Need for Balanced Dataset

Suppose we have a dataset for a loan eligibility prediction system, and only 10% of the loan applications in the dataset are rejected, while the remaining 90% are approved. This imbalance in the dataset creates a situation where loan rejections are relatively rare compared to approvals.

Now, consider a simplistic model that always predicts "Loan Approved" without taking into account any features or information about the loan applicant. This model essentially ignores any input data and consistently outputs the majority class.

Understanding Accuracy Calculation:

The accuracy of a model is typically calculated using the following formula:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Prediction}}$$

Example Calculation:

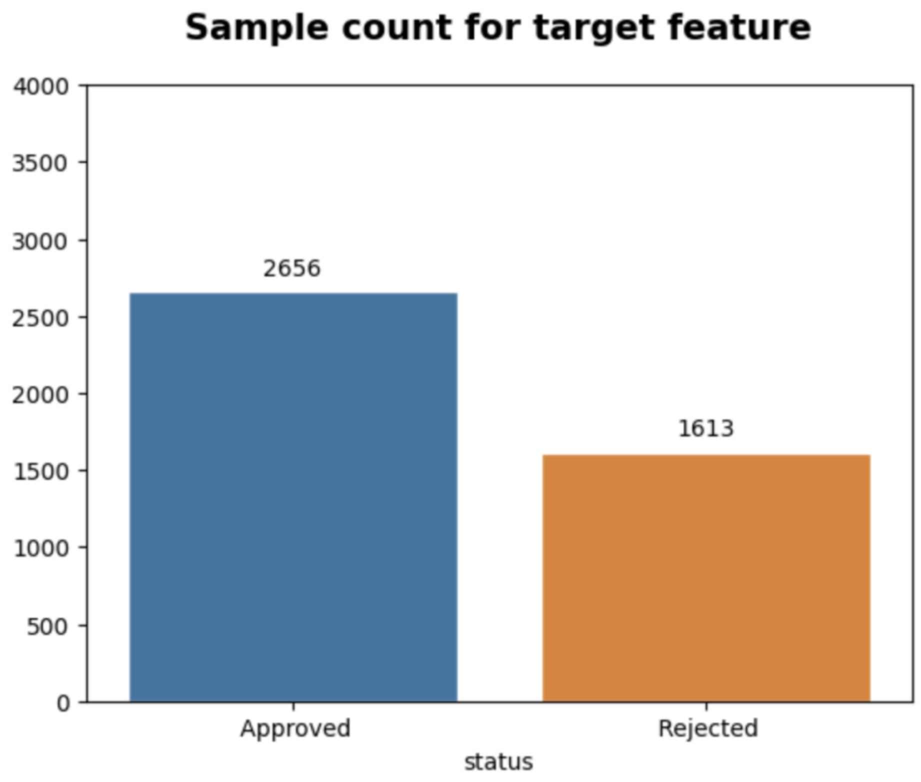
Suppose our loan eligibility dataset contains 1000 instances, with 900 instances labeled as "Loan Approved" (90%) and 100 instances labeled as "Loan Rejected" (10%).

Now, let's evaluate the accuracy of the simplistic "Loan Approved" model:

- **Number of Correct Predictions:** Since the model always predicts "Loan Approved," it will be correct for all instances where the actual label is "Loan Approved" (900 instances).
- **Total Number of Predictions:** The model predicts "Loan Approved" for all 1000 instances in the dataset.
- $\text{Accuracy} = \frac{900}{1000}$= 90%

So, the accuracy of this simple model without any feature consider is 90% **which is wrong**

MY DATASET



1. **Model Bias Towards the Majority Class:**
   - Models may develop a bias towards predicting the majority class (Approved) more frequently. This bias can occur because the model may achieve a seemingly high accuracy by predominantly predicting the dominant class, but its performance on the minority class (Rejected) may suffer.
2. **Misleading Accuracy:**
   - As discussed earlier, accuracy might be misleading in imbalanced datasets. If a model predicts "Approved" for every instance, it would have an accuracy of approximately 62% (2656426942692656). This high accuracy might give a false sense of the model's effectiveness.
3. **Poor Performance on Minority Class:**
   - The model may have difficulty learning patterns associated with the minority class (Rejected) due to its limited representation. As a result, the model may exhibit lower sensitivity or recall for the minority class, leading to a higher rate of false negatives.

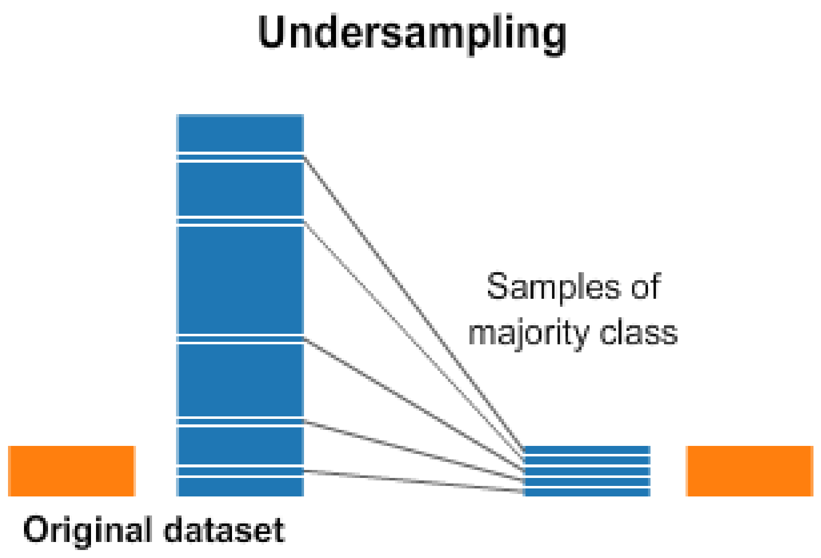
Mitigation Strategies:

- More sophisticated evaluation metrics, such as precision, recall, and F1 score, should be employed.

