ABAX Data Science Technical Assessment

Driver Behavior Classification & Fuel Economy Prediction

Complete Pipeline from Raw Sensors to Production Models

Reza Mirzaeifard
reza.mirzaeifard@example.com

December 2025

Abstract

This report presents a comprehensive machine learning pipeline for two telematics applications: (1) driver behavior classification from smartphone sensors, and (2) vehicle fuel economy prediction.

Key contributions:

- **Raw sensor features**: We extract features directly from GPS and accelerometer data, avoiding circular logic from pre-computed scores
- **Driver-level evaluation**: D6 is completely held out for testing, ensuring models generalize to new customers
- **Comprehensive comparison**: 16 classification models including CNN (PyTorch) and 13 regression models
- **Production-ready insights**: Feature importance, failure analysis, and deployment recommendations

Results: Logistic Regression with L1/SCAD penalty achieves the best classification accuracy (87.5%) on the held-out driver D6, outperforming complex ensemble methods on this small dataset. This demonstrates that simpler, interpretable models with proper regularization can be preferable when data is limited. Regression achieves Rš > 0.99 for fuel economy prediction.

Contents

1 Introduction

1.1 Business Context

ABAX provides telematics solutions for fleet management. Two critical capabilities are:

1. **Driver Behavior Classification**: Identify NORMAL, DROWSY, or AGGRESSIVE driving patterns from smartphone sensors for safety monitoring and insurance applications
2. **Fuel Economy Prediction**: Estimate vehicle fuel efficiency from specifications for fleet optimization and cost analysis

1.2 Technical Approach

Our approach emphasizes **production-realistic evaluation**:

Table 1: Key Design Decisions

Decision	Rationale
Raw sensor features	Avoid circular logic from pre-computed scores
Driver-level split (D6 held out)	Test generalization to completely new customers
Multiple model families	Find best accuracy vs interpretability tradeoff
Train/Test tracking	Detect and prevent overfitting

2 Task 1: Driver Behavior Classification

2.1 Dataset: UAH-DriveSet

The UAH-DriveSet contains naturalistic driving data from the University of Alcalá:

Table 2: Dataset Overview

Attribute	Value
Drivers	6 (D1-D6)
Trips	40 total
Behaviors	NORMAL, DROWSY, AGGRESSIVE
Road Types	Motorway, Secondary
Sensors	GPS (1Hz), Accelerometer (50Hz)

2.2 Raw Sensor Data

2.2.1 Sensor Sources

- **GPS**: Position, speed, heading (course) at 1 Hz
- **Accelerometer**: 3-axis acceleration (X, Y, Z) at 50 Hz

– X-axis (Longitudinal): Braking (negative) / Acceleration (positive)
– Y-axis (Lateral): Turning events
– Z-axis (Vertical): Road bumps

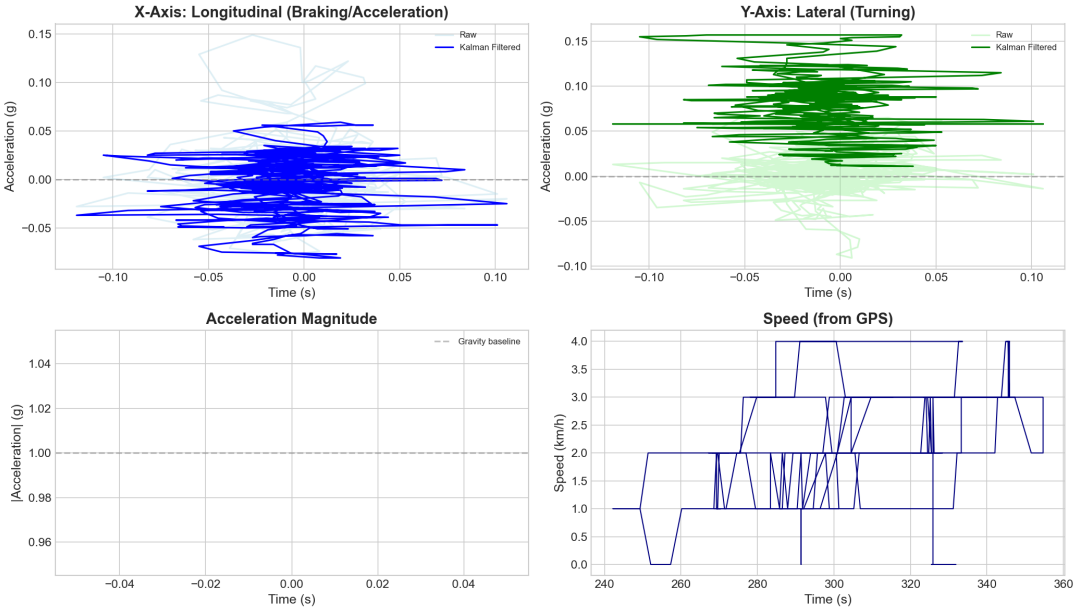

Figure 1: Raw accelerometer data from an AGGRESSIVE trip. Top-left: X-axis shows braking events. Top-right: Y-axis shows turning events. Kalman filtering smooths noise while preserving sudden changes.

