Pertusis challenge 1.1

Slim Fourati

2024-01-12

```
suppressPackageStartupMessages(library(package = "knitr"))
suppressPackageStartupMessages(library(package = "glmnet"))
suppressPackageStartupMessages(library(package = "edgeR"))
suppressPackageStartupMessages(library(package = "biomaRt"))
suppressPackageStartupMessages(library(package = "GSVA"))
suppressPackageStartupMessages(library(package = "agua"))
suppressPackageStartupMessages(library(package = "tidyverse"))
```

```
workDir <- "/Users/iew5629/Desktop/Projects/20231013_Pertussis/Workspace"
options(readr.show_col_types = FALSE)
agua::h2o_start()
```

```
# 2020
pts2020DF <- read_tsv(file           = file.path(workDir, "documents/2020LD_subject.tsv"))
sample2020DF <- read_tsv(file           = file.path(workDir, "documents/2020LD_specimen.tsv"))
ab2020DF <- read_tsv(file = file.path(workDir, "documents/2020LD_plasma_ab_titer.tsv"))
fcm2020DF <- read_tsv(file = file.path(workDir, "documents/2020LD_pbmc_cell_frequency.tsv"))
olink2020DF <- read_tsv(file = file.path(workDir, "documents/2020LD_plasma_cytokine_concentration.tsv"))
ge2020DF <- read_tsv(file = file.path(workDir, "documents/2020LD_pbmc_gene_expression.tsv"))
```

```
# 2021
pts2021DF <- read_tsv(file = file.path(workDir, "documents/2021LD_subject.tsv"))
sample2021DF <- read_tsv(file = file.path(workDir, "documents/2021LD_specimen.tsv"))
ab2021DF <- read_tsv(file = file.path(workDir, "documents/2021LD_plasma_ab_titer.tsv"))
fcm2021DF <- read_tsv(file = file.path(workDir, "documents/2021LD_pbmc_cell_frequency.tsv"))
olink2021DF <- read_tsv(file = file.path(workDir, "documents/2021LD_plasma_cytokine_concentration.tsv"))
ge2021DF <- read_tsv(file = file.path(workDir, "documents/2021LD_pbmc_gene_expression.tsv"))
```

```
# 2021
pts2022DF <- read_tsv(file = file.path(workDir, "documents/2022BD_subject.tsv"))
sample2022DF <- read_tsv(file = file.path(workDir, "documents/2022BD_specimen.tsv"))
ab2022DF <- read_tsv(file = file.path(workDir, "documents/2022BD_plasma_ab_titer.tsv"))
```

Challenge1.1

```
yDF <- ab2020DF %>%
  filter(isotype %in% "IgG" & antigen %in% "PT") %>%
  merge(y = sample2020DF, by = "specimen_id", all.x = TRUE) %>%
  merge(y = pts2020DF, by = "subject_id", all.x = TRUE) %>%
  filter(planned_day_relative_to_boost %in% 14) %>%
  select(subject_id, MFI_normalised)
```

pts info

```
xDF <- pts2020DF %>%
  dplyr::select(-dataset) %>%
  column_to_rownames(var = "subject_id") %>%
  mutate_if(is.character, as.factor) %>%
  mutate_if(is.Date, as.factor) %>%
  slice(match(yDF$subject_id, table = rownames(.)))

