Task 1: Exploring and Visualizing the Iris Dataset

1. Introduction and Problem Statement

The objective of this task is to perform an initial **Exploratory Data Analysis (EDA)** on the classic Iris Dataset. This is a foundational step in the data science pipeline that involves understanding the data's structure, cleaning it, and identifying patterns through visualization.

The primary goal is to analyze the physical characteristics of three iris species (*Setosa*, *Versicolor*, and *Virginica*) and determine how these features (sepal and petal dimensions) vary across the different classes.

2. Dataset Understanding and Description

The Iris dataset is a multivariate data set introduced by the British statistician and biologist Ronald Fisher in 1936. It consists of **150 samples** from three species of Iris.

Features Description:

- **Sepal Length (cm):** Measurement of the length of the flower's sepal.
- **Sepal Width (cm):** Measurement of the width of the flower's sepal.
- **Petal Length (cm):** Measurement of the length of the flower's petal.
- **Petal Width (cm):** Measurement of the width of the flower's petal.
- **Species (Target):** The specific class of the iris plant (Setosa, Versicolor, or Virginica).

3. Methodology

To complete this task, we will:

1. **Load** the data using `pandas`.
2. **Inspect** the data structure using `.shape`, `.columns`, and `.head()`.
3. **Clean** the data by checking for null values.
4. **Visualize** the relationships and distributions using `matplotlib` and `seaborn`.

```
# This Python 3 environment comes with many helpful analytics
libraries installed
# It is defined by the kaggle/python Docker image:
https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/"
directory
# For example, running this (by clicking run or pressing Shift+Enter)
will list all files under the input directory

import os
```

```
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that gets preserved as output when you create a version using "Save & Run All"
# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the current session
```

```
/kaggle/input/iris-dataset/iris.csv
```

2. Loading Dataset

In this section, we download and load the Iris dataset into the working environment using appropriate Python libraries. This step ensures that the data is available for inspection, cleaning, and further analysis.

```
import kagglehub

# Download latest version
path = kagglehub.dataset_download("himanshunakrani/iris-dataset")

print("Path to dataset files:", path)
```

```
Path to dataset files: /kaggle/input/iris-dataset
```

3. Dataset Understanding and Structure

In this section, we use Pandas to inspect the internal structure of the dataframe.
This includes checking the dimensions (shape), column names, and previewing the actual data.

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import kagglehub

# 1. Download and Load Data
path = kagglehub.dataset_download("himanshunakrani/iris-dataset")
# The file in this dataset is named 'iris.csv'
df = pd.read_csv(f"{path}/iris.csv")

# Set the visual style
sns.set_theme(style="whitegrid")

print("Data successfully loaded into 'df'.")
print("\n--- First 5 Rows ---")
print(df.head())
```

```
Data successfully loaded into 'df'.

--- First 5 Rows ---
   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
```

4. Data Cleaning and Preparation

Before visualizing, it is essential to ensure the data is complete and correctly formatted. We check for missing values and look at the statistical summary to identify any obvious anomalies or distributions.

```
# Check for missing values in the dataframe
print("Missing values per column:")
print(df.isnull().sum())

# Check data types of each column
print("\nData Types:")
print(df.dtypes)

# Statistical summary of numerical features
print("\nStatistical Summary:")
display(df.describe())
```

```
Missing values per column:
sepal_length    0
sepal_width     0
petal_length    0
petal_width     0
species         0
dtype: int64

Data Types:
sepal_length    float64
sepal_width     float64
petal_length    float64
petal_width     float64
species          object
dtype: object

