Analysis on mpg dataset

Data Description:

This Analysis aims primarily the visualization of the dataset to predict the Miles Per Gallon (MPG) using the factors provided in the data-set

mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
18	8	307	130	3504	12	70	1	chevrolet chevelle malibu
15	8	350	165	3693	11.5	70	1	buick skylark 320
18	8	318	150	3436	11	70	1	plymouth satellite
16	8	304	150	3433	12	70	1	amc rebel sst
17	8	302	140	3449	10.5	70	1	ford torino
15	8	429	198	4341	10	70	1	ford galaxie 500
14	8	454	220	4354	9	70	1	chevrolet impala
14	8	440	215	4312	8.5	70	1	plymouth fury iii
14	8	455	225	4425	10	70	1	pontiac catalina
15	8	390	190	3850	8.5	70	1	amc ambassador dpl

So, there it is, lots of numbers. We can see that the dataset has the following columns (with their type):

- **mpg**: continuous
- **cylinders**: multi-valued discrete
- **displacement**: continuous
- **horsepower**: continuous
- **weight**: continuous
- **acceleration**: continuous
- **model year**: multi-valued discrete
- **origin**: multi-valued discrete where 1 = USA, 2 = Europe, 3 = Japan
- **car name**: string (unique for each instance)

Title: Auto-Mpg Data
Number of Instances: 398
Number of Attributes: 9

All the attributes are self-explanatory

Data Pre-processing:

Checking out for null values in the dataset. It seems that horsepower has 6 null values. If we drop those values then there is no effect in the original dataset. So we will drop those rows.

```
df.isnull().sum()

mpg             0
cylinders       0
displacement    0
horsepower      6
weight          0
acceleration    0
model_year      0
origin          0
name            0
dtype: int64
```

There are few numerical attributes and categorical attributes which are

nums = ['mpg', 'displacement', 'horsepower', 'acceleration', 'weight']

cats = ['cylinders', 'model_year', 'origin']

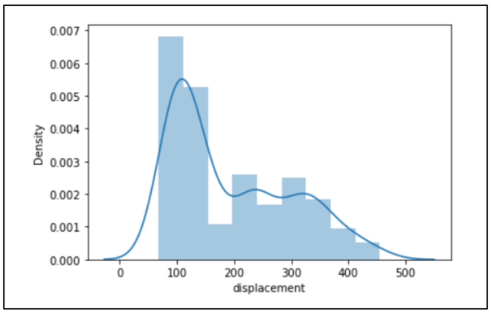
Let's first look what values all these data represent.

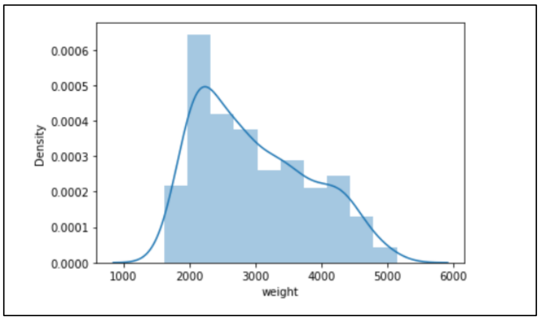
Statistical description of continuous features / numerical attributes

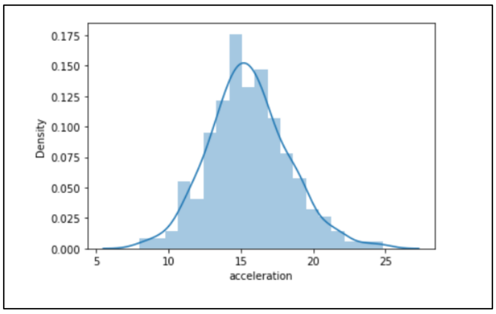
	mpg	displacement	horsepower	weight	acceleration
count	398.000000	398.000000	392.000000	398.000000	398.000000
mean	23.514573	193.425879	104.469388	2970.424623	15.568090
std	7.815984	104.269838	38.491160	846.841774	2.757689
min	9.000000	68.000000	46.000000	1613.000000	8.000000
25%	17.500000	104.250000	75.000000	2223.750000	13.825000
50%	23.000000	148.500000	93.500000	2803.500000	15.500000
75%	29.000000	262.000000	126.000000	3608.000000	17.175000
max	46.600000	455.000000	230.000000	5140.000000	24.800000

