Classification with Logistic Regression

Chris Bentz

17/01/2023

Load Packages

If the libraries are not installed yet, you need to install them using, for example, the command: install.packages("ggplot2"). For the Hrate package this is different, since it comes from github. The devtools library needs to be installed, and then the install_github() function is used.

```
library(dplyr)
library(class)
library(gmodels)
library(caret)
```

Load Data

Load data table with values per text file.

```
# load estimations from stringBase corpus
estimations.df <- read.csv("~/Github/NaLaFi/results/features.csv")
#head(features.csv)
```

Exclude subcorpora (if needed).

```
#selected <- c("natural")
#estimations.df <- estimations.df[!(estimations.df$subcorpus %in% selected), ]
```

Split into separate files by length of chunks in characters.

```
# choose number of characters
num.char = 1000
# subset data frame
estimations.df <- estimations.df[estimations.df$num.char == num.char, ]
```

Select relevant columns of the data frame, i.e. the measures to be included in classification and the ''corpus" or ''subcorpus" column.

```
estimations.subset <- estimations.df[c("corpus",
                                       "huni.chars",
                                       "hrate.chars",
                                       "ttr.chars",
                                       "rm.chars"
                                       )]
```

Remove NAs (whole row)

```
estimations.subset <- na.omit(estimations.subset)
```

Center and scale the data

```
estimations.scaled <- cbind(estimations.subset[1], scale(estimations.subset[2:ncol(estimations.subset)])
```

Create Training and Test Sets

```
# Generating seed
set.seed(1234)
# Randomly generating our training and test samples with a respective ratio of 2/3 and 1/3
datasample <- sample(2, nrow(estimations.scaled), replace = TRUE, prob = c(0.67, 0.33))
# Generate training set
train <- estimations.scaled[datasample == 1, 1:ncol(estimations.scaled)]
# Generate test set
test <- estimations.scaled[datasample == 2, 1:ncol(estimations.scaled)]
```

Building logistic regression model

The following code to run a logistic regression is adopted from https://datasciencedojo.com/blog/logistic-regression-in-r-tutorial/ (last accessed 16.01.2023).

```
# logistic regression estimation of labels
log.model <- glm(corpus ~., data = train, family = binomial(link = "logit"))
summary(log.model)
```

```
##
## Call:
## glm(formula = corpus ~ ., family = binomial(link = "logit"),
##     data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -5.0865   0.0010   0.2399   0.3453   1.8896
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.1691     0.2767   7.839 4.54e-15 ***
## huni.chars    0.6460     0.5340   1.210 0.226407
## hrate.chars  -1.9298     0.3584  -5.385 7.25e-08 ***
## ttr.chars     3.3325     0.9289   3.588 0.000334 ***
## rm.chars     -3.3707     0.4078  -8.265  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1367.48  on 1243  degrees of freedom
```

```
## Residual deviance:  498.62  on 1239  degrees of freedom
## AIC: 508.62
##
## Number of Fisher Scoring iterations: 8
```

```
# look at the training data and predicted values (y) to check the the dummy coding,
# i.e. "non-writing" is coded as 0, and "writing" as 1
head(train)
```

```
##           corpus huni.chars hrate.chars  ttr.chars   rm.chars
## 6426 non-writing -1.0475209  0.51776879 -0.4807788 -0.1577271
## 6427 non-writing -0.5761562  0.55644842 -0.4807788 -0.4741255
## 6428 non-writing -0.6689618  0.17362562 -0.4807788 -0.5620140
## 6429 non-writing -0.8685089  0.10247629 -0.4985218 -0.5445175
## 6431 non-writing -0.7845129  0.07857373 -0.5162648 -0.5270569
## 6432 non-writing -0.7134073  0.06980334 -0.5340079 -0.5884886
```

```
head(log.model$y)
```

```
## 6426 6427 6428 6429 6431 6432
##    0    0    0    0    0    0
```

Prediction

Make predictions using the logistic regression model with "trained", i.e. estimated coefficients.

```
log.predictions <- predict(log.model, test, type = "response")
head(log.predictions)
```

```
##      6430      6436      6439      6441      6451      6453
## 0.8069365 0.9411588 0.9432922 0.9460837 0.9454989 0.9186915
```

Assign a label according to the rule that the label is "writing" if the prediction probability is >0.5, else assign "non-writing".

```
log.prediction.rd <- ifelse(log.predictions > 0.5, "writing", "non-writing")
head(log.prediction.rd, 10)
```

```
##          6430          6436          6439          6441          6451
##     "writing"     "writing"     "writing"     "writing"     "writing"
##          6453          6454          6461          6464          6465
##     "writing"     "writing" "non-writing" "non-writing" "non-writing"
```

Model evaluation

```
# creating a dataframe from known (true) test labels
test.labels <- data.frame(test$corpus)
# combining predicted and known classes
class.comparison <- data.frame(log.prediction.rd, test.labels)
# giving appropriate column names
names(class.comparison) <- c("predicted", "observed")
# inspecting our results table
head(class.comparison)
```

```
##      predicted    observed
## 6430   writing non-writing
## 6436   writing non-writing
## 6439   writing non-writing
## 6441   writing non-writing
## 6451   writing non-writing
## 6453   writing non-writing
```

```
# get confusion matrix
cm <- confusionMatrix(class.comparison$predicted,
                      reference = class.comparison$observed)
print(cm)
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    non-writing writing
##   non-writing          98       6
##   writing              38     422
##
##                Accuracy : 0.922
##                  95% CI : (0.8967, 0.9427)
##     No Information Rate : 0.7589
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7682
##
##  Mcnemar's Test P-Value : 2.962e-06
##
##             Sensitivity : 0.7206
##             Specificity : 0.9860
##          Pos Pred Value : 0.9423
##          Neg Pred Value : 0.9174
##              Prevalence : 0.2411
##          Detection Rate : 0.1738
##    Detection Prevalence : 0.1844
##       Balanced Accuracy : 0.8533
##
##        'Positive' Class : non-writing
##
```

```
# get precision, recall, and f1 from the output list of confusionMatrix()
f1 <- cm[["byClass"]]["F1"]
recall <- cm[["byClass"]]["Recall"]
precision <- cm[["byClass"]]["Precision"]

# prepare data frame with results
lr.results <- data.frame(precision, recall, f1, row.names = NULL)
lr.results.rounded <- round(lr.results, 2)
print(lr.results.rounded)
```

```
##   precision recall   f1
## 1      0.94   0.72 0.82
```

Write to file.

```
write.csv(lr.results.rounded, file = paste("~/Github/NaLaFi/results/LR/results_logReg_",
                                           paste(num.char, ".csv", sep = ""),
                                         sep = ""), row.names = F)
```