Vehicle Style Multi-Class Classification

Zakariya Ba Alawi
Student ID: 220027
Department of Software Engineering
Alfaisal University
Riyadh, Saudi Arabia

Mohammed Bawazir
Student ID: 230035
Department of Software Engineering
Alfaisal University
Riyadh, Saudi Arabia

Ahmed Bin Halabi
Student ID: 220026
Department of Software Engineering
Alfaisal University
Riyadh, Saudi Arabia

Saad Alkeridis
Student ID: 220621
Department of Software Engineering
Alfaisal University
Riyadh, Saudi Arabia

Mohmamed Haythem
Student ID: 220601
Department of Software Engineering
Alfaisal University
Riyadh, Saudi Arabia

Abdulrahman Hisham
Student ID: 220335
Department of Software Engineering
Alfaisal University
Riyadh, Saudi Arabia

***Abstract*—This project develops and compares multiple supervised learning models to classify vehicle *style* (e.g., Sedan, SUV, Coupe) from technical vehicle specifications. Using a real-world automotive dataset containing 11,914 records and 16 vehicle-style classes, we implement a consistent preprocessing pipeline (missing-value imputation, standardization, and one-hot encoding) and evaluate five classifiers: Logistic Regression, K-Nearest Neighbors (KNN), Support Vector Machine (SVM with RBF kernel), Decision Tree, and a Multi-Layer Perceptron (MLP). Models are compared using accuracy, macro-averaged precision/recall/F1, confusion matrices, and One-vs-Rest (OvR) macro ROC-AUC when probabilities are available. The best performing model is a class-weighted Decision Tree (max depth = 18) achieving accuracy of 0.8477 and macro F1 of 0.8284 on a stratified 80/20 split.**

***Index Terms*—Multi-class classification, vehicle type recognition, scikit-learn, decision trees, SVM, KNN, neural networks, ROC-AUC, confusion matrix.**

I. Introduction

Vehicle categorization is a core requirement in automotive inventory systems, online marketplaces, recommendation engines, and search/filter experiences. In many practical settings, vehicles are described by a mixture of numeric specifications (horsepower, cylinders, MPG) and categorical descriptors (fuel type, drivetrain, transmission). Automatically classifying a vehicle into a style class (e.g., 4dr SUV, Sedan, Wagon) supports consistent labeling and reduces manual effort.

The objective of this project is to build a robust multi-class classifier that predicts *Vehicle Style* from technical specifications and to compare multiple machine learning algorithms using consistent preprocessing and evaluation. The work aligns with standard classification requirements including model comparison and metric-driven analysis.

II. Background and Related Work

Vehicle style prediction is a multi-class classification problem on structured (tabular) data, where each record contains mixed numeric attributes (e.g., engine size, cylinders, highway MPG) and categorical attributes (e.g., make, model, transmission). Classical supervised learning methods such as logistic regression, k-nearest neighbors (kNN), decision trees, and support vector machines (SVMs) remain strong baselines for tabular problems, especially when paired with careful preprocessing and standardized evaluation [1], [2].

A practical challenge in real datasets is that feature types are heterogeneous and may contain missing values. For this reason, modern implementations commonly use a unified preprocessing pipeline that imputes missing values, scales numeric features, and encodes categorical features (e.g., one-hot encoding) before training the classifier [3]. Using a single reusable pipeline also prevents training–test leakage and ensures each model is evaluated under identical preprocessing assumptions [3].

Another common issue is class imbalance, where frequent vehicle styles dominate the dataset while rarer styles have limited examples. In such cases, accuracy alone can be misleading; macro-averaged precision/recall/F1 are often preferred because they weight each class equally and better reflect performance on minority classes [4]–[6]. Confusion matrices are also essential to diagnose which styles are systematically confused (e.g., SUV variants vs. hatchbacks).

For probability-based models, ROC-AUC can be extended to multi-class classification using strategies such as One-vs-Rest (OvR) or pairwise averaging. Recent work discusses multi-class ROC/AUC methodology and nonparametric comparisons for multi-class ROC analysis [7]. Additionally, AUC-focused objectives for imbalanced multi-class settings have been explored to make evaluation more decision-aligned [8]. These ideas motivate reporting macro ROC-AUC where probability estimates are available, alongside macro F1 as the primary selection metric.

