Estimation of Vehicle Mass and Road Grade

CS116.O11.KHCL - Machine Learning with Python: Final Project

Nguyen Hoang Tan
MSSV: 215214113
21521413@gm.uit.edu.vn

***Abstract*—The Estimation of Vehicle Mass and Road Grade is a crucial aspect of modern transportation systems. This report details our approach, encompassing exploratory data analysis (EDA), feature engineering, data splitting, model selection, and evaluation, resulting in achieving the 1st rank in both public and private contest datasets.**

I. Introduction

The Estimation of Vehicle Mass and Road Grade is a multifaceted challenge within the domain of transportation engineering, encompassing critical aspects such as fuel efficiency, vehicle performance, and overall safety. This project leverages machine learning methodologies to address this complex problem. We employ **Random Forest Classifier** for Vehicle Mass and **K-Nearest Neighbors Regressor** for Road Grade estimation, focusing on the utilization of diverse signals collected from a vehicle to predict both its mass and the grade of the road it traverses.

II. Dataset Description

The dataset used in this project comprises eleven signals obtained from a vehicle, with the first nine serving as input features, and the last two as output variables. Notably, the data lacks time information, and the order of recordings has been deliberately scrambled. Each record in the dataset represents an individual frame, and the absence of temporal information necessitates an algorithmic approach that operates independently on each frame.

The signals include key parameters such as engine speed, vehicle speed, torque-related metrics, clutch and engine operation status, as well as the desired torque or torque limit. Of particular significance are the signals indicating road slope and the vehicle's mass, represented as either 38 t or 49 t.

III. Exploratory Data Analysis (EDA)

Before delving into the machine learning models, it is crucial to conduct an Exploratory Data Analysis (EDA) to gain insights into the dataset's characteristics and identify potential patterns or anomalies.

Data Integrity Check Checking for missing values. Fortunately, the dataset demonstrates completeness, as no null values are present across any of the features.

Outlier Detection Generate box plots for each feature (excluding the target variable, Vehicle_Mass) to identify potential outliers.

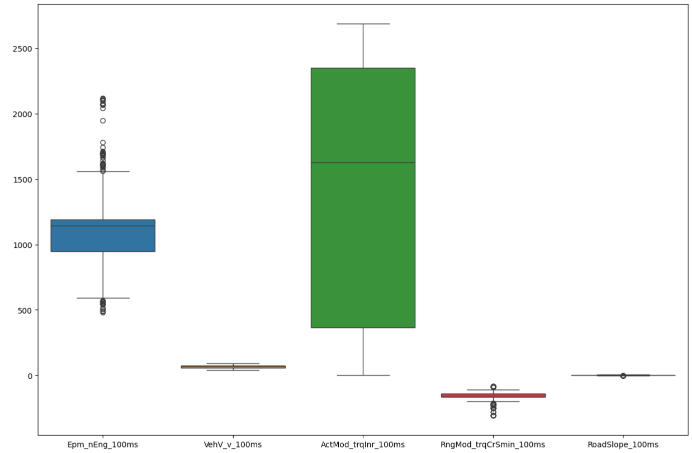


Fig. 1. Box Plot of Feature Columns

Figure 1 revealed the presence of numerous outliers across several features, potentially impacting the performance of machine learning models.

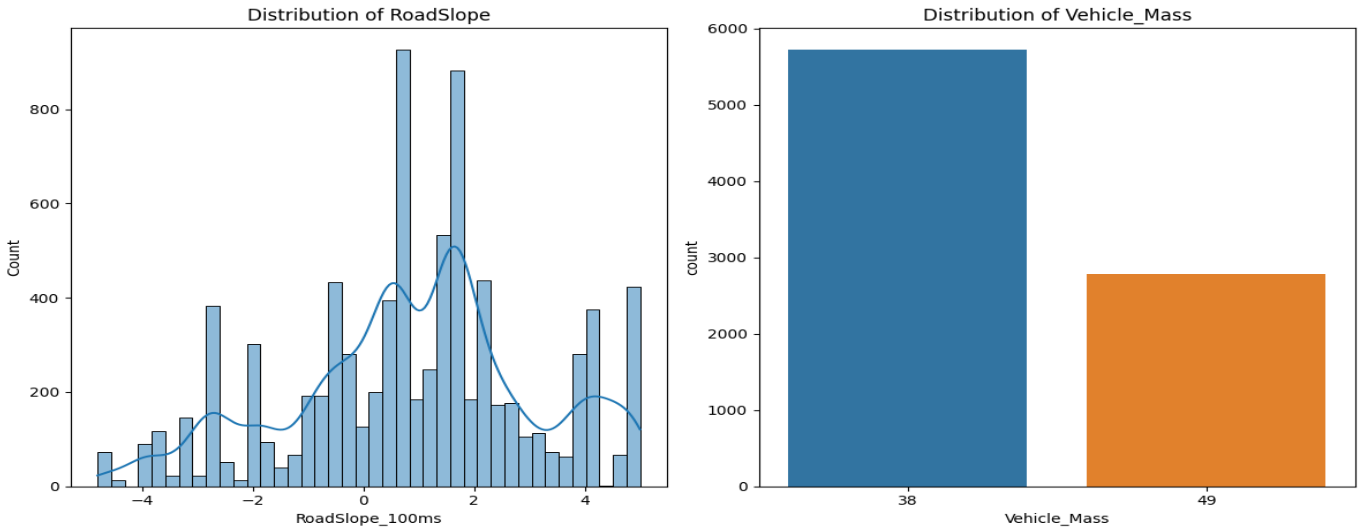
To mitigate the influence of outliers, We can utilize the **RobustScaler** during the feature scaling process.

