Classification with Logistic Regression

Chris Bentz

23/06/2023

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 = 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(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
## -3.5560   0.0559   0.2480   0.3533   1.9465
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   1.5503     0.1195  12.976  < 2e-16 ***
## huni.chars    0.9311     0.3468   2.685  0.00725 **
## hrate.chars  -2.1246     0.2389  -8.894  < 2e-16 ***
## ttr.chars     1.1265     0.3686   3.056  0.00224 **
## rm.chars     -4.0136     0.2957 -13.574  < 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: 2702.2  on 2461  degrees of freedom
```

```
## Residual deviance: 1028.8  on 2457  degrees of freedom
## AIC: 1038.8
##
## 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
## 6426 non-writing -1.0504593   0.5265667 -0.472599 -0.1464816
## 6427 non-writing -1.0504593   0.5265667 -0.472599 -0.1464816
## 6428 non-writing -0.5708853   0.5653248 -0.472599 -0.4621617
## 6429 non-writing -0.5708853   0.5653248 -0.472599 -0.4621617
## 6431 non-writing -0.6653072   0.1817249 -0.472599 -0.5498506
## 6432 non-writing -0.8683296   0.1104312 -0.490801 -0.5323938
```

```
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
## 0.9019674 0.8938968 0.9201951 0.9093527 0.9474441 0.9669337
```

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      6453      6454      6461
## "writing" "writing" "writing" "writing" "writing" "writing" "writing" "writing"
##      6464      6465
## "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
## 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(as.factor(class.comparison$predicted),
                      reference = as.factor(class.comparison$observed))
print(cm)
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    non-writing writing
##   non-writing         221      11
##   writing              64     868
##
##                Accuracy : 0.9356
##                  95% CI : (0.9199, 0.949)
##     No Information Rate : 0.7552
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8141
##
##  Mcnemar's Test P-Value : 1.92e-09
##
##             Sensitivity : 0.7754
##             Specificity : 0.9875
##          Pos Pred Value : 0.9526
##          Neg Pred Value : 0.9313
##              Prevalence : 0.2448
##          Detection Rate : 0.1899
##    Detection Prevalence : 0.1993
##       Balanced Accuracy : 0.8815
##
##        'Positive' Class : non-writing
##
```

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

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

```
##   accuracy precision recall   f1
## 1     0.94      0.95   0.78 0.85
```

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)
```