VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELGAUM - 590014

FDS Activity -1

Introduction to R programming language

Submitted By :

Shilpa BK	1SI19CS108
Shwetha Jogi	1SI19CS115
Samyaktha GA	1SI19CS102

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SIDDAGANGA INSTITUTE OF TECHNOLOGY, TUMKUR-572103

(An Autonomous Institute, Affiliated to Visvesvaraya Technological University, Belgaum, Approved by AICTE and Accredited by NBA, New Delhi)

2021-202

ACTIVITY 1

INTRODUCTION

What is R?

R is a Programming language for statistical computing and graphics supported by the R Core Team and the R Foundation for Statistical Computing. Created by statisticians Ross Ihka and Robert Gentleman, R is used among data miners and statisticians for data analysis and developing Statistical Software. Users have created packages to augment the functions of the R language.

Why Use R?

- R is not just a statistics package, it's a language.
- R is designed to operate the way that problems are thought about.
- R is both flexible and powerful.
- R has superb graphical capabilities that are far better than any other statistical language.
- R is a platform-independent, which means it can be used across all operating systems.

These R programming topics have been made use of in the following program:

Syntax for Writing Functions in R

```
func_name <- function (argument) {


statement


}
```

- Here, we can see that the reserved word function is used to declare a function in R.
- The statements within the curly braces form the body of the function. These braces are optional if the body contains only a single expression.
- Finally, this function object is given a name by assigning it to a variable, func_name.

R if statement

The syntax of if statement is:

```
if (test_expression) {

statement

}
```

If the test_expression is TRUE, the statement gets executed. But if it’s FALSE, nothing happens.Here, test_expression can be a logical or numeric vector, but only the first element is taken into consideration.

In the case of numeric vector, zero is taken as FALSE, rest as TRUE.

Flowchart of if statement

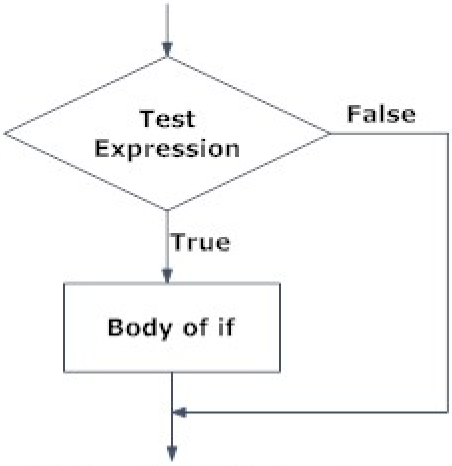


Fig: Operation of if statement

if…else statement

The syntax of if…else statement is:

```
if (test_expression) {

statement1

} else {

statement2   }
```

The else part is optional and is only evaluated if test_expression is FALSE.

It is important to note that else must be in the same line as the closing braces of the if statement.

Flowchart of if…else statement

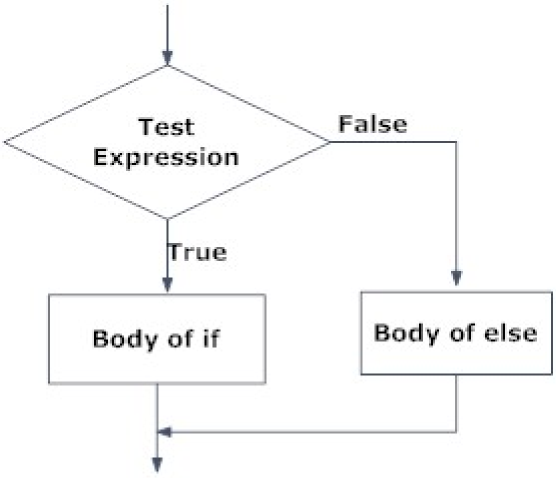


Fig: Operation of if...else statement

Syntax of for loop:

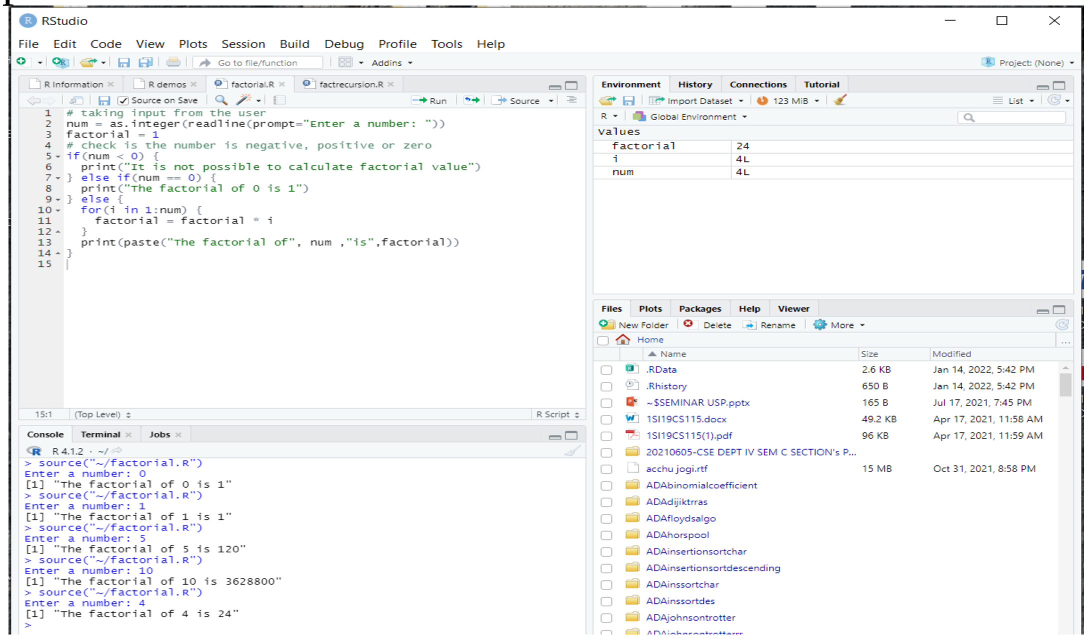
```
for (val in sequence)


{


Statement


}
```

Here, sequence is a vector and val takes on each of its value during the loop. In each iteration, statement is evaluated

R program to find the factorial of a number

```
# taking input from the user

num = as.integer(readline(prompt="Enter a number: "))

factorial = 1

# check is the number is negative, positive or zero

if(num < 0) {

print("It is not possible to calculate factorial value")

} else if(num == 0) {

print("The factorial of 0 is 1")

} else {

for(i in 1:num) {

factorial = factorial * i

}

print(paste("The factorial of", num ,"is",factorial))

}
```

Output:



```
# taking input from the user
num = as.integer(readline(prompt="Enter a number: "))
factorial = 1
# check is the number is negative, positive or zero
if(num < 0) {
  print("It is not possible to calculate factorial value")
} else if(num == 0) {
  print("The factorial of 0 is 1")
} else {
  for(i in 1:num) {
    factorial = factorial * i
  }
  print(paste("The factorial of", num ,"is",factorial))
}
```

```
> source("~/factorial.R")
Enter a number: 0
[1] "The factorial of 0 is 1"
> source("~/factorial.R")
Enter a number: 1
[1] "The factorial of 1 is 1"
> source("~/factorial.R")
Enter a number: 5
[1] "The factorial of 5 is 120"
> source("~/factorial.R")
Enter a number: 10
[1] "The factorial of 10 is 3628800"
> source("~/factorial.R")
Enter a number: 4
[1] "The factorial of 4 is 24"
>
```