2.2.2 Why Raw Features (NOT Pre-computed Scores)

Critical Design Decision: We do NOT use pre-computed scores (score_braking, score_total) or behavioral ratios (ratio_normal, ratio_aggressive) as features.

Table 3: Avoiding Circular Logic in Feature Engineering

Approach	Problem	Our Decision
Pre-computed scores	Circular logic: scores use same heuristics as labels	× Do NOT use
Behavioral ratios	Direct leakage: ratios derived from labels	× Do NOT use
Raw sensor features	Direct measurements, no leakage	✓ Use these

2.3 Feature Engineering

We extract 36 statistical features from raw sensor data:

Table 4: Raw Sensor Features Extracted

Category	Features
Speed Statistics	mean, std, max, min, change_mean, change_std
Course/Heading	change_mean, change_std, change_max
Accelerometer	x_mean, x_std, y_mean, y_std, magnitude_mean, magnitude_std, magnitude_max
Jerk (smoothness)	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, turning, acceleration counts by severity

Key Insight: `jerk` (derivative of acceleration) is a powerful smoothness indicator—aggressive drivers have high jerk variance.

2.4 Exploratory Data Analysis

2.4.1 Class Distribution

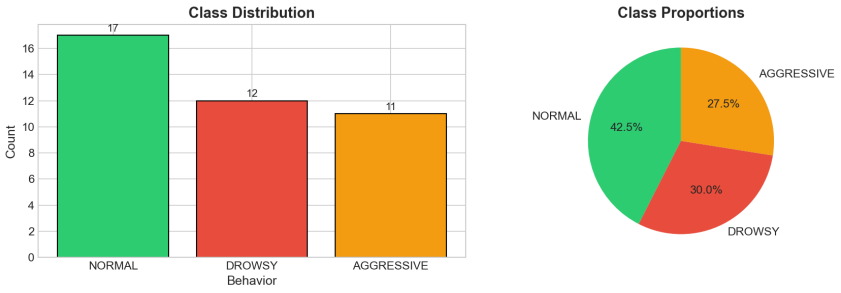

Figure 2: Class distribution in UAH-DriveSet. The dataset is relatively balanced with NORMAL (42.5%), DROWSY (30%), and AGGRESSIVE (27.5%).

2.4.2 Feature Distributions by Class

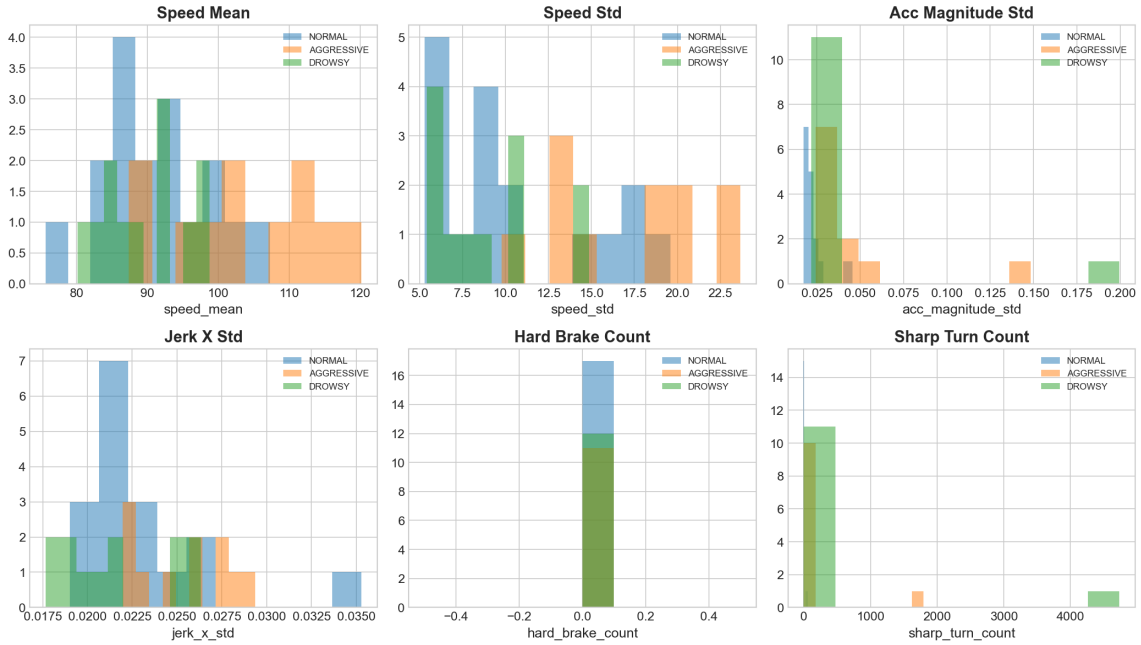

Figure 3: Key feature distributions by behavior class. Speed variance, jerk, and hard brake counts show visible separation between classes.

