Clustering and classification with genomic applications

VJ Carey
CSAMA 2011, Brixen

Slides by Gregoire Pau, EMBL, from last year’s course, are liberally re-used
Road map

• artificial example on digit recognition: getting acquainted with features, cases, clustering and classification errors

• some broad principles

• clustering review

• classification review

• cross-validation, prediction error estimation, and software
> library(ElemStatLearn)
> example(zip.train)
Questions about these images that could be solved using statistical analysis or machine learning:

Given a handwriting sample:

- what digit was written?
- how can I most efficiently infer the digit written from the data provided?
Building a learning procedure for handwritten digits

- formalize what is to be learned – what is the underlying process, and what aspects of it are to be clarified through the planned analysis?

- formalize the data representation
  - cases arise in an identical fashion from the process of interest
  - features are measured on cases in a uniform way
  - departures from these uniformity conditions are recorded in experimental metadata, and may be useful as features or for stratification

- look at the data and assess its agreement with your expectations
Fact: each digit you see is a 16 x 16 digitization of a scan

- What are the cases for the data seen here? How many cases?
- What are the features for the data seen here? How many features?
Fact: \texttt{num3} is a 9 x 256 matrix, first three rows are digitizations of '0', ... last three rows are digitizations of '2'.

There is no privileged representation of features for algorithmic analysis – a different view of 9 scans follows

```r
> par(mfrow=c(2,1))
> image(t(num3), col=gray((256:0)/256),axes=FALSE, xlab="feature")
> axis(1,at=seq(0,1,len=4),labels=round(seq(0,1,len=4)*256,0))
> par(mfrow=c(1,1))
```
feature
0 85 171 256
Another more familiar representation
What do we learn about features? What about cases?

> heatmap(num3)
The failure of case clustering to recover the natural groups may be rectifiable by standardizing features to have comparable variability. Explain and evaluate:

```r
> heatmap(scale(num3))
```

Error in `hclustfun(distfun(if (symm) x else t(x)))`:
- NA/NaN/Inf in foreign function call (arg 11)

```r
> drop = which(apply(num3,2,sd) < .2)
> heatmap(scale(num3[-drop]))
```
Getting around the defaults of heatmap

```r
> wclust = function(...) hclust(..., method="ward")
> heatmap(scale(num3[, -drop]), hclustfun=wclust)
```
Explain the interest of:

```r
> sum(apply(num3,2,mad) == 0)
[1] 184
> par(las=2)
> boxplot(data.frame(num3[,1:30]))
```
Visualization after dimension reduction, with labels

```r
> m1 = prcomp(num3)
> dim(m1$x)

[1] 9 9

> m1 = prcomp(num3)
> pairs(m1$x[,1:4], col=rep(1:3,each=3), pch=19, cex=1.5)
```
Using the ExpressionSet container to represent the features and labels

```r
> NN = zip.train[zip.train[,1] %in% c(0,1,2),]
> NNMAT = t(NN[1:25,-1])
> NNFAC = factor(NN[1:25,1])
> num25 = new("ExpressionSet", exprs=NNMAT)
> num25$lab = NNFAC

> num25

ExpressionSet (storageMode: lockedEnvironment)
assayData: 256 features, 25 samples
  element names: exprs
protocolData: none
phenoData
  sampleNames: 1 2 ... 25 (25 total)
  varLabels: lab
  varMetadata: labelDescription
featureData: none
experimentData: use 'experimentData(object)'
Annotation:

  It isn't from an array, but it is a nice unified representation....
Using MLInterfaces to compare learning procedures (and some timings)

First, diagonal LDA

```r
> g4spec = xvalSpec("LOG", 4, balKfold.xvspec(4))
> dldaSca = unix.time(
+   dlda1 <- MLearn(lab~., num25, dldaI, g4spec)
+ )
> print(dldaSca)

user  system elapsed
0.961 0.071 1.039

> confuMat(dlda1)

   predicted
given 0 1 2
0 7 0 2
1 0 9 0
2 1 0 6
```
Second, feed-forward neural network

```r
> nnutSca = unix.time(
+   nnet1 <- MLearn(lab~., num25, nnetI, g4spec,
+                   size=8, decay=0.01, MaxNWts=2500, maxit=200)
+ )
>
> print(nnutSca)

          user  system elapsed
   38.806   0.306  41.379

> confuMat(nnet1)

          predicted
given 0 1 2
  0 7 0 2
  1 0 9 0
  2 0 3 4
```
With two-core system

```r
library(multicore)
g4spec = xvalSpec("LOG", 4, balKfold.xvspec(4))
dldaMT = unix.time(dlda2 <- MLearn(lab~., num25, dldaI, g4spec))
print(dldaMT)

user  system elapsed
0.061 0.017  0.802

> confuMat(dlda2)

  predicted
given 0 1 2
  0 7 0 2
  1 0 9 0
  2 1 0 6

> nnutMT = unix.time(nnet2 <- MLearn(lab~., num25, nnetI, g4spec,
+ size=8, decay=0.01, MaxNWts=2500, maxit=200))

> print(nnutMT)

user  system elapsed
24.372 0.903  27.285

> confuMat(nnet2)

  predicted
given 0 1 2
  0 7 0 2
  1 0 9 0
  2 0 3 4
```
Summary of introduction

• Multivariate data arise through digitization of images
• Joint distribution of features may be very complex
• Interactive statistical analysis can be used to reduce feature complexity
• When case labels are available, assessment of discriminative capacity of features using PCA can be straightforward
• MLearn can be used to exercise celebrated methods against the data fairly simply
  – tuning parameters must be supplied manually
  – various species of cross-validation and feature elimination are supported