|  |  |  |
| --- | --- | --- |
| In | [1]: | import numpy as np import pandas as pd |
|  |  |  |
| In | [2]: | df = pd.read\_csv("D:\MIT ADT\Third Year - Sem 2\ML LAB\Assign 9 - PCA\Malwa reMemoryDump.csv") |
|  |  |  |
| In | [3]: | df.head() |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Out [ 3 ] : | **Raw\_Type** | **psIist\_nproc** | **psIist\_nppid** | **psIist\_avg\_threads** | **psIist\_nprocs64bit** | **psIist\_avg\_har** |
|  | 0 Benign | 45 | 17 | 10.555556 | 0 | 202.8. |
|  | 1 Benign | 47 | 19 | 11.531915 | 0 | 242.2: |
|  | 2 Benign | 40 | 14 | 14.725000 | 0 | 288.2: |
|  | 3 Benign | 32 | 13 | 13.500000 | 0 | 264.2‹ |
|  | 4 Benign | 42 | 16 | 11.452381 | 0 | 281.3: |

5 rows • 58 columns



In [4]:

*#d f[ Rao\_Type '] . un! que ) . sum( )*

|  |  |  |
| --- | --- | --- |
| In | [ 5 ] : | df = df.drop([”Raw\_Type"], axis=1) |
|  |  |  |
| In | [6]: | cat\_cols df.select\_dtypes(exclude=[“int64",'float64']).columns |
|  |  |  |
| In | [7]: | cat\_cols |
| Out | [ 7 ] : | Index(['SubType', 'Label'], dtype='object') |
| In | [ 8] : | from sklearn.preprocessing import LabelEncoder |
|  |  | lbl\_enc = LabelEncoder() |
|  |  | for i in cat\_cols:  df[i] = lbl\_enc.fit\_transform(df[i]) |
|  |  |  |
| In | [9]: | X = df.drop(['Label'], axis=1) |
|  |  | y = df["Label“] |

In [10]:

from sklearn.preprocessing import StandardScaler scaler = StandardScaler()

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

In [11]:

from sklearn.svm import SVC

from sklearn.model\_selection import train\_test\_split from sklearn.metrics import accuracy\_score

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

svm\_original = SVC(kernel='linear') svm\_original.fit(X\_train, y\_train) y\_pred\_original = svm\_original.predict(X\_test)

accuracy\_original accuracy\_score(y\_test, y\_pred\_original) print(”Accuracy Orignal: “, accuracy\_original)

Accuracy Orignal: 1.0

In [12]:

from sklearn.decomposition import PCA for i in range(1,6):

print("Components: ", i) pca = PCA(n\_components=i)

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

X\_train\_pca, X\_test\_pca, \_, \_ = train\_test\_split(X\_pca, y, test\_size=0.

2, random\_state=42)

svm\_pca = SVC(kernel='linear') svm\_pca.fit(X\_train\_pca, y\_train) y\_pred\_pca = svm\_pca.predict(X\_test\_pca)

accuracy\_pca accuracy\_score(y\_test, y\_pred\_pca) print("Accuracy PCA: ", i, "-->”, accuracy\_pca)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Components:  Accuracy PCA: | 1 | 1 | --> | 0.9652730375426621 |
| Components:  Accuracy PCA: | 2 | 2 | --> | 0.9841296928327645 |
| Components:  Accuracy PCA: | 3 | 3 | --> | 0.996160409556314 |
| Components:  Accuracy PCA: | 4 | 4 | --> | 0.9963310580204778 |
| Components:  Accuracy PCA: | 5 | 5 | --> | 0.997098976109215 |

In [13]:

import matplotlib.pyplot as pit

*# Get expLained van ance ratio*

explained\_variance\_ratio pca.explained\_variance\_ratio\_

*# PLot scree pLot*

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

plt.bar(range(1, len(explained\_variance\_ratio) + 1), explained\_variance\_rat io, alpha=0.5, align='center')

plt.xlabel('Principal Component') plt.ylabel('Proportion of Variance Explained') plt.title('Scree Plot')

plt.show()

