Data Mining Complex Data Types

Corey Waldner

School of Technology and Engineering, National University

TIM-8131 v1: Data Mining

Dr. Reza Hashemi

September 25, 2024

Data Mining Complex Data Types

The emergence of the World Wide Web in the 1990s marked the onset of the Information Age. However, some argue that we are, in fact, currently residing in the data age, given the colossal amounts of data being generated due to the advent of the internet (Han, Kamber, & Pei, 2012). Every industry collects substantial data, necessitating analysis to extract its value. For this purpose, data mining was developed, a process of sifting through vast amounts of data. Data Mining encompasses techniques for efficiently and automatically discovering previously unknown, valid, novel, practical, and understandable patterns in data. Data mining deals with large volumes of data, ranging from gigabytes to terabytes or even zettabytes (Bhatia, 2019). Essentially, data mining empowers businesses to identify historical patterns and forecast future behavior.

This technical paper will explore two data mining techniques applied to a large, intricate dataset. These techniques will be employed in the analysis of S&P 500 Stocks data. The S&P 500, globally renowned as the most prominent financial benchmark, monitors the performance of 500 major companies listed on U.S. stock exchanges. As of December 31, 2020, investments totaling over $5.4 trillion were linked to the index's performance. It is worth noting that due to the inclusion of multiple classes of stock for certain constituent companies, such as Alphabet's Class A (GOOGL) and Class C (GOOG), the index encompasses 505 stocks (Larxel, 2024).

Classification Technique – Random Forest

The initial method applied to this dataset is a classification technique. In data mining, classification assigns labels to each instance, record, or data object in a dataset based on their features or attributes. The classification approach aims to accurately predict class labels for new, unseen data (Utkarsh, 2023). The classification algorithm utilized on the S&P 500 dataset is

Random Forest. Random Forest is a widely used machine learning algorithm developed by Leo Breiman and Adele Cutler that aggregates the output of multiple decision trees to produce a single result (IBM, 2024). Decision trees are intuitive models that mimic human decision-making using an "if-then" rule structure resembling a flowchart. Each decision creates a new branch or "node," ultimately agreeing or disagreeing with the objective. However, decision trees have a crucial limitation: they tend to overfit, becoming overly complex as more data is added, capturing noise rather than meaningful patterns.

In order to tackle the problem of overfitting, the Random Forest algorithm utilizes a collection, or "forest," of numerous decision trees, each of which is trained on different random subsets of the data and features. This technique, known as bagging (Bootstrap Aggregating), helps to construct a resilient model that reduces variance and improves accuracy. Each tree in the forest generates a prediction, and the final result is determined by either the majority vote or the average of the predictions from all the trees (Gross, 2020).

In a Random Forest model, each tree only makes splits based on a random subset of the features. This approach introduces diversity among the trees and prevents them from focusing on the same dominant feature. As a result, Random Forest tends to generalize better and reduce overfitting compared to a single decision tree. An apt analogy for Random Forest is that a crowd (forest) of decision trees can collectively make better decisions than an individual decision tree, a concept known as the "wisdom of the crowd" (Ravindran, 2021).

Shallow Neural Network

The second technique used on the S&P 500 dataset was a Shallow Neural Network. A Shallow Neural Network falls under the realm of an Artificial Neural Network (ANN). ANN imitates the human brain's process of information processing. Put simply, the human brain takes in input through dendrites and outputs the necessary information through axons. Dendrites receive information, while axons relay the information to the correct neurons or 'nodes' for processing, allowing the brain to make decisions rapidly (Wood, 2022). The process is mirrored in an ANN, which transforms information and decisions through multiple hidden layers. In an ANN architecture, information is received in an input layer (analogous to dendrites), processed in a hidden layer(s) (analogous to neurons), and the result of the classification or decision is delivered through the output layer (analogous to axons).

The entire process is conducted using calculus-based optimization techniques. In simple mathematical terms, an ANN can be expressed as $Y = f(X)$, where $Y= f(g(b))$, and $X = g(b)$. Here, b, X, and Y represent vectors or, more generally, tensors. In this context, vectors are one-dimensional arrays of numbers, matrices are two-dimensional arrays, and tensors denote n-dimensional arrays. The training process of an ANN essentially entails determining optimal values for the coefficients b to complete the mapping (Sarker, 2021)

Training

The ANN training process comprises three primary stages: forward propagation, loss function calculation, and backward propagation. Forward propagation involves passing input data through the network, layer by layer, to generate an output in a forward direction. During this process, the data is processed by hidden layers, which apply activation functions and pass the transformed data to the next layer. Forward propagation ensures data moves linearly through the network, preventing circular data flow and generating a valid output (H2O.Ai, 2024).

