PROJECT

Accident Severity Prediction

By Saad Mohiuddin Ajaz
Mohammed Anwar Ahmed
Nahri Syed Hyder Hussaini

PROJECT OVERVIEW

OBJECTIVE:

To predict injury severity in traffic accidents using machine learning.

KEY FOCUS AREAS:

- Data Preprocessing
- Balancing Techniques
- Model Training and Evaluation
- Actionable Recommendations

PROJECT BENEFITS

Benefits of Accident Severity Prediction

- Enhanced Traffic Safety: Improve preventive measures and emergency responses.
- Data-Driven Insights: Facilitate data-driven decision-making in traffic management.
- Model Effectiveness: Develop robust predictive models for injury severity.

DATA OVERVIEW

Data Description

- **Dataset:** The Motor Vehicle Collisions dataset contains information from all police-reported motor vehicle collisions in NYC. It spans from April 2016, when crash reporting transitioned to an electronic system. The dataset comprises nearly 2,30,000 records, detailing various aspects of each collision.
- **Key Variables:** PERSON_INJURY, PERSON_AGE, PERSON_SEX, SAFETY_EQUIPMENT, EJECTION, BODILY_INJURY, and COMPLAINT.
- **Challenges:** Missing data and imbalanced data set which includes very less instances of minority class in target variable.

DATA PREPROCESSING

Data Preprocessing Steps

- **Cleaning Steps:**
- Handling missing values.
- Dropping irrelevant columns.
- **Feature Engineering:**
- Creation of new features (e.g., 'DATETIME' from 'CRASH_DATE' and 'CRASH_TIME').
- **Encoding:**
- One-hot encoding for categorical variables.
- Label encoding for the target variable.

```
# Check for missing values
missing_values = data.isnull().sum()

# Display columns with missing values.
print(missing_values[missing_values > 0])
```

```
# Drop irrelevant columns
data_cleaned = data.drop(columns=['UNIQUE_ID', 'COLLISION_ID', 'PERSON_ID', 'VEH
```

```
# Convert categorical variables using one-hot encoding
data_encoded = pd.get_dummies(data_cleaned, columns=['PERSON_TYPE', 'SAFETY_EQUI
```

```
# Create a new 'DATETIME' feature from 'CRASH_DATE' and 'CRASH_TIME'
data_encoded['CRASH_DATETIME'] = pd.to_datetime(data_encoded['CRASH_DATE'] + ' '

# Drop the original 'CRASH_DATE' and 'CRASH_TIME' columns
data_encoded = data_encoded.drop(columns=['CRASH_DATE', 'CRASH_TIME'])
```

Current Progress

```python
from sklearn.model_selection import train_test_split

# Define features and target
X = data_encoded.drop(columns=['PERSON_INJURY'])
y = data_encoded['PERSON_INJURY']

# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_

# Verify the split
print(f'Training data: {X_train.shape}, {y_train.shape}')
print(f'Test data: {X_test.shape}, {y_test.shape}')
```

```
Training data: (184528, 189), (184528,)
Test data: (46132, 189), (46132,)
```

```python
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix, roc_auc_scor

# Initialize and train the model
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)



# Predict class probabilities
y_pred_proba = model.predict_proba(X_test)

# Evaluate the model
y_pred = model.predict(X_test)
print(classification_report(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
print(f'ROC AUC Score: {roc_auc_score(y_test, y_pred_proba, multi_class="ovr")}')
```

Intermediate Results

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     45916
           1       0.89      0.29      0.44       215
           2       1.00      1.00      1.00         1

    accuracy                           1.00     46132
   macro avg       0.96      0.76      0.81     46132
weighted avg       1.00      1.00      1.00     46132

[[45908     8     0]
 [  153    62     0]
 [    0     0     1]]
ROC AUC Score: 0.914642002602385
```

THANK YOU

By Saad Mahmood Ajaz

Questions?

Email:saadajaz20@gmail.com