Exploratory Data Analysis and Preprocessing on the Iris Dataset

1. Introduction:

The Iris dataset is a well-known and beginner-friendly dataset often used in data science and machine learning. It includes measurements of iris flowers, such as petal length, petal width, sepal length, and sepal width. Each entry in the dataset is labelled as one of three species: Setose, Versicolor, or Virginica.

In this project, we'll use Python along with libraries like Pandas, Seaborn, and Scikit-learn to explore and analyse the dataset. We'll start by examining how the features are related, remove any duplicate entries and outliers to clean the data, and then prepare it for machine learning by scaling and splitting it into training and testing sets.

Visualizations such as pair plots and heatmaps will help us better understand the relationships between features and highlight any patterns or correlations. This project serves as a hands-on example of the key steps involved in transforming raw data into a well-prepared dataset for building machine learning models.

2. Objective:

This project focuses on performing a comprehensive exploratory data analysis (EDA) and data preprocessing on the Iris dataset using Python. The main purpose is to gain insights into the dataset, identify and fix any irregularities, and prepare the data for machine learning tasks.

The key steps involved in this process are:

- Importing and exploring the Iris dataset using popular Python tools
- Detecting and eliminating duplicate records to ensure data quality
- Identifying outliers through statistical analysis and removing them
- Creating visualizations to understand feature distributions and relationships
- Normalizing the data to bring all features to a similar scale
- Splitting the dataset into training and testing subsets for future model training

By completing these tasks, the dataset will be fully cleaned and organized, making it suitable for building reliable and accurate machine learning models.

3. Methodology:

This project was developed using Python and several widely-used libraries, including Pandas, NumPy, Seaborn, Matplotlib, and Scikit-learn. A systematic approach was taken to explore, clean, and prepare the Iris dataset for machine learning applications.

The process followed these main steps:

- **Loading the Dataset**
  The Iris dataset was imported using the load_iris() function available in Scikit-learn's datasets module.
- **Initial Exploration**
  Basic information about the dataset, such as shape, column names, and descriptive statistics, was obtained using .info() and .describe().
- **Checking for Missing Data**
  Ensured there were no missing values in the dataset using .isnull().sum().
- **Removing Duplicates**
  Identified and eliminated duplicate records using .duplicated() and .drop_duplicates().
- **Outlier Handling**
  Detected and removed outliers using the Interquartile Range (IQR) method to maintain data integrity.
- **Data Visualization**
  Generated a pairplot using Seaborn to explore relationships among features and a heatmap to analyze feature correlations.
- **Feature Scaling**
  Standardized the dataset using StandardScaler to ensure all features were on a similar scale.
- **Splitting the Data**
  Divided the data into training and testing sets using Scikit-learn's train_test_split() for future model development.

Each stage was carefully carried out, following best practices in data preprocessing to ensure the dataset was well-prepared and suitable for training accurate machine learning models.

4. Code and Implementation Details

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

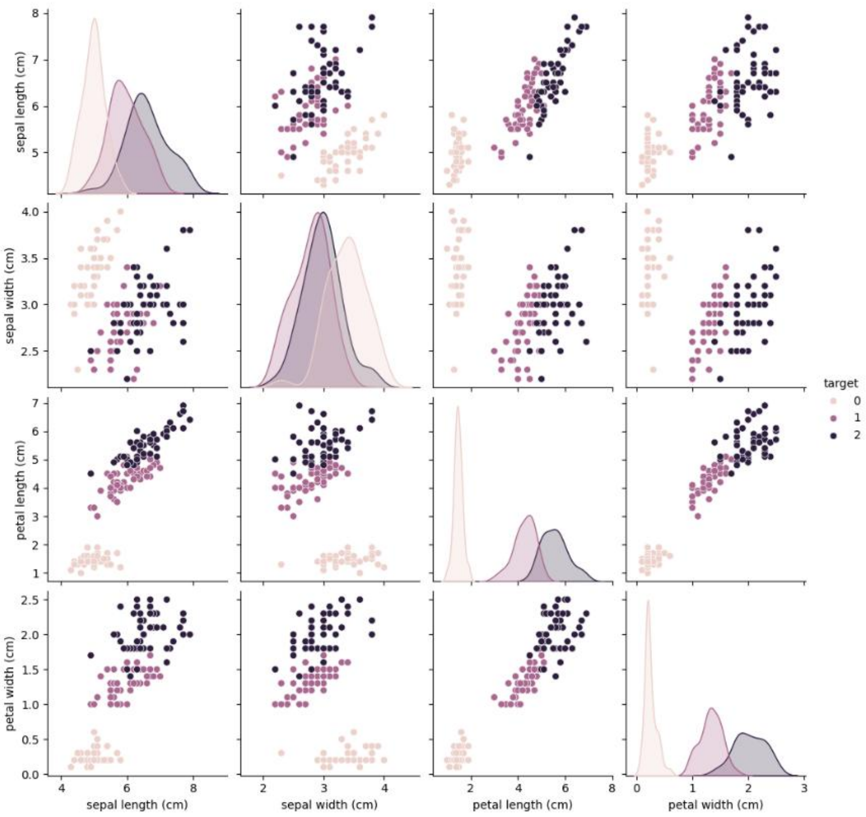
- Imports the Iris dataset
- Presents dataset structure and summary statistics
- Identifies and eliminates duplicate entries
- Uses the IQR technique to find and remove outliers
- Generates visualizations to explore feature patterns and relationships
- Applies feature scaling for normalization

