IRIS FLOWER CLASSIFICATION

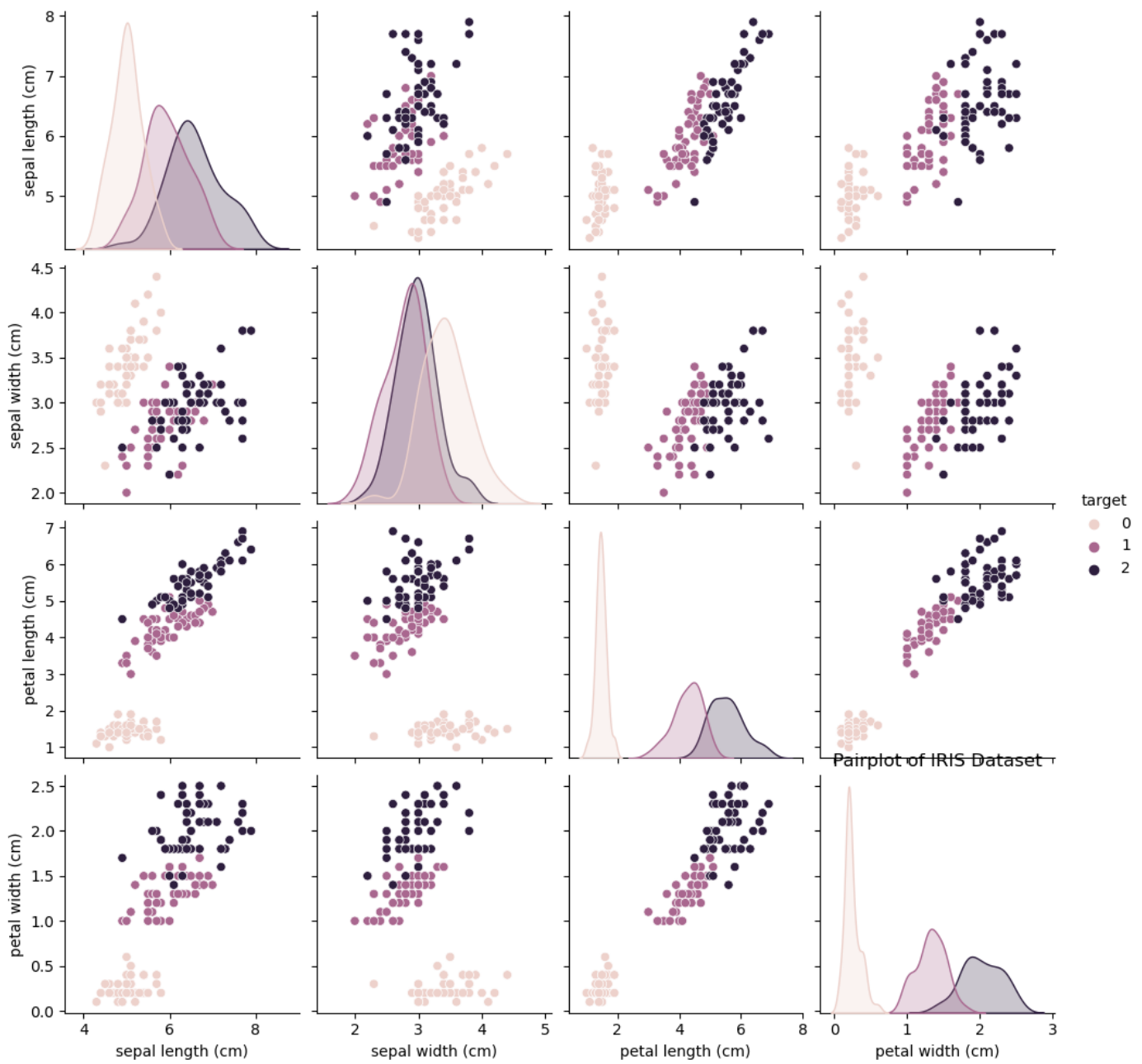
Iris flower has three species; setosa, versicolor, and virginica, which differs according to their measurements. Now assume that you have the measurements of the Iris flowers according to their species, and here your task is to train a machine learning model that can learn from the measurements of the iris species and classify them.