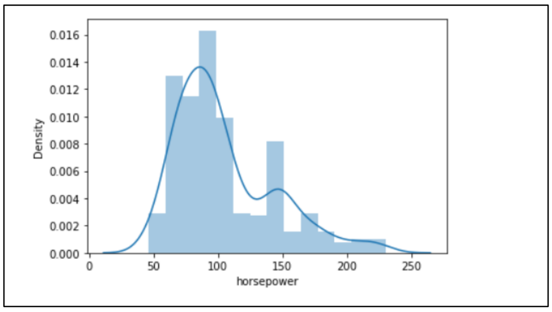
Median

Look at the distribution plots for the numerical values:



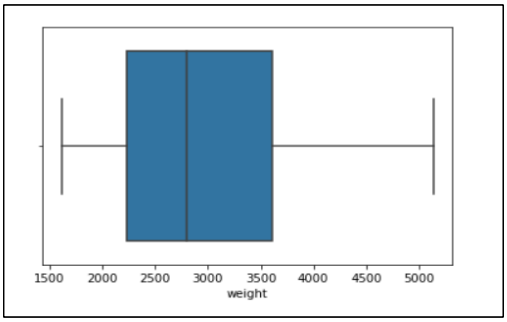


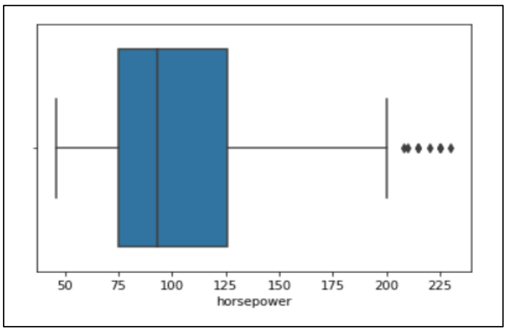


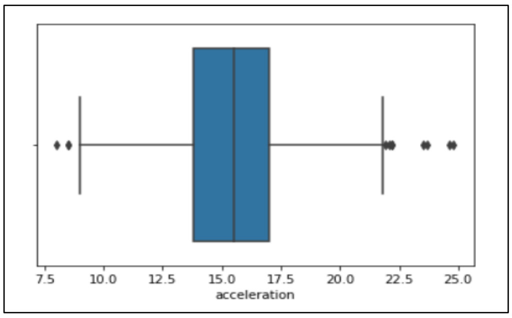


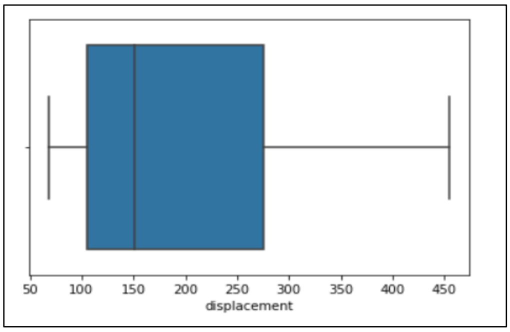
- acceleration is the only distribution which is gaussian.
- distributions of weight seem to be right-skewed gaussian.
- distributions of displacement & horsepower seem to be far from gaussian.

Boxplots:









It seems like that the horsepower and acceleration have outliers within them.

