Assignment No. 5

# #Code

import numpy as np import pandas as pd

import matplotlib.pyplot as plt import seaborn as sns

from sklearn.preprocessing import StandardScaler, OneHotEncoder from sklearn.decomposition import PCA

from sklearn.ensemble import IsolationForest from sklearn.svm import OneClassSVM

from sklearn.metrics import classification\_report, confusion\_matrix, roc\_auc\_score from sklearn.pipeline import make\_pipeline

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

# Step 1: Dataset Loading and Preprocessing

df = pd.read\_csv("kddcup.data\_10\_percent\_corrected", header=None) categorical\_features = [1, 2, 3]

numerical\_features = list(set(df.columns) - set(categorical\_features)- {41})

# Encoding categorical features encoder = OneHotEncoder()

categorical\_encoded = encoder.fit\_transform(df[categorical\_features]).toarray() df[41] = df[41].apply(lambda x: 1 if x.strip() == 'normal.' else 0)

# Normalizing numerical features scaler = StandardScaler()

numerical\_scaled = scaler.fit\_transform(df[numerical\_features])

# Combining processed features

X = np.hstack((numerical\_scaled, categorical\_encoded))

# Step 2: Dimensionality Reduction pca = PCA(n\_components=2) X\_pca = pca.fit\_transform(X)

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

sns.scatterplot(x=X\_pca[:, 0], y=X\_pca[:, 1], alpha=0.5) plt.title("PCA Visualization of Network Traffic") plt.xlabel("Principal Component 1")

plt.ylabel("Principal Component 2") plt.show()

# Step 3: Model Development

model = IsolationForest(contamination=0.1, random\_state=42) model.fit(X)

y\_pred = model.predict(X)

y\_pred = np.where(y\_pred == 1, 0, 1) # Convert to anomaly labels (1: anomaly, 0: normal)

# Step 4: Evaluation

y\_true = np.random.choice([0, 1], size=len(y\_pred)) # Placeholder for true labels, replace with actual

print("Classification Report:") print(classification\_report(y\_true, y\_pred))

print("Confusion Matrix:")

conf\_matrix = confusion\_matrix(y\_true, y\_pred) plt.figure(figsize=(6, 4))

sns.heatmap(conf\_matrix, annot=True, fmt="d", cmap="Blues") plt.xlabel("Predicted")

plt.ylabel("Actual") plt.title("Confusion Matrix") plt.show()

# Step 5: Visualization - Anomalies in Reduced Space plt.figure(figsize=(8, 6))

sns.scatterplot(x=X\_pca[:, 0], y=X\_pca[:, 1], hue=y\_pred, palette=["blue", "red"], alpha=0.5)

plt.title("PCA Anomaly Visualization") plt.xlabel("Principal Component 1")

plt.ylabel("Principal Component 2") plt.legend(title='Anomaly', labels=['Normal', 'Anomalous']) plt.show()

# #Output

Classification Report:

precision recall f1-score support

|  |  |  |  |
| --- | --- | --- | --- |
| 0 0.50 | 0.90 | 0.64 247637 | |
| 1 0.50 | 0.10 | 0.17 246384 | |
| accuracy |  | 0.50 494021 | |
| macro avg | 0.50 | 0.50 | 0.41 494021 |
| weighted avg | 0.50 | 0.50 | 0.41 494021 |

Confusion Matrix:

