

EDA Report - Automobile Dataset

Introduction

The purpose of this report is to document the Exploratory data analysis (EDA) carried out on the automobiles dataset.

```
In [416... # Import Libraries
import numpy as np
import pandas as pd
import seaborn as sns

from datetime import datetime
import matplotlib.pyplot as plt
%matplotlib inline
```

```
In [417... # Load the automobile dataset and print an error message
try:
    automobiles_df = pd.read_csv('automobile.txt')
except:
    print('This file is not available')
```

Data Cleaning

The following steps were followed to clean the automobile data:

1. The redundant columns were identified and removed
2. An additional column 'Count' was added to be used in the groupby functions
3. The data was checked to ensure that there are no duplicates in the data
4. The rows with missing data were identified and removed
5. Numerical data was manipulated to ensure that it was of integer type

Below is a description of the data in the automobiles dataset. The count of each column is 205. This indicates that an entry exists for each row and column in the data. The data needs to be checked to ensure that each entry is valid.

The mean, standard deviation, minimum, maximum 25th percentile, 50th percentile and 75th percentile all seem reasonable and do not indicate any obvious problems with the data.

```
In [418... # removed 'symbolizing' and 'normalized-losses' from
# automobiles_df dataframe
automobiles_df = automobiles_df.drop(
    ['symbolizing', 'normalized-losses'], axis = 1)
pd.set_option('display.float_format', lambda x: '%.0f' % x)
automobiles_df.describe()
```

Out[418...

	wheel- base	length	width	height	curb- weight	engine- size	compression- ratio	city- mpg	highway- mpg
count	205	205	205	205	205	205	205	205	205
mean	99	174	66	54	2556	127	10	25	31
std	6	12	2	2	521	42	4	7	7
min	87	141	60	48	1488	61	7	13	16
25%	94	166	64	52	2145	97	9	19	25
50%	97	173	66	54	2414	120	9	24	30
75%	102	183	67	56	2935	141	9	30	34
max	121	208	72	60	4066	326	23	49	54

In [419...

```
# add an extra column 'Count' for use in groupby functions
automobiles_df['Count'] = 1
```

Remove any duplicate rows

The data was checked for duplicates. None of the rows were duplicated and therefore, no data needed to be removed.

In [420...

```
# find duplicated rows, save them in
# 'duplicated_rows_df' dataframe.
# print length of 'duplicated_rows_df'
# to check is duplicates exist
duplicated_rows_df = automobiles_df[automobiles_df.duplicated()]
print(f"Number of duplicates: {len(duplicated_rows_df)}")
```

Number of duplicates: 0

In []:

Remove rows with missing data

Some automobiles in the database have missing values. I found that the missing data entries was replaced by a '?'. Therefore, all the rows with a '?' in at least one of the columns was removed. The dataset now consists of 193 rows and 25 columns.

In [421...

```
# loop though columns to check if entries with '?' exist
# if '?' exists, then drop the row from automobiles dataframe

for column in automobiles_df.columns:
    index_to_drop = automobiles_df[
        automobiles_df[column] == '?'].index
    automobiles_df.drop(index_to_drop, inplace=True)
```

I noticed that all the data in the columns were of type 'string'. I changed columns with numerical data to an integer data type. Due to the decimals in the rows, I needed to first change all the numerical data into a float and then an integer.

```
In [422... # change columns with numerical data to float
automobiles_df[
    ['wheel-base', 'length', 'width', 'height',
     'curb-weight', 'engine-size', 'bore', 'stroke',
     'compression-ratio', 'horsepower', 'peak-rpm',
     'city-mpg', 'highway-mpg',
     'price']
] = automobiles_df[
    ['wheel-base', 'length', 'width', 'height',
     'curb-weight', 'engine-size', 'bore',
     'stroke', 'compression-ratio', 'horsepower',
     'peak-rpm', 'city-mpg', 'highway-mpg', 'price']
].astype(float)

# change columns with type float to integers
automobiles_df[
    ['wheel-base', 'length', 'width', 'height', 'curb-weight',
     'engine-size', 'bore', 'stroke', 'compression-ratio',
     'horsepower', 'peak-rpm', 'city-mpg', 'highway-mpg', 'price']
] = automobiles_df[
    ['wheel-base', 'length', 'width', 'height',
     'curb-weight', 'engine-size', 'bore', 'stroke',
     'compression-ratio', 'horsepower', 'peak-rpm',
     'city-mpg', 'highway-mpg', 'price']].astype(np.int64)
```

