CS3244 Project Presentation Narration

Roderick:
Hello, I am Roderick from Group 35. Together with me are my groupmates, Tianwei, Alex, Joshua, and Rayan. We will be presenting on our group project – credit card approval prediction.

Here is a brief outline of our presentation.

First, let's go through the motivation behind this project, and an overview of the given dataset.

This project aims to improve the fairness and accuracy of credit card approvals. Traditional methods can be biased, leading to risky approvals or unfair rejections. Our goal is to use data-driven techniques to identify key factors of creditworthiness and build more transparent models.

We used the "Credit Card Approval" dataset from Kaggle. Each row represents an applicant with personal and financial details. However, the dataset is unlabelled, so we created our own good/bad labels using financial indicators to train predictive models.

Tianwei:
The second part focuses on data inspection, involving two input files: application record and credit record. Initially, these files do not contain labels, so labels were created manually to enable further analysis.

In the third part, data cleaning was conducted. Some feature variables were found to be impractical, and samples that did not conform to reality were removed. For instance, records with ages that were too large or too small were excluded. Additional data cleaning steps included converting "yes" and "no" labels to corresponding binary values (1 and 0), removing records of young individuals, and excluding non-binary samples. Finally, the dataset was sorted by ID to complete the cleaning process for the application recall fields.

For the second file, labels were created through analysis. The proportion of people with different overdue periods was calculated, and a 60-day overdue period was identified as a suitable threshold for distinguishing between bad and good accounts. The application recall form was subsequently re-labeled based on this analysis.

The fourth step involved Exploratory Data Analysis (EDA) to observe data distributions and assess the potential performance of models. The dataset contains both numerical and categorical features. Label distribution analysis revealed that approval data significantly outweighs rejection data, indicating an imbalanced dataset. Correlation analysis between numerical features and the labels showed that the correlation is quite small, suggesting that linear relationships may not strongly influence model performance. The distributions of numerical and categorical data further highlighted the data imbalance, with rejection labels being underrepresented.

To address this imbalance, oversampling was applied to the training dataset within the 70-85% range. A SMOTE (Synthetic Minority Oversampling Technique) method was used, taking into account both numerical and categorical features. This step aimed to mitigate the imbalance and ensure the model could learn from both approval and rejection data effectively.

Alex:
Before applying K-Means Clustering, a few preprocessing steps were done:
First, non numeric features were encoded. There was one ordinal variable, education type, that was encoded manually but the rest of the variables were encoded using one-hot encoding because there were a small enough amount of categories that not a significant amount of new

dimensions was added. The data was then standardized. The optimal cluster count was determined using inertia as a guide to see how compact each cluster was.

The initial K-Means was run using k = 5. The silhouette score, which finds how well separated the clusters are, was close to 0 indicating heavily overlapping clusters.

K-Means was reattempted using domain knowledge to select features. Instead of inertia, the Gap Statistic was used to examine how many clusters would be optimal because it was advised that this would be less subjective. The Gap Statistic indicated that 1 cluster would be optimal.

The key takeaway from K-Means clustering is that there are no clearly defined clusters in the data.

Roderick:
Moving on to feature engineering. Principal Component Analysis was used to reduce dimensionality and noise in our dataset. This helps uncover hidden structure and correlations, speeds up model training, and improves generalisation.

Alex:
PCA was run to see if there was a significant dimensional reduction possible. However, based on the results, the reduction would not be extremely significant and would lead to a loss in interpretability.

Roderick:
We used Linear Discriminant Analysis, to improve classification by maximising class separability. LDA also reduces dimensionality, but unlike PCA, it considers class labels—making it ideal for classification tasks. It works by computing the within-class and between-class scatter matrices, then solving an eigenproblem. We attempted to perform LDA on the dataset to reduce dimensionality, however, the classes were not separated very well. Both classes cluster at the -5 to 5 region, except some ‘rejected’ data points clustering at the -30 to -25 region.

Additionally, we used L1 Lasso regularisation, to improve model interpretability and perform feature selection. Lasso works by adding a penalty that shrinks less important coefficients to zero, effectively removing irrelevant features. After running Lasso regression, some irrelevant features with zero coefficients can be removed such as days employed, age, and student income type.

Joshua:
Our first model was a linear regression model with thresholding. We used it to see if a simple model could effectively capture the relationships.

