data-visualization

September 17, 2024

Data Visualization with Matplotlib and Seaborn using Iris dataset

[2]:
```
import seaborn as sns #import seaborn library
import matplotlib.pyplot as plt #import matplotlib library
import pandas as pd
```

Matplotlib: Matplotlib is a widely-used Python library for creating static, animated, and interactive visualizations. It provides a flexible and comprehensive framework for generating a variety of plots and charts. Its functionality is akin to MATLAB, offering a MATLAB-like interface for plotting data.

Seaborn: Seaborn is a high-level Python data visualization library built on top of Matplotlib. It provides a more user-friendly interface and includes additional features to simplify the creation of complex statistical plots. Seaborn comes with beautiful default styles and color palettes, making it easier to create aesthetically pleasing visualizations.

Iris Dataset: The Iris dataset is a classic dataset in machine learning and statistics, containing measurements of iris flowers. It is widely used for testing algorithms and visualizing data. The dataset includes 150 samples, divided equally among three species of iris flowers: Setosa, Versicolor, and Virginica.

PROGRAM 1: General Statistics Plot (Matplotlib or Seaborn): Write a Python program to create a plot that gives a general statistical summary of the Iris data. You can use seaborn's pairplot or pandas' describe() for guidance.

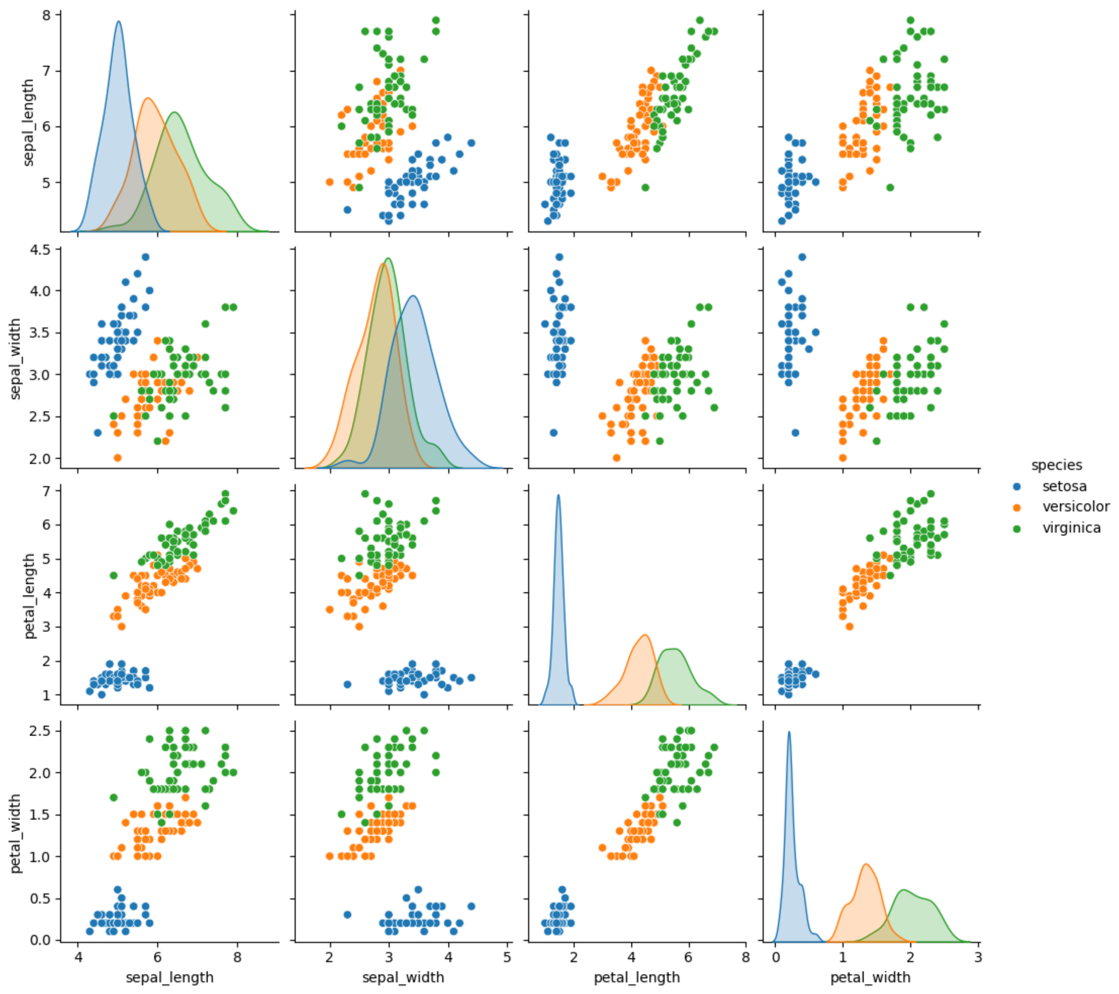
[3]:
```
iris=sns.load_dataset('iris')#loading iris dataset using seaborn
```

