# Assignment 2

*Suryoday T. Roy*

*January 14, 2018*

```
library("caret", lib.loc="~/R/win-library/3.4")
```

```
## Warning: package 'caret' was built under R version 3.4.3
```

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

```
## Loading required package: ggplot2
```

```
dataset<- data("GermanCredit")
head(GermanCredit)
```

## Q1. Selecting the numeric variables that are appropriate:

The data has 7 numeric variables: Duration in Months,
Credit Amount,
Installment as % of disposable income,
Present residence since (years),
Age (years),
Number of existing credits in this bank,
Number of people being liable to provide maintenance for

Each of these will potentaily add value to the clusters and thus should be retained for the clustering algorithm.

## Q2.& Q3. Use K-means

### Divide data into train and test, Extract and Scale all numeric variables

```
ind <- sample(c(TRUE, FALSE), nrow(GermanCredit), replace=TRUE, prob=c(0.632, 0.36
8))
data1 <- GermanCredit[ind, ]
dataTrain<-scale(data1[1:7])
data2<- GermanCredit[!ind, ]
dataTest<-scale(data2[1:7])
```

### Using Seed (1001)

```
set.seed(1001)
R2<-numeric(9)
members<-matrix(0,10,11)
members[,1]<-1:10
for (i in 1:10){
  Clust=kmeans(dataTrain,centers=i,nstart=50)
  R2[i]=round(Clust$betweenss/Clust$totss *100,2)
  for(j in 1:length(Clust$size)){members[i,j+1]<-Clust$size[j]}
}
Ans<-rbind(2:10,R2[-1])
rownames(Ans)<-c("No. of Clusters","R2")
Ans
```

```
##                  [,1]  [,2]  [,3] [,4]  [,5] [,6]  [,7]  [,8]  [,9]
## No. of Clusters  2.00  3.00  4.00  5.0  6.00  7.0  8.00  9.00 10.00
## R2              16.47 28.88 39.27 45.7 50.68 54.1 56.63 58.84 60.94
```

```
cat("\nCluster Size & No. of People in each cluster\n")
```

```
##
## Cluster Size & No. of People in each cluster
```

```
members
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
##  [1,]    1  631    0    0    0    0    0    0    0     0     0
##  [2,]    2  477  154    0    0    0    0    0    0     0     0
##  [3,]    3   91  129  411    0    0    0    0    0     0     0
##  [4,]    4   88  160  292   91    0    0    0    0     0     0
##  [5,]    5   82  216  127   91  115    0    0    0     0     0
##  [6,]    6  166  104   72  121   78   90    0    0     0     0
##  [7,]    7   97  121   99   90   65   82   77    0     0     0
##  [8,]    8  103   90   72   92   71   56   69   78     0     0
##  [9,]    9   72   69   31   59   56  103   78   92    71     0
## [10,]   10   65   65   42   67   63   96   23   38    82    90
```

## Using Seed (1)

```
set.seed(1)
R2<-numeric(9)
members<-matrix(0,10,11)
members[,1]<-1:10
for (i in 1:10){
  Clust=kmeans(dataTrain,centers=i,nstart=50)
  R2[i]=round(Clust$betweenss/Clust$totss *100,2)
  for(j in 1:length(Clust$size)){members[i,j+1]<-Clust$size[j]}
}
Ans<-rbind(2:10,R2[-1])
rownames(Ans)<-c("No. of Clusters","R2")
Ans
```

```
##                   [,1]  [,2]  [,3] [,4]  [,5] [,6]  [,7]  [,8] [,9]
## No. of Clusters  2.00  3.00  4.00  5.0  6.00  7.0  8.00  9.00   10
## R2              16.47 28.88 39.27 45.7 50.68 54.1 56.62 59.09   61
```

```
cat("\nCluster Size & No. of People in each cluster\n")
```

```
## 
## Cluster Size & No. of People in each cluster
```

```
members
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
##  [1,]    1  631    0    0    0    0    0    0    0     0     0
##  [2,]    2  154  477    0    0    0    0    0    0     0     0
##  [3,]    3  411  129   91    0    0    0    0    0     0     0
##  [4,]    4  292  160   88   91    0    0    0    0     0     0
##  [5,]    5  127   91   82  113  218    0    0    0     0     0
##  [6,]    6  166  121   72   78  104   90    0    0     0     0
##  [7,]    7   99   97  121   65   82   90   77    0     0     0
##  [8,]    8   90   65  105   68   56   91   74   82     0     0
##  [9,]    9   69  103   71   92   67   78   72   23    56     0
## [10,]   10   81   60   40   72   62   65   87   51    91    22
```

