Đồ án CS116

Lê Thị Kim Yến - 21521695
University of Information Technology, Ho Chi Minh City, Vietnam
Course: CS116.O11.KHCL
Lecturer: Nguyễn Vinh Tiệp - Đặng Văn Thìn

***Tóm tắt nội dung*—Đồ án tập trung giải quyết hai bài toán chính là regression và classification. Trong phần regression, em đã thử nghiệm và đánh giá nhiều mô hình như K-Nearest Neighbors (KNN) Regressor, Random Forest Regressor, Decision Tree Regressor và Gradient Boosting Regressor. Kết quả cho thấy mô hình KNN sau khi điều chỉnh tham số bằng GridSearchCV đạt được hiệu suất tốt nhất. Trong phần classification, em đã thử nghiệm các mô hình như Random Forest, Gradient Boosting, SVM, Bagging và Boosting với base là Decision Tree. Mô hình Bagging với base là Decision Tree cho thấy hiệu suất tốt nhất. Bên cạnh đó, việc thực hiện feature engineering bằng cách tạo ra các biến kết hợp để cải thiện hiệu suất của mô hình hồi quy. Dữ liệu đã được chuẩn hóa bằng cách sử dụng Robust Scaling. Kết quả trên tập train và public test cho thấy hiệu suất tích cực, tuy nhiên, kết quả trên private test giảm đáng kể, đề xuất cần nhiều sự cải thiện trong tương lai để đảm bảo tổng quát hóa mô hình.**

***Index Terms*—CS116-UIT, phân tích dữ liệu, mô hình hồi quy, mô hình phân loại.**

I. Introduction

Trong lĩnh vực tối ưu hóa hiệu suất xe ô tô, việc dự đoán các thông số đóng một vai trò quan trọng trong việc cải thiện hiệu quả và an toàn cho người sử dụng. Đồ án này sẽ tập trung vào việc phân tích và dự đoán giá trị độ dốc của đường đi (Regression) và trọng lượng của xe (Classification) dựa trên các thành phần khác nhau của một chiếc xe.

II. PHƯƠNG PHÁP ĐỀ XUẤT

A. Phân tích dữ liệu

*1) **Khám phá dữ liệu**:* Trước khi tiến hành xây dựng mô hình, việc thực hiện một phân tích chi tiết về dữ liệu là rất quan trọng. Bước này giúp chúng ta hiểu rõ hơn về mối quan hệ giữa các đặc trưng và giá trị độ dốc của đường, cũng như mối liên kết giữa các thông số và trọng lượng của xe. Mục tiêu là xác định các đặc điểm quan trọng, loại bỏ nhiễu trong dữ liệu và chuẩn bị dữ liệu một cách tốt nhất cho quá trình huấn luyện mô hình.

1.1. Hiển thị thông tin cơ bản về dữ liệu:

```
df.info()
```

Nhận xét:

- Dữ liệu có 8496 điểm dữ liệu với 11 đặc trưng non - null. Tất cả các đặc trưng đều là giá trị số (numeric values), không có giá trị phân loại (categorical values). Điều này có nghĩa là tất cả các thông tin trong dữ liệu đều được biểu diễn dưới dạng số học, và không có các biến phân loại như hạng mục hay nhóm. Điều này có thể ảnh hưởng đến việc lựa chọn mô hình và phương pháp xử lý dữ liệu trong quá trình phân tích và mô hình hóa.

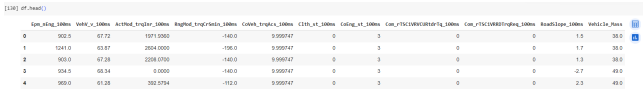
```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8496 entries, 0 to 8495
Data columns (total 11 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Epm_nEng_100ms              8496 non-null   float64
 1   VehV_v_100ms                8496 non-null   float64
 2   ActMod_trqInr_100ms         8496 non-null   float64
 3   RngMod_trqCrSmin_100ms      8496 non-null   float64
 4   CoVeh_trqAcs_100ms          8496 non-null   float64
 5   Clth_st_100ms               8496 non-null   int64
 6   CoEng_st_100ms              8496 non-null   int64
 7   Com_rTSC1VRVCURtdrTq_100ms  8496 non-null   int64
 8   Com_rTSC1VRRDTrqReq_100ms   8496 non-null   int64
 9   RoadSlope_100ms             8496 non-null   float64
 10  Vehicle_Mass                8496 non-null   float64
dtypes: float64(7), int64(4)
memory usage: 730.2 KB
```

