**DLMF Project Assignment – July 2023**

Rodrigo Maia - 20220517

Riccardo Gurzu - 20221235

Neftali Herculano - m20200732

Francisco Feria - 20221538

# Usage of Deep learning to improve Company Bankruptcy Prediction

## Abstract

Bankruptcy prediction is a critical task in financial risk assessment. This report documents the process of finding the best neural network model for predicting bankruptcy using deep learning techniques. The report provides an in-depth analysis of artificial neural networks, the role of nodes and layers, the importance of epochs, hyperparameter tuning, and the application of advanced techniques such as early stopping and dropout. The results and insights gained from the project are presented, along with suggestions for future research and enhancements.

The objective of our project is to find the best model in terms of Recall using deep learning techniques. It is challenging because the dataset has many features but less than 7 thousand rows. Also, a high imbalance positive bankruptcy of only 3% is characteristic of this type of study.

## 1. Introduction

Bankruptcy prediction is a fundamental challenge in financial risk assessment, requiring accurate modelling techniques. This report aims to find the best model for bankruptcy prediction using artificial neural networks (ANNs). ANNs are computational models inspired by the human brain and are highly effective in capturing complex relationships between variables.

The data were collected from the Taiwan Economic Journal for the years 1999 to 2009. Company bankruptcy was defined based on the Taiwan Stock Exchange business regulations. The data is rich in features in a total of 96, but not many rows, a total of 6819. We will have to take most of the main features to avoid overfitting.

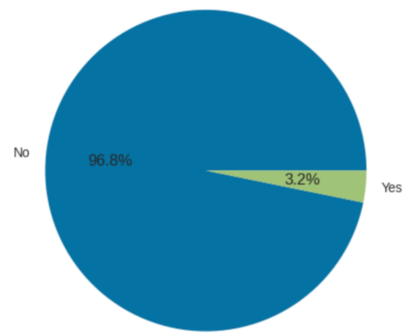
Another useful information about the data is the high level of ratios analysis. Normally, this kind of data has a high correlation between features because numerators and denominators can be similar in different features, causing overfitting.

2. Data Exploration

Data exploration is crucial to understanding the dataset's characteristics and addressing any data quality issues. Missing values and duplicated data were analysed with no ocurrencies, ensuring reliable model input. This step ensures that the ANNs receive clean and reliable data for training and evaluation.

At first, we notice a unique situation in bankruptcy cases. Normally, we have a low level of positive cases in this kind of study. Moreover, here, it is no different. Therefore, we found just 3.2% per cent of positive cases. That is crucial to find the best performance analysis. We learned from the classes that accuracy or F1 Score is not always the best estimate. The best practice is to find a pack of estimators to help us make a decision. We decided to use Recall because False Negatives represent the observations that were not classified as not a default when they defaulted, see Garcia J.(2022). The formula will give us exactly what we need in this business: True Positive/ (True Positive + False Negative). We understand that once we avoid a false negative, the false positives can be analyzed with additional traditional tools.

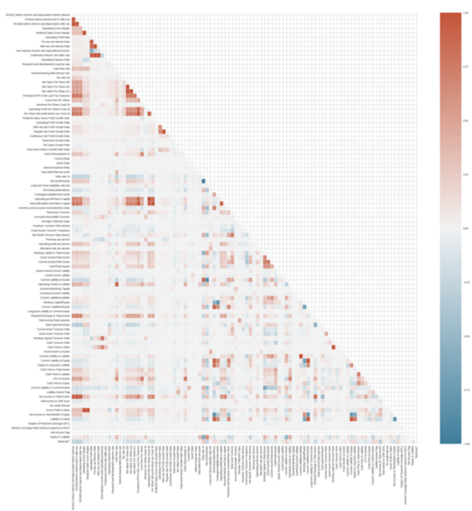
Bankruptcy (Yes/No)




Another technique we will have to use is a way to balance the data. We will use SMOTE function to balance the data. One important thing about this tool is not used on the test dataset because we know that feature analysis will not have this 50%/50% behaviour to predict bankruptcy.

