BA 885 Advanced Analytics II - Project Summary

Team Members: Yulong Gong, Scott McCoy, Antonio Moral, Yichi Zhang

Introduction

Dataset

This dataset comes from several years of IRS non-profit tax returns. This is a large dataset of 230,000 organizations, 240 tax return features, and complete data for five tax years. There are a number of informational categorical variables and hundreds of continuous financial figures like revenues, expenses, assets and liabilities. We thought this would be a good dataset for a neural network because of the size and complexity that we hoped to describe with a complex model.

Dataset Links:

-All Year CSV files from Kaggle - https://www.kaggle.com/irs/irs-990?select=irs_990_2017

-IRS Schedule N and Instructions - https://www.irs.gov/pub/irs-pdf/f990sn.pdf

-Data dictionary files from IRS -
https://www.irs.gov/statistics/soi-tax-stats-annual-extract-of-tax-exempt-organization-financial-data

Problem

One important feature in our dataset is a pair of categorical variables for partial and complete terminations. We decided to build a neural network that would use tax return figures to predict whether an organization will discontinue all or part of its operations in the future.

Definition of termination and partial liquidation per IRS instructions:

- Termination - “Liquidation, Termination, or Dissolution”
- Partial Liquidation - “Sale, Exchange, Disposition, or Other Transfer of More Than 25% of the Organization’s Assets”

Our analysis could help organizations determine the risk that a nonprofit stakeholder would cease to exist or substantially change it’s structure in the near future. These dissolution events are very rare in a given year, so being able to effectively identify the positive class will be important for our model.

Pre-Processing

Some preprocessing is needed to get this data into a machine learning usable format. We start with all observations having a 2013 year end, and merge all columns with 2012 observations giving us 2 years worth of data as features. Next we convert categorical columns to numeric, and create dummy indicators for categorical features with more than 2 possibilities. This results in a feature space of 542 features, with all 2012 column labels having a suffix of “t-1” to indicate a prior year value.

To create the labels, we create a list of companies that had a termination, partial liquidation, or either between 2014 and 2016. Next, we create 3 indicator variables that are 0 for companies not present in the lists and 1 for companies that are. Of the 266,000 observations, 1.35% had either a termination or partial liquidation in the subsequent 3 years, giving us an imbalanced classification problem.

Methodology

To predict terminations we use a feed-forward neural network. With the size and complexity of our input data, we tried many different combinations of layers of various sizes to find the architecture that best predicts our target. We also try various combinations of hyperparameters like different loss functions, optimization methods, number of epochs, training loops, or batch sizes.

Resampling

This dataset is extremely imbalanced. There are 262,779 records that have no termination or liquidation, and only 3608 records have termination or liquidation. Only about 1.4% of the target variable is 1 and the left 98.6% are all 0. In order to have predictive power for those 1.4% of 1s, we decided to use the resampling method that can help adjust the number of majority and minority classes on our dataset to make it relatively balanced.

The downsampling method reduces the count of training samples falling under the majority class. It keeps all the 1s in the training set, and randomly selects the same amount of 0s so that the downsampled dataset would be balanced. However, during this process, we would lose lots of valuable information. We used the downsampling method at the beginning to build the neural network, because it takes less time to train and gives reasonable results.

The upsampling method generates data points corresponding to minority class and injects into the dataset, so that the counts of both labels are almost the same. Using the upsampling would make sure that we have enough data to train on, but it would also take longer time because it basically doubles the dataset in our case. We used the upsampling method for further neural network training and hyperparameter tuning. The upsampled dataset generally gives better results than the downsampled dataset.

Models

Baseline Model

- **Multi-class**

There are no original targets in the datasets, and we choose to predict the business status, which are normal(262779), termination(1232), liquidation(2217) and both termination and liquidation(159). So the original task for us is a multiclass classification problem. And we decide to pick one from the tree based model, random forest, and one from the regression which is logistic regression. With the default setting, for the random forest model, the accuracy reaches to 0.78 with 1 the area under the ROC Curve (AUC) for training and 0.67 test AUC. For the logistic regression, the accuracy is 0.5 with 0.85 training AUC and 0.64 test AUC. To evaluate the model performance on an imbalanced data, accuracy is not the right matrix to go. But for AUC, the difference between training set and test set suggests overfitting. So we will try to develop a classification neural network to see whether it would have better performance in terms of AUC.