In [1]:
```
from sklearn.datasets import load_iris
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
```

In [2]:
```
# Load the IRIS dataset
iris = load_iris()
```

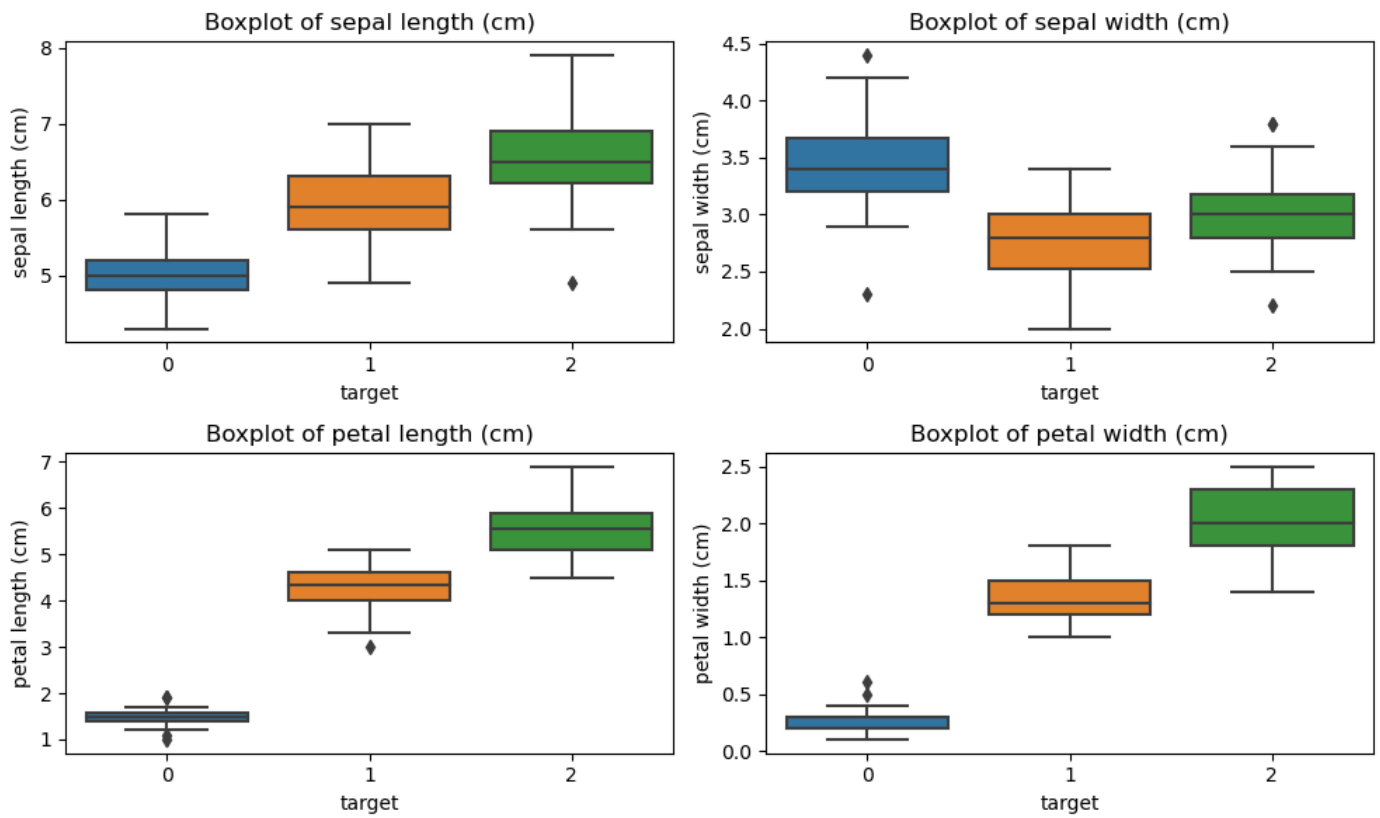
In [3]:
```
# Convert the dataset to a Pandas DataFrame for easier visualization
df = pd.DataFrame(data=iris.data, columns=iris.feature_names)
df['target'] = iris.target
```

In [4]:
```
# Pairplot: Visualize relationships between pairs of features
sns.pairplot(df, hue='target')
plt.title('Pairplot of IRIS Dataset')
plt.show()
```

