Assignment_3

Durga Chowdary

2023-10-15

#Summary

In this analysis, accidents were classified as either "yes" or "no" based on specific criteria, revealing consistent patterns in both categories. The dataset was divided into a training set (60%) and a validation set (40%) to develop a predictive model for future accidents. The model's performance was evaluated using metrics such as accuracy, precision, recall, and F1-score on the entire dataset. The Naive Bayes method, while effective, may have limitations due to its assumption of independence between variables. The model showed 50% accuracy, correctly identifying injuries 15.635% of the time and no injuries 87.08% of the time. Consideration of these results in the context of the dataset and objectives is essential.

```r
#Loading the libraries that are required for the task
library(class)
library(caret)
```

```
## Loading required package: ggplot2

## Loading required package: lattice
```

```r
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
```

```r
#Loading the data set and assigning it to buried variable.
AccidentsFull <- read.csv("accidentsFull.csv")
dim(AccidentsFull)
```

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

```r
AccidentsFull$INJURY = ifelse(AccidentsFull$MAX_SEV_IR %in% c(1,2),"yes","no")
table(AccidentsFull$INJURY) # as yes is greater then no
```

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

```r
t(t(names(AccidentsFull)))
```

```
##       [,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"
```

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

```
##    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
```

```r
pi_table1 <- ftable(sub_AccidentsFull)
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
```

```r
pi_table2 <- ftable(sub_AccidentsFull[,-1])
pi_table2
```

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

#2.1

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

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

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

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

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

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

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

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

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

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

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

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

```r
#Now we check the condition whether Injury = no

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

  } else if (sub_AccidentsFull$WEATHER_R[i] == "1" && sub_AccidentsFull$TRAF_CON_R[i] == "1"){
    prob_injury[i] = pair_b

  } else if (sub_AccidentsFull$WEATHER_R[i] == "1" && sub_AccidentsFull$TRAF_CON_R[i] == "2"){
    prob_injury[i] = pair_c

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

  } else if (sub_AccidentsFull$WEATHER_R[i] == "2" && sub_AccidentsFull$TRAF_CON_R[i] == "1"){
    prob_injury[i] = pair_e

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

}
```

```
## [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_AccidentsFull$prob_injury = prob_injury
sub_AccidentsFull$pred.prob  = ifelse(sub_AccidentsFull$prob_injury>0.5, "yes","no")


head(sub_AccidentsFull)
```

```
##   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_a <- naiveBayes(INJURY ~ TRAF_CON_R + WEATHER_R,
                 data = sub_AccidentsFull)
```

```r
new_AccidentsFull <- predict(new_a, newdata = sub_AccidentsFull,type = "raw")
sub_AccidentsFull$nbpred.prob <- new_AccidentsFull[,2]


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

```
## Warning: model fit failed for Resample01: usekernel=FALSE, fL=0, adjust=1 Error in NaiveBayes.defaul
##   Zero variances for at least one class in variables: TRAF_CON_R

## Warning: model fit failed for Resample04: usekernel=FALSE, fL=0, adjust=1 Error in NaiveBayes.defaul
##   Zero variances for at least one class in variables: TRAF_CON_R

## Warning: model fit failed for Resample06: usekernel=FALSE, fL=0, adjust=1 Error in NaiveBayes.defaul
##   Zero variances for at least one class in variables: TRAF_CON_R

## Warning: model fit failed for Resample14: usekernel=FALSE, fL=0, adjust=1 Error in NaiveBayes.defaul
##   Zero variances for at least one class in variables: TRAF_CON_R, WEATHER_R

## Warning: model fit failed for Resample16: usekernel=FALSE, fL=0, adjust=1 Error in NaiveBayes.defaul
##   Zero variances for at least one class in variables: TRAF_CON_R

## Warning: model fit failed for Resample19: usekernel=FALSE, fL=0, adjust=1 Error in NaiveBayes.defaul
##   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.defaul
##   Zero variances for at least one class in variables: TRAF_CON_R

## Warning: model fit failed for Resample25: usekernel=FALSE, fL=0, adjust=1 Error in NaiveBayes.defaul
##   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.
```

