Assignment-3

Srilaya Valmeekam

03-05-2022

```
library(caret)
```

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

```
## Loading required package: lattice
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(lattice)
library(knitr)
library(rmarkdown)
library(e1071)
```

```
#importing the Data set
library(readr)
UniversalBank <- read.csv("~/Downloads/UniversalBank.csv")
```

```
##The text that follows simply takes the data file, deletes the zip code and ID (as last time, but unne
SLV <- UniversalBank %>% select(Age, Experience, Income, Family, CCAvg, Education, Mortgage,Personal.Loa
SLV$CreditCard <- as.factor(SLV$CreditCard)
SLV$Personal.Loan <- as.factor((SLV$Personal.Loan))
SLV$Online <- as.factor(SLV$Online)
```

```
#The train data, validation data, and data division are generated in this way.
selected.var <- c(8,11,12)
set.seed(23)
Train_Index = createDataPartition(SLV$Personal.Loan, p=0.60, list=FALSE)
Train_Data = SLV[Train_Index,selected.var]
Validation_Data = SLV[-Train_Index,selected.var]
```

```
##A. Create a pivot table for the training data with Online as a column variable, CC as a row variable,
#Online is a column, and CC and LOAN are both rows in the final pivot table.
attach(Train_Data)
##ftable "function table".
ftable(CreditCard,Personal.Loan,Online)
```

```
##                           Online    0    1
## CreditCard Personal.Loan
## 0          0                     773 1127
##            1                      82  114
## 1          0                     315  497
##            1                      39   53
```

```
detach(Train_Data)
```

##In order to calculate the conditional probability that Loan=1, we add 53 (Loan=1 from ftable) to 497 (Loan=0 from ftable), which makes 550. 53/550 = 0.096363 or 9.64% of the time.

```
##B. Consider the task of classifying a customer who owns a bank credit card and is actively using onli
prop.table(ftable(Train_Data$CreditCard,Train_Data$Online,Train_Data$Personal.Loan),margin=1)
```

```
##                0          1
##
## 0 0  0.90409357 0.09590643
##    1  0.90813860 0.09186140
## 1 0  0.88983051 0.11016949
##    1  0.90363636 0.09636364
```

##A percentage pivot table that displays the chance of a loan based on credit card and online data is displayed by the code above.

```
##C. Create two separate pivot tables for the training data. One will have Loan (rows) as a function of
attach(Train_Data)
ftable(Personal.Loan,Online)
```

```
##               Online    0    1
## Personal.Loan
## 0                     1088 1624
## 1                      121  167
```

```
ftable(Personal.Loan,CreditCard)
```

```
##               CreditCard    0    1
## Personal.Loan
## 0                          1900  812
## 1                           196   92
```

```
detach(Train_Data)
```

##In the first column up there, “Internet” makes up a column, “Loans” adds a row, and “Credit Card” makes up a column.

```
##D. Compute the following quantities [P(A | B) means "the probability ofA given B"]:
prop.table(ftable(Train_Data$Personal.Loan,Train_Data$CreditCard),margin=)
```

```
##             0          1
##
## 0  0.63333333 0.27066667
## 1  0.06533333 0.03066667
```

```
prop.table(ftable(Train_Data$Personal.Loan,Train_Data$Online),margin=1)
```

```
##            0         1
##
## 0  0.4011799 0.5988201
## 1  0.4201389 0.5798611
```

SLVi) 92/288 = 0.3194 or 31.94%

SLVii) 167/288 = 0.5798 or 57.986%

SLViii) total loans= 1 from table (288) divide by total from table (3000) = 0.096 or 9.6%

SLViV) 812/2712 = 0.2994 or 29.94%

SLVV) 1624/2712 = 0.5988 or 59.88%

SLVVi) total loans=0 from table(2712) divided by total from table (3000) = 0.904 or 90.4%

##E. Use the quantities computed above to compute the naive Bayes probability P(Loan = 1 | CC = 1,Online = 1).

(0.3194 * 0.5798 * 0.096)/[(0.3194 * 0.5798 * 0.096)+(0.2994 * 0.5988 * 0.904)] = 0.0988505642823701 or 9.885%

##F. Compare this value with the one obtained from the pivot table in (B). Which is a more accurate estimate?

The discrepancy between 0.096363, or 9.64%, and 0.0988505642823701, or 9.885%, is not statistically significant. The pivot table value is the predicted value with greater accuracy because it is independent of the probabilities' independence. E examines the probability of each of those counts, whereas B uses a straightforward calculation from a tally. B is therefore more exact, whereas E is best for generality.

```
##G. Which of the entries in this table are needed for computing P(Loan = 1 | CC = 1, Online = 1)? Run
##training dataset
UniversalBank.SLV <- naiveBayes(Personal.Loan ~ ., data = Train_Data)
UniversalBank.SLV
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##     0     1
## 0.904 0.096
##
## Conditional probabilities:
##    Online
## Y           0         1
##   0 0.4011799 0.5988201
##   1 0.4201389 0.5798611
##
##    CreditCard
## Y           0         1
##   0 0.7005900 0.2994100
##   1 0.6805556 0.3194444
```

The pivot table in step B can be used to rapidly compute P(LOAN=1|CC=1,Online=1) without relying on the Naive Bayes model, while using the two tables created in step C makes it simple and obvious HOW you are getting P(LOAN=1|CC=1,Online=1)using the Naive Bayes model.

