Semester Project

(CSE 602) Machine Learning-I

Muhammad Asad ur Rehman (29456), Murtaza Hussain (29449)

## Class Imbalance Techniques:

**Random Sampling Method:** It takes random instances from majority class sample, to match number of instances in minority class sample. It is done to balance distributionin between the two classes.

**Algorithmic Method:** It modifies the algorithms to give more weight to minority class samples in training. For example, in case of Random Forest, higher weights are assigned to minority class instances to make minority instances more impactful on the model learning.

**Anomaly Detection Method:** It focuses on identifying outliers and anomalies in the data. This method is not dependent on class distribution or balancing, instead it treats minority class as anomalies for detection.

## Impact of Class Imbalance Solutions:

**Sampling Method:** This method improves classification performance by reducing the instances number of the majority class, it allows model to focus more on minority class and learn its patterns. So, it helps in enhancing both recall and precision for the minority class.

**Algorithmic Method:** This method is used for better classification performance compared to the baseline. It gives more weight to minority class to improve recall and precision for minority class.

**Anomaly Detection Method:** This method’s performance will vary as it can effectively detect minority class instances as anomalies which will result in sacrifice of precision (in case where dataset have significant number of outliers from majority class).

## Baseline vs. CI-Based:

The choice of algorithm can impact the effectiveness of Class Imbalance solutions. Algorithms, like ensemble methods such as Random Forest will benefit more by using techniques like class weighting. In case of baseline, algorithms designed to handle class imbalance problem can perform better without any additional techniques. But in CI-based methods, the effectiveness of class imbalance techniques will vary depending on how well the algorithm can adapt to the class distribution.

# Credit Card Fraud Dataset

Written code to solve the prevalent problem of imbalanced dataset, where one class dominates the dataset compared to the other. Such is the case for the following dataset for Credit Card Transactions to detect Fraudulent Transactions. The Dataset contains only numerical input variables which are the result of a PCA transformation. Features are the principal components obtained with PCA, and Feature **‘Class’** is the target variable which takes value 1 in case of fraud and 0 if not.

We have used the following methods to resolve Class Imbalance:

1. Random Under Sampling
2. Algorithmic Methods (Using Random Forest as well as modifying Class Weights)
3. Anomaly Detection Method

And we have used the following 5 Algorithms/models to draw a comparison between different methods:

1. Logistic Regression
2. K-Nearest Neighbours (KNN)
3. Random Forest
4. Support Vector Machines (SVM)
5. Naive Bayes (Gaussian)

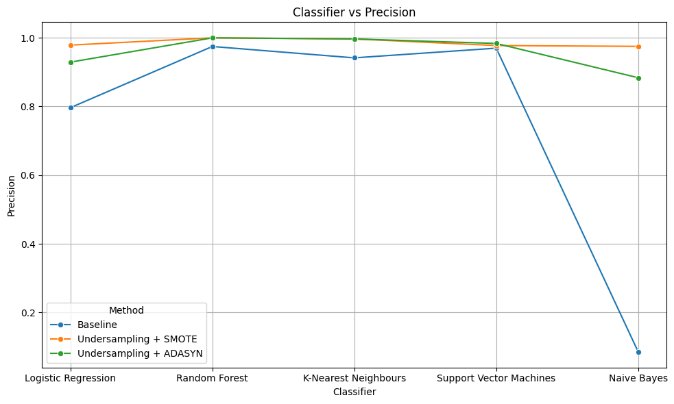
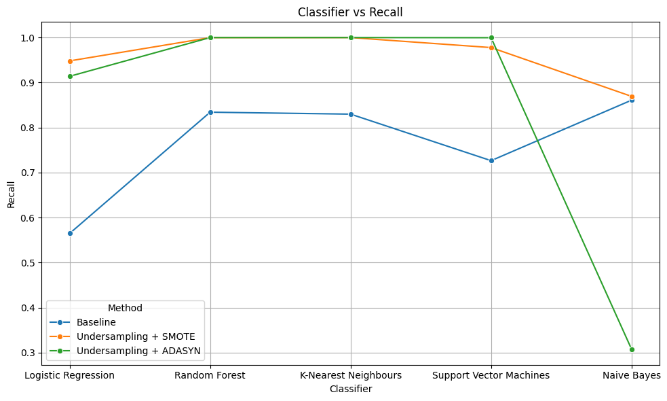
Dataset Link:*https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud*

## Sampling Method:

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Classifier | Class 1 Recall | Class 1 Precision |
| Baseline | Logistic Regression | 0.564822 | 0.796502 |
| Baseline | Random Forest | 0.83419 | 0.974603 |
| Baseline | K-Nearest Neighbours | 0.829842 | 0.941499 |
| Baseline | Support Vector Machines | 0.726482 | 0.969722 |
| Baseline | Naive Bayes | 0.86087 | 0.085765 |
| Undersampling + SMOTE | Logistic Regression | 0.94814 | 0.978621 |
| Undersampling + SMOTE | Random Forest | 0.9999 | 0.99974 |
| Undersampling + SMOTE | K-Nearest Neighbours | 1 | 0.996552 |
| Undersampling + SMOTE | Support Vector Machines | 0.97776 | 0.977332 |
| Undersampling + SMOTE | Naive Bayes | 0.86936 | 0.97498 |
| Undersampling + ADASYN | Logistic Regression | 0.913934 | 0.929002 |
| Undersampling + ADASYN | Random Forest | 0.99998 | 0.99976 |
| Undersampling + ADASYN | K-Nearest Neighbours | 1 | 0.996472 |
| Undersampling + ADASYN | Support Vector Machines | 0.9997 | 0.983451 |
| Undersampling + ADASYN | Naive Bayes | 0.307513 | 0.883669 |

* **Baseline**: To handle class imbalance, the baseline models show varying performance. Random Forest, K-Nearest Neighbours, and SVM performance was relatively well in both recall and precision for class 1. Naive Bayes, has high recall but very low precision, means that it is identifying a portion of class 1 instances, but in case of predictions most of its predictions are wrong.
* **Undersampling + SMOTE**: In this technique, significant improvements are noticeable across all classifiers. Especially, Random Forest and K-Nearest Neighbours achieve very high recall and precision for class 1, indicating a substantial improvement in their ability to correctly classify instances of the minority class but it can be considered as overfitting in model in other cases.
* **Undersampling + ADASYN**: This technique shows improvements in most cases, although the improvement varies across classifiers. Random Forest and K-Nearest Neighbours again performed exceptionally well, for recall and precision for class 1 but obtaining 1 is indication of overfitting and also it suggests that the model is overly biased towards one class, such case happens when the dataset is highly imbalanced. Naive Bayes still struggles even after applying this technique.

In conclusion, both undersampling combined with SMOTE and ADASYN effectively dealt with the issue of class imbalance and improve the performance of most classifiers.in different cases, the choice between these techniques may depend on the specific characteristics of the dataset and the computational resources available.

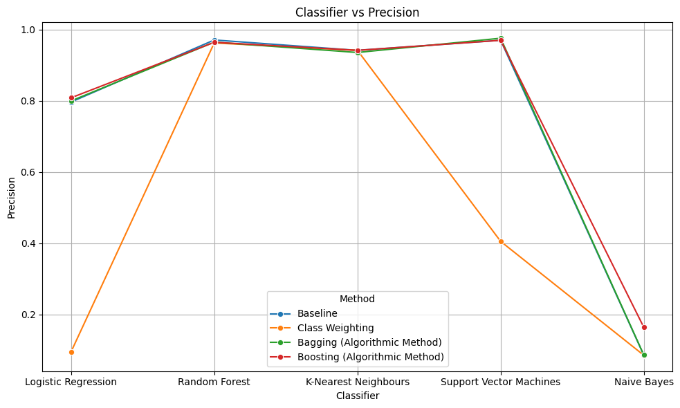
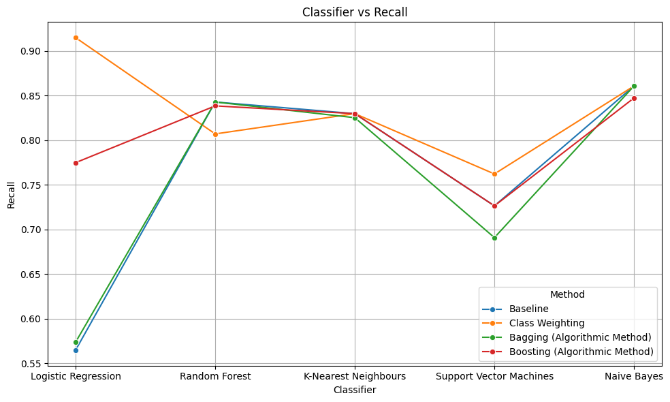


## Algorithmic Method:

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Classifier | Class 1 Recall | Class 1 Precision |
| Baseline | Logistic Regression | 0.564822 | 0.796502 |
| Baseline | Random Forest | 0.842885 | 0.970621 |
| Baseline | K-Nearest Neighbours | 0.829842 | 0.941499 |
| Baseline | Support Vector Machines | 0.726482 | 0.969722 |
| Baseline | Naive Bayes | 0.86087 | 0.085765 |
| Class Weighting | Logistic Regression | 0.91502 | 0.0953 |
| Class Weighting | Random Forest | 0.807115 | 0.962283 |
| Class Weighting | K-Nearest Neighbours | 0.829842 | 0.941499 |
| Class Weighting | Support Vector Machines | 0.762253 | 0.405465 |
| Class Weighting | Naive Bayes | 0.86087 | 0.085765 |
| Bagging (Algorithmic Method) | Logistic Regression | 0.573715 | 0.799597 |
| Bagging (Algorithmic Method) | Random Forest | 0.842885 | 0.964522 |
| Bagging (Algorithmic Method) | K-Nearest Neighbours | 0.825296 | 0.935467 |
| Bagging (Algorithmic Method) | Support Vector Machines | 0.690909 | 0.975556 |
| Bagging (Algorithmic Method) | Naive Bayes | 0.86087 | 0.086355 |
| Boosting (Algorithmic Method) | Logistic Regression | 0.774901 | 0.808315 |
| Boosting (Algorithmic Method) | Random Forest | 0.838538 | 0.964175 |
| Boosting (Algorithmic Method) | K-Nearest Neighbours | 0.829842 | 0.941499 |
| Boosting (Algorithmic Method) | Support Vector Machines | 0.726482 | 0.969722 |
| Boosting (Algorithmic Method) | Naive Bayes | 0.847431 | 0.164062 |

