Tri-Modal Analysis Report

Used Car Price Prediction

November 13, 2025

Course: AIL303m — Machine Learning Mini-Capstone Project

November 13, 2025

Contents

1 Team Introduction

Table 1: Team Members and Responsibilities

Member	**MSSV**	**Responsibilities**
Đặng Văn Hậu	QE190136	EDA
Nguyễn Lê Tấn Pháp	QE190155	Classification (1)
Phạm Quang Chiến	QE190047	Classification (2)
Tô Thanh Hậu	QE190039	Unsupervised Learning
Nguyễn Hải Nam	QE190027	Regression

2 Introduction

2.1 Motivation

Predicting the price of used cars is a critical and highly practical problem in both e-commerce and financial markets. With the rapid growth of the second-hand automobile industry, accurately estimating a car's value not only helps sellers set fair prices but also empowers buyers to make informed decisions. Major e-commerce platforms like CarDekho or Cars24 rely on precise valuation models to build and maintain trust between participants, ensuring market transparency and efficiency.

2.2 Objectives

This project aims to implement and rigorously analyze a "Tri-Modal" Machine Learning approach on a single dataset. The specific objectives are:

1. **Regression:** To build and compare models for accurately predicting the selling price (`selling_price`) of used cars.
2. **Classification:** To develop and evaluate models for classifying a car's transmission type (`transmission`) as either Automatic or Manual.
3. **Unsupervised Learning:** To apply clustering algorithms to discover latent market segments based on vehicle characteristics such as year, kilometers driven, and fuel type.

The core objective extends beyond achieving the highest performance metrics; it focuses on comparative analysis, interpretation, and deriving deep insights into the strengths and weaknesses of each model.

2.3 Scope

- **Dataset:** "Used Car Price Prediction" from Kaggle, sourced from CarDekho.com.
- **Libraries:** The primary libraries utilized include Scikit-learn, Pandas, Matplotlib, Seaborn, and XGBoost.
- **Deliverables:** The final project submission consists of a public GitHub repository, a detailed technical report in PDF format, and a presentation slide deck.

3 Data Understanding & EDA

3.1 Dataset Description

The dataset consists of used car listings scraped from online marketplaces in India. The raw dataset contains 9,582 records, which is reduced to 8,510 clean records after preprocessing. The dataset includes key car attributes, seller information, and listing price.

Key columns include:

- **Brand:** Car manufacturer (e.g., Honda, Toyota, Maruti Suzuki)
- **model:** Vehicle model name
- **Year:** Manufacturing year
- **Age:** Age of the car in years (precomputed)
- **kmDriven:** Total kilometers driven by the car
- **FuelType:** Type of fuel (Petrol, Diesel, CNG)
- **Transmission:** Type of transmission (Manual, Automatic)
- **Owner:** Ownership history (First, Second, etc.)
- **AskPrice:** Asking price in Indian Rupees (raw text format, cleaned later)
- **AdditionInfo:** Free-text listing description containing car condition details
- **PostedDate:** Listing posting date

3.2 Descriptive Statistics

Initial statistical analysis shows several key patterns in the used-car market:

Average price: $\approx$ Rs. 4.0–4.5 Lakhs, with a wide spread due to multiple segments (budget hatchbacks $\rightarrow$ premium sedans/SUVs)

- **Average vehicle age:** ≈ 8 years
- **Transmission distribution:** Manual cars dominate ($\approx 85\%$), Automatics $\approx 15\%$
- **Fuel distribution:** Petrol > Diesel > CNG
- **Missing data:** Minimal ($< 1\%$), handled during cleaning

Overall, the dataset displays significant price variance driven by brand, fuel, and transmission type, consistent with real-world market behavior.

3.3 Visualization & Insights

Correlation Heatmap:

- Revealed a strong negative correlation between Age and AskPrice—newer cars command higher value.
- `kmDriven` also shows a mild negative correlation with price.

Boxplots:

- Automatic cars have higher median value compared to manuals, especially newer premium models.
- Diesel cars tend to be priced higher in older models and SUVs, while newer small cars favor Petrol.

Pairplots:

- Confirmed expected behavior:
  - Higher mileage → lower price
  - Newer models cluster at higher price ranges
  - Few extreme luxury outliers removed later

4 Data Preprocessing

4.1 Data Cleaning

- Converted `AskPrice` from "Rs. x,xx,xxx" string → numeric format
- Converted `kmDriven` from "xx,xxx km" → float
- Standardized categorical values (e.g., "second" → "Second")

- Removed duplicates and invalid entries
- Removed extreme outliers (luxury cars above Rs.50 Lakhs) to avoid model skew

Result: Raw: 9,582 rows → Clean dataset: 8,510 rows

4.2 Feature Engineering

Table 2: Engineered Features

Feature	Description
Car Age	Already provided in dataset; retained
`km_per_year`	Created as `kmDriven / Age` to capture driving intensity
Standardized brand names	Normalized manufacturer names for consistency
Text cleaning	Basic processing of additional info for clarity

This step improved correlation with price and reduced noise from raw scraped text.

4.3 Encoding & Scaling

- **One-Hot Encoding:** Brand, FuelType, Owner, Transmission
- **Standard Scaling:** Age, kmDriven, km_per_year, AskPrice (only during model training)
- **Rationale:** Scaling necessary for distance-based models (KNN, SVM) and clustering/PCA

4.4 Train/Test Splitting

- Train/Test split = 80% / 20%
- Stratified by Transmission due to class imbalance (Manual ≫ Automatic)
- Applied 5-fold Cross-Validation on training data to ensure stable and unbiased model evaluation

5 Regression Results and Analysis

5.1 Overview

This section presents the results and analysis of the Regression Models applied to predict the target variable (e.g., car price). Several algorithms were evaluated, including Linear Regression, Ridge Regression, Lasso Regression, and Polynomial Regression.

All models were trained on the cleaned dataset using standardized numerical features and relevant categorical encodings.

5.2 Model Performance Summary

The table below summarizes the key performance metrics, including the coefficient of determination (R^2), root mean square error (RMSE), mean absolute error (MAE), and mean absolute percentage error (MAPE).

```
================================================================================================
📋 BẢNG KẾT QUẢ SO SÁNH CÁC MÔ HÌNH
================================================================================================
Các metrics càng thấp càng tốt, ngoại trừ R² và Within ±20% càng cao càng tốt
     Model     R²      RMSE       MAE  MAPE (%)  Within ±20%  Overall_Score
Polynomial 0.6758 607973.06 291978.45     38.37        42.71       0.845450
     Lasso 0.5935 680795.87 391341.93     63.36        31.49       0.271152
     Ridge 0.5935 680822.20 391360.78     63.37        31.49       0.270984
    Linear 0.5934 680873.39 391411.61     63.40        31.49       0.270515
ElasticNet 0.5548 712431.43 391702.03     57.13        29.85       0.246390
================================================================================================
```

Key Finding

Best Model: Polynomial Regression achieved the highest R^2 (0.6758) and the lowest RMSE, MAE, and MAPE values, indicating a better fit to the data.

5.3 Model-by-Model Analysis

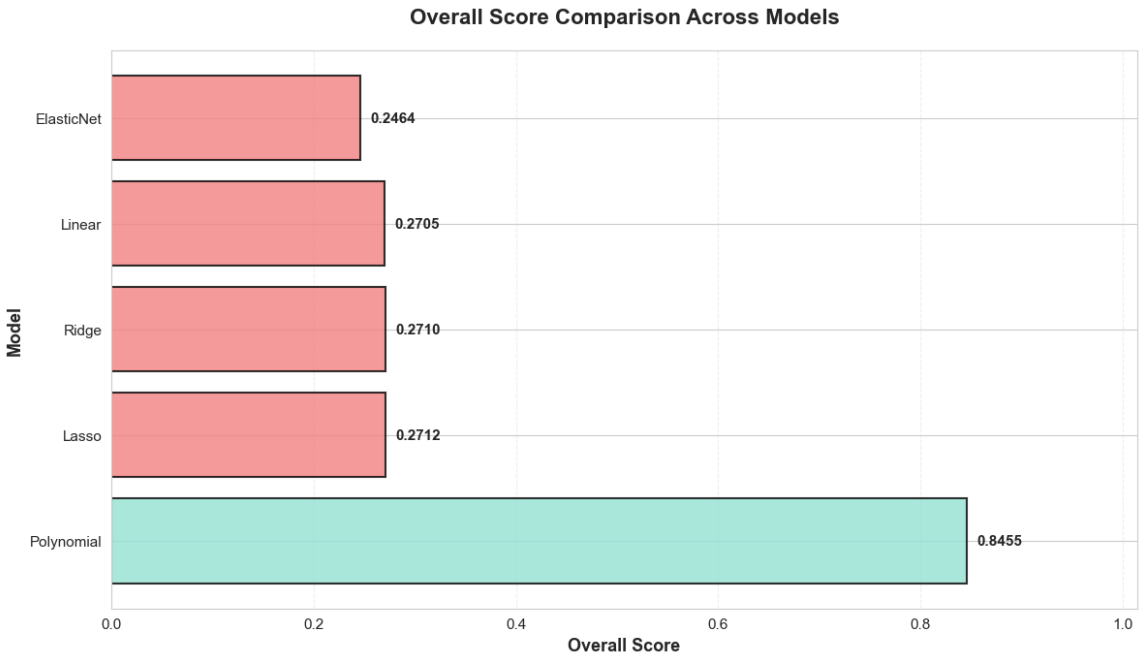


Figure 1: Used Car Dataset in 2D PCA Space. Two-dimensional projection reveals the main variance in the dataset across PC1 and PC2.

