Penalized Logistic Regression Approaches to Prediction of CRC

November 29, 2018

Simple Logistic Regression with Variable Selection

```
setwd("~/Documents/Post Geno. Analysis/project")
microbes <- read.csv("microbes.csv", row.names = 1)
metabolites <- read.csv("metabolites.csv", row.names = 1)
combined <-cbind(microbes, metabolites)
samples <- read.csv("samples.csv", row.names = 1)
labels <- samples$case
```

Microbes Analysis

We split the data into 80% training, and 20% testing.

```
library(caret)
```

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

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

```
set.seed(42)
training_rows <- createDataPartition(labels, p = 0.8, list = FALSE)
# Split data
y.train <- as.vector(labels[training_rows])
y.test <- as.vector(labels[-training_rows])
x.train <- as.matrix(microbes[training_rows, ])
x.test <- as.matrix(microbes[-training_rows, ])
```

Run Lasso, Ridge, and Elastic Net:

```
library(glmnet)
```

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

```
## Loading required package: foreach
```

```
## Loaded glmnet 2.0-16
```

```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following object is masked from 'package:glmnet':
##
##     auc
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
for (i in 0:10)
{
  assign(paste("fit", i, sep=""), cv.glmnet(x.train, y.train,type.measure="mse",alpha=i/10,family="bino
}

yhat0 <- predict(fit0, s=fit0$lambda.min, newx=x.test, type = "class")
yhat1 <- predict(fit1, s=fit1$lambda.min, newx=x.test, type = "class")
yhat2 <- predict(fit2, s=fit2$lambda.min, newx=x.test, type = "class")
yhat3 <- predict(fit3, s=fit3$lambda.min, newx=x.test, type = "class")
yhat4 <- predict(fit4, s=fit4$lambda.min, newx=x.test, type = "class")
yhat5 <- predict(fit5, s=fit5$lambda.min, newx=x.test, type = "class")
yhat6 <- predict(fit6, s=fit6$lambda.min, newx=x.test, type = "class")
yhat7 <- predict(fit7, s=fit7$lambda.min, newx=x.test, type = "class")
yhat8 <- predict(fit8, s=fit8$lambda.min, newx=x.test, type = "class")
yhat9 <- predict(fit9, s=fit9$lambda.min, newx=x.test, type = "class")
yhat10 <- predict(fit10, s=fit10$lambda.min, newx=x.test, type = "class")

models <- list(yhat0, yhat1, yhat2, yhat3, yhat4, yhat5, yhat6, yhat7, yhat8, yhat9, yhat10)

for (p in models) {
  accuracy <- 1 - sum(abs(y.test - as.integer(p))) / nrow(y.test)
  roc_obj <- roc(y.test, as.integer(p))
  print(paste("Accuracy:", accuracy, "AUC:", auc(roc_obj)))
}
```

```
## [1] "Accuracy:  AUC: 0.5"
## [1] "Accuracy:  AUC: 0.5"
## [1] "Accuracy:  AUC: 0.5"
## [1] "Accuracy:  AUC: 0.5"
## [1] "Accuracy:  AUC: 0.5"
## [1] "Accuracy:  AUC: 0.5"
## [1] "Accuracy:  AUC: 0.5"
## [1] "Accuracy:  AUC: 0.5"
## [1] "Accuracy:  AUC: 0.5"
## [1] "Accuracy:  AUC: 0.5"
## [1] "Accuracy:  AUC: 0.5"
```

Metabolites Analysis

We split the data into 80% training, and 20% testing.

```
library(caret)
set.seed(42)
training_rows <- createDataPartition(labels, p = 0.8, list = FALSE)
# Split data
y.train <- as.vector(labels[training_rows])
y.test <- as.vector(labels[-training_rows])
x.train <- as.matrix(metabolites[training_rows, ])
x.test <- as.matrix(metabolites[-training_rows, ])
```

