Module 3: Data Analytics With Python - Applied Statistics Lab 1: Case Study: Exploratory Data Analytics (EDA)

What is EDA?

Exploratory Data Analysis refers to the critical process of performing initial investigations on data to discover patterns, to spot anomalies, to test hypothesis and to check assumptions with the help of summary statistics and graphical representations.

It is a good practice to understand the data first and try to gather as many insights as possible from it. EDA is all about making sense of data in hand, before getting them dirty with it.

EDA helps determine how best to manipulate data sources to get the answers you need, making it easier for data scientists to discover patterns, spot anomalies, test a hypothesis, or check assumptions.

Objective

- Performing Exploratory Data Analysis
- Exploratory Data Analysis
    - General information on the dataset
    - Univariate Analysis
    - Bivariate Analysis
    - Multivariate Analysis
    - Conclusion

Case Study: IRIS dataset

Exploratory Analytics

The Iris dataset was used in R.A. Fisher's classic 1936 paper, The Use of Multiple

Measurements in Taxonomic Problems, and can also be found on the UCI Machine Learning Repository.
It includes **three iris species** with **50 samples** each as well as some properties about each flower. One flower species is linearly separable from the other two, but the other two are not linearly separable from each other.

Dataset

- the **ID** column
- 4 columns of measures on Sepal and Petal : **SepalLengthCm, SepalWidthCm, PetalLengthCm, PetalWidthCm**
- the column containing the labels : **Iris-setosa, Iris-versicolor, Iris-virginica**

We are going to perform an **exploratory data analysis** to understand the data and choose the best features.

What will we do?

Observe the data.

1. Look out for missing values and outliers.
2. Perform Descriptive Analysis
3. Perform univariate, bivariate, and multivariate analysis.
4. We will use swarm plots, box plots, histograms and KDEs.

```
# Importing Data processing libraries
import numpy as np
import pandas as pd
pd.plotting.register_matplotlib_converters()

# Plotting Libraries
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
sns.set(style="whitegrid")

# Reading Data
# Download the dataset from here -

iris=pd.read_csv("https://raw.githubusercontent.com/bluedataconsulting/AIMasteryProgram/main/Lab_Exercises/Module3/iris_data.csv")

iris.head()

 # Confirming the number of records for each species iris['class'].value_counts()
```

