mjpoccyfl

July 7, 2023

1 Seaborn

1.1 What is Seaborn?

1.1.1 Seaborn is a powerful data visualization library built on top of Matplotlib, specifically designed for statistical graphics. With its simple yet elegant syntax and extensive functionality, Seaborn empowers data scientists, analysts, and researchers to effortlessly create captivating visual representations of their data.

1.2 Types of Plots in Seaborn

[61]:
```
import seaborn as sns
iris = sns.load_dataset('iris')
```

[60]:
```
iris.head()
```

[60]:
```
   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
```

[62]:
```
iris.tail()
```

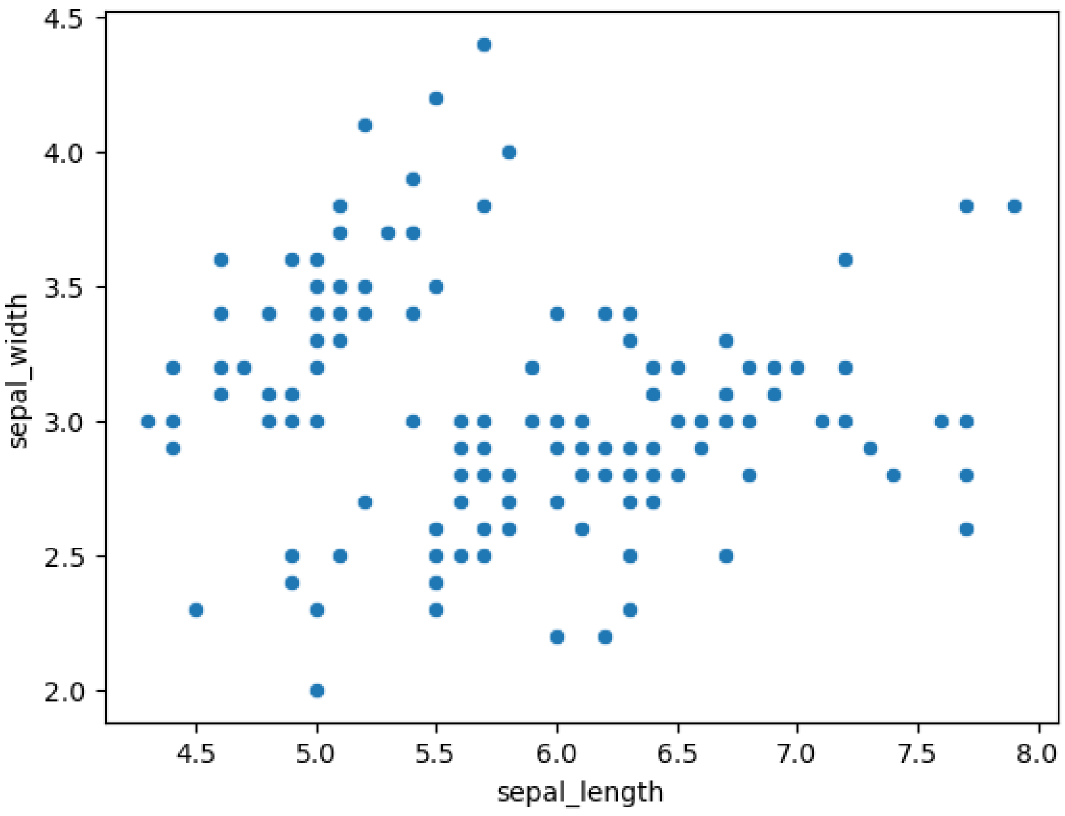
[62]:
```
     sepal_length  sepal_width  petal_length  petal_width    species
145           6.7          3.0           5.2          2.3  virginica
146           6.3          2.5           5.0          1.9  virginica
147           6.5          3.0           5.2          2.0  virginica
148           6.2          3.4           5.4          2.3  virginica
149           5.9          3.0           5.1          1.8  virginica
```

1.2.1 Scatter Plot

A scatter plot is a type of graph that shows the relationship between two sets of data points. It is represented by a collection of points on a two-dimensional plane, where each point represents the value of two different variables.

```
[45]: # Create scatter plot
      sns.scatterplot(x = iris.sepal_length, y= iris.sepal_width)
```

```
[45]: <AxesSubplot: xlabel='sepal_length', ylabel='sepal_width'>
```

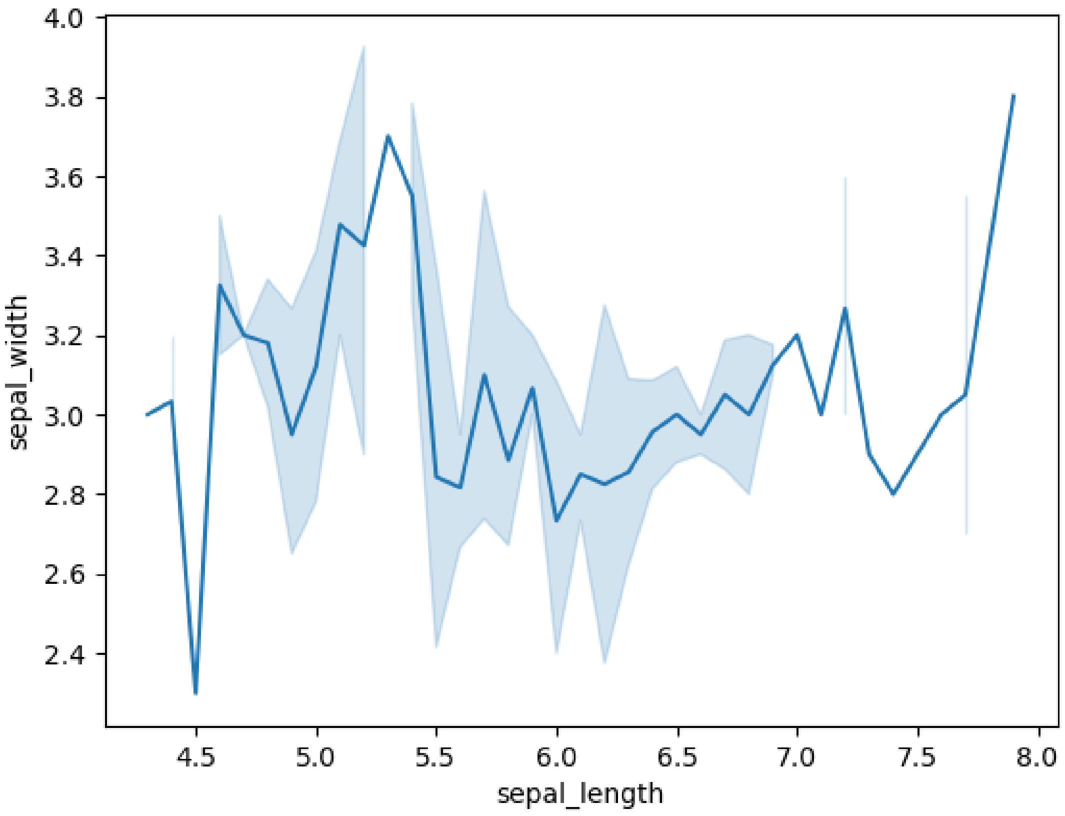


1.2.2 Line Plot

A line plot, also known as a line graph, is a simple and powerful way to visualize how a variable changes over time or across different categories. It uses lines to connect data points, representing the trend or pattern in the data.

```
[46]: # Create line plot
      sns.lineplot(x= iris.sepal_length, y= iris.sepal_width)
```

```
[46]: <AxesSubplot: xlabel='sepal_length', ylabel='sepal_width'>
```

