Drivesafe: Machine learning based Accident severity prediction website

Sneha Roy

24/05/2025

PROBLEM STATEMENT:

Develop a Machine Learning based Accident Severity Prediction System to help analyze and predict the severity of road accidents more accurately. By using an ML model , the system is designed to improve prediction accuracy while overcoming common challenges like handling of features, data imbalance and over fitting. To make it available and user- friendly, the model will be deployed as a Flask-based web application which will allow users to enter accident-related details and receive instant severity predictions. This can be especially useful for government authorities, emergency responders, and policymakers, helping them make informed decisions to improve road safety, allocate resources efficiently, and reduce the impact of accidents.

PRESENT MARKET OVERVIEW:

Growing Need for Road Safety Solutions

The global transportation sector is experiencing rapid growth, leading to an increase in the number of vehicles and traffic density. Consequently, road accidents have become a significant public safety concern. According to the World Health Organization (WHO), approximately 1.3 million people die each year as a result of road traffic crashes, with millions more suffering non-fatal injuries. The majority of these accidents are preventable, and severity prediction tools can help reduce fatalities by enabling faster and more efficient emergency response.

Rise of AI and ML in Transportation

With the advent of artificial intelligence (AI) and machine learning (ML), traffic and accident management has shifted from traditional reactive systems to predictive and preventive frameworks. Governments, insurance companies, smart city planners, and automotive industries are increasingly adopting ML-based solutions for:

- Predictive accident analytics
- Driver behavior monitoring
- Risk-based insurance pricing
- Smart traffic control and route planning

Market Demand for Predictive Safety Models

The road safety analytics market is expected to grow significantly. Key factors driving this demand include:

- Smart city initiatives that rely on data-driven traffic and safety insights
- Insurance tech (InsurTech) companies using ML models to assess claim risks and premiums
- Fleet management systems looking to monitor and minimize accident risks

PRODUCT INTRODUCTION:

The Accident Severity Prediction Web Application is a user-friendly, machine learning-powered tool designed to predict the severity level of a road accident based on various contributing factors. Built using a lightweight logistic regression model, this application offers an accessible interface where users can input accident-related data and receive real-time predictions of accident severity.

The primary objective of this product is to assist authorities, emergency services, traffic planners, and insurance analysts in understanding and responding to accidents more efficiently. By providing an early prediction of severity, the system can help:

- Prioritize emergency response based on predicted injury levels
- Assist traffic analysts in identifying high-risk patterns
- Aid insurance companies in automated risk assessment
- Support smart city frameworks with real-time safety analytics

Business need assessment

- Subscription-Based Model – Monthly or annual subscriptions for insurance companies and fleet operators.
- B2B Partnerships – Collaborate with traffic departments, insurance providers, and vehicle manufacturers.

Competitive Advantage:

- Hybrid ML Approach – By using ensemble model we can ensures more accurate predictions, reducing overfitting and improving feature handling.
- User-friendly Web Interface – Allows non-technical users (drivers, policymakers) to enter data and get predictions easily.
- Real-time Predictive Insights – Can be integrated with GPS systems, vehicle telematics, and smart city applications.

Target Audience

The Accident Severity Prediction Web Application is designed to serve a wide range of stakeholders across public, private, and research sectors. Its primary value lies in enabling data-driven decisions that improve road safety and operational efficiency.

1. Traffic and Emergency Response Authorities

- **Use Case:** Quickly assess accident severity to prioritize emergency services (ambulance, police, fire).
- **Benefit:** Helps allocate limited resources efficiently and save lives through faster interventions.

2. Government Transportation Departments

- **Use Case:** Analyze patterns of severe accidents to identify high-risk zones and faulty infrastructure.
- **Benefit:** Supports policy-making, infrastructure planning, and road safety improvement initiatives.

3. Insurance Companies

- **Use Case:** Use severity predictions to assess claim legitimacy and risk profiling.
- **Benefit:** Enhances fraud detection, accelerates claim processing, and enables usage-based insurance pricing.

4. Smart City and Urban Planners

- **Use Case:** Integrate with real-time traffic management systems for intelligent routing and risk alerts.
- **Benefit:** Supports the development of safer and more efficient urban transportation networks.

5. Fleet and Logistics Companies

- **Use Case:** Monitor accident risks across delivery and transport vehicles.
- **Benefit:** Reduces operational disruptions and supports driver training programs based on historical patterns.

6. Academic and Research Institutions

- **Use Case:** Use the model and platform as a base for further study in transportation engineering, AI in safety, or public policy.

- **Benefit:** Offers a practical, deployable use case for educational purposes or advanced research.