[4]:
```
species_counts=iris['species'].value_counts()#identifying the number of species
```

[5]:
```
# Display statistical summary using describe()
print("Statistical Summary:")
print(iris.describe(include='all'))
#Using Pair plot to visualize
sns.pairplot(iris, hue='species', height=2.5)
plt.show()
```

```
Statistical Summary:
        sepal_length  sepal_width  petal_length  petal_width species
count     150.000000   150.000000    150.000000   150.000000     150
unique           NaN          NaN           NaN          NaN       3
```

```
top           NaN         NaN         NaN         NaN   setosa
freq          NaN         NaN         NaN         NaN       50
mean     5.843333    3.057333    3.758000    1.199333      NaN
std      0.828066    0.435866    1.765298    0.762238      NaN
min      4.300000    2.000000    1.000000    0.100000      NaN
25%      5.100000    2.800000    1.600000    0.300000      NaN
50%      5.800000    3.000000    4.350000    1.300000      NaN
75%      6.400000    3.300000    5.100000    1.800000      NaN
max      7.900000    4.400000    6.900000    2.500000      NaN
```



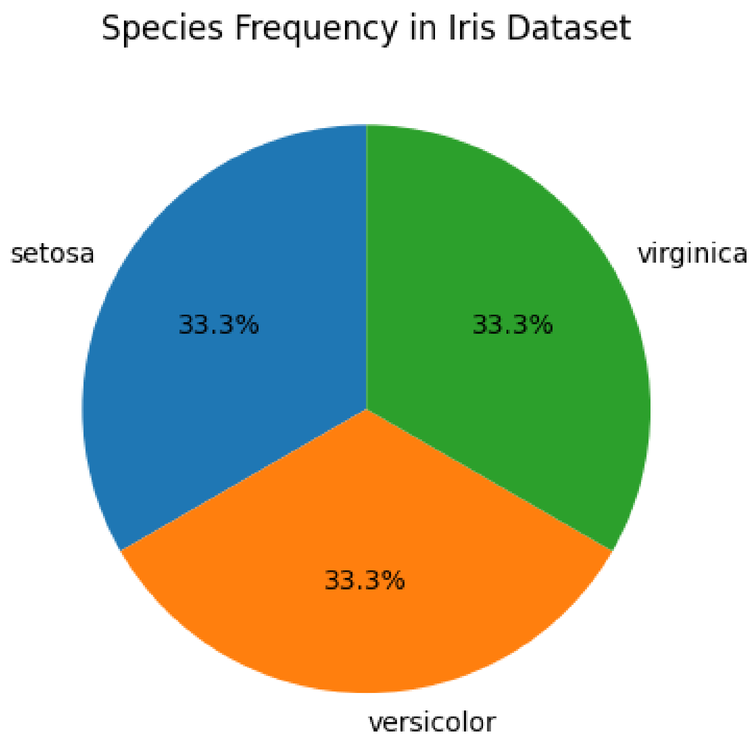
PROGRAM 2:Pie Plot for Species Frequency:Write a Python program to create a pie chart to display the frequency of the three species (setosa, versicolor, virginica) in the Iris dataset.

```
[6]: plt.figure(figsize=(8,6))#setting up the figure size
```

```
[6]: <Figure size 800x600 with 0 Axes>
```

