Seaborn

What Is Seaborn in Python?

Seaborn is one of an amazing library for visualization of the graphical statistical plotting in Python. Seaborn provides many color palettes and defaults beautiful styles to make the creation of many statistical plots in Python more attractive.

In the seaborn library, the plot that we create is divided into the following various categories:

Distribution plots: This type of plot is used for examining both types of distributions, i.e., univariate and bivariate distribution. Relational plots: This type of plot is used to understand the relation between the two given variables. Regression plots: Regression plots in the seaborn library are primarily intended to add an additional visual guide that will help to emphasize dataset patterns during the analysis of exploratory data. Categorical plots: The categorical plots are used to deals with categories of variables and how we can visualize them. Multi-plot grids: The multi-plot grids are also a type of plot that is a useful approach is to draw multiple instances for the same plot with different subsets of a single dataset. Matrix plots: The matrix plots are a type of arrays of the scatterplots.

Installation of seaborn library for Python

pip install seaborn

Required dependencies or prerequisites for the seaborn library:

Python installed with the latest version (3.6+). Numpy must be installed with version 1.13.3 or higher. SciPy must be installed with 1.0.1 or higher versions. Must have panda library with 0.22.0 or higher versions. statsmodel library must be installed with version 0.8.0 or higher. And should have matplotlib installed with 2.1.2 or higher versions.

Plotting Chart Using seaborn Library

In [1]:
```
# Importing Libraries in Jupyter Notebook
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [2]:
```
#Loading Dataset
mtcars=pd.read_csv('mtcars.csv')
```

In [3]:
```
# Featching first 5 rows
mtcars.head()
```

Out[3]:

	model	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
0	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
1	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
2	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
3	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
4	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2

Full form of all the columns

model -Car Model mpg - miles per gallon cyl -Cylinder disp hp drat wt qsec vs am gear - Gear carb - Carburetors

In [8]:
```
# use the info() function to print the summary of the data frame.
#It returns information regarding the index dtype, column dtypes, non-null values,
mtcars.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32 entries, 0 to 31
Data columns (total 12 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   model   32 non-null     object
 1   mpg     32 non-null     float64
 2   cyl     32 non-null     int64
 3   disp    32 non-null     float64
 4   hp      32 non-null     int64
 5   drat    32 non-null     float64
 6   wt      32 non-null     float64
 7   qsec    32 non-null     float64
 8   vs      32 non-null     int64
 9   am      32 non-null     int64
 10  gear    32 non-null     int64
 11  carb    32 non-null     int64
dtypes: float64(5), int64(6), object(1)
memory usage: 3.1+ KB
```

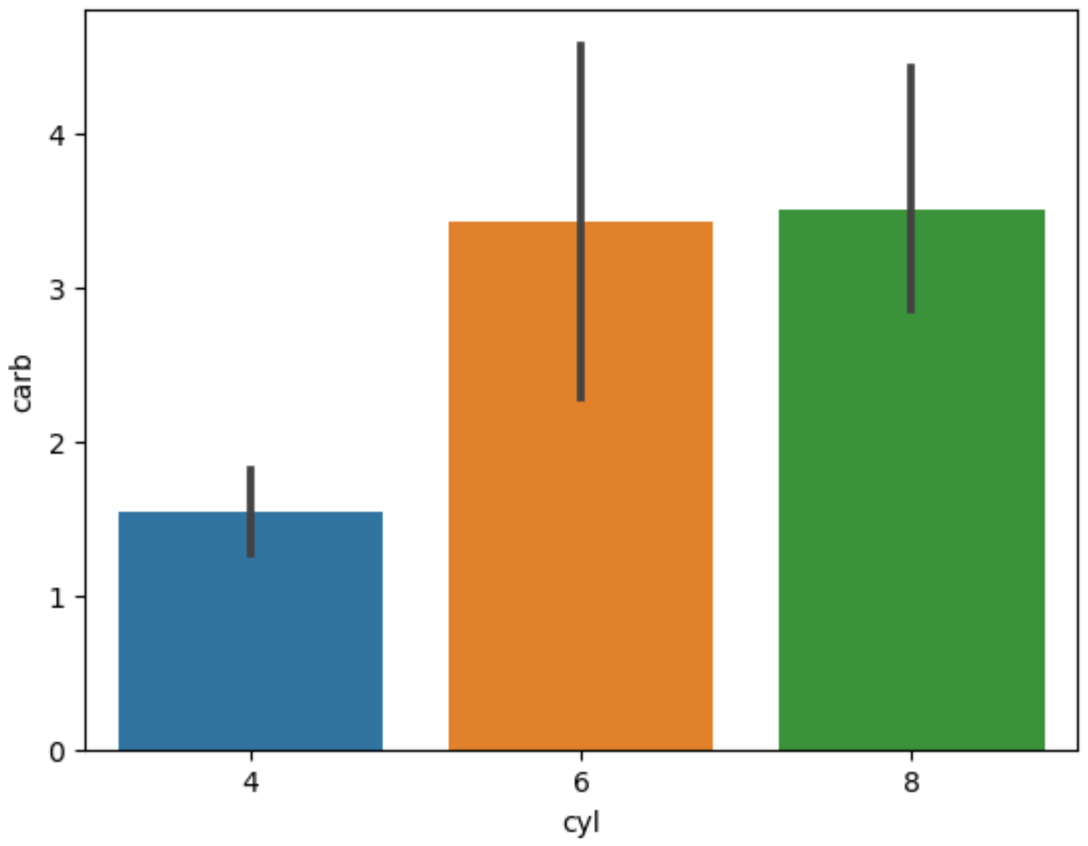
In [10]:
```
# check the shape of the mtcars dataframe.
mtcars.shape
```

Out[10]:
```
(32, 12)
```

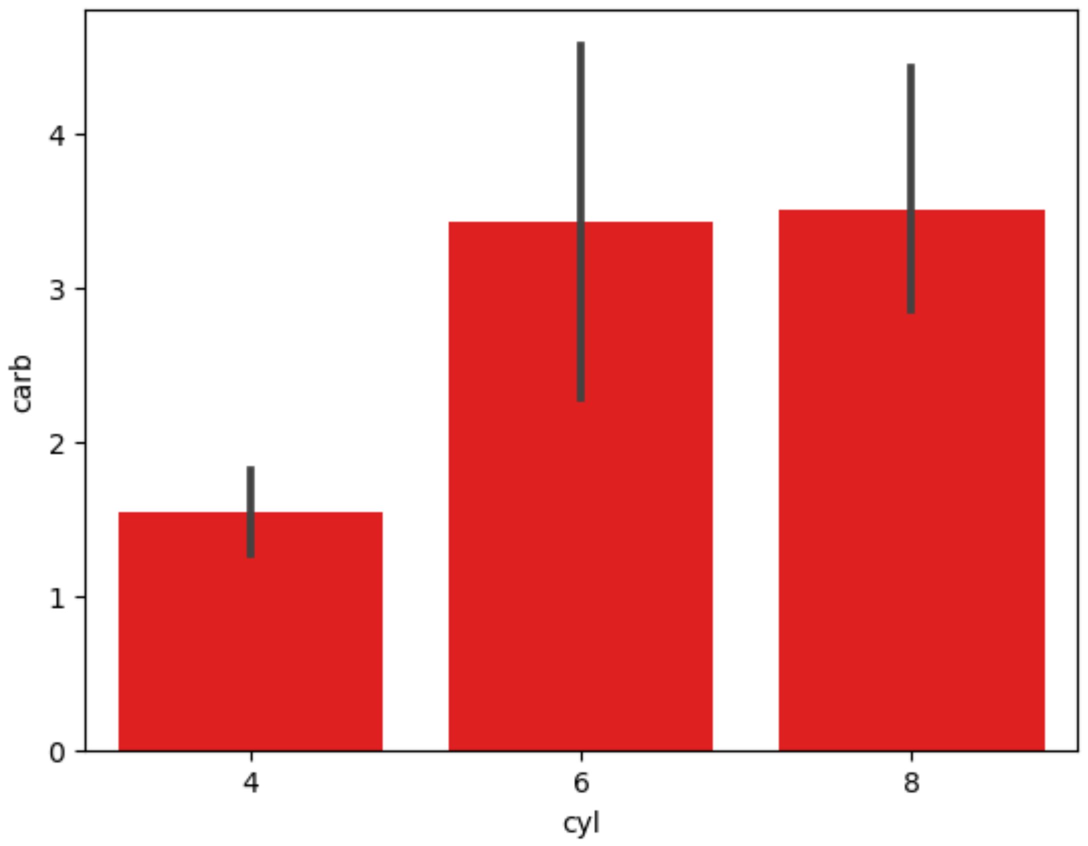
Python Seaborn Plotting Functions

Barplot A bar plot gives an estimate of the central tendency for a numeric variable with the height of each rectangle. It provides some indication of the uncertainty around that estimate using error bars. To build this plot, you usually choose a categorical column on the x-axis and a numerical column on the y-axis.

In [20]:
```
res = sns.barplot(mtcars, x='cyl', y='carb')
plt.show()
```

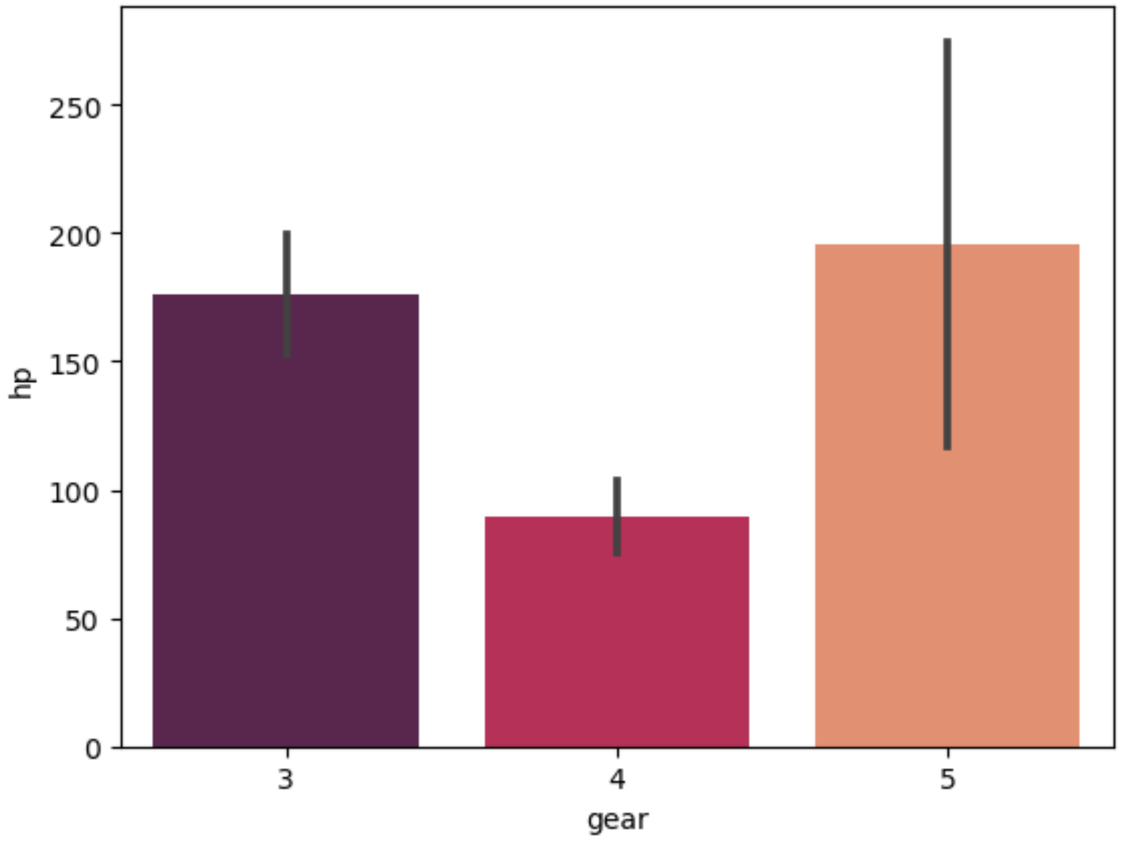


