Iris Flowers Classification Project

Ramesh Chandra Soren

Enrollment No.: 2022CSB086
Department: Computer Science and Technology

---

1. Problem Statement

The objective of this task is to classify Iris flowers into three distinct species: Iris-setosa, Iris-versicolor, and Iris-virginica. This classification will be based on the morphological measurements of the flowers, specifically the lengths and widths of their petals and sepals.

Steps to Follow:

a) **Data Acquisition:** Download the Iris dataset from the UCI Machine Learning Repository.

b) **Data Preparation:** Load the dataset into a suitable data structure (Pandas DataFrame).

c) **Feature Selection:** Use the provided features (sepal length, sepal width, petal length, petal width).

d) **Model Training:** Develop a classification model to categorize the Iris flowers.

e) **Model Evaluation:** Evaluate the model's performance using appropriate metrics.

f) **Visualization:** Plot the classification results to visually assess the model's performance.

The aim is to achieve accurate classification of Iris flower species and demonstrate the model's effectiveness through evaluation metrics and visualizations.

---

2. Implementation

2.1 Install and Import Libraries

```python
```

Uncomment and run if you need to install any libraries

!pip install pandas numpy scikit-learn matplotlib seaborn tensorflow

import pandas as pd import numpy as np from sklearn.datasets import load_iris from sklearn.model_selection import train_test_split from sklearn.svm import SVC from sklearn.metrics import confusion_matrix, classification_report import matplotlib.pyplot as plt import seaborn as sns

```
import pandas as pd
import numpy as np
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix, classification_report
import matplotlib.pyplot as plt
import seaborn as sns
```

2.2 Data Acquisition and Preparation

```
# Load the Iris dataset
iris = load_iris()
X = iris.data
y = iris.target

# Create a DataFrame
df = pd.DataFrame(X, columns=iris.feature_names)
df['class'] = pd.Categorical.from_codes(y, iris.target_names)

print(df.head())
print(df['class'].value_counts())
```

```
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width
(cm)  \
0                5.1               3.5                1.4
0.2
1                4.9               3.0                1.4
0.2
2                4.7               3.2                1.3
0.2
3                4.6               3.1                1.5
0.2
4                5.0               3.6                1.4
```

```
0.2

    class
0  setosa
1  setosa
2  setosa
3  setosa
4  setosa
class
setosa        50
versicolor    50
virginica     50
Name: count, dtype: int64
```

2.3 Feature Selection and Data Splitting

```
# Separate features and target
X = df.drop('class', axis=1)
y = df['class']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=42)

print("Training set shape:", X_train.shape)
print("Testing set shape:", X_test.shape)
```

```
Training set shape: (105, 4)
Testing set shape: (45, 4)
```

2.4 Model Training

```
# Train SVM model
svm = SVC(kernel='linear', random_state=42)
svm.fit(X_train, y_train)

print("Model training completed.")
```

```
Model training completed.
```

