Table of Contents

[Executive Summary 2](#_Toc195456275)

[Overview Of Solutions 3](#_Toc195456276)

[Data Exploration 4](#_Toc195456277)

[Feature Selection 16](#_Toc195456278)

[Data Modeling 17](#_Toc195456279)

[Model Building 19](#_Toc195456280)

# Executive Summary

This project aimed to accurately classify accident severity. Specifically, it focused on distinguishing between fatal and non-fatal injuries using machine learning. The dataset consisted of 18,957 entries. It included diverse data types and had significant missing values. An initial analysis revealed a pronounced class imbalance. Non-fatal injuries far outnumbered fatal ones. To improve model performance and simplify the dataset, feature selection was conducted. This involved removing redundant and irrelevant columns.

The data modeling pipeline included several key preprocessing steps. These steps were the imputation of missing values, feature selection, resampling using SMOTENC to address class imbalance, and encoding and scaling of features. Multiple models were evaluated, including Logistic Regression, Decision Tree, Random Forest, Neural Network, and LightGBM. The Random Forest model demonstrated the best performance in most metrics, except recall, and consequently f1, in which neural network did slightly better. After hyperparameter tuning using Randomized Search, the optimized Random Forest model achieved the best F1 score of 0.9613 during randomized search but testing metrics stayed similar to the default model. Still, the final evaluation of the fine-tuned model on test data yielded robust metrics: an accuracy of 0.9567, precision of 0.9760, recall of 0.7104, F1 score of 0.8223, and ROC AUC of 0.9687. This project successfully developed a high-performing machine learning pipeline for accident severity classification. The Random Forest model demonstrated strong predictive capability.

# Overview Of Solutions

The primary objective of this project was to build an effective machine learning model. This model aimed to classify accident severity, particularly to differentiate between fatal and non-fatal incidents. To achieve this, a structured pipeline was implemented. This pipeline encompassed feature selection, data preprocessing, and model training.

Several classification models were explored. These included Logistic Regression, Decision Tree, Random Forest, Neural Network, and LightGBM. A key component of the solution was the use of SMOTENC. This is a resampling technique specifically designed for datasets with categorical features. It was used to address the class imbalance identified during data exploration.

The machine learning pipeline streamlined the workflow. It encapsulated all necessary preprocessing steps. These steps included imputation for handling missing data, one-hot encoding for categorical variables, scaling of numerical features, and model training.

Among all evaluated models, the Random Forest classifier demonstrated the highest performance across most key metrics. It achieved an accuracy of 0.9586, precision of 0.9863, and ROC AUC of 0.9693. To further improve this performance, Randomized Search was used. This was done to fine-tune the hyperparameters of the Random Forest model. The fine-tuned model achieved the best F1 score of 0.9613 during randomized search but testing metrics stayed similar to the default model. It continued to perform well during final testing, with metrics indicating both high accuracy and strong generalization.

This end-to-end solution proved effective in handling real-world challenges. These challenges included missing data, imbalanced classes, and high feature dimensionality. As a result, a robust accident severity classification model was developed.

# Data Exploration

The initial data exploration was conducted using a Pandas DataFrame from the pandas library. This revealed a dataset containing 18,957 entries across 54 columns. The data types were diverse. They included float64 (6 columns), int64 (3 columns), and object (45 columns). There was a significant presence of missing values across numerous features. Using functions from both the pandas and numpy libraries, class counts highlighted a substantial imbalance in accident severity. There was a predominance of non-fatal injuries compared to fatalities and property damage.

Statistical assessments were performed with the help of pandas.describe() and numpy functions like mean, median, and variance. These assessments revealed insights into the distribution of numeric variables. For LATITUDE, LONGITUDE, x, and y, the close proximity of mean and median values suggested approximately symmetrical distributions. Conversely, TIME and FATAL\_NO exhibited larger disparities, indicating skewed distributions. Variance analysis further elucidated the spread of data. TIME and FATAL\_NO demonstrated high variance, reflecting a wide range of accident times and fatality counts. The large variances for x and y are likely due to coordinate scaling. In contrast, the low variances for LATITUDE and LONGITUDE suggested a concentration of accidents within a relatively confined geographical area. The mode() function from pandas was employed to identify the most frequent values for each column. This highlighted dominant categories and trends.

A screenshot of a computer screen

AI-generated content may be incorrect.