```
In [23]: #Assign Color to the bars
         res=sns.barplot(mtcars, x='cyl', y='carb', color='red')
```



Seaborn library also has the palette attribute which you can use to give different colors to the bars.

In [25]:
```python
res=sns.barplot(mtcars, x='gear', y='hp', palette='rocket')
```

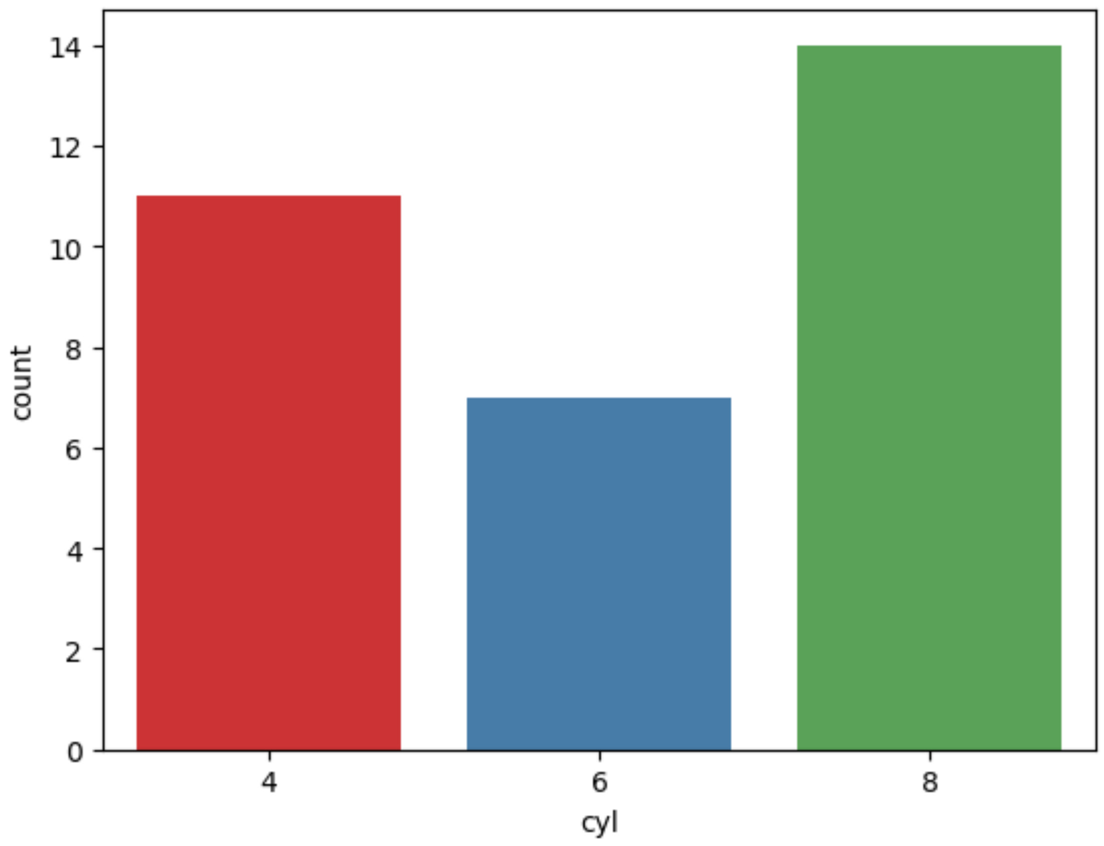


Countplot The countplot() function in the Python Seaborn library returns the count of total values for each category using bars.

The below count plot returns the number of vehicles for each category of cylinders.

In [26]:
```python
sns.countplot(mtcars, x='cyl', palette='Set1' )
```

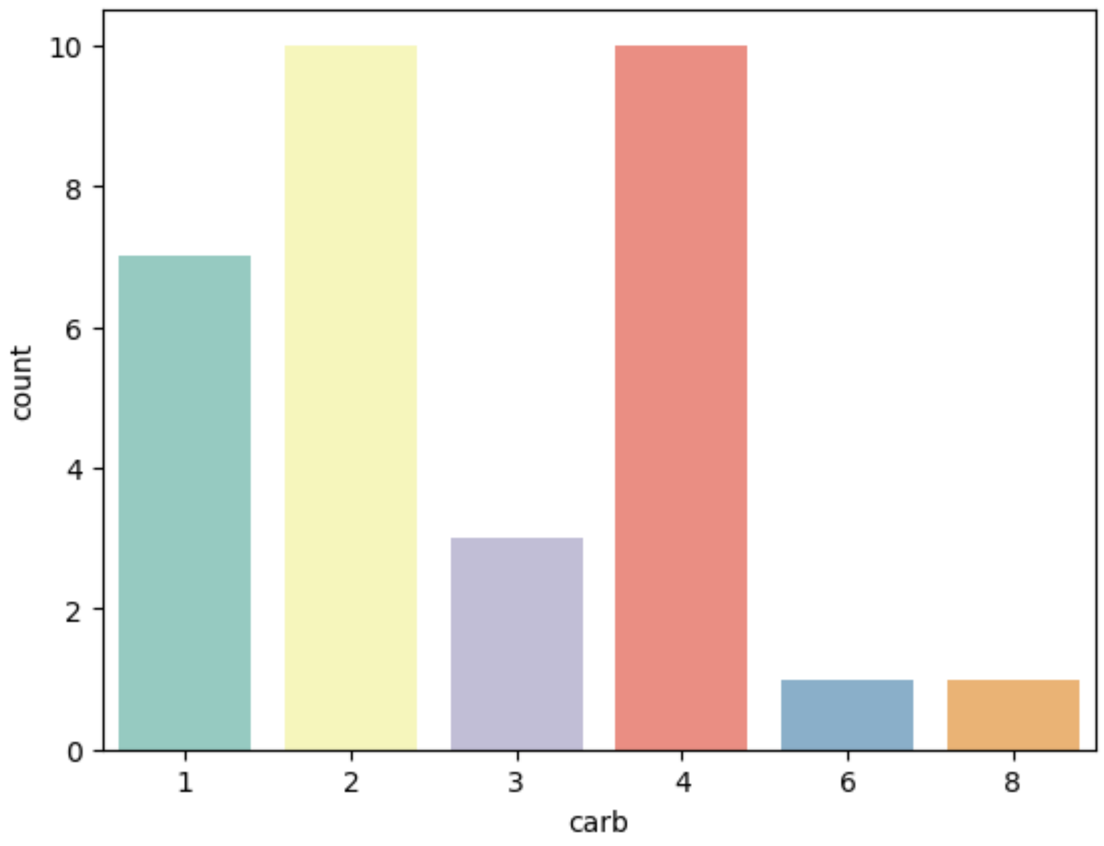
Out[26]:
```
<Axes: xlabel='cyl', ylabel='count'>
```



The next count plot shows the number of cars for each carburetor.

