Logistic regression classification Vignette

Oscar Trevizo

January 2023

Contents

1 Functions **1**

2 Simulate the data **4**

2.1 Build train and test sets 4

3 Boxplot and histogram **4**

4 Logistic regression (Generalized Linear Models GLM) **6**

4.1 Fit the model 6

4.2 Predicit 7

4.3 Confusion matrix 7

4.4 Prediction metrics 8

5 Linear Discriminant Analysis (LDA) **8**

5.1 Fit the model 8

5.2 Predict 9

5.3 Confusion matrix 9

5.4 LDA prediction metrics 10

6 References **10**

This code is based on lessons from Harvard Statistical Learning class [see references]. I expanded the material with my own scripts, notes and R documentation and I plan to continue adding examples overtime.

This vignette focuses on logistic regression based on the Generalized Linear Models from the “stats” library.

1 Functions

Adapted from R functions shared by faculty in Harvard data science class (2021). See references at the bottom of this notebook.

```r
###
#
# prediction.metrics function -- to return a list with all the metrics values
#
# Based on R functions shared by faculty in Harvard data science class (2021). See references.
#
# Input: truth and predicted lists.
#
# Returns a list with:
# [1] OBS = Observations or truth cases
# [2] Accuracy. ACC = sum(truth == predicted) * 100/length(truth)
# [3] Sensitivity. TPR True Positive Rate = TP/(TP + FN) = TP/P
# [4] Specificity. TNR True Negative Rate = TN/(FP + TN) = TN/N
# [5] Precision. Positive Predictive Value. PPV = TP/(TP + FP)
# [6] Negative Predictive Value. NPV = TN/(TN + FN)
# [7] False Discovery Rate. FDR = FP/(TP + FP)
# [8] False Positive Rate. FPR = FP/(FP + TN) = FP/N
# [9] True Positives. TP = sum(truth == 1 & predicted == 1)
# [10] True Negatives. TN = sum(truth == 0 & predicted == 0)
# [11] False Positives. FP = sum(truth == 0 & predicted == 1)
# [12] False Negatives. FN = sum(truth == 1 & predicted == 0)
# [13] Positives. P = TP + FN  # total number positives in the truth data
# [14] Negatives. N = FP + TN  # total number of negatives
#
prediction.metrics = function(truth, predicted) {
    # same length:
    if (length(truth) != length(predicted)) {
        stop("truth and predicted must be same length!")
    }
    # check for missing values (we are going to compute metrics on non-missing
    # values only)
    bKeep = !is.na(truth) & !is.na(predicted)
    predicted = predicted[bKeep]
    truth = truth[bKeep]
    # only 0 and 1:
    if (sum(truth %in% c(0, 1)) + sum(predicted %in% c(0, 1)) != 2 * length(truth)) {
        stop("only zeroes and ones are allowed!")
    }
    # how predictions align against known training/testing outcomes: TP/FP=
