Tutorial 8 & 9 MY OWN Code

Dawn Cheung

2024-04-28

Naive Bayes

Data set Titanic.csv provides information on the fate of passengers on the fatal maiden voyage of the ocean liner Titanic,. It includes the variables: economic status (class), sex, age and survival. We will train a naive Bayes classifier using this data set, and predict survival

a. Compute the probabilities P(Y = 1) (survived) and P(Y = 0) (did not survive).

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.3.2
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.2     ✔ readr     2.1.4
## ✔ forcats   1.0.0     ✔ stringr   1.5.0
## ✔ ggplot2   3.5.1     ✔ tibble    3.2.1
## ✔ lubridate 1.9.2     ✔ tidyr     1.3.0
## ✔ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to be
come errors
```

```
data = read.csv("~/Github/DSA1101 Slayers/datasets/Titanic.csv")
glimpse(data)
```

```
## Rows: 2,201
## Columns: 4
## $ Class    <chr> "3rd", "3rd", "3rd", "3rd", "3rd", "3rd", "3rd", "3rd", "3rd"…
## $ Sex      <chr> "Male", "Male", "Male", "Male", "Male", "Male", "Male", "Male…
## $ Age      <chr> "Child", "Child", "Child", "Child", "Child", "Child", "Child"…
## $ Survived <chr> "No", "No", "No", "No", "No", "No", "No", "No", "No", "No", "…
```

```
#data$Survived = as.factor(data$Survived)
#data$Sex = as.factor(data$Sex)
#data$Class = as.factor(data$Class)

attach(data)


prop.table(table(Survived))
```

```
## Survived
##       No      Yes
## 0.676965 0.323035
```

```
ProbY1 = prop.table(table(Survived))[[1]] #double bracket to remove the column name
ProbY0 = prop.table(table(Survived))[[2]]

ProbY1
```

```
## [1] 0.676965
```

```
ProbY0
```

```
## [1] 0.323035
```

```
#> tprior <- table(Survived) # Number of ppl survived & not survived
#> tprior
#Survived
#  No  Yes
#1490  711
#> tprior <- tprior/sum(tprior) # the probability scores
#> tprior
#Survived
#      No      Yes
#0.676965 0.323035
```

b. Compute the conditional probabilities P(Xi = xi |Y = 1) and P(Xi = xi |Y = 0), where i = 1, 2, 3, 4 for the feature variables X = {class, sex, age}.

```
#P(Xi = xi |Y = 1) = P(class = 3rd)
c = table(Survived, Class); c
```

```
##         Class
## Survived 1st 2nd 3rd Crew
##      No  122 167 528  673
##      Yes 203 118 178  212
```

```
s = table(Survived, Sex); s
```

```
##         Sex
## Survived Female Male
##      No     126 1364
##      Yes    344  367
```

```
a = table(Survived, Age); a
```

```
##         Age
## Survived Adult Child
##      No   1438    52
##      Yes   654    57
```

```
c = c/rowSums(c); c #remember that u shld divide by the total number of yes/no OF THE WHOLE D
ATASET
```

```
##         Class
## Survived        1st        2nd        3rd       Crew
##      No  0.08187919 0.11208054 0.35436242 0.45167785
##      Yes 0.28551336 0.16596343 0.25035162 0.29817159
```

```
s = c/rowSums(s); s
```

```
##         Class
## Survived          1st          2nd          3rd         Crew
##      No  5.495248e-05 7.522184e-05 2.378271e-04 3.031395e-04
##      Yes 4.015659e-04 2.334225e-04 3.521120e-04 4.193693e-04
```

```
a = c/rowSums(a); a
```

```
##         Class
## Survived          1st          2nd          3rd         Crew
##      No  5.495248e-05 7.522184e-05 2.378271e-04 3.031395e-04
##      Yes 4.015659e-04 2.334225e-04 3.521120e-04 4.193693e-04
```

```
#to add all rows: rowSums()

#        Class
#Survived        1st        2nd        3rd       Crew
#     No  0.08187919 0.11208054 0.35436242 0.45167785
#     Yes 0.28551336 0.16596343 0.25035162 0.29817159

#NOTE: completely wrong to use prop.table. we need to get the total divisible to be the entir
e dataset(?)
```

c. Predict survival for an adult female passenger in 2nd class cabin

```
#P(Y = survive |Age = Adult, Sex = Female, Class = 2nd)
Yum = ProbY1 * c[[2,2]] * s[[1,2]] * s[[1,2]]

NotYum = ProbY0 * c[[2,1]] * s[[1,1]] * s[[1,1]]

if (Yum > NotYum) {
  print("more likely to survive")
} else {
  print("death is inevitable")
}
```

```
## [1] "more likely to survive"
```

d. Compare your prediction in (c) with the one performed by the naiveBayes() function in package `e1071'.

