**Title:  Prediction of Accident Severity**

**CONTENTS:**

1. **Motivation**
2. **Significance**
3. **Objective**
4. **Features**

**Increment – I**

1. **Related work**
2. **Dataset**
3. **Detail Design of the methods and their Analysis**
   1. Preprocessing
   2. Finding the null and duplicate values in the dataset
   3. One Hot Encoding
   4. EDA – Exploratory Data Analysis
   5. Outliners plotted
   6. Outliners detected using IQR method
   7. Correlation with the target for IQR method
   8. Correlation Heat Map for the IQR method
   9. Plots of distribution of the numerical features for IQR method
   10. Plots of the distribution of the features with the target
   11. Clusters for IQR method
   12. SMOTE for IQR method
   13. PCA for IQR method
4. **Implementation**
   1. Libraries and tools used

**Part - I**

1. **Model Designed for IQR Outliners:**
   1. SVM (Support Vector Machine) Classifier for IQR
   2. Random Forest Classifier for IQR
   3. XGboost Classifier for IQR
   4. KNN (K- Nearest Neighbor) Classifier for IQR
2. **Preliminary Results of models designed for IQR Outliners:**
   1. SVM (Support Vector Machine) Classifier for IQR
   2. Random Forest Classifier for IQR
   3. XGboost Classifier for IQR
   4. KNN(K- Nearest Neighbor) Classifier for IQR
3. **Project Management - Implementation Report**
   1. Work Completed
   2. Description
   3. Responsibility (Task/ Person)
   4. Contribution (Member/ Percentage)
   5. Work To be Completed
   6. Description
   7. Responsibility (Task/ Person)
   8. Issues and concerns

**Increment - II**

**Part – II**

1. **Outliners using Z-Score**
   1. Outliners detected using Z-Score method
   2. Correlation with the target for Z-Score method
   3. Correlation Heat Map for the Z-Score method
   4. Plots of distribution of the numerical features for Z-Score method
   5. Plots of the distribution of the features with the target variable for Z-Score method
   6. Clusters for Z-Score method
   7. SMOTE for Z-Score method
   8. PCA for Z-Score method
2. **Model Designed for Z-Score Outliners:**
   1. SVM (Support Vector Machine) Classifier for Z-Score
   2. Random Forest Classifier for Z-Score
   3. XGboost Classifier for Z-Score
   4. KNN (K- Nearest Neighbor) Classifier for Z-Score
3. **Preliminary Results of models designed for Z-Score Outliners:**
   1. SVM (Support Vector Machine) Classifier for Z-Score
   2. Random Forest Classifier for Z-Score
   3. XGboost Classifier for Z-Score
   4. KNN (K- Nearest Neighbor) Classifier for Z-Score

**Part – III**

1. **Model Designed IQR without SMOTE and PCA:**
   1. SVM (Support Vector Machine) Classifier IQR without SMOTE and PCA
   2. Random Forest Classifier IQR without SMOTE and PCA
   3. XGboost Classifier IQR without SMOTE and PCA
   4. KNN (K- Nearest Neighbor) IQR without SMOTE and PCA
2. **Preliminary Results of models designed IQR without SMOTE and PCA:**
   1. SVM (Support Vector Machine) Classifier IQR without SMOTE and PCA
   2. Random Forest Classifier IQR without SMOTE and PCA
   3. XGboost Classifier IQR without SMOTE and PCA
   4. KNN (K- Nearest Neighbor) Classifier IQR without SMOTE and PCA

**Part – IV**

1. **Model Designed Z-Score without SMOTE and PCA:**
   1. SVM (Support Vector Machine) Classifier Z-Score without SMOTE and PCA
   2. Random Forest Classifier Z-Score without SMOTE and PCA
   3. XGboost Classifier Z-Score without SMOTE and PCA
   4. KNN (K- Nearest Neighbor) Classifier Z-Score without SMOTE and PCA
2. **Preliminary Results of models designed Z-Score without SMOTE and PCA:**
   1. SVM (Support Vector Machine) Classifier Z-Score without SMOTE and PCA
   2. Random Forest Classifier Z-Score without SMOTE and PCA
   3. XGboost Classifier Z-Score without SMOTE and PCA
   4. KNN (K- Nearest Neighbor) Classifier Z-Score without SMOTE and PCA
3. **References / Bibliography**
4. **Motivation:**

According to the World Health Organization (WHO), road traffic accidents caused an estimated 1.35 million deaths globally. The average rate of road traffic fatalities was 17.4 per 100,000 people. Low-income countries have a much higher annual road traffic fatality rate of 24.1 per 100,000 compared to high-income countries which have the lowest rate of 9.2 per 100,000.

A study discovered that driver experience, light conditions, and vehicle service year were the most significant factors contributing to fatal, severe, and light injuries, respectively.

Machine learning models can be used to analyse road accident data and predict possible accidents. This can help develop traffic safety control policies and identify patterns involved in dangerous crashes. The models can also be used to understand the primary and contributing factors that lead to road accidents, which can help develop preventive strategies.

1. **Significance:**

Nowadays, accidents are one of the major causes of death. These mainly occur randomly due to an external source that is not predicted, and this causes a harmful impact on life. Accidents can’t be avoided in many situations, but knowing the cause of accidents in different scenarios helps us prevent the occurrence of them to have a careful and safe ride.

Accidents might occur due to various reasons, such as human carelessness, vehicle defects, road conditions, and sometimes weather conditions.

Governments have come up with various rules and regulations to increase awareness while driving and educate regarding road safety.

Authorities and emergency services may more efficiently allocate resources, give priority to high-risk locations, and carry out focused actions to lessen the impact of accidents thanks to these predictive capabilities. In the end, a project like this may improve emergency response times, prevent fatalities, and save lives, all of which would make the transportation system safer and more durable.

1. **Objective:**

Accidents cause harm, damage, and loss and are unplanned. There might be small accidents and disasters sometimes even, which effect in physical, psychological, and emotional damage, which might impact communities and well-wishers including families.

To prevent them, it is important to understand the various reasons causing these accidents. This dataset contains different types of information about accidents based on the severity, which includes the exact latitude and longitude of the situation, the type of weather conditions, the state of the roads, and the vehicles used for driving along with the time of the incident.

Our project determines different types of accidents that may occur with severity, including weather conditions and obstacles on the road, along with the type of junctions and road types with reference to speed. This project intends to provide useful insights and support road safety.

1. **Features:**
   1. **Accident\_Index:** A unique identifier assigned to each reported accident.
   2. **Accident Date:** The date when the accident occurred.
   3. **Day\_of\_Week:** The day of the week when the accident took place.
   4. **Junction\_Control:** Describes the type of control at a road junction (e.g., traffic signals, roundabouts).
   5. **Junction\_Detail:** Provides details about the specific characteristics of a road junction.
   6. **Accident\_Severity:** Indicates the severity of the accident (e.g., slight, serious, fatal).
   7. **Latitude:** The geographic coordinate specifying the north-south position of the accident location.
   8. **Light\_Conditions:** Describes the lighting conditions at the time of the accident.
   9. **Local\_Authority\_(District):** The local administrative district where the accident occurred.
   10. **Carriageway\_Hazards:** Highlights any specific hazards present on the road.
   11. **Longitude:** The geographic coordinate specifying the east-west position of the accident location.
   12. **Number\_of\_Casualties:** The total number of casualties (injuries or fatalities) in the accident.
   13. **Number\_of\_Vehicles:** The total number of vehicles involved in the accident.
   14. **Police\_Force:** The law enforcement agency that reported and handled the accident.
   15. **Road\_Surface\_Conditions:** Describes the condition of the road surface at the time of the accident.
   16. **Road\_Type:** Specifies the type of road where the accident occurred (e.g., roundabout, dual carriageway).
   17. **Speed\_limit:** The legal speed limit for the road where the accident occurred.
   18. **Time:** The time of day when the accident happened.
   19. **Urban\_or\_Rural\_Area:** Indicates whether the accident occurred in an urban or rural area.
   20. **Weather\_Conditions:** Describes the weather conditions at the time of the accident.
   21. **Vehicle\_Type:** Specifies the type of vehicle involved in the accident (e.g., car, motorcycle).

**Increment – 1**

1. **Related work:**

This section represents a comprehensive review of the current research on deceasing the severity of the accidents. For the study, we used the US road accident dataset by Rohit Raj Jalheria, accident records causing in US for the year 2021-2022. To mitigate the negative effects of traffic accidents, a review of the literature's primary external and internal risk factors for accidents was conducted.

Most of the research covered in the evaluations of the literature was conducted using a small amount of data, or by taking only some features into the consideration, or in artificial environments. Determining the underlying connections between accident contributing variables and injury severity is essential since estimating the severity of road accidents is difficult and complex, especially when there are several contributing features are involved. The traditional statistical and regression technique, which is essentially predicated on the assumption of linear and non-linear correlations between input and output parameters, has been employed in most of the study on this issue, as was previously indicated. On the other hand, ML-based feature analysis has been the subject of much research recently. Because of this, some ensemble bagging techniques—such as Random Forest, Logistic Regression, Decision Tree, SVM, and XGboost techniques. Most of the models listed above focused about classification or prediction accuracy. To interpret the relationship between characteristics (such as contributing variables to traffic accidents) and accident severity, model interpretation is crucial when working with large datasets that contain several features.

Furthermore, they created a framework for real-time data mining on transportation networks. This study offered a good analysis of road accidents, but unfortunately, the lack of sufficient data prevented any firm conclusions from being made. We discovered several knowledge gaps in accident severity prediction by carrying out an extensive literature study. The majority of earlier research studies only made predictions about i) one or two factors, which is insufficient for a real-world scenario ii) many studies ignore the issue of class imbalance; iii) unobserved heterogeneity; and iv) most studies only use one accuracy measure to assess the algorithm's performance. Consequently, this study's main objective is to close the previously described knowledge gaps.

1. **Dataset:**

Dataset is taken from Kaggle and this is related to the accidents which took place in USA during 2021-2022 [1]

This dataset has total 21 columns and 307973 rows.

1. **Detail Design of the Methods and their Analysis:**

Installation part for implementing Methods:

1. Scikit-learn (1.2.2 and 1.1.3)
2. Imbalance-learn

A black background with white text

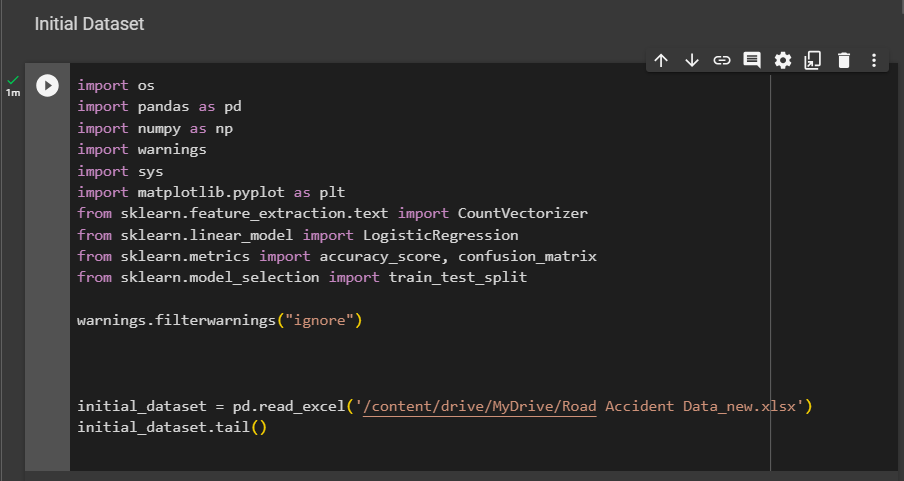
Description automatically generated

A black background with white text

Description automatically generated

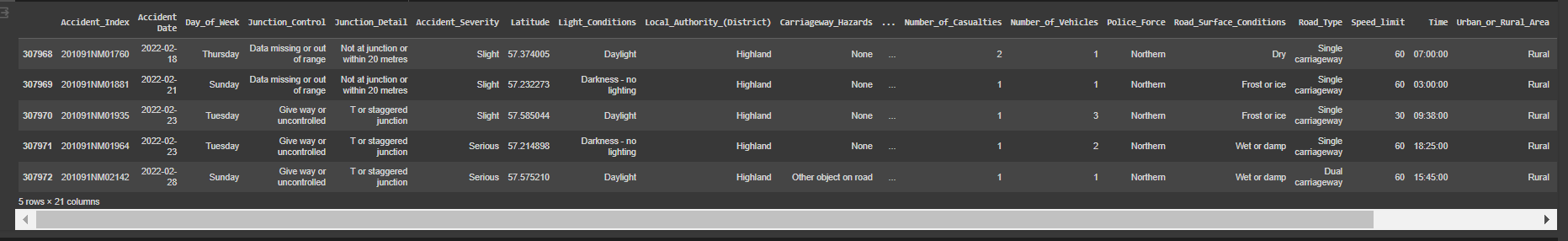
***fig: libraries installed***

Uploaded the dataset: it has 21 columns and 307973 rows, and most of the data is in the object-string format.