7. General Public (Future Scope)

- **Use Case:** In a future mobile version, commuters could input conditions and receive real-time risk assessments.
- **Benefit:** Promotes awareness and safer driving decisions among individual drivers.

Machine Learning Model Development

1. Dataset Collection and Preprocessing

The dataset used includes records of road accidents with attributes such as:

- **Weather conditions**
- **Road conditions**
- **Driver age**
- **Vehicle movement**
- **Accident type**
- **Target variable:** Accident severity (e.g., Minor, Serious, Fatal)

Preprocessing Steps:

- **Handling missing values:** Removed or imputed where necessary
- **Categorical encoding:** Applied pd.get_dummies() to convert categorical features into binary format
- **Train-Test Split:** The dataset was split into training and testing sets using an 80:20 ratio

2. Model Selection

For simplicity, interpretability, and fast computation, the **Logistic Regression** model was chosen.

- Works well for classification tasks
- Easy to implement and interpret
- Performs efficiently on small-to-medium sized datasets
- Suitable when input features are mostly categorical (after encoding)

3. Model Training

The logistic regression model was trained using the scikit-learn library.

```
from sklearn.linear_model import LogisticRegression
```

```
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)
```

- max_iter=1000 ensures convergence
- Model learns to predict the probability of each severity level based on training data patterns

4. Model Evaluation

The model's performance was evaluated using standard metrics:

```
from sklearn.metrics import accuracy_score
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
```

- **Accuracy Score:** Indicates how often the model predicts the correct severity class
- Depending on data balance, further metrics like F1-score**,** confusion matrix**,** or ROC **AUC** can be considered

5. Model Serialization

The trained model and its feature structure were saved using Python's pickle module:

```
import pickle
with open("model/accident_model.pkl", "wb") as f:
    pickle.dump((model, X.columns), f)
```

This enables the model to be reused in a Flask web application without retraining.

6. Integration with Web App

Once serialized, the model is loaded by the Flask backend and used for real-time prediction:

```
with open('model/accident_model.pkl', 'rb') as f:
    model, model_columns = pickle.load(f)
```

The input from users is converted to the same format as the training data to ensure compatibility with the model.

Website details

Main Components of the Website

1. Input Form Page

A form containing dropdown menus for all 14 features, such as:

o Age Band of Driver

o Sex of Driver

o Driving Experience

o Road & Weather Conditions

o Vehicle Movement

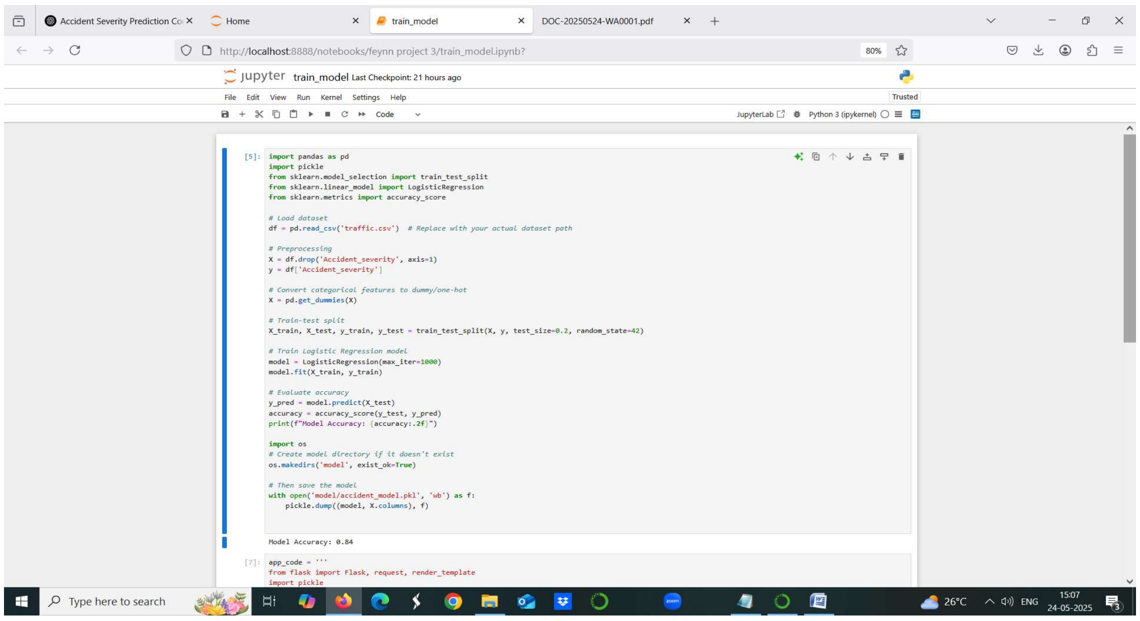
o Cause of Accident, etc.

