July 7, 2015: New Version of textir

(1) The package **textir** was changed on January 31, 2014, and some of my programs will not run under the new version.
For the programs to work, you need to restore the earlier version of textir from the archives.
You can do this as follows:

- Download the file zipped file textir_1.8-8.zip from http://cran.r-project.org/bin/windows/contrib/2.14/textir_1.8-8.zip
- In your Windows R session, go to packages and to install package(s) from local zip files. Enter the zipped file you have just downloaded.
- In your R session, type library(textir)

A copy of the file textir_1.8-8.zip is also available on my website.

(2) If you are just worried about the **normalize** function (which is no longer included in the new version of **textir**), you can replace the calls to the normalize function with the statements shown below. **Lines in red** denote the necessary changes.

Chapter 9, Example 1

```
## x <- normalize(fgl[,c(4,1)])
x=fgl[,c(4,1)]
x[,1]=(x[,1]-mean(x[,1]))/sd(x[,1])
x[,2]=(x[,2]-mean(x[,2]))/sd(x[,2])

## x <- normalize(fgl[,c(1:9)])
x=fgl[,c(1:9)]
for (j in 1:9) {
x[,j]=(x[,j]-mean(x[,j]))/sd(x[,j])
}
```

Chapter 9, Example 2

```
## x <- normalize(credit[,c(2,3,4)])
x=credit[,c(2,3,4)]
x[,1]=(x[,1]-mean(x[,1]))/sd(x[,1])
x[,2]=(x[,2]-mean(x[,2]))/sd(x[,2])
x[,3]=(x[,3]-mean(x[,3]))/sd(x[,3])
```

Chapter 11, Example 1 (Program 1)

```
## covars <- normalize(fgl[,1:9],s=sdev(fgl[,1:9]))
## sd(covars) ## convince yourself that features are standardized
covars=fgl[,1:9]
for (j in 1:9) {
covars[,j]=(covars[,j]-mean(covars[,j]))/sd(covars[,j])
}
apply(covars,2,sd)
apply(covars,2,mean)
```

Chapter 11, Example 1 (Program 2)

```
## covars <- normalize(fgl[,1:9],s=sdev(fgl[,1:9]))
covars=fgl[,1:9]
for (j in 1:9) {
covars[,j]=(covars[,j]-mean(covars[,j]))/sd(covars[,j])
}
```

Chapter 11, Example 1 (Program 3)

```
## covars <- normalize(fgl[,1:9],s=sdev(fgl[,1:9]))
covars=fgl[,1:9]
for (j in 1:9) {
covars[,j]=(covars[,j]-mean(covars[,j]))/sd(covars[,j])
}
```

Chapter 11, Example 2 (Program 1)

```
## covars <- normalize(fgl[,1:9],s=sdev(fgl[,1:9]))
covars=fgl[,1:9]
for (j in 1:9) {
covars[,j]=(covars[,j]-mean(covars[,j]))/sd(covars[,j])
}
```

Chapter 11, Example 2 (Program 2)

```
## covars <- normalize(fgl[,1:9],s=sdev(fgl[,1:9]))
covars=fgl[,1:9]
for (j in 1:9) {
covars[,j]=(covars[,j]-mean(covars[,j]))/sd(covars[,j])
}
```

Chapter 11, Example 2 (Program 3)

```
## covars <- normalize(fgl[,1:9],s=sdev(fgl[,1:9]))
covars=fgl[,1:9]
for (j in 1:9) {
covars[,j]=(covars[,j]-mean(covars[,j]))/sd(covars[,j])
}
apply(covars,2,sd)
apply(covars,2,mean)
```

(3) In Chapter 11 (Multinomial Logistic Regression) and in Chapter 19 (Text as Data: Text Mining and Sentiment Analysis) I make extensive use of the function **mnlm** from the package **textir**. Instead of working with the earlier version of **textir**, you may want to use the new version (which according to its developer Matt Taddy is more efficient). However the calling sequence has changed and you should consult the new documentation of **textir**.

Below I have listed the revised R-program for the Restaurant Review data from Chapter 19 (Text as Data: Text Mining and Sentiment Analysis) that makes use of the new (January 31, 2014) version of **textir**. **Lines in red** denote the necessary changes. The changes in the output are minor (mostly scale changes) and they do not affect the conclusions.