2.5 Model Evaluation

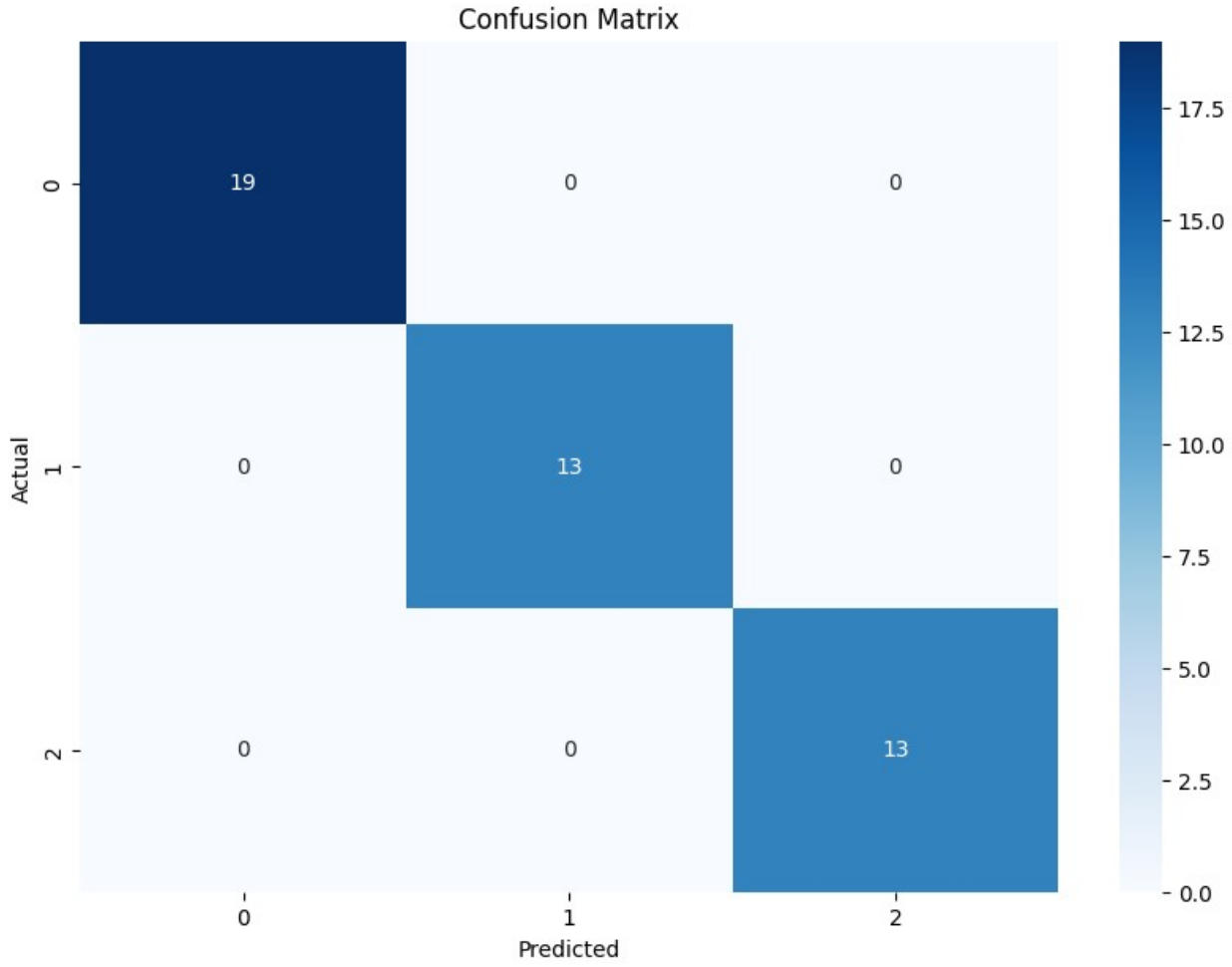
```
# Make predictions
y_pred = svm.predict(X_test)

# Create confusion matrix
cm = confusion_matrix(y_test, y_pred)

# Plot confusion matrix
plt.figure(figsize=(10,7))
```

```
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()

# Print classification report
print(classification_report(y_test, y_pred,
target_names=iris.target_names))
```




```
              precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        19
  versicolor       1.00      1.00      1.00        13
   virginica       1.00      1.00      1.00        13

    accuracy                           1.00        45
   macro avg       1.00      1.00      1.00        45
```

