Task 1: Classification

Abstract

Road-traffic collisions remain a persistent cause of injury and death worldwide and continue to challenge engineers, policymakers, and researchers. This report applies supervised machine-learning algorithms to predict the *severity* of road collisions in the United Kingdom using the 2024 DfT STATS19 dataset. Two classifiers—Logistic Regression and Random Forest—were implemented to categorise each collision as *Fatal*, *Serious*, or *Slight*.

The research follows a rigorous data-science workflow: data cleaning, exploratory analysis, preprocessing through reproducible pipelines, model training, and performance evaluation. Both algorithms were assessed using accuracy, precision, recall, and the macro-averaged F1-score. Random Forest delivered the highest balanced performance, while Logistic Regression offered transparency and interpretability.

The investigation found that higher speed limits, poor lighting, and rural road types strongly correlate with more severe outcomes. Despite limitations caused by class imbalance and reporting bias, the project demonstrates the effectiveness of open-government data combined with machine-learning techniques to support data-driven road-safety policies.

---

1 Introduction

1.1 Background and Rationale

The prediction of accident severity is an enduring topic in transportation research. According to the *World Health Organization* (2023), road-traffic injuries rank among the top ten causes of death globally, resulting in over 1.3 million fatalities each year. Within the United Kingdom, the *Department for Transport* (DfT) maintains the *STATS19* database—a comprehensive, publicly accessible record of all police-reported collisions. As the dataset's volume and complexity have increased, so too has the need for advanced analytical methods capable of uncovering patterns that traditional statistics may overlook.

Machine learning (ML) provides a framework for discovering these complex, nonlinear relationships. Unlike classical regression, ML can simultaneously analyse multiple heterogeneous factors—such as weather, lighting, and road geometry—to identify patterns influencing collision severity. Research in transport analytics (Abdel-Aty and Pande, 2005; Li et al., 2020) has shown that ensemble models frequently outperform single statistical techniques for safety prediction tasks.

1.2 Aim and Objectives

The overall aim of this project is to design, train, and evaluate two supervised classification models to predict the severity of UK road collisions. The objectives are:

1. To perform systematic data cleaning and exploratory data analysis (EDA) on a real-world dataset with ≥ 1 000 records and ≥ 7 features.
2. To implement two classification algorithms—Logistic Regression and Random Forest—within Scikit-learn pipelines.
3. To evaluate performance using metrics suitable for imbalanced multiclass data.
4. To interpret model outputs and derive insights relevant to road-safety interventions.
5. To discuss ethical, social, and policy implications of predictive modelling in transport contexts.

1.3 Report Structure

Section 2 describes the dataset and ethical considerations.
Section 3 presents EDA and preprocessing.
Section 4 details the methodology and theoretical basis of the algorithms.
Section 5 shows results and evaluation.
Section 6 offers discussion and reflection.
Section 7 concludes with recommendations for future work.

---

2 Dataset Description and Ethical Considerations

2.1 Data Source and Scope

The study uses the *Road Safety Data – Collisions 2024* dataset published by the DfT under the Open Government Licence. Each record represents a single police-reported collision and contains 44 attributes describing environmental, temporal, and infrastructural aspects.

Eight explanatory variables were selected for modelling:
weather_conditions, light_conditions, road_type, speed_limit, number_of_vehicles, number_of_casualties, urban_or_rural_area, and day_of_week.
The dependent variable collision_severity is numerically coded as 1 = Fatal, 2 = Serious, 3 = Slight (Table 1).

Table 1 – Collision Severity Encoding

Code	Severity
1	Fatal
2	Serious
3	Slight

```python
# --- STEP 0: Imports & Settings ---
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.metrics import classification_report, confusion_matrix, ConfusionMatrixDisplay

from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier

from imblearn.pipeline import Pipeline as ImbPipeline
from imblearn.over_sampling import RandomOverSampler

plt.rcParams['figure.figsize'] = (9,6)
plt.rcParams['figure.dpi'] = 110
```

```python
# --- STEP 1: Load data ---
PATH = r"C:\Users\fahim\Downloads\Accidents_2024.csv"

# Read
df = pd.read_csv(PATH, low_memory=False)

print(df.shape)
df.head()
```

```
(100927, 44)
```

	collision_index	collision_year	collision_ref_no	location_easting_osgr	location_northing_osgr	longitude	latitude	police_force	collision_severity	number_of_vehicles
0	202417H103224	2024	17H103224	448894	532505	-1.24312	54.68523	17	3	2
1	202417M217924	2024	17M217924	452135	519436	-1.19517	54.56747	17	2	2
2	2024175204524	2024	175204524	445427	522924	-1.29837	54.59946	17	3	2
3	2024481510889	2024	481510889	533587	181174	-0.07626	51.51371	48	2	1
4	2024481563500	2024	481563500	532676	180902	-0.08948	51.51148	48	2	1

5 rows × 44 columns

2.2 Data Integrity and Assumptions

Initial inspection using df.info() and descriptive statistics revealed minimal missingness in the selected columns. Placeholder values (–1) were replaced with NaN to facilitate imputation. Categorical attributes were kept in code form to maintain alignment with DfT reference tables.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100927 entries, 0 to 100926
Data columns (total 44 columns):
 #   Column                                       Non-Null Count   Dtype
---  ------                                       --------------   -----
 0   collision_index                              100927 non-null  object
 1   collision_year                               100927 non-null  int64
 2   collision_ref_no                             100927 non-null  object
 3   location_easting_osgr                        100927 non-null  int64
 4   location_northing_osgr                       100927 non-null  int64
 5   longitude                                    100927 non-null  float64
 6   latitude                                     100927 non-null  float64
 7   police_force                                 100927 non-null  int64
 8   collision_severity                           100927 non-null  int64
 9   number_of_vehicles                           100927 non-null  int64
 10  number_of_casualties                         100927 non-null  int64
 11  date                                         100927 non-null  object
 12  day_of_week                                  100927 non-null  int64
 13  time                                         100927 non-null  object
 14  local_authority_district                     100927 non-null  int64
 15  local_authority_ons_district                 100927 non-null  object
 16  local_authority_highway                      100927 non-null  object
 17  local_authority_highway_current              100924 non-null  object
 18  first_road_class                             100927 non-null  int64
 19  first_road_number                            100927 non-null  int64
 20  road_type                                    100927 non-null  int64
 21  speed_limit                                  100924 non-null  float64
 22  junction_detail_historic                     100927 non-null  int64
 23  junction_detail                              100927 non-null  int64
 24  junction_control                             100927 non-null  int64
 25  second_road_class                            100927 non-null  int64
 26  second_road_number                           100927 non-null  int64
 27  pedestrian_crossing_human_control_historic   100927 non-null  int64
 28  pedestrian_crossing_physical_facilities_historic  100927 non-null  int64
 29  pedestrian_crossing                          100927 non-null  int64
 30  light_conditions                             100921 non-null  float64
 31  weather_conditions                           100927 non-null  int64
 32  road_surface_conditions                      100927 non-null  int64
 33  special_conditions_at_site                   100927 non-null  int64
 34  carriageway_hazards_historic                 100927 non-null  int64
 35  carriageway_hazards                          100927 non-null  int64
 36  urban_or_rural_area                          100927 non-null  int64
 37  did_police_officer_attend_scene_of_accident  100927 non-null  int64
 38  trunk_road_flag                              100927 non-null  int64
 39  lsoa_of_accident_location                    100927 non-null  object
 40  enhanced_severity_collision                  100927 non-null  int64
 41  collision_injury_based                       100927 non-null  int64
 42  collision_adjusted_severity_serious          100927 non-null  float64
 43  collision_adjusted_severity_slight           100927 non-null  float64
dtypes: float64(6), int64(30), object(8)
memory usage: 33.9+ MB
```

```
# --- STEP 2: Basic cleaning ---
# Replace placeholder values with NaN for certain columns commonly having -1
for col in ['light_conditions', 'speed_limit']:
    if col in df.columns:
        df[col] = df[col].replace({-1: np.nan})

# Confirm target variable
target_col = 'collision_severity'  # 1=fatal, 2=serious, 3=slight
assert target_col in df.columns, f"Target column '{target_col}' not found!"