 Submit button labeled “Predict”.

2. Prediction Output Section

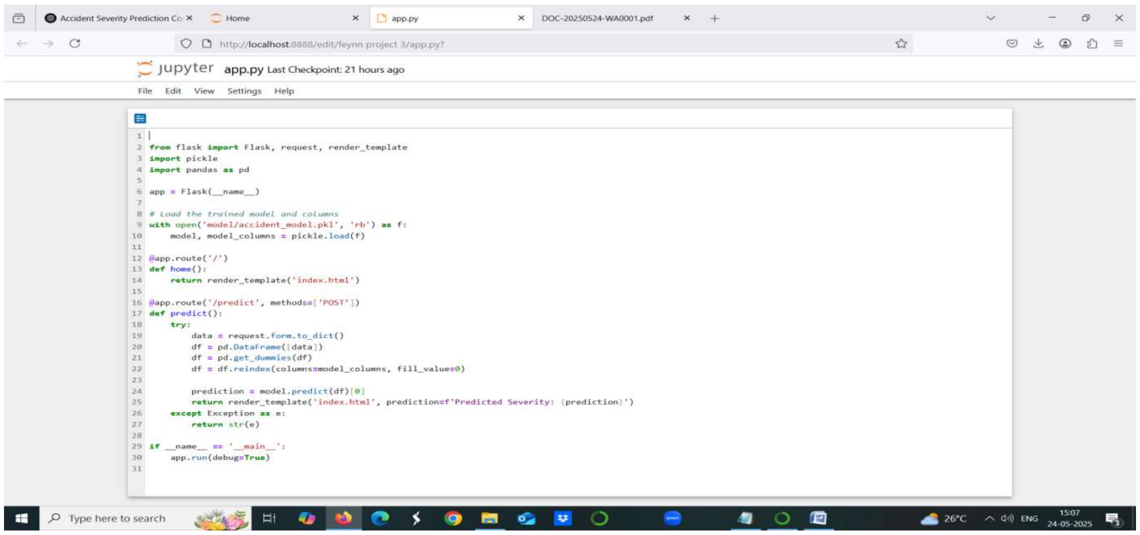
Displays the predicted severity (e.g., Slight, Serious, Fatal) after the form is submitted

CODE:



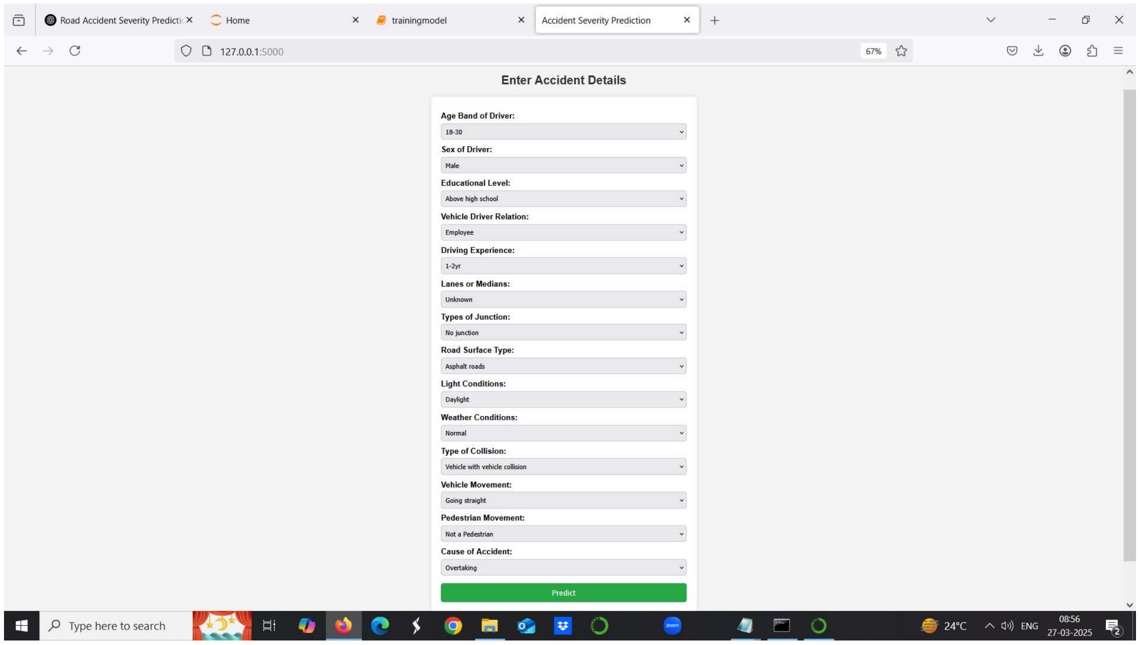
```
[5]: import pandas as pd
import pickle
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

# Load dataset
df = pd.read_csv('traffic.csv')  # Replace with your actual dataset path

# Preprocessing
X = df.drop('Accident_severity', axis=1)
y = df['Accident_severity']

# Convert categorical features to dummy/one-hot
X = pd.get_dummies(X)

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

# Train Logistic Regression model
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)

# Evaluate accuracy
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Model Accuracy: {accuracy:.2f}")

import os
# Create model directory if it doesn't exist
os.makedirs('model', exist_ok=True)

# Then save the model
with open('model/accident_model.pkl', 'wb') as f:
    pickle.dump((model, X.columns), f)

Model Accuracy: 0.84

[7]: app_code = '''
from flask import Flask, request, render_template
import pickle
```

