Motor Vehicle Collisions and Crashes
Final Project Report
Group 5

IST 718 Big Data Analytics

By
Kartik Kaul
Kunwar Uday Singh
Mahitha Chennamadhava
Sisira Pathakamuri

Project Overview

Introduction:

This project analyzes a comprehensive dataset of motor vehicle collisions in New York, exploring factors contributing to accidents and their impacts. It aims to identify patterns, high-risk areas, and causes, using advanced analytics and visualizations. The goal is to inform road safety strategies and create safer roads, with the report detailing findings, methodologies, and recommendations from the dataset analysis.[1]

Objective:

The main objective of this project is to meticulously analyze the Motor Vehicle Collisions dataset to unearth critical insights into road safety. We intend to recognize recurring patterns, such as peak collision times and frequent causative factors, to gain a holistic understanding of the conditions under which these accidents most often occur. A pivotal part of our analysis focuses on assessing high-risk zones or specific segments of roadways with increased collision incidences, aiming to pinpoint potential danger zones. Beyond mere frequency, we're also committed to delving deep into the causes of these collisions, discerning whether they stem from environmental factors, vehicular malfunctions, or human errors. Another integral aspect is the evaluation of accident severity in terms of injuries, fatalities, and property damage, which will help prioritize areas needing urgent interventions. Additionally, by examining temporal trends, we aim to ascertain the periods whether daily, weekly, or seasonally, when the risks are heightened. Based on our findings, we will be formulating recommendations tailored for policymakers, urban planners, and traffic authorities. These recommendations will be geared towards not only improving road infrastructure but also promoting safer travel conditions and proactive road safety measures for all.[1]

Dataset Description:

Source: Data.gov

The dataset provides a detailed account of road accidents occurring in a specified area. Below is the breakdown of the dataset's structure and the type of information it contains:

1. **Time and Date**: Timestamps indicating when each collision occurred.

2. **Location Data**: Geographical coordinates (latitude and longitude) or street names pinpointing the exact location of the accident.

3. **Vehicle Information**: Details regarding the vehicles involved, which could include type, make, model, and possibly the condition or any malfunctions.

4. **Injury and Fatality Data**: Information about the number and severity of injuries or fatalities resulting from each collision.

5. **Causal Factors**: Descriptions or codes signifying the reasons behind the accident, be it environmental conditions, human errors, or vehicle-related issues.

6. **Additional Attributes**: Other relevant data points such as on and off-street name, zip code, and borough.

The dataset consists of **31 variables** and **103,880 records**:

```
df.columns
```

```
['CRASH DATE',
 'CRASH TIME',
 'BOROUGH',
 'ZIP CODE',
 'LATITUDE',
 'LONGITUDE',
 'LOCATION',
 'ON STREET NAME',
 'CROSS STREET NAME',
 'OFF STREET NAME',
 'NUMBER OF PERSONS INJURED',
 'NUMBER OF PERSONS KILLED',
 'NUMBER OF PEDESTRIANS INJURED',
 'NUMBER OF PEDESTRIANS KILLED',
 'NUMBER OF CYCLIST INJURED',
 'NUMBER OF CYCLIST KILLED',
 'NUMBER OF MOTORIST INJURED',
 'NUMBER OF MOTORIST KILLED',
 'CONTRIBUTING FACTOR VEHICLE 1',
 'CONTRIBUTING FACTOR VEHICLE 2',
 'CONTRIBUTING FACTOR VEHICLE 3',
 'CONTRIBUTING FACTOR VEHICLE 4',
 'CONTRIBUTING FACTOR VEHICLE 5',
 'COLLISION_ID',
 'VEHICLE TYPE CODE 1',
 'VEHICLE TYPE CODE 2',
 'VEHICLE TYPE CODE 3',
 'VEHICLE TYPE CODE 4',
 'VEHICLE TYPE CODE 5']
```

Prediction, Inference and Goals:

Predictors:

Following are the top prospective predictors:

1. **Location**: These geospatial variables are essential for any location-based analysis and to identify high-risk zones.

2. **Crash date and time**: Temporal data can reveal patterns related to time-of-day, day-of-week, or seasonal trends in collisions.

3. **Contributing Factor**: The primary contributing factor can be instrumental in understanding the root cause of the collision and secondary factors can provide more depth to the analysis, especially in multi-vehicle collisions.

4. **Borough**: Different boroughs or districts can have varied infrastructure, traffic density, and patterns.

5. **Number of people injured or killed**: These provide a direct measure of the severity of the accident.

6. **Vehicle Type**: The type of vehicle involved can influence both the likelihood and the nature of a collision.

7. **On Street and cross street name**: Specific streets or intersections might have higher risks due to factors like traffic volume, visibility, or street design.

8. **Number of motorists injured or killed**: These shed light on the impact of collisions on motorists, which can vary based on vehicle type and collision dynamics.

Predictions:

1. **Number of people injured or killed**: Predicting the number of people killed or injured likely to be involved in accidents can guide vehicle-specific safety interventions.
2. **Severity of Accident**: Based on the number of people injured or killed we can classify accidents as "No Injury", "Injury", or "Fatal".

Inference Approach:

Utilizing a range of machine learning models to infer patterns and trends:
Decision Tree, Random Forest, Linear Regression and multi class logistic regression

Goals:

- Focus Areas: Identifying peak collision times, causative factors, and high-risk zones.
- In-Depth Analysis: Examining the causes of collisions including environmental, vehicular, and human factors.
- Severity Evaluation: Assessing accident severity based on injuries, fatalities, and property damage.
- Temporal Analysis: Identifying high-risk periods for collisions.
- Outcome: Formulating recommendations for policymakers and traffic authorities to improve road safety.

Data exploration.

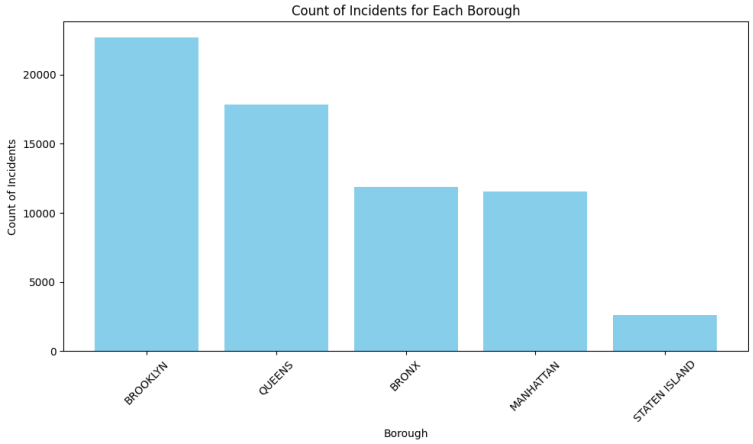
1. **Data Description:** Utilized df.describe().show() , df.columns to obtain descriptive statistics for each column.
2. **Null Values:** Counted null values in each column to assess data completeness. Significant null values found in 'BOROUGH', 'ZIP CODE', 'LATITUDE', 'LONGITUDE', and 'LOCATION' among others.
3. **Data Cleaning:** Dropped 11 rows with null values in critical columns to ensure data integrity for analysis.

4. **Data Type Conversion:** Cast 'ZIP CODE' and various injury and fatality count columns to integer type for correct data manipulation.
5. **Duplicate Removal:** Removed duplicate rows to avoid skewing the results with redundant information.
6. **Column Renaming:** Renamed columns for clarity, e.g., 'NUMBER OF PERSONS INJURED' to 'NUM_PER_INJUR'.
7. **Data Transformation:** Transformed the 'CRASH TIME' column into an 'O'Clock' format for standardized analysis.
8. **Handling Missing Values:** Filled NA/missing values for certain columns with default values where appropriate.

Data Exploration Insights:

1. **Vehicle Insights**: Using vehicle type code, we can determine which vehicle types are most often involved in collisions.
2. **Borough-based Analysis**: We can compare boroughs data to see if certain boroughs have higher collision rates or severities.
3. **Causal Analysis**: We can rank 'contributing factor vehicle' data to identify the most frequent reasons for collisions.