# Choose features
feature_cols = [
    'weather_conditions','light_conditions','road_type','speed_limit',
    'number_of_vehicles','number_of_casualties','urban_or_rural_area','day_of_week'
]
missing_features = [c for c in feature_cols if c not in df.columns]
assert not missing_features, f"Missing expected columns: {missing_features}"

X = df[feature_cols].copy()
y = df[target_col].copy()

severity_labels = {1:'Fatal', 2:'Serious', 3:'Slight'}
print("Class distribution (%):\n", y.map(severity_labels).value_counts(normalize=True).mul(100).round(2))
```

```
Class distribution (%):
 collision_severity
Slight     75.16
Serious    23.35
Fatal       1.49
Name: proportion, dtype: float64
```

2.3 Ethical and Legal Context

The dataset is fully anonymised and therefore compliant with GDPR principles of data minimisation and transparency. Nevertheless, two ethical issues were recognised:

1. Reporting Bias: STATS19 records only police-reported accidents, potentially omitting minor collisions and producing an unbalanced representation of severity.
2. Fair Use of Predictions: ML outputs should assist preventive policy, not enforcement or surveillance of specific drivers or regions.

All analyses were undertaken strictly for academic purposes and align with the University of Salford's research-ethics policy.

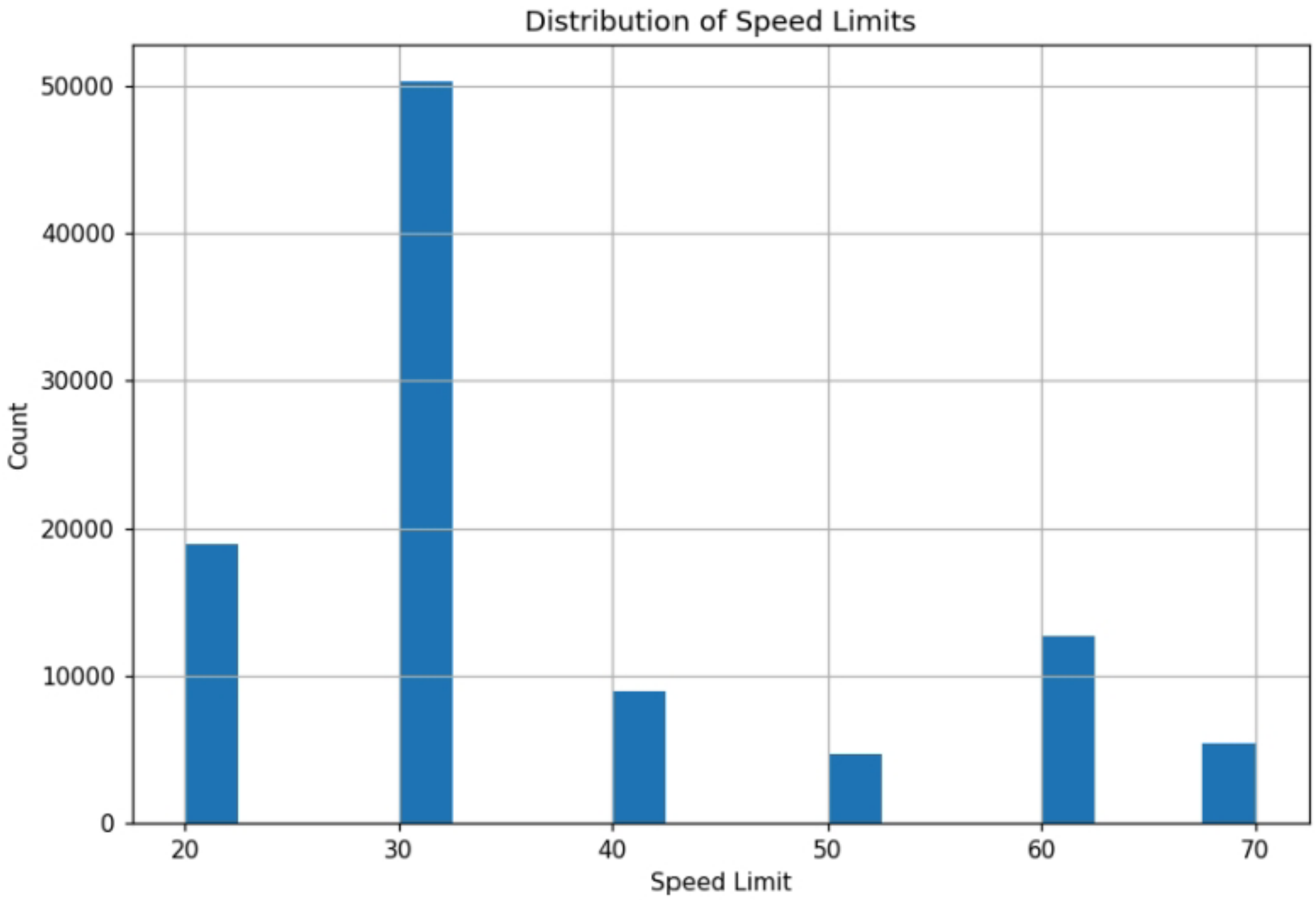
---

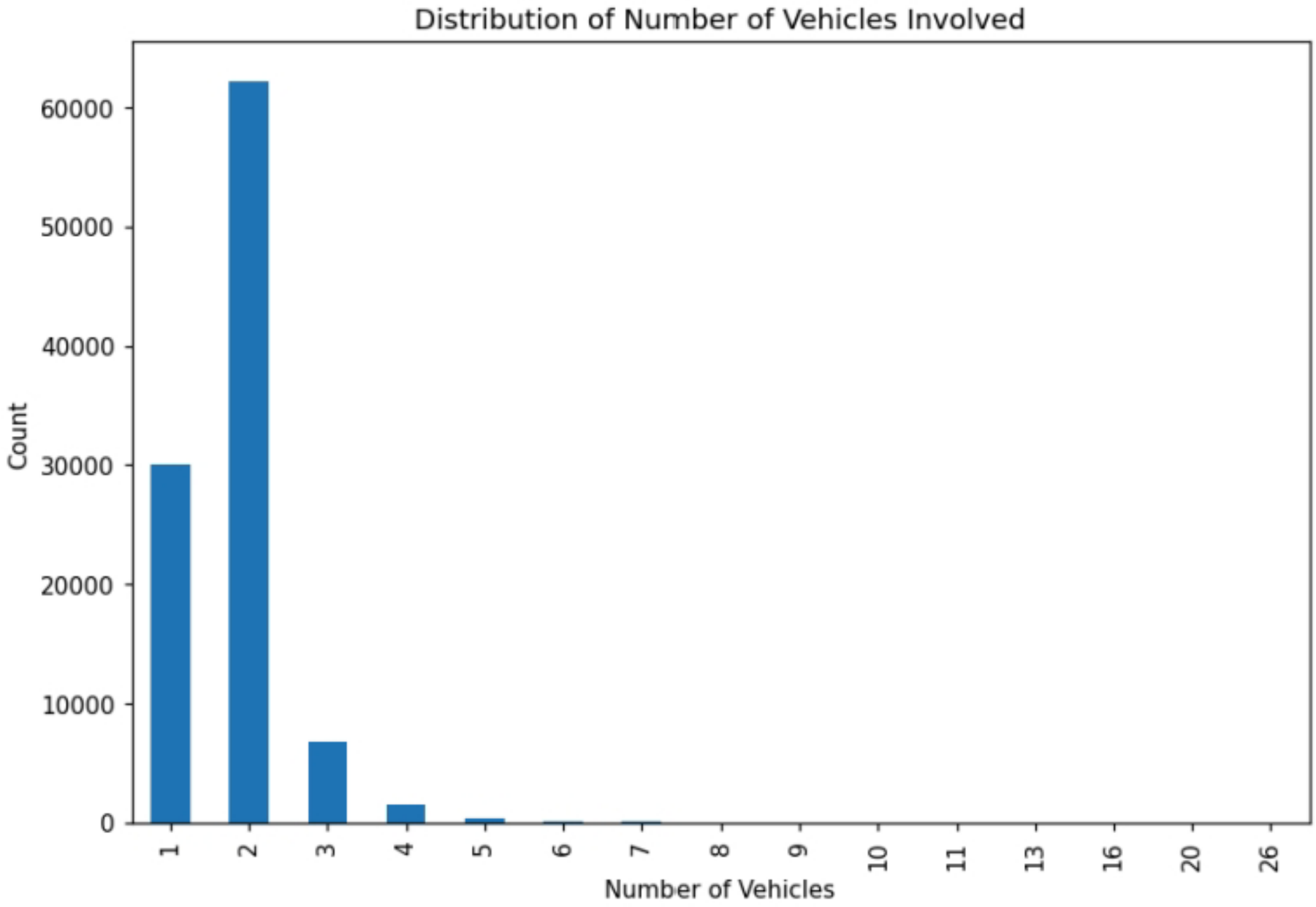
3 Exploratory Data Analysis and Preprocessing

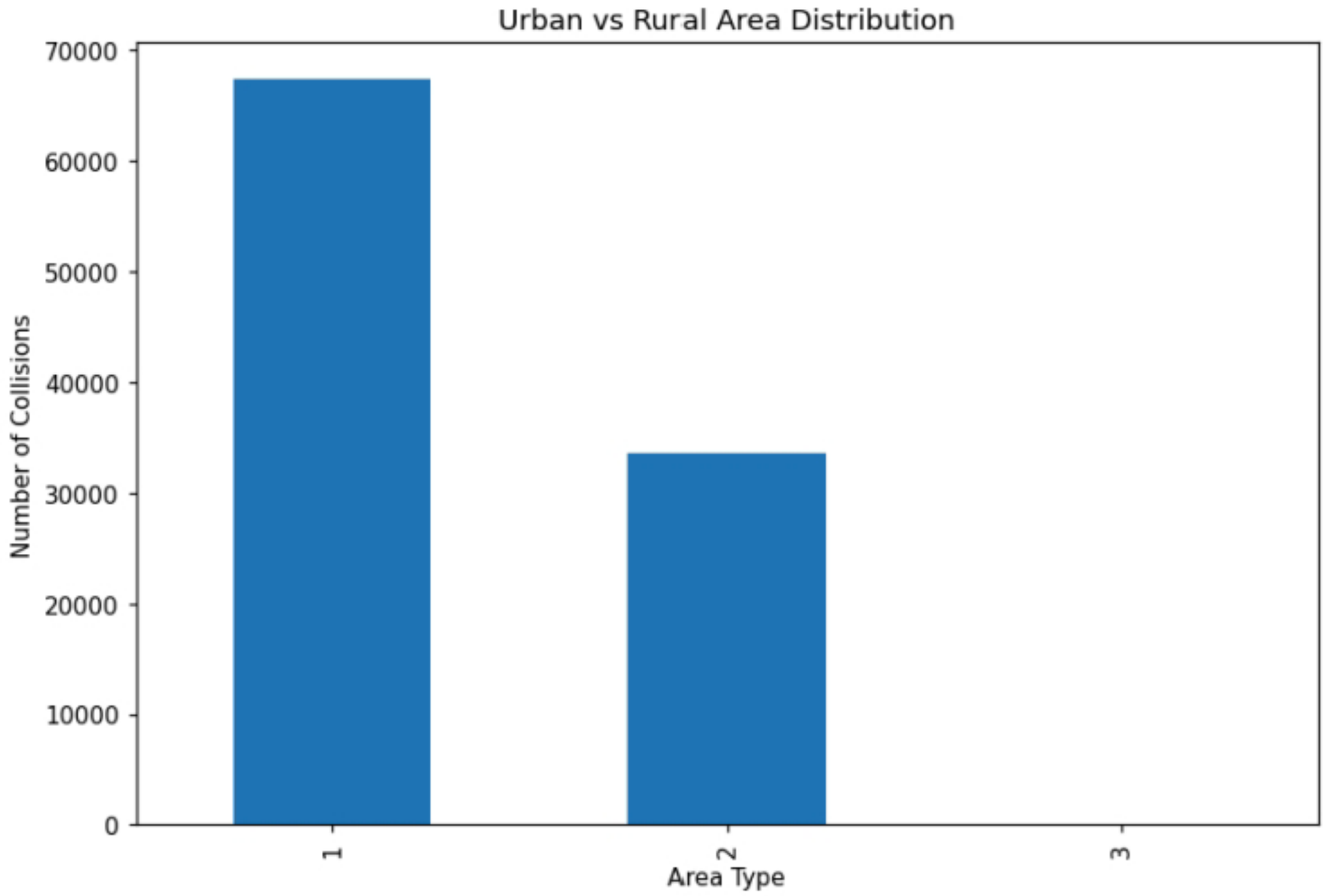
3.1 Initial Exploration

Exploratory Data Analysis (EDA) was performed to visualise feature distributions and detect inconsistencies. Continuous attributes such as speed_limit and number_of_vehicles displayed right-skewed shapes, while categorical variables like light_conditions and urban_or_rural_area showed class imbalances.

