Principal Component Analysis

Principal Component Analysis (PCA) is used to explain linear combinations through co-variance between them. They can drastically help basic analysis of the data helping us understand dimensionality of a data and in dimensional reduction. It also helps reduce noise from the data.

```
custc <-
read.csv("C:/Users/admin/Desktop/MVA/PROJECT/TelEco_Customer_Churn.csv")
dim(custc)

## [1] 7043   21

#structure of dataset

str(custc)

## 'data.frame':    7043 obs. of  21 variables:
##  $ customerID      : Factor w/ 7043 levels "0002-ORFBO","0003-MKNFE",..:
5376 3963 2565 5536 6512 6552 1003 4771 5605 4535 ...
##  $ gender          : Factor w/ 2 levels "Female","Male": 1 2 2 2 1 1 2 1 1
2 ...
##  $ SeniorCitizen   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Partner         : Factor w/ 2 levels "No","Yes": 2 1 1 1 1 1 1 1 2 1
...
##  $ Dependents      : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 2 1 1 2
...
##  $ tenure          : int  1 34 2 45 2 8 22 10 28 62 ...
##  $ PhoneService    : Factor w/ 2 levels "No","Yes": 1 2 2 1 2 2 2 1 2 2
...
##  $ MultipleLines   : Factor w/ 3 levels "No","No phone service",..: 2 1 1
2 1 3 3 2 3 1 ...
##  $ InternetService : Factor w/ 3 levels "DSL","Fiber optic",..: 1 1 1 1 2
2 2 1 2 1 ...
##  $ OnlineSecurity  : Factor w/ 3 levels "No","No internet service",..: 1 3
3 3 1 1 1 3 1 3 ...
##  $ OnlineBackup    : Factor w/ 3 levels "No","No internet service",..: 3 1
3 1 1 1 3 1 1 3 ...
##  $ DeviceProtection: Factor w/ 3 levels "No","No internet service",..: 1 3
1 3 1 3 1 1 3 1 ...
##  $ TechSupport     : Factor w/ 3 levels "No","No internet service",..: 1 1
1 3 1 1 1 1 3 1 ...
##  $ StreamingTV     : Factor w/ 3 levels "No","No internet service",..: 1 1
1 1 1 3 3 1 3 1 ...
##  $ StreamingMovies : Factor w/ 3 levels "No","No internet service",..: 1 1
1 1 1 3 1 1 3 1 ...
##  $ Contract        : Factor w/ 3 levels "Month-to-month",..: 1 2 1 2 1 1 1
1 1 2 ...
##  $ PaperlessBilling: Factor w/ 2 levels "No","Yes": 2 1 2 1 2 2 2 1 2 1
```

```
...
##  $ PaymentMethod   : Factor w/ 4 levels "Bank transfer (automatic)",..: 3
4 4 1 3 3 2 4 3 1 ...
##  $ MonthlyCharges  : num  29.9 57 53.9 42.3 70.7 ...
##  $ TotalCharges    : num  29.9 1889.5 108.2 1840.8 151.7 ...
##  $ Churn           : Factor w/ 2 levels "No","Yes": 1 1 2 1 2 2 1 1 2 1
...
```

```
sapply(custc, function(x) sum(is.na(x)))
```

```
##       customerID           gender    SeniorCitizen          Partner
##                0                0                0                0
##       Dependents           tenure     PhoneService    MultipleLines
##                0                0                0                0
##  InternetService   OnlineSecurity     OnlineBackup DeviceProtection
##                0                0                0                0
##      TechSupport      StreamingTV  StreamingMovies         Contract
##                0                0                0                0
## PaperlessBilling    PaymentMethod   MonthlyCharges     TotalCharges
##                0                0                0               11
##            Churn
##                0
```

```
custc <- custc[complete.cases(custc),]  ## to remove which has null values
sapply(custc, function(x) sum(is.na(x)))
```

```
##       customerID           gender    SeniorCitizen          Partner
##                0                0                0                0
##       Dependents           tenure     PhoneService    MultipleLines
##                0                0                0                0
##  InternetService   OnlineSecurity     OnlineBackup DeviceProtection
##                0                0                0                0
##      TechSupport      StreamingTV  StreamingMovies         Contract
##                0                0                0                0
## PaperlessBilling    PaymentMethod   MonthlyCharges     TotalCharges
##                0                0                0                0
##            Churn
##                0
```

```
dim(custc)
```

```
## [1] 7032   21
```

```
quant_var_df<-
data.frame(custc$tenure,custc$MonthlyCharges,custc$TotalCharges)
quant_var_df
```

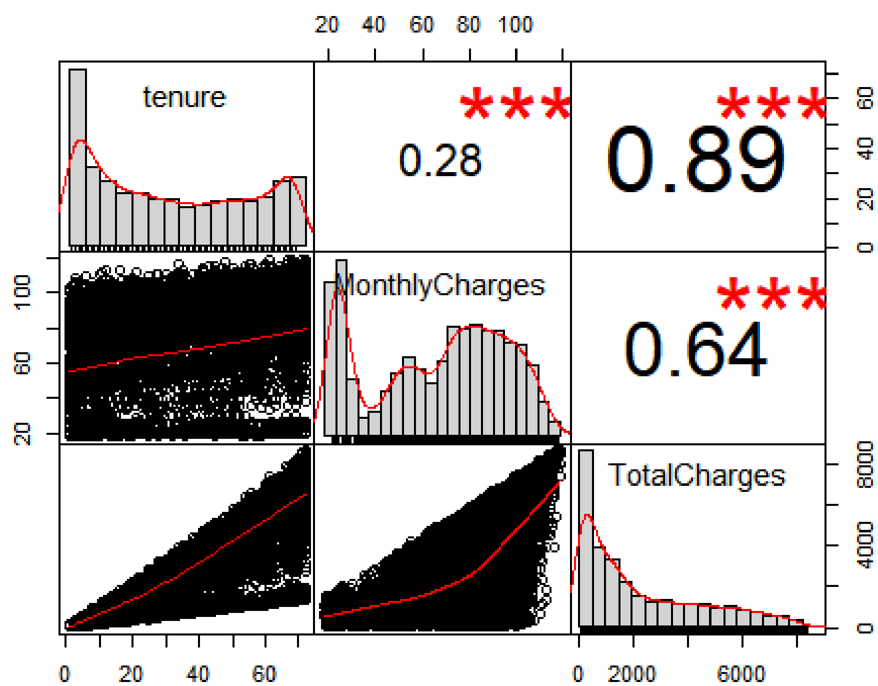
```
##      custc.tenure custc.MonthlyCharges custc.TotalCharges
## 1               1                29.85              29.85
## 2              34                56.95            1889.50
## 3               2                53.85             108.15
## 4              45                42.30            1840.75
```

```
## 5                2                 70.70                151.65
## 6                8                 99.65                820.50
## 7               22                 89.10               1949.40
## 8               10                 29.75                301.90
## 9               28                104.80               3046.05
## 10              62                 56.15               3487.95
## 11              13                 49.95                587.45
## 12              16                 18.95                326.80
## 13              58                100.35               5681.10
## 14              49                103.70               5036.30
## 15              25                105.50               2686.05

## 7020             2                 20.05                 39.25
## 7021            55                 60.00               3316.10
## 7022             1                 75.75                 75.75
## 7023            38                 69.50               2625.25
## 7024            67                102.95               6886.25
## 7025            19                 78.70               1495.10
## 7026            12                 60.65                743.30
## 7027            72                 21.15               1419.40
## 7028            24                 84.80               1990.50
## 7029            72                103.20               7362.90
## 7030            11                 29.60                346.45
## 7031             4                 74.40                306.60
## 7032            66                105.65               6844.50
```

```
#install.packages("PerformanceAnalytics")
library(PerformanceAnalytics)
```

```
## Warning: package 'PerformanceAnalytics' was built under R version 3.6.2

## Loading required package: xts

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo

##
## Attaching package: 'PerformanceAnalytics'

## The following object is masked from 'package:graphics':
##
##     legend
```

```
quant_var=c('tenure','MonthlyCharges','TotalCharges')
chart.Correlation(custc[quant_var], method = c("spearman"),histogram = TRUE,
pch = "19", cex= 0.7)

## Warning in cor.test.default(as.numeric(x), as.numeric(y), method =
method):
## Cannot compute exact p-value with ties

## Warning in cor.test.default(as.numeric(x), as.numeric(y), method =
method):
## Cannot compute exact p-value with ties

## Warning in cor.test.default(as.numeric(x), as.numeric(y), method =
method):
## Cannot compute exact p-value with ties
```



Principal component analysis (**PCA**) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables (entities each of which takes on various numerical values) into a set of values of linearly uncorrelated variables called principal components.

PCA will help mathematically reduce correlated variables into smaller number of variables. Basically, can help us remove unnecessary data/variables/columns required for further analysis.

Normalization is a step to scale down the data to a same statistical level to apply analysis techniques. We aim to scale down data to the maximum level by bringing it on a same plane mathematically and apply various techniques to understand the covariance between the variables.

```
# apply PCA
pca<-prcomp(quant_var_df[,],scale=TRUE)
pca

## Standard deviations (1, .., p=3):
## [1] 1.4764138 0.8722163 0.2438052
##
## Rotation (n x k) = (3 x 3):
##                            PC1         PC2        PC3
## custc.tenure         0.5672112  0.60697524  0.5566440

## custc.MonthlyCharges 0.4857136 -0.79237469  0.3690862
## custc.TotalCharges   0.6650968  0.06101971 -0.7442600

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

(eigen_custc <- pca$sdev^2)

## [1] 2.1797978 0.7607612 0.0594410

names(eigen_custc) <- paste("PC",1:3,sep="")
eigen_custc

##       PC1       PC2       PC3
## 2.1797978 0.7607612 0.0594410

sumlambdas <- sum(eigen_custc)
sumlambdas

## [1] 3
```

```
propvar <- eigen_custc/sumlambdas
propvar

##        PC1        PC2        PC3
## 0.72659927 0.25358707 0.01981367

#pc1 holds 72 % of variance ,pc1 n pc2 holds 97% of variance




cumvar_custc <- cumsum(propvar)
cumvar_custc

##       PC1       PC2       PC3
## 0.7265993 0.9801863 1.0000000

matlambdas <- rbind(eigen_custc,propvar,cumvar_custc)
rownames(matlambdas) <- c("Eigenvalues","Prop. variance","Cum. prop.
variance")
round(matlambdas,4)

##                        PC1    PC2    PC3
## Eigenvalues         2.1798 0.7608 0.0594
## Prop. variance      0.7266 0.2536 0.0198
## Cum. prop. variance 0.7266 0.9802 1.0000

summary(pca)

## Importance of components:
##                           PC1    PC2     PC3
## Standard deviation     1.4764 0.8722 0.24381
## Proportion of Variance 0.7266 0.2536 0.01981
## Cumulative Proportion  0.7266 0.9802 1.00000

pca$rotation

##                             PC1         PC2        PC3
## custc.tenure          0.5672112  0.60697524  0.5566440
## custc.MonthlyCharges 0.4857136 -0.79237469  0.3690862
## custc.TotalCharges   0.6650968  0.06101971 -0.7442600

print(pca)

## Standard deviations (1, .., p=3):
## [1] 1.4764138 0.8722163 0.2438052
##
## Rotation (n x k) = (3 x 3):
##                             PC1         PC2        PC3
## custc.tenure          0.5672112  0.60697524  0.5566440
## custc.MonthlyCharges 0.4857136 -0.79237469  0.3690862
## custc.TotalCharges   0.6650968  0.06101971 -0.7442600
```

```r
# Sample scores stored in pca$x
pca$x
```

```
##                PC1           PC2           PC3
##  [1,] -1.9515181174  8.274669e-02 -4.014401e-01
##  [2,] -0.2057779131  2.351249e-01  6.880996e-02
##  [3,] -1.5179746956 -5.225050e-01 -1.100453e-01
##  [4,] -0.2023974032  8.916668e-01  1.545552e-01
##  [5,] -1.2331817270 -9.651127e-01  8.238317e-02
##  [6,] -0.4309064740 -1.561191e+00  3.539964e-01
##  [7,]  0.0535276173 -9.067434e-01  1.714106e-01
##  [8,] -1.6653308417  3.152631e-01 -2.878860e-01
##  [9,]  0.7674138152 -1.142341e+00  1.400150e-01
## [10,]  0.8973564193  9.916304e-01  1.691605e-01
## [11,] -1.1861082253 -1.348713e-01 -6.579932e-02
## [12,] -1.6937293946  7.487461e-01 -2.924833e-01
## [13,]  2.1619897920 -2.123429e-01 -9.940702e-02
## [14,]  1.8189019936 -5.404888e-01 -5.070428e-02
## [15,]  0.6037602435 -1.244654e+00  1.987681e-01
## [16,]  3.2740742087 -2.204729e-01 -4.186422e-01
## [17,] -0.6301094728  1.612950e+00  3.162192e-01
## [18,]  3.0640565996 -1.231471e-02 -2.852363e-01
## [19,] -1.1880182233 -3.489180e-01 -5.002402e-02
## [20,]  0.0203757132 -9.588209e-01  1.887876e-01
## [21,] -1.7904296418 -1.750922e-01 -2.844342e-01
## [22,] -1.8089863908  6.240915e-01 -3.318747e-01
## [23,] -2.1109628329  3.379546e-01 -5.172521e-01
## [24,]  0.8704939770  7.944130e-01  1.188204e-01
## [25,]  0.5007551564  5.653584e-01  8.662914e-02
## [26,] -0.4301563628  1.700044e-01  7.569452e-02
## [27,]  1.6182040881 -4.831103e-01 -5.514404e-02
## [28,] -1.9457649575  7.353816e-02 -3.972614e-01
## [29,]  2.5244263349  4.183942e-01 -1.318245e-01
## [30,] -0.7071827757 -4.108153e-01  3.984007e-02
## [31,]  2.7164281617  2.437100e-01 -2.101828e-01
## [32,] -0.8240035185 -1.617463e+00  3.767725e-01
## [33,] -0.2234287466 -1.806825e-01  2.786652e-02
```

```r
#observation values after applying pc

# Identifying the scores by their survival status
custch_pca <- cbind(data.frame(custc$Churn),pca$x)
custch_pca

##      custc.Churn           PC1           PC2           PC3
## 1             No -1.9515181174  8.274669e-02 -4.014401e-01
## 2             No -0.2057779131  2.351249e-01  6.880996e-02
## 3            Yes -1.5179746956 -5.225050e-01 -1.100453e-01
## 4             No -0.2023974032  8.916668e-01  1.545552e-01
## 5            Yes -1.2331817270 -9.651127e-01  8.238317e-02
## 6            Yes -0.4309064740 -1.561191e+00  3.539964e-01
## 7             No  0.0535276173 -9.067434e-01  1.714106e-01
## 8             No -1.6653308417  3.152631e-01 -2.878860e-01
## 9            Yes  0.7674138152 -1.142341e+00  1.400150e-01
## 10            No  0.8973564193  9.916304e-01  1.691605e-01



# Means of scores for all the PC's classified by Churn
#2 means are significantly different
churnmeansPC <- aggregate(custch_pca[,-1],by=list(Churn=custc$Churn),mean)
churnmeansPC

##   Churn        PC1        PC2         PC3
## 1    No  0.1442825  0.2285482 -0.01357851
## 2   Yes -0.3985718 -0.6313507  0.03750982

churnmeansPC <- churnmeansPC[rev(order(churnmeansPC$Churn)),]
churnmeansPC

##   Churn        PC1        PC2         PC3
## 2   Yes -0.3985718 -0.6313507  0.03750982
## 1    No  0.1442825  0.2285482 -0.01357851

churnfmeans <- t(churnmeansPC[,-1])
churnfmeans

##               2           1
## PC1 -0.39857182  0.14428253
## PC2 -0.63135072  0.22854823
## PC3  0.03750982 -0.01357851

colnames(churnfmeans) <- t(as.vector(churnmeansPC[1]))
churnfmeans

##             Yes          No
## PC1 -0.39857182  0.14428253
## PC2 -0.63135072  0.22854823
## PC3  0.03750982 -0.01357851
```

```r
# Standard deviations of scores for all the PC's classified by Survival
status
tabsdsPC <- aggregate(custch_pca[,-1],by=list(Churn=custc$Churn),sd)
tabfsds <- t(tabsdsPC[,-1])
colnames(tabfsds) <- t(as.vector(tabsdsPC[1]))
tabfsds
```

```
##            No       Yes
## PC1 1.5283502 1.2382399
## PC2 0.8299302 0.6456628
## PC3 0.2486151 0.2258550
```

```r
t.test(PC1~custc$Churn,data=custch_pca)
```

```
##
##  Welch Two Sample t-test
##
## data:  PC1 by custc$Churn
## t = 15.216, df = 4050.7, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.4729099 0.6127988
## sample estimates:
##  mean in group No mean in group Yes
##         0.1442825        -0.3985718
```

```r
t.test(PC2~custc$Churn,data=custch_pca)
```

```
##
##  Welch Two Sample t-test
##
## data:  PC2 by custc$Churn
## t = 45.545, df = 4224, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.8228840 0.8969138
## sample estimates:
##  mean in group No mean in group Yes
##         0.2285482        -0.6313507
```

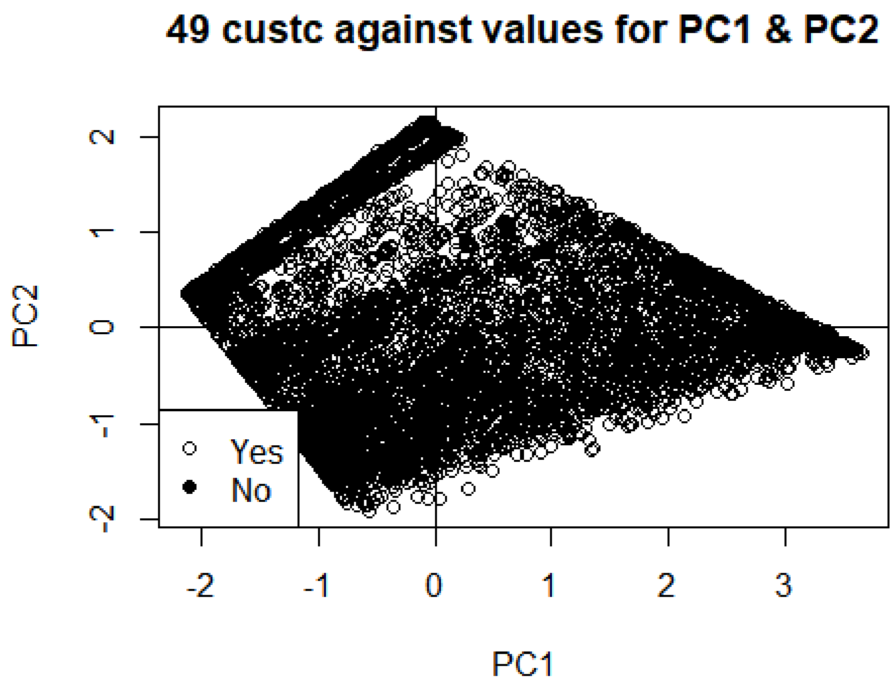
```r
t.test(PC3~custc$Churn,data=custch_pca)
```

```
##
##  Welch Two Sample t-test
##
## data:  PC3 by custc$Churn
## t = -8.1531, df = 3614.5, p-value = 4.842e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.06337385 -0.03880279
```

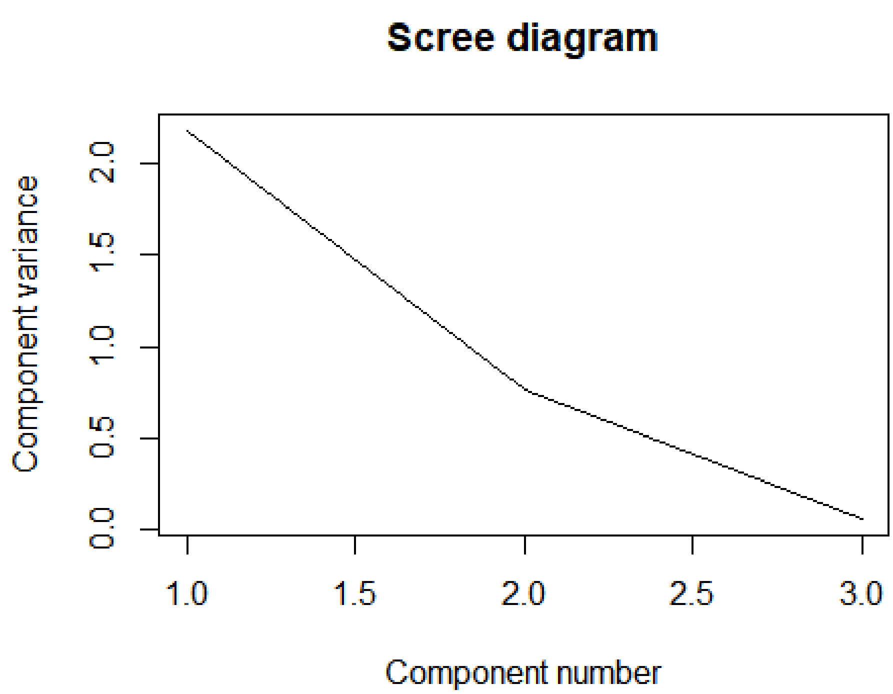
```
## sample estimates:
##  mean in group No mean in group Yes
##       -0.01357851        0.03750982

# F ratio tests
var.test(PC1~custc$Churn,data=custch_pca)

##
##  F test to compare two variances
##
## data:  PC1 by custc$Churn
## F = 1.5235, num df = 5162, denom df = 1868, p-value < 2.2e-16
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  1.412655 1.640810
## sample estimates:
## ratio of variances
##           1.523478

var.test(PC2~custc$Churn,data=custch_pca)

##
##  F test to compare two variances
##
## data:  PC2 by custc$Churn
## F = 1.6522, num df = 5162, denom df = 1868, p-value < 2.2e-16
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  1.532045 1.779483
## sample estimates:
## ratio of variances
##           1.652234

var.test(PC3~custc$Churn,data=custch_pca)

##
##  F test to compare two variances
##
## data:  PC3 by custc$Churn
## F = 1.2117, num df = 5162, denom df = 1868, p-value = 7.603e-07
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  1.123557 1.305021
## sample estimates:
## ratio of variances
##           1.211701
```

```r
# Levene's tests (one-sided)

library(car)

## Warning: package 'car' was built under R version 3.6.2

## Loading required package: carData

(LTPC1 <- leveneTest(PC1~custc$Churn,data=custch_pca))

## Levene's Test for Homogeneity of Variance (center = median)
##         Df F value    Pr(>F)
## group    1  202.47 < 2.2e-16 ***
##       7030
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

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

## [1] 1.281636e-45

(LTPC2 <- leveneTest(PC2~custc$Churn,data=custch_pca))

## Levene's Test for Homogeneity of Variance (center = median)
##         Df F value    Pr(>F)
## group    1  125.82 < 2.2e-16 ***
##       7030
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

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

## [1] 2.956281e-29

(LTPC3 <- leveneTest(PC3~custc$Churn,data=custch_pca))

## Levene's Test for Homogeneity of Variance (center = median)
##         Df F value   Pr(>F)
## group    1  22.763 1.87e-06 ***
##       7030
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

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

## [1] 9.348651e-07
```

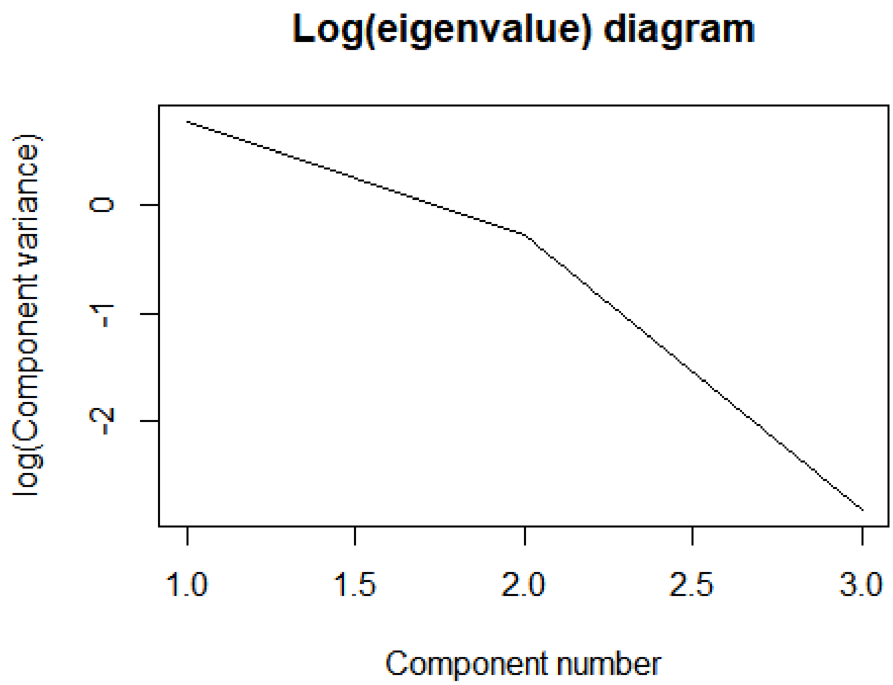
```r
# Plotting the scores for the first and second components
library(pander)

## Warning: package 'pander' was built under R version 3.6.2

library(Hmisc)

## Warning: package 'Hmisc' was built under R version 3.6.2

## Loading required package: lattice

## Warning: package 'lattice' was built under R version 3.6.2

## Loading required package: survival

## Warning: package 'survival' was built under R version 3.6.2

## Loading required package: Formula

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.6.2

##
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:base':
##
##     format.pval, units

panderOptions('knitr.auto.asis', FALSE)
plot(custch_pca$PC1, custch_pca$PC2,pch=ifelse(custch_pca$Churn ==
"Yes",1,16),xlab="PC1", ylab="PC2", main="49 custc against values for PC1 &
PC2")
abline(h=0)
abline(v=0)
legend("bottomleft", legend=c("Yes","No"), pch=c(1,16))
```




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

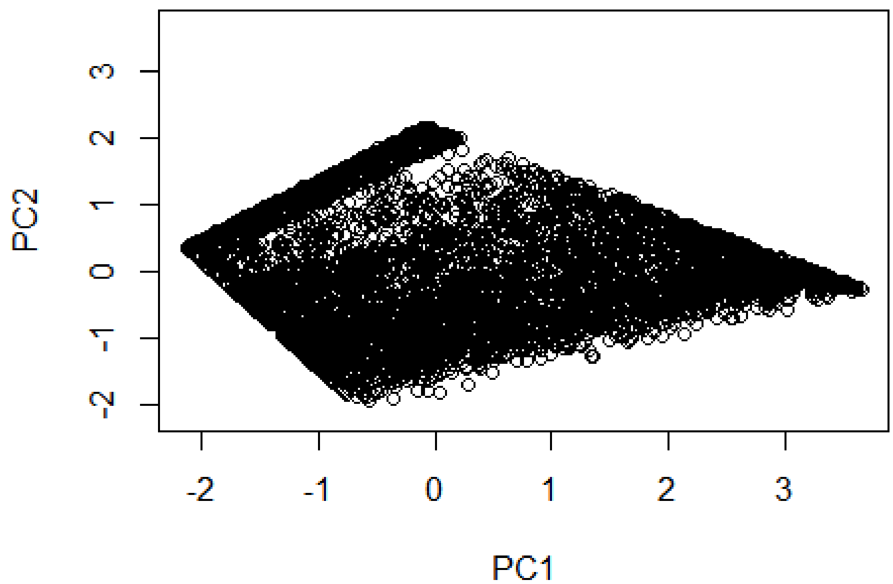


```r
plot(log(eigen_custc), xlab = "Component number",ylab = "log(Component
variance)", type="l",main = "Log(eigenvalue) diagram")
```

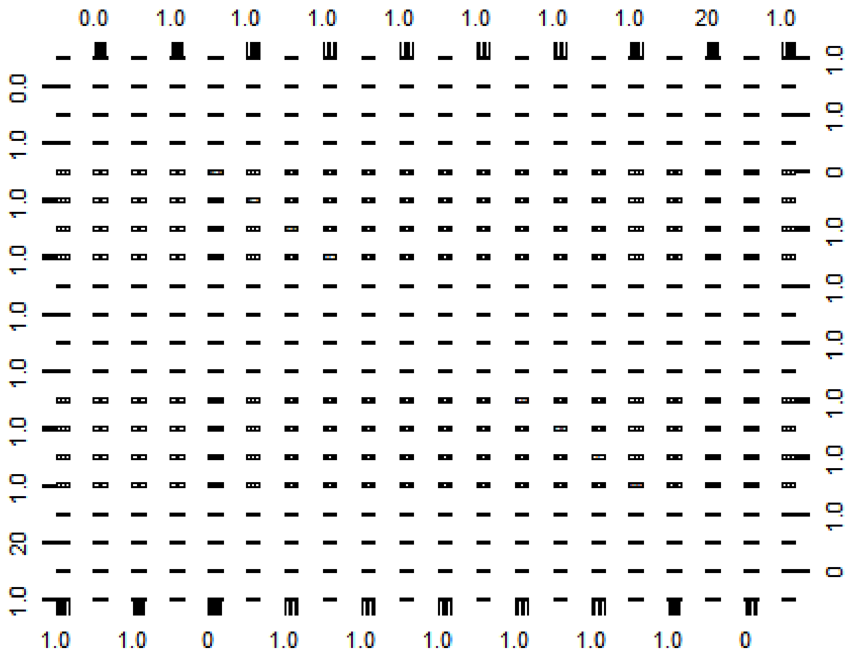



```r
#print(summary(pca))
#View(pca)

#diagonal --sum(diag)=3
diag(cov(pca$x))

##       PC1       PC2       PC3
## 2.1797978 0.7607612 0.0594410

xlim <- range(pca$x[,1])
pca$x[,1]

##    [1] -1.9515181174 -0.2057779131 -1.5179746956 -0.2023974032 -
1.2331817270
##    [6] -0.4309064740  0.0535276173 -1.6653308417  0.7674138152
0.8973564193
##   [11] -1.1861082253 -1.6937293946  2.1619897920  1.8189019936
0.6037602435
##   [16]  3.2740742087 -0.6301094728  3.0640565996 -1.1880182233
0.0203757132
##   [21] -1.7904296418 -1.8089863908 -2.1109628329  0.8704939770
0.5007551564
##   [26] -0.4301563628  1.6182040881 -1.9457649575  2.5244263349 -
0.7071827757
##   [31]  2.7164281617 -0.8240035185 -0.2234287466 -2.1101409530 -
```

```
1.6983790844
##   [36]  2.9390800391 -1.1315132999  0.8464122916  1.0787347018 -
0.3070467123
##   [41] -1.2946810008  1.6991072532 -1.6147285864  1.7060831501 -
0.6467191608
##   [46]  1.1774959607 -1.5955094988 -1.0747476421  1.2614177488
1.4130686595
##   [51]  1.1117349992 -0.1002462334 -0.2513008137 -0.7925726785
1.4371748482
##   [56]  0.0058500045  2.4511113853  2.8870104098 -1.0143950484
3.2384095870
##   [61]  1.6244319511  1.9718985962  0.7472986890 -0.8902630870 -
0.4817718003
##   [66] -1.1086837808  0.9656548941  0.2624100692 -0.7114076720 -
0.6847292619
##   [71] -1.6359162061 -0.6142815618  2.8982927952 -0.2230537832 -
1.3026115583
##   [76]  2.4141355517  0.2850785830 -1.2566413738  0.0542321487 -
0.6502948705
##   [81] -1.1378569399 -1.2291236838 -0.9444958290  1.4507848257 -
0.7220611428
##   [86] -0.3846204720 -0.0030818705 -0.7125539661 -0.8081866279 -
0.9541424857
##   [91]  0.3067312555 -1.2142917778  2.0926292176  2.9148695723
3.0160631002
##   [96] -0.6413049941  1.6479968688 -1.9765052522 -0.5997390887
0.4404161856
##  [101] -2.1101409530 -2.1224691526  1.0044670596  0.6834885058

## [7023,]  0.3051443073  2.331649e-02  7.191053e-02
## [7024,]  2.7655486394 -2.581811e-02 -2.590993e-01
## [7025,] -0.3169951455 -7.192543e-01  1.249540e-01
## [7026,] -0.9907460207 -4.372108e-01 -8.384002e-03
## [7027,] -0.0435386195  2.105030e+00  6.457500e-01
## [7028,]  0.0423843984 -7.429302e-01  1.505214e-01
## [7029,]  3.0249832177  1.040728e-01 -2.991418e-01
## [7030,] -1.6315721993  3.451417e-01 -2.816752e-01
## [7031,] -1.0817665431 -1.008931e+00  1.222548e-01
## [7032,]  2.7737792286 -1.227807e-01 -2.349467e-01

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



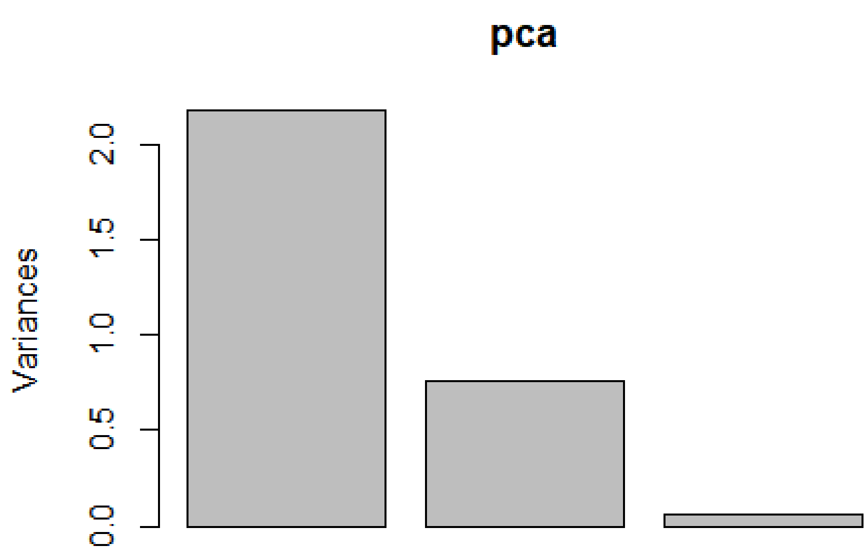

```
pca$rotation[,1]

##          custc.tenure custc.MonthlyCharges   custc.TotalCharges
##             0.5672112            0.4857136            0.6650968

pca$rotation

##                            PC1         PC2        PC3
## custc.tenure         0.5672112  0.60697524  0.5566440
## custc.MonthlyCharges 0.4857136 -0.79237469  0.3690862
## custc.TotalCharges   0.6650968  0.06101971 -0.7442600

plot(custc[,-1])
```



```
pca$x
```

```
##                PC1           PC2           PC3
##  [1,] -1.9515181174  8.274669e-02 -4.014401e-01
##  [2,] -0.2057779131  2.351249e-01  6.880996e-02
##  [3,] -1.5179746956 -5.225050e-01 -1.100453e-01
##  [4,] -0.2023974032  8.916668e-01  1.545552e-01
##  [5,] -1.2331817270 -9.651127e-01  8.238317e-02
##  [6,] -0.4309064740 -1.561191e+00  3.539964e-01
##  [7,]  0.0535276173 -9.067434e-01  1.714106e-01
##  [8,] -1.6653308417  3.152631e-01 -2.878860e-01
##  [9,]  0.7674138152 -1.142341e+00  1.400150e-01
## [10,]  0.8973564193  9.916304e-01  1.691605e-01
## [11,] -1.1861082253 -1.348713e-01 -6.579932e-02
## [12,] -1.6937293946  7.487461e-01 -2.924833e-01
## [13,]  2.1619897920 -2.123429e-01 -9.940702e-02
## [14,]  1.8189019936 -5.404888e-01 -5.070428e-02
## [15,]  0.6037602435 -1.244654e+00  1.987681e-01
## [16,]  3.2740742087 -2.204729e-01 -4.186422e-01
## [17,] -0.6301094728  1.612950e+00  3.162192e-01
## [18,]  3.0640565996 -1.231471e-02 -2.852363e-01
## [19,] -1.1880182233 -3.489180e-01 -5.002402e-02
## [20,]  0.0203757132 -9.588209e-01  1.887876e-01
## [21,] -1.7904296418 -1.750922e-01 -2.844342e-01
## [22,] -1.8089863908  6.240915e-01 -3.318747e-01
## [23,] -2.1109628329  3.379546e-01 -5.172521e-01
## [24,]  0.8704939770  7.944130e-01  1.188204e-01
```

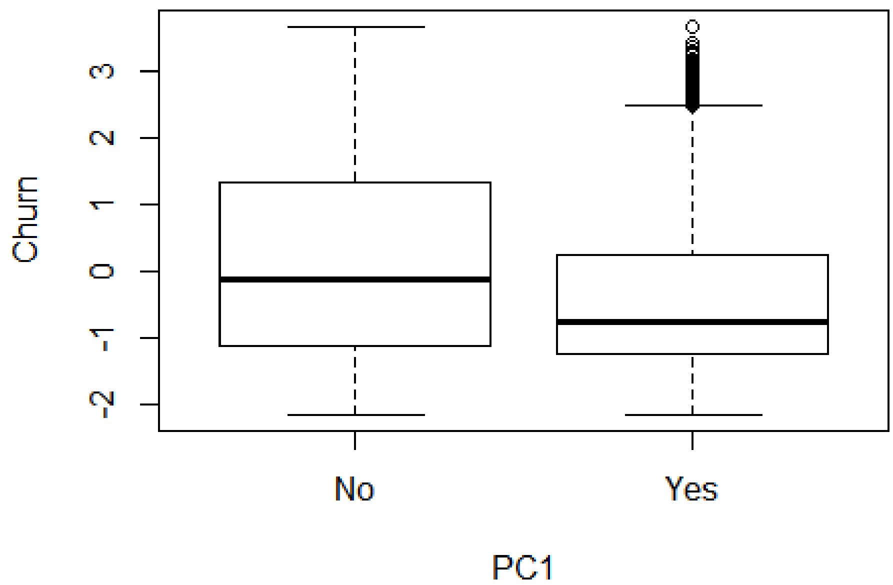
```
## [7023,]  0.3051443073  2.331649e-02  7.191053e-02
## [7024,]  2.7655486394 -2.581811e-02 -2.590993e-01
## [7025,] -0.3169951455 -7.192543e-01  1.249540e-01
## [7026,] -0.9907460207 -4.372108e-01 -8.384002e-03
## [7027,] -0.0435386195  2.105030e+00  6.457500e-01
## [7028,]  0.0423843984 -7.429302e-01  1.505214e-01
## [7029,]  3.0249832177  1.040728e-01 -2.991418e-01
## [7030,] -1.6315721993  3.451417e-01 -2.816752e-01
## [7031,] -1.0817665431 -1.008931e+00  1.222548e-01
## [7032,]  2.7737792286 -1.227807e-01 -2.349467e-01

plot(pca)
```

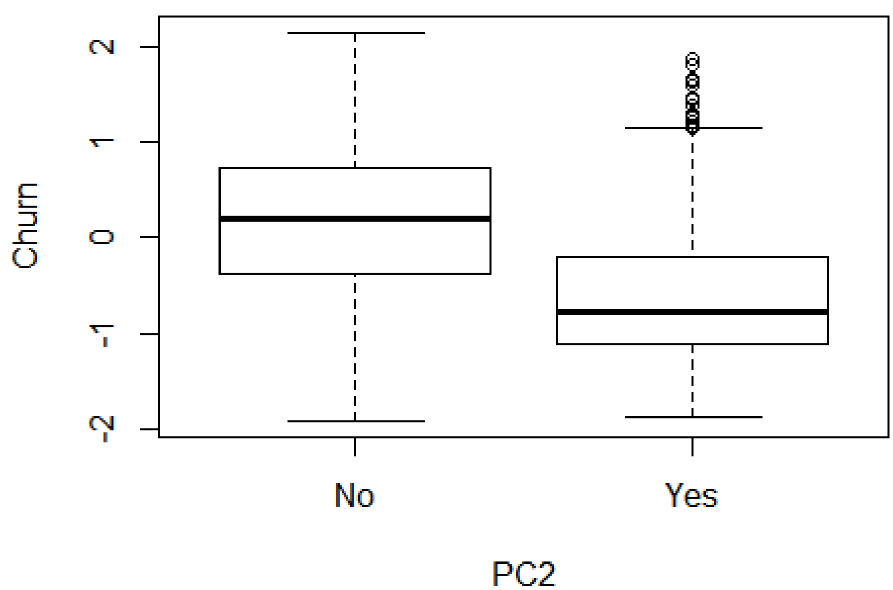


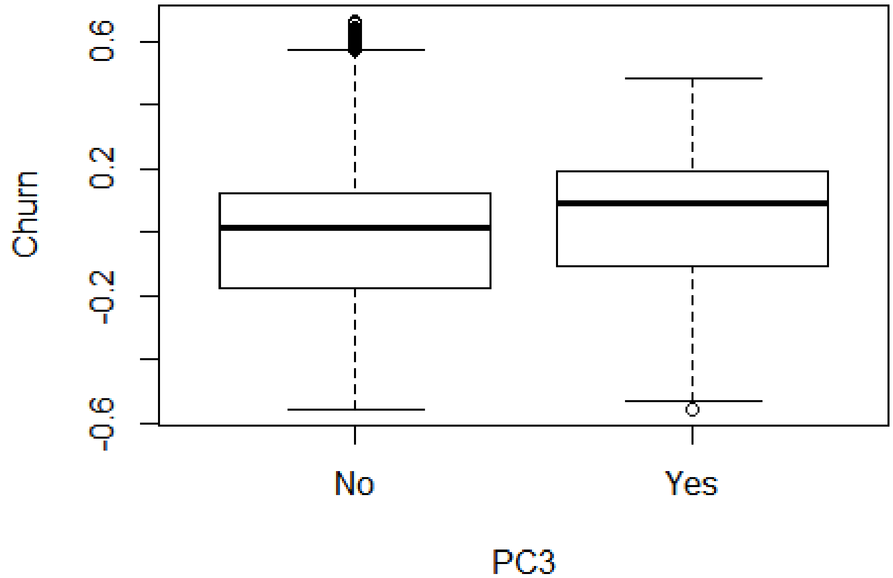
```r
#get the original value of the data based on PCA
center <- pca$center
scale <-  pca$scale
new_custc <- as.matrix(quant_var_df)
new_custc
```

```
##         custc.tenure custc.MonthlyCharges custc.TotalCharges
##    [1,]            1                29.85              29.85
##    [2,]           34                56.95            1889.50
##    [3,]            2                53.85             108.15
##    [4,]           45                42.30            1840.75
##    [5,]            2                70.70             151.65


## [7009,]           39                20.15             826.00
## [7010,]           12                19.20             239.00
## [7011,]           12                59.80             727.80
## [7012,]           72               104.95            7544.30
## [7013,]           63               103.50            6479.40
## [7014,]           44                84.80            3626.35
## [7015,]           18                95.05            1679.40
## [7016,]            9                44.20             403.35
## [7017,]           13                73.35             931.55
## [7018,]           68                64.10            4326.25
## [7019,]            6                44.40             263.05
## [7020,]            2                20.05              39.25
## [7021,]           55                60.00            3316.10
## [7022,]            1                75.75              75.75
## [7023,]           38                69.50            2625.25
## [7024,]           67               102.95            6886.25
## [7025,]           19                78.70            1495.10
## [7026,]           12                60.65             743.30
## [7027,]           72                21.15            1419.40
## [7028,]           24                84.80            1990.50
## [7029,]           72               103.20            7362.90
## [7030,]           11                29.60             346.45
## [7031,]            4                74.40             306.60
## [7032,]           66               105.65            6844.50
```

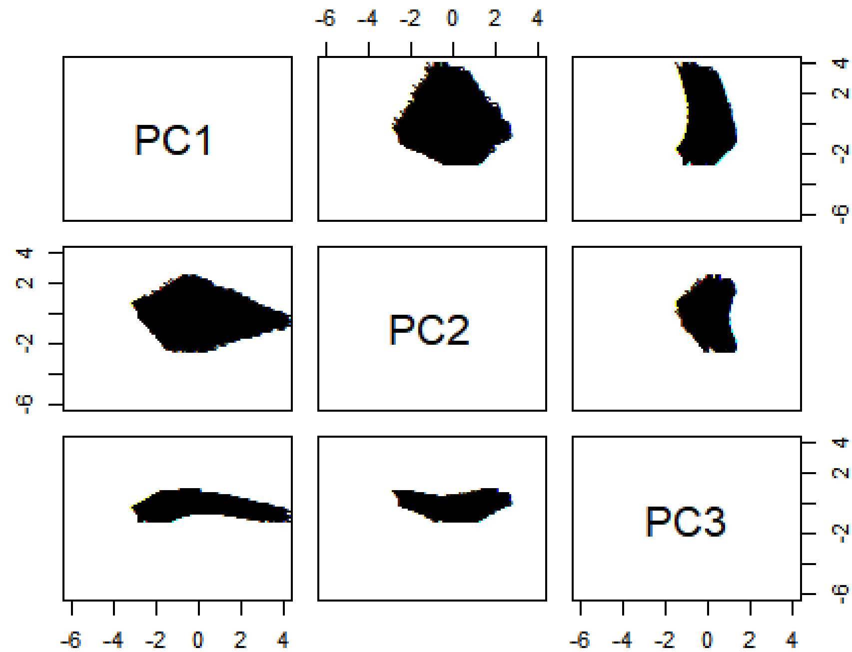
```r
out <- sapply(1:3,
function(i){plot(custc$Churn,pca$x[,i],xlab=paste("PC",i,sep=""),ylab="Churn"
)})
```


Churn
3
2
1
0
-1
-2
No
Yes
PC1


Churn
2
1
0
-1
-2
No
Yes
PC2




```
out

##       [,1]        [,2]        [,3]
## stats Numeric,10  Numeric,10  Numeric,10
## n     Numeric,2   Numeric,2   Numeric,2
## conf  Numeric,4   Numeric,4   Numeric,4
## out   Numeric,67  Numeric,16  Numeric,84
## group Numeric,67  Numeric,16  Numeric,84
## names Character,2 Character,2 Character,2

pairs(pca$x[,1:3], ylim = c(-6,4),xlim = c(-
6,4),panel=function(x,y,...){text(x,y,custc$Churn)})
```



Conclusion:

Thus, PCA helps represents data in a lower scale by removing redundant features by finding orthogonal principal components mentioned above as PC1, PC2 and PC3.

However, majority of our data here is non-liner nominal data. Here in our case, PCA is not useful to much of a higher extent. We can thus look to explore difference methods to fulfill our question of funding the cause of customer churn.