Assignment 3 - Part II

Matthew Dunne

July 20, 2018

Data

We will take the numerical variables from the GermanCredit data set, split into train and test data, and scale the train data by standardizing.

```
library(caret)
data("GermanCredit")
#just the numeric variables
newdata<-GermanCredit[c(1:7)]
#train and test data
intrain<-createDataPartition(y=newdata$Duration,p=0.7,list=FALSE)
train<-newdata[intrain,]
test<-newdata[-intrain,]
#scale the train data
scaledtrain<-as.data.frame(apply(X=train, MARGIN = 2, FUN = function(x){(x-mean(x))/sd(x)}))
```

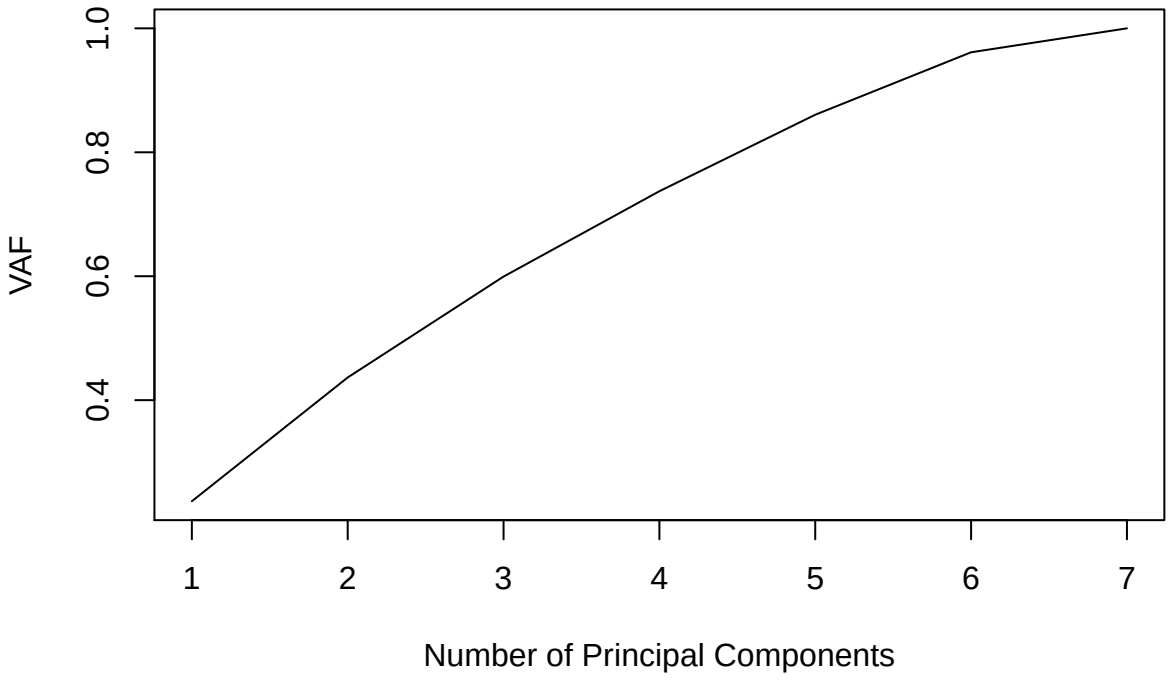
We also have to scale the testing data but do so by *using the means and standard deviations from the train data.*

```
means<-apply(X=train, MARGIN = 2, FUN = function(x){mean(x)})
sds<-apply(X=train, MARGIN = 2, FUN = function(x){sd(x)})
test$Duration<-(test$Duration-means[1])/sds[1]
test$Amount<-(test$Amount-means[2])/sds[2]
test$InstallmentRatePercentage<-(test$InstallmentRatePercentage-means[3])/sds[3]
test$ResidenceDuration<-(test$ResidenceDuration-means[4])/sds[4]
test$Age<-(test$Age-means[5])/sds[5]
test$NumberExistingCredits<-(test$NumberExistingCredits-means[6])/sds[6]
test$NumberPeopleMaintenance<-(test$NumberPeopleMaintenance-means[7])/sds[7]
```

Running the Principal Component Analysis

We use the princomp function on the scaled training data and collect the cumulative sum of variance accounted for by each additional PCA factor. We then plot this cumulative variance.

```
#run PCA on scaled train data
Train.PCA<-princomp(scaledtrain)
#cumulative sum of variance as we progress through principal components
VAF<-cumsum(Train.PCA$sdev^2/sum(Train.PCA$sdev^2))
plot(1:7, VAF, type = "l", xlab = "Number of Principal Components", ylab = "VAF", main = "Cumulative Su
```



Based on the kink in the plot, we would choose **six Principal Components**.

Plotting the Components

First we will look at the loadings.

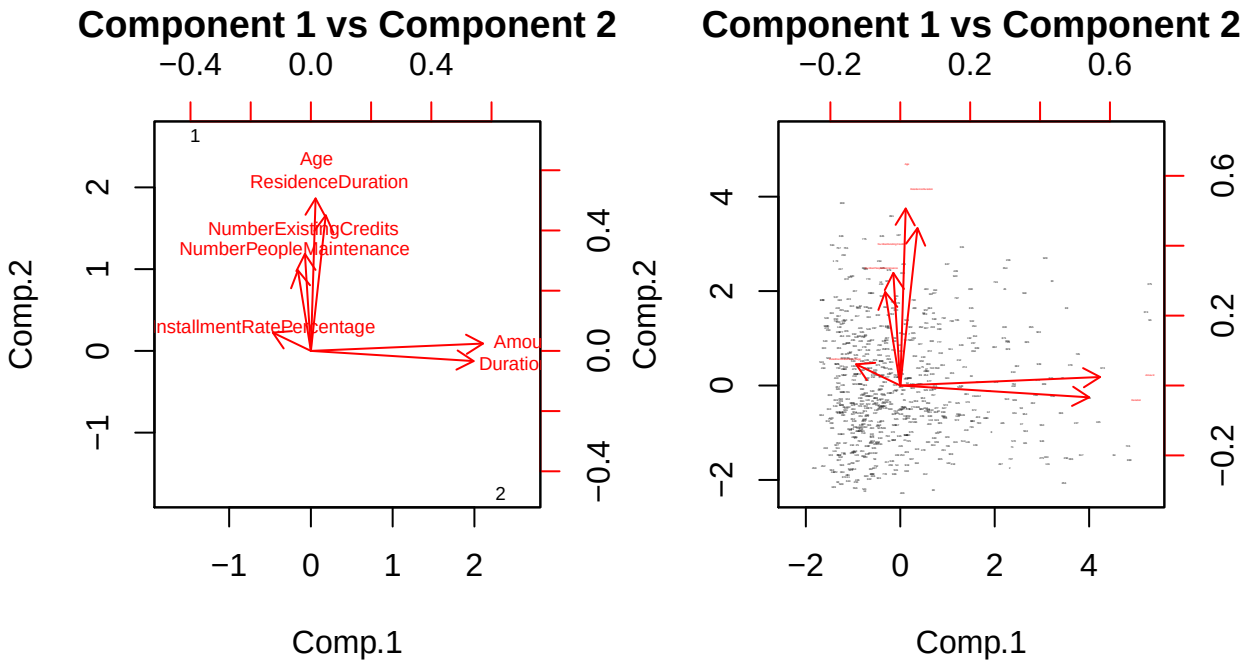
```
Train.PCA$loadings
```

```
##
## Loadings:
##                           Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7
## Duration                  -0.676        -0.257 -0.170  0.167         0.647
## Amount                    -0.715         0.137                      -0.680
## InstallmentRatePercentage  0.158        -0.770 -0.372  0.333 -0.159 -0.319
## ResidenceDuration                -0.563 -0.256  0.409         0.665
## Age                              -0.633         0.271 -0.138 -0.701  0.116
## NumberExistingCredits            -0.403  0.113 -0.721 -0.518  0.187
## NumberPeopleMaintenance          -0.334  0.492 -0.270  0.755
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7
## SS loadings     1.000  1.000  1.000  1.000  1.000  1.000  1.000
## Proportion Var  0.143  0.143  0.143  0.143  0.143  0.143  0.143
## Cumulative Var  0.143  0.286  0.429  0.571  0.714  0.857  1.000
```

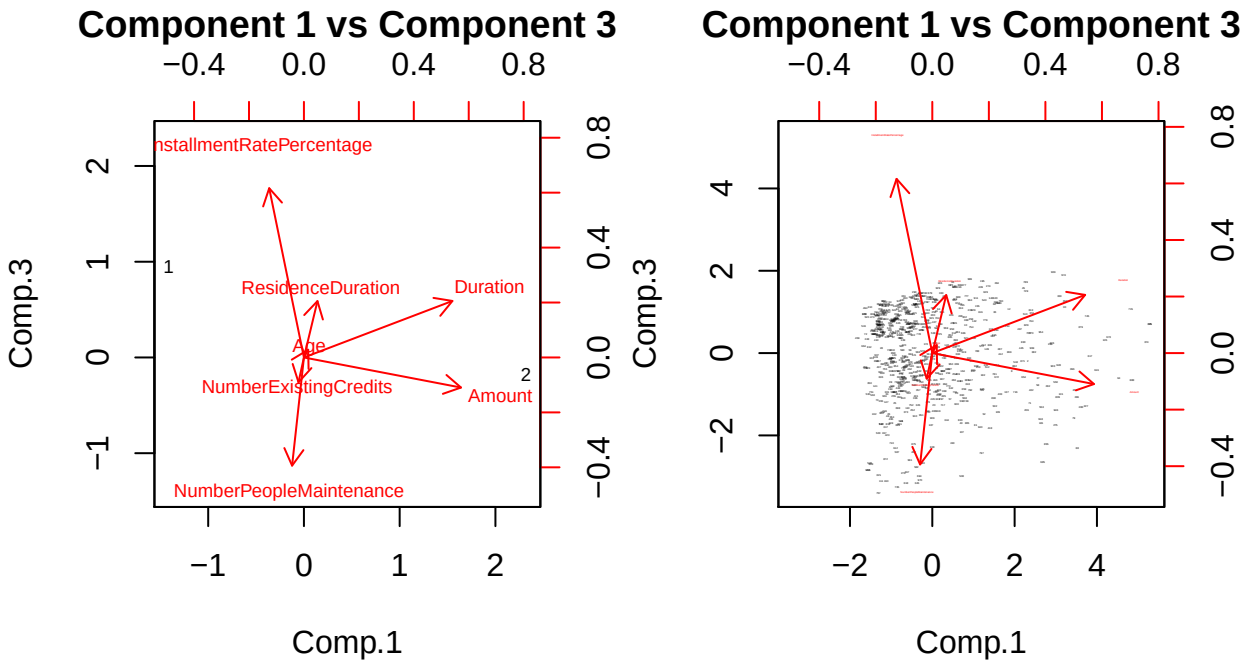
Then we make pairwise comparisons of all the loadings. The graphs on the left show the axes and the graphs on the right show the data points.

```
par(mfrow=c(1,2))
# do -Train.PCA so axes are pointing in right direction (higher age etc. meaning higher positive number
```

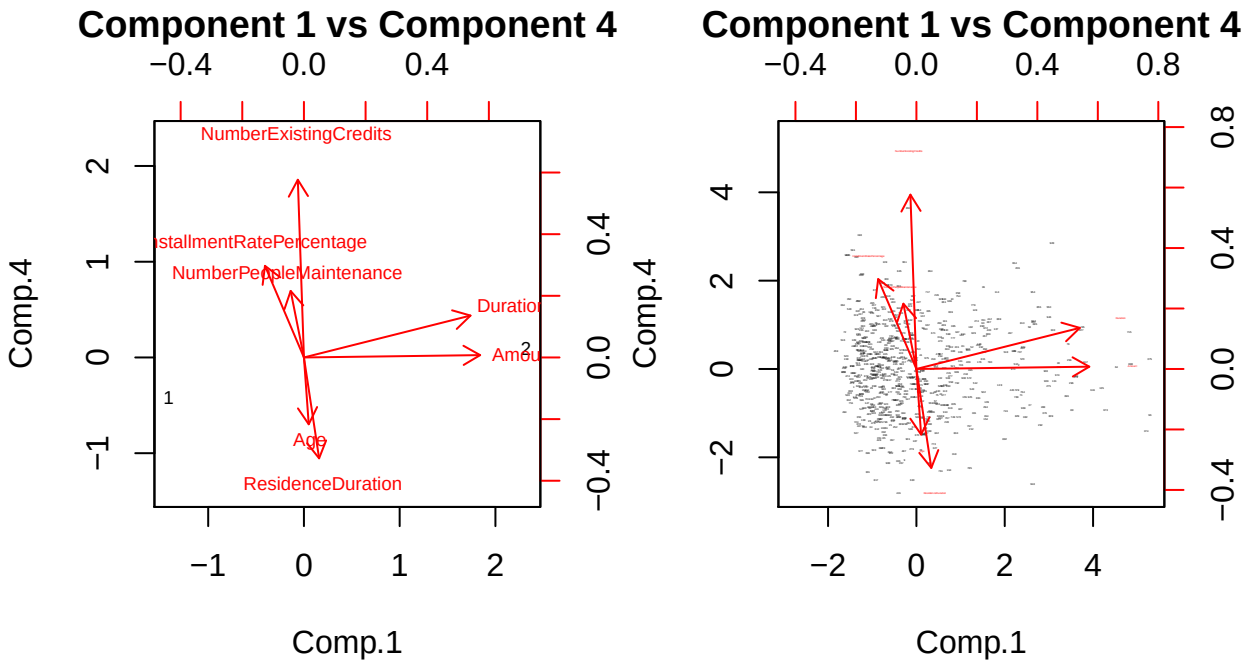
```
biplot(-Train.PCA$scores[1:2, c(1,2)], -Train.PCA$loadings[,c(1,2)], main = "Component 1 vs Component 2
biplot(-Train.PCA$scores[1:700, c(1,2)], -Train.PCA$loadings[,c(1,2)], main = "Component 1 vs Component
```



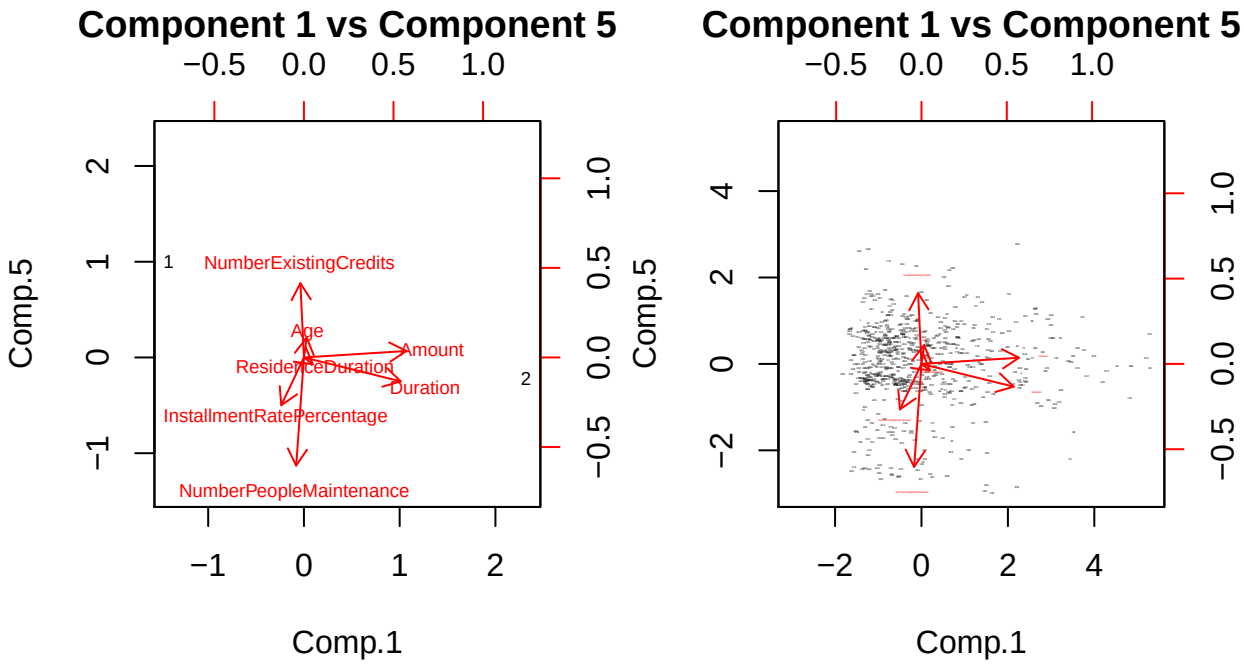
Here we see that characteristics of the person (Age, Residence Duration, etc.) are orthogonal to characteristics about the loan itself (Duration, Amount). It seems that loan duration, amount, and to a lesser extent installment rate percentage are the biggest influence on variance (PC1). And personal charachteristics of the borrower (age, number of existing credits, etc.) have the second most influence (PC2).



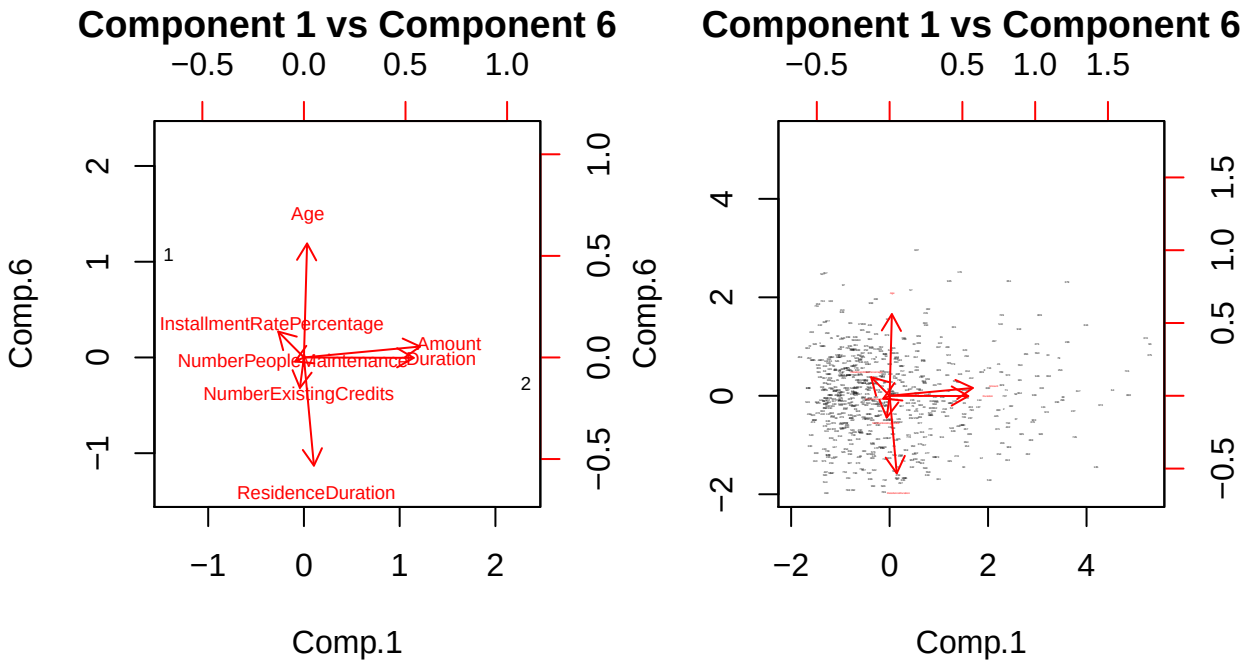
Again duration and amount form one axis with installment rate percentage and number of dependents forming opposite ends of the other axis.



At this point the graph starts to get less interpretable. What is noticable here is that Number of Existing Credits is now in the opposite direction from Age, Duration. The opposite of PC1 vs PC2



Here again we see a different set of variables influencing the Y-axis.



Interestingly, Age and Duration are now pointing in opposite directions.

Orthogonality of the Loadings

Show that the loadings are orthogonal to each other.

```
round(t(Train.PCA$loadings) %*% Train.PCA$loadings,2)
```

```
##         Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7
## Comp.1       1      0      0      0      0      0      0
## Comp.2       0      1      0      0      0      0      0
## Comp.3       0      0      1      0      0      0      0
## Comp.4       0      0      0      1      0      0      0
## Comp.5       0      0      0      0      1      0      0
## Comp.6       0      0      0      0      0      1      0
## Comp.7       0      0      0      0      0      0      1
```

We see that the dot product of loadings against other loadins always = 0 and the dot product of a loading with itself = 1.

Orthogonality of Scores

The covariance matrix of the schores shows 0's along the diagonals.

```
round(cov(Train.PCA$scores), 2)
```

```
##         Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7
## Comp.1    1.66    0.0   0.00   0.00   0.00   0.00   0.00
## Comp.2    0.00    1.4   0.00   0.00   0.00   0.00   0.00
## Comp.3    0.00    0.0   1.14   0.00   0.00   0.00   0.00
```

```
## Comp.4   0.00    0.0   0.00   0.96   0.00   0.00   0.00
## Comp.5   0.00    0.0   0.00   0.00   0.86   0.00   0.00
## Comp.6   0.00    0.0   0.00   0.00   0.00   0.71   0.00
## Comp.7   0.00    0.0   0.00   0.00   0.00   0.00   0.27
```

As does the correlation matrix.

```
round(cor(Train.PCA$scores), 2)
```

```
##        Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7
## Comp.1      1      0      0      0      0      0      0
## Comp.2      0      1      0      0      0      0      0
## Comp.3      0      0      1      0      0      0      0
## Comp.4      0      0      0      1      0      0      0
## Comp.5      0      0      0      0      1      0      0
## Comp.6      0      0      0      0      0      1      0
## Comp.7      0      0      0      0      0      0      1
```

So we can conclude that the scores are orthogonal.

Holdout Validation

Perform a holdout validation on the Principal Components Analysis solution. Try to reconstruct the test data by using the model you have created to create new scores and using the loadings from the original model.

```
#generate the new factor scores using the previously generated model on the test data.
new_factors<-predict(Train.PCA, newdata=test)
#Use the same loadings from the model to "re-create" the test data on the newly generated factor scores
#but only the top six factors
re_test<-new_factors[,1:6] %*% t(Train.PCA$loadings)[1:6,]
```

Here is the correlation matrix for the actual test data and the test data re-created with the top six principal components.

```
#correlation matrix for actual test data and re-created test data
round(cor(test, re_test), 2)
```

```
##                           Duration Amount InstallmentRatePercentage
## Duration                      0.93   0.80                       0.08
## Amount                        0.81   0.93                      -0.40
## InstallmentRatePercentage     0.06  -0.39                       0.98
## ResidenceDuration             0.03   0.02                       0.00
## Age                          -0.05   0.02                       0.17
## NumberExistingCredits         0.01   0.08                       0.01
## NumberPeopleMaintenance       0.04   0.10                      -0.05
##                           ResidenceDuration   Age NumberExistingCredits
## Duration                               0.01 -0.02                  0.04
## Amount                                 0.03  0.00                  0.04
## InstallmentRatePercentage              0.01  0.16                 -0.01
## ResidenceDuration                      1.00  0.24                  0.16
## Age                                    0.24  1.00                  0.19
## NumberExistingCredits                  0.16  0.19                  1.00
## NumberPeopleMaintenance                0.03  0.16                  0.12
##                           NumberPeopleMaintenance
## Duration                                     0.07
## Amount                                       0.06
## InstallmentRatePercentage                   -0.07
```

```
## ResidenceDuration                          0.03
## Age                                        0.17
## NumberExistingCredits                      0.12
## NumberPeopleMaintenance                    1.00
```

Though it is difficult to see with the printout, we do see high correlation between the original test data and the re-constructed test data (as demonstrated by the diagonals) with correlations for variables ranging from .93 to 1. Also there is correlation between different variables: Duration and Amount, Amount and Installment Rate Percentage.

The R^2 is:

```
#check the R^2 of the re-created test data with the original test data. Have to convert scaled test dat
#into matrix in order to convert into vector
round(cor(as.vector(as.matrix(test)), as.vector(re_test)), 2)
```

```
## [1] 0.98
```

Extremely good!

Varimax Rotation

Rotate the loadings using varimax(). The rotation does not change R^2 but can make the Factors and Loadings more interpretable.

```
rotated_loadings<-varimax(Train.PCA$loadings[,1:6])
new_loadings<-rotated_loadings$loadings
new_loadings
```

