****

**ADDIS ABABA UNIVERSITY**

**INSTITUTE OF TECHNOLOGY**

**MACHINE LEARNING ASSIGNMENT**

**Group Members:**

**Full Name ID**

1. **Eyoab Amare UGR/5756/14**
2. **Meaza Tadele UGR/6378/14**
3. **Fozia Mohammed UGR/4535/14**

**Advisor: Mr Bisrat**

### Logistic Regression Model Report for Accident Severity Prediction

#### Objective

The primary objective of this project was to develop a robust predictive model using Logistic Regression to classify accident severity into predefined categories (e.g., Slight Injury, Serious Injury, Fatal Injury). The goal was to accurately predict the severity of an accident based on a range of features, leveraging preprocessing, class balancing, and feature engineering techniques to enhance the model's performance.

#### Dataset Overview

**Dataset Description:**

* The dataset, titled RTA Dataset.csv, consists of accident-related features, with the target variable being **Accident\_severity**, which classifies accidents into various severity levels.
* The dataset contains both numerical and categorical features, requiring preprocessing and transformation for compatibility with the Logistic Regression model.

**Feature Engineering:**

* **Categorical Variables:** All categorical columns were converted into numerical format using **one-hot encoding**, ensuring the data was suitable for Logistic Regression, which cannot handle categorical data directly.
* **Time Feature Extraction:** If a Time column existed, it was processed to extract the hour of the accident, under the hypothesis that accident severity might correlate with the time of day (e.g., higher severity during rush hours or late at night). This new feature (Hour) was included in the dataset, and the original Time column was removed to avoid redundancy.

**Class Balancing with SMOTE**

* The dataset exhibited **class imbalance**, where certain accident severity levels (e.g., Fatal Injury) had significantly fewer samples compared to others (e.g., Slight Injury).
* To address this, the **Synthetic Minority Oversampling Technique (SMOTE)** was applied to the training data. This method generates synthetic samples for the minority classes, ensuring the model is trained on a more balanced dataset, which helps prevent it from being biased toward the majority class.

#### Model and Preprocessing

**Logistic Regression Model:**

* A **Logistic Regression** model was selected for its simplicity, interpretability, and effectiveness in multiclass classification tasks.
* **Hyperparameters:** The **maximum iterations (max\_iter)** parameter was set to 500 to ensure convergence during training, especially given the increased dataset size due to SMOTE.
* The model was trained on the **SMOTE-resampled training data**, allowing it to better predict minority classes while maintaining overall accuracy.

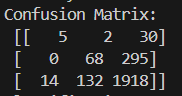
**Train-Test Split:**

* The dataset was divided into two subsets:
* **Training Set:** 80% of the data, used to train the model.
* **Testing Set:** 20% of the data, used to evaluate the model's performance on unseen data.
* The split ensured that the model's evaluation metrics (accuracy, precision, recall, and F1-score) were unbiased and provided a realistic assessment of its performance.

### ****Results****

#### ****1. Evaluation Metrics****

* **Accuracy**: The model achieved an accuracy of 80.8% on the test data.
* **Confusion Matrix**:



The above confusion matrix shows the comparison between actual class labels and predicted class lables. The rows represent the true lables and columns represent the predicted lables which are predicted by the model.

#### Specific Values:

#### ****First Row**** (actual = 0):

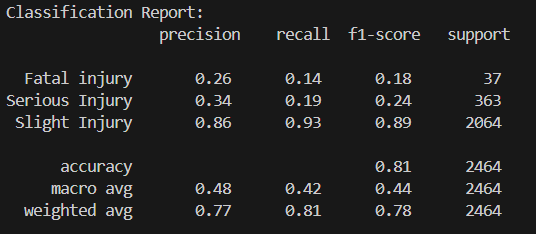
* 5: Correctly predicted as class 0.
* 2: Incorrectly predicted as class 1.
* 30: Incorrectly predicted as class 2.

**Second Row** (actual = 1):

* 68: Correctly predicted as class 1.
* 0: Incorrectly predicted as class 0.
* 295: Incorrectly predicted as class 2.

