Stat 432 Homework 8

Giang Le

Assigned: Oct 11, 2021; Due: 11:59 PM CT, Oct 19, 2021

Contents

Question 1: Logistic Regression

We will use the Cleveland clinic heart disease dataset, which has been used in the lecture note. You can directly download the data from our course website. The goal is to predict the label `num > 0`. The following code prepares the data. I removed `ca` and `thal` because they contain missing values.

```
heart = read.csv("processed_cleveland.csv")
heart$Y = as.factor(heart$num > 0)
heart = subset(heart, select = -c(num, ca, thal))
```

We are going to perform three models:

- A logistic regression
- A logistic regression with Ridge penalty

And we will evaluate them using two different criteria:

- Classification error
- Area under the ROC curve

Also, please note that, to keep things simpler, we will not use cross-validation for this question. Instead, all the evaluations will be just on the training dataset. We are of course at the risk of over-fitting, but Part III will address that issue. In addition, since no cross-validation is needed, you should be using the `glmnet()` function instead of `cv.glmnet()`. The syntax of this function is almost identical to its cross-validation version, except that you will not have the cross-validation feature to help you select the best λ. However, the function will still produce all the coefficients for each λ value. If you need more details, please see the documentation provided at CRAN.

Part I [40 Points]

Complete the following questions for logistic regression:

- Fit logistic regression to the heart data and report the most significant variable.
- Using 0.5 as the cut-off value of predicted probability, produce the confusion table of the training data. What is the classification error associated with this model?
- What is the sensitivity and specificity of this model? Choose a new cut-off value that would give a higher sensitivity, and report the confusion table and sensitivity associated with this new cut-off value.
- Produce the ROC curve plot associated with your logistic regression and report the AUC.

First, I fit a logistic regression model to the filtered heart data, using all predictor variables.

```
logistic.fit <- glm(Y ~ ., data = heart, family = binomial)
summary(logistic.fit)
```

```
##
## Call:
## glm(formula = Y ~ ., family = binomial, data = heart)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3192  -0.6484  -0.2130   0.5886   2.6819
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -7.191858   2.510711  -2.864  0.00418 **
## age          0.023549   0.020797   1.132  0.25749
## sex          1.955728   0.400915   4.878 1.07e-06 ***
## cp           0.810770   0.179562   4.515 6.32e-06 ***
## trestbps     0.019365   0.009724   1.992  0.04643 *
## chol         0.005369   0.003263   1.646  0.09982 .
## fbs         -0.215480   0.445551  -0.484  0.62865
## restecg      0.201612   0.163232   1.235  0.21678
## thalach     -0.023967   0.009166  -2.615  0.00893 **
## exang        0.989378   0.362936   2.726  0.00641 **
## oldpeak      0.496534   0.183807   2.701  0.00691 **
## slope        0.320861   0.320731   1.000  0.31711
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 417.98  on 302  degrees of freedom
## Residual deviance: 252.43  on 291  degrees of freedom
## AIC: 276.43
##
## Number of Fisher Scoring iterations: 5
```

The variable with the lowest p-value is sex, and it is the most significant variable.

Below I produce the confusion table of the training data. Before that, I create predictions based on the training data.

```
pred = predict(logistic.fit, newdata = heart, type = "response")
# Confusion table
table(pred > 0.5, heart$Y)
```

```
##
##         FALSE TRUE
##   FALSE   138   34
##   TRUE     26  105
```

The error rate of this model is (34+26)/(138+34+26+105) approximately 0.198 or near 20%.

Sensitivity (also called "Recall") is the defined as the true positive rate. According to the confusion table above, the sensitivity is TP/(TP + FN) = 105/(105 + 34) ~ 0.755

Specificity is the defined as the true negative rate. The specificity is TN/(TN + FP) = 138/(138+26) ~ 0.841

We can choose a lower cut off to have a higher sensitivity. Let's choose 0.4 as a cut off.

```
# New confusion table with a higher cutoff
table(pred > 0.4, heart$Y)
```

```
##
##         FALSE TRUE
##   FALSE   126   27
##   TRUE     38  112
```

