EDA PROJECT FOR AUTOMOBILE DATASET

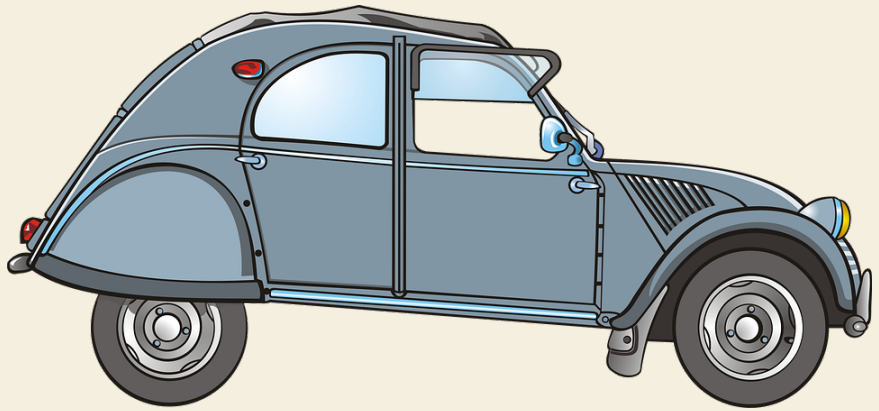
Data Science Foundation – UpX Academy

Rohit Subudhi

About Me

- Close to 10 years in IT Sector with consulting work experience
- Oracle ERP Technical Consultant and have worked in SCM, Master Data Management (MDM)
- Great with SQL, PLSQL
- Gathered interest on Data Analytics based on the MDM work done in the past which mostly revolves around data quality and data profiling
- Avid Quora reader
- Absolutely love travelling, like reading books. Want to learn more on Sanskrit, our culture and roots.

Project Description



- **Domain:** Automobile
- **Topic:** Automobile data
- **Technology Used:** Anaconda, Python, Jupyter Notebook
- **Dataset File Name:** Automobile_data.txt
- The dataset contains data of cars from different companies with multiple parameters of different models. The file is also having the price of each model of the car.
- The idea of this project to understand the data, clean the data, exploring a univariate and bivariate analysis of automobile data

Dataset Description

Attribute	Attribute Range
1. symboling	-3, -2, -1, 0, 1, 2, 3.
2. normalized-losses	continuous from 65 to 256.
3. make	alfa-romero, audi, bmw, chevrolet,dodge,honda, isuzu, jaguar, mazda, mercedes-benz, mercury, mitsubishi, nissan, peugot, plymouth, porsche, renault, saab, subaru, toyota, volkswagen volvo
4. fuel-type	diesel, gas.
5. aspiration	std, turbo.
6. num-of-doors	four, two.
7. body-style	hardtop, wagon, sedan, hatchback, convertible.
8. drive-wheels	4wd, fwd, rwd.
9. engine-location	front, rear.
10. wheel-base	continuous from 86.6120.9.
11. length	continuous from 141.1 to 208.1.
12. width	continuous from 60.3 to 72.3.
13. height	continuous from 47.8 to 59.8.
14. curb-weight	continuous from 1488 to 4066.
15. engine-type	dohc, dohcv, l, ohc, ohcf, ohcv, rotor.
16. num-of-cylinders	eight, five, four, six, three, twelve, two.
17. engine-size	continuous from 61 to 326.
18. fuel-system	1bbl, 2bbl, 4bbl, idi, mfi, mpfi, spdi, spfi.
19. bore	continuous from 2.54 to 3.94.
20. stroke	continuous from 2.07 to 4.17.
21. compression-ratio	continuous from 7 to 23.
22. horsepower	continuous from 48 to 288.
23. peak-rpm	continuous from 4150 to 6600
24. city-mpg	continuous from 13 to 49.
25. highway-mpg	continuous from 16 to 54.
26. price	continuous from 5118 to 45400.

- The file had **26** columns and **205** observations
- Missing values were identified by " ? "
- Missing values are present in "normalized-losses, num-of-doors, bore, stroke, horsepower, peak-rpm, price" columns
- Each observation consists of three types of entities:
  - The specification of a car in terms of various characteristics
    - 10 categorical variables (e.g. the make, fuel type, number of doors, etc.)
    - 14 continuous variables (e.g. length, weight, height, price, etc.).
  - Insurance risk rating is the degree to which the auto is more risky than its price indicates. A value of +3 indicates that the auto is risky, -3 that it is probably pretty safe
  - Normalized losses is the relative average loss payment per insured vehicle year. This value represents the average loss per car per year

Loading & Cleaning Data

- Start with importing the necessary

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

- Read the text file and load into a pandas Dataframe

```
automobile_df = pd.read_csv('Automobile_data.txt', sep=",", na_values='?')
```

- The parameter na_values='?' considers "?" as null values.

