lab7_ch10.R

sathvik

2021-03-06

```
# Chapter 10
# 171EC146: Sathvik S Prabhu

library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

```
library(tm)
```

```
## Loading required package: NLP
```

```
library(e1071)
library(gmodels)

sms_raw <-read.csv("/home/sathvik/EC8/ML/Lab/Lab6/sms_spam.csv")
str(sms_raw)
```

```
## 'data.frame':    5559 obs. of  2 variables:
##  $ type: Factor w/ 2 levels "ham","spam": 1 1 1 2 2 1 1 1 2 1 ...
##  $ text: Factor w/ 5156 levels " #  in mca. But not conform.",..: 1651 2566 257 626 3308 190 357 339
```

```
sms_raw[1,1]
```

```
## [1] ham
## Levels: ham spam
```

```
sms_raw[1,2]
```

```
## [1] Hope you are having a good week. Just checking in
## 5156 Levels:  #  in mca. But not conform. ...
```

```
sms_raw[1:3,]
```

```
##   type                                               text
## 1  ham Hope you are having a good week. Just checking in
## 2  ham                             K..give back my thanks.
## 3  ham       Am also doing in cbe only. But have to pay.
```

```
sms_raw$type <- factor(sms_raw$type)
str(sms_raw)
```

```
## 'data.frame':    5559 obs. of  2 variables:
##  $ type: Factor w/ 2 levels "ham","spam": 1 1 1 2 2 1 1 1 2 1 ...
##  $ text: Factor w/ 5156 levels " #  in mca. But not conform.",..: 1651 2566 257 626 3308 190 357 339
```

```
table(sms_raw$type)
```

```
##
```

```
##  ham spam
## 4812  747
```

```
sms_corpus <- Corpus(VectorSource(sms_raw$text))
print(sms_corpus)
```

```
## <<SimpleCorpus>>
## Metadata:  corpus specific: 1, document level (indexed): 0
## Content:  documents: 5559
```

```
inspect(sms_corpus[1:3])
```

```
## <<SimpleCorpus>>
## Metadata:  corpus specific: 1, document level (indexed): 0
## Content:  documents: 3
##
## [1] Hope you are having a good week. Just checking in
## [2] K..give back my thanks.
## [3] Am also doing in cbe only. But have to pay.
```

```
sms_raw[1:3,2]
```

```
## [1] Hope you are having a good week. Just checking in
## [2] K..give back my thanks.
## [3] Am also doing in cbe only. But have to pay.
## 5156 Levels:  #  in mca. But not conform. ...
```

```
corpus_clean <- tm_map(sms_corpus, tolower)
```

```
## Warning in tm_map.SimpleCorpus(sms_corpus, tolower): transformation drops
## documents
```

```
corpus_clean <- tm_map(corpus_clean, removeNumbers)
```

```
## Warning in tm_map.SimpleCorpus(corpus_clean, removeNumbers): transformation
## drops documents
```

```
inspect(corpus_clean[1:3])
```

```
## <<SimpleCorpus>>
## Metadata:  corpus specific: 1, document level (indexed): 0
## Content:  documents: 3
##
## [1] hope you are having a good week. just checking in
## [2] k..give back my thanks.
## [3] am also doing in cbe only. but have to pay.
```

```
corpus_clean <- tm_map(corpus_clean, removeWords, stopwords())
```

```
## Warning in tm_map.SimpleCorpus(corpus_clean, removeWords, stopwords()):
## transformation drops documents
```

```
corpus_clean <- tm_map(corpus_clean, removePunctuation)
```

```
## Warning in tm_map.SimpleCorpus(corpus_clean, removePunctuation): transformation
## drops documents
```

```
corpus_clean <- tm_map(corpus_clean, stripWhitespace)
```

```
## Warning in tm_map.SimpleCorpus(corpus_clean, stripWhitespace): transformation
## drops documents
```

```r
inspect(corpus_clean[1:3])
```

```
## <<SimpleCorpus>>
## Metadata:  corpus specific: 1, document level (indexed): 0
## Content:  documents: 3
##
## [1] hope good week just checking  kgive back thanks
## [3]  also cbe pay
```

```r
sms_dtm <- DocumentTermMatrix(corpus_clean)
sms_dtm
```

```
## <<DocumentTermMatrix (documents: 5559, terms: 7921)>>
## Non-/sparse entries: 42657/43990182
## Sparsity           : 100%
## Maximal term length: 40
## Weighting          : term frequency (tf)
```