Statistical Summary:

       sepal_length  sepal_width  petal_length  petal_width
count    150.000000   150.000000    150.000000   150.000000
mean       5.843333     3.054000      3.758667     1.198667
```

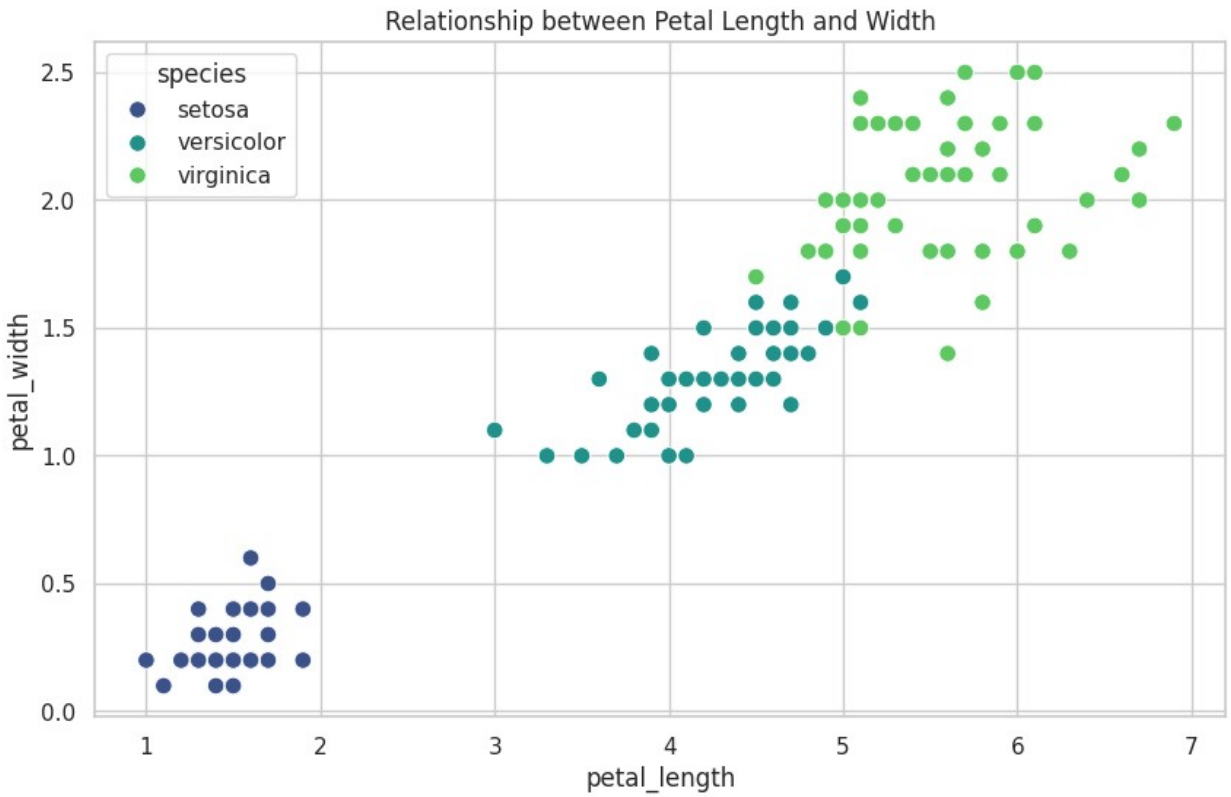
```
std         0.828066     0.433594     1.764420     0.763161
min         4.300000     2.000000     1.000000     0.100000
25%         5.100000     2.800000     1.600000     0.300000
50%         5.800000     3.000000     4.350000     1.300000
75%         6.400000     3.300000     5.100000     1.800000
max         7.900000     4.400000     6.900000     2.500000
```

5. Exploratory Data Analysis (EDA)

a. Visualization 1: Scatter Plot

Objective: Analyze the relationship between two continuous variables.
We plot Sepal Length vs. Sepal Width.
The species are mapped to different colors (hue) to observe whether they form distinct natural clusters based on these dimensions.

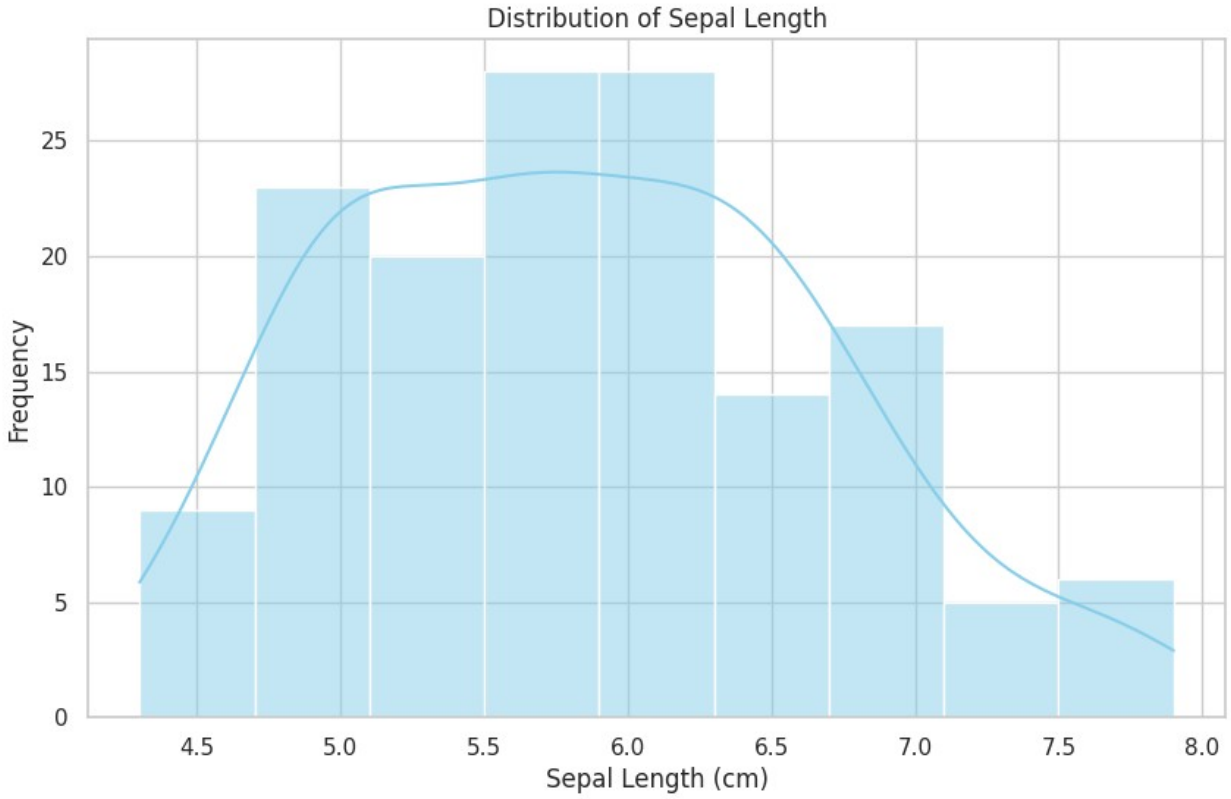
```
# 1. Scatter Plot: Petal Length vs Petal Width
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df, x='petal_length', y='petal_width',
hue='species', palette='viridis', s=70)
plt.title('Relationship between Petal Length and Width')
plt.show()
```



b. Visualization 2: Histogram

Objective: Examine the data distribution of a single variable.
We generate a histogram for Petal Length.
This helps visualize the frequency distribution and identify whether the data is multimodal (having more than one peak).