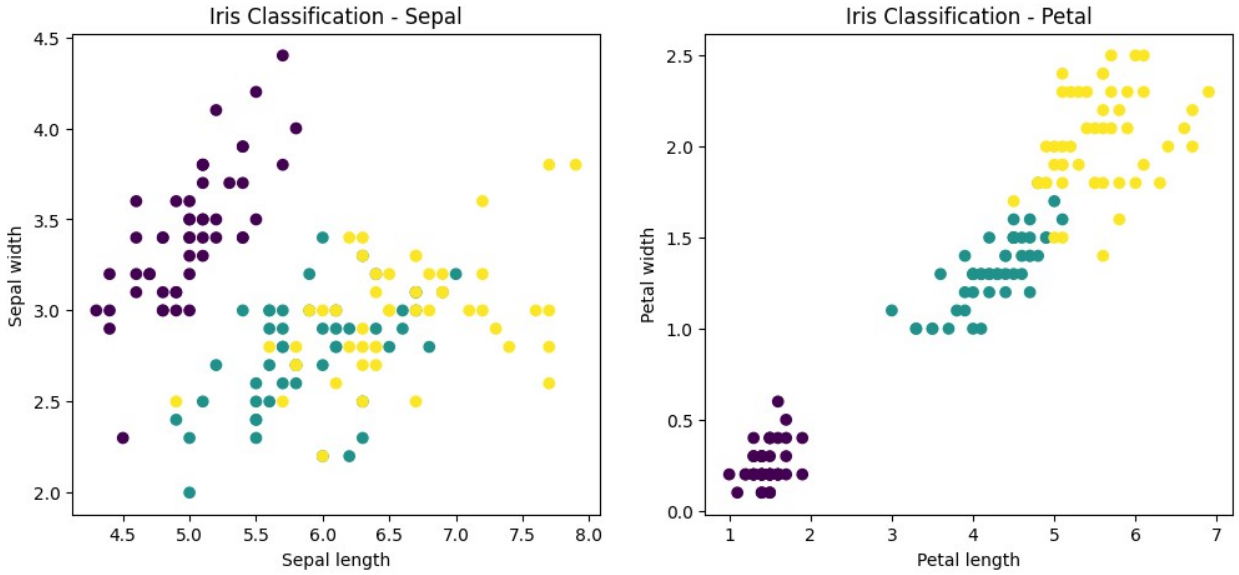
```
weighted avg       1.00      1.00      1.00        45
```

2.6 Visualization

```
plt.figure(figsize=(12,5))
plt.subplot(121)
# Access the columns by name  or use iloc
# Convert y to numerical values using .cat.codes
plt.scatter(X.iloc[:, 0], X.iloc[:, 1], c=y.cat.codes, cmap='viridis')
plt.xlabel('Sepal length')
plt.ylabel('Sepal width')
plt.title('Iris Classification - Sepal')

plt.subplot(122)
# Access the columns by name or use iloc
# Convert y to numerical values using .cat.codes
plt.scatter(X.iloc[:, 2], X.iloc[:, 3], c=y.cat.codes, cmap='viridis')
plt.xlabel('Petal length')
plt.ylabel('Petal width')
plt.title('Iris Classification - Petal')

plt.show()
```

