# FML

HILDA

2023-10-15

```
library(readr)
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
```

```
accidentsFull<- read_csv("C:/Users/gandu/Downloads/accidentsFull.csv")
```

```
## Rows: 42183 Columns: 24
```

```
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## dbl (24): HOUR_I_R, ALCHL_I, ALIGN_I, STRATUM_R, WRK_ZONE, WKDY_I_R, INT_HWY...
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
View(accidentsFull)
accidentsFull$INJURY <- ifelse(accidentsFull$MAX_SEV_IR>0, "yes", "no")
head(accidentsFull)
```

```
## # A tibble: 6 × 25
##   HOUR_I_R ALCHL_I ALIGN_I STRATUM_R WRK_ZONE WKDY_I_R INT_HWY LGTCON_I_R
##      <dbl>   <dbl>   <dbl>     <dbl>    <dbl>    <dbl>   <dbl>      <dbl>
## 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
## # ℹ 17 more variables: MANCOL_I_R <dbl>, PED_ACC_R <dbl>, RELJCT_I_R <dbl>,
## #   REL_RWY_R <dbl>, PROFIL_I_R <dbl>, SPD_LIM <dbl>, SUR_COND <dbl>,
## #   TRAF_CON_R <dbl>, TRAF_WAY <dbl>, VEH_INVL <dbl>, WEATHER_R <dbl>,
## #   INJURY_CRASH <dbl>, NO_INJ_I <dbl>, PRPTYDMG_CRASH <dbl>, FATALITIES <dbl>,
## #   MAX_SEV_IR <dbl>, INJURY <chr>
```

```
#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?

#FORMING A TABLE SUBJECT TO INJURY.
injury.table <- table(accidentsFull$INJURY)
show(injury.table)
```