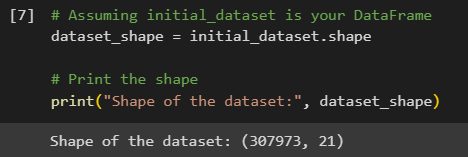
***fig : loading data***

Output of the head of the dataset after loading:



***fig: tail part of the dataset***

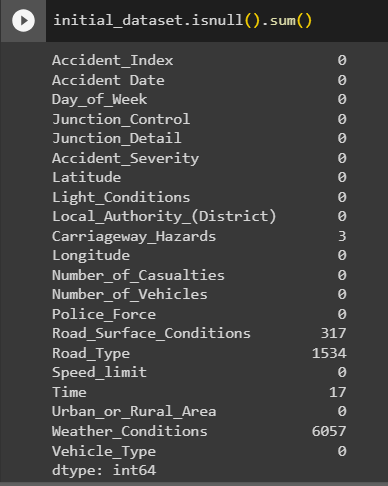
This is the shape of the dataset:



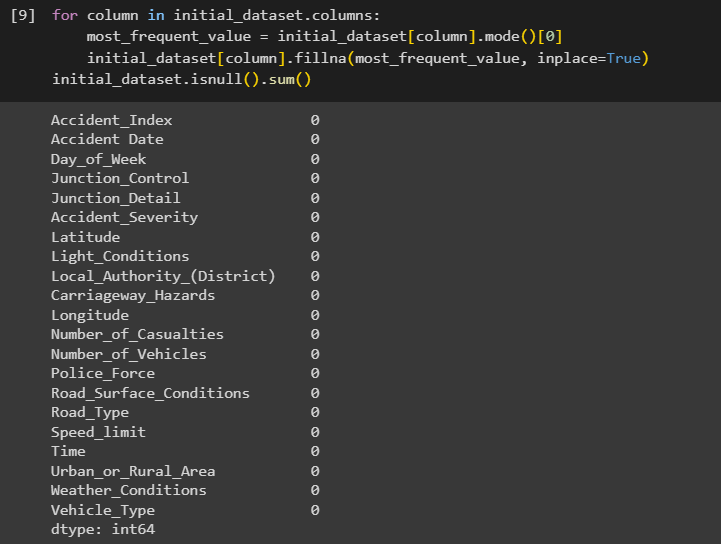
***fig: dataset shape***

* 1. **Pre-processing:**
  2. **Finding the null and duplicate values in the dataset:**

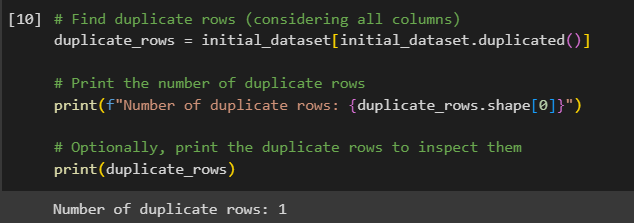
This whole dataset has the null values in the columns Road\_Surface\_Condition, Road\_Type, Time,Weather\_Conditions. So to remove these null values, replaced the values with the most frequent values in the dataset. When comes to duplicate values, dataset has only 1 duplicated value, which is dropped in cleaning.



***fig: before deleting the null values***

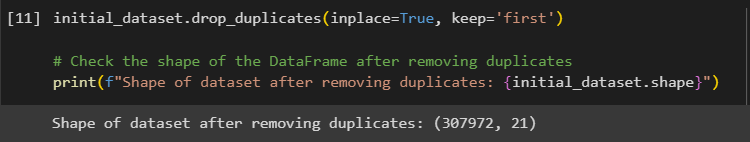


***fig: after deleting the null values***



***fig: duplicate rows***

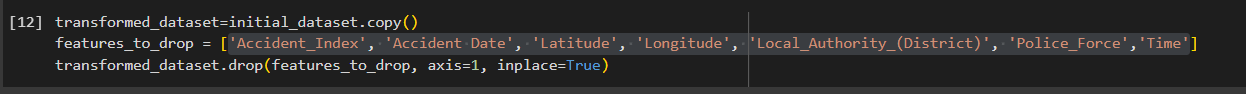
After deleting the duplicate value the shape of the dataset is changed to : (307972, 21).



***fig: after deleting the duplicate rows***

Now found the unique values in the dataset for each row and changing them into the int values using ONE-HOT Encoding.

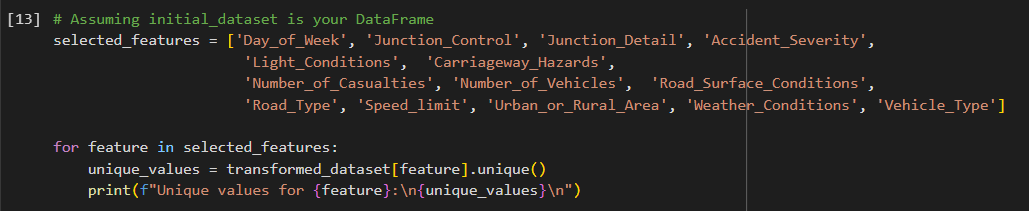
For this, the initial\_dataset copy is given to the transformed\_datset and dropped some of the columns, which are not being used in the further implementation. 'Accident\_Index', 'Accident Date', 'Latitude', 'Longitude', 'Local\_Authority\_(District)', 'Police\_Force', 'Time' are the columns which are dropped. Rest all the columns are converted using the one-hot encoding process.



***fig: storing the original dataset into the transformed dataset***

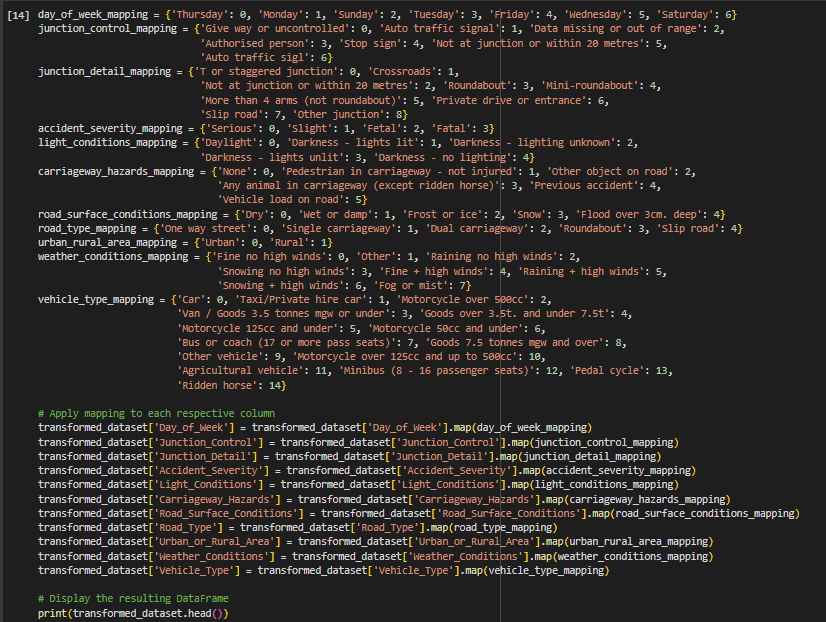
Before changing the columns, found out the unique values in each columns.

* 1. **ONE-HOT Encoding:**



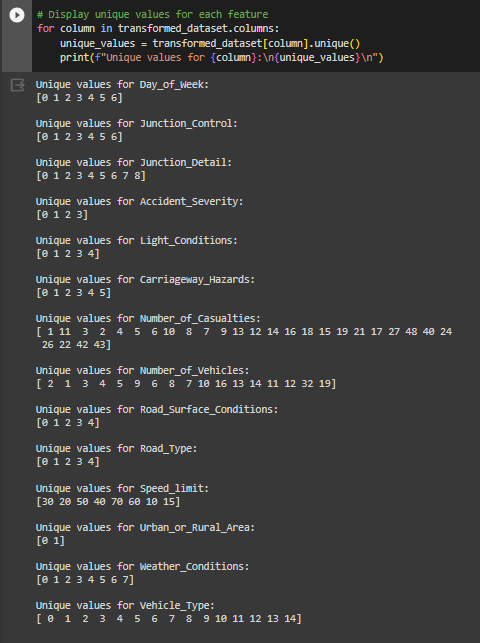
***fig: finding the unique values***

After this each unique value in the column are assigned with the values manually.



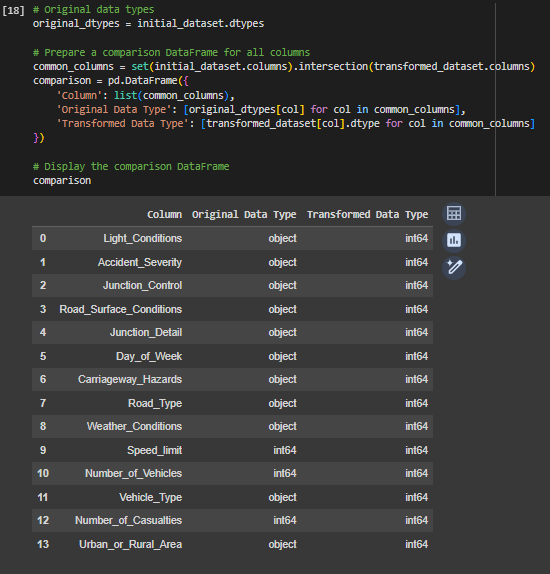
***fig: assigning the int values to each unique value***

After changing each value, unique values are printed from each column.



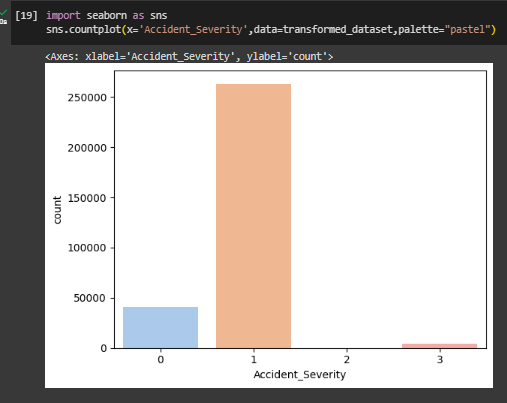
***fig: unique values***

Once after all the columns in the datasets which are needed for the different model are converted, both dtypes before converting into int and after converting are compared for better understanding.



***fig: overall datatypes comparison before and after changing the values***

After this target value is Visualized to get the original number of values individually.



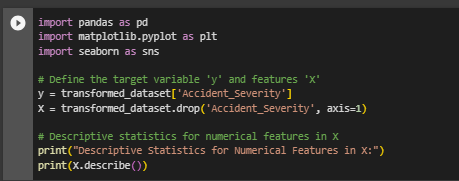
***fig: showing the target value***

As there is a miss classification and imbalance in the target value SMOTE should be applied to equate all the target values. Before doing this done the EDA part to get even more better understanding of the features and the target values.

* 1. **EDA (Exploratory Data Analysis):**

This is important to get the visualization of the features related with the target variable which is being used and the distribution of data before model created.

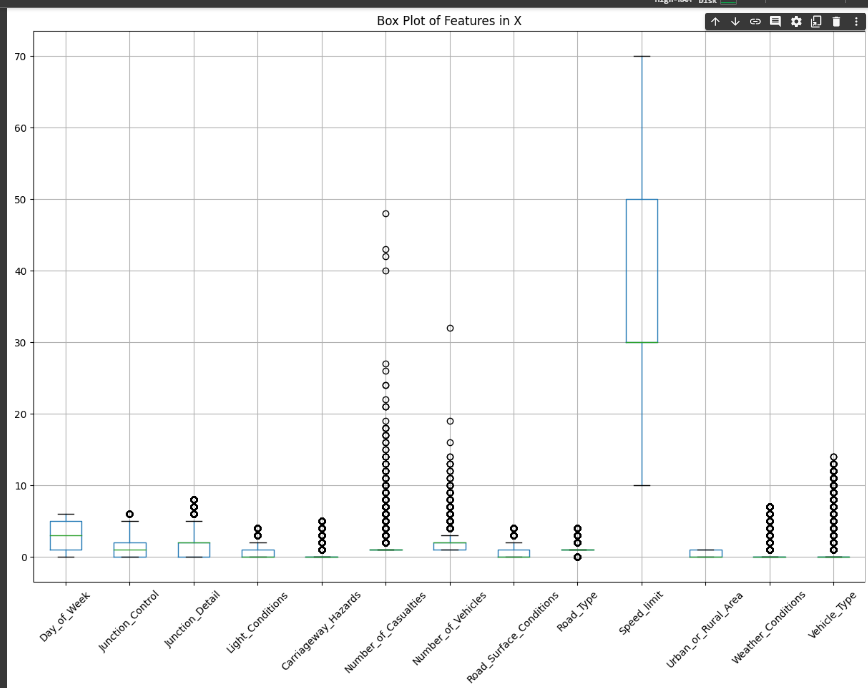
In this dataset Target: ‘Accident\_Severity’ which is represented by the y variable.



***fig: data used for EDA***

Outliners: Here outliners are plotted and then detected.

* 1. **Outliners Plotted:**

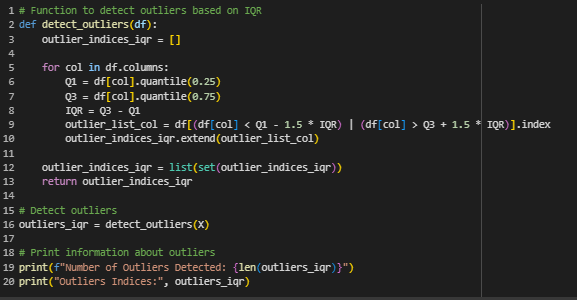


***fig: outliners***

* 1. **Outliners Detection using IQR method:**

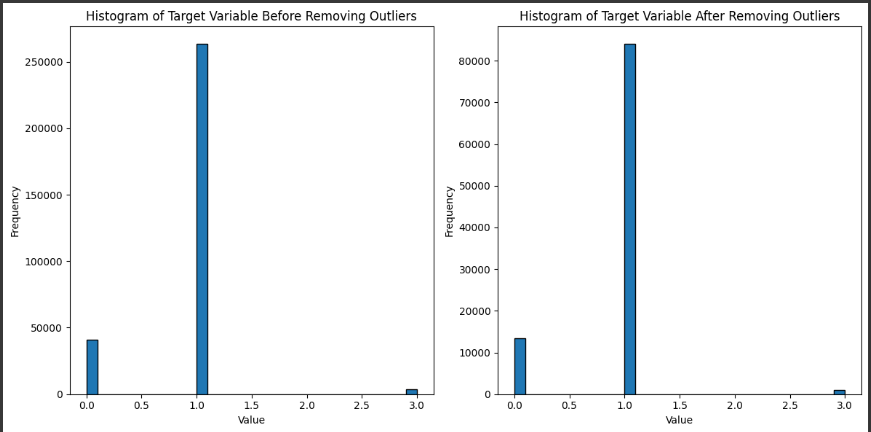
Outliners are the datapoints which in long distances from the more datapoints regions. It is important to detect the outliners to get more accuracy and good insights in the dataset spreading and data distribution. These are the errors that happened during the entry of the data.