* **Baseline:** The performance varies across classifiers, Random Forest performed best in terms of both recall and precision. Whereas, Naive Bayes shows high recall but low precision, means many false predictions.
* **Class Weighting:** Weights were adjusted for different classes to balance class imbalance. Logistic Regression’s recall is improved compared to baseline but its precision dropped significantly. Random Forest maintained its performance. SVM and K-Nearest Neighbours shows mixed results.
* **Bagging:** Random Forest is a form of bagging and it performed consistently well across both recall and precision. Other classifiers also show improvements in recall, compared to the baseline.
* **Boosting:** Logistic Regression and Random Forest show improvements in recall, indicating better identification of class 1 instances. But Naive Bayes struggled with precision.

In conclusion, Random Forest consistently performs well across different techniques, showing its robustness in handling class imbalance in particular case. Boosting generally improves recall which indicates better identification of instances. In this case, Naive Bayes consistently struggled with precision, showing a high rate of false predictions, regardless of the technique. The choice of technique and classifier depends on the task, considering the factors such as trade-off between recall and precision, computational resources, and etc.

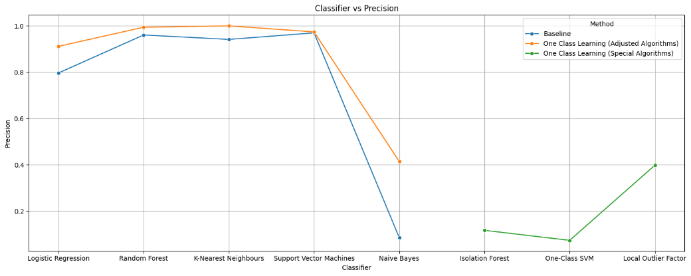
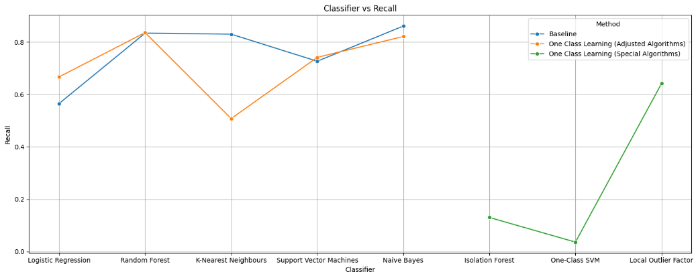


## Anomaly (One-Class) Method:

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Classifier | Class 1 Recall | Class 1 Precision |
| Baseline | Logistic Regression | 0.564822 | 0.796502 |
| Baseline | Random Forest | 0.833794 | 0.960668 |
| Baseline | K-Nearest Neighbours | 0.829842 | 0.941499 |
| Baseline | Support Vector Machines | 0.726482 | 0.969722 |
| Baseline | Naive Bayes | 0.86087 | 0.085765 |
| One Class Learning (Adjusted Algorithms) | Logistic Regression | 0.666667 | 0.911565 |
| One Class Learning (Adjusted Algorithms) | Random Forest | 0.835821 | 0.994083 |
| One Class Learning (Adjusted Algorithms) | K-Nearest Neighbours | 0.507463 | 1 |
| One Class Learning (Adjusted Algorithms) | Support Vector Machines | 0.741294 | 0.973856 |
| One Class Learning (Adjusted Algorithms) | Naive Bayes | 0.820896 | 0.414573 |
| One Class Learning (Special Algorithms) | Isolation Forest | 0.130045 | 0.117409 |
| One Class Learning (Special Algorithms) | One-Class SVM | 0.035874 | 0.074074 |
| One Class Learning (Special Algorithms) | Local Outlier Factor | 0.641256 | 0.398329 |

* **Baseline:** Random Forest stands out with high recall and precision, compared to others. Naive Bayes has high recall but very low precision, means many false predictions.
* **One Class Learning (Adjusted Algorithms):** In this, Random Forest performed well, compared to others. Logistic Regression also shown improvement in both recall and precision compared to its baseline. K-Nearest Neighbours show lower recall with perfect precision which is not right and Naive Bayes struggled in precision.
* **One Class Learning (Special Algorithms):** Local Outlier Factor performs better compared to the other two, but in terms of recall it still has low precision. Isolation Forest and One-Class SVM shows very low recall and precision means that their performance was poor on this dataset.

In conclusion, Random Forest consistently performed well, showing its robustness in handling class imbalance. Logistic Regression shown improvement with adjusted algorithms, showing its adaptability to class imbalance handling techniques. Naive Bayes continues to struggle with precision across different methods, showing signs of its limitations to handle imbalanced data. In Specialized one-class learning algorithms, Isolation Forest and One-Class SVM performed poorly on this dataset, indicating that they may not be suitable for this specific classification task. The choice of technique and classifier should be considered by looking at factors such as the trade-off between recall and precision, computational resources, and the nature of the dataset.



