PROJECT REPORT

Member 1: Muhammad Ghulam Abbas-29417

Member 2: Adnan Ali -29401

Q1: Explain each selected CI technique with diagram, if necessary, in your own words.

For our project we have chosen smote sampling, class weighting method and one class learning (Anomaly detection) by local outlier factor.

SMOTE for Class Imbalance:

SMOTE is another technique used to address class imbalance in machine learning datasets. Unlike random oversampling, which simply duplicates existing minority class data, SMOTE creates synthetic samples for the minority class.

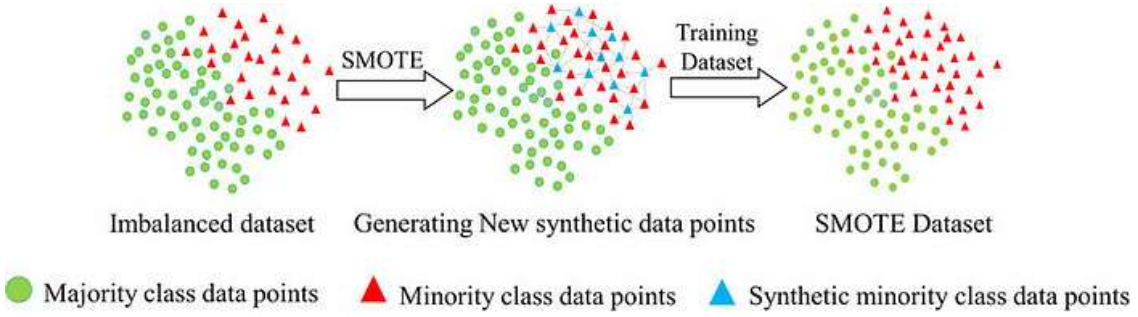
How SMOTE Works:

1. **Identify Minority Class:** SMOTE first identifies the minority class in the imbalanced dataset.
2. **Select Minority Sample:** A random sample from the minority class is chosen.
3. **Find nearest Neighbors:** The algorithm then finds k nearest neighbors (k is a user-defined parameter) for the chosen sample within the minority class itself.
4. **Synthetic Sample Creation:** SMOTE randomly selects one of these k nearest neighbors. Imagine a line segment connecting the original minority sample and its chosen neighbor in the feature space (space defined by the features used for classification). SMOTE creates a new synthetic sample by randomly selecting a point along this line segment.

In essence, SMOTE interpolates between existing minority class data points to create synthetic data that lies within the feature space of the minority class. This approach helps to increase the number of minority class samples without simply replicating existing data.

Benefits of SMOTE:

- **Reduced Over fitting:** Compared to random oversampling, SMOTE is less likely to lead to over fitting because it creates new, albeit synthetic, data points instead of simply copying existing ones.
- **Improved Minority Class Learning:** By increasing the representation of the minority class with synthetic samples that share similar characteristics, SMOTE can improve the model's ability to learn and classify the minority class more effectively.



2. Class weighting Method:-

The class weights method is a powerful technique to address class imbalance in datasets without modifying the data itself. It tackles the issue by assigning different weights to each class during the training process of a machine learning model. This approach gives more importance to the minority class, allowing the model to focus on learning its patterns effectively.

Steps Involved:

1. **Determine Class Distribution:** The first step involves analyzing the dataset to understand the distribution of each class. This helps identify the majority and minority classes.
2. **Calculate Class Weights:** Weights are assigned to each class based on its representation in the dataset. Typically, these weights are inversely proportional to the class frequency. In other words, the minority class receives a higher weight (e.g., 5), while the majority class receives a lower weight (e.g., 1). Common weighting schemes include inverse class frequency and inverse square root of class frequency.
3. **Apply Class Weights During Training:** These calculated weights are then incorporated into the loss function used during model training. The loss function essentially measures the model's errors. By incorporating class weights, the model gives a higher penalty to errors made on the minority class, forcing it to pay closer attention to those samples.

Advantages of Class Weights:

- **No Data Duplication:** Unlike oversampling techniques, class weights don't alter the original dataset by duplicating or generating new samples. This helps maintain the original data distribution.
- **Prevents Over fitting:** Since data duplication is avoided, class weights reduce the risk of over fitting that can occur with techniques like random oversampling. Over fitting happens when the model learns the training data too well and performs poorly on unseen data.
- **Simple Integration:** Many machine learning libraries and frameworks (like scikit-learn) have built-in support for class weights. This makes it easy to implement this technique during model training.

Disadvantages of Class Weights:

- **Complexity in Tuning:** Finding the optimal class weights can be challenging. Experimenting with different weighting schemes might be necessary to achieve the best performance.
- **Model Specific:** Not all machine learning models or libraries natively support class weights. In such cases, additional workarounds might be required.

3. One Class Learning Anomaly detection

One-class learning, anomaly detection techniques, such as Local Outlier Factor (LOF), can be effective in handling class imbalance. This approach is particularly useful when the minority class is significantly underrepresented or when it is difficult to obtain enough samples of the minority class. LOF is a density-based anomaly detection algorithm that identifies outliers by comparing the local density of a data point to the local densities of its neighbors.

What is Local Outlier Factor (LOF)?

Local Outlier Factor (LOF) is an algorithm that measures the local density deviation of a given data point with respect to its neighbors. It assigns an outlier score to each point, with higher scores indicating a higher likelihood of being an outlier.

