Assignment 2

Assignment 2

Biomedical Data Science

Due on Thursday 18th March 2020, 5:00pm

The assignment is marked out of 100 points, and will contribute to 30% of your final mark. Please knit this document in PDF format and submit using the gradescope link on Learn. If you can't knit to PDF directly, knit it to word and you should be able to either convert to PDF or print it and scan to PDF using a scanning app on your phone. If you have any code that doesn't run you won't be able to knit the document so comment it as you might still get some grades for partial code. Clear and reusable code will be rewarded so pay attention to indentation, choice of variable identifiers, comments, error checking, etc. An initial code chunk is provided after each subquestion but create as many chunks as you feel is necessary to make a clear report. Add plain text explanations in between the chunks as and when required and any comments necessary within code chunks to make it easier to follow your code/reasoning.

Problem 1 (27 points)

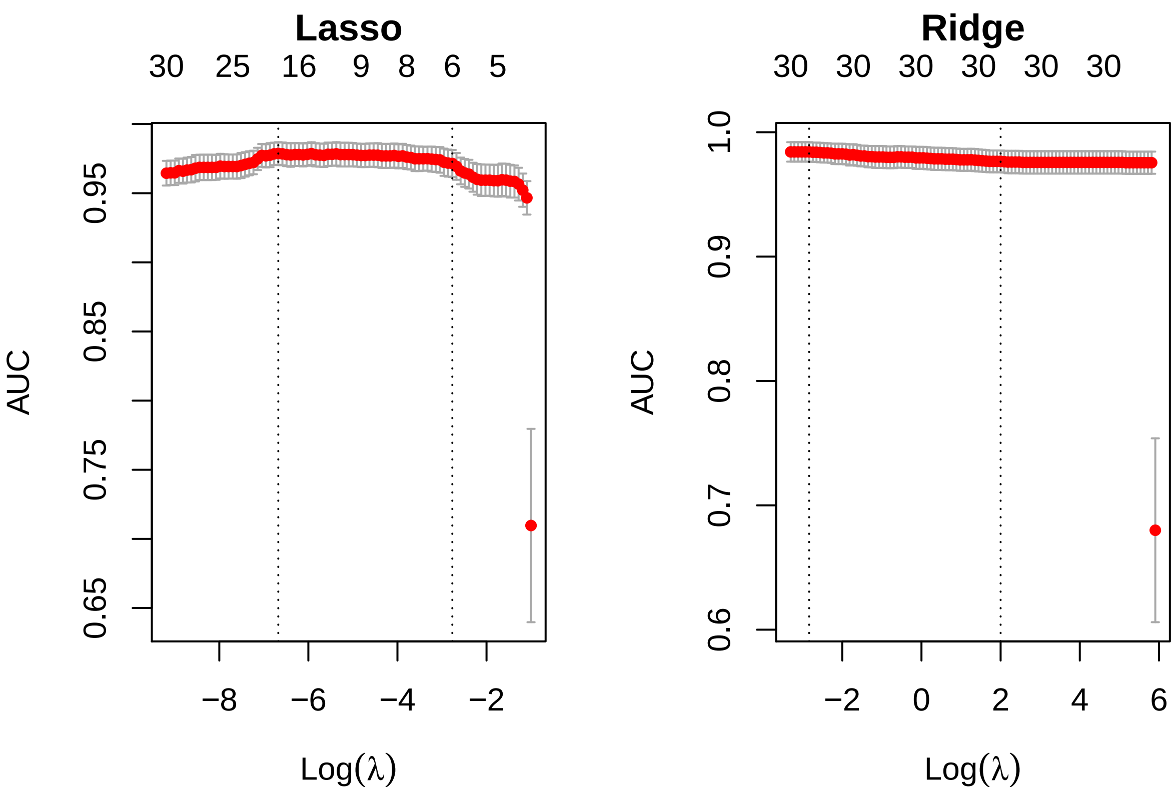
File wdbc2.csv (available from the accompanying zip folder on Learn) refers to a study of breast cancer where the outcome of interest is the type of the tumour (benign or malignant, recorded in column "diagnosis"). The study collected 30 imaging biomarkers on 569 patients.

Problem 1.a (7 points)

Using package caret, create a data partition so that the training set contains 70% of the observations (set the random seed to 984065 beforehand). Fit both a ridge regression model and a lasso model which uses cross-validation on the training set to diagnose the type of tumour from the 30 biomarkers. Then use a plot to help identify the penalty parameter λ that maximizes the AUC. Note: There is no need to use the prepare.glmnet() function from lab 4, using as.matrix() with the required columns is sufficient.

```r
# Enter code here.
set.seed(984065)
wdbc2.dt <- fread("data/wdbc2.csv")
wdbc2.dt$diagnosis <- ifelse(wdbc2.dt$diagnosis=='malignant',1,0)

ind <- createDataPartition(wdbc2.dt$diagnosis, p=0.7, list=FALSE)
data.train <- wdbc2.dt[ind,]
data.test <- wdbc2.dt[-ind,]

# check that the split is actually 70-30
# table(wdbc2.dt$diagnosis)[1] / (table(wdbc2.dt$diagnosis)[1] + table(wdbc2.dt$diagnosis)[2])
# table(data1.train$diagnosis)[1] / (table(data1.train$diagnosis)[1] + table(data1.train$diagnosis)[2])
# table(data1.test$diagnosis)[1] / (table(data1.test$diagnosis)[1] + table(data1.test$diagnosis)[2])

X.train <- data.train[, !c("diagnosis","id"), with=FALSE]
X.test  <- data.test [, !c("diagnosis","id"), with=FALSE]
y.train <- data.train[,  c("diagnosis"), with=FALSE]
y.test  <- data.test [,  c("diagnosis"), with=FALSE]
```

```
fit.cv.lasso <- cv.glmnet(as.matrix(X.train), as.matrix(y.train), family='binomial', type.measure = c('
fit.cv.ridge <- cv.glmnet(as.matrix(X.train), as.matrix(y.train), family='binomial', type.measure = c('
par(mfrow=c(1,2), mar=c(4,4,5,2))
plot(fit.cv.lasso, main="Lasso")
plot(fit.cv.ridge, main="Ridge")
```



Problem 1.b (2 points)

Create a data table that for each value of 'lambda.min' and 'lambda.1se' for each model fitted in problem 1.a reports: * the corresponding AUC, * the corresponding model size. Use 3 significant digits for floating point values and comment on these results. Hint: The AUC values are stored in the field called 'cvm'.

```
# Enter code here.
lasso.min.pos    <- which(fit.cv.lasso$lambda==fit.cv.lasso$lambda.min)
lasso.1se.pos    <- which(fit.cv.lasso$lambda==fit.cv.lasso$lambda.1se)
lasso.model      <- 'lasso'
lasso.lambda.min <- round(fit.cv.lasso$lambda.min,3)
lasso.lambda.1se <- round(fit.cv.lasso$lambda.1se,3)
lasso.model.min  <- round(fit.cv.lasso$nzero[lasso.min.pos],3)
lasso.model.1se  <- round(fit.cv.lasso$nzero[lasso.1se.pos],3)
lasso.auc.min    <- round(fit.cv.lasso$cvm[lasso.min.pos],3)
lasso.auc.1se    <- round(fit.cv.lasso$cvm[lasso.1se.pos],3)

ridge.min.pos    <- which(fit.cv.ridge$lambda==fit.cv.ridge$lambda.min)
ridge.1se.pos    <- which(fit.cv.ridge$lambda==fit.cv.ridge$lambda.1se)
ridge.model      <- 'ridge'
ridge.lambda.min <- round(fit.cv.ridge$lambda.min,3)
ridge.lambda.1se <- round(fit.cv.ridge$lambda.1se,3)
ridge.model.min  <- round(fit.cv.ridge$nzero[ridge.min.pos],3)
ridge.model.1se  <- round(fit.cv.ridge$nzero[ridge.1se.pos],3)
ridge.auc.min    <- round(fit.cv.ridge$cvm[ridge.min.pos],3)
ridge.auc.1se    <- round(fit.cv.ridge$cvm[ridge.1se.pos],3)

model.lasso.row  <- c(lasso.model, lasso.lambda.min, lasso.model.min, lasso.auc.min, lasso.lambda.1se, l
model.ridge.row  <- c(ridge.model, ridge.lambda.min, ridge.model.min, ridge.auc.min, ridge.lambda.1se, r
results.train    <- as.data.table(rbind(model.lasso.row, model.ridge.row))
cols             <- c('model', 'lambda.min', 'variables.min', 'auc.min', 'lambda.1se', 'variables.1se',
setnames(results.train, cols)

results.train
```

```
##    model lambda.min variables.min auc.min lambda.1se variables.1se auc.1se
## 1: lasso      0.001            20   0.979      0.063             6   0.971
## 2: ridge      0.059            30   0.984      7.389            30   0.977
```

Problem 1.c (7 points)

Perform both backward (we'll later refer to this as model B) and forward (model S) stepwise selection on the same training set derived in problem 1.a. Report the variables selected and their standardized regression coefficients in decreasing order of the absolute value of their standardized regression coefficient. Discuss the results and how the different variables entering or leaving the model influenced the final result.

```
full.model <- suppressWarnings(glm(data.train$diagnosis ~ . , data=data.train, family='binomial'))
modelB     <- suppressWarnings(stepAIC(full.model, direction="back", trace=FALSE))
null.model <- suppressWarnings(glm(data.train$diagnosis ~ 1 , data=data.train, family='binomial'))
modelS     <- suppressWarnings(stepAIC(null.model, scope=list(upper=full.model), direction="forward", t

modelB
```

```
##
## Call:  glm(formula = data.train$diagnosis ~ radius + perimeter + concavepoints +
##     radius.stderr + texture.stderr + radius.worst + texture.worst +
##     area.worst + smoothness.worst + compactness.worst + concavity.worst +
##     concavepoints.worst, family = "binomial", data = data.train)
##
## Coefficients:
##         (Intercept)               radius            perimeter
##           -59.98039              1.08925             -0.38156
##       concavepoints        radius.stderr       texture.stderr
##           103.44514             14.28442             -2.94382
##        radius.worst        texture.worst           area.worst
##             5.39339              0.43813             -0.03597
##    smoothness.worst    compactness.worst      concavity.worst
##            43.34720            -16.81214             20.07203
## concavepoints.worst
##           -28.09322
##
## Degrees of Freedom: 398 Total (i.e. Null);  386 Residual
## Null Deviance:       527.3
## Residual Deviance: 73.47     AIC: 99.47
```

```
modelS
```

```
##
## Call:  glm(formula = data.train$diagnosis ~ perimeter.worst + concavity +
##     texture.worst + radius.stderr + area.stderr + smoothness.worst +
##     radius + concavity.worst + perimeter.stderr + area.worst +
##     compactness.worst + perimeter + radius.worst + texture.stderr,
##     family = "binomial", data = data.train)
##
## Coefficients:
##       (Intercept)    perimeter.worst          concavity      texture.worst
##         -64.81578            0.23889           33.09024            0.38552
##     radius.stderr        area.stderr   smoothness.worst             radius
##          19.83368            0.03095           57.43884            0.81858
##   concavity.worst   perimeter.stderr         area.worst  compactness.worst
##          10.18660           -1.25851           -0.03928          -18.73717
##         perimeter       radius.worst     texture.stderr
##          -0.27491            3.97715           -2.39510
##
## Degrees of Freedom: 398 Total (i.e. Null);  384 Residual
## Null Deviance:       527.3
## Residual Deviance: 75.29     AIC: 105.3
```

Problem 1.d (3 points)

Compare the goodness of fit of model B and model S in an appropriate way.

```
# Enter code here.
# Chi-square goodness of fit tests and deviance
signif(pchisq(modelB$null.deviance - modelB$deviance, df=12, lower.tail=FALSE),2)
```

```
## [1] 1.5e-89
```

```
signif(pchisq(modelS$null.deviance - modelS$deviance, df=14, lower.tail=FALSE),2)
```

```
## [1] 1.4e-87
```

Problem 1.e (2 points)

Compute the training AUC for model B and model S.

```
# Enter code here.
auc.modelB <- roc(data.train$diagnosis, modelB$fitted.values)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
auc.modelS <- roc(data.train$diagnosis, modelS$fitted.values)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```
auc.modelB$auc
```

```
## Area under the curve: 0.9936
```

```
auc.modelS$auc
```

```
## Area under the curve: 0.9929
```

Problem 1.f (6 points)

Use the four models to predict the outcome for the observations in the test set (use the lambda at 1 standard error for the penalised models). Plot the ROC curves of these models (on the same plot, using different colours) and report their test AUCs. Compare the training AUCs obtained in problems 1.b and 1.e with the test AUCs and discuss the fit of the different models.

```
# Enter code here.
modelB.pred <- predict(modelB, newdata=data.test, type='response')
modelS.pred <- predict(modelS, newdata=data.test, type='response')
lasso.pred  <- predict(fit.cv.lasso, newx=as.matrix(X.test), s=lasso.lambda.1se, type='response')
ridge.pred  <- predict(fit.cv.ridge, newx=as.matrix(X.test), s=ridge.lambda.1se, type='response')

roc.modelB <- roc(data.test$diagnosis, modelB.pred , plot=TRUE, col='blue4', direction="<")
```

```
## Setting levels: control = 0, case = 1
```

```
roc.modelS <- roc(data.test$diagnosis, modelS.pred , plot=TRUE, col='red4', direction="<", add=TRUE, qu
```

```
## Setting levels: control = 0, case = 1
```

```
roc.lasso  <- roc(data.test$diagnosis, lasso.pred  , plot=TRUE, col='orange4', direction="<", add=TRUE,
```

```
## Setting levels: control = 0, case = 1
```

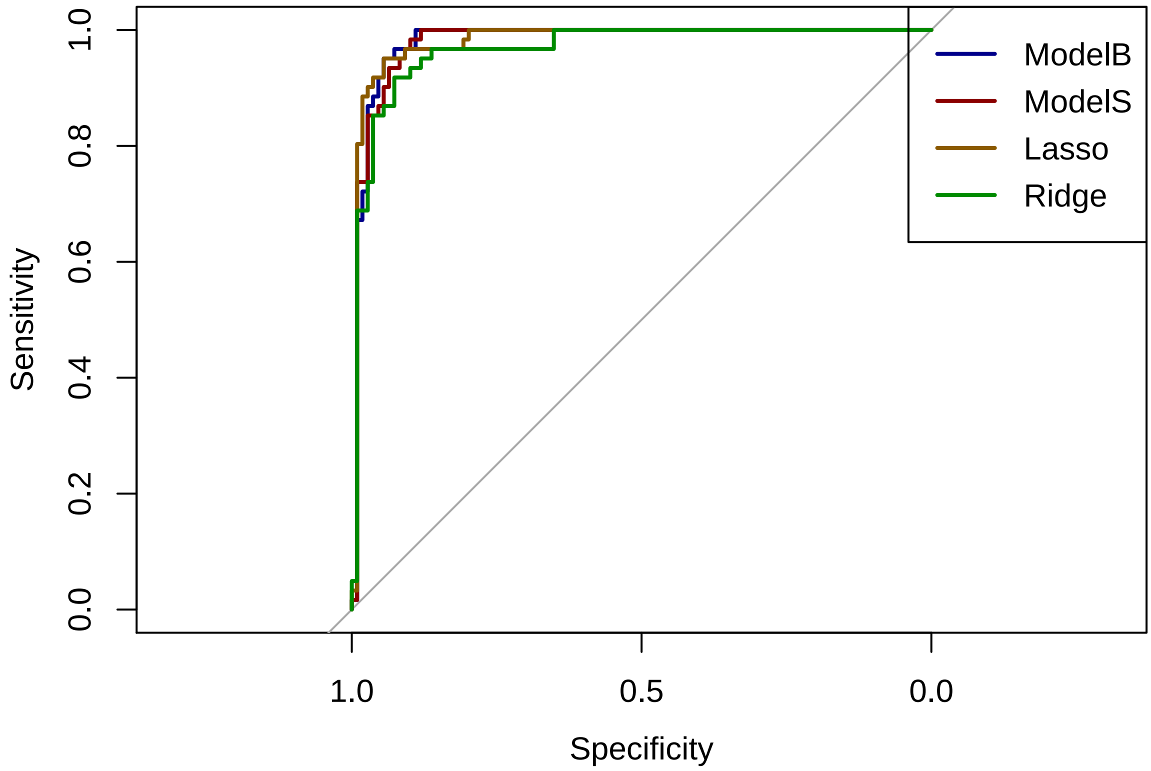
```
## Warning in roc.default(data.test$diagnosis, lasso.pred, plot = TRUE, col =
## "orange4", : Deprecated use a matrix as predictor. Unexpected results may be
## produced, please pass a numeric vector.
```

```
roc.ridge  <- roc(data.test$diagnosis, ridge.pred  , plot=TRUE, col='green4', direction="<", add=TRUE,
```

```
## Setting levels: control = 0, case = 1
```

```
## Warning in roc.default(data.test$diagnosis, ridge.pred, plot = TRUE, col =
## "green4", : Deprecated use a matrix as predictor. Unexpected results may be
## produced, please pass a numeric vector.
```

```
legend(x = 'topright', legend = c('ModelB', 'ModelS', 'Lasso', 'Ridge'),
       col=c('blue4', 'red4', 'orange4', 'green4'), lwd=2)
```



```
row1 <- c('lasso' , signif(roc.lasso$auc,4), signif(lasso.auc.1se,4))
row2 <- c('ridge' , signif(roc.ridge$auc,4), signif(ridge.auc.1se,4))
row3 <- c('modelB', signif(roc.modelB$auc,4), signif(auc.modelB$auc,4))
row4 <- c('modelS', signif(roc.modelS$auc,4), signif(auc.modelS$auc,4))
results.final <- as.data.table(rbind(row1,row2,row3,row4))
cols <- c('model', 'AUC.test', 'AUC.train')
setnames(results.final, cols)
results.final[order(-AUC.test),]
```

```
##     model AUC.test AUC.train
## 1:  lasso   0.9806     0.971
## 2: modelB   0.9803    0.9936
## 3: modelS   0.9791    0.9929
## 4:  ridge   0.9668     0.977
```

Problem 2 (40 points)

File GDM.raw.txt (available from the accompanying zip folder on Learn) contains 176 SNPs to be studied for association with incidence of gestational diabetes (a form of diabetes that is specific to pregnant women). SNP names are given in the form "rs1234_X" where "rs1234" is the official identifier (rsID), and "X" (one of A, C, G, T) is the reference allele.

Problem 2.a (3 points)

Read file GDM.raw.txt into a data table named gdm.dt. Impute missing values in gdm.dt according to SNP-wise median allele count.

```
#' This is a function that takes as impute a column of a data.table and imputes
#' the NAs with its mean / mode if the vector is numeric or categorical respectively.
#' @param x A vector of numeric or categorical values for which the NAs will be imputed.
impute.to.median <- function(x) {
    if (all(na.omit(x) %in% 0L:2L)){
      x[is.na(x)] = median(x, na.rm=TRUE)
    }
    return(x)
  }
```

```
# Enter code here.
gdm.dt <- fread("data/GDM.raw.txt")
numcols <- colnames(gdm.dt)
gdm.dt %>% .[, (numcols) := lapply(.SD, impute.to.median), .SDcols = numcols]
```

Problem 2.b (8 points)

Write function univ.glm.test <- function(x, y, order = FALSE) where x is a data table of SNPs, y is a binary outcome vector, and order is a boolean. The function should fit a logistic regression model for each SNP in x, and return a data table containing SNP names, regression coefficients, odds ratios, standard errors and p-values. If order is set to TRUE, the output data table should be ordered by increasing p-value.

```
# run univariate tests of associations for all SNPs(columns of az)
univ.glm.test <- function(x, y, ordering=FALSE) {
  stopifnot(all(na.omit(y) %in% 0L:1L))
  output <- NULL
    for (i in 1:ncol(x)){
      regr <- glm(y ~ x[[i]], family='binomial')

      data <- transpose(as.data.table(coef(summary(regr))[-1, -3]))
      data <- cbind(data, exp(coef(regr))[2]) # odds ration calculation
      data <- cbind(data, colnames(X[1,])[i]) # keep column name as argument on the output
      output <- rbind(output, data)
    }
  # assign better column names
    colnames(output) <- c("beta","std.error", "p.value", "odds ratio", "snp_full")
    return(output[order(output$"p.value"*ordering)])
    # The requirement is strictly to write the function with an argument called
    # 'order', not 'ordering'. When the argument was called 'order', there was a
    # problem because there is the function 'order()' as well. In any case,
    # strictly speaking with the function argument 'order', I would solve it like:
    #
    # if(!order) {return(output)} else {return(output[order(output$"p.value"), ])}
  }
```

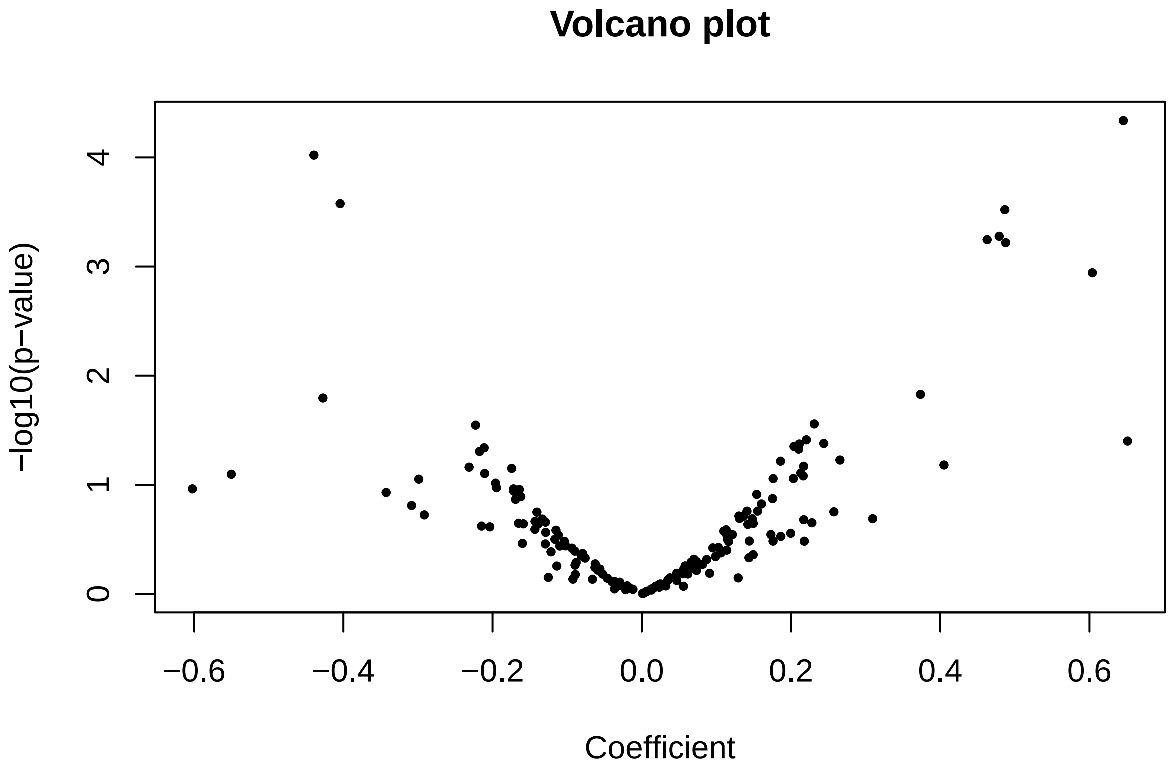
Problem 2.c (5 points)

Using function univ.glm.test(), run an association study for all the SNPs in gdm.dt against having gestational diabetes (column "pheno"). For the SNP that is most strongly associated to increased risk of gestational diabetes and the one with most significant protective effect, report the summary statistics from the GWAS as well as the 95% and 99% confidence intervals on the odds ratio.

```
X <- as.data.table(gdm.dt[,4:ncol(gdm.dt)])
target <- gdm.dt$pheno
association <- data.table(univ.glm.test(x=X,y=target, ordering=FALSE), key=c('snp_full'))
association
```

```
##             beta  std.error    p.value odds ratio     snp_full
##   1:  0.14214661 0.11864045 0.23086663  1.1527456 rs10150332_A
##   2: -0.07876094 0.10208514 0.44039753  0.9242609 rs10488683_A
##   3:  0.07335711 0.14381095 0.60998558  1.0761148  rs1052248_G
##   4: -0.03621755 0.13986052 0.79567012  0.9644305 rs10767664_C
##   5: -0.06274131 0.11130078 0.57295182  0.9391864 rs10770141_A
##  ---
## 172: -0.21128464 0.10579572 0.04581431  0.8095436   rs972283_A
## 173: -0.15998626 0.16926370 0.34456217  0.8521555  rs9816226_C
## 174: -0.17436723 0.09651276 0.07081291  0.8399884   rs987237_C
## 175: -0.04611920 0.12793644 0.71848429  0.9549281  rs9939609_A
## 176:  0.11644920 0.11969475 0.33061059  1.1235004  rs9941349_A
```

```
plot(association[, .(beta, -log10(p.value))],
     pch = 19, cex = 0.5,
     main = "Volcano plot",
     xlab = "Coefficient",
     ylab = "-log10(p-value)"
     )
abline(h = -log10(5e-8), lty = 2, col = "red") # genome-wide significance threshold
```



```
# Biggest risk
threshold = 0.05
risk <- association[p.value < threshold,]
risk <- risk[order(-beta),]
biggest_risk <- risk[1,]
biggest_protect <- risk[dim(risk)[1],]
sns_risk <- biggest_risk[,snp_full][1]
sns_protect <- biggest_protect[,snp_full][1]

sns_risk.dt <- gdm.dt[,..sns_risk]
sns_protect.dt <- gdm.dt[,..sns_protect]
data <- cbind(target, sns_risk.dt, sns_protect.dt)

risk_logistic    <- glm(data[[1]] ~ data[[2]], family='binomial')
protect_logistic <- glm(data[[1]] ~ data[[3]], family='binomial')

results <- as.data.table(coef(summary(risk_logistic))[,-3])
results <- cbind(results, confint(risk_logistic, level=0.95), confint(risk_logistic, level=0.99))
```

```
## Waiting for profiling to be done...
## Waiting for profiling to be done...
```

```
temp <- as.data.table(coef(summary(protect_logistic))[,-3])
temp <- cbind(temp, confint(protect_logistic, level=0.95), confint(protect_logistic, level=0.99))
```

```
## Waiting for profiling to be done...
## Waiting for profiling to be done...
```

```
results <- rbind(results, temp)
sns_name <- c(sns_risk, sns_risk, sns_protect, sns_protect)
sns_role <- c('risk','risk','protect','protect')
```

```
results <- cbind(sns_name, sns_role, results)
beta <- c('Intercept', 'beta1','Intercept', 'beta1')
results <- cbind(beta, results)
results
```

```
##         beta    sns_name sns_role    Estimate Std. Error     Pr(>|z|)
## 1: Intercept rs1423096_T     risk  0.08241388 0.07340542 2.615556e-01
## 2:     beta1 rs1423096_T     risk  0.65106408 0.31665472 3.977583e-02
## 3: Intercept rs2237897_T  protect  0.37439772 0.09727068 1.185867e-04
## 4:     beta1 rs2237897_T  protect -0.43944560 0.11261333 9.530178e-05
##          2.5 %     97.5 %      0.5 %     99.5 %
## 1: -0.06137965  0.2264890 -0.1065641  0.2718783
## 2:  0.04920779  1.3002071 -0.1347336  1.5188279
## 3:  0.18468070  0.5662477  0.1253780  0.6271024
## 4: -0.66191949 -0.2200563 -0.7326623 -0.1515496
```

Problem 2.d (4points)

Merge your GWAS results with the table of gene names provided in file GDM.annot.txt (available from the accompanying zip folder on Learn). For SNPs that have p-value $< 10^{-4}$ (hit SNPs) report SNP name, effect allele, chromosome number and corresponding gene name. Separately, report for each 'hit SNP' the names of the genes that are within a 1Mb window from the SNP position on the chromosome. Note: That's genes that fall within +/- 1,000,000 positions using the 'pos' column in the dataset.

```
# Enter code here.
gdm.annot.dt <- data.table(fread('data/GDM.annot.txt'), key=c('snp'))
pk <- c('snp')
association[,snp:=substring(snp_full, 1, nchar(snp_full)-2)]
association[,allele:=substring(snp_full, nchar(snp_full),nchar(snp_full))]
# association<- association[,-'snp_full']
association.ext <- merge(association,
                         gdm.annot.dt,
                         by=pk,
                         all=TRUE)[order(snp)]
association.ext
```

```
##              snp        beta  std.error    p.value odds ratio     snp_full
##   1: rs10150332  0.14214661 0.11864045 0.23086663  1.1527456 rs10150332_A
##   2: rs10488683 -0.07876094 0.10208514 0.44039753  0.9242609 rs10488683_A
##   3:  rs1052248  0.07335711 0.14381095 0.60998558  1.0761148  rs1052248_G
##   4: rs10767664 -0.03621755 0.13986052 0.79567012  0.9644305 rs10767664_C
##   5: rs10770141 -0.06274131 0.11130078 0.57295182  0.9391864 rs10770141_A
##  ---
## 172:   rs972283 -0.21128464 0.10579572 0.04581431  0.8095436   rs972283_A
## 173:  rs9816226 -0.15998626 0.16926370 0.34456217  0.8521555  rs9816226_C
## 174:   rs987237 -0.17436723 0.09651276 0.07081291  0.8399884   rs987237_C
## 175:  rs9939609 -0.04611920 0.12793644 0.71848429  0.9549281  rs9939609_A
## 176:  rs9941349  0.11644920 0.11969475 0.33061059  1.1235004  rs9941349_A
##      allele chrom       pos         gene
##   1:      A    14  79936964        NRXN3
##   2:      A    11  18053545         TPH1
##   3:      G     6  31556581      NFKBIL1
##   4:      C    11  27725986         BDNF
##   5:      A    11   2193840           TH
##  ---
## 172:      A     7 130466854 LOC105375508
## 173:      C     3 185834499
## 174:      C     6  50803050       TFAP2B
## 175:      A    16  53820527          FTO
## 176:      A    16  53825488          FTO
```

```
report1 <- association.ext[p.value < 1e-4, c('snp','allele','chrom','gene','pos')]
report1[,c('snp','allele','chrom','gene')]
```

```
##          snp allele chrom   gene
## 1: rs12243326      A    10 TCF7L2
## 2:  rs2237897      T    11  KCNQ1
```

```
report2 <- association.ext[p.value < 1e-4 & (pos >= report1[,pos][1] - 1000000 & pos <=report1[,pos][1
                           | (pos >= report1[,pos][2] - 1000000 & pos <=report1[,pos][2] + 1000000),
                           c('snp','gene')]
report2
```

```
##          snp     gene
## 1: rs10770141       TH
## 2: rs12243326   TCF7L2
## 3:   rs163184    KCNQ1
## 4:  rs2041139 CACNA2D4
## 5:  rs2237892    KCNQ1
## 6:  rs2237897    KCNQ1
## 7:   rs231362    KCNQ1
## 8:   rs391300     SMG6
## 9:  rs4523957     SMG6
```

Problem 2.e (8 points)

Build a weighted genetic risk score that includes all SNPs with p-value $< 10^{-4}$, a score with all SNPs with p-value $< 10^{-3}$, and a score that only includes SNPs on the FTO gene (hint: ensure that the ordering of SNPs is respected). Add the three scores as columns to the gdm.dt data table. Fit the three scores in separate logistic regression models to test their association with gestational diabetes, and for each report odds ratio, 95% confidence interval and p-value.

```
# Genetic risk score
snps.grs.3   <- association.ext[p.value < 1e-3]
snps.grs.4   <- association.ext[p.value < 1e-4]
snps.grs.FTO <- association.ext[gene == 'FTO']

gdm.grs.3   <- gdm.dt[, .SD, .SDcols = snps.grs.3$snp_full]
gdm.grs.4   <- gdm.dt[, .SD, .SDcols = snps.grs.4$snp_full]
gdm.grs.FTO <- gdm.dt[, .SD, .SDcols = snps.grs.FTO$snp_full]

weighted.score.3   <- as.matrix(gdm.grs.3) %*% snps.grs.3$beta
weighted.score.4   <- as.matrix(gdm.grs.4) %*% snps.grs.4$beta
weighted.score.FTO <- as.matrix(gdm.grs.FTO) %*% snps.grs.FTO$beta

gdm.dt[,weighted.risk.3:=weighted.score.3]
gdm.dt[,weighted.risk.4:=weighted.score.4]
gdm.dt[,weighted.risk.FTO:=weighted.score.FTO]

risk.3.logistic   <- glm(pheno ~ weighted.risk.3, data=gdm.dt, family='binomial')
risk.4.logistic   <- glm(pheno ~ weighted.risk.4, data=gdm.dt, family='binomial')
risk.FTO.logistic <- glm(pheno ~ weighted.risk.FTO, data=gdm.dt, family='binomial')

report.3 <- NULL
report.3 <- transpose(as.data.table(coef(summary(risk.3.logistic))[-1, -3]))
report.3 <- cbind(report.3,
                  exp(coef(risk.3.logistic))[2],
                  exp(confint(risk.3.logistic, level=0.95)[2,1]),
                  exp(confint(risk.3.logistic, level=0.95)[2,2]),
                  "p.value 1e-3")
```

```
## Waiting for profiling to be done...
## Waiting for profiling to be done...
```

```
colnames(report.3) <- c("beta","std.error", "p.value", "odds ratio", "odds ratio 2.5%", "odds ratio 97.
report.4 <- NULL
report.4 <- transpose(as.data.table(coef(summary(risk.4.logistic))[-1, -3]))
report.4 <- cbind(report.4,
                  exp(coef(risk.4.logistic))[2],
                  exp(confint(risk.4.logistic, level=0.95)[2,1]),
                  exp(confint(risk.4.logistic, level=0.95)[2,2]),
                  "p.value 1e-4")
```

```
## Waiting for profiling to be done...
## Waiting for profiling to be done...
```

```
colnames(report.4) <- c("beta","std.error", "p.value", "odds ratio", "odds ratio 2.5%", "odds ratio 97.
report.FTO <- NULL
report.FTO <- transpose(as.data.table(coef(summary(risk.3.logistic))[-1, -3]))
report.FTO <- cbind(report.FTO,
                  exp(coef(risk.FTO.logistic))[2],
                  exp(confint(risk.FTO.logistic, level=0.95)[2,1]),
                  exp(confint(risk.FTO.logistic, level=0.95)[2,2]),
                  "gene FTO")
```

```
## Waiting for profiling to be done...
## Waiting for profiling to be done...
```

```
colnames(report.FTO) <- c("beta","std.error", "p.value", "odds ratio", "odds ratio 2.5%", "odds ratio 9

report <- rbind(report.3, report.4, report.FTO)
report[,3:7]
```

```
##         p.value odds ratio odds ratio 2.5% odds ratio 97.5%   identifier
## 1: 7.813912e-09   1.451854       1.2814405         1.651126 p.value 1e-3
## 2: 2.759214e-08   2.729432       1.9243530         3.911052 p.value 1e-4
## 3: 7.813912e-09   1.413857       0.8191201         2.452615     gene FTO
```

Problem 2.f (4 points)

File GDM.test.txt (available from the accompanying zip folder on Learn) contains genotypes of another 40 pregnant women with and without gestational diabetes (assume that the reference allele is the same one that was specified in file GDM.raw.txt). Read the file into variable gdm.test. For the set of patients in gdm.test, compute the three genetic risk scores as defined in problem 2.e using the same set of SNPs and corresponding weights. Add the three scores as columns to gdm.test (hint: use the same columnnames as before).

```r
# Enter code here.
gdm.test <- fread("data/GDM.test.txt")
X <- as.data.table(gdm.test[,4:ncol(gdm.test)])
target <- gdm.test$pheno

association.test <- data.table(univ.glm.test(x=X,y=target, ordering=TRUE), key=c('snp_full'))

pk <- c('snp')
setnames(association.test, "snp_full", "snp")
association.test.ext <- merge(association.test,
                              gdm.annot.dt,
                              by=pk,
                              all=TRUE)[order(snp)]

snps.grs.3   <- association.test.ext[p.value < 1e-3]
snps.grs.4   <- association.test.ext[p.value < 1e-4]
snps.grs.FTO <- association.test.ext[gene == 'FTO']

gdm.grs.3   <- gdm.test[, .SD, .SDcols = snps.grs.3$snp]
gdm.grs.4   <- gdm.test[, .SD, .SDcols = snps.grs.4$snp]
gdm.grs.FTO <- gdm.test[, .SD, .SDcols = snps.grs.FTO$snp]

weighted.score.3   <- as.matrix(gdm.grs.3) %*% snps.grs.3$beta
weighted.score.4   <- as.matrix(gdm.grs.4) %*% snps.grs.4$beta
weighted.score.FTO <- as.matrix(gdm.grs.FTO) %*% snps.grs.FTO$beta

gdm.test[,weighted.risk.3:=weighted.score.3]
gdm.test[,weighted.risk.4:=weighted.score.4]
gdm.test[,weighted.risk.FTO:=weighted.score.FTO]
```

Problem 2.g (4 points)

Use the logistic regression models fitted in problem 2.e to predict the outcome of patients in gdm.test. Compute the test log-likelihood for the predicted probabilities from the three genetic risk score models.

```r
# Enter code here.
pred.3   <- predict(risk.3.logistic   , newdata=gdm.test, type='response')
pred.4   <- predict(risk.4.logistic   , newdata=gdm.test, type='response')
pred.FTO <- predict(risk.FTO.logistic, newdata=gdm.test, type='response')

pred.3
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
## NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
## 27 28 29 30 31 32 33 34 35 36 37 38 39 40
## NA NA NA NA NA NA NA NA NA NA NA NA NA NA
```

```r
pred.4
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
## NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
## 27 28 29 30 31 32 33 34 35 36 37 38 39 40
## NA NA NA NA NA NA NA NA NA NA NA NA NA NA
```

```r
pred.FTO
```

```
##         1         2         3         4         5         6         7         8 
## 0.5130166 0.7138790 0.5130166 0.7138790 0.5130166 0.5130166 0.5432992 0.5130166 
##         9        10        11        12        13        14        15        16 
## 0.6615708 0.5130166 0.7138790 0.5130166 0.5130166 0.7138790 0.5130166 0.5130166 
##        17        18        19        20        21        22        23        24 
## 0.5130166 0.7138790 0.5130166 0.5958924 0.5130166 0.5130166 0.6859098 0.8552665 
##        25        26        27        28        29        30        31        32 
## 0.7138790 0.5653122 0.7138790 0.5130166 0.5130166 0.7678520 0.7138790 0.5130166 
##        33        34        35        36        37        38        39        40 
## 0.5827327 0.5130166 0.5130166 0.5130166 0.5130166 0.7138790 0.7138790 0.5130166
```

Problem 2.h (4points)

File GDM.study2.txt (available from the accompanying zip folder on Learn) contains the summary statistics from a different study on the same set of SNPs. Perform a meta-analysis with the results obtained in problem 2.c (hint: remember that the effect alleles should correspond) and produce a summary of the meta-analysis results for the set of SNPs with meta-analysis p-value $< 10^{-4}$ sorted by increasing p-value.

```
# Enter code here.
gdm.meta <- fread("data/GDM.study2.txt")

gdm.meta <- gdm.meta[snp %in% association.ext$snp]
association.ext <- association.ext[snp %in% gdm.meta$snp]

gdm.meta <- gdm.meta[order(snp, effect.allele)]
association.ext <- association.ext[order(snp, allele)]
all.equal(gdm.meta$snp, association.ext$snp)
```

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

```
effect <- association.ext$allele == gdm.meta$effect.allele
other <- association.ext$allele == gdm.meta$other.allele
table(effect, other)
```

```
##        other
## effect  FALSE TRUE
##   FALSE     2   27
##   TRUE    147    0
```

```
# clearning data that don't match
gdm.meta <- gdm.meta[-which((effect+other)==0),]
association.ext <- association.ext[-which((effect+other)==0),]

# gdm.meta <- gdm.meta[order(snp, effect.allele)]
# association.ext <- association.ext[order(snp, allele)]
# all.equal(gdm.meta$snp, association.ext$snp)
#
# effect <- association.ext$allele == gdm.meta$effect.allele
# other <- association.ext$allele == gdm.meta$other.allele
# table(effect, other)

beta1 <- association.ext$beta
beta2 <- gdm.meta$beta

weight.association.ext <- 1 / association.ext$std.error^2
weight.gdm.meta<- 1 / gdm.meta$se^2
```

```
head(weight.association.ext)
```

```
## [1] 95.95663 48.35218 51.12222 80.72417 32.39329 80.53452
```

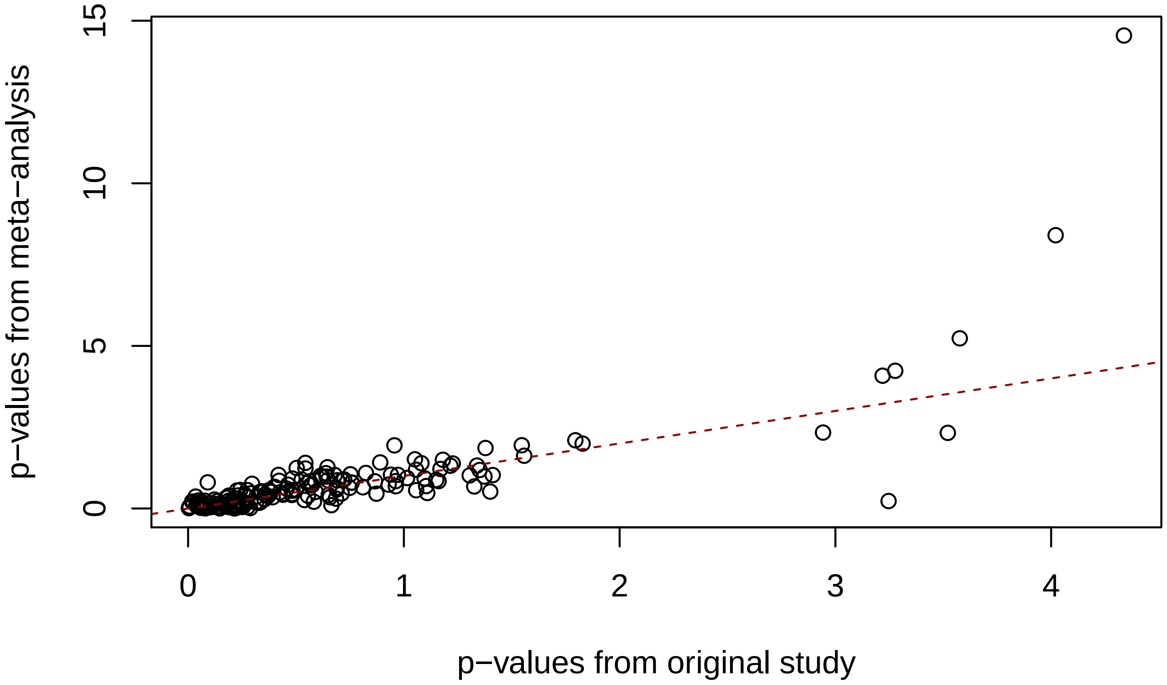
```
head(weight.gdm.meta)
```

```
## [1]  9.141470  6.919489  7.069651 10.430712  3.328963  9.751846
```

```
beta.ma <- (weight.association.ext * beta1 + weight.gdm.meta * beta2) / (weight.association.ext + weigh
se.ma <- sqrt(1 / (weight.association.ext + weight.gdm.meta))
pval.ma <- 2 * pnorm(abs(beta.ma / se.ma), lower.tail = FALSE)

cat('Meta analysis mean beta:', mean(beta.ma, na.rm=TRUE),
    '\nMeta analysis mean sd:', mean(se.ma, na.rm=TRUE),
    '\nMeta analysis mean p.value:', mean(pval.ma, na.rm=TRUE))
```

```
## Meta analysis mean beta: 0.01742904
## Meta analysis mean sd: 0.1293352
## Meta analysis mean p.value: 0.4029081
```

```
plot(-log10(association.ext$p.value), -log10(pval.ma),
     xlab = "p-values from original study",
     ylab = "p-values from meta-analysis"
     )
abline(a=0,b=1,col="red4", lty=2)
```



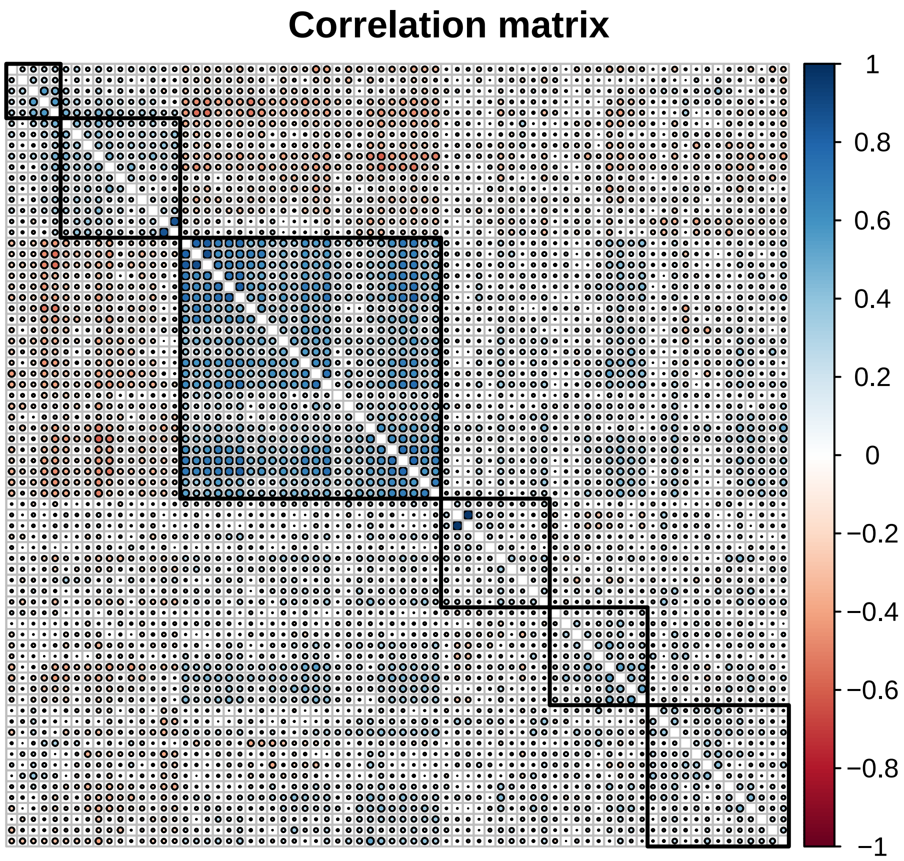
Problem 3 (33 points)

File nki.csv (available from the accompanying zip folder on Learn) contains data for 144 breast cancer patients. The dataset contains a binary outcome variable ("Event", indicating the insurgence of further complications after operation), covariates describing the tumour and the age of the patient, and gene expressions for 70 genes found to be prognostic of survival.

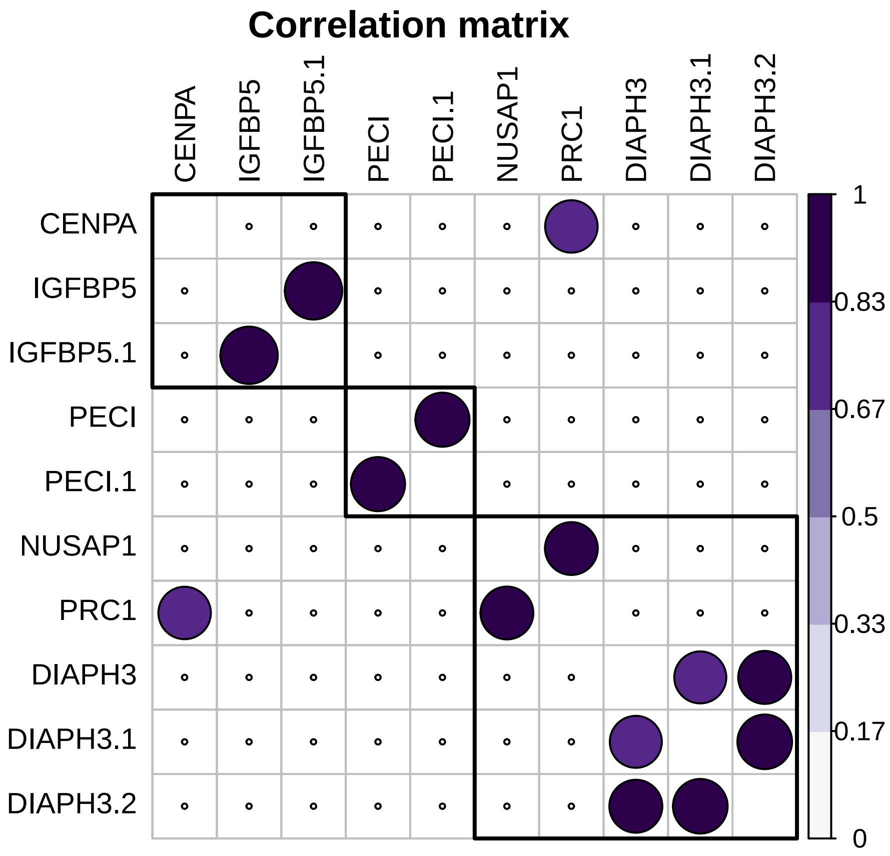
Problem 3.a (6 points)

Compute the matrix of correlations between the gene expression variables, and display it so that a block structure is highlighted. Discuss what you observe. Write some code to identify the unique pairs of (distinct) variables that have correlation coefficient greater than 0.80 in absolute value and report their correlation coefficients.

```
# Enter code here.
nki.dt <- fread("data/nki.csv")
numcols <- sapply(nki.dt, is.numeric)
cor.nki_full <- nki.dt[, ..numcols] %>% cor(use="pairwise.complete")

corrplot(cor.nki_full,
         order="hclust",
         addrect=6,
         diag=FALSE,
         tl.pos='n',
         tl.col="black",
         tl.cex = 0.9,
         outline=TRUE,
         title="Correlation matrix",
         cl.lim=c(-1, 1),
         mar=c(0,0,1.5,0))
```



```
cor.nki <- cor.nki_full*(abs(cor.nki_full)>0.8)
# remove rows where they only have autocorrelation present
cor.nki <- cor.nki[-which(abs(rowSums(cor.nki))==1),-which(abs(colSums(cor.nki))==1)]

corrplot(cor.nki,
         order="hclust",
         addrect=3,
         diag=FALSE,
         tl.col="black",
         tl.cex = 0.9,
         outline=TRUE,
         title="Correlation matrix",
         col = brewer.pal(n=11, name="PuOr"),
         cl.lim=c(0, 1),
         mar=c(0,0,1.5,0))
```



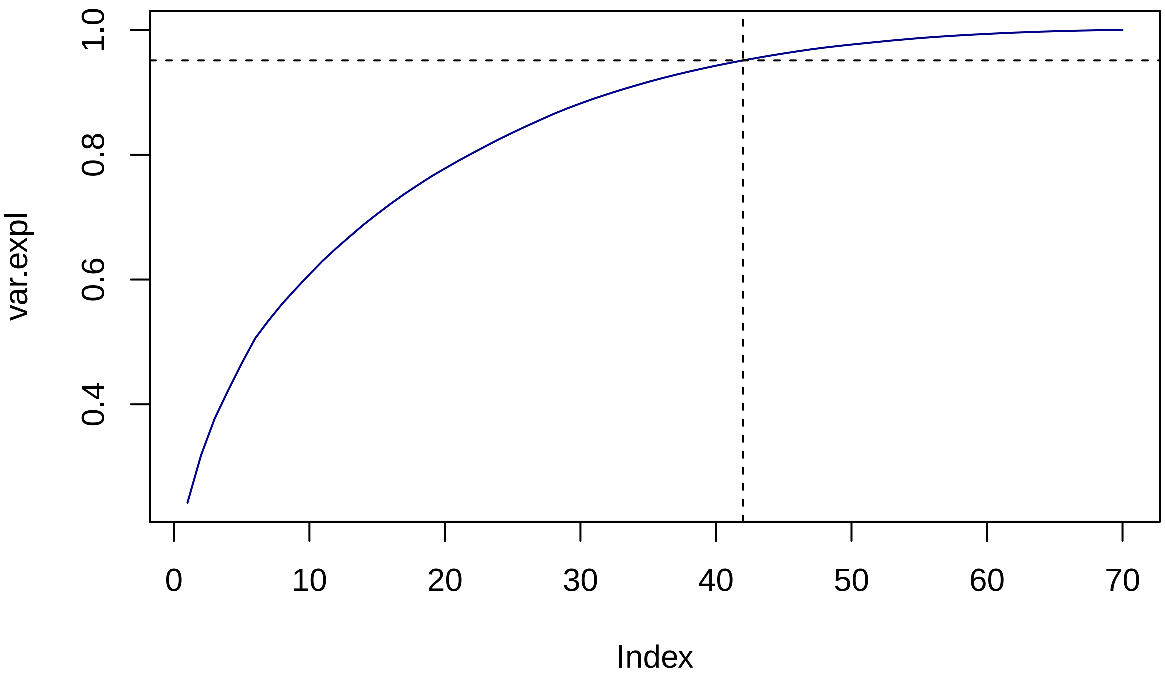
Problem 3.b (8 points)

Run PCA (only over the columns containing gene expressions), in order to derive a patient-wise summary of all gene expressions (dimensionality reduction). Decide which components to keep and justify your decision. Test if those principal components are associated with the outcome in unadjusted logistic regression models and in models adjusted for age, estrogen receptor and grade. Justify the difference in results between unadjusted and adjusted models.

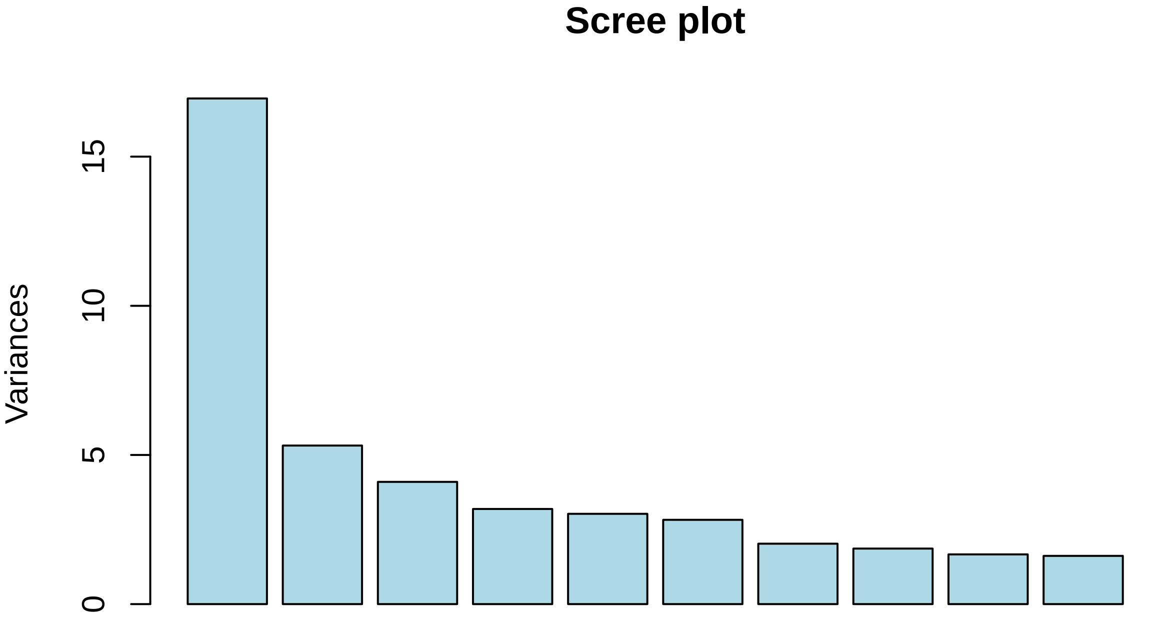
```
# Enter code here.
numcols[c('Event', 'Diam', 'LymphNodes', 'EstrogenReceptor', 'Grade', 'Age')] <- FALSE

pca.vars <- prcomp(nki.dt[, ..numcols], center=T, scale=T)
var.expl <- cumsum(pca.vars$sdev^2 / sum(pca.vars$sdev^2))

summary(pca.vars)
```

```
## Importance of components:
##                           PC1     PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     4.1171 2.30541 2.02437 1.78597 1.73982 1.68091 1.42309
## Proportion of Variance 0.2422 0.07593 0.05854 0.04557 0.04324 0.04036 0.02893
## Cumulative Proportion  0.2422 0.31808 0.37662 0.42219 0.46543 0.50580 0.53473
##                            PC8     PC9    PC10    PC11    PC12    PC13    PC14
## Standard deviation     1.36441 1.29119 1.2715 1.24741 1.18388 1.15101 1.13883
## Proportion of Variance 0.02659 0.02382 0.0231 0.02223 0.02002 0.01893 0.01853
## Cumulative Proportion  0.56132 0.58514 0.6082 0.63046 0.65049 0.66941 0.68794
##                           PC15    PC16    PC17    PC18    PC19    PC20    PC21
## Standard deviation     1.09473 1.07016 1.04187 1.00234 0.99086 0.94095 0.93322
## Proportion of Variance 0.01712 0.01636 0.01551 0.01435 0.01403 0.01265 0.01244
## Cumulative Proportion  0.70506 0.72142 0.73693 0.75128 0.76531 0.77796 0.79040
##                           PC22    PC23    PC24    PC25    PC26    PC27    PC28
## Standard deviation     0.90727 0.89675 0.88859 0.86019 0.84462 0.82782 0.82368
## Proportion of Variance 0.01176 0.01149 0.01128 0.01057 0.01019 0.00979 0.00969
## Cumulative Proportion  0.80216 0.81364 0.82492 0.83549 0.84569 0.85548 0.86517
##                           PC29    PC30    PC31    PC32    PC33    PC34   PC35
## Standard deviation     0.78694 0.75594 0.73942 0.70569 0.69414 0.67129 0.6639
## Proportion of Variance 0.00885 0.00816 0.00781 0.00711 0.00688 0.00644 0.0063
## Cumulative Proportion  0.87401 0.88218 0.88999 0.89710 0.90399 0.91042 0.9167
##                           PC36    PC37    PC38    PC39    PC40    PC41    PC42
## Standard deviation     0.63815 0.61964 0.59947 0.58447 0.57195 0.55097 0.53820
## Proportion of Variance 0.00582 0.00549 0.00513 0.00488 0.00467 0.00434 0.00414
## Cumulative Proportion  0.92254 0.92802 0.93316 0.93804 0.94271 0.94705 0.95118
##                           PC43    PC44    PC45    PC46    PC47    PC48    PC49
## Standard deviation     0.52029 0.51211 0.49533 0.48712 0.47079 0.44565 0.41879
## Proportion of Variance 0.00387 0.00375 0.00351 0.00339 0.00317 0.00284 0.00251
## Cumulative Proportion  0.95505 0.95880 0.96230 0.96569 0.96886 0.97170 0.97420
##                           PC50    PC51   PC52    PC53    PC54    PC55    PC56
## Standard deviation     0.40556 0.39328 0.3925 0.38502 0.36669 0.36205 0.33734
## Proportion of Variance 0.00235 0.00221 0.0022 0.00212 0.00192 0.00187 0.00163
## Cumulative Proportion  0.97655 0.97876 0.9810 0.98308 0.98500 0.98687 0.98850
##                           PC57    PC58    PC59    PC60    PC61    PC62    PC63
## Standard deviation     0.32150 0.30744 0.28898 0.28186 0.27274 0.25622 0.24118
## Proportion of Variance 0.00148 0.00135 0.00119 0.00113 0.00106 0.00094 0.00083
## Cumulative Proportion  0.98998 0.99133 0.99252 0.99365 0.99472 0.99565 0.99649
##                           PC64    PC65    PC66    PC67    PC68   PC69    PC70
## Standard deviation     0.23024 0.21442 0.19886 0.19371 0.17927 0.1677 0.09833
## Proportion of Variance 0.00076 0.00066 0.00056 0.00054 0.00046 0.0004 0.00014
## Cumulative Proportion  0.99724 0.99790 0.99846 0.99900 0.99946 0.9999 1.00000
```

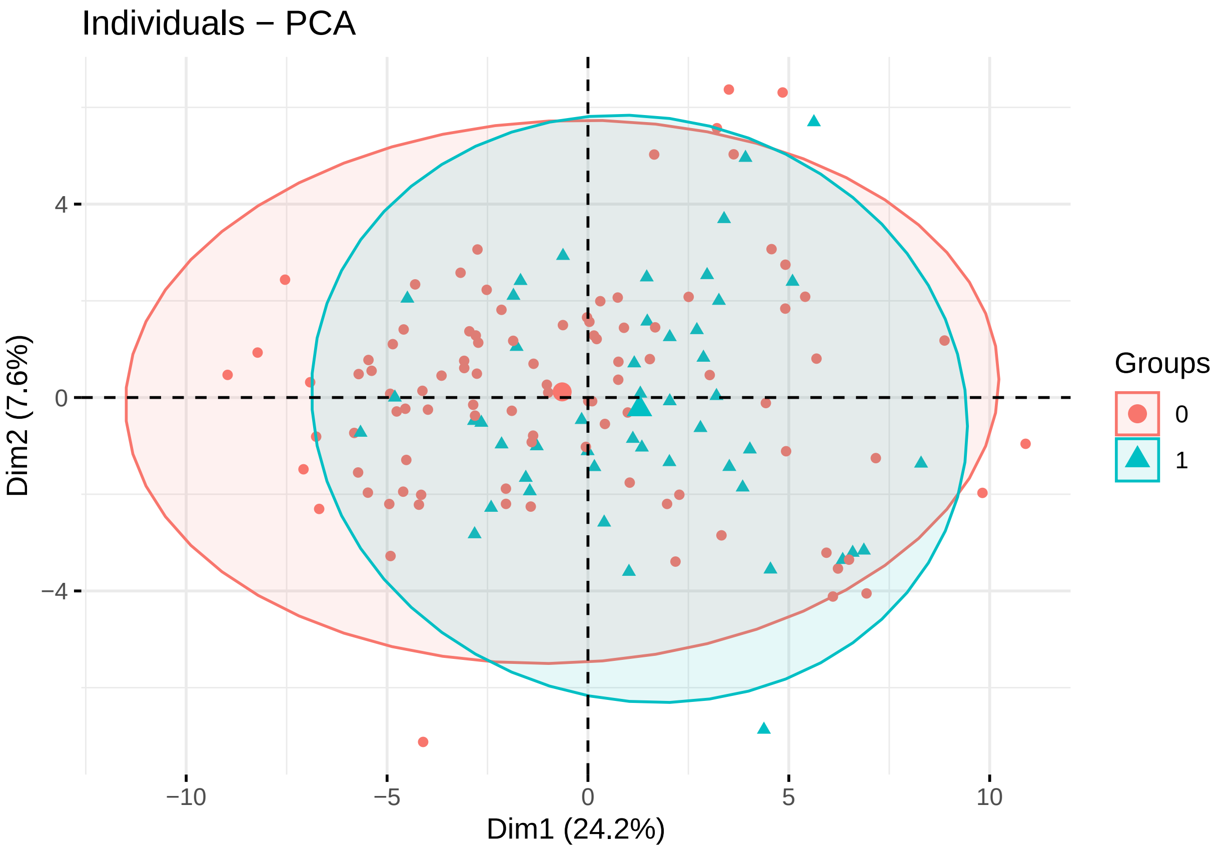
```r
# cumulative variance explained plot
plot(var.expl, type='l', col='blue4')

# calculate first 90%, 95% variance explained automatically.
abline(h=0.9511847, v=42, lty=2)
```



```r
# scree plot
screeplot(pca.vars, main="Scree plot", col='lightblue')
```



```r
# PCA plot
fviz_pca_ind(pca.vars, geom='point',
             habillage=nki.dt$Event,
             addEllipses=T,
             ellipse.level=0.95)
```



```
# PCA biplot
# fviz_pca_biplot(pca.vars, geom='point', repel = T)

pca.embeddings.95 <- as.data.frame(pca.vars$x[,1:42])

# all PCs are othogonal, checking that the correlation matrix is all white except autocorrelations.
# res1 <- cor(pca.embeddings.95, method='pearson')
# corrplot(res1, method= "color", order = "hclust", tl.pos = 'n')

## Models
#
# beta.Z <- as.matrix(lmodel$coefficients[2:123])
# V <- as.matrix(crimeData.pca1$rotation)
# beta.X <- V %*% beta.Z
# beta.X

data.embeded     <- as.data.frame(cbind(nki.dt[,1], pca.embeddings.95))
data.embeded.adj <- as.data.frame(cbind(nki.dt[,1], nki.dt[,4:6], pca.embeddings.95))

model     <- glm(Event~., data=data.embeded     , family='binomial')
model.adj <- glm(Event~., data=data.embeded.adj, family='binomial')
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(model)
```

```
##
## Call:
```

```
## glm(formula = Event ~ ., family = "binomial", data = data.embeded)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5622  -0.4036  -0.0104   0.0675   3.4493
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.69481    0.79564  -3.387 0.000707 ***
## PC1          0.55978    0.18355   3.050 0.002290 **
## PC2         -0.06842    0.15515  -0.441 0.659221
## PC3          0.62519    0.24568   2.545 0.010935 *
## PC4         -1.25179    0.48186  -2.598 0.009382 **
## PC5         -0.08599    0.23513  -0.366 0.714565
## PC6          0.74780    0.33238   2.250 0.024461 *
## PC7          0.02495    0.34292   0.073 0.941998
## PC8          0.04694    0.27135   0.173 0.862653
## PC9          0.47010    0.35968   1.307 0.191216
## PC10        -1.04938    0.48219  -2.176 0.029536 *
## PC11        -1.54897    0.53098  -2.917 0.003532 **
## PC12         0.07232    0.43895   0.165 0.869132
## PC13        -0.36607    0.36062  -1.015 0.310046
## PC14        -0.31123    0.40088  -0.776 0.437543
## PC15        -0.47155    0.37796  -1.248 0.212176
## PC16        -1.18771    0.52651  -2.256 0.024081 *
## PC17        -0.81375    0.38486  -2.114 0.034479 *
## PC18        -0.75850    0.46976  -1.615 0.106384
## PC19         3.17912    0.99653   3.190 0.001422 **
## PC20        -0.21795    0.46738  -0.466 0.640984
## PC21        -0.80290    0.53328  -1.506 0.132172
## PC22         0.62281    0.64242   0.969 0.332307
## PC23         1.97124    0.75226   2.620 0.008782 **
## PC24        -1.54956    0.66614  -2.326 0.020008 *
## PC25        -0.36892    0.49271  -0.749 0.454009
## PC26         0.11901    0.56651   0.210 0.833608
## PC27         0.78966    0.48303   1.635 0.102089
## PC28        -1.43775    0.70439  -2.041 0.041239 *
## PC29        -0.42265    0.59864  -0.706 0.480178
## PC30        -0.52763    0.55293  -0.954 0.339966
## PC31         1.75629    0.80495   2.182 0.029120 *
## PC32        -0.28186    0.51983  -0.542 0.587669
## PC33        -2.00846    0.76640  -2.621 0.008777 **
## PC34        -0.92011    0.60905  -1.511 0.130861
## PC35        -0.41835    0.69394  -0.603 0.546598
## PC36        -0.18634    0.67284  -0.277 0.781828
## PC37         0.70620    0.70401   1.003 0.315806
## PC38         2.15027    0.98875   2.175 0.029651 *
## PC39        -1.56093    0.75960  -2.055 0.039884 *
## PC40         1.09829    0.66805   1.644 0.100174
## PC41         2.37649    1.07157   2.218 0.026571 *
## PC42        -1.06355    0.86688  -1.227 0.219872
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 183.316  on 143  degrees of freedom
## Residual deviance:  71.416  on 101  degrees of freedom
## AIC: 157.42
##
## Number of Fisher Scoring iterations: 9
```

```
summary(model.adj)
```

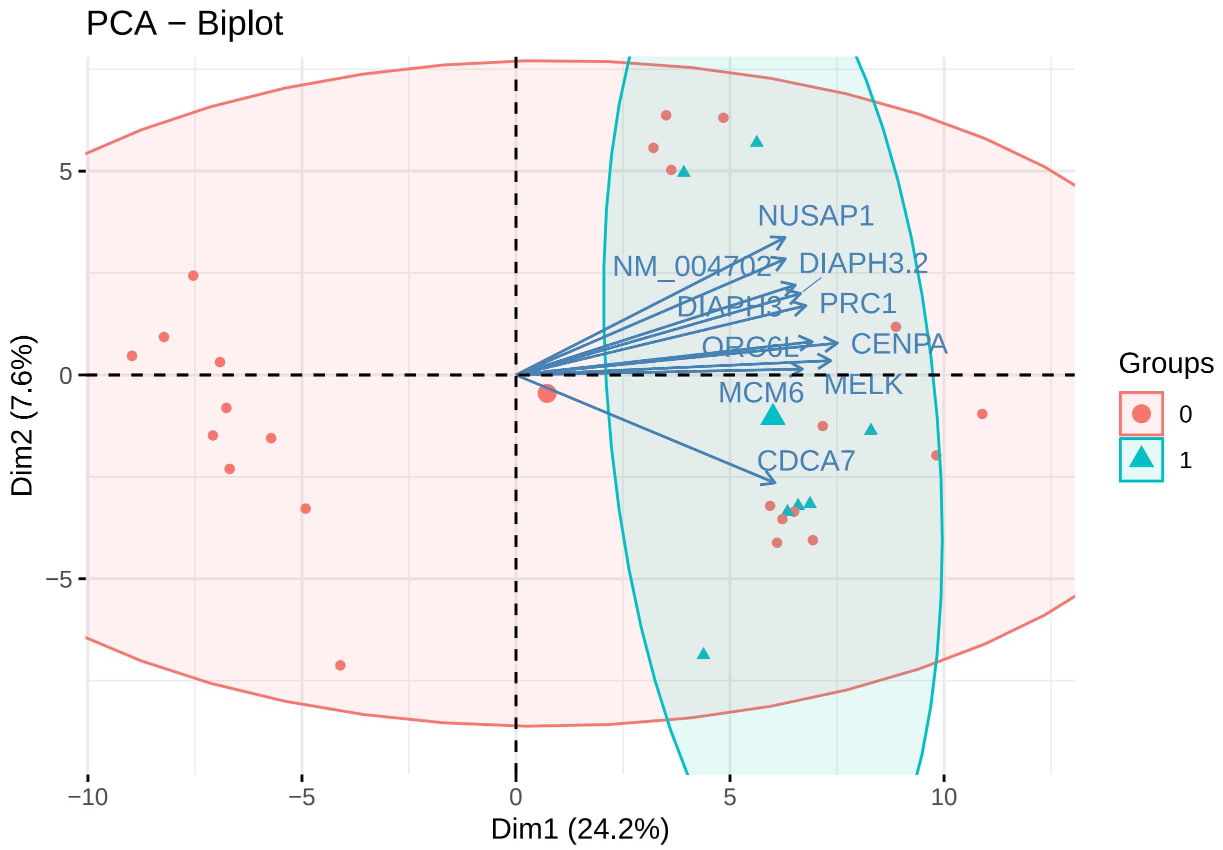
```
##
## Call:
## glm(formula = Event ~ ., family = "binomial", data = data.embeded.adj)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.95896  -0.31596  -0.00462   0.02414   2.73979
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)                8.9235     5.4584   1.635  0.10209
## EstrogenReceptorPositive   0.9137     2.9675   0.308  0.75816
## GradePoorly diff           0.8489     1.3982   0.607  0.54377
## GradeWell diff             0.9822     1.2850   0.764  0.44465
## Age                       -0.3050     0.1248  -2.443  0.01455 *
## PC1                        0.7326     0.2973   2.464  0.01372 *
## PC2                       -0.1089     0.2947  -0.369  0.71182
## PC3                        0.7543     0.3289   2.293  0.02182 *
## PC4                       -1.5600     0.5964  -2.616  0.00890 **
## PC5                       -0.1001     0.2996  -0.334  0.73830
## PC6                        0.8508     0.4448   1.913  0.05579 .
## PC7                        0.2613     0.4130   0.633  0.52690
## PC8                        0.2602     0.3506   0.742  0.45802
## PC9                        0.4646     0.4486   1.036  0.30041
## PC10                      -1.3649     0.5977  -2.284  0.02239 *
## PC11                      -1.6256     0.5543  -2.933  0.00336 **
## PC12                       0.1567     0.5411   0.290  0.77217
## PC13                      -0.2706     0.4454  -0.607  0.54354
## PC14                      -0.7907     0.5492  -1.440  0.14993
## PC15                      -0.5473     0.4637  -1.180  0.23792
## PC16                      -1.4157     0.5943  -2.382  0.01721 *
## PC17                      -0.6467     0.4489  -1.441  0.14970
## PC18                      -1.1272     0.5916  -1.905  0.05673 .
## PC19                       4.3515     1.3403   3.247  0.00117 **
## PC20                      -0.2178     0.6464  -0.337  0.73613
## PC21                      -1.1895     0.7461  -1.594  0.11087
## PC22                       0.2722     0.8581   0.317  0.75109
## PC23                       2.4944     0.9286   2.686  0.00723 **
## PC24                      -1.5903     0.7029  -2.262  0.02367 *
## PC25                      -0.2657     0.5461  -0.487  0.62654
## PC26                       0.6941     0.7415   0.936  0.34921
## PC27                       0.7377     0.5655   1.305  0.19205
## PC28                      -2.3176     0.9264  -2.502  0.01236 *
## PC29                      -0.5082     0.7014  -0.725  0.46867
## PC30                      -0.2429     0.6660  -0.365  0.71532
## PC31                         1.9403     0.9225   2.103  0.03543 *
## PC32                        -0.3864     0.5788  -0.668  0.50433
## PC33                        -2.0100     0.8543  -2.353  0.01864 *
## PC34                        -1.6495     0.8443  -1.954  0.05073 .
## PC35                        -0.8643     1.0053  -0.860  0.38990
## PC36                        -0.5447     0.8716  -0.625  0.53201
## PC37                         1.0922     0.9860   1.108  0.26797
## PC38                         2.3179     1.1272   2.056  0.03976 *
## PC39                        -1.7144     0.8660  -1.980  0.04773 *
## PC40                         1.0340     0.8062   1.283  0.19963
## PC41                         4.0163     1.5948   2.518  0.01179 *
## PC42                        -1.6523     1.0158  -1.627  0.10384
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 183.316  on 143  degrees of freedom
## Residual deviance:  63.498  on  97  degrees of freedom
## AIC: 157.5
##
## Number of Fisher Scoring iterations: 9
```

Problem 3.c (8 points)

Use plots to compare with the correlation structure observed in problem 2.a and to examine how well the dataset may explain your outcome. Discuss your findings and suggest any further steps if needed.

```
# Enter code here.
fviz_pca_biplot(pca.vars,
                repel=T,
                select.var=list(contrib=10),
                select.ind = list(contrib=30),
                label="var",
                habillage = nki.dt$Event,
                addEllipses=TRUE,
                ellipse.level=0.90,
                xlim=c(-9,12),
                ylim=c(-9,7))
```

```
## Coordinate system already present. Adding new coordinate system, which will replace the existing one
```



```
pc1.rotations.top <- sort(pca.vars$rotation[,1], decreasing=TRUE)[1:10]
pc2.rotations.top <- sort(pca.vars$rotation[,2], decreasing=TRUE)[1:10]

pc1.rotations.top
```

```
##     CENPA      MELK     ORC6L      PRC1      MCM6  DIAPH3.2    DIAPH3 NM_004702
## 0.2130018 0.2086196 0.1963327 0.1920987 0.1898039 0.1885517 0.1851385 0.1784957
##    NUSAP1  C16orf61
## 0.1782456 0.1773431
```

```
pc2.rotations.top
```

```
##      PECI    PECI.1      ECT2    NUSAP1   RTN4RL1 NM_004702    SERF1A    SCUBE2
## 0.2869100 0.2735517 0.1741105 0.1707437 0.1631555 0.1442509 0.1439098 0.1184746
##       DTL    DIAPH3
## 0.1135092 0.1116969
```

Problem 3.d (11 points)

Based on the models we examined in the labs, fit an appropriate model with the aim to provide the most accurate prognosis you can for patients. Discuss and justify your decisions.

```
# Enter code here.
data.full <- nki.dt
data.full[data.full$Diam == unique(nki.dt$Diam)[1], ]$Diam <- 0
data.full[data.full$Diam == unique(nki.dt$Diam)[2], ]$Diam <- 1
data.full$Diam <- as.factor(data.full$Diam)
data.full[data.full$LymphNodes == unique(nki.dt$LymphNodes)[1], ]$LymphNodes <- 0
data.full[data.full$LymphNodes == unique(nki.dt$LymphNodes)[2], ]$LymphNodes <- 1
data.full$LymphNodes <- as.factor(data.full$LymphNodes)
data.full[data.full$EstrogenReceptor == unique(nki.dt$EstrogenReceptor)[1], ]$EstrogenReceptor <- 0
data.full[data.full$EstrogenReceptor == unique(nki.dt$EstrogenReceptor)[2], ]$EstrogenReceptor <- 1
data.full$EstrogenReceptor <- as.factor(data.full$EstrogenReceptor)
data.full[data.full$Grade == unique(nki.dt$Grade)[1], ]$Grade <- 0
data.full[data.full$Grade == unique(nki.dt$Grade)[2], ]$Grade <- 1
data.full[data.full$Grade == unique(nki.dt$Grade)[3], ]$Grade <- 2
data.full$Grade <- as.factor(data.full$Grade)

ind <- createDataPartition(data.full$Event, p=0.85, list=FALSE)
data.train <- data.full[ind,]
data.test <- data.full[-ind,]

X.train <- data.train[, !c("Event"), with=FALSE]
X.test  <- data.test [, !c("Event"), with=FALSE]
y.train <- data.train[,  c("Event"), with=FALSE]
y.test  <- data.test [,  c("Event"), with=FALSE]

pca.vars.full <- prcomp(data.full[, ..numcols], center=T, scale=T)
pca.embeddings.95.train <- as.data.frame(pca.vars.full$x[ind,1:42])
pca.embeddings.95.test <- as.data.frame(pca.vars.full$x[-ind,1:42])
pca.embeddings.adj.train <- as.data.frame(cbind(data.train[,1], data.train[,4:6], pca.embeddings.95.trai
pca.embeddings.adj.test <- as.data.frame(cbind(data.test[,1], data.test[,4:6], pca.embeddings.95.test))
data.embeded.train     <- as.data.frame(cbind(data.train[,1], pca.embeddings.95.train))
data.embeded.adj.train <- as.data.frame(cbind(data.train[,1:6], pca.embeddings.95.train))
data.embeded.test     <- as.data.frame(cbind(data.test[,1], pca.embeddings.95.test))
data.embeded.adj.test <- as.data.frame(cbind(data.test[,1:6], pca.embeddings.95.test))


logistic.simple  <- glm(Event~Diam + LymphNodes + EstrogenReceptor + Grade + Age,data=data.train, famil
logistic.full    <- glm(Event~., data=data.train, family='binomial')
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
logistic.pca     <- glm(Event~., data=data.embeded.train, family='binomial')
```

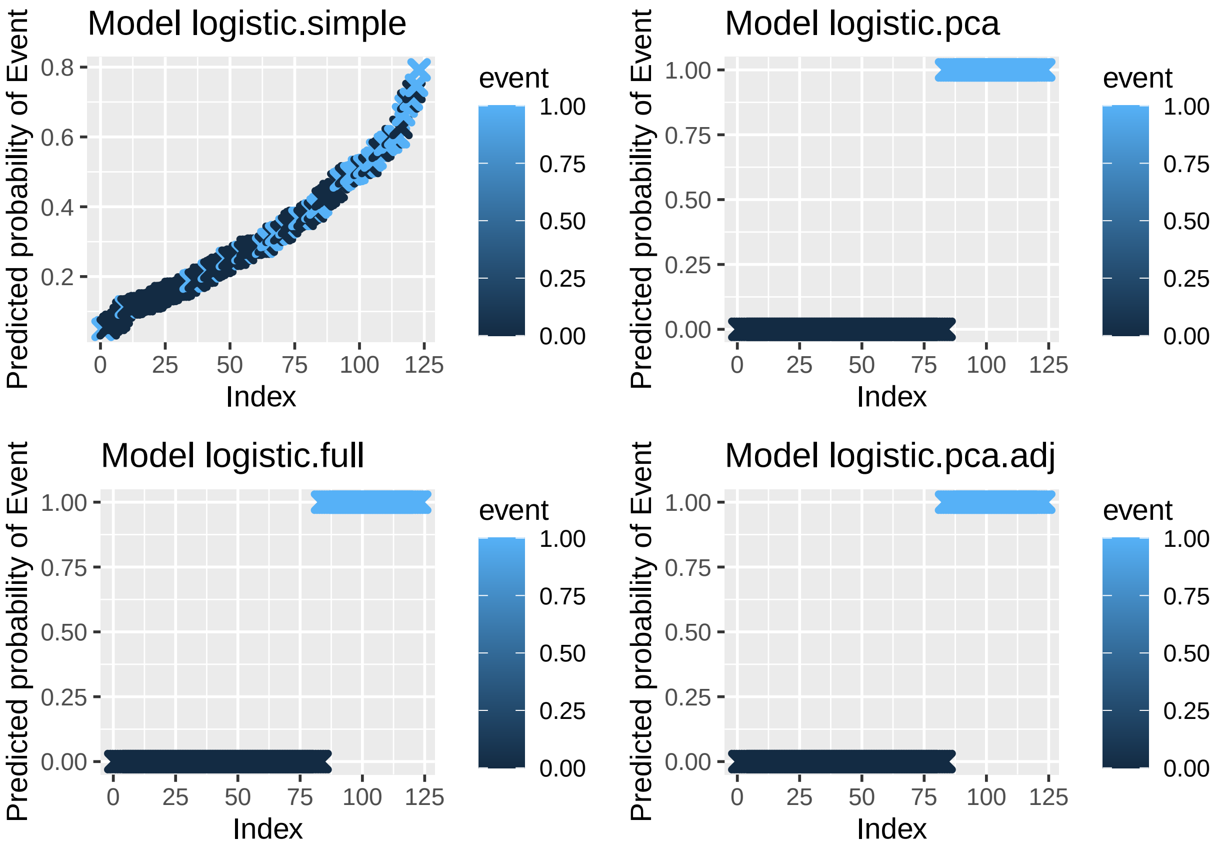
```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
logistic.pca.adj <- glm(Event~., data=data.embeded.adj.train, family='binomial')
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
# https://github.com/StatQuest/logistic_regression_demo/blob/master/logistic_regression_demo.R
#TODO: make it a function
predicted.data <- data.frame(
  probability.of.event=logistic.simple$fitted.values,
  event=data.train$Event)
predicted.data <- predicted.data[
  order(predicted.data$probability.of.event, decreasing=FALSE),]
predicted.data$rank <- 1:nrow(predicted.data)
```

```
p1 <- ggplot(data=predicted.data, aes(x=rank, y=probability.of.event)) +
  geom_point(aes(color=event), alpha=1, shape=4, stroke=2) +
  xlab("Index") +
  ylab("Predicted probability of Event") +
  ggtitle('Model logistic.simple')

predicted.data <- data.frame(
  probability.of.event=logistic.pca$fitted.values,
  event=data.train$Event)
predicted.data <- predicted.data[
  order(predicted.data$probability.of.event, decreasing=FALSE),]
predicted.data$rank <- 1:nrow(predicted.data)
p2 <- ggplot(data=predicted.data, aes(x=rank, y=probability.of.event)) +
  geom_point(aes(color=event), alpha=1, shape=4, stroke=2) +
  xlab("Index") +
  ylab("Predicted probability of Event") +
  ggtitle('Model logistic.pca')

predicted.data <- data.frame(
  probability.of.event=logistic.full$fitted.values,
  event=data.train$Event)
predicted.data <- predicted.data[
  order(predicted.data$probability.of.event, decreasing=FALSE),]
predicted.data$rank <- 1:nrow(predicted.data)
p3 <- ggplot(data=predicted.data, aes(x=rank, y=probability.of.event)) +
  geom_point(aes(color=event), alpha=1, shape=4, stroke=2) +
  xlab("Index") +
  ylab("Predicted probability of Event") +
  ggtitle('Model logistic.full')

predicted.data <- data.frame(
  probability.of.event=logistic.pca.adj$fitted.values,
  event=data.train$Event)
predicted.data <- predicted.data[
  order(predicted.data$probability.of.event, decreasing=FALSE),]
predicted.data$rank <- 1:nrow(predicted.data)
p4 <- ggplot(data=predicted.data, aes(x=rank, y=probability.of.event)) +
  geom_point(aes(color=event), alpha=1, shape=4, stroke=2) +
  xlab("Index") +
  ylab("Predicted probability of Event") +
  ggtitle('Model logistic.pca.adj')

grid.arrange(p1, p2, p3, p4, ncol=2, nrow=2)
```



```
pred.logistic.simple  <- predict(logistic.simple , newdata=data.test, type='response')
pred.logistic.full    <- predict(logistic.full   , newdata=data.test, type='response')
pred.logistic.pca     <- predict(logistic.pca    , newdata=data.embeded.test, type='response')
pred.logistic.pca.adj <- predict(logistic.pca.adj, newdata=data.embeded.adj.test, type='response')

log.loss.logistic.simple  <- sum(log(1+exp(- data.test$Event %*% pred.logistic.simple)))
log.loss.logistic.full    <- sum(log(1+exp(- data.test$Event %*% pred.logistic.full)))
log.loss.logistic.pca     <- sum(log(1+exp(- data.test$Event %*% pred.logistic.pca)))
log.loss.logistic.pca.adj <- sum(log(1+exp(- data.test$Event %*% pred.logistic.pca.adj)))

names    <- c("logistic.simple", "logistic.full", "logistic.pca", "logistic.pca.adj")
log.loss <- c(log.loss.logistic.simple, log.loss.logistic.full, log.loss.logistic.pca, log.loss.logistic

final.results <- cbind(names, signif(log.loss,6))
final.results
```

```
##      names
## [1,] "logistic.simple"  "0.0630591"
## [2,] "logistic.full"    "0.0181499"
## [3,] "logistic.pca"     "0.0181499"
## [4,] "logistic.pca.adj" "0.00671535"
```