- See the first look of the data using 'head', 'tail' and 'info' method of the dataframe to view the first and the last 10 rows
- The 'Shape' and 'Columns' attributes of the dataframe show the size and the column names
- Identify column with null values and null value counts
  - automobile_df.isnull().sum()
  - automobile_df.columns[automobile_df.isna().any()].tolist()

```
missing_col_names = automobile_df.columns[automobile_df.isna().any()].tolist()
missing_col_names
```

```
['normalized-losses',
 'num-of-doors',
 'bore',
 'stroke',
 'horsepower',
 'peak-rpm',
 'price']
```

- For the column *"normalized-losses"* total missing values is 41, which is 20% of the data. This should not Imputed (fill the missing) with any value as it will simply add biasness to the whole dataset, so we **drop this column**

```
automobile_df.isnull().sum()
```

```
symboling           0
normalized-losses   41
```

Loading & Cleaning Data

- Find out all the rows with missing values (NaN). This results 12 rows × 26 columns

```
automobile_df[automobile_df['num-of-doors'].isnull() |
              automobile_df['bore'].isnull()         |
              automobile_df['stroke'].isnull()       |
              automobile_df['horsepower'].isnull()   |
              automobile_df['peak-rpm'].isnull()     |
              automobile_df['price'].isnull()
             ]
```

	symboling	normalized-losses	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wheel-base	...	engine-size	fuel-system	bore	stroke	compression-ratio	horsep
9	0	NaN	audi	gas	turbo	two	hatchback	4wd	front	99.5	...	131	mpfi	3.13	3.40	7.0	
27	1	148.0	dodge	gas	turbo	NaN	sedan	fwd	front	93.7	...	98	mpfi	3.03	3.39	7.6	
44	1	NaN	isuzu	gas	std	two	sedan	fwd	front	94.5	...	90	2bbl	3.03	3.11	9.6	
45	0	NaN	isuzu	gas	std	four	sedan	fwd	front	94.5	...	90	2bbl	3.03	3.11	9.6	
55	3	150.0	mazda	gas	std	two	hatchback	rwd	front	95.3	...	70	4bbl	NaN	NaN	9.4	
56	3	150.0	mazda	gas	std	two	hatchback	rwd	front	95.3	...	70	4bbl	NaN	NaN	9.4	
57	3	150.0	mazda	gas	std	two	hatchback	rwd	front	95.3	...	70	4bbl	NaN	NaN	9.4	
58	3	150.0	mazda	gas	std	two	hatchback	rwd	front	95.3	...	80	mpfi	NaN	NaN	9.4	
63	0	NaN	mazda	diesel	std	NaN	sedan	fwd	front	98.8	...	122	idi	3.39	3.39	22.7	
129	1	NaN	porsche	gas	std	two	hatchback	rwd	front	98.4	...	203	mpfi	3.94	3.11	10.0	
130	0	NaN	renault	gas	std	four	wagon	fwd	front	96.1	...	132	mpfi	3.46	3.90	8.7	
131	2	NaN	renault	gas	std	two	hatchback	fwd	front	96.1	...	132	mpfi	3.46	3.90	8.7	

12 rows × 26 columns

- Based on the analysis, treat the missing values to the dataframe
  - Drop the "normalized-losses" column
  - Drop all rows where missing value is found

```
#First drop the "normalized-losses" column
automobile_df.drop('normalized-losses', axis=1, inplace=True)

#Remove all the rows where NaN values are found. 12 rows should be removed
null_cols = ['num-of-doors', 'bore', 'stroke', 'horsepower', 'peak-rpm', 'price']
automobile_df.dropna(axis=0, inplace=True, subset = null_cols)

postmissing_df_shape = automobile_df.shape
print(postmissing_df_shape)
```

```
(193, 25)
```

- Check all the unique values for dataframe columns of dtype = 'Object'. Analyze these columns information and see if they can be changed to **"category"**

```
def get_unique_vals(df):
    dict_un={}
    for i in df.columns:
        if df[i].dtype == 'O':
            dict_un[i]= df[i].unique().tolist()
    return dict_un

print (automobile_df['symboling'].unique().tolist())

dict_un = get_unique_vals(automobile_df)
dict_un
```

```
[3, 1, 2, 0, -1, -2]

{'aspiration': ['std', 'turbo'],
 'body-style': ['convertible', 'hatchback', 'sedan', 'wagon', 'hardtop'],
 'drive-wheels': ['rwd', 'fwd', '4wd'],
 'engine-location': ['front', 'rear'],
 'engine-type': ['dohc', 'ohcv', 'ohc', 'l', 'ohcf'],
 'fuel-system': ['mpfi', '2bbl', 'mfi', '1bbl', 'spfi', 'idi', 'spdi'],
 'fuel-type': ['gas', 'diesel'],
 'make': ['alfa-romero',
  'audi',
  'bmw',
  'chevrolet'
```