```
In [27]: sns.countplot(mtcars, x='carb', palette='Set3' )
```

```
Out[27]: <Axes: xlabel='carb', ylabel='count'>
```

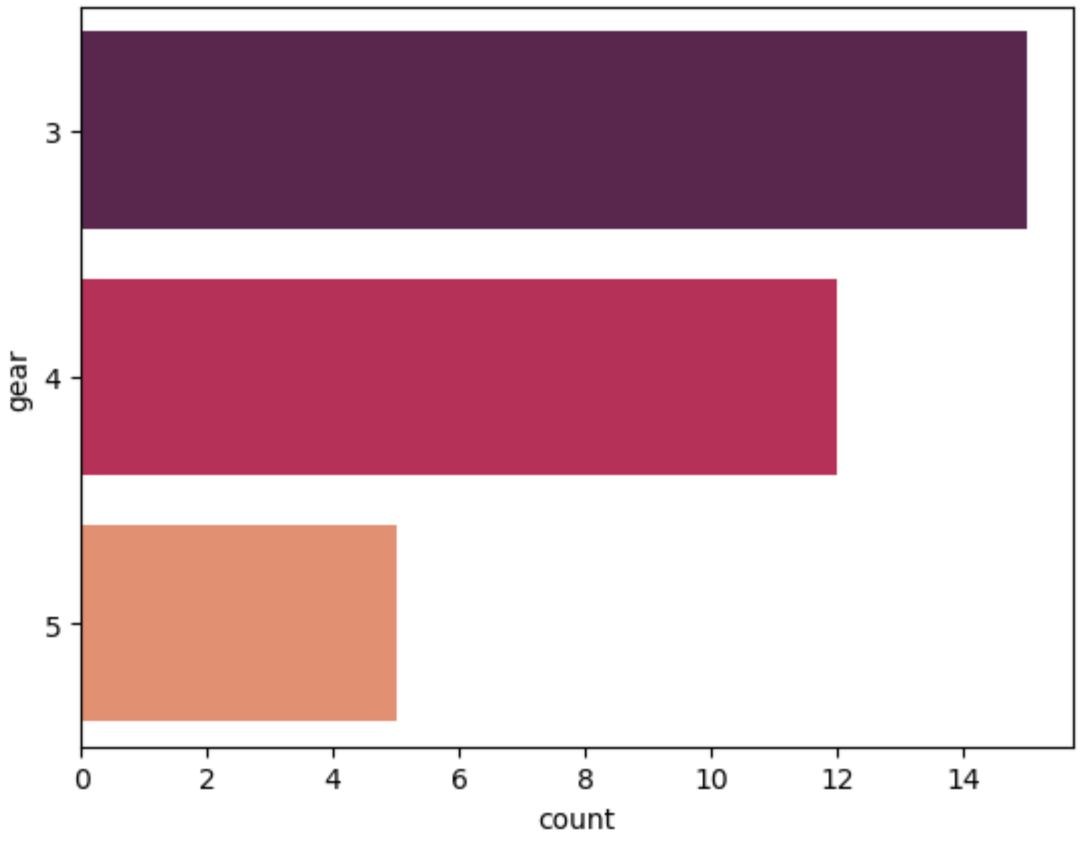


Python Seaborn allows you to create horizontal count plots where the feature column is in the y-axis and the count is on the x-axis.

The below visualization shows the count of cars for each category of gear.

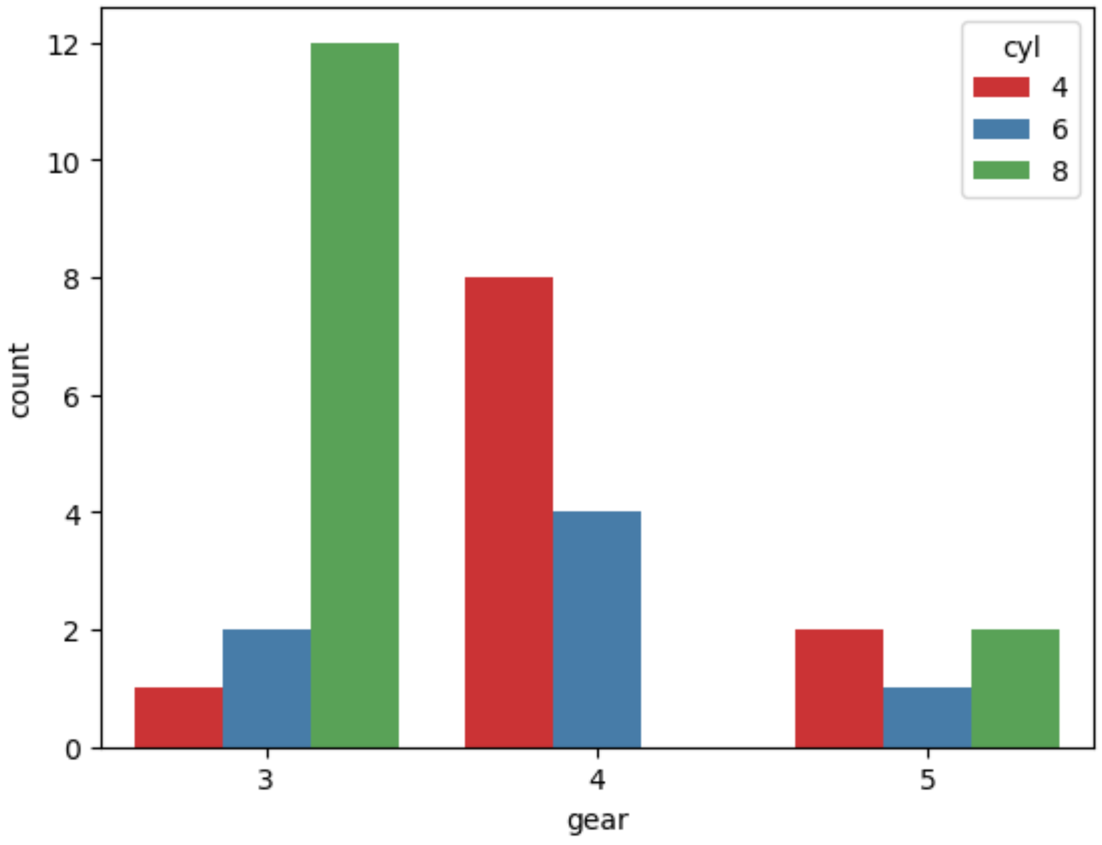
```
In [28]: sns.countplot(mtcars,y='gear',palette='rocket')
```

