6372 Unit 9 Homework

Rashmi Patel

6/28/2021

## PCA Conceptual questions

1. TRUE/FALSE Principle component analysis is a predictive modeling technique such as linear regression and lda.

* **False-PCA is more of an EDA tool**

1. TRUE or FALSE? Technically speaking, PCA should not be applied to categorical variables.

* **True- It is challenging to find a suitable way to represent distances between variable categories**

1. An analyst conducts a pca on continuous variables 1 through 20 and settled on reducing the variable down to 4. The analyst then proceeds to conduct a linear regression using the 4 pca components as predictors and the response is variable 1. Why is this a horrible idea?

* **This is a terrible idea because PCA is meant to be used on predictor variables, not outcome ones.**

1. Why is it important to conduct PCA on standardized variables (aka using the correlation matrix)?

* **Variable scale is hugely important when it comes to PCA, they must be standardized otherwise, the loadings will not serve any purpose.**

## Exercise #1 PCA Basics

The example conducted in class did not do a very good job of illustrating the interpretation goals of PCA. For this reason, we will switch to a baseball data set to go over the basics and play around with interpretation. The baseball data set is located in the Lahman package. The data set is quite comprehensive having baseball player statistics dating back to 1871. We are going to examine the earliest year, 2016, by itself. Lets take a quick summary to see what variables we have.

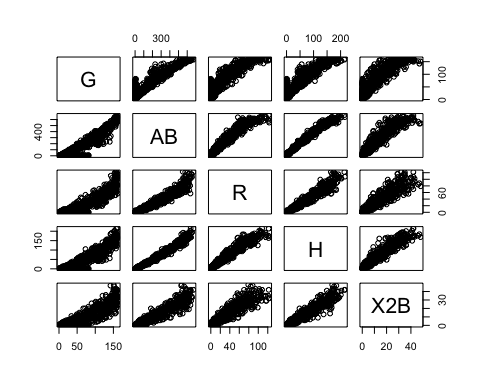
library(Lahman)  
data(Batting)  
index<-which(Batting$yearID==2016)  
Bat16<-Batting[index,]  
summary(Bat16)

## playerID yearID stint teamID lgID   
## Length:1483 Min. :2016 Min. :1.000 ATL : 60 AA: 0   
## Class :character 1st Qu.:2016 1st Qu.:1.000 SDN : 58 AL:734   
## Mode :character Median :2016 Median :1.000 LAN : 55 FL: 0   
## Mean :2016 Mean :1.092 PIT : 55 NA: 0   
## 3rd Qu.:2016 3rd Qu.:1.000 SEA : 54 NL:749   
## Max. :2016 Max. :4.000 LAA : 53 PL: 0   
## (Other):1148 UA: 0   
## G AB R H   
## Min. : 1.00 Min. : 0.0 Min. : 0.00 Min. : 0.00   
## 1st Qu.: 13.00 1st Qu.: 0.0 1st Qu.: 0.00 1st Qu.: 0.00   
## Median : 31.00 Median : 11.0 Median : 1.00 Median : 1.00   
## Mean : 47.51 Mean :111.6 Mean : 14.66 Mean : 28.51   
## 3rd Qu.: 69.00 3rd Qu.:155.0 3rd Qu.: 18.00 3rd Qu.: 36.00   
## Max. :162.00 Max. :672.0 Max. :123.00 Max. :216.00   
##   
## X2B X3B HR RBI   
## Min. : 0.000 Min. : 0.0000 Min. : 0.000 Min. : 0.00   
## 1st Qu.: 0.000 1st Qu.: 0.0000 1st Qu.: 0.000 1st Qu.: 0.00   
## Median : 0.000 Median : 0.0000 Median : 0.000 Median : 0.00   
## Mean : 5.566 Mean : 0.5887 Mean : 3.783 Mean : 13.99   
## 3rd Qu.: 7.000 3rd Qu.: 0.0000 3rd Qu.: 3.000 3rd Qu.: 15.50   
## Max. :48.000 Max. :11.0000 Max. :47.000 Max. :133.00   
##   
## SB CS BB SO   
## Min. : 0.000 Min. : 0.000 Min. : 0.00 Min. : 0.00   
## 1st Qu.: 0.000 1st Qu.: 0.000 1st Qu.: 0.00 1st Qu.: 0.00   
## Median : 0.000 Median : 0.000 Median : 0.00 Median : 4.00   
## Mean : 1.711 Mean : 0.675 Mean : 10.17 Mean : 26.29   
## 3rd Qu.: 1.000 3rd Qu.: 0.000 3rd Qu.: 13.00 3rd Qu.: 38.00   
## Max. :62.000 Max. :18.000 Max. :116.00 Max. :219.00   
##   
## IBB HBP SH SF   
## Min. : 0.0000 Min. : 0.000 Min. : 0.0000 Min. : 0.0000   
## 1st Qu.: 0.0000 1st Qu.: 0.000 1st Qu.: 0.0000 1st Qu.: 0.0000   
## Median : 0.0000 Median : 0.000 Median : 0.0000 Median : 0.0000   
## Mean : 0.6285 Mean : 1.113 Mean : 0.6912 Mean : 0.8186   
## 3rd Qu.: 0.0000 3rd Qu.: 1.000 3rd Qu.: 1.0000 3rd Qu.: 1.0000   
## Max. :20.0000 Max. :24.000 Max. :13.0000 Max. :15.0000   
##   
## GIDP   
## Min. : 0.000   
## 1st Qu.: 0.000   
## Median : 0.000   
## Mean : 2.508   
## 3rd Qu.: 3.000   
## Max. :26.000   
##

For those of you who do not know too much about baseball, the first 5 variables are just general information, G is the number of games played while the rest are information for players batting ability. G is games, AB is number of batting attempts, R-Runs, H-hits, X2B and X3B are doubles and triples, HR-homeruns, RBI-Runs Batted In. These are all general information on how well the batters can hit the ball. SB-stolen bases and CS- caught stealing are statistics about a players ability to run the bases, BB and IBB are when the batter gets a walk, HBP is hit by a pitch, SH and SF are sacrifice hit and fly’s where the player doesn’t get a hit but their hit is still productive, and lastley GIDP is grounded into double plays.

Sports data sets lend themselves well to PCA. We will use this example to go through similar concepts discussed in class. For starters lets just start off with just a few variables in the set to verify PCA is doing what we expect it to. Here is a quick scatterplot matrix. The variables here are highly correlated with each other.

reduced<-Bat16[,6:10]  
pairs(reduced)



Let’s take a quick look at the summary statistics and in particular lets calculate the variance of each variable and add them up to obtain the total variance.

apply(reduced,2,summary)

## G AB R H X2B  
## Min. 1.00000 0.0000 0.00000 0.00000 0.000000  
## 1st Qu. 13.00000 0.0000 0.00000 0.00000 0.000000  
## Median 31.00000 11.0000 1.00000 1.00000 0.000000  
## Mean 47.50573 111.6392 14.66217 28.50708 5.565745  
## 3rd Qu. 69.00000 155.0000 18.00000 36.00000 7.000000  
## Max. 162.00000 672.0000 123.00000 216.00000 48.000000

var.raw<-apply(reduced,2,var)  
var.raw

## G AB R H X2B   
## 2054.72517 31658.76653 665.77851 2363.88035 96.32951

#Total variance  
sum(var.raw)

## [1] 36839.48

We have been talking about the covariance matrix a lot lately. An estiamte for any given set of continouos variables can be obtained using the cov function. You can see that the diagonals of this matrix are the same as the variances calculated one at a time from before.

cov(reduced)

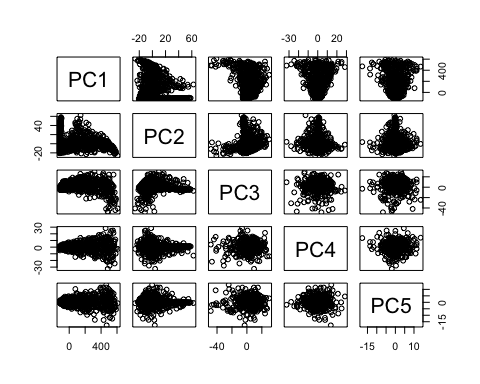
## G AB R H X2B  
## G 2054.7252 7423.272 1048.0832 1999.3891 391.50924  
## AB 7423.2723 31658.767 4479.3423 8574.1257 1678.82164  
## R 1048.0832 4479.342 665.7785 1229.5547 240.89233  
## H 1999.3891 8574.126 1229.5547 2363.8803 462.93830  
## X2B 391.5092 1678.822 240.8923 462.9383 96.32951

#Another way to get total variance  
sum(diag(cov(reduced)))

## [1] 36839.48

Running PCA is relatively straight forward. The following script conducts a PCA using the covariance matrix (nonstandardarized variables) and stores the results in an object. This object contains the eigenvectors, eigenvalue, and the new principle component vectors. Lets start by producing a correlation matrix to verify that new principle component variables are uncorrelated.

pc.result<-prcomp(reduced,scale.=FALSE)  
pc.scores<-pc.result$x  
pairs(pc.scores)



cor(pc.scores)

## PC1 PC2 PC3 PC4 PC5  
## PC1 1.000000e+00 3.883426e-15 -9.277437e-15 1.552181e-14 -1.433233e-14  
## PC2 3.883426e-15 1.000000e+00 -7.001366e-17 5.291058e-15 -5.983462e-15  
## PC3 -9.277437e-15 -7.001366e-17 1.000000e+00 -1.141962e-15 -6.650849e-16  
## PC4 1.552181e-14 5.291058e-15 -1.141962e-15 1.000000e+00 3.636272e-16  
## PC5 -1.433233e-14 -5.983462e-15 -6.650849e-16 3.636272e-16 1.000000e+00

We can again verify that the total variance in the new PC variables is exactly the same as the original data. The eigenvectors are stored inside of “pc.result” as well in the “rotation” object.

var.pca<-apply(pc.scores,2,var)  
var.pca

## PC1 PC2 PC3 PC4 PC5   
## 36462.584984 301.411677 50.103822 19.963419 5.416166

#Total Variance of PC's  
sum(var.pca)

## [1] 36839.48

#Total Variance of Original Variables.  
sum(var.raw)

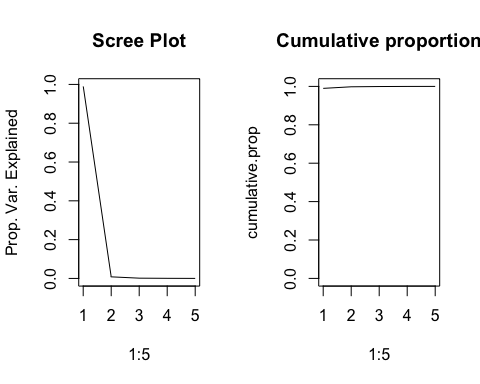
## [1] 36839.48

#List of eigenvectors  
pc.result$rotation

## PC1 PC2 PC3 PC4 PC5  
## G 0.22024127 0.97397742 -0.05059877 0.01730636 -0.001420923  
## AB 0.93156585 -0.19396824 0.30281622 -0.05283321 0.008507431  
## R 0.13202633 -0.04546002 -0.57626905 -0.80501489 0.019168837  
## H 0.25258860 -0.10598044 -0.73885937 0.57082967 -0.230667442  
## X2B 0.04946609 -0.02111491 -0.16656204 0.15170255 0.972805577

A scree plot of the eigenvalues used to determine how many pc’s to keep can be plotted in the following way:

par(mfrow=c(1,2))  
eigenvals<-(pc.result$sdev)^2  
plot(1:5,eigenvals/sum(eigenvals),type="l",main="Scree Plot",ylab="Prop. Var. Explained")  
cumulative.prop<-cumsum(eigenvals/sum(eigenvals))  
plot(1:5,cumulative.prop,type="l",main="Cumulative proportion",ylim=c(0,1))



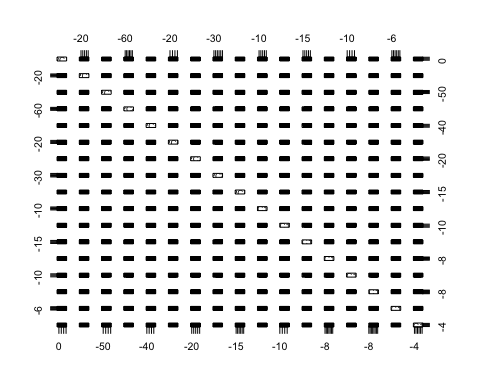
par(mfrow=c(1,1))

Since all of the variables are not on the same scale, we see a very similar phenomenon that we discussed in the pre live session. To conduct the pca on the correlation matrix, just set scale.=TRUE inside of the prcomp function.

## HW Assignment #1

1. Conduct the PCA analysis but use the entire set of variables starting with the column 6, the Games played variable, all the way down to the end at GIDP. Provide a scree plot and determine the amount of PC’s needed to retain approximately 90% of the total variation in the data set.

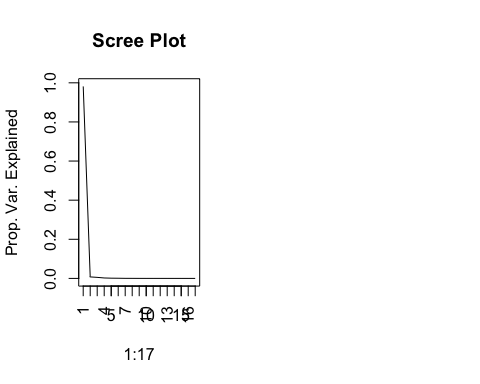
batfull <- Bat16[,6:22]  
pc.result2<-prcomp(batfull,scale.=FALSE)  
pc.scores2<-pc.result2$x  
pairs(pc.scores2)



cor(pc.scores2)

## PC1 PC2 PC3 PC4 PC5  
## PC1 1.000000e+00 -8.225935e-14 -6.513115e-17 -7.125810e-15 1.831074e-15  
## PC2 -8.225935e-14 1.000000e+00 -2.476390e-15 2.348867e-16 9.142583e-16  
## PC3 -6.513115e-17 -2.476390e-15 1.000000e+00 -4.164557e-16 -1.986087e-16  
## PC4 -7.125810e-15 2.348867e-16 -4.164557e-16 1.000000e+00 2.117756e-15  
## PC5 1.831074e-15 9.142583e-16 -1.986087e-16 2.117756e-15 1.000000e+00  
## PC6 9.627222e-15 2.196905e-16 3.057819e-16 1.598283e-15 2.041250e-15  
## PC7 1.847453e-14 3.063298e-15 -6.125906e-16 1.582150e-15 1.703187e-15  
## PC8 -1.025982e-14 -3.476807e-15 3.831626e-16 1.021772e-15 1.240179e-15  
## PC9 -1.847341e-14 -6.842301e-15 8.306578e-17 -2.183744e-16 -1.877649e-16  
## PC10 -4.849812e-15 -5.323694e-15 1.273454e-15 1.332100e-16 8.119559e-16  
## PC11 -1.598617e-15 -1.988538e-15 -8.806498e-16 3.182558e-15 -9.421444e-16  
## PC12 3.924110e-15 1.243093e-15 -9.061726e-17 1.010226e-15 2.342181e-16  
## PC13 8.119414e-16 -3.995649e-18 -3.374084e-16 2.099940e-15 -4.127787e-16  
## PC14 1.046033e-14 3.270567e-15 -2.345644e-16 -1.097005e-15 -1.224782e-15  
## PC15 -1.639031e-15 -1.030404e-15 -3.354842e-16 5.042718e-16 -1.163909e-15  
## PC16 -1.752310e-15 -6.786044e-16 -2.618838e-16 1.665822e-15 -2.177907e-16  
## PC17 -1.313515e-15 -1.086132e-15 3.465742e-16 -3.547729e-16 4.288925e-16  
## PC6 PC7 PC8 PC9 PC10  
## PC1 9.627222e-15 1.847453e-14 -1.025982e-14 -1.847341e-14 -4.849812e-15  
## PC2 2.196905e-16 3.063298e-15 -3.476807e-15 -6.842301e-15 -5.323694e-15  
## PC3 3.057819e-16 -6.125906e-16 3.831626e-16 8.306578e-17 1.273454e-15  
## PC4 1.598283e-15 1.582150e-15 1.021772e-15 -2.183744e-16 1.332100e-16  
## PC5 2.041250e-15 1.703187e-15 1.240179e-15 -1.877649e-16 8.119559e-16  
## PC6 1.000000e+00 7.321460e-16 -1.597795e-15 -5.631656e-16 -1.489276e-16  
## PC7 7.321460e-16 1.000000e+00 -3.287246e-16 1.036264e-15 6.546113e-16  
## PC8 -1.597795e-15 -3.287246e-16 1.000000e+00 -5.472353e-16 -2.884891e-16  
## PC9 -5.631656e-16 1.036264e-15 -5.472353e-16 1.000000e+00 -3.881256e-16  
## PC10 -1.489276e-16 6.546113e-16 -2.884891e-16 -3.881256e-16 1.000000e+00  
## PC11 -1.120155e-15 1.860638e-15 -1.517743e-15 -7.317286e-16 2.556724e-17  
## PC12 -6.205188e-16 -1.638723e-16 -9.839125e-16 2.008311e-16 -4.618522e-16  
## PC13 -5.248746e-17 2.811449e-16 -5.121637e-16 3.690748e-17 -5.924287e-16  
## PC14 1.011205e-15 -3.391002e-16 6.571569e-16 2.024569e-16 -5.041759e-17  
## PC15 4.699885e-16 -9.553449e-16 9.770837e-16 -9.114251e-17 6.923744e-16  
## PC16 -1.003562e-15 8.938691e-16 -5.947671e-16 4.925667e-16 -6.363497e-16  
## PC17 4.620240e-17 3.506311e-16 -2.366228e-17 5.573763e-16 1.101398e-16  
## PC11 PC12 PC13 PC14 PC15  
## PC1 -1.598617e-15 3.924110e-15 8.119414e-16 1.046033e-14 -1.639031e-15  
## PC2 -1.988538e-15 1.243093e-15 -3.995649e-18 3.270567e-15 -1.030404e-15  
## PC3 -8.806498e-16 -9.061726e-17 -3.374084e-16 -2.345644e-16 -3.354842e-16  
## PC4 3.182558e-15 1.010226e-15 2.099940e-15 -1.097005e-15 5.042718e-16  
## PC5 -9.421444e-16 2.342181e-16 -4.127787e-16 -1.224782e-15 -1.163909e-15  
## PC6 -1.120155e-15 -6.205188e-16 -5.248746e-17 1.011205e-15 4.699885e-16  
## PC7 1.860638e-15 -1.638723e-16 2.811449e-16 -3.391002e-16 -9.553449e-16  
## PC8 -1.517743e-15 -9.839125e-16 -5.121637e-16 6.571569e-16 9.770837e-16  
## PC9 -7.317286e-16 2.008311e-16 3.690748e-17 2.024569e-16 -9.114251e-17  
## PC10 2.556724e-17 -4.618522e-16 -5.924287e-16 -5.041759e-17 6.923744e-16  
## PC11 1.000000e+00 -1.013901e-15 1.618363e-15 1.364856e-16 -9.150168e-16  
## PC12 -1.013901e-15 1.000000e+00 6.827919e-16 3.760592e-17 -9.247050e-16  
## PC13 1.618363e-15 6.827919e-16 1.000000e+00 -1.205585e-15 3.406223e-16  
## PC14 1.364856e-16 3.760592e-17 -1.205585e-15 1.000000e+00 6.099934e-16  
## PC15 -9.150168e-16 -9.247050e-16 3.406223e-16 6.099934e-16 1.000000e+00  
## PC16 1.002936e-15 6.862335e-16 6.889785e-16 8.230884e-16 7.881224e-16  
## PC17 8.441392e-16 3.491099e-16 6.681854e-16 1.142874e-16 2.106901e-16  
## PC16 PC17  
## PC1 -1.752310e-15 -1.313515e-15  
## PC2 -6.786044e-16 -1.086132e-15  
## PC3 -2.618838e-16 3.465742e-16  
## PC4 1.665822e-15 -3.547729e-16  
## PC5 -2.177907e-16 4.288925e-16  
## PC6 -1.003562e-15 4.620240e-17  
## PC7 8.938691e-16 3.506311e-16  
## PC8 -5.947671e-16 -2.366228e-17  
## PC9 4.925667e-16 5.573763e-16  
## PC10 -6.363497e-16 1.101398e-16  
## PC11 1.002936e-15 8.441392e-16  
## PC12 6.862335e-16 3.491099e-16  
## PC13 6.889785e-16 6.681854e-16  
## PC14 8.230884e-16 1.142874e-16  
## PC15 7.881224e-16 2.106901e-16  
## PC16 1.000000e+00 8.489137e-16  
## PC17 8.489137e-16 1.000000e+00

par(mfrow=c(1,2))  
eigenvals<-(pc.result2$sdev)^2  
  
plot(1:17,eigenvals/sum(eigenvals),type="l",main="Scree Plot",ylab="Prop. Var. Explained")  
axis(1, at = seq(1, 17, by = 1), las=2)



Based on the scree plot, we will need two PC’s to retain approximately 90% of total variation of the dataset.

1. Provide the eigenvector matrix and examine the loading (coefficients) that determine the linear combinations of each principle component. Veryify that PC1 is eseentially a weighted average of all the variables together (minus the SH, sacrifice hit variable.)

#Eigenvector Matrix  
S <- cov(batfull)  
S

## G AB R H X2B X3B  
## G 2054.72517 7423.27231 1048.083247 1999.38913 391.509238 40.4922377  
## AB 7423.27231 31658.76653 4479.342282 8574.12570 1678.821642 172.8265447  
## R 1048.08325 4479.34228 665.778507 1229.55469 240.892334 25.7374659  
## H 1999.38913 8574.12570 1229.554688 2363.88035 462.938299 48.0926579  
## X2B 391.50924 1678.82164 240.892334 462.93830 96.329514 9.2639032  
## X3B 40.49224 172.82654 25.737466 48.09266 9.263903 2.0871050  
## HR 287.01946 1227.37641 182.779500 332.32947 65.483920 5.6090096  
## RBI 1011.27638 4325.04107 626.830808 1181.47478 234.486247 21.8900185  
## SB 120.50739 519.53862 79.319212 145.47607 27.132729 4.3876789  
## CS 45.75153 195.44812 29.152609 54.27828 10.319628 1.5322130  
## BB 719.42478 3013.70046 446.674872 814.55545 160.232144 16.6039964  
## SO 1619.42279 6831.28067 970.046720 1802.62888 355.243673 35.5029711  
## IBB 49.87966 209.79975 31.218527 58.76748 11.625996 1.1709569  
## HBP 77.48518 323.78516 47.200914 87.68273 17.190928 1.8293544  
## SH 13.13605 57.08689 6.671575 14.05549 2.408982 0.5982534  
## SF 57.43255 245.22197 35.062837 66.98610 13.164767 1.2991583  
## GIDP 172.34493 732.52878 101.332918 199.11415 38.981103 3.3142579  
## HR RBI SB CS BB SO  
## G 287.0194640 1011.276382 120.507395 45.7515267 719.424778 1619.42279  
## AB 1227.3764081 4325.041071 519.538621 195.4481214 3013.700455 6831.28067  
## R 182.7795001 626.830808 79.319212 29.1526090 446.674872 970.04672  
## H 332.3294736 1181.474778 145.476071 54.2782812 814.555448 1802.62888  
## X2B 65.4839203 234.486247 27.132729 10.3196283 160.232144 355.24367  
## X3B 5.6090096 21.890019 4.387679 1.5322130 16.603996 35.50297  
## HR 61.8421626 189.919911 15.564003 6.0710827 125.699743 285.88263  
## RBI 189.9199115 646.443188 61.742296 23.5057184 428.738163 947.08695  
## SB 15.5640029 61.742296 25.952024 7.0530147 48.410009 109.19882  
## CS 6.0710827 23.505718 7.053015 2.8821470 18.433100 40.88583  
## BB 125.6997428 428.738163 48.410009 18.4331001 348.346232 681.08990  
## SO 285.8826330 947.086950 109.198822 40.8858348 681.089903 1668.48635  
## IBB 9.4961962 32.630691 3.059114 1.0950848 24.451321 44.04827  
## HBP 13.0603761 44.653121 5.288528 2.0638346 31.890535 72.51212  
## SH 0.2061779 3.814271 2.247308 0.7099476 3.477516 12.04112  
## SF 9.9153797 35.844748 3.773404 1.3289949 24.378406 51.39670  
## GIDP 28.5556660 102.617835 9.628763 3.7332913 69.559379 151.39792  
## IBB HBP SH SF GIDP  
## G 49.8796614 77.4851802 13.1360493 57.4325518 172.344929  
## AB 209.7997462 323.7851598 57.0868921 245.2219682 732.528778  
## R 31.2185266 47.2009145 6.6715752 35.0628368 101.332918  
## H 58.7674772 87.6827331 14.0554944 66.9861030 199.114149  
## X2B 11.6259957 17.1909277 2.4089820 13.1647675 38.981103  
## X3B 1.1709569 1.8293544 0.5982534 1.2991583 3.314258  
## HR 9.4961962 13.0603761 0.2061779 9.9153797 28.555666  
## RBI 32.6306908 44.6531213 3.8142711 35.8447479 102.617835  
## SB 3.0591144 5.2885282 2.2473080 3.7734036 9.628763  
## CS 1.0950848 2.0638346 0.7099476 1.3289949 3.733291  
## BB 24.4513215 31.8905349 3.4775162 24.3784065 69.559379  
## SO 44.0482690 72.5121239 12.0411237 51.3966997 151.397916  
## IBB 3.4509306 2.3079753 0.1577819 1.9028709 5.106459  
## HBP 2.3079753 6.2422188 0.4418921 2.5003194 6.849998  
## SH 0.1577819 0.4418921 2.4497667 0.3211362 1.088766  
## SF 1.9028709 2.5003194 0.3211362 2.8597884 6.006468  
## GIDP 5.1064594 6.8499981 1.0887663 6.0064683 21.165088

#Coefficient loadings  
pc.result2$rotation

## PC1 PC2 PC3 PC4 PC5  
## G 0.213191057 9.634960e-01 -0.151757653 -0.028054144 0.031406927  
## AB 0.901755626 -2.016421e-01 -0.114190555 0.247945261 -0.005072979  
## R 0.127908089 -4.719224e-02 0.005725335 -0.392434357 -0.250293876  
## H 0.244280484 -1.269436e-01 -0.283510234 -0.256324196 -0.018720200  
## X2B 0.047861479 -2.496044e-02 -0.044739206 -0.080416313 0.045177986  
## X3B 0.004920320 -2.242089e-03 -0.011316315 0.004068684 -0.051728880  
## HR 0.035193092 -7.899234e-03 0.111293000 -0.243659325 0.244209062  
## RBI 0.123598108 -4.632786e-02 0.080081672 -0.586494258 0.608941759  
## SB 0.014795306 -1.041542e-02 -0.028704667 0.053951968 -0.286427492  
## CS 0.005562903 -2.298801e-03 -0.010612486 0.018715412 -0.085378314  
## BB 0.086321684 3.206259e-02 0.178627526 -0.527404337 -0.635288168  
## SO 0.195484270 9.298973e-02 0.910235740 0.131350751 0.035584600  
## IBB 0.006015504 -7.895414e-04 -0.003188427 -0.071969284 -0.008122328  
## HBP 0.009253523 3.588812e-03 0.010731130 -0.011153411 -0.012437473  
## SH 0.001590727 2.508194e-05 -0.003681986 0.057007375 -0.039970236  
## SF 0.006992190 -2.813248e-03 -0.006224507 -0.027293301 0.018134176  
## GIDP 0.020840798 -5.212596e-03 -0.032238719 -0.015244615 0.067012871  
## PC6 PC7 PC8 PC9 PC10  
## G 0.03713006 0.001805638 0.001480898 -0.001256269 0.0007240277  
## AB -0.21516207 -0.146899749 0.028043808 0.033242060 -0.0250431798  
## R 0.43985726 -0.372487997 0.571498872 0.135940318 0.1617284166  
## H 0.51244211 0.622840613 -0.125890696 -0.321852221 -0.0499509303  
## X2B 0.10010977 0.285730253 -0.075335375 0.888665676 0.1250332462  
## X3B 0.05015479 -0.023390932 -0.013619459 0.037575541 -0.1193349890  
## HR 0.01842864 -0.193663953 0.197402480 -0.226842727 -0.0088557097  
## RBI -0.10368104 -0.186399794 -0.321314224 0.067077710 -0.0623326674  
## SB 0.40089996 -0.471842593 -0.672846029 0.003567585 0.0729468560  
## CS 0.09737788 -0.088745081 -0.119238606 0.019645988 -0.0427800051  
## BB -0.46925166 0.144204859 -0.145054648 -0.030028484 -0.0361480222  
## SO 0.24914413 0.190637463 -0.036287261 -0.030939445 0.0352218887  
## IBB -0.04212594 0.035681852 -0.088328569 -0.046132195 -0.1020551034  
## HBP 0.02637238 -0.008400064 0.072445977 0.042886445 -0.4991623907  
## SH 0.01359474 -0.037820008 -0.037521451 0.009911489 0.0335999340  
## SF -0.02694276 -0.018411222 -0.051224315 0.004437482 0.0180548509  
## GIDP -0.12951643 0.080858249 -0.047969368 -0.152925180 0.8161809225  
## PC11 PC12 PC13 PC14 PC15  
## G -0.003016073 0.0045918025 -0.001107765 0.001397153 -0.0012807478  
## AB -0.032537657 0.0032306336 0.003380238 -0.011549267 0.0006522803  
## R 0.200458232 0.0519155486 0.056961505 -0.100851881 -0.0196040499  
## H -0.021297126 0.0166483401 -0.013744847 0.060939730 -0.0243602612  
## X2B -0.224359302 -0.1217795722 -0.126790100 0.014926197 -0.0141088560  
## X3B 0.116454216 0.1550218390 -0.019655063 -0.126478861 0.4742604214  
## HR -0.702793787 -0.2358566153 -0.354518216 0.137126454 -0.0133781722  
## RBI 0.258574363 0.1220396570 0.108718398 0.034076933 0.0588319831  
## SB -0.126995327 -0.1364599865 -0.005891310 0.011496281 -0.1140486414  
## CS -0.027918959 0.0030603832 0.020730426 0.106223574 0.5569426030  
## BB -0.063777457 -0.0001363035 -0.005548535 0.095616944 -0.0036966742  
## SO 0.055348641 0.0102124017 0.018909841 -0.024509489 -0.0016585392  
## IBB -0.006263520 -0.0586457974 -0.376812900 -0.890185204 0.0552422235  
## HBP 0.341086840 -0.7785572071 -0.066671211 0.118555759 0.0117198201  
## SH 0.353914944 0.2501967256 -0.830949736 0.328855266 -0.0569813746  
## SF 0.145601591 0.0139760419 0.052651528 -0.113555277 -0.6470105614  
## GIDP 0.216201879 -0.4432749037 -0.045314681 -0.033326182 0.1482930725  
## PC16 PC17  
## G 0.0001962413 -0.0001832461  
## AB 0.0036323849 -0.0038338482  
## R 0.0674776276 -0.0052309429  
## H -0.0095747262 0.0018999134  
## X2B -0.0452778351 0.0127531682  
## X3B -0.8208249605 -0.1599882546  
## HR -0.2010581908 0.0657130571  
## RBI 0.0681321713 -0.0376540739  
## SB -0.0236217642 -0.1549325206  
## CS 0.1730788804 0.7783160975  
## BB -0.0144353420 -0.0127262994  
## SO 0.0017385009 0.0053437775  
## IBB 0.1678608062 0.0467883642  
## HBP -0.0280738299 -0.0239343671  
## SH 0.0334985679 0.0063054040  
## SF -0.4510595389 0.5793588540  
## GIDP -0.1010140327 0.0055120386

mean(pc.result2$rotation[,1])

## [1] 0.1203274

1. Verify that PC2 has big negative loadings on triples (X3B), stolen bases (SB), caught stealing (CS), and sacrifice hits (SH). This variable could be interpreted to be a general indication of a players speed or general utility since all of the variables require situation awareness and running ability. (You dont need to provide an answer here, just verify))

pc.result2$rotation[,2]

## G AB R H X2B   
## 9.634960e-01 -2.016421e-01 -4.719224e-02 -1.269436e-01 -2.496044e-02   
## X3B HR RBI SB CS   
## -2.242089e-03 -7.899234e-03 -4.632786e-02 -1.041542e-02 -2.298801e-03   
## BB SO IBB HBP SH   
## 3.206259e-02 9.298973e-02 -7.895414e-04 3.588812e-03 2.508194e-05   
## SF GIDP   
## -2.813248e-03 -5.212596e-03

Verified

##PCA as an exploratory technique for Classification This exercise is designed to walk you through how PCA can be used as an informative unsupervised analysis of your predictors to get a high level view of whether the predictors are actually going to do a good job or not before a predictive model for categorical responses is even applied.

The following data set is a breast cancer data set that has numerous measurements taken from tumor biopsies. The goal of using this data set is to predict using the metrics alone if the biopsy is cancer or not. When continuous variables are available it is often helpful to create a pairs plot of data color coded by the response status (Diagnostis). The first variable is an id number and is not needed.

bc<-read.table("https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/wdbc.data",header=F,sep=",")  
names(bc)<- c('id\_number', 'diagnosis', 'radius\_mean',   
 'texture\_mean', 'perimeter\_mean', 'area\_mean',   
 'smoothness\_mean', 'compactness\_mean',   
 'concavity\_mean','concave\_points\_mean',   
 'symmetry\_mean', 'fractal\_dimension\_mean',  
 'radius\_se', 'texture\_se', 'perimeter\_se',   
 'area\_se', 'smoothness\_se', 'compactness\_se',   
 'concavity\_se', 'concave\_points\_se',   
 'symmetry\_se', 'fractal\_dimension\_se',   
 'radius\_worst', 'texture\_worst',   
 'perimeter\_worst', 'area\_worst',   
 'smoothness\_worst', 'compactness\_worst',   
 'concavity\_worst', 'concave\_points\_worst',   
 'symmetry\_worst', 'fractal\_dimension\_worst')  
#Getting a look at the distribution  
table(bc$diagnosis)

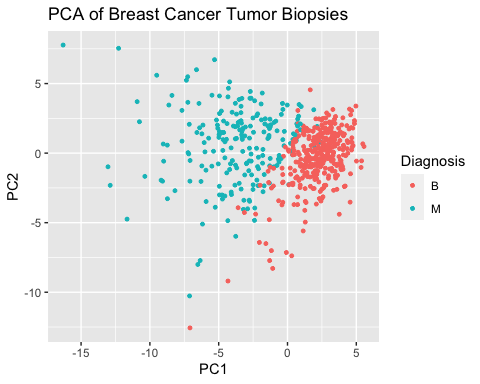
##   
## B M   
## 357 212

#Scatter plots color coded by response for just the first few variables  
#pairs(bc[,3:6],col=bc$diagnosis)

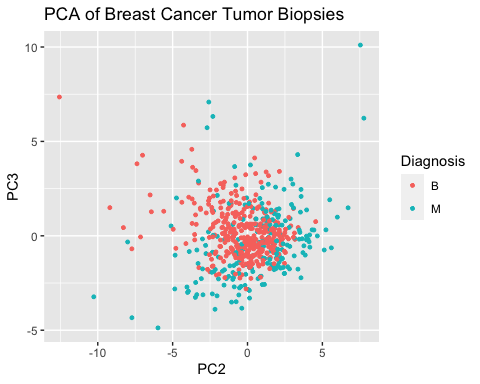
So we can see from this pairs plot of just the first few variables, seperation between the cancer and non cancer groups are pretty well seperated. Unfortunately we may not always see clear seperations but that does not necesarily mean that something like LDA or some other predcictive tool won’t work. It could be due to the fact we cant see the seperation of the groups unless we can actually see in higher dimensions. One way to still get at this, is to conduct a PCA analysis and provide a some scatterplots for the first few PC’s. If seperation exists in the PC’s, then a predictive model will probably do well.

Below we will conduct PCA on all of the predictors and plot the first few PC’s against each other and look for speration. The number of PCs to explore can be dictated by the scree plot.

pc.bc<-prcomp(bc[,-c(1,2)],scale.=TRUE)  
pc.bc.scores<-pc.bc$x  
#Adding the response column to the PC's data frame  
pc.bc.scores<-data.frame(pc.bc.scores)  
pc.bc.scores$Diagnosis<-bc$diagnosis  
#Use ggplot2 to plot the first few pc's  
library(ggplot2)  
ggplot(data = pc.bc.scores, aes(x = PC1, y = PC2)) +  
 geom\_point(aes(col=Diagnosis), size=1)+  
 ggtitle("PCA of Breast Cancer Tumor Biopsies")



ggplot(data = pc.bc.scores, aes(x = PC2, y = PC3)) +  
 geom\_point(aes(col=Diagnosis), size=1)+  
 ggtitle("PCA of Breast Cancer Tumor Biopsies")



So we can see in the first graphic a clear seperation exists for the two cancer groups. So the PCA is telling us in effect what we already know from looking at the original variables. The power of this approach is that you only need to look at 2-4 graphs each time, versus potentially having to examine massive scatterplot matrices to see if anything is there or not!

##HW Assignment 2 1. Given what we see in the PCA analysis, its not too suprising that an LDA will probably do a good job here in predicting the categorical responses. Perform an LDA on the original set of variables and calculate a confusion matrix. Note: For this problem you do not have to do a training and test set split, lets recognize that the prediction performance that we obtain is protentially biased too low due to overfitting. The main point here is that the accuracy is pretty good as expected via the PCA look.

library(MASS)  
mylda<-lda(diagnosis~.,data=bc)  
pred<-predict(mylda,newdata=bc)$class #Predictions can come in many forms, the class form provides the categorical level of your response.  
Truth<-bc$diagnosis  
x<-table(pred,Truth) # Creating a confusion matrix  
x

## Truth  
## pred B M  
## B 355 18  
## M 2 194

#Missclassification Error  
ME<-(x[2,1]+x[1,2])/569  
ME

## [1] 0.03514938

#Calculating overall accuracy  
1-ME

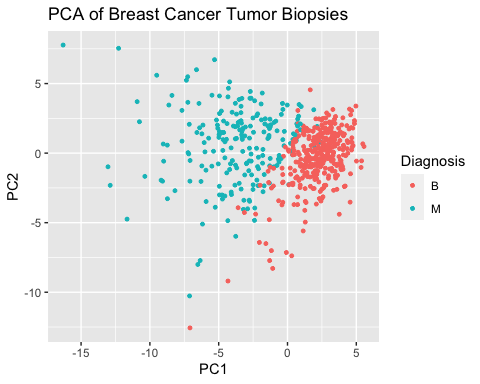
## [1] 0.9648506

1. Consider now another great sanity check when building predictive models. The code below takes the original data set and randomly scrambles the response variable. This effectively breaks up any relationship that existed between the predictors and the response.

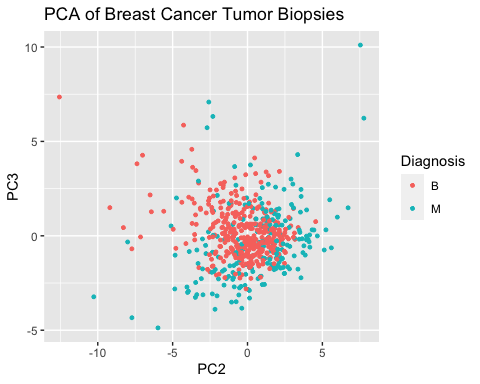
fake<-bc  
fake$diagnosis<-sample(fake$diagnosis,569,replace=F)

1. Plot PC1 and PC2 using the scrambled data set.
2. Perform an LDA with this data set and look at the confusion matrix. Do they correspond?

#A)  
pc.bc<-prcomp(fake[,-c(1,2)],scale.=TRUE)  
pc.bc.scores<-pc.bc$x  
#Adding the response column to the PC's data frame  
pc.bc.scores<-data.frame(pc.bc.scores)  
pc.bc.scores$Diagnosis<-bc$diagnosis  
#Use ggplot2 to plot the first few pc's  
library(ggplot2)  
ggplot(data = pc.bc.scores, aes(x = PC1, y = PC2)) +  
 geom\_point(aes(col=Diagnosis), size=1)+  
 ggtitle("PCA of Breast Cancer Tumor Biopsies")



ggplot(data = pc.bc.scores, aes(x = PC2, y = PC3)) +  
 geom\_point(aes(col=Diagnosis), size=1)+  
 ggtitle("PCA of Breast Cancer Tumor Biopsies")



#B)  
mylda<-lda(diagnosis~.,data=fake)  
pred<-predict(mylda,newdata=fake)$class #Predictions can come in many forms, the class form provides the categorical level of your response.  
Truth<-bc$diagnosis  
x<-table(pred,Truth) # Creating a confusion matrix  
x

## Truth  
## pred B M  
## B 329 173  
## M 28 39

#Missclassification Error  
ME<-(x[2,1]+x[1,2])/569  
ME

## [1] 0.3532513

#Calculating overall accuracy  
1-ME

## [1] 0.6467487

Yes, the confusion matrices look similar. This is not surprising as the data has simply been rescrambled, and not changed.

Note: This little trick is extremely helpful when you are predicting a response that is heavily imbalance (ex: lots of Cancer obs, few Healthy ones ). LDA and other algorithms can behave quite wierdly in extreme cases and prediction performances can look good all the time. By conducting a seperate analysis on scrambled data, if the prediction performance still looks good, you’ve recognized a problem. We can discuss this topic more as we get closer to finishing up Project 2.