dav-using-matplotlib-seaborn

September 17, 2024

0.1 DATAVISUALISATION USING MATPLOTLIB AND SEABORN USING IRIS DATASET

Data visualization is a key component of data analysis, helping to convey information and insights through graphical representations. Visualizations make complex data more understandable, highlight trends, patterns, and outliers, and facilitate better decision-making.

0.2 1.Matplotlib

Matplotlib is a powerful and widely-used library in Python for creating static, interactive, and animated visualizations. It provides a variety of plotting functions and customization options, making it highly versatile for data visualization tasks.

Key Concepts in Matplotlib: 1.Figure and Axes: A Figure is the overall window or page that holds one or more Axes (plots). Each Axes represents a specific plot or graph.

2.Plot Types: Matplotlib supports various plot types including line plots, scatter plots, bar charts, histograms, and more.

3.Customization: You can customize plots in Matplotlib by modifying attributes like colors, labels, titles, and axis limits.

Main Features of Matplotlib 1.High-quality 2D and 3D plots. 2. Extensive customization options (color, labels, grids, etc.). 3.Integration with other libraries like NumPy, pandas, and more.

Basic Syntax: To use Matplotlib, first import the pyplot module, which provides a MATLAB-like interface for creating plots1.High-quality 2D and 3D plots.

[2]:

```
import matplotlib.pyplot as plt
```

Common Plot Types in Matplotlib: 1.Line plots for continuous data.

2.Scatter plots for visualizing the relationship between two numerical variables.

3.Histograms for understanding the distribution of data.

[ ]:

```

```

0.2.1 2.Seaborn

Seaborn is a Python data visualization library built on top of Matplotlib. It is designed to provide a high-level interface for drawing attractive and informative statistical graphics. Seaborn works seamlessly with pandas data structures and simplifies the process of creating complex visualizations with just a few lines of code.

- **Main Features of Seaborn:** Automatic handling of data frames and statistical plotting. O Built-in themes for aesthetics and improved readability. Complex plots like violin plots, pair plots, heatmaps, and more, with minimal code

Basic Syntax: You can start using Seaborn by importing it as

[3]:
```
import seaborn as sns
```

- **plot Types in seaborn:** Pair plots to show pairwise relationships in a dataset.

Box plots and violin plots for visualizing the distribution of data

Heatmaps for visualizing the correlation between variables.

0.2.2 3.Iris Dataset

The Iris dataset is a classic dataset in the field of statistics and machine learning. It consists of 150 observations of iris flowers, with each observation having four features (sepal length, sepal width, petal length, and petal width) and a label indicating the species of the iris (Setosa, Versicolor, or Virginica). The Iris dataset is commonly used for practice in data analysis and visualization because of its simplicity and well-defined structure.

The Iris dataset consists of 150 observations from 3 species of iris flowers (setosa, versicolor, and virginica). Each observation contains the following features: sepal length (cm) sepal width (cm) petal length (cm) petal width (cm)

[5]:
```
#To load iris dataset
import seaborn as sns
import matplotlib.pyplot as plt
# Load the Iris dataset
iris=sns.load_dataset('iris')
print(iris)
```

```
     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
..            …            …             …            …        …
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
```