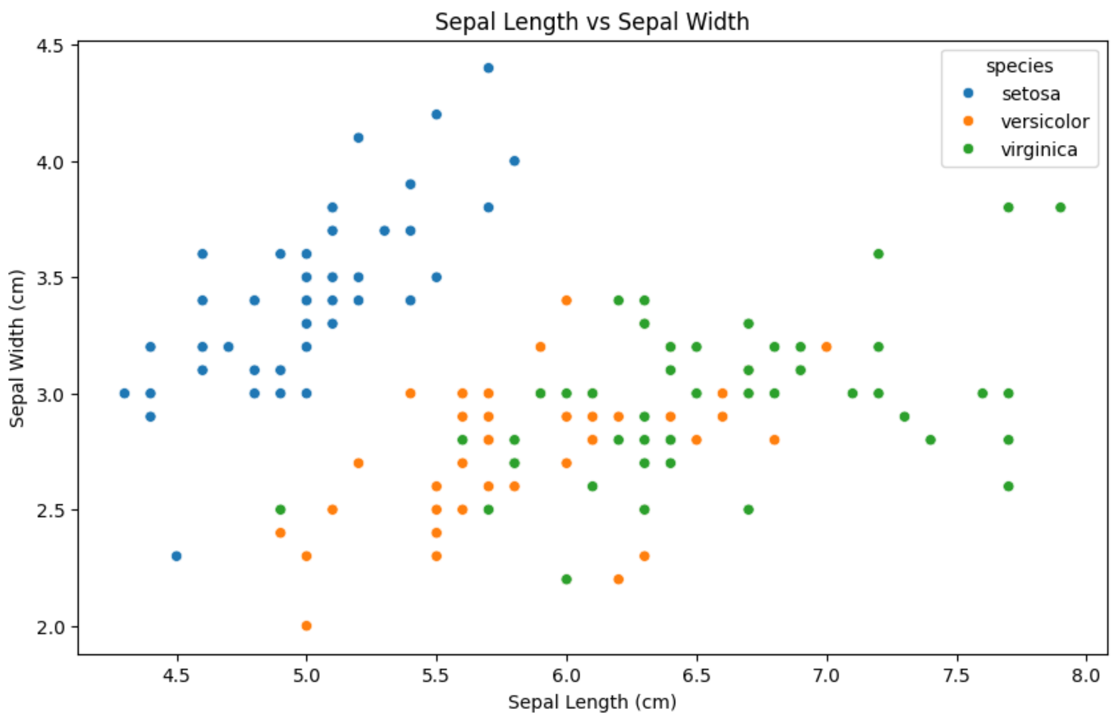
```
<Figure size 800x600 with 0 Axes>
```

[7]:
```
plt.pie(species_counts, labels=species_counts.index, autopct='%1.
  ↪1f%%',startangle=90) #preparing pie chart for the species
plt.title('Species Frequency in Iris Dataset')
plt.show()
```