III. Dataset Description

A. Data Source and Shape

We use the *Car Features and MSRP* dataset (CSV). After selecting the target and relevant features, the working dataset contains 11,914 samples and 10 input features with a 16-class target label. The data is split into training (9,531) and testing

(2,383) sets using a stratified 80/20 split with a fixed random seed for reproducibility.

B. Target Variable

The target column is **Vehicle Style**, a multi-class label with 16 classes. The largest classes include Sedan and 4dr SUV, indicating class imbalance.

C. Selected Features

To focus on technical attributes and avoid identity memorization, we exclude price and identity-like columns. The final feature set is shown in Table I.

TABLE I
SELECTED INPUT FEATURES AND TARGET.

Target (Y): Vehicle Style (16 classes)
Numeric (scaled): Engine HP, Engine Cylinders, Number of Doors, highway MPG, city mpg, Popularity
Categorical (one-hot): Engine Fuel Type, Transmission Type, Driven_Wheels, Vehicle Size

D. Dropped Columns (Rationale)

We drop MSRP to avoid price bias and ensure the model relies on technical attributes. We also drop Make/Model/Year as they behave like identifiers and can encourage memorization rather than learning general vehicle-style patterns. Market Category is removed due to high missingness and messy multi-label formatting.

TABLE II
NOTABLE MISSING VALUES (ORIGINAL DATASET).

Column	Missing Count
Market Category	3742
Engine HP	69
Engine Cylinders	30
Number of Doors	6
Engine Fuel Type	3

E. Class Distribution

The dataset is imbalanced. Table III lists the most frequent classes.

TABLE III
TOP VEHICLE STYLE CLASSES (COUNTS).

Vehicle Style	Count
Sedan	3048
4dr SUV	2488
Coupe	1211
Convertible	793
4dr Hatchback	702
Crew Cab Pickup	681
Extended Cab Pickup	623
Wagon	592

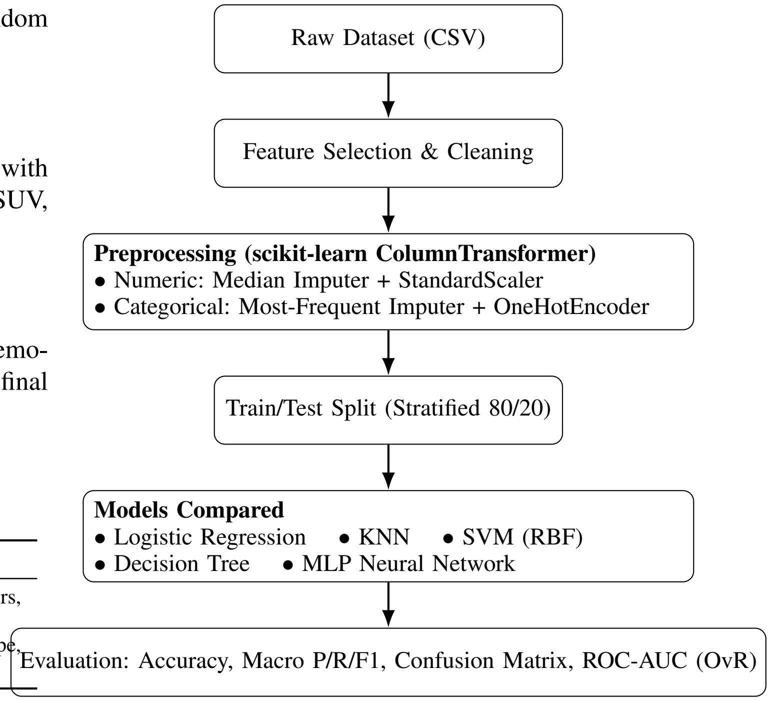

Fig. 2. End-to-end pipeline for vehicle style multi-class classification.

