Multinomial Lasso Regression with Lasso Model traning/testing, confusion matrix

Minjun Park

5/6/2020

Install missing packages

```
list.of.packages <- c("glmnet", "qpcR", "caret", "DMwR", "e1071")
new.packages <- list.of.packages[!(list.of.packages %in% installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages)
```

```
m <- "./D2K_BCM_DATASET/4kensembled.tsv"
dat <- read.csv(m, sep='\t', header = TRUE)
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 3.0-2
```

```
library(qpcR)
```

```
## Loading required package: MASS
```

```
## Loading required package: minpack.lm
```

```
## Loading required package: rgl
```

```
## Warning: package 'rgl' was built under R version 3.6.2
```

```
## Loading required package: robustbase
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
## Warning: package 'lattice' was built under R version 3.6.2
```

```
## Loading required package: ggplot2
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:qpcR':
##
##     RMSE
```

```
library(DMwR)
```

```
## Loading required package: grid
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
```

```
##   as.zoo.data.frame zoo
```

```
library(e1071)
```

```
#See how many times they overlap
amddata <- dat[2:4440]

amddata <- as.matrix(amddata)

## set seed
set.seed(1)

amddata <- as.data.frame(amddata)
table(amddata[1])
```

```
##
##   1   2   3   4
##  97 160  98  52
```

```
amddata$mgs_level <- factor(amddata$mgs_level)
balanced_data <-SMOTE(mgs_level ~ ., amddata, perc.over = 350, perc.under = 400)
#table(balanced_data[1])
#balanced_data

X_amddata <- as.matrix(balanced_data[,2:4439])
#X_amddata <- as.matrix(amddata[,2:4439])
y <- balanced_data[,1]
#y <- amddata[,1]
y <- factor(y)

kfolds <- 10
amddata <- amddata[sample(nrow(amddata)),] #this shuffles the training data
folds <- cut(seq(1,nrow(amddata)),breaks=kfolds,labels=FALSE) #this creates k folds on training data
folds
```

```
##   [1]  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
##  [26]  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  2  2  2  2  2  2  2  2  2
##  [51]  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2
##  [76]  2  2  2  2  2  2  2  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3
## [101]  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  4  4  4
## [126]  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4
## [151]  4  4  4  4  4  4  4  4  4  4  4  4  4  4  5  5  5  5  5  5  5  5  5  5  5
## [176]  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5
## [201]  5  5  5  5  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6
## [226]  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6  7  7  7  7  7  7
## [251]  7  7  7  7  7  7  7  7  7  7  7  7  7  7  7  7  7  7  7  7  7  7  7  7  7
## [276]  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8
## [301]  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8
## [326]  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9
## [351]  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9 10 10 10 10 10 10 10 10 10
## [376] 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
## [401] 10 10 10 10 10 10 10
```

