Lab_05R

36-290 – Statistical Research Methodology

Week 5 Thursday – Fall 2021

Preliminaries

Goal

Today you will create a logistic regression model for classifying stars vs. quasars and you will assess your performance by computing a test-set misclassification rate.

Data

We'll begin by importing data on stars and quasars:

```
file.path = "https://raw.githubusercontent.com/pefreeman/36-290/master/EXAMPLE_DATASETS/STAR_QUASAR/Star_Quasar.Rda
ta"
load(url(file.path))
rm(file.path)
```

The data frame `df` has 8 measurements each for 500 quasars (faraway active galactic nuclei that look like stars) and for 500 stars. The first five columns are u-, g-, r-, i-, and z-band magnitudes, the sixth column is redshift (high for quasars, approximately zero for stars), the seventh column is redshift error, and the eighth column is a factor variable that denotes the class (`QSO` or `STAR`).

The goal is to see if you can correctly classify each object. We will set up a predictor data frame with four colors and a magnitude, and a response vector that is a factor variable with two levels ("QSO" and "STAR"). (Including redshift as a predictor would be cheating: the redshift is how we know for sure whether the objects are quasars or stars in the first place!)

```
col.ug = df$u.mag - df$g.mag
col.gr = df$g.mag - df$r.mag
col.ri = df$r.mag - df$i.mag
col.iz = df$i.mag - df$z.mag
mag.r  = df$r.mag
predictors = data.frame(col.ug,col.gr,col.ri,col.iz,mag.r)
response   = df$class
```

Questions

Question 1

Split the data into training and test sets. Then use `ggpairs()` to display the (full) predictor space, while using the argument `mapping=aes(color=response)` to use separate colors for quasars and for stars. Based on what you see, do you expect a clean separation between quasars and stars? (In other words, do you expect a low misclassification rate?)

```
library(GGally)
```

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

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

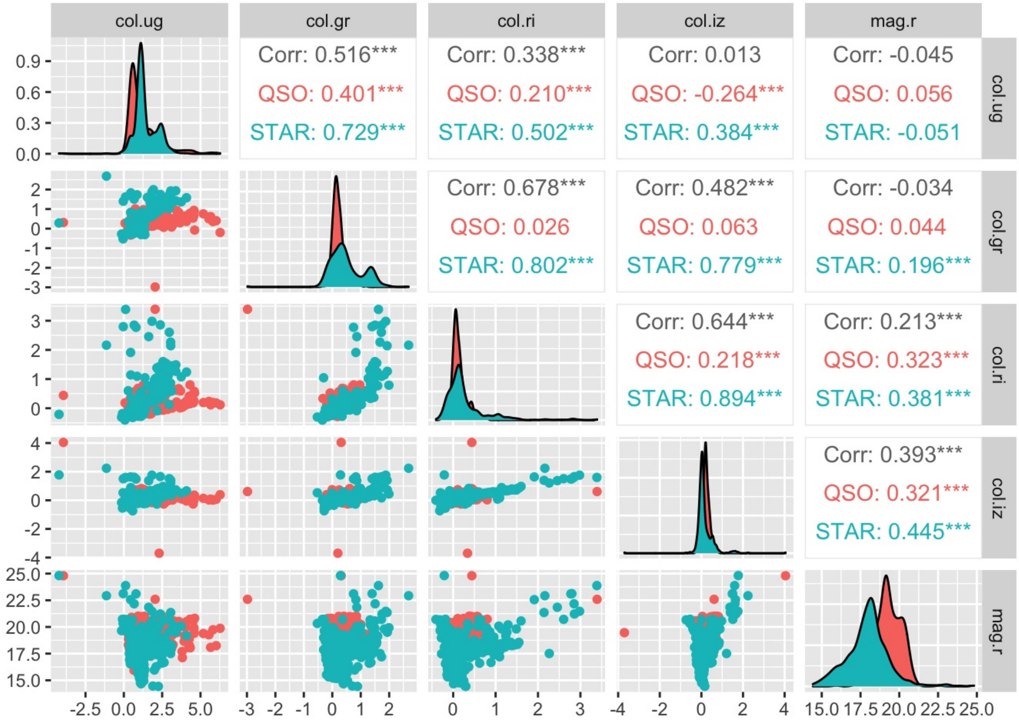
```
library(ggplot2)

set.seed(100)
fraction=.7
sp = sample(nrow(predictors), round(fraction*nrow(predictors)))
pred.train = predictors[sp ,]
pred.test = predictors[-sp ,]

respdf = data.frame(response)
set.seed(100)
fraction=.7
sr = sample(length(response), round(fraction*length(response)))
resp.train = respdf[sr ,]
resp.test = respdf[-sr ,]



ggpairs(predictors,progress=FALSE, mapping=aes(color=response))
```




```
I don't expect too clear of a separation because there does seem to be a bit of overlap to some extent.
```