IQR means Interquartile Range, this is one of the techniques to detect the outliners, where it will be concentrating more on the middle part of data. it will divide the whole data points into two quartiles and the median of each is found and these called as the Lower Bound and Upper Bound. Once after they are found, lowest bound is subtracted from highest bound and the resulting value is the outliner. Mostly to do this we will add 1.5 to the first quartile and then Multiply with the feature point of the data for lower bound. For upper instead of adding 1.5 we will subtract 1.5 value, to get the 25th% for Q1 and the 75th% for Q2.

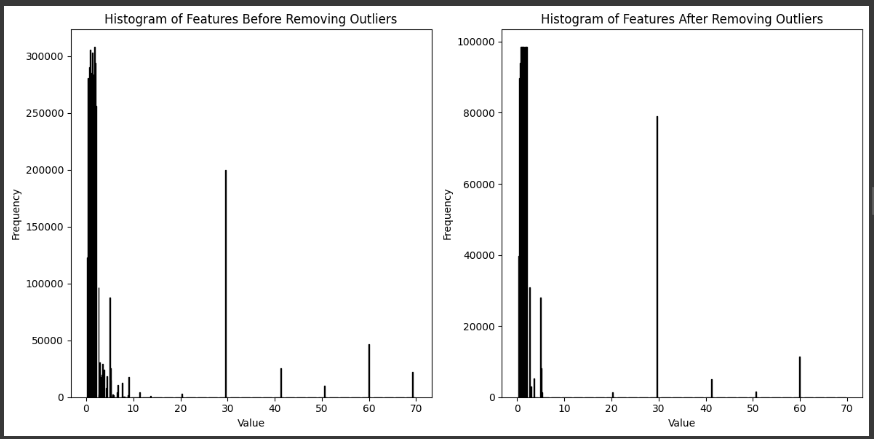


***fig: outliners detected***

Here Q1 represents the lower bound and Q2 represents the upper bound. Overall are outliners are 209509. Now all these outliners are dropped. Which decreased the dataset shape to (98463,13). Here we can see 13 columns instead of 21 because, we dropped some of the columns while changing dataset into the int values.



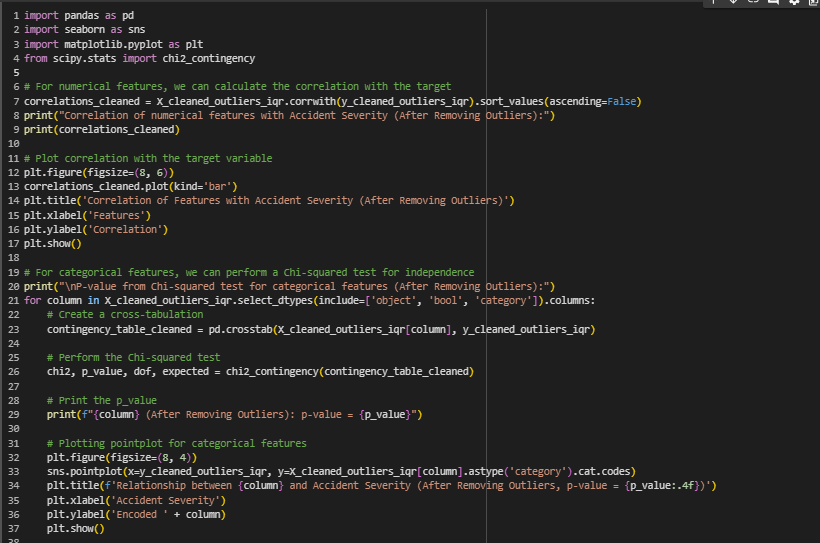
***fig: plots for the target before and after removing the outliers.***



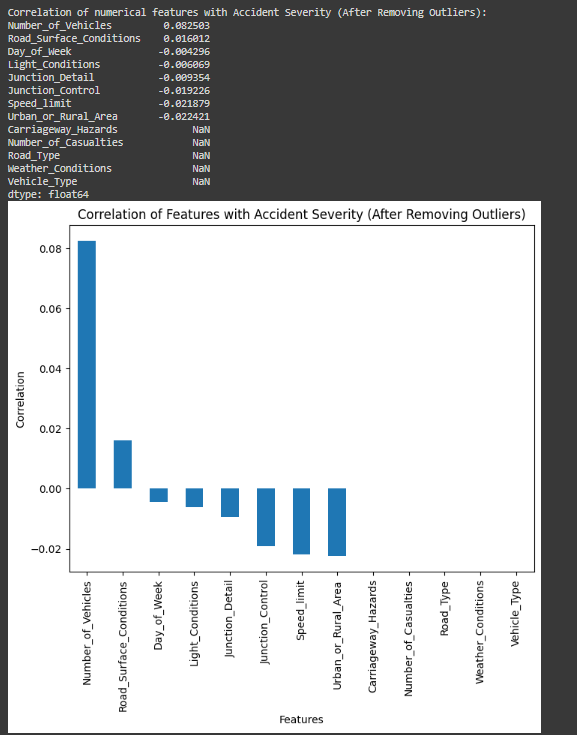
***fig: plots for the features before and after removing the outliners***

* 1. **Correlation with the target for IQR outliners:**

It is found with the outliners cleared data and we found some positive values and some negative values along with NAN values for some of the features.



***fig: program to find the correlation***



***fig: output of the correlation data***

**Features with the positive correlation:**

|  |  |
| --- | --- |
| Number\_of\_Vehicles | 0.082503 |
| Road\_Surface\_Conditions | 0.016012 |

**Features with the negative correlation:**

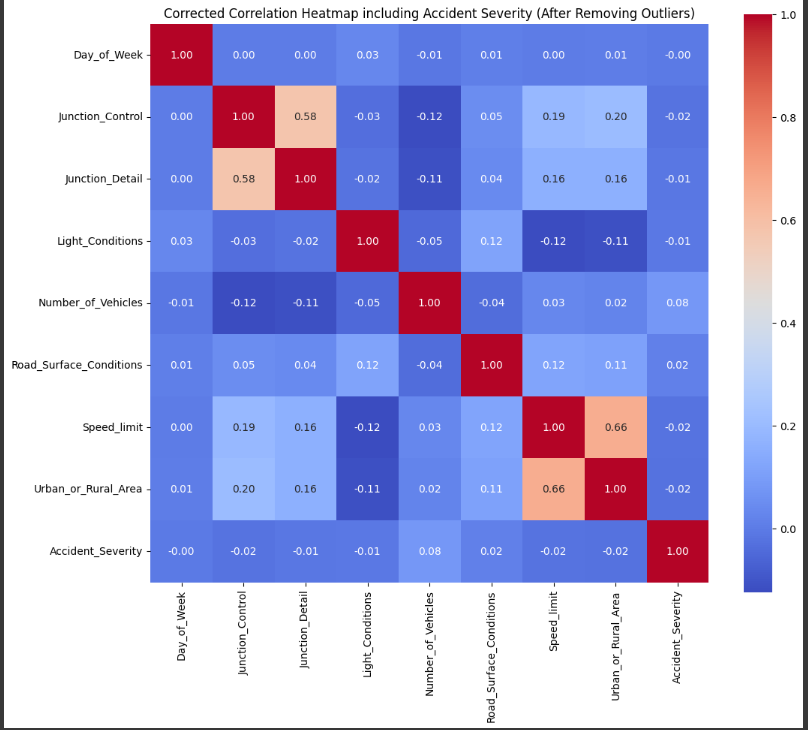
|  |  |
| --- | --- |
| Day\_of\_Week | -0.004296 |
| Light\_Conditions | -0.006069 |
| Junction\_Detail | -0.009354 |
| Junction\_Control | -0.019226 |
| Speed\_limit | -0.021879 |
| Urban\_or\_Rural\_Area | -0.022421 |

**Features with the NAN values:**

|  |  |
| --- | --- |
| Carriageway\_Hazards | NaN |
| Number\_of\_Casualties | NaN |
| Road\_Type | NaN |
| Weather\_Conditions | NaN |
| Vehicle\_Type | NaN |

* 1. **Correlation Heatmap for IQR outliners:**

Correlation is found between the outliners cleaned data and the target variable, and the overall correlation is stored in the variable correlation\_matrix\_cleaned.



***fig: correlation heatmap***

* 1. **Plots of Distribution of the numerical features for IQR outliners:**

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

***fig: distribution of the numerical features after the outliners is found***

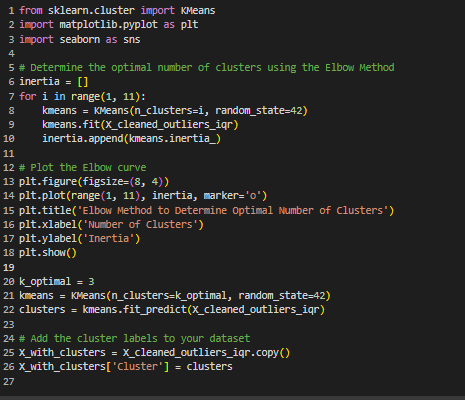
* 1. **Plots of Distribution of the features with the target variable for IQR outliners:**

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

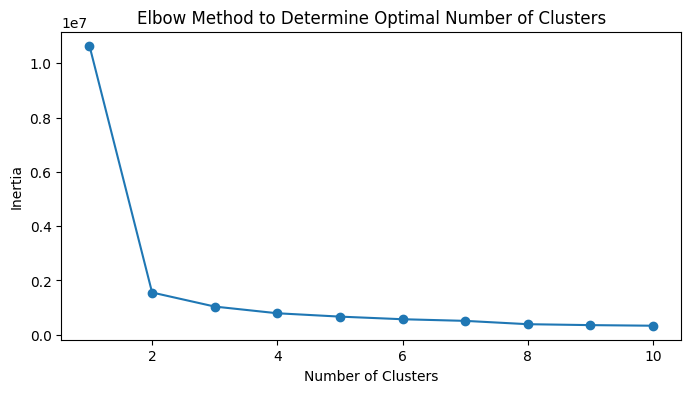
***fig:distribution of the features with the target variable***

* 1. **Clusters for IQR outliners:[7]**

Clusters are small groups of objects which are similar to other groups. K means is used to find the variance in the minimum level in each cluster. These are used to understand the differences in the groups and their characteristics.



***fig: clusters program***



***fig: clusters using the elbow method***

For finding the clusters elbow method has been used. In this we will be concentrating more on the range of clusters by defining the k values.

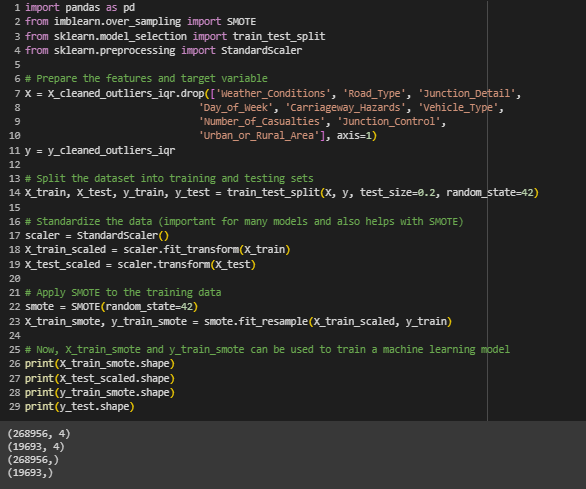
After removing outliners the shape f the X and y are defined by X\_cleaned and y\_cleaned .X\_cleaned shape = (98463, 13) , y\_cleaned shape = (98463,)

As the target value has the imbalance to balance it , SMOTE technique is used, here the input is the X\_cleaned which is the after the outliners deletion.

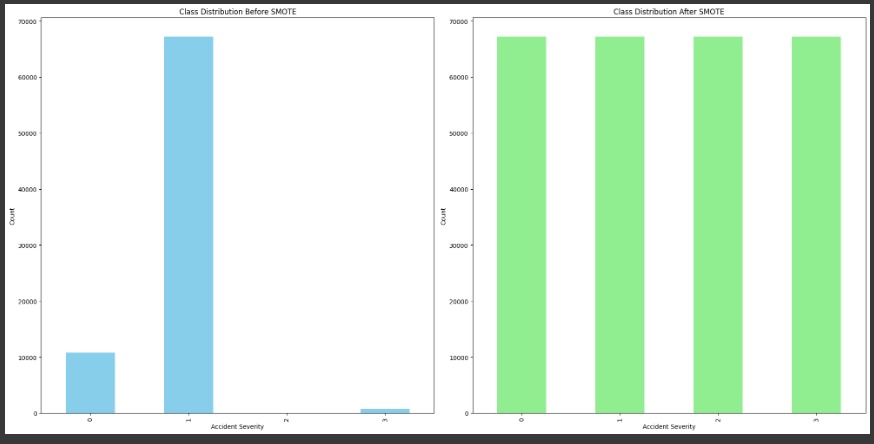
* 1. **SMOTE for IQR outliners:**

Synthetic Minority Over-sampling Technique is used for balancing the target value. Because if one class has more instance and the other have less, this will create the effect to the performance of the model.

By doing this, more samples are added to the less class instances which will increase the class instances and the performance will be improved.