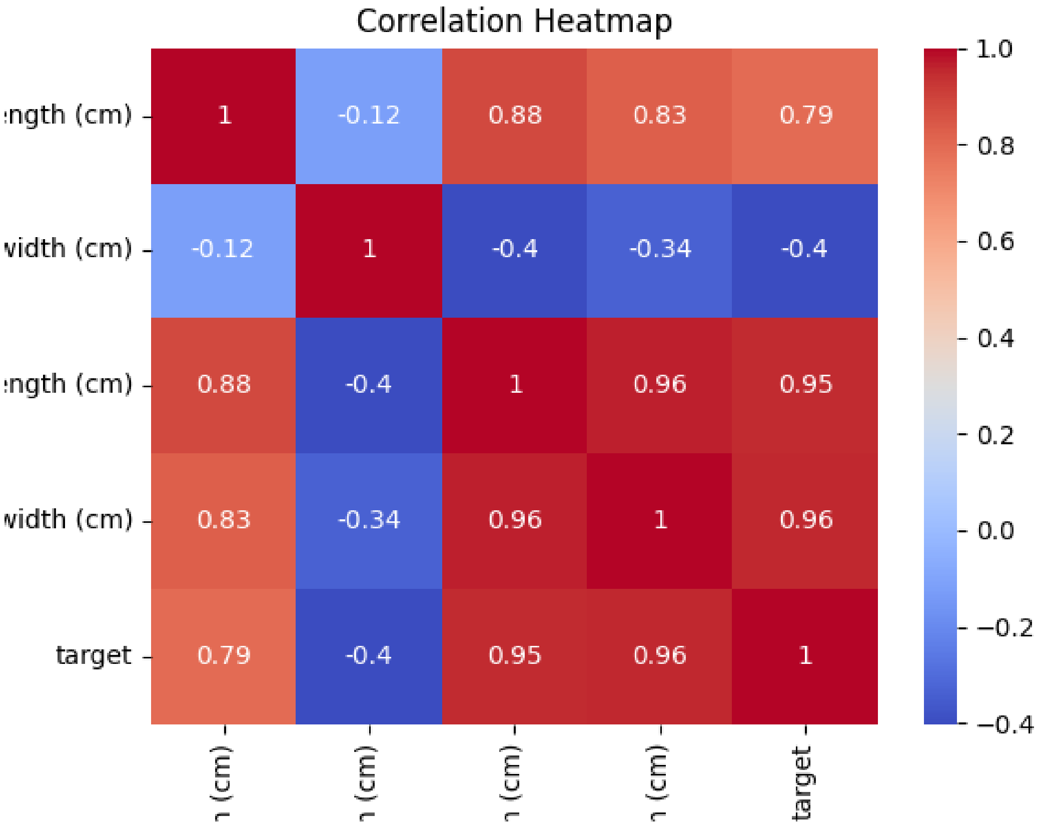
- Divides the dataset into training and testing sets

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

- The dataset was free from any missing values.
- Duplicate entries were detected and removed to maintain the quality of the data.
- Outliers were identified using the Interquartile Range (IQR) method and removed, resulting in a slight reduction in the number of records.
- The pairplot illustrated clear clusters among the three iris species, particularly in petal-based features:
    - Setosa (class 0) formed a distinct and separate group across most feature combinations.
    - Petal length and width stood out as the most effective features for distinguishing between species.
- The correlation heatmap indicated:
    - A strong positive relationship between petal length and petal width
    - Lower or negative correlations involving sepal length and width
- Feature normalization was carried out using StandardScaler, bringing all feature values onto a similar scale for modeling.
- The processed dataset was split into training and test sets in an 80:20 ratio to support model evaluation.

6. Conclusion

In this project, a full exploratory data analysis (EDA) and preprocessing pipeline was carried out on the Iris dataset. The dataset was examined for missing values and duplicate entries, both of which were properly addressed. Outliers were identified and removed using the Interquartile Range (IQR) method to improve the quality of feature distributions.

Through visual analysis using pairplots and correlation heatmaps, we explored the relationships between features and observed how effectively they differentiate the iris species. Notably, petal length and petal width emerged as the most influential features for class separation.

To prepare the data for machine learning, feature scaling was performed to standardize the input values, followed by splitting the dataset into training and testing subsets. This end-to-end process mirrors real-world data science workflows, where a clean and well-understood dataset forms the backbone of building reliable and accurate predictive models.