CAR PRICE PREDICTION

Data Mining

Ph.D Le Thi Nhan

Nguyen Than Toan – 20C14008
thanhtoan12th@gmail.com

Contents

Problem statement

Nowadays, cars become one of the most popular verhicles. With a car, the people can move faster and safer. Beside of these helpful, the people who want to own a car also worry about the pricing and the quality of the car. So, develop a tool for predict the price of car is realy necessary. It help the people can estimate the pricing of a car that suitable base on their demands.

In this project, I will try to build a model can predict the price of a car from attributes of car such as: symboling, fuel type, num of doors, .etc. from the dataset I collected online from link: https://gist.github.com/jalaliamin/ff2fca9c2a808deae53270c186c2d39e. Input: the car's attributes such as: engine-size, horsepower, etc and the output is the price of the car.

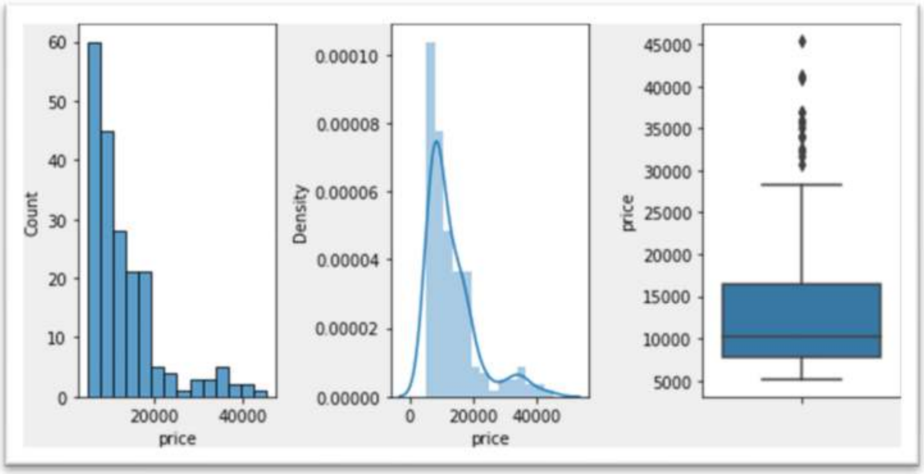
Data description

This dataset has 205 rows and 26 columns. Overview of dataset was show in table below:

No.	Column Name	Non Null Count	Value Type	Data Type
1	symboling	205	int64	Nominal
2	normalized-losses	164	float64	Numeric
3	make	205	string	Nominal
4	fuel-type	205	string	Nominal
5	aspiration	205	string	Nominal
6	num-of-doors	203	string	Ordinal
7	body-style	205	string	Nominal
8	drive-wheels	205	string	Nominal
9	engine-location	205	string	Nominal
10	wheel-base	205	float64	Numeric
11	length	205	float64	Numeric
12	width	205	float64	Numeric
13	height	205	float64	Numeric
14	curb-weight	205	int64	Numeric
15	engine-type	205	object	Nominal
16	num-of-cylinders	205	object	Ordinal
17	engine-size	205	int64	Numeric
18	fuel-system	205	string	Nominal
19	bore	201	float64	Numeric
20	stroke	201	float64	Numeric
21	compression-ratio	205	float64	Numeric
22	horsepower	203	int64	Numeric
23	peak-rpm	203	int64	Numeric
24	city-mpg	205	int64	Numeric
25	highway-mpg	205	int64	Numeric
26	price	201	int64	Numeric

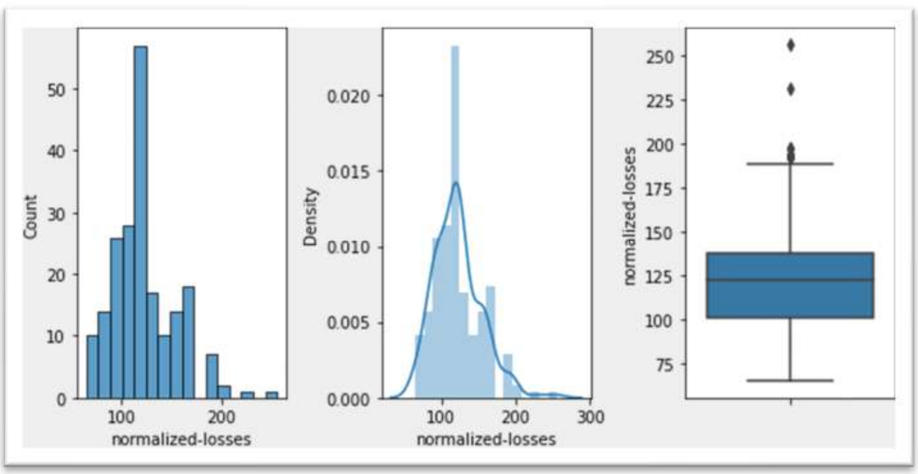
Data understanding

1. Data type numeric



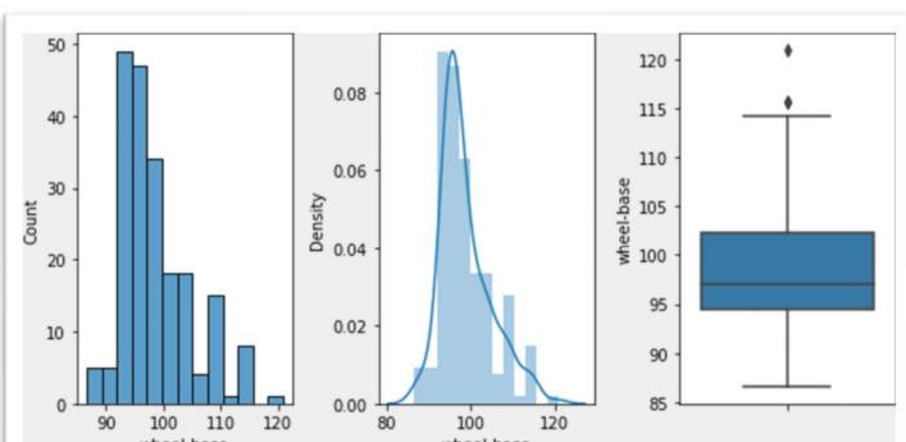
Skew: 1.81
count 201.000000
mean 13207.129353
std 7947.066342
min 5118.000000
25% 7775.000000
50% 10295.000000
75% 16500.000000
max 45400.000000
Upper outlier: 14 , Lower outlier: 0
Mean before drop outlier: 13207.13
Mean drop outlier: 11503.18
Skew after drop outlier: 1.02

Figure 1: Distribution of column price



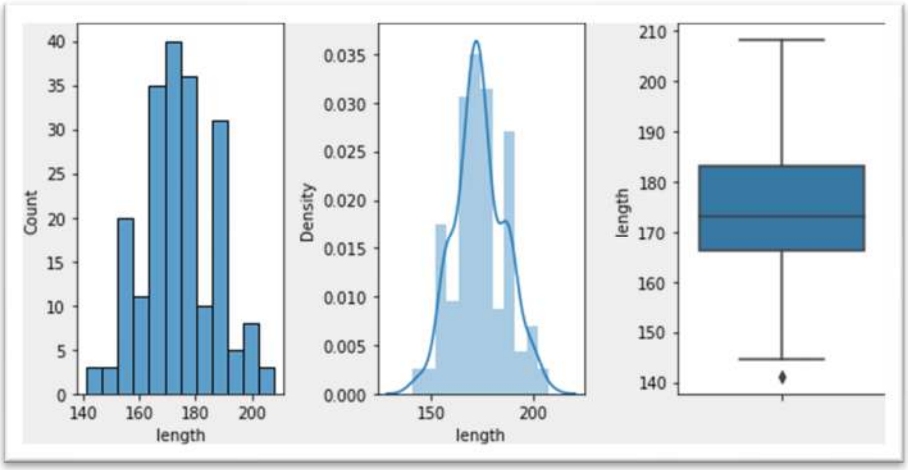
Skew: 0.85
count 205.000000
mean 122.000000
std 31.681008
min 65.000000
25% 101.000000
50% 122.000000
75% 137.000000
max 256.000000
Upper outlier: 8 , Lower outlier: 0
Mean before drop outlier: 122.00
Mean after drop outlier: 118.56
Skew after drop outlier: 0.26

Figure 2: Distribution of column normalized-losses



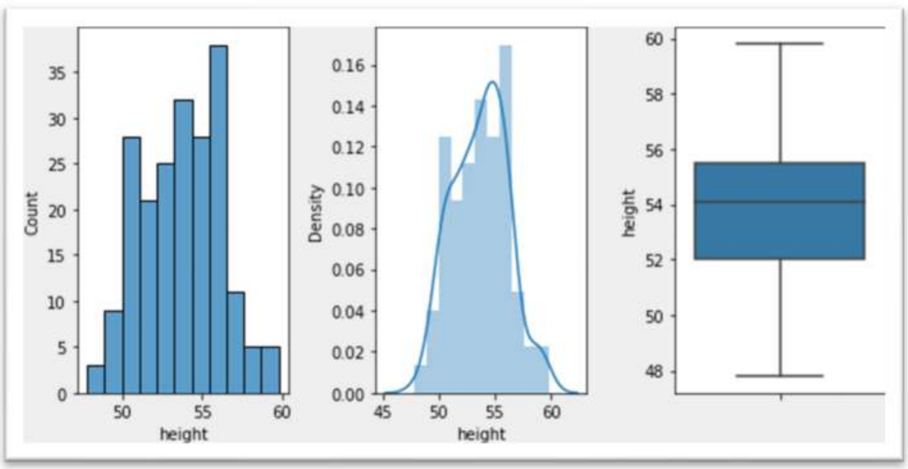
Skew: 1.05
count 205.000000
mean 98.756585
std 6.021776
min 86.600000
25% 94.500000
50% 97.000000
75% 102.400000
max 120.900000
Upper outlier: 3 , Lower outlier: 0
Mean before drop outlier: 98.76
Mean after drop outlier: 98.48
Skew after drop outlier: 0.89

