Program:

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

import numpy as np

from sklearn.model\_selection import train\_test\_split

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import classification\_report, accuracy\_score, confusion\_matrix, roc\_curve, roc\_auc\_score

from sklearn.preprocessing import StandardScaler

from sklearn.decomposition import PCA

from pandas.plotting import scatter\_matrix, parallel\_coordinates

from mpl\_toolkits.mplot3d import Axes3D

sns.set(style='whitegrid', palette='pastel', font\_scale=1.1)

data = {

    'MonthlyCharges': [29.85, 56.95, 53.85, 42.30, 70.70, 99.65, 89.10, 65.80],

    'Tenure': [1, 34, 2, 45, 5, 10, 3, 12],

    'ContractType': [0, 1, 0, 1, 0, 1, 0, 1],

    'Churn': [1, 0, 1, 0, 1, 0, 1, 0]

}

df = pd.DataFrame(data)

X = df[['MonthlyCharges', 'Tenure', 'ContractType']]

y = df['Churn']

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.25, random\_state=42)

model = DecisionTreeClassifier(criterion='entropy', max\_depth=3, random\_state=42)

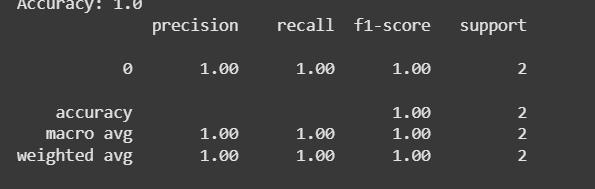
model.fit(X\_train, y\_train)

y\_pred = model.predict(X\_test)

y\_prob = model.predict\_proba(X\_test)[:, 1]

print("Accuracy:", accuracy\_score(y\_test, y\_pred))

print(classification\_report(y\_test, y\_pred))



cm = confusion\_matrix(y\_test, y\_pred)

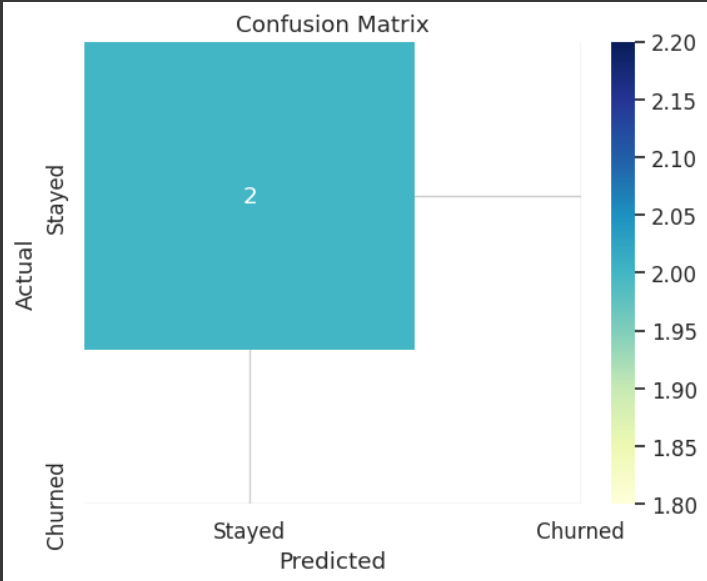
sns.heatmap(cm, annot=True, fmt='d', cmap='YlGnBu', xticklabels=['Stayed', 'Churned'], yticklabels=['Stayed', 'Churned'])

plt.title('Confusion Matrix')

plt.xlabel('Predicted')

plt.ylabel('Actual')

plt.show()



fpr, tpr, \_ = roc\_curve(y\_test, y\_prob)

roc\_auc = roc\_auc\_score(y\_test, y\_prob)

plt.figure(figsize=(8, 5))

plt.plot(fpr, tpr, label=f'AUC = {roc\_auc:.2f}', color='navy')

plt.plot([0, 1], [0, 1], linestyle='--', color='gray')

