Supervised Learning: Regression

IBM Machine Learning - Project 2
Thanh Huynh
March 2021

Main objective

- The main objective of this analysis is to predict price(£) of used Ford cars using a Linear Regression and different regularization regressions.
- This analysis attempts to try both train-test-split and cross-validation to have an overview of how these two methods can lead to different decisions in terms of model selection.
- The data set is split into three sets: training set (60%), validation set (20%), and test set (20%) for cross-validation purpose.

About the data

- The data set used in this analysis is a part of 100,000 UK Used Car Data Set published on Kaggle in July 2020 by a member (Aditya).
- The author scraped the data from 100,000 listings, then cleaned them and removed existing duplicates. The cleaned data were then separated into .csv files corresponding with each car manufacturer.
- The Ford data set was selected for this analysis. This data set has 17,965 records and 9 variables. During the analysis, some duplicates were detected and removed, remaining 17,811 records.

Variable name	Type	Description
model	string	Model of a car
year	integer	Manufacture year
price	integer	Selling price
transmission	string	Transmission type
mileage	integer	Mileage of a car
fuelType	integer	Fuel type
tax	integer	Current tax
mpg	float	Miles per gallon (or equivalent number for electric cars)
engineSize	float	Size of a car engine

Data exploration

- After removing duplicates, the EDA is conducted on the training set.
- The data description is as follows.

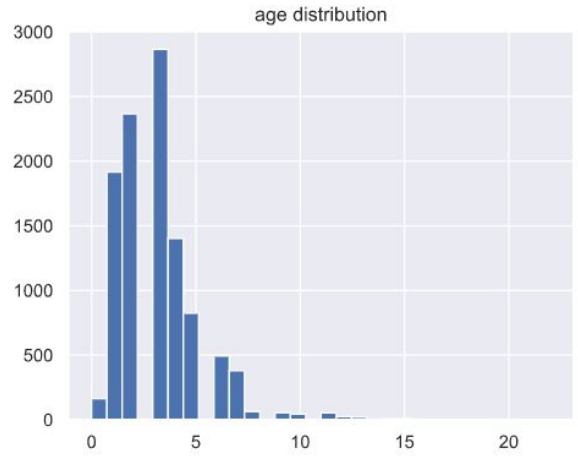
```
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   model         10686 non-null  object
 1   year          10686 non-null  int64
 2   price         10686 non-null  int64
 3   transmission  10686 non-null  object
 4   mileage       10686 non-null  int64
 5   fuelType      10686 non-null  object
 6   tax           10686 non-null  int64
 7   mpg           10686 non-null  float64
 8   engineSize    10686 non-null  float64
```

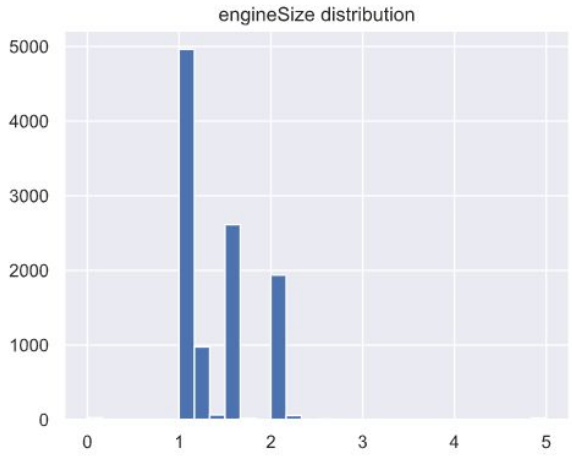
Data exploration

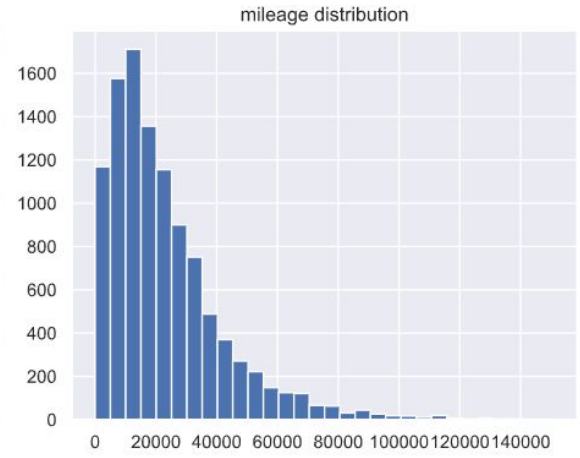
- Select cars that manufactured before 2021 (year <= 2020)
- Add in a new column, age, and remove year.