Here the sensitivity is TP/(TP + FN) = 112/(112+7) = 0.941

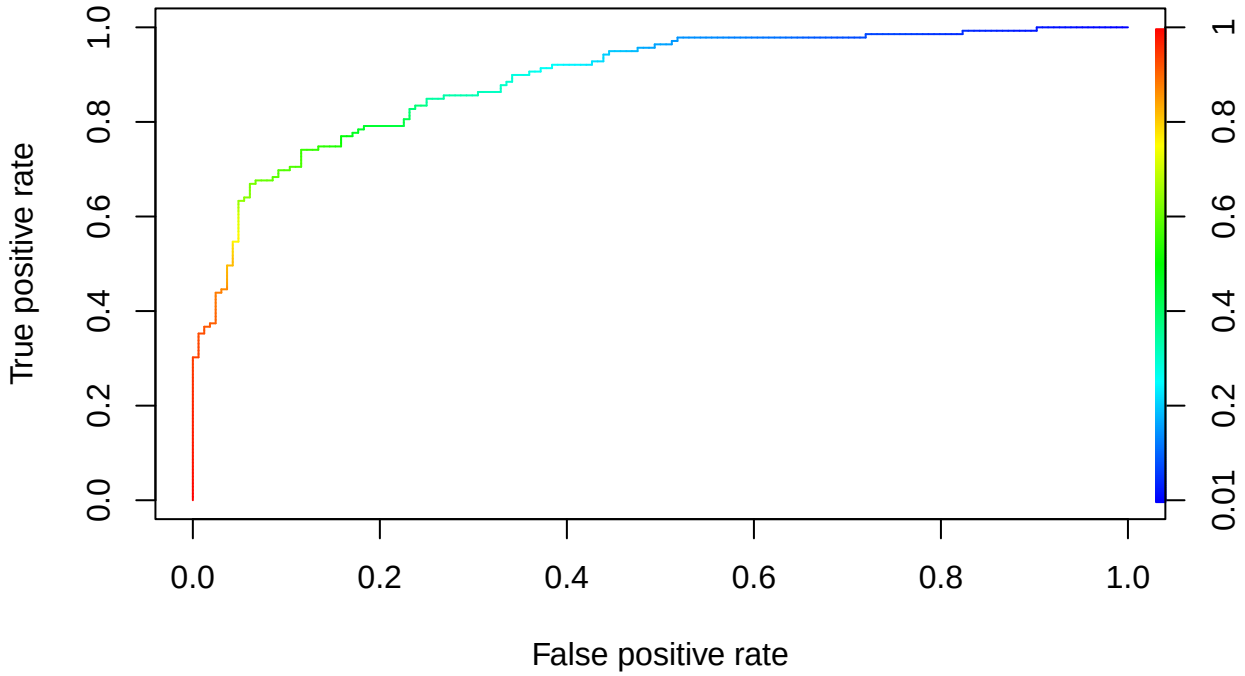
Produce the ROC curve plot associated with your logistic regression and report the AUC. I installled the package ROCR and produced the ROC plot, which is the plot between the true positive rate and the false positive rate.

```
install.packages("ROCR", repos = "http://cran.us.r-project.org")
```

```
##
## The downloaded binary packages are in
##  /var/folders/9c/3_mgdyf12z7dvb8rt4d60nt80000gn/T//Rtmp3NBcie/downloaded_packages
```

```
library(ROCR)
roc <- prediction(pred, heart$Y)

# calculates the ROC curve
perf <- performance(roc,"tpr","fpr")
plot(perf,colorize=TRUE)
```



I calculate the area under the curve (AUC) below.

```
performance(roc, measure = "auc")@y.values[[1]]
```

```
## [1] 0.8893666
```

Part II [40 Points]

Complete the following questions for logistic regression with Ridge penalty :

- Use the `glmnet()` function to produce a set of coefficients across many λ values.

- Since we will not perform cross-validation, let's just use one of the λ values. You can extract all the coefficients using the `coef()` function. This will give you a matrix of 100 columns, associated with 100 different λ values. Let's use the coefficients associated with the 40th smallest λ value. Based on these coefficients, calculate the predicted (using training data) probabilities of all observations. Use a histogram to plot all of them.
- Using 0.5 as the cut-off value of predicted probability, produce the confusion table of the training data. What is the classification error associated with this model?
- Produce the ROC curve p61-lot associated with your model and report the AUC.