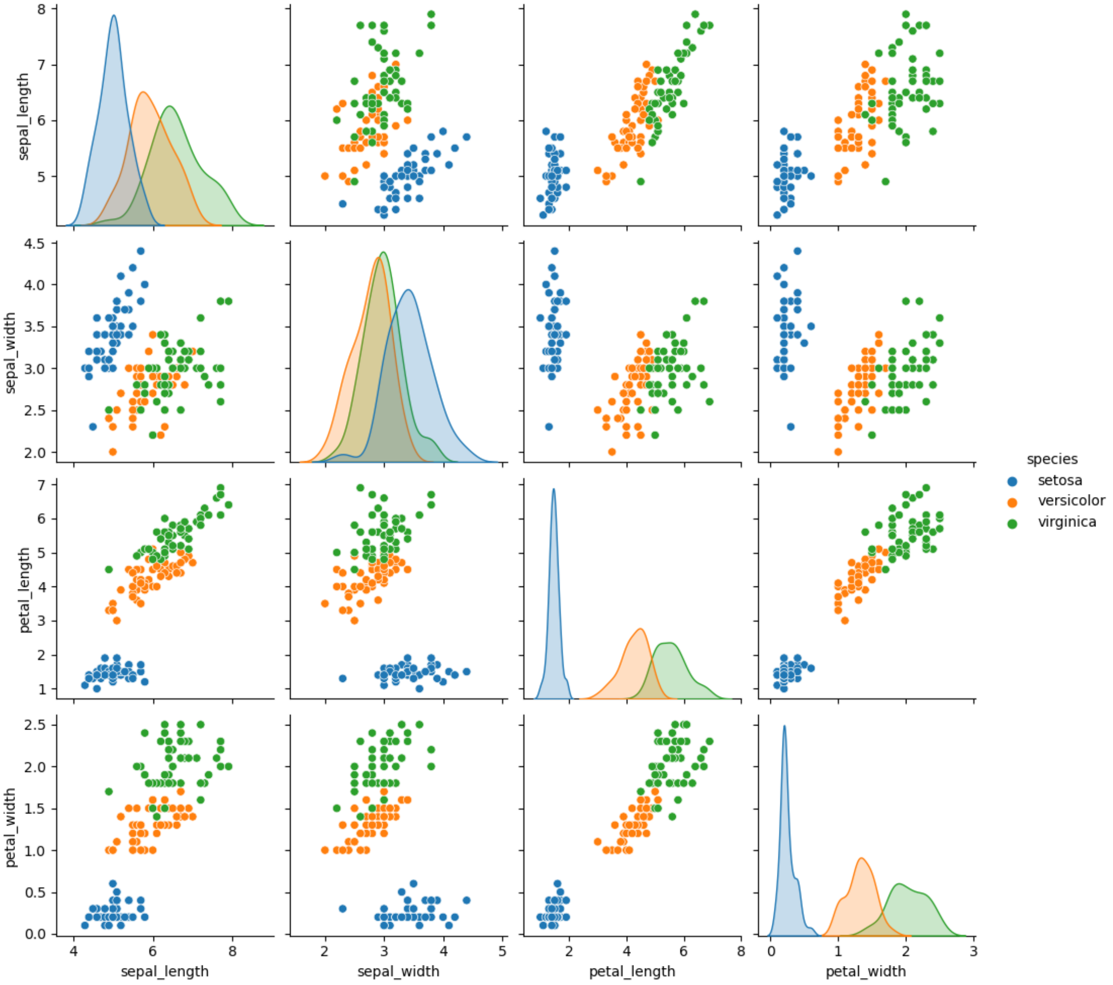
```
148            6.2          3.4           5.4          2.3  virginica
149            5.9          3.0           5.1          1.8  virginica

[150 rows x 5 columns]
```

0.2.3 Sample codes

1.General statistics plot(matplotlib or seaborn):