Next, we have to define the weight label as follows:

df['weight-label'] = pd.cut(x=df['weight'], bins=[1613, 2224, 2804, 3608, 5140], labels=['1613-2224', '2224-2804', '2804-3608','3608-5140'])

After defining the weight label, we can segregate the name column into company and car name. This step is followed:

df['company'] = df['name'].apply(lambda x: x.split()[0])

df['car_name'] = df['name'].apply(lambda x: ' '.join(x.split()[1:]))

Drop the column name from the dataset

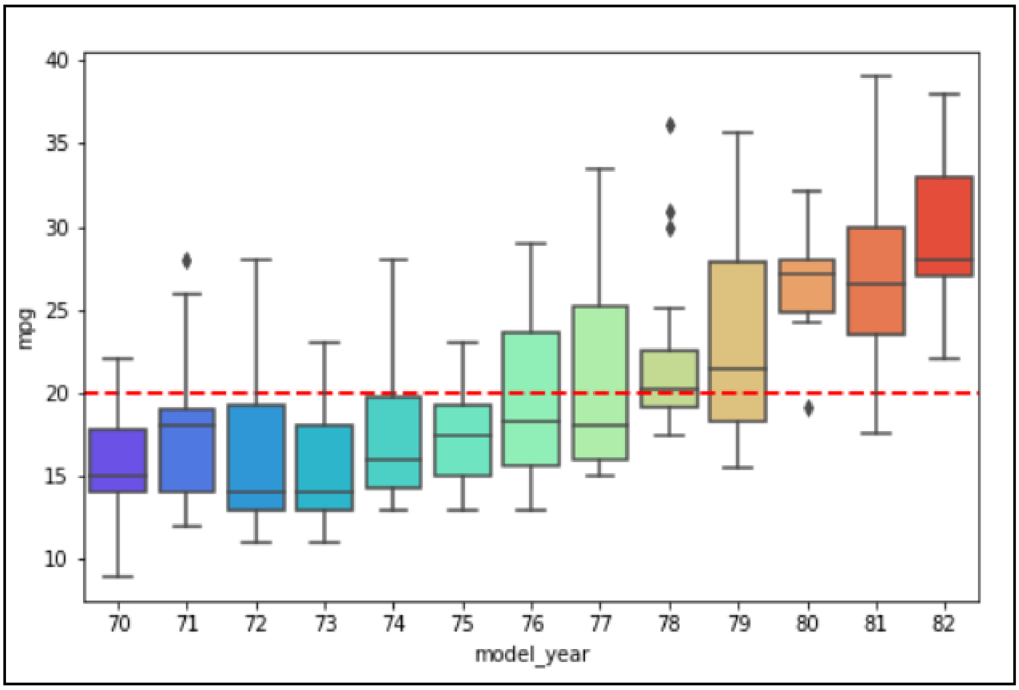
df.drop('name',axis=1,inplace=True)

Considering the models in USA

```
model_usa = df[df['origin']=='usa']
```

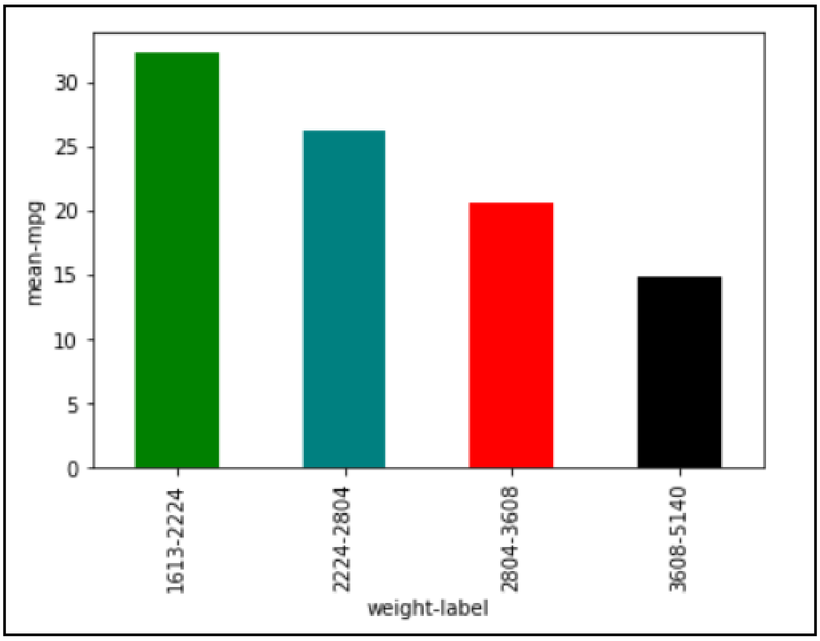
grouping the model_year parameter a description of mpg is made with the boxplots shown below

