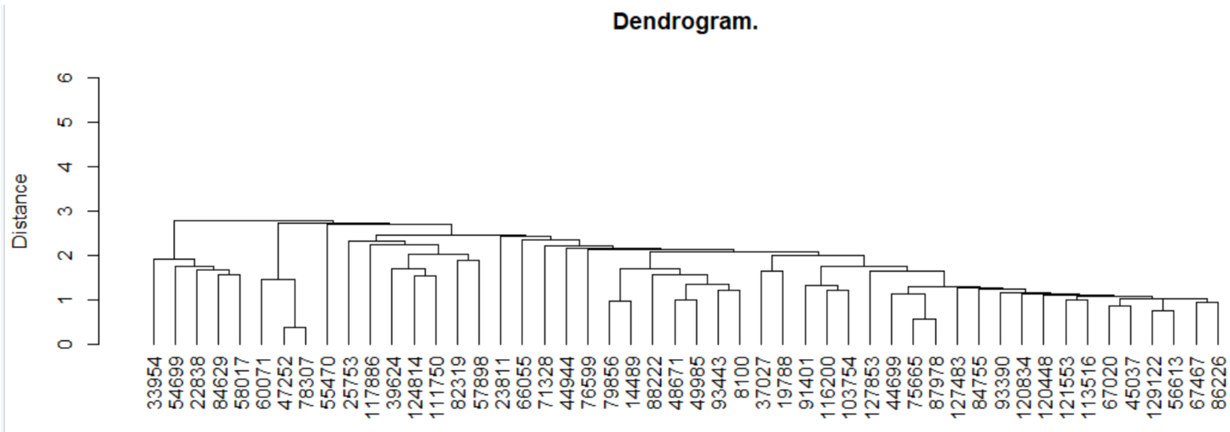
Cluster Analysis

```
install.packages("cluster", lib="/Library/Frameworks/R.framework/Versions/3.5/Resources/library")
library(cluster)
View(loan)
# take a random sample of size 50 from a dataset mydata
# sample without replacement
mysample <- loan[sample(1:nrow(loan), 50,replace=FALSE),]

# Standardizing the data with scale()
matstd.loan<- scale(mysample[,c(1,4,5,8,9,11,12)])
# Creating a (Euclidean) distance matrix of the standardized data
dist.employ <- dist(matstd.loan, method="euclidean")
# Invoking hclust command (cluster analysis by single linkage method)
clusemploy.nn <- hclust(dist.employ, method = "single")

#Plotting

# Create extra margin room in the dendrogram, on the bottom (Countries labels)
par(mar=c(8, 4, 4, 2) + 0.1)
# Object "clusemploy.nn" is converted into a object of class "dendrogram"
# in order to allow better flexibility in the (vertical) dendrogram plotting.
plot(as.dendrogram(clusemploy.nn),ylab="Distance",ylim=c(0,6),
    main="Dendrogram.")
```



We will use agnes function as it allows us to select option for data standardization, the distance measure and clustering algorithm in one single function

?agnes

(agn.employ <- agnes(mysample, metric="euclidean", stand=TRUE, method = "single"))

View(agn.employ)

```
Call:    agnes(x = mysample, metric = "euclidean", stand = TRUE, method = "single")
Agglomerative coefficient:  0.3965915
Order of objects:
 [1] 75665  56613  86226  129122 87978  120448 120834 67020  67467  45037  44699  93390
[13] 84755  127483 49985  127853 88222  79856  93443  8100   113516 117886 25753  19788
[25] 14489  44944  124814 111750 48671  121553 37027  57898  23811  116200 71328  103754
[37] 91401  39624  76599  66055  55470  60071  47252  78307  84629  58017  54699  22838
[49] 33954  82319
Height (summary):
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.7899  2.2374  3.0788  2.8983  3.3932  4.4512

Available components:
[1] "order"     "height"    "ac"        "merge"     "diss"      "call"      "method"
[8] "order.lab" "data"
```