    # true/false positives, TN/FN=true/false negatives
    TP = sum(truth == 1 & predicted == 1)
    TN = sum(truth == 0 & predicted == 0)
    FP = sum(truth == 0 & predicted == 1)
    FN = sum(truth == 1 & predicted == 0)
    P = TP + FN  # total number of positives in the truth data
    N = FP + TN  # total number of negatives
    # Add the following output to return (OAT 11/9/2021)
    OBS = length(truth)
    ACC = sum(truth == predicted)/length(truth)
    TPR = TP/P
    TNR = TN/N
    PPV = TP/(TP + FP)
    NPV = TN/(TN + FN)
```

```
    FDR = FP/(TP + FP)
    FPR = FP/N

    # Returned a named list
    output <- list(OBS=OBS, ACC=ACC, TPR=TPR, TNR=TNR, PPV=PPV,
                   NPV=NPV, FDR=FDR, FPR=FPR, TP=TP,
                   TN=TN, FP=FP, FN=FN, P=P, N=N)
    return(output)
}

print.the.metrics = function(metrics){
  cat(' OBS = ', metrics$OBS, '..................number of observations')
  cat('\n ACC = ', metrics$ACC, '..................Accuracy')
  cat('\n TPR = ', metrics$TPR, '..................True Positive Rate')
  cat('\n TNR = ', metrics$TNR, '..................True Negative Rate')
  cat('\n PPV = ', metrics$PPV, '..................Positive Predictive Value (Precision)')
  cat('\n NPV = ', metrics$NPV, '..................Negative Predictive Value')
  cat('\n FDR = ', metrics$FDR, '..................False Discover Rate')
  cat('\n FPR = ', metrics$FPR, '..................False Positive Rate')
  cat('\n TP  = ', metrics$FP, '..................True Positives')
  cat('\n TN  = ', metrics$TN, '..................True Negatives')
  cat('\n FP  = ', metrics$TN, '..................False Positives')
  cat('\n FN  = ', metrics$FN, '..................False Negatives')
  cat('\n P   = ', metrics$P, '..................Positives')
  cat('\n N   = ', metrics$N, '..................Negatives')


}
# Logistic regression
lgr.pred.ftn = function(formula, df.train, df.test){
  glm.fit <- glm(formula, data = df.train, family = binomial)
  glm.probs <- predict(glm.fit, newdata = df.test, type = "response")
  glm.pred <- rep(0, dim(df.test)[1])
  glm.pred[glm.probs>0.5]=1
  return(glm.pred)
}

# Linear Discriminant Analysis (LDA)
lda.pred.ftn = function(formula, df.train, df.test){
  lda.fit <- lda(formula, data = df.train)
  lda.pred <- predict(lda.fit, df.test)
  lda.class <- lda.pred$class
  return(lda.class)
}

# Quadratic Discriminant Analysis (QDA)
qda.pred.ftn = function(formula, df.train, df.test){
  qda.fit <- qda(formula, data = df.train)
  qda.pred <- predict(qda.fit, df.test)
  qda.class <- qda.pred$class
  return(qda.class)
}
```