***fig: SMOTE program***

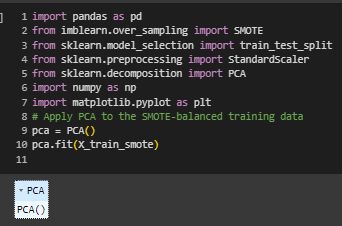


***fig: target values before and after smote***

* 1. **PCA for IQR outliners:**

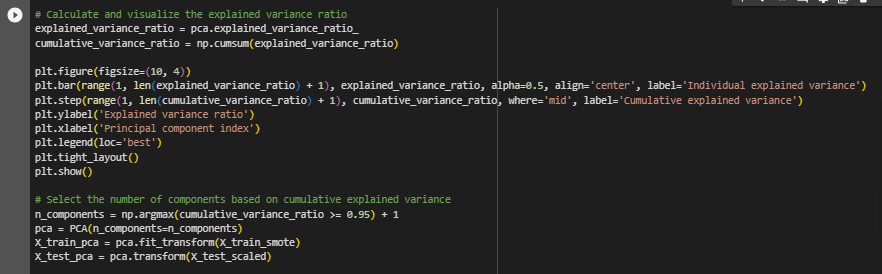
Principle Component Analysis is mainly important for the reduction of the dimensions of the dataset, which decrease the complexity of the data by possible correlated variables being converted in to the uncorrelated viable as PC.

This is used for the large datasets, which might be overfitting and have a issue to visualize. Simplification of the data is done ,by handling the multicollinearity data and use for noise reduction.

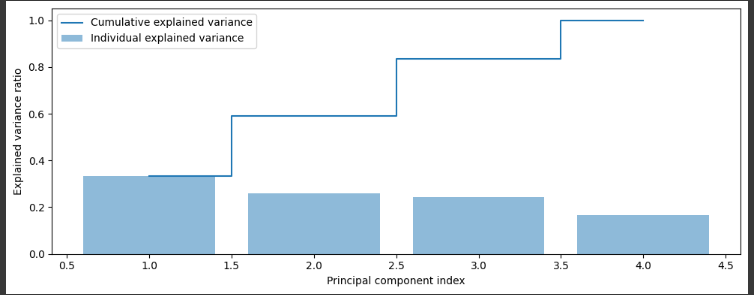


***fig: PCA model***

Here the input will the smote x variable trained data, using this variance is calculated, after creating a PCA model and fitting it.

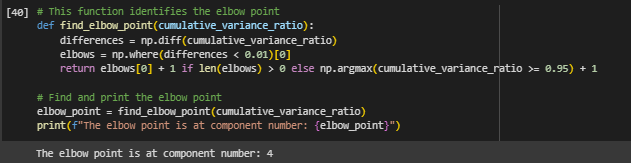


***fig: variance***



***fig: Variance of PCA***

Using the elbow point cumulative variance is calculated and the component number is 4.



***fig: elbow point cumulative variance***

1. **Implementation:**
   1. **Libraries and Tools used:**

**OS**: An operating system-interactive Python library that is used to manage file paths and directories inside of projects.

**Pandas**: Pandas is a robust data manipulation package that offers flexible data structures like dataframes and is used to handle and analyze structured data.

**Numpy** is a core Python module for numerical operations that is used in machine learning projects for effective array manipulation and mathematical computations.

**warnings**: This Python module manages error messages that arise while running code to guarantee cleaner outputs and smoother code execution.

**sys**: A module that gives users access to certain variables that the Python interpreter uses or maintains; it is usually used for parameters and functions that are unique to the system.

**matplotlib.pyplot**: An all-inclusive charting toolkit that improves data exploration and analysis through the creation of static, animated, and interactive visuals in Python.

**CountVectorizer** (from sklearn.feature\_extraction.text): A text-based machine learning tool, frequently used for natural language processing, that transforms a set of text documents into a matrix of token counts.

**LogisticRegression** (from sklearn.linear\_model): This project's predictive modeling uses an implementation of logistic regression, a well-liked approach for binary classification problems.

Model performance measures, such as accuracy and confusion matrix, are provided via the accuracy\_score and confusion\_matrix functions (from sklearn.metrics) to evaluate classification outcomes.

**train\_test\_split** (from sklearn.model\_selection): An essential tool for validating and evaluating models, this tool divides datasets into training and testing sets.

**seaborn**: A Matplotlib-based data visualization package used to produce eye-catching and educational statistics visualizations.

**chi2\_contingency (from scipy.stats):** A function for conducting a chi-square test of independence, useful for analyzing relationships between categorical variables in contingency tables.

**KMeans (from sklearn.cluster):** An implementation of the KMeans clustering algorithm, utilized for unsupervised clustering of data points based on similarity.

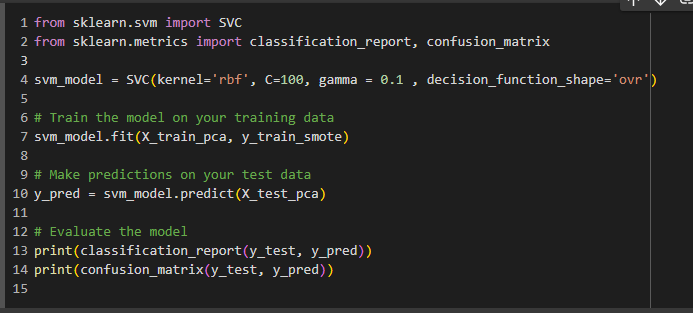
**SMOTE (from imblearn.over\_sampling):** A technique for handling imbalanced datasets by oversampling the minority class, enhancing model performance on rare instances.

**StandardScaler (from sklearn.preprocessing):** Used for standardizing features by removing the mean and scaling to unit variance, a crucial step in preparing data for machine learning models.

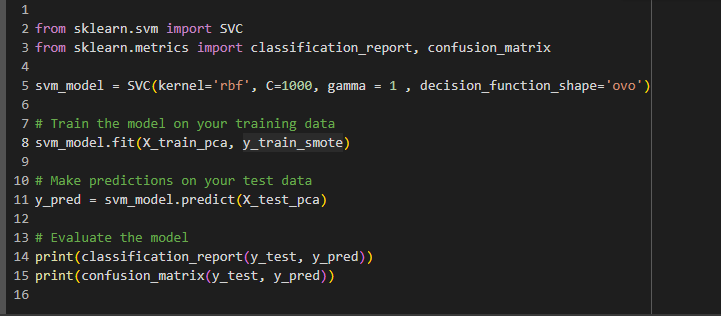
Random Forest, SVM, XGBoost: Algorithms implemented from scikit-learn for machine learning tasks, including Random Forest for ensemble learning, SVM for support vector machines, and XGBoost for boosting-based models.

1. **Models Designed for IQR outliners:** 
   1. **SVM for IQR outliners:**

In SVM I have used the rbf, while selecting hyper tuning parameters as C=100 and gamma = 0.1, I used the decision function as ovr since the target has 4 different values init. Ovr represents one vs rest. After this I used the smote data ain the y parameter and pca in the X and fitted the model and then predicted the model. The overall output has 51 accuracy.



***fig: model of SVM for rbf C- 100, gamma - 0.1, decision\_function\_shape – ovr***

******

***fig: model of SVM for rbf C- 1000, gamma – 1, decision\_function\_shape – ovo***

For linear, C= 100 and gamma = 0.1 got the accuracy of 85, here I have choosed the trainng data of 50000 rows

******

***fig: model of SVM for linear C- 100, gamma – 0.1***

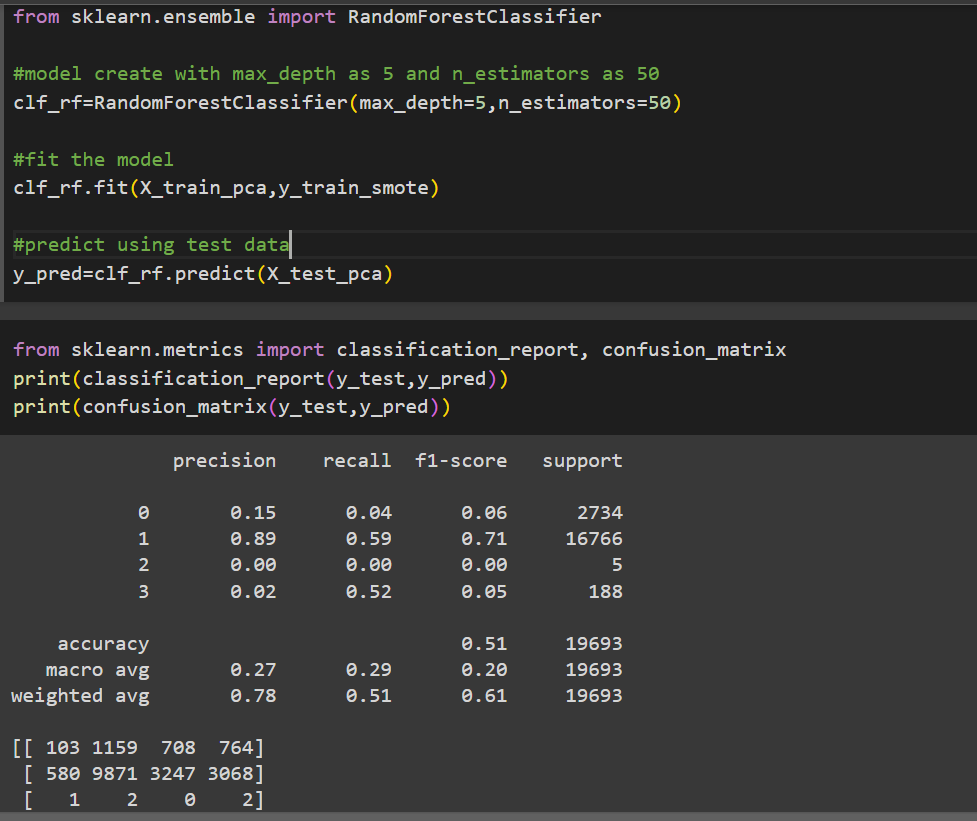
* 1. **Random Forest classifier for IQR outliners:**

Random Forests are one of the widely used supervised learning, it is an ensemble of tree predictors, where each tree relies on values from a randomly sampled vector, distributed independently and uniformly across all the trees. It is resistant to high noise and overfitting.

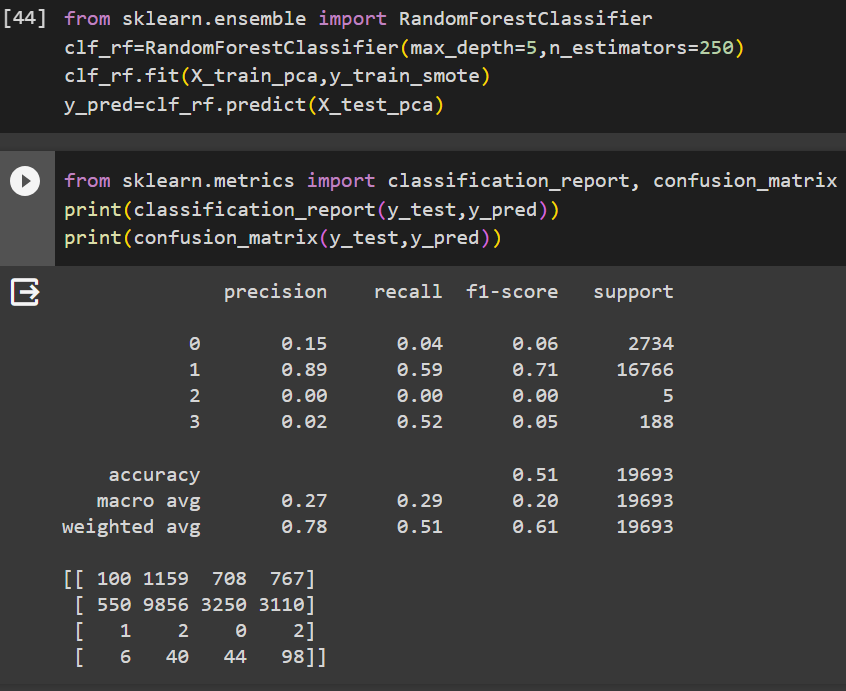
Designed the random forest classifier ensemble model using sklearn libraries (RandomForestClassifier)

Trained the model with training data and implemented the data on the test data using different values for parameters (Maximum\_Depth and Number of estimators). Model’s accuracy lies between 51 and 52 percentage for different values of Maximum depth and number of estimators.

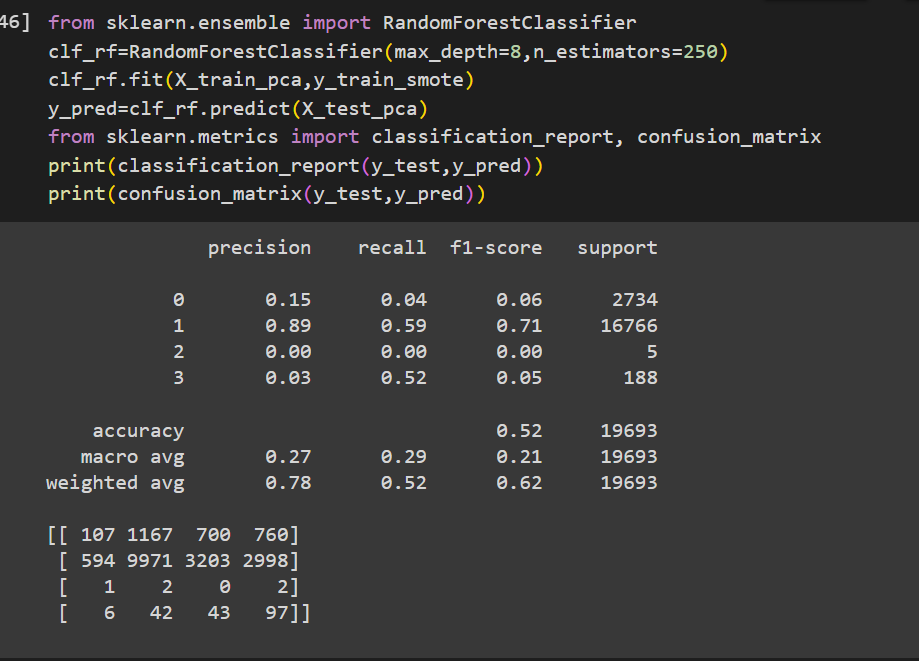
Used grid search cross validation to find out the best parameters (Maximum Depth, Number of estimators). The values of best parameters suggested by grid search are maximum depth is 10 and number of estimators is 100. The accuracy of the model when used these parameters is 51 percentage.