	count	mean	std	min	25%	50%	75%	max
model_year								
70	22.0	15.272727	3.507568	9.0	14.000	15.00	17.750	22.0
71	20.0	18.100000	4.655500	12.0	14.000	18.00	19.750	28.0
72	18.0	16.277778	4.860471	11.0	13.000	14.00	19.250	28.0
73	29.0	15.034483	3.212506	11.0	13.000	14.00	18.000	23.0
74	15.0	18.333333	4.775932	13.0	14.500	16.00	20.500	28.0
75	20.0	17.550000	2.855742	13.0	15.000	17.50	19.250	23.0
76	22.0	19.431818	4.981622	13.0	15.625	18.25	23.625	29.0
77	18.0	20.722222	5.973525	15.0	16.000	18.00	25.250	33.5
78	22.0	21.772727	4.800216	17.5	19.200	20.20	22.600	36.1
79	23.0	23.478261	6.419286	15.5	18.350	21.50	27.900	35.7
80	7.0	25.914286	4.106673	19.1	23.950	26.40	27.950	32.1
81	13.0	27.530769	6.066353	17.6	23.500	26.60	30.000	39.0
82	20.0	29.450000	4.871777	22.0	26.750	28.00	32.500	38.0



The red dotted line indicates the mean mpg line for the models in USA

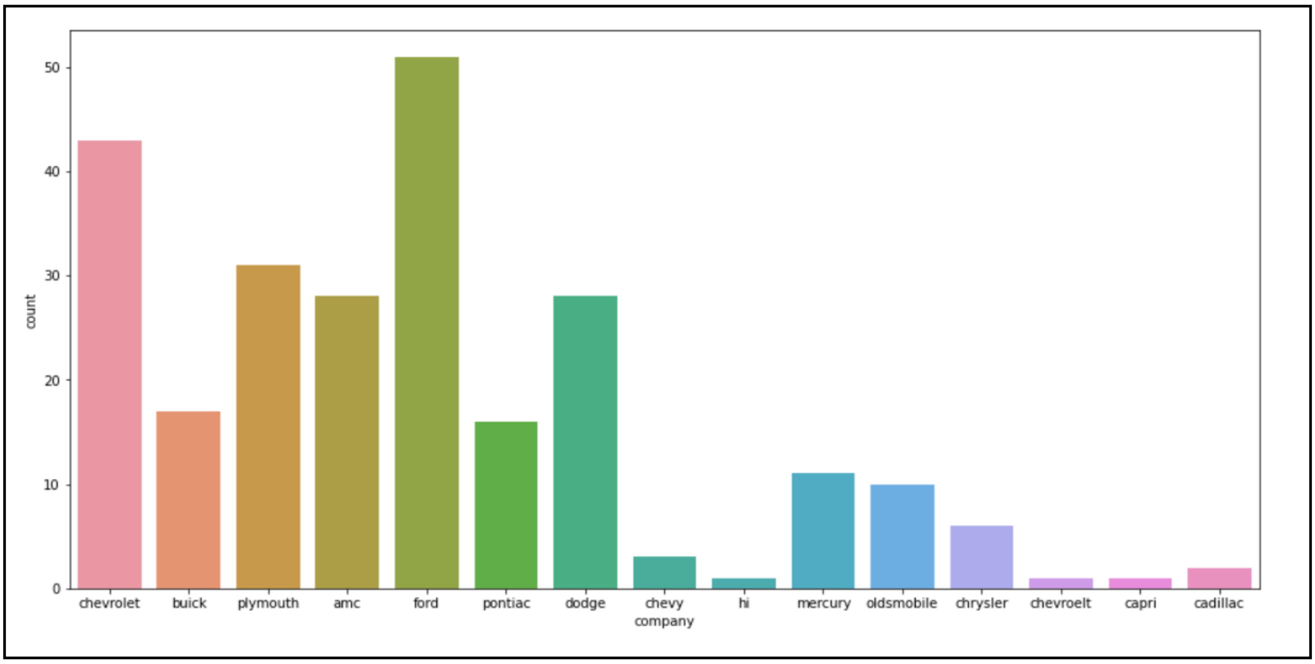
Similarly, we can group the weight labels visualize the output for the mpg



