San Diego RIPA REPORT

Are there patterns in police stops based on factors such as the officer's years of experience, type of assignment, location (school or not), perceived age, perceived gender, and perceived LGBT status of the person stopped?

Contents

Figures

Tables

Introduction

In California, under the Racial and Identity Profiling Act of 2015 (RIPA), state and local law enforcement agencies are obligated to gather information on individuals who are stopped and to submit this data to the California Department of Justice. The information that is recorded includes the person's age, gender, observed race, any disabilities that are not disclosed on identification, the date, time, and location of the stop, the reason for the stop, any actions taken during the stop, any search information, evidence found, and property seized. The aim of collecting this data is to systematically document and analyze stops and searches in order to identify any disparities that may exist across different demographic groups.

To further investigate the effects of perceived age and gender in police stops, we plan on clustering the RIPA data to understand the characteristics of stops and identify any significant patterns. This can involve visualizing the clusters, calculating descriptive statistics for each cluster, and interpreting the results. Then we can use the clusters as input for a classification algorithm.

Data Understanding

The dataset contains information on all police stops made by the San Diego Police Department that conform to the requirements of the Racial and Identity Profiling Act of 2015 (RIPA). The data collection began on July 1, 2018. Prior to that date, the data is limited to vehicle stops and is available in a separate dataset.

Each row in the dataset represents a person stopped and includes a unique identifier for the person and for the stop. A single stop may involve multiple people, so some stop_id values may have more than one associated pid. The dataset includes basic information about the stop, such as the date, time, duration, and location.

The dataset includes information on the officer's perception of the age, gender, and sexual orientation of the person stopped, as well as whether the officer perceived the person to have limited English proficiency. The dataset also includes information on the officer's assignment at the time of the stop, the reason for the stop, and any actions taken during the stop. The dataset indicates whether the stop was made in response to a call for service, whether the stop occurred at a school, and the name of the school if applicable.

The dataset consists of data from 653,601 instances and 29 attributes.

Here are the attributes of the dataset, along with their descriptions, data types, and statistics:

Table 1: Attributes description

Attribute	Description	Type	Statistics
stop_id	unique identifier for stop	Numeric	Min: 2,443 Max: 590,817
ori	agency originating identifier	Ordinal	CA0371100 is the SDPD ORI
agency	agency	Ordinal	SD
exp_years	officer years of experience in law enforcement	Numeric	Min: 1 Max: 50 Mean: 6.269 Std: 7.14
date_stop	date stop occurred	Date	From: 01-Jul-2018 To: 31-Dec-2022
time_stop	time stop began	Time	24 hrs
stopduration	duration of time for stop in minutes	Numeric	Min: 1min Max: 1440 min Mean: 28.673 Std: 50.364
stop_in_response_to_cfs	was the stop made in response to a call for service?	Binary	0 = No 1 = Yes
officer_assignment_key	type of officer assignment at time of stop (code)	Numeric	Min: 1 Max: 10 Mean: 1.410 Std: 1.769
assignment	type of officer assignment at time of stop (description)	Nominal	https://oag.ca.gov/sites/all/files/agweb/pdfs/ripa/stop-data-reg-final-text-110717.pdf?
intersection	location of stop - intersecting street name	Nominal	-
address_block	location of stop - hundred block	Numeric	Min: 0 Max: 99,999,900 Mean: 6,661.789 Std: 294,946.3
land_mark	location of stop - landmark	Nominal	-
address_street	location of stop - street name	Nominal	-
highway_exit	location of stop - highway exit	Nominal	-
isschool	did stop occur at a school?	Binary	0 = No 1 = Yes

school_name	name of school where stop occurred	Nominal	-
address_city	name of city where stop occurred	Nominal	-
beat	location of stop - SDPD beat	Numeric	Min: 111 Max: 999 Mean: 509.863 Std: 244.543
beat_name	location of stop - SDPD beat/neighborhood name	Nominal	-
pid	unique identifier for person on a stop	Numeric	Min: 0 Max: 52 Mean: 1.224 Std: 1.043
isstudent	was person stopped a student?	Binary	0 = No 1 = Yes
perceived_limited_english	officer's perception that the person stopped has limited or no fluency in English	Binary	0 = No 1 = Yes
perceived_age	officer's perception of the approximate age of the person stopped	Numeric	Min: 1 Max: 120 Mean: 37.113 Std: 13.384
perceived_gender	officer's perception of the gender of the person stopped (description)	Nominal	Male Female Transgender man/boy Transgender woman/girl
gender_nonconforming	officer's perception of whether the person stopped is gender nonconforming	Binary	0 = No 1 = Yes
Gend	officer's perception of the gender of the person stopped (code)	Numeric	Min: 0 Max: 4 Mean: 1.275 Std: 0.459
gend_nc	officer's perception of whether the person stopped is gender nonconforming	Numeric	No data = No 5 = Yes
perceived_lgbt	officer's perception of whether the person stopped is lesbian, gay, bisexual or transgender	Binary	Yes No

Below, we present various graphs that help to better understand the data.

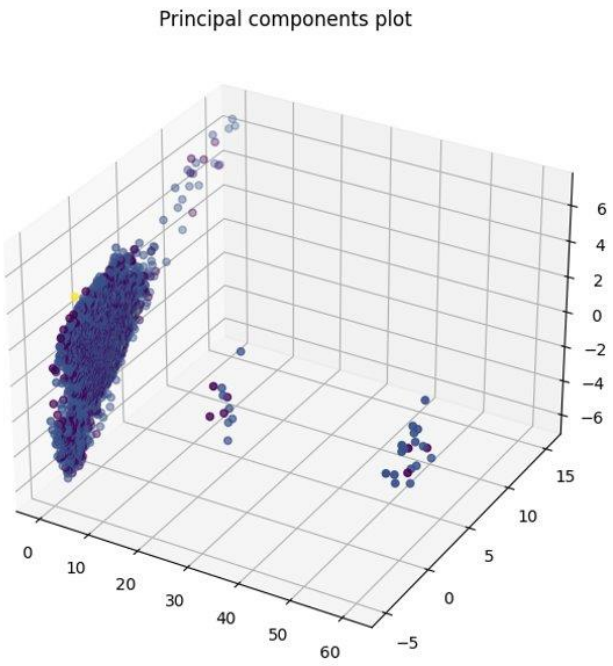

Figure 1: Display the occurrence of stops

We create a 3D scatter plot to display the relationship between the values on the x, y, and z axes, and the color of each point represents additional information. For example, a light blue color may represent male gender, while a dark blue color may represent the female gender.

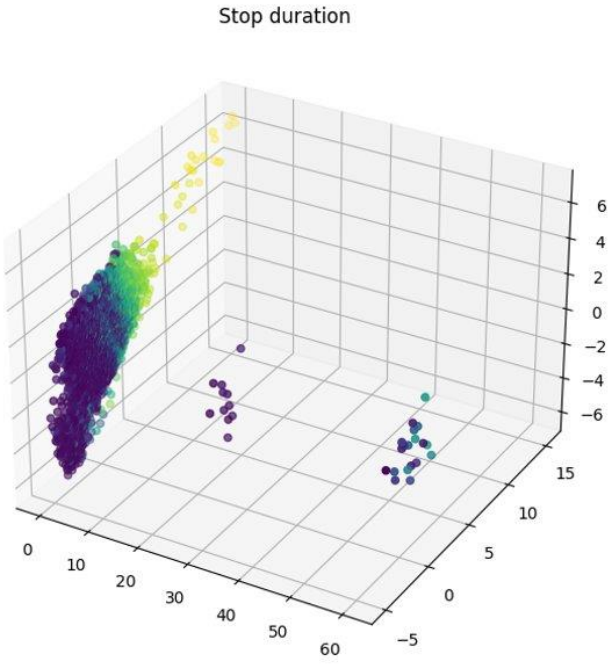

Figure 2: Display the duration of each person's stops

In this graph, we can observe the duration of each person's stops, where the dark blue color indicates short stops, and the yellow color indicates long stops.

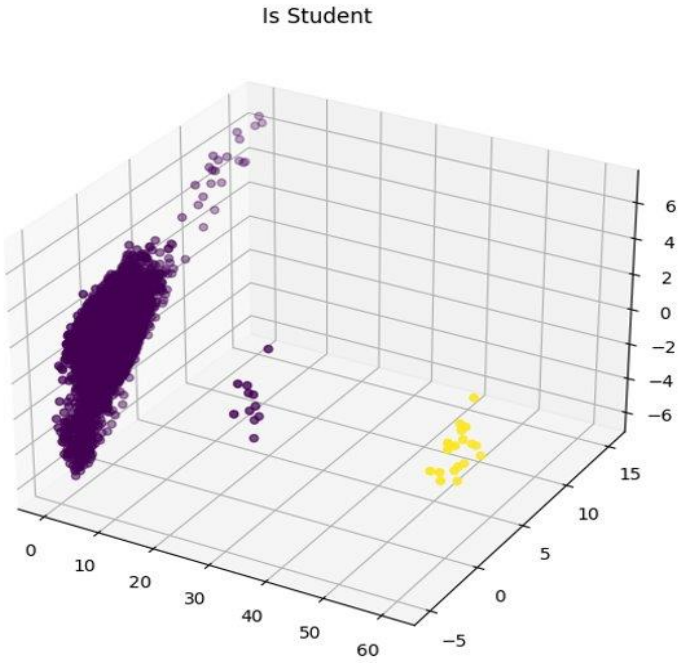

Figure 3: Display whether a stopped person was a student or not

This graph depicts whether or not the person who stopped was a student. The information is conveyed through the use of color, with yellow indicating individuals who were students and another color representing individuals who were not students.

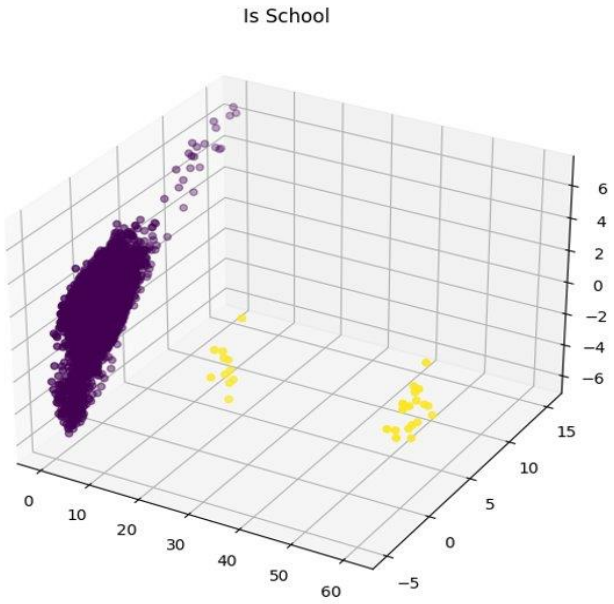

Figure 4: Display the occurrence of stops is on school locations or not

This graph illustrates whether a stop occurred on school locations or not. The color yellow is used to represent stops that occurred on school locations, but only for individuals who were not students.

Data preparation

Data preparation is a crucial step in machine learning (ML) as it can significantly impact the accuracy and reliability of the model. One of the critical aspects of data preparation is data cleaning, which involves identifying and fixing missing or incorrect data, removing outliers, and handling duplicate records. Another important aspect is resampling, where the dataset is modified to address class imbalance, allowing the model to learn from all classes equally. Dimensionality reduction is another technique used to simplify complex datasets by reducing the number of features without losing significant information. This step not only speeds up the model training but also helps avoid the high dimensionality data. Data preparation plays a vital role in building accurate and reliable ML models, and it is essential to invest significant time and effort in this phase of the ML pipeline.

Resampling

Resampling techniques are methods used to deal with imbalanced data sets, where one class is significantly more frequent than another. Three common resampling techniques are undersampling, oversampling, and SMOTE.

Undersampling

Undersampling involves reducing the number of samples in the majority class to match the number of samples in the minority class. This technique is simple to implement, but it can result in a loss of information since some samples are discarded. Undersampling can also increase the risk of overfitting since the model is trained on a smaller set of data.

Oversampling

Oversampling, on the other hand, involves increasing the number of samples in the minority class to match the number of samples in the majority class. This technique is useful when there are limited data points in the minority class. Oversampling can be done in various ways, such as random oversampling or generating synthetic data using SMOTE.