## Using Seed (30345678)

```
set.seed(30345678)
R2<-numeric(9)
members<-matrix(0,10,11)
members[,1]<-1:10
for (i in 1:10){
  Clust=kmeans(dataTrain,centers=i,nstart=50)
  R2[i]=round(Clust$betweenss/Clust$totss *100,2)
  for(j in 1:length(Clust$size)){members[i,j+1]<-Clust$size[j]}
}
Ans<-rbind(1:10,R2)
rownames(Ans)<-c("No. of Clusters","R2")
Ans
```

```
##                  [,1]  [,2]  [,3]  [,4] [,5]  [,6] [,7]  [,8]  [,9] [,10]
## No. of Clusters     1  2.00  3.00  4.00  5.0  6.00  7.0  8.00  9.00 10.00
## R2                  0 16.47 28.88 39.27 45.7 50.68 54.1 56.63 59.07 60.94
```

```
cat("\nCluster Size & No. of People in each cluster\n")
```

```
## 
## Cluster Size & No. of People in each cluster
```

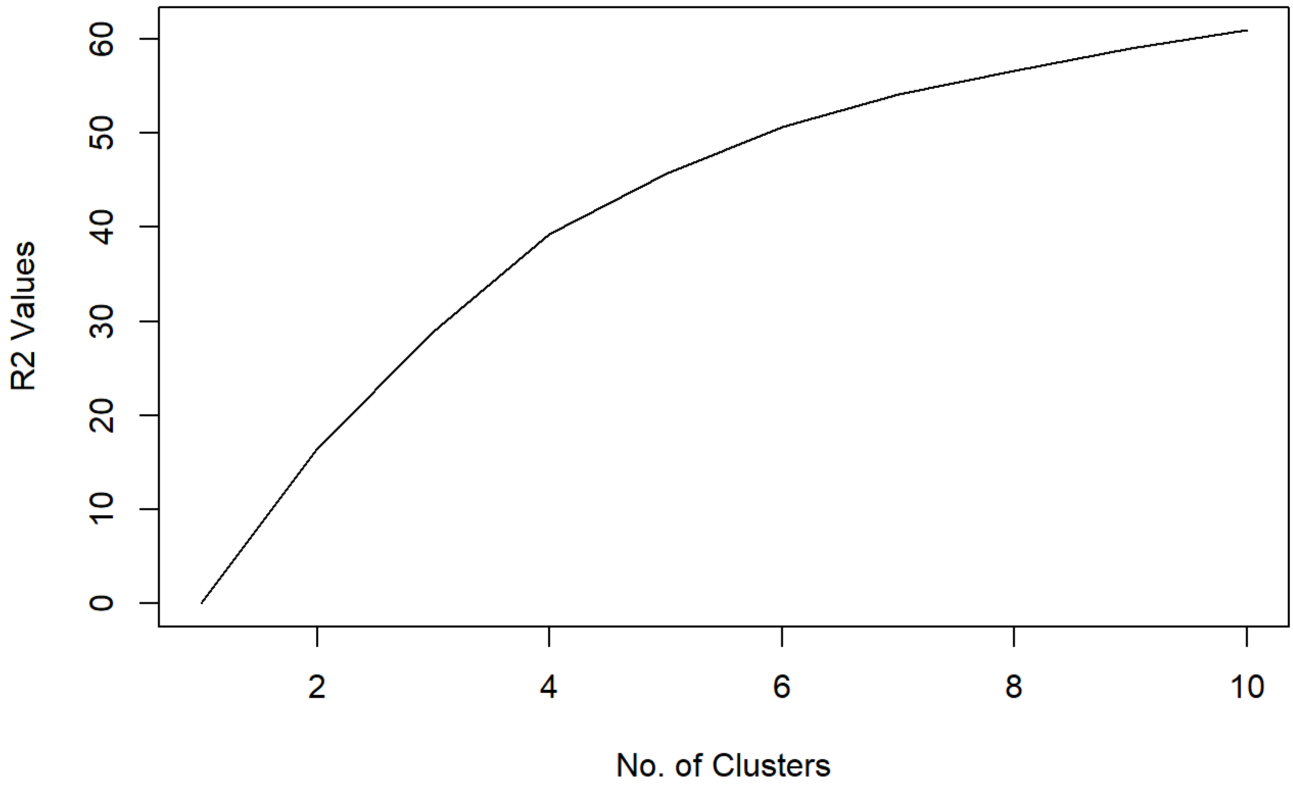
```
members
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
##  [1,]    1  631    0    0    0    0    0    0    0     0     0
##  [2,]    2  477  154    0    0    0    0    0    0     0     0
##  [3,]    3   91  129  411    0    0    0    0    0     0     0
##  [4,]    4  292   88   91  160    0    0    0    0     0     0
##  [5,]    5   91  127  115  216   82    0    0    0     0     0
##  [6,]    6   72  166  104   90   78  121    0    0     0     0
##  [7,]    7   90   99  121   97   65   77   82    0     0     0
##  [8,]    8   72   71   90   64   82  104   56   92     0     0
##  [9,]    9   23   66   70   91   67   67   82   66    99     0
## [10,]   10   90   96   63   82   65   65   38   42    23    67
```

```
RKMeans<-R2
```

## Q4. and Q5. Perform Scree Test and Plot

```
plot.window(xlim = c(0,30),ylim = c(0,150))
plot(Ans[1,],Ans[2,],type = 'l',xlab = "No. of Clusters",ylab = "R2 Values")
```



As per scree plot , it appears that the marginal utility of increasing no. of clusters is continually decreasing (slope becomes less steep). However, there is no clear “elbow” in the plot that can be used to decide on the number of clusters accurately. contd.

## Q6.a

Using VAF/R2 criteria, the highest R2 value is at k=10 , R2= 60.51. However, we should take a closer look at the marginal utility of each extra cluster.

```
Increment<- numeric(length(Ans[2,])-1)
for(i in 2:length(Ans[2,])){Increment[i-1]<-round((Ans[2,i]-Ans[2,i-1])/Ans[2,i-1]
*100,2)}
Increment
```

```
## [1]  -Inf 75.35 35.98 16.37 10.90  6.75  4.68  4.31  3.17
```

At the 7th cluster, the incremental effect of cluster size on R2 falls below 10% and thus we can select this as our ideal no. of clusters.

## Q6 b.

Interpretability of Segments: FOr k=7 let us take a look at the centers :

```
set.seed(30345678)
Clust<-kmeans(dataTrain,centers=7,nstart=50)
R2<-round(Clust$betweenss/Clust$totss *100,2)
CC<-as.matrix( Clust$centers)
Relsize<-numeric()
for(i in 1:7){Relsize[i]<-round(Clust$size[i]/sum(Clust$size)*100,2)}
Relsize
```

```
## [1] 15.37 13.00 12.20 19.18 14.26 10.30 15.69
```

```
R2
```

```
## [1] 54.1
```

Looking at the centroids, we can see the following:

```
CC
```

```
##      Duration      Amount InstallmentRatePercentage ResidenceDuration
## 1 -0.42126229 -0.12787277                -1.3293353       0.002718521
## 2 -0.07998974 -0.41033484                 0.7267142       0.828805347
## 3  1.61523110  1.95330850                -0.4876877      -0.033967116
## 4 -0.07736945 -0.28851589                 0.3485118      -0.102297278
## 5 -0.06406836  0.01611746                -0.1155837      -0.011783539
## 6 -0.44240180 -0.41400245                 0.3046284       0.870881921
## 7 -0.33401233 -0.44427907                 0.5589765      -1.098778815
##           Age NumberExistingCredits NumberPeopleMaintenance
## 1 -0.50284602            -0.4925003              -0.4101847
## 2 -0.28702042            -0.7135299              -0.4101847
## 3  0.07313406            -0.1798534              -0.4101847
## 4 -0.28604886             1.2355496              -0.4101847
## 5  0.20792279             0.3386121               2.4340628
## 6  2.04441700             0.1660547              -0.3664270
## 7 -0.50815988            -0.7135299              -0.4101847
```

```
HighLow<- matrix(0,nrow(CC),ncol(CC))
for (i in 1:nrow(CC)){
  for (j in 1:ncol(CC)){
    if (CC[i,j]>0.6) {HighLow[i,j]<-"+H"
    } else if (CC[i,j]<(-0.6)){HighLow[i,j]<-"-H"
    } else if (CC[i,j]<0){HighLow[i,j]<-"-L"
    } else {HighLow[i,j]<-"+L"}
  }
}
HighLow
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7]
## [1,] "-L" "-L" "-H" "+L" "-L" "-L" "-L"
## [2,] "-L" "-L" "+H" "+H" "-L" "-H" "-L"
## [3,] "+H" "+H" "-L" "-L" "+L" "-L" "-L"
## [4,] "-L" "-L" "+L" "-L" "-L" "+H" "-L"
## [5,] "-L" "+L" "-L" "-L" "+L" "+L" "+H"
## [6,] "-L" "-L" "+L" "+H" "+H" "+L" "-L"
## [7,] "-L" "-L" "+L" "-H" "-L" "-H" "-L"
```

Thus, the clusters can then be interpreted as: c1: Cutomers with high Duration, High Amount Credit C2: Customers with High number of dependents C3: Customers with High number of credits with this bank C4: Customers with Very Low (-H) credits with bank and having few years in present residence C5: Customers with High Installment rates as % of income and Having spent longer time in current residence C6: Aged customers with long periods in current residence and high existing credits C7: Customers with very low interest as % of income

Since these seem to Mutually exclusive and collectively exhaustive, this is a good selection of clusters.

