Assignment_3_Godey

2023-10-14

R Markdown

```
#Loading the libraries for the task
library(class)
library(caret)
```

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

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

```
library(e1071)
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(e1071)
```

```
#Loading the data set and assigning it to car_accidents variable
car_accidents <- read.csv("/Users/shivagodey/Downloads/accidentsFull.csv")
dim(car_accidents)
```

```
## [1] 42183    24
```

```
car_accidents$INJURY = ifelse(car_accidents$MAX_SEV_IR %in% c(1,2),"yes","no")
table(car_accidents$INJURY)
```

```
## 
##    no   yes 
## 20721 21462
```

```
t(t(names(car_accidents)))
```

```
##       [,1]            
##  [1,] "HOUR_I_R"      
##  [2,] "ALCHL_I"       
##  [3,] "ALIGN_I"       
##  [4,] "STRATUM_R"     
##  [5,] "WRK_ZONE"      
##  [6,] "WKDY_I_R"      
##  [7,] "INT_HWY"       
##  [8,] "LGTCON_I_R"    
##  [9,] "MANCOL_I_R"    
## [10,] "PED_ACC_R"     
## [11,] "RELJCT_I_R"    
## [12,] "REL_RWY_R"     
## [13,] "PROFIL_I_R"    
## [14,] "SPD_LIM"       
## [15,] "SUR_COND"      
## [16,] "TRAF_CON_R"    
## [17,] "TRAF_WAY"      
## [18,] "VEH_INVL"      
## [19,] "WEATHER_R"     
## [20,] "INJURY_CRASH"  
## [21,] "NO_INJ_I"      
## [22,] "PRPTYDMG_CRASH"
## [23,] "FATALITIES"    
## [24,] "MAX_SEV_IR"    
## [25,] "INJURY"
```

```
#Creating the pivot tables
sub_car_accidents <- car_accidents[1:24,c("INJURY","WEATHER_R","TRAF_CON_R")]
sub_car_accidents
```

```
##    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
```

```
pi_table1 <- ftable(sub_car_accidents)
pi_table1
```

```
##                  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
```

```
pi_table2 <- ftable(sub_car_accidents[,-1])
pi_table2
```

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

pair_1 = sum(sub_car_accidents WEATHER_R == 1 & sub_car_accidents TRAF_CON_R == 0) pair_2 = sum(sub_car_accidents WEATHER_R == 1 & sub_car_accidents TRAF_CON_R == 1) pair_3 = sum(sub_car_accidents WEATHER_R == 1 & sub_car_accidents TRAF_CON_R == 2) pair_4 = sum(sub_car_accidents WEATHER_R == 2 & sub_car_accidents TRAF_CON_R == 0) pair_5 = sum(sub_car_accidents WEATHER_R == 2 & sub_car_accidents TRAF_CON_R == 1) pair_6 = sum(sub_car_accidents WEATHER_R == 2 & sub_car_accidents TRAF_CON_R == 2)

dual_1 = sum(sub_car_accidents WEATHER_R == 1 & sub_car_accidents TRAF_CON_R == 0 & sub_car_accidents$INJURY==“YES”)

dual_2 = sum(sub_car_accidents WEATHER_R == 1 & sub_car_accidents TRAF_CON_R == 1 & sub_car_accidents$INJURY==“YES”)

dual_3 = sum(sub_car_accidents WEATHER_R == 1 & sub_car_accidents TRAF_CON_R == 2 & sub_car_accidents$INJURY==“YES”)

dual_4 = sum(sub_car_accidents WEATHER_R == 2 & sub_car_accidents TRAF_CON_R == 0 & sub_car_accidents$INJURY==“YES”)

dual_5 = sum(sub_car_accidents WEATHER_R == 2 & sub_car_accidents TRAF_CON_R == 1 & sub_car_accidents$INJURY==“YES”)

dual_6 = sum(sub_car_accidents WEATHER_R == 2 & sub_car_accidents TRAF_CON_R == 2 & sub_car_accidents$INJURY==“YES”)

prob_1 <- dual_1 / pair_1 cat(“P(INJURY = Yes | WEATHER_R = 1 and TRAF_CON_R = 0):”, prob_1, “”)

prob_2 <- dual_2 / pair_2 cat(“P(INJURY = Yes | WEATHER_R = 1 and TRAF_CON_R = 1):”, prob_2, “”)

prob_3 <- dual_3 / pair_3 cat(“P(INJURY = Yes | WEATHER_R = 1 and TRAF_CON_R = 2):”, prob_3, “”)

prob_4 <- dual_4 / pair_4 cat(“P(INJURY = Yes | WEATHER_R = 2 and TRAF_CON_R = 0):”, prob_4, “”)

prob_5 <- dual_5 / pair_5 cat(“P(INJURY = Yes | WEATHER_R = 2 and TRAF_CON_R = 1):”, prob_5, “”)

prob_6 <- dual_6 / pair_6 cat(“P(INJURY = Yes | WEATHER_R = 2 and TRAF_CON_R = 2):”, prob_6, “”)

#2.1

```
#INJURY = YES
pair_1 = pi_table1[3,1]/pi_table2[1,1]
cat("P(INJURY = Yes | WEATHER_R = 1 and TRAF_CON_R = 0):", pair_1, "\n")
```

```
## P(INJURY = Yes | WEATHER_R = 1 and TRAF_CON_R = 0): 0.6666667
```

```
pair_2 = pi_table1[3,2]/pi_table2[1,2]
cat("P(INJURY = Yes | WEATHER_R = 1 and TRAF_CON_R = 1):", pair_2, "\n")
```

```
## P(INJURY = Yes | WEATHER_R = 1 and TRAF_CON_R = 1): 0
```

```
pair_3 = pi_table1[3,3]/pi_table2[1,3]
cat("P(INJURY = Yes | WEATHER_R = 1 and TRAF_CON_R = 2):", pair_3, "\n")
```

```
## P(INJURY = Yes | WEATHER_R = 1 and TRAF_CON_R = 2): 0
```

```
pair_4 = pi_table1[4,1]/pi_table2[2,1]
cat("P(INJURY = Yes | WEATHER_R = 2 and TRAF_CON_R = 0):", pair_4, "\n")
```

```
## P(INJURY = Yes | WEATHER_R = 2 and TRAF_CON_R = 0): 0.1818182
```

```
pair_5 = pi_table1[4,2]/pi_table2[2,2]
cat("P(INJURY = Yes | WEATHER_R = 2 and TRAF_CON_R = 1):", pair_5, "\n")
```

```
## P(INJURY = Yes | WEATHER_R = 2 and TRAF_CON_R = 1): 0
```

```
pair_6 = pi_table1[4,3]/pi_table2[2,3]
cat("P(INJURY = Yes | WEATHER_R = 2 and TRAF_CON_R = 2):", pair_6, "\n")
```

```
## P(INJURY = Yes | WEATHER_R = 2 and TRAF_CON_R = 2): 1
```

```
#check the condition whether Injury = no

dual_1 = pi_table1[1,1]/pi_table2[1,1]
cat("P(INJURY = no | WEATHER_R = 1 and TRAF_CON_R = 0):", dual_1, "\n")
```

```
## P(INJURY = no | WEATHER_R = 1 and TRAF_CON_R = 0): 0.3333333
```

```
dual_2 = pi_table1[1,2]/pi_table2[1,2]
cat("P(INJURY = no | WEATHER_R = 1 and TRAF_CON_R = 1):", dual_2, "\n")
```

```
## P(INJURY = no | WEATHER_R = 1 and TRAF_CON_R = 1): 1
```

```
dual_3 = pi_table1[1,3]/pi_table2[1,3]
cat("P(INJURY = no | WEATHER_R = 1 and TRAF_CON_R = 2):", dual_3, "\n")
```

```
## P(INJURY = no | WEATHER_R = 1 and TRAF_CON_R = 2): 1
```

```
dual_4 = pi_table1[2,1]/pi_table2[2,1]
cat("P(INJURY = no | WEATHER_R = 2 and TRAF_CON_R = 0):", dual_4, "\n")
```

```
## P(INJURY = no | WEATHER_R = 2 and TRAF_CON_R = 0): 0.8181818
```

```
dual_5 = pi_table1[2,2]/pi_table2[2,2]
cat("P(INJURY = no | WEATHER_R = 2 and TRAF_CON_R = 1):", dual_5, "\n")
```

```
## P(INJURY = no | WEATHER_R = 2 and TRAF_CON_R = 1): 1
```

```
dual_6 = pi_table1[2,3]/pi_table2[2,3]
cat("P(INJURY = no | WEATHER_R = 2 and TRAF_CON_R = 2):", dual_6, "\n")
```

```
## P(INJURY = no | WEATHER_R = 2 and TRAF_CON_R = 2): 0
```

```
#Now probability of the total occurences.
```

---

else if(sub_buried WEATHER_R[i] == 1 && sub_buried TRAF_CON_R[i] == 0){ prob_injury[i] = dual_1

} else if(sub_buried WEATHER_R[i] == 1 && sub_buried TRAF_CON_R[i] == 1){ prob_injury[i] = dual_2 } else if(sub_buried WEATHER_R[i] == 1 && sub_buried TRAF_CON_R[i] == 2){ prob_injury[i] = dual_3 } else if(sub_buried WEATHER_R[i] == 2 && sub_buried TRAF_CON_R[i] == 0){ prob_injury[i] = dual_4 } else if(sub_buried WEATHER_R[i] == 2 && sub_buried TRAF_CON_R[i] == 1){ prob_injury[i] = dual_5 }

```
#cutoff is 0.5 and for 24 records
# Assuming you have calculated the conditional probabilities already, you can use
them to classify the 24 accidents.
# Let's say you have a dataframe named 'new_data' containing these 24 records.

prob_injury <- rep(0,24)
for(i in 1:24){
  print(c(sub_car_accidents$WEATHER_R[i],sub_car_accidents$TRAF_CON_R[i]))

  if(sub_car_accidents$WEATHER_R[i] == "1" && sub_car_accidents$TRAF_CON_R[i] == "
0"){
    prob_injury[i] = pair_1

  } else if (sub_car_accidents$WEATHER_R[i] == "1" && sub_car_accidents$TRAF_CON_
R[i] == "1"){
    prob_injury[i] = pair_2

  } else if (sub_car_accidents$WEATHER_R[i] == "1" && sub_car_accidents$TRAF_CON_
R[i] == "2"){
    prob_injury[i] = pair_3

  }
  else if (sub_car_accidents$WEATHER_R[i] == "2" && sub_car_accidents$TRAF_CON_R[
i] == "0"){
    prob_injury[i] = pair_4

  } else if (sub_car_accidents$WEATHER_R[i] == "2" && sub_car_accidents$TRAF_CON_
R[i] == "1"){
    prob_injury[i] = pair_5

  }
  else if(sub_car_accidents$WEATHER_R[i] == "2" && sub_car_accidents$TRAF_CON_R[i]
== "2"){
    prob_injury[i] = pair_6
  }

}
```

```
## [1] 1 0
## [1] 2 0
## [1] 2 1
## [1] 1 1
## [1] 1 0
## [1] 2 0
## [1] 2 0
## [1] 1 0
## [1] 2 0
## [1] 2 0
## [1] 2 0
## [1] 1 2
## [1] 1 0
## [1] 1 0
## [1] 1 0
## [1] 1 0
## [1] 2 0
## [1] 2 0
## [1] 2 0
## [1] 2 0
## [1] 1 0
## [1] 1 0
## [1] 2 2
## [1] 2 0
```

```
#cutoff 0.5

sub_car_accidents$prob_injury = prob_injury
sub_car_accidents$pred.prob  = ifelse(sub_car_accidents$prob_injury>0.5, "yes","n
o")


head(sub_car_accidents)
```

```
##   INJURY WEATHER_R TRAF_CON_R prob_injury pred.prob
## 1    yes         1          0   0.6666667       yes
## 2     no         2          0   0.1818182        no
## 3     no         2          1   0.0000000        no
## 4     no         1          1   0.0000000        no
## 5     no         1          0   0.6666667       yes
## 6    yes         2          0   0.1818182        no
```

#Compute manually the naive Bayes conditional probability of an injury given WEATHER_R = 1 and TRAF_CON_R = 1.

```
IY = pi_table1[3,2]/pi_table2[1,2]
I = (IY * pi_table1[3, 2]) / pi_table2[1, 2]
cat("P(INJURY = Yes | WEATHER_R = 1 and TRAF_CON_R = 1):", IY, "\n")
```

```
## P(INJURY = Yes | WEATHER_R = 1 and TRAF_CON_R = 1): 0
```

```
IN = pi_table1[1,2]/pi_table2[1,2]
N = (IY * pi_table1[3, 2]) / pi_table2[1, 2]
cat("P(INJURY = no | WEATHER_R = 1 and TRAF_CON_R = 1):", IN, "\n")
```

```
## P(INJURY = no | WEATHER_R = 1 and TRAF_CON_R = 1): 1
```

#2.4

```
new_b <- naiveBayes(INJURY ~ TRAF_CON_R + WEATHER_R,
                 data = sub_car_accidents)

new_car_accidents <- predict(new_b, newdata = sub_car_accidents,type = "raw")
sub_car_accidents$nbpred.prob <- new_car_accidents[,2]


new_c <- train(INJURY ~ TRAF_CON_R + WEATHER_R,
      data = sub_car_accidents, method = "nb")
```

```
## Warning: model fit failed for Resample09: usekernel=FALSE, fL=0, adjust=1 Error
in NaiveBayes.default(x, y, usekernel = FALSE, fL = param$fL, ...) :
##   Zero variances for at least one class in variables: TRAF_CON_R
```

```
## Warning: model fit failed for Resample15: usekernel=FALSE, fL=0, adjust=1 Error
in NaiveBayes.default(x, y, usekernel = FALSE, fL = param$fL, ...) :
##   Zero variances for at least one class in variables: TRAF_CON_R
```

```
## Warning: model fit failed for Resample16: usekernel=FALSE, fL=0, adjust=1 Error
in NaiveBayes.default(x, y, usekernel = FALSE, fL = param$fL, ...) :
##   Zero variances for at least one class in variables: TRAF_CON_R, WEATHER_R
```

```
## Warning: model fit failed for Resample18: usekernel=FALSE, fL=0, adjust=1 Error
in NaiveBayes.default(x, y, usekernel = FALSE, fL = param$fL, ...) :
##   Zero variances for at least one class in variables: TRAF_CON_R
```

```
## Warning: model fit failed for Resample20: usekernel=FALSE, fL=0, adjust=1 Error
in NaiveBayes.default(x, y, usekernel = FALSE, fL = param$fL, ...) :
##   Zero variances for at least one class in variables: TRAF_CON_R
## Warning: model fit failed for Resample21: usekernel=FALSE, fL=0, adjust=1 Error
in NaiveBayes.default(x, y, usekernel = FALSE, fL = param$fL, ...) :
##   Zero variances for at least one class in variables: TRAF_CON_R
```

```
## Warning: model fit failed for Resample22: usekernel=FALSE, fL=0, adjust=1 Error
in NaiveBayes.default(x, y, usekernel = FALSE, fL = param$fL, ...) :
##   Zero variances for at least one class in variables: TRAF_CON_R, WEATHER_R
```

```
## Warning: model fit failed for Resample23: usekernel=FALSE, fL=0, adjust=1 Error
in NaiveBayes.default(x, y, usekernel = FALSE, fL = param$fL, ...) :
##   Zero variances for at least one class in variables: TRAF_CON_R
```

```
## Warning: model fit failed for Resample24: usekernel=FALSE, fL=0, adjust=1 Error
in NaiveBayes.default(x, y, usekernel = FALSE, fL = param$fL, ...) :
##   Zero variances for at least one class in variables: TRAF_CON_R
```

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

```
predict(new_c, newdata = sub_car_accidents[,c("INJURY", "WEATHER_R", "TRAF_CON_
R")])
```

```
##  [1] yes no  no  yes yes no  no  yes no  no  no  yes yes yes yes yes no  no  no
## [20] no  yes yes no  no
## Levels: no yes
```

```
predict(new_c, newdata = sub_car_accidents[,c("INJURY", "WEATHER_R", "TRAF_CON_
R")],
                                    type = "raw")
```

```
##  [1] yes no  no  yes yes no  no  yes no  no  no  yes yes yes yes yes no  no  no
## [20] no  yes yes no  no
## Levels: no yes
```

#Let us now return to the entire dataset. Partition the data into training (60%) and validation (40%). 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.What is the overall error of the validation set?

```
accident = car_accidents[c(-24)]

set.seed(1)
acc.index = sample(row.names(accident), 0.6*nrow(accident)[1])
valid.index = setdiff(row.names(accident), acc.index)


acc.df = accident[acc.index,]
valid.df= accident[valid.index,]

dim(acc.df)
```

```
## [1] 25309    24
```

```
dim(valid.df)
```

```
## [1] 16874    24
```

```
norm.values <- preProcess(acc.df[,], method = c("center", "scale"))
acc.norm.df <- predict(norm.values, acc.df[, ])
valid.norm.df <- predict(norm.values, valid.df[, ])

levels(acc.norm.df)
```

```
## NULL
```

```
class(acc.norm.df$INJURY)
```

```
## [1] "character"
```

```
acc.norm.df$INJURY <- as.factor(acc.norm.df$INJURY)

class(acc.norm.df$INJURY)
```

```
## [1] "factor"
```

```
nb_model <- naiveBayes(INJURY ~ WEATHER_R + TRAF_CON_R, data = acc.norm.df)

predictions <- predict(nb_model, newdata = valid.norm.df)

#Ensure that factor levels in validation dataset match those in training dataset
valid.norm.df$INJURY <- factor(valid.norm.df$INJURY, levels = levels(acc.norm.df$I
NJURY))

# Show the confusion matrix
confusionMatrix(predictions, valid.norm.df$INJURY)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   no  yes
##        no  1285 1118
##        yes 6934 7537
##
##                Accuracy : 0.5228
##                  95% CI : (0.5152, 0.5304)
##     No Information Rate : 0.5129
##     P-Value [Acc > NIR] : 0.005162
##
##                   Kappa : 0.0277
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.15635
##             Specificity : 0.87083
##          Pos Pred Value : 0.53475
##          Neg Pred Value : 0.52083
##              Prevalence : 0.48708
##          Detection Rate : 0.07615
##    Detection Prevalence : 0.14241
##       Balanced Accuracy : 0.51359
##
##        'Positive' Class : no
##
```

```
# Calculate the overall error rate
error_rate <- 1 - sum(predictions == valid.norm.df$INJURY) / nrow(valid.norm.df)
error_rate
```

```
## [1] 0.4771838
```

#Summary

When an accident is recorded for the first time and no more information is provided, it is presumed that injuries may occur (INJURY = Yes). In order to appropriately represent the greatest degree of harm in the accident, designated as MAX_SEV_IR, this assumption has been made. According to the instructions, there is likely some degree of harm if MAX_SEV_IR equals 1 or 2 (harm = Yes). Conversely, if MAX_SEV_IR = 0, it means that there isn't an inferred injury (INJURY = No). Therefore, when there is a dearth of more information regarding the accident, it is deemed prudent to infer the presence of some degree of harm until fresh evidence demonstrates otherwise. In all, there are "20721 NO and yes are 21462."

The following actions were done in order to create a new data frame with 24 records and just 3 variables (traffic, weather, and injury): made a pivot table with the following variables: traffic, weather, and injury. In this phase, the data were arranged into these particular columns in a tabular format. Dropped the variable Injury: Since it wasn't required for the analysis that followed, the variable Injury was eliminated from the data frame. Bayes probabilities were computed. - For every one of the first 24 entries in the data frame, the probability of an injury was estimated using Bayes probabilities.

Categorized accidents using a cutoff of 0.5. - The probabilities obtained in Step 3 were used to categorize each accident as either likely to result in an injury or not likely, based on a 0.5 cutoff threshold. We computed the naive bayes conditional probability of injury with given attributes WEATHER_R = 1 and TRAF_CON_R =1. The results are as follows. -If INJURY = YES, the probability is 0. -If INJURY - NO , the probability is 1.

The Naive Bayes model's predictions and the exact Bayes classification have the following results: [1] yes no no yes yes no no yes no no no yes yes yes yes yes no no no no [21] yes yes no no Levels: no yes [1] yes no no yes yes no no yes no no no yes yes yes yes yes no no no no [21] yes yes no no Levels: no yes

In this context, the records are categorized as either "yes" or "no." The key observation is that both categories have the same values at certain positions, indicating that the ranking or order of observations is consistent between the two classifications. This suggests that both classifications assign similar importance to the factors and have a shared understanding of the data. In the next step, the analysis plans to include the entire dataset and divide it into two sets: a training set (comprising 60% of the data) and a validation set (40% of the data). The objective is to develop a model that can predict future accidents, including those involving new or unseen data. To achieve this, the model will be trained using the training data following the dataset split. The evaluation of the model's performance and its ability to predict future accidents will be based on the entire dataset, and metrics like accuracy, precision, recall, and F1-score will be used for a comprehensive assessment. -Validation set: This set is used to valid the data in it by using reference as training dataset so that we can know how well our model is trained when we give the unknown data(new data). It would classify the validation set by considering the training set.

Normalising the data comes next, once the data frame has been segmented. By guaranteeing that every segment is represented as a single row, this normalisation technique enables more precise decision-making. It is crucial that the qualities being analysed have constant levels and are either integer or numeric values in order to guarantee the validity of comparisons. In addition to preventing mistakes in the analysis process, this uniformity in attribute levels and data types guarantees that the operations performed on the data produce accurate and significant findings for decision-making.

Printing the classes: The classifications can be printed or displayed for additional research or reporting. They show whether or not each mishap is likely to result in an injury. These procedures imply that you have run a statistical analysis utilising the given factors (traffic and weather) to forecast the possibility of harm in accidents, and you have then classified these events using a 0.5 probability cutoff. In a variety of situations, this knowledge can be helpful for risk assessment and decision-making.

You noted in your investigation various variations in the precise Bayes and Naive Bayes techniques' classifications in the "Yes" category. The Naive Bayes classifier's assumption of independence, which might not always hold true, may be the cause of this disparity. When accurate probabilities and conditional dependencies are important, the exact Bayes technique is thought to be better; but, for bigger datasets, it may be computationally intensive.

Additionally, you mentioned that the overall error rate for the validation set is approximately 0.47 when expressed as a decimal. This suggests that the Naive Bayes classifier performs quite well and accurately on this dataset. Here are the confusion matrix and statistics for your classification model: Accuracy: The accuracy of your model is 0.5, indicating that 50% of the predictions are correct. Sensitivity: Sensitivity, also known as true positive rate or recall, is 0.15635. This means that your model correctly identifies positive cases (e.g., injuries) 15.635% of the time. Specificity: Specificity is 0.8708, indicating that your model correctly identifies negative cases (e.g., no injuries) 87.08% of the time. In summary, your model appears to perform reasonably well, but it may have limitations in accurately predicting injuries, especially for positive cases. The Naive Bayes method is effective, but it simplifies the assumption of independence between variables, which may not always hold true. The specific results and their implications should be considered in the context of your specific dataset and objectives.