MACHINE LEARNING

2024 – 2025

SUBMITED TO :
ENGINNERING COLLEGE AJMER
UNDER THE GUIDENCE OF
MR : VISHNU PRAKASH
SHARMA

What is ML?

Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that makes it more similar to humans: The ability to learn.

What is Exploratory Data Analysis (EDA)?

Exploratory Data Analysis (EDA) is a crucial initial step in data science projects. It involves analyzing and visualizing data to understand its key characteristics, uncover patterns, and identify relationships between variables refers to the method of studying and exploring record sets to apprehend their predominant traits, discover patterns, locate outliers, and identify relationships between variables. EDA is normally carried out as a preliminary step before undertaking extra formal statistical analyses or modeling.

Key aspects of EDA include:

- **Distribution of Data: Examining the distribution of data points to understand their range, central tendencies (mean, median), and dispersion (variance, standard deviation).**
- **Graphical Representations: Utilizing charts such as histograms, box plots, scatter plots, and bar charts to visualize relationships within the data and distributions of variables.**
- **Outlier Detection: Identifying unusual values that deviate from other data points. Outliers can influence statistical analyses and might indicate data entry errors or unique cases.**

- Correlation Analysis: Checking the relationships between variables to understand how they might affect each other. This includes computing correlation coefficients and creating correlation matrices.
- Handling Missing Values: Detecting and deciding how to address missing data points, whether by imputation or removal, depending on their impact and the amount of missing data.
- Summary Statistics: Calculating key statistics that provide insight into data trends and nuances.
- Testing Assumptions: Many statistical tests and models assume the data meet certain conditions (like normality or homoscedasticity). EDA helps verify these assumptions.

```
import pandas as pd
# Mount Google Drive
from google.colab import drive
drive.mount('/content/drive')

df = pd.read_csv("/content/drive/MyDrive/Iris.csv")
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount,
```

+ C

Double-click (or enter) to edit

[1]
```
!pwd
```

```
/content
```

```
[ ] !ls /content/drive/MyDrive/Iris.csv
```

```
/content/drive/MyDrive/Iris.csv
```

```
[ ] import pandas as pd
    df=pd.read_csv("/content/drive/MyDrive/Iris.csv")
```

Double-click (or enter) to edit

```
# importting Libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings as wr
wr.filterwarnings('ignore')
```

```
print(df.head())
```

```
   Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm      Species
0   1            5.1           3.5            1.4           0.2  Iris-setosa
1   2            4.9           3.0            1.4           0.2  Iris-setosa
2   3            4.7           3.2            1.3           0.2  Iris-setosa
3   4            4.6           3.1            1.5           0.2  Iris-setosa
4   5            5.0           3.6            1.4           0.2  Iris-setosa
```

```
[ ] df.shape
```

```
(150, 6)
```

```python
#data information
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Id             150 non-null    int64  
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    object 
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

```
df.describe()
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
min	1.000000	4.300000	2.000000	1.000000	0.100000
25%	38.250000	5.100000	2.800000	1.600000	0.300000
50%	75.500000	5.800000	3.000000	4.350000	1.300000
75%	112.750000	6.400000	3.300000	5.100000	1.800000
max	150.000000	7.900000	4.400000	6.900000	2.500000

```
[ ]  #column to list
     df.columns.tolist()
```

```
['Id',
 'SepalLengthCm',
 'SepalWidthCm',
 'PetalLengthCm',
 'PetalWidthCm',
 'Species']
```

```
[ ]  # check for missing values:
     df.isnull().sum()
```

```
[ ] # check for missing values:
    df.isnull().sum()
```

