| **Practical Assignment No. 11** | |
|---|---|
| Title: | **Apache Spark** |
| Problem Statement: | Demonstrate application of Apache spark to analyse streaming data from social media. |
| Objective: | To understand Apache Spark machine learning techniques for developing big data processing applications. |
| Outcome: | CO606.4: Utilize Apache Spark machine learning techniques for developing big data processing applications. |
| Software or Hardware Requirements: | Anaconda/Java/GCC |
| Theory: | **Clustering** is an **unsupervised machine learning** technique used to group similar data points into clusters such that data points in the same cluster are **more similar** to each other than to those in other clusters.<br><br>**K-Means** is one of the most popular and efficient clustering algorithms, especially suitable for large datasets.<br><br>In this experiment, we use **Apache Spark**, a distributed computing framework, to implement K-Means clustering efficiently on the **Seeds dataset**, which contains information about wheat kernels and their geometrical features.<br><br>**K-Means Clustering Algorithm**<br><br>**Definition:**<br><br>The **K-Means** algorithm partitions the dataset into **K clusters** such that each data point belongs to the cluster with the **nearest mean (centroid)**.<br><br>**Algorithm Steps:**<br><br>1. **Initialize:** Choose the number of clusters, **K**, and randomly select **K initial centroids**.<br><br>2. **Assign Step:** Assign each data point to the nearest centroid using the **Euclidean distance** formula: |

$$d(x_i, c_j) = \sqrt{\sum_{k=1}^{n} (x_{ik} - c_{jk})^2}$$

3. **Update Step:**

Recalculate centroids by taking the **mean** of all data points in each cluster:

$$c_j = \frac{1}{N_j} \sum_{x_i \in C_j} x_i$$

4. **Repeat:**
   Repeat steps 2 and 3 until centroids no longer change (convergence).

**Termination Condition:**

The algorithm stops when:

- Cluster assignments do not change between iterations, or
- The change in centroids is below a threshold.

## K-Means in Apache Spark

**Why Use Spark?**

Apache Spark provides a **distributed data processing framework**, allowing efficient computation of large datasets using parallel processing.
Spark's **MLlib** library includes scalable implementations of machine learning algorithms such as K-Means.

**Spark MLlib Implementation Workflow:**

1. Load the dataset into a Spark DataFrame or RDD.
2. Convert features into a feature vector using **VectorAssembler**.
3. Apply the **KMeans()** class from `pyspark.ml.clustering`.
4. Train the model and evaluate cluster centers.

5. Visualize or analyze clustering results.

## Implementation Steps

**Libraries Used:**

- `pyspark.sql` for Spark DataFrame operations
- `pyspark.ml.feature.VectorAssembler` for combining features
- `pyspark.ml.clustering.KMeans` for the clustering algorithm
- `pyspark.ml.evaluation.ClusteringEvaluator` for evaluating model performance

## Evaluation Metrics

**(a) Silhouette Score**

Measures how similar a data point is to its own cluster compared to other clusters.
Range: **[-1, 1]**, where a higher score indicates better-defined clusters.

**(b) Within-Cluster Sum of Squares (WCSS)**

Represents compactness of clusters; lower values indicate tighter clusters:

$$WCSS = \sum_{i=1}^{k} \sum_{x_j \in C_i} ||x_j - \mu_i||^2$$

## Choosing the Number of Clusters (K)

The optimal number of clusters is determined using the **Elbow Method**:

1. Run K-Means for different values of K (e.g., 2 to 10).
2. Plot the **WCSS** vs. **K**.
3. The point where the WCSS curve bends (elbow point) gives the best K.

## Advantages of K-Means

-

| | |
|---|---|
| | • Fast and efficient for large datasets.<br>• Works well when clusters are well-separated and spherical.<br>• Easily scalable with **Spark's distributed processing**.<br>**Limitations**<br>• Requires specifying K in advance.<br>• Sensitive to initial centroid positions.<br>• Not suitable for non-spherical or overlapping clusters.<br>• Performance affected by outliers.<br>**Applications**<br>• Image compression and segmentation<br>• Market segmentation and customer grouping<br>• Anomaly detection<br>• Document clustering<br>• Agricultural data analysis (e.g., seed or crop type classification) |
| Input/Datasets/Test Cases: | (Seeds Dataset (UCI Repository))<br>Dataset- Name of the Dataset:<br>Description of the Dataset:<br>Dataset Characteristics:<br>Subject Area:<br>Associated Tasks:<br>Feature Type:<br># Instances:<br># Features: |
| Results: | Execute code for analysing and streaming data from social media.<br>Take a print of this code with output for submission as part of results. |
| Analysis and conclusion: | Write your own analysis of output and conclusion( Minimum 1 statement of Analysis, Minimum 1 Statement Conclusion) |

| | |
|---|---|
| References: | Reference /Links(min Any 2, include dataset ref.). Write references in IEEE format. |