CMPT310 Assignment 1 – Diar Shakimov

1.a:

Problem Definition:

We want to classify the car as fuel-efficient or not. This is a simple binary choice thus we should use classification. Knowing we have mutliple features we'd like to model, KNN with a threshhold would be a good pick as it splits it into two classes and is easy to implement as we just enumerate and compare the features to find the distance between cars. For the threshhold, we can pick median mpg (22.750) as it can help to balance the classes and reduce class imbalances.

1.b Preprocessing:

For preprocessing we pick out and clean data to maximize performance in the real world.

Cylinders: one-hot to avoid imposing linearity, better for capturing nonlinearity while standard would be good if linear, but may risk implying order.

Displacement: standard is continuous with wide range thus standardization stabilizes KNN distances and logistic and linear regression optimization.

Horsepower: standard, similar to displacement since standardizing improves scale-consistency and strength from regularizing.

Weight: standard as with large magnitudes standardizing makes it so that it doesn't dominate distance based models.

Acceleration: standrad. Continous with smaller range but will still standardize to be consistent.

Model Year: standard good for era trends but can lead to time leakage. Raw: large scale so avoid.

Origin: One-hot is categorical and avoids fake ordering so its good. Raw and standard impose order so we should avoid.

Car name: drop: high cardinality text, noisy.

1.c General

3.c.1: Can we predict 2019 cars? Not too reliably with just this dataset since it comes from older vehicles, feature mpg relationships and distributions shift over decades. Without reweighting or recent data, predictions may be biased.

3.c.2: Scale differences and model difficulty KNN: Yes, sensitive to feature scales as large scale features dominate distance. This can be mitigated with standardization.
Logistic regression: The model is scale-sensitive for optimization and regularization so the different scales mess up gradients and effective regularization. This can be mitigated by performing standardization.
Decision trees: Not sensitive to feature scaling so split thresholds are order-based thus no scaling needed.

Setup: KNN with features2 (cylinders=one_hot; displacement, horsepower, weight, acceleration=standard; origin=one_hot). Label efficient if mpg >= dataset median(22.75). 10-fold cross-validation, shuffled once with fixed seed.

Metrics: Accuracy and F1 because accuracy fits a balanced threshold and F1 summarizes what we got from precision and recall if any residual inbalance remains.

Results (mean +- std across folds):

- K=3: Accuracy 0.913 +- 0.030, F1 0.913 +- 0.030
- K=5: Accuracy 0.916 +- 0.043, F1 0.916 +- 0.041
- K=7: Accuracy 0.911 +- 0.046, F1 0.913 +- 0.040

Discussion: K=5 slightly outperforms others and is less variance-prone than K=3, so we would select K=5. Using the median threshold keeps classes balanced; standardization is essential for KNN.

Part III Regression

3.e.1 Best configuration found: features2 with lambda = 0.0. features2 is cylinders=one_hot, displacement=standard, horsepower=standard, weight=standard, acceleration=standard, origin=one_hot. The best overall run occurred with polynomial order = 2.

3.e.2 Average 10 fold RMSE in mpg (best combo) 3.841 mpg

Search Grids I used:

orders = {1,2,3}

Lambdas = {0.00, 0.01, …, 0.10} for orders 1-2 and {0, 20, …, 200} for orders 3.

Report MPG RMSE = (stand(RMSE)) * sigma from std_y