Run Lasso, Ridge, and Elastic Net:

```
library(glmnet)
library(pROC)

for (i in 0:10)
{
  assign(paste("fit", i, sep=""), cv.glmnet(x.train, y.train,type.measure="mse",alpha=i/10,family="bino
}

yhat0 <- predict(fit0, s=fit0$lambda.min, newx=x.test, type = "class")
yhat1 <- predict(fit1, s=fit1$lambda.min, newx=x.test, type = "class")
yhat2 <- predict(fit2, s=fit2$lambda.min, newx=x.test, type = "class")
yhat3 <- predict(fit3, s=fit3$lambda.min, newx=x.test, type = "class")
yhat4 <- predict(fit4, s=fit4$lambda.min, newx=x.test, type = "class")
yhat5 <- predict(fit5, s=fit5$lambda.min, newx=x.test, type = "class")
yhat6 <- predict(fit6, s=fit6$lambda.min, newx=x.test, type = "class")
yhat7 <- predict(fit7, s=fit7$lambda.min, newx=x.test, type = "class")
yhat8 <- predict(fit8, s=fit8$lambda.min, newx=x.test, type = "class")
yhat9 <- predict(fit9, s=fit9$lambda.min, newx=x.test, type = "class")
yhat10 <- predict(fit10, s=fit10$lambda.min, newx=x.test, type = "class")

models <- list(yhat0, yhat1, yhat2, yhat3, yhat4, yhat5, yhat6, yhat7, yhat8, yhat9, yhat10)

for (p in models) {
  accuracy <- 1 - sum(abs(y.test - as.integer(p))) / nrow(y.test)
  roc_obj <- roc(y.test, as.integer(p))
  print(paste("Accuracy:", accuracy, "AUC:", auc(roc_obj)))
}
```

```
## [1] "Accuracy:  AUC: 0.6"
## [1] "Accuracy:  AUC: 0.66875"
## [1] "Accuracy:  AUC: 0.71875"
## [1] "Accuracy:  AUC: 0.71875"
## [1] "Accuracy:  AUC: 0.61875"
## [1] "Accuracy:  AUC: 0.61875"
## [1] "Accuracy:  AUC: 0.61875"
## [1] "Accuracy:  AUC: 0.61875"
## [1] "Accuracy:  AUC: 0.61875"
## [1] "Accuracy:  AUC: 0.66875"
## [1] "Accuracy:  AUC: 0.61875"
```

Combined Analysis

We split the data into 80% training, and 20% testing.

```
library(caret)
set.seed(42)
training_rows <- createDataPartition(labels, p = 0.8, list = FALSE)
# Split data
y.train <- as.vector(labels[training_rows])
y.test <- as.vector(labels[-training_rows])
x.train <- as.matrix(combined[training_rows, ])
x.test <- as.matrix(combined[-training_rows, ])
```

Run Lasso, Ridge, and Elastic Net:

```
library(glmnet)
library(pROC)

for (i in 0:10)
{
  assign(paste("fit", i, sep=""), cv.glmnet(x.train, y.train,type.measure="mse",alpha=i/10,family="bino
}

yhat0 <- predict(fit0, s=fit0$lambda.min, newx=x.test, type = "class")
yhat1 <- predict(fit1, s=fit1$lambda.min, newx=x.test, type = "class")
yhat2 <- predict(fit2, s=fit2$lambda.min, newx=x.test, type = "class")
yhat3 <- predict(fit3, s=fit3$lambda.min, newx=x.test, type = "class")
yhat4 <- predict(fit4, s=fit4$lambda.min, newx=x.test, type = "class")
yhat5 <- predict(fit5, s=fit5$lambda.min, newx=x.test, type = "class")
yhat6 <- predict(fit6, s=fit6$lambda.min, newx=x.test, type = "class")
yhat7 <- predict(fit7, s=fit7$lambda.min, newx=x.test, type = "class")
yhat8 <- predict(fit8, s=fit8$lambda.min, newx=x.test, type = "class")
yhat9 <- predict(fit9, s=fit9$lambda.min, newx=x.test, type = "class")
yhat10 <- predict(fit10, s=fit10$lambda.min, newx=x.test, type = "class")

models <- list(yhat0, yhat1, yhat2, yhat3, yhat4, yhat5, yhat6, yhat7, yhat8, yhat9, yhat10)

for (p in models) {
  accuracy <- 1 - sum(abs(y.test - as.integer(p))) / nrow(y.test)
  roc_obj <- roc(y.test, as.integer(p))
  print(paste("Accuracy:", accuracy, "AUC:", auc(roc_obj)))
}
```

```
## [1] "Accuracy:  AUC: 0.6"
## [1] "Accuracy:  AUC: 0.6"
## [1] "Accuracy:  AUC: 0.65"
## [1] "Accuracy:  AUC: 0.65"
## [1] "Accuracy:  AUC: 0.65"
## [1] "Accuracy:  AUC: 0.61875"
## [1] "Accuracy:  AUC: 0.65"
## [1] "Accuracy:  AUC: 0.61875"
## [1] "Accuracy:  AUC: 0.61875"
## [1] "Accuracy:  AUC: 0.66875"
## [1] "Accuracy:  AUC: 0.65"
```