SMOTE

SMOTE (Synthetic Minority Over-sampling Technique) is an oversampling method that involves generating synthetic samples for the minority class. The SMOTE algorithm creates synthetic samples by randomly selecting a minority class sample and then finding its k-nearest neighbors. It then generates new samples by interpolating between the selected sample and its k-nearest neighbors. SMOTE can help to reduce overfitting and improve classification accuracy by creating new samples that are similar to the existing ones. However, SMOTE can also introduce noise and may not be effective when the minority class is too sparse.

Dimensionality reduction

PCA (Principal Component Analysis) and LDA (Linear Discriminant Analysis) are both popular dimensionality reduction techniques used in machine learning and data analysis. They are used to reduce the number of features in a dataset while still retaining the maximum amount of information possible. However, they differ in their objectives, assumptions, and algorithms.

PCA

This is an unsupervised linear transformation technique that projects high-dimensional data onto a lower-dimensional space by identifying the principal components that explain the maximum variance in the data. The main goal of PCA is to find a smaller set of uncorrelated variables (known as principal components) that explain most of the variance in the data. PCA assumes that the principal components are orthogonal to each other and that they represent the directions of maximum variance in the data.

LDA

LDA, on the other hand, is a supervised linear transformation technique that attempts to find a lower-dimensional representation of the data that maximizes the separation between classes. The main goal of LDA is to project the data onto a lower-dimensional space while still preserving the class separability. LDA assumes that the data follows a normal distribution and that the classes have the same covariance matrix.

Outlier detection

Outlier detection is a crucial task in many applications, such as fraud detection, network intrusion detection, and anomaly detection in sensor data. An outlier is a data point that deviates significantly

from the rest of the data points in a dataset. Outliers can be caused by measurement errors, data corruption, or rare events.

Local Outlier Factor (LOF)

Local Outlier Factor (LOF) is a density-based outlier detection technique that measures the local density of a data point relative to its k-nearest neighbors. The LOF score of a data point indicates how much it deviates from the local density of its neighbors. A high LOF score indicates that a data point is an outlier, while a low LOF score indicates that a data point is a typical point.

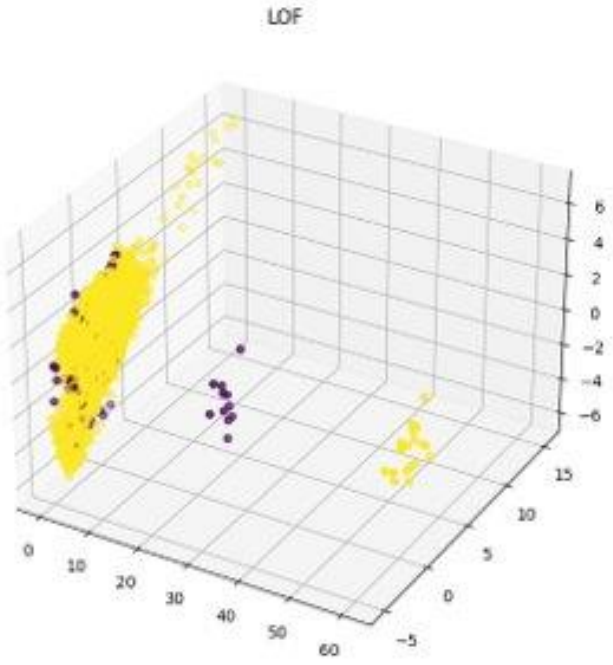

Figure 5: LOF plot

The LOF algorithm works as follows:

1. For each data point, find its k-nearest neighbors.
2. Calculate the reachability distance of each data point to its k-nearest neighbors.
3. Calculate the local reachability density of each data point as the inverse of the average reachability distance of its k-nearest neighbors.
4. Calculate the LOF score of each data point as the average ratio of the local reachability density of a data point to the local reachability density of its k-nearest neighbors.

LOF is sensitive to the choice of the parameter k, which determines the size of the local neighborhood. A small value of k can result in a high sensitivity to noise, while a large value of k can lead to the loss of local details.

Isolation Forest (ISF)

Isolation Forest (ISF) is a tree-based outlier detection technique that isolates outliers by randomly partitioning the data into subspaces. The algorithm works by constructing a random forest of isolation trees, where each isolation tree is built by recursively partitioning the data into subsets until each subset contains only one data point.

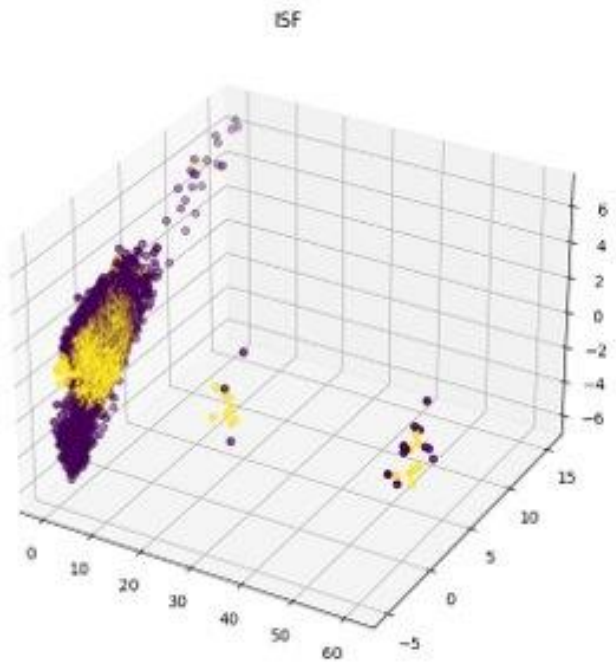

Figure 6: ISF plot

The ISF algorithm works as follows:

1. Randomly select a feature and a split point to divide the data into two subsets.
2. Recursively repeat step 1 until each subset contains only one data point or the maximum tree depth is reached.
3. Calculate the anomaly score of each data point as the average depth of the isolation trees in which the data point appears. A low anomaly score indicates that a data point is an outlier, while a high anomaly score indicates that a data point is a typical point.

ISF has several advantages over other outlier detection techniques. First, it can handle high-dimensional data efficiently because it only uses a random subset of features at each split. Second, it can detect outliers that are in clusters because it does not assume a specific data distribution.

One-Class Support Vector Machine (OCSVM)

One-Class Support Vector Machine (OCSVM) is a supervised outlier detection technique that learns a boundary around a set of normal data points. The algorithm works by finding the hyperplane that maximizes the margin between the normal data points and the origin.

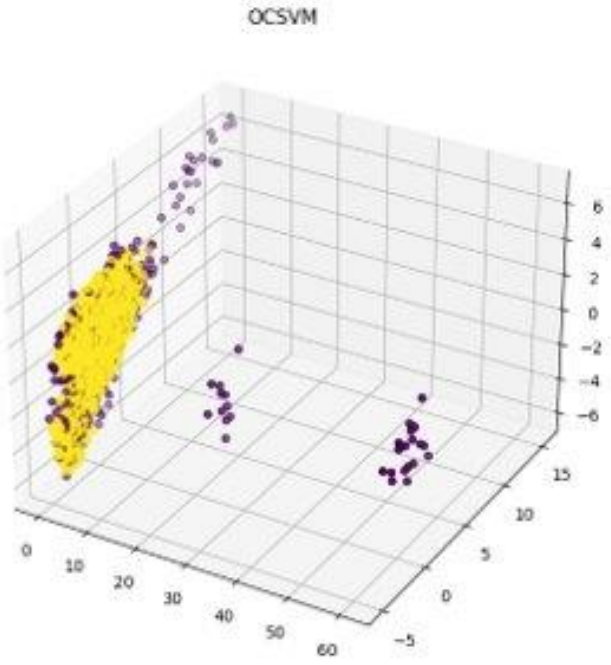

Figure 7: OCSVM plot

The OCSVM algorithm works as follows:

1. Transform the data into a high-dimensional space using a kernel function.
2. Train a support vector machine on the transformed data using a one-class classification approach. In one-class classification, the goal is to find a decision boundary that separates the normal data points from the origin.
3. Calculate the outlier score of each data point as the distance between the data point and the decision boundary. A high outlier score indicates that a data point is an outlier, while a low outlier score indicates that a data point is a typical point.

OCSVM has several advantages over other outlier detection techniques. First, it can handle high-dimensional data efficiently by using kernel functions. Second, it can detect outliers that are located in clusters or have complex shapes because it does not assume a specific data distribution. Third, it can handle imbalanced data where the number of normal data points is much larger than the number of outliers.

However, OCSVM also has some limitations. It requires a training set of normal data points, which may not be representative of the entire data distribution. It also requires tuning of the hyperparameters, such as the kernel function and the regularization parameter, which can be time-consuming and difficult.

Ensemble

Ensemble methods work by training several outlier detection models on the same dataset using different techniques or algorithms, and then combining their results to obtain a final prediction. This approach can help to reduce the risk of false positives and false negatives in the outlier detection process, as different models may identify different outliers or use different criteria for identifying outliers.

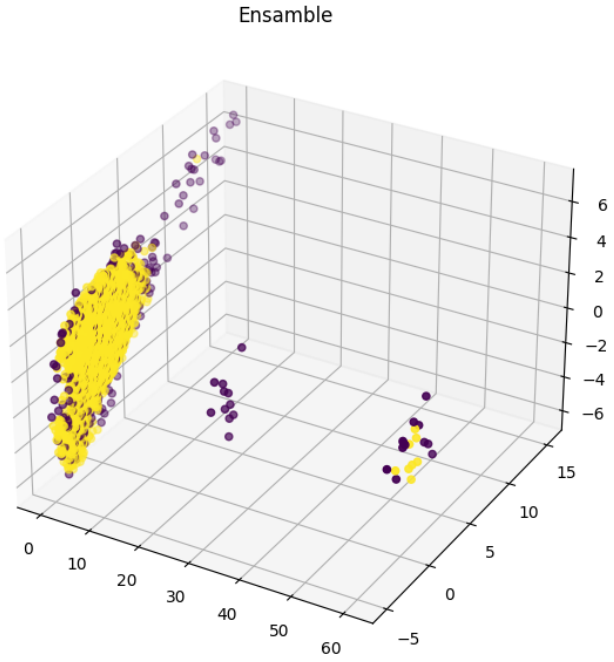

Figure 8: Ensemble outliers plot

By combining their outputs through majority voting, the ensemble method can provide a more robust and accurate outlier detection process.

Modeling & Results

Classification

Classification methods are algorithms used in machine learning to predict the class of an observation based on a set of features. In this explanation, we will discuss six popular classification methods: k-Nearest Neighbors (kNN), Decision Trees (DT), Random Forests (RF), Naïve Bayes, Support Vector Machines (SVM), and Multilayer Perceptron (MLP).

k-Nearest Neighbors (kNN)

The kNN algorithm is a non-parametric algorithm that uses a distance metric to find the k closest observations in the training set to the observation being predicted. The majority class of these k nearest neighbors is assigned as the predicted class. One of the advantages of kNN is that it is very simple and easy to implement. However, it can be sensitive to the choice of k and the distance metric used.

Table 2: KNN performance

Dataset	Accuracy	Precision	Recall	F1 Score
Original	0.675303	0.617464	0.675303	0.634751
Normalized	0.675303	0.616827	0.675303	0.635576
Standardized	0.679588	0.619756	0.679588	0.638065
PCA	0.675507	0.616102	0.675507	0.635053
LDA	0.675916	0.614803	0.675916	0.634157
Downsampled	0.347292	0.602548	0.347292	0.42763
Upsampled	0.526063	0.611254	0.526063	0.551931
SMOTE	0.499745	0.614424	0.499745	0.537416

The Original, Normalized, Standardized, PCA, and LDA preprocessing techniques have similar performance metrics, with accuracy ranging from 67.5% to 67.9%. These techniques do not seem to significantly improve or worsen the model's performance compared to the original dataset.

The Downsampled dataset has the lowest accuracy, precision, and F1 score, with accuracy at only 34.7%. This indicates that Downsampling may not be a suitable preprocessing technique for this problem, as it significantly reduces the model's performance.