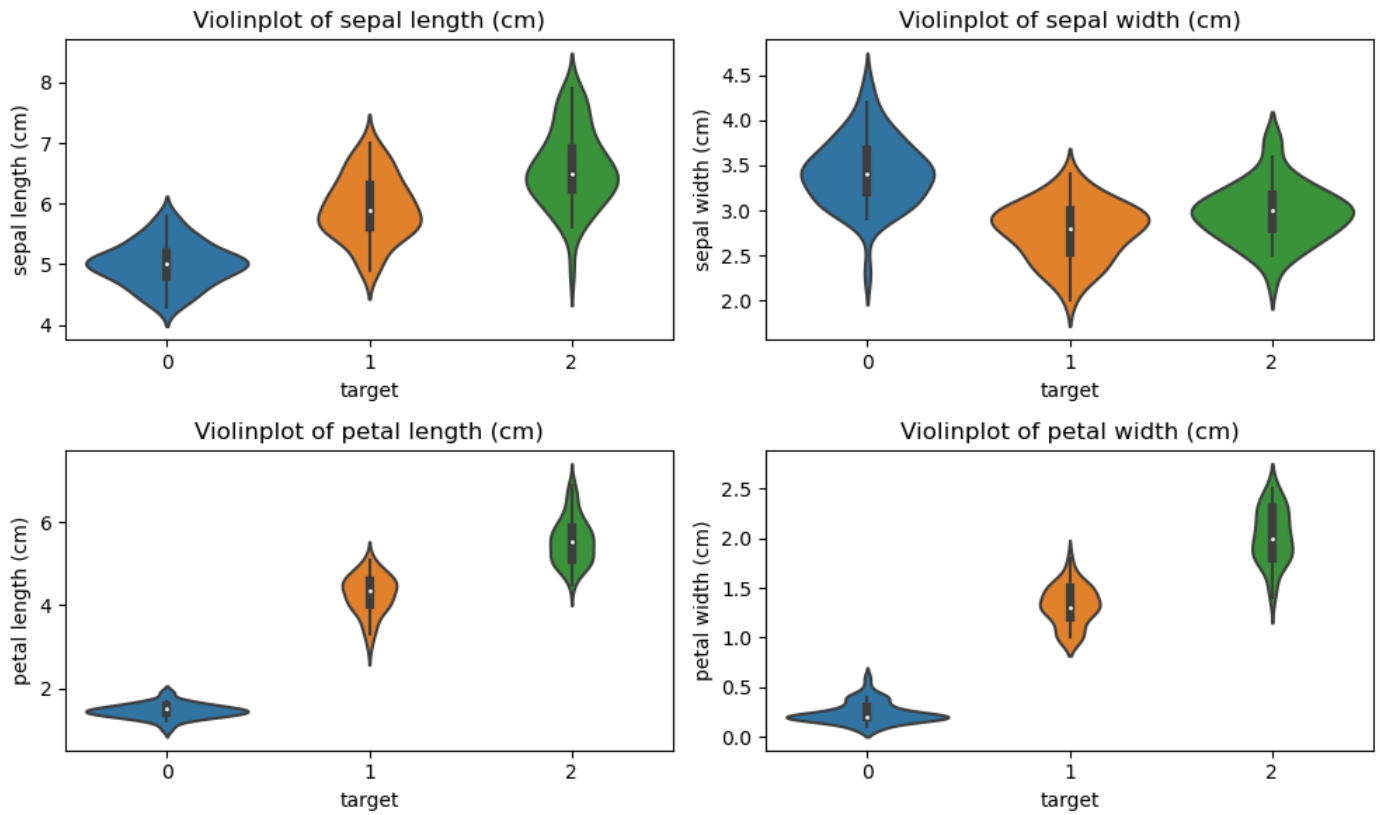


In [5]:

```
# Boxplot: Visualize the distribution of each feature across different classes
plt.figure(figsize=(10, 6))
for i, feature in enumerate(iris.feature_names):
    plt.subplot(2, 2, i+1)
    sns.boxplot(x='target', y=feature, data=df)
    plt.title(f'Boxplot of {feature}')
plt.tight_layout()
plt.show()
```