```
# 2. Histogram: Distribution of Sepal Length
plt.figure(figsize=(10, 6))
sns.histplot(df['sepal_length'], kde=True, color='skyblue')
plt.title('Distribution of Sepal Length')
plt.xlabel('Sepal Length (cm)')
plt.ylabel('Frequency')
plt.show()
```



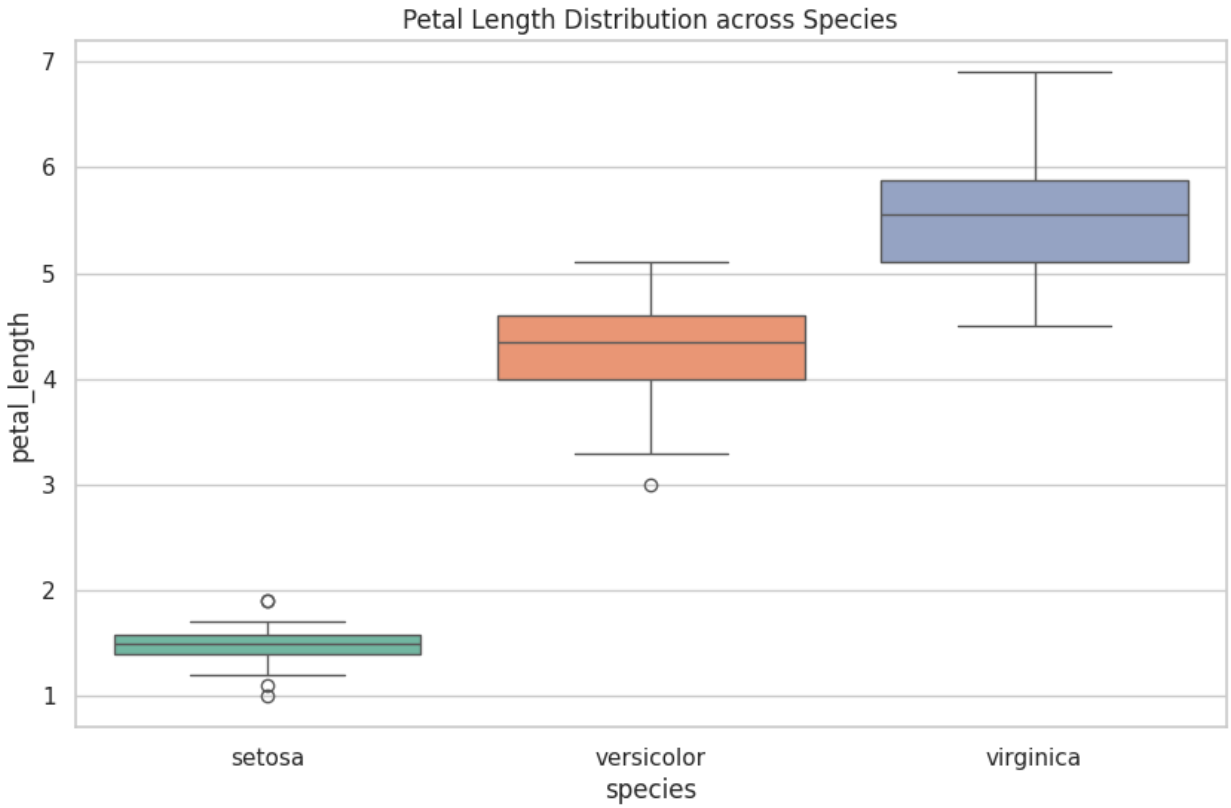
c. Visualization 3: Box Plot

Objective: Detect outliers and visualize the spread (variance) of values.
We use a box plot to compare Sepal Width across different species.
This visualization highlights the median, the interquartile range (IQR), and any potential outliers.

