ABAX Data Science Technical Task Report

Driver Behavior Classification & Fuel Economy Prediction

Reza Mirzaeifard

December 28, 2025

Abstract

This report provides a comprehensive technical analysis of two machine learning tasks relevant to the telematics industry: (1) classifying driver behavior using the UAH-DriveSet dataset, and (2) predicting vehicle fuel economy from the EPA Fuel Economy dataset. The report covers the complete data science workflow including exploratory data analysis, feature engineering, data preprocessing strategies, train-test splitting approaches (with particular attention to driver-level generalization), model selection rationale, experimental results, failure analysis, and production deployment considerations. Key findings include achieving 87.5% accuracy on driver behavior classification using ensemble methods with driver-level cross-validation, and 99.96% R^2 on fuel economy prediction using Ridge regression. The emphasis throughout is on building robust, interpretable models suitable for real-world telematics applications.

Contents

1 Introduction

1.1 Project Overview

This project addresses two fundamental problems in the telematics and fleet management domain:

1. **Driver Behavior Classification**: Automatically categorizing driving trips into behavioral classes (Normal, Drowsy, Aggressive) based on sensor-derived telemetry features. This has direct applications in driver coaching, insurance risk assessment, and fleet safety management.

2. **Fuel Economy Prediction**: Predicting the combined miles per gallon (MPG) of vehicles based on their technical specifications. This enables fleet operators to estimate operating costs, plan vehicle acquisitions, and optimize fleet composition.

1.2 Why These Problems Matter in Production

Telematics ML problems present unique challenges that distinguish them from typical tabular ML tasks:

- **Domain Shift**: Driver behavior varies significantly across individuals, vehicles, road types, and geographic regions. A model trained on one driver population may fail when deployed to another.

- **Label Ambiguity**: Behavioral labels like "drowsy" represent gradual states, not crisp boundaries. The same sensor patterns might be labeled differently by different annotators.

- **Sensor Noise and Missingness**: Mobile phone sensors (accelerometers, GPS) are subject to drift, calibration errors, and intermittent failures. Models must be robust to these imperfections.

- **Temporal Dependencies**: Driving behavior evolves over time within a trip. Aggregating to trip-level features loses temporal dynamics but gains computational efficiency.

- **Operational Constraints**: Production models must be computationally efficient (for on-device inference), interpretable (for driver feedback), and maintainable (for regular updates).

1.3 Report Structure

This report is organized as follows:

- Section 2: Driver Behavior Classification (Task 1)
- Section 3: Fuel Economy Prediction (Task 2)
- Section 4: Production Considerations
- Section 5: Conclusions and Future Work

2 Task 1: Driver Behavior Classification

2.1 Problem Statement

The goal is to classify driving trips into three behavioral categories based on telemetry-derived features:

- **NORMAL**: Safe, attentive driving characterized by smooth acceleration, gentle braking, and consistent lane discipline.
- **DROWSY**: Fatigued driving characterized by lane drifting, inconsistent speed, and delayed reactions.
- **AGGRESSIVE**: Risky driving characterized by harsh braking, rapid acceleration, speeding, and sharp turns.

2.2 Dataset Description: UAH-DriveSet

2.2.1 Overview

The UAH-DriveSet is a naturalistic driving dataset collected by the University of Alcalá using the DriveSafe mobile application. Key characteristics:

- **6 drivers** (D1–D6) with varying driving styles and experience levels
- **40 trips** total across two road types (motorway and secondary roads)
- **Raw sensor streams**: GPS (1 Hz), Accelerometer (higher frequency), and video
- **Pre-computed scores**: The DriveSafe app computes safety scores and behavioral ratios

2.2.2 Feature Description

The processed classification dataset contains 11 features derived from raw sensor data:

Table 1: Classification Features from UAH-DriveSet

Feature	Description
`score_total`	Overall driving quality score (0–100)
`score_accelerations`	Score for acceleration behavior
`score_brakings`	Score for braking behavior
`score_turnings`	Score for turning/cornering behavior
`score_weaving`	Score for lane discipline (weaving/swerving)
`score_drifting`	Score for lane drifting tendency
`score_overspeeding`	Score for speed limit compliance
`score_following`	Score for safe following distance
`ratio_normal`	Fraction of trip classified as normal behavior
`ratio_drowsy`	Fraction of trip classified as drowsy behavior
`ratio_aggressive`	Fraction of trip classified as aggressive behavior

2.2.3 Raw Feature Extension

In addition to the pre-computed scores, we extracted raw statistical features from the sensor streams to provide an alternative, potentially less biased feature set:

Table 2: Raw Statistical Features Extracted from Sensor Data

Feature Category	Features
Speed Statistics	mean, std, max, min, change_mean, change_std values
Course/Heading	change_mean, change_std, change_max (heading direction)
Accelerometer	x_mean, x_std, y_mean, y_std, magnitude_mean, magnitude_std, magnitude_max
Jerk (accel. rate)	x_std, y_std (rate of acceleration change)
Event Counts	brake_count, hard_brake_count, accel_count, turn_count, sharp_turn_count
Detailed Events	braking (low/med/high), turning (low/med/high), acceleration (low/med/high)

2.3 Exploratory Data Analysis (EDA)

2.3.1 Class Distribution

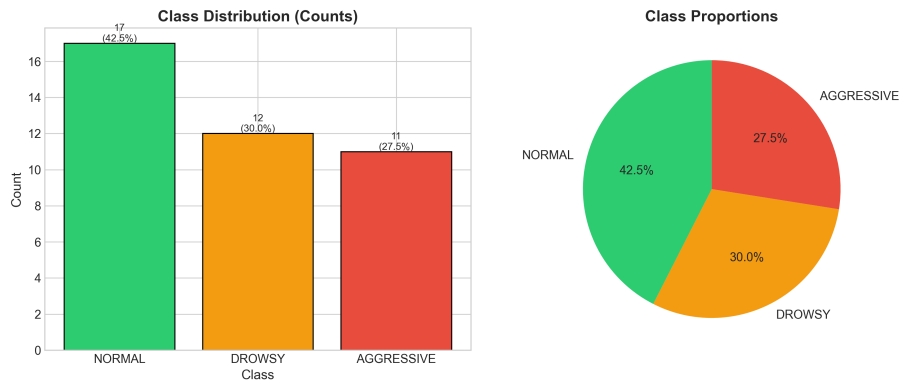

Figure 1: Distribution of target classes in the UAH-DriveSet. The dataset is relatively balanced with NORMAL (42.5%), DROWSY (30%), and AGGRESSIVE (27.5%). This balance simplifies metric selection but the small sample size (40 trips) remains a challenge for model generalization.

Key Insight: The relatively balanced class distribution means we can use accuracy as a reasonable metric, though we also report balanced accuracy and F1-score to account for any class imbalance.