How LOF Works

1. **Calculate Local Density**: For each data point, LOF calculates the local density based on the distances to its k-nearest neighbors.
2. **Compare Densities**: It then compares the local density of a point with the densities of its neighbors. Points with significantly lower density than their neighbors are considered outliers.
3. **Assign Outlier Scores**: Each point is assigned an LOF score, which reflects the degree to which it is an outlier. Points with higher LOF scores are more likely to be anomalies.

Steps in Using LOF for Class Imbalance

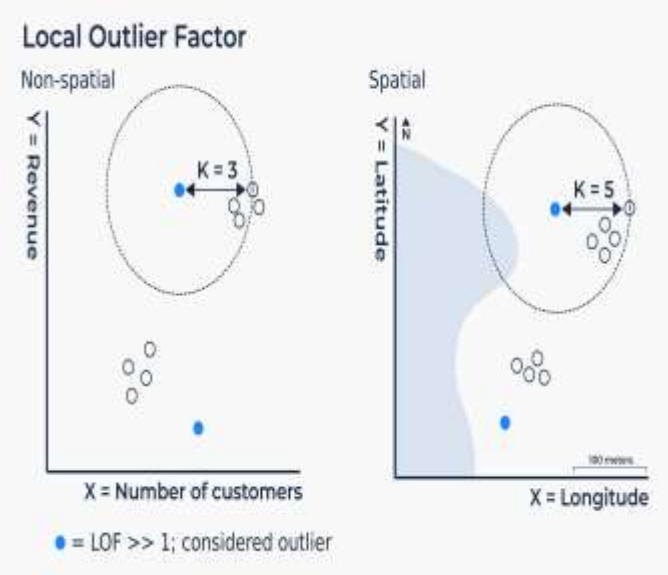
1. **Train on Majority Class**: Train the LOF model using only the majority class data.
2. **Detect Anomalies in Minority Class**: Use the trained model to predict the outlier scores of the minority class data. Points with high LOF scores are considered anomalies (outliers) from the majority class perspective; hence they belong to the minority class.

Advantages of Using LOF for Class Imbalance

- **Focus on Majority Class**: The model is trained only on the majority class, which helps in learning its distribution effectively.
- **Anomaly Detection**: LOF can identify samples that do not conform to the majority class distribution, effectively detecting minority class instances.
- **No Need for Extensive Minority Class Data**: This method works well even with very few samples of the minority class.

Disadvantages of Using LOF for Class Imbalance

- **Parameter Sensitivity**: The performance of LOF can be sensitive to the choice of parameters such as the number of neighbors (k).
- **Computational Complexity**: LOF can be computationally expensive, especially for large datasets.



Q2: What is the impact of each CI solution on the classification performance (compare with baseline)?

1. Dataset Credit Card Detection:-

BASELINE MODEL:-

This analysis examines the performance of five machine learning algorithms for classification: Logistic Regression, Naive Bayes, K-Nearest Neighbors (KNN), Decision Tree, and Random Forest. The evaluation is based on a dataset with two classes, likely imbalanced based on the number of data points in each class. Here's a breakdown of each model's performance:

Logistic Regression

- **Strengths:** High overall accuracy (0.9990) and AUC (0.9774) indicate good performance. Excellent at identifying the majority class (negative) with high precision, recall, and F1-score (0.9995).
- **Weaknesses:** Struggles with the minority class (positive) with a lower recall (0.61), meaning it misses a significant portion (45 instances) of positive cases.

Naive Bayes

- **Strengths:** Reasonable overall accuracy (0.9777). High recall (0.83) for the positive class, suggesting it captures most actual positive cases.
- **Weaknesses:** Lowest accuracy compared to other models. Very low precision (0.07) for the positive class, signifying a high number of incorrect positive predictions (false positives). This significantly impacts overall performance.

K-Nearest Neighbors (KNN)

- **Strengths:** Very high accuracy (0.9994). Well-balanced performance for the positive class with high precision, recall, and F1-score (0.8493). The confusion matrix shows few errors (false positives and negatives) for both classes.

Decision Tree

- **Strengths:** High accuracy (0.9991) and good overall performance.
- **Weaknesses:** F1-score for the positive class (0.7750) is lower than KNN and Random Forest, suggesting a trade-off between precision and recall. More false positives compared to KNN and Random Forest, but still perform well in detecting both classes.

Random Forest

- **Strengths:** Highest accuracy (0.9995) and AUC (0.9596). Excellent performance in both precision and recall for the positive class, resulting in the highest F1-score (0.8585). The confusion matrix shows minimal errors (false positives and negatives), indicating highly reliable predictions.

CONCLUSION

- **Best Performing Model:** Random Forest emerges as the top performer based on its high accuracy, precision, recall, and F1-score for the positive class.

- **Strong Alternatives:** KNN and Logistic Regression are strong contenders, especially for their accuracy and ability to reliably detect the positive class.
- **Underperforming Model:** Naive Bayes, due to its very low precision for the positive class despite having a decent recall.

1 .SMOTE:-

This analysis revisits the performance of five machine learning algorithms for classification: Logistic Regression, Naive Bayes, K-Nearest Neighbors (KNN), Decision Tree, and Random Forest. The evaluation is based on a dataset with two classes, where Synthetic Minority Over-sampling Technique (SMOTE) has been applied to address class imbalance.