Hình 1: Tổng quan dữ liệu

1.2. Hiển thị một số dòng đầu tiên của dữ liệu:

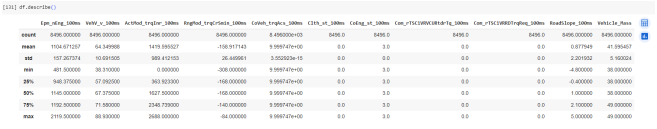
```
df.head()
```



Hình 2: 5 dòng đầu tiên của dữ liệu

1.3. Phân tích thống kê mô tả cơ bản:

```
df.describe()
```



Hình 3: Thống kê mô tả dữ liệu

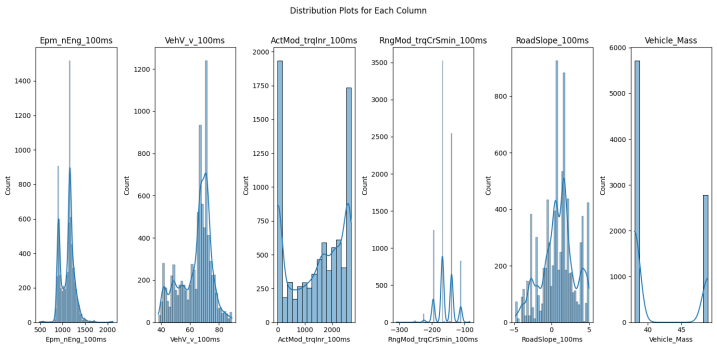
Nhận xét:

- Biến Epm_nEng_100ms và ActMod_trqInr_100ms có phạm vi giá trị khá lớn nên cần được scale lại.
- Các biến CoVeh_trqAcs_100ms, Clth_st_100ms, CoEng_st_100ms, Com_rTSC1VRVCURtdrTq_100ms và Com_rTSC1VRRDTrqReq_100ms: đều có giá trị không biến đổi cho tất cả quan sát. Điều này có thể chỉ ra rằng chúng là các biến hằng số và không đóng góp nhiều thông tin cho quá trình dự đoán. Do đó, có thể xem xét loại bỏ

chúng khỏi tập dữ liệu để giảm kích thước của không gian đặc trưng và cải thiện hiệu suất của mô hình mà không làm mất thông tin quan trọng.

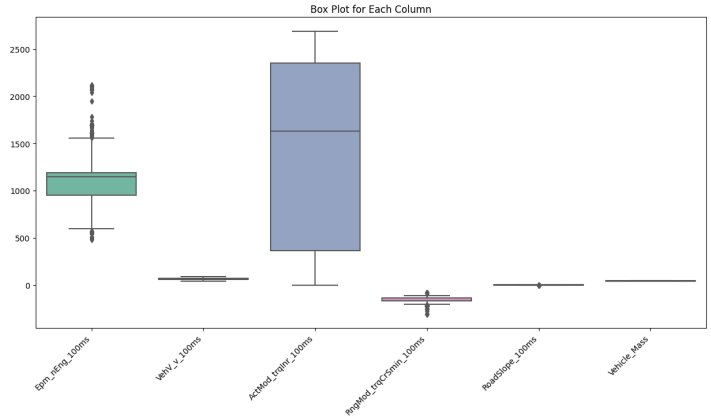
1.4. Phân phối của dữ liệu:

Vẽ biểu đồ histogram để quan sát phân phối của dữ liệu:



Hình 4: Biểu đồ histogram biểu diễn phân phối cho từng đặc trưng

Vẽ biểu đồ boxplot cho các cột của dữ liệu:



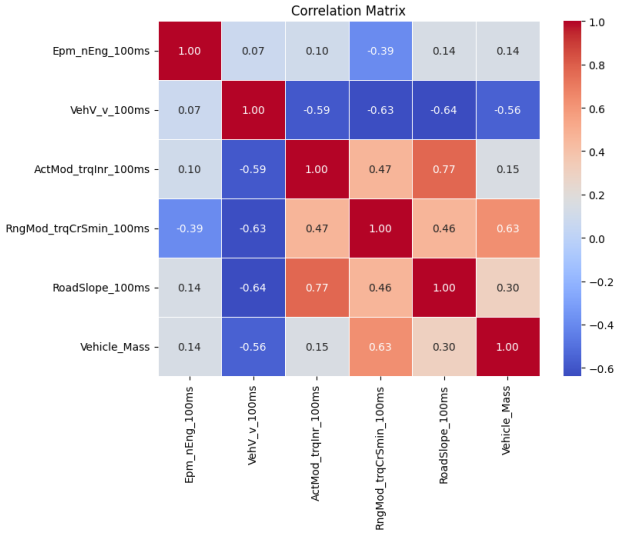
Hình 5: Biểu đồ box plot biểu diễn phân phối cho từng đặc trưng

Nhận xét:

- Ta có thể thấy biến Epm_nEng_100ms và RngMod_trqCrSmin_100ms có nhiều giá trị ngoại lệ nhưng không nên loại bỏ các giá trị ngoại lệ này do có thể động cơ xe phải làm việc dưới nhiều điều kiện khác nhau về độ cao, áp suất, nhiệt độ,...
- Dữ liệu chứa nhiều ngoại lệ nên việc chuẩn hóa thông thường như Standard Scaling hoặc Z-score normalization có thể ảnh hưởng đến độ chính xác của mô hình.
- Do vậy, chúng ta có thể cân nhắc sử dụng Robust Scaling vì Robust Scaling sử dụng phân vị thay vì giá trị trung bình và độ lệch chuẩn như trong chuẩn hóa thông thường. Điều này làm giảm ảnh hưởng của các giá trị cực lớn hoặc cực nhỏ đến quá trình chuẩn hóa.
- Nhãn classification khá lệch nên có thể cân nhắc sử dụng các mô hình phân loại phù hợp.

1.5. Tính chất tương quan:

Do dữ liệu chứa nhiều ngoại lệ nên chúng ta sử dụng hệ số tương quan Spearman thay vì hệ số tương quan Pearson.



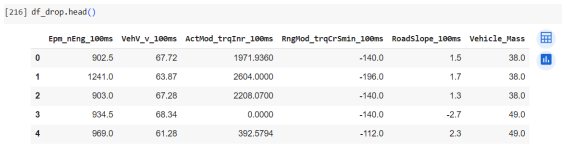
Hình 6: Ma trận tương quan giữa các đặc trưng và target

Nhận xét:

- Epm_nEng_100ms và RngMod_trqCrSmin_100ms: Có một mối quan hệ tiêu cực mạnh (-0.391924), cho thấy vòng tua động cơ giảm khi mô-men xoắn cần thiết để vận hành tăng.
- VehV_v_100ms và ActMod_trqInr_100ms: Mối quan hệ tiêu cực mạnh (-0.589086), có thể chỉ ra rằng vận tốc giảm khi mô-men xoắn của động cơ tăng.
- VehV_v_100ms và RngMod_trqCrSmin_100ms: Mối quan hệ tiêu cực mạnh (-0.630903), cho thấy vận tốc giảm khi mô-men xoắn cần thiết để vận hành tăng.
- ActMod_trqInr_100ms và RoadSlope_100ms: Mối quan hệ rất mạnh (0.770068), có thể chỉ ra rằng mô-men xoắn của động cơ tăng khi độ dốc đường tăng.
- RngMod_trqCrSmin_100ms và RoadSlope_100ms: Mối quan hệ tích cực (0.463229), có thể chỉ ra rằng mô-men xoắn cần thiết để vận hành tăng khi độ dốc đường tăng.
- RngMod_trqCrSmin_100ms và Vehicle_Mass: Mối quan hệ tương đối mạnh (0.629106), cho thấy mô-men xoắn cần thiết để vận hành tăng khi khối lượng của phương tiện tăng. Ta có thể sử dụng các mối quan hệ này cho bước feature engineering.

*2) **Lựa chọn đặc trưng**:* Trong phần thống kê mô tả, ta nhận thấy một số đặc trưng có giá trị không đổi trong mọi quan sát. Vì vậy, chúng không đóng góp nhiều thông tin có ý nghĩa cho quá trình dự đoán. Do vậy, trong bước này, ta tiến hành loại bỏ các đặc trưng có giá trị là hằng số.

*3) **Thiết kế đặc trưng**:* Dựa vào kết quả hệ số tương quan ở trên, ta thấy RngMod_trqCrSmin_100ms tương quan dương mạnh với biến target RoadSlope_100ms, mà RngMod_trqCrSmin_100ms lại tương quan âm mạnh với VehV_v_100ms. Do vậy, ta có thể cân nhắc tạo ra 1 biến mới kết hợp giữa RngMod_trqCrSmin_100ms và VehV_v_100ms.



[216] df_drop.head()

	Epm_nEng_100ms	VehV_v_100ms	ActMod_trqInr_100ms	RngMod_trqCrSmin_100ms	RoadSlope_100ms	Vehicle_Mass
0	902.5	67.72	1971.9360	-140.0	1.5	38.0
1	1241.0	63.87	2604.0000	-196.0	1.7	38.0
2	903.0	67.28	2208.0700	-140.0	1.3	38.0
3	934.5	68.34	0.0000	-140.0	-2.7	49.0
4	969.0	61.28	392.5794	-112.0	2.3	49.0

Hình 7: Dataframe sau khi drop các feature hằng số

Bảng I: Kết quả thử nghiệm feature engineering

Feature	Độ chính xác
Combined_VehV_Rng	0.7742
Combined_VehV_Act	0.7450
Combined_VehV_Rng và Combined_VehV_Act	0.7651

Tương tự, ta cũng có thể tạo 1 biến kết hợp giữa ActMod_trqInr_100ms và VehV_v_100ms.

```
[201] def feature_engineering(X):
        X['Combined_VehV_Rng'] = 1 / X['VehV_v_100ms'] * X['RngMod_trqCrSmin_100ms']
        X['Combined_VehV_Act'] = 1 / X['VehV_v_100ms'] * X['ActMod_trqInr_100ms']
        return X
```

Hình 8: Kết hợp các biến tương quan mạnh để tạo ra biến mới

Tiến hành thử nghiệm 2 feature này với mô hình tốt nhất ở phần mô hình, ta có kết quả như trong bảng I (với độ đo được cung cấp trong đề bài). Ta có thể thấy, đặc trưng Combined_VehV_Rng cải thiện được hiệu suất của mô hình nhưng đặc trưng Combined_VehV_Act làm giảm hiệu suất của mô hình. Do vậy, chúng ta chỉ giữ lại đặc trưng Combined_VehV_Rng cho các bước tiếp theo.

*4) **Tách tập dữ liệu**:* Tiến hành chia tập dữ liệu thành tập huấn luyện và tập kiểm tra để đánh giá hiệu suất của mô hình. Ở bước này, chúng ta sử dụng thư viện train_test_split của sklearn với tỉ lệ tập test = 0.2 và random_state = 42

*5) **Chuẩn hóa dữ liệu**:* Như đã đề cập ở trên, chúng ta sẽ sử dụng Robust Scaling để chuẩn hóa tập X_train và X_test.

```
rb_scaler = RobustScaler().fit(X_train)
X_train = rb_scaler.transform(X_train)
X_test = rb_scaler.transform(X_test)
```

```
[218] X_train[0], X_test[0]

(array([0.04149798, 0.28400913, 0.29442455, 0.        , 0.24993703,
        0.21610902]),
 array([ 0.63461538, -1.01351589, -0.80592757,  1.        , -0.51973411,
        -0.77697309]))
```

Hình 9: Tập dữ liệu sau khi áp dụng Robust Scaling

B. Mô hình

Trong phần này sẽ trình bày quá trình thử nghiệm và đánh giá các mô hình tuân theo độ đo được cung cấp.

*1) **Mô hình hồi quy**:* Ở phần này, chúng ta sẽ thử nghiệm và đánh giá một số mô hình hồi quy để dự đoán biến RoadSlope_100ms.

1.1. Lựa chọn mô hình Thử train dữ liệu (chưa feature engineering) với nhiều mô hình khác nhau để tìm ra mô hình phù hợp với dữ liệu. Một số mô hình được sử dụng trong quá trình thử nghiệm:

Bảng II: Kết quả thử nghiệm mô hình hồi quy

Feature	Độ chính xác
KNN	0.3329
Random Forest	0.1508
Decision Tree	0.7069
Gradient Boosting	-0.3388

Nhận xét:

- K-Nearest Neighbors (KNN) Regressor
- Random Forest Regressor
- Decision Tree Regressor
- Gradient Boosting Regressor

```
[30] knn_regressor.fit(X_train, y_train['RoadSlope_100ms'])
     rf_regressor.fit(X_train, y_train['RoadSlope_100ms'])
     dt_regressor.fit(X_train, y_train['RoadSlope_100ms'])
     gb_regressor.fit(X_train, y_train['RoadSlope_100ms'])
```

▾ GradientBoostingRegressor
GradientBoostingRegressor()

Hình 10: Thử nghiệm với một số mô hình hồi quy

Kết quả sau khi thử nghiệm được trình bày trong bảng II. Có thể thấy, 2 mô hình hoạt động tốt nhất là KNN và Decision Tree. Chúng ta tiếp tục fine-tuning 2 mô hình này để tìm ra mô hình tốt nhất.

1.2. Điều chỉnh tham số Sử dụng GridSearchCV để tìm ra tham số tốt nhất cho cả 2 mô hình KNN và Decision Tree.

Param grid đối với mô hình KNN:

$$\text{param\_grid} = \{ \tag{1}$$
$$\text{'n\_neighbors'} : [1, 5], \tag{2}$$
$$\text{'weights'} : ['uniform', 'distance'], \tag{3}$$
$$\text{'p'} : [1, 2], \tag{4}$$
$$\text{'leaf\_size'} : [5, 10, 15] \tag{5}$$
$$\} \tag{6}$$

Kết quả thu được mô hình KNN với tham số tốt nhất là:

- 'n_neighbors': 1,
- 'weights': 'uniform' hoặc 'distance'
- 'p': 1,
- 'leaf_size': 10

Tương tự, param grid với mô hình Decision Tree:

$$\text{param\_grid} = \{ \tag{7}$$
$$\text{'max\_depth'} : [None, 5, 10], \tag{8}$$
$$\text{'min\_samples\_split'} : [2, 4, 6], \tag{9}$$
$$\text{'min\_samples\_leaf'} : [1, 2, 4], \tag{10}$$
$$\} \tag{11}$$

Kết quả thu được mô hình Decision Tree với tham số tốt nhất là:

Bảng III: Kết quả thử nghiệm mô hình hồi quy sau khi fine-tune

	Với tham số mặc định	**Với tham số GridSearchCV**
KNN	0.33	0.76
Decision Tree	0.70	0.71

Bảng IV: Kết quả thử nghiệm mô hình phân loại

Mô hình	**Độ chính xác**
Random Forest	0.9991
Gradient Boosting	0.9952
Bagging with Decision Tree	0.9943
Boosting with Decision Tree	0.9925

- 'max_depth': None,
- 'min_samples_leaf': 1,
- 'min_samples_split': 2

Fit 2 mô hình này với dữ liệu cho kết quả ở bảng III. Nhìn chung, mô hình tốt nhất là KNN Regressor sau khi điều chỉnh tham số với GridSearchCV.

*2) **Mô hình phân loại**:* Ở phần này, chúng ta sẽ thử nghiệm và đánh giá một số mô hình phân loại để dự đoán biến Vehicle_Mass.

Thử nghiệm với một số mô hình classification như:

- Random Forest
- Gradient Boosting
- SVM

Bên cạnh đó, vì dữ liệu bị lệch nên chúng ta sẽ sử dụng Bagging hoặc Boosting để tạo ra một mô hình mạnh mẽ và cân bằng hiệu suất trên dữ liệu bị lệch. Kết quả thu được sau khi chạy các mô hình classification được trình bày trong bảng IV.

Random Forest cho kết quả bị overfitting trong khi các phương pháp Bagging và Boosting cho kết quả ổn định hơn. Do vậy, chúng ta có thể cân nhắc sử dụng Bagging với base là Decision Tree (0.994).

Ngoài ra, kết quả phân loại cũng khá tốt (đều trên 0.98) nên không cần thiết phải thực hiện thêm việc điều chỉnh tham số do có thể dẫn đến overfitting.

III. Kết quả

Sau khi thực hiện Feature Engineering, scale giá trị với Robust Scaling và đưa vào 2 mô hình Regression (KNN đã GridSearch) và Classification (Bagging base Decision Tree) tốt nhất, chúng ta thu được kết quả như trong bảng V.

IV. Kết luận

Dựa vào sự tương quan mạnh mẽ giữa biến Rng-Mod_trqCrSmin_100ms và VehV_v_100ms, việc tạo ra biến mới Combined_VehV_Rng đã mang lại cải thiện đáng kể trong hiệu suất dự đoán của mô hình hồi quy. Sự kết hợp giữa tạo biến mới và sử dụng GridSearchCV đã giúp chúng tôi tìm ra mô hình tối ưu, và các chỉ số đánh giá đều cho thấy hiệu suất tích cực. Tuy nhiên, dù kết quả đánh giá trên public test có vẻ khả quan nhưng kết quả trên private test lại giảm đáng kể, chứng tỏ mô hình chưa được tổng quát hóa và cần nhiều sự cải thiện trong tương lai.

Bảng V: Kết quả thu được trên tập train, public test và private test

	Regression	**Classification**	**Combined**
Train	0.7742	0.994	1.7682
Public test	1.00	0.998222	1.99822
Private test	0.994533	0.793903	0.864124

Tài liệu

Lecture Slides

https://neptune.ai/blog/ensemble-learning-guide