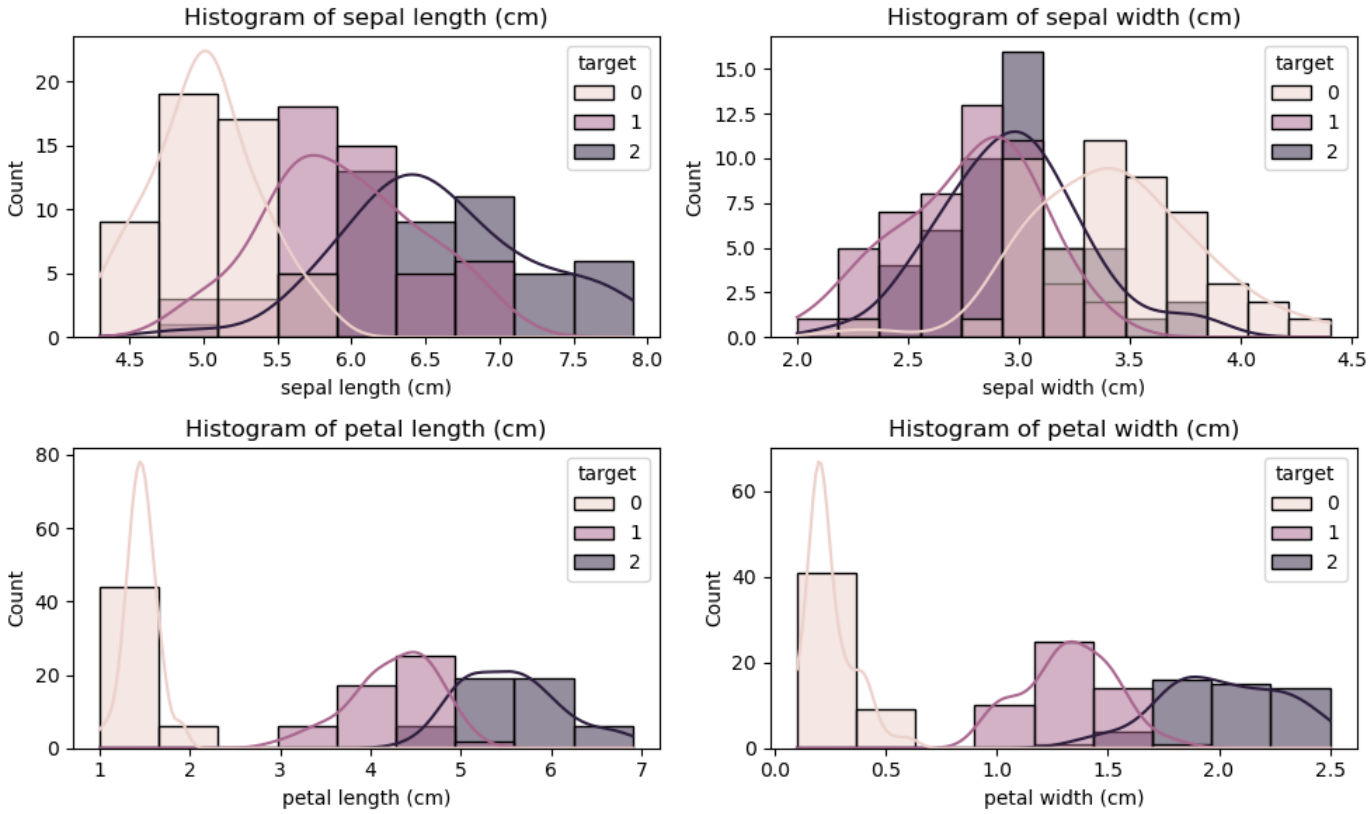


In [6]:
```
# Violinplot: Similar to boxplot, but provides a richer description of the data distribu
plt.figure(figsize=(10, 6))
for i, feature in enumerate(iris.feature_names):
    plt.subplot(2, 2, i+1)
    sns.violinplot(x='target', y=feature, data=df)
    plt.title(f'Violinplot of {feature}')
plt.tight_layout()
plt.show()
```