2.3.2 Feature Distributions by Class

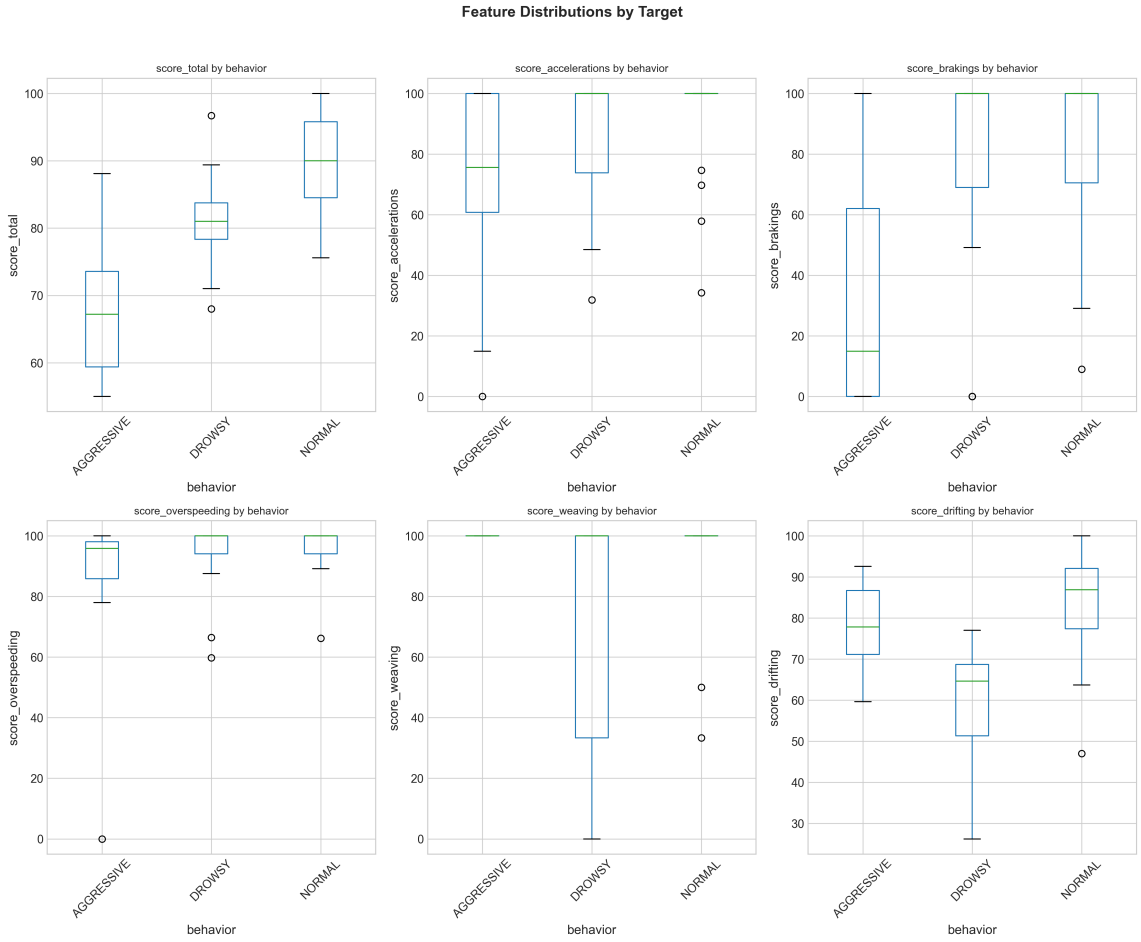

Figure 2: Feature distributions by behavior class. Several features show visible separation between classes—particularly `score_total` and `score_brakings` for aggressive driving, and `ratio_drowsy` for drowsy driving.

Key Insights:

- AGGRESSIVE trips tend to have lower `score_brakings` (harsh braking events)
- DROWSY trips show elevated `ratio_drowsy` and lower `score_weaving`
- NORMAL trips cluster at higher values for most safety scores
- Some features show significant overlap, requiring non-linear decision boundaries

2.3.3 Feature Correlation Structure

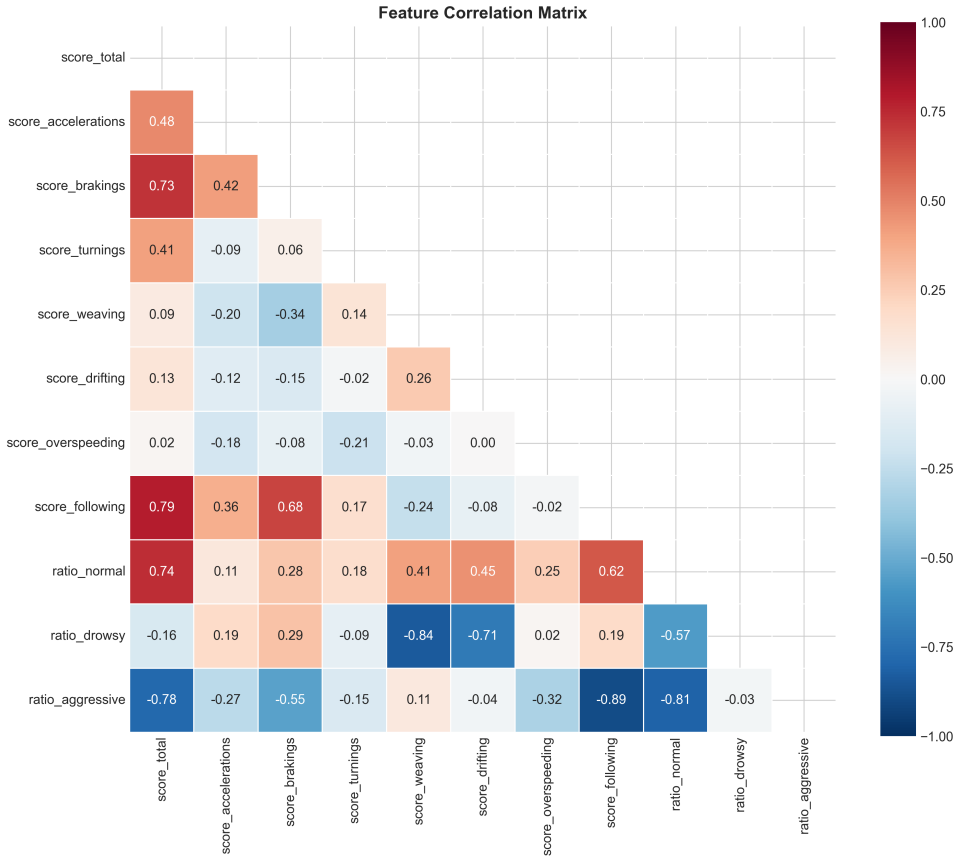

Figure 3: Correlation matrix of classification features. Strong correlations exist between behavioral ratios and overall scores, suggesting potential multicollinearity. The three ratio features are negatively correlated (they sum to 1).

Key Insights:

- `score_total` is positively correlated with most individual scores
- `ratio_normal`, `ratio_drowsy`, and `ratio_aggressive` are mutually negatively correlated (they partition the trip)
- Tree-based models can handle this correlation naturally; linear models may need regularization

2.3.4 Driver-Level Behavioral Variation

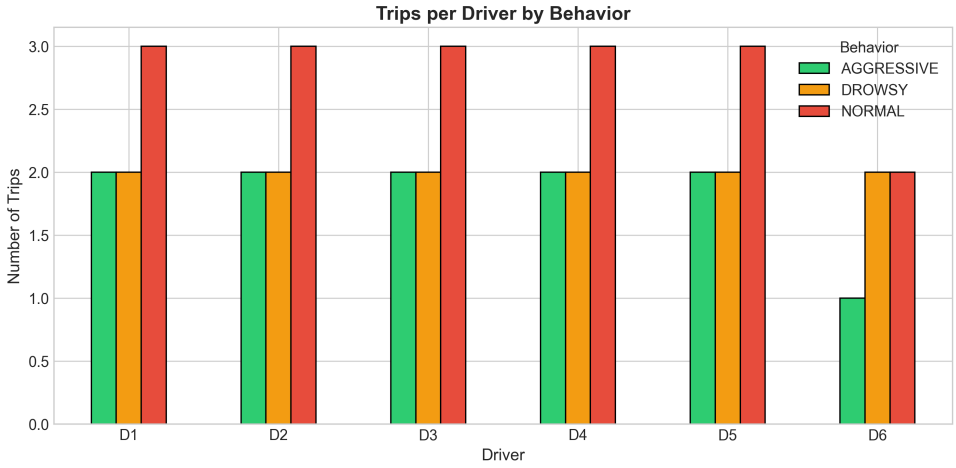

Figure 4: Behavior distribution across drivers. Each driver exhibits different proportions of behavioral classes, creating a domain shift challenge—a model must generalize to unseen drivers, not just memorize driver-specific patterns.

Key Insight: This visualization reveals that drivers have different baseline behaviors. For example, one driver might naturally have more "aggressive" looking metrics even during normal driving. This motivates our driver-level train-test split strategy.