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

```
#Evaluating the probability of the INJURY
injury.probablilty =  scales::percent(injury.table["yes"]/(injury.table["yes"]+injury.table["n
o"]),0.01)
injury.probablilty
```

```
##      yes
## "50.88%"
```

```
##Given that our data set contains around 51% of events that ended in an accident, we can reason
ably estimate that an injury will occur because the likelihood of an injury is slightly higher.
```

```
#2. Select the first 24 records in the dataset and look only at the response (INJURY) and the tw
o predictors WEATHER_R and TRAF_CON_R.
##Create a pivot table that examines INJURY as a function of the two predictors for these 12 rec
ords.
##Use all three variables in the pivot table as rows/columns.

#THE VARIABLES' CONVERSION TO CATEGORICAL TYPE
# ASSUMED TO BE THE LAST COLUMN, IDENTIFY THE TARGET VARIABLE COLUMN INDEX
target_col_index <- dim(accidentsFull)[2]

#TRANSFORMING EVERY COLUMN INTO A FACTOR, EXCEPT THE TARGRT VARIABLE
accidentsFull[, 1:(target_col_index - 1)] <- lapply(accidentsFull[, 1:(target_col_index - 1)], a
s.factor)

#make a new subset containing just the necessary data.
new.df <- accidentsFull[1:24, c('INJURY','WEATHER_R','TRAF_CON_R')]
new.df
```

```
## # A tibble: 24 × 3
##    INJURY WEATHER_R TRAF_CON_R
##    <chr>  <fct>     <fct>
##  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
## # ℹ 14 more rows
```

```
#Using the six possible combinations of the creditors, compute the Bayes conditional probability
of an injury (injury = yes).

#P(Injury=yes|WEATHER_R = 1, TRAF_CON_R = 0) can be found:
numerator1 <- 2/3 * 3/12
denominator1 <- 3/12
prob1 <- numerator1/denominator1

#P(Injury=yes|WEATHER_R = 1, TRAF_CON_R = 0) can be found:
numerator2 <- 0 * 3/12
denominator2 <- 1/12
prob2 <- numerator2/denominator2

#P(Injury=yes| WEATHER_R = 1, TRAF_CON_R = 2) can be found as follows:
numerator3 <- 0 * 3/12
denominator3 <- 1/12
prob3 <- numerator3/denominator3

#P(Injury=yes| WEATHER_R = 2, TRAF_CON_R = 0) can be found as follows:
numerator4 <- 1/3 * 3/12
denominator4 <- 6/12

prob4 <- numerator4/denominator4

#Applying WEATHER_R = 2 and TRAF_CON_R = 1 to P(Injury=yes):
numerator5 <- 0 * 3/12
denominator5 <- 1/12
prob5 <- numerator5/denominator5

#P(Injury=yes| WEATHER_R = 2, TRAF_CON_R = 2) can be found as follows:
numerator6 <- 0 * 3/12
denominator6 <- 0
prob6 <- numerator6/denominator6
a<-c(1,2,3,4,5,6)
b<-c(prob1,prob2,prob3,prob4,prob5,prob6)
prob.df<-data.frame(a,b)
names(prob.df)<-c('Option #','Probability')
prob.df %>% mutate_if(is.numeric, round, 3)
```

```
##   Option # Probability
## 1        1       0.667
## 2        2       0.000
## 3        3       0.000
## 4        4       0.167
## 5        5       0.000
## 6        6         NaN
```

```
#In the above 12 observations there is no observation with (Injury=yes, WEATHER_R = 2, TRAF_CON_
R =2). The conditional probability here is undefined, since the denominator is zero.

#Using these probabilities and a cutoff of 0.5, classify the 24 accidents.

#UPDATING THE SUBSET WITH PROBABILITY RESULTS
new.df.prob<-new.df
head(new.df.prob)
```

```
## # A tibble: 6 × 3
##   INJURY WEATHER_R TRAF_CON_R
##   <chr>  <fct>     <fct>
## 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)

new.df.prob$PROB_INJURY <- rep(probability.injury, length.out = nrow(new.df.prob))

#INCLUDING A COLUMN FOR PREDICTION OF INJURIES WITH A 0.5 CUTOFF.
new.df.prob$PREDICT_PROB<-ifelse(new.df.prob$PROB_INJURY>.5,"yes","no")
new.df.prob
```

```
## # A tibble: 24 × 5
##    INJURY WEATHER_R TRAF_CON_R PROB_INJURY PREDICT_PROB
##    <chr>  <fct>     <fct>            <dbl> <chr>
##  1 yes    1         0                0.667 yes
##  2 no     2         0                0.167 no
##  3 no     2         1                0     no
##  4 no     1         1                0     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
## # ℹ 14 more rows
```

```
#The Naive Bayes Conditional Probability of Injury is manually computed with Weather R = 1 and T
raff Con R = 1.

#To find P(Injury=yes| WEATHER_R = 1, TRAF_CON_R =1):
#Probability of injury involved in accidents
#=(proportion of WEATHER_R =1 when Injury = yes)
#*(proportion of TRAF_CON_R =1 when Injury = yes)
#*(proportion of Injury = yes in all cases)
man.prob <- 2/3 * 0/3 * 3/12
man.prob
```

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

```
##FURTHERMORE, WE ARE COMPARING OUR CLASSIFICATIONS WITH BAYES' TO DETERMINE IF THEY ARE EQUIVAL
ENT.

## AND TO VERIFY IF THE OBSERVATIONS ARE ORDERED AND RANKED
#LOADING THE PACKAGES AND CLASSIFIER RUNNING NAIVE BAYES
library(e1071)
library(klaR)
```

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

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
library(caret)
```

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

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

```
nb<-naiveBayes(INJURY ~ ., data = new.df)
predict(nb, newdata = new.df,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
```

```
#CAREFULLY INSPECTING THE MODEL, USING THE TRAINING, AND PROJECTING FUNCTIONS.
library(caret)
x=new.df[,-3]
y=new.df$INJURY
model <- train(x,y,'nb', trControl = trainControl(method = 'cv',number=10))
```

```
## Warning: Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
```

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

```
## Warning: Setting row names on a tibble is deprecated.
```

```
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, 22, 22, 21, 22, ...
## 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.
```

```
##We can use the classification model for prediction now that it has been generated.
model.pred<-predict(model$finalModel,x)
model.pred
```