- **Binary**
  Furthermore, we could also consider this problem to be a binary classification problem, and in this case, the target would be normal and termination or liquidation. For the same rationale, we choose random forest and logistic regression. For the regression model, we choose logistic regression. With the default setting, the accuracy reached 0.68 and the AUC is 0.64 on the test set. For the tree based model, we choose the random forest from the imblearn package which is a custom sklearn API specifically built for imbalanced datasets. On the test set, this model performs well with 0.71 accuracy and 0.66 AUC. Those two baselines give similar results. The accuracy would be around 0.7, and the AUC would be around 0.65.And we will also develop a neural network to check whether it improves the performance.

Multi-class Neural Network

For the muli-class neural network, we decide to start with a simple structure. To be specific, we choose a one layer fully connected neural network with 100 units and ReLU as the activation function. For the output, we decided to let the output be the probability of each class, so there is also a dense layer with 4 units and softmax as the activation function in the end. To properly evaluate the model, we choose to compile the model in a way that the optimizer is Adam, the loss function is categorical cross entropy. In order to have sufficient evaluation matrix, auc, precision, recall are applied.

Based on the results we see from the baseline model, there are potential overfitting issues. To prevent that from happening, we add a drop out layer with 0.5 probability. We also utilized the early stopping methodology by monitoring the validation set AUC. By applying these two methods, we are expected to see the degree of overfitting is reduced.

For the training parameters, we choose the learning rate to be 0.001, with 500 epochs and 3000 as batch size. Then we choose to train the model with the resample dataset, but test it on the original test. The training process ends with only 22 epochs with 0.78 training auc, 0.85 validation auc. The high validation auc indicates that there are more constraints on the training set and the validation set is simpler compared to the training set.

And when put into evaluation mode, we got 0.86 auc in the test set.

Even though the model evaluation matrix is appealing, there are still issues that need to be fixed. Based on the loss plot shown in Figure 1, the loss curve tends to converge around 18 epochs. Also, as is shown in the confusion matrix plot, all predictions for 0,1,2 tend to skew towards 3. In other words, the model tries to predict all observations into 3, it is obvious in the 0 and 1. Even for 3 the model mislabeled around 25% of the data, which indicates that even if the model gets a good auc, it fails to capture the characteristics of the dataset. So we would try a different model.

Figure 1. Loss vs epochs

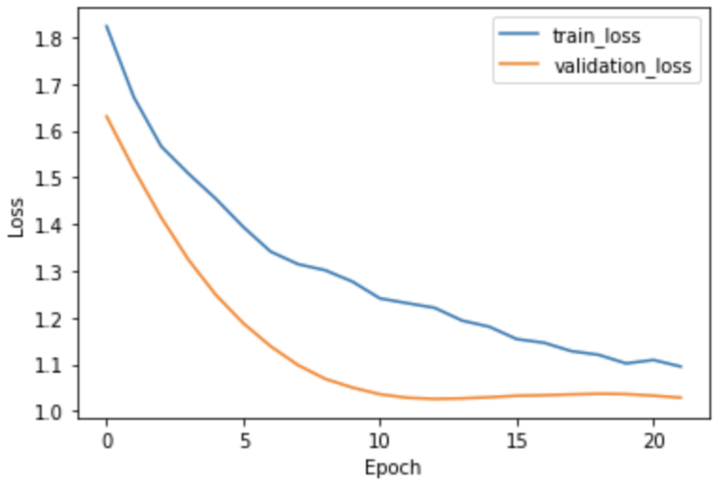
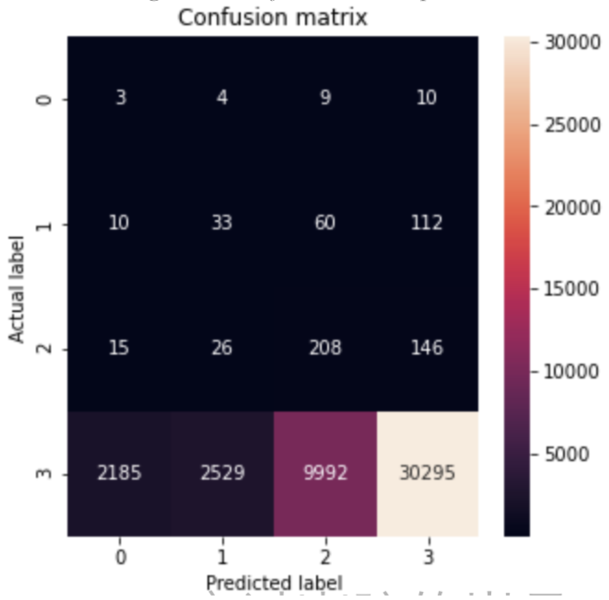


