**Batch: H3-3 Roll No.: 16014022050**

**Experiment No.: 8**

|  |
| --- |
| **Title: To implement clustering using K-means algorithm** |

**AIM:** To understand the Clustering algorithm.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Expected Outcome of Experiment:**

CO4: Understand the basic concept and techniques of Machine Learning clustering.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Books/ Journals/ Websites referred:**

1. <https://uc-r.github.io/kmeans_clustering>
2. <https://en.wikipedia.org/wiki/K-means_clustering>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Pre Lab/ Prior Concepts:**

* Machine Learning
* Unsupervised Learning
* Distance Metrics
* K-means Algorithm

**Steps to perform Experiment:**

1. Understand K-means Algorithm
2. Select and Preprocess Data
3. Initialize Cluster Centers
4. Assign Data Points to Clusters
5. Update Cluster Centers
6. Repeat Assign and Update Steps
7. Choose Number of Clusters (k)
8. Evaluate Clustering Results
9. Interpret Results

**Description of the dataset used in implementation:**

**a. Title: Spending in mall**

**b. Source: Kaggle**

**c. Number of instances: 200**

**d. Number of attributes: 5**

**e. Attribute information: Customer Id, Gender, Annual Income, Spending Score**

**Rstudio code:**

dataset <- data.frame(mall)

X = dataset[4:5]

set.seed(6)

wcss = vector()

for (i in 1:10) wcss[i] = sum(kmeans(X, i)$withinss)

plot(x = 1:10,

y = wcss,

type = 'b',

main = paste('The Elbow Method'),

xlab = 'Number of clusters',

ylab = 'WCSS')

set.seed(29)

kmeans = kmeans(x = X,

centers = 5,

iter.max = 300,

nstart = 10)

library(cluster)

