Assignment 2

Johnny Lee, s1687781

5th April 2022

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.

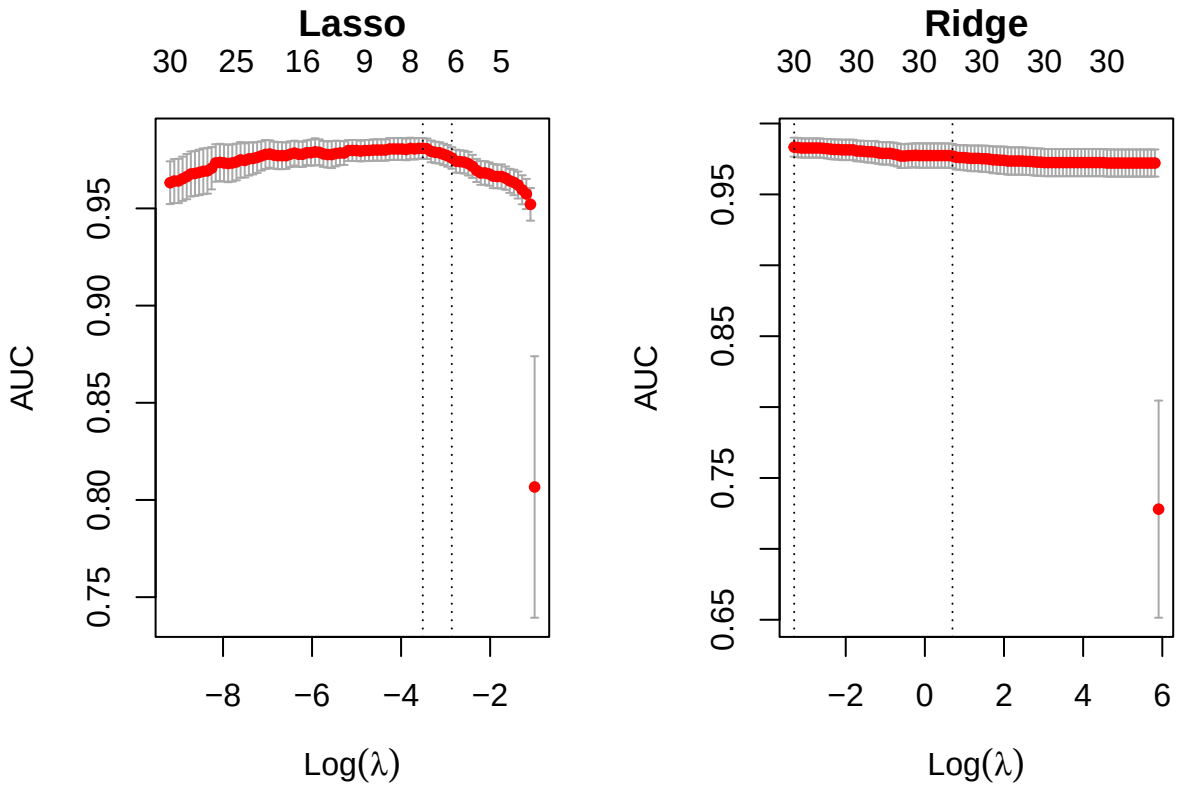
Answer

```
breast <- read.csv("data_assignment2/wdbc2.csv")
```

```
set.seed(984065)
#redefining diagnosis from benign and malignant to 0 and 1
breast$diagnosis <- ifelse(breast$diagnosis == "benign", 0, 1)
#Splitting into training and testing data set
split.index <- createDataPartition(breast$diagnosis,
                                   p = .7, list = TRUE)$Resample1
#train and test data sets
train.breast <- breast[split.index, ]
test.breast <- breast[-split.index, ]

#define explanatory and response variable matrix
biomarkers.x <- as.matrix(subset(train.breast, select = -c(id, diagnosis)))
biomarkers.y <- as.matrix(subset(train.breast, select = c(diagnosis)))
```

```
set.seed(984065)
#Fitting ridge regression on training data
fit.lasso <- cv.glmnet(biomarkers.x, biomarkers.y , alpha = 1,
                       family = "binomial", type.measure = "auc")
#Fitting lasso regression on training data
fit.ridge <- cv.glmnet(biomarkers.x, biomarkers.y , alpha = 0,
                       family = "binomial", type.measure = "auc")

#plotting lambda values for Lasso and Ridge Regression
par(mfrow=c(1,2), mar=c(4,4,5,2))
plot(fit.lasso, main="Lasso")
```

```
plot(fit.ridge, main="Ridge")
```




```
#computing optimal lambda values for Lasso and Ridge Regression
cat("The optimal value of lambda for Lasso is:", fit.ridge$lambda.min)
```

```
## The optimal value of lambda for Lasso is: 0.03677378
```

```
cat("The optimal value of lambda for Ridge is:", fit.lasso$lambda.min)
```

```
## The optimal value of lambda for Ridge is: 0.02982835
```

The plot displays the mean cross-validated error in red with bars corresponding to standard errors. The leftmost dotted line in each plot corresponds to the λ that minimizes the error (`lambda.min` in the fitted object); the dotted line to the right corresponds to the largest value of λ such that the error is within one standard error from the minimum (`fit.lasso$lambda.1se` in the fitted object)

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'.

Answer

```
#retrieving lambda min for Lasso and Ridge
lambdamin.lasso <- fit.lasso$lambda.min
lambdamin.ridge <- fit.ridge$lambda.min
#retrieving the index of optimal lambda for Lasso and Ridge
idx.lambdamin.lasso <- which(lambdamin.lasso == fit.lasso$lambda)
idx.lambdamin.ridge <- which(lambdamin.ridge == fit.ridge$lambda)

#retrieving lambda lse for Lasso and Ridge
lambda1se.lasso <- fit.lasso$lambda.1se
lambda1se.ridge <- fit.ridge$lambda.1se
#retrieving the index of lambda lse for Lasso and Ridge
idx.lambda1se.lasso <- which(lambda1se.lasso == fit.lasso$lambda)
idx.lambda1se.ridge <- which(lambda1se.ridge == fit.ridge$lambda)

#Computing AUC values with the index
AUC.lambdamin.lasso <- signif(fit.lasso$cvm[idx.lambdamin.lasso],3)
AUC.lambda1se.lasso <- signif(fit.lasso$cvm[idx.lambda1se.lasso],3)
AUC.lambdamin.ridge <- signif(fit.ridge$cvm[idx.lambdamin.ridge],3)
AUC.lambda1se.ridge <- signif(fit.ridge$cvm[idx.lambda1se.ridge],3)

#Model size of each lambda values
ms.lasso.min <- fit.lasso$nzero[idx.lambdamin.lasso]
ms.lasso.1se <- fit.lasso$nzero[idx.lambda1se.lasso]
ms.ridge.min <- fit.ridge$nzero[idx.lambdamin.lasso]
ms.ridge.1se <- fit.ridge$nzero[idx.lambda1se.lasso]

#tabulating model statistics for Lasso and Ridge
dt <-data.table(model = c("Lasso.min", "Lasso.1se",
                          "Ridge.min", "Ridge.1se"),
                Lambda = c(signif(lambdamin.lasso,3),
                           signif(lambda1se.lasso,3),
                           signif(lambdamin.ridge,3),
                           signif(lambda1se.ridge,3)),
                ModelSize = c(ms.lasso.min, ms.lasso.1se,
                              ms.ridge.min, ms.ridge.1se),
                AUC = c(AUC.lambdamin.lasso, AUC.lambda1se.lasso,
                        AUC.lambdamin.ridge, AUC.lambda1se.ridge))
kable(dt, caption = "Lambda values with its model size and AUC") %>%
  kable_styling(latex_options = "hold_position")
```

Table 1: Lambda values with its model size and AUC

model	Lambda	ModelSize	AUC
Lasso.min	0.0298	8	0.981
Lasso.1se	0.0572	6	0.976
Ridge.min	0.0368	30	0.983
Ridge.1se	2.0100	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.