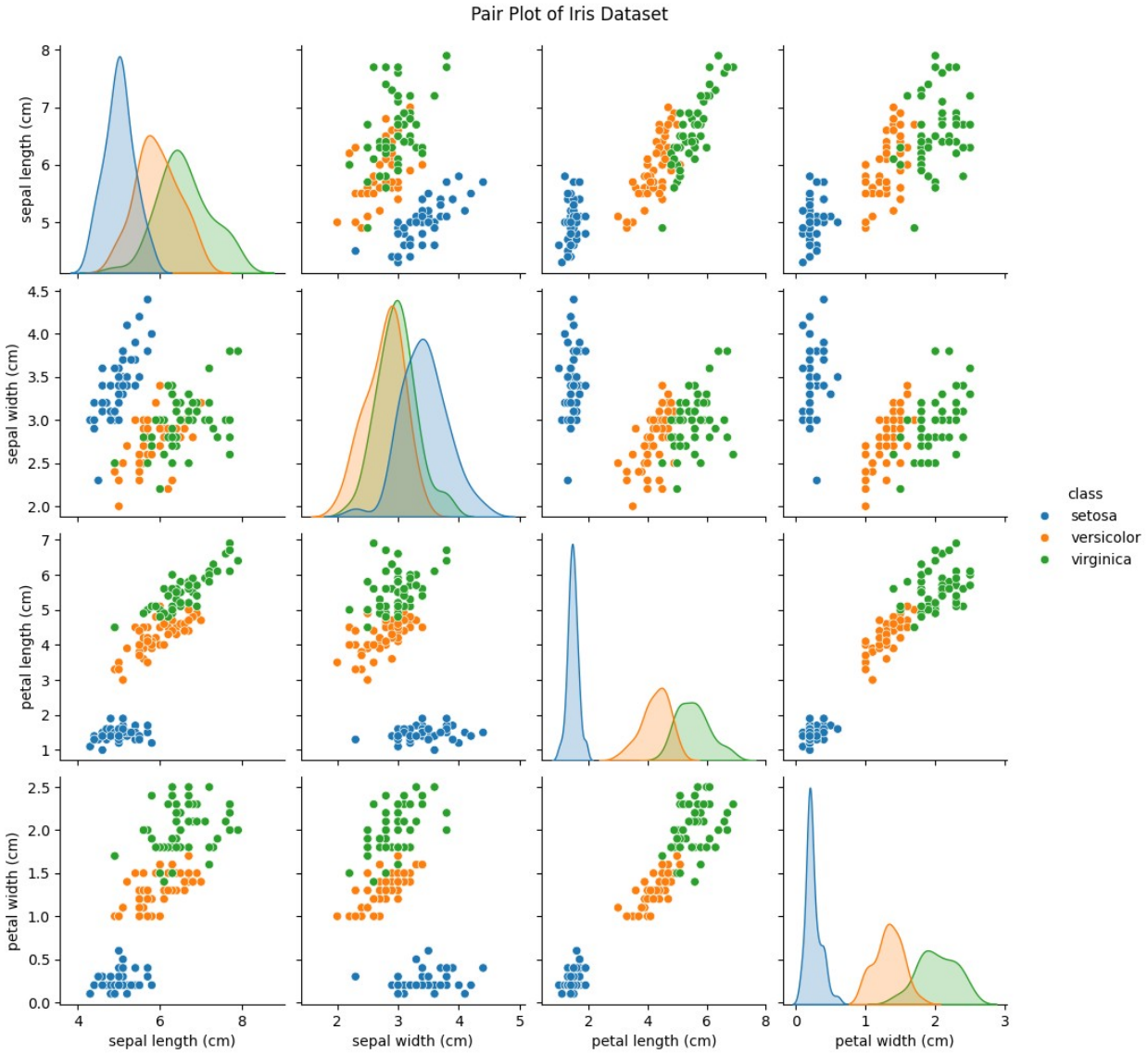


##3. Additional Visualizations

###3.1 Pair Plot

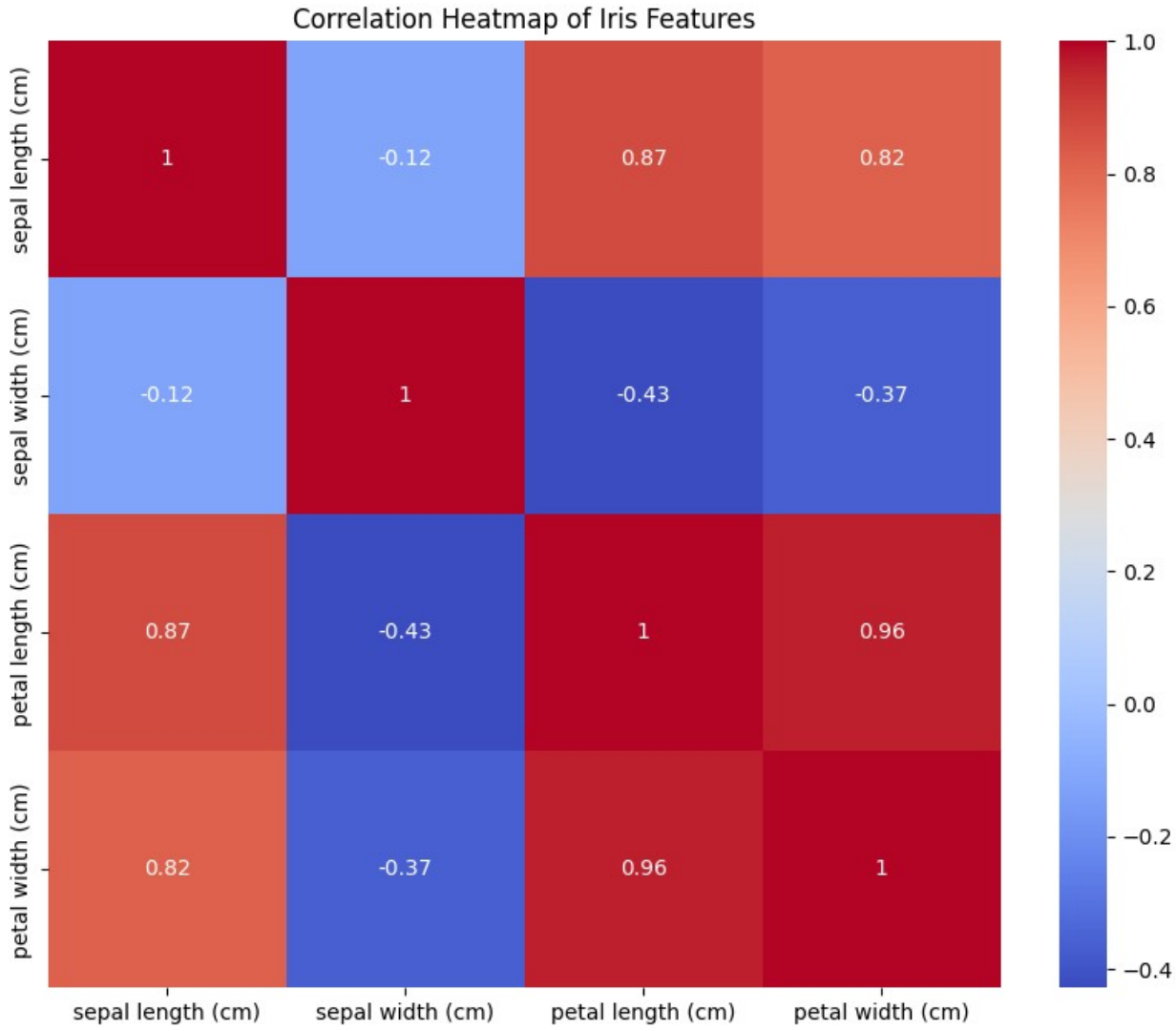
```
plt.figure(figsize=(12, 10))
sns.pairplot(df, hue='class', height=2.5)
plt.suptitle('Pair Plot of Iris Dataset', y=1.02)
plt.show()
```

```
<Figure size 1200x1000 with 0 Axes>
```