Data stories and visualisations

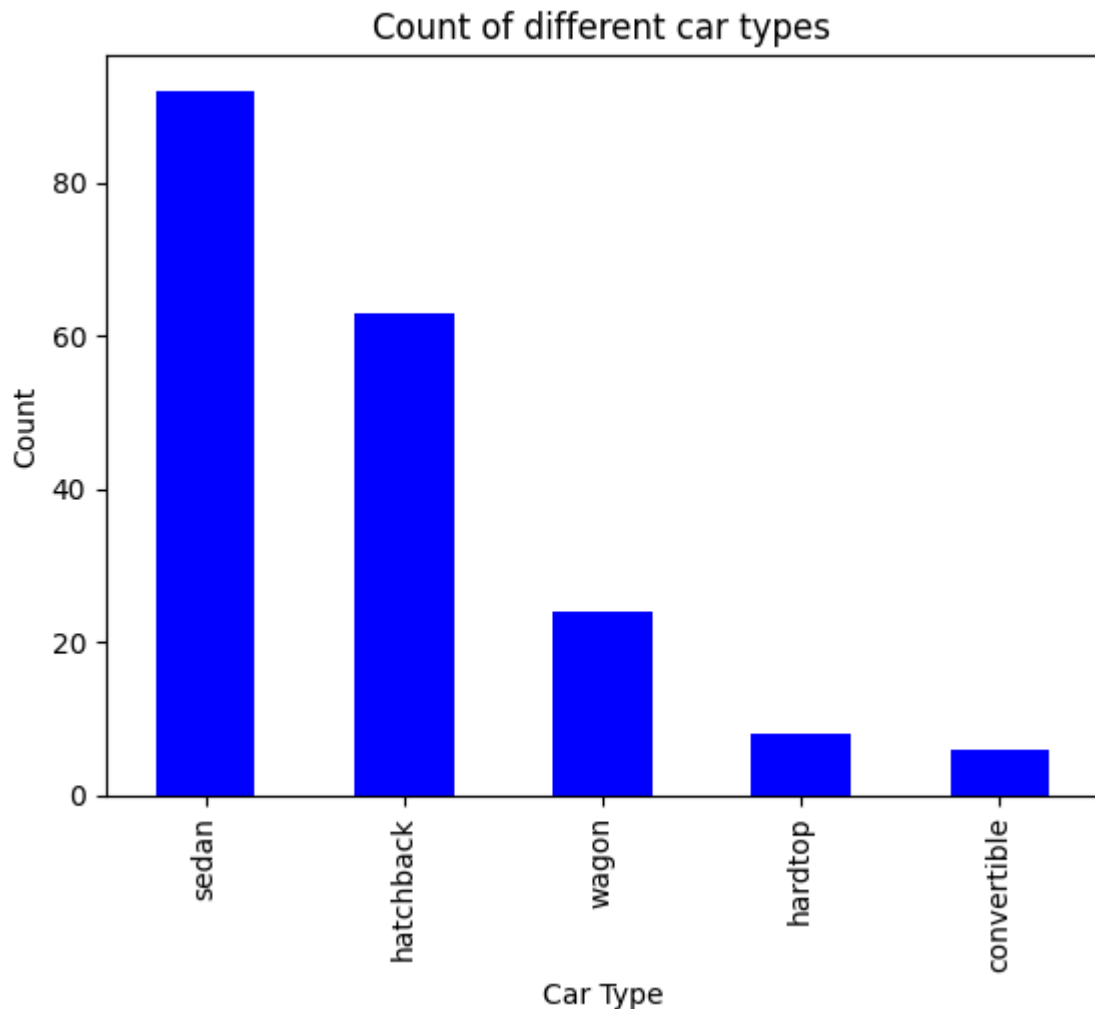
Different types of cars

The bar chart below shows the number of different car types in the data. The car types consist of 'convertible', 'hardtop', 'hatchback', 'sedan' and 'wagon'.

The most common car type in the data is the sedan (see bar chart below), followed by the hatchback and then the wagon. This makes sense since sedans are larger cars and would make better family cars than hatchbacks, hardtops and convertibles. Hardtop and convertible is the least common car type in the data. I will analyse the types of cars in more detail below.

```
In [423... car_types_df = automobiles_df.groupby(
    'body-style')['Count'].sum().sort_values(ascending=False)

ax = car_types_df.plot.bar(
    title='Count of different car types',
    rot = 90, xlabel='Car Type', ylabel='Count',
    color='Blue')
```



In [424... *# create dataframe grouped by body style and make to be used in the graphs*

```
car_type_make_df = automobiles_df.groupby(
    ['body-style', 'make'], as_index=False)['Count'].sum()
```

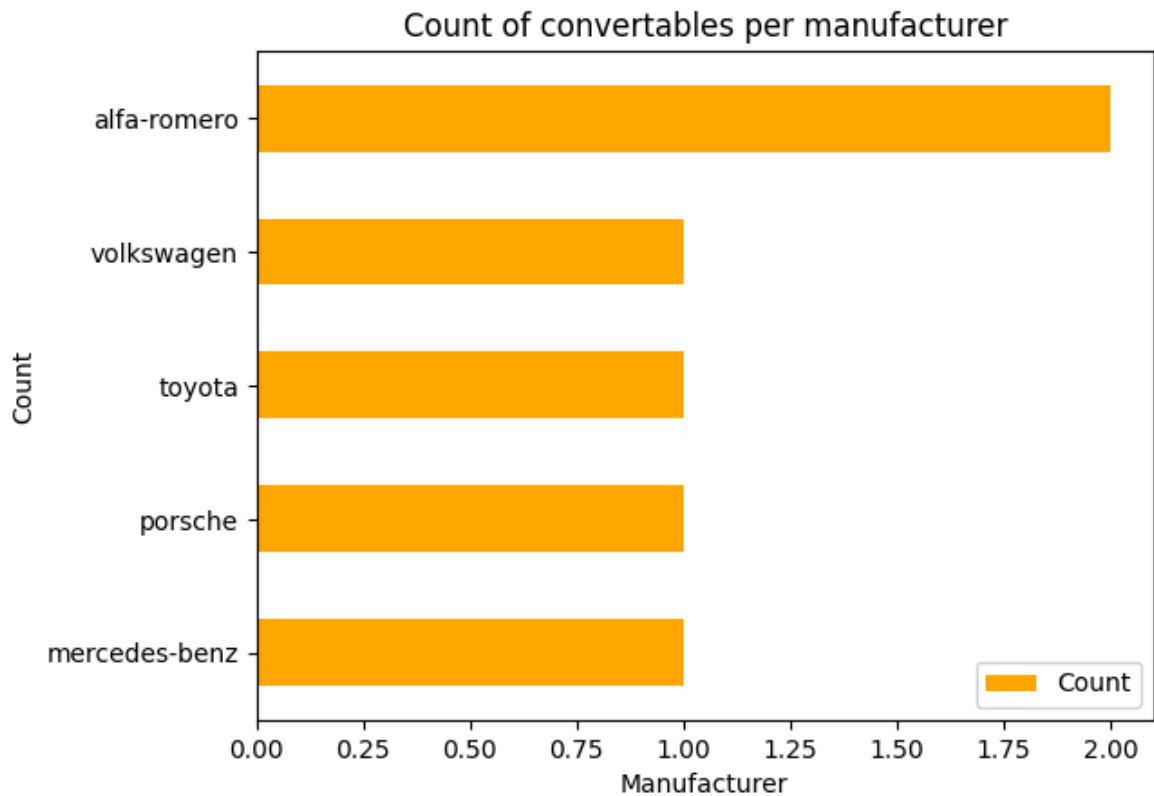
The bar chart below shows the number of convertables from each manufacturer. Two Alpha-Romeo convertables appear in the data while Toyota, Volkswagen, Porsche and Mercedes-Benze convertables only once in the data.

In [425... *# filter dataframe so that it only contains convertables*

```
convertable_and_make_df = car_type_make_df[
    car_type_make_df['body-style'] == 'convertible'
].drop('body-style', axis=1).sort_values(by='Count')

# create horizontal bar chart of number
# of convertables per manufactuter
ax = convertable_and_make_df.plot.barh(
    title='Count of convertables per manufacturer',
    x='make', y='Count', rot = 0, xlabel='Manufacturer',
    ylabel='Count', color='orange')

plt.show()
```