Exploratory Data Analysis

General information on the dataset

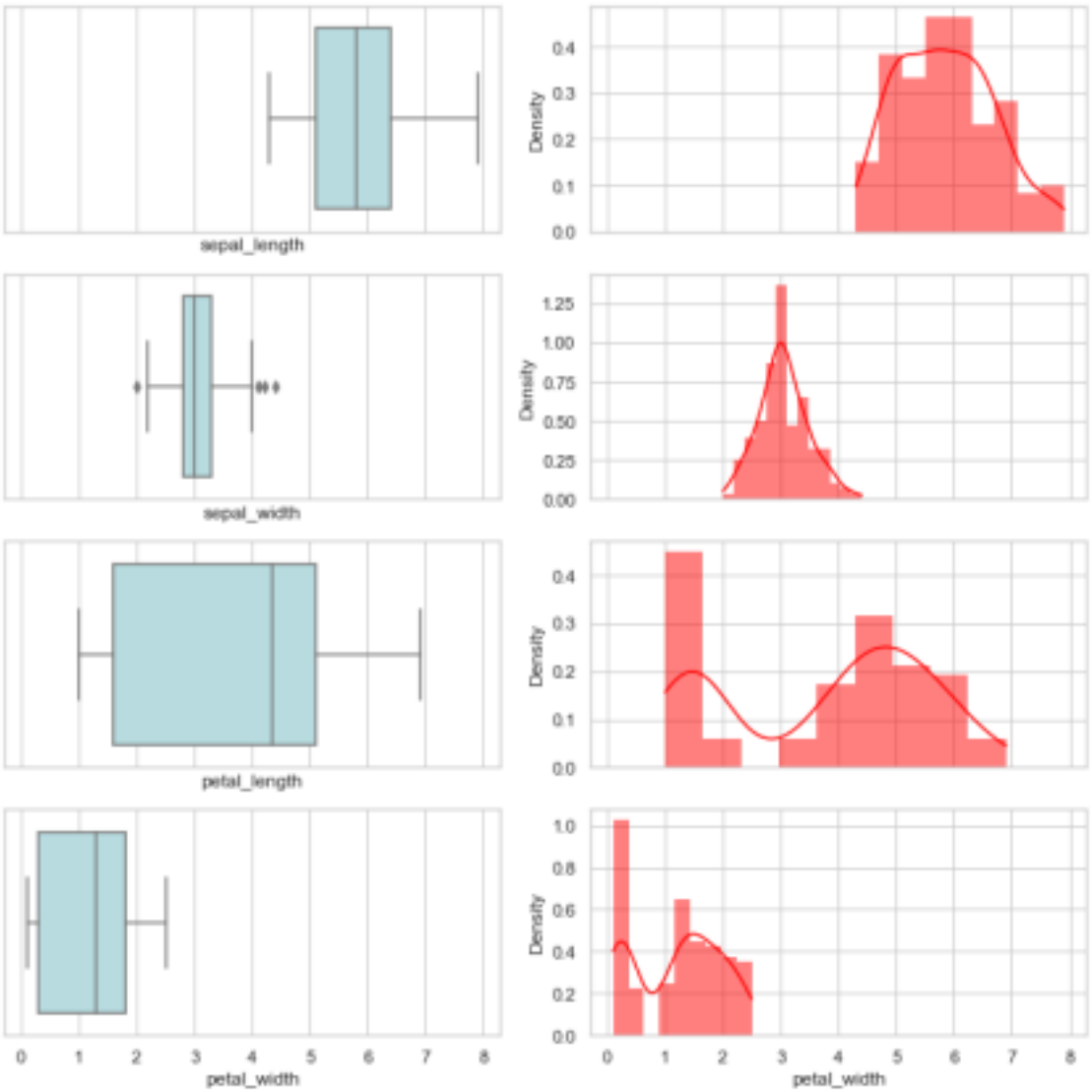
```
# Viewing Data
iris.head()

 # Shape of Data
iris.shape

# Finding null count
iris.info()

# Descriptive analysis
iris.describe()
```

Univariate Analysis

```
# Distinct Species values
iris["class"].unique()

array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'],  dtype=object)

# Dividing data
setosa = iris.loc[iris['class'] == "Iris-setosa"]
versicolor = iris.loc[iris['class'] == "Iris-versicolor"]
virginica = iris.loc[iris['class'] == "Iris-virginica"]

iris.columns
```

'sepal_length', 'sepal_width', 'petal_length', 'petal_width'

```
# Setting up subplots
f,axes=plt.subplots(ncols=2,nrows=4,figsize=(10,10),sharex=True)      l=['sepal_length',
'sepal_width', 'petal_length', 'petal_width'] for i in range(4):

 #Ploting the boxplot
 sns.boxplot(x = iris.loc[:,l[i]], ax=axes[i][0],  color='powderblue')

 # Ploting the KDE
 sns.histplot(data=iris.loc[:,l[i]], color="red", kde=True,  stat="density", linewidth=0,
ax=axes[i][1])

 plt.tight_layout()
```



