The University of Calgary

Loan Default Classification Analysis

BTMA 531 L01

DA-8

Rupin Sehgal (30088101)

Rajko Gajic (30089231)

Rayff McMahon (30046287)

***Problem Formulation:***

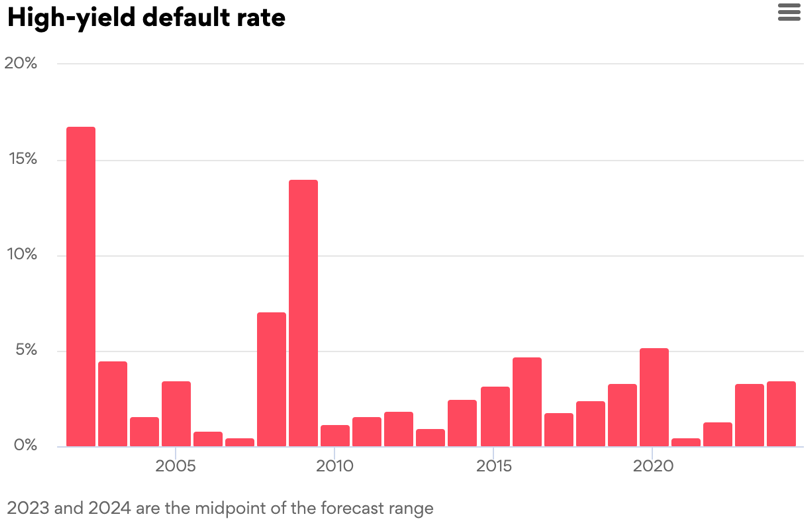
Chart

Description automatically generated with low confidenceBanks earn major revenue from lending loans, but these often carry risk as borrowers may default on the loan. According to a recent study, one and six Canadians have said they are likely to default on a major loan or mortgage (Heaven ,2023). As concerns continue to rise over inflation, elevated costs of goods, and the ability to afford housing, bankruptcies and renegotiations of loans are likely to rise as well. With the bank of Canada expected to hold interest rates steady, debt-servicing costs could reach potential record highs and credit losses tend to historically follow this trend (Lord, 2023). The growing financial stressors in households and macroeconomic factors have pushed banks to increase reserve funds. Recently the big six banks of Canada increased their provisions for credit losses to $2.5 billion, significantly up from the previous years $373 million (Dobby, 2023).

Interest Rates

To potentially mitigate this risk, banks have collected past data on the loan borrowers to classify whether a new borrower is likely to default. We will be predicting loan defaults for banks using multiple data preprocessing techniques and predictive tools to decrease potential losses, which in turn affect the economy's performance.

When analyzing our models, the key metrics will be the model accuracy, specificity, and sensitivity. Accuracy will disclose how many of the predicted binary classification values for default and non-default match with our test sample values. Specificity represents our type one error or when the model predicts a non-default classification, but the actual is a default. Whereas sensitivity represents our type two error or when the model predicts a defaulter, but instead they are a non-defaulter.

Chart, bar chart

Description automatically generatedForecasts for Leveraged loan and High-yield default rates

***Data:***

Once we had selected and imported our dataset it become evident that our dataset had many categorical variables that could provide significant insight to our models. The following variables were dummy coded, Application Type, Grade, Sub Grade, Employment Duration, Verification Status, Payment Plan, Initial List. From there to reduce noise, we removed all unnecessary variables that provided no value to our model. Also, we removed the original dummy coded variables as they had been coded to be binary for all the possible options. Furthermore, we removed one of the dummy coded categories to prevent multi collinearity from affecting our models. The following categories were removed, ID, Sub Grade, Payment Plan, Application Type, Grade A, Verification Status Not Verified, Batch Enrolled, Employment Duration, Loan Title, Accounts Delinquent, Sub grade A1, Payment Plan, Grade, Verification Status, Initial List Status, Application Type Joint, Employment Duration Mortgage, Initial List Status F. Now that the data had been prepped, we used the sample function to create the train and test subsets; the train subset was made up of 70% of our data and the test is 30% of our data.