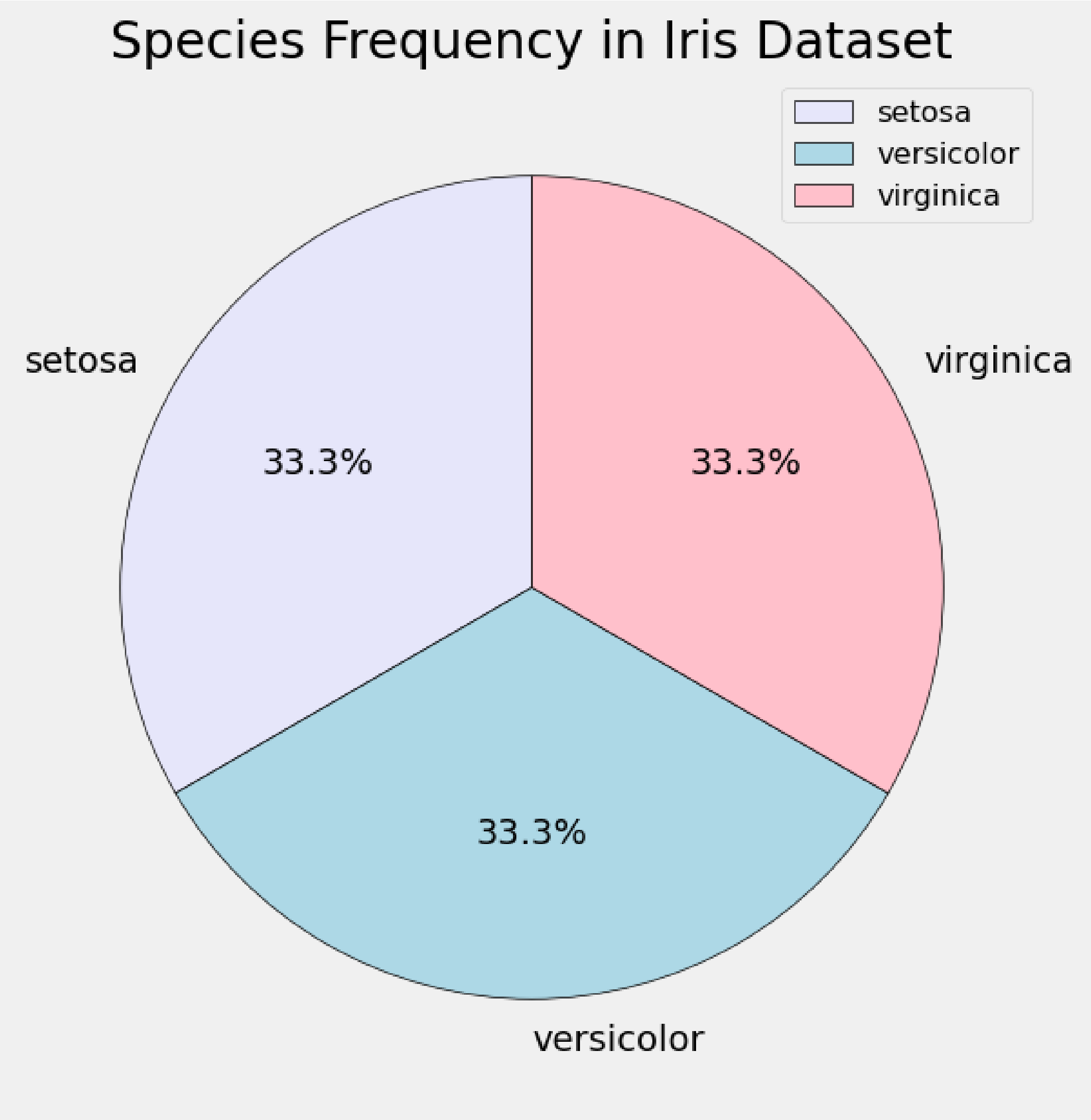
```
[6]: sns.pairplot(iris, hue='species', height=2.5)
     plt.show()
```

```
C:\Users\anits-csm\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118:
UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```



2.Pieplot for species frequency:

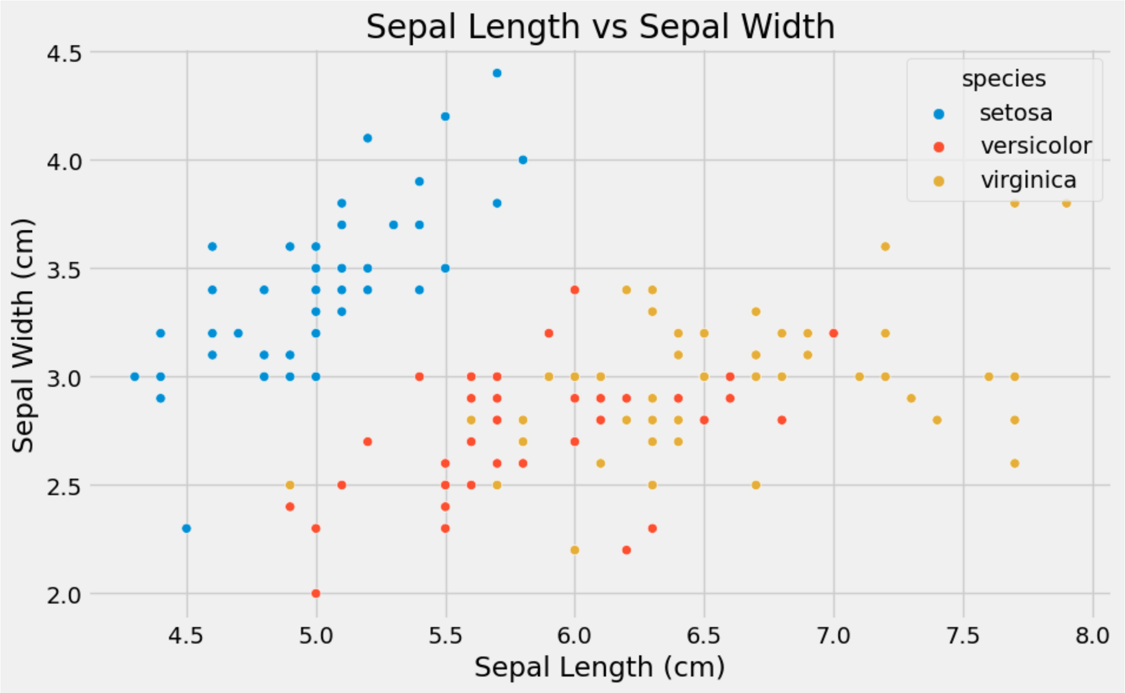
```
[39]: species_counts = iris['species'].value_counts()
      plt.figure(figsize=(7,7))
      plt.tight_layout()
      plt.style.use('fivethirtyeight')
      plt.pie(species_counts, labels=species_counts.index, autopct='%1.1f%%',␣
        ↪startangle=90,wedgeprops={'edgecolor':
        ↪'black'},colors=['lavender','lightblue','pink'])
      plt.title('Species Frequency in Iris Dataset')
      plt.legend(fontsize='small')
      plt.show()
```



3.Relationship between sepal length and sepal width

[33]:

```
plt.figure(figsize=(10, 6))
sns.scatterplot(x='sepal_length', y='sepal_width', hue='species', data=iris)
plt.title('Sepal Length vs Sepal Width')
plt.xlabel('Sepal Length (cm)')
plt.ylabel('Sepal Width (cm)')
plt.show()
```

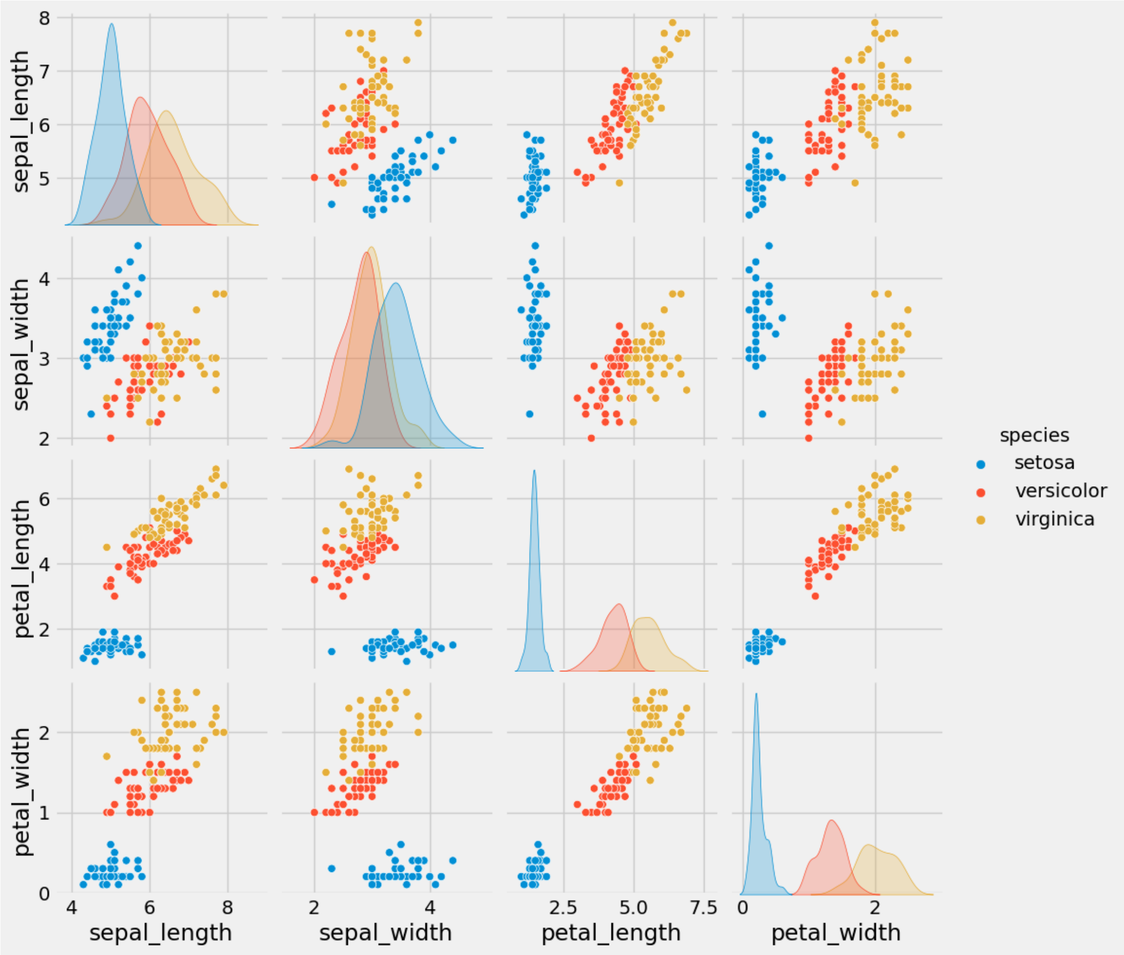


4.Distribution of sepal and petal features

[43]:

```
sns.pairplot(iris, hue='species', height=2.5)
plt.show()
```

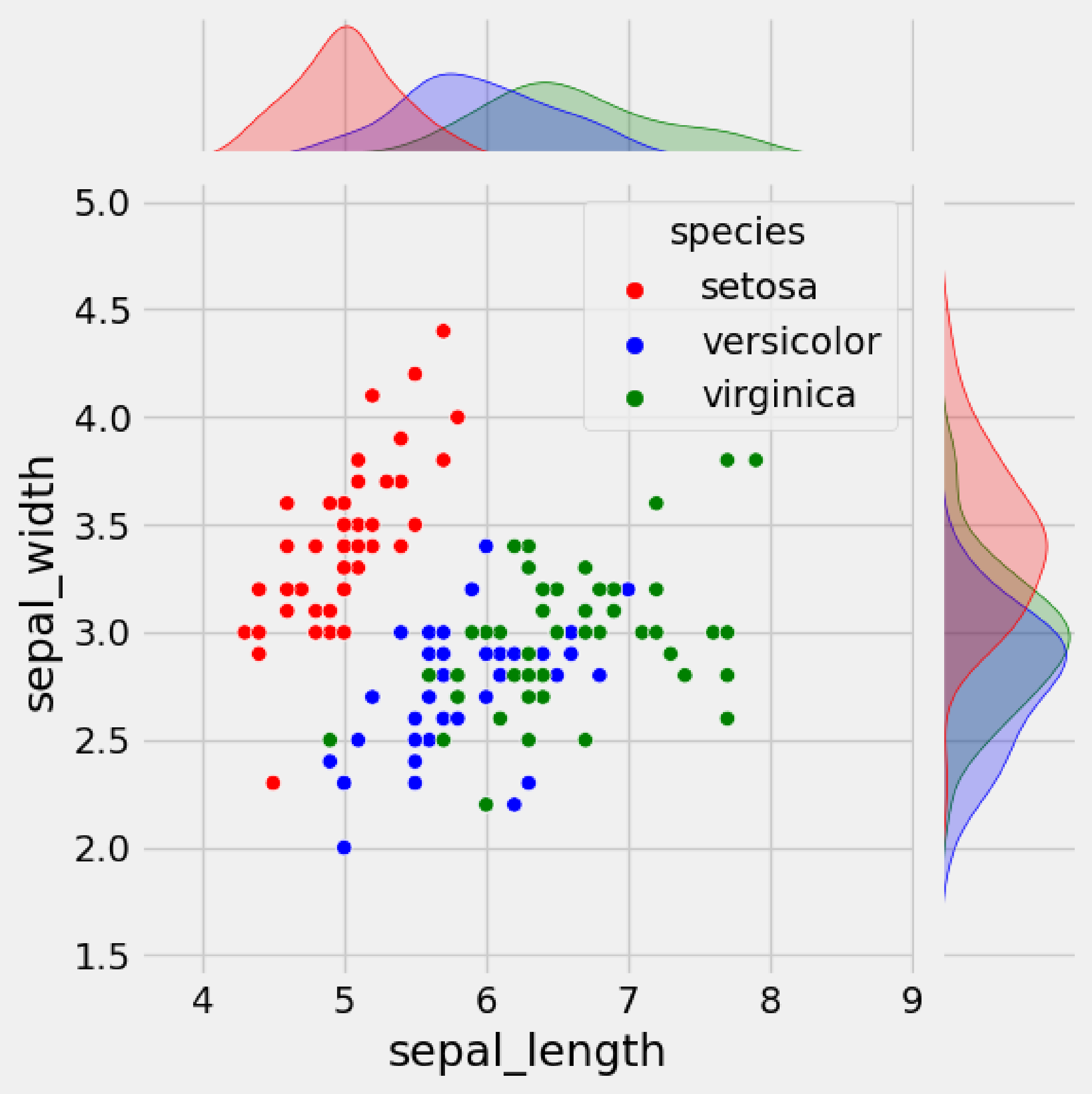
```
C:\Users\anits-csm\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118:
UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```