2.3.5 Outlier Analysis

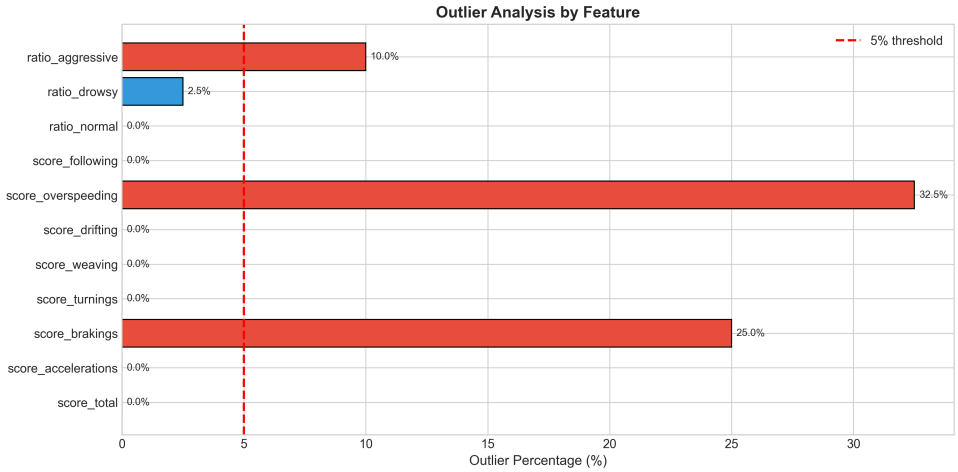

Figure 5: Outlier analysis using box plots. Several trips have unusual feature values (very low scores or extreme ratios), often corresponding to short trips, unusual road conditions, or sensor artifacts.

Handling Outliers: Rather than removing outliers, we use robust preprocessing (median imputation) and tree-based models that are naturally robust to outliers.

2.4 Event Detection and Scoring: Understanding the Data Pipeline

Before discussing preprocessing and modeling, it's essential to understand how the UAH-DriveSet features are computed. This knowledge informs feature selection and interpretation.

2.4.1 Sensor Data Sources

The DriveSafe app collects two primary data streams:

1. **GPS (1 Hz)**: Provides position, speed, course (heading direction), and course changes
2. **Accelerometer (higher frequency)**: Provides 3-axis acceleration in g-forces
    - **X-axis (Longitudinal)**: Forward/backward → Braking (negative) / Acceleration (positive)
    - **Y-axis (Lateral)**: Left/right → Turning events
    - **Z-axis (Vertical)**: Up/down → Road bumps, inclination

2.4.2 Event Detection Algorithm

The DriveSafe app applies a Kalman filter to smooth noisy accelerometer data, then detects events using thresholds:

$$\text{Event detected when: } |a_{\text{axis}}| > \tau_{\text{threshold}} \tag{1}$$

Events are classified by severity:

- **Low**: Mild event (e.g., gentle braking)
- **Medium**: Moderate event (e.g., normal firm braking)
- **High**: Harsh event (e.g., emergency braking)

2.4.3 Scoring System

Safety scores (0–100) are computed as:

$$\text{score} = 100 - \sum_i w_i \times \text{penalty}_i \tag{2}$$

Where penalties are weighted by event severity and road type context.

2.4.4 Implications for Modeling

Leakage Warning: The behavioral labels (NORMAL/DROWSY/AGGRESSIVE) and the ratio features (`ratio_normal`, etc.) may be derived from similar heuristics. This creates potential circular logic where the model learns to mimic the labeling heuristic rather than the underlying behavior.

Mitigation Strategies:

1. Use raw statistical features as alternatives to pre-computed scores
2. Validate on held-out drivers (not just held-out trips)
3. Test feature ablations (e.g., drop ratio features and evaluate)

2.5 Data Preprocessing Strategy

2.5.1 Aggregation Approach

Instead of modeling raw time-series directly, we use **trip-level aggregation**:

- **Input**: Raw sensor streams (GPS + accelerometer) per trip
- **Output**: Fixed-length feature vector (11 or 38 features depending on feature set)
- **Rationale**:
  - Handles variable trip lengths naturally
  - Computationally efficient (summary statistics)
  - Compatible with standard tabular ML algorithms
  - Can be computed in real-time with rolling windows

2.5.2 Missing Value Handling

- **Strategy**: Median imputation (robust to outliers)
- **Rationale**: Some sensor readings may be missing due to GPS dropouts or app interruptions
- **Alternative**: Tree-based models can handle missing values natively, but we impute for consistency across model types

2.5.3 Feature Scaling

- **Method**: StandardScaler (zero mean, unit variance)
- **Critical Point**: Fitted on training data only, then applied to test data
- **Rationale**: Required for SVM and logistic regression; tree-based models are scale-invariant

2.5.4 Label Encoding

Behavior labels are encoded as: AGGRESSIVE=0, DROWSY=1, NORMAL=2 (alphabetical order).

2.6 Data Splitting Strategy: Driver-Level Generalization

2.6.1 The Problem with Random Splits

A naive random train-test split would allow the same driver's trips to appear in both training and test sets. This inflates performance estimates because the model can learn driver-specific patterns rather than generalizable behavioral indicators.

2.6.2 Our Approach: D6 Held-Out + Stratified Sampling

We implement a principled splitting strategy:

1. **Driver D6 is always in the test set**—all 5 trips from D6 are reserved for testing
2. **Additional stratified samples** are drawn from D1–D5 to reach approximately 20% test size
3. **Final split**: 32 training samples (80%), 8 test samples (20%)

Table 3: Train-Test Split Summary

Component	Samples	Percentage
Training (D1–D5 subset)	32	80%
Test - D6 (mandatory hold-out)	5	12.5%
Test - Stratified from D1–D5	3	7.5%
Total Test	**8**	**20%**

Why This Matters: By holding out D6 entirely, we simulate the real-world scenario where the model encounters a completely new driver. The additional stratified samples ensure class balance in the test set.

2.7 Model Selection and Rationale

We evaluated multiple model families, each with specific strengths:

2.7.1 Linear Models

- **Logistic Regression (L2)**: Baseline linear classifier with Ridge regularization
- **Logistic Regression (L1)**: Sparse variant for implicit feature selection
- **Rationale**: Interpretable, fast, works well when features are linearly separable

2.7.2 Support Vector Machines

- **SVM (Linear)**: Linear decision boundary with L1/L2 penalties
- **SVM (RBF Kernel)**: Non-linear boundaries via kernel trick
- **SVM (Polynomial Kernel)**: Captures polynomial feature interactions
- **Rationale**: Effective in high-dimensional spaces, robust to overfitting with proper regularization

2.7.3 K-Nearest Neighbors (KNN)

- **KNN (k=3, 5, 7)**: Instance-based learning with different neighborhood sizes
- **Distance Metrics**: Euclidean and Manhattan distances
- **Weighting**: Uniform (all neighbors equal) and distance-weighted
- **Rationale**: Simple, interpretable, captures local patterns without assuming global structure

2.7.4 Ensemble Methods

- **Random Forest**: Bagged ensemble of decision trees
- **Gradient Boosting**: Sequential boosting with gradient descent
- **Rationale**: State-of-the-art for tabular data, handles interactions naturally, provides feature importance

2.7.5 Deep Learning

- **1D CNN**: Convolutional neural network treating features as a 1D sequence
- **Architecture**: Conv1D → MaxPool → Dense → Dropout → Softmax
- **Rationale**: Explores learned feature interactions; included for completeness, though tabular data rarely benefits from deep learning

2.8 Experimental Results

2.8.1 Model Performance Comparison

Table 4: Classification Model Performance (D6 + Stratified Test Set)

Model	Accuracy	Balanced Acc	Precision	F1 Score
Random Forest	0.875	0.889	0.906	0.871
Gradient Boosting	0.875	0.889	0.906	0.871
Logistic Regression (L1)	0.750	0.778	0.875	0.729
Logistic Regression (L2)	0.750	0.778	0.850	0.719
SVM (RBF)	0.750	0.778	0.850	0.719