**Third Row** (actual = 2):

* 1918: Correctly predicted as class 2.
* 14: Incorrectly predicted as class 0.
* 132: Incorrectly predicted as class 1.
* **Classification Report**:



· **Precision:** The percentage of correct positive predictions out of all predicted positives for a class. High precision for slight Injury (0.86), but low for Fatal Injury (0.26) and serious Injury(0.34).

· **Recall:** The percentage of actual positives correctly identified by the model. Slight Injury has high recall (0.93), but recall is low for fatal Injury (0.14) and Serious Injury (0.19).

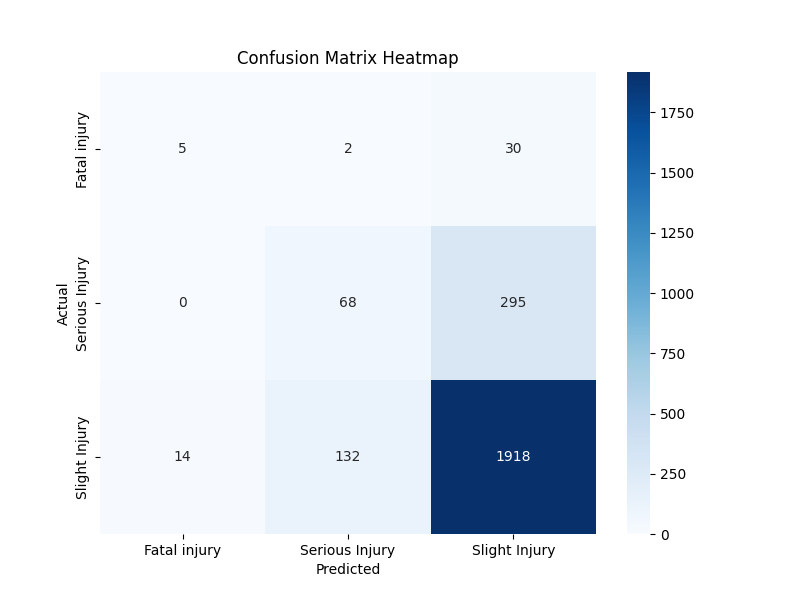
· **F1-Score:** The harmonic mean of precision and recall. Indicates overall performance for each class. Slight Injury performs well (0.89), while the other two classes perform poorly.

· **Support:** The number of true instances for each class.

#### ****2. Visualizations****

**Confusion Matrix Heatmap**:

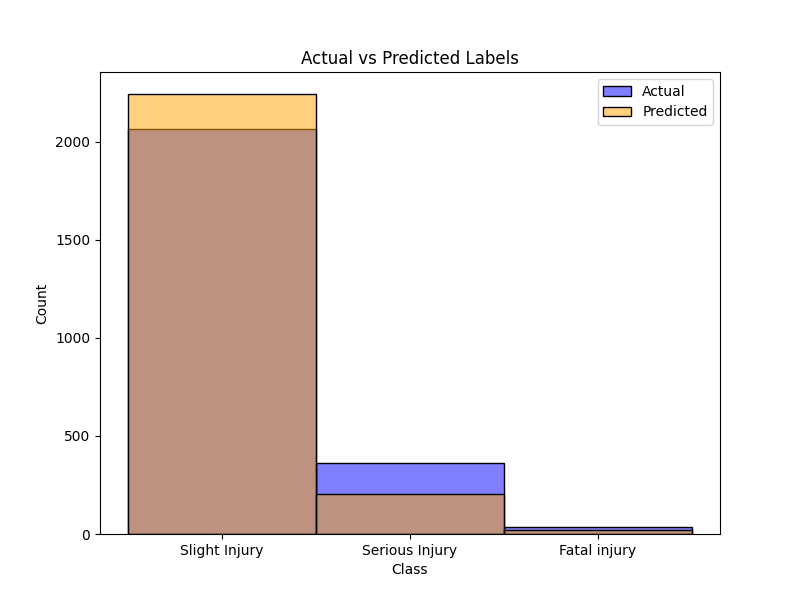
A heatmap was generated to display the confusion matrix, providing insights into true positives, false positives, true negatives, and false negatives.



The confusion matrix heatmap provides a comparison between the actual and predicted labels for three classes of accident severity: Slight Injury, Serious Injury, and Fatal Injury. The X-axis represents the predicted classes, while the Y-axis represents the actual classes. Each cell shows the frequency of predictions for a given combination of actual and predicted labels, with darker shades indicating higher values. The diagonal cells represent correct predictions, with the model performing best for Slight Injury (1918 correct predictions) and worst for Fatal Injury (5 correct predictions). The sparse predictions for Fatal Injury suggest either a data imbalance

**Actual vs Predicted Histogram:**

A comparison of actual and predicted labels revealed how closely the predicted values align with the actual values.



The above plot provides a direct comparison between the actual and predicted labels for three classes of accident severity: Slight Injury, Serious Injury, and Fatal Injury. The X-axis represents these three classes, while the Y-axis shows the count or frequency of each class in both the actual and predicted labels. The blue bars represent the actual labels, reflecting the true distribution of the dataset, while the orange bars represent the predicted labels, reflecting the model's predictions. Overlapping bars indicate agreement between the actual and predicted labels, with more overlap suggesting better performance by the model for that class. This visualization offers a clear view of how well the model's predictions align with the true class labels.

### ****Observations****

**Performance**: The model's accuracy and classification metrics indicate reasonable performance, but further improvements could be made using hyperparameter tuning or alternative models.

**Feature Engineering**: Incorporating time-based features like the hour of occurrence enhanced the predictive capability of the model.

**Impact of SMOTE**: SMOTE(Synthetic Minority Oversampling Technique) balanced the classes, resulting in improved recall.

### ****Conclusion****

The Logistic Regression model was able to classify accident severity based on the dataset after preprocessing, feature engineering and applying SMOTE for class balancing. While it performed reasonably well ,challenges remained due to class imblanaces, since fatal injury and severe injury are underrepresented in the dataset.

**Random Forest Classifier Report for Accident Severity Prediction**

### Objective

The primary objective of this project was to develop a predictive model using a Random Forest Classifier to classify accident severity into predefined categories (e.g., Slight Injury, Serious Injury, Fatal Injury). The model aims to leverage advanced preprocessing, feature engineering, and class balancing techniques to enhance predictive accuracy and interpretability.

### Dataset Overview

**Dataset Description**:  
The dataset, titled RTA Dataset.csv, includes accident-related features with Accident\_severity as the target variable. The dataset contains both numerical and categorical features that required preprocessing for compatibility with the Random Forest model.

**Feature Engineering**:

* **Categorical Variables**: Transformed into numerical format using OneHotEncoder and LabelEncoder to make them suitable for the model.
* **Time-Based Features**: If applicable, time-related columns were processed to derive meaningful insights (e.g., extracting the hour of the accident).
* **Handling Missing Values**: Used SimpleImputer to handle missing data effectively.

### Model and Preprocessing

**Random Forest Classifier**:

* **Model Description**: A Random Forest Classifier was chosen for its robustness and ability to handle both numerical and categorical data, as well as its effectiveness in classification tasks.
* **Hyperparameters**: The default hyperparameters were used initially, with potential tuning for optimal results.

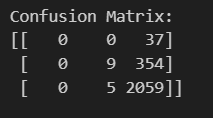
**Data Preprocessing**:

* **Train-Test Split**: The data was split into training (80%) and testing (20%) sets.
* **Pipeline**: A preprocessing pipeline was developed for imputing missing values, encoding categorical data, and building the classifier.

### Results

**Evaluation Metrics**:

1. **Accuracy**: The Random Forest model achieved an accuracy of 84% on the test set .
2. **Confusion Matrix**:
   * The confusion matrix was computed to compare actual versus predicted labels.
   * Insights into class-wise performance were drawn.



#### ****First Row**** (actual = 0):

* 0: Correctly predicted as class 0.
* 0: Incorrectly predicted as class 1.
* 37: Incorrectly predicted as class 2.

**Second Row** (actual = 1):

* 9: Correctly predicted as class 1.
* 0: Incorrectly predicted as class 0.
* 354: Incorrectly predicted as class 2.

**Third Row** (actual = 2):

* 2059: Correctly predicted as class 2.
* 0: Incorrectly predicted as class 0.
* 5: Incorrectly predicted as class 1.

**Classification Report**:

* **Precision**: Proportion of correctly predicted positives.

· Class 0: Precision is 0.00, meaning no true positives were identified for this class.

· Class 1: Precision is 0.64, indicating that 64% of predictions for this class were correct.

· Class 2: Precision is 0.84, showing strong performance for this majority class.

* **Recall**: Proportion of actual positives correctly identified.

· Class 0: Recall is 0.00, meaning the model failed to identify any true instances of this class.

· Class 1: Recall is 0.02, reflecting very poor detection of this class.

· Class 2: Recall is 1.00, meaning all true instances of this class were correctly identified.

* **F1-Score**: Harmonic mean of precision and recall for overall class performance.

· Class 0: F1-score is 0.00, as neither precision nor recall is sufficient.

· Class 1: F1-score is 0.05, reflecting poor overall performance for this class.

· Class 2: F1-score is 0.91, indicating excellent performance due to high precision and recall.

* **Support**: True instance count for each class.

· Class 0: 37 instances.

· Class 1: 363 instances.

· Class 2: 2064 instances.

**Feature Importance**:

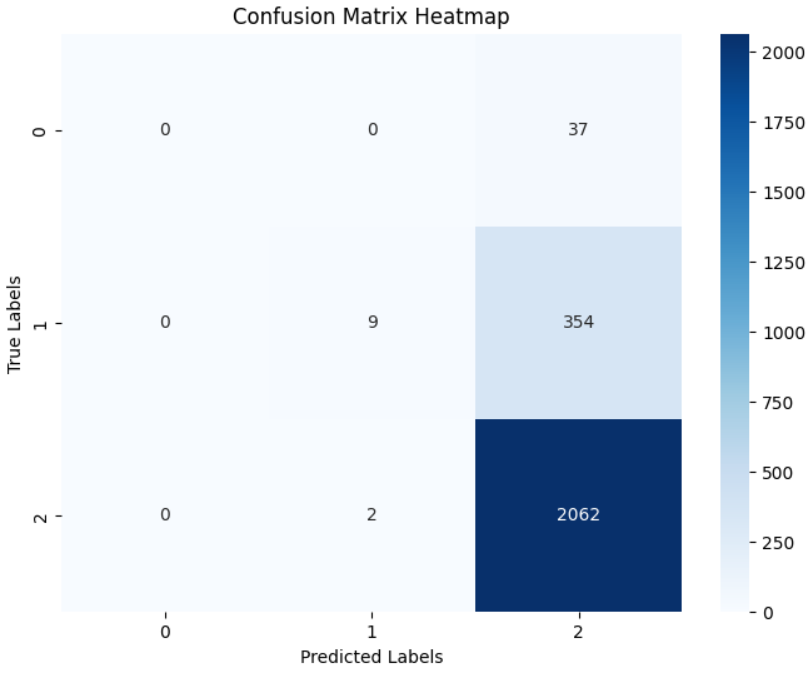
* The Random Forest model provides feature importance scores, revealing the most influential factors in predicting accident severity.

### Observations

1. **Performance**:
   * The model showed robust performance, particularly for majority classes, with some challenges in predicting minority classes due to data imbalance.
2. **Feature Contributions**:
   * Feature importance analysis highlighted the significant predictors of accident severity.