Question 2

Using code on pages 156-158 of ISLR, carry out a logistic regression analysis of the star-quasar data, and display both the misclassification rate and a table of predictions versus test-set responses (i.e., display the confusion matrix). (Note: it may help you to use the `contrasts()` function to determine the mapping from the factor levels to actual numbers. See the top of page 158.) Challenges: can you create a vector of predicted factors in one line using the `ifelse()` function (which is *not* what ISLR does), and can use compute the misclassification rate using just one logical comparison?

```
glm.fit = glm(resp.train~.,data=pred.train,family=binomial)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = resp.train ~ ., family = binomial, data = pred.train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -6.1472  -0.6232   0.0719   0.5116   5.3629
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  25.1067     2.4643  10.188  < 2e-16 ***
## col.ug       -0.7554     0.1810  -4.173 3.01e-05 ***
## col.gr        2.1697     0.5454   3.978 6.94e-05 ***
## col.ri        4.1750     0.6506   6.417 1.39e-10 ***
## col.iz       -5.2994     0.8679  -6.106 1.02e-09 ***
## mag.r        -1.3313     0.1318 -10.098  < 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: 970.38  on 699  degrees of freedom
## Residual deviance: 556.61  on 694  degrees of freedom
## AIC: 568.61
##
## Number of Fisher Scoring iterations: 6
```

```
coef(glm.fit)
```

```
## (Intercept)      col.ug      col.gr      col.ri      col.iz       mag.r
##  25.1067241  -0.7554035   2.1697027   4.1750092  -5.2993924  -1.3312903
```

```
glm.probs=predict(glm.fit,type="response")

contrasts (response)
```

```
##      STAR
## QSO     0
## STAR    1
```

```
#create a vector of class predictions based on whether the predicted probability of STAR is greater than or less th
an 0.5:

glm.pred=rep("QSO", 1000)
glm.pred[glm.probs >.5]="STAR"


tab = table(glm.pred,response)
tab
```

```
##         response
## glm.pred QSO STAR
##     QSO  268  258
##     STAR 232  242
```

```
(327+216)/1000
```

```
## [1] 0.543
```

```
mean(glm.pred==response )
```

```
## [1] 0.51
```

```
logistic regression correctly predicted the movement of the market 54.3 % of the time

USING IFELSE:


for (ii in 1:length(glm.probs)) {
  ifelse(glm.probs[ii] > 0.5, glm.pred[ii] =="STAR", glm.pred[ii] == "QSO")

}
table(glm.pred,response)





glm.probs=predict(glm.fit,newdata=test,type="response")
glm.pred = rep(NA, length(glm.probs))
for (ii in 1:length(glm.prob)) {
  if(glm.prob[ii] > 0.5) {
    glm.pred[ii] ="STAR"
  } else {
    glm.pred[ii] = "QSO"
  }

}
```

Question 3

Compute the sensitivity and specificity of logistic regression using definitions on this web page (https://en.wikipedia.org/wiki/Confusion_matrix). There can be some ambiguity regarding tables: assume that predicting that a QSO is a QSO is a “true positive” here, as opposed to predicting a star is a star (which is a “true negative”).