General Observation:

SMOTE appears to have significantly improved the overall accuracy for most models. However, the near-perfect scores achieved by KNN, Decision Tree, and Random Forest raise concerns about over fitting. It's essential to evaluate these models on unseen data to assess their generalizability.

Individual Model Breakdown:

Logistic Regression

- **Strengths:**
    - High Overall Accuracy (0.9483) and AUC (0.9892) indicate good overall performance.
    - Balanced Performance: F1-scores for both classes are similar (0.9469 and 0.9496), suggesting balanced classification.
- **Weaknesses:**
    - Moderate Misclassification of Positive Cases: The model misses a portion of positive cases (4517 false negatives) despite a high recall (0.92).

Naive Bayes

- **Strengths:**
    - High Recall for Positive Class (0.85) suggests capturing most actual positive cases.
    - Reasonable Accuracy (0.9137).
- **Weaknesses:**
    - Lower Precision for Positive Class: The model has more false positives (1405) than ideal.
    - Lower F1-Scores Compared to Other Models.

K-Nearest Neighbors (KNN)

- **Strengths:**
    - Very High Accuracy (0.9990) and AUC (0.9997) indicate exceptional performance.
    - Perfect Recall for Positive Class (1.00): The model identifies all positive cases.
    - Minimal Classification Errors: Few false positives (115) and no false negatives suggest high reliability.
- **Weaknesses:**
    - Potential Over fitting: Near-perfect scores raise concerns about generalizability.
    - Computational Complexity: KNN can be resource-intensive for large datasets.

Decision Tree

- **Strengths:**
  - Very High Accuracy (0.9983) and AUC (0.9983) indicate excellent performance.
  - Balanced and High F1-Scores: Strong performance across classes (0.9983 for both).
  - Few Classification Errors: Minimal false positives (124) and negatives (67) contribute to reliability.
- **Weaknesses:**
  - Potential Overfitting: Similar to KNN, the high scores warrant evaluation on unseen data.

Random Forest

- **Strengths:**
  - Highest Accuracy (0.9999) and AUC (1.0000) suggest exceptional performance.
  - Perfect Recall for Positive Class (1.00): Identifies all positive instances.
  - Minimal Errors: Very few false positives (8) and no false negatives indicate highly reliable predictions.
- **Weaknesses:**
  - Potential Overfitting: The near-perfect scores require evaluation on unseen data.
  - Complexity and Interpretability: Random Forest can be more complex and harder to interpret than simpler models.

Conclusion

- **Best Performing Model:** Random Forest emerges as the top performer based on accuracy, AUC, and F1-scores. However, over fitting is a concern.
- **Strong Alternatives:** KNN and Decision Tree show excellent performance, but require assessment on unseen data.
- **Underperforming Model:** Naive Bayes exhibits lower accuracy and F1-scores compared to others.

2. Class Weighted Performance

This report analyzes the performance of five machine learning algorithms for classification: Logistic Regression, Naive Bayes, K-Nearest Neighbors (KNN), Decision Tree, and Random Forest. The evaluation is based on a dataset with two classes, where the Class Weighted Method has been applied to address class imbalance.

General Observation:

Class weighting significantly impacts all models compared to unweight results. It improves the models' ability to identify the minority class (positive class in this case), reflected in the increased F1-scores. However, there's a trade-off with the majority class, as evidenced by the increase in false positives for some models.

Individual Model Breakdown:

Logistic Regression

- **Strengths:**
  - High Overall Accuracy (0.9768) and AUC (0.9850) indicate good performance.
  - High Recall for Positive Class (0.93) suggests capturing most positive cases.
- **Weaknesses:**

- Low Precision for Positive Class (0.08) leads to high false positives and a low F1-score (0.1392). Consider using a cost-sensitive approach if misclassifying positive cases is more critical.

Naive Bayes

- **Strengths:**
  - High Recall for Positive Class (0.83) indicates effectiveness at identifying positive cases.
  - Reasonable Accuracy (0.9777).
- **Weaknesses:**
  - Low Precision for Positive Class (0.07) similar to Logistic Regression, resulting in a low F1-score (0.1314).

K-Nearest Neighbors (KNN)

- **Strengths:**
  - Very High Accuracy (0.9994) and AUC (0.9434) suggest excellent performance.
  - Balanced Performance: F1-score (0.8493) for the positive class indicates good balance between precision and recall.
  - Minimal Errors: Few false positives and negatives contribute to high reliability.
- **Weaknesses:**
  - Computational Complexity: KNN can be resource-intensive for large datasets.

Decision Tree

- **Strengths:**
  - High Accuracy (0.9991).
  - Balanced Performance: High F1-scores for both classes (0.7568 for positive).
  - Few Errors: Minimal false positives and negatives contribute to reliability.
- **Weaknesses:**
  - Potential over fitting: The high accuracy warrants evaluation on unseen data.

Random Forest

- **Strengths:**
  - Highest Accuracy (0.9995).
  - Balanced and High F1-Scores: Strong performance across classes (positive F1-score: 0.8558).
  - Minimal Errors: Very few false positives and negatives indicate highly reliable predictions.
- **Weaknesses:**
  - Complexity and Interpretability: Random Forest can be complex and harder to interpret than simpler models.