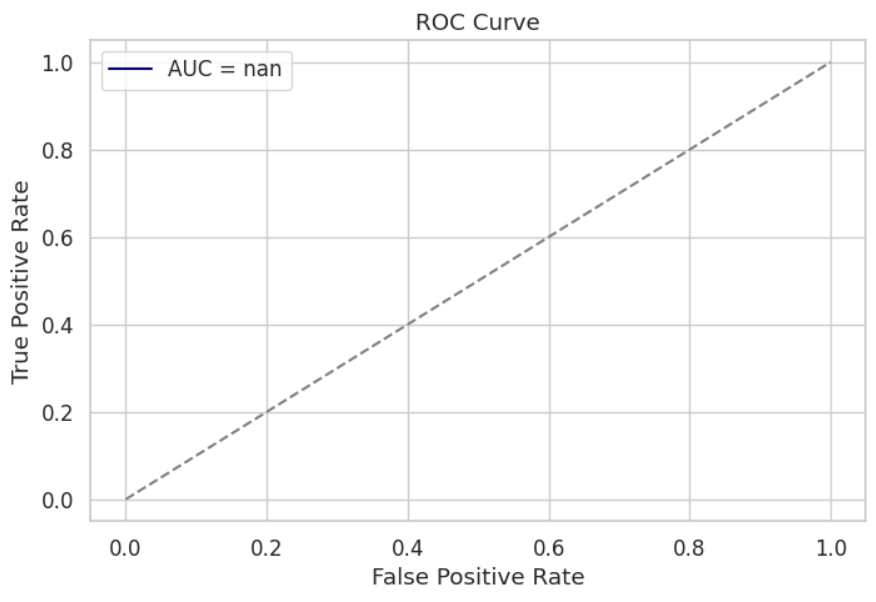
plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('ROC Curve')

plt.legend()

plt.show()

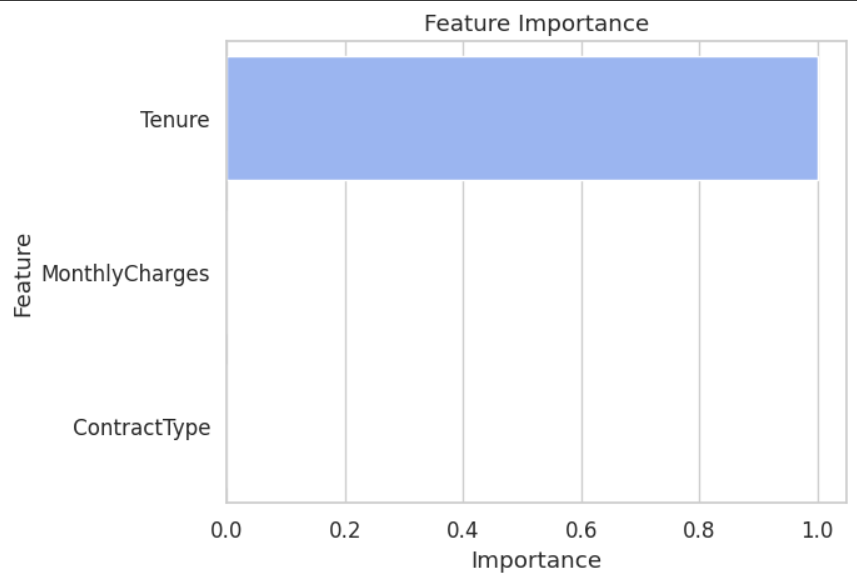


feature\_importance = pd.DataFrame({'Feature': X.columns, 'Importance': model.feature\_importances\_}).sort\_values(by='Importance', ascending=False)

sns.barplot(data=feature\_importance, x='Importance', y='Feature', palette='coolwarm')