The Upsampled and smote techniques show moderate improvements in precision, but their accuracy and F1 scores are lower than the original dataset. This suggests that Upsampling and SMOTE may improve the model's ability to correctly predict gender, but overall performance remains less than ideal.

The highest F1 score is achieved with the std preprocessing technique, at 0.638, suggesting that standardization may be the most suitable preprocessing technique for this problem.

Decision Trees (DT)

The DT algorithm is a tree-based model that uses a hierarchical structure to split the data based on the feature that provides the most information gain. Each node in the tree represents a split on a feature, and the leaves represent the predicted class. Decision trees are easy to interpret and can handle both numerical and categorical data. However, they can suffer from overfitting if the tree is too deep.

Table 3: Decision Tree performance

Dataset	Accuracy	Precision	Recall	F1 Score
Original	0.605631	0.614531	0.605631	0.609884
Normalized	0.607161	0.619765	0.607161	0.613068
Standardized	0.607161	0.620447	0.607161	0.613372
PCA	0.605988	0.614584	0.605988	0.610097
LDA	0.603999	0.60736	0.603999	0.605643
Downsampled	0.254718	0.594254	0.254718	0.343814
Upsampled	0.569418	0.606986	0.569418	0.584876
SMOTE	0.563705	0.608361	0.563705	0.582452

The highest accuracy (0.607) is achieved by both the normalized and standardized datasets, followed closely by the original, PCA, and LDA datasets.

The highest precision (0.620) is obtained by the standardized dataset, followed by the normalized dataset.

The highest recall (0.607) is achieved by both the normalized and standardized datasets, followed closely by the original, PCA, and LDA datasets.

The highest F1 score (0.613) is obtained by the standardized dataset, followed by the normalized dataset.

Normalization and standardization preprocessing techniques perform the best in terms of accuracy, precision, recall, and F1 score. The downsampled dataset yields the lowest performance across all metrics, while the upsampled and SMOTE datasets have a lower performance compared to the original, PCA, and LDA datasets.

Random Forests (RF)

RF is an ensemble method that combines multiple DT models to improve accuracy and reduce overfitting. RF generates many decision trees on random subsets of the training data and features, and the predicted class is determined by the majority vote of all the trees. RF can handle noisy data and perform well with high-dimensional data. However, it can be slow to train and requires more computational resources than DT.

Table 4: Random Forest performance

Dataset	Accuracy	Precision	Recall	F1 Score
Original	0.716566	0.639385	0.716566	0.631203
Normalized	0.722483	0.648796	0.722483	0.639412
Standardized	0.724472	0.654638	0.724472	0.640339
PCA	0.685759	0.621635	0.685759	0.639405
LDA	0.712537	0.61789	0.712537	0.629069
Downsampled	0.30256	0.62395	0.30256	0.387164
Upsampled	0.678721	0.634016	0.678721	0.648088
SMOTE	0.646639	0.623686	0.646639	0.633477

The highest accuracy is achieved using standardization with an accuracy of 0.7245, followed by normalization with 0.7225. The lowest accuracy is achieved with the downsampled dataset at 0.3026.

In terms of precision, the highest value is obtained using standardization at 0.6546, while the lowest precision is seen in the LDA preprocessed dataset at 0.6179.

The recall metric is the same as the accuracy for each preprocessing method. This might be an indication that the dataset is relatively balanced across different classes, or it could be a coincidence.

The highest F1 Score is obtained using the standardized dataset with a value of 0.6403. The lowest F1 Score is seen in the downsampled dataset at 0.3872.

Based on these results, the standardization preprocessing technique yielded the best overall performance for the Random Forest algorithm in predicting a person's gender using the police stop dataset.

Naïve Bayes

The Naïve Bayes algorithm is a probabilistic model that uses Bayes' theorem to calculate the probability of a given observation belonging to each class. Naïve Bayes assumes that all features are independent, which can result in some loss of accuracy but makes it fast and efficient. It is often used in text classification and spam filtering.

Table 5: Naïve Bayes performance

Dataset	Accuracy	Precision	Recall	F1 Score
Original	0.03504	0.627103	0.03504	0.065481
Normalized	0.015148	0.608027	0.015148	0.026583
Standardized	0.015148	0.608027	0.015148	0.026583
PCA	0.727787	0.531277	0.727787	0.614197
LDA	0.726767	0.634262	0.726767	0.617061
Downsampled	0.274049	0.583222	0.274049	0.220592
Upsampled	0.012547	0.629406	0.012547	0.024041
SMOTE	0.037438	0.608871	0.037438	0.069767

Without any further preprocessing, the model's accuracy is quite low at 3.5%, with precision at 62.7% and F1 score at 6.5%. The low accuracy, recall, and F1 score indicate that the model does not generalize well and has difficulty predicting the correct gender.

Normalizing the data further decreases the accuracy to 1.5%, with a similar decrease in recall and F1 score. The precision remains nearly the same at 60.8%. The preprocessing technique has not improved the model's performance.

Standardizing the data also results in an accuracy of 1.5%, with precision at 60.8% and F1 score at 2.7%. This preprocessing method does not improve the model's performance either.

Using PCA, the model achieves its highest accuracy of 72.8%. The precision is lower at 53.1%, but the recall and F1 score are significantly higher at 72.8% and 61.4%, respectively. PCA seems to be the best preprocessing technique for improving the model's performance.

LDA also results in a high accuracy of 72.7%, with a higher precision of 63.4% and an F1 score of 61.7%. LDA is another effective preprocessing technique for this dataset.

Downsampling the data leads to a lower accuracy of 27.4%, with precision at 58.3% and F1 score at 22.1%. This preprocessing method does not improve the model's performance.

Upsampling the data yields the lowest accuracy of 1.25%, with precision at 62.9% and F1 score at 2.4%. This technique is not effective for improving the model's performance.

The SMOTE results in an accuracy of 3.7%, with precision at 60.9% and F1 score at 7.0%. This preprocessing method does not improve the model's performance.

Support Vector Machines (SVM)

SVM is a linear and non-linear classification algorithm that finds the best hyperplane to separate the data points into different classes. The hyperplane is chosen to maximize the margin between the classes, and the observations closest to the hyperplane are called support vectors. SVM can handle complex data and is effective in high-dimensional spaces. However, it can be sensitive to the choice of kernel function and the parameter tuning.

Table 6: SVM accuracies

Dataset	Accuracy	Precision	Recall	F1 Score
Original	0.724319	0.524638	0.724319	0.608516
Normalized	0.72891	0.531309	0.72891	0.614618
Standardized	0.728757	0.584861	0.728757	0.614638
PCA	0.728859	0.531299	0.728859	0.614593
LDA	0.72891	0.531309	0.72891	0.614618

Downsampled	0.430276	0.571143	0.430276	0.473959
Upsampled	0.430633	0.600264	0.430633	0.473429
SMOTE	0.293023	0.615163	0.293023	0.377149

Without any further preprocessing, the SVM achieves an accuracy of 72.43%, with a precision of 52.46%, recall of 72.43%, and an F1 score of 60.85%.

After normalizing the data, there is a slight improvement in the algorithm's performance with an accuracy of 72.89%, precision of 53.13%, recall of 72.89%, and an F1 score of 61.46%.

Standardizing the data also leads to better results, with an accuracy of 72.87%, precision of 58.49%, recall of 72.87%, and an F1 score of 61.46%.

Using Principal Component Analysis for dimensionality reduction, the algorithm achieves an accuracy of 72.89%, precision of 53.13%, recall of 72.89%, and an F1 score of 61.46%.

Implementing Linear Discriminant Analysis for dimensionality reduction yields the same results as with PCA, showing an accuracy of 72.89%, precision of 53.13%, recall of 72.89%, and an F1 score of 61.46%.

Using a downsampled dataset, the performance of the algorithm decreases significantly with an accuracy of 43.03%, precision of 57.11%, recall of 43.03%, and an F1 score of 47.40%.

Similarly, with an upsampled dataset, the performance drops to an accuracy of 43.06%, precision of 60.03%, recall of 43.06%, and an F1 score of 47.34%.

Applying SMOTE results in the lowest performance, with an accuracy of 29.30%, precision of 61.52%, recall of 29.30%, and an F1 score of 37.71%.

Multilayer Perceptron (MLP)

MLP is a type of artificial neural network that consists of multiple layers of interconnected neurons. The input layer receives the features, and the output layer predicts the class. The hidden layers perform nonlinear transformations on the data, allowing for complex relationships to be learned. MLP can handle complex data and can be used for both classification and regression.

However, it can be prone to overfitting if the model is too complex, and it requires more computational resources than some of the other algorithms.

Table 7: MLP accuracies

Dataset	Accuracy	Precision	Recall	F1 Score
Original	0.723656	0.600027	0.723656	0.60895
Normalized	0.727379	0.60948	0.727379	0.615841
Standardized	0.72386	0.631093	0.72386	0.62492
PCA	0.728654	0.636214	0.728654	0.615346
LDA	0.72891	0.531309	0.72891	0.614618
Downsampled	0.516679	0.59805	0.516679	0.540119
Upsampled	0.361624	0.627456	0.361624	0.341477
SMOTE	0.399929	0.605348	0.399929	0.446206

The original dataset was used without any further preprocessing. The model achieved an accuracy of 72.37%, precision of 60.00%, recall of 72.37%, and F1 score of 60.89%.

The dataset was normalized. The model's performance improved slightly, with an accuracy of 72.74%, precision of 60.95%, recall of 72.74%, and F1 score of 61.58%.

The dataset was standardized. The model achieved an accuracy of 72.39%, precision of 63.11%, recall of 72.39%, and F1 score of 62.49%. The precision improved compared to the original and normalized datasets.

Principal Component Analysis was applied to the dataset. The model's performance improved further, with an accuracy of 72.87%, precision of 63.62%, recall of 72.87%, and F1 score of 61.53%. The precision continued to improve.

The model achieved with LDA an accuracy of 72.89%, precision of 53.13%, recall of 72.89%, and F1 score of 61.46%. Although the accuracy and recall are similar to PCA, the precision is much lower.

The model's performance dropped significantly with downsampling, with an accuracy of 51.67%, precision of 59.80%, recall of 51.67%, and F1 score of 54.01%.

The model's performance further decreased with upsampling, with an accuracy of 36.16%, precision of 62.75%, recall of 36.16%, and F1 score of 34.15%. Upsampling, however, improved the precision compared to downsampling.

With SMOTE, the model achieved an accuracy of 39.99%, precision of 60.53%, recall of 39.99%, and F1 score of 44.62%. SMOTE's performance is better than upsampling but worse than downsampling, in terms of accuracy and F1 score.

Clustering

Clustering is a type of unsupervised learning that groups data points together based on their similarities. In this response, we will compare four popular clustering methods: kMeans, DBSCAN, EM, and Agglomerative clustering.

We performed clustering on all datasets. All results can be found in the Appendix. In this section we will present the clustering for the dataset that yielded the best classification results for most models: The standardized dataset.

kMeans

kMeans is a centroid-based clustering method that partitions the dataset into k clusters, where k is a user-specified number of clusters. It starts by randomly selecting k initial cluster centers, and then iteratively assigns each data point to the nearest cluster center based on the Euclidean distance. Once all the data points are assigned, the cluster centers are updated by calculating the mean of all the data points in the cluster. This process continues until convergence.

Advantages:

1. Simple and easy to implement.
2. Works well when the clusters are well-separated and the data is normally distributed.

Disadvantages:

1. Requires the number of clusters to be specified beforehand, which can be difficult to determine.
2. Can be sensitive to the initial randomly chosen centroids.
3. Not suitable for clusters with irregular shapes or different sizes.

In our investigation, we decided to create 5 clusters, hoping they would naturally separate our chosen classes. However, we have found that the clusters mainly focussed on defining regions in the dataspace. In the case of KMeans, the clusters divided the main region of the data into four segments. None of which correspond with a predetermined attribute. Moreover, the remaining cluster focussed solely on the students' category.

