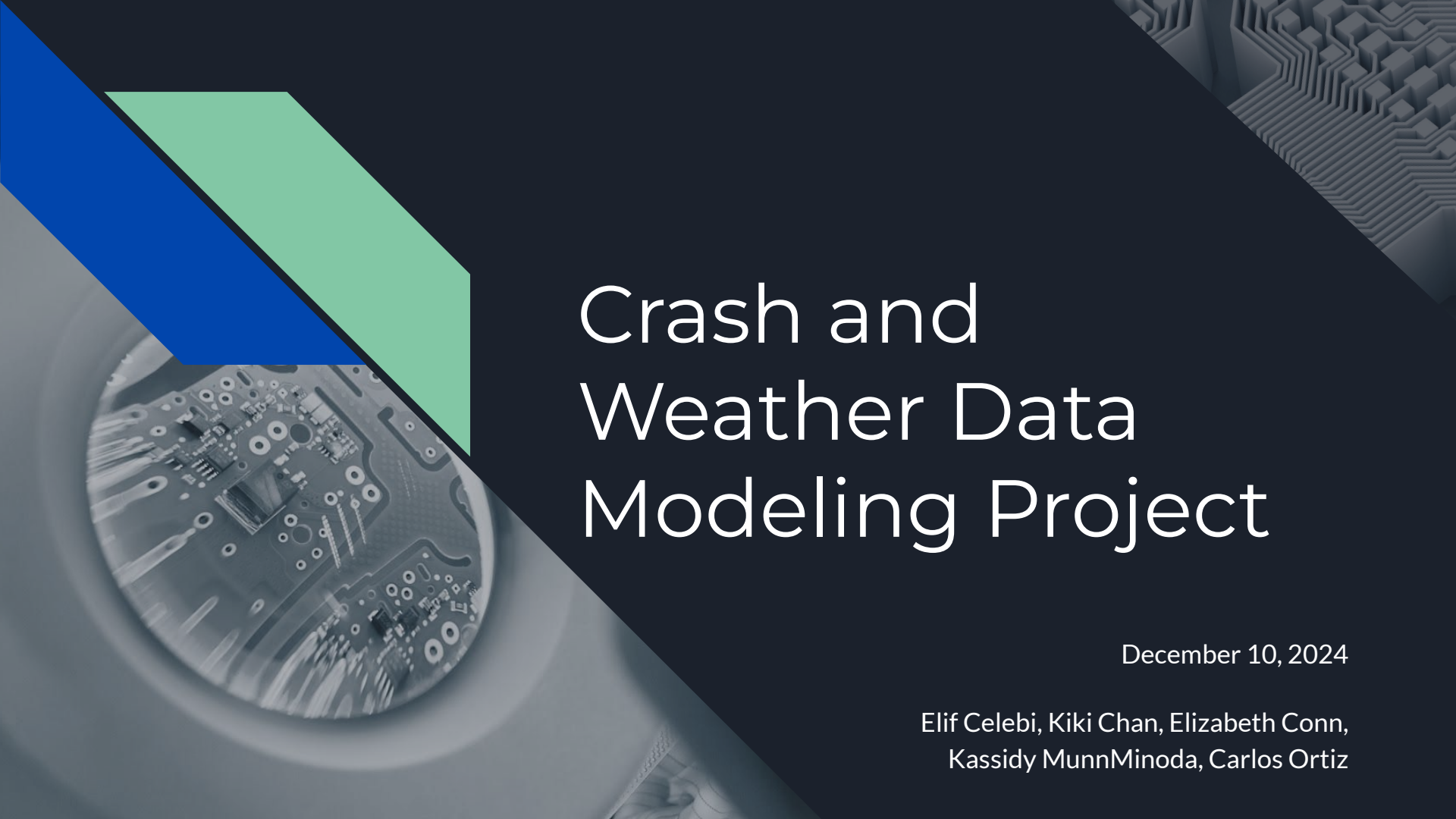


Crash and Weather Data Modeling Project

December 10, 2024

Elif Celebi, Kiki Chan, Elizabeth Conn,
Kassidy MunnMinoda, Carlos Ortiz

Overview

This project analyzes crash and weather data to build a predictive model evaluating the likelihood of injuries during motor vehicle accidents. The workflow includes data preprocessing, dataset integration, machine learning model implementation, optimization, and documentation. The goal is to derive meaningful predictive power with at least 75% classification accuracy or an R-squared value of 0.80, aligning weather conditions with crash patterns.