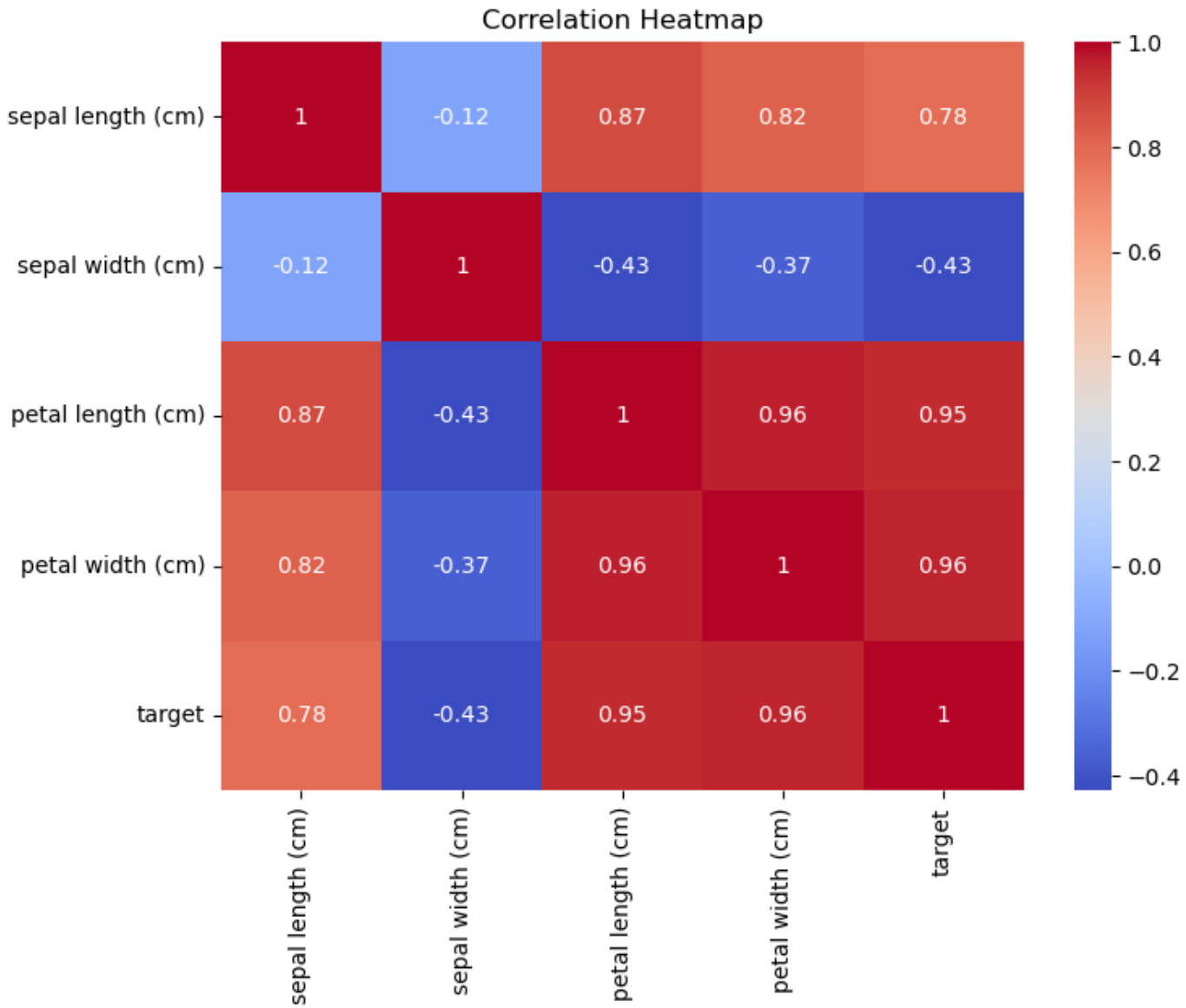
In [7]:
```
# Histogram: Visualize the distribution of each feature
plt.figure(figsize=(10, 6))
```

```
for i, feature in enumerate(iris.feature_names):
    plt.subplot(2, 2, i+1)
    sns.histplot(data=df, x=feature, hue='target', kde=True)
    plt.title(f'Histogram of {feature}')
plt.tight_layout()
plt.show()
```



In [8]:

```
# Correlation Heatmap: Visualize the correlation between features
plt.figure(figsize=(8, 6))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```



In [9]:
```
# Data Preparation

X = iris.data
y = iris.target
```

In [10]:
```
# Model Selection
model = RandomForestClassifier()
```

In [11]:
```
# Training and Evaluation
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42

model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy}")
```

```
Accuracy: 1.0
```

In [ ]: