Bankruptcy Prediction

Braden Anderson, Hien Lam

October 17, 2022

I Introduction

Bankruptcy prediction is an essential use case in the finance sector notably for creditors, investors, and policy makers. It entails accurately discerning a company's risk of bankruptcy in order to diagnose the underlying root cause, prepare contingency plans, or execute corrective action. In this case study, we classified whether a company was bankrupt using XGBoost and LightGBM machine learning techniques then ascertained their predictive performance with a slew of metrics e.g., F1, AUC, accuracy, precision, and recall. Lastly, we compared our best performing model to those conducted by [1].

II Methods

II.a Data Preprocessing

The bankruptcy dataset consisted of 43,405 observations and 64 synthetic attributes derived from a company's financial records for one of five fiscal years. The exact method for deriving these attributes is described in [1]. The binary target represented the company's bankruptcy status at the end of a fixed five-year period. For example, the label associated with observations in year one denoted the company's bankruptcy status five years later, while the label associated with observations in year two signified the company's status four years later. The 43,405 instances in the data set are comprised of 7027, 10173, 10503, 9792, and 5910 observations from years one through five, respectively.

All 64 features in the dataset contained at least one missing value, with the largest number of missing values for a single feature being 17,049. The task of selecting an imputation strategy was incorporated into the baseline modeling phase, where a total of eight different imputation methods (mean, median, KNN-5, KNN-20, KNN-30, KNN-40, KNN-100, and KNN-150) were applied to each of three different models (Random Forest, LightGBM and Xgboost), which were evaluated with 10-fold cross-validation. The relative performance of these 24 models will be discussed further in the baseline modeling section.

Other than the handling of missing values discussed previously, these data required no additional preprocessing due to the extensive work performed in [1] when assembling the dataset. The nature of the data, which consisted of synthetic attributes that were created by randomly combining economic indicators with arithmetical operations also meant there was little exploratory data analysis to conduct. Since the goal of this work was to utilize the data collected in [1], and in general, such randomly generated features do not lend themselves to meaningful insights in the context of this problem, an in-depth discussion on exploratory data analysis will be excluded here.

II.b Gradient Boosted Trees

Boosting is an ensemble method that combines several weak learners into one strong learner. The weak learners are trained sequentially and each one works to correct the errors of their predecessor. XGBoost [4] and LightGBM [3], the methodologies conducted in this case study, utilized *gradient* boosting which trains each weak learner on the residuals of the tree that came before it. The boosting approach allows the algorithm to learn slowly, which alleviates chances of overfit. Another benefit is that it allows us to use any differentiable loss function. This means we can choose a specific loss function that is geared towards our task, such as a robust loss function that helps make the model more resistant to outliers. Some intuition for why boosting may be a good idea is provided next. Typically, the weak learners that we combine via boosting are very shallow decision trees. Since these trees are shallow, they do not have the capacity to overfit the data – in other words, each weak learner typically has high bias and low variance. Boosting allows us to decrease bias, and hopefully achieve a strong learner with both low bias and low variance (the slow learning part is key to keeping the variance low).

LightGBM builds the tree horizontally (grows leaf-wise) while XGBoost operates vertically (grows level-wise. Both algorithms can implement DART (Dropout Additive Regression Trees) booster which drops trees as another way to curb overfit.

II.c Establishing a Baseline

Prior to any model building, the 43405 observations were split into cross-validation and final test sets containing 90% and 10% of the data, respectively. We conducted baseline classifier models of LightGBM and XGBoost using its default parameters on eight different imputation strategies, the best two of which were shown below along with a dummy model that predicted using zero information other than the most frequent class. Please refer to **Table 1** to observe the 10-fold cross validation results from the best performing model within each algorithm. The dummy model achieved 95% accuracy which aligned with our intuition given the class imbalance of the data thus we focused on F1 and AUC going forward as appropriate metrics for model consideration. LightGBM with mean imputation strategy performed the best out of all the baseline models in terms of train time while maintaining equivalent metric production.