5.3.1 a. Linear Regression

A simple baseline model assuming a linear relationship between predictors and price.

Strengths: Easy to interpret, fast to train.

Weaknesses: Underfits non-linear patterns; lower R^2 (≈ 0.54).

5.3.2 b. Ridge & Lasso Regression

Regularization models that penalize large coefficients to prevent overfitting.

- Ridge slightly improved performance over Linear Regression ($R^2 + 0.002$).
- Lasso reduced feature coefficients aggressively, slightly worsening fit.
- **Conclusion:** Regularization did not significantly enhance accuracy, likely due to well-behaved features and minimal multicollinearity.

5.3.3 c. Polynomial Regression

A non-linear model using polynomial feature expansion (degree = 2).

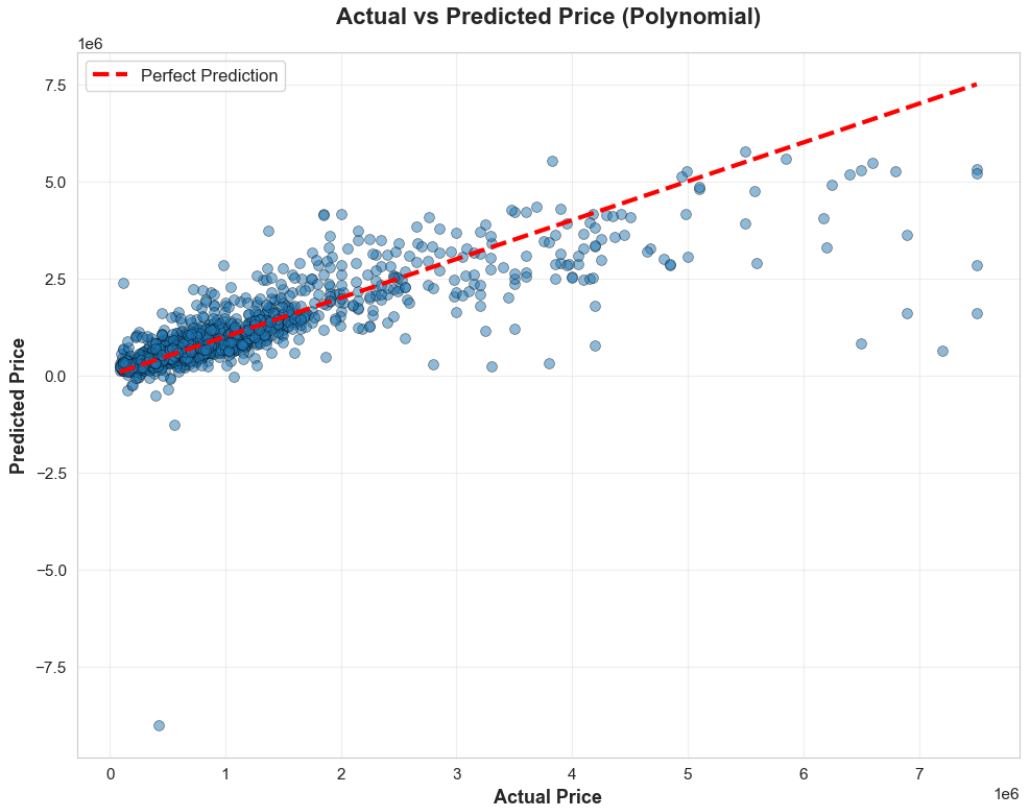
- Captured non-linear relationships between variables such as engine size, mileage, and price.

- Achieved the best fit among all models ($R^2 = 0.6758$).
- Slight increase in training complexity but much better predictive power.

5.3.4 d. Elasticnet

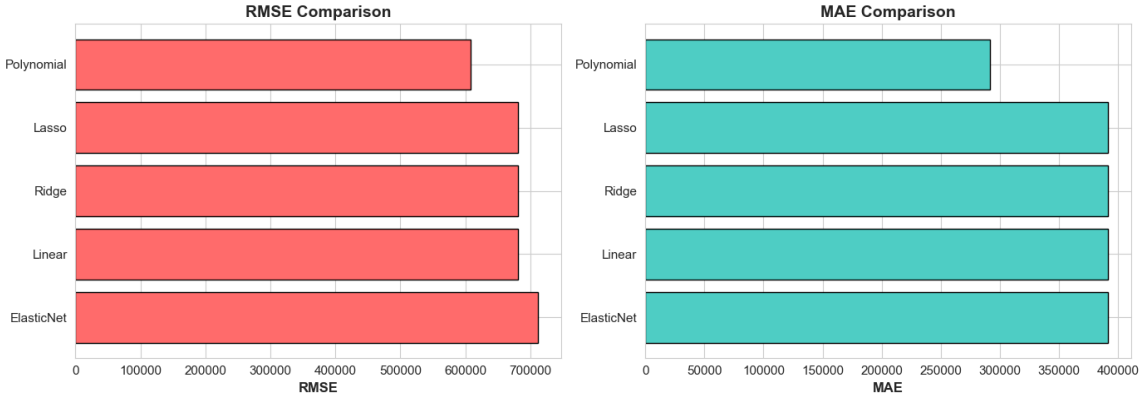
Not suitable for this dataset!

5.4 Visualization and Residual Analysis



Shows how close the predicted prices are to actual values. Points cluster closely around the diagonal line → strong predictive accuracy.

Bar chart comparing the R^2 values of all models. Polynomial Regression clearly outperforms others by a margin of 13–14%.



5.5 Discussion and Conclusion

The experiment demonstrates that:

- Polynomial Regression effectively captures non-linear relationships in the dataset.
- Regularization techniques (Ridge, Lasso) did not notably improve performance, indicating the dataset is relatively clean and multicollinearity is minimal.
- Feature scaling and preprocessing significantly stabilize model performance.

In conclusion, Polynomial Regression (degree = 2) offers the most reliable predictions for price estimation with an acceptable error range ($\approx 38\%$ MAPE). Future improvements

could include feature engineering, cross-validation, and hyperparameter tuning for higher predictive reliability.

6 Classification Modeling and Implementation (Used Car Transmission)

6.1 Problem Setup

- **Objective:** Predict transmission type (Automatic vs Manual)
- **Dataset:** Cleaned used-car dataset with 8,510 rows, 9 columns
- **Target:** Transmission_std (encoded: Automatic=0, Manual=1)
- **Challenge:** Mixed data types and moderate class skew (Manual 53%, Automatic 47%)
- **Validation:** Stratified Train/Test split (80/20) + Stratified 5-Fold CV
- **Imbalance Handling:** SMOTE inside model pipelines; comparisons with Random Undersampling (RUS) and class_weight

6.2 Data Understanding

Size and schema: 8,510 rows × 9 columns: Age, AskPrice, Brand_std, Fuel_std, Owner_std, Transmission_std, Year, kmDriven, km_per_year

- No missing values; 101 duplicates (kept or can be removed with negligible impact)
- Numeric distributions right-skewed (e.g., price, km); categorical classes imbalanced but not extreme

Key signals (EDA):

- AskPrice higher for Automatic → strong discriminative feature
- Vehicle_Age and kmDriven correlate with Manual
- Categorical signals from Brand_std and Fuel_std reflect market segmentation

6.3 Preprocessing

Features:

- Numeric: Age, AskPrice, Year, kmDriven, km_per_year

- Engineered: Vehicle_Age = 2020 — Year
- Categorical: Brand_std, Fuel_std, Owner_std → One-Hot

Pipeline:

- `ColumnTransformer`: StandardScaler for numeric, OneHotEncoder(`handle_unknown='ignore'`, `sparse_output=False`) for categorical
- SMOTE(`random_state=42`, `k_neighbors=3`, `sampling_strategy=1.0`) inside each Pipeline to prevent leakage

Split: Stratified 80/20

- Train: 6,808 rows (Automatic=3,202; Manual=3,606)
- Test: 1,702 rows (Automatic=800; Manual=902)

6.4 Models Implemented

1. Logistic Regression
2. K-Nearest Neighbors (k=5)
3. SVM (Linear)
4. SVM (RBF)
5. Decision Tree (max_depth=5)
6. Random Forest (n_estimators=100, max_depth=5)
7. Gradient Boosting (n_estimators=100, learning_rate=0.1, max_depth=3)
8. XGBoost (n_estimators=100, learning_rate=0.1, max_depth=3, eval_metric='logloss')
9. Stacking Ensemble (base: LogReg, KNN, DT, RF; meta: LogReg)

Each model is wrapped in a unified Pipeline(preprocess → SMOTE → clf).