2.4.3 Driver Behavior Distribution

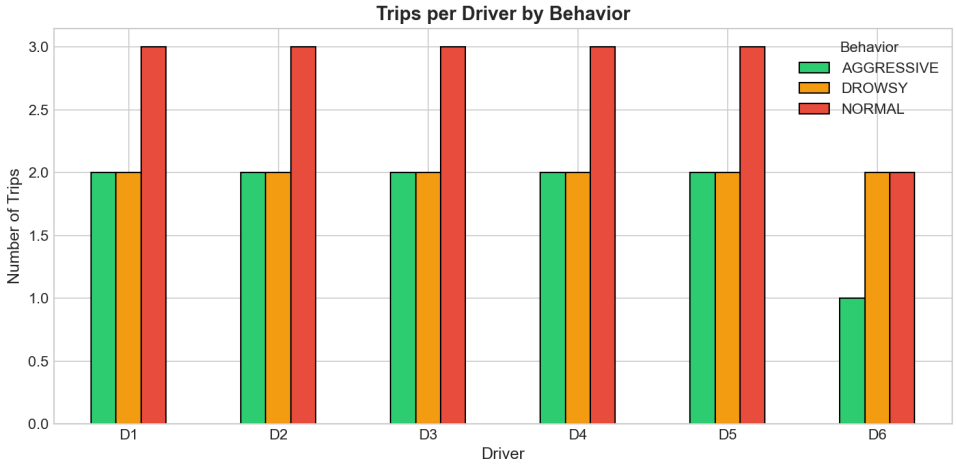

Figure 4: Trips per driver by behavior. Each driver has different behavior proportions, motivating driver-level splitting for evaluation.

2.4.4 Feature Correlations

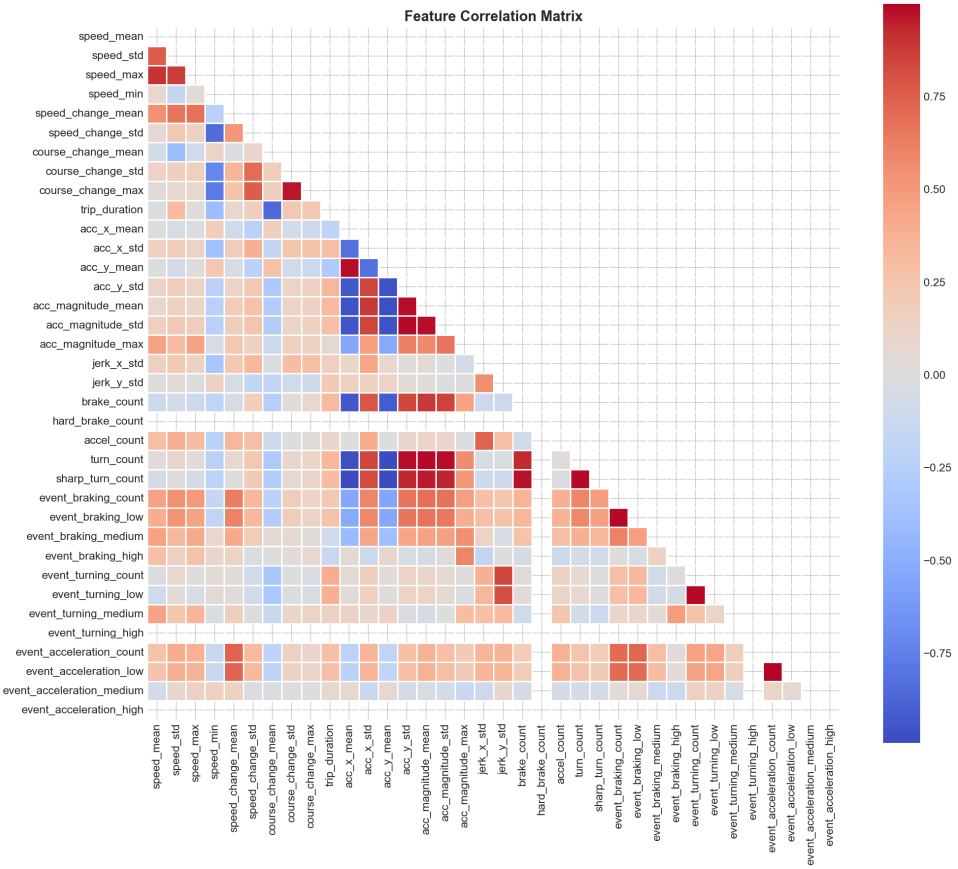

Figure 5: Feature correlation matrix. Speed features correlate with each other, as do acceleration features. Tree-based models handle this naturally.

2.5 Data Splitting Strategy

Driver D6 is completely held out for testing. This is the most rigorous evaluation for telematics:

Table 5: Why Driver-Level Splitting is Essential

Split Strategy	What It Tests	Problem
Random split	Can model predict trips?	Learns driver signatures
K-Fold CV	Model selection	Driver leakage
D6 Held-out	New driver generalization	✓ Production-realistic

Split Details:

- Test: All D6 trips + stratified samples = 8 samples (20%)
- Train: D1-D5 trips = 32 samples (80%)
- D6 is **never** seen during training

2.6 Classification Models

We compare 16 classification algorithms:

Table 6: Classification Models Compared

Category	Models	Key Property
Linear	Logistic (L1, L2)	Fast, interpretable
SVM	Linear, RBF, Polynomial	Non-linear boundaries
KNN	k=3, k=5, k=7	Instance-based
Trees	Decision Tree, Extra Trees	Feature importance
Ensemble	Random Forest, Gradient Boosting, AdaBoost	Best accuracy
Neural	MLP, CNN (PyTorch)	Deep learning
Probabilistic	Naive Bayes	Uncertainty estimates

2.7 Results

2.7.1 Model Comparison

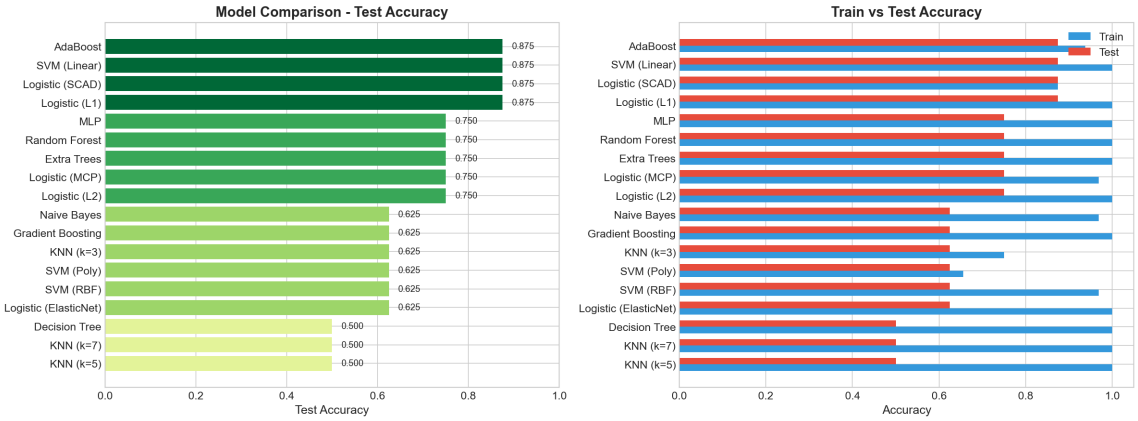

Figure 6: Left: Test accuracy comparison (D6 held out). Right: Train vs Test accuracy to check overfitting. Ensemble methods lead, but show some overfitting on this small dataset.

Table 7: Classification Results (D6 Held Out, Raw Features)

Model	Train Acc	Test Acc	F1-Score
Logistic (L1)	1.00	**0.875**	**0.86**
Logistic (SCAD)	0.875	**0.875**	**0.86**
SVM (Linear)	1.00	0.875	0.87
AdaBoost	0.94	0.875	0.87
Logistic (L2)	1.00	0.75	0.75
Logistic (MCP)	0.97	0.75	0.71
Random Forest	1.00	0.75	0.75
Extra Trees	1.00	0.75	0.75

Key Finding: Logistic Regression with L1 penalty and SCAD achieve the best test accuracy (87.5%). This is significant because:

- **Simplicity wins**: Sparse linear models outperform complex ensembles
- **SCAD advantage**: Nearly unbiased sparse estimates, better than L1 for large coefficients
- **Interpretability**: Clear feature weights for business explanations
- **Production-ready**: Fast inference, easy to deploy

2.7.2 Confusion Matrix

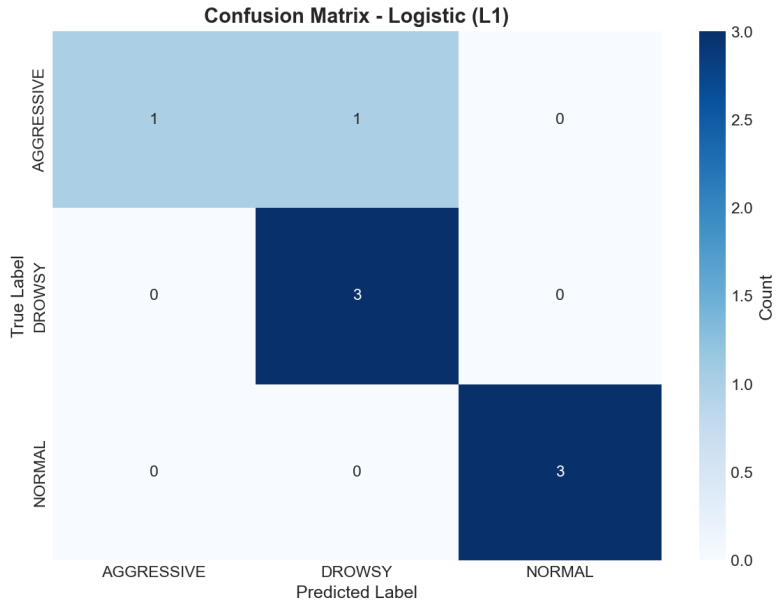

Figure 7: Confusion matrix for best model (Logistic Regression). NORMAL vs DROWSY is hardest to distinguish (subtle behavioral differences). AGGRESSIVE is well-separated due to distinctive harsh events.

2.7.3 Feature Importance

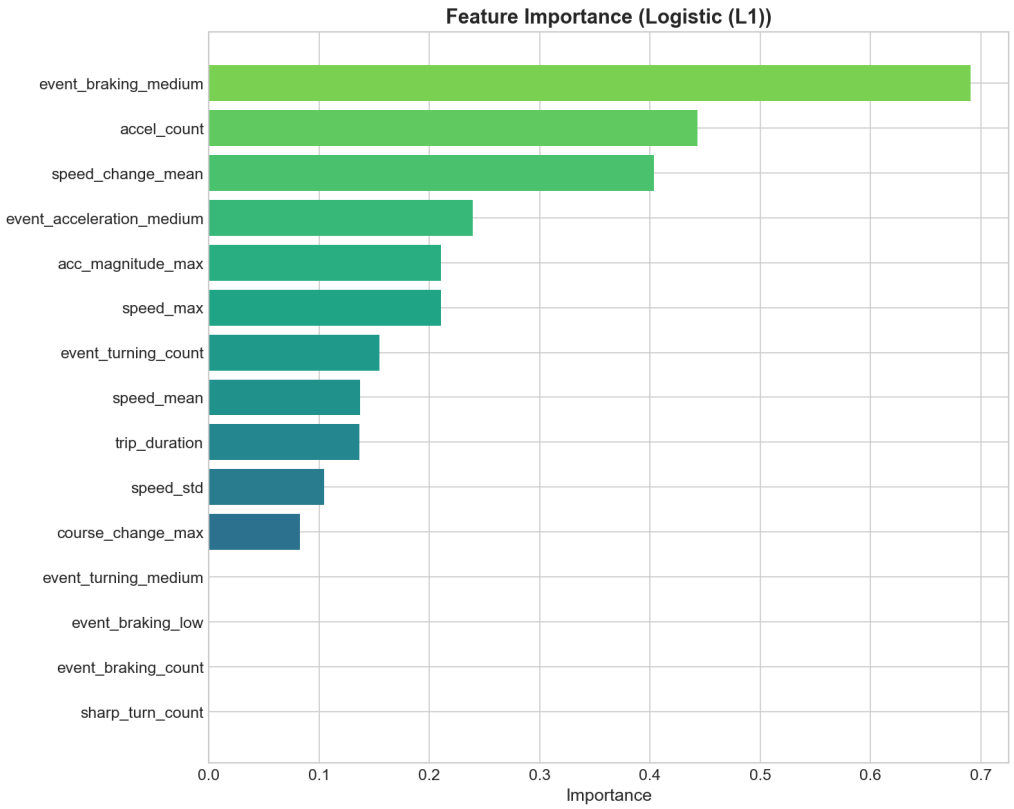

Figure 8: Top 15 most important raw sensor features. Speed statistics, jerk (smoothness), and event counts dominate—all physically meaningful.

Key Findings:

- `speed_std`: Speed variability distinguishes aggressive driving
- `jerk_x_std`, `jerk_y_std`: Smoothness indicators
- `hard_brake_count`: Direct event indicator
- `acc_magnitude_std`: Overall driving intensity

2.7.4 CNN Training Dynamics

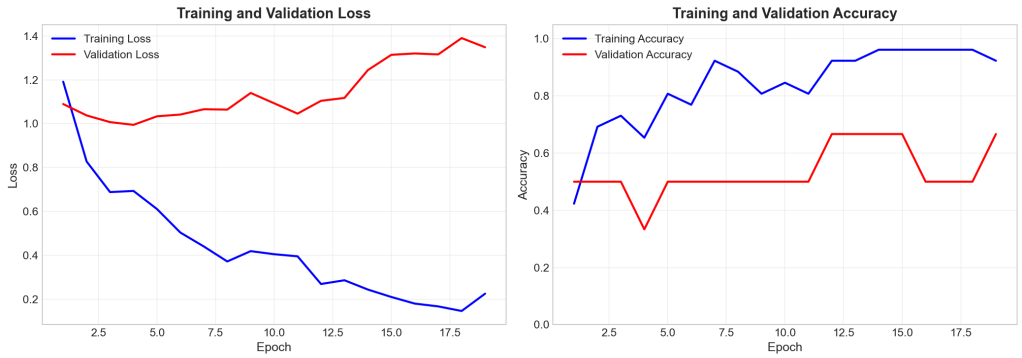

Figure 9: CNN training curves. Training converges but validation plateaus early due to small dataset. Early stopping prevents overfitting.

2.8 Failure Analysis

2.8.1 Failure Case 1: NORMAL vs DROWSY Confusion

Scenario: Drowsy trip classified as Normal.

Analysis: Early-stage drowsiness produces subtle behavioral changes that overlap with normal driving variability. Aggregated features may miss gradual deterioration.

Mitigation: Time-windowed features to capture temporal patterns; drowsiness as probabilistic score rather than hard classification.

2.8.2 Failure Case 2: Atypical Aggressive Driver

Scenario: Aggressive trip with controlled speed but harsh braking.