Cont…

Loading & Cleaning Data

- All the columns with datatype as 'Object' have categorical values. Also, the *'symboling'* column is a categorical. The *'make'* column could be considered as a categorical value, but I thought not to do it.

```
cat_columns = ['fuel-type',
               'aspiration',
               'num-of-doors',
               'body-style',
               'drive-wheels',
               'engine-location',
               'engine-type',
               'num-of-cylinders',
               'fuel-system',
               'symboling'
              ]

for col in cat_columns:
    automobile_df[col] = automobile_df[col].astype('category')

automobile_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 193 entries, 0 to 204
Data columns (total 25 columns):
symboling            193 non-null category
make                 193 non-null object
fuel-type            193 non-null category
aspiration           193 non-null category
num-of-doors         193 non-null category
body-style           193 non-null category
drive-wheels         193 non-null category
engine-location      193 non-null category
wheel-base           193 non-null float64
length               193 non-null float64
width                193 non-null float64
height               193 non-null float64
curb-weight          193 non-null int64
engine-type          193 non-null category
num-of-cylinders     193 non-null category
engine-size          193 non-null int64
fuel-system          193 non-null category
bore                 193 non-null float64
stroke               193 non-null float64
compression-ratio    193 non-null float64
horsepower           193 non-null float64
peak-rpm             193 non-null float64
city-mpg             193 non-null int64
highway-mpg          193 non-null int64
price                193 non-null float64
dtypes: category(10), float64(10), int64(4), object(1)
```

- After removing all observations including missing values, 193 rows and 25 columns are retained. 10 are Categorical columns and 14 are continuous values

Cont…

Summary of the Data Cleaning Activity

- First loaded the data into a DataFrame. **The dataset had a shape of 205 rows, 26 columns.**
- Initial analysis suggested that there are missing values in the text file. Instead of keeping it blank the missing values are marked with the character "?".
- The column "*normalized-losses*" has 20% of data which are missing. So instead of imputing the column with mean/median value it was best to remove the column itself for any analysis. Filling missing values to this column might introduce biasness to the data.
- The columns *[num-of-doors, bore, stroke, horsepower, peak-rpm, price etc.]* have few missing values and based on the earlier analysis we decided to remove the rows where we have missing values in any of these columns.
- **After removing observations with missing values, 193 rows and 25 columns are retained.**
- Columns *[fuel-type, aspiration, num-of-doors, body-style, drive-wheels, engine-location, engine-type, num-of-cylinders, fuel-system, symboling]* were converted into **Categorical**.

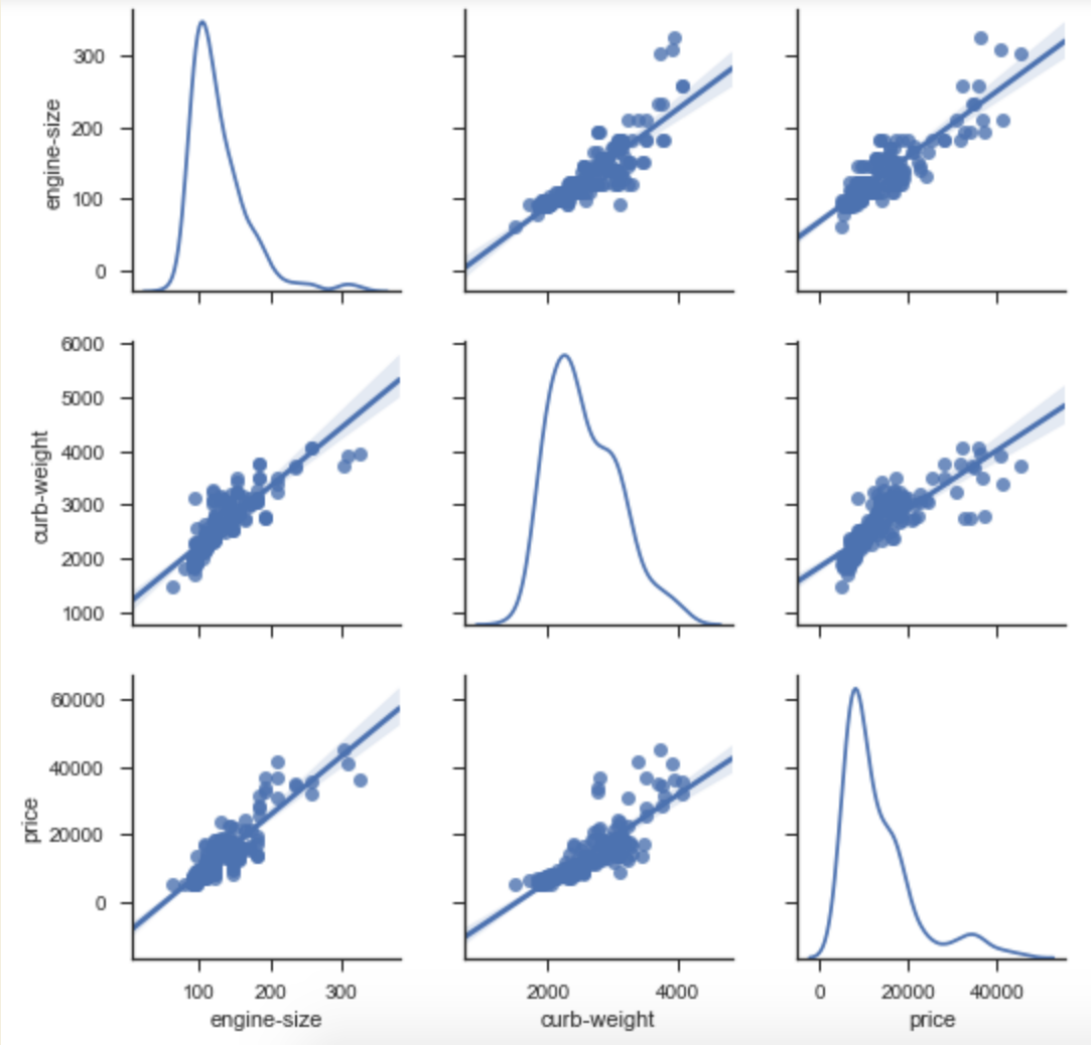
Business Question Answered

Engine Size, Curb-Weight, Price

More the engine size more is the curb weight of the car, and more is the price of the car.