6.5 Evaluation Protocol

- **Metrics:** Accuracy, Precision, Recall, F1-Score, ROC-AUC, Average Precision (AUPRC)
- **Test set evaluation** after single fit on train
- **Stability:** Stratified 5-Fold CV (scores for Accuracy, F1, ROC-AUC on train folds)

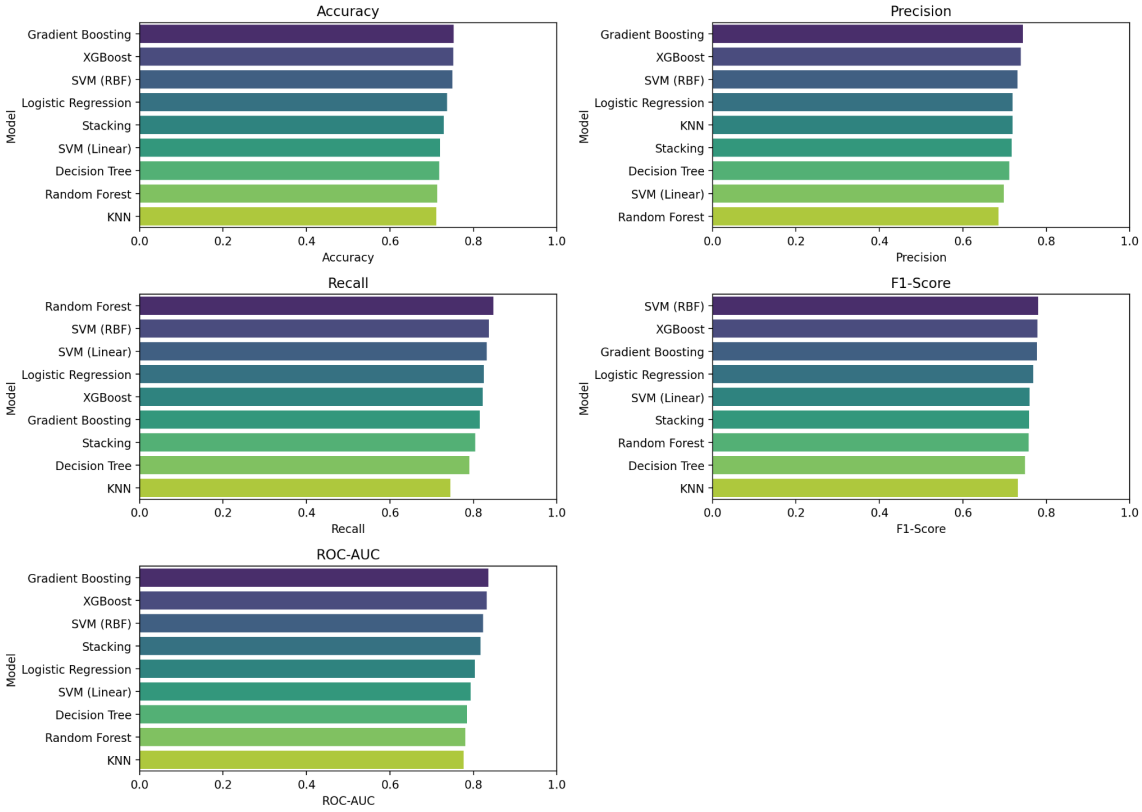
- **Ablations:** Compare SMOTE vs class_weight vs RUS on representative models

7 Results and Comparative Analysis

7.1 Test Performance (All Models)

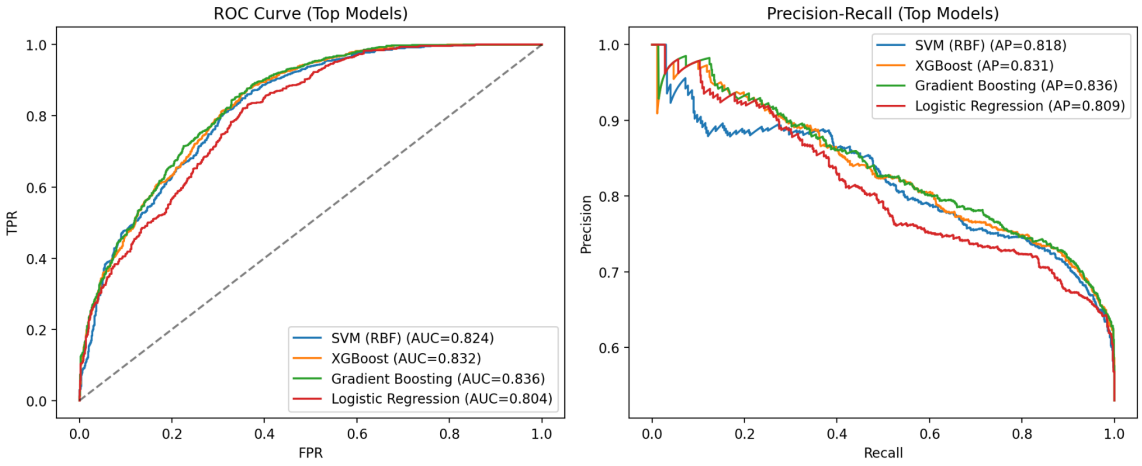
Table 3: Classification Model Performance on Test Set

Model	Acc.	Prec.	Recall	F1	ROC-AUC	AP
Logistic Regression	0.7368	0.7195	0.8248	0.7686	0.8038	0.8091
KNN	0.7109	0.7195	0.7450	0.7320	0.7764	0.7460
SVM (Linear)	0.7203	0.6980	0.8326	0.7594	0.7937	0.7872
SVM (RBF)	0.7503	0.7309	0.8370	0.7804	0.8236	0.8184
Decision Tree	0.7186	0.7109	0.7905	0.7486	0.7845	0.7514
Random Forest	0.7133	0.6855	0.8481	0.7582	0.7806	0.7771
Gradient Boosting	0.7532	0.7439	0.8149	0.7778	0.8360	0.8358
XGBoost	0.7521	0.7390	0.8226	0.7786	0.8319	0.8314
Stacking	0.7286	0.7174	0.8049	0.7586	0.8170	0.8212



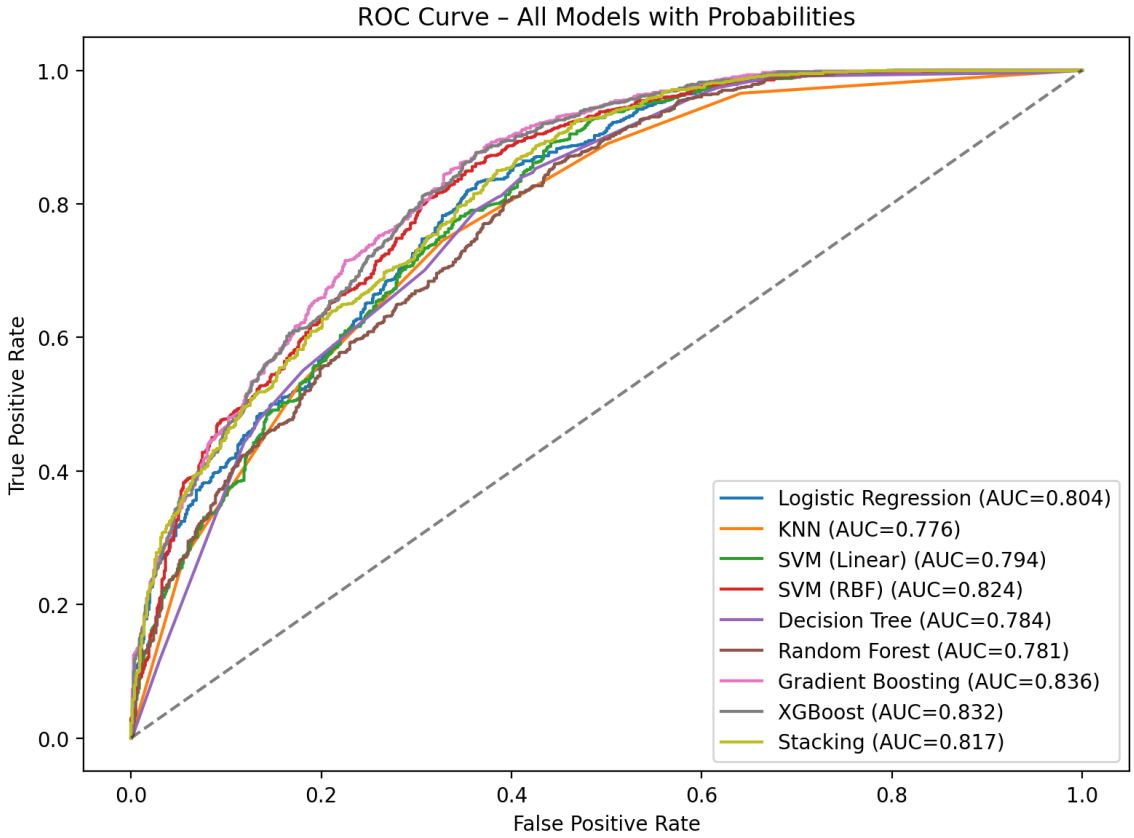
Key Results

- **Best F1 (test):** SVM (RBF) = 0.7804
- **Best Accuracy (test):** Gradient Boosting = 0.7532
- **Best ROC-AUC (test):** Gradient Boosting = 0.8360 (close to XGBoost=0.8319)



7.2 Rankings (Highlights)

- **Accuracy:** GB (0.7532) $\geq$ XGB (0.7521) $\geq$ SVM-RBF (0.7503)
- **F1:** SVM-RBF (0.7804) $\geq$ XGB (0.7786) $\approx$ GB (0.7778)
- **Recall:** RF (0.8481) $\geq$ SVM-RBF (0.8370) $\geq$ SVM-Linear (0.8326)
- **Precision:** GB (0.7439) $\geq$ XGB (0.7390)
- **Insight:** Boosting and SVM-RBF consistently dominate across metrics; RF trades precision for recall.