```
# get lambda min from cv function
cv.lasso <- cv.glmnet(X_amddata, y, family = "multinomial", alpha = 1, nlambda = 100, nfolds = 10)
cv.lasso$lambda.min
```

```
## [1] 0.001821968
```

```
#balanced_data
#table(balanced_data[1])

tot <- c()
s1_var_names <- c()
s2_var_names <- c()
s3_var_names <- c()
s4_var_names <- c()
true_num <- c()
false_num <- c()
true_val <- c()
pred_val <- c()


for(i in 1:kfolds){
  set.seed(i)
  t_ind <- which(folds==i,arr.ind=TRUE) #this segments the data by fold
  ivalid <- amddata[t_ind, ] #this selects fold i for cv test
  itrain <- amddata[-t_ind, ] #this selects remaining k-1 folds for cv train

  # use smote to balance data
  #itrain <- as.data.frame(itrain)
  #itrain$mgs_level <- factor(itrain$mgs_level)
  #itrain <- SMOTE(mgs_level ~ ., itrain)

  end <- length(itrain)

  X <- itrain[,2:end]
  X <- as.matrix(X)
  y <- itrain[,1]
  y <- factor(y)

  # fit glmnet code using itrain
  fitlasso <- glmnet(X, y, family="multinomial", alpha=1, lambda = cv.lasso$lambda.min, type.multinomial

  # record vars with non-zero coeff
  # get column names to extract nonzero coefficients
  coln <- colnames(X)

  coeff_glm <- coef(fitlasso)
  coeff_s1 <- coeff_glm[1]
  coeff_s2 <- coeff_glm[2]
  coeff_s3 <- coeff_glm[3]
  coeff_s4 <- coeff_glm[4]

  # get the indeces
  vec_s1 <- coeff_s1[["1"]]@i
  vec_s2 <- coeff_s2[["2"]]@i
  vec_s3 <- coeff_s3[["3"]]@i
  vec_s4 <- coeff_s4[["4"]]@i

  # get the name of genes
```

```
  vars_s1 <- coln[vec_s1]
  s1_var_names <- append(s1_var_names, vars_s1)
  vars_s2 <- coln[vec_s2]
  s2_var_names <- append(s2_var_names, vars_s2)
  vars_s3 <- coln[vec_s3]
  s3_var_names <- append(s3_var_names, vars_s3)
  vars_s4 <- coln[vec_s4]
  s4_var_names <- append(s4_var_names, vars_s4)

  # predict using ivalid and save ivalid predictions
  true <- ivalid[,1]
  ivalid <- as.data.frame(ivalid)

  ivalid_x <- as.matrix(ivalid[,2:end])
  #sample <- ivalid[,2:end]
  pred <- predict(fitlasso, ivalid_x, s ="lambda.min", type ="class")

  # to get the F1 score
  true_val <- append(true_val, true)
  pred_val <- append(pred_val, pred)

  # count how many true predictions there are
  count_true = 0
  count_false = 0
  for (i in c(1:length(true))){
    if (true[i] == pred[i]){
      count_true = count_true + 1
    }
    else {count_false = count_false + 1}
  }
  true_num <- append(true_num, count_true)
  false_num <- append(false_num, count_false)


  # rbind to save non-zero coeff from all 10 folds
  comb <- qpcR:::cbind.na(true, pred)
  tot <- qpcR:::cbind.na(tot, comb)
}
```

```
true_val
```

```
##   [1] 3 1 2 3 2 2 2 4 2 1 4 3 2 3 3 1 2 4 2 1 3 2 2 3 1 3 3 2 3 4 1 2 2 4 1 2 4
##  [38] 3 3 1 2 1 3 4 1 2 4 2 1 2 1 3 4 2 2 2 1 4 2 2 1 2 2 3 3 1 1 3 1 2 1 3 3 2
##  [75] 1 1 4 2 1 2 4 4 1 1 2 4 2 1 2 4 2 3 3 1 3 2 2 2 2 3 4 3 2 4 3 3 3 1 2 3 2
## [112] 1 2 4 2 1 1 3 4 4 3 2 1 2 2 1 1 2 1 3 1 3 3 2 3 2 2 1 1 1 2 2 1 1 1 2 1 2
## [149] 2 2 1 2 2 2 2 2 1 1 2 3 2 3 1 2 2 2 2 3 1 1 2 3 2 1 1 1 1 1 3 2 1 1 2 2 2
## [186] 2 1 1 2 3 4 1 2 2 2 4 2 4 2 1 3 2 2 1 4 1 3 2 4 3 2 2 1 3 2 2 1 1 3 3 1 1
## [223] 1 2 3 2 2 1 2 2 2 3 1 4 4 1 3 4 2 1 1 1 4 3 2 4 1 3 3 4 3 4 2 3 3 2 2 3 2
## [260] 1 4 2 2 3 2 2 2 3 2 1 4 4 1 1 4 2 3 4 2 3 1 2 2 2 2 2 2 1 2 1 3 2 3 3 1 2
## [297] 3 2 2 2 3 4 4 3 1 2 2 2 2 2 1 1 3 2 1 1 2 2 3 4 2 3 3 4 1 3 1 4 3 2 1 4 3
## [334] 4 2 2 3 2 3 3 1 2 2 2 1 2 4 3 3 3 3 2 3 3 4 2 4 3 2 2 2 2 1 3 3 2 2 2 1 2
## [371] 3 4 2 3 3 1 2 2 3 1 3 2 2 2 3 2 2 4 2 3 2 2 1 3 3 3 2 4 3 3 2 1 3 4 4 2 2
```