Description of cluster merging

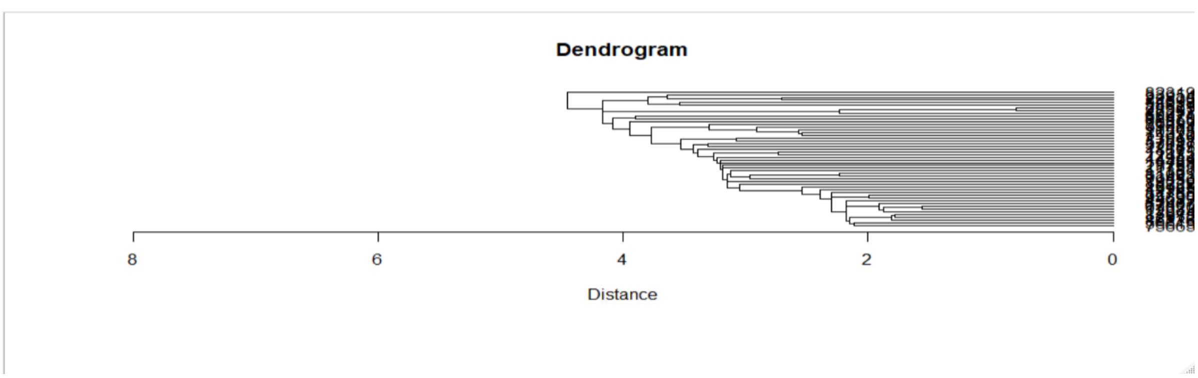
agn.employ$merge

```
> agn.employ$merge
      [,1] [,2]
 [1,]  -47  -50
 [2,]  -29  -30
 [3,]  -26  -43
 [4,]  -23    3
 [5,]   -2    2
 [6,]    5   -7
 [7,]  -19  -44
 [8,]   -1  -27
 [9,]    8    4
[10,]    6  -46
[11,]    9   10
[12,]  -16  -39
[13,]  -42    1
[14,]   11    7
[15,]   14  -41
[16,]   15  -40
[17,]  -14  -17
[18,]   16  -11
[19,]   17  -31
[20,]  -10  -48
[21,]  -22  -33
[22,]   19  -18
[23,]   -4   12
[24,]   18   -3
[25,]  -34  -36
[26,]   23  -13
[27,]   24  -20
[28,]   27   26
[29,]   28  -21
[30,]   29  -32
[31,]   30  -28
[32,]   31  -35
[33,]   32  -37
[34,]   33   21
[35,]   22  -49
[36,]   -8  -25
[37,]   34   -5
[38,]   37   36
[39,]   38   25
[40,]   -9  -12
[41,]   20  -24
[42,]   39   35
[43,]   40   41
[44,]   -6  -45
[45,]   42  -38
[46,]   45   44
[47,]   46   13
[48,]   47   43
[49,]   48  -15
```

#Dendogram

plot(as.dendrogram(agn.employ), xlab= "Distance",xlim=c(8,0),

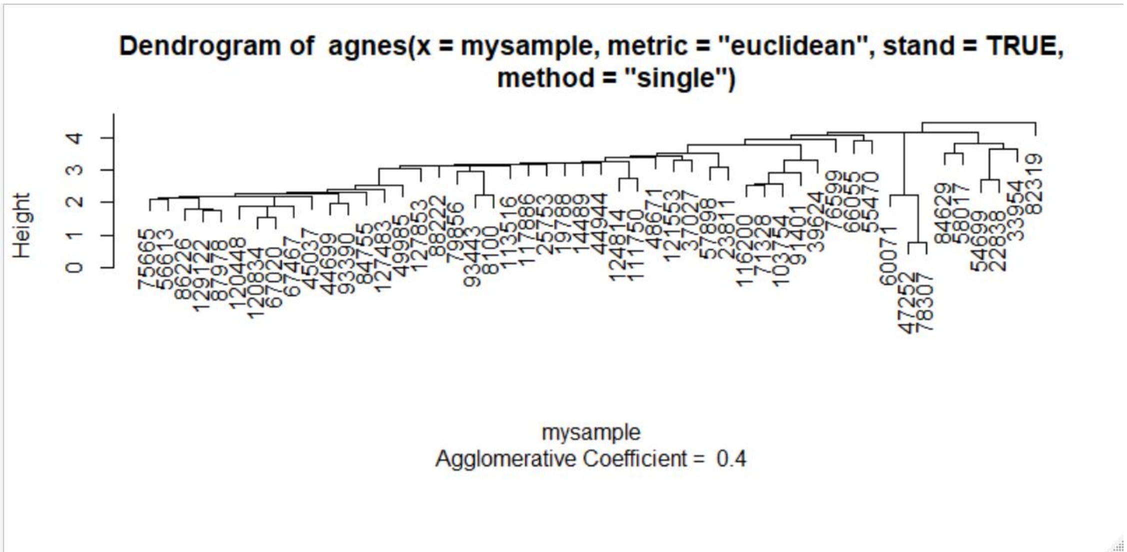
horiz = TRUE,main="Dendrogram")