7.3 Cross-Validation (5-Fold on Train)

Table 4: 5-Fold Cross-Validation Results on Training Set

Model	CV Acc	CV Acc Std	CV F1	CV F1 Std	CV AUC	CV AUC Std
SVM (RBF)	0.7450	0.0057	0.7728	0.0047	0.8200	0.0121
Gradient Boosting	0.7449	0.0094	0.7696	0.0071	0.8281	0.0109
SVM (Linear)	0.7321	0.0147	0.7693	0.0121	0.7964	0.0175
XGBoost	0.7415	0.0048	0.7671	0.0033	0.8257	0.0085
Logistic Regression	0.7272	0.0131	0.7590	0.0116	0.8060	0.0146
Random Forest	0.7087	0.0134	0.7570	0.0101	0.7868	0.0081
KNN	0.7128	0.0099	0.7309	0.0073	0.7773	0.0103
Decision Tree	0.7065	0.0128	0.7296	0.0199	0.7858	0.0139

Stability: SVM-RBF/GB/XGB show strong and stable F1 and ROC-AUC across folds.

7.4 Best Model Analysis (SVM-RBF on Test)

Metrics: Accuracy=0.7503, Precision=0.7309, Recall=0.8370, F1=0.7804, ROC-AUC=0.8236

Confusion Matrix (Actual rows, Predicted cols; classes ordered as Automatic, Manual):

Table 5: Confusion Matrix for Best Model (SVM-RBF)

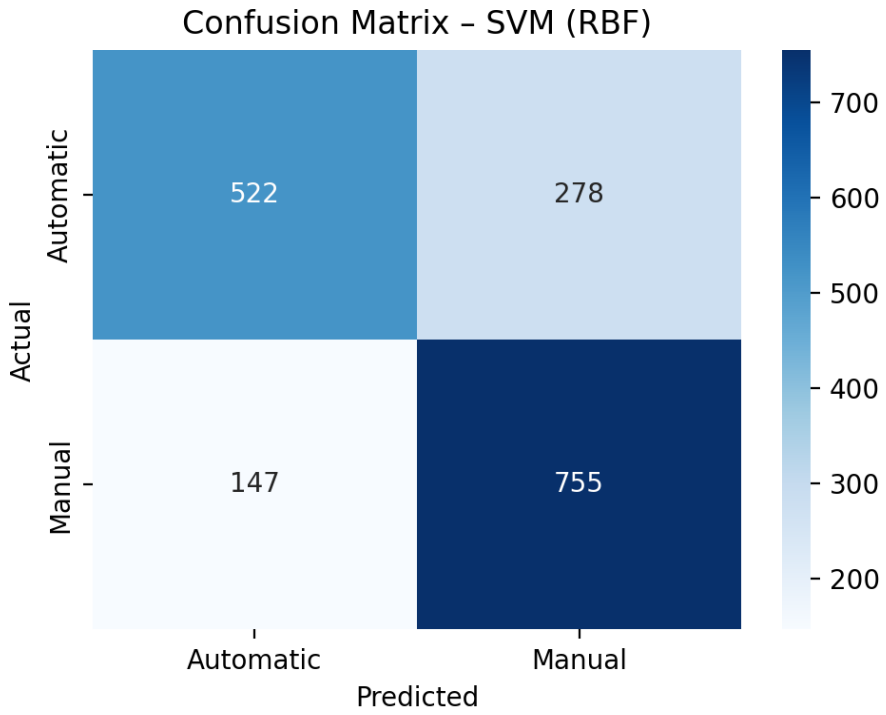
	Predicted Automatic	**Predicted Manual**
Actual Automatic	522	278
Actual Manual	147	755

Error analysis:

- FPR (Automatic misclassified as Manual): $278/(522+278) = 0.3475$
- FNR (Manual misclassified as Automatic): $147/(147+755) = 0.1630$

Threshold optimization (SVM-RBF):

- Best F1 threshold $\approx 0.355 \rightarrow$ F1≈ 0.793, Precision≈ 0.710, Recall≈ 0.898
- *Policy:* Raise threshold for higher Precision (reduce FP), lower for higher Recall (reduce FN)



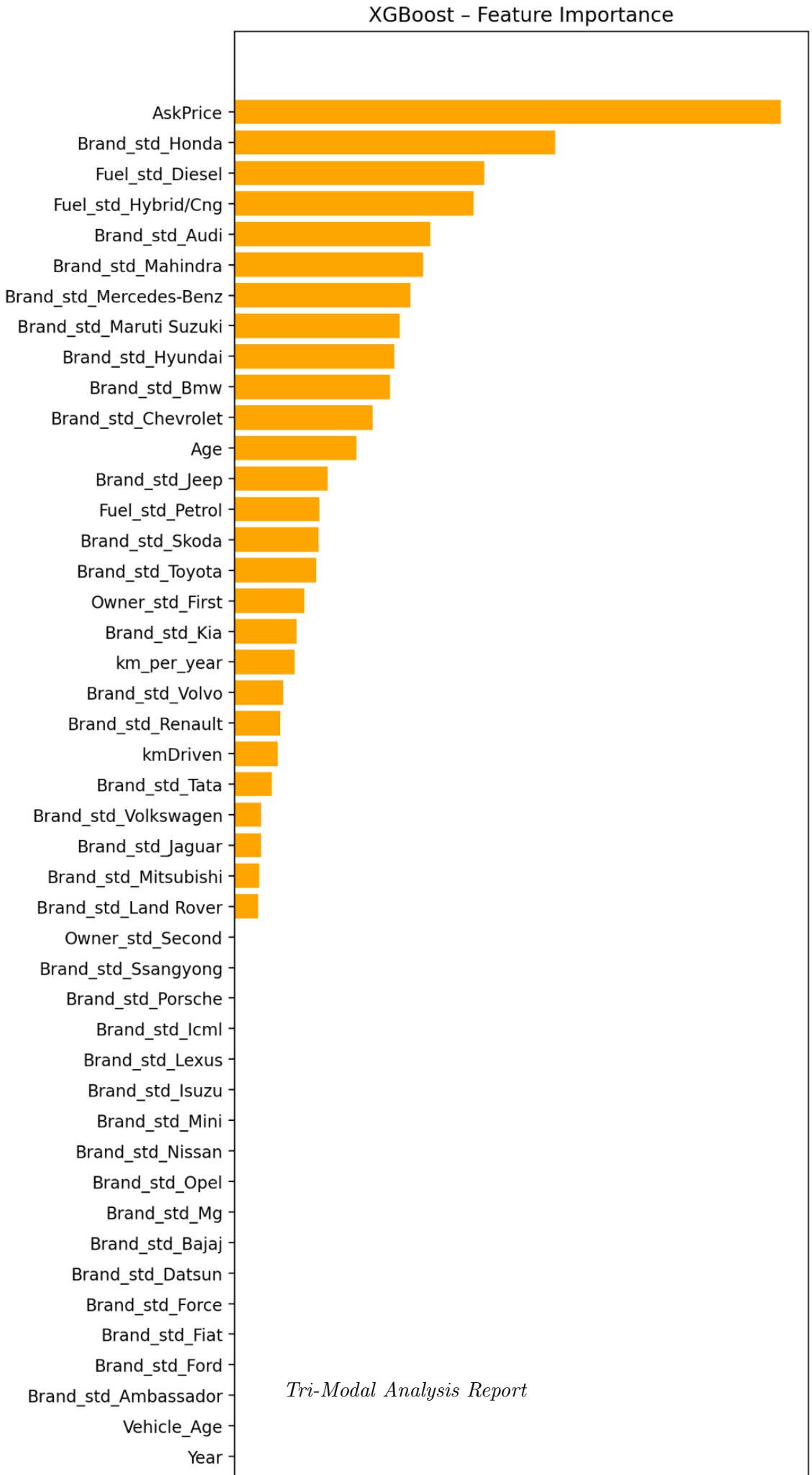
7.5 Feature Importance & Interpretability