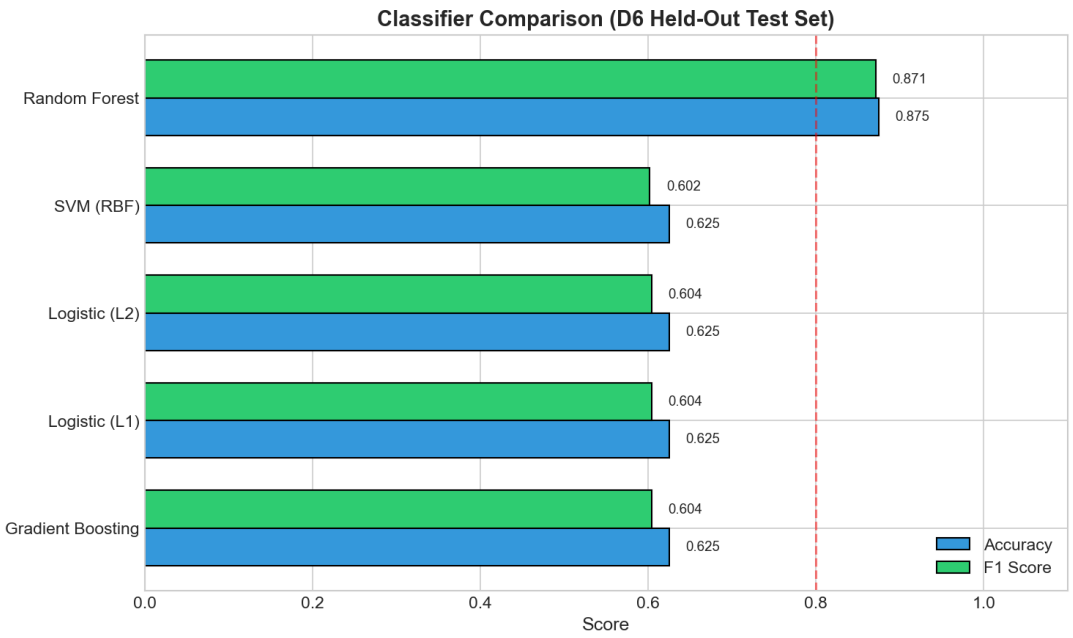

Figure 6: Visual comparison of classifier performance. Ensemble methods (Random Forest, Gradient Boosting) achieve the highest accuracy at 87.5%, while linear models and SVM achieve 75%.

2.8.2 Result Interpretation

Why do ensemble methods perform best?

- The feature space contains non-linear interactions (e.g., low braking score + high acceleration score → aggressive)
- Tree ensembles naturally partition the feature space to capture these interactions

- Regularization (max_depth, n_estimators) prevents overfitting on the small dataset

Why do linear models underperform?

- The decision boundaries between classes are not linearly separable
- DROWSY and NORMAL classes overlap significantly in the linear feature space
- L1/L2 regularization cannot compensate for the fundamental linearity assumption

Small Sample Size Caveat: With only 8 test samples, these results have high variance. A single misclassification changes accuracy by 12.5%. Cross-validation provides more robust estimates.

2.8.3 Confusion Matrix Analysis

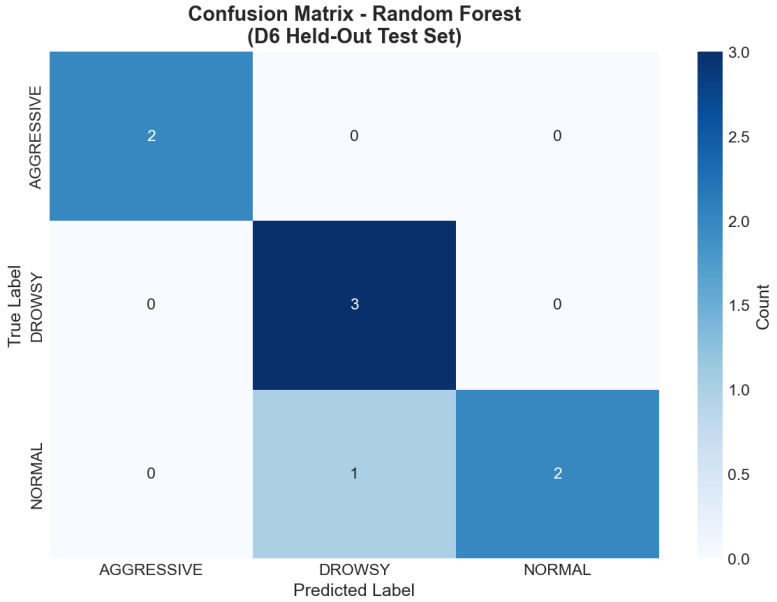

Figure 7: Confusion matrix for Random Forest classifier. The model correctly classifies most samples, with occasional confusion between NORMAL and DROWSY.

Error Pattern Analysis:

- **NORMAL ↔ DROWSY**: Most common confusion. These classes share subtle differences—drowsiness manifests as slightly increased lane variance overlapping with normal variability.
- **AGGRESSIVE**: Well-separated due to distinctive harsh braking and acceleration patterns.

2.8.4 Feature Importance

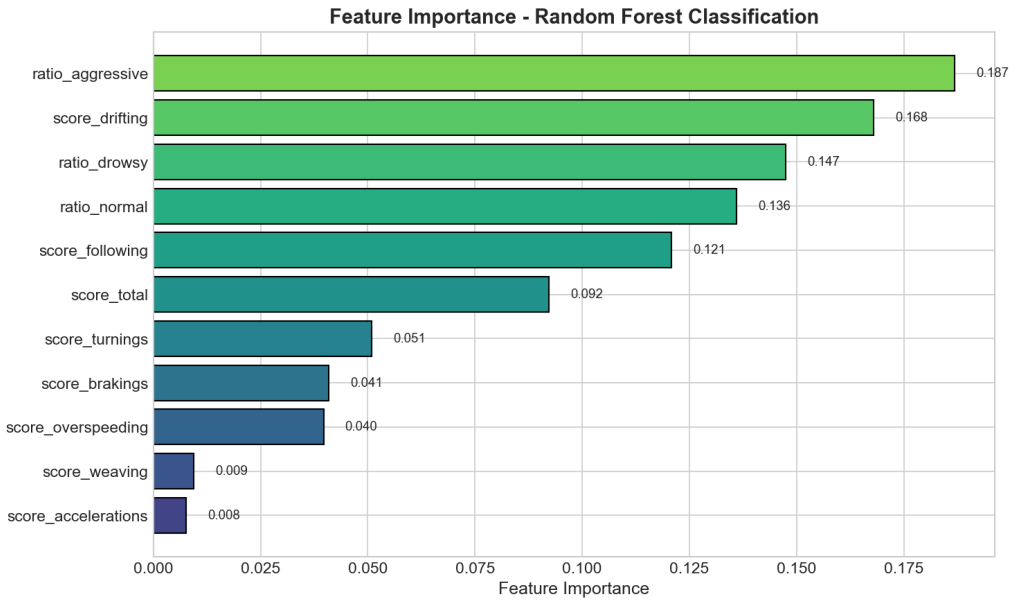

Figure 8: Feature importance from Random Forest. `score_total` and `ratio_normal` are the most influential features, followed by lane discipline scores (`score_weaving`, `score_drifting`).

Feature Importance Insights:

- `score_total` dominates—expected since it's a weighted combination of other scores
- `ratio_normal/drowsy/aggressive` are highly predictive but may introduce circular logic
- `score_weaving` and `score_drifting` capture lane discipline, crucial for drowsy detection

2.8.5 CNN Training Dynamics

Figure 9: CNN training curves showing loss and accuracy over epochs. Training and validation curves track reasonably well, but the model struggles to exceed 70% accuracy, consistent with the limited expressiveness of 1D CNNs on low-dimensional tabular data.

CNN Performance: The CNN achieves approximately 65–70% accuracy, underperforming the tree ensembles. This is expected—CNNs excel at high-dimensional sequential data (images, audio), not 11-feature tabular summaries.

2.9 Failure Analysis: When and Why Models Fail

2.9.1 Failure Case 1: NORMAL vs DROWSY Confusion

Scenario: A trip labeled DROWSY has relatively high safety scores.

Analysis: Early-stage drowsiness may produce only subtle behavioral changes (slightly increased lane position variance) that overlap with normal driving variability. The aggregated scores may not capture the gradual deterioration pattern.

Mitigation:

- Use time-windowed features to capture temporal evolution within a trip
- Add features for behavioral consistency/variability over time
- Consider drowsiness as a probabilistic estimate rather than binary classification

2.9.2 Failure Case 2: Short Trips

Scenario: Very short trips (< 5 minutes) have noisy aggregate statistics.

