Exploratory Data Analysis and Preprocessing on the Iris Dataset

1. Introduction:

The Iris dataset is a classic and beginner-friendly dataset used in the world of data science and machine learning. It contains simple, structured data about different types of iris flowers, including measurements like petal length, sepal width, and more. Each flower in the dataset belongs to one of three species: Setosa, Versicolor, or Virginica.

In this project, the goal is to explore this dataset using Python and popular libraries like Pandas, Seaborn, and Scikit-learn. We'll dive into the data to understand how the features relate to each other, clean it by removing duplicates and outliers, and prepare it for machine learning by scaling and splitting it. Visualizations like pairplots and heatmaps help bring insights to life, making it easier to spot trends and correlations.

Overall, this project walks through the essential steps of turning raw data into something clean, organized, and ready to be used for training a machine learning model.

2. Objective:

The objective of this project is to perform a complete exploratory data analysis and preprocessing on the Iris dataset using Python. The aim is to understand the dataset's structure, visualize feature relationships, clean any inconsistencies, and prepare the data for machine learning tasks.

The specific goals are:

- To load and inspect the Iris dataset using Python libraries
- To check for and remove duplicate records
- To detect and remove outliers using statistical methods
- To visualize the distribution of features and correlations using plots
- To scale the data using normalization techniques
- To split the data into training and testing sets for future model building

By achieving these steps, the dataset becomes ready for training reliable machine learning models.

3. Methodology:

This project was implemented using Python, along with several popular libraries such as Pandas, NumPy, Seaborn, Matplotlib, and Scikit-learn. The methodology followed a step-by-step process to ensure the dataset was thoroughly explored, cleaned, and prepared for machine learning.

The following steps were carried out:

- Dataset Loading:
  The Iris dataset was imported using load_iris() from Scikit-learn's built-in datasets.
- Initial Inspection:
  Displayed dataset shape, column names, and basic statistical summary using .info() and .describe() functions.
- Missing Values Check:
  Verified that there were no null values using .isnull().sum().
- Duplicate Removal:
  Detected and removed any duplicate rows using .duplicated() and .drop_duplicates().
- Outlier Detection and Removal:
  Applied the Interquartile Range (IQR) method to identify and eliminate outliers.
- Data Visualization:
  Used Seaborn to create a pairplot for understanding relationships between features.
  Created a correlation heatmap to observe feature interdependencies.
- Feature Scaling:
  Standardized the feature values using StandardScaler to bring them to a common scale.
- Train-Test Split:
  Split the data into training and testing sets using train_test_split() from Scikit-learn.

Each step was carefully implemented to follow best practices in data preprocessing and to ensure the dataset was in a clean and usable state for future modeling.

4. Code and Implementation Details

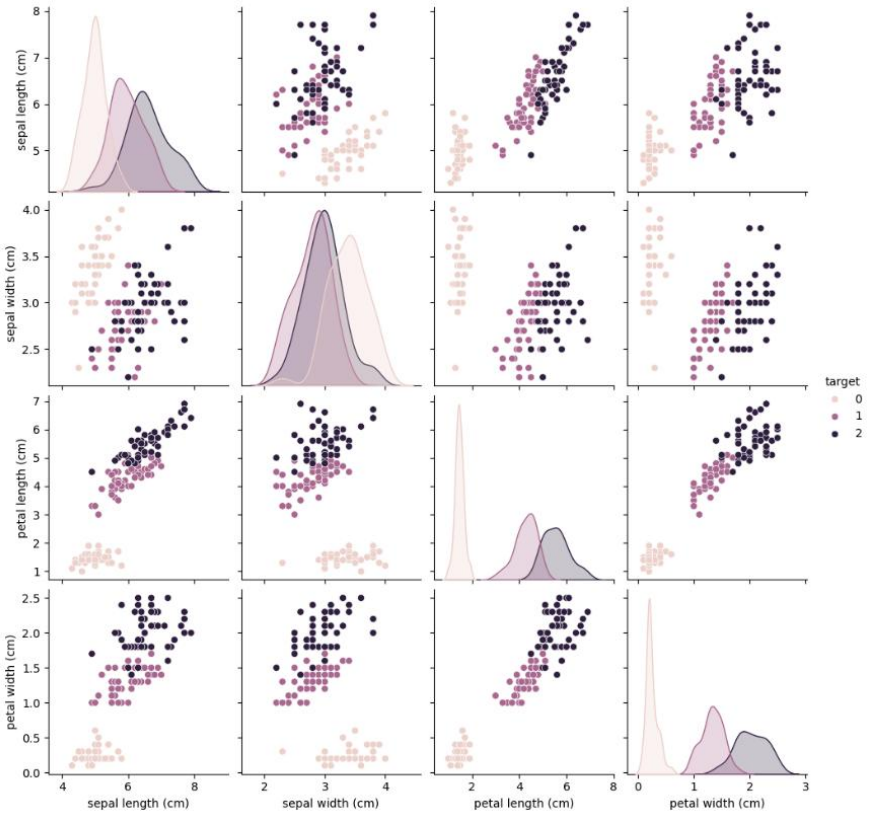
The Iris dataset was analyzed and preprocessed using a structured Python script. The script performs the following tasks:

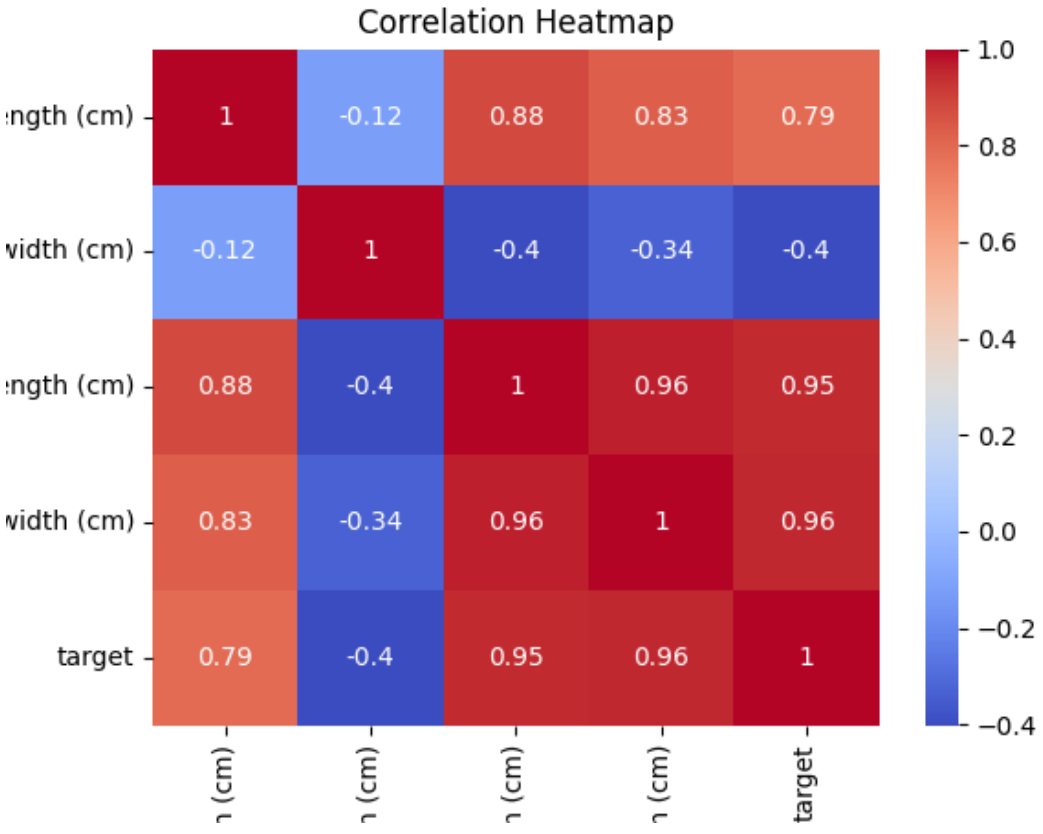
- Loads the dataset
- Displays basic info and statistics
- Removes duplicates
- Detects and removes outliers using the IQR method
- Visualizes feature distributions and correlations
- Scales the features
- Splits the data for training and testing

Code:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

iris = load_iris()
df = pd.DataFrame(data=iris.data, columns=iris.feature_names)
df['target'] = iris.target

print("Shape:", df.shape)
print(df.info())
print(df.describe())

print("Missing values:\n", df.isnull().sum())

print("Duplicate rows:", df.duplicated().sum())
df = df.drop_duplicates()

Q1 = df.quantile(0.25)
Q3 = df.quantile(0.75)
IQR = Q3 - Q1
df = df[~((df < (Q1 - 1.5 * IQR)) | (df > (Q3 + 1.5 * IQR))).any(axis=1)]

sns.pairplot(df, hue='target')
plt.savefig('screenshots/pairplot.png')
plt.show()

sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.title("Feature Correlation")
plt.savefig('screenshots/heatmap.png')
plt.show()

X = df.drop('target', axis=1)
y = df['target']

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2,
random_state=42)
print("Train shape:", X_train.shape)
print("Test shape:", X_test.shape)
```

Screenshots:





5. Results and Observations

- No missing values were found in the dataset.
- A few duplicate records were identified and removed to ensure data integrity.
- Outlier detection using the Interquartile Range (IQR) method resulted in the removal of some extreme values, slightly reducing the dataset size.
- The pairplot clearly showed distinct clustering between the three iris species, especially in petal-related features.
    - Setosa (class 0) formed an isolated cluster in almost every plot.
    - Petal length and petal width were highly effective in separating species.
- The correlation heatmap revealed:
    - A strong positive correlation between petal length and petal width
    - Weaker or negative correlations with sepal measurements.
- Feature scaling using StandardScaler normalized all input values to a standard range, preparing the data for model training.
- The final cleaned dataset was successfully split into training and testing sets using an 80-20 ratio.

6. Conclusion

In this project, we performed a complete exploratory data analysis and preprocessing workflow on the Iris dataset. The dataset was inspected for missing values and duplicates, both of which were handled appropriately. Outliers were removed using the IQR method to ensure cleaner feature distributions.

Using visual tools like pairplots and correlation heatmaps, we gained valuable insights into how the features relate to one another and how well they separate different species of iris flowers. We found that petal-related features were the most effective in distinguishing between classes.

Finally, we applied feature scaling to normalize the data and split it into training and testing sets, making the dataset fully ready for machine learning model development. This structured approach reflects real-world practices in AI and data science, where clean and well-understood data is the foundation of accurate predictive modeling.