Classification with Logistic Regression

Chris Bentz

16/06/2023

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 = 100
# 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(as.factor(corpus) ~., data = train, family = binomial(link = "logit"))
summary(log.model)
```

```
##
## Call:
## glm(formula = as.factor(corpus) ~ ., family = binomial(link = "logit"),
##     data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4177  -0.6925   0.4182   0.7369   3.4416
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.8213     0.1032  -7.961 1.70e-15 ***
## huni.chars    0.2363     0.2217   1.066 0.286375
## hrate.chars  -0.8564     0.2182  -3.924 8.71e-05 ***
## ttr.chars     0.8248     0.2168   3.805 0.000142 ***
## rm.chars     -3.6844     0.2020 -18.241  < 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: 2879.3  on 2078  degrees of freedom
```

```
## Residual deviance: 1894.7  on 2074  degrees of freedom
## AIC: 1904.7
##
## Number of Fisher Scoring iterations: 7
```

```
# 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
## 3362 non-writing -0.5325711  -0.2858029 -0.6159075 -0.6531764
## 3363 non-writing -0.5574374  -0.3117013 -0.6159075 -0.6531764
## 3364 non-writing -0.5552832  -0.3553763 -0.6159075 -0.6531764
## 3365 non-writing -0.5952446  -0.3242245 -0.6862393 -0.6531764
## 3367 non-writing -0.6479335  -0.4086616 -0.6159075 -0.6531764
## 3368 non-writing -0.4623995  -0.2764010 -0.5455758 -0.6531764
```

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

```
## 3362 3363 3364 3365 3367 3368
##    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)
```

```
##      3366      3372      3375      3377      3387      3389
## 0.7807584 0.7682184 0.7640438 0.7542286 0.2356900 0.2703194
```

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)
```

```
##          3366          3372          3375          3377          3387
##     "writing"     "writing"     "writing"     "writing" "non-writing"
##          3389          3390          3397          3400          3401
## "non-writing" "non-writing"     "writing" "non-writing"     "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
## 3366     writing non-writing
## 3372     writing non-writing
## 3375     writing non-writing
## 3377     writing non-writing
## 3387 non-writing non-writing
## 3389 non-writing non-writing
```

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

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    non-writing writing
##   non-writing         342      54
##   writing             130     459
##
##                Accuracy : 0.8132
##                  95% CI : (0.7874, 0.8371)
##     No Information Rate : 0.5208
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6233
##
##  Mcnemar's Test P-Value : 3.219e-08
##
##             Sensitivity : 0.7246
##             Specificity : 0.8947
##          Pos Pred Value : 0.8636
##          Neg Pred Value : 0.7793
##              Prevalence : 0.4792
##          Detection Rate : 0.3472
##    Detection Prevalence : 0.4020
##       Balanced Accuracy : 0.8097
##
##        '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      0.86   0.72 0.79
```

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)
```