Figure 2. Confusion matrix plot



	0	1	2	3
0	3	4	9	10
1	10	33	60	112
2	15	26	208	146
3	2185	2529	9992	30295

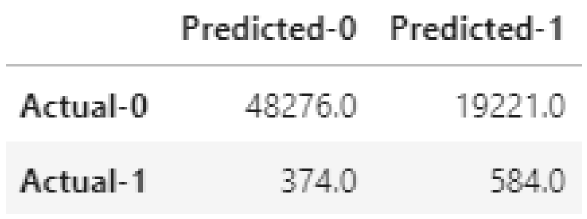
Binary Neural Network (Full-sample)

For the full sample, fully connected network predicting full or partial terminations we applied numerous architectures to try to optimize predictive power. See supplementary tables for full results of various network structures and hyperparameters.

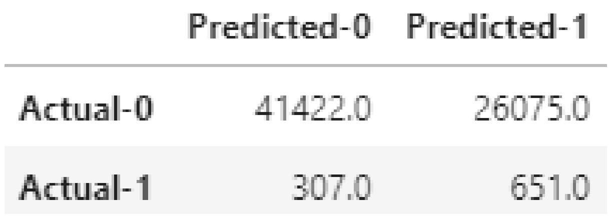
None of these models significantly outperformed the baseline balanced random forest classifier in terms of AUC. However, our models were able to identify more instances of the positive class by tinkering with the class weights when training the model. Passing class weights to the model's .fit() method will cause the model to further emphasize examples from the under-represented class when training. Our dataset had around 70 negative class observations for every one positive class observation, meaning that a class weight of 1-70 would much better balance the number of positive predictions made.

One of the best performing models had an architecture of 5 hidden layers (100-15-20-15-10), used the relu activation function at each layer, had a weight balance of 1-100, and had two separate normalization layers: a 50% dropout layer and an l1 regularization layer. Though the AUC of 0.65 was slightly lower than the baseline random forest score of 0.66, this model found a much higher percentage of the positive class (recall rate) without significantly reducing the precision score.

Random Forest Baseline: Neural Network:



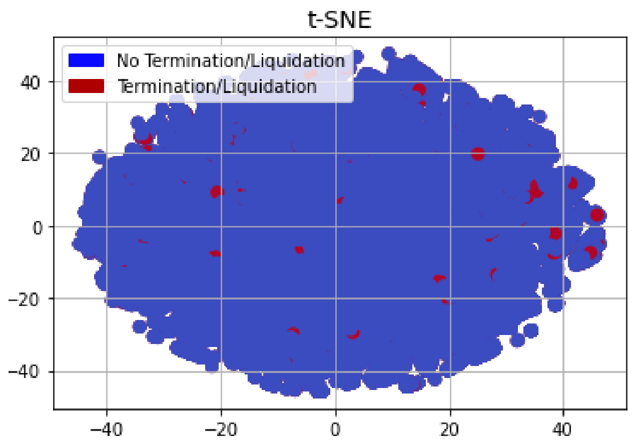
	Predicted-0	Predicted-1
Actual-0	48276.0	19221.0
Actual-1	374.0	584.0

