COMP9417 Project: Can I Speak to the Manager?
Machine Learning for
Customer Feedback Classification

Group Name: Gazebo

Group Members:

1. Xaeils Diomampo (z5361702)
2. Ahmad Nasiruddin Dzulkifli (z5442313)
3. Zonglin Li (z5233101)
4. Kevin Zhou (z5342593)
5. George Jiao (z5483184)

Link to the presentation folder:

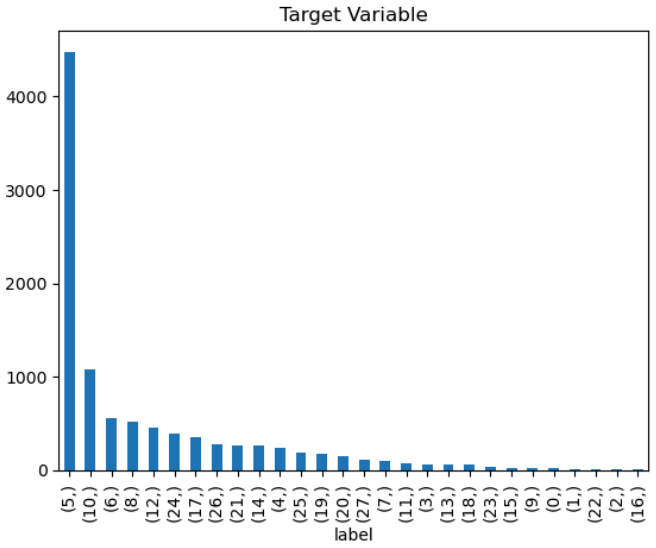
https://unsw-my.sharepoint.com/:v:/g/personal/z5342593_ad_unsw_edu_au/EfLi9uSEVdZPlBIBhD8-IrgBosZF98EarTnDfQuNfMCNwQ?e=YLknev&nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJTdHJlYW1XZWJBcHAiLCJyZWZlcnJhbFZpZXciOiJTaGFyZURpYWxvZy1MaW5rIiwicmVmZXJyYWxBcHBQbGF0Zm9ybSI6IldlYiIsInJlZmVycmFsTW9kZSI6InZpZXcifX0%3D

Introduction:

The task is to create a multiclass classification model for 10000 training instances of 28 classes. However, the main issues for this task are the imbalanced data for the 28 classes and the fact that there are 300 features, making it harder to identify the predictive features. To handle these issues is by feature selection, and by using the Synthetic Minority Oversampling Technique (SMOTE) method to handle imbalanced data.

Exploratory Data Analysis (EDA) and Literature review:

We identified that the training instances contain 300 numerical features. It has no null values, and all of the features are normalised. When we tried to use a heatmap to see any correlation between the features, there seemed to be no correlation between the features at all. So, now we take a look at the distribution of the target variable, which we have identified to be very imbalanced. We can see that class 5 has more than 4000 instances, but the rest of them have fewer than 1000 instances, and one of them has fewer than 5 instances, which is too low to make a model.



Therefore, to handle imbalanced data, we plan to use SMOTE which will create synthetic data, using K-Nearest Neighbours (KNN) (Chawla, N. V. et al, 2002).

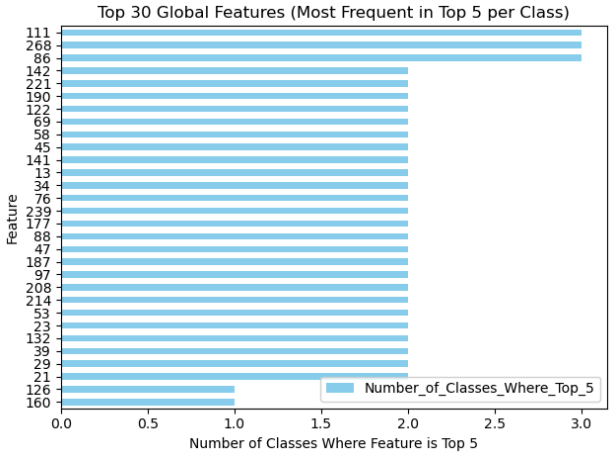
Next, we will also use different kinds of feature selection methods, and pick the one that we believe is the suitable one. The methods are logistic regression and random forest feature importance, and greedy selection, in which will be explained in the Methodology section.