Borough based analysis:



The borough with the highest count (Brooklyn) indicates the area with the most collisions. This could be due to various factors, including higher traffic, infrastructure issues, or a larger population.
Conversely, the borough with the lowest count (Staten Island) had the fewest collisions, indicating potentially better traffic conditions, fewer vehicles, or effective traffic safety measures. This breakdown by borough can be crucial for city planners or traffic safety officials when deciding where to allocate resources or implement safety interventions.

Geospatial Analysis:

The results tell us the number of incidents or occurrences at each specific location in your dataset. If some locations have particularly high counts, they might be considered "hotspots" or areas with a high concentration of incidents.

This can be valuable information for many purposes, like identifying areas of concern, or planning interventions or strategies based on geographic concentrations.

LATITUDE	LONGITUDE	count
40.668896	-73.95339	1
40.679344	-73.859535	9
40.840775	-73.87246	6
40.84744	-73.89968	15
40.65047	-73.917366	1
40.54048	-74.153404	2
40.738403	-73.93864	11
40.786854	-73.82236	6
40.605022	-74.00264	1
40.7802	-73.98755	2
40.81689	-73.86383	4
40.727192	-73.89328	7
40.639996	-74.02442	7
40.634117	-73.94588	7
40.74128	-73.90257	31
40.85429	-73.90027	32
40.687332	-73.927795	1
40.696033	-73.98453	80
40.758602	-73.91379	3
40.653564	-73.934494	5
40.685898	-73.83069	1
40.597023	-73.98815	1
40.756397	-73.804535	12
40.613052	-73.94913	17
40.81572	-73.925064	13
40.642445	-74.00592	7
40.70921	-73.87017	11
40.836937	-73.927124	12
40.766705	-73.89024	11
40.64571	-73.91206	2
40.70768	-73.74512	5
40.740623	-73.78024	2
40.662743	-73.921936	22
40.66016	-73.75579	3
40.66812	-73.92842	1
40.668507	-73.92561	56

Vehicle Type Analysis:

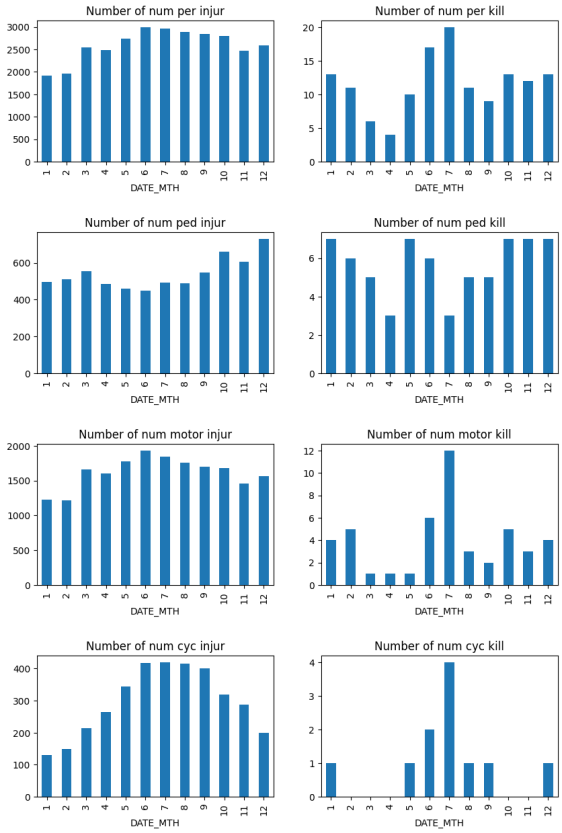
The initial output lists various vehicle types and the number of times they appear in the dataset.
The vehicle type "Sedan" is the most frequent, appearing 139964 times.
The next most frequent is "Station wagon/sports utility" with 105379 occurrences.
The third is "Taxi" with 8519 occurrences.