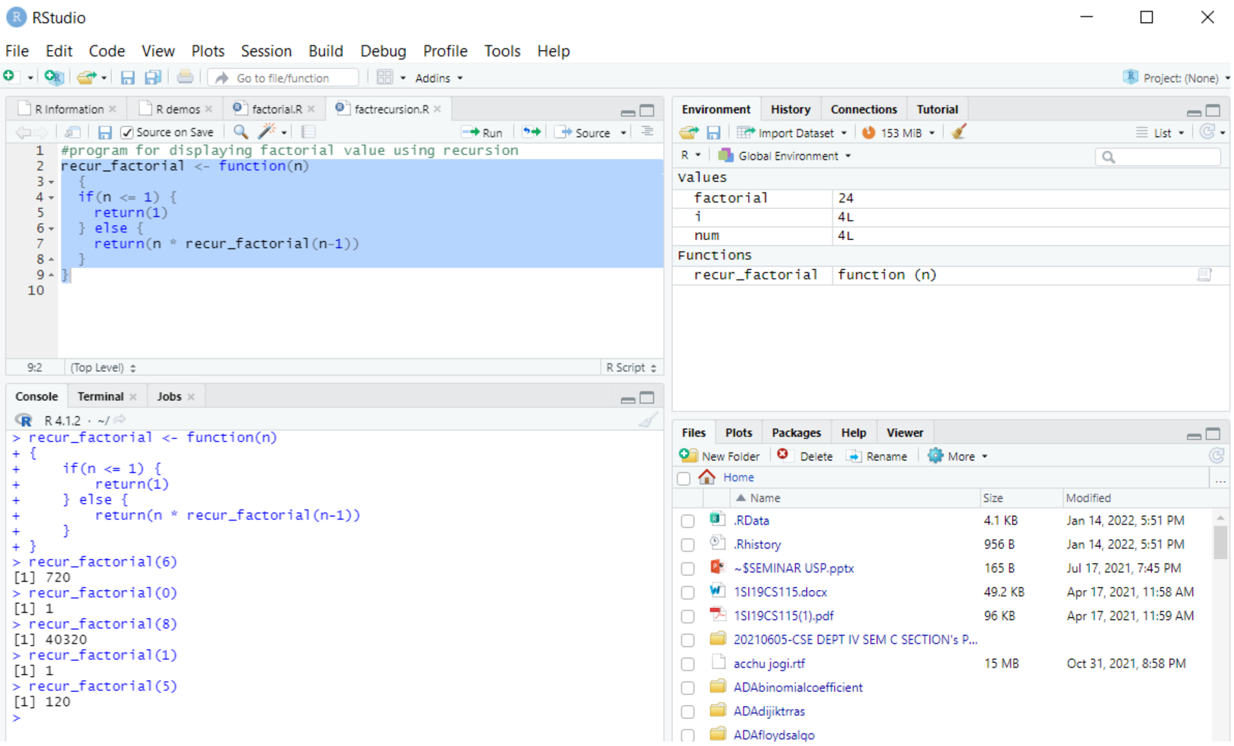
Values	
factorial	24
i	4L
num	4L

Name	Size	Modified
.RData	2.6 KB	Jan 14, 2022, 5:42 PM
.Rhistory	650 B	Jan 14, 2022, 5:42 PM
~$SEMINAR USP.pptx	165 B	Jul 17, 2021, 7:45 PM
1SI19CS115.docx	49.2 KB	Apr 17, 2021, 11:58 AM
1SI19CS115(1).pdf	96 KB	Apr 17, 2021, 11:59 AM
20210605-CSE DEPT IV SEM C SECTION's P...		
acchu jogi.rtf	15 MB	Oct 31, 2021, 8:58 PM
ADAbinomialcoefficient		
ADAdijiktrras		
ADAfloydsalgo		
ADAhorspool		
ADAinsertionsortchar		
ADAinsertionsortdescending		
ADAinssortchar		
ADAinssortdes		
ADAjohnsontrotter		

R Program to Find the Factorial of a Number Using Recursion

```
recur_factorial <- function(n) {

if(n <= 1) {

return(1)

} else {

return(n * recur_factorial(n-1))

}

}
```

Output



```
#program for displaying factorial value using recursion
recur_factorial <- function(n)
  {
  if(n <= 1) {
    return(1)
  } else {
    return(n * recur_factorial(n-1))
  }
}
```

```
> recur_factorial <- function(n)
+ {
+     if(n <= 1) {
+         return(1)
+     } else {
+         return(n * recur_factorial(n-1))
+     }
+ }
> recur_factorial(6)
[1] 720
> recur_factorial(0)
[1] 1
> recur_factorial(8)
[1] 40320
> recur_factorial(1)
[1] 1
> recur_factorial(5)
[1] 120
>
```

Values

factorial	24
i	4L
num	4L

Functions

recur_factorial	function (n)

Name	Size	Modified
.RData	4.1 KB	Jan 14, 2022, 5:51 PM
.Rhistory	956 B	Jan 14, 2022, 5:51 PM
~$SEMINAR USP.pptx	165 B	Jul 17, 2021, 7:45 PM
1SI19CS115.docx	49.2 KB	Apr 17, 2021, 11:58 AM
1SI19CS115(1).pdf	96 KB	Apr 17, 2021, 11:59 AM
20210605-CSE DEPT IV SEM C SECTION's P...		
acchu jogi.rtf	15 MB	Oct 31, 2021, 8:58 PM
ADAbinomialcoefficient		
ADAdijiktrras		
ADAfloydsalgo		

ACTIVITY 2

Crop yield prediction using K-means Clustering Algorithm

INTRODUCTION

Crop yield prediction is an **important agricultural problem**. Each and Every farmer is always tries to know, how much yield will get from his expectation. In the past, yield prediction was calculated by analyzing farmer's previous experience on a particular crop. The science of training dataset to learn and produce models for future predictions is widely used. Agriculture plays a critical role in the global economy. With the continuing expansion of the human population understanding worldwide crop yield is central to addressing food security challenges and reducing the impacts of climate change.
The Agricultural yield is primarily depends on weather conditions, pests and planning of harvest operation. Accurate information about history of crop yield is an important thing for making decisions related to agricultural risk management. This research focuses on evolution of a prediction model which may be used to predict crop yield production. The proposed method use K-means clustering algorithm to predict the crop yield production based on the Dataset.

PROBLEM STATEMENT

Crop yield prediction model using K-means clustering intends to commit to the improvement of the agricultural sector. It proposes to improve the crop yield and decrease the costs required for growing the crop yields because the smallholder farmers will know what exactly they need to produce in particular season so that there will be effective high crop yields.

RStudio

RStudio is an integrated development environment (IDE) for R [15]. R is free and open source. It is a language that is not difficult to use and relatively intuitive. Another reason for using R is because there are people from all over the world writing packages; these packages contain functions and data sets that you can install and use for free. Moreover, R has an extraordinary data visualization as well as graphics abilities.Before the implementation, I had to call some libraries and install some packages to run the K-mean clustering. These packages are:

- library("readxl"): this package is used to read Excel files.
- library(cluster): this package is used to implement the clustering algorithms.
- library(factoextra): this package is used to extract and visualize the data.