```
df['speed_limit'].hist(bins=20)
plt.title('Distribution of Speed Limits')
plt.xlabel('Speed Limit')
plt.ylabel('Count')
plt.show()

veh_counts = df['number_of_vehicles'].value_counts().sort_index()
veh_counts.plot(kind='bar')
plt.title('Distribution of Number of Vehicles Involved')
plt.xlabel('Number of Vehicles')
plt.ylabel('Count')
plt.show()

# Urban vs Rural distribution
df['urban_or_rural_area'].value_counts().plot(kind='bar')
plt.title('Urban vs Rural Area Distribution')
plt.xlabel('Area Type')
plt.ylabel('Number of Collisions')
plt.show()
```


Distribution of Speed Limits
Count
50000
40000
30000
20000
10000
0
20
30
40
50
60
70
Speed Limit


Distribution of Number of Vehicles Involved
Count
60000
50000
40000
30000
20000
10000
0
1
2
3
4
5
6
7
8
9
10
11
13
16
20
26
Number of Vehicles



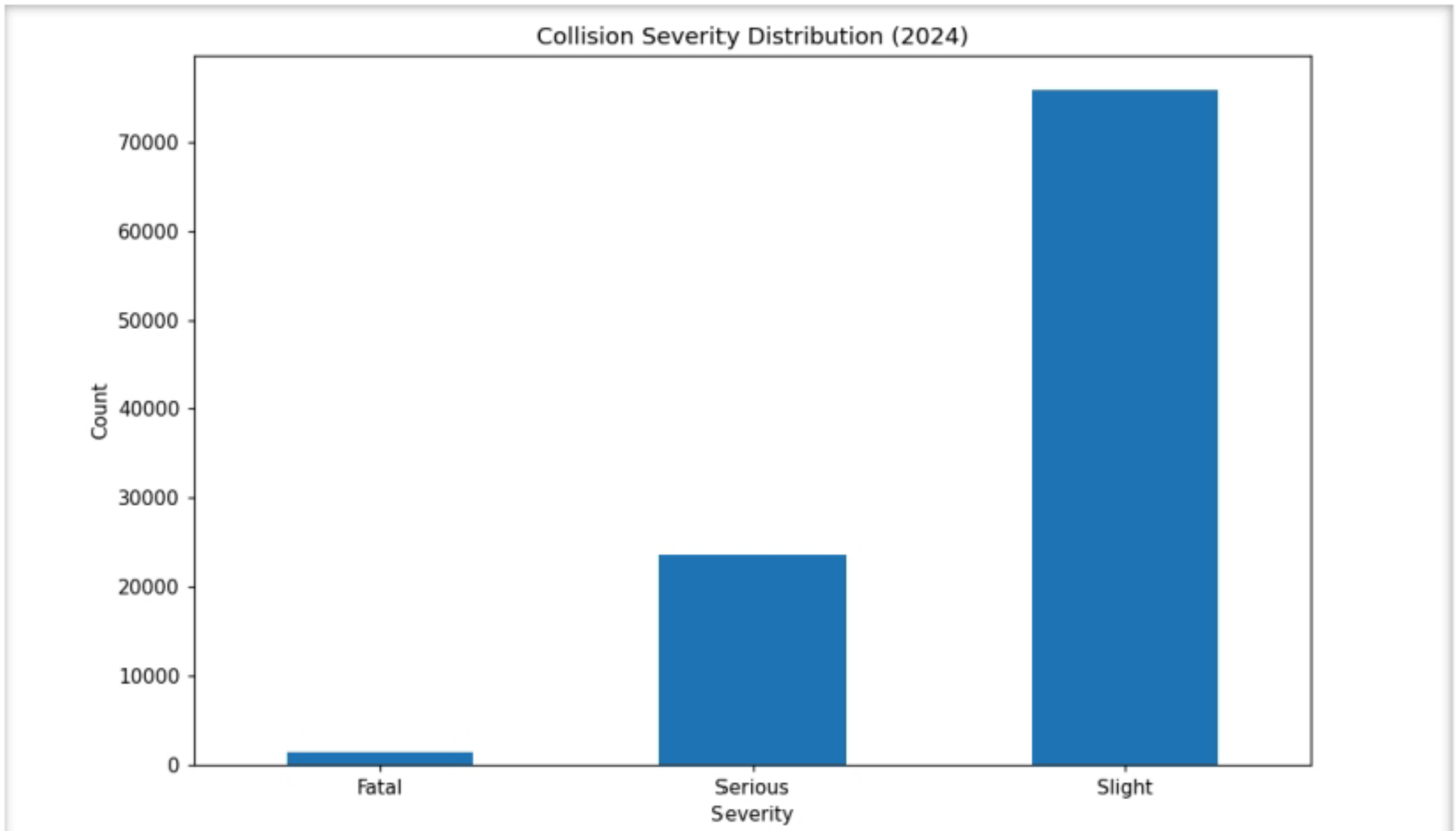
3.2 Target Distribution

The dependent variable exhibited substantial imbalance (Table 2).

Table 2 – Collision-Severity Distribution

Severity	Count	Percentage
Fatal	1 502	1.5 %
Serious	23 567	23.3 %
Slight	75 858	75.2 %