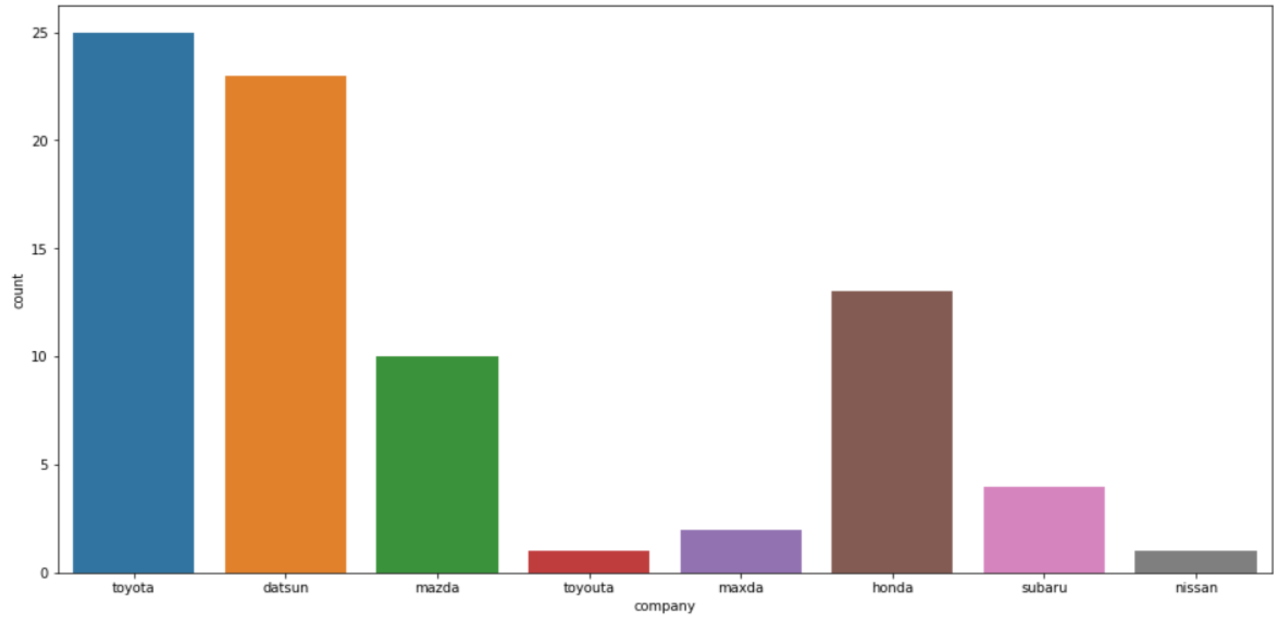
With increase in weight the model's mpg decreases

I have used the dataset for finding the companies so that we can know about the production of models in different parts of USA, Europe and Japan