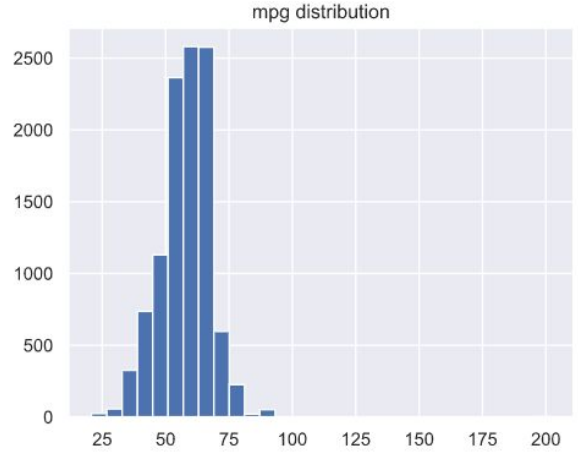
	year	price	mileage	tax	mpg	engineSize
count	10686.000000	10686.000000	10686.000000	10686.000000	10686.000000	10686.000000
mean	2016.874602	12267.263148	23275.151039	113.256130	58.003294	1.348615
std	2.038652	4728.282340	19308.759976	62.171836	10.153216	0.426191
min	1998.000000	675.000000	1.000000	0.000000	20.800000	0.000000
25%	2016.000000	8998.000000	9938.500000	30.000000	52.300000	1.000000
50%	2017.000000	11283.500000	18238.000000	145.000000	58.900000	1.200000
75%	2018.000000	15295.000000	31000.000000	145.000000	65.700000	1.500000
max	2060.000000	54995.000000	151000.000000	580.000000	201.800000	5.000000

	model	transmission	fuelType
count	10685	10685	10685
unique	21	3	5
top	Fiesta	Manual	Petrol
freq	3960	9224	7256

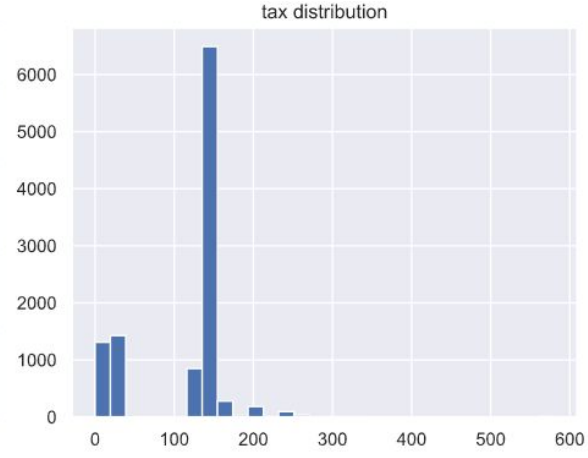

age distribution
3000
2500
2000
1500
1000
500
0
0
5
10
15
20


engineSize distribution
5000
4000
3000
2000
1000
0
0
1
2
3
4
5


mileage distribution
1600
1400
1200
1000
800
600
400
200
0
0
20000
40000
60000
80000
100000
120000
140000


mpg distribution
2500
2000
1500
1000
500
0
25
50
75
100
125
150
175
200


price distribution
2000
1500
1000
500
0
0
10000
20000
30000
40000
50000


tax distribution
6000
5000
4000
3000
2000
1000
0
0
100
200
300
400
500
600

Data exploration

- Except for tax and mpg, all features are right-skewed, and also there are zero values in engineSize (electric cars).
- Square root transformation might be a good choice to eliminate the skewness in this case.
- The target is kept the unchanged.