	0
Id	0
SepalLengthCm	0
SepalWidthCm	0
PetalLengthCm	0
PetalWidthCm	0
Species	0

dtype: int64

```python
#checking duplicate values
df.nunique()
```

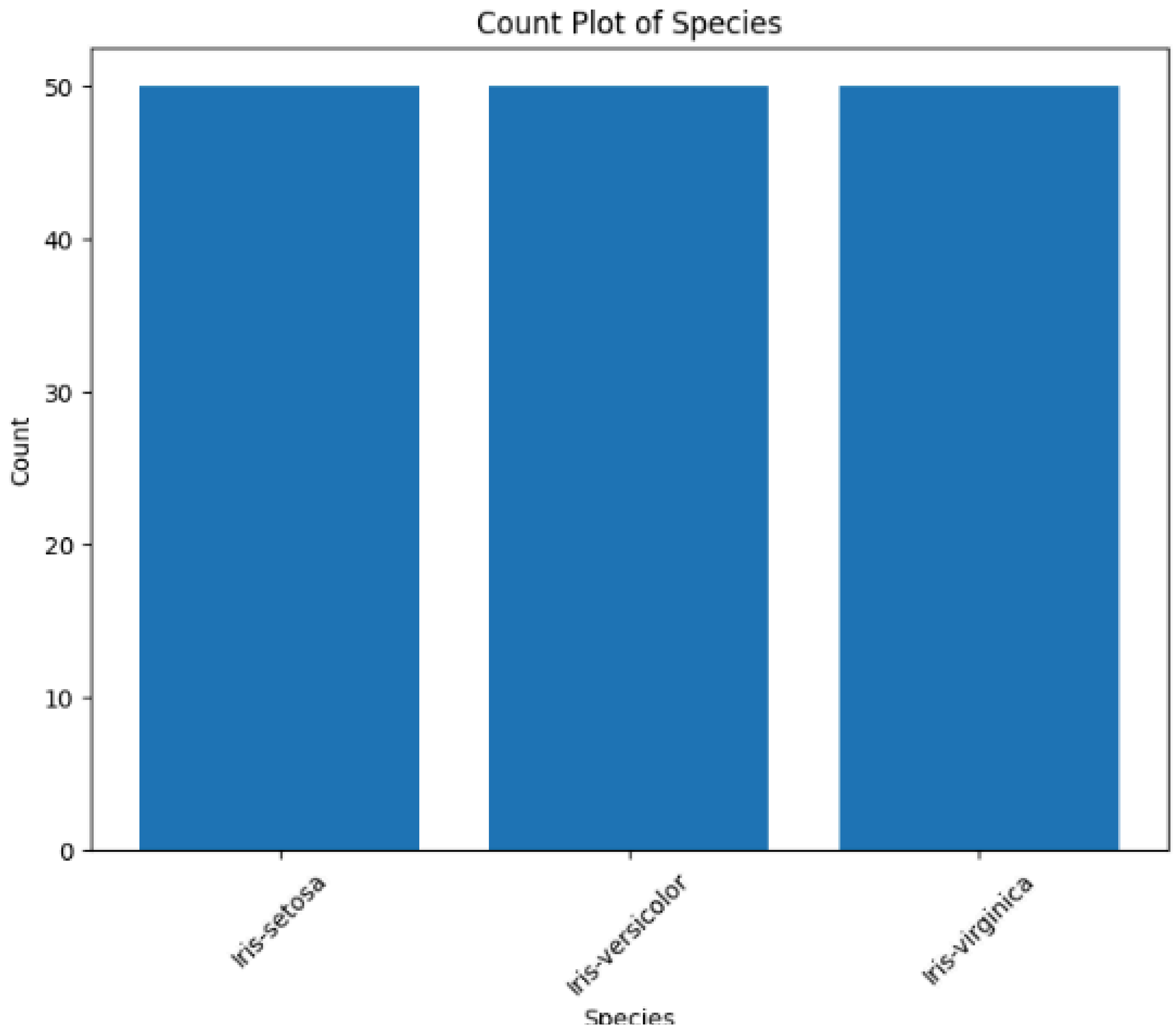
	0
Id	150
SepalLengthCm	35
SepalWidthCm	23
PetalLengthCm	43
PetalWidthCm	22
Species	3

