Notebook3

Jered Hightower, Hanniyah Hammid, Sai Gonuguntla

10/8/2022

https://www.kaggle.com/datasets/gauravtopre/bank-customer-churn-dataset

Import our Data Set

```
df <- read.csv("churn.csv")

str(df)
```

```
## 'data.frame':    10000 obs. of  12 variables:
##  $ customer_id     : int  15634602 15647311 15619304 15701354 15737888 15574012 15592531 15656148 15
##  $ credit_score    : int  619 608 502 699 850 645 822 376 501 684 ...
##  $ country         : chr  "France" "Spain" "France" "France" ...
##  $ gender          : chr  "Female" "Female" "Female" "Female" ...
##  $ age             : int  42 41 42 39 43 44 50 29 44 27 ...
##  $ tenure          : int  2 1 8 1 2 8 7 4 4 2 ...
##  $ balance         : num  0 83808 159661 0 125511 ...
##  $ products_number : int  1 1 3 2 1 2 2 4 2 1 ...
##  $ credit_card     : int  1 0 1 0 1 1 1 1 0 1 ...
##  $ active_member   : int  1 1 0 0 1 0 1 0 1 1 ...
##  $ estimated_salary: num  101349 112543 113932 93827 79084 ...
##  $ churn           : int  1 0 1 0 0 1 0 1 0 0 ...
```

Data Cleanup

Make all attributes numeric

```
df$country<- factor(df$country)
df$gender <- factor(df$gender)


# In Years
df$age <- as.numeric(df$age)
df$tenure <- as.numeric(df$tenure)

# Number of Products
df$products_number <- as.numeric(df$products_number)

str(df)
```

```
## 'data.frame':    10000 obs. of  12 variables:
##  $ customer_id     : int  15634602 15647311 15619304 15701354 15737888 15574012 15592531 15656148 15
##  $ credit_score    : int  619 608 502 699 850 645 822 376 501 684 ...
##  $ country         : Factor w/ 3 levels "France","Germany",..: 1 3 1 1 3 3 1 2 1 1 ...
##  $ gender          : Factor w/ 2 levels "Female","Male": 1 1 1 1 1 2 2 1 2 2 ...
```

```
##  $ age             : num  42 41 42 39 43 44 50 29 44 27 ...
##  $ tenure          : num  2 1 8 1 2 8 7 4 4 2 ...
##  $ balance         : num  0 83808 159661 0 125511 ...
##  $ products_number : num  1 1 3 2 1 2 2 4 2 1 ...
##  $ credit_card     : int  1 0 1 0 1 1 1 1 0 1 ...
##  $ active_member   : int  1 1 0 0 1 0 1 0 1 1 ...
##  $ estimated_salary: num  101349 112543 113932 93827 79084 ...
##  $ churn           : int  1 0 1 0 0 1 0 1 0 0 ...
```