First I use the `glmnet()` function to produce a set of coefficients across many λ values. I use Ridge penalty by setting alpha to be 0.

```
install.packages("glmnet", repos = "http://cran.us.r-project.org")
```

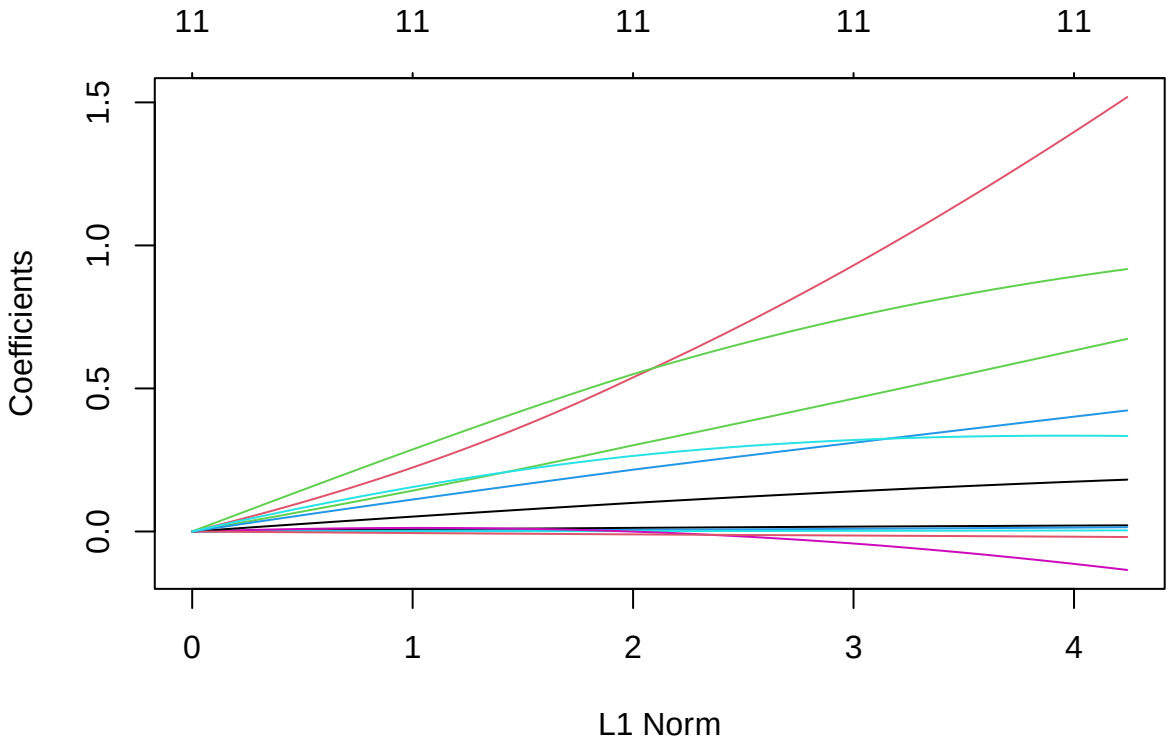
```
##
## The downloaded binary packages are in
##  /var/folders/9c/3_mgdyf12z7dvb8rt4d60nt80000gn/T//Rtmp3NBcie/downloaded_packages
```

```
library(glmnet)
```

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

```
## Loaded glmnet 4.1-2
```

```
ridge.fit = glmnet(x = heart[,1:11], y = heart$Y,  family = "binomial", alpha=0)
plot(ridge.fit)
```




```
ridge.fit$lambda
```

```
##   [1] 215.21069674 196.09197160 178.67170131 162.79899982 148.33638538
##   [6] 135.15858975 123.15147316 112.21103571 102.24251658  93.15957320
##  [11]  84.88353349  77.34271433  70.47180076  64.21128021  58.50692706
##  [16]  53.30933291  48.57347871  44.25834474  40.32655538  36.74405534
```

