Clustering Assignment

1. Analyze the given telecom customer dataset to identify customer segments
2. Write a report on the customer segments. For each segment, specify
   a. Segment size (percentage)
   b. Segment characteristics (important features) and description (narrative)
   c. Segment revenue share (% of revenue)
   d. Percent of segment customers at risk of leaving
   e. Revenue at risk per segment
3. Based on above analysis, present recommendations. Also include a brief technical summary in your report describing methods used that address rubric items 2-5 (Separate technical report not needed if using R Notebooks).

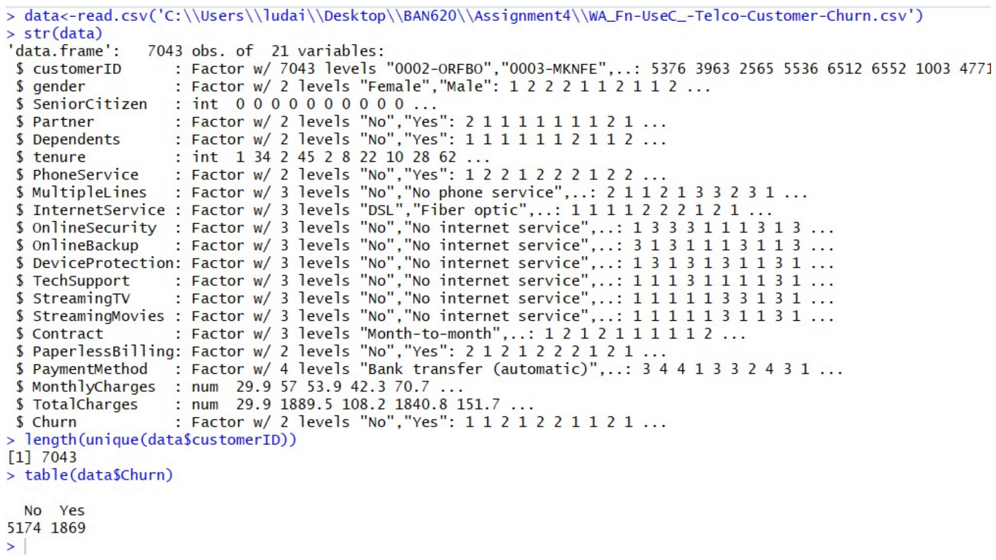
Extra-credit (10 points)

1. Develop a Shiny App to help analyze customer segments that has
   a. Multiple inputs (No of clusters, method options)
   b. Visualizations of cluster features (important features and relative weights)
   c. Visualizations for cluster/segment metrics listed in 2
   d. Narrative that describes business problem, approach and instructions on how to interpret results/visualizations
   e. Publish on ShinyApps, submit screenshot and code.
      Shiny apps/collab filterwn/R notebook

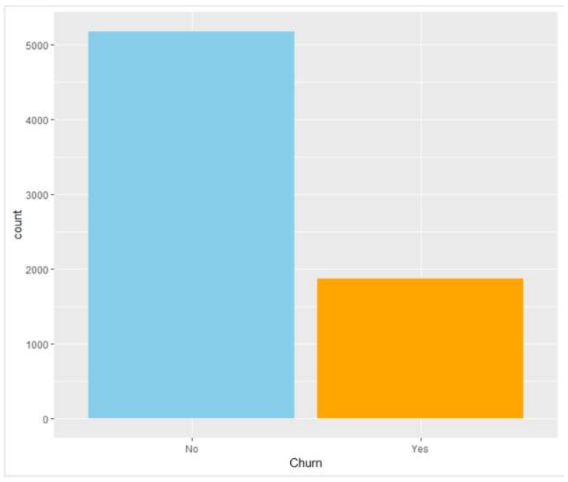
Rubric

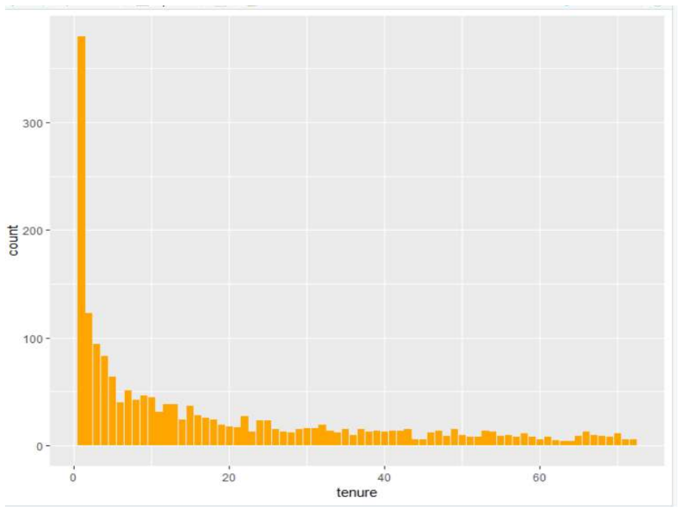
1. All items in 2a-2e about customer segments are included - 30
2. Multiple clustering algorithms/distance methods are evaluated - 20
3. An analysis of best number of clusters or distance metric is included -20
4. Report includes visualizations -20
5. Appropriate pre-processing is used – 10