Don’t hard-code numbers! If you saved your confusion matrix above to the variable `tab`, then, e.g.,

```
TP = tab[1,1]
FP = tab[2,1]
```

etc. Map your table to `TP`, `FP`, `TN`, and `FN`, and use these to compute sensitivity and specificity, and then define each in words. In a perfect world, what would the sum of sensitivity and specificity be?

```
TP = tab[1,1]

FP = tab[2,1]

TN = tab[2,2]

FN = tab[1,2]


TP
```

```
## [1] 268
```

```
FP
```

```
## [1] 232
```

```
TN
```

```
## [1] 242
```

```
FN
```

```
## [1] 258
```

```
TPR = TP/(TP+FN)
TNR = TN/(TN+FP)

TPR
```

```
## [1] 0.5095057
```

```
TNR
```

```
## [1] 0.5105485
```

```
TP=327
FP=173
TN=216
FN=284

Sensitivity= True Positive Rate (TPR)= TP/(TP+FN)=0.5351882
Specificity= True Negative Rate (TNR)= TN/(TN+FP)=0.5552699


in an ideal the sum would be 2
```

Question 4

An astronomer might be more interested to know what proportion of objects that are predicted to be quasars actually are quasars. Compute this quantity and determine from the confusion matrix wikipedia page what this quantity is called.

```
ACC = (TP+TN)/(TP+TN+FP+FN)
ACC
```

```
## [1] 0.51
```

```


proportion of correct classifications is called accuracy
and ACC=0.51
```

Question 5

While we didn't discuss this explicitly in the notes, we can attempt to visualize the distributions of the predicted binomial probabilities $\hat{p}$ versus class. Do that below. I'm going to leave it as ambiguous about how exactly you might do this.

```
library(car)
```

```
## Loading required package: carData
```

```
newdf = data.frame(predictors, response)
newdf$reponse <- with(newdf, ifelse(response == "QSO", 0, ifelse(response=="STAR", 1, NA)))


set.seed(100)
fraction=.7
trainingIndex = sample(nrow(newdf), round(fraction*nrow(newdf)))
newdfpred.train = newdf[trainingIndex ,]
newdfpred.test = newdf[-trainingIndex ,]


#fit logit model
#fitLogit <- glm(newdfpred.train~.,data=newdfpred.train, family=binomial(link="logit"))

#predLogit <- predict(fitLogit, newdata=newdfpred.test, type="response", se.fit=TRUE)


#forplot= data.frame(p=class,outcome=ifelse(predLogit$fit>0.5,1,0))

#gg = ggplot(forplot, aes(x=p, y= outcome)) + geom_point() +
  #stat_smooth(method="glm", method.args=list(family="binomial"), se=FALSE)

#gg
```

```
weird error in above chunk prevented R from knitting
```

Question 6

You should be sufficiently comfortable with setting up basic analyses that you are going to do something different here: you are going to perform an analysis using a method not described in class. Linear discriminant analysis basically assumes that the predictors for the `QSO` class and for the `STAR` class are each sampled from a multivariate (specifically p-dimensional) normal distribution; the means are different for each class, but the "widths" of the distributions (as encoded in a covariance matrix) are the same. For each test datum, you can determine the estimated probability density for quasars, and the estimated probability density for stars; if the former is larger, we predict the datum is a quasar, and if the latter is larger, we predict the datum is a star. Those details aside, go to pages 160 and 161 of ISLR and implement an LDA analysis. Compute the misclassification rate and display the confusion matrix. Does LDA do better than logistic regression? Does it do worse?

```
library(MASS)
lda.fit = lda(resp.train~.,data=pred.train)
lda.fit
```

```
## Call:
## lda(resp.train ~ ., data = pred.train)
##
## Prior probabilities of groups:
##       QSO      STAR
## 0.4971429 0.5028571
##
## Group means:
##        col.ug    col.gr    col.ri    col.iz    mag.r
## QSO  1.199928 0.2465992 0.1354409 0.2168516 19.34465
## STAR 1.380844 0.5040136 0.2712696 0.1419613 17.85286
##
## Coefficients of linear discriminants:
##               LD1
## col.ug -0.4132095
## col.gr  1.4426281
## col.ri  1.2462419
## col.iz -1.0601386
## mag.r  -0.7967577
```

```
lda.pred=predict(lda.fit, data=resp.test)
names(lda.pred)
```