```
Out[28]: <Axes: xlabel='count', ylabel='gear'>
```



From the above plot, you can see that we have 15 vehicles with 3 gears, 12 vehicles with 4 gears, and 5 vehicles with 5 gears.

```
In [31]: #In the below count plot, you have the count of cars for each category of gears tha

         sns.countplot(mtcars, x='gear', hue='cyl', palette="Set1")

Out[31]: <Axes: xlabel='gear', ylabel='count'>
```

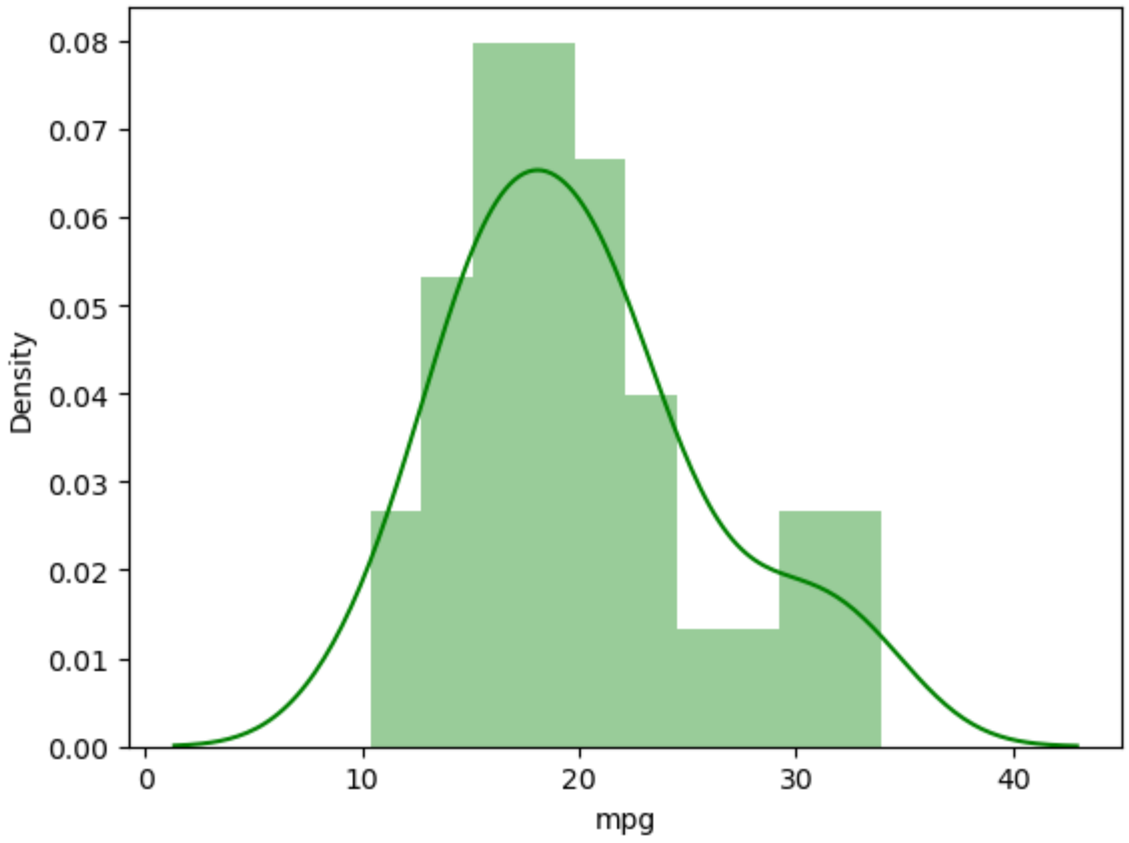


Distribution Plot: The Seaborn library supports the distplot() function that creates the distribution of any continuous data.

In [37]:
```
#mpg metrics measure the total distance the car can travel per gallon of fuel.
sns.distplot(mtcars.mpg, bins=10,color='g')
```