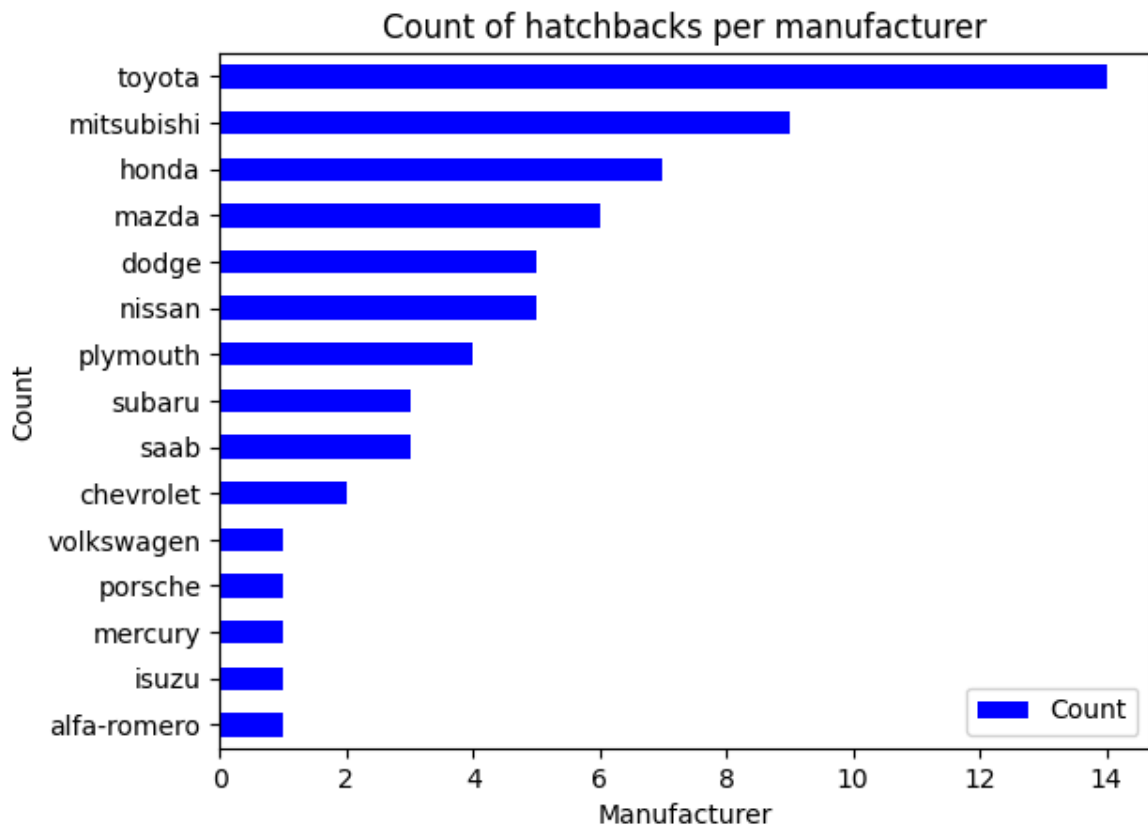
The graph below shows the count of hatchbacks per manufacturer in the dataset. Toyota has the most hatchbacks followed by Mazda, Mitsubishi then Honda. The expensive brands like Porsche and Alpha-Romeo only have 2 and 1 hatchback in the data respectively.

In [426...

```
# filter dataframe so that it only contains hatchbacks
convertable_and_make_df = car_type_make_df[
    car_type_make_df['body-style'] == 'hatchback'
].drop('body-style', axis=1).sort_values(by='Count')

# create horizontal bar chart of number
# of hatchbacks per manufacturer
ax = convertable_and_make_df.plot.barh(
    title='Count of hatchbacks per manufacturer', x='make', y='Count',
    rot = 0, xlabel='Manufacturer', ylabel='Count',
    color='blue')

plt.show()
```

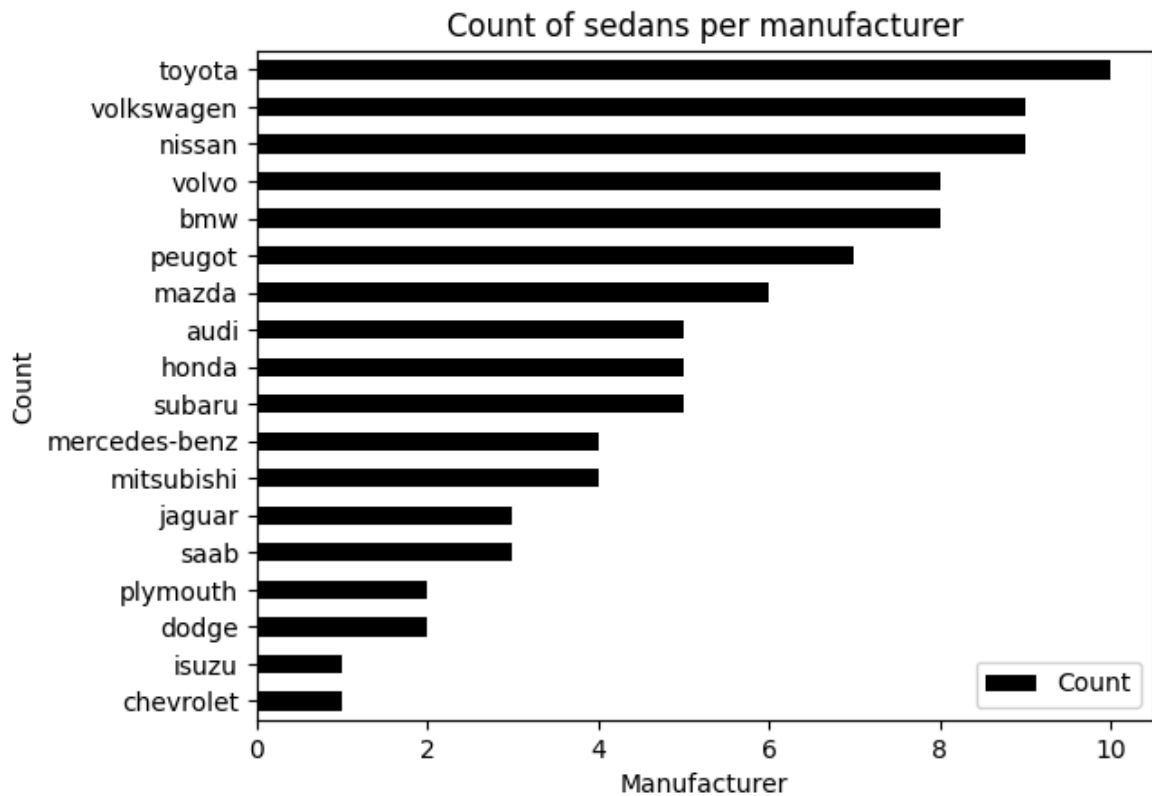


Toyota also has the highest number of sedans. This can be seen in the graph below. Porsche sedans do not appear in the data at all. Volkswagen and Nissan have the second highest number of sedans followed by Volvo and BMW. Only three Jaguar sedans, two Plymouth sedans and one Chevrolet sedan can be found in the data.

```
In [427... # filter dataframe so that it only contains sedans
convertable_and_make_df = car_type_make_df[
    car_type_make_df['body-style'] == 'sedan'
].drop('body-style', axis=1).sort_values(by='Count')

# create horizontal bar chart of number
# of sedans per manufacturer
ax = convertable_and_make_df.plot.barh(
    title='Count of sedans per manufacturer', x='make', y='Count',
    rot = 0, xlabel='Manufacturer', ylabel='Count',
    color='black')

plt.show()
```