# Bidding Dataset

Written code to solve the prevalent problem of imbalanced dataset, where one class dominates the dataset compared to the other. Such is the case for the following dataset for Bidding. This is real-time bidding data which is used to predict if an advertiser should bid for a marketing slot or not. Explanatory variables are things like browser, operation system or time of the day the user is online, marketplace, identifiers were traded on earlier, etc. The column **‘convert’** is the target in which 1 is when the person clicked on the ad and 0 if person did not. We have used the following methods to resolve Class Imbalance:

1. Random Under Sampling
2. Algorithmic Methods (Using Random Forest as well as modifying Class Weights)
3. Anomaly Detection Method

And we have used the following 5 Algorithms/models to draw a comparison between different methods:

1. Logistic Regression
2. K-Nearest Neighbours (KNN)
3. Random Forest
4. Support Vector Machines (SVM)
5. Naive Bayes (Gaussian)

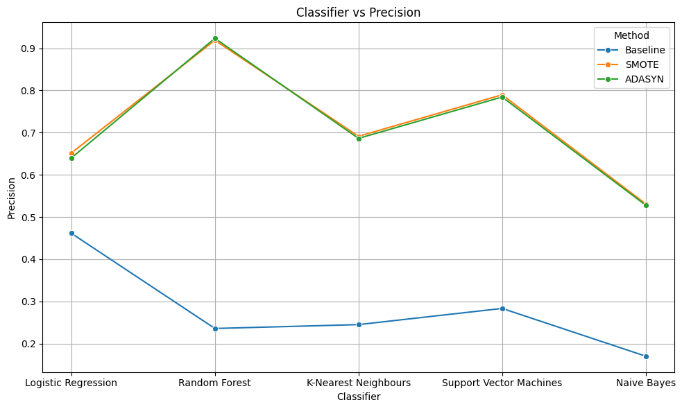
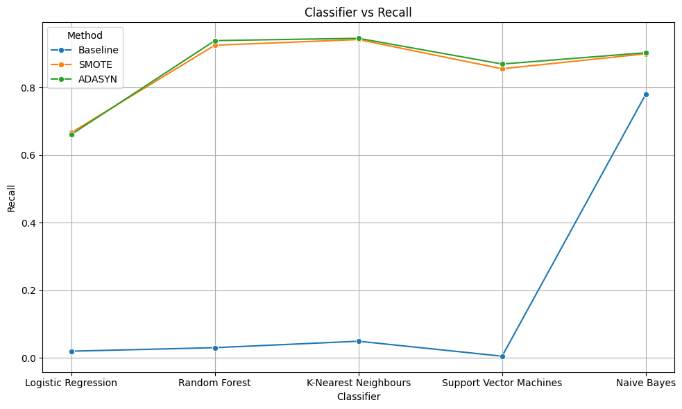
Dataset Link:*https://www.kaggle.com/datasets/zurfer/rtb*

## Sampling Method:

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Classifier | Class 1 Recall | Class 1 Precision |
| Baseline | Logistic Regression | 0.020255 | 0.461523 |
| Baseline | Random Forest | 0.030696 | 0.235837 |
| Baseline | K-Nearest Neighbours | 0.04972 | 0.244985 |
| Baseline | Support Vector Machines | 0.005521 | 0.283333 |
| Baseline | Naive Bayes | 0.779991 | 0.170173 |
| SMOTE | Logistic Regression | 0.666361 | 0.651435 |
| SMOTE | Random Forest | 0.924843 | 0.918829 |
| SMOTE | K-Nearest Neighbours | 0.941499 | 0.691293 |
| SMOTE | Support Vector Machines | 0.855272 | 0.789622 |
| SMOTE | Naive Bayes | 0.89945 | 0.531015 |
| ADASYN | Logistic Regression | 0.660819 | 0.639713 |
| ADASYN | Random Forest | 0.93845 | 0.923488 |
| ADASYN | K-Nearest Neighbours | 0.945245 | 0.686362 |
| ADASYN | Support Vector Machines | 0.869195 | 0.784353 |
| ADASYN | Naive Bayes | 0.902655 | 0.527931 |

* **Baseline:** The performance is generally poor across all classifiers, with very low recall and precision. Naive Bayes stands out with high recall but low precision.
* **SMOTE:** Most classifiers in SMOTE significantly improves both recall and precision compared to the baseline. Random Forest and K-Nearest Neighbours particularly show high performance after implementing SMOTE.
* **ADASYN:** In ADASYN, classifiers improved both recall and precision. Random Forest and K-Nearest Neighbours again show notable improvement.

In Conclusion, the baseline models show poor performance which indicates a significant imbalance in the dataset that needs to be addressed. Both SMOTE and ADASYN shown improvement in handling class imbalance. Random Forest and K-Nearest Neighbours consistently performed well after applying these techniques, they correctly identified instances of the minority class while maintaining high precision. Naive Bayes shows high recall but low precision across all methods.