xDF <- sapply(xDF, as.numeric) %>%
  scale() %>%
  as.matrix()
fit <- cv.glmnet(x = xDF,
                 y = rank(yDF$MFI_normalised),
                 nfolds = nrow(xDF))
plot(fit)
```

```
set.seed(seed= 3)
trainDF <- xDF %>%
  as.data.frame() %>%
  mutate(MFI_normalised = yDF$MFI_normalised)

auto_fit <- auto_ml() %>%
     set_engine("h2o", max_runtime_secs = 5) %>%
     set_mode("regression") %>%
     fit(MFI_normalised ~ ., data = trainDF)

plot(predict(auto_fit, trainDF)$.pred,
     yDF$MFI_normalised)

cor.test(predict(auto_fit, trainDF)$.pred,
     yDF$MFI_normalised)

cor.test(rank(-1 * predict(auto_fit, trainDF)$.pred),
     rank(-1 * yDF$MFI_normalised))
```

BL ab

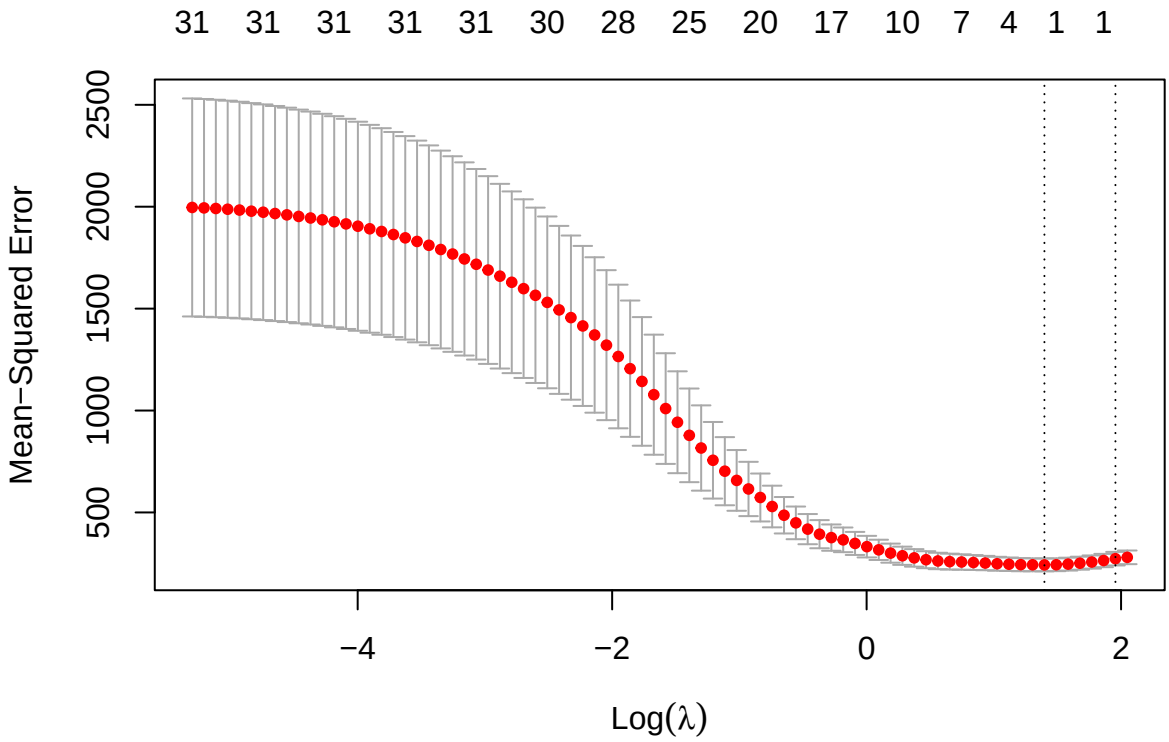
```
xDF <- ab2020DF %>%
  merge(y = sample2020DF, by = "specimen_id", all.x = TRUE) %>%
  merge(y = pts2020DF, by = "subject_id", all.x = TRUE) %>%
  filter(planned_day_relative_to_boost %in% 0 &
           grepl(pattern = "IgG", isotype)) %>%
  mutate(cname = paste0(isotype, "_", antigen),
         cname = make.names(cname)) %>%
  dplyr::select(subject_id, cname, MFI_normalised) %>%
  distinct() %>%
  pivot_wider(names_from = cname, values_from = MFI_normalised) %>%
  column_to_rownames(var = "subject_id") %>%
  slice(match(yDF$subject_id, table = rownames(.)))