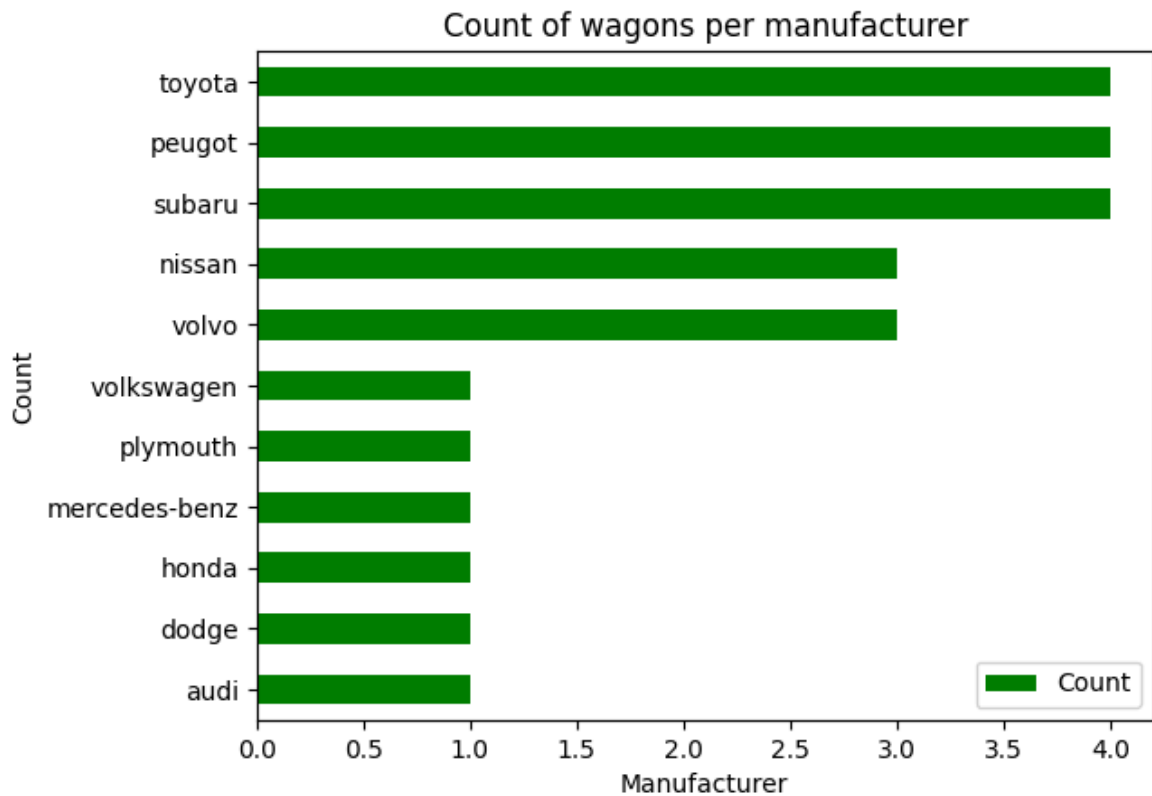


According to the bar chart below, 4 Subaru, 4 Toyota and 4 Peugeot wagons appear in the dataset. 3 Nissan and 3 Volvo wagons appear in the dataset. There are no Jaguar or Porsche wagons. This may be because Jaguar and Porsche do not manufacture wagons.

```
In [428... # filter dataframe so that it only contains wagons
convertable_and_make_df = car_type_make_df[
    car_type_make_df['body-style'] == 'wagon'
].drop('body-style', axis=1).sort_values(by='Count')

# create horizontal bar chart of number
# of wagons per manufactuter
ax = convertable_and_make_df.plot.barh(
    title='Count of wagons per manufacturer', x='make', y='Count',
    rot = 0, xlabel='Manufacturer', ylabel='Count',
    color='green')

plt.show()
```

