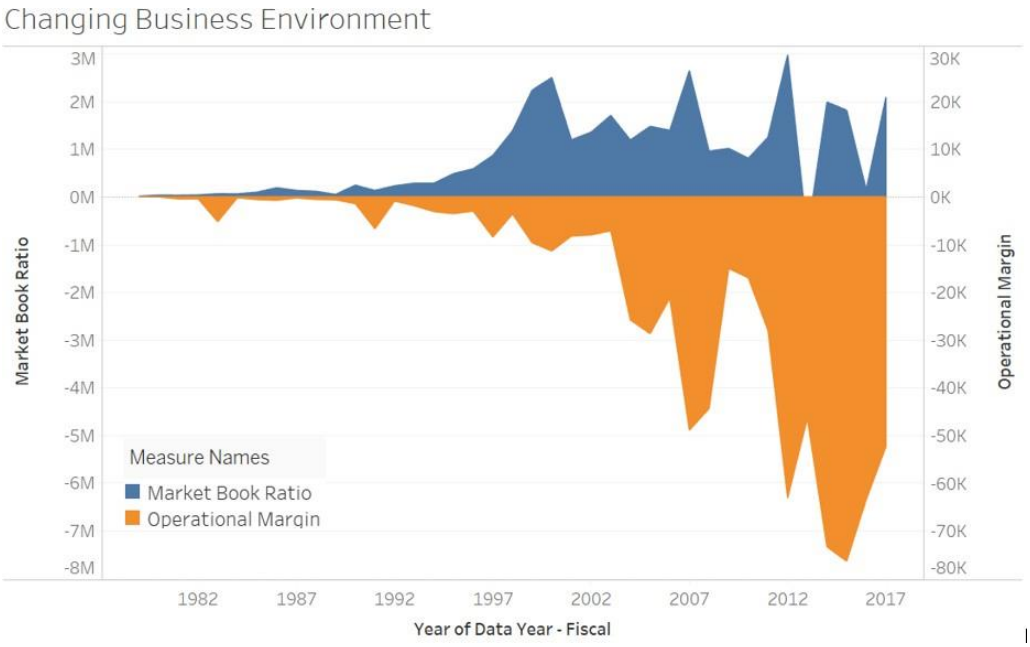
PROJECT REPORT

Introduction

If we think of our dataset in the context of a lending scenario, the data represents snapshots of companies at one-year intervals. Each snapshot provides a different perspective on whether the company is showing signs of bankruptcy. As a lending company, each measurement point would result in a different decision on whether to provide a loan or not based on the company ratios at that point in time. With several different views of individual companies over the years, our assumption regarding the data is the following: each line item represents like a separate company between the years 1979 to 2017 because every year a company's financial health can change.

In the article, 'Machine learning models and bankruptcy prediction' (Barboza, Kimura & Altman, 2017), they selected the same number of solvent and insolvent firms which was a balanced set. Although we were working with a different dataset, the main assumptions we made were in line with the article, that each line item represented a different company. Certain information was not available to us such as individual company IDs, so we moved forward under this assumption.

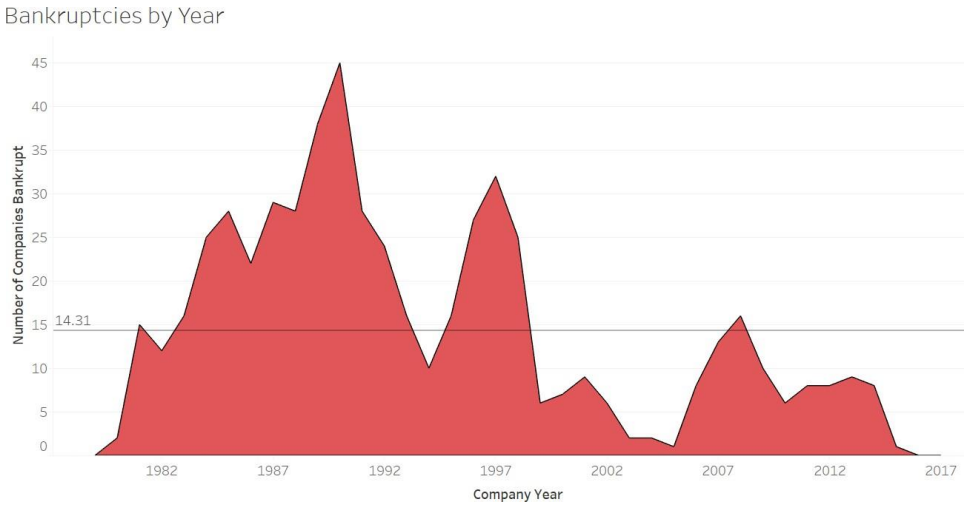
Ratios over a certain timeframe are solid indicators of financial stability but they are not completely reliable. As our initial data exploration showed below, the business ecosystem is changing – with the emerging competitive landscape, companies are finding ways to cut costs and squeeze margins to fend off the competition. Companies are taking on more risk which we see reflected in the market book ratio. We start to see more extreme numbers from 2002 onwards as it becomes a riskier business environment to operate within (Appendix 1).