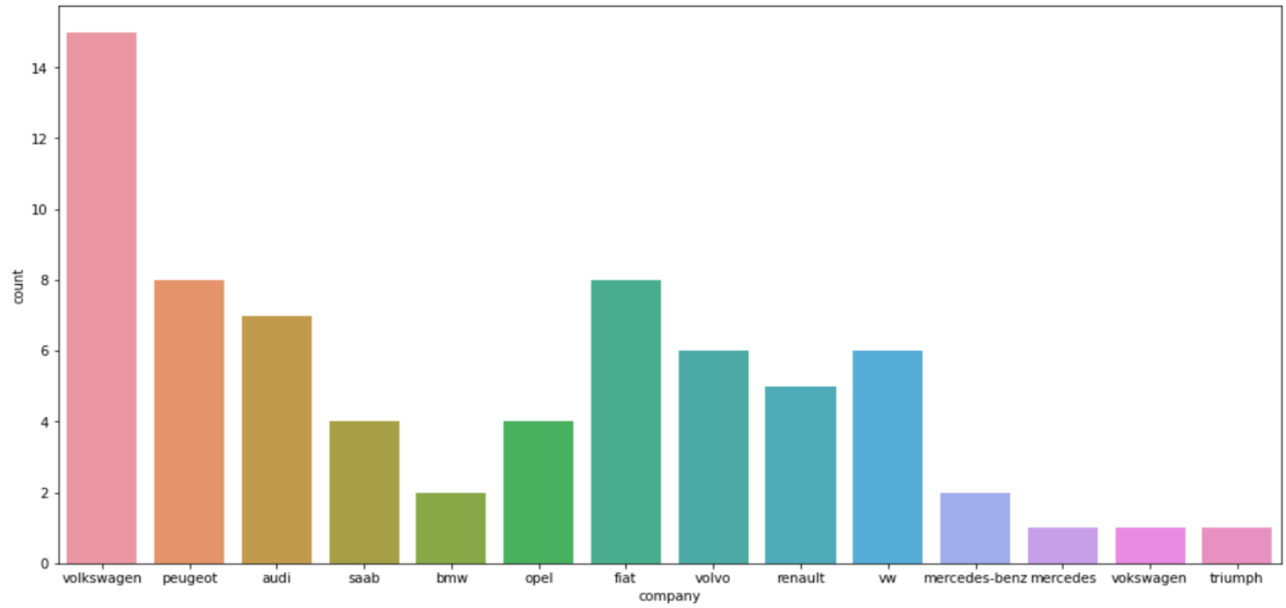
In USA,



In Japan,



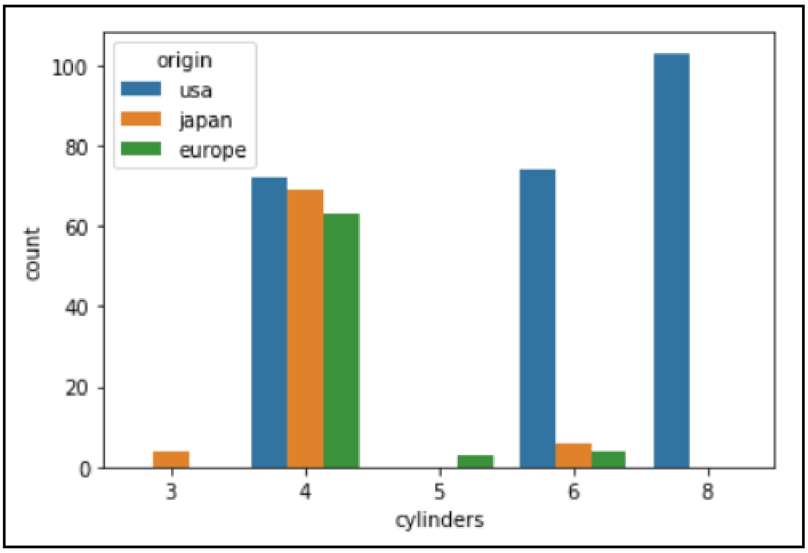
In Europe,



Next, I have used the mpg column to define three levels as low medium and high

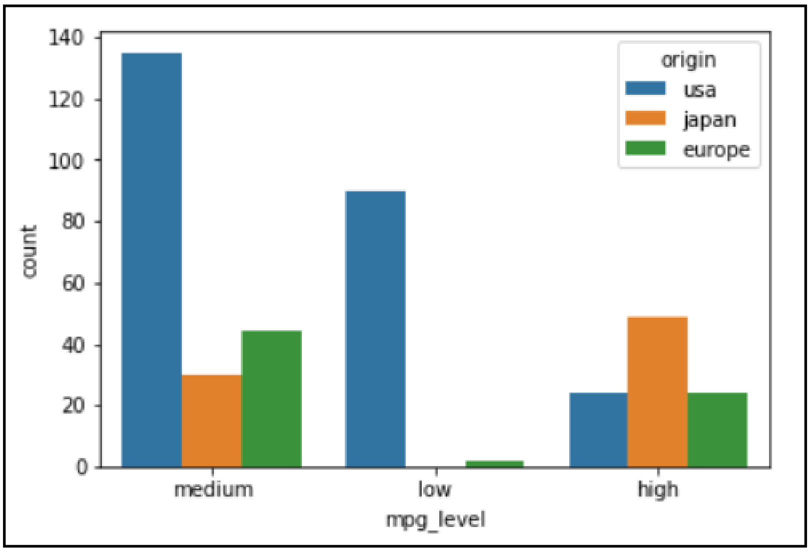
```
df['mpg_level'] = df['mpg'].apply(lambda x: 'low' if x<17 else 'high' if x>29 else 'medium')
```

A count plot for the cylinders with defined origin

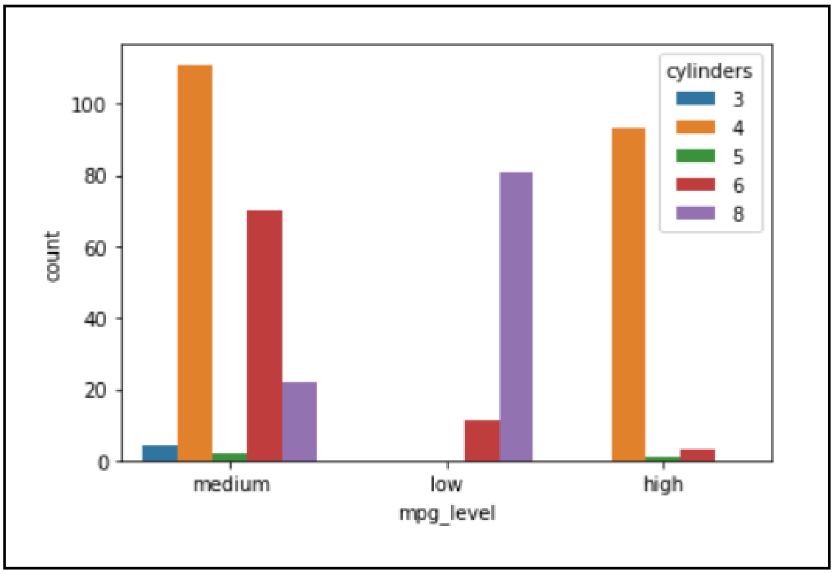


- Japan is the only origin with vehicles having 3 cylinders.
