Computerübung zu Multivariaten Verfahren

Klaus Schliep & Klaus Hechenbichler

18. Februar 2004

Daten

Bevor mit dem Einstieg in die eigentliche Anwendung von multivariaten statistischen Verfahren begonnen werden kann, müssen zunächst die zu untersuchenden Datensätze eingelesen werden:

```
load('C:/Windows/Desktop/carci.rda')
load('C:/Windows/Desktop/nci.rda')
load('C:/Windows/Desktop/leukemia.rdata')
load('C:/Windows/Desktop/srbct.rdata')
load('C:/Windows/Desktop/colon.rdata')
```

Eine Beschreibung dieser Datensätze befindet sich im Anhang. Alle wurden bereits einer umfangreichen Vorbehandlung (inclusive Normalisierungen) unterzogen, so daß sofort mit der statistischen Analyse begonnen werden kann. Für diese Vorarbeit bedanken wir uns bei Anne-Laure Boulesteix.

1 Clusteranalyse

1.1 Hierarchische Clusteranalyse

Hierarchische (insbesondere agglomerative) Clusteranalysen zählen zu den gängigsten multivariaten Verfahren. Die folgende Syntax zeigt, wie die verschiedenen Distanz- und Linkage-Verfahren zum Einsatz kommen:

```
y <- factor(dd[,1])
genes <- dd[,-1]

dimnames(genes)[[1]] <- y

dist.Euklid <- dist(genes, method='euclidian')
dist.Manhattan <- dist(genes, method='manhattan')
```

```
fit.euk1<-hclust(dist.Euklid, method='average')
fit.euk2<-hclust(dist.Euklid, method='complete')
fit.euk3<-hclust(dist.Euklid, method='single')

fit.euk1
fit.euk2
fit.euk3

windows()
plot(fit.euk1)
windows()
plot(fit.euk2)
windows()
plot(fit.euk3)

tree.euk1 <- cutree(fit.euk1, 2)
tree.euk2 <- cutree(fit.euk2, 2)
tree.euk3 <- cutree(fit.euk3, 2)

table(tree.euk1, y)
table(tree.euk2, y)
table(tree.euk3, y)
```

Frage: Welche Distanzmaße und welche Linkage-Verfahren führen in diesem Fall zu einem guten Ergebnis? Untersuchen Sie diese Fragestellung anhand des Vergleichs der Clusterzugehörigkeiten mit der (hier ja bekannten) Klassenvariablen!

1.2 K-Means Clustering und Prediction Around Medoids

Da diese beiden nichthierarchischen Verfahren sehr ähnlich sind, sollen sie gemeinsam zur Anwendung kommen. PAM gilt als die etwas robustere Variante, so daß der Schwerpunkt auf diesem Verfahren liegen soll.

```
fit.kmeans <- kmeans(genes, center=2)
sum(fit.kmeans$withinss/(fit.kmeans$size-1))
table(fit.kmeans$clust, y)

library(cluster)

fit.pam <- pam(dist.Euklid, k=2, diss=TRUE)

windows()
```

```
plot(fit.pam)
windows()
clusplot(fit.pam, labels=3)

table(fit.pam$clust, y)
```

Frage: Für welche Clustereinteilung mit Hilfe von k-means würden Sie sich anhand der Sum of Squares in den Clustern nach 10 Durchläufen dieses zufallsabhängigen Algorithmus entscheiden? Für welche Anzahl von Clustern ergibt sich bei PAM die beste Clusterzuordnung im Sinne von besonders homogenen Clustern? Betrachten Sie dazu die Silhouette-Plots!

1.3 Self-Organizing Maps

Self-Organizing Maps, die eine Ordnung oder Nachbarschaft zwischen verschiedenen Clustern kennen, können mit Hilfe des entsprechenden R-Packages für ein- oder zweidimensionale Karten erzeugt werden:

```
library(som)

fit.som <- som(nci.x, xdim=4, ydim=1)
summary(fit.som)
plot(fit.som)

pred <- som.project(fit.som, nci.x)
table(nci.y, pred$x)
```

Frage: Interpretieren Sie die Plots, die sich bei diesem Verfahren ergeben! Welche Cluster sind besonders eng benachbart?

2 Klassifikation

2.1 Lineare Diskriminanzanalyse

Ebenso wie die hierarchischen Verfahren für die Clusteranalysen ist die lineare Diskriminanzanalyse die gängigste Technik im Bereich der Klassifikation:

```
train.genes <- dd[1:38, ]
test.genes <- dd[39:72, ]

library(MASS)

fit.lda <- lda(y~ ., train.genes)
pred.lda <- predict(fit.lda, test.genes)
table(test.genes$y, pred.lda$class)
```

Frage: Wieso muß eine derartige Analyse mit großer Vorsicht interpretiert werden? Welche Vorarbeit sollte deshalb geleistet werden, bevor man eine LDA anwendet?