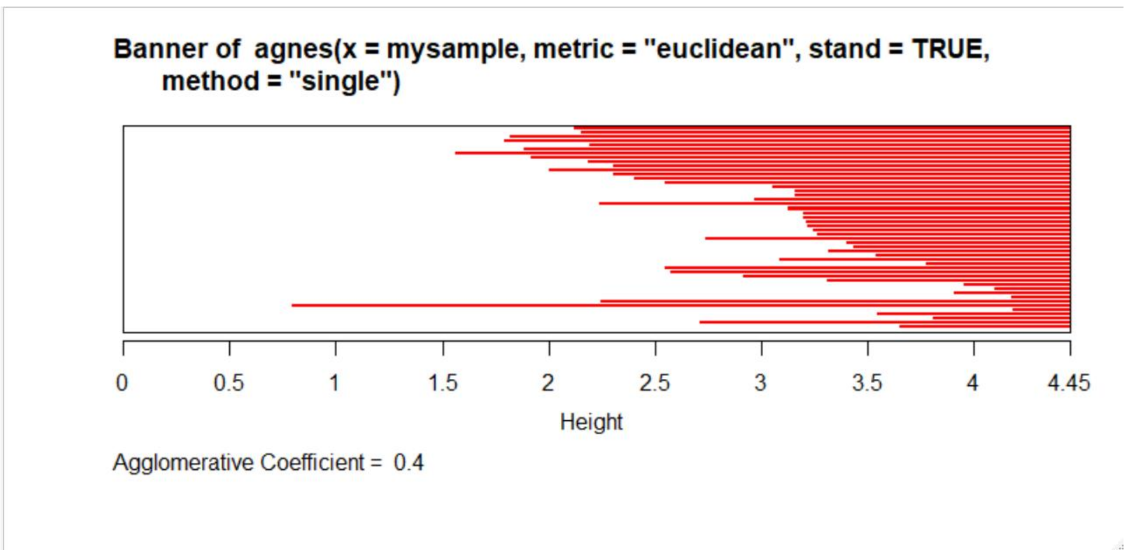


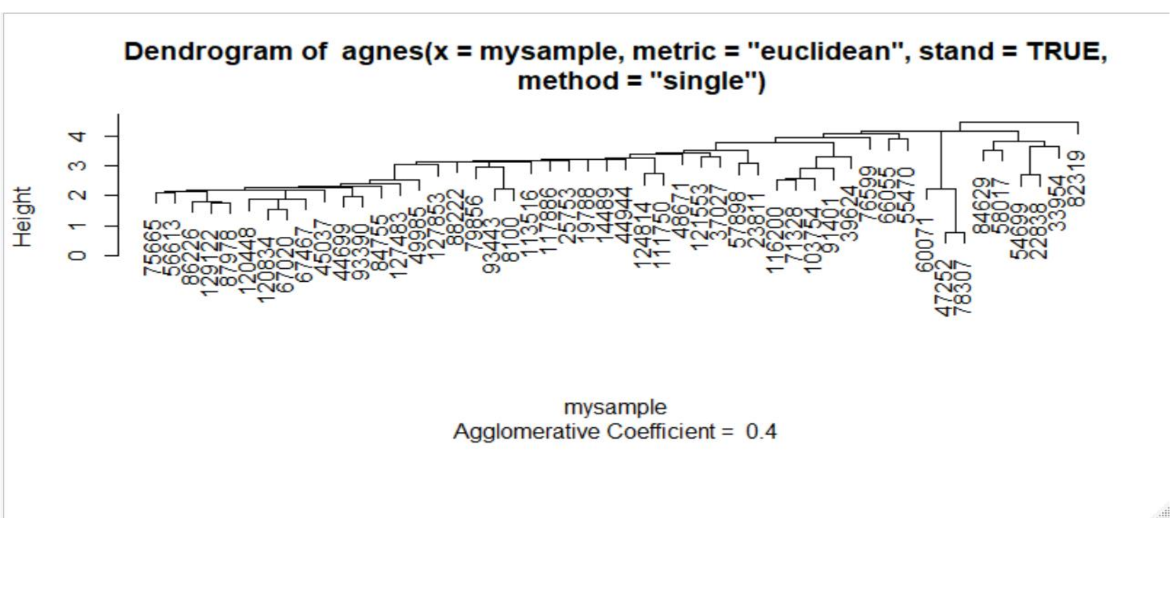
#Interactive Plots

plot(agn.employ,ask=TRUE)

plot(agn.employ, which.plots=2)


Dendrogram of agnes(x = mysample, metric = "euclidean", stand = TRUE, method = "single")
Height
0 1 2 3 4
mysample
Agglomerative Coefficient = 0.4


Banner of agnes(x = mysample, metric = "euclidean", stand = TRUE, method = "single")
0 0.5 1 1.5 2 2.5 3 3.5 4 4.45
Height
Agglomerative Coefficient = 0.4


Dendrogram of agnes(x = mysample, metric = "euclidean", stand = TRUE, method = "single")
Height
0 1 2 3 4
mysample
Agglomerative Coefficient = 0.4

Conclusion - As shown above we have implemented the cluster analysis on a sample of 50 rows. Since these 50 rows are random it does not give a clear view of the clusters. In this case dendrogram is not the most efficient way of cluster representation. We already know that there are 2 groups in our data namely Defaulters and Non defaulters hence are now going to implement K means clustering for better understanding of cluster formation on our dataset.

K Means Clustering

```
> loan[1:12] <- lapply(loan[1:9], as.numeric)

> str(loan)
'data.frame':	130718 obs. of  12 variables:
 $ loan_status   : num  1 1 1 1 1 1 1 1 1 1 ...
 $ loan_amnt     : num  4500 2500 4000 1000 5000 ...
 $ int_rate      : num  11.31 13.56 17.97 23.4 7.56 ...
 $ issue_d       : num  2018 2018 2018 2018 2018 ...
 $ purpose       : num  0 1 1 0 0 1 1 1 0 0 ...
 $ dti           : num  4.64 15.09 19.1 20.78 14.78 ...
 $ emp_length    : num  10 5 5 3 5 1 1 1 1 8 ...
 $ home_ownership: num  0 0 1 0 1 1 1 0 0 1 ...
 $ annual_inc    : num  38500 42000 60000 60000 98000 ...
 $ term          : num  1 1 1 1 1 1 1 1 1 1 ...
 $ loan_amnt.1   : num  4500 2500 4000 1000 5000 ...
 $ int_rate.1    : num  11.31 13.56 17.97 23.4 7.56 ...
 - attr(*, "na.action")= 'omit' Named int  2 5 6 7 10 12 13 15 16 17 ...
  ..- attr(*, "names")= chr  "152" "215" "269" "271" ...

> matstd.loan <- scale(loan[,1:9])
```

