International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 09 Issue: 01 | Jan 2022 www.irjet.net p-ISSN: 2395-0072

Machine Learning, K-means Algorithm Implementation with R

Mrs. Kavita Ganesh Kurale[1], Mrs. Rohini Sudhir Patil[2]

[1]Sr. Lecturer, Dept. of Computer Engineering, Sant Gajanan Maharaj Rural Polytechnic, Mahgaon, Maharashtra,India.
[2]HOD, Dept. of Computer Engineering, Sant Gajanan Maharaj Rural Polytechnic, Mahgaon, Maharashtra, India.

ABSTRACT: *Machine learning (ML) is that the growing technology and scientific study of algorithms that enables computers to find out automatically from previous knowledge. Machine learning uses numerous algorithms to create mathematical models and makes predictions exploitation previous knowledge available. Machine learning is artificial intelligent application. Machine learning either supervised or unsupervised learning. K-Means is most typically used algorithm that is unsupervised learning algorithmic program used for cluster analysis. During this paper we tend to worked with the implementation of K-Means and R Programming.*

Keywords: *Machine Learning, K-means, R Programming, supervised and unsupervised learning.*

I. INTRODUCTION

Machine learning is one of an application of artificial intelligence (AI) .In Machine learning the systems can automatically learn and improve the performance by using previously calculated results Machine learning focuses on implementation of new computer programs . It can access data and use this data then learn new things and calculate results.

The learning can be starts by observing knowledge, using direct experience on previously used knowledge, or instruction, and then patterns are calculated in that data and make decisions in the future based on the examples. The main aim is to allow the computers learn automatically without manual interference and compute results based on already computed results.

Some Machine Learning Methods

II. Classes of Machine Learning

1. Supervised Learning

2. Unsupervised Learning

Supervised machine learning algorithms supervised learning is a learning, we train the machine here some data is given which is provided with the correct answer. After that, the machine is provided with a new data set then machine again forced work on that newly data set so that supervised learning algorithm analyses the training data set of training examples and produces a correct outcome from sorted data. After sufficient training the system provides targets for any new input. The learning algorithm can also compare its output with the correct output and find errors in order to modify the model accordingly.

In opposite to the present, unsupervised machine learning algorithms are used when the knowledge used to train is not classified or labeled The system doesn't figure out the right output; however it explores the knowledge and might draw inferences from datasets to describe hidden structures from unlabeled data.

Semi-supervised machine learning algorithms fall nearly in between supervised and unsupervised learning, since they use both labeled and unlabeled data for training – generally a small quantum of labeled data and a large quantum of unlabeled data. The systems that use this approach are suitable to considerably improve learning accuracy. Generally, semi-supervised learning is chosen when the acquired labeled data requires good and relevant resources in order to train it/ learn from it. Else, acquiring unlabeled data generally does not require additional resources.

Reinforcement machine learning algorithms is a learning approach that interacts with its environment by producing actions and discovers breaches or rewards. Trial and error search and delayed reward are the most applicable characteristics of reinforcement learning. In order to maximize its performance this approach allows machines and software agents to automatically determine the ideal actions within a specific context. Simple price feedback is needed for the agent to learn which action is stylish; this is known as the underpinning signal. Machine learning performs analysis of huge quantities of data. While it generally delivers faster, more accurate results in order to identify profitable opportunities or dangerous risks, it should also require extra time and also resources to train it properly. Combination of machine learning and AI and cognitive technologies can turn it into effective in processing of large volumes of information.

III. Clustering

Clustering is the most popular approach in unsupervised learning where data is grouped based on the similarity of the data- points. Clustering has numerous real- life usages where it can be used in a variety of situations. Clustering is used in colorful fields like image recognition, pattern analysis, medical informatics, genomics, data compression etc. in machine

learning this is part of the unsupervised learning algorithm. This is because the data- points present aren't labeled and there's no explicit mapping of input and outputs.

IV. K- MEANs Clustering

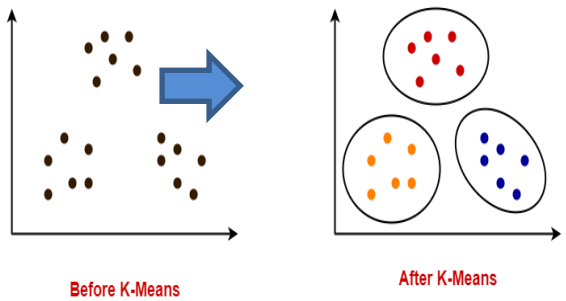
K-Means Clustering K- means is one of the simplest unsupervised learning algorithms that answer the well- known clustering problem. The procedure follows an easy and straightforward Method to classify a given data set through a particular number of clusters. The main thing here is for every cluster; define k centers, one for each. These centers should be placed during a cunning way due to different position causes different result. So, the better choice is to place them is important as possible far away from each other. Figure below K-Means Clustering. The next step is to take each data point from a given data set and assign it to the nearest center. When all data point completed, the primary step is completed and an early group age is completed. At this point we need tore-calculate k new centroids of the clusters resulting from the previous step. (1)

The KMeans algorithm is very simple [3]:

1. Select the value of Initial centroids i.e. K.
2. Repeat step no 3 and step no 4 for all data points in given dataset.
3. Find the closest data point from those centroids in the Dataset.
4. Form K cluster. Clusters are formed by assigning each point to its nearest centroid.
5. For each cluster in data set new global centroid are computed.