It is more important now than ever to develop strong tools for accurately evaluating risk in lending and deploying machine learning algorithms to help drive business decisions. The sections below will explore our approach to building a model to solve this address this issue as well as detail our findings.

Data Preprocessing & Exploration

In an initial exploration, our data showed that in 1990 and 1997, and 2008 (financial crisis) there were significant jumps in the number of companies that went bankrupt. Several companies that went bankrupt in 2008 had used large amounts of financial leverage and ultimately succumbed under the burden of debt. The previous peaks of 1990 and 1997 were much higher than the yearly average of 14 bankruptcies in the data (low number of records considering we had 92,872 observations). Although this is difficult to predict, the goal of our model is to build a tool to detect the signs of bankruptcy before it occurs.



After some preliminary data visualization, we then profiled the data in Python, which showed we had 15 variables, 92,872 observations, 27 duplicate rows and 27,305 missing observations to work with. Based on this, we did some data preprocessing to increase the future accuracy of our model as several of the variables were highly skewed. We removed the duplicate rows and created a data frame to separately handle the missing values across all variables. We grouped the variables by year as well as whether the company went bankrupt or not and imputed the median value where the data was missing. We chose the median value since the average was skewed by the higher values, which would inherently lead to a bias in our data. After this was completed, we merged all entries back into the original table.

After this step, the variables ‘Assets Growth’, ‘Sales Growth’, and ‘Employee Growth’ had 131 missing values remaining. These values could potentially represent companies that entered the market in that particular year, however, without this confirmed information we determined that they were a small number of variables and decided to remove these rows. Based on our initial assumptions, these missing cells did not represent anything meaningful and were missing completely at random (no pattern of missingness). Since this is the easiest type of missing data to deal with and we can say with confidence that deleting this type of observation does not add bias to the model, we removed them (131/92K+ remaining observations had minimal effect on the overall data). We also dropped the fiscal year variable since it did not provide any information to the classifier with regards to the distinction between classes. From this point we had 92,156 companies that did not go bankrupt and 555 that did.

We then cross referenced a correlation matrix where the maximum was 0.58, so there was no need to drop any variables since there was minimal risk of co-linearity in the model. Since most of the variables were skewed, we used StandardScaler to scale the X variables (not Y – removed

Bankruptcy), which transformed the data so that the distribution had a mean value of 0 and a standard deviation of 1. Then we merged the final set together which was ready to be used for modelling.

Modelling

We split up our modelling into four different approaches: random undersampling (Appendix 2), random oversampling (Appendix 3), random oversampling using the Synthetic Minority Over-Sampling Technique (SMOTE) (Appendix 4), and a combination of undersampling & oversampling using SMOTE and Edited Nearest Neighbours (SMOTE ENN) (Appendix 5). The models we built for each technique include: Logistic Regression, Naïve Bayes, Random Forest, and Support Vector Machines. For each model, we split the data into 80% for training, and 20% for testing. For model validation, we used 10-fold cross validation on the training data to judge how the models performed outside the sample. Each Appendix shows a table summary of individual class accuracies for each (before & after tuning), as well as the optimal parameters used (Appendix 6).

Random Undersampling

We split the cleaned data into train and test with an 80/20 ratio and used the training set to build our models. The training set had 73,725 records for non-bankrupt companies and only 443 records for bankrupt companies, which made it highly imbalanced. The model would learn limited information about bankrupt companies, so we used resampling techniques to balance the dataset. We performed all the resampling steps on the training set, without altering the test set.