This is evident from the pair plot, all have a positive correlation to each other.

```
plt.figure(figsize=(9,7))
ax = sns.pairplot(data=automobile_df,
                  kind='reg',
                  vars=["engine-size", "curb-weight", "price"],
                  diag_kind='kde')
plt.show()
```

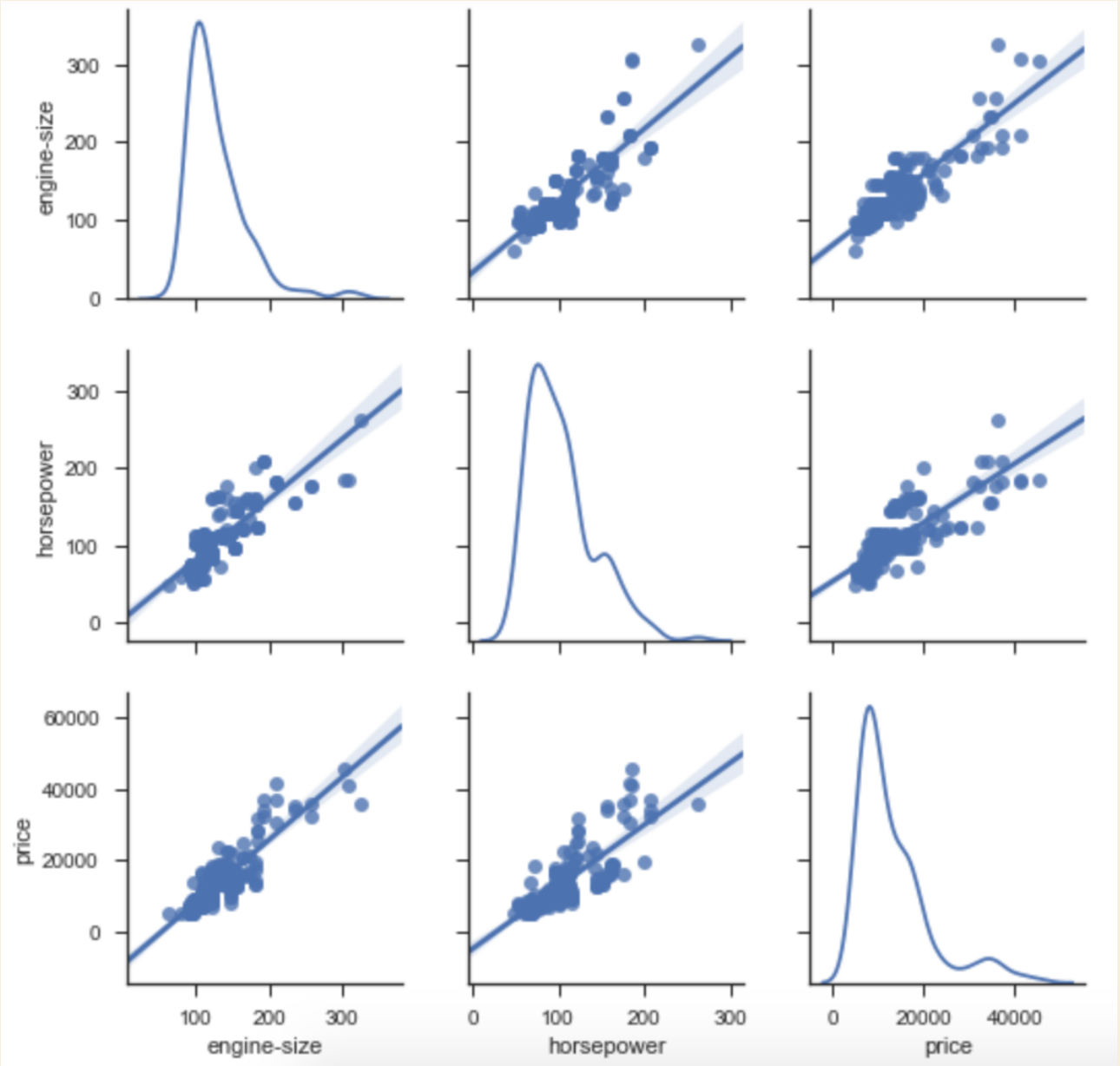


Business Question Answered

Engine Size, Horsepower & Price

All these variables are very highly correlated to each other.