2 Simulate the data

Play with two sets of Normally distributed sets of data with different means. We can change the number of samples and we can move the means around.

```
# From Harvard data science class (see references at the end of this notebook)

set.seed(11)

N = 1000
mu = 4

# Our measuring variable is continuous, numeric...
# ...it has two Normal distribution waves
x <- c(rnorm(N), rnorm(N, mean=mu))

# Our outcome is categorical, A and B xxxx times each
# ...the idea is to match A and B to a number x
y <- rep(c("A", "B"), each=N)

# Make a data.frame with 1 and 0 values for Y
df <- data.frame(Y=ifelse(y=="A",0, 1), X=x)
```

2.1 Build train and test sets

```
set.seed(12321)

# Get 2:1 random sample ratio for Train:Test sets
sampleTrain <- sample(c(TRUE,FALSE,TRUE), nrow(df), rep=TRUE)
df.train <- df[sampleTrain,]
df.test <- df[!sampleTrain,]
```

3 Boxplot and histogram

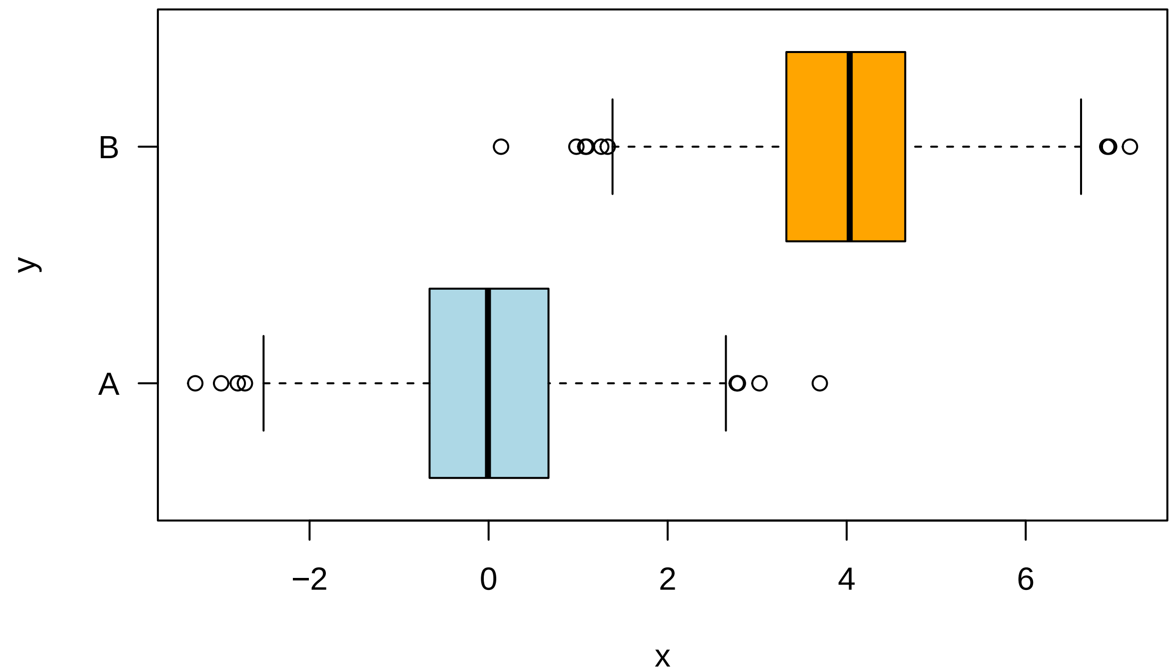
In a boxplot, we want to have the categorical variable in the horizontal axis.

That is why we see a formula x~y below.

The chart gives us more information if we plot it horizontally in this case.

The histogram adds information to visualize the behavior relationship between the outcome, categorical valuable, and predictor, numeric variable.

