Data Science - Project 1 Documentation

Jiten Nilawar

January 20, 2024

Introduction

This document provides documentation for the data science task performed on the Iris dataset. The task includes Exploratory Data Analysis (EDA) and the implementation of a basic machine learning model for flower species prediction.

Exploratory Data Analysis (EDA)

Data Loading

The Iris dataset was loaded from the provided CSV file using the pandas library.

Summary Statistics and Visualization

Summary statistics were generated to understand the central tendency and dispersion of the data. Visualizations, such as histograms, box plots, and pair plots, were created to explore the distribution of each feature and relationships between features.

```
# Import necessary libraries
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score,
    classification_report, confusion_matrix

# Load the Iris dataset
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
column_names = ["sepal_length", "sepal_width", "petal_length", "petal_width", "
    class"]
iris_df = pd.read_csv(url, header=None, names=column_names)

# Split the dataset into features (X) and target variable (y)
X = iris_df.drop("class", axis=1)
y = iris_df["class"]

# Split the dataset into training and testing sets (80% training, 20% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
    random_state=42)

# Initialize the Decision Tree classifier
clf = DecisionTreeClassifier(random_state=42)

# Train the model on the training set
clf.fit(X_train, y_train)

# Make predictions on the testing set
y_pred = clf.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average="weighted")
```

```
recall = recall_score(y_test, y_pred, average="weighted")

# Display evaluation metrics
print("Accuracy:", accuracy)
print("Precision:", precision)
print("Recall:", recall)

# Display classification report and confusion matrix
print("\nClassification Report:")
print(classification_report(y_test, y_pred))

print("\nConfusion Matrix:")
print(confusion_matrix(y_test, y_pred))
```

Listing 1: EDA Code

Data Science Task

Algorithm Choice

For the predictive modeling task, a Decision Tree classifier was chosen due to its simplicity and effectiveness for classification tasks, especially when dealing with small to medium-sized datasets.

Feature Selection

All four features (sepal length, sepal width, petal length, and petal width) were used for training the model, as they are essential for distinguishing between different iris species.

Training and Evaluation

The dataset was split into training and testing sets. The Decision Tree classifier was trained on the training set and evaluated on the testing set. Evaluation metrics included accuracy, precision, recall, classification report, and confusion matrix.

```
# Import necessary libraries
import seaborn as sns
import matplotlib.pyplot as plt

# Load the Iris dataset
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
column_names = ["sepal_length", "sepal_width", "petal_length", "petal_width", "
    class"]
iris_df = pd.read_csv(url, header=None, names=column_names)

# Set the style for seaborn plots
sns.set(style="whitegrid")

# Distribution of each feature using histograms
plt.figure(figsize=(12, 8))
for i, column in enumerate(iris_df.columns[:-1]):
    plt.subplot(2, 2, i+1)
    sns.histplot(iris_df[column], kde=True, bins=20, color='skyblue')
    plt.title(f"{column} Distribution")

plt.tight_layout()
plt.show()

# Box plots for each feature
plt.figure(figsize=(12, 8))
for i, column in enumerate(iris_df.columns[:-1]):
    plt.subplot(2, 2, i+1)
    sns.boxplot(x="class", y=column, data=iris_df)
```

```
    plt.title(f"{column} Distribution by Class")

plt.tight_layout()
plt.show()

# Pairplot to visualize relationships between features
sns.pairplot(iris_df, hue="class", markers=["o", "s", "D"])
plt.show()
```

Listing 2: Data Science Task Code

Challenges Faced

During the EDA, ensuring proper visualization and interpretation of pair plots for relationships between features can be challenging, especially in larger datasets. In the data science task, tuning hyperparameters of the model for better performance might be required in more complex datasets.

Conclusion

In conclusion, this data science project effectively explored the Iris dataset through Exploratory Data Analysis (EDA) and implemented a Decision Tree classifier for flower species prediction. The chosen features and algorithm demonstrated satisfactory performance, as indicated by accuracy, precision, and recall metrics. The project provides valuable insights into dataset characteristics and lays the groundwork for further applications in machine learning and data science.