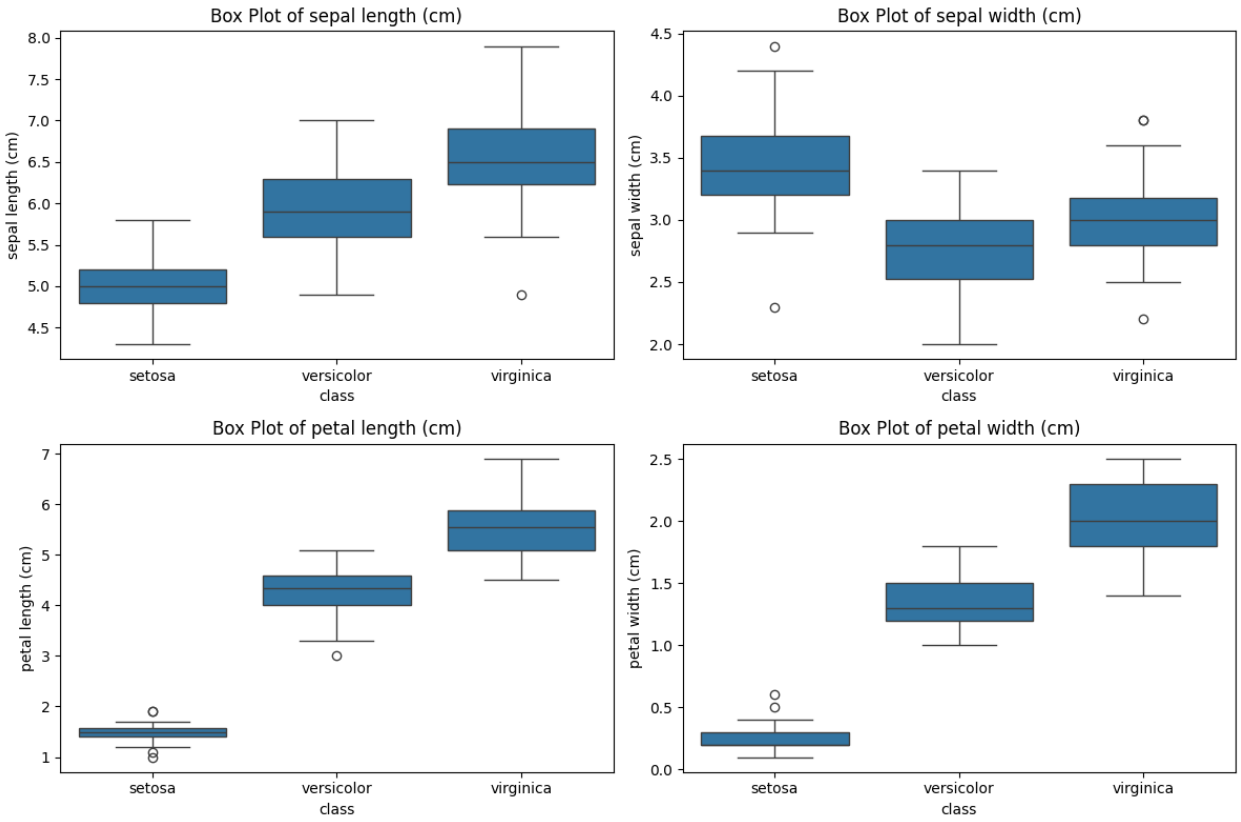
3.2 Correlation Heatmap

```
plt.figure(figsize=(10, 8))
sns.heatmap(df.drop('class', axis=1).corr(), annot=True,
cmap='coolwarm')
plt.title('Correlation Heatmap of Iris Features')
plt.show()
```