## Q6 c. Testing on dataTest

```
set.seed(30345678)
Clust<-kmeans(dataTest,centers=CC)
R2<-round(Clust$betweenss/Clust$totss *100,2)
CC<-as.matrix(Clust$centers)
Relsize<-numeric()
for(i in 1:7){Relsize[i]<-round(Clust$size[i]/sum(Clust$size)*100,2)}
Relsize
```

```
## [1] 18.43 13.01 12.20 14.91 15.18 10.30 15.99
```

```
R2
```

```
## [1] 52
```

Test Relative Sizes(%) = 11.57 15.98 16.80 18.73 12.40 10.47 14.05

Train Relative Sizes (%) = 11.46 14.13 19.47 14.91 14.13 10.68 15.23

Test R2 = 53.38

Train R2 = 53.48

Thus, within reasonable limits, this is a very stable test performance in VAF as well as relative sizes.Each cluster is fairly evenly distributed suggesting that this is an acceptable clustering result.

```
HighLow<- matrix(0,nrow(CC),ncol(CC))
for (i in 1:nrow(CC)){
  for (j in 1:ncol(CC)){
    if (CC[i,j]>0.6) {HighLow[i,j]<-"+H"
    } else if (CC[i,j]<(-0.6)){HighLow[i,j]<-"-H"
    } else if (CC[i,j]<0){HighLow[i,j]<-"-L"
    } else {HighLow[i,j]<-"+L"}
  }
}
HighLow
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7]
## [1,] "-L" "-L" "-H" "-L" "-L" "+L" "-L"
## [2,] "-L" "-L" "+H" "+H" "-L" "-H" "-L"
## [3,] "+H" "+H" "-L" "-L" "-L" "+L" "-L"
## [4,] "-L" "-L" "+L" "+L" "+L" "+H" "-L"
## [5,] "-L" "-L" "-L" "+L" "+L" "+L" "+H"
## [6,] "-L" "-L" "+L" "+H" "+H" "-L" "-L"
## [7,] "-L" "-L" "+L" "-H" "-L" "-H" "-L"
```

We can see that interpretation of clusters remains exactly same as we have retained the centers from the training data set clusters ####Q7. KO Means

```
R2<-numeric()
for (i in 3:5){
KOClust <- komeans(data=data1[1:7],nclust=i,nloops = 50,lnorm = 2, tolerance = 0.0
01,seed=30345678)
R2[i]<-KOClust$VAF
}
RKOMeans<-R2[3:5]
```

## Q8. Comparing KMeans and KOMeans

```
Tab<-rbind(3:5,round(RKOMeans*100,2),RKMeans[3:5])
rownames(Tab)<-c("No. of CLusters","VAF as per KOMeans", "VAF as per KMeans")
Tab
```

```
##                     [,1]  [,2]  [,3]
## No. of CLusters     3.00  4.00  5.00
## VAF as per KOMeans 34.66 48.09 58.95
## VAF as per KMeans  28.88 39.27 45.70
```

Thus, as expected, KOMeans provides a better VAF value for same number of clusters. On Average KOMeans has ~25% better explainability of Variance as compared to KMeans at the same level of detail (no. of clusters)