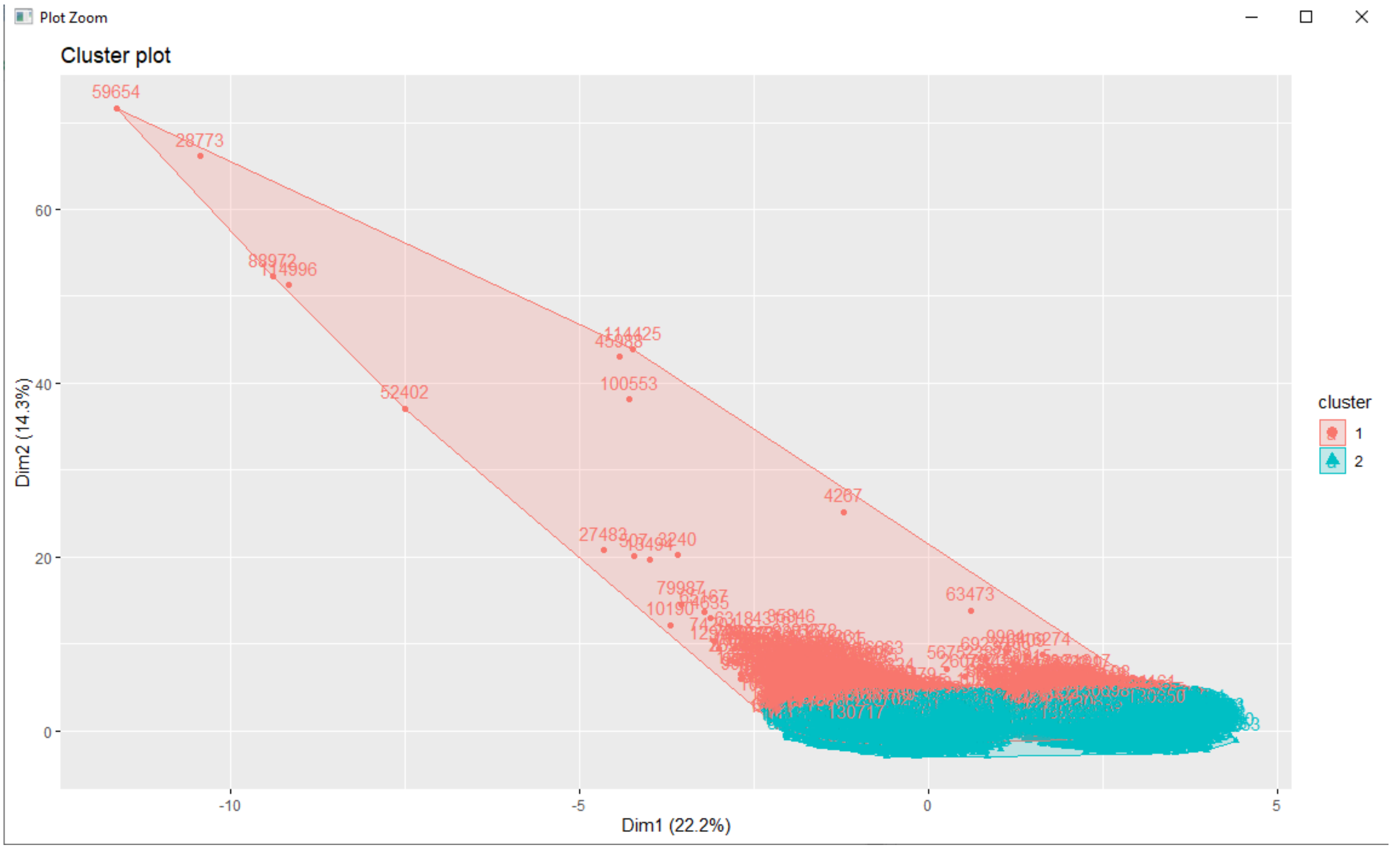
<u>This is plot for k = 2</u>

```
> k2 <- kmeans(loan, centers = 2, nstart = 30)

> str(k2)
List of 9
 $ cluster     : int [1:130718] 2 2 2 2 2 2 2 2 2 2 ...
 $ centers     : num [1:2, 1:10] 8.14e-01 7.56e-01 2.19e+04 1.35e+04 1.25e+01
...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:2] "1" "2"
  .. ..$ : chr [1:10] "loan_status" "loan_amnt" "int_rate" "issue_d" ...
 $ totss       : num 7.71e+14
 $ withinss    : num [1:2] 4.79e+14 9.20e+13
 $ tot.withinss: num 5.71e+14
 $ betweenss   : num 2.01e+14
 $ size        : int [1:2] 12568 118150
 $ iter        : int 1
 $ ifault      : int 0
 - attr(*, "class")= chr "kmeans"

> k2
K-means clustering with 2 clusters of sizes 12568, 118150

Cluster means:
  loan_status loan_amnt int_rate  issue_d   purpose      dti emp_length
```

```
1   0.8138129  21862.73 12.45166 2016.199 0.2720401 14.05636   6.338001
2   0.7556073  13476.26 13.68950 2016.154 0.2266780 19.69935   6.366196
  home_ownership annual_inc      term
1      0.7713240  203313.40 0.8138129
2      0.6212019   70681.98 0.7556073

Clustering vector:
   [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 1 2
2 2 2 2
  [40] 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2
  [79] 2 2 2 2 1 2 2 2 2 2 2 2 2 1 2 2 2 1 2 2 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2
 [118] 1 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 1
 [157] 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 1 2 2 2 2 2 2 2 2 2 2 1 2 2 1
2 2 1 2
 [196] 2 2 2 2 1 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1
2 2 1 2
 [235] 2 2 2 2 2 2 2 1 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2
2 2 2 1
 [274] 2 2 2 2 2 2 2 2 2 1 2 2 2 1 2 2 2 2 2 1 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2
 [313] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 2 1 2 2 1 2 2 2