Figure 3: Distribution of column wheel-base



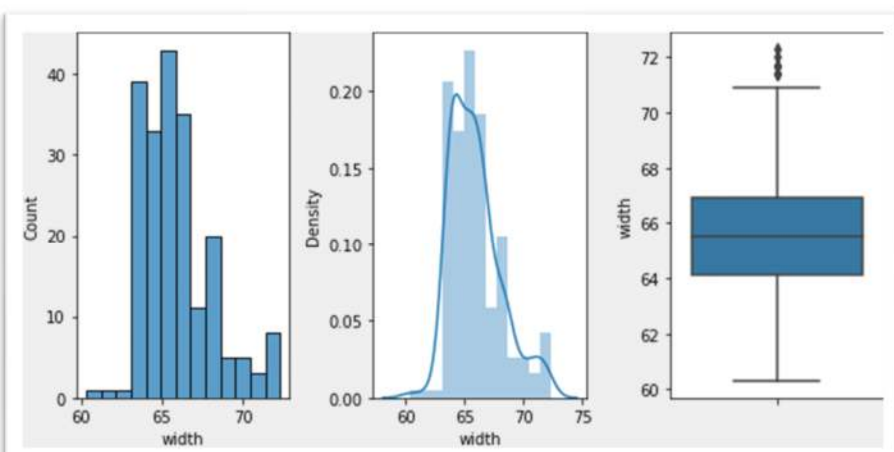
Skew: 0.16
count 205.000000
mean 174.049268
std 12.337289
min 141.100000
25% 166.300000
50% 173.200000
75% 183.100000
max 208.100000
Upper outlier: 0 , Lower outlier: 1
Mean before drop outlier: 174.05
Mean after drop outlier: 174.21
Skew after drop outlier: 0.22

Figure 4: Distribution of column length



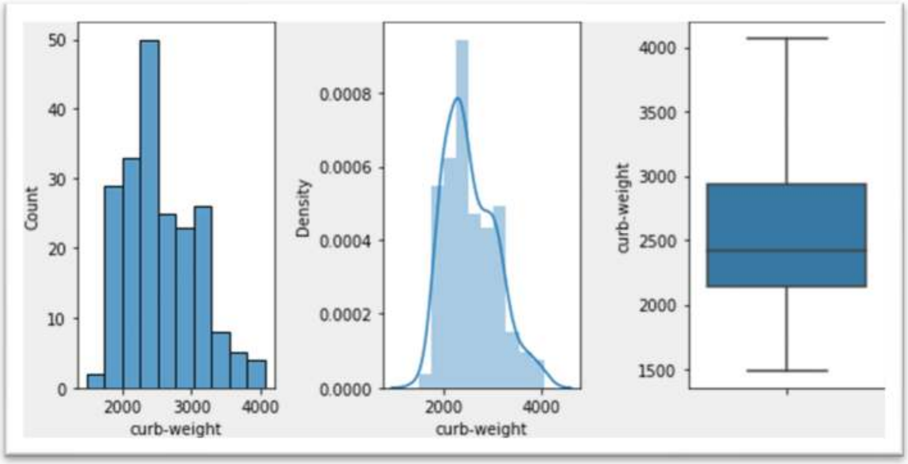
Skew: 0.06
count 205.000000
mean 53.724878
std 2.443522
min 47.800000
25% 52.000000
50% 54.100000
75% 55.500000
max 59.800000
Upper outlier: 0 , Lower outlier: 0

Figure 5: Distribution of column height



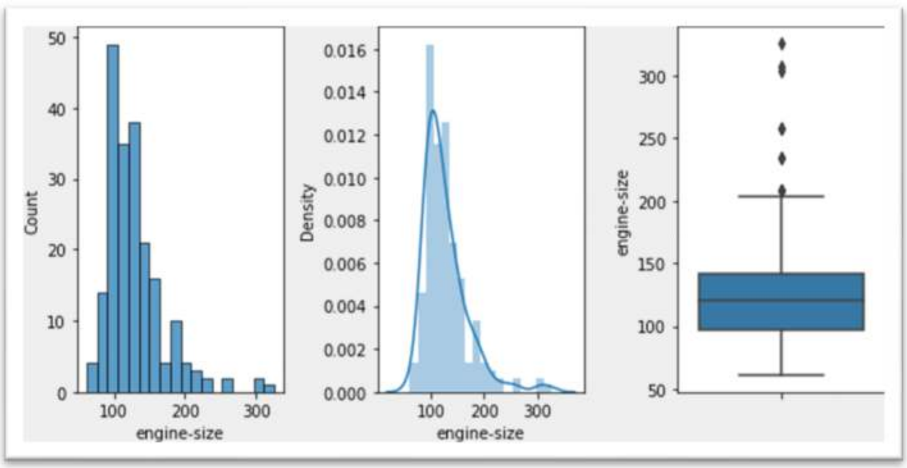
Skew: 0.90
count 205.000000
mean 65.907805
std 2.145204
min 60.300000
25% 64.100000
50% 65.500000
75% 66.900000
max 72.300000
Upper outlier: 8 , Lower outlier: 0
Mean before drop outlier: 65.91
Mean after drop outlier: 65.67
Skew after drop outlier: 0.59

Figure 6: Distribution of column width



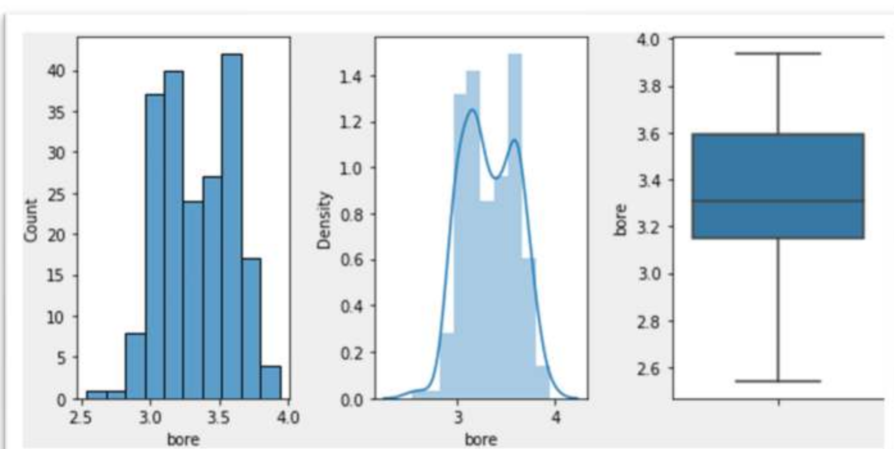
Skew: 0.68
count 205.000000
mean 2555.565854
std 520.680204
min 1488.000000
25% 2145.000000
50% 2414.000000
75% 2935.000000
max 4066.000000
Upper outlier: 0 , Lower outlier: 0

Figure 7: Distribution of column curb-weight



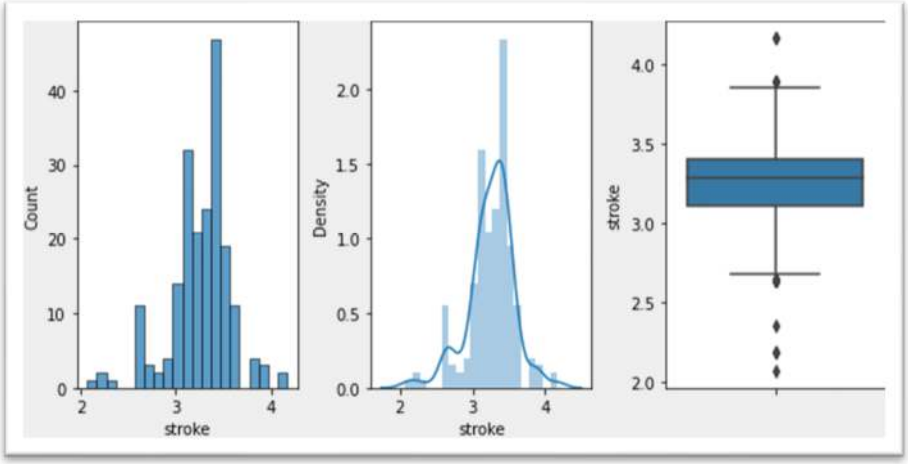
Skew: 1.95
count 205.000000
mean 126.907317
std 41.642693
min 61.000000
25% 97.000000
50% 120.000000
75% 141.000000
max 326.000000
Upper outlier: 10 , Lower outlier: 0
Mean before drop outlier: 126.91
Mean after drop outlier: 120.34
Skew after drop outlier: 0.78

