Generalized linear models - Logistic Regression

Marta Belchior Lopes

October, 2023

This practical addresses a binary classification problem using regularized logistic regression to predict the tumor type of lower-grade glioma patients. The data used was extracted form The Cancer Genome Atlas data portal.

1 Matrix construction

1.1 Loading the data

```
# Load a lower-grade glioma (LGG) gene expression (RNA-seq) dataset with 2 class labels: astrocytoma (LGG-a) and
oligodendroglioma (LGG-od)

load("~/LGG_glioma.Rdata")

## Ydata is the RNA-seq data and Xdata is the response variable with the corresponding class
dim(Xdata)
```

```
## [1]  381 20176
```

```
# RNA-seq data from 381 patients measured over 20501 genes
Xdata[1:5,1:8] # subsample for matrix visualization
```

```
##                A1BG A1CF    A2BP1    A2LD1     A2M   A2ML1   A4GALT   A4GNT
## TCGA-CS-4938 94.1095    0 22.6558  13.4844 14783.71 146.0038  73.4214 0.0000
## TCGA-CS-4941 72.2326    0 524.4997 144.0856 17944.72 521.2941 159.7654 1.0352
## TCGA-CS-4942 74.4533    0 368.5121  51.4083 19269.89 174.3945  44.2907 0.2460
## TCGA-CS-4943 29.9858    0  44.9983  13.9821 11719.76 179.0148  35.5421 1.3043
## TCGA-CS-4944 26.7132    0 105.4092  18.0154 10894.96 159.2746 114.9938 0.5044
```

```
Ydata[1:10]
```

```
##  [1] "LGG-a"  "LGG-a"  "LGG-a"  "LGG-a"  "LGG-a"  "LGG-a"  "LGG-a"  "LGG-od"
```

```
length(Ydata)
```

```
## [1] 381
```

```
summary(as.factor(Ydata))
```

```
##  LGG-a LGG-od
##    193    188
```

```
# Load R packages
library(ggplot2)
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 4.1.1
```

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

```
## Warning: package 'Matrix' was built under R version 4.1.1
```

```
## Loaded glmnet 4.1-4
```

```
library(pROC)
```

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

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

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

1.2 Data preprocessing

```
## Data filtering
# removing samples with standard deviation zero
Xdata_sd <- sapply(seq(ncol(Xdata)), function(ix) (sd(Xdata[,ix])))
Xdata <- Xdata[,Xdata_sd != 0]
dim(Xdata)
```

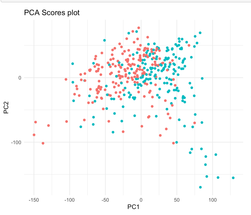
```
## [1]  381 20176
```

```
# 381 patients measured over 20176 genes

## Data normalization
# computing the z-score
Xdata_sc <- scale(Xdata)
```

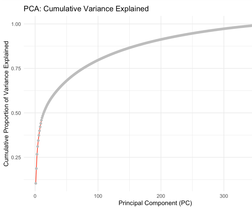
2 Exploratory data analysis

2.1 Principal component analysis (PCA)

```
# Perform PCA
glioma_pca <- prcomp(Xdata, scale = TRUE)

# Scores of the principal components
scores <- as.data.frame(glioma_pca$x)
scores$class <- Ydata

# Variance explained by each component
variance_explained <- glioma_pca$sdev^2
prop_variance_explained <- variance_explained / sum(variance_explained)
cumulative_prop_variance <- cumsum(prop_variance_explained)

# Plot PCA scores plot
ggplot(scores, aes(x = PC1, y = PC2, color = class)) +
  geom_point() +
  labs(title = paste("PCA Scores plot"),
       x = "PC1",
       y = "PC2") +
  theme_minimal()
```




```
# Plot % of variance explained
pca_summary <- data.frame(
  PC = 1:length(cumulative_prop_variance),
  Cumulative_Prop_Variance = cumulative_prop_variance
)

ggplot(pca_summary, aes(x = PC, y = Cumulative_Prop_Variance)) +
  geom_line(color = "red") +
  geom_point(color = "gray") +
  labs(title = "PCA: Cumulative Variance Explained",
       x = "Principal Component (PC)",
       y = "Cumulative Proportion of Variance Explained") +
  theme_minimal()
```



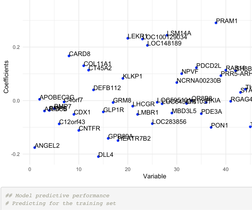
3 Regularized logistic regression

3.1 Model training

```
# Partition the data into training and test sets
set.seed(2021) # for reproducibility
test_ID <- sample(1:dim(Xdata_sc)[1], round(dim(Xdata_sc)[1]*0.25), replace=FALSE)

Ydata[Ydata=="LGG-a"] <- 1
Ydata[Ydata=="LGG-od"] <- 0

# train set
Xdata_train <- as.matrix(Xdata_sc[-test_ID,])
Ydata_train <- as.factor(Ydata[-test_ID])
# test set
Xdata_test <- as.matrix(Xdata_sc[test_ID,])
Ydata_test <- as.factor(Ydata[test_ID])

# Building the sparse logistic regression model (lambda optimized by cross-validation)

set.seed(1974) # for reproducibility

glioma_fit <- cv.glmnet(Xdata_train, Ydata_train, family="binomial", nfolds=10, alpha=1, type.measure="auc")
glioma_coef <- glioma_fit$glmnet.fit$beta[,which(glioma_fit$cvm == max(glioma_fit$cvm))]
genes_selected <- which(glioma_coef != 0)
length(genes_selected)
```

```
## [1] 50
```

```
data <- data.frame(
  x = 1:length(genes_selected),
  y = glioma_coef[genes_selected],
  labels = names(glioma_coef[genes_selected])
)

ggplot(data, aes(x = x, y = y, label = labels)) +
  geom_point(color = "blue") +
  geom_text(hjust = 0, vjust = 0) +
  labs(title = "Selected variable's coefficients",
       x = "Variable",
       y = "Coefficients") +
  theme_minimal()
```




```
## Model predictive performance
# Predicting for the training set
```

3.2 Model evaluation

```
pred_train <- predict(glioma_fit, Xdata_train, type="class", type.measure = "auc", s = "lambda.min")
# Confusion matrix for the train set
table(Ydata_train,pred_train)
```

```
##           pred_train
## Ydata_train   0   1
##           0 127  16
##           1   4 139
```

```
# Calculate AOC for the train set
roc_obj <- roc(as.numeric(as.character(Ydata_train)), as.numeric(pred_train))
```

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

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

```
auc(roc_obj)
```

```
## Area under the curve: 0.9301
```

```
# Predicting for a test set
pred_test <- predict(glioma_fit,Xdata_test,type="class")

# Confusion matrix for the test set
table(Ydata_test,pred_test)
```

```
##           pred_test
## Ydata_test  0  1
##          0 33 12
##          1  5 45
```

```
# Calculate AOC value for the test set
roc_obj <- roc(as.numeric(as.character(Ydata_test)), as.numeric(pred_test))
```

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

```
auc(roc_obj)
```

```
## Area under the curve: 0.8167
```