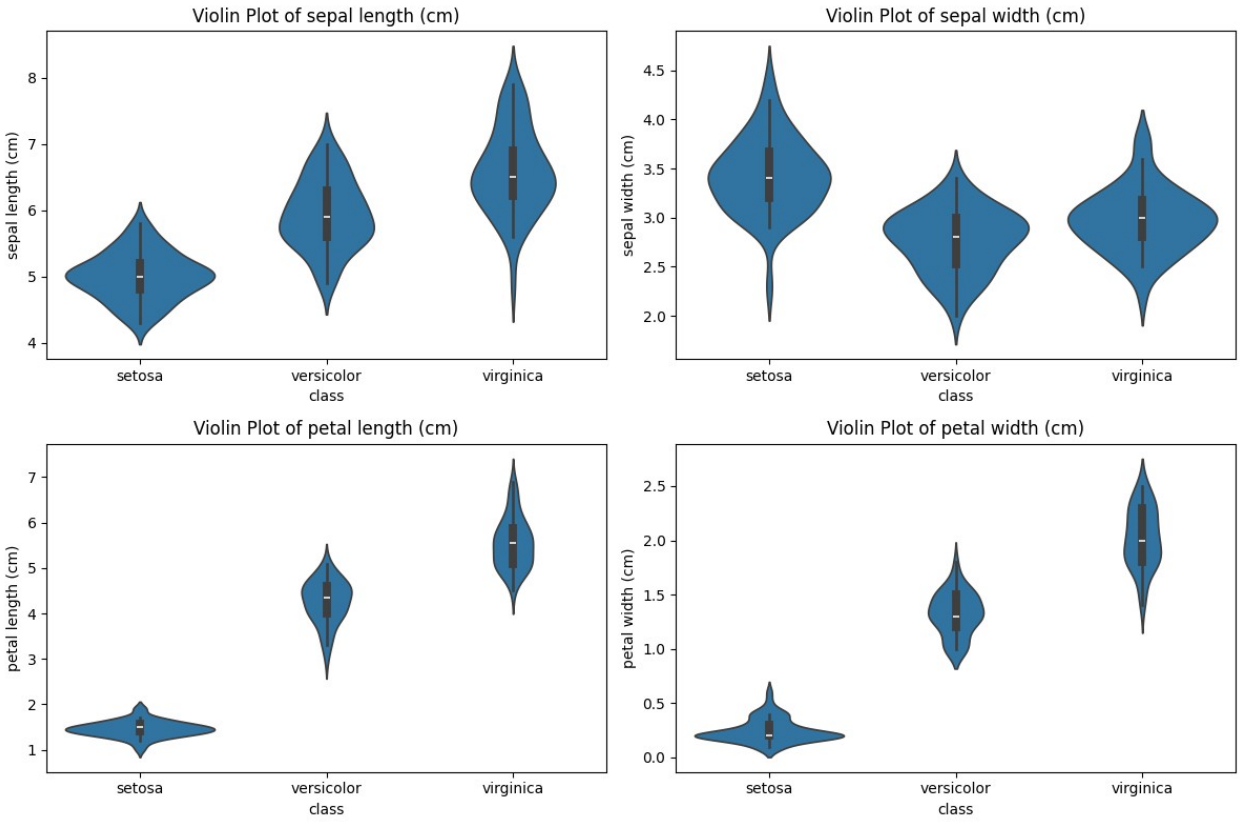
###3.3 Box Plots

```
plt.figure(figsize=(12, 8))
for i, feature in enumerate(iris.feature_names):
    plt.subplot(2, 2, i+1)
    sns.boxplot(x='class', y=feature, data=df)
    plt.title(f'Box Plot of {feature}')
plt.tight_layout()
plt.show()
```



###3.4 Violin Plots

```
plt.figure(figsize=(12, 8))
for i, feature in enumerate(iris.feature_names):
    plt.subplot(2, 2, i+1)
    sns.violinplot(x='class', y=feature, data=df)
    plt.title(f'Violin Plot of {feature}')
plt.tight_layout()
plt.show()
```