Correlation Analysis Table I provides insights into the relationships between variables, highlights correlations between features. Notably, '**RoadSlope_100ms**' displays significant positive correlations with '**ActMod_trqInr_100ms**' and '**RngMod_trqCrSmin_100ms**''.

	RoadSlope_100ms	**Vehicle_Mass**
RoadSlope_100ms	1.000000	0.257673
ActMod_trqInr_100ms	0.743515	0.084114
RngMod_trqCrSmin_100ms	0.459027	0.604168
Vehicle_Mass	0.257673	1.000000
Epm_nEng_100ms	0.138132	0.156700
VehV_v_100ms	-0.705378	-0.630015

TABLE I
Correlation Matrix of Features

Additionally, the strong negative correlation (-0.63) between '**VehV_v_100ms**' and '**RngMod_trqCrSmin_100ms**' suggests the possibility of creating a new combined feature for improving model performance.



IV. Data Preprocessing

Having identified outliers, correlations, and distributions in the previous step, this section focuses on optimizing the dataset for the task.

A. Feature Engineering and Reformatting

- Irrelevant constant features (‘**CoVeh_trqAcs_100ms**‘, ‘**Com_rTSC1VRVCURtdrTq**‘, ‘**Clth_st**‘, ‘**CoEng_st**‘, ‘**Com_rTSC1VRRDTrqReq**‘) are dropped as they provide no discriminatory information to distinguish between different instances. .
- The ‘**Vehicle_Mass**‘’ column is reformatted to binary encoding for classification.

We create a new feature ‘**Combined_VehV_RngMod**‘ by combining ’**RngMod_trqCrSmin**’ and ’**VehV_v**’ using formula 1. This combination is motivated by the strong negative correlation of -0.63 observed between these two variables.

$$Combined\_VehV\_RngMod = \frac{RngMod\_trqCrSmin\_100ms}{VehV\_v\_100ms} \tag{1}$$

B. Task-specific Dataset Splitting

We initiate the dataset split into features and targets for both the regression and classification tasks, employing the same feature set for both predictions.

However, we also introduce an alternative ‘**MultiTask**‘ approach. In this strategy, we utilize the predicted vehicle mass values to augment the prediction of road slope.

Train-Dev-Test Splitting The dataset is partitioned into training, development, and test sets for both regression and classification tasks. The distribution of the dataset across these sets is as follows:

- Training Set: 70%
- Development Set: 15%
- Test Set: 15%

C. Feature Scaling

As we observed numerous outliers in various features (Figure 1). We address this problem by applying Robust scaling to the features to ensure their uniformity across different scales.

I also experimented with alternative scaling methods to assess their impact on the overall model performance. The evaluation results are presented in Table II.

TABLE II
Model Performance with Different Scaling Methods

Scaling Method	Public Sets	Private Sets
RobustScaler	88.03	86.63
StandardScaler	87.79	85.26
MinMaxScaler	83.66	-
MaxAbsScaler	83.66	-
Normalizer	65.83	-
PowerTransformer	85.87	-
QuantileTransformer	85.33	-

Note: Due to limited submit attempts for private sets, evaluation was performed with only two scalers.

V. Models Selection

As mentioned above, we will use 2 separate models to address the distinct tasks at hand.

A. Vehicle Mass Classifier Task

The objective of this task is to train a classifier to categorize a vehicle’s mass into two classes: 38 t or 49 t.

Random Forest Classifier An ensemble learning algorithm that operates by constructing a multitude of decision trees during training and outputting the mode of the classes for classification tasks.

- Each decision tree in the forest is trained on a random subset of the training data and provides an independent prediction

- The final classification result is determined by aggregating the predictions from all the individual trees.