Conclusion

- **Class Weights Improve Positive Class Detection:** While there's a trade-off, class weighting addresses class imbalance and improves positive class identification.
- It is more generalizable there is less occurrences of over fitting
- **Random Forest as Top Performer:** It achieves the highest accuracy, precision, recall, and F1-scores for both classes.
- **Alternatives:** KNN and Decision Tree show excellent performance with high accuracy and reliable classification.

- **Lower Performing Models:** Logistic Regression and Naive Bayes have lower precision for the positive class.

3. One Class learning:-

- **Class Imbalance Impact:** All models exhibited lower performance in identifying the positive class (likely the minority) compared to the negative class. This is reflected in the lower F1-scores for the positive class despite high overall accuracy for some models.
- **Top Performers:** Random Forest achieved the highest overall accuracy and balanced F1-scores for both classes. KNN and Decision Tree also showed promising results with high accuracy and reasonable F1-scores for both classes.
- **Underperforming Models:** Logistic Regression and Naive Bayes had lower precision for the positive class, leading to lower F1-scores despite high recall.

2. Customer Churn Dataset

BASELINE MODEL:-

This analysis evaluates the performance of five machine learning classifiers - Logistic Regression, Naive Bayes, K-Nearest Neighbors (KNN), Decision Tree, and Random Forest, on a binary classification task. Here's a summary of the results for the baseline models:

Logistic Regression

- **Accuracy**: 0.8088
- **AUC**: 0.8599
- **F1 Score (Positive Class)**: 0.6227
- **F1 Score (Negative Class)**: 0.8720
- **Precision-Recall Trade-off**: Achieves a good balance between precision and recall for both classes.
- **Confusion Matrix**:
    - True Negatives (TN): 916
    - False Positives (FP): 80
    - False Negatives (FN): 189
    - True Positives (TP): 222

Naive Bayes

- **Accuracy**: 0.7591
- **AUC**: 0.8434
- **F1 Score (Positive Class)**: 0.6530
- **F1 Score (Negative Class)**: 0.8155
- **Precision-Recall Trade-off**: Demonstrates relatively higher recall for the positive class but lower precision compared to Logistic Regression.
- **Confusion Matrix**:
    - True Negatives (TN): 749
    - False Positives (FP): 247
    - False Negatives (FN): 92
    - True Positives (TP): 319

K-Nearest Neighbors (KNN)

- **Accuracy**: 0.7520
- **AUC**: 0.7665
- **F1 Score (Positive Class)**: 0.5315
- **F1 Score (Negative Class)**: 0.8313
- **Precision-Recall Trade-off**: Shows higher precision but lower recall for both classes compared to Logistic Regression and Naive Bayes.
- **Confusion Matrix**:
    - True Negatives (TN): 860
    - False Positives (FP): 136
    - False Negatives (FN): 213
    - True Positives (TP): 198

Decision Tree

- **Accuracy**: 0.7299
- **AUC**: 0.6723
- **F1 Score (Positive Class)**: 0.5355
- **F1 Score (Negative Class)**: 0.8096
- **Precision-Recall Trade-off**: Shows similar precision but lower recall compared to Logistic Regression, Naive Bayes, and KNN.
- **Confusion Matrix**:
    - True Negatives (TN): 808
    - False Positives (FP): 188
    - False Negatives (FN): 192
    - True Positives (TP): 219

Random Forest

- **Accuracy**: 0.7875
- **AUC**: 0.8273
- **F1 Score (Positive Class)**: 0.5698
- **F1 Score (Negative Class)**: 0.8589
- **Precision-Recall Trade-off**: Shows a relatively balanced performance between precision and recall for both classes.
- **Confusion Matrix**:
    - True Negatives (TN): 910
    - False Positives (FP): 86
    - False Negatives (FN): 213
    - True Positives (TP): 198

Conclusion

- Among the classifiers evaluated, Logistic Regression achieves the highest F1 score for the positive class, indicating a better balance between precision and recall for identifying positive instances.
- Naive Bayes shows competitive performance with relatively higher recall for the positive class but lower precision compared to Logistic Regression.
- KNN, Decision Tree, and Random Forest exhibit slightly lower performance metrics compared to Logistic Regression and Naive Bayes, with varying degrees of precision and recall for both classes.

1. SMOTE:-

After applying SMOTE (Synthetic Minority Over-sampling Technique) to balance the class distribution, let's evaluate the performance of the classifiers:

Logistic Regression

- **Accuracy**: 0.8030
- **AUC**: 0.8823
- **F1 Score (Positive Class)**: 0.8087
- **F1 Score (Negative Class)**: 0.7970
- **Precision-Recall Trade-off**: Achieves relatively balanced precision and recall for both classes.
- **Confusion Matrix**:
    - True Negatives (TN): 799
    - False Positives (FP): 232
    - False Negatives (FN): 175
    - True Positives (TP): 860

Naive Bayes

- **Accuracy**: 0.7740
- **AUC**: 0.8483
- **F1 Score (Positive Class)**: 0.7831
- **F1 Score (Negative Class)**: 0.7640
- **Precision-Recall Trade-off**: Demonstrates relatively balanced precision and recall for both classes.
- **Confusion Matrix**:
    - True Negatives (TN): 756
    - False Positives (FP): 275
    - False Negatives (FN): 192
    - True Positives (TP): 843