2 2 2 2
 [352] 2 2 2 2 1 2 2 2 2 1 2 2 2 2 2 2 1 2 1 1 2 2 2 2 2 2 2 2 1 2 2 2 2 2 1
2 2 2 2
 [391] 1 2 2 2 2 2 2 2 2 2 2 1 2 1 1 2 2 2 2 2 2 2 2 2 2 1 1 1 2 2 2 2 2 2 1
2 2 2 2
 [430] 2 2 2 2 2 2 2 2 2 2 1 2 2 2 1 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2
 [469] 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 1 2 1 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2
2 2 2 1
 [508] 2 2 2 1 2 2 1 2 2 2 1 2 2 2 2 2 2 1 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2
2 1 2 1
 [547] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 1 2
2 2 2 2
 [586] 2 1 2 2 1 1 2 2 2 1 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2
2 2 2 2
 [625] 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 1 2 2 2 2 2 2
1 2 2 2
 [664] 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 1 2 1 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 1 2
2 2 2 2
 [703] 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 1 1 2 1 2 1 2 2 2 2 2 2 2 2 1 1 1 2 2
2 2 2 2
 [742] 1 2 2 2 2 1 2 1 2 2 2 2 2 2 2 2 2 2 1 2 1 2 2 2 2 2 2 2 1 2 1 2 2 2 1
2 2 1 1
 [781] 2 2 2 1 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2
 [820] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 1 2 2 2 2 2 1 1 2 1 2 1 2 2 2
2 2 1 2
 [859] 2 2 2 2 2 1 2 1 2 2 1 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2
 [898] 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 1
 [937] 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 1 2
2 2 2 2
 [976] 2 1 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2
 [ reached getOption("max.print") -- omitted 129718 entries ]

Within cluster sum of squares by cluster:
[1] 4.786701e+14 9.204252e+13
 (between_SS / total_SS =  26.0 %)
```

```
Available components:

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

