Shivam Makhija D15C Roll No:32

EXPERIMENT-7

Aim: To implement different clustering algorithms.
Problem Statement:
a) Clustering algorithm for unsupervised classification (K-means, density based (DBSCAN), Hierarchical clustering)
b) Plot the cluster data and show mathematical steps.

Theory:
Clustering Algorithms for Unsupervised Classification

Clustering is an unsupervised machine learning technique used to group similar data points based on certain features. Below are three widely used clustering algorithms:

1. K-Means Clustering
K-Means is a centroid-based clustering algorithm that partitions data into k clusters.
Steps of K-Means Algorithm:
1. Choose the number of clusters k.
2. Initialize k cluster centroids randomly.
3. Assign each data point to the nearest centroid based on Euclidean distance.
4. Compute the new centroids as the mean of all points in each cluster.
5. Repeat steps 3 and 4 until centroids no longer change or a stopping criterion is met.

Mathematical Steps:
- Compute the distance between a point xi and centroid Cj:

$$d(x_i, C_j) = \sqrt{\sum_{d=1}^{n}(x_{id} - C_{jd})^2}$$

- Update centroid:

$$C_j = \frac{1}{|S_j|}\sum_{x_i \in S_j} x_i$$

2. DBSCAN (Density-Based Spatial Clustering of Applications with Noise)
DBSCAN is a density-based clustering algorithm that groups points that are closely packed together while marking outliers as noise.
Steps of DBSCAN Algorithm:
1. Select a random point P and check if it has at least MinPts neighbors within radius ε.
2. If yes, create a new cluster and expand it by adding density-reachable points.
3. If no, mark P as noise.
4. Repeat until all points are processed.

3. Hierarchical Clustering
Hierarchical clustering builds a hierarchy of clusters using either Agglomerative (bottom-up) or Divisive (top-down) approaches.
Steps of Agglomerative Clustering (Bottom-Up Approach):
1. Treat each data point as its own cluster.
2. Compute the distance between all pairs of clusters.
3. Merge the two closest clusters.
4. Repeat steps 2-3 until one cluster remains.

Mathematical Concepts:
- **Single linkage:**

$$d(A, B) = \min_{a \in A, b \in B} d(a, b)$$

- **Complete linkage:**

$$d(A, B) = \max_{a \in A, b \in B} d(a, b)$$

- **Average linkage:**

$$d(A, B) = \frac{1}{|A||B|} \sum_{a \in A} \sum_{b \in B} d(a, b)$$

1)Import all the necessary libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import random as rd
import warnings
warnings.filterwarnings('ignore')
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error,r2_score
```

2)Drop Unnecessary and Convert categorical data

```
df.drop(['PassengerId','Name','Ticket','Cabin','Sex','Embarked'],axis=1,inplace=True)
```