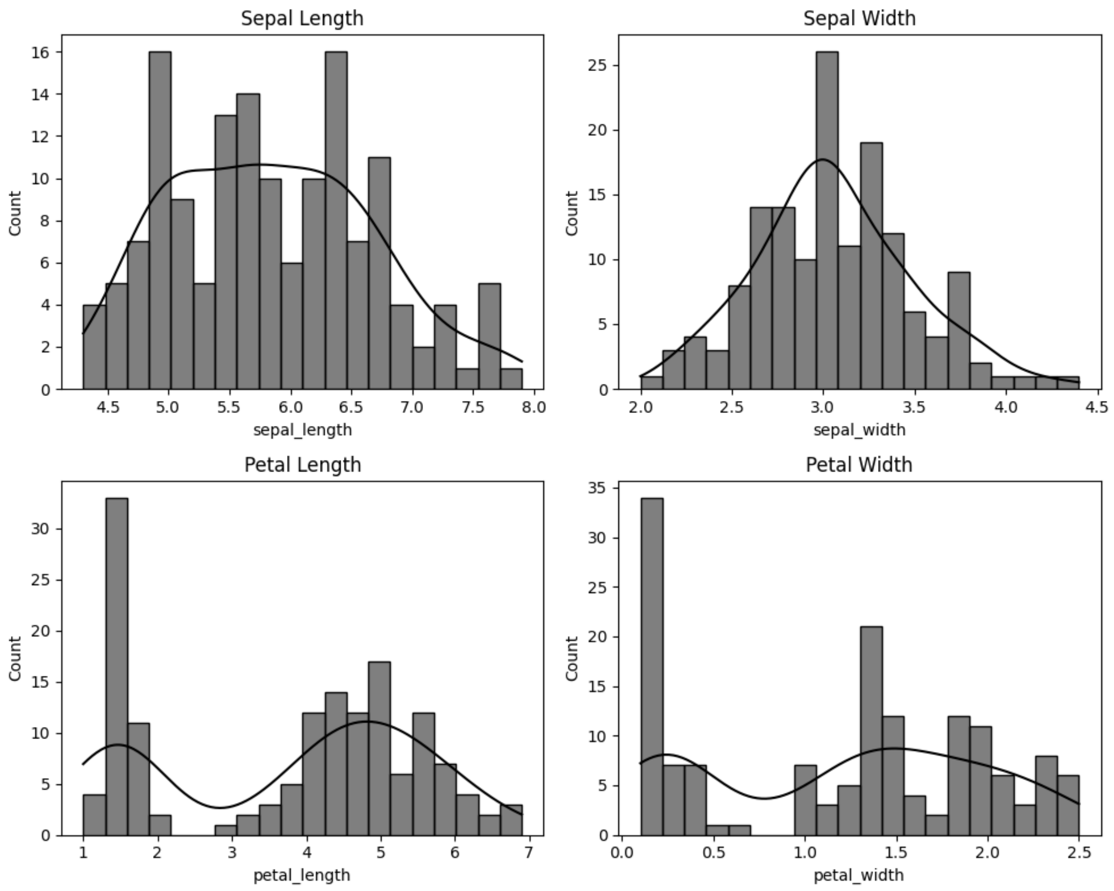
PROGRAM 3:Relationship Between Sepal Length and Width:Write a Python program to create a scatter plot to find the relationship between sepal length and sepal width for the Iris dataset.

[8]:
```
plt.figure(figsize=(10, 6)) #setting the figure size
sns.scatterplot(x='sepal_length', y='sepal_width', hue='species',data=iris)␣
  ↪#setting the x and y axis
plt.title('Sepal Length vs Sepal Width') #setting the title
plt.xlabel('Sepal Length (cm)') #setting the x axis and y axis titles
plt.ylabel('Sepal Width (cm)')
plt.show() #code to display visualization
```