- Additionally, techniques like resampling (oversampling or undersampling) and the use of appropriate algorithms designed for imbalanced datasets can help improve the model's performance and fairness in real-world applications.

Balancing Technique

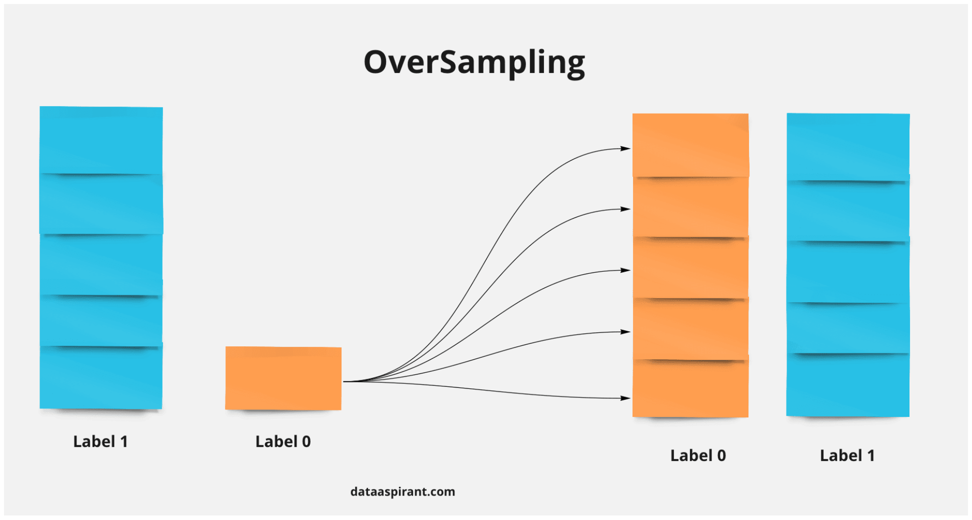
1. UnderSampling



Undersampling is not preferred because

- **Information Loss:** Undersampling's information loss can be minimized by carefully selecting instances for removal, ensuring that the retained majority class instances adequately represent the overall distribution.
- **Addressing Overfitting Risks:** Undersampling can increase the risk of overfitting, especially when the dataset is already limited. By reducing the number of instances in the majority class, the model might become overly sensitive to the remaining instances

1. Oversampling



- **Random Oversampling:** Random Oversampling is a technique used to address class imbalance by increasing the number of instances in the minority class. It involves duplicating or creating synthetic instances of the minority class randomly, aiming to balance the class distribution. While it's a straightforward approach, it may lead to overfitting due to the introduction of redundant instances.
- **SMOTE (Synthetic Minority Over-sampling Technique):** SMOTE generates synthetic instances for the minority class by creating artificial samples along the line segments connecting existing minority class instances. It aims to improve the model's ability to generalize to the minority class but may introduce noise in well-separated regions. **Borderline SMOTE** is an extension that specifically focuses on generating synthetic samples near the decision boundary, providing a more targeted oversampling approach.
- **ADASYN (Adaptive Synthetic Sampling):** ADASYN is an adaptive oversampling technique that adjusts the synthetic sample generation based on the density of the minority class. It places a higher emphasis on generating synthetic samples in regions with lower density, offering adaptability to varying complexities within the minority class. ADASYN's dynamic approach helps address situations where the density of the minority class varies across the feature space.

Borderline SMOTE focuses on borderline instances near the decision boundary, while ADASYN adapts its oversampling strategy based on the density of the minority class.

Model Trained on Unbalanced Dataset

We conducted model training on our loan eligibility dataset, which inherently has a class imbalance, with a significantly higher number of approved instances compared to rejected instances.

1. **Without Parameter tuning:**

- Accuracy: 70%
- The logistic regression model, when trained on the unbalanced dataset, demonstrated a 70% accuracy. However, it is crucial to highlight that this accuracy might be misleading, as the model's performance on the minority class (rejected instances) was suboptimal.

2. **With Parameter Tuning:**

   - Accuracy: 85%
   - After parameter tuning, we achieved an accuracy of 85%. While this represents an improvement, it is essential to acknowledge that the increased accuracy is largely driven by better performance on the majority class (approved instances).

Learnings:

1. **Class Imbalance Impact:**

   - The class imbalance significantly impacted model performance. Both models favored the majority class, leading to high accuracy but poor recall and precision for the minority class (rejected instances).

2. **Misleading Accuracy:**

   - The achieved accuracy, especially after parameter tuning, may provide a false sense of security. It was crucial to consider additional metrics like precision, recall, and F1 score, particularly when the consequences of misclassifying rejected instances are significant.

3. **Potential Solutions:**

   - Considering the class imbalance, I will be using oversampling techniques such as SMOTE and ADASYN to provide the models with a more balanced training set. This is expected to potentially improve their ability to identify rejected instances.