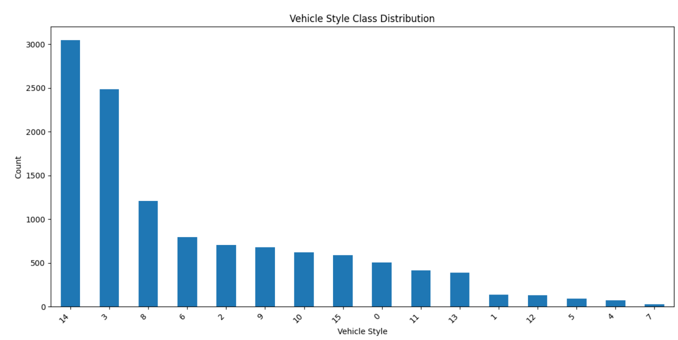

Fig. 1. Vehicle Style class distribution (bar chart)

IV. METHODOLOGY

A. Train/Test Split

We use an 80/20 split with `stratify=y` to preserve the class distribution in both train and test sets. Label encoding is applied to the target labels to enable ROC-AUC computations.

B. Preprocessing Pipeline

A single scikit-learn pipeline is applied to all models for a fair comparison:

- **Numeric**: median imputation + StandardScaler
- **Categorical**: most-frequent imputation + OneHotEncoder (handle unknown categories)

Using a pipeline avoids data leakage by fitting preprocessing only on training data.

C. Models Implemented

We train five classifiers:

1) Logistic Regression (class-weighted baseline)
2) KNN ($k = 15$)
3) SVM (RBF kernel, $C = 5$, probability enabled, class-weighted)
4) Decision Tree (class-weighted, max depth $= 18$)
5) Neural Network (MLP, hidden layers $(128, 64)$, ReLU, Adam, early stopping)

TABLE IV
MODEL CONFIGURATION SUMMARY.

Model	**Key settings**
Logistic Regression	class_weight=balanced, max_iter=2000
KNN	n_neighbors=15
SVM (RBF)	C=5.0, gamma=scale, probability=True, class_weight=balanced
Decision Tree	max_depth=18, class_weight=balanced
MLP	(128,64), ReLU, Adam, max_iter=300, early_stopping=True

D. Evaluation Metrics

We report accuracy and macro-averaged precision, recall, and F1. Macro-averaging gives each class equal weight:

$$\begin{aligned} \text{Precision}_{\text{macro}} &= \frac{1}{K}\sum_{i=1}^{K}\text{Precision}_i, \\ \text{Recall}_{\text{macro}} &= \frac{1}{K}\sum_{i=1}^{K}\text{Recall}_i, \\ \text{F1}_{\text{macro}} &= \frac{1}{K}\sum_{i=1}^{K}\frac{2\,\text{Precision}_i\,\text{Recall}_i}{\text{Precision}_i + \text{Recall}_i}. \end{aligned} \tag{1}$$

We emphasize macro metrics to reduce majority-class dominance. Confusion matrices show class-wise errors, and multi-class ROC-AUC is computed using One-vs-Rest (OvR) with macro averaging for models that provide probability estimates.

V. RESULTS

A. Overall Comparison

Table V summarizes the test-set performance. The Decision Tree achieves the best macro F1 and the highest accuracy, making it the overall winner.

TABLE V
MODEL PERFORMANCE ON THE TEST SET (2,383 SAMPLES).

Model	**Acc**	**Prec$_M$**	**Rec$_M$**	**F1$_M$**	**AUC$_M$**
Decision Tree	0.8477	0.8046	0.8656	0.8284	0.9701
MLP	0.7398	0.7227	0.6908	0.6931	0.9805
KNN	0.7104	0.6765	0.6592	0.6599	0.9788
SVM (RBF)	0.6496	0.6044	0.7346	0.6306	0.9781
Logistic Reg.	0.5661	0.5017	0.6630	0.5313	0.9578

B. Confusion Matrix Analysis

Figure 3 should include the confusion matrix screenshot for the best model (Decision Tree). This visualization highlights which vehicle styles are most frequently confused (e.g., similar body types such as Sedan vs Wagon, or pickup variants).