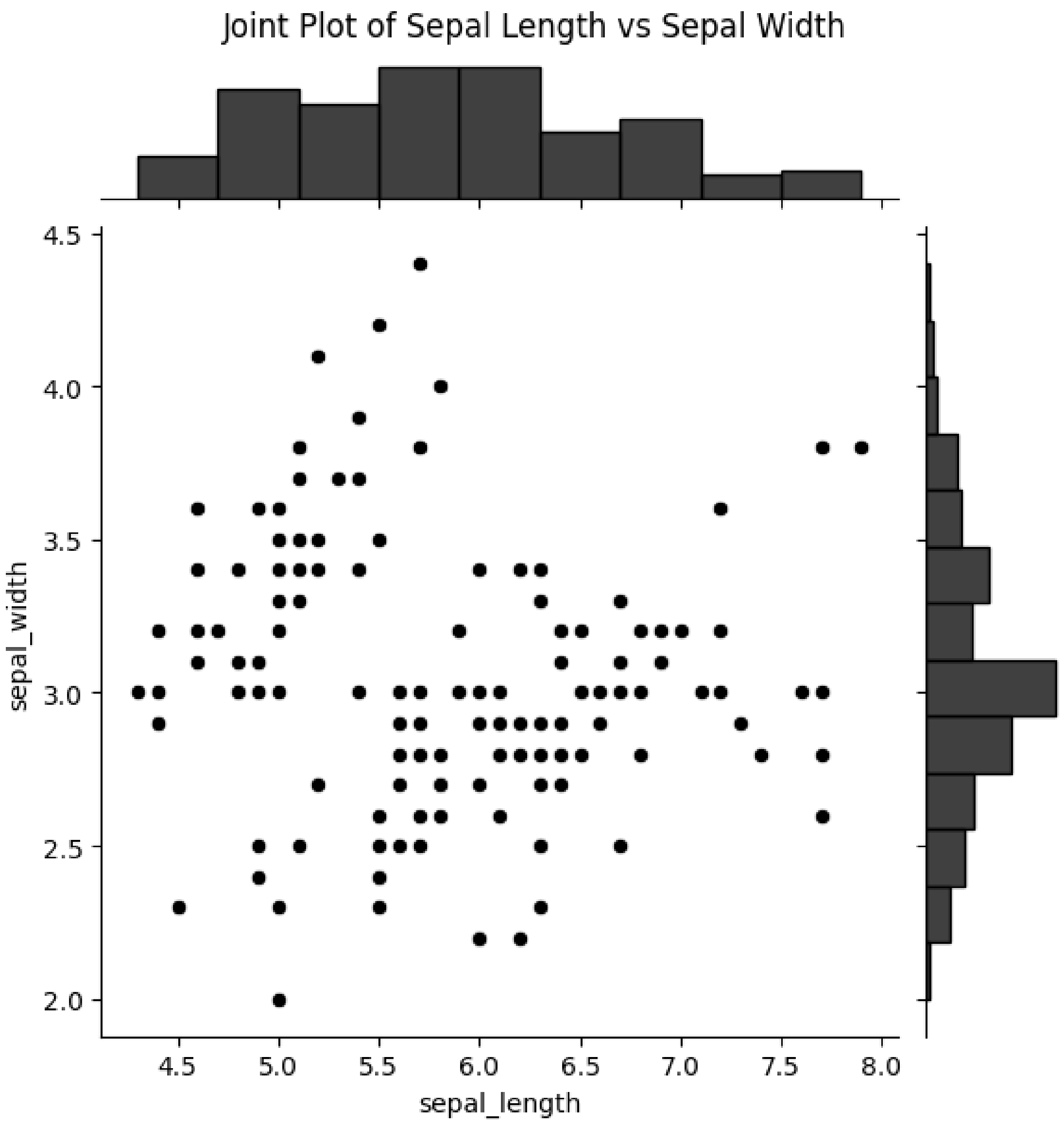
PROGRAM 4:Distribution of Sepal and Petal Features:Write a Python program to create a plot that shows how the length and width of sepal length, sepal width, petal length, and petal width are distributed.

[9]:
```
# Define features and titles
features = ['sepal_length', 'sepal_width', 'petal_length', 'petal_width']
titles = ['Sepal Length', 'Sepal Width', 'Petal Length', 'Petal Width']
# Create a figure with subplots for each feature
fig, axes = plt.subplots(2, 2, figsize=(10, 8))
# Plot histograms with KDE for each feature
for ax, feature, title in zip(axes.flat, features, titles):
    sns.histplot(iris[feature], kde=True, bins=20, ax=ax, color='black')
    ax.set_title(title)
# Adjust layout and show plot
plt.tight_layout()
plt.show()
```



PROGRAM 5:Jointplot of Sepal Length vs Sepal Width:Write a Python program to create a joint plot to describe the individual distributions on the same plot between sepal length and sepal width.