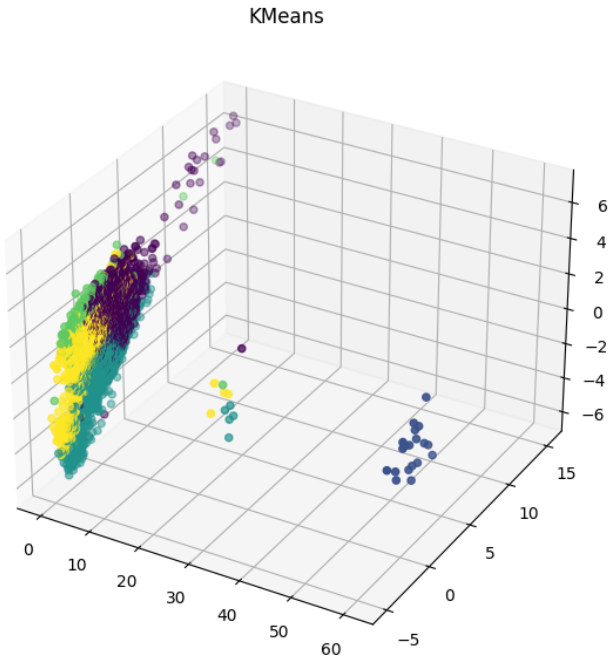

Figure 9: KMeans clustering using 5 clusters plotted over the first three principal components of the standardized dataset.

DBSCAN

DBSCAN is a density-based clustering method that groups together points that are close to each other in a high-density region and separates points in low-density regions. It requires two parameters: epsilon (ε) and minimum points (MinPts). A point is considered a core point if it has at least MinPts points within a distance of ε. Points that are within the ε distance of a core point are considered part of the same cluster.

Advantages:

1. Can detect clusters with irregular shapes and different sizes.
2. Does not require the number of clusters to be specified beforehand.
3. Can handle noise points and outliers.

Disadvantages:

1. Can be sensitive to the choice of parameters ε and MinPts.
2. Not suitable for datasets with varying densities or when the clusters have different densities.

For DBSCAN, we iterated over the minimum distance and minimum number of samples to decide what the best parameters were. We only performed manual iterations and made our decision based on our subjective perspective of what would be the better results by looking at the plots based on the top three principal components.

Figure 10 shows how DBSCAN performed with these hyperparameters. It clustered the main region of the data into two parts, the larger blue cluster, and the smaller green cluster at the top. It is still not clear why this separation happened, but it can also be observed in the KMeans clustering. In addition, it separated the two islands to the right into two separate clusters. The purple points were not clustered.

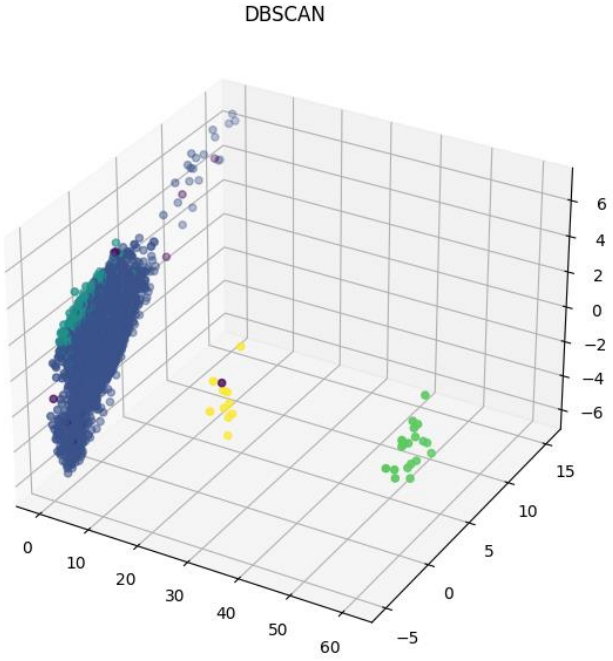

Figure 10: DBSCAN using eps=5 and min_samples=5 plotted over the first three principal components of the standardized dataset.

EM

EM (Expectation Maximization) is a probabilistic clustering method that assumes that the data points are generated from a mixture of Gaussian distributions. It estimates the parameters of the Gaussian distributions that best fit the data using the Expectation-Maximization algorithm. The algorithm iteratively updates the parameters until convergence.

Advantages:

1. Can handle clusters with different shapes and sizes.
2. Can handle missing data and incomplete data.
3. Provides probability values that indicate how likely a data point belongs to a particular cluster.

Disadvantages:

1. Can be computationally expensive, especially for large datasets.
2. Requires the number of clusters and the covariance structure to be specified beforehand.

We chose the same number of clusters for EM as we did for KMeans. Here we can notice the difference between how EM decided to take the different components of the data into account when performing clustering.

The main region in the dataset was divided into four parts. Three of the clusters focussed on whether the police stops were long or not. The fourth cluster seems to be spread around the green and purple clusters. The final blue cluster incorporated all stops that happened in schools.

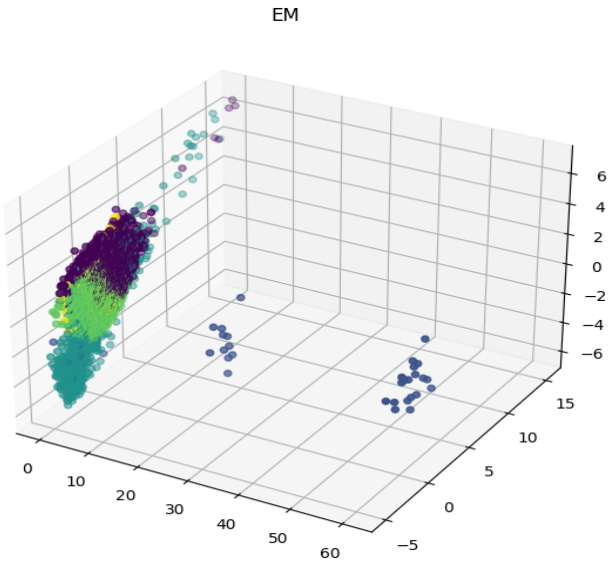

Figure 11: EM clusters plotted over the first three principal components of the standardized dataset.

Agglomerative clustering

Agglomerative clustering is a hierarchical clustering method that starts by considering each data point as a separate cluster and then iteratively merges the two closest clusters based on a linkage criterion. The linkage criterion determines the distance between two clusters. There are several linkage criteria, including single linkage, complete linkage, and average linkage.

Advantages:

1. Can handle datasets with varying shapes and sizes.
2. Provides a hierarchy of clusters that can be visualized as a dendrogram.
3. Does not require the number of clusters to be specified beforehand.

Disadvantages:

1. Can be computationally expensive, especially for large datasets.
2. Sensitive to noise and outliers.
3. Difficult to determine the optimal number of clusters from the dendrogram.

Figure 12 shows how the Agglomerative Clustering performed similar to EM. Its major differences are separating the students from the non-student stops that happened in schools, and the fourth clustering at the top of the main region, resembling the KMeans and DBSCAN clustering.

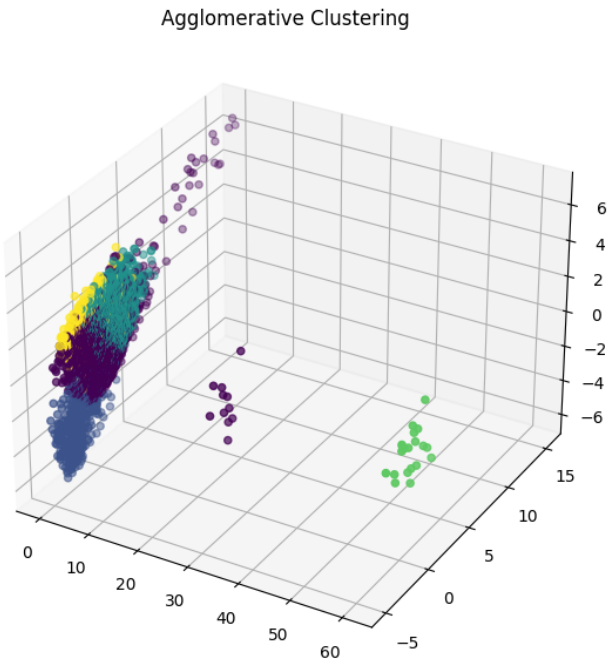

Figure 12: Agglomerative Clustering plotted over the first three principal components of the standardized dataset.

Discussion of results

Classification

In this project, we evaluated the performance of various classification algorithms in predicting a person's gender using the RIPA police stop dataset from the city of San Diego.

The effectiveness of these algorithms was found to be contingent on the preprocessing technique employed. Specifically:

1- kNN: The standardization preprocessing technique produced the highest F1 score, indicating that it may be the optimal preprocessing technique for this problem.

2- Decision Tree: Both normalization and standardization preprocessing techniques demonstrated superior performance in terms of accuracy, precision, recall, and F1 score.

3- Random Forest: The standardization preprocessing technique resulted in the best overall performance in predicting a person's gender using the police stop dataset.

4- Naive Bayes: The PCA and LDA preprocessing techniques substantially enhanced the model's performance, with PCA attaining the maximum accuracy and F1 score.

5- SVM: Algorithm performance was improved by normalizing, standardizing, and employing PCA or LDA for dimensionality reduction, with PCA and LDA achieving the highest accuracy and F1 score.

6- MLP: The Principal Component Analysis preprocessing technique ameliorated the model's performance in accuracy, precision, and F1 score, while LDA yielded comparable accuracy and recall but diminished precision.

Interestingly, the downsampling, upsampling, and SMOTE techniques generally did not augment the model's performance and, in certain instances, resulted in inferior performance compared to the original dataset. This observation implies that these techniques may not be appropriate for this problem and dataset.

The standardization, normalization, PCA, and LDA preprocessing techniques appear to be the most efficacious in enhancing the performance of the algorithms in predicting a person's gender using the RIPA police stop dataset. Nevertheless, the choice of the most suitable preprocessing technique is contingent on the specific algorithm and the performance metrics deemed most significant for the given problem.

These results may indicate a bias in how police treat different genders because the classification algorithms' performance relies on the patterns and trends present in the dataset. If the dataset, in this case, the RIPA police stop dataset, inherently contains biases in terms of gender, the algorithms will likely capture and replicate these biases in their predictions.

For instance, if the dataset reveals that police are more likely to stop certain genders for specific reasons, this information will be embedded in the predictive models. As mentioned earlier, standardization, normalization, PCA, and LDA preprocessing techniques have shown to be the most effective for this problem. In doing so, they might emphasize certain features or trends in the

data that cause bias, leading to predictions that relate to existing disparities in police treatment of different genders.

