Day 10 – Dimensionality Reduction using PCA

📔 Today's Highlights

In this session, we focused on the concept of **Dimensionality Reduction**, and implemented **Principal Component Analysis (PCA)** on a vehicle dataset. Dimensionality reduction helps simplify large datasets by reducing the number of features while preserving essential patterns and variance.

What is Dimensionality Reduction?

 A technique used to **reduce the number of input features** in a dataset.

 It removes redundant or less-informative features while keeping important ones.

 Helps in improving **model performance**, **training speed**, and **visualization**.

🔍 What is PCA (Principal Component Analysis)?

 PCA is a **linear dimensionality reduction technique**.

 It transforms the data to a new coordinate system such that:

o The greatest variance lies on the **first principal component**.

o The second greatest on the second, and so on.

 PCA is **unsupervised** and is often used before classification or clustering.

🗀 Dataset Used: vehicle-2.csv

 This dataset contains 19 numeric features related to vehicle shape, aspect ratio, skewness, and other properties.

 The target column (class) is a categorical feature indicating vehicle type.

Steps Performed:

✅ 1. Importing Libraries & Loading Data

```
import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.decomposition import PCA

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import LabelEncoder, SimpleImputer
```

✅ **2. Data Cleaning & Encoding**

```
# Label encode the target class

le = LabelEncoder()

vehdf['class'] = le.fit_transform(vehdf['class'])

# Impute missing values with median

X = vehdf.iloc[:, 0:19]
```

imputer = SimpleImputer(strategy='median')

newdf = pd.DataFrame(imputer.fit_transform(X), columns=X.columns)

✅ 3. Outlier Detection & Visualization

� Used **boxplots**, **distribution plots**, and **IQR** technique to identify and filter outliers.

� Visualizations were created using seaborn.

✅ 4. Correlation Heatmap

sns.heatmap(newdf.corr(), annot=True, cmap='viridis', fmt=".2f")

plt.title("Feature Correlation Matrix")

plt.show()

✅ 5. PCA Application

pca = PCA(n_components=2)

principal_components = pca.fit_transform(newdf)

pca_df = pd.DataFrame(data=principal_components, columns=['PC1', 'PC2'])

pca_df['Class'] = le.inverse_transform(vehdf['class'])

✅ 6. PCA Visualization

sns.scatterplot(x='PC1', y='PC2', hue='Class', data=pca_df, palette='Set2')

plt.title("PCA Result: Vehicle Data in 2D")

plt.xlabel("Principal Component 1")

plt.ylabel("Principal Component 2")

plt.show()

📊 Summary & Observations

� PCA reduced the 19-dimensional data into just **2 components** while still separating vehicle classes clearly.

� **Visualization** showed that distinct clusters for different vehicle types could be formed using just PC1 and PC2.

� PCA helped improve **interpretability** and is useful for **feature selection** before classification.

📝 Conclusion

� Understood and implemented **dimensionality reduction** using PCA.

� Performed extensive **EDA** and **outlier removal**.

� Learned how PCA transforms high-dimensional data into lower dimensions while preserving structure.

� Visualized the results to understand class separability in reduced space.