Looking at the centroids[ for k=5]

```
KOC<-as.matrix( KOClust$Centroids)
```

For interpretability:

```
KOC
```

```
##       [,1]  [,2]  [,3]  [,4]  [,5]  [,6]  [,7]
## [1,] -0.74 -0.43 -0.76 -0.64 -0.72 -0.73 -0.21
## [2,] -0.27 -0.20  0.04  1.46  0.90  0.16 -0.14
## [3,]  1.45  1.61 -0.72 -0.10 -0.24 -0.07 -0.13
## [4,] -0.17  0.03 -0.36  0.05  0.38  0.33  2.72
## [5,]  0.38 -0.19  1.39 -0.36 -0.01  0.60 -0.15
```

```
HighLow<- matrix(0,nrow(KOC),ncol(KOC))
for (i in 1:ncol(KOC)){
  for (j in 1:nrow(KOC)){
    if (KOC[j,i]>0.6) {HighLow[j,i]<-"+H"
    } else if (KOC[j,i]<(-0.6)){HighLow[j,i]<-"-H"
    } else if (KOC[j,i]<0){HighLow[j,i]<-"-L"
    } else {HighLow[j,i]<-"+L"}
  }
}
HighLow
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7]
## [1,] "-H" "-L" "-H" "-H" "-H" "-H" "-L"
## [2,] "-L" "-L" "+L" "+H" "+H" "+L" "-L"
## [3,] "+H" "+H" "-H" "-L" "-L" "-L" "-L"
## [4,] "-L" "+L" "-L" "+L" "+L" "+L" "+H"
## [5,] "+L" "-L" "+H" "-L" "-L" "+L" "-L"
```

Interpreting based on same coding we have: C1: Customers with very low values for all features except no. of dependents and number of existing credits C2: Customers with High number of dependents C3: Aged Customers seeking very low credit amount with high % of income going into installments C4: Customers with high duration and credit C5: Customers seeking high credit amount with high periods spent in current residence

In this case the residence feature is not as important as in the KMeans clustering and perhaps could be dropped. In addition, we can see that these are not mutually exclusive groups with members in more than one of the 32 groups formed from 5 initial clusters.

Thus we need to look at the cross tab:

### Crosstabbing KMeans and KOMeans for n=5

```
set.seed(30345678)
KClust<-kmeans(dataTrain,centers=5,nstart=50)
table(KClust$cluster,KOClust$Group)
```

```
## 
##      0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22
##   1 15  9  0  3  0  0  0  0  0  0  0  0  0  0  0  0 20 93  2 53  2  7  0
##   2  0  0  0  0 23  6 15  2  0  0  0  0  0  0  0  0  0  0  0  0 12  7 10
##   3  0  0  0  0  0  0  0  0  8 12  4  5  3  2  1  2  0  0  0  0  0  0  0
##   4  9 56  3 19  0  7  0 25  0  0  0  0  0  0  0  0  0  2  0  0  0  3  0
##   5  2  0 34 10  0  0  1  1  0  0  0  0  0  0  0  0  2  0 33 21  0  0  4
## 
##     23 24 25 26 27 28 29 30 31
##   1 14  0  0  0  0  0  0  0  0
##   2  7  0  0  0  0  0  0  0  0
##   3  0  9 15  7  8  5  4  4  2
##   4  3  0  0  0  0  0  0  0  0
##   5  5  0  0  0  0  0  0  0  0
```

Based on the cross tabulated data- we should select cluster nos. 1,13,17,20 and 21 as the 5 KOMeans clusters with the maximum no. of members assigned to those groups ie. ~50% of all members accounted for (318/637).

## Q9. Summary of Observations

In summary:
1. In case of K-Means clustering we are able to form 7 Mutually Exclusive and Collectively Exhaustive clusters that have high interpretability juding by their centroids. In addition, the clusters perform well in test for relative sizes of clusters as well as VAF.
2. The Variance Accounted For (or R2) of the 7 clusters is ~53%. This value is achieved with only 5 clusters in case of KOMeans
3. However, with KOMeans the number of Mutually Exclusive clusters formed is 32. Looking at the overlapping 5 cluster centroids, interpretability is weaker.Thus, we focus on the 5 mutually exclusive groups identified from the cross tabulation against comparable 5 cluster Kmeans results

## Q10 a.

Recruiting candidates :
* In order to be most effective we need to make the tele-operator teams general rather than cluster-focused. This will ensure that based on any update from the present data collected, the recruit can be placed in any of the segments. Also, it helps reduce number of operators needed.
* In addition, to make the recruitment easier we should focus on prospective recruits whose “contrast” is higher i.e. they fit into one particular segment very well and not at all in others. This will provide better results in the A&U part

## Q10 b.

Since the purpose of the research is to be able to more clearly identify the various sectors and learn about them(assumption), the A&U studies will reveal more details.
Focus groups are more useful when looking at a representative sample to generalize to a population.
In our case, we are trying identify how to market better to specific types of customers and thus their attitude towards the product and patterns of usage can provide useful inputs to market to them more effectively.
Thus, I would focus on recruiting for A&Us

## Q10 c.

The centroids of the various clusters reveal the characteristics for the different clusters.
Based on the traits of this ideal customer i.e. based on the consumer profile, we can match against the data available for the recruit and assign him/her to the right cluster.