It's also obvious and the pairplot confirms it

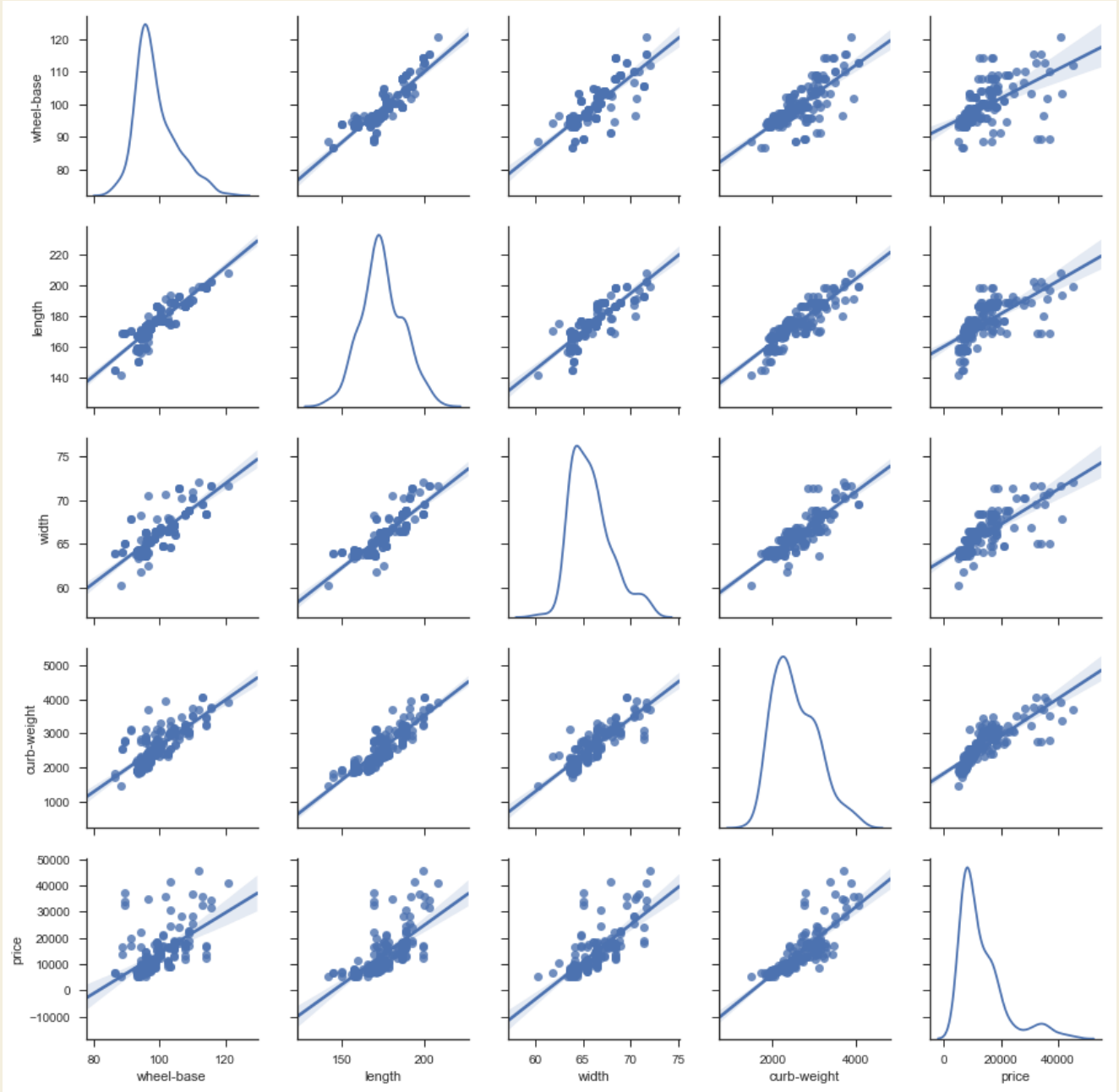


Business Question Answered

Wheel Base, Length, Width, Curb-Weight & Price

All these variables are very highly correlated to each other.