K-Means Clustering Algorithm

K-Means clustering algorithm is defined as an unsupervised learning method having an iterative process in which the dataset are grouped into k number of predefined non-overlapping clusters or subgroups, making the inner points of the cluster as similar as possible while trying to keep the clusters at distinct space it allocates the data points to a cluster so that the sum of the squared distance between the clusters centroid and the data point is at a minimum, at this position the centroid of the cluster is the arithmetic mean of the data points that are in the clusters.

This algorithm is an iterative algorithm that partitions the dataset according to their features into K number of predefined non- overlapping distinct clusters or subgroups. It makes the data points of inter clusters as similar as possible and also tries to keep the clusters as far as possible. It allocates the data points to a cluster if the sum of the squared distance between the cluster's centroid and the data points is at a minimum, where the cluster's centroid is the arithmetic mean of the data points that are in the cluster. A less variation in the cluster results in similar or homogeneous data points within the cluster.

In unsupervised learning techniques that is a popular clustering technique. We just start by choosing a random number of K of data points from our sample; these represent the initial centroids, the cluster centers, and their numbers equal the number of clusters. Then, we allocate each sample to the closest cluster center. This step can be done by calculating the Euclidean distance between all the random cluster centers and any other data point. Finally, this can be generated until there is no re-assignment of the samples to the cluster centroids.

Euclidean Distance between two points in space:

If **p** = (p_1, p_2) and **q** = (q_1, q_2) then the distance is given by

$$d(\mathbf{p}, \mathbf{q}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2}.$$

Euclidean distance

We then assign each data point to the cluster center closest to it.

Assigning each point to the nearest cluster:

If each cluster centroid is denoted by c_i, then each data point x is assigned to a cluster based on

$$\arg \min_{c_i \in C} dist(c_i, x)^2$$

here *dist()* is the euclidean distance

Finding the new centroid from the clustered group of points:

$$c_i = \frac{1}{|S_i|} \sum_{x_i \in S_i} x_i$$

S_i is the set of all points assigned to the *ith* cluster.

For new centroid

Since we started with choosing random of K clusters of data points, it will not give us great and efficient results. So, we repeat the process and instead of using random initial centroids from our data points, we will calculate the actual cluster centers using the following techniques to get the optimal number of clusters centers.

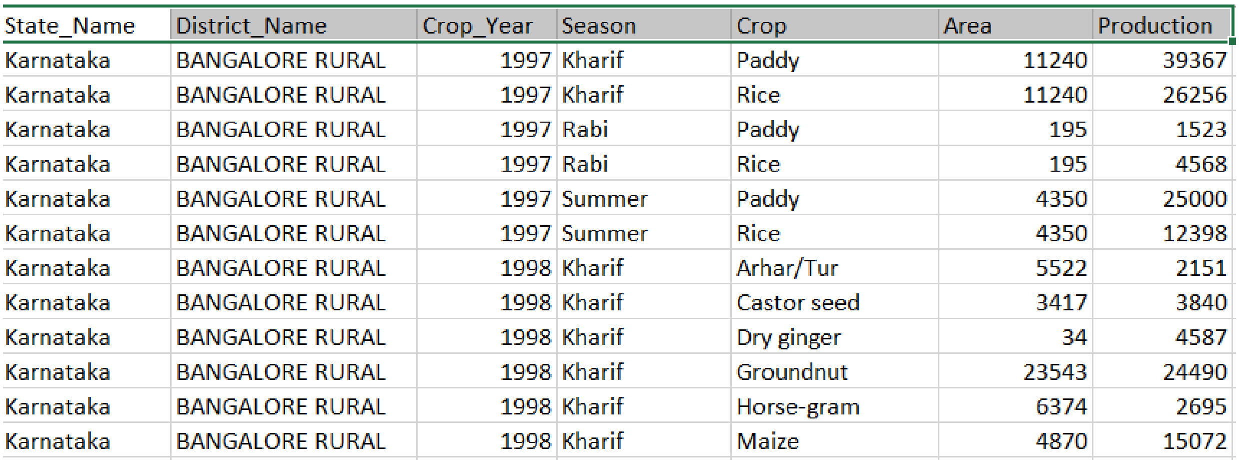
- **Elbow Method**

This method provides us with an image on what a good k number of clusters would be based on the sum of squared distance (SSE) between data points and their assigned clusters' centroids.

Dataset Description

In any ML analysis, you need data. And any model can only be powerful if you feed it with the right data. The on-target data should have the precise features and the right outcomes because it will affect the relevance and the usability of the model as well as the findings. The data applied for my work was obtained from Kaggle, the world's largest data science community with powerful tools and resources. It was made for crop yield prediction.

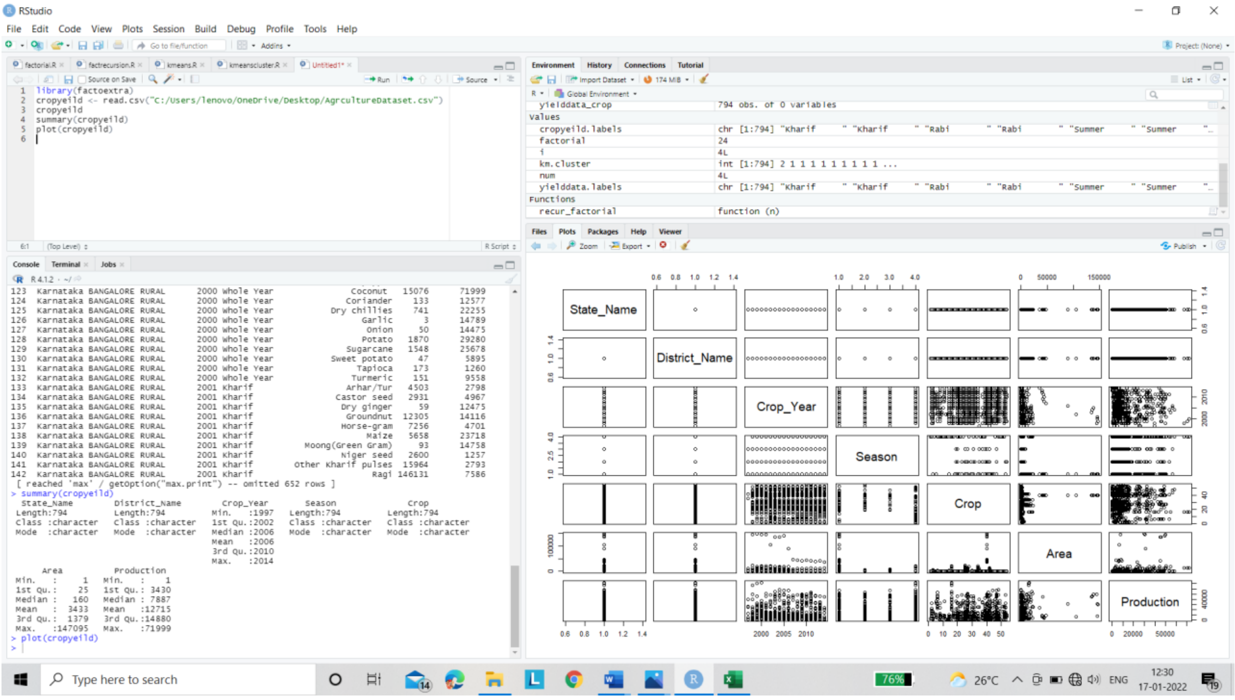
The Dataset consisting of State name, district name, crop year, season, crop, area and production of Bangalore urban district of Karnataka. There are nearly 800 observations in the dataset.