Model	Imputation Strategy	Average Validation F1-Score	Average Validation Accuracy (%)	Average Validation AUC	Precision	Recall	Average Fit Time (s)
XGBClassifier	Mean	0.7369	97.92	0.9742	0.9448	0.6052	8.495
XGBClassifier	Median	0.7368	97.92	0.9752	0.9461	0.6041	10.429
LGBMClassifier	Mean	0.7354	97.93	0.9737	0.9611	0.5962	1.604
LGBMClassifier	Median	0.7217	97.84	0.9744	0.9550	0.5808	0.904
Null (No Information)	--	--	95.18	--	--	--	--

Table 1: Baseline Models

II.d Modeling

With the mean imputer and LightGBM classifier pipeline as the baseline, we began the hyperparameter tuning process by constructing a broad search space across candidate values for

eight different hyperparameters. Since the size of the search space grows combinatorically with the number of candidate values, the parameter grid was constructed such that beneficial deviations from the default values could be detected while keeping computational cost at an acceptable level. For the training instances sampling ratio and the column subsampling ratio, we considered two candidate values of 0.95 and the default value of 1.0. The maximum base learner depth is unlimited by default, which was included in the search space along with a maximum depth of 10. The maximum number of tree leaves per base learner included candidate values of 30, 31, and 32, which is the default value and ± 1. This combination of hyperparameters was evaluated for two different boosting types ("gbdt" [2], and "dart" [3]), three different numbers of base estimators (100, 750, and 1000), two different class weighting strategies (balanced, None), and 25 different learning rates evenly spaced between 0.00199 and 0.5011 on a base 10 logarithmic scale. This initial search therefore considered $2 * 2 * 2 * 2 * 2 * 3 * 3 * 25 = 7200$ candidate models, which were each evaluated using 10-fold cross validation and ranked according to their mean validation F1-Score. The best performing model from this initial search selected the default hyperparameter values for the subsample ratio, column sample ratio, maximum base learner depth and maximum number of base learner leaves, but deviated from the defaults by selecting 1000 base estimators, the "dart" boosting type, a balanced weighting strategy, and a learning rate of 0.5011, achieving an average validation F1-Score of 0.7972, a 0.062 improvement over the baseline model with default hyperparameters.

Noting that both the learning rate and number of base estimators selected by the initial search were the largest values in the search space, a second search was performed to consider additional candidate values. The next search evaluated model sizes of 1000, 1300, 1500, 2000, and 2500 at 25 different learning rates between 0.25 and 1.0, from which 1300 base estimators and learning rate of 0.6683 was selected.

Due to the importance of the learning rate hyperparameter, one final search was performed which held fixed all previously identified hyperparameters and evaluated the model at 501 different learning rates. The first 500 values were evenly spaced on a base 10 logarithmic scale between 0.5012 and 0.8354, which is ± 25% around the best identified value of 0.6683, and the 501st value was the best-known value of 0.6683. Somewhat surprisingly, the best of all 501 candidate learning rate values was the previously identified best value of 0.6683, where the model achieved an average validation F1-Score of 0.799391.

III Results

III.a Final Test Set Performance

The models from [1] and their mean metrics were reflected in **Table 2**. Our best model was LightGBM with the following non-default hyperparameters listed below:

- DART booster
- Number of base estimators (n_estimators): 1300
- Learning rate: 0.6683439
- Class weighting: balanced

We noted that the comparison between our best performing model and that of [1] was a bit misleading since the latter's models were different every year. Nonetheless, our model produced 0.05 mean AUC improvement and substantially smaller standard deviation while being more equivalent across each fold. We surmised the consistency was due to the regularization DART booster.

Source	Model	Mean AUC	STD AUC
Anderson & Lam	LightGBM	0.979	0.006753
[1]	Random Forest	0.854	.0330
[1]	XGB	0.934	.0266

Table 2: Model Comparison

Figure 1 called to attention the impressive 0.9997 F1 and 99.9974% accuracy produced from the training data. The metrics on the 10% hold out test set produced a slightly smaller F1 and accuracy. The false negative error i.e., inaccurately predicting a company as not bankrupt when they were is more concerning than the contrary (false positive). In the latter situation, perhaps the company and its stakeholders are called to be more proactive to alleviate the effects of bankruptcy and believed they succeeded. However, the former situation is detrimental as the affected parties could presumed that the company was safe and proceeded with little preparation to avoid bankruptcy. We noted the model's potential room for improvement given its 60/4331 misclassification of companies as not bankrupt when they were (false negative).

