Chapter 9 - Ex5: Price Prediction

The car company wants to enter a new market and needs an estimation of exactly which variables affect the car prices.
The goal is:

Which variables are significant in predicting the price of a car
How well do those variables describe the price of a car

Attribute Information:

Attribute: Attribute Range About this file

1. model - > Ford Car Brands
2. year - >Production Year
3. price - >Price of car in USD
4. transmission - > Automatic,Manual, Semi-Auto
5. mileage -> Number of miles traveled
6. fuel_Type -> Petrol,Diesel,Hybrid,Electric,Other
7. tax -> Annual Tax
8. mpg - > Miles per Gallon
9. engineSize - > Car's Engine Size

For more information: https://www.kaggle.com/datasets/adhurimquku/ford-car-price-prediction (https://www.kaggle.com/datasets/adhurimquku/ford-car-price-prediction)

Requirements:

Dự đoán giá xe dựa trên bộ dữ liệu ford (train.csv có 14370 mẫu và test.csv có 3596 mẫu)

Yêu cầu: Hãy đọc dữ liệu, áp dụng Linear Regression để thực hiện dự đoán giá xe dựa trên những thông tin được cung cấp.

1. Đọc dữ liệu train.csv, tiền xử lý dữ liệu nếu cần
2. Tạo X_train, X_test, y_train, y_test từ dữ liệu ở câu 1 với tỷ lệ dữ liệu test là 0.2
3. Áp dụng thuật toán Linear Regression: fit model, tìm độ chính xác, đánh giá mô hình bằng kiểm tra underfiting và overfiting?
4. Đọc dữ liệu test.csv. Tiền xử lý dữ liệu như train.csv. Tìm kết quả cho dữ liệu test.
5. Ghi kết quả vào file test_pred.csv
6. Áp dụng Pipeline. Lưu kết quả khi áp dụng Pipeline vào file test_pred.csv (thêm 1 cột kết quả mới)

1. EDA

1.1. Read data train.csv

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
df = pd.read_csv("ford/train.csv")
```

In [3]:
```
df.shape
```

Out[3]: (14370, 9)

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14370 entries, 0 to 14369
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   model         14370 non-null  object 
 1   year          14370 non-null  int64  
 2   price         14370 non-null  int64  
 3   transmission  14370 non-null  object 
 4   mileage       14370 non-null  int64  
 5   fuelType      14370 non-null  object 
 6   tax           14370 non-null  int64  
 7   mpg           14370 non-null  float64
 8   engineSize    14370 non-null  float64
dtypes: float64(2), int64(4), object(3)
memory usage: 1010.5+ KB
```

In [5]:
```
# Nhận xét: không có dữ liệu null
# các cột đúng kiểu dữ liệu
```

In [6]:
```
df.head()
```

Out[6]:

	model	year	price	transmission	mileage	fuelType	tax	mpg	engineSize
0	Fiesta	2017	8498	Manual	28637	Diesel	0	78.5	1.5
1	Fiesta	2013	5490	Manual	45740	Petrol	30	54.3	1.2
2	Fiesta	2018	13181	Semi-Auto	10414	Petrol	145	48.7	1.0
3	B-MAX	2014	7998	Manual	9181	Petrol	30	55.4	1.0
4	EcoSport	2016	7890	Manual	30035	Diesel	30	64.2	1.5

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
TRUNG TÂM TIN HỌC

In [7]:
```
# xem thông tin mô tả các cột số
```

In [8]:
```
df.describe()
```

Out[8]:

	year	price	mileage	tax	mpg	engineSize
count	14370.000000	14370.000000	14370.000000	14370.000000	14370.000000	14370.000000
mean	2016.869311	12270.363814	23333.277801	113.133403	57.929910	1.350292
std	2.056454	4747.275442	19427.195845	62.388703	10.176018	0.434927
min	1996.000000	675.000000	1.000000	0.000000	20.800000	0.000000
25%	2016.000000	8999.000000	9995.250000	30.000000	52.300000	1.000000
50%	2017.000000	11250.000000	18216.500000	145.000000	58.900000	1.200000
75%	2018.000000	15299.000000	31000.750000	145.000000	65.700000	1.500000
max	2060.000000	54995.000000	174000.000000	580.000000	201.800000	5.000000

In [9]:
```
# xem thông tin mô tả các cột object
```

In [10]:
```
df.describe(include='object')
```

Out[10]:

	model	transmission	fuelType
count	14370	14370	14370
unique	24	3	5
top	Fiesta	Manual	Petrol
freq	5233	12403	9746

In [11]:
```
# loại bỏ ký tự khoảng trắng ở đầu chuỗi của cột 'model'
```

In [12]:
```
df['model'] = df['model'].str.strip()
```

In [13]:
```
df.describe(include='object')
```

Out[13]:

	model	transmission	fuelType
count	14370	14370	14370
unique	23	3	5
top	Fiesta	Manual	Petrol
freq	5233	12403	9746

In [14]:
```
df_cat = df.select_dtypes('object')
cols_cat = df_cat.columns
cols_cat
```

Out[14]:
```
Index(['model', 'transmission', 'fuelType'], dtype='object')
```

In [15]:
```
cols_cont = list(set(df.columns) - set(cols_cat))
cols_cont
```

Out[15]:
```
['tax', 'engineSize', 'price', 'year', 'mileage', 'mpg']
```

1.2. Data Visualization

1.2.1. Single variables

1.2.1.1 Category variables

In [16]:
```
from analysis.analyzer import TTTH_Analyzer
_analyzer = TTTH_Analyzer()
```