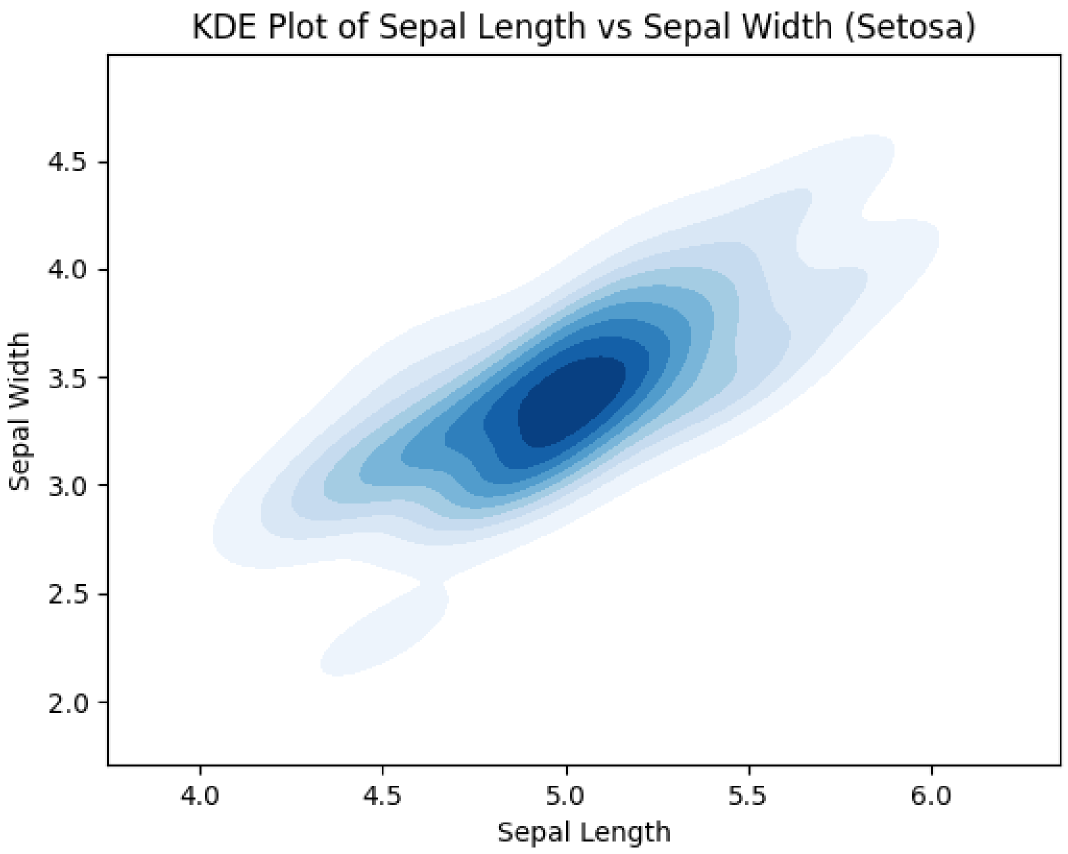
```
[10]: #Create a joint plot for Sepal Length vs Sepal Width
sns.jointplot(data=iris, x='sepal_length', y='sepal_width', kind='scatter',␣
 ↪color='black')
# Add title to the plot
plt.suptitle('Joint Plot of Sepal Length vs Sepal Width', y=1.02)
# Show the plot
plt.show()
```



PROGRAM 6:KDE Plot for Setosa Species (Sepal Length vs Sepal Width):Write a Python program using seaborn to create a KDE (Kernel Density Estimate) plot of sepal length versus sepal width for the setosa species of the Iris dataset.