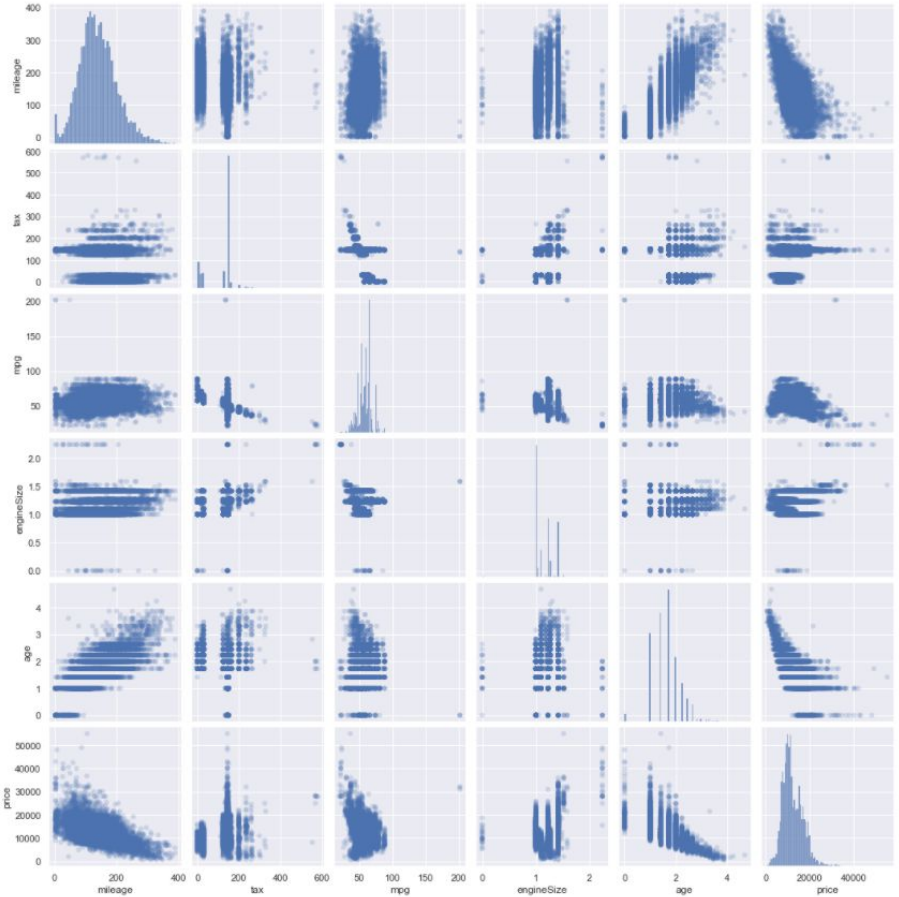
Data exploration

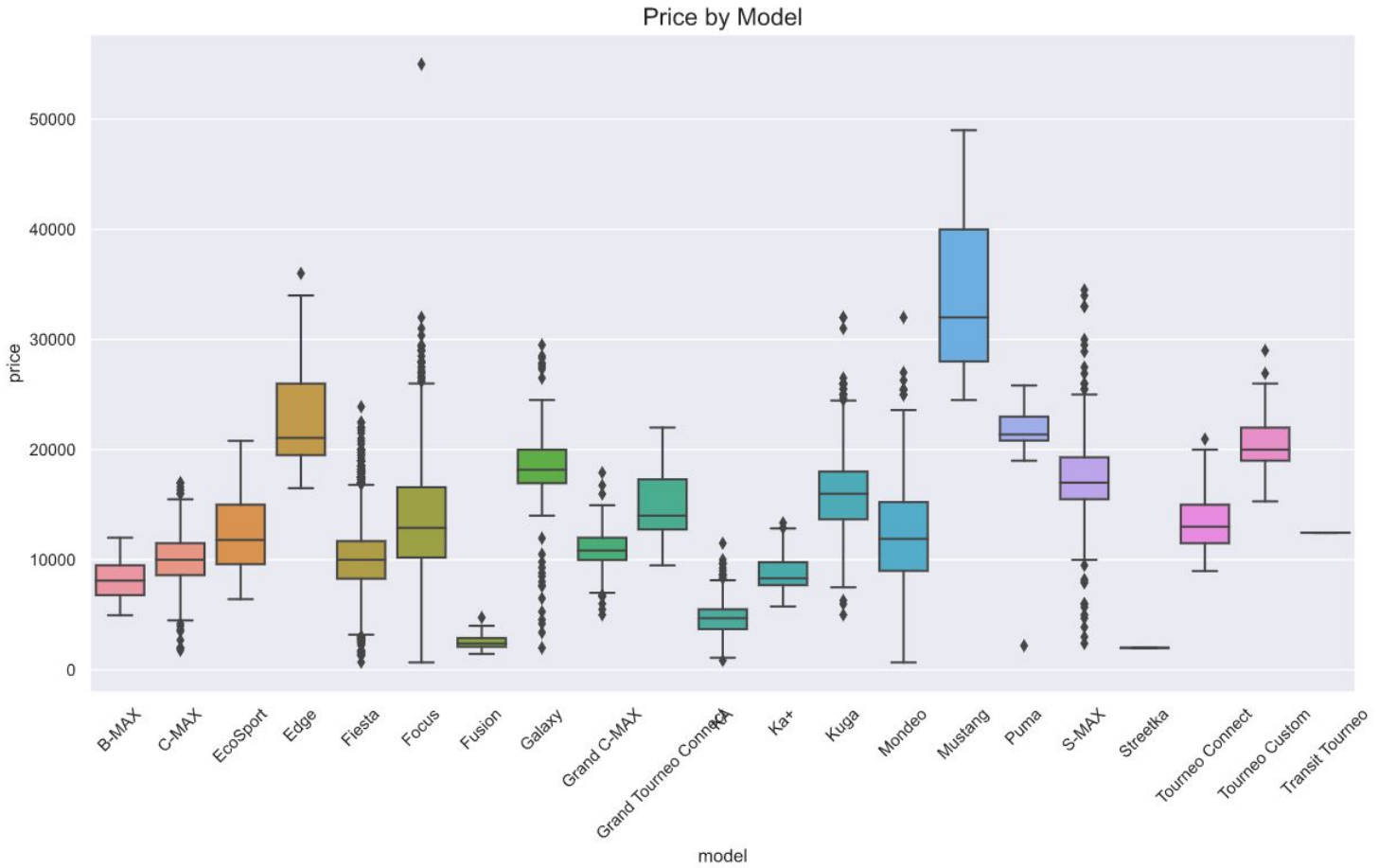
After taking square roots of skewed features, check all numerical variables again. This pair plot shows that:

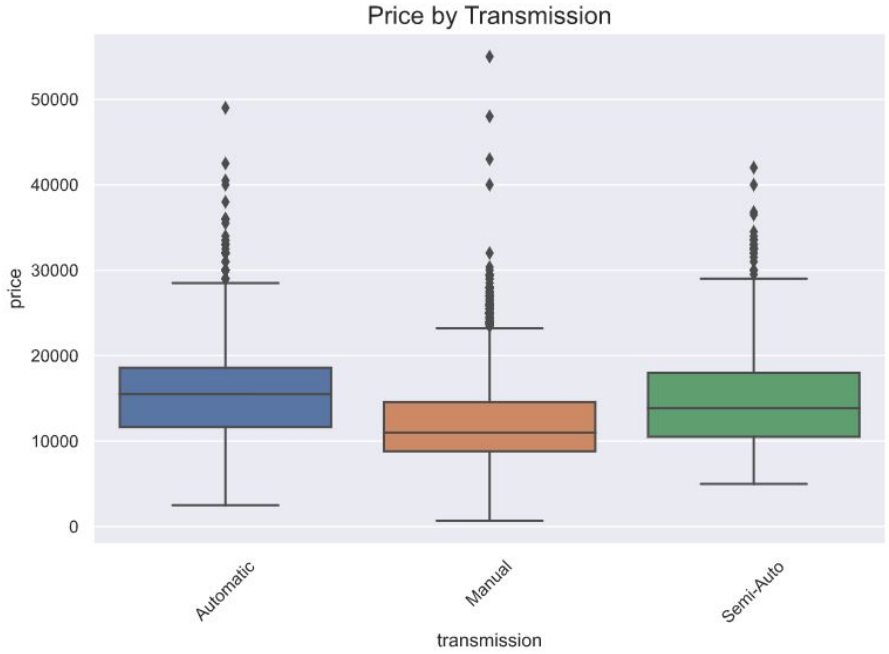
- *age* has a linear relationship with *price*. It looks quite like polynomial.
- *age* also has a linear relationship with *mileage* (the older the more miles). This is multicollinearity.

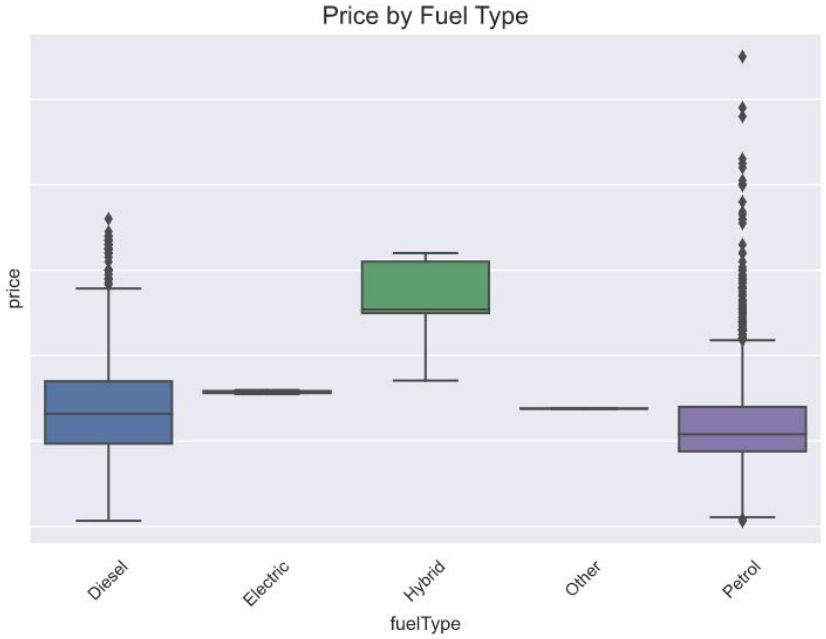
Multicollinearity might be resolved by regularization later.