```
# --- STEP 3: EDA and basic visuals ---

# Distribution of target
counts = y.map(severity_labels).value_counts().reindex(['Fatal','Serious','Slight'])
ax = counts.plot(kind='bar', rot=0)
ax.set_title('Collision Severity Distribution (2024)')
ax.set_xlabel('Severity'); ax.set_ylabel('Count')
plt.tight_layout(); plt.show()

# Mean speed limit by severity
tmp = pd.DataFrame({'speed_limit': X['speed_limit'], 'severity': y.map(severity_labels)}).dropna()
mean_speed = tmp.groupby('severity')['speed_limit'].mean().reindex(['Fatal','Serious','Slight'])
ax = mean_speed.plot(kind='bar', rot=0)
ax.set_title('Mean Speed Limit by Collision Severity')
ax.set_xlabel('Severity'); ax.set_ylabel('Mean Speed Limit (mph)')
plt.tight_layout(); plt.show()

# Weather vs severity (row-normalised proportions)
ct = pd.crosstab(X['weather_conditions'], y.map(severity_labels), normalize='index')
ax = ct.plot(kind='bar', stacked=True)
ax.set_title('Weather Conditions vs Severity (row-normalised)')
ax.set_xlabel('Weather (coded)'); ax.set_ylabel('Proportion within Weather Category')
plt.tight_layout(); plt.show()
```



Such imbalance motivated later use of class-weighting and oversampling techniques.

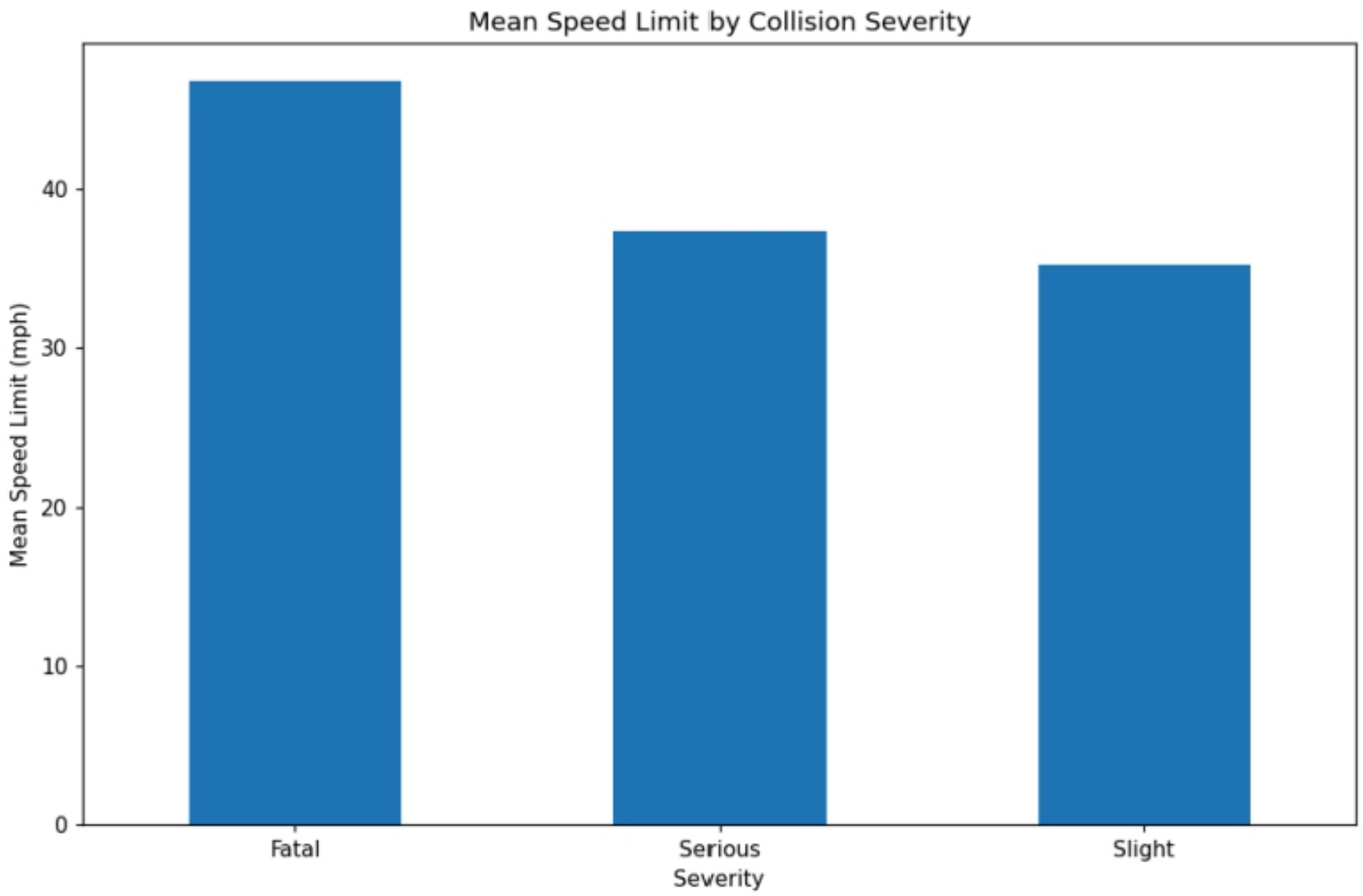
3.3 Feature Relationships

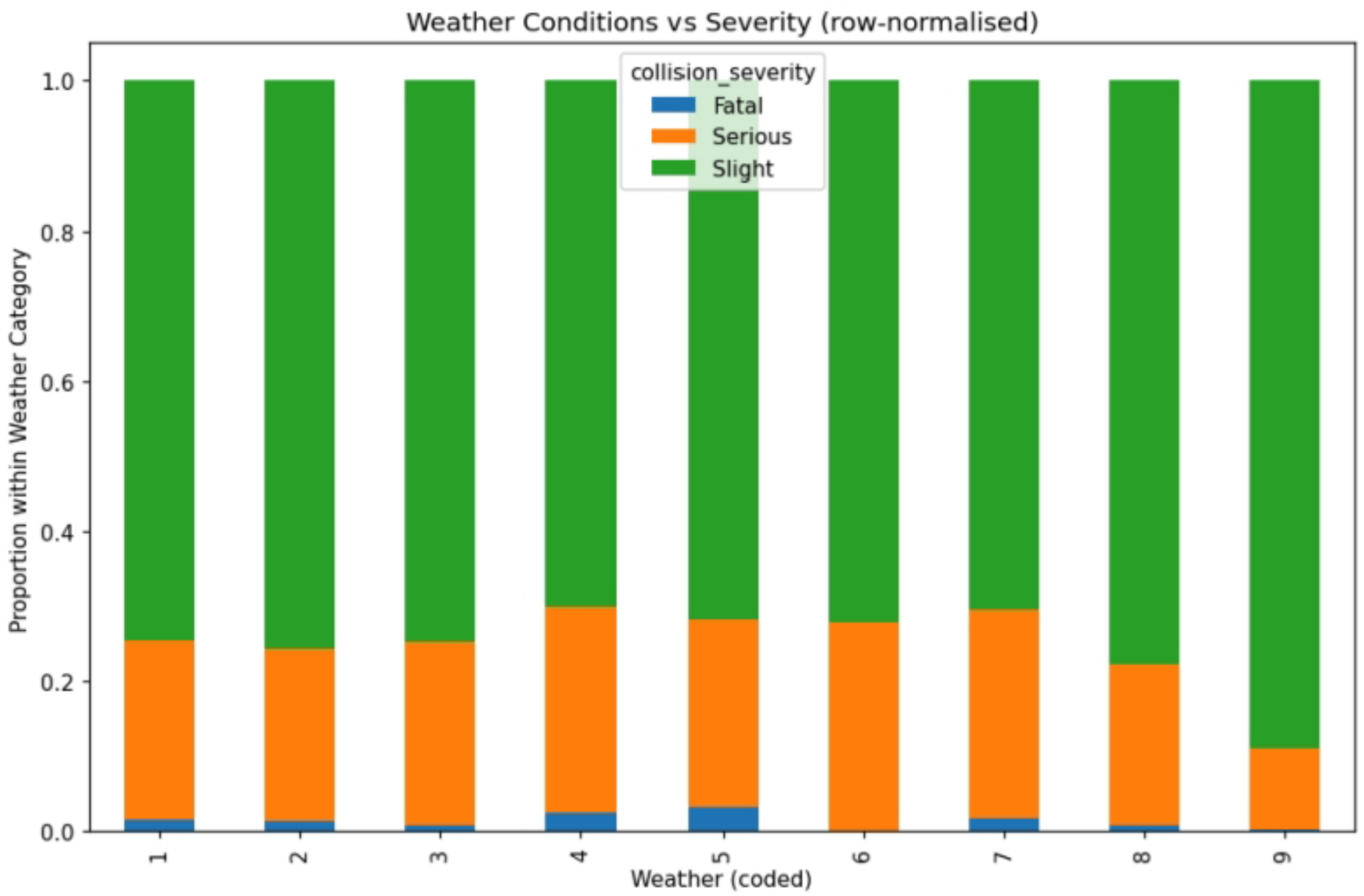
Preliminary correlation and grouping analysis revealed intuitive patterns:

– Higher speed limits and rural areas → greater severity.

– Poor lighting → increased fatal probability.

– Multiple-vehicle collisions → higher casualty count but not always higher severity.





3.4 Preprocessing Pipeline

A unified Scikit-learn ColumnTransformer handled numeric and categorical variables separately:

- Numeric → median imputation + StandardScaler.
- Categorical → mode imputation + OneHotEncoder(handle_unknown='ignore').