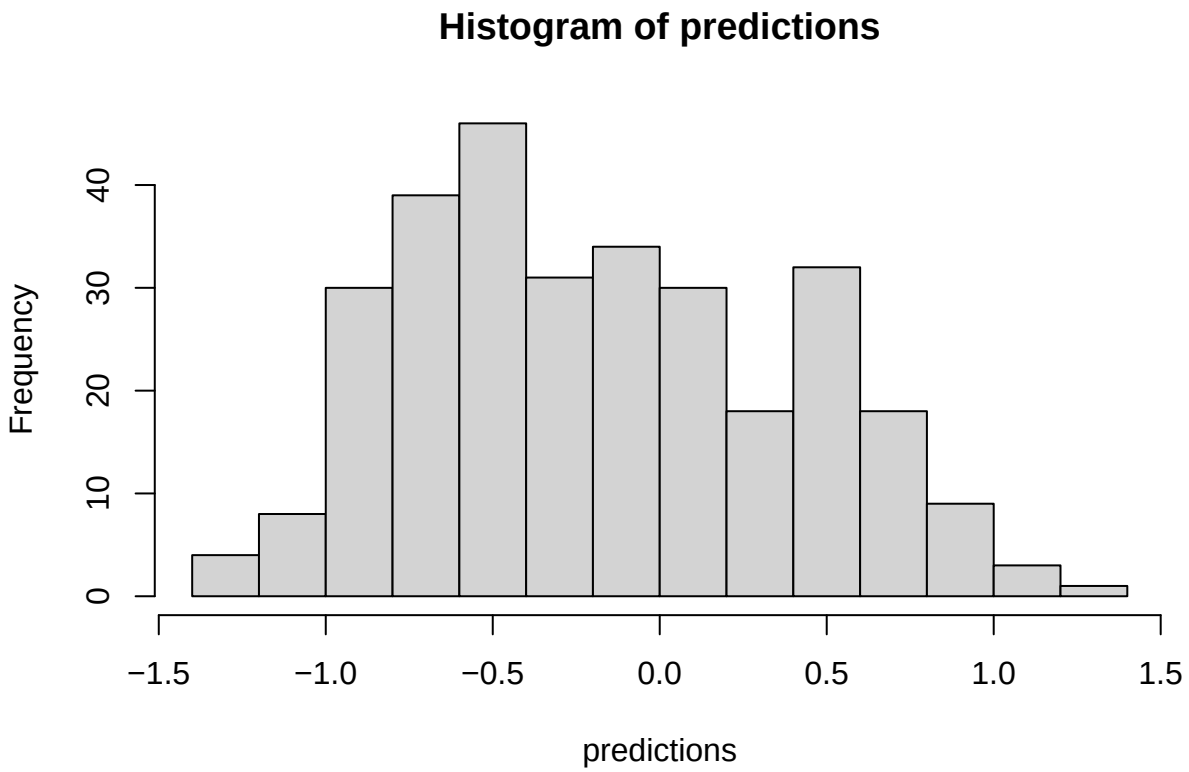
```
##  [21]  33.47981474  30.50556028  27.79553038  25.32625207  23.07633764
##  [26]  21.02629941  19.15838092  17.45640316  15.90562442  14.49261259
##  [31]  13.20512883  12.03202158  10.96312994   9.98919570   9.10178308
##  [36]   8.29320576   7.55646022   6.88516512   6.27350603   5.71618504
##  [41]   5.20837492   4.74567725   4.32408436   3.93994462   3.58993080
##  [46]   3.27101124   2.98042362   2.71565100   2.47440005   2.25458117
##  [51]   2.05429039   1.87179290   1.70550797   1.55399535   1.41594268
##  [56]   1.29015424   1.17554049   1.07110871   0.97595437   0.88925327
##  [61]   0.81025446   0.73827369   0.67268749   0.61292779   0.55847697
##  [66]   0.50886342   0.46365739   0.42246735   0.38493651   0.35073981
##  [71]   0.31958106   0.29119036   0.26532181   0.24175135   0.22027483
##  [76]   0.20070622   0.18287603   0.16662983   0.15182689   0.13833901
##  [81]   0.12604935   0.11485148   0.10464839   0.09535171   0.08688093
##  [86]   0.07916267   0.07213007   0.06572224   0.05988366   0.05456376
##  [91]   0.04971646   0.04529979   0.04127548   0.03760868   0.03426763
##  [96]   0.03122339   0.02844959   0.02592220   0.02361935   0.02152107
```

```
#Getting the cofficients associated with the 40th smallest lambda value.
# The index is 61 because lambda is ordered decreasingly.
coef(ridge.fit, s = ridge.fit$lambda[61])
```

```
## 12 x 1 sparse Matrix of class "dgCMatrix"
##                         s1
## (Intercept) -1.5306317862
## age          0.0079723202
## sex          0.2663049205
## cp           0.1665218522
## trestbps     0.0031153407
## chol         0.0006362841
## fbs          0.0121656823
## restecg      0.0597541176
## thalach     -0.0062508067
## exang        0.3302887787
## oldpeak      0.1284934991
## slope        0.1756962931
```

I calculate the predicted probabilities of all observations, based on the training data. The predict() function gives 303 predicted values (not shown).

```
predictions = predict(ridge.fit, newx = as.matrix(heart[,1:11]),
                      s = ridge.fit$lambda[61])
hist(predictions)
```




```
# Confusion table with a cut off of 0.5
table(predictions > 0.5, heart$Y)
```

```
##
##         FALSE TRUE
##   FALSE   163   94
##   TRUE      1   45
```