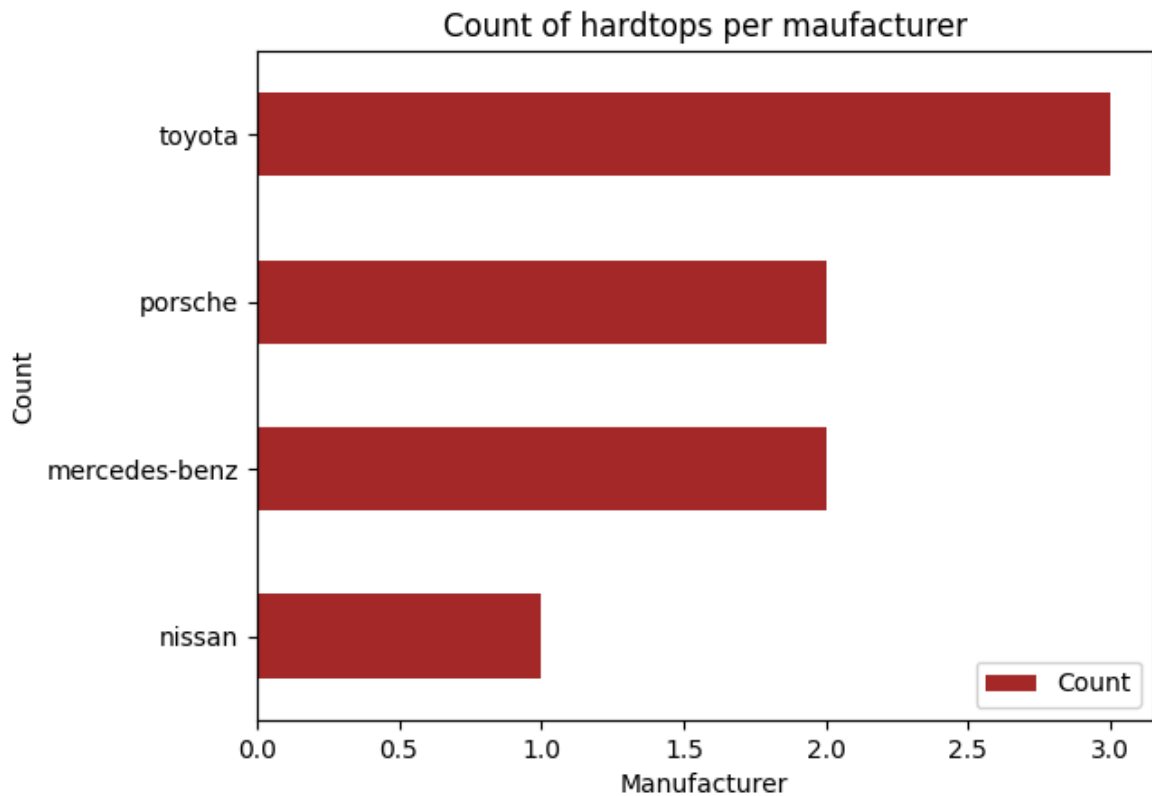


Based on the bar chart below, Toyota manufactured three out of the eight hardtops in the dataset. Porsche and Mercedes-Benz each manufactured two of the eight hardtops. Nissan manufactured the remaining hardtop.

```
In [429... # filter dataframe so that it only contains hardtops
convertable_and_make_df = car_type_make_df[
    car_type_make_df['body-style'] == 'hardtop'
].drop('body-style', axis=1).sort_values(by='Count')

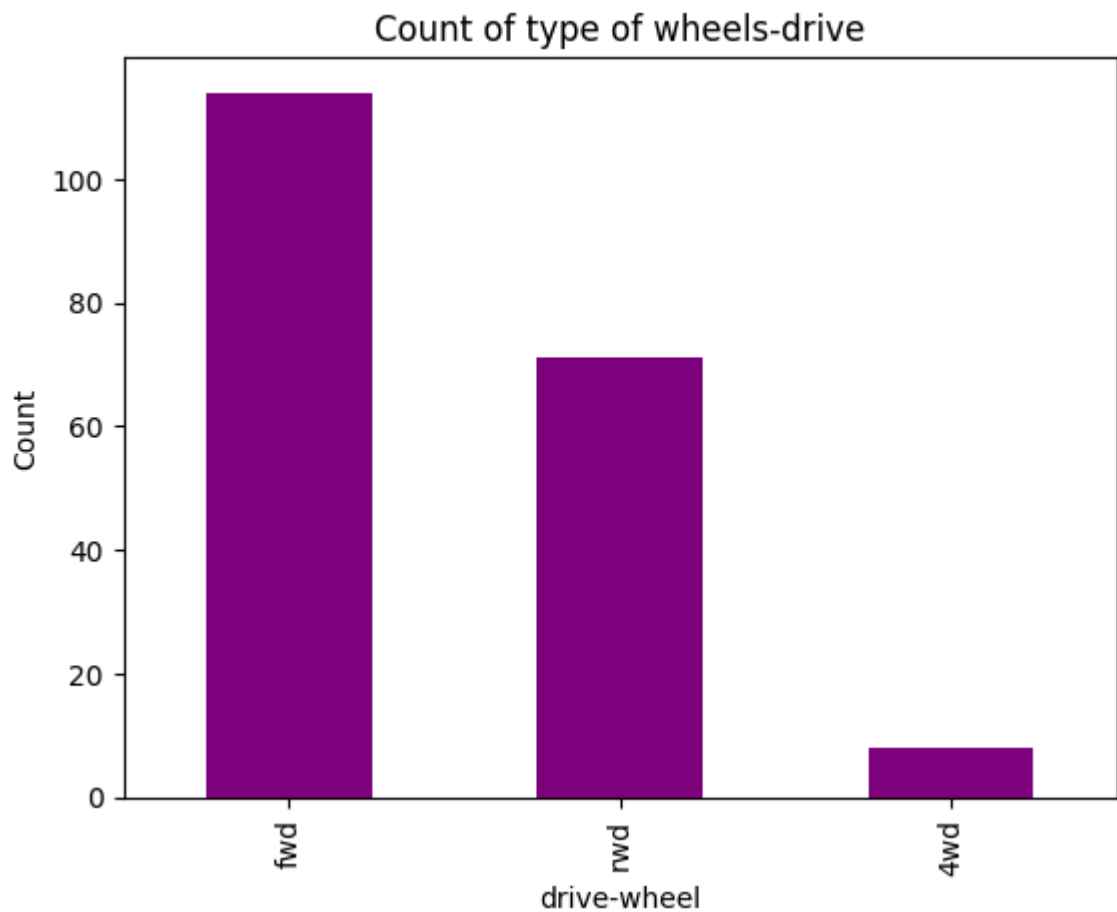
# create horizontal bar chart of number
# of hardtops per manufacturer
ax = convertable_and_make_df.plot.barh(
    title='Count of hardtops per manufacturer', x='make', y='Count',
    rot = 0, xlabel='Manufacturer', ylabel='Count',
    color='brown')

plt.show()
```



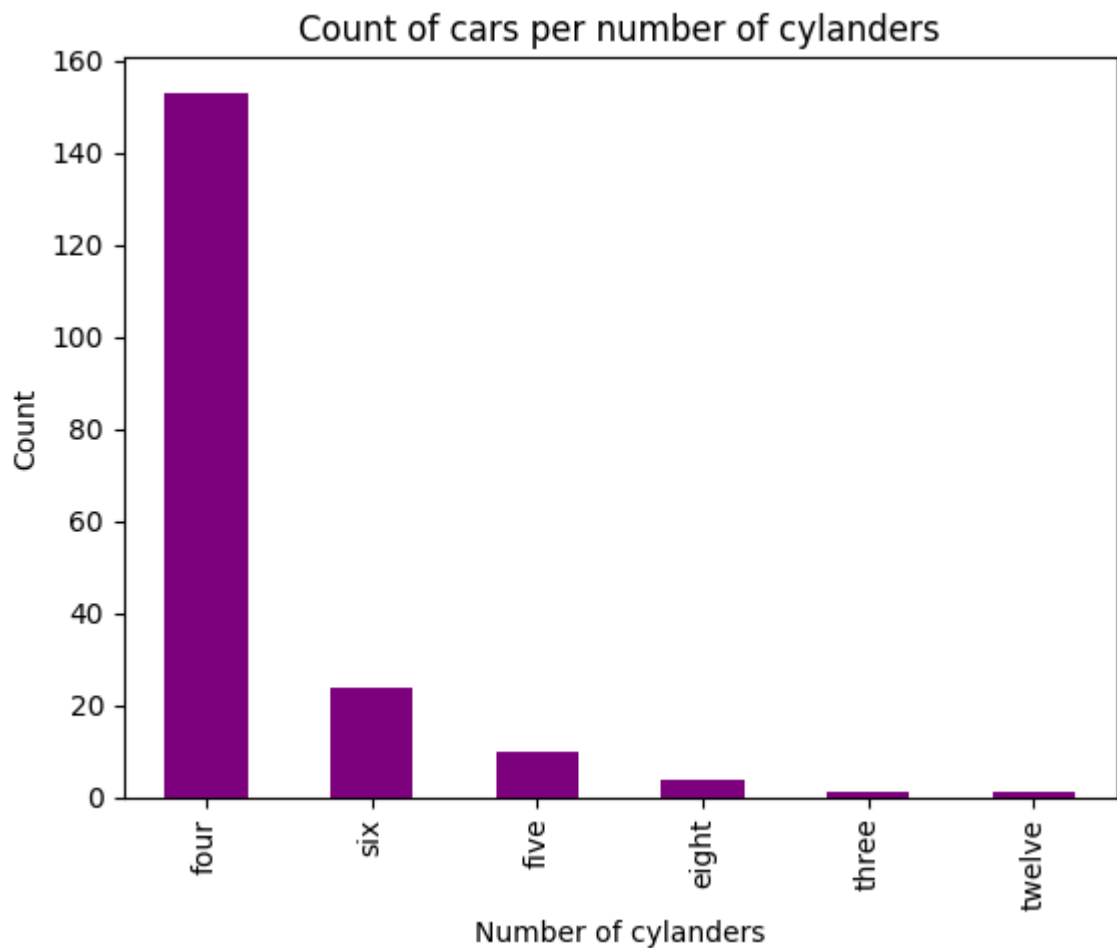
The graph below shows the number of cars in the dataset that have a front-wheel drive, rear-wheel drive and four-wheel drive. The graph shows that the majority of cars have front-wheel drive. The minority of cars are four-wheel drive cars.

```
In [ ]: wheel_drive_df = automobiles_df.groupby(  
        'drive-wheels')['Count'].sum().sort_values(ascending=False)  
ax = wheel_drive_df.plot.bar(  
    title='Count of type of wheel-drive',  
    rot = 90, xlabel='drive-wheel', ylabel='Count',  
    color='purple')
```