```
library("e1071")

M1 <- naiveBayes(Survived ~ Age + Sex + Class, data)#, laplace=0)

newdata = cbind(Age = "Adult", Sex = "Female", Class = "2nd") #maybe use data.frame() instead

predict(M1, newdata = newdata, type = "class")
```

```
## [1] Yes
## Levels: No Yes
```

```
predict(M1, newdata = newdata, type = "raw")
```

```
##             No       Yes
## [1,] 0.2060556 0.7939444
```

```
#prediction is the sameeeee
```

Naive Bayes + Deision Trees, ROC, AUC

Consider the data set Titanic.csv again.

a. Fit a decision tree of on all the three feature variables, called M2, which uses minsplit = 1 and information gain.

```
library(rpart)
```

```
## Warning: package 'rpart' was built under R version 4.3.2
```

```
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 4.3.2
```

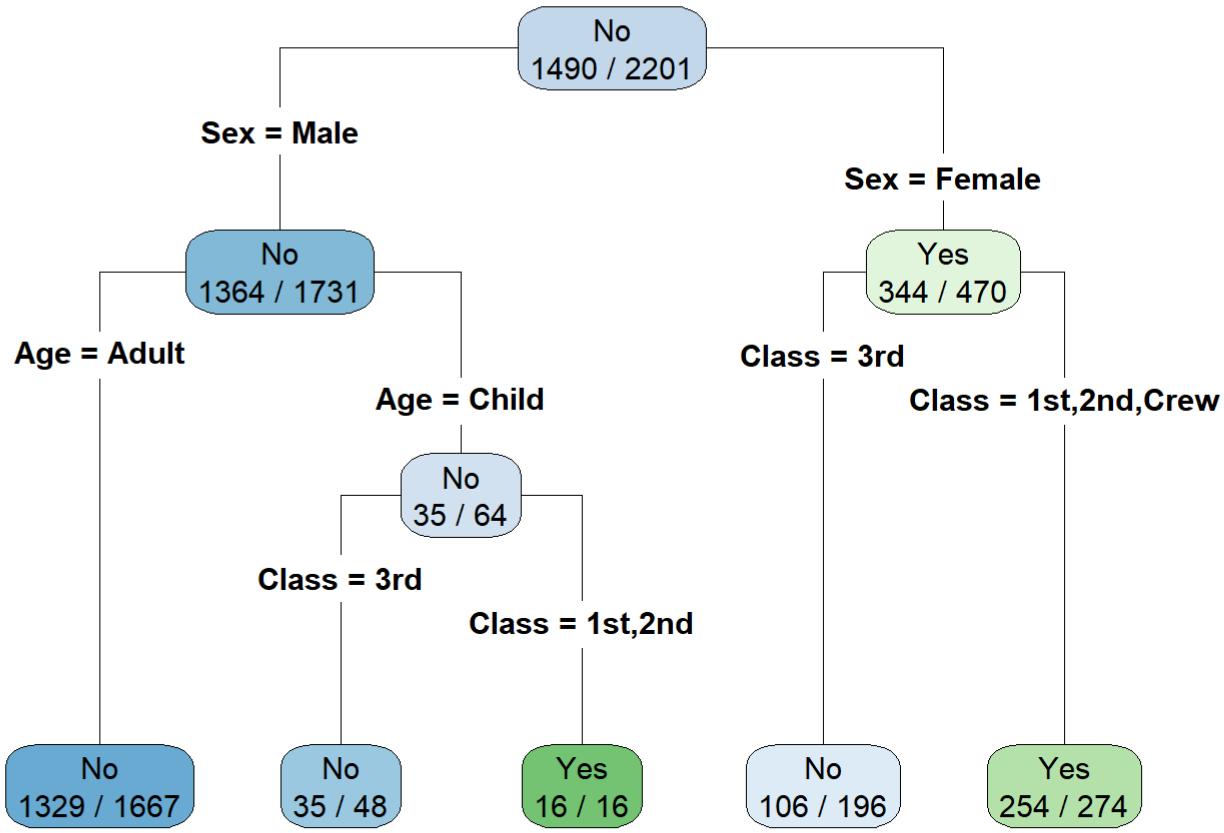
```
M2 <- rpart(Survived ~ Age + Sex + Class,
             method = "class",
             data = data,
             control = rpart.control(minsplit = 1),
             parms = list(split = "information"))
summary(M2)
```

```
## Call:
## rpart(formula = Survived ~ Age + Sex + Class, data = data, method = "class",
##     parms = list(split = "information"), control = rpart.control(minsplit = 1))
##   n= 2201
##
##           CP nsplit rel error    xerror       xstd
## 1 0.30661041      0 1.0000000 1.0000000 0.03085662
## 2 0.02250352      1 0.6933896 0.6933896 0.02750982
## 3 0.01125176      2 0.6708861 0.6807314 0.02732928
## 4 0.01000000      4 0.6483826 0.6652602 0.02710334
##
## Variable importance
##   Sex Class   Age
##    69    26     5
##
## Node number 1: 2201 observations,    complexity param=0.3066104
##   predicted class=No   expected loss=0.323035  P(node) =1
##     class counts:  1490   711
##    probabilities: 0.677 0.323
##   left son=2 (1731 obs) right son=3 (470 obs)
##   Primary splits:
##       Sex   splits as  RL,   improve=217.234400, (0 missing)
##       Class splits as  RRLL, improve= 76.688460, (0 missing)
##       Age   splits as  LR,   improve=  9.780301, (0 missing)
##
## Node number 2: 1731 observations,    complexity param=0.01125176
##   predicted class=No   expected loss=0.2120162  P(node) =0.7864607
##     class counts:  1364   367
##    probabilities: 0.788 0.212
##   left son=4 (1667 obs) right son=5 (64 obs)
##   Primary splits:
##       Age   splits as  LR,   improve=9.673810, (0 missing)
##       Class splits as  RLLL, improve=9.491997, (0 missing)
##
## Node number 3: 470 observations,    complexity param=0.02250352
##   predicted class=Yes  expected loss=0.2680851  P(node) =0.2135393
##     class counts:   126   344
##    probabilities: 0.268 0.732
##   left son=6 (196 obs) right son=7 (274 obs)
##   Primary splits:
##       Class splits as  RRLR, improve=66.429510, (0 missing)
##       Age   splits as  RL,   improve= 1.432219, (0 missing)
##   Surrogate splits:
##       Age splits as  RL, agree=0.619, adj=0.087, (0 split)
##
## Node number 4: 1667 observations
##   predicted class=No   expected loss=0.2027594  P(node) =0.757383
##     class counts:  1329   338
##    probabilities: 0.797 0.203
##
## Node number 5: 64 observations,    complexity param=0.01125176
##   predicted class=No   expected loss=0.453125  P(node) =0.02907769
##     class counts:    35    29
##    probabilities: 0.547 0.453
##   left son=10 (48 obs) right son=11 (16 obs)
```

```
##   Primary splits:
##       Class splits as  RRL-, improve=16.04363, (0 missing)
##
## Node number 6: 196 observations
##   predicted class=No   expected loss=0.4591837  P(node) =0.08905043
##     class counts:   106    90
##    probabilities: 0.541 0.459
##
## Node number 7: 274 observations
##   predicted class=Yes  expected loss=0.0729927  P(node) =0.1244889
##     class counts:    20   254
##    probabilities: 0.073 0.927
##
## Node number 10: 48 observations
##   predicted class=No   expected loss=0.2708333  P(node) =0.02180827
##     class counts:    35    13
##    probabilities: 0.729 0.271
##
## Node number 11: 16 observations
##   predicted class=Yes  expected loss=0  P(node) =0.007269423
##     class counts:     0    16
##    probabilities: 0.000 1.000
```

b. Plot the tree M2.

```
rpart.plot(M2, type = 4, extra = 2, clip.right.labs = FALSE)
```



c. Plot the ROC curves and derive the AUC values for the two classifiers (naive Bayes from question 1 and decision tree). Which classifier has larger AUC value?

```
library(dplyr)
library(ROCR)

ncol(data)



sur = ifelse(Survived == "Yes", 1,0)

sur = as.factor()

data

# > typeof(data.frame(data[,-ncol(data)]))
# [1] "list"
#HUHHHH

#do NB first
nb_prediction = predict(M1,
    newdata = data.frame(data[,-ncol(data)]),
    type = "raw")

dt_prediction = predict(M2, newdata = data.frame(data[,-ncol(data)]), type = "raw")

nb_predicted_score = nb_prediction[,c("Yes")]
dt_predicted_score = dt_prediction[,c("Yes")]

actual_class = data$Survived == "Yes"

nb_pred = prediction(nb_predicted_score, actual_class)
dt_pred = prediction(dt_predicted_score, actual_class)

nb_perf <- performance(nb_pred , "tpr", "fpr")
dt_perf <- performance(dt_pred , "tpr", "fpr")

auc1 <- performance(pre_nb, "auc")@y.values[[1]]
```

Logistic Regression (Tutorial 9)

Consider the data set Titanic.csv again. NOTE: I have changed M2 in tut 9 to M3 to prevent confusion with decision tree model in tut 8

1. Perform logistic regression using all the feature variables to predict the survival status, called model M3.