x2DF <- ab2021DF %>%
```

```
  merge(y = sample2021DF, by = "specimen_id", all.x = TRUE) %>%
  merge(y = pts2021DF, by = "subject_id", all.x = TRUE) %>%
  filter(planned_day_relative_to_boost %in% 0) %>%
  mutate(cname = paste0(isotype, "_", antigen),
         cname = make.names(cname)) %>%
  dplyr::select(subject_id, cname, MFI_normalised) %>%
  distinct() %>%
  pivot_wider(names_from = cname, values_from = MFI_normalised) %>%
  column_to_rownames(var = "subject_id")

xDF <- xDF[, intersect(colnames(xDF), colnames(x2DF))]
x2DF <- x2DF[, colnames(xDF)]

xDF <- scale(xDF)
x2DF <- scale(x2DF)
fit <- cv.glmnet(x = as.matrix(xDF),
                 y = rank(-yDF$MFI_normalised),
                 nfolds = nrow(xDF),
                 grouped = FALSE)
plot(fit)
```



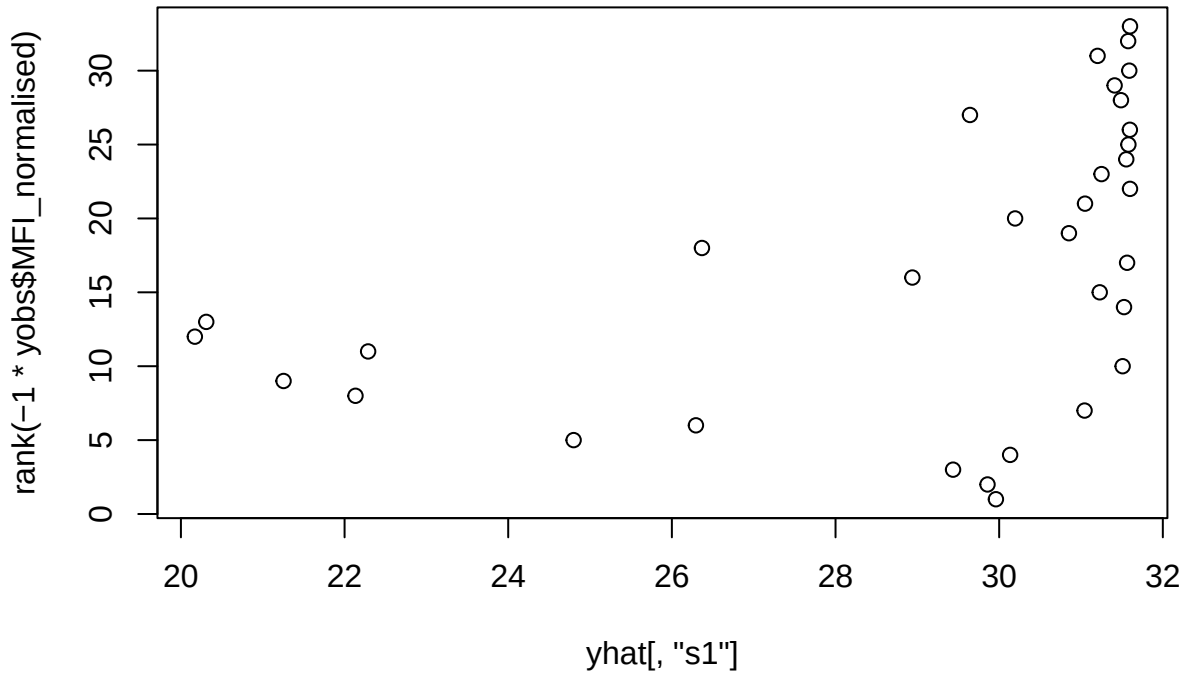

```
coef(fit, s = fit$lambda.min) %>%
  as.matrix() %>%
  as.data.frame() %>%
  rownames_to_column() %>%
  filter(s1 != 0) %>%
  kable()
```

rowname	s1
(Intercept)	29.0000000
IgG1_PT	-3.7301913
IgG1_FHA	-0.0465839

```
yhat <- predict(fit, newx = as.matrix(x2DF), s = fit$lambda.min, type = "response")

yobs <- ab2021DF %>%
  filter(isotype %in% "IgG" & antigen %in% "PT") %>%
  merge(y = sample2021DF, by = "specimen_id", all.x = TRUE) %>%
  merge(y = pts2021DF, by = "subject_id", all.x = TRUE) %>%
  filter(planned_day_relative_to_boost %in% 14) %>%
  dplyr::select(subject_id, MFI_normalised) %>%
  slice(match(rownames(x2DF), table = .$subject_id))

plot(yhat[, "s1"], rank(-1 * yobs$MFI_normalised))
```



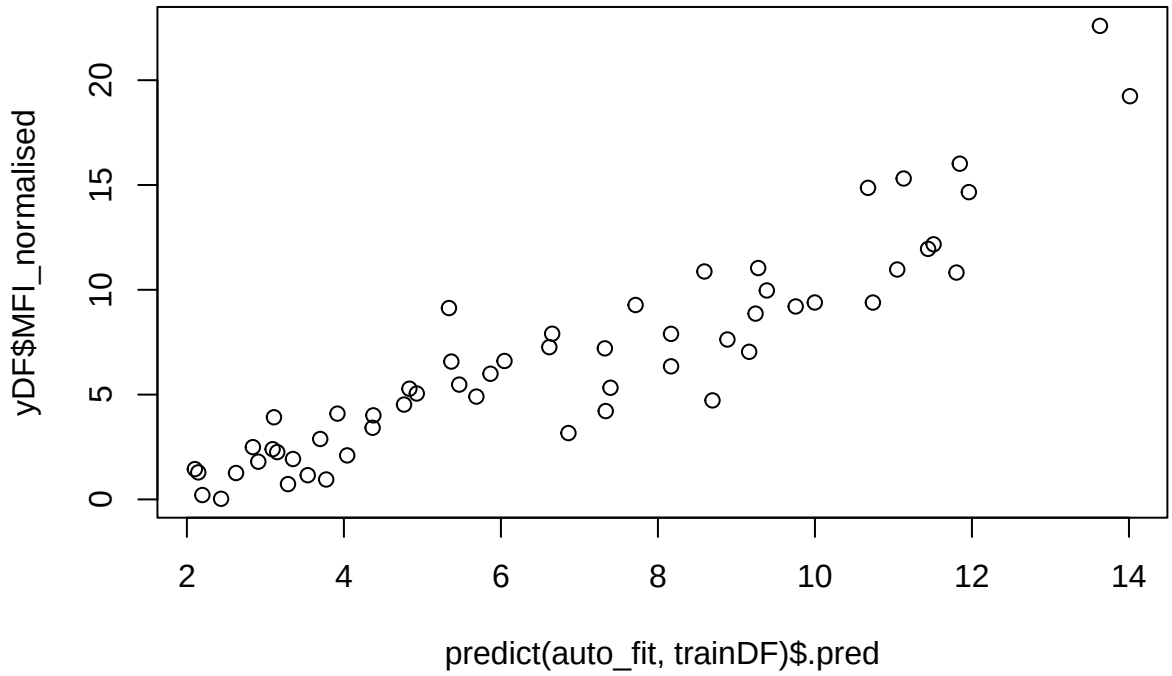

```
cor.test(yhat[, "s1"], rank(-1 * yobs$MFI_normalised))
```

```
##
##  Pearson's product-moment correlation
##
## data:  yhat[, "s1"] and rank(-1 * yobs$MFI_normalised)
## t = 2.8888, df = 31, p-value = 0.006995
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.1392581 0.6941116