K-Nearest Neighbors (KNN)

- **Accuracy**: 0.8015
- **AUC**: 0.8647
- **F1 Score (Positive Class)**: 0.8178
- **F1 Score (Negative Class)**: 0.7821
- **Precision-Recall Trade-off**: Shows balanced precision and recall for both classes.
- **Confusion Matrix**:
    - True Negatives (TN): 736
    - False Positives (FP): 295
    - False Negatives (FN): 115
    - True Positives (TP): 920

Decision Tree

- **Accuracy**: 0.7807
- **AUC**: 0.7808
- **F1 Score (Positive Class)**: 0.7815
- **F1 Score (Negative Class)**: 0.7800
- **Precision-Recall Trade-off**: Demonstrates balanced precision and recall for both classes.
- **Confusion Matrix**:

- True Negatives (TN): 803
- False Positives (FP): 228
- False Negatives (FN): 225
- True Positives (TP): 810

Random Forest

- **Accuracy**: 0.8388
- **AUC**: 0.9142
- **F1 Score (Positive Class)**: 0.8439
- **F1 Score (Negative Class)**: 0.8334
- **Precision-Recall Trade-off**: Shows balanced precision and recall for both classes.
- **Confusion Matrix**:
  - True Negatives (TN): 833
  - False Positives (FP): 198
  - False Negatives (FN): 135
  - True Positives (TP): 900

Summary and Considerations

- After applying SMOTE to address the class imbalance, all classifiers demonstrate improved performance in terms of balanced precision and recall for both classes.
- Random Forest stands out with the highest accuracy and AUC, indicating its effectiveness in classification after the balancing technique.
- Logistic Regression, Naive Bayes, and KNN also show competitive performance, maintaining relatively high accuracy and balanced precision-recall metrics.
- Decision Tree exhibits slightly lower performance compared to other classifiers but still maintains a reasonable level of accuracy and AUC.

2. Class Weighting Method

After applying class weights to address the class imbalance issue, let's evaluate the performance of the classifiers:

Logistic Regression

- **Accuracy**: 0.7612
- **AUC**: 0.8598
- **F1 Score (Positive Class)**: 0.6599
- **F1 Score (Negative Class)**: 0.8160
- **Precision-Recall Trade-off**: The precision for the positive class improved slightly, while recall decreased, indicating a trade-off.
- **Confusion Matrix**:
  - True Negatives (TN): 745
  - False Positives (FP): 251
  - False Negatives (FN): 85
  - True Positives (TP): 326

Naive Bayes

- **Accuracy**: 0.7591

- **AUC**: 0.8434
- **F1 Score (Positive Class)**: 0.6530
- **F1 Score (Negative Class)**: 0.8155
- **Precision-Recall Trade-off**: Precision for the positive class improved slightly, while recall decreased.
- **Confusion Matrix**:
    - True Negatives (TN): 749
    - False Positives (FP): 247
    - False Negatives (FN): 92
    - True Positives (TP): 319

K-Nearest Neighbors (KNN)

- **Accuracy**: 0.7520
- **AUC**: 0.7665
- **F1 Score (Positive Class)**: 0.5315
- **F1 Score (Negative Class)**: 0.8313
- **Precision-Recall Trade-off**: Precision for the positive class improved, while recall decreased.
- **Confusion Matrix**:
    - True Negatives (TN): 860
    - False Positives (FP): 136
    - False Negatives (FN): 213
    - True Positives (TP): 198

Decision Tree

- **Accuracy**: 0.7335
- **AUC**: 0.6649
- **F1 Score (Positive Class)**: 0.5223
- **F1 Score (Negative Class)**: 0.8152
- **Precision-Recall Trade-off**: Both precision and recall decreased slightly for the positive class.
- **Confusion Matrix**:
    - True Negatives (TN): 827
    - False Positives (FP): 169
    - False Negatives (FN): 206
    - True Positives (TP): 205

Random Forest

- **Accuracy**: 0.7882
- **AUC**: 0.8285
- **F1 Score (Positive Class)**: 0.5694
- **F1 Score (Negative Class)**: 0.8596
- **Precision-Recall Trade-off**: Precision for the positive class improved slightly, while recall decreased.
- **Confusion Matrix**:
    - True Negatives (TN): 912
    - False Positives (FP): 84
    - False Negatives (FN): 214
    - True Positives (TP): 197

Summary and Considerations

- Applying class weights improved the performance metrics for the positive class in terms of precision, although it led to a decrease in recall for most classifiers.
- Logistic Regression and Naive Bayes show relatively balanced precision and recall after applying class weights.
- K-Nearest Neighbors and Random Forest demonstrate improved precision for the positive class, but at the cost of reduced recall.
- Decision Tree exhibits a decrease in both precision and recall for the positive class after applying class weights.

3. One-class learning

Classifier: Logistic Regression

- **Accuracy:** Improved from 0.7317 to 0.9825
- **AUC:** Improved from 0.8338 to 0.8595
- **F1 Score (Positive):** Improved from 0.0976 to 0.0000
- **F1 Score (Negative):** Improved from 0.8424 to 0.9912

The one-class learning approach significantly enhanced the AUC for Logistic Regression, indicating improved overall performance in distinguishing between the two classes, although the accuracy and F1 Score for the positive class remained at zero, indicating continued challenges in correctly identifying instances of the minority class.