VEH_TYPE_1	incident_count
Sedan	31407
Station Wagon/Spo...	22907
Taxi	1700
Pick-up Truck	1348
Bus	1206
Bike	1137
Box Truck	1109
NULL	1088
E-Bike	625
Motorcycle	512
Ambulance	456
E-Scooter	400
Van	388
Tractor Truck Diesel	352
Dump	209
Moped	200
PK	145
Garbage or Refuse	143
Convertible	129
Flat Bed	101

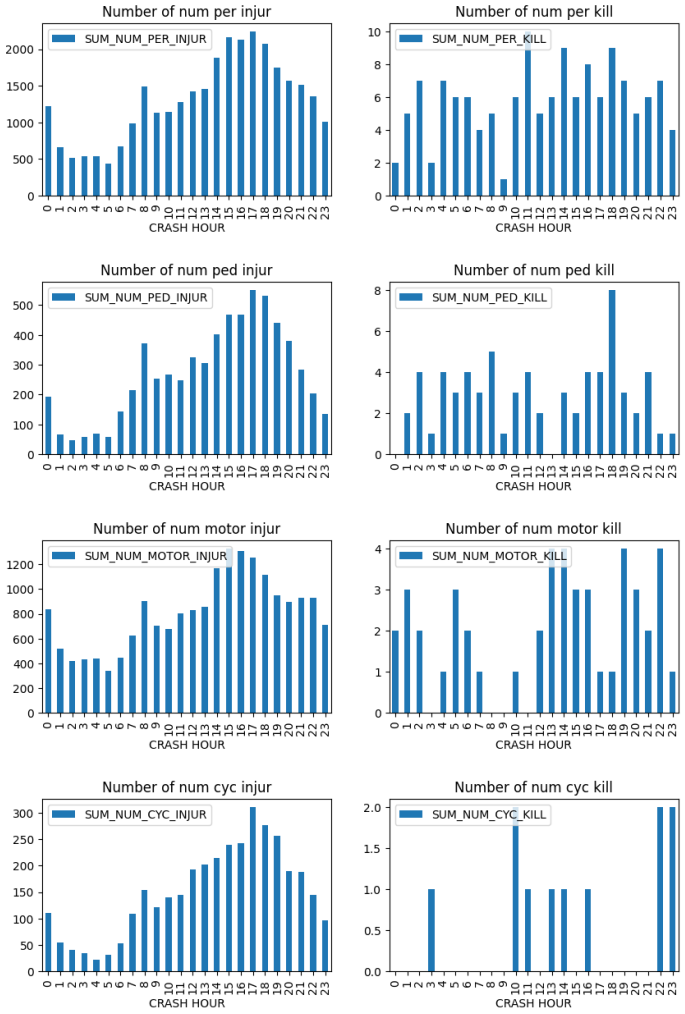
Data Analysis and Modelling strategy:

1. Data cleaning and preprocessing using Spark which includes dropping all irrelevant columns, removing outliers and null values and correcting data types of variables.
2. Perform exploratory data analysis to know about the variables and their descriptive statistics.
3. Performing resampling i.e., up and down sampling to make equal number of examples for both classes.
4. Perform feature selection to avoid underfitting and overfitting.
5. Perform feature scaling using standard scaler.
6. Model creation and training.
7. Model evaluation using appropriate evaluation metrics.

Interesting/surprising results.



The trend of injuries and fatalities showed specific patterns and seasonal variations as some months showed higher incidents, which could indicate seasonal factors affecting collision rates.