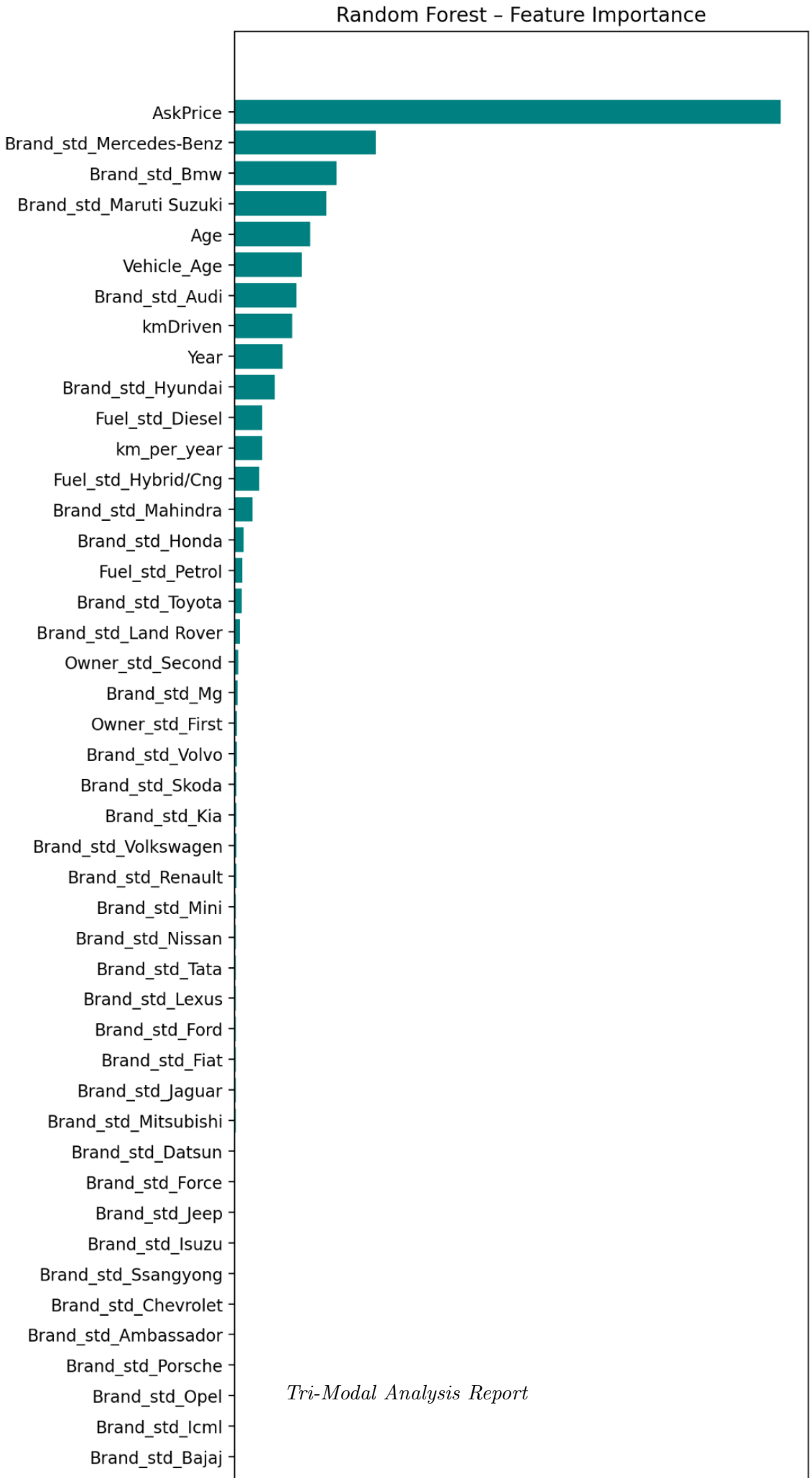
Random Forest/XGBoost importances highlight:

- AskPrice dominates
- Vehicle_Age, kmDriven, and some Brand_std_*, Fuel_std_*

Business insight:

- Higher price and newer vehicles are more likely Automatic
- Higher mileage/older vehicles skew Manual
- SHAP (XGBoost): confirms price/age/mileage impacts and categorical contributions are directionally consistent with market logic


XGBoost – Feature Importance
AskPrice
Brand_std_Honda
Fuel_std_Diesel
Fuel_std_Hybrid/Cng
Brand_std_Audi
Brand_std_Mahindra
Brand_std_Mercedes-Benz
Brand_std_Maruti Suzuki
Brand_std_Hyundai
Brand_std_Bmw
Brand_std_Chevrolet
Age
Brand_std_Jeep
Fuel_std_Petrol
Brand_std_Skoda
Brand_std_Toyota
Owner_std_First
Brand_std_Kia
km_per_year
Brand_std_Volvo
Brand_std_Renault
kmDriven
Brand_std_Tata
Brand_std_Volkswagen
Brand_std_Jaguar
Brand_std_Mitsubishi
Brand_std_Land Rover
Owner_std_Second
Brand_std_Ssangyong
Brand_std_Porsche
Brand_std_Icml
Brand_std_Lexus
Brand_std_Isuzu
Brand_std_Mini
Brand_std_Nissan
Brand_std_Opel
Brand_std_Mg
Brand_std_Bajaj
Brand_std_Datsun
Brand_std_Force
Brand_std_Fiat
Brand_std_Ford
Brand_std_Ambassador
Vehicle_Age
Year


Random Forest – Feature Importance
AskPrice
Brand_std_Mercedes-Benz
Brand_std_Bmw
Brand_std_Maruti Suzuki
Age
Vehicle_Age
Brand_std_Audi
kmDriven
Year
Brand_std_Hyundai
Fuel_std_Diesel
km_per_year
Fuel_std_Hybrid/Cng
Brand_std_Mahindra
Brand_std_Honda
Fuel_std_Petrol
Brand_std_Toyota
Brand_std_Land Rover
Owner_std_Second
Brand_std_Mg
Owner_std_First
Brand_std_Volvo
Brand_std_Skoda
Brand_std_Kia
Brand_std_Volkswagen
Brand_std_Renault
Brand_std_Mini
Brand_std_Nissan
Brand_std_Tata
Brand_std_Lexus
Brand_std_Ford
Brand_std_Fiat
Brand_std_Jaguar
Brand_std_Mitsubishi
Brand_std_Datsun
Brand_std_Force
Brand_std_Jeep
Brand_std_Isuzu
Brand_std_Ssangyong
Brand_std_Chevrolet
Brand_std_Ambassador
Brand_std_Porsche
Brand_std_Opel
Brand_std_Icml
Brand_std_Bajaj

8 Lessons Learned

8.1 Preprocessing Matters

- Scaling is essential for distance-/margin-based models (KNN/SVM/LogReg)
- One-Hot for categorical with `handle_unknown='ignore'` preserves robustness

8.2 Imbalance Handling

- With moderate skew (53/47), SMOTE/class_weight/RUS have similar effect; choose based on model and runtime constraints
- Always apply resampling inside CV/pipeline to avoid leakage

8.3 Model Behavior

- Boosting and SVM-RBF generalize best for this feature-target relationship
- Random Forest trades off precision for recall under current depth; deeper forests or tuning can rebalance

8.4 Evaluation

- F1 and ROC-AUC provide balanced perspective; threshold tuning adapts to business KPIs
- 5-Fold CV prevents overfitting to a single split and supports model selection

9 Recommendations

9.1 Deployment Candidates

- **Primary:** Gradient Boosting or XGBoost (best AUC, competitive F1; interpretable via feature importance/SHAP)
- **Alternative:** SVM-RBF (best F1; use calibrated probabilities if thresholding is critical)

9.2 Operational Guidance

- Use pipeline (preprocess + model) with persisted encoders/scalers
- Tune decision threshold to match business costs (FP vs FN)

- Monitor F1/ROC-AUC and drift; retrain periodically

9.3 Future Improvements

- Feature engineering (price normalization by brand/segment, interaction terms)
- Cost-sensitive learning if misclassification costs differ
- Additional data for minority patterns or better class balance in specific subsegments