Standardize variables

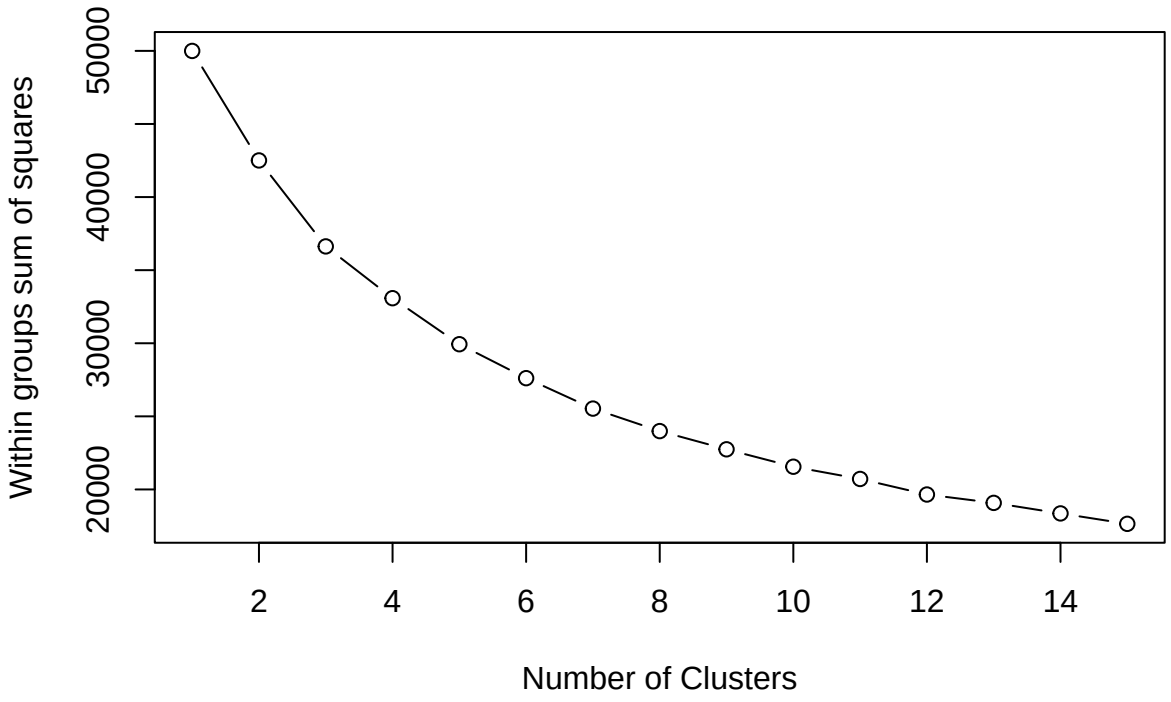
```
original <- df

df <- df[,-c(1, 3, 4, 8, 9, 10, 12)]
str(df)
```

```
## 'data.frame':    10000 obs. of  5 variables:
##  $ credit_score    : int  619 608 502 699 850 645 822 376 501 684 ...
##  $ age             : num  42 41 42 39 43 44 50 29 44 27 ...
##  $ tenure          : num  2 1 8 1 2 8 7 4 4 2 ...
##  $ balance         : num  0 83808 159661 0 125511 ...
##  $ estimated_salary: num  101349 112543 113932 93827 79084 ...
```

