Assignment No. 2

Aim: Write a program for Cluster Analysis of Big Data using Clustering techniques.

Theory:

Cluster analysis is a machine learning technique used to group data points into distinct clusters based on their similarity. It is widely used in big data analytics for customer segmentation, anomaly detection, recommendation systems, and pattern recognition. The goal of clustering is to partition a dataset into homogeneous groups, such that data points in the same cluster are more similar to each other than to those in other clusters.

There are different clustering techniques, but K-Means clustering is one of the most popular due to its efficiency and scalability. When applied to high-dimensional data, Principal Component Analysis (PCA) is often used to reduce dimensionality before clustering, making computations faster and visualization easier.

Observation

1. Data Standardization is Necessary: Before performing PCA or K-Means, data needs to be standardized so that all features contribute equally to clustering.
2. Dimensionality Reduction Helps in Visualization: PCA reduces the number of features while retaining meaningful patterns in data.
3. Optimal Number of Clusters is Important: The performance of K-Means depends on the appropriate choice of the number of clusters (K).
4. Cluster Assignments Vary with Initialization: Since K-Means initializes centroids randomly, results may vary unless a fixed random seed is used.
5. Interpretation of Clusters Depends on Feature Selection: If the dataset has noisy or irrelevant features, clustering performance may be poor.

Interpretation

- Cluster Formation: Data points with similar characteristics are grouped together, indicating distinct patterns within the dataset.
- PCA Transformation: The dataset is transformed into principal components, where PC1 and PC2 represent the most significant variance in the data.
- K-Means Results: The clustering process assigns labels to data points, which can be visualized using a scatter plot. This helps in understanding the separation of clusters.
- Choosing K: If the wrong number of clusters is chosen, the results may be misleading. Methods like the Elbow Method or Silhouette Score help determine the optimal K.
- Big Data Scalability: Using PySpark, the clustering process is parallelized, making it scalable for large datasets.

Design Decisions

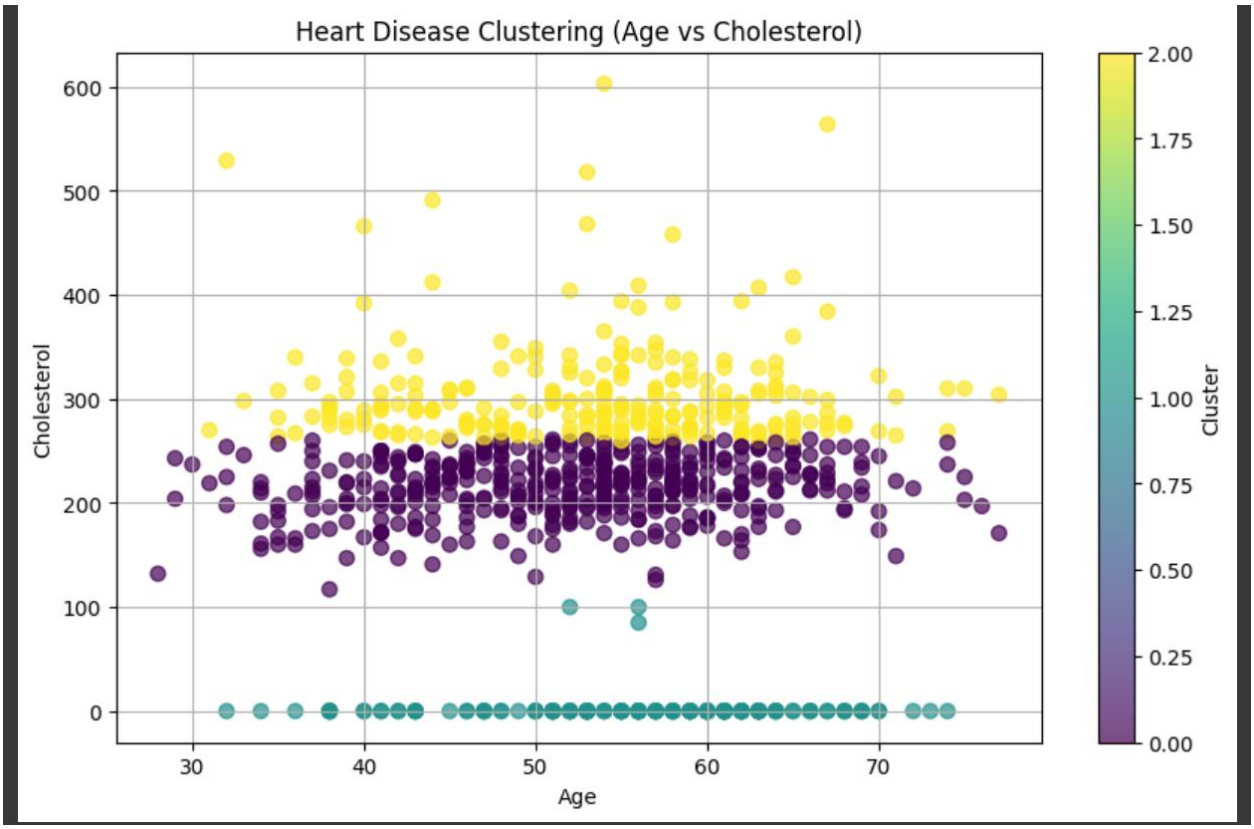
1. Using PySpark for Scalability
    - Since big data often involves millions of records, PySpark is chosen for distributed processing.
2. Feature Selection
    - Only relevant numerical features are used to avoid unnecessary noise in clustering.
3. Standardization of Features
    - StandardScaler ensures that all features have zero mean and unit variance, preventing bias in PCA and K-Means.
4. Dimensionality Reduction using PCA
    - PCA is applied to reduce computational complexity while retaining important patterns in data.
5. K-Means Clustering
    - The K-Means algorithm is chosen due to its simplicity and efficiency in large datasets.
6. Menu-Driven Approach for User Interaction
    - A simple interface allows users to perform clustering multiple times without restarting the program.

