Credit Risk Score Calculation and Anomaly Detection

Section A, Team 3

Mia Kwon, Zhixuan Li, Jason Liao, Yitian Wang, Sihan Zhou

Process Overview

Task 1: Create a response variable based on customers' credit records & Define/Clean Predictor Variables

Task 2: Create models (Lasso, Logistic Regression, Decision Tree, Random Forest, K-means)

Task 3: Check on and avoid over-fitting (K-fold validation)

Business Understanding

In our economy, the credit card industry has always been crucial in driving consumer spending and providing people access to credit. However, it occasionally faces the risks of payment defaults which are influenced by a multitude of various personal and macroeconomic factors. Therefore, a thorough understanding of consumer behaviors and current economic trends combined with the strategic implementation of preventive measures can help mitigate the risks associated with delayed payments.

By compiling a detailed profile on an individual's personal traits and qualities, credit card companies are able to make informed decisions regarding credit approvals. This multifaceted approach can help them identify applicants who are likely to maintain a healthy credit relationship, thereby safeguarding the interests of the issuer while extending credit facilities to deserving candidates.

Data Preparation

1. Creating Responsive Variable

In this case, despite we having “status” variable representing the status of the loan for the particular month (0: Loan is 1-29 days past due; 1: Loan is 30-59 days past due; 2: Loan is 60-89 days overdue; 3: Loan is 90-119 days overdue; 4: Loan is 120-149 days overdue; 5: The loan is significantly overdue (more than 150 days) or written off as bad debts; C: Indicates that the client paid off their loan for that month; and X: Indicates that the client had no loan for the month), there is no given criteria and value representing the clients’ approval features based on the status of loan and overdue time. Therefore, we need to create a new responsive (approval) variable based on loan status and overdue time. Based on our perception of risk of default and research on bank policies, we considered two methods to create the responsive(approval) variable: #1. The criteria of approval is whether a client has overdue for more than 59 days in the past two months (including the current month). In other words, we assign “not approve” to clients with "months_balance” in 0 or -1 and “status” from 2 to 5; and #2. Power month indicators(<=0) to time overdue (status 0-5), I.e.,(time_overdue^month_indicator), indicated as overdue factor. The longer the record history is from now, the smaller the overdue factor, and the less significant it is to determine if one should be approved. Similarly, the shorter the overdue period, the less sever the overdue is, the smaller the overdue factor is, and the more likely we shall approve one’s application. Then I plus one to the overdue factor, getting a result-range from 1 to 2. For result close to 1(with a smaller overdue value), we should approve. For it closer to 2 (with a large overdue value), we should not. Then our group shall determine a threshold, so that customers below the threshold will be approved, and who are out of the range are not approved. The advantage of adopting #1 is it is straightforward, easy to understand, and easy to apply.

However, this method is comparatively arbitrary: it ignored the importance of historical records and other statuses and generated a binary outcome. The outcome it produces is highly unbalanced: only about 1% of customers won't be approved, which means our model must be highly discerning, sensitive, and accurate to beat the null model of everyone approving. The alternative method (#2), on the other hand, preserves all status and historical records when capturing the important feature that 1. the longer the overdue is, the less it mattered, and 2. the larger the overdue is, the more it mattered. However, this method is more complicated compared to the previous method, which makes it harder to evaluate its effectiveness. After comparing and contrasting two methods, we decided to pick up method 1, which produces a more comprehendible result. Despite the simplicity and comprehensibility of method 1, the binary responsive variable created many troubles when we tried to build a sensitive and accurate model.

2. Cleaning Feature Variables

After having a responsive variable, we sorted the feature variables into three types: binary variables, continuous variables, and categorical variables. For the binary variables, we used the ifelse() function to assign either 1 or 0 to each value, converting them into numeric binary variables. The corresponding variables are 'Gender', 'Car', 'Realty', 'Mobile Phone', 'Work Phone', 'Phone', and 'Email'. Among the continuous variables that had numerical values, 'Children' and 'Income' were considered to not require further cleaning, thus we decided to leave them as they are. The columns 'days_employed' and 'days_birth' had negative values since they count backward from the date the data was collected. To convert them into appropriate numerical values, the absolute values of them were divided by 365 and then rounded down to the nearest whole number. Furthermore, we renamed each column to 'Age' and 'Work_year' to better

reflect the information. For categorical variables with multiple values('Edu_type'), we transformed non-numeric categorical variables into dummy variables. For those with fewer values ('Marital_status' and 'Inc_type'), we factorized the variables using as.factor() function and converted the factor back into numeric values using as.numeric() function. Each unique level or category was assigned a numeric value.

3. Handling Imbalance

After cleaning the entire data, we are moving towards fitting our cleaned data into models, figuring out the model that produces the most accurate prediction on approval condition. At this step, we must first deal with the imbalance in the dataset: only about 2% agents in our dataset appeared to not be approved. When dealing with highly imbalanced datasets, particularly in classification problems, the standard methods might not perform well because they may simply predict the majority class most of the time. In this case, a naive model might predict "approved" for all agents and still achieve high accuracy, but it would miss predicting the "not approved" cases, which are of primary interest. In other words, if we directly apply the unbalanced dataset to models, the null model of approving all could generate the best predictive outcomes. We want to solve this. We have three plannings to address the imbalance: 1. To expand the not-approve dataset till it matches the approve dataset. 2. To randomly select a sample from approve dataset that is slightly fewer than the not-approved dataset, and 3. To use models sensitive to minorities(like LASSO). Due to the reason that duplicating minority samples can cause over-fitting, we excluded plan #1 and work majorly on plan #2. After counting the number of not-approve in our data set as 637, we randomly selected 600 approve samples and combine them

into a new dataset “combined_data”. Then we split the new dataset into training data and testing data for model applications.

Modeling

Model selection:

The supervised models we selected are Logistic regression, LASSO regression, the Classification Tree, Random Forest, and the unsupervised model we worked on is k-means.

- **LASSO**

The reason for choosing LASSO is to avoid over-fitting considering the number of variables we have, so that we want to add a penalizing term to each variable. After splitting “combined_data” into training dataset and testing dataset by 70% 30%, we fit the training dataset into LASSO regression. Post data splitting, we delved into fitting the training dataset into the LASSO regression. Recognizing the importance of the lambda parameter in the LASSO model that controls the strength of the penalty, we adopted cross-validation. This iterative process aimed to identify the optimal lambda that minimized prediction error. By balancing model complexity and training data fit, cross-validation ensures our model is neither too simplistic nor overly tailored to the training data. Upon finalizing the best LASSO model with the optimal lambda, we transitioned to evaluating its performance on the testing dataset. It was essential to understand how our model fares when faced with data it hasn't been trained on. To provide a comprehensive view of the model's performance, we computed the F1 score as a better evaluative indicator for binary prediction to consider both precision (how many of the identified positive cases are actual

positives) and recall (how many of the actual positive cases were identified). An f1 score of 0.58 was achieved. The 0.58 f1 score means that the model's predictions have a moderate level of accuracy: it's better than random guessing but is still quite away from perfect. While 0.58 isn't a particularly low score, it suggests that there might be aspects of the model or data that could be refined or optimized to achieve better results. Compared the 0.7 F1 score we got from random forest, we believe that LASSO is not the most appropriate tool for binary prediction.

- **Logistic Regression**

The purpose behind choosing logistic regression was to form a baseline model for our experiment before we decided to pursue other alternatives. While using this type of model, we decided to perform it 5 times using different variables for each version. We determined our variables by factors such financial stability, family status, loan/credit history, demographic information, and asset ownership. The reason being that each one of them is a common metric of creditworthiness. While running these models, we also used the training data (70%) and testing data (30%) as well as a threshold of .5 for each of our models. After running each model through cross-validation, we looked at each model f1 score and determined that the highest one was loan/credit history with an f1 score of .60.