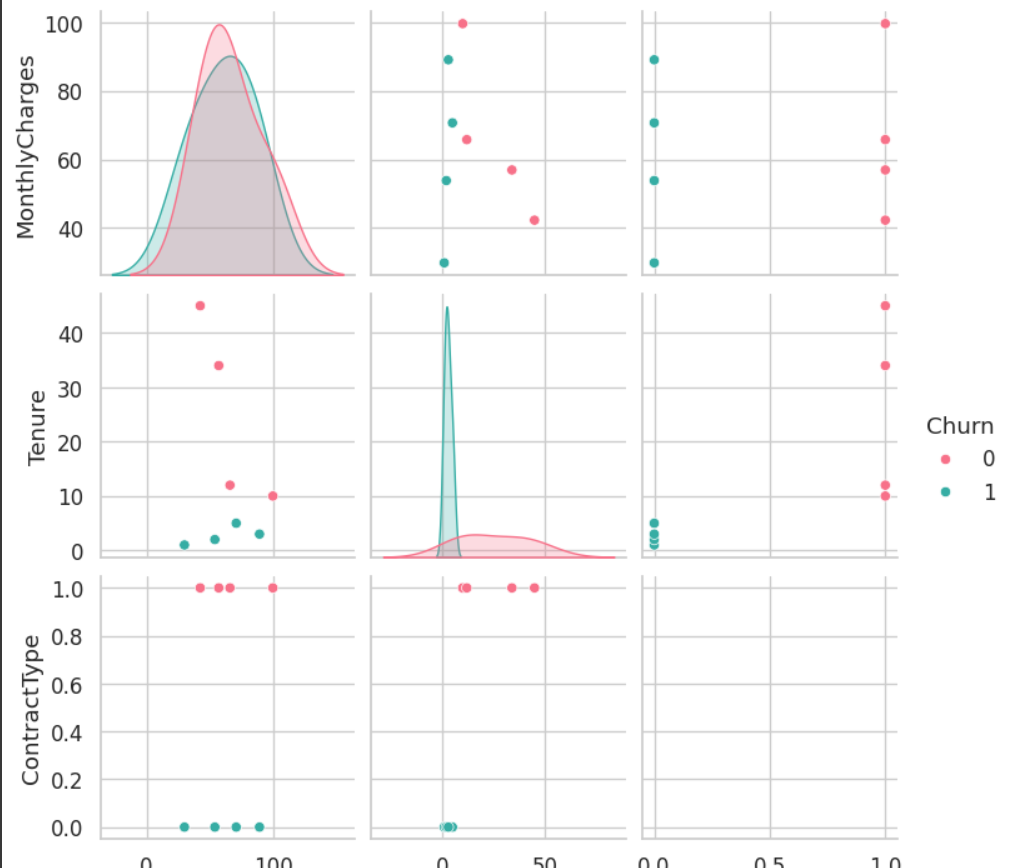
plt.title('Feature Importance')

plt.show()



sns.pairplot(df, hue='Churn', palette='husl')

plt.show()