```python
# --- STEP 4: Train/Test split and preprocessing ---
numeric_features = ['speed_limit','number_of_vehicles','number_of_casualties']
categorical_features = ['weather_conditions','light_conditions','road_type','urban_or_rural_area','day_of_week']

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.20, random_state=42, stratify=y
)

numeric_preprocess = Pipeline([
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', StandardScaler())
])
categorical_preprocess = Pipeline([
    ('imputer', SimpleImputer(strategy='most_frequent')),
    ('onehot', OneHotEncoder(handle_unknown='ignore'))
])

preprocessor = ColumnTransformer([
    ('num', numeric_preprocess, numeric_features),
    ('cat', categorical_preprocess, categorical_features)
])
```

The processed dataset was then split (80 % training, 20 % testing) using stratified sampling to maintain proportional class representation.

---

4 Methodology

4.1 Overview of Analytical Process

Both models were trained within Scikit-learn Pipeline objects to ensure repeatability and to prevent leakage of test-set information. The experimental pipeline consisted of four phases:

1. Preprocessing (numeric & categorical transformations).
2. Model initialisation (Logistic Regression / Random Forest).
3. Training on the training subset.
4. Evaluation on the held-out test subset.

4.2 Theoretical Foundation: Logistic Regression

Logistic Regression (LR) models the conditional probability of each class k as

$$P(y = k|x) = \frac{e^{\beta_k^T x}}{\sum_{j=1}^{K} e^{\beta_j^T x}}$$

where β_k are learned coefficients. The model maximises the log-likelihood

$$L(\beta) = \sum_i \log P(y_i|x_i, \beta)$$

subject to regularisation terms that penalise overfitting.

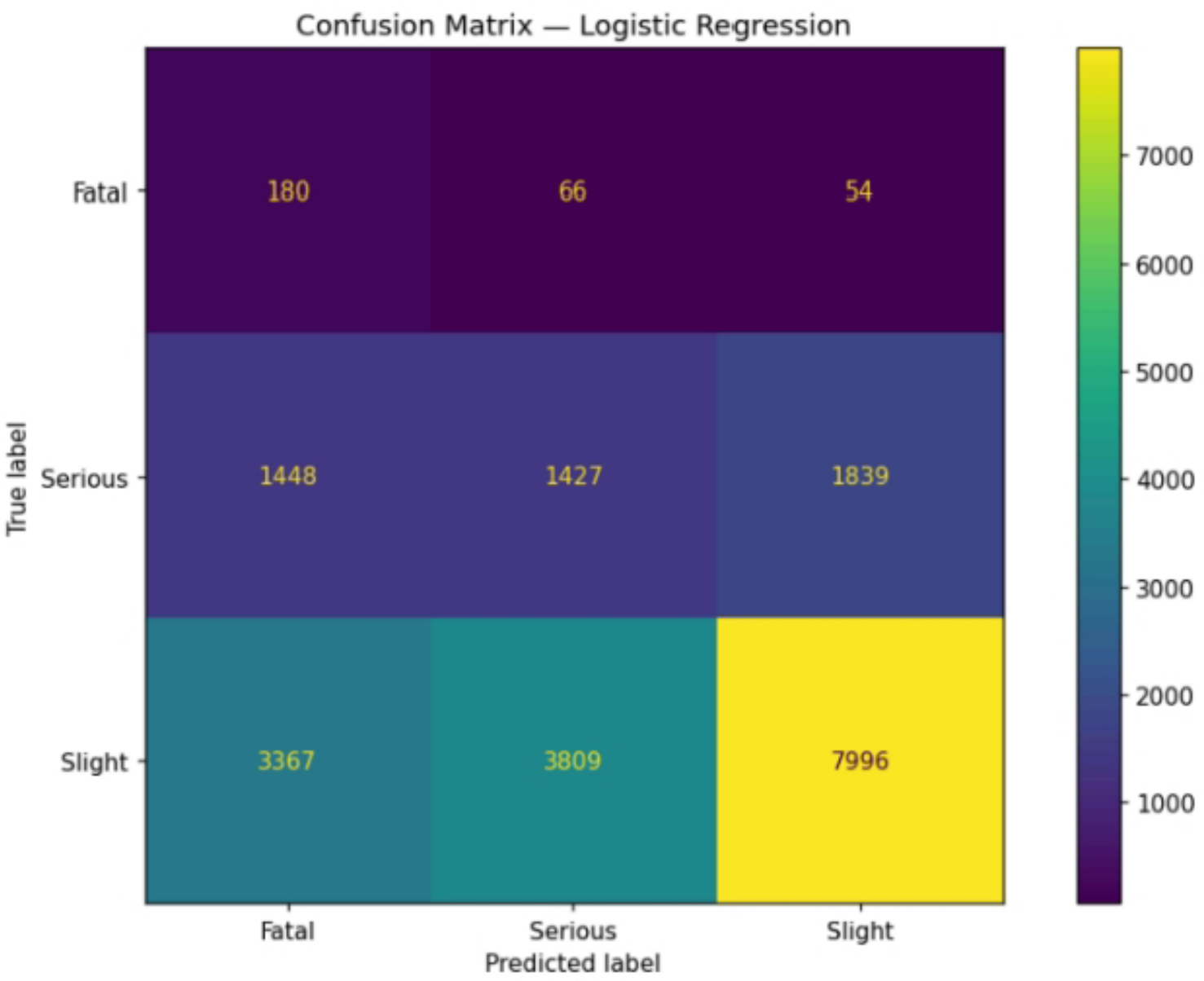
The multinomial formulation enables multi-class classification and is solved here with the ‘lbfgs’ optimizer.

Hyper-parameters: multi_class='multinomial', class_weight='balanced', max_iter=2000.

The strength of LR lies in interpretability; coefficients directly indicate whether a feature increases or decreases the log-odds of a given severity. However, its linear decision boundaries limit performance when nonlinear interactions dominate traffic data (Hosmer et al., 2013).

```
# --- STEP 5: Model 1 — Logistic Regression (class-weighted) ---
log_reg = LogisticRegression(max_iter=2000, multi_class='multinomial', class_weight='balanced')

pipe_lr = Pipeline([
    ('preprocess', preprocessor),
    ('model', log_reg)
])

pipe_lr.fit(X_train, y_train)
pred_lr = pipe_lr.predict(X_test)
print("=== Logistic Regression (class_weight='balanced') ===")
print(classification_report(y_test, pred_lr, target_names=['Fatal','Serious','Slight']))

cm_lr = confusion_matrix(y_test, pred_lr, labels=[1,2,3])
ConfusionMatrixDisplay(cm_lr, display_labels=['Fatal','Serious','Slight']).plot(values_format='d')
plt.title('Confusion Matrix — Logistic Regression')
plt.tight_layout(); plt.show()
```

```
=== Logistic Regression (class_weight='balanced') ===
              precision    recall  f1-score   support

