Aya Alsadi

101115964

STAT 4601

Data Mining

Final Project

Due: Dec-28-2020

PART I. DATA Visualization

1. *The data set:*

The file audit.csv contains characteristics of 2000 individual tax returns. The dependent variables are (TARGET_Adjusted) and (RISK_Adjustment). The data set contains a total of 12 variables including: ID (Unique identifier for each person), Age (Age of person), Employment (Type of employment), Education (Highest level of education), Marital (Current marital status), Occupation (Type of occupation), Income (Amount of income declared), Gender (Gender of person), Deductions (Total amount of expenses that a person claims in their financial statement), Hours (Average hours worked on a weekly basis), Risk_Adjustment (monetary amount of any adjustment to the person's financial status as a result of a productive audit. This is a measure of the size of the risk associated with the person), TARGET_Adjusted (binary target variable for classification modeling (0/1), indicating non-productive and productive audits, respectively. Productive audits are those that result in an adjustment being made to a client's financial statement.)

The excel file was loaded from RStudio. the original data is shown below:

```
> library(readxl)
> AuditData <- read_excel("C:/Users/olivierM/Desktop/AuditData.xlsx")
> View (AuditData)
```

AuditData

Filter

	ID	Age	Employment	Education	Marital	Occupation	Income	Gen
1	1004641	38	Private	College	Unmarried	Service	81838.00	Fe
2	1010229	35	Private	Associate	Absent	Transport	72099.00	M
3	1024587	32	Private	HSgrad	Divorced	Clerical	154676.74	M
4	1038288	45	Private	Bachelor	Married	Repair	27743.82	M
5	1044221	60	Private	College	Married	Executive	7568.23	M
6	1047095	74	Private	HSgrad	Married	Service	33144.40	M
7	1047698	43	Private	Bachelor	Married	Executive	43391.17	M
8	1053888	35	Private	Yr12	Married	Machinist	59906.65	M
9	1061323	25	Private	Associate	Divorced	Clerical	126888.91	Fe
10	1062363	22	Private	HSgrad	Absent	Sales	52466.49	Fe
11	1068642	48	Private	College	Divorced	Service	291416.11	Fe

The data can be treated as a matrix. Since this is a large set (n=2000 units), it is not convenient to display all in the R window. Parts of the matrix are shown below:

```
> mdata=AuditData;
> mdata[1:5,1:7]
```

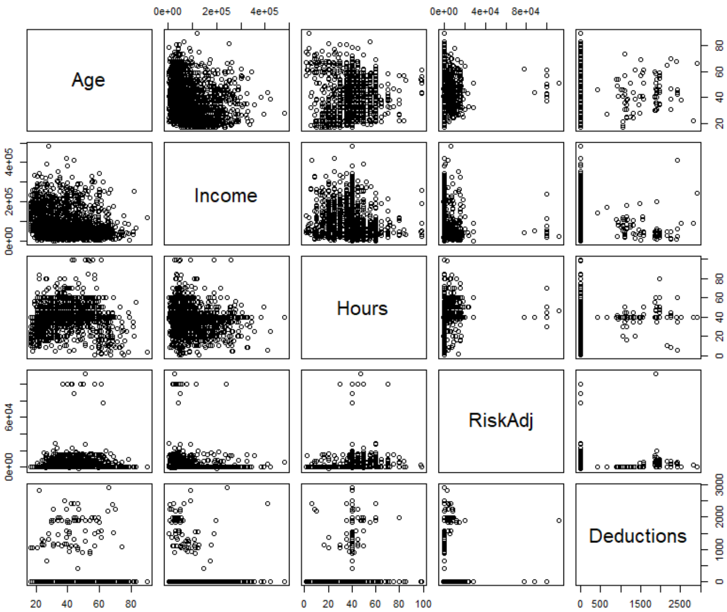
```
# A tibble: 5 x 7
       ID   Age Employment Education Marital   Occupation  Income
    <dbl> <dbl> <chr>      <chr>     <chr>     <chr>        <dbl>
1 1004641    38 Private    College   Unmarried Service      81838
2 1010229    35 Private    Associate Absent    Transport    72099
3 1024587    32 Private    HSgrad    Divorced  Clerical    154677.
4 1038288    45 Private    Bachelor  Married   Repair       27744.
5 1044221    60 Private    College   Married   Executive     7568.
```

```
> mdata[10:15,2:8]
# A tibble: 6 x 7
    Age Employment Education  Marital  Occupation  Income Gender
  <dbl> <chr>      <chr>      <chr>    <chr>        <dbl> <chr>
1    22 Private    HSgrad     Absent   Sales       52466. Female
2    48 Private    College    Divorced Service    291416. Female
3    60 Private    Vocational Widowed  Clerical    24155. Male
4    21 Private    College    Absent   Service    143255. Female
5    21 Private    College    Absent   Machinist  120555. Male
6    50 Private    Master     Married  Executive   34919. Male
```

2. A visual look into the data

The second command shows a matrix of scatters for five quantitative variables. Each scatter describes the degree of linear relationship between two variables.

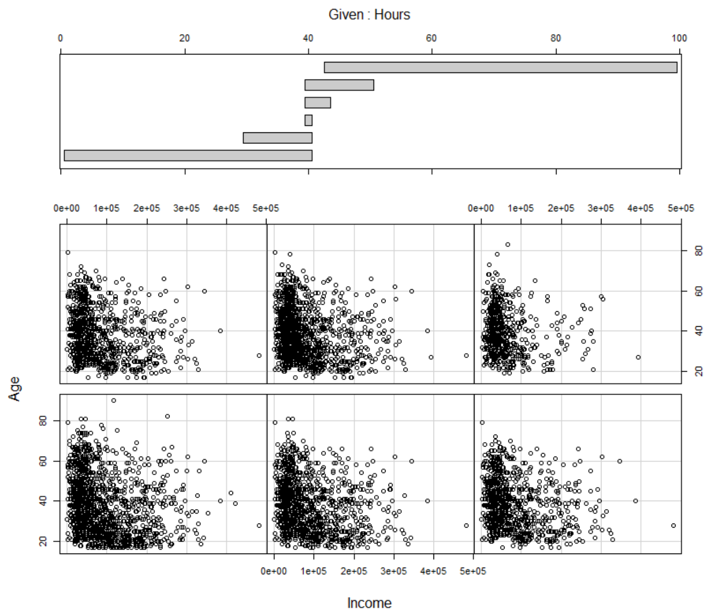
```
> mData2 = cbind (Age, Income, Hours, RiskAdj, Deductions);
> pairs(mData2)
```



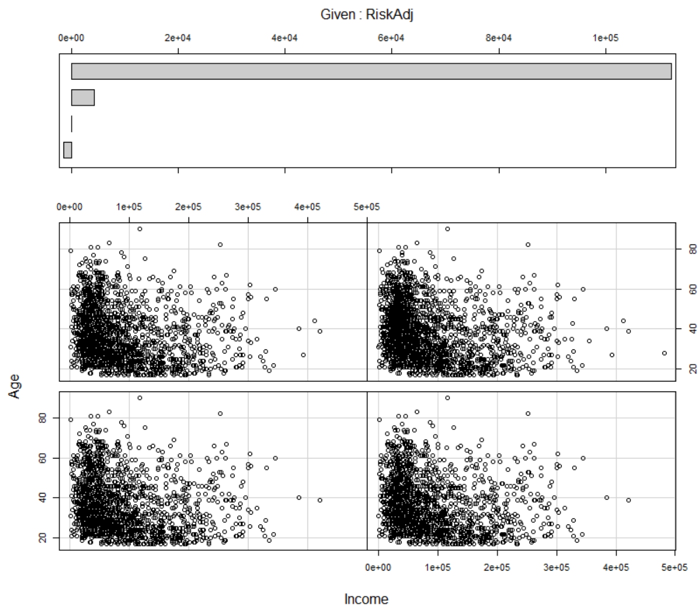
3. Conditional Plots

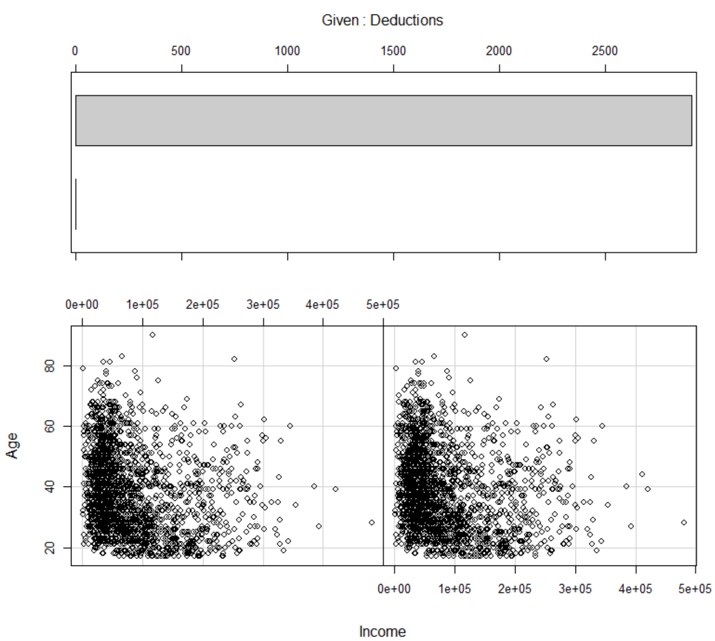
To investigate further some of the more complicated relationships we can look at conditional plotting. We present several plots corresponding to different ranges of the conditional variable. For instance, we plot "Age" against "Income", given "Hours". We also look at other conditional relationships' as shown below.

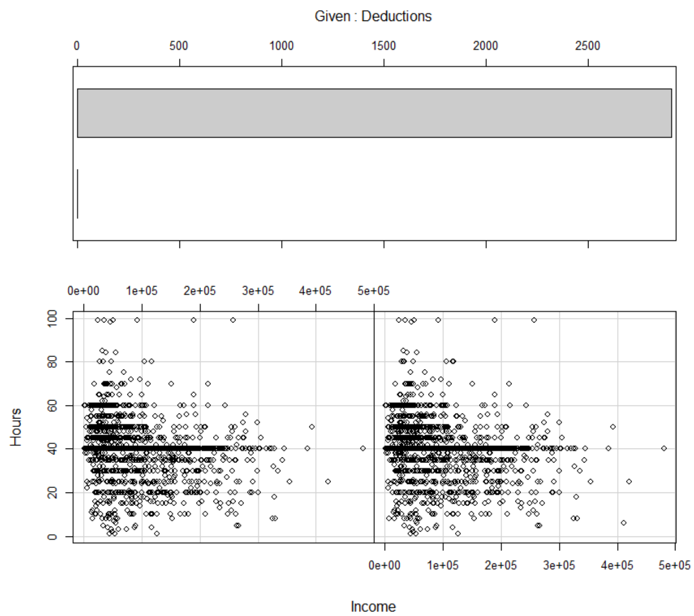
```
> coplot(Age~Income|Hours)
> coplot(Age~Income|RiskAdj)
> coplot(Age~Income|Deductions)
> coplot(Hours~Income|Deductions)
> coplot(Hours~Income|RiskAdj)
```

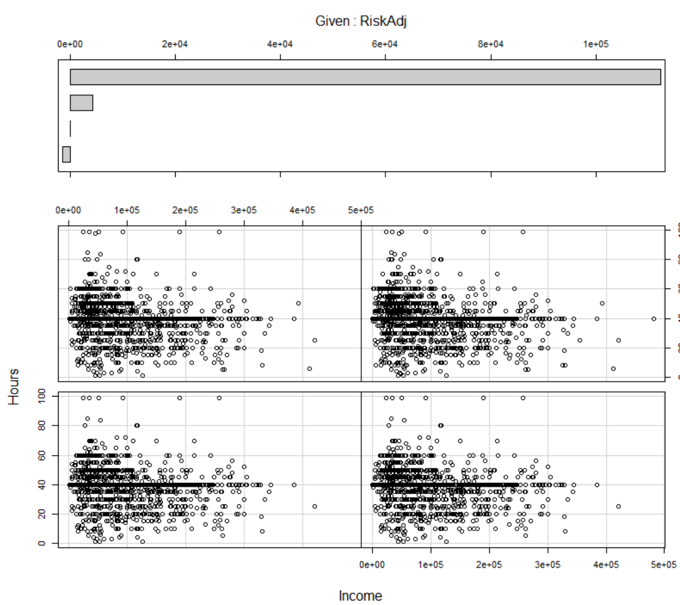


In both the figure above, the lower six panels show the pairwise plots for “Age” against “Income” for different ranges of “Hours” as shown in the upper panel. The default function selects 6 different subsets of the third variable “Hours” with an equal number of cases in each. In addition, an overlap of 0.5 is allowed.


Given : RiskAdj
0e+00
2e+04
4e+04
6e+04
8e+04
1e+05
0e+00
1e+05
2e+05
3e+05
4e+05
5e+05
Age
20
40
60
80
Income

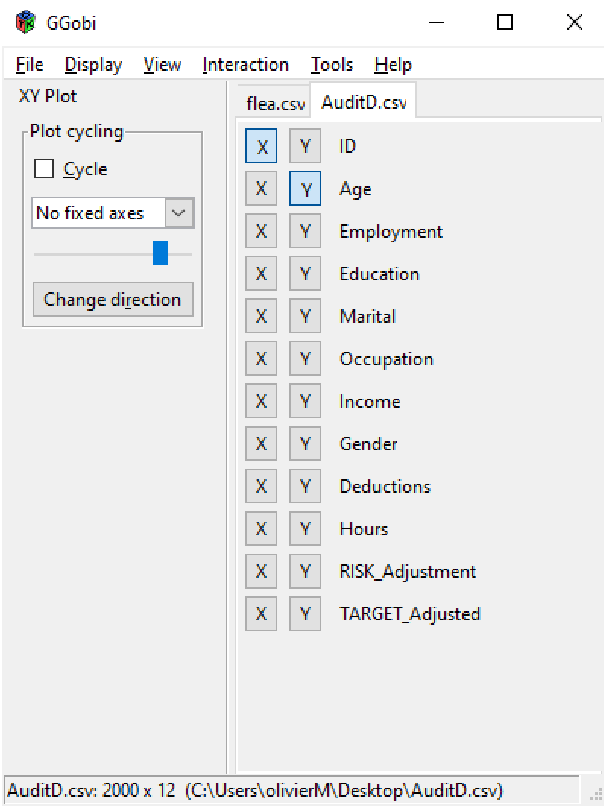

Given : Deductions
0
500
1000
1500
2000
2500
0e+00
1e+05
2e+05
3e+05
4e+05
5e+05
Age
20
40
60
80
Income


Given : Deductions
Hours
Income


Given : RiskAdj
Hours
Income

4. *Data Visualization in Ggobi*

We continue our investigation of the data using the program Ggobli. Data Visualization in Ggobi can be done thru the use of a package that may be called from R or used alone. In this work, we used Ggobli alone independently of the R program. The excel data Audit.csv was first loaded from the console.



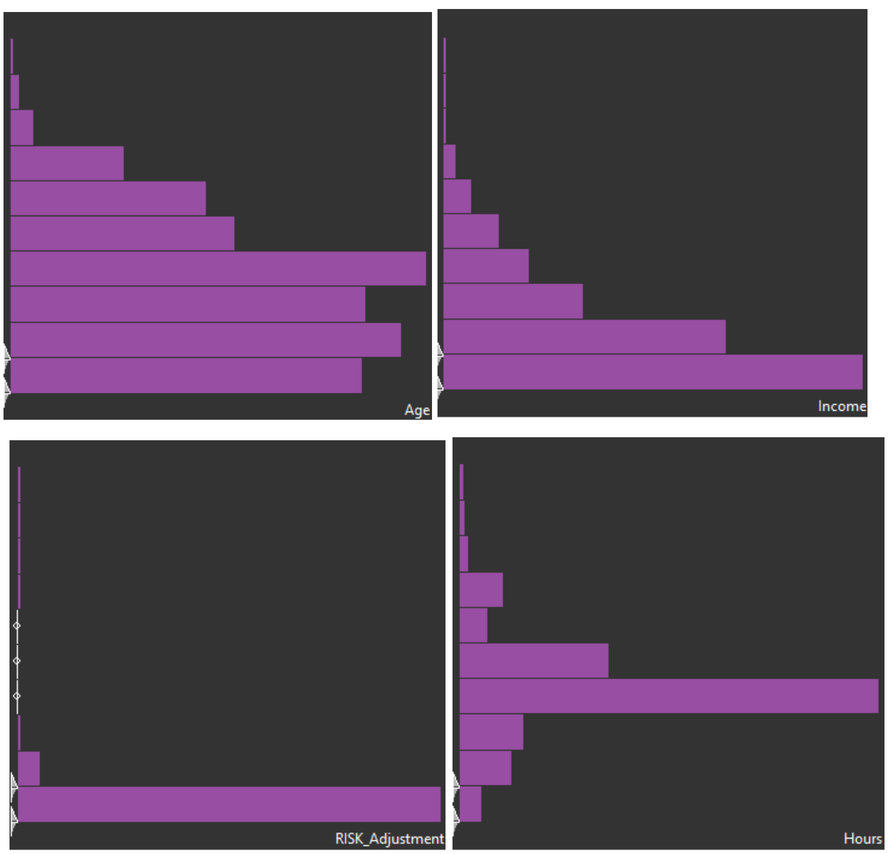
Ggobi is well suited for visualizing high dimensional data and dynamic graphics and for investigating hidden relationships.

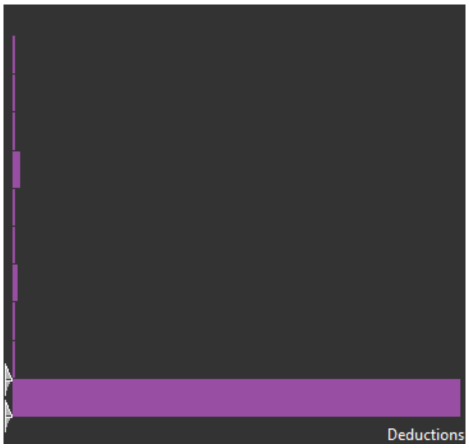
1D graphics

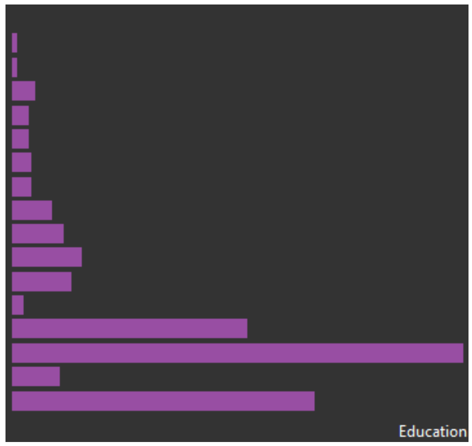
In order to investigate the data, we will start with a "tour" in 1D. This represents the projection of the 12 variables/axes on a 1-dimensional display. We begin with 1D plots of the quantitative variables in the data set.

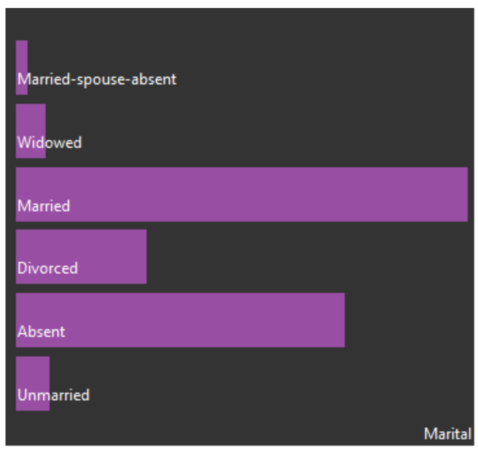
Histograms and Bar charts: These types of chart are well suited for visualising the distribution of qualitative data (bar chart) and quantitative data (histogram). We would like to know if our data is skewed to the left or right or relatively symmetric, and whether there are severe outliers that could be removed during the preprocessing (or cleaning) phase.

Figure 1. Histograms and Bar charts.









Comments:

The graphs reveal a few facts about the data. The variables “Age”, “Deduction”, “Risk adjustment”, and “Income” are skewed to the left. The variable “Hours” is relatively symmetric. As for the categorical items, “Gender” and “Marital”, there appear to be more male than female in the sample, and more married or absent individuals in the sample with respect to “Marital status”.

Formatted: Highlight

2D plots: Scatter matrix

This represents the projection of the 12 variables/axes on a 2-dimensional display. The procedure used in the grand tour projects the data by selecting two directions. This allows the observations of the data from all directions.

A common 2D plot is the Scatterplot Matrix. This plots the variables pairwise and allows to see if there are some types of relationships between variables by displaying pairwise plots.

High correlations (close to straight line scatter) indicate some linear relationship between pairs of variables but low correlations may indicate either no relationship or the presence of some nonlinear relationship in combination with other variables.

Figure 2. Scatterplot matrix for “Age” and “Income”.

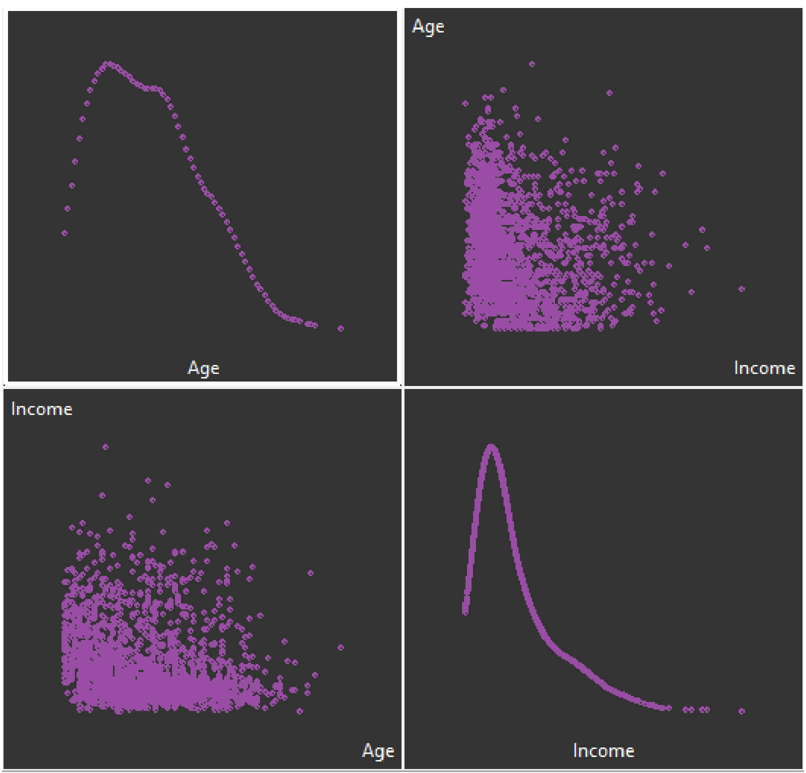
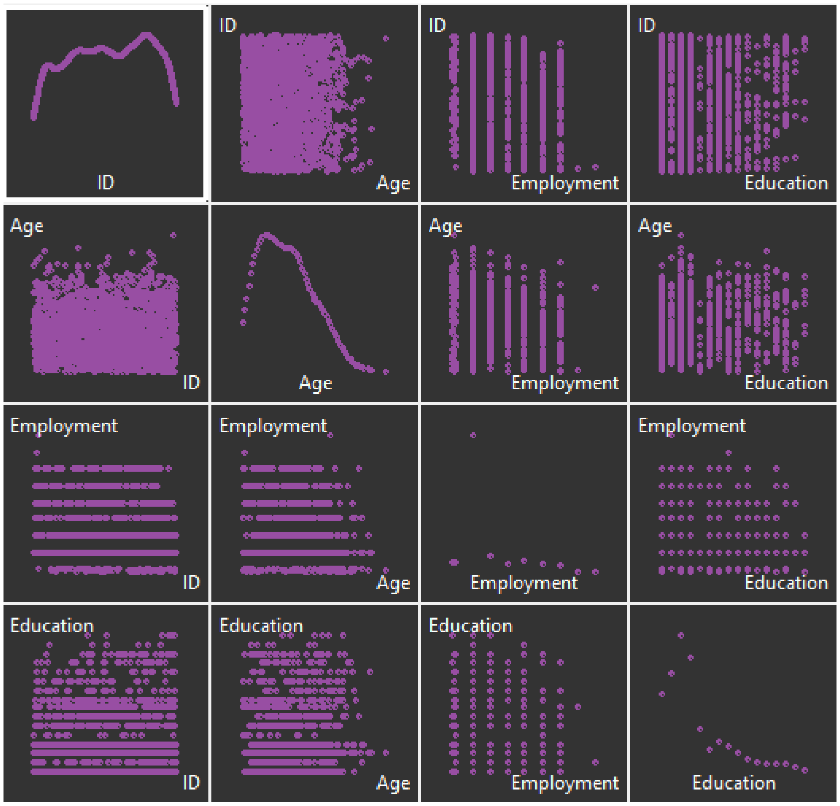


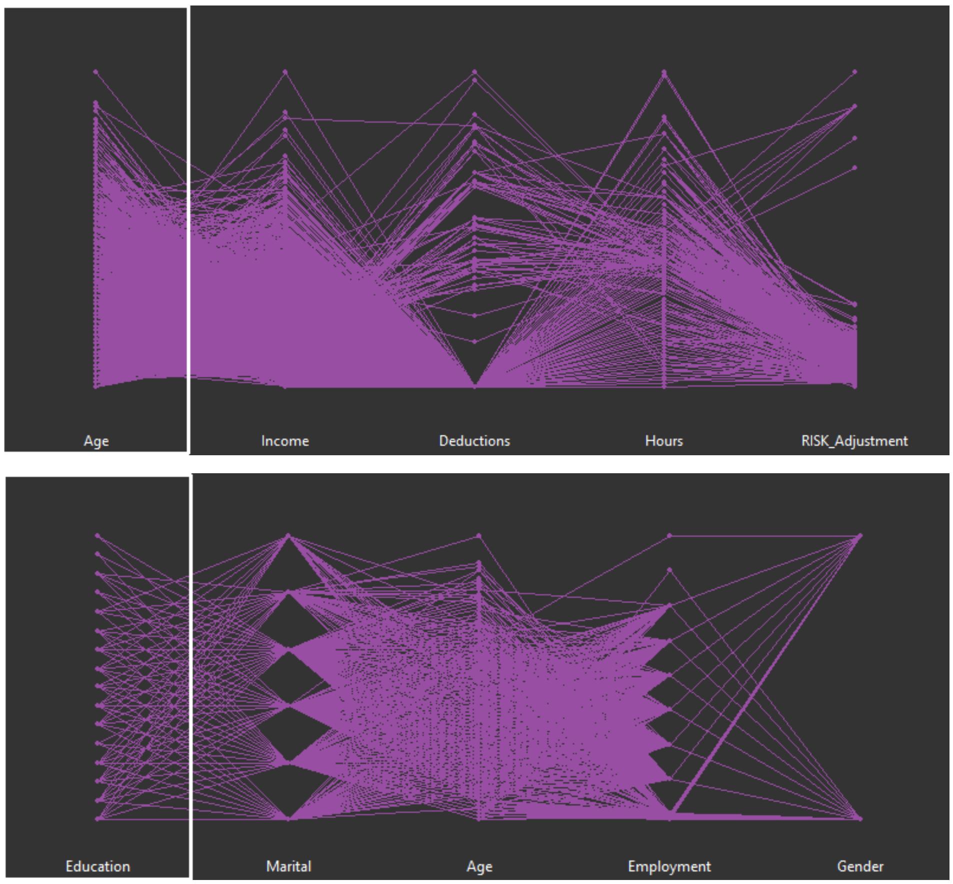
Figure 3. Scatterplot matrix for "Age", "Employment, "Education" and "Id".



High dimensional data visualization.

The Parallel Coordinates Plot is a method for investigating high dimensional data. Each data point has a value on each of the axes which are plotted vertically rather than at right angles to each other. We show in figure 4 below, two parallel coordinates plots for two subsets of the data set.

Figure 4. Parallel Coordinates Plot



Conclusion:

This section was meant to get a visual feeling of the data set “Audit.csv”. we presented 1D plots for both numerical and categorical variables. We also investigated relationships between in the data with 2D graphs. The use of Ggobi was useful in getting in-depth informations about the distributions of the variables and their relationships. Although this set contains few variables, dimension reduction is a key step in the multivariable analysis. In the next section, we shall select a subset of the variables in the data set and apply a dimension reduction technique known as PCA (principal component analysis).

PART II. Dimension reduction

The aim of dimension reduction is to compress the number of variables in the original data into a small set so as to ease the subsequent data analyses (such as regression, correlation analyses, hyp otheses testing, etc.) that might be performed on the data. This step, although crucial in general, does appear non pertinent in our study of the data "Audit.csv". Indeed, there are 6 quantitative va riables and only 5 pertinent for such analysis. Thus, this section purpose is to demonstrate the ap plication of a classic dimension reduction algorithm, namely PCA (Principal Component Analysi s). Although purely exploratory, PCA results can be used as a basis for investigating other data. Thus, one limitation of this section is the inability to generalized the result of this study.

1. *Methodology:*

For the purpose of this analysis, we selected a subset of 5 variables: "*Age", "Income", "Hours", "Deductions" and "Risk adjustment".* These variables are all measured on a numerical scale.

Preprocessing: The missing values were first removed and the data scaled before the cluster analysis.

In R, the first three cases of the set are shown below:

```
> Deductions=AuditData$Deductions;
> Data1=cbind(Age, Income, Hours, RiskAdj, Deductions);
> head(Data1)


> df=Data1;
> df=na.omit(df);
> df=scale(df);
> head(df, n = 3)
             Age      Income        Hours    RiskAdj Deductions
[1,] -0.04578664 -0.04094215  2.626809833 -0.2422672 -0.1983193
[2,] -0.26662253 -0.18082681 -0.828923452 -0.2422672 -0.1983193
[3,] -0.48745842  1.00526611 -0.006129813 -0.2422672 -0.1983193

> dim(df)
[1] 2000    5
```

2. *Running the PCA algorithm*

With a total of p=5 variables, exactly 5 principals' components were extracted, along with their standard deviation and proportion of variance.

```
> data.pca=prcomp(df, center = TRUE, scale. = TRUE);
> summary(data.pca)

Importance of components:
                          PC1    PC2    PC3    PC4    PC5
Standard deviation     1.1964 1.0103 0.9632 0.9581 0.8380
Proportion of Variance 0.2863 0.2041 0.1855 0.1836 0.1404
Cumulative Proportion  0.2863 0.4904 0.6760 0.8596 1.0000
```

The variance of each component is measured by the eigenvalue for that component. The higher the eigenvalue, the better the component captures the total information in the original data set.

```
> eig.val=get_eigenvalue(data.pca)
> eig.val
      eigenvalue variance.percent cumulative.variance.percent
Dim.1  1.4313662         28.62732                    28.62732
Dim.2  1.0206774         20.41355                    49.04087
Dim.3  0.9277295         18.55459                    67.59546
Dim.4  0.9180146         18.36029                    85.95575
Dim.5  0.7022124         14.04425                   100.00000
```

Comments:

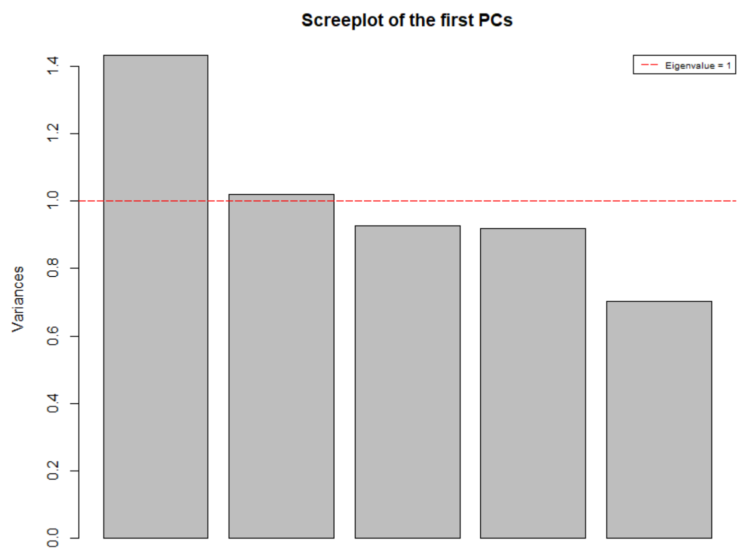
The first 4 components amount to a total of 85.96% of the total variance. This means that about 85.96% of the variation or information in the original data set is captured altogether by the first 4 (best) components. The first three component capture about 68% of the total variance. That is, by reducing the dimension from 5 to 3, we capture 67.6% of the total information. A look at the table above shows that the last component contributes very little in term of percentage to the total variance (about 14%).

How many components must be retained? The use of the scree plot can help answer this question.

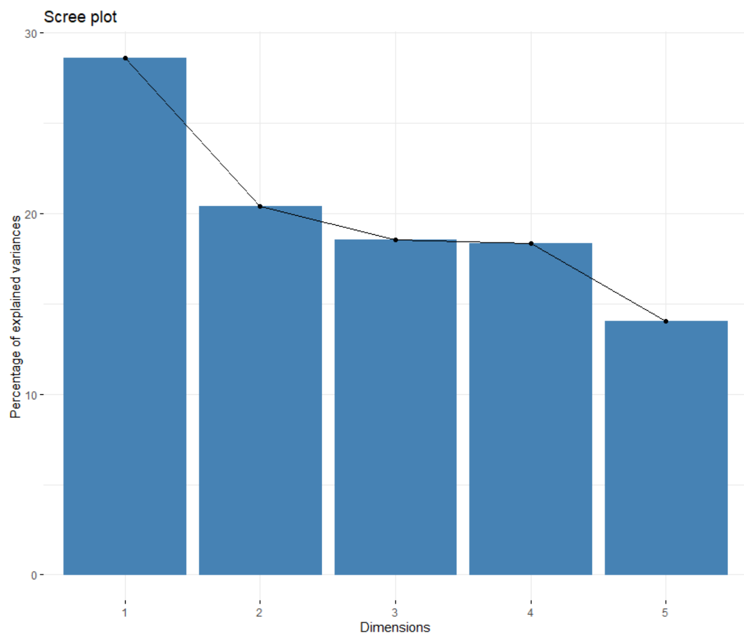
3. *Scree plot, for deciding on the number of components*

The scree plot graphs the variance on the y-axis and the number of the component on the x-axis. It shows the variation of the variance captures as we increase the number of components. In gene ral, less variance is captured as the number of components is included in the model. This allows f or dimension reduction (we can suppress those components with small contribution in terms of v ariance).

```
> screeplot(data.pca,main = "Screeplot of the first PCs")
> abline(h = 1, col="red", lty=5)
> legend("topright", legend=c("Eigenvalue = 1"),col=c("red"), lty=5, cex=0.6)
> fviz_eig(data.pca)
```

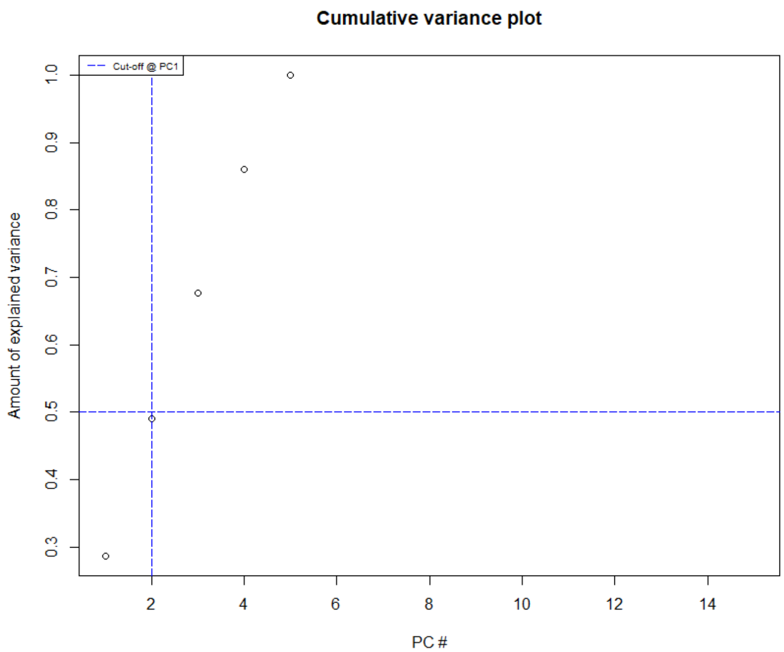


Graphs should be numbered and there is not label on the horizontal axis for this one.



The cumulative variance plot is also helpful in determining the appropriate number of principal components.

```
The graph below shows the cumulative plot for

> cumpro=cumsum (data.pca$sdev^2 / sum(data.pca$sdev^2))
> plot(cumpro[0:15], xlab = "PC #", ylab = "Amount of explained variance", ma
in = "Cumulative variance plot")
> abline(v = 2, col="blue", lty=5)
> abline(h = 0.5, col="blue", lty=5)
> legend ("topleft", legend=c ("Cut-off @ PC1"), col=c("blue"), lty=5, cex=0.
6)
```

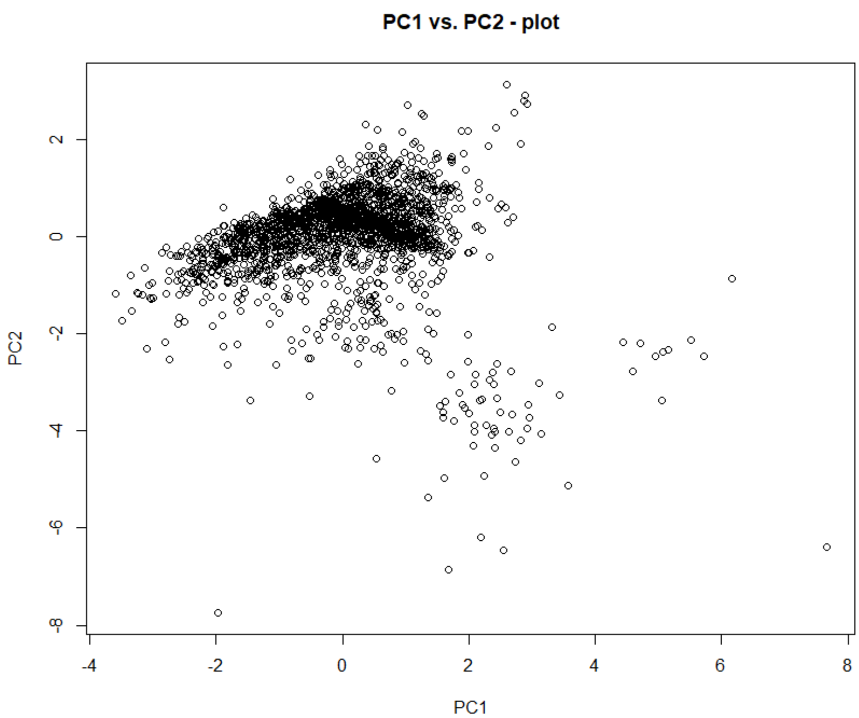


Comments: The scree plots suggest that 3 or 4 components could be optimal. However, the origi nal number of variables (p=5) is too small to make this finding pertinent.

4. *Plot of the first two principal components*

The plot below shows the representation of the data points in the first 2 components plane.

```
> plot(data.pca$x[,1], data.pca$x[,2], xlab="PC1", ylab = "PC2", main = "PC1
vs. PC2 - plot")
```



The coordinates (or scores) of the participants on the components are:

```
> data.pca$x[1:5,1:5]
            PC1         PC2        PC3         PC4        PC5
[1,]  0.9836440  1.73169248 -0.5056904  1.13916939 -1.2174492
[2,] -0.5281612 -0.17170175  0.3580401 -0.27738971  0.6126585
[3,] -0.9845120  0.06628008 -0.3592115  0.13929232 -0.4734396
[4,]  1.4410183  0.78664207 -0.3435072 -0.01416525 -0.1533557
[5,]  2.0027353 -0.34889351 -0.1786144 -1.40343912  0.1958866
```

The correlation between the variables and the components are:

```
> data.pca
Standard deviations (1, ..., p=5):
[1] 1.1963972 1.0102858 0.9631871 0.9581308 0.8379811

Rotation (n x k) = (5 x 5):
                 PC1        PC2          PC3         PC4           PC5
Age        0.5095489 -0.2598393  0.430095139 -0.45792095 -0.5274182970
Income    -0.5872600 -0.2483125 -0.341786131 -0.03972703 -0.6892536830
Hours      0.4286868  0.5774494 -0.263591025  0.44078682 -0.4679813792
RiskAdj    0.3865196 -0.2244748 -0.792917190 -0.37913576  0.1665895460
Deductions 0.2496316 -0.6978450 -0.001585471  0.67133689  0.0008076244
```

High correlations (close to 1 or -1) indicate a strong relationship between the variable and the component. That is, the component captures the variability in that variable, for instance, the matrix above shows that Risk adjustment has a large loading on PC3 (-0.79) while Income has a very low relationship with PC4 (-0.0397).

5. *Quality of representation*

The representation of a variable on a component can be measured with the cosine squared index and its contribution to the component.

```
> res.var=get_pca_var(data.pca);
> x1=res.var$cos2[,1:5];
> x2=res.var$contrib[,1:5];

> x1
                Dim.1      Dim.2        Dim.3       Dim.4        Dim.5
Age        0.37163998 0.06891253 1.716131e-01 0.192499941 1.953345e-01
Income     0.49364134 0.06293406 1.083753e-01 0.001448844 3.336005e-01
Hours      0.26304552 0.34034269 6.445886e-02 0.178363820 1.537891e-01
RiskAdj    0.21384238 0.05143085 5.832799e-01 0.131959017 1.948785e-02
Deductions 0.08919696 0.49705732 2.332051e-06 0.413742935 4.580230e-07

> x2
               Dim.1     Dim.2        Dim.3      Dim.4        Dim.5
Age        25.964005  6.751647 1.849818e+01 20.9691599 2.781701e+01
Income     34.487425  6.165911 1.168178e+01  0.1578237 4.750706e+01
Hours      18.377234 33.344784 6.948023e+00 19.4293019 2.190066e+01
RiskAdj    14.939740  5.038893 6.287177e+01 14.3743927 2.775208e+00
Deductions  6.231596 48.698766 2.513719e-04 45.0693218 6.522571e-05
```

The large numbers mean the item is well represented on the component.

Interpretation of the best 2 components:

We first rank the quality of representation of each of the 42 items of the first dimension (from large to small).

```
> xa1=res.var$cos2;
> sort (xa1, decreasing=TRUE)
```

```
5.832799e-01 4.970573e-01 4.936413e-01 4.137429e-01 3.716400e-01 3.403427e-01
2.630455e-01 2.138424e-01 1.953345e-01 1.924999e-01 1.783638e-01 1.716131e-01
1.319590e-01 1.083753e-01 8.919696e-02 6.891253e-02 6.445886e-02 6.293406e-02
1.948785e-02 1.448844e-03 2.332051e-06 4.580230e-07
```

Dimension#1: The first dimension is strongly correlated with the following items*: (Age, Income)*, so the first dimension can be labelled SOCIO-ECONOMIC STATUS.

Dimension#2: The second dimension is strongly correlated with the following items *(Hours, Deductions)*. "Deductions" is the total amount of expenses that a person claims in their financial statement, while "Hours" denotes the "Average hours worked on a weekly basis". So, the second dimension can be labelled FINANCIAL STATUS.

Conclusion:

We performed data reduction on a group of 5 numerical variables from the data set Audit.csv. Principal Component Analysis was used to extract 5 components, from which about 3 or 4 seem to be adequate to represent the original data. With only 3 components, we captured 67.6% of the total variance. With 4 components, about 85% of the total variance was captured.
The main limitation of this analysis lies in the fact that the original data set is not well suited for t his type of analysis. Dimension reduction is usually applied on large data sets with hundreds of dimensions or variables.

PART III. Data reduction. Clustering.

1. *The data set and reason for Clustering:*

The Audit data file contains characteristics of 2000 individual tax returns. The variables include Age (Age of person), Education (Highest level of education), Outcome (Amount of income declared), Deductions (Total amount of expenses that a person claims in their financial statement), Hours (Average hours worked on a weekly basis), Risk Adjustment (monetary amount of any adjustment to the person's financial status as a result of a productive audit).

Our data set contains an overwhelming amount of data in terms of number of cases *(n=2000).* We want to consider reducing the sample size n (number of cases). One methodology is to cluster together similar objects or cases. The goal is to group together sample units that look alike in such a way as to increase the degree of homogeneity within each cluster and heterogeneity between the clusters.

2. Kmeans as a clustering algorithm

Clustering is an unsupervised learning, because we do not know which class or cluster an observation belongs to. Various algorithms exist for cluster analysis. We shall use the commonly used Kmeans algorithm,

K-means clustering is an unsupervised machine learning algorithm for partitioning a given data set into a set of k groups or clusters, where k represents the number of groups pre-specified by the analyst. We applied the kmeans algorithm to the Audit data.

3. Methodology:

We selected a subset of features, because the Kmeans procedure works well on numerical attributes. The selected variables were combined in one data set, namely "Data1" which includes "*Age", "Income", "Hours", and "Risk adjustment".* The classification of observations into groups requires some methods for computing the distance or the (dis)similarity between each pair of observations. We use the Euclidean distance for simplicity.

Preprocessing:

The missing values were first removed and the data scaled before the cluster analysis.

```
> df=Data1;
> df=na.omit(df);
> df=scale(df);
> head(df, n = 3)
             Age      Income        Hours    RiskAdj
[1,] -0.04578664 -0.04094215  2.626809833 -0.2422672
[2,] -0.26662253 -0.18082681 -0.828923452 -0.2422672
[3,] -0.48745842  1.00526611 -0.006129813 -0.2422672
```

4. Selecting the optimal number of clusters

The Kmeans procedures takes an integer K as input, for the number of categories in the data. We applied 3 different methods for selecting the right number K of clusters in our data set. The "Elbow method", the "Silhouette method", and the "Gap statistic method". Unfortunately, the Gap method did not converge for our data set.

The other two methods produced different results, Elbow method (k = 4), Silhouette method (k=7). We decided to be conservative and chose a value close to the average of the two methods recommendations. We chose k = 5 for our analysis.

Figure 1. Optimal number of clusters. "Elbow method"

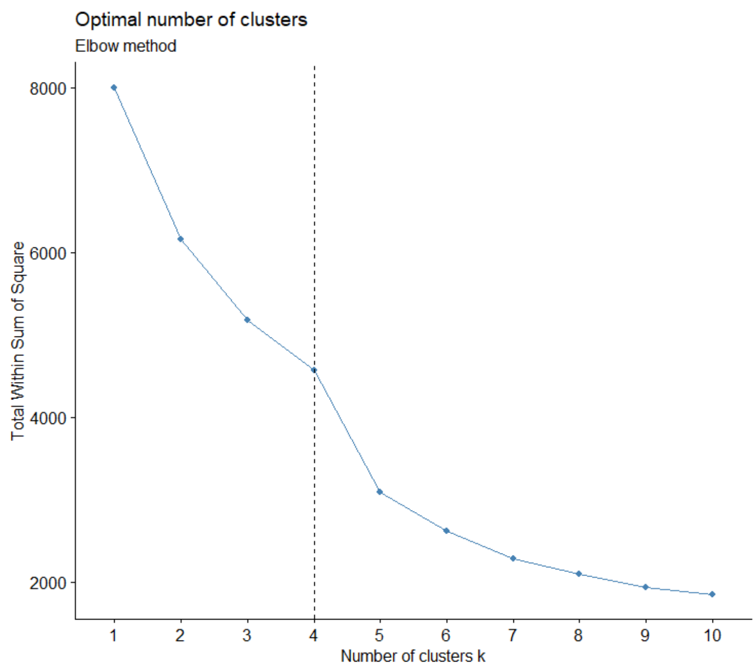
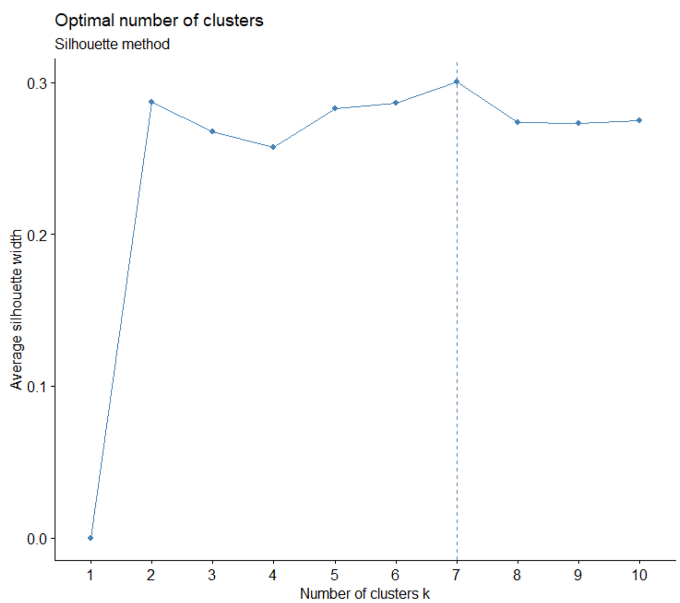
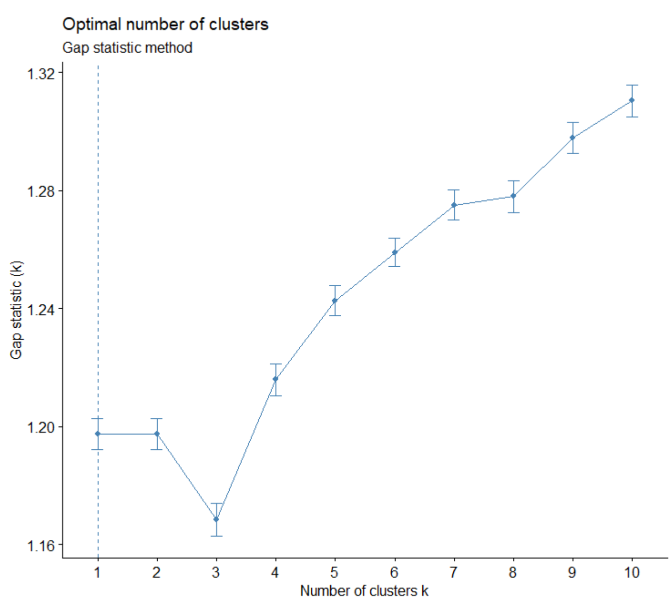


Figure 2. Optimal number of clusters. “Silhouette and Gap methods”





5. *Running Kmeans, for k = 5*

The k-means algorithm tries to cluster objects that are ‘close’ together, using a distance measure. The following will consider Euclidean distance. The algorithm works as follow:

1. Select a number of points K to be the centers (centroids)
2. Start with a random initialization for the centers.
3. Find the distance from each point to every center; assign the point to closest center;
4. For each set of points assigned to a center, find the centroid of the cluster;
5. Take that value as the new center;
6. Repeat the process until the centers do not seem to move.

The results in R are as follow:

```
> set.seed (123)
> km.res=kmeans (df, 5, nstart = 25)
> print(km.res)

K-means clustering with 5 clusters of sizes 367, 355, 712, 12, 554

Cluster means:
          Age     Income      Hours      RiskAdj
1  0.05650791 -0.4765976  1.3495740  0.125844874
2 -0.48061562  1.7569470 -0.4729203 -0.170213252
3 -0.69413021 -0.2567961 -0.2556258 -0.180245915
4  0.69646732 -0.3109788  0.2818480 11.531038989
5  1.14755098 -0.4733471 -0.2685624  0.007587111


Clustering vector:

   [1] 1 3 2 1 5 5 1 3 3 3 2 5 2 3 5 3 3 3 5 1 2 3 3 3 2 3 3 1 5 3 3 3 2 5
  [35] 3 2 2 2 5 3 5 5 3 1 3 5 5 3 5 2 1 3 3 5 3 3 3 3 1 1 5 3 5 5 1 3 5 1
  [69] 5 3 3 3 5 5 3 5 1 3 2 1 1 1 3 1 2 5 3 1 3 1 3 5 5 5 2 5 1 3 3 5 2 1
 [103] 5 3 1 5 5 3 1 5 5 5 5 5 3 2 1 5 5 3 1 3 2 3 5 1 2 3 2 3 3 2 3 5 1 2
 [137] 1 3 5 3 5 3 3 5 1 3 3 2 3 1 3 1 5 3 2 5 3 3 2 3 5 5 1 3 5 5 3 1 5 5
 [171] 5 5 5 3 3 5 3 1 1 1 3 1 3 3 3 1 3 5 3 3 5 5 1 5 3 5 3 2 3 2 3 3 3 3
 [205] 5 3 3 3 1 3 1 1 5 3 3 2 5 3 2 5 3 1 5 5 3 3 3 3 5 2 1 5 5 5 5 3 1 1
 [239] 2 3 2 1 3 3 5 5 3 3 3 1 5 3 1 5 1 2 3 2 3 5 2 1 3 5 1 5 5 3 5 2 3 5
 [273] 2 5 3 5 3 2 3 1 5 5 5 1 5 1 2 5 2 3 5 3 3 2 1 5 3 3 3 3 1 3 2 3 3 1


Within cluster sum of squares by cluster: [1] 630.25876 760.94887 712.64083 3
0.26892 878.73915(between_SS / total_SS = 62.3 %)
```

We aggregated the data set with the cluster number and mean, so that each cluster is shown with its mean for all four covariates.:

```
> aggregate (Data1, by=list(cluster=km.res$cluster), mean)

  cluster      Age    Income    Hours    RiskAdj
1       1 39.38965  51506.95 56.47684  3070.7439
2       2 32.09296 207010.00 34.32676   601.0648
3       3 29.19242  66809.89 36.96770   517.3736
4       4 48.08333  63037.60 43.50000 98211.4167
5       5 54.21119  51733.26 36.81047  2084.2527
```

The table below shows the first six participants of the study with their scores on each variable (unscaled) and the cluster to which they belong.

```
> dd = cbind (Data1, cluster = km.res$cluster);
> head(dd)

     Age    Income Hours RiskAdj cluster
[1,]  38  81838.00    72       0       1
[2,]  35  72099.00    30       0       3
[3,]  32 154676.74    40       0       2
[4,]  45  27743.82    55    7298       1
[5,]  60   7568.23    40   15024       5
[6,]  74  33144.40    30       0       5
```

The next table shows the first six participants of the study with their scores on each variable (scale) and the cluster to which they belong.

```
> dd = cbind (df, cluster = km.res$cluster);
> head(dd)

             Age      Income        Hours    RiskAdj cluster
[1,] -0.04578664 -0.04094215  2.626809833 -0.2422672       1
[2,] -0.26662253 -0.18082681 -0.828923452 -0.2422672       3
[3,] -0.48745842  1.00526611 -0.006129813 -0.2422672       2
[4,]  0.46949710 -0.81791576  1.228060646  0.6325964       1
[5,]  1.57367655 -1.10770480 -0.006129813  1.5587673       5
[6,]  2.60424404 -0.74034534 -0.828923452 -0.2422672       5
```

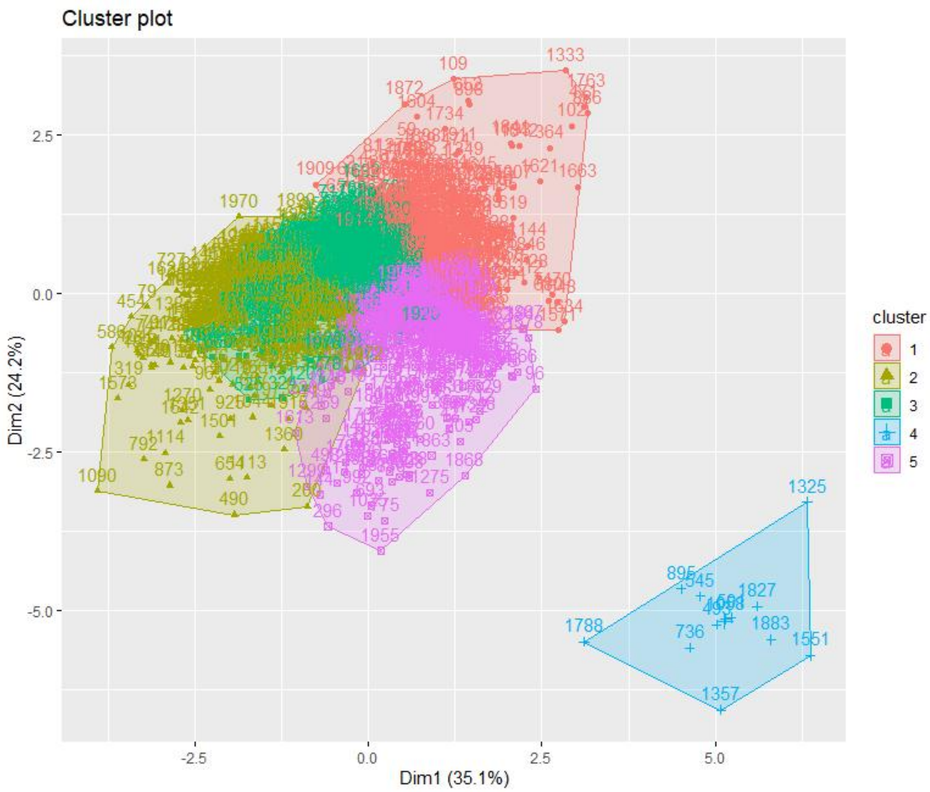
```
> head (km.res$cluster, 5)

[1] 1 3 2 1 5
```

6. Graph of the clusters

We can get a visual look at the spatial configuration of the k=5 groupings. If we print the results, we see that our groupings resulted in 5 cluster sizes which are respectively: 367, 355, 712, 12, 554. We see the cluster centers (means) for the groups across the four variables “Age”, “Income”, “Hours”, and “Risk adjustment”. We also get the cluster assignment for each observation represented by its ID, as show below.

```
> fviz_cluster (km.res, df)
```

Figure 3. Cluster plot on first two dimensions.



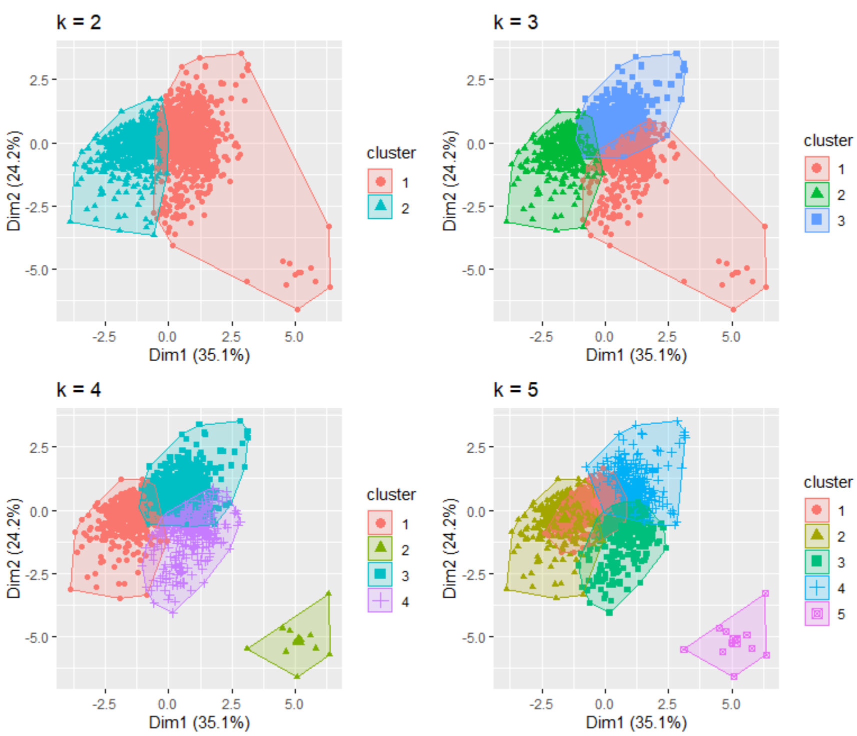
Comments: We have 5 relatively distinct clusters. The total within-cluster sum of square is a measure of the compactness (i.e., goodness) of the clustering (should be as small as possible). In our case, we have the following goodness fit indices (within sum = 3012.86).

```
> km.res$tot.withinss
[1] 3012.857
> km.res$betweenss
[1] 4983.143
```

7. Varying the number of clusters k

Because the number of clusters (k) must be set before we start the algorithm, it is often advantageous to use several different values of k and examine the differences in the results. We executed the same process for 2, 3, 4, and 5 clusters, and the results are shown in the figure below. This graph allows a comparison of different clustering models and appreciation of the improvement in the data reduction, as we varied the number of clusters.

```
> k3=kmeans(df, centers = 3, nstart = 25);
> k4=kmeans(df, centers = 4, nstart = 25);
> k5=kmeans(df, centers = 5, nstart = 25);
> k2=kmeans(df, centers = 2, nstart = 25);
> p1=fviz_cluster(k2, geom = "point", data = df) + ggtitle("k = 2")
> p2=fviz_cluster(k3, geom = "point",  data = df) + ggtitle("k = 3")
> p3=fviz_cluster(k4, geom = "point",  data = df) + ggtitle("k = 4")
> p4=fviz_cluster(k5, geom = "point",  data = df) + ggtitle("k = 5")
> grid.arrange(p1, p2, p3, p4, nrow = 2)
```



Comments: Although k=5 clusters seem to be optimal, the data reduction is relatively good for even small clusters such as k=2 or k=3. We see that the goodness fit increases but very slowly as the number of clusters increases. Indeed, the total within-cluster sum of square measures for each model is:

```
> wws = c(k2$tot.withinss, k3$tot.withinss, k4$tot.withinss, k5$tot.withinss)
> wws
```

```
[1] 6153.580 5171.976 3616.830 3012.857
```

Conclusion:

The K-means clustering procedure is simple and fast. Furthermore, it can efficiently deal with very large data sets. However, there are some weaknesses of the k-means approach.

The limitations include non reproducibility and the impossibility of solutions. Different runs on the same data set can lead to situations where some points can be assigned to different centers for different initializations. Thus, the algorithm can end up with different clusters after a new initialization. Moreover, if the initial center is moved to an empty region of the plane, the algorithm will fail to find the clusters. Although random initialization can resolve this issue. Another potential disadvantage of K-means clustering is that it requires to pre-specify the number of clusters. The method is also sensitive to outliers.

Finally, there is no definitive answer to the question of the optimal number of clusters. The number k is subjective and depends on the method used for measuring similarities and the parameters used for partitioning. Another solution consists of inspecting the dendrogram produced using hierarchical clustering to see if it suggests a particular number of clusters. However, this approached led to uninterpretable results for our data set.

In this section, we demonstrated how to compute clusters using the R program. Additionally, we described different methods for choosing the optimal number of clusters in a data set. These methods include the elbow, the silhouette and the gap statistic methods

PART IV. Data reduction. Classification, Supervised learning.

In this section, we apply the logistic regression to classify subjects according to whether the audits were productive or productive audits. Productive audits are those that result in an adjustment being made to a client's financial statement. The dependent variable or binary target variable for classification modeling (0/1), is TARGET_Adjusted. The logistic procedure is limited to only two-class classification problems. There is an extension, called multinomial logistic regression, for multiclass classification problem.

Logistic regression is used to predict the class (or category) of individuals based on one or multiple predictor variables (x). It is used to model a binary outcome, that is a variable, which can have only two possible values: 0 or 1, yes or no. but in order to predict the probability that an audit will turn out productive or not, we must define a list of predictors or factors that may affect the probability of the dependent variable.

We chose the following features: “Age”, “Income”, “Hours”;

The model is as follow:

Logit (TARGET_Adjusted) = B0 + B1*Age + B2*Income + B3*Hours + error;

Where logit(Y) denotes the log of the odds of Y=1;

It should be noted that Logistic regression does not return directly the class of observations. It allows us to estimate the probability (p) of class membership. The probability will range between 0 and 1. We choose the threshold probability at which the category flips from one to the other. By default, this is set to p = 0.5.

1. Supervised Learning

Classification is a form of supervised learning where we have prior information on what the result (or class/group) should look like. In our case, we know the class label of the examples of the training set (0/1). Building a logistic ‘classifier’ is nothing more than running the logistic regression of TARGET_Adjusted on Age, Income and Hours. In so doing we hope to determine which features are important for the classification, and which values of those features lead to success or failure. Once such values are determined based on the known cases, we have a rule or “classifier” that can be used to predict the outcome/result for new cases.

2. Methodology

The data has already been loaded from the previous section. Moreover, all required R packages have been included before the analysis. We first partitioned our data set into two: testing and training. We also installed a few extra packages.

```
> install.packages("caret",
+                  repos = "http://cran.r-project.org",
+                  dependencies = c("Depends", "Imports", "Suggests"))

install.packages("gower");
library(gower);
library(caret);
library("mgcv");
library(readr);

> inTrain=createDataPartition (Dep,p=.75,list = FALSE);
> df=AuditData;
> training=df[inTrain,];
> testing=df[-inTrain,];
> nrow(training)
[1] 1500
> nrow(testing)
[1] 500
```

3. Running/Building the logistic classifier (Training)

On the training set, we built the classifier using the generalized linear model procedure:

```
> Dep1=training$TARGET_Adjusted;
> set.seed(123);
> model1=glm(Dep1~training$Age+training$Income+training$Hours, family = binom
ial);
> model1

Call:  glm(formula = Dep1 ~ training$Age + training$Income + training$Hours,
    family = binomial)

Coefficients:
    (Intercept)     training$Age  training$Income   training$Hours
     -4.075e+00        3.745e-02       -6.844e-06        4.253e-02

Degrees of Freedom: 1499 Total (i.e. Null);  1496 Residual
Null Deviance:      1591
Residual Deviance: 1408       AIC: 1416
```

The estimated mode is as follow:

Logit (TARGET_Adjusted) = - 4.07 + 0.03745*Age – 0.000*Income + 0.0425*Hours

As shown below, all predictors are significant at the 5% level (p-value < 0.05).

```
> summary(model1);

Call:
glm(formula = Dep1 ~ training$Age + training$Income + training$Hours,
    family = binomial)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0137  -0.7337  -0.5226  -0.2294   3.0522

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)     -4.075e+00  3.969e-01 -10.266  < 2e-16 ***
training$Age     3.745e-02  5.250e-03   7.132 9.88e-13 ***
training$Income -6.844e-06  1.332e-06  -5.137 2.79e-07 ***
training$Hours   4.253e-02  5.908e-03   7.198 6.10e-13 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1590.8  on 1499  degrees of freedom
Residual deviance: 1408.2  on 1496  degrees of freedom
AIC: 1416.2


Number of Fisher Scoring iterations: 5

> model1$coefficients
    (Intercept)    training$Age training$Income   training$Hours
  -4.074940e+00    3.744654e-02   -6.844244e-06    4.252753e-02
> model1$df.null
[1] 1499
```

```
> model1$df.residual
[1] 1496
> model1$null.deviance
[1] 1590.787
> model1$deviance
[1] 1408.202
```

4. Making predictions

The classifier can be used o make predictions we can test the classifier quality on the testing set.

```
> # Make predictions
> probabs=model1%>%predict(testing, type = "response");
> m.prob=predict(model1, data=testing, type="response");
> predicted.class=ifelse(probabs>0.5, "pos", "neg");
```

The predicted probabilities are (for a few sample of the testing set):

```
> probabs
          1           2           3           4           5           6 
7           8 
0.462543181 0.121111309 0.440109690 0.455407546 0.436613330 0.186452845 0.090
616226 0.115404115 
          9          10          11          12          13          14 
15          16 
0.058218761 0.427423500 0.058383197 0.082219334 0.159664203 0.211450328 0.220
812616 0.458738311 
         17          18          19          20          21          22 
23          24 
0.288381909 0.092887833 0.266937599 0.222050305 0.129543414 0.417079423 0.344
346201 0.150647028 
         25          26          27          28          29          30 
31          32 
0.174426341 0.130899159 0.052173389 0.230361611 0.244928417 0.151676983 0.057
487107 0.294920368 
         33          34          35          36          37          38 
39          40 
0.173836039 0.364188762 0.263951031 0.175819308 0.480775551 0.257605146 0.298
529020 0.284986372 
```

The predicted classes are:

```
> predicted.class
    1     2     3     4     5     6     7     8     9    10    11    12    13 
14    15    16    17 
"neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" 
"neg" "neg" "neg" "neg" 
   18    19    20    21    22    23    24    25    26    27    28    29    30 
31    32    33    34 
"neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" 
"neg" "neg" "neg" "neg" 
   35    36    37    38    39    40    41    42    43    44    45    46    47 
48    49    50    51 
"neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" 
"neg" "neg" "pos" "neg" 
   52    53    54    55    56    57    58    59    60    61    62    63    64 
65    66    67    68 
"neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" "neg" 
"neg" "neg" "neg" "neg" 
```

```
   69     70     71     72     73     74     75     76     77     78     79     80     81 
82     83     84     85 
"neg"  "neg"  "neg"  "neg"  "neg"  "neg"  "pos"  "neg"  "neg"  "pos"  "neg"  "neg"  "neg" 
"neg"  "neg"  "neg"  "neg"
```

5. Comparing prediction and Observation

We can set a confusion table (contingency table 2x2) to compare the error rate on the testing set.

```
> m.pred=rep("0", dim(training)[1]);
> m.pred[m.prob>.5]="1";
> m.pred

   [1] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0
" "0" "0" "0" "0" "0" "0"
  [25] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0
" "0" "0" "0" "0" "0" "0"
  [49] "0" "1" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0
" "0" "0" "0" "0" "0" "0"
  [73] "0" "0" "1" "0" "0" "1" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0
" "0" "0" "0" "0" "0" "0"
  [97] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0
" "0" "0" "0" "0" "0" "0"
 [121] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "1" "0
" "0" "0" "0" "0" "0" "0"
 [145] "1" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0
" "0" "0" "0" "0" "0" "0"
 [169] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0
" "0" "0" "0" "0" "0" "0"
 [193] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0
" "0" "0" "0" "0" "0" "0"
 [217] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0
" "0" "0" "0" "0" "0" "0"


> table(m.pred, Dep1);
      Dep1
m.pred    0    1
     0 1139  303
     1   27   31
```

The total prediction rate of success is (1139+31)/(27+303+1139+31) = 0.78 = 78%. Our classifier has about 80% success rate on the testing set and 20% error rate.

Conclusion:

We used a logistic regression model to built binary classifier for the variable TARGET_Adjusted. The model predicts whether the audits were productive or productive audits using the estimated equation: Logit (TARGET_Adjusted) = - 4.07 + 0.03745*Age – 0.000*Income + 0.0425*Hours. This model has an 80% success rate on new data. It can be used to predict future outcome for new data.

~~We used a logistic regression model to builtdd a a binary classifier for the variable TARGET_Adjusted. The model predicts whether the audits were productive or productive audits using the estimated equation: Logit (TARGET_Adjusted) = - 4.07 + 0.03745*Age - 0.000*Income + 0.0425*Hours. This model has an 80% success rate on new data. It can be used to predict future outcome for new data.~~

References

1. Hartigan, JA, and MA Wong. 1979. "Algorithm AS 136: A K-means clustering algorithm." Applied Statistics. Royal Statistical Society, 100–108.

2. MacQueen, J. 1967. "Some Methods for Classification and Analysis of Multivariate Observations." In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics, 281–97. Berkeley, Calif.: University of California Press. http://projecteuclid.org:443/euclid.bsmsp/1200512992.

3. Charrad, Malika, Nadia Ghazzali, Véronique Boiteau, and Azam Niknafs. 2014. "NbClust: An R Package for Determining the Relevant Number of Clusters in a Data Set." Journal of Statistical Software 61: 1–36. http://www.jstatsoft.org/v61/i06/paper.