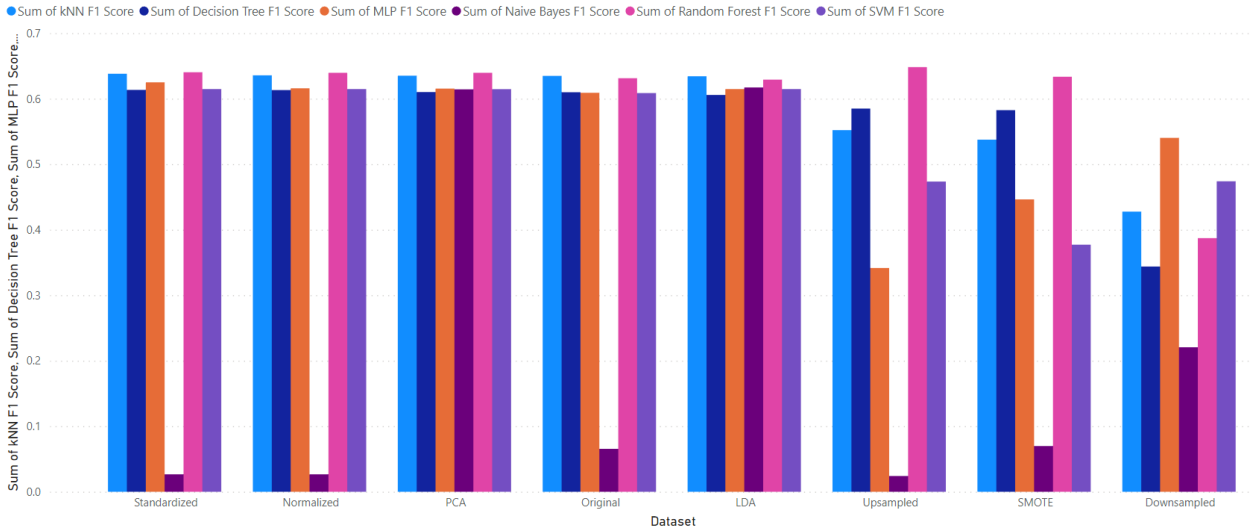

Figure 13: Bar chart for all dataset with all models

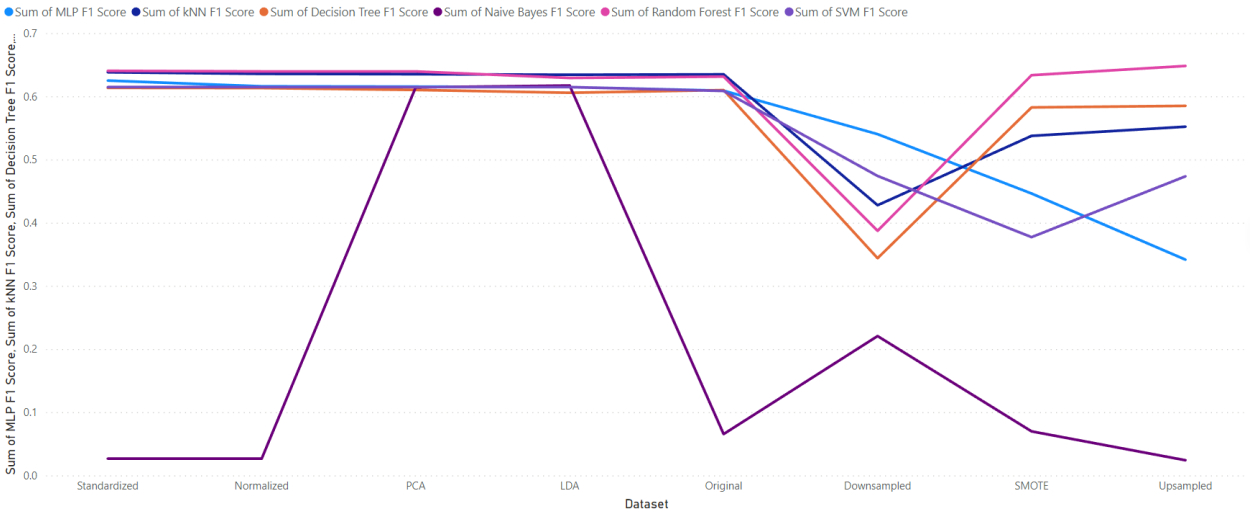

Figure 14: Line graph for all dataset with all models

Clustering

We explored the performance of different clustering techniques, specifically KMeans, DBSCAN, EM, and Agglomerative Clustering, in separating our chosen classes based on various attributes. The results obtained from these methods provided interesting insights into the underlying structure of the dataset and highlighted the strengths and limitations of each technique.

In the case of KMeans, we observed that the algorithm primarily focused on defining regions in the data space, which did not correspond with any pre-determined class attribute. This suggests that KMeans may not be suitable for identifying our chosen class in our dataset, as it appears to be more sensitive to the distribution of data points rather than our intended classification goal

DBSCAN, on the other hand, showed a more nuanced clustering behavior. By manually iterating over the minimum distance and minimum number of samples, we were able to identify clusters that better reflect the top three principal components of the data. However, the subjective nature of this approach may limit the generalizability of our findings, as different people may arrive at different conclusions based on their interpretation of the plots.

The EM algorithm displayed a different approach to clustering, taking into account various components of the data. Although the main region of the dataset was divided into four parts, the clusters seemed to focus on specific attributes, such as the duration of police stops and whether the stops occurred in schools. This suggests that the EM algorithm may be more suitable for identifying meaningful clusters in our dataset, as it considers the attribute information in addition to the distribution of data points.

Lastly, Agglomerative Clustering showed a similar performance to EM, with some notable differences. This technique was able to separate student and non-student stops that occurred in schools and identified a fourth cluster at the top of the main region. This finding indicates that Agglomerative Clustering may offer additional insights into the underlying structure of the dataset that are not captured by the other techniques.

We highlight here the importance of carefully selecting the appropriate clustering technique for a given dataset, as each method offers unique insights into the data's underlying structure. While KMeans may not be suitable for identifying meaningful clusters in our dataset, DBSCAN, EM, and Agglomerative Clustering offer more nuanced perspectives that may help uncover hidden patterns and relationships within the data. Further experimentation is needed to explore the generalizability of our findings, and evaluate the performance of these clustering techniques against different aspects of the data.

Conclusion

Classification

This project investigated the performance of various classification algorithms in predicting a person's gender using the RIPA police stop dataset from the city of San Diego. The effectiveness of these algorithms was found to depend on the preprocessing technique employed. Standardization, normalization, PCA, and LDA preprocessing techniques were identified as the most efficacious in improving the performance of the algorithms, although the optimal technique varied depending on the specific algorithm and performance metrics considered most critical.

However, it is essential to note that the classification algorithms' performance is highly reliant on the patterns and trends present in the dataset. If the RIPA police stop dataset inherently contains gender biases, the algorithms will likely capture and replicate these biases in their predictions. This observation highlights the need for careful consideration when employing machine learning algorithms for predictive tasks, especially in sensitive domains such as law enforcement, where the consequences of biased predictions can be severe.

Future work could explore alternative preprocessing techniques, feature selection methods, and classification algorithms to improve the gender prediction accuracy of police stop data further. By doing so, machine learning can be used as a powerful tool in understanding and addressing disparities in police treatment of different genders, ultimately contributing to a more equitable and just society.

Clustering

All clustering methods have their strengths and weaknesses, and the choice of which method to use depends on the characteristics of the dataset and the goals of the analysis. kMeans is suitable for well-separated clusters with a normal distribution, while DBSCAN is suitable for irregular shaped clusters with different densities. EM is suitable for Gaussian mixture models and can handle missing data, while Agglomerative clustering provides a hierarchy of clusters that can be useful for exploratory analysis.

In this project, we investigated the performance of four clustering techniques, namely KMeans, DBSCAN, EM, and Agglomerative Clustering, in separating our chosen classes based on various attributes. Our findings revealed that different clustering techniques offer unique insights into the

underlying structure of the dataset and emphasize the importance of selecting the appropriate method for a given dataset.

KMeans demonstrated limitations in identifying our chosen class, as it was more sensitive to the distribution of data points rather than the classification goal. DBSCAN, on the other hand, provided better results by reflecting the top three principal components of the data. However, its subjective nature may limit the generalizability of our findings. The EM algorithm showed promise in identifying meaningful clusters, taking into account both attribute information and data point distribution. Agglomerative Clustering also performed well, offering additional insights not captured by other techniques.

Based on our observations, we recommend using DBSCAN, EM, or Agglomerative Clustering for datasets similar to ours, as they provide more nuanced perspectives on the data's underlying structure. However, further work is necessary to assess the generalizability of our findings and evaluate the performance of these clustering techniques against different aspects of the data. Ultimately, the choice of clustering technique should be guided by the specific needs and goals of the project, considering the strengths and limitations of each method.