In [17]:
```
for col in cols_cat:
    print('*' *50)
    print('Phân tích biến', col)
    _analyzer.analyze_category_variable(variable_name=col, df=df)
```

```
**************************************************
Phân tích biến model
Class count of model:

==========
Fiesta                 5233
Focus                  3648
Kuga                   1752
EcoSport                932
C-MAX                   454
Ka+                     435
Mondeo                  422
B-MAX                   286
S-MAX                   238
Grand C-MAX             202
Galaxy                  173
KA                      165
Edge                    164
Puma                     68
```

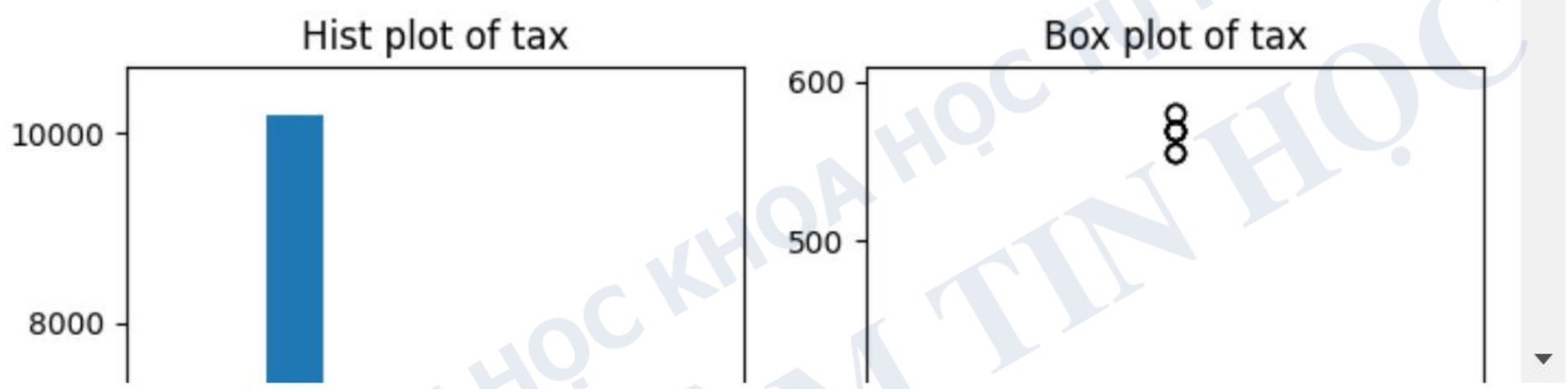
In [18]:
```
# Nhận xét: Các biến phân bố không chuẩn, tập trung vào 1 vài giá trị của từng t
```

1.2.1.2 Continuous variables

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
TRUNG TÂM TIN HỌC

In [19]:
```
for col in cols_cont:
    print('_'*50)
    print('phân tích biến: ', col)
    _analyzer.analyze_numeric_variable(variable_name=col, df=df)
```

```
__________________________________________________
phân tích biến:  tax
=====
central tendency of tax: {'mean': 113.13340292275574, 'median': 145.0, 'mod
e': 145, 'min': 0, 'max': 580, 'range': 580}
=====
Dispersion of tax:
 {'range': 580, 'q1': 30.0, 'q3': 145.0, 'iqr': 115.0, 'var': 3892.3502864915
495, 'skew': -0.5094147962884833, 'kurtosis': 1.5117421211461028}
=====
```



Nhận xét:

- Tất cả biến liên tục đều có phân phối không chuẩn
- Có nhiều outliers
- Thang đo giữa các biến chênh lệch lớn

1.2.2 Multiple variables

1.2.2.1 Category variables với Output

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
TRUNG TÂM TIN HỌC

In [21]:
```
col2 = 'price'
for i in range(0,len(cols_cat)):
    col1 = cols_cat[i]
    print('_'*50)
    print('phân tích 2 biến cat-cont: ', col1, ' và ', col2)
    _analyzer.analyze_continous_vs_categories(continous_var=col2,
                                              category_vars=col1,
                                              df=df)
```

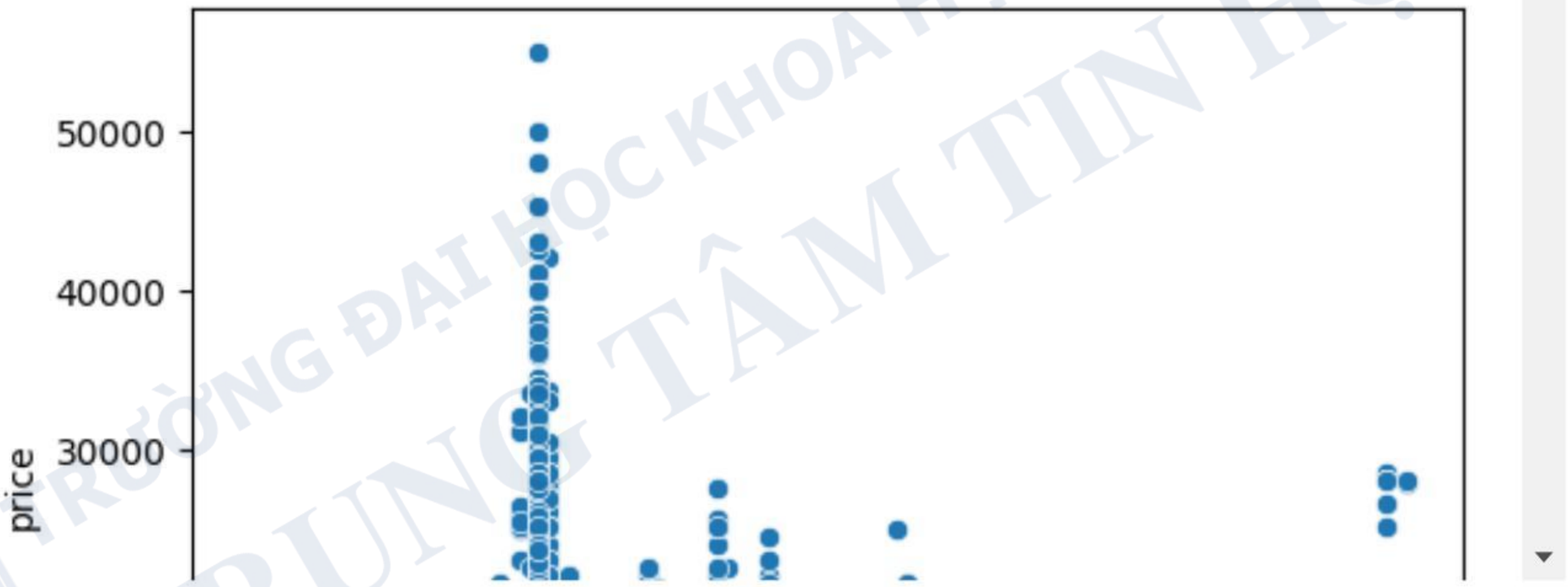
```
__________________________________________________
phân tích 2 biến cat-cont:  model  và  price
                sum_sq       df           F  PR(>F)
C(model)  1.375339e+11     22.0  481.445654     0.0
Residual  1.862949e+11  14347.0         NaN     NaN
======
```