- Europe is the only origin with vehicles having 5 cylinders.
- USA is the only origin with vehicles having 8 cylinders.
- All origins have 4-cylinder vehicles and in almost equal proportion, also because 4 is dominating in cylinders.
- All origins have 6-cylinder vehicles but dominated by USA due the fact that it is dominating in origin.

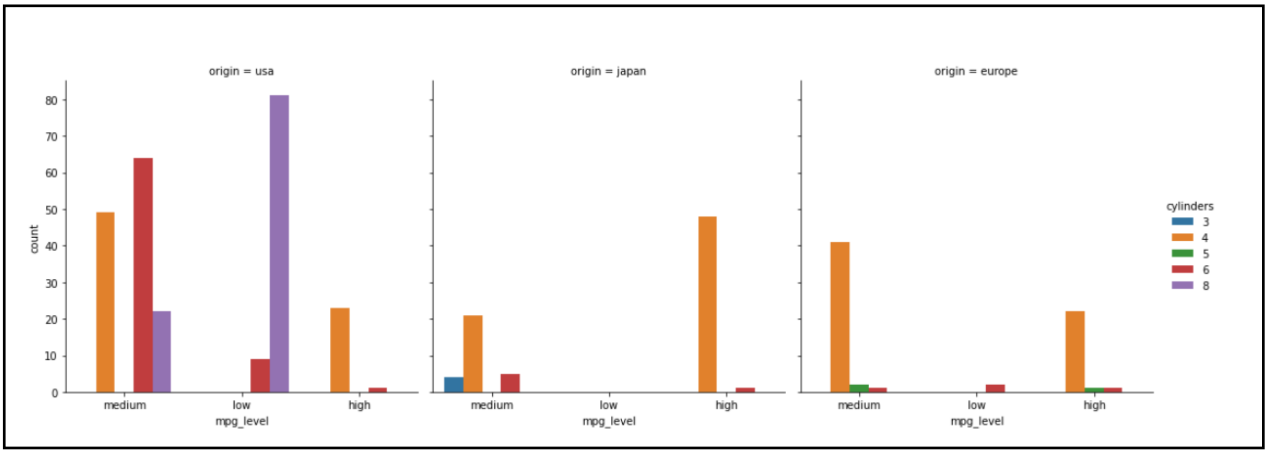
With the mpg_level, I have made a plot with the origins producing the required output