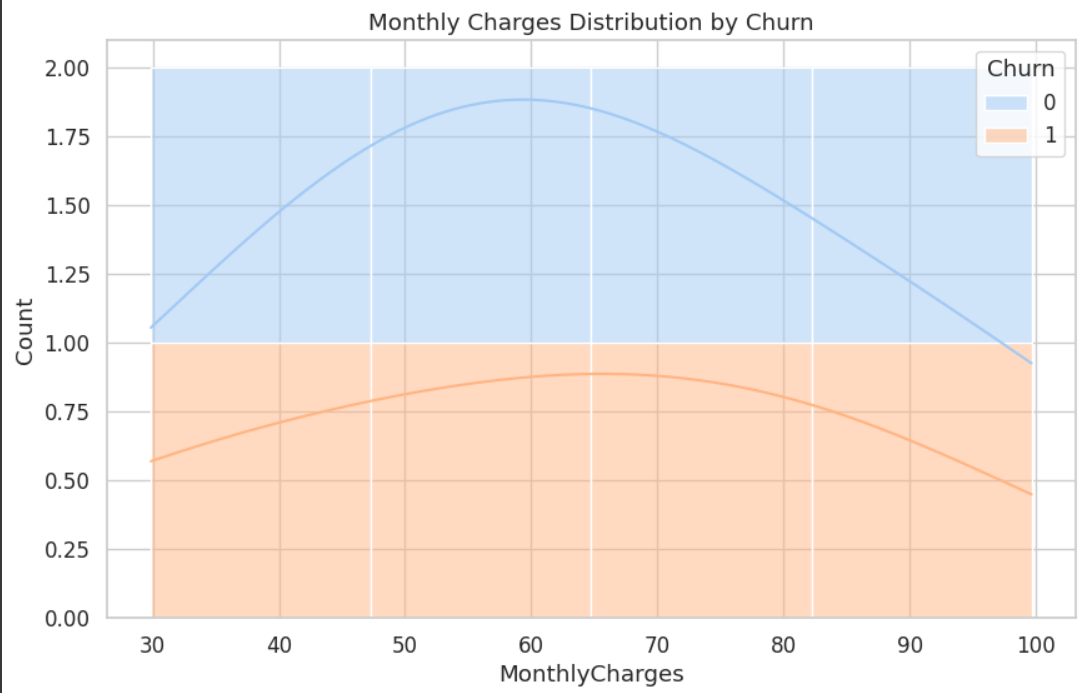


plt.figure(figsize=(10, 6))

sns.histplot(data=df, x='MonthlyCharges', hue='Churn', kde=True, multiple='stack')

plt.title('Monthly Charges Distribution by Churn')

plt.show()

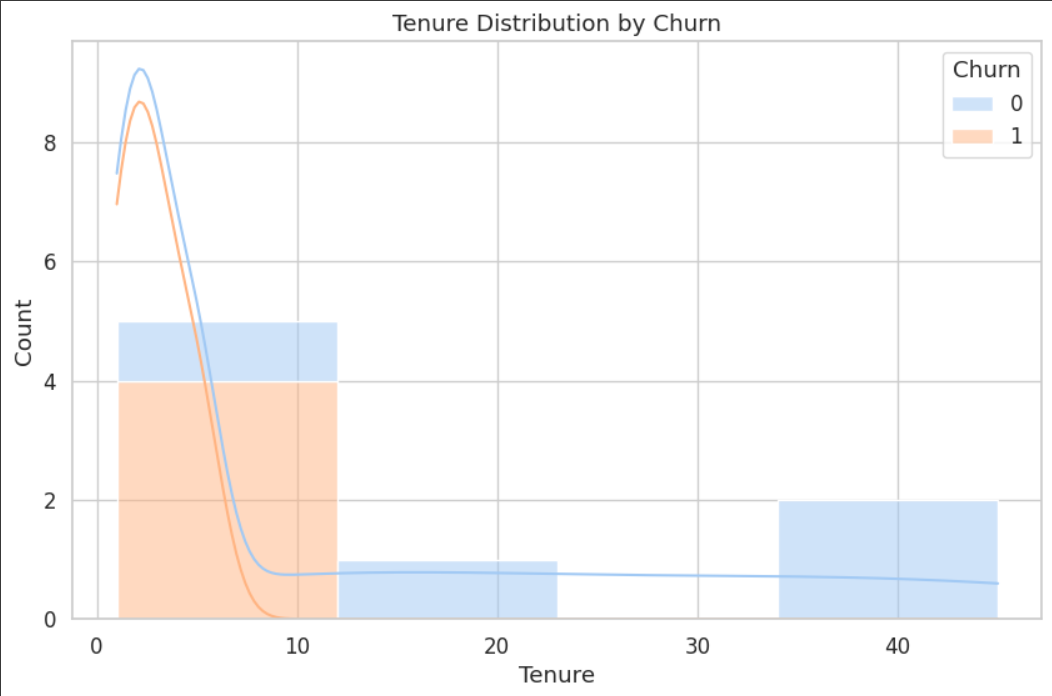


plt.figure(figsize=(10, 6))

sns.histplot(data=df, x='Tenure', hue='Churn', kde=True, multiple='stack')

plt.title('Tenure Distribution by Churn')