As part of preparing the data:
We converted all categorical features l, into a numerical format using one-hot encoding.
Then, we scaled all the numerical features to ensure they were on a similar range,
Next we trained our model.
To find the optimal threshold level, we used the ROC curve method, specifically focusing on maximizing Youden's Index. Youden's Index helps find the threshold that best balances sensitivity – correctly identifying approvals – and specificity – correctly identifying disapprovals.
This approach is particularly suitable here because it doesn't assume an equal number of approvals and disapprovals in the data, and it's well-suited for risk-based decisions like credit approval where balancing different types of errors is crucial.

We ran the model twice—first on the original, unbalanced dataset, and then again after balancing it.
Performance improved across all metrics with the balanced data.

Overall accuracy was 72%
It does reasonably well at identifying disapprovals, with a specificity of 0.554—so it flags 55% of those who should be rejected.
The ROC-AUC is 0.699, suggesting the model has some ability to distinguish between approvals and disapprovals.
Here's a bright spot: precision is high at 0.907. When the model predicts an approval, it's right 91% of the time, meaning false positives are minimal.
Recall is decent at 0.750, meaning it finds about 75% of the true approvals, but still misses a quarter of them.
The F1 score is 0.812, showing a good balance between precision and recall when it comes to identifying approved applications.

Bottom line: balancing helped, and the model's high precision makes it reliable for approvals. However, its lower specificity leads to approving more risky applicants, a concern since approving the wrong applicant can mean losing the principal, while rejecting a good one only loses interest. It's strong on approvals but needs better rejection accuracy.

Looking into the model, we identified the top factors that most hinder credit approval chances.
On the right side of the chart, the top five stand out clearly:
Income Type - Pensioner: Being a pensioner boosts your approval odds significantly, with a coefficient of +0.8244.
Family Status - Single/Not Married: This status increases your chances by +0.5400.
Family Member Count - Zero: Having no family members adds a +0.5013 boost.
Days Employed - Clean Record: A clean employment history contributes +0.4949.
Family Status - Widow: This status also positively impacts approval by +0.4911.
Hence, we would advise consumers that if they have these characteristics it will significantly hinder your credit approval chances as they signal lower financial reliability.

Rayan:
One of the models that we used for this problem dataset is logistic regression which works really well with these kinds of binary classification tasks. During the implementation I started with data pre-processing where I used one-hot encoding to convert all the categorical variables and I also dropped unnecessary columns like 'id' and then I handled missing values to ensure data quality. To improve model performance we applied feature scaling using standard scaler followed by dimensionality reduction using PCA.

This approach preserved 95% of the variance while significantly reducing the number of features from an original dataset and this helped eliminate multicollinearity and improve computational efficiency. After this I split the data into 70-30 train test ratio and implemented GridSearchCV to find optimal hyperparameters. The search space included various regularisation strengths and different penalty terms.

Our initial evaluation with default 0.5 threshold revealed high accuracy, good precision, but very low specificity. The performance indicated that our model was very liberal in approving applications. While it got all true approvals, it also incorrectly approved many rejections.

To balance our model, I used the ROC curve to identify an optimum threshold of 0.85. This significantly improved specificity, which means the model was correctly identifying many more rejections now. The confusion matrix showed 498 correctly identified rejections as compared to 173 previously. Moreover, since the dataset was heavily biassed towards approval, we balanced it to 70%, which produced slightly balanced results.

The ROC-AUC curve improved to 0.69, which indicated moderate discriminative ability. This demonstrated that addressing class imbalance was crucial for developing an effective credit approval system.

We can also extract highly useful insights from the model coefficients and outputs. By exponentiating each coefficient, we obtain the odds ratio. For example, if owning a car has a coefficient of 0.7, then the odds of getting approved increase by a factor of roughly two. Additionally, we can calculate 95% confidence intervals for these odds ratios to identify which factors are robust predictors versus which ones are noisy or uncertain.

With tools like SHAP (SHapley Additive exPlanations) or linear explainers, we can break down predictions to see exactly how each feature contributes to a particular outcome. Finally, to ensure our predicted probabilities are trustworthy, we can plot calibration curves and compute the Brier score. One of the models we used for this prominent dataset is logistic regression, which works particularly well for binary classification tasks.

During the implementation, I started with data preprocessing, where I used one-hot encoding to convert all the categorical variables. I also dropped unnecessary columns, such as IDs, and handled missing values to ensure data quality. To improve model performance, we applied feature scaling using a standard scaler, followed by dimensionality reduction.