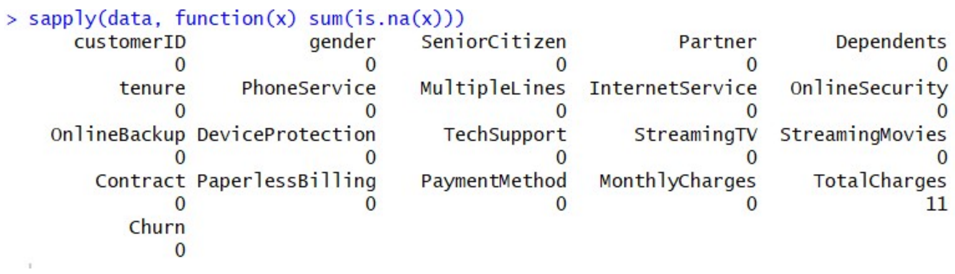
Descriptive data



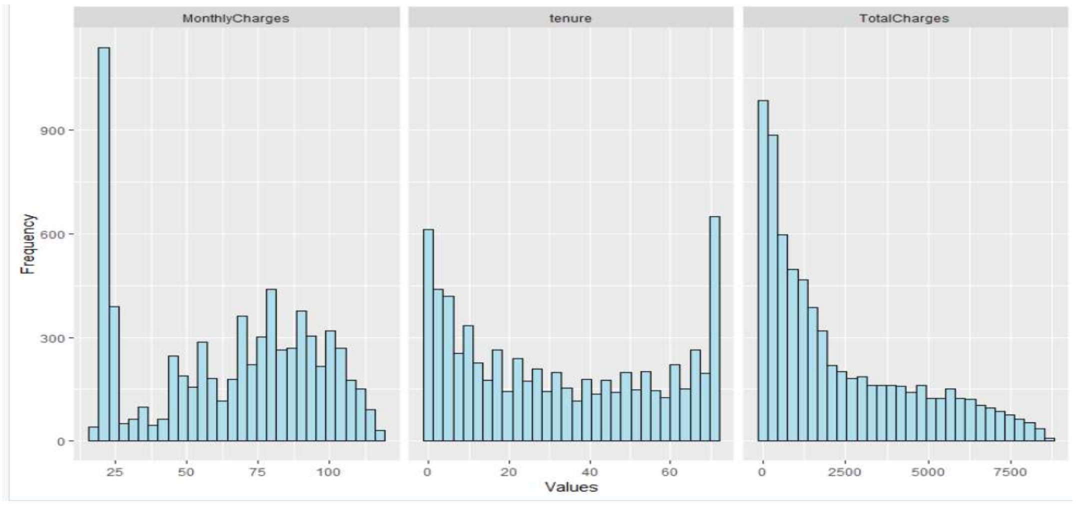
```
> data<-read.csv('C:\\Users\\ludai\\Desktop\\BAN620\\Assignment4\\WA_Fn-UseC_-Telco-Customer-Churn.csv')
> str(data)
'data.frame':	7043 obs. of  21 variables:
 $ customerID      : Factor w/ 7043 levels "0002-ORFBO","0003-MKNFE",..: 5376 3963 2565 5536 6512 6552 1003 4771
 $ gender          : Factor w/ 2 levels "Female","Male": 1 2 2 2 1 1 2 1 1 2 ...
 $ SeniorCitizen   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Partner         : Factor w/ 2 levels "No","Yes": 2 1 1 1 1 1 1 1 2 1 ...
 $ Dependents      : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 2 1 1 2 ...
 $ tenure          : int  1 34 2 45 2 8 22 10 28 62 ...
 $ PhoneService    : Factor w/ 2 levels "No","Yes": 1 2 2 1 2 2 2 1 2 2 ...
 $ MultipleLines   : Factor w/ 3 levels "No","No phone service",..: 2 1 1 2 1 3 3 2 3 1 ...
 $ InternetService : Factor w/ 3 levels "DSL","Fiber optic",..: 1 1 1 1 2 2 2 1 2 1 ...
 $ OnlineSecurity  : Factor w/ 3 levels "No","No internet service",..: 1 3 3 3 1 1 1 3 1 3 ...
 $ OnlineBackup    : Factor w/ 3 levels "No","No internet service",..: 3 1 3 1 1 1 3 1 1 3 ...
 $ DeviceProtection: Factor w/ 3 levels "No","No internet service",..: 1 3 1 3 1 3 1 1 3 1 ...
 $ TechSupport     : Factor w/ 3 levels "No","No internet service",..: 1 1 1 3 1 1 1 1 3 1 ...
 $ StreamingTV     : Factor w/ 3 levels "No","No internet service",..: 1 1 1 1 1 3 3 1 3 1 ...
 $ StreamingMovies : Factor w/ 3 levels "No","No internet service",..: 1 1 1 1 1 3 1 1 3 1 ...
 $ Contract        : Factor w/ 3 levels "Month-to-month",..: 1 2 1 2 1 1 1 1 1 2 ...
 $ PaperlessBilling: Factor w/ 2 levels "No","Yes": 2 1 2 1 2 2 2 1 2 1 ...
 $ PaymentMethod   : Factor w/ 4 levels "Bank transfer (automatic)",..: 3 4 4 1 3 3 2 4 3 1 ...
 $ MonthlyCharges  : num  29.9 57 53.9 42.3 70.7 ...
 $ TotalCharges    : num  29.9 1889.5 108.2 1840.8 151.7 ...
 $ Churn           : Factor w/ 2 levels "No","Yes": 1 1 2 1 2 2 1 1 2 1 ...
> length(unique(data$customerID))
[1] 7043
> table(data$Churn)

  No  Yes 
5174 1869 
> 
```





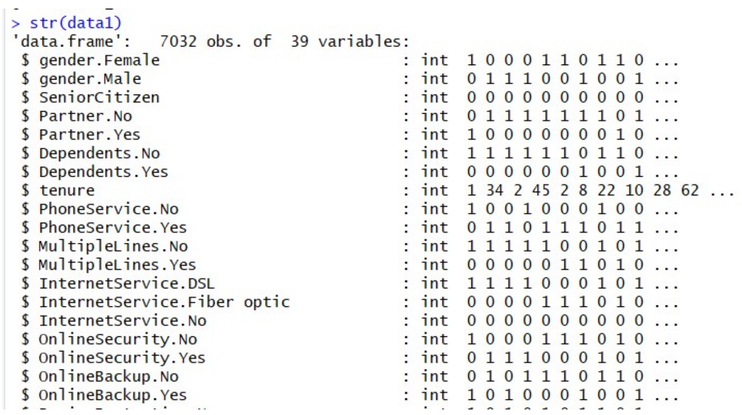


```
> sapply(data, function(x) sum(is.na(x)))
      customerID           gender    SeniorCitizen          Partner       Dependents 
               0                0                0                0                0 
          tenure     PhoneService    MultipleLines  InternetService   OnlineSecurity 
               0                0                0                0                0 
    OnlineBackup DeviceProtection      TechSupport      StreamingTV  StreamingMovies 
               0                0                0                0                0 
        Contract PaperlessBilling    PaymentMethod   MonthlyCharges     TotalCharges 
               0                0                0                0               11 
           Churn 
               0 
```



# Data Preparation

The raw data contains 7043 rows (customers) and 21 columns (features). I use sapply to check the number if missing values in each column. We found that there are 11 missing values in "TotalCharges" columns. Since the Monthly Charges and Total Charges are correlated. So, one of them can be removed from the model. I remove TotalCharges with missing values. Looking at the data structure, some data columns need recoding. For instance, changing values from "No phone service" and "No internet service" to "No", for consistency. The "Churn" column is our target. There are 5174 "No" and 1869 "Yes" in Churn column. The churn mostly happens in short tenure. Since we need to identify customer segments, factor variables should be changed to dummy variables.