The heatmap visualizes the mean values of numeric features grouped by accident classification (ACCLASS). It reveals how these values vary across "Fatal," "Non-Fatal Injury," and "Property Damage O" categories. FATAL\_NO has values only for fatal accidents, as expected. Other features like OBJECTID, INDEX, x, and y show no clear pattern. TIME, LATITUDE, and LONGITUDE vary only slightly across accident types. Overall, the heatmap offers a clear comparison of average feature values based on accident severity.

A screenshot of a computer screen

AI-generated content may be incorrect.

This heatmap illustrates the median values of various numeric features grouped by accident classification (ACCLASS). It includes "Fatal," "Non-Fatal Injury," and "Property Damage O." FATAL\_NO contains values only in fatal accidents. The remaining features show some variation across categories but no strong patterns. Overall, the heatmap offers insight into how the central tendencies of these numeric features differ with accident severity.

A screenshot of a computer screen

AI-generated content may be incorrect.

This heatmap shows the variance of numeric features across different accident classifications (ACCLASS), helping to visualize how data spread varies with accident severity. ACCNUM and INDEX display very high variance, especially in "Non-Fatal Injury" cases, indicating a wide range of values within those categories, Likely doe to them being unique identifiers. In contrast, LATITUDE and LONGITUDE show consistently low variance, suggesting location data is relatively stable regardless of accident type. Overall, the heatmap reveals which features have greater variability across accident classes, offering insights into data distribution patterns.

A screenshot of a report

AI-generated content may be incorrect.

This heatmap illustrates the standard deviation of numeric features across different accident classifications (ACCLASS), providing insight into the variability of each feature. ACCNUM and INDEX exhibit high standard deviations in all categories, indicating significant spread in their values. Conversely, LATITUDE and LONGITUDE show very low standard deviations, suggesting minimal variation in location data across accidents. As expected, FATAL\_NO only shows variability for fatal accidents, reinforcing its specific relevance to that category.

A graph of a number of values

AI-generated content may be incorrect.

This bar chart shows the number of missing values across various columns in the dataset. Columns like DISABILITY, ALCOHOL, PEDESTRIAN, and CYCLIST have a high number of missing entries, indicating inconsistent or conditional data entry. In contrast, fields such as ACCLOC, VISIBILITY, and LIGHT have very few or no missing values, suggesting reliable data collection. This highlights the need for data-cleaning strategies, including imputation or removal of heavily incomplete fields, to ensure accurate analysis which is done later.

A screenshot of a map

AI-generated content may be incorrect.

This scatter plot visualizes the geospatial distribution of accidents within a city, using latitude and longitude to map each incident. Accidents are color-coded by severity: blue for non-fatal injuries, orange for fatal accidents, and green for property damage only. The plot shows a high concentration of accidents—especially non-fatal injuries—around major roads and intersections, while fatal and property damage-only accidents are less frequent but occur in similar regions. Overall, the visualization highlights accident hotspots and provides insight into how accident severity varies across the city’s geography.

A map of accident damage

AI-generated content may be incorrect.

This density map visualizes the spatial concentration of accidents across a city, categorized by severity using the ACCLASS variable. Darker shaded areas indicate higher accident density, with non-fatal injury accidents (blue) showing the widest spread and highest concentrations. Fatal accidents (orange) are less dense but tend to occur in similar high-traffic areas, while property damage-only accidents (green) are the least dense and more localized. Overall, the map highlights key accident hotspots, emphasizing the widespread nature of non-fatal injuries compared to more localized severe and minor incidents.

A graph of a number of people with orange lines

AI-generated content may be incorrect.

This line chart illustrates the trend of accident classifications (ACCLASS) over the years. The y-axis represents the count of accidents, while the x-axis represents the years. The plot shows how the number of accidents for each category has changed over time. It also suggests that the number of both fatal and non fatal accidents had rise around 2012 and normalized soon after.