## sample estimates:
##      cor
## 0.460549
```

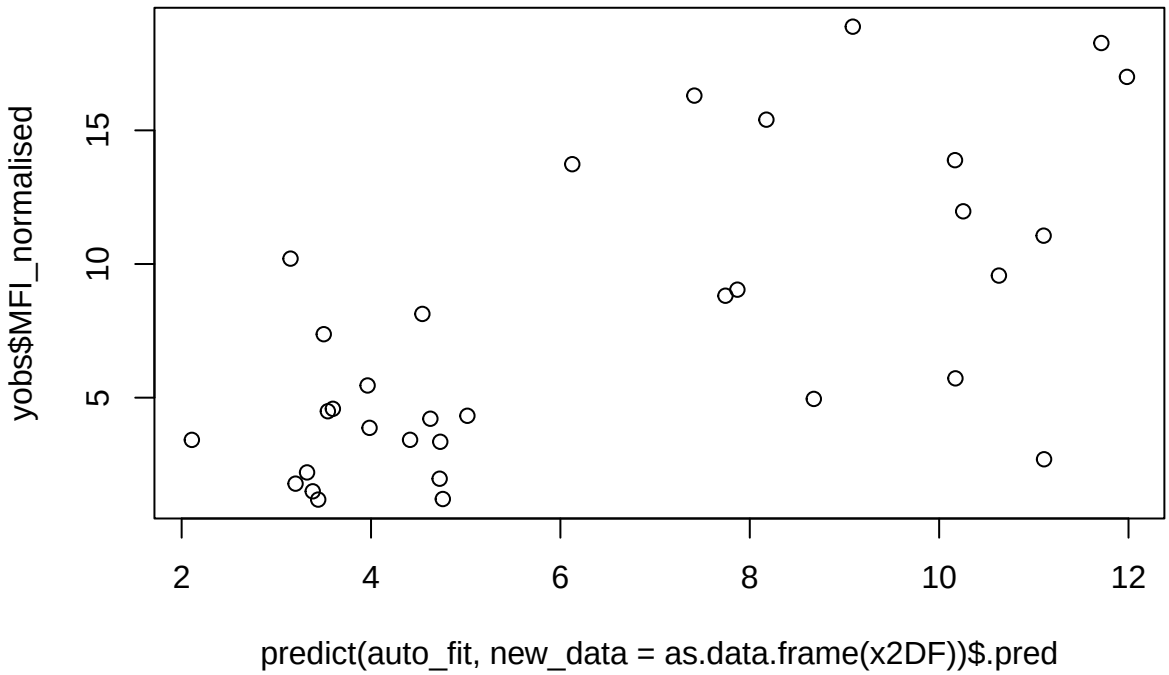
```r
set.seed(seed= 3)
trainDF <- xDF %>%
  as.data.frame() %>%
  mutate(MFI_normalised = yDF$MFI_normalised)

auto_fit <- auto_ml() %>%
     set_engine("h2o", max_runtime_secs = 5) %>%
     set_mode("regression") %>%
     fit(MFI_normalised ~ ., data = trainDF)

plot(predict(auto_fit, trainDF)$.pred,
     yDF$MFI_normalised)
```




```r
plot(predict(auto_fit, new_data = as.data.frame(x2DF))$.pred,
     yobs$MFI_normalised)
```




```
cor.test(predict(auto_fit, new_data = as.data.frame(x2DF))$.pred,
     yobs$MFI_normalised)
```

```
##
##  Pearson's product-moment correlation
##
## data:  predict(auto_fit, new_data = as.data.frame(x2DF))$.pred and yobs$MFI_normalised
## t = 4.7687, df = 31, p-value = 4.163e-05
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.3955285 0.8123888
## sample estimates:
##       cor
## 0.6505062
```

```
cor.test(rank(-1 * predict(auto_fit, new_data = as.data.frame(x2DF))$.pred),
     rank(-1 * yobs$MFI_normalised))
```

```
##
##  Pearson's product-moment correlation
##
## data:  rank(-1 * predict(auto_fit, new_data = as.data.frame(x2DF))$.pred) and rank(-1 * yobs$MFI_nor
## t = 4.2079, df = 31, p-value = 0.0002043
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.3274030 0.7839718
## sample estimates:
##       cor
## 0.6029412
```

FCM

```
xDF <- fcm2020DF %>%
  merge(y = sample2020DF, by = "specimen_id", all.x = TRUE) %>%
  merge(y = pts2020DF, by = "subject_id", all.x = TRUE) %>%
  filter(planned_day_relative_to_boost %in% 0) %>%
  dplyr::select(subject_id, cell_type_name, percent_live_cell) %>%
  pivot_wider(names_from = cell_type_name, values_from = percent_live_cell) %>%
  column_to_rownames(var = "subject_id") %>%
  slice(match(yDF$subject_id, table = rownames(.)))
xDF[is.na(xDF)] <- 0

x2DF <- fcm2021DF %>%