2.2 Nächste-Nachbarn-Verfahren

Nächste-Nachbarn-Verfahren zählen zu den einfachsten und intuitivsten statistischen Verfahren, da sie einzig und allein den Grad der Ähnlichkeit zwischen zwei Beobachtungen für die Klassifikation ausnutzen.

```
train.genes <- dd[1:38, -1]
test.genes <- dd[39:72, -1]
train.y <- dd[1:38, 1]
test.y <- dd[39:72, 1]

library(class)

pred.knn1 <- knn(train.genes, test.genes, train.y, k=1)
table(test.y, pred.knn1)
sum(diag(table(test.y, pred.knn1)))
```

Frage: Variieren Sie die Anzahl der betrachteten nächsten Nachbarn k! Welcher Wert führt zum geringsten Generalisierungsfehler?

2.3 Klassifikationsbäume

Klassifikationsbäume haben im Vergleich mit den bisher behandelten Verfahren den Vorteil, daß bereits implizit eine Variablenselektion durchgeführt wird, denn nur ein kleiner Teil der möglichen Genexpressionen wird zur Trennung zwischen den Klassen verwendet.

```
train <- dd[1:38, ]
test <- dd[39:72, ]

library(rpart)

fit.rpart <- rpart(y~ ., train, minsplit=5)
fit.rpart
summary(fit.rpart)
plot(fit.rpart)
text(fit.rpart, use.n=TRUE)

pred.resub <- predict(fit.rpart, type='class')
table(train$y, pred.resub)
```

```
pred.rpart <- predict(fit.rpart, test, type='class')
table(test$y, pred.rpart)
```

Frage: Welche Variablen des Datensatzes können mit Hilfe dieses Verfahrens als besonders wichtig eingestuft werden? Bestimmen Sie die Fehlklassifikationsrate sowohl anhand von Resubstitution als auch über getrennte Lern- und Teststichproben! Was fällt hier auf?

2.4 Prediction Analysis For Microarrays

Pamr ist ein Verfahren, das speziell auf die besonderen Anforderungen von Microarry-Daten, also wenig Beobachtungen bei vielen Variablen, ausgerichtet ist. Auch hier findet je nach Parameterwahl eine mehr oder weniger rigide Variablenselektion statt.

```
library(pamr)
n<-dim(dd[,-1])[2]
train <- list(x=t(dd[1:38,-1]), y=factor(dd[1:38,1]), geneid=as.character(1:n))
test <- list(x=t(dd[39:72,-1]), y=factor(dd[39:72,1]), geneid=as.character(1:n))

pamr.trained <- pamr.train(train)
pamr.trained.cv <- pamr.cv(pamr.trained, train)
pamr.plotcv(pamr.trained.cv)

pamr.confusion(pamr.trained.cv, threshold=3)
pamr.listgenes(pamr.trained, train, threshold=3)

pamr.diagnosis1 <- pamr.predict(pamr.trained, test$x, threshold=3, type='class')
table(test$y, pamr.diagnosis1)
pamr.diagnosis2 <- pamr.predict(pamr.trained, test$x, threshold=3, type='posterior')
```

Frage: Wählen sie mit Hilfe einer Kreuzvalidierung einen geeigneten Threshold-Wert und bestimmen sie den damit erzielten Generalisierungsfehler!

Anhang: Beschreibung der Datensätze

Leukemia

This 'benchmark' is included in the `R` library `golubEsets`. It contains the expression levels of 7129 genes for leukemia patients. The dataset is divided into a training set containing 27 ALL and 11 AML patients, and a test containing 20 ALL and 14 AML patients. We put the two datasets together and thresholded the data and filtered genes. This selection procedure leaves us with 3571 genes.

Colon

This data set contains the expression levels of $p = 2000$ genes for $n = 62$ patients from two classes : 22 patients are healthy patients, where as 40 have colon cancer. The genes are already selected genes. The dataset can be downloaded from `http://microarray.princeton.edu/oncology`. We standardized each array to zero mean and unit variance.

Carcinoma

This dataset comprises the expression levels of 7463 genes for 18 normal tissues and 18 carcinomas. It is available at `http://microarray.princeton.edu/oncology`. We standardized each array to zero mean and unit variance.

SRBCT

This dataset contains the expression levels of 2308 genes for 83 Single Round Blue Cells Tumor patients of 4 different types (EWS, BL, NB, and RMS). 5 samples from other tissues are available as well, but we eliminated them from the dataset. The data is publicly available at `http://www.thep.lu.se/pub/Preprints/01/lu_tp_01_06_supp.html`. The 2308 are already selected genes. We standardized each array to zero mean and unit variance across the genes.

NCI

This dataset comprises the expression levels of 5244 genes for 61 patients with 8 different tumor types. The raw data is available at `http://genome-www.stanford.edu/nci60`. We work with preprocessed data.