Next pages show box plots of categorical features.




Price by Model
price
50000
40000
30000
20000
10000
0
B-MAX
C-MAX
EcoSport
Edge
Fiesta
Focus
Fusion
Galaxy
Grand C-MAX
Grand Tourneo Connect
KA
Ka+
Kuga
Mondeo
Mustang
Puma
S-MAX
Streetka
Tourneo Connect
Tourneo Custom
Transit Tourneo
model





On average, car prices vary among models, transmission, and fuel types.

Feature engineering and model variations

Feature engineering is applied in order to create model variations. Each model is evaluated based on its root mean square error.

- Apply one-hot encoding to categorical features: *model*, *transmission*, and *fuelType*
- Apply square root transformation to features that have a skew value greater than 0.75 (*engineSize*, *mileage*, *age*)
- Scale numerical features
- Add polynomial features

All these engineering steps are performed on training and validation sets, using K-fold cross-validation with k=5.

Feature engineering and model variations

- The model that has encoded features performs better, which is understandable because it has more information to predict the target. RMSEs of validation sets are slightly higher than training sets, which is expected. There is no sign of overfitting.
- The transformation improves all models. The one that has encoded features is the best so far.
- Scaling features is a preparation for regularization later. RMSEs of both training set and validation set should stay the same.

	Model	RMSE train	RMSE test
0	not encoded	2418.464757	2497.794531
1	one hot encoded	1826.674079	1888.968386
2	not encoded + squareroot	2386.249946	2459.074519
3	one hot encoded + squareroot	1717.389147	1740.250662
4	one hot encoded + squareroot + scaled	1717.389147	1740.250662

Feature engineering and model variations

- Add polynomial features to the latest model (encoded, square root transformed, and scaled) and fit the model again.
- It looks like the third polynomial degree transformation returns the best model. At degree 4 and above, as the model gets more and more complex, it starts overfitting.

	Model	Number of features	RMSE train	RMSE test
0	Degree = 1	30	1717.389147	1.740251e+03
1	Degree = 2	45	1605.687007	1.601615e+03
2	Degree = 3	80	1523.975464	1.532800e+03
3	Degree = 4	150	1409.705577	2.016472e+03
4	Degree = 5	276	1345.656334	9.623810e+03
5	Degree = 6	486	1224.846494	1.839370e+05
6	Degree = 7	816	1166.142822	1.107770e+07
7	Degree = 8	1311	1075.154458	2.514588e+08
8	Degree = 9	2026	1018.556904	8.078347e+10
9	Degree = 10	3027	990.406644	1.393925e+12

Cross-validation and Regularization

- Use the same data pipeline: one-hot encoding, square root transformation, standard scaling, and polynomial features adding.
- Use cross-validation to fit the linear regression model again, and then attempt to tune the hyperparameter to find a proper combination of alpha and polynomial degree for regularization. Regularized models include Lasso, Ridge, and Elastic Net.
- Each model is evaluated based on its average root mean squared error (from 5 folds).

Cross-validation and Regularization

- Iterate over different polynomial degree (1, 2, 3) and alphas.
- Result tables are sorted by RMSE in ascending order.