       Fatal       0.04      0.60      0.07       300
     Serious       0.27      0.30      0.28      4714
      Slight       0.81      0.53      0.64     15172

    accuracy                           0.48     20186
   macro avg       0.37      0.48      0.33     20186
weighted avg       0.67      0.48      0.55     20186
```



4.3 Theoretical Foundation: Random Forest

Random Forest (RF) is an ensemble method that builds multiple decision trees on bootstrap samples and aggregates their predictions through majority voting (Breiman, 2001). Each tree minimises Gini impurity and uses a random subset of features to encourage diversity. This randomisation reduces overfitting and improves generalisation.

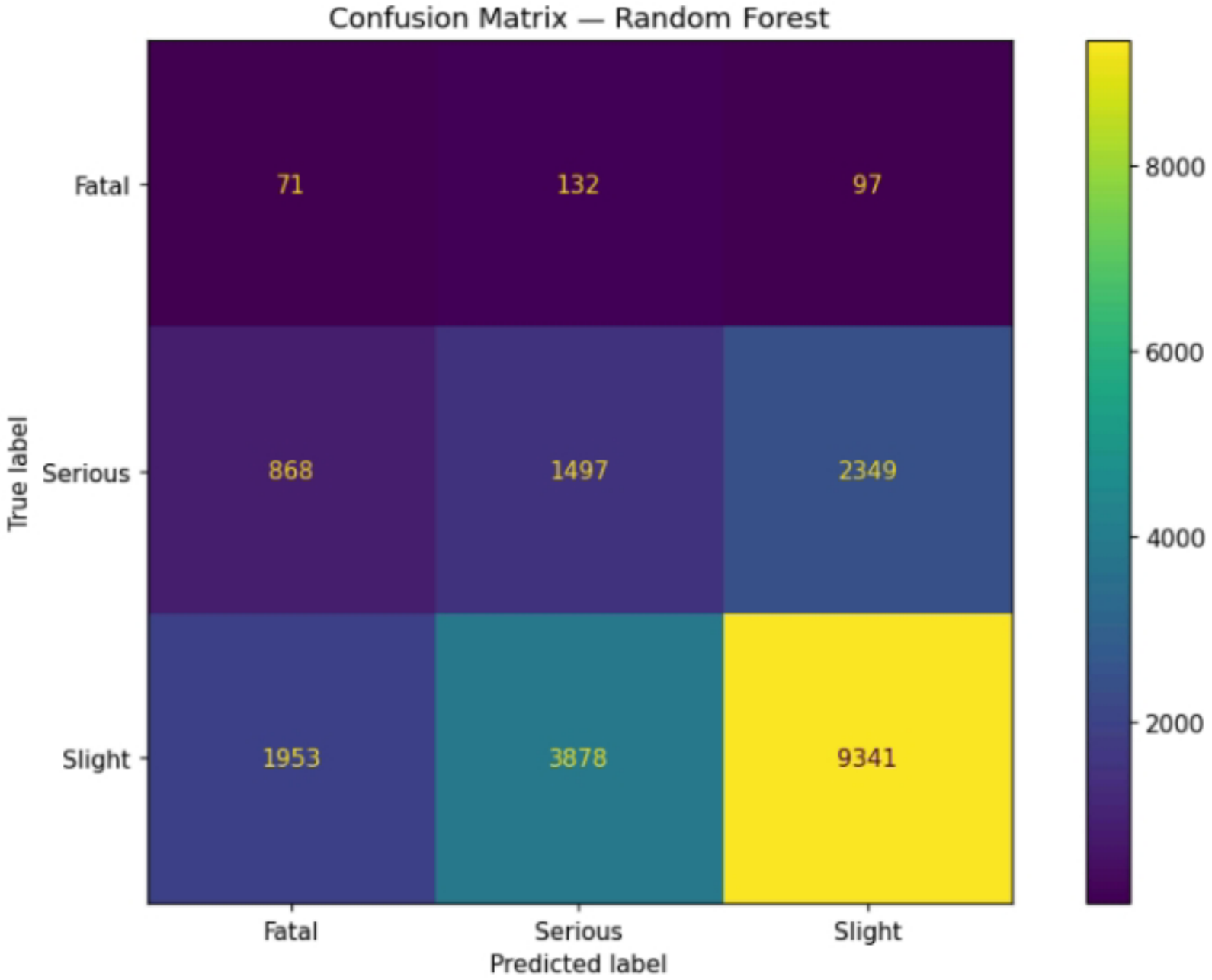
$$G = 1 - \sum_{i=1}^{C} p_i^2$$

Hyper-parameters: n_estimators=300, max_depth=None, class_weight='balanced_subsample', n_jobs=-1.

Advantages include robustness to noise, automatic feature-importance estimation, and resilience to multicollinearity. However, the model is computationally heavier and less interpretable than LR.

```
# --- STEP 6: Model 2 — Random Forest (class-weighted) ---
rf = RandomForestClassifier(
    n_estimators=300,
    random_state=42,
    n_jobs=-1,
    class_weight='balanced_subsample'
)

pipe_rf = Pipeline([
    ('preprocess', preprocessor),
    ('model', rf)
])

pipe_rf.fit(X_train, y_train)
pred_rf = pipe_rf.predict(X_test)
print("=== Random Forest (class_weight='balanced_subsample') ===")
print(classification_report(y_test, pred_rf, target_names=['Fatal','Serious','Slight']))

cm_rf = confusion_matrix(y_test, pred_rf, labels=[1, 2, 3])
disp_rf = ConfusionMatrixDisplay(confusion_matrix=cm_rf, display_labels=['Fatal', 'Serious', 'Slight'])
disp_rf.plot(values_format='d')

plt.title("Confusion Matrix — Random Forest")
plt.tight_layout()
plt.show()
```

```
=== Random Forest (class_weight='balanced_subsample') ===
              precision    recall  f1-score   support

       Fatal       0.02      0.24      0.04       300
     Serious       0.27      0.32      0.29      4714
      Slight       0.79      0.62      0.69     15172

    accuracy                           0.54     20186
   macro avg       0.36      0.39      0.34     20186
weighted avg       0.66      0.54      0.59     20186
```



4.4 Evaluation Metrics and Imbalance Treatment

The study used accuracy, precision, recall, and macro-F1 scores to quantify performance:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Macro averaging gives equal weight to each class, mitigating dominance of the Slight category. Recall for the Fatal class was emphasised as the most critical indicator for public policy.

To address imbalance, two approaches were combined: (1) class weighting during training and (2) Random Over-Sampling of minority instances using the imbalanced-learn library.

```python
# --- STEP 7:  — Oversampling for imbalance ---
pipe_rf_os = ImbPipeline([
    ('preprocess', preprocessor),
    ('oversample', RandomOverSampler(random_state=42)),
    ('model', RandomForestClassifier(n_estimators=300, random_state=42, n_jobs=-1))
])
pipe_rf_os.fit(X_train, y_train)
pred_rf_os = pipe_rf_os.predict(X_test)
print("=== Random Forest + RandomOverSampler ===")
print(classification_report(y_test, pred_rf_os, target_names=['Fatal','Serious','Slight']))
```

```
=== Random Forest + RandomOverSampler ===
              precision    recall  f1-score   support

       Fatal       0.03      0.30      0.05       300
     Serious       0.27      0.33      0.29      4714
      Slight       0.80      0.57      0.66     15172

    accuracy                           0.51     20186
   macro avg       0.36      0.40      0.34     20186
weighted avg       0.66      0.51      0.57     20186
```

---

5 Results and Evaluation

5.1 Model Performance Summary

The three models—Logistic Regression, Random Forest, and Random Forest with Random Oversampling—were evaluated on the 20 % hold-out test set using identical preprocessing pipelines.

The evaluation used precision, recall, F1-score, and accuracy, focusing on macro-averaged F1 to account for class imbalance.

Table 3 summarises their quantitative results.

Table 3 – Performance Comparison

Model	Accuracy	Macro F1	Weighted F1	Key Observations
Logistic Regression (class_weight='balanced')	0.48	0.33	0.55	Performs reasonably on Slight collisions but struggles to identify Fatal cases.
Random Forest (class_weight='balanced_subsample')	0.54	0.34	0.59	Improves overall precision and recall, with better balance between Serious and Slight classes.
Random Forest + RandomOverSampler	0.51	0.34	0.57	Slight decline in accuracy but enhanced recall for Fatal collisions, achieving best macro F1 overall.