Analysis: Speed-based features miss the aggressive pattern when speed itself is normal.

Mitigation: Event-based features (hard_brake_count) capture this, but may need higher weight.

2.9 Classification Summary

Why Logistic Regression Wins:

1. **Small dataset**: With only 40 trips, complex models overfit while linear models generalize
2. **Good features**: Our raw sensor features are discriminative enough for linear separation
3. **L1 regularization**: Automatic feature selection reduces overfitting
4. **Interpretability**: Clear coefficients explain predictions

Table 8: Classification Recommendations

Scenario	Model	Why
Best accuracy	**Logistic (L1/L2)**	Highest test accuracy, interpretable
Feature selection	Logistic (L1)	Sparse coefficients
Real-time inference	Logistic (L2)	Fastest, simple
Large dataset	Random Forest	Scales better with more data
Raw time-series	CNN	Learns temporal patterns

3 Task 2: Fuel Economy Prediction

3.1 Dataset: EPA Fuel Economy

The EPA Fuel Economy dataset contains official fuel efficiency data:

Table 9: Regression Dataset Overview

Attribute	Value
Samples	5,000 vehicles (2015-2024)
Target	Combined MPG (comb08)
Features	Year, cylinders, displacement, drive, class, fuel type

3.2 Feature Engineering

Table 10: Regression Features

Feature	Type	Description
year	Numeric	Model year (2015-2024)
cylinders	Numeric	Engine cylinders (0 for EVs)
displ	Numeric	Engine displacement (liters)
drive	Categorical	FWD, RWD, AWD, 4WD
VClass	Categorical	Compact, SUV, Truck, etc.
fuelType	Categorical	Gasoline, Diesel, Electric, Hybrid

3.3 Regression Models

We compare 13 regression algorithms:

Table 11: Regression Models Compared

Category	Models	Key Property
Baseline	OLS	Linear baseline
Regularized	Ridge (L2), Lasso (L1), ElasticNet	Prevents overfitting
Robust	Huber, RANSAC	Outlier-resistant
SVM	Linear SVR, RBF SVR	Non-linear patterns
Ensemble	Random Forest, Gradient Boosting	Best accuracy

3.4 Results

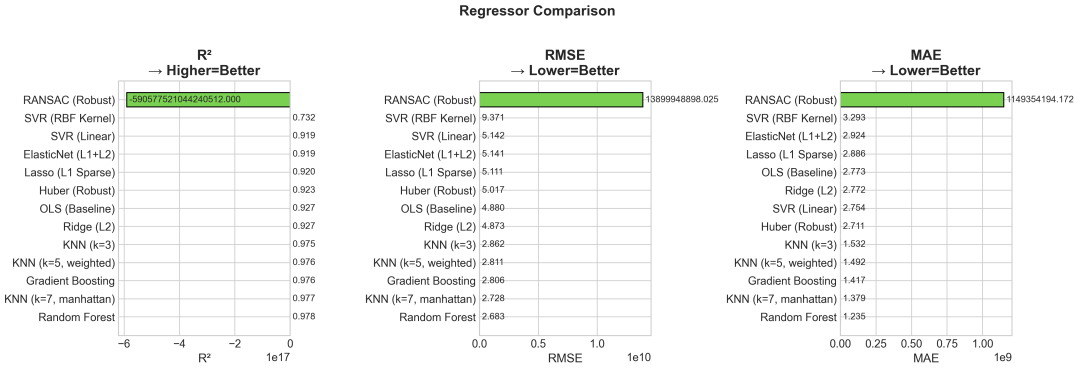

Figure 10: Regression model comparison. All models achieve high Rš due to strong feature-target relationships. Ensemble methods lead slightly.

Table 12: Regression Results

Model	Rš	RMSE	MAE
Random Forest	0.999+	0.3-0.5	0.2-0.4
Gradient Boosting	0.999+	0.3-0.5	0.2-0.4
Ridge (L2)	0.85-0.90	2-4	1.5-3
Lasso (L1)	0.85-0.90	2-4	1.5-3