In [22]:
```
# các biến category có ảnh hưởng tới price
```

1.2.2.2 Continous variables với Output

In [23]:
```
col2 = 'price'
for i in range(0,len(cols_cont)):
    col1 = cols_cont[i]
    print('_'*50)
    if col1 != col2:
        print('phân tích biến ', col1, ' và output', col2)
        df = df.dropna(axis=0, subset=[col1, col2])
        print('bảng phương sai: ', df[[col1, col2]].corr())
        sns.scatterplot(data=df, x=col1, y=col2)
        plt.show()
        sns.pairplot(df[[col1, col2]])
        plt.show()
```

```
__________________________________________________
phân tích biến  tax  và output price
bảng phương sai:              tax     price
tax    1.000000  0.409687
price  0.409687  1.000000
```



In [24]:
```
# các biến continous có ảnh hưởng tới price
```

1.2.2.3 Continous variables vs Continous variables

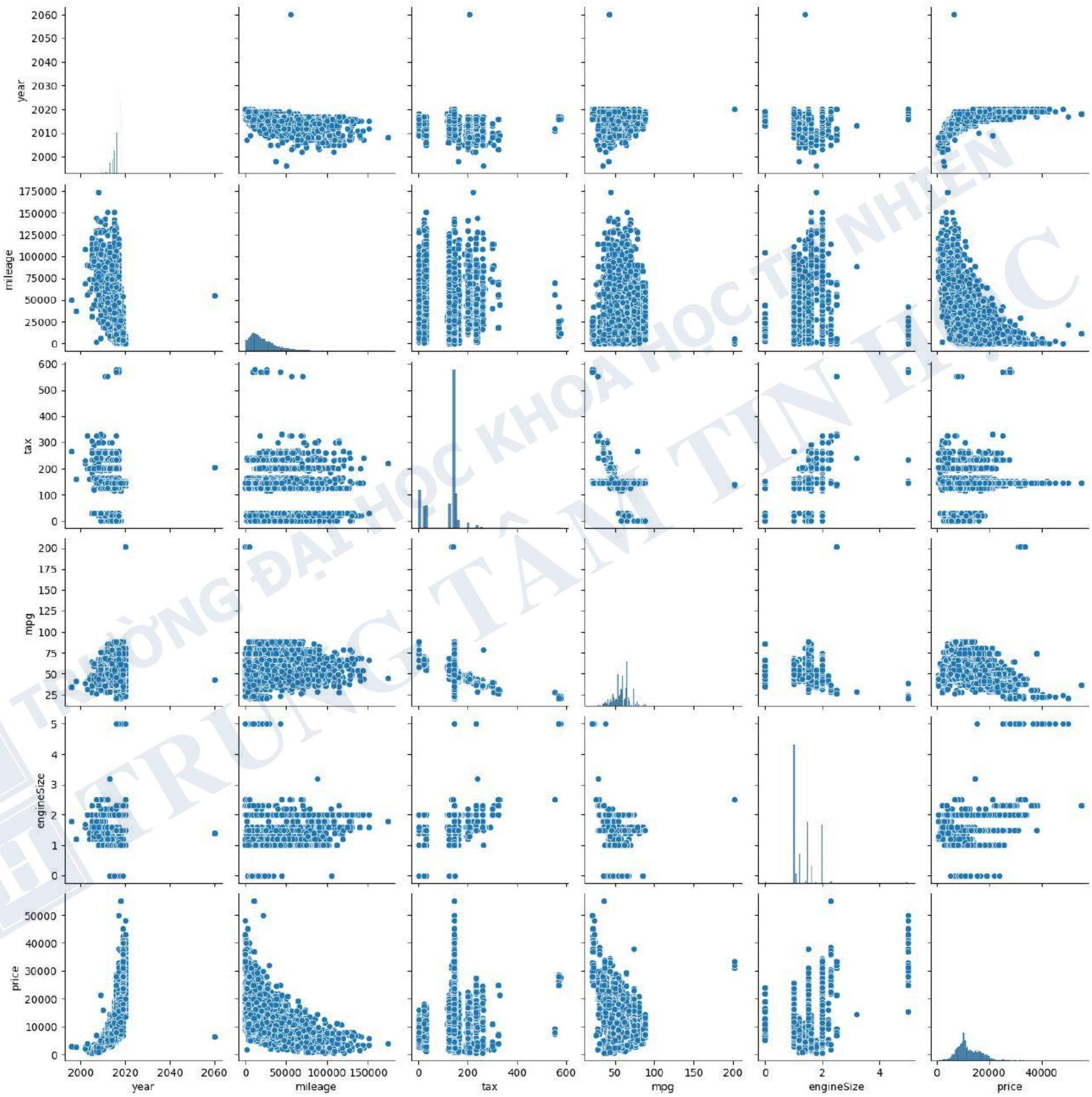
In [25]:
```
df[cols_cont].corr()
```

Out[25]:

	tax	engineSize	price	year	mileage	mpg
tax	1.000000	0.186671	0.409687	0.301014	-0.263182	-0.503602
engineSize	0.186671	1.000000	0.414711	-0.132079	0.210307	-0.260038
price	0.409687	0.414711	1.000000	0.634671	-0.530423	-0.348553
year	0.301014	-0.132079	0.634671	1.000000	-0.701774	-0.025817
mileage	-0.263182	0.210307	-0.530423	-0.701774	1.000000	0.123335
mpg	-0.503602	-0.260038	-0.348553	-0.025817	0.123335	1.000000

In [26]:
```
# các biến số đều có tương quan với 'price' (abs(corr) > 0.3), nhưng mức độ tương
# các biến input có tương quan với nhau nhưng mức độ tương quan không mạnh (trừ
# số lượng biến ít nên cân nhắc giữ lại hết
```

In [27]:
```
sns.pairplot(df[['year', 'mileage', 'tax', 'mpg', 'engineSize', 'price']])
plt.show()
```



2. Preprocessing & Build model

2.1 Preprocessing

In [28]:
```
df.model.value_counts()
```

Out[28]:
```
Fiesta                5233
Focus                 3648
Kuga                  1752
EcoSport               932
C-MAX                  454
Ka+                    435
Mondeo                 422
B-MAX                  286
S-MAX                  238
Grand C-MAX            202
Galaxy                 173
KA                     165
Edge                   164
Puma                    68
Tourneo Custom          56
Mustang                 49
Grand Tourneo Connect   46
Tourneo Connect         27
Fusion                  15
Streetka                 2
Escort                   1
Transit Tourneo          1
Ranger                   1
Name: model, dtype: int64
```

In [29]:
```
# cập nhật lại cột 'model': 13 giá trị xuất hiện nhiều nhất thì giữ nguyên, các
```

In [30]:
```
df['model'] = df['model'].map(
    lambda x: x if x in (df.model.value_counts()[:13].index.tolist()) else 'Othe
```

In [31]:
```
df.model.value_counts()
```

Out[31]:
```
Fiesta          5233
Focus           3648
Kuga            1752
EcoSport         932
C-MAX            454
Ka+              435
Mondeo           422
B-MAX            286
Other            266
S-MAX            238
Grand C-MAX      202
Galaxy           173
KA               165
Edge             164
Name: model, dtype: int64
```

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
TRUNG TÂM TIN HỌC

In [32]:
```
df.fuelType.value_counts()
```

Out[32]:
```
Petrol      9746
Diesel      4601
Hybrid        21
Electric       1
Other          1
Name: fuelType, dtype: int64
```

In [33]:
```
# cập nhật lại cột 'fuelType': 2 giá trị xuất hiện nhiều nhất thì giữ nguyên, cá
```

In [34]:
```
df['fuelType'] = df['fuelType'].map(
    lambda x: x if x in (df.fuelType.value_counts()[:2].index.tolist()) else 'Otl
```

In [35]:
```
df.fuelType.value_counts()
```

Out[35]:
```
Petrol    9746
Diesel    4601
Other       23
Name: fuelType, dtype: int64
```

In [36]:
```
# cột 'transmission' là biến phân loại có thứ tự:
# tiến hành đổi giá trị sang dạng số: 'Manual' => 0; 'Semi-Auto' => 1; 'Automati
```

In [37]:
```
df['transmission'] = df['transmission'].map(
    lambda x: 0 if x == 'Manual' else (1 if x == 'Semi-Auto' else 2))
```

In [38]:
```
# các biến 'model', 'fuelType' là các biến phân loại không có thứ tự
df_now= pd.get_dummies(data = df,
                       columns = ['model', 'fuelType'], drop_first=True)
df_now.head()
```

Out[38]:

	year	price	transmission	mileage	tax	mpg	engineSize	model_C-MAX	model_EcoSport	model_I
0	2017	8498	0	28637	0	78.5	1.5	0	0	
1	2013	5490	0	45740	30	54.3	1.2	0	0	
2	2018	13181	1	10414	145	48.7	1.0	0	0	
3	2014	7998	0	9181	30	55.4	1.0	0	0	
4	2016	7890	0	30035	30	64.2	1.5	0	1	

5 rows × 22 columns

In [39]:
```
# Tạo 2 tập dữ liệu input & output
x = df_now.drop('price',axis=1)
y = df_now['price']
```

In [40]:
```
x.head()
```

Out[40]:

	year	transmission	mileage	tax	mpg	engineSize	model_C-MAX	model_EcoSport	model_Edge	n
0	2017	0	28637	0	78.5	1.5	0	0	0	
1	2013	0	45740	30	54.3	1.2	0	0	0	
2	2018	1	10414	145	48.7	1.0	0	0	0	
3	2014	0	9181	30	55.4	1.0	0	0	0	
4	2016	0	30035	30	64.2	1.5	0	1	0	