```
pred_val <- as.numeric(pred_val)
pred_val
```

```
##   [1] 3 2 2 3 1 3 2 4 2 1 2 3 2 3 3 2 2 1 2 1 4 2 2 3 1 2 3 2 3 4 1 2 2 3 1 4 1
##  [38] 2 2 1 2 3 3 2 1 2 2 4 1 2 3 3 3 2 2 2 1 4 2 2 2 2 2 3 1 1 1 3 1 2 1 3 3 2
##  [75] 1 1 1 2 1 2 3 4 2 2 2 2 2 2 2 2 3 2 2 1 3 1 2 2 2 3 1 3 3 1 3 2 1 2 2 3 3
## [112] 2 2 3 2 1 2 3 2 1 2 2 2 2 2 2 2 2 1 4 2 2 3 2 2 1 4 2 1 3 2 2 1 2 2 3 2 2
## [149] 2 3 2 2 3 2 2 1 1 3 2 2 4 3 1 2 2 2 2 2 1 1 2 2 4 3 1 1 3 1 2 2 1 2 2 2 1
## [186] 2 1 2 1 3 2 1 2 2 2 1 2 2 3 1 3 1 2 1 1 2 3 2 2 3 3 2 2 4 3 2 3 3 3 3 2 1
## [223] 2 2 3 2 1 2 2 2 2 3 2 4 4 1 2 2 2 2 1 4 3 1 1 4 1 3 3 3 3 4 2 3 3 2 3 1 2
## [260] 2 4 2 3 1 2 4 2 3 2 1 3 4 2 1 4 2 2 1 2 2 1 3 3 2 2 2 2 3 3 3 3 2 4 3 1 2
## [297] 3 2 2 1 3 2 3 3 1 1 3 2 2 2 1 1 3 2 1 3 2 3 3 4 2 2 4 4 1 3 2 1 4 2 2 3 4
## [334] 3 2 2 3 2 3 2 1 2 2 2 3 1 2 3 3 2 1 2 3 3 2 2 4 1 3 2 2 3 2 3 3 2 2 3 1 2
## [371] 3 2 2 2 3 1 2 2 3 2 3 1 2 3 3 1 2 4 2 2 2 2 1 2 3 3 2 3 1 3 2 1 2 2 2 2 2
```

```
t_fac <- factor(true_val)
p_fac <- factor(pred_val)

confusionMatrix(p_fac, t_fac)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   2   3   4
##          1  51  14   8  10
##          2  33 119  23  16
##          3  12  21  60  11
##          4   1   6   7  15
##
## Overall Statistics
##
##                Accuracy : 0.602
##                  95% CI : (0.5526, 0.6499)
##     No Information Rate : 0.3931
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4283
##
##  Mcnemar's Test P-Value : 0.001574
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3 Class: 4
## Sensitivity            0.5258   0.7438   0.6122  0.28846
## Specificity            0.8968   0.7085   0.8576  0.96056
## Pos Pred Value         0.6145   0.6230   0.5769  0.51724
## Neg Pred Value         0.8580   0.8102   0.8746  0.90212
## Prevalence             0.2383   0.3931   0.2408  0.12776
## Detection Rate         0.1253   0.2924   0.1474  0.03686
## Detection Prevalence   0.2039   0.4693   0.2555  0.07125
## Balanced Accuracy      0.7113   0.7261   0.7349  0.62451
```

Apply model on test set

```
a <- "/Users/minjunp/Documents/rice/glmnet/4ktestdata.tsv"
testdat <- read.csv(a, sep='\t', header=TRUE)
testdat <- testdat[,3:4441]

end <- length(testdat)

X_test <- testdat[,2:end]
X_test <- as.matrix(X_test)
y <- testdat[,1]
y <- factor(y)

end <- length(amddata)
X_train <- amddata[,2:end]
X_train <- as.matrix(X_train)
y_train <- amddata[,1]
y_train <- factor(y_train)
fitlasso <- glmnet(X_train, y_train, family="multinomial", alpha=1, lambda = cv.lasso$lambda.min,
                   type.multinomial = "ungrouped")
pred <- predict(fitlasso, X_test, s ="lambda.min", type ="class")

count_true = 0
count_false = 0
for (i in c(1:length(true))){
  if (true[i] == pred[i]){