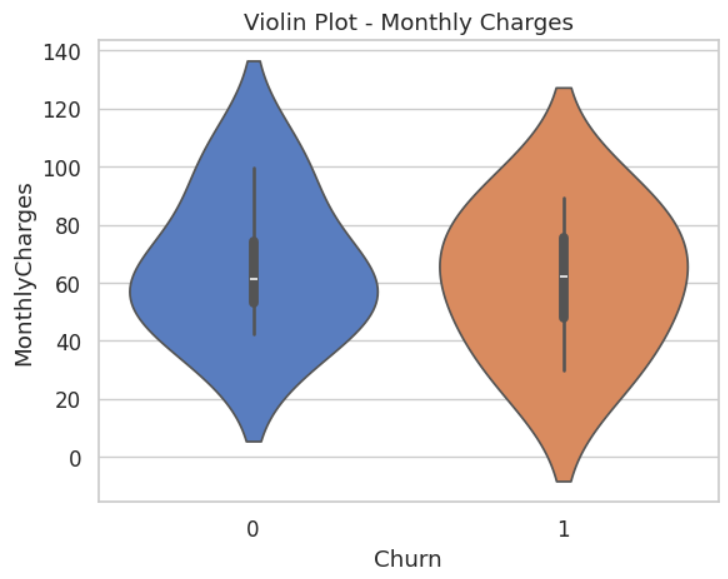
plt.show()



sns.violinplot(data=df, x='Churn', y='MonthlyCharges', palette='muted')

plt.title('Violin Plot - Monthly Charges')

plt.show()



plt.figure(figsize=(8, 5))

sns.heatmap(df.corr(), annot=True, cmap='coolwarm', linewidths=0.5)

plt.title('Correlation Matrix')

plt.show()

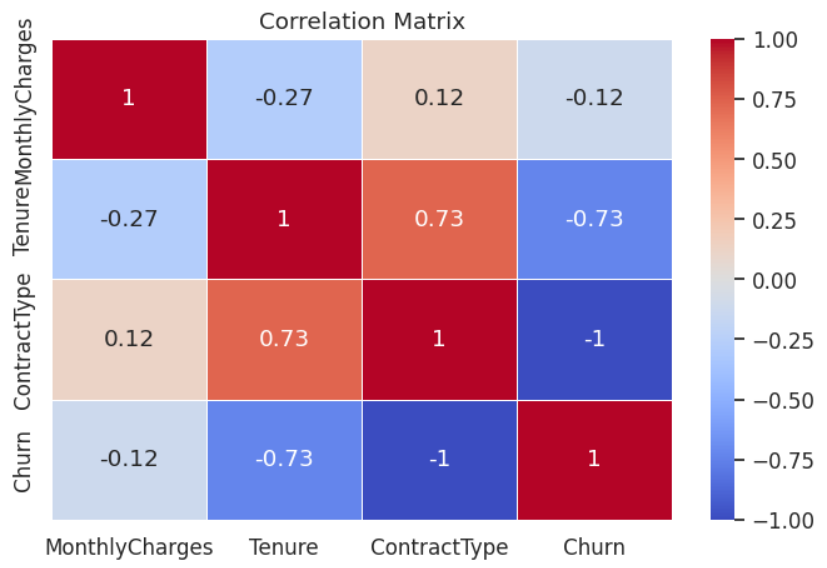


fig = plt.figure(figsize=(8, 6))

ax = fig.add\_subplot(111, projection='3d')

ax.scatter(df['MonthlyCharges'], df['Tenure'], df['ContractType'], c=df['Churn'], cmap='coolwarm', s=100)