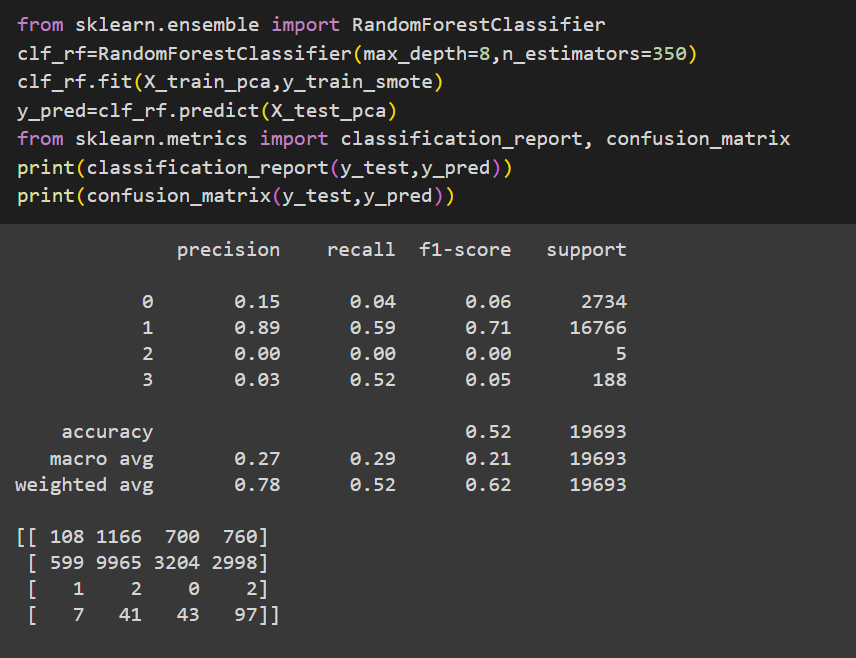
***fig:rf using max\_depth as 5 and n\_estimators as 50***



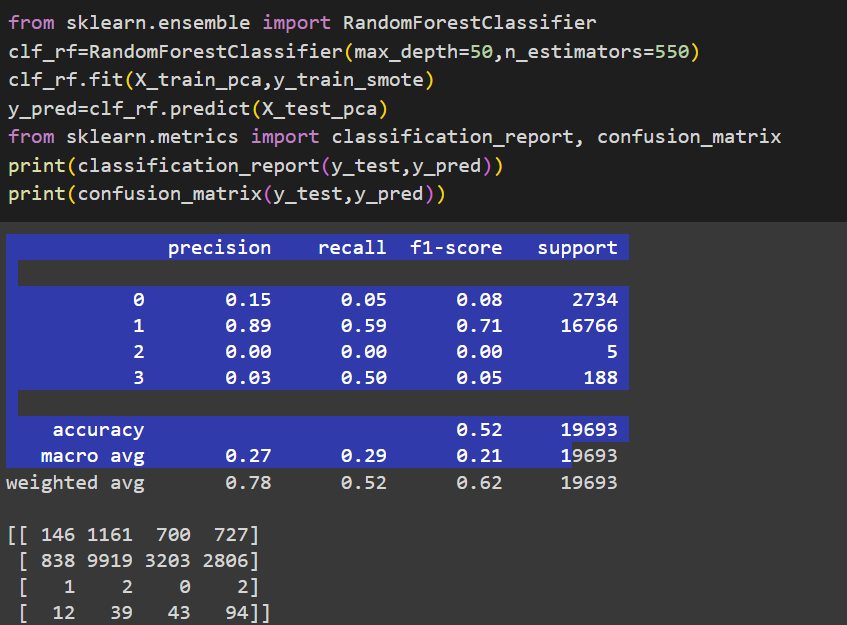
***fig:rf using max\_depth as 5 and n\_estimators as 250***



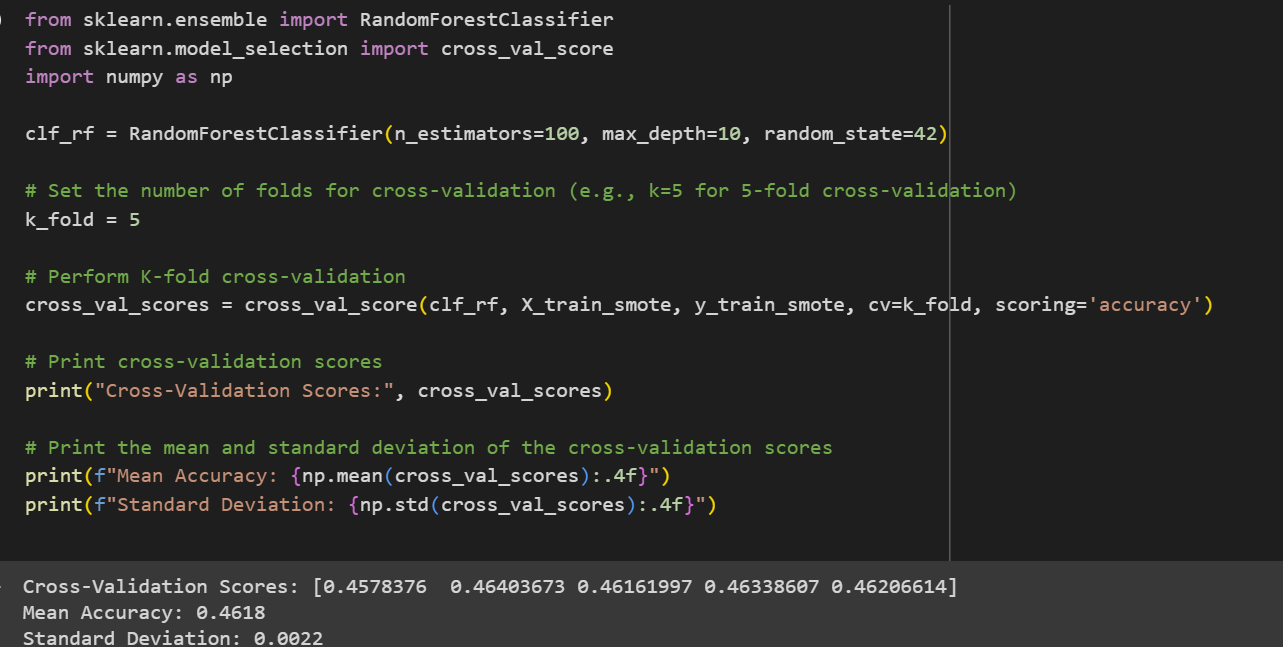
***fig:rf using max\_depth as 8 and n\_estimators as 250***



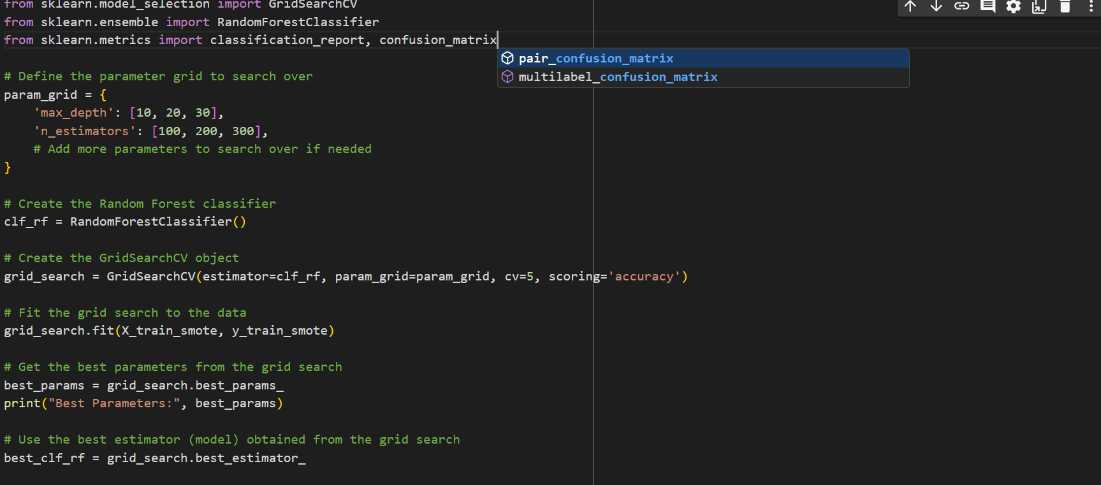
***fig:rf using max\_depth as8 and n\_estimators as 350***

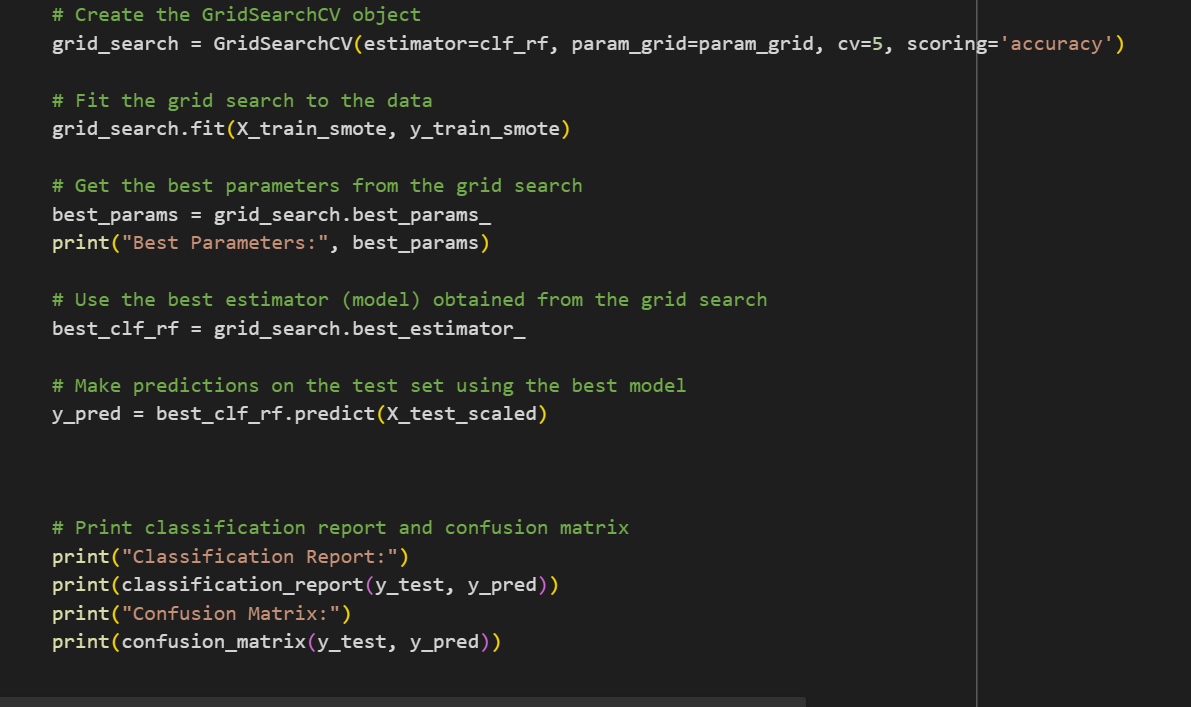


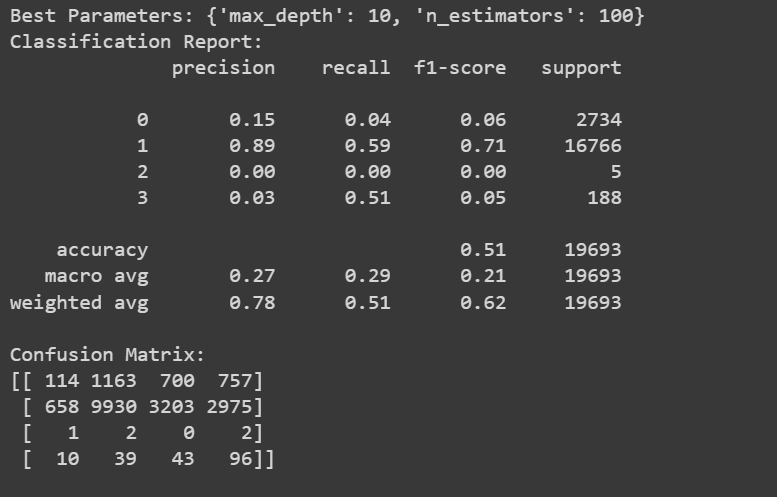
***fig:rf using max\_depth as 50 and n\_estimators as 550***



***fig:k-fold cross-validation***



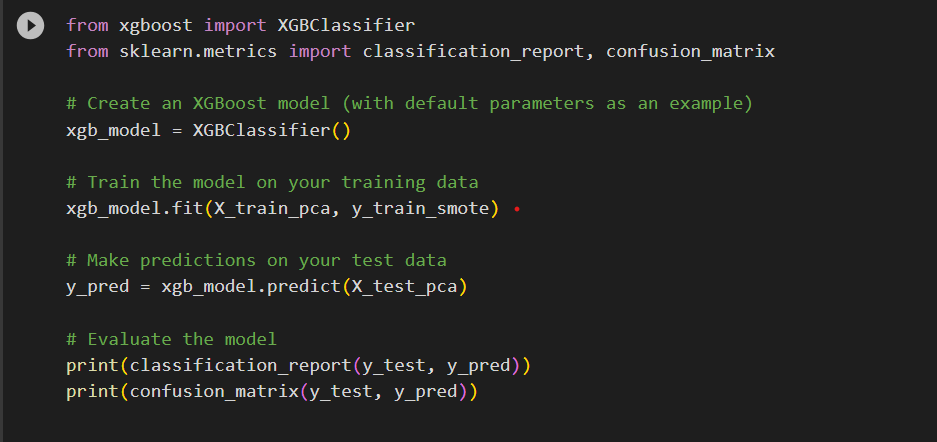


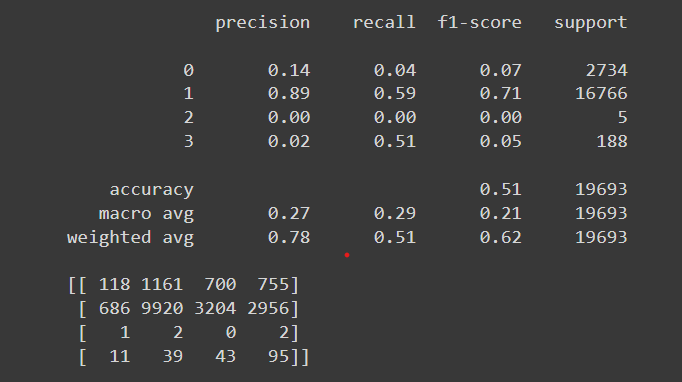


***Fig: grid search cross validation***

* 1. **XGboost Classifier for IQR outliners:**

XGBoost is an efficient and scalable machine learning algorithm that utilizes gradient boosting to combine the predictions of weak learners, often decision trees, resulting in high predictive accuracy and robust models.