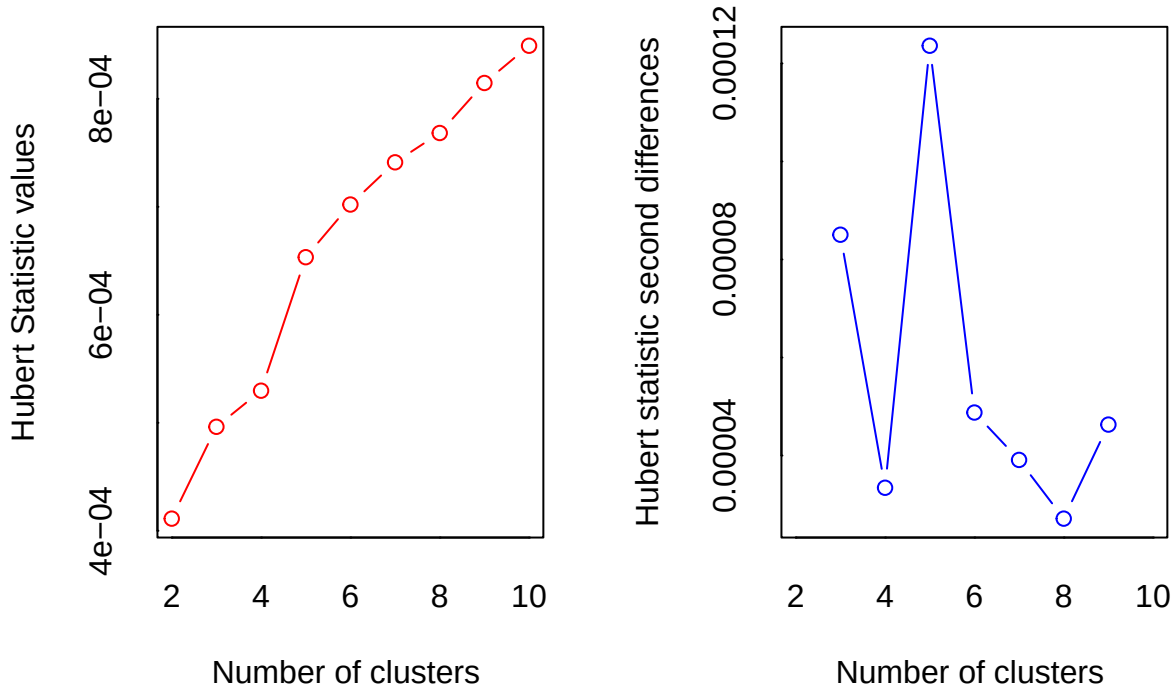
```
df <- scale(df)
```

Determine number of clusters

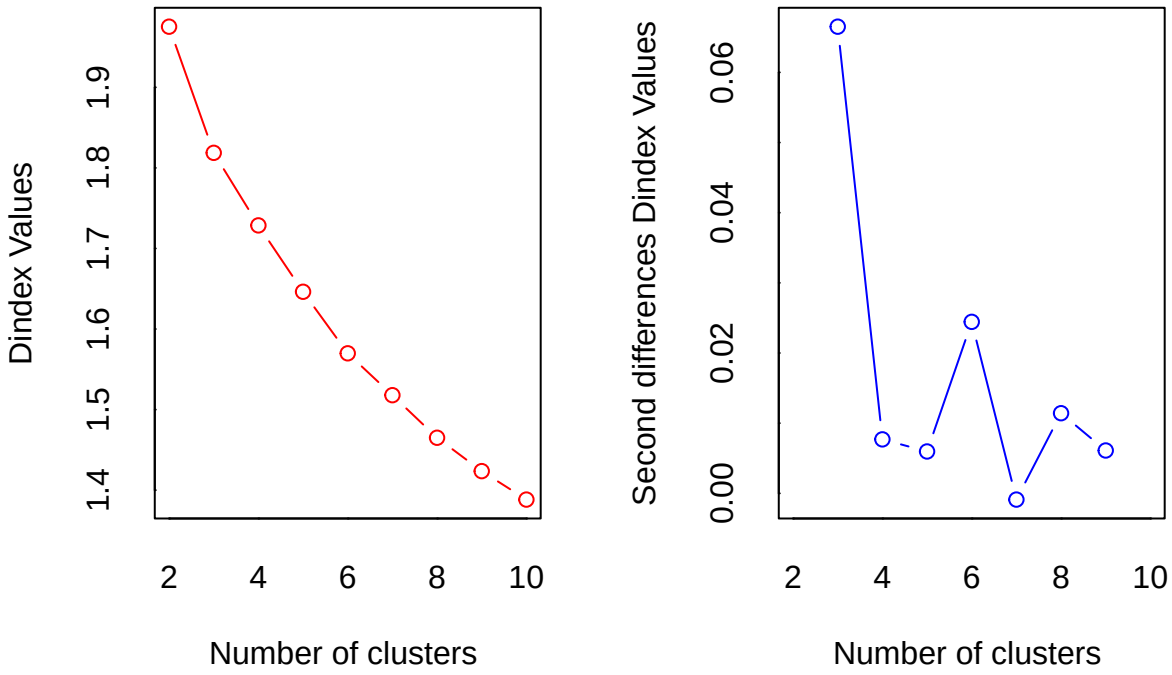
```
set.seed(1234)

wss <- (nrow(df)-1)*sum(apply(df,2,var))
for (i in 2:15) wss[i] <- sum(kmeans(df, centers=i, iter.max = 20)$withinss)
plot(1:15, wss, type="b", xlab="Number of Clusters", ylab="Within groups sum of squares")
```



```r
# Find optimal cluster amount
i <- sample(1:nrow(df), nrow(df)*.1, replace = FALSE)
sample <- df[i,]

library(NbClust)
nc <- NbClust(sample, min.nc=2, max.nc=10, method="kmeans")
```




```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 index second differences plot.
##
```




```
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Dindex
##                 second differences plot) that corresponds to a significant increase of the value of
##                 the measure.
##
## *******************************************************************
## * Among all indices:
## * 7 proposed 2 as the best number of clusters
## * 7 proposed 3 as the best number of clusters
## * 1 proposed 5 as the best number of clusters
## * 3 proposed 6 as the best number of clusters
## * 2 proposed 8 as the best number of clusters
## * 2 proposed 9 as the best number of clusters
## * 1 proposed 10 as the best number of clusters
##
##                     ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  2
##
##
## *******************************************************************
```

K-Means Cluster Analysis

```
set.seed(1234)
fit <- kmeans(df, 2) # 2 cluster solution

# get cluster means
aggregate(df,by=list(cluster=fit$cluster), mean)
```

```
##   cluster credit_score         age     tenure     balance estimated_salary
## 1       1 -0.007584739 -0.01582091  0.8773875 -0.10382860      0.009450213
## 2       2  0.007272733  0.01517010 -0.8412951  0.09955749     -0.009061468
```

```
# append cluster assignment
df <- data.frame(df, fit$cluster)
```

Data Analysis

```
gt <- table(original$gender, fit$cluster)
gt
```

```
##
##             1    2
##   Female 2203 2340
##   Male   2692 2765
```

```
ct <- table(original$credit_card, fit$cluster)
ct
```

```
##
##        1    2
##   0 1406 1539
##   1 3489 3566
```

```
at <- table(original$active_member, fit$cluster)
at
```

```
##
##        1    2
##   0 2428 2421
##   1 2467 2684
```

```
cht <- table(original$churn, fit$cluster)
cht
```

```
##
##        1    2
##   0 3955 4008
##   1  940 1097
```

```
# No clear correlation to any of our factors
```

Hierarchial Clustering

```
library(flexclust)
```

```
## Loading required package: grid
```

```
## Loading required package: lattice
```