Bivariate Data Analysis

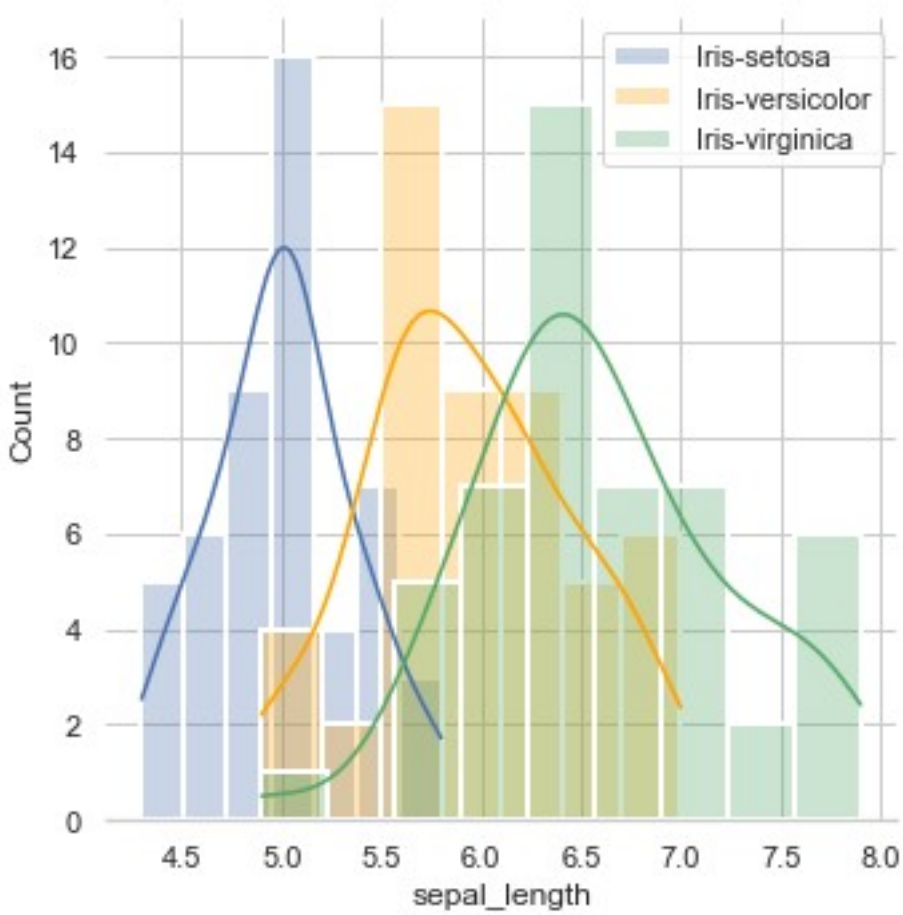
- sepal_length vs class
- sepal_width vs class
- petal_length vs class
- petal_width vs class

sepal_length vs class

```
# Set up the matplotlib figure
f, axes = plt.subplots(ncols = 1,figsize=(5, 5), sharex=True)
sns.despine(left=True)#sepa l_length-vs-class
```

 # Ploting the histogram with KDE

```
sns.histplot(data=setosa["sepal_length"],label="Iris
setosa",color='b',kde=True,linewidth=2,alpha=0.3)
sns.histplot(data=versicolor["sepal_length"],label="Iris
versicolor",kde=True,color='orange',linewidth=2,alpha=0.3)
sns.histplot(data=virginica["sepal_length"],label="Iris
virginica",kde=True,color='g',linewidth=2,alpha=0.3)

plt.legend()
plt.tight_layout()
```