Tools and Technologies Used

Python: Programming language.

Keras-Tuner: For hyperparameter optimization.

TensorFlow: Neural network model implementation.

Scikit-learn: Random Forest classifier and preprocessing utilities.

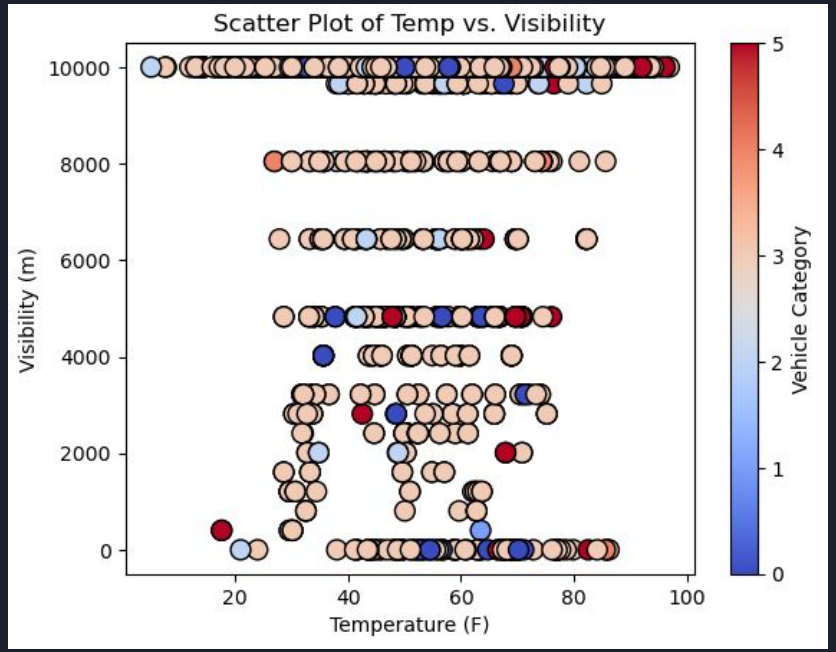
Pandas and NumPy: Data manipulation and analysis.

Matplotlib: Data visualization.

Data Visualization

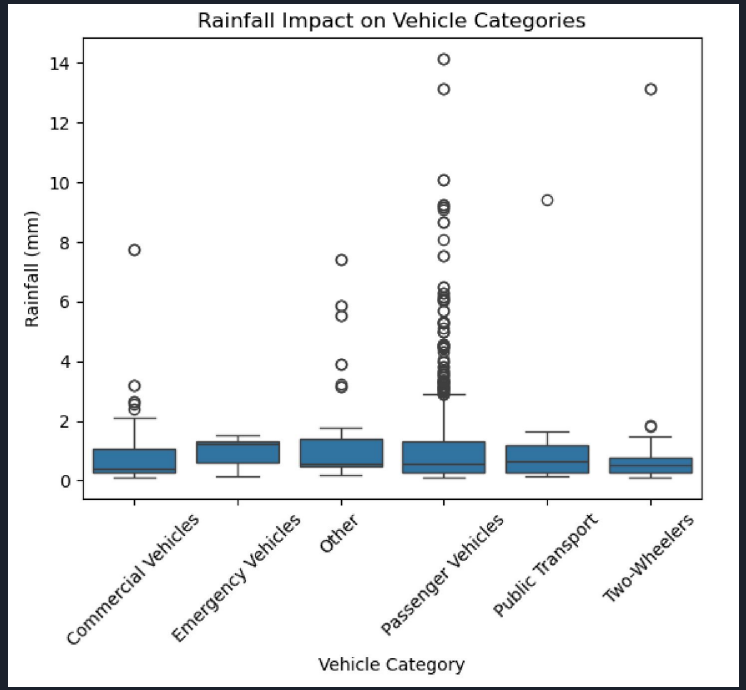
Categorized the vehicle:

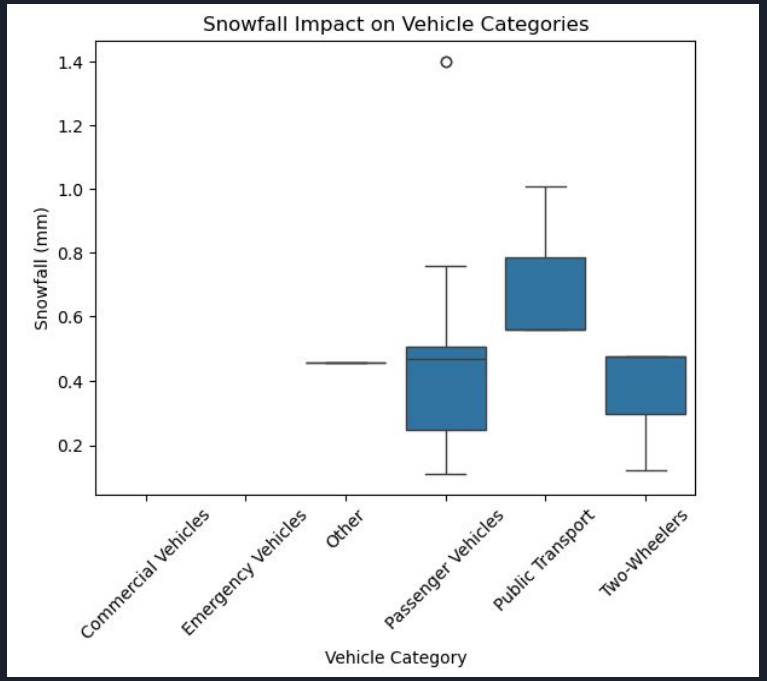
- Passenger Vehicle - Sedan, 4 dr sedan, Station Wagon/Sport Utility Vehicle, Convertible, Taxi, 3-Door
- Commercial Vehicles - Pick-up Truck, Box Truck, Dump, Flat Bed, Garbage or Refuse, Tanker, Tractor Truck Diesel, Tractor Truck Gasoline, Van, Carry All
- Emergency Vehicles - Ambulance
- Two-Wheelers - Motorcycle, Bike, E-Bike, Moped, E-Scooter, MOTOR SCOO
- Other - Unspecified or Blank



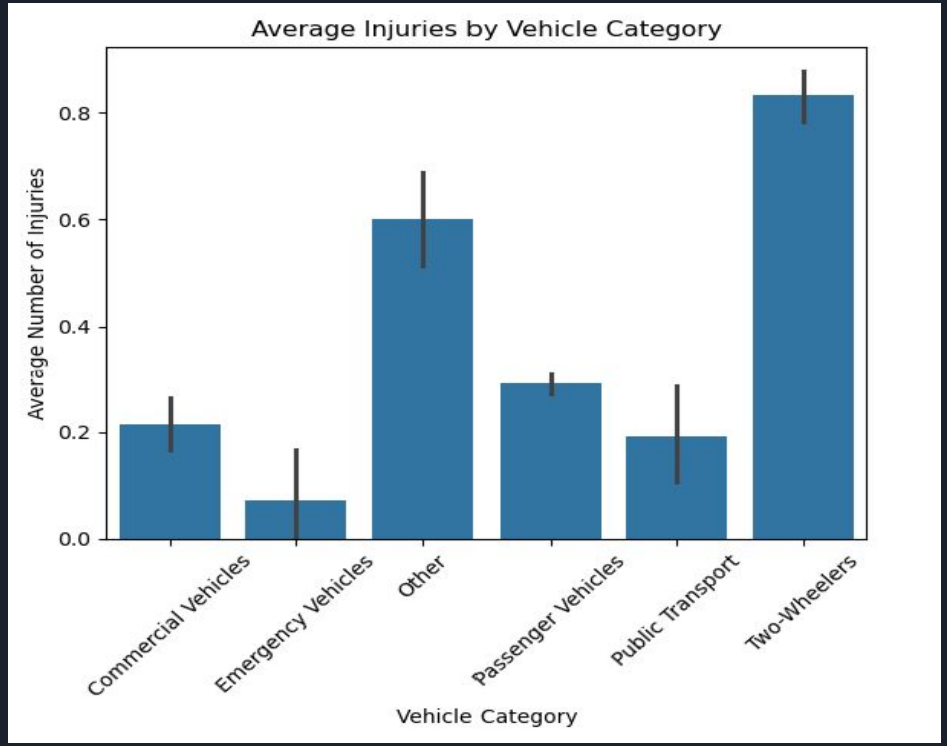
Commercial Vehicles: 0, Emergency Vehicles: 1, Other: 2, Passenger Vehicles: 3, Public Transport: 4, Two-Wheelers: 5