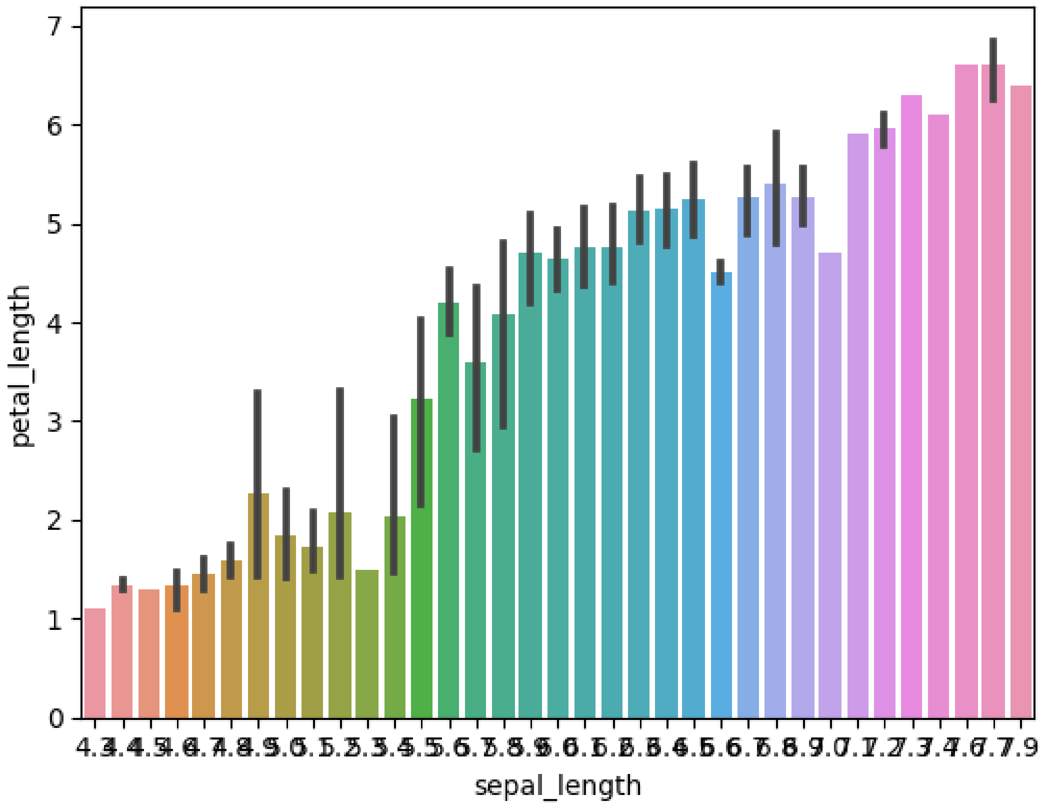


1.2.3 Bar plot

A bar plot is a simple and popular way to represent categorical data using rectangular bars. It shows the frequency, count, or any other numeric value associated with different categories. Each category is represented by a bar, and the length or height of the bar corresponds to the value it represents. It's like comparing different categories using colorful bars!

[47]:
```
sns.barplot(x= 'sepal_length', y= 'petal_length', data= iris)
```

[47]:
```
<AxesSubplot: xlabel='sepal_length', ylabel='petal_length'>
```

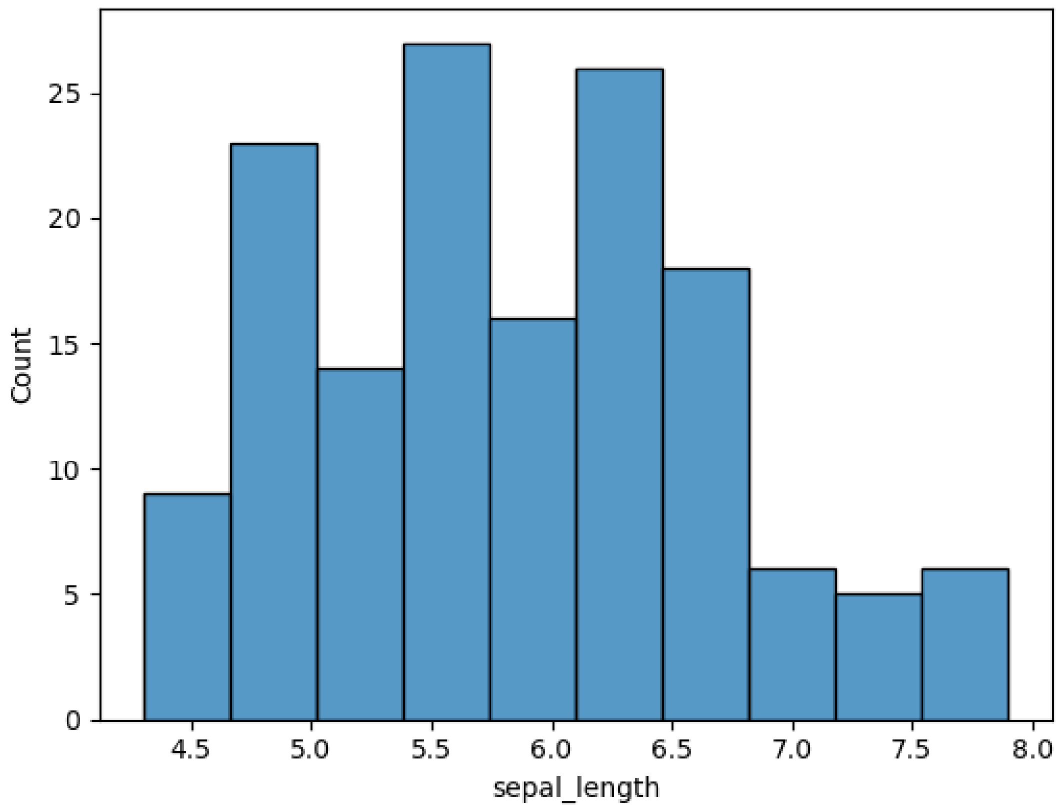


1.2.4 Histogram

A Histogram is like a visual summary of data, showing how frequently certain values occur in a dataset. It looks like a bar chart with each bar representing a range of values and the height representing the number of occurrences.