```
> fviz_cluster(k2, data = loan)
```



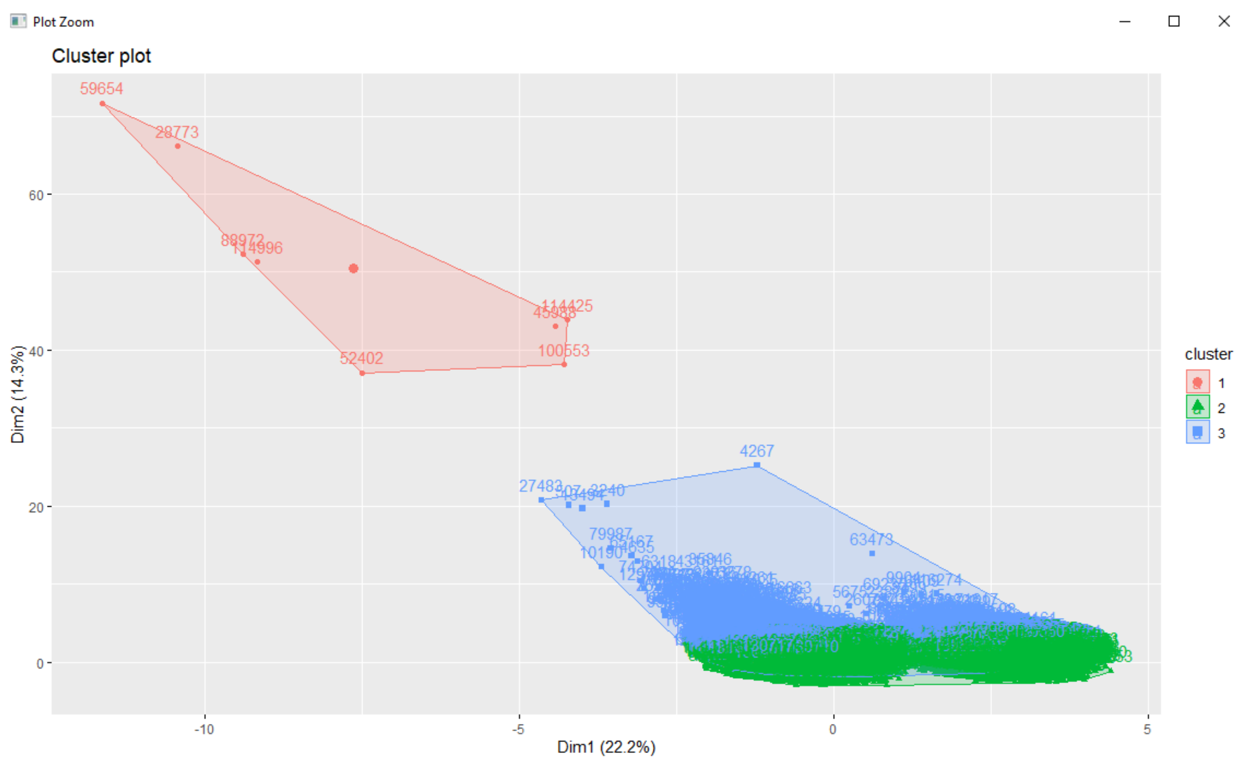