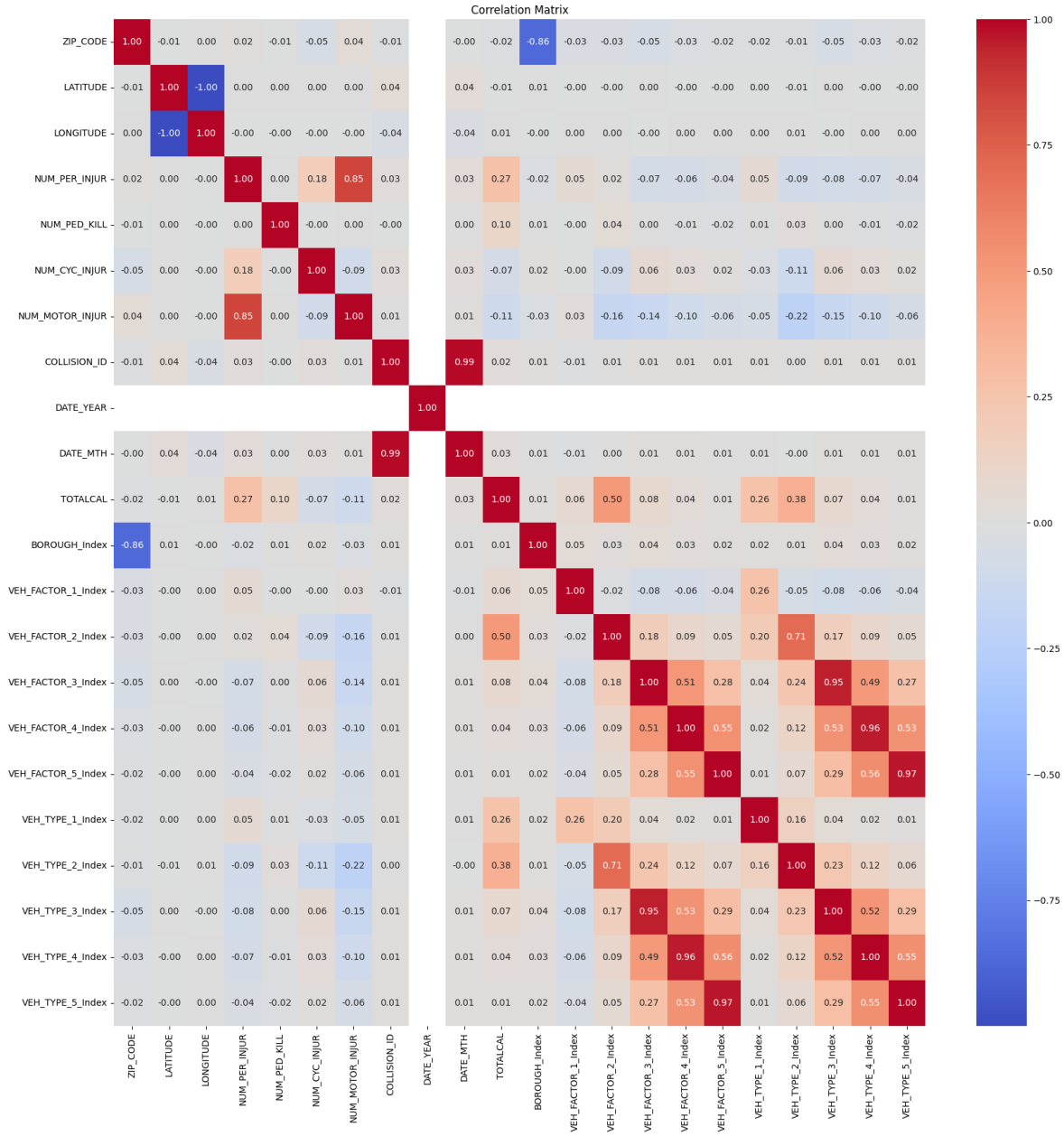
Also, analysis shows hourly variations in the number of incidents which can be due to peak rush in the evening with most of incidents occurring between 15-22 hour of the day i.e 3pm – 10pm.

Methods used to solve the problem.

Feature Engineering:

Vector Assembly: A Vector Assembler is used to combine a set of input columns into a single vector column. The input columns for the vector assembler include various features like 'ZIP_CODE', 'LATITUDE', 'LONGITUDE', and specific injury and accident type counts. The exact list of features used is determined by excluding certain columns like 'CRASH TIME', 'CRASH DATE', and others that are either non-numeric or not relevant for the prediction task. Output: The output is a DataFrame with an additional column named "features", which contains the assembled feature vector for each record.

Correlation Matrix: This is done using the Correlation.corr function in PySpark. The correlation matrix includes the features assembled in the previous step. It is visualized using Seaborn and Matplotlib, helps in understanding how different features are related to each other.



Machine Learning Models implemented:

1. **Linear Regression:** The goal was to predict the 'TOTALCAL' (a derived column representing total casualties).

- 'ZIP_CODE', 'LATITUDE', 'LONGITUDE', etc., as features. These features are predictors for the target variable 'TOTALCAL'.
- Used Vector Assembler to combine these feature columns into a single vector column named 'features'.
- The dataset is split into training and test sets, to evaluate the model's performance on test data.

- A Linear Regression object is initialized with 'features' as the input column and 'TOTALCAL' as the target column.
- The model is trained on the training dataset. The trained model is used to make predictions on the test dataset.
- Model Evaluation: The predictions are evaluated using the Root Mean Squared Error (RMSE) and R-squared metrics.

