Principal Component Analysis

10/7/2020

Loading Necessary Libraries

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

```
## corrplot 0.84 loaded
```

```
library(tidyverse)
```

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

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

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

```
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
## The following object is masked from 'package:purrr':
##
##     some
```

```
library(factoextra)
```

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

Loading the Lending Dataset

Loading Lending_Data.csv. Making a copy and attaching the copy:

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

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

```
##  $ delinq_2yrs        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ loan_amnt          : num  7000 2000 12000 9350 6000 ...
##  $ Bins_loan_amt      : num  6 2 10 8 5 8 5 10 2 8 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   member_id = col_character(),
##   ..   loan_status = col_character(),
##   ..   int_rate = col_character(),
##   ..   Bin_int = col_double(),
##   ..   dti = col_double(),
##   ..   Bin_dti = col_double(),
##   ..   Default_flag = col_double(),
##   ..   No_of_Enquiry = col_double(),
##   ..   enq_buckets = col_character(),
##   ..   annual_inc = col_double(),
##   ..   Income_bins = col_double(),
##   ..   home_ownership = col_character(),
##   ..   purpose = col_character(),
##   ..   open_acc = col_double(),
##   ..   emp_length = col_character(),
##   ..   verification_status = col_character(),
##   ..   delinq_2yrs = col_double(),
##   ..   loan_amnt = col_double(),
##   ..   Bins_loan_amt = col_double()
##   .. )
##  - attr(*, ".internal.selfref")=<externalptr>
```

Data Cleaning

Converting certain categorical variables to factors. Cleaning interate rate from string data type to numerical data type. Trimming wide spaces and creating bins for necessary continuos variables:

```
Lend[, member_id := factor(member_id)]
Lend[, loan_status := factor(loan_status)]
Lend[, Default_flag := factor(Default_flag)]

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

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

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

```
## Classes 'data.table' and 'data.frame':   35808 obs. of  20 variables:
##  $ member_id          : Factor w/ 35808 levels "LC1","LC10","LC100",..: 1 2 3 4 5 6 7 8 9 10 ...
```

```
##  $ loan_status        : Factor w/ 2 levels "Charged Off",..: 1 2 2 2 2 2 2 2 2 2 ...
##  $ int_rate           : num  11.7 16 10.7 12.7 19.7 ...
##  $ Bin_int            : num  10 16 8 11 22 1 23 10 5 16 ...
##  $ dti                : num  1.06 2.61 11.34 14 13.01 ...
##  $ Bin_dti            : num  2 3 11 14 13 11 5 10 24 14 ...
##  $ Default_flag       : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 1 1 1 ...
##  $ No_of_Enquiry      : num  0 1 1 1 0 0 3 0 1 2 ...
##  $ enq_buckets        : chr  "0" "1-4" "1-4" "1-4" ...
##  $ annual_inc         : num  110000 135000 75000 51000 41500 ...
##  $ Income_bins        : num  9 11 6 4 3 4 12 7 6 4 ...
##  $ home_ownership     : Factor w/ 5 levels "MORTGAGE","NONE",..: 1 5 1 5 1 1 1 5 5 1 ...
##  $ purpose            : Factor w/ 14 levels "car","credit_card",..: 2 10 4 2 3 3 8 2 10 3 ...
##  $ open_acc           : num  6 3 7 5 8 5 4 7 6 9 ...
##  $ emp_length         : chr  "LT 1year" "10+ years" "2 years" "1 year" ...
##  $ verification_status: Factor w/ 3 levels "Not Verified",..: 1 2 2 2 3 3 1 1 1 2 ...
##  $ delinq_2yrs        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ loan_amnt          : num  7000 2000 12000 9350 6000 ...
##  $ Bins_loan_amt      : num  6 2 10 8 5 8 5 10 2 8 ...
##  $ no_of_acct         : chr  "6-10" "LT5" "6-10" "LT5" ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   member_id = col_character(),
##   ..   loan_status = col_character(),
##   ..   int_rate = col_character(),
##   ..   Bin_int = col_double(),
##   ..   dti = col_double(),
##   ..   Bin_dti = col_double(),
##   ..   Default_flag = col_double(),
##   ..   No_of_Enquiry = col_double(),
##   ..   enq_buckets = col_character(),
##   ..   annual_inc = col_double(),
##   ..   Income_bins = col_double(),
##   ..   home_ownership = col_character(),
##   ..   purpose = col_character(),
##   ..   open_acc = col_double(),
##   ..   emp_length = col_character(),
##   ..   verification_status = col_character(),
##   ..   delinq_2yrs = col_double(),
##   ..   loan_amnt = col_double(),
##   ..   Bins_loan_amt = col_double()
##   .. )
##  - attr(*, ".internal.selfref")=<externalptr>
##  - attr(*, "index")= int
##   ..- attr(*, "__open_acc")= int  75 113 157 195 377 382 458 611 628 642 ...
```

Data Splitting

Splitting Data into training and testing and default and non default borrowers for further analysis:

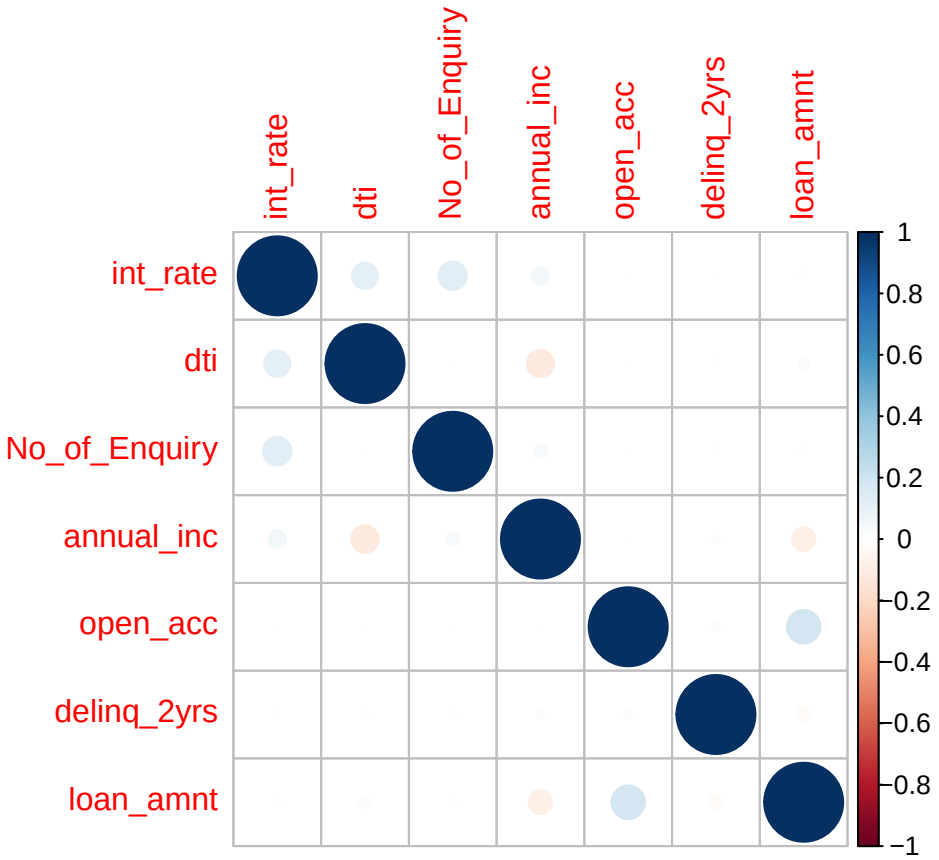
```
#Training Testing

## 75% of the sample size
smp_size = floor(0.75 * nrow(Lend))
```

```
## set the seed to make our partition reproducible
set.seed(123)
train_ind = sample(seq_len(nrow(Lend)), size = smp_size)

train = Lend[train_ind, ]
test = Lend[-train_ind, ]

#default & nondefault data

defaultdata =  filter(Lend, Default_flag == 1)
nondefault = filter(Lend, Default_flag == 0)
view(defaultdata)
view(nondefault)
defaultdata = setDT(defaultdata)
nondefault = setDT(nondefault)
```

Correlation Analysis

Let us check the correlation of each variable by pairs:

```
mat <- round(cor(Lend[, c(3,5,8,10,14,17,18)], use = "pair"), 2)
corrplot(mat)
```



```
mat
```

```
##                int_rate   dti No_of_Enquiry annual_inc open_acc delinq_2yrs
## int_rate           1.00  0.11          0.13       0.05     0.00        0.00
## dti                0.11  1.00          0.00      -0.12     0.00       -0.01
## No_of_Enquiry      0.13  0.00          1.00       0.03     0.00       -0.01
## annual_inc         0.05 -0.12          0.03       1.00     0.00        0.01
## open_acc           0.00  0.00          0.00       0.00     1.00        0.01
## delinq_2yrs        0.00 -0.01         -0.01       0.01     0.01        1.00
## loan_amnt         -0.01  0.02         -0.01      -0.09     0.18       -0.03
##                loan_amnt
## int_rate           -0.01
## dti                 0.02
## No_of_Enquiry      -0.01
## annual_inc         -0.09
## open_acc            0.18
## delinq_2yrs        -0.03
## loan_amnt           1.00
```

Principal Component Analysis:

Assigning default flags and numerical data for performing PCA into Lend_PCA:

```
Lend_PCA = Lend[, c(7,3,5,8,10,14,18)]
Lend_PCA
```

```
##         Default_flag int_rate   dti No_of_Enquiry annual_inc open_acc loan_amnt
##     1:             1    11.71  1.06             0     110000        6      7000
##     2:             0    15.96  2.61             1     135000        3      2000
##     3:             0    10.65 11.34             1      75000        7     12000
##     4:             0    12.69 14.00             1      51000        5      9350
##     5:             0    19.69 13.01             0      41500        8      6000
##    ---
## 35804:             1    13.49 19.13             0      48000        6     14000
## 35805:             0     9.99 11.40             1      50000        4     20000
## 35806:             0     9.99 21.12             2      45000        9      6400
## 35807:             0    15.23  7.64             1      30000        3      1500
## 35808:             0     8.49  7.10             0     107000       11      6000
```

Using prcomp to compute the principal components (eigenvalues and eigenvectors). With scale=TRUE, variable means are set to zero, and variances set to one

```
PCA_Data <- prcomp(Lend_PCA[, c(2,3,4,5,6,7)],scale=TRUE)
PCA_Data
```

```
## Standard deviations (1, .., p=6):
## [1] 1.1046047 1.0811152 1.0454206 0.9583681 0.9036363 0.8849328
##
## Rotation (n x k) = (6 x 6):
##                       PC1           PC2         PC3        PC4         PC5
## int_rate      -0.09562723 -0.7061927415  0.08031202  0.2899078  0.46291537
```

```
## dti            0.25994780 -0.4521514772 -0.53064118  0.3553719 -0.35348269
## No_of_Enquiry -0.13803378 -0.5438666419  0.30398089 -0.6820655 -0.29338659
## annual_inc    -0.44022257  0.0001169514  0.56076768  0.5061564 -0.05404644
## open_acc       0.53996076 -0.0292878372  0.48268242  0.2406027 -0.52229259
## loan_amnt      0.64720609 -0.0142175352  0.26855693 -0.1018187  0.54678432
##                      PC6
## int_rate      -0.4328510
## dti            0.4417770
## No_of_Enquiry  0.2036232
## annual_inc     0.4823157
## open_acc      -0.3793719
## loan_amnt      0.4466085
```

```
summary(PCA_Data)
```

```
## Importance of components:
##                            PC1    PC2    PC3    PC4    PC5    PC6
## Standard deviation     1.1046 1.0811 1.0454 0.9584 0.9036 0.8849
## Proportion of Variance 0.2034 0.1948 0.1822 0.1531 0.1361 0.1305
## Cumulative Proportion  0.2034 0.3982 0.5803 0.7334 0.8695 1.0000
```

sample scores stored in PCA_Data$x singular values (square roots of eigenvalues) stored in PCA_Data$sdev loadings (eigenvectors) are stored in PCA_Data$rotation variable means stored in PCA_Data$center variable standard deviations stored in PCA_Data$scale A table containing eigenvalues and %'s accounted, follows Eigenvalues are sdev^2

Storing eigen values in eigen_lend

```
(eigen_lend <- PCA_Data$sdev^2)
```

```
## [1] 1.2201515 1.1688101 1.0929043 0.9184695 0.8165586 0.7831060
```

```
names(eigen_lend) <- paste("PC",1:6,sep="")
eigen_lend
```

```
##       PC1       PC2       PC3       PC4       PC5       PC6
## 1.2201515 1.1688101 1.0929043 0.9184695 0.8165586 0.7831060
```

Sum of eigen values should give the number of Principal Components

```
sumlambdas_lend <- sum(eigen_lend)
sumlambdas_lend
```

```
## [1] 6
```

Proportion of Variance

```
propvar_lend <- eigen_lend/sumlambdas_lend
propvar_lend
```

```
##       PC1       PC2       PC3       PC4       PC5       PC6
## 0.2033586 0.1948017 0.1821507 0.1530782 0.1360931 0.1305177
```

Cummulative summation of proportion of variance must result in 1 at the end

```
cumvar_lend <- cumsum(propvar_lend)
cumvar_lend
```

```
##       PC1       PC2       PC3       PC4       PC5       PC6
## 0.2033586 0.3981603 0.5803110 0.7333892 0.8694823 1.0000000
```

Creating a matrix for the above eigen values, proportion of variance, and cummulative summation of proportion of variance

```
matlambdas_lend <- rbind(eigen_lend,propvar_lend,cumvar_lend)
matlambdas_lend
```

```
##                    PC1       PC2       PC3       PC4       PC5       PC6
## eigen_lend   1.2201515 1.1688101 1.0929043 0.9184695 0.8165586 0.7831060
## propvar_lend 0.2033586 0.1948017 0.1821507 0.1530782 0.1360931 0.1305177
## cumvar_lend  0.2033586 0.3981603 0.5803110 0.7333892 0.8694823 1.0000000
```

Setting name for the matrix and rounding decimals to 4th decimal place

```
rownames(matlambdas_lend) <- c("Eigenvalues","Prop. variance","Cum. prop. variance")
round(matlambdas_lend,4)
```

```
##                       PC1    PC2    PC3    PC4    PC5    PC6
## Eigenvalues         1.2202 1.1688 1.0929 0.9185 0.8166 0.7831
## Prop. variance      0.2034 0.1948 0.1822 0.1531 0.1361 0.1305
## Cum. prop. variance 0.2034 0.3982 0.5803 0.7334 0.8695 1.0000
```

```
summary(PCA_Data)
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4    PC5    PC6
## Standard deviation     1.1046 1.0811 1.0454 0.9584 0.9036 0.8849
## Proportion of Variance 0.2034 0.1948 0.1822 0.1531 0.1361 0.1305
## Cumulative Proportion  0.2034 0.3982 0.5803 0.7334 0.8695 1.0000
```

```
print(PCA_Data)
```

```
## Standard deviations (1, .., p=6):
## [1] 1.1046047 1.0811152 1.0454206 0.9583681 0.9036363 0.8849328
##
## Rotation (n x k) = (6 x 6):
##                        PC1           PC2         PC3        PC4         PC5
## int_rate      -0.09562723 -0.7061927415  0.08031202  0.2899078  0.46291537
## dti            0.25994780 -0.4521514772 -0.53064118  0.3553719 -0.35348269
## No_of_Enquiry -0.13803378 -0.5438666419  0.30398089 -0.6820655 -0.29338659
```

```
## annual_inc    -0.44022257  0.0001169514  0.56076768  0.5061564 -0.05404644
## open_acc       0.53996076 -0.0292878372  0.48268242  0.2406027 -0.52229259
## loan_amnt      0.64720609 -0.0142175352  0.26855693 -0.1018187  0.54678432
##                       PC6
## int_rate      -0.4328510
## dti            0.4417770
## No_of_Enquiry  0.2036232
## annual_inc     0.4823157
## open_acc      -0.3793719
## loan_amnt      0.4466085
```

```
PCA_Data$rotation
```

```
##                       PC1           PC2         PC3        PC4         PC5
## int_rate      -0.09562723 -0.7061927415  0.08031202  0.2899078  0.46291537
## dti            0.25994780 -0.4521514772 -0.53064118  0.3553719 -0.35348269
## No_of_Enquiry -0.13803378 -0.5438666419  0.30398089 -0.6820655 -0.29338659
## annual_inc    -0.44022257  0.0001169514  0.56076768  0.5061564 -0.05404644
## open_acc       0.53996076 -0.0292878372  0.48268242  0.2406027 -0.52229259
## loan_amnt      0.64720609 -0.0142175352  0.26855693 -0.1018187  0.54678432
##                       PC6
## int_rate      -0.4328510
## dti            0.4417770
## No_of_Enquiry  0.2036232
## annual_inc     0.4823157
## open_acc      -0.3793719
## loan_amnt      0.4466085
```

Sample scores

```
head(PCA_Data$x)
```

```
##             PC1         PC2        PC3         PC4         PC5        PC6
## [1,] -1.3801894  1.32926532  0.5540296  0.08139190  0.90292817 -0.6053680
## [2,] -2.5311671 -0.06973285  0.5106531 -0.03395348  1.04703349 -0.6802187
## [3,] -0.2859976  0.31886583 -0.0085235 -0.37409461  0.23696131  0.3482107
## [4,] -0.5519403 -0.23567944 -0.6933140 -0.32791067  0.41257557  0.1167969
## [5,] -0.5111143 -1.02714021 -0.6203363  0.94227214  1.02733671 -1.4940065
## [6,] -0.2964153  1.85140419 -0.8783566 -0.40582363 -0.04499685  0.6644681
```

Identifying the scores by their Default flag status

```
lendtyp_pca <- cbind(data.frame(Default_flag),PCA_Data$x)
head(lendtyp_pca)
```

```
##   Default_flag        PC1         PC2        PC3         PC4         PC5
## 1            1 -1.3801894  1.32926532  0.5540296  0.08139190  0.90292817
## 2            0 -2.5311671 -0.06973285  0.5106531 -0.03395348  1.04703349
## 3            0 -0.2859976  0.31886583 -0.0085235 -0.37409461  0.23696131
## 4            0 -0.5519403 -0.23567944 -0.6933140 -0.32791067  0.41257557
## 5            0 -0.5111143 -1.02714021 -0.6203363  0.94227214  1.02733671
```

```
## 6            0 -0.2964153  1.85140419 -0.8783566 -0.40582363 -0.04499685
##          PC6
## 1 -0.6053680
## 2 -0.6802187
## 3  0.3482107
## 4  0.1167969
## 5 -1.4940065
## 6  0.6644681
```

Means of scores for all the Principal Components classified by Default flag status

```
tabmeansPC_lend <- aggregate(lendtyp_pca[,2:7],
                             by=list(Default_flag=Lend_PCA$Default_flag),mean)
tabmeansPC_lend
```

```
##   Default_flag          PC1         PC2          PC3           PC4         PC5
## 1            0 -0.001821377  0.07722583  0.001368908  0.0003074859 -0.02443986
## 2            1  0.011256596 -0.47727640 -0.008460221 -0.0019003454  0.15104488
##          PC6
## 1  0.0270281
## 2 -0.1670409
```

```
tabmeansPC_lend <- tabmeansPC_lend[rev(order(tabmeansPC_lend$Default_flag)),]
tabmeansPC_lend
```

```
##   Default_flag          PC1         PC2          PC3           PC4         PC5
## 2            1  0.011256596 -0.47727640 -0.008460221 -0.0019003454  0.15104488
## 1            0 -0.001821377  0.07722583  0.001368908  0.0003074859 -0.02443986
##          PC6
## 2 -0.1670409
## 1  0.0270281
```

```
tabfmeans_lend <- t(tabmeansPC_lend[,-1])
tabfmeans_lend
```

```
##                2            1
## PC1  0.011256596 -0.0018213765
## PC2 -0.477276402  0.0772258335
## PC3 -0.008460221  0.0013689084
## PC4 -0.001900345  0.0003074859
## PC5  0.151044876 -0.0244398559
## PC6 -0.167040898  0.0270280964
```

```
colnames(tabfmeans_lend) <- t(as.vector(tabmeansPC_lend[1]))
tabfmeans_lend
```

```
##                1            0
## PC1  0.011256596 -0.0018213765
## PC2 -0.477276402  0.0772258335
## PC3 -0.008460221  0.0013689084
## PC4 -0.001900345  0.0003074859
## PC5  0.151044876 -0.0244398559
## PC6 -0.167040898  0.0270280964
```

Standard deviations of scores for all the Principal Components classified by Default flag status

```
tabsdsPC_lend <- aggregate(lendtyp_pca[,2:7],
                           by=list(Default_flag=Lend_PCA$Default_flag),sd)
tabfsds_lend <- t(tabsdsPC_lend[,-1])
colnames(tabfsds_lend) <- t(as.vector(tabsdsPC_lend[1]))
tabfsds_lend
```

```
##             0         1
## PC1 1.1116382 1.0601393
## PC2 1.0686564 1.0343588
## PC3 1.0543919 0.9882295
## PC4 0.9568392 0.9678582
## PC5 0.8981711 0.9225403
## PC6 0.8871766 0.8522089
```

T test

```
t.test(PC1~Lend_PCA$Default_flag,data=lendtyp_pca)
```

```
##
##  Welch Two Sample t-test
##
## data:  PC1 by Lend_PCA$Default_flag
## t = -0.80268, df = 6882.7, p-value = 0.4222
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.04501712  0.01886117
## sample estimates:
## mean in group 0 mean in group 1
##    -0.001821377     0.011256596
```

```
t.test(PC2~Lend_PCA$Default_flag,data=lendtyp_pca)
```

```
##
##  Welch Two Sample t-test
##
## data:  PC2 by Lend_PCA$Default_flag
## t = 34.959, df = 6824.1, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.5234085 0.5855960
## sample estimates:
## mean in group 0 mean in group 1
##      0.07722583     -0.47727640
```

```
t.test(PC3~Lend_PCA$Default_flag,data=lendtyp_pca)
```

```
##
##  Welch Two Sample t-test
##
```

```
## data:  PC3 by Lend_PCA$Default_flag
## t = 0.64545, df = 6953.8, p-value = 0.5187
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.02002289  0.03968115
## sample estimates:
## mean in group 0 mean in group 1
##     0.001368908    -0.008460221
```

```
t.test(PC4~Lend_PCA$Default_flag,data=lendtyp_pca)
```

```
##
##  Welch Two Sample t-test
##
## data:  PC4 by Lend_PCA$Default_flag
## t = 0.14969, df = 6660.7, p-value = 0.881
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.02670562  0.03112129
## sample estimates:
## mean in group 0 mean in group 1
##    0.0003074859   -0.0019003454
```

```
t.test(PC5~Lend_PCA$Default_flag,data=lendtyp_pca)
```

```
##
##  Welch Two Sample t-test
##
## data:  PC5 by Lend_PCA$Default_flag
## t = -12.508, df = 6607.5, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.2029876 -0.1479819
## sample estimates:
## mean in group 0 mean in group 1
##     -0.02443986      0.15104488
```

```
t.test(PC6~Lend_PCA$Default_flag,data=lendtyp_pca)
```

```
##
##  Welch Two Sample t-test
##
## data:  PC6 by Lend_PCA$Default_flag
## t = 14.834, df = 6853.9, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.168422 0.219716
## sample estimates:
## mean in group 0 mean in group 1
##       0.0270281      -0.1670409
```

F ratio test

```r
var.test(PC1~Lend_PCA$Default_flag,data=lendtyp_pca)
```

```
##
##  F test to compare two variances
##
## data:  PC1 by Lend_PCA$Default_flag
## F = 1.0995, num df = 30820, denom df = 4986, p-value = 1.429e-05
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  1.053615 1.146663
## sample estimates:
## ratio of variances
##           1.099515
```

```r
var.test(PC2~Lend_PCA$Default_flag,data=lendtyp_pca)
```

```
##
##  F test to compare two variances
##
## data:  PC2 by Lend_PCA$Default_flag
## F = 1.0674, num df = 30820, denom df = 4986, p-value = 0.00276
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  1.022856 1.113188
## sample estimates:
## ratio of variances
##           1.067416
```

```r
var.test(PC3~Lend_PCA$Default_flag,data=lendtyp_pca)
```

```
##
##  F test to compare two variances
##
## data:  PC3 by Lend_PCA$Default_flag
## F = 1.1384, num df = 30820, denom df = 4986, p-value = 3.515e-09
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  1.090861 1.187198
## sample estimates:
## ratio of variances
##           1.138383
```

```r
var.test(PC4~Lend_PCA$Default_flag,data=lendtyp_pca)
```

```
##
##  F test to compare two variances
##
## data:  PC4 by Lend_PCA$Default_flag
## F = 0.97736, num df = 30820, denom df = 4986, p-value = 0.2851
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
```

```
##  0.9365594 1.0192698
## sample estimates:
## ratio of variances
##          0.9773597
```

```
var.test(PC5~Lend_PCA$Default_flag,data=lendtyp_pca)
```

```
##
##  F test to compare two variances
##
## data:  PC5 by Lend_PCA$Default_flag
## F = 0.94787, num df = 30820, denom df = 4986, p-value = 0.01237
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.9082980 0.9885125
## sample estimates:
## ratio of variances
##          0.9478671
```

```
var.test(PC6~Lend_PCA$Default_flag,data=lendtyp_pca)
```

```
##
##  F test to compare two variances
##
## data:  PC6 by Lend_PCA$Default_flag
## F = 1.0837, num df = 30820, denom df = 4986, p-value = 0.0002293
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  1.038506 1.130220
## sample estimates:
## ratio of variances
##           1.083747
```

Levene's tests (one-sided)

```
(LTPC1 <- leveneTest(PC1~Lend_PCA$Default_flag,data=lendtyp_pca))
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value Pr(>F)
## group     1  1.2534 0.2629
##       35806
```

```
(LTPC1 <- leveneTest(PC1~Lend_PCA$Default_flag,data=lendtyp_pca))
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value Pr(>F)
## group     1  1.2534 0.2629
##       35806
```

```
(p_PC1_1sided <- LTPC1[[3]][1]/2)
```

```
## [1] 0.1314576
```

```
(LTPC2 <- leveneTest(PC2~Lend_PCA$Default_flag,data=lendtyp_pca))
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value    Pr(>F)
## group     1   12.85 0.0003379 ***
##       35806
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
(p_PC2_1sided=LTPC2[[3]][1]/2)
```

```
## [1] 0.0001689694
```

```
(LTPC3 <- leveneTest(PC3~Lend_PCA$Default_flag,data=lendtyp_pca))
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value Pr(>F)
## group     1  0.3572 0.5501
##       35806
```

```
(p_PC3_1sided <- LTPC3[[3]][1]/2)
```

```
## [1] 0.2750405
```

```
(LTPC4 <- leveneTest(PC4~Lend_PCA$Default_flag,data=lendtyp_pca))
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value    Pr(>F)
## group     1  22.322 2.314e-06 ***
##       35806
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
(p_PC4_1sided <- LTPC4[[3]][1]/2)
```

```
## [1] 1.157097e-06
```

```
(LTPC5 <- leveneTest(PC5~Lend_PCA$Default_flag,data=lendtyp_pca))
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value  Pr(>F)
## group     1  5.3409 0.02084 *
##       35806
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
(p_PC5_1sided <- LTPC5[[3]][1]/2)
```

```
## [1] 0.01041834
```

```
(LTPC6 <- leveneTest(PC6~Lend_PCA$Default_flag,data=lendtyp_pca))
```

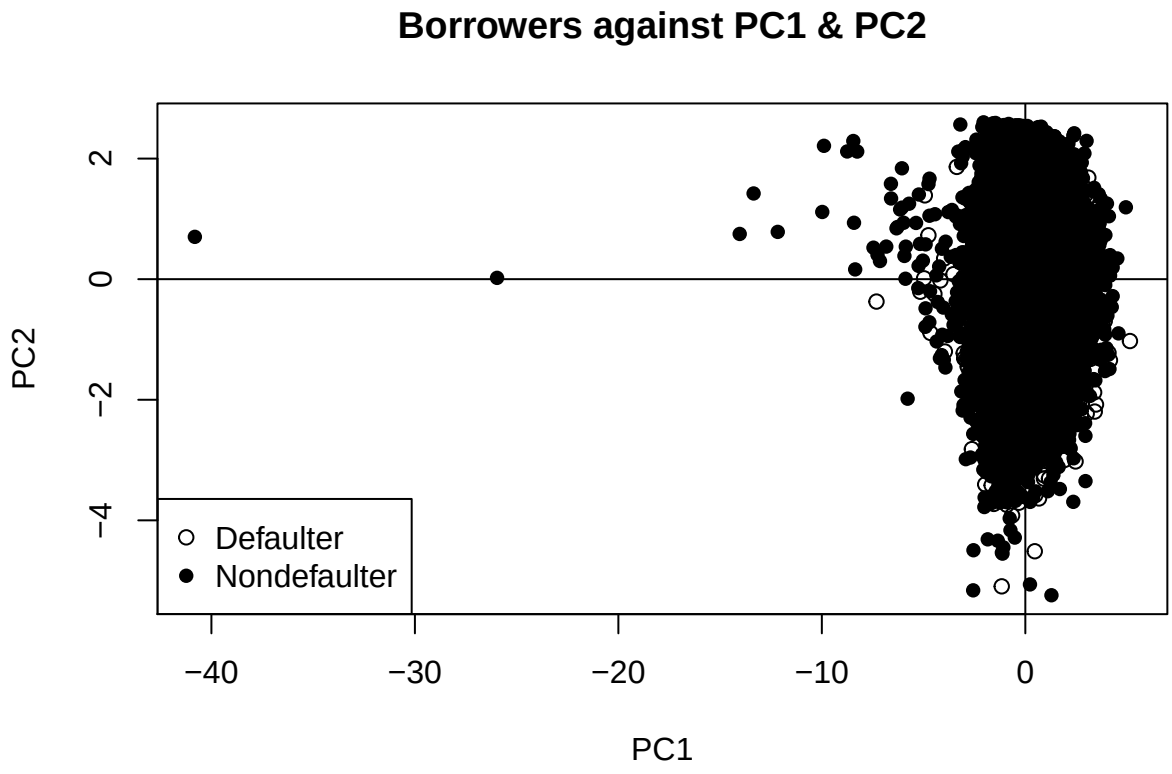
```
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value Pr(>F)
## group     1  2.8479 0.0915 .
##       35806
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
(p_PC6_1sided <- LTPC6[[3]][1]/2)
```

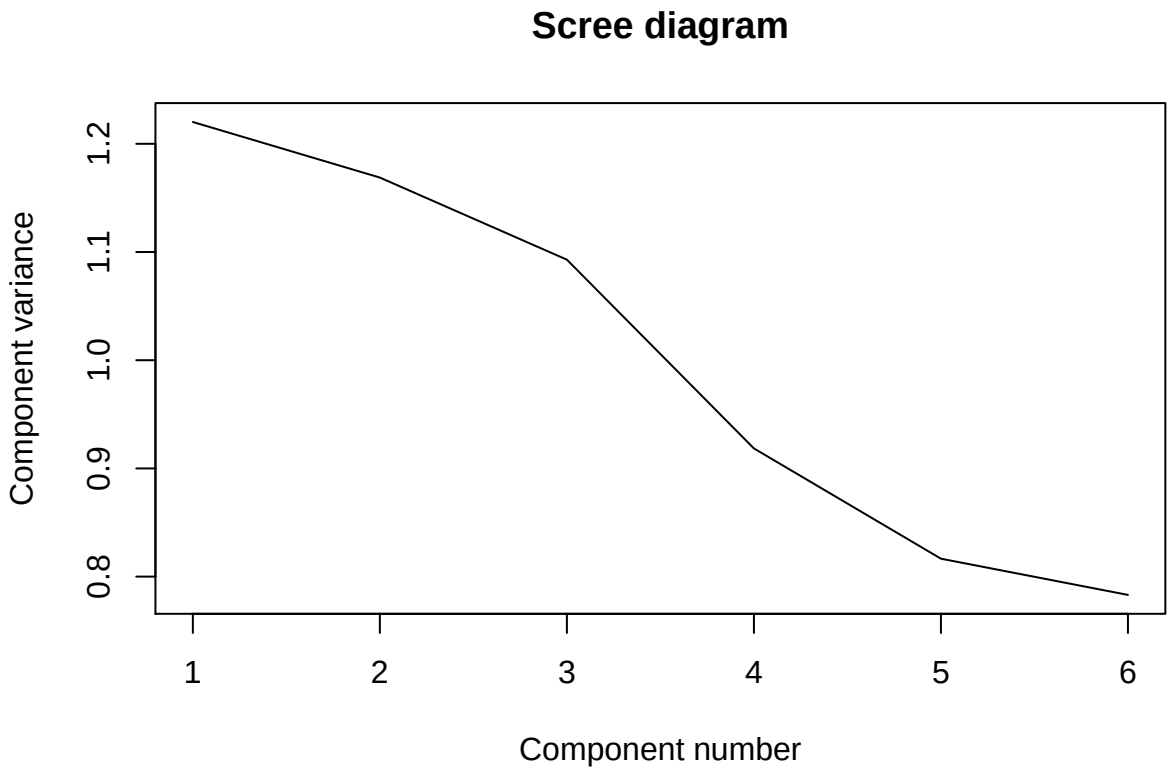
```
## [1] 0.04575006
```

Plotting the scores for the first and second components

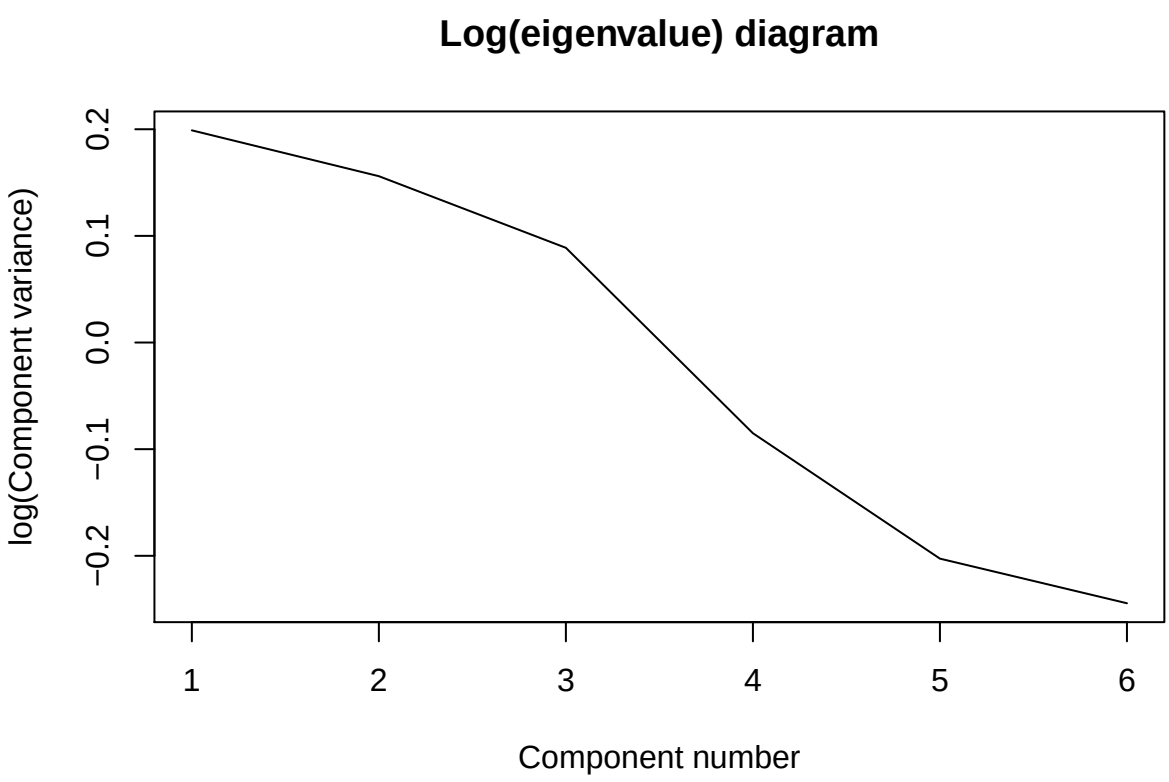
```
plot(lendtyp_pca$PC1, lendtyp_pca$PC2,pch=ifelse(lendtyp_pca$Default_flag == 1,1,16),
     xlab="PC1", ylab="PC2", main="Borrowers against PC1 & PC2")
  abline(h=0)
  abline(v=0)
  legend("bottomleft", legend=c("Defaulter","Nondefaulter"), pch=c(1,16))
```



```
plot(eigen_lend, xlab = "Component number", ylab = "Component variance",
     type = "l", main = "Scree diagram")
```




```
plot(log(eigen_lend), xlab = "Component number",
     ylab = "log(Component variance)", type="l",main = "Log(eigenvalue) diagram")
```




```
print(summary(PCA_Data))
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4    PC5    PC6
## Standard deviation     1.1046 1.0811 1.0454 0.9584 0.9036 0.8849
## Proportion of Variance 0.2034 0.1948 0.1822 0.1531 0.1361 0.1305
## Cumulative Proportion  0.2034 0.3982 0.5803 0.7334 0.8695 1.0000
```

```
#View(PCA_Data)
diag(cov(PCA_Data$x))
```

```
##       PC1       PC2       PC3       PC4       PC5       PC6
## 1.2201515 1.1688101 1.0929043 0.9184695 0.8165586 0.7831060
```

```
xlim <- range(PCA_Data$x[,1])
head(PCA_Data$x[,1])
```

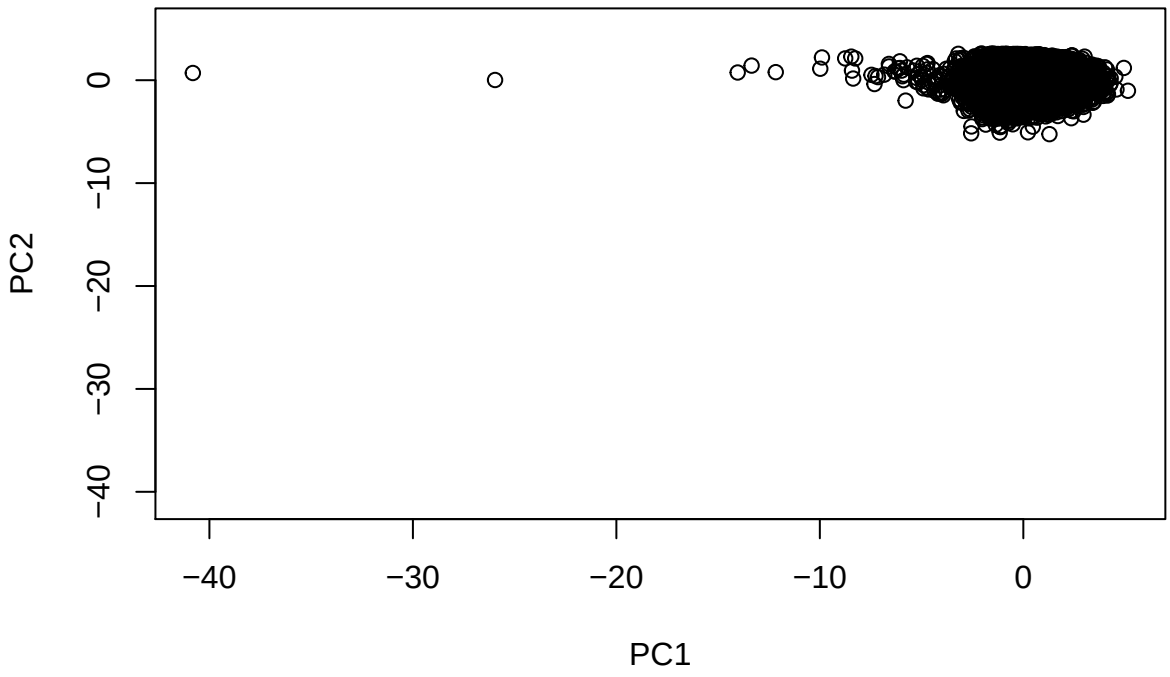
```
## [1] -1.3801894 -2.5311671 -0.2859976 -0.5519403 -0.5111143 -0.2964153
```

```
head(PCA_Data$x)
```

```
##             PC1         PC2        PC3         PC4         PC5        PC6
## [1,] -1.3801894  1.32926532  0.5540296  0.08139190  0.90292817 -0.6053680
```

```
## [2,] -2.5311671 -0.06973285  0.5106531 -0.03395348  1.04703349 -0.6802187
## [3,] -0.2859976  0.31886583 -0.0085235 -0.37409461  0.23696131  0.3482107
## [4,] -0.5519403 -0.23567944 -0.6933140 -0.32791067  0.41257557  0.1167969
## [5,] -0.5111143 -1.02714021 -0.6203363  0.94227214  1.02733671 -1.4940065
## [6,] -0.2964153  1.85140419 -0.8783566 -0.40582363 -0.04499685  0.6644681
```

```
plot(PCA_Data$x,xlim=xlim,ylim=xlim)
```




```
PCA_Data$rotation[,1]
```

```
##      int_rate           dti No_of_Enquiry    annual_inc      open_acc
##   -0.09562723    0.25994780   -0.13803378   -0.44022257    0.53996076
##     loan_amnt
##    0.64720609
```

```
PCA_Data$rotation
```

```
##                       PC1           PC2         PC3         PC4         PC5
## int_rate      -0.09562723 -0.7061927415  0.08031202  0.2899078  0.46291537
## dti            0.25994780 -0.4521514772 -0.53064118  0.3553719 -0.35348269
## No_of_Enquiry -0.13803378 -0.5438666419  0.30398089 -0.6820655 -0.29338659
## annual_inc    -0.44022257  0.0001169514  0.56076768  0.5061564 -0.05404644
## open_acc       0.53996076 -0.0292878372  0.48268242  0.2406027 -0.52229259
## loan_amnt      0.64720609 -0.0142175352  0.26855693 -0.1018187  0.54678432
```

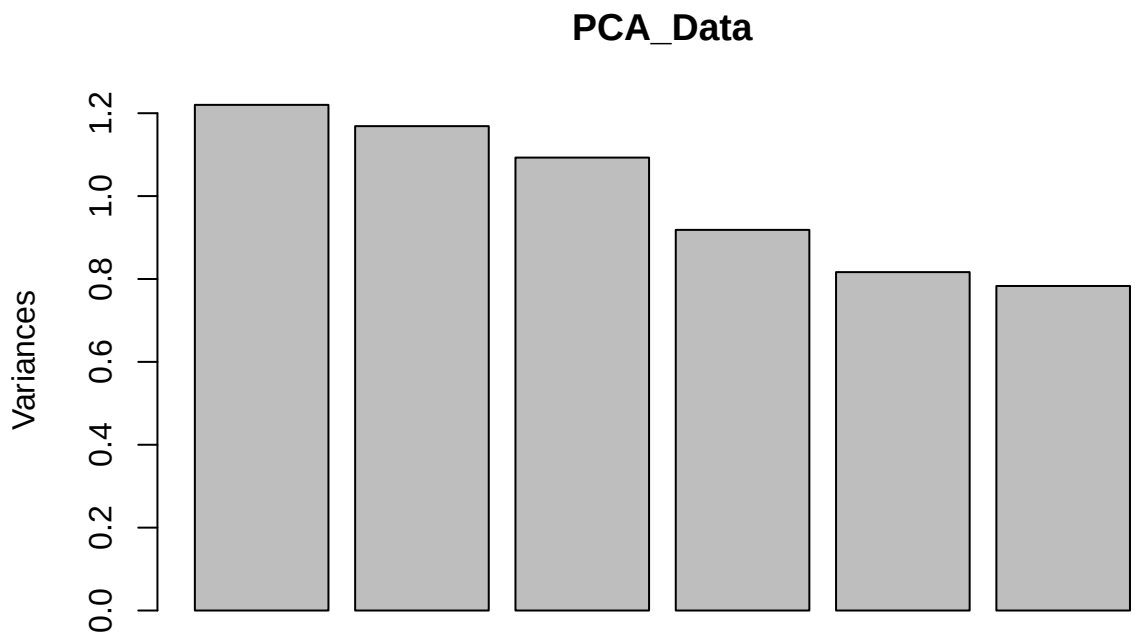
```
##                      PC6
## int_rate      -0.4328510
## dti            0.4417770
## No_of_Enquiry  0.2036232
## annual_inc     0.4823157
## open_acc      -0.3793719
## loan_amnt      0.4466085
```

plot(Lend_PCA[,-1])

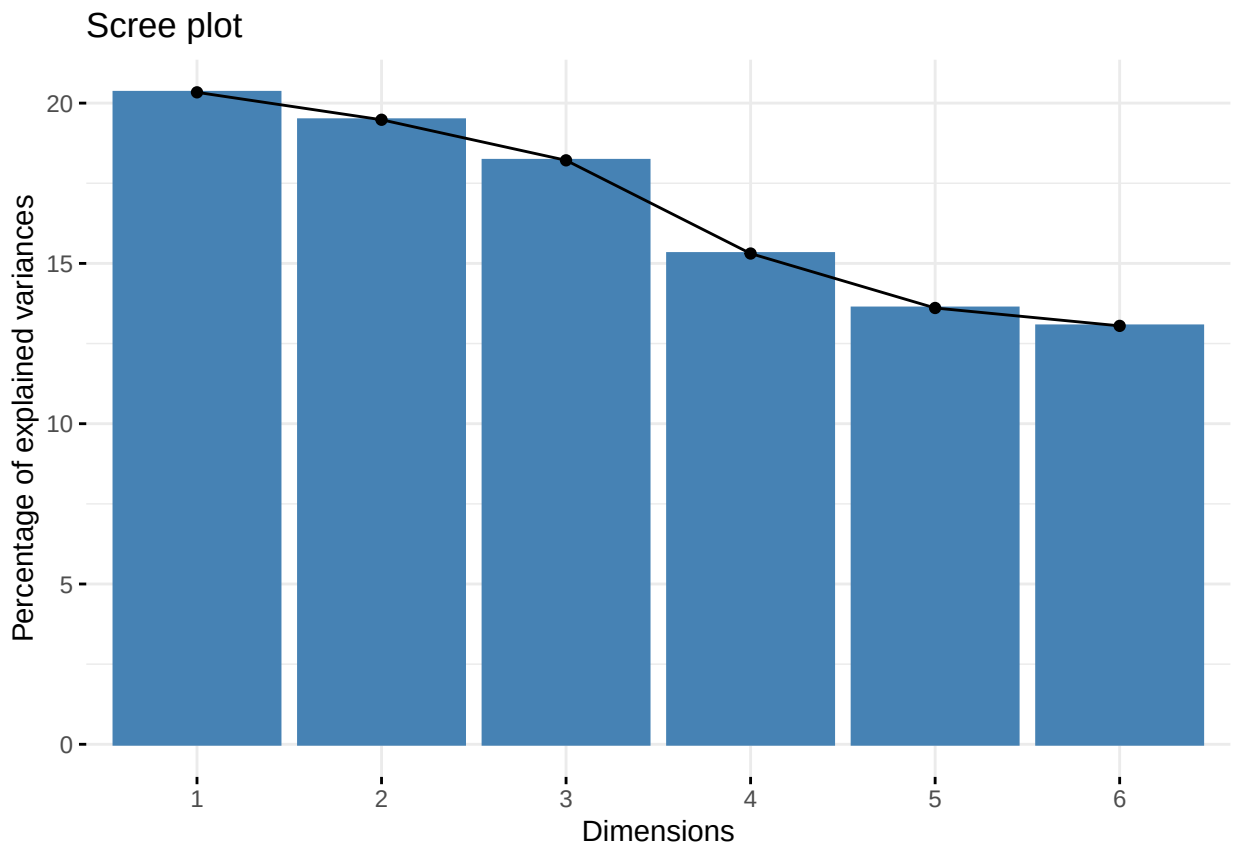
```
head(PCA_Data$x)
```

```
##             PC1         PC2        PC3         PC4         PC5        PC6
## [1,] -1.3801894  1.32926532  0.5540296  0.08139190  0.90292817 -0.6053680
## [2,] -2.5311671 -0.06973285  0.5106531 -0.03395348  1.04703349 -0.6802187
## [3,] -0.2859976  0.31886583 -0.0085235 -0.37409461  0.23696131  0.3482107
## [4,] -0.5519403 -0.23567944 -0.6933140 -0.32791067  0.41257557  0.1167969
## [5,] -0.5111143 -1.02714021 -0.6203363  0.94227214  1.02733671 -1.4940065
## [6,] -0.2964153  1.85140419 -0.8783566 -0.40582363 -0.04499685  0.6644681
```

```
plot(PCA_Data)
```




```
fviz_eig(PCA_Data)
```



get the original value of the data based on PCA

```
center <- PCA_Data$center
scale <- PCA_Data$scale
new_lend <- as.matrix(Lend_PCA[,-1])
head(new_lend)
```

```
##      int_rate   dti No_of_Enquiry annual_inc open_acc loan_amnt
## [1,]    11.71  1.06             0  110000.00        6      7000
## [2,]    15.96  2.61             1  135000.00        3      2000
## [3,]    10.65 11.34             1   75000.00        7     12000
## [4,]    12.69 14.00             1   51000.00        5      9350
## [5,]    19.69 13.01             0   41500.00        8      6000
## [6,]     5.42 11.30             0   53200.08        5     10000
```

```
head(drop(scale(new_lend,center=center, scale=scale)%*%PCA_Data$rotation[,1]))
```

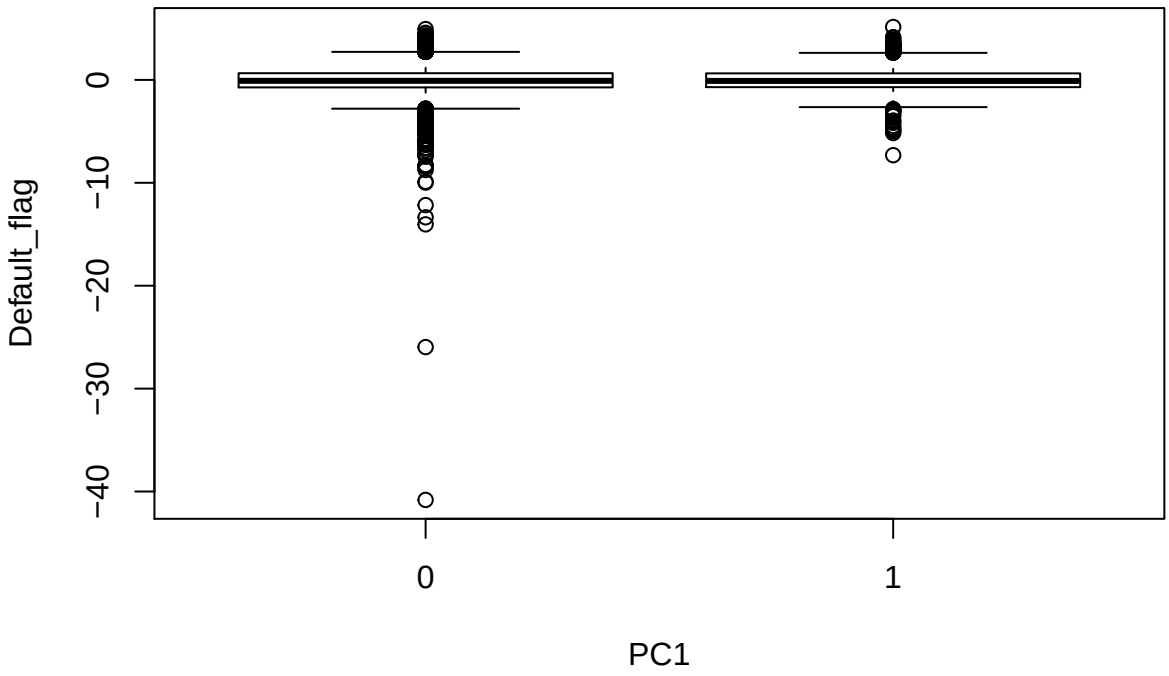
```
## [1] -1.3801894 -2.5311671 -0.2859976 -0.5519403 -0.5111143 -0.2964153
```

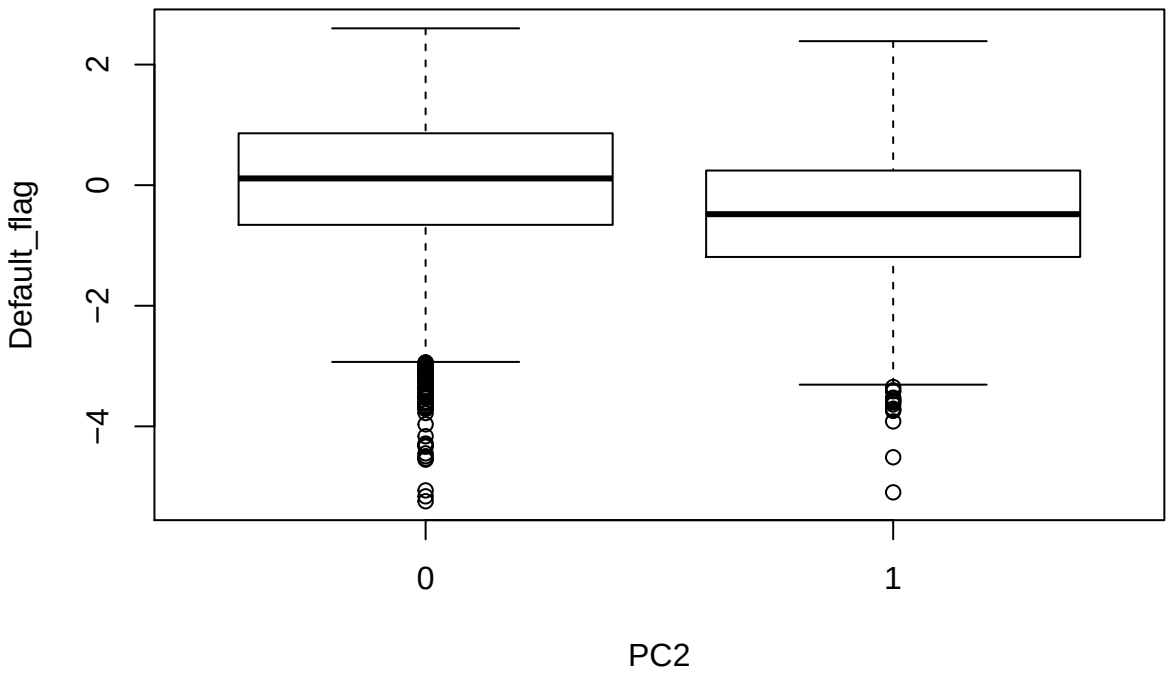
```
head(predict(PCA_Data)[,1])
```

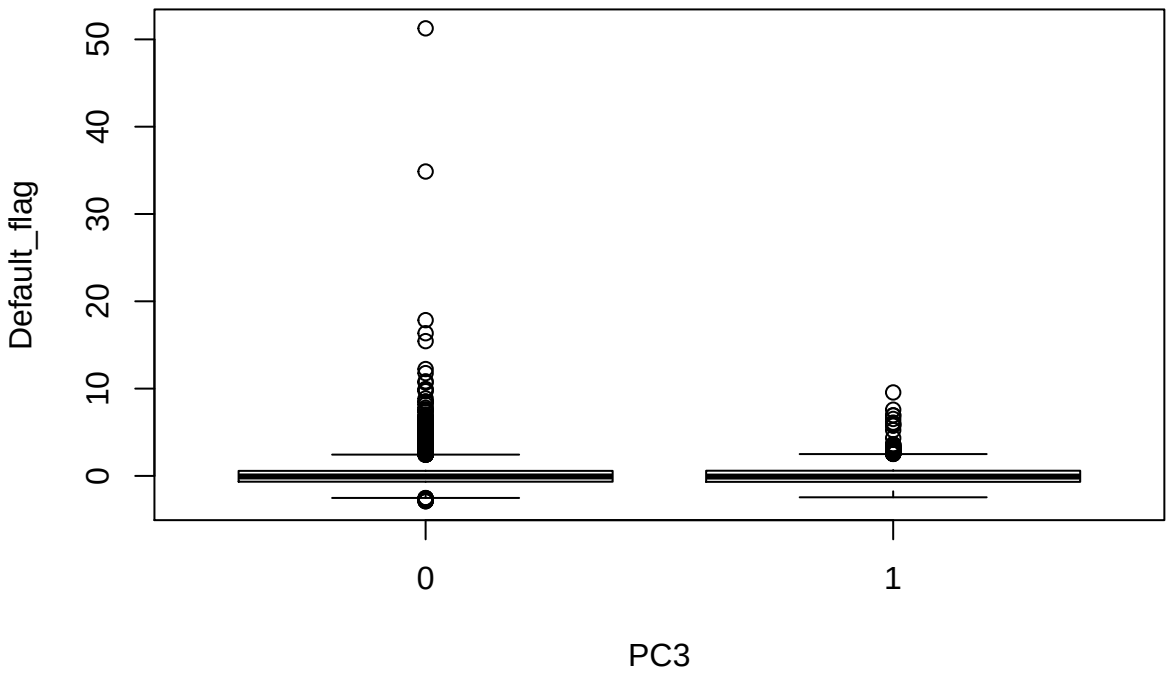
```
## [1] -1.3801894 -2.5311671 -0.2859976 -0.5519403 -0.5111143 -0.2964153
```

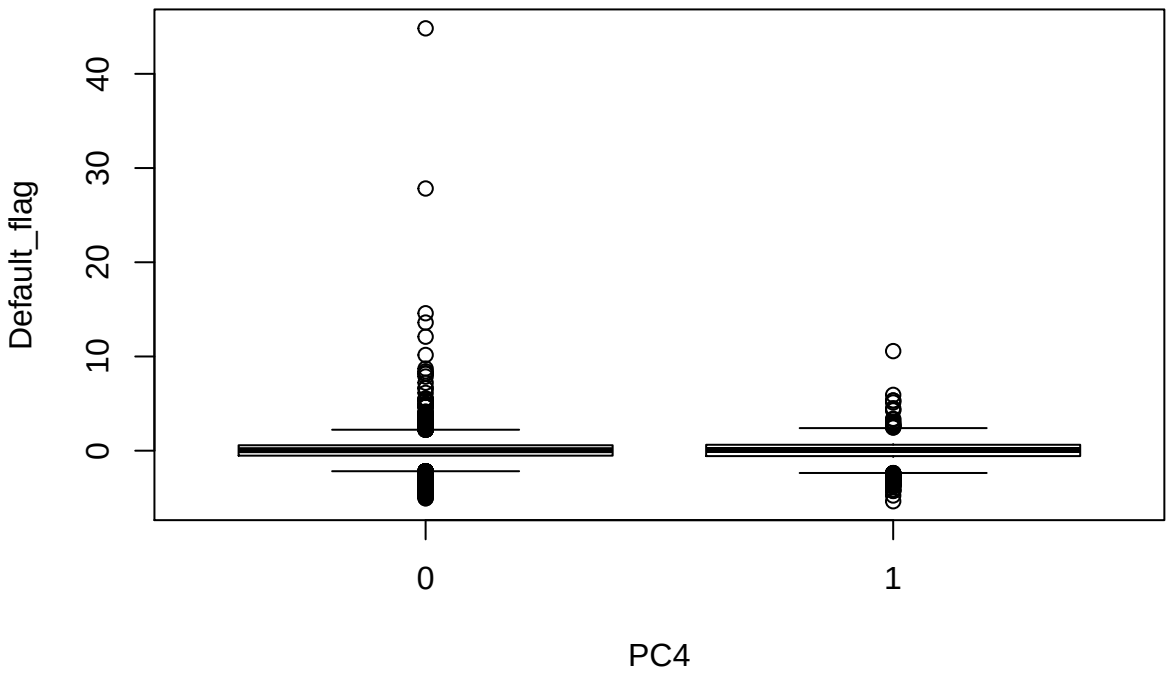
The aboved two gives us the same thing. predict is a good function to know.

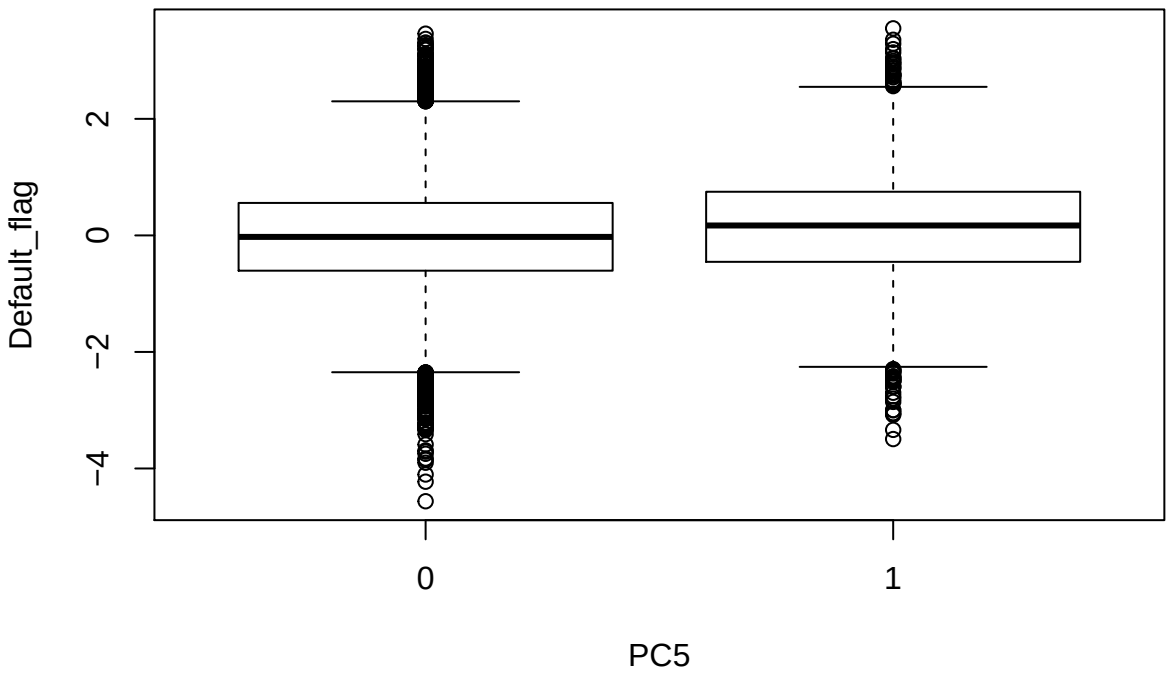
```
out <- sapply(1:6, function(i){plot(Lend_PCA$Default_flag,PCA_Data$x[,i],xlab=paste
                                    ("PC",i,sep=""),ylab="Default_flag")})
```

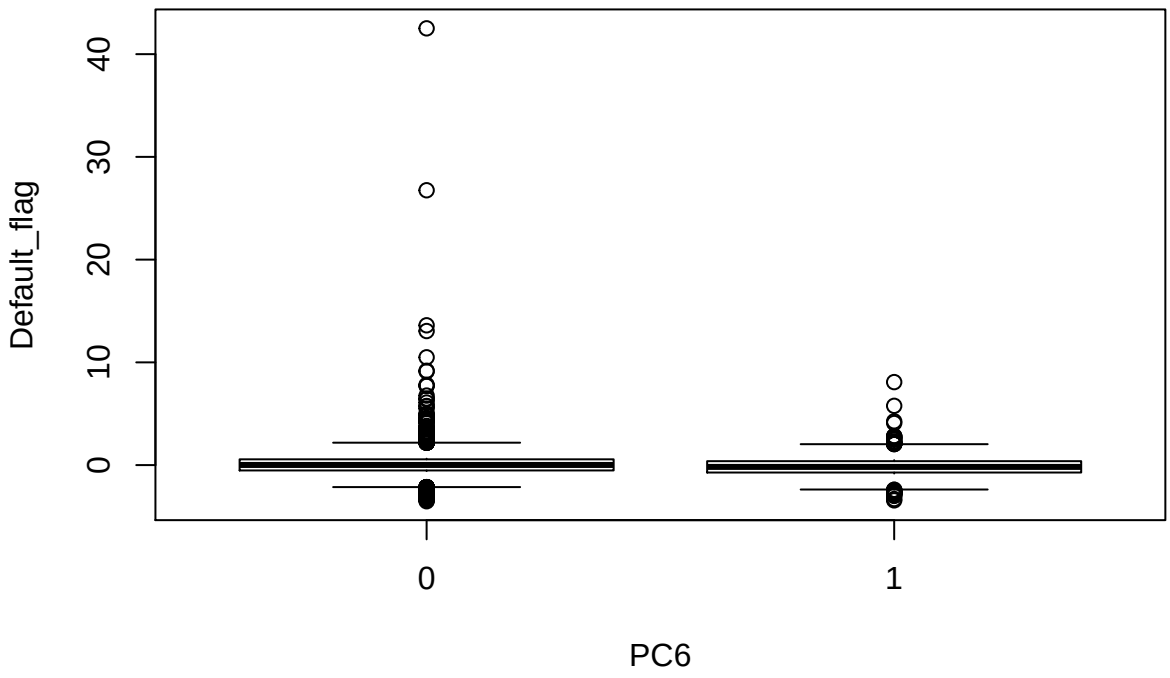



Default_flag
2
0
−2
−4
0
1
PC2

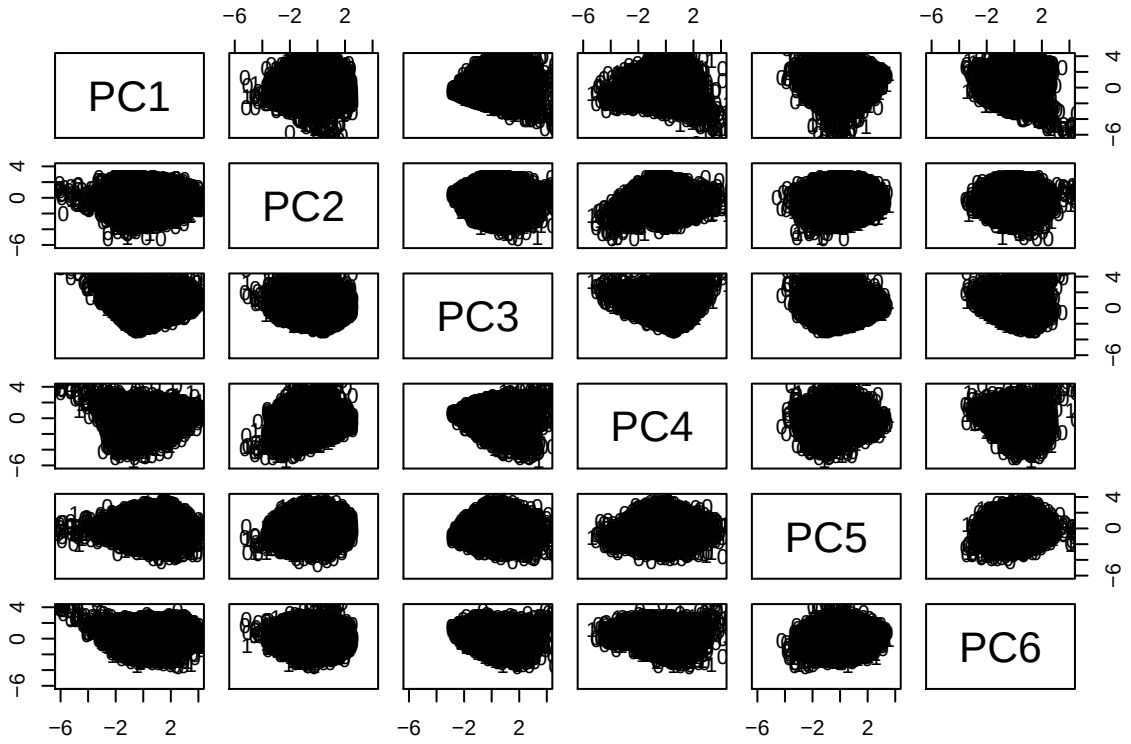

Default_flag
0
10
20
30
40
50
0
1
PC3


Default_flag
0
10
20
30
40
0
1
PC4


Default_flag
2
0
−2
−4
0
1
PC5

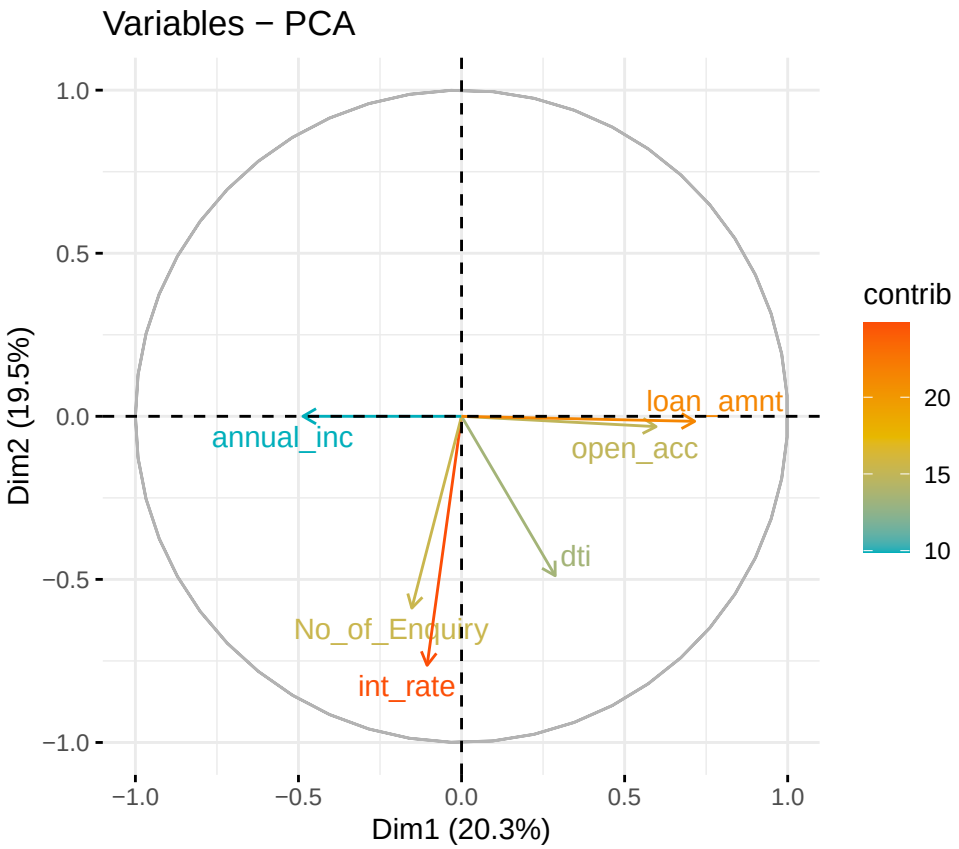



```
pairs(PCA_Data$x[,1:6], ylim = c(-6,4),
      xlim = c(-6,4),panel=function(x,y,...){text(x,y,Lend_PCA$Default_flag)})
```



Color by contributions to the Principal Components and avoid text overlapping

```
fviz_pca_var(PCA_Data,col.var = "contrib",
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),repel = TRUE)
```


Variables − PCA
1.0
0.5
0.0
−0.5
−1.0
Dim2 (19.5%)
−1.0
−0.5
0.0
0.5
1.0
Dim1 (20.3%)
annual_inc
loan_amnt
open_acc
dti
No_of_Enquiry
int_rate
contrib
20
15
10