Ex-8: Class Imbalance- SMOTE Technique

Question:

1. Download the Kaggle credit card dataset

2. Check if the target class has imbalance and display it.

3. Apply any one classifier on the data.

4. Apply SMOTE technique on the dataset.

5. Implement the same classifier after SMOTE.

6. a) Compare the classification accuracy and b) show the results using visualization using PCA and 2D Scatter plot

Code:

import pandas as pandas

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

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

from sklearn.tree import DecisionTreeClassifier

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

data = pandas.read\_csv(r'C:\Users\admin\Desktop\sem-5\AdvancedML\creditcard.csv')

X = data.drop('Class', axis=1)

y = data['Class']

data.head()

sns.countplot(x=y)

plt.title("")

plt.show()

print( y.value\_counts())

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

model =DecisionTreeClassifier(random\_state=42)

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

test\_preds = model.predict(X\_test)

test\_accuracy = accuracy\_score(y\_test, test\_preds)

print("Train Accuracy:", model.score(X\_train, y\_train))

print("Test Accuracy:", test\_accuracy)

print(classification\_report(y\_test, test\_preds))

print("\n",confusion\_matrix(y\_test, test\_preds))

from imblearn.over\_sampling import SMOTE

smote = SMOTE(random\_state=42)

X\_smote, y\_smote = smote.fit\_resample(X, y)

print("After SMOTE, class distribution:")

print(y\_smote.value\_counts())

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_smote, y\_smote, test\_size=0.2, random\_state=42)

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

test\_preds = model.predict(X\_test)

test\_accuracy = accuracy\_score(y\_test, test\_preds)

print("Train Accuracy:", model.score(X\_train, y\_train))

print("Test Accuracy:", test\_accuracy)

print(classification\_report(y\_test, test\_preds))

print("\n",confusion\_matrix(y\_test, test\_preds))

print("After SMOTE, class distribution:")

print(y\_smote.value\_counts())

sns.countplot(x=y\_smote)

plt.title("")

plt.show()

plt.figure(figsize=(14, 6))

plt.subplot(1, 2, 1)

plt.scatter(X\_pca\_before[:, 0], X\_pca\_before[:, 1], c=y, cmap='coolwarm', alpha=0.5)

plt.xlabel('PCA Component 1')

plt.ylabel('PCA Component 2')

plt.title('PCA 2D Scatter Plot Before SMOTE')

plt.subplot(1, 2, 2)

plt.scatter(X\_pca\_after[:, 0], X\_pca\_after[:, 1], c=y\_smote, cmap='coolwarm', alpha=0.5)

plt.xlabel('PCA Component 1')

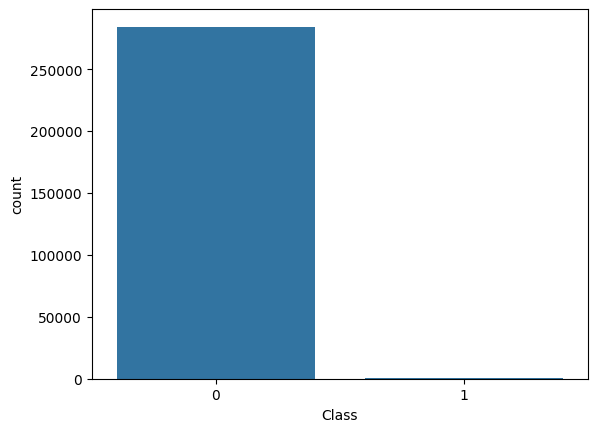
plt.ylabel('PCA Component 2')

plt.title('PCA 2D Scatter Plot After SMOTE')

plt.tight\_layout()

plt.show()

output:



Class

0 284315

1 492

Train Accuracy: 1.0

Test Accuracy: 0.9990519995786665

precision recall f1-score support

0 1.00 1.00 1.00 56864

1 0.70 0.80 0.74 98

accuracy 1.00 56962

macro avg 0.85 0.90 0.87 56962

weighted avg 1.00 1.00 1.00 56962

[[56830 34]

[ 20 78]]

After SMOTE, class distribution:

Class

0 284315

1 284315

Name: count, dtype: int64

Train Accuracy: 1.0

Test Accuracy: 0.9986282820111496

precision recall f1-score support

0 1.00 1.00 1.00 56750

1 1.00 1.00 1.00 56976

accuracy 1.00 113726

macro avg 1.00 1.00 1.00 113726

weighted avg 1.00 1.00 1.00 113726