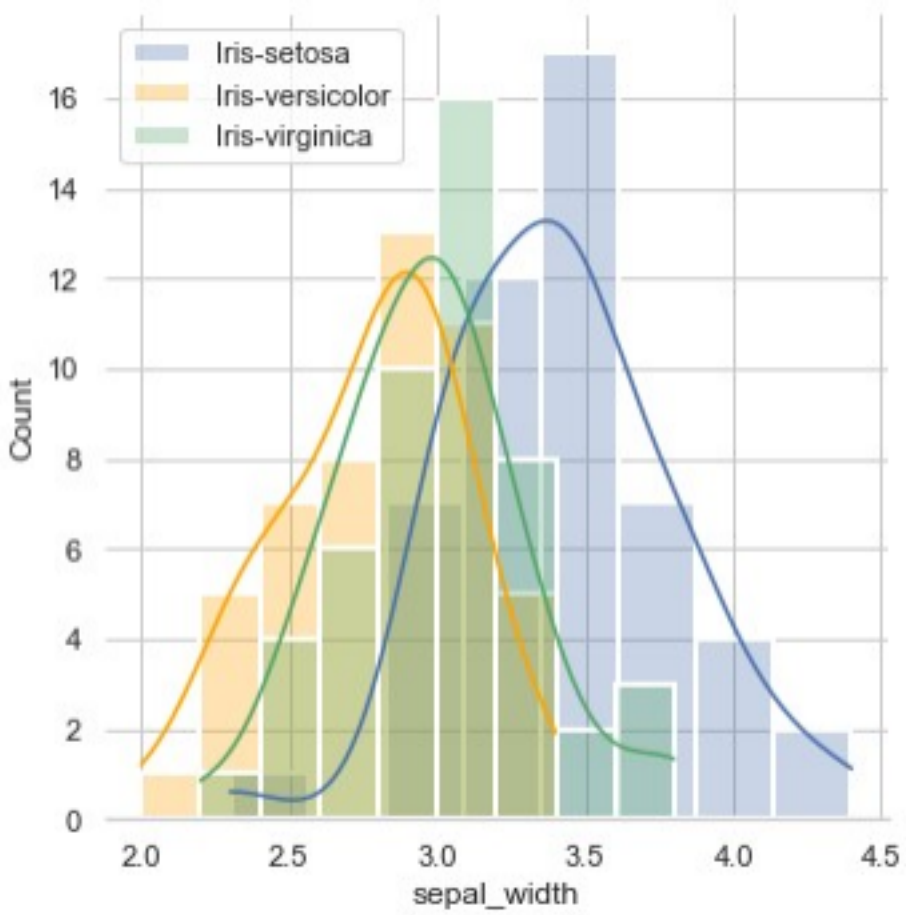


- sepal_width vs class

 # Set up the matplotlib figure

```
f, axes = plt.subplots(ncols = 1,figsize=(5, 5), sharex=True) sns.despine(left=True)
sns.histplot(data=setosa["sepal_width"],label="Iris
setosa",color='b',kde=True,linewidth=2,alpha=0.3)
sns.histplot(data=versicolor["sepal_width"],label="Iris
versicolor",kde=True,color='orange',linewidth=2,alpha=0.3)
sns.histplot(data=virginica["sepal_width"],label="Iris-
virginica",kde=True,color='g',linewidth=2,alpha=0.3)
```