```
C:\Users\malam\AppData\Local\Temp\ipykernel_4560\118028255.py:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(mtcars.mpg, bins=10,color='g')
```

Out[37]:
```
<Axes: xlabel='mpg', ylabel='Density'>
```



Heatmap

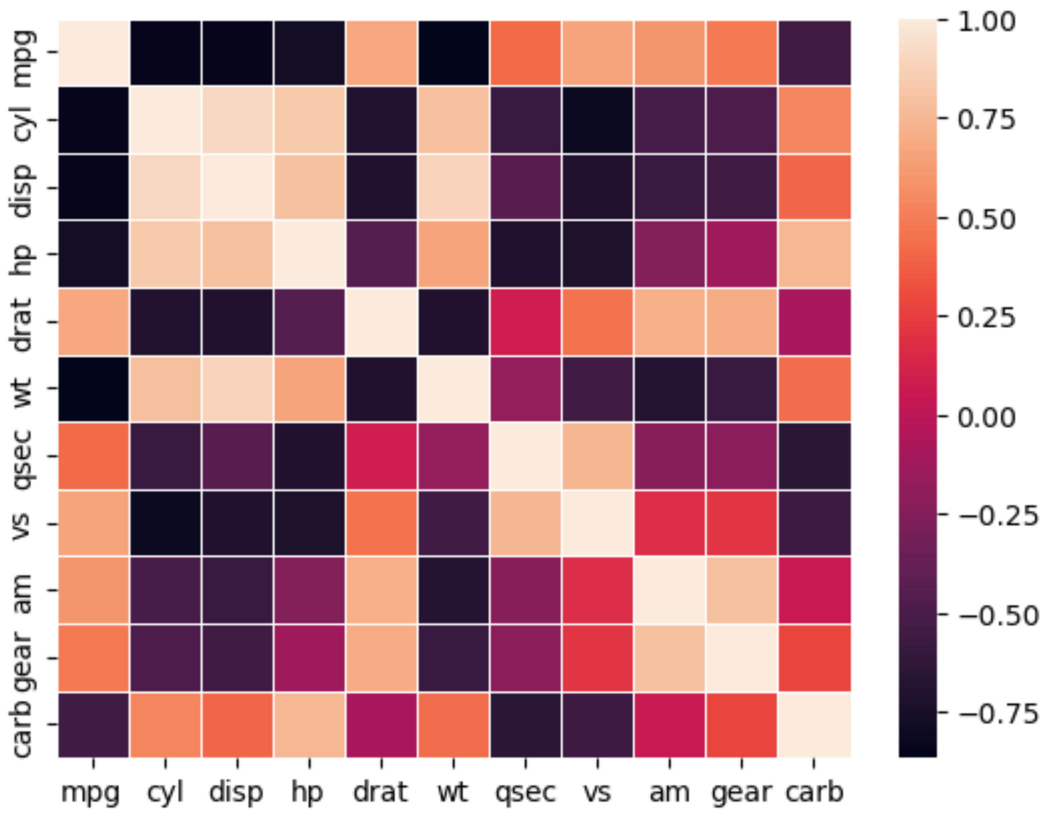
Heatmaps in the Seaborn library lets you visualize matrix-like data. The values of the variables are contained in a matrix and are represented as colors.

Below is an example of the heatmap where you are finding the correlation between each variable in the mtcars dataset.

In [40]:
```
#finding the correlation between each variable in the mtcars dataset.
sns.heatmap(mtcars.corr(), cbar=True, linewidth=0.5)
```

```
C:\Users\malam\AppData\Local\Temp\ipykernel_4560\4150125478.py:2: FutureWarning: Th
e default value of numeric_only in DataFrame.corr is deprecated. In a future versio
n, it will default to False. Select only valid columns or specify the value of nume
ric_only to silence this warning.
  sns.heatmap(mtcars.corr(), cbar=True, linewidth=0.5)
```

Out[40]:
```
<Axes: >
```



Scatterplot

The Seaborn scatterplot() function helps you create plots that can draw relationships between two continuous variables.

In [42]:
```
iris=sns.load_dataset('iris')
```

In [43]:
```
iris.head()
```

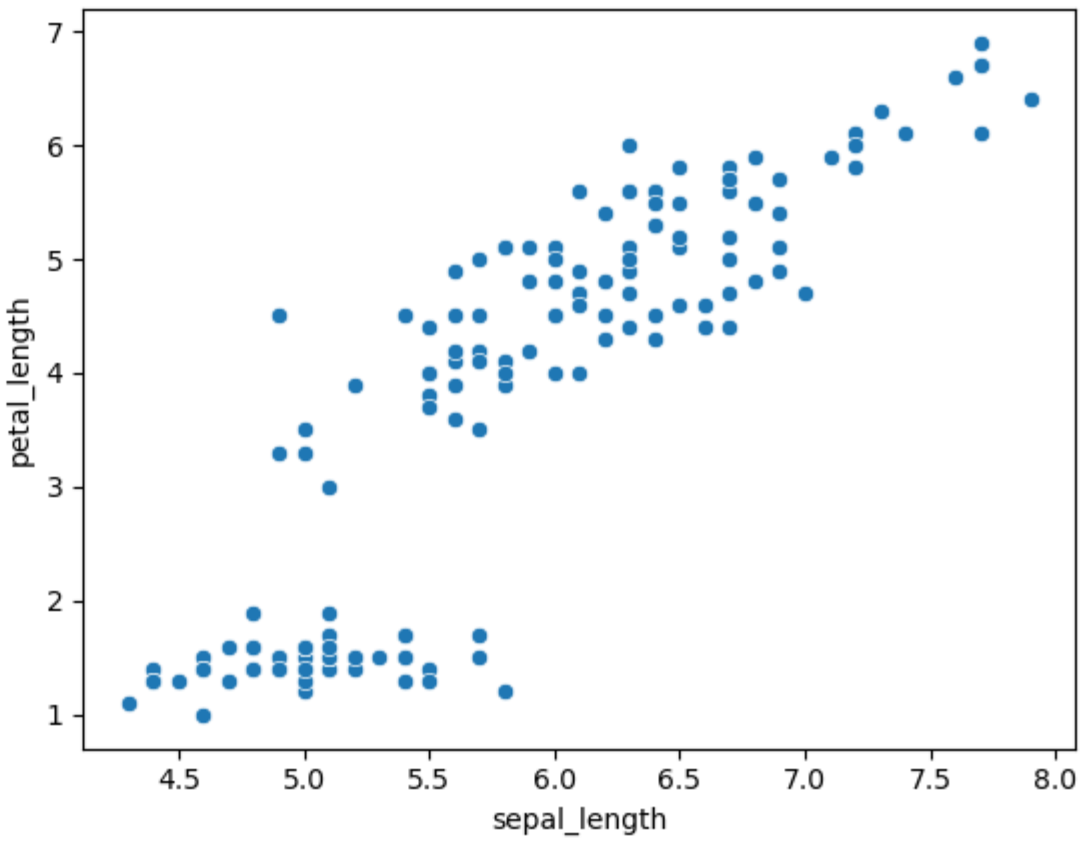
Out[43]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

The scatter plot below shows the relationship between sepal length and petal length for different species of iris flowers.

In [45]:
```
sns.scatterplot(x='sepal_length', y='petal_length',data=iris)
```

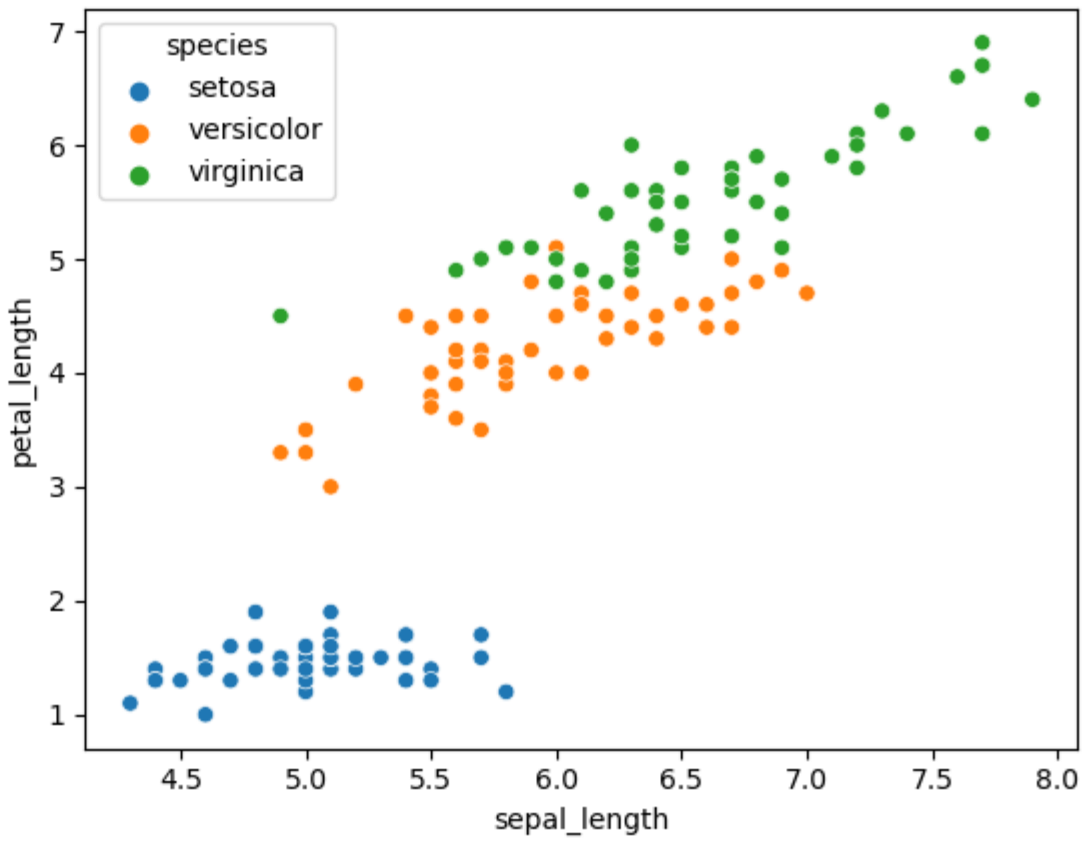
Out[45]:
```
<Axes: xlabel='sepal_length', ylabel='petal_length'>
```



