bayes_classifier_CRO.R

blaskec

2023-01-31

```
#install.packages(c("dplyr", "stringr", "rsample"))
#install.packages("rsample")
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
```

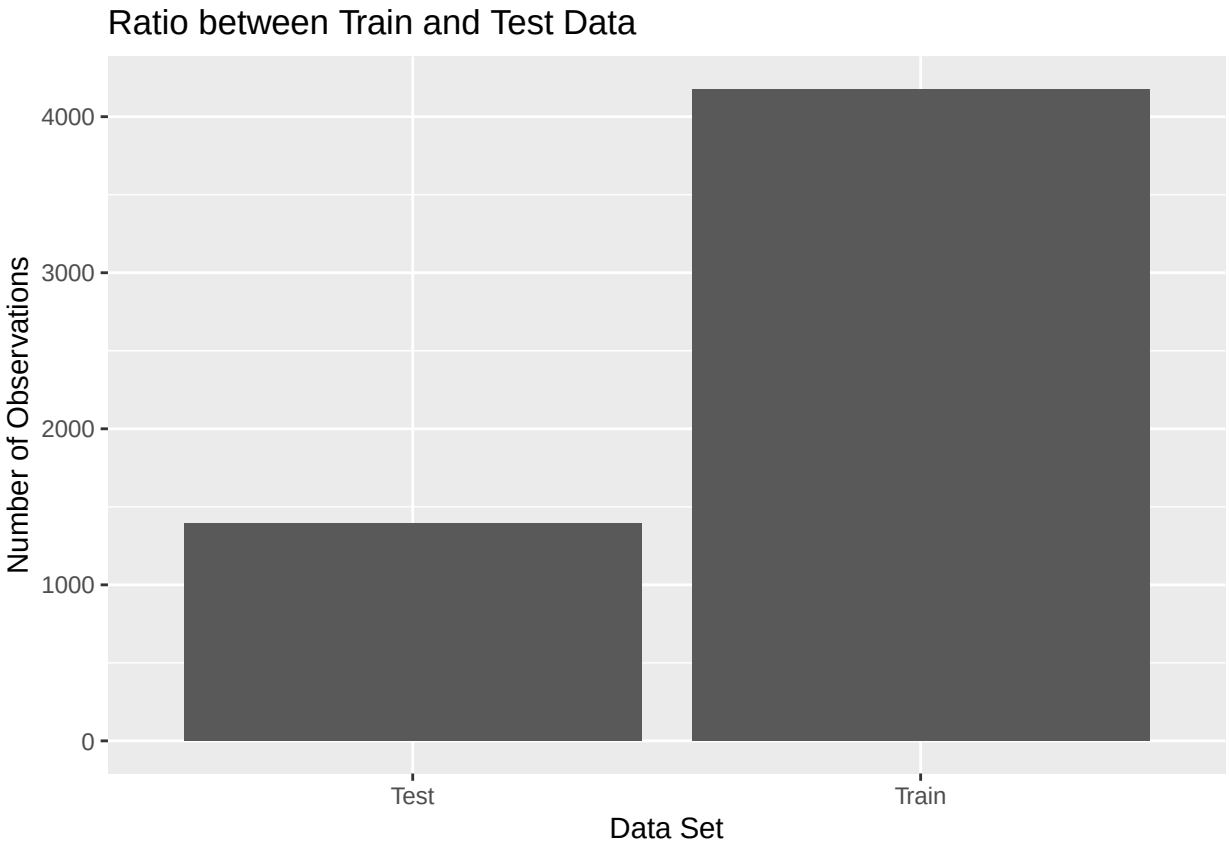
```
library("stringr")

#Čitanje podataka iz spam.csv
#Izvor podataka: SMS Spam Collection DataseT
#https://www.kaggle.com/datasets/uciml/sms-spam-collection-dataset
#Postavljanje sjemena na JMBAG (Luka Blašković)
set.seed(0303088177)

spam <- read.csv("spam.csv")

#Spajanje više stupaca u jedan
#Spaja stupce poruka u jedan dok prvi stupac preimenuje u "label"
spam <- spam %>%
  tidyr::unite(col = "msg", 2:5, sep = " ", na.rm = TRUE) %>%
  rename("label" = v1)

#label (Ham - 'Dobar/Stvarni SMS', Spam - 'Loš/Neželjena pošta SMS')
#msg (SMS message content)

#Podjela podataka na uzorke za treniranje i testiranje.