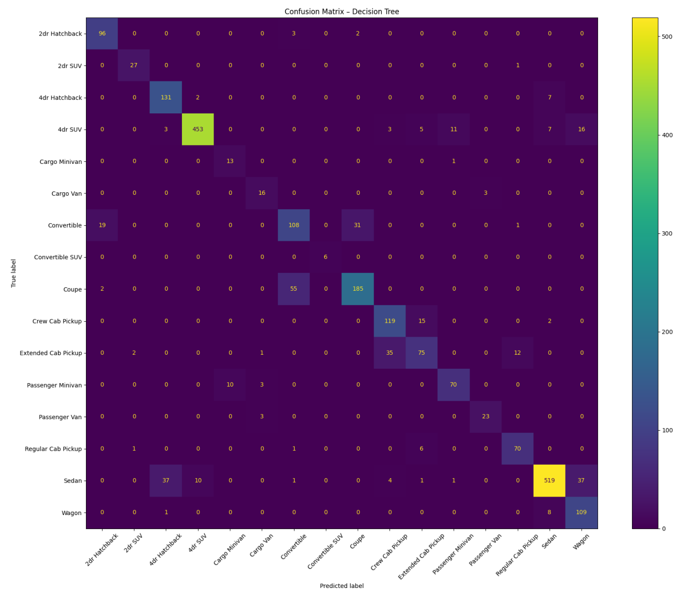

Fig. 3. Confusion matrix for the best model (Decision Tree)

C. ROC Curves (OvR)

ROC curves are plotted using One-vs-Rest for the top 6 most frequent test classes to keep the figure readable. Figure 4 should include the ROC curve screenshot exported from the notebook.

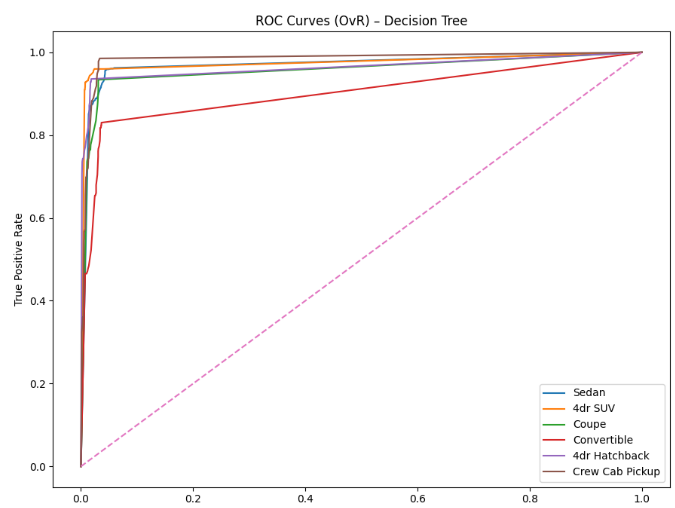

Fig. 4. Multi-class ROC (OvR) curves for top 6 classes.

VI. Discussion

A. Why the Decision Tree Won

The Decision Tree performed best because it can naturally model non-linear decision boundaries and interactions between categorical descriptors (after one-hot encoding) and numeric specifications. With class weighting enabled, the model also improved performance on smaller classes, which is reflected in the strong macro recall and macro F1.

B. Why Logistic Regression Underperformed

Logistic Regression provides a linear decision boundary in the transformed feature space. Vehicle style classification likely depends on complex non-linear interactions (e.g., drivetrain + size + doors + MPG) that are not well captured by a linear model alone, resulting in lower macro F1.

C. Precision vs Recall Tradeoff

In this use case, misclassifying a vehicle style can cause incorrect recommendations or incorrect filtering in a marketplace. Depending on the application, recall may be prioritized (do not miss SUVs) or precision may be prioritized (avoid labeling sedans as SUVs). Macro metrics provide a balanced view across all categories.

D. Benefits and Practical Implications

A deployed classifier can:

- reduce manual labeling effort in vehicle listings,
- enable consistent taxonomy in inventory systems,
- improve recommendations and search/filter relevance,
- help data analytics teams segment vehicles by style using only specifications.

VII. Conclusion

This project implemented and compared five classification models for vehicle style prediction using a consistent scikit-learn pipeline and a stratified train/test split. Across all metrics, the class-weighted Decision Tree (max depth = 18) achieved the best overall performance with accuracy 0.8477 and macro F1 0.8284. The results indicate that non-linear, rule-based models can capture vehicle-style structure effectively when trained on mixed technical features.