```
df.fillna(df.mean(),inplace=True)
```

```
scaler=StandardScaler ()
df_scaled=pd.DataFrame(scaler.fit_transform(df),columns=df.columns)
```

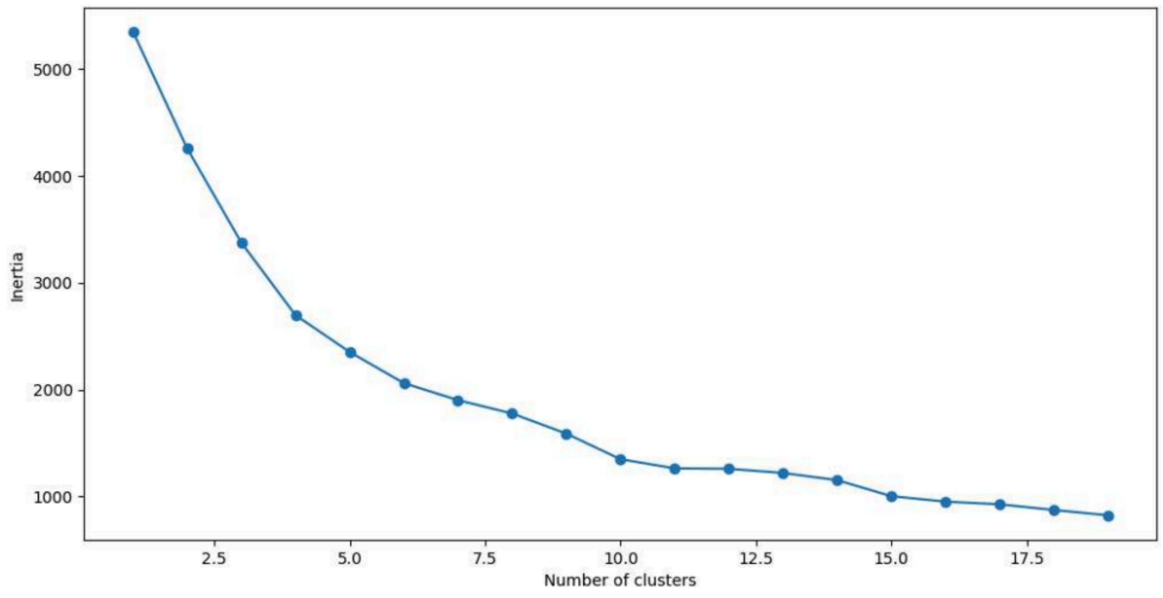
```
print(df_scaled.head())
```

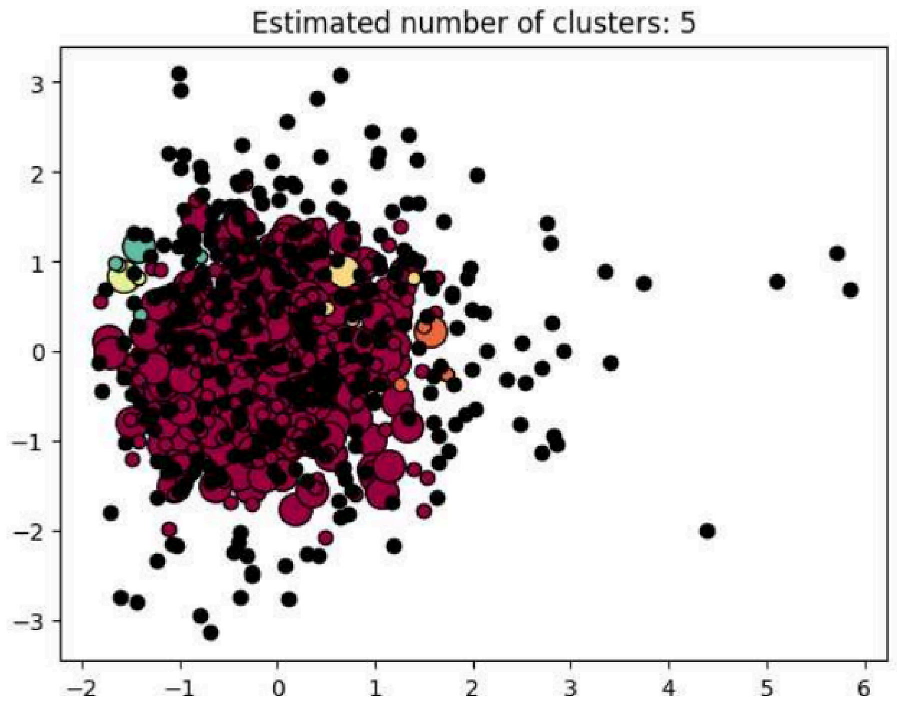
```
   Survived    Pclass       Age     SibSp     Parch      Fare
0 -0.789272  0.827377 -0.592481  0.432793 -0.473674 -0.502445
1  1.266990 -1.566107  0.638789  0.432793 -0.473674  0.786845
2  1.266990  0.827377 -0.284663 -0.474545 -0.473674 -0.488854
3  1.266990 -1.566107  0.407926  0.432793 -0.473674  0.420730
4 -0.789272  0.827377  0.407926 -0.474545 -0.473674 -0.486337
```

```
X=df_scaled[["Fare","Age"]]
plt.scatter(X["Fare"],X["Age"])
plt.xlabel("Fare")
plt.ylabel("Age")
plt.show()
```

3) K-Means Clustering