[48]:
```
sns.histplot(data= iris, x='sepal_length', bins=10)
```

[48]:
```
<AxesSubplot: xlabel='sepal_length', ylabel='Count'>
```



1.2.5 Box Plot

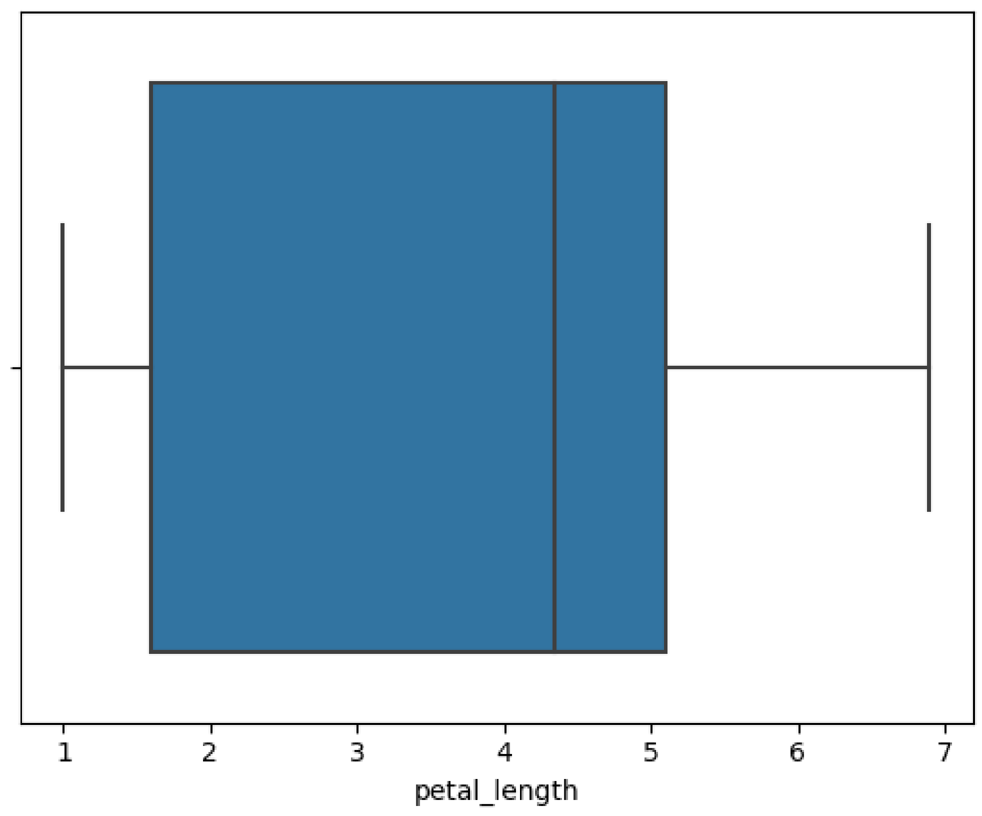
A Box Plot is a graphical representation of the distribution of a dataset. It displays key statistical measures such as the median, quartiles, and potential outliers.

A box plot consists of a rectangle (the box) and two lines (whiskers) extending from it. The rectangle represents the middle 50% of the data, with the median shown as a line in the middle. The whiskers represent the data range, excluding potential outliers.

By visualizing a box plot, you can quickly understand the spread, central tendency, and presence of outliers in a dataset. It provides a concise summary of the distribution, making it useful for comparing multiple groups or identifying unusual data points.

[49]:
```
sns.boxplot(x='petal_length', data= iris)
```

[49]:
```
<AxesSubplot: xlabel='petal_length'>
```



1.2.6 Heat Map

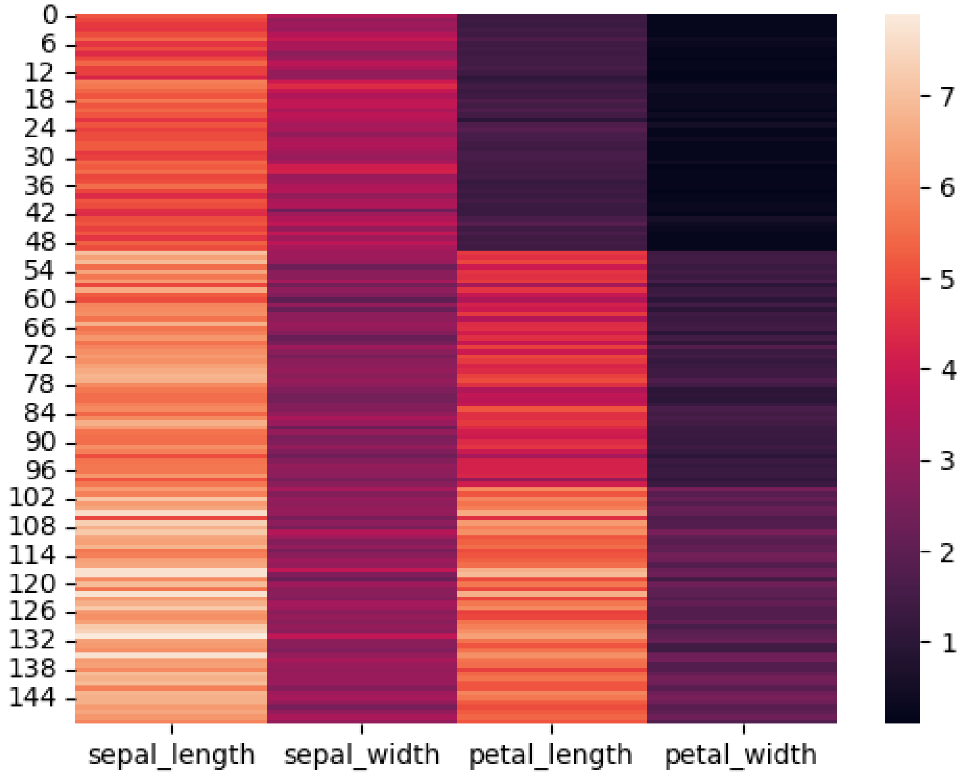
A heatmap is like a colorful map for your data! It uses different shades of colors to represent the intensity or magnitude of values in a dataset.

Think of it as a grid or a table where each cell is filled with a color that corresponds to the value it represents. The colors typically range from cool shades (like blue or green) for lower values to warmer shades (like red or orange) for higher values. This color scheme helps us quickly identify patterns, trends, and variations in the data.