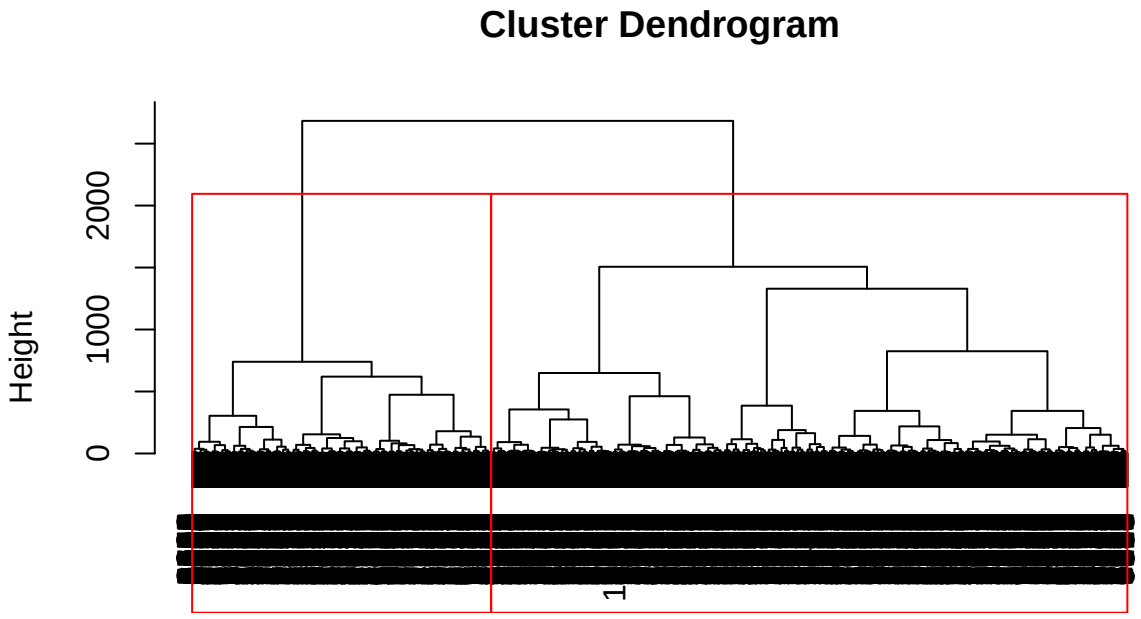
```
## Loading required package: modeltools
```

```
## Loading required package: stats4
```

```
# Subset Data to make some sense
data <- original[,-c(1, 3, 4, 8, 9 , 10, 12)]

data <- scale(data)

# Ward Hierarchical Clustering
d <- dist(data, method = "euclidean") # distance matrix
fit <- hclust(d, method="ward")
```

```
## The "ward" method has been renamed to "ward.D"; note new "ward.D2"
```

```
plot(fit) # display dendogram

groups <- cutree(fit, k=2) # cut tree into 5 clusters
# draw dendogram with red borders around the 5 clusters
rect.hclust(fit, k=2, border="red")
```




```
for(c in 2:16){
  cluster_cut <- cutree(fit, c)
  table_cut <- table(cluster_cut, original$churn)
  print(table_cut)

  ri <- randIndex(table_cut)
  print(paste("cut=", c, "Rand index = ", ri))
}
```

```
##
## cluster_cut    0    1
##           1 2835  362
##           2 5128 1675
## [1] "cut= 2 Rand index =  -0.0378469153422799"
##
## cluster_cut    0    1
##           1 2835  362
##           2 1995  490
##           3 3133 1185
## [1] "cut= 3 Rand index =  -0.00243131063132192"
```

```
##
## cluster_cut    0    1
##           1 2835  362
##           2 1995  490
##           3 2450  746
##           4  683  439
## [1] "cut= 4 Rand index =  0.0268703146703495"
##
## cluster_cut    0    1
##           1 2835  362
##           2 1995  490
##           3 1092  370
##           4  683  439
##           5 1358  376
## [1] "cut= 5 Rand index =  0.0321494880039421"
##
## cluster_cut    0    1
##           1 1875  261
##           2 1995  490
##           3 1092  370
##           4  683  439
##           5  960  101
##           6 1358  376
## [1] "cut= 6 Rand index =  0.0119891265800136"
##
## cluster_cut    0    1
##           1 1875  261
##           2  984  312
##           3 1092  370
##           4 1011  178
##           5  683  439
##           6  960  101
##           7 1358  376
## [1] "cut= 7 Rand index =  0.0111784019731331"
##
## cluster_cut    0    1
##           1  750  122
##           2  984  312
##           3 1092  370
##           4 1125  139
##           5 1011  178
##           6  683  439
##           7  960  101
##           8 1358  376
## [1] "cut= 8 Rand index =  0.00366863860524379"
##
## cluster_cut    0    1
##           1  750  122
##           2  984  312
##           3 1092  370
##           4  516   53
##           5 1011  178
##           6  683  439
##           7  609   86
```

```
##           8  960  101
##           9 1358  376
## [1] "cut= 9 Rand index =  0.000387266979210034"
##
## cluster_cut    0    1
##          1   750  122
##          2   984  312
##          3  1092  370
##          4   516   53
##          5   554  100
##          6   683  439
##          7   457   78
##          8   609   86
##          9   960  101
##          10 1358  376
## [1] "cut= 10 Rand index =  -0.00116390333642853"
##
## cluster_cut    0    1
##          1   750  122
##          2   984  312
##          3  1092  370
##          4   516   53
##          5   554  100
##          6   304  157
##          7   457   78
##          8   609   86
##          9   960  101
##          10  379  282
##          11 1358  376
## [1] "cut= 11 Rand index =  0.00175004477807289"
##
## cluster_cut    0    1
##          1   750  122
##          2   553  257
##          3  1092  370
##          4   516   53
##          5   554  100
##          6   304  157
##          7   457   78
##          8   609   86
##          9   960  101
##          10  379  282
##          11 1358  376
##          12  431   55
## [1] "cut= 12 Rand index =  0.00258368380404147"
##
## cluster_cut    0    1
##          1   750  122
##          2   553  257
##          3  1092  370
##          4   516   53
##          5   554  100
##          6   304  157
##          7   457   78
```

```
##          8   609   86
##          9   960  101
##          10  379  282
##          11  587  198
##          12  431   55
##          13  771  178
## [1] "cut= 13 Rand index =  0.00352002242850073"
##
## cluster_cut   0   1
##          1  750 122
##          2  553 257
##          3  598 251
##          4  516  53
##          5  554 100
##          6  304 157
##          7  494 119
##          8  457  78
##          9  609  86
##          10 960 101
##          11 379 282
##          12 587 198
##          13 431  55
##          14 771 178
## [1] "cut= 14 Rand index =  0.00512139745539474"
##
## cluster_cut   0   1
##          1  750 122
##          2  553 257
##          3  598 251
##          4  516  53
##          5  554 100
##          6  304 157
##          7  494 119
##          8  457  78
##          9  609  86
##          10 392  34
##          11 379 282
##          12 568  67
##          13 587 198
##          14 431  55
##          15 771 178
## [1] "cut= 15 Rand index =  0.00251364079758401"
##
## cluster_cut   0   1
##          1  750 122
##          2  253 202
##          3  598 251
##          4  516  53
##          5  554 100
##          6  304 157
##          7  494 119
##          8  300  55
##          9  457  78
##          10 609  86
```