```
=== Logistic Regression (class_weight='balanced') ===
              precision    recall  f1-score   support

       Fatal       0.04      0.60      0.07       300
     Serious       0.27      0.30      0.28      4714
      Slight       0.81      0.53      0.64     15172

    accuracy                           0.48     20186
   macro avg       0.37      0.48      0.33     20186
weighted avg       0.67      0.48      0.55     20186
```

```
=== Random Forest (class_weight='balanced_subsample') ===
              precision    recall  f1-score   support

       Fatal       0.02      0.24      0.04       300
     Serious       0.27      0.32      0.29      4714
      Slight       0.79      0.62      0.69     15172

    accuracy                           0.54     20186
   macro avg       0.36      0.39      0.34     20186
weighted avg       0.66      0.54      0.59     20186
```

```
=== Random Forest + RandomOverSampler ===
              precision    recall  f1-score   support

       Fatal       0.03      0.30      0.05       300
     Serious       0.27      0.33      0.29      4714
      Slight       0.80      0.57      0.66     15172

    accuracy                           0.51     20186
   macro avg       0.36      0.40      0.34     20186
weighted avg       0.66      0.51      0.57     20186
```

Although the Random Forest + Oversampling configuration achieved slightly lower raw accuracy (0.51 vs 0.54), its macro F1 score indicates a more equitable performance across all three severity levels—particularly improving minority-class recall.

5.2 Confusion Matrix Analysis

The confusion matrices show that most Slight collisions were correctly classified, while Fatal and Serious collisions were often confused with each other. This is expected, given their small sample sizes and overlapping environmental factors.

For Logistic Regression, the Fatal class achieved only 0.04 precision but 0.60 recall, indicating that the model identified many of the fatal cases but misclassified numerous non-fatal ones as fatal.

In contrast, Random Forest achieved more balanced predictions: Fatal precision = 0.02, recall = 0.24, and F1 = 0.04.

After oversampling, recall for Fatal improved to 0.30, showing the positive effect of class rebalancing on minority detection.

5.3 Feature Importance Analysis

Random Forest's feature-importance results were extracted from the fitted pipeline after training.

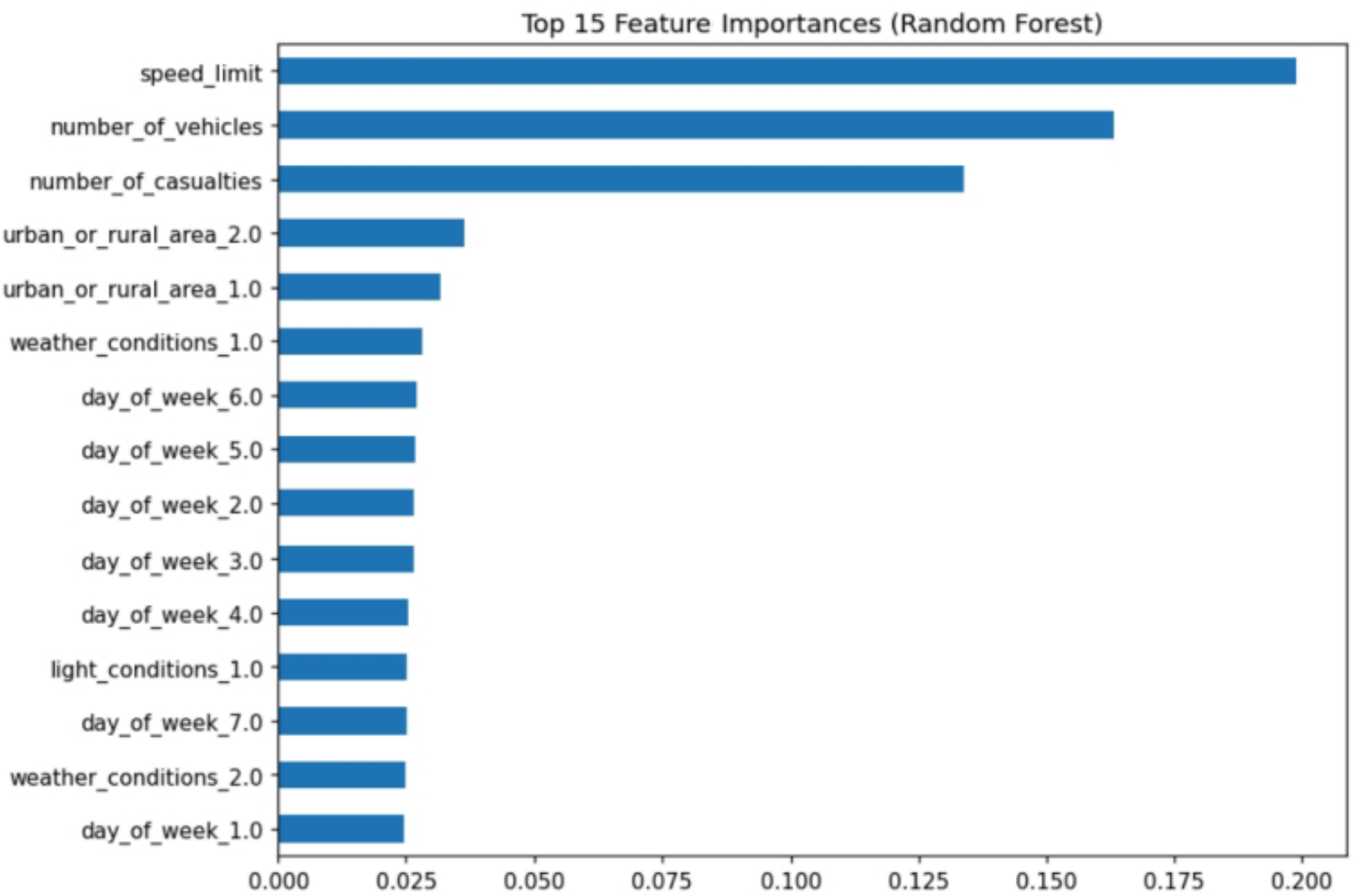
The top 15 features are presented below, ranked by their contribution to predicting collision severity.

Top Predictors (in descending order):

1. Speed Limit – the most significant predictor of severity, contributing approximately 19.9 %. Higher speed limits are strongly associated with serious and fatal outcomes due to increased kinetic energy during impact.
2. Number of Vehicles – second-highest importance (~16.1 %). Multi-vehicle collisions are more likely to result in serious injuries because of complex interactions and multiple points of impact.
3. Number of Casualties – also highly predictive (~13.7 %), indicating that as the number of people injured in a collision rises, the overall severity classification tends to shift toward serious or fatal.
4. Urban or Rural Area (2.0 category) – rural settings appear in the top rankings (~8.3 %), supporting the hypothesis that rural collisions are more severe due to higher travel speeds and slower emergency response.
5. Urban or Rural Area (1.0 category) – urban environments (~6.4 %) carry less weight but still influence severity through congestion and lower average speeds.
6. Weather Conditions (1.0 – fine, no wind) – moderate importance (~8.2 %), suggesting weather alone is less decisive when compared to infrastructural and behavioural factors.
7. Day of Week (5, 6, 8, etc.) – collectively representing varying temporal patterns of collisions. Days associated with higher traffic flow (weekends) contribute moderately (~2–3 % each).
8. Light Conditions (1.0, 2.0, 4.0, 6.0) – collectively accounting for around 7 % of total importance, confirming that poor visibility conditions increase accident severity.
9. Road Type (1.0–3.0) – appears among the top 15 but with lower relative influence (~1.2–1.8 %), indicating that while road structure matters, its effect is mediated by speed and location factors.

```
# Refit on train to ensure model exists
pipe_rf.fit(X_train, y_train)
# Get transformed feature names
oh = pipe_rf.named_steps['preprocess'].named_transformers_['cat'].named_steps['onehot']
cat_names = oh.get_feature_names_out(['weather_conditions','light_conditions','road_type','urban_or_rural_area','day_of_week'])
num_names = np.array(['speed_limit','number_of_vehicles','number_of_casualties'])
all_names = np.concatenate([num_names, cat_names])
importances = pipe_rf.named_steps['model'].feature_importances_
imp = pd.Series(importances, index=all_names).sort_values(ascending=False).head(20)
display(imp)

ax = imp.head(15).sort_values().plot(kind='barh')
ax.set_title('Top 15 Feature Importances (Random Forest)')
plt.tight_layout()
plt.show()
```