K-means algorithm Properties[3]:

1. Efficient while processing large data set.
2. It works only on number values.
3. The clusters shape is convex.



Objective of the K-means

The objective of the K-means clustering is to minimize the Euclidean distance that each point has from the centroid of the cluster.

$$Dist_{XY} = \sqrt{\sum_{k=1}^{m} (X_{ik} - X_{jk})^2}$$

Euclidean distance Formula

V. Implementation of K-Means with R-Programming

Step 1: Generation of Data

Here some random data is generated. Two vectors are defined vector1 and vector2 and create a 2-D array named data points which defines data points i.e. (x,y) coordinate pairs.

```
> vector1 <- c(1, 1.5, 2, 2.5,3, 3.5, 4,4.5)

> vector2 <- c(1, 2, 3, 4,5,6,7,8)

> datapoints<-array(c(vector1,vector2), dim = c(8,2))

> print(datapoints)
```

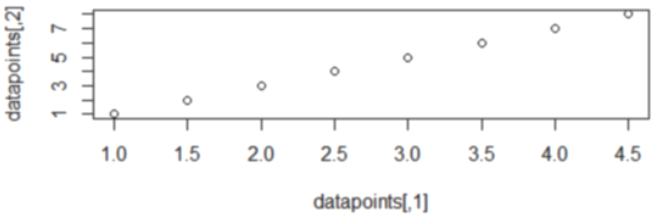
The data Points defined here is a 2-D array. The first column indicated the X coordinates, and the second column represent Y-coordinates.

It is defined as shown below:

```
[,1] [,2]
[1,] 1.0  1
[2,] 1.5  2
[3,] 2.0  3
[4,] 2.5  4
[5,] 3.0  5
[6,] 3.5  6
[7,] 4.0  7
[8,] 4.5  8
```

Now in following diagram we plotted the data points and visualize them using the plot function in R programming. The output is shown as below:

```
>plot(datapoints)
```



Step 2: Initiate Random Centroids for k-Clusters

We will initialize 2 clusters with 2 centroids (1.5, 2) and (3,5).

> k=2

> c1=c(1.5,2)

> c2=c(3,5)

> centroid=array(c(1.5,2,3,5), dim= c(k,2))

> print(centroid)

We define the k =2 number of clusters. An array of two co-ordinate pairs is the centroids. the two clusters is shown below is the array centroid containing the coordinates :

[,1] [,2]

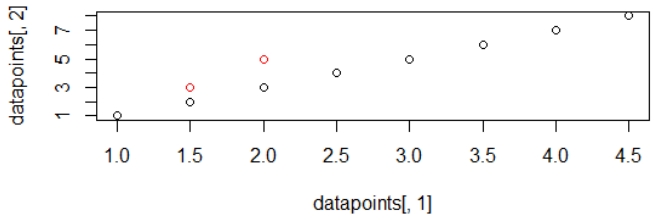
[1,] 1.5 3

[2,] 2.0 5

Using the plot function , We will plot the data points and the initial centroids on the same plot. We use the points function to specify the centroids,. The points function is used to highlight points of interest using different colors. Centroids are represented using the color red.

> plot(datapoints[,1], datapoints[,2])

>points(centroid[,1], centroid[,2], col="red")



Step 3: From each point Distance Calculation

Distance between the centroid and the remaining points are calculated using Euclidean distance formula. The Euclidean distance is defined as follows:

$$Dist_{XY} = \sqrt{\sum_{k=1}^{m} (X_{ik} - X_{jk})^2}$$

We will use the above equation above in the following sub-section. Here we are calculated the Euclidean distance formula in three steps.

Calculate the distance between the corresponding X and Y coordinates of the data-points and the centroid.

Calculate the sum of the square of the differences computed in Step 1.

Find the square root of the sum of squares of differences which is calculated in Step 2.