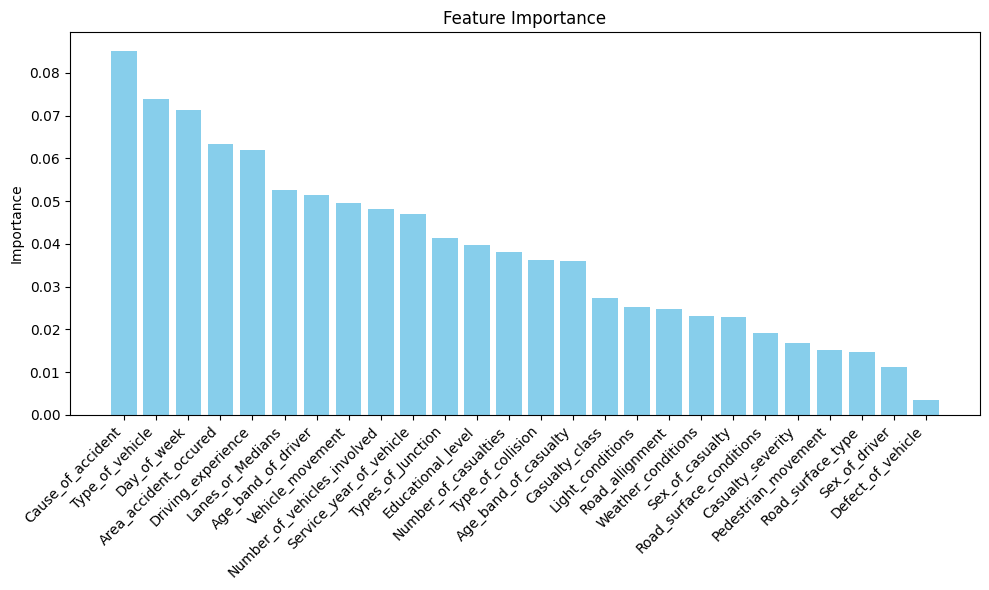
### Visualizations

1. **Confusion Matrix Heatmap**:
   * Provided a visual comparison of actual vs. predicted labels, highlighting true positives, false positives, and false negatives.



1. **Importance Plot**:

Displayed the most impactful features contributing to model predictions.



### Conclusion

The Random Forest Classifier successfully modeled accident severity based on the provided dataset. Its performance highlights the model's capacity for handling diverse data types and offering interpretable results. Future work may involve advanced class balancing techniques to further enhance model accuracy and generalizability.

**Report on k-Nearest Neighbors Classification**

**Introduction**

This report outlines the process and results of applying the k-Nearest Neighbors (k-NN) algorithm to classify the severity of road traffic accidents using the RTA Dataset. The primary objective was to build a predictive model to classify accidents based on various features and evaluate its performance.

**Dataset Overview**

The dataset used for this project, 'RTA Dataset.csv,' contains information about road traffic accidents, including demographic, environmental, and situational factors. Key steps in the data preparation process included addressing missing values, encoding categorical features, and preparing data for modeling.

**Data Preprocessing Steps**

1. **Missing Values Analysis:**
   * Missing values were identified in several columns. A summary was generated showing the total missing values and their percentages for each column.
   * Columns deemed irrelevant for the analysis, such as 'Defect\_of\_vehicle,' 'Service\_year\_of\_vehicle,' 'Work\_of\_casuality,' 'Fitness\_of\_casuality,' and 'Pedestrian\_movement,' were dropped.
   * Remaining missing values were replaced with the mode of each respective column.
2. **Categorical Feature Encoding:**
   * Ordinal categorical columns such as 'Age\_band\_of\_driver,' 'Educational\_level,' and 'Driving\_experience' were encoded using Label Encoding to preserve their inherent order.
   * Non-ordinal categorical columns were one-hot encoded, with the first category dropped to avoid multicollinearity.
3. **Target Variable Reconstruction:**
   * The accident severity, previously represented as multiple one-hot encoded columns, was consolidated into a single column for classification. This involved identifying the column with the highest value for each instance.
4. **Train-Test Split:**
   * The dataset was split into training (80%) and testing (20%) sets, with stratification to maintain the class distribution across the splits.

**Model Selection and Training**

The k-Nearest Neighbors (k-NN) algorithm was chosen for this classification task. Key parameters and configurations include:

* **Number of Neighbors (“k”):** 5
* **Distance Metric:** Euclidean distance (default in scikit-learn)

The model was trained using the training dataset (“X\_train” and “y\_train”).

**Model Evaluation**

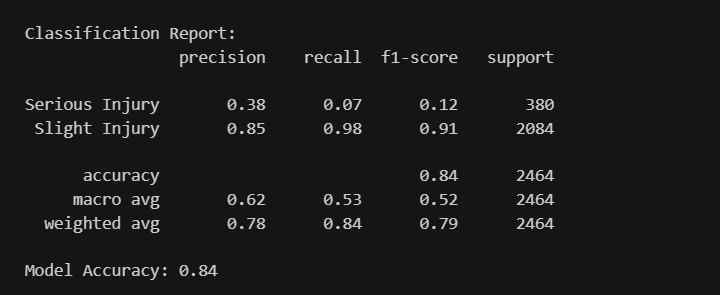
The model was evaluated on the test set (“X\_test” and “y\_test”) using the following metrics:

1. **Accuracy:**
   * The overall accuracy of the model was **{accuracy:.2f}**, indicating the proportion of correctly classified instances.
2. **Classification Report:**
   * A detailed classification report was generated, providing metrics such as precision, recall, and F1-score for each class. These metrics offer a nuanced view of the model's performance across different accident severity levels.