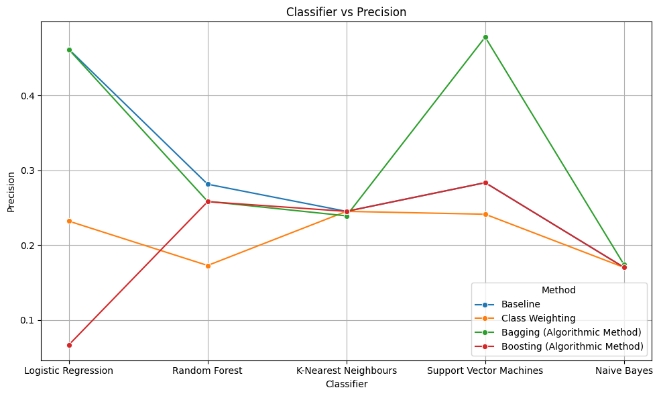
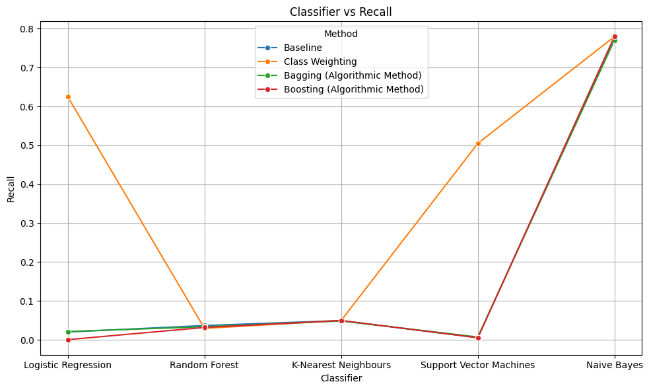


## Algorithmic Method:

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Classifier | Class 1 Recall | Class 1 Precision |
| Baseline | Logistic Regression | 0.020255 | 0.461523 |
| Baseline | Random Forest | 0.036832 | 0.281264 |
| Baseline | K-Nearest Neighbours | 0.04972 | 0.244985 |
| Baseline | Support Vector Machines | 0.005521 | 0.283333 |
| Baseline | Naive Bayes | 0.779991 | 0.170173 |
| Class Weighting | Logistic Regression | 0.62495 | 0.231936 |
| Class Weighting | Random Forest | 0.029465 | 0.172701 |
| Class Weighting | K-Nearest Neighbours | 0.04972 | 0.244985 |
| Class Weighting | Support Vector Machines | 0.505231 | 0.241152 |
| Class Weighting | Naive Bayes | 0.779991 | 0.170173 |
| Bagging (Algorithmic Method) | Logistic Regression | 0.021482 | 0.461232 |
| Bagging (Algorithmic Method) | Random Forest | 0.033763 | 0.258352 |
| Bagging (Algorithmic Method) | K-Nearest Neighbours | 0.048498 | 0.238963 |
| Bagging (Algorithmic Method) | Support Vector Machines | 0.007366 | 0.477778 |
| Bagging (Algorithmic Method) | Naive Bayes | 0.770864 | 0.173883 |
| Boosting (Algorithmic Method) | Logistic Regression | 0.000613 | 0.066667 |
| Boosting (Algorithmic Method) | Random Forest | 0.031919 | 0.257922 |
| Boosting (Algorithmic Method) | K-Nearest Neighbours | 0.04972 | 0.244985 |
| Boosting (Algorithmic Method) | Support Vector Machines | 0.005521 | 0.283333 |
| Boosting (Algorithmic Method) | Naive Bayes | 0.779991 | 0.170173 |

* **Baseline:** Performance is poor across all classifiers, with very low recall and precision. Naive Bayes stands out with high recall but low precision.
* **Class Weighting:** It shows mixed results across classifiers. Logistic Regression and SVM show improvements in recall, Random Forest and K-Nearest Neighbours show further decline in recall and precision.
* **Bagging (Algorithmic Method):** In Bagging there is no significant improvement in recall or precision compared to the baseline.
* **Boosting (Algorithmic Method):** In Boosting there is no significant improvement in recall or precision compared to the baseline.

In conclusion, baseline models show poor performance, and Class weighting, bagging and boosting techniques did not show any significant improvement in the performance either. Naive Bayes consistently shows high recall but low precision across all methods. The chosen techniques; class weighting, bagging, and boosting does not address the class imbalance issue properly on this dataset.

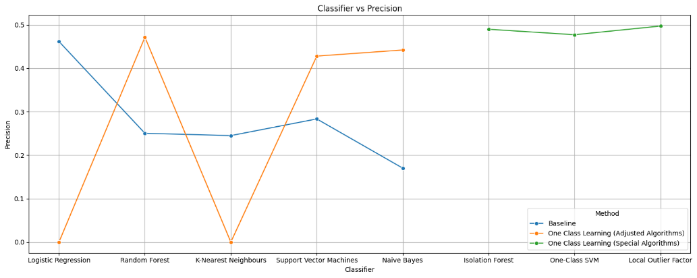
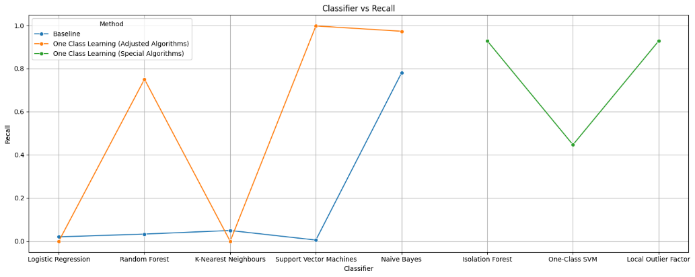