```
#REMEMBER MUST CONVERT TO 0 and 1
sur = (Survived == "Yes")
sur = as.factor(sur)
data$sur = sur #so sur is a completely new column to mutate onto the database
attach(data)
```

```
## The following object is masked _by_ .GlobalEnv:
##
##     sur
```

```
## The following objects are masked from data (pos = 6):
##
##     Age, Class, Sex, Survived
```

```
glimpse(data)
```

```
## Rows: 2,201
## Columns: 5
## $ Class    <chr> "3rd", "3rd", "3rd", "3rd", "3rd", "3rd", "3rd", "3rd", "3rd"…
## $ Sex      <chr> "Male", "Male", "Male", "Male", "Male", "Male", "Male", "Male…
## $ Age      <chr> "Child", "Child", "Child", "Child", "Child", "Child", "Child"…
## $ Survived <chr> "No", "No", "No", "No", "No", "No", "No", "No", "No", "No", "…
## $ sur      <fct> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE…
```

```
#note that glm is an in-built function-- no library needed
M3 <- glm(sur ~ Age + Sex + Class,
          data = data,
          family = binomial(link = "logit")) #family decides the transformation g from g(y)

summary(M3)
```

```
##
## Call:
## glm(formula = sur ~ Age + Sex + Class, family = binomial(link = "logit"),
##     data = data)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.0438     0.1679  12.171  < 2e-16 ***
## AgeChild      1.0615     0.2440   4.350 1.36e-05 ***
## SexMale      -2.4201     0.1404 -17.236  < 2e-16 ***
## Class2nd     -1.0181     0.1960  -5.194 2.05e-07 ***
## Class3rd     -1.7778     0.1716 -10.362  < 2e-16 ***
## ClassCrew    -0.8577     0.1573  -5.451 5.00e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2769.5  on 2200  degrees of freedom
## Residual deviance: 2210.1  on 2195  degrees of freedom
## AIC: 2222.1
##
## Number of Fisher Scoring iterations: 4
```

2. Write down the fitted equation of model M3.

```
paste0("log[phat/(1-phat)] = ", M3$coeff[1]," + ", M3$coeff[2], "*I(Age = Child) + ", M3$coef
f[3], "*I(Sex = Male) + ", M3$coeff[4], "*I(Class = 2nd) + ", M3$coeff[5], "*I(Class = 3rd) +
", M3$coeff[6], "*I(Class = Crew)")
```

```
## [1] "log[phat/(1-phat)] = 2.04383742228878 + 1.06154237626018*I(Age = Child) + -2.42006034
580599*I(Sex = Male) + -1.01809495158843*I(Class = 2nd) + -1.77776221774166*I(Class = 3rd) +
-0.857676155374125*I(Class = Crew)"
```

```
#remember the * to show multiplication lol
```

3. Interpret the coefficient of the variable `Sex' in M3.

Note that female is reference. Male is indicated by indicator.

When other variables are the same, compared to females, the log odds of surviving for a male is less than females by 2.42.

The odds of surviving for a male is e^-2.42 times (which is 11.25 less times) the odds of surviving for a females.

4. Interpret the coefficient of the variable `Age' in M3.

When other variables are the same, compared to an Adult, the log odds of surviving for a child is more than adults by 1.0615.

The odds of surviving for a child is e^1.0615 times (which is 2.89 more times) the odds of surviving for a adult.

5. Observe and compare the ROC curves and AUC for the two classifiers: naive Bayes (from Tutorial 8) and logistic regression.

```
library(ROCR)
```

```
## Warning: package 'ROCR' was built under R version 4.3.3
```

```
#ROC for Log Reg
pred = predict(M3, type = "response")
preObj = prediction(pred, data$Survived) #notice here we don't need to use sur, the OG column
is fine
rocObj = performance(preObj, measure = "tpr", x.measure = "fpr")
plot(rocObj)

#getting AUC value for Log Reg
aucLR = performance(preObj, measure = "auc")
aucLR@y.values[[1]] #0.7597259
```

```
## [1] 0.7597259
```

```r
#ROC for NB
#Naive Bayes requires more formatting!!!
naiveB = predict(M1, data[1:3], type = "raw") #for NB u need to specify what response variabl
es (from the database) are needed


score = naiveB[, 2]
#OK SO naiveB[,c("Yes")] and naiveB[,2] IS THE SAME bc u see what predict() returns
#                 No       Yes
#   [1,] 0.69605930 0.3039407
#   [2,] 0.69605930 0.3039407
#   [3,] 0.69605930 0.3039407
#yea so basically u want that Yes column
#these are the predicted Yes'es by the way

preObjNB = prediction(score, data$sur)
rocObjNB = performance(preObjNB, measure = "tpr", x.measure = "fpr")
plot(rocObjNB, add = TRUE, col = "red") #so to add on to our prev graph

#getting AUC value for NB
aucNB = performance(preObjNB, measure = "auc")
aucNB@y.values[[1]]
```

```
## [1] 0.7164944
```

```r
legend("bottomright", c("Logistic Regression", "Naive Bayes"), col = c("black", "red"), lty =
1)
```