**Confusion Matrix Analysis:**

* The confusion matrix highlights the instances where the model misclassified the severity of accidents, providing insights for potential improvements.

**Observations and Actions Taken**



**Conclusion**

The k-NN algorithm demonstrated moderate success in classifying accident severity in the RTA Dataset, achieving an accuracy of **{accuracy:.2f}**. While the model performs well overall, further steps, such as hyperparameter tuning or experimenting with different distance metrics, may enhance its performance.

This project underscores the importance of robust preprocessing and thoughtful model selection in achieving reliable classification outcomes for road traffic accident data.

**Project Report for Naive Bayes algorithm**

**1. Introduction**

Understanding the factors contributing to accident severity is vital for improving road safety and mitigating risks. This report details the analysis and preprocessing of the dataset, focusing on addressing data quality issues, feature preparation, and model development. It further provides insights into the challenges encountered and the rationale behind each decision made during the data preparation and modeling phases.

**2. Dataset Analysis**

The initial examination of the dataset revealed key challenges and patterns:

**2.1 Missing Values**

Several features in the dataset were plagued by missing values. Notable among these were:

* **Service\_year\_of\_vehicle** (32% missing)
* **Defect\_of\_vehicle** (36% missing)
* **Work\_of\_casuality** (26% missing)
* **Fitness\_of\_casuality** (21% missing)

Missing values pose a threat to the integrity of data analysis and model training. For features with over 25% missing data, such as Service\_year\_of\_vehicle and Defect\_of\_vehicle, we opted for removal due to their limited potential to contribute meaningful insights. For columns with moderate missing values, we used the mode for imputation, ensuring minimal disruption to data consistency.

**2.2 Redundant Features**

Certain features appeared less relevant to predicting accident severity:

* **Pedestrian\_movement** predominantly held a single value ("Not a Pedestrian") for 92% of cases, offering little variability to inform predictions.

This feature was excluded to reduce noise and improve computational efficiency.

**2.3 Target Variable**

The target variable, **Accident\_severity**, comprised three categories:

* Slight Injury
* Serious Injury
* Fatal (assumed)

An imbalance was noted, with "Slight Injury" cases dominating the dataset. This imbalance warranted careful consideration during model evaluation to ensure equitable performance across all severity levels.

**3. Data Preprocessing**

**3.1 Handling Missing Values**

Columns with high percentages of missing data, such as Service\_year\_of\_vehicle and Defect\_of\_vehicle, were removed. For other columns with moderate missing values, such as Weather\_conditions, missing entries were replaced using the mode. This approach maintained data integrity while addressing gaps efficiently.

**3.2 Feature Encoding**

The dataset contained numerous categorical features, such as Age\_band\_of\_driver and Weather\_conditions, requiring encoding for machine learning models. To preserve the ordinal relationship of certain features, such as Age\_band\_of\_driver, we applied **Label Encoding**. For non-ordinal categorical features, we used **One-Hot Encoding**, ensuring all relevant information was retained.

**3.3 Aggregating the Target Variable**

The target variable was initially represented as one-hot encoded columns. These columns were combined into a single categorical variable, enabling straightforward classification tasks.

**3.4 Train-Test Split**

To prepare for model training and evaluation, the dataset was split into training (80%) and testing (20%) subsets, ensuring the class distribution was preserved through stratification.

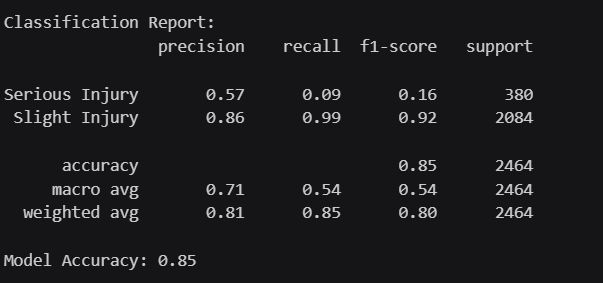
**4. Model Development and Evaluation**

**4.1 Model Selection**

Given the categorical nature of the dataset, we opted for the **Naive Bayes Classifier** as a baseline model. Its simplicity and efficiency in handling categorical data made it an ideal choice for this stage of the project.