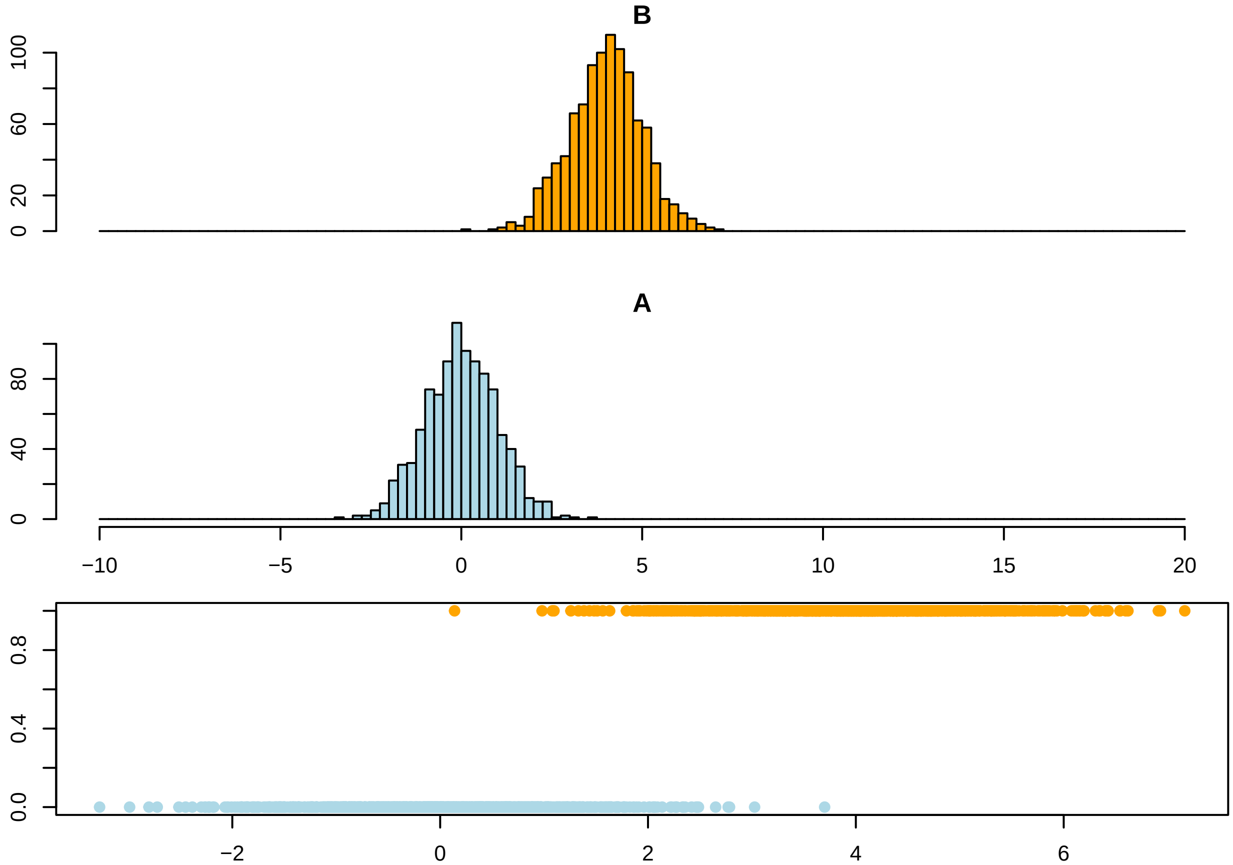
```
# From Harvard data science class (see references)
boxplot(x~y, col=c("lightblue","orange"), horizontal=T, las=1)
```




```r
# Now place a histogram on top of another histogram

oldpar <- par(mfrow=c(3, 1), mar=c(2,2,1,1))

breaks <- seq(-10, 20, by=0.25)

# Histogram for 'B'
hist(x[y=="B"], breaks=breaks, col='orange', main="B", xaxt='n')

# Histogram for 'A'
hist(x[y=="A"], breaks=breaks, col='lightblue', main="A")

plot(x, ifelse(y=="A", 0,1), breaks=breaks, col=ifelse(y=="A", "lightblue","orange"), pch=19)
```

```
## Warning in plot.window(...): "breaks" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "breaks" is not a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "breaks" is not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "breaks" is not a
## graphical parameter

## Warning in box(...): "breaks" is not a graphical parameter
```

```
## Warning in title(...): "breaks" is not a graphical parameter
```




```
par(oldpar)
```

4 Logistic regression (Generalized Linear Models GLM)

Needs library{stats}

4.1 Fit the model

```
##
#
# GLA from library{stats}
#
##
glm.fit <- glm(Y~X, data=df.train, family = binomial)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = Y ~ X, family = binomial, data = df.train)
```

```
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.5991  -0.0311  -0.0007   0.0262   3.7959
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -7.7363     0.7380  -10.48   <2e-16 ***
## X             3.8413     0.3516   10.92   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1837.55  on 1325  degrees of freedom
## Residual deviance:  167.88  on 1324  degrees of freedom
## AIC: 171.88
##
## Number of Fisher Scoring iterations: 9
```

4.2 Predicit

- Make predictions using the test dataset.

```r
##
#
# predict{stats}
#
# Continued based on ISLR 4.6.2 p.156-158
#

# Get the probability that the device is located in Room #3
glm.probs <- predict(glm.fit, newdata = df.test, type = "response")

# Per ISLR, we need contrasts() and use the variable as a logical vector.
# Note, I already had converted Room as.factor
# contrasts(df$Y)

# Initiated glm.pred vector
glm.pred = rep(0, dim(df.test)[1])
# Adjust the probability. Here is something one can play with after looking at the 'table' that follows
glm.pred[glm.probs>0.5]=1
```