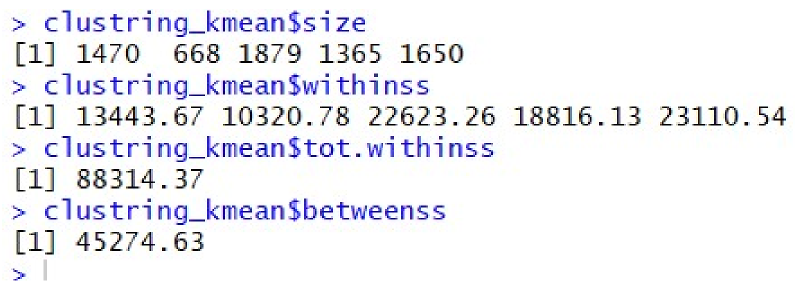


```
> str(data1)
'data.frame':	7032 obs. of  39 variables:
 $ gender.Female                   : int  1 0 0 0 1 1 0 1 1 0 ...
 $ gender.Male                     : int  0 1 1 1 0 0 1 0 0 1 ...
 $ SeniorCitizen                   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Partner.No                      : int  0 1 1 1 1 1 1 1 0 1 ...
 $ Partner.Yes                     : int  1 0 0 0 0 0 0 0 1 0 ...
 $ Dependents.No                   : int  1 1 1 1 1 1 0 1 1 0 ...
 $ Dependents.Yes                  : int  0 0 0 0 0 0 1 0 0 1 ...
 $ tenure                          : int  1 34 2 45 2 8 22 10 28 62 ...
 $ PhoneService.No                 : int  1 0 0 1 0 0 0 1 0 0 ...
 $ PhoneService.Yes                : int  0 1 1 0 1 1 1 0 1 1 ...
 $ MultipleLines.No                : int  1 1 1 1 1 0 0 1 0 1 ...
 $ MultipleLines.Yes               : int  0 0 0 0 0 1 1 0 1 0 ...
 $ InternetService.DSL             : int  1 1 1 1 0 0 0 1 0 1 ...
 $ InternetService.Fiber optic     : int  0 0 0 0 1 1 1 0 1 0 ...
 $ InternetService.No              : int  0 0 0 0 0 0 0 0 0 0 ...
 $ OnlineSecurity.No               : int  1 0 0 0 1 1 1 0 1 0 ...
 $ OnlineSecurity.Yes              : int  0 1 1 1 0 0 0 1 0 1 ...
 $ OnlineBackup.No                 : int  0 1 0 1 1 1 0 1 1 0 ...
 $ OnlineBackup.Yes                : int  1 0 1 0 0 0 1 0 0 1 ...
```

# Create clustering

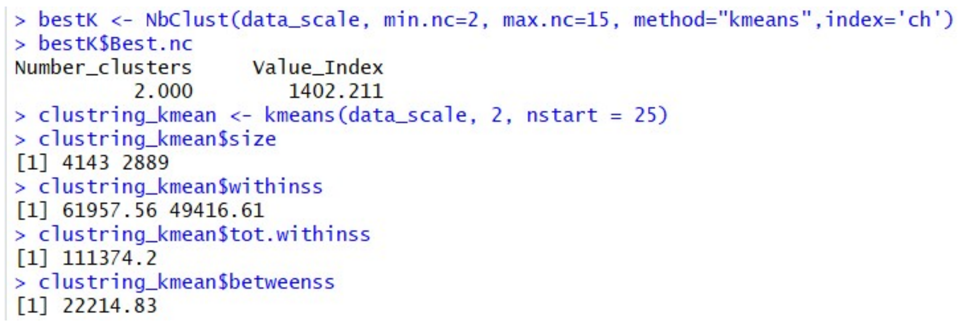
# Using K-mean k=5 clusters

```
data_scale<-  scale(data)

clustring_kmean <- kmeans(data_scale, 5, nstart = 25)

clustring_kmean$size

clustring_kmean$withinss

clustring_kmean$tot.withinss

clustring_kmean$betweenss
```



```
> clustring_kmean$size
[1] 1470  668 1879 1365 1650
> clustring_kmean$withinss
[1] 13443.67 10320.78 22623.26 18816.13 23110.54
> clustring_kmean$tot.withinss
[1] 88314.37
> clustring_kmean$betweenss
[1] 45274.63
> 
```

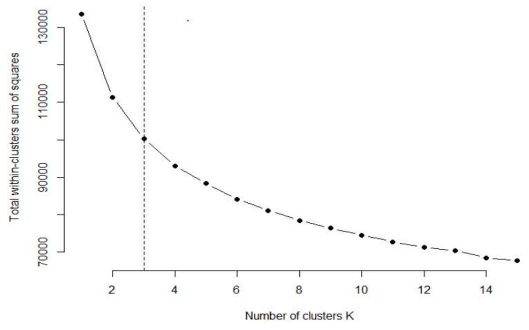
# using NbClust function for number of clusters



```
> bestK <- NbClust(data_scale, min.nc=2, max.nc=15, method="kmeans",index='ch')
> bestK$Best.nc
Number_clusters     Value_Index
          2.000        1402.211
> clustring_kmean <- kmeans(data_scale, 2, nstart = 25)
> clustring_kmean$size
[1] 4143 2889
> clustring_kmean$withinss
[1] 61957.56 49416.61
> clustring_kmean$tot.withinss
[1] 111374.2
> clustring_kmean$betweenss
[1] 22214.83
```

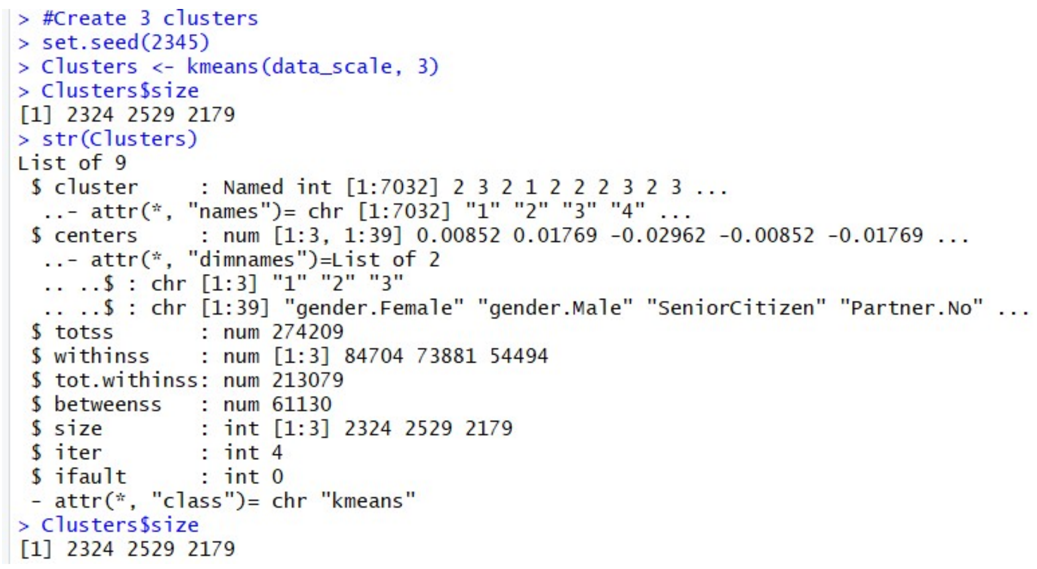
# Using Elbow method for number of clustering

```
> # Compute and plot wss for k = 2 to k = 15
> k.max <- 15 # Maximal number of clusters
> data <- data_scale
> wss <- sapply(1:k.max,
+               function(k){kmeans(data, k, nstart=10 )$tot.withinss})
> plot(1:k.max, wss,
+      type="b", pch = 19, frame = FALSE,
+      xlab="Number of clusters K",
+      ylab="Total within-clusters sum of squares")
> abline(v = 3, lty =2)
```



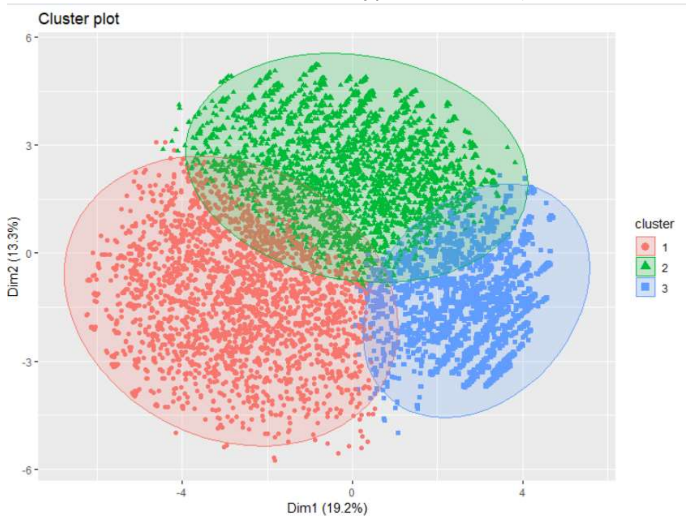
Elbow method gets an optimal number of clusters k=3

# Create k=3 K-mean clusters



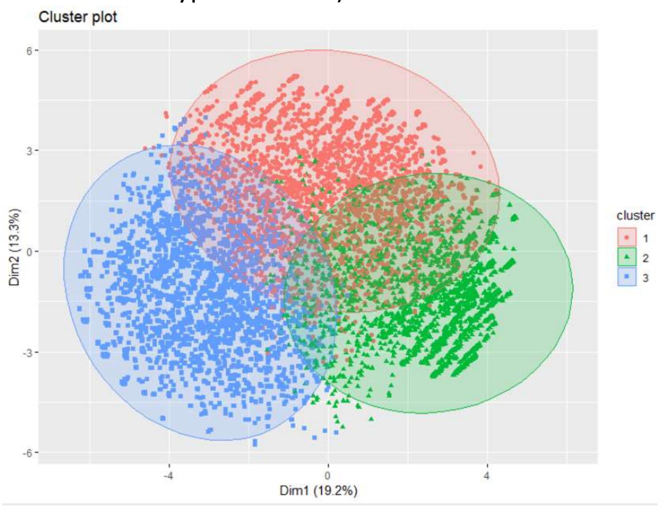
```
> #Create 3 clusters
> set.seed(2345)
> Clusters <- kmeans(data_scale, 3)
> Clusters$size
[1] 2324 2529 2179
> str(Clusters)
List of 9
 $ cluster     : Named int [1:7032] 2 3 2 1 2 2 2 3 2 3 ...
  ..- attr(*, "names")= chr [1:7032] "1" "2" "3" "4" ...
 $ centers     : num [1:3, 1:39] 0.00852 0.01769 -0.02962 -0.00852 -0.01769 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:3] "1" "2" "3"
  .. ..$ : chr [1:39] "gender.Female" "gender.Male" "SeniorCitizen" "Partner.No" ...
 $ totss       : num 274209
 $ withinss    : num [1:3] 84704 73881 54494
 $ tot.withinss: num 213079
 $ betweenss   : num 61130
 $ size        : int [1:3] 2324 2529 2179
 $ iter        : int 4
 $ ifault      : int 0
 - attr(*, "class")= chr "kmeans"
> Clusters$size
[1] 2324 2529 2179
```

```
fviz_cluster(Clusters, data = data_scale, geom = "point",
        stand = FALSE, frame.type = "norm")
```

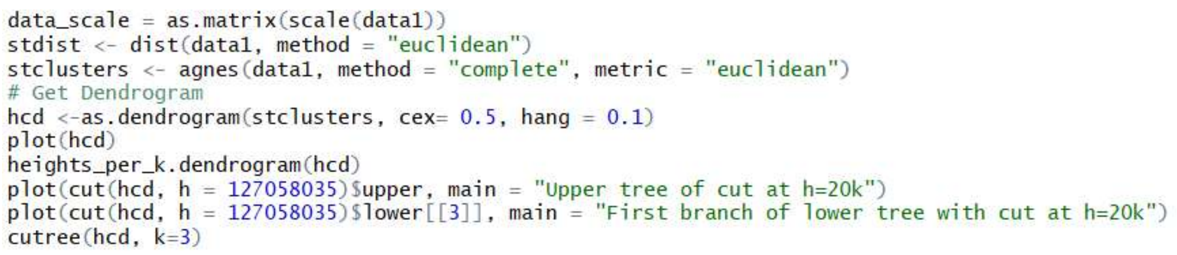


# PAM Clustering

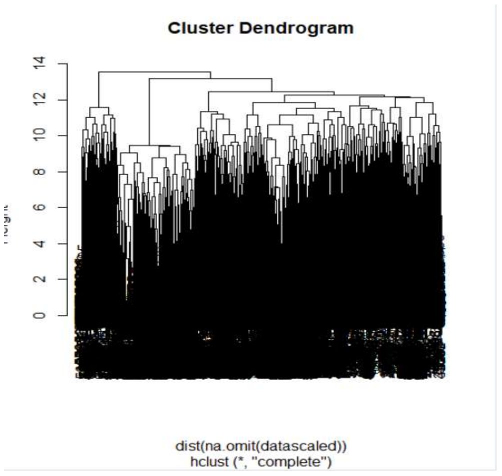
```
library("cluster")
pam.clusters<- pam(data_scale, 3)
# Visualize pam clusters
fviz_cluster(pam.clusters, stand = FALSE, geom = "point",
          frame.type = "norm")
```



# Create hierarchical cluster



```
data_scale = as.matrix(scale(data1))
stdist <- dist(data1, method = "euclidean")
stclusters <- agnes(data1, method = "complete", metric = "euclidean")
# Get Dendrogram
hcd <-as.dendrogram(stclusters, cex= 0.5, hang = 0.1)
plot(hcd)
heights_per_k.dendrogram(hcd)
plot(cut(hcd, h = 127058035)$upper, main = "Upper tree of cut at h=20k")
plot(cut(hcd, h = 127058035)$lower[[3]], main = "First branch of lower tree with cut at h=20k")
cutree(hcd, k=3)
```

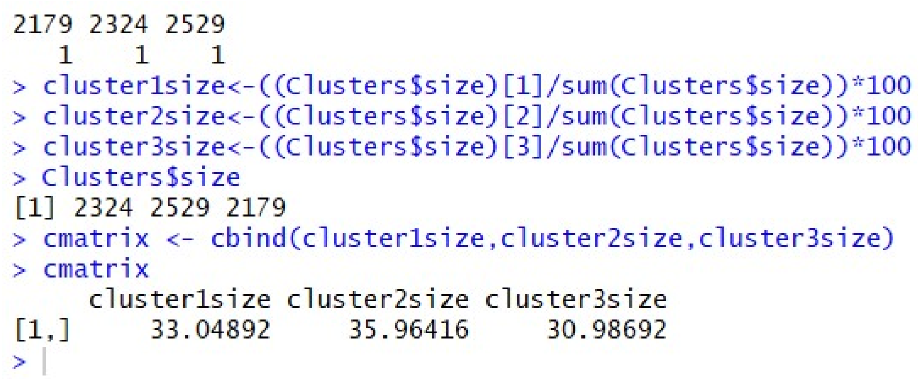


The hierarchical method does not require the number of clusters k as an input, but it needs a termination condition. Since our data is large, it took long time to run. The huge and messy cluster dendrogram is hard to understand and analyze. We know the hierarchical method does not work well on very large data sets and it works poorly with mixed data types. PAM (Partitioning Around Medoids) starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids. It works effectively for small data sets, but it does not scale well for large data sets (due to the computational complexity). Each method has its strength and weakness, it is not easy to say which one is better. But We can choose clustering method based on our data types and analyzing purpose. I choose kmeans method for the following question analysis.

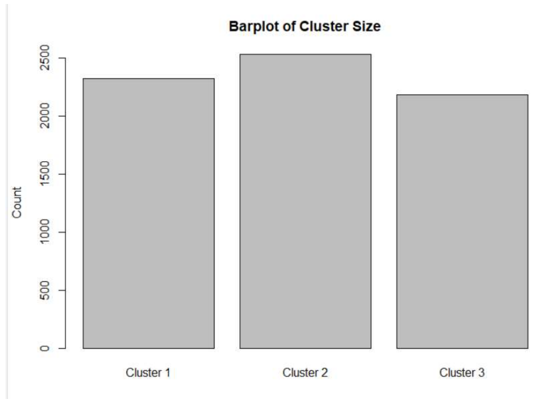
# Questions Analysis

Based on the above data analysis, I choose k=3 kmeans method for customer segments.

a. Segment size (percentage)

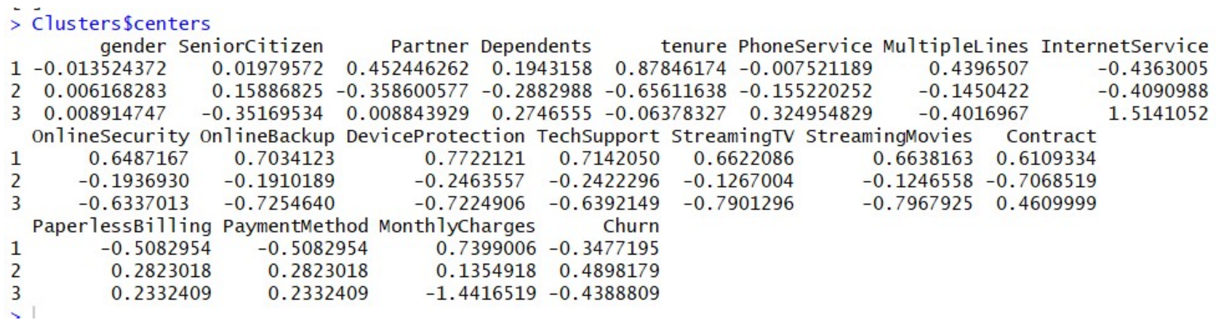


```
2179 2324 2529
   1    1    1
> cluster1size<-((Clusters$size)[1]/sum(Clusters$size))*100
> cluster2size<-((Clusters$size)[2]/sum(Clusters$size))*100
> cluster3size<-((Clusters$size)[3]/sum(Clusters$size))*100
> Clusters$size
[1] 2324 2529 2179
> cmatrix <- cbind(cluster1size,cluster2size,cluster3size)
> cmatrix
     cluster1size cluster2size cluster3size
[1,]     33.04892     35.96416     30.98692
> 
```



.

b. Segment characteristics (important features) and description (narrative)



```
> Clusters$centers
        gender SeniorCitizen      Partner Dependents      tenure PhoneService MultipleLines InternetService
1 -0.013524372    0.01979572  0.452446262  0.1943158  0.87846174 -0.007521189     0.4396507      -0.4363005
2  0.006168283    0.15886825 -0.358600577 -0.2882988 -0.65611638 -0.155220252    -0.1450422      -0.4090988
3  0.008914747   -0.35169534  0.008843929  0.2746555 -0.06378327  0.324954829    -0.4016967       1.5141052
  OnlineSecurity OnlineBackup DeviceProtection TechSupport StreamingTV StreamingMovies   Contract
1      0.6487167    0.7034123        0.7722121   0.7142050   0.6622086       0.6638163  0.6109334
2     -0.1936930   -0.1910189       -0.2463557  -0.2422296  -0.1267004      -0.1246558 -0.7068519
3     -0.6337013   -0.7254640       -0.7224906  -0.6392149  -0.7901296      -0.7967925  0.4609999
  PaperlessBilling PaymentMethod MonthlyCharges      Churn
1       -0.5082954    -0.5082954      0.7399006 -0.3477195
2        0.2823018     0.2823018      0.1354918  0.4898179
3        0.2332409     0.2332409     -1.4416519 -0.4388809
> 
```

clusters$withinss would tell us the sum of the square of the distance from each data point to the cluster center, one component per cluster. Lower is better. If withinss value is high, this may indicate either outliers are in data or you need to create more clusters.
Clusters$betweenss is sum of the squared distance between cluster centers.
Clusters$Centers is a matrix of cluster centers.

According to Clusters$centers analysis:

For cluster1, Partner(Yes), tenure(high), MultipleLines(Yes), StreamingTV(Yes), StreamingMovies(Yes), OnlineSecurity(Yes), OnlineBackup(Yes), Deviceprotection(Yes), paperlessBilling(No), paymentMethod(No), and TechSupport(Yes) are the important features, which these variables are high dissimilar to other clusters, and the rest of them are narrative attributes.

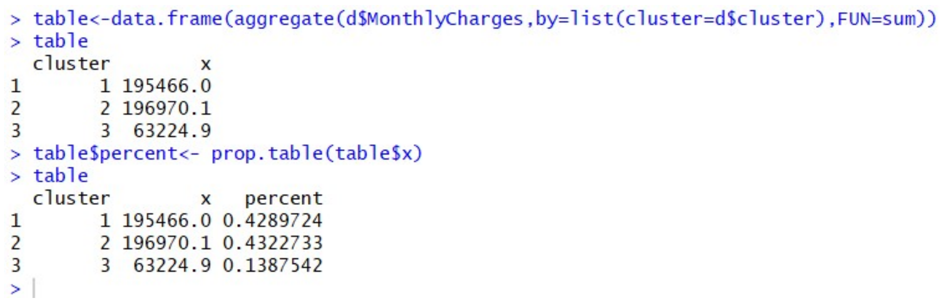
For cluster2, partner(no), Dependent(No) contract(No), Churn(Yes) are the important features, and the rest of them are narrative attributes.

For cluster3, Seniorcitizen(Yes), Internetservice(Yes), phoneService(yes), OnlineBackup(no), Monthlycharge(low) are the important features, and the rest of them are narrative attributes.

Compared to these three clusters' features, I am interested in cluster2. The churn value of cluster2 is highest, which means the highest leaving risk. We find the customer segment characteristics are "no partner", "no dependents", "low tenure", "no MultipleLine", "no internetService", "no onlineSecurity", "no contract". I suggest that company should focus on this customer segment for reducing churn risk.

c. Segment revenue share (% of revenue)

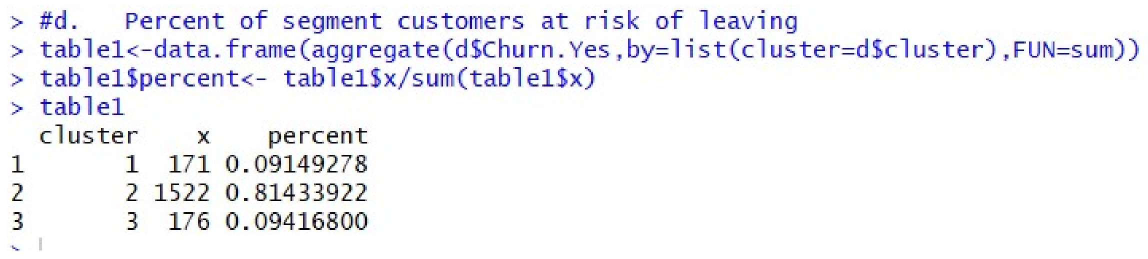
Since TotalCharges column has missing value, and it is correlated to Monthlycharges, the TotalCharges column has removed from original data. So, I use MonthlyCharges to calculate segment revenue share. The cluster3 has low revenue share since this customer segment has high Internet service.



```
> table<-data.frame(aggregate(d$MonthlyCharges,by=list(cluster=d$cluster),FUN=sum))
> table
  cluster        x
1       1 195466.0
2       2 196970.1
3       3  63224.9
> table$percent<- prop.table(table$x)
> table
  cluster        x   percent
1       1 195466.0 0.4289724
2       2 196970.1 0.4322733
3       3  63224.9 0.1387542
> 
```

d. Percent of segment customers at risk of leaving

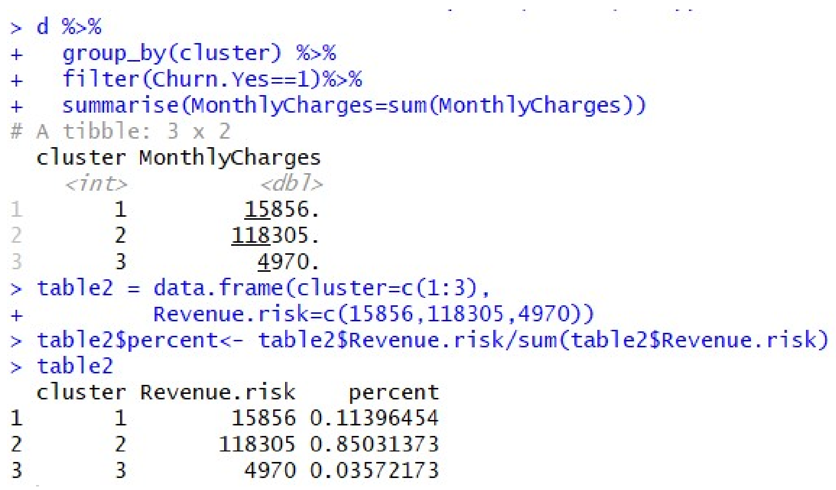
The cluster2 has the highest risk for churn(leaving). The result is same as question b.



```
> #d.   Percent of segment customers at risk of leaving
> table1<-data.frame(aggregate(d$Churn.Yes,by=list(cluster=d$cluster),FUN=sum))
> table1$percent<- table1$x/sum(table1$x)
> table1
  cluster    x    percent
1       1  171 0.09149278
2       2 1522 0.81433922
3       3  176 0.09416800
```

e. Revenue at risk per segment

The cluster2 has highest revenue at risk, 85% revenue at risk. The company should target at this customer segment for reducing the customer churn. We find this segment customers have short tenure without contract. So, company can make contract and increase tenure for decreasing leaving risk.

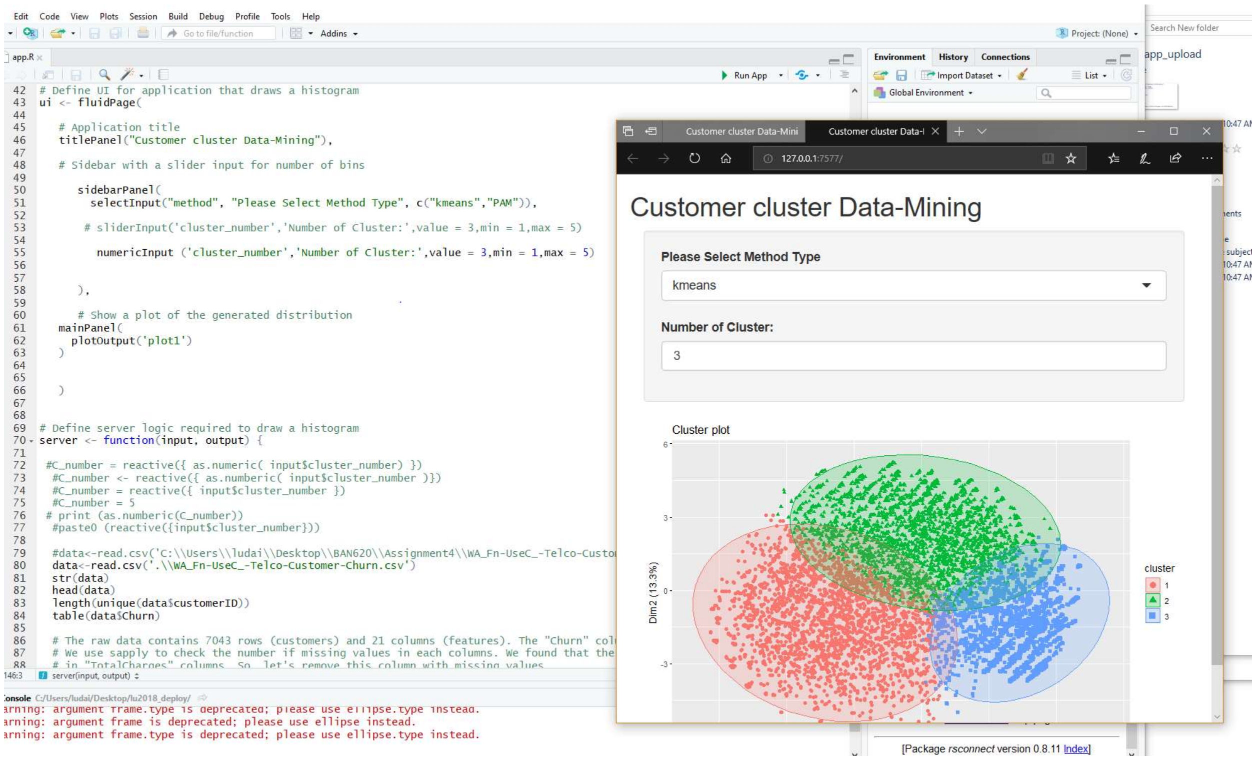


```
> d %>%
+   group_by(cluster) %>%
+   filter(Churn.Yes==1)%>%
+   summarise(MonthlyCharges=sum(MonthlyCharges))
# A tibble: 3 x 2
  cluster MonthlyCharges
    <int>          <dbl>
1       1         15856.
2       2        118305.
3       3          4970.
> table2 = data.frame(cluster=c(1:3),
+          Revenue.risk=c(15856,118305,4970))
> table2$percent<- table2$Revenue.risk/sum(table2$Revenue.risk)
> table2
  cluster Revenue.risk    percent
1       1        15856 0.11396454
2       2       118305 0.85031373
3       3         4970 0.03572173
```

Extra credit: ShinyApp

Code:

```r
# Define UI for application 
ui <- fluidPage(

   # Application title
   titlePanel("Customer cluster Data-Mining"),

   # Sidebar with a slider input for number of bins

      sidebarPanel(
        selectInput("method", "Please Select Method Type", c("kmeans","PAM")),

       # sliderInput('cluster_number','Number of Cluster:',value = 3,min = 1,max = 5)

         numericInput ('cluster_number','Number of Cluster:',value = 3,min = 1,max = 5)


      ),

      # Show a plot of the generated distribution
   mainPanel(
     plotOutput('plot1')
   )

   )


# Define server logic required to draw a histogram
server <- function(input, output) {

 #C_number = reactive({ as.numeric( input$cluster_number) })
  #C_number <- reactive({ as.numberic( input$cluster_number )})
  #C_number = reactive({ input$cluster_number })
  #C_number = 5
 # print (as.numberic(C_number))
  #paste0 (reactive({input$cluster_number}))

  #data<-read.csv('C:\\Users\\ludai\\Desktop\\BAN620\\Assignment4\\WA_Fn-UseC_-Telco-Customer-Churn.csv')
  data<-read.csv('.\\WA_Fn-UseC_-Telco-Customer-Churn.csv')
  str(data)
  head(data)
  length(unique(data$customerID))
  table(data$Churn)

  # The raw data contains 7043 rows (customers) and 21 columns (features). The "Churn" column is our target.
  # We use sapply to check the number if missing values in each columns. We found that there are 11 missing values
  # in "TotalCharges" columns. So, let's remove this column with missing values.
  sapply(data, function(x) sum(is.na(x)))
  data <- data[complete.cases(data), ]
  summary(data)
```



```r
# Define UI for application that draws a histogram
ui <- fluidPage(

   # Application title
   titlePanel("Customer cluster Data-Mining"),

   # Sidebar with a slider input for number of bins

      sidebarPanel(
        selectInput("method", "Please Select Method Type", c("kmeans","PAM")),

       # sliderInput('cluster_number','Number of Cluster:',value = 3,min = 1,max = 5)

         numericInput ('cluster_number','Number of Cluster:',value = 3,min = 1,max = 5)


      ),

      # Show a plot of the generated distribution
   mainPanel(
     plotOutput('plot1')
   )


   )


# Define server logic required to draw a histogram
server <- function(input, output) {

 #C_number = reactive({ as.numeric( input$cluster_number) })
  #C_number <- reactive({ as.numberic( input$cluster_number )})
  #C_number = reactive({ input$cluster_number })
  #C_number = 5
 # print (as.numberic(C_number))
  #paste0 (reactive({input$cluster_number}))

  #data<-read.csv('C:\\Users\\ludai\\Desktop\\BAN620\\Assignment4\\WA_Fn-UseC_-Telco-Custo
  data<-read.csv('.\\WA_Fn-UseC_-Telco-Customer-Churn.csv')
  str(data)
  head(data)
  length(unique(data$customerID))
  table(data$Churn)

  # The raw data contains 7043 rows (customers) and 21 columns (features). The "Churn" col
  # We use sapply to check the number if missing values in each columns. We found that the
  # in "TotalCharges" columns. So, let's remove this column with missing values
146:3   server(input, output)
```