```
In [47]: # using hue parameter you can easily differentiate the three types of iris flowers
```

```
In [48]: sns.scatterplot(x='sepal_length', y='petal_length',data=iris, hue='species')
```

```
Out[48]: <Axes: xlabel='sepal_length', ylabel='petal_length'>
```

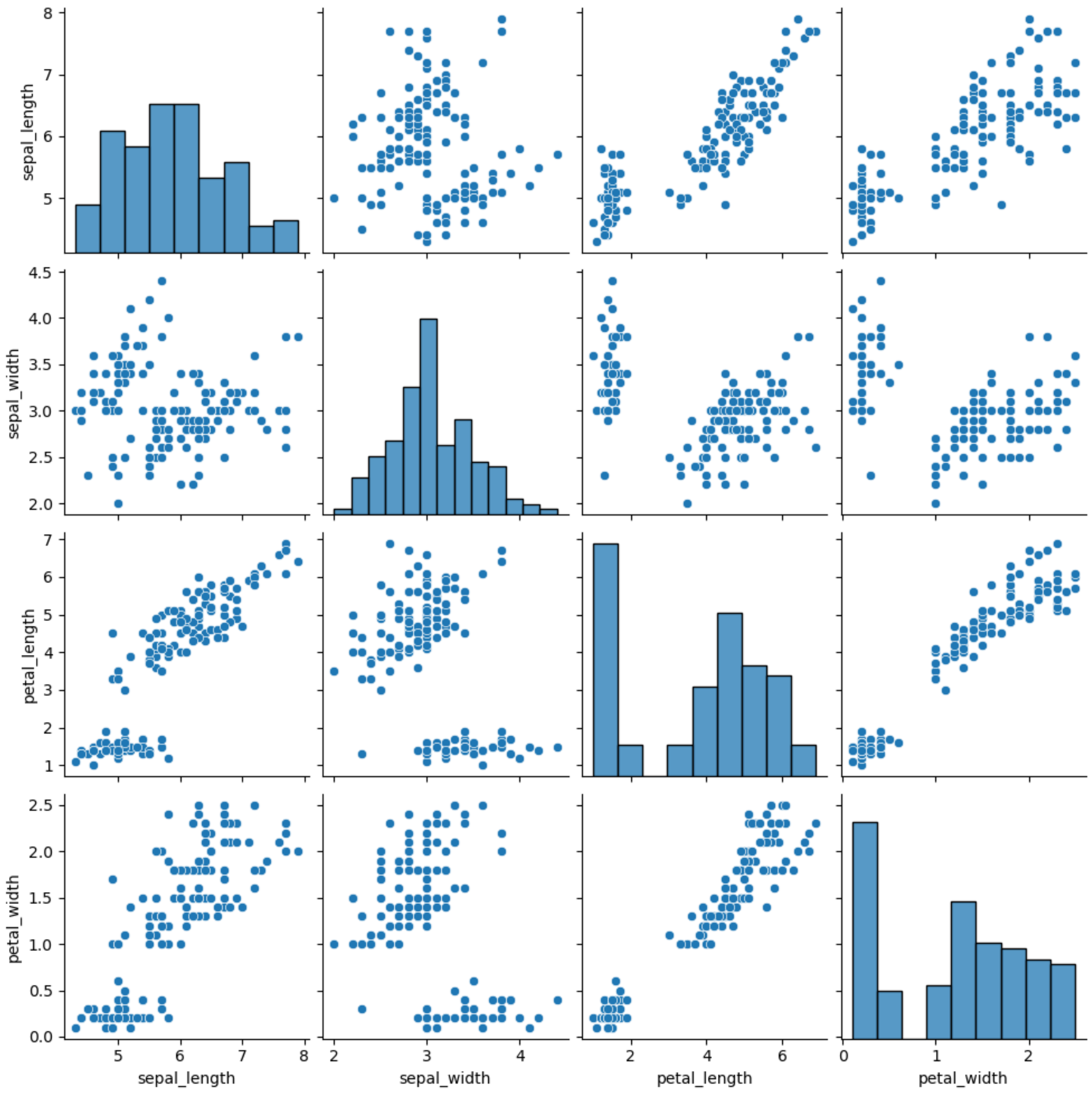


Pairplot The Python Seaborn library lets you visualize data using pair plots that produce a matrix of relationships between each variable in the dataset.

In the below plot, all the plots are histograms that represent the distribution of each feature.

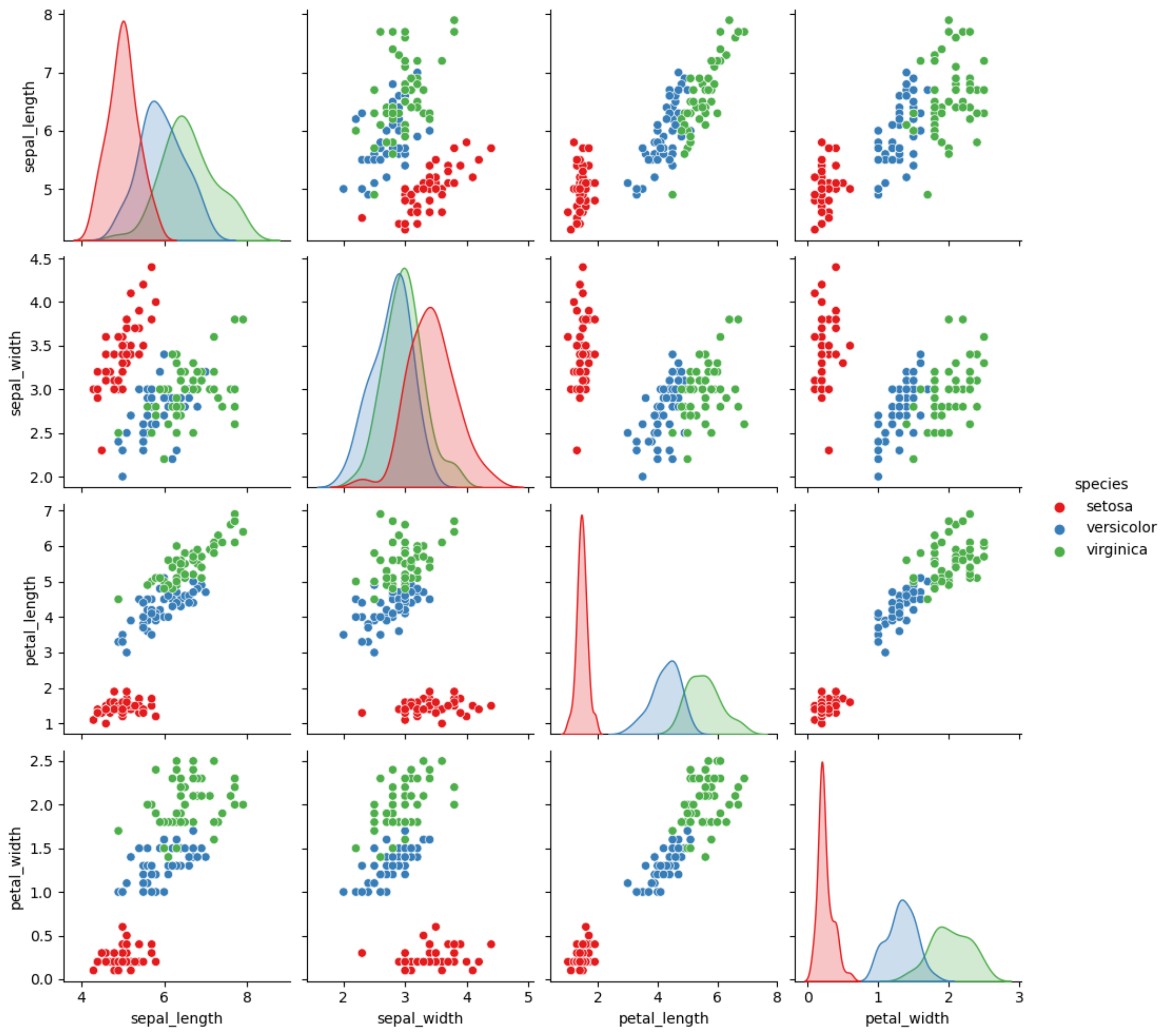
In [49]:
```
sns.pairplot(iris)
```