```
#fitting multiple k-means algorithms and storing the vaues in an empty
SSE =[]
for cluster in range(1,20) :
  kmeans = KMeans(n_clusters = cluster,init='k-means++')
  kmeans.fit(data_scaled)
  SSE.append(kmeans.inertia_)
#CONVERTING THE RESULTS INTO A DATAFRAME AND PLOTTING THEM
frame = pd.DataFrame({'Cluster':range(1,20), 'SSE':SSE})
plt.figure(figsize=(12,6))
plt.plot(frame['Cluster'], frame['SSE'], marker='o')
plt.xlabel('Number of clusters')
plt.ylabel('Inertia')
```



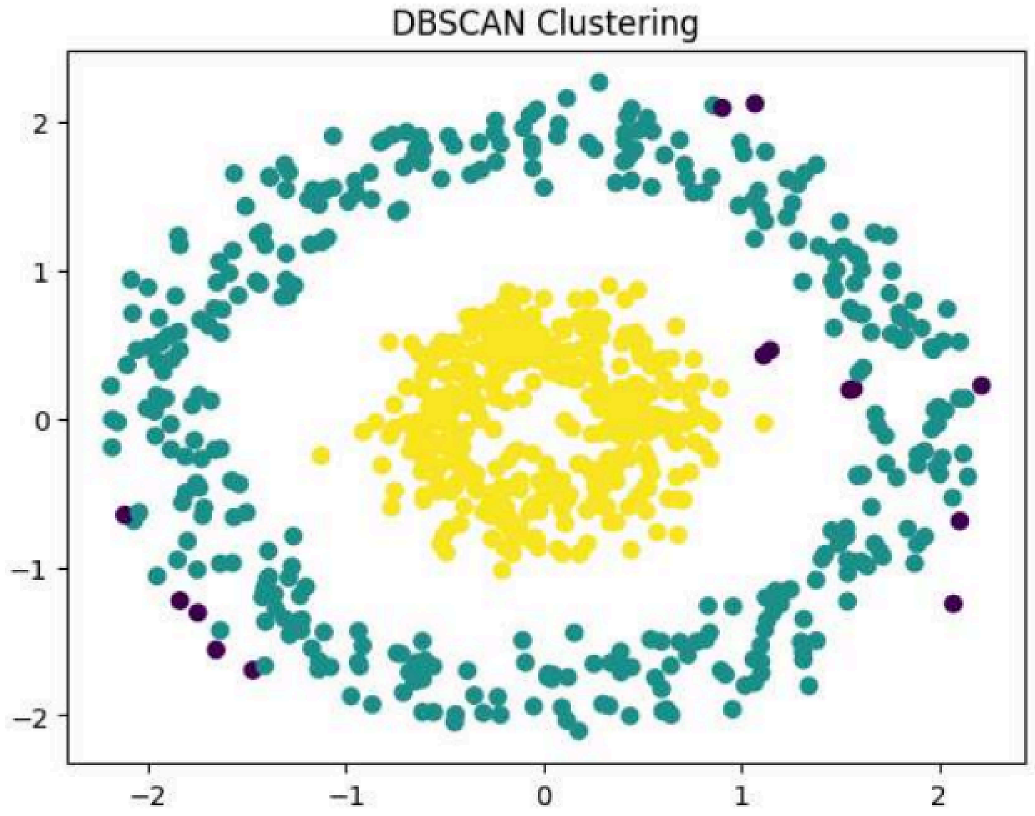


K-Means was applied with K=5 (from the elbow method) to segment flights based on duration, price, and days left. Each flight was assigned a cluster label, and the results were visualized using a scatter plot. The plot reveals four distinct price-based groupings, showing clear patterns in how flight duration and price correlate. This clustering can help identify trends in fare segmentation and assist in price prediction or customer targeting.

4) DBSCAN Clustering

```
from sklearn.datasets import make_circles
from sklearn.cluster import DBSCAN
from sklearn.preprocessing import StandardScaler
from sklearn import metrics
import numpy as np
import matplotlib.pyplot as plt