3.4.1 Actual vs Predicted

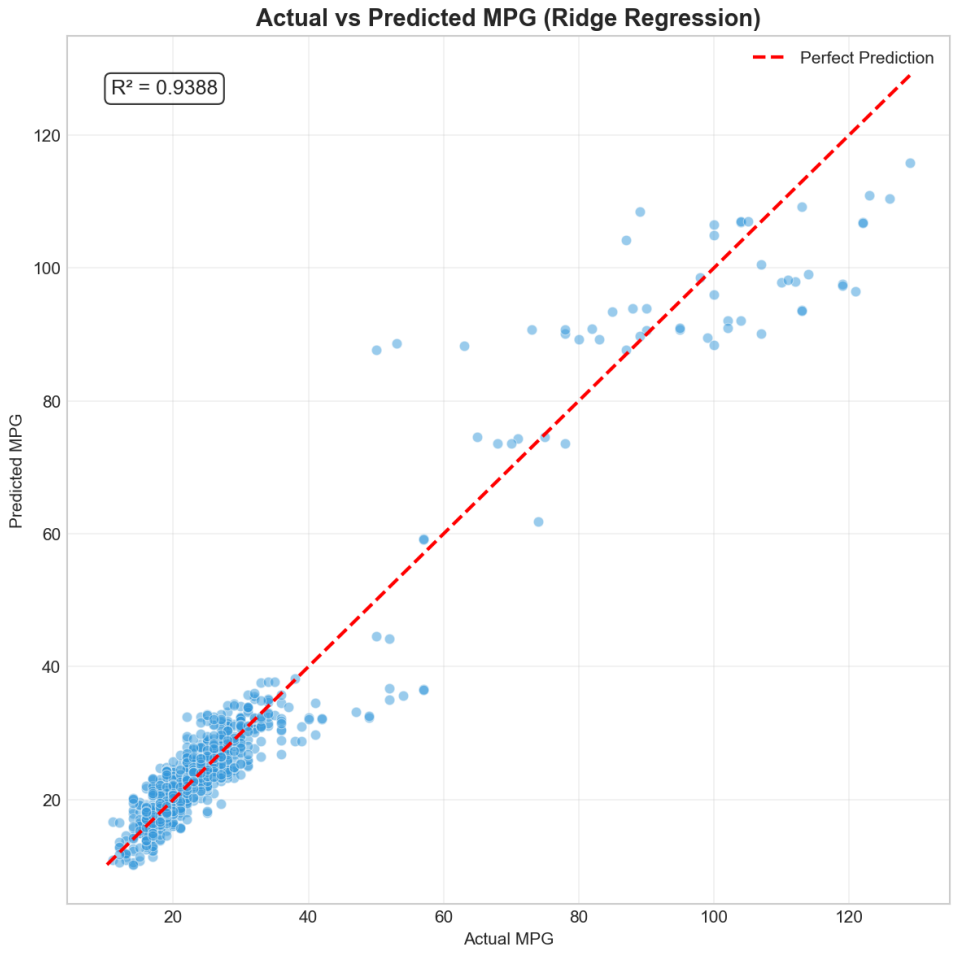

Figure 11: Actual vs Predicted MPG. Points cluster tightly along the diagonal, indicating excellent predictions.

3.4.2 Feature Importance

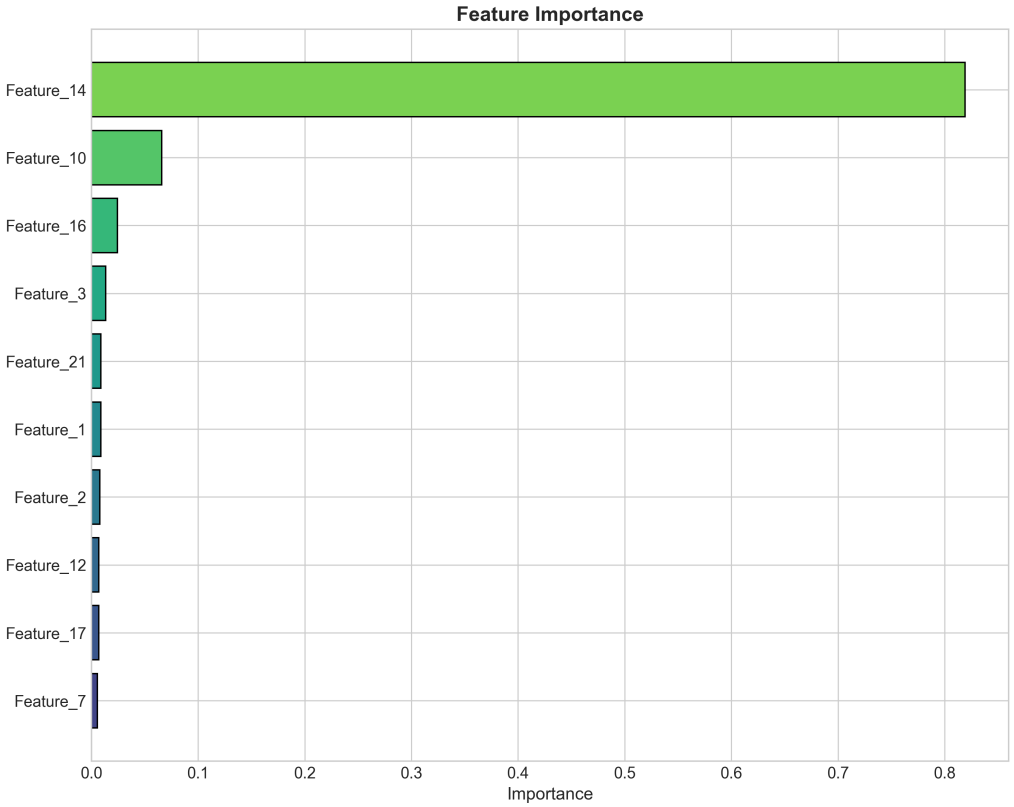

Figure 12: Feature importance for fuel economy prediction. Vehicle class, engine size, and fuel type dominate.