Answer

```
#define full model and null model
train.breast <- train.breast[,-c(1)]
full.model <- glm(diagnosis~., data = train.breast, family = "binomial")
null.model <- glm(diagnosis~1, data = train.breast, family = "binomial")

#perform forward and backward stepwise selection
model.B<- stepAIC(full.model, direction = "back", trace=FALSE)
model.S<- stepAIC(null.model, scope = list(upper = full.model),
                  direction = "forward", trace=FALSE)
```

The forward model takes a null model(which is a model without any feature) and only considers the intercept as a starting point and then progress towards the full model (with 30 features) by adding features. It can be seen here that the final model (forward) considers more features than the backward model. The forward model considers 15 features instead of 14 (in the backwards model)

```
#Computing standardised regression coefficients of each model
B.coef <- lm.beta(model.B)$standardized.coefficients
B.order<- order(abs(B.coef), decreasing = TRUE)
S.coef <- lm.beta(model.S)$standardized.coefficients
S.order<- order(abs(S.coef), decreasing = TRUE)

#tabulating the model coefficients for backward and forward stepwise selection
table <- list(B.coef[B.order], S.coef[S.order])
kable(table, col.names = "Coefficients", caption = "Coefficients of Model B and Model S") %>%
  kable_styling(latex_options = "hold_position")
```

Table 2: Coefficients of Model B and Model S

	Coefficients
radius.worst	53.047377
area.worst	-40.610055
perimeter	-18.500646
concavity.worst	8.472730
concavepoints	8.398555
radius	7.378246
radius.stderr	7.371562
texture.worst	5.605099
compactness.worst	-5.320776
concavepoints.worst	-3.831640
texture.stderr	-3.722656
smoothness.worst	2.008449
(Intercept)	0.000000

	Coefficients
area.worst	-44.347062
radius.worst	39.117706
perimeter.worst	16.396414
perimeter	-13.329464
radius.stderr	10.235291
compactness.worst	-5.930017
radius	5.544839
concavity	5.364296
texture.worst	4.932050
perimeter.stderr	-4.761290
concavity.worst	4.299931
texture.stderr	-3.028769
smoothness.worst	2.661371
area.stderr	2.278462
(Intercept)	0.000000

Problem 1.d (3 points)

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

Answer

```
#computing the AIC values of each model
cat("Model B AIC:", model.B$aic)
```

```
## Model B AIC: 99.47075
```

```
cat("Model S AIC:", model.S$aic)
```

```
## Model S AIC: 105.2948
```

```
#Testing goodness of fit of model B and model S
cat("Model B deviance:", model.B$deviance)
```

```
## Model B deviance: 73.47075
```

```
cat("Model S deviance:", model.S$deviance)
```

```
## Model S deviance: 75.29482
```

```
pchisq(model.B$null.deviance - model.B$deviance, df = 12, lower.tail = FALSE)
```

```
## [1] 1.462963e-89
```

```
pchisq(model.S$null.deviance - model.S$deviance, df = 14, lower.tail = FALSE)
```

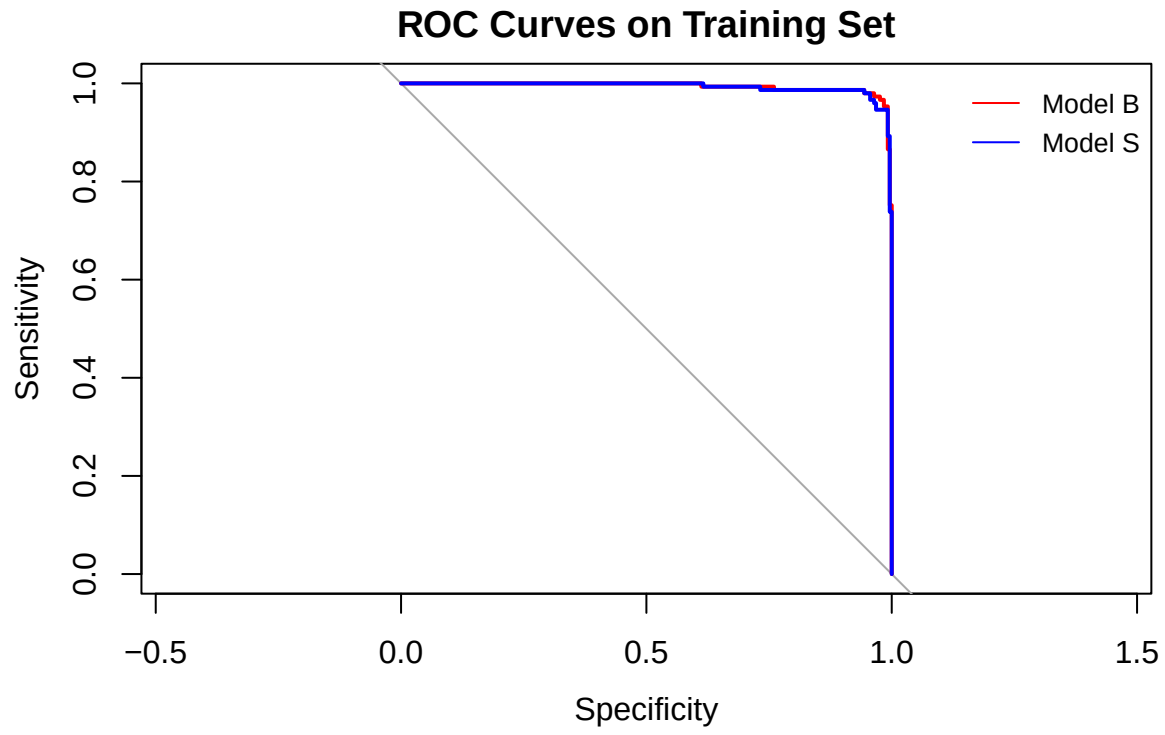
```
## [1] 1.350598e-87
```

Problem 1.e (2 points)

Compute the training AUC for model B and model S.

Answer

```
invisible({capture.output({
model.B.auc <- roc(train.breast$diagnosis, model.B$fitted.values, plot = TRUE,
    xlim = c(0,1), col = "red", main = "ROC Curves on Training Set")
model.S.auc <- roc(train.breast$diagnosis, model.S$fitted.values, plot = TRUE,
    add = TRUE, col = "blue")
legend("topright", legend = c("Model B", "Model S"),
         col = c("red", "blue"), lty = 1, cex = 0.8, bty = "n")
})})
```




```
dt <- data.table(c("Model B", "Model S"),
                    c(model.B.auc$auc, model.S.auc$auc))
setnames(dt, c("Model", "Accuracy") )
kable(dt, caption = "Training Accuracy of Model B and Model S") %>%
  kable_styling(latex_options = "hold_position")
```

Table 3: Training Accuracy of Model B and Model S

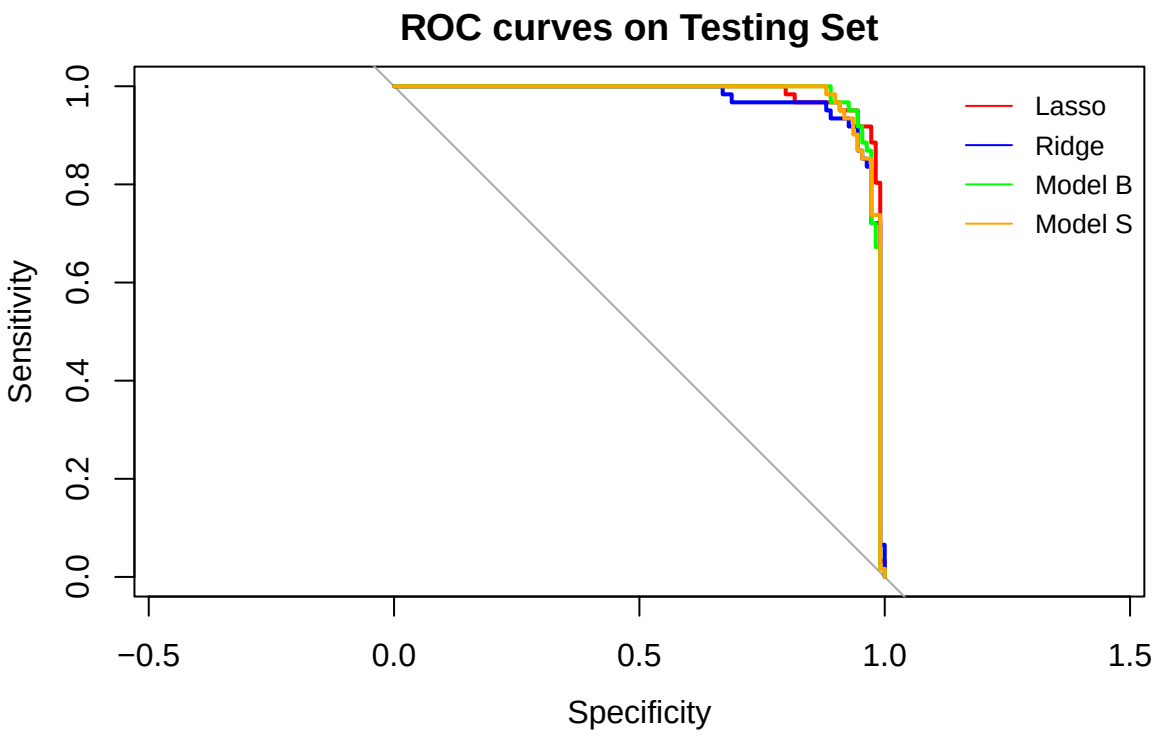
Model	Accuracy
Model B	0.9936376
Model S	0.9929396

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 sameplot, 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.