    count_true = count_true + 1
  }
  else {count_false = count_false + 1}
}
y
```

```
##  [1] 2 3 4 3 1 3 2 4 3 2 1 4 4 4 4 2 4 3 3 3 3 3 2 2 3 2 1 2 2 2 2 3 4 1 1 1 3 4
## [39] 3 3 2 2 1 1 2 2
## Levels: 1 2 3 4
```

```
paste(as.character(pred), collapse=", ")
```

```
## [1] "1, 1, 1, 3, 1, 2, 2, 1, 4, 1, 1, 2, 1, 1, 3, 1, 3, 2, 2, 2, 1, 3, 2, 2, 2, 3, 1, 1, 1, 2, 2, 1,
```

```
pred <- as.factor(pred)
```

Save genes that are selected by the model that appear at least 5 times

```
s1_var_name <- as.data.frame(table(s1_var_names))
s2_var_name <- as.data.frame(table(s2_var_names))
s3_var_name <- as.data.frame(table(s3_var_names))
s4_var_name <- as.data.frame(table(s4_var_names))

new_s1_table <- s1_var_name[which(s1_var_name["Freq"] > 5),]
new_s2_table <- s2_var_name[which(s2_var_name["Freq"] > 5),]
new_s3_table <- s3_var_name[which(s3_var_name["Freq"] > 5),]
new_s4_table <- s4_var_name[which(s4_var_name["Freq"] > 5),]
```

```
new_s1_table
```

```
##        s1_var_names Freq
## 17  ENSG00000112562   10
## 21  ENSG00000117090    9
## 25  ENSG00000125414    9
## 29  ENSG00000132205   10
## 31  ENSG00000134817    9
## 32  ENSG00000136167    6
## 36  ENSG00000140274    6
## 40  ENSG00000147570    9
## 41  ENSG00000151224    8
## 62  ENSG00000165972    8
## 64  ENSG00000166800    6
## 74  ENSG00000173838    9
## 76  ENSG00000177173    7
## 77  ENSG00000178440    9
## 80  ENSG00000182477    9
## 84  ENSG00000186204    6
## 86  ENSG00000188501    9
## 87  ENSG00000188782    6
## 88  ENSG00000188801    6
## 91  ENSG00000196796   10
## 95  ENSG00000201499    9
## 96  ENSG00000203914    6
## 101 ENSG00000205444    9
## 105 ENSG00000207002    9
## 108 ENSG00000213036    7
## 110 ENSG00000213538    9
## 115 ENSG00000215452    8
## 116 ENSG00000215480    8
## 119 ENSG00000220091    7
## 122 ENSG00000224426   10
## 123 ENSG00000224771    8
## 128 ENSG00000225611    6
## 131 ENSG00000226553    8
## 134 ENSG00000226801   10
## 139 ENSG00000227742    7
## 142 ENSG00000228037   10
## 144 ENSG00000228668    7
## 145 ENSG00000228961    8
## 146 ENSG00000229233    9
## 154 ENSG00000231888    9
## 155 ENSG00000231989    9
## 157 ENSG00000232184    6
## 159 ENSG00000232727    8
## 161 ENSG00000233039    7
## 162 ENSG00000233090    6
## 174 ENSG00000235957    8
## 181 ENSG00000236709    8
## 188 ENSG00000239455    8
## 191 ENSG00000239820   10
## 197 ENSG00000241131   10
## 198 ENSG00000241233    6
```

```
## 200 ENSG00000241420   10
## 201 ENSG00000242375   10
## 202 ENSG00000242477   10
## 203 ENSG00000242574   10
## 204 ENSG00000242707   10
## 205 ENSG00000243469    7
## 206 ENSG00000243845    8
## 207 ENSG00000244378    8
## 211 ENSG00000248817    7
## 212 ENSG00000249014    6
## 215 ENSG00000250942    6
## 216 ENSG00000251032    8
## 217 ENSG00000251155    9
## 221 ENSG00000251621    6
## 223 ENSG00000252892   10
## 224 ENSG00000253679    9
## 225 ENSG00000253817    6
## 226 ENSG00000254006    8
## 230 ENSG00000255240    7
## 231 ENSG00000255353    9
## 240 ENSG00000258535    9
## 247 ENSG00000260518    9
## 252 ENSG00000261114    8
## 259 ENSG00000263220   10
## 267 