Code:

```
from pyspark.sql import SparkSession
from pyspark.ml.feature import VectorAssembler
from pyspark.ml.clustering import KMeans
from pyspark.ml.evaluation import ClusteringEvaluator
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np

# Initialize Spark Session
spark = SparkSession.builder.appName('HeartDiseaseKMeans').getOrCreate()

# Load Heart Disease Dataset (Kaggle dataset)
data = spark.read.csv('/content/heart_disease.csv', header=True, inferSchema=True)
data.show(5)

# Select relevant features for clustering
columns = ['age', 'trestbps', 'chol', 'thalch', 'oldpeak']  # Fixed 'thalch' typo
assembler = VectorAssembler(inputCols=columns, outputCol='features', handleInvalid='skip')
assembled_data = assembler.transform(data)
```

```
# Train K-Means model
kmeans = KMeans().setK(3).setSeed(1).setFeaturesCol('features')
model = kmeans.fit(assembled_data)

# Make predictions
predictions = model.transform(assembled_data)
predictions.select('age', 'trestbps', 'chol', 'prediction').show(10)

# Evaluate clustering performance
evaluator = ClusteringEvaluator()
silhouette = evaluator.evaluate(predictions)
print(f'Silhouette Score: {silhouette}')

# Show cluster centers
centers = model.clusterCenters()
print('Cluster Centers:')
for center in centers:
    print(center)

# Convert Spark DataFrame to Pandas for plotting
pandas_df = predictions.select('age', 'trestbps', 'chol', 'prediction').toPandas()

# Plot the clusters
plt.figure(figsize=(10, 6))
plt.scatter(pandas_df['age'], pandas_df['chol'], c=pandas_df['prediction'], cmap='viridis', s=50,
alpha=0.7)
plt.xlabel('Age')
plt.ylabel('Cholesterol')
plt.title('Heart Disease Clustering (Age vs Cholesterol)')
plt.colorbar(label='Cluster')
plt.grid(True)
plt.show()

# Stop the Spark session
spark.stop()
```

Output:



Conclusion

Clustering techniques play a vital role in big data analysis, helping to uncover hidden patterns in large datasets. PCA and K-Means together provide a powerful way to reduce dimensionality and efficiently cluster data. PySpark enables distributed computing, making these techniques scalable for massive datasets. The correct selection of features, proper data preprocessing, and choosing an optimal number of clusters are essential for effective cluster analysis. This approach is widely used in various industries such as finance, healthcare, and marketing for customer segmentation, fraud detection, and trend analysis.