Methodology:

Feature Selection:

1. Logistic Regression Feature Importance

We fit a logistic regression model with the data, and rank the features based on coefficient values. From that, we identified the top 5 features for each class, and we searched for the classes with the most classes in the top 5 features.



We identified that 111, 268 and 86 are the only 3 variables which appear as top 5 in 3 different classes, while 25 different features appear as top 5 in 2 different classes.

2. Random Forest Feature Importance

   We used logistic regression coefficient-based backward elimination to select features before Random Forest model training.

   After that, Random Forest was fed the chosen key features (best_features). The chosen key features are the same as the previous section. This method enhanced overall classification performance by lowering noise and concentrating on pertinent data.

We selected four different models: Logistic Regression, K-Nearest Neighbors (KNN), Random Forest, and Neural Network in order to cover a range of model complexity and strengths for this multi-class classification problem. In all models, we used accuracy, macro-averaged F1-score, and weighted-averaged F1-score to properly measure performance across both frequent and rare classes. We focused on the below three models only, as the logistic regression model didn't give valuable results.

KNN:

We chose KNN as a simple and intuitive baseline model. The basic idea is to classify a new sample based on the majority label among its *k* closest neighbors in the training data.

We tested several K values (3, 5, 7, 11), and used ‘weights= distance’ give more importance to closer neighbors. To better understand the model's behavior, we extended our search to odd K values between 1 and 29 and plotted both accuracy and macro F1-score.

The best performance was observed at **K = 7**, where the model achieved a validation accuracy of approximately **74.5%** and a macro F1-score of around **0.36**, striking a balance between overall accuracy and minority class performance.

KNN served as a useful baseline model due to its simplicity and transparency. However, its limitations became apparent in this high-dimensional, imbalanced classification task. While it provided us with a good starting point, we later focused on more advanced models to improve overall performance.

Random Forest:

We used a random forest model to solve the multi-class classification task. The dataset was imbalanced, which means some classes had many samples while others had very few. To handle this issue, we set class_weight='balanced' in the model. This makes the model pay more attention to the rare classes during training.

300 trees, a maximum depth of 20, and at least 3 samples in each leaf were used in the Random Forest Model. These settings gave us a good balance between training time and performance. We trained the model on 80% of the training data and used the other 20% to test how well the model worked.

Neural Network:

We built a feedforward neural network using TensorFlow/Keras, refining it based on insights from exploratory data analysis. Feature selection was guided by identifying low-variance and redundant features, leading to a reduced input space of 89 features from 300.

The final model includes three dense layers **(512 → 256 → 128)** with Swish activation, chosen for its smoother gradient flow compared to ReLU. Each layer integrates batch normalisation for training stability, dropout **(0.5 → 0.4 → 0.3)** to reduce overfitting, and L2 regularisation **(λ = 0.002)** in the first two layers to further control model complexity. To handle the class imbalance across 28 categories, we used square-root adjusted inverse

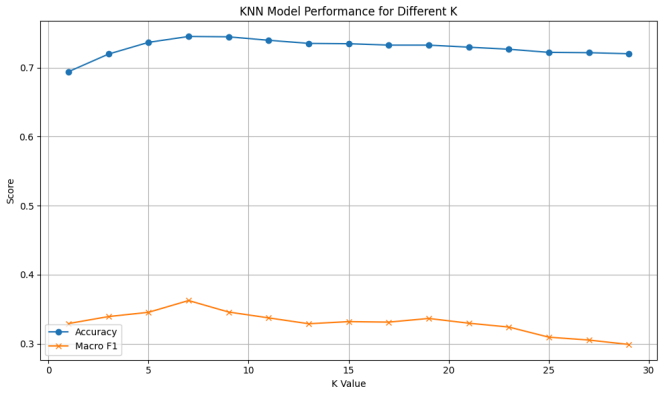
frequency class weights. This approach helped balance learning across all classes without over-emphasising the rare ones.