mpg_level with cylinders produced the following graph

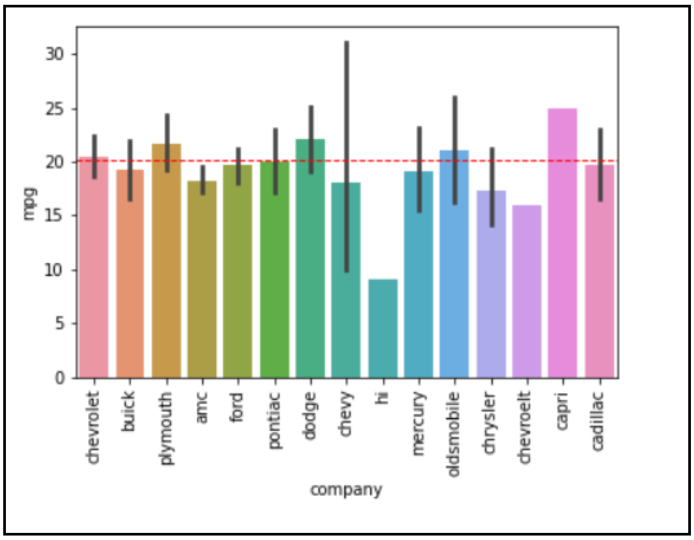


A seaborn catplot is used to layout the following output

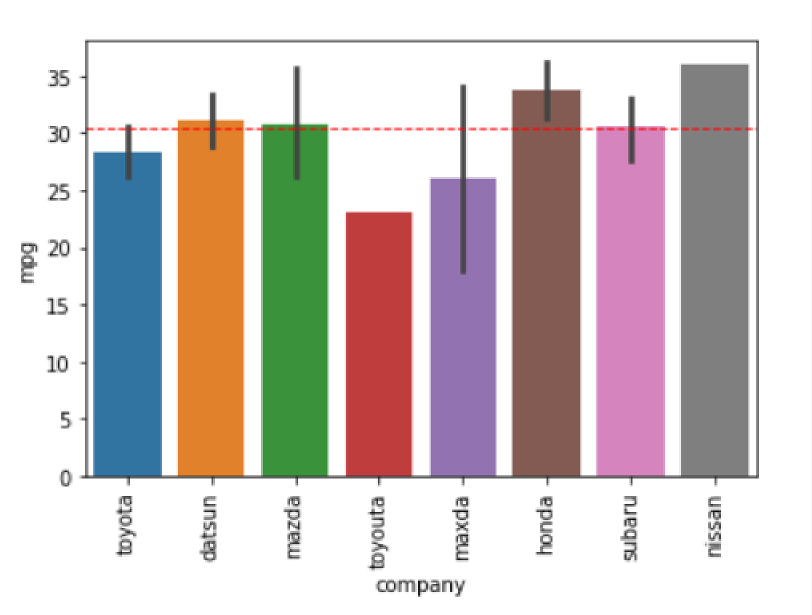


The visualization part is over with the bar plots in action.

In USA,



In Japan,



In Europe,