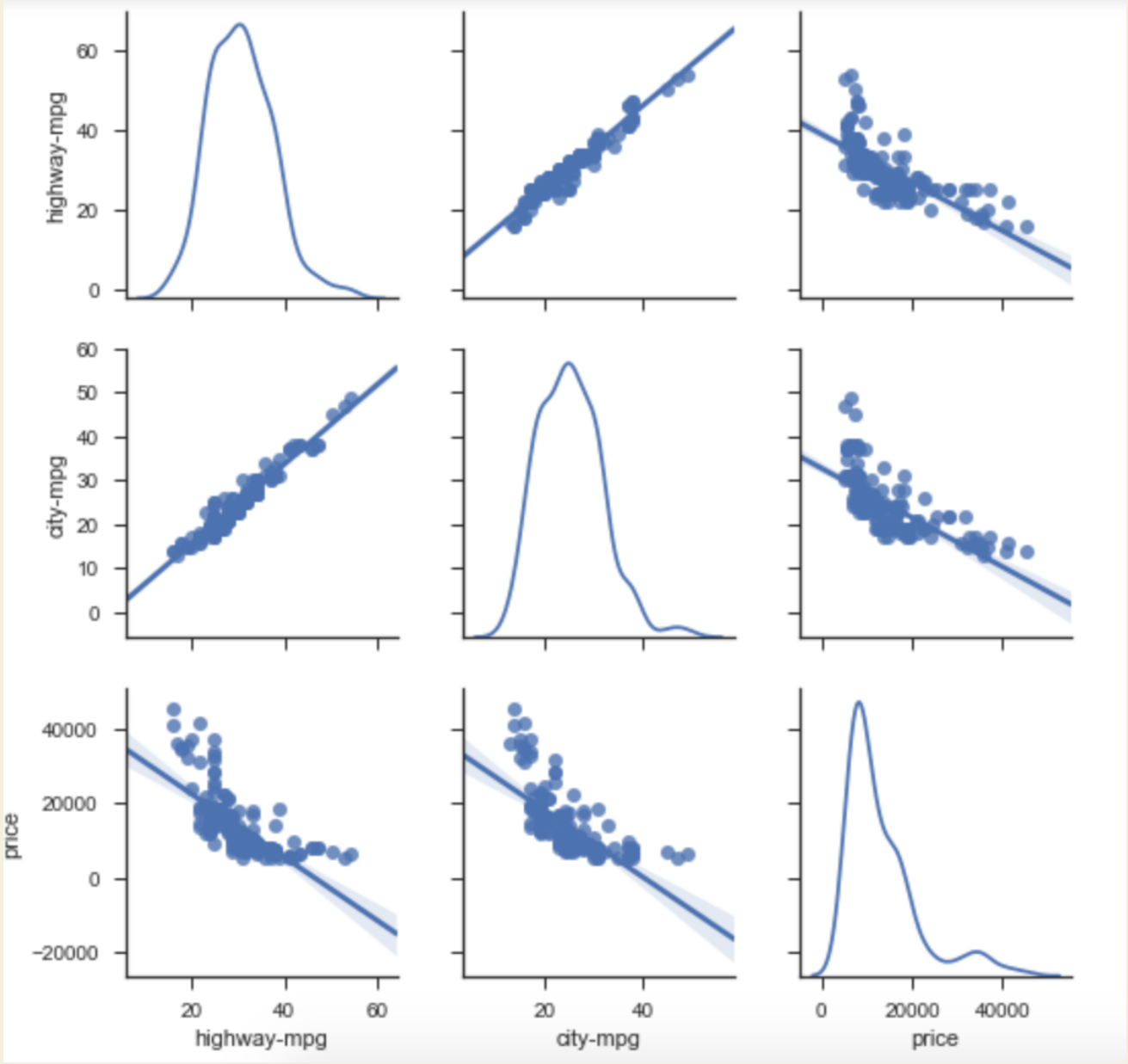
It's also obvious and the pairplot confirms it



Business Question Answered

Highway, City mpg & Price

Highway and City mpg are highly related to each other. Lesser the value of mpg, costlier the car is.

