

Fetal Health

Finn de Lange

2025-05-24

Data import

```
data <- read_csv("fetal_health.csv")

for (col_name in names(data)) {
  if (is.numeric(data[[col_name]])) {
    hist(data[[col_name]],
         main = paste("Histogram of", col_name),
         xlab = col_name,
         col = "skyblue",
         border = "white")
  } else {
    print(paste(col_name, "is not numeric and was skipped."))
  }
}
```

A visual inspection revealed that many of these variables do not follow multivariate normality. We can further test this using a Mardia test.

A quick disclaimer that in order to improve readability, many of the outputs have been suppressed. If you would like to see these, I recommend downloading this RMD file and running it yourself.

```
library(MVN)
data <- read_csv("fetal_health.csv")
MVN::mvn(data = data, mvnTest = "mardia")[2]
```

Based on the outputs, our data is not multivariate normal. I considered transforming some of the data using logarithms, but I decided to try fitting a model without transformation first. If we can properly transform the data, using LDA or QDA might be a viable choice for classification instead.

Given that the fetal health column is recorded and can tell us if the infant was classified as normal, suspect or pathological, we should be able to create model that can classify into two groups - normal or of concern.

We can first create a new column by collapsing the suspect and pathological groups into a single group, and re-code the system to use 0 for normal and 1 for of concern.

```
data <- data %>%
  mutate(
    recoded = ifelse(fetal_health == 1, 0, 1)
  ) %>%
  dplyr::select(-fetal_health)

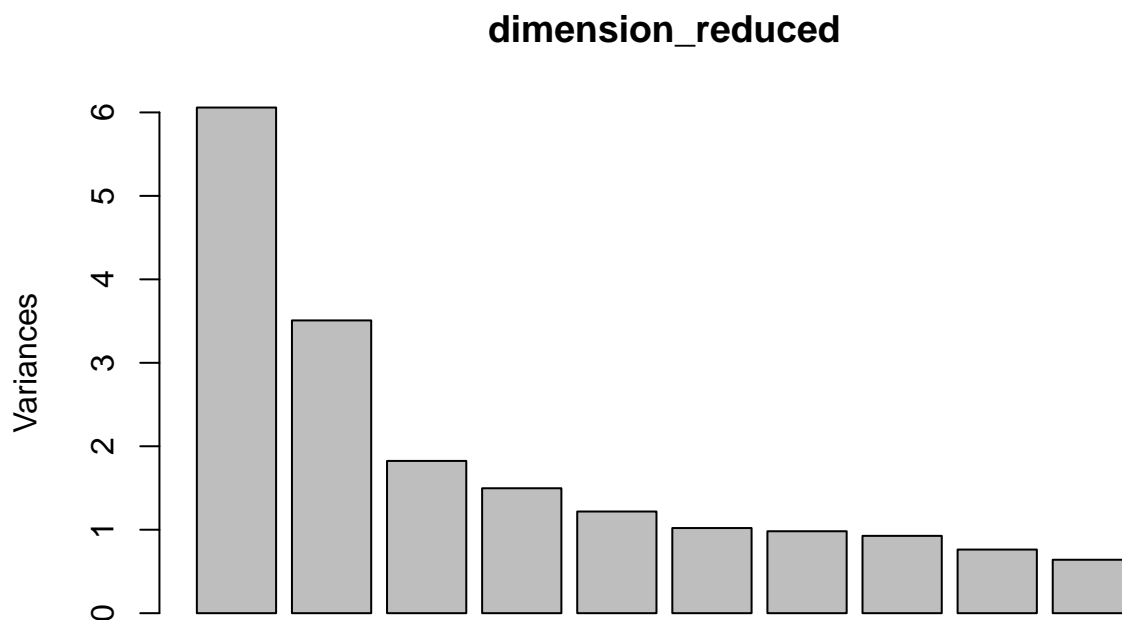
plot_data <- data
colnames(plot_data) <- paste0(seq_along(data))
```

Rudimentary model fitting

From the variance inflation factor analysis above, we can see that there are a few variables that correlate very highly. In particular, we can see that 3x3 block in the lower right of the matrix. These are the mode, mean and median of the histogram, so a high level of multicollinearity is expected. Similarly, the other variables in high association with each other are histogram width, minimum and number of peaks, which we also expect.

For now, we shall leave these terms in our data. Since we are using principal components analysis, the resultant components constructed are orthogonal.

```
dimension_reduced <- prcomp(data[,1:21], scale = TRUE, center = TRUE)
screplot(dimension_reduced)
```



```
sum(dimension_reduced$sdev[1:5]^2)/sum(dimension_reduced$sdev^2)
```

```
## [1] 0.671692
```

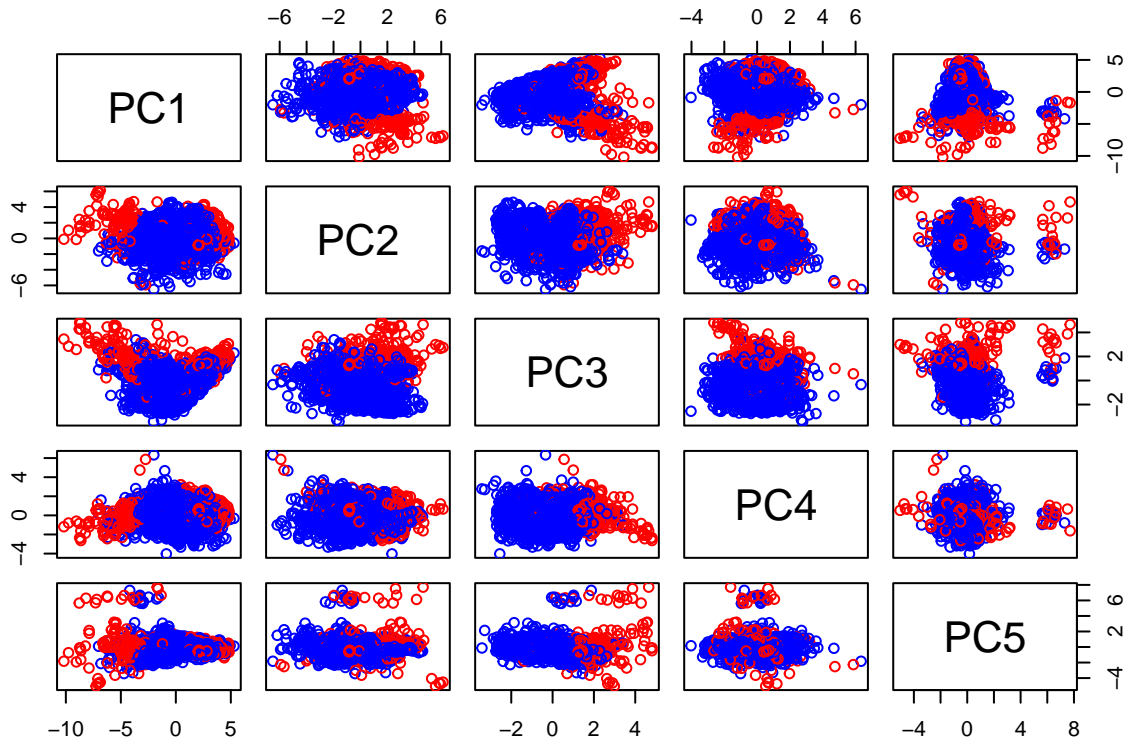
```
round(dimension_reduced$sdev^2 / sum(dimension_reduced$sdev^2), 2)
```

```
## [1] 0.29 0.17 0.09 0.07 0.06 0.05 0.05 0.04 0.04 0.03 0.03 0.02 0.02 0.02 0.01
## [16] 0.01 0.01 0.01 0.00 0.00 0.00
```

Based on the screeplot and proportion of variance explained, using four of five principal components seems to be a viable option. Given this, we can now begin trying to fit a logistic model.

Let us now consider if we can find any meaningful separation using the principal components.

```
colours <- ifelse(data$recoded == 1, 'red', 'blue')
pairs(dimension_reduced$x[,1:5], col=colours)
```



Based on the plot, we can see that there is significant overlap in many of the principal component pairings. However, there do appear to be a few pairings when there is better separation than others, such as all of the PC3 groupings. Additionally, all of the PC5 groupings show a smaller cluster that appears to be a mix of normal and of concern infants. This may be worth further investigation at another time, but unfortunately does not aim in solving our problem.

In our next step we will attempt to fit the logistic model, which will involve dredging over all the possible combinations. I have chosen to use AICc as my dredging criteria. We will first consider if any of our derived variables require transformation.

```
library(mgcv)

derived_data <- as.data.frame(dimension_reduced$x[,1:5])
derived_data$code <- data$recoded

gam.fit <- gam(code ~ s(PC1) + s(PC2) + s(PC3) + s(PC4) + s(PC5), family = "binomial", data = derived_data)

plot(gam.fit)
```

It does not appear like our data requires any transformations, although the first principal component suggests possibly transforming with to a 7th or 8th degree, which I have chosen not to do, as visually it does not appear especially strong. Now we can move on to dredging our model. I intend to avoid any interaction terms with more than two variables, as it makes interpretation of the model more difficult.

```
library(MuMIn)

anova(glm(code ~ PC1 * PC2 * PC3 * PC4 * PC5, data = derived_data, family = binomial), test='Chisq')

full_model <- glm(code ~ PC1 * PC2 + PC1 * PC3 + PC2 * PC3 + PC1 * PC4 + PC1 * PC5 + PC2 * PC5 + PC4 * PC5, data = derived_data, family = binomial)

options(na.action = "na.fail")

dredged_model <- dredge(full_model)

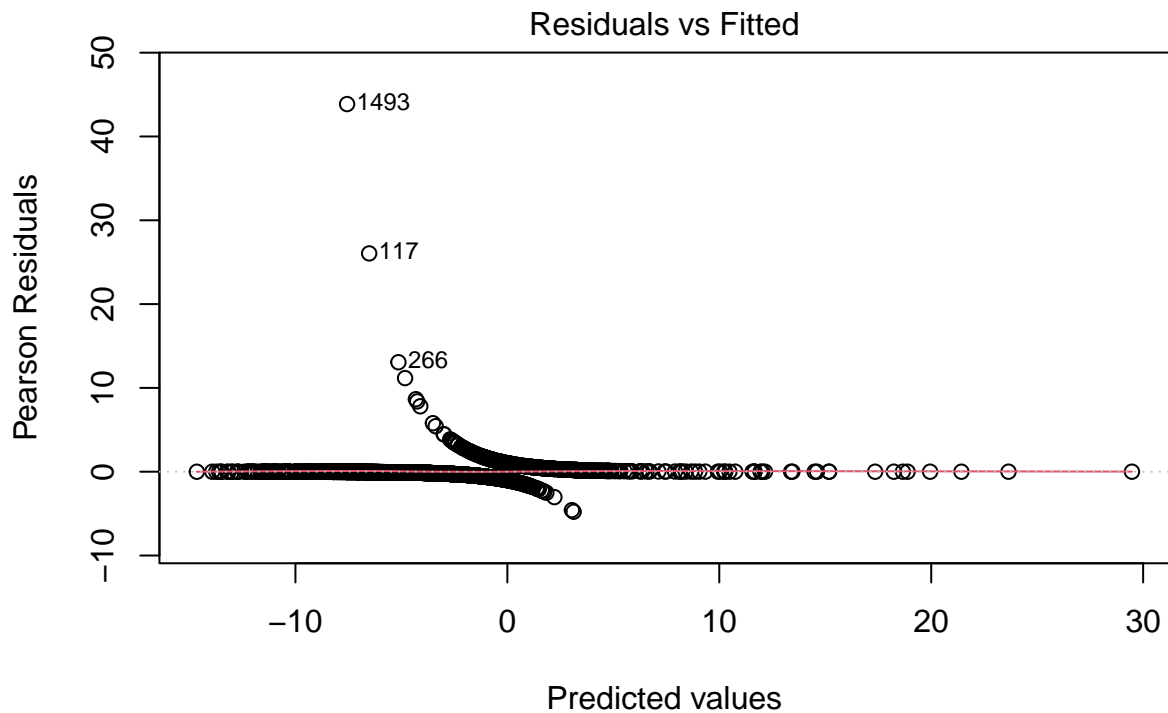
chosen_model <- get.models(dredged_model, 2)[[1]]

summary(chosen_model)
```

I ended up choosing the second model provided by dredge, as it had a comparable AICc to the first model, but the first model also included a non-significant interaction term while the second model does not.

```
deviance <- deviance(chosen_model)
df <- df.residual(chosen_model)

plot(chosen_model, which=1)
```



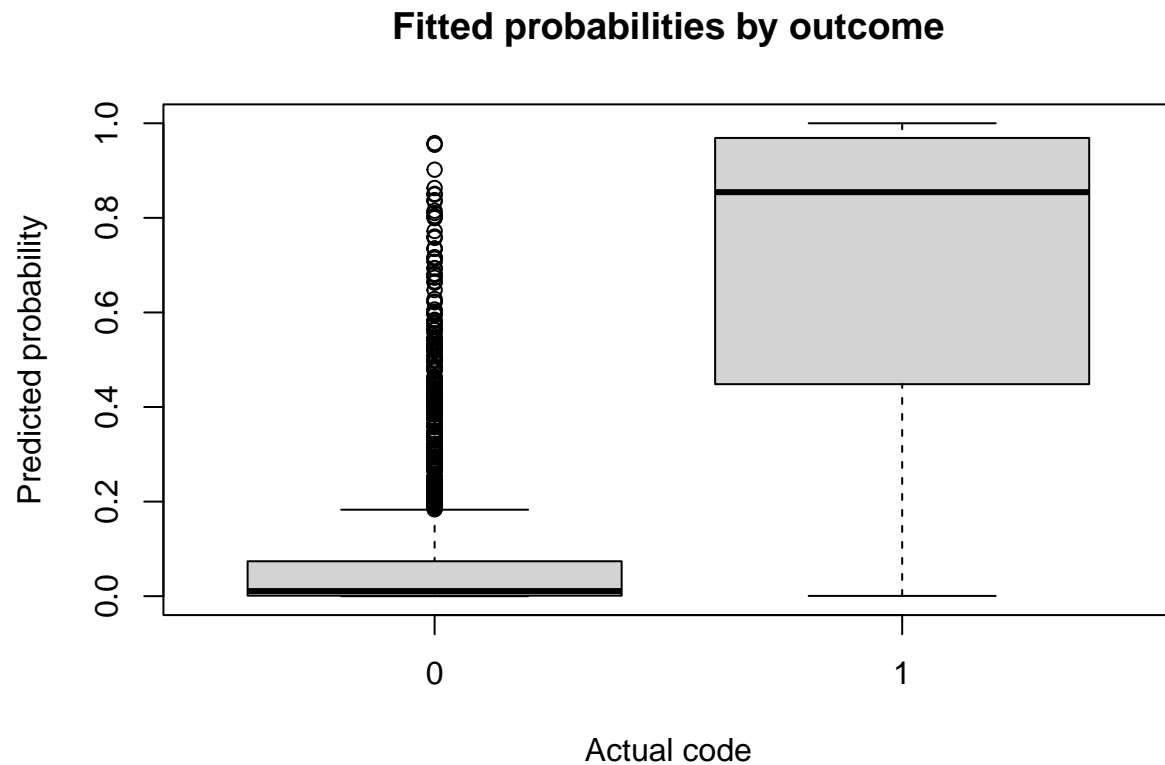
```
glm(code ~ PC1 + PC2 + PC3 + PC4 + PC5 + PC1:PC2 + PC1:PC3 + PC1:PC4 + PC1:PC5, data = derived_data, family = binomial)
```

```
# p-value for goodness-of-fit test
1 - pchisq(deviance, df)
```

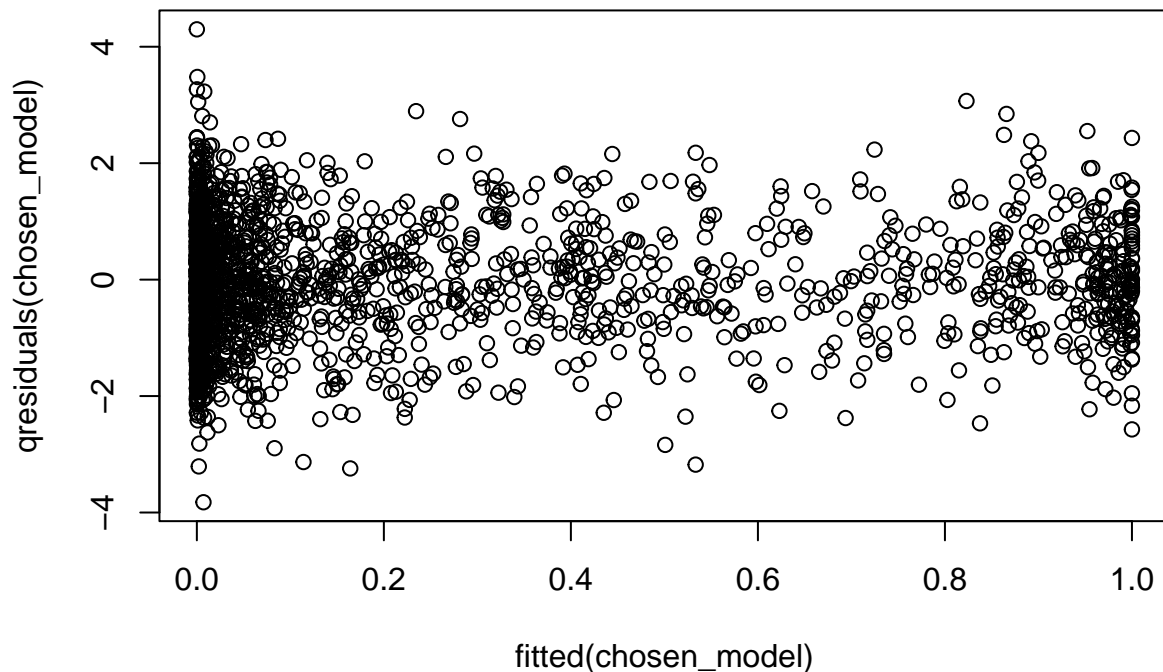
```
## [1] 1
```

Since we are using principal components, this goodness-of-fit test does not tell us much, but we can still check the randomised quantile residuals.

```
boxplot(fitted(chosen_model) ~ derived_data$code,  
        main = "Fitted probabilities by outcome",  
        xlab = "Actual code", ylab = "Predicted probability")
```



```
library(statmod)  
plot(fitted(chosen_model), qresiduals(chosen_model))
```



Based on the outputs here, we can see that our residuals are mostly random noise, and that there is a clear separation between the groups or normal and of concern infants.

Lastly, let us consider our cross-validation to determine how well our model can predict the health of the infants.

```
library(crossval)

data <- as.data.frame(data)

predfun.lm <- function(train.x, train.y, test.x, test.y) {
  glm.fit <- glm(train.y ~ ., data = train.x, family = binomial)
  ynew <- predict(glm.fit, test.x, type = 'response')
  mean((ynew - test.y)^2)
}

y.fit1 <- "code"
cv.out = crossval(predfun.lm, X = derived_data[,c('PC1', 'PC2', 'PC3', 'PC4', 'PC5')], Y = derived_data$
MSPE_1 <- cv.out$stat
MSPE_1se <- cv.out$stat.se

MSPE_1

## [1] 0.07512697
```

```
MSPE_1se
```

```
## [1] 0.003989696
```

So our model fits extremely well. To put our derived variables back into a more interpretable form, let us consider what each principal component may be representing, but investigating their loadings. It is especially of interest to see the loadings of PC3 as it has the greatest estimated coefficient.

```
dimension_reduced$rotation[, 1:5]%*%diag(dimension_reduced$sd[1:5]) -> new_scale
class(new_scale)<-"loadings"
new_scale
```

```
##
## Loadings:
##
##           [,1]  [,2]  [,3]
## baseline value      0.552 -0.519  0.459
## accelerations      -0.107 -0.522 -0.252
## fetal_movement      -0.198         0.137
## uterine_contractions -0.314         -0.163
## light_decelerations  -0.686         0.271
## severe_decelerations -0.142  0.147
## prolonged_decelerations -0.499  0.213  0.455
## abnormal_short_term_variability  0.358  0.229  0.664
## mean_value_of_short_term_variability -0.779 -0.253 -0.101
## percentage_of_time_with_abnormal_long_term_variability  0.534  0.264  0.433
## mean_value_of_long_term_variability         -0.181 -0.584
## histogram_width      -0.772 -0.530
## histogram_min         0.817  0.267
## histogram_max      -0.329 -0.711  0.252
## histogram_number_of_peaks -0.618 -0.449  0.145
## histogram_number_of_zeroes -0.303 -0.238
## histogram_mode         0.646 -0.688
## histogram_mean         0.760 -0.610
## histogram_median        0.679 -0.705
## histogram_variance      -0.715 -0.188  0.359
## histogram_tendency      0.173 -0.423
##
##           [,4]  [,5]
## baseline value      -0.533  0.116
## accelerations      -0.533  0.116
## fetal_movement         0.700
## uterine_contractions -0.364 -0.509
## light_decelerations      -0.391
## severe_decelerations      -0.221
## prolonged_decelerations -0.118  0.312
## abnormal_short_term_variability  0.113
## mean_value_of_short_term_variability
## percentage_of_time_with_abnormal_long_term_variability  0.259
## mean_value_of_long_term_variability  0.463  0.211
## histogram_width      0.129
## histogram_min      -0.346
## histogram_max      -0.290
## histogram_number_of_peaks  0.192  0.109
```

```
## histogram_number_of_zeroes          0.362 -0.224
## histogram_mode
## histogram_mean
## histogram_median
## histogram_variance          -0.117
## histogram_tendency          0.585 -0.126
##
##          [,1] [,2] [,3] [,4] [,5]
## SS loadings   6.058 3.508 1.824 1.497 1.218
## Proportion Var 0.288 0.167 0.087 0.071 0.058
## Cumulative Var 0.288 0.456 0.542 0.614 0.672
```

PC1 - Possibly represents overall fetal heart rate level and distribution. Higher scores tend to correlate with lower variation, fewer decelerations, and higher histogram averages (mean, media, mode. - Could benefit from being mirrored from interpretability. - Loads lowly on mean value of long term variability, *only*.

PC2 - Appears to focus more on histogram shape and overall heart rate dispersion, loading very low on the histogram averages that PC1 loaded highly on. - Could be used as a contrast to PC1.

PC3 - Loads highest on variabilities and movements. - Does not have any especially large loadings.

PC4 - Loads highest on accelerations, histogram tendency and variability. - Does not load highly on decelerations at all. - Hardly loads on the histogram values.

PC5 - Loads very highly on fetal movement, followed by uterine contractions, although the latter is negatively loaded. - Loads against most decelerations except for prolonged decelerations. - Barely loads on any summary statistics, with almost none of the average/max/min/variability variables being loaded on.

Overall it is very difficult to definitively claim what any of these derived variables represent, especially given my limited knowledge of neo-natal care, and cardiology. However, what is clear from this analysis is that we can accurately classify children into of concern and healthy groupings, which may prove beneficial to health professionals. That said, there is a significant issue in the fact that the mathematics used to get to this conclusion has arrived at a fairly complex model based on derived components. Because of this, the it is not simple equation nor rule of thumb that a practitioner may use, and would in all likelihood require a computer with the model pre-loaded, and then require the inputting of all measurements, significantly delaying the process.

However, one useful piece of information revealed in this analysis is that the baseline value is important in the loadings of the first three principal components, accounting for much of the variation, but also that many of the histogram averages are only loaded by the first two principal components, in opposite directions. This suggests that a less complicated model should be possible, perhaps based on these variables, but likely requiring more as well.

I noticed that in the PCA pairs plots, the third principal component seems to do a fairly good job at separation, so I decided to investigate and noticed that the highest loading on this component was abnormal short term viability. We will now fit a simple logistic regression model using just this predictor to see how well it performs alone.

```
abnormal_logistics <- glm(recoded ~ abnormal_short_term_variability, family=binomial, data = data)
summary(abnormal_logistics)
```

```
##
## Call:
## glm(formula = recoded ~ abnormal_short_term_variability, family = binomial,
##      data = data)
##
## Coefficients:
```

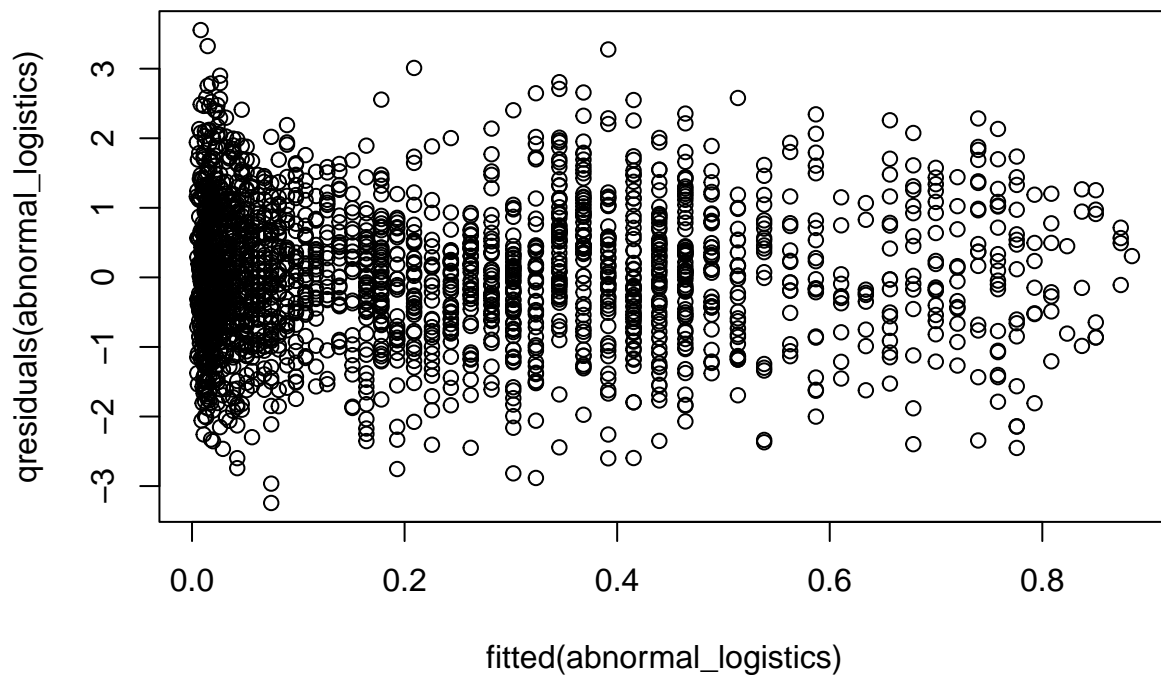


```
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)      -6.577585   0.314157  -20.94  <2e-16 ***
## abnormal_short_term_variability  0.098986   0.005267   18.79  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##    Null deviance: 2248.7  on 2125  degrees of freedom
## Residual deviance: 1640.6  on 2124  degrees of freedom
## AIC: 1644.6
##
## Number of Fisher Scoring iterations: 6
```

```
deviance <- deviance(abnormal_logistics)
df <- df.residual(abnormal_logistics)
1 - pchisq(deviance, df) # fits very well, check the randomised quantile data
```

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

```
plot(fitted(abnormal_logistics), qresiduals(abnormal_logistics)) # mostly white noise
```



The model appears fairly consistent with our checks, and also has a much lower AIC than our principal components based model, so let us now consider it's 10 fold cross validation outputs.

```
x.fit1 <- names(abnormal_logistics$coefficients)[-1]
y.fit1 <- "recoded"
cv.out = crossval(predfun.lm, X = as.data.frame(data[, "abnormal_short_term_variability"]), Y = data[, y
MSPE_1 <- cv.out$stat
MSPE_1se <- cv.out$stat.se

MSPE_1
```

```
## [1] 0.1206224
```

```
MSPE_1se
```

```
## [1] 0.003316772
```

Next steps

- Logging some of the variables
- Removing some of the variables
- Using an ROC

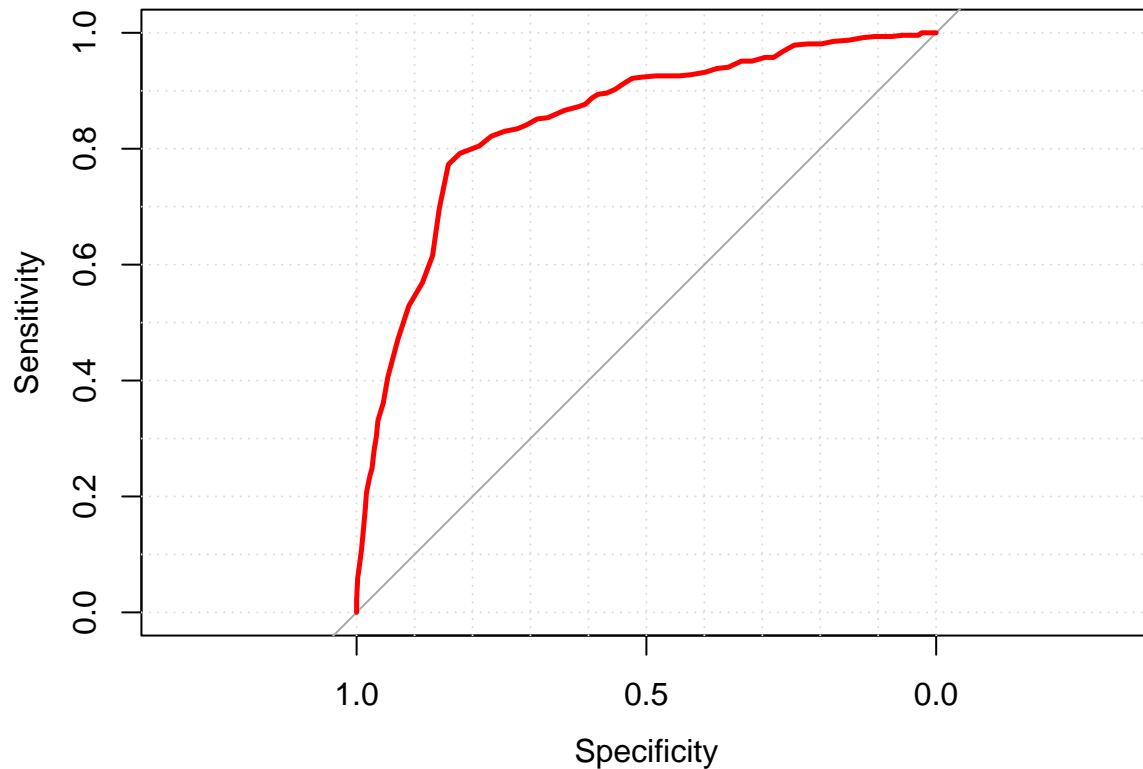
```
library(pROC)

fetal_roc <- roc(response = data$recoded, predictor = fitted.values(abnormal_logistics))

coords(fetal_roc, "best", inut="threshold")

##   threshold specificity sensitivity
## 1 0.3345881    0.8416918    0.7728238

plot(fetal_roc, col="red", grid=TRUE, lwd=2.5)
```



```
auc(fetal_roc)
```

```
## Area under the curve: 0.8477
```

From the further ROC analysis, we can see that at best the second, simpler model can achieve a specificity of 0.84 and sensitivity of 0.77, which is not bad, but a bit low for medical test results.

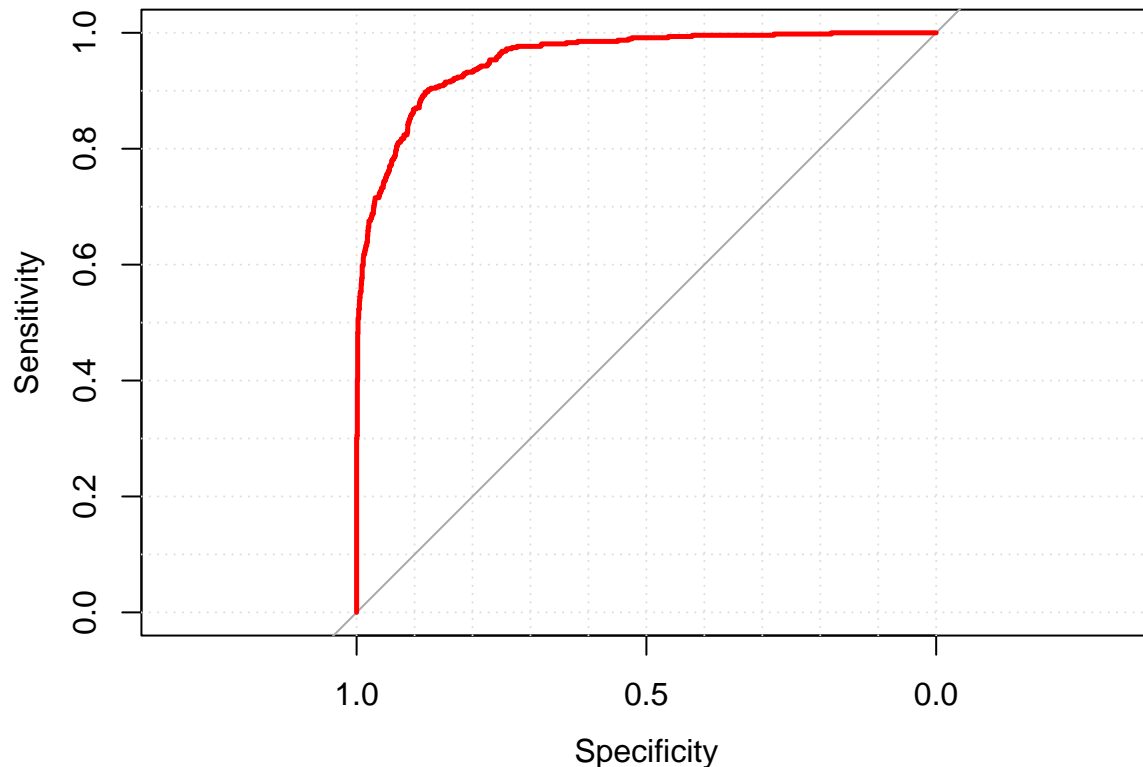
As such, let us now perform the same analysis on our prior, more complicated model.

```
fetal_roc_PC <- roc(response = data$recoded, predictor = fitted.values(chosen_model))
```

```
coords(fetal_roc_PC, "best", input="threshold")
```

```
##   threshold specificity sensitivity
## 1 0.2342552    0.881571    0.8980892
```

```
plot(fetal_roc_PC, col="red", grid=TRUE, lwd=2.5)
```



```
auc(fetal_roc_PC)
```

```
## Area under the curve: 0.9555
```

The second model does a significantly better job discriminating, with both the sensitivity and specificity at similar values, both of which pushing on the edge of 90%. While this is a notable improvement from the simpler model, it would still give false test results a little over 10% of the time.

In conclusion, the simpler model has a significantly lower sensitivity than the complex model, and a slightly lower specificity, which suggests that the more complex second model is more suitable. However, as the second model still under performs compared to the regulator preferences of 95%-99%+ accuracy, so I would not recommend widespread use of either model.

While I would prefer to increase the sensitivity, even at risk of decreasing the specificity, I cannot find an efficient way to do this short of classing every infant as at risk, which would defeat the point of this model entirely.

The only way I can find that would provide a decent attempt at finding this is to trawl through the possible indices for the sensitivities to find one with an accuracy in the desired range.

```
chosen_threshold <- 1100
```

```
fetal_roc_PC$thresholds[1291]
```

```
## [1] 0.08648535
```

```
fetal_roc_PC$sensitivities[chosen_threshold]
```

```
## [1] 0.9808917
```

```
fetal_roc_PC$specificities[chosen_threshold]
```

```
## [1] 0.6610272
```

By essentially guessing and checking I was able to find a threshold that gives a sensitivity of 98% in return for a 66% specificity. this means only 2% of at risk infants would be missed, and that roughly 34% of healthy infants would be class as being at risk.

I believe that this is a suitable trade off, as the risks of an at risk infant not being detected are significantly greater than the risks of further tests or possible treatment.

If we seek to reach the lower bound of this, that being a 95% sensitivity, we find that the specificity increases to 77%. While this is good, as we now only mis-classify 23% of healthy infants as at-risk, an 11% reduction, we have now more than doubled the number of infants we are declaring healthy despite being at risk. Although 5% in total is fairly low, the decision of which threshold to use should be left to medical practitioners.

```
chosen_threshold <- 1291
```

```
fetal_roc_PC$sensitivities[chosen_threshold]
```

```
## [1] 0.9511677
```

```
fetal_roc_PC$specificities[chosen_threshold]
```

```
## [1] 0.7703927
```

Ultimately, it is possible for us to develop a model that will provide a suitable level of accuracy in the identification of at risk infants, with trade offs for the number of false positives we are willing to deal with in exchange for higher successful identification of true positives. The decision regarding the threshold should be left to practitioners, who can decide if the costs, risks, and additional requires of further testing or possible treatment on a healthy infant are worth the higher successful identification of at risk children.

This will also largely depend on the proportion of children we expect to be at risk.

```
round(sum(data$recoded) / nrow(data), 4)
```

```
## [1] 0.2215
```

```
0.2215 * nrow(data)
```

```
## [1] 470.909
```

```
0.23 * (nrow(data) - sum(data$recoded))
```

```
## [1] 380.65
```

As seen, in our cases only 22.15% of the infants in our data were actually at risk, meaning that if we used to 95% model we would have further classed a further 381 infants as at risk when they were actually healthy. However, the majority of cancer tests return false positives, with very few positive mamograms actually leading to breast cancer diagnoses, in exchange for patients receiving peace of mind, and the practitioners knowing they are more likely to catch cancer cases than not.

It would be my naive recommendation to use the 95% sensitive model, as I believe that while successfully classing as many at risk infants as possible correctly, the extra load on the system from testing healthy infants may result in significant delays in an already backlogged health system. As stated many times, the actual decision should ultimately lie with those who best understand our health system, namely the doctors and nurses who conduct much of the testing and treatment.

Finally, I would like to touch on the fact that our model is very complicated. Even though we do have a high level of sensitivity, and a relatively good level of specificity, the fact is that our model is not something that can be applied easily, and will definitely need a computer to be effectively used. As such, I would recommend the automation of test result interpretation by loading results directly into a program with the model pre-loaded.

Another alternative would be to try to develop a model for this data using multivariate techniques such as LDA or QDA. Firstly, there is sufficient data to make these claims, but we need to further investigate the MVN and equal covariance assumptions, which we can do using Kolmogorov-Smirnov and Multivariate Levene's tests respectively.

One advantage to this is that we do not have to collapse the groups into two, so we can work with three groups as per the original data.

```
library(mvabund)
data <- read_csv("fetal_health.csv")
data_matrix <- as.matrix(data[, -22])

dimesion_manova <- manova(data_matrix ~ data$fetal_health)
abs_resids <- abs(dimesion_manova$resid)

maonva_permutations <- manylm(abs_resids ~ data$fetal_health, cor.type = "R", test = "F")
anova(maonva_permutations, resamp = 'perm.resid')
```

```
## Analysis of Variance Table
##
## Model: manylm(formula = abs_resids ~ data$fetal_health, test = "F",
## Model:      cor.type = "R")
##
## Overall test for all response variables
## Test statistics:
##              Res.Df Df.diff val(F) Pr(>F)
## (Intercept)      2125
## data$fetal_health  2124      1  4837  0.002 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Arguments:
## Test statistics calculated assuming unconstrained correlation response
## P-value calculated using 999 iterations via residual (without replacement) resampling.
```

Based on the highly significant P-value output, we can conclude that the null hypothesis of equal covariance must be rejected. Even if this fails, QDA is still possible provided MVN holds, although based on the plots, it does seem unlikely.

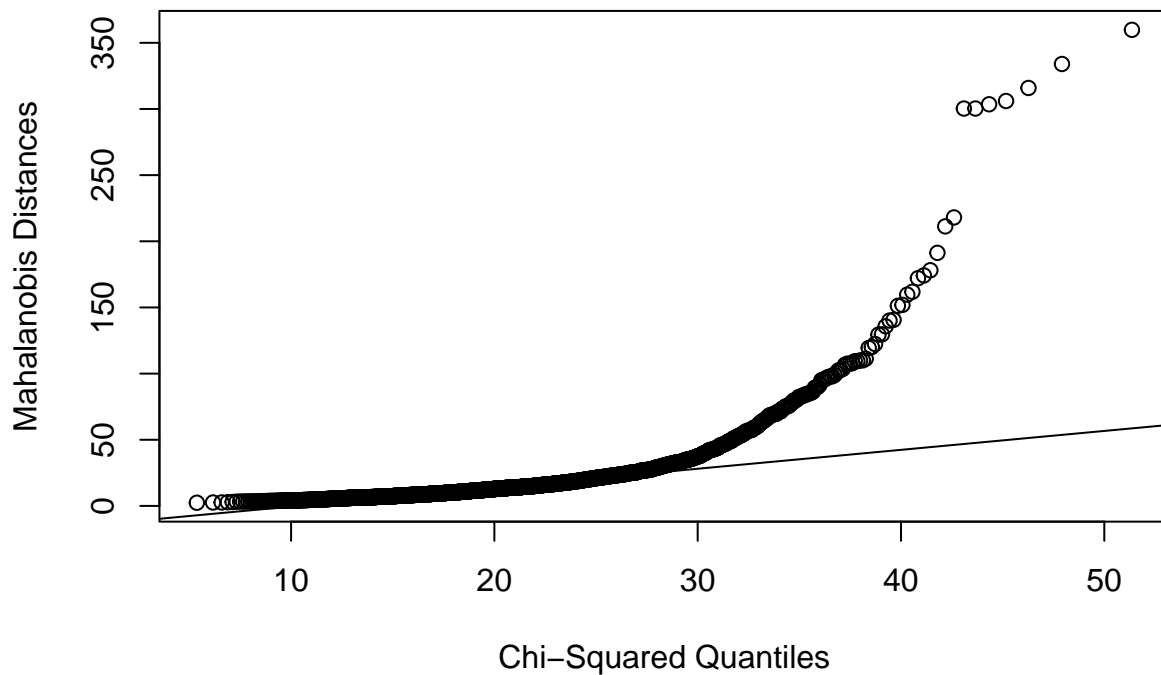
```

resid_mat <- residuals(dimesion_manova)
Sigma <- cov(resid_mat)

maha <- mahalanobis(resid_mat, center = colMeans(resid_mat), cov = Sigma)

qqplot(qchisq(ppoints(nrow(data))), df = ncol(resid_mat)), y = maha,
       xlab = "Chi-Squared Quantiles", ylab = "Mahalanobis Distances")
qqline(maha, distribution = function(p) qchisq(p, df = ncol(resid_mat)))

```



```

ks.test(maha, "pchisq", df = ncol(resid_mat))

```

```

##
## Asymptotic one-sample Kolmogorov-Smirnov test
##
## data: maha
## D = 0.42082, p-value < 2.2e-16
## alternative hypothesis: two-sided

```

So the data is definitely not MVN. This means that neither LDA nor QDA are good choices, but we can use PLS-DA.

```

library(mixOmics)

X <- data_matrix

```

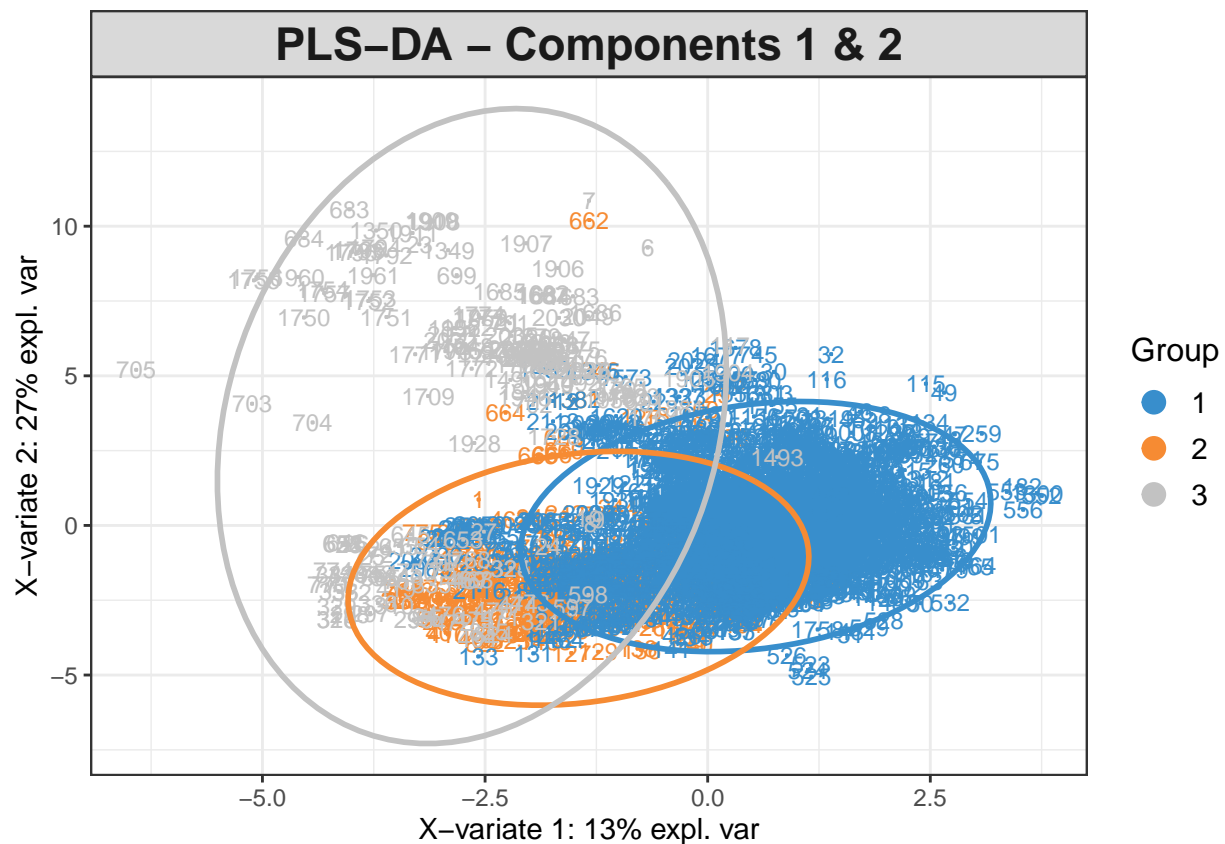
```

Y <- data$fetal_health

plsda_model <- plsda(X, Y, ncomp=2)

plotIndiv(plsda_model,
  comp = c(1, 2),
  group = Y,
  ellipse = TRUE,
  legend = TRUE,
  legend.title = "Group",
  title = "PLS-DA - Components 1 & 2")

```

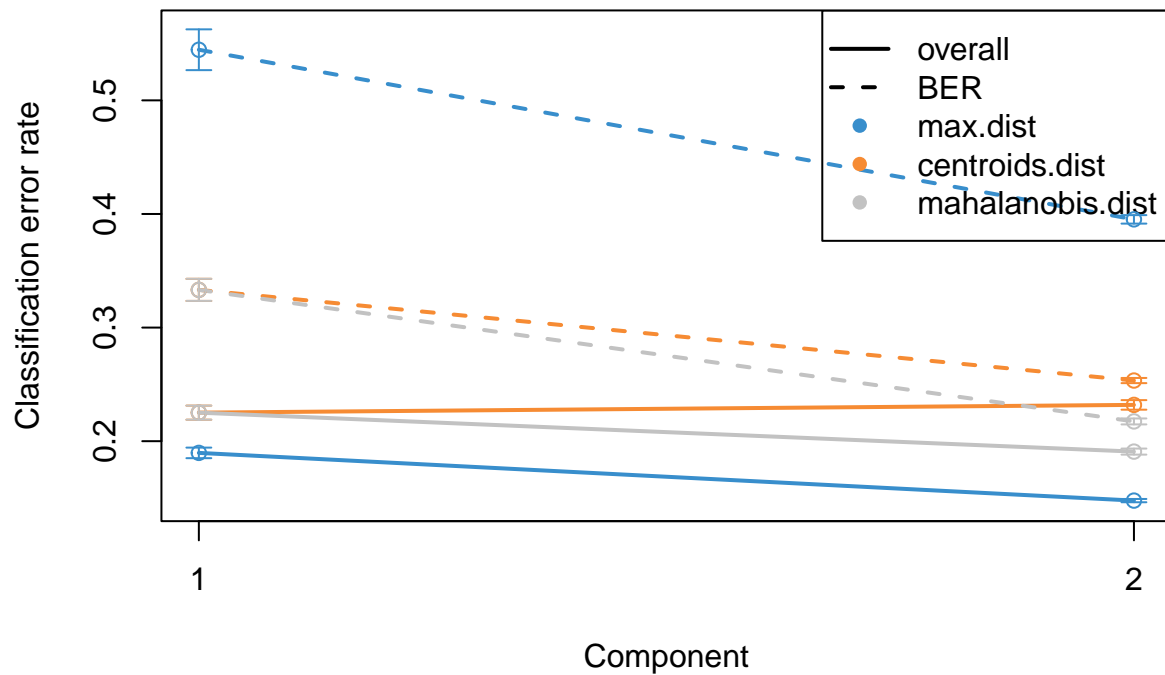


```

set.seed(123)
perf <- perf(plsda_model, validation = "Mfold", folds = 5, nrepeat = 10)

plot(perf, criterion = "error.rate")

```

The results from this are not especially useful, as we can see that the second group does not inhabit an especially unique area, and significantly overlaps with the other groups. Let us now return to the collapsed group problem and repeat this step.

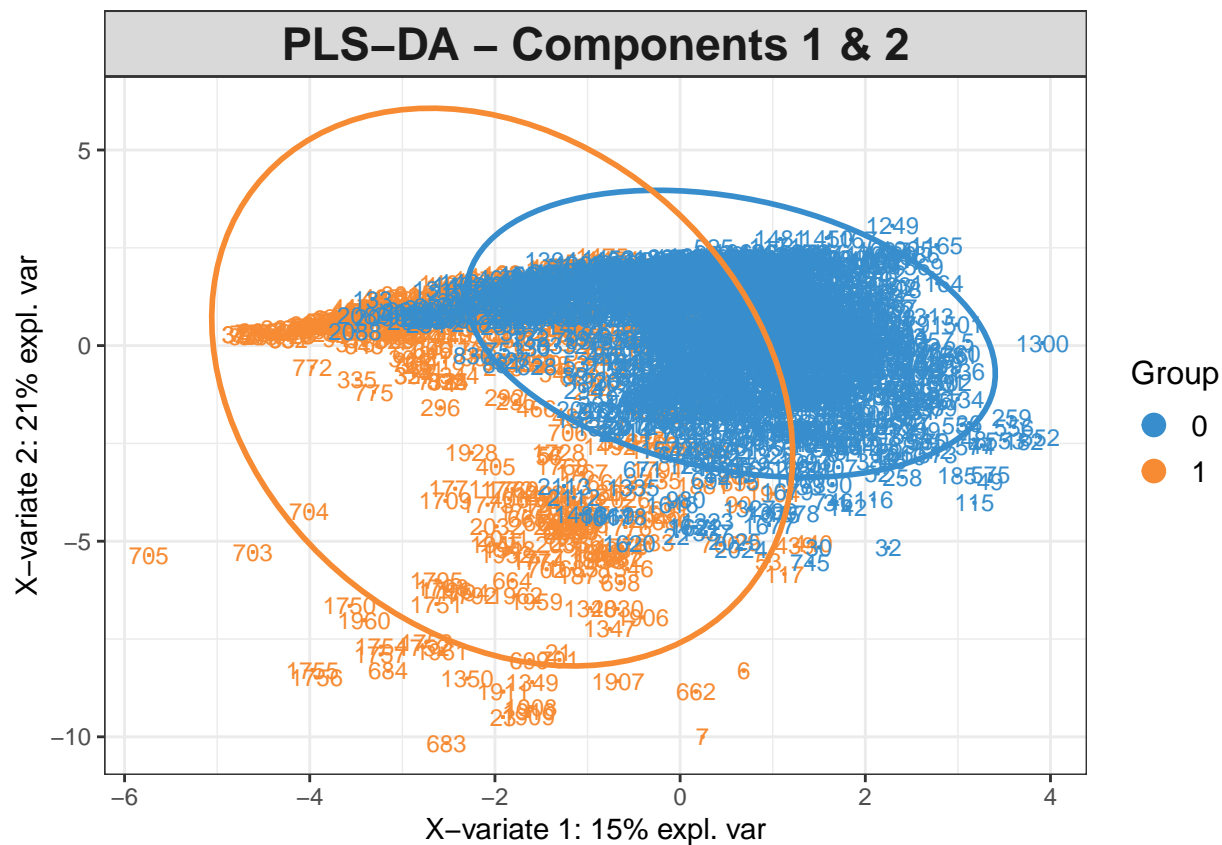
```
data <- data %>%
  mutate(
    recoded = ifelse(fetal_health == 1, 0, 1)
  ) %>%
  dplyr::select(-fetal_health)

data_matrix <- as.matrix(data[, -22])

X <- data_matrix
Y <- data$recoded

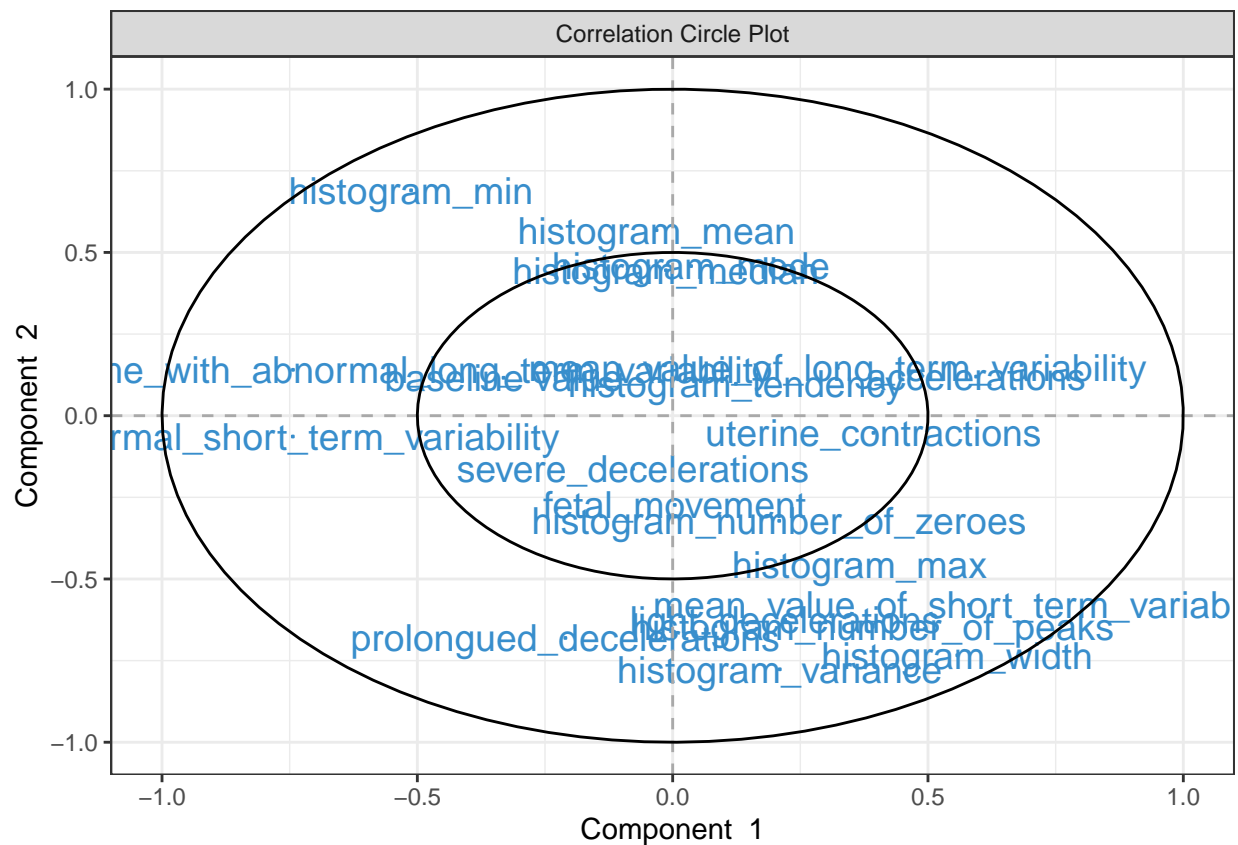
plsda_model <- plsda(X, Y, ncomp=2)

plotIndiv(plsda_model,
  comp = c(1, 2),
  group = Y,
  ellipse = TRUE,
  legend = TRUE,
  legend.title = "Group",
  title = "PLS-DA - Components 1 & 2")
```

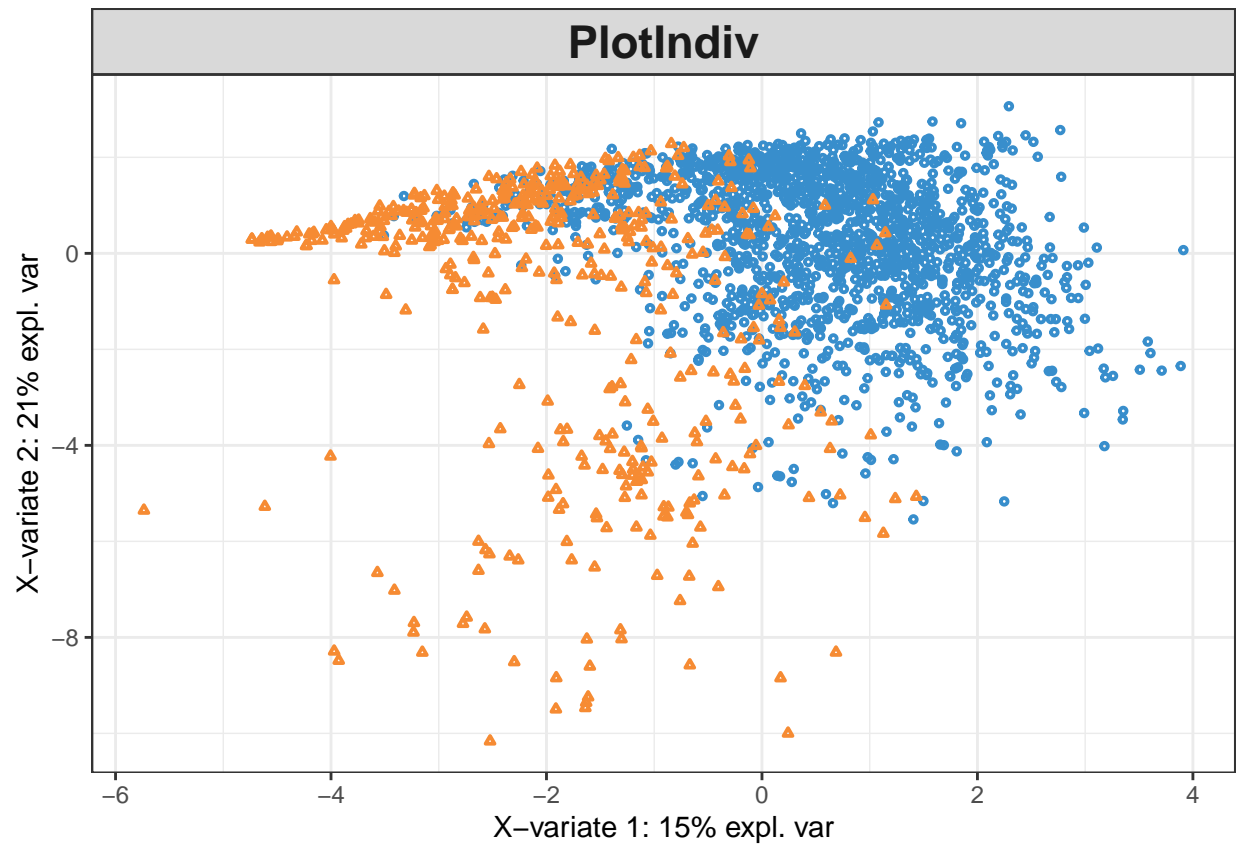


We can see that this solution is actually much worse than the logistic regression solution, as there is significant overlap between these groups. Not only this, but it is clear from both PLS-DA analyses that the proportions of variance explained are far too low in both solutions.

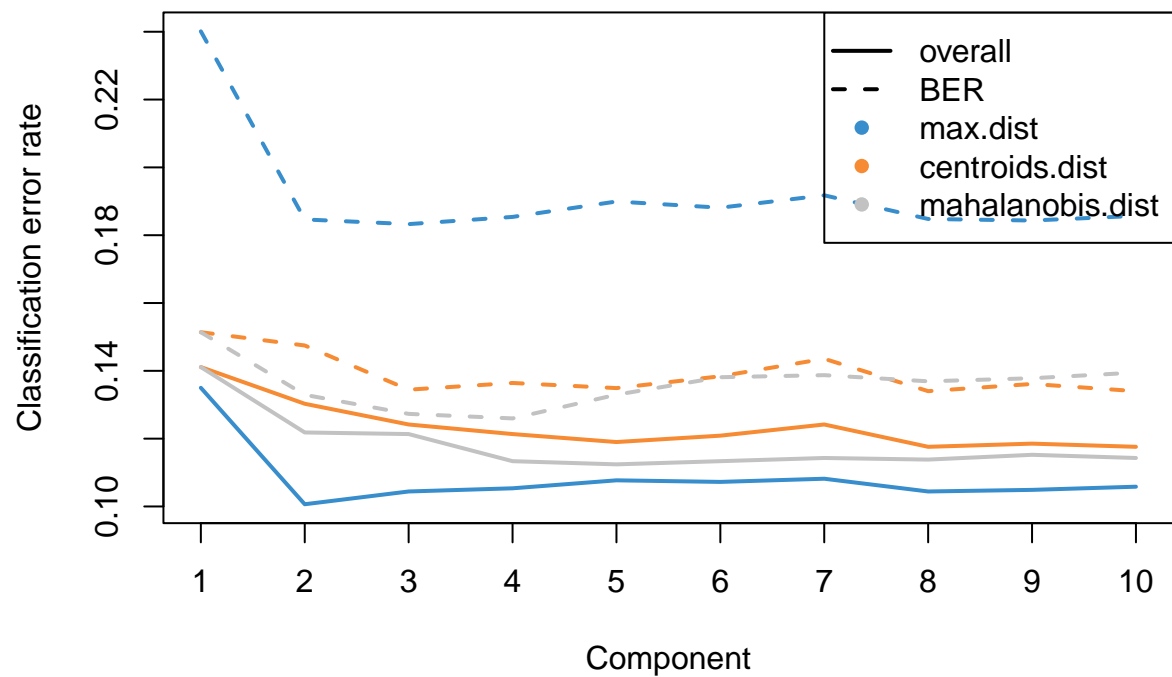
```
plsdout <- plsda(X, Y, ncomp=10)
plotVar(plsdout)
```



```
plotIndiv(plsdaout, comp=1:2, ind.names=FALSE, cex=0.5)
```



```
health.perf <- perf(plsdaout)
plot(health.perf)
```

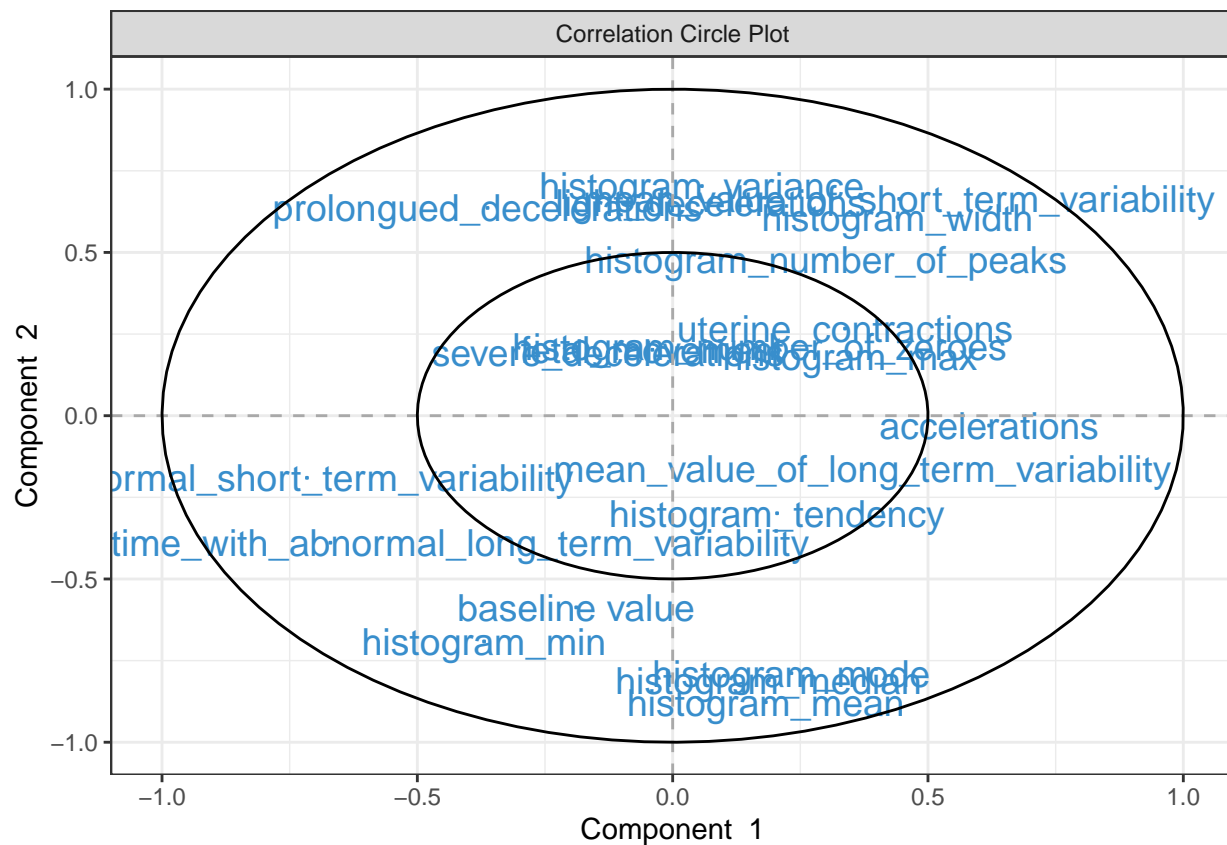


There is fairly good indication that using two components is sufficient.

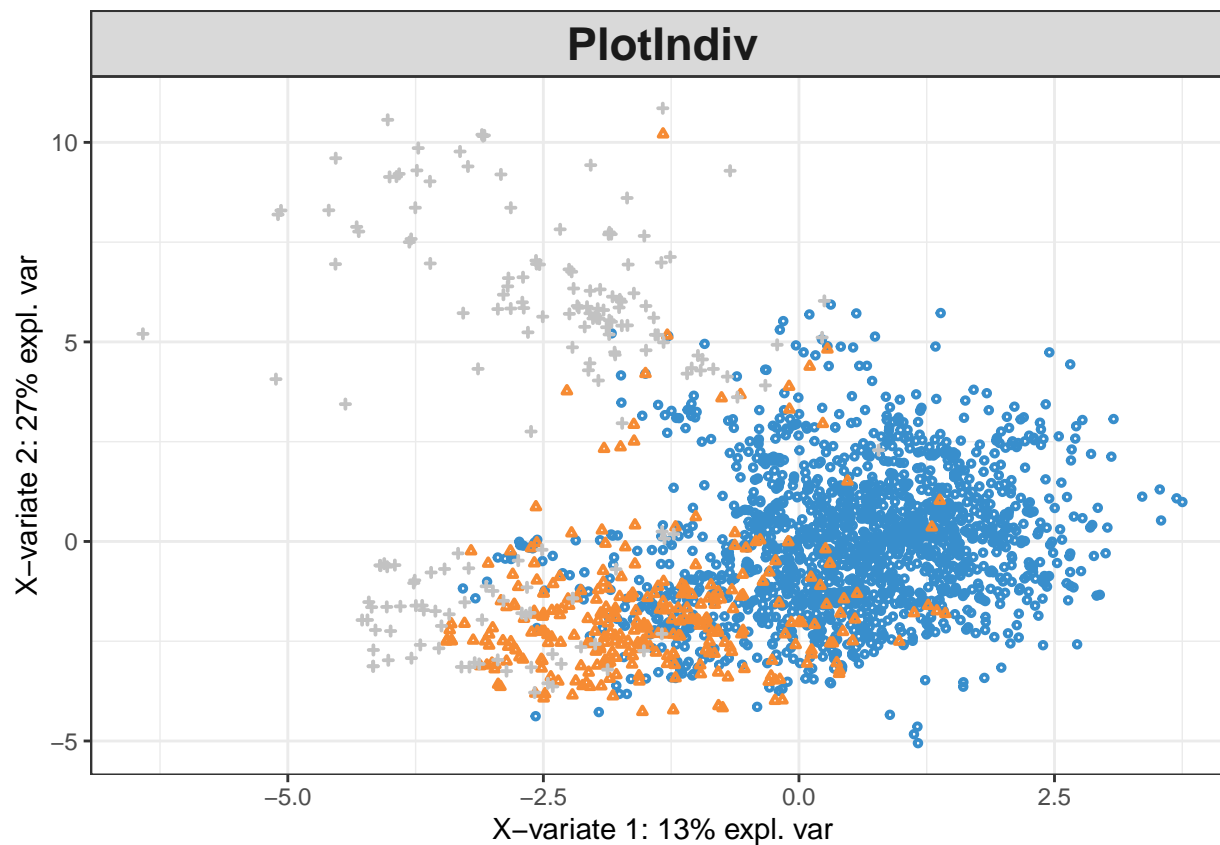
```
data <- read_csv("fetal_health.csv")
data_matrix <- as.matrix(data[, -22])

X <- data_matrix
Y <- data$fetal_health

plsdaout <- plsda(X, Y, ncomp=10)
plotVar(plsdaout)
```



```
plotIndiv(plsdaout, comp=1:2, ind.names=FALSE, cex=0.5)
```

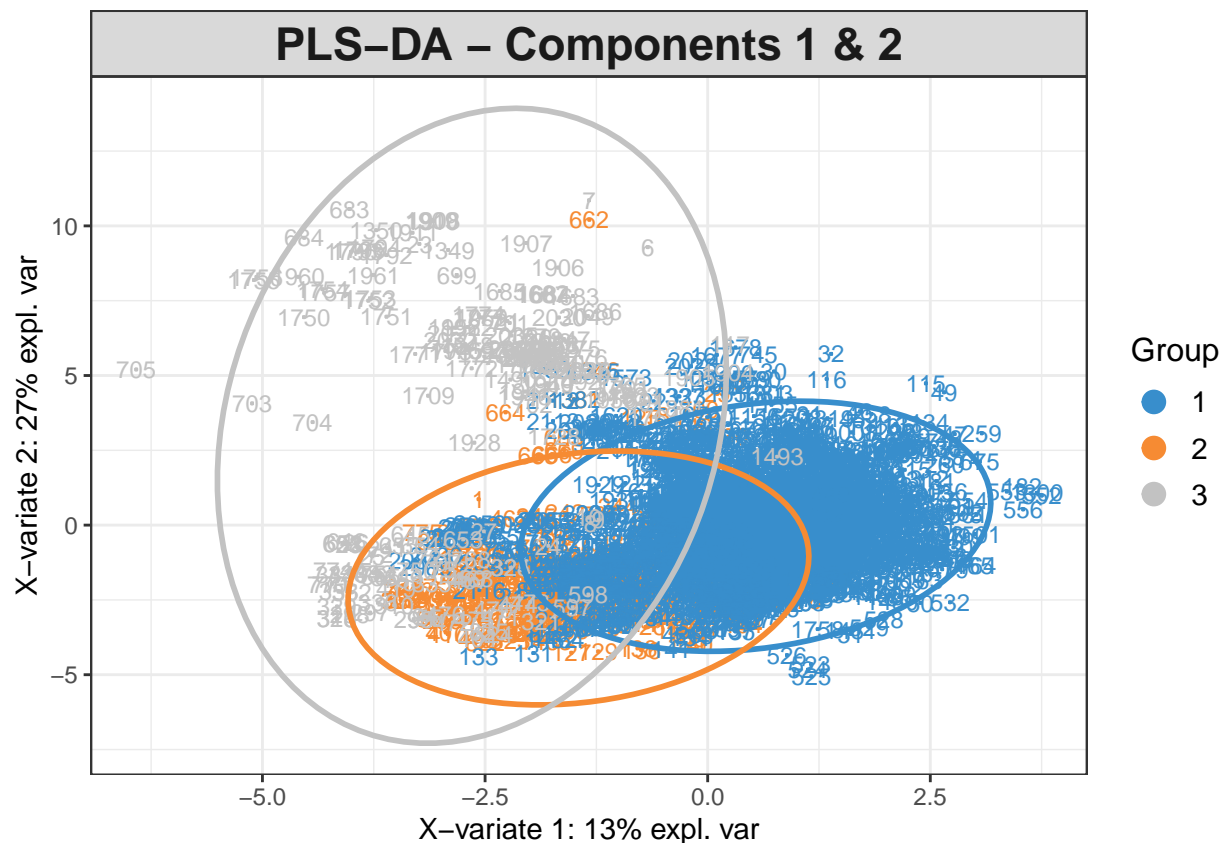


Let's try a cross-validation method using the three class solution, as a model capable of classifying the three groups is more valuable than classing into the collapsed groups.

```
X <- data_matrix
Y <- data$fetal_health

plsda_model <- plsda(X, Y, ncomp=5)

plotIndiv(plsda_model,
  comp = c(1, 2),
  group = Y,
  ellipse = TRUE,
  legend = TRUE,
  legend.title = "Group",
  title = "PLS-DA - Components 1 & 2")
```



```
cv_perf <- perf(plsda_model, validation = "Mfold", folds = 5, nrepeat = 18)
cv_perf$error.rate
```

```
## $overall
##           max.dist centroids.dist mahalanobis.dist
## comp1 0.1884865      0.2279450      0.2279450
## comp2 0.1481133      0.2335894      0.1917790
## comp3 0.1379220      0.2142521      0.1564231
## comp4 0.1378959      0.2054458      0.1483746
## comp5 0.1382617      0.2032508      0.1382617
##
## $BER
##           max.dist centroids.dist mahalanobis.dist
## comp1 0.5366564      0.3331652      0.3331652
## comp2 0.3960886      0.2532972      0.2177843
## comp3 0.3726414      0.2355769      0.2065629
## comp4 0.3723834      0.2304133      0.2170042
## comp5 0.3657994      0.2293186      0.2121886
```

```
error_rates <- cv_perf$error.rate$overall

accuracy <- 1 - error_rates

best_accuracy <- apply(accuracy, 1, max)
best_accuracy
```



```
##      comp1      comp2      comp3      comp4      comp5
## 0.8115135 0.8518867 0.8620780 0.8621041 0.8617383
```

When we use three components, we can achieve an accuracy of 86%, which is the best while balancing for complexity. This is similar to the mutually maximise AUC solution for the logistic regression method, suggesting that this value may truly be the limit in our combined classification ability of all groups.

Finally, let. us see if a Naive Bayes Classifier can do a better job.

```
library(e1071)
library(caTools)
library(caret)

set.seed(8904)
split <- sample.split(data$fetal_health, SplitRatio = 0.8)
training_data <- subset(data, split == TRUE)
testing_data <- subset(data, split == FALSE)

scaled_training <- scale(training_data)
scaled_testing <- scale(testing_data)

nb_classifier <- naiveBayes(fetal_health ~ ., data = training_data)

nb_preds <- predict(nb_classifier, newdata = testing_data)

cm <- table(testing_data$fetal_health, nb_preds)
confusionMatrix(cm)
```

```
## Confusion Matrix and Statistics
##
##      nb_preds
##      1      2      3
## 1 277    33    21
## 2   5    48     6
## 3   0    14    21
##
## Overall Statistics
##
##              Accuracy : 0.8141
##              95% CI : (0.7738, 0.85)
##      No Information Rate : 0.6635
##      P-Value [Acc > NIR] : 3.765e-12
##
##              Kappa : 0.5803
##
## Mcnemar's Test P-Value : 1.005e-09
##
## Statistics by Class:
##
##              Class: 1 Class: 2 Class: 3
## Sensitivity          0.9823    0.5053    0.43750
## Specificity          0.6224    0.9667    0.96286
## Pos Pred Value       0.8369    0.8136    0.60000
## Neg Pred Value       0.9468    0.8716    0.93077
```

## Prevalence	0.6635	0.2235	0.11294
## Detection Rate	0.6518	0.1129	0.04941
## Detection Prevalence	0.7788	0.1388	0.08235
## Balanced Accuracy	0.8023	0.7360	0.70018

The Naive Bayes Classifier solution achieved 83.29% accuracy, which is a slight downgrade compared to the PLS-DA and Logistic Regression solutions, suggesting that one of those methods may be a better choice. In particular, we can see that the Naive Bayes Classifier mis-identifies many of the pathological cases as being at-risk, and two as normal.