4.3 Confusion matrix

```r
##
#
# Continued based on ISLR 4.6.2 p.156-158
#
# Numbers outside of the diagonal are either false positives or false negatives
#
```

```
table(glm.pred, df.test$Y)
```

```
##
## glm.pred   0   1
##        0 314  11
##        1   8 341
```

```
mean(glm.pred == df.test$Y)
```

```
## [1] 0.9718101
```

4.4 Prediction metrics

- Now I will use the function calculate accuracy, sensitivity, and specificity

```
##
#
# Based on functions from above
#

lgr.pred <- lgr.pred.ftn(Y~X, df.train, df.test)

lgr.metrics <- prediction.metrics(df.test$Y, lgr.pred)

print.the.metrics(lgr.metrics)
```

```
##  OBS =  674 ..................number of observations
##  ACC =  0.9718101 .................Accuracy
##  TPR =  0.96875 .................True Positive Rate
##  TNR =  0.9751553 .................True Negative Rate
##  PPV =  0.9770774 .................Positive Predictive Value (Precision)
##  NPV =  0.9661538 .................Negative Predictive Value
##  FDR =  0.02292264 .................False Discover Rate
##  FPR =  0.02484472 .................False Positive Rate
##  TP  =  8 ..................True Positives
##  TN  =  314 ..................True Negatives
##  FP  =  314 ..................False Positives
##  FN  =  11 ..................False Negatives
##  P   =  352 ..................Positives
##  N   =  322 ..................Negatives
```

5 Linear Discriminant Analysis (LDA)

Needs library{MASS}

5.1 Fit the model

```
##
#
# LDA from library{MASS}
#
##
lda.fit <- lda(Y~X, data = df.train)
summary(lda.fit)
```

```
##         Length Class  Mode
## prior   2      -none- numeric
## counts  2      -none- numeric
## means   2      -none- numeric
## scaling 1      -none- numeric
## lev     2      -none- character
## svd     1      -none- numeric
## N       1      -none- numeric
## call    3      -none- call
## terms   3      terms  call
## xlevels 0      -none- list
```

5.2 Predict

```
lda.pred <- predict(lda.fit, df.test)

names(lda.pred)
```

```
## [1] "class"     "posterior" "x"
```

5.3 Confusion matrix

```
##
#
# Continued based on ISLR 4.6.3 p.161-162
#
#

lda.class <- lda.pred$class

table(lda.class, df.test$Y)
```

```
##
## lda.class   0   1
##         0 314  12
##         1   8 340
```

```
mean(lda.class == df.test$Y)
```

```
## [1] 0.9703264
```

- The confusion matrix is based on the test set.
- The confusion matrix indicates the number of observations correctly predicted not to be in Y.
- And it indicated the number of observations correctly predicted to be in Y.
- The `mean()` function calculates the diagonals over the total.
- These results parallel those from linear regression in Problem 1.

5.4 LDA prediction metrics

- Now I will use the function from from above

```
##
#
# Based on functions from above
#

lda.pred <- lda.pred.ftn(Y~X, df.train, df.test)

lda.metrics <- prediction.metrics(df.test$Y, lda.pred)

print.the.metrics(lda.metrics)
```

```
##  OBS =  674 ...................number of observations
##  ACC =  0.9703264 ..................Accuracy
##  TPR =  0.9659091 ..................True Positive Rate
##  TNR =  0.9751553 ..................True Negative Rate
##  PPV =  0.9770115 ..................Positive Predictive Value (Precision)
##  NPV =  0.9631902 ..................Negative Predictive Value
##  FDR =  0.02298851 ..................False Discover Rate
##  FPR =  0.02484472 ..................False Positive Rate
##  TP  =  8 ..................True Positives
##  TN  =  314 ..................True Negatives
##  FP  =  314 ..................False Positives
##  FN  =  12 ..................False Negatives
##  P   =  352 ..................Positives
##  N   =  322 ..................Negatives
```

6 References

- Harvard "Elements of Statistical Learning" (2021) taught by professors Dr. Sivachenko, Dr. Farutin
- Book "An Introduction to Statistical Learning with Applications in R" (ISLR) by Gareth James et al