State_Name	District_Name	Crop_Year	Season	Crop	Area	Production
Karnataka	BANGALORE RURAL	1997	Kharif	Paddy	11240	39367
Karnataka	BANGALORE RURAL	1997	Kharif	Rice	11240	26256
Karnataka	BANGALORE RURAL	1997	Rabi	Paddy	195	1523
Karnataka	BANGALORE RURAL	1997	Rabi	Rice	195	4568
Karnataka	BANGALORE RURAL	1997	Summer	Paddy	4350	25000
Karnataka	BANGALORE RURAL	1997	Summer	Rice	4350	12398
Karnataka	BANGALORE RURAL	1998	Kharif	Arhar/Tur	5522	2151
Karnataka	BANGALORE RURAL	1998	Kharif	Castor seed	3417	3840
Karnataka	BANGALORE RURAL	1998	Kharif	Dry ginger	34	4587
Karnataka	BANGALORE RURAL	1998	Kharif	Groundnut	23543	24490
Karnataka	BANGALORE RURAL	1998	Kharif	Horse-gram	6374	2695
Karnataka	BANGALORE RURAL	1998	Kharif	Maize	4870	15072

Crop yield prediction Dataset

EXECUTION AND OUTPUT:



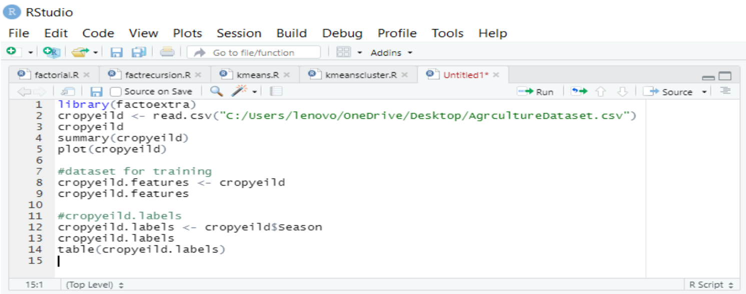
```
1 library(factoextra)
2 cropyeild <- read.csv("C:/Users/lenovo/OneDrive/Desktop/AgrcultureDataset.csv")
3 cropyeild
4 summary(cropyeild)
5 plot(cropyeild)
6
```

```
yielddata_crop        794 obs. of 0 variables
Values
  cropyeild.labels    chr [1:794] "Kharif     " "Kharif     " "Rabi       " "Rabi       " "Summer     " "Summer     "...
  factorial           24
  i                   4L
  km.cluster          int [1:794] 2 1 1 1 1 1 1 1 1 1 ...
  num                 4L
  yielddata.labels    chr [1:794] "Kharif     " "Kharif     " "Rabi       " "Rabi       " "Summer     " "Summer     "...
Functions
  recur_factorial     function (n)
```

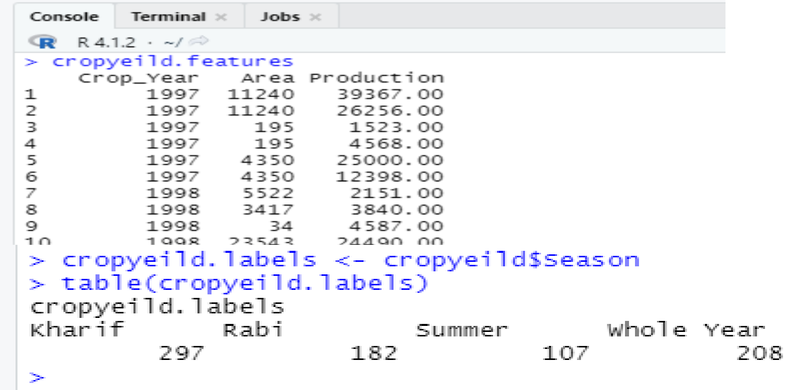
```
123  Karnataka BANGALORE RURAL      2000 Whole Year                Coconut  15076       71999
124  Karnataka BANGALORE RURAL      2000 Whole Year              Coriander    133       12577
125  Karnataka BANGALORE RURAL      2000 Whole Year           Dry chillies    741       22255
126  Karnataka BANGALORE RURAL      2000 Whole Year                 Garlic      3       14789
127  Karnataka BANGALORE RURAL      2000 Whole Year                  Onion     50       14475
128  Karnataka BANGALORE RURAL      2000 Whole Year                 Potato   1870       29280
129  Karnataka BANGALORE RURAL      2000 Whole Year              Sugarcane   1548       25678
130  Karnataka BANGALORE RURAL      2000 Whole Year           Sweet potato     47        5895
131  Karnataka BANGALORE RURAL      2000 Whole Year                Tapioca    173        1260
132  Karnataka BANGALORE RURAL      2000 Whole Year               Turmeric    151        9558
133  Karnataka BANGALORE RURAL      2001 Kharif                  Arhar/Tur   4503        2798
134  Karnataka BANGALORE RURAL      2001 Kharif                Castor seed   2931        4967
135  Karnataka BANGALORE RURAL      2001 Kharif                 Dry ginger     59       12475
136  Karnataka BANGALORE RURAL      2001 Kharif                  Groundnut  12305       14116
137  Karnataka BANGALORE RURAL      2001 Kharif                 Horse-gram   7256        4701
138  Karnataka BANGALORE RURAL      2001 Kharif                      Maize   5658       23718
139  Karnataka BANGALORE RURAL      2001 Kharif          Moong(Green Gram)     93       14758
140  Karnataka BANGALORE RURAL      2001 Kharif                 Niger seed   2600        1257
141  Karnataka BANGALORE RURAL      2001 Kharif        Other Kharif pulses  15964        2793
142  Karnataka BANGALORE RURAL      2001 Kharif                       Ragi 146131        7586
 [ reached 'max' / getOption("max.print") -- omitted 652 rows ]
> summary(cropyeild)
  State_Name         District_Name         Crop_Year        Season              Crop          
 Length:794         Length:794         Min.   :1997   Length:794         Length:794        
 Class :character   Class :character   1st Qu.:2002   Class :character   Class :character  
 Mode  :character   Mode  :character   Median :2006   Mode  :character   Mode  :character  
                                       Mean   :2006                                        
                                       3rd Qu.:2010                                        
                                       Max.   :2014                                        
      Area            Production   
 Min.   :     1   Min.   :    1  
 1st Qu.:    25   1st Qu.: 3430  
 Median :   160   Median : 7887  
 Mean   :  3433   Mean   :12715  
 3rd Qu.:  1379   3rd Qu.:14880  
 Max.   :147095   Max.   :71999  
> plot(cropyeild)
> 
```

Fig 1: Libraries are installed and Data set is imported for analysis and plotting, Summary of the dataset and also plot of dataset is shown in console.