****

****

***Fig: XGboost Classifier for IQR outliners***

* 1. **K-Nearest Neighbors (KNN) for IQR outliners:**

**Libraries Used:** Scikit-learn for KNeighborsClassifier, metrics, model selection, and numpy.

**KNN Logic:**

We used the KNN model in our implementation, with different values of n\_neighbors to determine the optimal one for our dataset. Before training the model, we pre-processed our data using SMOTE to balance the classes and PCA to reduce dimensionality.

To evaluate the performance of each model variant, we employed a loop over n\_neighbors' values from 1 to 25 and used 10-fold cross-validation. The accuracy of each model was the metric used to determine the optimal number of neighbors.

The KNN model's performance depends on the number of neighbors, which has an optimal value that we specified based on our results. The model achieved a % accuracy among the tested values of 65%, with a K value of 2.

The K-Nearest Neighbors (KNN) model has attained an accuracy rate of 65%, which is okay. However, it is essential to recognize that this level of accuracy does not fully meet our predictive goals. The complexity of the dataset, combined with the inherent difficulties of the classification task, are likely factors that contribute to this current performance metric.

A computer screen with text

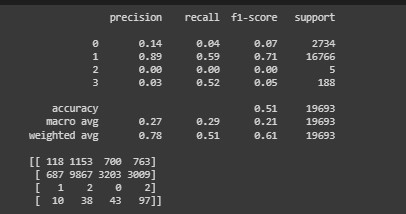
Description automatically generated

**A graph with blue lines

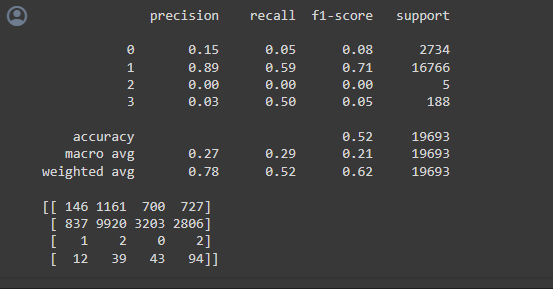
Description automatically generated**

***Fig: K-Nearest Neighbors (KNN) for IQR outliners***

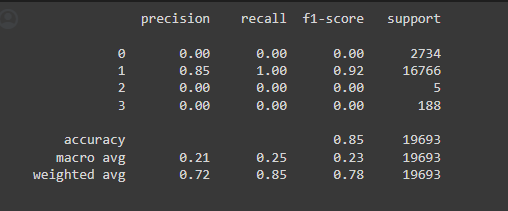
1. **Preliminary results for IQR outliners:**
   1. **Results for SVM For rbf:**



***fig: model of SVM for rbf C- 100, gamma - 0.1, decision\_function\_shape – ovr***

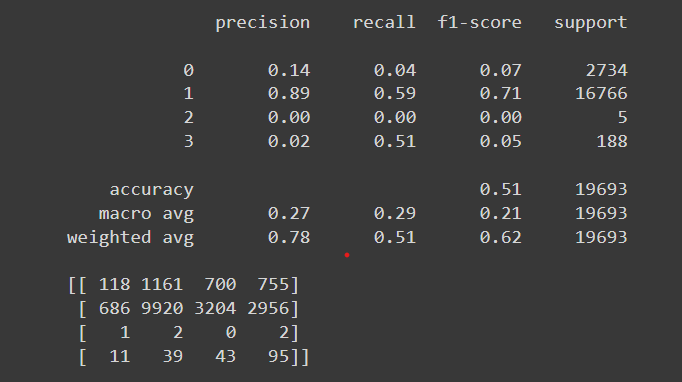
******

***fig: model of SVM for rbf C- 1000, gamma - 1,decision\_function\_shape – ovo***

******

***fig: model of SVM for linear C- 100, gamma – 0.1***

* 1. **Results for XGBoost:**

****

***fig: XGBoost Results***

* 1. **Results for Random Forest:**

**A screenshot of a computer screen

Description automatically generated**

***fig: random forest results***

* 1. **Results for K-Nearest Neighbors:**

A screenshot of a computer

Description automatically generated

***Fig: KNN results***

1. **Project Management - Implementation Status Report:**

**One-Hot Encoding**: Transformed categorical features using one-hot encoding for improved model compatibility.

**Exploratory Data Analysis (EDA):** Conducted insightful EDA, guiding key decisions in the modelling process.

Outliers Handling: Implemented effective outlier detection for enhanced data integrity and model performance

**Outliners:** Outliers are the data that cause inefficient accuracy to value and data is beyond the limit.

**Responsibility:** Outlier visualization before and after removing outliers, Implementing XGBoost Classifier.

**SMOTE:** In Data pre-processing step**,** Implemented SMOTE method to balance the classes in target variable as we have observed a huge class imbalance in target variable during data analysis.

**PCA:** Implemented Principal Component Analysis to reduce the dimensionality.

I used imputation methods to fill in missing or null values. I also considered removing records or features with significant missing data and removing duplicate entries to prevent biased models. Every data point in the dataset was made sure to be unique and accounted for. I analysed the relationships between features and the target variable.

Worked on the k-Nearest Neighbors (kNN) classifier to predict accident severity. To improve the performance of the kNN classifier, I selected only the relevant features.

**K-Nearest Neighbors:** Worked on the k-Nearest Neighbors (kNN) classifier to predict accident severity.

**Responsibility**: Cleaning null and duplicate data, K-Nearest Neighbors

**Contribution:** Contributed to cleaning data by removing null values and duplicates, implemented the kNN classifier, and documented 25% of the project.

**Increment - II**

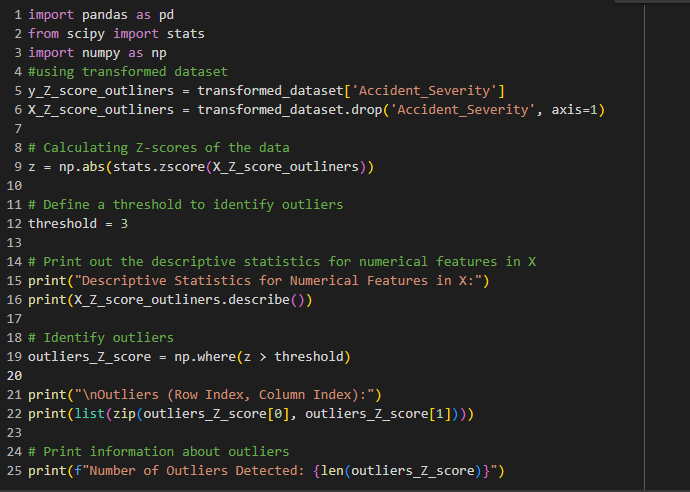
**Part – II**

1. **Outliners using Z-Score**

**Z-score: [2]**

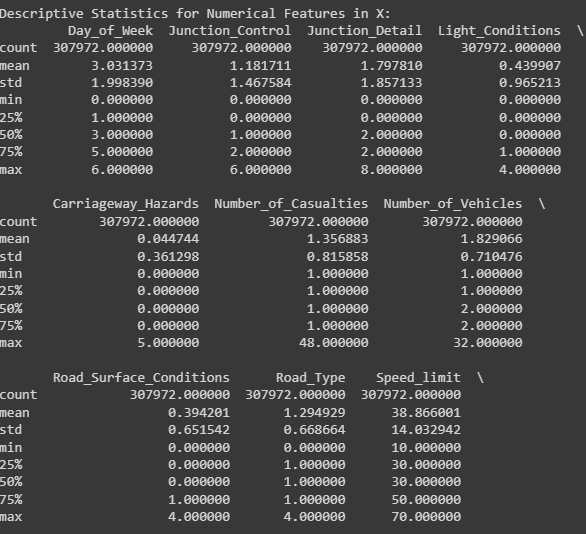
z-score is a statistical measurement that describes a value's relationship to the mean of a group of values, measured in terms of standard deviations from the mean. It's calculated by taking the difference between the value and the mean, and then dividing this by the standard deviation. This score indicates how many standard deviations an element is from the mean and allows for comparisons between different data sets.

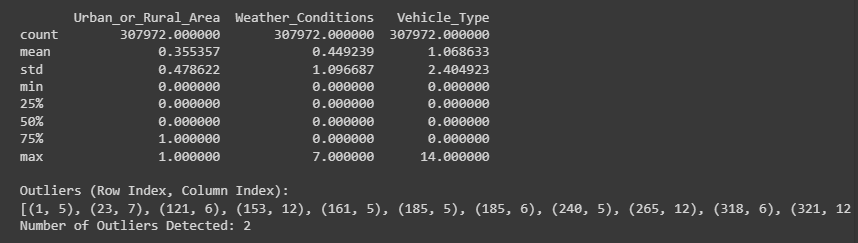
* 1. **Outliners detected using Z-Score method**



***fig: z-score outliners description and detection program***

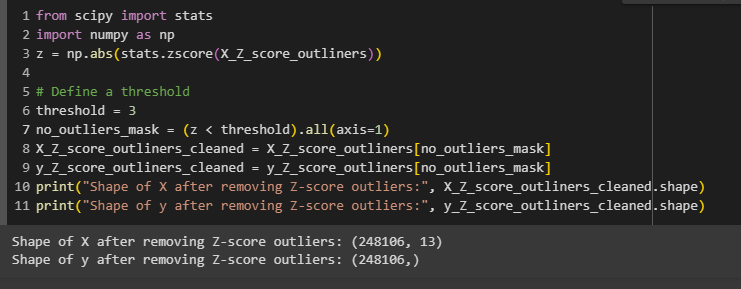
**Output:**





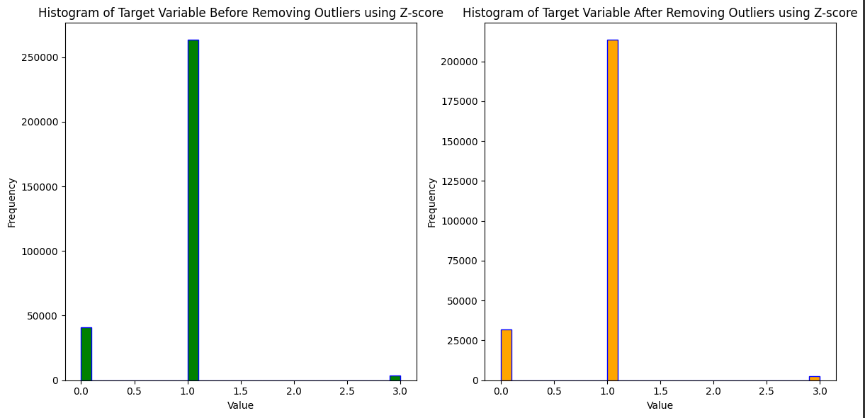
***fig: description of the outliners and detection***

Now outliners are removed here

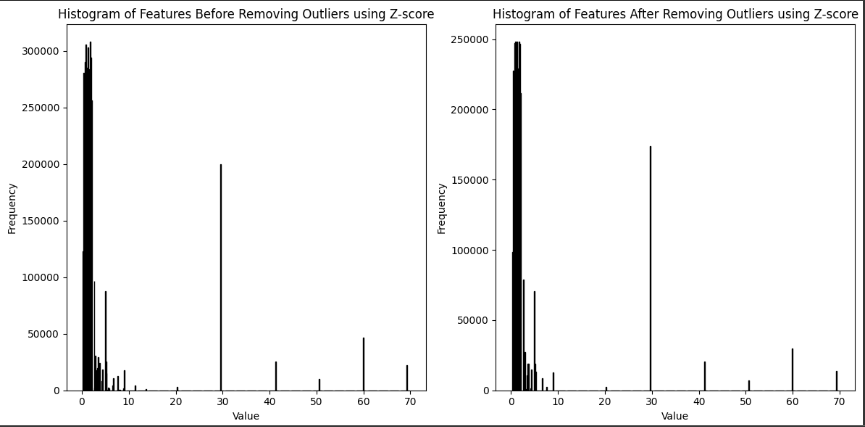


***Fig: Z-score outliners after deletion***

Plotted the target value before and after removing the outliners using the Z-score

