FML Assignment 3

Prasanth Yethirajula

2023-10-13

Summary: The "accidentsFull.csv" file has data on over 42,000 car crashes in the U.S. from 2001. These crashes range from no harm to deadly. The data also includes details like the day it happened, the weather, and the kind of road. Companies might want a way to quickly understand how bad a new crash is based on initial details, some of which come from GPS reports.

We aim to figure out if a newly reported crash caused any harm based on a value called "MAX_SEV_IR". If "MAX_SEV_IR" is 1 or 2, we'll label it as "yes" for injury. If it's 0, we'll label it as "no" for no injury.

```
# Install and load the libraries
library(caret)
```

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

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

```
library(e1071)
```

```
# Read the dataset
accidents <- read.csv("C:/Users/drpra/Downloads/accidentsFull.csv")
# Display first six records
head(accidents)
```

```
##   HOUR_I_R ALCHL_I ALIGN_I STRATUM_R WRK_ZONE WKDY_I_R INT_HWY LGTCON_I_R
## 1        0       2       2         1        0        1       0          3
## 2        1       2       1         0        0        1       1          3
## 3        1       2       1         0        0        1       0          3
## 4        1       2       1         1        0        0       0          3
## 5        1       1       1         0        0        1       0          3
## 6        1       2       1         1        0        1       0          3
##   MANCOL_I_R PED_ACC_R RELJCT_I_R REL_RWY_R PROFIL_I_R SPD_LIM SUR_COND
## 1          0         0          1         0          1      40        4
## 2          2         0          1         1          1      70        4
## 3          2         0          1         1          1      35        4
## 4          2         0          1         1          1      35        4
## 5          2         0          0         1          1      25        4
## 6          0         0          1         0          1      70        4
```

```
##   TRAF_CON_R TRAF_WAY VEH_INVL WEATHER_R INJURY_CRASH NO_INJ_I PRPTYDMG_CRASH
## 1          0        3        1         1             1       1               0
## 2          0        3        2         2             0       0               1
## 3          1        2        2         2             0       0               1
## 4          1        2        2         1             0       0               1
## 5          0        2        3         1             0       0               1
## 6          0        2        1         2             1       1               0
##   FATALITIES MAX_SEV_IR
## 1          0          1
## 2          0          0
## 3          0          0
## 4          0          0
## 5          0          0
## 6          0          1
```

Question 1: Using the information in this dataset, if an accident has just been reported and no further information is available, what should the prediction be? (INJURY = Yes or No?) Why?

Create a dummy variable called 'Injury'

```
# value "yes" if MAX_SEV_IR = 1 or 2, and otherwise "no."
accidents$INJURY= ifelse(accidents$MAX_SEV_IR>0, "YES", "NO")
table(accidents$INJURY)
```

```
##
##    NO   YES
## 20721 21462
```

If an accident has just been reported and no information is available, it is assumed that injuries have occurred, i.e. (INJURY = Yes). The purpose is to forecast whether or not an accident will result in an injury (MAX_SEV_IR = 1 or 2) or not (MAX_SEV_IR = 0).So, if you have no exact knowledge about a new accident and wish to make an initial forecast, it would be appropriate to predict that there is a potential of injury ("INJURY" = "Yes") because injuries occurred in a proportion of accidents in the historical data.

There are a total of "20721 NO and 21462 YES"

Question 2: Select the first 24 records in the dataset and look only at the response (INJURY) and the two predictors WEATHER_R and TRAF_CON_R. Create a pivot table that examines INJURY as a function of the two predictors for these 24 records. Use all three variables in the pivot table as rows/columns.

```
#selecting first 24 records and look at response (INJURY) and 2 predictors WEATHER_R and TRAF_CON_R
#CONVERTING THE VARIABLES TO CATEGORICAL TYPE
# IDENTIFYING THE TARGET VARIABLE COLUMN INDEX (ASSUMING IT'S THE LAST COLUMN)
target_col = dim(accidents)[2]

#CONVERTING ALL COLUMNS EXCEPT THE TARGRT VARIABLE TO FACTORS
accidents[, 1:(target_col - 1)] = lapply(accidents[, 1:(target_col - 1)], as.factor)

#create a new subset with only the required records
accidents_24 = accidents[1:24, c("INJURY", "WEATHER_R", "TRAF_CON_R")]
accidents_24
```

```
##    INJURY WEATHER_R TRAF_CON_R
## 1     YES         1          0
## 2      NO         2          0
## 3      NO         2          1
## 4      NO         1          1
## 5      NO         1          0
## 6     YES         2          0
## 7      NO         2          0
## 8     YES         1          0
## 9      NO         2          0
## 10     NO         2          0
## 11     NO         2          0
## 12     NO         1          2
## 13    YES         1          0
## 14     NO         1          0
## 15    YES         1          0
## 16    YES         1          0
## 17     NO         2          0
## 18     NO         2          0
## 19     NO         2          0
## 20     NO         2          0
## 21    YES         1          0
## 22     NO         1          0
## 23    YES         2          2
## 24    YES         2          0
```

```
#create a  pivot table
x1= ftable(accidents_24)
x2= ftable(accidents_24[,-1]) # table for fatality
x1
```

```
##                   TRAF_CON_R 0 1 2
## INJURY WEATHER_R
## NO     1                     3 1 1
##        2                     9 1 0
## YES    1                     6 0 0
##        2                     2 0 1
```

```
x2
```

```
##           TRAF_CON_R  0  1  2
## WEATHER_R
## 1                     9  1  1
## 2                    11  1  1
```

2(1) Compute the exact Bayes conditional probabilities of an injury (INJURY = Yes) given the six possible combinations of the predictors.

```
# P(INJURY= YES|WEATHER_R= 1, TRAF_CON_R= 0)
Prob1= x1[3,1]/x2[1,1]
Prob1
```

```
## [1] 0.6666667
```

```
# P(INJURY= YES|WEATHER_R= 2, TRAF_CON_R= 0)
Prob2= x1[4,1]/x2[2,1]
Prob2
```

```
## [1] 0.1818182
```

```
# P(INJURY= YES|WEATHER_R= 1, TRAF_CON_R= 1)
Prob3= x1[3,2]/x2[1,2]
Prob3
```

```
## [1] 0
```

```
# P(INJURY= YES|WEATHER_R= 2, TRAF_CON_R= 1)
Prob4= x1[4,2]/x2[2,2]
Prob4
```

```
## [1] 0
```

```
# P(INJURY= YES|WEATHER_R= 1, TRAF_CON_R= 2)
Prob5= x1[3,3]/x2[1,3]
Prob5
```

```
## [1] 0
```

```
# P(INJURY= YES|WEATHER_R= 2, TRAF_CON_R= 2)
Prob6= x1[4,3]/x2[2,3]
Prob6
```

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

2(2) Classify the 24 accidents using these probabilities and a cutoff of 0.5.

```
#Adding probability
accidents_24_prob= accidents_24
head(accidents_24_prob)
```

```
##   INJURY WEATHER_R TRAF_CON_R
## 1    YES         1          0
## 2     NO         2          0
## 3     NO         2          1
## 4     NO         1          1
## 5     NO         1          0
## 6    YES         2          0
```

```
probability.injury = c(0.667, 0.167, 0, 0, 0.667, 0.167, 0.167, 0.667, 0.167, 0.167, 0.167, 0)

accidents_24_prob$PROB_INJ = rep(probability.injury, length.out = nrow(accidents_24_prob))

#Add column for injury prediction based on cutoff of 0.5.
accidents_24_prob$PROB_PREDICT=ifelse(accidents_24_prob$PROB_INJ>.5,"YES","NO")
accidents_24_prob
```

```
##    INJURY WEATHER_R TRAF_CON_R PROB_INJ PROB_PREDICT
## 1     YES         1          0    0.667          YES
## 2      NO         2          0    0.167           NO
## 3      NO         2          1    0.000           NO
## 4      NO         1          1    0.000           NO
## 5      NO         1          0    0.667          YES
## 6     YES         2          0    0.167           NO
## 7      NO         2          0    0.167           NO
## 8     YES         1          0    0.667          YES
## 9      NO         2          0    0.167           NO
## 10     NO         2          0    0.167           NO
## 11     NO         2          0    0.167           NO
## 12     NO         1          2    0.000           NO
## 13    YES         1          0    0.667          YES
## 14     NO         1          0    0.167           NO
## 15    YES         1          0    0.000           NO
## 16    YES         1          0    0.000           NO
## 17     NO         2          0    0.667          YES
## 18     NO         2          0    0.167           NO
## 19     NO         2          0    0.167           NO
## 20     NO         2          0    0.667          YES
## 21    YES         1          0    0.167           NO
## 22     NO         1          0    0.167           NO
## 23    YES         2          2    0.167           NO
## 24    YES         2          0    0.000           NO
```

2(3) Compute manually the naive Bayes conditional probability of an injury given WEATHER_R = 1 and TRAF_CON_R = 1.

```
# Naive bayes is calculated for independent variables

numerator= 6/9 * 0 * 9/24
denominator= (6/9 * 0 * 9/24)+(5/15 * 2/15 * 15/24)
naive_bayes= numerator/denominator
naive_bayes
```

```
## [1] 0
```

2(4) Run a naive Bayes classifier on the 24 records and two predictors. Check the model output to obtain probabilities and classifications for all 24 records. Compare this to the exact Bayes classification. Are the resulting classifications equivalent? Is the ranking (= ordering) of observations equivalent?

```
# Install and load the necessary libraries
library(e1071)
library(klaR)
```

```
## Loading required package: MASS
```

```
library(caret)

nb=naiveBayes(INJURY ~ ., data =accidents_24 )
predict(nb, newdata = accidents_24,type = "raw")
```

```
##              NO         YES
##  [1,] 0.4285714 0.571428571
##  [2,] 0.7500000 0.250000000
##  [3,] 0.9977551 0.002244949
##  [4,] 0.9910803 0.008919722
##  [5,] 0.4285714 0.571428571
##  [6,] 0.7500000 0.250000000
##  [7,] 0.7500000 0.250000000
##  [8,] 0.4285714 0.571428571
##  [9,] 0.7500000 0.250000000
## [10,] 0.7500000 0.250000000
## [11,] 0.7500000 0.250000000
## [12,] 0.3333333 0.666666667
## [13,] 0.4285714 0.571428571
## [14,] 0.4285714 0.571428571
## [15,] 0.4285714 0.571428571
## [16,] 0.4285714 0.571428571
## [17,] 0.7500000 0.250000000
## [18,] 0.7500000 0.250000000
## [19,] 0.7500000 0.250000000
## [20,] 0.7500000 0.250000000
## [21,] 0.4285714 0.571428571
## [22,] 0.4285714 0.571428571
## [23,] 0.6666667 0.333333333
## [24,] 0.7500000 0.250000000
```

```
#Check the model with caret package
x= accidents_24[,-3]
y=accidents_24$INJURY
model = train(x,y,'nb', trControl = trainControl(method = 'cv',number=10))
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo,
## : There were missing values in resampled performance measures.
```

```
model
```

```
## Naive Bayes
##
## 24 samples
##  2 predictor
##  2 classes: 'NO', 'YES'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 21, 22, 21, 22, 21, 21, ...
## Resampling results across tuning parameters:
##
##   usekernel  Accuracy  Kappa
##   FALSE      1         1
##    TRUE      1         1
##
## Tuning parameter 'fL' was held constant at a value of 0
## Tuning
```

```
##  parameter 'adjust' was held constant at a value of 1
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were fL = 0, usekernel = FALSE and adjust
##  = 1.
```

```
#Now the generated classification model can be used for prediction
model.predicted=predict(model$finalModel,x)
model.predicted
```

```
## $class
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20
## YES  NO  NO  NO  NO YES  NO YES  NO  NO  NO  NO YES  NO YES YES  NO  NO  NO  NO
##  21  22  23  24
## YES  NO YES YES
## Levels: NO YES
##
## $posterior
##               NO         YES
## 1  0.0008326395 0.999167361
## 2  0.9997000900 0.000299910
## 3  0.9997000900 0.000299910
## 4  0.9988014383 0.001198562
## 5  0.9988014383 0.001198562
## 6  0.0033222591 0.996677741
## 7  0.9997000900 0.000299910
## 8  0.0008326395 0.999167361
## 9  0.9997000900 0.000299910
## 10 0.9997000900 0.000299910
## 11 0.9997000900 0.000299910
## 12 0.9988014383 0.001198562
## 13 0.0008326395 0.999167361
## 14 0.9988014383 0.001198562
## 15 0.0008326395 0.999167361
## 16 0.0008326395 0.999167361
## 17 0.9997000900 0.000299910
## 18 0.9997000900 0.000299910
## 19 0.9997000900 0.000299910
## 20 0.9997000900 0.000299910
## 21 0.0008326395 0.999167361
## 22 0.9988014383 0.001198562
## 23 0.0033222591 0.996677741
## 24 0.0033222591 0.996677741
```

```
#Creating a confusion matrix to visualize classification errors.
table(model.predicted$class,y)
```

```
##       y
##        NO YES
##   NO   15   0
##   YES   0   9
```

```
#Comparing
accidents_24_prob$PREDICT_PROB_NB<-model.predicted$class
accidents_24_prob
```

```
##    INJURY WEATHER_R TRAF_CON_R PROB_INJ PROB_PREDICT PREDICT_PROB_NB
## 1     YES         1          0    0.667          YES             YES
## 2      NO         2          0    0.167           NO              NO
## 3      NO         2          1    0.000           NO              NO
## 4      NO         1          1    0.000           NO              NO
## 5      NO         1          0    0.667          YES              NO
## 6     YES         2          0    0.167           NO             YES
## 7      NO         2          0    0.167           NO              NO
## 8     YES         1          0    0.667          YES             YES
## 9      NO         2          0    0.167           NO              NO
## 10     NO         2          0    0.167           NO              NO
## 11     NO         2          0    0.167           NO              NO
## 12     NO         1          2    0.000           NO              NO
## 13    YES         1          0    0.667          YES             YES
## 14     NO         1          0    0.167           NO              NO
## 15    YES         1          0    0.000           NO             YES
## 16    YES         1          0    0.000           NO             YES
## 17     NO         2          0    0.667          YES              NO
## 18     NO         2          0    0.167           NO              NO
## 19     NO         2          0    0.167           NO              NO
## 20     NO         2          0    0.667          YES              NO
## 21    YES         1          0    0.167           NO             YES
## 22     NO         1          0    0.167           NO              NO
## 23    YES         2          2    0.167           NO             YES
## 24    YES         2          0    0.000           NO             YES
```

Question 3: Let us now return to the entire dataset. Partition the data into training (60%) and validation (40%).

3(1) Run a naive Bayes classifier on the complete training set with the relevant predictors (and INJURY as the response). Note that all predictors are categorical. Show the confusion matrix.

```
# Set. seed for reproducing the same sequence
set.seed(123)

# Split the data into training and validation sets
trainIndex <- createDataPartition(accidents$INJURY, p = 0.6, list = FALSE)
train_data <- accidents[trainIndex, ]
validation_data <- accidents[-trainIndex, ]

# Run a Naive Bayes classifier on the training set
nb_model <- naiveBayes(INJURY ~ WEATHER_R + TRAF_CON_R, data = train_data)

# Use the model to predict on the validation set
predictions <- predict(nb_model, newdata = validation_data)

# Create the confusion matrix
confusionMatrix(as.factor(validation_data$INJURY),predictions)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   NO  YES
##        NO  1294 6994
##        YES 1064 7520
##
##                Accuracy : 0.5224
##                  95% CI : (0.5148, 0.53)
##     No Information Rate : 0.8602
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.0326
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.5488
##             Specificity : 0.5181
##          Pos Pred Value : 0.1561
##          Neg Pred Value : 0.8760
##              Prevalence : 0.1398
##          Detection Rate : 0.0767
##    Detection Prevalence : 0.4912
##       Balanced Accuracy : 0.5334
##
##        'Positive' Class : NO
##
```

```
conf_matrix <- table(predictions, validation_data$INJURY)
print(conf_matrix)
```

```
##
## predictions   NO  YES
##         NO  1294 1064
##         YES 6994 7520
```

3(2) What is the overall error of the validation set?

```
error_rate <- 1 - sum(diag(conf_matrix)) / sum(conf_matrix)
print(paste("Overall error of the validation set:", error_rate))
```

```
## [1] "Overall error of the validation set: 0.477596017069701"
```

Overview: The validation set shows that we got things wrong 47% of the time and right 52% of the time. In some situations, making mistakes 47% of the time might be okay. But in other cases, especially where safety is crucial, even a 4% mistake rate could be too much.