5 rows × 21 columns

In [41]:
```
y.head()
```

Out[41]:
```
0     8498
1     5490
2    13181
3     7998
4     7890
Name: price, dtype: int64
```

2.2 Build & Test model

In [42]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error
import math
```

In [43]:
```
# Tạo tập dữ liệu train & test
x_train, x_test, y_train, y_test = train_test_split(x, y,
                                                    test_size=0.2,
                                                    random_state=0)
x_train.shape, x_test.shape
```

Out[43]:
```
((11496, 21), (2874, 21))
```

In [44]:
```
# giá trị các cột chênh lệch lớn
# có outliers và  không có phân phối chuẩn
# sử dụng RobustScaler
```

In [45]:
```
from sklearn.preprocessing import RobustScaler
```

In [46]:
```
scaler = RobustScaler()
```

In [47]:
```
# scaling
x_train_sc = scaler.fit_transform(x_train)

x_test_sc = scaler.transform(x_test)
```

In [48]:
```
# Khởi tạo model
model = LinearRegression()
# huấn luyện model
model.fit(x_train_sc,y_train)
```

Out[48]:
```
LinearRegression()
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [49]:
```
model.intercept_
```

Out[49]:
```
9919.868415600473
```

In [50]:
```
model.coef_
```

Out[50]:
```
array([ 2095.63462969,   196.331569  , -1464.41230948,  -168.8295403 ,
        -973.53079365,  1630.67729335,  1180.60695331,  1838.44098987,
        9286.69136305,  1138.36487448,  3193.76921646,  6921.25230393,
        1723.36958174,  -477.71885701, -2965.43627835,  3812.06692307,
        3056.9288787 ,  6889.35995192,  6377.78125728,  5016.75844524,
         253.85952049])
```

In [51]:
```
# Dự đoán
yhat_train = model.predict(x_train_sc)
yhat_test = model.predict(x_test_sc)
```

In [52]:
```
print('Score train: ', r2_score(y_train, yhat_train))
print('Score test: ', r2_score(y_test, yhat_test))
```

```
Score train:  0.839757464687812
Score test:  0.828899457543307
```

In [53]:
```
# Mô hình trên có score của train và test gần như nhau
# và khoảng 83%: không bị overfitting và underfiiting
```

In [54]:
```
df.price.std()
```

Out[54]:
```
4747.275442306809
```

In [55]:
```
RMSE_train = math.sqrt(mean_squared_error(y_train, yhat_train))
RMSE_train
```

Out[55]:
```
1904.236408884038
```

In [56]:
```
RMSE_test = math.sqrt(mean_squared_error(y_test, yhat_test))
RMSE_test
```

Out[56]: 1947.13584306267

In [57]:
```
# RMSE ~ 40%std => có thể sử dụng model này để dự đoán price được
```

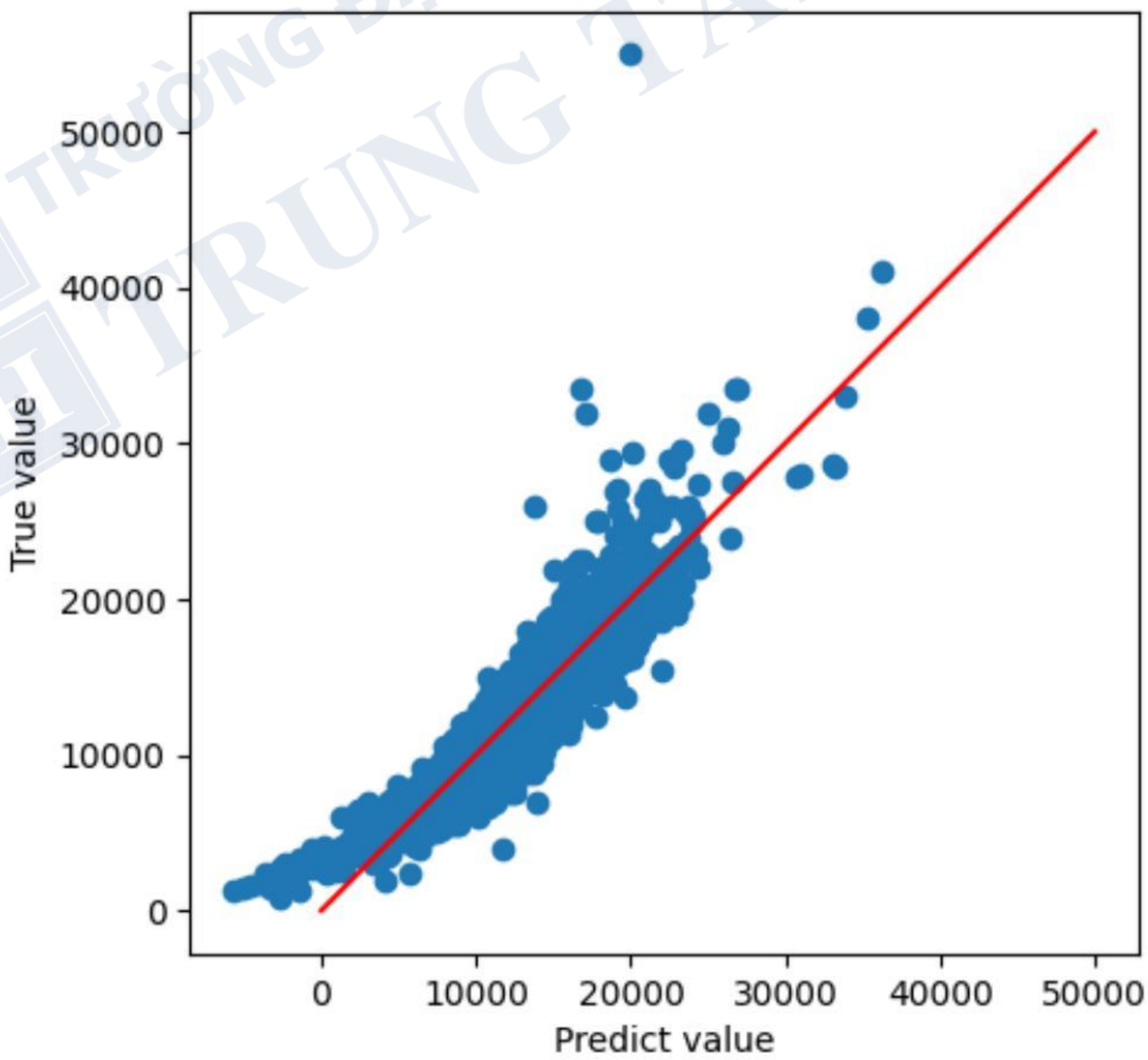
In [58]:
```
MAE_train = mean_absolute_error(y_train, yhat_train)
MAE_train
```