ENSG00000266923    6
## 268 ENSG00000267005   10
## 275 ENSG00000268555   10
## 279 ENSG00000271776    8
## 280 ENSG00000271793    7
## 283 ENSG00000272058    6
## 284 ENSG00000272081   10
## 285 ENSG00000272239   10
## 286 ENSG00000272256    7
## 288 ENSG00000272269   10
## 290 ENSG00000272372    7
## 292 ENSG00000272715   10
## 299 ENSG00000273703    8
## 300 ENSG00000273821    6
## 303 ENSG00000274475    6
## 304 ENSG00000275491   10
## 313 ENSG00000279450    8
## 316 ENSG00000279773    9
## 317 ENSG00000280189    7
## 320 ENSG00000281849    9
## 322 ENSG00000283294    6
```

```
new_s2_table
```

```
##        s2_var_names Freq
## 5   ENSG00000079459    8
## 7   ENSG00000096006    9
## 9   ENSG00000101190    6
## 12  ENSG00000104714    8
## 17  ENSG00000112761    8
## 18  ENSG00000115718    8
```

```
## 21  ENSG00000117009    9
## 22  ENSG00000117594    8
## 42  ENSG00000133392    6
## 47  ENSG00000139656   10
## 51  ENSG00000146722    6
## 58  ENSG00000163885    9
## 59  ENSG00000163898    6
## 64  ENSG00000166211    9
## 66  ENSG00000167139    8
## 67  ENSG00000167165   10
## 75  ENSG00000173124    7
## 77  ENSG00000173811    7
## 84  ENSG00000179673   10
## 88  ENSG00000182531    7
## 90  ENSG00000183022    6
## 96  ENSG00000185904    6
## 98  ENSG00000186458   10
## 101 ENSG00000188076    6
## 103 ENSG00000196274    9
## 117 ENSG00000206633    6
## 118 ENSG00000207002    7
## 125 ENSG00000213926    9
## 128 ENSG00000215472    6
## 135 ENSG00000223387   10
## 137 ENSG00000223511    9
## 139 ENSG00000223941    7
## 142 ENSG00000224771    8
## 144 ENSG00000225056    9
## 146 ENSG00000225344    8
## 149 ENSG00000225873    7
## 150 ENSG00000226161    6
## 154 ENSG00000226957   10
## 155 ENSG00000227077    8
## 163 ENSG00000228492    8
## 164 ENSG00000228560    9
## 167 ENSG00000228961    8
## 168 ENSG00000229657    6
## 169 ENSG00000229700    9
## 170 ENSG00000229851    7
## 171 ENSG00000230146    6
## 173 ENSG00000230398    8
## 193 ENSG00000234521    7
## 195 ENSG00000234743    7
## 196 ENSG00000234770   10
## 197 ENSG00000234919    6
## 198 ENSG00000235086    8
## 200 ENSG00000235248    9
## 201 ENSG00000235268    9
## 205 ENSG00000235649    7
## 207 ENSG00000235786    8
## 215 ENSG00000236839    7
## 217 ENSG00000237263    6
## 221 ENSG00000237828    9
## 222 ENSG00000237973    8
```

```
## 224 ENSG00000238151   10
## 228 ENSG00000239831    6
## 230 ENSG00000240401    9
## 233 ENSG00000241409    9
## 237 ENSG00000242696    7
## 247 ENSG00000246422    6
## 248 ENSG00000246448   10
## 253 ENSG00000249731    7
## 258 ENSG00000250896    7
## 265 ENSG00000253138    8
## 268 ENSG00000253919    6
## 269 ENSG00000254006    6
## 270 ENSG00000254044   10
## 276 ENSG00000254831   10
## 279 ENSG00000254998    7
## 280 ENSG00000255126    7
## 281 ENSG00000255184    7
## 284 ENSG00000255426    7
## 288 ENSG00000256007    9
## 289 ENSG00000256293   10
## 292 ENSG00000256812    6
## 294 ENSG00000257553    8
## 298 ENSG00000258308    9
## 299 ENSG00000258384   10
## 303 ENSG00000258809   10
## 304 ENSG00000258904    8
## 305 ENSG00000259091    9
## 306 ENSG00000259099    7
## 312 