For training the model we use only numeric dataset, So by exrtracting numeric values we use the dataset where it consist of crop_Year, Area and Production, For the further analysis we use class label as season and compare the clustering method, K-means with our given data Crop_year, Area and Poduction and see if they essential successfully clusters the corresponding data with its corresponding season and predicts the crop yield.



```
library(factoextra)
cropyeild <- read.csv("C:/Users/lenovo/OneDrive/Desktop/AgrcultureDataset.csv")
cropyeild
summary(cropyeild)
plot(cropyeild)

#dataset for training
cropyeild.features <- cropyeild
cropyeild.features

#cropyeild.labels
cropyeild.labels <- cropyeild$Season
cropyeild.labels
table(cropyeild.labels)

```



```
> cropyeild.features
   Crop_Year   Area Production
1       1997  11240   39367.00
2       1997  11240   26256.00
3       1997    195    1523.00
4       1997    195    4568.00
5       1997   4350   25000.00
6       1997   4350   12398.00
7       1998   5522    2151.00
8       1998   3417    3840.00
9       1998     34    4587.00
10      1998  23543   24490.00
> cropyeild.labels <- cropyeild$Season
> table(cropyeild.labels)
cropyeild.labels
Kharif          Rabi            Summer          Whole Year
          297             182             107              208
> 
```

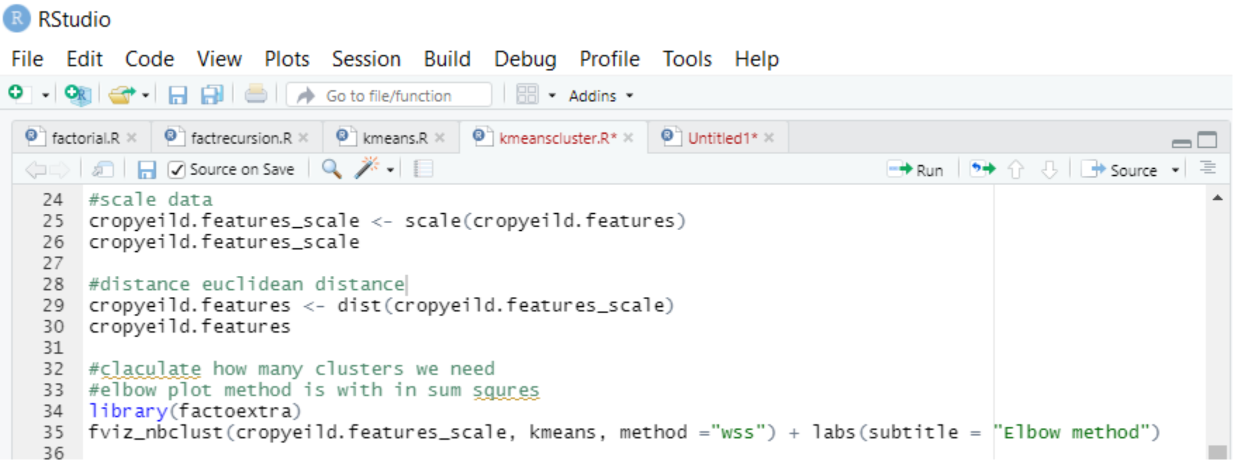
Before applying k-means

STEP 1: we scale our dataset, it is important that data should be balanced.

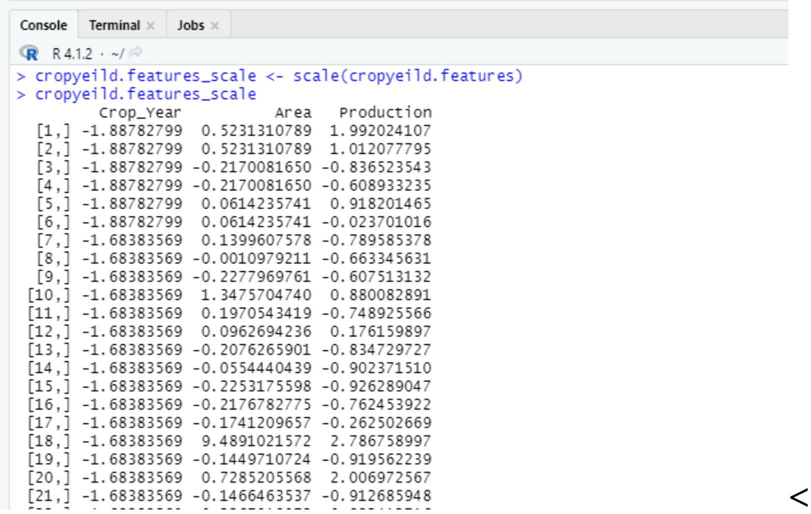
STEP 2: Calculate the distance matrix between our observations (EUCLIDEAN DISTANCE)

STEP 3: Calculate how many clusters we need because using this algorithm we should have fixed input & how many clusters we need in order to calculate centroid

The number of clusters can be find out by within sum squares method and Elbow method.

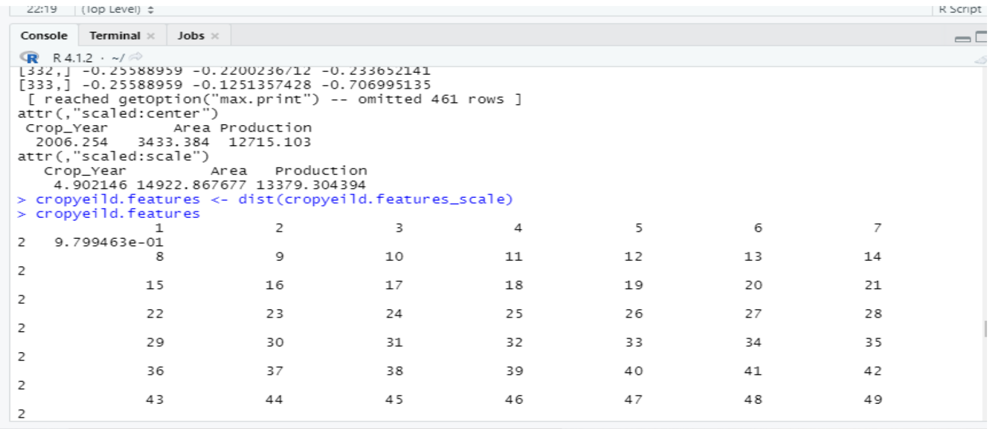


```
#scale data
cropyeild.features_scale <- scale(cropyeild.features)
cropyeild.features_scale

#distance euclidean distance
cropyeild.features <- dist(cropyeild.features_scale)
cropyeild.features

#claculate how many clusters we need
#elbow plot method is with in sum squres
library(factoextra)
fviz_nbclust(cropyeild.features_scale, kmeans, method ="wss") + labs(subtitle = "Elbow method")

```



```
> cropyeild.features_scale <- scale(cropyeild.features)
> cropyeild.features_scale
        Crop_Year          Area   Production
  [1,] -1.88782799  0.5231310789  1.992024107
  [2,] -1.88782799  0.5231310789  1.012077795
  [3,] -1.88782799 -0.2170081650 -0.836523543
  [4,] -1.88782799 -0.2170081650 -0.608933235
  [5,] -1.88782799  0.0614235741  0.918201465
  [6,] -1.88782799  0.0614235741 -0.023701016
  [7,] -1.68383569  0.1399607578 -0.789585378
  [8,] -1.68383569 -0.0010979211 -0.663345631
  [9,] -1.68383569 -0.2277969761 -0.607513132
 [10,] -1.68383569  1.3475704740  0.880082891
 [11,] -1.68383569  0.1970543419 -0.748925566
 [12,] -1.68383569  0.0962694236  0.176159897
 [13,] -1.68383569 -0.2076265901 -0.834729727
 [14,] -1.68383569 -0.0554440439 -0.902371510
 [15,] -1.68383569 -0.2253175598 -0.926289047
 [16,] -1.68383569 -0.2176782775 -0.762453922
 [17,] -1.68383569 -0.1741209657 -0.262502669
 [18,] -1.68383569  9.4891021572  2.786758997
 [19,] -1.68383569 -0.1449710724 -0.919562239
 [20,] -1.68383569  0.7285205568  2.006972567
 [21,] -1.68383569 -0.1466463537 -0.912685948
```