```
> perc.var.2 <- round(100*(1 - k2$betweenss/k2$totss),1)

> names(perc.var.2) <- "Perc. 2 clus"

> perc.var.2
Perc. 2 clus
          74
```

<u>This is a plot for k=3</u>

```
> k3 <- kmeans(loan, centers = 3, nstart = 30)

> str(k3)
List of 9
 $ cluster     : int [1:130718] 2 2 2 2 2 2 2 2 2 2 ...
 $ centers     : num [1:3, 1:10] 6.25e-01 7.53e-01 8.08e-01 1.30e+04 1.30e+04
...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:3] "1" "2" "3"
  .. ..$ : chr [1:10] "loan_status" "loan_amnt" "int_rate" "issue_d" ...
```

```
 $ totss       : num 7.71e+14
 $ withinss    : num [1:3] 1.87e+13 6.52e+13 1.77e+14
 $ tot.withinss: num 2.6e+14
 $ betweenss   : num 5.11e+14
 $ size        : int [1:3] 8 110235 20475
 $ iter        : int 2
 $ ifault      : int 0
 - attr(*, "class")= chr "kmeans"

> k3
K-means clustering with 3 clusters of sizes 8, 110235, 20475

Cluster means:
  loan_status loan_amnt int_rate  issue_d   purpose      dti emp_length
1   0.6250000  12968.75 10.71875 2016.000 0.3750000  0.25125   7.500000
2   0.7525378  13049.77 13.74208 2016.152 0.2259899 19.94149   6.351848
3   0.8079121  20920.46 12.64775 2016.194 0.2581685 14.93951   6.425690
  home_ownership annual_inc      term
1      0.6250000 6434514.12 0.6250000
2      0.6119744   66739.23 0.7525378
3      0.7630281  170834.85 0.8079121