Figure 8: Distribution of column engine-size



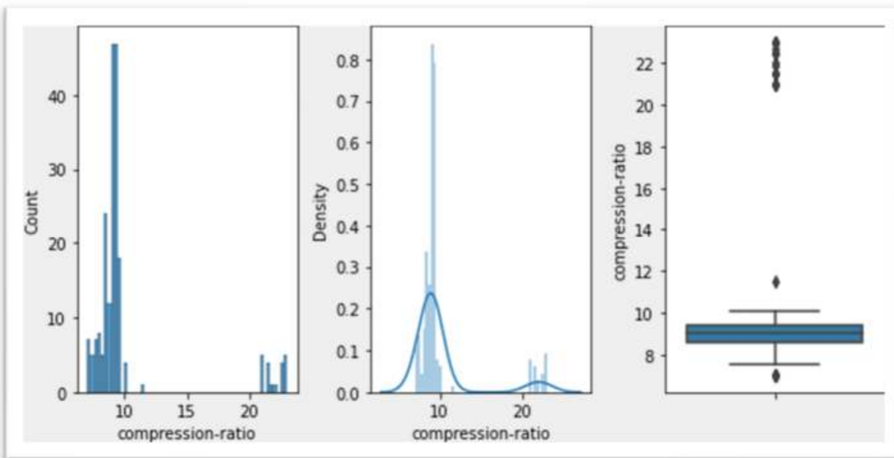
Skew: 0.02
count 201.000000
mean 3.329751
std 0.273539
min 2.540000
25% 3.150000
50% 3.310000
75% 3.590000
max 3.940000
Upper outlier: 0 , Lower outlier: 0

Figure 9: Distribution of column bore



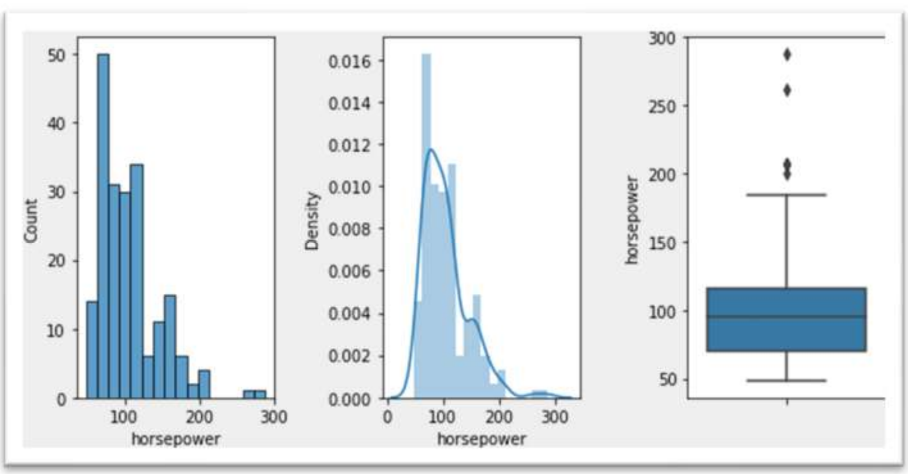
Skew: -0.68
count 201.000000
mean 3.255423
std 0.316717
min 2.070000
25% 3.110000
50% 3.290000
75% 3.410000
max 4.170000
Upper outlier: 5 , Lower outlier: 15
Mean before drop outlier: 3.26
Mean after drop outlier: 3.30
Skew after drop outlier: -0.15

Figure 10: Distribution of column stroke



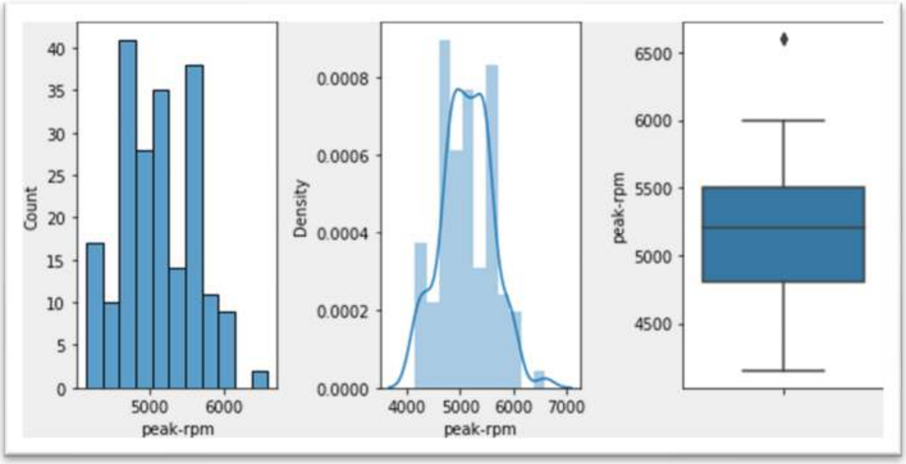
Skew: 2.61
count 205.000000
mean 10.142537
std 3.972040
min 7.000000
25% 8.600000
50% 9.000000
75% 9.400000
max 23.000000
Upper outlier: 21 , Lower outlier: 7
Mean before drop outlier: 10.14
Mean after drop outlier: 8.92
Skew after drop outlier: -0.80

Figure 12: Distribution of column compression ratio



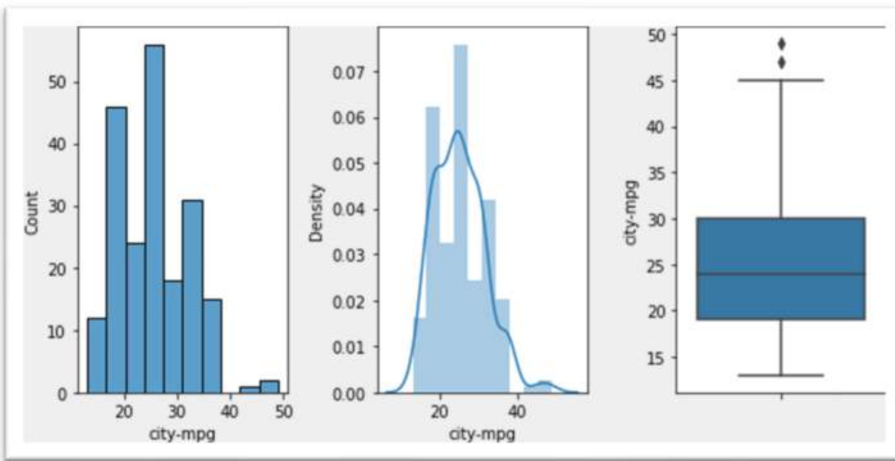
Skew: 1.40
count 205.000000
mean 104.256158
std 39.519211
min 48.000000
25% 70.000000
50% 95.000000
75% 116.000000
max 288.000000
Upper outlier: 6 , Lower outlier: 0
Mean before drop outlier: 104.26
Mean after drop outlier: 100.51
Skew after drop outlier: 0.79

Figure 11: Distribution of column horsepower



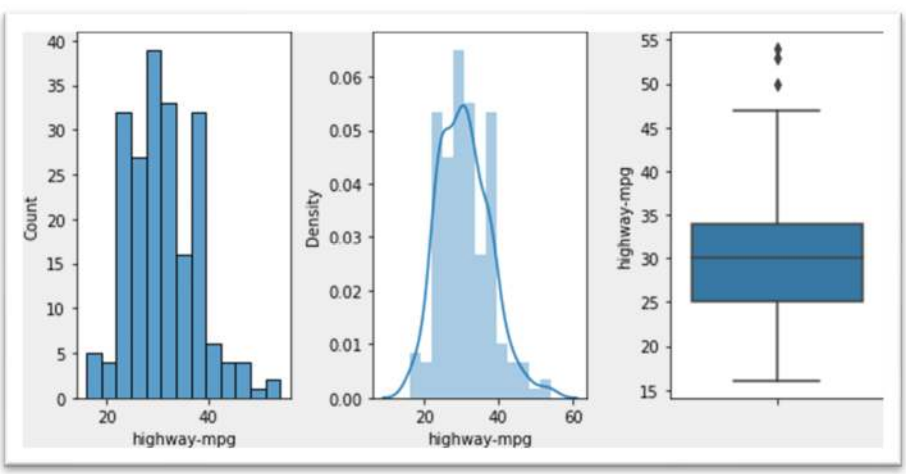
Skew: 0.07
count 205.000000
mean 5125.369458
std 476.979093
min 4150.000000
25% 4800.000000
50% 5200.000000
75% 5500.000000
max 6600.000000
Upper outlier: 2 , Lower outlier: 0
Mean before drop outlier: 5125.37
Mean after drop outlier: 5110.84
Skew after drop outlier: -0.16

Figure 13: : Distribution of column peak-rpm



Skew: 0.66
count 205.000000
mean 25.219512
std 6.542142
min 13.000000
25% 19.000000
50% 24.000000
75% 30.000000
max 49.000000
Upper outlier: 2 , Lower outlier: 0
Mean before drop outlier: 25.22
Mean after drop outlier: 25.00
Skew after drop outlier: 0.40

Figure 14: : Distribution of column city-mpg



Skew: 0.54
count 205.000000
mean 30.751220
std 6.886443
min 16.000000
25% 25.000000
50% 30.000000
75% 34.000000
max 54.000000
Upper outlier: 3 , Lower outlier: 0
Mean before drop outlier: 30.75
Mean after drop outlier: 30.43
Skew after drop outlier: 0.25