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

```
lda.class=lda.pred$class
lda.class
```

```
##   [1] STAR QSO  QSO  STAR STAR QSO  STAR QSO  STAR STAR QSO  QSO  QSO  STAR QSO
##  [16] QSO  QSO  QSO  QSO  STAR STAR QSO  QSO  QSO  STAR QSO  QSO  STAR QSO  STAR
##  [31] STAR STAR QSO  QSO  QSO  QSO  QSO  STAR STAR QSO  STAR QSO  STAR QSO  QSO
##  [46] QSO  QSO  STAR STAR STAR STAR QSO  STAR QSO  QSO  QSO  QSO  STAR STAR QSO
##  [61] QSO  STAR QSO  QSO  STAR STAR QSO  STAR STAR QSO  STAR QSO  QSO  STAR STAR
##  [76] QSO  QSO  QSO  QSO  STAR QSO  QSO  QSO  QSO  STAR QSO  STAR STAR QSO  STAR
##  [91] STAR STAR STAR STAR STAR STAR QSO  STAR QSO  STAR QSO  QSO  QSO  QSO  STAR
## [106] STAR STAR STAR QSO  QSO  STAR QSO  QSO  QSO  STAR QSO  QSO  QSO  QSO  STAR
## [121] QSO  QSO  STAR QSO  QSO  STAR STAR STAR QSO  QSO  STAR STAR QSO  STAR STAR
## [136] STAR QSO  STAR STAR QSO  STAR QSO  QSO  QSO  QSO  STAR QSO  STAR STAR QSO
## [151] QSO  STAR QSO  QSO  QSO  STAR STAR QSO  QSO  QSO  QSO  QSO  STAR STAR QSO
## [166] STAR STAR STAR STAR STAR STAR QSO  QSO  STAR STAR QSO  QSO  QSO  STAR STAR
## [181] QSO  QSO  STAR STAR QSO  QSO  STAR QSO  QSO  QSO  STAR STAR STAR STAR STAR
## [196] STAR QSO  STAR QSO  QSO  QSO  STAR QSO  QSO  STAR QSO  QSO  STAR QSO  STAR
## [211] QSO  STAR STAR QSO  QSO  STAR STAR QSO  STAR STAR STAR QSO  QSO  STAR STAR
## [226] QSO  QSO  STAR QSO  QSO  QSO  STAR QSO  QSO  QSO  QSO  STAR STAR STAR QSO
## [241] STAR QSO  STAR QSO  STAR STAR STAR QSO  STAR QSO  QSO  QSO  QSO  STAR STAR
## [256] QSO  QSO  QSO  STAR QSO  QSO  STAR QSO  STAR STAR QSO  STAR QSO  QSO  QSO
## [271] QSO  STAR STAR STAR STAR STAR QSO  QSO  QSO  QSO  STAR STAR QSO  STAR STAR
## [286] QSO  STAR QSO  STAR QSO  QSO  STAR QSO  QSO  STAR QSO  STAR QSO  QSO  STAR
## [301] STAR STAR STAR QSO  QSO  QSO  STAR QSO  STAR STAR QSO  STAR QSO  STAR QSO
## [316] QSO  QSO  QSO  QSO  QSO  QSO  QSO  QSO  STAR STAR STAR QSO  QSO  STAR QSO
## [331] QSO  QSO  QSO  QSO  STAR QSO  STAR STAR STAR STAR QSO  STAR STAR STAR STAR
## [346] QSO  STAR STAR STAR STAR STAR QSO  QSO  QSO  