ENSG00000260034    7
## 316 ENSG00000260282    8
## 319 ENSG00000260851    6
## 323 ENSG00000260979    7
## 325 ENSG00000261212    7
## 327 ENSG00000261509    8
## 329 ENSG00000261692    8
## 331 ENSG00000261833    9
## 346 ENSG00000267565    8
## 347 ENSG00000267612   10
## 354 ENSG00000270547    8
## 356 ENSG00000271065    9
## 357 ENSG00000271156    6
## 364 ENSG00000272157    7
## 366 ENSG00000272249   10
## 368 ENSG00000272742    9
## 371 ENSG00000272865    9
## 373 ENSG00000273232   10
## 374 ENSG00000273267   10
## 381 ENSG00000273682   10
## 384 ENSG00000273777    6
## 386 ENSG00000274181    6
## 389 ENSG00000274606    6
## 393 ENSG00000275869    6
## 395 ENSG00000275994    7
## 396 ENSG00000276087    6
```

```
## 397 ENSG00000276093    8
## 398 ENSG00000276397   10
## 400 ENSG00000276653    6
## 401 ENSG00000277233   10
## 406 ENSG00000277745    7
## 411 ENSG00000279030    7
## 412 ENSG00000279187    9
## 419 ENSG00000280029    7
## 422 ENSG00000280436   10
## 426 ENSG00000281849    7
## 430 ENSG00000283578   10
```

```
new_s3_table
```

```
##        s3_var_names Freq
## 4   ENSG00000100078    8
## 10  ENSG00000112214    8
## 12  ENSG00000115112    9
## 13  ENSG00000115425    6
## 18  ENSG00000122136    7
## 26  ENSG00000134258    7
## 27  ENSG00000138061    9
## 31  ENSG00000143512    8
## 35  ENSG00000158516    6
## 38  ENSG00000161643    8
## 49  ENSG00000167476   10
## 50  ENSG00000167755    9
## 58  ENSG00000175170    6
## 71  ENSG00000185291    7
## 76  ENSG00000188312    6
## 82  ENSG00000196449    7
## 87  ENSG00000201201    7
## 89  ENSG00000203942    8
## 92  ENSG00000204837    7
## 94  ENSG00000205871    9
## 101 ENSG00000215319    6
## 103 ENSG00000215771    7
## 104 ENSG00000217241    9
## 106 ENSG00000220548    7
## 108 ENSG00000224830    7
## 109 ENSG00000224953    6
## 111 ENSG00000225163    8
## 112 ENSG00000225208   10
## 113 ENSG00000225420    9
## 114 ENSG00000225611    6
## 117 ENSG00000225950    9
## 118 ENSG00000226161    6
## 123 ENSG00000227416   10
## 124 ENSG00000227582   10
## 130 ENSG00000229515    9
## 133 ENSG00000230216    7
## 134 ENSG00000230397    7
## 140 ENSG00000231859    7
## 141 ENSG00000232411    7
## 142 ENSG00000232606    9
```

```
## 146 ENSG00000233060    8
## 151 ENSG00000234369    6
## 156 ENSG00000235038   10
## 160 ENSG00000236129    7
## 161 ENSG00000236230    9
## 165 ENSG00000236576    7
## 166 ENSG00000237506    7
## 167 ENSG00000237672    9
## 173 ENSG00000240194    6
## 176 ENSG00000240964    9
## 177 ENSG00000241243    8
## 179 ENSG00000242169    9
## 181 ENSG00000243547    9
## 182 ENSG00000244211    9
## 185 ENSG00000246422    8
## 188 ENSG00000249731    6
## 190 ENSG00000250357    7
## 191 ENSG00000250378    8
## 197 ENSG00000251209    6
## 198 ENSG00000251294    6
## 199 ENSG00000251449    7
## 200 ENSG00000251689    9
## 201 ENSG00000252821    8
## 205 ENSG00000253552   10
## 207 ENSG00000253796   10
## 214 ENSG00000255194   10
## 217 ENSG00000255484    6
## 218 ENSG00000255723   10
## 221 ENSG00000256804    7
## 223 ENSG00000257023    9
## 226 ENSG00000258308    9
## 227 ENSG00000258401   10
## 231 