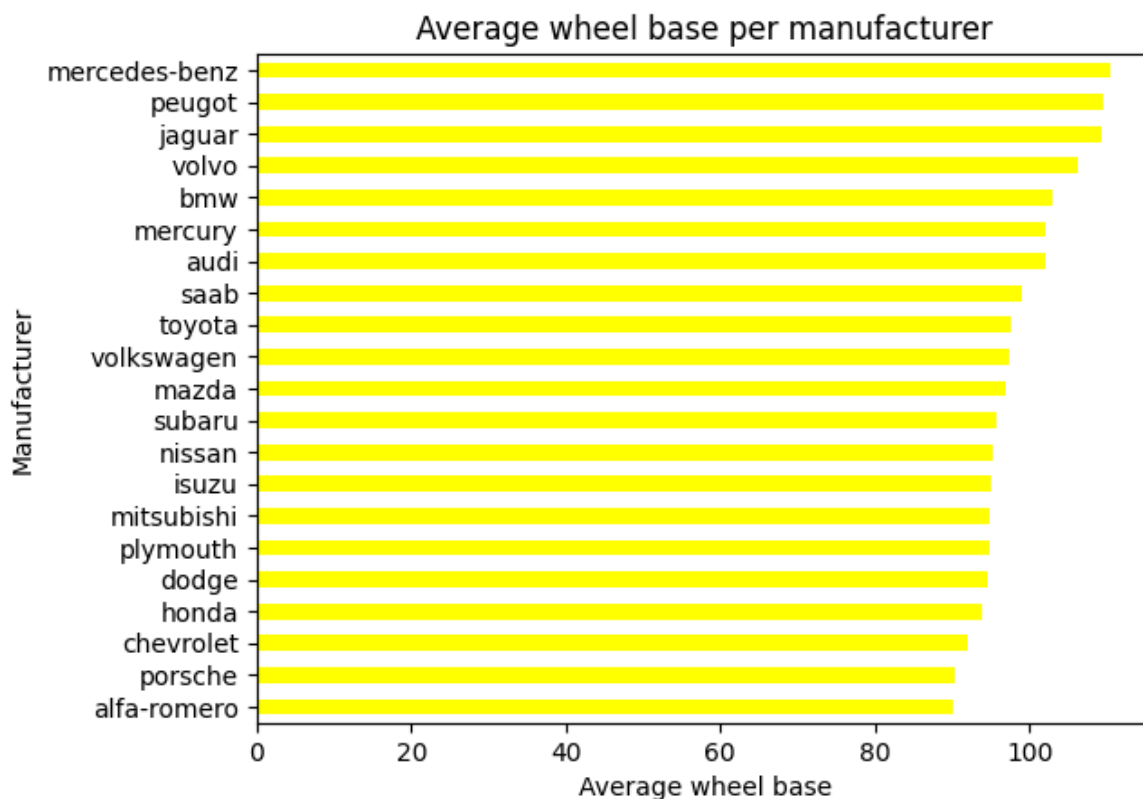
The bar chart below shows the number of cars with either two, three, four, five, six, eight or twelve cylinders. As seen below, most cars have four cylinders, followed by six cylinders, then five, then eight cylinders. It is possible that cars with a high number of cylinders are more expensive. Only one car, a Jaguar sedan, has twelve cylinders. Four of the five cars with eight cylinders are manufactured by Mercedes-Benz and one car with eight cylinders are manufactured by Porsche.

```
In [431... cylanders_df = automobiles_df.groupby(
    'num-of-cylinders')['Count'].sum().sort_values(ascending=False)
ax = cylanders_df.plot.bar(
    title='Count of cars per number of cylanders',
    rot = 90, xlabel='Number of cylanders', ylabel='Count',
    color='purple')
```



The bar chart below shows the average (over all models) wheel base per car Manufacturer. Mercedes-Benz has the highest wheel base, followed closely by Peugeot, Jaguar and Volvo. The Alfa-Romeo has the smallest average wheel base, followed by Porsche and Chevrolet.

```
In [432... wheel_base_df = automobiles_df.groupby(  
    'make')['wheel-base'].mean().sort_values()  
ax = wheel_base_df.plot.barh(  
    title='Average wheel base per manufacturer',  
    rot = 0, xlabel='Average wheel base', ylabel='Manufacturer',  
    color='yellow')
```



Which are the 5 most expensive cars?

In [433...

```
# create dataframes of the 5 most expensive and add 'label' column
expensive_cars_df = automobiles_df.nlargest(5, 'price')
expensive_cars_df['label'] = 'Expensive'

# create dataframe of the 5 cheapest cars and add 'label' column
cheapest_cars_df = automobiles_df.nsmallest(5, 'price')
cheapest_cars_df['label'] = 'Cheaper'

# combine 'expensive_cars_df' and 'cheapest_cars_df'
combined_df = pd.concat([expensive_cars_df, cheapest_cars_df])
```

In [434...

```
# create bar charts to compare the cheapest and most expensive cars

# create car_compare dataframe
car_compare = combined_df.groupby('label', as_index=False)[
    ['city-mpg', 'highway-mpg', 'price']].mean()

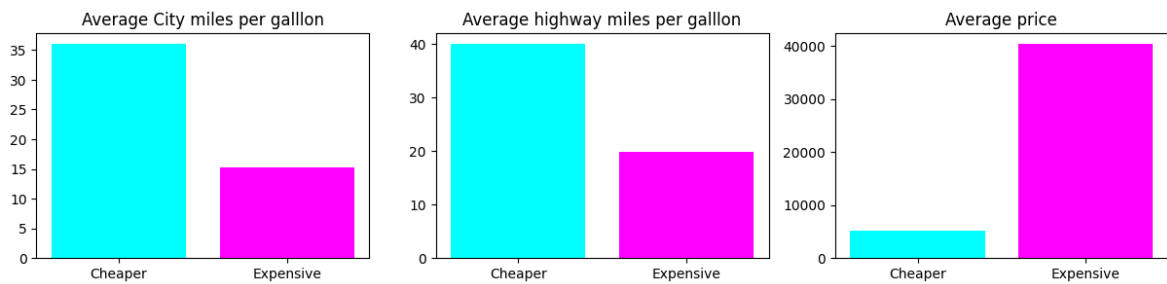
# specify 3 subplots for size 15 by 3
fig, axes = plt.subplots(nrows=1, ncols=3, figsize=(15, 3))

# plot bar chart of average city miles per
# gallon for expensive and cheap cars
axes[0].bar(car_compare['label'], car_compare[
    'city-mpg'], color=["Cyan", "Magenta"])
axes[0].set_title('Average City miles per gallon')

# plot bar chart of average highway miles per
# gallon for expensive and cheap cars
axes[1].bar(car_compare['label'], car_compare[
    'highway-mpg'], color=["Cyan", "Magenta"])
axes[1].set_title('Average highway miles per gallon')
```

```
# plot bar chart of average price for expensive and cheap cars
axes[2].bar(car_compare['label'], car_compare[
    'price'], color=["Cyan", "Magenta"])
axes[2].set_title('Average price')

plt.show()
```



