Customer Personality Analysis

Ivan Meng

Title and Introduction

In this project, we choose the Customer Personality Analysis as our dataset for further exploration. This dataset is a comprehensive record on a group of customers daily spend and correlating values. It contains 29 variables (columns) and 2240 observations (rows), which includes values like customers' data of birth, education level, amount of money spend on wine, and etc. This dataset was acquired from Kaggle and can be accessed through https://www.kaggle.com/datasets/imakash3011/customer-personality-analysis?resource=download (https://www.kaggle.com/datasets/imakash3011/customer-personality-analysis?resource=download).

There are several reasons we choose this dataset. One of the most important reason is that customers are always the main concern of a company. A better understanding of customers behavior can help the company to conduct a more efficient products service accordingly. Specifically speaking, this analysis help a company to modify its product based on its target customers from different types of customer segments. For example, a company can decide which particular group of customer they should spend mony on to market for a specific type of product based on this analysis.

Through this analysis, we expected to find a common trend among the various customers and hopefully can split them into different groups based on the cluster results. We may need to get rid of several columns of data before the actual analysis to clean up any unnecessary noise that may affect the visulization.

Load the packages `tidyverse`, `factoextra`, `cluster`, and `factoextra`.

```
# import packages
library(tidyverse)
library(factoextra)
library(cluster)
library(factoextra)
library(naniar)
library(corrplot)
library("readxl")
library(plotROC)
```

Importing Data

```
# import dataset
customers <- read.delim("~/Downloads/marketing_campaign.csv")
# customers <-read_excel("campaign.xlsx")
# view the top 6 rows of dataset
head(customers)
```

```
##     ID Year_Birth  Education Marital_Status Income Kidhome Teenhome Dt_Customer
## 1 5524       1957 Graduation         Single  58138       0        0  04-09-2012
## 2 2174       1954 Graduation         Single  46344       1        1  08-03-2014
## 3 4141       1965 Graduation       Together  71613       0        0  21-08-2013
##   Recency MntWines MntFruits MntMeatProducts MntFishProducts MntSweetProducts
## 1      58      635        88             546             172               88
## 2      38       11         1               6               2                1
## 3      26      426        49             127             111               21
##   MntGoldProds NumDealsPurchases NumWebPurchases NumCatalogPurchases
## 1           88                 3               8                  10
## 2            6                 2               1                   1
## 3           42                 1               8                   2
##   NumStorePurchases NumWebVisitsMonth AcceptedCmp3 AcceptedCmp4 AcceptedCmp5
## 1                 4                 7            0            0            0
## 2                 2                 5            0            0            0
## 3                10                 4            0            0            0
##   AcceptedCmp1 AcceptedCmp2 Complain Z_CostContact Z_Revenue Response
## 1            0            0        0             3        11        1
## 2            0            0        0             3        11        0
## 3            0            0        0             3        11        0
##  [ reached 'max' / getOption("max.print") -- omitted 3 rows ]
```

Display dataset

```
# Get the information of the dataset
glimpse(customers)
```

```
## Rows: 2,240
## Columns: 29
## $ ID                 <int> 5524, 2174, 4141, 6182, 5324, 7446, 965, 6177, 485…
## $ Year_Birth         <int> 1957, 1954, 1965, 1984, 1981, 1967, 1971, 1985, 19…
## $ Education          <chr> "Graduation", "Graduation", "Graduation", "Graduat…
## $ Marital_Status     <chr> "Single", "Single", "Together", "Together", "Marri…
## $ Income             <int> 58138, 46344, 71613, 26646, 58293, 62513, 55635, 3…
## $ Kidhome            <int> 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1,…
## $ Teenhome           <int> 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1,…
## $ Dt_Customer        <chr> "04-09-2012", "08-03-2014", "21-08-2013", "10-02-2…
## $ Recency            <int> 58, 38, 26, 26, 94, 16, 34, 32, 19, 68, 11, 59, 82…
## $ MntWines           <int> 635, 11, 426, 11, 173, 520, 235, 76, 14, 28, 5, 6,…
## $ MntFruits          <int> 88, 1, 49, 4, 43, 42, 65, 10, 0, 0, 5, 16, 61, 2, …
## $ MntMeatProducts    <int> 546, 6, 127, 20, 118, 98, 164, 56, 24, 6, 6, 11, 4…
## $ MntFishProducts    <int> 172, 2, 111, 10, 46, 0, 50, 3, 3, 1, 0, 11, 225, 3…
## $ MntSweetProducts   <int> 88, 1, 21, 3, 27, 42, 49, 1, 3, 1, 2, 1, 112, 5, 1…
## $ MntGoldProds       <int> 88, 6, 42, 5, 15, 14, 27, 23, 2, 13, 1, 16, 30, 14…
## $ NumDealsPurchases  <int> 3, 2, 1, 2, 5, 2, 4, 2, 1, 1, 1, 1, 1, 3, 1, 1, 3,…
## $ NumWebPurchases    <int> 8, 1, 8, 2, 5, 6, 7, 4, 3, 1, 1, 2, 3, 6, 1, 7, 3,…
## $ NumCatalogPurchases <int> 10, 1, 2, 0, 3, 4, 3, 0, 0, 0, 0, 0, 4, 1, 0, 6, 0…
## $ NumStorePurchases  <int> 4, 2, 10, 4, 6, 10, 7, 4, 2, 0, 2, 3, 8, 5, 3, 12,…
## $ NumWebVisitsMonth  <int> 7, 5, 4, 6, 5, 6, 6, 8, 9, 20, 7, 8, 2, 6, 8, 3, 8…
## $ AcceptedCmp3       <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,…
## $ AcceptedCmp4       <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,…
## $ AcceptedCmp5       <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,…
## $ AcceptedCmp1       <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,…
## $ AcceptedCmp2       <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,…
## $ Complain           <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,…
## $ Z_CostContact      <int> 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,…
## $ Z_Revenue          <int> 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11…
## $ Response           <int> 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0,…
```

Tidying data

```
# Summarize the missing values in the data
miss_var_summary(customers)
```

```
## # A tibble: 29 × 3
##    variable            n_miss pct_miss
##    <chr>                <int>    <dbl>
##  1 Income                  24     1.07
##  2 ID                       0     0
##  3 Year_Birth               0     0
##  4 Education                0     0
##  5 Marital_Status           0     0
##  6 Kidhome                  0     0
##  7 Teenhome                 0     0
##  8 Dt_Customer              0     0
##  9 Recency                  0     0
## 10 MntWines                 0     0
## 11 MntFruits                0     0
## 12 MntMeatProducts          0     0
## 13 MntFishProducts          0     0
## 14 MntSweetProducts         0     0
## 15 MntGoldProds             0     0
## 16 NumDealsPurchases        0     0
## 17 NumWebPurchases          0     0
## 18 NumCatalogPurchases      0     0
## 19 NumStorePurchases        0     0
## 20 NumWebVisitsMonth        0     0
## 21 AcceptedCmp3             0     0
## 22 AcceptedCmp4             0     0
## 23 AcceptedCmp5             0     0
## 24 AcceptedCmp1             0     0
## 25 AcceptedCmp2             0     0
## 26 Complain                 0     0
## 27 Z_CostContact            0     0
## 28 Z_Revenue                0     0
## 29 Response                 0     0
```

```
# Check the which variables are numeric in the dataset.
sapply(customers, is.numeric)
```

```
##                   ID          Year_Birth           Education      Marital_Status
##                 TRUE                TRUE               FALSE               FALSE
##               Income             Kidhome            Teenhome         Dt_Customer
##                 TRUE                TRUE                TRUE               FALSE
##              Recency            MntWines           MntFruits     MntMeatProducts
##                 TRUE                TRUE                TRUE                TRUE
##     MntFishProducts    MntSweetProducts        MntGoldProds   NumDealsPurchases
##                 TRUE                TRUE                TRUE                TRUE
##     NumWebPurchases NumCatalogPurchases   NumStorePurchases   NumWebVisitsMonth
##                 TRUE                TRUE                TRUE                TRUE
##        AcceptedCmp3        AcceptedCmp4        AcceptedCmp5        AcceptedCmp1
##                 TRUE                TRUE                TRUE                TRUE
##        AcceptedCmp2            Complain       Z_CostContact           Z_Revenue
##                 TRUE                TRUE                TRUE                TRUE
##            Response
##                 TRUE
```

From the above output, we learn that the income variabes has 24 rows of missing data in the dataset. Since the percentage is very low, we can drop all the missing values and would not influence our analysis. The variables: Education, Marital_Status, and Dt_Customer are categorical, the rest variables in the dataset are numerical.

```
# remove missing values
customers <- na.omit(customers)
dim(customers)
```

```
## [1] 2216   29
```

Create a new variable “Age” of customer from “Year of Birth” by current year “2020”.

```
# Create variable "Age"
customers <- customers %>%
  mutate(Age = 2022-customers$Year_Birth)
```

Combine variables “Teenhome” and “Kidhome” into a new variable “Child” to show how many children the customer have.

```
# create variable "Child"
customers <- customers %>%
  mutate(Child_home = Kidhome+Teenhome)
```

Create a variable “Total_expense” of products for each customers.

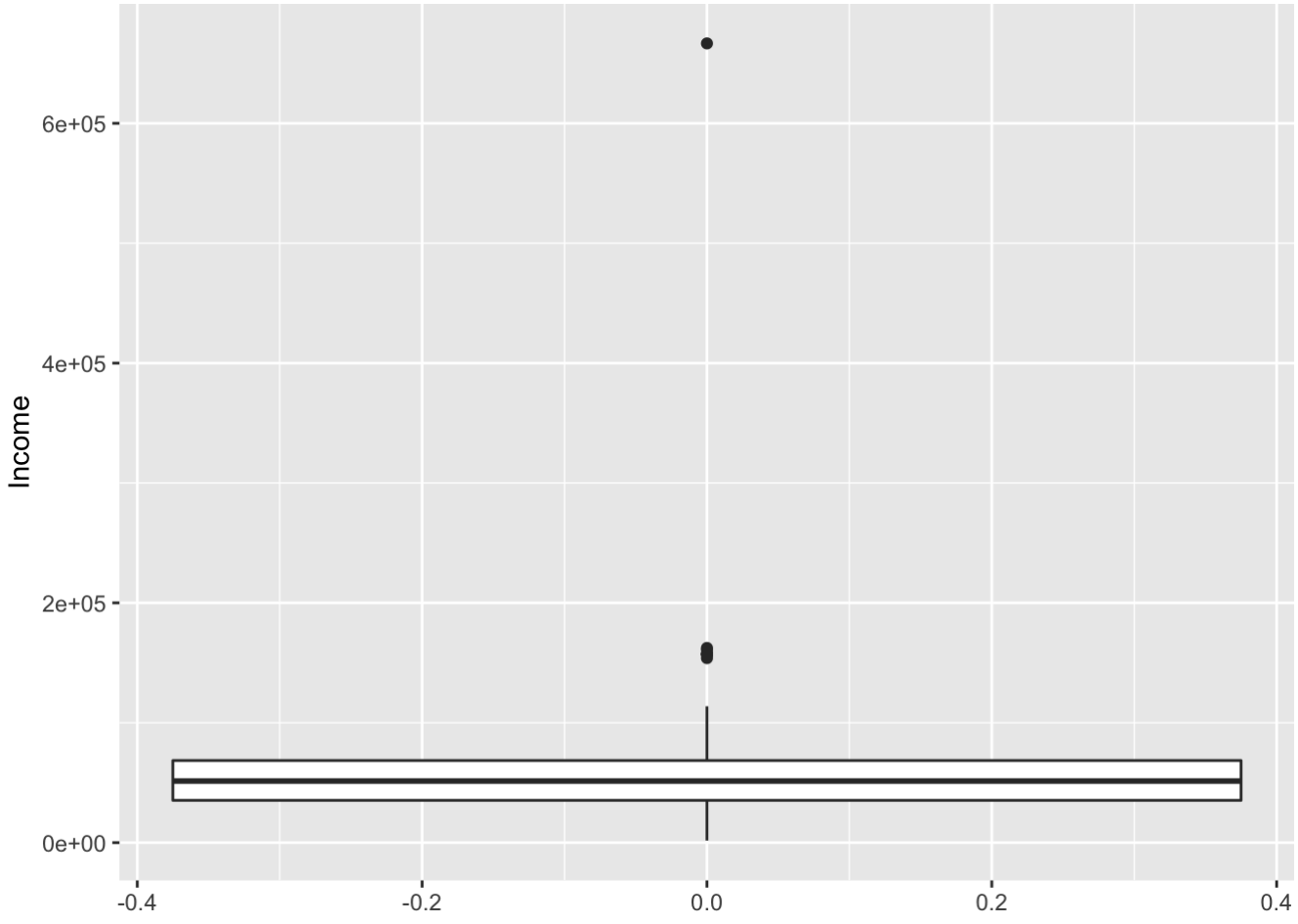
```
# create variable "Total_expense"
customers <- customers %>%
  mutate(Total_expense = MntMeatProducts +
                         MntFishProducts +
                         MntWines +
                         MntFruits +
                         MntSweetProducts +
                         MntGoldProds)
```

Use boxplot to find the outliers in the dataset.

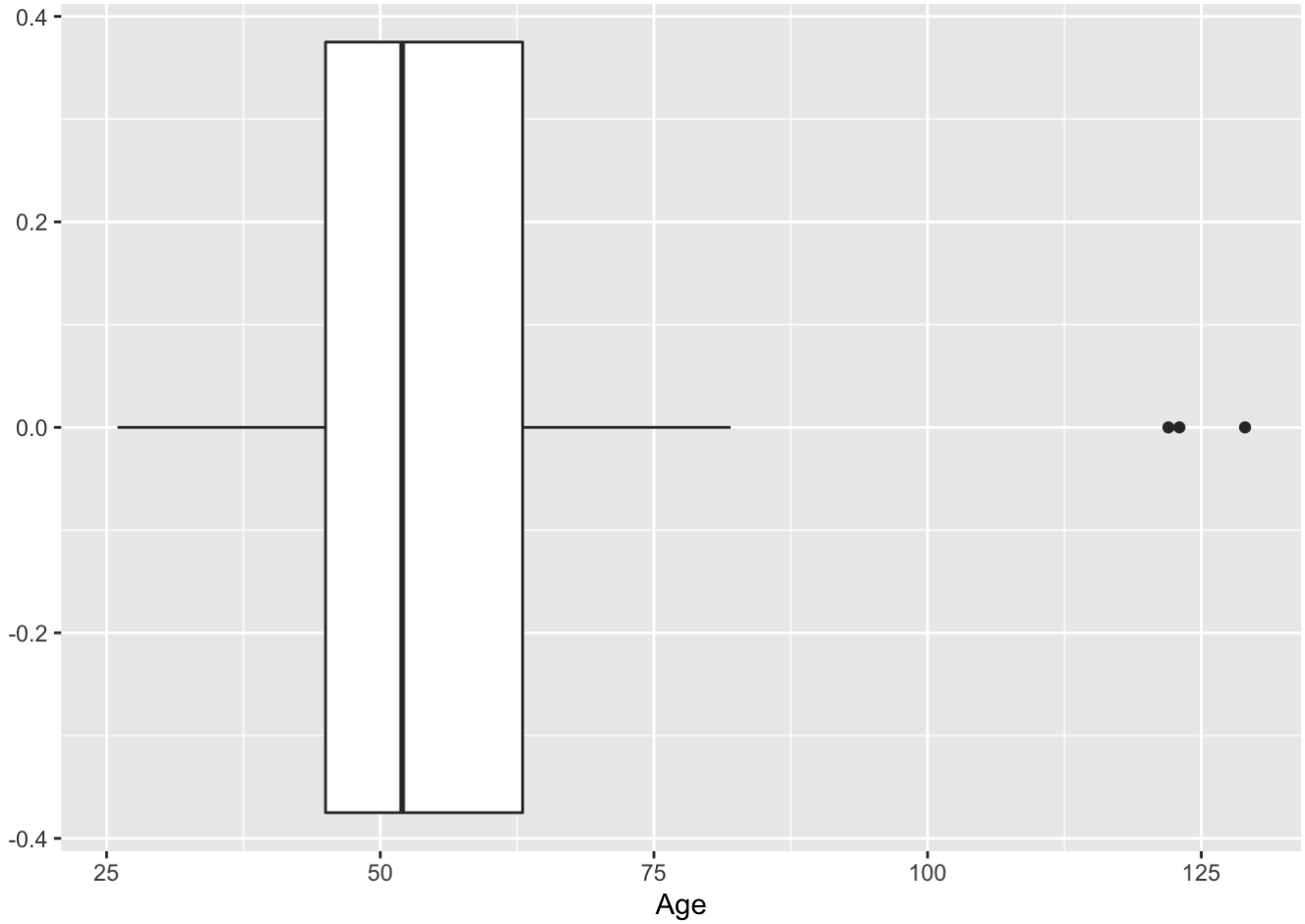
```
# create variable "Accepted"
customers <- customers %>%
  mutate(Accepted = AcceptedCmp1 +
                    AcceptedCmp2 +
                    AcceptedCmp3 +
                    AcceptedCmp4 +
                    AcceptedCmp5)
```

Plot the data to find the outliers

```
# Boxplot of the income
ggplot(customers, aes(y = Income)) + geom_boxplot()
```



```
# Boxplot of age
ggplot(customers, aes(Age)) + geom_boxplot()
```




```
# get the valuues of Q1, Q3, and IQR
summary(customers$Income)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1730   35303   51382   52247   68522  666666
```

```
IQR(customers$Income)
```

```
## [1] 33219
```

```
summary(customers$Age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   26.00   45.00   52.00   53.18   63.00  129.00
```

```
IQR(customers$Age)
```

```
## [1] 18
```

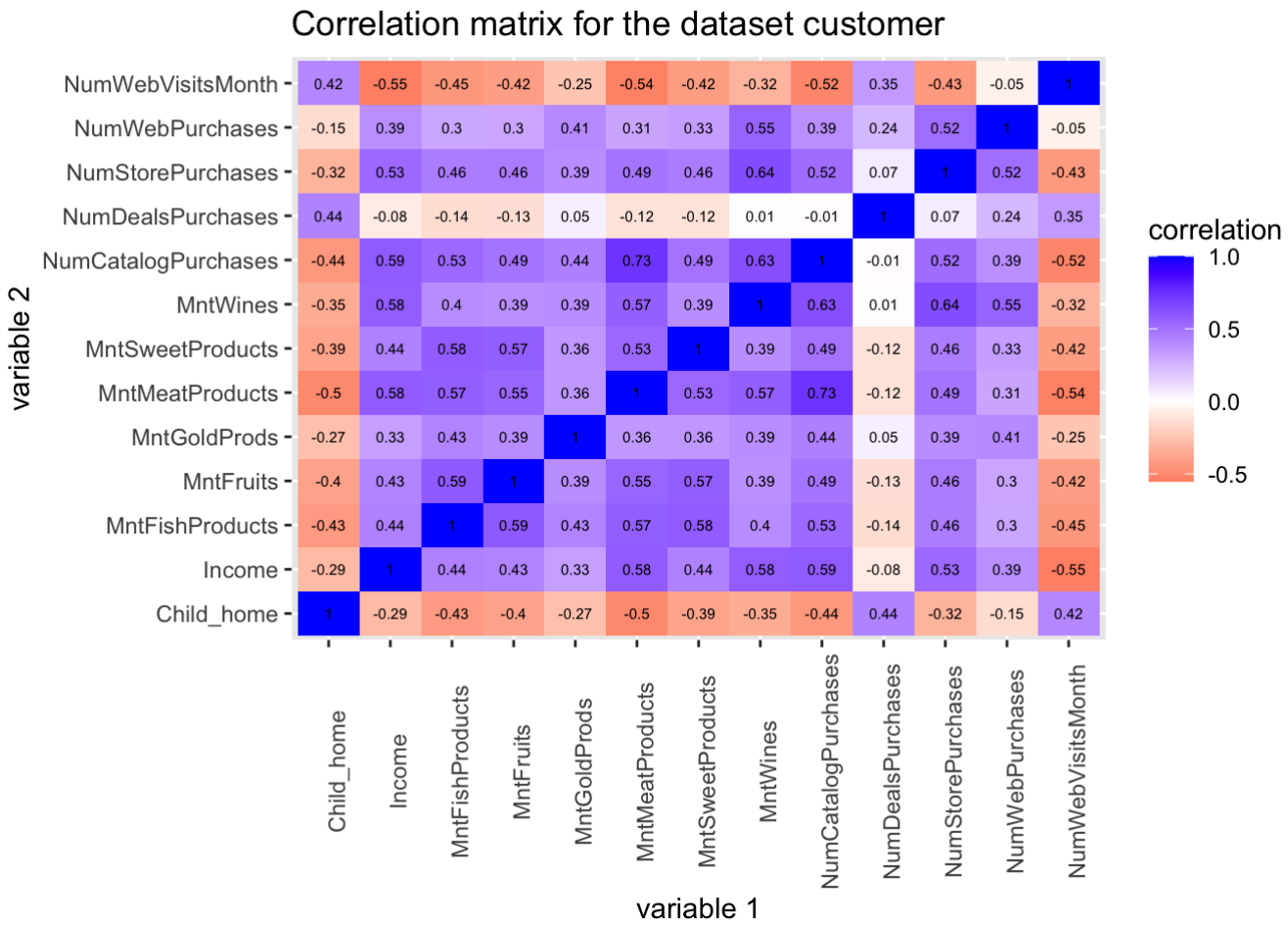
```
#remove the outliers of the dataset
customers <- customers %>%
  filter(Income < 1300000 & Age < 100)
```

Exploratory Data Analysis

```
# Find the correlations among the 14 variables
customers_nums <- customers %>%
  select(Income, Child_home, MntWines, MntFruits, MntMeatProducts,
         MntFishProducts, MntSweetProducts, NumDealsPurchases,
         MntGoldProds, NumWebPurchases, NumCatalogPurchases,
         NumStorePurchases, NumWebVisitsMonth) %>%
  as.data.frame()
```

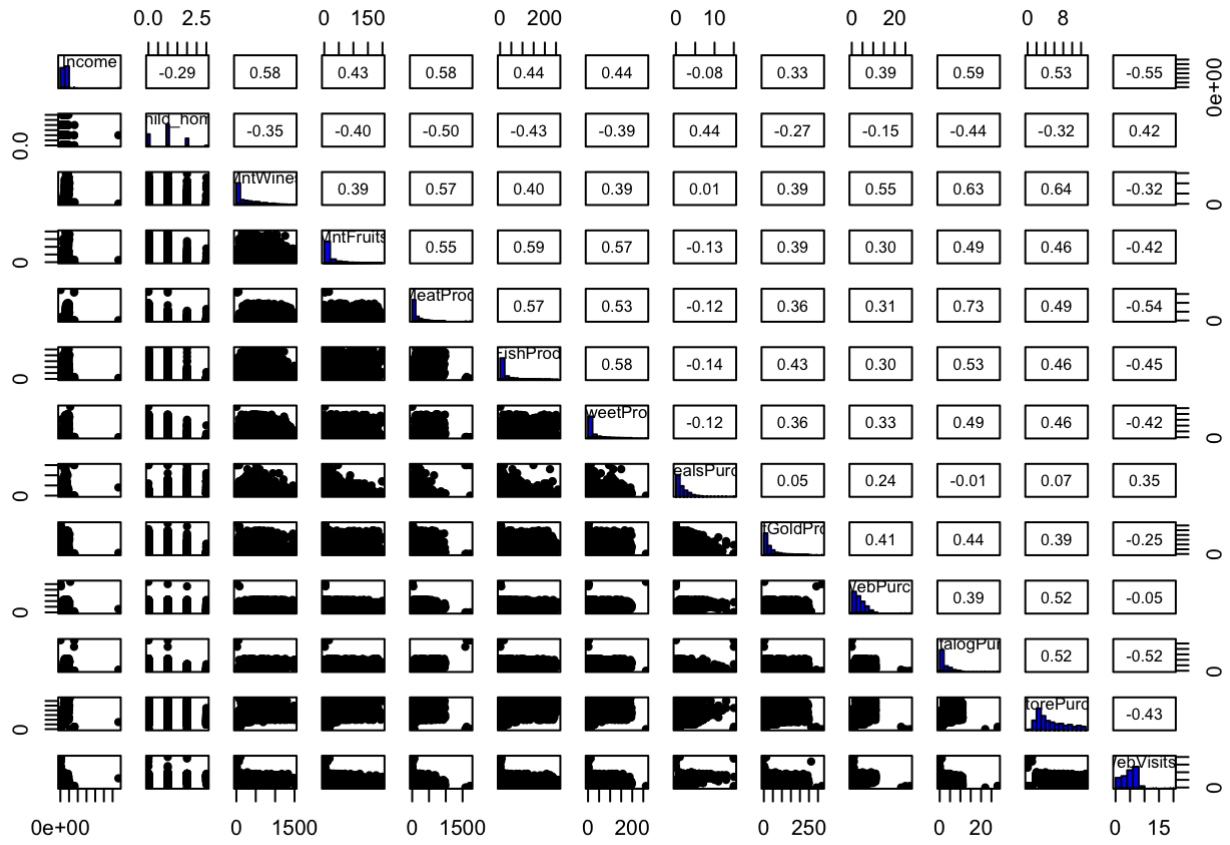
Heatmap with correlation matrix

```
# create correlation matrix
cor(customers_nums, use = "pairwise.complete.obs") %>%
  # Save as a data frame
  as.data.frame %>%
  # Convert row names to an explicit variable
  rownames_to_column %>%
  # Pivot so that all correlations appear in the same column
  pivot_longer(-1, names_to = "other_var", values_to = "correlation") %>%
  ggplot(aes(rowname, other_var, fill = correlation)) +
  # Heatmap with geom_tile
  geom_tile() +
  # Change the scale to make the middle appear neutral
  scale_fill_gradient2(low="red",mid="white",high="blue") +
  # Overlay values
  geom_text(aes(label = round(correlation,2)), color = "black", size = 2) +
  # Give title and labels
  labs(title = "Correlation matrix for the dataset customer", x = "variable 1", y = "var
iable 2") +
  theme(axis.text.x = element_text(angle = 90))
```



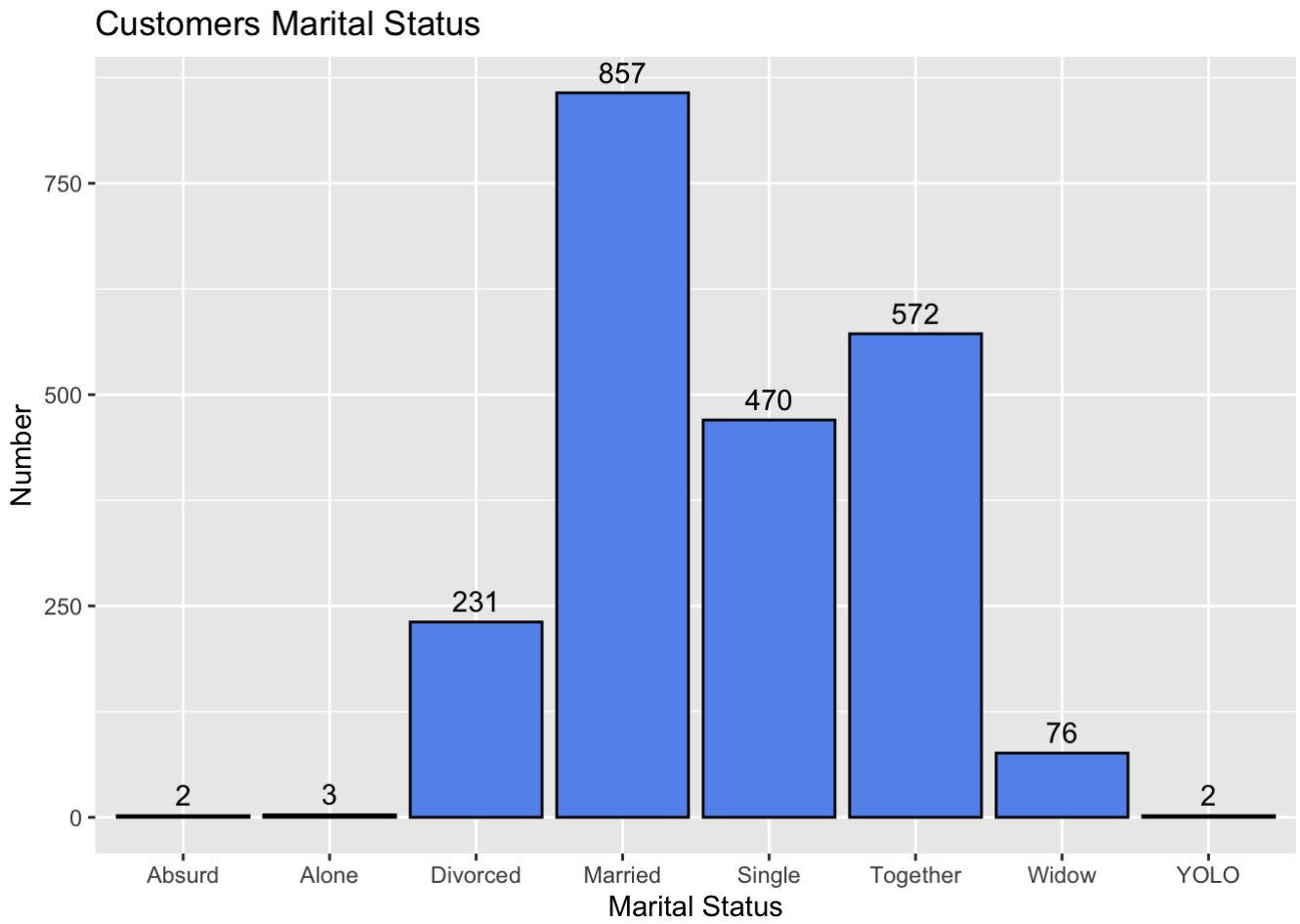
A correlation matrix with univariate and bivariate graphs

```
# A package for building a correlation matrix with univariate and bivariate graphs
# install.packages(psych)
library(psych)
pairs.panels(customers_nums,
             method = "pearson", # correlation coefficient method
             hist.col = "blue", # color of histogram
             smooth = FALSE, density = FALSE, ellipses = FALSE)
```



Marital status of customers in each category

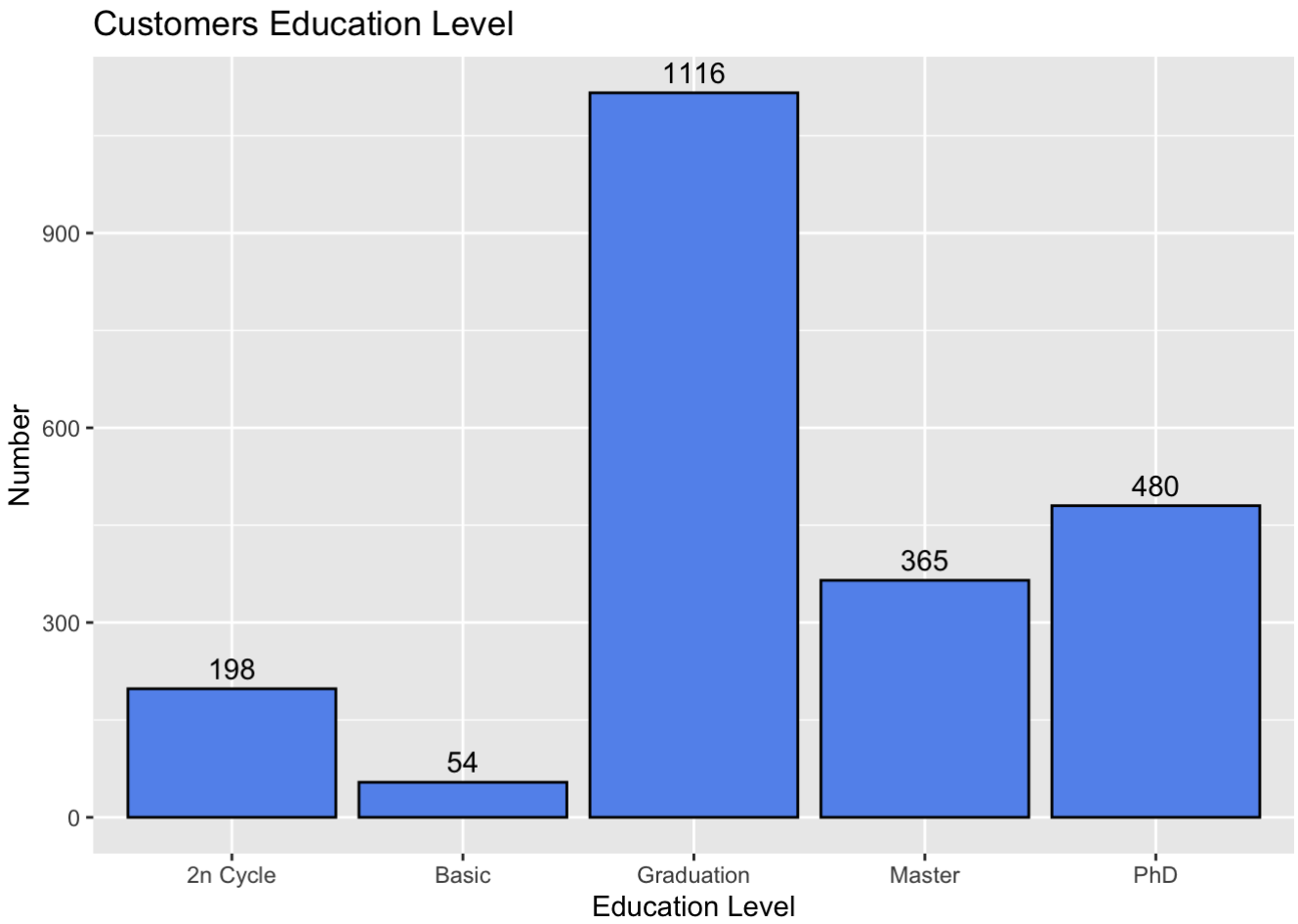
```
# calculate number of customers in each marital status category
plotdata <- customers %>% count(Marital_Status)

# plot the distribution of race
ggplot(plotdata, aes(x = Marital_Status, y = n)) +
  geom_bar(fill = "cornflowerblue", color="black", stat = "identity") +
  geom_text(aes(label = n), vjust=-0.5) +
  labs(x = "Marital Status", y = "Number", title = "Customers Marital Status")
```



Observations! There are 857 (38%) customers in the dataset are “Married”. There are 571 (25%) customers in the dataset are “Together”. There are 470 (21%) customers in the dataset are “Single”. There are 231 (10%) customers in the dataset are “Divorced”. There are 76 (3%) customers in the dataset are “Widow”. There is no significant amount of customers who are “Alone”,“Absurd”and “YOLO”. Most of our customers are in couple.

Education level of customers in each category

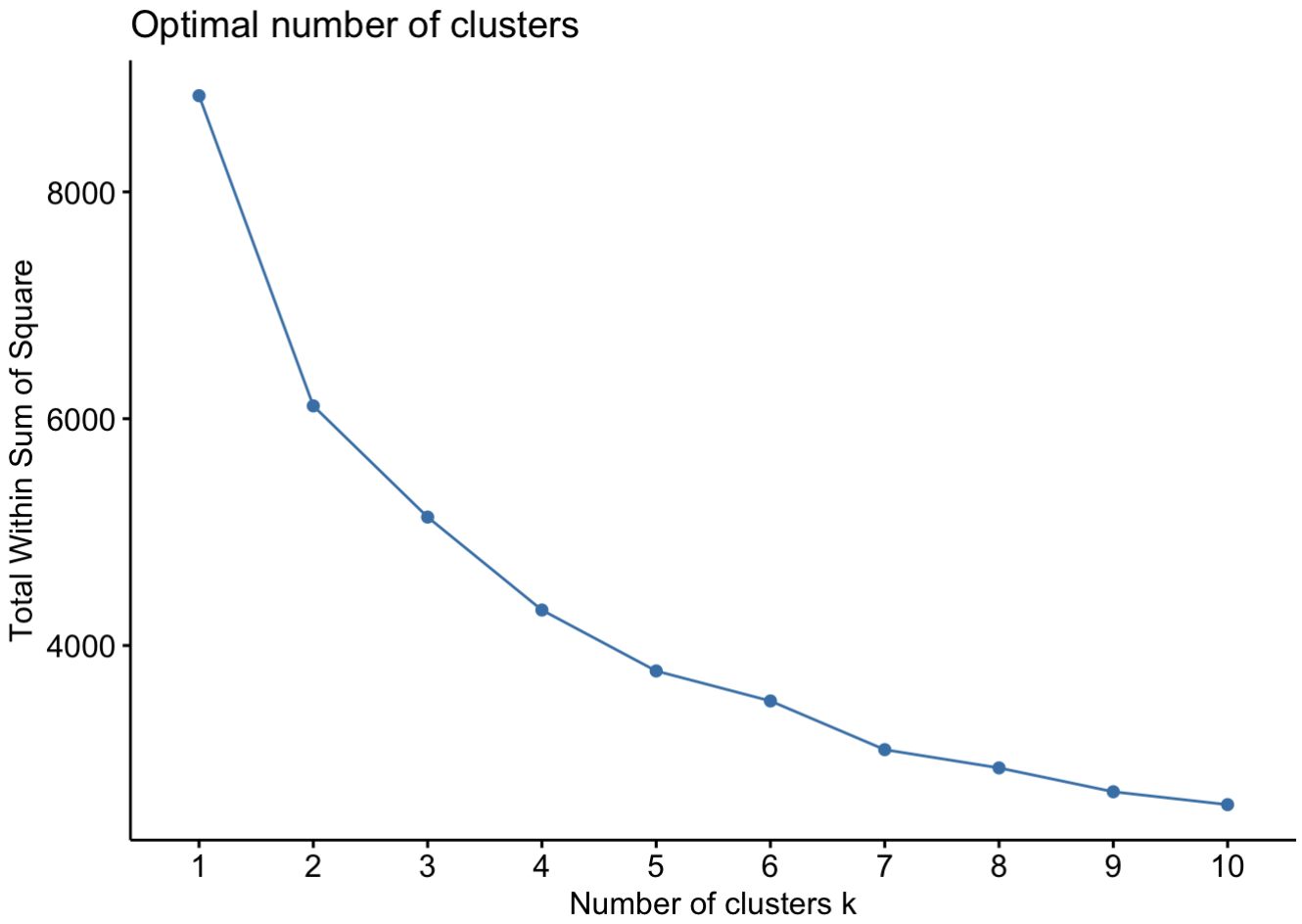
```
# calculate number of participants in each marital status category
plotdata <- customers %>% count(Education)

# plot the distribution of race
ggplot(plotdata, aes(x = Education, y = n)) +
  geom_bar(fill = "cornflowerblue", color="black", stat = "identity") +
  geom_text(aes(label = n), vjust=-0.5) +
  labs(x = "Education Level", y = "Number", title = "Customers Education Level")
```



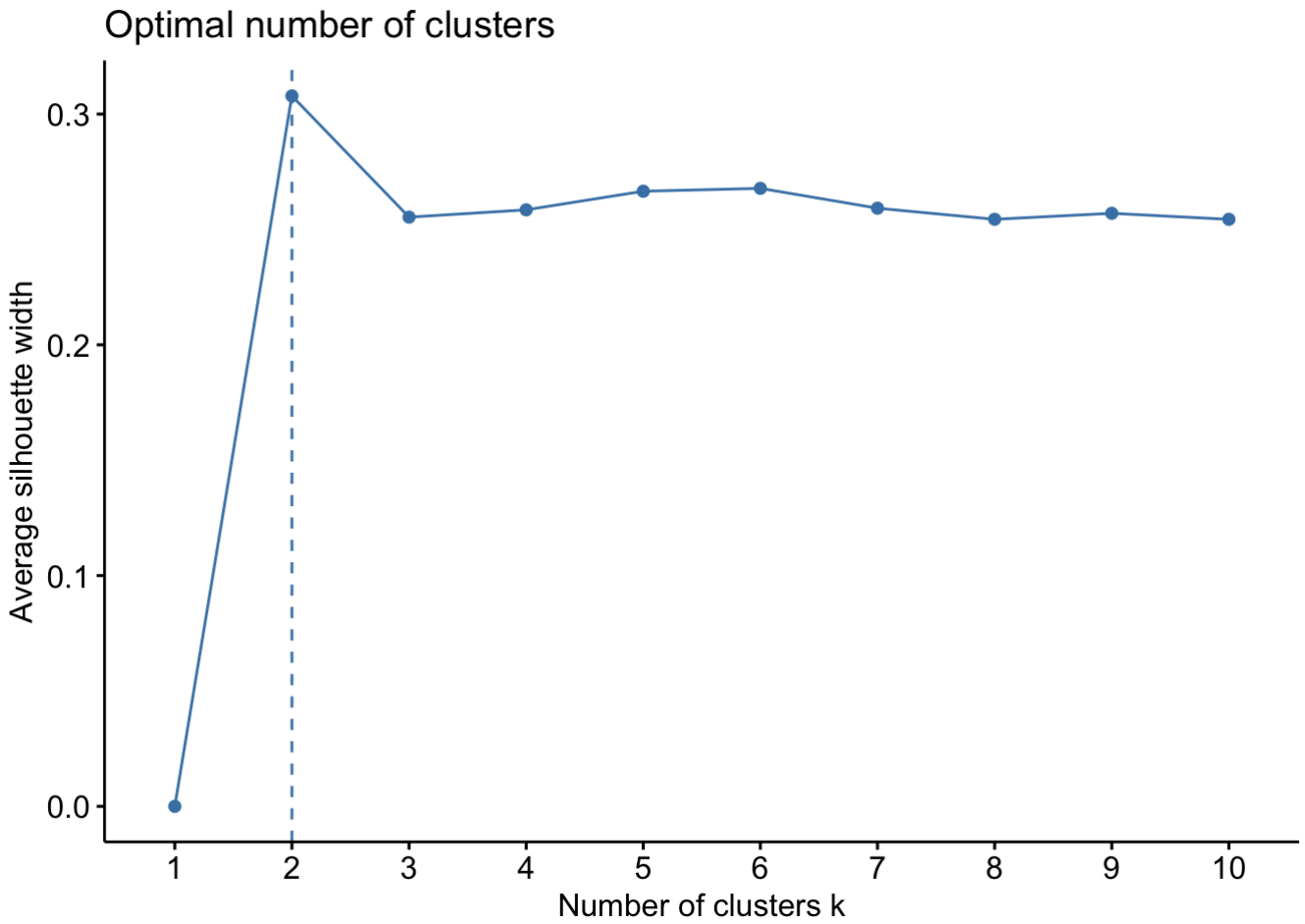
Observations! There are 1115 (50%) of customers in the dataset are 'Graduate'. There are 480 (21%) of customers in the dataset possess 'PhD' degree. There are 365 (16.5%) of customers in the dataset possess 'Master' degree. There are 198 (9%) of customers in the dataset possess '2n cycle'. There are 54 (2%) of customers in the dataset possess 'Basic' education.

Clustering

```
# scale the dataset and find the number of clusters needed
customers_scale = customers %>%
  select(Income,Recency,Age,Total_expense) %>%
  scale

# we choose to use two different method to validify the number of clusters
fviz_nbclust(customers_scale, kmeans, method = "wss")
```

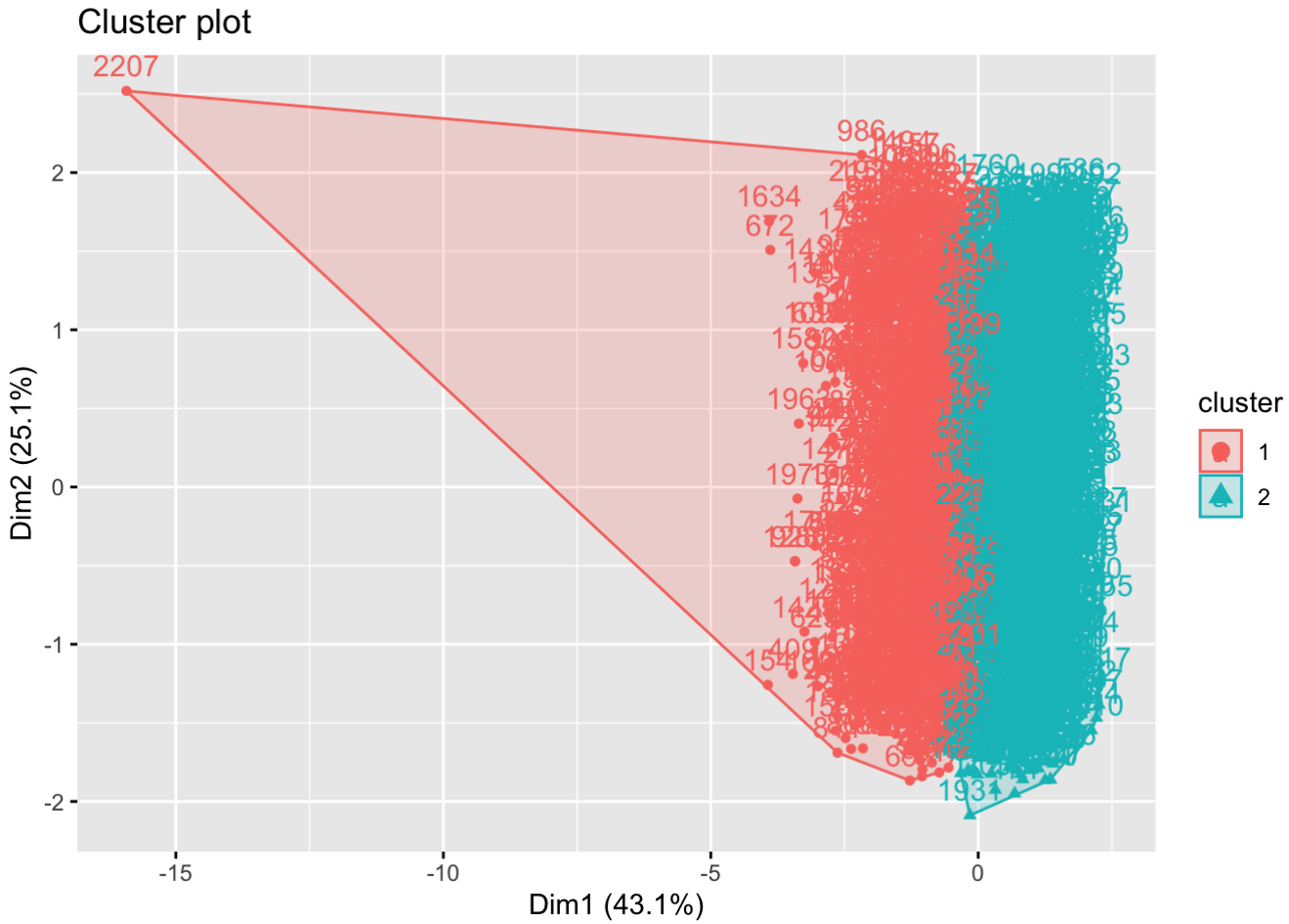



```
fviz_nbclust(customers_scale, kmeans, method = "silhouette")
```



Based on the graph, we can conclude that a number of cluster of 2 should be a good choice for this dataset. However, the average silhouette width is about 0.3, which is fairly low. This indicated that the structure of this dataset is weak and could be artificial.

```
# calculate kmeans result with previously decided number of clusters
kmeans_cus <- customers_scale %>% kmeans(2)
# visualization with `fviz_cluster()`
fviz_cluster(kmeans_cus, data = customers_scale)
```

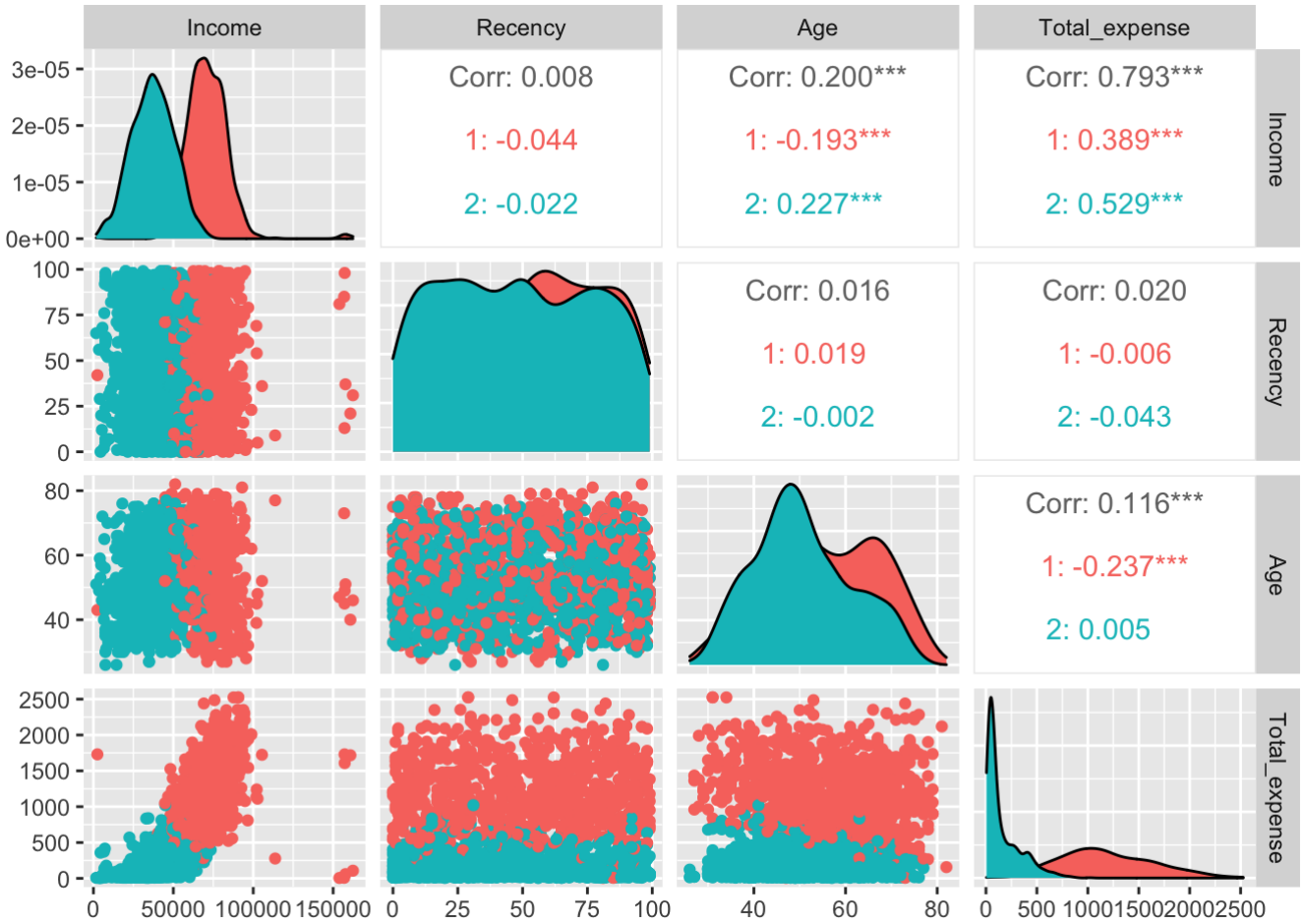


Based on the plot, there is one outlier that differentiate significantly from all other dataset. After examine it in the dataset, we decide to exclude it to get a better visualization.

```
customers_scale <- customers_scale[-c(2207),]
# calculate kmeans result with previously decided number of clusters
kmeans_cus <- customers_scale %>% kmeans(2)
# visualization with `fviz_cluster()`
fviz_cluster(kmeans_cus, data = customers_scale)
```

In this plot, the two clusters did not have a clear separation from each other. They have a heavily overlapping area and everything sort of remains in the center. To better understand these two clusters, we can perform a ggpair upon the dataset.

```
# we can also do a ggpair() to help us better visualize how things relate to each other
customer_pam <- customers_scale %>%
  pam(k = 2)

# we need to also remove the oulier in customer to match the updated dataset
customers <- customers[-c(2207),]

# add a new column cluster
customer_npam <- customers %>%
  mutate(cluster = as.factor(customer_pam$clustering))

# perform ggpair
library(GGally)
ggpairs(customer_npam, columns = c("Income","Recency","Age","Total_expense"), aes(color
 = cluster))
```



As we can see in the plot, there are several findings we can conclude. First of all, the only valid correlation among all four variables we choose is between Total expense and income, which has a correlation factor of 0.793. All the other factor do not have a strong correlation with one anther. Another interesting relationship we found is that when the recency is the same, the customers from cluster 1 always have a higher total expense compare to customers from cluster 2. This relationship is also shown in the Age vs. Total_expense plot. In general, customers from cluster 1 seems to spend more money compare to customers from cluster 2 when all other conditions are the same.

To further find details about these two clusters, we choose to calculate the average of each of these four variables to see if they have any difference.

```
customer_npam %>% group_by(cluster) %>%summarise(mean_Income = mean(Income))
```

```
## # A tibble: 2 × 2
##   cluster mean_Income
##   <fct>         <dbl>
## 1 1            71294.
## 2 2            37188.
```

```
customer_npam %>% group_by(cluster) %>%summarise(mean_Recency= mean(Recency))
```

```
## # A tibble: 2 × 2
##   cluster mean_Recency
##   <fct>          <dbl>
## 1 1               50.2
## 2 2               48.1
```

```
customer_npam %>% group_by(cluster) %>%summarise(mean_Age = mean(Age))
```

```
## # A tibble: 2 × 2
##   cluster mean_Age
##   <fct>      <dbl>
## 1 1           56.2
## 2 2           50.7
```

```
customer_npam %>% group_by(cluster) %>%summarise(mean_Total_expense= mean(Total_expens
e))
```

```
## # A tibble: 2 × 2
##   cluster mean_Total_expense
##   <fct>                <dbl>
## 1 1                    1187.
## 2 2                     165.
```

Despite the little different in the mean age and recency, we found a interesting relationship in the mean income and total expense. In general, the mean income of customers from cluster 1 is way higher than customers from cluster 2. The actual income from cluster 1 is about twice the number compare to that from cluster 2. This make sense and gives a reaonable explanation for why on the previous ggpairs customers from cluster 1 always spend more compare to customers from cluster 2. Another interesting fact we found is that the average total expense for customers in cluster 1 is also way higher compare to that from cluster 2. But instead of two times higher like it was in the income, the total expense from cluster 1 is 10 times higher compare to cluster 2.

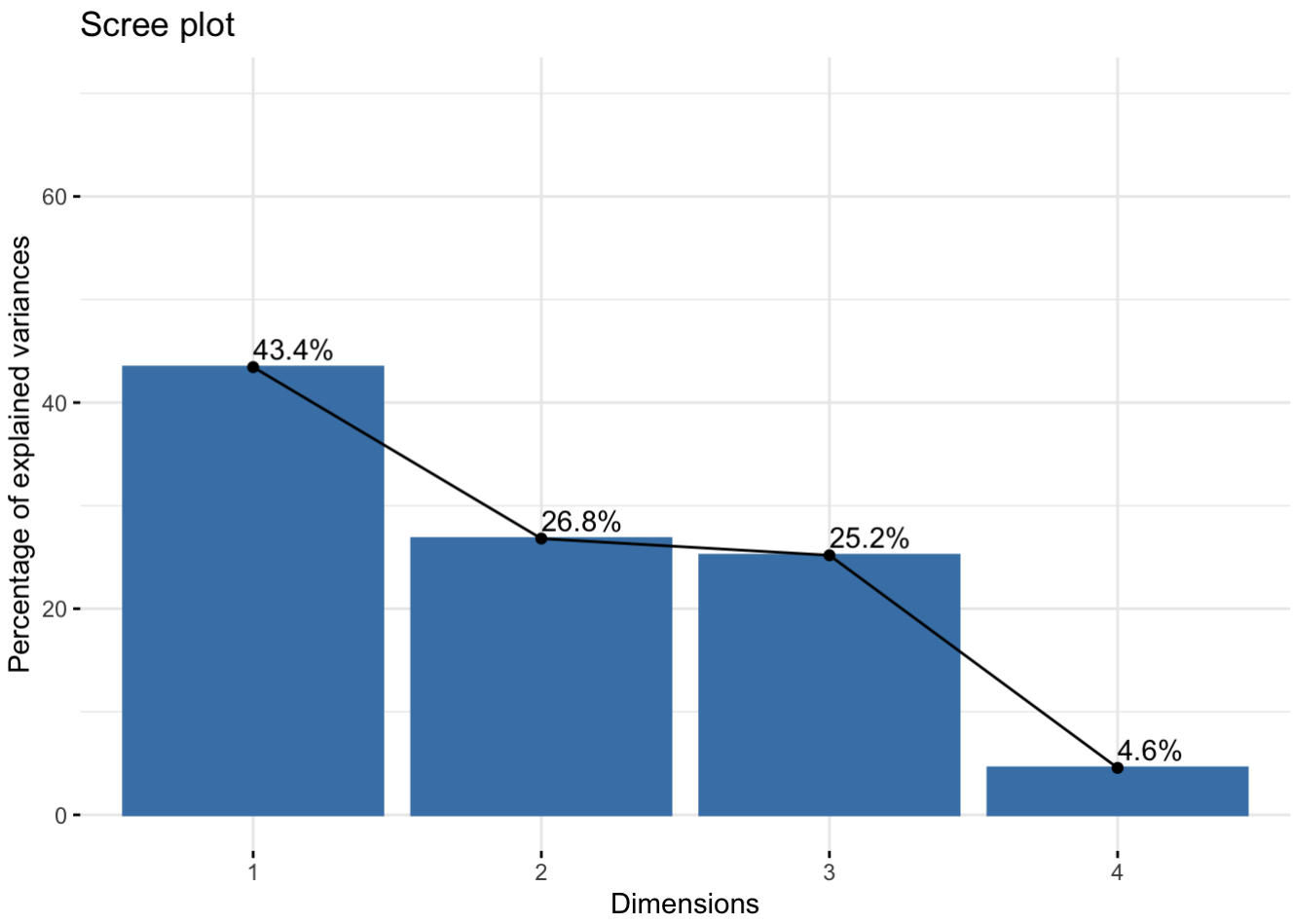
Dimensionality reduction

In this part, we choose to conduct PCA over the same four variables we did before to see if there is any interesting finding

```
# first conduct PCA using prcomp()
pca_customers <- customers_scale %>%
  prcomp()

# Look at percentage of variance explained for each PC in a table
get_eigenvalue(pca_customers)
```

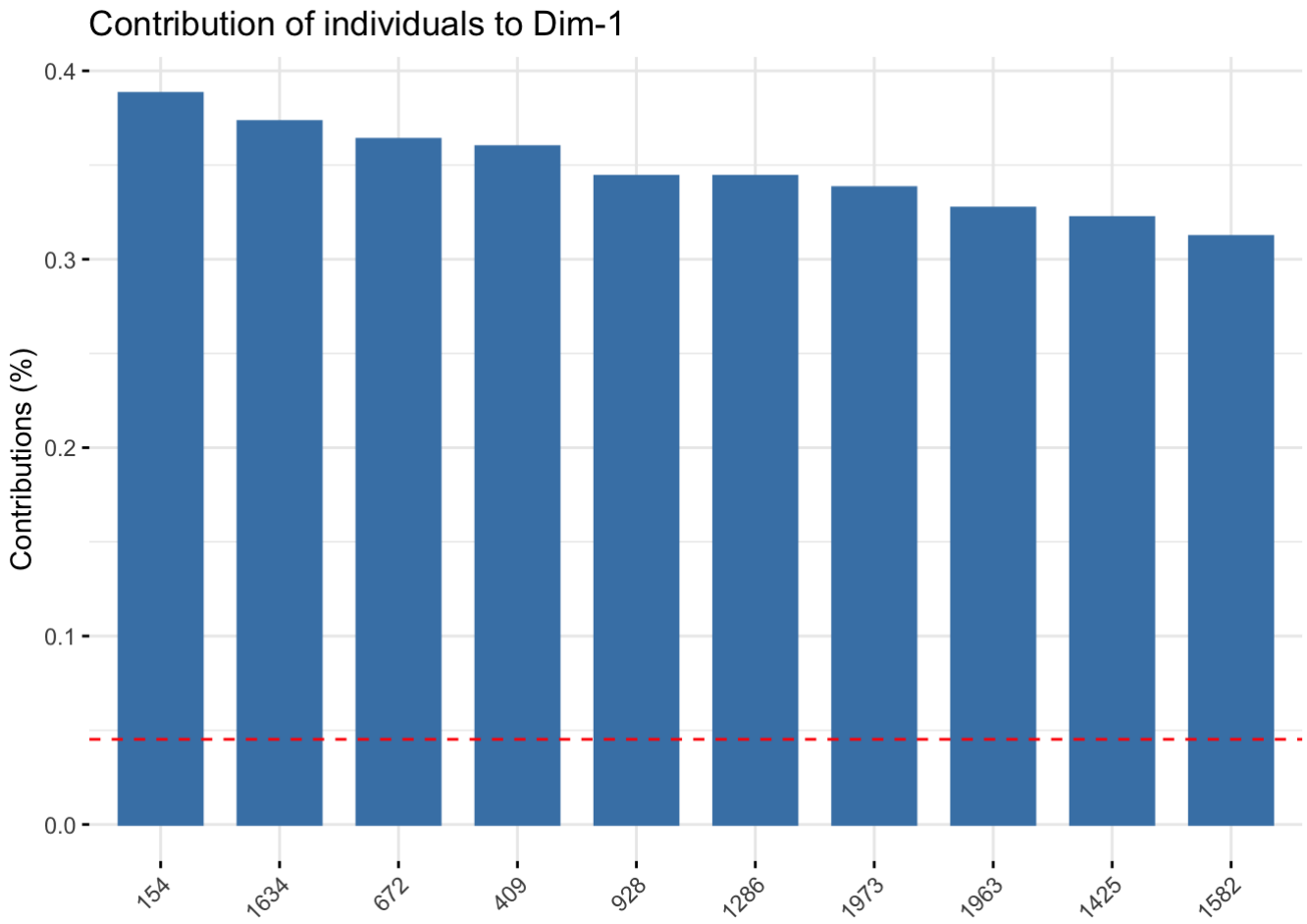
```
##       eigenvalue variance.percent cumulative.variance.percent
## Dim.1  1.6210392        43.443176                    43.44318
## Dim.2  1.0003840        26.809874                    70.25305
## Dim.3  0.9398442        25.187434                    95.44048
## Dim.4  0.1701339         4.559516                   100.00000
```

```
# Visualize percentage of variance explained for each PC in a scree plot
fviz_eig(pca_customers, addlabels = TRUE, ylim = c(0, 70))
```

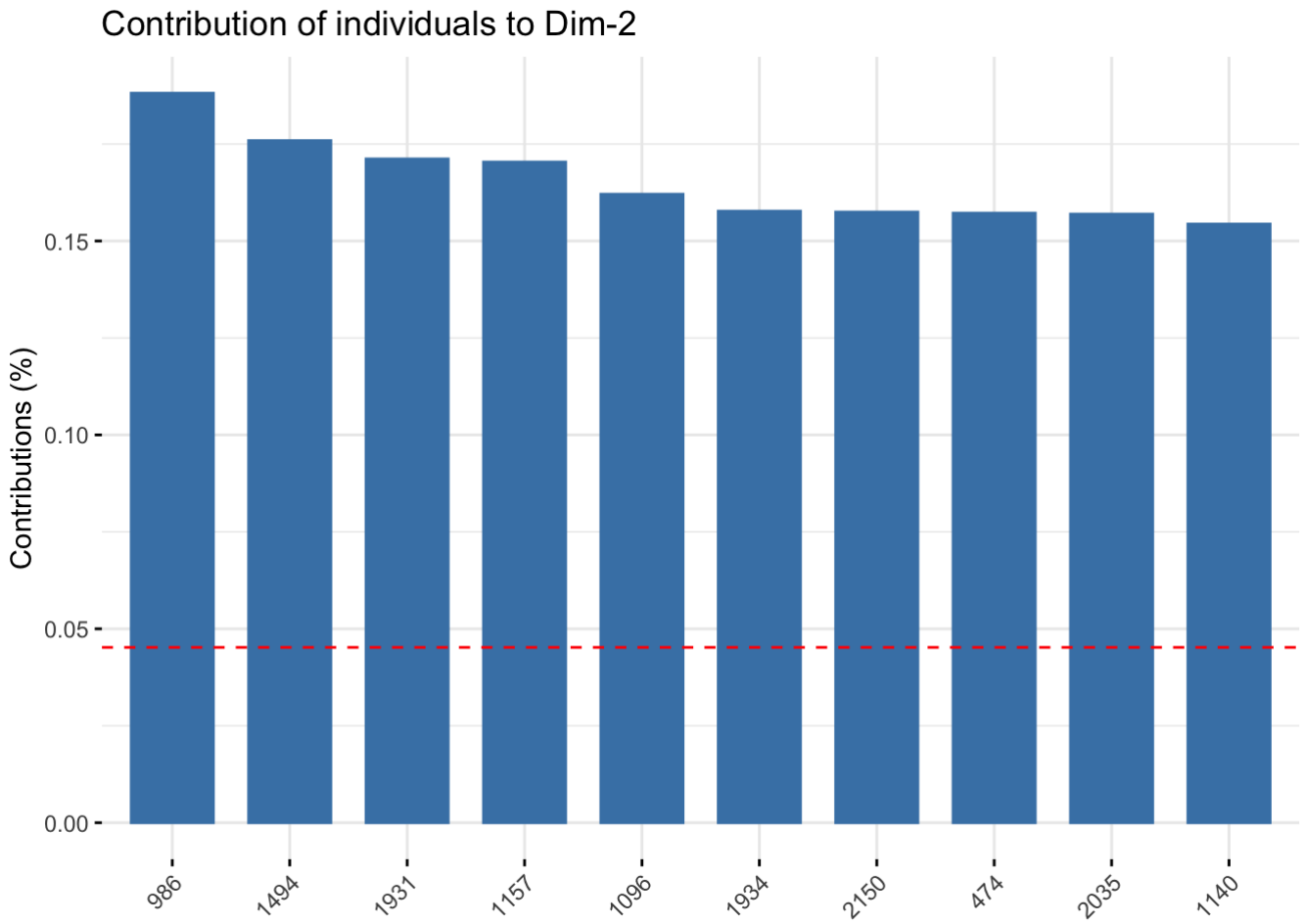


As we can see in the plot, in order to keep about 80% of the variance, first 3 components should be kept. To further find out information about each of the PCs, we can visualize the contribution of each individual customers for each PC.

```
# We can find the contributions within each of the first 2 PCs
fviz_contrib(pca_customers, choice = "ind", axes = 1, top = 10)
```

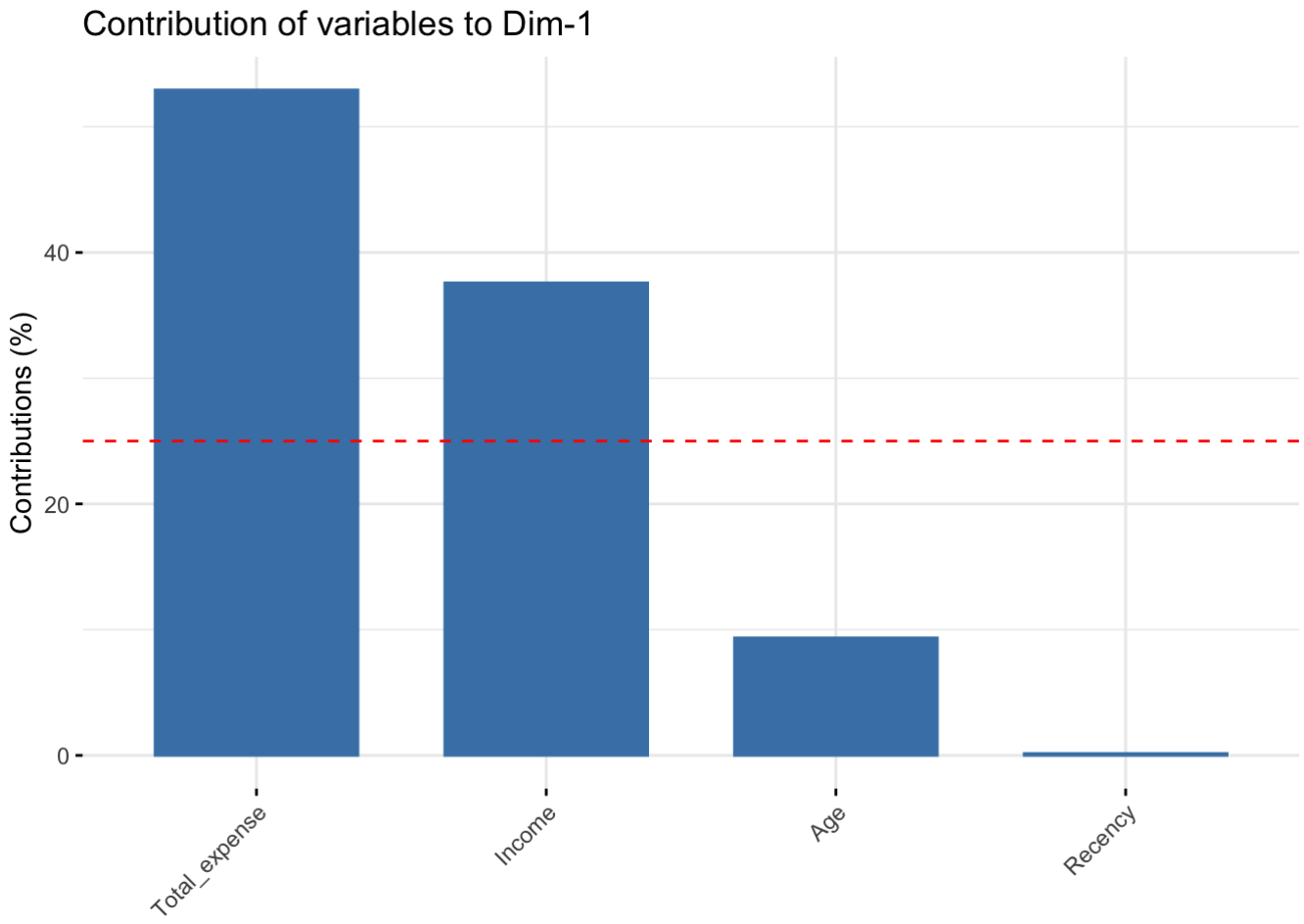



```
fviz_contrib(pca_customers, choice = "ind", axes = 2, top = 10)
```

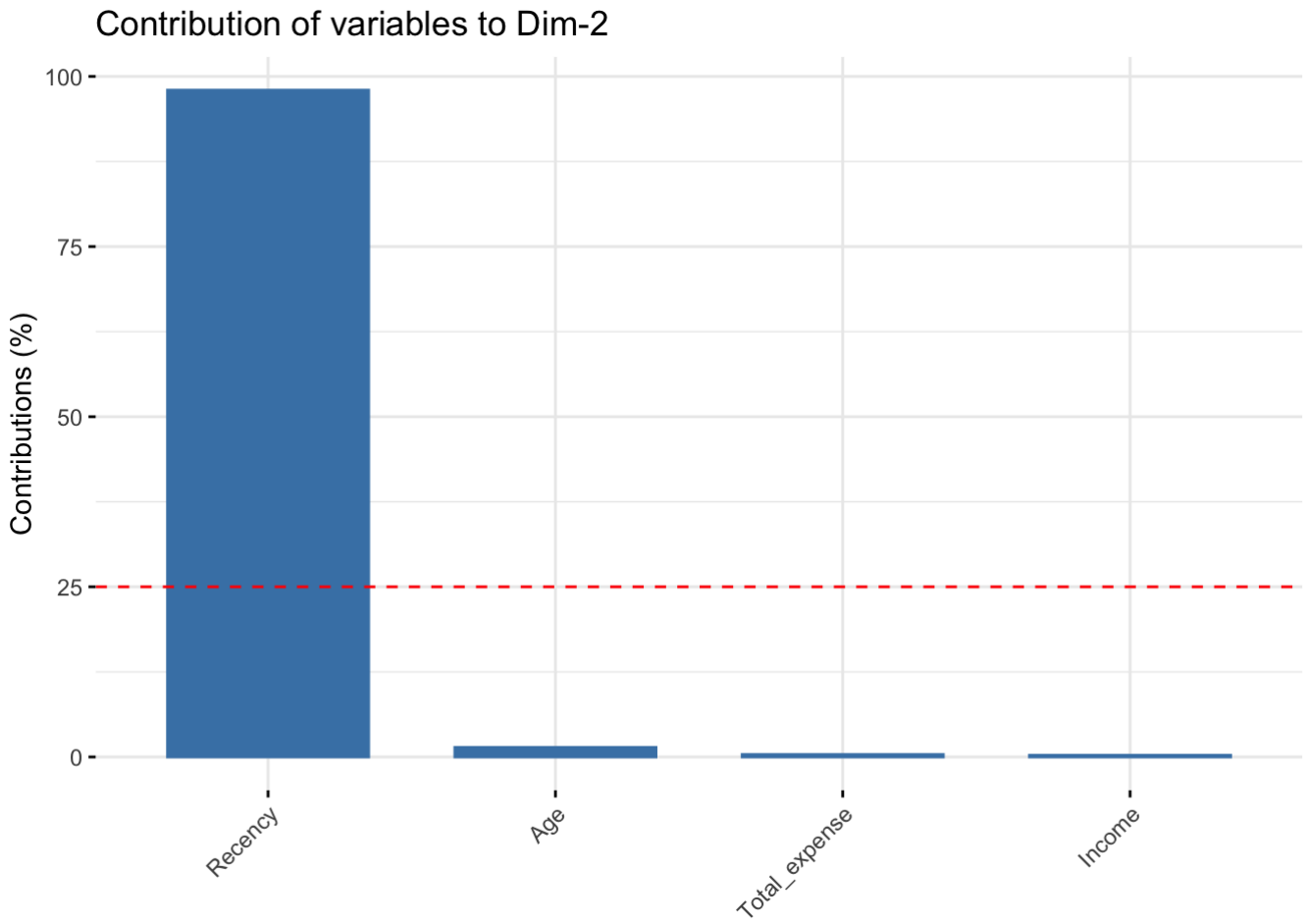


As we can see, the average contribution for each individuals are very low, which is around 0.5%. The top 10 contribution for each PCs are quite different are all way above the average line. We can keep on discussing these two PCs by visualizing the contribution of variables

```
# Visualize the 5 top contributions of the variables to the PCs as a percentage
fviz_contrib(pca_customers, choice = "var", axes = 1, top = 5) # on PC1
```



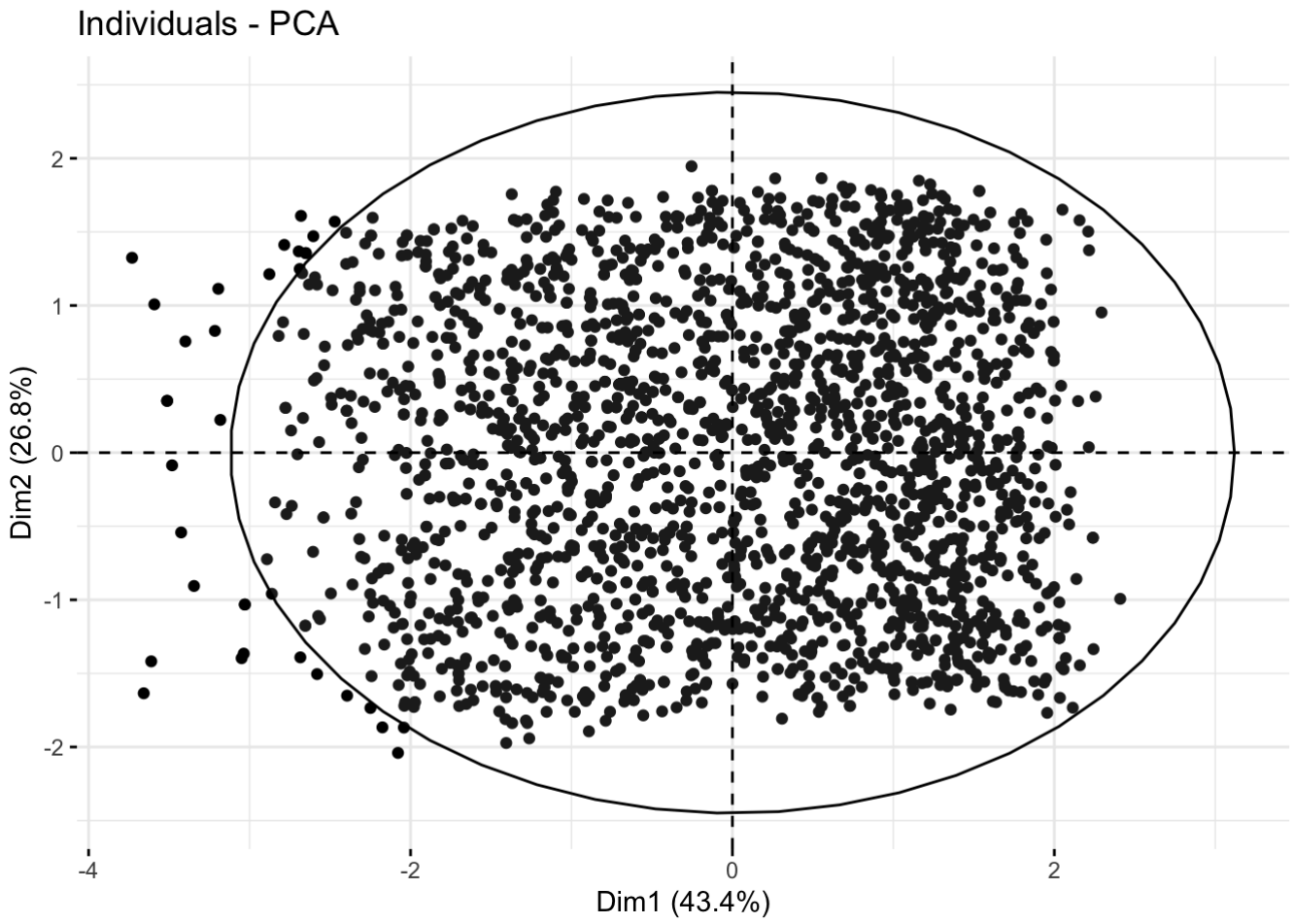

```
fviz_contrib(pca_customers, choice = "var", axes = 2, top = 5) # on PC2
```



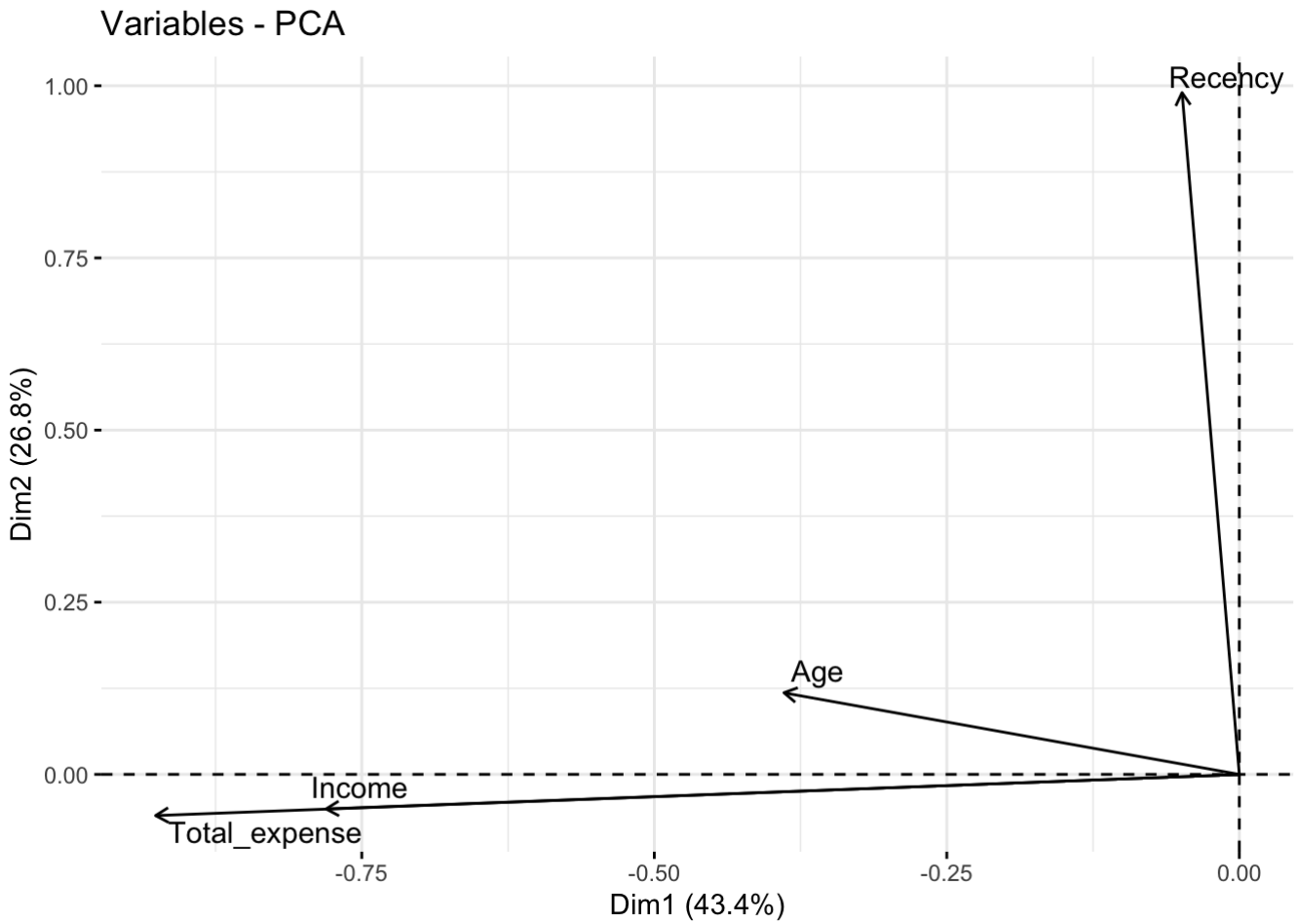
Based on the graph, we can see that for PC1, total expense and income contribute the most, which is about 50% and 35% respectively. On the other hand, Recency has the most contribution for PC2, which is about 98%.

Now we can visualize all the individuals and variables together to have a better view.

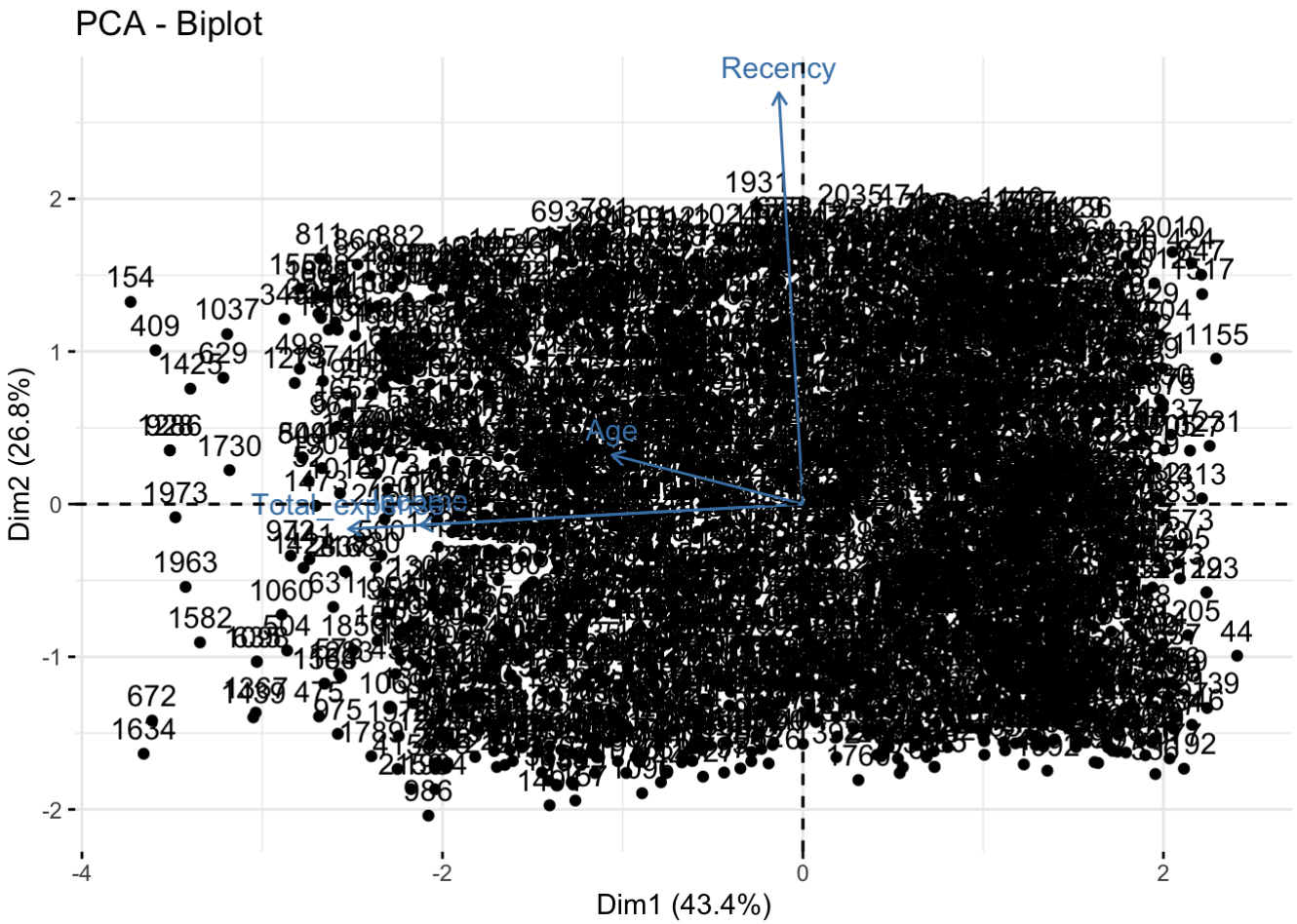
```
# Visualize the individuals according to PC1 and PC2
fviz_pca_ind(pca_customers,
             geom.ind = "point", # show points only (nbut not "text")
             palette = c("#00AFBB", "#E7B800", "#FC4E07"),
             addEllipses = TRUE, # Concentration ellipses
             legend.title = "outcome"
             )
```




```
# Visualize the contributions of the variables to the PCs in a correlation circle
fviz_pca_var(pca_customers, col.var = "black",
             repel = TRUE) # Avoid text overlapping
```




```
fviz_pca_biplot(pca_customers)
```



As shown in the cirlce plot, we can see that both income and total expense has a highly negative relationship with dimension 1, but they both have a weak relationship with dimension 2. On the other hand, recency has a highly positive relationship with dimension 2, but a weak relationship with dimension 1. The plot also tells that total expense and the income are highly conrrelated, whereas recency is not correlated with any other 3 elements. These relationship is validated based on the early contribution plot that showed the contribution for each dimension.

Classification and Cross-validation

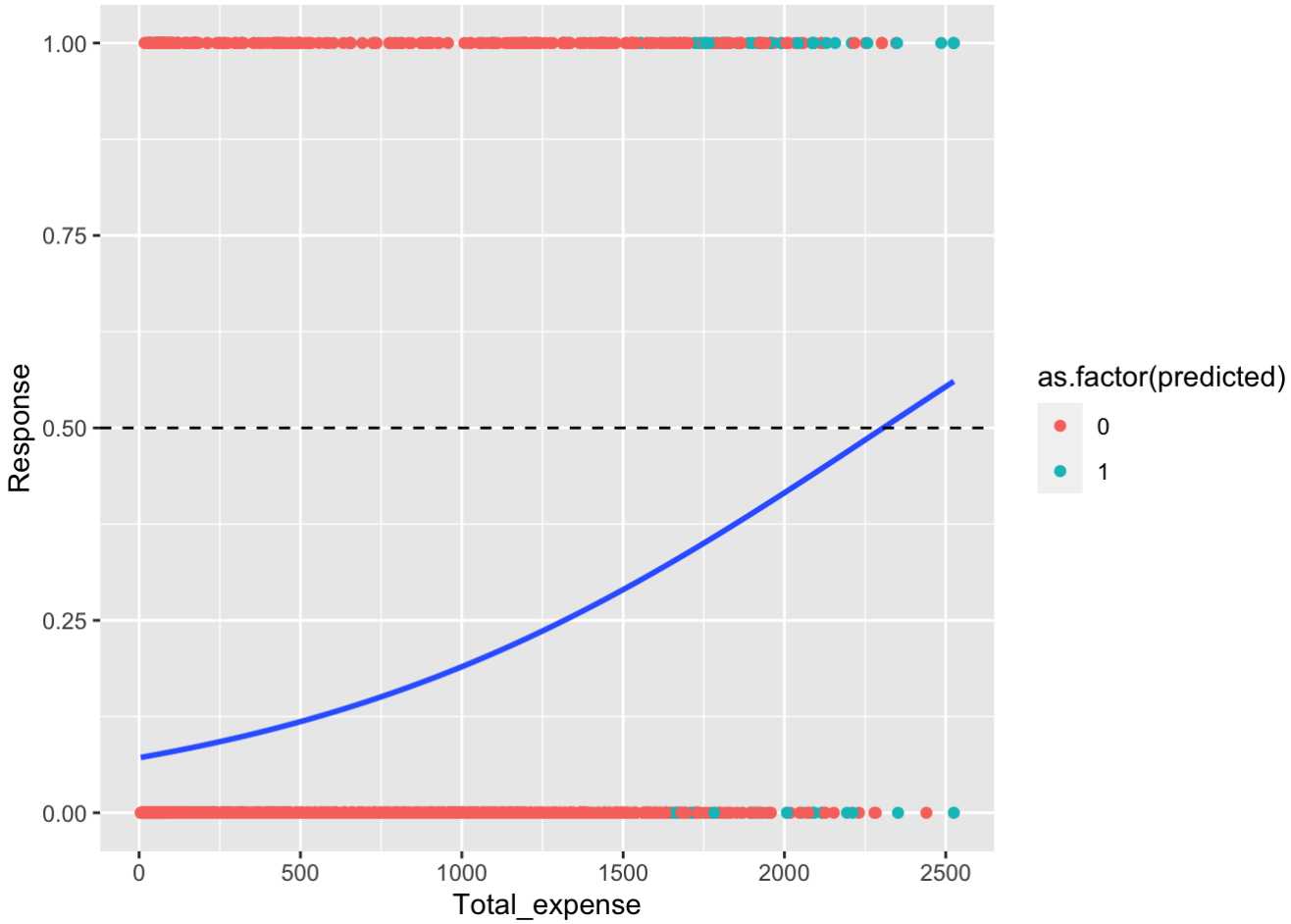
Fit a logistic regression model

```
fit <- glm(Response ~ Income + Recency + Age + Total_expense, data = customers, family =
"binomial")
summary(fit)
```

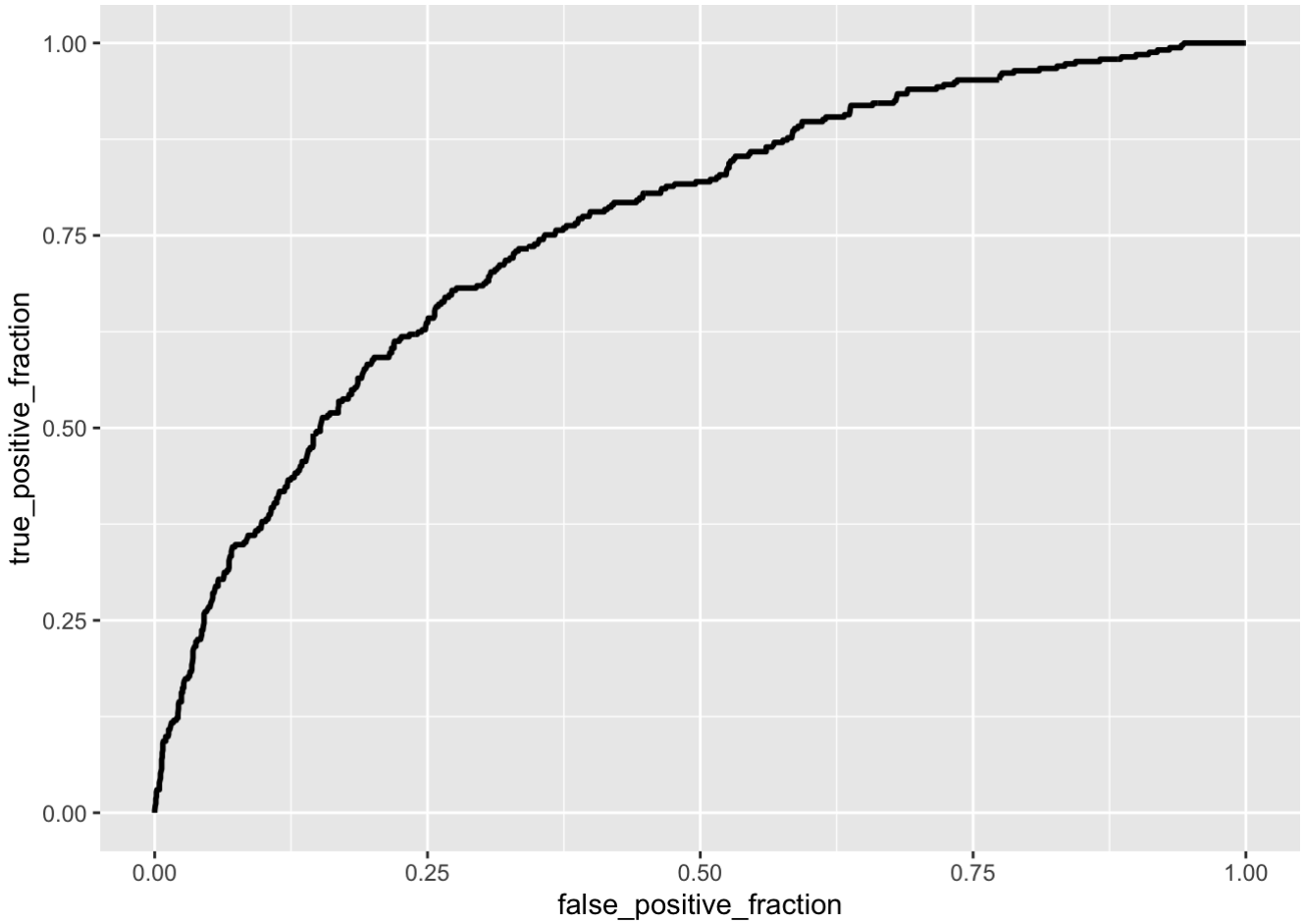
```
##
## Call:
## glm(formula = Response ~ Income + Recency + Age + Total_expense,
##     family = "binomial", data = customers)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6929  -0.5752  -0.4133  -0.2649   2.6915
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -5.870e-01  3.381e-01  -1.736  0.08251 .
## Income        -1.763e-05  5.361e-06  -3.288  0.00101 **
## Recency       -2.351e-02  2.397e-03  -9.806  < 2e-16 ***
## Age           -8.026e-03  5.418e-03  -1.481  0.13848
## Total_expense  1.693e-03  1.757e-04   9.636  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1874.2  on 2211  degrees of freedom
## Residual deviance: 1612.0  on 2207  degrees of freedom
## AIC: 1622
##
## Number of Fisher Scoring iterations: 5
```

```
# Calculate a predicted probability
log_customers <- customers %>%
  mutate(score = predict(fit, type = "response"),
         predicted = ifelse(score < 0.5, 0, 1)) %>%
  select(Total_expense, Response, score, predicted)
```

```
# Visualize predicted and actual transmissions
ggplot(log_customers, aes(Total_expense, Response)) +
  geom_point(aes(color = as.factor(predicted))) +
  geom_smooth(method = "glm", se = FALSE,
              method.args = list(family = "binomial")) +
  ylim(0,1) +
  geom_hline(yintercept = 0.5, lty = 2)
```




```
ROC <- ggplot(log_customers) +
  geom_roc(aes(d = Response, m = score), n.cuts = 0)
ROC
```



calculate ROC

```
# Calculate the area under the curve with function calc_auc()
calc_auc(ROC)
```

```
##   PANEL group       AUC
## 1     1    -1 0.7606547
```

Separate the set into a training set to train the model and a test set to test the model

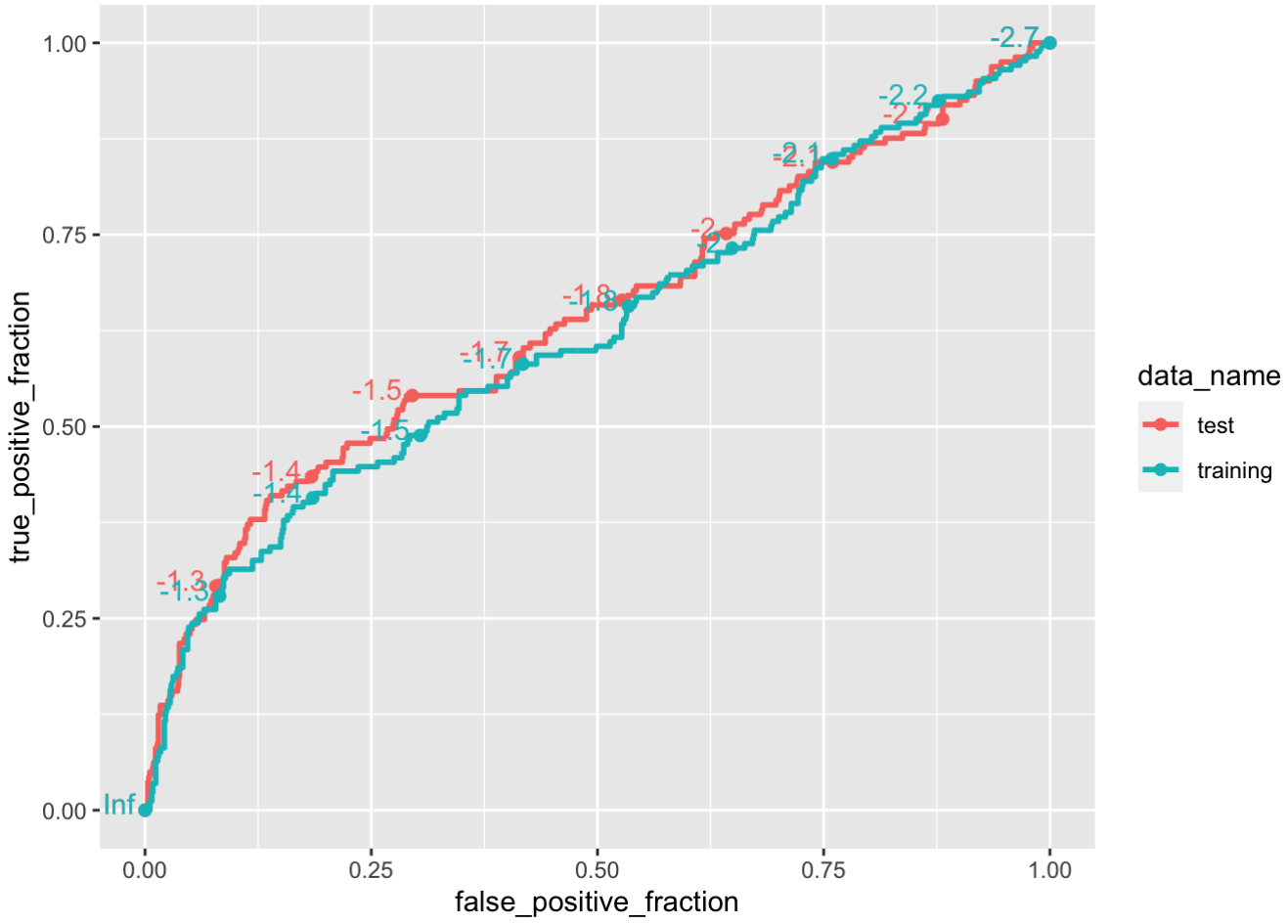
```
# Select a fraction of the data for training purposes
train <- sample_frac(customers, size = 0.5)

# Select the rest of the data for the test dataset
test <- anti_join(customers, train, by = "ID")
```

Then we fit the model on the train set

```
# Fit a logistic model in the
fit <- glm(Response ~ Income, data = train, family = "binomial")

# Results in a data frame for training data
df_train <- data.frame(
  probability = predict(fit, newdata = train),
  outcome = train$Response,
  data_name = "training")

# Results in a data frame for test data
df_test <- data.frame(
  probability = predict(fit, newdata = test),
  outcome = test$Response,
  data_name = "test")

# Combined results
df_combined <- rbind(df_train, df_test)
```

Let's evaluate the performance of our classifier on the train and test sets

```
ROC <- ggplot(df_combined) +
  geom_roc(aes(d = outcome, m = probability, color = data_name, n.cuts = 0))
ROC
```



The receiver operating characteristic (ROC) curve evaluates the performance of binary classification algorithms. It provides a graphical representation of a classifier's performance. From the graph above, we can learn the factor income is not the main factor to predict the response of customers. We also learn that the variables "Response" and "Income" have low correlations.

We can compare the AUC based on the train and test set

```
# Compare test and training AUCs
calc_auc(ROC)
```

```
##   PANEL group       AUC
## 1     1     1 0.6394558
## 2     1     2 0.6221428
```

k-fold cross-validation

```
# Choose number of folds
k = 10

# Randomly order rows in the dataset
data <- customers[sample(nrow(customers)), ]

# Create k folds from the dataset
folds <- cut(seq(1:nrow(data)), breaks = k, labels = FALSE)
```

Fit the model and repeat the process for each k-fold

```
# Use a for loop to get diagnostics for each test set
diags_k <- NULL

for(i in 1:k){
  # Create training and test sets
  train <- data[folds != i, ] # all observations except in fold i
  test <- data[folds == i, ]  # observations in fold i

  # Train model on training set (all but fold i)
  fit <- glm(Response ~ Income, data = train, family = "binomial")

  # Test model on test set (fold i)
  df <- data.frame(
    probability = predict(fit, newdata = test, type = "response"),
    outcome = test$Response)

  # Consider the ROC curve for the test dataset
  ROC <- ggplot(df) +
    geom_roc(aes(d = outcome, m = probability, n.cuts = 0))

  # Get diagnostics for fold i (AUC)
  diags_k[i] <- calc_auc(ROC)$AUC
}
```

Finally, find the average performance on new data

```
# Average performance
mean(diags_k)
```

```
## [1] 0.6269806
```

```
##
sysname
##
"Darwin"
##
release
##
"21.5.0"
##
version
## "Darwin Kernel Version 21.5.0: Tue Apr 26 21:08:22 PDT 2022; root:xnu-8020.121.3~4/RE
LEASE_X86_64"
##
nodename
##                                                                                "MacBo
ok-Pro.local"
##
machine
##
"x86_64"
##
login
##
"root"
##
user
##
"zixiangmeng"
##                                                                                    e
ffective_user
##
"zixiangmeng"
```