Heatmaps are particularly useful when dealing with large datasets or when you want to identify relationships and patterns between two variables. They make it easy to spot areas of higher or lower values, clusters, or outliers.

```
[50]: iris_2 = iris.drop('species', axis=1)
      sns.heatmap(iris_2)
```

```
[50]: <AxesSubplot: >
```

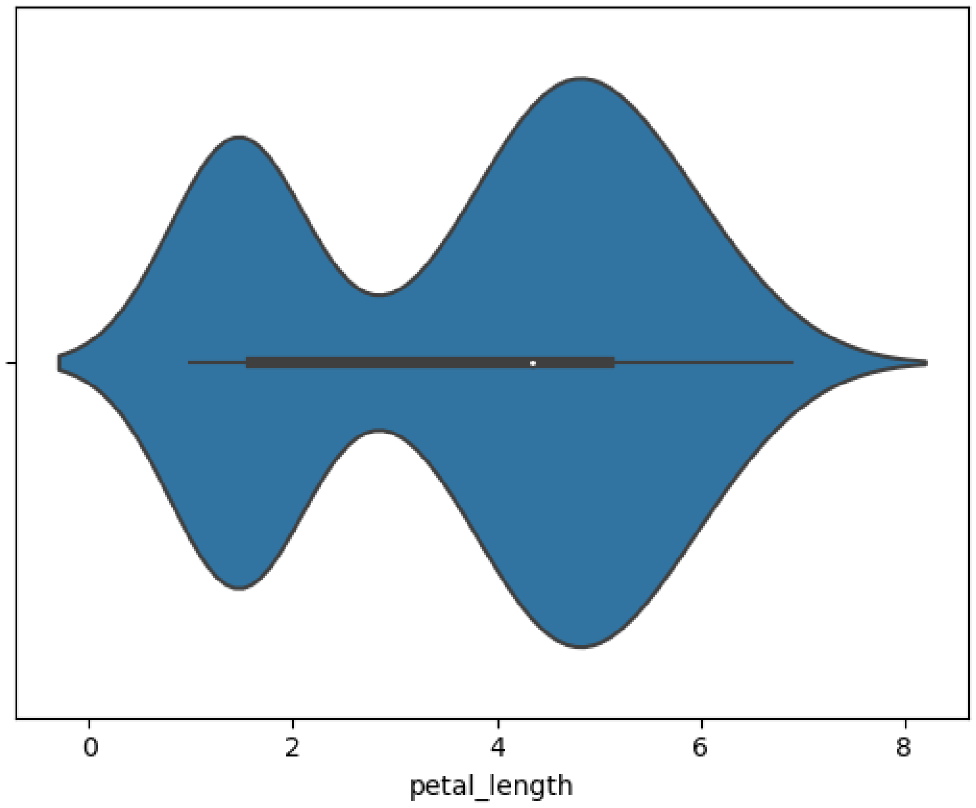


1.2.7 Violinplot

A violin plot is a data visualization technique that combines aspects of a box plot and a kernel density plot. It is used to display the distribution and density of a continuous variable or multiple groups. The shape of the plot resembles a violin, hence the name!

[51]:
```
sns.violinplot(x='petal_length', data=iris)
```

[51]:
```
<AxesSubplot: xlabel='petal_length'>
```



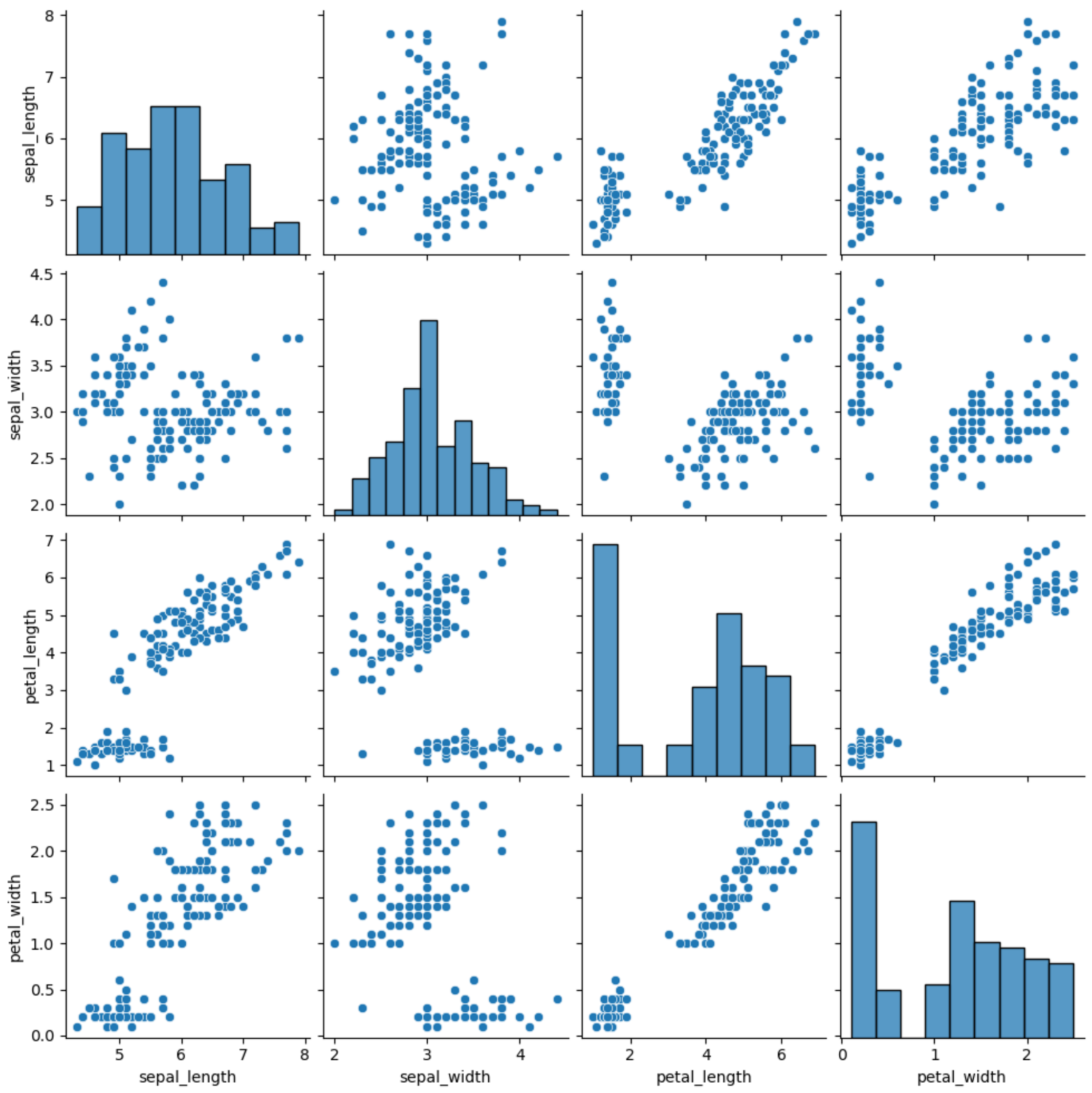
1.2.8 Pairplot

A pair plot is like a snapshot of the relationships between multiple variables in a dataset. It creates a grid of scatter plots, where each variable is compared with every other variable, showcasing patterns, correlations, and distributions.