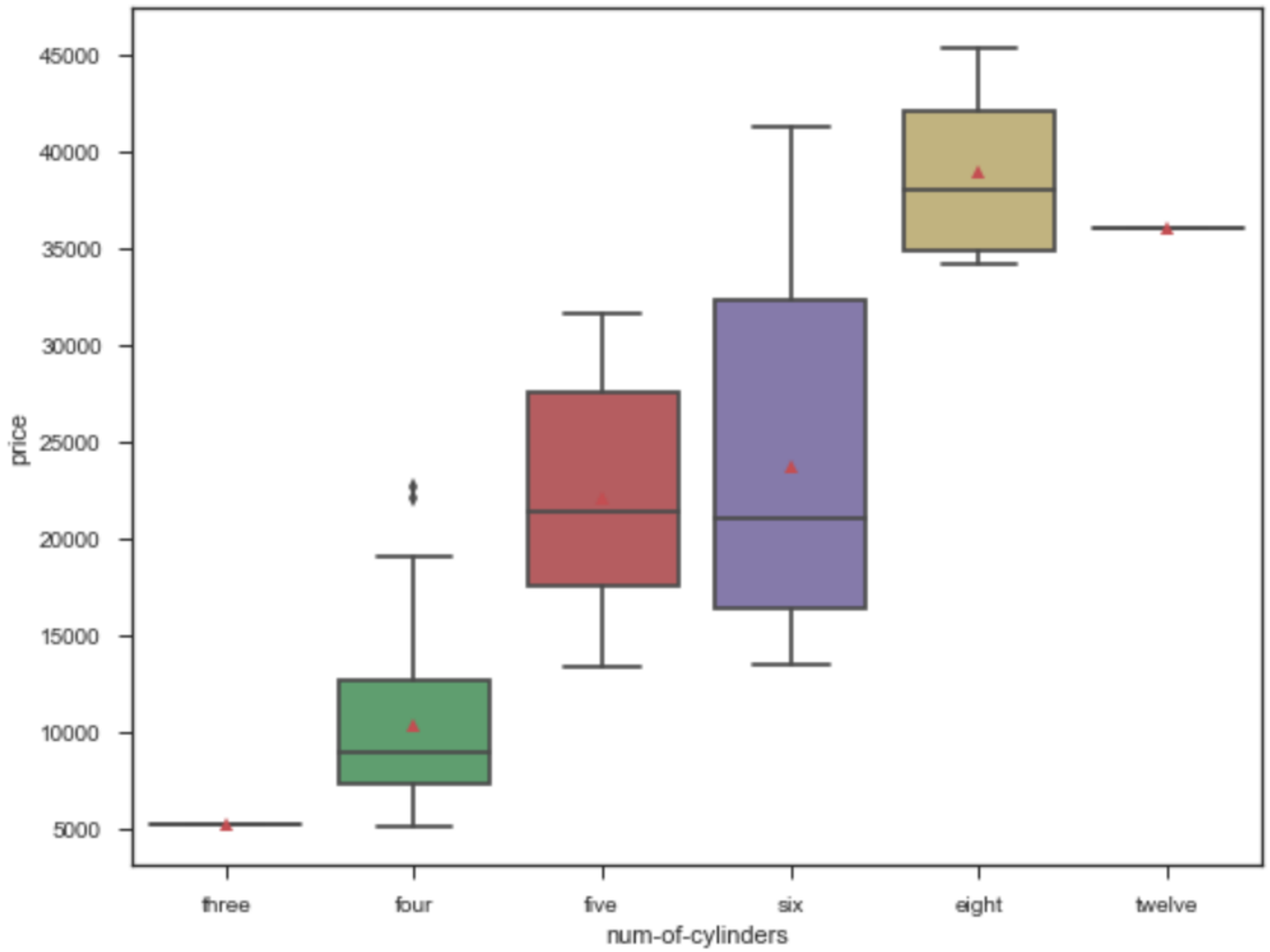


Business Question Answered

No. of Cylinders & Price

More the number of cylinders more is the price of the car. Average cars are mostly with four cylinders however the high performance cars are with more number of cylinders and thus the price increases

```
plt.figure(figsize=(9,7))
ax = sns.boxplot(x="num-of-cylinders", y="price", data=automobile_df,
                 order=['three', 'four', 'five', 'six', 'eight', 'twelve'],
                 showmeans=True)
plt.title('Number of Cylinders, Price Relation')
plt.show()
```

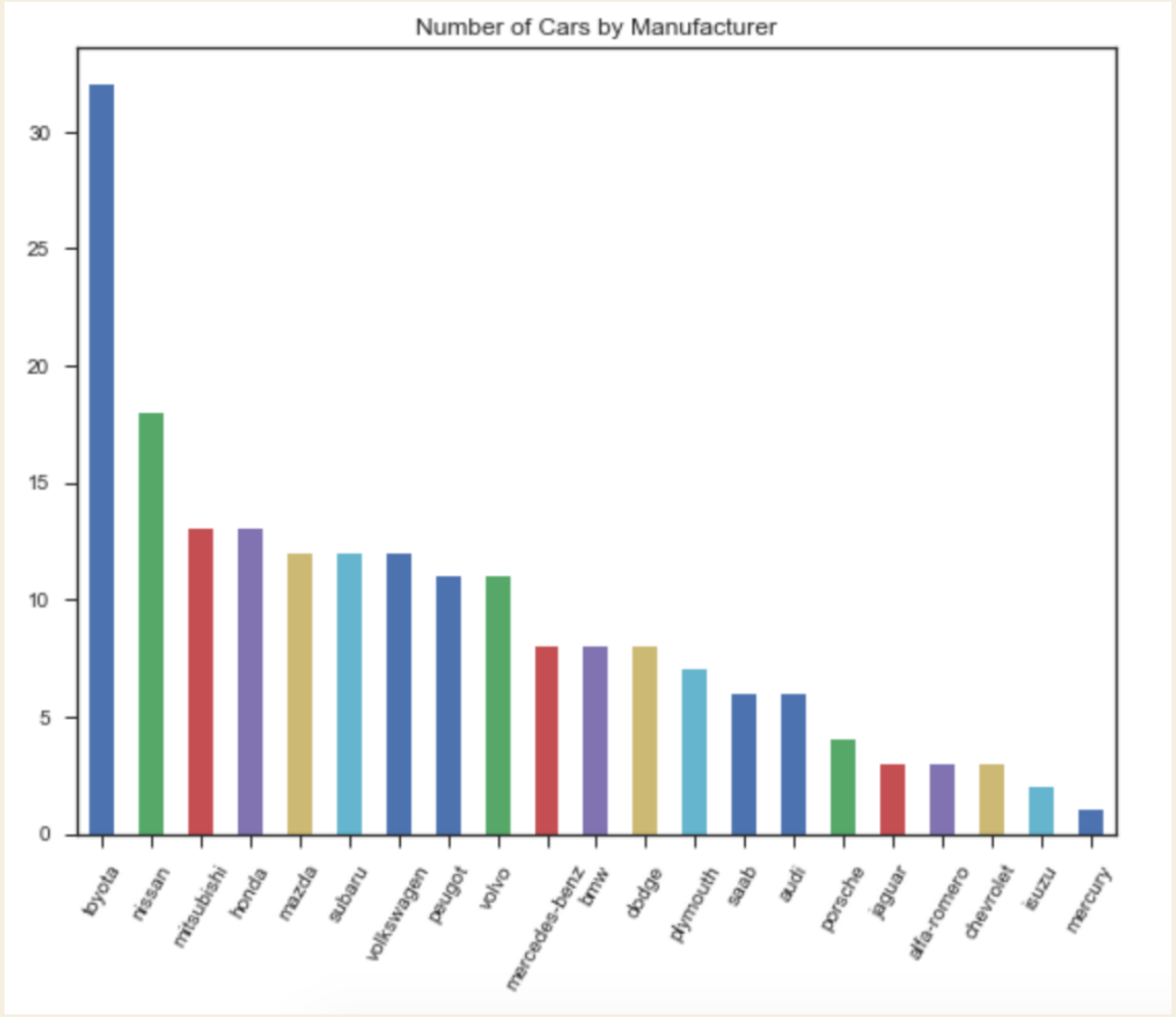


Business Question Answered

Most number of Cars

Toyota makes the most number of cars and its one of the largest car manufacturer in terms of volume