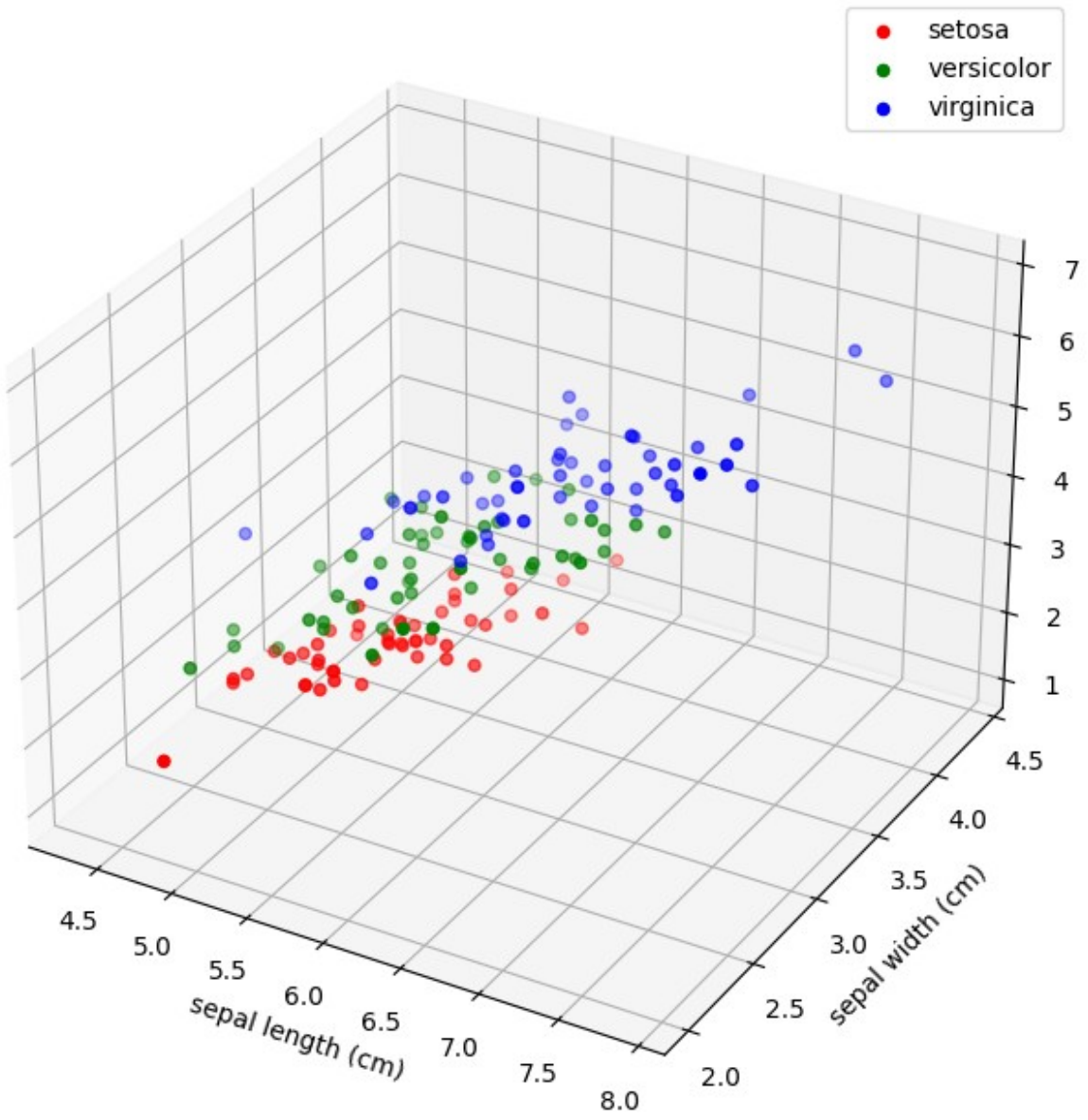
###3.5 3D Scatter Plot

```
from mpl_toolkits.mplot3d import Axes3D

fig = plt.figure(figsize=(12, 8))
ax = fig.add_subplot(111, projection='3d')
colors = ['r', 'g', 'b']

for i, species in enumerate(iris.target_names):
    species_data = df[df['class'] == species]
    ax.scatter(species_data.iloc[:, 0], species_data.iloc[:, 1],
species_data.iloc[:, 2],
               c=colors[i], label=species)

ax.set_xlabel(iris.feature_names[0])
ax.set_ylabel(iris.feature_names[1])
ax.set_zlabel(iris.feature_names[2])
ax.legend()
plt.title('3D Scatter Plot of Iris Dataset')
plt.show()
```


3D Scatter Plot of Iris Dataset
setosa
versicolor
virginica
sepal length (cm)
sepal width (cm)
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
2.0
2.5
3.0
3.5
4.0
4.5
1
2
3
4
5
6
7