Training was carried out using the Adam optimiser **(learning rate = 0.0005)**, along with early stopping (patience = 20, monitored by validation AUC), adaptive learning rate reduction **(factor = 0.5, patience = 8)**, and model checkpointing. We used an 80/20 stratified train-validation split to maintain class distribution. We evaluated performance using accuracy, F1 scores, and class-specific AUCs, and conducted error analysis on both the validation set and the labelled portion of Test Set 2.

Result:

KNN Model Performance

As shown in Figure 1, we tested a range of K values from 1 to 29 (odd numbers only) to determine the optimal number of neighbors. Both accuracy and macro F1-score were plotted for each value. The performance peaked at K = 7, where the model achieved a validation accuracy of 74.5% and macro F1-score of 0.36



Random Forest:

The weighted F1 score is high because the random forest performed well on the majority of the classes, which had more samples. However, the macro F1 score is low because some rare classes still had low recall and F1 values. For example, classes like 1, 2, 9, 13, and 14 had zero F1 scores due to low number of samples (only 1–3 samples). Random Forest showed strong predictive ability for common classes but limited generalization for rare classes. While the use of SMOTE and class weighting helped, imbalanced data is still a major challenge.

The model is also tested using test set 2, and the result is summarized below.

Dataset	**Accuracy**	**Macro F1**	**Weighted F1**
Validation Set	73.6%	0.3409	0.6981
Test Set 2	52.97%	0.3432	0.5306

Neural Network:

The final model was trained on 89 selected features and evaluated on both the validation set and labelled portion of Test Set 2. It successfully predicted all 28 classes, demonstrating strong generalisation and effective handling of class imbalance.

Dataset	**Accuracy**	**Macro F1**	**Weighted F1**
Validation Set	73.6%	0.3984	0.7186
Test Set 2	52.97%	0.4041	0.5320

The model performed consistently well on common classes such as 5, 6, 8, 10, and 17, each achieving high F1 scores. Moderate performance was observed for mid-frequency classes like 12, 21, and 25. Rare classes with minimal representation (e.g., 0–3, 9, 13, 14) remained difficult to classify accurately.

Training dynamics showed smooth convergence with no signs of overfitting. Validation performance plateaued by epoch 20, with a peak AUC of 0.942 and stable precision, recall, and accuracy across epochs.

Discussion:

Our group tested four different models individually and observed the following findings:

- K-Nearest Neighbours (KNN):
  Simple to implement but unsuitable for high-dimensional and imbalanced datasets.
  KNN performed reasonably well for frequent classes but failed to classify rare

classes, resulting in a lower macro F1 score.

- Random Forest:
  Random Forest handled majority classes effectively and achieved a relatively high weighted F1 score. However, it showed a strong bias towards classes with more samples, leading to poor recall for minority classes and a low macro F1 score.

- Neural Network:
  When carefully tuned and combined with feature selection, the neural network achieved a better balance between frequent and rare classes. It consistently produced higher macro F1 scores on both the validation set and the test set, showing better generalisation ability.

Across all models, class imbalance remained the most significant challenge, particularly for classes with only a few samples. Models predict frequent classes better, while minority classes suffered from low recall and F1-scores.

Based on the results, the neural network gave the best weighted F1 score of 0.7186 on the validation set and 0.5320 on Test Set 2. Therefore, we decided to select the neural network model to generate predictions for both Test Set 1 and Test Set 2 submissions.

The model's predictions successfully met all specifications with (1000×28) and (1818×28) probability matrices, though inspection revealed minor class imbalance effects in low-frequency categories. This aligns with the observed distribution shift challenge noted in Test Set 2, suggesting potential value in domain adaptation techniques for real-world deployment.

Conclusion

This project investigated four classification algorithms for customer feedback categorisation, including KNN, Random Forest, and Neural Networks.

Through our evaluation, we found that Random Forest worked better on majority classes, while simpler models like KNN struggled with high-dimensional and unbalanced data.

After doing feature selection and tuning hyperparameters, the Neural Network gave the best results across all classes. It had the highest macro and weighted F1 scores among all models.

In the end, we chose the Neural Network because it had better generalization, more stable training, and handled class imbalance more effectively.

In the future, we could improve further by using ensemble methods and advanced sampling techniques like ADASYN to help with minority classes.

References:

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. Journal of artificial intelligence research, 16, 321-357.