dtype: int64

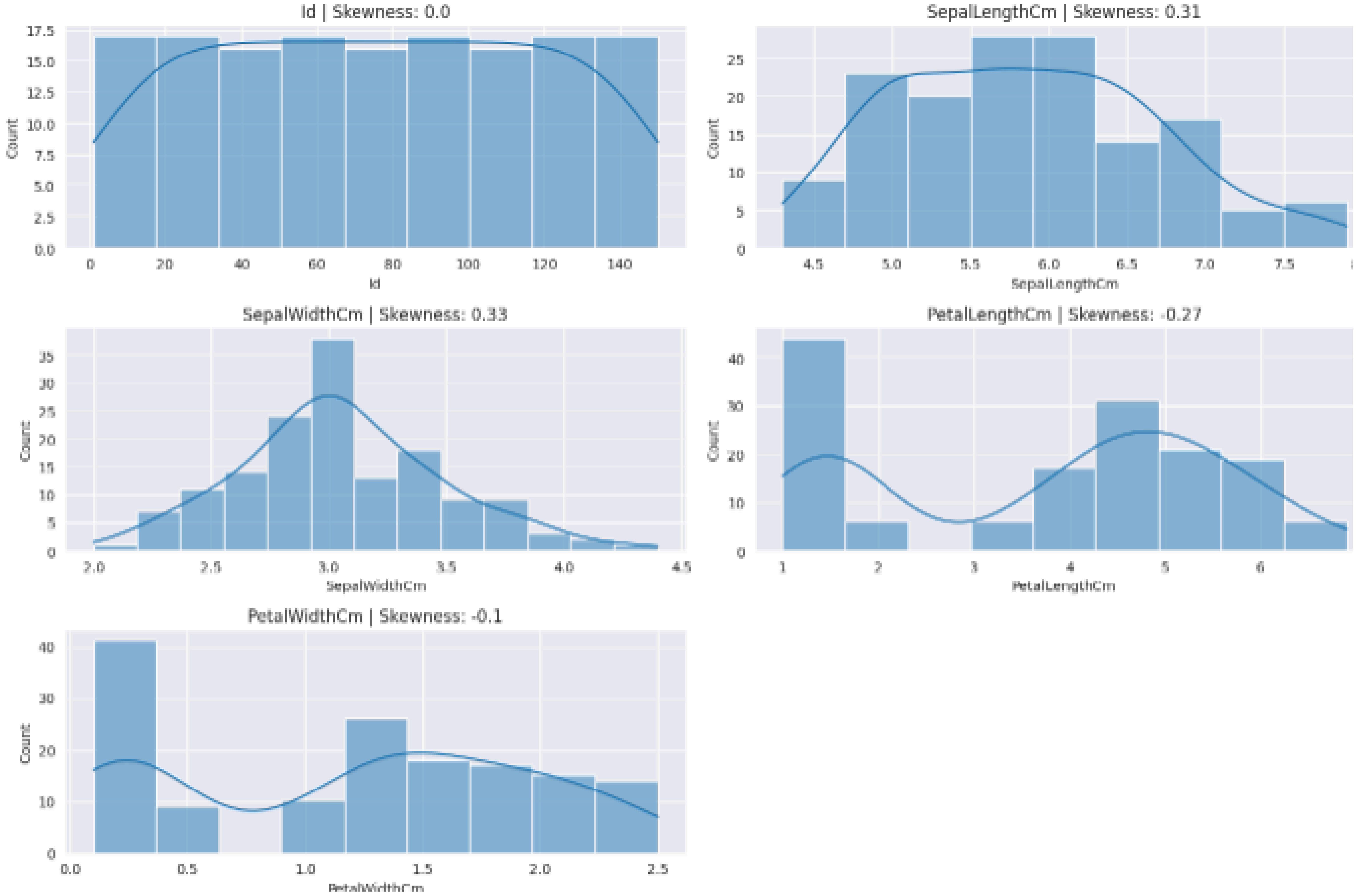
```python
import matplotlib.pyplot as plt
import seaborn as sns
```

```python
import matplotlib.pyplot as plt
import seaborn as sns
```

