MAST90138 Multivariate Statistics for Data Science Assignment3

Yini Lai (Student ID: 1127650)

18/10/2020

```
library(knitr)
library(tidyverse)
library(MASS)
library(pls)
library(randomForest)
```

```
# Loading data
# Training
Raintrain <- read.table("XGtrainRain.txt", sep = ",", header = TRUE)
# Test
Raintest <- read.table("XGtestRain.txt", sep = ",", header = TRUE)
```

Problem 1

(a)

```
# QDA
# fitqda <- qda(G ~., data = Raintrain)

# Logistic
fitlogistic <- glm(G ~., data = Raintrain, family=binomial(link = "logit"))
summary(fitlogistic)
```

```
##
## Call:
## glm(formula = G ~ ., family = binomial(link = "logit"), data = Raintrain)
##
## Deviance Residuals:
##    [1]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##   [26]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##   [51]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##   [76]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##  [101]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##  [126]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##
## Coefficients: (216 not defined because of singularities)
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -7.849e+01  2.197e+06       0        1
## X1          -6.096e+00  6.929e+05       0        1
## X2          -3.648e+00  2.508e+05       0        1
## X3           7.099e+00  7.048e+05       0        1
## X4           2.206e+00  6.562e+05       0        1
```

```
## X5          -1.762e+00  2.616e+05       0        1
## X6          -6.876e+00  5.976e+05       0        1
## X7          -7.968e+00  5.748e+05       0        1
## X8           1.400e+00  2.706e+05       0        1
## X9          -8.330e+00  8.410e+05       0        1
## X10          3.033e+00  6.766e+05       0        1
## X11          3.041e+00  1.267e+05       0        1
## X12         -1.171e+00  6.845e+04       0        1
## X13          7.694e+00  4.191e+05       0        1
## X14          1.208e+00  2.703e+05       0        1
## X15         -5.311e-01  1.023e+05       0        1
## X16          5.480e-01  3.312e+05       0        1
## X17         -5.274e+00  7.846e+05       0        1
## X18         -2.870e+00  2.189e+05       0        1
## X19         -3.808e+00  4.476e+05       0        1
## X20          2.175e+00  1.423e+05       0        1
## X21          7.505e-01  9.251e+04       0        1
## X22         -1.518e+00  3.275e+05       0        1
## X23          4.828e+00  5.118e+05       0        1
## X24         -3.696e+00  1.467e+05       0        1
## X25          4.100e+00  2.623e+05       0        1
## X26          2.522e+00  1.347e+05       0        1
## X27          4.579e+00  4.698e+05       0        1
## X28          1.881e+00  6.870e+04       0        1
## X29          1.528e+00  1.493e+05       0        1
## X30         -3.299e+00  5.277e+05       0        1
## X31          8.721e-01  4.400e+04       0        1
## X32          1.003e+00  3.187e+05       0        1
## X33         -7.368e+00  5.158e+05       0        1
## X34          5.462e+00  1.279e+05       0        1
## X35          1.186e+00  1.831e+05       0        1
## X36          3.455e+00  1.632e+05       0        1
## X37         -1.095e+00  6.362e+04       0        1
## X38         -1.550e+00  1.321e+05       0        1
## X39          4.118e+00  7.676e+04       0        1
## X40         -7.575e-01  1.527e+05       0        1
## X41          1.277e+00  1.951e+05       0        1
## X42         -4.701e-01  1.092e+05       0        1
## X43          6.747e+00  5.202e+05       0        1
## X44         -6.909e-01  7.319e+04       0        1
## X45          2.016e+00  1.311e+05       0        1
## X46          4.356e+00  5.016e+05       0        1
## X47         -6.055e+00  3.809e+05       0        1
## X48          4.080e+00  4.671e+05       0        1
## X49          2.443e+00  6.632e+04       0        1
## X50          2.778e-01  1.856e+05       0        1
## X51         -3.830e+00  2.867e+05       0        1
## X52          1.257e+00  1.944e+05       0        1
## X53         -7.575e+00  6.747e+05       0        1
## X54          8.986e-01  9.623e+04       0        1
## X55          3.864e+00  4.235e+05       0        1
## X56          1.977e+00  1.409e+05       0        1
## X57          2.835e+00  2.869e+05       0        1
## X58         -2.642e-01  6.229e+04       0        1
```

```
## X59        -2.418e+00  3.463e+05       0        1
## X60        -2.652e+00  6.309e+04       0        1
## X61         1.366e+00  3.139e+05       0        1
## X62        -3.345e+00  5.384e+05       0        1
## X63        -4.317e+00  1.669e+05       0        1
## X64         1.519e-01  6.396e+04       0        1
## X65        -4.366e-01  1.028e+05       0        1
## X66        -7.615e-01  6.079e+04       0        1
## X67        -5.395e-01  8.733e+04       0        1
## X68         8.354e+00  7.134e+05       0        1
## X69        -2.723e+00  1.263e+05       0        1
## X70        -3.518e+00  2.954e+05       0        1
## X71        -5.808e+00  4.465e+05       0        1
## X72         2.748e+00  3.045e+05       0        1
## X73         2.008e+00  2.290e+05       0        1
## X74         1.136e+00  1.258e+05       0        1
## X75        -4.882e+00  4.379e+05       0        1
## X76         1.616e+00  1.053e+05       0        1
## X77         3.854e+00  1.211e+05       0        1
## X78        -3.404e+00  6.276e+05       0        1
## X79         9.164e-01  3.116e+05       0        1
## X80         4.525e-01  9.043e+04       0        1
## X81         1.719e+00  4.453e+05       0        1
## X82         8.778e+00  5.176e+05       0        1
## X83         5.079e-01  3.677e+05       0        1
## X84         4.751e+00  1.182e+05       0        1
## X85        -7.464e+00  5.297e+05       0        1
## X86        -2.200e+00  8.595e+04       0        1
## X87         2.359e-01  4.743e+04       0        1
## X88        -4.010e+00  8.058e+05       0        1
## X89         5.425e+00  9.621e+05       0        1
## X90         5.431e+00  2.830e+05       0        1
## X91        -5.671e+00  3.993e+05       0        1
## X92         1.852e+00  1.126e+05       0        1
## X93        -1.576e+00  4.614e+05       0        1
## X94        -8.014e-01  1.682e+05       0        1
## X95        -2.458e+00  4.991e+04       0        1
## X96        -7.706e+00  1.091e+06       0        1
## X97        -3.625e-01  7.686e+04       0        1
## X98         1.363e-01  1.930e+05       0        1
## X99         3.712e-01  1.591e+05       0        1
## X100        2.426e+00  4.670e+05       0        1
## X101        1.054e+00  1.211e+05       0        1
## X102        2.362e-01  1.808e+05       0        1
## X103       -3.146e+00  2.238e+05       0        1
## X104        3.563e+00  1.892e+05       0        1
## X105        5.218e+00  9.163e+05       0        1
## X106        1.305e-01  1.344e+05       0        1
## X107        1.777e+00  6.172e+05       0        1
## X108       -5.086e+00  2.973e+05       0        1
## X109       -2.121e+00  2.066e+05       0        1
## X110       -2.891e-02  1.214e+05       0        1
## X111       -9.743e-01  5.597e+05       0        1
## X112       -4.487e+00  3.274e+05       0        1
```

```
## X113        2.762e-01  1.548e+05       0        1
## X114       -3.022e+00  2.773e+05       0        1
## X115        3.785e+00  2.811e+05       0        1
## X116       -3.325e+00  1.521e+05       0        1
## X117        7.086e+00  3.443e+05       0        1
## X118       -4.193e+00  3.482e+05       0        1
## X119       -2.470e+00  1.385e+05       0        1
## X120        6.300e+00  4.805e+05       0        1
## X121       -6.150e+00  5.085e+05       0        1
## X122        9.146e+00  6.043e+05       0        1
## X123        7.002e+00  7.849e+05       0        1
## X124        3.169e+00  5.663e+05       0        1
## X125       -2.574e+00  3.034e+05       0        1
## X126       -3.395e+00  1.470e+05       0        1
## X127        8.306e-01  5.153e+05       0        1
## X128       -2.256e+00  8.019e+04       0        1
## X129        4.167e+00  5.999e+04       0        1
## X130       -3.785e+00  4.720e+05       0        1
## X131        1.991e+00  1.106e+05       0        1
## X132        8.520e+00  7.433e+05       0        1
## X133       -4.428e-01  1.718e+05       0        1
## X134       -1.349e+00  3.550e+05       0        1
## X135       -5.280e+00  7.164e+05       0        1
## X136       -1.110e+00  2.370e+05       0        1
## X137        4.482e+00  2.243e+05       0        1
## X138        1.721e+00  2.091e+05       0        1
## X139       -4.150e+00  3.586e+05       0        1
## X140        2.402e+00  2.109e+05       0        1
## X141       -6.919e+00  7.945e+05       0        1
## X142        1.274e+00  3.791e+05       0        1
## X143        1.815e-01  5.508e+05       0        1
## X144       -1.385e+00  1.480e+05       0        1
## X145       -3.773e+00  3.576e+05       0        1
## X146        4.886e+00  4.146e+05       0        1
## X147       -5.839e+00  7.687e+05       0        1
## X148        5.586e+00  8.637e+05       0        1
## X149        3.002e+00  4.822e+05       0        1
## X150               NA         NA      NA       NA
## X151               NA         NA      NA       NA
## X152               NA         NA      NA       NA
## X153               NA         NA      NA       NA
## X154               NA         NA      NA       NA
## X155               NA         NA      NA       NA
## X156               NA         NA      NA       NA
## X157               NA         NA      NA       NA
## X158               NA         NA      NA       NA
## X159               NA         NA      NA       NA
## X160               NA         NA      NA       NA
## X161               NA         NA      NA       NA
## X162               NA         NA      NA       NA
## X163               NA         NA      NA       NA
## X164               NA         NA      NA       NA
## X165               NA         NA      NA       NA
## X166               NA         NA      NA       NA
```

```
## X167                NA         NA       NA       NA
## X168                NA         NA       NA       NA
## X169                NA         NA       NA       NA
## X170                NA         NA       NA       NA
## X171                NA         NA       NA       NA
## X172                NA         NA       NA       NA
## X173                NA         NA       NA       NA
## X174                NA         NA       NA       NA
## X175                NA         NA       NA       NA
## X176                NA         NA       NA       NA
## X177                NA         NA       NA       NA
## X178                NA         NA       NA       NA
## X179                NA         NA       NA       NA
## X180                NA         NA       NA       NA
## X181                NA         NA       NA       NA
## X182                NA         NA       NA       NA
## X183                NA         NA       NA       NA
## X184                NA         NA       NA       NA
## X185                NA         NA       NA       NA
## X186                NA         NA       NA       NA
## X187                NA         NA       NA       NA
## X188                NA         NA       NA       NA
## X189                NA         NA       NA       NA
## X190                NA         NA       NA       NA
## X191                NA         NA       NA       NA
## X192                NA         NA       NA       NA
## X193                NA         NA       NA       NA
## X194                NA         NA       NA       NA
## X195                NA         NA       NA       NA
## X196                NA         NA       NA       NA
## X197                NA         NA       NA       NA
## X198                NA         NA       NA       NA
## X199                NA         NA       NA       NA
## X200                NA         NA       NA       NA
## X201                NA         NA       NA       NA
## X202                NA         NA       NA       NA
## X203                NA         NA       NA       NA
## X204                NA         NA       NA       NA
## X205                NA         NA       NA       NA
## X206                NA         NA       NA       NA
## X207                NA         NA       NA       NA
## X208                NA         NA       NA       NA
## X209                NA         NA       NA       NA
## X210                NA         NA       NA       NA
## X211                NA         NA       NA       NA
## X212                NA         NA       NA       NA
## X213                NA         NA       NA       NA
## X214                NA         NA       NA       NA
## X215                NA         NA       NA       NA
## X216                NA         NA       NA       NA
## X217                NA         NA       NA       NA
## X218                NA         NA       NA       NA
## X219                NA         NA       NA       NA
## X220                NA         NA       NA       NA
```

```
## X221                NA         NA       NA       NA
## X222                NA         NA       NA       NA
## X223                NA         NA       NA       NA
## X224                NA         NA       NA       NA
## X225                NA         NA       NA       NA
## X226                NA         NA       NA       NA
## X227                NA         NA       NA       NA
## X228                NA         NA       NA       NA
## X229                NA         NA       NA       NA
## X230                NA         NA       NA       NA
## X231                NA         NA       NA       NA
## X232                NA         NA       NA       NA
## X233                NA         NA       NA       NA
## X234                NA         NA       NA       NA
## X235                NA         NA       NA       NA
## X236                NA         NA       NA       NA
## X237                NA         NA       NA       NA
## X238                NA         NA       NA       NA
## X239                NA         NA       NA       NA
## X240                NA         NA       NA       NA
## X241                NA         NA       NA       NA
## X242                NA         NA       NA       NA
## X243                NA         NA       NA       NA
## X244                NA         NA       NA       NA
## X245                NA         NA       NA       NA
## X246                NA         NA       NA       NA
## X247                NA         NA       NA       NA
## X248                NA         NA       NA       NA
## X249                NA         NA       NA       NA
## X250                NA         NA       NA       NA
## X251                NA         NA       NA       NA
## X252                NA         NA       NA       NA
## X253                NA         NA       NA       NA
## X254                NA         NA       NA       NA
## X255                NA         NA       NA       NA
## X256                NA         NA       NA       NA
## X257                NA         NA       NA       NA
## X258                NA         NA       NA       NA
## X259                NA         NA       NA       NA
## X260                NA         NA       NA       NA
## X261                NA         NA       NA       NA
## X262                NA         NA       NA       NA
## X263                NA         NA       NA       NA
## X264                NA         NA       NA       NA
## X265                NA         NA       NA       NA
## X266                NA         NA       NA       NA
## X267                NA         NA       NA       NA
## X268                NA         NA       NA       NA
## X269                NA         NA       NA       NA
## X270                NA         NA       NA       NA
## X271                NA         NA       NA       NA
## X272                NA         NA       NA       NA
## X273                NA         NA       NA       NA
## X274                NA         NA       NA       NA
```

```
## X275                NA         NA       NA       NA
## X276                NA         NA       NA       NA
## X277                NA         NA       NA       NA
## X278                NA         NA       NA       NA
## X279                NA         NA       NA       NA
## X280                NA         NA       NA       NA
## X281                NA         NA       NA       NA
## X282                NA         NA       NA       NA
## X283                NA         NA       NA       NA
## X284                NA         NA       NA       NA
## X285                NA         NA       NA       NA
## X286                NA         NA       NA       NA
## X287                NA         NA       NA       NA
## X288                NA         NA       NA       NA
## X289                NA         NA       NA       NA
## X290                NA         NA       NA       NA
## X291                NA         NA       NA       NA
## X292                NA         NA       NA       NA
## X293                NA         NA       NA       NA
## X294                NA         NA       NA       NA
## X295                NA         NA       NA       NA
## X296                NA         NA       NA       NA
## X297                NA         NA       NA       NA
## X298                NA         NA       NA       NA
## X299                NA         NA       NA       NA
## X300                NA         NA       NA       NA
## X301                NA         NA       NA       NA
## X302                NA         NA       NA       NA
## X303                NA         NA       NA       NA
## X304                NA         NA       NA       NA
## X305                NA         NA       NA       NA
## X306                NA         NA       NA       NA
## X307                NA         NA       NA       NA
## X308                NA         NA       NA       NA
## X309                NA         NA       NA       NA
## X310                NA         NA       NA       NA
## X311                NA         NA       NA       NA
## X312                NA         NA       NA       NA
## X313                NA         NA       NA       NA
## X314                NA         NA       NA       NA
## X315                NA         NA       NA       NA
## X316                NA         NA       NA       NA
## X317                NA         NA       NA       NA
## X318                NA         NA       NA       NA
## X319                NA         NA       NA       NA
## X320                NA         NA       NA       NA
## X321                NA         NA       NA       NA
## X322                NA         NA       NA       NA
## X323                NA         NA       NA       NA
## X324                NA         NA       NA       NA
## X325                NA         NA       NA       NA
## X326                NA         NA       NA       NA
## X327                NA         NA       NA       NA
## X328                NA         NA       NA       NA
```

```
## X329                NA         NA      NA       NA
## X330                NA         NA      NA       NA
## X331                NA         NA      NA       NA
## X332                NA         NA      NA       NA
## X333                NA         NA      NA       NA
## X334                NA         NA      NA       NA
## X335                NA         NA      NA       NA
## X336                NA         NA      NA       NA
## X337                NA         NA      NA       NA
## X338                NA         NA      NA       NA
## X339                NA         NA      NA       NA
## X340                NA         NA      NA       NA
## X341                NA         NA      NA       NA
## X342                NA         NA      NA       NA
## X343                NA         NA      NA       NA
## X344                NA         NA      NA       NA
## X345                NA         NA      NA       NA
## X346                NA         NA      NA       NA
## X347                NA         NA      NA       NA
## X348                NA         NA      NA       NA
## X349                NA         NA      NA       NA
## X350                NA         NA      NA       NA
## X351                NA         NA      NA       NA
## X352                NA         NA      NA       NA
## X353                NA         NA      NA       NA
## X354                NA         NA      NA       NA
## X355                NA         NA      NA       NA
## X356                NA         NA      NA       NA
## X357                NA         NA      NA       NA
## X358                NA         NA      NA       NA
## X359                NA         NA      NA       NA
## X360                NA         NA      NA       NA
## X361                NA         NA      NA       NA
## X362                NA         NA      NA       NA
## X363                NA         NA      NA       NA
## X364                NA         NA      NA       NA
## X365                NA         NA      NA       NA
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2.0017e+02  on 149  degrees of freedom
## Residual deviance: 8.7024e-10  on   0  degrees of freedom
## AIC: 300
##
## Number of Fisher Scoring iterations: 25
```

```
# Source code of QDA function
#qda.default <-
#  function(x, grouping, prior = proportions,
#           method = c("moment", "mle", "mve", "t"),
#          CV = FALSE, nu = 5, ...)
#{
#    if(is.null(dim(x))) stop("'x' is not a matrix")
#    x <- as.matrix(x)
```

```
#     if(any(!is.finite(x)))
#         stop("infinite, NA or NaN values in 'x'")
#     n <- nrow(x)
#     p <- ncol(x)
#     if(n != length(grouping))
#         stop("nrow(x) and length(grouping) are different")
#     g <- as.factor(grouping)
#     lev <- levels(g)
#     counts <- as.vector(table(g))
#     names(counts) <- lev
#     if(any(counts < p+1)) stop("some group is too small for 'qda'")
```

For QDA: Error in qda.default(x, grouping, ...) : some group is too small for 'qda': This may be because we have 365 predictors with only 150 observations in the dataset. For QDA, it calculates the parameters for each level of the response variable separately. In this case, level 0 has 58 observations and level 1 has 92 observations. With 365 predictors, these observations are not enough for parameters estimation. Thus this kind of error occurs. This may also be verify by looking at the source code of QDA function (last line of the source code showing above).

For Logistics Regression: Coefficients: (216 not defined because of singularities): This is because we have 365 variables with only 150 observations. In this case, the design matrix is singular which results in some of the coefficients to be zero.

$\Pr(>|z|) = 1$: This may mean that those variables are not significant in predicting the response variable. This may be because we have a large set of predictors and these predictors are time series which means there maybe some serial correlation/autocorrelation between them. Therefore, including all of these variables may make them not important in predicting the response.

In addition, we can see that the deviance Residuals are zero. This may also because our the amount of predictors are way more than the number of observations. In this case, the fitted model may be perfectly fit all the data points in the training dataset. It is a case of overfitting. If we apply this model to the test data set, it may result in a super large variance.

In general, I wouldn't apply these two classifiers to the test set.

Logistics Regression: Since we include all the predictors to the model, and the number of predictors are way greater than the number of observations when we train the classifier, it is a overfitting model and will cause large variance when applying to the test set.

QDA: Also because of the number of observations is much smaller than the number of predictors, the error occurs when we train the model.

(b)

```
XCtrain <- scale(Raintrain[,-366], scale = FALSE) # Centralize design matrix
PCX <- prcomp(Raintrain[,-366], retx = TRUE)
gamma <- PCX$rotation # Projection matrix/Loading matrix
PCAtrain <- PCX$x
PCAcomp <- XCtrain%*%gamma
(PCAtrain - PCAcomp) %>% head(3)
```

```
##      PC1 PC2 PC3 PC4 PC5 PC6 PC7 PC8 PC9 PC10 PC11 PC12 PC13 PC14 PC15 PC16
## [1,]   0   0   0   0   0   0   0   0   0    0    0    0    0    0    0    0
## [2,]   0   0   0   0   0   0   0   0   0    0    0    0    0    0    0    0
## [3,]   0   0   0   0   0   0   0   0   0    0    0    0    0    0    0    0
```

```
##      PC17 PC18 PC19 PC20 PC21 PC22 PC23 PC24 PC25 PC26 PC27 PC28 PC29 PC30 PC31
## [1,]    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## [2,]    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## [3,]    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
##      PC32 PC33 PC34 PC35 PC36 PC37 PC38 PC39 PC40 PC41 PC42 PC43 PC44 PC45 PC46
## [1,]    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## [2,]    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## [3,]    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
##      PC47 PC48 PC49 PC50 PC51 PC52 PC53 PC54 PC55 PC56 PC57 PC58 PC59 PC60 PC61
## [1,]    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## [2,]    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## [3,]    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
##      PC62 PC63 PC64 PC65 PC66 PC67 PC68 PC69 PC70 PC71 PC72 PC73 PC74 PC75 PC76
## [1,]    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## [2,]    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## [3,]    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
##      PC77 PC78 PC79 PC80 PC81 PC82 PC83 PC84 PC85 PC86 PC87 PC88 PC89 PC90 PC91
## [1,]    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## [2,]    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## [3,]    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
##      PC92 PC93 PC94 PC95 PC96 PC97 PC98 PC99 PC100 PC101 PC102 PC103 PC104
## [1,]    0    0    0    0    0    0    0    0     0     0     0     0     0
## [2,]    0    0    0    0    0    0    0    0     0     0     0     0     0
## [3,]    0    0    0    0    0    0    0    0     0     0     0     0     0
##      PC105 PC106 PC107 PC108 PC109 PC110 PC111 PC112 PC113 PC114 PC115 PC116
## [1,]     0     0     0     0     0     0     0     0     0     0     0     0
## [2,]     0     0     0     0     0     0     0     0     0     0     0     0
## [3,]     0     0     0     0     0     0     0     0     0     0     0     0
##      PC117 PC118 PC119 PC120 PC121 PC122 PC123 PC124 PC125 PC126 PC127 PC128
## [1,]     0     0     0     0     0     0     0     0     0     0     0     0
## [2,]     0     0     0     0     0     0     0     0     0     0     0     0
## [3,]     0     0     0     0     0     0     0     0     0     0     0     0
##      PC129 PC130 PC131 PC132 PC133 PC134 PC135 PC136 PC137 PC138 PC139 PC140
## [1,]     0     0     0     0     0     0     0     0     0     0     0     0
## [2,]     0     0     0     0     0     0     0     0     0     0     0     0
## [3,]     0     0     0     0     0     0     0     0     0     0     0     0
##      PC141 PC142 PC143 PC144 PC145 PC146 PC147 PC148 PC149 PC150
## [1,]     0     0     0     0     0     0     0     0     0     0
## [2,]     0     0     0     0     0     0     0     0     0     0
## [3,]     0     0     0     0     0     0     0     0     0     0
```

The result show that the difference between the PCA obtained from the function and the manually computed PCA are 0 which means they are exactly the same.

```
PLX <- plsr(G ~., data = Raintrain)
# PLS components
PLStrain <- PLX$scores
PLScomp <- as.matrix(scale(Raintrain[,-366], center = T, scale = F))%*%PLX$projection
(PLStrain - PLScomp) %>% head(3)
```

```
##   Comp 1 Comp 2 Comp 3 Comp 4 Comp 5 Comp 6 Comp 7 Comp 8 Comp 9 Comp 10
## 1      0      0      0      0      0      0      0      0      0       0
## 2      0      0      0      0      0      0      0      0      0       0
## 3      0      0      0      0      0      0      0      0      0       0
```

```
##   Comp 11 Comp 12 Comp 13 Comp 14 Comp 15 Comp 16 Comp 17 Comp 18 Comp 19
## 1       0       0       0       0       0       0       0       0       0
## 2       0       0       0       0       0       0       0       0       0
## 3       0       0       0       0       0       0       0       0       0
##   Comp 20 Comp 21 Comp 22 Comp 23 Comp 24 Comp 25 Comp 26 Comp 27 Comp 28
## 1       0       0       0       0       0       0       0       0       0
## 2       0       0       0       0       0       0       0       0       0
## 3       0       0       0       0       0       0       0       0       0
##   Comp 29 Comp 30 Comp 31 Comp 32 Comp 33 Comp 34 Comp 35 Comp 36 Comp 37
## 1       0       0       0       0       0       0       0       0       0
## 2       0       0       0       0       0       0       0       0       0
## 3       0       0       0       0       0       0       0       0       0
##   Comp 38 Comp 39 Comp 40 Comp 41 Comp 42 Comp 43 Comp 44 Comp 45 Comp 46
## 1       0       0       0       0       0       0       0       0       0
## 2       0       0       0       0       0       0       0       0       0
## 3       0       0       0       0       0       0       0       0       0
##   Comp 47 Comp 48 Comp 49 Comp 50 Comp 51 Comp 52 Comp 53 Comp 54 Comp 55
## 1       0       0       0       0       0       0       0       0       0
## 2       0       0       0       0       0       0       0       0       0
## 3       0       0       0       0       0       0       0       0       0
##   Comp 56 Comp 57 Comp 58 Comp 59 Comp 60 Comp 61 Comp 62 Comp 63 Comp 64
## 1       0       0       0       0       0       0       0       0       0
## 2       0       0       0       0       0       0       0       0       0
## 3       0       0       0       0       0       0       0       0       0
##   Comp 65 Comp 66 Comp 67 Comp 68 Comp 69 Comp 70 Comp 71 Comp 72 Comp 73
## 1       0       0       0       0       0       0       0       0       0
## 2       0       0       0       0       0       0       0       0       0
## 3       0       0       0       0       0       0       0       0       0
##   Comp 74 Comp 75 Comp 76 Comp 77 Comp 78 Comp 79 Comp 80 Comp 81 Comp 82
## 1       0       0       0       0       0       0       0       0       0
## 2       0       0       0       0       0       0       0       0       0
## 3       0       0       0       0       0       0       0       0       0
##   Comp 83 Comp 84 Comp 85 Comp 86 Comp 87 Comp 88 Comp 89 Comp 90 Comp 91
## 1       0       0       0       0       0       0       0       0       0
## 2       0       0       0       0       0       0       0       0       0
## 3       0       0       0       0       0       0       0       0       0
##   Comp 92 Comp 93 Comp 94 Comp 95 Comp 96 Comp 97 Comp 98 Comp 99 Comp 100
## 1       0       0       0       0       0       0       0       0         0
## 2       0       0       0       0       0       0       0       0         0
## 3       0       0       0       0       0       0       0       0         0
##   Comp 101 Comp 102 Comp 103 Comp 104 Comp 105 Comp 106 Comp 107 Comp 108
## 1        0        0        0        0        0        0        0        0
## 2        0        0        0        0        0        0        0        0
## 3        0        0        0        0        0        0        0        0
##   Comp 109 Comp 110 Comp 111 Comp 112 Comp 113 Comp 114 Comp 115 Comp 116
## 1        0        0        0        0        0        0        0        0
## 2        0        0        0        0        0        0        0        0
## 3        0        0        0        0        0        0        0        0
##   Comp 117 Comp 118 Comp 119 Comp 120 Comp 121 Comp 122 Comp 123 Comp 124
## 1        0        0        0        0        0        0        0        0
## 2        0        0        0        0        0        0        0        0
## 3        0        0        0        0        0        0        0        0
##   Comp 125 Comp 126 Comp 127 Comp 128 Comp 129 Comp 130 Comp 131 Comp 132
## 1        0        0        0        0        0        0        0        0
```

```
## 2        0        0        0        0        0        0        0        0
## 3        0        0        0        0        0        0        0        0
##   Comp 133 Comp 134 Comp 135 Comp 136 Comp 137 Comp 138 Comp 139 Comp 140
## 1        0        0        0        0        0        0        0        0
## 2        0        0        0        0        0        0        0        0
## 3        0        0        0        0        0        0        0        0
##   Comp 141 Comp 142 Comp 143 Comp 144 Comp 145 Comp 146 Comp 147 Comp 148
## 1        0        0        0        0        0        0        0        0
## 2        0        0        0        0        0        0        0        0
## 3        0        0        0        0        0        0        0        0
##   Comp 149
## 1        0
## 2        0
## 3        0
```

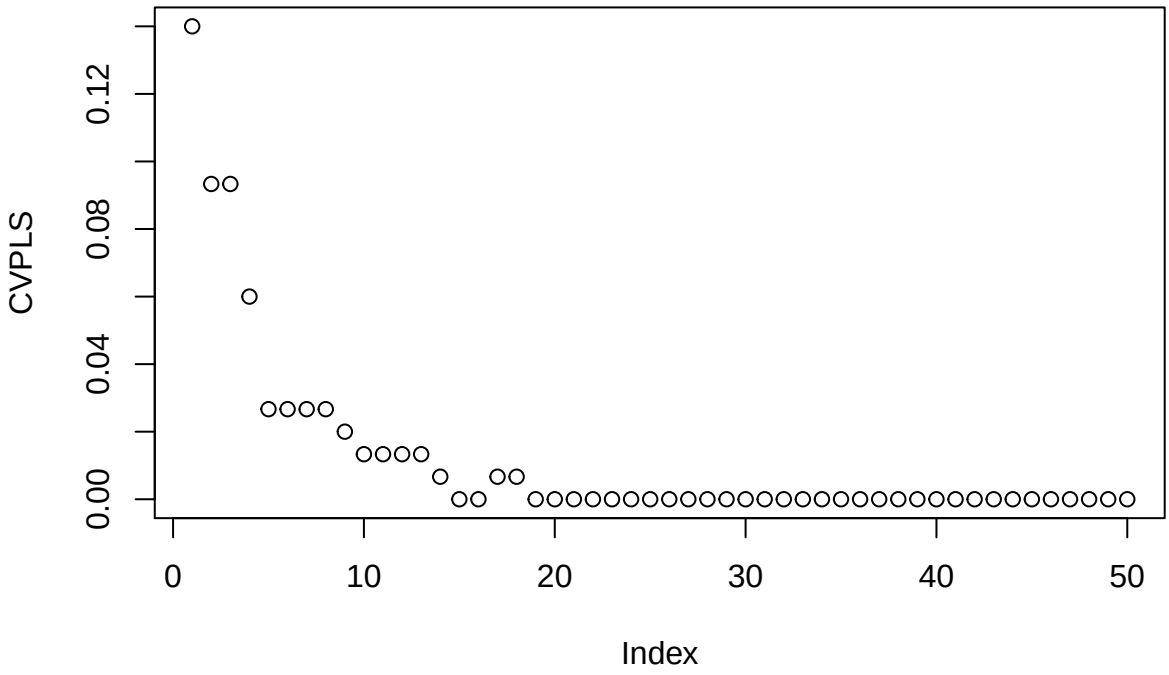
The result show that the difference between the PLS components obtained from the function and the manually computed PLS components are 0 which means they are exactly the same.

(c)

```r
# PLS set
dtPLS <- data.frame(cbind(Raintrain[,366], PLStrain)) %>% rename(G = V1)

# PCA set
dtPCA <- data.frame(cbind(Raintrain[,366], PCAtrain)) %>% rename(G = V1)
```

```r
# PLS with qda
n <- dim(Raintrain)[1]
CVPLS = rep(0,50)

for (k in 1:50) {
  for (i in 1:n) {
    fitPLS <- qda(G ~., data = dtPLS[-i, 1: (k + 1)])
    qdaclass <- predict(fitPLS, dtPLS[i, 1: (k + 1)])$class
    # Cumulative number of prediction error
    CVPLS[k] <- CVPLS[k] + sum(qdaclass != dtPLS[i, 1])
  }
  CVPLS[k] <- CVPLS[k]/n
}
```

```r
plot(CVPLS, main = "QDA:No. of PLS Compnents vs CV error")
```




```
which(CVPLS == min(CVPLS))
```

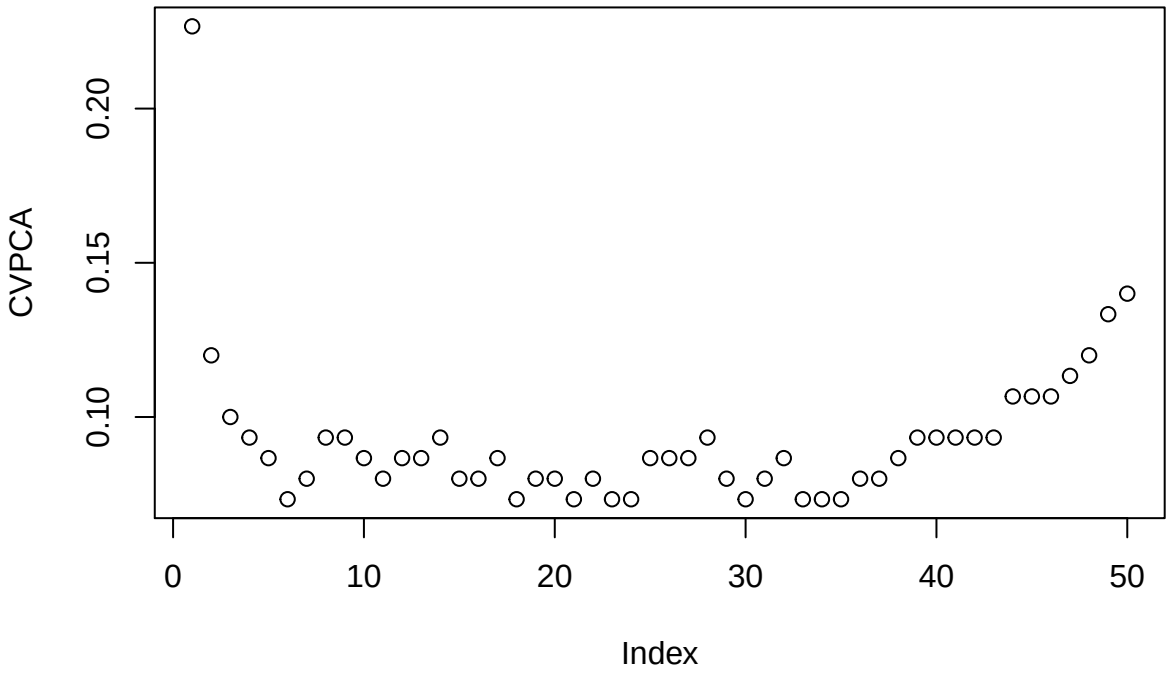
```
##  [1] 15 16 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41
## [26] 42 43 44 45 46 47 48 49 50
```

```
minQDAPLS <- 15
```

For QDA classifier with the PLS components, from the result we can see that the minimum amount of components we need to include in order to make the classification error as small as possible is 15.

```
# PCA with qda
n <- dim(Raintrain)[1]
CVPCA = rep(0,50)

for (k in 1:50) {
  for (i in 1:n) {
    fitPCA <- qda(G ~., data = dtPCA[-i, 1: (k + 1)])
    qdaclass <- predict(fitPCA, dtPCA[i, 1: (k + 1)])$class
    CVPCA[k] <- CVPCA[k] + sum(qdaclass != dtPCA[i, 1])
  }
  CVPCA[k] <- CVPCA[k]/n
}
```

```
plot(CVPCA, main = "QDA:No. of PCA Compnents vs CV error")
```




```
which(CVPCA == min(CVPCA))
```

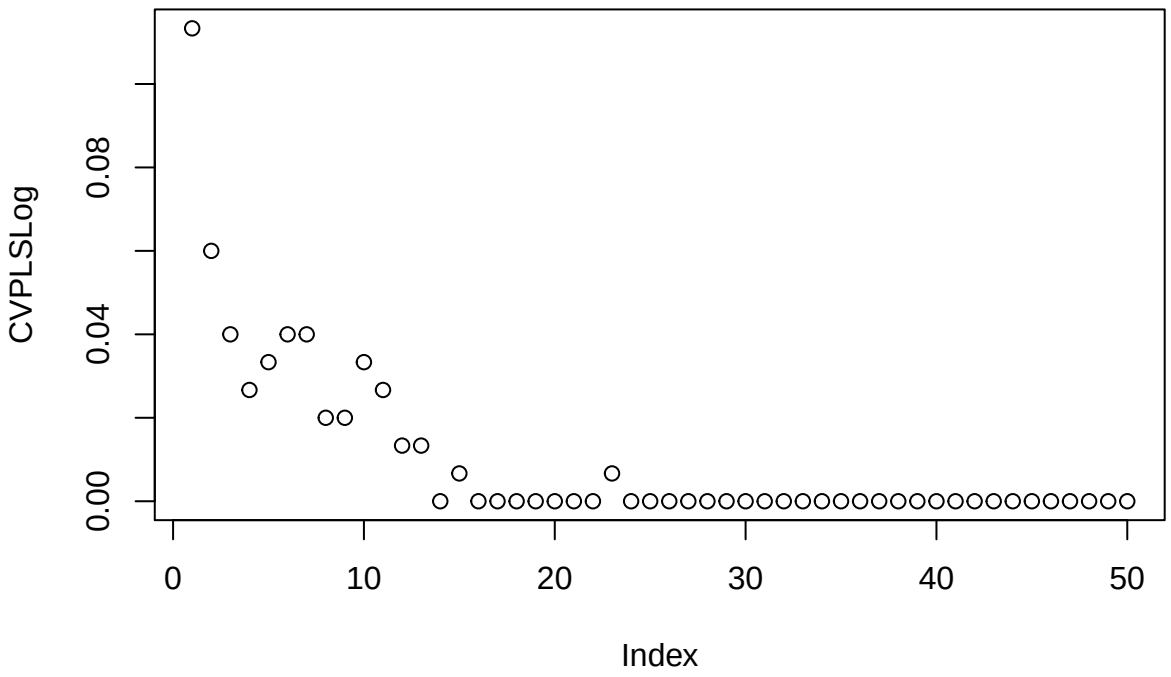
```
## [1]  6 18 21 23 24 30 33 34 35
```

```
minQDAPCA <- 6
```

For QDA classifier with the PCA components, from the result we can see that the minimum amount of components we need to include in order to make the classification error as small as possible is 6.

```
# PLS with logistic
n <- dim(Raintrain)[1]
CVPLSLog = rep(0,50)

for (k in 1:50) {
  for (i in 1:n) {
    fitPLS <- glm(G ~., data = dtPLS[-i, 1: (k + 1)], family=binomial(link = "logit"),
                  control = list(maxit = 50))
    glm.probs <- predict(fitPLS, newdata = dtPLS[i, 1:(k + 1)], type="response")
    glm.pred <- ifelse(glm.probs > 0.5, 1, 0)
    CVPLSLog[k] <- CVPLSLog[k] + sum(glm.pred != dtPLS[i, 1])
  }
  CVPLSLog[k] <- CVPLSLog[k]/n
}
```

```
plot(CVPLSLog, main = "Logistic: No. of PLS Compnents vs CV error")
```




```
which(CVPLSLog == min(CVPLSLog))
```

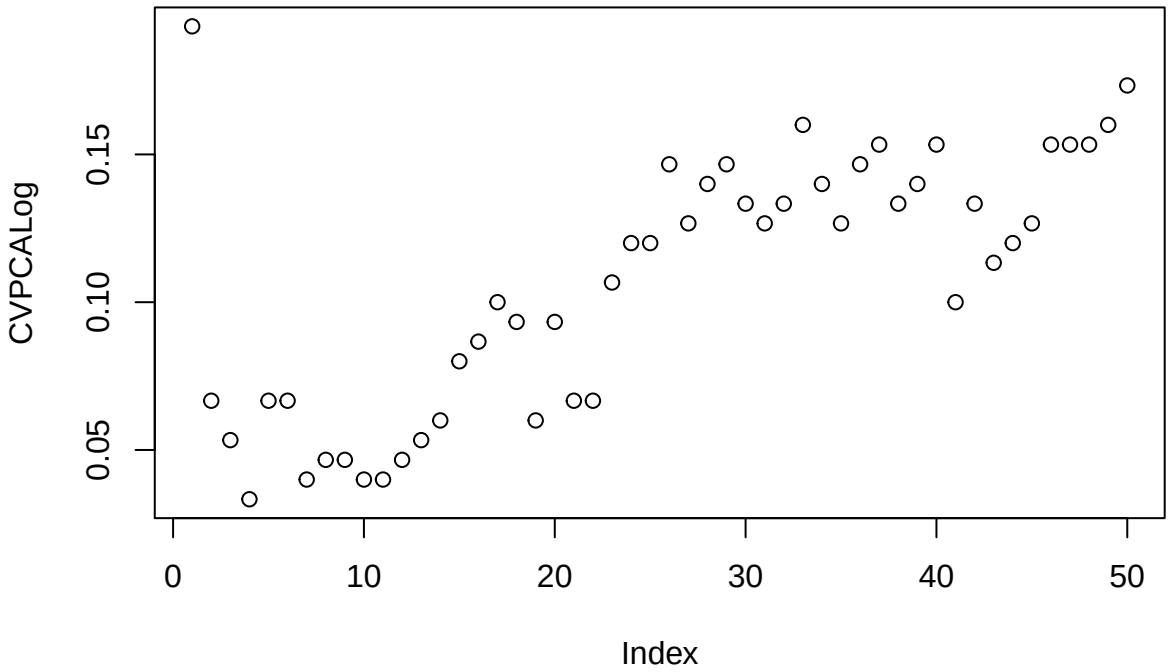
```
##  [1] 14 16 17 18 19 20 21 22 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40
## [26] 41 42 43 44 45 46 47 48 49 50
```

```
minLogPLS <- 14
```

For Logistic classifier with the PLS components, from the result we can see that the minimum amount of components we need to include in order to make the classification error as small as possible is 14.

```
# PCA with logistic
n <- dim(Raintrain)[1]
CVPCALog = rep(0,50)

for (k in 1:50) {
  for (i in 1:n) {
    fitPCA <- glm(G ~., data = dtPCA[-i, 1: (k + 1)], family=binomial(link = "logit"),
                  control = list(maxit = 50))
    glm.probs <- predict(fitPCA, newdata = dtPCA[i, 1: (k + 1)], type="response")
    glm.pred <- ifelse(glm.probs > 0.5, 1, 0)
    CVPCALog[k] <- CVPCALog[k] + sum(glm.pred != dtPCA[i, 1])
  }
  CVPCALog[k] <- CVPCALog[k]/n
}
```

```
plot(CVPCALog, main = "Logistic: No. of PCA Compnents vs CV error")
```




```
which(CVPCALog == min(CVPCALog))
```

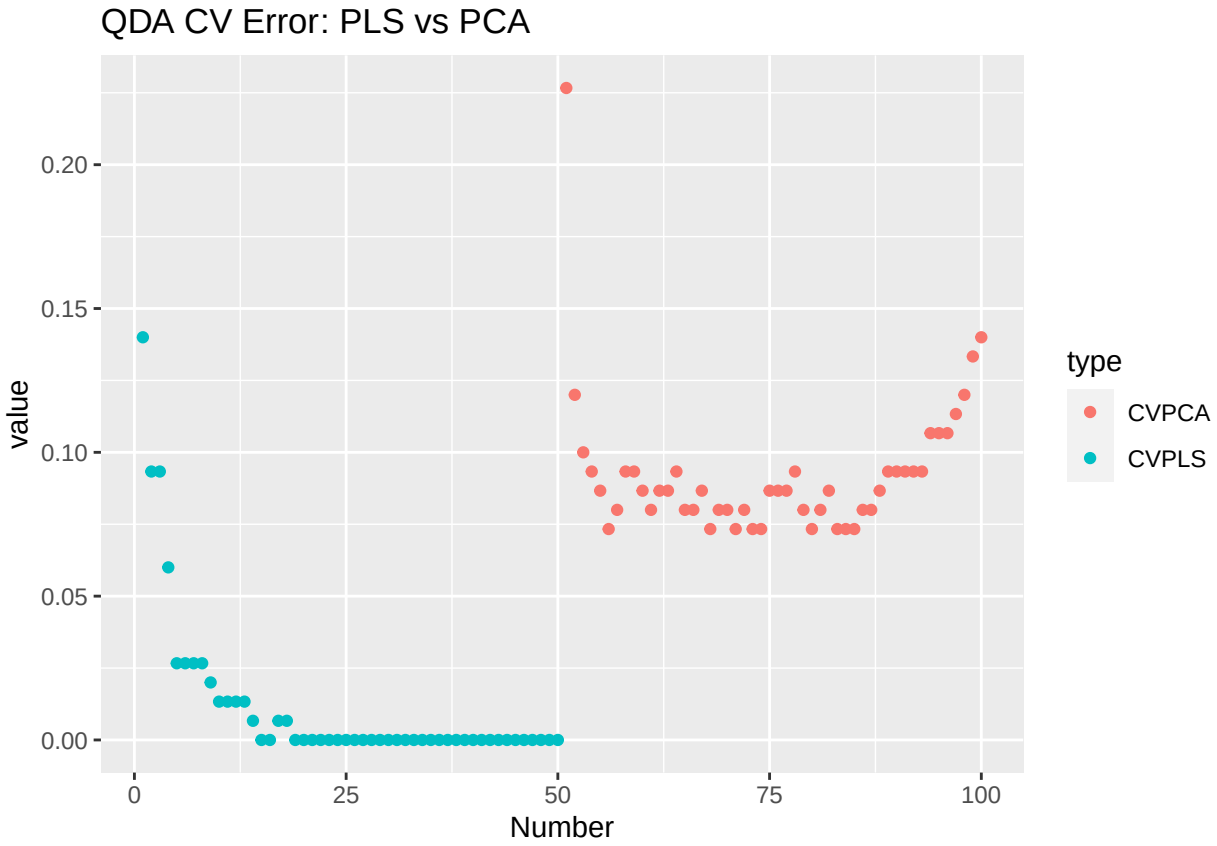
```
## [1] 4
```

```
minLogPCA <- 4
```

For Logistic classifier with the PCA components, from the result we can see that the minimum amount of components we need to include in order to make the classification error as small as possible is 4.

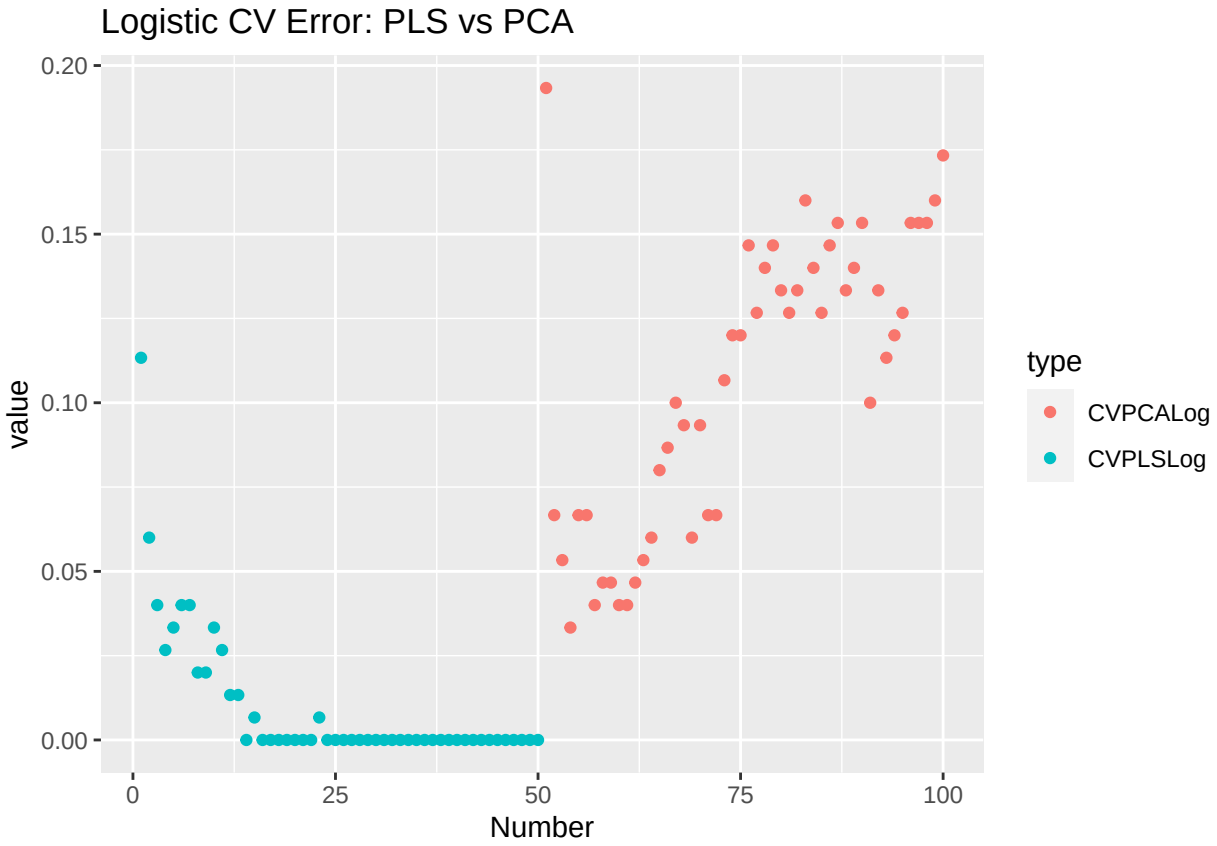
(d)

```
QDA <- data.frame(cbind(CVPLS, CVPCA)) %>%
  gather(key ="type", value = "value", CVPLS, CVPCA) %>% mutate(Number = seq(1, 100, 1))

QDA %>% ggplot(aes(x = Number, y = value, color = type)) + geom_point() +
  ggtitle("QDA CV Error: PLS vs PCA")
```



For QDA classifier, I prefer to choose PLS. This is because from the graph showing above, we can see that overall, the classification error rate for PLS is less than PCA. Even just with first few components, it seems that PLS performs better than PCA.

```
Log <- data.frame(cbind(CVPLSLog, CVPCALog)) %>%
  gather(key ="type", value = "value", CVPLSLog, CVPCALog) %>% mutate(Number = seq(1, 100, 1))

Log %>% ggplot(aes(x = Number, y = value, color = type)) + geom_point() +
  ggtitle("Logistic CV Error: PLS vs PCA")
```



For logistic classifier, I would also prefer to choose PLS. This is because from the graph, we can see that overall, logistic classifier with PLS has lower error rate compare to that with PCA.

In general, without verifying by test set, I would choose PLS as predictors, since it considers the relationship between response and predictors when doing dimension reduction. This means it will keep as much variance between response and predictors as possible or keep most of the information about the relationship between Z and X. In this way, it may perform better in predicting the response than the PCA which only keep most of the information about the design matrix itself.

(e)

```
nnew <- dim(Raintest)[1]
# PCA for test data: only includes X
PCAtest <- data.frame((as.matrix(Raintest[,1:365]) - matrix(rep(1, nnew), nrow = nnew) %*%
              colMeans(Raintrain[,1:365]))%*% gamma)
```

```
# PLS for test data: only includes X
PLStest <- data.frame((as.matrix(Raintest[,1:365]) - matrix(rep(1, nnew ), nrow = nnew) %*%
              colMeans(Raintrain[,1:365])) %*% PLX$projection)
```

```
# QDA with PLS on test
minQDAPLS <- 15
fitPLS <- qda(G ~., data = dtPLS[, 1: (minQDAPLS + 1)])
qdaclass <- predict(fitPLS, PLStest[, 1:minQDAPLS])$class
PLStester <- sum(qdaclass != Raintest[,366])/nnew
```

```
PLStester %>% kable(col.names = NULL, caption = "QDA w PLS test error")
```

Table 1: QDA w PLS test error

0.097561

```
# QDA with PCA on test
minQDAPCA <- 6
fitPCA <- qda(G ~., data = dtPCA[, 1: (minQDAPCA + 1)])
qdaclass <- predict(fitPCA, PCAtest[, 1:minQDAPCA])$class
PCAtester <- sum(qdaclass != Raintest[,366])/nnew
PCAtester %>% kable(col.names = NULL, caption = "QDA w PCA test error")
```

Table 2: QDA w PCA test error

0.097561

By applying PLS and PCA QDA models to the test set, they both result in 9.76% test error. It seems that these two classifier have the same performance in this case however there are less components involve in the PCA classifier than the PLS classifier.

```
# Logistic with PLS on test
minLogPLS <- 14
fitPLS <- glm(G ~., data = dtPLS[, 1: (minLogPLS + 1)], family=binomial(link = "logit"),
                  control = list(maxit = 50))
glm.probs <- predict(fitPLS, newdata = PLStest[, 1:minLogPLS], type="response")
glm.pred <- ifelse(glm.probs > 0.5, 1, 0)
PLSLogtest <- sum(glm.pred != Raintest[,366])/nnew
PLSLogtest %>% kable(col.names = NULL, caption = "Logistic w PLS test error")
```

Table 3: Logistic w PLS test error

0.1219512

```
# Logistic with PCA on test
minLogPCA <- 4
fitPCA <- glm(G ~., data = dtPCA[, 1: (minLogPCA + 1)], family=binomial(link = "logit"),
                  control = list(maxit = 50))
glm.probs <- predict(fitPCA, newdata = PCAtest[, 1:minLogPCA], type="response")
glm.pred <- ifelse(glm.probs > 0.5, 1, 0)
PCALogtest <- sum(glm.pred != Raintest[,366])/nnew
PCALogtest %>% kable(col.names = NULL, caption = "Logistic w PCA test error")
```

Table 4: Logistic w PCA test error

0.0243902

By applying PLS and PCA Logistic models to the test set, the logistic model with PLS result in 12.20% error and the logistic model with PCA result in 2.44% error.

It seems that in this case the PCA classifier perform better than the PLS classifier. Also, the PCA classifier may involve less components than the PLS classifier.

Overall, it seems that in this case, applying PCA to both QDA and Logistic perform better than applying PLS to both QDA and Logistic which is different from my guess/conclusion in previous question.

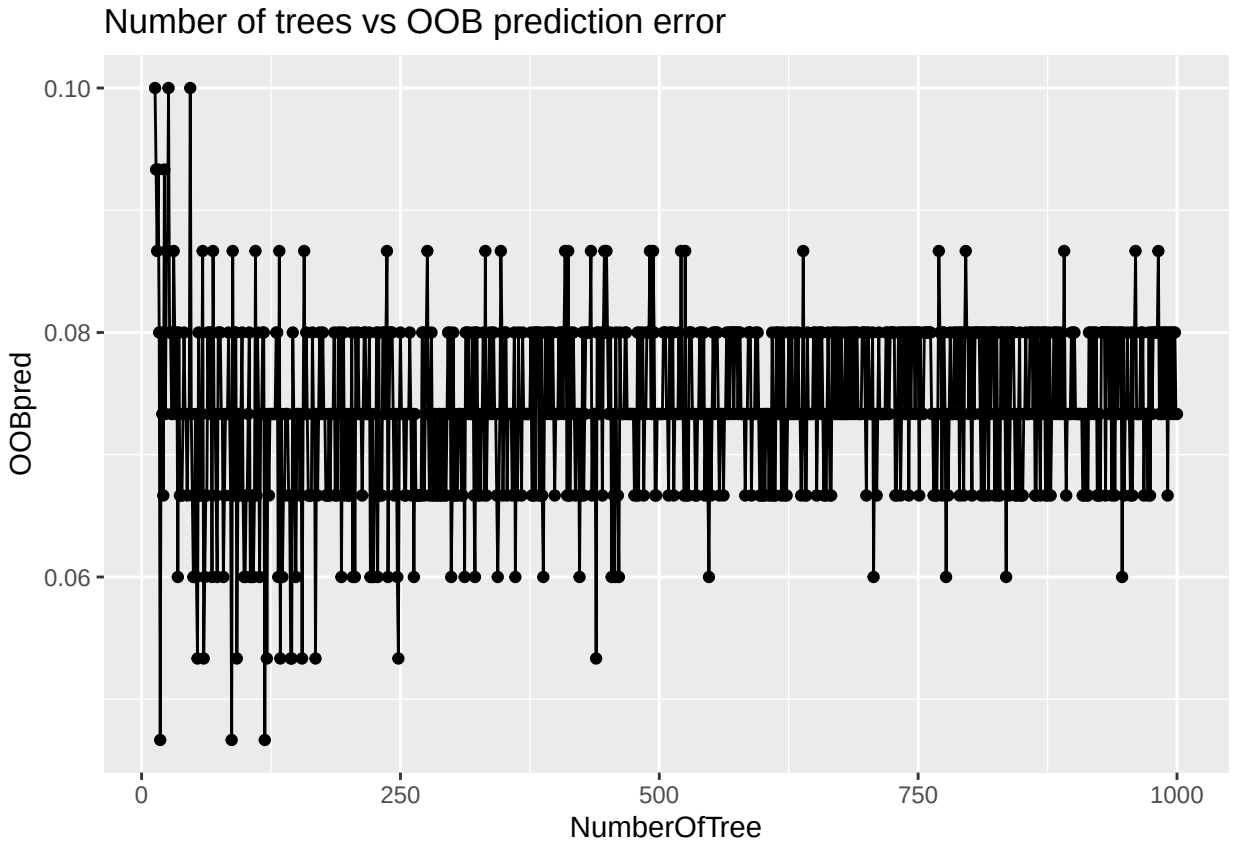
In this particular example, the reason why PCA performs better than PLS in prediction may be because PLS includes more predictors (maybe more information when doing dimension reduction) which results in overfitting. In other words, these additional information may be not that useful for prediction. The rainfall amount is recorded on a daily basis. There maybe some correlation between these features. The overfitting problem cause more variance which brings more errors when we apply to the test set.

Problem 2

(a)

```
set.seed(90138)
Raintrain$G <- as.factor(Raintrain$G)
m.rf <- randomForest(formula=G ~. , data = Raintrain, importance=TRUE)
OOBpred <- predict(m.rf)
```

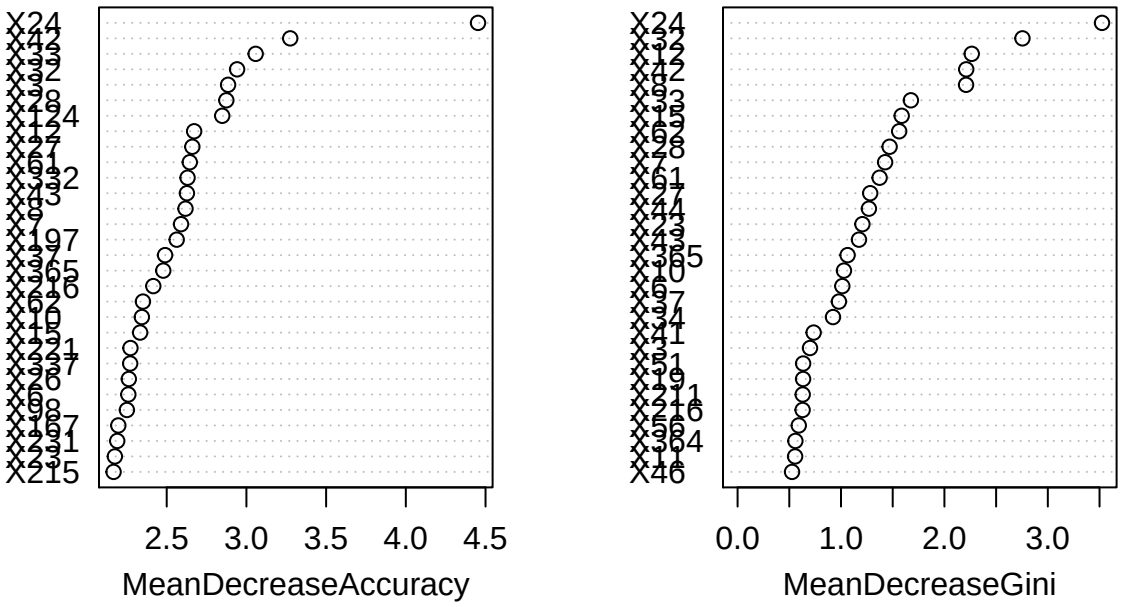
```
n <- dim(Raintrain)[1]
OOBpred <- rep(0, 1000)

for (ntree in 1: 1000) {
  fit <- randomForest(G ~., data=Raintrain, ntree=ntree)
  OOBpred[ntree] <- sum(predict(fit) != Raintrain[,366])/n
}

OOB <- data.frame(cbind(seq(1, 1000, 1), OOBpred)) %>% rename(NumberOfTree = V1)
na.omit(OOB) %>% ggplot(aes(x = NumberOfTree, y = OOBpred)) + geom_point() + geom_line() +
  ggtitle("Number of trees vs OOB prediction error")
```



We may want to take the B where the OOB error curve stabilizes. From the graph, we can see that the OOB error curve tends to stabilizes when Number of Tree is around 150 while while at 1000 trees boosting continues to improve. Therefore, in this case, we may choose B = 150.

(b)

```
set.seed(90138)
m.rf1 <- randomForest(formula=G ~. , ntree = 150, data = Raintrain, importance=TRUE)
varImpPlot(m.rf1)
```

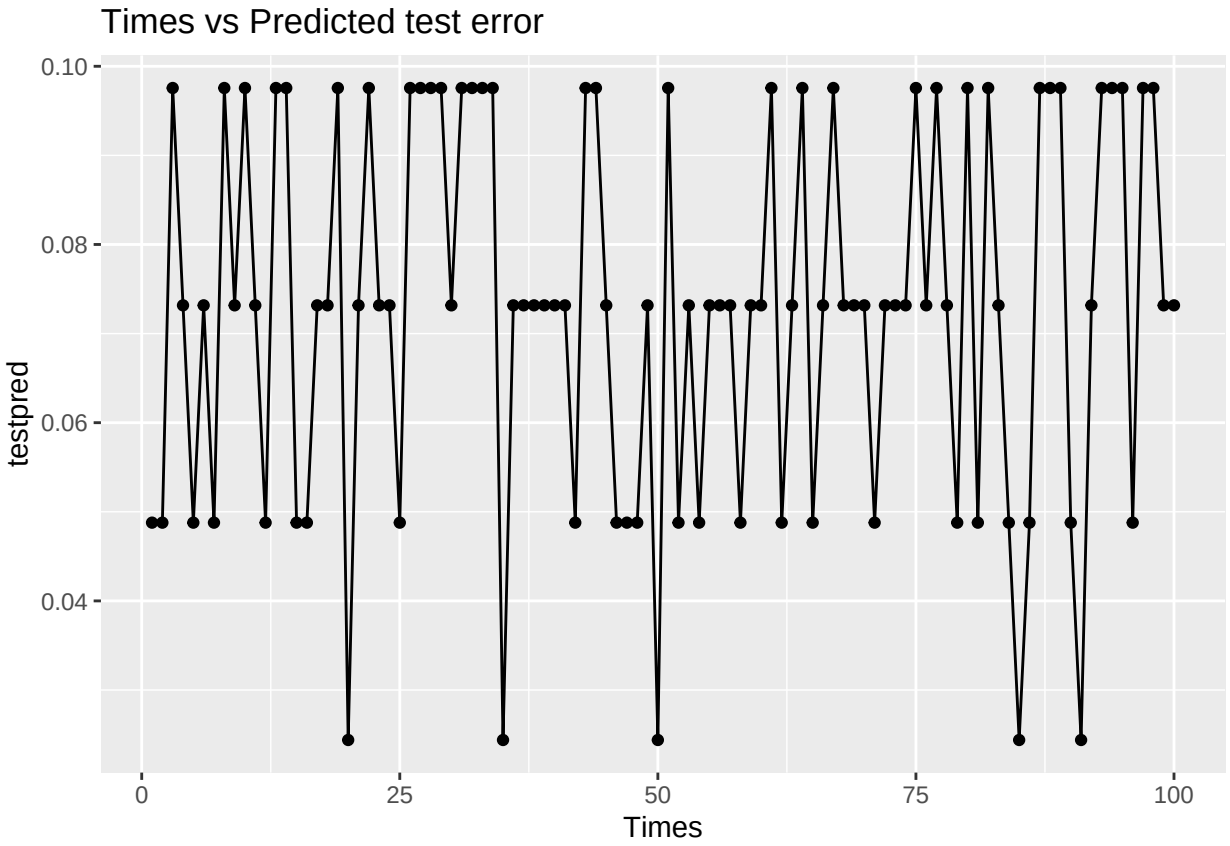


From the graphs, we can see that most of the selected features are either within the range between X1 to X60 or somewhat close to X365. These means that these features(rainfall in these days) play an important part when using the rainforest classifier to do classification.

In reality, features range between X1 to X60 or features somewhat close to X365 represent the days in December to February. This means that it can be easier to classify a place to north or south in Austrila by looking at its rainfall amount in summer.

(c)

```
ntest <- dim(Raintest)[1]
Raintest$G <- as.factor(Raintest$G)
testpred <- rep(0, 100)

for (i in 1: 100) {
  fit <- randomForest(formula=G ~. , ntree = 150, data = Raintrain)
  testpred[i] <- sum(predict(fit, Raintest) != Raintest[,366])/ntest
}

testerror <- data.frame(cbind(seq(1, 100, 1), testpred)) %>% rename(Times = V1)
testerror %>% ggplot(aes(x = Times, y = testpred)) + geom_point() + geom_line() +
  ggtitle("Times vs Predicted test error")
```



From the graph we can see that if train the radomforest for 100 times and apply them to the test data, we would get different test error. This is because here we include all the features in the model, some of them are significant and some of them are insignificant. When growing a bootstrap tree, before splitting, randomforest would select the features randomly as candidates for splitting. In this way, it can be the case that it would select those insignificant features at a time which may result in different outcome finally. This may also because the number of trees is still not large enough in this case.

In order to make the randomforest more stable, we can first select a set of features which are significant in prediction. In this way, we can reduce those misleading variables and make sure that the predictors are useful in prediction. We can also increase the number of trees in randomforest classifier as more trees will increase the prediction accuracy and stability. In other words, the final classification result of each radomforest will be the integration of all the sub-trees within that randomforest.

Problem 3

```
# QDA with PLS on test
PLStester %>% kable(col.names = NULL, caption = "QDA w PLS test error")
```

Table 5: QDA w PLS test error

0.097561

```
# QDA with PCA on test
PCAtester %>% kable(col.names = NULL, caption = "QDA w PCA test error")
```

Table 6: QDA w PCA test error

0.097561

```
# Logistic with PLS on test
PLSLogtest %>% kable(col.names = NULL, caption = "Logistic w PLS test error")
```

Table 7: Logistic w PLS test error

0.1219512

```
# Logistic with PCA on test
PCALogtest %>% kable(col.names = NULL, caption = "Logistic w PCA test error")
```

Table 8: Logistic w PCA test error

0.0243902

```
# Random Forest test error range
range(testpred) %>% kable(col.names = NULL, caption = "Randomforest test error")
```

Table 9: Randomforest test error

0.0243902
0.0975610

By comparing these results, we can see that the logistic regression model with PCA works the best and the logistic regression model with PLS works the worst. QDA with PLS has the same test error rate as QDA with PCA. They are slightly better than the Logistic with PLS but worst than Logistic with PCA. Randomforest is not that stable. Its result is somewhat between QDA and Logistic with PCA.

This kind of makes sense, because for the training set, we only have 150 observations with 365 predictors in this case. The number of observations is way less than the predictors. Thus it is crucial to reduce the variance. If we fit a linear model, although it is not that flexible, it can reduce variance. However, QDA assumes there is a quadratic decision boundary and each class has its own covariance matrix. To some extend, this can reduce the bias but may bring more variance at the same time. With relatively few training observations, this may be the reason why logistic with PCA works better than QDA.

Comparing Logistic with PCA and Logistic with PLS, it seems that logistic with PCA works better than logistic with PLS. The reason why this happens may be because PLS tries to keep as much information about the relationship between response and predictors as possible. But in this case, as mentioned before, the number of observations is not large enough. PLS may allow the logistic model to perfect fit the training data, but when applying it to the test set, the large variance may result in large test error. PCA only considers the covariance of the design matrix itself, which may not result in perfect fit to the training data and also reduce the variance to some extend. We can also see these situations from the **graphs on Problem 1(d)** where it shows that PLS with logistic may perfect fit the training data while the one with PCA not.

The result from randomforest classifier is not stable. As mentioned in Problem 2(c), this may be because we

include all of the predictors without selecting some important predictors. When building a randomforest, it may select those insignificant predictors which may come up with
a different result. Thus this may influence the prediction. Also, the number of trees may be not large enough to make the result stable.