```
## $class
##  [1] yes no  no  no  no  yes no  yes no  no  no  no  yes no  yes yes no  no  no 
## [20] no  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
```

```
##MATRIX OF CONFUSION BUILDING TO VISUALIZE THE CLASSIFICATION ERRORS.
table(model.pred$class,y)
```

```
##      y
##       no yes
##   no  15   0
##   yes  0   9
```

```
#VALUE COMPARED TO RESULTS GENERATED MANUALLY
new.df.prob$PREDICT_PROB_NB<-model.pred$class
new.df.prob
```

```
## # A tibble: 24 × 6
##    INJURY WEATHER_R TRAF_CON_R PROB_INJURY PREDICT_PROB PREDICT_PROB_NB
##    <chr>  <fct>     <fct>            <dbl> <chr>        <fct>
##  1 yes    1         0                0.667 yes          yes
##  2 no     2         0                0.167 no           no
##  3 no     2         1                0     no           no
##  4 no     1         1                0     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
## # ℹ 14 more rows
```

#3. dividing the data into 40% validation and 60% training

```
#Let's go back to the complete dataset now.
set.seed(223)
train.index <- sample(c(1:dim(accidentsFull)[1]), dim(accidentsFull)[1]*0.6)
train.df <- accidentsFull[train.index,]
valid.df <- accidentsFull[-train.index,]

#1.OPERATING A naive Bayes classifier on the entire training set, displaying the confusion matri
x and using injury as the response and relevant predictors.
#DEFINING THE USED VARIABLES

library(e1071)
library(klaR)
library(caret)
vars <- c ("INJURY", "HOUR_I_R",  "ALIGN_I" ,"WRK_ZONE",  "WKDY_I_R",
        "INT_HWY",  "LGTCON_I_R", "PROFIL_I_R", "SPD_LIM", "SUR_COND",
       "TRAF_CON_R",   "TRAF_WAY",   "WEATHER_R")

nbTotal <- naiveBayes(INJURY ~ ., data = train.df)
#train.df$INJURY <- factor(train.df$INJURY)
predicted<-predict(nbTotal,valid.df[,-25])
confusionMatrix(as.factor(valid.df$INJURY),predicted)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   no  yes
##        no  8428    0
##        yes    0 8446
##
##                Accuracy : 1
##                  95% CI : (0.9998, 1)
##     No Information Rate : 0.5005
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity : 1.0000
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 1.0000
##              Prevalence : 0.4995
##          Detection Rate : 0.4995
##    Detection Prevalence : 0.4995
##       Balanced Accuracy : 1.0000
##
##        'Positive' Class : no
##
```

```
#2. TOTAL MISTAKE IN THE VALIDATION SET


actual <- factor(valid.df$INJURY, levels = c("yes", "no"))
predicted <- factor(predict(nbTotal, valid.df[, vars]), levels = c("yes", "no"))
confusionMatrix(actual, predicted, positive = "yes")
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction  yes   no
##        yes 5888 2558
##        no  5192 3236
##                                          
##                Accuracy : 0.5407         
##                  95% CI : (0.5332, 0.5483)
##     No Information Rate : 0.6566         
##     P-Value [Acc > NIR] : 1              
##                                          
##                   Kappa : 0.0811         
##                                          
##  Mcnemar's Test P-Value : <2e-16         
##                                          
##             Sensitivity : 0.5314         
##             Specificity : 0.5585         
##          Pos Pred Value : 0.6971         
##          Neg Pred Value : 0.3840         
##              Prevalence : 0.6566         
##          Detection Rate : 0.3489         
##    Detection Prevalence : 0.5005         
##       Balanced Accuracy : 0.5450         
##                                          
##        'Positive' Class : yes            
## 
```

```
ver=1-.5354
verp=scales::percent(ver,0.01)
paste("Overall Error: ",verp)
```

```
## [1] "Overall Error:  46.46%"
```

1. The forecast is “Yes” for new accident reports.
2. 5534239. The naive Bayes conditional probability of an injury with WEATHER_R = 1 and TRAF_CON_R = 1 is 0. A total of 0.477420884200545 is the error rate.