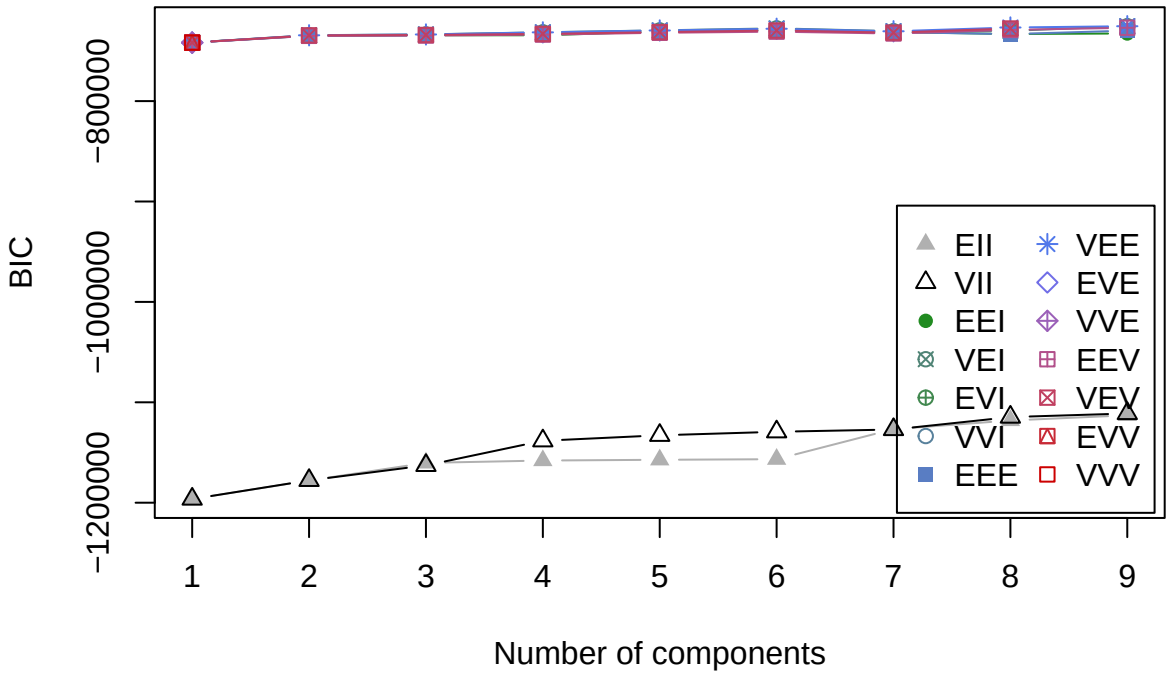
```
##           11 392  34
##           12 379 282
##           13 568  67
##           14 587 198
##           15 431  55
##           16 771 178
## [1] "cut= 16 Rand index =  0.00386563750919265"
```