  merge(y = sample2021DF, by = "specimen_id", all.x = TRUE) %>%
  merge(y = pts2021DF, by = "subject_id", all.x = TRUE) %>%
  filter(planned_day_relative_to_boost %in% 0) %>%
  dplyr::select(subject_id, cell_type_name, percent_live_cell) %>%
  pivot_wider(names_from = cell_type_name, values_from = percent_live_cell) %>%
  column_to_rownames(var = "subject_id")
x2DF[is.na(x2DF)] <- 0

xDF <- xDF[, intersect(colnames(xDF), colnames(x2DF))]
x2DF <- x2DF[, colnames(xDF)]

xDF <- scale(xDF)
x2DF <- scale(x2DF)
fit <- cv.glmnet(x = as.matrix(xDF),
                 y = rank(-yDF[match(rownames(xDF), table = yDF$subject_id), "MFI_normalised"]),
                 nfolds = nrow(xDF),
                 grouped = FALSE)
plot(fit)
coef(fit, s = fit$lambda.min) %>%
  as.matrix() %>%
  as.data.frame() %>%
  rownames_to_column() %>%
  filter(s1 != 0) %>%
  kable()

yhat <- predict(fit, newx = as.matrix(x2DF), s = fit$lambda.min, type = "response")

yobs <- ab2021DF %>%
  filter(isotype %in% "IgG" & antigen %in% "PT") %>%
  merge(y = sample2021DF, by = "specimen_id", all.x = TRUE) %>%
  merge(y = pts2021DF, by = "subject_id", all.x = TRUE) %>%
  filter(planned_day_relative_to_boost %in% 14) %>%
  dplyr::select(subject_id, MFI_normalised) %>%
  slice(match(rownames(x2DF), table = .$subject_id))

plot(yhat[as.character(yobs$subject_id), "s1"], rank(-1 * yobs$MFI_normalised))
cor.test(yhat[as.character(yobs$subject_id), "s1"], rank(-1 * yobs$MFI_normalised))
```

Olink

```
xDF <- olink2020DF %>%
  merge(y = sample2020DF, by = "specimen_id", all.x = TRUE) %>%
  merge(y = pts2020DF, by = "subject_id", all.x = TRUE) %>%
  filter(planned_day_relative_to_boost %in% 0) %>%
  dplyr::select(subject_id, protein_id, protein_expression) %>%
  pivot_wider(names_from = protein_id, values_from = protein_expression) %>%
  column_to_rownames(var = "subject_id") %>%
  slice(match(yDF$subject_id, table = rownames(.)))

