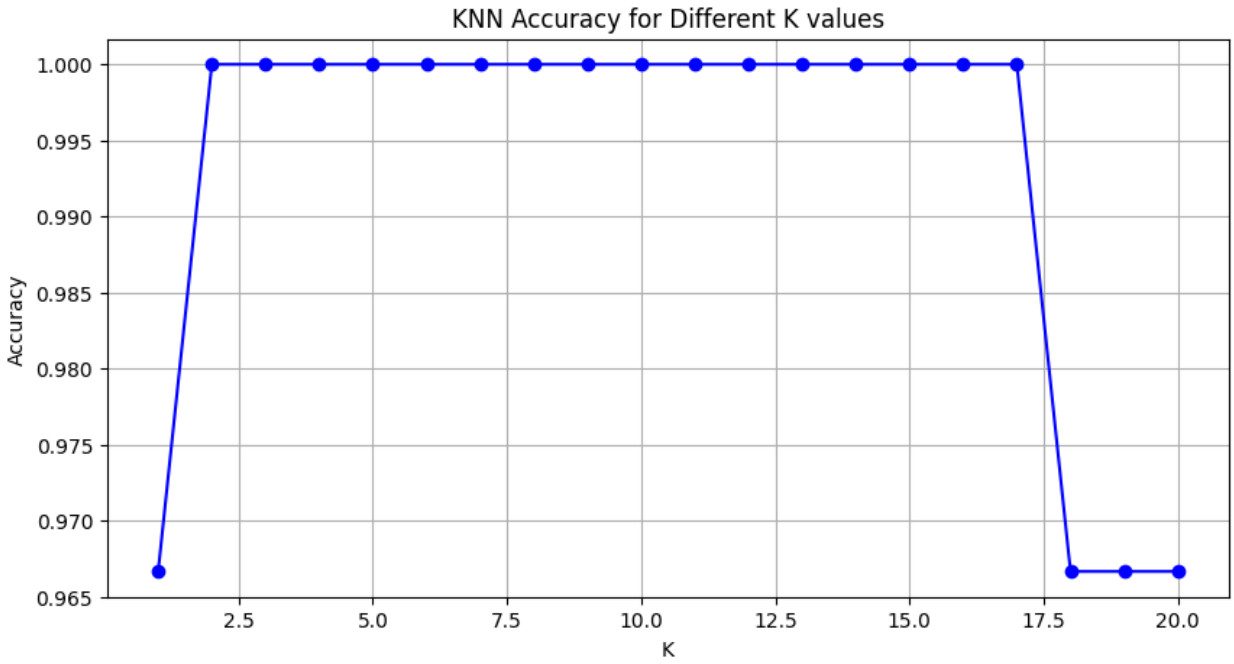
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
from matplotlib.colors import ListedColormap
df = pd.read_csv('/content/Iris.csv')
df = df.drop(columns=['Id'])
print("Dataset Head:\n", df.head())
X = df.drop('Species', axis=1)
y = df['Species']
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y,
test_size=0.2, random_state=42)
accuracy_scores = []
k_values = range(1, 21)
for k in k_values:
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train, y_train)
    y_pred = knn.predict(X_test)
    acc = accuracy_score(y_test, y_pred)
    accuracy_scores.append(acc)
plt.figure(figsize=(10, 5))
plt.plot(k_values, accuracy_scores, marker='o', color='blue')
plt.title('KNN Accuracy for Different K values')
plt.xlabel('K')
plt.ylabel('Accuracy')
plt.grid(True)
plt.show()
best_k = k_values[np.argmax(accuracy_scores)]
print(f"\n Best K: {best_k} with Accuracy:
{max(accuracy_scores):.2f}")
knn_final = KNeighborsClassifier(n_neighbors=best_k)
knn_final.fit(X_train, y_train)
y_final_pred = knn_final.predict(X_test)
print("\n Confusion Matrix:")
print(confusion_matrix(y_test, y_final_pred))
print("\n Classification Report:")
print(classification_report(y_test, y_final_pred))
def plot_decision_boundaries(X, y, model, title):
    X = X[:, :2]
    h = .02
    x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
    y_min, y_max = X[:, 1].min() - 1, X[:, 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()])
    if isinstance(Z[0], str):
        le = LabelEncoder()
        Z = le.fit_transform(Z)
    Z = Z.reshape(xx.shape)
    cmap_light = ListedColormap(['#FFAAAA', '#AAFFAA', '#AAAAFF'])
    cmap_bold = ListedColormap(['#FF0000', '#00FF00', '#0000FF'])
    plt.figure(figsize=(8, 6))
    plt.contourf(xx, yy, Z, cmap=cmap_light, alpha=0.8)
    sns.scatterplot(x=X[:, 0], y=X[:, 1], hue=y, palette=cmap_bold,
edgecolor='k')
    plt.title(title)
    plt.xlabel('Feature 1')
    plt.ylabel('Feature 2')
    plt.show()
X_reduced = X_scaled[:, :2]
X_train_r, X_test_r, y_train_r, y_test_r = train_test_split(X_reduced,
y, test_size=0.2, random_state=42)
knn_2d = KNeighborsClassifier(n_neighbors=best_k)
knn_2d.fit(X_train_r, y_train_r)
le = LabelEncoder()
y_encoded = le.fit_transform(y_test_r)
plot_decision_boundaries(X_test_r, y_encoded, knn_2d, "KNN Decision
Boundary (2 Features)")
```