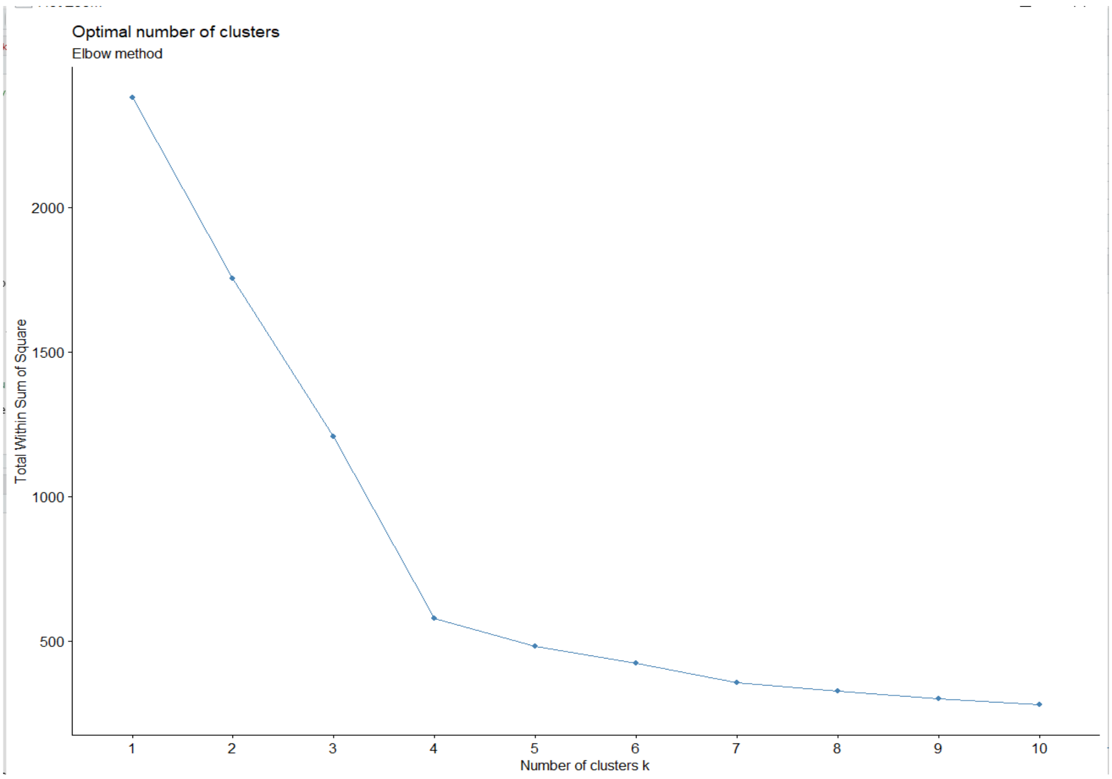
<= scaled data



```
[332,] -0.25588959 -0.2200236712 -0.233652141
[333,] -0.25588959 -0.1251357428 -0.706995135
 [ reached getOption("max.print") -- omitted 461 rows ]
attr(,"scaled:center")
 Crop_Year        Area Production
  2006.254   3433.384  12715.103
attr(,"scaled:scale")
   Crop_Year          Area   Production
    4.902146 14922.867677 13379.304394
> cropyeild.features <- dist(cropyeild.features_scale)
> cropyeild.features
              1            2            3            4            5            6            7
2  9.799463e-01
              8            9           10           11           12           13           14
2
             15           16           17           18           19           20           21
2
             22           23           24           25           26           27           28
2
             29           30           31           32           33           34           35
2
             36           37           38           39           40           41           42
2
             43           44           45           46           47           48           49
2
```