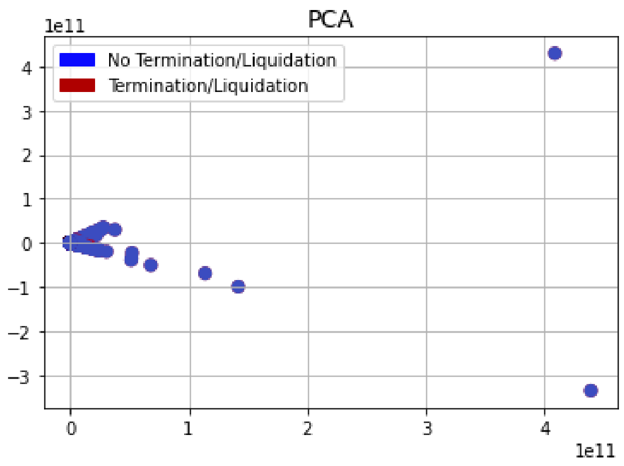


	Predicted-0	Predicted-1
Actual-0	41422.0	26075.0
Actual-1	307.0	651.0

Conclusions and Findings

From our selection of models, our best results came from our 5-hidden layer neural network using the Relu activation function, with an AUC of a little over 0.64 and a recall rate of 0.68. These results are satisfactory given the nature of the data used in the analysis. As mentioned above, our dataset is extremely imbalanced, and thus, developing a classification model becomes a difficult task. This low presence of positive instances in the dataset makes it so the most reliable way for a model to achieve a high accuracy is to simply classify all data points as 0, or in our case as companies that were not terminated or liquidated in the studied timeframe. This model would then show ~98% accuracy, but would be completely useless for deployment as it cannot identify positive instances. Because of the difficulties our initial models had when making predictions, our team graphed the data points based on their primary features, and the findings show why we were obtaining inaccurate results as there is no evident linear separability between both classes, as can be seen below.