The graphs above show the average miles per gallon in the city and on the highway as well as the average price for the five cheapest and five most expensive cars. The miles per gallon is an indication of the fuel efficiency of a car.

The cheaper cars have a higher average miles per gallon in the city and on the highway compared to the average of the 5 most expensive cars. More specifically, the average miles per gallon for the cheaper cars are almost double the average miles per gallon for the expensive cars. The cheaper cars are therefore more fuel efficient than the 5 most expensive cars. The third graph shows that the average price of the 5 most expensive cars are significantly more expensive than the cheaper cars. The expensive cars therefore are not worth the money spent.

```
In [435... # plot number of car models on a bar chart
combined_check_df = combined_df.groupby(
    ['label', 'make', 'body-style', 'fuel-type',
     'num-of-doors', 'city-mpg', 'highway-mpg'],
    as_index=False)['engine-size'].mean()

print(combined_check_df)
```

	label	make	body-style	fuel-type	num-of-doors	city-mpg	\
0	Cheaper	chevrolet	hatchback	gas	two	47	
1	Cheaper	mazda	hatchback	gas	two	30	
2	Cheaper	mitsubishi	hatchback	gas	two	37	
3	Cheaper	subaru	hatchback	gas	two	31	
4	Cheaper	toyota	hatchback	gas	two	35	
5	Expensive	bmw	sedan	gas	four	15	
6	Expensive	bmw	sedan	gas	two	16	
7	Expensive	mercedes-benz	hardtop	gas	two	14	
8	Expensive	mercedes-benz	sedan	gas	four	14	
9	Expensive	porsche	convertible	gas	two	17	

	highway-mpg	engine-size
0	53	61
1	31	91
2	41	92
3	36	97
4	39	92
5	20	209
6	22	209
7	16	304
8	16	308
9	25	194

The table above includes the make, body-style, fuel-type, num-of-doors, city-mpg, highway-mpg and engine-size of the five cheapest and five most expensive cars. BMW, Mercedes-benz and Porsche manufacture the most expensive cars. While the most expensive cars are made by only three manufacturers, there is a wider variety of manufacturers amongst cheaper cars. These include Chevrolet, Mazda, Mitsubishi, Subaru and Toyota. All the cheapest cars are hatchbacks. Both the expensive and cheapest cars use gas instead of diesel. It is interesting to note that Mercedes-Benz cars on this list have the lowest miles per gallon in the city and on the highway, and is therefore the least fuel efficient car compared to the other cars on this list. Only two of the cars on the list have four doors. These are the BMW and Mercedes-Benz. All the cheaper cars have two doors.

Which manufacturer builds the most fuel efficient vehicles?

```
In [436... # create horizontal bar charts of the average miles
# per gallon for each car manufacturer

# create dataframe of the average fuel efficiency
# (miles per gallon) in the city per car manufacturer
car_manufact_city_mpg = automobiles_df.groupby(
    'make', as_index=False)[['city-mpg']].mean().sort_values(
        by='city-mpg')

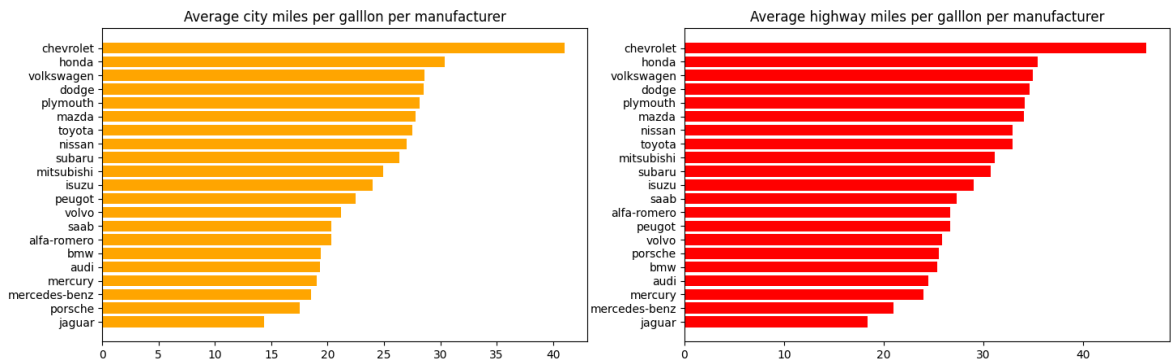
# create dataframe of the average fuel efficiency
# (miles per gallon) on the highway per car manufacturer
car_manufact_highway_mpg = automobiles_df.groupby(
    'make', as_index=False)[['highway-mpg']].mean().sort_values(
        by='highway-mpg')

# specify the number and size of subplots
fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(17, 5))
```

```
# create horizontal bar chart of average miles
# per gallon used in the city per car manufacturer
axes[0].barh(car_manufact_city_mpg['make'],
              car_manufact_city_mpg['city-mpg'], color='orange')
axes[0].set_title('Average city miles per gallon per manufacturer')

# create horizontal bar chart of average miles
# per gallon used on the highway per car manufacturer
axes[1].barh(car_manufact_highway_mpg['make'],
              car_manufact_highway_mpg['highway-mpg'], color='red')
axes[1].set_title('Average highway miles per gallon per manufacturer')

plt.show()
```



The graphs above shows the average miles per gallon per manufacturer in the city (left) and on the highway (right).

Chevrolet cars have the highest average miles per gallon on the highway and in the city. Chevrolet therefore manufactures the most fuel efficient cars. The second most fuel efficient car manufacturer is Honda, followed by Volkswagen, Dodge, Plymouth and Mazda.

It is interesting to note that, on average, Nissan is more fuel efficient on the highway than Toyota, while Toyota is more fuel efficient than Nissan in the city. The same applies to Mitsubishi and Subaru.

Jaguar manufactures the least fuel efficient cars for driving on the highway and in the city. While, on average, Porsche makes the second least fuel efficient cars for driving in the city, its fuel efficiency improves significantly on the highway.