2. **Random Forest Classifier** : The goal was to classify the severity of accidents into categories (like 'Fatal', 'Injury', 'No Injury'). Steps:

- Feature Selection and Transformation:
    1. A subset of the features used in regression, 'VEH_TYPE_1_Index', 'ZIP_CODE', 'LONGITUDE', 'LATITUDE','BOROUGH_Index', 'DATE_MTH' are features chosen based relevance to the classification task.
    2. Applied StringIndexer to convert the categorical 'Severity' column into numeric labels.
    3. Used VectorAssembler to combine feature columns into a single vector.

- Initialize a RandomForestClassifier with the specified parameters.
- Created a pipeline that includes feature vector assembly, label indexing, and the classifier.
- Fit the pipeline to the DataFrame, which involves executing all stages in the pipeline on the training data.
- After training, extract feature importance from the Random Forest model.
- Use the trained model to make predictions on new data created.

3. Decision Tree Classifier: The goal was similar to the Random Forest Classifier, to classify the severity of accidents into categories.

- Similar to the Random Forest model, features are selected, transformed, and assembled.
- A DecisionTreeClassifier is initialized with the feature column and label column specified.
- A pipeline is created, including the feature assembler, label indexer, and the decision tree classifier.
- Train the model on the training data, made predictions using the trained model on the test data.

4. Logistic Regression (Multi-class): The goal was to perform multi-class classification on the severity of accidents using the hyperparameter tuning Steps [2]:

- Similar steps of feature selection, transformation, and assembly are followed.
- Initialize a LogisticRegression model for multi-class classification.
- Hyperparameter Tuning:

1. Defined a parameter grid for hyperparameters like 'regParam' and 'elasticNetParam'.
2. Used Cross Validation with a MulticlassClassificationEvaluator for finding the best model.

- The cross-validation process trains multiple models with different combinations of hyperparameters and selects the best one based on the accuracy.
- Model Evaluation: The best model from cross-validation is used to make predictions on the test data.

Results summary:

Regression Model	R-Squared(%)	RMSE
Linear regression	73.03	0.1698

The linear regression model has an R-squared value of 72.21%, indicating that approximately 72% of the variability in the dependent variable can be explained by the model. The Root Mean Square Error (RMSE) is 0.1698, reflecting the average deviation of the predicted values from the observed values.

Classification Model	Accuracy(%)	Precision(%)	Recall(%)	F1 score
Random Forest	66.19	70.37	66.19	57
Decision Tree	66.79	68.87	66.79	59.22
Multi class logistic regression	63.87	67.70	63.87	56

Comparative analysis:

We can compare the performance of the three classification models: Random Forest, Decision Tree, and Multi-class Logistic Regression using the metrics of Accuracy, Precision, Recall, and F1 score:

The Decision Tree model has a slightly higher accuracy than the Random Forest, and both outperform the Multi-class Logistic Regression.

The Random Forest model has the highest precision, suggesting that when it predicts a class label, it is correct more often than the other models. Recall is identical to accuracy in this summary, which may be due to the way the models are evaluated. The Decision Tree has a slightly higher recall than the Random Forest, indicating it is slightly better at identifying all relevant instances.

The F1 score is the harmonic mean of precision and recall, giving a balance between the two. The Decision Tree model has the highest F1 score, indicating a better balance between precision and recall compared to the Random Forest model.

Decision Tree model appears to have a slight edge overall due to its higher accuracy and F1 score. It may be because of computational efficiency of the models, the interpretability of the results, and how they handle overfitting.

Problems encountered:

- Had to deal with too many categorical variables.
- Data size is too large
- Weak correlation with predicted variable.

- Finding the right set of features for random forest classifier

Achieved your prediction and inference goals.

Yes, we achieved our prediction and inference goals. However the results can be improvised by performing and trying more feature selection and transformation options.

Citations

1. Baryshnikova, Nadezhda. “Motor vehicle collisions” Volume 337, New York, 2023. https://www.sciencedirect.com/science/article/pii/S026974912301597X

2. Abramovich, Felix. “Multiclass classification by multinomial logistic regression” Volume 1, New York, 2020. https://arxiv.org/abs/2003.01951