Cluster Analysis

10/15/2020

Loading required libraries

```
library(cluster)
library(data.table)
library(magrittr)
library(stringr)
library(ggplot2)
library(knitr)
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
library(tidyverse)
```

```
## -- Attaching packages ------------------------------------------------------ tidyverse 1.3.0 --
```

```
## v tibble  3.0.3     v purrr   0.3.4
## v tidyr   1.1.2     v dplyr   1.0.2
## v readr   1.3.1     v forcats 0.5.0
```

```
## -- Conflicts --------------------------------------------------------- tidyverse_conflicts() --
## x dplyr::between()   masks data.table::between()
## x tidyr::extract()   masks magrittr::extract()
## x dplyr::filter()    masks stats::filter()
## x dplyr::first()     masks data.table::first()
## x dplyr::lag()       masks stats::lag()
## x dplyr::last()      masks data.table::last()
## x purrr::set_names() masks magrittr::set_names()
## x purrr::transpose() masks data.table::transpose()
```

```
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

Data Loading

```
Lending_Data <- read_csv('Lending_Data.csv')
```

```
## Parsed with column specification:
## cols(
##   member_id = col_character(),
##   loan_status = col_character(),
##   int_rate = col_character(),
##   Bin_int = col_double(),
##   dti = col_double(),
##   Bin_dti = col_double(),
##   Default_flag = col_double(),
##   No_of_Enquiry = col_double(),
##   enq_buckets = col_character(),
##   annual_inc = col_double(),
##   Income_bins = col_double(),
##   home_ownership = col_character(),
##   purpose = col_character(),
##   open_acc = col_double(),
##   emp_length = col_character(),
##   verification_status = col_character(),
##   delinq_2yrs = col_double(),
##   loan_amnt = col_double(),
##   Bins_loan_amt = col_double()
## )
```

```
Lend = copy(Lending_Data)
Lend = setDT(Lend)
view(Lend)
str(Lend)
```

```
## Classes 'data.table' and 'data.frame':   35808 obs. of  19 variables:
##  $ member_id          : chr  "LC1" "LC10" "LC100" "LC1000" ...
##  $ loan_status        : chr  "Charged Off" "Fully Paid" "Fully Paid" "Fully Paid" ...
##  $ int_rate           : chr  "11.71%" "15.96%" "10.65%" "12.69%" ...
##  $ Bin_int            : num  10 16 8 11 22 1 23 10 5 16 ...
##  $ dti                : num  1.06 2.61 11.34 14 13.01 ...
##  $ Bin_dti            : num  2 3 11 14 13 11 5 10 24 14 ...
##  $ Default_flag       : num  1 0 0 0 0 0 0 0 0 0 ...
##  $ No_of_Enquiry      : num  0 1 1 1 0 0 3 0 1 2 ...
##  $ enq_buckets        : chr  "0" "1-4" "1-4" "1-4" ...
##  $ annual_inc         : num  110000 135000 75000 51000 41500 ...
##  $ Income_bins        : num  9 11 6 4 3 4 12 7 6 4 ...
##  $ home_ownership     : chr  "MORTGAGE" "RENT" "MORTGAGE" "RENT" ...
##  $ purpose            : chr  "credit_card" "other" "educational" "credit_card" ...
##  $ open_acc           : num  6 3 7 5 8 5 4 7 6 9 ...
##  $ emp_length         : chr  "LT 1year" "10+ years" "2 years" "1 year" ...
##  $ verification_status: chr  "Not Verified" "Source Verified" "Source Verified" "Source Verified" ..
##  $ delinq_2yrs        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ loan_amnt          : num  7000 2000 12000 9350 6000 ...
##  $ Bins_loan_amt      : num  6 2 10 8 5 8 5 10 2 8 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   member_id = col_character(),
##   ..   loan_status = col_character(),
##   ..   int_rate = col_character(),
##   ..   Bin_int = col_double(),
```

```
##   ..   dti = col_double(),
##   ..   Bin_dti = col_double(),
##   ..   Default_flag = col_double(),
##   ..   No_of_Enquiry = col_double(),
##   ..   enq_buckets = col_character(),
##   ..   annual_inc = col_double(),
##   ..   Income_bins = col_double(),
##   ..   home_ownership = col_character(),
##   ..   purpose = col_character(),
##   ..   open_acc = col_double(),
##   ..   emp_length = col_character(),
##   ..   verification_status = col_character(),
##   ..   delinq_2yrs = col_double(),
##   ..   loan_amnt = col_double(),
##   ..   Bins_loan_amt = col_double()
##   .. )
##  - attr(*, ".internal.selfref")=<externalptr>
```

Data Cleaning

```
Lend[, member_id := factor(member_id)]
Lend[, loan_status := factor(loan_status)]
Lend[, home_ownership := factor(home_ownership)]
Lend[, purpose := factor(purpose)]
Lend[, verification_status := factor(verification_status)]

Lend[, int_rate := gsub('[%]', '', int_rate)]
Lend[, int_rate := trimws(int_rate)]
Lend[, int_rate := suppressWarnings(as.numeric(int_rate))]

Lend[open_acc %in% c(1,2,3,4,5), 'x' := 'LT5']
Lend[open_acc %in% c(6,7,8,9,10), 'x' := '6-10']
Lend[open_acc %in% c(11,12,13,14,15), 'x' := '11-15']
Lend[open_acc >15, 'x' := '15+']
Lend = Lend %>% rename(no_of_acct = x)
str(Lend)
```

```
## Classes 'data.table' and 'data.frame':   35808 obs. of  20 variables:
##  $ member_id          : Factor w/ 35808 levels "LC1","LC10","LC100",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ loan_status        : Factor w/ 2 levels "Charged Off",..: 1 2 2 2 2 2 2 2 2 2 ...
##  $ int_rate           : num  11.7 16 10.7 12.7 19.7 ...
##  $ Bin_int            : num  10 16 8 11 22 1 23 10 5 16 ...
##  $ dti                : num  1.06 2.61 11.34 14 13.01 ...
##  $ Bin_dti            : num  2 3 11 14 13 11 5 10 24 14 ...
##  $ Default_flag       : num  1 0 0 0 0 0 0 0 0 0 ...
##  $ No_of_Enquiry      : num  0 1 1 1 0 0 3 0 1 2 ...
##  $ enq_buckets        : chr  "0" "1-4" "1-4" "1-4" ...
##  $ annual_inc         : num  110000 135000 75000 51000 41500 ...
##  $ Income_bins        : num  9 11 6 4 3 4 12 7 6 4 ...
##  $ home_ownership     : Factor w/ 5 levels "MORTGAGE","NONE",..: 1 5 1 5 1 1 1 5 5 1 ...
##  $ purpose            : Factor w/ 14 levels "car","credit_card",..: 2 10 4 2 3 3 8 2 10 3 ...
##  $ open_acc           : num  6 3 7 5 8 5 4 7 6 9 ...
```

```
##  $ emp_length         : chr  "LT 1year" "10+ years" "2 years" "1 year" ...
##  $ verification_status: Factor w/ 3 levels "Not Verified",..: 1 2 2 2 3 3 1 1 1 2 ...
##  $ delinq_2yrs        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ loan_amnt          : num  7000 2000 12000 9350 6000 ...
##  $ Bins_loan_amt      : num  6 2 10 8 5 8 5 10 2 8 ...
##  $ no_of_acct         : chr  "6-10" "LT5" "6-10" "LT5" ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   member_id = col_character(),
##   ..   loan_status = col_character(),
##   ..   int_rate = col_character(),
##   ..   Bin_int = col_double(),
##   ..   dti = col_double(),
##   ..   Bin_dti = col_double(),
##   ..   Default_flag = col_double(),
##   ..   No_of_Enquiry = col_double(),
##   ..   enq_buckets = col_character(),
##   ..   annual_inc = col_double(),
##   ..   Income_bins = col_double(),
##   ..   home_ownership = col_character(),
##   ..   purpose = col_character(),
##   ..   open_acc = col_double(),
##   ..   emp_length = col_character(),
##   ..   verification_status = col_character(),
##   ..   delinq_2yrs = col_double(),
##   ..   loan_amnt = col_double(),
##   ..   Bins_loan_amt = col_double()
##   .. )
##  - attr(*, ".internal.selfref")=<externalptr>
##  - attr(*, "index")= int
##   ..- attr(*, "__open_acc")= int  75 113 157 195 377 382 458 611 628 642 ...
```

```
view(Lend)
```

Clustering for purpose at which the loan was taken

Creating a sub-dataset for loan purpose and scalling it:

```
vals = c('int_rate', 'dti', 'No_of_Enquiry', 'annual_inc',
         'open_acc', 'delinq_2yrs', 'loan_amnt')

Lend_purpose = Lend[, c(13, 3, 5, 8, 10, 14, 17, 18)]

Lend_purpose = Lend_purpose %>%
  group_by(purpose)  %>%
  summarise_at(vals, mean, na.rm = TRUE)

setDT(Lend_purpose)

Lend_purpose = Lend_purpose %>%
  remove_rownames %>%
  column_to_rownames(var = "purpose")
```

```
matstd_Lend_purpose = scale(Lend_purpose)
```

Creating a (Euclidean) distance matrix of the standardized data:

```
dist_Lend_purpose <- dist(matstd_Lend_purpose, method = "euclidean")
```

Invoking hclust command (cluster analysis by single linkage method):

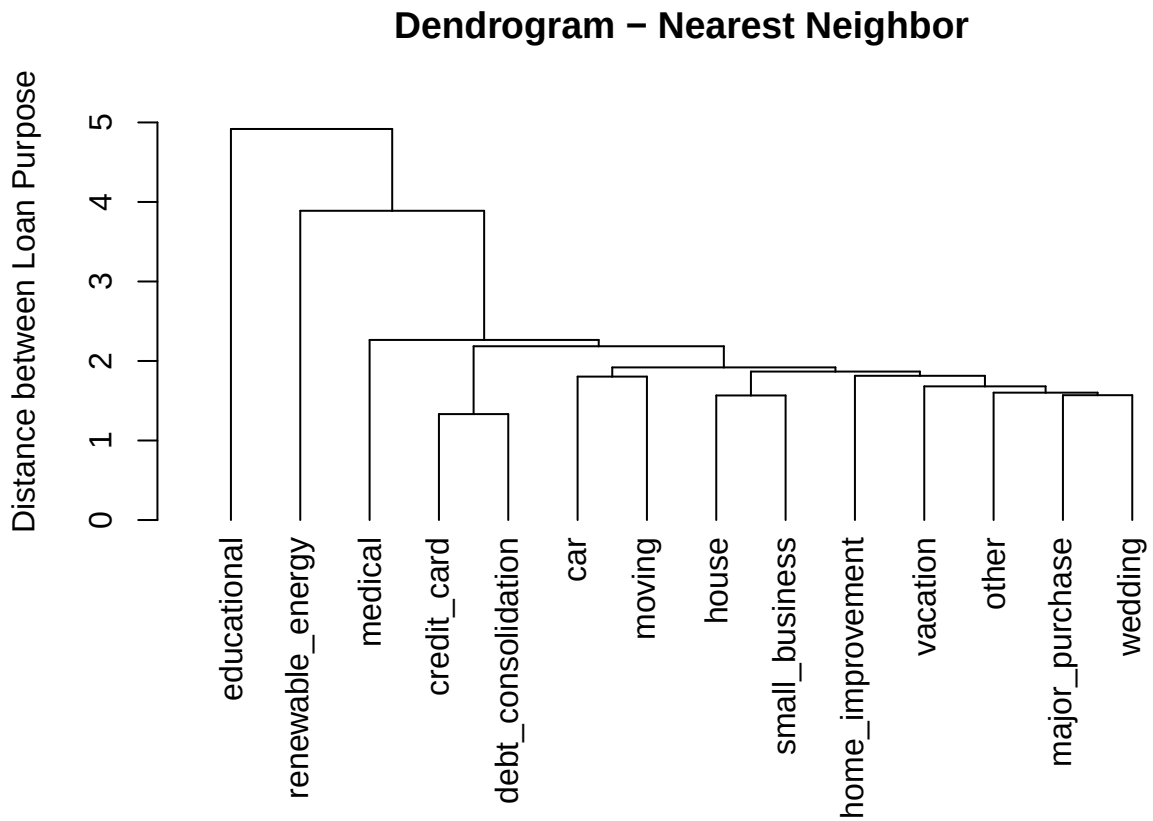
```
cluspurpose_nn <- hclust(dist_Lend_purpose, method = "single")
```

Plotting vertical dendrogram

Create extra margin room in the dendrogram, on the bottom

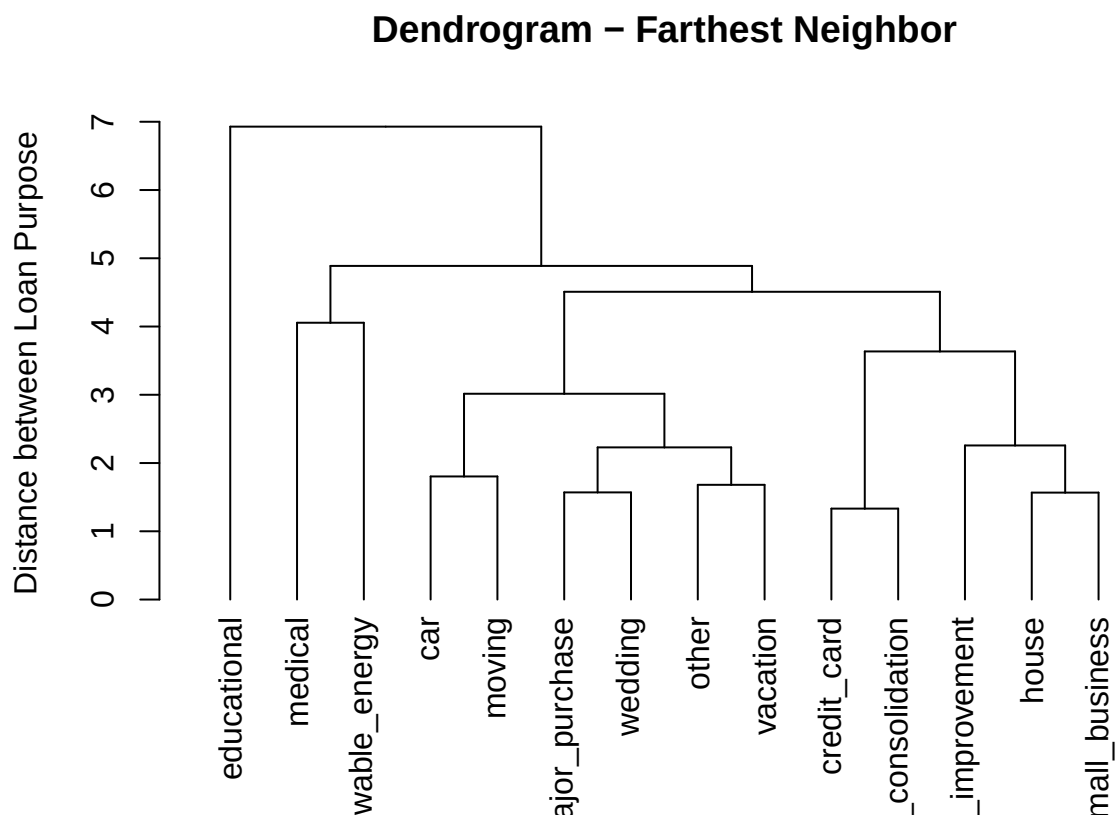
For Nearest Neighbor - Single Linkage:

```
par(mar = c(8, 4, 2, 1) + 0.1)
plot(as.dendrogram(cluspurpose_nn),
     ylab = "Distance between Loan Purpose",
     ylim = c(0, 5.5),
     main = "Dendrogram - Nearest Neighbor")
```



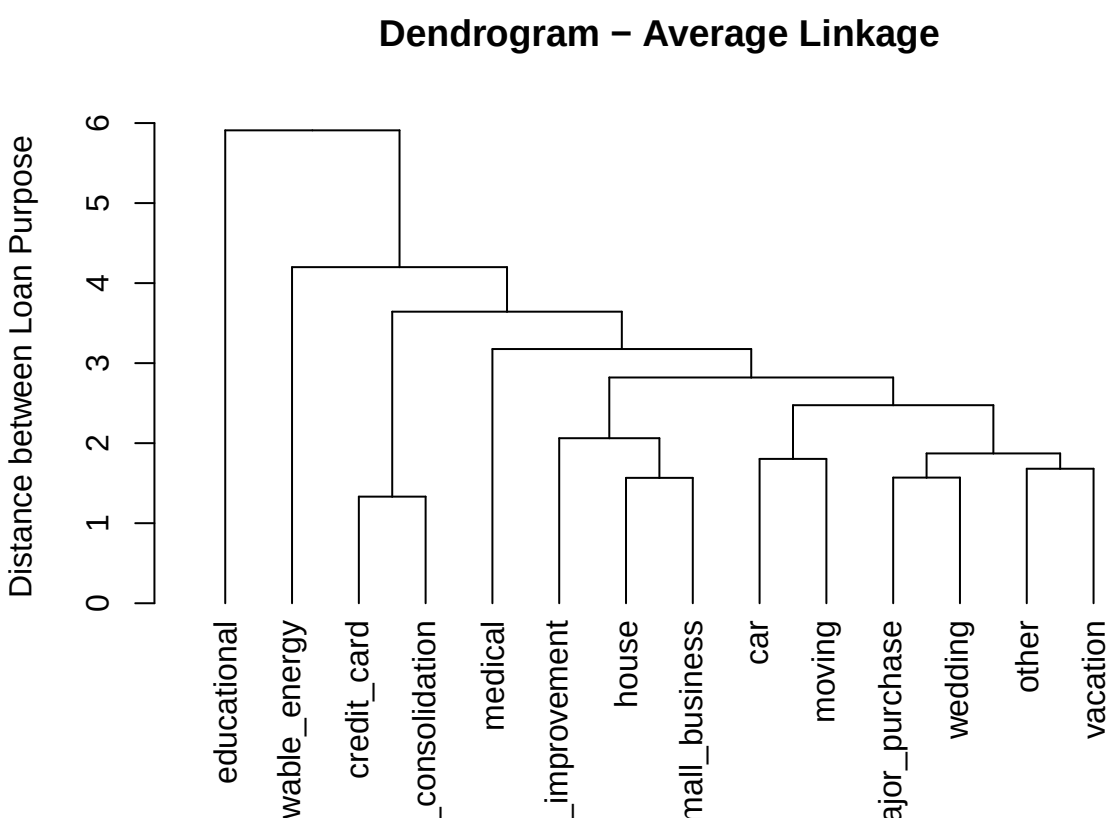
For Farthest Neighbor - Complete Linkage:

```
cluspurpose_fn <- hclust(dist_Lend_purpose)

plot(as.dendrogram(cluspurpose_fn),
     ylab = "Distance between Loan Purpose",
     main = "Dendrogram - Farthest Neighbor")
```



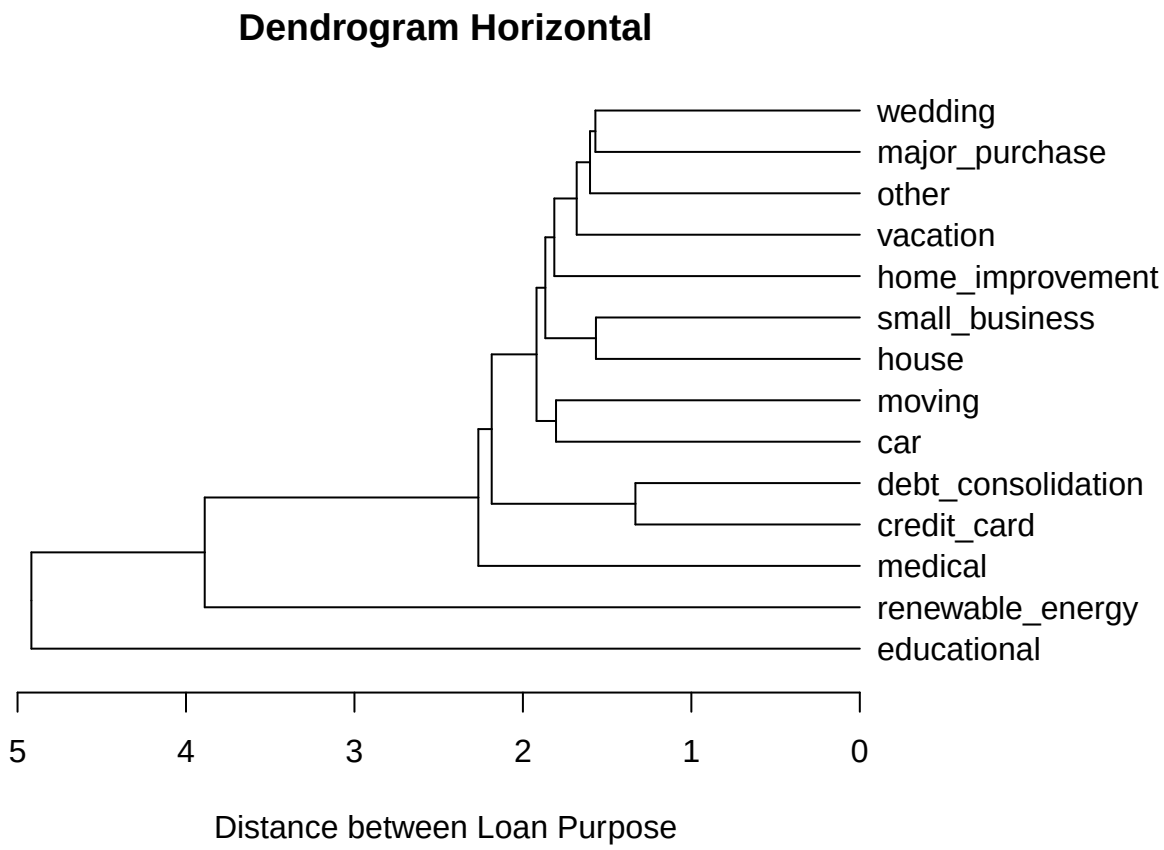
For Average Linkage:

```
cluspurpose_avl <- hclust(dist_Lend_purpose, method = "average")

plot(as.dendrogram(cluspurpose_avl),
     ylab = "Distance between Loan Purpose",
     main = "Dendrogram - Average Linkage")
```



Horizontal Nearest Neightbore for better interpretation:

```
par(mar=c(4, 1, 2, 8) + 0.1)

plot(as.dendrogram(cluspurpose_nn),
     xlab = "Distance between Loan Purpose",
     horiz = TRUE,
     main = "Dendrogram Horizontal")
```



Agnes Function

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

```
agn_purpose <- agnes(Lend_purpose, metric = "euclidean", stand = TRUE, method = "single")
```

Description of cluster merging

```
agn_purpose$merge
```

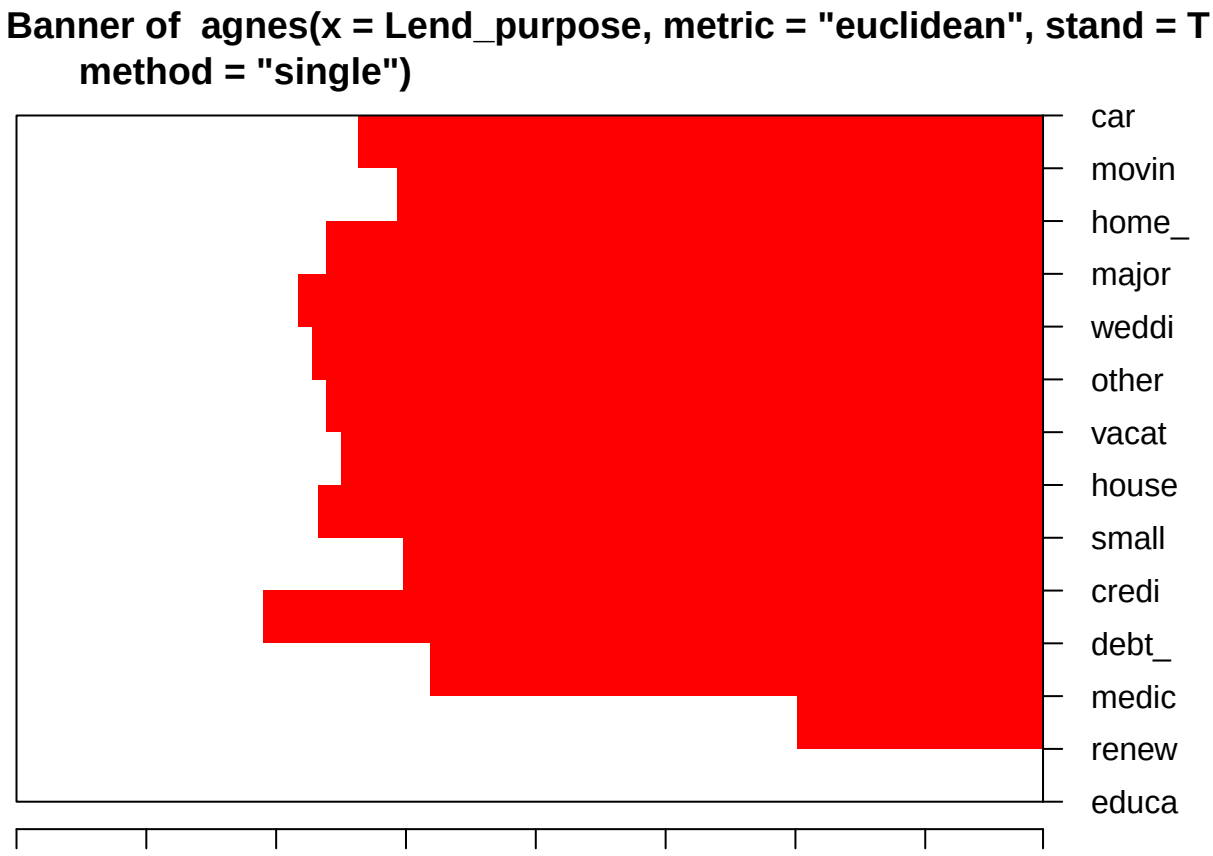
```
##       [,1] [,2]
##  [1,]   -2   -3
##  [2,]   -7  -14
##  [3,]    2  -10
##  [4,]   -6  -12
##  [5,]    3  -13
##  [6,]   -5    5
##  [7,]    6    4
##  [8,]   -1   -9
##  [9,]    8    7
## [10,]    9    1
## [11,]   10   -8
## [12,]   11  -11
## [13,]   12   -4
```

Dendrogram:

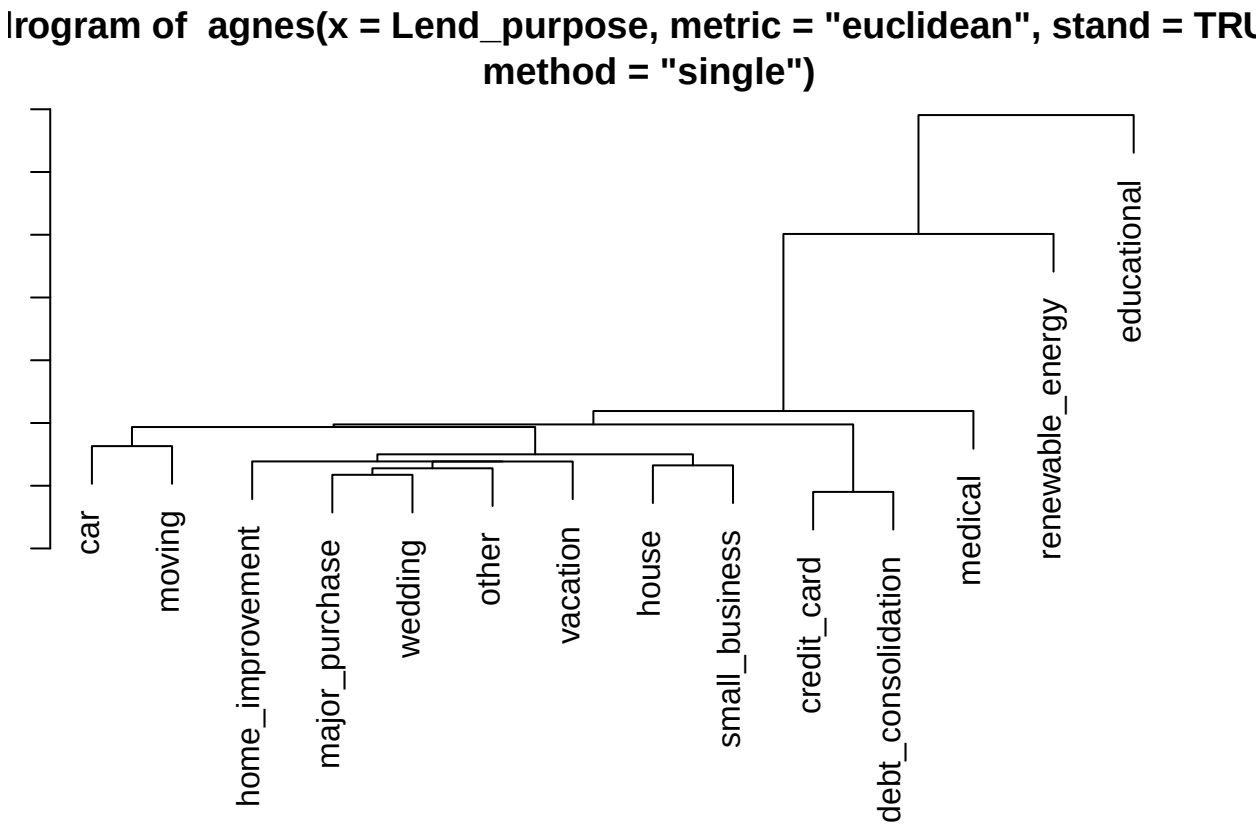
```
par(mar = c(8, 1, 2, 2) + 0.1)

# plot(as.dendrogram(agn_purpose,
#                    xlab = "Distance between Loan Purpose",
#                    horiz = TRUE,
#                    main = "Dendrogram"))
```

Interactive Plots:

```
par(mar = c(1, 1, 2, 2) + 0.1)

#plot(agn_purpose,ask=FALSE)

plot(agn_purpose, which.plots = 1)
```




```
par(mar = c(1, 1, 2, 2) + 0.1)

plot(agn_purpose, which.plots = 2)
```



K-Means Clustering

K-means for Loan Purpose data

K-means, k=2, 3, 4, 5, 6 Centers (k's) are numbers thus, 10 random sets are chosen

Computing the percentage of variation accounted for. Two clusters:

```
kmeans2_purpose <- kmeans(matstd_Lend_purpose, 1, nstart = 10)
perc_var_2 <- round(100*(1 - kmeans2_purpose$betweenss/kmeans2_purpose$totss), 1)
names(perc_var_2) <- "Perc. 2 clus"
perc_var_2
```

```
## Perc. 2 clus
##          100
```

Computing the percentage of variation accounted for. Three clusters:

```
kmeans3_purpose <- kmeans(matstd_Lend_purpose, 3, nstart = 10)
perc_var_3 <- round(100*(1 - kmeans3_purpose$betweenss/kmeans3_purpose$totss), 1)
names(perc_var_3) <- "Perc. 3 clus"
perc_var_3
```

```
## Perc. 3 clus
##           48
```

Computing the percentage of variation accounted for. Four clusters:

```
kmeans4_purpose <- kmeans(matstd_Lend_purpose, 4, nstart = 10)
perc_var_4 <- round(100*(1 - kmeans4_purpose$betweenss/kmeans4_purpose$totss), 1)
names(perc_var_4) <- "Perc. 4 clus"
perc_var_4
```

```
## Perc. 4 clus
##         35.2
```

Computing the percentage of variation accounted for. Five clusters:

```
kmeans5_purpose <- kmeans(matstd_Lend_purpose, 5, nstart = 10)
perc_var_5 <- round(100*(1 - kmeans5_purpose$betweenss/kmeans5_purpose$totss), 1)
names(perc_var_5) <- "Perc. 5 clus"
perc_var_5
```

```
## Perc. 5 clus
##         24.2
```

Computing the percentage of variation accounted for. Six clusters:

```
kmeans6_purpose <- kmeans(matstd_Lend_purpose, 6, nstart = 10)
perc_var_6 <- round(100*(1 - kmeans6_purpose$betweenss/kmeans6_purpose$totss), 1)
names(perc_var_6) <- "Perc. 6 clus"
perc_var_6
```

```
## Perc. 6 clus
##         17.1
```

Saving four k-means clusters in a list:

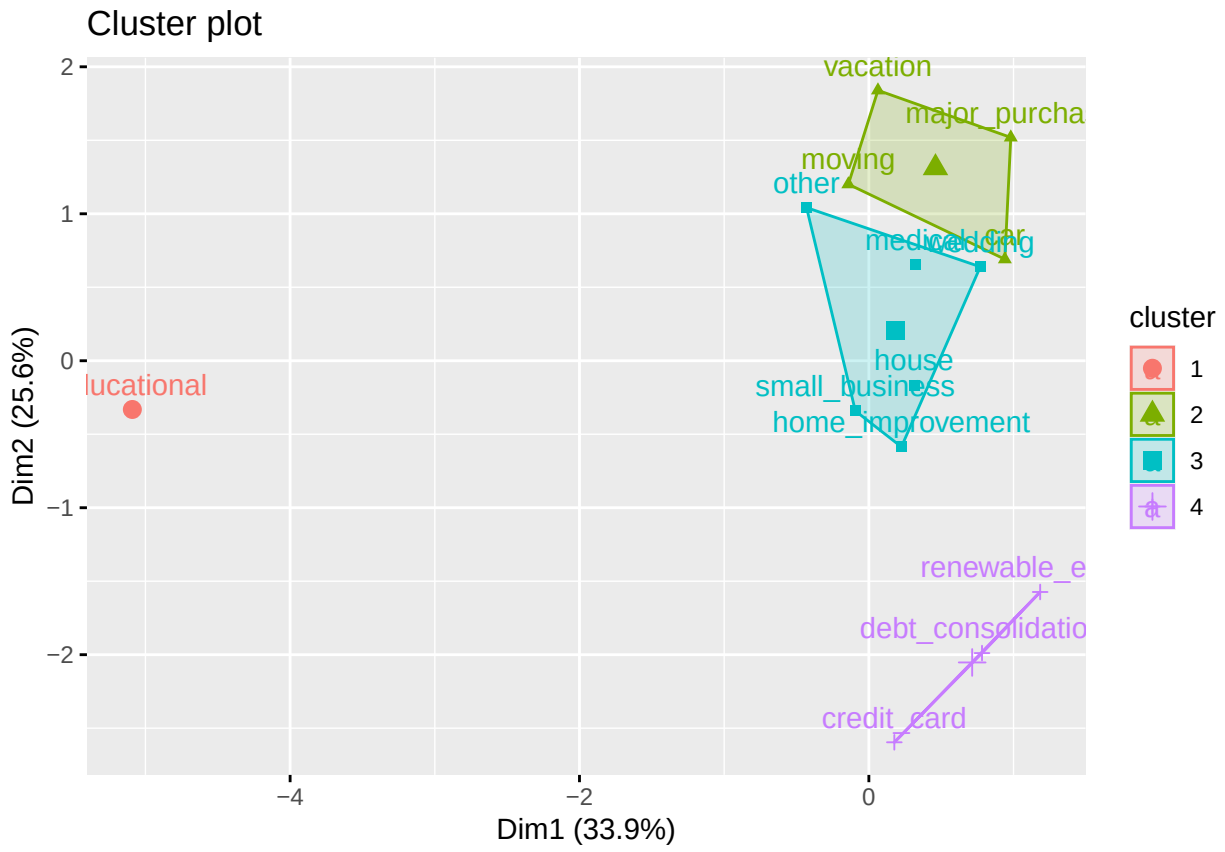
```
clus_1 <- matrix(names(kmeans4_purpose$cluster[kmeans4_purpose$cluster == 1]),
            ncol = 1,
            nrow = length(kmeans4_purpose$cluster[kmeans4_purpose$cluster == 1]))
colnames(clus_1) <- "Cluster 1"

clus_2 <- matrix(names(kmeans4_purpose$cluster[kmeans4_purpose$cluster == 2]),
            ncol = 1,
            nrow = length(kmeans4_purpose$cluster[kmeans4_purpose$cluster == 2]))
colnames(clus_2) <- "Cluster 2"

clus_3 <- matrix(names(kmeans4_purpose$cluster[kmeans4_purpose$cluster == 3]),
            ncol = 1,
            nrow = length(kmeans4_purpose$cluster[kmeans4_purpose$cluster == 3]))
colnames(clus_3) <- "Cluster 3"

clus_4 <- matrix(names(kmeans4_purpose$cluster[kmeans4_purpose$cluster == 4]),
```

```
            ncol = 1,
            nrow = length(kmeans4_purpose$cluster[kmeans4_purpose$cluster == 4]))
colnames(clus_4) <- "Cluster 4"

list(clus_1, clus_2, clus_3, clus_4)
```

```
## [[1]]
##      Cluster 1
## [1,] "educational"
##
## [[2]]
##      Cluster 2
## [1,] "car"
## [2,] "major_purchase"
## [3,] "moving"
## [4,] "vacation"
##
## [[3]]
##      Cluster 3
## [1,] "home_improvement"
## [2,] "house"
## [3,] "medical"
## [4,] "other"
## [5,] "small_business"
## [6,] "wedding"
##
## [[4]]
##      Cluster 4
## [1,] "credit_card"
## [2,] "debt_consolidation"
## [3,] "renewable_energy"
```

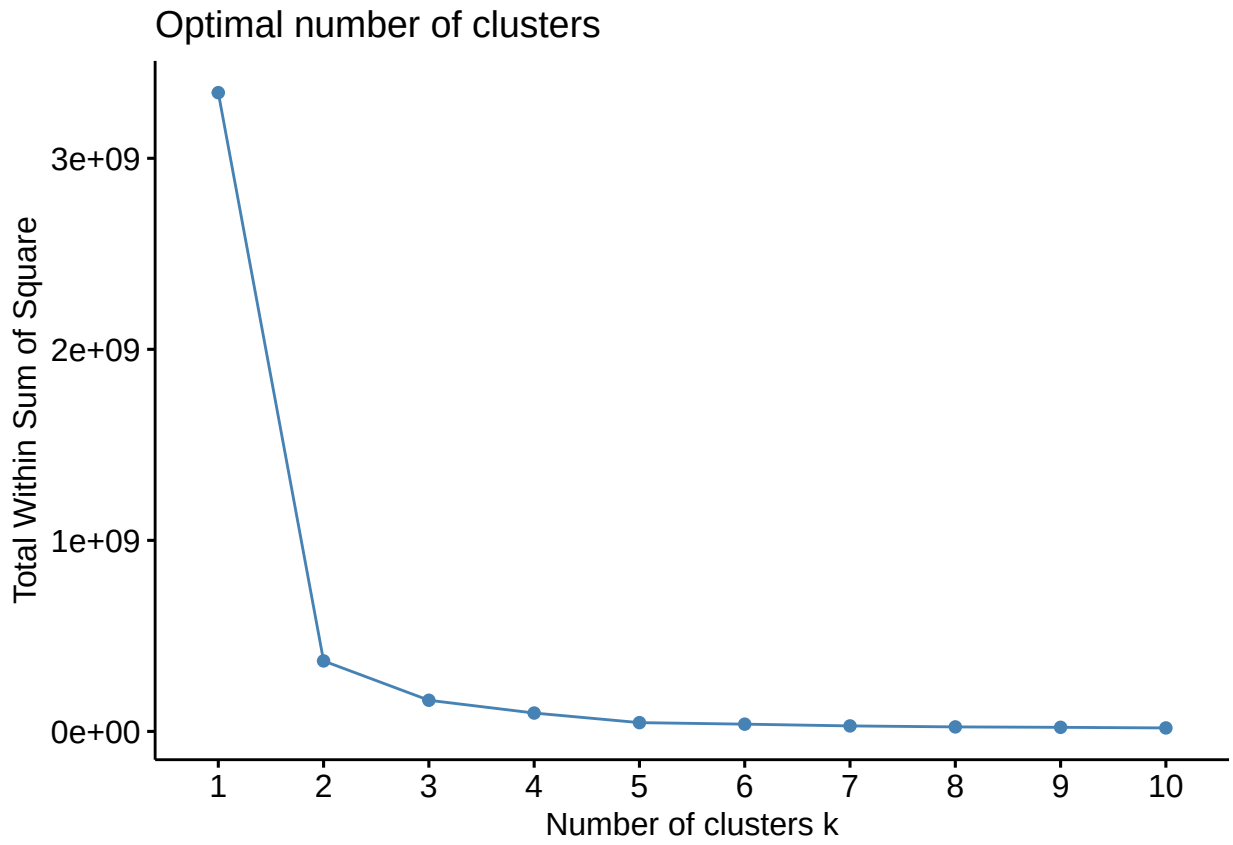
Visualizing the four clusters for the loan purpose:

```
fviz_cluster(kmeans4_purpose, data = Lend_purpose)
```



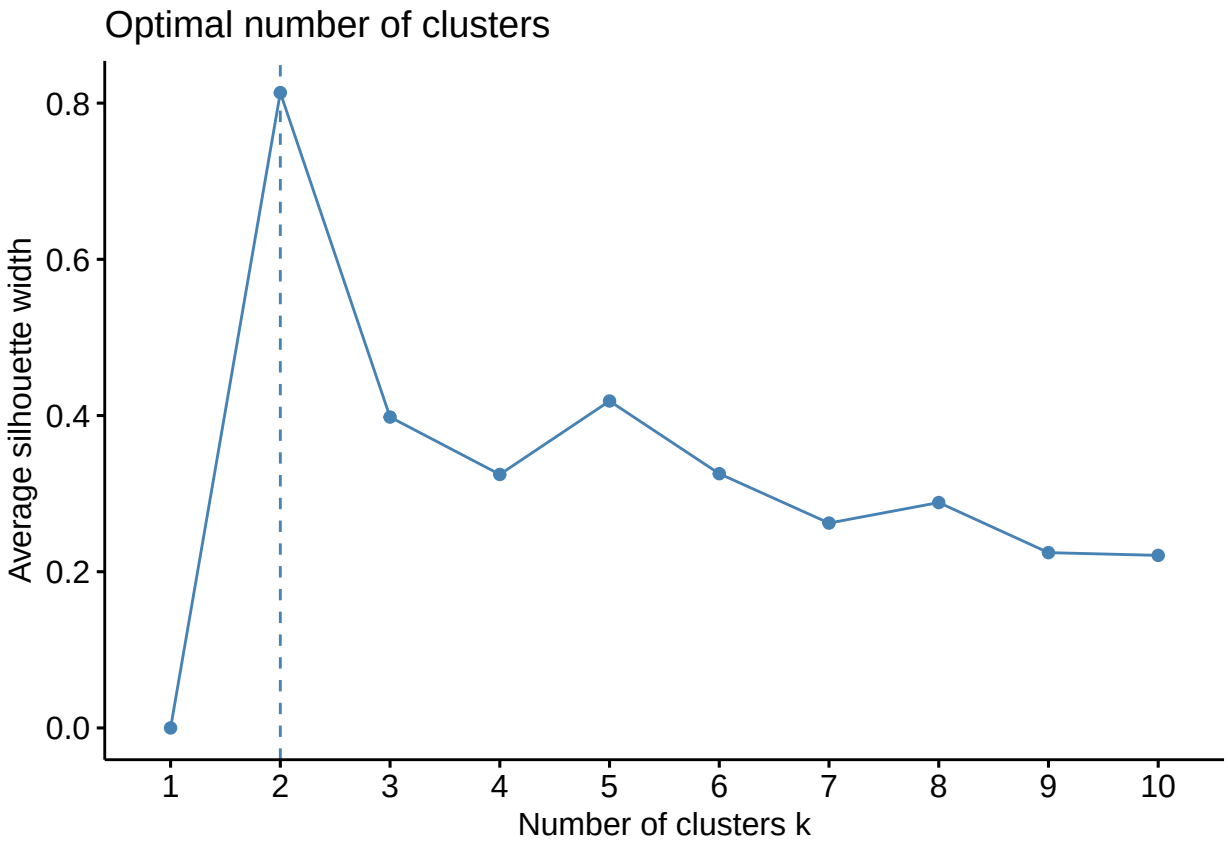
Interpreting optimal number of clusters using wss method:

```
fviz_nbclust(Lend_purpose, kmeans, method = "wss")
```



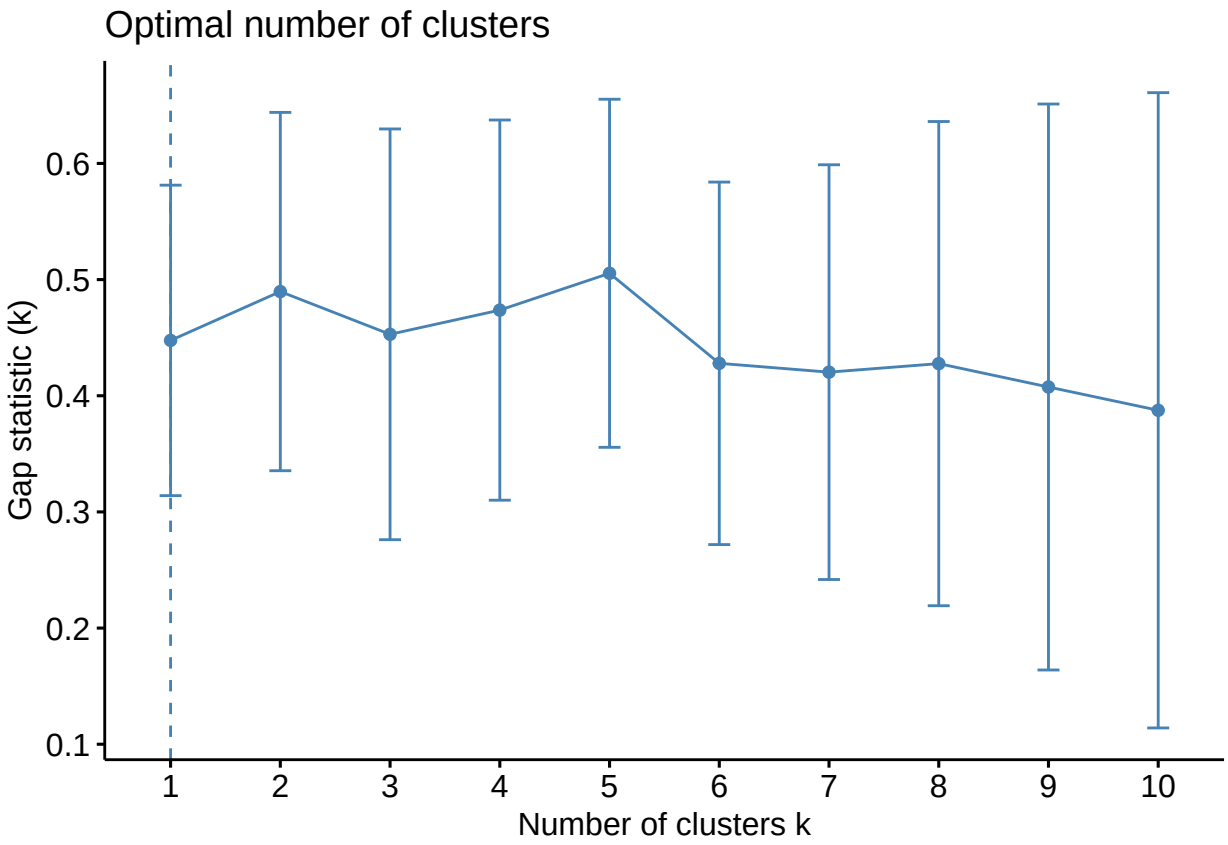
Interpreting optimal number of clusters using silhouette method:

```
fviz_nbclust(Lend_purpose, kmeans, method = "silhouette")
```



Interpreting optimal number of clusters using gap stat method:

```
gap_stat <- clusGap(Lend_purpose, FUN = kmeans, nstart = 25, K.max = 10, B = 50)
fviz_gap_stat(gap_stat)
```



K-means for all the lending data

K-means, k=2, 3, 4, 5, 6 Centers (k's) are numbers thus, 10 random sets are chosen

```
vals = c('int_rate', 'dti', 'No_of_Enquiry', 'annual_inc',
         'open_acc', 'delinq_2yrs', 'loan_amnt')

Lend_kmeans = Lend[, c(1, 3, 5, 8, 10, 14, 17, 18)]
setDT(Lend_kmeans)
Lend_kmeans
```

```
##        member_id int_rate   dti No_of_Enquiry annual_inc open_acc delinq_2yrs
##     1:       LC1    11.71  1.06             0     110000        6           0
##     2:      LC10    15.96  2.61             1     135000        3           0
##     3:     LC100    10.65 11.34             1      75000        7           0
##     4:    LC1000    12.69 14.00             1      51000        5           0
##     5:   LC10000    19.69 13.01             0      41500        8           0
##    ---
## 35804:    LC9995    13.49 19.13             0      48000        6           0
## 35805:    LC9996     9.99 11.40             1      50000        4           0
## 35806:    LC9997     9.99 21.12             2      45000        9           0
## 35807:    LC9998    15.23  7.64             1      30000        3           0
## 35808:    LC9999     8.49  7.10             0     107000       11           0
##        loan_amnt
```

```
##     1:      7000
##     2:      2000
##     3:     12000
##     4:      9350
##     5:      6000
##    ---
## 35804:     14000
## 35805:     20000
## 35806:      6400
## 35807:      1500
## 35808:      6000
```

```
Lend_kmeans = Lend_kmeans %>%
  remove_rownames %>%
  column_to_rownames(var = "member_id")

matstd_Lend_kmeans = scale(Lend_kmeans)
```

Computing the percentage of variation accounted for. Two clusters:

```
kmeans2_lend_kmeans <- kmeans(matstd_Lend_kmeans, 2, nstart = 10)
perc_var_2 <- round(100*(1 - kmeans2_lend_kmeans$betweenss/kmeans2_lend_kmeans$totss), 1)
names(perc_var_2) <- "Perc. 2 clus"
perc_var_2
```

```
## Perc. 2 clus
##         88.8
```

Computing the percentage of variation accounted for. Three clusters:

```
kmeans3_lend_kmeans <- kmeans(matstd_Lend_kmeans, 3, nstart = 10)
perc_var_3 <- round(100*(1 - kmeans3_lend_kmeans$betweenss/kmeans3_lend_kmeans$totss), 1)
names(perc_var_3) <- "Perc. 3 clus"
perc_var_3
```

```
## Perc. 3 clus
##         79.6
```

Computing the percentage of variation accounted for. Four clusters:

```
kmeans4_lend_kmeans <- kmeans(matstd_Lend_kmeans, 4, nstart = 10)
```

```
## Warning: did not converge in 10 iterations
```

```
perc_var_4 <- round(100*(1 - kmeans4_lend_kmeans$betweenss/kmeans4_lend_kmeans$totss), 1)
names(perc_var_4) <- "Perc. 4 clus"
perc_var_4
```

```
## Perc. 4 clus
##         71.9
```

Computing the percentage of variation accounted for. Five clusters:

```
kmeans5_lend_kmeans <- kmeans(matstd_Lend_kmeans, 5, nstart = 10)
```

```
## Warning: Quick-TRANSfer stage steps exceeded maximum (= 1790400)
```

```
perc_var_5 <- round(100*(1 - kmeans5_lend_kmeans$betweenss/kmeans5_lend_kmeans$totss), 1)
names(perc_var_5) <- "Perc. 5 clus"
perc_var_5
```

```
## Perc. 5 clus
##         65.9
```

Computing the percentage of variation accounted for. Six clusters:

```
kmeans6_lend_kmeans <- kmeans(matstd_Lend_kmeans, 6, nstart = 10)
perc_var_6 <- round(100*(1 - kmeans6_lend_kmeans$betweenss/kmeans6_lend_kmeans$totss), 1)
names(perc_var_6) <- "Perc. 6 clus"
perc_var_6
```

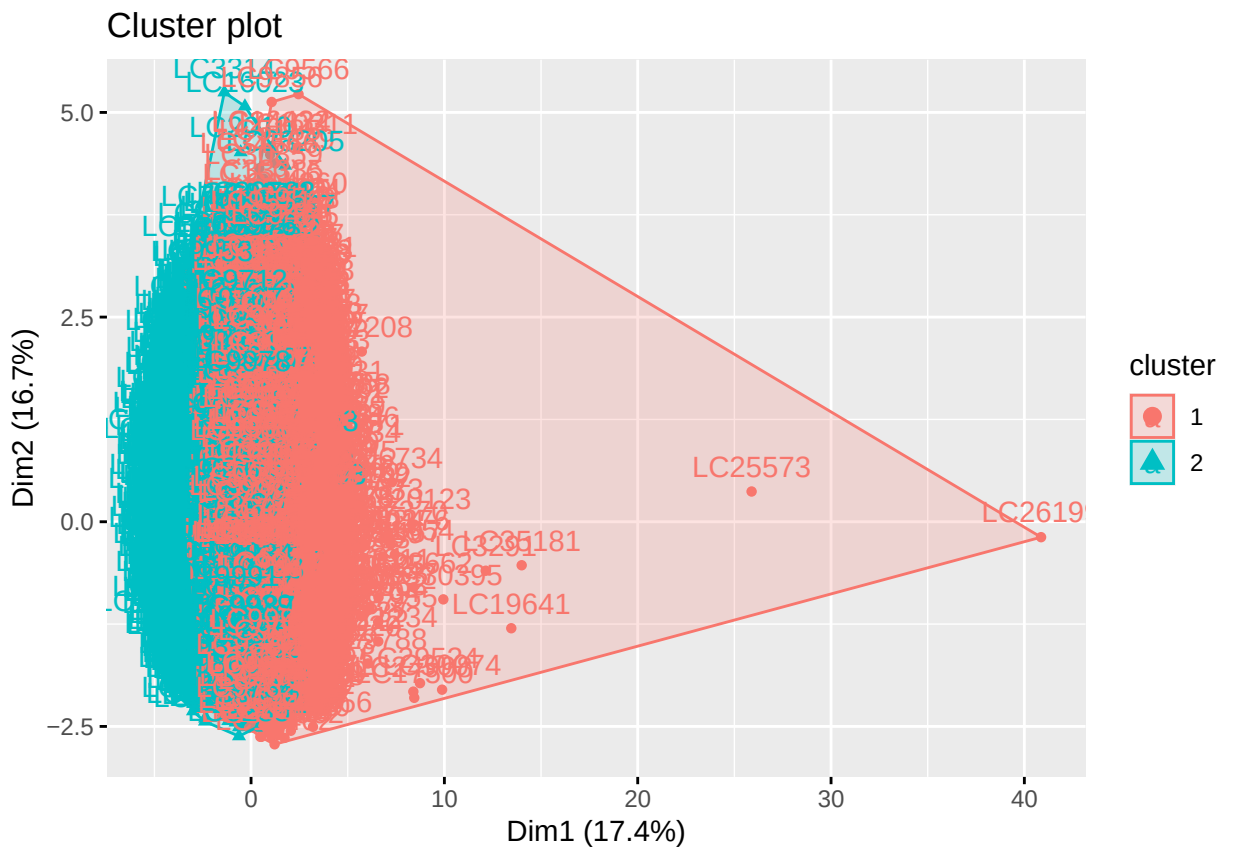
```
## Perc. 6 clus
##         61.2
```

Saving four k-means clusters in a list:

```
clus_1 <- matrix(names(kmeans4_lend_kmeans$cluster[kmeans4_lend_kmeans$cluster == 1]),
                 ncol = 1,
                 nrow = length(kmeans4_lend_kmeans$cluster[kmeans4_lend_kmeans$cluster == 1]))
colnames(clus_1) <- "Cluster 1"

clus_2 <- matrix(names(kmeans4_lend_kmeans$cluster[kmeans4_lend_kmeans$cluster == 2]),
                 ncol = 1,
                 nrow = length(kmeans4_lend_kmeans$cluster[kmeans4_lend_kmeans$cluster == 2]))
colnames(clus_2) <- "Cluster 2"

clus_3 <- matrix(names(kmeans4_lend_kmeans$cluster[kmeans4_lend_kmeans$cluster == 3]),
                 ncol = 1,
                 nrow = length(kmeans4_lend_kmeans$cluster[kmeans4_lend_kmeans$cluster == 3]))
colnames(clus_3) <- "Cluster 3"

clus_4 <- matrix(names(kmeans4_lend_kmeans$cluster[kmeans4_lend_kmeans$cluster == 4]),
                 ncol = 1,
                 nrow = length(kmeans4_lend_kmeans$cluster[kmeans4_lend_kmeans$cluster == 4]))
colnames(clus_4) <- "Cluster 4"

#list(clus_1, clus_2, clus_3, clus_4)
```

Visualizing the two clusters for the complete lending data:

```
fviz_cluster(kmeans2_lend_kmeans, data = Lend_kmeans)
```



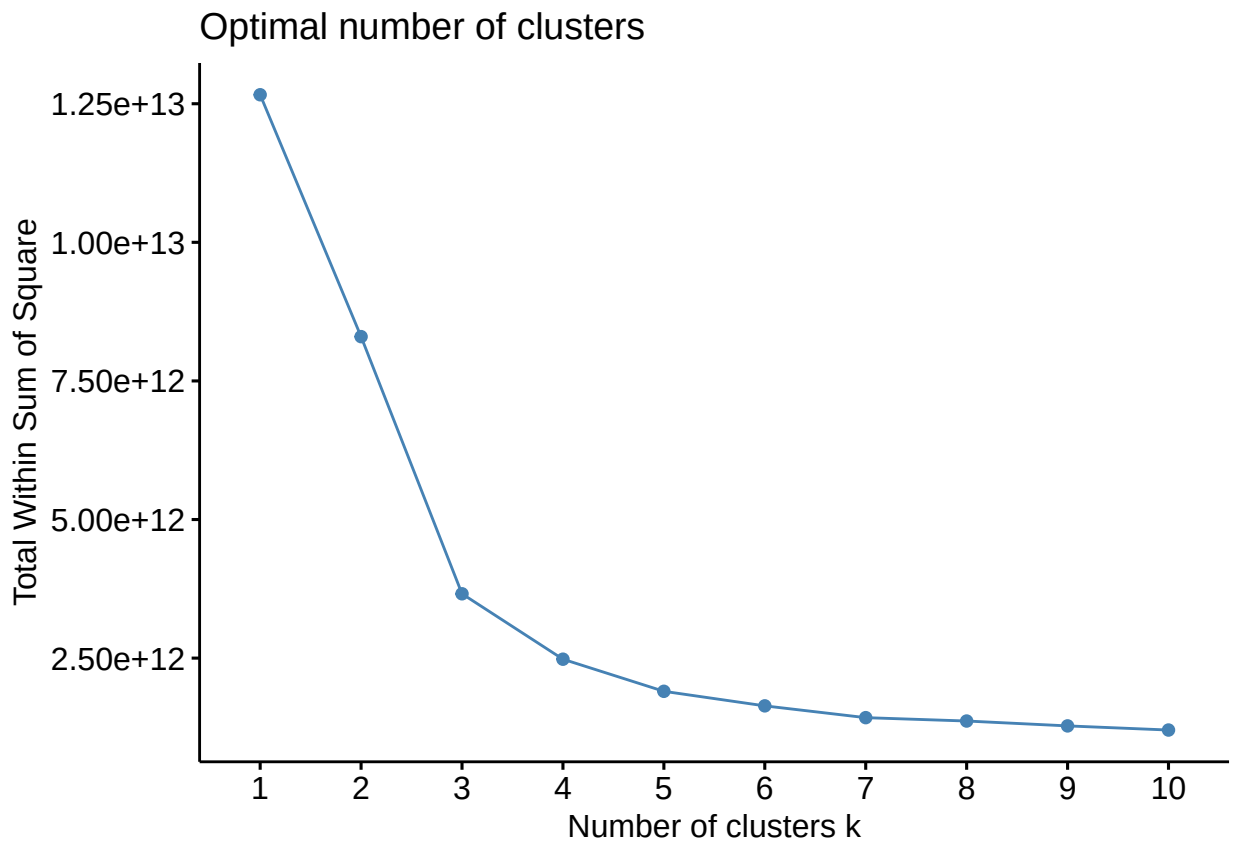
Creating sample for visualizing optimum number clusters of a large dataset:

```
#set the seed to make our partition reproducible
set.seed(123)
smp_size = floor(0.10 * NROW(Lend_kmeans))
train = Lend_kmeans[sample((NROW(Lend_kmeans)), size = smp_size),]
head(train)
```

```
##         int_rate   dti No_of_Enquiry annual_inc open_acc delinq_2yrs loan_amnt
## LC12685    14.54 15.94             0      43200       16           0     19000
## LC4702     17.99 19.04             0     105000        7           0      5000
## LC4509      7.49 12.70             2     111000        7           0      5750
## LC12479    10.00  8.88             1      40800       25           0      3900
## LC18676     9.99 11.06             1      60000        8           0      6000
## LC5952     13.49  6.32             1     120000        7           0      2500
```

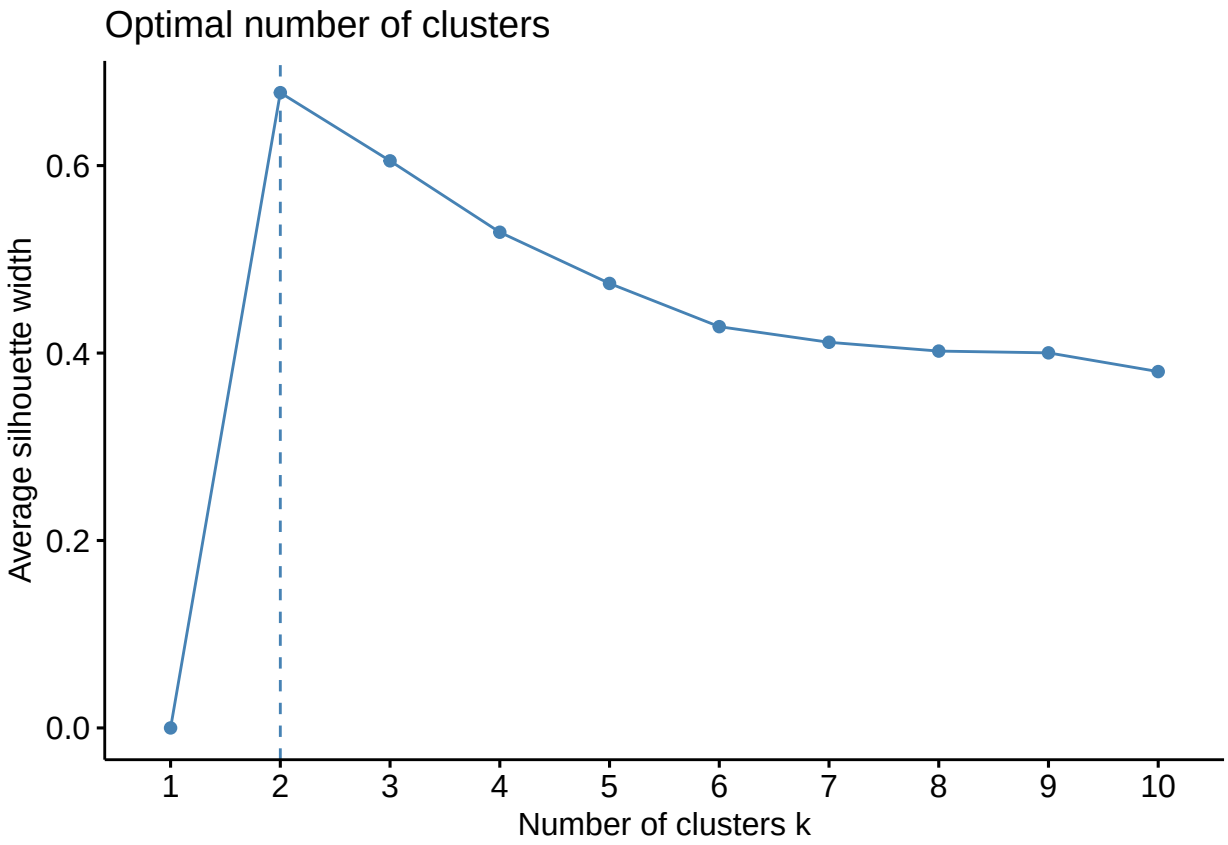
Interpreting optimal number of clusters using wss method:

```
fviz_nbclust(train, kmeans, method = "wss")
```



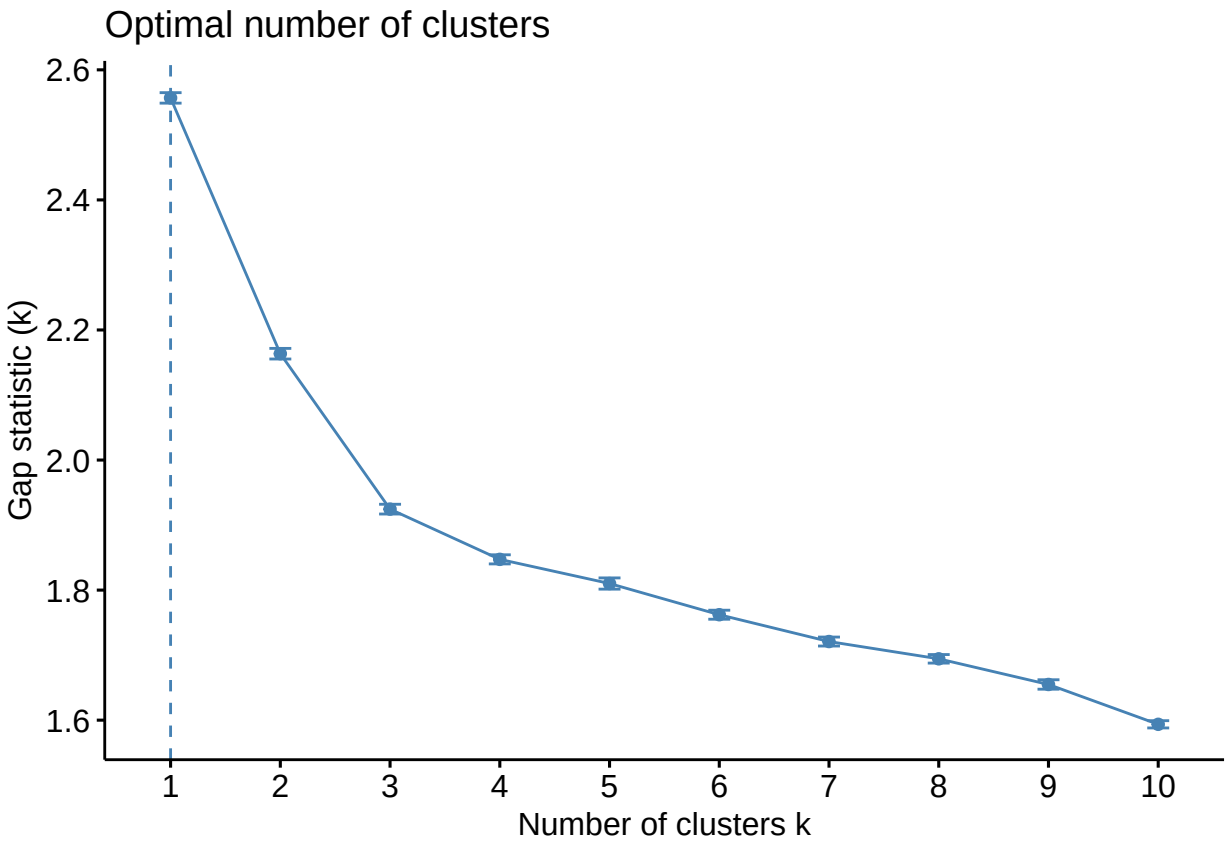
Interpreting optimal number of clusters using silhouette method:

```
fviz_nbclust(train, kmeans, method = "silhouette")
```



Interpreting optimal number of clusters using gap stat method:

```
gap_stat <- clusGap(train, FUN = kmeans, nstart = 25, K.max = 10, B = 50)
fviz_gap_stat(gap_stat)
```


Optimal number of clusters
Gap statistic (k)
2.6
2.4
2.2
2.0
1.8
1.6
1
2
3
4
5
6
7
8
9
10
Number of clusters k