Box Plot


Rainfall Impact on Vehicle Categories
Rainfall (mm)
0
2
4
6
8
10
12
14
Commercial Vehicles
Emergency Vehicles
Other
Passenger Vehicles
Public Transport
Two-Wheelers
Vehicle Category


Snowfall Impact on Vehicle Categories
Snowfall (mm)
0.2
0.4
0.6
0.8
1.0
1.2
1.4
Commercial Vehicles
Emergency Vehicles
Other
Passenger Vehicles
Public Transport
Two-Wheelers
Vehicle Category

Average Injuries



Deep Neural Network

Inputs (feature set)	Temperature, visibility, humidity, wind speed, weather description, borough, zip code, contributing factor, vehicle type
Output (target variable)	Number of persons injured
Hidden layers	2 layers with 8 and 5 nodes
Activation functions	ReLU for hidden layers, Sigmoid for output
Loss function	Binary Crossentropy
Optimizer	Adam
Validation accuracy	59.7%
Optimization	Best model hyperparameters with validation accuracy of 59.3%

Data Cleaning and Random Forest

- Filtering by distance
- Accuracy: 96.8%
- Advantages
  - Can model complex relationships
  - Combining predictions of many trees leads to more generalized results

```
# Ensure merged_data_cleaned is a standalone DataFrame
merged_data_cleaned = merged_data_cleaned.copy()

# Apply the distance calculation
merged_data_cleaned.loc[:, 'distance'] = merged_data_cleaned.apply(calculate_distance, axis=1)

# Drop rows with invalid distances
merged_data_cleaned = merged_data_cleaned.dropna(subset=['distance'])

# Filter by proximity threshold
merged_data_filtered = merged_data_cleaned[merged_data_cleaned['distance'] <= 10000]
```

```
# Reshape y if it's already a NumPy array
y = y.ravel()

# Using Random Forest
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train a model
clf = RandomForestClassifier()
clf.fit(X_train, y_train)

# Predict and evaluate
y_pred = clf.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.9682866961217477
```

Results and Observations

Neural Network:

- Validation accuracy: 59.7%.
- Keras Tuner was useful for exploring hyperparameter configurations but required significant runtime (11 minutes for 60 trials).
- Neural network accuracy was limited due to the structured nature of the data.

Random Forest:

- Test accuracy: 96.8%.
- Performed significantly better than the neural network, demonstrating the suitability of ensemble methods for this dataset.

Challenges and Considerations

01 Ensuring precise alignment of crash and weather data required careful datetime normalization.

02 Filtering rows by proximity using geodesic distances helped improve the quality of the merged data.

03 Missing or mismatched coordinates were handled through cleaning and filtering steps.

Future Improvements

01 Expand Data Sources: Incorporate additional datasets, such as traffic density or road conditions, to improve model robustness.

02 Experiment with Advanced Models: Explore gradient boosting techniques such as XGBoost or LightGBM for further performance gains.

03 Experiment with Advanced Models: Explore gradient boosting techniques such as XGBoost or LightGBM for further performance gains.

04 Refine Feature Engineering: Investigate additional derived features to enhance predictive accuracy.

Thank you!