CHAPTER 19: TEXT AS DATA: TEXT MINING AND SENTIMENT ANALYSIS

Example 1: Restaurant Reviews

```
library(textir)

data(we8there)          ## 6166 reviews and 2640 bigrams
dim(we8thereCounts)
dimnames(we8thereCounts)
dim(we8thereRatings)
we8thereRatings[1:3,]
## ratings (restaurants ordered on overall rating from 5 to 1)
as.matrix(we8thereCounts)
as.matrix(we8thereCounts)[12,400]   ## count for bigram 400 in review 12

## get to know what's in the matrix
g1=min(as.matrix(we8thereCounts)[,]) ## min count over reviews/bigrams
g2=max(as.matrix(we8thereCounts)[,]) ## max count over reviews/bigrams
g1
g2
## a certain bigram was mentioned in a certain review 13 times
hh=as.matrix(we8thereCounts)[,1000]
hh
## here we look at the frequencies of the bigram in column 1000
## the data are extremely sparce

overall=as.matrix(we8thereRatings[,5])
## overall rating

## we determine frequencies of the 2640 different bigrams
## this will take some time
nn=2640
cowords=dim(nn)
for (i in 1:nn) {
cowords[i]=sum(as.matrix(we8thereCounts)[,i])
}
cowords
cowords[7]
plot(sort(cowords,decreasing=TRUE))

## analysis per review
## we determine the frequencies of bigrams per review
## this will take some time
nn=6166
coreview=dim(nn)
for (i in 1:nn) {
coreview[i]=sum(as.matrix(we8thereCounts)[i,])
}
plot(sort(coreview,decreasing=TRUE))

## Multinomial logistic regression and fitted reduction
## we8mnlm=mnlm(we8thereCounts,overall,bins=5)
## bins: for faster inference if covariates are factors
## covariate is a factor with 5 levels
cl <- NULL
we8mnlm <- mnlm(cl,covars=overall,counts=we8thereCounts,bins=5)
```

```
we8mnlm
## we8mnlm$intercept          ## estimates of alphas
## we8mnlm$loadings           ## estimates of betas
## fitted(we8mnlm)
## as.matrix(fitted(we8mnlm))[1,]   ## fitted counts for first review

## extract coefficients
B <- coef(we8mnlm)
B
B[1,]   ## estimates of alpha
B[2,]   ## estimates of beta
mean(B[2,]==0) ## sparsity in loadings
## some big loadings in IR
B[2,order(B[2,])[1:10]]
B[2,order(-B[2,])[1:10]]

## following provides fitted multinomial probabilities
pred=predict(we8mnlm,overall,type="response")
pred[1,]    ## predicted multinomial probs for review 1
sum(pred[1,])     ## must add to one

## following predicts inverse prediction (fitted reduction)
## predinv=predict(we8mnlm,we8thereCounts,type="reduction")
predinve <- srproj(B,we8thereCounts)
predinv=predinve[,1]
predinv[1:10]     ## prints predicted ratings for first 10 reviews
plot(predinv)
plot(predinv~overall)
corr(predinv,overall)
boxplot(predinv~overall)
## procedure works. Predicted ratings increase with actual ratings
## question of cutoff. Which cutoff to use for excellent review?

## ROC curve for classification of y with p
roc <- function(p,y){
  y <- factor(y)
  n <- length(p)
  p <- as.vector(p)
  Q <- p > matrix(rep(seq(0,1,length=500),n),ncol=500,byrow=TRUE)
  fp <- colSums((y==levels(y)[1])*Q)/sum(y==levels(y)[1])
  tp <- colSums((y==levels(y)[2])*Q)/sum(y==levels(y)[2])
  plot(fp, tp, xlab="1-Specificity", ylab="Sensitivity")
  abline(a=0,b=1,lty=2,col=8)
}

c2=overall==4
c3=overall==5
c=c2+c3
min=min(predinv)
max=max(predinv)
pp=(predinv-min)/(max-min)

## plot of ROC curve
roc(p=pp, y=c)

cut <- 0
truepos <- c==1 & predinv>=cut
```

```
trueneg <- c==0 & predinv<cut
# hit-rate / sensitivity (predict good review if review is good)
sum(truepos)/sum(c==1)

sum(trueneg)/sum(c==0)
## Zero may be a good cutoff.
## Sensitivity (true positive rate) of 0.89
## False positive rate of 1 – 0.81 = 0.19
## If inverse prediction > 0, conclude overall quality rating 4 or 5.
```