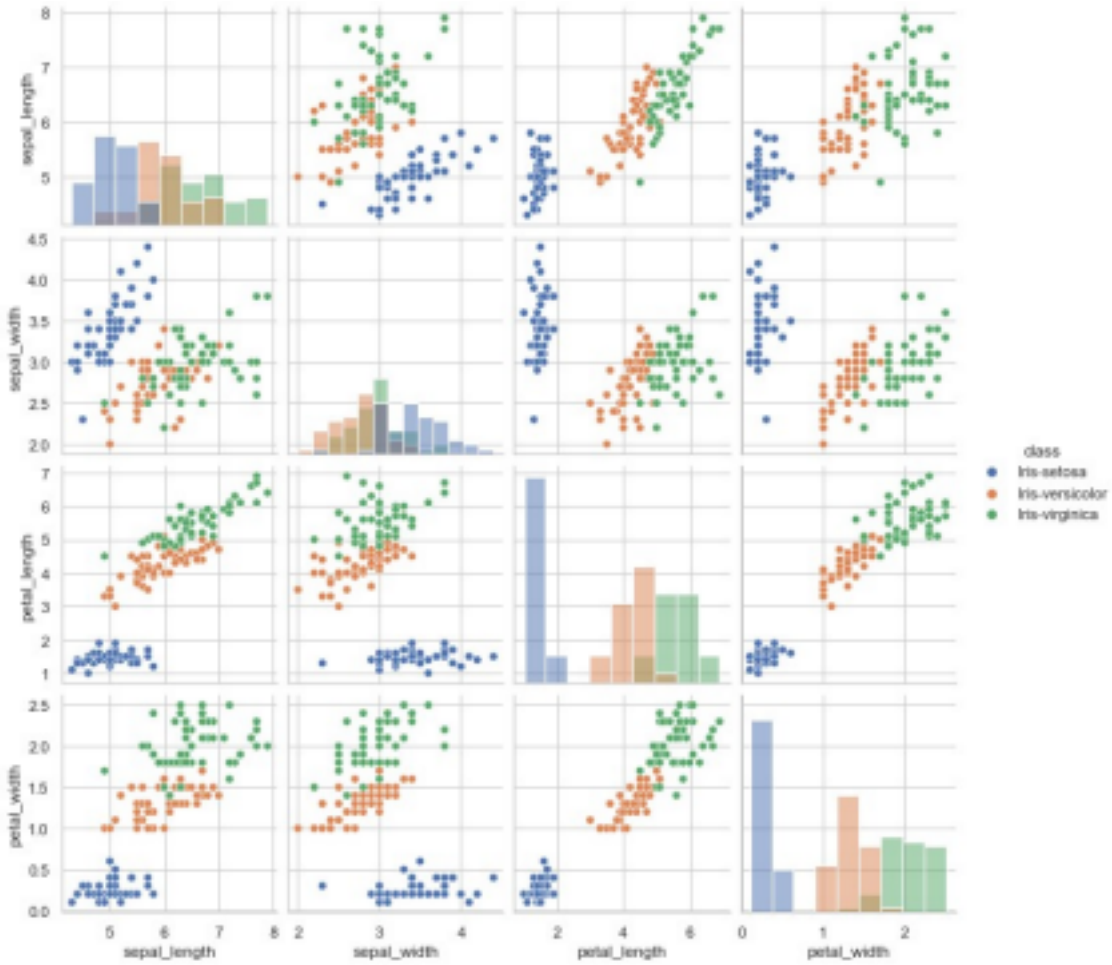
```
plt.legend()
plt.tight_layout()
```



Multivariate Analysis

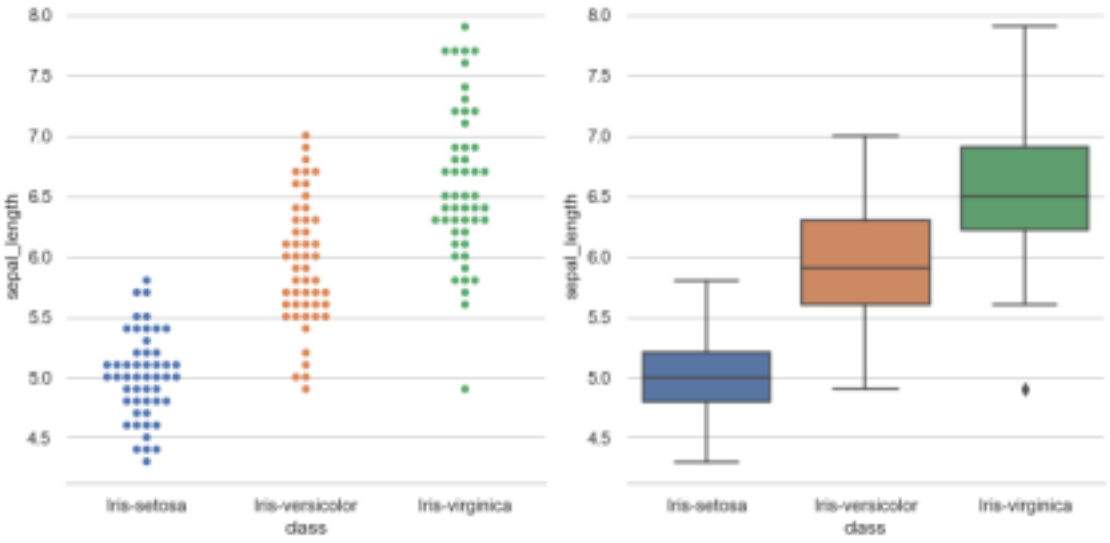
All species vs All Species

```
sns.pairplot(iris, hue="class", diag_kind="hist") plt.show()
```