In addition to the Random Forest classifier, we explored various other classifiers before selecting the final model. The experimentation involved evaluating different algorithms to assess their performance on the Vehicle Mass Classification task.

TABLE III
G-MEAN SCORES FOR VEHICLE MASS CLASSIFICATION MODELS

Classifier	**Public Sets**	**Private Sets**
SVC	0.9959	-
Logistic Regression	0.9748	-
Decision Tree Classification	0.9977	0.9945
Random Forest Classification	0.9988	0.9984

Note: Due to limited submit attempts for private sets, evaluation was performed with only two best classifiers.

B. Road Slope Regression Task

In this task, the goal is to predict the road slope based on the given features. We'll assess the performance of selected models, including KNN Regressor, to achieve accurate predictions.

KNN Regressor Predicts the target variable by considering the average or weighted average of the k-nearest data points in the feature space.

The choice of the hyperparameter `n_neighbors` in KNN Regressor is crucial, as it determines the number of neighbors that contribute to the prediction. This parameter needs to be carefully tuned for optimal performance.

Figure 2 shows the experimental results conducted with different values of `n_neighbors`.

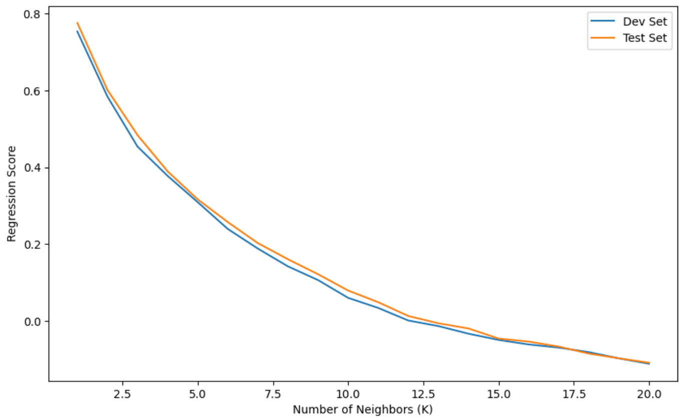


Fig. 2. Regression Performance vs. Number of Neighbors (K)

In addition to the KNN Regressor, I also evaluated various regression models, from basic ones like SVR, and Bagging, to more complex techniques such as Gradient Boosting and XGBoost.

Among the regression models, Gradient Boosting, AdaBoost, and K-Nearest Neighbors (KNN) show promising results on both development and test sets. Gradient Boosting and AdaBoost consistently achieve positive scores, indicating strong generalization capabilities. However, in the private test, KNN (`n_neighbors = 1`) with tuned parameters outperforms these boosting algorithms

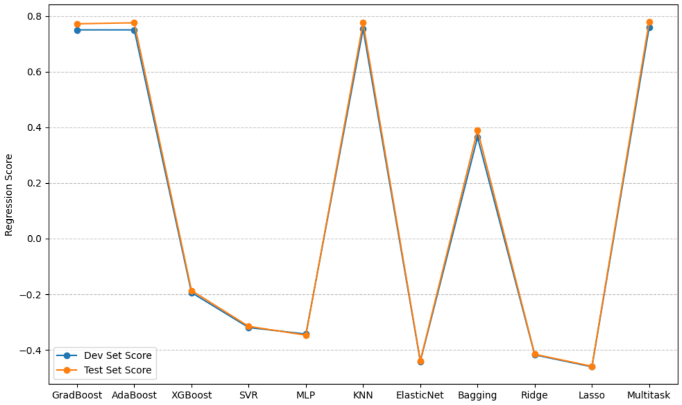


Fig. 3. Comparison of Models Performance on Dev and Test Sets

C. MultiTask Approach

In Figure 3, the MultiTask approach exhibits competitive performance on both the development and test sets, showcasing results slightly better than KNN. Notably, this approach demonstrates effectiveness in the private test as well.

The concept behind this approach involves utilizing the predicted value of '**Vehicle_mass**'', given its high G-score (above 0.998), as an input to predict the road slope.

VI. EVALUATION

Lastly, we will evaluate the performance of the two selected models, **RandomForest Classifier** and **KNN Regressor**, on both the public and private contest datasets.

	Public Set		Private Set
	Dev	Test	
Classification Task	0.999	1.000	0.998
Regression Task	0.754	0.776	0.795
Overall Score	0.840	0.854	0.866

TABLE IV
PERFORMANCE SCORES ON PUBLIC AND PRIVATE SETS

$$OverallScore = 0.35 \times classification + 0.65 \times regression$$

Note: The classification performance is evaluated using the G-mean score, and the regression score is defined by the contest.

This approach achieved the **1st** place on both the public test and private test of the course contest with scores of **199.86** and **86.63**, respectively. You can find the notebook, submission source code, and model parameters on GitHub."