A graph of a number of different types of injury

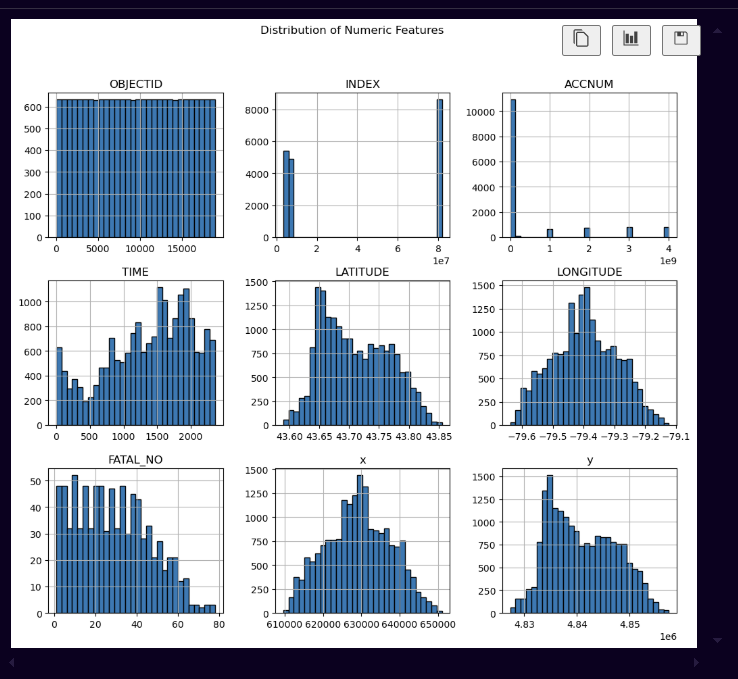
AI-generated content may be incorrect.

This line chart illustrates the trend of accident classifications (ACCLASS) over the months. The y-axis represents the count of accidents, while the x-axis represents the months. The plot shows how the number of accidents for each category changes throughout the year. It suggests that most accidents happen between June and October.

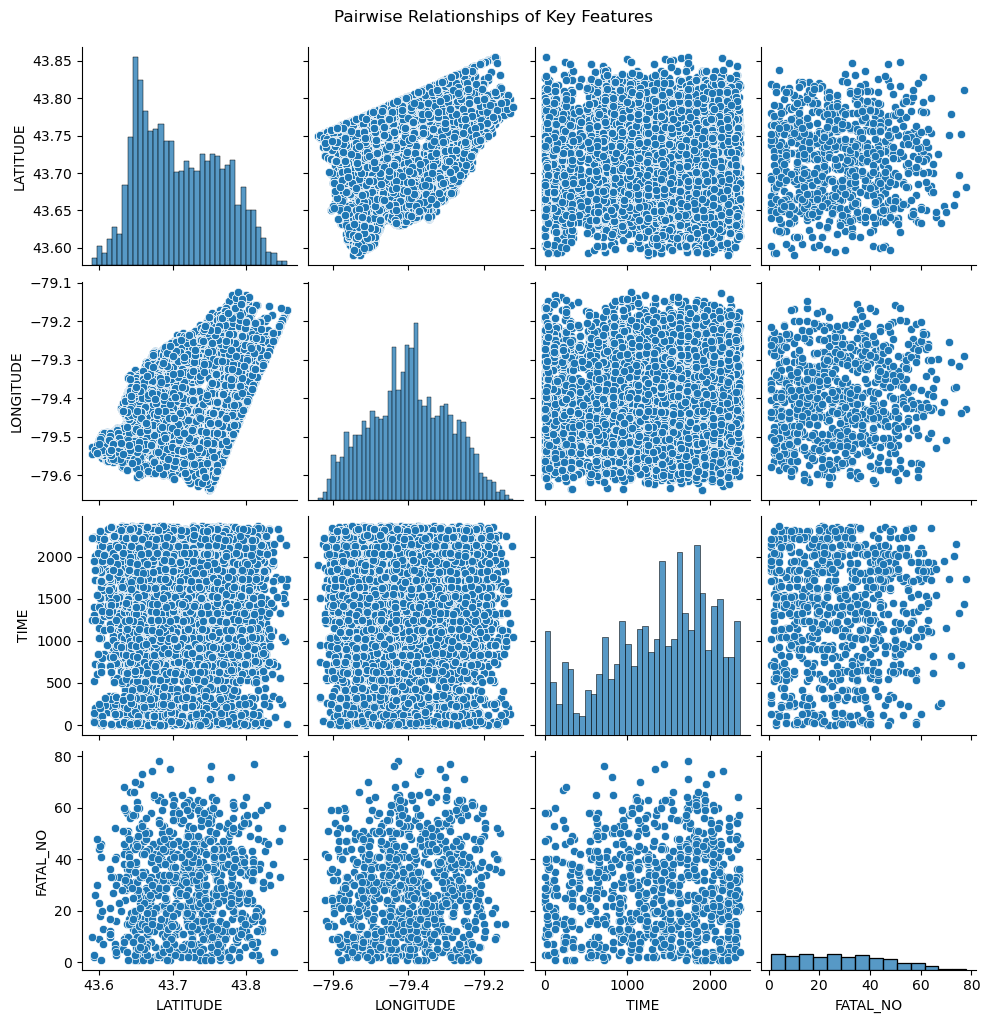
A graph of a number of hours

AI-generated content may be incorrect.