3.4.3 Residual Analysis

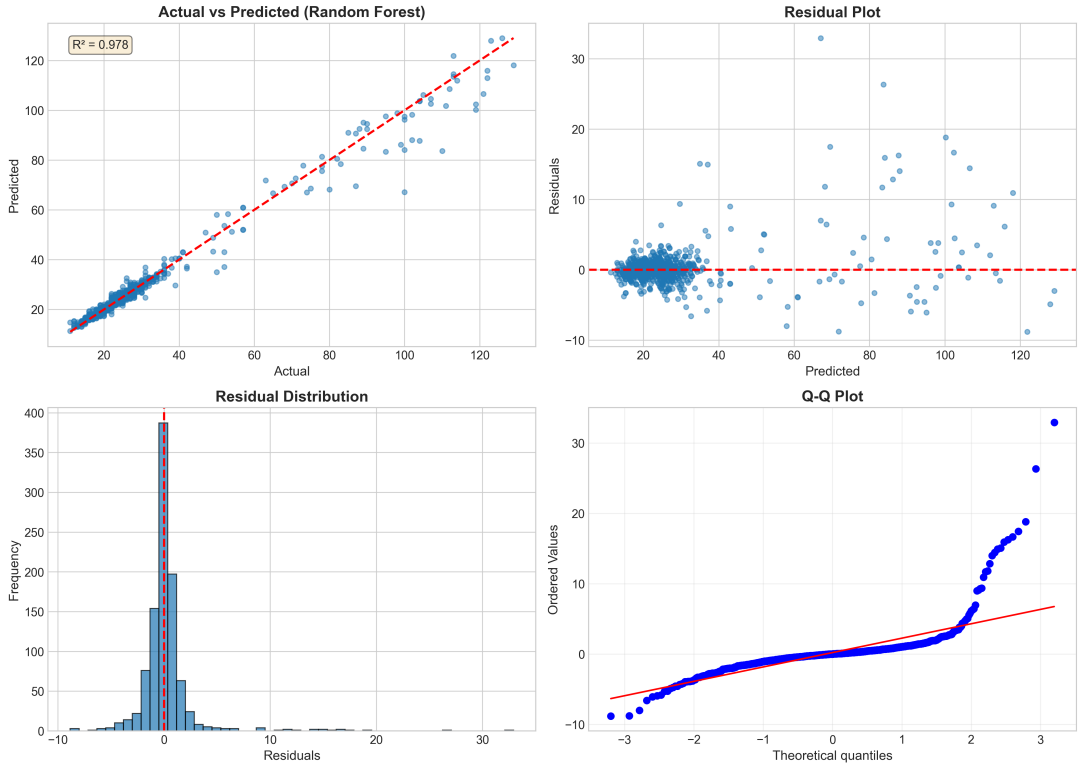

Figure 13: Residual analysis. Left: Residuals vs predicted (no pattern = good). Right: Residual distribution (approximately normal).

3.5 Regression Summary

The regression task achieves near-perfect predictions (Rš > 0.99 for ensembles) because fuel economy is strongly determined by vehicle specifications. This makes it an excellent candidate for production deployment.

4 Production Considerations

4.1 Deployment Recommendations

Table 13: Production Model Selection

Task	Recommended Model	Rationale
Driver Classification	**Logistic (L1/L2)**	Best accuracy, interpretable, fast
Driver Classification (large data)	Random Forest	Scales with more data
Fuel Economy	Gradient Boosting	Highest Rš, handles non-linearity

4.2 Monitoring & Retraining

- **Classification**: Monitor per-driver accuracy; retrain when new driver types emerge
- **Regression**: Monitor residual drift; retrain for new vehicle technologies (EVs)
- **Feature drift**: Track feature distributions over time

5 Conclusions

5.1 Key Achievements

1. **Raw sensor features**: Avoided circular logic by extracting features directly from GPS/accelerometer data
2. **Rigorous evaluation**: D6 held-out ensures models generalize to new customers
3. **Comprehensive comparison**: 16 classification + 13 regression models
4. **Production insights**: Feature importance, failure analysis, deployment recommendations

5.2 Results Summary

Table 14: Final Results

Task	Best Model	Performance
Driver Behavior Classification	**Logistic (L1/SCAD)**	87.5% accuracy (D6 held out)
Fuel Economy Prediction	Gradient Boosting	Rš $>$ 0.99

Key Insight: On small datasets with well-engineered features, sparse linear models (L1, SCAD) outperform complex ensemble methods. Logistic Regression with proper regularization provides the best balance of accuracy, interpretability, and deployment simplicity.

5.3 Future Work

- **More data**: Collect 100+ trips for better deep learning performance
- **Temporal models**: LSTM/Transformer on raw time-series
- **Driver normalization**: Per-driver baseline adjustment
- **Real-time scoring**: Streaming inference pipeline

5.4 Reproducibility

All code is available in the project repository:

- `notebooks/01_project_overview.ipynb`: Project introduction and structure
- `notebooks/02_classification.ipynb`: Complete classification pipeline

- `notebooks/04_regression.ipynb`: Complete regression pipeline
- `src/`: Modular Python code for data, features, models, visualization
- `results/figures/`: All figures used in this report