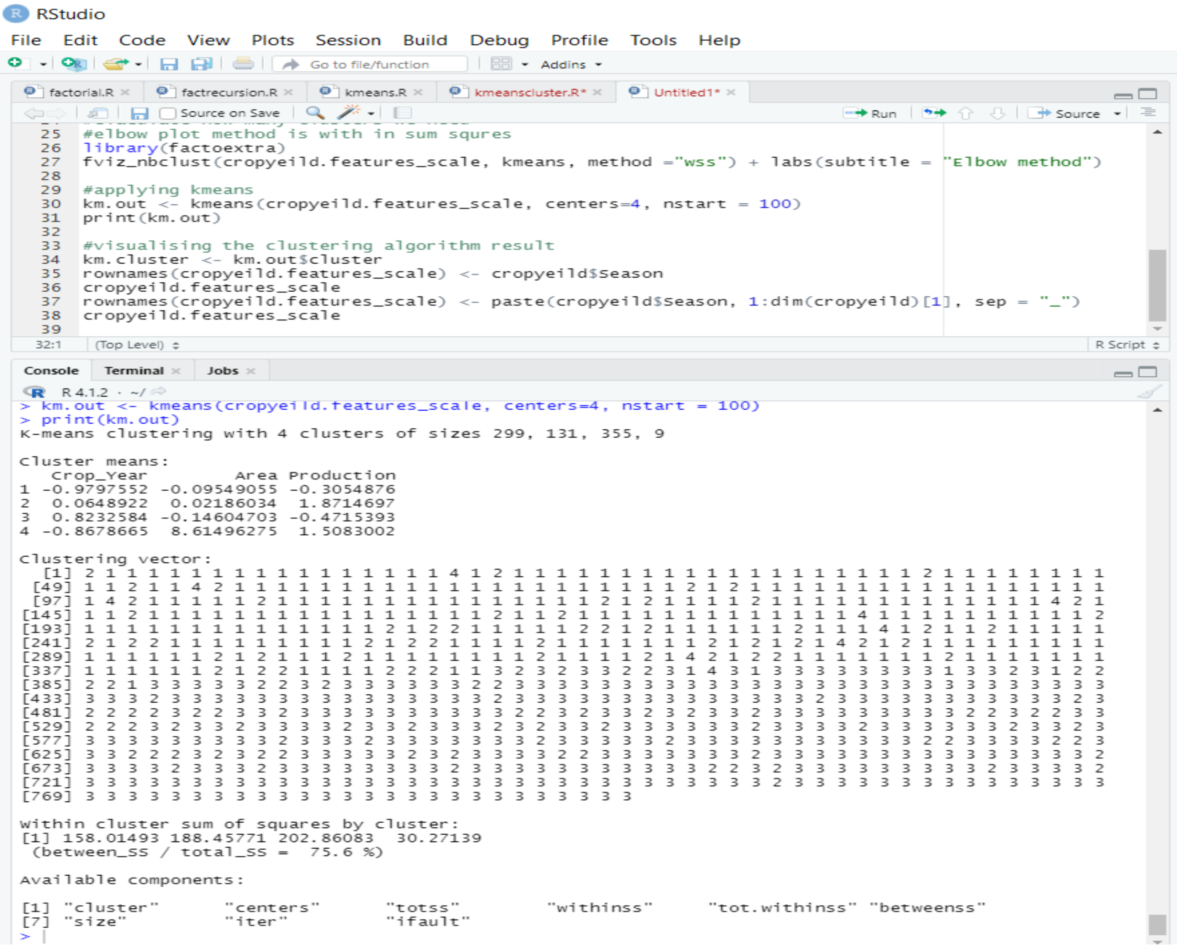
Distance metric

Elbow Method



Optimal Number of clusters

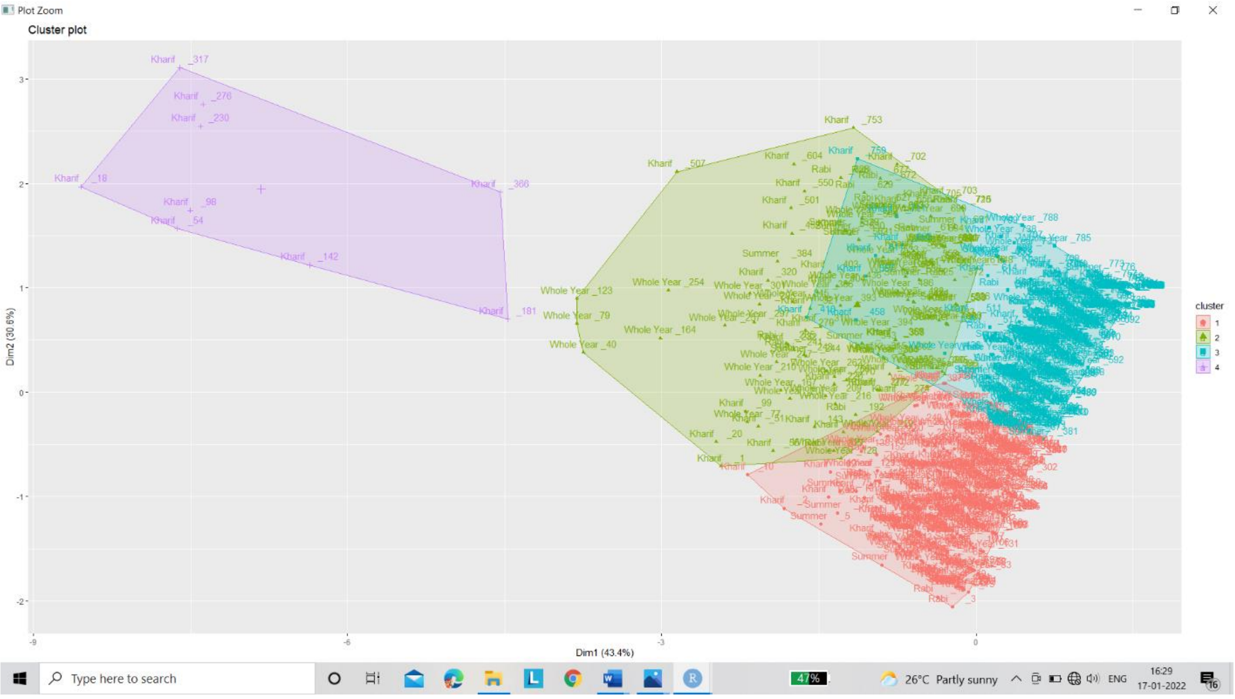
Applying K-Means Clustering

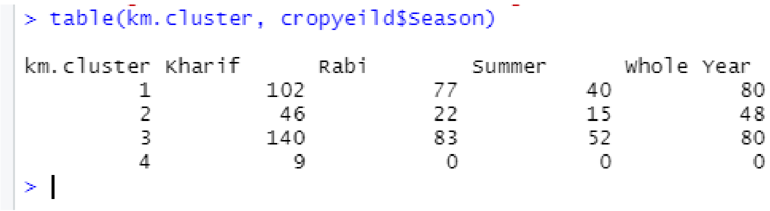


```r
#elbow plot method is with in sum squres
library(factoextra)
fviz_nbclust(cropyeild.features_scale, kmeans, method ="wss") + labs(subtitle = "Elbow method")

#applying kmeans
km.out <- kmeans(cropyeild.features_scale, centers=4, nstart = 100)
print(km.out)

#visualising the clustering algorithm result
km.cluster <- km.out$cluster
rownames(cropyeild.features_scale) <- cropyeild$Season
cropyeild.features_scale
rownames(cropyeild.features_scale) <- paste(cropyeild$Season, 1:dim(cropyeild)[1], sep = "_")
cropyeild.features_scale

```

```
> km.out <- kmeans(cropyeild.features_scale, centers=4, nstart = 100)
> print(km.out)
K-means clustering with 4 clusters of sizes 299, 131, 355, 9

Cluster means:
   Crop_Year        Area Production
1 -0.9797552 -0.09549055 -0.3054876
2  0.0648922  0.02186034  1.8714697
3  0.8232584 -0.14604703 -0.4715393
4 -0.8678665  8.61496275  1.5083002

Clustering vector:
  [1] 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 4 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1
 [49] 1 1 2 1 1 4 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [97] 1 4 2 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 2 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 4 2 1
[145] 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 4 1 1 1 1 1 1 1 1 1 1 2
[193] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 2 2 1 1 1 1 1 2 2 1 2 1 1 1 1 1 1 2 1 1 1 4 1 2 1 1 2 1 1 1 1 1
[241] 2 1 2 2 1 1 1 1 1 1 1 1 1 2 1 2 2 1 1 1 1 2 1 1 1 1 1 1 1 2 1 2 1 2 1 4 2 1 2 1 1 1 1 1 1 1 1 1
[289] 1 1 1 1 1 1 2 1 2 1 1 1 2 1 1 1 1 1 1 1 1 2 1 1 1 1 2 1 4 2 1 2 2 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1
[337] 1 1 1 1 1 1 2 1 2 2 1 1 1 1 2 2 2 1 1 3 2 3 2 3 3 2 2 3 1 4 3 1 3 3 3 3 3 3 3 3 1 3 3 2 3 1 2 2
[385] 2 2 1 3 3 3 3 3 2 2 3 2 3 3 3 3 3 3 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
[433] 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 2 3
[481] 2 2 2 2 3 2 2 3 3 2 3 3 3 3 3 3 3 3 3 3 2 2 3 2 3 3 2 3 2 3 3 2 3 3 3 3 3 3 3 3 3 2 2 3 2 2 3 3
[529] 2 2 2 3 2 3 3 2 3 3 3 3 2 3 3 2 3 3 3 2 3 2 3 3 2 3 3 3 3 3 3 2 3 3 3 3 2 3 3 3 3 3 3 2 3 3 2 3
[577] 3 3 3 3 3 3 3 3 3 2 3 3 3 2 3 3 3 3 3 3 3 2 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 2 2 3 3 3 3 2 2 3
[625] 3 3 2 2 2 3 2 3 2 2 3 3 3 3 3 2 3 2 3 3 3 3 2 2 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2
[673] 3 3 3 3 2 3 3 3 2 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 2 2 3 2 3 3 3 3 3 3 3 3 3 2 3 3 3 3 2
[721] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
[769] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3

Within cluster sum of squares by cluster:
[1] 158.01493 188.45771 202.86083  30.27139
 (between_SS / total_SS =  75.6 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"
[7] "size"         "iter"         "ifault"
> 
```

Visualizing the clustering algorithm Result for number of clusters K=4:





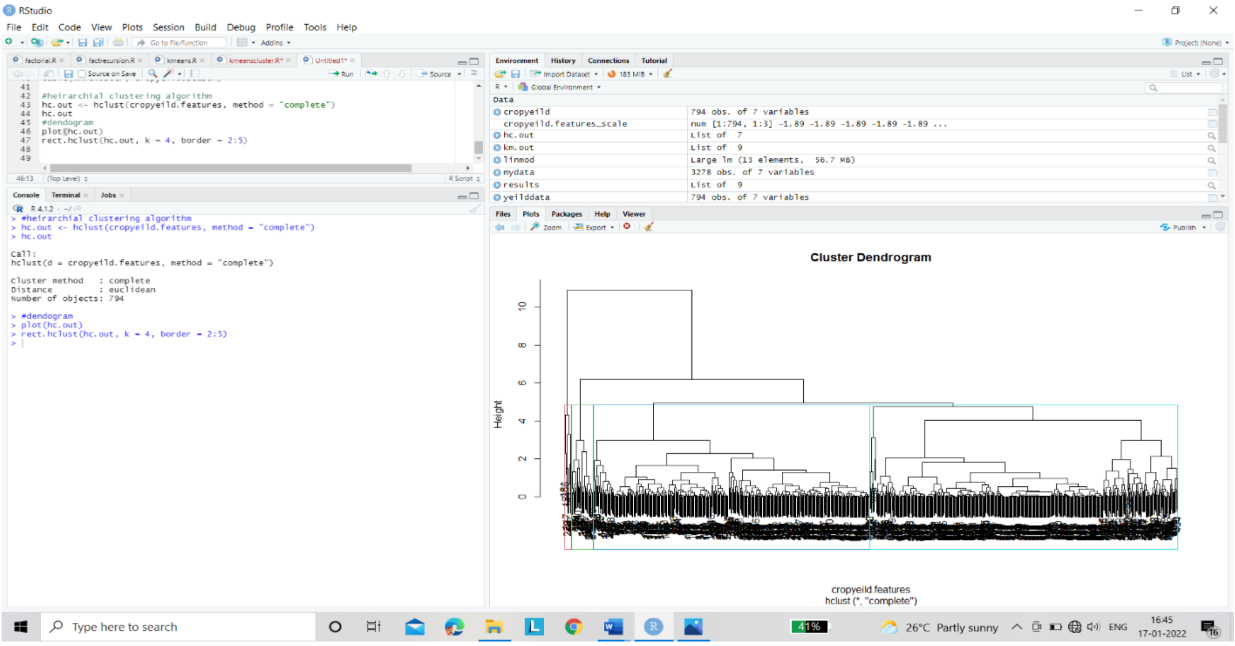
```
> table(km.cluster, cropyeild$Season)

km.cluster Kharif       Rabi        Summer      Whole Year
         1        102         77          40          80
         2         46         22          15          48
         3        140         83          52          80
         4          9          0           0           0
> |
```

Implementation of H-Clustering Algorithm

A **Hierarchical clustering** method works via grouping data into a tree of clusters. Hierarchical clustering begins by treating every data points as a separate cluster. Then, it repeatedly executes the subsequent steps:

1. Identify the clusters which can be closest together, and
2. Merge the maximum comparable clusters. We need to continue these steps until all the clusters are merged together.

In Hierarchical Clustering, the aim is to produce a hierarchical series of nested clusters. A diagram called **Dendrogram** (A Dendrogram is a tree-like diagram that statistics the sequences of merges or splits) graphically represents this hierarchy and is an inverted tree that describes the order in which factors are merged (bottom-up view) or cluster are break up (top-down view).



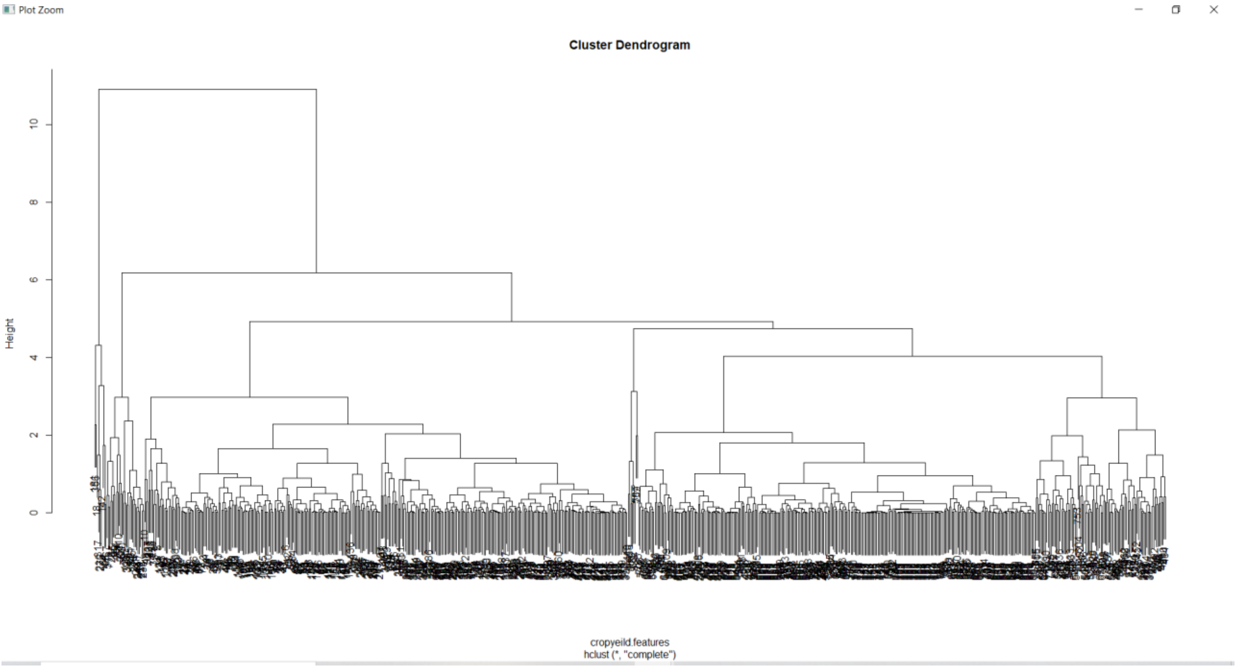
```
#heirarchial clustering algorithm
hc.out <- hclust(cropyeild.features, method = "complete")
hc.out
#dendogram
plot(hc.out)
rect.hclust(hc.out, k = 4, border = 2:5)
```

```
> #heirarchial clustering algorithm
> hc.out <- hclust(cropyeild.features, method = "complete")
> hc.out

Call:
hclust(d = cropyeild.features, method = "complete")

Cluster method   : complete
Distance         : euclidean
Number of objects: 794

> #dendogram
> plot(hc.out)
> rect.hclust(hc.out, k = 4, border = 2:5)
> 
```

```
Data
cropyeild                    794 obs. of 7 variables
  cropyeild.features_scale   num [1:794, 1:3] -1.89 -1.89 -1.89 -1.89 -1.89 ...
hc.out                       List of  7
km.out                       List of  9
linmod                       Large lm (13 elements,  56.7 MB)
mydata                       3278 obs. of 7 variables
results                      List of  9
yeilddata                    794 obs. of 7 variables
```

H-Clustering Implementation and Plotting Dendogram and to visualize this do rectangle for h cluster (k=4)

Plot of h-clustering Dendogram



CONCLUSION

In this we was able to do a predictive analysis of agricultural data to forecast the production of crop yields using K-means clustering. From the previously discussed results, I can clearly state that K-means clustering in R is a good decision for predicting crop yields. Making up such an analysis using previous data can be very fruitful especially for smallholder farmers that would like to implement precision agriculture in their crop yields' production.

This work, thus, helps them to differentiate between the samples that will have high production yields and the ones that will have low production yields.