Chart

Description automatically generatedUnfortunately, our dataset has a very high level of complexity with 69 variables after dummy coding (70th variable is our predicted class Loan.Status). To help reduce complexity in our data, while ensuring we retained as much data information as possible, we decided to employee principal component analysis. PCA is a widely used technique machine learning for reducing the dimensionality of large datasets while retaining the essential information. PCA is so useful because it allows us to simplify complex data sets by breaking them down into smaller, more manageable pieces. In essence, it works by identifying the patterns and relationships that exist between the different variables in a data set and using these to create new variables, or principal components, that capture as much of the original information as possible. From there it becomes possible to only retain a smaller subset of principal components that can explain most of the variation. By reducing the number of variables, PCA makes it easier to analyze and interpret complex data sets, and it can also help to eliminate any redundant or irrelevant information. This, in turn, can lead to better decision-making and more accurate predictions. After conducting PCA on our data, we plotted the proportion of variance explained by our principal components. By look at this graph, we can see that 12 principal components are able to explain much of the variation. After determining the number of PCs, we wanted to keep, we applied the PCA model to create our train and test subsets.

Chart, bar chart

Description automatically generatedOur group also recognized that our predicted class Loan Status suffered from the class imbalance problem. A class imbalance is a situation in which the distribution of classes in a dataset is heavily skewed, meaning that there are significantly more instances of one class than another. This can pose a challenge in machine learning models because the model tends to focus on predicting the majority class, while ignoring the minority class, which can result in poor performance on the minority class. To mitigate this, we implemented various techniques such as oversampling the minority class, under sampling the majority class, and using both sampling. Over sampling replicated cases of the minority class until the imbalance in resolved. Under sampling does the opposite, removing cases of the majority class until the imbalance is resolved. Both sampling combines the two previous techniques to resolve the imbalance. All these sampling methods were done using the ovun.sample command from the ROSE package. These techniques helped to improve the performance of our models on both the majority and minority classes.

Chart, bar chart

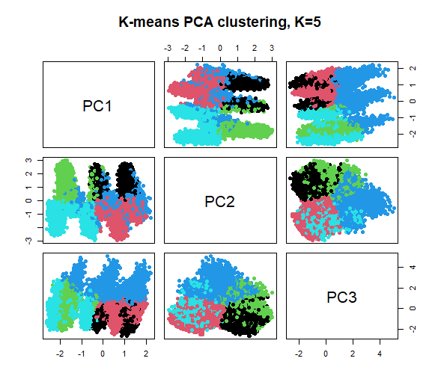
Description automatically generated

Overall, the preprocessing steps such as dummy coding and PCA helped to reduce the complexity of our dataset and eliminate noise, while addressing the class imbalance problem helped to improve the accuracy and performance of our machine learning models.

***K-means clustering***

We performed a K-means clustering on our PCA training set. The clustering was only used for the first 3 principal components. We did PCA because regular clustering was not working on the regular dataset as the set was far too large. We used the elbow method for finding the level of k to use for our clustering. We found that after 5 clusters the total withins was decreasing at a diminishing amount.

K-means clustering is not a model with a determined output, that means the figures we get from the model are abstract and need to be interpreted. The main problem we ran into was that without PCA the model did not work, but PCA made the K-means clusters much harder to understand. However, by looking at the breakdown of the components within each principal component we were able to get somewhat more of an idea of what the clusters were. For example, when we look at principal components 1 and 2, we see that the green and teal clusters are almost entirely separate from the other 3 clusters that are mixed. However, when looking at principal components 1 and 3 we see that the green and teal clusters are mixed together, as well as the red and black, whereas the blue is entirely on its own. This means blue has more dissimilarity from the other clusters when looking at PC1 and PC2.



***Cross Validation***

Once we have our dataset in hand, it's crucial to evaluate the performance of our machine learning models. However, it's important to ensure that our models generalize well and don't just memorize the training data. This is where cross-validation comes in handy.

Table

Description automatically generatedCross-validation is a technique used to evaluate the performance of machine learning models by dividing the dataset into k folds, where k is typically set to 5 or 10. The model is trained on k-1 folds and evaluated on the remaining fold. This process is repeated k times, each time using a different fold as the test set, and the results are averaged to obtain an overall performance metric. The advantage of cross-validation is that it provides a more reliable estimate of the model's performance than simply training and testing on a single split of the data. This idea is more interpretable with the visual on the left (scikit, n.d.). For our cross validation we used the trainControl command from the caret package with 5 folds. This control for our cross validation is then implemented into our models.

Overall, cross-validation is a powerful technique for evaluating the performance of machine learning models, but it requires careful preprocessing and appropriate choice of performance metrics to obtain reliable and meaningful results.

***Logistic Regression (GLM)***

Logistic regression is a method used in machine learning to model the probability of a binary outcome based on predictor variables. It is a type of generalized linear model (GLM) that is commonly used in classification problems where the response variable is binary, such as in predicting loan default or predicting customer churn. The response variable is modeled as a function of the predictor variables using a logistic function. This function maps any real-valued input to the range [0,1], which represents the probability of the binary outcome. The logistic function is defined as (Saini, 2021).:

*P(y=1|x) = 1 / (1 + exp(-z))*

where P(y=1|x) is the probability of the positive outcome (y=1) given the input variables (x), and z is a linear combination of the predictor variables, each weighted by a coefficient (Saini, 2021).:

*z = β0 + β1x1 + β2x2 + ... + βnxn*

where β0 is the intercept term and β1, β2, ..., βn are the coefficients for each predictor variable. The coefficients are estimated from the training data using maximum likelihood estimation (Saini, 2021). Once the model is trained, it can be used to predict the probability of the binary outcome for new data points.

To optimize the performance of our logistic models, we explored various preprocessing methods. Our initial approach involved training a basic GLM model and using the ROC curve to identify an optimal threshold for classification of our test set predictions. We used the caret package's confusionMatrix function to compute performance measures. Regrettably, the accuracy, sensitivity, and specificity of this model were suboptimal at 54.1%, 48.1%, and 54.7%, respectively.

To improve upon this, we assigned weights of 10 and 1 to defaulters and non-defaulters, respectively, and employed 5-fold cross-validation to over-sample the data. Subsequently, we built a GLM model on the non-PCA data, which exhibited a higher sensitivity of 81.9% but a lower accuracy of 26.2% and specificity of 20.7%. This indicates that the model struggled to identify defaulters but performed better with non-defaulters.

We then conducted oversampling with PCA, resulting in a better accuracy and specificity of 69.7% and 73.7%, respectively, but a sensitivity of 29.9%. Our subsequent attempts at under sampling without PCA and under sampling with PCA produced mixed results. The former resulted in a better sensitivity of 64.7%, but at the cost of reduced accuracy (40.7%) and specificity (38.3%). The latter yielded an improved accuracy of 75.1% and specificity of 80.3%, but a lowered sensitivity of 23.4%.

We also tested a GLM model with both sampling and no PCA, which performed similarly to our initial model but with reduced specificity. Finally, we implemented a model with both sampling and PCA, which produced the highest accuracy (79.3%) and specificity (85.5%), but a lower sensitivity of 17.5%.

Our observations suggest that PCA was able to capture non-defaulter patterns but was less effective in identifying defaulters. Additionally, both oversampling and under sampling had mixed effects on the model's performance measures. Furthermore, we found that using weights for defaulters and non-defaulters resulted in higher sensitivity, but at the cost of specificity.

Overall, logistic regression is a versatile and interpretable model that can effectively handle continuous and categorical predictor variables. We can use the insights gained from our explorations to select the appropriate preprocessing methods for our logistic models depending on the specific objectives and requirements of our analysis.

***K-Nearest-Neighbor***

KNN is a supervised machine learning algorithm used for both classification and regression problems. It is considered a non-parametric method since it doesn't make any assumptions about the underlying data distribution. Instead, it works by finding the k closest data points (i.e., neighbors) to a new observation and classifying or predicting based on the majority vote or average of their labels, respectively.

The choice of k is a hyperparameter that can be tuned to optimize performance. One of the main advantages of KNN is its simplicity and interpretability. It is also easy to implement and can capture non-linear decision boundaries and handle multi-class problems (IBM, n.d.).

However, KNN has some limitations and assumptions that need to be considered. One of the main challenges of KNN is its computational complexity and memory requirements, especially for large datasets, since it needs to calculate the distances between each observation and all other observations in the dataset. Another limitation of KNN is its sensitivity to irrelevant and redundant features, which can affect its performance and lead to the curse of dimensionality problem. Therefore, sampling or dimensionality reduction techniques such as PCA helped to reduce the number of features and improve performance (IBM, n.d.).

The choice of k can also affect the performance of KNN. A small value of k can lead to overfitting, which means that the model is too complex and captures noise instead of the underlying patterns in the data. On the other hand, a large value of k can lead to underfitting, which means that the model is too simple and fails to capture the complexity of the data. Therefore, the choice of k needs to be tuned carefully based on the dataset and the problem at hand.

For our model, we used a testing range for k from 2 to 10. From there we setup our model to predict Loan Status using all other variables and we set the tuneGrid to our k range. Our first KNN was a both sampled PCA model with a weight of 10 for defaulter and 1 for nondefaulter and our second model included PCA and no weights. Unfortunately, neither model performed very well, even with the use of weights and sampling. Both models classified a miniscule number of records as defaulter(less than 10) and classified most predictions as non-defaulter.

In summary, KNN is a simple and powerful algorithm that can be used for both classification and regression problems, but it has some limitations and assumptions that need to be taken into consideration. Its performance can be improved by tuning the hyperparameters, applying feature selection or dimensionality reduction techniques, and choosing an appropriate distance metric.

***Neural Network***

Neural networks are a useful for pattern recognition and prediction in classification problem sets. They consist of three types of nodes, connections, or weights, as well as layers and hidden layers. Networks that have more than one hidden node, such as ours, are considered a deep learning neural network. These models are adaptive in response to changing input and output values. Adjustments can be made to layers, nodes, and weights to further improve the models’ training capabilities. Activation functions such as ‘logistic’ or ‘tanh’ can be used to change weights or algorithms such as backpropagation and gradient descent. A problem with neural networks Is the need a lot of training data, however we do have a larger set which was an advantage.

Two neural networks were used in our analysis, the first being a base level to give us initial insight to how well the model performs and the second being our best model after adjustments to the data and model were done. For the base level all numeric and dummy variables were used with the nueralnet function in R to predict loan status. Min/Max scaling was applied to the training set numerical values and then further used on the entire dataset’s numeric values. From there the scaled data combined with the categorical values were again split into the training and test sets to begin implementation of the network. The output from the neural network was then used to compute predictions based on test dataset. Then applying the which.max function to the predictions gave us our comparison values to use against the actual test classifications. These values as well are used for the confusion matrix to see sensitivity and specificity.

Now for the second network, the first change was the usage of the data from the principal component analysis are our new inputs. To balance out the class imbalance we used the ROSE package “both” sampling method and applied this to the new input data. The new training set was then used in the neural network along with two hidden layers and one node each. Backpropagation with weigh backtracking was implemented into the formula as well to assist the model when learning from mistakes. We were able to run a few more combinations of layers and nodes, however certain combinations came with much longer run times. Whether the number of layers increased or the number of nodes increased, significantly more time was required for each model to train. So overall, two layers with one node each was the best performing model from what we were capable of running. Unfortunately, using these hidden layers with neurons for deep learning creates a “black box” effect as it becomes less and less clear how the individual neurons across the layers arrive at the final output.

To be expected our base model which had the unbalanced classes, and all the numeric and dummy variables did not perform well. Largely due to the imbalance, the model almost only predicted status zero for all values with an accuracy of (91%), sensitivity of (99%) and specificity of (.02%). These results showed us how the model was just guessing that everyone fell into the non-default class as most of the input data consisted of that class and the model only predicted sixty values as a defaulter. This is where implementing pca, sample balancing techniques and adjustments to the model formula were needed.

After using different layers, neurons, and algorithms we were able to see a significant improvement for the model in all aspects compared with our base. We did see that as the number of layers increased the three metrics all got closer together around (50%), showing even predictions and errors across the classes. From our best model with two hidden layers, one node each and backpropagation, we were able to produce and accuracy of (60%), sensitivity of (62%) and specificity of (40%). We believed this was our best model as it predicted loan status in the most beneficial way for a real-world bank. Based on our parameters this model predicted the greatest number of true non defaulters and true defaulters, meaning the most amount of profitable loans would be given out. The model had the lowest number of false positives which translates to the lowest default rate and credit loss for the loans predicted to be given out. Finally, the model did predict more false negatives or individuals predicted to default that would not have defaulted. Although this occurred, we believe these do not carry as great of a financial burden to banks as the false positives since no money would have been lent in these cases. Overall if we were able to continue to increase the complexity of the neural network without the significant run times, we likely would’ve been able to further adjust the model to gain even better outputs to assist banks.

***Recommendation/Discussion***

We recommend that banks use logistic regression in determining whether they should give an individual a loan or not. We made this choice mainly based on specificity but also on accuracy. Initially we only looked at specificity because we thought that giving out more loans that would default would be the main metric. However, after playing around with the models we found that sometimes the model would force a high specificity and sacrifice the accuracy, which would be massively detrimental as banks would only give loans to few individuals that were almost guaranteed to not default. So, we chose the logistic regression model because among our models it had the best balance of specificity and accuracy, with 83.7% and 77.8% respectively. We went about it this way because although specificity is very important, we did not pick out KNN model which had the highest specificity at 91% because its accuracy was lower than the logistic regression model at 72.1%. We found the KNN model to be giving out fewer loans in general.

We learned many lessons while working on this project. The first being the class imbalance that we had to work through. This class imbalance meant that we had to oversample and under sample which involves altering our dataset more than we would have liked. Ideally, future students will want to pick more balanced datasets. This will save them time and effort, as well as creating more accurate models for their test sets as nothing must be modified. The next nugget of wisdom we would give to future groups is to try using a better computer or running the models on campus with available resources. We found that an issue with our neural networks was that the run-time, especially with larger datasets, would take a very long time. We found that a lot of learning and developing these models is letting them run, seeing the results, and then using that information to make small tweaks to the model. Having to go back and forth and change the model till we got it right took a lot of time, and most of the time was not writing the code but just waiting for R studio to run the code. We recommend that future students try to use the computers in the lab as for some students they will find that the desktops there have better computational power than their own personal laptop. Finally, we recommend that students try as many models as possible on their data. As we sat through the presentations our group got lots of ideas to run our code that we did not think of prior. For example, we would have loved to try Naïve Bayes and SVM, but we could not get it to work with our data. We recommend that students really take the time in advance to play around with all the different models and not just settle with models that work with the data. Overall, we are content with our project as we learned a great deal about data processing and how real-world data interacts with statistical models. We are also happy with logistic regression model for giving us very reasonable metrics in predicting defaulters when looking at home loan data.

***References***

3.1. cross-validation: Evaluating estimator performance. scikit. (n.d.). Retrieved April 14, 2023, from https://scikit-learn.org/stable/modules/cross\_validation.html

Dobby, C. (2023, March 3). Canada’s biggest banks set aside $2.5 billion to cover an expected wave of loan defaults. Toronto Star. Retrieved April 8, 2023, from https://www.thestar.com/business/2023/03/03/canadas-big-six-banks-set-aside-25-billion-as-they-prepare-for-credit-losses.html

Heaven, P. (2023, February 13). Posthaste: One in six Canadians say they are likely to default on a major loan as rate hikes pinch. The Financial Post. Retrieved April 8, 2023, from, https://financialpost.com/executive/executive-summary/debt-default-inflation-interest-rates

Lord, C. (2023, March 4). Canada’s banks brace for possible wave of loan defaults. Why that matters. Global News. Retrieved April 9, 2023 from, https://globalnews.ca/news/9526547/canada-banks-loan-defaults/

Mallees, N. (2023, April 9). Bank of Canada expected to hold interest rate again as inflation cools. Global News. Retrieved April 9, 2023, from, https://globalnews.ca/news/9611953/canada-interest-rate-april-2023-preview/

Rothman, L. (2023, January 20). Increase in high yield and leveraged loan default forecasts. Pensions & Investments. Retrieved April 8, 2023, from, https://www.pionline.com/interactive/increase-high-yield-and-leveraged-loan-default-forecasts#:~:text=Fitch%20now%20expects%20a%202.5,the%20default%20rate%20was%201.6%25

Saini, A. (2021, August 26). Conceptual understanding of logistic regression for data science beginners. Analytics Vidhya. Retrieved April 14, 2023, from https://www.analyticsvidhya.com/blog/2021/08/conceptual-understanding-of-logistic-regression-for-data-science-beginners/

Trading Economics. (2023). Canada Interest Rate. Retrieved April 9, 2023, from https://tradingeconomics.com/canada/interest-rate

What is the K-nearest neighbors algorithm? IBM. (n.d.). Retrieved April 14, 2023, from, https://www.ibm.com/topics/knn#:~:text=The%20k%2Dnearest%20neighbors%20algorithm%2C%20also%20known%20as%20KNN%20or,of%20an%20individual%20data%20point