Answer

```
#prediction of Lasso and Ridge Regression Model
lasso.pred <- predict(fit.lasso, newx = as.matrix(test.breast[,-c(1,2)]), s = "lambda.1se")
ridge.pred <- predict(fit.ridge, newx = as.matrix(test.breast[,-c(1,2)]), s = "lambda.1se")

#prediction of Model B and Model S
B.pred <- predict(model.B, newdata = test.breast, type = "response")
S.pred <- predict(model.S, newdata = test.breast, type = "response")

#Plotting ROC
invisible({capture.output({
lasso.auc <- roc(test.breast$diagnosis, lasso.pred, plot = TRUE, xlim = c(0,1),
                 col = "red", main = "ROC curves on Testing Set")$auc
ridge.auc <- roc(test.breast$diagnosis, ridge.pred, plot = TRUE, col = "blue",
                 add = TRUE)$auc
B.auc <- roc(test.breast$diagnosis, B.pred, plot = TRUE, col = "green",
             add = TRUE)$auc
S.auc <- roc(test.breast$diagnosis, S.pred, plot = TRUE, col = "orange",
             add = TRUE)$auc
legend("topright", legend = c("Lasso", "Ridge","Model B","Model S"),
         col = c("red", "blue", "green", "orange"), lty = 1, cex = 0.8, bty = "n")
})})
```




```
#Tabulating AUC values
train.auc <- c(AUC.lambda1se.lasso, AUC.lambda1se.ridge,
               model.B.auc$auc, model.S.auc$auc)
test.auc <- c(lasso.auc, ridge.auc, B.auc, S.auc)
models <- c("Lasso Regression", "Ridge Regression", "Model B", "Model S")

# We make a table and voila!
table <- data.table(models, train.auc, test.auc)
kable(table, caption = "Training and Testing Acuraccy of each Model") %>%
  kable_styling(latex_options = "hold_position")
```

Table 4: Training and Testing Acuraccy of each Model

models	train.auc	test.auc
Lasso Regression	0.9760000	0.9808994
Ridge Regression	0.9770000	0.9712739
Model B	0.9936376	0.9802978
Model S	0.9929396	0.9790946

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.

Answer

```
gdm.dt <- data.table(fread("data_assignment2/GDM.raw.txt"))
```

```
#Performing Imputation using median
for (colnm in colnames(gdm.dt,-1)){
  gdm.dt[[colnm]][is.na(gdm.dt[[colnm]])] <- median(gdm.dt[[colnm]], na.rm = T)
}
```

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.

Answer

```
univ.glm.test <- function(x, y, order = FALSE){
  stopifnot(nrow(x) == length(y))
  #predefine data table
  output <- data.table("SNP" = character(),
                       "coefficients" = numeric(), "odds.ratios" = numeric(),
                       "std.error" = numeric(), "p.value" = numeric())
  #run logistric regression on each SNP
  for (i in 1:ncol(x)){
    regr <- glm(y ~ x[[i]], family = binomial(link = "logit"))
    summarised <- coef(summary(regr))
    output <- rbind(output, list(names(x)[i], summarised[2,1],
                                 exp(summarised[2,1]), summarised[2,2],
                                 summarised[2,4]))
  }
  #case when order set as TRUE
  if(order == TRUE){
    output <- output[order(p.value)]
  }
  return(output)
}
```

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.

Answer

```
#defining x with SNP values
x <- gdm.dt[, 4:ncol(gdm.dt)]
#defining y containing only pheno
y <- gdm.dt[[3]]
#performing logistic regression on each SNP
study <- univ.glm.test(x, y)
kable(head(study), caption = "Logistic Regression on Pheno vs each SNP") %>%
  kable_styling(latex_options = "hold_position")
```

Table 5: Logistic Regression on Pheno vs each SNP

SNP	coefficients	odds.ratios	std.error	p.value
rs7513574_T	0.0021575	1.002160	0.1051372	0.9836280
rs1627238_A	0.1146379	1.121467	0.1138224	0.3138559
rs1171278_C	0.1214094	1.129087	0.1138073	0.2860628
rs1137100_A	0.0601048	1.061948	0.1104238	0.5862285
rs2568958_A	0.1493799	1.161114	0.1233800	0.2259989
rs1514175_A	0.0562296	1.057841	0.1052359	0.5931203

```
index <- which(study$p.value == min(study$p.value))
kable(study[index, ],
      caption=" most strongly associated to increased risk of gestational diabetes") %>%
  kable_styling(full_width = F, position = "center", latex_options = "hold_position")
```

Table 6: most strongly associated to increased risk of gestational diabetes

SNP	coefficients	odds.ratios	std.error	p.value
rs12243326_A	0.6454198	1.906787	0.1583787	4.6e-05

```
strong.coef <- study[index, ]$coefficients
strong.se <- study[index,]$std.error
confidence.int.95 <- round(exp(strong.coef + 1.96 * strong.se*c(-1,1)), 3)
confidence.int.99 <- round(exp(strong.coef + 2.576 * strong.se*c(-1,1)),3)
cat(" SNP most strongly associated to increased risk of gestational diabetes, ",
    "\n 95% Confidence Interval is", confidence.int.95,
    "\n 99% Confidence Interval is", confidence.int.99)
```

```
##  SNP most strongly associated to increased risk of gestational diabetes,
##  95% Confidence Interval is 1.398 2.601
##  99% Confidence Interval is 1.268 2.867
```

```
newindex <- which(study$odds.ratio < 1)
best <- study[newindex,]

# Select the SNP with lowest p value
```

```
idx <- which(best$p.value == min(best$p.value))
best.SNP <- best[idx]

kable(best.SNP,
      caption= "most protective effect on gestational diabetes") %>%
  kable_styling(full_width = F, position = "center", latex_options = "hold_position")
```

Table 7: most protective effect on gestational diabetes

SNP	coefficients	odds.ratios	std.error	p.value
rs2237897_T	-0.4394456	0.6443936	0.1126133	9.53e-05

```
#Compute Confidence Interval
best.coef <- best.SNP$coefficients
best.se <- best.SNP$std.error
best.ci.95 <- round(exp(best.coef +1.96*best.se *c(-1,1)),3)
best.ci.99 <- round(exp(best.coef +2.576*best.se *c(-1,1)),3)
cat("\n SNPs with most protective effect on gestational diabetes,",
    "\n 95% Confidence Interval is", best.ci.95,
    "\n 99% Confidence Interval is", best.ci.99)
```

```
##
##  SNPs with most protective effect on gestational diabetes,
##  95% Confidence Interval is 0.517 0.804
##  99% Confidence Interval is 0.482 0.861
```

We can see that SNP rs1423096_T has the highest odds ratio (1.91758) and hence is the most strongly associated to increased risk of gestational diabetes. In fact, this SNP increases the odds of having gestational diabetes by about 92!. The SNP with most significant protective effect is rs2237897_T and it reduced the risk of diabetes by about 35%.

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.

Answer

```
gene.names <- fread("data_assignment2/GDM.annot.txt")

#spliting the string of snp into snp and effect.allele
new.study <- study %>% copy() %>%
  .[, full.snp := SNP] %>%
  .[, c("SNP", "effect.allele") := do.call(Map, c(f = c, strsplit(SNP, "_")))]

#merging dataset with gene.names and my study with snp
merge.dt <- merge(new.study, gene.names, by.x = "SNP", by.y="snp", all = TRUE)

#finding hit.snps that has p-value<1e-4
hit.snps <- merge.dt[p.value<1e-4]

#tabulating hit.snps
kable(hit.snps[,c("SNP", "effect.allele","chrom","gene")],
      caption= "SNPs that have p-value $< 10^{-4}$") %>%
  kable_styling(latex_options = "hold_position")
```

Table 8: SNPs that have p-value $< 10^{-4}$

SNP	effect.allele	chrom	gene
rs12243326	A	10	TCF7L2
rs2237897	T	11	KCNQ1

```
#computing the gene list for 1st SNP that are within 1Mb window
hit.snp.window <- data.table()
idx <- which(hit.snps$SNP==hit.snps$SNP[1])
window.val <- merge.dt[(merge.dt$pos>= hit.snps$pos[idx] - 1e6) &
                         (merge.dt$pos<= hit.snps$pos[idx] + 1e6)]
hit.snp.window <- rbind(hit.snp.window, window.val)
caption <- paste("Gene within $1$Mb Window for",hit.snps$SNP[1])
kable(unique(data.table(hit.snp.window$gene)), col.names = "gene",
      caption = caption) %>% kable_styling(latex_options = "hold_position")
```

Table 9: Gene within 1Mb Window for rs12243326

gene
TCF7L2

```
#computing the gene list for 2nd SNP that are within 1Mb window
hit.snp.window <- data.table()
idx <- which(hit.snps$SNP == hit.snps$SNP[2])
window.val <- merge.dt[(merge.dt$pos >= hit.snps$pos[idx] - 1e6) &
```

```
                          (merge.dt$pos <= hit.snps$pos[idx] + 1e6)]
hit.snp.window <- rbind(hit.snp.window, window.val)
caption <- paste("Gene within $1$Mb Window for",hit.snps$SNP[2])
kable(unique(data.table(hit.snp.window$gene)), col.names = "gene",
      caption = caption) %>% kable_styling(latex_options = "hold_position")
```

Table 10: Gene within 1Mb Window for rs2237897

gene
TH
KCNQ1
CACNA2D4
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.