Appendix

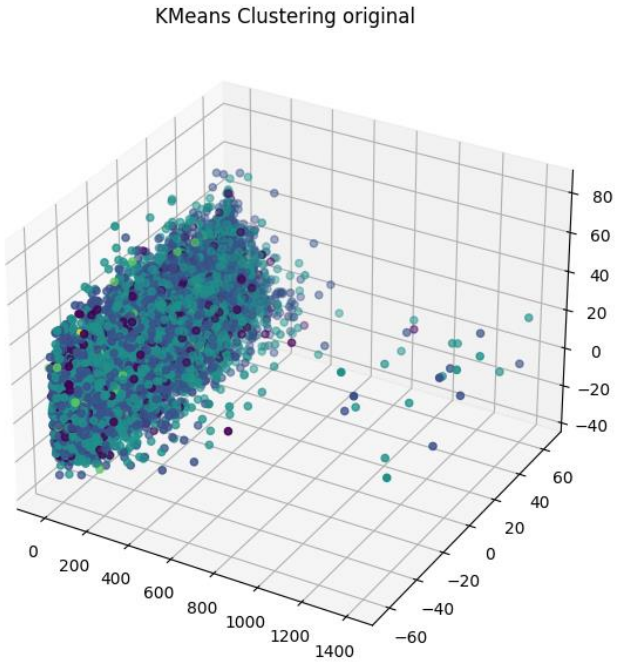

Figure 15: KMeans clustering for original dataset

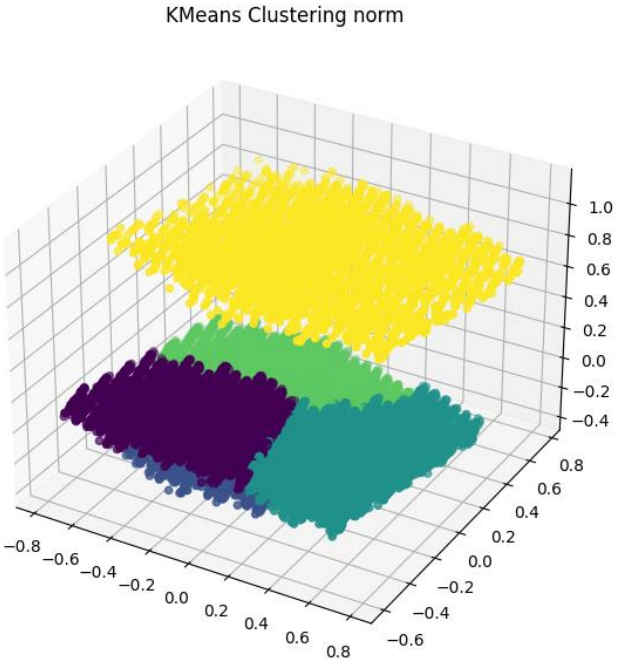

Figure 16: : KMeans clustering for normalized dataset

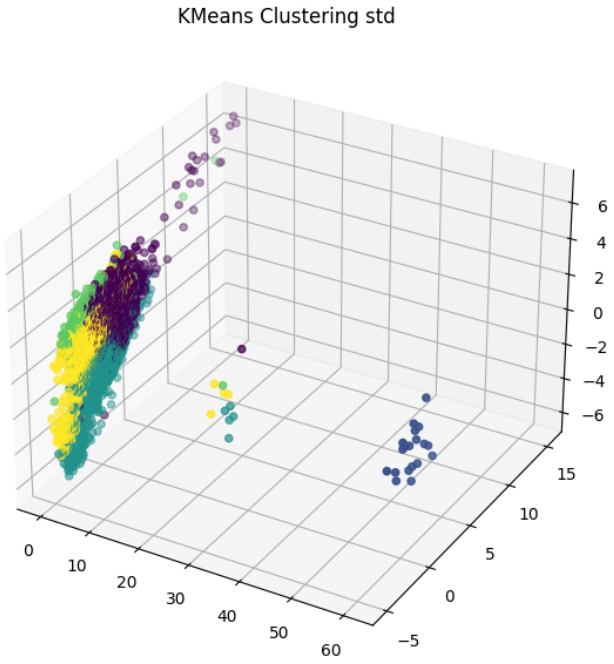

Figure 17: KMeans clustering for standardized dataset

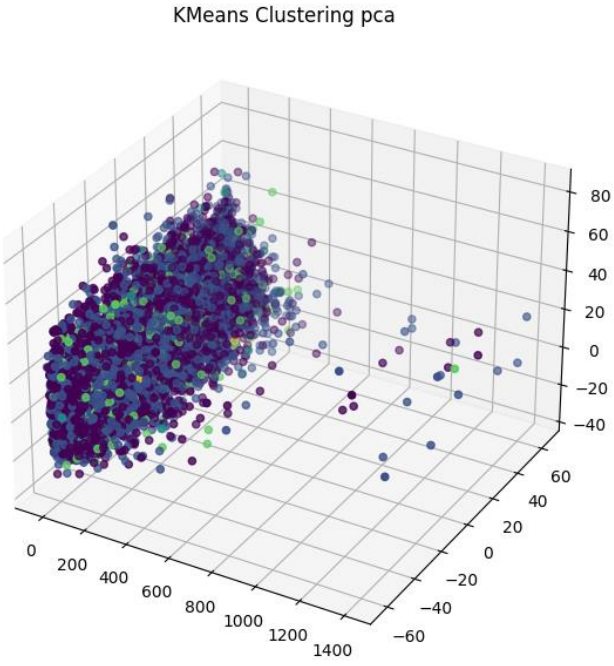

Figure 18: KMeans clustering after reduction dimension by PCA

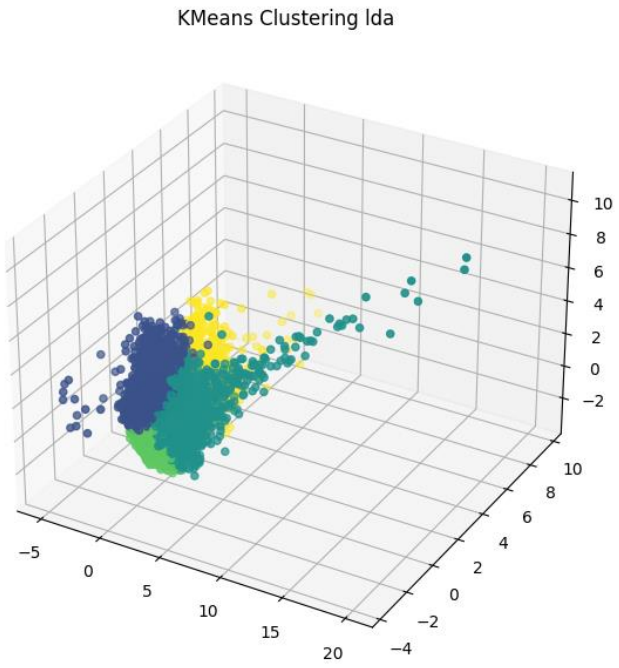

Figure 19: KMeans clustering after reduction dimension by LDA

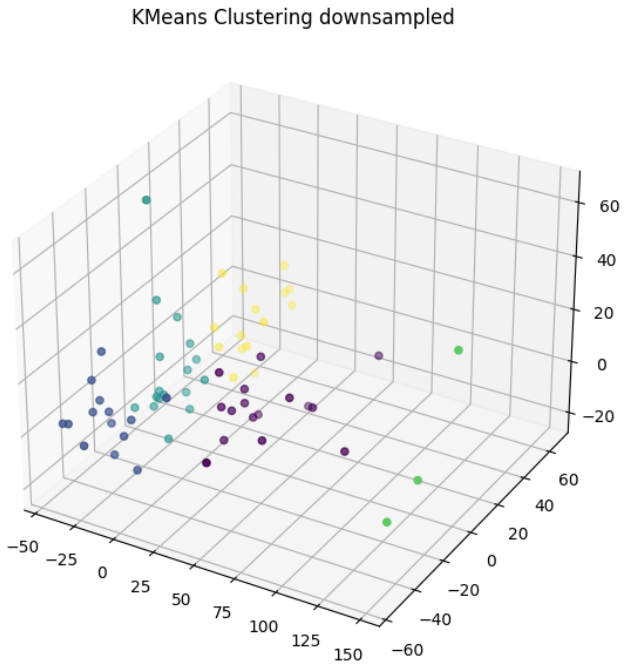

Figure 20: KMeans clustering for downsampled dataset

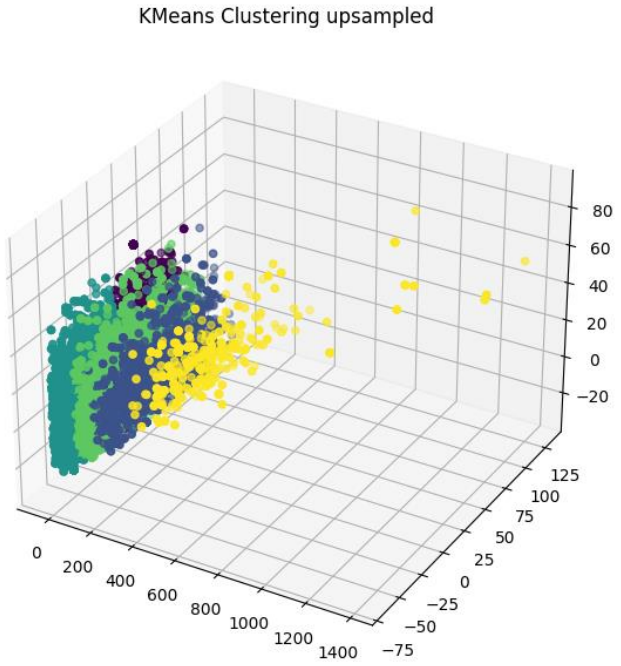

Figure 21: KMeans clustering for upsampled dataset

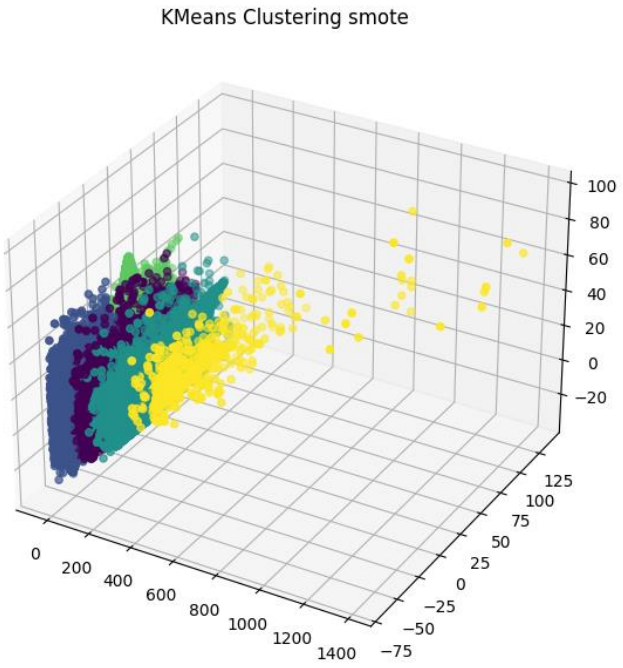

Figure 22: KMeans clustering for SMOTE dataset

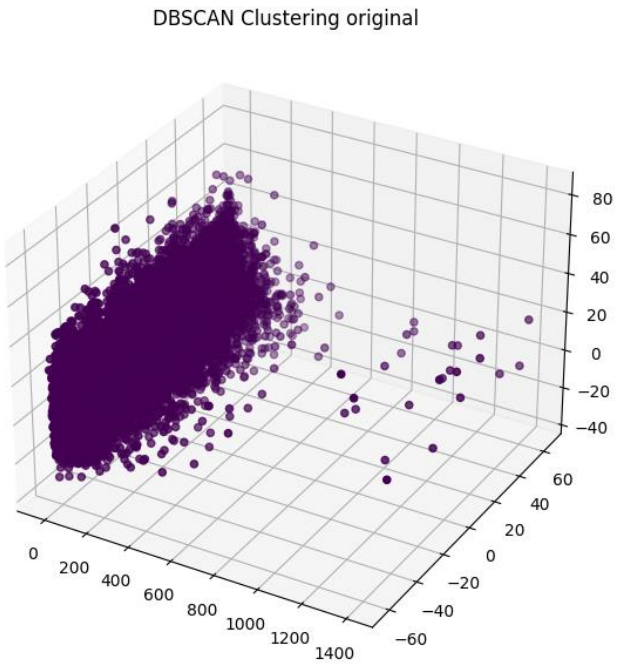

Figure 23: DBSCAN clustering for original dataset

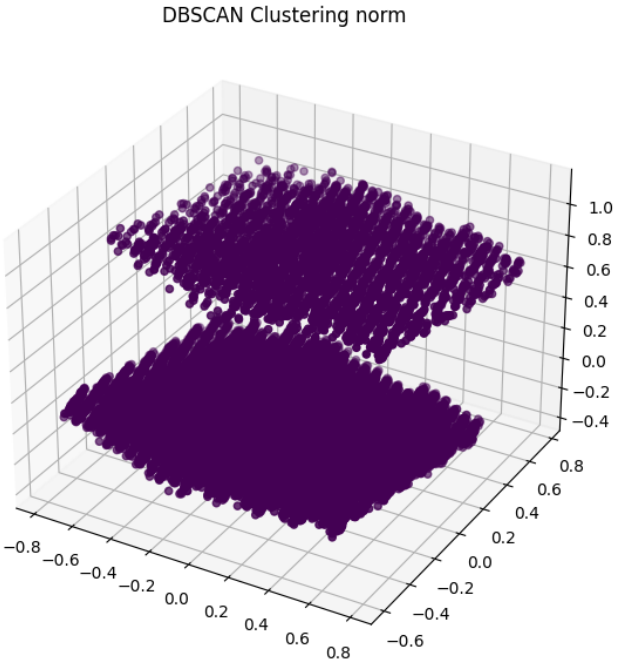

Figure 24: DBSCAN clustering for normalized dataset

