Exploring and Visualizing a Simple Dataset

Task-01

June 11, 2025
Muhammad Furqan Qureshi
DHC: 3634

Contents

Introduction

Objective:

In this task, I worked with the **Iris Dataset** to practice loading, inspecting, and visualizing a dataset. I learned how to explore the dataset's features, identify patterns, and create visualizations that provide insights into the data. The Iris Dataset is a classic dataset in machine learning, containing information about three species of iris flowers, with features such as sepal length, sepal width, petal length, and petal width.

Dataset:

The Iris dataset consists of 150 rows and 5 columns. The columns are:

- **sepal_length**: The length of the sepal.
- **sepal_width**: The width of the sepal.
- **petal_length**: The length of the petal.
- **petal_width**: The width of the petal.
- **species**: The species of the flower (setosa, versicolor, virginica).

Data Loading and Inspection

Step 1: Loading the Dataset

I loaded the Iris dataset using the `seaborn` library, which provides easy access to built-in datasets. I used the following code to load the dataset

```
import seaborn as sns
import pandas as pd

# Load the Iris dataset
iris = sns.load_dataset('iris')
```

Step 2: Inspecting the Dataset

Once the data was loaded, I printed the first few rows of the dataset to understand its structure and get a quick overview of the features

```
# Display the first few rows of the dataset
print(iris.head())
```

Step 3: Checking Dataset Shape and Columns

I checked the shape of the dataset to understand the number of rows and columns. I also printed the column names

```
# Check the shape of the dataset (rows, columns)
print(f"Shape of dataset: {iris.shape}")

# Print the column names
print(f"Columns: {iris.columns}")
```

Step 4: Summary Statistics

I used the `describe()` function to generate summary statistics for the numeric columns. This provided a quick overview of the distributions of features

```
# Display summary statistics for numeric columns
print(iris.describe())
```

Exploratory Data Analysis (EDA)

Step 1: Checking for Missing Values

I used the `isnull()` function to check if there were any missing values in the dataset. This is an essential step to ensure the integrity of the dataset before visualizing or training a model

```
# Check for missing values
print(iris.isnull().sum())
```

In this case, there were no missing values in the dataset, so I proceeded to the next step.

Step 2: Checking Data Types

I also checked the data types of each column to ensure that they were appropriate for analysis. This can help identify any issues (such as numeric columns stored as strings)

```
# Check data types of each column
print(iris.info())
```

Data Visualization

Step 1: Scatter Plot to Show Relationships Between Features

I wanted to visualize the relationships between the numeric features of the Iris dataset. A scatter plot matrix (pair plot) is a great way to examine how each feature is related to the others

```
# Create a pairplot to visualize relationships between
features
sns.pairplot(iris, hue='species')
plt.show()
```

In the pairplot, each pair of features is plotted against each other, and the data points are colored by species. This helps to identify patterns and correlations between features.

Step 2: Histograms to Show Feature Distributions

Next, I visualized the distribution of each feature using histograms. This helps in understanding the spread and distribution of each feature

```
# Create histograms to visualize distributions of features
iris.hist(bins=20, figsize=(10, 8))
plt.show()
```

This gives me insight into how the values for each feature are distributed across the dataset. For example, I could see that `petal_length` had a bimodal distribution.

Step 3: Box Plots to Identify Outliers

Box plots are useful for identifying outliers in the data. I created box plots for each feature to check for any unusual values or potential outliers.

```
# Create box plots to visualize potential outliers
plt.figure(figsize=(10, 8))
sns.boxplot(data=iris)
plt.show()
```

The box plots display the spread of each feature and highlight any points that fall outside the typical range of values.

Conclusion

In this task, I explored the Iris dataset and visualized various aspects of the data to gain insights. I was able to.

- Load and inspect the dataset to understand its structure.
- Use summary statistics and data types to understand the distribution of the data.
- Visualize the relationships between features using scatter plots, histograms, and box plots.

From the visualizations, I learned that:

- There are clear relationships between the features, such as `petal_length` and `petal_width`, which can help in classification tasks.
- The dataset is well-separated by species, and certain features like `petal_length` and `petal_width` can be good predictors for species classification.
- There were no significant outliers or missing data, which makes the dataset ready for model building.

These insights would be crucial for building a machine learning model to classify the species of iris flowers based on the features.

References

Seaborn Documentation: https://seaborn.pydata.org/

Iris Dataset: Available from the `seaborn` library, or Kaggle.