Figure 15: : Distribution of column highway-mpg

2. Data type ordinal and norminal

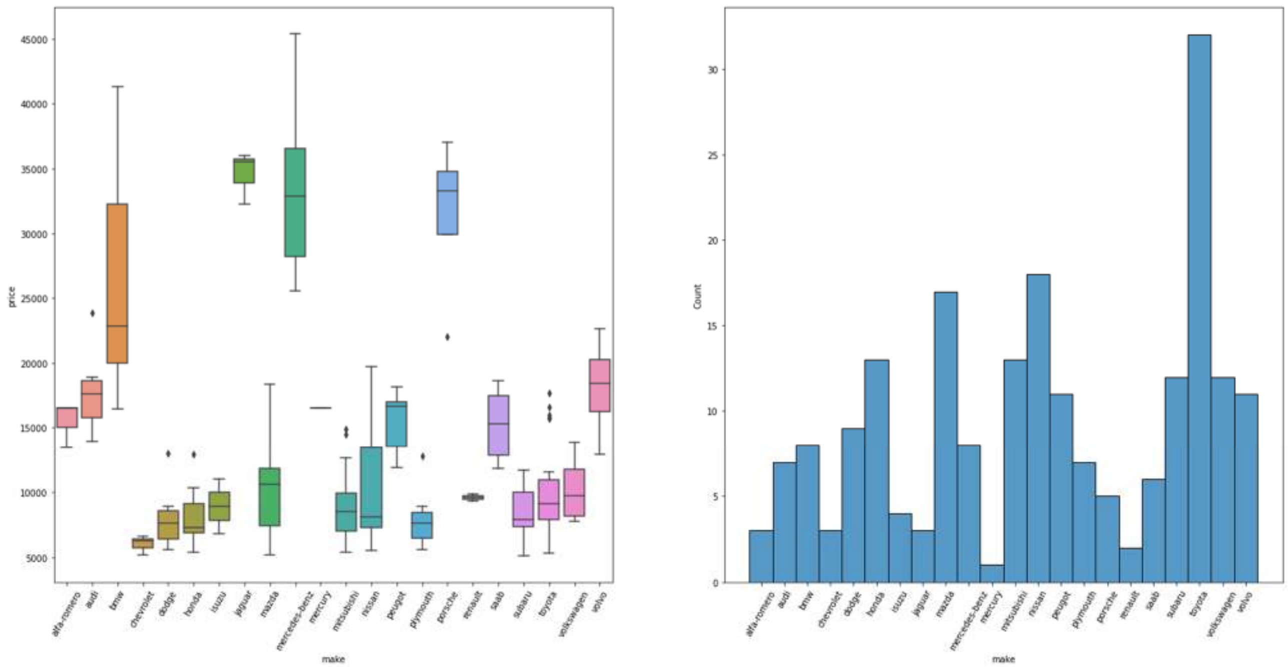


Figure 16: : Distribution of column make

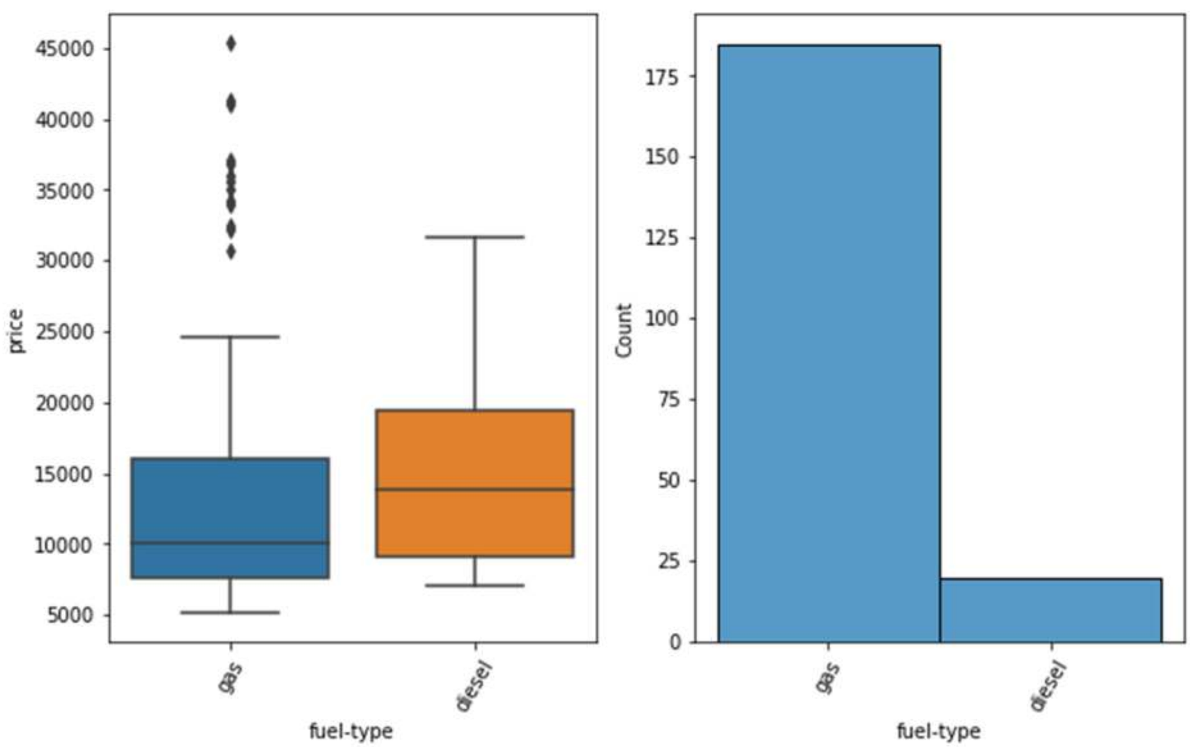


Figure 17: : Distribution of column fuel-type

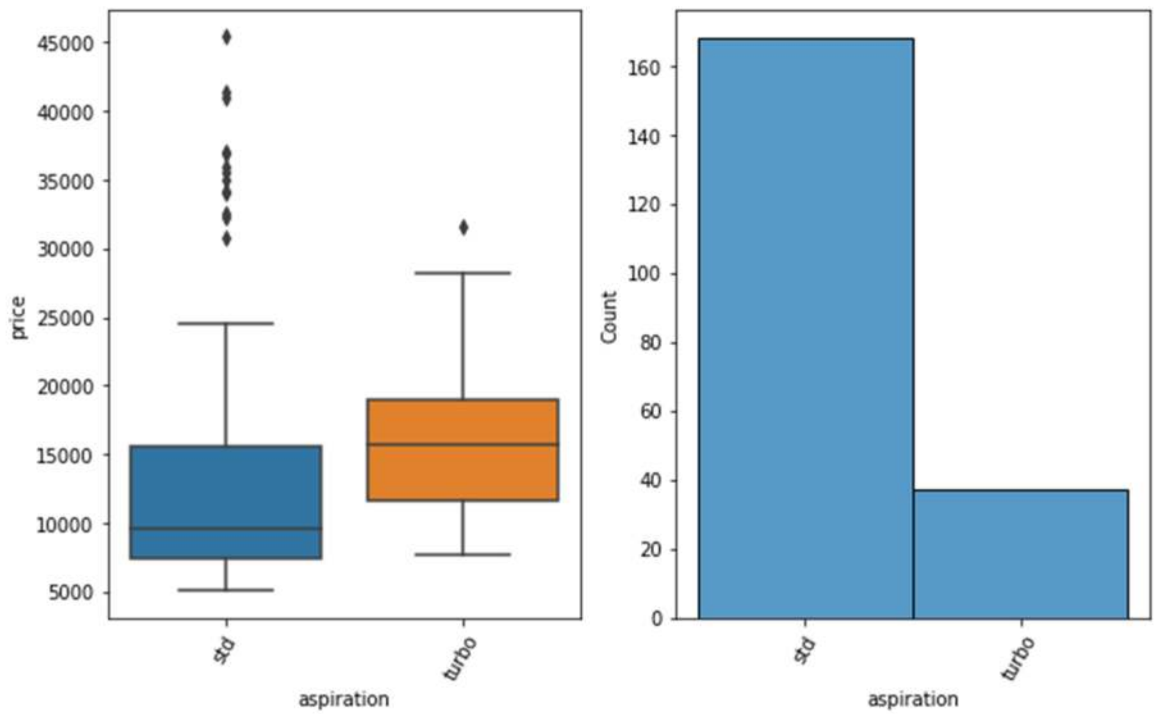


Figure 18: Distribution of column aspiration

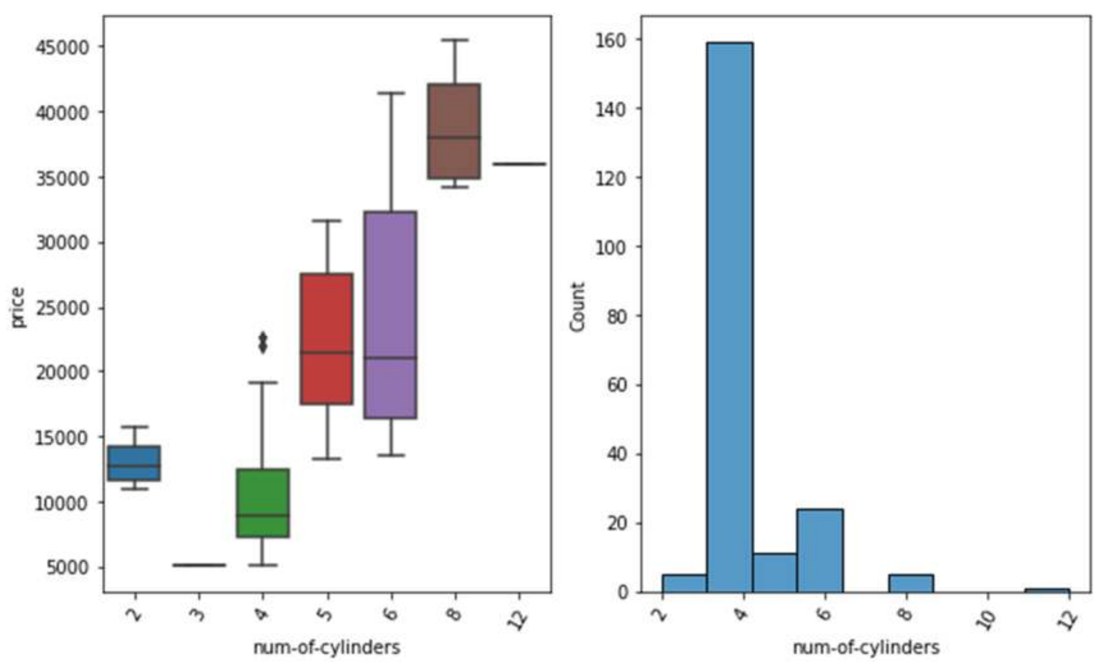


Figure 19: : Distribution of column num-of-cylinders

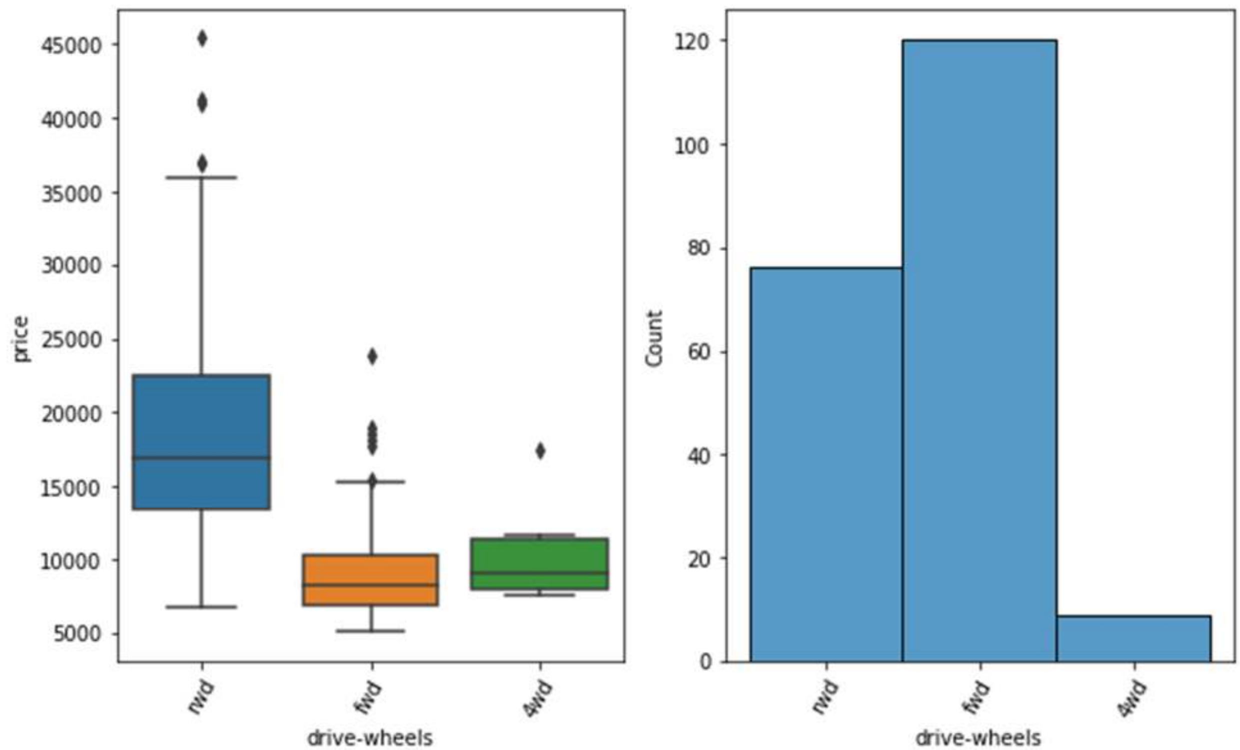


Figure 20: Distribution of column drive-wheels

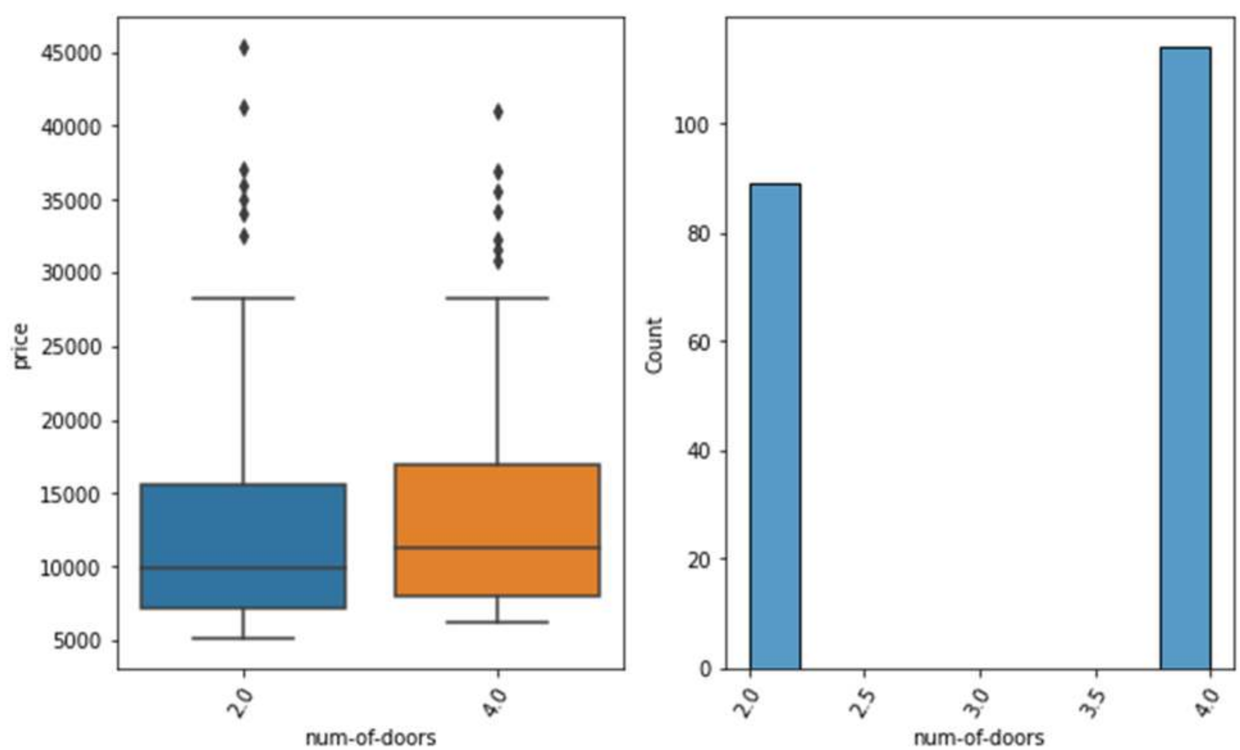


Figure 21: : Distribution of column num-of-doors

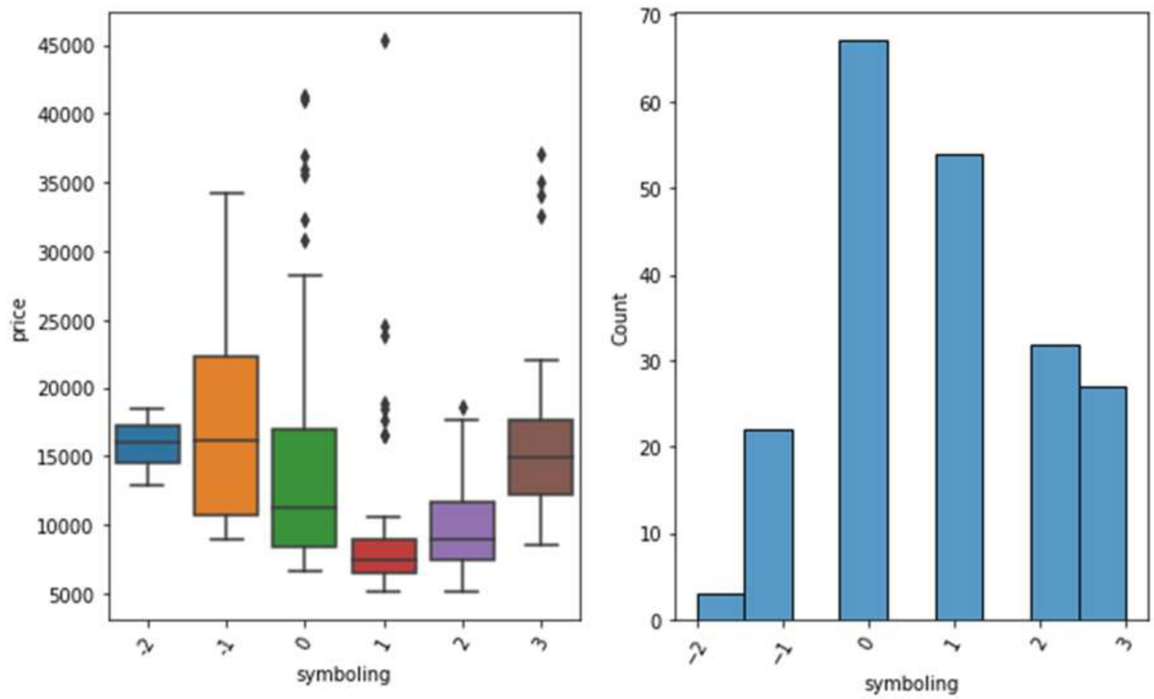


Figure 22: : Distribution of column symboling

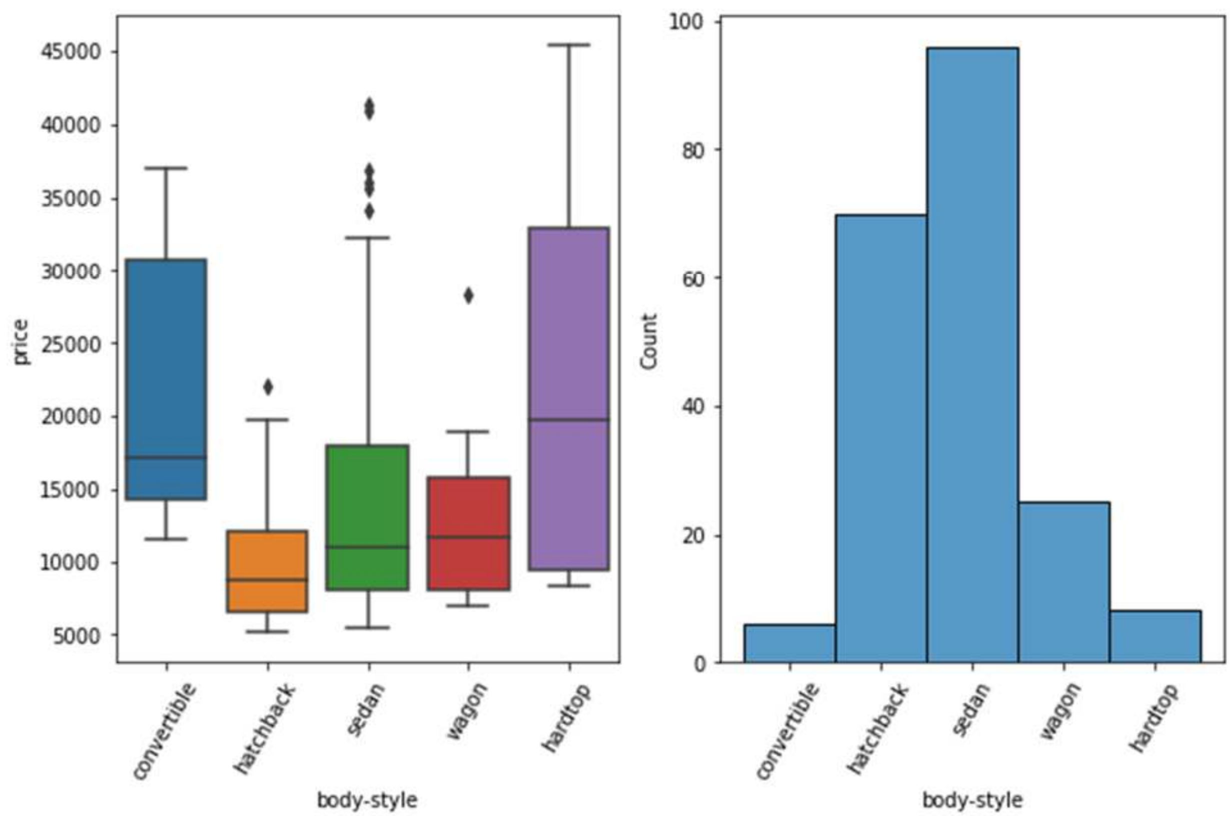


Figure 23: Distribution of column body-style

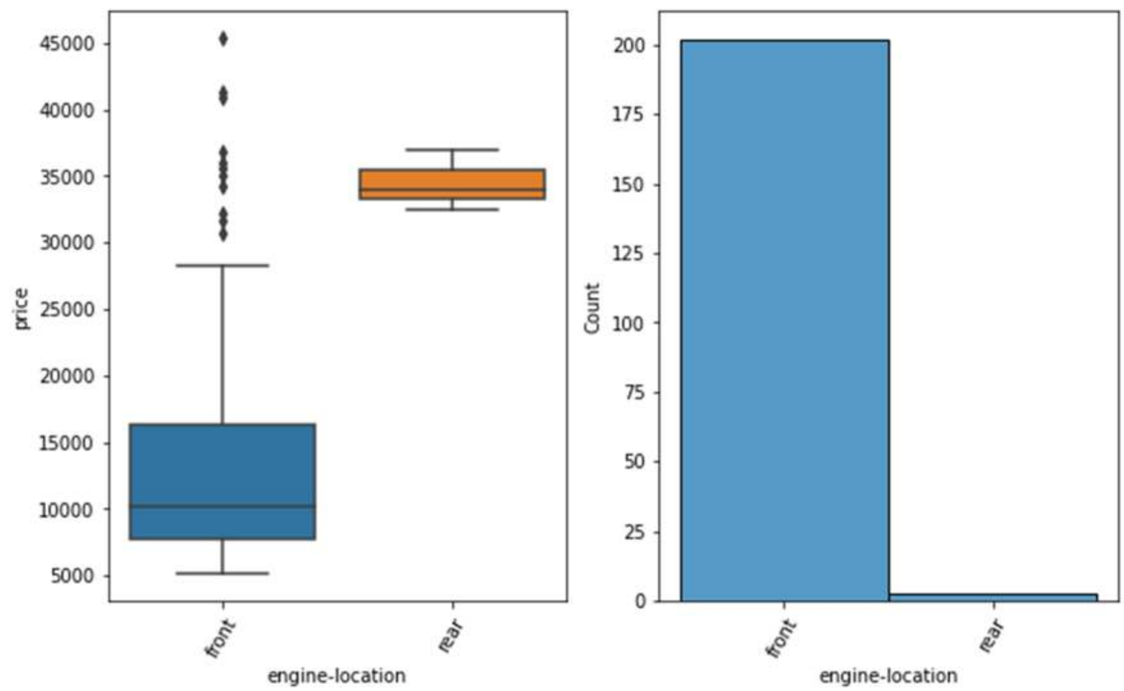


Figure 24: Distribution of column engine-location

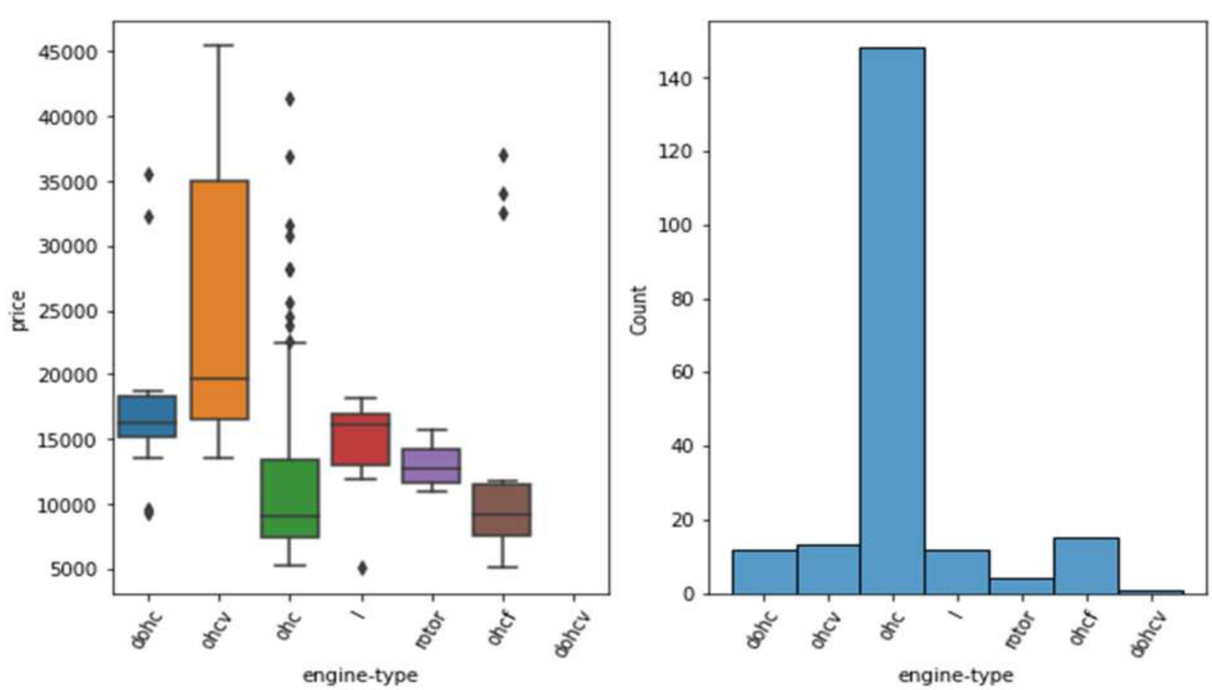


Figure 25: Distribution of column engine-type

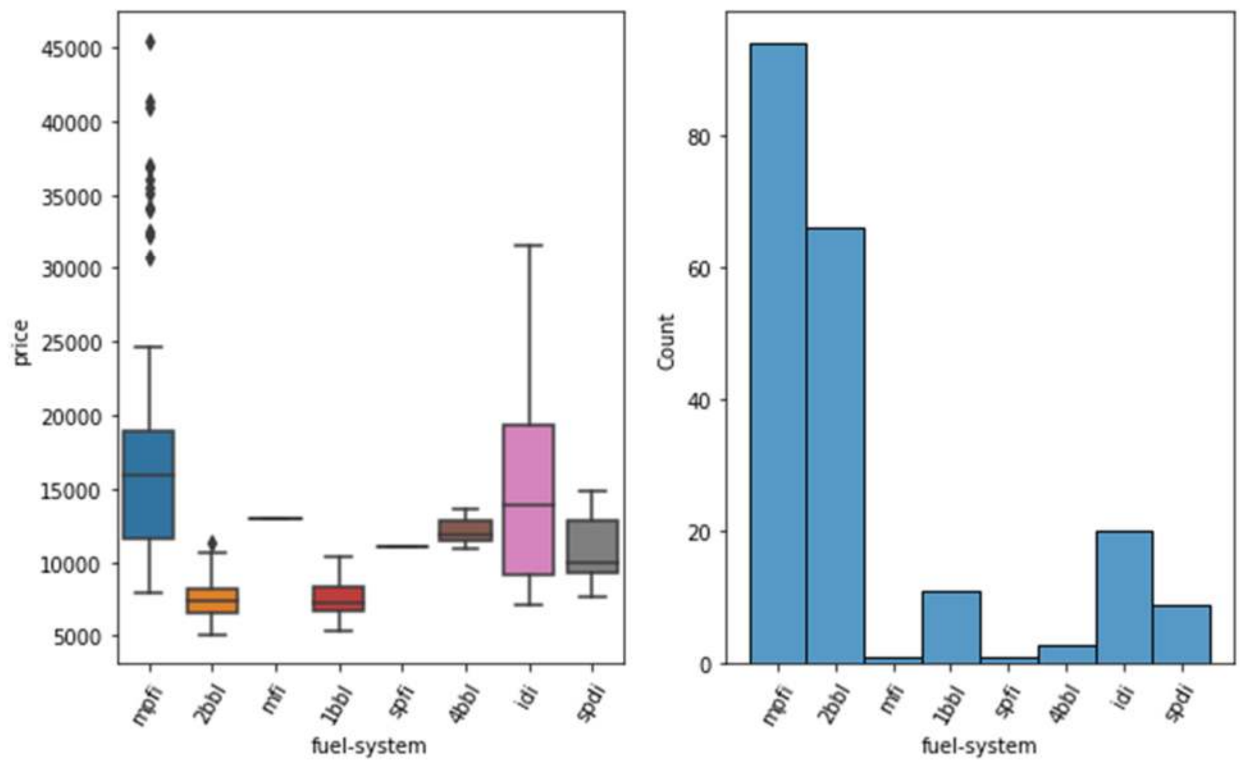


Figure 26: Distribution of column fuel-system

Data preprocessing

1. Data type numeric

- Fill all missing values with mean of each column: normalized-losses, horsepower, peak-rpm, bore, stroke.
- Use pearson to compute corelation of columns to select only the usefull attributes to use for predictions

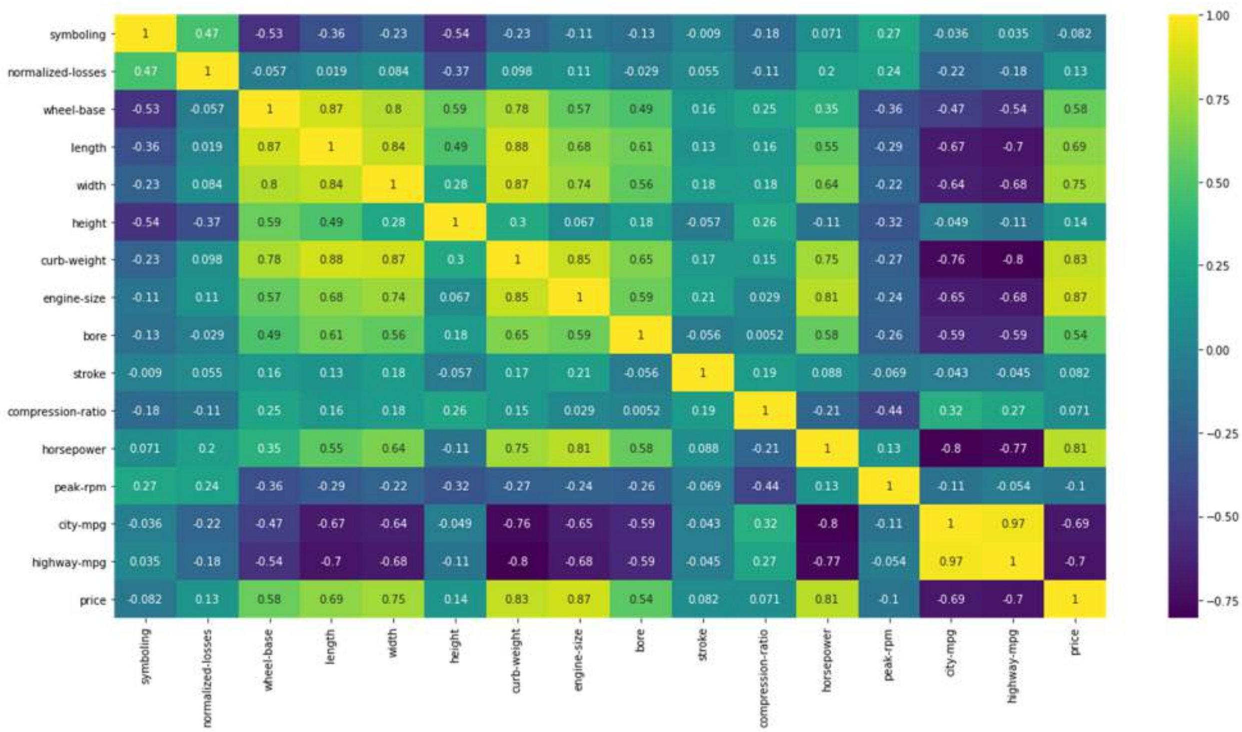


Figure 27: Heatmap about correlation of columns in current dataset

- Base on figure 27, horsepower, engine-size, curb-weight, bore, width, length, wheel-base, city-mpg, highway-mpg have strongly positive or negative relationship with price, so all of them are the usefull attributes for prediction.