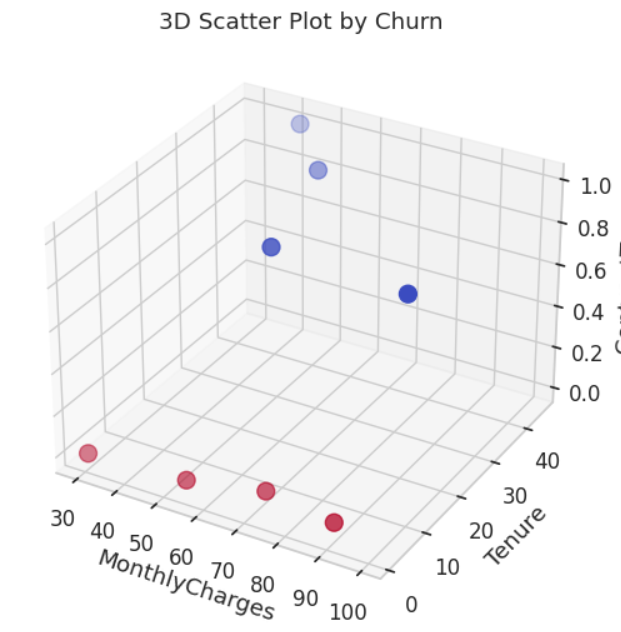
ax.set\_xlabel('MonthlyCharges')

ax.set\_ylabel('Tenure')

ax.set\_zlabel('ContractType')

plt.title('3D Scatter Plot by Churn')

plt.show()



df\_scaled = pd.DataFrame(X\_scaled, columns=X.columns)

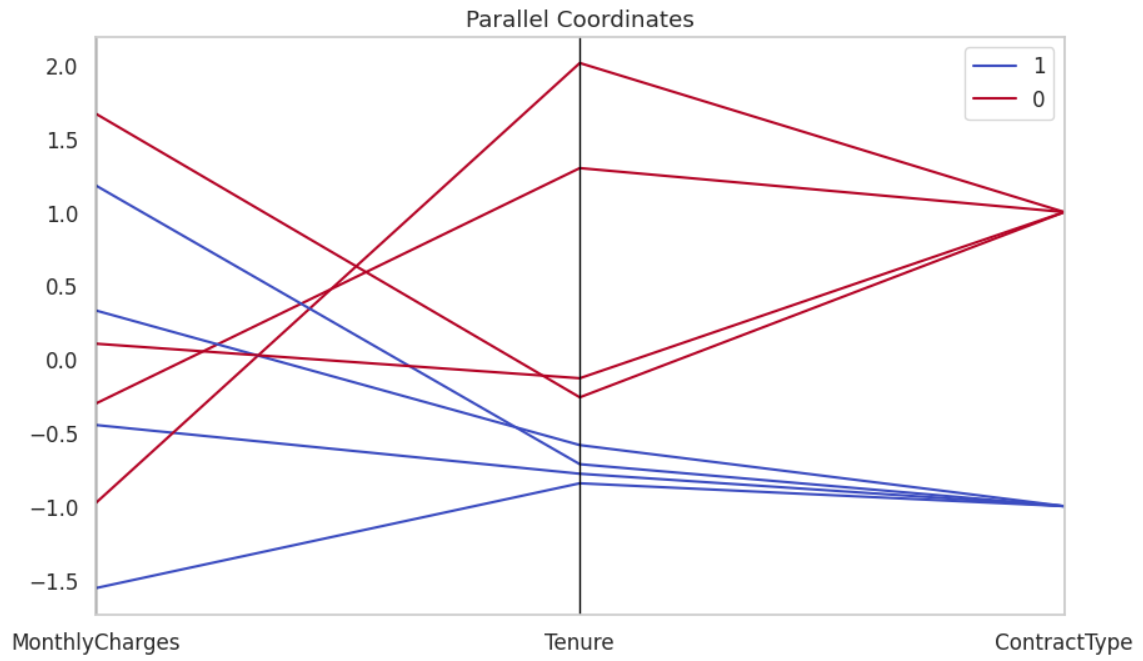
df\_scaled['Churn'] = y.values

plt.figure(figsize=(10, 6))