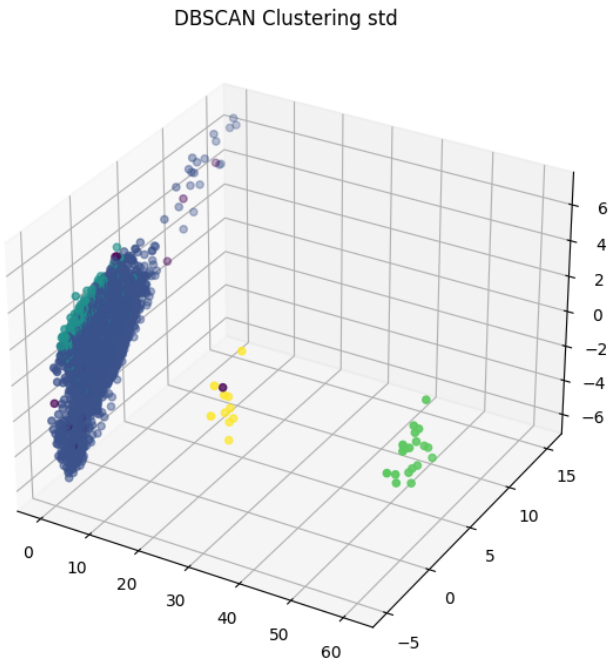

Figure 25: DBSCAN clustering for standardized dataset

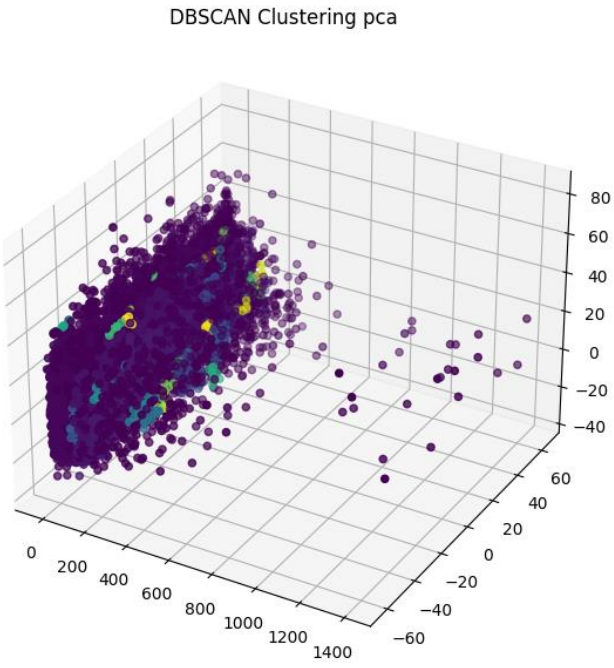

Figure 26: DBSCAN clustering after reduction dimension by PCA

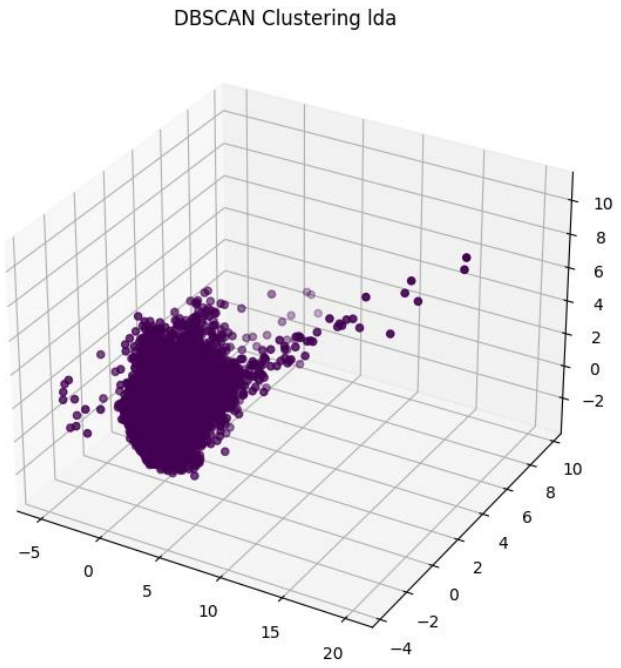

Figure 27: DBSCAN clustering after reduction dimension by LDA

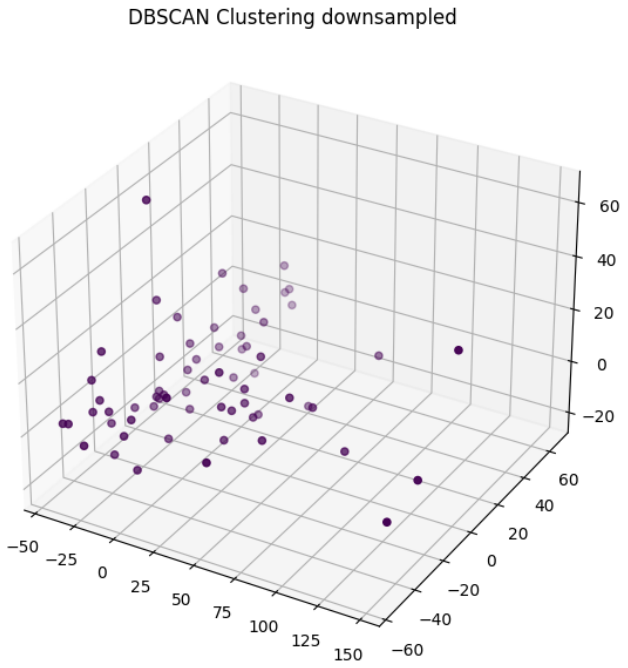

Figure 28: DBSCAN clustering for downsampled dataset

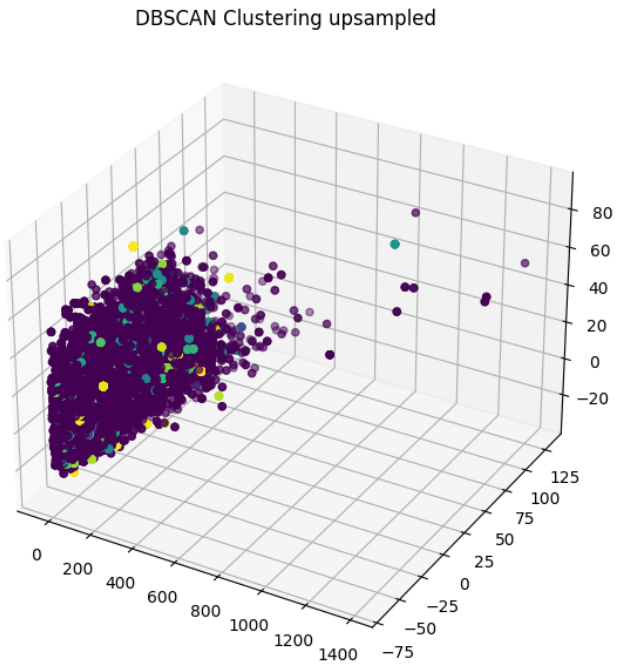

Figure 29: DBSCAN clustering for upsampled dataset

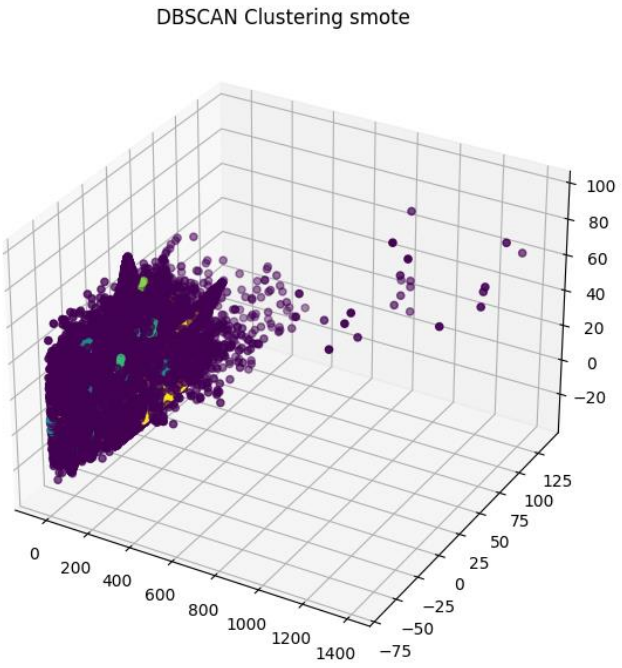

Figure 30: DBSCAN clustering for SMOTE dataset

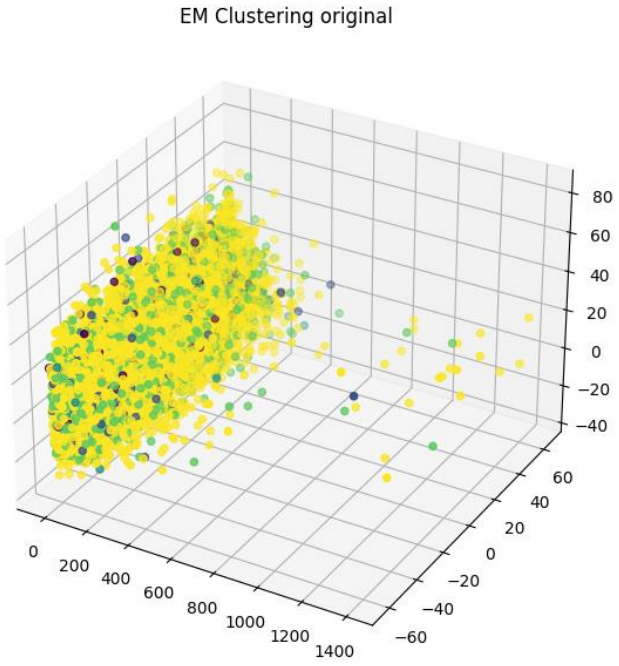

Figure 31: EM clustering for original dataset

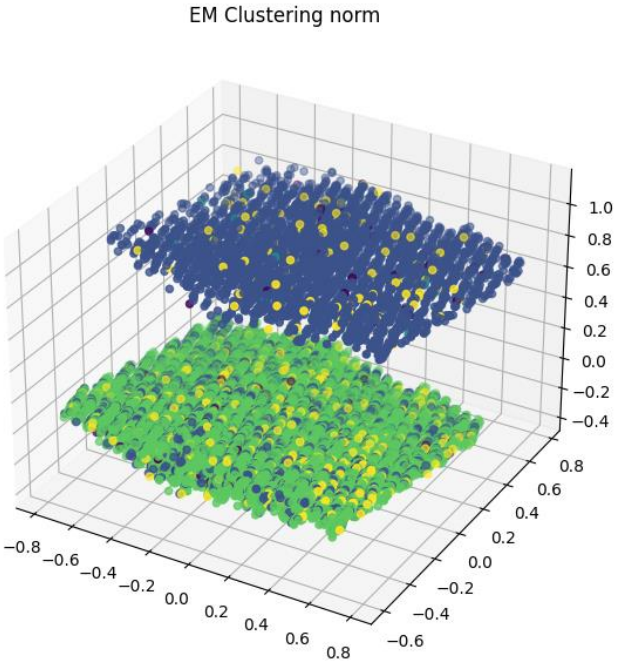

Figure 32: EM clustering for normalized dataset

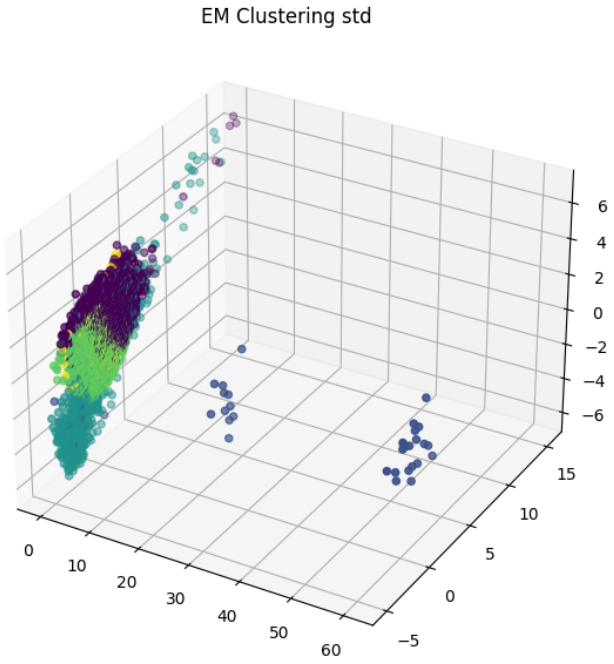

Figure 33: EM clustering for standardized dataset

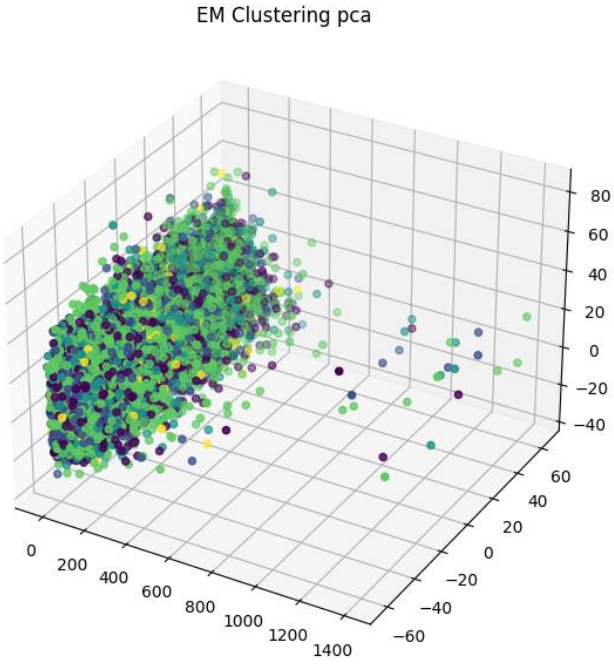

Figure 34: EM clustering after reduction dimension by PCA

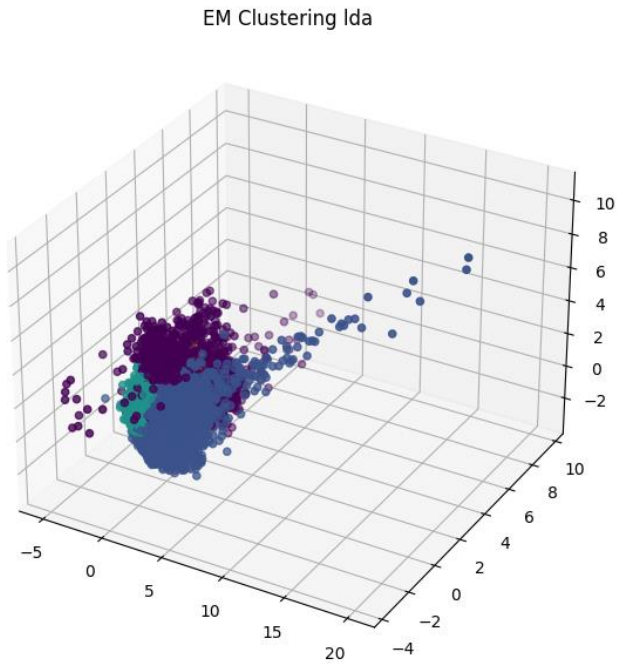