VIII. Future Work

Future improvements include:

- hyperparameter tuning with GridSearchCV for all models (especially SVM/MLP),
- adding engineered features (e.g., MPG ratios, power-to-cylinder interactions),
- incorporating *Market Category* via robust multi-label encoding,
- using ensembles (Random Forest, Gradient Boosting) and comparing to the Decision Tree baseline,
- evaluating calibration quality and deployment considerations (latency and interpretability).

IX. Screenshots / Appendix

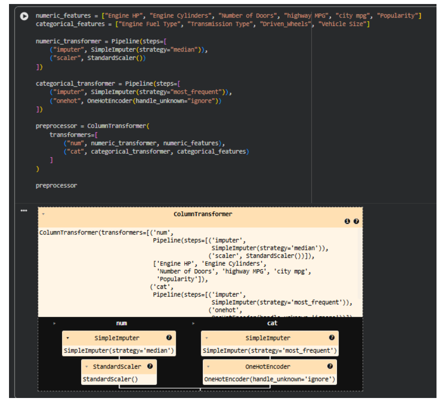

Fig. 5. Preprocessing pipeline (ColumnTransformer + model pipeline) used consistently across all classifiers.

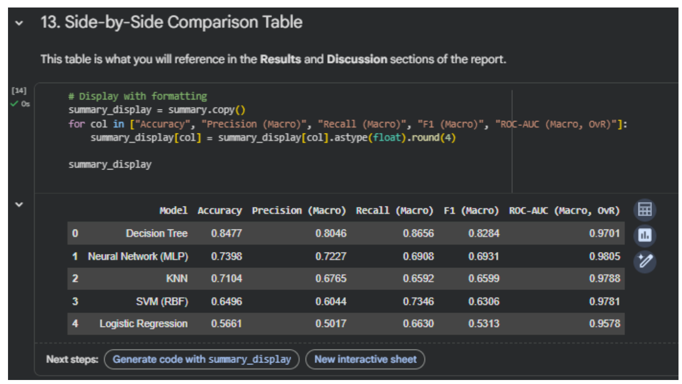

Fig. 6. Final model comparison table showing test-set metrics (Accuracy, Macro Precision/Recall/F1, and Macro ROC-AUC where applicable).

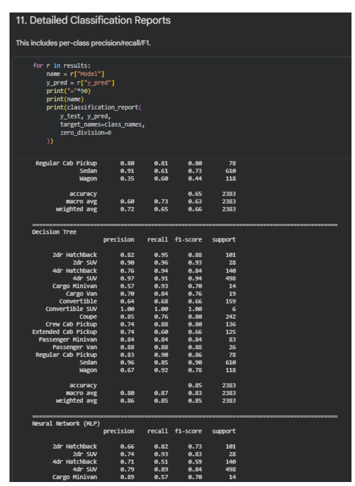

Fig. 7. Classification report for the best-performing model (per-class precision, recall, and F1-score).

Fig. 8. Winner selection summary based on overall performance, emphasizing macro F1 due to class imbalance.

References

[1] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning: with Applications in R*, 2nd ed. Springer, 2021.

[2] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, 2nd ed. O'Reilly Media, 2019.

[3] scikit-learn developers, "scikit-learn user guide: Pipelines and composite estimators," https://scikit-learn.org/stable/modules/compose.html, 2025, accessed: 2025-12-18.

[4] J. A. Opitz, "A closer look at classification evaluation metrics and a critical reflection of common evaluation practice," *Transactions of the Association for Computational Linguistics*, 2024.

[5] Z. Chen *et al.*, "A review of learning from imbalanced data," *Artificial Intelligence Review*, 2024.

[6] M. Rezvani and X. Wang, "A comprehensive review of imbalanced data classification: Challenges, solutions, and future directions," *Applied Soft Computing*, 2023.

[7] J. Xu, "Comparing multi-class classifier performance by multi-class roc analysis: A nonparametric approach," *Neurocomputing*, 2024.

[8] P. Gao, Q. Xu, P. Wen, H. Shao, Y. He, and Q. Huang, "Towards decision-friendly auc: Learning multi-classifier with aucμ," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023, https://ojs.aaai.org/index.php/AAAI/article/view/25926.