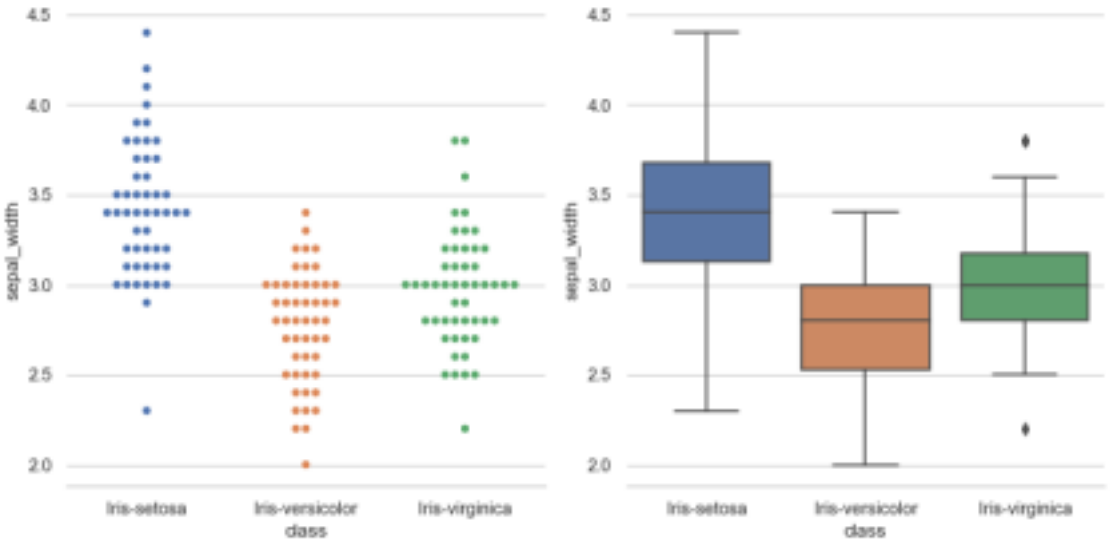


Plotting Swarm and Box Plots

```
# Set up the matplotlib figure
f, axes = plt.subplots(ncols = 2, figsize=(10, 5), sharex=True) sns.despine(left=True)

# Plot the Swarmplot
sns.swarmplot(x=iris['class'], y=iris['sepal_length'], ax=axes[0])

# Plot the Boxplot
sns.boxplot(x=iris['class'], y=iris['sepal_length'], ax=axes[1]) plt.tight_layout()
```

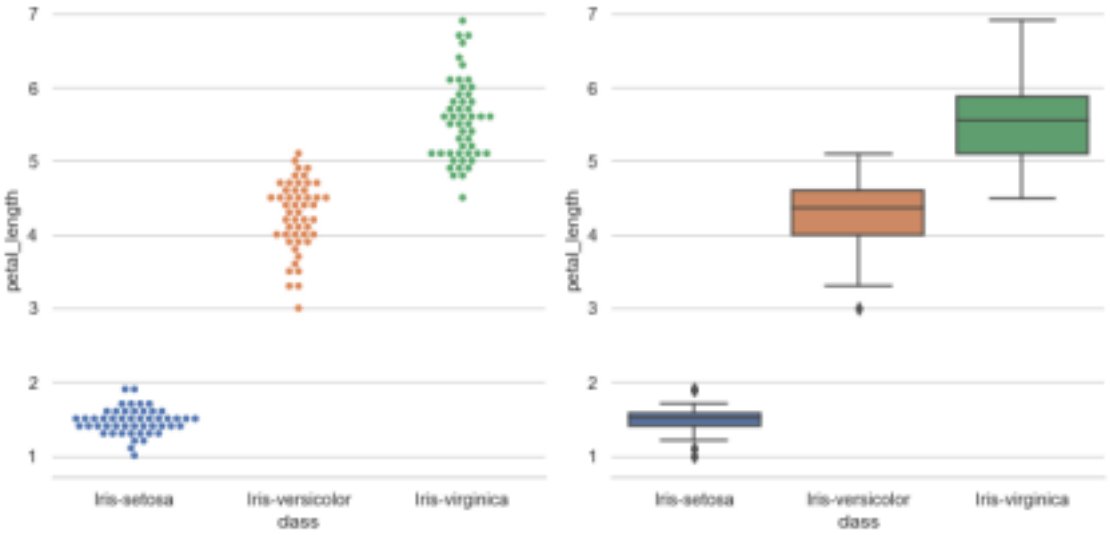



```
# Set up the matplotlib figure
f, axes = plt.subplots(ncols = 2, figsize=(10, 5), sharex=True) sns.despine(left=True)

# Plot the Swarmplot
sns.swarmplot(x=iris['class'], y=iris['sepal_width'], ax=axes[0])

# Plot the Boxplot
sns.boxplot(x=iris['class'], y=iris['sepal_width'], ax=axes[1]) plt.tight_layout()
```