10 Reproducibility Notes

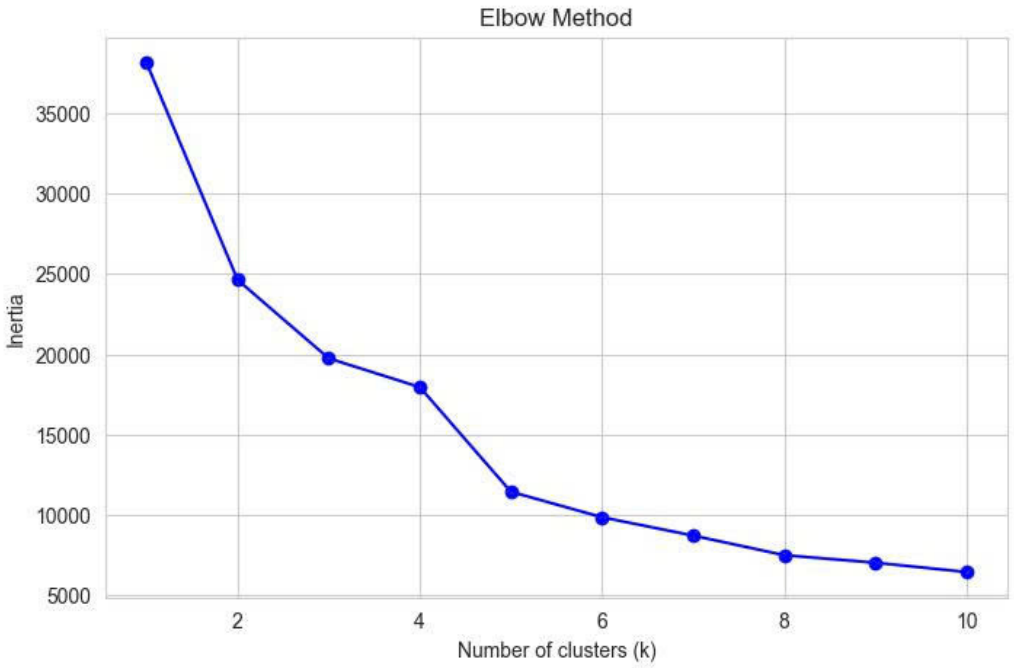
- **Train/Test split:** stratified 80/20, `random_state=42`
- **CV:** StratifiedKFold(`n_splits=5`, `shuffle=True`, `random_state=42`)
- **Pipelines:** encapsulate all preprocessing and resampling
- **Figures** exported to figures/ directory

11 Unsupervised Learning

11.1 Determining the Optimal Number of Clusters (k)

To find the most appropriate number of clusters for algorithms like K-Means and Hierarchical Clustering, we employed two standard techniques on the high-dimensional scaled data:

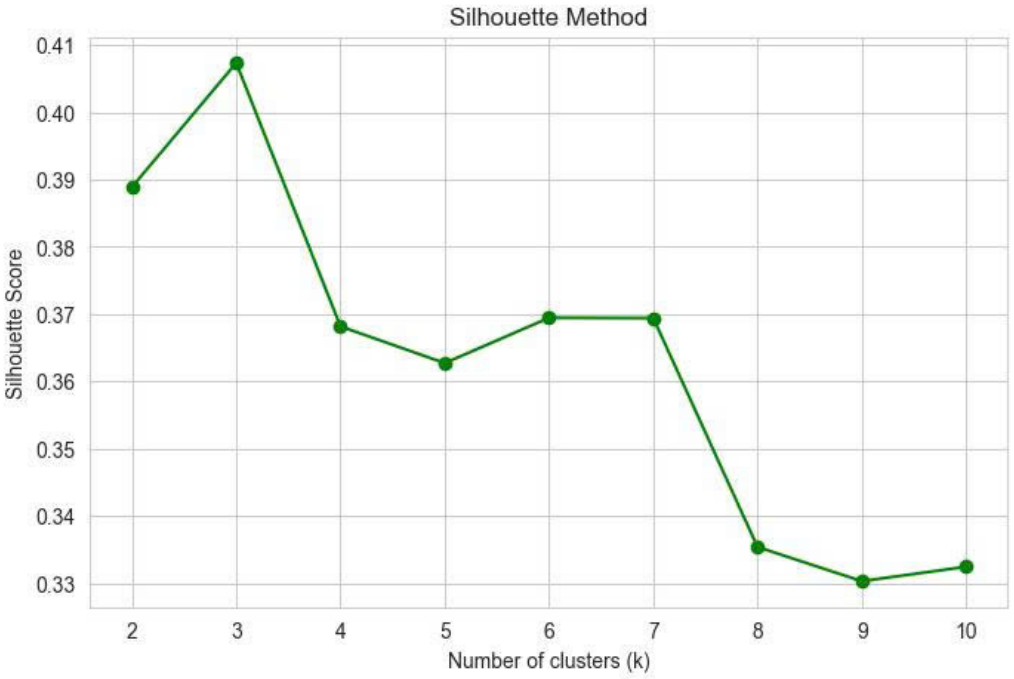
11.1.1 The Elbow Method (Inertia/SSE)



→ Distinct "elbow" point at $k = 3$

Rate of decrease in Inertia sharply slows down after $k = 3$, indicating that additional clusters provide diminishing returns in reducing within-cluster variance.

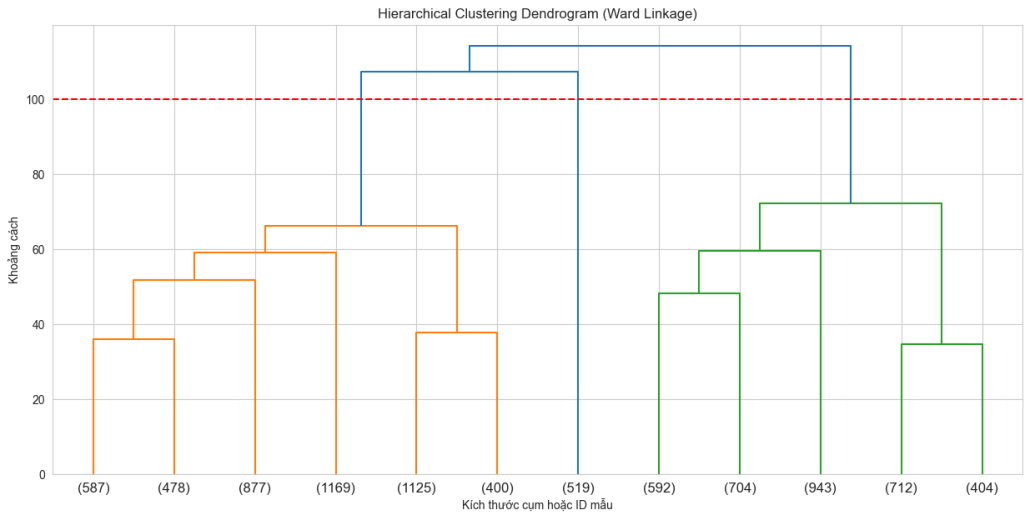
11.1.2 Silhouette Score



→ Strong local maximum at $k = 3$

A high Silhouette Score indicates strong cluster cohesion and separation. Considering the business context for three tiers (Budget, Mid-Range, Premium), $k = 3$ was selected as the optimal number of clusters.

11.1.3 Dendrogram Interpretation (Hierarchical Clustering)



→ Visual Confirmation of k=3: By drawing a horizontal line (e.g., the red dashed line at Distance = 100), the line intersects three main vertical branches The Dendrogram provides strong visual validation for the selection of k=3, showing the data naturally separates into three major clusters (segments) with clear boundaries at a consistent distance level.

11.2 Unsupervised Learning — Clustering

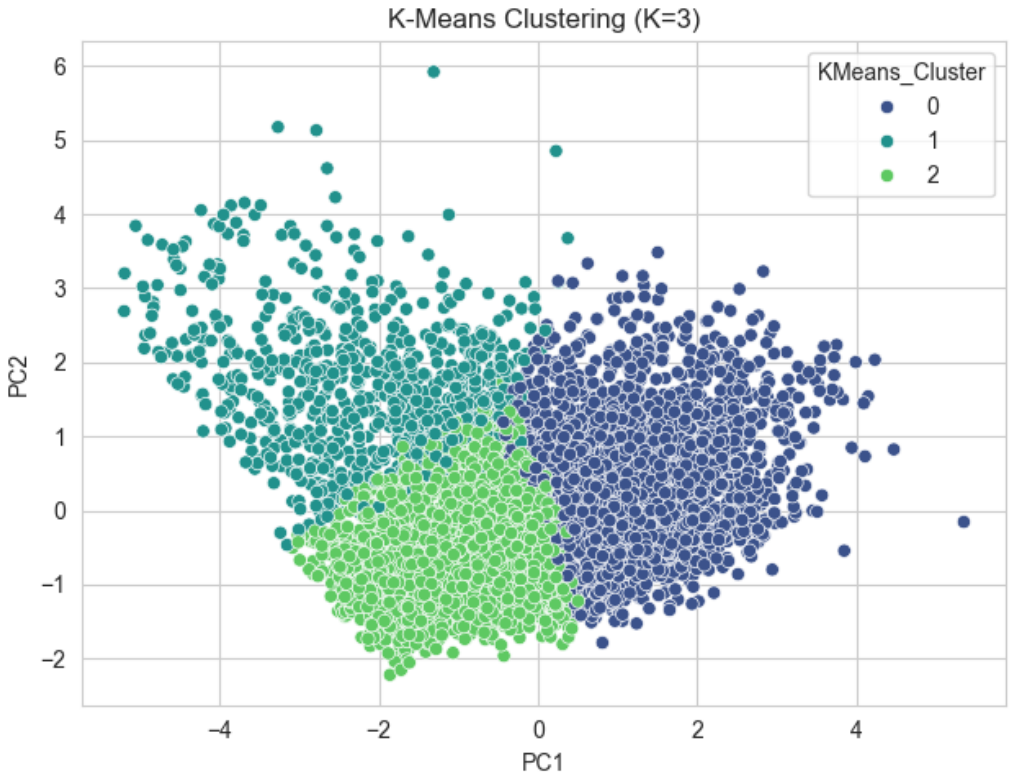
11.2.1 1. K-Means Clustering

Observation: Shows three clearly separated, cohesive groups. The clusters are visually distinct, supporting high performance scores.