Out[58]: 1378.5465733225863

In [59]:
```
MAE_test = mean_absolute_error(y_test, yhat_test)
MAE_test
```

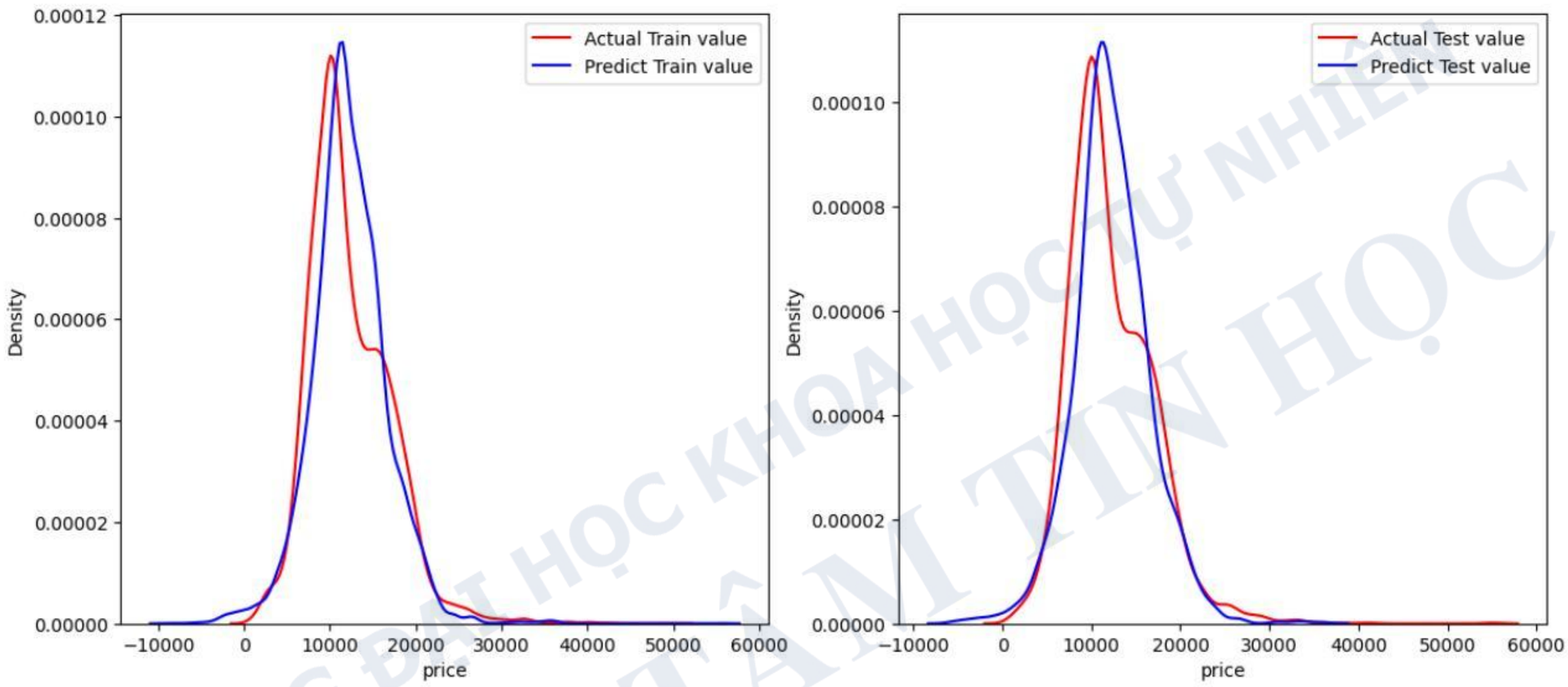
Out[59]: 1368.339697714957

In [60]:
```
plt.figure(figsize=(5,5))
plt.scatter(yhat_test, y_test)
plt.xlabel('Predict value')
plt.ylabel('True value')
plt.plot([0,50000], [0,50000], 'k-', color='r')
plt.show()
```



TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
TRUNG TÂM TIN HỌC

In [61]:
```
plt.figure(figsize=(14,6))
plt.subplot(1,2,1)
ax1 = sns.kdeplot(y_train, color='r', label='Actual Train value')
sns.kdeplot(yhat_train, color='b', label='Predict Train value', ax=ax1)
plt.legend()
plt.subplot(1,2,2)
ax2 = sns.kdeplot(y_test, color='r', label='Actual Test value')
sns.kdeplot(yhat_test, color='b', label='Predict Test value', ax=ax2)
plt.legend()
plt.show()
```