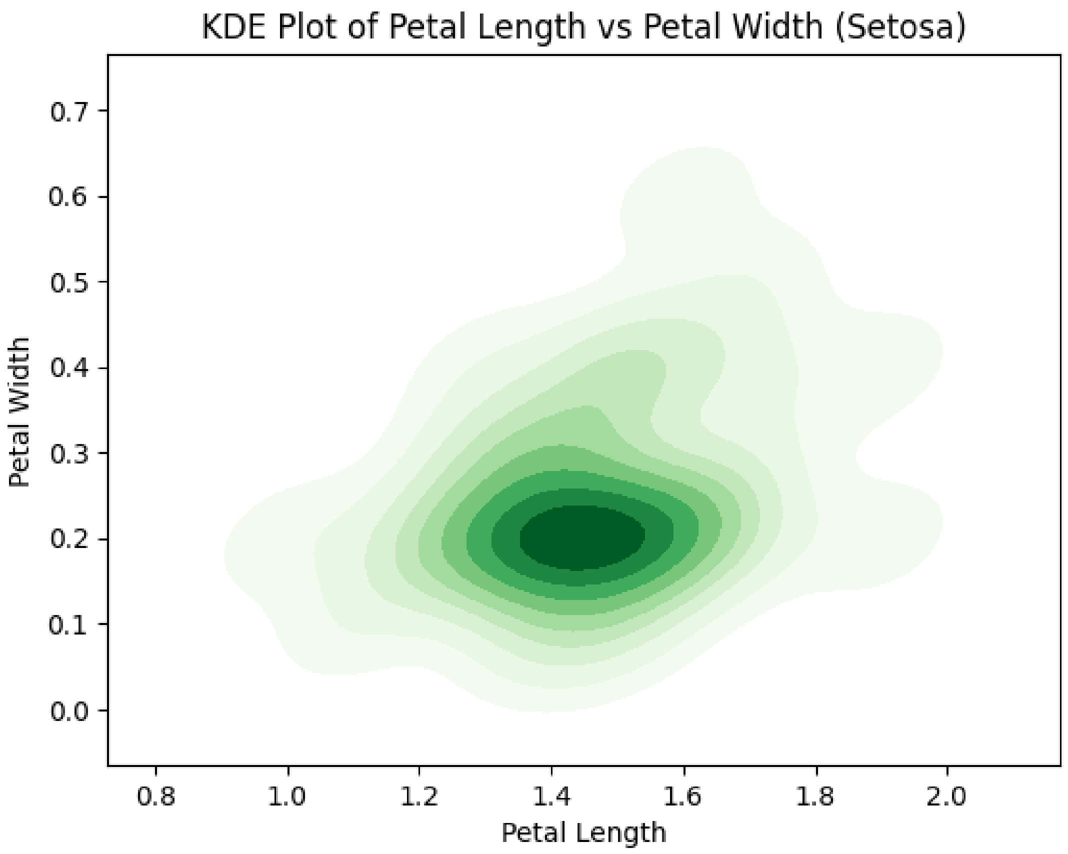
[11]:
```python
setosa = iris[iris['species'] == 'setosa']
# Create a KDE plot for Sepal Length vs Sepal Width
sns.kdeplot(data=setosa, x='sepal_length', y='sepal_width', fill=True,␣
 ↪cmap='Blues')
# Add title and labels
plt.title('KDE Plot of Sepal Length vs Sepal Width (Setosa)')
plt.xlabel('Sepal Length')
plt.ylabel('Sepal Width')
```

```
# Show the plot
plt.show()
```



PROGRAM 7:KDE Plot for Setosa Species (Petal Length vs Petal Width):Write a Python program using seaborn to create a KDE plot of petal length versus petal width for the setosa species.

[13]:
```
setosa = iris[iris['species'] == 'setosa']#setting the column with setosa␣
↪species
# Create a KDE plot for Petal Length vs Petal Width
sns.kdeplot(data=setosa, x='petal_length', y='petal_width', fill=True,␣
↪cmap='Greens')
# Add title and labels
plt.title('KDE Plot of Petal Length vs Petal Width (Setosa)')
plt.xlabel('Petal Length')
plt.ylabel('Petal Width')
# Show the plot
plt.show()
```


KDE Plot of Petal Length vs Petal Width (Setosa)
Petal Width
Petal Length
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
1.0
1.2
1.4
1.6
1.8
2.0

```
[ ]:
```