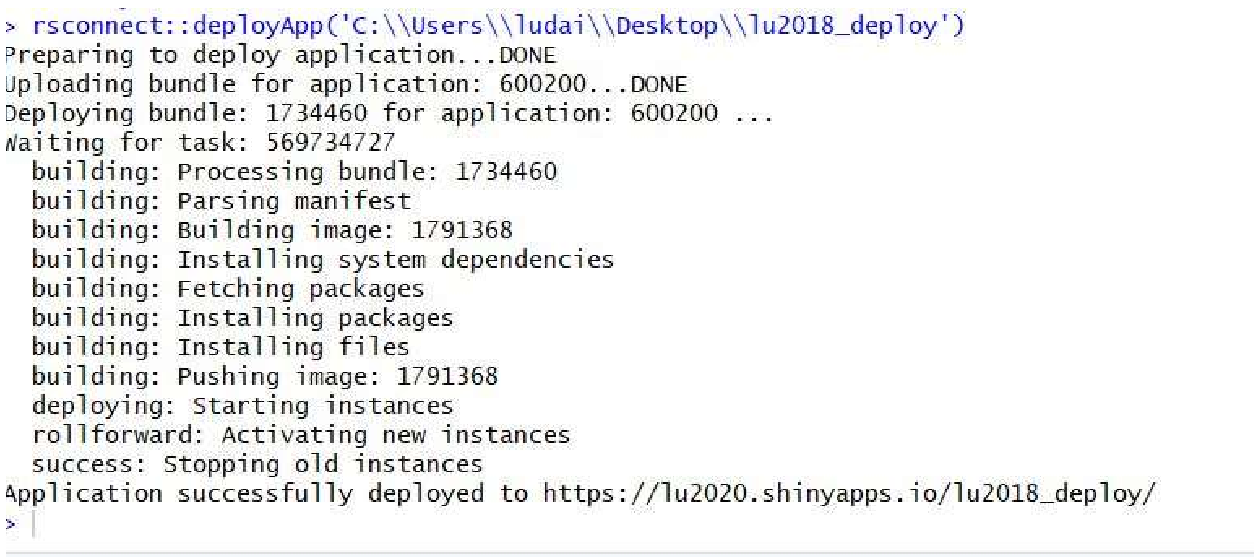
Customer cluster Data-Mining

Please Select Method Type: kmeans

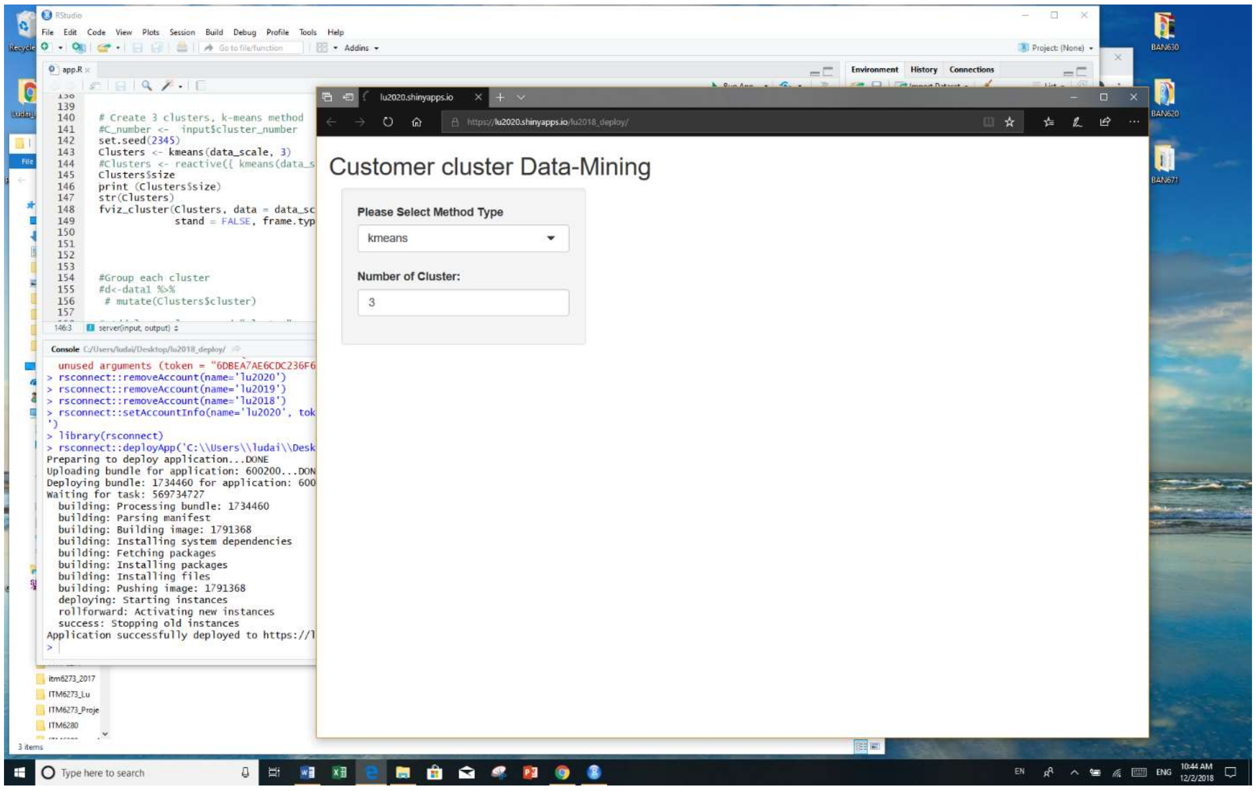
Number of Cluster: 3

```
Console C:/Users/ludai/Desktop/lu2018_deploy/
arning: argument frame.type is deprecated; please use ellipse.type instead.
arning: argument frame is deprecated; please use ellipse instead.
arning: argument frame.type is deprecated; please use ellipse.type instead.
```

# ShinyApp upload to Web



```
> rsconnect::deployApp('C:\\Users\\ludai\\Desktop\\lu2018_deploy')
Preparing to deploy application...DONE
Uploading bundle for application: 600200...DONE
Deploying bundle: 1734460 for application: 600200 ...
Waiting for task: 569734727
  building: Processing bundle: 1734460
  building: Parsing manifest
  building: Building image: 1791368
  building: Installing system dependencies
  building: Fetching packages
  building: Installing packages
  building: Installing files
  building: Pushing image: 1791368
  deploying: Starting instances
  rollforward: Activating new instances
  success: Stopping old instances
Application successfully deployed to https://lu2020.shinyapps.io/lu2018_deploy/
> |
```



```
    # Create 3 clusters, k-means method
    #C_number <-  input$cluster_number
    set.seed(2345)
    Clusters <- kmeans(data_scale, 3)
    #Clusters <- reactive({ kmeans(data_s
    Clusters$size
    print (Clusters$size)
    str(Clusters)
    fviz_cluster(Clusters, data = data_sc
                 stand = FALSE, frame.typ




    #Group each cluster
    #d<-data1 %>%
     # mutate(Clusters$cluster)
```

```
  unused arguments (token = "6DBEA7AE6CDC236F6
> rsconnect::removeAccount(name='lu2020')
> rsconnect::removeAccount(name='lu2019')
> rsconnect::removeAccount(name='lu2018')
> rsconnect::setAccountInfo(name='lu2020', tok
')
> library(rsconnect)
> rsconnect::deployApp('C:\\Users\\ludai\\Desk
Preparing to deploy application...DONE
Uploading bundle for application: 600200...DON
Deploying bundle: 1734460 for application: 600
Waiting for task: 569734727
  building: Processing bundle: 1734460
  building: Parsing manifest
  building: Building image: 1791368
  building: Installing system dependencies
  building: Fetching packages
  building: Installing packages
  building: Installing files
  building: Pushing image: 1791368
  deploying: Starting instances
  rollforward: Activating new instances
  success: Stopping old instances
Application successfully deployed to https://l
> |
```

Customer cluster Data-Mining

Please Select Method Type: kmeans

Number of Cluster: 3