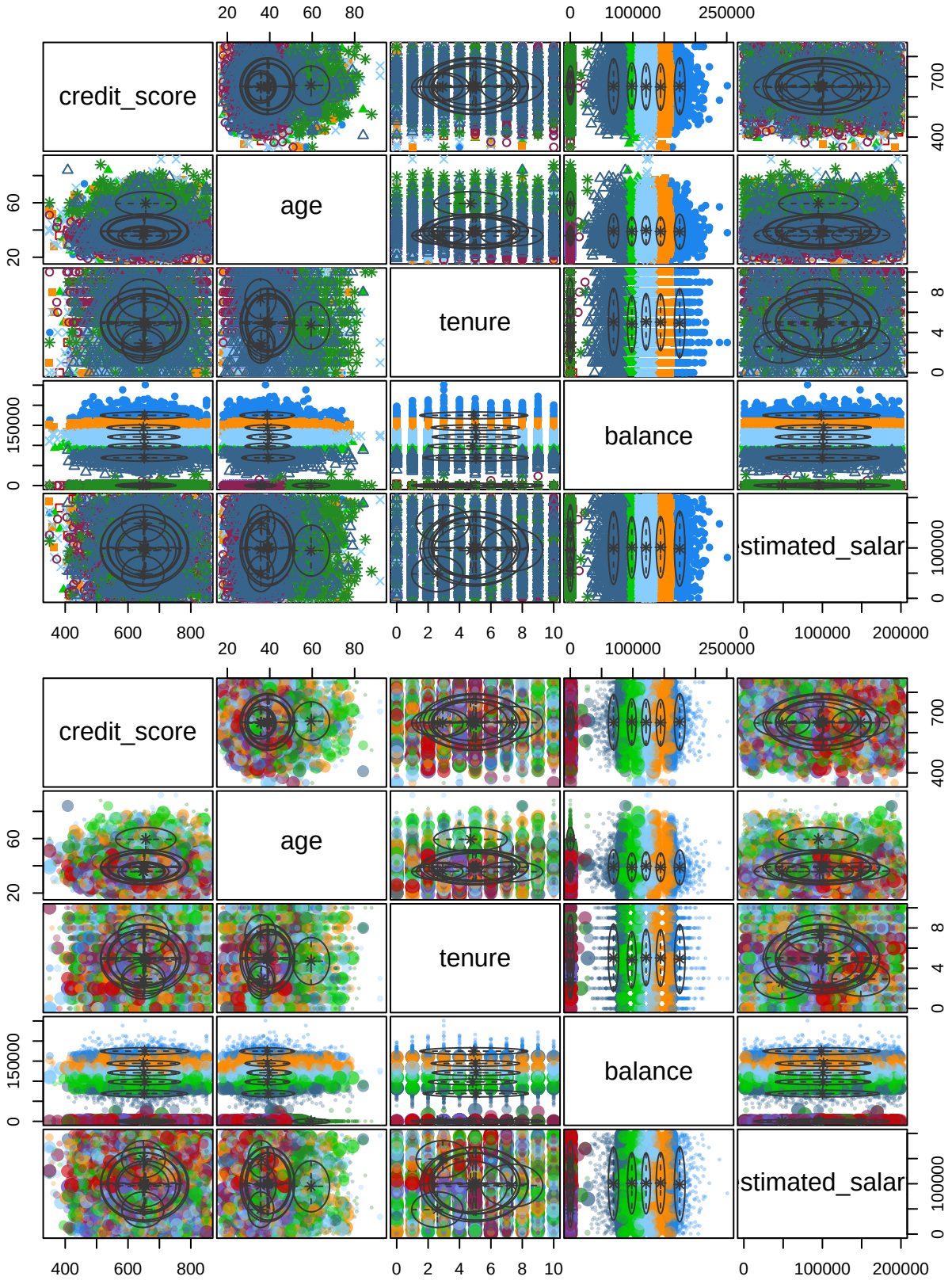
Model Based Clustering

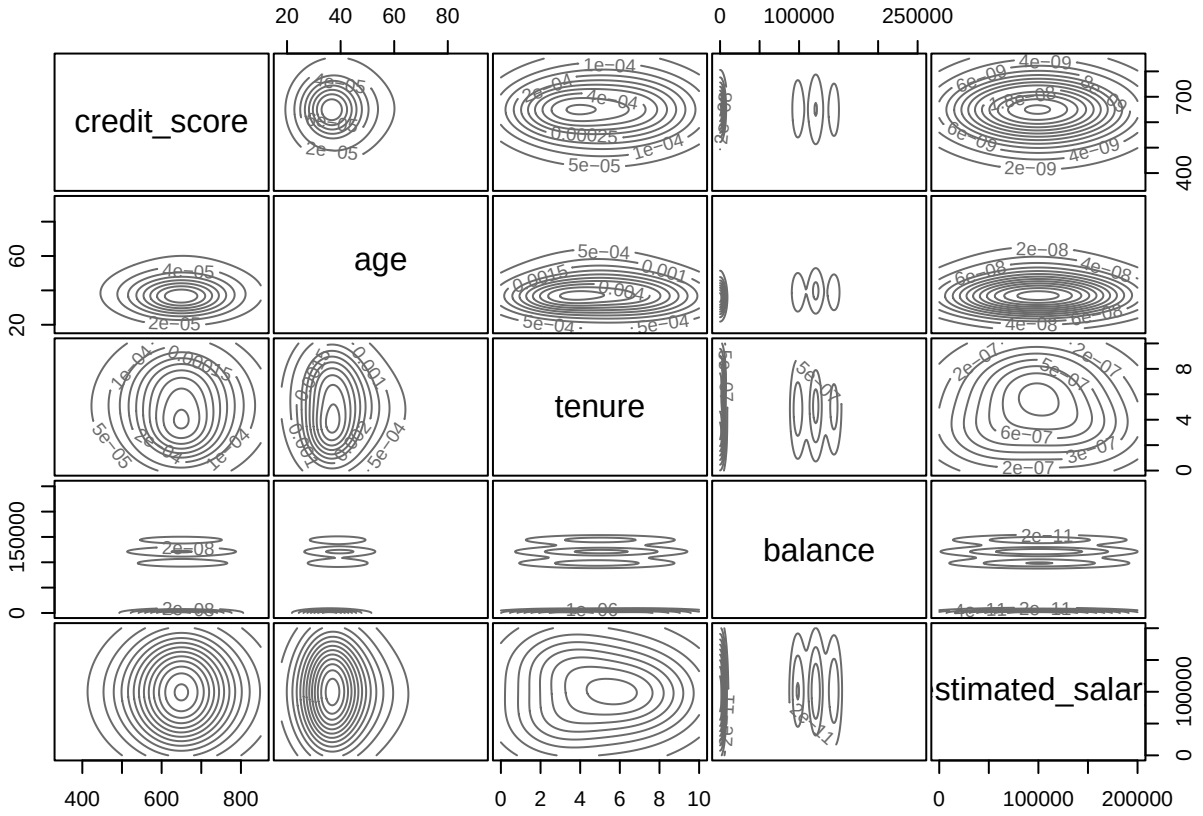
```
data <- original[,-c(1, 3, 4, 8, 9 , 10, 12)]

library(mclust)
```

```
## Package 'mclust' version 5.4.10
## Type 'citation("mclust")' for citing this R package in publications.
```

```
fit <- Mclust(data)
plot(fit) # plot results
```




credit_score
age
tenure
balance
stimated_salar
20 40 60 80
0 100000 250000
400 600 800
0 2 4 6 8 10
0 100000 200000
credit_score
age
tenure
balance
stimated_salar
20 40 60 80
0 100000 250000
400 600 800
0 2 4 6 8 10
0 100000 200000




```
summary(fit) # display the best model
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VEI (diagonal, equal shape) model with 9 components:
##
##  log-likelihood     n df       BIC       ICL
##       -362370.4 10000 66 -725348.8 -728158.8
##
## Clustering table:
##    1    2    3    4    5    6    7    8    9
##  571  837 1620  721 1487 2014 1684  383  683
```

Results of the Algorithms and Insights

Overall, the algorithms showed that this dataset was not well represented by clustering.

When doing kMeans, our graph showed no clear elbow to give any idea how many clusters should be used. There were no significant knees in the Hubert index and D index graphs either. The clusters showed no significant correlation with any of the factors I excluded.

Hierarchical clustering faced similar issues since the rand index showed that no matter the clustering amount, nothing significantly correlated with churn. It was about as good as random guessing.

The Model-Based clustering suggested 7 clustering groups an was still overall. The extremely low BIC still suggested that this wasn't a good model.

In conclusion, a different model could likely represent this dataset better.