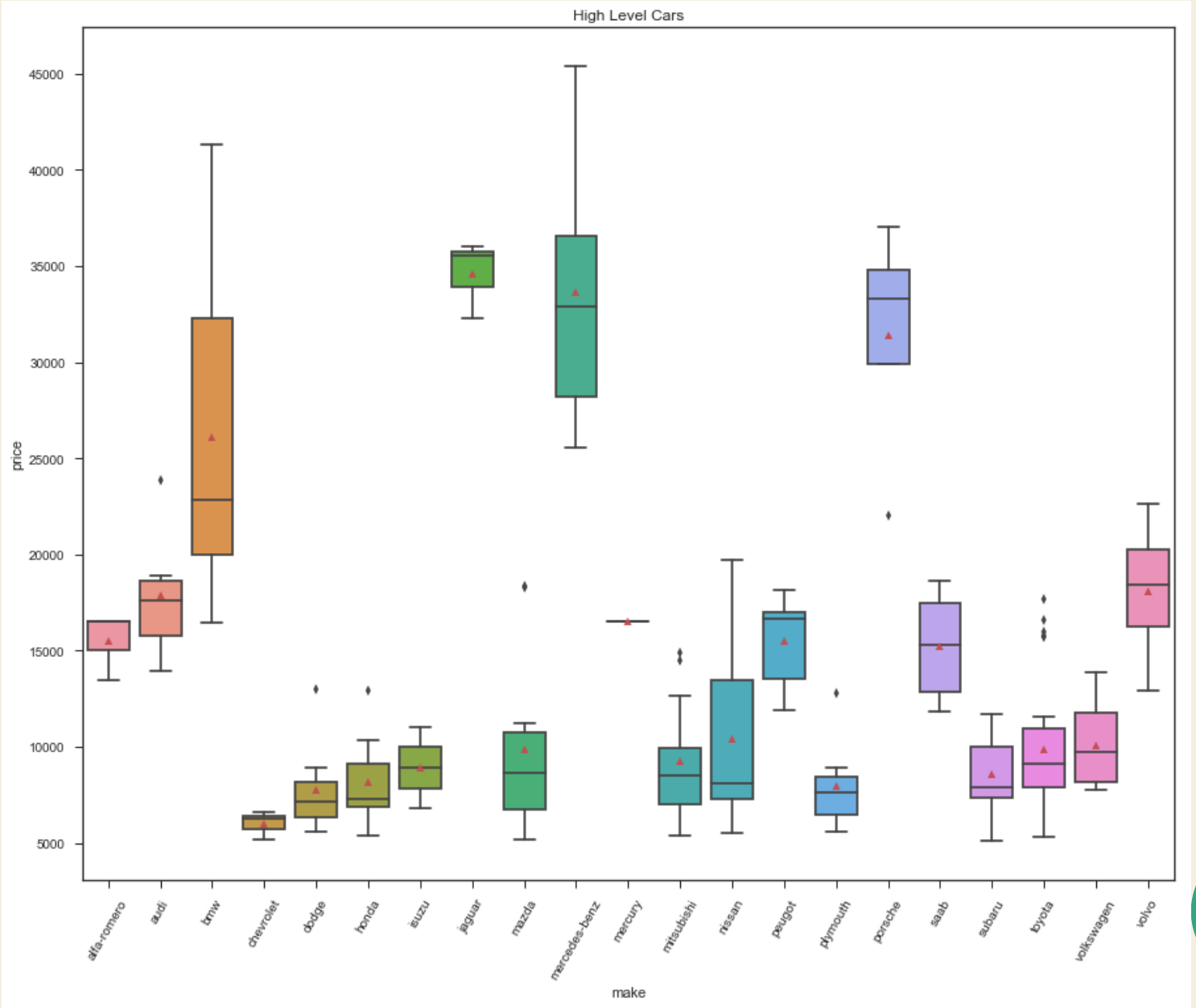
```
plt.figure(figsize=(9,7))
ax = automobile_df['make'].value_counts().plot(kind='bar')
plt.title('Number of Cars by Manufacturer')
plt.xticks(rotation=60)
plt.show()
```



Business Question Answered

List of High Performance Cars

Brands drive values and price. Cars of BMW, Jaguar, Mercedes-Benz, Porsche have a much high price compared to other brands. Chevrolet is one of the cheapest cars in the range



Summary

With this EDA project we found

- Loading and cleaning the dataset
- How the data set are distributed
- Correlation between different fields and how they are related
- Factors affecting Price of the cars

THANK YOU!