STAR STAR QSO  QSO  QSO  STAR
## [361] QSO  STAR QSO  STAR STAR QSO  QSO  QSO  QSO  STAR QSO  STAR QSO  QSO  QSO
## [376] QSO  STAR STAR QSO  QSO  STAR STAR STAR STAR QSO  QSO  QSO  QSO  STAR STAR
## [391] STAR STAR QSO  QSO  QSO  QSO  QSO  QSO  STAR QSO  STAR QSO  QSO  QSO  QSO
## [406] QSO  QSO  STAR QSO  QSO  STAR STAR QSO  STAR STAR QSO  STAR STAR QSO  STAR
## [421] QSO  STAR QSO  STAR QSO  STAR QSO  QSO  QSO  STAR QSO  STAR QSO  QSO  QSO
## [436] QSO  STAR QSO  QSO  STAR QSO  QSO  QSO  QSO  STAR QSO  QSO  QSO  QSO  QSO
## [451] STAR QSO  STAR QSO  STAR QSO  QSO  STAR QSO  STAR STAR QSO  STAR STAR STAR
## [466] STAR QSO  STAR STAR QSO  QSO  STAR QSO  QSO  STAR QSO  STAR QSO  QSO  STAR
## [481] QSO  QSO  QSO  QSO  QSO  STAR QSO  STAR QSO  QSO  STAR QSO  QSO  QSO  QSO
## [496] QSO  QSO  QSO  STAR QSO  STAR STAR STAR STAR QSO  QSO  QSO  STAR QSO  STAR
## [511] QSO  STAR STAR STAR QSO  STAR QSO  QSO  QSO  QSO  STAR QSO  STAR STAR QSO
## [526] STAR STAR STAR STAR QSO  STAR STAR QSO  QSO  QSO  STAR QSO  QSO  QSO  QSO
## [541] QSO  STAR QSO  STAR QSO  QSO  STAR QSO  STAR STAR QSO  QSO  QSO  QSO  STAR
## [556] QSO  QSO  STAR STAR QSO  STAR STAR QSO  QSO  QSO  STAR QSO  QSO  STAR STAR
## [571] QSO  QSO  QSO  QSO  STAR QSO  QSO  STAR STAR STAR STAR STAR STAR QSO  QSO
## [586] STAR QSO  QSO  QSO  QSO  STAR STAR STAR QSO  STAR QSO  STAR STAR STAR STAR
## [601] QSO  QSO  STAR QSO  QSO  QSO  QSO  STAR STAR QSO  STAR STAR STAR QSO  STAR
## [616] QSO  QSO  STAR STAR QSO  QSO  QSO  QSO  QSO  STAR STAR QSO  QSO  STAR STAR
## [631] QSO  STAR STAR QSO  STAR QSO  QSO  QSO  QSO  STAR STAR QSO  QSO  STAR STAR
## [646] STAR QSO  QSO  STAR QSO  QSO  QSO  QSO  STAR QSO  STAR STAR STAR STAR QSO
## [661] QSO  STAR QSO  QSO  STAR STAR STAR QSO  STAR QSO  STAR QSO  QSO  STAR STAR
## [676] STAR QSO  STAR QSO  STAR STAR STAR STAR QSO  STAR QSO  QSO  STAR QSO  QSO
## [691] STAR STAR QSO  STAR QSO  QSO  STAR QSO  STAR QSO
## Levels: QSO STAR
```

```
#table(lda.class,resp.test)
#error: all arguments must have same length?
```

Loading [MathJax]/jax/output/HTML-CSS/jax.js