2.3 Make prediction on Test data

In [63]:
```
df_test = pd.read_csv("ford/test.csv")
```

In [64]:
```
df_test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3596 entries, 0 to 3595
Data columns (total 8 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   model         3596 non-null   object 
 1   year          3596 non-null   int64  
 2   transmission  3596 non-null   object 
 3   mileage       3596 non-null   int64  
 4   fuelType      3596 non-null   object 
 5   tax           3596 non-null   int64  
 6   mpg           3596 non-null   float64
 7   engineSize    3596 non-null   float64
dtypes: float64(2), int64(3), object(3)
memory usage: 224.9+ KB
```

In [65]:
```
df_test.describe()
```

Out[65]:

	year	mileage	tax	mpg	engineSize
count	3596.000000	3596.000000	3596.000000	3596.000000	3596.000000
mean	2016.855117	23479.818409	114.112903	57.815350	1.352864
std	2.025945	19652.574849	60.487749	9.922919	0.422036
min	2000.000000	1.000000	0.000000	22.100000	0.000000
25%	2016.000000	9949.500000	125.000000	52.300000	1.000000
50%	2017.000000	18338.000000	145.000000	58.900000	1.200000
75%	2018.000000	31152.000000	145.000000	65.700000	1.600000
max	2020.000000	177644.000000	555.000000	201.800000	5.000000

In [66]:
```
df_test.head()
```

Out[66]:

	model	year	transmission	mileage	fuelType	tax	mpg	engineSize
0	B-MAX	2015	Semi-Auto	47360	Petrol	160	44.1	1.6
1	Ka+	2019	Manual	1553	Petrol	145	47.9	1.2
2	Focus	2019	Manual	7445	Petrol	145	58.9	1.0
3	Fiesta	2020	Manual	9	Petrol	150	55.4	1.0
4	Fiesta	2017	Manual	8606	Petrol	0	65.7	1.0

In [67]:
```
df_test['transmission'] = df_test['transmission'].map(
    lambda x: 0 if x == 'Manual' else (1 if x == 'Semi-Auto' else 2))
```

In [68]:
```
df_test['model'] = df_test['model'].str.strip()
```

In [69]:
```
df_test['model'] = df_test['model'].map(
    lambda x: x if x in (df_test.model.value_counts()[:13].index.tolist()) else
```

In [70]:
```
df_test['fuelType'] = df_test['fuelType'].map(
    lambda x: x if x in (df_test.fuelType.value_counts()[:2].index.tolist()) else
```

In [71]:
```
df_test.describe(include='object')
```

Out[71]:

	model	fuelType
count	3596	3596
unique	14	3
top	Fiesta	Petrol
freq	1324	2433

In [72]:
```
df_test_now= pd.get_dummies(data = df_test, columns = ['model', 'fuelType'], dro
df_test_now.head()
```

Out[72]:

	year	transmission	mileage	tax	mpg	engineSize	model_C-MAX	model_EcoSport	model_Edge	n
0	2015	1	47360	160	44.1	1.6	0	0	0	
1	2019	0	1553	145	47.9	1.2	0	0	0	
2	2019	0	7445	145	58.9	1.0	0	0	0	
3	2020	0	9	150	55.4	1.0	0	0	0	
4	2017	0	8606	0	65.7	1.0	0	0	0	

5 rows × 21 columns

In [73]:
```
X_test_now = df_test_now
X_test_now.head()
```

Out[73]:

	year	transmission	mileage	tax	mpg	engineSize	model_C-MAX	model_EcoSport	model_Edge	n
0	2015	1	47360	160	44.1	1.6	0	0	0	
1	2019	0	1553	145	47.9	1.2	0	0	0	
2	2019	0	7445	145	58.9	1.0	0	0	0	
3	2020	0	9	150	55.4	1.0	0	0	0	
4	2017	0	8606	0	65.7	1.0	0	0	0	

5 rows × 21 columns

In [74]:
```
X_test_sc = scaler.transform(X_test_now)
X_test_sc = pd.DataFrame(X_test_sc, columns=X_test_now.columns)
X_test_sc.head()
```

Out[74]:

	year	transmission	mileage	tax	mpg	engineSize	model_C-MAX	model_EcoSport	mo
0	-1.0	1.0	1.392282	0.130435	-1.104478	0.8	0.0	0.0	
1	1.0	0.0	-0.790043	0.000000	-0.820896	0.0	0.0	0.0	
2	1.0	0.0	-0.509338	0.000000	0.000000	-0.4	0.0	0.0	
3	1.5	0.0	-0.863602	0.043478	-0.261194	-0.4	0.0	0.0	
4	0.0	0.0	-0.454026	-1.260870	0.507463	-0.4	0.0	0.0	

5 rows × 21 columns

In [75]:
```
Yhat_test_now = model.predict(X_test_sc)
```

In [76]:
```
df_test_now.head()
```

Out[76]:

	year	transmission	mileage	tax	mpg	engineSize	model_C-MAX	model_EcoSport	model_Edge	n
0	2015	1	47360	160	44.1	1.6	0	0	0	
1	2019	0	1553	145	47.9	1.2	0	0	0	
2	2019	0	7445	145	58.9	1.0	0	0	0	
3	2020	0	9	150	55.4	1.0	0	0	0	
4	2017	0	8606	0	65.7	1.0	0	0	0	