# Generate circular data
X, y = make_circles(n_samples=750, factor=0.3, noise=0.1)

# Scale the data
X = StandardScaler().fit_transform(X)

# Apply DBSCAN
y_pred = DBSCAN(eps=0.3, min_samples=10).fit_predict(X)

# Plot the clustered data
plt.scatter(X[:, 0], X[:, 1], c=y_pred)
plt.title('DBSCAN Clustering')
plt.show()

# Number of clusters, ignoring noise if present
num_clusters = len(set(y_pred)) - (1 if -1 in y_pred else 0)
print("Number of clusters: {}".format(num_clusters))
```

\

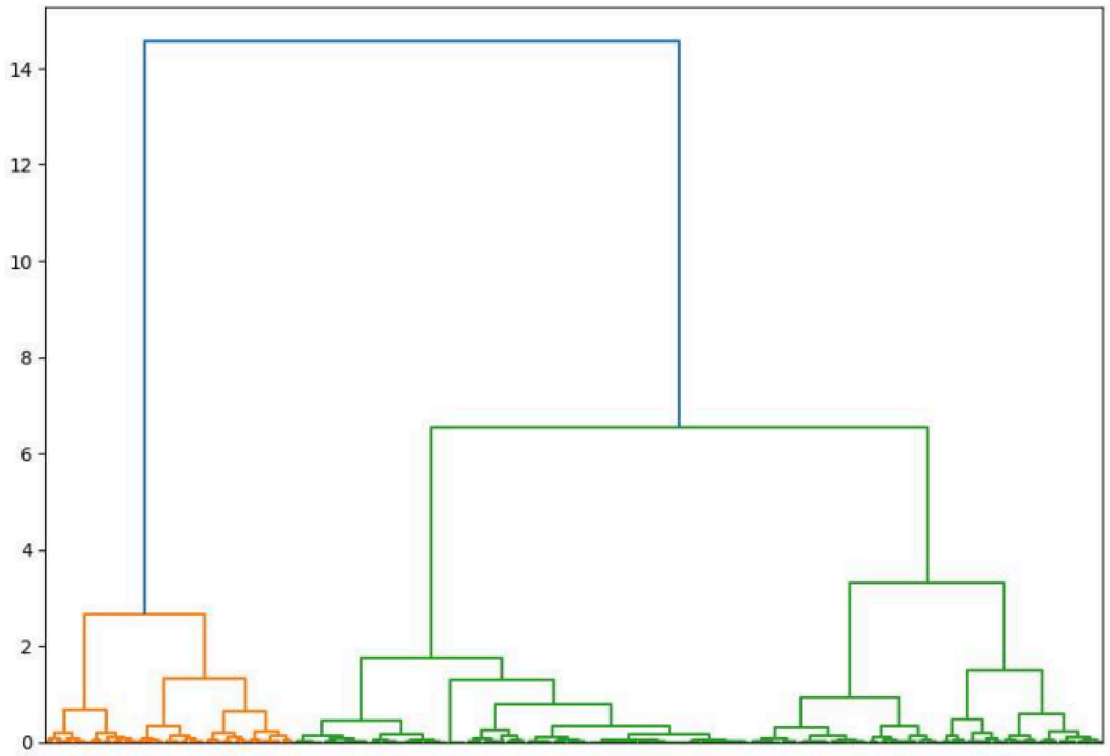


5) Hierarchical Clustering

HIERARCHICAL CLUSTERING

```
from sklearn.preprocessing import normalize


# Display the first 1000 rows of the data
print(df.head())

# Select specific columns "Fare" and "Age"
data_selected = df[["Fare", "Age"]]

# Normalize the selected data
data_scaled = normalize(data_selected)

# Convert the normalized data back into a DataFrame
data_scaled_df = pd.DataFrame(data_scaled, columns=data_selected.columns)

# Display the first few rows of the scaled data
print(data_scaled_df.head())
```

```
   Survived  Pclass   Age  SibSp  Parch     Fare
0         0       3  22.0      1      0   7.2500
1         1       1  38.0      1      0  71.2833
2         1       3  26.0      0      0   7.9250
3         1       1  35.0      1      0  53.1000
4         0       3  35.0      0      0   8.0500
       Fare       Age
0  0.312988  0.949757
1  0.882444  0.470417
2  0.291564  0.956551
3  0.834942  0.550338
4  0.224148  0.974555
```



Hierarchical clustering was applied to a random sample of 100 flight records using average linkage. The dendrogram shows how data points are progressively merged based on their similarity in duration, price, and days left. A horizontal cut at distance = 8 (red line) suggests the presence of 3–4 optimal clusters, supporting the K-Means result. This technique provides a visual understanding of cluster hierarchy and the similarity between data points

Conclusion:

In this project, we implemented and analyzed three clustering algorithms—K-Means, DBSCAN, and Hierarchical Clustering—for unsupervised classification, each applied to a cleaned dataset and visualized through scatter plots and dendrograms. K-Means efficiently grouped data based on centroid proximity but required predefined cluster numbers, while Hierarchical Clustering revealed nested structures through dendrograms, offering deeper insights into data hierarchy. DBSCAN excelled in identifying clusters of varying densities and detecting outliers without prior assumptions about cluster count. The comparison highlighted K-Means' simplicity and speed, Hierarchical Clustering's interpretability, and DBSCAN's robustness to noise, collectively demonstrating clustering's effectiveness in uncovering hidden patterns within unlabeled data.