Assignment 2

Biomedical Data Science (MATH11174), 22/23, Semester 2

2023-04-06

Due on Thursday, 6th of April 2023, 5:00pm

Pay Attention

The assignment is marked out of 100 points, and will contribute to ***30%*** of your final mark. The aim of this assignment is to produce a precise report in biomedical studies with the help of statistical and machine learning. Please complete this assignment using **Quarto/Rmarkdown file and render/knit this document only in PDF format** (rendering while solving the questions will prevent sudden panic before submission!). Submit using the **gradescope link on Learn** and ensure that **all questions are tagged accordingly**. You can simply click render on the top left of Rstudio (`Ctrl+Shift+K`). If you cannot render/knit to PDF directly, open **Terminal** in your RStudio (`Alt+Shift+R`) and type `quarto tools install tinytex`, otherwise please follow this link. If you have any code that does not run you will not be able to render nor knit the document so comment it as you might still get some grades for partial code.

Codes that are **clear and reusable will be rewarded**. Codes without proper indentation, choice of variable identifiers, **comments**, efficient code, etc will be penalised. An initial code chunk is provided after each subquestion but **create as many chunks as you feel is necessary** to make a clear report. Add plain text explanations in between the chunks when required to make it easier to follow your code and reasoning. Ensure that all answers containing multiple values should be presented and formatted only with `kable()` and `kable_styling()` otherwise penalised (**no use of `print()` or `cat()`**). All plots must be displayed with clear title, label and legend otherwise penalised.

This is an **individual assignment**, and **no public discussions** will be allowed. If you have any question, please ask on Piazza by specifying your `Post to` option to `instructors`. To join Piazza, please follow this link.

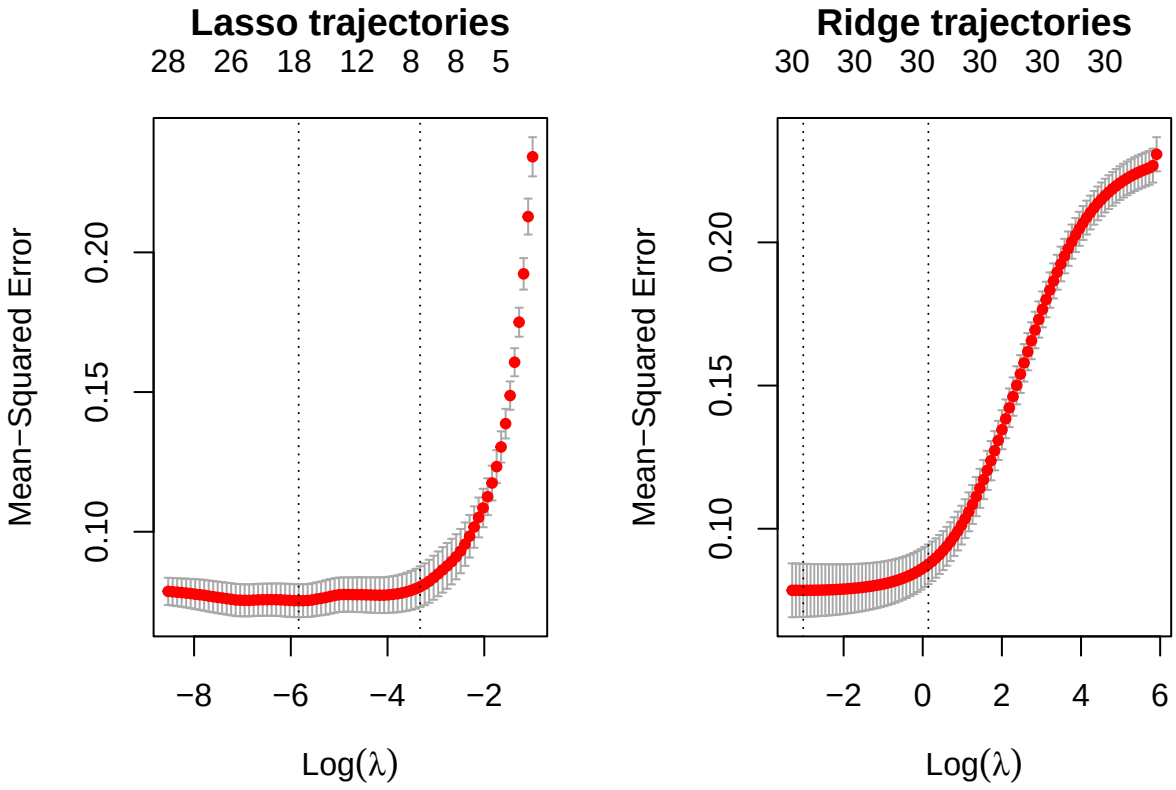
Problem 1 (27 points)

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 and Lasso regression model which use 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 maximises the AUC and report the λ for both ridge and Lasso regression using `kable()`.
- ***Note : there is no need to use the `prepare.glmnet()` function from lab 4, using `as.matrix()` with the required columns is sufficient.***

```r
wdbc2 <- fread("wdbc2.csv")[,-1]
wdbc2$diagnosis = ifelse(wdbc2$diagnosis=="malignant",1,0)
set.seed(984065)
trainIndex <- createDataPartition(wdbc2$diagnosis,
                                  p = 0.7, list = FALSE)

# Extract the training and testing datasets using the indexes
trainData <- wdbc2[trainIndex, ]
testData <- wdbc2[-trainIndex, ]

train_x = as.matrix(trainData[,-1])
train_y = as.matrix(trainData[, 1])
# Extract test sets for biomarkers X and for outcome Y from the test data
test_x = as.matrix(testData[,-1])
test_y = as.matrix(testData[,1])

fit.lasso <- cv.glmnet(train_x,train_y) # same as setting alpha=1
fit.ridge <- cv.glmnet(test_x, test_y, alpha=0)

par(mfrow=c(1,2), mar=c(4,4,5,2))
plot(fit.lasso, main="Lasso trajectories")
plot(fit.ridge, main="Ridge trajectories")
```




```r
# Extract the optimal lambdas and AUCs for both ridge and Lasso regression
opt_lambdas <- data.frame(method=c("Lasso", "Ridge"),
                          lambda_min=c(fit.lasso$lambda.min,
```

```
                                        fit.ridge$lambda.min),
                           lambda_1se=c(fit.lasso$lambda.1se,
                                        fit.ridge$lambda.1se),
                           AUC_min=c(fit.lasso$cvm[fit.lasso$lambda == fit.lasso$lambda.min],
                                     fit.ridge$cvm[fit.ridge$lambda == fit.ridge$lambda.min]),
                           AUC_1se=c(fit.lasso$cvm[fit.lasso$lambda == fit.lasso$lambda.1se],
                                     fit.ridge$cvm[fit.ridge$lambda == fit.ridge$lambda.1se]))
# Round the numerical values to 3 significant figures
opt_lambdas[,2:4] <- round(opt_lambdas[,2:4], 3)

# Print the table of optimal lambdas and AUCs
kable(opt_lambdas, align="c")
```

method	lambda_min	lambda_1se	AUC_min	AUC_1se
Lasso	0.003	0.036	0.075	0.0803235
Ridge	0.049	1.153	0.078	0.0875880

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** that contains the corresponding λ, AUC and model size.
- Use 3 significant figures for floating point values and comment on these results.
- ***Note : The AUC values are stored in the field called `cvm`.***

```
# Extract lambda.min and lambda.1se values for ridge and Lasso models
ridge_lambdas <- c(fit.ridge$lambda.min,
                   fit.ridge$lambda.1se)
lasso_lambdas <- c(fit.lasso$lambda.min,
                   fit.lasso$lambda.1se)

# Extract corresponding AUC values for ridge and Lasso models
ridge_auc <- c(max(fit.ridge$cvm),
               max(fit.ridge$cvm)-fit.ridge$cvsd[which.max(fit.ridge$cvm)])
lasso_auc <- c(max(fit.lasso$cvm),
               max(fit.lasso$cvm)-fit.lasso$cvsd[which.max(fit.lasso$cvm)])

# Extract corresponding model size for ridge and Lasso models
ridge_size <- c(sum(fit.ridge$glmnet.fit$beta!=0, na.rm=TRUE))
lasso_size <- c(sum(fit.lasso$glmnet.fit$beta!=0, na.rm=TRUE))

# Create a data table to show the results
result_table <- data.table(Model = c("Ridge", "Lasso"),
                           `Lambda.min` = round(c(ridge_lambdas[1],
                                                  lasso_lambdas[1]),3),
                           `Lambda.1se` = round(c(ridge_lambdas[2],
                                                  lasso_lambdas[2]),3),
                           AUC = round(c(ridge_auc[1],
                                         lasso_auc[1]),3),
                           `Model size` = round(c(ridge_size,
                                                  lasso_size),3))

# Display the result table
result_table
```

```
##    Model Lambda.min Lambda.1se   AUC Model size
```

```
## 1: Ridge      0.049      1.153 0.231       3000
## 2: Lasso      0.003      0.036 0.234       1207
```

Problem 1.c (7 points)

- Perform both backward (we denote this as **model B**) and forward (**model S**) stepwise selection on the same training set derived in **problem 1.a**. Mute all the trace by setting `trace = FALSE`.
- Report the variables selected and their standardised regression coefficients in increasing order of the absolute value of their standardised regression coefficient.
- Discuss the results and how the different variables entering or leaving the model influenced the final result.
- ***Note : You can mute the warning by assigning `{r warning = FALSE}` for the chunk title***

```
full.model <- lm(diagnosis ~ radius + texture + perimeter + area + smoothness +
    compactness + concavity + concavepoints + symmetry + fractaldimension, wdbc2)
model.back <- stepAIC(full.model, direction="back") # backward elimination
```

```
## Start:  AIC=-1404.91
## diagnosis ~ radius + texture + perimeter + area + smoothness +
##     compactness + concavity + concavepoints + symmetry + fractaldimension
##
##                    Df Sum of Sq    RSS     AIC
## - fractaldimension  1    0.0023 46.348 -1406.9
## - perimeter         1    0.0043 46.350 -1406.8
## <none>                          46.346 -1404.9
## - concavity         1    0.1643 46.510 -1404.9
## - symmetry          1    0.3288 46.675 -1402.9
## - smoothness        1    0.3942 46.740 -1402.1
## - compactness       1    0.4071 46.753 -1401.9
## - area              1    1.0986 47.445 -1393.6
## - concavepoints     1    1.6428 47.989 -1387.1
## - radius            1    2.3369 48.683 -1378.9
## - texture           1    3.9922 50.338 -1359.9
##
## Step:  AIC=-1406.88
## diagnosis ~ radius + texture + perimeter + area + smoothness +
##     compactness + concavity + concavepoints + symmetry
##
##                 Df Sum of Sq    RSS     AIC
## - perimeter      1    0.0033 46.352 -1408.8
## <none>                       46.348 -1406.9
## - concavity      1    0.1697 46.518 -1406.8
## - symmetry       1    0.3274 46.676 -1404.9
## - smoothness     1    0.4394 46.788 -1403.5
## - compactness    1    0.5690 46.917 -1401.9
## - area           1    1.0969 47.445 -1395.6
## - concavepoints  1    1.6463 47.995 -1389.0
## - radius         1    2.3645 48.713 -1380.6
## - texture        1    3.9957 50.344 -1361.8
##
## Step:  AIC=-1408.84
## diagnosis ~ radius + texture + area + smoothness + compactness +
##     concavity + concavepoints + symmetry
##
##                 Df Sum of Sq    RSS     AIC
```

```
## <none>                       46.352 -1408.8
## - concavity      1    0.1665 46.518 -1408.8
## - symmetry       1    0.3251 46.677 -1406.9
## - smoothness     1    0.4363 46.788 -1405.5
## - compactness    1    0.5700 46.922 -1403.9
## - area           1    1.5083 47.860 -1392.6
## - concavepoints  1    1.8309 48.182 -1388.8
## - texture        1    3.9939 50.345 -1363.8
## - radius         1    5.8966 52.248 -1342.7
```

```
null.model <- lm(diagnosis ~ 1, data=wdbc2) # only include the intercept
sel.forw <- stepAIC(null.model, scope=list(upper=full.model), direction="forward")
```

```
## Start:  AIC=-827.22
## diagnosis ~ 1
##
##                    Df Sum of Sq     RSS      AIC
## + concavepoints     1    73.807  58.688 -1288.56
## + perimeter         1    67.121  65.374 -1227.17
## + radius            1    66.508  65.988 -1221.86
## + concavity         1    60.705  71.790 -1173.90
## + area              1    58.630  73.866 -1157.69
## + compactness       1    43.962  88.534 -1054.62
## + texture           1    21.484 111.012  -925.88
## + smoothness        1    16.717 115.779  -901.96
## + symmetry          1    13.871 118.625  -888.14
## <none>                          132.496  -827.22
## + fractaldimension  1     0.001 132.495  -825.22
##
## Step:  AIC=-1288.56
## diagnosis ~ concavepoints
##
##                    Df Sum of Sq    RSS     AIC
## + radius            1    5.5657 53.122 -1343.3
## + texture           1    5.1547 53.533 -1338.9
## + perimeter         1    3.2791 55.409 -1319.3
## + fractaldimension  1    2.0321 56.656 -1306.6
## + area              1    1.2238 57.464 -1298.5
## + smoothness        1    0.6440 58.044 -1292.8
## + compactness       1    0.5740 58.114 -1292.2
## <none>                         58.688 -1288.6
## + symmetry          1    0.0050 58.683 -1286.6
## + concavity         1    0.0001 58.688 -1286.6
##
## Step:  AIC=-1343.26
## diagnosis ~ concavepoints + radius
##
##                    Df Sum of Sq    RSS     AIC
## + texture           1    3.6072 49.515 -1381.3
## + area              1    2.0293 51.093 -1363.4
## + perimeter         1    1.0110 52.111 -1352.2
## + symmetry          1    0.7350 52.387 -1349.2
## + smoothness        1    0.3794 52.743 -1345.3
## + concavity         1    0.3287 52.794 -1344.8
## <none>                         53.122 -1343.3
```

```
## + fractaldimension  1    0.1738 52.949 -1343.1
## + compactness       1    0.1318 52.991 -1342.7
##
## Step:  AIC=-1381.27
## diagnosis ~ concavepoints + radius + texture
##
##                    Df Sum of Sq    RSS     AIC
## + area              1   2.05099 47.464 -1403.3
## + perimeter         1   1.02291 48.492 -1391.2
## + smoothness        1   0.90256 48.613 -1389.7
## + symmetry          1   0.76487 48.750 -1388.1
## + fractaldimension  1   0.19447 49.321 -1381.5
## <none>                          49.515 -1381.3
## + concavity         1   0.10423 49.411 -1380.5
## + compactness       1   0.04885 49.466 -1379.8
##
## Step:  AIC=-1403.34
## diagnosis ~ concavepoints + radius + texture + area
##
##                    Df Sum of Sq    RSS     AIC
## + smoothness        1   0.36303 47.101 -1405.7
## + symmetry          1   0.28536 47.179 -1404.8
## + compactness       1   0.16904 47.295 -1403.4
## <none>                          47.464 -1403.3
## + perimeter         1   0.02784 47.436 -1401.7
## + fractaldimension  1   0.01085 47.453 -1401.5
## + concavity         1   0.00405 47.460 -1401.4
##
## Step:  AIC=-1405.71
## diagnosis ~ concavepoints + radius + texture + area + smoothness
##
##                    Df Sum of Sq    RSS     AIC
## + compactness       1  0.251808 46.849 -1406.8
## + symmetry          1  0.179077 46.922 -1405.9
## <none>                          47.101 -1405.7
## + fractaldimension  1  0.079733 47.022 -1404.7
## + perimeter         1  0.014521 47.087 -1403.9
## + concavity         1  0.002021 47.099 -1403.7
##
## Step:  AIC=-1406.76
## diagnosis ~ concavepoints + radius + texture + area + smoothness +
##     compactness
##
##                    Df Sum of Sq    RSS     AIC
## + symmetry          1   0.33142 46.518 -1408.8
## + concavity         1   0.17278 46.677 -1406.9
## <none>                          46.849 -1406.8
## + perimeter         1   0.00952 46.840 -1404.9
## + fractaldimension  1   0.00783 46.842 -1404.8
##
## Step:  AIC=-1408.8
## diagnosis ~ concavepoints + radius + texture + area + smoothness +
##     compactness + symmetry
##
```

```
##                     Df Sum of Sq    RSS     AIC
## + concavity          1  0.166491 46.352 -1408.8
## <none>                           46.518 -1408.8
## + fractaldimension   1  0.007663 46.510 -1406.9
## + perimeter          1  0.000074 46.518 -1406.8
##
## Step:  AIC=-1408.84
## diagnosis ~ concavepoints + radius + texture + area + smoothness +
##     compactness + symmetry + concavity
##
##                     Df Sum of Sq    RSS     AIC
## <none>                           46.352 -1408.8
## + perimeter          1 0.0032749 46.348 -1406.9
## + fractaldimension   1 0.0012823 46.350 -1406.8
```

Problem 1.d (3 points)

- Compare the goodness of fit of **model B** and **model S**
- Interpret and explain the results you obtained.
- Report the values using `kable()`.

Problem 1.e (2 points)

- Plot the ROC curve of the trained model for both **model B** and **model S**. Display with clear title, label and legend.
- Report AUC values in 3 significant figures for both **model B** and **model S** using `kable()`.
- Discuss which model has a better performance.

Problem 1.f (6 points)

- Use the four models to predict the outcome for the observations in the test set (use the λ 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.
- Display with clear title, label and legend.
- 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 in this chunk
```

Problem 2 (40 points)

File `GDM.raw.txt` (available from the accompanying zip folder on Learn) contains 176 `SNP`s 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 in file `GDM.raw.txt` into a data table named `gdm.dt`.

```
gdm.dt <- setDT(fread("GDM.raw.txt"))
```

- Impute missing values in `gdm.dt` according to `SNP`-wise median `allele` count.

```
kable(table(is.na(gdm.dt)),
      caption = "Missing Values") |>
  kable_styling(full_width = F,
                position = "center",
                latex_options = "hold_position")
```

Table 1: Missing Values

Var1	Freq
FALSE	140582
TRUE	649

```
for (colnm in colnames(gdm.dt[, -1])) {
  gdm.dt[[colnm]][is.na(gdm.dt[[colnm]])] <- mean(gdm.dt[[colnm]], na.rm = T)
}
{rs7513574_T <- lm(pheno ~ rs7513574_T, data = gdm.dt)}
kable(coef(summary(rs7513574_T)), caption = "Summary Statistics") |>
  kable_styling(full_width = F, position = "center", latex_options = "hold_position")
```

Table 2: Summary Statistics

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	0.5298927	0.0262469	20.1887885	0.0000000
rs7513574_T	-0.0001470	0.0262289	-0.0056047	0.9955295

- Display first 10 rows and first 7 columns using `kable()`.

```
kable(gdm.dt[1:10,0:7])
```

ID	sex	pheno	rs7513574_T	rs1627238_A	rs1171278_C	rs1137100_A
1	0	0	1	0	0	2
2	0	0	0	0	0	1
4	0	1	2	1	1	1
5	0	1	0	1	1	1
6	0	1	0	1	1	1
7	0	0	1	1	1	0
8	0	0	0	0	0	1
12	0	1	1	1	1	1
13	0	1	2	0	0	2
18	0	0	1	0	0	0

Problem 2.b (8 points)

- Write function `univ.glm.test()` where it takes 3 arguements, `x`, `y` and `order`.
- `x` is a data table of `SNP`s, `y` is a binary outcome vector, and `order` is a boolean which takes `false` as a default value.
- 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.

```
univ.glm.test <- function(x, y,order=FALSE) {
  regr <- glm(y ~ .,data=x,family = "binomial")
  output <- data.table(coef(summary(regr)))#$coefficients)
  output<-cbind(colnames(x),output[-1,1], exp(coef(regr))[-1], output[-1,c(2,4)])
  colnames(output)<-c("snp","coefficients","odds_ratios","standard_errors","pvalues")
  if(order){
    my_matrix[order(my_matrix[, -1]), ]
  }
  output
}
```

Problem 2.c (5 points)

- Using function `univ.glm.test()`, run an association study for all the `SNP`s in `gdm.dt` against having gestational diabetes (column `pheno`) and name the output data table as `gdm.as.dt`.
- Print the first 10 values of the output from `univ.glm.test()` using `kable()`.
- 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 using `kable()` from the GWAS.
- Report the 95% and 99% confidence intervals on the odds ratio using `kable()`.

Problem 2.d (4 points)

- 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 `SNP`s that have p-value $< 10^{-4}$ (`hit SNP`s) report `SNP` name, effect `allele`, chromosome number, corresponding `gene` name and `pos`.
- Using `kable()`, report for each `snp.hit` the names of the genes that are within a 1Mb window from the `SNP` position on the chromosome.
- ***Note: That are genes that fall within +/- 1,000,000 positions using the pos column in the dataset.***

“{r} gdm.annot <- fread(“GDM.annot.txt”) gdm gene.indx <- c()

for (i in 1:nrow(gdm.as.dt)){ snp.id <-gsub(“_.”,“ “,gdm.as.dt$snp[i]) gene.indx <- -append(gene.indx, which(snp.id == gdm.annot$snp)) }

gene <- gdm.annot[gene.indx,“gene”]

gdm.as.dt <- cbind(gene,gdm.as.dt)

snp.hit <- gdm.as.dt[gdm.as.dt$pvalues < 1e-4,]

snp.hit <- snp.hit$snp

allele <- gsub(“_.”,“ “,snp.hit)

snp.n <- gsub(“_.”,“ “,snp.hit)

other <- gdm.annot[gdm.annot$snp %in% snp.n,]

snp.hit <- cbind(snp.hit,allele, other[,-“snp”])

within <- NA

for(i in 1:nrow(snp.hit)){ pos <- snp.hit$pos[i]$abwindow < -abs(gdm.annot$pos - pos)

within[i] <- gdm.annot[abwindow <= 1000000]$gene$gene[i]] %>% paste0(., collapse = ",") } snp.hit <- cbind(snp.hit,within)

snp.hit <- snp.hit[,list(snp.hit,effect_allele =allele, chrom,pos,gene,within_1Mb_gene=within)]

kable(snp.hit, caption = "SNPs have p < 1e-4" |> kable_styling(full_width = F,position = "center",latex_option = "hold_position"))

```
## Problem 2.e (8 points)

-   Build a weighted genetic risk score that includes all `SNP`s with p-value $< 10^{-4}$, a score with
-   ***Hint: ensure that the ordering of `SNP`s 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 gestatio
-   Report odds ratio, $95\%$ confidence interval and p-value using `kable()` for each score.


```r
## Answer in this chunk
```

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 `SNP`s and corresponding weights.
- Add the three scores as columns to `gdm.test` ***(hint: use the same columnnames as before).***

```
## Answer in this chunk
```

Problem 2.g (4 points)

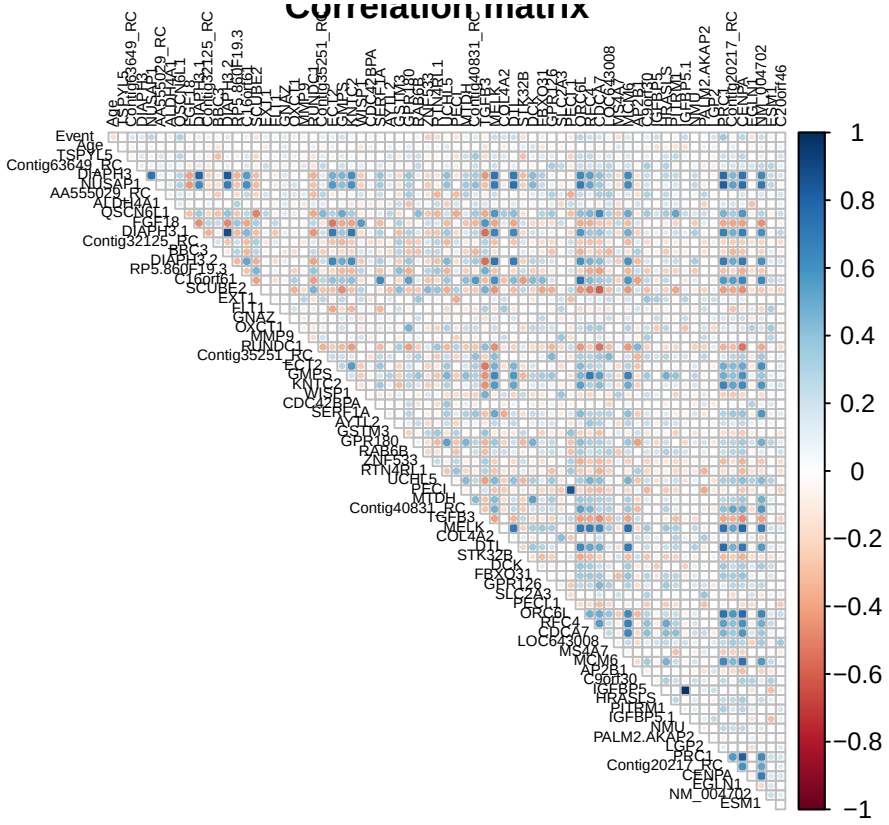
- Use the logistic regression models fitted in **problem 2.e** to predict the outcome of patients in `gdm.test`.
- Compute the test log-likelihood for the predicted probabilities from the three genetic risk score models and present them using `kable()`

```
#Answer in this chunk
```

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 `SNP`s.
- Perform a meta-analysis with the results obtained in **problem 2.c** (***hint : remember that the effect alleles should correspond***)
- Produce a summary of the meta-analysis results for the set of `SNP`s with meta-analysis p-value $< 10^{-4}$ sorted by increasing p-value using `kable()`.

```
#Answer in this chunk
```
```

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 correlation matrix between the gene expression variables, and display it so that a block structure is highlighted using the `corrplot` package.

```
nki <- fread("nki.csv")
genes <- nki[,6:76]
numcols <- sapply(nki, is.numeric)
cor.genes <- nki[, ..numcols] %>% #subset of numeric columns
             cor(use="pairwise.complete")
dim(cor.genes)
```

```
## [1] 72 72
```

- ```
  corrplot(cor.genes,
           # remove the diagonal elements
           diag=FALSE,
           # change the colour and size of the labels
           tl.col="black", tl.cex = 0.5,
           title="Correlation matrix",
           # display the upper triangle only
           type = 'upper',
           # change the size of the margins (bottom, left, top, right)
           mar=c(0,0,0,0))
  ```
  ```



- Discuss what you observe.

  ANS: In corrplot, each small rectangle represents the correlation between two variables, with color indicating the strength of the correlation. Red indicates positive correlation, while blue indicates negative correlation. A color bar can be used to view the correlation values represented by the colors.

- Identify the unique pairs of (distinct) variables that have correlation coefficient greater than 0.80 in absolute value and report their correlation coefficients.

```
# Identify pairs of variables with correlation coefficient > 0.8
high.corr.pairs <- which(abs(cor.genes) > 0.8
                         & upper.tri(cor.genes,
                                     diag = FALSE),
                         arr.ind = TRUE)

corr.pairs.values.08 <- cor.genes[high.corr.pairs]
# Return a table contains the locations and the correlations.
cbind(high.corr.pairs, corr.pairs.values.08)
```

```
##           row col corr.pairs.values.08
## DIAPH3      5  11            0.8031368
## DIAPH3      5  14            0.8338591
## DIAPH3.1   11  14            0.8868741
## PECI       38  50            0.8697836
## IGFBP5     59  62            0.9775030
## NUSAP1      6  66            0.8298356
## PRC1       66  68            0.8175424
```

Some explanation for the table above:

The 4^{th} column represents the value of correlation coefficient; the 1^{st} column shows the name of the selected array, and the row location and column location for the coefficient were shown in the second and third column.

Problem 3.b (8 points)

- Perform PCA analysis (only over the columns containing gene expressions) in order to derive a patient-wise summary of all gene expressions (dimensionality reduction).

```
apply(nki, 2, is.na) %>% colSums() %>% sort
```

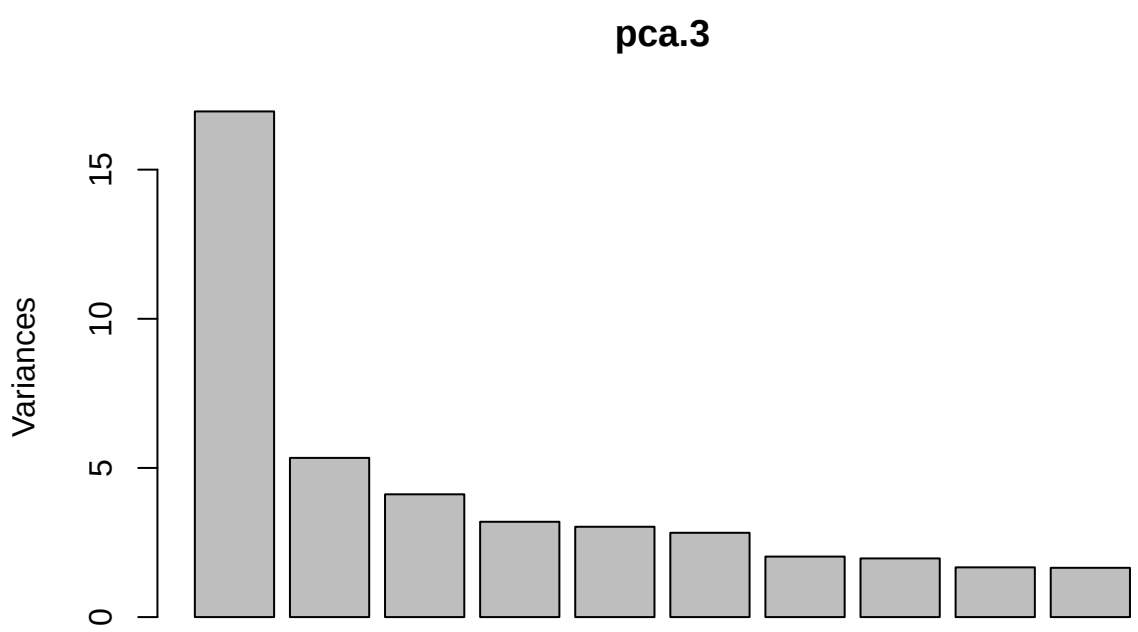
```
##            Event             Diam       LymphNodes EstrogenReceptor
##                0                0                0                0
##            Grade              Age           TSPYL5   Contig63649_RC
##                0                0                0                0
##           DIAPH3           NUSAP1      AA555029_RC          ALDH4A1
##                0                0                0                0
##          QSCN6L1            FGF18        DIAPH3.1   Contig32125_RC
##                0                0                0                0
##             BBC3         DIAPH3.2     RP5.860F19.3         C16orf61
##                0                0                0                0
##           SCUBE2             EXT1             FLT1             GNAZ
##                0                0                0                0
##            OXCT1             MMP9           RUNDC1   Contig35251_RC
##                0                0                0                0
##             ECT2             GMPS            KNTC2            WISP1
##                0                0                0                0
##         CDC42BPA           SERF1A            AYTL2            GSTM3
##                0                0                0                0
##           GPR180            RAB6B           ZNF533          RTN4RL1
##                0                0                0                0
##            UCHL5             PECI             MTDH   Contig40831_RC
##                0                0                0                0
##            TGFB3             MELK           COL4A2              DTL
##                0                0                0                0
##           STK32B              DCK           FBXO31           GPR126
##                0                0                0                0
##           SLC2A3           PECI.1            ORC6L             RFC4
##                0                0                0                0
##            CDCA7        LOC643008            MS4A7             MCM6
##                0                0                0                0
##            AP2B1          C9orf30           IGFBP5           HRASLS
##                0                0                0                0
##           PITRM1         IGFBP5.1              NMU      PALM2.AKAP2
##                0                0                0                0
##             LGP2             PRC1   Contig20217_RC            CENPA
##                0                0                0                0
##            EGLN1         NM_004702             ESM1         C20orf46
##                0                0                0                0
```

By checking the result above, we can see that the genes data set contains 0 missing values, in other words, genes data set is a complete data set. Then we can perform Principal Component Analysis by following process:

```
# Perform the PCA
pca.3 <- prcomp(genes,center = T,scale. = T)
summary(pca.3)
```

```
## Importance of components:
##                            PC1     PC2     PC3     PC4     PC5    PC6     PC7
## Standard deviation     4.1172 2.31010 2.02869 1.78758 1.73997 1.6810 1.42418
## Proportion of Variance 0.2387 0.07516 0.05797 0.04501 0.04264 0.0398 0.02857
## Cumulative Proportion  0.2387 0.31392 0.37188 0.41689 0.45953 0.4993 0.52790
##                            PC8     PC9    PC10    PC11    PC12    PC13    PC14
## Standard deviation     1.40252 1.29120 1.28546 1.26842 1.21396 1.18307 1.14124
## Proportion of Variance 0.02771 0.02348 0.02327 0.02266 0.02076 0.01971 0.01834
## Cumulative Proportion  0.55560 0.57908 0.60236 0.62502 0.64577 0.66549 0.68383
##                           PC15    PC16   PC17    PC18    PC19    PC20    PC21
## Standard deviation     1.09847 1.08009 1.0591 1.00933 0.99123 0.95170 0.93355
## Proportion of Variance 0.01699 0.01643 0.0158 0.01435 0.01384 0.01276 0.01227
## Cumulative Proportion  0.70083 0.71726 0.7331 0.74740 0.76124 0.77400 0.78627
##                           PC22    PC23    PC24    PC25    PC26    PC27    PC28
## Standard deviation     0.92516 0.89693 0.89015 0.88453 0.85741 0.83358 0.82514
## Proportion of Variance 0.01206 0.01133 0.01116 0.01102 0.01035 0.00979 0.00959
## Cumulative Proportion  0.79833 0.80966 0.82082 0.83184 0.84219 0.85198 0.86157
##                           PC29    PC30   PC31    PC32    PC33    PC34    PC35
## Standard deviation     0.79633 0.76445 0.7538 0.72723 0.70422 0.68396 0.66701
## Proportion of Variance 0.00893 0.00823 0.0080 0.00745 0.00698 0.00659 0.00627
## Cumulative Proportion  0.87050 0.87873 0.8867 0.89418 0.90117 0.90776 0.91402
##                           PC36    PC37    PC38    PC39    PC40    PC41    PC42
## Standard deviation     0.66178 0.62098 0.59979 0.59248 0.58190 0.56997 0.54785
## Proportion of Variance 0.00617 0.00543 0.00507 0.00494 0.00477 0.00458 0.00423
## Cumulative Proportion  0.92019 0.92562 0.93069 0.93564 0.94040 0.94498 0.94921
##                           PC43   PC44   PC45    PC46    PC47   PC48    PC49
## Standard deviation     0.52336 0.5193 0.4985 0.49317 0.48251 0.4617 0.43447
## Proportion of Variance 0.00386 0.0038 0.0035 0.00343 0.00328 0.0030 0.00266
## Cumulative Proportion  0.95307 0.9569 0.9604 0.96379 0.96707 0.9701 0.97273
##                           PC50   PC51    PC52    PC53    PC54    PC55   PC56
## Standard deviation     0.40946 0.3954 0.39282 0.38812 0.38210 0.36302 0.3471
## Proportion of Variance 0.00236 0.0022 0.00217 0.00212 0.00206 0.00186 0.0017
## Cumulative Proportion  0.97509 0.9773 0.97947 0.98159 0.98364 0.98550 0.9872
##                           PC57    PC58    PC59    PC60    PC61    PC62    PC63
## Standard deviation     0.3370 0.31981 0.30723 0.28880 0.27833 0.27243 0.25439
## Proportion of Variance 0.0016 0.00144 0.00133 0.00117 0.00109 0.00105 0.00091
## Cumulative Proportion  0.9888 0.99024 0.99157 0.99274 0.99383 0.99488 0.99579
##                           PC64    PC65    PC66    PC67    PC68    PC69   PC70
## Standard deviation     0.23727 0.22915 0.21208 0.19884 0.19092 0.17722 0.1676
## Proportion of Variance 0.00079 0.00074 0.00063 0.00056 0.00051 0.00044 0.0004
## Cumulative Proportion  0.99658 0.99732 0.99796 0.99851 0.99903 0.99947 0.9999
##                           PC71
## Standard deviation     0.09814
## Proportion of Variance 0.00014
## Cumulative Proportion  1.00000
```

```
plot(pca.3)
```

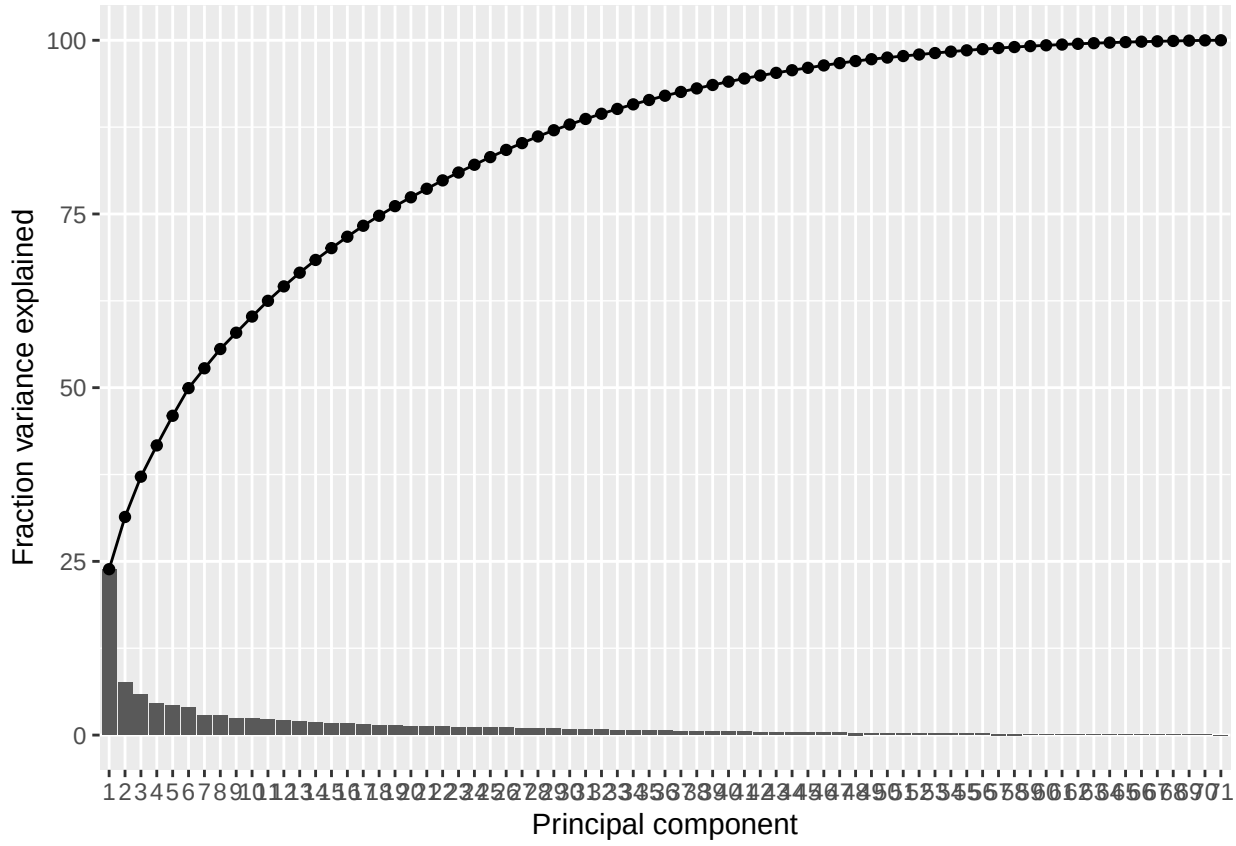



```
pc_eigenvalues <- pca.3$sdev^2
pc_eigenvalues <- tibble(PC = factor(1:length(pc_eigenvalues)),
                         variance = pc_eigenvalues) %>%
  # add a new column with the percent variance
  mutate(pct = variance/sum(variance)*100) %>%
  # add another column with the cumulative variance explained
  mutate(pct_cum = cumsum(pct))
# Print the result
pc_eigenvalues
```

```
## # A tibble: 71 x 4
##    PC    variance   pct pct_cum
##    <fct>    <dbl> <dbl>   <dbl>
##  1 1        17.0  23.9     23.9
##  2 2         5.34  7.52    31.4
##  3 3         4.12  5.80    37.2
##  4 4         3.20  4.50    41.7
##  5 5         3.03  4.26    46.0
##  6 6         2.83  3.98    49.9
##  7 7         2.03  2.86    52.8
##  8 8         1.97  2.77    55.6
##  9 9         1.67  2.35    57.9
## 10 10        1.65  2.33    60.2
## # i 61 more rows
```

```
pc_eigenvalues %>%
  ggplot(aes(x = PC)) +
```

```
  geom_col(aes(y = pct)) +
  geom_line(aes(y = pct_cum,
                group = 1)) +
  geom_point(aes(y = pct_cum)) +
  labs(x = "Principal component",
       y = "Fraction variance explained")
```




```
pc_scores <- pca.3$x
pc_scores <- pc_scores %>%
  # convert to a tibble retaining the sample names as a new column
  as_tibble(rownames = "sample")

# print the result
pc_scores
```

```
## # A tibble: 144 x 72
##    sample     PC1    PC2    PC3     PC4    PC5    PC6     PC7     PC8     PC9
##    <chr>    <dbl>  <dbl>  <dbl>   <dbl>  <dbl>  <dbl>   <dbl>   <dbl>   <dbl>
##  1 1      -4.91    0.190 -1.30   1.91    0.157 -1.38   0.242  -0.302  -0.875
##  2 2       1.47    1.53   0.704  1.98   -1.41   0.421 -0.771  -0.0618 -0.0198
##  3 3      -3.16    2.64   1.89   0.0103 -2.73   0.814 -2.42    0.671  -1.20
##  4 4      -2.15    1.79   1.16  -0.187   0.924  0.559  1.10    0.571   1.75
##  5 5       4.94   -1.10   0.244 -2.04    1.77  -3.41  -0.0687 -0.665  -2.37
##  6 6       0.0405  1.60  -0.102 -4.07   -1.50  -0.111  0.508   1.91   -2.32
##  7 7      -1.28   -1.09   0.336 -1.03   -0.486  0.689  0.365  -0.113  -0.592
##  8 8       9.82   -1.95  -0.425  1.61   -2.42   0.378  0.916   0.493   1.70
```

```
##  9 9       2.01   -1.53   0.749 -1.03   -2.69   0.770  1.15   -2.26   -1.16
## 10 10     -5.73   -1.61   1.76  -1.57   -0.693  0.156 -1.06    0.247   0.839
## # i 134 more rows
## # i 62 more variables: PC10 <dbl>, PC11 <dbl>, PC12 <dbl>, PC13 <dbl>,
## #   PC14 <dbl>, PC15 <dbl>, PC16 <dbl>, PC17 <dbl>, PC18 <dbl>, PC19 <dbl>,
## #   PC20 <dbl>, PC21 <dbl>, PC22 <dbl>, PC23 <dbl>, PC24 <dbl>, PC25 <dbl>,
## #   PC26 <dbl>, PC27 <dbl>, PC28 <dbl>, PC29 <dbl>, PC30 <dbl>, PC31 <dbl>,
## #   PC32 <dbl>, PC33 <dbl>, PC34 <dbl>, PC35 <dbl>, PC36 <dbl>, PC37 <dbl>,
## #   PC38 <dbl>, PC39 <dbl>, PC40 <dbl>, PC41 <dbl>, PC42 <dbl>, PC43 <dbl>, ...
```

```
pc_loadings <- pca.3$rotation %>%
  as_tibble(rownames = "gene")

# print the result
pc_loadings
```

```
## # A tibble: 71 x 72
##    gene           PC1     PC2      PC3     PC4     PC5     PC6      PC7      PC8
##    <chr>        <dbl>   <dbl>    <dbl>   <dbl>   <dbl>   <dbl>    <dbl>    <dbl>
##  1 Age        0.00747  0.0717 -0.0758   0.0525 -0.0163  0.0146 -0.0987   3.61e-1
##  2 TSPYL5     0.0669  -0.0891  0.0441  -0.0418  0.0692 -0.147  -0.171    1.48e-1
##  3 Contig63~  0.0592  -0.0897  0.0122  -0.129   0.165   0.144   0.223   -8.31e-2
##  4 DIAPH3     0.185    0.111   0.0142  -0.0570 -0.114   0.142  -0.0793   4.73e-2
##  5 NUSAP1     0.178    0.170  -0.00342 -0.0935 -0.0706  0.134  -0.0548   4.35e-2
##  6 AA555029~  0.0671  -0.183  -0.0585   0.0876  0.222   0.0483 -0.114    6.50e-4
##  7 ALDH4A1    0.00636  0.0748  0.205   -0.297   0.104  -0.0984 -0.0374   4.29e-2
##  8 QSCN6L1    0.148   -0.142   0.188   -0.0547 -0.0483 -0.0849  0.0431  -2.57e-2
##  9 FGF18     -0.135   -0.140  -0.0413  -0.104   0.0660  0.158  -0.105    1.04e-1
## 10 DIAPH3.1   0.176    0.0848  0.0978   0.122  -0.180   0.186   0.00610 -1.40e-2
## # i 61 more rows
## # i 63 more variables: PC9 <dbl>, PC10 <dbl>, PC11 <dbl>, PC12 <dbl>,
## #   PC13 <dbl>, PC14 <dbl>, PC15 <dbl>, PC16 <dbl>, PC17 <dbl>, PC18 <dbl>,
## #   PC19 <dbl>, PC20 <dbl>, PC21 <dbl>, PC22 <dbl>, PC23 <dbl>, PC24 <dbl>,
## #   PC25 <dbl>, PC26 <dbl>, PC27 <dbl>, PC28 <dbl>, PC29 <dbl>, PC30 <dbl>,
## #   PC31 <dbl>, PC32 <dbl>, PC33 <dbl>, PC34 <dbl>, PC35 <dbl>, PC36 <dbl>,
## #   PC37 <dbl>, PC38 <dbl>, PC39 <dbl>, PC40 <dbl>, PC41 <dbl>, PC42 <dbl>, ...
```

```
top_genes <- pc_loadings %>%
  # select only the PCs we are interested in
  select(gene, PC1, PC2) %>%
  # convert to a "long" format
  pivot_longer(matches("PC"),
               names_to = "PC",
               values_to = "loading") %>%
  # for each PC
  group_by(PC) %>%
  # arrange by descending order of loading
  arrange(desc(abs(loading))) %>%
  # take the 10 top rows
  slice(1:15) %>%
  # pull the gene column as a vector
  pull(gene) %>%
  # ensure only unique genes are retained
  unique()
```

```
top_genes
```

```
##  [1] "CENPA"       "MELK"        "ORC6L"       "PRC1"        "MCM6"
##  [6] "DIAPH3.2"    "DIAPH3"      "NM_004702"   "NUSAP1"      "C16orf61"
## [11] "DIAPH3.1"    "RFC4"        "CDCA7"       "DTL"         "GMPS"
## [16] "PECI"        "PECI.1"      "SLC2A3"      "COL4A2"      "PALM2.AKAP2"
## [21] "EXT1"        "C9orf30"     "AA555029_RC" "ECT2"        "MMP9"
## [26] "RTN4RL1"     "WISP1"       "RAB6B"
```

The amount of variability explained by the components can be computed bearing in mind that the square root of the eigenvalues is stored in vector sdev of the PCA object. The variance explained by the principal components can be visualised through a scree plot.

- Decide which components to keep and justify your decision.

Looking at the screeplot we can see that after the 7^{th} or 8^{th} variable the curve flattens and there does not seem to be much more gain to be had by adding more components. You may decide that it is important that for example 60% of the variation is explained, in that case you would check the cumulative proportion and keep 9 or 10 components. Or you could say that you will only keep components that explain at least 1 standard deviation of the data in which case you would keep 9 components. A good idea is to check all three.

- 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.

```
model_data <- nki %>%
  select(Event,Age, EstrogenReceptor,
         Grade, top_genes)
```

```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use `all_of()` or `any_of()` instead.
##   # Was:
##   data %>% select(top_genes)
##
##   # Now:
##   data %>% select(all_of(top_genes))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
##unadjusted logistic regression
mod_unadjusted <- glm(Event ~. ,
                      data = model_data[,-c(2,3,4)],
                      family = binomial())
summary(mod_unadjusted)
```

```
##
## Call:
## glm(formula = Event ~ ., family = binomial(), data = model_data[,
##     -c(2, 3, 4)])
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8091  -0.7738  -0.4443   0.8381   2.1450
##
```

```
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.1150     0.4908  -0.234    0.815
## CENPA         0.5522     1.7372   0.318    0.751
## MELK         -0.6547     2.2404  -0.292    0.770
## ORC6L         2.2635     1.6493   1.372    0.170
## PRC1          3.7203     2.2871   1.627    0.104
## MCM6         -3.1216     2.7879  -1.120    0.263
## DIAPH3.2      2.7344     4.7131   0.580    0.562
## DIAPH3       -0.3887     2.4929  -0.156    0.876
## NM_004702     2.1480     1.4919   1.440    0.150
## NUSAP1        2.4022     2.1092   1.139    0.255
## C16orf61     -3.5337     2.4181  -1.461    0.144
## DIAPH3.1     -5.7453     3.5064  -1.639    0.101
## RFC4          0.8772     2.4404   0.359    0.719
## CDCA7         0.1909     0.8922   0.214    0.831
## DTL          -0.5929     2.4763  -0.239    0.811
## GMPS          1.7594     2.0012   0.879    0.379
## PECI          2.3127     3.3651   0.687    0.492
## PECI.1       -3.2706     3.3143  -0.987    0.324
## SLC2A3       -0.4804     1.7111  -0.281    0.779
## COL4A2        2.8803     2.1984   1.310    0.190
## PALM2.AKAP2   1.1749     1.5655   0.750    0.453
## EXT1          2.2291     2.1171   1.053    0.292
## C9orf30      -1.0395     2.5699  -0.404    0.686
## AA555029_RC  -1.1806     1.6465  -0.717    0.473
## ECT2         -0.3485     2.0142  -0.173    0.863
## MMP9          0.2439     1.0091   0.242    0.809
## RTN4RL1      -1.9496     1.6389  -1.190    0.234
## WISP1         0.8820     1.6856   0.523    0.601
## RAB6B        -1.1648     1.0471  -1.112    0.266
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 183.32  on 143  degrees of freedom
## Residual deviance: 139.06  on 115  degrees of freedom
## AIC: 197.06
##
## Number of Fisher Scoring iterations: 5
```

```
##adjusted logistic regression
mod_adjusted <- glm(Event ~. ,
                    data = model_data,
                    family = binomial())
summary(mod_adjusted)
```

```
##
## Call:
## glm(formula = Event ~ ., family = binomial(), data = model_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8152  -0.6871  -0.4196   0.7446   2.4486
##
## Coefficients:
```

```
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)                4.38425    2.02196   2.168   0.0301 *
## Age                       -0.09690    0.04556  -2.127   0.0334 *
## EstrogenReceptorPositive -0.05985    0.95851  -0.062   0.9502
## GradePoorly diff          -0.32508    0.60964  -0.533   0.5939
## GradeWell diff            -0.29916    0.65095  -0.460   0.6458
## CENPA                      0.57778    1.80623   0.320   0.7491
## MELK                      -1.32774    2.33944  -0.568   0.5703
## ORC6L                      2.35997    1.74835   1.350   0.1771
## PRC1                       3.40269    2.33148   1.459   0.1444
## MCM6                      -2.94411    3.00926  -0.978   0.3279
## DIAPH3.2                   3.46929    5.13316   0.676   0.4991
## DIAPH3                    -0.60352    2.67070  -0.226   0.8212
## NM_004702                  2.75833    1.61653   1.706   0.0879 .
## NUSAP1                     2.84769    2.22707   1.279   0.2010
## C16orf61                  -3.90031    2.49394  -1.564   0.1178
## DIAPH3.1                  -6.45687    3.65782  -1.765   0.0775 .
## RFC4                       0.79269    2.61542   0.303   0.7618
## CDCA7                      0.02911    0.94049   0.031   0.9753
## DTL                       -0.33850    2.64044  -0.128   0.8980
## GMPS                       2.38689    2.16902   1.100   0.2711
## PECI                       2.21444    3.50095   0.633   0.5270
## PECI.1                    -3.92055    3.50541  -1.118   0.2634
## SLC2A3                    -0.59292    1.81946  -0.326   0.7445
## COL4A2                     2.70868    2.33537   1.160   0.2461
## PALM2.AKAP2                2.03196    1.70782   1.190   0.2341
## EXT1                       1.65535    2.24911   0.736   0.4617
## C9orf30                   -0.78909    2.82331  -0.279   0.7799
## AA555029_RC               -1.34908    1.74238  -0.774   0.4388
## ECT2                      -0.29740    2.16590  -0.137   0.8908
## MMP9                       0.24252    1.05361   0.230   0.8180
## RTN4RL1                   -1.59204    1.78196  -0.893   0.3716
## WISP1                      0.72296    1.84903   0.391   0.6958
## RAB6B                     -1.27753    1.12158  -1.139   0.2547
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 183.32  on 143  degrees of freedom
## Residual deviance: 133.41  on 111  degrees of freedom
## AIC: 199.41
##
## Number of Fisher Scoring iterations: 5
```

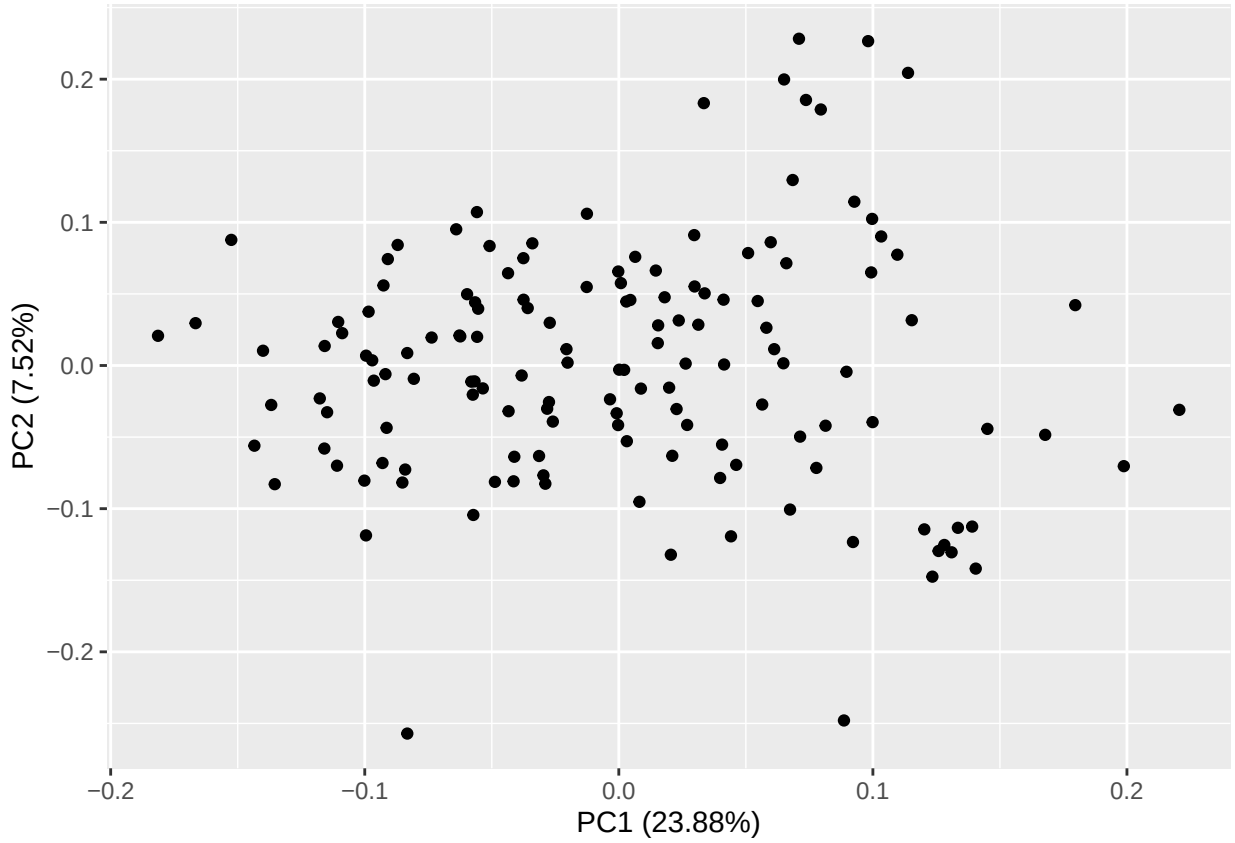
Problem 3.c (8 points)

- Use PCA plots to compare the main driverswith the correlation structure observed in **problem 3.a**.
- Examine how well the dataset may explain your outcome.
- Discuss your findings in full details and suggest any further steps if needed.

```
## Answer in this chunk
#install.packages("ggfortify")
library(ggfortify)
```

```
## Warning: package 'ggfortify' was built under R version 4.2.3
```

```
autoplot(pca.3)
```



1. The genes that rank in the top 10 or 15 in PC1 and PC2 (in the result 3b)are considered as the main drivers.
2. This refers to the correlation coefficients in point a, we can see the correlation between these main drivers(gene).
3. And the next step in my plan is to pick up all these main drivers and discuss for if the result of event will affect by those main driverrs with strong correlations.

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 with several experiments and evidences.

```
# Fit the model
model <- glm(Event ~., data = model_data,
             family = binomial) %>%
  stepAIC(trace = FALSE)
# Summarize the final selected model
summary(model)
```

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

```
## glm(formula = Event ~ Age + PRC1 + NM_004702 + NUSAP1 + C16orf61 +
##     DIAPH3.1 + PECI.1 + COL4A2, family = binomial, data = model_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6494  -0.7990  -0.4500   0.8523   2.2587
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  3.25476    1.71594   1.897   0.0579 .
## Age         -0.08394    0.03876  -2.166   0.0303 *
## PRC1         3.23606    1.73433   1.866   0.0621 .
## NM_004702    1.95958    1.20375   1.628   0.1035
## NUSAP1       2.82698    1.73530   1.629   0.1033
## C16orf61    -2.54592    1.78611  -1.425   0.1540
## DIAPH3.1    -3.98071    1.58022  -2.519   0.0118 *
## PECI.1      -2.47937    1.43746  -1.725   0.0846 .
## COL4A2       2.93963    1.70367   1.725   0.0844 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 183.32  on 143  degrees of freedom
## Residual deviance: 143.20  on 135  degrees of freedom
## AIC: 161.2
##
## Number of Fisher Scoring iterations: 5
```