Clustering vector:
   [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 3 2
2 2 2 3
  [40] 2 2 2 2 2 2 2 2 3 3 3 2 2 2 2 2 2 2 2 2 2 2 3 3 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2
  [79] 2 2 2 2 3 2 2 2 3 2 2 2 2 3 3 2 2 3 2 2 3 3 2 3 2 3 2 2 2 3 2 2 2 2 2
2 2 2 3
 [118] 3 2 2 2 3 2 2 2 2 3 2 2 2 2 2 2 3 2 2 2 3 2 2 2 2 2 2 2 3 2 2 2 2 2 3
2 2 2 3
 [157] 2 3 3 2 2 2 3 2 2 2 2 3 2 3 2 2 3 2 2 2 3 2 2 2 2 2 2 2 2 3 2 3 2 2 3
2 2 3 3
 [196] 2 2 2 3 3 2 2 2 3 3 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3
2 2 3 2
 [235] 2 2 2 2 2 2 3 3 2 2 2 3 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2
2 2 2 3
 [274] 2 3 2 2 2 2 3 2 2 3 2 2 2 3 3 2 2 2 2 3 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2
 [313] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 3 2 2 2 2 2 2 3 2 2 3 2 2 2
2 2 2 2
 [352] 2 3 2 2 3 2 2 2 2 3 2 2 2 2 2 2 3 2 3 3 2 2 2 2 3 2 2 2 3 2 2 2 2 2 3
3 3 2 2
 [391] 3 2 2 2 2 2 2 2 2 2 2 2 3 2 3 3 2 2 2 2 2 2 2 2 2 2 2 3 3 3 2 2 2 2 2 2 3
2 2 2 2
 [430] 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 3 2 3 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2
 [469] 2 2 2 2 2 3 2 2 2 2 2 2 3 2 2 3 2 3 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 3 2
3 2 2 3
 [508] 2 2 2 3 2 2 3 2 2 2 3 2 2 2 2 2 2 2 3 2 2 3 2 2 3 3 2 2 2 2 2 3 2 2 2 2
2 3 2 3
 [547] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 2 2 2 2 3 2 2 2 2 2 2 3 2 2 2 2 2 2 3 2
2 2 2 2
 [586] 2 3 2 2 3 3 2 2 2 3 3 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2
2 2 2 2
 [625] 2 2 3 2 2 3 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 3 2 2 2 3 2 2
3 2 2 2
 [664] 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 3 2 3 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 3 2
2 2 2 2
 [703] 2 2 2 2 2 2 2 2 3 2 2 2 2 2 3 2 3 3 2 3 3 3 2 2 2 2 2 2 3 2 2 3 3 3 2 2
2 2 2 2
 [742] 3 2 2 2 2 3 2 3 2 2 2 2 2 2 2 2 2 2 3 2 3 2 2 2 2 2 2 3 3 2 3 2 2 3 3
3 2 3 3
```

```
 [781] 2 2 2 3 2 2 2 3 3 2 2 2 2 2 2 2 2 2 2 2 3 3 3 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2
 [820] 2 2 2 2 2 2 2 3 2 3 2 2 2 2 2 3 2 2 3 2 3 2 3 2 2 2 3 3 2 3 2 3 2 2 2
2 2 3 2
 [859] 2 2 2 2 2 3 3 3 2 2 3 2 2 2 2 2 2 3 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2
 [898] 2 2 3 2 3 2 2 2 2 2 3 2 2 2 2 2 2 3 2 2 2 3 2 3 2 2 2 2 2 2 2 2 2 2 2
2 2 2 3
 [937] 2 2 2 2 2 3 2 3 2 2 2 2 2 2 2 3 3 2 2 2 2 2 3 2 2 2 2 2 2 3 3 2 2 3 2
2 2 2 2
 [976] 2 3 2 2 2 2 2 2 3 2 2 2 2 2 3 2 2 3 2 2 2 3 2 2 2
 [ reached getOption("max.print") -- omitted 129718 entries ]

Within cluster sum of squares by cluster:
[1] 1.866222e+13 6.517413e+13 1.766134e+14
 (between_SS / total_SS =  66.2 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss
"
[6] "betweenss"    "size"         "iter"         "ifault"

> fviz_cluster(k3, data = loan)
```




```
> perc.var.3 <- round(100*(1 - k3$betweenss/k3$totss),1)

> names(perc.var.3) <- "Perc. 3 clus"

> perc.var.3
Perc. 3 clus
        33.8
```

Conclusion - In our example we can see that there are more than 2 variables(dimensions) to perform clustering hence kmeans by default calculates the principal component analysis of the variables and plots the first two principal components to explain the majority of variance. Here we can conclude that the percentage of variance for pc1(22.2%) and pc2(14.3%) in case of k = 2 is quite low . Therefore the clusters are not visibly separate from each other. We have also tried to plot cluster for k = 3 just for analysis of higher no. of k values.