Name	**Zunaira Hameed**
Tasks	**Basic Level**
Domain	**Data Analysis**
Company	**Codveda**

Overview:

This document summarizes the tasks completed during the **Basic Data Analysis** internship at Codveda. It covers **data cleaning**, **exploratory data analysis (EDA)**, and **basic data visualizations** performed on the **Iris dataset**. Below is a breakdown of the steps, code, and explanations for each task. Code is highlighted with yellow color.

Task 1: Data Cleaning and Preprocessing

Step 1: Import Dependencies

```
import pandas as pd
import warnings
warnings.filterwarnings('ignore')
```

Explanation:

- pandas is imported for data manipulation, such as loading and cleaning datasets.
- warnings.filterwarnings('ignore') disables unnecessary warnings that may appear during the execution of the code.

Step 2: Connect Google Colab with Google Drive

```
from google.colab import drive
drive.mount('/content/drive')
```

Explanation: This step mounts your Google Drive in Google Colab, allowing you to access files stored there.

Step 3: Load and Read Dataset

```
data = pd.read_csv("/content/drive/MyDrive/iris.csv")
data.head()
```

Explanation:

- The Iris dataset is loaded into a DataFrame using pd.read_csv().
- head() is used to display the first 5 rows of the dataset to confirm successful loading.

Step 4: Check for Missing Values

```
data.isnull().sum()
```

Explanation: This step checks for missing values in each column of the dataset. It returns the sum of null values in each column.

Step 5: Check for Duplicates

data.duplicated().sum()

Explanation: This checks for duplicate rows in the dataset. It helps ensure the data is clean and contains no repetitive entries.

Step 6: Separate Dependent and Independent Columns

x = data.drop(columns=['species'])
y = data['species']

Explanation:

- x contains the independent variables (features), which are all columns except 'species'.
- y contains the dependent variable (target), which is the 'species' column.

Step 7: Encode Categorical Data

label_map = {'setosa': 0, 'versicolor': 1, 'virginica': 2}
data['species'].replace(label_map, inplace=True)
data.head()

Explanation:

- The species column, which contains categorical data, is encoded into numeric values using a label_map.
- After encoding, head() is used to verify the changes.

Task 2: Exploratory Data Analysis (EDA)

Step 1: Dataset Summary

```
print(data.describe())
```

Explanation: This computes summary statistics for the numerical columns of the dataset, including mean, standard deviation, and percentiles.

Step 2: Mode of Each Column

```
print("Mode of each column: ")
for column in data.columns:
  mode_value = data[column].mode()
  print(f"{column}: {mode_value}")
```

Explanation: The mode (most frequent value) of each column is computed using mode(). This gives insights into the most common values in each feature.

Step 3: Median of Each Column

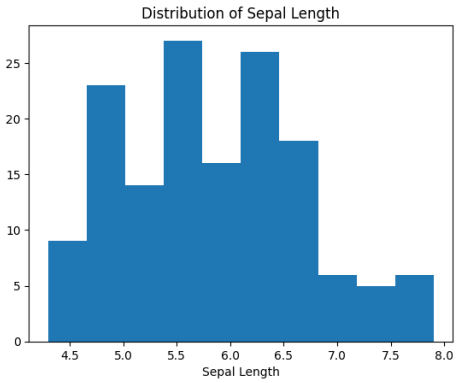
```
print("Median of each column: ")
print(data.median())
```

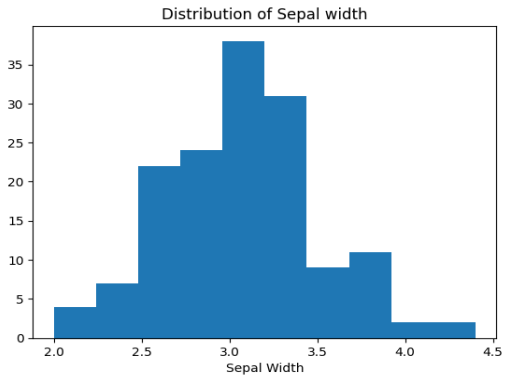
Explanation: This calculates the median value for each column in the dataset, providing a measure of central tendency.