library("rsample")
#Varijabla 'strata' se koristi za provođenje stratifiranog uzorkovanja (kako bi omjer bio u redu)
split <- rsample::initial_split(spam, strata = label)
train_sample <- rsample::training(split)
test_sample <- rsample::testing(split)

dim(train_sample)
```

```
## [1] 4178    2
```

```
dim(test_sample)
```

```
## [1] 1394    2
```

```
library(ggplot2)

#Data frame sa dimenzijama svakog dataseta
data_dim <- data.frame(Data = c("Train", "Test"),
                       Dimensions = c(dim(train_sample)[1], dim(test_sample)[1]))

#Bar plot omjera train i test datasetova
ggplot(data_dim, aes(x = Data, y = Dimensions)) +
  geom_col() +
  ggtitle("Ratio between Train and Test Data") +
  xlab("Data Set") +
  ylab("Number of Observations")
```




```
# distribucija podataka (Ham ~85% | Sam ~15%) u train podacima
prop.table(table(train_sample$label))
```

```
##
##       ham      spam
## 0.8659646 0.1340354
```

```r
# distribucija podataka (Ham ~85% | Sam ~15%) u test podacima
prop.table(table(test_sample$label))
```

```
##
##       ham      spam
## 0.8658537 0.1341463
```

```r
############################################################

# Da bismo obradili poruke, prvo ih je potrebno podijeliti u pojedinačne riječi
# te dodatno očistiti podatke.
string_cleaner <- function(text_vector) {
  tx <- text_vector %>%
    #Uklanjanje svih interpunkcijskih znakova
    str_replace_all("[^[:alnum:] ]+", "") %>%
    #Zamjena svih stringova u mala slova
    str_to_lower() %>%
    #Transformacija svega što izgleda ko hyperlink u '_url_'
    str_replace_all("\\b(http|www.+)\\b", "_url_") %>%
    #Transformacija dugih sekvenci brojeva u '_longnum_'
    str_replace_all("\\b(\\d{7,})\\b", "_longnum_") %>%
    str_split(" ") #Odvajanje razmakom

  tx <- lapply(tx, function(x) x[nchar(x) > 1]) #Brisanje riječi od 1 znaka

  tx
}

train_sample <- train_sample %>%
  mutate(msg_list = string_cleaner(.$msg))

#Primjeri train skupina riječi
head(train_sample)
```

```
##    label
## 1    ham
## 5    ham
## 7    ham
## 8    ham
## 11   ham
## 15   ham
##
## 1                                                    Go until jurong point, crazy.. Available only in
## 5
## 7                                                                                  Even my brothe
## 8  As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)' has been set as your calle
## 11                                                  I'm gonna be home soon and i don't want to tal
## 15
##
## 1                                                                 go, until, jurong, point, crazy, ava
## 5
## 7                                                                                      even, my, b
```

```
## 8  as, per, your, request, melle, melle, oru, minnaminunginte, nurungu, vettam, has, been, set, as, y
## 11                                                          im, gonna, be, home, soon, and, dont, w
## 15
```

```
#1. label:ham, msg: "Fair enough, anything going on?",
#msg_list: c("fair", "enough", "anything", "going", "on")

#2. label:ham, msg: "   Hi :)finally i completed the course:)",
#msg_list: c("hi", "finally", "completed", "the", "course")

#3. label:spam, msg: "07732584351 - Rodger Burns - MSG = We tried to call you re your reply to our sms
#for a free nokia mobile",
#msg_list:c("_longnum_", "rodger", "burns", "msg", "we", "tr [...]