Linear

	Average RMSE
Degree = 3	1553.308686
Degree = 4	1609.889661
Degree = 2	1614.204357
Degree = 1	1727.132111
Degree = 5	4570.797917
Degree = 6	27767.413533

Lasso

	Average RMSE
Degree = 3, alpha = 0.05	1553.293103
Degree = 3, alpha = 0.01	1553.299960
Degree = 3, alpha = 0.005	1553.303938
Degree = 3, alpha = 0.1	1553.325592
Degree = 3, alpha = 0.3	1553.597472

Ridge

	Average RMSE
Degree = 3, alpha = 0.005	1553.311059
Degree = 3, alpha = 0.01	1553.313498
Degree = 3, alpha = 0.05	1553.335332
Degree = 3, alpha = 0.1	1553.368107
Degree = 3, alpha = 0.3	1553.551703

Elastic Net

	Average RMSE
Degree = 3, alpha = 0.005	1614.754948
Degree = 3, alpha = 0.01	1654.121040
Degree = 2, alpha = 0.005	1664.682197
Degree = 2, alpha = 0.01	1701.521926
Degree = 1, alpha = 0.005	1805.164786

Cross-validation and Regularization

- The metrics among Lasso, Ridge, and Linear regression are not significantly different.
- The best model is Lasso Regression with polynomial degree = 3 and alpha = 0.05.
- Elastic Net has the highest RMSE.

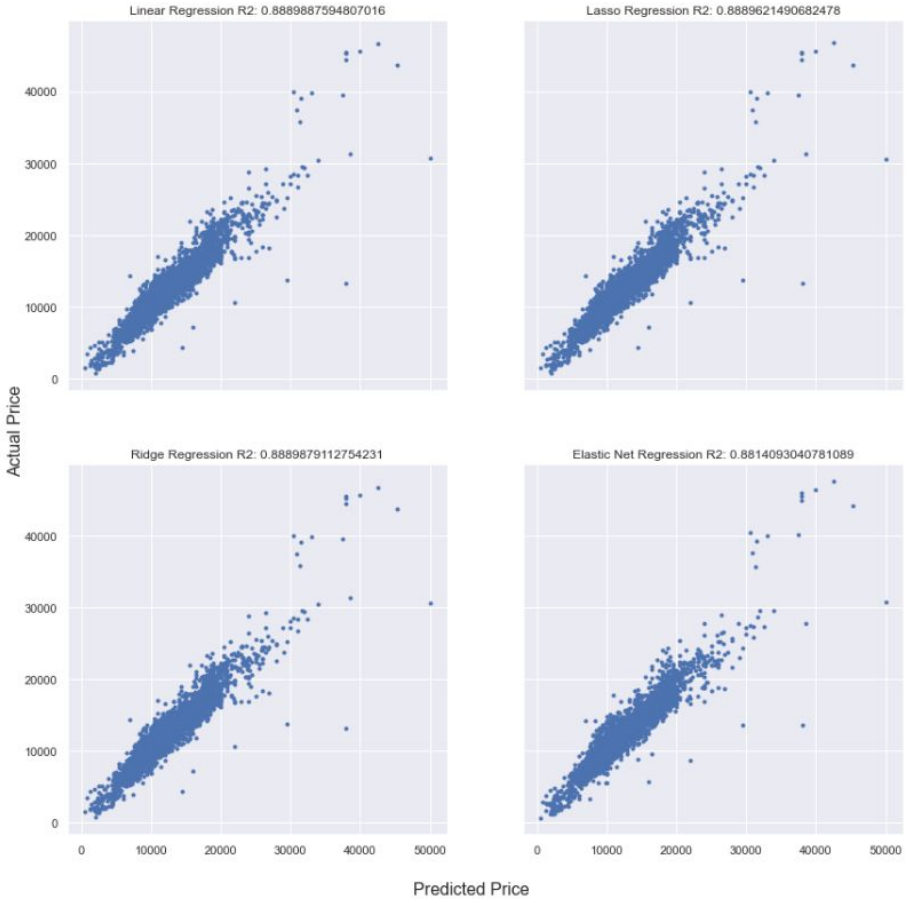
Model	Average RMSE	Average R2
Lasso	1553.293103	0.891003
Linear	1553.308686	0.891002
Ridge	1553.311059	0.891001
Elastic Net	1614.754948	0.882201