This line chart illustrates the trend of accident classifications (ACCLASS) over the hours. The y-axis represents the count of accidents, while the x-axis represents the hours. The plot shows how the number of accidents for each category changes throughout the day, specifically between 8 AM and 10 AM. The highest number of accidents happen at 8 AM, lower gradually till 9, and rise again from 9 to 10 AM.



The histogram of TIME indicated that accidents occur fairly evenly throughout the day. LONGITUDE and LATITUDE histograms confirmed that a big chunk of accidents are clustered at a specific geographical area.The FATAL\_NO histogram showed a right-skewed distribution, indicating a lower frequency of high fatality accidents. The x and y histograms also displayed bell-shaped distributions, mirroring LONGITUDE and LATITUDE and suggesting they are highly, if not directly correlated.



The pairplot revealed a strong negative linear relationship between LATITUDE and LONGITUDE, confirming a clear geographical trend in accident locations. Both coordinates exhibited unimodal, approximately normal distributions, suggesting clustering around a central point. However, TIME and FATAL\_NO showed scattered patterns with no discernible linear relationships when paired with any of the other variables. This implied that accident times and fatality counts are not strongly linearly correlated with location or with each other in this dataset.

A screenshot of a graph

AI-generated content may be incorrect.

The correlation heatmap revealed relationships between numerical variables in the accident data. Perfect correlations (1.00) between LATITUDE/y and LONGITUDE/x indicated these are identical coordinates presented differently and one pair needs to be removed to avoid multicollinearity. Strong positive correlations were observed between OBJECTID and INDEX (0.88), and INDEX and ACCNUM (0.78), suggesting these are related unique identifiers that need to be removed. A moderate positive correlation (0.42) existed between LATITUDE and LONGITUDE, reflecting their geographical association. However, TIME and FATAL\_NO showed very weak correlations with all other variables, implying limited linear relationships. This suggested that these factors may be influenced by non-linear relationships or other variables not captured in this heatmap. Additionally, the heatmap only reflected linear relationships between numerical data, and did not show relationships between non-numerical data.A graph of a number of blue rectangular objects

AI-generated content may be incorrect.

The class distribution chart highlighted the outcomes of accidents, clearly showing a predominance of non-fatal injuries compared to fatalities and property damage incidents. This class imbalance necessitated the use of SMOTENC during data modeling to ensure a balanced dataset for training the model.

# Feature Selection

Before performing feature selection, the dataset was imputed using mean strategy for numeric columns and most frequent class for categorical column.

Categorical values were temporarily encoded using ordinal encoder, as it leaves the features and dimensionality in tact unlike one hot and doesn’t cause leakage like target encoder. This step is only performed for feature selection, a different encoding strategy is used for training.

Feature selection was carried out using two complementary techniques: Random Forest with the Boruta algorithm and Mutual Information. The Boruta method, which builds on Random Forest, identifies statistically relevant features by comparing them against randomized shadow features in multiple iterations. This helps isolate features that consistently contribute to accurate predictions. The features selected by Boruta were:

['DATE', 'STREET1', 'STREET2', 'INJURY', 'TIME', 'LATITUDE', 'LONGITUDE', 'FATAL\_NO']

In parallel, Mutual Information was used to assess the dependency between each feature and the target variable, producing a ranking based on individual predictive strength.

A graph with blue and white lines

AI-generated content may be incorrect.