x2DF <- olink2021DF %>%
  merge(y = sample2021DF, by = "specimen_id", all.x = TRUE) %>%
  merge(y = pts2021DF, by = "subject_id", all.x = TRUE) %>%
  filter(planned_day_relative_to_boost %in% 0) %>%
  dplyr::select(subject_id, protein_id, protein_expression) %>%
  pivot_wider(names_from = protein_id, values_from = protein_expression) %>%
  column_to_rownames(var = "subject_id")
x2DF[is.na(x2DF)] <- 0

xDF <- xDF[, intersect(colnames(xDF), colnames(x2DF))]
x2DF <- x2DF[, colnames(xDF)]
xDF <- scale(xDF)
x2DF <- scale(x2DF)
fit <- cv.glmnet(x = as.matrix(xDF),
                 y = rank(-yDF[match(rownames(xDF), table = yDF$subject_id), "MFI_normalised"]),
                 nfolds = nrow(xDF),
                 grouped = FALSE)
plot(fit)
coef(fit, s = fit$lambda.min) %>%
  as.matrix() %>%
  as.data.frame() %>%
  rownames_to_column() %>%
  filter(s1 != 0) %>%
  kable()

yhat <- predict(fit, newx = as.matrix(x2DF), s = fit$lambda.min, type = "response")

yobs <- ab2021DF %>%
  filter(isotype %in% "IgG" & antigen %in% "PT") %>%
  merge(y = sample2021DF, by = "specimen_id", all.x = TRUE) %>%
  merge(y = pts2021DF, by = "subject_id", all.x = TRUE) %>%
  filter(planned_day_relative_to_boost %in% 14) %>%
  dplyr::select(subject_id, MFI_normalised) %>%
  slice(match(rownames(x2DF), table = .$subject_id))

plot(yhat[as.character(yobs$subject_id), "s1"], rank(-1 * yobs$MFI_normalised))
cor.test(yhat[as.character(yobs$subject_id), "s1"], rank(-1 * yobs$MFI_normalised))
```

GE

```r
# extract d0 GE
d0RawMat <- ge2020DF %>%
  merge(y = sample2020DF, by = "specimen_id", all.x = TRUE) %>%
  merge(y = pts2020DF, by = "subject_id", all.x = TRUE) %>%
  filter(planned_day_relative_to_boost %in% 0) %>%
  dplyr::select(subject_id, versioned_ensembl_gene_id, raw_count) %>%
  pivot_wider(names_from = subject_id, values_from = raw_count) %>%
  column_to_rownames(var = "versioned_ensembl_gene_id")

# TMM normalization
dge <- DGEList(counts       = d0RawMat,
               remove.zeros = TRUE)
dge <- calcNormFactors(object = dge, method = "TMM")
normalizedCounts <- cpm(dge, normalized.lib.sizes = TRUE, log = TRUE)

# convert ensembl_gene_id to gene_symbol
human <- useMart("ensembl", dataset = "hsapiens_gene_ensembl")
id2symbol <- getBM(attributes  = c("ensembl_gene_id_version", "hgnc_symbol"),
                   filters     = "ensembl_gene_id_version",
                   values      = rownames(normalizedCounts),
                   mart        = human)
geneD0Mat <- normalizedCounts %>%
  as.data.frame() %>%
  rownames_to_column(var = "ensembl_gene_id_version") %>%
  merge(y = id2symbol, by = "ensembl_gene_id_version") %>%
  select(-ensembl_gene_id_version) %>%
  slice(order(apply(select(., -hgnc_symbol), MARGIN = 1, FUN = var), decreasing = TRUE)) %>%
  filter(!duplicated(hgnc_symbol) & hgnc_symbol != "") %>%
  column_to_rownames(var = "hgnc_symbol") %>%
  as.matrix()

# GSVA with BTM
load(file = "/Users/iew5629/Downloads/GeneSets.rda")
BTM.geneSets <- BTM.geneSets[!grepl(pattern = "TBA", names(BTM.geneSets))]

flag <- BTM.geneSets %>%
  stack() %>%
  filter(values %in% rownames(geneD0Mat)) %>%
  group_by(ind) %>%
  summarize(n = n()) %>%
  arrange(desc(n)) %>%
  filter(n >= 4)

gsD0Mat <- gsva(expr = geneD0Mat, gset.idx.list = BTM.geneSets[flag$ind], verbose = FALSE)

# prediction
xDF <- t(gsD0Mat)

fit <- cv.glmnet(x = as.matrix(xDF),