parallel\_coordinates(df\_scaled, class\_column='Churn', colormap='coolwarm')

plt.title('Parallel Coordinates')

plt.show()



pca = PCA(n\_components=2)

X\_pca = pca.fit\_transform(X\_scaled)

plt.figure(figsize=(8, 6))

plt.scatter(X\_pca[:, 0], X\_pca[:, 1], c=y, cmap='coolwarm', s=100, edgecolor='k')

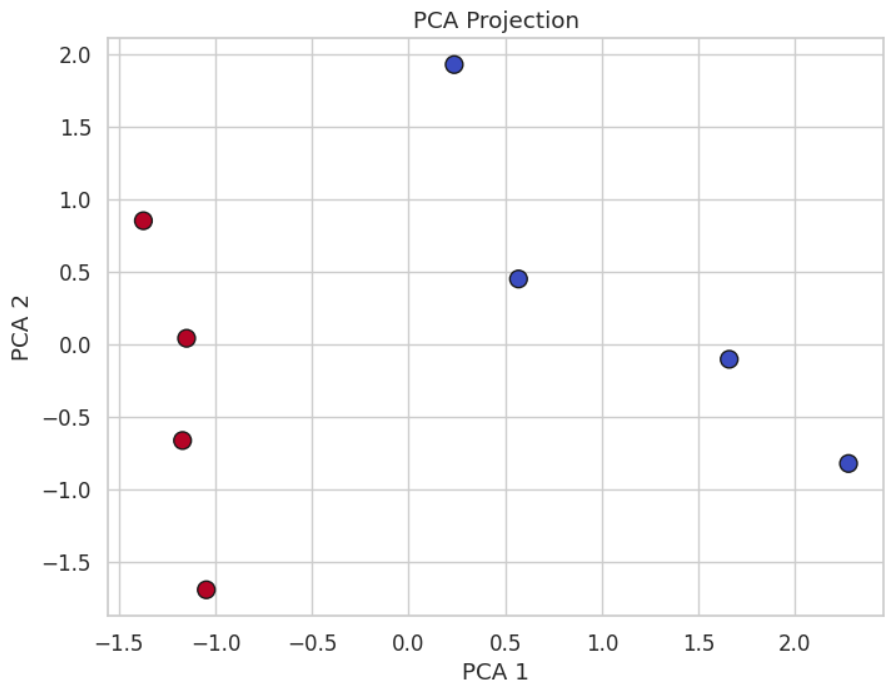
plt.title('PCA Projection')

plt.xlabel('PCA 1')

plt.ylabel('PCA 2')

plt.grid(True)

plt.show()



xx, yy = np.meshgrid(np.linspace(X\_scaled[:, 0].min(), X\_scaled[:, 0].max(), 100),

                     np.linspace(X\_scaled[:, 1].min(), X\_scaled[:, 1].max(), 100))

Z = model.predict(np.c\_[xx.ravel(), yy.ravel(), np.zeros\_like(xx.ravel())])

Z = Z.reshape(xx.shape)

plt.figure(figsize=(8, 6))

plt.contourf(xx, yy, Z, alpha=0.4, cmap='coolwarm')

plt.scatter(X\_scaled[:, 0], X\_scaled[:, 1], c=y, s=100, edgecolor='k', cmap='coolwarm')

plt.title('Decision Boundary (2D)')

plt.xlabel('MonthlyCharges (scaled)')

plt.ylabel('Tenure (scaled)')

plt.grid()

plt.show()