```r
# Data preparation: Training and testing sets
sms_raw_train <- sms_raw[1:4169, ]
sms_raw_test<- sms_raw[4170:5559, ]
sms_dtm_train <- sms_dtm[1:4169, ]
sms_dtm_test<- sms_dtm[4170:5559, ]
sms_corpus_train <- corpus_clean[1:4169]
sms_corpus_test<- corpus_clean[4170:5559]

prop.table(table(sms_raw_train$type))
```

```
##
##       ham      spam
## 0.8647158 0.1352842
```

```r
prop.table(table(sms_raw_test$type))
```

```
##
##       ham      spam
## 0.8683453 0.1316547
```

```r
# Visualizing text data - word clouds
#wordcloud(sms_corpus_train, min.freq = 40, random.order = FALSE)

spam <- subset(sms_raw_train, type == "spam")
ham <- subset(sms_raw_train, type == "ham")

#wordcloud(spam$text, max.words = 40, scale = c(3, 0.5))
#wordcloud(ham$text, max.words = 40, scale = c(3, 0.5))

#Data preparation - creating indicator features for frequent words
#findFreqTerms(sms_dtm_train, 5)

Dictionary <- function(x) {
  if( is.character(x) ) {
    return (x)
  }
  stop('x is not a character vector')
}
```

```r
sms_dict <- Dictionary(findFreqTerms(sms_dtm_train, 5))

sms_train <- DocumentTermMatrix(sms_corpus_train, list(dictionary = sms_dict))
sms_test<- DocumentTermMatrix(sms_corpus_test,list(dictionary = sms_dict))

convert_counts <- function(x) {
  x <- ifelse(x > 0, 1, 0)
  x <- factor(x, levels = c(0, 1), labels = c("No", "Yes"))
  return(x)
}

sms_train <- apply(sms_train, MARGIN = 2, convert_counts)
sms_test<- apply(sms_test, MARGIN = 2, convert_counts)

# Step 3 - training a model on the data

sms_classifier <- naiveBayes(sms_train, sms_raw_train$type)

# Step 4 - evaluating model performance
sms_test_pred <- predict(sms_classifier, sms_test)

CrossTable(sms_test_pred, sms_raw_test$type,
           prop.chisq = FALSE, prop.t = FALSE,
           dnn = c('predicted', 'actual'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |-------------------------|
##
##
## Total Observations in Table:  1390
##
##
##              | actual
##    predicted |       ham |      spam | Row Total |
## -------------|-----------|-----------|-----------|
##          ham |      1203 |        32 |      1235 |
##              |     0.974 |     0.026 |     0.888 |
##              |     0.997 |     0.175 |           |
## -------------|-----------|-----------|-----------|
##         spam |         4 |       151 |       155 |
##              |     0.026 |     0.974 |     0.112 |
##              |     0.003 |     0.825 |           |
## -------------|-----------|-----------|-----------|
## Column Total |      1207 |       183 |      1390 |
##              |     0.868 |     0.132 |           |
## -------------|-----------|-----------|-----------|
##
##
```

```
predicted_prob <- predict(sms_classifier, sms_test,type="raw")
head(predicted_prob)
```

```
##               ham         spam
## [1,] 9.999997e-01 2.590072e-07
## [2,] 9.999998e-01 1.857600e-07
## [3,] 9.997430e-01 2.570181e-04
## [4,] 9.999568e-01 4.317725e-05
## [5,] 8.365242e-11 1.000000e+00
## [6,] 9.995900e-01 4.099759e-04
```

```
## Chapter 10
sms_results<-read.csv("/home/sathvik/EC8/ML/Lab/Lab7/sms_results.csv")
table(sms_results$actual_type, sms_results$predict_type)
```

```
##
##         ham spam
##   ham  1202    5
##   spam   29  154
```

```
library(gmodels)
CrossTable(sms_results$actual_type, sms_results$predict_type)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## | Chi-square contribution |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  1390
##
##
##                          | sms_results$predict_type
## sms_results$actual_type |       ham |      spam | Row Total |
## ------------------------|-----------|-----------|-----------|
##                     ham |      1202 |         5 |      1207 |
##                         |    16.565 |   128.248 |           |
##                         |     0.996 |     0.004 |     0.868 |
##                         |     0.976 |     0.031 |           |
##                         |     0.865 |     0.004 |           |
## ------------------------|-----------|-----------|-----------|
##                    spam |        29 |       154 |       183 |
##                         |   109.256 |   845.876 |           |
##                         |     0.158 |     0.842 |     0.132 |
##                         |     0.024 |     0.969 |           |
##                         |     0.021 |     0.111 |           |
## ------------------------|-----------|-----------|-----------|
##            Column Total |      1231 |       159 |      1390 |
##                         |     0.886 |     0.114 |           |
```

```
## ------------------------|-----------|-----------|-----------|
##
##
```

```
library(caret)
```

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

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