It's like having multiple scatter plots side by side, allowing us to quickly identify any interesting connections or trends among the variables. It's a powerful tool for exploratory data analysis and gaining insights into the data.

```
[52]: sns.pairplot(iris)
```

```
[52]: <seaborn.axisgrid.PairGrid at 0x7f84c9c19930>
```



1.2.9 Joint plot

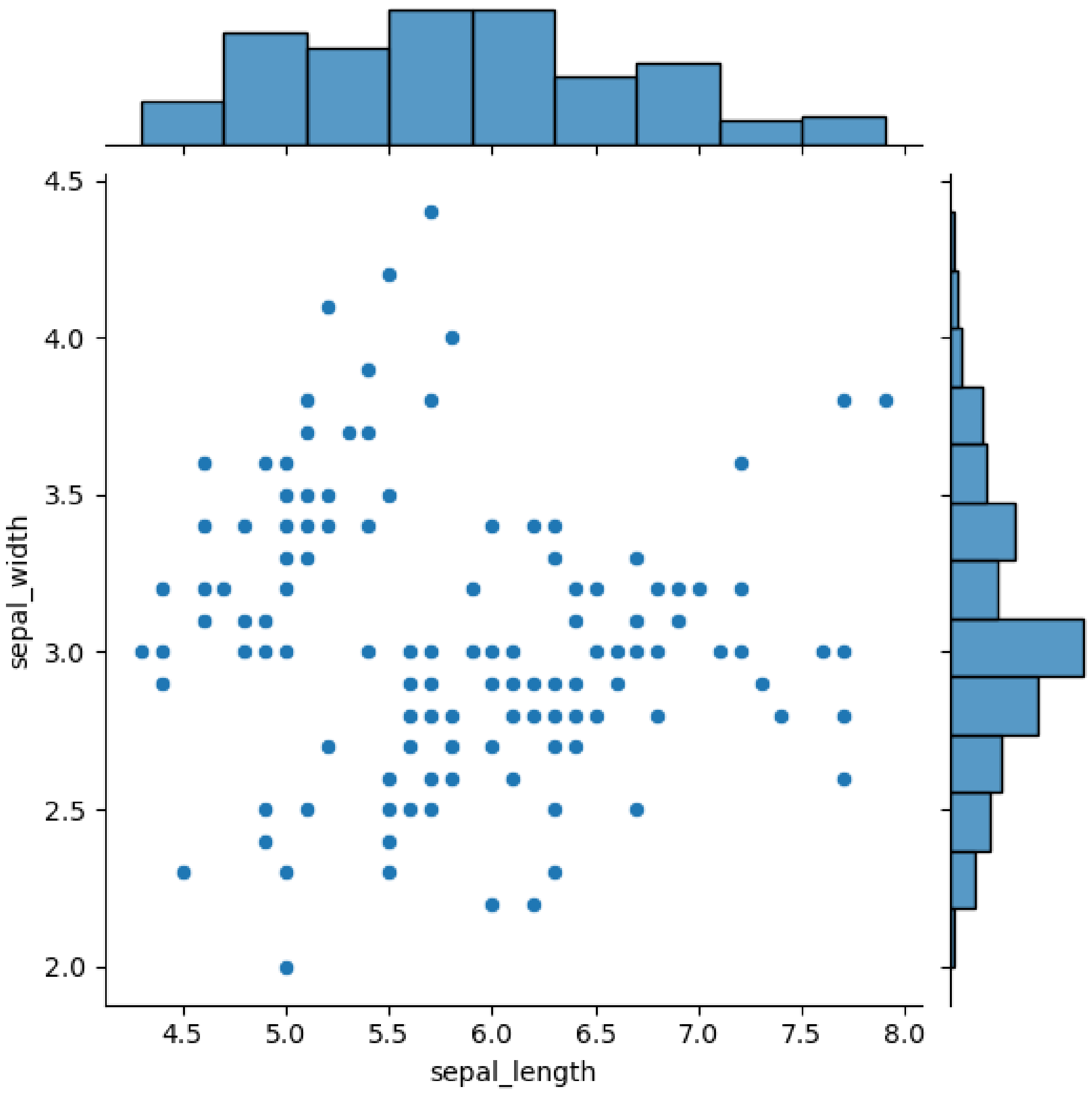
A joint plot is a visualization in Seaborn that allows us to explore the relationship between two numerical variables in a single plot. It combines a scatter plot and histograms to provide insights into the distribution and correlation of the variables.

By using a joint plot, we can quickly assess how the two variables interact with each other and identify any potential patterns or trends. It's like having two visualization techniques rolled into one!

To create a joint plot, we can use the jointplot() function in Seaborn. It takes the two variables we want to examine as input and generates a scatter plot with histograms on the top and side.

```
[53]: sns.jointplot(x='sepal_length', y='sepal_width', data= iris, kind='scatter')
```

```
[53]: <seaborn.axisgrid.JointGrid at 0x7f84c9f93bb0>
```



1.2.10 KDE plot

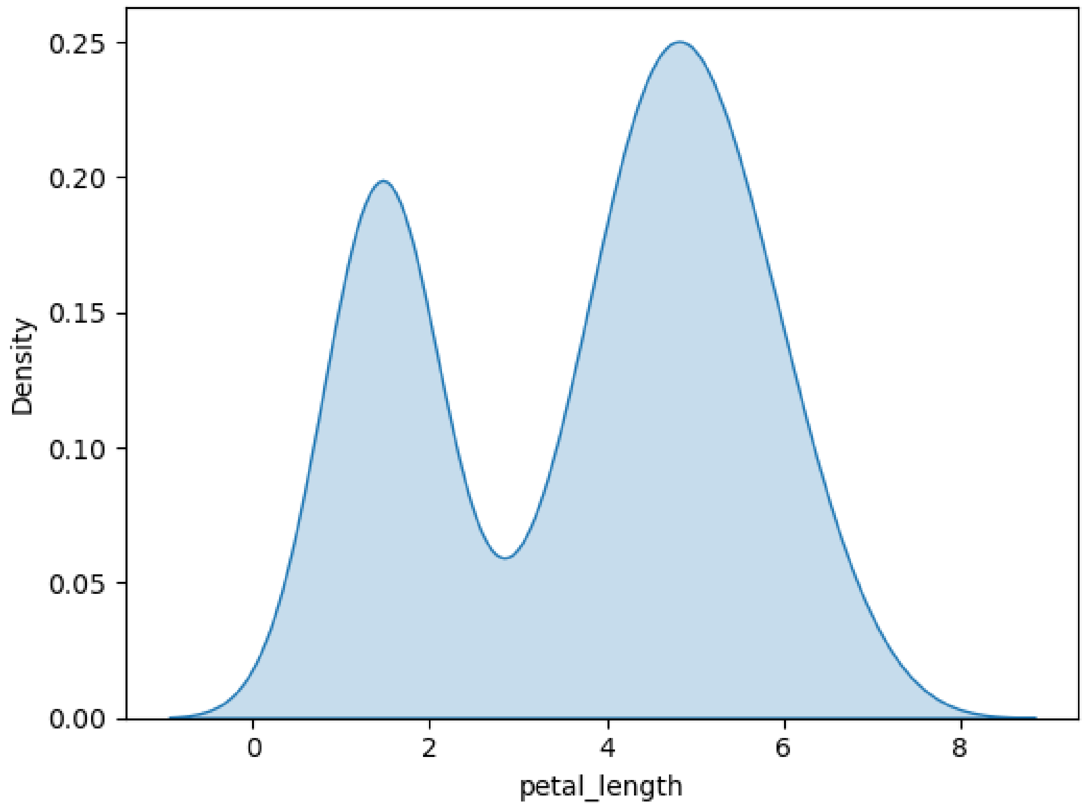
A KDE (Kernel Density Estimation) plot is a type of plot in data visualization that showcases the distribution of a dataset. It provides a smooth and continuous representation of the data's probability density, helping us understand how values are spread out.

```
[54]: sns.kdeplot(iris['petal_length'], shade=True)
```

```
/tmp/ipykernel_2478/2536236113.py:1: FutureWarning:
```

```
`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(iris['petal_length'], shade=True)
```

[54]:
```
<AxesSubplot: xlabel='petal_length', ylabel='Density'>
```



1.2.11 Swarm plot

A swarm plot is a type of categorical scatter plot that shows the distribution of categorical data points.

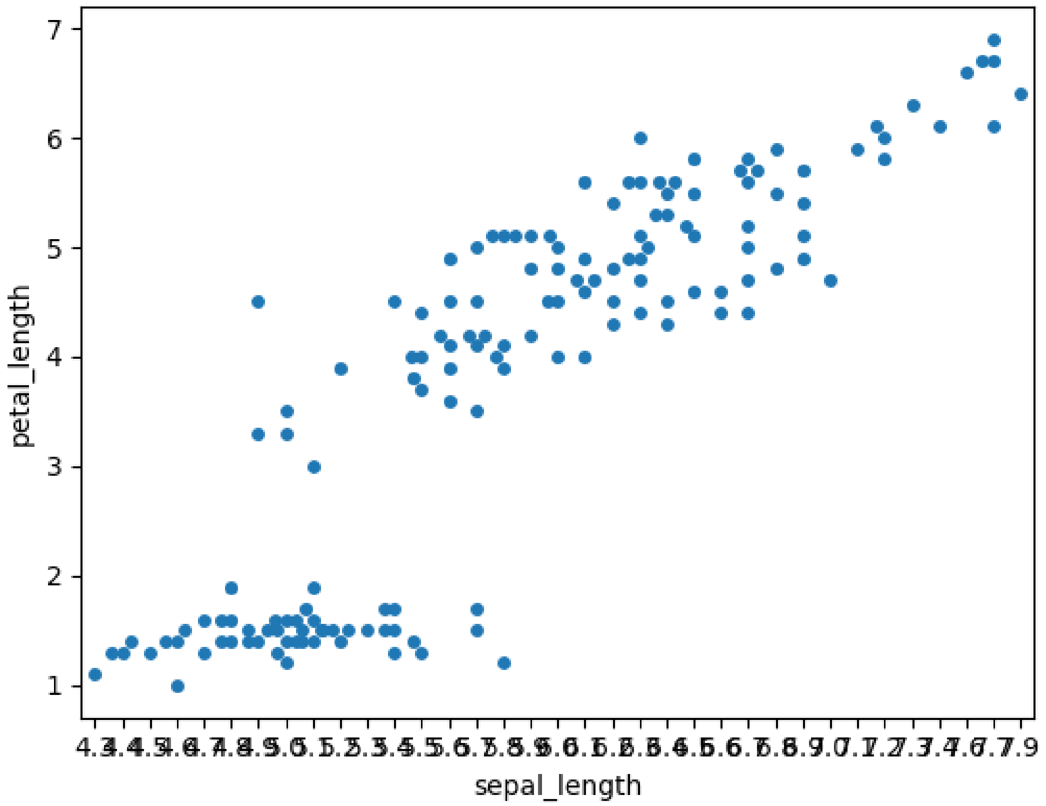
In a swarm plot, each data point is represented by a small "bee" symbol. The bees are aligned along the categorical axis, and they spread out horizontally to avoid overlapping with each other. This way, we can see the density and distribution of the data points within each category.

[55]:
```python
sns.swarmplot(x='sepal_length', y='petal_length', data = iris)
```

```
/opt/conda/lib/python3.10/site-packages/seaborn/categorical.py:3543:
UserWarning: 33.3% of the points cannot be placed; you may want to decrease the
```

```
size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
/opt/conda/lib/python3.10/site-packages/seaborn/categorical.py:3543:
UserWarning: 25.0% of the points cannot be placed; you may want to decrease the
size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
/opt/conda/lib/python3.10/site-packages/seaborn/categorical.py:3543:
UserWarning: 40.0% of the points cannot be placed; you may want to decrease the
size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
/opt/conda/lib/python3.10/site-packages/seaborn/categorical.py:3543:
UserWarning: 30.0% of the points cannot be placed; you may want to decrease the
size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
/opt/conda/lib/python3.10/site-packages/seaborn/categorical.py:3543:
UserWarning: 14.3% of the points cannot be placed; you may want to decrease the
size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
/opt/conda/lib/python3.10/site-packages/seaborn/categorical.py:3543:
UserWarning: 28.6% of the points cannot be placed; you may want to decrease the
size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
/opt/conda/lib/python3.10/site-packages/seaborn/categorical.py:3543:
UserWarning: 16.7% of the points cannot be placed; you may want to decrease the
size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
/opt/conda/lib/python3.10/site-packages/seaborn/categorical.py:3543:
UserWarning: 22.2% of the points cannot be placed; you may want to decrease the
size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
```

```
[55]: <AxesSubplot: xlabel='sepal_length', ylabel='petal_length'>
```



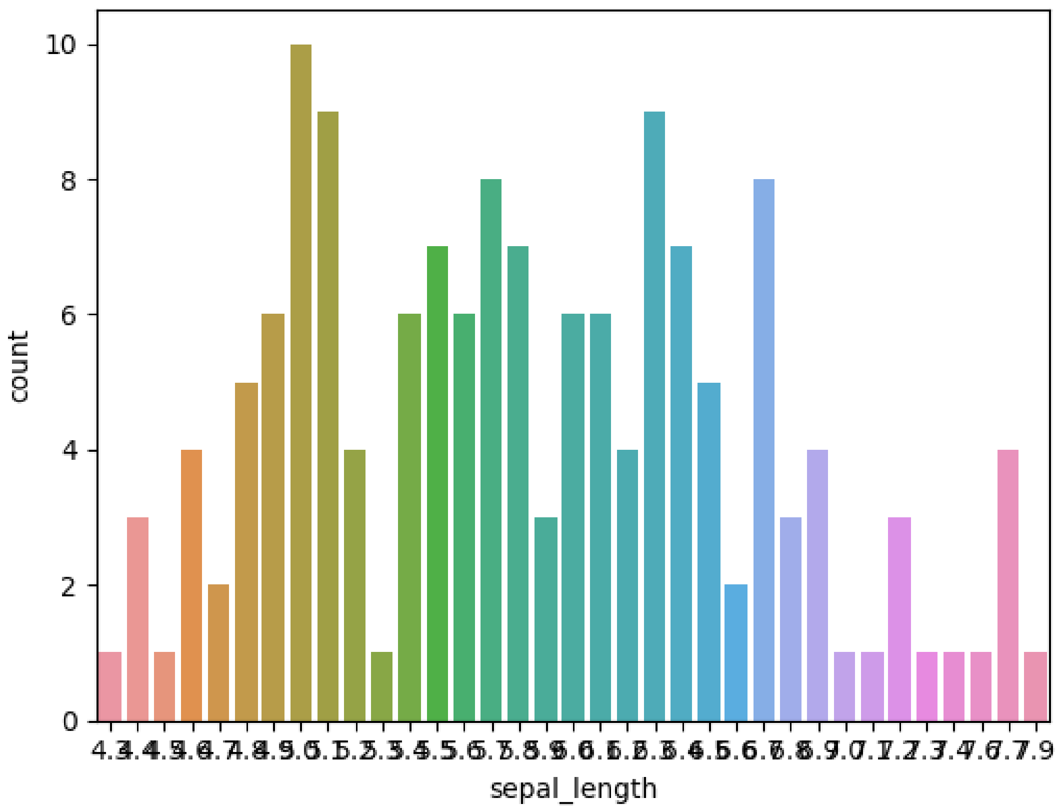
1.2.12 Count plot

A count plot is a type of plot that displays the count or frequency of different categories in a categorical variable. It helps you visualize the distribution and comparison of categorical data.

It counts the occurrences of each category and represents them as bars or columns on the plot, with the height of each bar indicating the count.

[56]:
```
sns.countplot(x='sepal_length', data= iris)
```

[56]:
```
<AxesSubplot: xlabel='sepal_length', ylabel='count'>
```

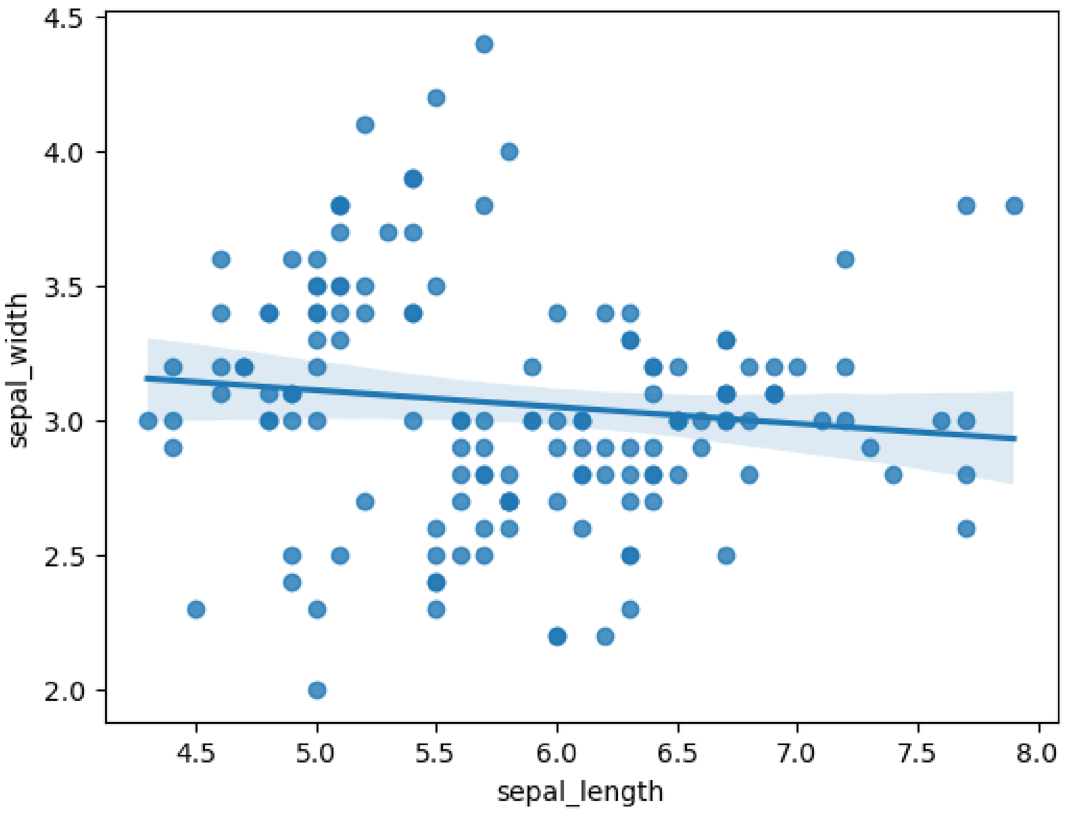


1.2.13 Reg plot (Regression plot)

Regplot is a statistical plotting function used to visualize the relationship between two variables. It helps us understand how one variable (called the independent variable or predictor) affects another variable (called the dependent variable or response).

```
[57]: sns.regplot(x='sepal_length', y='sepal_width', data= iris)
```

```
[57]: <AxesSubplot: xlabel='sepal_length', ylabel='sepal_width'>
```

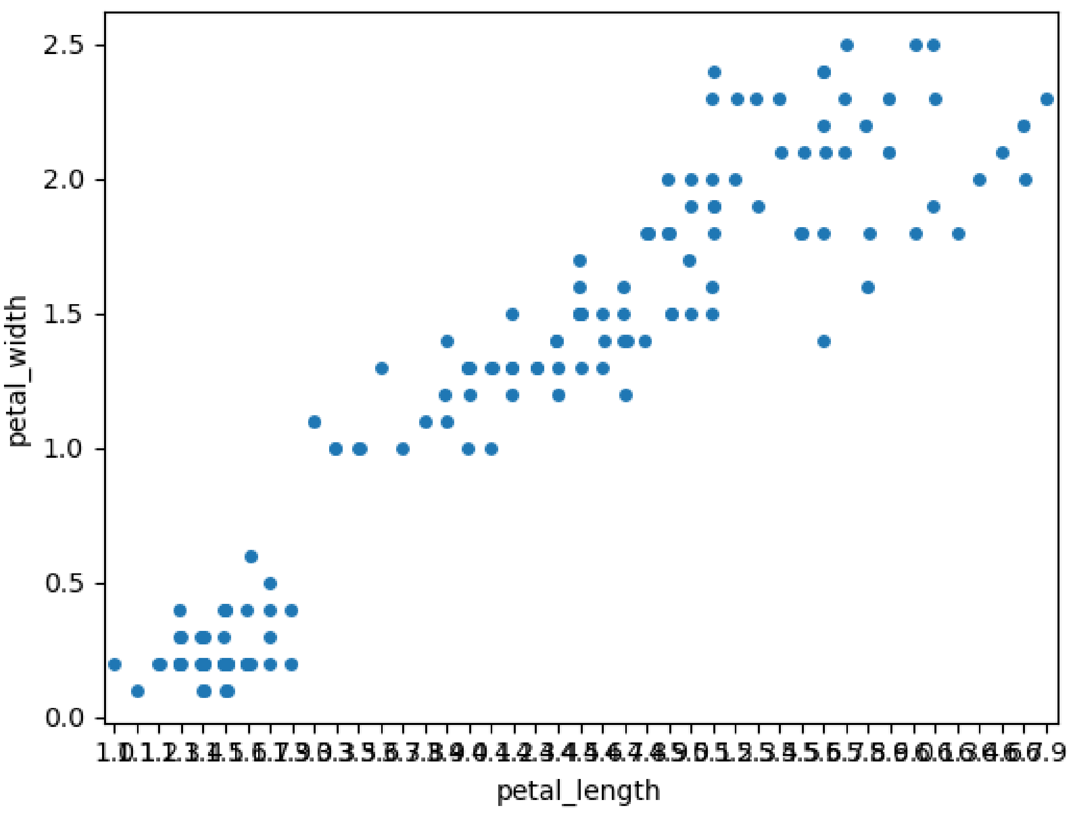


1.2.14 Strip plot

A strip plot is a way to visualize data points along a continuous axis, such as a number line. It shows individual data points as small marks or "strips" in a single dimension, allowing you to see the distribution and density of the data.

```
[58]: sns.stripplot(x='petal_length', y='petal_width', data=iris)
```

```
[58]: <AxesSubplot: xlabel='petal_length', ylabel='petal_width'>
```

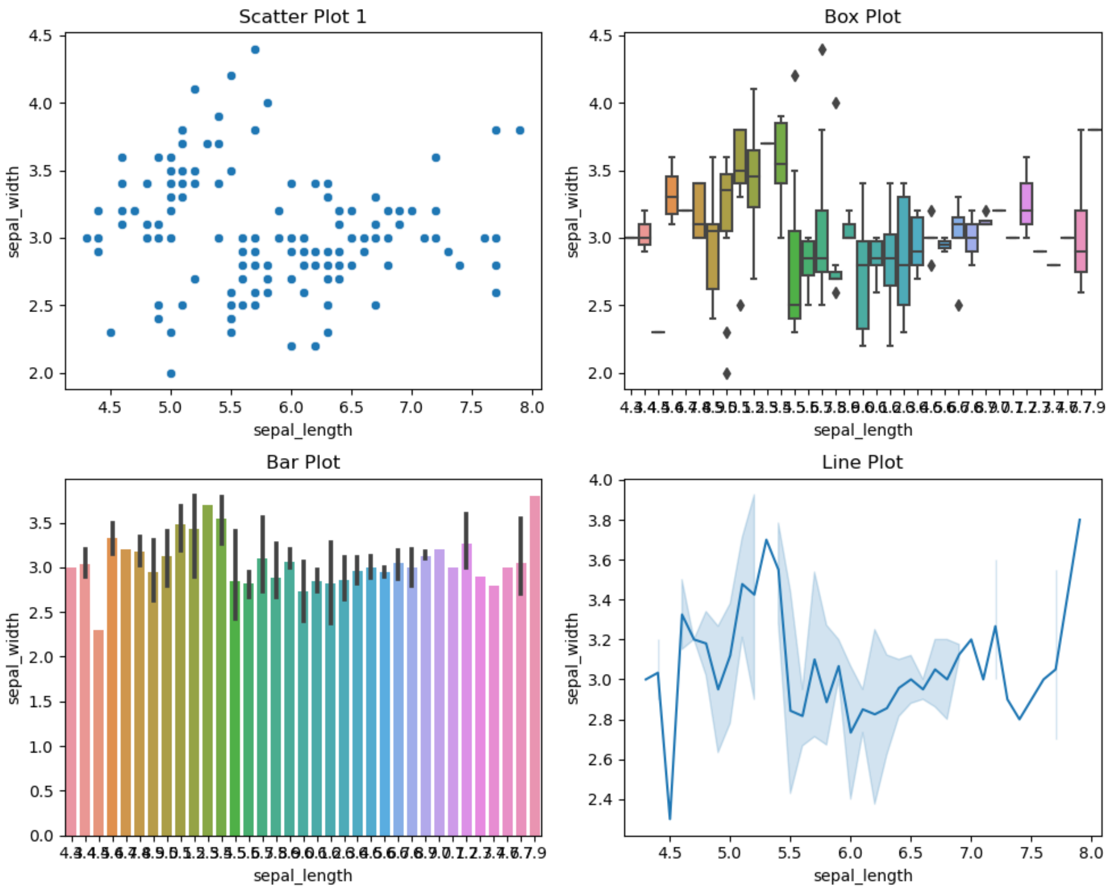


1.2.15 Subplot

Seaborn is primarily focused on creating high-level statistical graphics, and it does not have direct support for creating subplots. However, you can still use Seaborn in conjunction with Matplotlib to create subplots and incorporate Seaborn plots within them.

[59]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Create subplots
fig, axes = plt.subplots(2, 2, figsize=(10, 8))

# Subplot 1
sns.scatterplot(x=iris['sepal_length'], y= iris['sepal_width'], ax=axes[0, 0])
axes[0, 0].set_title('Scatter Plot 1')

# Subplot 2
sns.boxplot(x=iris['sepal_length'], y= iris['sepal_width'], ax=axes[0, 1])
axes[0, 1].set_title('Box Plot')

# Subplot 3
```

```
sns.barplot(x=iris['sepal_length'], y= iris['sepal_width'], ax=axes[1, 0])
axes[1, 0].set_title('Bar Plot')

# Subplot 4
sns.lineplot(x=iris['sepal_length'], y= iris['sepal_width'], ax=axes[1, 1])
axes[1, 1].set_title('Line Plot')

# Adjust spacing between subplots
plt.tight_layout()

# Display the subplots
plt.show()
```


Scatter Plot 1
Box Plot
Bar Plot
Line Plot
sepal_width
sepal_length