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)

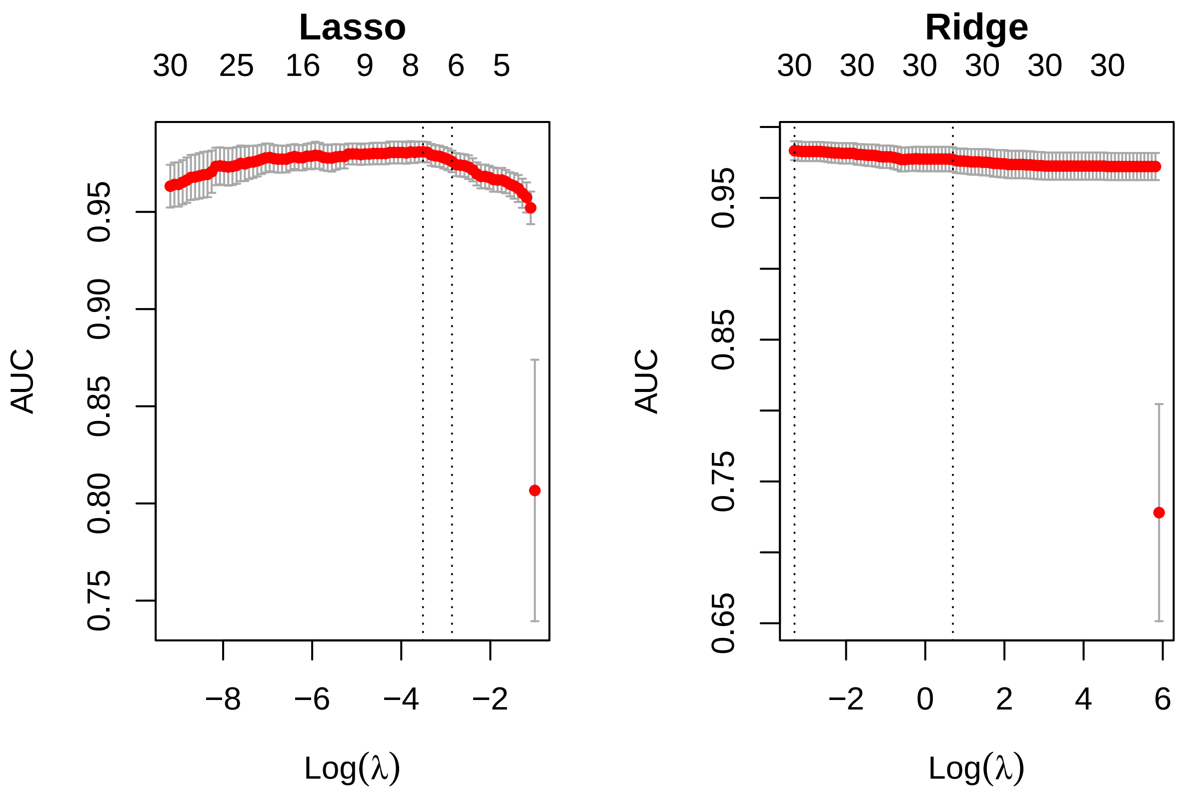
Using package caret, create a data partition so that the training set contains 70% of the observations (set the random seed to 984065 beforehand). Fit both a ridge regression model and a lasso model which uses cross-validation on the training set to diagnose the type of tumour from the 30 biomarkers. 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")
```



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

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

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

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

We performed cross validation on Lasso and Ridge regression. To do that, we split the training and testing dataset into $7:3$ ratio and run `cv.glmnet()`.

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). To that, we obtained optimal values of λ that maximises AUC for Lasso and Ridge of 0.0368 and 0.0298 respectively.

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 1se for Lasso and Ridge
lambda1se.lasso <- fit.lasso$lambda.1se
lambda1se.ridge <- fit.ridge$lambda.1se
#retrieving the index of lambda 1se 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(full_width = F, position = "center", 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

λ is a penalty parameter that shrinks the coefficient to zero for Lasso and near zero for Ridge. This is the reason why, Lasso has a smaller model size compared to Ridge regression. From Table 1, Lasso with $\lambda = 0.0298$ will have model size of 8 and Ridge with $\lambda = 0.0368$ will have model size of 30 (unchanged). Looking at Table 1. we can see that `lambda.min` for both Lasso and Ridge regression have low AUC values compared to `lambda.1se`. This is because, the `lambda.min` values maximises the AUC values and minimises the error. Then, we compare the model accuracy among Lasso and Ridge. Lasso has the AUC value of 0.981 and Ridge has 0.983. From this, Ridge regression represents the training dataset better than Lasso.

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

```r
#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)
```

```r
#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(full_width = F, position = "center", 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

We perform a forward and backward stepwise selection here using `stepAIC()`. We will specify backward stepwise selection as Model B and forward stepwise selection as Model S.

The forward model starts from a null model and adds up the parameters towards the full model. On the other hand, barckward model starts from a full model with 30 parameters and removes. This process is controlled by the AIC value where the the algorithm will converge when the AIC value does not decrease any more.

Looking at Table 2 below. we can see that there are 14 features for Model B and 15 features for Model S. Although there is a difference in the number of feature used, we can detect the common features. This includes, `radius.worst`, `area.worst`, `perimeter`, `radius`, `radius.stderr`, `texture.worse`, `smoothness.worst`, `concavity.worst`. We can conclude that these features are indeed highly influential in detecting breast cancer.

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
```

To compare, the goodness of fit of Model B and Model S, we computed the AIC values for each model and conducted χ^2-test.

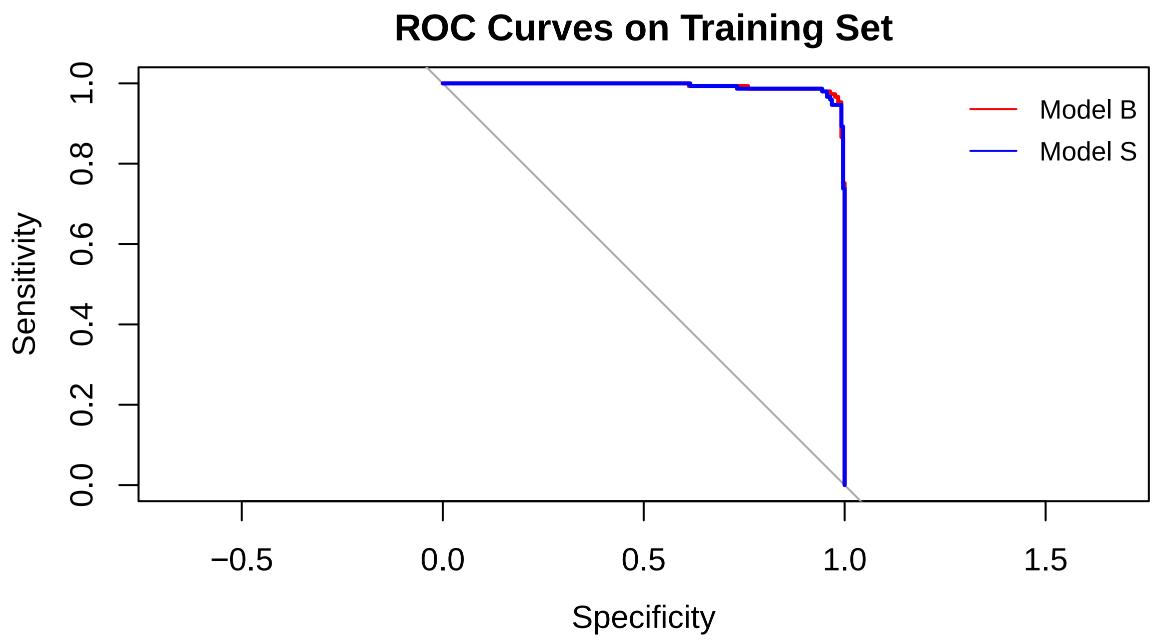
Model B has a AIC value of 99.5 and Model S has a AIC value of 105.3. Model B has a smaller AIC value thus a better model. To further validate this, we can also refer to the result from χ^2-test. We obtained the p-value for each model where Model B has $1.46e - 89$ and Model S has $1.35e - 87$. Since Model B has a lower p-value, we can conclude that Model B is a better fit to the training dataset.

Problem 1.e (2 points)

Compute the training AUC for model B and model S.

Answer

```r
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")
})})
```



```r
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(full_width = F, position = "center", latex_options = "hold_position")
```

Table 3: Training Accuracy of Model B and Model S

Model	Accuracy
Model B	0.9936376
Model S	0.9929396

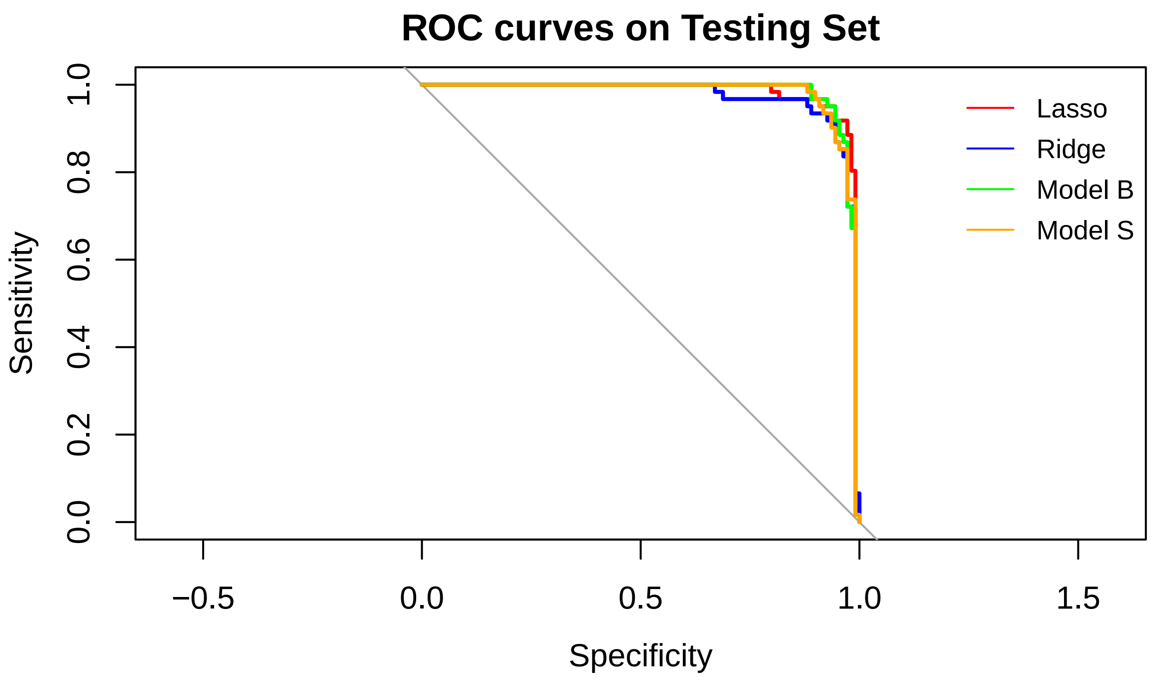
We computed the ROC curve using `roc()`, to compare the two models, Model B and Model S. Visually, we could not tell much difference. However, looking at Table 3, we can see that both models have high accuracy with the values of 0.9936 and 0.9929 respectively. Thus, like Q1 d), we can conclude that Model B is a better model.

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")
})})
```



```r
#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(full_width = F, position = "center", 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

We computed the ROC curve using the same method as Q1 e) but on testing data. From the plot, Lasso, Ridge, Model B and Model S are denoted in red, blue, green, orange respectively. We can clearly see the difference in the performance of each model. To scrutinise further, we look in Table 4 where we tabulated the accuracy on training and testing data for each model. All models obtained a high and similar accuracy. Therefore we can say that no model is neither overfitted nor underfitted. To pick the best model among the 4 models, we choose Model B as the best model. It has the highest accuracy on training data with the accuracy of 0.9936. Although Lasso has the highest accuracy on testing data with 0.9809, Model B also has a similar accuracy with 0.9803.

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

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

```r
#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 = TRUE)
}
```

We conducted imputation where the missing values are imputed with the median values. In here we do not use mean imputation as the values are categorical and not continuous. Thus imputed according to SNP-wise median allele count.

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

```r
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

```r
#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 first 6 SNPs") %>%
  kable_styling(latex_options = "hold_position")
```

Table 5: Logistic Regression on Pheno vs first 6 SNPs

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

```r
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

```r
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
```

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

# Select the SNP with lowest p value
```

```r
idx <- which(best$odds.ratios == min(best$odds.ratios))
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
rs11575839_C	-0.6022542	0.5475759	0.3758156	0.1090394

```r
#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 0.262 1.144
##  99% Confidence Interval is 0.208 1.442
```

With the predefined function `univ_glm_test()`, we conducted the logistic regression on `pheno` vs each `SNP`. From this, we obtained their `coefficients`, `odds.ratios`, `std.error` and `p.value` and tabulated in Table 5.

To compute the `SNP` with the most strongly associated to increased risk of gestational diabetes, we find the `SNP` with the lowest p-value. As a result, we observe `rs12243326_A` with p-value of $4.6e-5$ in Table 6. To that, its' 95% and 99% confidence intervals are $(1.398, 2.601), (1.268, 2.867)$ respectively. By taking exponential this computes the confidence interval of the odds ratios. Since it does not include 1, we have sufficient evidence to say that `rs12243326_A` is the most strongly associated to increased risk of gestational diabetes.

To compute the `SNP` with the most protective effect on gestational diabetes, we find the `SNP` with the lowest odds ratios. As a result, we observe `rs11575839_C` with odds ratios of 0.548 in Table 7. To that, its' 95% and 99% confidence intervals are $(0.262, 1.144), (0.208, 1.442)$ respectively. Since it contains 1 in the confidence interval, we do not have sufficient evidence to accept that `rs11575839_C` is the most protective effect on gestational diabetes.

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.

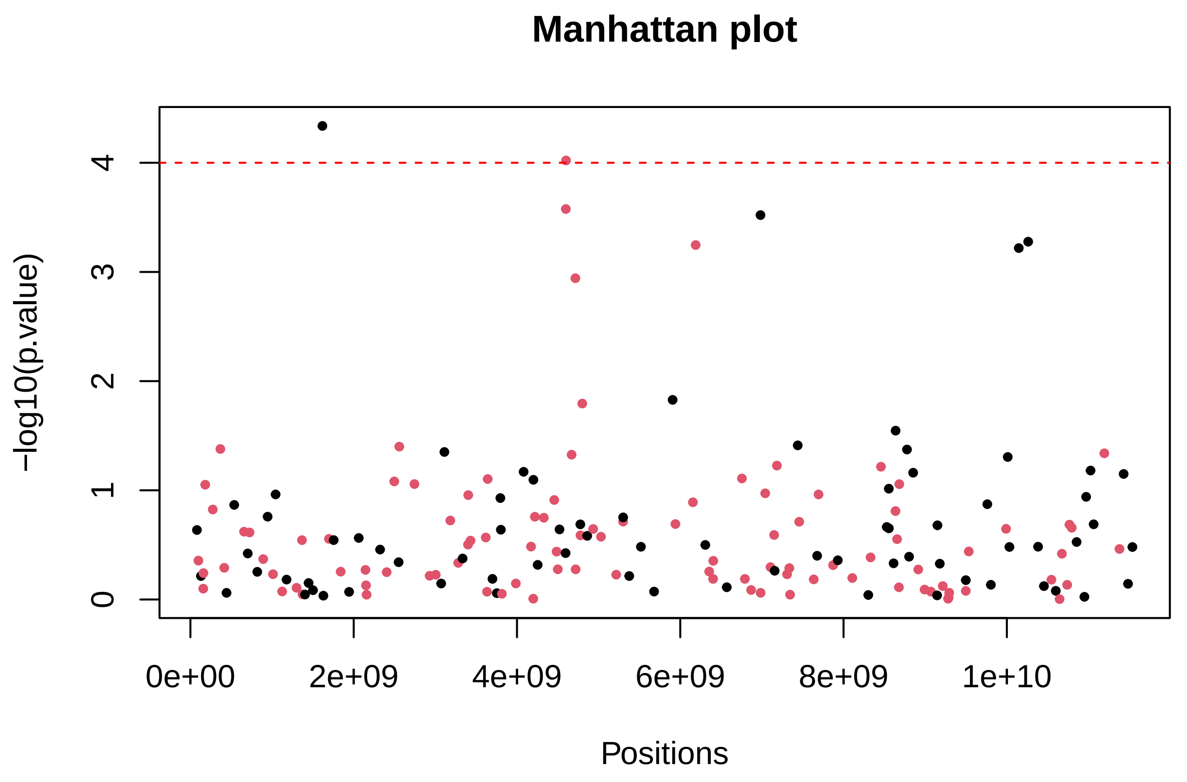
Answer

```r
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

```r
merge.dt <- merge.dt %>% copy() %>%
  .[, pos := as.numeric(pos)] %>%
  .[, cum.pos := cumsum(pos)] %>%
  .[, chrom := as.numeric(chrom)]
chrom.cols <- 1 + merge.dt$chrom %% 2 # 1 for even, 2 for odd chromosomes
with(merge.dt, plot(cum.pos, -log10(p.value), col = chrom.cols, pch = 20,
                    cex = 0.8, main = "Manhattan plot", xlab = "Positions"))
abline(h = -log10(1e-4), lty = 2, col = "red")
```



```
#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

In Table 8, we have `SNP`s that have p-value less than $1e-4$. The `SNP` names are `rs12243326_A` and `rs2237897_T` and we will denote them as `hit SNP`. They have chromosome number of 10 and 11 and gene of `TCF7L2` and `KCNQ1` repectively. This is further evident in the auxiliary Manhatten plot we have above where the points are located above the red dotted lines.

Then we further computed the genes within $1Mb$ window for each `SNP` in Table 9 and Table 10 repectively. Thus, those patient with mutation in the proposed genes in Table 9 and Table 10, have higher chance of being diagnosed with malignant tumour.

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

In this question, we constructed the weighted genetic risk score for that includes all `SNPs` with p-value $< 1e-4$, a score with all `SNPs` with p-value $< 1e-3$, and a score that only includes `SNPs` on the `FTO` gene. The weighted genetic risk scores are constructed in way such that both direction and assoication of the effect to the score for each `SNP`. These scores are then computed in Table 11. Looking at the p-value and the 95% confidence interval, we can see that the `Score3` has p-value of 0.215 and confidence interval of $(0.818, 2.445)$. Its p-value suggest in the insignificance and since the confidence interval contains 1, there is insufficient evidence for association.

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)]
kable(head(gdm.test[, c(180, 181, 182)]),
      caption = "first 6 rows of gdm.test after adding scores") %>%
  kable_styling(latex_options = "hold_position")
```

Table 12: first 6 rows of gdm.test after adding scores

scores.1	scores.2	scores.3
-0.2334714	-1.0420490	0.0000000
-0.8788912	-1.6874688	0.4740752
0.0000000	0.0000000	0.0000000
-0.4394456	-0.4394456	0.4740752
0.0000000	0.0000000	0.0000000
-0.8788912	-1.6874688	0.0000000

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

```r
#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
```

```r
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
```

```r
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

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

```r
#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 13: Confusion Matrix

	FALSE	TRUE
FALSE	2	27
TRUE	147	0

```r
#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",
      col.names = "Weight") %>%
  kable_styling(latex_options = "hold_position")
```

Table 14: Weight of Studies

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

```
#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 15: 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

According to `Lab5`, when genome-wide association studies of the same phenotype are performed in multiple independent studies, it is common to merge the results by performing a meta analysis. This help establish a statistical power and reducing the false-positive findings.

We harmonise two different studies and compute the confusion matrix to detect false-positive findings. In Table 13, there are two `SNPs` where the allele does not match with the corresponding allele in the other study. Hence, we removed these values and continues our analysis. We negated the values for `my.study` to keep the direction of the effect consistence.

Now we performed the meta analysis by computing the weights for each study. In Table 14, this weight is the inverse variance weighting where the bigger the value has bigger power. Then, we computed the `beta`, `std.error` and `p.value` for the following analysis. These values are then presented in Table 15.

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.

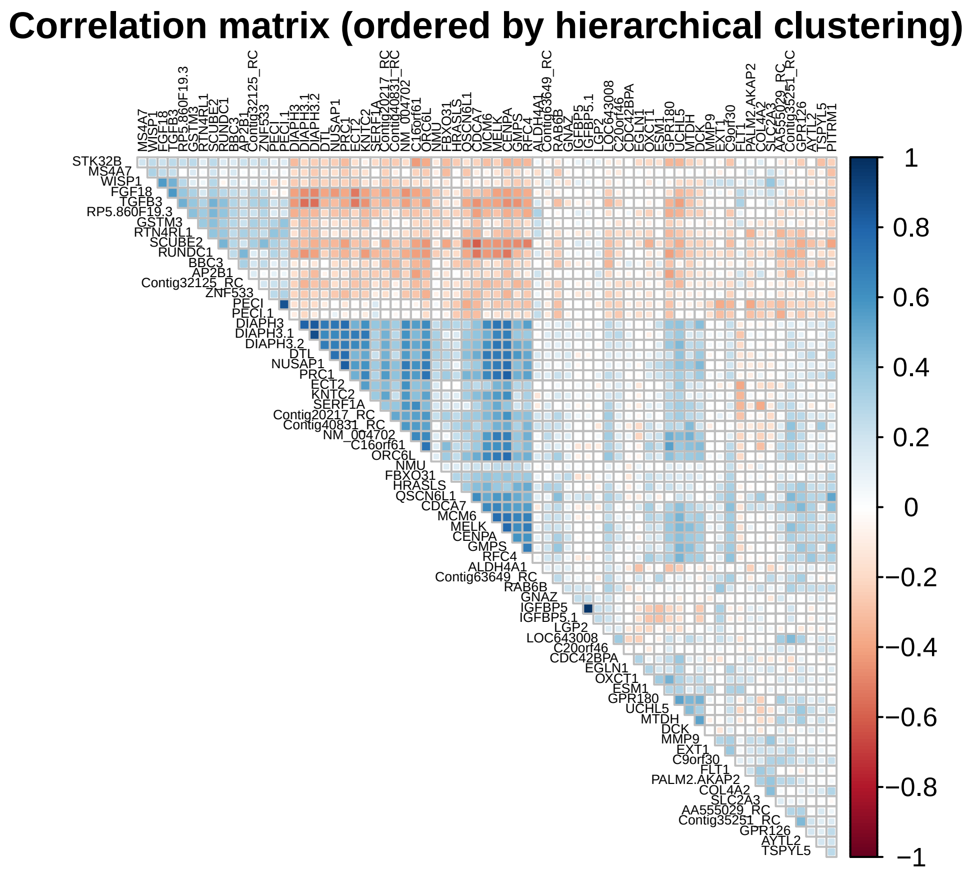
Problem 3.a (6 points)

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

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

```r
#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 16: 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

We computed the correlation between the features, and displayed as a correlation matrix with a block structure being highlighted with respect to its correlation coefficients. We can see that the blue parts represent a positive correlation where most of this is found in the region between the correlation of between the region of `DIAPH3-RFC4`. However, we observed negative correlation between the region of `STK32B-PECI.1` to `DIAPH3-RFC4`. Furthermore the positive and negative correlation is spread through the correlation matrix.

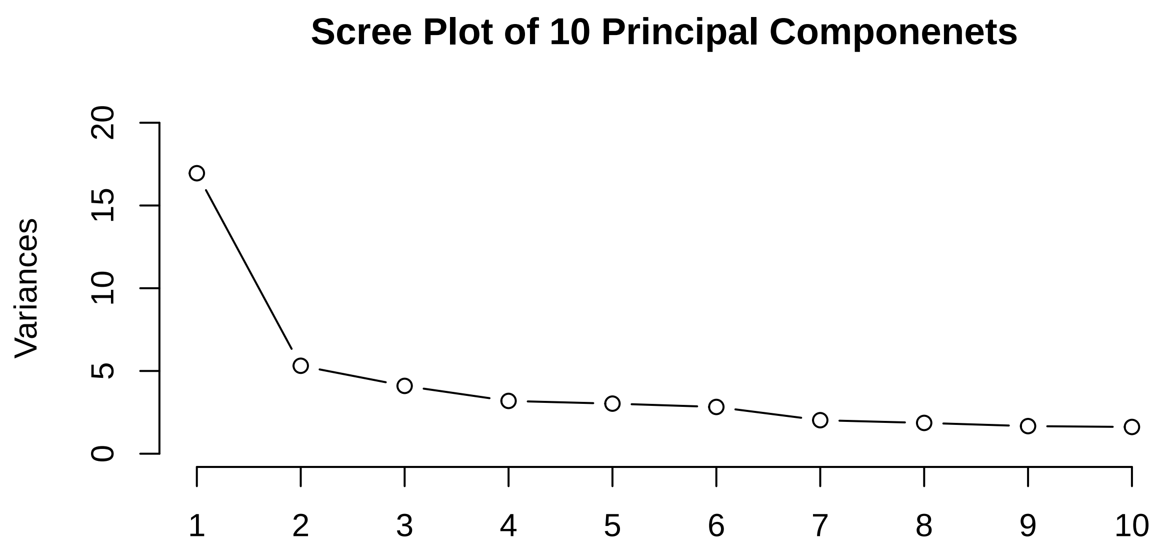
We also computed the genes that have high correlation among each other in Table 16. Interestingly, `DIAP3` is highly correlated to `DIAPH3.1` and `DIAPH3.2`. This is self-explanatory as `DIAP3` is the prefix. Similar observation is reported for `PECI` and `IFGBP5`. The correlation between `IFGBP5` and `IFGBP5.1` is 0.9775030 which is the strongest positive correlation while others are less than 0.9.

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.

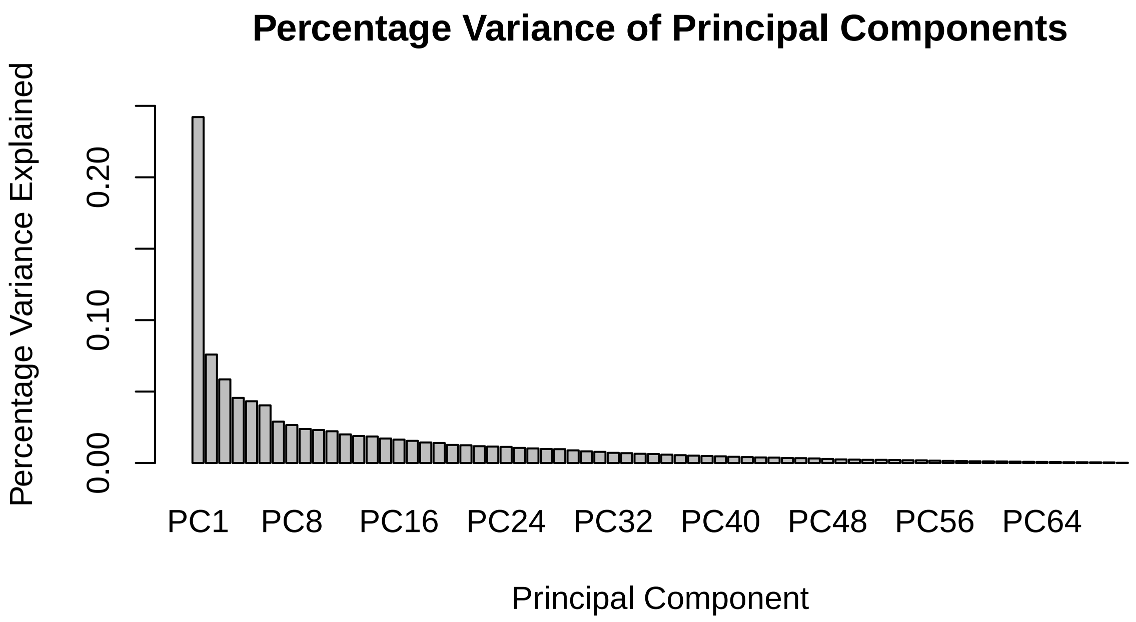
Answer

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

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

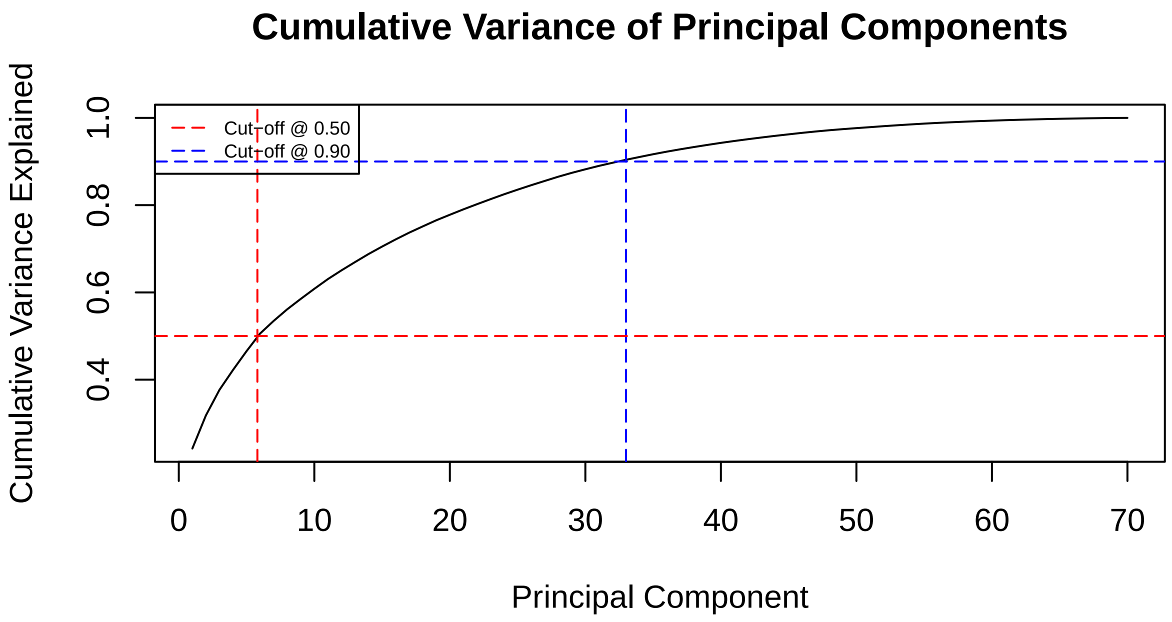


We conducted principal component analysis using `prcomp()`. To visualise the principal components and its variance, we plot the screen plot. We can see that the first principal component takes up the most variance as the gradient is the steepest. From the next component, we can see that the gradient become gentle. After component 3, we observe that the gradient does not change significantly. From here we may think that using 3 principal component is sufficient for the analysis. To further validate this, let use compute the percentage variance explained and cumulative variance explained by each principal component.

```r
#Percentage Variance explained by each Principal Component
barplot(summary(nki.pca)$importance[2,], ylim = c(0, 0.25),
        xlab = "Principal Component", ylab = "Percentage Variance Explained",
        main = "Percentage Variance of Principal Components")
```



```r
#Cumulative Variance explained by each Principal Component
plot(cumsum(nki.pca$sdev^2 / sum(nki.pca$sdev^2)), type = "line",
     xlab = "Principal Component", ylab = "Cumulative Variance Explained",
     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)
```



Looking at the two plots above, we can see that the first component has the highest percentage variance explained and the percentage explained is 24.2%. The next components adds up to 31.8% and 37.7% respectively from the percentage variance explained and cumulative variance explained plot. From component 4, we see that the increase in percentage is much smaller compared to the first three components. As a result, we will chose the first three components to continue our analysis with its visualisation in the next part.

```r
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")

#tabulating 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 17: 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

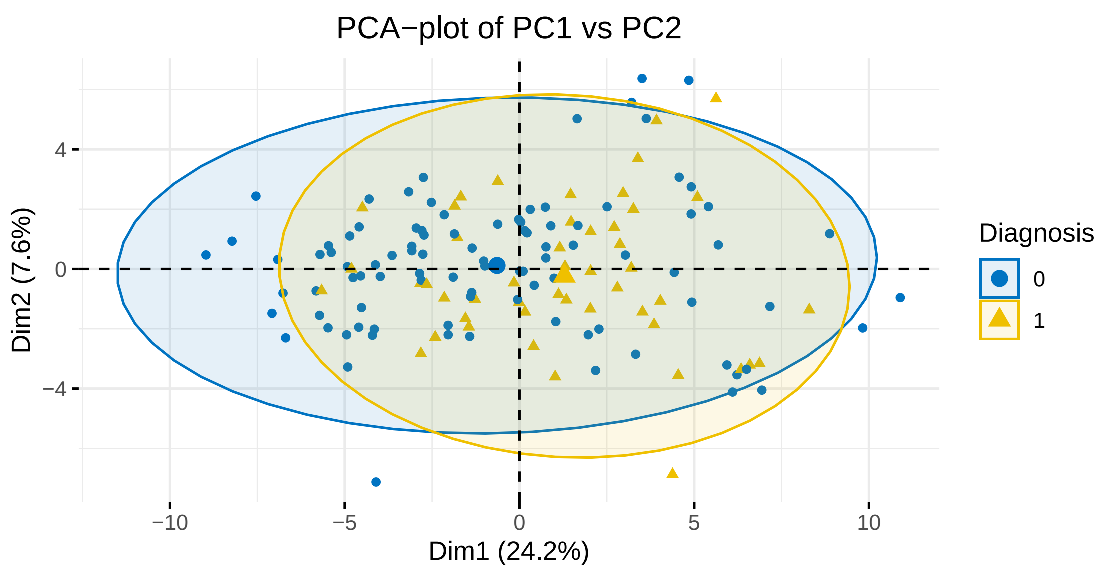
We conducted logistic regression for both unadjusted and adjusted model. The unadjusted model only contains each principal component whereas the adjusted model includes other variables such as `age`, `Grade` and `EstrogrenReceptor`. From Table 17, we can see that p-values for the first and third components in unadjusted model are 0.00943 and 0.00863 and they can be considered significant. For the adjusted model, only the third component is significant with the value of 0.0245. Thus, the unadjusted model has 2 components and adjusted model has 1 component that are significant. However, if we revisit the cumulative variance explained plot, we can see that the 3 components can only explain 37.6% of the variance. As a result we can infer that PCA is not so helpful in this model.

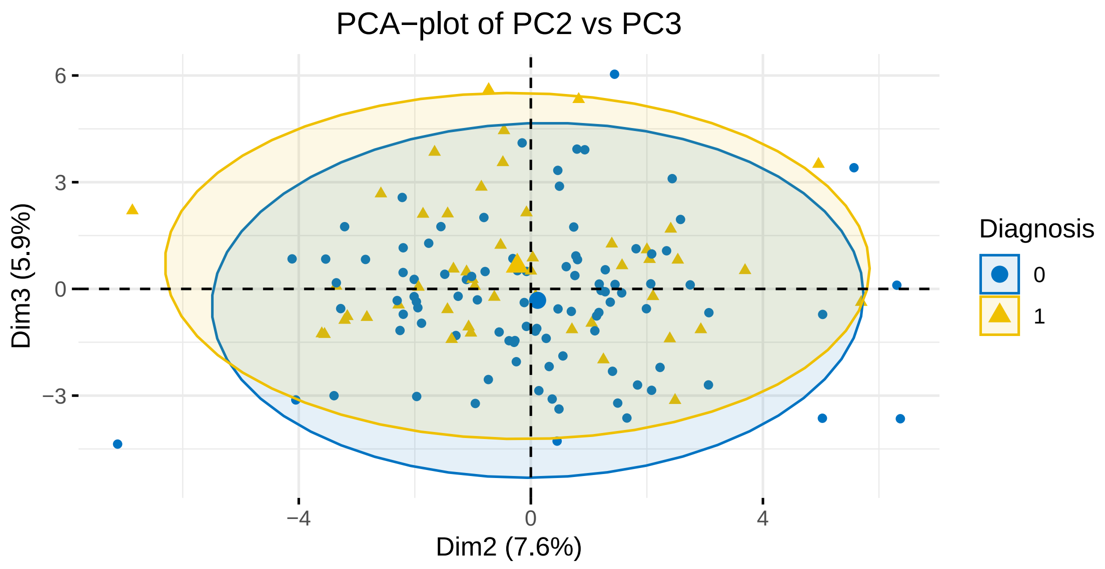
Problem 3.c (8 points)

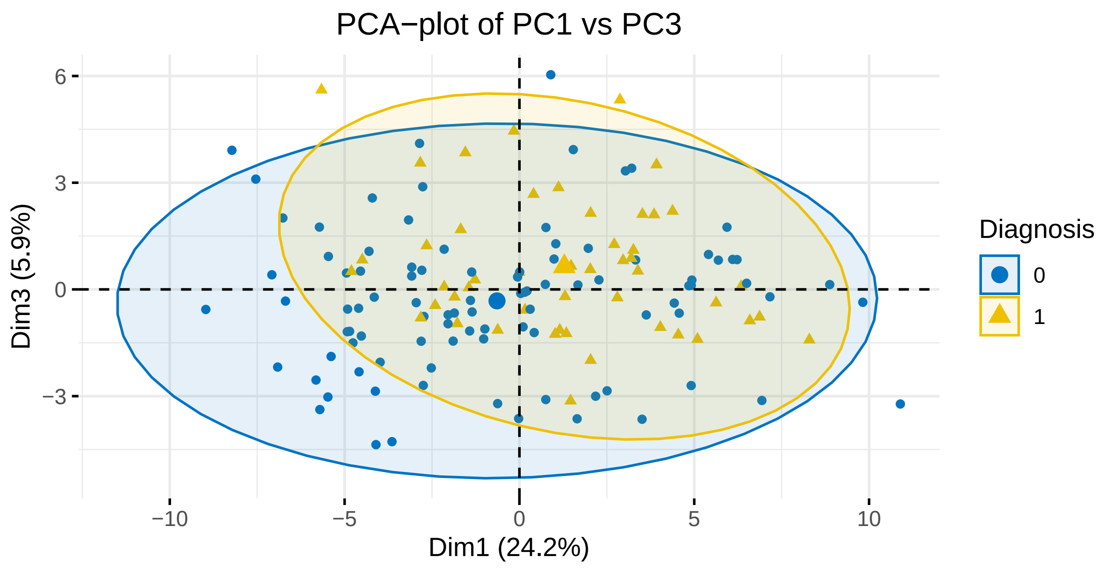
Use plots to compare with the correlation structure observed in problem 2.a and to examine how well the dataset may explain your outcome. Discuss your findings and suggest any further steps if needed.

Answer

```r
#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%)
4
0
−4
−10
−5
0
5
10
Diagnosis
0
1


PCA−plot of PC2 vs PC3
Dim3 (5.9%)
Dim2 (7.6%)
6
3
0
−3
−4
0
4
Diagnosis
0
1


PCA−plot of PC1 vs PC3
Dim3 (5.9%)
Dim1 (24.2%)
6
3
0
−3
−10
−5
0
5
10
Diagnosis
0
1

Now we plot the principal components against each other for the selected first three components. We first analyse the scatter plot against each component using `fviz_pca_ind()`.

The datapoints in the above plot are the individual counties. They were artificially grouped and coloured by quartile. The x-axis in the first plot is the first principle component and the value between brackets is the proportion of the variance it explains. The y-axis in the first plot is the second component and in the second graph the counties are projected on the plane formed by the second and third components. Same applies for the third plot. Looking at all plots we can see that the ellipses overlap. To overcome this we could just look at the largest and lowest quartiles.

Furthermore, each component simply describes the variation in a dataset. A dataset which contains a few variables that are associated with our outcome of interest and a lot more variables that aren't won't be very good at separating individual observations of the outcome. This isn't entirely the case with our dataset as while the ellipses overlap, for the first two principal components and first and third principal components show that they do separate slightly in the right order along the x and y axes.

Looking in closely, centroids are separated in the right order along the x-axis for the first principal component and marginally so along the y-axis for the second principal component. However, in the second graph the ellipses completely overlap. Removing a few variables which weren't associated with our outcome of interest in the correlation matrix may help improve matters. Finally, first and second components and first and third components contain the large variation in our data it is therefore a good idea to examine the subset of most extreme values for the first two components as they are likely to contain outliers.

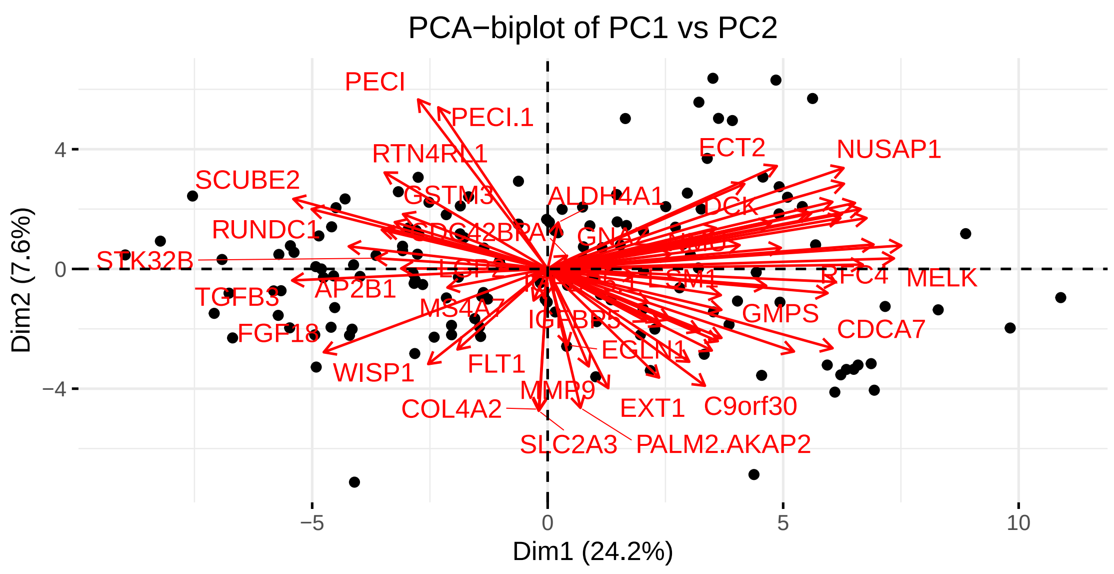
The plot below is known as biplot and we used `fviz_pca_biplot()` to have a visual representation of the linear composition of the components. The dots are the same datapoints then in the previous graph but without colour and each arrow represents the direction of each variable in the the 2D plane of the first three components.

For the first component we would look at the variables with greatest magnitude along the x-axis, i.e. furthest to the left and furthest to the right. The first component mainly assigns large positive values to indicators of `MELK` and `RFC4` and large negative values to indicators of `STK32B` and `TGFB3` from the first and thrid plots. The actual coefficients for each variable for the first two principal components is given below.

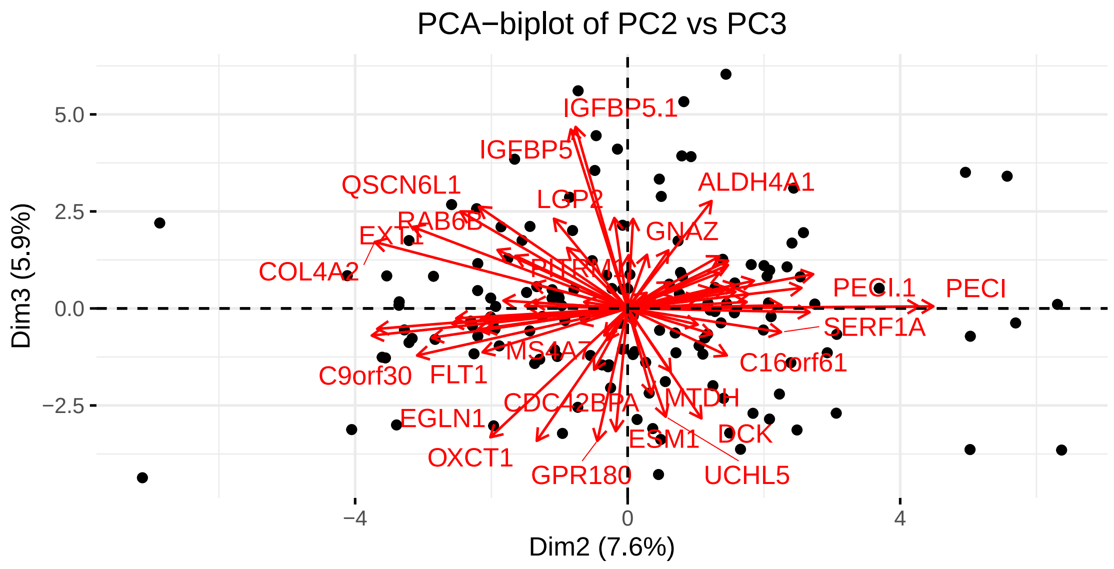
For the second component we look at the variables with the greatest magnitude along the y-axis, i.e. the highest and lowest variables. For the component we can see that `PECI`, `PECI.1`, `COL4A2` and `SLC2A3` carry the most weight from the first and second plots. This roughly follows the blocks that we observed in the correlation plot which isn't surprising given that the components are the eigenvectors of the variance-covariance matrix of the the predictor variables.

For the third component we look at the variables with the greatest magnitude along the y-axis. For the component, we can see that assignment on `IGFBP5`, `IGFBP5.1` for large positive values and `OXCT1` and `GPR180` for the large negative values from the second and third plots.