After the forward propagation process, the loss function is computed. This function evaluates the disparity, or "error," between the network's predicted and target outputs. The loss function quantifies the extent to which the network's predictions deviate, thereby aiding in the network's adjustment (Medium, 2023).

After the initial calculations, backpropagation adjusts the network's parameters. It involves computing the gradient of the loss function about the neural network's weights and biases. This backward process employs the chain rule from calculus to iteratively modify the weights, effectively reducing errors in subsequent forward passes (D2L, 2024).

Appling both Techniques

The objective was to determine whether the Random Forest Classifier or the Shallow Neural Network could accurately predict if the S&P 500 index would be higher or lower than the previous day. To achieve this, several preprocessing steps were necessary to prepare the dataset for analysis. Initially, the dataset contained approximately 98,000 missing values out of 1.8 million rows, primarily associated with companies not part of the S&P 500 during specific trading days or due to non-trading days. These missing values were dropped to maintain data integrity. Subsequently, a new DataFrame (DF) was created to establish a prediction target, incorporating a target variable that indicated whether the closing price of the S&P 500 for the next day was higher than the previous day's closing price.

Specifically, if the next day's closing price was higher, the target variable was assigned a value of 1; if it was lower or the same, it was assigned a value of 0. This binary target facilitated effective classification by both machine learning algorithms. Finally, the preprocessed dataset was split into training and testing sets, allowing for the evaluation of model performance on unseen data for both the Random Forest Classifier and the Shallow Neural Network.

Performance

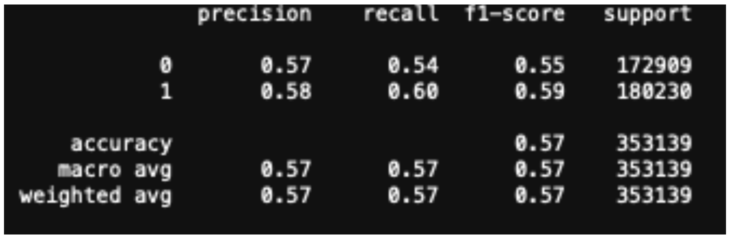
The dataset used in this analysis was very large, leading to significant processing times for both algorithms. The goal was to leverage the entire dataset to maximize the model's ability to predict whether the following day's stock price would be higher. Various strategies could be employed to improve processing efficiency, such as reducing the dataset to a random sample, utilizing alternative classification algorithms, reducing the number of features, or implementing parallel processing.

One notable advantage of the Random Forest algorithm is its ability to process in parallel (Ishwaran & Lu, 2024), significantly enhancing speed and efficiency. The code was modified to utilize multiple CPU cores, allowing the algorithm to process the trees concurrently. Initially, features were reduced to expedite processing, but subsequent tests showed that reintroducing the four original features improved the model's predictive performance. In contrast, the Shallow Neural Network also exhibited slow performance. However, it required no modifications to the code and could process the data, albeit at a reduced speed.

Results

As shown in Figure 1, the Random Forest model displayed balanced performance between classes (0 and 1) in precision, recall, and F1-score. The accuracy achieved by Random Forest was 57%, which is better than random guessing, which would have a 50/50 chance. The confusion matrix indicates a reasonable number of correct predictions (True Positives and True

Figure 1
Output Random Forest



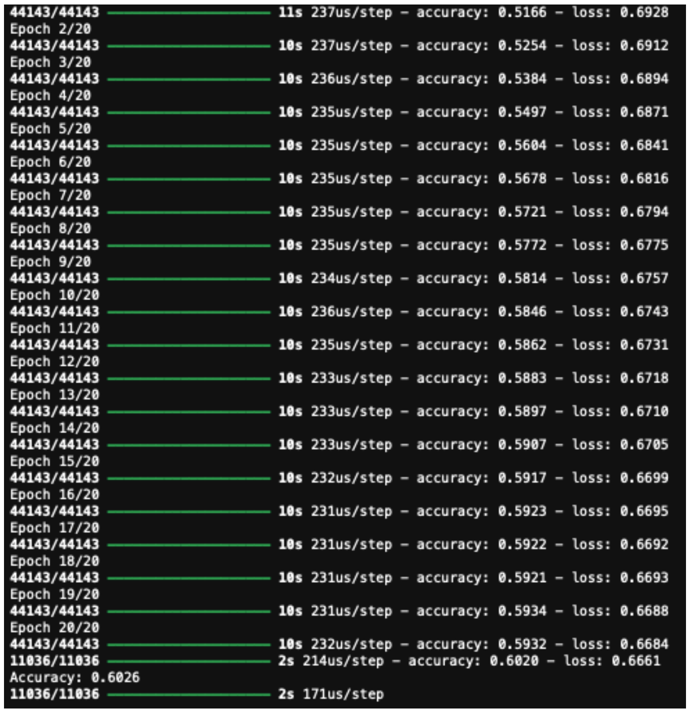
```
              precision    recall  f1-score   support

           0       0.57      0.54      0.55    172909
           1       0.58      0.60      0.59    180230

    accuracy                           0.57    353139
   macro avg       0.57      0.57      0.57    353139
weighted avg       0.57      0.57      0.57    353139
```

Note. This was performed with Python utilizing the Random Forest algorithm.