############################################################

#Izgradnja rječnika
#Sljedeći korak je izračunati vjerojatnosti pojavljivanja bilo koje riječi u svakoj vrsti poruke
#Izdvajanje svih jedinstvenih riječi u skupu podataka
vocab <- train_sample %>%
  select(msg_list) %>%
  unlist() %>%
  unique() %>%
  tibble::enframe(name = NULL, value = "word")

vocab
```

```
## # A tibble: 7,922 x 1
##    word
##    <chr>
##  1 go
##  2 until
##  3 jurong
##  4 point
##  5 crazy
##  6 available
##  7 only
##  8 in
##  9 bugis
## 10 great
## # ... with 7,912 more rows
```

```
#Poput prethodne funkcije, samo filtrira samo 'ham' labele i uključuje iste riječi
ham_vocab <- train_sample %>%
  filter(label == "ham") %>%
  select(msg_list) %>%
  tibble::deframe() %>%
  unlist()
head(ham_vocab)
```

```
## [1] "go"        "until"     "jurong"    "point"     "crazy"     "available"
```

```r
#Poput prethodne funkcije, samo filtrira samo 'spam' labele i uključuje iste riječi
spam_vocab <- train_sample %>%
  filter(label == "spam") %>%
  select(msg_list) %>%
  tibble::deframe() %>%
  unlist()
head(spam_vocab)
```

```
## [1] "free"  "entry" "in"    "wkly"  "comp"  "to"
```

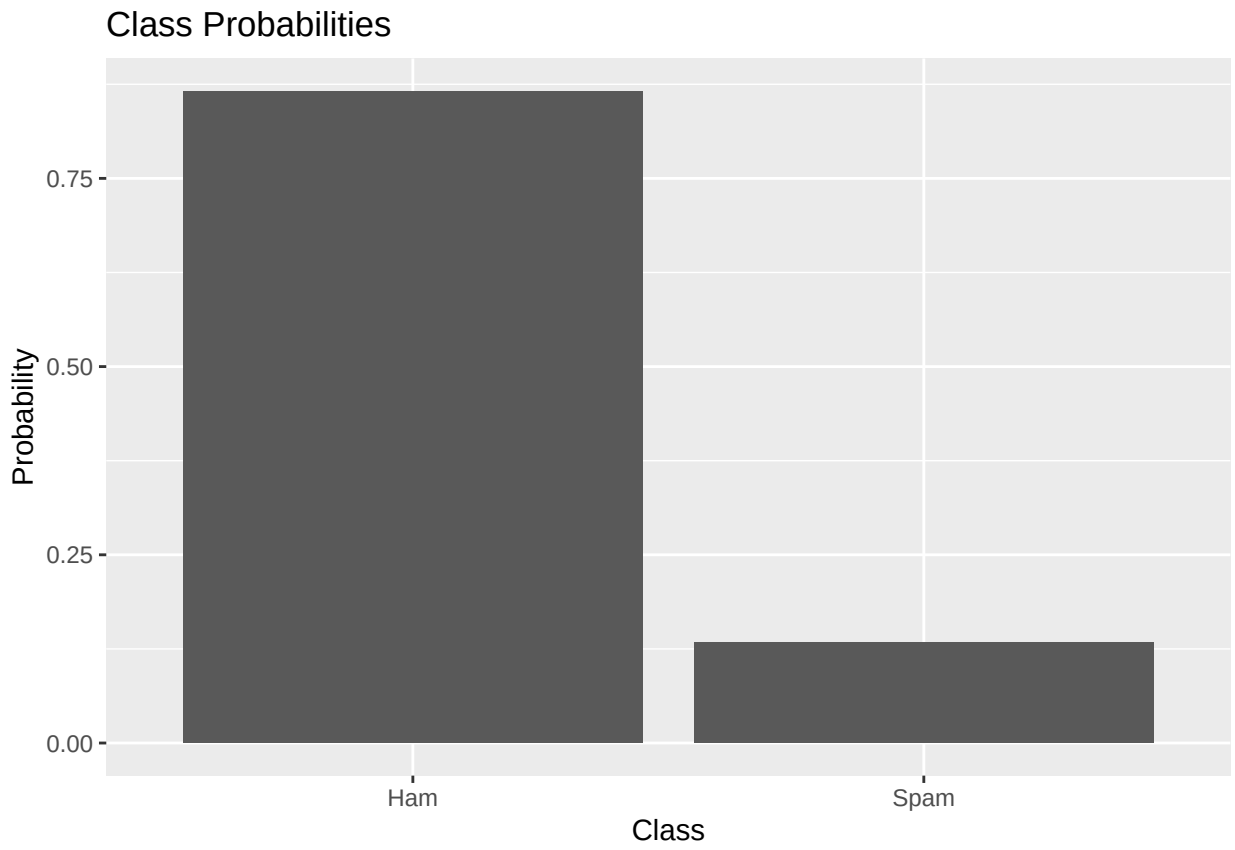
```r
#Izbroji koliko se često riječi pojavljuju u svakoj od kateogrije
#i tome pridruži s ham_vocabs, spam_vocabs
vocab <- table(ham_vocab) %>%
  tibble::as_tibble() %>%
  rename(ham_n = n) %>%
  left_join(vocab, ., by = c("word" = "ham_vocab"))

vocab <- table(spam_vocab) %>%
  tibble::as_tibble() %>%
  rename(spam_n = n) %>%
  left_join(vocab, ., by = c("word" = "spam_vocab"))

vocab
```

```
## # A tibble: 7,922 x 3
##    word      ham_n spam_n
##    <chr>     <int>  <int>
##  1 go          191     22
##  2 until        16      5
##  3 jurong        1     NA
##  4 point        12     NA
##  5 crazy         7      4
##  6 available    13      1
##  7 only        105     55
##  8 in          617     64
##  9 bugis         5     NA
## 10 great        74      8
## # ... with 7,912 more rows
```

```r
############################################################

#Sljedeći korak je, koristeći rječnik od gore, pretvoriti ove brojeve u vjerojatnosti
#budući da nam je to potrebno za samu klasifikaciju

word_n <- c("unique" = nrow(vocab),
            "ham" = length(ham_vocab),
            "spam" = length(spam_vocab))
class_probs <- prop.table(table(train_sample$label))
class_probs #Ham~0.866, Spam~0.134
```

```
##
##       ham      spam
## 0.8659646 0.1340354
```

```r
library(ggplot2)

#Izrada dataframea s podacima vjerojatnostima da je Ham ili Spam
class_probs_df <- data.frame(Class = c("Ham", "Spam"),
                             Probability = c(class_probs["ham"], class_probs["spam"]))

#Bar plot omjera vjerojatnosti (Ham podaci/Spam podaci)
ggplot(class_probs_df, aes(x = Class, y = Probability)) +
  geom_col() +
  ggtitle("Class Probabilities") +
  xlab("Class") +
  ylab("Probability")
```




```r
############################################################

#Ispod se nalazi pomoćna funkcija word_probabilities za izračunavanje
#vjerojatnosti pojavljivanja riječi u određenoj kategoriji
#Ova funkcija povezuje količinu riječi u kategoriji s ukupnom količinom riječi u kategoriji
#te Laplacian smoothing osigurava da rezultat nikada nije 0.
