Ford Ka Demographic Clustering Analysis

Alan Montgomery

This analysis creates customer segments for Ford Ka based upon the demographic data using k-Means. It requires the Excel spreadsheet with the data (FordKaData.xlsx). This script creates clusters using the demographic data, but does not consider the psychographic data.

1 Setup

This portion of the script loads the packages into the session.

```
# load in additional packages to extend R functionality
if (!require(lattice)) {install.packages("lattice"); library(lattice)}
if (!require(gplots)) {install.packages("gplots"); library(gplots)}
if (!require(ggplot2)) {install.packages("ggplot2"); library(ggplot2)}
if (!require(reshape2)) {install.packages("reshape2"); library(reshape2)}
if (!require(openxlsx)) {install.packages("openxlsx"); library(openxlsx)}
```

2 Transform data

This portion of the analysis defines the working directory. You will need to change the working directory to where the data is stored on your system. It then reads in the data from an Excel worksheet and standardizes the data. The standardization creates a new standardized variable `xford` that has a mean of 0 and a standard deviation of 1. There are many ways to standardize data, and this is not the only approach.

```
# set to your correct working directory
setwd("~/Documents/class/data science/hw/ford ka/data/") #!! set to your directory!!

# read in Ford Ka datasets from the Excel file
forddemo=read.xlsx("FordKaData.xlsx",sheet=1,startRow=7,colNames=T,rowNames=F,cols=2:10)  # read the de
fordpsyc=read.xlsx("FordKaData.xlsx",sheet=4,startRow=7,colNames=T,rowNames=F,cols=2:63)  # read the ps
fordquest=read.xlsx("FordKaData.xlsx",sheet=5,startRow=7,colNames=T,rowNames=F,cols=2)  # read the ques

# transform the data to make it easier to use
fordquest=paste0(1:62,',',fordquest$Statement)  # convert the question list into a character string to
afordquest=strtrim(fordquest,30)  # truncate the strings to the first 30 characters since some question
ford=cbind(forddemo,fordpsyc)  # create a new dataframe with both demogrpahic and psychographic data

# create some lists of variables which we will use later in the script
nqlist=1:62  # sequence of numbers from 1 to 62
qlist=paste0("Q",nqlist)
# let's try to cluster our questions by transposing the question data
nshortqlist=c(30,57,53,1,4,12)  # short list of questions
shortqlist=paste0("Q",nshortqlist)  # append Q in front of the numbers to generate a list of questions
shortqname=strtrim(fordquest[nshortqlist],30)  # the first 30 characters of the strings
nvars=match(qlist,colnames(ford))   # define list of numeric variables

# define a list of demographic variables that we will use later in the script
dlist=c("Age","MaritalStatus","Gender","NumberChildren","IncomeCategory","FirstTimePurchase")
```

```r
# create new standardized datasets using the scale function (set the mean of the new variable to 0 and
xforddemo=scale(forddemo)
xfordpsyc=scale(fordpsyc)
xford=scale(ford)
```

3 Exploratory Analysis

3.1 Understanding the data structure

The following lines are for looking at the structure of the data, and are meant to encourage familiarity with the structure of the data. Notice that `ford` is a data frame, which is a special type of list in R. `str` is quite helpful for describing both the structure of the data as well as the first few observations of each of the data.

```r
# to list the variables in each data frame
ls(forddemo)
```

```
## [1] "Age"               "AgeCategory"       "ChildrenCategory"
## [4] "FirstTimePurchase" "Gender"            "IncomeCategory"
## [7] "MaritalStatus"     "NumberChildren"    "PreferenceGroup"
```

```r
ls(fordpsyc)
```

```
##  [1] "Q1"  "Q10" "Q11" "Q12" "Q13" "Q14" "Q15" "Q16" "Q17" "Q18" "Q19"
## [12] "Q2"  "Q20" "Q21" "Q22" "Q23" "Q24" "Q25" "Q26" "Q27" "Q28" "Q29"
## [23] "Q3"  "Q30" "Q31" "Q32" "Q33" "Q34" "Q35" "Q36" "Q37" "Q38" "Q39"
## [34] "Q4"  "Q40" "Q41" "Q42" "Q43" "Q44" "Q45" "Q46" "Q47" "Q48" "Q49"
## [45] "Q5"  "Q50" "Q51" "Q52" "Q53" "Q54" "Q55" "Q56" "Q57" "Q58" "Q59"
## [56] "Q6"  "Q60" "Q61" "Q62" "Q7"  "Q8"  "Q9"
```

```r
ls(ford)
```

```
##  [1] "Age"               "AgeCategory"       "ChildrenCategory"
##  [4] "FirstTimePurchase" "Gender"            "IncomeCategory"
##  [7] "MaritalStatus"     "NumberChildren"    "PreferenceGroup"
## [10] "Q1"                "Q10"               "Q11"
## [13] "Q12"               "Q13"               "Q14"
## [16] "Q15"               "Q16"               "Q17"
## [19] "Q18"               "Q19"               "Q2"
## [22] "Q20"               "Q21"               "Q22"
## [25] "Q23"               "Q24"               "Q25"
## [28] "Q26"               "Q27"               "Q28"
## [31] "Q29"               "Q3"                "Q30"
## [34] "Q31"               "Q32"               "Q33"
## [37] "Q34"               "Q35"               "Q36"
## [40] "Q37"               "Q38"               "Q39"
## [43] "Q4"                "Q40"               "Q41"
## [46] "Q42"               "Q43"               "Q44"
## [49] "Q45"               "Q46"               "Q47"
## [52] "Q48"               "Q49"               "Q5"
## [55] "Q50"               "Q51"               "Q52"
## [58] "Q53"               "Q54"               "Q55"
## [61] "Q56"               "Q57"               "Q58"
## [64] "Q59"               "Q6"                "Q60"
## [67] "Q61"               "Q62"               "Q7"
## [70] "Q8"                "Q9"
```

```
# remember these data sets are made up of lists of objects
typeof(ford)      # notice that ford is a list
```

```
## [1] "list"
```

```
names(ford)       # this is the list of object names within ford
```

```
##  [1] "PreferenceGroup"   "Gender"            "Age"
##  [4] "MaritalStatus"     "NumberChildren"    "FirstTimePurchase"
##  [7] "AgeCategory"       "ChildrenCategory"  "IncomeCategory"
## [10] "Q1"                "Q2"                "Q3"
## [13] "Q4"                "Q5"                "Q6"
## [16] "Q7"                "Q8"                "Q9"
## [19] "Q10"               "Q11"               "Q12"
## [22] "Q13"               "Q14"               "Q15"
## [25] "Q16"               "Q17"               "Q18"
## [28] "Q19"               "Q20"               "Q21"
## [31] "Q22"               "Q23"               "Q24"
## [34] "Q25"               "Q26"               "Q27"
## [37] "Q28"               "Q29"               "Q30"
## [40] "Q31"               "Q32"               "Q33"
## [43] "Q34"               "Q35"               "Q36"
## [46] "Q37"               "Q38"               "Q39"
## [49] "Q40"               "Q41"               "Q42"
## [52] "Q43"               "Q44"               "Q45"
## [55] "Q46"               "Q47"               "Q48"
## [58] "Q49"               "Q50"               "Q51"
## [61] "Q52"               "Q53"               "Q54"
## [64] "Q55"               "Q56"               "Q57"
## [67] "Q58"               "Q59"               "Q60"
## [70] "Q61"               "Q62"
```

```
class(ford)       # the ford object itself is a special type of list known as a data.frame
```

```
## [1] "data.frame"
```

```
attributes(ford)  # this prints an objects attributes -- which usually has names of columns and rows
```

```
## $names
##  [1] "PreferenceGroup"   "Gender"            "Age"
##  [4] "MaritalStatus"     "NumberChildren"    "FirstTimePurchase"
##  [7] "AgeCategory"       "ChildrenCategory"  "IncomeCategory"
## [10] "Q1"                "Q2"                "Q3"
## [13] "Q4"                "Q5"                "Q6"
## [16] "Q7"                "Q8"                "Q9"
## [19] "Q10"               "Q11"               "Q12"
## [22] "Q13"               "Q14"               "Q15"
## [25] "Q16"               "Q17"               "Q18"
## [28] "Q19"               "Q20"               "Q21"
## [31] "Q22"               "Q23"               "Q24"
## [34] "Q25"               "Q26"               "Q27"
## [37] "Q28"               "Q29"               "Q30"
## [40] "Q31"               "Q32"               "Q33"
## [43] "Q34"               "Q35"               "Q36"
## [46] "Q37"               "Q38"               "Q39"
## [49] "Q40"                "Q41"                "Q42"
## [52] "Q43"                "Q44"                "Q45"
## [55] "Q46"                "Q47"                "Q48"
## [58] "Q49"                "Q50"                "Q51"
## [61] "Q52"                "Q53"                "Q54"
## [64] "Q55"                "Q56"                "Q57"
## [67] "Q58"                "Q59"                "Q60"
## [70] "Q61"                "Q62"
##
## $row.names
##   [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17
##  [18]  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34
##  [35]  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51
##  [52]  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68
##  [69]  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85
##  [86]  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102
## [103] 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119
## [120] 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136
## [137] 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153
## [154] 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170
## [171] 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187
## [188] 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204
## [205] 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221
## [222] 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238
## [239] 239 240 241 242 243 244 245 246 247 248 249 250
##
## $class
## [1] "data.frame"
```

```r
str(ford)         # a more verbose way of checking information about an object is with structure
```

```
## 'data.frame':    250 obs. of  71 variables:
##  $ PreferenceGroup  : num  1 3 2 3 1 1 1 3 1 1 ...
##  $ Gender           : num  2 1 2 1 2 2 1 1 2 2 ...
##  $ Age              : num  44 24 34 44 41 26 33 48 32 34 ...
##  $ MaritalStatus    : num  3 2 3 3 1 1 3 3 1 3 ...
##  $ NumberChildren   : num  0 1 1 0 2 1 0 0 3 0 ...
##  $ FirstTimePurchase: num  2 1 2 2 1 1 2 2 2 2 ...
##  $ AgeCategory      : num  5 1 3 5 5 2 3 6 3 3 ...
##  $ ChildrenCategory : num  0 1 1 0 2 1 0 0 2 0 ...
##  $ IncomeCategory   : num  6 3 1 3 4 4 6 4 1 4 ...
##  $ Q1               : num  6 7 5 4 5 6 7 6 6 6 ...
##  $ Q2               : num  2 7 4 2 5 6 7 7 4 2 ...
##  $ Q3               : num  4 7 6 5 7 4 3 3 6 4 ...
##  $ Q4               : num  3 5 5 4 6 4 3 4 1 5 ...
##  $ Q5               : num  1 4 7 2 7 5 5 5 3 1 ...
##  $ Q6               : num  5 4 5 4 3 3 4 3 4 5 ...
##  $ Q7               : num  5 5 3 5 4 5 4 4 4 3 ...
##  $ Q8               : num  3 4 5 4 5 2 4 3 4 4 ...
##  $ Q9               : num  4 5 4 3 4 5 4 5 3 5 ...
##  $ Q10              : num  4 5 5 4 2 3 3 3 4 3 ...
##  $ Q11              : num  4 4 5 4 5 4 7 4 4 4 ...
##  $ Q12              : num  5 4 5 4 4 4 4 4 3 5 ...
##  $ Q13              : num  4 4 6 6 4 5 5 5 6 4 ...
##  $ Q14              : num  7 2 3 5 5 1 1 1 7 5 ...
##  $ Q15                : num  6 3 3 6 4 2 3 4 7 7 ...
##  $ Q16                : num  7 4 4 7 3 4 5 4 5 6 ...
##  $ Q17                : num  6 4 2 6 2 4 3 5 7 6 ...
##  $ Q18                : num  5 3 4 5 5 5 3 4 5 6 ...
##  $ Q19                : num  5 4 3 5 4 4 4 4 2 6 ...
##  $ Q20                : num  6 2 4 5 5 1 2 2 5 7 ...
##  $ Q21                : num  7 4 2 6 5 3 2 4 6 7 ...
##  $ Q22                : num  5 4 3 6 5 5 4 5 7 7 ...
##  $ Q23                : num  2 7 3 1 3 7 7 7 3 2 ...
##  $ Q24                : num  1 1 4 1 4 1 1 1 5 1 ...
##  $ Q25                : num  2 4 2 1 5 5 5 4 1 3 ...
##  $ Q26                : num  3 3 4 2 2 3 3 4 2 3 ...
##  $ Q27                : num  1 4 5 2 2 5 4 5 4 3 ...
##  $ Q28                : num  1 5 3 2 3 4 6 5 2 2 ...
##  $ Q29                : num  2 7 3 3 1 5 4 5 5 1 ...
##  $ Q30                : num  2 4 4 1 4 3 5 5 4 3 ...
##  $ Q31                : num  4 1 7 2 6 2 1 1 7 3 ...
##  $ Q32                : num  4 5 5 5 7 6 3 5 5 2 ...
##  $ Q33                : num  5 5 7 4 7 3 5 4 5 3 ...
##  $ Q34                : num  4 5 5 5 7 4 2 3 4 6 ...
##  $ Q35                : num  3 3 7 4 5 4 3 4 5 4 ...
##  $ Q36                : num  4 3 6 4 7 6 4 5 3 4 ...
##  $ Q37                : num  4 4 5 3 7 4 3 5 7 4 ...
##  $ Q38                : num  3 7 7 3 7 2 5 4 4 5 ...
##  $ Q39                : num  5 4 3 6 2 3 3 6 5 4 ...
##  $ Q40                : num  3 3 2 2 1 3 2 4 5 4 ...
##  $ Q41                : num  5 7 1 4 3 6 7 7 1 4 ...
##  $ Q42                : num  5 6 1 5 2 4 5 4 2 3 ...
##  $ Q43                : num  4 4 1 4 2 3 5 4 3 4 ...
##  $ Q44                : num  3 7 1 2 3 7 6 7 3 3 ...
##  $ Q45                : num  4 6 4 4 4 6 7 7 5 4 ...
##  $ Q46                : num  4 6 3 5 5 6 7 7 5 4 ...
##  $ Q47                : num  4 7 4 4 5 7 6 6 2 4 ...
##  $ Q48                : num  5 6 4 3 2 7 6 7 5 4 ...
##  $ Q49                : num  4 6 3 3 5 6 6 6 3 5 ...
##  $ Q50                : num  4 7 2 4 3 7 6 6 4 4 ...
##  $ Q51                : num  5 1 4 5 4 2 1 1 3 6 ...
##  $ Q52                : num  4 1 4 2 4 2 2 1 7 5 ...
##  $ Q53                : num  2 1 3 3 6 1 2 1 6 5 ...
##  $ Q54                : num  4 1 5 5 4 2 1 2 4 3 ...
##  $ Q55                : num  5 1 6 4 5 1 1 1 3 5 ...
##  $ Q56                : num  4 1 3 4 5 2 1 1 4 5 ...
##  $ Q57                : num  5 5 4 4 4 5 4 5 6 3 ...
##  $ Q58                : num  3 4 4 2 5 4 5 5 7 4 ...
##  $ Q59                : num  4 3 5 5 4 4 4 4 6 4 ...
##  $ Q60                : num  4 5 3 5 3 4 3 6 2 3 ...
##  $ Q61                : num  4 4 4 5 4 4 5 4 2 3 ...
##  $ Q62                : num  2 5 4 3 5 4 4 4 2 4 ...
```

3.2 Exploratory Analysis of the Data

`summary` is a good first step to get an overall sense of what the data looks like by examining selected percentiles and mean. A quick examination of the data shows that the psychographic questions range between 1 and

7, with most having a median response of 4. Most of the demographic data is categorical, so we need to understand the coding to learn the meaning. For example, IncomeCategory of 1 represents <100k, 2 represents 100k-150k, . . . and 6 is >300k.

```r
# descriptive statistics for all the variables
summary(ford)
```

```
##  PreferenceGroup     Gender          Age        MaritalStatus
##  Min.   :1.000   Min.   :1.00   Min.   :20.00   Min.   :1.000
##  1st Qu.:1.000   1st Qu.:1.00   1st Qu.:29.00   1st Qu.:1.000
##  Median :2.000   Median :1.00   Median :36.00   Median :1.000
##  Mean   :1.784   Mean   :1.48   Mean   :36.36   Mean   :1.872
##  3rd Qu.:2.000   3rd Qu.:2.00   3rd Qu.:43.00   3rd Qu.:3.000
##  Max.   :3.000   Max.   :2.00   Max.   :58.00   Max.   :3.000
##  NumberChildren  FirstTimePurchase  AgeCategory    ChildrenCategory
##  Min.   :0.000   Min.   :1.000     Min.   :1.000   Min.   :0.000
##  1st Qu.:0.000   1st Qu.:2.000     1st Qu.:2.000   1st Qu.:0.000
##  Median :0.000   Median :2.000     Median :4.000   Median :0.000
##  Mean   :0.728   Mean   :1.852     Mean   :3.768   Mean   :0.624
##  3rd Qu.:1.000   3rd Qu.:2.000     3rd Qu.:5.000   3rd Qu.:1.000
##  Max.   :4.000   Max.   :2.000     Max.   :6.000   Max.   :2.000
##  IncomeCategory       Q1            Q2             Q3
##  Min.   :1.00   Min.   :1.0   Min.   :1.00   Min.   :1.000
##  1st Qu.:2.00   1st Qu.:4.0   1st Qu.:2.00   1st Qu.:4.000
##  Median :4.00   Median :5.0   Median :4.00   Median :4.000
##  Mean   :3.68   Mean   :5.1   Mean   :4.06   Mean   :4.444
##  3rd Qu.:5.00   3rd Qu.:6.0   3rd Qu.:6.00   3rd Qu.:5.000
##  Max.   :6.00   Max.   :7.0   Max.   :7.00   Max.   :7.000
##        Q4              Q5              Q6              Q7
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :2.00
##  1st Qu.:3.000   1st Qu.:2.000   1st Qu.:3.000   1st Qu.:3.00
##  Median :4.000   Median :4.000   Median :4.000   Median :4.00
##  Mean   :4.236   Mean   :3.848   Mean   :3.992   Mean   :3.88
##  3rd Qu.:5.000   3rd Qu.:5.000   3rd Qu.:5.000   3rd Qu.:5.00
##  Max.   :7.000   Max.   :7.000   Max.   :7.000   Max.   :6.00
##        Q8              Q9             Q10             Q11
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :2.000
##  1st Qu.:3.000   1st Qu.:3.000   1st Qu.:3.000   1st Qu.:3.000
##  Median :4.000   Median :4.000   Median :4.000   Median :4.000
##  Mean   :3.916   Mean   :3.904   Mean   :3.916   Mean   :3.984
##  3rd Qu.:5.000   3rd Qu.:5.000   3rd Qu.:5.000   3rd Qu.:5.000
##  Max.   :7.000   Max.   :7.000   Max.   :7.000   Max.   :7.000
##       Q12             Q13             Q14             Q15
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :2.000
##  1st Qu.:3.000   1st Qu.:3.000   1st Qu.:2.000   1st Qu.:4.000
##  Median :4.000   Median :4.000   Median :5.000   Median :5.000
##  Mean   :4.072   Mean   :3.988   Mean   :4.132   Mean   :4.972
##  3rd Qu.:5.000   3rd Qu.:5.000   3rd Qu.:6.000   3rd Qu.:6.000
##  Max.   :7.000   Max.   :6.000   Max.   :7.000   Max.   :7.000
##       Q16             Q17             Q18             Q19
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.000
##  1st Qu.:3.000   1st Qu.:3.000   1st Qu.:4.000   1st Qu.:4.000
##  Median :5.000   Median :5.000   Median :5.000   Median :5.000
##  Mean   :4.512   Mean   :4.444   Mean   :4.532   Mean   :4.688
##  3rd Qu.:6.000   3rd Qu.:6.000   3rd Qu.:5.750   3rd Qu.:6.000
##  Max.   :7.000   Max.   :7.000   Max.   :7.000   Max.   :7.000  
##       Q20             Q21             Q22             Q23      
##  Min.   :1.000   Min.   :2.000   Min.   :1.000   Min.   :1.00  
##  1st Qu.:2.000   1st Qu.:4.000   1st Qu.:4.000   1st Qu.:3.00  
##  Median :4.000   Median :5.000   Median :5.000   Median :4.00  
##  Mean   :3.832   Mean   :4.912   Mean   :4.992   Mean   :4.12  
##  3rd Qu.:5.000   3rd Qu.:6.000   3rd Qu.:6.000   3rd Qu.:6.00  
##  Max.   :7.000   Max.   :7.000   Max.   :7.000   Max.   :7.00  
##       Q24             Q25             Q26             Q27      
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.00  
##  1st Qu.:1.000   1st Qu.:2.000   1st Qu.:2.000   1st Qu.:2.00  
##  Median :2.000   Median :3.000   Median :3.000   Median :4.00  
##  Mean   :2.376   Mean   :3.148   Mean   :3.012   Mean   :3.46  
##  3rd Qu.:3.000   3rd Qu.:4.000   3rd Qu.:4.000   3rd Qu.:4.00  
##  Max.   :6.000   Max.   :7.000   Max.   :7.000   Max.   :7.00  
##       Q28            Q29             Q30             Q31       
##  Min.   :1.00   Min.   :1.000   Min.   :1.000   Min.   :1.000  
##  1st Qu.:2.00   1st Qu.:3.000   1st Qu.:2.000   1st Qu.:2.000  
##  Median :3.00   Median :3.000   Median :3.000   Median :4.000  
##  Mean   :3.12   Mean   :3.448   Mean   :3.344   Mean   :4.056  
##  3rd Qu.:4.00   3rd Qu.:4.000   3rd Qu.:4.000   3rd Qu.:6.000  
##  Max.   :7.00   Max.   :7.000   Max.   :6.000   Max.   :7.000  
##       Q32             Q33             Q34             Q35       
##  Min.   :2.000   Min.   :1.000   Min.   :1.000   Min.   :1.000  
##  1st Qu.:4.000   1st Qu.:4.000   1st Qu.:4.000   1st Qu.:4.000  
##  Median :5.000   Median :5.000   Median :5.000   Median :5.000  
##  Mean   :4.604   Mean   :4.564   Mean   :4.496   Mean   :4.584  
##  3rd Qu.:6.000   3rd Qu.:6.000   3rd Qu.:5.000   3rd Qu.:6.000  
##  Max.   :7.000   Max.   :7.000   Max.   :7.000   Max.   :7.000  
##       Q36             Q37             Q38             Q39       
##  Min.   :2.000   Min.   :1.000   Min.   :2.000   Min.   :1.000  
##  1st Qu.:4.000   1st Qu.:4.000   1st Qu.:4.000   1st Qu.:2.000  
##  Median :4.000   Median :5.000   Median :5.000   Median :4.000  
##  Mean   :4.452   Mean   :4.836   Mean   :4.616   Mean   :3.444  
##  3rd Qu.:5.000   3rd Qu.:6.000   3rd Qu.:6.000   3rd Qu.:4.000  
##  Max.   :7.000   Max.   :7.000   Max.   :7.000   Max.   :7.000  
##       Q40             Q41             Q42             Q43       
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.000  
##  1st Qu.:2.250   1st Qu.:2.000   1st Qu.:2.000   1st Qu.:2.000  
##  Median :3.000   Median :4.000   Median :3.000   Median :3.000  
##  Mean   :3.368   Mean   :3.912   Mean   :3.148   Mean   :3.392  
##  3rd Qu.:4.000   3rd Qu.:6.000   3rd Qu.:4.000   3rd Qu.:4.000  
##  Max.   :7.000   Max.   :7.000   Max.   :7.000   Max.   :7.000  
##       Q44            Q45             Q46             Q47       
##  Min.   :1.00   Min.   :1.000   Min.   :1.000   Min.   :1.000  
##  1st Qu.:3.00   1st Qu.:4.000   1st Qu.:4.000   1st Qu.:3.250  
##  Median :4.00   Median :5.000   Median :5.000   Median :5.000  
##  Mean   :4.26   Mean   :4.744   Mean   :4.752   Mean   :4.768  
##  3rd Qu.:6.00   3rd Qu.:6.000   3rd Qu.:6.000   3rd Qu.:6.000  
##  Max.   :7.00   Max.   :7.000   Max.   :7.000   Max.   :7.000  
##       Q48             Q49             Q50             Q51       
##  Min.   :1.000   Min.   :2.000   Min.   :1.000   Min.   :1.000  
##  1st Qu.:4.000   1st Qu.:4.000   1st Qu.:4.000   1st Qu.:2.000  
##  Median :5.000   Median :5.000   Median :5.000   Median :4.000  
##  Mean   :4.776   Mean   :4.776   Mean   :4.812   Mean   :3.308  
##  3rd Qu.:6.000   3rd Qu.:6.000   3rd Qu.:6.000   3rd Qu.:4.000  
##  Max.   :7.000   Max.   :7.000   Max.   :7.000   Max.   :7.000  
##       Q52             Q53             Q54            Q55       
##  Min.   :1.000   Min.   :1.000   Min.   :1.00   Min.   :1.000  
##  1st Qu.:2.000   1st Qu.:2.000   1st Qu.:2.00   1st Qu.:2.000  
##  Median :4.000   Median :4.000   Median :3.00   Median :3.000  
##  Mean   :3.532   Mean   :3.616   Mean   :3.16   Mean   :3.136  
##  3rd Qu.:5.000   3rd Qu.:5.000   3rd Qu.:4.00   3rd Qu.:4.000  
##  Max.   :7.000   Max.   :7.000   Max.   :7.00   Max.   :6.000  
##       Q56             Q57             Q58             Q59      
##  Min.   :1.000   Min.   :1.000   Min.   :2.000   Min.   :1.00  
##  1st Qu.:2.000   1st Qu.:3.000   1st Qu.:4.000   1st Qu.:3.00  
##  Median :3.000   Median :4.000   Median :4.000   Median :4.00  
##  Mean   :3.148   Mean   :4.316   Mean   :4.384   Mean   :4.32  
##  3rd Qu.:4.000   3rd Qu.:5.000   3rd Qu.:5.000   3rd Qu.:5.00  
##  Max.   :7.000   Max.   :7.000   Max.   :7.000   Max.   :7.00  
##       Q60             Q61            Q62       
##  Min.   :1.000   Min.   :1.00   Min.   :1.000  
##  1st Qu.:3.000   1st Qu.:3.00   1st Qu.:3.000  
##  Median :4.000   Median :4.00   Median :4.000  
##  Mean   :3.772   Mean   :3.68   Mean   :3.672  
##  3rd Qu.:5.000   3rd Qu.:5.00   3rd Qu.:5.000  
##  Max.   :7.000   Max.   :7.00   Max.   :7.000  
```

```
# to print an individual variable, enter it by itself
ford$Age
```

```
##   [1] 44 24 34 44 41 26 33 48 32 34 55 43 42 20 36 45 27 33 35 20 42 41 54
##  [24] 30 48 56 42 41 32 43 41 47 37 32 30 25 54 32 26 26 42 27 26 24 48 43
##  [47] 23 42 40 32 25 41 44 31 47 27 34 40 44 23 30 35 41 48 41 40 40 47 30
##  [70] 33 24 39 42 41 32 26 50 35 20 34 44 34 36 33 55 46 32 30 28 31 32 29
##  [93] 40 33 57 26 27 46 38 44 40 27 41 37 34 28 44 46 30 29 37 44 41 41 51
## [116] 26 28 58 35 28 44 45 51 27 21 23 31 38 34 43 31 37 42 27 40 29 22 25
## [139] 47 43 39 27 55 39 40 42 39 48 39 26 37 31 42 31 24 35 29 48 51 43 37
## [162] 40 30 27 26 48 37 44 30 40 20 51 32 40 26 29 34 28 55 30 42 21 45 20
## [185] 44 38 26 33 30 40 49 32 31 40 20 50 22 29 35 26 36 41 30 28 49 32 52
## [208] 43 43 43 24 28 35 58 39 29 48 43 29 49 43 39 49 23 39 47 41 31 40 22
## [231] 20 39 27 56 20 36 29 42 24 25 34 43 22 26 36 52 42 34 34 43
```

3.2.1 Frequency Tabulations

The next step in our exploratory analysis is to look more carefully at the frequency of each value through a cross-tabulation using `xtabs`. Cross-tabs are especially helpful when data is categorical. For example, there are 250 observations and for ChildrenCategory there are 148 with no children (0), 48 with one child (1), and 54 with two or more children (2).

```
# create tables to describe the data
xtabs(~Age,data=ford)
```

```
## Age
## 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 
##  8  2  4  4  6  4 13 10  7  9 11  8 11  6 11  7  5  7  3  9 14 13 12 13 11 
## 45 46 47 48 49 50 51 52 54 55 56 57 58 
##  3  3  5  8  4  2  4  2  2  4  2  1  2
```

```
xtabs(~AgeCategory,data=ford)
```

```
## AgeCategory
##  1  2  3  4  5  6
## 24 43 47 31 63 42
```

```
xtabs(~ChildrenCategory,data=ford)
```

```
## ChildrenCategory
##   0   1   2
## 148  48  54
```

```
xtabs(~FirstTimePurchase,data=ford)
```

```
## FirstTimePurchase
##   1   2
##  37 213
```

```
xtabs(~Gender,data=ford)
```

```
## Gender
##   1   2
## 130 120
```

```
xtabs(~IncomeCategory,data=ford)
```

```
## IncomeCategory
##  1  2  3  4  5  6
## 23 46 46 46 51 38
```

```
xtabs(~MaritalStatus,data=ford)
```

```
## MaritalStatus
##   1   2   3
## 127  28  95
```

```
xtabs(~NumberChildren,data=ford)
```

```
## NumberChildren
##   0   1   2   3   4
## 148  48  31  20   3
```

```
xtabs(~PreferenceGroup,data=ford)
```

```
## PreferenceGroup
##   1   2   3
## 116  72  62
```

3.2.2 Understanding relationships with Cross Tabulations

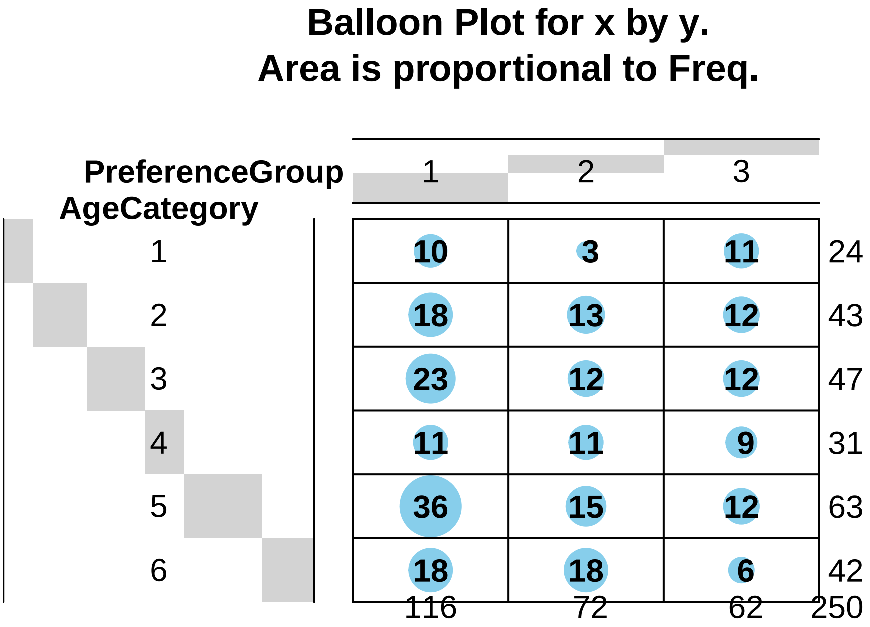
To understand the relationship between the variables we can look at the cross-tabulation between two variables. The following cross-tabulation between PreferenceGroup and AgeCategory shows the relationship between these variables, or how is age related to preferences. If we look at the first age category of 1 we see that 10 and 11 people either where in the top or middle of the PreferenceGroup (remember that 1 is the top and 3 is the middle), while only 3 were negative about the Ka. On the other hand 18 versus 24 were quite negative about the Ford Ka for the 44 and over age category. This suggests that younger individuals are most positive about the Ford Ka.

```
# to see the relationship between two variables do a cross-tab
xtabs(~PreferenceGroup+AgeCategory,data=ford)
```

```
##                  AgeCategory
## PreferenceGroup  1  2  3  4  5  6
##                1 10 18 23 11 36 18
##                2  3 13 12 11 15 18
##                3 11 12 12  9 12  6
```

The visual representation of the table is with the BalloonPlot below. Again notice that the size of the circles that represent those that do not like the Ford Ka (PreferenceGroup=2) tends to increase as age increases.

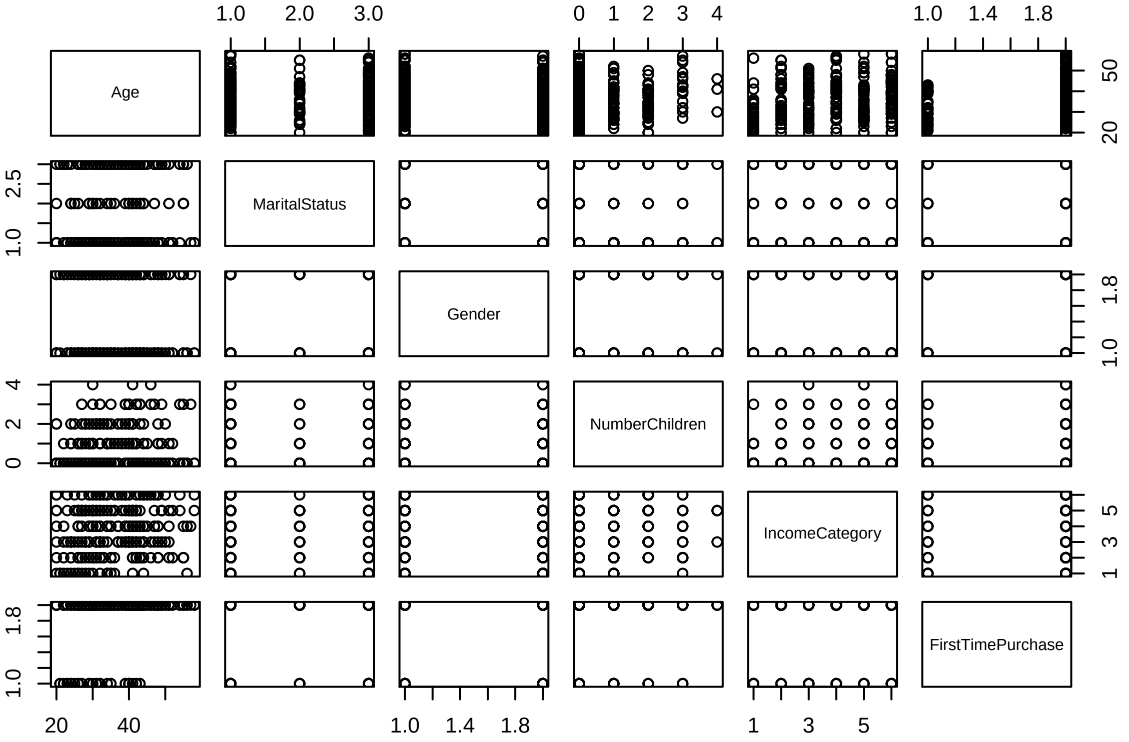
```
# here is a better visual representation of a table with a BalloonPlot (uncomment the line if you want
balloonplot(table(ford$PreferenceGroup,ford$AgeCategory),xlab="PreferenceGroup",ylab="AgeCategory")
```



3.3 Visualizing relationships with scatterplot matrices

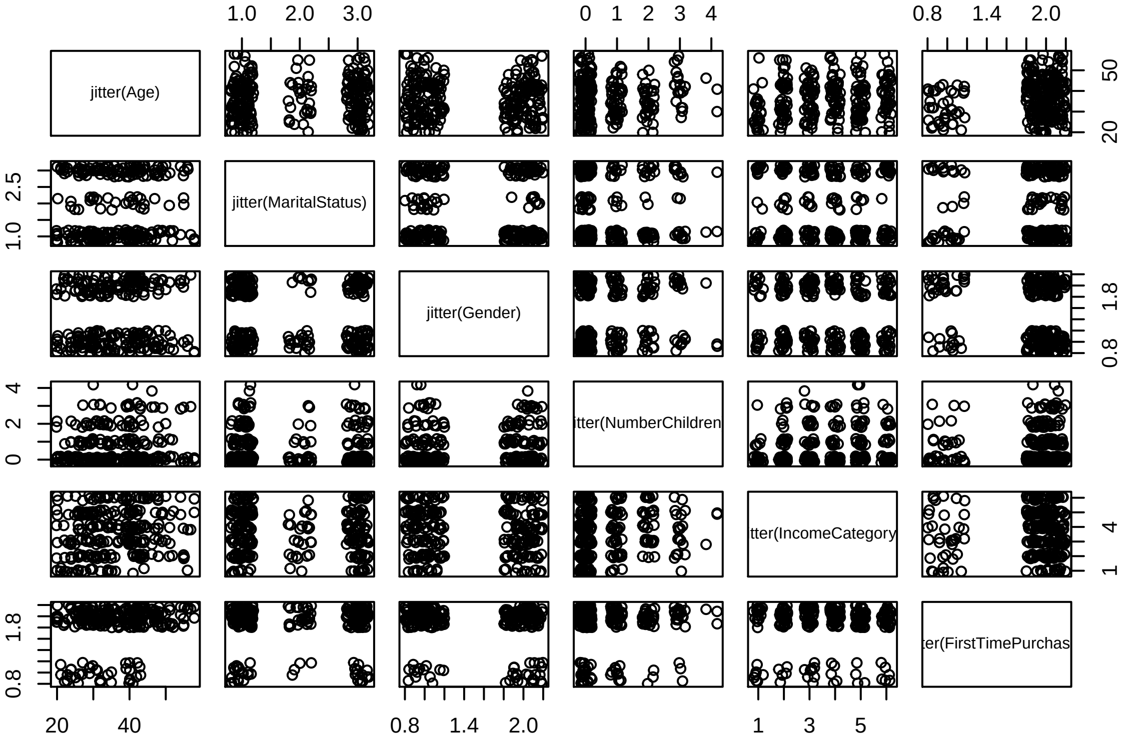
The last illustration is a scatterplot of each of the demographic variables against one another. This is helpful in trying to see relationships amongst the variables.

```
# let's plot all pairs of data in a matrix plot
pairs(~Age+MaritalStatus+Gender+NumberChildren+IncomeCategory+FirstTimePurchase,data=ford)
```



Unfortunately in the previous scatterplot is a bit difficult to see the relationships since most of the observations are plotted on top of one another. For example, Gender versus FirstTimePurchase only has four dots, because there are only two values for each. An alternative is to add a small amount of random variation, so instead of all values of Male being 1 some will be 0.99 or 1.01 or 1.02, using the `jitter` function. For example, the Age versus NumberChildren shows that older people tend to have more children.

```
pairs(~jitter(Age)+jitter(MaritalStatus)+jitter(Gender)+jitter(NumberChildren)
      +jitter(IncomeCategory)+jitter(FirstTimePurchase),data=ford)
```



4 Determining the Number of Clusters for k-Means

4.1 Computing many alternative cluster solutions

Our next analysis is to compute a large number of k-Means solutions with k varying from 2, 3, . . . , to 30. The goal is to get a sense of which value of k gives us a good tradeoff between explaining most of the variation without having too many clusters.

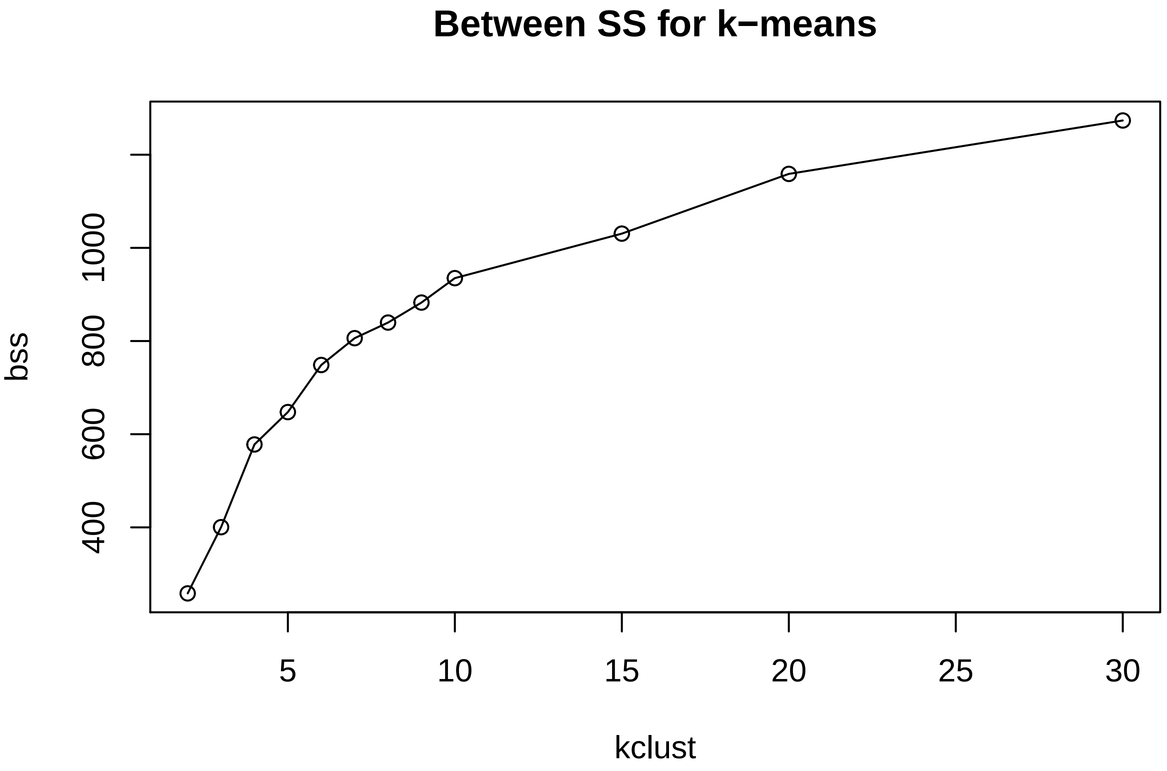
```
# compute multiple cluster solutions
grpA2=kmeans(xford[,dlist],centers=2)
grpA3=kmeans(xford[,dlist],centers=3)
grpA4=kmeans(xford[,dlist],centers=4)
grpA5=kmeans(xford[,dlist],centers=5)
grpA6=kmeans(xford[,dlist],centers=6)
grpA7=kmeans(xford[,dlist],centers=7)
grpA8=kmeans(xford[,dlist],centers=8)
grpA9=kmeans(xford[,dlist],centers=9)
grpA10=kmeans(xford[,dlist],centers=10)
grpA15=kmeans(xford[,dlist],centers=15)
grpA20=kmeans(xford[,dlist],centers=20)
grpA30=kmeans(xford[,dlist],centers=30)

# compute between and within SS
kclust=c(2:10,15,20,30)
bss=c(grpA2$betweenss,
      grpA3$betweenss,grpA4$betweenss,grpA5$betweenss,grpA6$betweenss,
      grpA7$betweenss,grpA8$betweenss,grpA9$betweenss,grpA10$betweenss,
      grpA15$betweenss,grpA20$betweenss,grpA30$betweenss)
wss=c(grpA2$tot.withinss,
      grpA3$tot.withinss,grpA4$tot.withinss,grpA5$tot.withinss,grpA6$tot.withinss,
      grpA7$tot.withinss,grpA8$tot.withinss,grpA9$tot.withinss,grpA10$tot.withinss,
      grpA15$tot.withinss,grpA20$tot.withinss,grpA30$tot.withinss)
```

4.2 Finding the “Right” number of clusters

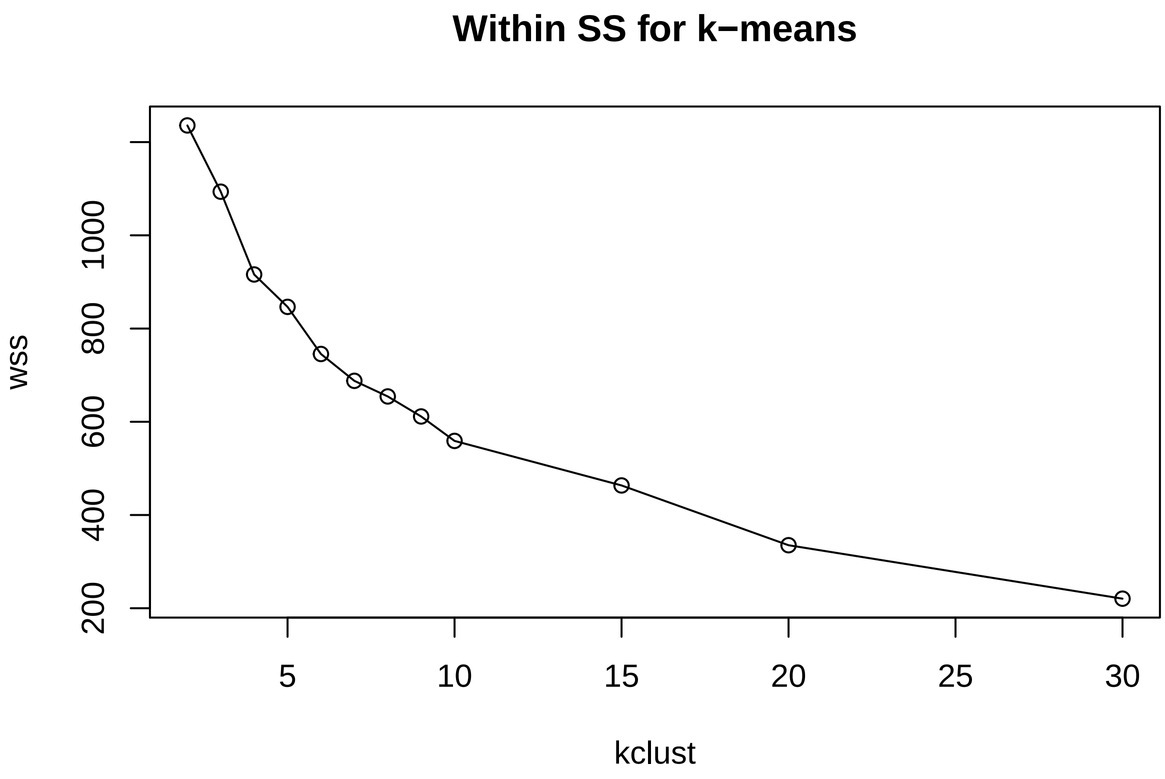
There is no *correct* solution since this is an unsupervised learning exercise. One way to discern a good solution is to look at the rate that the within sum-of-squares decreases as the value of k is increased. The within sum-of-squares (WSS) represents the cohesion of the solution, or in plain words it represents how much variation is explained by the cluster. The between sum-of-squares (BSS) illustrates the separation in the cluster solution, or in plain words how much variation shows up across the cluster solutions. If most of the variation is within the cluster than that suggests that the cluster is not made up of observations that all look similar to one another. Our preference is to have solutions with very little within sum-of-squares since it gives solutions that are cohesive or in plain words all the observations seem similar to one another.

The following scree plots the within sum-of-squares against the value of k. We can locate a good solution by finding a kink in the graph. Specifically we are looking for a value of k that is small but gives us enough separation amongst the cluster solutions. This analysis suggests that there are a range of values between 4 and 9 that explain most of the variation. It seems like any of these values would give a good solution. Notice that there is a small increase around k=8 which might seem counter-intuitive, that a larger value of k gives a worse solution, but this is due the the random starting points that are used and does happen occassionally.

```
# plot the results and look for the "Hockey-Stick" effect
par(mfrow=c(1,1))
plot(kclust,bss,type="l",main="Between SS for k-means")
points(kclust,bss)
```



```
plot(kclust,wss,type="l",main="Within SS for k-means")
points(kclust,wss)
```



```
#plot(kclust,bss/(wss+bss),type="l",main="R-Squared for k-means")
#points(kclust,bss/(wss+bss))
```

5 k-Means Analysis of Demographics

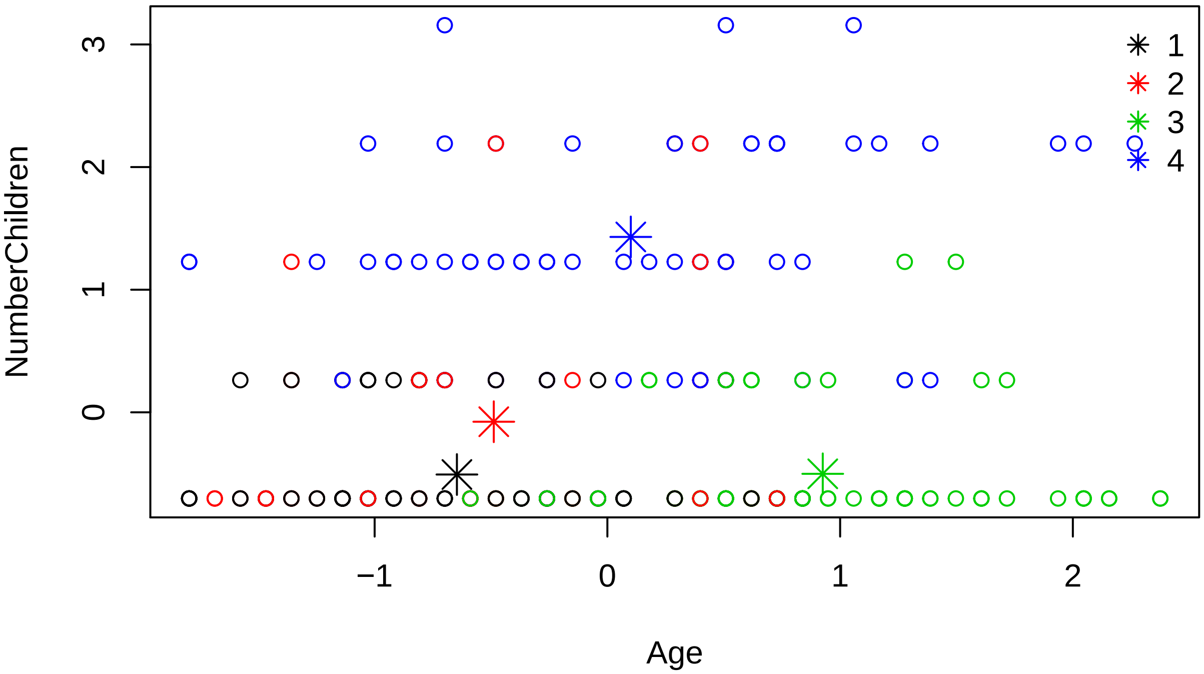
In this final step we choose a value of k=5. Again any solution between 4 and 8 would be supported from the scree plot, but 4 appears to be closest to the kink. The solution is given below. The `kmeans` algorithm returns a list as its output which includes the centers or means of each of the clusters (`grpB$centers`) and the solutions or integer values of the cluster that is associated with each cluster (`grpB$cluster`).

```
# set the random number seed so the samples will be the same if regenerated
set.seed(1248765792)

# compute a k-means cluster with k=4 using just demographic variables
k=4  # save the number of clusters to a variable
(grpB=kmeans(xford[,dlist],centers=k))  #!! change the =4 to whatever value you decide from part 2 !!
```

```
## K-means clustering with 4 clusters of sizes 84, 37, 72, 57
##
## Cluster means:
##          Age MaritalStatus      Gender NumberChildren IncomeCategory
## 1 -0.6465077  -0.143140686 -0.03139077     -0.5071140     -0.5011037
## 2 -0.4880897  -0.007627273  0.33689165     -0.0765577     -0.2264270
## 3  0.9253953   0.359531531 -0.40395804     -0.5013712      0.4689700
## 4  0.1006579  -0.238249886  0.33783882      1.4303323      0.2930644
##   FirstTimePurchase
## 1         0.4159496
## 2        -2.3945208
## 3         0.4159496
## 4         0.4159496
##
## Clustering vector:
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##   3   2   1   3   2   2   3   3   4   1   3   3   1   1   1   3   1   1
##  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
##   1   1   3   4   3   4   3   3   3   4   2   4   1   3   3   1   1   2
##  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
##   4   4   1   1   3   4   1   1   3   4   1   3   4   3   1   1   1   1
##  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
##   3   1   1   1   3   1   4   3   3   3   3   4   2   3   1   4   1   2
##  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
##   3   1   4   1   3   4   4   2   4   1   1   4   3   3   2   4   1   1
##  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108
##   2   2   4   1   4   2   2   4   4   4   2   1   1   1   4   1   3   4
## 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126
##   1   4   1   3   4   3   3   1   1   3   2   4   1   3   3   4   2   2
## 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144
##   1   3   4   3   4   4   2   1   2   1   2   1   4   3   4   1   3   3
## 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162
##   4   2   3   3   4   1   4   2   4   4   2   2   1   3   3   1   1   2
## 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
##   1   1   1   3   1   3   1   3   1   3   1   3   1   1   4   4   4   1
## 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198
##   3   2   3   1   3   3   1   1   4   3   4   4   2   4   4   3   1   1
## 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216
##   1   4   3   3   2   1   3   1   3   2   4   3   2   1   4   3   2   2
## 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234
##   4   3   2   4   4   4   3   2   3   3   3   3   2   1   1   1   2   3
## 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250
##   1   1   1   4   1   4   1   1   1   1   3   3   1   1   3   2
##
## Within cluster sum of squares by cluster:
## [1] 278.0527 165.4263 231.3111 226.9964
##  (between_SS / total_SS =  39.6 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

The following analysis is meant to give a visualization of the data and the cluster solution by plotting the scatter plot of two selected variables, NumberChildren versus Age, along with the centroids which are illustrated by asterisks. It appears that cluster 4 are those with large families, while cluster 1 and 2 represent younger people with no families, and cluster 3 are older people without children. This explanation is not perfect, since there are some in cluster 2 that have children, but there tend to be fewer of those individuals.

```r
# plot the solutions against the Age and NumberChildren
# since the data is categorical most of the plots will overlay one another,
# so instead we jitter the points -- which adds a small random number to each
par(mfrow=c(1,1),mar=c(5,4,4,1)+.1)
plot(xford[,"Age"],xford[,"NumberChildren"],xlab="Age",ylab="NumberChildren",col=grpB$cluster)
points(grpB$centers[,c("Age","NumberChildren")],col=1:k,pch=8,cex=2)
legend("topright",pch=8,bty="n",col=1:k,as.character(1:k))
```

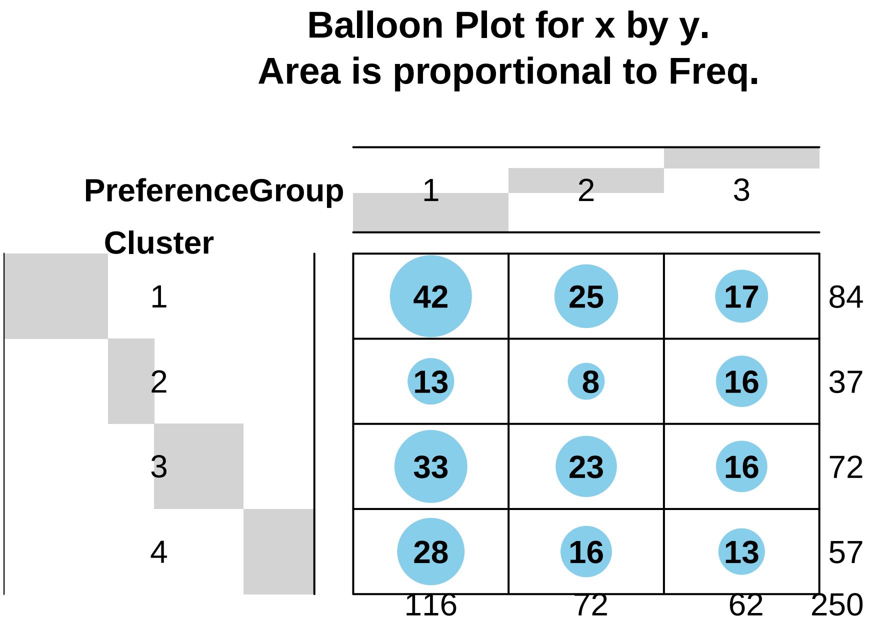


To check the external validity of the cluster solution we need to compare the clusters with a measure, variable or result that is outside of the data. In this case we can compare the cluster solutions to the preference groups. Remember that preference groups correspond to Ford Ka choosers (PreferenceGroup=1), those that are in the middle (PreferenceGroup=3), and those that do not like the Ka (PreferenceGroup=2). We find that the largest cluster and also the one with the highest relative number of people who like the Ford Ka are in Cluster=1.

```
# compare the cluster solutions with the PreferenceGroup
xtabs(~ford$PreferenceGroup+grpB$cluster)
```

```
##                     grpB$cluster
## ford$PreferenceGroup  1  2  3  4
##                    1 42 13 33 28
##                    2 25  8 23 16
##                    3 17 16 16 13
```

```
# here is a more visualize representation of a table with a BalloonPlot
balloonplot(table(ford$PreferenceGroup,grpB$cluster),xlab="PreferenceGroup",ylab="Cluster")
```



The final table gives the centers or centroids that are associated with each cluster. In other words we have the averages of the variables within each cluster. For example, the average of Age in cluster 3 is 0.925. Remember that age has been standardized. The original mean of Age was 36.36 and standard deviation was 9.11. However, we have standardized the data in xford, and the mean of Age is now 0 and the standard deviation is 1. The mean of Age in cluster 3 of 0.925 shows that the average Age within this cluster is almost one over the original standard deviation.

```
# summarize the centroids
grpBcenter=t(grpB$centers)   # create variable with the transpose of the centroids
rownames(grpBcenter)=dlist   # add the variable names
print(grpBcenter)   # print the centroid values for each question
```

```
##                             1            2          3          4
## Age                -0.64650765 -0.488089659  0.9253953  0.1006579
## MaritalStatus      -0.14314069 -0.007627273  0.3595315 -0.2382499
## Gender             -0.03139077  0.336891648 -0.4039580  0.3378388
## NumberChildren     -0.50711403 -0.076557697 -0.5013712  1.4303323
## IncomeCategory     -0.50110372 -0.226427012  0.4689700  0.2930644
## FirstTimePurchase   0.41594962 -2.394520772  0.4159496  0.4159496
```

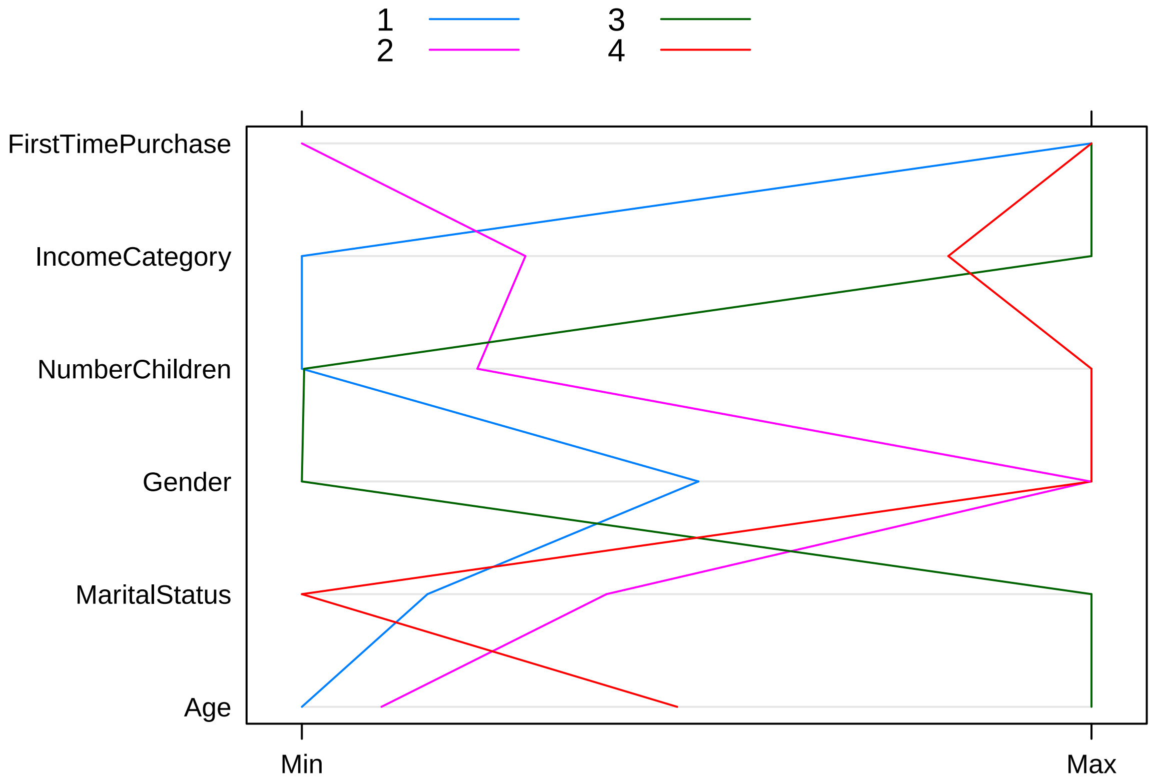
The `parallelplot` gives a graphical depiction of this matrix. Which allows us to attach the following meaning to each of the clusters by examining the corresponding average values. Specifically we suggest the following meaning and names with each of the clusters:

Cluster	Name	Description
1	Just Starting Out	This cluster is made up of those that are
2	New Buyers	This cluster is made up of those that are
3	Experienced	This cluster is made up of those that are relatively older, more likley to be men with smaller families, but higher incomes.

Cluster	Name	Description
4	Family	This cluster has more families (e.g., higher number of children). The other variables show slighlty more women, more married, more income, and older, but none of these variables is as high as the number of children.

Judging from the relationship between the PreferenceGroup and Cluster (see `xtabs(~ford$PreferenceGroup+grpB$cluster))` we would think the ‘Just Starting Out’ cluster would be the best choice to target since there are more people who like the Ford Ka than any of the other segments. This suggests that we should stress first time buyers that are younger, single and have higher incomes.

```
parallelplot(t(grpBcenter),auto.key=list(text=as.character(1:k),space="top",columns=3,lines=T))   # crea
```


FirstTimePurchase
IncomeCategory
NumberChildren
Gender
MaritalStatus
Age
Min
Max