Out[49]:
```
<seaborn.axisgrid.PairGrid at 0x279c6627cd0>
```



In [50]:
```
#You can convert the diagonal visuals to KDE plots and the rest to scatter plots us
sns.pairplot(iris,hue='species', palette='Set1')
```

Out[50]:
```
<seaborn.axisgrid.PairGrid at 0x279c71b9e10>
```



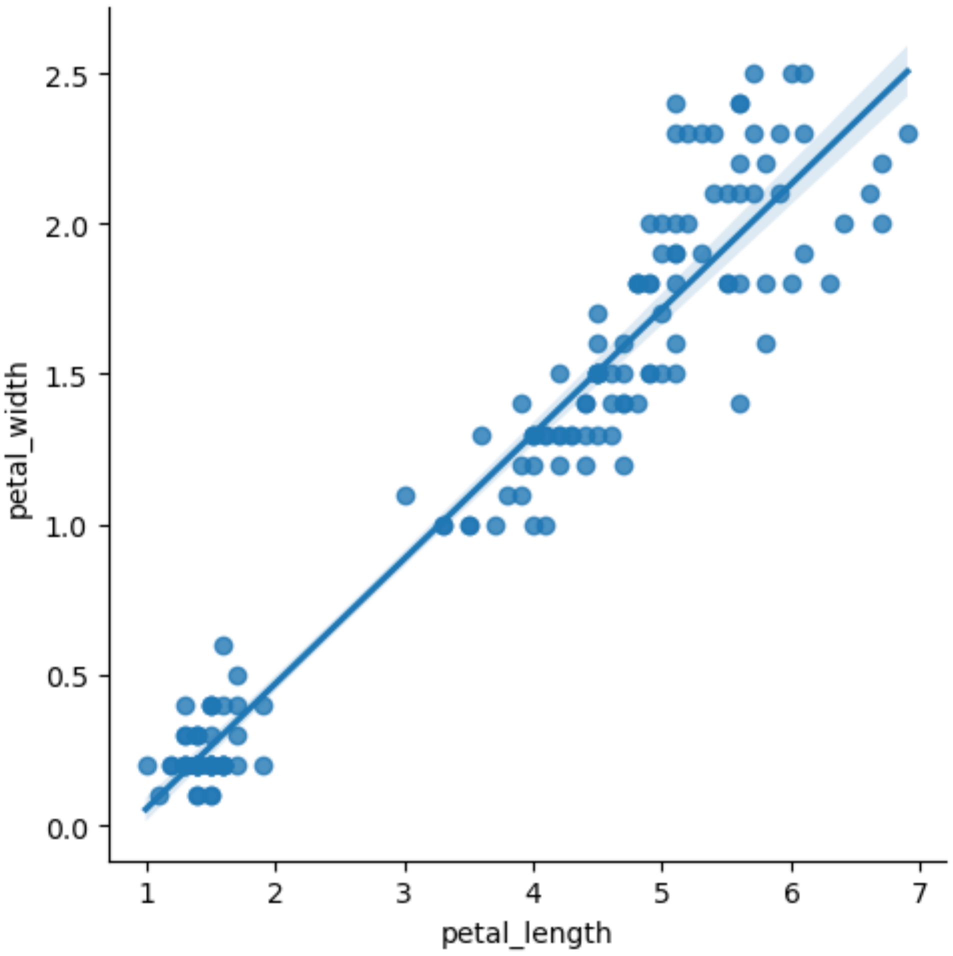
Linear Regression Plot

The lmplot() function in the Seaborn library draws a linear relationship as determined through regression for the continuous variables.

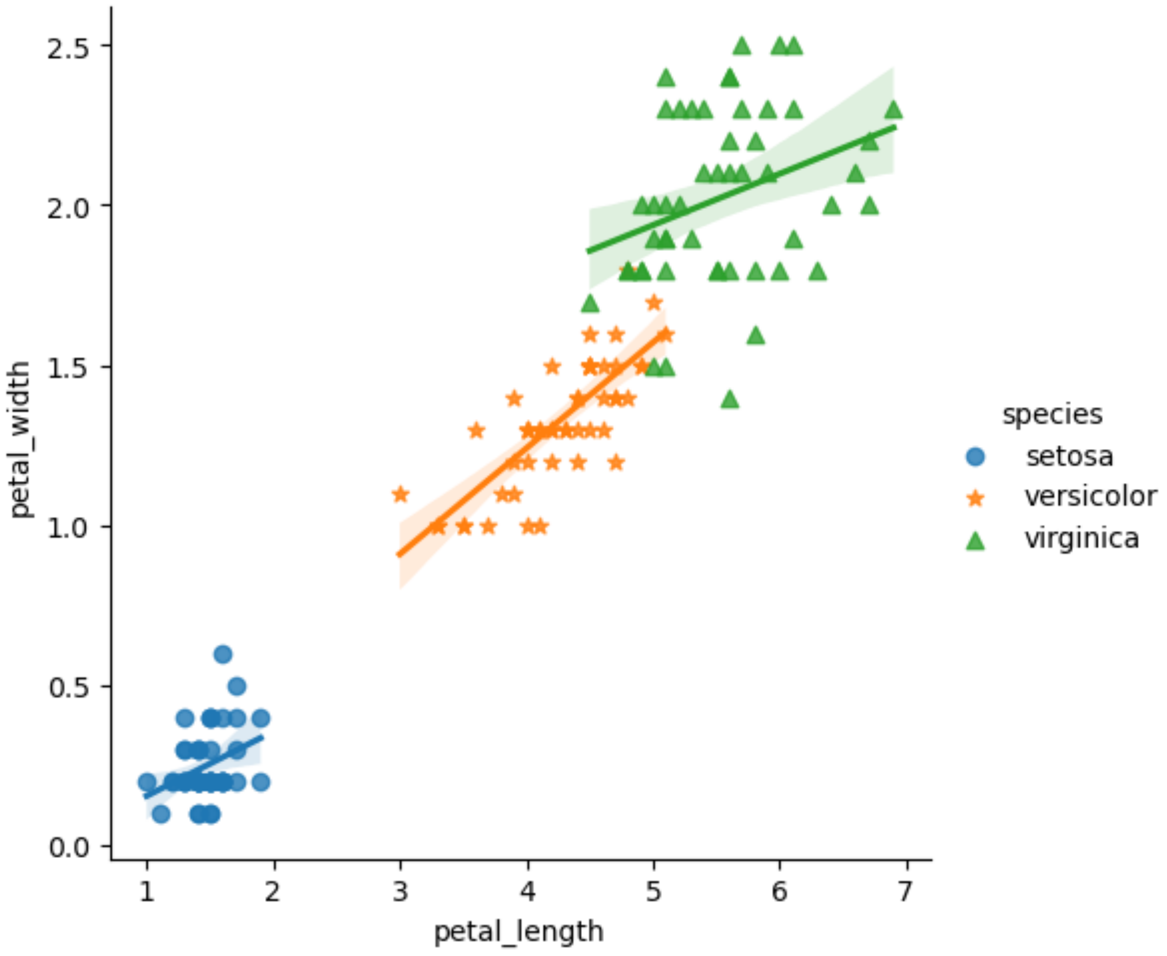
The plot below shows the relationship between petal length and petal width of the different species of iris flowers.