- Scale value between width and length by using MinMaxScaler method. Data distribution of width and length after apply MinMaxScaler:

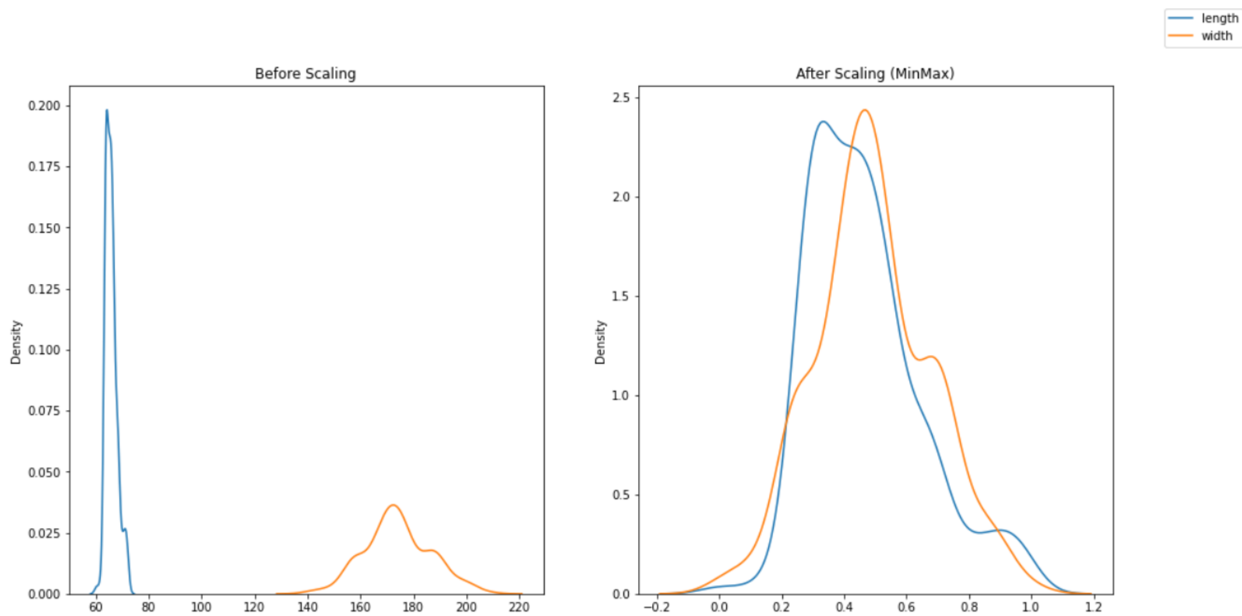


Figure 28: Data distribution of width and length

- The distribution of the “price” column is also imbalance, has high variance between Mean and IQ2, the number of none values is small, so removing rows that exists missing values at the “price” column is the best solution for reprocessing.