This horizontal bar chart ranks features by their Mutual Information scores, highlighting how strongly each feature is related to the target variable. The top 15 features from this ranking were selected for further consideration:

['LONGITUDE', 'DATE', 'LATITUDE', 'INJURY', 'FATAL\_NO', 'STREET2', 'TIME', 'STREET1', 'OFFSET', 'HOOD\_140', 'HOOD\_158', 'NEIGHBOURHOOD\_158', 'IMPACTYPE', 'DRIVCOND', 'NEIGHBOURHOOD\_140']

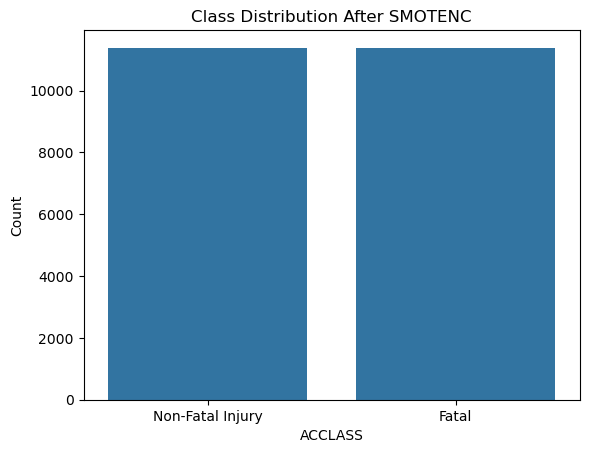
To finalize the feature set for model training, a voting strategy was applied. Only those features that appeared in both the Boruta-selected set and the top 15 Mutual Information features were retained. This intersection ensured the selection of variables that were both statistically significant and highly correlated with the target outcome. The resulting feature set:

['STREET1', 'FATAL\_NO', 'LONGITUDE', 'LATITUDE', 'TIME', 'DATE', 'STREET2', 'INJURY']

# Data Modeling

This section outlines the data modeling process to prepare the data for model training.

* Initial Feature Reduction:
  + Dropping columns OBJECTID, INDEX, and ACCNUM because they are unique identifiers and do not contribute to the predictive power of the model.
  + Removing columns x and y due to their perfect correlation with LONGITUDE and LATITUDE, as they represent the same geographical information.
* Target Definition and Data Cleaning:
  + Defining the target variable ACCLASS and removing rows where it is missing to ensure the model is trained on complete data.
  + Removing rows where ACCLASS is 'Property Damage O' to focus the analysis on predicting fatal and non-fatal injuries. This simplifies the problem to a binary classification.
* Feature Selection:
  + The feature set was reduced to only relevant features using the feature selection strategy described in the Feature Selection section of this document.
* Data Splitting:
  + Splitting the data into features (X) and target (y) using a 70/30 ratio, with 70% of the data used for training and 30% for testing. The data is also stratified on the target variable.
* Resampling:
  + The training portion of the dataset was resampled using SMOTENC as it works well with categorical data, which our dataset mostly consists of.
  + As a result, the records with Fatal and Non-Fatal target classes were balanced:



# Model Building

The model building phase involved creating and evaluating several machine learning pipelines. These pipelines were designed for classifying accident severity.

Each pipeline consisted of several steps:

* Feature Subset Extraction (Mostly useful for testing in the deployed environment):
  + Custom transformer that extracts features selected in the feature selection phase from the testing data in case the original set of features (not only selected) is passed to the model for prediction.
* Imputation (Used for testing in our case as the training set was already imputed during feature selection):
  + Fills the missing values in categorical columns with the most frequent category in said column.
  + Fills the missing values in numeric columns with the mean of all values in said column.
* Feature Scaling:
  + Standardizes all numeric features to be between 0 and 1 to prevent features overpowering each other.
* One Hot Encoding:
  + Replaces categorical features with numeric binary features.
* Classifier:
  + Last step of the pipeline. The machine learning model to be trained and tested on data preprocessed in previous steps.

5 pipelines were created using various machine learning algorithms.

The algorithms tested included Logistic Regression, Decision Tree, Random Forest, Neural Network (MLPClassifier), and LightGBM. SVC was planned to be tested as well, but it took too long to train (even with linear kernel) considering our devices’ low computational power and the size of the dataset. Each model was trained on a balanced, feature selected, imputed, scaled and encoded dataset. It was then evaluated on a feature selected, imputed, scaled and encoded test set using various metrics such as accuracy, precision, recall, F1 score, and ROC AUC. Confusion matrices and ROC curves were generated to further assess the performance of each model.

