

# Classification with Multilayer Perceptron (MLP)

Chris Bentz

18/01/2023

## 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.

```
# install the latest version of neuralnet with bug fixes: devtools::install_github("bips-hb/neuralnet")
library(neuralnet)
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("shuffled", "random")
#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)]
```

## Implement MLP classifier

This is based on code given at [http://uc-r.github.io/ann\\_classification](http://uc-r.github.io/ann_classification) (last accessed 18.01.2023)

```
# choose hidden layer structure (for adding to file name later)
hidden <- c(2,3)

set.seed(123)
# start time
start_time <- Sys.time()
classifier.mlp <- neuralnet(corpus == "writing" ~ .,
  data = train,
  hidden = hidden,
  threshold = 0.1, # defaults to 0.01
  rep = 10, # number of reps in which new initial values are used,
  # (essentially the same as a for loop)
  stepmax = 100000, # defaults to 100K
  linear.output = FALSE,
  algorithm = "rprop+", # defaults to "rprop+",
  # i.e. resilient backpropagation
  err.fct = 'ce',
  act.fct = 'logistic',
  likelihood = TRUE,
  lifesign = 'minimal')
```

## hidden: 2, 3	thresh: 0.1	rep: 1/10	steps: 1376	error: 152.36093	aic: 350.72186	bic:
## hidden: 2, 3	thresh: 0.1	rep: 2/10	steps: 40597	error: 118.64337	aic: 283.28675	bic:
## hidden: 2, 3	thresh: 0.1	rep: 3/10	steps: 5819	error: 120.05309	aic: 286.10617	bic:
## hidden: 2, 3	thresh: 0.1	rep: 4/10	steps: 1009	error: 153.80158	aic: 353.60316	bic:
## hidden: 2, 3	thresh: 0.1	rep: 5/10	steps: 75107	error: 132.98773	aic: 311.97546	bic:
## hidden: 2, 3	thresh: 0.1	rep: 6/10	steps: 3776	error: 127.65344	aic: 301.30687	bic:

```
## hidden: 2, 3    thresh: 0.1    rep: 7/10    steps: 1346    error: 153.61127    aic: 353.22254    bic
## hidden: 2, 3    thresh: 0.1    rep: 8/10    steps: 3813    error: 125.48123    aic: 296.96247    bic
## hidden: 2, 3    thresh: 0.1    rep: 9/10    steps: 9713    error: 130.35213    aic: 306.70425    bic
## hidden: 2, 3    thresh: 0.1    rep: 10/10   steps: 1407    error: 153.51019    aic: 353.02037    bic
```

```
#classifier.mlp
```

```
end_time <- Sys.time()
```

```
end_time - start_time
```

```
## Time difference of 38.2575 secs
```