2. Data type ordinal an norminal

- The figures about the distribution of attributes such as engine-location, aspiration, fuel-type, engine-type show the imbalance of their value. These columns would be remove because these columns can make the accuracy of model become lower.
- With “drive-wheels” attribute, although there is an imbalance, by applying oversampling, this attribute can be device into two groups: “fwd” and “not-fwd”.
- In a same situation with “drive-wheels” attribute, by applying oversampling, the “fuel-system” attribute can be device into two groups: “mpfi” and “another-fuel-system”.
- Data distribution of body-style is imbalance. In this case can apply downsampling to make data become balance. The values of body-style attribute can be device into two group: sedan and another-style (include hatchback, wagon, hardtop, convertible).
- After applying downsampling for drive-wheels, fuel-system, body-style to decrease imbalance in distribute of these attributes, I use one-hot encoding technique to create additional features based on the number of unique values in the categorical feature. Every unique value in the category will be added as a feature.
- With “make” attribute, I think it is an importance attribute for this model. Because the reputation and quality of the manufacturer also dorminant affect the price of the product. Moreover, distribution of the values of this attribute is balanced relatively. So this attribute can be use for this model. One-hot encoding also a best technique to process data of this attribute.

- With “num-of-doors” and “num-of-cylinders”, these attributes are also ordinal data. To process these attributes, I convert all the string of number values to number and fill all missing value with mode of each attributes.

3. Outliers

Basic statistical about outliers in current dataset:

	Ratio of outliers(%)	Difference (Mean)	Difference (Mean %)	Difference (Median)	Up	Low
wheel-base	1.463415	0.276387	0.997201	0.1	114.25	82.65
bore	0	0	1	0	4.23	2.5
engine-size	4.878049	6.563727	0.948279	10	207	31
curb-weight	0	0	1	0	4120	960
horsepower	2.926829	3.746045	0.964069	0	185	1
length_new	0	0	1	0	1	0
width_new	3.902439	0.019601	0.958056	0.008333	0.9	-0.03
highway-mpg	1.492537	0.327981	0.989312	0	47.5	11.5
city-mpg	0.995025	0.229356	0.990891	0	46.5	2.5
price	6.829268	1668.268	0.873684	607	29568	-5280

The number of outliers in current dataset have small ratio and difference between mean values in both case have outliers and not have outliers also small. So, the outliers in this dataset can be accepted.

Solution and Evaluation

Applying multible model available on sklearn to predict price of car:

- LinearRegression: basic modle, one of the most important and widely used regression techniques.
- RandomForestRegressor:a regression base on multiple decision trees.
- DecisionTreeRegressor: a regression base on decision tree.
- KneighborsRegressor: a regression based on k-nearest neighbors.
- SVR: Linear support vector regression.

Try to train each models for 10 times, after train model, I calulate mean of each values: R^2, Mean Squared Error in both train data and test data to evaluation the accuracy of models above.

	Model	Score Train	Score Test	ABS Mean	MSE Train	MSE Test	Time
1	LinearRegression	0.94471	0.897396	0.047314	3547327	5910358	2.1
2	KNeighborsRegressor	0.859966	0.769522	0.090444	8984404	13276442	1.5
3	DecisionTreeRegressor	0.998456	0.872464	0.125992	99051.37	7346568	2.6
4	RandomForestRegressor	0.989133	0.920464	0.068669	697245	4581584	139.64
5	SVR	-0.10612	-0.08831	0.017812	70967070	62690579	2.4

RandomForestRegressor model show the best performance to predict price in this dataset but It take the most time to train. Affter train and test, I also visualize result of RandomForestRegressor model by scatter plot.

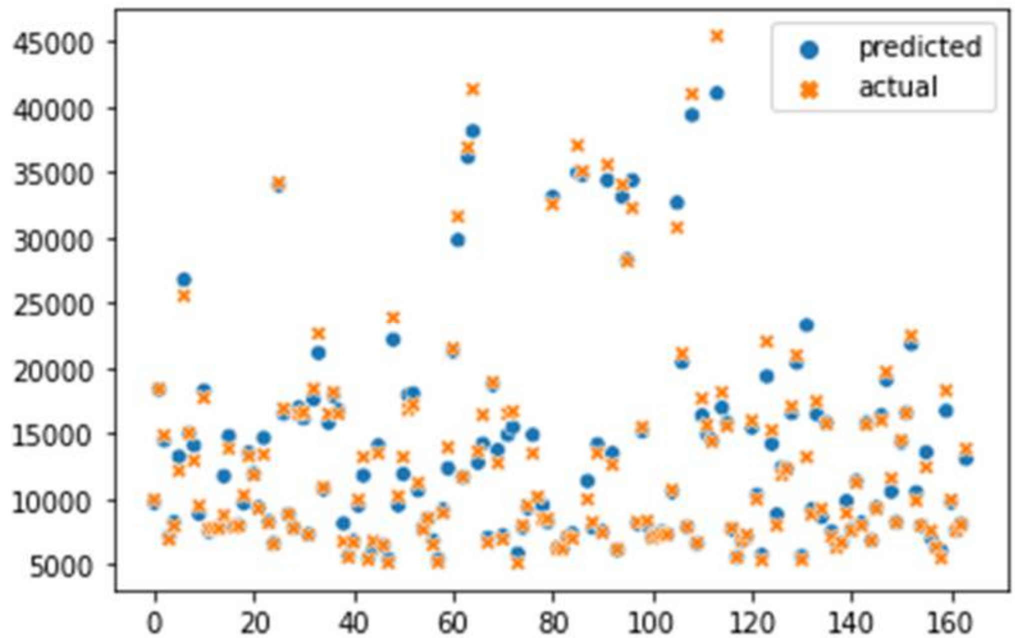


Figure 29: Result predicted value of RandomForestRegressor in train dataset

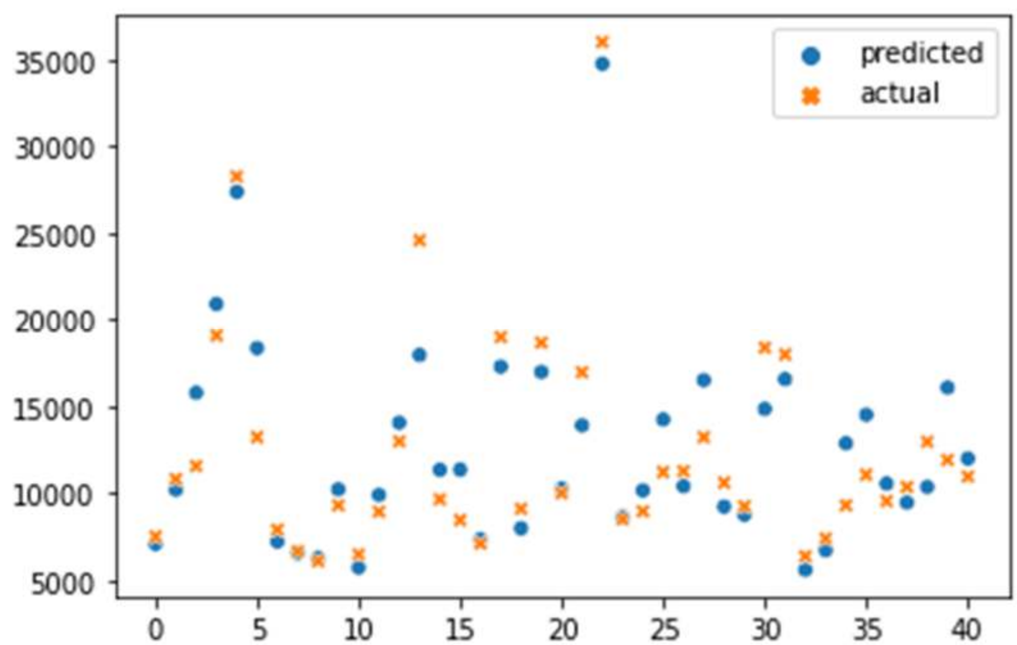


Figure 30: Result predicted value of RandomForestRegressor in test dataset

Conclusion

In this project, I have oppertunities to apply knowledge from data mining subject to analysis dataset, choice the best method to preprocess data, build model prediction and evalue the accuracy of model. Via this project, I know the importance of understanding dataset and preprocessing data. If we do not understand our data, we can not do reprocess data well, so the accuracy of our model is also bad. When we choice algorithm for training the model, we should try to do with multiple algorithms and evaluate the result in the training data to receive the best solution to deal with our problems.

Reference

https://realpython.com/linear-regression-in-python/

https://scikit-learn.org/stable/index.html

https://seaborn.pydata.org/

https://pandas.pydata.org/