After evaluating the default configurations of these models, a comparison of their performance metrics was conducted. This was done to identify the best-performing algorithm. The Random Forest model generally showed the strongest results across several metrics. The only metric it lost to Neural Network was recall (and consequently f1), so a Randomized Search was performed on the Random Forest model's hyperparameters. This was done with the scoring strategy set to f1 as an attempt to improve it’s weakest metric. The best parameters found through this search were then used to train a fine-tuned Random Forest model. The fine tuned model achived a very high f1 of 0.9613 during randomized search. However, the metrics on the testing set remained similar to the default Random Forest, meaning that it was already at the highest possible performance achievable with the current dataset or that the randomized search didn’t have enough iterations(15 with 3 folds = 45) to find better parameters (More iterations would be too computationally heavy).

\*Note: In confusion matrices 0 represents positive label (Fatal), and 1 negative (Non-Fatal Injury)

**Logistic Regression Performance**

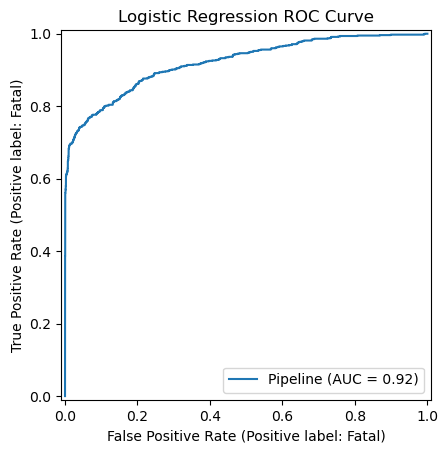
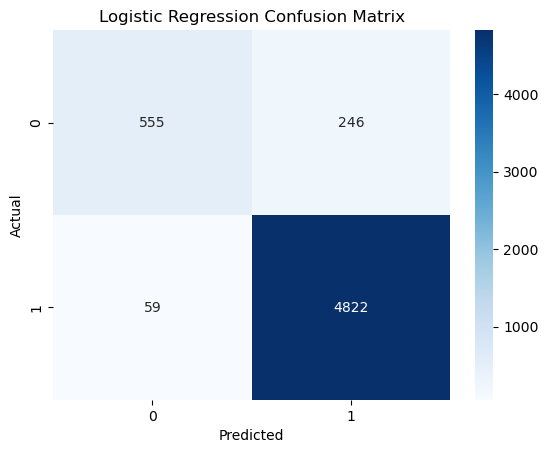
Accuracy : 0.9463

Precision: 0.9039

Recall : 0.6929

F1 Score : 0.7845

ROC AUC : 0.9209



Logistic Regression showed solid baseline performance, with relatively high precision but limited recall. This means it was good at identifying non-fatal cases but missed some fatal ones. The ROC AUC is respectable, placing it behind Random Forest and Neural Network. This model could serve well when false positives are less concerning.

**Decision Tree Performance**

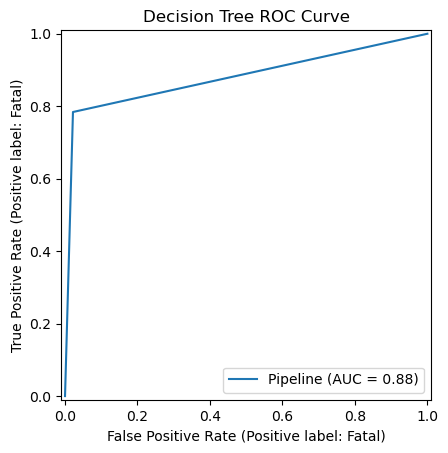
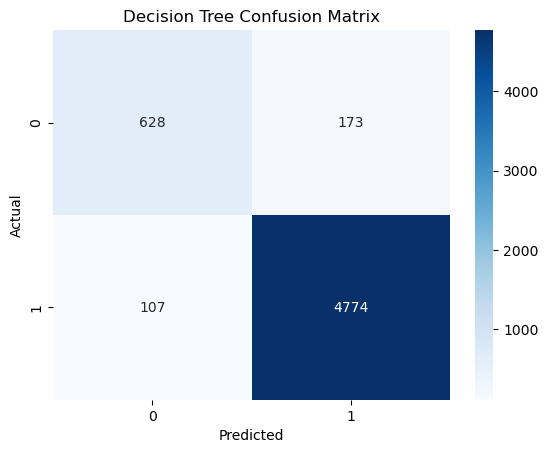
Accuracy : 0.9507

Precision: 0.8544

Recall : 0.7840

F1 Score : 0.8177

ROC AUC : 0.8810



The Decision Tree showed better balance between precision and recall than Logistic Regression. However, it slightly underperformed in ROC AUC compared to others. The model likely overfits slightly, as single trees are prone to, which explains the higher recall but lower AUC.