Answer

```
#Defining each weighted genetic risk score
hit.snp.1 <- merge.dt[p.value < 1e-4]
gdm.1 <- gdm.dt[, .SD, .SDcols = merge.dt[p.value <1e-4]$full.snp]
names(gdm.1) <- gsub("_.", "", x = names(gdm.1))
wgrs.1 <- as.matrix(gdm.1) %*% hit.snp.1$coefficients

hit.snp.2 <- merge.dt[p.value < 1e-3]
gdm.2 <- gdm.dt[, .SD, .SDcols = merge.dt[p.value <1e-3]$full.snp]
names(gdm.2) <- gsub("_.", "", x = names(gdm.2))
wgrs.2 <- as.matrix(gdm.2) %*% hit.snp.2$coefficients

hit.snp.3 <- merge.dt[gene=="FTO"]
gdm.3 <- gdm.dt[, .SD, .SDcols = merge.dt[gene == "FTO"]$full.snp]
names(gdm.3) <- gsub("_.", "", x = names(gdm.3))
wgrs.3 <- as.matrix(gdm.3) %*% hit.snp.3$coefficients

#Adding 3 columns to gdm.dt
scores <- c("score.1", "score.2", "score.3")
gdm.dt <- gdm.dt %>% copy() %>%
  .[,`:=`(scores.1 = wgrs.1, scores.2 = wgrs.2, scores.3 = wgrs.3)]
```

```
y <- gdm.dt[[3]]
x <- gdm.dt[,180:182]
#performing logistic regression on each score
wgrs.snp <- univ.glm.test(x, y)
#computing confidence intervals
wgrs.snp <- wgrs.snp %>%
  .[, lower.conf.int:=round(exp(coefficients + 1.96 * std.error*-1), 3)] %>%
  .[, upper.conf.int:=round(exp(coefficients + 1.96 * std.error), 3)] %>%
  .[, !"coefficients"] %>% .[, !"std.error"]
#tabulating the statistics of logistic regression on each score
kable(head(wgrs.snp), caption = "Logistic regression on Pheno vs Score") %>%
  kable_styling(latex_options = "hold_position")
```

Table 11: Logistic regression on Pheno vs Score

SNP	odds.ratios	p.value	lower.conf.int	upper.conf.int
scores.1	2.729433	0.0000000	1.915	3.890
scores.2	1.451854	0.0000000	1.279	1.648
scores.3	1.413857	0.2151883	0.818	2.445

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).

Answer

```
gdm.test <- setDT(fread("data_assignment2/GDM.test.txt"))
```

```
#Computing the weight genetic risk scores
gdm1.colname <- colnames(gdm.1)
gdm.test.1 <- gdm.test[,..gdm1.colname]
wgrs.test.1 <- as.matrix(gdm.test.1) %*% hit.snp.1$coefficients
gdm2.colname <- colnames(gdm.2)
gdm.test.2 <- gdm.test[,..gdm2.colname]
wgrs.test.2 <- as.matrix(gdm.test.2) %*% hit.snp.2$coefficients
gdm3.colname <- colnames(gdm.3)
gdm.test.3 <- gdm.test[,..gdm3.colname]
wgrs.test.3 <- as.matrix(gdm.test.3) %*% hit.snp.3$coefficients

#Adding 3 columns to gdm.test
scores <- c("score.1", "score.2", "score.3")
gdm.test <- gdm.test %>% copy() %>%
  .[,`:=`(scores.1 = wgrs.test.1, scores.2 = wgrs.test.2, scores.3 = wgrs.test.3)]
```

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.

Answer

```
#fitting each score and predicting its values
fit1 <- glm(y ~ scores.1, family = binomial(link = "logit"), data=gdm.dt)
pred.1 <- predict(fit1, newdata = gdm.test[,180], type="response")
fit2 <- glm(y ~ scores.2, family = binomial(link = "logit"), data=gdm.dt)
pred.2 <- predict(fit2, newdata = gdm.test[,181], type="response")
fit3 <- glm(y ~ scores.3, family = binomial(link = "logit"), data=gdm.dt)
pred.3 <- predict(fit3, newdata = gdm.test[,182], type="response")

#computing the log liklihood
cat("The log likelihood of score 1:",
    sum(gdm.test$pheno*log(pred.1)+(1-gdm.test$pheno)*log(1-pred.1)))
```

```
## The log likelihood of score 1: -25.06824
```

```
cat("The log likelihood of score 1:",
    sum(gdm.test$pheno*log(pred.2)+(1-gdm.test$pheno)*log(1-pred.2)))
```

```
## The log likelihood of score 1: -24.77693
```

```
cat("The log likelihood of score 1:",
    sum(gdm.test$pheno*log(pred.3)+(1-gdm.test$pheno)*log(1-pred.3)))
```

```
## The log likelihood of score 1: -28.05355
```

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.

Answer

```
gdm.study <- setDT(fread("data_assignment2/GDM.study2.txt"))
```

```
#Computing Confusion Matrix to check flip
gdm.study <- gdm.study[order(snp, effect.allele)]
meta.study <- new.study[order(SNP, effect.allele)]
are.equal <- gdm.study$effect.allele == meta.study$effect.allele
not.equal <- gdm.study$other.allele == meta.study$effect.allele
kable(table(are.equal, not.equal), caption = "Confusion Matrix") %>%
  kable_styling(latex_options = "hold_position")
```

Table 12: Confusion Matrix

	FALSE	TRUE
FALSE	2	27
TRUE	147	0

```
#Remove two false values
false.removed <- which(are.equal == FALSE & not.equal == FALSE)
meta.study <- meta.study[-false.removed]
gdm.study <- gdm.study[-false.removed]
are.equal <- are.equal[-false.removed]
not.equal <- not.equal[-false.removed]

#retrieve the coefficients from both studies
beta1 <- gdm.study$beta
beta2 <- meta.study$coefficients
beta2[not.equal] <- -beta2[not.equal]

#compute weight
weight1 <- 1/ gdm.study$se^2
weight2 <- 1/ meta.study$std.error^2
kable(list(head(weight1), head(weight2)), caption = "Weight of Studies") %>%
  kable_styling(latex_options = "hold_position")
```

Table 13: Weight of Studies

x	x
9.141470	95.95663
6.919489	48.35218
7.069651	51.12222
10.430711	80.72417
3.328963	32.39329
9.751846	80.53452

```r
#meta analysis computing beta, std.error and p-value
beta.meta <- (weight1*beta1 + weight2*beta2)/(weight1 + weight2)
se.meta <- sqrt(1 / (weight1 + weight2))
p.value.meta <- 2 * pnorm(abs(beta.meta / se.meta), lower.tail = F)

#tabulating the study with p-value<1e-4
summary <- data.table(snp=meta.study$SNP, beta = beta.meta,
                      std.error = se.meta, p.value = p.value.meta)
summary <- summary[which(summary$p.value<1e-4),]
summary <- summary[order(summary$p.value)]
kable(summary, caption = "Summary of the study with p-value$<1e-4$") %>%
  kable_styling(latex_options = "hold_position")
```

Table 14: Summary of the study with p-value$< 1e-4$

snp	beta	std.error	p.value
rs12243326	0.8918988	0.1129379	0.00e+00
rs2237897	-0.5903702	0.1002930	0.00e+00
rs3786897	-0.6069063	0.1141012	1.00e-07
rs2237892	-0.4834871	0.1066965	5.90e-06
rs4506565	-0.5396749	0.1299622	3.29e-05
rs7903146	0.5353424	0.1331728	5.82e-05
rs7901695	0.5409567	0.1374089	8.26e-05

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.

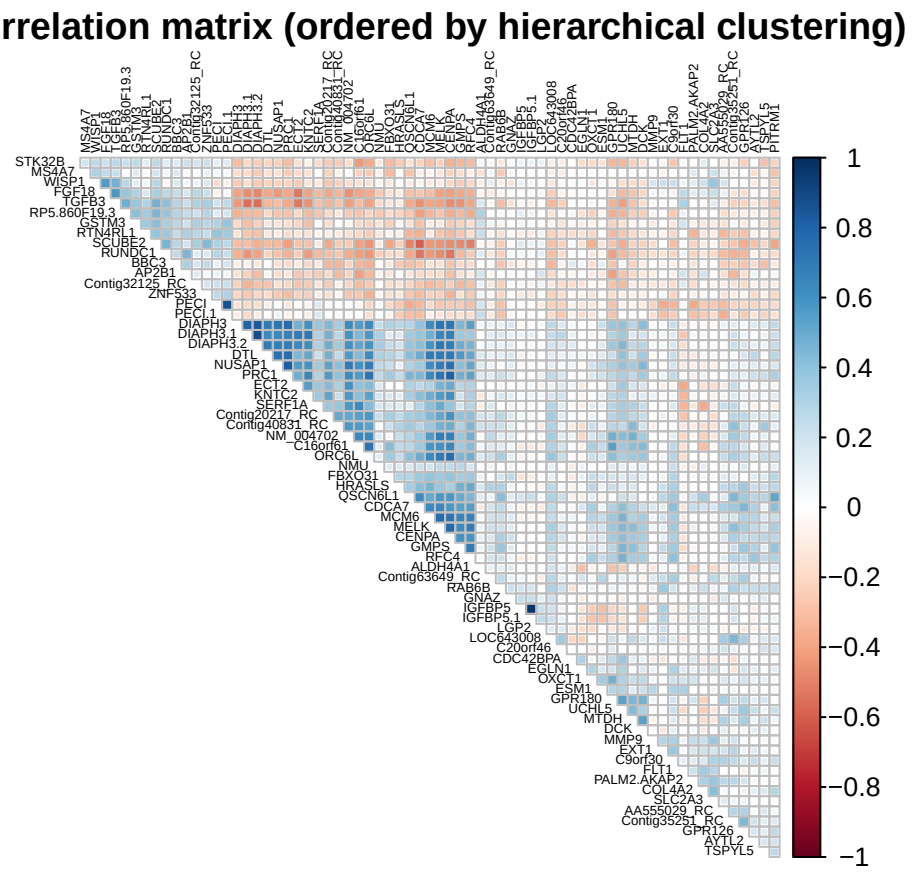
Problem 3.a (6 points)

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.