Negatives). However, there were still significant misclassifications (False Positives and False Negatives), highlighting areas where the model's predictive power could be improved.

As shown in Figure 2, the results from the Shallow Neural Network are displayed. After completing 20 training epochs (iterations), the model achieved an accuracy of 60.95% on the test set, indicating that it correctly predicted whether the stock would be higher or lower around 61% of the time. The test loss was 0.6650, slightly lower than the final training loss, which suggests the model performed well on unseen data without overfitting. Throughout the 20 epochs, accuracy steadily increased, starting from 51.29% in the first epoch and reaching 59.47% by the last epoch. The loss also consistently decreased, indicating that the model became better at making accurate predictions as training continued. Toward the final epochs, the rate of improvement slowed, suggesting the model began to converge and stabilize as it learned a reliable pattern in the data. Overall, the Shallow Neural Network exhibited slightly higher predictability than the Random Forest.

Figure 2
Output for Shallow Neural Network



```
44143/44143 ━━━━━━━━━━━━━━━━━━━━ 11s 237us/step - accuracy: 0.5166 - loss: 0.6928
Epoch 2/20
44143/44143 ━━━━━━━━━━━━━━━━━━━━ 10s 237us/step - accuracy: 0.5254 - loss: 0.6912
Epoch 3/20
44143/44143 ━━━━━━━━━━━━━━━━━━━━ 10s 236us/step - accuracy: 0.5384 - loss: 0.6894
Epoch 4/20
44143/44143 ━━━━━━━━━━━━━━━━━━━━ 10s 235us/step - accuracy: 0.5497 - loss: 0.6871
Epoch 5/20
44143/44143 ━━━━━━━━━━━━━━━━━━━━ 10s 235us/step - accuracy: 0.5604 - loss: 0.6841
Epoch 6/20
44143/44143 ━━━━━━━━━━━━━━━━━━━━ 10s 235us/step - accuracy: 0.5678 - loss: 0.6816
Epoch 7/20
44143/44143 ━━━━━━━━━━━━━━━━━━━━ 10s 235us/step - accuracy: 0.5721 - loss: 0.6794
Epoch 8/20
44143/44143 ━━━━━━━━━━━━━━━━━━━━ 10s 235us/step - accuracy: 0.5772 - loss: 0.6775
Epoch 9/20
44143/44143 ━━━━━━━━━━━━━━━━━━━━ 10s 234us/step - accuracy: 0.5814 - loss: 0.6757
Epoch 10/20
44143/44143 ━━━━━━━━━━━━━━━━━━━━ 10s 236us/step - accuracy: 0.5846 - loss: 0.6743
Epoch 11/20
44143/44143 ━━━━━━━━━━━━━━━━━━━━ 10s 235us/step - accuracy: 0.5862 - loss: 0.6731
Epoch 12/20
44143/44143 ━━━━━━━━━━━━━━━━━━━━ 10s 233us/step - accuracy: 0.5883 - loss: 0.6718
Epoch 13/20
44143/44143 ━━━━━━━━━━━━━━━━━━━━ 10s 233us/step - accuracy: 0.5897 - loss: 0.6710
Epoch 14/20
44143/44143 ━━━━━━━━━━━━━━━━━━━━ 10s 233us/step - accuracy: 0.5907 - loss: 0.6705
Epoch 15/20
44143/44143 ━━━━━━━━━━━━━━━━━━━━ 10s 232us/step - accuracy: 0.5917 - loss: 0.6699
Epoch 16/20
44143/44143 ━━━━━━━━━━━━━━━━━━━━ 10s 231us/step - accuracy: 0.5923 - loss: 0.6695
Epoch 17/20
44143/44143 ━━━━━━━━━━━━━━━━━━━━ 10s 231us/step - accuracy: 0.5922 - loss: 0.6692
Epoch 18/20
44143/44143 ━━━━━━━━━━━━━━━━━━━━ 10s 231us/step - accuracy: 0.5921 - loss: 0.6693
Epoch 19/20
44143/44143 ━━━━━━━━━━━━━━━━━━━━ 10s 231us/step - accuracy: 0.5934 - loss: 0.6688
Epoch 20/20
44143/44143 ━━━━━━━━━━━━━━━━━━━━ 10s 232us/step - accuracy: 0.5932 - loss: 0.6684
11036/11036 ━━━━━━━━━━━━━━━━━━━━ 2s 214us/step - accuracy: 0.6020 - loss: 0.6661
Accuracy: 0.6026
11036/11036 ━━━━━━━━━━━━━━━━━━━━ 2s 171us/step
```

Note. This was performed with Python utilizing the Shallow Neural Network algorithm.

Future Research

In future research, enhancing the learning models by incorporating additional features would be beneficial. The current dataset comprises three different files: stock data, company listings, and the daily values of the overall S&P 500 index. A new dataset could be generated by merging these files based on the stock symbol and including additional features such as Sector and Industry to explore potential correlations. For instance, analyzing gains in the consumer electronics sector might reveal a correlation with increases in the financial services sector. Moreover, it would be worthwhile to consider geographical features, such as assessing whether companies headquartered in specific regions, like Amazon, Microsoft, Costco, Boeing, or Starbucks in Washington state, have a significant impact on the overall performance of the S&P 500 index.

In addition to integrating new features, there is potential to fine-tune further and refine the algorithms. For Random Forest, it would be beneficial to delve deeper into hyperparameter tuning and consider incorporating techniques like feature importance analysis or boosting to enhance performance. As for the Shallow Neural Network, expanding it into a deeper network with additional hidden layers could enhance its predictive capability by capturing more intricate relationships in the data.

Conclusion

This technical paper explores two data mining techniques, Random Forest and Shallow Neural Network, which were applied to the S&P 500 dataset to predict whether the index would be higher the following day. The results indicated that Random Forest demonstrated balanced performance, but it left room for improvement due to misclassifications. On the other hand, the Shallow Neural Network exhibited slightly higher accuracy, achieving a 61% success rate in predicting stock price movements. Both techniques displayed strengths in handling large datasets, but their predictive power could benefit from further tuning and feature engineering. This research underscores the potential of data mining in financial markets and demonstrates how these tools can uncover meaningful patterns.

References

Bhatia, P. (2019). Introduction to Data Mining. In Data Mining and Data Warehousing: Principles and Practical Techniques. *Cambridge: Cambridge University Press*, pp. 17-27.

D2L. (2024, September 14). *5.3. Forward Propagation, Backward Propagation, and Computational Graphs*. Retrieved from Dive into Deep Learning: https://d2l.ai/chapter_multilayer-perceptrons/backprop.html

Gross, K. (2020, April 6). *ree-Based Models: How They Work (In Plain English!)*. Retrieved from Dataiku: https://blog.dataiku.com/tree-based-models-how-they-work-in-plain-english

H2O.Ai. (2024, September 14). *What is a Neural Network?* Retrieved from H20.Ai: https://h2o.ai/wiki/forward-propagation/#:~:text=Forward%20propagation%20is%20the%20way,loss%20function%20or%20error%20rate.

Han, J., Kamber, M., & Pei, J. (2012). *Data Mining: Concepts and Techniques, 3rd Edition.* Elsevier: Morgan Kaufmann.

IBM. (2024, September 25). *What is random forest?* Retrieved from IBM Topics: https://www.ibm.com/topics/random-forest

Ishwaran, H., & Lu, M. (2024, September 26). *Parallel Processing*. Retrieved from Random Forest SRC: https://www.randomforestsrc.org/articles/parallel.html

Larxel, A. (2024, September 25). *S&P 500 Stocks (daily updated)*. Retrieved from Kaggle: https://www.kaggle.com/datasets/andrewmvd/sp-500-stocks

Medium. (2023, May 26). *Deep Learning Course — Lesson 5: Forward and Backward Propagation*. Retrieved from Medium: https://medium.com/@nerdjock/deep-learning-course-lesson-5-forward-and-backward-propagation-ec8e4e6a8b92

Ravindran, D. (2021, June 19). *Tree-Based Machine Learning Algorithms Explained*. Retrieved from Medium: https://medium.com/analytics-vidhya/tree-based-machine-learning-algorithms-explained-b50937d3cf8e

Sarker, I. (2021). Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions. *SN Computer Science*, 2(6).

Utkarsh. (2023, May 25). *Scaler*. Retrieved from Classification in Data Mining: https://www.scaler.com/topics/data-mining-tutorial/classification-in-data-mining/

Wood, T. (2022, July 5). *How similar are Neural Networks to our Brains?* Retrieved from Fast Data Science: https://fastdatascience.com/ai-in-research/how-similar-are-neural-networks-to-our-brains/