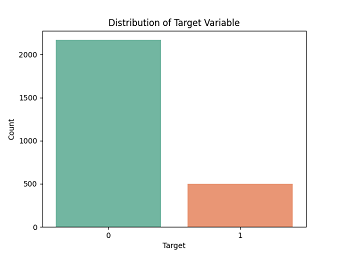
**Overview:**

We trained and evaluated three machine learning algorithms—Random Forest, LightGBM, and K-Nearest Neighbors (KNN)—on a dataset for some classification task. The dataset contains various features related to gender, age, number of policies, car category, car color, car make, local government area (LGA), state, product name, and the target variable. The target variable indicates whether a specific outcome occurred (1) or not (0).

Imbalanced Dataset:



It's crucial to note that the dataset exhibits class imbalance as shown in the plot above, where the instances of one class significantly outnumber the instances of the other. In our case, the positive class (1) is underrepresented compared to the negative class (0).

**Model Performance:**

The performance of each model is noteworthy, with each model showcasing unique strengths in accuracy, dataset handling, and efficiency, respectively. Below is the breakdown of each model:

1. **Random Forest:**

- **Accuracy:** 76.6%

- **Precision-Recall Trade-off:** Achieved relatively high precision (83%) for the negative class (0), but lower precision (29%) for the positive class (1). The recall for the positive class is also relatively low (18%).

**- F1-Score:** The weighted F1-score is 74%, indicating a reasonable balance between precision and recall.

**2. LightGBM:**

- **Accuracy:** 78.1%

- **Precision-Recall Trade-off:** Similar to Random Forest, achieved high precision (82%) for the negative class (0) but lower precision (29%) and recall (12%) for the positive class (1).

- **F1-Score:** The weighted F1-score is 74%, suggesting a comparable performance to Random Forest.

3. **K-Nearest Neighbors (KNN):**

- **Accuracy:** 78.7%

- **Precision-Recall Trade-off:** Higher precision (82%) and recall (95%) for the negative class (0), but lower precision (22%) and recall (6%) for the positive class (1).

- **F1-Score:** The weighted F1-score is 73%, indicating a reasonably balanced performance, but with a trade-off between precision and recall for the positive class.

**Recommendation:**

- **If Precision is Critical:**

- **Recommendation:** Random Forest or LightGBM

- Both Random Forest and LightGBM exhibit higher precision for the positive class compared to KNN. Choose based on other considerations such as interpretability and computational efficiency.

- **If Recall is Critical:**

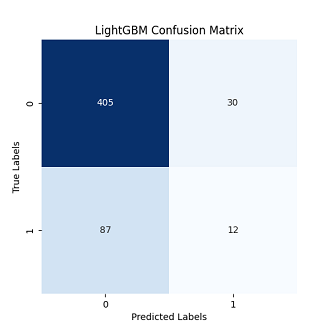
- **Recommendation:** KNN

- KNN shows higher recall for the positive class, making it suitable if identifying positive instances is a priority, even at the cost of precision.

- **Overall Balanced Performance:**

- **Recommendation:** LightGBM

- LightGBM provide a balanced performance, with reasonable accuracy and F1-scores and also exhibit higher precision for positive class which is our main aim (to be able to predict which customer will make an insurance claim). Choose based on other considerations such as model complexity and ease of interpretability. Below is the plot for the LightGBM output and the break down down of its output



Here's the breakdown of the plot above:

**True Negative (TN): 405**

This is the number of instances that were correctly predicted as the negative class. In other words, the model correctly identified 405 instances as the negative class.

**False Positive (FP): 30**

This is the number of instances that were incorrectly predicted as the positive class. The model wrongly classified 30 instances as positive when they were actually negative.

**False Negative (FN): 87**

This is the number of instances that were incorrectly predicted as the negative class. The model wrongly classified 87 instances as negative when they were actually positive.

**True Positive (TP): 12**

This is the number of instances that were correctly predicted as the positive class. In other words, the model correctly identified 12 instances as the positive class.

In summary:

The model correctly predicted the negative class 405 times.

The model correctly predicted the positive class 12 times.

The model incorrectly predicted the negative class 30 times.

The model incorrectly predicted the positive class 87 times.

.