The error rate of this model is (94+1)/303 ~ 0.3135 or approx. 31.35%

Part III [10 Points]

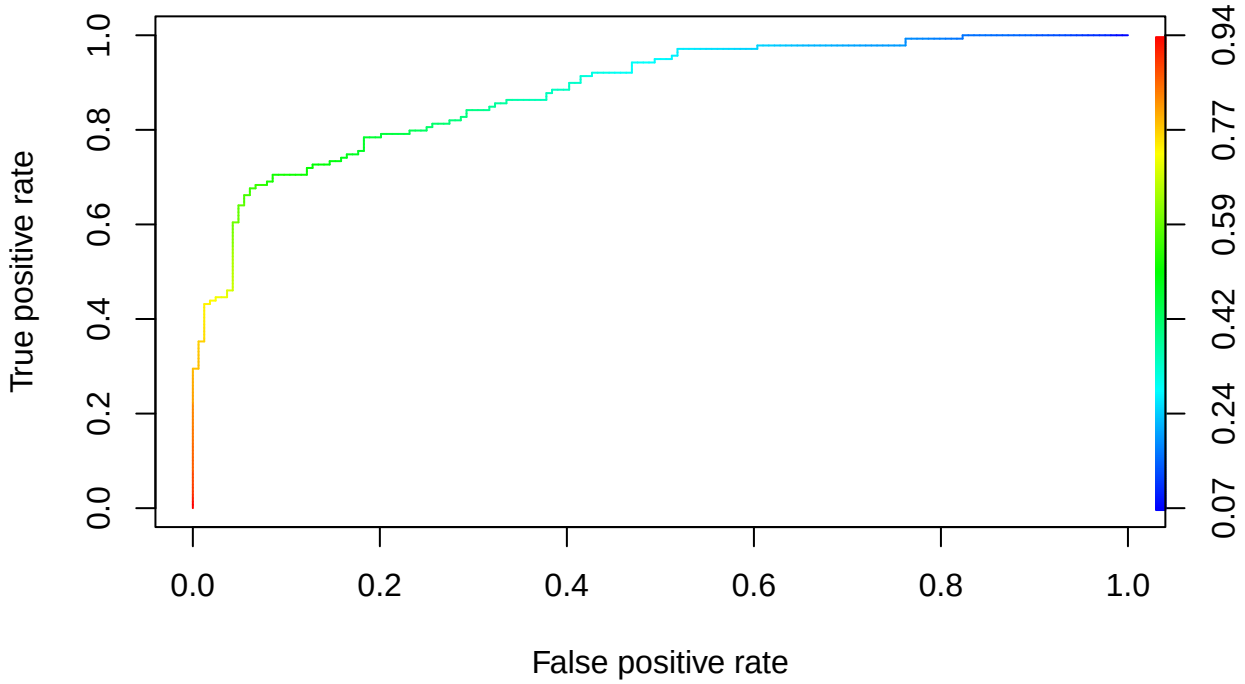
In this last part, we will use a built-in feature of the `glmnet` package. Read the documentation of the `cv.glmnet()` function at CRAN and understand how to specify the `type.measure` argument so that the cross-validation uses the AUC as the selection criterion of λ to pick the best model. Implement a 10-fold cross-validation Ridge regression using our data and report the best λ value (`"lambda.min"`). What is the cross-validation AUC associated with this penalty?

```
set.seed(662095561)
library(glmnet)
auc.fit = cv.glmnet(x = as.matrix(heart[,1:11]), y = heart$Y,
                    type.measure = "auc", nfold = 10,
                    family = "binomial", alpha=0)
# Best lambda value
auc.fit$lambda.min
```

```
## [1] 0.182876
```

To calculate the AUC, I used the library ROC and plot the AUC curve. The cross-validation AUC is 0.88190917.

```
pred_auc = predict(auc.fit, newx = as.matrix(heart[,1:11]),
                   y = heart$Y, type = "response", s = "lambda.min")
roc_auc <- prediction(pred_auc, heart$Y)
# calculates the ROC curve
perf_auc<- performance(roc_auc,"tpr","fpr")
plot(perf_auc,colorize=TRUE)
```




```
performance(roc_auc, measure = "auc")@y.values[[1]]
```

```
## [1] 0.8819091
```