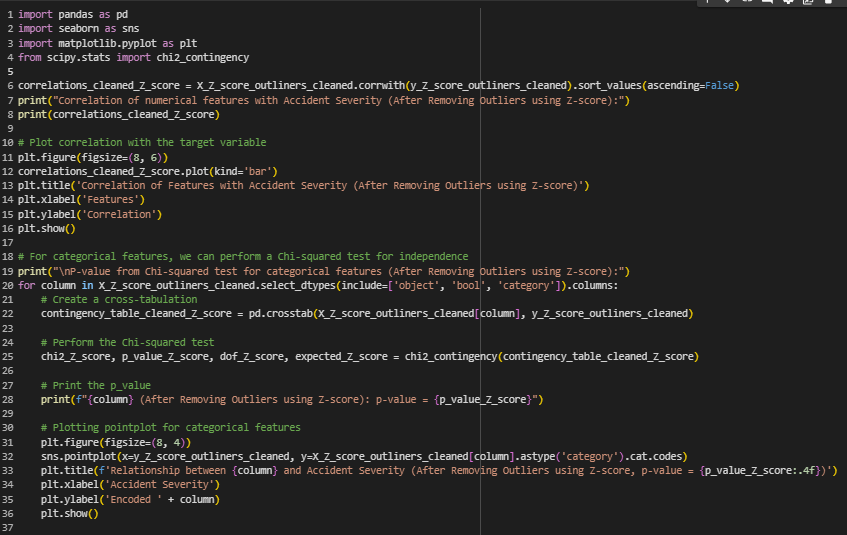


***fig: target value before and after z-score outliners***



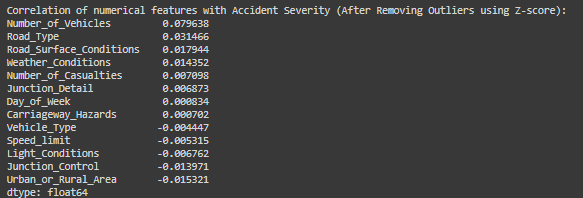
***fig: features before and after z-score outliners***

* 1. **Correlation with the target for Z-Score method**

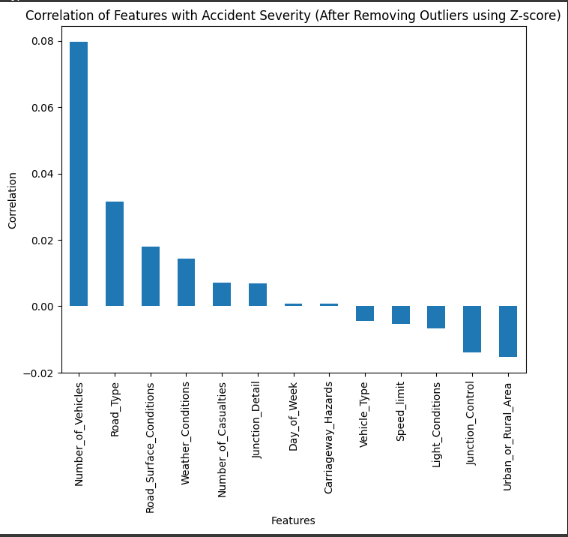


***fig: correlation with the target for z-score method***

**Output:**

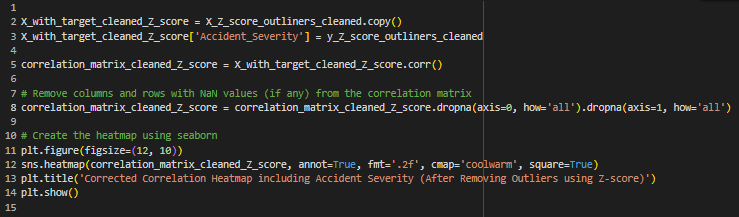


***fig: correlation values***



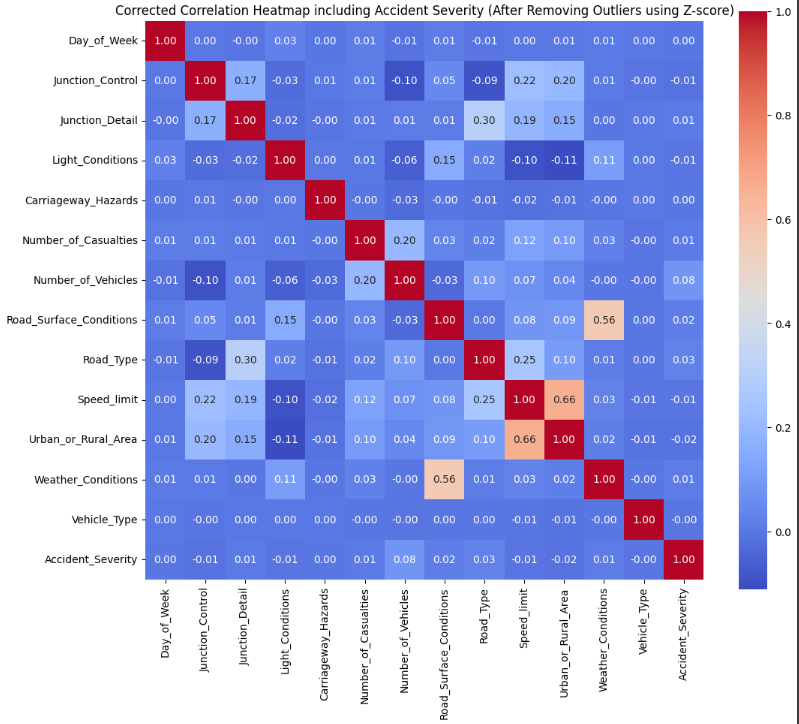
***fig: correlation plot***

* 1. **Correlation Heat Map for the Z-Score method**



***fig: correlation heat map program***

Output:



***fig: correlation heat map output plot***

* 1. **Plots of distribution of the numerical features for Z-Score method:**

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

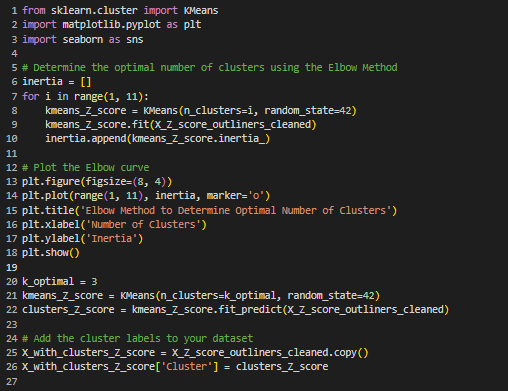
***fig: plots of distribution of the numerical features for z-score method***

* 1. **Plots of the distribution of the features with the target variable for Z-Score method**

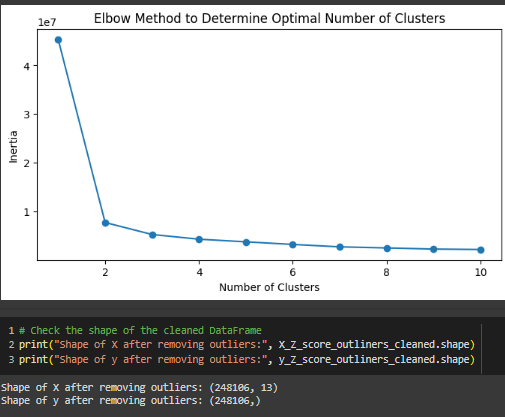
|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

***fig: plots of the distribution of the features with the target variable for z-score method***

* 1. **Clusters for Z-Score method:**



***fig: clusters program z-score***



***fig: clusters for z-score method output***

* 1. **SMOTE for Z-Score method:**



***fig: program for smote of z-score***

**Plots of SMOTE:**

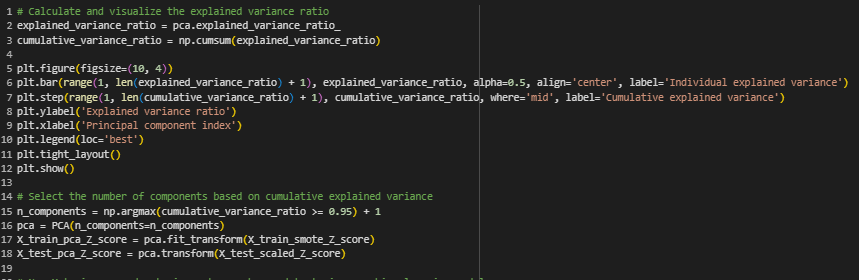


***fig: plots for smote***

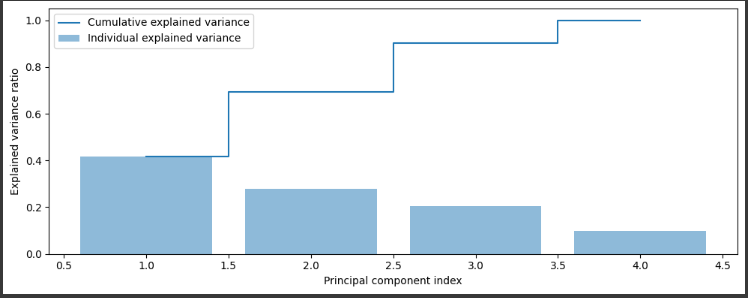
* 1. **PCA for Z-Score method**



***fig: pca for z-score program***

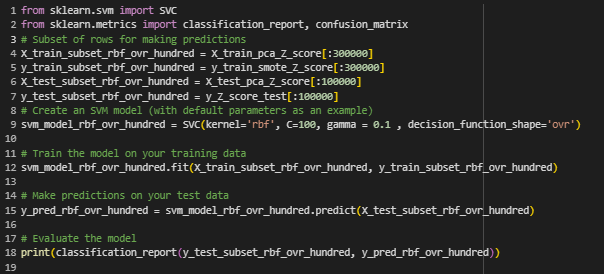


***fig: pca for z-score for variance***

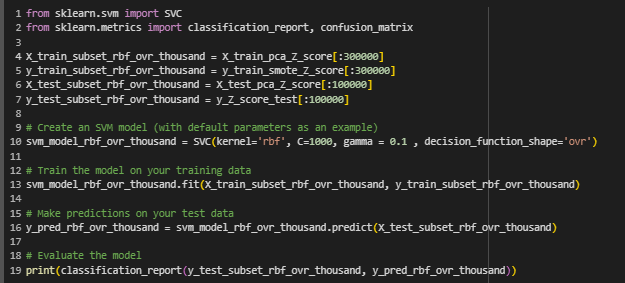


***fig: pca for z-score output varience***

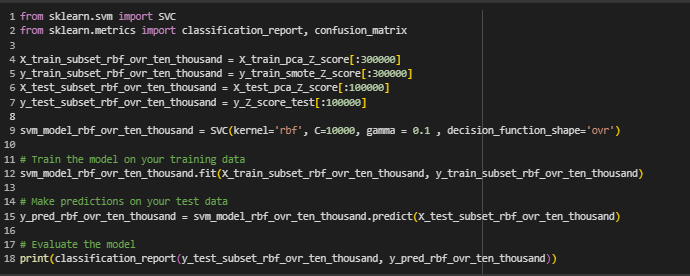
1. **Model Designed for Z-Score Outliners:**
   1. **SVM (Support Vector Machine)Classifier for Z-Score**



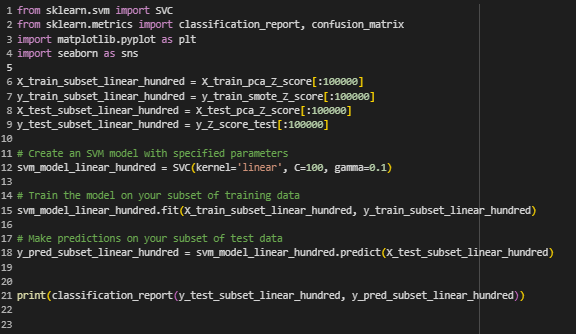
***fig: model of SVM for rbf C- 100, gamma - 0.1,decision\_function\_shape – ovr***



***fig: model of SVM for rbf C- 1000, gamma - 0.1,decision\_function\_shape – ovr***



***fig: model of SVM for rbf C- 10000, gamma - 0.1,decision\_function\_shape – ovr***



***fig: model of SVM for linear C- 100, gamma - 0.1***

* 1. **Random Forest Classifier for Z-Score**

**A screenshot of a computer program

Description automatically generated**

***Fig: model for RF with max\_depth as 88 and n\_estimators as 880***

**A screenshot of a computer program

Description automatically generated**

***Fig:model for rf using max\_depth as 100 and n\_estimators as 800***

***A computer screen shot of a program

Description automatically generated***

***Fig: model for RF with subsets of data***

* 1. **XGboost Classifier for Z-Score**

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generated**

* 1. **KNN(K- Nearest Neighbor) Classifier for Z-Score**

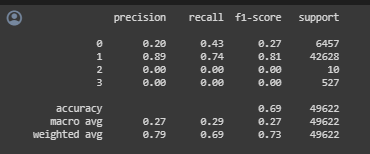
**A screenshot of a computer program

Description automatically generated**

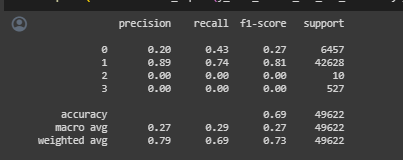
**A screenshot of a computer program

Description automatically generated**

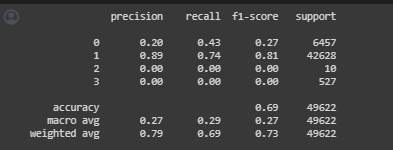
1. **Preliminary Results of models designed for Z-Score Outliners:**
   1. **SVM (Support Vector Machine)Classifier for Z-Score**



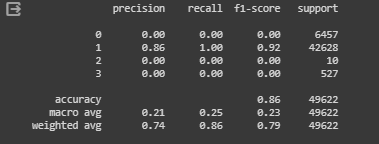
***fig: model of SVM result for rbf C- 100, gamma - 0.1,decision\_function\_shape – ovr***