Our first step was to use the cleansed data and apply random undersampling, and then run it through multiple models to see which one provided us with the most accurate results. Random undersampling involves randomly selecting records from the majority class (which is the non-bankrupt companies) to delete from the training dataset and reduce the observations until the number of examples was equal for each class. This brought the number of observations from 73,725 for non-bankrupt companies and 443 bankrupt companies to 886 in each. This technique ensured our model did not primarily learn from the companies who did not go bankrupt and that it had equal information to learn about companies that did go bankrupt, as this is the goal of our model.

After building the first logistic regression model, the results improved after tuning, with 70 observations with true positives, meaning the model correctly predicted 70 companies that did go bankrupt, and 13,742 true negatives for the correctly predicted companies that did not go bankrupt, for an AUC of 0.72 (remaining values shown in Appendix 2). We went through this process for the rest of the models, which gave us an AUC of 0.69 for Naïve Bayes, 0.93 for Random Forest, and 0.67 for SVM. Random Forest generated the best results, which were 100 true positive values and 15,473 true negative values (before tuning), and 102 true positives and 15,430 true negatives after tuning.

Since the number of true positives (predicted bankrupt companies) was still low, we tried alternative methods below. However, the results stayed consistent throughout the different models in that the hyper-parameter tuned models were generally higher in AUC – except for Naïve Bayes which was lower before tuning, and Random Forest was 0.01 higher in AUC before tuning. Recall for this model was 0.91 for non-bankrupt companies and 0.84 for non-bankrupt companies for the model before tuning (102 true positives, 15,430 true negatives). This was our overall highest recall for bankrupt companies, which was a promising option to meet our end goal.

Random Oversampling

With random oversampling, the algorithm created multiple copies of the same entries and increased the number of observations until it equalized. This potentially introduces overfitting (AUC = 1) because you are copying the data multiple times and creating synthetic results. This process brought the number of observations from 73,725 for non-bankrupt companies and 443 bankrupt companies to 147,450 in each. However, the recall scores indicate that there is overfitting using this technique as recall was 1.00 for non-bankrupt companies. This approach is therefore not recommended, since 100% of bankrupt companies were correctly predicted, and out of the companies that did go bankrupt, only 2% were predicted accurately.

For oversampling, the AUCs for all the models were lower for Logistic Regression & Naïve Bayes (0.66 and 0.56 respectively), and the same for Random Forest & SVM (0.93 and 0.67 respectively). The Random Forest model which was again the highest performing model after parameter tuning (and 0.94 without tuning), with 0.84 recall for non-bankrupt companies and 0.89 recall for bankrupt companies (15,473 TP, 100 TN).

Oversampling Using the Synthetic Minority Over-Sampling Technique (SMOTE)

By creating synthetic samples of existing data which closely mimics the actual data, this technique uses oversampling to bring the number of observations from 73,725 for non-bankrupt companies and 443 bankrupt companies to 147,450 in each.

Although we expected that this should have given us improved overall performance compared to the other techniques on their own, we found that the results were the same as random undersampling (AUC of 0.72 for LR, 0.69 for Naïve Bayes, 0.93 for Random Forest, and 0.67 for SVM). Random Forest had the highest performance again, with 100 true positives, 15,473 true negatives and an AUC of 0.93. The recall scores indicate that there is also overfitting using this technique as recall was 0.96 for non-bankrupt companies and 0.05 for bankrupt companies, so this technique is also not recommended.

Combination of Undersampling & Oversampling Using SMOTE ENN

This technique combines random oversampling first and then undersampling, which creates a balance between improving the bias towards the class with less observations (bankrupt companies), while also applying undersampling to the class with more observations to reduce the bias towards it (non-bankrupt companies). It also adds the Edited Nearest Neighbor algorithm. ENN is used to remove observations with different labels from a minimum of half of its 'k nearest neighbors', and then handle the remaining observations using undersampling (which worked well previously).