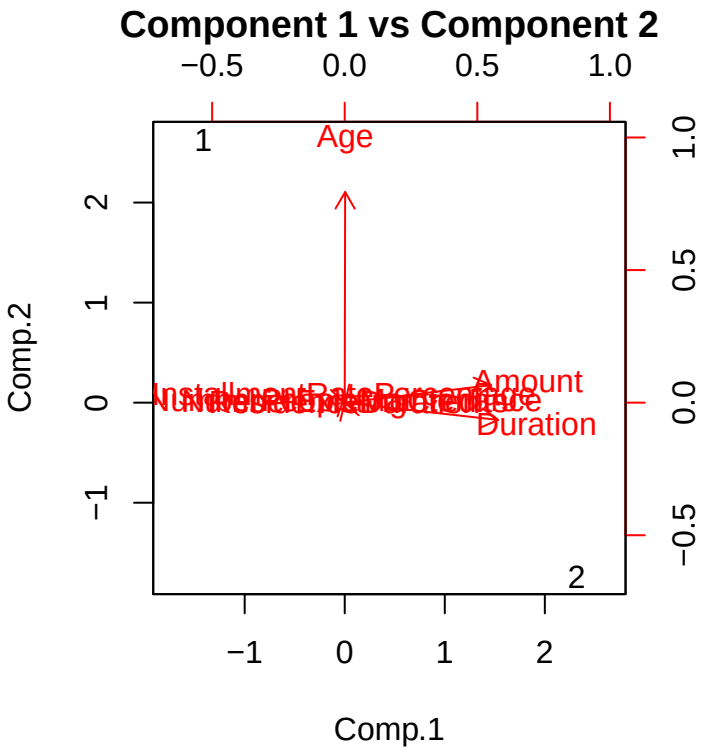
```
##
## Loadings:
##                           Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6
## Duration                  -0.723        -0.223
## Amount                    -0.690         0.228
## InstallmentRatePercentage               -0.948
## ResidenceDuration                                              0.999
## Age                               -0.993
## NumberExistingCredits                            -0.999
## NumberPeopleMaintenance                                  1.000
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6
## SS loadings     1.000  1.000  1.000  1.000  1.000  1.000
## Proportion Var  0.143  0.143  0.143  0.143  0.143  0.143
## Cumulative Var  0.143  0.286  0.429  0.571  0.714  0.857
```

These loadings are far more interpretable. The first PC is more clearly Duration and Amount, the second more clearly Age, etc.

Then we will plot the rotated loadings, 1 vs 2.

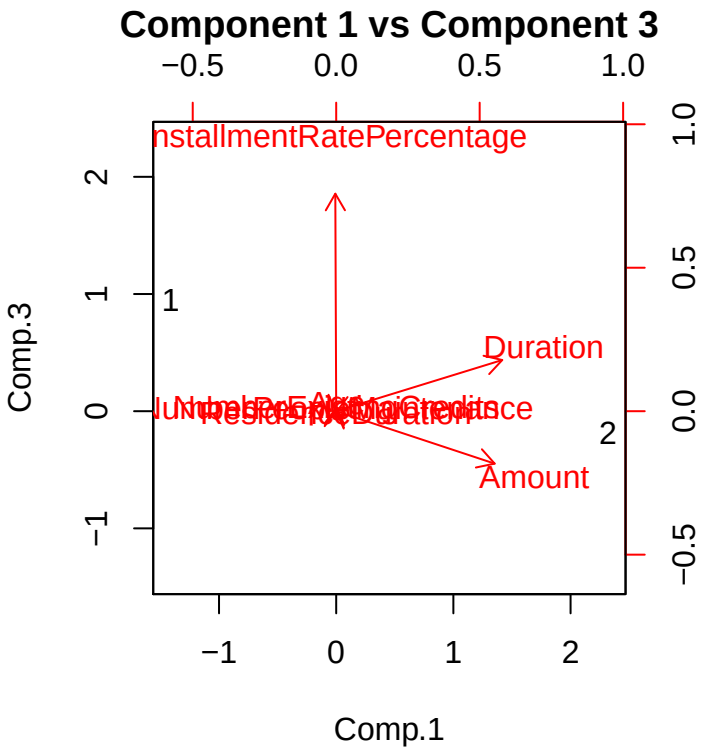
```
biplot(-Train.PCA$scores[1:2, c(1,2)], -new_loadings[,c(1,2)], main = "Component 1 vs Component 2")
```

```
## Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length
## = arrow.len): zero-length arrow is of indeterminate angle and so skipped
```



and 1 vs 3

```
biplot(-Train.PCA$scores[1:2, c(1,3)], -new_loadings[,c(1,3)], main = "Component 1 vs Component 3")
```



Because Duration and Amount are also part of PC3 they are noticeably less perpendicular to Installment Rate Percentage than they were versus PC2.

Commentary on PCA Solution

Our PCA solution does more to help us understand the nature of our data rather to reduce its dimensionality.

Using the scree plot and the loadings we derived from using the varimax function, we might conclude several things:

-We keep adding significant amount of information by including up to six PCs, though at a slowly decaying rate for each additional PC. The seventh PC does not add that much. Reducing from seven to six dimensions hardly does much to reduce dimensions, and we cannot focus on only one or two factors to explain the data.

-That being said, we can at least rank factors in order of importance to explain the variance in the data. Duration and Amount of loan are the most important, with Age (or at least mostly Age) second most important and explaining almost as much variance.