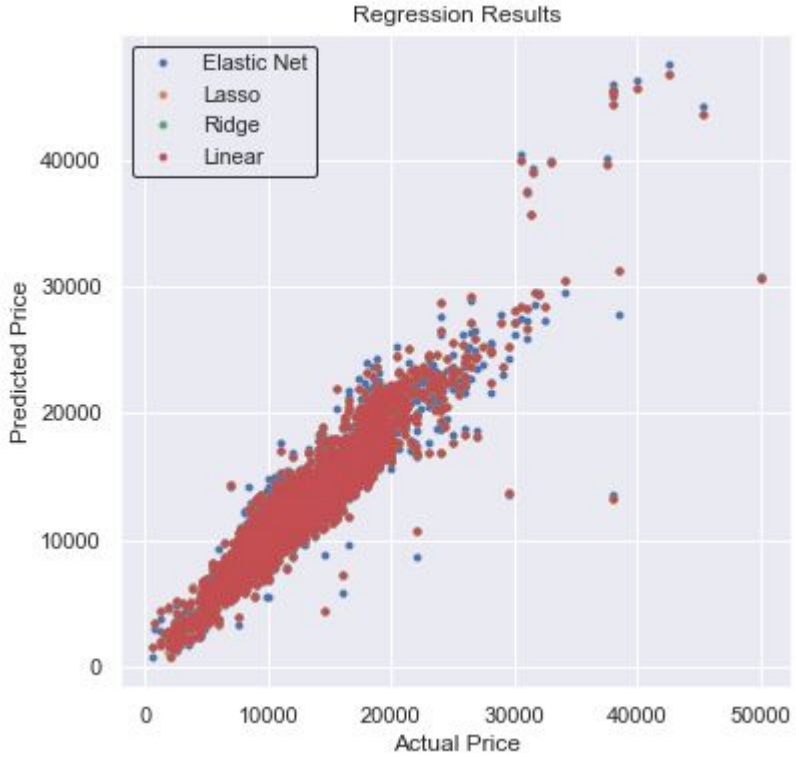
Predict on the test set

Fit four models to the unseen test set and calculate the R2 score for each model.

- Linear regression with third degree polynomial features
- Lasso regression with third degree polynomial features and alpha = 0.05
- Ridge regression with third degree polynomial features and alpha = 0.005
- Elastic Net regression with third degree polynomial features and alpha = 0.05

Next page shows scatter plots (true vs predicted price) and R2 scores.

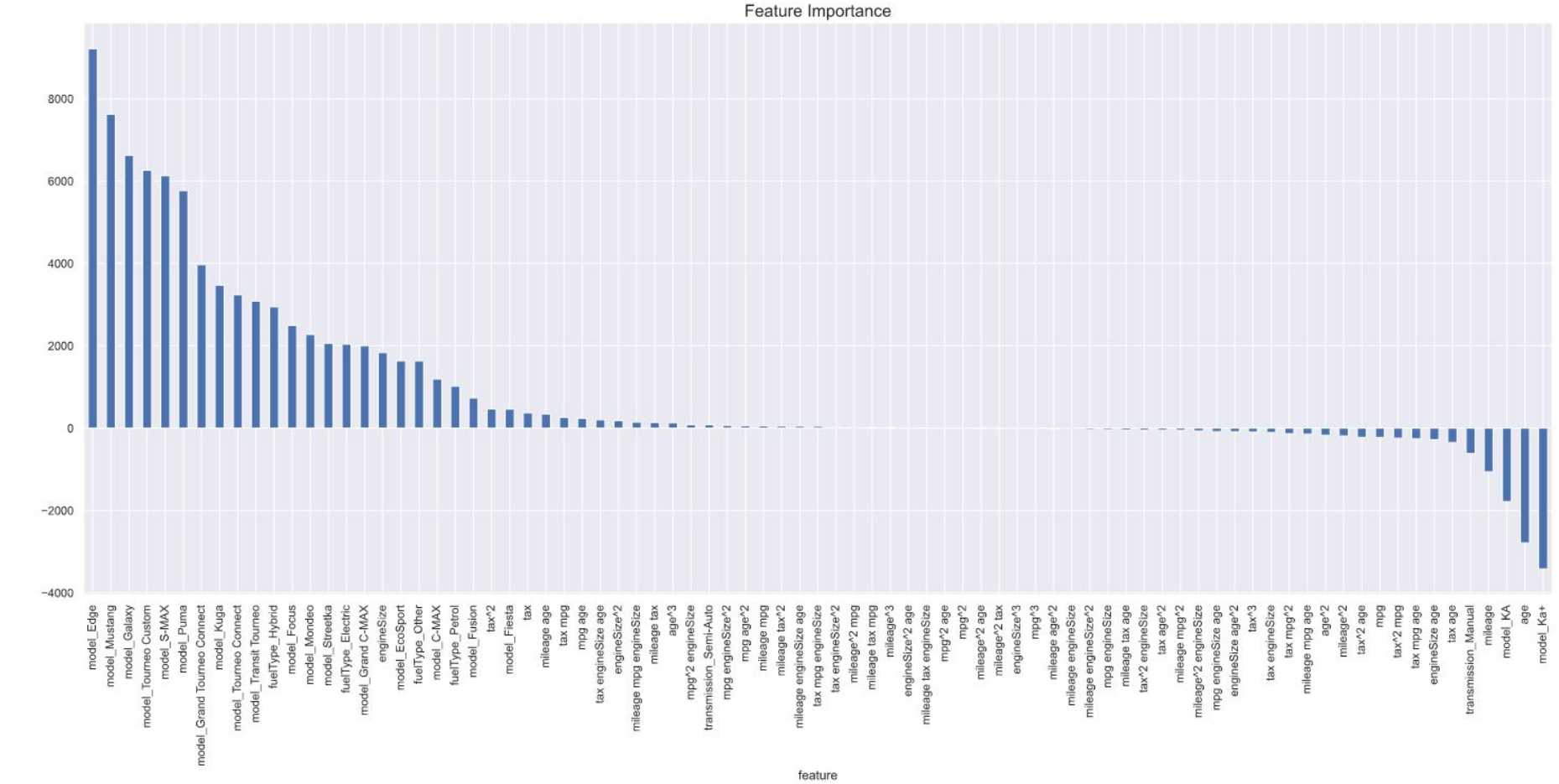

Linear Regression R2: 0.8889887594807016
Lasso Regression R2: 0.8889621490682478
Ridge Regression R2: 0.8889879112754231
Elastic Net Regression R2: 0.8814093040781089
Actual Price
Predicted Price


Regression Results
Elastic Net
Lasso
Ridge
Linear
Predicted Price
Actual Price

Predict on the test set

These plots show that all four models perform pretty well with no sign of overfitting. Linear Regression is the best model (R2=0.888988). R2 scores among Lasso, Ridge, and Linear regression are not significantly different.

Next page shows a plot of feature importance of the Linear Regression. The main drivers of this model are features that indicate whether or not the car model is Edge, Mustang, Galaxy, Tourneo Custom, S-MAX, Puma, or Grand Tourneo Connect. These are all derived from the categorical feature - model. Among numerical features, age and mileage have the strongest predictive power. Most interaction terms and polynomial features have low estimates in comparison to others.


Feature Importance
8000
6000
4000
2000
0
−2000
−4000
model_Edge
model_Mustang
model_Galaxy
model_Tourneo Custom
model_S-MAX
model_Puma
model_Grand Tourneo Connect
model_Kuga
model_Tourneo Connect
model_Transit Tourneo
fuelType_Hybrid
model_Focus
model_Mondeo
model_Streetka
fuelType_Electric
model_Grand C-MAX
engineSize
model_EcoSport
fuelType_Other
model_C-MAX
fuelType_Petrol
model_Fusion
tax^2
model_Fiesta
tax
mileage age
tax mpg
mpg age
tax engineSize age
engineSize^2
mileage mpg engineSize
mileage tax
age^3
mpg^2 engineSize
transmission_Semi-Auto
mpg engineSize^2
mpg age^2
mileage mpg
mileage tax^2
mileage engineSize age
tax mpg engineSize
tax engineSize^2
mileage^2 mpg
mileage tax mpg
mileage^3
engineSize^2 age
mileage tax engineSize
mpg^2 age
mpg^2
mileage^2 age
mileage^2 tax
engineSize^3
mpg^3
mileage age^2
mileage engineSize
mileage engineSize^2
mpg engineSize
mileage tax age
tax^2 engineSize
tax age^2
mileage mpg^2
mileage^2 engineSize
mpg engineSize age
engineSize age^2
tax^3
tax engineSize
tax mpg^2
mileage mpg age
age^2
mileage^2
tax^2 age
mpg
tax^2 mpg
tax mpg age
engineSize age
tax age
transmission_Manual
mileage
model_KA
age
model_Ka+
feature

Conclusion

This analysis shows that feature engineering can have a large effect on the model performance, and if the data are sufficiently large, cross-validation should be preferred over train-test-split to construct model evaluation. In my case, even though the predictors have high multicollinearity, their coefficients were not shrunk by regularized models, and it is shown that regularization does not always produce a better model. In the end, the Linear regression has the highest R2 when predicting on the test set, and categories of car model appear to be the most important features to predict a car price.

While researching further analysis, I found a suggestion of using grouped Lasso when a model have categorical features, which is worth trying in this case.

Jupyter Notebook can be found here:
https://github.com/thuynh323/IBM-Machine-Learning/blob/master/2-Supervised-Learning-Regression/Project-2.ipynb