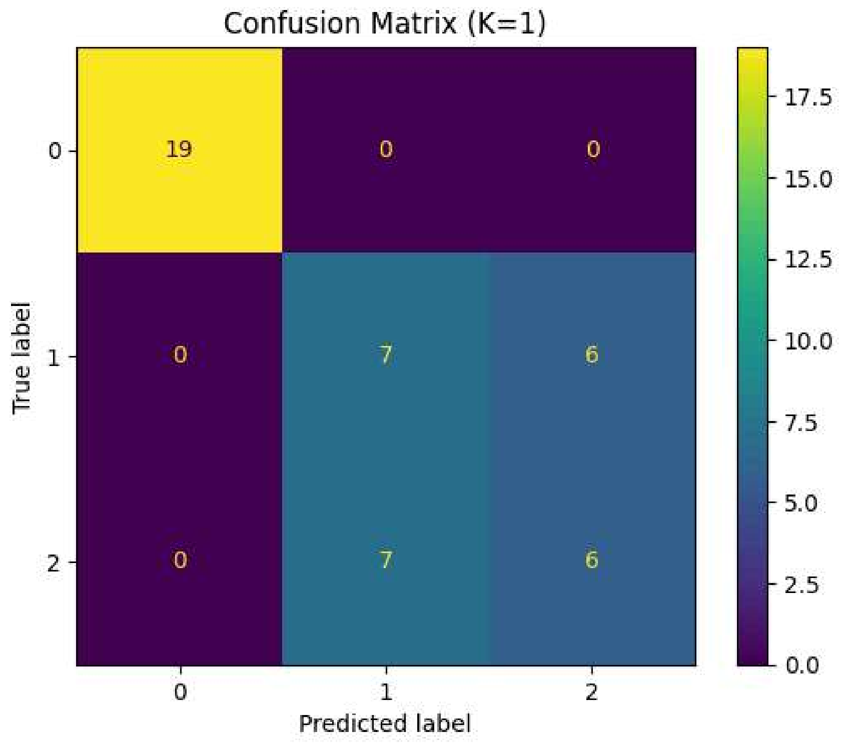
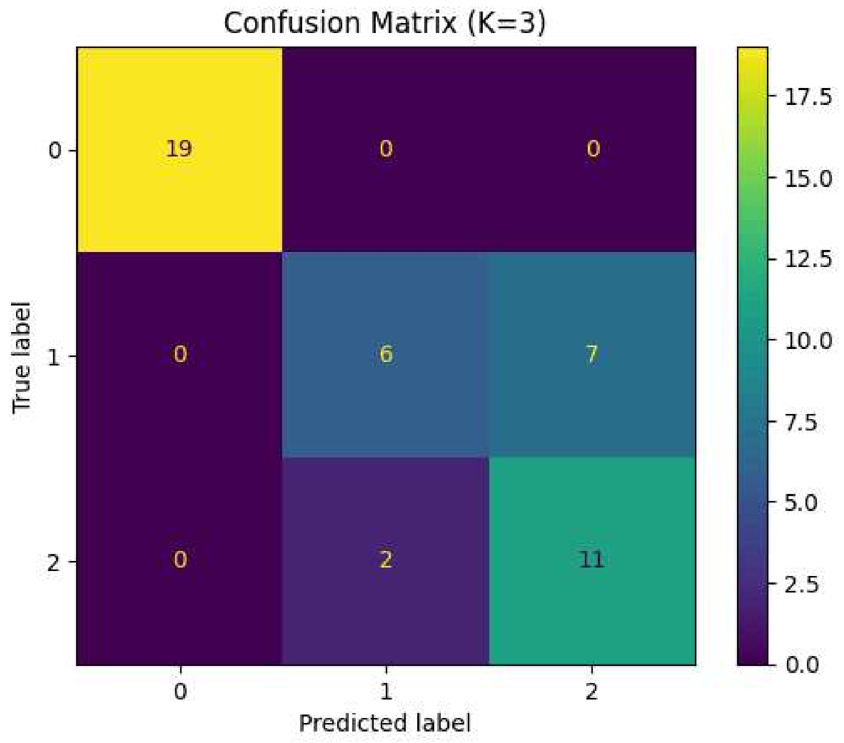
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, ConfusionMatrixDisplay
from matplotlib.colors import ListedColormap


iris = load_iris()
X = iris.data[:, :2]
y = iris.target


scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)