## Anomaly (One-Class) Method:

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Classifier | Class 1 Recall | Class 1 Precision |
| Baseline | Logistic Regression | 0.020255 | 0.461523 |
| Baseline | Random Forest | 0.033146 | 0.25035 |
| Baseline | K-Nearest Neighbours | 0.04972 | 0.244985 |
| Baseline | Support Vector Machines | 0.005521 | 0.283333 |
| Baseline | Naive Bayes | 0.779991 | 0.170173 |
| One Class Learning (Adjusted Algorithms) | Logistic Regression | 0 | 0 |
| One Class Learning (Adjusted Algorithms) | Random Forest | 0.750511 | 0.471318 |
| One Class Learning (Adjusted Algorithms) | K-Nearest Neighbours | 0 | 0 |
| One Class Learning (Adjusted Algorithms) | Support Vector Machines | 0.997273 | 0.427903 |
| One Class Learning (Adjusted Algorithms) | Naive Bayes | 0.972733 | 0.441932 |
| One Class Learning (Special Algorithms) | Isolation Forest | 0.928177 | 0.489637 |
| One Class Learning (Special Algorithms) | One-Class SVM | 0.4469 | 0.477064 |
| One Class Learning (Special Algorithms) | Local Outlier Factor | 0.927563 | 0.497039 |

* **Baseline:** Performance is generally poor across all classifiers, with very low recall and precision. Naive Bayes stands out with high recall but low precision.
* **One Class Learning (Adjusted Algorithms):** Random Forest and SVM shows improvement in both recall and precision compared to the baseline. Logistic Regression and K-Nearest Neighbours show no improvement, likely due to something related to the nature of data.
* **One Class Learning (Special Algorithms):** Isolation Forest, One-Class SVM, and Local Outlier Factor shows improvement in both recall and precision compared to the baseline.

In conclusion, baseline models performed poorly, in terms of recall and precision. One Class Learning, adjusted algorithms shown improvements in both recall and precision, especially Random Forest and SVM classifiers. Specialized algorithms like Isolation Forest, One-Class SVM, and Local Outlier Factor also performed well on this dataset, indicating their effectiveness in identifying anomalies or outliers on this imbalanced dataset. Such significant improvements in the performance of classifiers will particularly lead to correct identification of instances of the minority class while maintaining high precision.

****

# Stroke Dataset

Written code to solve the prevalent problem of imbalanced dataset, where one class dominates the dataset compared to the other. Such is the case for the following dataset related to Strokes. A stroke also known as a cerebrovascular accident (CVA) happens when part of the brain loses its blood supply and the part of the body that the blood-deprived brain cells control stops working, it is a medical emergency because strokes can lead to death or permanent disability. There are opportunities to treat strokes but those treatments need to be started in the first few hours after the signs of a stroke begin. Those signs and physical condition of the person are the features here and the target column which is imbalanced is of **‘stroke’** which tells if it’s true or false represented by 1s and 0s.

We have used the following methods to resolve Class Imbalance:

1. Random Under Sampling
2. Algorithmic Methods (Using Random Forest as well as modifying Class Weights)
3. Anomaly Detection Method

And we have used the following 5 Algorithms/models to draw a comparison between different methods:

1. Logistic Regression
2. K-Nearest Neighbours (KNN)
3. Random Forest
4. Support Vector Machines (SVM)
5. Naive Bayes (Gaussian)

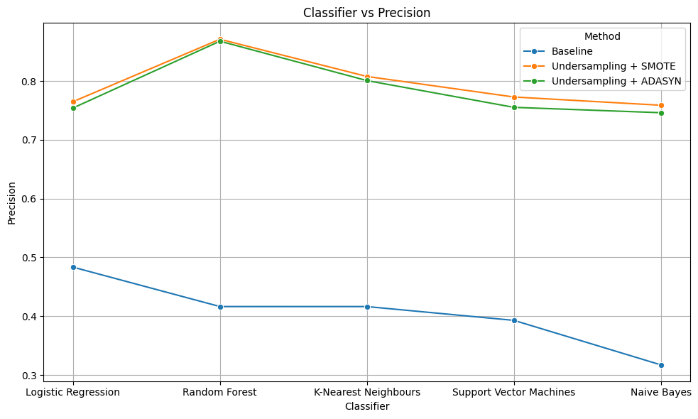
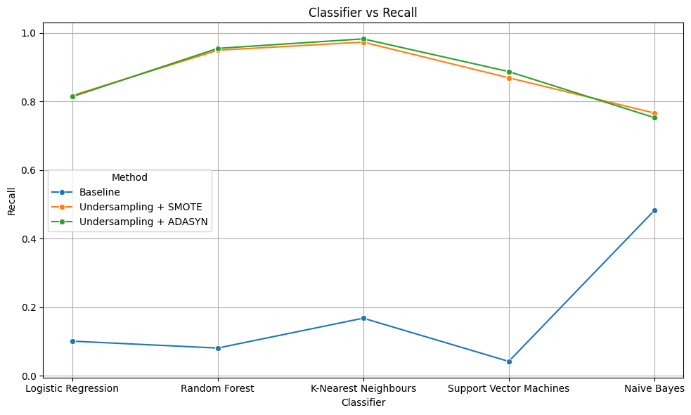
Dataset Link:*https://www.kaggle.com/datasets/shashwatwork/cerebral-stroke-predictionimbalaced-dataset*