                 y = rank(-yDF[match(rownames(xDF), table = yDF$subject_id), "MFI_normalised"]),
                 nfolds = nrow(xDF),
                 grouped = FALSE)
plot(fit)
```

```r
coef(fit, s = fit$lambda.min) %>%
  as.matrix() %>%
  as.data.frame() %>%
  rownames_to_column() %>%
  filter(s1 != 0) %>%
  kable()

# extract d0 GE of 2021
d0RawMat <- ge2021DF %>%
  merge(y = sample2021DF, by = "specimen_id", all.x = TRUE) %>%
  merge(y = pts2021DF, by = "subject_id", all.x = TRUE) %>%
  filter(planned_day_relative_to_boost %in% 0) %>%
  dplyr::select(subject_id, versioned_ensembl_gene_id, raw_count) %>%
  pivot_wider(names_from = subject_id, values_from = raw_count) %>%
  column_to_rownames(var = "versioned_ensembl_gene_id")

# TMM normalization
dge <- DGEList(counts       = d0RawMat,
               remove.zeros = TRUE)
dge <- calcNormFactors(object = dge, method = "TMM")
normalizedCounts <- cpm(dge, normalized.lib.sizes = TRUE, log = TRUE)

# convert ensembl_gene_id to gene_symbol
id2symbol <- getBM(attributes  = c("ensembl_gene_id_version", "hgnc_symbol"),
                   filters     = "ensembl_gene_id_version",
                   values      = rownames(normalizedCounts),
                   mart        = human)
geneD0Mat <- normalizedCounts %>%
  as.data.frame() %>%
  rownames_to_column(var = "ensembl_gene_id_version") %>%
  merge(y = id2symbol, by = "ensembl_gene_id_version") %>%
  select(-ensembl_gene_id_version) %>%
  slice(order(apply(select(., -hgnc_symbol), MARGIN = 1, FUN = var), decreasing = TRUE)) %>%
  filter(!duplicated(hgnc_symbol) & hgnc_symbol != "") %>%
  column_to_rownames(var = "hgnc_symbol") %>%
  as.matrix()

# GSVA with BTM
gsD0Mat <- gsva(expr = geneD0Mat, gset.idx.list = BTM.geneSets[flag$ind], verbose = FALSE)

# prediction
x2DF <- t(gsD0Mat)
yhat <- predict(fit, newx = as.matrix(x2DF), s = fit$lambda.min, type = "response")

yobs <- ab2021DF %>%
  filter(isotype %in% "IgG" & antigen %in% "PT") %>%
  merge(y = sample2021DF, by = "specimen_id", all.x = TRUE) %>%
  merge(y = pts2021DF, by = "subject_id", all.x = TRUE) %>%
  filter(planned_day_relative_to_boost %in% 14) %>%
  dplyr::select(subject_id, MFI_normalised) %>%
  slice(match(rownames(x2DF), table = .$subject_id))

plot(yhat[as.character(yobs$subject_id), "s1"], rank(-1 * yobs$MFI_normalised))
```

```
cor.test(yhat[as.character(yobs$subject_id), "s1"], rank(-1 * yobs$MFI_normalised))
```

2022 test set

```
set.seed(seed= 3)
# train on 2020 and 2021 baseline ab only
yDF <- ab2020DF %>%
  filter(isotype %in% "IgG" & antigen %in% "PT") %>%
  merge(y = sample2020DF, by = "specimen_id", all.x = TRUE) %>%
  merge(y = pts2020DF, by = "subject_id", all.x = TRUE) %>%
  filter(planned_day_relative_to_boost %in% 14) %>%
  select(subject_id, MFI_normalised)

xDF <- ab2020DF %>%
  merge(y = sample2020DF, by = "specimen_id", all.x = TRUE) %>%
  merge(y = pts2020DF, by = "subject_id", all.x = TRUE) %>%
  filter(planned_day_relative_to_boost %in% 0 &
           grepl(pattern = "IgG", isotype)) %>%
  mutate(cname = paste0(isotype, "_", antigen),
         cname = make.names(cname)) %>%
  dplyr::select(subject_id, cname, MFI_normalised) %>%
  distinct() %>%
  pivot_wider(names_from = cname, values_from = MFI_normalised) %>%
  column_to_rownames(var = "subject_id") %>%
  slice(match(yDF$subject_id, table = rownames(.)))