word_probabilities <- function(word_n, category_n, vocab_n, smooth = 1) {
  prob <- (word_n + smooth) / (category_n + smooth * vocab_n)
  prob
}
#Funkcija uzima učestalost riječi, količinu riječi u kategoriji,
#količinu ukupnih jedinstvenih riječi i vrijednost izglađivanja
```

```
#Vraća vjerojatnost pripadnosti riječi kategoriji
#Također, na početku zamjenjuje nepostojeće NA vrijednosti s 0
#te primjenjuju pomoćnu word_probabilities funkciju na svaki redak
vocab <- vocab %>%
  tidyr::replace_na(list(ham_n = 0, spam_n = 0)) %>%
  rowwise() %>%
  mutate(ham_prob = word_probabilities(
    ham_n, word_n["ham"], word_n["unique"])) %>%
  mutate(spam_prob = word_probabilities(
    spam_n, word_n["spam"], word_n["unique"])) %>%
  ungroup()

#Finalni rječnik (data frame vjerojatnosti)
vocab
```

```
## # A tibble: 7,922 x 5
##    word      ham_n spam_n  ham_prob spam_prob
##    <chr>     <int>  <int>     <dbl>     <dbl>
##  1 go          191     22 0.00351   0.00115
##  2 until        16      5 0.000311  0.000299
##  3 jurong        1      0 0.0000365 0.0000498
##  4 point        12      0 0.000237  0.0000498
##  5 crazy         7      4 0.000146  0.000249
##  6 available    13      1 0.000256  0.0000997
##  7 only        105     55 0.00194   0.00279
##  8 in          617     64 0.0113    0.00324
##  9 bugis         5      0 0.000110  0.0000498
## 10 great        74      8 0.00137   0.000449
## # ... with 7,912 more rows
```

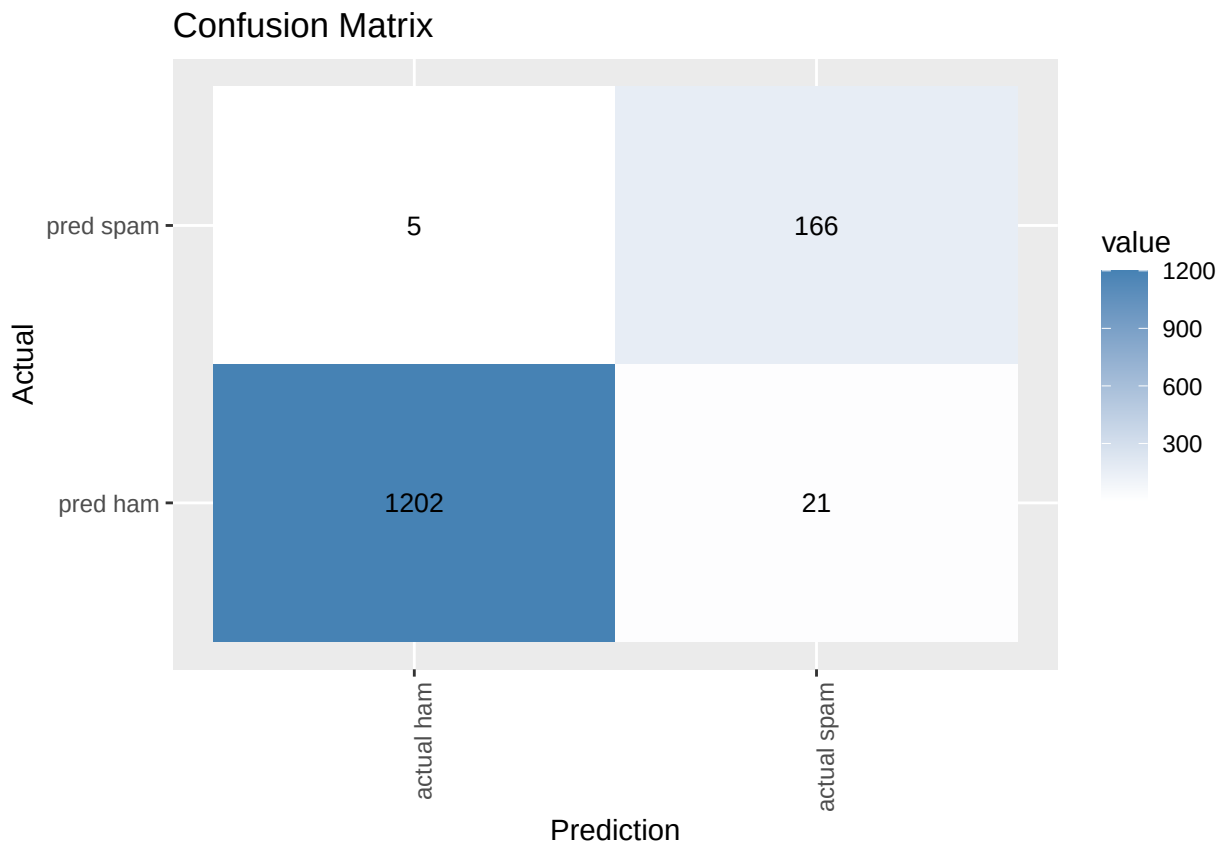
```
############################################################

#Klasifikacija
#Množenjem vjerojatnosti za sve riječi u poruci za svaku od kategorija
#također dodavanje osnovne vjerojatnosti kategorija u umnožak


#Uzima neobrađenu poruku za input te vraća klasifikaciju
classifier <- function(msg, prob_df, ham_p = 0.5, spam_p = 0.5) {
  clean_message <- string_cleaner(msg) %>% unlist()

  probs <- sapply(clean_message, function(x) {
    filter(prob_df, word == x) %>%
      select(ham_prob, spam_prob)
  })

  if (!is.null(dim(probs))) {
    ham_prob <- prod(unlist(as.numeric(probs[1, ])), na.rm = TRUE)
    spam_prob <- prod(unlist(as.numeric(probs[2, ])), na.rm = TRUE)
    ham_prob <- ham_p * ham_prob
    spam_prob <- spam_p * spam_prob