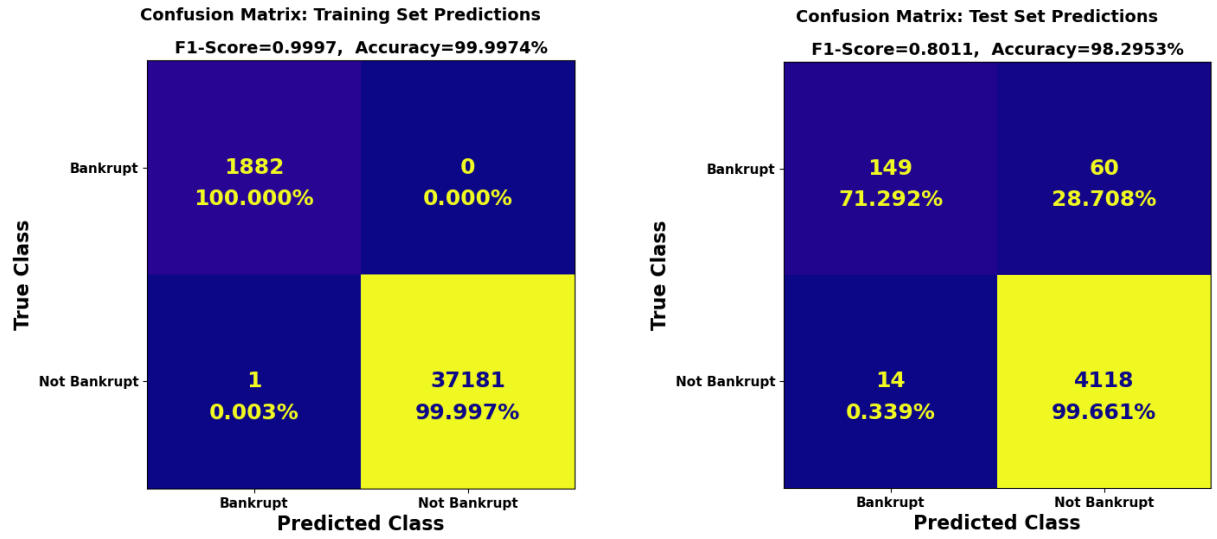

Figure 1: Final Model Train and Test Set Confusion Matrices

IV Conclusion

The LightGBM grid search produced the best model with a concession of 116s mean train time. Given the critical economic and political implication to accurately predict bankruptcy, we believe

this heightened time was well worth the tradeoff considering the observed performance increase. As previously mentioned, the risk of misclassifying cases of bankruptcy far outweighed the negative impact caused by predicting that a successful company will become insolvent. This desire to ensure cases of bankruptcy were not misclassified further highlighted the benefit of the Anderson & Lam model proposed above, which correctly identified 71.29% of bankruptcy instances in the test set, a substantial improvement over the base LightGBM and XGBoost models which correctly classify only 57.89% and 60.20% of bankrupt companies respectively. Finally, we have shown that our model outperformed the one proposed in [1] across all validation metrics, which we attributed to a more comprehensive hyperparameter tuning process and increased training data. Our improvement over [1] suggested that the task of bankruptcy prediction may be better accomplished by fitting a single estimator to all available data, rather than creating specialized models for each year's financial records.

V References

[1] [MZSTJT].pdf (pwr.edu.pl)

[2] LightGBM: A Highly Efficient Gradient Boosting Decision Tree (neurips.cc)

[3] [1505.01866] DART: Dropouts meet Multiple Additive Regression Trees (arxiv.org)

[4] [1603.02754] XGBoost: A Scalable Tree Boosting System (arxiv.org)

VI Appendix

VI.a Code

Please refer to the attached python source code and python notebook.