```
Dataset Head:
    SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm
Species
0            5.1           3.5            1.4           0.2  Iris-
setosa
1            4.9           3.0            1.4           0.2  Iris-
setosa
2            4.7           3.2            1.3           0.2  Iris-
setosa
3            4.6           3.1            1.5           0.2  Iris-
setosa
4            5.0           3.6            1.4           0.2  Iris-
setosa
```



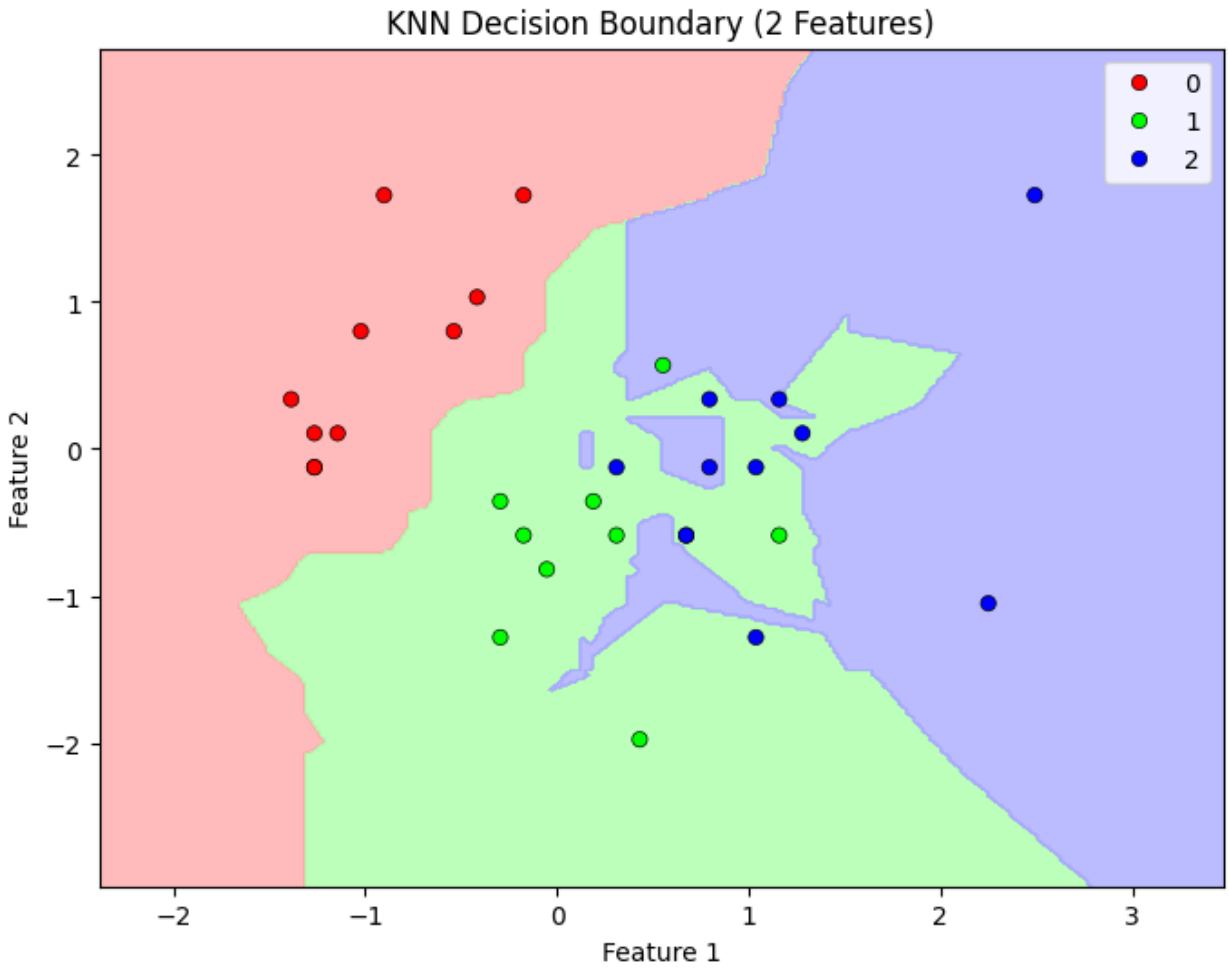

```
 Best K: 2 with Accuracy: 1.00

 Confusion Matrix:
[[10  0  0]
 [ 0  9  0]
 [ 0  0 11]]

 Classification Report:
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        10
Iris-versicolor       1.00      1.00      1.00         9
 Iris-virginica       1.00      1.00      1.00        11

       accuracy                           1.00        30
      macro avg       1.00      1.00      1.00        30
   weighted avg       1.00      1.00      1.00        30
```


KNN Decision Boundary (2 Features)
0
1
2
Feature 2
Feature 1
2
1
0
−1
−2
−2
−1
0
1
2
3