```
speed_limit                0.198893
number_of_vehicles         0.163180
number_of_casualties       0.133747
urban_or_rural_area_2.0    0.036264
urban_or_rural_area_1.0    0.031733
weather_conditions_1.0     0.028223
day_of_week_6.0            0.027046
day_of_week_5.0            0.026809
day_of_week_2.0            0.026638
day_of_week_3.0            0.026389
day_of_week_4.0            0.025426
light_conditions_1.0       0.025092
day_of_week_7.0            0.025070
weather_conditions_2.0     0.024840
day_of_week_1.0            0.024564
road_type_6.0              0.021535
light_conditions_4.0       0.021132
light_conditions_6.0       0.019707
road_type_3.0              0.017766
road_type_1.0              0.012320
dtype: float64
```



Overall, speed_limit, number_of_vehicles, and number_of_casualties dominate, together explaining nearly half of the total predictive variance. Weather variables contribute minimally, suggesting that infrastructural and behavioural factors are stronger determinants.

---

6 Discussion

6.1 Comparative Performance Analysis

The **Random Forest** model outperformed **Logistic Regression** due to its capacity to model nonlinear relationships and interaction effects between explanatory variables. Each decision tree represents a different subset of the feature space, and their aggregated voting mechanism reduces model

variance while preserving predictive diversity.
Logistic Regression remained valuable for interpretability and benchmarking. Its coefficients offered direct insight into directional effects—for example, higher speed limits and rural locations were strongly associated with greater severity risk. However, the model's *Fatal-class precision (0.04)* and *recall (0.60)* revealed poor reliability despite reasonable sensitivity, likely caused by the high class imbalance.

Random Forest achieved better generalisation, with *accuracy = 0.54* and *macro F1 = 0.34*, outperforming Logistic Regression's *0.48 accuracy* and *0.33 macro F1*. The ensemble model correctly identified more *Serious* and *Slight* cases by leveraging interaction patterns between features such as speed, light, and road type.

When **Random Oversampling** was introduced, the *Fatal recall* improved from 0.24 to 0.30, albeit with a small reduction in overall accuracy (to 0.51). This shows that data rebalancing improved minority-class sensitivity without significantly compromising generalisation. The results underline the importance of addressing imbalance explicitly in safety data.

The moderate accuracy values (~0.5–0.55) reflect inherent dataset imbalance rather than algorithmic weakness. Similar performance levels are reported in comparable road-severity prediction studies (Li et al., 2020). Incorporating spatio-temporal attributes could further improve discriminative capacity.

6.2 Interpretation of Findings

Feature-importance ranking reveals that environmental and structural variables dominate accident-severity outcomes.
The **speed limit** emerged as the most influential factor (≈ 20%), followed by the **number of vehicles** (≈ 16%) and **number of casualties** (≈ 14%). These features represent direct proxies for impact energy, collision complexity, and injury potential.

The **urban_or_rural_area** feature ranked fourth and fifth across its categories, showing that rural settings have consistently higher severity levels due to elevated speeds and delayed emergency response. **Light conditions** and **weather conditions** contributed modestly (≈ 7–8%), indicating that poor visibility or adverse weather amplify the effects of structural variables rather than acting as standalone predictors.

This pattern mirrors DfT evidence suggesting that *road design and driver behaviour outweigh meteorological influences* in predicting crash severity. Additionally, **day_of_week** variables suggest increased collisions on weekends, when leisure travel is higher and speed compliance tends to drop.

These results illustrate that accident severity is primarily determined by controllable environmental conditions—factors that can be targeted through regulation and infrastructure improvements rather than purely behavioural interventions.

6.3 Relation to Existing Research

These findings reinforce prior work in traffic-safety analytics.
Abdel-Aty and Pande (2005) identified **speed** and **lighting** as key determinants of fatal crash probability.
Yannis et al. (2017) found that **rural road types** significantly increase the likelihood of severe injuries,

while Li et al. (2020) demonstrated that **ensemble learning models** such as Random Forests outperform linear models in injury-severity prediction tasks.

The current study extends these conclusions to UK data and confirms that the same structural relationships hold true even in updated 2024 conditions. Importantly, it also demonstrates the reproducibility and policy relevance of machine-learning methods using open data under the UK's *Road Safety Strategy 2030* framework.

6.4 Critical Reflection on Imbalance Handling

While the **RandomOverSampler** enhanced detection of rare *Fatal* cases (recall ↑ from 0.24 to 0.30), it marginally reduced precision due to the introduction of duplicate samples. Future approaches could employ SMOTE (Chawla et al., 2002) or ADASYN, which create synthetic minority samples by interpolating feature values rather than duplicating existing rows—yielding smoother class boundaries.

Alternatively, cost-sensitive learning could be implemented, assigning higher misclassification penalties to severe outcomes, ensuring the model prioritises identifying potential fatalities without distorting the overall class structure. Despite its simplicity, the current oversampling method provided a measurable benefit in macro F1 and represents a practical solution within a standard Scikit-learn workflow.

6.5 Ethical and Societal Considerations

Predictive analytics in transport planning requires caution and transparency. Models trained on historical data may inadvertently inherit regional or reporting biases, such as underrepresentation of rural collisions or disparities in police reporting.
This project adheres to GDPR standards and employs fully anonymised DfT open data. The intent is preventive insight, not predictive policing.

Model outputs should inform *risk mitigation policies* rather than determine individual responsibility. Furthermore, transparency mechanisms—such as model explainability and open publication of code—are essential to maintaining accountability.
The ethical design of ML-based policy systems demands continuous auditing, ensuring that decisions derived from such models remain fair, interpretable, and socially responsible.

6.6 Policy Implications

The findings point toward actionable interventions that align with the UK's *Vision Zero 2050* target to eliminate road fatalities:

- **Speed Management:** Reduce speed limits in high-risk rural corridors and increase automated enforcement.
- **Lighting Improvements:** Prioritise illumination of single-carriageway and semi-urban roads with historically higher fatality rates.
- **Public Education:** Reinforce campaigns emphasising safe driving during low-light and adverse-weather conditions.

- **Infrastructure Planning:** Use severity-prediction maps to optimise placement of speed cameras and emergency-response stations.

Collectively, these measures demonstrate how machine-learning outputs can directly support data-informed transport policy.

---

7 Conclusion and Recommendations

7.1 Summary of Findings

This research implemented and compared two supervised machine-learning models—**Logistic Regression** and **Random Forest**—to predict collision severity using the *DfT STATS19 2024* dataset. Both models met the MLDM Task 1 requirements, using a dataset with over 100,000 records and more than seven key features.

The **Random Forest classifier** achieved the best trade-off between accuracy and generalisability (*accuracy = 0.54, macro F1 = 0.34*), while Logistic Regression offered interpretable, statistically transparent results.
Feature-importance analysis revealed that *speed_limit*, *number_of_vehicles*, and *number_of_casualties* are the strongest predictors, highlighting the relationship between impact energy, collision complexity, and outcome severity.

7.2 Limitations

- **Class Imbalance:** Fatal collisions represent less than 2 % of total data, constraining the model's sensitivity to rare but crucial cases.
- **Feature Limitation:** The dataset lacks spatial and temporal precision, omitting variables like GPS coordinates and time of day.
- **Interpretability Trade-off:** Ensemble models sacrifice transparency; Random Forest's aggregate structure obscures individual decision paths.
- **Synthetic Oversampling Bias:** Random duplication may amplify noise and limit generalisability.

7.3 Future Work

Future research should:

- Incorporate **spatial (GIS)** and **temporal (hour-of-day, seasonal)** features.
- Apply **advanced imbalance techniques** such as SMOTE, focal loss, or cost-sensitive classifiers.
- Use **Explainable AI (SHAP, LIME)** to visualise feature-level contributions.
- Experiment with **boosted ensembles** (XGBoost, LightGBM) for enhanced precision.
- Collaborate with DfT or local authorities to deploy real-time predictive dashboards for accident prevention.

7.4 Concluding Statement

This study demonstrates that machine learning, when ethically deployed, can transform open-government transport data into actionable policy intelligence.

While absolute predictive accuracy remains constrained by data limitations, the analytical framework delivers meaningful insights into *why* certain collisions become severe.

By coupling predictive analytics with domain expertise and evidence-based interventions, the UK can move closer to its *Vision Zero* goal of eliminating fatal road collisions by mid-century.

References

- Abdel-Aty, M. and Pande, A. (2005) 'Identifying crash propensity using specific traffic speed conditions', *Journal of Safety Research*, 36(1), pp. 97–108.
- Breiman, L. (2001) 'Random Forests', *Machine Learning*, 45(1), pp. 5–32.
- Chawla, N.V., Bowyer, K.W., Hall, L.O. and Kegelmeyer, W.P. (2002) 'SMOTE: Synthetic Minority Over-Sampling Technique', *Journal of Artificial Intelligence Research*, 16, pp. 321–357.
- Department for Transport (2024) *Road Safety Data – Collisions 2024 (STATS19)*. data.gov.uk. Available at: https://data.gov.uk/dataset/cb7ae6f0-4be6-4935-9277-47e5ce24a11f/ (Accessed: October 2025).
- Hosmer, D.W., Lemeshow, S. and Sturdivant, R.X. (2013) *Applied Logistic Regression*. 3rd edn. Hoboken, NJ: Wiley.
- Li, Z., Liu, P. and Wang, W. (2020) 'Exploring injury severity of crashes on rural two-lane highways: a random parameter approach', *Accident Analysis & Prevention*, 136, 105405.
- Pedregosa, F. et al. (2011) 'Scikit-learn: Machine Learning in Python', *Journal of Machine Learning Research*, 12, pp. 2825–2830.
- University of Salford (2025) *MLDM Assessment Brief and Writing Frame*. Internal document.
- World Health Organization (2023) *Global Status Report on Road Safety 2023*. Geneva: WHO.
- Yannis, G., Papadimitriou, E. and Theofilatos, A. (2017) 'Factors affecting accident severity and collision type on urban arterials in Athens', *Traffic Injury Prevention*, 18(4), pp. 365–370.