5.jointplot of sepal length vs sepal width

[47]:

```
sns.jointplot(x='sepal_length', y='sepal_width', data=iris,␣
 ↪kind='scatter',hue='species',palette={'setosa':'red','versicolor':
 ↪'blue','virginica':'green'})
plt.show()
```

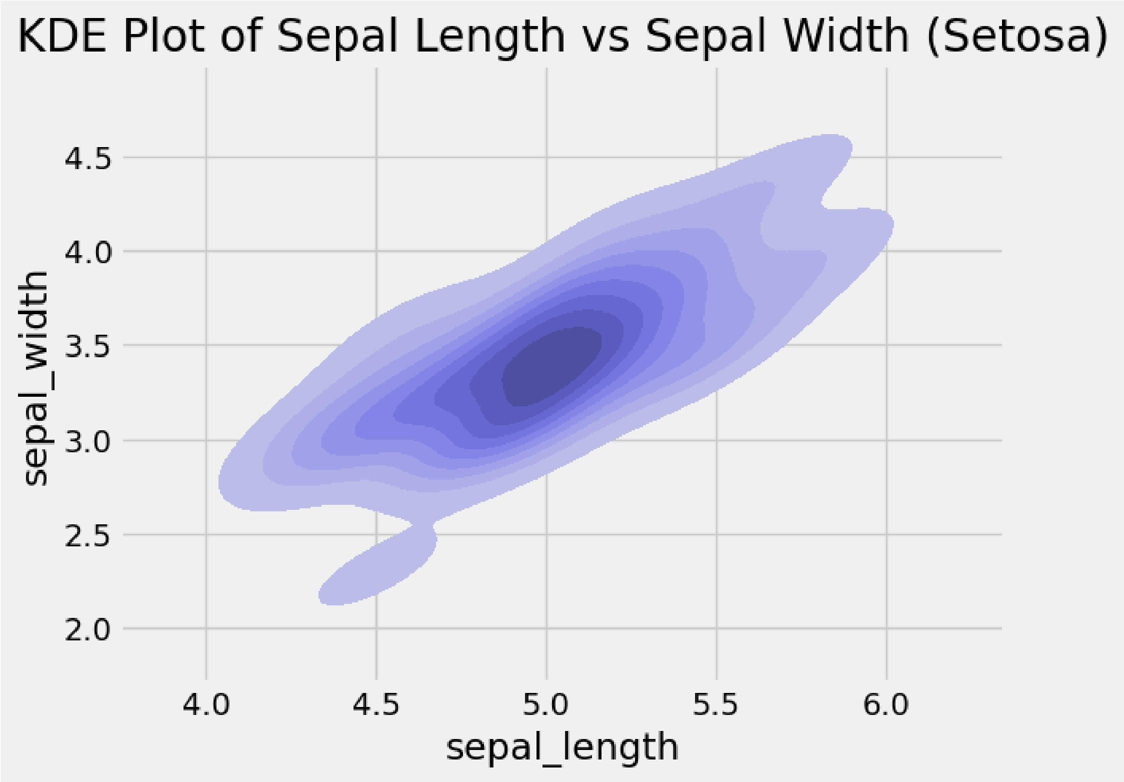


6. KDE Plot for Setosa Species (Sepal Length vs Sepal Width)

[55]:
```
setosa = iris[iris['species'] == 'setosa']
sns.kdeplot(x='sepal_length', y='sepal_width',␣
  ↪data=setosa,shade=True,color='lavender')
plt.title('KDE Plot of Sepal Length vs Sepal Width (Setosa)')
plt.show()
```

