```
from google.colab import files
uploaded = files.upload()

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.preprocessing import StandardScaler, label_binarize
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
from sklearn.multiclass import OneVsRestClassifier
from itertools import cycle

df = pd.read_csv('IRIS.csv')
display(df.head())
df.info()
print(df.isnull().sum())

plt.figure(figsize=(6,4))
sns.countplot(x='species', data=df, palette='pastel')
plt.title("Distribusi Jumlah Data per Kelas")
plt.show()

sns.pairplot(df, hue='species')
plt.suptitle("Pairplot Hubungan Antar Fitur", y=1.02)
plt.show()

plt.figure(figsize=(8,6))
sns.heatmap(df.drop('species', axis=1).corr(), annot=True, cmap='coolwarm')
plt.title("Heatmap Korelasi Antar Fitur")
plt.show()

X = df.drop('species', axis=1)
y = df['species']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_sta
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

model_log = LogisticRegression(max_iter=200)
model_tree = DecisionTreeClassifier(random_state=42)
model_log.fit(X_train, y_train)
model_tree.fit(X_train, y_train)
y_pred_log = model_log.predict(X_test)
y_pred_tree = model_tree.predict(X_test)

fig, axes = plt.subplots(1, 2, figsize=(12,5))
sns.heatmap(confusion_matrix(y_test, y_pred_log), annot=True, fmt='d', cmap='Blues
axes[0].set_title("Confusion Matrix - Logistic Regression")
sns.heatmap(confusion_matrix(y_test, y_pred_tree), annot=True, fmt='d', cmap='Greer
axes[1].set_title("Confusion Matrix - Decision Tree")
plt.tight_layout()
plt.show()

print(classification_report(y_test, y_pred_log))
print(classification_report(y_test, y_pred_tree))

y_bin = label_binarize(y, classes=df['species'].unique())
n_classes = y_bin.shape[1]
```

```
X_train, X_test, y_train_bin, y_test_bin = train_test_split(X, y_bin, test_size=0.2
clf = OneVsRestClassifier(LogisticRegression(max_iter=200))
y_score = clf.fit(X_train, y_train_bin).predict_proba(X_test)

fpr = dict()
tpr = dict()
roc_auc = dict()
plt.figure(figsize=(8,6))
colors = cycle(['blue', 'red', 'green'])
for i, color in zip(range(n_classes), colors):
    fpr[i], tpr[i], _ = roc_curve(y_test_bin[:, i], y_score[:, i])
    roc_auc[i] = auc(fpr[i], tpr[i])
    plt.plot(fpr[i], tpr[i], color=color, lw=2, label=f'Kelas {i+1} (AUC = {roc_auc
plt.plot([0, 1], [0, 1], 'k--', lw=2)
plt.title("ROC Curve - Logistic Regression")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.legend(loc="lower right")
plt.show()

akurasi_log = accuracy_score(y_test, y_pred_log)
akurasi_tree = accuracy_score(y_test, y_pred_tree)
precision_log = precision_score(y_test, y_pred_log, average='macro')
precision_tree = precision_score(y_test, y_pred_tree, average='macro')
recall_log = recall_score(y_test, y_pred_log, average='macro')
recall_tree = recall_score(y_test, y_pred_tree, average='macro')
f1_log = f1_score(y_test, y_pred_log, average='macro')
f1_tree = f1_score(y_test, y_pred_tree, average='macro')

hasil = pd.DataFrame({
    'Model': ['Logistic Regression', 'Decision Tree'],
    'Akurasi': [akurasi_log, akurasi_tree],
    'Precision': [precision_log, precision_tree],
    'Recall': [recall_log, recall_tree],
    'F1-score': [f1_log, f1_tree]
})
display(hasil)

plt.figure(figsize=(8,5))
sns.barplot(x='Model', y='Akurasi', data=hasil, palette='Set2')
plt.title('Perbandingan Akurasi Model')
plt.ylim(0,1)
plt.show()
```