5 rows × 21 columns

In [77]:
```
df_test_now.tail()
```

Out[77]:

	year	transmission	mileage	tax	mpg	engineSize	model_C-MAX	model_EcoSport	model_Edge
3591	2017	1	35680	145	52.3	2.0	0	0	0
3592	2008	0	116000	145	47.1	1.2	0	0	0
3593	2018	0	7223	145	60.1	1.0	0	0	0
3594	2015	0	30324	235	43.5	2.2	0	0	0
3595	2018	0	5343	145	60.1	1.0	0	0	0

5 rows × 21 columns

In [78]:
```
df_test['yhat_price'] = Yhat_test_now
df_test.head()
```

Out[78]:

	model	year	transmission	mileage	fuelType	tax	mpg	engineSize	yhat_price
0	B-MAX	2015	1	47360	Petrol	160	44.1	1.6	8339.453955
1	Ka+	2019	0	1553	Petrol	145	47.9	1.2	11006.182351
2	Focus	2019	0	7445	Petrol	145	58.9	1.0	15302.881859
3	Fiesta	2020	0	9	Petrol	150	55.4	1.0	15061.023316
4	Fiesta	2017	0	8606	Petrol	0	65.7	1.0	10789.684713

2.4 Pipeline

In [79]:
```
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import make_column_transformer
```

In [80]:
```
df.head()
```

Out[80]:

	model	year	price	transmission	mileage	fuelType	tax	mpg	engineSize
0	Fiesta	2017	8498	0	28637	Diesel	0	78.5	1.5
1	Fiesta	2013	5490	0	45740	Petrol	30	54.3	1.2
2	Fiesta	2018	13181	1	10414	Petrol	145	48.7	1.0
3	B-MAX	2014	7998	0	9181	Petrol	30	55.4	1.0
4	EcoSport	2016	7890	0	30035	Diesel	30	64.2	1.5

In [81]:
```
Input=[('column_tr', make_column_transformer((OneHotEncoder(),
                                              ['model', 'fuelType']),
                                             remainder='passthrough')),
       ('scaler', RobustScaler()),
       ('model', LinearRegression())]
```

In [82]:
```
pipe = Pipeline(Input)
```

In [83]:
```
X = df.drop('price', 1)
X.head()
```

Out[83]:

	model	year	transmission	mileage	fuelType	tax	mpg	engineSize
0	Fiesta	2017	0	28637	Diesel	0	78.5	1.5
1	Fiesta	2013	0	45740	Petrol	30	54.3	1.2
2	Fiesta	2018	1	10414	Petrol	145	48.7	1.0
3	B-MAX	2014	0	9181	Petrol	30	55.4	1.0
4	EcoSport	2016	0	30035	Diesel	30	64.2	1.5

In [84]:
```
y = df['price']
y.head()
```

Out[84]:
```
0     8498
1     5490
2    13181
3     7998
4     7890
Name: price, dtype: int64
```

In [85]:
```
X_train, X_test, y_train, y_test = train_test_split(X,
                                                    y,
                                                    test_size=0.2, random_state=
```

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
TRUNG TÂM TIN HỌC

In [86]:
```
pipe.fit(X_train, y_train)
```

Out[86]:
```
Pipeline(steps=[('column_tr',
                 ColumnTransformer(remainder='passthrough',
                                   transformers=[('onehotencoder',
                                                  OneHotEncoder(),
                                                  ['model', 'fuelType'])])),
                ('scaler', RobustScaler()), ('model', LinearRegression())])
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [87]:
```
pipe.predict(X_test)
```

Out[87]:
```
array([12119.375,  9678.875,  8939.125, ..., 10132.125, 14133.125,
       13586.5  ])
```

In [88]:
```
pipe.score(X_train, y_train)
```

Out[88]:
```
0.8397253458139918
```

In [89]:
```
pipe.score(X_test, y_test)
```

Out[89]:
```
0.8289646142081378
```

In [90]:
```
data_pipe = df_test[['model', 'year', 'transmission', 'mileage', 'fuelType', 'ta
```

In [91]:
```
y_new = pipe.predict(data_pipe)
```

In [92]:
```
# Lưu cả kết quả với Pipeline vào file (với 1 cột mới)
```

In [93]:
```
df_test['price_pipe'] = y_new
```

In [94]:
```
df_test.head()
```

Out[94]:

	model	year	transmission	mileage	fuelType	tax	mpg	engineSize	yhat_price	price_pipe
0	B-MAX	2015	1	47360	Petrol	160	44.1	1.6	8339.453955	8348.500
1	Ka+	2019	0	1553	Petrol	145	47.9	1.2	11006.182351	11010.625
2	Focus	2019	0	7445	Petrol	145	58.9	1.0	15302.881859	15290.125
3	Fiesta	2020	0	9	Petrol	150	55.4	1.0	15061.023316	15052.375
4	Fiesta	2017	0	8606	Petrol	0	65.7	1.0	10789.684713	10781.500

In [95]:
```
df_test["yhat_price"] = df_test["yhat_price"].apply(lambda x: round(x))
```

In [96]:
```
df_test.head()
```

Out[96]:

	model	year	transmission	mileage	fuelType	tax	mpg	engineSize	yhat_price	price_pipe
0	B-MAX	2015	1	47360	Petrol	160	44.1	1.6	8339	8348.500
1	Ka+	2019	0	1553	Petrol	145	47.9	1.2	11006	11010.625
2	Focus	2019	0	7445	Petrol	145	58.9	1.0	15303	15290.125
3	Fiesta	2020	0	9	Petrol	150	55.4	1.0	15061	15052.375
4	Fiesta	2017	0	8606	Petrol	0	65.7	1.0	10790	10781.500

In [97]:
```
df_test.to_csv('test_pred.csv', index = None)
```

In [ ]:
```

```

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
TRUNG TÂM TIN HỌC