## Sampling Method:

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Classifier | Class 1 Recall | Class 1 Precision |
| Baseline | Logistic Regression | 0.101082 | 0.483481 |
| Baseline | Random Forest | 0.080913 | 0.416445 |
| Baseline | K-Nearest Neighbours | 0.168029 | 0.416405 |
| Baseline | Support Vector Machines | 0.041971 | 0.392949 |
| Baseline | Naive Bayes | 0.482067 | 0.31724 |
| Undersampling + SMOTE | Logistic Regression | 0.817176 | 0.764742 |
| Undersampling + SMOTE | Random Forest | 0.949427 | 0.870942 |
| Undersampling + SMOTE | K-Nearest Neighbours | 0.973092 | 0.807674 |
| Undersampling + SMOTE | Support Vector Machines | 0.869084 | 0.772736 |
| Undersampling + SMOTE | Naive Bayes | 0.766412 | 0.758717 |
| Undersampling + ADASYN | Logistic Regression | 0.813921 | 0.753992 |
| Undersampling + ADASYN | Random Forest | 0.954791 | 0.867632 |
| Undersampling + ADASYN | K-Nearest Neighbours | 0.982555 | 0.800644 |
| Undersampling + ADASYN | Support Vector Machines | 0.887263 | 0.755161 |
| Undersampling + ADASYN | Naive Bayes | 0.752954 | 0.746006 |

* **Baseline**: Performance varies across all classifiers, with relatively low recall and precision. Naive Bayes stands out with moderate recall and precision, indicating better performance compared to other baseline models, in this case.
* **Undersampling + SMOTE**: Significant improvement in both recall and precision across most classifiers compared to the baseline. Random Forest, K-Nearest Neighbours, and Support Vector Machines shows notable improvement with this technique.
* **Undersampling + ADASYN**: This also resulted in considerable improvement in both recall and precision across most classifiers compared to the baseline. Random Forest, K-Nearest Neighbours, and Support Vector Machines shows notable improvement with this technique.

In conclusion, baseline models performed poorly, Naive Bayes was relatively better in performance compared to other baseline models. Undersampling with SMOTE and ADASYN effectively worked with class imbalance issues, leading to significant improvements in the performance of classifiers. Random Forest, K-Nearest Neighbours, and Support Vector Machines consistently showed notable improvements with both SMOTE and ADASYN techniques. Combining undersampling with techniques like SMOTE or ADASYN leads to noticeable improvements in the performance, it starts handling imbalanced datasets properly and identifies instances of the minority class with good precision.

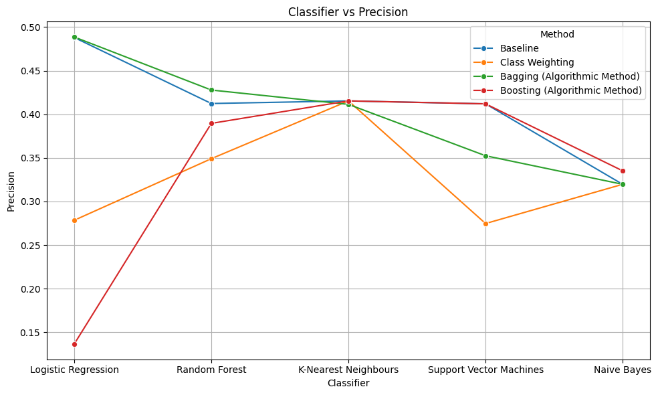
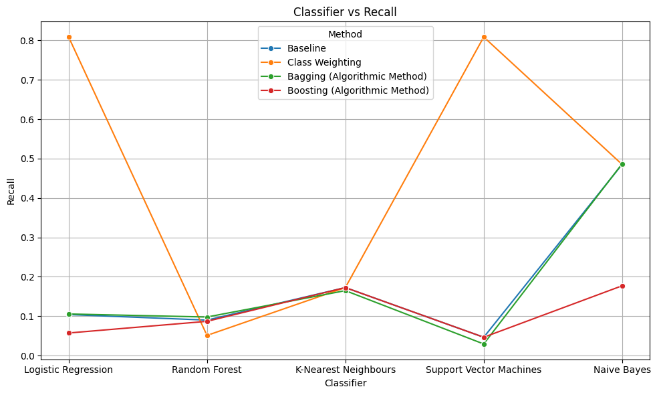


## Algorithmic Method:

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Classifier | Class 1 Recall | Class 1 Precision |
| Baseline | Logistic Regression | 0.104191 | 0.488081 |
| Baseline | Random Forest | 0.090177 | 0.412368 |
| Baseline | K-Nearest Neighbours | 0.172735 | 0.41536 |
| Baseline | Support Vector Machines | 0.046645 | 0.412 |
| Baseline | Naive Bayes | 0.485308 | 0.319683 |
| Class Weighting | Logistic Regression | 0.808769 | 0.278446 |
| Class Weighting | Random Forest | 0.051296 | 0.349087 |
| Class Weighting | K-Nearest Neighbours | 0.172735 | 0.41536 |
| Class Weighting | Support Vector Machines | 0.808781 | 0.274694 |
| Class Weighting | Naive Bayes | 0.485308 | 0.319683 |
| Bagging (Algorithmic Method) | Logistic Regression | 0.105753 | 0.488664 |
| Bagging (Algorithmic Method) | Random Forest | 0.097965 | 0.427822 |
| Bagging (Algorithmic Method) | K-Nearest Neighbours | 0.164983 | 0.411372 |
| Bagging (Algorithmic Method) | Support Vector Machines | 0.029542 | 0.352448 |
| Bagging (Algorithmic Method) | Naive Bayes | 0.48687 | 0.319696 |
| Boosting (Algorithmic Method) | Logistic Regression | 0.057437 | 0.136534 |
| Boosting (Algorithmic Method) | Random Forest | 0.08704 | 0.389571 |
| Boosting (Algorithmic Method) | K-Nearest Neighbours | 0.172735 | 0.41536 |
| Boosting (Algorithmic Method) | Support Vector Machines | 0.046645 | 0.412 |
| Boosting (Algorithmic Method) | Naive Bayes | 0.177277 | 0.33516 |

* **Baseline**: Performance varies across all classifiers, with low recall and precision. Naive Bayes shows relatively better performance compared to other baseline models.
* **Class Weighting**: Logistic Regression and SVM show improvements in recall, Random Forest and K-Nearest Neighbours doesn’t show any improvement.
* **Bagging (Algorithmic Method)**: There is no significant improvement in recall or precision compared to the baseline, after using this technique.
* **Boosting (Algorithmic Method)**: There is no significant improvement in recall or precision compared to the baseline.

In conclusion, baseline models performed poorly, Naive Bayes shows relatively better performance compared to other baseline models. Class weighting, bagging, and boosting techniques did not lead to substantial improvements either. Logistic Regression and SVM classifiers showed some improvements in recall with class weighting. So, the chosen techniques did not effectively address the class imbalance issue on this dataset.



## Anomaly (One-Class) Method:

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Classifier | Class 1 Recall | Class 1 Precision |
| Baseline | Logistic Regression | 0.101082 | 0.483481 |
| Baseline | Random Forest | 0.091803 | 0.418171 |
| Baseline | K-Nearest Neighbours | 0.168029 | 0.416405 |
| Baseline | Support Vector Machines | 0.041971 | 0.392949 |
| Baseline | Naive Bayes | 0.482067 | 0.31724 |
| One Class Learning (Adjusted Algorithms) | Logistic Regression | 0.801382 | 0.631293 |
| One Class Learning (Adjusted Algorithms) | Random Forest | 0.625216 | 0.626298 |
| One Class Learning (Adjusted Algorithms) | K-Nearest Neighbours | 0.207254 | 0.693642 |
| One Class Learning (Adjusted Algorithms) | Support Vector Machines | 0 | 0 |
| One Class Learning (Adjusted Algorithms) | Naive Bayes | 0.839378 | 0.597786 |
| One Class Learning (Special Algorithms) | Isolation Forest | 0.340591 | 0.375 |
| One Class Learning (Special Algorithms) | One-Class SVM | 0.247278 | 0.336864 |
| One Class Learning (Special Algorithms) | Local Outlier Factor | 1 | 0.500779 |

* **Baseline:** Performance was poor across all classifiers, Naive Bayes relatively performed better compared to other classifiers.
* **One Class Learning (Adjusted Algorithms):** Logistic Regression, Random Forest, K-Nearest Neighbours and Naive Bayes show improvements in both recall and precision compared to the baseline but Support Vector Machines show 0 which might be due to nature of data.
* **One Class Learning (Special Algorithms):** Isolation Forest, One-Class SVM, and Local Outlier Factor demonstrate improvements in either recall or precision compared to the baseline. Local Outlier Factor stands out with perfect recall but moderate precision, it can be either overfitted or due to nature of the data.

In conclusion, the baseline models show poor performance, Naive Bayes shows relatively better precision, while K-Nearest Neighbours has higher recall but lower precision. One Class Learning with adjusted algorithms shown promising improvements in both recall and precision, particularly with Logistic Regression, Random Forest, and Naive Bayes classifiers. And specialized algorithms, like Local Outlier Factor can effectively identify outliers in imbalanced datasets, leading to perfect recall in this scenario with moderate precision, which can be due to nature of data. So, techniques like One Class Learning with adjusted algorithms or specialized algorithms can lead to significant improvements in the performance of classifiers.