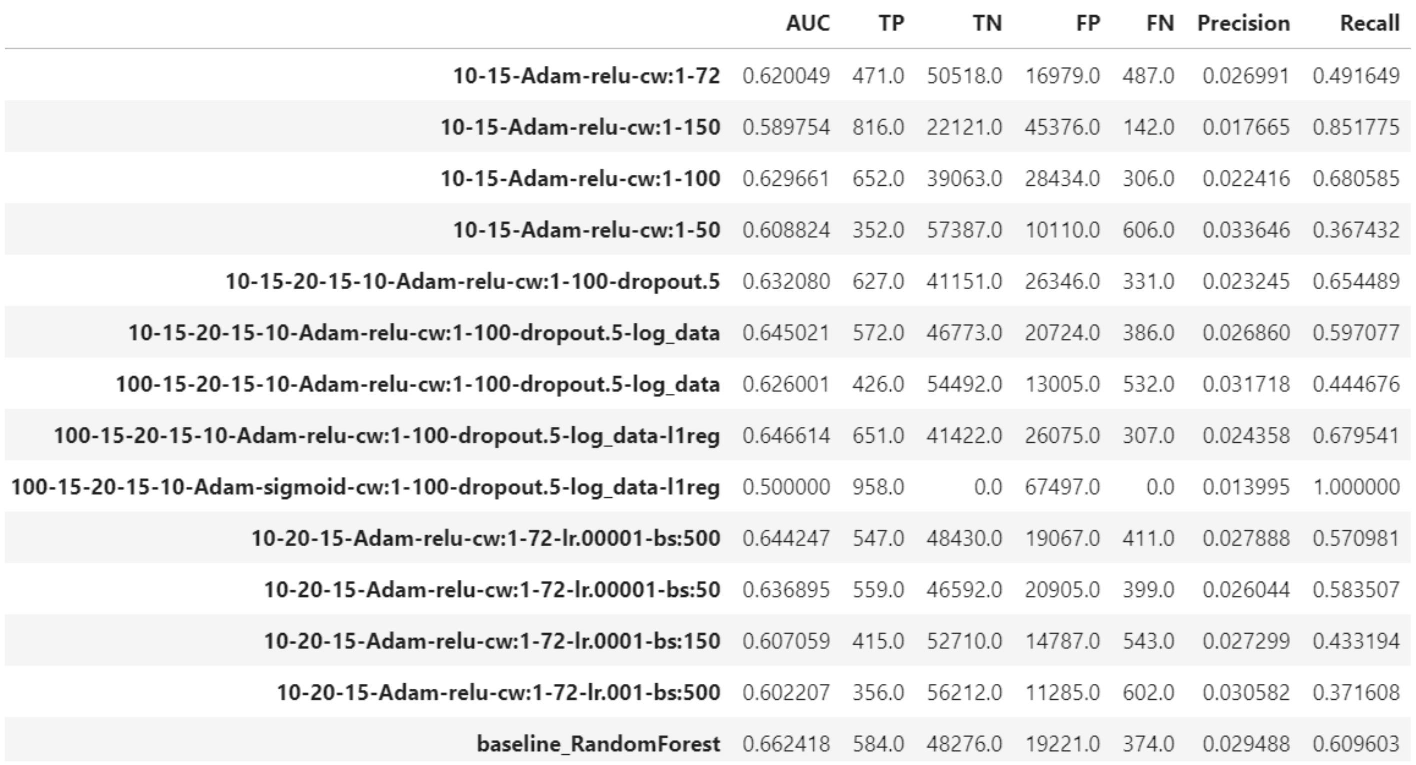




To address this issue, our team resorted to data normalization techniques such as upsampling to even out the sample classes in the data, as well as focusing on the AUC and Recall rate metrics as they consider the True Positive and the True Negative rates when assessing a models' performance. By normalizing the data, all the models performed better than the first iterations using unchanged data as the differences in magnitude between the variables was no longer heavily affecting the predictions. Alongside this preprocessing, upsampling the data provided the best results in our project. Upsampling is an effective technique as it increases the number of positive instances, to create a more balanced dataset that is more useful for training. Although successful, there are ways to improve the project in the future. Firstly, having more data, particularly for companies that were terminated, would be ideal to further improve the model as it still struggles to classify around 60% of the true positives. Secondly, more testing of different architectures could potentially result in better performing models that we were not able to test for the current version of this project. Finally, further feature engineering could aid with training and predictions by creating more meaningful attributes for the data and thus can be more easily captured by the models developed.

Appendix

Results table for fully-connected, full-sample binary models:



	AUC	TP	TN	FP	FN	Precision	Recall
10-15-Adam-relu-cw:1-72	0.620049	471.0	50518.0	16979.0	487.0	0.026991	0.491649
10-15-Adam-relu-cw:1-150	0.589754	816.0	22121.0	45376.0	142.0	0.017665	0.851775
10-15-Adam-relu-cw:1-100	0.629661	652.0	39063.0	28434.0	306.0	0.022416	0.680585
10-15-Adam-relu-cw:1-50	0.608824	352.0	57387.0	10110.0	606.0	0.033646	0.367432
10-15-20-15-10-Adam-relu-cw:1-100-dropout.5	0.632080	627.0	41151.0	26346.0	331.0	0.023245	0.654489
10-15-20-15-10-Adam-relu-cw:1-100-dropout.5-log_data	0.645021	572.0	46773.0	20724.0	386.0	0.026860	0.597077
100-15-20-15-10-Adam-relu-cw:1-100-dropout.5-log_data	0.626001	426.0	54492.0	13005.0	532.0	0.031718	0.444676
100-15-20-15-10-Adam-relu-cw:1-100-dropout.5-log_data-l1reg	0.646614	651.0	41422.0	26075.0	307.0	0.024358	0.679541
100-15-20-15-10-Adam-sigmoid-cw:1-100-dropout.5-log_data-l1reg	0.500000	958.0	0.0	67497.0	0.0	0.013995	1.000000
10-20-15-Adam-relu-cw:1-72-lr.00001-bs:500	0.644247	547.0	48430.0	19067.0	411.0	0.027888	0.570981
10-20-15-Adam-relu-cw:1-72-lr.00001-bs:50	0.636895	559.0	46592.0	20905.0	399.0	0.026044	0.583507
10-20-15-Adam-relu-cw:1-72-lr.0001-bs:150	0.607059	415.0	52710.0	14787.0	543.0	0.027299	0.433194
10-20-15-Adam-relu-cw:1-72-lr.001-bs:500	0.602207	356.0	56212.0	11285.0	602.0	0.030582	0.371608
baseline_RandomForest	0.662418	584.0	48276.0	19221.0	374.0	0.029488	0.609603

Github link:
https://github.com/AntonioMoralCevallos/Predicting-Non-profit-Terminations-and-Liquidations---BA-885