Predicting Likelihood of Financial Distress

Nicholas Kotsiantos – Heath Reineke – David Wallach
EECS 349 – Machine Learning – Northwestern University

1. Overview

For the economy to grow and remain prosperous, individuals and businesses need access to credit. Banks grant credit if they believe the borrower will repay the amount in full and on time. If more borrowers default on their loans than expected, the cycle of credit becomes disturbed, which can result in massive consequences. To help mitigate this risk of granting loans to unsafe borrowers, banks rely on credit scoring algorithms to help predict if certain borrowers will experience financial distress in the coming years.

2. Project Goal

Our goal for this project is to build a model that most accurately predicts if a borrower will experience financial distress within two years while also highlighting which features are most important for the prediction. We will first train our dataset on several machine learning algorithms, seeking to find the one that best predicts the likelihood of financial distress as this information will be critical for banks to best avoid borrowers that will likely default on their loans. In addition to this, we will then perform ablation to help discover which features are most important as this information will help borrowers make the best financial decisions for themselves.

3. Data

We found a dataset of historical data on 150,000 borrowers on kaggle.com. The dataset contains 10 numeric attributes such as age, debt ratio and monthly income to name a few and then a binary classification for each example of whether the borrower experienced financial distress or not within two years.

The one problem we had with this dataset was that it was imbalanced. ZeroR had an accuracy of 93% as approximately 140,000 examples belonged to one class, so the classifiers we tried could not perform much better than ZeroR. To combat this issue, we resampled various forms of our dataset by under-sampling the over-represented class by different degrees. We removed anywhere from about 90,000 to 130,000 instances from the over-represented class, yielding 5 different-sized datasets each with different ZeroR values.

To handle missing attributes, we employed two tactics. We either filled in the missing values with the mean of that attribute or we just removed that example from the dataset altogether, so we ended up with a variety of datasets to work with.

4. Approach

To approach the task, we first needed to train several learners on the various variations of the dataset to ensure that we found the best overall learner. Once we found the best learner, we then began to drop attributes from the dataset to figure out which attribute was the most important.

4.1 Classifiers

Initially, we used Weka to try train our dataset with the following classifiers to find the best overall one:

1. Decision Tree
2. kNN
3. Logistic Regression
4. Naive Bayes
5. Multilayer Perceptron

4.2 Dropping Features

After using Weka to train our dataset on the 5 different learners, we then wrote a python script to drop the i^{th} feature from the dataset and used the scikit learn library to train the dataset without the dropped featured on the different learners.

We were looking for the feature that reduced the model's accuracy the most when compared to the model's accuracy when it was trained on the full dataset. We felt this was an accurate measurement of how important a feature was.

4.3 Testing & Evaluation

Since our dataset did not come with a test set, we had to resort to the accuracy of the 10-fold cross validation. Like mentioned previously, we had 5 different-sized variations of our dataset, so we just took the average accuracy for each learner.

5. Results

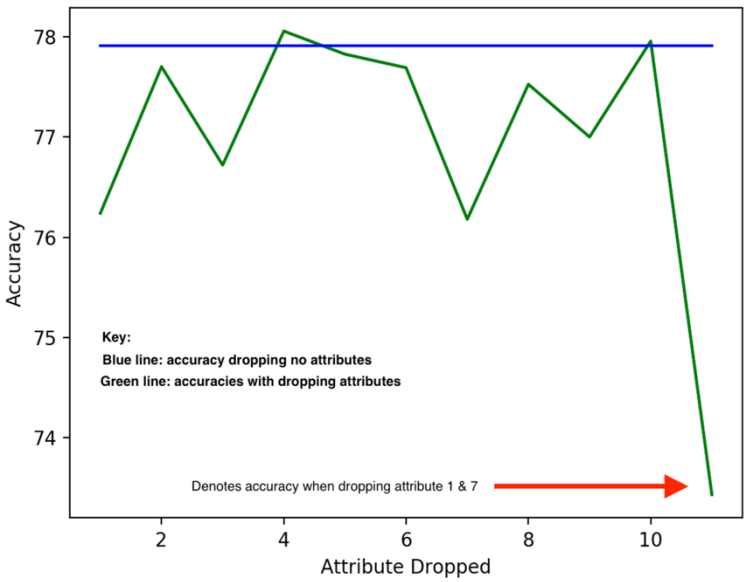
After training the 5 different learners on our dataset, the Decision Tree classifier achieved the highest accuracy, correctly classifying 82.09% percent of instances. The Decision Tree classifier was the optimal choice for our task because there were a few attributes that had very high information gain relative to the rest, so the Decision Tree classifier picked up on this and was able to construct a good model.

Classifier	Accuracy
ZeroR	70.92755%
Decision Tree	82.08595%
Nearest Neighbor	76.04915
Multilayer Perceptron	80.5424%
Logistic Regression	78.413175%
Naive Bayes	67.697675%

On the other hand, the Naive Bayes and Nearest Neighbor Classifiers performed the worst with accuracies of 67.70% and 76.05% respectively. As for Naive Bayes, it likely the case that it performed so poorly because certain features might have been slightly correlated. For example, it is probably the case that monthly income and debt ratio are somewhat correlated. One of the assumptions made with Naive Bayes is that the features are independent of each other. As such, it comes with no surprise that Naive Bayes performed so poorly since that assumption of feature independence was likely violated.

We initially hypothesized that Nearest Neighbor would be one of the best classifiers for this dataset as we felt that similar borrowers would face a similar outcome. However, with Nearest Neighbor, the classifier puts equal weight on all features. As we mentioned earlier, we found out that a few features had very high information gain relative to the rest. Since all the features are treated with equal weight, it now makes sense why Nearest Neighbor did not perform so well. In the future, it would be interesting to train the dataset using kNN with a distance metric that incorporates feature weighting.

Once we chose the Decision Tree as our best classifier, we then proceeded to drop features from the dataset one by one to see which dropped feature resulted in the accuracy to dip the most as this would be an indication of that feature's importance. As you can see in the chart, the largest dips in accuracy occurred when feature #1 and feature #7 were dropped, namely NumberOfTimes90DaysLate and RevolvingUtilizationOfUnsecuredLines respectively. Moreover, we see an enormous dip in accuracy when both feature #1 and feature #7 were dropped, further highlighting their importance.



6. Conclusion

Although 82.09% is not a bad prediction, banks will obviously still grant loans to the wrong borrowers since a perfect model is just not possible. However, even a slight increase in the percent of granted loans that are paid back in full and on time will have a profound impact on the stability of economic activity. Moreover, by highlighting the features that are most important, borrowers will now be able to make smarter decisions in the future to improve on these features. Overtime, as borrowers continue to make more informed decisions, they should become less risky borrowers.

7. Work Distribution

David did most of the python work and creating the various datasets along with a lot of the website. Nick wrote the abstract and the final report along with helping out with Weka and solving problems. Heath ran the classifiers through Weka and scikit over the various datasets and helped with website.