Answer

```
nki <- read.csv("data_assignment2/nki.csv")
```

```
#calculating correlation between features
nki.cor <- cor(nki[,7:76], use = "pairwise.complete")

#computing correlation plot between features
corrplot(nki.cor, order = "hclust", diag = FALSE, tl.col = "black",
         tl.cex = 0.4, method = "square", type = 'upper', mar=c(0,0,1,0),
         title="Correlation matrix (ordered by hierarchical clustering)")
```



```r
#Computing genes that have coefficient > 0.8
gene1 <- gene2 <- corr <- c()
for (i in 1:nrow(nki.cor)){
  for (j in 1:ncol(nki.cor)){
    if (abs(nki.cor[i,j]) > 0.8 & nki.cor[i,j] != 1){
      gene1 <- c(gene1, rownames(nki.cor)[i])
      gene2 <- c(gene2, colnames(nki.cor)[j])
      corr <- c(corr, nki.cor[i,j])
    }
  }
}

#Defining table for list of pairs of genes with correlation > 0.8
cor0.8 <- data.table(gene1, gene2, corr)
kable(unique(cor0.8, by = "corr"),
      caption = "List of pairs of Genes with Correlation Coefficient $> 0.8$") %>%
  kable_styling(latex_options = "hold_position")
```

Table 15: List of pairs of Genes with Correlation Coefficient > 0.8

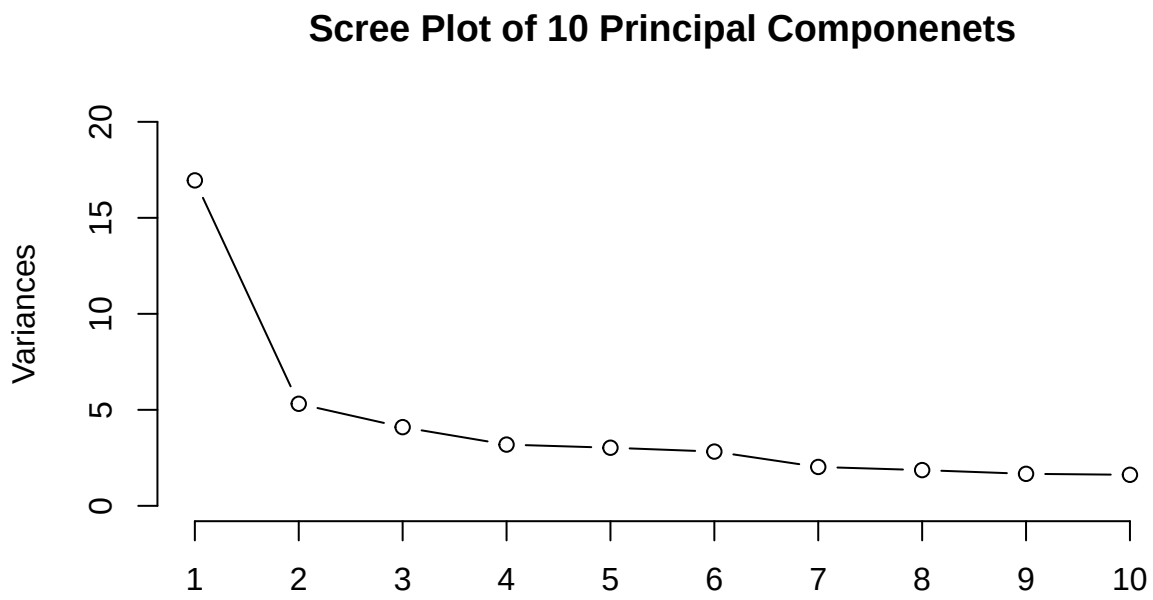
gene1	gene2	corr
DIAPH3	DIAPH3.1	0.8031368
DIAPH3	DIAPH3.2	0.8338591
NUSAP1	PRC1	0.8298356
DIAPH3.1	DIAPH3.2	0.8868741
PECI	PECI.1	0.8697836
IGFBP5	IGFBP5.1	0.9775030
PRC1	CENPA	0.8175424

Problem 3.b (8 points)

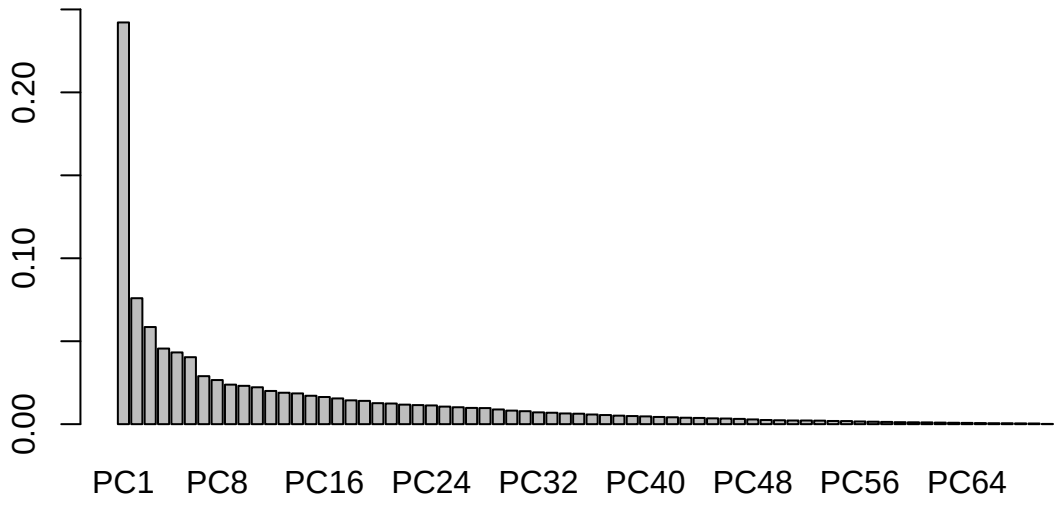
Run PCA (only over the columns containing gene expressions), in order to derive a patient-wise summary of all gene expressions (dimensionality reduction). Decide which components to keep and justify your decision. Test if those principal components are associated with the outcome in unadjusted logistic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

Answer

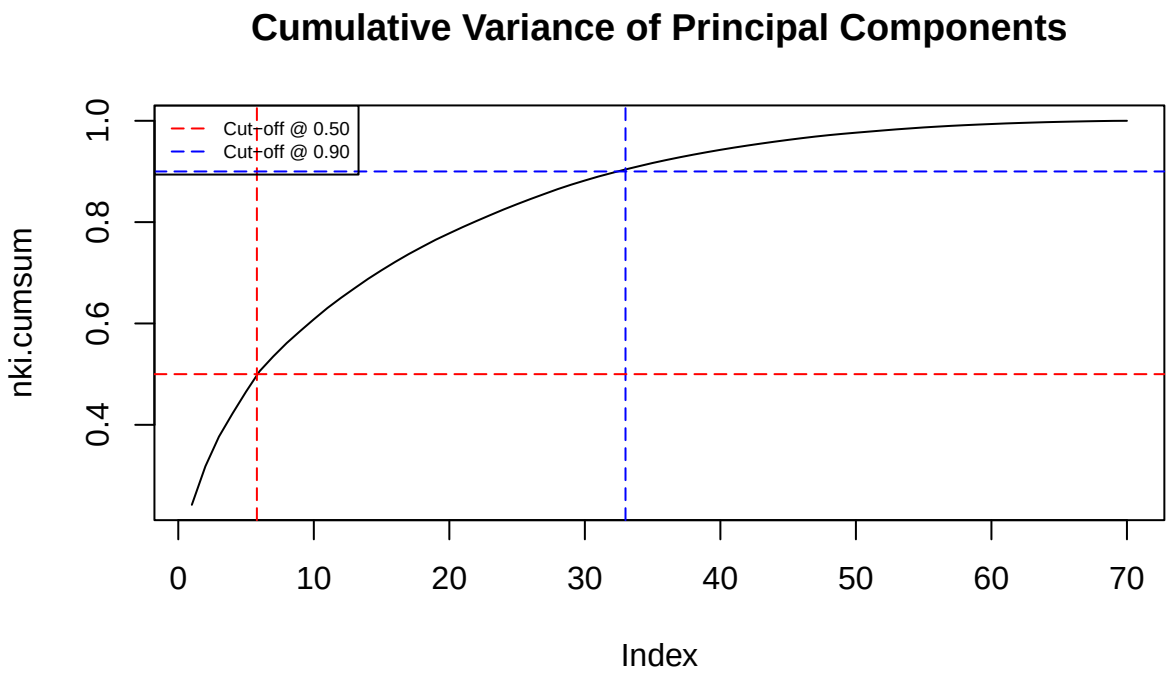
```
#conducting principal component analysis
nki.pca <- prcomp(nki[,7:76], center = TRUE, scale = TRUE)

#Scree Plot for each Principal Components
plot(nki.pca, type = "line", main = "Scree Plot of 10 Principal Componenets",
     ylim = c(0,20))
```




```
#Percentage Variance explained by each Principal Components
barplot(summary(nki.pca)$importance[2,], ylim = c(0, 0.25))
```




```
#Cummulative Variance explained by each Principal Components
nki.cumsum <- cumsum(nki.pca$sdev^2 / sum(nki.pca$sdev^2))
plot(nki.cumsum, type = "line", main = "Cumulative Variance of Principal Components")
abline(v = 5.8, col="red", lty=5); abline(v = 33, col="blue", lty=5)
abline(h = 0.50, col="red", lty=5); abline(h = 0.90, col="blue", lty=5)
legend("topleft", legend = c("Cut-off @ 0.50", "Cut-off @ 0.90"),
       col=c("red","blue"), lty=5, cex=0.6)
```




```
pc <- nki.pca$x[,1:3] #retrieving the first 3 principal components
#fitting logistic regression on each principal component
pc1.fit <- glm(nki$Event~pc[,1], family = "binomial")
pc2.fit <- glm(nki$Event~pc[,2], family = "binomial")
pc3.fit <- glm(nki$Event~pc[,3], family = "binomial")

#fitting adjusted logistic regression on each principal component
pc1.fit.adj <- glm(nki$Event~pc[,1]+ nki$EstrogenReceptor +
                     nki$Grade + nki$Age, family = "binomial")
pc2.fit.adj <- glm(nki$Event~pc[,2]+ nki$EstrogenReceptor +
                     nki$Grade + nki$Age, family = "binomial")
pc3.fit.adj <- glm(nki$Event~pc[,3]+ nki$EstrogenReceptor +
                     nki$Grade + nki$Age, family = "binomial")

#Define table for Association test for each principal component
model <- c("Unadjusted PC1", "Unadjusted PC2", "Unadjusted PC3",
           "Adjusted PC1", "Adjusted PC2", "Adjusted PC3")
coefficients <- c(pc1.fit$coefficients[2], pc2.fit$coefficients[2],
                  pc3.fit$coefficients[2], pc1.fit.adj$coefficients[2],
                  pc2.fit.adj$coefficients[2], pc3.fit.adj$coefficients[2])
p.value <- c(summary(pc1.fit)$coefficients[2,4],
             summary(pc2.fit)$coefficients[2,4],
             summary(pc3.fit)$coefficients[2,4],
             summary(pc1.fit.adj)$coefficients[2,4],
             summary(pc2.fit.adj)$coefficients[2,4],
             summary(pc3.fit.adj)$coefficients[2,4])
dt <- data.table(model, coefficients, p.value)
kable(dt, caption="Association Test for each Principal Component") %>%
  kable_styling(latex_options = "hold_position")
```

Table 16: Association Test for each Principal Component

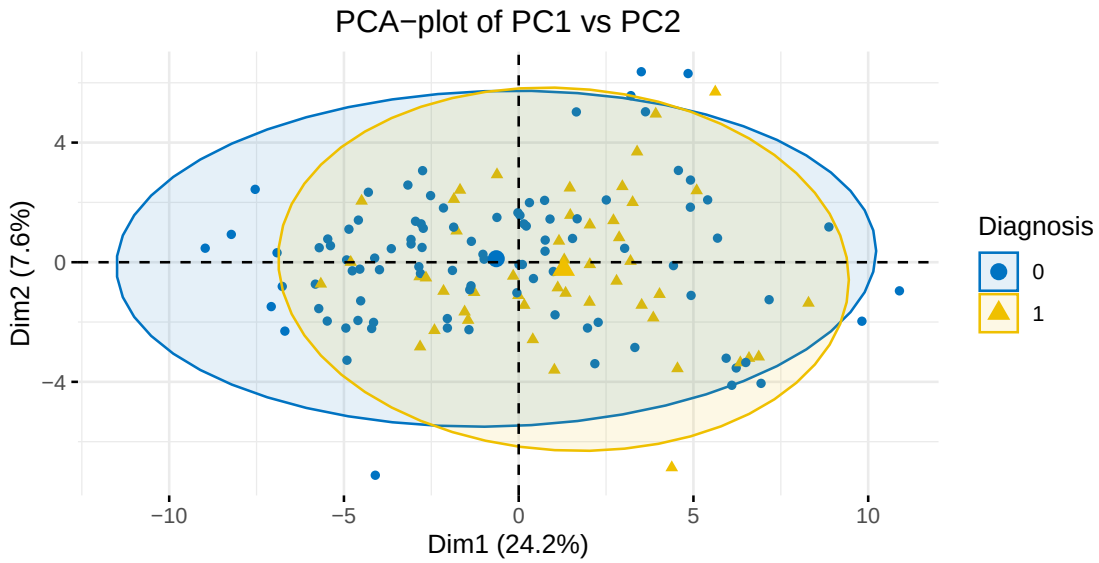
model	coefficients	p.value
Unadjusted PC1	0.1176835	0.0094250
Unadjusted PC2	-0.0671668	0.3885231
Unadjusted PC3	0.2435485	0.0086300
Adjusted PC1	0.0721592	0.2723812
Adjusted PC2	0.0051644	0.9550636
Adjusted PC3	0.2183706	0.0245456

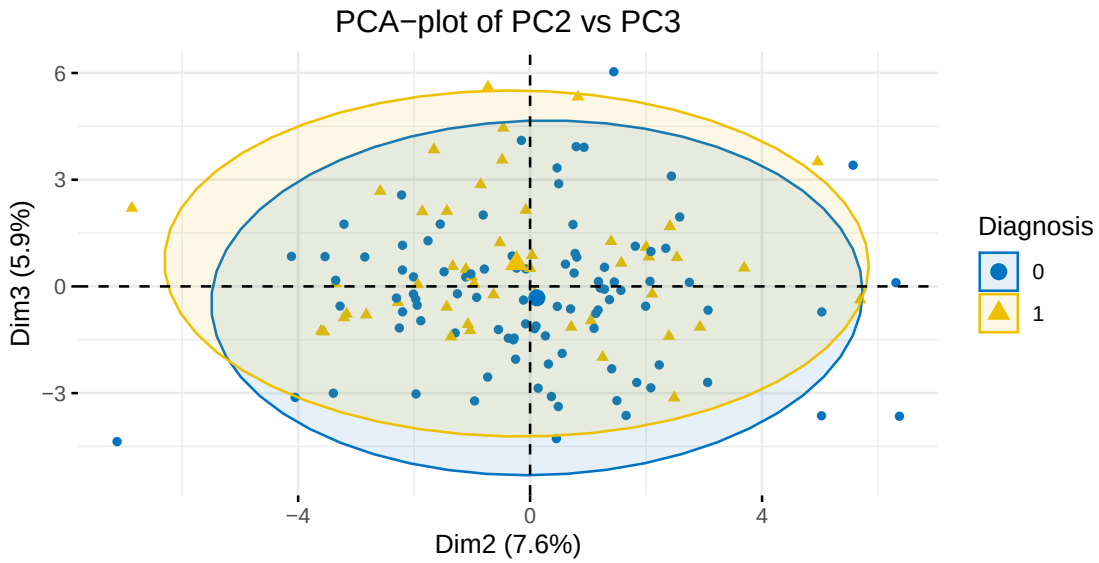
Problem 3.c (8 points)

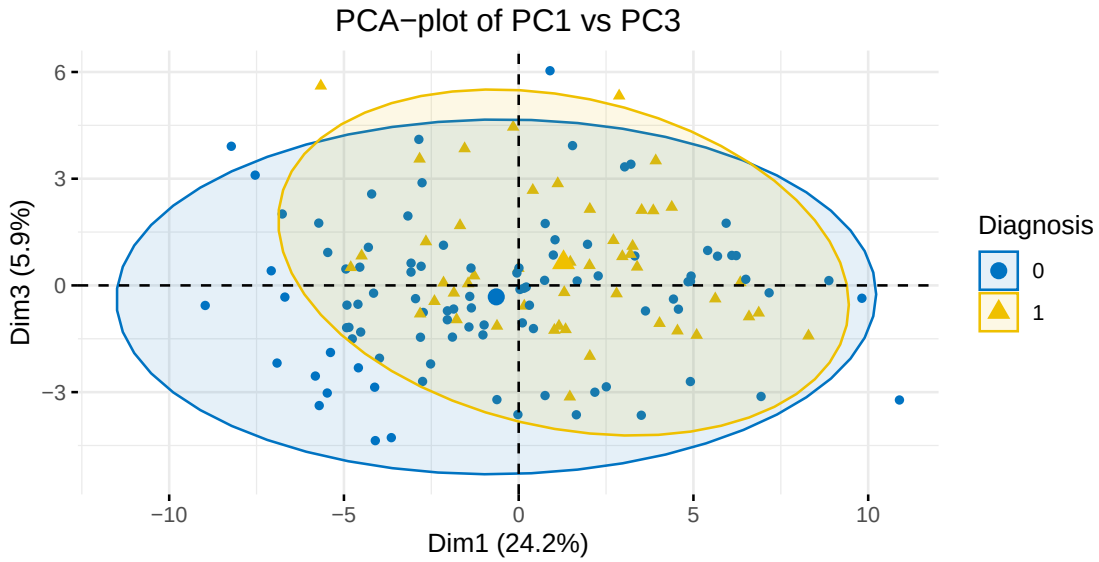
Use plots to compare with the correlation structure observed in problem 2.a and to examine how well the dataset may explain your outcome. Discuss your findings and suggest any further steps if needed.