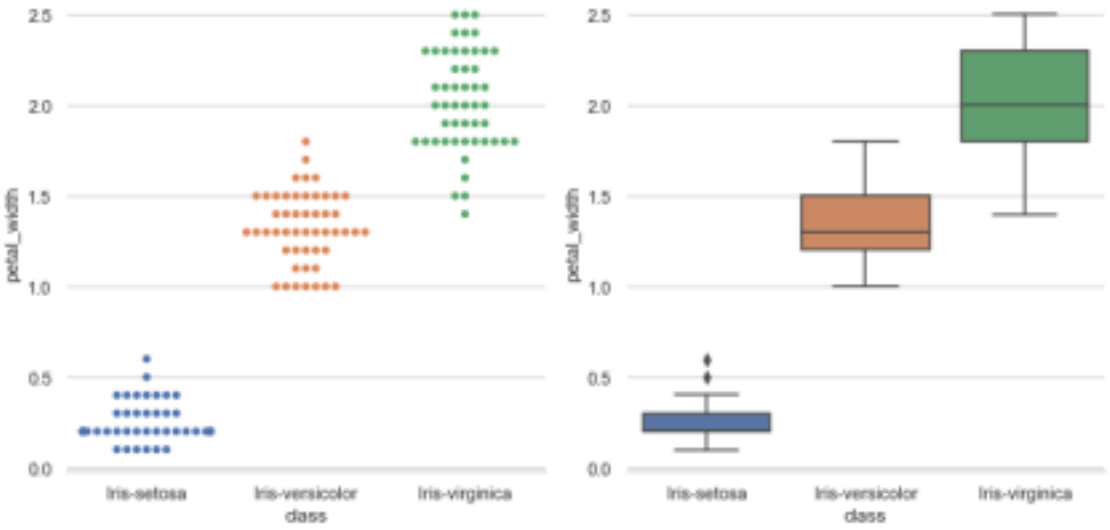



```
# Set up the matplotlib figure
f, axes = plt.subplots(ncols = 2, figsize=(10, 5), sharex=True) sns.despine(left=True)

# Plot the Swarmplot
```

```
sns.swarmplot(x=iris['class'], y=iris['petal_length'],  ax=axes[0] ,size=4.5)

# Plot the Boxplot
sns.boxplot(x=iris['class'], y=iris['petal_length'], ax=axes[1]) plt.tight_layout()
```




```
# Set up the matplotlib figure
f, axes = plt.subplots(ncols = 2, figsize=(10,5), sharex=True) sns.despine(left=True)

# Plot the Swarmplot
sns.swarmplot(x=iris['class'], y=iris['petal_width'], ax=axes[0])

# Plot the Boxplot
sns.boxplot(x=iris['class'], y=iris['petal_width'], ax=axes[1])

plt.tight_layout()
```



End of EDA