testing precision dropped to 0.53, which is a bit lower than model-2. Despite this improvement, the testing recall remained low at 0.425, indicating the model still struggled with false positive predictions, leading to a substantial number of misclassifications. This is a sign of overfitting.



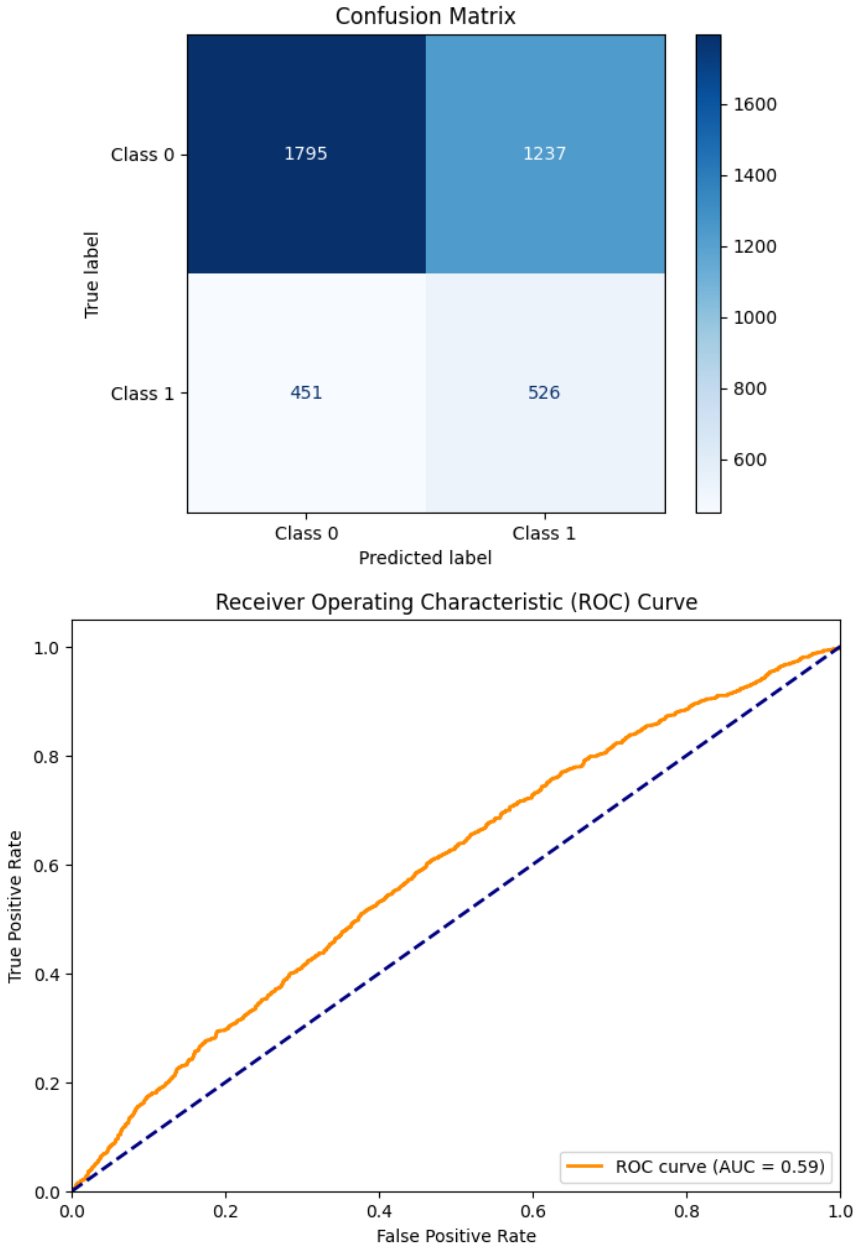


Figure 7

g. Return prediction model

For the first return prediction model, a simple architecture with one hidden layer consisting of 32 nodes was implemented. The model showed signs of overfitting after 47 epochs. Its performance indicated no predictive power, as the R-squared value was only 0.0028, demonstrating that the predictions were highly inaccurate.