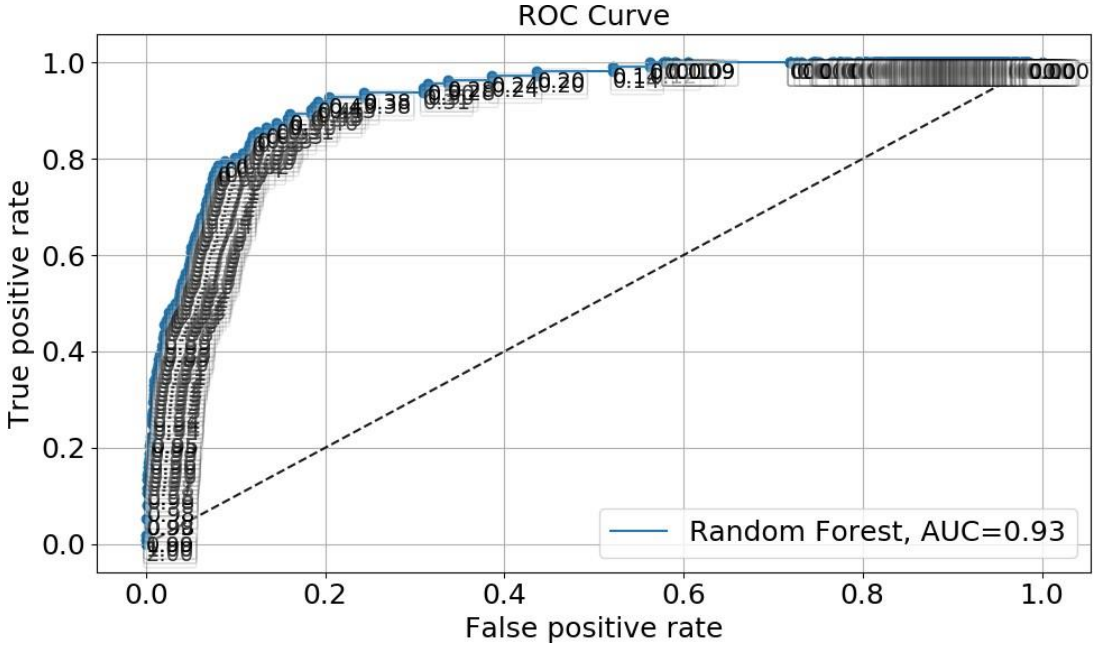
This brought the number of observations from 73,725 for non-bankrupt companies and 443 bankrupt companies to 142,251 in each (5,199 less observations than the previous two techniques, but 141,365 more than random undersampling alone). The results were the same as random undersampling & SMOTE (AUC of 0.72 for LR, 0.69 for Naïve Bayes, 0.93 for Random Forest, and 0.67 for SVM). The true positives/true negatives stayed the same for Random Forest.

Overall

The cross-validation scores for the two most high performing models were 0.93 vs. 0.94 AUC for Random Forest using undersampling, and Random Forest using SMOTE ENN. The cross-

validation technique split the data, fit the model and computed the score 10 consecutive times, and changed the splits each time cross validation was completed within the tuning process. For the rest of the models, the cross-validation score was within 0.01-0.05 of the AUC, which meant that the models performed well in the test set as well as the train set, there was no overfitting in the model, and the results of our final models are reliable. In terms of overall performance, we chose AUC as our performance metric over accuracy as it makes more sense in this case if we can detect true positives (bankruptcies) over accuracy.

Overall, it seemed that random undersampling performed better overall because we did not introduce synthetic or multiple copies of same data into the prediction, and the technique was more suitable for the data. This dataset had a class imbalance with sufficient observations in the minority class (bankrupt companies) required to make an accurate prediction, so undersampling was able to increase the sensitivity of a classifier to the bankrupt companies. The only concern with this approach is that a large amount of data is discarded, and our model was learning from 886 observations on either end rather than the full dataset. To ensure this is addressed, we can use the SMOTE ENN method to balance the two. Even though the recall is lower with 0.89 for bankrupt companies and 0.84 for non-bankrupt companies, the distribution is more favourable, and the approach can be repeated to minimize future risk of any changes in the data. The ROC curve is below, showing a plot of the Type I Error of the decision rule for the chosen model:



Feature Selection

In terms of the features that were generated from the model, all variables had explanatory value with the following highest-ranking variables: return on equity, EPS, operational margin, and leverage ratio (0.0049-0.0039). We used the original dataset to find these features since any oversampling or undersampling techniques would violate the independence assumption and provide us with inaccurate feature weighting. Performing variable selection before the changes were made to the data structure ensured that it stayed intact and that the results are reliable. We used cross-validation (repeating 10-fold) to estimate the generalization of the model to ensure there was no inherent bias or large variance between samples. This confirmed that our features

could be accurately interpreted and used in deploying our model across the organization. The feature importance weights are shown below:

Weight	Feature
0.0049 ± 0.0001	Return_on_Equity
0.0044 ± 0.0001	EPS
0.0042 ± 0.0002	Operational_Margin
0.0039 ± 0.0002	Leverage_Ratio
0.0038 ± 0.0001	Market_Book_Ratio
0.0036 ± 0.0002	Employee_Growth
0.0032 ± 0.0001	Asset_Turnover
0.0031 ± 0.0001	Liquidity
0.0029 ± 0.0001	Productivity
0.0025 ± 0.0001	Sales_Growth
0.0025 ± 0.0001	Assets_Growth
0.0015 ± 0.0002	Profitability
0.0015 ± 0.0002	Tobins_Q

Conclusion

Based on these results, we recommend selecting the Random Forest model using SMOTE ENN for the bankruptcy problem, to balance the dataset, avoid adding bias to the model, and ensure a repeatable approach for the business in the future.

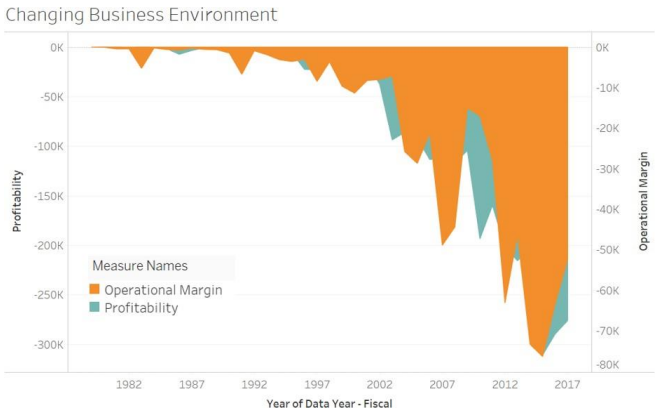
As a lending company and a potential investor in a company, we need to feed important and accurate information into our model for the features that we know have an impact in predicting whether a company goes bankrupt, such as return on equity, earnings per share, operational margin, leverage ratio, and market book ratio. To use this in a practical setting in the financial world, we could embed this model in the backend of our website for loan applications so that customers (potential companies) could fill out an application online, and our tool would tell the customer whether it was accepted or not. This would automate a large portion of the backend work our representatives would have to do going through loan applications, and our tool could record and provide the customers with specific reasons as to why their loan was approved or denied (ratios, margin, growth, etc.).

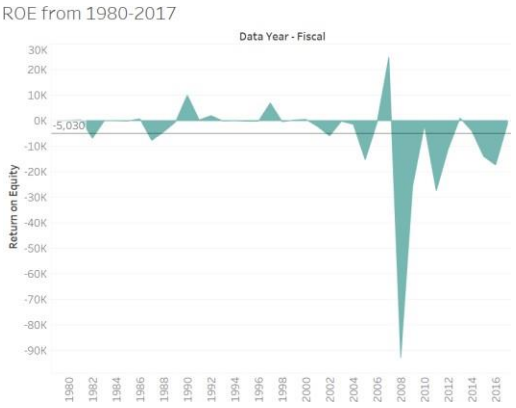
The company needs to continually monitor this data as it can change over time, resulting in declining model performance and potential model drift in the predictive power. Once the model is in production, we recommend having a team monitor the model as new information becomes available to ensure accuracy long-term.

Ultimately the model is a tool to decide if the lenders should loan the prospective company any money. The lending company can use this as another consideration and evaluation tool to minimize overall risk as the business landscape evolves, and to stay relevant in increasingly competitive markets within the financial industry.