clusplot(x = X,

clus = kmeans$cluster,

lines = 0,

shade = TRUE,

color = TRUE,

labels = 2,

plotchar = FALSE,

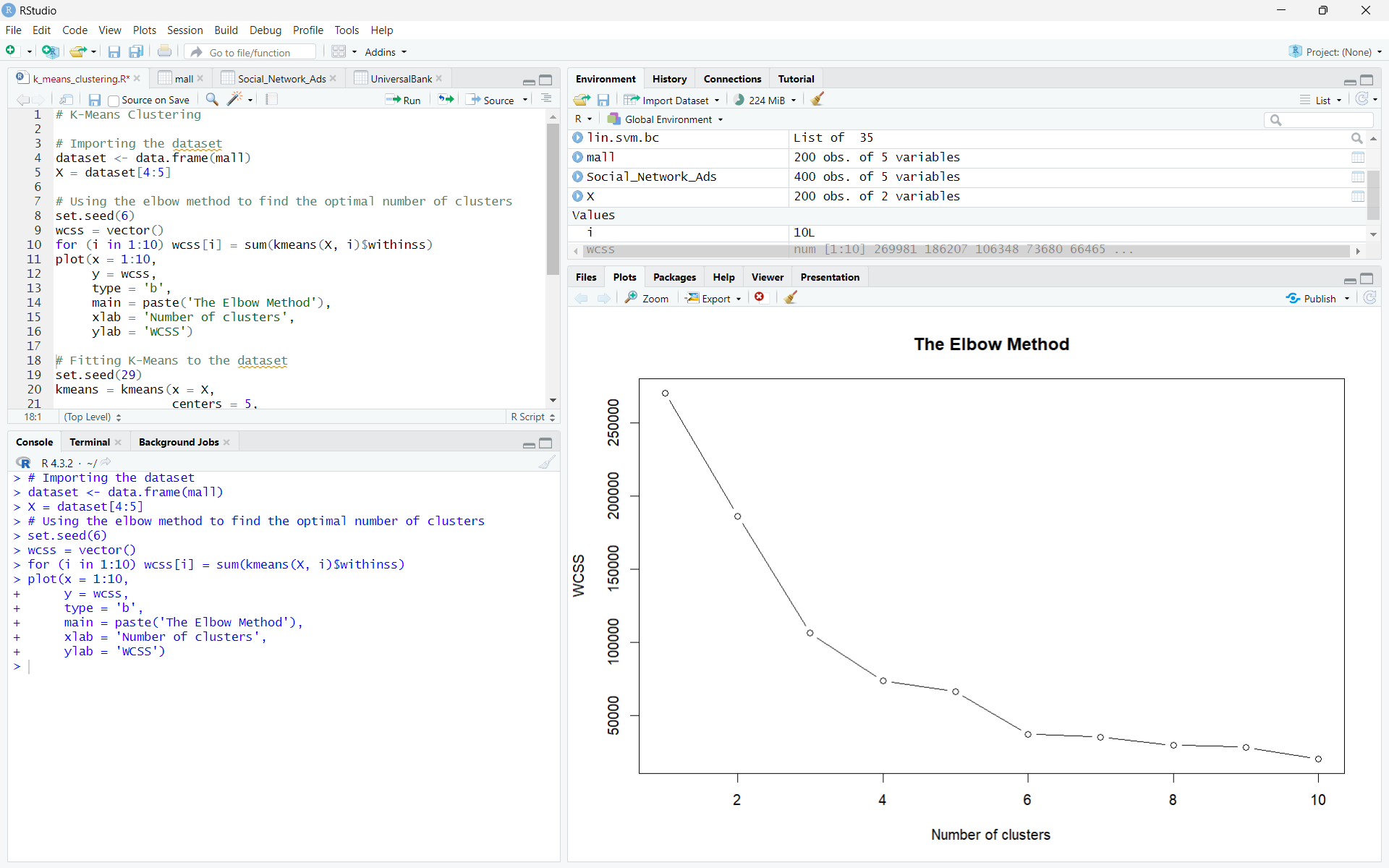
span = TRUE,

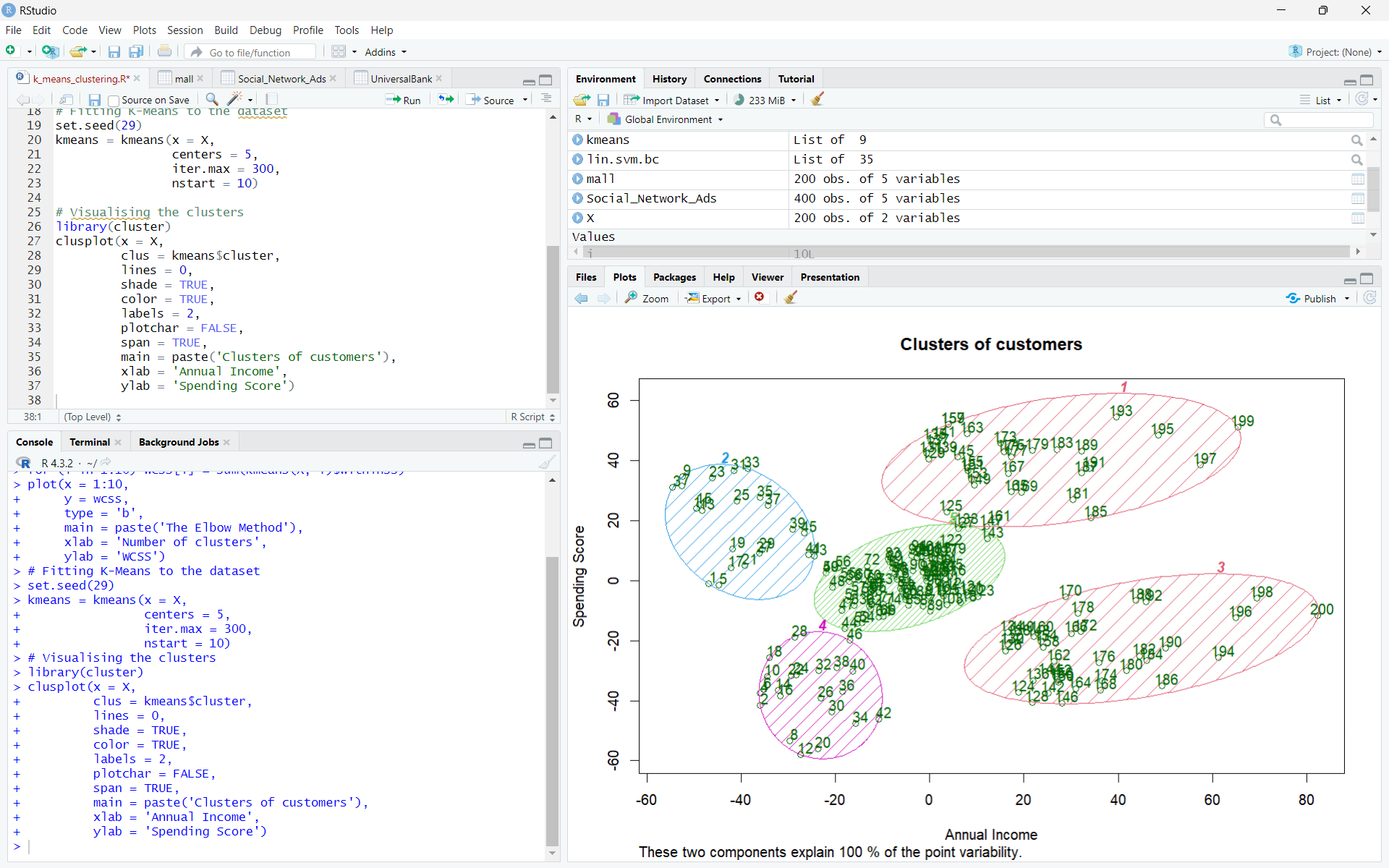
main = paste('Clusters of customers'),

xlab = 'Annual Income',

ylab = 'Spending Score')

**Output:**





**Conclusion:**

We successfully implemented clustering using K-means algorithm to understand the clustering algorithm.

**Post Lab Questions**

1. **Explain Apriori algorithm with suitable Numerical based example.**

The Apriori algorithm is a popular algorithm used in data mining for frequent itemset mining and association rule learning. It is based on the concept that a subset of a frequent itemset must also be frequent. Here's how it works:

1. Generate Candidate Itemsets:
   1. Begin by identifying all unique items in the dataset.
   2. Generate all possible combinations of these items to form candidate itemsets.
2. Scan Database:
   1. Count the occurrences of each candidate itemset in the database.
   2. Discard itemsets that do not meet the minimum support threshold (a user-defined parameter).
3. Generate New Candidate Itemsets:
   1. Based on the frequent itemsets found in the previous step, generate new candidate itemsets by joining them.
   2. Prune itemsets that contain subsets which are not frequent.
4. Repeat Steps 2 and 3:
   1. Repeat the process until no new frequent itemsets can be found.

Example:

Let's consider a transactional dataset where each transaction represents a set of items bought together:

Transaction ID Items

1 {bread, milk}

2 {bread, butter}

3 {bread, milk, butter}

4 {bread, milk}

5 {bread, butter}

Suppose we want to find frequent itemsets with a minimum support count of 2.

Generate Candidate Itemsets:

Unique items: {bread, milk, butter}

Candidate 1-itemsets: {bread}, {milk}, {butter}

Scan Database:

Count occurrences:

{bread}: 4

{milk}: 2

{butter}: 2

Discard {milk} and {butter} as they don't meet the minimum support.

Generate New Candidate Itemsets:

Generate candidate 2-itemsets by joining {bread} with itself: {bread, bread}

Prune {bread, bread} as it contains a subset that is not frequent.

Repeat:

Since no new frequent itemsets can be generated, stop.

Frequent Itemsets: {bread}

1. **Write a program to implement the Apriori algorithm.**

**from collections import defaultdict**

def generate\_candidates(itemsets, k):

candidates = set()

for itemset1 in itemsets:

for itemset2 in itemsets:

if len(itemset1.union(itemset2)) == k:

candidates.add(itemset1.union(itemset2))

return candidates

def apriori(dataset, min\_support):

# Step 1: Generate Candidate 1-itemsets

unique\_items = set()

for transaction in dataset:

unique\_items.update(transaction)

frequent\_itemsets = []

candidates = [{item} for item in unique\_items]

# Step 2, 3, 4: Scan Database, Generate New Candidate Itemsets, Repeat

k = 2

while candidates:

# Count occurrences

counts = defaultdict(int)

for transaction in dataset:

for candidate in candidates:

if candidate.issubset(transaction):

counts[candidate] += 1

# Prune based on support

frequent\_itemsets.extend([itemset for itemset, count in counts.items() if count >= min\_support])

# Generate new candidates

candidates = generate\_candidates(set(frequent\_itemsets), k)

k += 1

return frequent\_itemsets

# Example dataset

dataset = [

{"bread", "milk"},

{"bread", "butter"},

{"bread", "milk", "butter"},

{"bread", "milk"},

{"bread", "butter"}

]

min\_support = 2

result = apriori(dataset, min\_support)

print("Frequent Itemsets:", result)