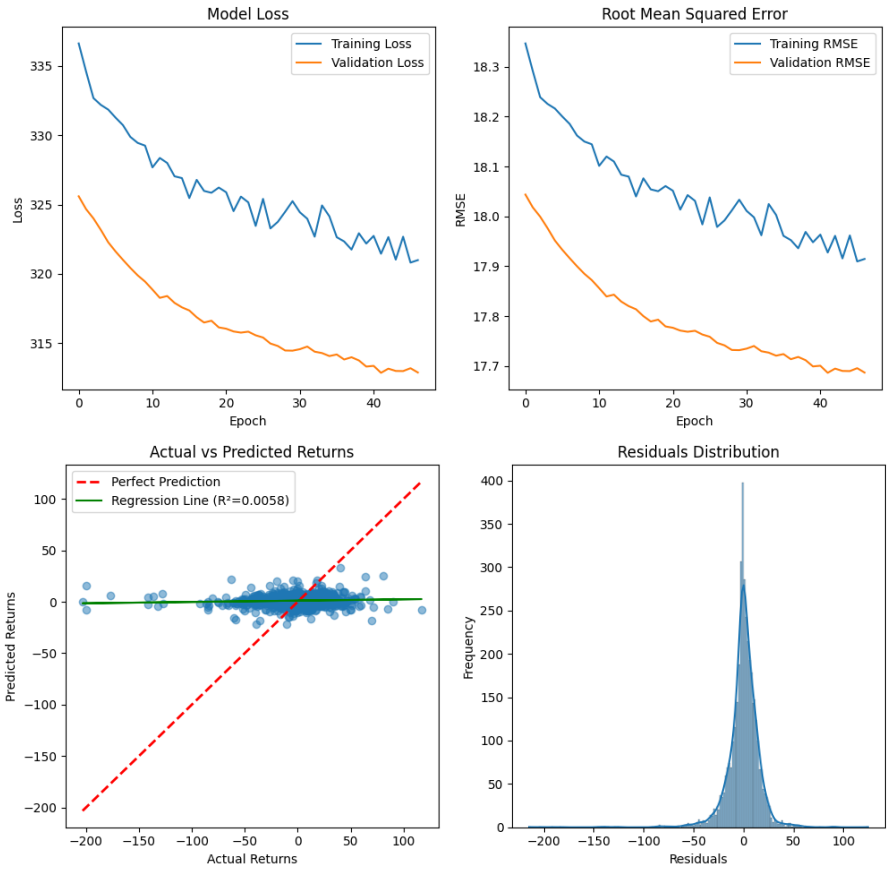
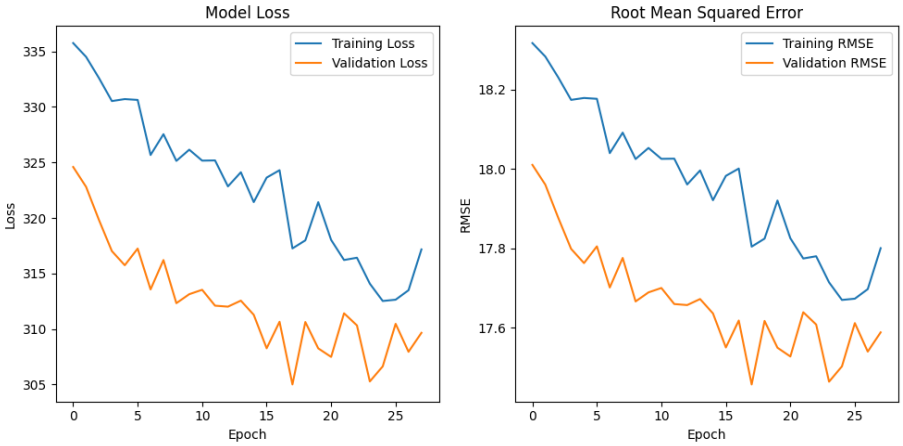


Figure 8

h. Return prediction model -2

The second Sell-recommendation model increased its complexity by adding three additional hidden layers, with the largest layer consisting of 128 nodes. While the model performed decently, it began overfitting after 28 epochs, reaching the overfitting phase faster than the baseline model. The R-squared value remained at 0.0028, identical to that of the baseline model. This indicates that increasing the model's complexity did not improve its ability to capture patterns in the data.



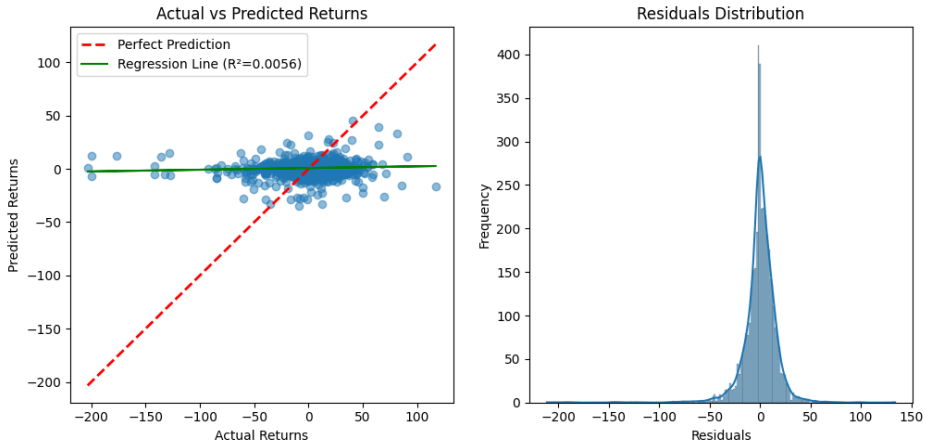
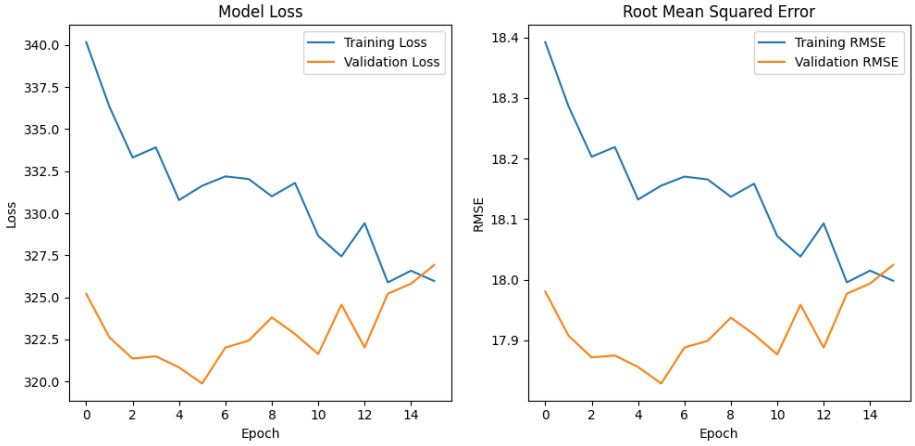


Figure 9

i. Return prediction mode – 3

The third Buy-recommendation model further increased its complexity by incorporating an adjustable learning rate, adding batch normalization to mitigate overfitting, and increasing the number of nodes to 256 in the most complex layers. Despite these adjustments, the model began overfitting after just 16 epochs.

Even with regularization techniques in place, the model entered the overfitting phase much earlier than the previous models. The R-square is 0.0036 which is better than previous two models however, its performance was extremely poor, failing to make any meaningful predictions, rendering it ineffective for this task.



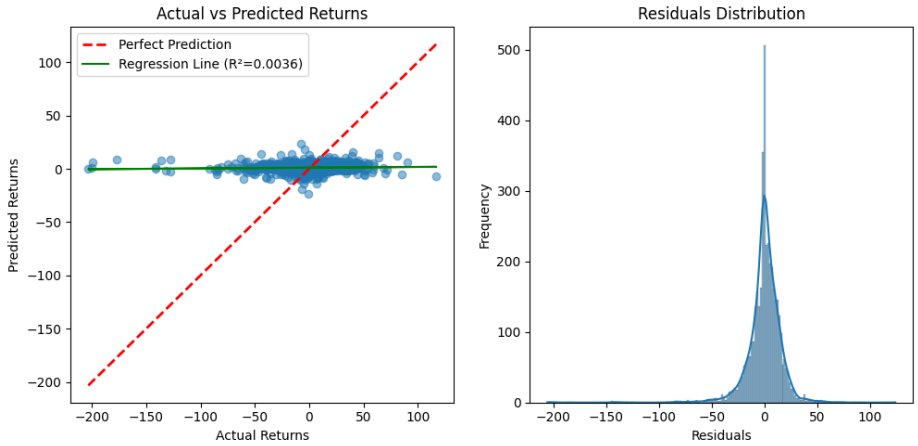


Figure 10

5. Summary

For the classification models (Models 1–6), the testing accuracy is only marginally better than that of a random model. The F1-scores are notably low, indicating that these models perform poorly overall. However, one positive aspect is their ability to predict the target outcome (1: buy or sell stocks) with relatively high testing precision. Among the Buy-recommendation models, Model 2 performed the best, while for Sell-recommendation predictions, Model 6 showed the highest overall performance.

The return prediction models (Models 7–9), which use continuous target variables, performed extremely poorly. With R-squared values of 0.0013, 0.0028, and 0.0036, and RMSE values around 16.97, these models failed to capture any meaningful patterns in the data, rendering them ineffective for this task.

In conclusion, between the two types of predictive models (classification and regression), only the classification models showed any predictive potential. While they demonstrated decent testing precision, their low testing recall remains a significant issue, leading to frequent misclassifications. This is particularly problematic as high false positives (true positives predicted as 1) result in greater financial costs compared to false negatives.

Table 2

Model	1	2	3	4	5	6	7	8	9
Tr. Acc	0.635	0.645	0.635	0.755	0.76	0.76			
ROC	0.61	0.65	0.635	0.58	0.6	0.59			
Te. Acc	0.589	0.54	0.58	0.6	0.49	0.58			
Te. Pre	0.544	0.8	0.6	0.526	0.7	0.53			
Te. Rec	0.439	0.428	0.437	0.306	0.285	0.425			
F1	0.239	0.342	.262	0.16	.20	0.225			
Epochs	35	37	50	41	29	49			
R-sq							0.0013	0.0028	0.0036
RMSE							16.98	16.99	16.97

To effectively use these models in practice, robust risk management strategies are crucial. For example, defining a clear cut-loss point can help mitigate substantial investment

losses caused by misclassifications. Further improvements to these models are necessary to enhance their recall and reduce the financial impact of mispredictions.

6. References

[1] Chiaravanont, T. (n.d.). *Investment recommendation* [Final project for Fundamental Machine Learning, Master of Science in Business Analytics, Kent State University].

[2] Tsai, P.-F., Gao, C.-H., & Yuan, S.-M. *Stock selection using machine learning based on financial ratios*. Mathematics 2023, 11, 4758.