```
In [52]: sns.lmplot(x='petal_length',y='petal_width', data=iris)
```

```
Out[52]: <seaborn.axisgrid.FacetGrid at 0x279c657eec0>
```



```
In [53]: #The hue parameter can differentiate between each species of flower and you can set
         sns.lmplot(x='petal_length',y='petal_width', data=iris, hue='species', markers=["o"

Out[53]: <seaborn.axisgrid.FacetGrid at 0x279c65a2aa0>
```



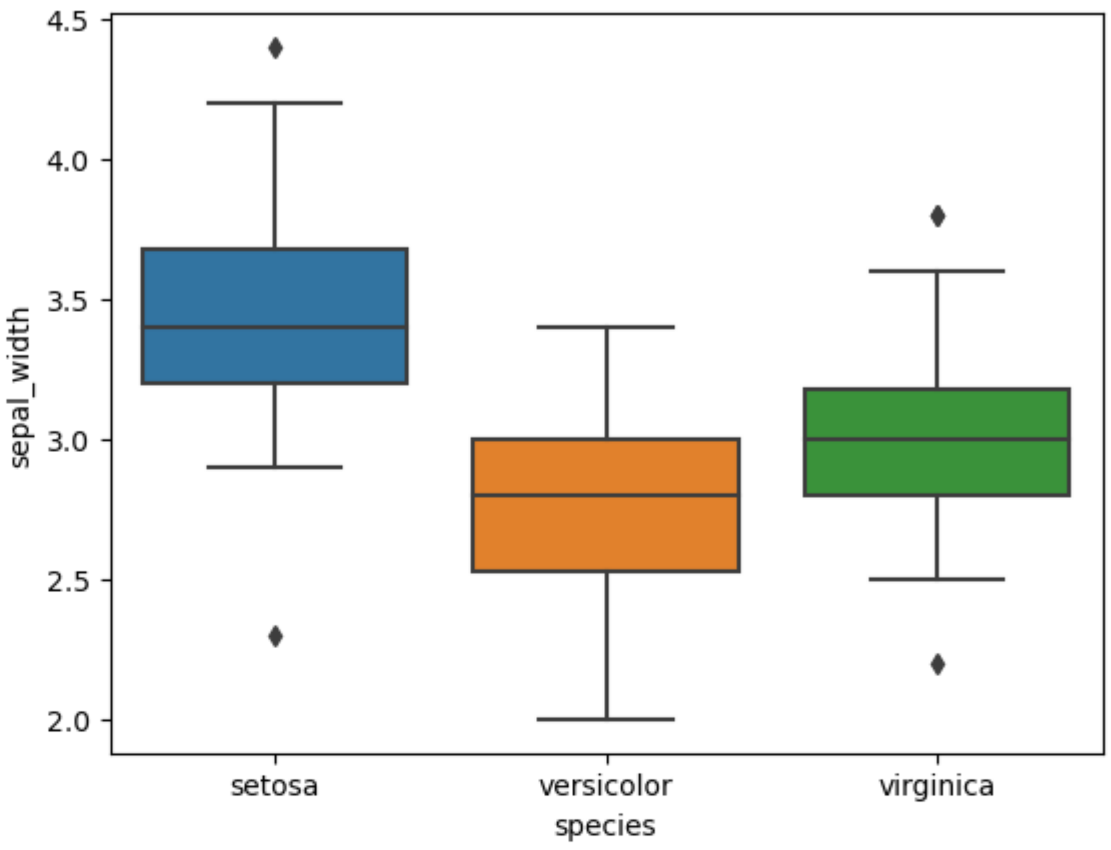
Boxplot

A boxplot, also known as a box and whisker plot, depicts the distribution of quantitative data. The box represents the quartiles of the dataset. The whiskers show the rest of the distribution, except for the outlier points.

The boxplot below shows the distribution of the three species of iris flowers based on their sepal width.

In [54]:
```
sns.boxplot(x='species', y='sepal_width', data=iris)
```

Out[54]:
```
<Axes: xlabel='species', ylabel='sepal_width'>
```



```
In [5]: labels=['cyl','mpg','hp','am']
        sns.pieplot(mtcars, labels=labels, color='red', autopct='%.OF%%')
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[5], line 2
      1 labels=['cyl','mpg','hp','am']
----> 2 sns.pieplot(mtcars, labels=labels, color='red', autopct='%.OF%%')

AttributeError: module 'seaborn' has no attribute 'pieplot'
```

```
In [ ]:
```