Assignment 2

Yile Shi (s2168022)

Assignment 2

Biomedical Data Science

Due on Tuesday 5th April 2022, 5:00pm

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

Problem 1 (27 points)

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

Problem 1.a (7 points)

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

Solution:

```
# Read dataset into R
wdbc <- fread("assignment2/wdbc2.csv")
head(wdbc, 5)
```

```
##         id diagnosis radius texture perimeter   area smoothness compactness
## 1:   842302 malignant  17.99   10.38    122.80 1001.0    0.11840     0.27760
## 2:   842517 malignant  20.57   17.77    132.90 1326.0    0.08474     0.07864
```

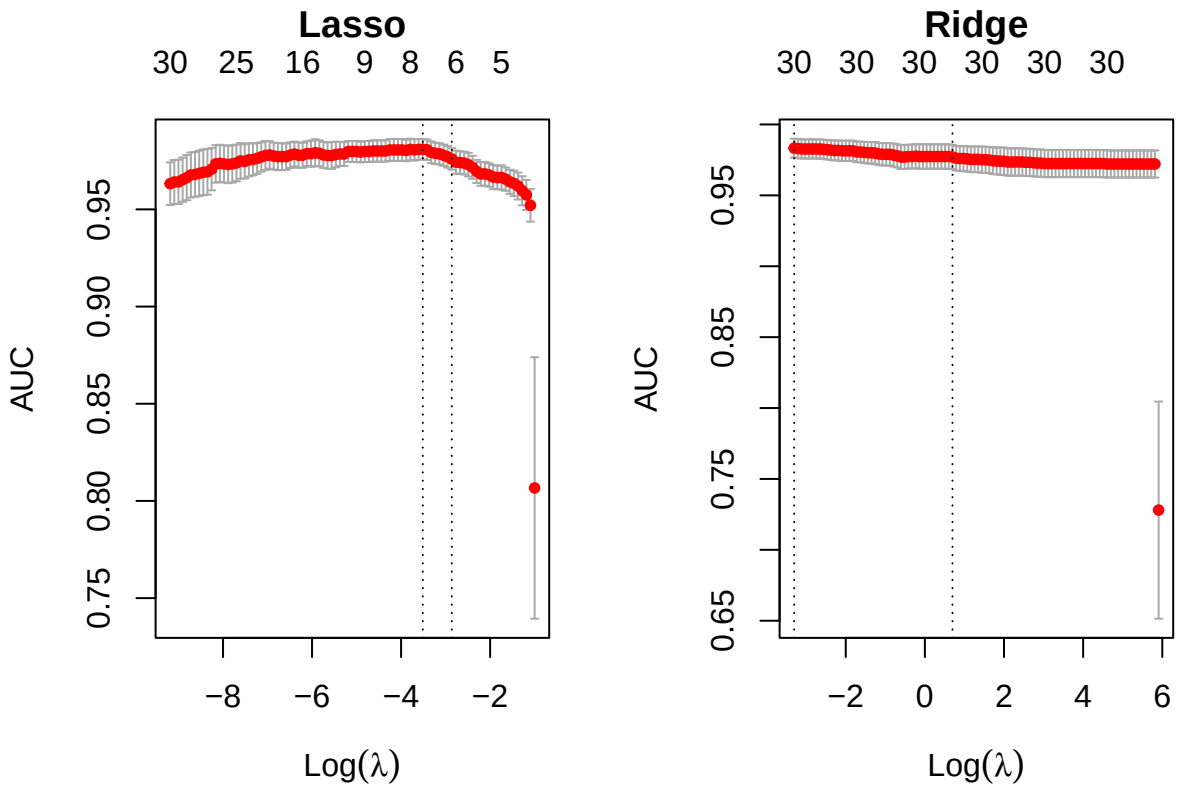
```
## 3: 84300903 malignant  19.69   21.25    130.00 1203.0    0.10960     0.15990
## 4: 84348301 malignant  11.42   20.38     77.58  386.1    0.14250     0.28390
## 5: 84358402 malignant  20.29   14.34    135.10 1297.0    0.10030     0.13280
##    concavity concavepoints symmetry fractaldimension radius.stderr
## 1:    0.3001       0.14710   0.2419          0.07871        1.0950
## 2:    0.0869       0.07017   0.1812          0.05667        0.5435
## 3:    0.1974       0.12790   0.2069          0.05999        0.7456
## 4:    0.2414       0.10520   0.2597          0.09744        0.4956
## 5:    0.1980       0.10430   0.1809          0.05883        0.7572
##    texture.stderr perimeter.stderr area.stderr smoothness.stderr
## 1:         0.9053            8.589      153.40          0.006399
## 2:         0.7339            3.398       74.08          0.005225
## 3:         0.7869            4.585       94.03          0.006150
## 4:         1.1560            3.445       27.23          0.009110
## 5:         0.7813            5.438       94.44          0.011490
##    compactness.stderr concavity.stderr concavepoints.stderr symmetry.stderr
## 1:            0.04904          0.05373              0.01587         0.03003
## 2:            0.01308          0.01860              0.01340         0.01389
## 3:            0.04006          0.03832              0.02058         0.02250
## 4:            0.07458          0.05661              0.01867         0.05963
## 5:            0.02461          0.05688              0.01885         0.01756
##    fractaldimension.stderr radius.worst texture.worst perimeter.worst
## 1:                0.006193        25.38         17.33          184.60
## 2:                0.003532        24.99         23.41          158.80
## 3:                