***fig: model of SVM result for rbf C- 1000, gamma - 0.1,decision\_function\_shape – ovr***



***fig: model of SVM result for rbf C- 10000, gamma - 0.1,decision\_function\_shape – ovr***



***fig: model of SVM result for linear C- 100, gamma - 0.1***

* 1. **Random Forest Classifier for Z-Score**

**A screenshot of a computer screen

Description automatically generated**

***Fig: results for RF model with max\_depth 88 and n\_estimators 880***

***A screenshot of a computer screen

Description automatically generated***

***Fig:Results for RF model with max\_depth as 100 and n\_estimators as 800***

***A screenshot of a computer program

Description automatically generated***

***Fig:Results for RF Model with subsets of data***

* 1. **XGboost Classifier for Z-Score**

**A screenshot of a computer

Description automatically generated**

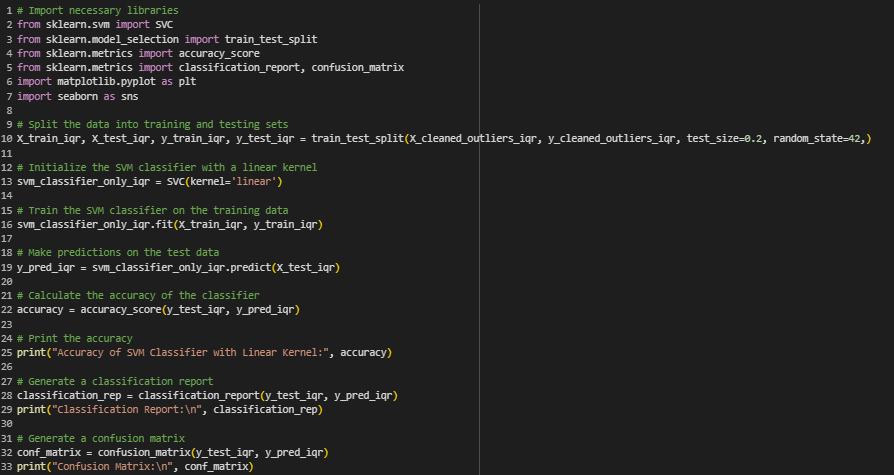
* 1. **KNN (K- Nearest Neighbor) Classifier for Z-Score**

**A screenshot of a computer screen

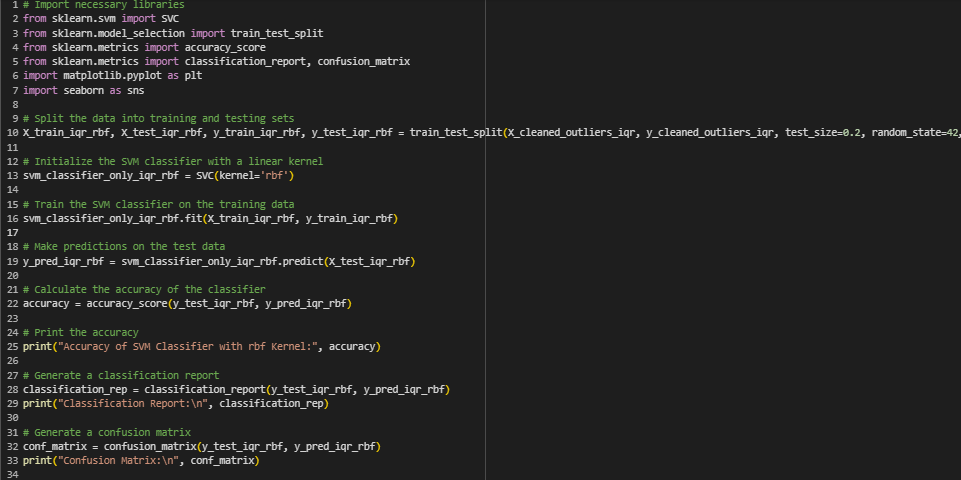
Description automatically generated**

**Part – III**

1. **Model Designed IQR without SMOTE and PCA:**
   1. **SVM (Support Vector Machine) Classifier IQR without SMOTE and PCA**



***fig: svm linear for iqr without smote and pca***



***fig: svm rbf for iqr without smote and pca***

* 1. **Random Forest Classifier IQR without SMOTE and PCA**

**A screen shot of a computer

Description automatically generated**

***Fig:Random Forest Classifier IQR without SMOTE and PCA***

* 1. **XGboost Classifier IQR without SMOTE and PCA**

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

* 1. **KNN(K- Nearest Neighbor) Classifier IQR without SMOTE and PCA**

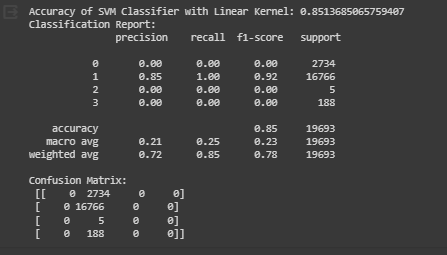
**A screenshot of a computer program

Description automatically generated**

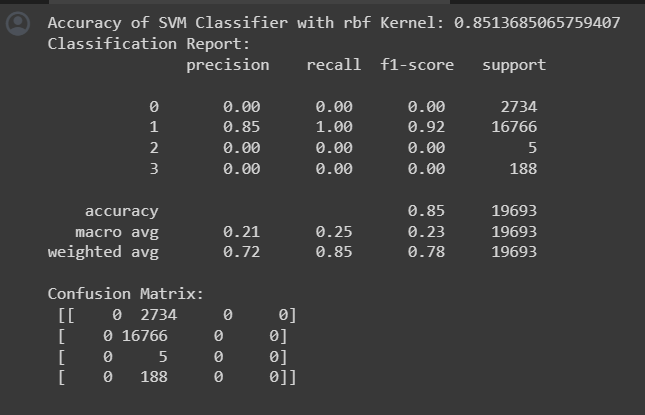
**A screenshot of a computer screen

Description automatically generated**

1. **Preliminary Results of models designed IQR without SMOTE and PCA:**
   1. **SVM (Support Vector Machine)Classifier IQR without SMOTE and PCA**



***fig: svm linear for iqr without smote and pca***



***fig: svm rbf for iqr without smote and pca***

* 1. **Random Forest Classifier IQR without SMOTE and PCA**

**A screenshot of a computer

Description automatically generated**

***Fig:Random Forest Classifier IQR without SMOTE and PCA***

* 1. **XGboost Classifier IQR without SMOTE and PCA**

**A screenshot of a computer

Description automatically generated**

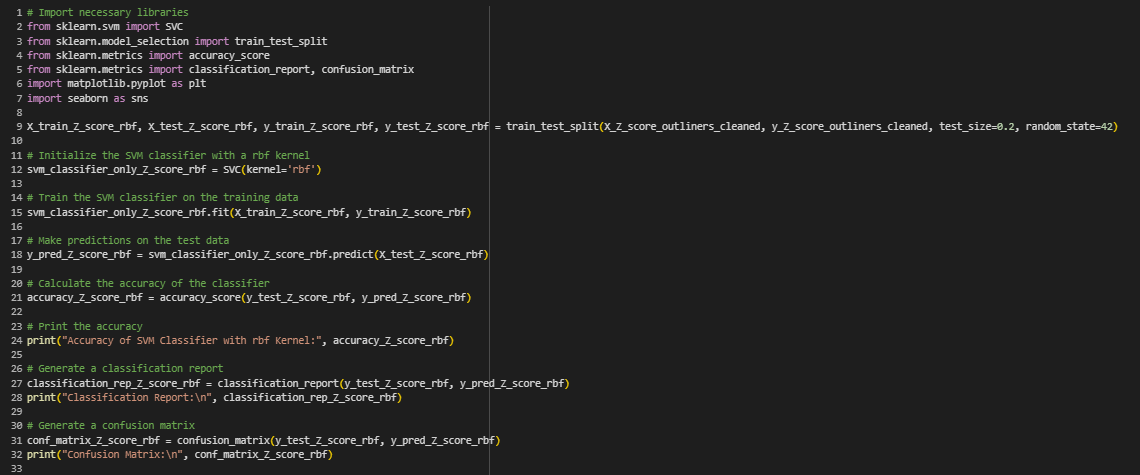
* 1. **KNN(K- Nearest Neighbor) Classifier IQR without SMOTE and PCA**

A screenshot of a computer screen

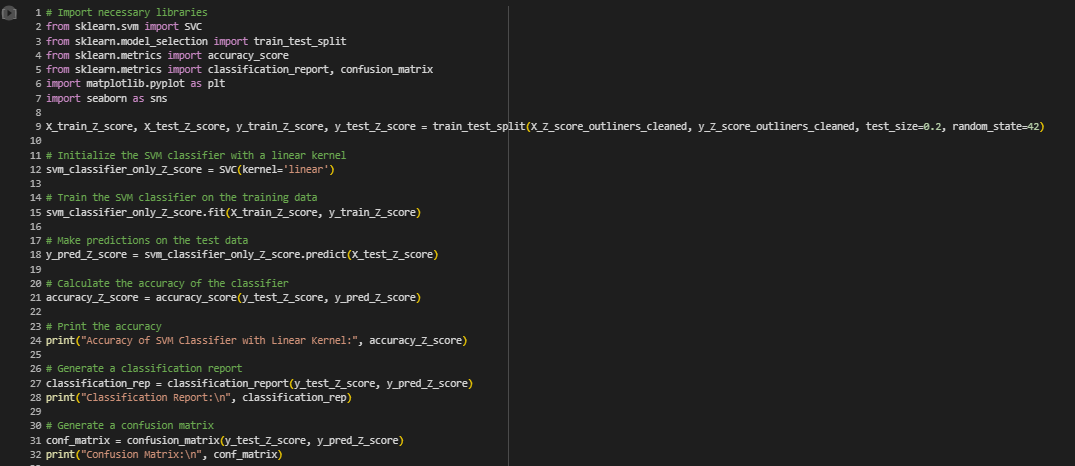
Description automatically generated

**Part – IV**

1. **Model Designed Z-Score without SMOTE and PCA:**
   1. **SVM (Support Vector Machine) Classifier Z-Score without SMOTE and PCA**

**

***fig: svm rbf z-score without smote and pca***



***fig: svm linear z-score without smote and pca***

* 1. **Random Forest Classifier Z-Score without SMOTE and PCA**

**A computer screen shot of a program

Description automatically generated**

***Fig: Random Forest Classifier with Z-score without SMOTE and PCA***

* 1. **XGboost Classifier Z-Score without SMOTE and PCA**

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

* 1. **KNN (K- Nearest Neighbor) Classifier Z-Score without SMOTE and PCA**

A computer screen shot of text

Description automatically generated

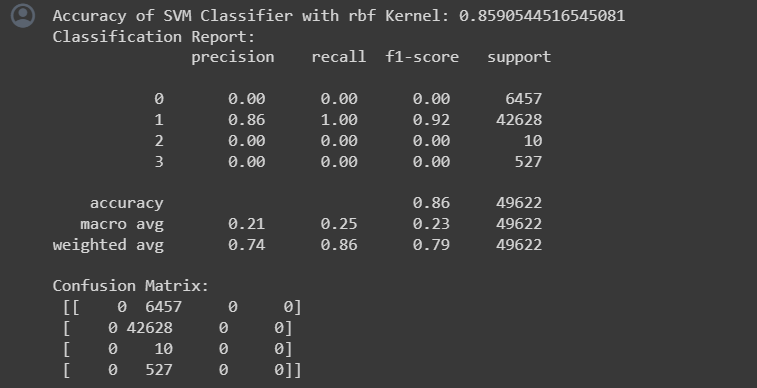
A screen shot of a computer program

Description automatically generated

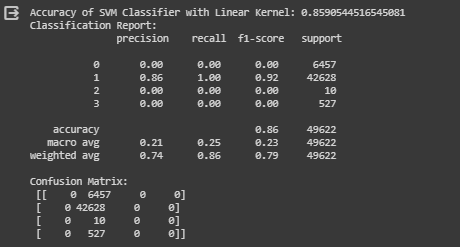
A screenshot of a computer screen

Description automatically generated

1. **Preliminary Results of Models Designed Z-Score Without SMOTE And PCA:**
   1. **SVM (Support Vector Machine) Classifier Z-Score without SMOTE and PCA**



***fig: svm rbf z-score without smote and pca output***



***fig: svm linear z-score without smote and pca output***

* 1. **Random Forest Classifier Z-Score without SMOTE and PCA**

**A screenshot of a computer screen

Description automatically generated**

***Fig: Random Forest Classifier Z-Score without SMOTE and PCA***

1. **References / Bibliography**

[1] Dataset <https://www.kaggle.com/datasets/jhalls/accident-data>

Outliners <https://www.kaggle.com/code/nareshbhat/outlier-the-silent-killer>

<https://www.sciencedirect.com/science/article/pii/S2590198223000611>

<https://ieeexplore.ieee.org/document/9693055>

<https://towardsdatascience.com/what-is-eda-yes-another-post-on-eda-d8b5c06269a9>

<https://rpubs.com/Magguire/913785>

<https://www.kaggle.com/code/faressayah/support-vector-machine-pca-tutorial-for-beginner>

<http://prody.csb.pitt.edu/manual/reference/dynamics/pca.html>

<https://www.kaggle.com/code/shubhisaxena12/eda-and-pca-analysis>

<https://journals.sagepub.com/doi/10.1177/1536867X20909688>

[2] Z-score

<https://www.investopedia.com/terms/z/zscore.asp#:~:text=Z%2Dscore%20is%20a%20statistical,identical%20to%20the%20mean%20score.>

[3] SMOTE

<https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/>

[4] PCA

<https://www.sartorius.com/en/knowledge/science-snippets/what-is-principal-component-analysis-pca-and-how-it-is-used-507186>

[5] IQR

<https://towardsdatascience.com/why-1-5-in-iqr-method-of-outlier-detection-5d07fdc82097>

[6] outliners

<https://www.freecodecamp.org/news/how-to-detect-outliers-in-machine-learning/>