    if (ham_prob > spam_prob) {
```

```
      classification <- "ham"
    } else if (ham_prob < spam_prob) {
      classification <- "spam"
    } else {
      classification <- "unknown"
    }
  } else {
    classification <- "unknown"
  }

  classification
}

#Primjena klasifikacije na skup testnih podataka
#Funkcija uzima 4 ulaza: poruku, podatkovni okvir (vocab)
#i osnovne vjerojatnosti (classprobs["ham"] i class_probs["spam"])
#Poruka je u ovom slučaju naš testni uzorak s početka
#Poziv funkcije traje 3-5 minuta na donekle jakom računalu
spam_classification <- sapply(test_sample$msg,
                              function(x) classifier(x, vocab, class_probs["ham"],
                                                     class_probs["spam"]), USE.NAMES = FALSE)

############################################################

#Provjera performansi našeg klasifikatora
fct_levels <- c("ham", "spam", "unknown")

test_sample <- test_sample %>%
  mutate(label = factor(.$label, levels = fct_levels),
         .pred = factor(spam_classification, levels = fct_levels))

performance <- yardstick::metrics(test_sample, label, .pred)

performance
```

```
## # A tibble: 2 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy multiclass     0.981
## 2 kap      multiclass     0.917
```

```
#Na skupu podataka od 5574 redaka, rezulitralo je visokom točnošću
#od 0.984 s Cohenovom kappa vrijednošću od 0.927.
#To ukazuje na gotovo savršeno slaganje i visoku preciznost modela.

library(ggplot2)
library(reshape2)

#Confusion matrix
cm <- table(paste("actual", test_sample$label), paste("pred", test_sample$.pred))

cm_melted <- melt(cm)
ggplot(cm_melted, aes(x=Var1, y=Var2, fill=value)) +
```

```
  geom_tile() +
  geom_text(aes(label=value), color="black", size=3.5) +
  scale_fill_gradient(low = "white", high = "steelblue") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(title = "Confusion Matrix", x = "Prediction", y = "Actual")
```




```
#1202 poruka za koje je predviđeno da su stvarne, i jesu  stvarne
#5 poruka za koje je predviđeno da su neželjene, su ustvari stvarne

#21 za koje je predviđeno da su stvarne, su ustvari neželjene
#166 za koje je predviđeno da su neželjene, i jesu neželjene
```