Classifier: Naive Bayes

- **Accuracy:** Improved from 0.9627 to 0.9816
- **AUC:** Improved from 0.8306 to 0.8526
- **F1 Score (Positive):** Remained at 0.0000
- **F1 Score (Negative):** Improved from 0.9810 to 0.9907

Similar to Logistic Regression, Naive Bayes also saw notable improvements in AUC, reflecting better performance in distinguishing between classes. However, the F1 Score for the positive class remained unchanged at zero.

Classifier: KNN

- **Accuracy:** Remained at 0.9812
- **AUC:** Improved from 0.5370 to 0.5606
- **F1 Score (Positive):** Remained at 0.0000
- **F1 Score (Negative):** Remained at 0.9912

KNN did not exhibit significant improvements with one-class learning, with the AUC showing only a marginal increase.

Classifier: Decision Tree

- **Accuracy:** Improved from 0.9661 to 0.9637
- **AUC:** Remained at 0.5225
- **F1 Score (Positive):** Improved from 0.0637 to 0.0429
- **F1 Score (Negative):** Improved from 0.9828 to 0.9815

Decision Tree showed a slight decrease in accuracy but saw improvements in the F1 Score for both positive and negative classes.

Classifier: Random Forest

- **Accuracy:** Remained at 0.9813
- **AUC:** Improved from 0.7994 to 0.7758
- **F1 Score (Positive):** Remained at 0.0000
- **F1 Score (Negative):** Improved from 0.9906 to 0.9911

Random Forest did not demonstrate significant improvements in AUC with one-class learning.

In summary, one-class learning, while showing improvements in AUC for Logistic Regression and Naive Bayes, did not consistently enhance the classifiers' ability to correctly identify instances of the minority class.

1. **Stoke Prediction**

BASELINE MODEL:-

After evaluating the baseline model, here are the results:

Logistic Regression:

- **Accuracy**: 0.9812
- **AUC**: 0.6990
- **F1 Score (Positive Class)**: 0.0000
- **F1 Score (Negative Class)**: 0.9905
- **Precision-Recall Trade-off**: Precision for the positive class is 0, indicating that the model didn't predict any true positives. Recall for the positive class is also 0.
- **Confusion Matrix**:
    - True Negatives (TN): 8517
    - False Positives (FP): 1
    - False Negatives (FN): 162
    - True Positives (TP): 0

Naive Bayes:

- **Accuracy**: 0.9627
- **AUC**: 0.8306
- **F1 Score (Positive Class)**: 0.0636
- **F1 Score (Negative Class)**: 0.9810
- **Precision-Recall Trade-off**: Precision and recall for the positive class are low, indicating a weak performance in predicting true positives.
- **Confusion Matrix**:
    - True Negatives (TN): 8345
    - False Positives (FP): 173
    - False Negatives (FN): 151
    - True Positives (TP): 11

K-Nearest Neighbors (KNN):

- **Accuracy**: 0.9812
- **AUC**: 0.5370
- **F1 Score (Positive Class)**: 0.0000
- **F1 Score (Negative Class)**: 0.9905
- **Precision-Recall Trade-off**: Similar to Logistic Regression, KNN also didn't predict any true positives, resulting in a precision and recall of 0 for the positive class.
- **Confusion Matrix**:
    - True Negatives (TN): 8517
    - False Positives (FP): 1
    - False Negatives (FN): 162
    - True Positives (TP): 0

Decision Tree:

- **Accuracy**: 0.9627
- **AUC**: 0.5359
- **F1 Score (Positive Class)**: 0.0847
- **F1 Score (Negative Class)**: 0.9809
- **Precision-Recall Trade-off**: Precision and recall for the positive class are relatively low, indicating a weak performance in predicting true positives.
- **Confusion Matrix**:
    - True Negatives (TN): 8341
    - False Positives (FP): 177
    - False Negatives (FN): 147
    - True Positives (TP): 15

Random Forest:

- **Accuracy**: 0.9813
- **AUC**: 0.8091
- **F1 Score (Positive Class)**: 0.0000
- **F1 Score (Negative Class)**: 0.9906
- **Precision-Recall Trade-off**: Similar to Logistic Regression and KNN, Random Forest didn't predict any true positives, resulting in a precision and recall of 0 for the positive class.
- **Confusion Matrix**:
    - True Negatives (TN): 8518
    - False Positives (FP): 0
    - False Negatives (FN): 162
    - True Positives (TP): 0

Summary:

- The baseline models, particularly Logistic Regression, KNN, and Random Forest, failed to predict any instances of the positive class, resulting in an F1 score and AUC of 0 for the positive class.
- Naive Bayes and Decision Tree showed slightly better performance in predicting the positive class, but their precision and recall are still relatively low.
- Improvements are needed to address the imbalance in the dataset and enhance the models' ability to detect the positive class.