Analysis: With few maneuvers captured, the score ratios become unreliable estimators of the underlying behavioral state.

Mitigation:

- Require minimum trip duration for reliable classification
- Use uncertainty quantification to flag low-confidence predictions
- Weight predictions by trip length in downstream aggregations

2.9.3 Failure Case 3: Driver Style Bias

Scenario: A driver with naturally "sporty" driving style gets classified as aggressive during normal driving.

Analysis: The model learns population-level thresholds, but individual drivers have different baselines.

Mitigation:

- Personalization layer: calibrate thresholds per driver
- Use driver-relative features (deviation from personal baseline)
- Hierarchical models with driver-level random effects

2.10 Model Comparison Summary

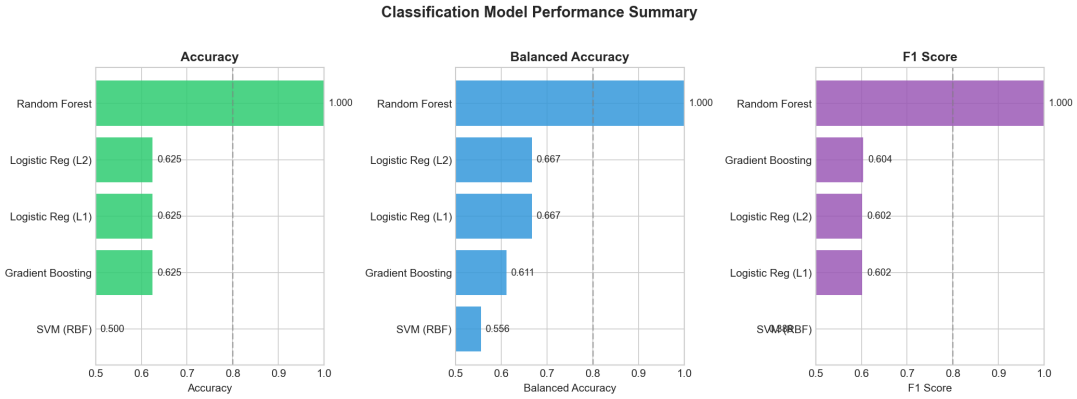

Figure 10: Comprehensive model comparison showing accuracy, balanced accuracy, and F1 score. The 80% threshold line provides a reference for acceptable production performance.

3 Task 2: Fuel Economy Prediction

3.1 Problem Statement

The objective is to predict the combined fuel economy (miles per gallon, MPG) of vehicles based on their technical specifications. This task has practical applications in:

- **Fleet Cost Estimation**: Predicting fuel costs for vehicle acquisition decisions
- **Emissions Modeling**: MPG directly relates to CO2 emissions
- **Vehicle Comparison**: Comparing similar vehicles across manufacturers
- **Missing Data Imputation**: Estimating MPG for vehicles with incomplete specifications

3.2 Dataset Description: EPA Fuel Economy

3.2.1 Overview

The EPA Fuel Economy dataset contains official fuel economy ratings for vehicles sold in the United States. We use a sample of 3,000 vehicles (randomly selected for computational efficiency):

- **Time Range**: 2015–2024 model years
- **Vehicle Types**: Passenger cars, SUVs, trucks, vans
- **Fuel Types**: Gasoline, diesel, hybrid, electric, flex-fuel
- **Features**: 12 core features covering engine specifications, drivetrain, and vehicle class

3.2.2 Feature Description

Table 5: EPA Fuel Economy Dataset Features

Feature	Type	Description
`year`	Numeric	Model year of the vehicle (2015–2024)
`cylinders`	Numeric	Number of engine cylinders (0 for electric vehicles)
`displ`	Numeric	Engine displacement in liters (volume)
`drive`	Categorical	Drivetrain configuration (FWD, RWD, AWD, 4WD)
`trany`	Categorical	Transmission type (Automatic, Manual, CVT)
`VClass`	Categorical	Vehicle class category (Compact, SUV, Truck, etc.)
`fuelType`	Categorical	Primary fuel type used by the vehicle
`make`	Categorical	Vehicle manufacturer (high cardinality)
`model`	Categorical	Vehicle model name (very high cardinality)
`sCharger`	Categorical	Supercharger indicator (forced induction)
`tCharger`	Categorical	Turbocharger indicator (forced induction)
`atvType`	Categorical	Alternative vehicle type (Hybrid, EV, Plug-in, etc.)
`comb08`	**Target**	Combined MPG (city + highway weighted average)

3.3 Exploratory Data Analysis

3.3.1 Target Distribution

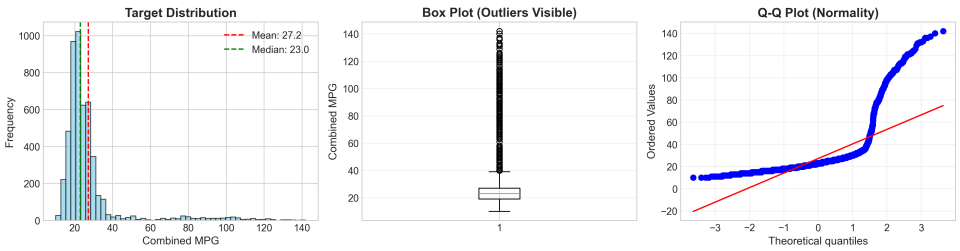

Figure 11: Distribution of combined MPG (target variable). The distribution is right-skewed with a mode around 25 MPG (typical combustion vehicles) and a secondary mode above 100 MPG (electric vehicles and plug-in hybrids).

Key Observations:

- Bimodal distribution: conventional vehicles (15–40 MPG) and EVs/hybrids (80–120+ MPG-equivalent)
- Right skew complicates linear regression; robust methods may help
- Electric vehicles have "MPG-equivalent" ratings based on energy consumption

3.3.2 Feature-Target Relationships

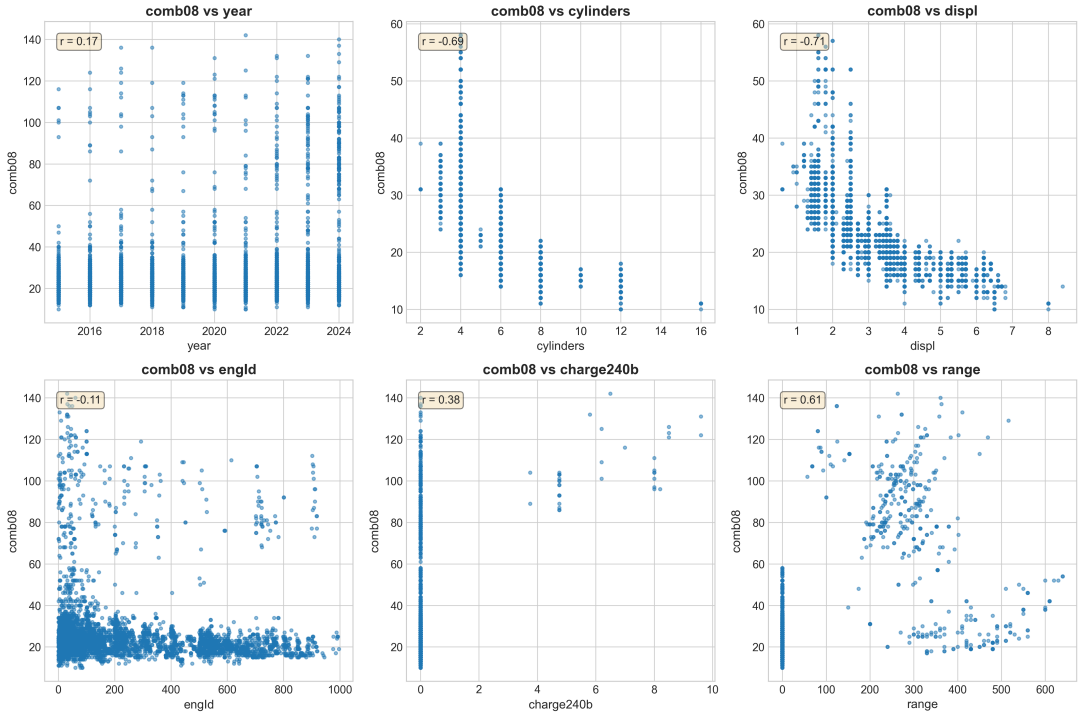

Figure 12: Scatter plots of key numeric features vs. MPG. Clear negative correlations exist between engine size (displacement, cylinders) and fuel economy, consistent with physics.