x2DF <- ab2021DF %>%
  merge(y = sample2021DF, by = "specimen_id", all.x = TRUE) %>%
  merge(y = pts2021DF, by = "subject_id", all.x = TRUE) %>%
  filter(planned_day_relative_to_boost %in% 0) %>%
  mutate(cname = paste0(isotype, "_", antigen),
         cname = make.names(cname)) %>%
  dplyr::select(subject_id, cname, MFI_normalised) %>%
  distinct() %>%
  pivot_wider(names_from = cname, values_from = MFI_normalised) %>%
  column_to_rownames(var = "subject_id")

yobs <- ab2021DF %>%
  filter(isotype %in% "IgG" & antigen %in% "PT") %>%
  merge(y = sample2021DF, by = "specimen_id", all.x = TRUE) %>%
  merge(y = pts2021DF, by = "subject_id", all.x = TRUE) %>%
  filter(planned_day_relative_to_boost %in% 14) %>%
  dplyr::select(subject_id, MFI_normalised) %>%
  slice(match(rownames(x2DF), table = .$subject_id))

x3DF <- ab2022DF %>%
  merge(y = sample2022DF, by = "specimen_id", all.x = TRUE) %>%
  merge(y = pts2022DF, by = "subject_id", all.x = TRUE) %>%
  filter(planned_day_relative_to_boost %in% 0) %>%
  mutate(cname = paste0(isotype, "_", antigen),
         cname = make.names(cname)) %>%
  dplyr::select(subject_id, cname, MFI_normalised) %>%
```

```
  distinct() %>%
  pivot_wider(names_from = cname, values_from = MFI_normalised) %>%
  column_to_rownames(var = "subject_id")

xDF <- xDF[, intersect(colnames(xDF), colnames(x2DF))]
x2DF <- x2DF[, colnames(xDF)]
x3DF <- x3DF[, colnames(xDF)]

xDF <- scale(xDF)
x2DF <- scale(x2DF)
x3DF <- scale(x3DF)

trainDF <- rbind(xDF, x2DF) %>%
  as.data.frame() %>%
  mutate(MFI_normalised = c(yDF$MFI_normalised, yobs$MFI_normalised))

auto_fit <- auto_ml() %>%
     set_engine("h2o", max_runtime_secs = 5) %>%
     set_mode("regression") %>%
     fit(MFI_normalised ~ ., data = trainDF)

yhat <- predict(auto_fit, new_data = x3DF)$.pred
rhat <- rank(-1 * yhat)
print(cbind(rownames(x3DF), rhat))
```

```
##             rhat
##  [1,] "97"  "2.5"
##  [2,] "98"  "15.5"
##  [3,] "99"  "15.5"
##  [4,] "100" "17"
##  [5,] "101" "10.5"
##  [6,] "102" "10.5"
##  [7,] "103" "7.5"
##  [8,] "104" "18.5"
##  [9,] "105" "10.5"
## [10,] "106" "18.5"
## [11,] "107" "7.5"
## [12,] "108" "14"
## [13,] "109" "4"
## [14,] "110" "21"
## [15,] "111" "10.5"
## [16,] "112" "20"
## [17,] "114" "5"
## [18,] "115" "13"
## [19,] "116" "1"
## [20,] "117" "2.5"
## [21,] "118" "6"
```