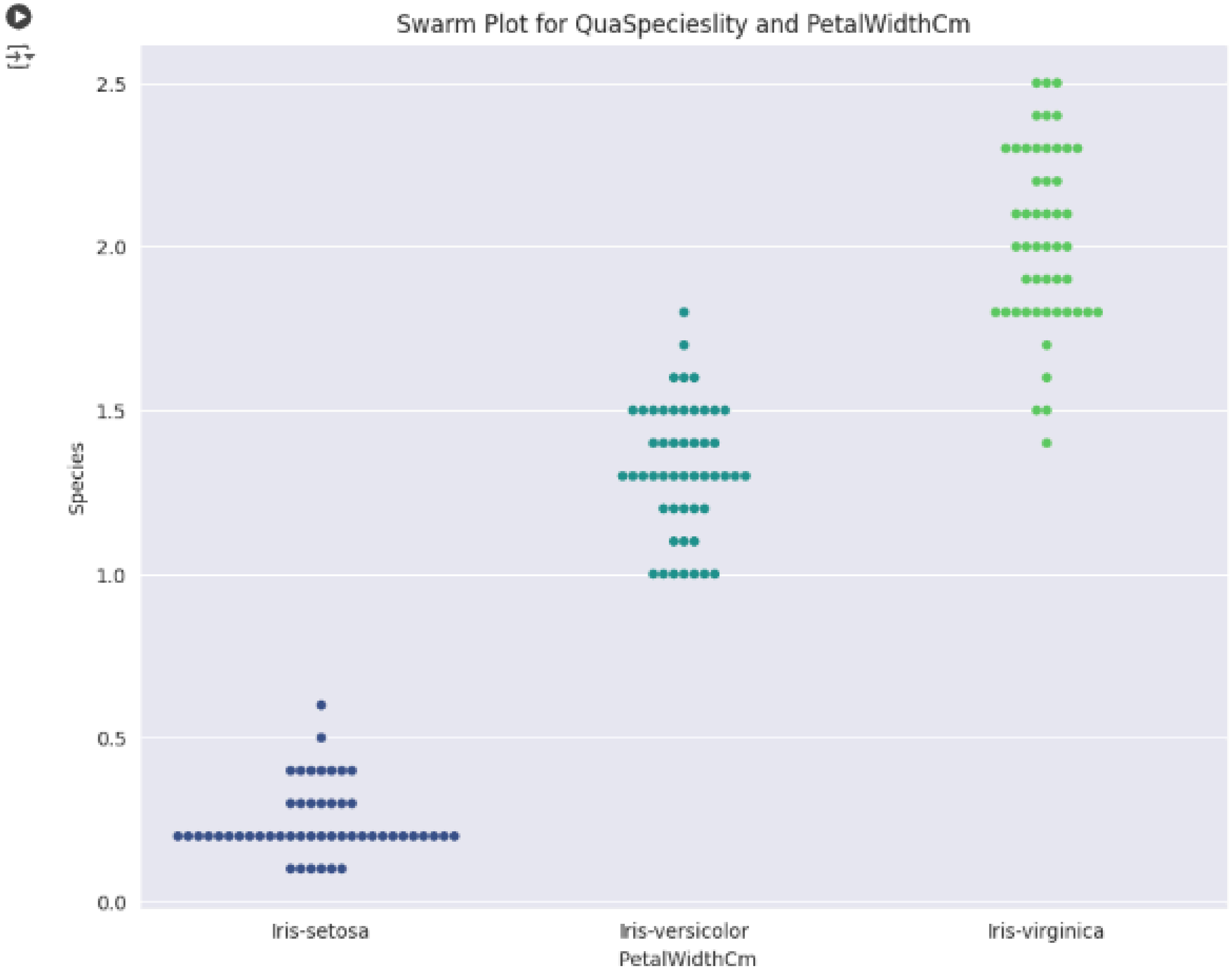
```python
[ ] import matplotlib.pyplot as plt

    # Assuming 'df' is your DataFrame
    quality_counts = df['Species'].value_counts()

    # Using Matplotlib to create a count plot
    plt.figure(figsize=(8, 6))
    plt.bar(quality_counts.index, quality_counts)
    plt.title('Count Plot of Species')
    plt.xlabel('Species')
    plt.ylabel('Count')
    plt.xticks(rotation=45)  # Optional: Rotate x labels for better readability
    plt.show()
```


Count Plot of Species
Count
50
40
30
20
10
0
Iris-setosa
Iris-versicolor
Iris-virginica
Species

```python
# Set Seaborn style
sns.set_style("darkgrid")

# Identify numerical columns
numerical_columns = df.select_dtypes(include=["int64", "float64"]).columns

# Plot distribution of each numerical feature
plt.figure(figsize=(14, len(numerical_columns) * 3))
for idx, feature in enumerate(numerical_columns, 1):
    plt.subplot(len(numerical_columns), 2, idx)
    sns.histplot(df[feature], kde=True)
    plt.title(f"{feature} | Skewness: {round(df[feature].skew(), 2)}")

# Adjust layout and show plots
plt.tight_layout()
plt.show()
```


Id | Skewness: 0.0
Count
Id
SepalLengthCm | Skewness: 0.31
Count
SepalLengthCm
SepalWidthCm | Skewness: 0.33
Count
SepalWidthCm
PetalLengthCm | Skewness: -0.27
Count
PetalLengthCm
PetalWidthCm | Skewness: -0.1
Count
PetalWidthCm

```python
# Assuming 'df' is your DataFrame
plt.figure(figsize=(10, 8))

# Using Seaborn to create a swarm plot
sns.swarmplot(x="Species", y="PetalWidthCm", data=df, palette='viridis')

plt.title('Swarm Plot for QuaSpecieslity and PetalWidthCm')
plt.xlabel(' PetalWidthCm')
plt.ylabel('Species')
plt.show()
```


Swarm Plot for QuaSpecieslity and PetalWidthCm
2.5
2.0
1.5
1.0
0.5
0.0
Species
Iris-setosa
Iris-versicolor
Iris-virginica
PetalWidthCm


Thank You!!