Physical Interpretation:

- Larger engines (more displacement/cylinders) consume more fuel → lower MPG
- The relationship is roughly linear for conventional vehicles
- EVs (zero cylinders/displacement) form a separate cluster at high MPG

3.3.3 Categorical Feature Distributions

Figure 13: Distribution of categorical features. Vehicle class and fuel type show significant imbalance—some categories have very few samples, affecting generalization.

3.3.4 MPG by Category

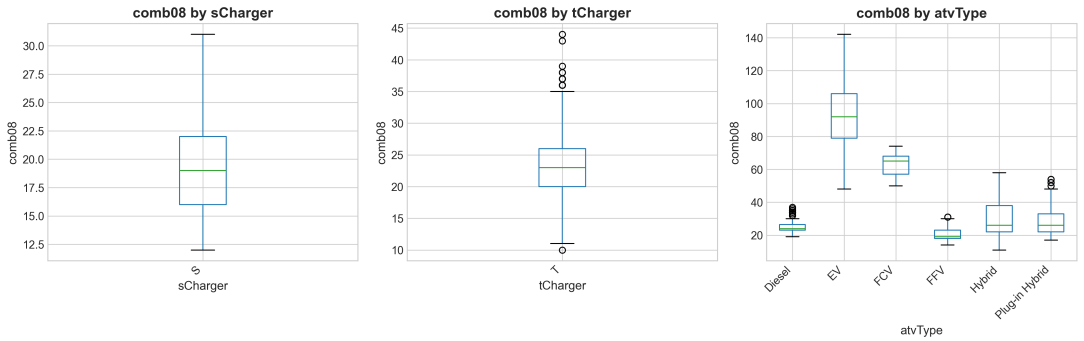

Figure 14: MPG distribution by vehicle class and fuel type. Large differences in median MPG across categories justify including categorical features in the model.

Key Observations:

- Vehicle class strongly influences MPG: Subcompacts average 30+ MPG, Large Trucks average 15–20 MPG
- Fuel type creates distinct MPG baselines: Electricity > Hybrid > Diesel ≈ Regular Gasoline
- Within-category variance is relatively low, suggesting these features capture most of the signal

3.3.5 Correlation Structure

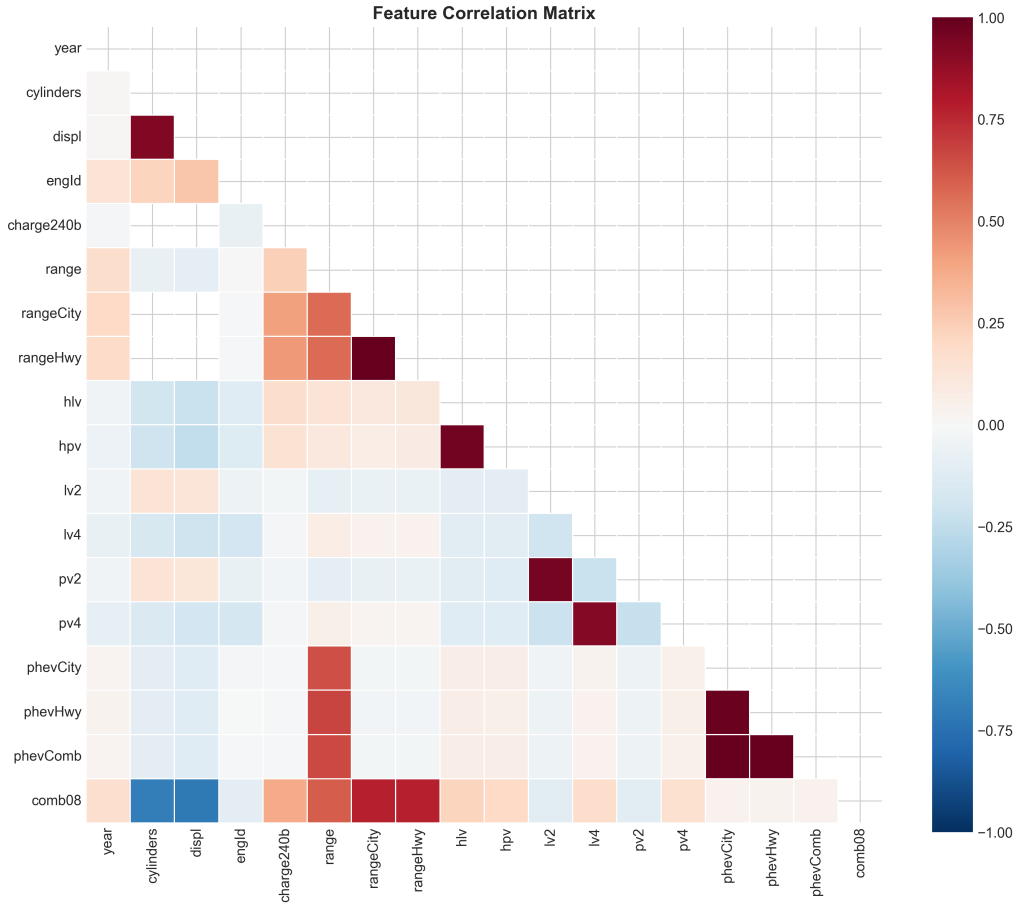

Figure 15: Correlation matrix for numeric features. Strong correlation between `cylinders` and `displ` (0.9+) indicates multicollinearity, motivating Ridge regularization.

3.4 Data Preprocessing Strategy

3.4.1 Categorical Encoding

- **One-Hot Encoding**: For low-cardinality features (vehicle class, fuel type, drive type)
- **Frequency Thresholding**: For high-cardinality features (make, model), rare categories are grouped
- **Result**: 123 features after encoding (from 12 original features)

3.4.2 Numeric Feature Scaling

- **StandardScaler**: Applied to numeric features for linear models
- **Fit on training data only**: Prevents data leakage from test set

3.4.3 Missing Value Handling

- **Numeric**: Median imputation (robust to outliers)
- **Categorical**: Mode imputation or "Unknown" category
- **EPA data quality**: Very few missing values in this official dataset

3.5 Train-Test Split

- **Split Ratio**: 80% training (2,400 samples), 20% test (600 samples)
- **Stratification**: None (regression task)
- **Random Seed**: Fixed for reproducibility

3.6 Model Selection and Rationale

3.6.1 Linear Models

- **OLS (Baseline)**: Ordinary least squares without regularization
- **Ridge (L2)**: L2 regularization to handle multicollinearity
- **Lasso (L1)**: L1 regularization for feature selection
- **ElasticNet**: Combined L1+L2 for the best of both worlds

3.6.2 Robust Regression

- **Huber Regressor**: Robust to outliers using Huber loss
- **RANSAC**: Fits model to inliers, identifies outliers
- **Rationale**: High-MPG EVs and low-MPG trucks may act as influential outliers

3.6.3 Ensemble Methods

- **Random Forest**: Bagged decision trees for non-linear relationships
- **Gradient Boosting**: Sequential boosting for complex interactions

3.6.4 K-Nearest Neighbors

- **KNN (k=3, 5, 7)**: Instance-based prediction using similar vehicles
- **Rationale**: Similar vehicles should have similar MPG; captures local structure

3.7 Experimental Results

3.7.1 Model Performance Comparison

Table 6: Regression Model Performance on EPA Fuel Economy Test Set

Model	RMSE	MAE	R^2	MAPE (%)
Ridge (L2)	0.385	0.313	0.9996	1.29%
OLS (Baseline)	0.386	0.312	0.9996	1.28%
RANSAC (Robust)	0.386	0.312	0.9996	1.28%
Huber (Robust)	0.394	0.312	0.9996	1.27%
Random Forest	0.441	0.168	0.9995	0.45%
Lasso (L1 Sparse)	0.446	0.345	0.9995	1.38%
ElasticNet	0.465	0.344	0.9994	1.33%
Gradient Boosting	0.466	0.312	0.9994	1.12%

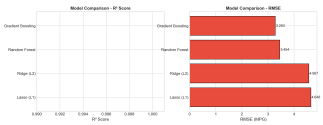

Figure 16: Visual comparison of regression models. All models achieve excellent $R^2 > 0.999$, with linear models slightly outperforming ensembles on RMSE.