```r
predict(new_c, newdata = sub_AccidentsFull[,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
```

```r
predict(new_c, newdata = sub_AccidentsFull[,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?

```r
accident = AccidentsFull[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,]
```

```r
valid.df= accident[valid.index,]

dim(acc.df)
```

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

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

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

```r
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
```

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

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

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

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

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

```r
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$INJURY))

# 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

In cases where an accident has just been reported with no additional information available, it is assumed that there may be injuries (INJURY = Yes). This assumption is made in order to accurately reflect the maximum level of injury in the accident, denoted as MAX_SEV_IR. The instructions establish that if MAX_SEV_IR equals 1 or 2, it implies there is some degree of injury (INJURY = Yes). On the other hand, if MAX_SEV_IR is equal to 0, it signifies there is no implied injury (INJURY = No). Therefore, until new information proves otherwise, it is considered wise to assume the presence of some degree of harm when there is a lack of additional information about the accident.

- There are "20721 NO and yes are 21462" in total.

To obtain a new data frame with 24 records and only 3 variables (Injury, Weather, and Traffic), the following steps were taken:

Created a pivot table with the variables Injury, Weather, and Traffic. - This step involved organizing the data in a tabular form with these specific columns.

Dropped the variable Injury. - The variable Injury was removed from the data frame because it wasn't needed for the subsequent analysis.

Calculated Bayes probabilities. - Bayes probabilities were computed to estimate the likelihood of an injury for each of the first 24 records in the data frame.

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

Summary

In cases where an accident has been reported with minimal information, it is assumed that there may be injuries (labeled as INJURY = Yes). This assumption is made to reflect the worst-case scenario for injury severity, denoted as MAX_SEV_IR. If MAX_SEV_IR equals 1 or 2, it implies some degree of injury (INJURY = Yes). Conversely, if MAX_SEV_IR is 0, it indicates no implied injury (INJURY = No). Therefore, until new information is available, it's prudent to assume the presence of some harm when details about the accident are limited.

In total, there are 20,721 cases of "No" and 21,462 cases of "Yes."

To create a new dataset with 24 records and only three variables (Injury, Weather, and Traffic), the following steps were taken:

1. Created a pivot table with the variables Injury, Weather, and Traffic, organizing the data into a table format with these specific columns.
2. Removed the Injury variable from the dataset since it wasn't needed for the subsequent analysis.
3. Calculated Bayes probabilities to estimate the likelihood of injury for the first 24 records in the dataset.
4. Categorized accidents using a 0.5 cutoff threshold based on the probabilities from step 3.

The Naive Bayes model's predictions and the exact Bayes classification both resulted in "Yes" and "No" labels for the 24 records. Notably, both classifications assigned similar importance to the factors, indicating a shared understanding of the data.

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

The next step involves using the entire dataset and splitting it into a training set (60% of the data) and a validation set (40% of the data). The goal is to develop a model that can predict future accidents, including unseen data. Model evaluation will consider the entire dataset, with metrics like accuracy, precision, recall, and F1-score used for assessment.

Normalization of data is the subsequent step to ensure consistent attribute levels and data types, enabling precise analysis and decision-making.

The classifications, indicating the likelihood of injury in accidents, can be printed or displayed for further analysis and reporting.

In summary, the analysis aims to predict injury likelihood in accidents based on Weather and Traffic variables, categorizing accidents using a 0.5 probability cutoff. Discrepancies between exact Bayes and Naive Bayes methods may arise due to the latter's independence assumption. The exact Bayes method is preferred when precise probabilities and dependencies are crucial but may be computationally intensive for larger datasets.

The error rate for the validation set is approximately 0.47 (as a decimal), suggesting that the Naive Bayes classifier performs reasonably well on this dataset.

The model's performance is as follows: - Accuracy: 50% of predictions are correct. - Sensitivity (recall): Identifies positive cases (injuries) correctly 15.635% of the time. - Specificity: Identifies negative cases (no injuries) correctly 87.08% of the time.

In summary, the model performs reasonably well, but it may have limitations in predicting injuries, especially for positive cases. The Naive Bayes method simplifies the assumption of variable independence, which may not always hold true. Consider these results in the context of your dataset and objectives.