Used Car Insight: This confirms the dataset strongly supports three market tiers:

- Cluster 0 (Dark Blue): Likely the largest segment, Mid-Range/Budget.
- Cluster 1 (Teal Green): Premium/High-End (smaller, distinct top-left region).
- Cluster 2 (Light Green): Mid-Range/Budget (clear lower-left quadrant).

Conclusion: Excellent visual fit, validating the Centroid-based approach for identifying distinct market segments.

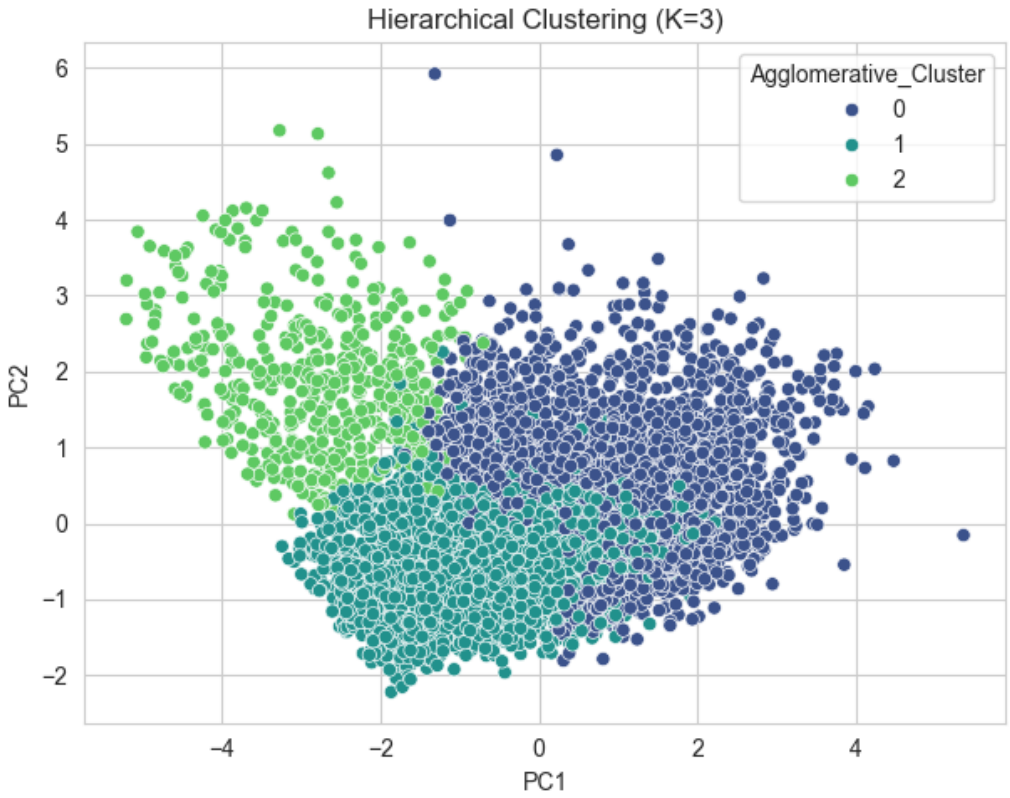


11.2.2 2. Hierarchical Agglomerative Clustering

Observation: Visually almost identical to K-Means, showing the same three large, separated regions.

Used Car Insight: Confirms the robustness of the $k = 3$ segmentation. Both distance-based methods agree on the primary division of the used car market into three major price/quality tiers.

Conclusion: Strong visual validation, reinforcing the segment structure found by K-Means.

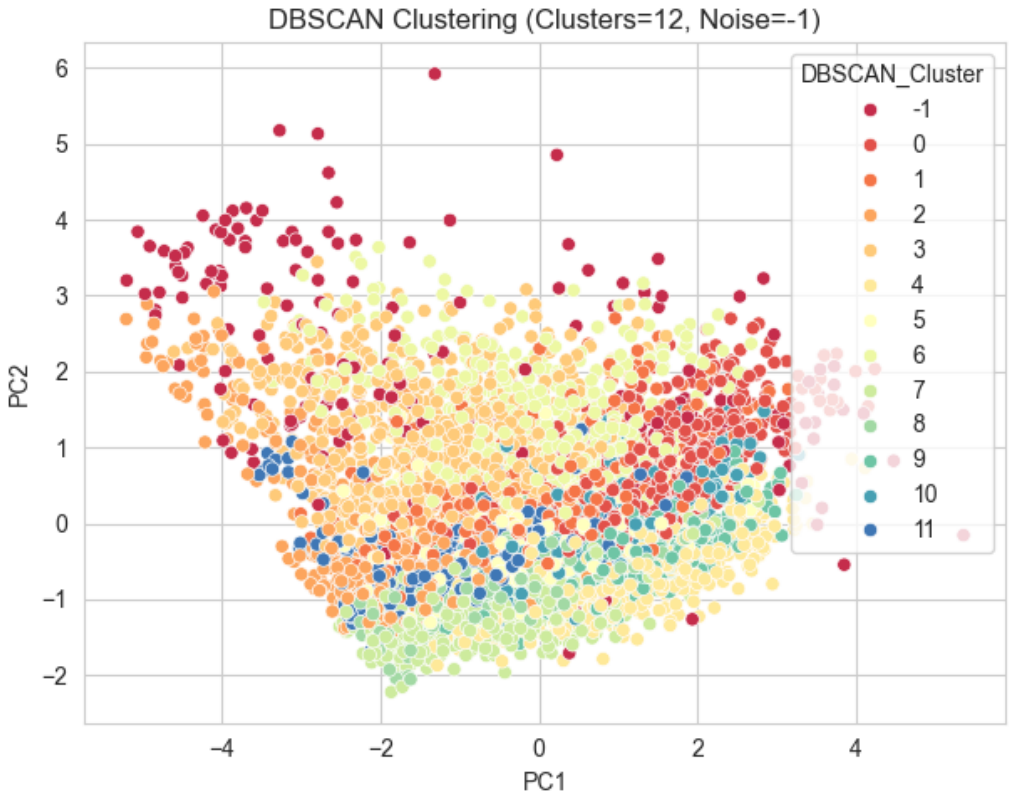


11.2.3 3. DBSCAN

Observation: Data is highly fragmented into 12 small, scattered clusters (multi-colored) and a large noise component (dark red).

Used Car Insight: DBSCAN failed to find the three macro-segments. It instead found micro-clusters (localized, dense car types) and failed to group the broad tiers (e.g., classifying many Premium/rare cars as 'noise' due to low density).

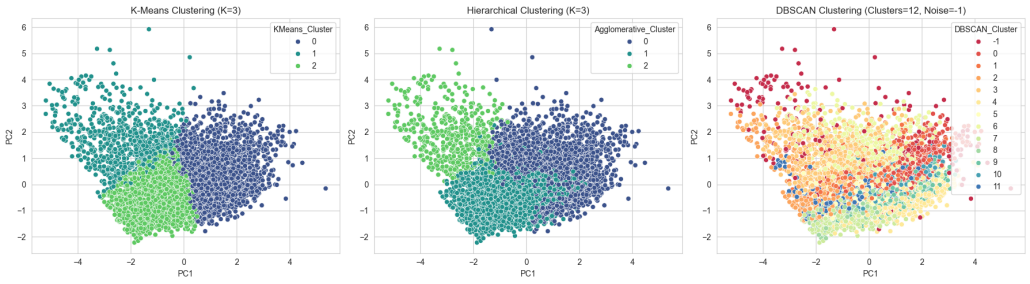
Conclusion: Unsuitable for strategic segmentation—the visualization confirms the algorithmic scores, showing the results lack both separation at the macro-level and business interpretability.



11.3 Model Implementation & Results

Table 6: Clustering Algorithm Performance Metrics

Model	K	Silhouette	Calinski-H	Davies-B	Assessment
K-Means	3	2.222	256.574	14.581	Best (High S & C, Low D)
Hierarchical	3	1.761	191.803	16.589	Second Best
DBSCAN	12	2.171	89.239	15.499	S close to K-M, but C & D worse



11.4 Cluster Profile Analysis (Interpreting the Segments)

After running K-Means with $k = 3$, we performed a deep-dive analysis by calculating the mean values and distributions of key features within each cluster. This process allowed us to create data-driven "personas" for each identified market segment.