- **Decision Tree**

We chose the decision tree as it can evaluate which variables play a crucial role in prediction and model the non-linear relationships between features and the target variable. We tried several combinations of feature variables and finally selected all the features except for ID as our predictor variables. We first factorized the target variable 'Approval' in train and test data and removed all NA values detected in the data set. Then, we tried to balance the slight class

imbalance using ROSE. After running the model, we calculated the f1 score using sensitivity and specificity. The cross-validation result showed that the highest f1 score is 0.68.

- **Random Forest**

In order to boost prediction performance and accuracy, we further created a Random Forest model. Random Forest aggregates multiple decision trees to enhance predictive accuracy and reduce overfitting. For this model, we used the same train and test data that we used for the decision tree. We handled the missing values by removing them and also used all features except for ID as our predictor variables. After running the model, we assessed the performance using the 10-fold cross-validation. The highest f1 score was around 0.72, which was the highest among all of the models we created. The Random Forest model showed the highest level of accuracy and performance in prediction for our dataset.

- **K-Means**

We also adopted K-means unsupervised clustering tool. We first used elbow method to determine the optimizing clustering number to be three, then we conducted k-means with 3 clusterings. After getting the k-means results, we used a heatmap for visualization, trying to find the traits in the three clusters. However, unfortunately, it turned out that the k-means and its visualization is quite messy and does not make much sense. Therefore, we rejected this method.

Evaluation

The evaluation of the data mining results, particularly for predictive models, is crucial to ensure its utility and reliability in real-world scenarios. In the given problem of credit card approval prediction, the F1 score serves as a robust metric, especially given the class imbalance. In repeated testing of the models, an F1 score of around 0.6-0.7 suggests a reasonably good balance

between precision (how many of the predicted approvals were actual approvals) and recall (how many of the actual approvals were predicted).

For businesses, interpreting this score in the context of their operations is vital. If a model wrongly approves a credit card application, it might result in credit default, leading to financial losses. Conversely, if a model wrongly declines an application, the company misses out on potential revenue and risks frustrating the customer.

A business case should be developed taking into account both direct and indirect costs and benefits. Direct costs include potential losses from wrong approvals, while benefits might include revenue from correctly approved applications and enhanced customer satisfaction. Indirect costs could include loss of customer trust due to wrong declines. The ROI (Return on Investment) can be computed by considering the net gains (or losses) achieved through the model against the costs of developing and deploying the model.

In some instances, projecting expected improvement might be difficult. This is because the actual financial implications of each false positive and false negative can vary significantly. Alternatives could involve creating more conservative or liberal models based on the company's risk appetite and adjusting the threshold based on business objectives.

Deployment

Once the best model, in this case, Random Forest, is identified and fine-tuned, it can be deployed in the credit card application processing pipeline. As new applications arrive, they can be fed into this model to get real-time predictions on whether to approve or decline. The model can also be integrated with the business's IT systems, ensuring smooth operations.

Several issues need attention during deployment. First, the model's performance should be continuously monitored to ensure it remains consistent. Model drift, where the model's performance degrades over time due to changing data patterns, is a real concern. Regular retraining or updating of the model might be necessary.

Ethical considerations are paramount. Ensuring that the model does not inadvertently discriminate against certain groups of people based on sensitive attributes like race, gender, or age is crucial. Bias in predictions can lead to legal implications and reputational damage.

Risks and Mitigation:

Several risks are associated with the proposed plan. These include:

Model Drift: As mentioned, over time, the data's underlying distribution might change, leading to degraded performance. Regularly monitoring and retraining can mitigate this.

Over-reliance on the Model: Sole reliance on the model without human oversight might lead to systematic errors. It's advisable to have a manual review process, especially for borderline cases.

Data Privacy Concerns: Handling personal data brings about concerns related to data privacy and protection. Ensuring data encryption, anonymization, and compliance with regulations like GDPR can address these concerns.

Model Bias: If not checked, the model might inherit biases present in the training data. Regular fairness audits and perhaps using fairness-enhancing interventions during model training can help mitigate this.

By being proactive and continuously monitoring both the model's performance and the external environment, many of these risks can be managed effectively.

Appendix:

Contribution: equal contributions from all group members. 20% for each team mem