# Projektaufgaben Block 3

*Carlo Michaelis, 573479; David Hinrichs, 572347; Lukas Ruff, 572521*

*10 Januar 2017*

## 1 SPAM vs. HAM: Naive Bayes

In dieser Aufgabe beschäftigen wir uns mit dem Spam vs. Ham Klassifizierungsproblem.

### 1.1 Einlesen der Daten

```
# Read train/test data and dictionary
colFeatures <- c("docID", "wordID", "wordCount")

trainFeatures <- read.table(file = './data/train-features.txt',
                            col.names = colFeatures)
testFeatures <- read.table(file = './data/test-features.txt',
                           col.names = colFeatures)

YTrain <- read.table(file = './data/train-labels.txt', col.names = "Y")
YTest <- read.table(file = './data/test-labels.txt', col.names = "Y")

# Read dictionary
colDict <- c("wordID", "word")
dict <- read.table(file = './data/dictionary.txt', col.names = colDict)
nWords <- dim(dict)[1]
```

### 1.2 Erzeugen der Featurematrizen

```
# Build (sparse) feature matrices
XTrain <- sparseMatrix(i = trainFeatures$docID, j = trainFeatures$wordID,
                       x = trainFeatures$wordCount)
XTest <- sparseMatrix(i = testFeatures$docID, j = testFeatures$wordID,
                      x = testFeatures$wordCount)
```

### 1.3 Erzeuge Wahrscheinlichkeiten für den Naive-Bayes-Classifier

```
# Generate probabilites
indSpam <- as.logical(YTrain[[1]])
indHam <- !(YTrain[[1]])

phiSpam <- (colSums(XTrain[indSpam, ]) + 1) / (sum(XTrain[indSpam, ]) + nWords)
phiHam <- (colSums(XTrain[indHam, ]) + 1) / (sum(XTrain[indHam, ]) + nWords)
```

Zähler und Nenner wurden derart angepasst, dass für den Fall, dass keine Trainingsdaten vorliegen, für die bedingten Verteilungen der Wörter in einem Dokument a priori diskrete Gleichverteilungen mit Wahrscheinlichkeiten $\frac{1}{|V|} = \frac{1}{2500}$ angenommen werden.

## 1.4 Vorhersage auf den Testdaten

```
# Prior probabilities of message being Spam or Ham are equal:
sum(indSpam) == sum(indHam)
```

```
## [1] TRUE
```

```
# Predict test labels (using logarithm)
postSpam <- rowSums(t(t(XTest) * log(phiSpam)))
postHam <- rowSums(t(t(XTest) * log(phiHam)))
predTest <- (postSpam > postHam) * 1

# Number of errors
sum(YTest[[1]] != predTest)
```

```
## [1] 6
```

```
# SPAM indicators
dict$word[rank(-phiSpam) <= 25]
```

```
##  [1] s        email    address  order    report   mail     our
##  [8] send     program  d        one      list     name     receive
## [15] free     please   http     work     money    com      business
## [22] nt       internet day      over
## 2500 Levels: a ability able above absolutely abstract abstracts ... zur
```

```
# HAM indicators
dict$word[rank(-phiHam) <= 25]
```

```
##  [1] s           email       address     language    university
##  [6] one         information please      http        include
## [11] e           linguistic  fax         www         de
## [16] conference  english     workshop    paper       word
## [21] research    edu         abstract    papers      submission
## 2500 Levels: a ability able above absolutely abstract abstracts ... zur
```