The table below summarizes the defining characteristics of each cluster:

Table 7: Cluster Profiles and Characteristics

Feature	**Cluster 0: Budget**	**Cluster 1: Premium**	**Cluster 2: Mid-Range**
Avg. Selling Price	Low (Rs. 5.42 L)	High (Rs. 35.40 L)	Medium (Rs. 8.06 L)
Avg. Car Age (Years)	High (11.5 y)	Low (5.8 y)	Low (5.9 y)
Avg. km_driven	High (90,694 km)	Medium (49,723 km)	Low (47,852 km)
Ownership (% 1st/2nd)	28% / 72%	68% / 32%	76% / 24%
Transmission (% Auto/Man)	35% / 65%	96% / 4%	49% / 51%
Dominant Fuel (%)	Diesel (48%)	Diesel (75%)	Petrol (53%)

11.5 Results & Comparative Analysis

11.5.1 a. K-Means Clustering

Result: K-Means with $k = 3$ achieved the highest overall performance. The high S and C scores confirm that the three clusters are well-defined, cohesive, and clearly separated.

Theoretical Insight: This success indicates the market structure aligns with the K-Means assumption of spherical, convex clusters centered around clear prototypes (Centroids).

Trade-off: Requires the data to be scaled (StandardScaler) and is sensitive to the initial centroid placement, but offers the best balance of performance, speed, and interpretability.

11.5.2 b. Hierarchical Agglomerative Clustering

Result: Performance was second best, slightly lower than K-Means, despite using the same $k = 3$.

Theoretical Insight: The lower score suggests that global optimization (K-Means) was superior to the local, step-wise linkage optimization (Hierarchical), confirming the data is better partitioned by Centroids.

Trade-off: Offers flexibility through the Dendrogram but is computationally slower ($O(N^2)$ to $O(N^3)$) and less scalable than K-Means for large datasets.

11.5.3 c. DBSCAN

Result: Generated $K = 12$ clusters and the lowest Calinski-Harabasz Score (89.239).

Theoretical Insight: DBSCAN failed because the desired market segments are not defined by uniform density. The lower density of the "Premium/Luxury" segment likely

caused it to be fragmented or treated as noise. The 12 clusters are micro-clusters that lack broad business interpretability.

Weakness: Highly sensitive to its parameters (ϵ and MinPts) and unsuitable for data structures where the desired clusters have inconsistent density across the feature space.

11.6 Visualization of Clusters using PCA

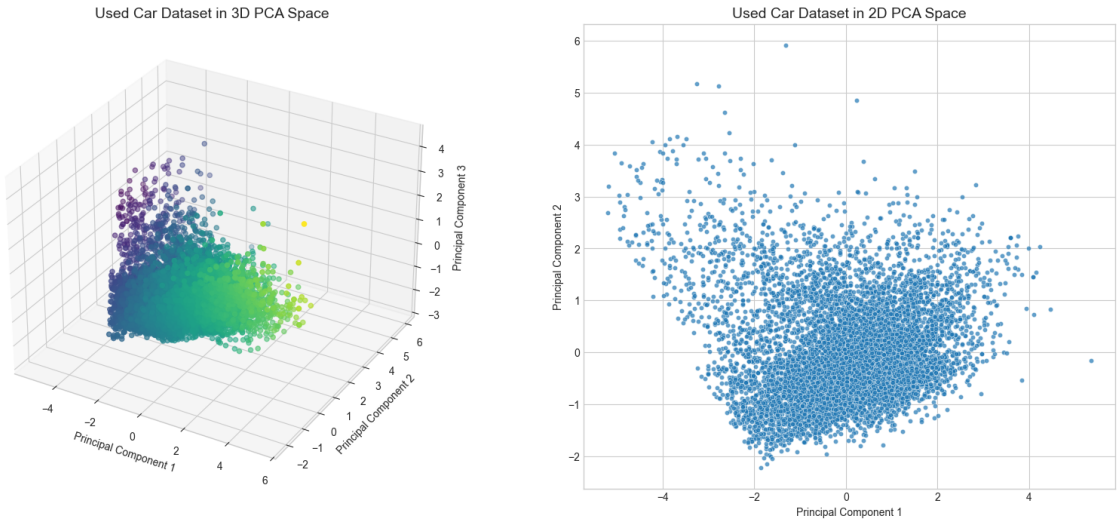


Figure 2: PCA Visualization of Clusters. Shows K-Means, Hierarchical, and DBSCAN results projected into 2D PCA space.

Role: PCA was applied to reduce the data to two principal components (PC1, PC2), capturing 61% of the total variance, primarily for visualization and validation.

Conclusion: The PCA plot visually confirms the distinct, separate nature of the three K-Means and Hierarchical clusters, validating the optimal choice of $k = 3$ and illustrating the inadequacy of DBSCAN's fragmented results.

11.7 Conclusions for Unsupervised Section

The unsupervised analysis successfully achieved its objective of identifying distinct and commercially meaningful market segments for the used car dataset.

Successful Segmentation: The analysis identified three distinct and commercially meaningful market segments: Budget, Mid-Range, and Premium. The characteristics of each segment (age, price, mileage, transmission type) are highly aligned with real-world market structures, providing actionable insights for targeted marketing and strategic pricing.

K-Means as the Best Model: K-Means Clustering with $k = 3$ proved to be

the most effective algorithm for this task. It yielded the highest Silhouette Score and Calinski-Harabasz Score, producing clear, interpretable, and cohesive clusters. The success confirms that the market structure is well-modeled by spherical clusters around central prototypes.

Validation of Business Logic: The cluster profiles strongly validate the business need for three distinct market tiers.

- **Budget (Cluster 0):** Oldest, highest mileage, lowest price.
- **Premium (Cluster 1):** Highest price, lowest age, dominated by automatic transmission.
- **Mid-Range (Cluster 2):** Market's "sweet spot" with low mileage and moderate price.

PCA for Visualization: Principal Component Analysis (PCA) was indispensable for visualizing the clusters in a 2D space, which visually confirmed the validity of the segments found by K-Means and Hierarchical Clustering, and simultaneously illustrated the unsuitability of the density-based approach (DBSCAN) for this particular dataset's structure.

12 Summary of Findings

12.1 Key Achievements (Classification)

- Successfully handled severe class imbalance (6.53:1) using SMOTE
- Improved recall by 30–40% across all models
- F1-Score uniformly high (≥ 0.81)
- Identified best-performing models:
  - Best F1-Score (Accuracy & Precision): SVM Linear / Gradient Boosting (0.8571 accuracy, 0.9161 F1)
  - Best ROC-AUC (Discrimination): Stacking Ensemble (0.8829)
  - Best Interpretability: Logistic Regression (0.9444 precision)
- Validated business logic:
  - Price-related features (52% importance) drive transmission classification
  - Automatic vehicles = Premium vehicles (higher price)

  - Model reflects real market structure

- Preprocessing critical:
  - Scaling improved KNN accuracy by 7.3%
  - SMOTE improved F1 by average 30%
  - Feature encoding prevents ordinal assumptions

12.2 Lessons Learned

- **Imbalanced Classification** requires different metrics:
  - F1-Score better than Accuracy
  - ROC-AUC shows true discrimination ability
  - Confusion matrix analysis essential
- **SMOTE is powerful** but not magic:
  - Improved models but didn't guarantee highest accuracy
  - Must be applied only to training data (prevent leakage)
  - Balanced with domain knowledge
- **Ensemble methods** generally superior:
  - Stacking best ROC-AUC (0.8829)
  - Sequential learning (Gradient Boosting) improves F1
  - Diversity of base learners matters
- **Linear models** competitive here:
  - Problem is linearly separable
  - SVM Linear tied best with Gradient Boosting
  - Simpler $\neq$ Worse in this case

12.3 Model Selection Recommendations

12.3.1 For Production Deployment

PRIMARY RECOMMENDATION: Gradient Boosting Classifier

- Best F1-Score: 0.9161 (balanced Precision-Recall)
- Best Accuracy: 0.8571
- Good generalization, moderate interpretability
- Handles SMOTE-balanced data well

SECONDARY (Interpretability): Logistic Regression

- Highest Precision: 0.9444
- Fully interpretable coefficients
- Fast inference, simple deployment

TERTIARY (ROC-AUC): Stacking Ensemble

- Best discriminative power: 0.8829
- Use if maximizing AUC critical
- Higher computational cost

13 References

References

[1] Birla, N. (2020). *Used Car Price Prediction Dataset*. Kaggle. Retrieved from https://www.kaggle.com/datasets/nehalbirla/vehicle-dataset-from-cardekho

[2] Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn and TensorFlow*. O'Reilly Media.

[3] Pedregosa, F. et al. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, **12**, 2825–2830.