```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.preprocessing import StandardScaler
import seaborn as sns

iris = load_iris()
X = iris.data
y = iris.target
feature_names = iris.feature_names
target_names = iris.target_names

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)

k_range = range(1, 20)
k_scores = []

for k in k_range:
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train_scaled, y_train)
    y_pred = knn.predict(X_test_scaled)
    k_scores.append(accuracy_score(y_test, y_pred))

plt.figure(figsize=(10, 6))
plt.plot(k_range, k_scores, marker='o')
plt.xlabel('Number of Neighbors (k)')
plt.ylabel('Accuracy')
plt.title('KNN Accuracy vs. k Value')
plt.grid(True)
plt.show()

optimal_k = k_range[np.argmax(k_scores)]
print(f"Optimal k: {optimal_k}")

knn = KNeighborsClassifier(n_neighbors=optimal_k)
knn.fit(X_train_scaled, y_train)
y_pred = knn.predict(X_test_scaled)

print(f"Accuracy with k={optimal_k}: {accuracy_score(y_test, y_pred):.4f}")

plt.figure(figsize=(8, 6))
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=target_names, yticklabels=target_names)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()

plt.figure(figsize=(12, 5))
for i, pair in enumerate([(0, 1), (2, 3)]):
    plt.subplot(1, 2, i+1)

    x_idx, y_idx = pair
    X_subset = X[:, [x_idx, y_idx]]

    X_subset_scaled = scaler.fit_transform(X_subset)

    knn = KNeighborsClassifier(n_neighbors=optimal_k)
    knn.fit(X_subset_scaled, y)

    h = 0.02
    x_min, x_max = X_subset_scaled[:, 0].min() - 1, X_subset_scaled[:, 0].max() + 1
    y_min, y_max = X_subset_scaled[:, 1].min() - 1, X_subset_scaled[:, 1].max() + 1
    xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))

    Z = knn.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)

    plt.contourf(xx, yy, Z, alpha=0.3, cmap=plt.cm.RdYlBu)
    scatter = plt.scatter(X_subset_scaled[:, 0], X_subset_scaled[:, 1], c=y, edgecolors='k', cmap=plt.cm.RdYlBu)

    plt.xlabel(f'Scaled {feature_names[x_idx]}')
    plt.ylabel(f'Scaled {feature_names[y_idx]}')
    plt.title(f'KNN Decision Boundaries - Features {x_idx+1} & {y_idx+1}')

plt.tight_layout()
plt.show()
```

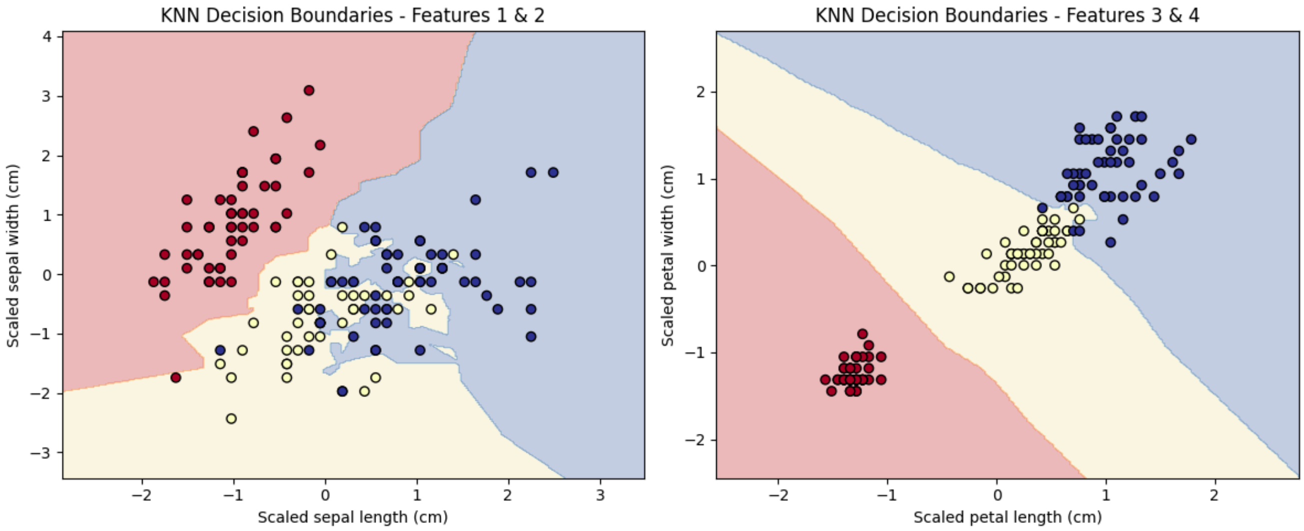


Figure_3.png

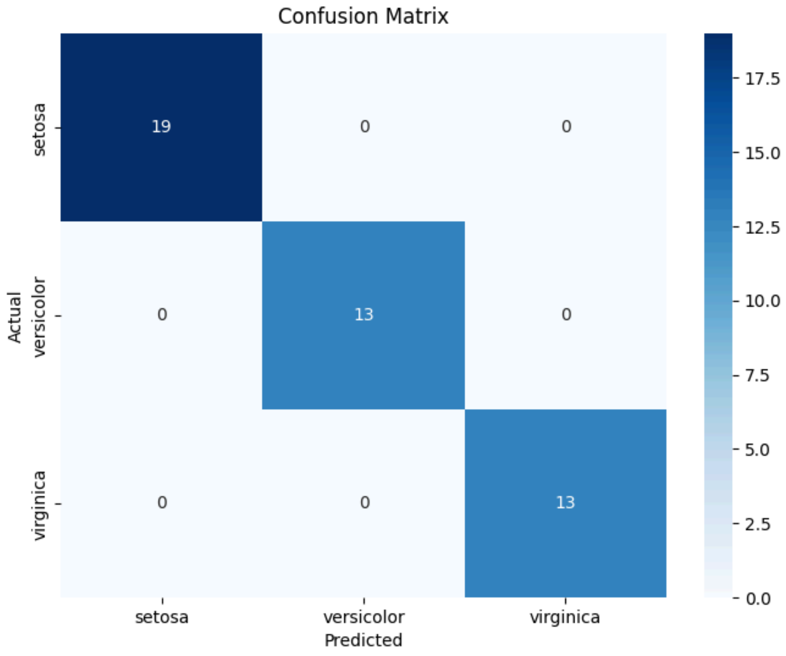
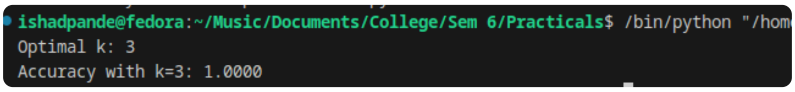


Figure_2.png



```
ishadpande@fedora:~/Music/Documents/College/Sem 6/Practicals$ /bin/python "/hom
Optimal k: 3
Accuracy with k=3: 1.0000
```

image-2.png