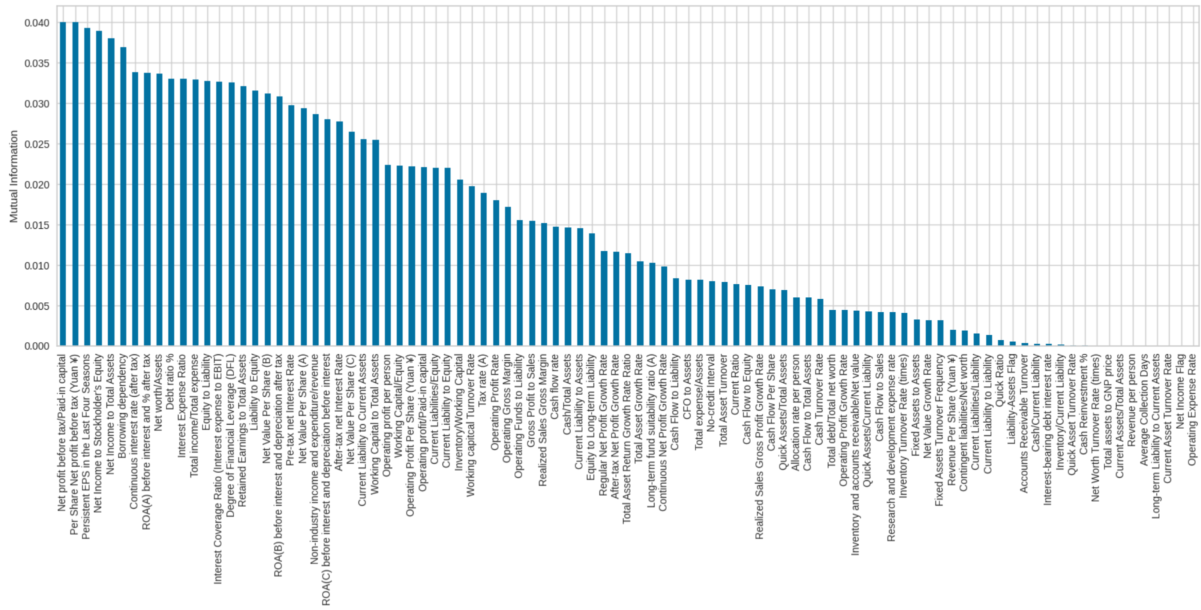
We decided to look deeper into the dataset, applying histograms, correlation analysis and boxplots. The most important finding was the fact that there are many correlations between the features that we must treat. We can identify those correlations with the red and blue colours in the heatmap. We will have to reduce the number of features to run the model and avoid noise in the dataset.

Correlation Heatmap



### 3. Feature Selection

Feature selection is essential to reduce model complexity and focus on relevant information. Correlation analysis and domain knowledge were leveraged to identify the most informative features. By reducing the feature set from 96 to 20, the models achieve better generalization and avoid overfitting. We got support from Sanjoy Modal on Kaggle, using mutual information analysis to the target (Bankrupt?).




So we found the 20 best features to analyze: 'ROA(A) before interest and % after tax', 'ROA(B) before interest and depreciation after tax', 'Non-industry income and expenditure/revenue', 'Continuous interest rate (after tax)', 'Net Value Per Share (B)', 'Persistent EPS in the Last Four Seasons', 'Per Share Net profit before tax (Yuan ¥)', 'Interest Expense Ratio', 'Debt ratio %', 'Net worth/Assets', 'Borrowing dependency', 'Net profit before tax/Paid-in capital', 'Retained Earnings to Total Assets', 'Total income/Total expense', 'Net Income to Total Assets', 'Net Income to Stockholder's Equity', 'Liability to Equity', 'Degree of Financial Leverage (DFL)', 'Interest Coverage Ratio (Interest expense to EBIT)' and 'Equity to Liability'.
After that, we apply SMOTE in the train and validation dataset to balance the data. From 97% of the "0" value in the target, we finished with 50%.

**4. Initial Neural Network Model**

Artificial neural networks consist of interconnected nodes arranged in layers. Each node applies a transformation to the input data and passes it to the next layer. The initial model architecture, a Multilayer Perceptron (MLP), is a classic neural network structure. Gradually increasing the number of epochs allows the model to iteratively adjust its weights and biases, improving its predictive performance.

We notice a lot of literature about the good number of hidden layers. We decide to test the reference model from 2 to 10 hidden layers. We found the best recall results on 4 and 6 hidden layers analysis, and we will comment further on this report. We also decided to randomly mix the number of nodes; according to Brownlee(2018), experimentation is a good way to fine-tune the best balance of layers and nodes.

Even considering that we made a feature selection moving from 96 to 20, we learned that LeakyRelu could provide a non-zero output for negative input values, which can help to avoid discarding potentially important information and thus perform better than ReLU in scenarios where the data has a lot of noise or outliers.

After we ran ten different models, we found improvement in Validation Recall after model 6 until model 10. We decided to pick model 9 as the best model, reaching 1.0 on Recall. The model 9 was set with Hidden Layers (512/256/128/64/32/16) and LeakyReLU activation.
We also decided to pick model 7, to see how a simpler Hidden Layers (64/128/256/64) configuration could perform in the next steps.

| model_name | Model 6 | Model 7 | Model 8 | Model 09 | Model 10 |
|---|---|---|---|---|---|
| train_loss | 0.236 | 0.17 | 0.293 | 0.147 | 0.149 |
| train_acc | 0.895 | 0.929 | 0.875 | 0.939 | 0.945 |
| train_recall | 0.922 | 0.935 | 0.896 | 1.0 | 0.96 |
| train_precision | 0.874 | 0.925 | 0.859 | 0.891 | 0.933 |
| val_loss | 0.248 | 0.188 | 0.3 | 0.176 | 0.165 |
| val_acc | 0.885 | 0.918 | 0.866 | 0.931 | 0.936 |
| val_recall | 0.914 | 0.926 | 0.896 | 1.0 | 0.957 |
| val_precision | 0.864 | 0.91 | 0.845 | 0.878 | 0.918 |
| F1 | 0.888297 | 0.91793 | 0.869753 | 0.935037 | 0.937094 |

**5. Model Optimization Techniques**

Two key techniques were employed to enhance the model 9 performance: early stopping and dropout regularization. Early stopping monitors the model's performance on validation data and stops training when the performance plateaus, preventing overfitting. Dropout regularization randomly drops out nodes during training, preventing the model from relying too heavily on any single node and enhancing its generalization capability.

We decided to use the best model for performance measures for these experiments. The model 9 reached a F1 score of 0.94, but the main factor was the Recall of 1.0, which was our objective in the project.

The dropout function helped to improve the accuracy of the validation dataset but was not effective in improving Recall. Dropout randomly deactivates a fraction of the neurons during each training iteration, forcing the network to learn robust representations by relying on different subsets of neurons. However, this random deactivation can result in the loss of valuable information, especially in the case of small or imbalanced classes. As Recall focuses on identifying true positives, the random dropout of neurons may remove crucial features or patterns related to the minority class, leading to decreased recall performance.

We also try the model, including early stopping. It helped to improve the accuracy of the model but was not effective in improving Recall. However, different from the dropout function, the change in Recall was lower, and that could be a good tool for further analysis of this model to find the best-balanced measures and not only the Recall.

## 6. Hyperparameter Tuning

Hyperparameters are settings that control the behaviour of neural network models. Through hyperparameter tuning, the best configuration is identified. Techniques such as random and grid search explore different combinations of hyperparameters to find the optimal values, leading to improved model performance.

We decided to apply a hyperparameter model to maximize Recall by finding the best number of nodes from the best model 9. The tuner found the best number of nodes: 508/246/122/56/32/10. The hyperparameter improved model 9 to a new model 14, just tuning the best nodes. F1-score grew to 0.98 from 0.94.

| model_name | Reference | Model 2 | Model 3 | Model 4 | Model 5 | Model 6 | Model 7 | Model 8 | Model 09 | Model 10 | Model 11 | Model 12 | Model 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| train_loss | 0.44 | 0.354 | 0.346 | 0.341 | 0.297 | 0.236 | 0.17 | 0.293 | 0.147 | 0.149 | 0.251 | 0.377 | 0.029 |
| train_acc | 0.846 | 0.85 | 0.848 | 0.856 | 0.873 | 0.895 | 0.929 | 0.875 | 0.939 | 0.945 | 0.892 | 0.829 | 0.991 |
| train_recall | 0.85 | 0.838 | 0.899 | 0.836 | 0.842 | 0.922 | 0.935 | 0.896 | 1.0 | 0.96 | 0.965 | 0.756 | 1.0 |
| train_precision | 0.857 | 0.859 | 0.815 | 0.871 | 0.899 | 0.874 | 0.925 | 0.859 | 0.891 | 0.933 | 0.842 | 0.885 | 0.981 |
| val_loss | 0.448 | 0.377 | 0.365 | 0.353 | 0.3 | 0.248 | 0.188 | 0.3 | 0.176 | 0.165 | 0.259 | 0.394 | 0.107 |
| val_acc | 0.834 | 0.842 | 0.844 | 0.851 | 0.869 | 0.885 | 0.918 | 0.866 | 0.931 | 0.936 | 0.889 | 0.825 | 0.98 |
| val_recall | 0.842 | 0.829 | 0.9 | 0.836 | 0.843 | 0.914 | 0.926 | 0.896 | 1.0 | 0.957 | 0.969 | 0.753 | 1.0 |
| val_precision | 0.852 | 0.852 | 0.809 | 0.861 | 0.889 | 0.864 | 0.91 | 0.845 | 0.878 | 0.918 | 0.836 | 0.881 | 0.962 |
| F1 | 0.84697 | 0.840343 | 0.852077 | 0.848316 | 0.865389 | 0.888297 | 0.91793 | 0.869753 | 0.935037 | 0.937094 | 0.8976 | 0.811987 | 0.980632 |

We decided to do the same exercise in the model 7, to a new model 16. Also showed good improvement performance, which we decided is on very important tool to improvel models.