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


k_values = [1, 3, 5, 7]
for k in k_values:
    model = KNeighborsClassifier(n_neighbors=k)
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    acc = accuracy_score(y_test, y_pred)
    print(f"\nK={k}")
    print("Accuracy:", acc)
    print("Confusion Matrix:")
    print(confusion_matrix(y_test, y_pred))
    disp = ConfusionMatrixDisplay.from_predictions(y_test, y_pred)
    disp.ax_.set_title(f'Confusion Matrix (K={k})')
    plt.show()
```

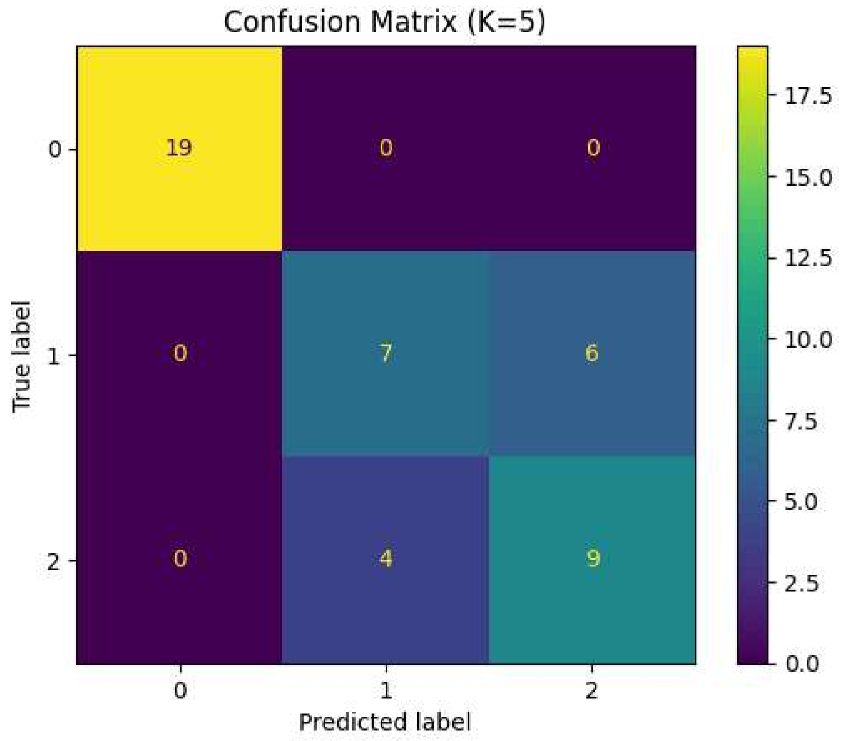
```
K=1
Accuracy: 0.7111111111111111
Confusion Matrix:
[[19  0  0]
 [ 0  7  6]
 [ 0  7  6]]
```



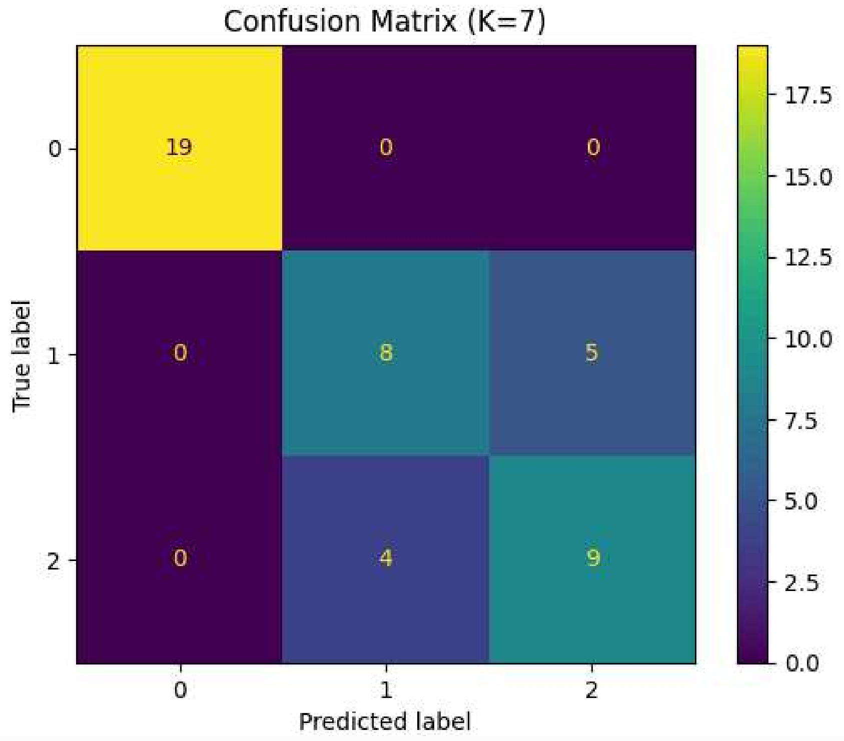
```
K=3
Accuracy: 0.8
Confusion Matrix:
[[19  0  0]
 [ 0  6  7]
 [ 0  2 11]]
```



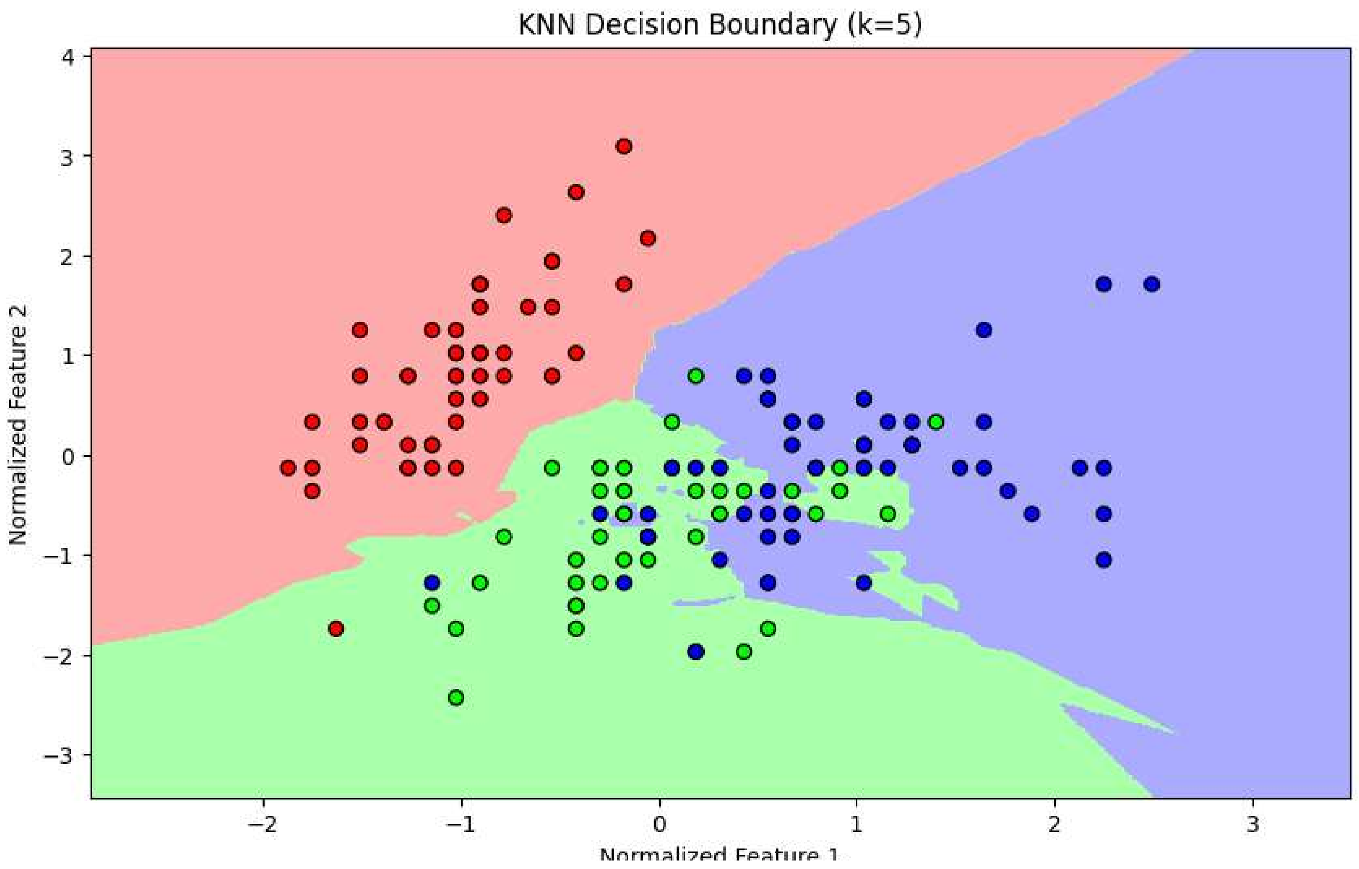
```
K=5
Accuracy: 0.7777777777777778
Confusion Matrix:
[[19  0  0]
 [ 0  7  6]
 [ 0  4  9]]
```



```
K=7
Accuracy: 0.8
Confusion Matrix:
[[19  0  0]
 [ 0  8  5]
 [ 0  4  9]]
```



```
model = KNeighborsClassifier(n_neighbors=5)
model.fit(X_train, y_train)

h = .02  # step size in mesh
x_min, x_max = X_scaled[:, 0].min() - 1, X_scaled[:, 0].max() + 1
y_min, y_max = X_scaled[:, 1].min() - 1, X_scaled[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h),
                     np.arange(y_min, y_max, h))
Z = model.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)

plt.figure(figsize=(10, 6))
cmap_light = ListedColormap(['#FFAAAA', '#AAFFAA', '#AAAAFF'])
cmap_bold = ListedColormap(['#FF0000', '#00FF00', '#0000FF'])
plt.contourf(xx, yy, Z, cmap=cmap_light)
plt.scatter(X_scaled[:, 0], X_scaled[:, 1], c=y, cmap=cmap_bold, edgecolor='k', s=40)
plt.title("KNN Decision Boundary (k=5)")
plt.xlabel("Normalized Feature 1")
plt.ylabel("Normalized Feature 2")
plt.show()
```


KNN Decision Boundary (k=5)
Normalized Feature 2
Normalized Feature 1