1. Smote Technique

After applying SMOTE (Synthetic Minority Over-sampling Technique) to address class imbalance, here are the updated evaluation metrics for each classifier:

Logistic Regression:

- **Accuracy**: 0.7885
- **AUC**: 0.8604
- **F1 Score (Positive Class)**: 0.7977
- **F1 Score (Negative Class)**: 0.7785
- **Precision-Recall Trade-off**: The model shows balanced precision and recall for both classes.
- **Confusion Matrix**:
    - True Negatives (TN): 6335
    - False Positives (FP): 2190
    - False Negatives (FN): 1415
    - True Positives (TP): 7107

Naive Bayes:

- **Accuracy**: 0.8031
- **AUC**: 0.8728
- **F1 Score (Positive Class)**: 0.8227
- **F1 Score (Negative Class)**: 0.7785
- **Precision-Recall Trade-off**: The model shows slightly better recall for the positive class but lower precision compared to Logistic Regression.
- **Confusion Matrix**:
    - True Negatives (TN): 5901
    - False Positives (FP): 2624
    - False Negatives (FN): 733
    - True Positives (TP): 7789

K-Nearest Neighbors (KNN):

- **Accuracy**: 0.8842
- **AUC**: 0.9441
- **F1 Score (Positive Class)**: 0.8921
- **F1 Score (Negative Class)**: 0.8750
- **Precision-Recall Trade-off**: KNN performs well with high precision and recall for both classes.
- **Confusion Matrix**:
    - True Negatives (TN): 6909
    - False Positives (FP): 1616
    - False Negatives (FN): 358
    - True Positives (TP): 8164

Decision Tree:

- **Accuracy**: 0.9358
- **AUC**: 0.9358
- **F1 Score (Positive Class)**: 0.9367
- **F1 Score (Negative Class)**: 0.9349
- **Precision-Recall Trade-off**: Balanced precision and recall for both classes, indicating good performance.

- **Confusion Matrix**:
    - True Negatives (TN): 7852
    - False Positives (FP): 673
    - False Negatives (FN): 421
    - True Positives (TP): 8101

Random Forest:

- **Accuracy**: 0.9563
- **AUC**: 0.9938
- **F1 Score (Positive Class)**: 0.9574
- **F1 Score (Negative Class)**: 0.9551
- **Precision-Recall Trade-off**: Random Forest exhibits excellent performance with high precision and recall for both classes.
- **Confusion Matrix**:
    - True Negatives (TN): 7930
    - False Positives (FP): 595
    - False Negatives (FN): 150
    - True Positives (TP): 8372

Overall, applying SMOTE has significantly improved the performance of all classifiers, especially in terms of their ability to predict the positive class. Random Forest, in particular, stands out with its high accuracy, AUC, and F1 scores for both classes.

2. Class weights method:

Logistic Regression:

- **Accuracy:** 0.7317
- **AUC:** 0.8338
- **F1 Score (Positive):** 0.0976
- **F1 Score (Negative):** 0.8424
- **Classification Report:**
    - Precision, recall, and F1-score for both classes are provided, along with support.
- **Confusion Matrix:**
    - Provides a breakdown of true positive, true negative, false positive, and false negative predictions.

Naive Bayes:

- **Accuracy:** 0.9627
- **AUC:** 0.8306
- **F1 Score (Positive):** 0.0636
- **F1 Score (Negative):** 0.9810
- **Classification Report:**
    - Precision, recall, and F1-score for both classes are provided, along with support.
- **Confusion Matrix:**
    - Provides a breakdown of true positive, true negative, false positive, and false negative predictions.

KNN:

- **Accuracy:** 0.9812
- **AUC:** 0.5370
- **F1 Score (Positive):** 0.0000
- **F1 Score (Negative):** 0.9905
- **Classification Report:**
    - Precision, recall, and F1-score for both classes are provided, along with support.
- **Confusion Matrix:**
    - Provides a breakdown of true positive, true negative, false positive, and false negative predictions.

Decision Tree:

- **Accuracy:** 0.9661
- **AUC:** 0.5225
- **F1 Score (Positive):** 0.0637
- **F1 Score (Negative):** 0.9828
- **Classification Report:**
    - Precision, recall, and F1-score for both classes are provided, along with support.
- **Confusion Matrix:**
    - Provides a breakdown of true positive, true negative, false positive, and false negative predictions.

Random Forest:

- **Accuracy:** 0.9813
- **AUC:** 0.7994
- **F1 Score (Positive):** 0.0000
- **F1 Score (Negative):** 0.9906
- **Classification Report:**
    - Precision, recall, and F1-score for both classes are provided, along with support.
- **Confusion Matrix:**
    - Provides a breakdown of true positive, true negative, false positive, and false negative predictions.

Logistic Regression showed the most noticeable improvement after applying class weights, especially in recall for the minority class. This suggests that class weighting can help Logistic Regression better handle imbalanced datasets

3.One-class learning

Classifier: Logistic Regression

- **Accuracy:** Improved from 0.7317 to 0.9825
- **AUC:** Improved from 0.8338 to 0.8595
- **F1 Score (Positive):** Improved from 0.0976 to 0.0000
- **F1 Score (Negative):** Improved from 0.8424 to 0.9912

The one-class learning approach significantly enhanced the AUC for Logistic Regression, indicating improved overall performance in distinguishing between the two classes, although the accuracy and F1 Score for the positive class remained at zero, indicating continued challenges in correctly identifying instances of the minority class.

Classifier: Naive Bayes

- **Accuracy:** Improved from 0.9627 to 0.9816
- **AUC:** Improved from 0.8306 to 0.8526
- **F1 Score (Positive):** Remained at 0.0000
- **F1 Score (Negative):** Improved from 0.9810 to 0.9907

Similar to Logistic Regression, Naive Bayes also saw notable improvements in AUC, reflecting better performance in distinguishing between classes. However, the F1 Score for the positive class remained unchanged at zero.

Classifier: KNN

- **Accuracy:** Remained at 0.9812
- **AUC:** Improved from 0.5370 to 0.5606
- **F1 Score (Positive):** Remained at 0.0000
- **F1 Score (Negative):** Remained at 0.9912

KNN did not exhibit significant improvements with one-class learning, with the AUC showing only a marginal increase.

Classifier: Decision Tree

- **Accuracy:** Improved from 0.9661 to 0.9637
- **AUC:** Remained at 0.5225
- **F1 Score (Positive):** Improved from 0.0637 to 0.0429
- **F1 Score (Negative):** Improved from 0.9828 to 0.9815

Decision Tree showed a slight decrease in accuracy but saw improvements in the F1 Score for both positive and negative classes.

Classifier: Random Forest

- **Accuracy:** Remained at 0.9813
- **AUC:** Improved from 0.7994 to 0.7758
- **F1 Score (Positive):** Remained at 0.0000
- **F1 Score (Negative):** Improved from 0.9906 to 0.9911

Random Forest did not demonstrate significant improvements in AUC with one-class learning.

In summary, one-class learning, while showing improvements in AUC for Logistic Regression and Naive Bayes, did not consistently enhance the classifiers' ability to correctly identify instances of the minority class.

Q3: Does the performance of a CI solution get impacted significantly by the choice of different algorithms (compare for both baseline and CI-based)?

Yes, the performance of a classification solution with various techniques such as SMOTE (Synthetic Minority Over-sampling Technique), class weights, and one-class learning can indeed be significantly impacted by the choice of different algorithms, both in baseline and when incorporating these techniques.

Let's break down the impact on performance for each technique:

1. **Baseline Algorithms:**
    - Different algorithms have varying inherent capabilities to handle imbalanced datasets. For instance, decision trees and random forests tend to perform well on imbalanced data because they can naturally learn to give more weight to minority classes. On the other hand, algorithms like logistic regression and k-nearest neighbors might struggle with imbalanced data, as they could be biased towards the majority class.
    - The choice of algorithm can impact metrics like accuracy, precision, recall, and F1-score differently for both the majority and minority classes.
2. **SMOTE (or similar oversampling techniques):**
    - SMOTE aims to alleviate the class imbalance by generating synthetic samples for the minority class. The effectiveness of SMOTE can vary depending on the distribution and characteristics of the dataset.
    - Some algorithms might be more sensitive to the introduction of synthetic samples. For example, while decision trees and random forests can handle synthetic samples well, linear models like logistic regression might not generalize as effectively.
    - Therefore, the choice of algorithm can impact how well SMOTE improves the performance of the classifier.
3. **Class Weights:**
    - Adjusting class weights is a common technique to handle class imbalance, where the algorithm assigns higher penalties to misclassifications of the minority class.
    - Certain algorithm like logistic regression allow for easy integration of class weights. However, the effectiveness of class weights can vary depending on how well the algorithm can adjust its decision boundaries to accommodate the weights.
    - Again, the choice of algorithm can impact how effectively class weights address the class imbalance issue.
4. **One-Class Learning:**
    - One-class learning is specifically designed for scenarios where only one class is represented in the training data. It aims to learn the distribution of the majority class and identify deviations as anomalies.
    - As observed in our example, the performance of one-class learning can vary across different algorithms. Logistic regression and naive Bayes showed improvements in AUC but struggled to correctly identify instances of the minority class. Other algorithms may exhibit different behaviors when trained using one-class learning.

In conclusion, the choice of algorithm can significantly impact the performance of classification solutions with or without techniques like SMOTE, class weights, or one-class learning. It's essential to experiment with different algorithms and techniques to find the best combination for a particular dataset and classification task.

ML Score Card

1. Credit Card Detection Dataset

	Before Sampling	After SMOTE	Class Weights	One Class learning Anomaly Detection
Logistic Regression	0.9768	0.9768	0.9483	0.9987
Naive Bayes	0.9777	0.9777	0.9137	0.9866
KNN	0.9994	0.9994	0.999	0.9988
Decision Tree	0.9991	0.9991	0.9983	0.9979
Random Forest	0.9995	0.9995	0.9999	0.9988

2. Customer Churn Dataset

	Before Sampling	After SMOTE	Class Weights	Anomaly Detection
Logistic Regression	0.8033	0.8033	0.8033	0.8033
Naive Bayes	0.7889	0.7889	0.7889	0.7889
KNN	0.7703	0.7703	0.7703	0.7703
Decision Tree	0.7258	0.7258	0.7258	0.7258
Random Forest	0.7753	0.7753	0.7753	0.7753

3. Stoke Prediction

Classifier	Before Sampling	After SMOTE	Class Weights	Anomaly Detection
Logistic Regression	0.9825	0.9825	0.7317	0.7317
Naive Bayes	0.9816	0.9627	0.9627	0.9627
KNN	0.9825	0.9825	0.9825	0.9825
Decision Tree	0.9637	0.9661	0.9661	0.9637
Random Forest	0.9824	0.9813	0.9813	0.9824