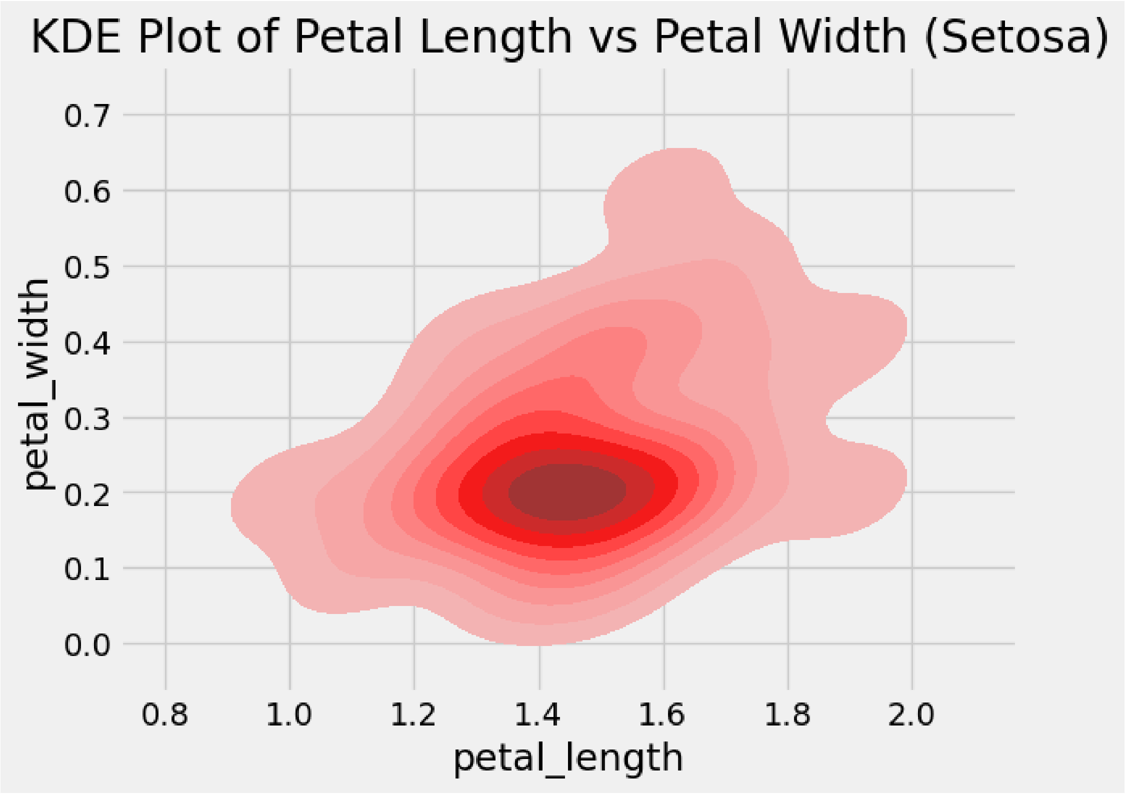
```
C:\Users\anits-csm\AppData\Local\Temp\ipykernel_5912\479660616.py:2:
FutureWarning:

`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(x='sepal_length', y='sepal_width',
data=setosa,shade=True,color='lavender')
```



7. KDE Plot for Setosa Species (Petal Length vs Petal Width):

```
[57]: sns.kdeplot(x='petal_length', y='petal_width',␣
       ↪data=setosa,fill=True,color='red')
      plt.title('KDE Plot of Petal Length vs Petal Width (Setosa)')
      plt.show()
```



0.2.4 Conclusion:

The report highlights the significance of data visualization in simplifying complex datasets and aiding decision-making. It contrasts Matplotlib and Seaborn as essential Python libraries for data visualization, showcasing their key features and applications. Using the Iris dataset, the document illustrates practical visualization techniques such as scatter plots, KDE plots, and pair plots to reveal trends and patterns in data. Overall, it demonstrates how these libraries transform raw data into insightful, visually compelling representations for effective analysis.