FIGURE :ML MODEL CODE



```python

from flask import Flask, request, render_template
import pickle
import pandas as pd

app = Flask(__name__)

# Load the trained model and columns
with open('model/accident_model.pkl', 'rb') as f:
    model, model_columns = pickle.load(f)

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/predict', methods=['POST'])
def predict():
    try:
        data = request.form.to_dict()
        df = pd.DataFrame([data])
        df = pd.get_dummies(df)
        df = df.reindex(columns=model_columns, fill_value=0)

        prediction = model.predict(df)[0]
        return render_template('index.html', prediction=f'Predicted Severity: {prediction}')
    except Exception as e:
        return str(e)

if __name__ == '__main__':
    app.run(debug=True)

```

FIGURE:API CODE



Enter Accident Details

Age Band of Driver: 18-30

Sex of Driver: Male

Educational Level: Above high school

Vehicle Driver Relation: Employee

Driving Experience: 1-2yr

Lanes or Medians: Unknown

Types of Junction: No junction

Road Surface Type: Asphalt roads

Light Conditions: Daylight

Weather Conditions: Normal

Type of Collision: Vehicle with vehicle collision

Vehicle Movement: Going straight

Pedestrian Movement: Not a Pedestrian

Cause of Accident: Overtaking

FIGURE:WEBSITE

Product prototype

The Accident Severity Prediction Web Application prototype demonstrates how machine learning can be integrated into a user-facing system to deliver real-time accident risk analysis. This prototype includes a functioning frontend interface, a backend model processing unit, and a lightweight ML model that performs on-the-fly severity classification based on user inputs.

1. System Components

frontend (User Interface)

- **Technology Used:** HTML, CSS (can be extended with Bootstrap)
- **Purpose:** Allows users to input accident-related information
- **Interface Fields:**
  - Weather condition (e.g., Clear, Rainy, Foggy)
  - Road condition (e.g., Wet, Dry)
  - Driver age group (e.g., 18–25, 26–35)
  - Type of vehicle movement (e.g., Turning, Straight)
  - Any other relevant features from the dataset

Backend (Server Logic)

- **Technology Used:** Flask (Python micro web framework)
- **Responsibilities:**
  - Accept form data via HTTP POST
  - Preprocess input to match training format
  - Load the trained ML model
  - Predict accident severity
  - Send result back to the frontend

ML Model

- **Type:** Logistic Regression
- **Trained On:** Cleaned and preprocessed accident dataset
- **Output:** Accident severity label (e.g., Minor, Serious, Fatal)

Business model

B2B Model (Business-to-Business)

Monetization Strategies:

- API Licensing for Businesses – Charge organizations for API access to integrate predictions into their systems.
- Enterprise Subscription – Monthly/yearly subscriptions for real-time accident risk analysis.
- Government Contracts – Partner with traffic authorities for road safety analytics & policy planning.
- Insurance Risk Assessment Tool – Sell risk prediction reports to insurance companies for premium pricing & claim evaluation.

B2C Model (Business-to-Consumer)

Monetization Strategies:

- Freemium Model – Basic features can be free whereas advanced insights & premium reports require subscription.
- In-App Purchases – We can have a personalized accident risk analysis based on driver behavior.
- Ad-Based Revenue – Partner with car safety brands, insurance firms, and automotive services to display relevant ads.
- Premium Alerts Service – we can provide real-time SMS/email alerts for accident-prone routes as a paid feature

GITHUB LINK:

https://github.com/snrou/feynn-accident-severity.git