```
##
## Attaching package: 'ggplot2'
```

```
## The following object is masked from 'package:NLP':
##
##     annotate
```

```
#sms_results=data.frame("predict_type"=sms_test_pred,"actual_type"=sms_raw_test$type)

confusionMatrix(sms_results$predict_type,sms_results$actual_type,positive = "spam")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  ham spam
##       ham  1202   29
##       spam    5  154
##
##                Accuracy : 0.9755
##                  95% CI : (0.966, 0.983)
##     No Information Rate : 0.8683
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8867
##
##  Mcnemar's Test P-Value : 7.998e-05
##
##             Sensitivity : 0.8415
##             Specificity : 0.9959
##          Pos Pred Value : 0.9686
##          Neg Pred Value : 0.9764
##              Prevalence : 0.1317
##          Detection Rate : 0.1108
##    Detection Prevalence : 0.1144
##       Balanced Accuracy : 0.9187
##
##        'Positive' Class : spam
##
```

```
sensitivity(sms_results$predict_type, sms_results$actual_type,
            positive = "spam")
```

```
## [1] 0.8415301
```

```
specificity(sms_results$predict_type, sms_results$actual_type,
              negative = "ham")
```

```
## [1] 0.9958575
```

```
posPredValue(sms_results$predict_type, sms_results$actual_type,
             positive = "spam")
```

```
## [1] 0.9685535
```

```
library(vcd)
```

```
## Loading required package: grid
```

```
Kappa(table(sms_results$actual_type, sms_results$predict_type))
```

```
##             value     ASE     z Pr(>|z|)
## Unweighted 0.8867 0.01909 46.45        0
## Weighted   0.8867 0.01909 46.45        0
```

```
library(irr)
```

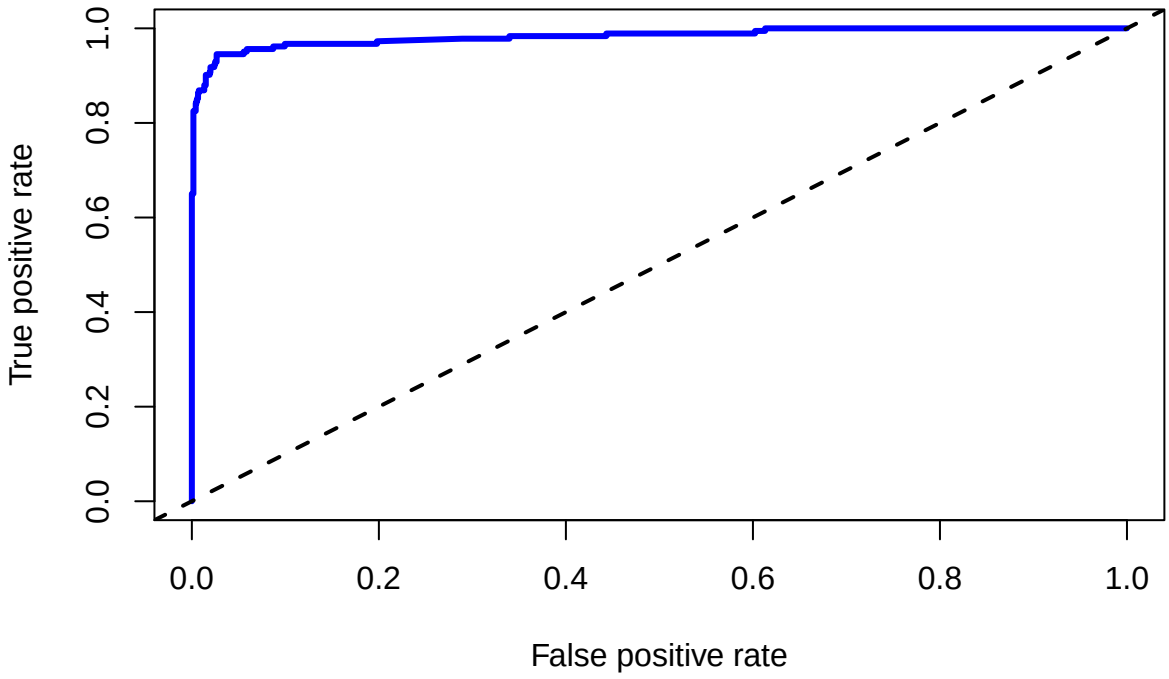
```
## Loading required package: lpSolve
```

```
kappa2(sms_results[1:2])
```

```
##  Cohen's Kappa for 2 Raters (Weights: unweighted)
##
##  Subjects = 1390
##    Raters = 2
##     Kappa = 0.887
##
##         z = 33.2
##   p-value = 0
```