0.004571        23.57         25.53          152.50
## 4:                0.009208        14.91         26.50           98.87
## 5:                0.005115        22.54         16.67          152.20
##    area.worst smoothness.worst compactness.worst concavity.worst
## 1:     2019.0           0.1622            0.6656          0.7119
## 2:     1956.0           0.1238            0.1866          0.2416
## 3:     1709.0           0.1444            0.4245          0.4504
## 4:      567.7           0.2098            0.8663          0.6869
## 5:     1575.0           0.1374            0.2050          0.4000
##    concavepoints.worst symmetry.worst fractaldimension.worst
## 1:              0.2654         0.4601                0.11890
## 2:              0.1860         0.2750                0.08902
## 3:              0.2430         0.3613                0.08758
## 4:              0.2575         0.6638                0.17300
## 5:              0.1625         0.2364                0.07678
```

```
# convert strings in 'diagnosis' to numeric
wdbc$diagnosis <- ifelse(wdbc$diagnosis == 'malignant', 1, 0)

# set random seed beforehand
set.seed(984065)

# create training and testing sets
train.idx <- createDataPartition(y = wdbc$diagnosis, p = 0.7)$Resample1
train.wdbc <- wdbc[train.idx, -c(1)]
test.wbdc <- wdbc[-train.idx, -c(1)]
```

```
# fit a Ridge regression and a Lasso regression using CV on training set
# first, we need matrices for x and y in the regression function
train.x <- train.wdbc[, -c(1)]
```

```r
train.x.mat <- as.matrix(train.x)
# convert strings in 'diagnosis' to numeric
train.y.mat <- train.wdbc$diagnosis

# now we fit regressions using CV
set.seed(984065)
fit.cv.lasso <- cv.glmnet(train.x.mat, train.y.mat,
                          family = "binomial", type.measure = "auc")
fit.cv.ridge <- cv.glmnet(train.x.mat, train.y.mat, alpha=0,
                          family = "binomial", type.measure = "auc")
```

```r
# make plots to find the best lambdas that maximise the AUC
par(mfrow = c(1,2), mar = c(4,4,5,2))
plot(fit.cv.lasso, main = "Lasso")
plot(fit.cv.ridge, main = "Ridge")
```




```r
# the values of best lambdas
cat("The best lambda in Lasso regression: ", fit.cv.lasso$lambda.min, "\n")
```

```
## The best lambda in Lasso regression:  0.02982835
```

```r
cat("The best lambda in Ridge regression: ", fit.cv.ridge$lambda.min)
```

```
## The best lambda in Ridge regression:  0.03677378
```

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'.

Solution:

```
# now we create the required data table
rep.dt <- data.table(model = c("Lasso(lambda.min)", "Lasso(lambda.1se)",
                               "Ridge(lambda.min)", "Ridge(lambda.1se)"),
                     lambda = c(round(fit.cv.lasso$lambda.min, 3),
                                round(fit.cv.lasso$lambda.1se, 3),
                                round(fit.cv.ridge$lambda.min, 3),
                                round(fit.cv.ridge$lambda.1se, 3)),
                     AUC = c(round(fit.cv.lasso$cvm[fit.cv.lasso$index[1]], 3),
                             round(fit.cv.lasso$cvm[fit.cv.lasso$index[2]], 3),
                             round(fit.cv.ridge$cvm[fit.cv.ridge$index[1]], 3),
                             round(fit.cv.ridge$cvm[fit.cv.ridge$index[2]], 3)),
                     modelsize = c(round(fit.cv.lasso$nzero[fit.cv.lasso$index[1]], 3),
                                   round(fit.cv.lasso$nzero[fit.cv.lasso$index[2]], 3),
                                   round(fit.cv.ridge$nzero[fit.cv.ridge$index[1]], 3),
                                   round(fit.cv.ridge$nzero[fit.cv.ridge$index[2]], 3)))
rep.dt
```

```
##                model lambda   AUC modelsize
## 1: Lasso(lambda.min)  0.030 0.981         8
## 2: Lasso(lambda.1se)  0.057 0.976         6
## 3: Ridge(lambda.min)  0.037 0.983        30
## 4: Ridge(lambda.1se)  2.009 0.977        30
```

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.

Solution:

```
# define the full model and null model
full.model <- glm(diagnosis ~ ., data = train.wdbc, family = "binomial")
null.model <- glm(diagnosis ~ 1, data = train.wdbc, family = "binomial")
```

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

# forward stepwise selection
```

```
model.S <- stepAIC(null.model, scope = list(upper = full.model),
                   direction = "forward", trace = FALSE)
```

```
# selected variables with standardized regression coefficients in model.B
# get the standardized coefficients
std.coef.B <- lm.beta(model.B)$standardized.coefficients

# order the coefficients in decreasing order of their absolute values
std.coef.B <- std.coef.B[order(abs(std.coef.B), decreasing = TRUE)]

cat("selected variables in model.B: \n")
```

```
## selected variables in model.B:
```

```
round(std.coef.B, 3)
```

```
##        radius.worst          area.worst           perimeter     concavity.worst
##              53.047             -40.610             -18.501               8.473
##       concavepoints              radius       radius.stderr       texture.worst
##               8.399               7.378               7.372               5.605
##   compactness.worst concavepoints.worst      texture.stderr    smoothness.worst
##              -5.321              -3.832              -3.723               2.008
##         (Intercept)
##               0.000
```

```
# selected variables with standardized regression coefficients in model.S
# get the standardized coefficients
std.coef.S <- lm.beta(model.S)$standardized.coefficients

# order the coefficients in decreasing order of their absolute values
std.coef.S <- std.coef.S[order(abs(std.coef.S), decreasing = TRUE)]

cat("selected variables in model.S: \n")
```

```
## selected variables in model.S:
```

```
std.coef.S
```

```
##        area.worst      radius.worst   perimeter.worst         perimeter
##        -44.347062         39.117707         16.396414        -13.329464
##     radius.stderr compactness.worst            radius         concavity
##         10.235291         -5.930017          5.544839          5.364296
##     texture.worst  perimeter.stderr   concavity.worst    texture.stderr
##          4.932050         -4.761290          4.299930         -3.028769
##  smoothness.worst       area.stderr       (Intercept)
##          2.661371          2.278462          0.000000
```

Problem 1.d (3 points)

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

Solution:

Here we consider using AIC as an appropriate criterion to compare the goodness-of-fit to the training data of two models.

```
data.frame(model.B = round(model.B$aic, 3),
           model.S = round(model.S$aic, 3),
           row.names = "AIC")
```

```
##     model.B model.S
## AIC  99.471 105.295
```

```
# model.B
signif(pchisq(model.B$null.deviance - model.B$deviance,
              df = model.B$df.null - model.B$df.residual,
              lower.tail = FALSE), 3)
```

```
## [1] 1.46e-89
```

```
# model.S
signif(pchisq(model.S$null.deviance - model.S$deviance,
              df = model.S$df.null - model.S$df.residual,
              lower.tail = FALSE), 3)
```

```
## [1] 1.35e-87
```

Problem 1.e (2 points)

Compute the training AUC for model B and model S.

Solution:

```
# predict values on training set
pred.B <- predict(model.B, newdata = train.wdbc, type = "response")
pred.S <- predict(model.S, newdata = train.wdbc, type = "response")
```

```
# auc
auc.B <- roc(train.y.mat ~ pred.B)$auc
```

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

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

```
auc.S <- roc(train.y.mat ~ pred.S)$auc
```

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

```
cat("AUC of Model.B: ", round(auc.B, 3), "\n")
```

```
## AUC of Model.B:  0.994
```

```
cat("AUC of Model.S: ", round(auc.S, 3))
```

```
## AUC of Model.S:  0.993
```

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.

Solution:

```
# predict values of testing set
pred.Lasso.test <- predict(fit.cv.lasso, newx = as.matrix(test.wbdc[, -c(1)]),
                      s = fit.cv.lasso$lambda.1se, type = "response")
pred.Ridge.test <- predict(fit.cv.ridge, newx = as.matrix(test.wbdc[, -c(1)]),
                      s = fit.cv.ridge$lambda.1se, type = "response")
pred.B.test <- predict(model.B, newdata = test.wbdc[, -c(1)], type = "response")
pred.S.test <- predict(model.S, newdata = test.wbdc[, -c(1)], type = "response")

# plot ROC curves of four models
roc.Lasso <- roc(test.wbdc$diagnosis, pred.Lasso.test,
                 plot = TRUE, col = "blue", xlim = c(0, 1),
                 main = "ROC curves (on testing data)")
```

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

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

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

```
roc.Ridge <- roc(test.wbdc$diagnosis, pred.Ridge.test,
                 plot = TRUE, col = "red", add = TRUE, xlim = c(0, 1))
```

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

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

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

```
roc.B <- roc(test.wbdc$diagnosis, pred.B.test,
                plot = TRUE, col = "green", add = TRUE, xlim = c(0, 1))
```

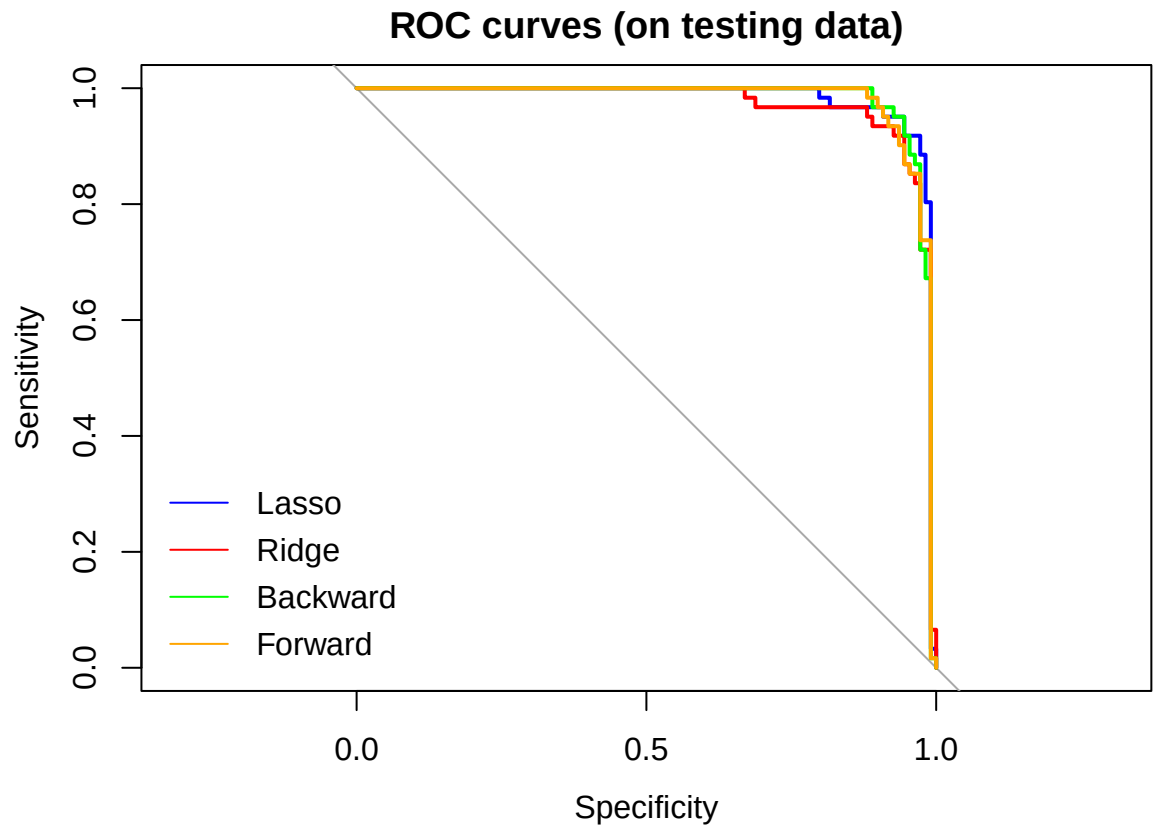
```
## Setting levels: control = 0, case = 1
```

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

```
roc.S <- roc(test.wbdc$diagnosis, pred.S.test,
                plot = TRUE, col = "orange",add = TRUE,  xlim = c(0, 1))
```

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

```
# add legend
legend("bottomleft", legend = c("Lasso", "Ridge", "Backward", "Forward"),
       col = c("blue", "red", "green", "orange"),
       lty = c(1, 1, 1, 1), bty = "n")
```




```
# data frame to report the AUC of each model on both training and testing sets
data.frame(train.AUC = c(round(fit.cv.lasso$cvm[fit.cv.lasso$index[2]], 3),
                         round(fit.cv.ridge$cvm[fit.cv.ridge$index[2]], 3),
                         round(auc.B, 3), round(auc.S, 3)),
           test.AUC = c(round(roc.Lasso$auc, 3), round(roc.Ridge$auc, 3),
                        round(roc.B$auc, 3), round(roc.S$auc, 3)),
           row.names = c("Lasso", "Ridge", "Backward", "Forward"))
```

```
##           train.AUC test.AUC
## Lasso         0.976    0.981
## Ridge         0.977    0.971
## Backward      0.994    0.980
## Forward       0.993    0.979
```

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.

Solution:

```
# read data into R
gdm.dt <- data.table(read.table("assignment2/GDM.raw.txt", header = TRUE))
head(gdm.dt, 5)
```

```
##    ID   sex pheno rs7513574_T rs1627238_A rs1171278_C rs1137100_A rs2568958_A
## 1:  1 FALSE     0           1           0           0           2           0
## 2:  2 FALSE     0           0           0           0           1           0
## 3:  4 FALSE     1           2           1           1           1           1
## 4:  5 FALSE     1           0           1           1           1           0
## 5:  6 FALSE     1           0           1           1           1           1
##    rs1514175_A rs1555543_C rs10923931_C rs516636_A rs574367_G rs543874_C
## 1:           1           2            0          0          0          0
## 2:           0           1            0          1          0          1
## 3:           1           2            0          0          0          0
## 4:           2           2            1          0          0          0
## 5:           0           0            0          0          0          0
##    rs7554506_A rs340874_G rs2867125_A rs6548238_A rs7561317_C rs6545814_T
## 1:           0          0           0          NA          0          0
## 2:           0          1           0           0          0          0
## 3:           0          1           0           0          0          0
## 4:           0          1           0           0          0          1
## 5:           0          0           0           0          0          1
##    rs713586_C rs11899863_C rs7578597_C rs887912_C rs243021_C rs2890652_T
## 1:          0            0           0          0          1          0
## 2:          0            0           0          0          1          0
## 3:          0            0           0          1          1          0
## 4:          1            1           0          0          0          0
## 5:          1            1           1          0          0          0
##    rs2925757_C rs3923113_C rs13389219_T rs7578326_A rs2943641_A rs1801282_C
## 1:           0           0            1           0           1           0
```

```
## 2:           0           0           0           1           0           0
## 3:           0           1           1           1           0           0
## 4:           0           0           0           0           0           1
## 5:           0           0           0           0           0           0
##    rs6780569_C rs831571_T rs4607103_G rs13078807_T rs11708067_G rs187230_A
## 1:           0          0           1            0           1          1
## 2:           0          0           1            0           0          1
## 3:           0          0           0            0           1          0
## 4:           0          0           0            0           2          1
## 5:           1          0           1            0           0          1
##    rs4402960_T rs1470579_C rs7647305_G rs9816226_C rs266729_G rs1501299_C
## 1:           0           0           0           0          1           1
## 2:           0           0           1           0          0           2
## 3:           2           2           1           0          2           1
## 4:           0           0           0           0          0           2
## 5:           0           0           0           0          0           2
##    rs16861329_C rs6815464_A rs4688985_A rs1801214_A rs10938397_T rs2227306_G
## 1:            0           0           1           2            0           1
## 2:            1           0           0           0            0           1
## 3:            1           1           0           0            0           1
## 4:            1           0           0           0            0           0
## 5:            1           0           0           0            0           0
##    rs2886920_G rs13107325_T rs459193_G rs2112347_A rs4457053_C rs261967_G
## 1:           1            0          0           0           2          2
## 2:           1            0          0           1           0          0
## 3:           1            0          1           1           1          0
## 4:           0            0          0           0           1          1
## 5:           0            0          1           0           1          1
##    rs4836133_A rs7754840_G rs7756992_A rs9356744_C rs2206734_T rs1052248_G
## 1:           0           0           0           0           0           1
## 2:           1           0           0           0           0           1
## 3:           0           0           0           0           0           0
## 4:           1           1           1           1           0           1
## 5:           2           0           0           0           0           1
##    rs11575839_C rs206936_G rs9470794_A rs1535500_T rs987237_C rs9395950_T
## 1:            0          1           0           1          1           0
## 2:            0          1           0           2          2           0
## 3:            0          0           0           2          0           0
## 4:            0          1           0           1          1           0
## 5:            1          1           0           1          0           1
##    rs17168486_T rs2191349_T rs6954897_G rs864745_A rs1635852_C rs849134_G
## 1:           2           1           1          0           0         NA
## 2:           0           1           0          1           1          1
## 3:           1           1           0          1           1          1
## 4:           2           1           0          0           0          0
## 5:           1           2           0          0           0          0
##    rs4607517_A rs6467136_T rs2167270_G rs972283_A rs516946_C rs896854_C
## 1:           1           0           2          1          0          0
## 2:           2           1           1          1          0          1
## 3:           0           1           2          2          1         NA
## 4:           0           2           1          0          0          2
## 5:           0           0           2          1          0          1
##    rs13266634_G rs3802177_A rs7041847_G rs17584499_T rs2383208_A rs10965250_T
## 1:            1           1           1            1           0            0
```

```
## 2:            1           1           1            0           0            0
## 3:            1           1           1            1           0            0
## 4:            1           1           0            2           0            0
## 5:            1           1           2            1           0            0
##    rs10811661_A rs2183825_T rs824248_G rs11142387_A rs13292136_A rs2796441_T
## 1:           NA           1          1            2           0            1
## 2:            0           0          0            1           0            0
## 3:            0           0          1            1           0            2
## 4:            0           1          0            0           0            0
## 5:            0           1          0            1           1            0
##    rs12779790_T rs10882066_C rs1111875_A rs5015480_G rs7087591_T rs7901695_T
## 1:            0            0           1           1           1           0
## 2:            1            1           1           0           0           0
## 3:            1            1           2           0           0           0
## 4:            1            0           2           0           0           0
## 5:            1            2           0           0           0           0
##    rs4506565_T rs7903146_C rs12243326_A rs2334499_T rs10770141_A rs231362_T
## 1:           0           0            1           1            1          1
## 2:           0           0            0           0            1          0
## 3:           0           0            0           1            1          1
## 4:           0           0            0           1            0          1
## 5:           0           0            0           1            1          1
##    rs2237892_C rs163184_T rs2237897_T rs4929949_C rs5215_C rs2056246_A
## 1:           2          2           2           1        1           2
## 2:           2          2           2           1        1           1
## 3:           0          0           0           1        0           1
## 4:           0          1           1           2        1           0
## 5:           0          1           0           1        2           2
##    rs10488683_A rs685249_T rs508924_C rs4923461_T rs6265_G rs10767664_C
## 1:            0          2          2           1        1            1
## 2:            1          1          1           0        0            0
## 3:            1          1          1           0        0            0
## 4:            1          0          0           0        0            0
## 5:            0          2          2           0        0            0
##    rs2030323_C rs3817334_T rs10838738_G rs1552224_T rs1387153_A rs10830962_T
## 1:           1           0            0           0           1            1
## 2:           0           0            0           0           0            0
## 3:           0           0            0           0           1            1
## 4:           0           1            1           0           1            1
## 5:           0           0            0           0           0            0
##    rs10830963_A rs2041139_T rs73040004_C rs10842994_G rs7138803_C rs1531343_T
## 1:            0           0            0           0           1           0
## 2:            0           0            1           0           1           0
## 3:            1           0            0           0           0           0
## 4:            1           0            0           0           0           0
## 5:            0           1            1           0           0           0
##    rs7961581_C rs7957197_A rs4771122_G rs1359790_A rs11847697_A rs10150332_A
## 1:           1           0           1           0            0           NA
## 2:           1           0           0           1            0           0
## 3:           0           0           1           2            0           0
## 4:           0           1           0           2            0           0
## 5:           0           0           0           1            0           0
##    rs1884082_G rs7172432_G rs2241423_G rs12898654_T rs7178572_G rs7177055_A
## 1:           0           1           2            1           0           0
```

```
## 2:           0           2           0           0           1           1
## 3:           1           1           0           0           0           0
## 4:           0           1           0           0           2           2
## 5:           0           1           0           0           0           0
##    rs11634397_A rs2028299_C rs8042680_A rs7359397_G rs1421085_T rs1558902_C
## 1:            1           0           1           0           0           0
## 2:            1           0           0           1           1           1
## 3:            1           0           1           0           0           0
## 4:            2           0           0           1           1           1
## 5:            0           0           0           1           0           0
##    rs1121980_G rs17817449_T rs8050136_A rs9939609_A rs9941349_A rs12149832_A
## 1:           0           0           0           0           0           0
## 2:           1           1           1           1           1           1
## 3:           0           0           0           0           0           0
## 4:           1           1           1           1           1           1
## 5:           0           0           0           0           0           0
##    rs11642841_G rs6499500_C rs7202877_T rs4523957_G rs391300_C rs75493593_C
## 1:            0           0           0           1          1           0
## 2:            1           0           2           1          1           1
## 3:            0           1           0           0          0           0
## 4:            1           0           0           1          1           0
## 5:            0           2           1           2          2           0
##    rs75418188_T rs13342232_A rs13342692_C rs117767867_T rs757210_T rs4430796_T
## 1:            0            0            0             0         1           1
## 2:            1            1            1             1         1           1
## 3:            0            0            0             0         1           1
## 4:            0            0            0             0         0           0
## 5:            0            0            0             0         1           1
##    rs7501939_C rs2331841_C rs6567160_G rs571312_G rs17782313_T rs12970134_C
## 1:           1           0           0          0           0           1
## 2:           1           0           0          0           0           0
## 3:           1           0           0          0           0           0
## 4:           0           1           0          0           0           0
## 5:           1           1           0          0           0           0
##    rs1423096_T rs3786897_A rs29941_T rs8108269_T rs2287019_A rs3810291_T
## 1:           0           0         1           0           0           0
## 2:           0           0         1           0           0           2
## 3:           0           0         1           1           0           1
## 4:           0           0         2           2           0           1
## 5:           0           0         0           0           0           0
##    rs6017317_G rs1800961_G rs5945326_C
## 1:           1           0           1
## 2:           0           0           0
## 3:           0           0           1
## 4:           2           0           2
## 5:           2           0           0
```

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.

```
# Enter code here.
```

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.

```
# Enter code here.
```

Problem 2.d (4points)

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

```
# Enter code here.
```

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.

```
# Enter code here.
```

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).

```
# Enter code here.
```

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.

```
# Enter code here.
```

Problem 2.h (4points)

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

```
# Enter code here.
```

Problem 3 (33 points)

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

Problem 3.a (6 points)

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

```
# Enter code here.
```

Problem 3.b (8 points)

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

```
# Enter code here.
```

Problem 3.c (8 points)

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

```
# Enter code here.
```

Problem 3.d (11 points)

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

```
# Enter code here.
```