Appendix

Appendix 1: Changing Business Environment





Appendix 2: Individual Class Accuracies for Random Undersampling

** Note, below is the Confusion Matrix for reference (bankruptcy 1 for yes, 0 for no, predicted values on top and actual values on the left)*

	0	1
0	TN	FN
1	FP	TP

Model	**TP**	**TN**	**FP**	**FN**	**Recall for Bankrupt**	**Recall for Non-Bankrupt**	**AUC**
1) Logistic Regression *(before tuning)*	57	12911	55	5520	0.70	0.51	0.61
Logistic Regression *(after tuning)*	70	13742	42	4689	0.62	0.75	0.72
2) Naïve Bayes *(before tuning)*	45	16368	67	2063	0.40	0.89	0.78
Naïve Bayes *(after tuning)*	107	925	5	17506	0.96	0.05	0.69
3) Random Forest *(before tuning)*	102	15430	10	3001	0.91	0.84	0.94

Random Forest *(after tuning)*	100	15473	12	2958	0.89	0.84	0.93
4) Support Vector Machines *(before tuning)*	53	13263	59	5168	0.47	0.72	0.62
Support Vector Machines *(after tuning)*	50	15621	62	2810	0.45	0.85	0.67

Appendix 3: Individual Class Accuracies for Random Oversampling

Model	**TP**	**TN**	**FP**	**FN**	**Recall for Bankrupt**	**Recall for Non-Bankrupt**	**AUC**
1) Logistic Regression *(before tuning)*	64	12669	48	5762	0.57	0.69	0.65
Logistic Regression *(after tuning)*	68	12601	44	5830	0.61	0.68	0.66
2) Naïve Bayes *(before tuning)*	107	1126	5	17305	0.96	0.06	0.50
Naïve Bayes *(after tuning)*	108	183	4	18248	0.96	0.01	0.56
3) Random Forest *(before tuning)*	2	18426	110	5	0.02	1.00	0.92
Random Forest *(after tuning)*	100	15473	12	2958	0.89	0.84	0.93
4) Support Vector Machines *(before tuning)*	53	13263	59	5168	0.47	0.72	0.62
Support Vector Machines *(after tuning)*	50	15621	62	2810	0.45	0.85	0.67

Appendix 4: Individual Class Accuracies for Oversampling SMOTE

Model	TP	TN	FP	FN	Recall for Bankrupt	Recall for Non-Bankrupt	AUC
1) Logistic Regression *(before tuning)*	70	12539	42	5892	0.62	0.68	0.67
Logistic Regression *(after tuning)*	70	13742	42	4689	0.62	0.75	0.72
2) Naïve Bayes *(before tuning)*	104	1464	8	16967	0.93	0.08	0.51
Naïve Bayes *(after tuning)*	107	925	5	17506	0.96	0.05	0.69
3) Random Forest *(before tuning)*	45	18153	67	278	0.40	0.98	0.94
Random Forest *(after tuning)*	100	15473	12	2958	0.89	0.84	0.93
4) Support Vector Machines *(before tuning)*	53	13263	59	5168	0.47	0.72	0.62
Support Vector Machines *(after tuning)*	50	15621	62	2810	0.45	0.85	0.67

Appendix 5: Individual Class Accuracies for Undersampling & Oversampling (SMOTE ENN)

Model	TP	TN	FP	FN	Recall for Bankrupt	Recall for Non-Bankrupt	AUC
1) Logistic Regression *(before tuning)*	75	10970	37	7461	0.67	0.60	0.67
Logistic Regression *(after tuning)*	70	13742	42	4689	0.62	0.75	0.72

2) Naïve Bayes *(before tuning)*	104	1463	8	16968	0.93	0.08	0.50
Naïve Bayes *(after tuning)*	107	925	5	17506	0.96	0.05	0.69
3) Random Forest *(before tuning)*	55	18025	57	406	0.49	0.98	0.94
Random Forest *(after tuning)*	100	15473	12	2958	0.89	0.84	0.93
4) Support Vector Machines *(before tuning)*	53	13263	59	5168	0.47	0.72	0.62
Support Vector Machines *(after tuning)*	50	15621	62	2810	0.45	0.85	0.67

Appendix 6: Hyper Parameter Tuning for Optimal Technique (SMOTE ENN)

Used Grid Search to find the optimal parameters for each model.

Model	**Parameters Used**
Logistic Regression	'C': 50, 'penalty': 'l2', 'solver': 'lbfgs'
Naïve Bayes	'var_smoothing': 0.001
Random Forest	'bootstrap': True, 'max_depth': 110, 'max_features': 3, 'min_samples_leaf': 3, 'min_samples_split': 8, 'n_estimators': 200}
Support Vector Machines	'C': 10, 'kernel': 'linear'