Answer

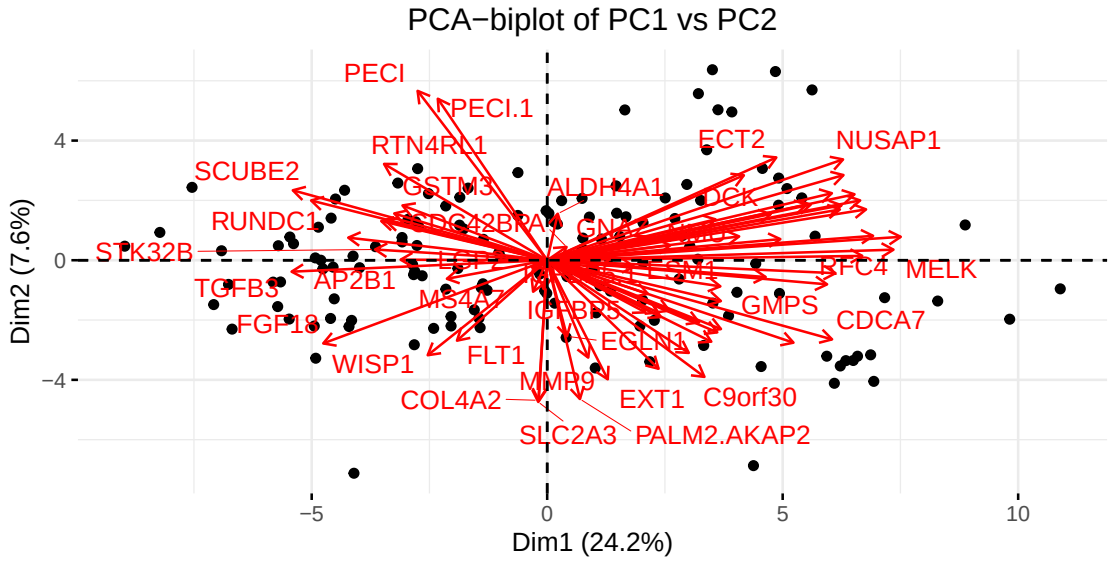
```
#Plotting PCA plot of each principal Component
p1 <- fviz_pca_ind(nki.pca, geom = 'point', habillage = nki$Event,
                   axes = c(1,2), addEllipses = T, palette = "jco",
                   legend.title = "Diagnosis") +
  ggtitle("PCA-plot of PC1 vs PC2") + theme(plot.title = element_text(hjust = 0.5))

p2 <- fviz_pca_ind(nki.pca, geom = 'point', habillage = nki$Event, axes = c(2,3),
             addEllipses = T, palette="jco", legend.title="Diagnosis") +
  ggtitle("PCA-plot of PC2 vs PC3") + theme(plot.title = element_text(hjust = 0.5))

p3 <- fviz_pca_ind(nki.pca, geom='point', habillage = nki$Event, axes = c(1,3),
             addEllipses = T, palette="jco", legend.title="Diagnosis") +
  ggtitle("PCA-plot of PC1 vs PC3") + theme(plot.title = element_text(hjust = 0.5))
grid.arrange(p1,p2,p3)
```

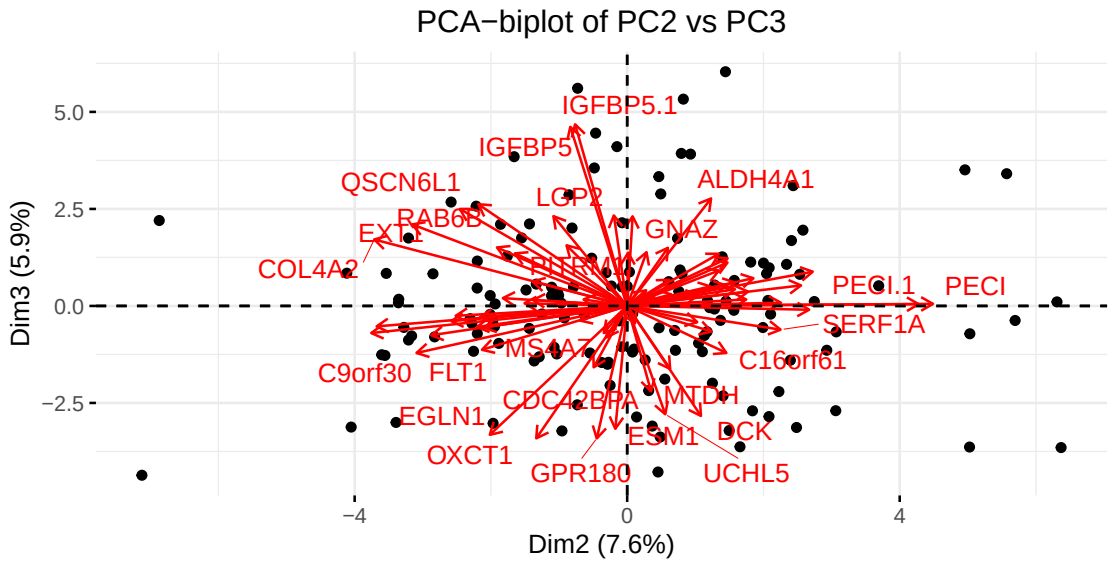

PCA−plot of PC1 vs PC2
Dim2 (7.6%)
Dim1 (24.2%)
Diagnosis
0
1

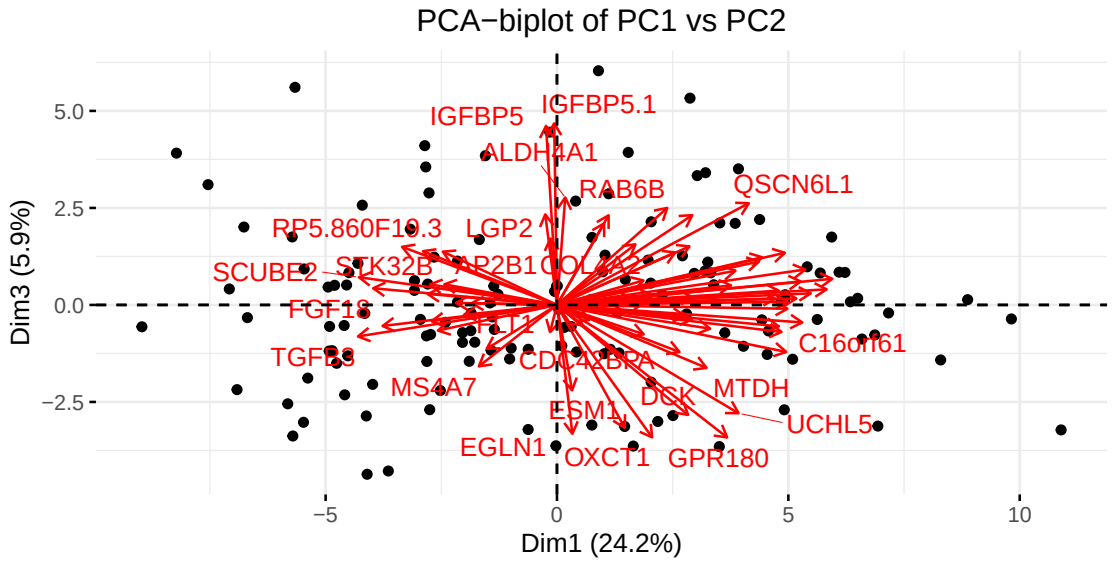

PCA−plot of PC2 vs PC3
Dim3 (5.9%)
Dim2 (7.6%)
Diagnosis
0
1


PCA−plot of PC1 vs PC3
Dim3 (5.9%)
Dim1 (24.2%)
Diagnosis
0
1

```r
#plotting biplot for each Principal Component
p1 <- fviz_pca_biplot(nki.pca, geom ='point', repel = T, col.var = "red", axes = c(1,2)) +
  ggtitle("PCA-biplot of PC1 vs PC2") + theme(plot.title = element_text(hjust = 0.5))
p2 <- fviz_pca_biplot(nki.pca, geom ='point', repel = T, col.var = "red", axes = c(2,3)) +
  ggtitle("PCA-biplot of PC2 vs PC3") + theme(plot.title = element_text(hjust = 0.5))
p3 <- fviz_pca_biplot(nki.pca, geom ='point', repel = T, col.var = "red", axes = c(1,3)) +
  ggtitle("PCA-biplot of PC1 vs PC2") + theme(plot.title = element_text(hjust = 0.5))
grid.arrange(p1,p2,p3)
```