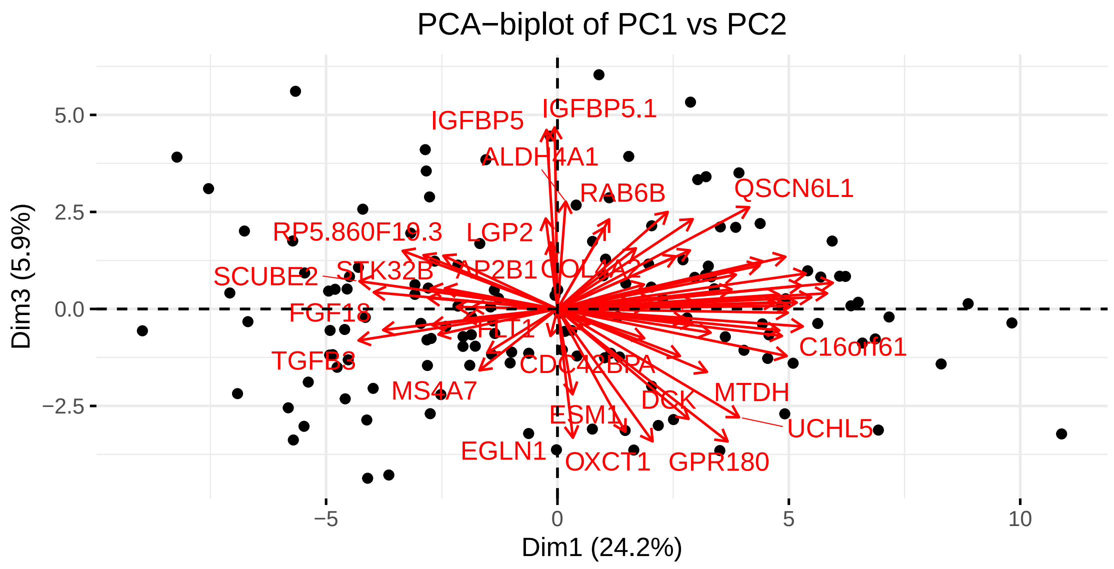
```r
#plotting biplot for each Principal Component
p1 <- fviz_pca_biplot(nki.pca, geom ='point', repel = T, col.var = "red", axes = c(1,2)) +
  ggtitle("PCA-biplot of PC1 vs PC2") + theme(plot.title = element_text(hjust = 0.5))
p2 <- fviz_pca_biplot(nki.pca, geom ='point', repel = T, col.var = "red", 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
SCUBE2
GSTM3
ALDH4A1
ECT2
NUSAP1
RUNDC1
CDC42BPA
GNAZ
DCK
STK32B
AP2B1
ESM1
CFC4
MELK
TGFB3
MS4A7
GMPS
FGF18
IGFBP5
CDCA7
WISP1
FLT1
EGLN1
MMP9
COL4A2
EXT1
C9orf30
SLC2A3
PALM2.AKAP2
Dim1 (24.2%)
Dim2 (7.6%)
−5
0
5
10
4
0
−4



PCA−biplot of PC2 vs PC3
IGFBP5.1
IGFBP5
QSCN6L1
ALDH4A1
LGP2
RAB6B
GNAZ
EXT1
COL4A2
PECI.1
PECI
SERF1A
MS4A7
C16orf61
C9orf30
FLT1
MTDH
CDC42BPA
EGLN1
DCK
ESM1
OXCT1
GPR180
UCHL5
Dim2 (7.6%)
Dim3 (5.9%)
−4
0
4
5.0
2.5
0.0
−2.5



PCA−biplot of PC1 vs PC2
IGFBP5
IGFBP5.1
ALDH4A1
RAB6B
QSCN6L1
RP5.860F19.3
LGP2
SCUBE2
STK32B
AP2B1
COL4A2
FGF18
FLT1
TGFB3
CDC42BPA
C16orf61
MS4A7
DCK
MTDH
ESM1
UCHL5
EGLN1
OXCT1
GPR180
Dim1 (24.2%)
Dim3 (5.9%)
−5
0
5
10
5.0
2.5
0.0
−2.5

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

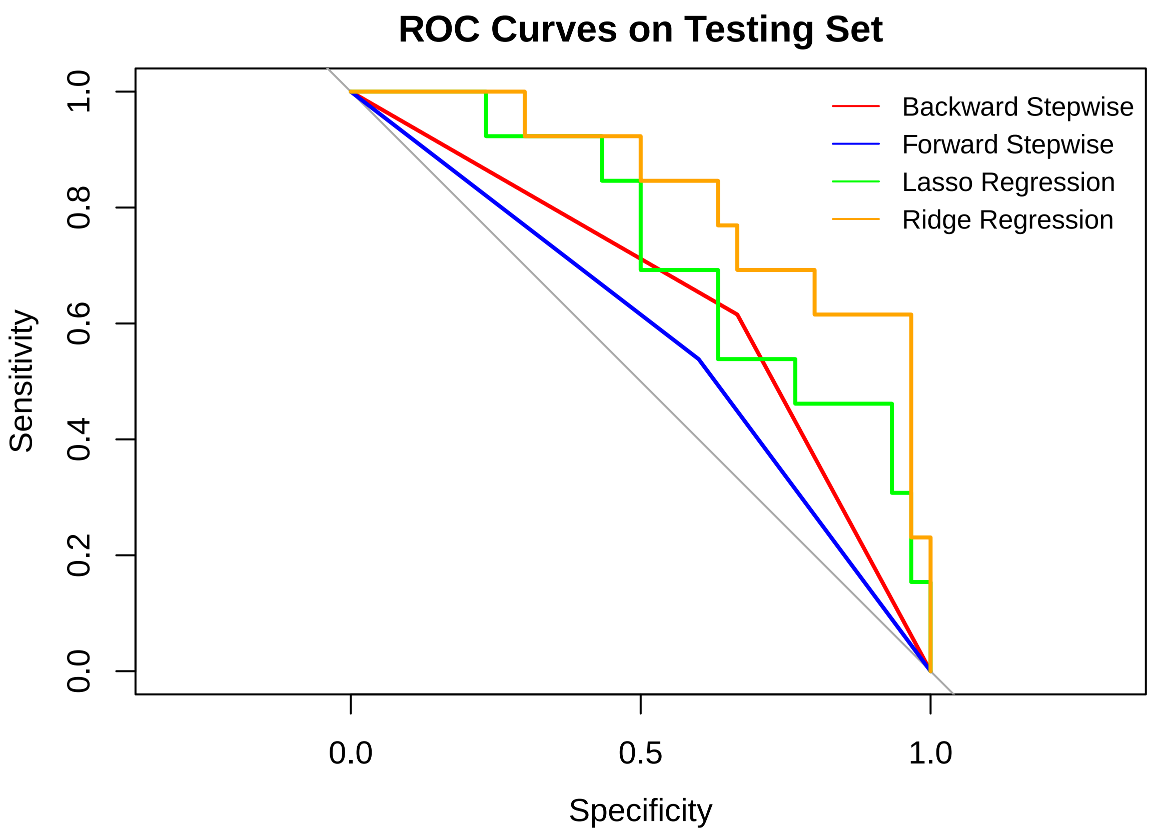
```r
prepare.glmnet <- function(data, formula=~ .) {
  ## create the design matrix to deal correctly with factor variables,
  ## without losing rows containing NAs
  old.opts <- options(na.action='na.pass')
  x <- model.matrix(formula, data)
  options(old.opts)
  ## remove the intercept column, as glmnet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

```r
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 multiple-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
})})
```

```r
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 model size
lambda.lasso.idx <- which(fit.lasso$lambda.min == fit.lasso$lambda)
lambda.ridge.idx <- which(fit.ridge$lambda.min == fit.ridge$lambda)
#model size of Lasso and Ridge regression of lambda min
```

```r
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("topright", 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 18: 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

As we mentioned in Q3 b), PCA does not seem to work well in this dataset as we need nearly 32 components so that can best represent the variance. Thus we attempted several analysis with Lasso, Ridge, backward and forward stepwise selection.

We preprocessed the data using our predefined function `prepare.glmnet()`. This function takes a dataframe as an input and conducts multiple hot encoding on the dataset i.e. changes the categorical variable to numerical variable.

First, split the dataset into training and testing dataset with a 7 : 3 ratio. After that we continue by conducting logistic regression using `glm()` to compute the full and null models on the training data. With these components, we can conduct backward and forward stepwise selection using `stepAIC()`. Secondly, we continued with the training data previously. We conduct a cross validation on the logistic regression with Lasso and Ridge penalties. Thus, we used `cv.glmnet()` with AUC as a measure.

With the models trained using the training dataset, we now tested on the testing set and plot of ROC curves for each model. From the plot, Backward Stepwsie, Forward Stepwise, Lasso and Ridge are denoted in red, blue, green, orange respectively. Just by looking at the ROC curve, we can see that Ridge regression has the highest AUC value. This is further evident in Table 18. where the AUC value for Ridge regression is 0.826. Since Ridge regression shrinks the regression coefficients to near zero it will employ all coefficients. As a result, this is not a simple model but we still believe Ridge outperforms other models as shown in Table 18. Thus, Ridge regression is the best model that best describes the dataset and a model that can be used to predict and diagnosis on breast cancer.