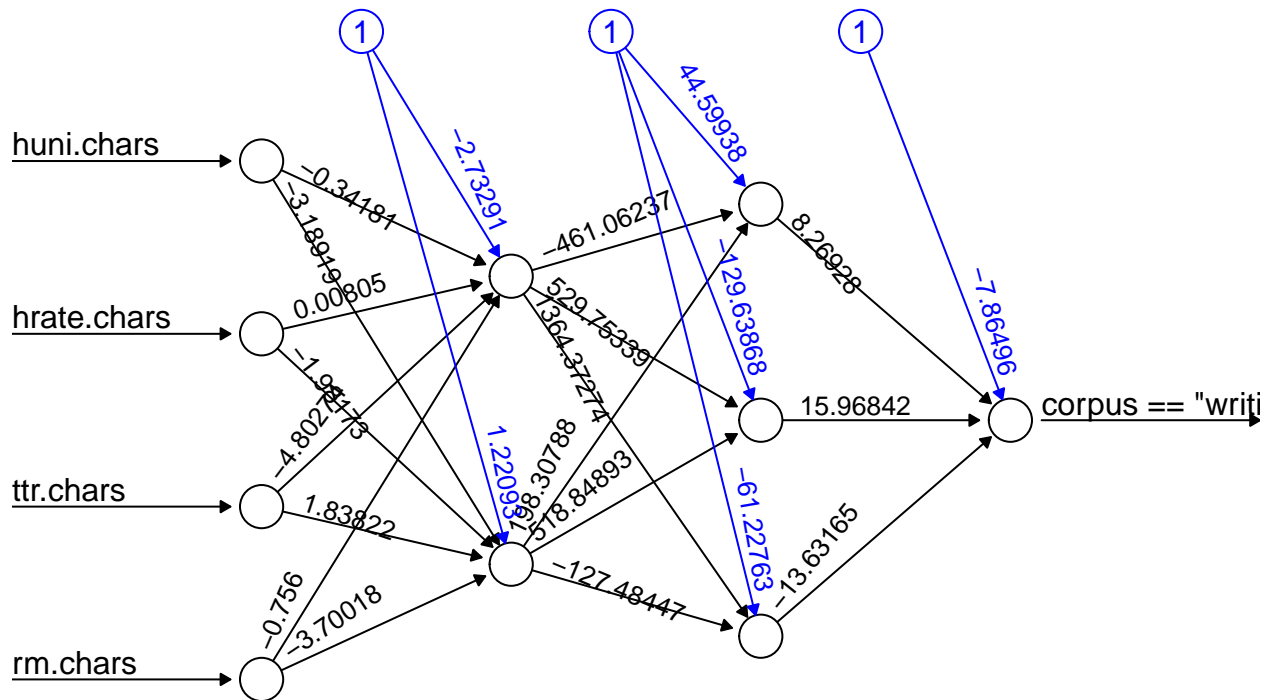
```
# results matrix (each column represents one repetition)
```

```
# classifier.mlp$result.matrix
```

## Visualize the NN

Visualize the nn with the best weights after training.

```
mlp.plot <- plot(classifier.mlp, rep = 'best')
```



Error: 118.643374 Steps: 40597

```
mlp.plot
```

```
## NULL
```

## Predict with NN

Predict response values based on the “best” repetition (epoche), i.e. the one with the lowest error in terms of cross entropy.

```

# get prediction using the predict() function
mlp.predictions <- predict(classifier.mlp, test,
                           rep = which.min(classifier.mlp$result.matrix[1,]),
                           all.units = FALSE)
# assign a label according to the rule that the label is "writing" if the prediction probability is >0
mlp.predictions.rd <- ifelse(mlp.predictions > 0.5, "writing", "non-writing")
head(mlp.predictions.rd, 10)

##      [,1]
## 6430 "non-writing"
## 6436 "writing"
## 6439 "writing"
## 6441 "writing"
## 6451 "writing"
## 6453 "writing"
## 6454 "writing"
## 6461 "non-writing"
## 6464 "non-writing"
## 6465 "non-writing"

#table(test$corpus == "non-writing", predictions[, 1] > 0.5)

```

## 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(mlp.predictions.rd, test.labels)
# giving appropriate column names
names(class.comparison) <- c("predicted", "observed")
# inspecting our results table
head(class.comparison)

##      predicted  observed
## 6430 non-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      113      2
## writing           23     426
##

```

```
##           Accuracy : 0.9557
##           95% CI : (0.9353, 0.9711)
##      No Information Rate : 0.7589
##      P-Value [Acc > NIR] : < 2.2e-16
##
##           Kappa : 0.8721
##
##  McNemar's Test P-Value : 6.334e-05
##
##           Sensitivity : 0.8309
##           Specificity : 0.9953
##      Pos Pred Value : 0.9826
##      Neg Pred Value : 0.9488
##           Prevalence : 0.2411
##      Detection Rate : 0.2004
##      Detection Prevalence : 0.2039
##      Balanced Accuracy : 0.9131
##
##      '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
mlp.results <- data.frame(precision, recall, f1, row.names = NULL)
mlp.results.rounded <- round(mlp.results, 2)
print(mlp.results.rounded)
```

```
## precision recall f1
## 1      0.98    0.83 0.9
```

Write to file.

```
write.csv(mlp.results.rounded, file = paste("~/Github/NaLaFi/results/MLP/results_MLP",
                                           paste(hidden, collapse = ""), "_",
                                           num.char, ".csv",
                                           sep = "", collapse = " "),
          row.names = F)
```