But compared to the probability determined manually in step E, the model's prediction is less possible. The earlier approaches and the Naive Bayes model both forecast probabilities. More probable than the one from

step B is the estimated probability. This might be the case because step E requires manual computation, which carries the risk of error when rounding fractions and only yields an approximation.

```
## SLV confusion matrix about Train_Data
##Training
predicting.class <- predict(UniversalBank.SLV, newdata = Train_Data)
confusionMatrix(predicting.class, Train_Data$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 2712  288
##          1    0    0
##
##                Accuracy : 0.904
##                  95% CI : (0.8929, 0.9143)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : 0.5157
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 1.000
##             Specificity : 0.000
##          Pos Pred Value : 0.904
##          Neg Pred Value :   NaN
##              Prevalence : 0.904
##          Detection Rate : 0.904
##    Detection Prevalence : 1.000
##       Balanced Accuracy : 0.500
##
##        'Positive' Class : 0
##
```

This model had a low degree of specificity despite its high sensitivity. Despite the fact that the reference data held all actual values, the model predicted that all values would be 0. Even if the model totally missed all values of 1, it would still produce a 90.4% accuracy because of the sizeable amount of 0.

```
predicting.prob <- predict(UniversalBank.SLV, newdata=Validation_Data, type="raw")
predicting.class <- predict(UniversalBank.SLV, newdata = Validation_Data)
confusionMatrix(predicting.class, Validation_Data$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1808  192
##          1    0    0
##
##                Accuracy : 0.904
##                  95% CI : (0.8902, 0.9166)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : 0.5192
##
```

```
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 1.000
##             Specificity : 0.000
##          Pos Pred Value : 0.904
##          Neg Pred Value :   NaN
##              Prevalence : 0.904
##          Detection Rate : 0.904
##    Detection Prevalence : 1.000
##       Balanced Accuracy : 0.500
##
##        'Positive' Class : 0
##
```

Let's now examine the model visually and select the optimal threshold.

```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
roc(Validation_Data$Personal.Loan,predicting.prob[,1])
```

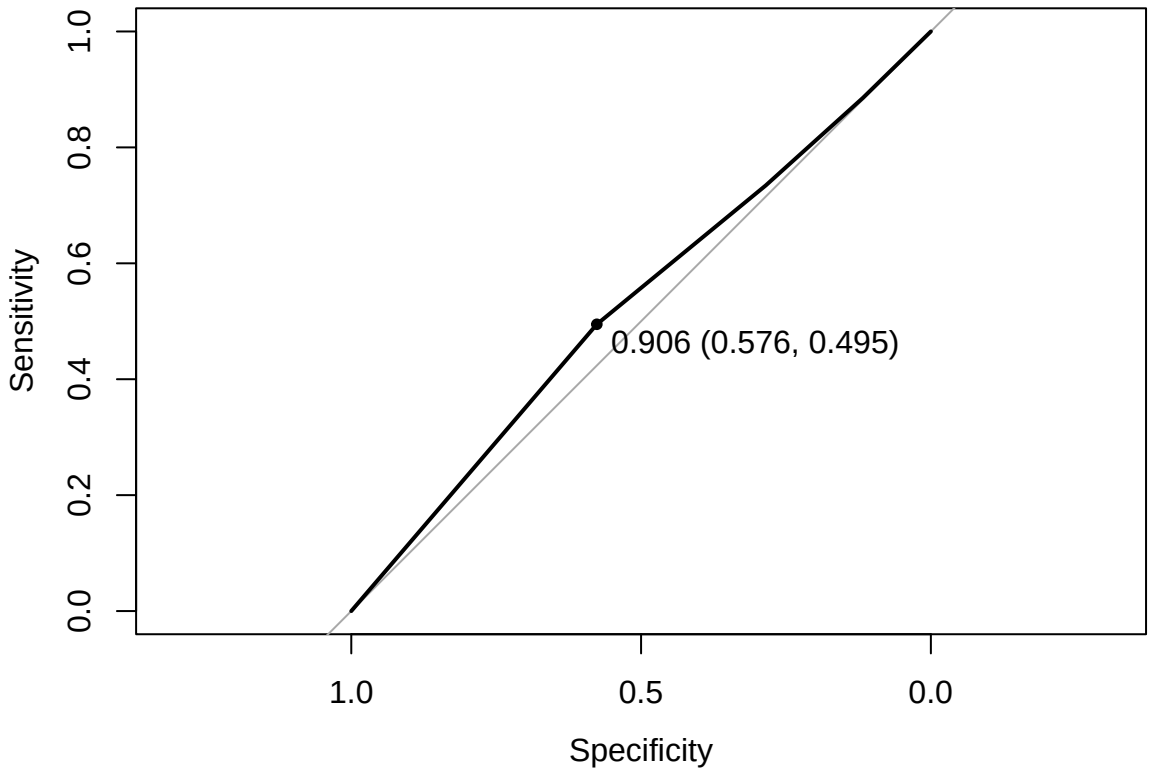
```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
##
## Call:
## roc.default(response = Validation_Data$Personal.Loan, predictor = predicting.prob[,     1])
##
## Data: predicting.prob[, 1] in 1808 controls (Validation_Data$Personal.Loan 0) < 192 cases (Validatio
## Area under the curve: 0.5302
```

```
plot.roc(Validation_Data$Personal.Loan,predicting.prob[,1],print.thres="best")
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```



Therefore, it can be shown that the model is enhanced by selecting a limit of 0.906, which would reduce sensitivity to 0.495 and raise specificity to 0.576.