Using PCA (Principal Component Analysis), this approach preserved 95% of the variance while significantly reducing the number of features in the original dataset. This helped eliminate noise and improved computational efficiency. Afterward, I split the data into a 70-30 train-test ratio and implemented a grid search with cross-validation to find the optimal hyperparameters.

The hyperparameter tuning included testing various regularization strengths and different penalty terms. Our initial evaluation with the default 0.5 threshold revealed high accuracy and good precision but very low specificity. The performance indicated that the model was overly liberal in approving applications. While it correctly identified all true approvals, it also incorrectly approved many rejections.

To balance the model, I used the ROC curve to identify an optimal threshold of 0.85. This significantly improved specificity, meaning the model was better at correctly identifying rejections. As a result, the confusion matrix showed 498 correctly identified rejections compared to only 173 previously. Moreover, since the dataset was heavily biased toward approvals, we balanced it to a 70-30 ratio, which produced slightly more balanced results. The ROC-AUC curve improved to 0.69, indicating moderate discriminative ability. This demonstrated that addressing class imbalance was crucial for developing an effective credit approval system.

Roderick:
Now, I will be introducing our Polynomial Ridge Regression model. We used it to capture non-linear relationships, such as interactions between income and employment. Ridge's L2 regularisation helps control complexity and multicollinearity, improving classification accuracy for credit approvals.

It works by expanding features into polynomial terms up to a chosen degree, fitting the model by minimising a regularised loss function, then tuning the regularisation strength λ using k-fold cross-validation.

The hyperparameters we used for our model are shown here.

The model improved specificity, meaning fewer false positives—a key goal for banks to avoid risky approvals. While there was a small drop in some other metrics, the model showed moderate ability in identifying both approvals and rejections. Order 3 performed best, making the model moderately suitable overall, with better balance and far fewer costly misclassifications.

Here is the ROC and PR curve with thresholds, and the top 8 terms that contribute to the credit card approval.

Our next model is a Decision Tree model, which we selected because they produce clear, interpretable “if-then” rules. They also naturally capture non-linear relationships and offer visual insights into which applicant traits drive approval decisions.

At each node, the tree selects the feature and threshold that best reduce impurity. It then splits the data and repeats this process until a stopping condition is met. To predict, the tree simply follows the splits and assigns the majority class.

For this model, we used the original processed dataset and applied Lasso for feature selection. To avoid overfitting, we pruned the tree with the following parameters.

The model showed some improvement in precision and specificity, thanks to pruning and feature selection reducing false positives. It performed well in identifying approvals but struggled with rejections due to data imbalance. Overall, it’s not suitable for credit approval tasks, as false positives—approving risky applicants—are costlier than false negatives.
Here is the ROC and PR curve for our decision tree.

Decision trees offer clear, interpretable logic behind approvals. We found that pensioner status was the most influential—retirees were always rejected. Surprisingly, applicants earning over $1,057,500 were also consistently rejected. And those with large families, particularly more than 6 children, had a high likelihood of rejection. These patterns highlight how certain traits strongly influence decisions, even when they may seem counterintuitive.

To reduce overfitting and improve prediction accuracy, we trained a Random Forest model. Each tree is trained on a random subset of applicants using bootstrap sampling. At each split, only a random set of features is considered. We built many trees—1,000 in our case—and the final prediction is made by majority vote. Other tuned parameters are shown on the slide.

For our Random Forest model, we saw a significant boost in specificity, meaning fewer false positives—critical in credit approvals. However, recall dropped slightly due to increased false negatives after balancing the data. Overall, this model identifies approvals well, but struggles with rejections. Because false positives are more costly in this context, this model is ultimately not suitable.

Here is the ROC and PR curve for our random forest model, and the top 10 gini importances which correspond to the most important user statistics in deciding credit card approval outcomes.

Now, let’s compare all the models we trained so far. This is a breakdown of all the evaluation metrics for all our models.

Unfortunately, none of the models performed exceptionally well in predicting credit card approval outcomes. Some possible causes of this include the curse of dimensionality from one-hot encoding and polynomial expansion, and synthetic data, which may not fully reflect real-world applicants.

If we had to choose a model, there are 3 models that stand out. Logistic regression is reliable and probabilistic—ideal for thresholding in credit decisions. Polynomial Ridge Regression with order 3 gave the highest precision and specificity, meaning fewer risky false positives. Lastly, Decision Tree scored the highest in ROC-AUC and PR-AUC, offering the best separation.
That brings us to the end of our presentation. We hope this gave you insight into how machine learning can be used to improve credit card approval processes, and the challenges involved in building fair, interpretable models.