classification

bf2506

2023-05-06

```
library(tidyverse)
library(caret)
```

1. Load Data

```
load("final_data.RData")
final_data = final_data %>%
  mutate(binary_recovery_time = as.factor(binary_recovery_time))
levels(final_data$binary_recovery_time) = c("low", "high")
```

2. Train/test split

```
set.seed(2)
training_rows <- createDataPartition(final_data$binary_recovery_time,
                                     p = 0.8,
                                     list = F)
```

```
train_x <- model.matrix(binary_recovery_time~., final_data %>% dplyr::select(-id, -recovery_time))[trai
train_y <- final_data$binary_recovery_time[training_rows]
test_x <- model.matrix(binary_recovery_time~., final_data %>% dplyr::select(-id, -recovery_time))[-trai
test_y <- final_data$binary_recovery_time[-training_rows]

test_x_df = as.data.frame(test_x)
test_y_df = as.data.frame(test_y)
train_x_df = as.data.frame(train_x)
```

```
training_set <- final_data[training_rows,]
```

Setting methods

```
ctrl2 = trainControl(method = "cv", number = 10)
```

(a) Perform a logistic regression using the training data.

```
set.seed(2)
#fit a logistic regression using caret
model.glm = train(train_x, train_y, method = "glm", metric = "Accuracy",
                  trControl = ctrl2)
summary(model.glm)
```

```
##
```

```
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4670  -1.1407   0.6120   0.8681   1.6411
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -7.460e+01  1.388e+01  -5.374 7.70e-08 ***
## age            1.720e-02  1.092e-02   1.575 0.115307
## gender1       -3.090e-01  8.555e-02  -3.611 0.000304 ***
## race2         -3.677e-02  1.961e-01  -0.188 0.851260
## race3         -1.764e-01  1.070e-01  -1.648 0.099291 .
## race4          1.095e-03  1.439e-01   0.008 0.993928
## smoking1       2.959e-01  9.780e-02   3.026 0.002480 **
## smoking2       4.111e-01  1.549e-01   2.654 0.007949 **
## height         4.321e-01  8.133e-02   5.313 1.08e-07 ***
## weight        -4.735e-01  8.678e-02  -5.457 4.84e-08 ***
## bmi            1.437e+00  2.506e-01   5.737 9.66e-09 ***
## hypertension1  2.714e-01  1.415e-01   1.918 0.055148 .
## diabetes1     -3.350e-02  1.193e-01  -0.281 0.778852
## sbp           -1.345e-04  9.589e-03  -0.014 0.988810
## ldl           -5.396e-04  2.260e-03  -0.239 0.811341
## vaccine1      -5.989e-01  8.919e-02  -6.715 1.88e-11 ***
## severity1      6.580e-01  1.656e-01   3.974 7.07e-05 ***
## studyB        -1.103e+00  1.211e-01  -9.111  < 2e-16 ***
## studyC         3.074e-02  1.557e-01   0.197 0.843475
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 3577.1  on 2887  degrees of freedom
## Residual deviance: 3253.9  on 2869  degrees of freedom
## AIC: 3291.9
##
## Number of Fisher Scoring iterations: 4
```

```
model.glm$finalModel #My generalized linear model (GLM)
```

```
##
## Call:  NULL
##
## Coefficients:
##   (Intercept)            age        gender1          race2          race3
##    -7.460e+01      1.720e-02     -3.090e-01     -3.677e-02     -1.764e-01
##         race4       smoking1       smoking2         height         weight
##     1.095e-03      2.959e-01      4.111e-01      4.321e-01     -4.735e-01
##           bmi  hypertension1      diabetes1            sbp            ldl
##     1.437e+00      2.714e-01     -3.350e-02     -1.345e-04     -5.396e-04
##      vaccine1      severity1         studyB         studyC
##    -5.989e-01      6.580e-01     -1.103e+00      3.074e-02
##
## Degrees of Freedom: 2887 Total (i.e. Null);  2869 Residual
```

```
## Null Deviance:       3577
## Residual Deviance: 3254  AIC: 3292
```

```
contrasts(final_data$binary_recovery_time)
```

```
##      high
## low     0
## high    1
```

```
#We first consider the simple classifier with a cut-off of 0.5 and evaluate its performance on the test

test.pred.prob = predict(model.glm$finalModel, newdata = test_x_df, type = "response")
test.pred = rep("low", length(test.pred.prob))
test.pred[test.pred.prob > 0.5] = "high"

confusionMatrix = confusionMatrix(data = as.factor(test.pred),
                reference = test_y,
                positive = "high")
confusionMatrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction low high
##       low   54   42
##       high 169  456
##
##                Accuracy : 0.7074
##                  95% CI : (0.6726, 0.7403)
##     No Information Rate : 0.6907
##     P-Value [Acc > NIR] : 0.1772
##
##                   Kappa : 0.1873
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9157
##             Specificity : 0.2422
##          Pos Pred Value : 0.7296
##          Neg Pred Value : 0.5625
##              Prevalence : 0.6907
##          Detection Rate : 0.6325
##    Detection Prevalence : 0.8669
##       Balanced Accuracy : 0.5789
##
##        'Positive' Class : high
##
```

```
#Testing error rate is 0.2926491
1 - confusionMatrix$overall["Accuracy"]
```

```
##  Accuracy
## 0.2926491
```

```
#Training error rate is 0.2939751
train.pred.prob = predict(model.glm$finalModel, newdata = train_x_df, type = "response")
train.pred.prob[train.pred.prob > 0.5] = "high"
```

```r
train.pred.prob[train.pred.prob < 0.5] = "low"
table(train_y, train.pred.prob)
```

```
##        train.pred.prob
## train_y high  low
##    low   682  214
##    high 1825  167
```

```r
mean(train.pred.prob != train_y)
```

```
## [1] 0.2939751
```