```
# Visualizing performance tradeoffs
library(ROCR)
pred <- prediction(predictions = sms_results$prob_spam,labels = sms_results$actual_type)

perf <- performance(pred, measure = "tpr", x.measure = "fpr")
plot(perf, main = "ROC curve for SMS spam filter",
     col = "blue", lwd = 3)
abline(a = 0, b = 1, lwd = 2, lty = 2)
```




```
perf.auc <- performance(pred, measure = "auc")
str(perf.auc)
```

```
## Formal class 'performance' [package "ROCR"] with 6 slots
##   ..@ x.name      : chr "None"
##   ..@ y.name      : chr "Area under the ROC curve"
##   ..@ alpha.name  : chr "none"
##   ..@ x.values    : list()
##   ..@ y.values    :List of 1
##   .. ..$ : num 0.983
##   ..@ alpha.values: list()
```

```
unlist(perf.auc@y.values)
```

```
## [1] 0.9829999
```

```
# Estimating future performance
credit <- read.csv("/home/sathvik/EC8/ML/Lab/Lab5/credit.csv")

library(caret)
library(C50)
library(irr)

# Holdout method
random_ids <- order(runif(1000))
credit_train <- credit[random_ids[1:500],]
credit_validate <- credit[random_ids[501:750], ]
credit_test <- credit[random_ids[751:1000], ]

in_train <- createDataPartition(credit$default, p = 0.75,
                                list = FALSE)
```

```
credit_train <- credit[in_train, ]
head(credit_train[,1:5])
```

```
##   checking_balance months_loan_duration credit_history    purpose amount
## 1          < 0 DM                     6       critical   radio/tv   1169
## 2      1 - 200 DM                    48         repaid   radio/tv   5951
## 5          < 0 DM                    24        delayed  car (new)   4870
## 6         unknown                    36         repaid  education   9055
## 7         unknown                    24         repaid  furniture   2835
## 8      1 - 200 DM                    36         repaid car (used)   6948
```

```
credit_test <- credit[-in_train, ]
head(credit_test[,1:5])
```

```
##    checking_balance months_loan_duration credit_history   purpose amount
## 3           unknown                    12       critical education   2096
## 4            < 0 DM                    42         repaid furniture   7882
## 12           < 0 DM                    48         repaid  business   4308
## 13      1 - 200 DM                     12         repaid  radio/tv   1567
## 17          unknown                    24       critical  radio/tv   2424
## 18           < 0 DM                    30   fully repaid  business   8072
```

```
# Cross Validation
folds <- createFolds(credit$default, k = 10)
str(folds)
```

```
## List of 10
##  $ Fold01: int [1:100] 14 25 30 32 36 42 56 85 88 95 ...
##  $ Fold02: int [1:100] 4 17 20 45 47 62 75 107 110 111 ...
##  $ Fold03: int [1:100] 5 11 23 26 33 40 52 65 71 82 ...
##  $ Fold04: int [1:100] 7 16 19 34 46 55 58 78 86 91 ...
##  $ Fold05: int [1:100] 39 51 53 54 60 63 66 73 104 106 ...
##  $ Fold06: int [1:100] 1 18 28 29 49 50 70 77 81 94 ...
##  $ Fold07: int [1:100] 9 37 68 72 100 117 118 130 138 166 ...
##  $ Fold08: int [1:100] 2 8 13 21 22 41 69 76 79 116 ...
##  $ Fold09: int [1:100] 10 43 44 64 67 74 84 93 101 102 ...
##  $ Fold10: int [1:100] 3 6 12 15 24 27 31 35 38 48 ...
```

```
credit01_train <- credit[folds$Fold01, ]
credit01_test <- credit[-folds$Fold01, ]

set.seed(123)
folds <- createFolds(credit$default, k = 10)

cv_results <- lapply(folds, function(x) {
  credit_train <- credit[x, ]
  credit_test <- credit[-x, ]
  credit_train$default<-as.factor(credit_train$default)
  credit_model <- C5.0(default ~ ., data = credit_train)
  credit_pred <- predict(credit_model, credit_test)
  credit_actual <- credit_test$default
  kappa <- kappa2(data.frame(credit_actual, credit_pred))$value
  return(kappa)
})

# kappa statistics
```

```r
str(cv_results)
```

```
## List of 10
##  $ Fold01: num 0.127
##  $ Fold02: num 0.0595
##  $ Fold03: num 0.138
##  $ Fold04: num 0.242
##  $ Fold05: num 0.111
##  $ Fold06: num 0.138
##  $ Fold07: num 0.0678
##  $ Fold08: num 0.228
##  $ Fold09: num 0.0811
##  $ Fold10: num 0.19
```

```r
mean(unlist(cv_results))
```

```
## [1] 0.1382527
```