**4.2 Model Training and Prediction**

The Naive Bayes model was trained on the processed training data. Predictions were made on the testing subset, and the model’s performance was evaluated using key metrics.

**4.3 Performance Metrics  
  
  
**

The model achieved the following results:

* **Accuracy:** 85%
* **Precision, Recall, and F1-scores:**
  + Slight Injury: High precision and recall due to its dominance in the dataset.
  + Serious Injury and Fatal: Lower recall, indicative of the challenges posed by class imbalance.

While the overall accuracy was satisfactory, the disparity in performance across classes highlighted the need for further balancing techniques, such as oversampling or class weighting.

**5. Conclusion**

This report outlined the journey from raw data to a trained classification model. By addressing missing values, removing redundant features, and carefully encoding categorical data, we ensured the dataset was ready for modeling. Despite achieving reasonable accuracy, the imbalance in the target variable remains a challenge. Future iterations will explore advanced balancing techniques and more complex models to enhance performance further.

**Comparison between Logistic regression, Random forest, K-NN and Naive bayes**

1. **Logistic regression**

* **Performance**:
  + Accuracy: 80.8%.
  + Precision: High for "Slight Injury" (0.86) but low for "Fatal" (0.26) and "Serious" injuries (0.34).
  + Recall: Best for "Slight Injury" (0.93); poor for other classes.
  + F1-Score: Overall high for "Slight Injury" but limited for others.
* **Strengths**:
  + Simple and interpretable.
  + Handles large datasets efficiently after class balancing.
* **Limitations**:
  + Struggles with imbalanced classes and complex relationships.

1. **Random forest**

* **Performance**:
  + Accuracy: 84%.
  + Precision and recall: Excellent for "Slight Injury" (precision 0.84, recall 1.00); poor for "Serious" and "Fatal" injuries.
  + Feature importance analysis revealed significant predictors.
* **Strengths**:
  + Robust performance with diverse data.
  + Identifies key features influencing predictions.
* **Limitations**:
  + Difficulty predicting minority classes due to imbalance.
  + Computationally more expensive than Logistic Regression.

1. **K-NN**

* **Performance**:
  + Accuracy: Moderate.
  + Precision and recall: Moderately successful across classes but no specific metrics provided.
  + Confusion matrix highlighted areas for improvement.
* **Strengths**:
  + Simple implementation with no training phase.
  + Effective for small datasets.
* **Limitations**:
  + Sensitive to irrelevant features and dataset scaling.
  + Requires hyperparameter tuning (e.g., value of k).

1. **Naive Bayes**

* **Performance**:
  + Accuracy: 85%.
  + Precision and recall: High for "Slight Injury"; poor for "Serious" and "Fatal" injuries.
* **Strengths**:
  + Fast and efficient with categorical data.
  + Works well as a baseline for comparison.
* **Limitations**:
  + Assumes feature independence, which may not hold.
  + Challenges with imbalanced classes.

**Summary**

**Accuracy**:

1. **Naive Bayes** achieved the highest accuracy (85%), followed by **Random Forest** (84%), indicating their robustness for the overall dataset.
2. **Logistic Regression** (80.8%) also performed well but struggled more with minority class predictions.
3. **k-NN** showed moderate performance but lacked specific metrics for direct comparison.

**Class Imbalance Handling**:

1. **Logistic Regression** demonstrated effective handling of imbalanced classes through SMOTE, improving recall for minority classes.
2. **Random Forest** and **Naive Bayes** had lower recall and F1-scores for minority classes despite decent overall accuracy.
3. **k-NN** did not explicitly address class imbalance, which likely affected its performance.

#### ****Interpretability and Insights****

* **Logistic Regression** is highly interpretable due to its linear nature, making it suitable for understanding how individual features impact predictions.
* **Random Forest** provides feature importance scores, which can help identify key predictors of accident severity.
* **Naive Bayes** and **k-NN** are less interpretable, with Naive Bayes assuming feature independence and k-NN relying on the proximity of samples.