Difference: datapointi–centroid

dist_frm_clst1<-(datapoints[,]- centroid[1,])^2

> dist_frm_clst1=sqrt(dist_frm_clst1[,1]+ dist_frm_clst1[,2])

> dist_frm_clst1

[1] 0.7071068 1.8027756 1.5811388 1.1180340 3.8078866 3.0413813 6.0415230 [8] 5.2201533

Square of difference: (datapointi–centroid)2

>dist_frm_clst2=(datapoints[,]-centroid[2,])^2

Addition and Square root:

>dist_frm_clst2=sqrt(dist_frm_clst2[,1]+ dist_frm_clst2[,2])

> dist_frm_clst2

[1] 1.414214 4.609772 1.000000 2.692582 3.162278 1.802776 5.385165 3.041381

Here the dist_frm_clst1 is the distance which is between each point and the centroid-1. Likewise , we calculate the distances for centroid-2.

tot_dist=array(c(dist_frm_clst1,dist_frm_clst2), dim = c(8,2))

> tot_dist

[,1] [,2]

[1,] 0.7071068 1.414214

[2,] 1.8027756 4.609772

[3,] 1.5811388 1.000000

[4,] 1.1180340 2.692582

[5,] 3.8078866 3.162278

[6,] 3.0413813 1.802776

[7,] 6.0415230 5.385165

[8,] 5.2201533 3.041381

Step 4: Compare, finalize the Closest Centroids

Let's create a logical comparing vector dist_frm_clst_1 and dis_frm_clst2. This vector will be made up of the Boolean values TRUE and FALSE. For example create this vector using a conditional statement.

We write the condition as follows: distance to the first cluster is less than the second cluster's distance. Points here that satisfy given condition belong to cluster 1. The remaining points are belongs to cluster 2.

c(tot_dist[,1]<= tot_dist[,2])

[1] TRUE TRUE FALSE TRUE FALSE FALSE FALSE FALSE

Using the logical vector above, we obtain the elements of the first cluster. The operation used below is an example of conditional selection. Elements that satisfy this condition in the array dataPoints are printed.

datapoints[,1][c(tot_dist[,1] <= tot_dist[,2])]

[1] 1.0 1.5 2.5

To find the centroid of the newly formed cluster, we take the mean of all the points obtained above. The thinking is as follows: We need to find a point closest to all the cluster data points. Therefore, averaging the data points results in a point closest to the remaining points.

>mean(datapoints[,1][c(tot_dist[,1] <= tot_dist[,2])])

We calculate the mean using the R function mean. This is an example of how we select elements conditionally that belongs to a cluster and how we find its centroid.

[1] 1.666667

c1 = c(mean(datapoints[,1][c(tot_dist[,1] <= tot_dist[,2])]),mean(datapoints[,2][c(tot_dist[,1] <= tot_dist[,2])]))

We compute the X and Y coordinates of the centroid using the code above. We store the X coordinate in c1 and y-coordinates in c2. We copy the data in these lists to a new array called new_centroid.

> new_centroid[1,] = c1

> new_centroid[2,] = c2

The new_centroid contains the updated centroid of the formed clusters. Therefore, we have implemented the algorithm successfully.

> new_centroid

[,1] [,2]

[1,] 1.666667 2.333333

[2,] 3.400000 5.800000

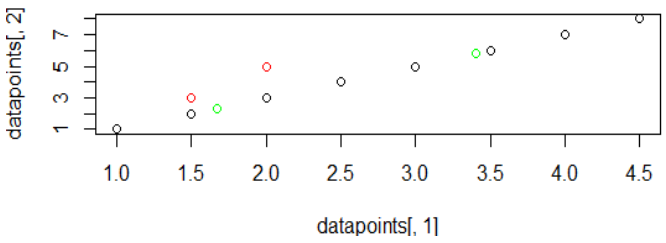
Let's plot the new centroids using the following code:

plot(datapoints[,1], datapoints[,2])

> points(centroid[,1],centroid[,2],col="red")

>points(new_centroid[,1],new_centroid[,2],col="green")

The old and updated centroids are shown in the figure below.



VI. CONCLUSION

Kmeans clustering is one of the most popular and widely used clustering algorithms, usually the apply when solving clustering tasks to get an idea of the structure of the dataset. The main aim of kmeans algorithm is to group data points into distinct non-overlapping subgroups such that single group contain same type of data item. Here we implemented Kmeans algorithm using r-programming and computed new global centroid for clusters successfully. Data is generated using vector in r and Euclidean distance formula is used for distance calculation. We calculated distance using mean function in r and new centroid plotted on graph. Hence we followed all K means algorithm steps for centroid computation

REFERENCES

[1] International Journal of Pure and Applied Mathematics Volume 117 No. 7 2017, 157-164 ISSN: 1311-8080 (printed version); ISSN: 1314-3395 (on-line version) url: http://www.ijpam.eu Special Issue "A k-means Clustering Algorithm on Numeric Data"

[2] International Journal of Information & Computation Technology. ISSN 0974-2239 Volume 4, Number 17 (2014),

pp. 1847-1860 © International Research Publications House http://www. Irphouse.com A Review ON K-means DATA Clustering APPROACH

[3] 2017 6th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions), Sep. 20-22, 2017, AIIT, Amity University Uttar Pradesh, Noida, India "A Detailed Study of Clustering Algorithms"

[4] https://data-flair.training/blogs/using-r-for-data-science/