Figure 35: EM clustering after reduction dimension by LDA

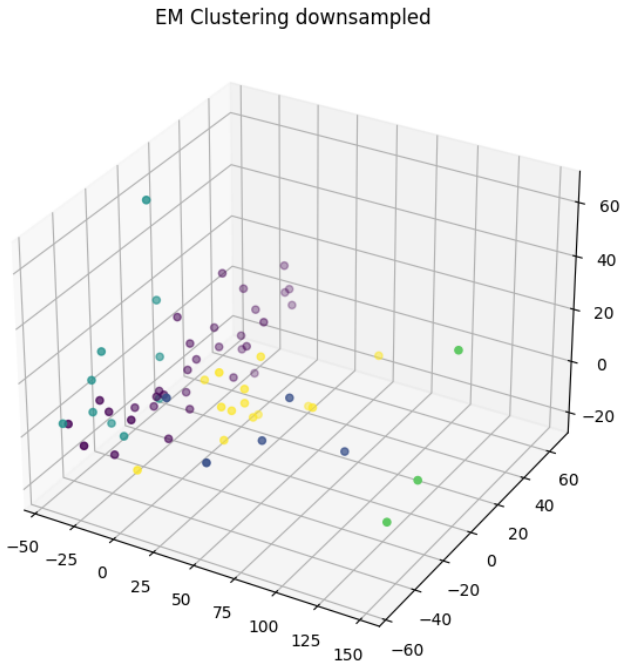

Figure 36: EM clustering for downsampled dataset

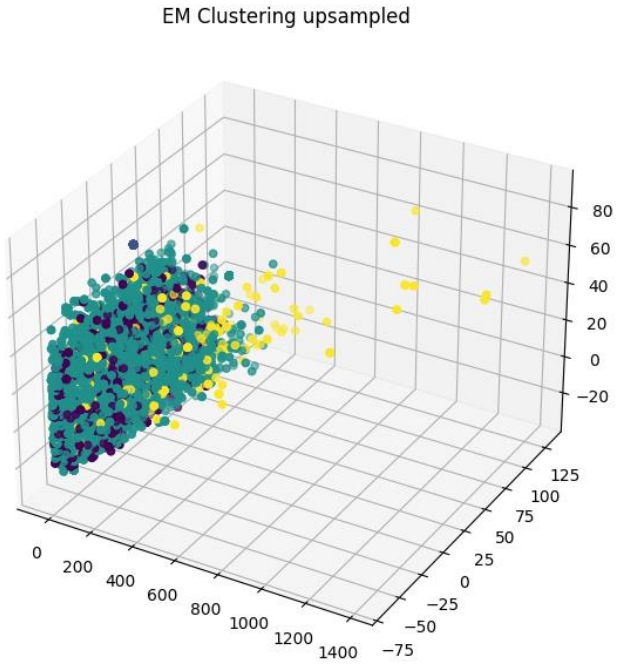

Figure 37: EM clustering for upsampled dataset

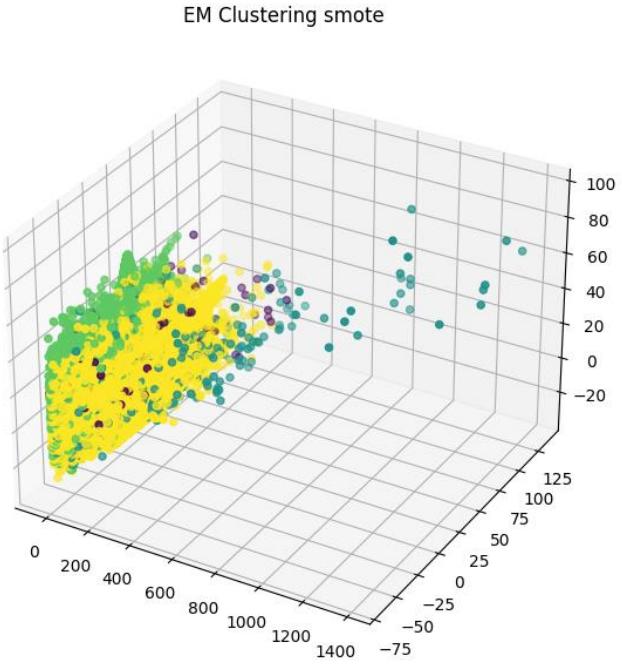

Figure 38: EM clustering for SMOTE dataset

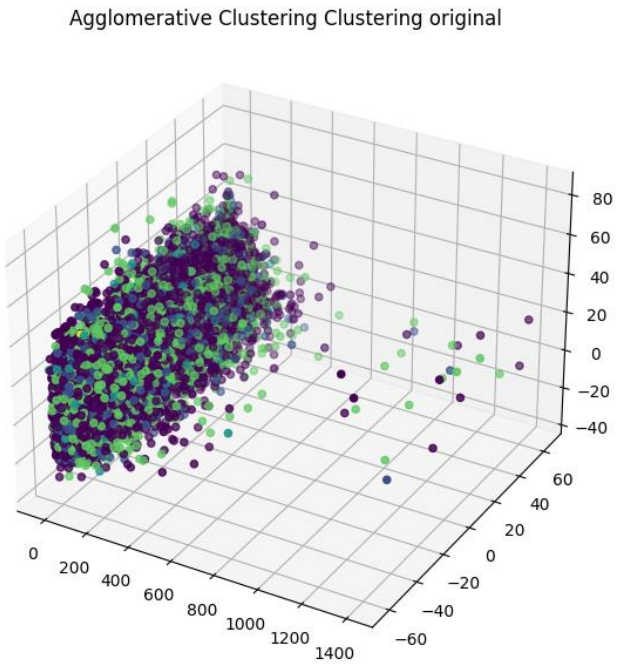

Figure 39: Agglomerative clustering for original dataset

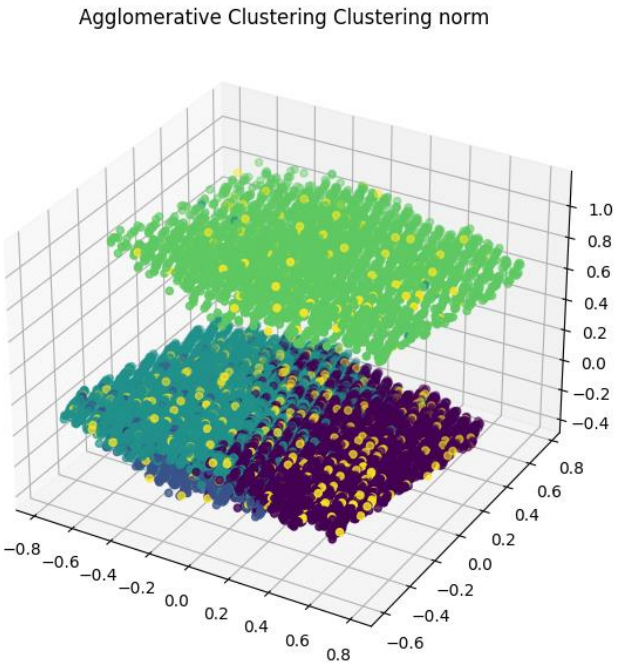

Figure 40: Agglomerative clustering for normalized dataset

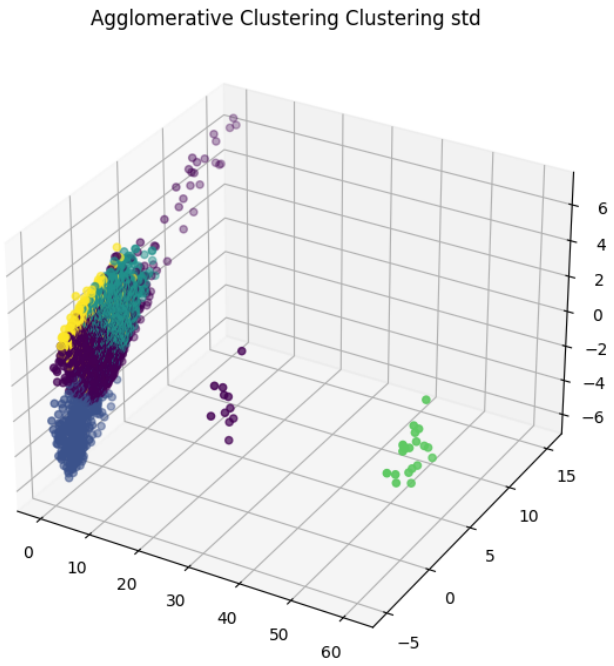

Figure 41: Agglomerative clustering for standardized dataset

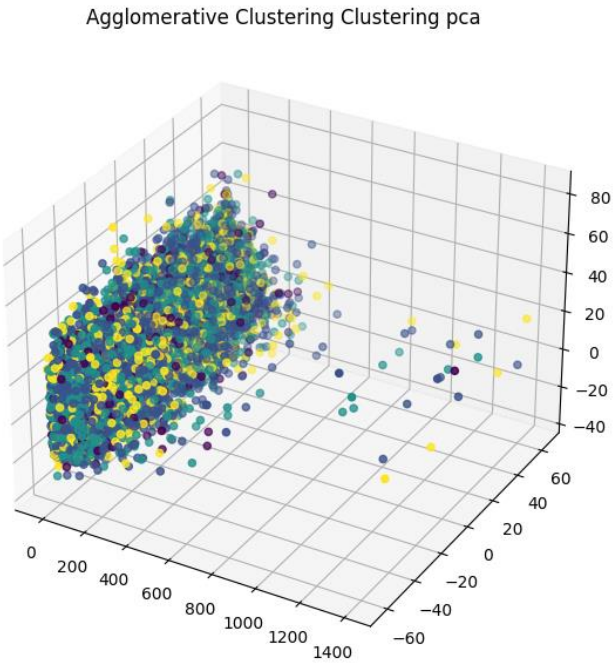

Figure 42: Agglomerative clustering after reduction dimension by PCA

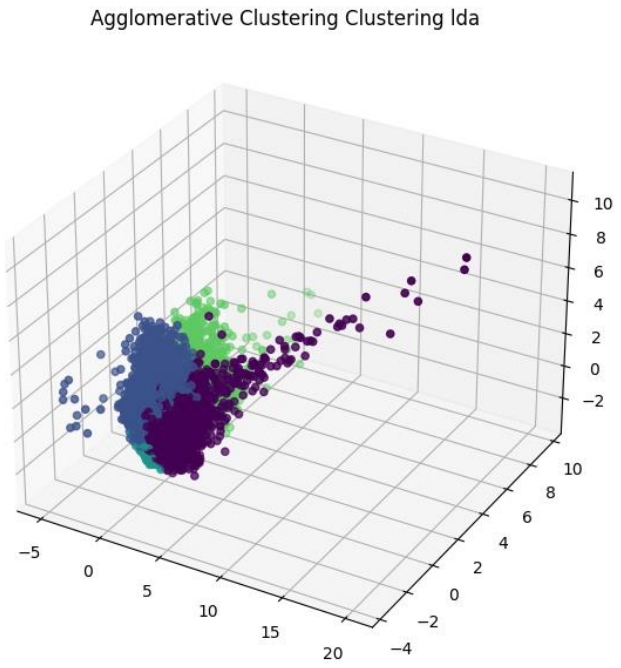

Figure 43: Agglomerative clustering after reduction dimension by LDA

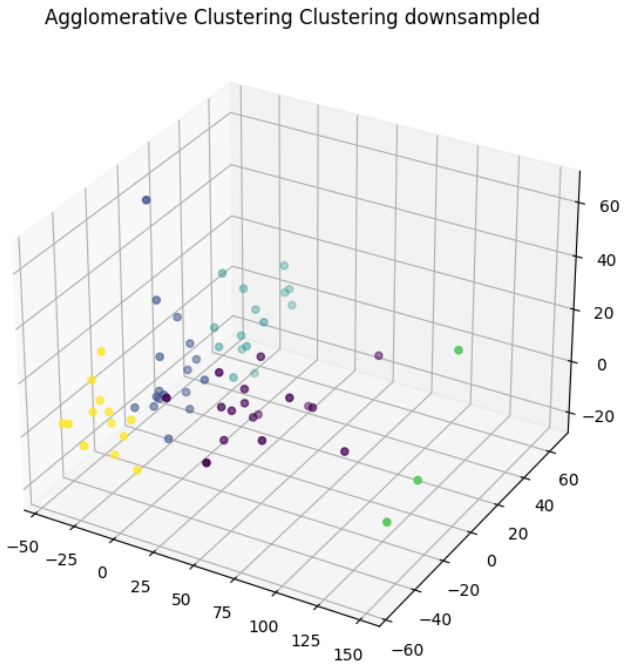

Figure 44: Agglomerative clustering for downsampled dataset