Which vehicles have the largest engine capacity.

In [437...

```
# sorted automobile cars by engine size in descending order
sorted_automobiles_df = automobiles_df.sort_values(
    by='engine-size', ascending=False)
sorted_automobiles_df = sorted_automobiles_df[
    ['make', 'num-of-doors', 'body-style', 'engine-size',
     'num-of-cylinders', 'horsepower']]

sorted_automobiles_df.head()
```

Out[437...

	make	num-of-doors	body-style	engine-size	num-of-cylinders	horsepower
49	jaguar	two	sedan	326	twelve	262
73	mercedes-benz	four	sedan	308	eight	184
74	mercedes-benz	two	hardtop	304	eight	184
47	jaguar	four	sedan	258	six	176
48	jaguar	four	sedan	258	six	176

The table above shows the 5 cars with the largest engine size in the dataset.

Three of the five cars with the largest engine capacity is manufactured by Jaguar. However, as seen from the graphs above, the Jaguar is the least fuel efficient car. Two Mercedes-Benz cars have the second and third largest engine size. Four of the cars are Sedans while one Mercedes-Benz has a hard-top.

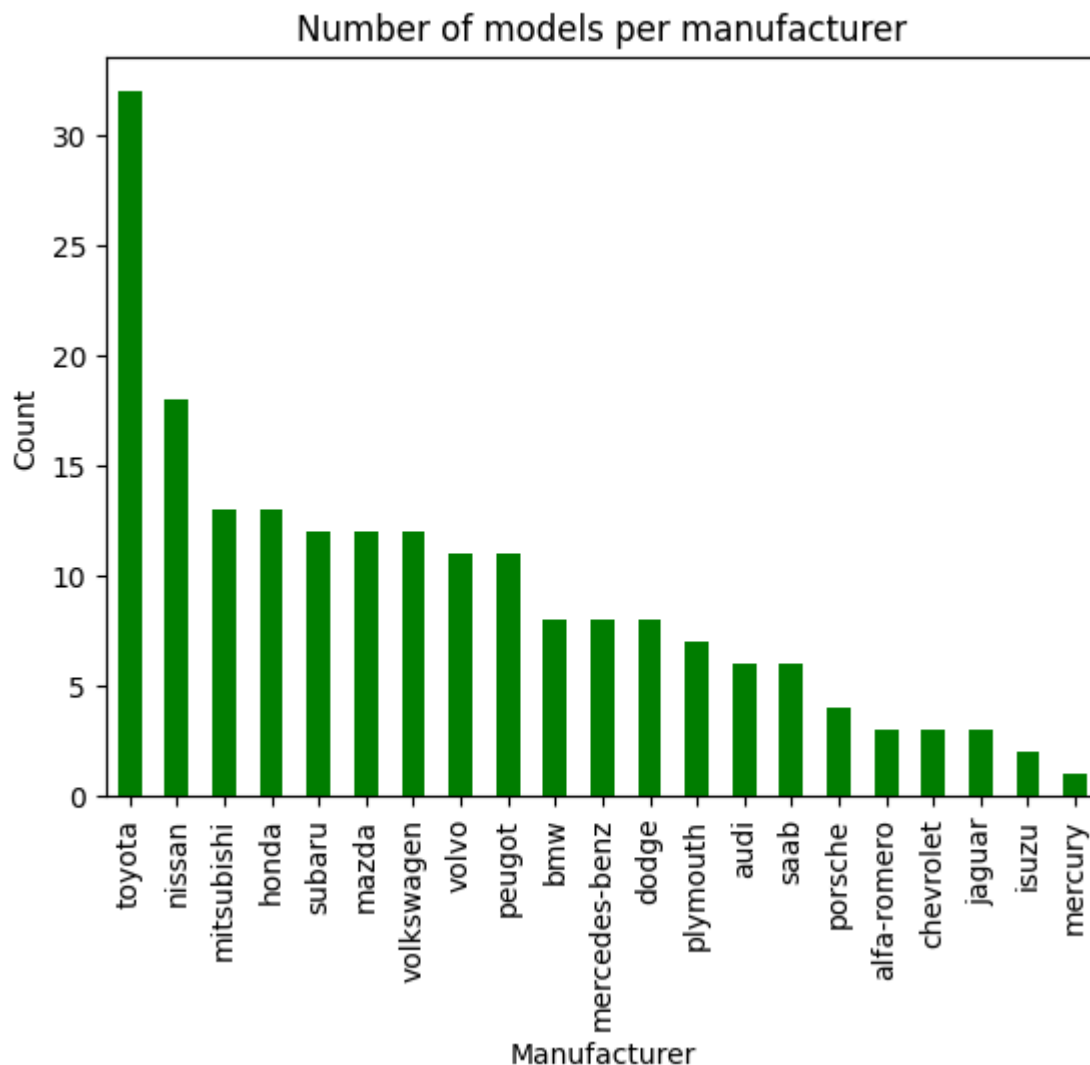
Which vehicle manufacturer has the most car models in the dataset

In [438...

```
# group automobile_df dataframe into car manufacturer and include count
model_count_df = automobiles_df.groupby(
    'make', as_index=True)['Count'].sum()
model_count_df = model_count_df.sort_values(ascending=False)

# plot number of car models on a bar chart
ax = model_count_df.plot.bar(
    title='Number of models per manufacturer',
    rot = 90, xlabel='Manufacturer',
    ylabel='Count', color='green')

plt.show()
```



Toyota cars appear most often in the dataset, followed by the Nissan. This is expected since the graphs above indicate that Toyota manufactures the highest number of sedans, hatchbacks and wagons and hard-tops. Mitsubishi and Honda have manufactured a similar number of cars. Subaru, Mazda and Volkswagen also have a similar number of cars. The more expensive cars like the Jaguar and the Porsche appear less often. The most fuel efficient car, the Chevrolet only appears 3 time in the dataset.

Conclusion

Toyota manufactures the most cars in the dataset. In fact, Toyota has manufactured the highest number of hatchbacks, sedans, wagons and hard-tops. Mercury is the least popular manufacturer.

Sedans are the most popular car-type in the dataset. This makes sense since it is the most family friendly car-type in the dataset. Hatchbacks are the second most popular car, followed by wagons. Hardtops and convertable are the least popular car types. Most cars have four cylanders and only the more expensive cars (porsche and Mercedes-Benz) have the 8 or twelve cylanders. The majority of cars are front-wheel drive and only a small number of cars are four-wheel drive.

Mercedes-Benz Peugeot and Jaguar have the largest wheel base. It is interesting to note that Jaguar also has the largest engine and the most cylinders.

The graphs above show that expensive cars are not very fuel efficient as the cheaper cars have more miles per gallon in the city and on the highway. Based on the fuel spend, expensive cars are not worth their price-tag.

The report was written by Charné Munjeri