Step 4: Data Visualization (Histograms, Box Plots, Scatter Plots, Correlations)

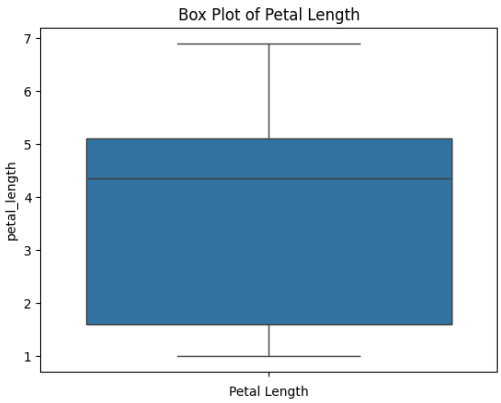
```
import matplotlib.pyplot as plt
import seaborn as sns

# Example: Histogram for Sepal Length
plt.hist(data['sepal_length'])
plt.title('Distribution of Sepal Length')
plt.xlabel('Sepal Length')
```

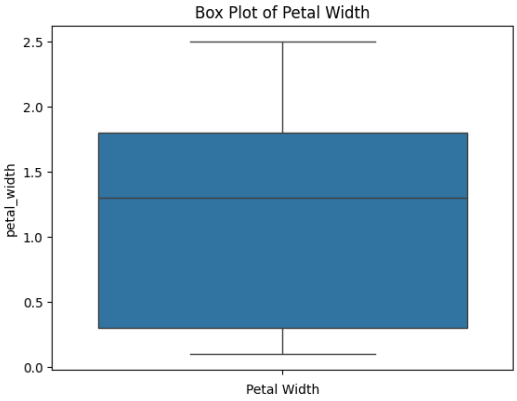
Histogram for Sepal length and Sepal width



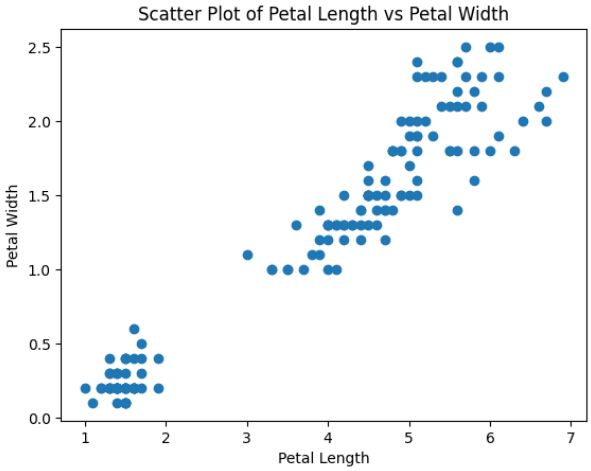


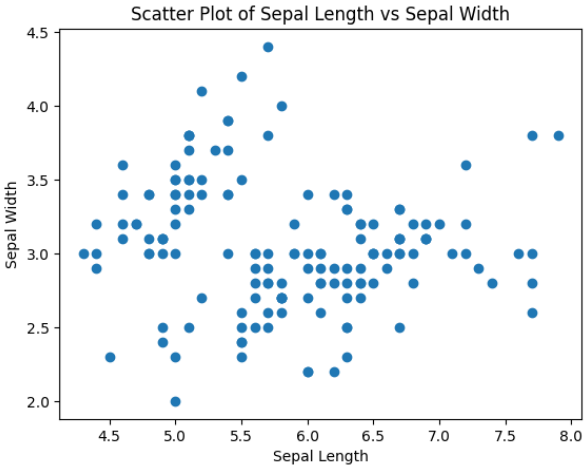
Box plot for Petal length and Petal width



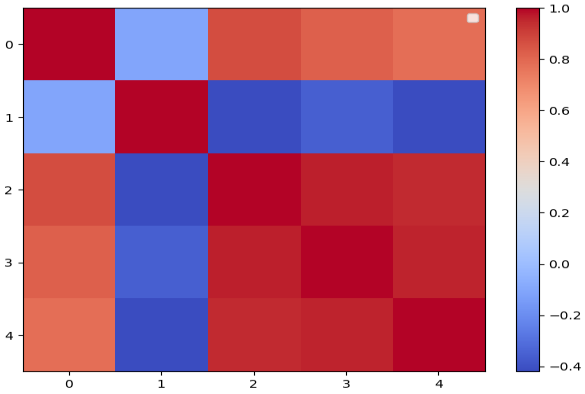


Scatter plot for: (Petal length vs Petal width) and (Sepal length vs Sepal width)





Correlation between Numerical columns



Explanation:

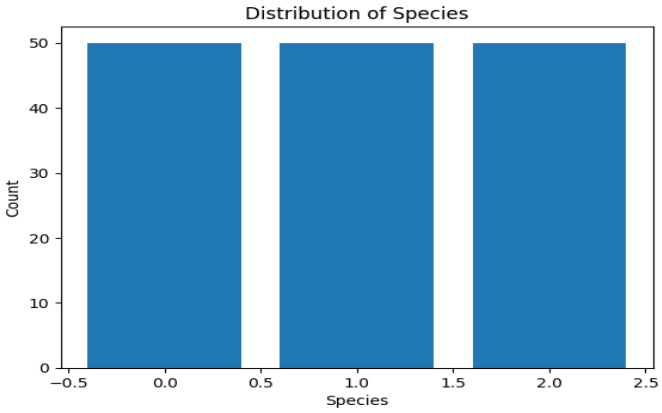
- Various plots are created to visualize the distribution and relationships between different features. These include:
  - **Histograms**: To see the distribution of individual variables.
  - **Box Plots**: To check for outliers and understand the spread.
  - **Scatter Plots**: To explore relationships between variables.
  - **Heatmap**: A correlation matrix to visualize the relationships between numerical columns.

Task 3: Basic Data Visualization

Step 1: Bar Plot for Species Distribution

```
species_count = data['species'].value_counts()
plt.bar(species_count.index, species_count.values)
plt.title('Distribution of Species')
plt.xlabel('Species')
plt.ylabel('Count')
```

Bar Plot for Species distribution

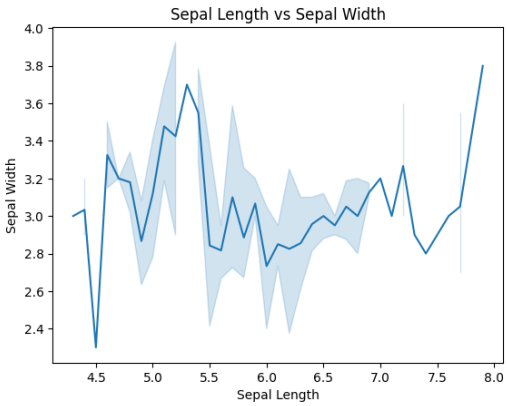


Explanation: A **bar plot** is used to visualize the distribution of different species in the dataset.

Step 2: Line Chart for Sepal Length vs Sepal Width

```
sns.lineplot(data=data, x='sepal_length', y='sepal_width')
plt.title('Sepal Length vs Sepal Width')
plt.xlabel('Sepal Length')
plt.ylabel('Sepal Width')
```

Line Chart for Sepal Length vs Sepal Width



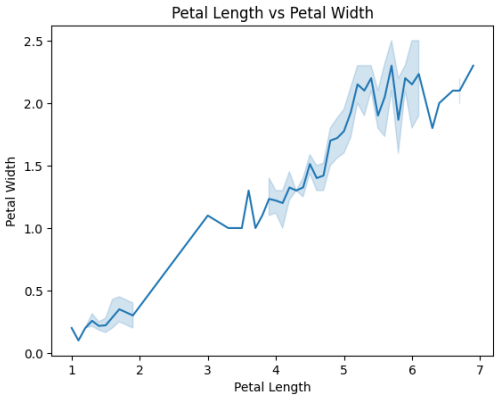
Explanation: A **line chart** is used to visualize the relationship between sepal length and sepal width.

Step 3: Line Chart for Petal Length vs Petal Width

```
sns.lineplot(data=data, x='petal_length', y='petal_width')
plt.title('Petal Length vs Petal Width')
plt.xlabel('Petal Length')
```

plt.ylabel('Petal Width')

Line Chart for Petal Length vs Petal Width



Explanation: Another **line chart** is used to explore the relationship between petal length and petal width.

Conclusion

This internship allowed me to gain valuable hands-on experience with key **data analysis techniques**:

- **Data cleaning** and **preprocessing** to prepare data for analysis.
- Performing **exploratory data analysis** (EDA) to understand the dataset.
- Creating basic **data visualizations** to summarize and present findings effectively.

By utilizing **Python**, **Pandas**, **Matplotlib**, and **Seaborn**, I have strengthened my skills in **data manipulation**, **analysis**, and **visualization**, and I look forward to applying them in future projects!