## 7. Results and Discussion

The final model, utilizing the best hyperparameters, achieved the objective results in bankruptcy prediction. High accuracy, Recall, and precision indicate the model's effectiveness in identifying companies at risk of financial distress. Recall, in particular, is crucial for bankruptcy prediction as it measures the model's ability to correctly identify companies that are truly at risk, minimizing false negatives and the potential financial impact.

Finally, we applied the best model 14 in the test dataset. The model reached 0.96 precision, 0.93 recall, 0.95 F1-score. The results were appropriate, but further investigation needs to be done, especially in the positive bankruptcy cases.

```
Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.95      0.97      1987
           1       0.22      0.51      0.31        59

    accuracy                           0.93      2046
   macro avg       0.60      0.73      0.64      2046
weighted avg       0.96      0.93      0.95      2046

AUC-ROC: 0.8646882703675587
```

We also tested the prediction on model 16, which performed similar, but we decided to keep model 9 as the best model due to the objective of the project (recall results during the model training)

**8. Conclusion and Future Work**

In conclusion, this report highlights the importance of deep learning in machine learning models by using artificial neural networks, node and layer configurations, epochs, hyperparameter tuning, early stopping, and dropout regularization in bankruptcy prediction. The findings demonstrate the effectiveness of the proposed deep learning techniques in financial risk assessment. We could follow an evolution in the results when we apply more hidden layers and more nodes. We also noticed a good improvement using hyperparameter tunning. Future research can explore alternative architectures, incorporate additional data sources, and further investigate advanced deep-learning techniques to enhance the model's performance.

## References


Shapiro,  A H, Sudhof, M, Wilson D (2020). "Measuring News Sentiment." Federal Reserve Bank of San Francisco Working Paper 2017-01

Liang, D., Lu, C.-C., Tsai, C.-F., and Shih, G.-A., 2016. “Financial Ratios and Corporate Governance Indicators in Bankruptcy Prediction: A Comprehensive Study. European Journal of Operational Research”, vol. 252, no. 2, pp. 561-572.

Srikari Rallabandi, 2023. "Activation functions: ReLU vs Leaky ReLU", Article available at location: https://medium.com/mlearning-ai

*Brownlee* J., 2019. “How to Configure the Number of Layers and Nodes in a Neural Network”. Article available at location: https://machinelearningmastery.com/how-to-configure-the-number-of-layers-and-nodes-in-a-neural-network/

## Appendix

| model_name | Reference | Model 2 | Model 3 | Model 4 | Model 5 | Model 6 | Model 7 | Model 8 | Model 09 | Model 10 | Model 11 | Model 12 | Model 14 | Model 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| train_loss | 0.44 | 0.354 | 0.346 | 0.341 | 0.297 | 0.236 | 0.17 | 0.293 | 0.147 | 0.149 | 0.251 | 0.377 | 0.029 | 0.034 |
| train_acc | 0.846 | 0.85 | 0.848 | 0.856 | 0.873 | 0.895 | 0.929 | 0.875 | 0.939 | 0.945 | 0.892 | 0.829 | 0.991 | 0.992 |
| train_recall | 0.85 | 0.838 | 0.899 | 0.836 | 0.842 | 0.922 | 0.935 | 0.896 | 1.0 | 0.96 | 0.965 | 0.756 | 1.0 | 1.0 |
| train_precision | 0.857 | 0.859 | 0.815 | 0.871 | 0.899 | 0.874 | 0.925 | 0.859 | 0.891 | 0.933 | 0.842 | 0.885 | 0.981 | 0.984 |
| val_loss | 0.448 | 0.377 | 0.365 | 0.353 | 0.3 | 0.248 | 0.188 | 0.3 | 0.176 | 0.165 | 0.259 | 0.394 | 0.107 | 0.068 |
| val_acc | 0.834 | 0.842 | 0.844 | 0.851 | 0.869 | 0.885 | 0.918 | 0.866 | 0.931 | 0.936 | 0.889 | 0.825 | 0.98 | 0.986 |
| val_recall | 0.842 | 0.829 | 0.9 | 0.836 | 0.843 | 0.914 | 0.926 | 0.896 | 1.0 | 0.957 | 0.969 | 0.753 | 1.0 | 1.0 |
| val_precision | 0.852 | 0.852 | 0.809 | 0.861 | 0.889 | 0.864 | 0.91 | 0.845 | 0.878 | 0.918 | 0.836 | 0.881 | 0.962 | 0.973 |
| F1 | 0.84697 | 0.840343 | 0.852077 | 0.848316 | 0.865389 | 0.888297 | 0.91793 | 0.869753 | 0.935037 | 0.937094 | 0.8976 | 0.811987 | 0.980632 | 0.986315 |