```
plt.figure(figsize=(10, 6))
sns.boxplot(
    x='species',
```

```
    y='petal_length',
    hue='species',
    data=df,
    palette='Set2',
    legend=False
)

plt.title('Petal Length Distribution across Species')
plt.show()
```



6. Conclusion and Key Insights

Based on the visualizations and analysis above, we can draw the following conclusions:

1. **Structure:** The Iris dataset contains 150 rows and 5 columns with no missing values, making it clean and ready for analysis.
2. **Species Separation:** The scatter plot shows that **Iris setosa** is easily distinguishable from **Versicolor** and **Virginica**, which exhibit some overlap. Petal dimensions are particularly effective for differentiating species.
3. **Feature Importance:** Petal length and petal width are stronger indicators for species classification compared to sepal measurements.

4. **Distribution:** Histograms reveal that most features follow a relatively normal distribution. Petal length, in particular, shows a bimodal distribution, indicating clear differences between flower groups.

5. **Outliers:** Box plots highlight very few outliers. Versicolor has a consistent sepal width, while Virginica shows a wider spread. Setosa has some minor outliers, but overall, the dataset is clean and consistent.

Conclusion:
This analysis confirms that the petal dimensions are strong differentiators for species classification. Visualizations such as scatter plots, histograms, and box plots provide valuable insights into data distribution, clustering, and variability, helping guide further analysis or model building.