**Random Forest Performance**

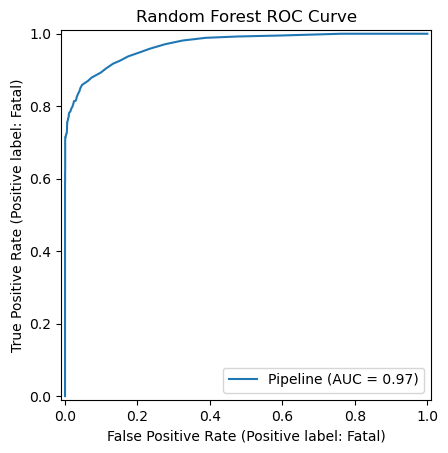
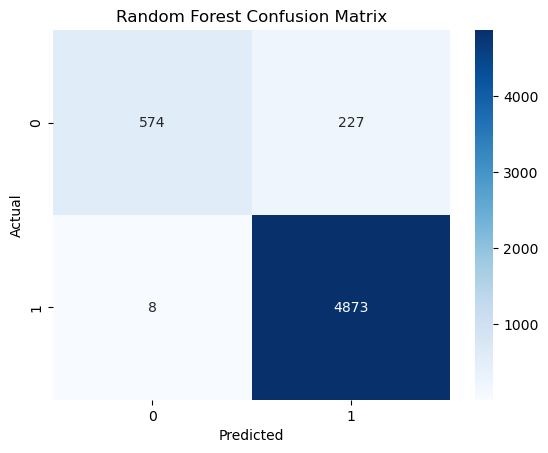
Accuracy : 0.9586

Precision: 0.9863

Recall : 0.7166

F1 Score : 0.8301

ROC AUC : 0.9693



Random Forest had the best accuracy, precision, and ROC AUC of all models. Its recall wasn't the highest but still competitive. The ROC curve showed strong separation ability, and the confusion matrix confirmed few false positives. It ranked highest overall except in recall.

**Neural Network Performance**

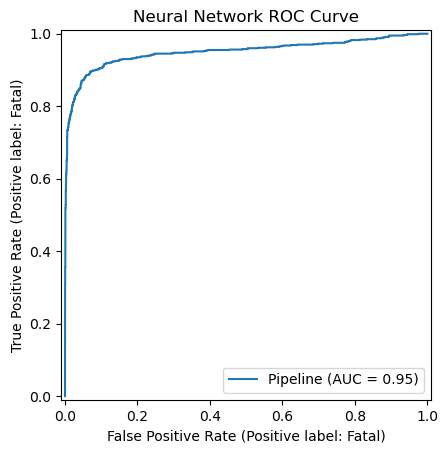
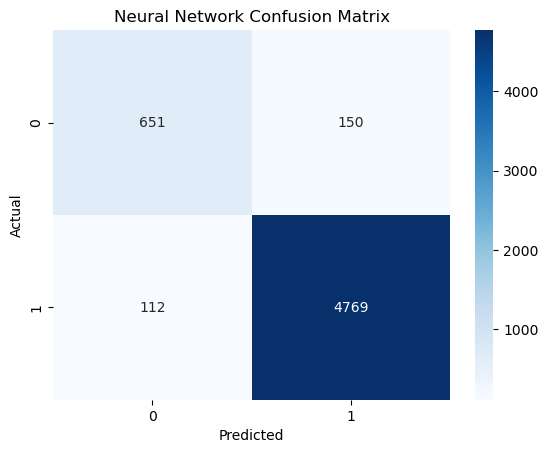
Accuracy : 0.9539

Precision: 0.8532

Recall : 0.8127

F1 Score : 0.8325

ROC AUC : 0.9503



Neural Network achieved the highest recall and F1 score, meaning it was best at capturing fatalities. However, it sacrificed precision and had slightly lower ROC AUC than Random Forest. This model is best when recall is the priority, especially for safety-critical applications.

