Iris Dataset Classification with SVM

Author: Sai Adith Prakash

Introduction

The purpose of this project is to classify the Iris dataset using Support Vector Machine (SVM). The Iris dataset is a well-known dataset in the machine learning community, containing measurements of four features of iris flowers (sepal length, sepal width, petal length, and petal width) for three different species. The goal is to predict the species of iris flowers based on these features.

What is SVM?

Support Vector Machine (SVM) is a supervised machine learning algorithm that can be used for both classification and regression tasks. However, it is mostly used for classification problems. SVM works by finding the hyperplane that best divides a dataset into classes. The hyperplane is chosen to maximize the margin, which is the distance between the hyperplane and the nearest data points from each class.

Why this project is useful?

This project is useful for several reasons:

- It provides a hands-on example of how to apply SVM to a real-world dataset.
- It demonstrates the importance of exploratory data analysis (EDA) in understanding the dataset and its characteristics.
- It shows how to preprocess data, train a machine learning model, and evaluate its performance.
- It serves as an educational tool for those learning machine learning concepts and techniques.
- It highlights the importance of data visualization in the model development process.

Modules Used

The following Python modules are used in this project:

- **pandas:** For data manipulation and analysis.
- **seaborn:** For data visualization.
- **matplotlib:** For creating static, animated, and interactive visualizations.
- **sklearn:** For machine learning algorithms and tools.

Code Snippets and Explanations

1. Loading the Iris dataset

The Iris dataset is loaded from the sklearn library, and a DataFrame is created to store the data.

```
from sklearn import datasets
import pandas as pd

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

print("First few rows of the dataset:")
print(df.head())
```

2. Summary Statistics

Summary statistics of the dataset are generated to understand the distribution and spread of the features.

```
print("Summary statistics:")
print(df.describe())
```

3. Exploratory Data Analysis (EDA)

Various visualizations are created to explore the dataset.

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

sns.pairplot(df, hue='species', palette='viridis')
plt.title('Pair Plot of Iris Dataset')
plt.show()

df.drop('species', axis=1).hist(bins=20, figsize=(10, 7), grid=False,
color='blue')
plt.suptitle('Feature Histograms')
plt.show()

plt.figure(figsize=(12, 6))
for i, feature in enumerate(df.columns[:-1]):
    plt.subplot(2, 2, i+1)
    sns.boxplot(x='species', y=feature, data=df, hue='species',
```

```
palette='viridis', legend=False)
    plt.title(f'Box Plot of {feature}')
plt.tight_layout()
plt.show()

correlation_matrix = df.corr()
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f',
linewidths=0.5)
plt.title('Feature Correlation Heatmap')
plt.show()

plt.figure(figsize=(6, 4))
sns.countplot(x='species', data=df, palette='viridis')
plt.title('Class Distribution in Iris Dataset')
plt.show()
```

4. Data Preprocessing

The features are standardized using StandardScaler before training the SVM model.

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

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

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

5. Training the SVM Model

An SVM model with a linear kernel is trained on the preprocessed data.

```
from sklearn.svm import SVC

svm_model = SVC(kernel='linear', probability=True)
svm_model.fit(X_train_scaled, y_train)
```

6. Model Evaluation

The trained SVM model is evaluated using accuracy score, confusion matrix, and classification report.

```
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report

y_pred = svm_model.predict(X_test_scaled)

accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)

print(f"Model Accuracy: {accuracy:.2f}")
print("Confusion Matrix:")
print(conf_matrix)
print("Classification Report:")
print(class_report)
```

Conclusion and Future Work

In conclusion, this project demonstrates the application of Support Vector Machine (SVM) to classify the Iris dataset. The project highlights the importance of data preprocessing, exploratory data analysis, and model evaluation in the machine learning pipeline. The SVM model achieved high accuracy, demonstrating its effectiveness for this classification task.

Future work could include exploring other classification algorithms, performing hyperparameter tuning to optimize the SVM model, and applying the model to other datasets. Additionally, incorporating more advanced data visualization techniques and feature engineering could further improve the model's performance and provide deeper insights into the dataset.