PCA−biplot of PC1 vs PC2
PECI
PECI.1
RTN4RL1
ECT2
NUSAP1
SCUBE2
GSTM3
ALDH4A1
RUNDC1
CDC42BPA
GNAZ
STK32B
LGP2
RFC4
MELK
TGFB3
AP2B1
MS4A7
IGFBP5
GMPS
CDCA7
FGF18
EGLN1
WISP1
FLT1
MMP9
COL4A2
EXT1
C9orf30
SLC2A3
PALM2.AKAP2
Dim2 (7.6%)
Dim1 (24.2%)


PCA−biplot of PC2 vs PC3
IGFBP5.1
IGFBP5
QSCN6L1
ALDH4A1
LGP2
RAB6B
GNAZ
EXT1
COL4A2
PECI.1
PECI
SERF1A
MS4A7
C16orf61
C9orf30
FLT1
CDC42BPA
MTDH
EGLN1
ESM1
DCK
OXCT1
GPR180
UCHL5
Dim3 (5.9%)
Dim2 (7.6%)


PCA−biplot of PC1 vs PC2
IGFBP5
IGFBP5.1
ALDH4A1
RAB6B
QSCN6L1
RP5.860F19.3
LGP2
SCUBE2
STK32B
AP2B1
COL4A2
FGF18
FLT1
C16orf61
TGFB3
CDC42BPA
MS4A7
DCK
MTDH
ESM1
UCHL5
EGLN1
OXCT1
GPR180
Dim3 (5.9%)
Dim1 (24.2%)

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.

Answer

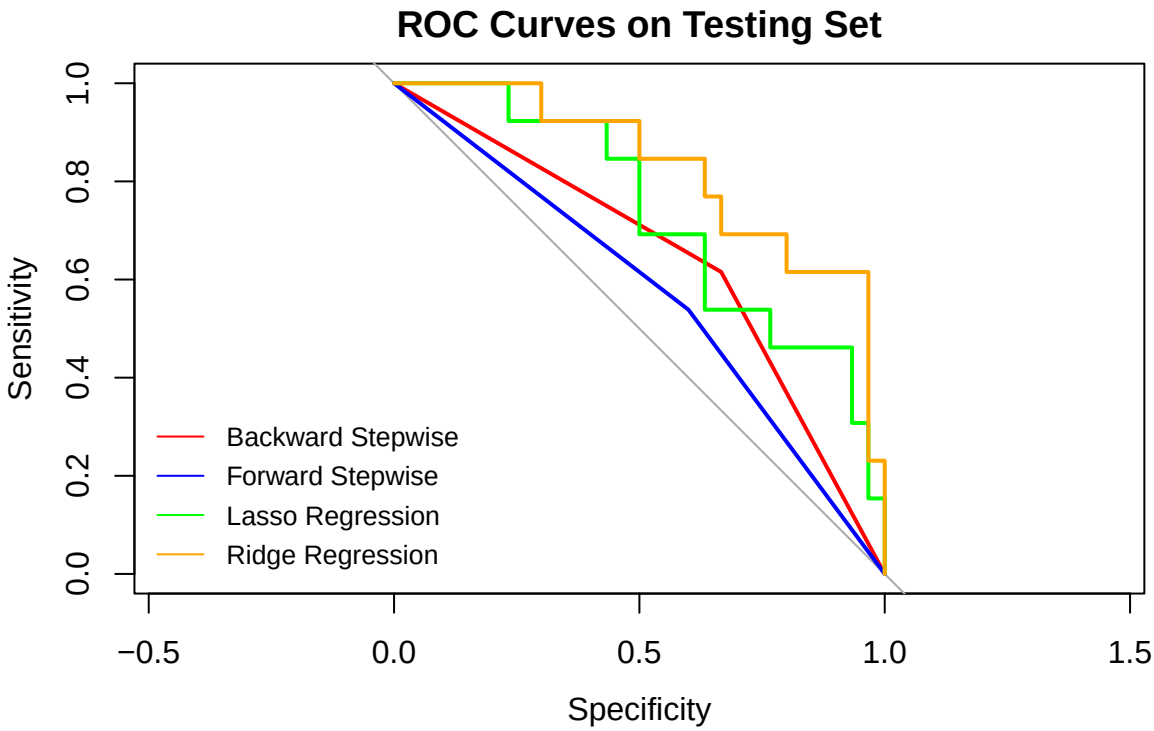
```
prepare.glmnet <- function(data, formula=~ .) {
  ## create the design matrix to deal correctly with factor variables,
  ## without losing rows containing NAs
  old.opts <- options(na.action='na.pass')
  x <- model.matrix(formula, data)
  options(old.opts)
  ## remove the intercept column, as glmnet will add one by default
  x <- x[, -match("(Intercept)", colnames(x))]
  return(as.data.frame(x))
}
```

```
invisible({capture.output({
  set.seed(984065)
  #split dataset into training and testing
  split.index <- createDataPartition(nki$Event, p = 0.7)$Resample1
  nki.y.train <- nki$Event[split.index]
  nki.y.test <- nki$Event[-split.index]

  #perform one-hot encoding
  nki.1hot <- prepare.glmnet(nki, ~.)
  nki.train <- nki.1hot[split.index,]
  nki.test <- nki.1hot[-split.index,]

  #define full and null model
  nki.full <- glm(Event ~., data = nki.train, family = binomial)
  nki.null <- glm(Event ~ 1, data = nki.train, family = binomial)

  #Forward and backward stepwise models
  nki.forward <- stepAIC(nki.null, scope = list(upper = nki.full),
                         direction = "forward")
  nki.backward <- stepAIC(nki.full, scope = list(upper = nki.null),
                          direction = "backward")

  #model size of backward and forward stepwise model
  ms.backward <- length(nki.backward$anova$Deviance)-1
  ms.forward <- length(nki.forward$anova$Deviance)-1
})})
```

```
set.seed(984065)
#Fitting Lasso and ridge regression on training data
fit.lasso <- cv.glmnet(as.matrix(nki.train[,-c(1)]), nki.y.train, alpha = 1,
                       family = "binomial", type.measure = "auc")
fit.ridge <- cv.glmnet(as.matrix(nki.train[,-c(1)]), nki.y.train, alpha = 0,
                       family = "binomial", type.measure = "auc")

#retrieve index of the lambda min for auc values
lambda.lasso.idx <- which(fit.lasso$lambda.min == fit.lasso$lambda)
lambda.ridge.idx <- which(fit.ridge$lambda.min == fit.ridge$lambda)
```

```r
lambda.lasso.auc <- signif(fit.lasso$cvm[lambda.lasso.idx],3)
lambda.ridge.auc <- signif(fit.ridge$cvm[lambda.ridge.idx],3)

#model size of Lasso and Ridge regression of lambda min
ms.lasso <- fit.lasso$nzero[lambda.lasso.idx]
ms.ridge <- fit.ridge$nzero[lambda.ridge.idx]
```

```r
invisible({capture.output({
#Lasso and Ridge Regression model
lasso.pred <- predict(fit.lasso, newx = as.matrix(nki.test[,-c(1)]), s="lambda.min")
ridge.pred <- predict(fit.ridge, newx = as.matrix(nki.test[,-c(1)]), s="lambda.min")

#Backward and Forward Stepwise model
backward.pred <- predict(nki.backward, newdata = nki.test, type = "response")
forward.pred <- predict(nki.forward, newdata = nki.test, type = "response")

#Compute AUC values for each model
backward.auc <- roc(nki.y.test, backward.pred, plot = TRUE, xlim = c(0,1),
                    col = "red", main = "ROC Curves on Testing Set")
forward.auc <- roc(nki.y.test, forward.pred,
                   plot = TRUE, add = TRUE, col = "blue")
lasso.auc <- roc(nki.y.test, lasso.pred,
                   plot = TRUE, add = TRUE, col = "green")
ridge.auc <- roc(nki.y.test, ridge.pred,
                   plot = TRUE, add = TRUE, col = "orange")
legend("bottomleft", legend = c("Backward Stepwise", "Forward Stepwise",
                                "Lasso Regression", "Ridge Regression"),
       col = c("red", "blue", "green", "orange"),
       lty = 1, cex = 0.8, bty = "n")
})})
```




```
#Defining the table for the accuracy and model size of each model
Model <- c("Backward Stepwise", "Forward Stepwise",
           "Lasso Regression", "Ridge Regression")
AUC <- c(backward.auc$auc, forward.auc$auc, lasso.auc$auc, ridge.auc$auc)
model.size <- c(ms.backward, ms.forward, ms.lasso, ms.ridge)
dt <- data.table(Model, AUC, model.size)
kable(dt, caption="Accuracy of Models") %>%
  kable_styling(latex_options = "hold_position")
```

Table 17: Accuracy of Models

Model	AUC	model.size
Backward Stepwise	0.6410256	55
Forward Stepwise	0.5692308	21
Lasso Regression	0.7307692	49
Ridge Regression	0.8256410	76