**LightGBM Performance**

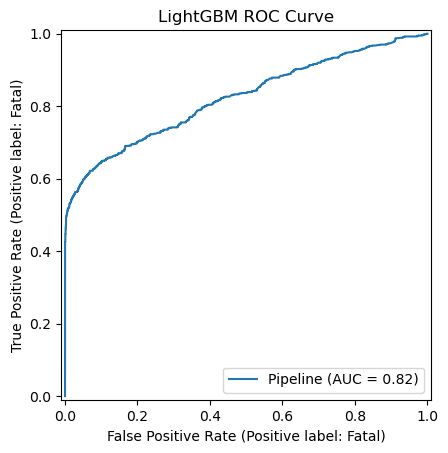
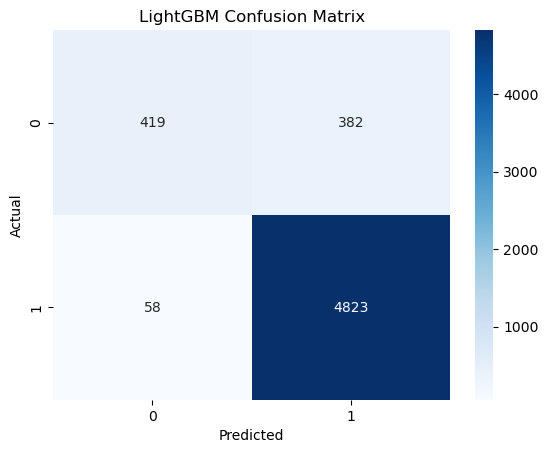
Accuracy : 0.9226

Precision: 0.8784

Recall : 0.5231

F1 Score : 0.6557

ROC AUC : 0.8237



LightGBM underperformed across most metrics, especially recall and F1. While it's efficient and scalable, its default configuration did not adapt well to this dataset. The confusion matrix likely revealed a high number of false negatives, making it a less reliable choice for this task.

**Metrics Comparison**

Accuracy -> Best: Random Forest (0.9586)

Precision -> Best: Random Forest (0.9863)

Recall -> Best: Neural Network (0.8127)

F1 -> Best: Neural Network (0.8325)

Auc -> Best: Random Forest (0.9693)

**Randomized Search On Random Forest**

Best RF Parameters:

{'classifier\_\_n\_estimators': 200, 'classifier\_\_min\_samples\_split': 2, 'classifier\_\_min\_samples\_leaf': 1, 'classifier\_\_max\_features': 'sqrt', 'classifier\_\_max\_depth': None}

Best RF f1:

0.9613477106459619

**Fine-Tuned Random Forest Performance**

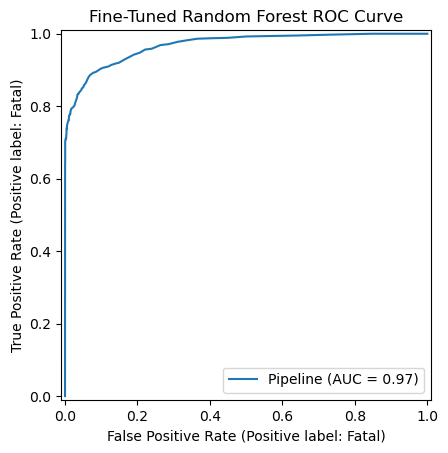
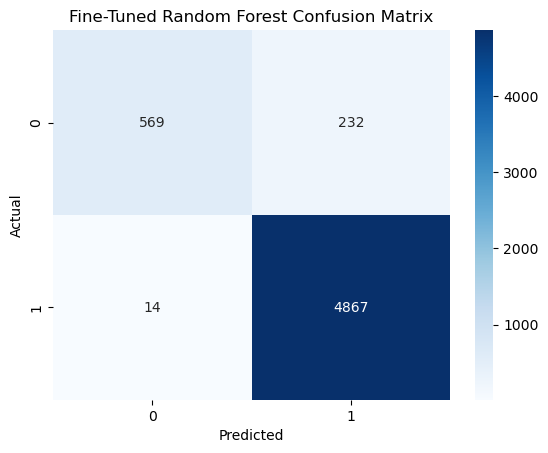
Accuracy : 0.9567

Precision: 0.9760

Recall : 0.7104

F1 Score : 0.8223

ROC AUC : 0.9687



The fine-tuned Random Forest had slightly lower recall and F1 compared to Neural Network, but matched or nearly matched its default version across all metrics. It didn’t drastically outperform the original model, suggesting that the initial configuration was already near optimal.