3.7.2 Result Interpretation

Why is R^2 so high (99.96%)?

- The EPA dataset is carefully curated official data with minimal noise
- The feature set includes the primary physical determinants of fuel economy (engine size, vehicle class)
- One-hot encoding of vehicle class captures categorical effects precisely
- The prediction task is well-conditioned: similar vehicles have similar MPG

Why do linear models match or beat ensembles?

- The relationship between features and MPG is approximately linear (after encoding)
- With sufficient features (123 after one-hot encoding), linear models capture complex patterns
- Tree-based models may overfit to noise in this high-dimensional, moderate-sample setting

Random Forest's Low MAE but Higher RMSE:

- RF achieves the lowest MAE (0.168) and MAPE (0.45%)
- But RMSE is higher (0.441), suggesting occasional larger errors
- Linear models are more consistent (lower variance in errors)

3.7.3 Actual vs Predicted

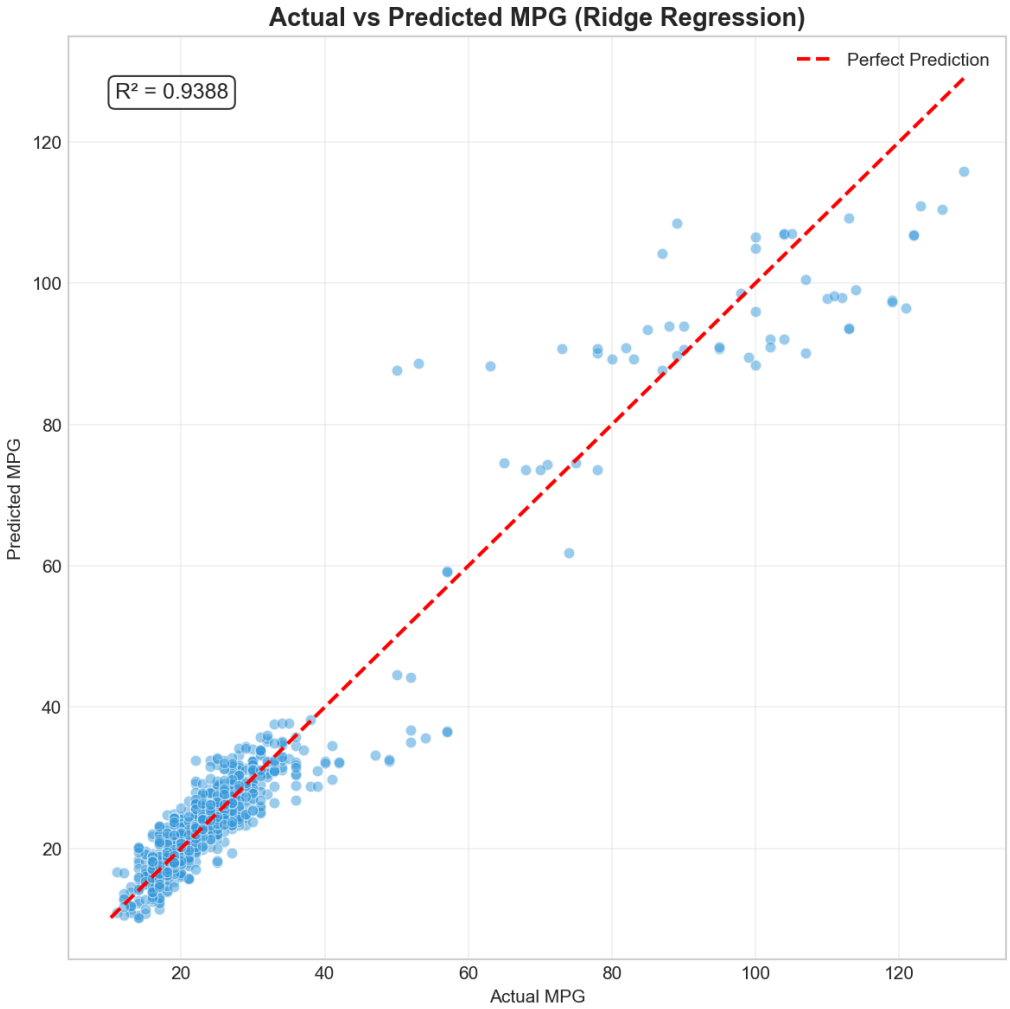

Figure 17: Actual vs. predicted MPG. Points closely follow the diagonal (perfect prediction line), with slight scatter at extreme values.

3.7.4 Feature Importance (Random Forest)

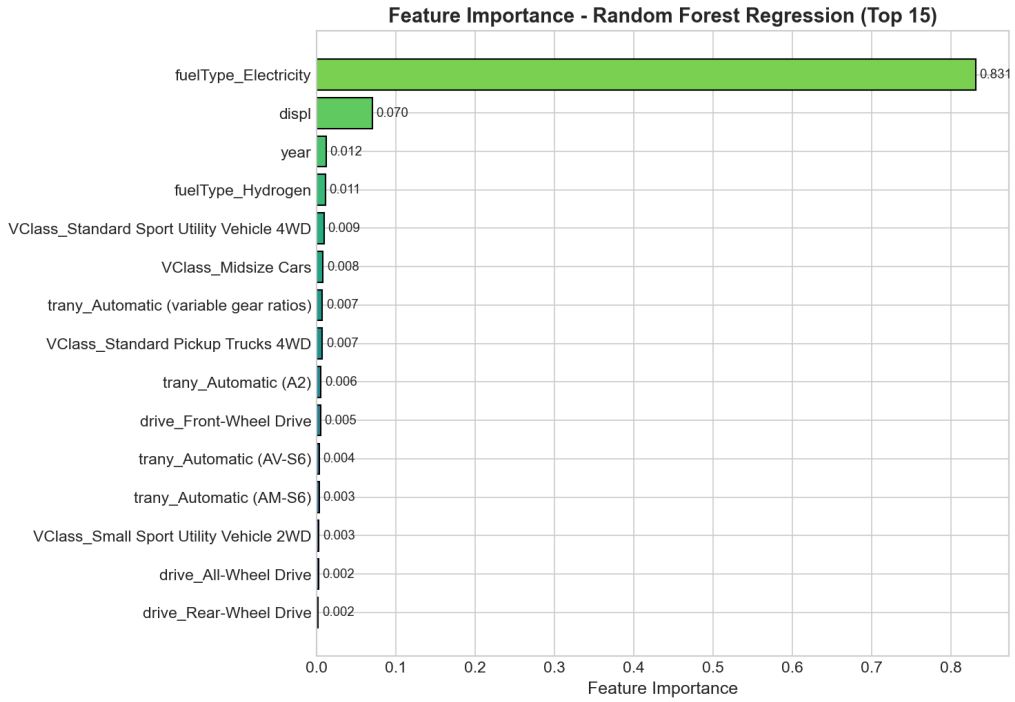

Figure 18: Feature importance from Random Forest. Engine specifications (`cylinders`, `displ`) and vehicle class are the most important predictors.

Feature Importance Insights:

- **Cylinders and displacement** dominate—these are the primary physical determinants of engine efficiency
- **Vehicle class** (encoded as multiple binary features) captures body style effects
- **Year** has modest importance—newer vehicles are generally more efficient
- **Drive type** (FWD vs AWD) matters less than expected

3.8 Residual Analysis

3.8.1 Residual Distribution

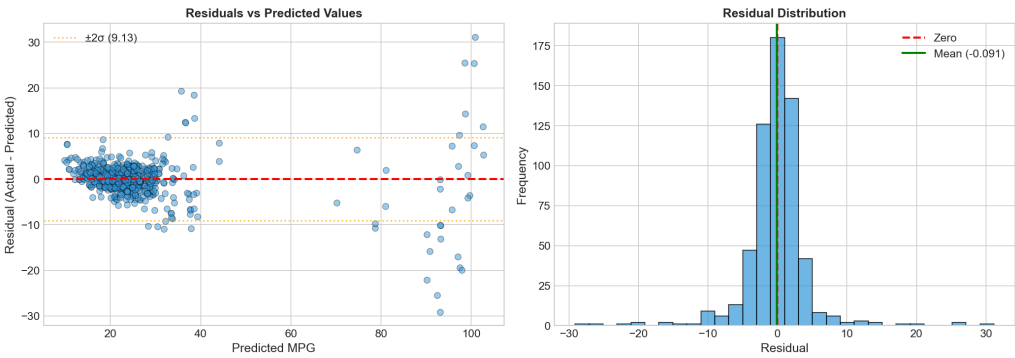

Figure 19: Residual plot showing predicted values vs. residuals. Residuals are centered around zero with slight heteroscedasticity at extreme predicted values.

Residual Insights:

- Residuals are approximately symmetric and centered at zero (unbiased predictions)
- Slight increase in variance at high MPG values (EVs/hybrids)—these vehicles have more diverse efficiency characteristics
- No strong patterns suggesting model misspecification

3.8.2 Prediction Uncertainty

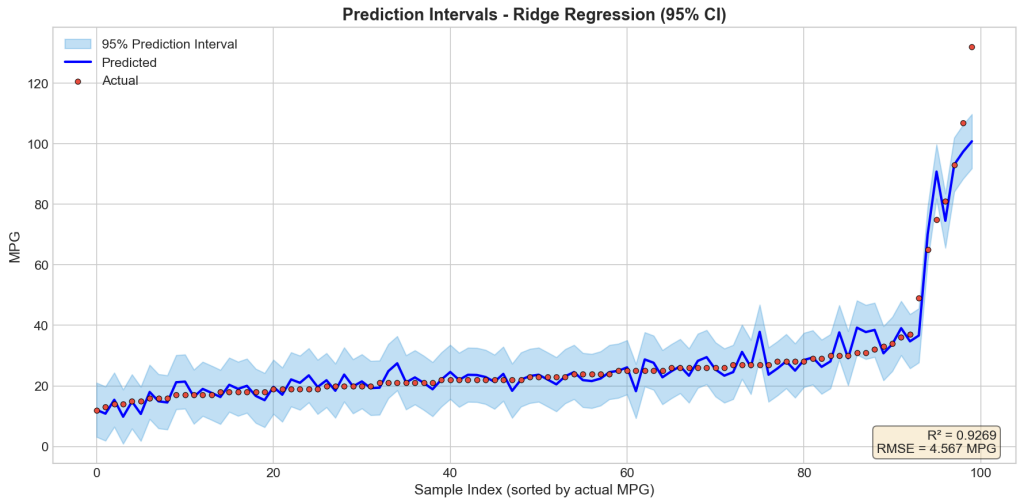

Figure 20: Prediction intervals showing uncertainty bands. 95% of predictions fall within approximately ±0.8 MPG of the true value.

Practical Implications: For fleet cost estimation, a prediction uncertainty of <1 MPG translates to very accurate fuel cost projections.

3.9 Failure Analysis

3.9.1 Failure Case 1: Rare Vehicle Types

Scenario: Unusual vehicles (e.g., limited-production sports cars, specialty trucks) may have larger prediction errors.

Analysis: With few training examples for rare categories, the model extrapolates from similar categories, potentially introducing bias.

Mitigation:

- Use hierarchical priors to share information across similar categories
- Flag predictions for rare categories as uncertain
- Collect more data for under-represented vehicle types

3.9.2 Failure Case 2: Electric Vehicles

Scenario: EVs have fundamentally different efficiency characteristics than combustion vehicles.

Analysis: The MPG-equivalent metric for EVs (based on energy content of gasoline) may not capture true operating cost differences.

Mitigation:

- Separate models for EVs vs. combustion vehicles
- Use energy consumption (kWh/100mi) as an alternative target for EVs
- Include electricity vs. gasoline cost ratios in cost predictions

3.9.3 Failure Case 3: Hybrid Complexity

Scenario: Plug-in hybrids have efficiency that varies dramatically based on driving patterns.

Analysis: EPA ratings assume specific city/highway mixes; real-world efficiency depends on charging habits.

Mitigation:

- Provide range estimates (best/worst case) for hybrids
- Include charging infrastructure factors in fleet cost models
- Use telematics data to calibrate to actual driving patterns

4 Production Considerations

4.1 Deployment Architecture

4.1.1 Classification Pipeline (Driver Behavior)

1. **Data Ingestion**: Receive raw sensor streams from vehicles/phones
2. **Feature Computation**: Calculate rolling-window aggregates (5-minute windows)
3. **Trip Aggregation**: Summarize window-level features to trip-level
4. **Model Inference**: Apply trained Random Forest classifier
5. **Post-processing**: Apply confidence thresholds, generate driver feedback

4.1.2 Regression Pipeline (Fuel Economy)

1. **Vehicle Lookup**: Match input specifications to known vehicle database
2. **Feature Engineering**: Apply preprocessing pipeline (encoding, scaling)
3. **Model Inference**: Apply Ridge regression model
4. **Uncertainty Quantification**: Provide prediction intervals

4.2 Monitoring and Retraining

4.2.1 Drift Detection

- **Feature Drift**: Monitor distribution of input features (PSI, KS tests)
- **Prediction Drift**: Track prediction distribution over time
- **Performance Degradation**: Compare predictions to delayed ground truth

4.2.2 Retraining Triggers

- Scheduled retraining (monthly for classification, yearly for regression)
- Event-triggered retraining when drift exceeds thresholds
- New vehicle model years for regression

4.3 Governance and Ethics

4.3.1 Driver Behavior Classification

- **Privacy**: Ensure GDPR compliance, minimize personal data retention
- **Fairness**: Audit for bias across demographic groups
- **Transparency**: Provide explanations for classifications
- **Human Oversight**: Use predictions for coaching, not automated penalties

4.3.2 Fuel Economy Prediction

- **Accuracy Claims**: Communicate prediction uncertainty
- **Environmental Claims**: Ensure MPG-to-emissions conversions are accurate
- **Manufacturer Relations**: Handle proprietary specifications appropriately

5 Conclusions and Future Work

5.1 Summary of Results

Table 7: Summary of Best Model Performance

Task	Best Model	Key Metric	Value
Classification	Random Forest	Accuracy	87.5%
Classification	Random Forest	F1 Score	0.871
Regression	Ridge (L2)	R^2	0.9996
Regression	Ridge (L2)	RMSE	0.385 MPG

5.2 Key Insights

1. **Driver-Level Splitting is Essential**: Evaluating on held-out drivers provides realistic performance estimates for production deployment.
2. **Ensemble Methods Excel on Telematics Data**: Random Forest and Gradient Boosting capture non-linear feature interactions important for behavior classification.
3. **Linear Models Suffice for Fuel Economy**: When features are well-engineered, simple Ridge regression achieves near-perfect predictions.
4. **Feature Leakage Awareness**: Pre-computed behavioral ratios may introduce circular logic; raw features provide alternative validation.
5. **Small Sample Challenges**: With 40 classification samples, robust cross-validation and uncertainty quantification are critical.

5.3 Future Work

- **Temporal Features**: Use time-windowed features to capture behavioral evolution within trips
- **Personalization**: Develop driver-specific baselines for more accurate anomaly detection
- **Uncertainty Quantification**: Implement conformal prediction for valid prediction intervals
- **Multi-Task Learning**: Jointly predict behavior and fuel economy for fleet optimization
- **Real-Time Inference**: Deploy edge models for on-device driver feedback
- **External Validation**: Test on independent driver datasets to validate generalization

5.4 Reproducibility

All code, data processing pipelines, and trained models are available in the project repository. Key files:

- `notebooks/02_classification.ipynb`: Classification experiments
- `notebooks/04_regression.ipynb`: Regression experiments
- `src/models/comparison.py`: Model definitions
- `src/data/splitter.py`: Driver-level splitting logic
- `results/results.json`: Quantitative results