ENSG00000258673    7
## 232 ENSG00000258758    9
## 233 ENSG00000258927    8
## 236 ENSG00000260152   10
## 237 ENSG00000260672   10
## 238 ENSG00000260979    6
## 241 ENSG00000262730   10
## 244 ENSG00000263499    9
## 245 ENSG00000263609   10
## 246 ENSG00000263990    9
## 247 ENSG00000264672    7
## 249 ENSG00000264924   10
## 250 ENSG00000264956    7
## 251 ENSG00000265630    9
## 254 ENSG00000267169    6
## 256 ENSG00000267645    9
## 263 ENSG00000271590    6
## 269 ENSG00000272157   10
## 271 ENSG00000272239    8
## 273 ENSG00000272600    6
## 284 ENSG00000273983   10
## 286 ENSG00000274181   10
```

```
## 288 ENSG00000274478    7
## 290 ENSG00000275371    9
## 291 ENSG00000276523    6
## 292 ENSG00000276704    8
## 293 ENSG00000277233    6
## 296 ENSG00000278902    6
## 301 ENSG00000279958   10
## 303 ENSG00000281103    7
## 305 ENSG00000283267    6
## 307 ENSG00000283473   10
## 308 ENSG00000283611    6
```

```
new_s4_table
```

```
##        s4_var_names Freq
## 4   ENSG00000073598   10
## 8   ENSG00000107562    9
## 20  ENSG00000141682    6
## 22  ENSG00000147255    8
## 24  ENSG00000151468    6
## 29  ENSG00000161103    6
## 35  ENSG00000164107    8
## 41  ENSG00000168631    6
## 51  ENSG00000182366    6
## 52  ENSG00000183148    6
## 59  ENSG00000185390    6
## 63  ENSG00000189051    6
## 67  ENSG00000198398    9
## 73  ENSG00000205976   10
## 76  ENSG00000207217    8
## 83  ENSG00000214651    6
## 88  ENSG00000223745    9
## 90  ENSG00000225179    9
## 91  ENSG00000225362   10
## 92  ENSG00000226337   10
## 94  ENSG00000226954    6
## 97  ENSG00000227470    7
## 98  ENSG00000228035    7
## 101 ENSG00000228661    8
## 105 ENSG00000229703    6
## 106 ENSG00000229994    8
## 115 ENSG00000231830    9
## 118 ENSG00000232581    8
## 120 ENSG00000233405    7
## 122 ENSG00000233589    8
## 131 ENSG00000235397    6
## 134 ENSG00000235558    7
## 137 ENSG00000235776    6
## 145 ENSG00000239367    9
## 149 ENSG00000240634    8
## 153 ENSG00000242150    7
## 154 ENSG00000242296    7
## 156 ENSG00000244002    7
## 161 ENSG00000248988    6
## 165 ENSG00000249514    8
```

```
## 170 ENSG00000250260    6
## 172 ENSG00000250740    7
## 179 ENSG00000254187   10
## 186 ENSG00000256374    7
## 198 ENSG00000259201    6
## 199 ENSG00000259675    6
## 203 ENSG00000260185    7
## 204 ENSG00000260433    6
## 205 ENSG00000261038    6
## 207 ENSG00000261405    9
## 211 ENSG00000263574    9
## 212 ENSG00000263862    7
## 213 ENSG00000264775    6
## 224 ENSG00000270050    8
## 225 ENSG00000270269    8
## 226 ENSG00000271711    7
## 239 ENSG00000273415    6
## 240 ENSG00000273445    6
## 244 ENSG00000274611    7
## 252 ENSG00000278791    6
## 253 ENSG00000279752    7
## 256 ENSG00000280308    7
```

Save genes selected – uncomment if you want to save it as csv files

```
#write.csv(s1_var_name,"./4k_stage1_genes.csv")
#write.csv(s2_var_name,"./4k_stage2_genes.csv")
#write.csv(s3_var_name,"./4k_stage3_genes.csv")
#write.csv(s4_var_name,"./4k_stage4_genes.csv")

#comb <- qpcR:::cbind.na(s1_var_name, s2_var_name, s3_var_name, s4_var_name)
#write.csv(comb,"./4k_stages_combined_genes.csv")
```