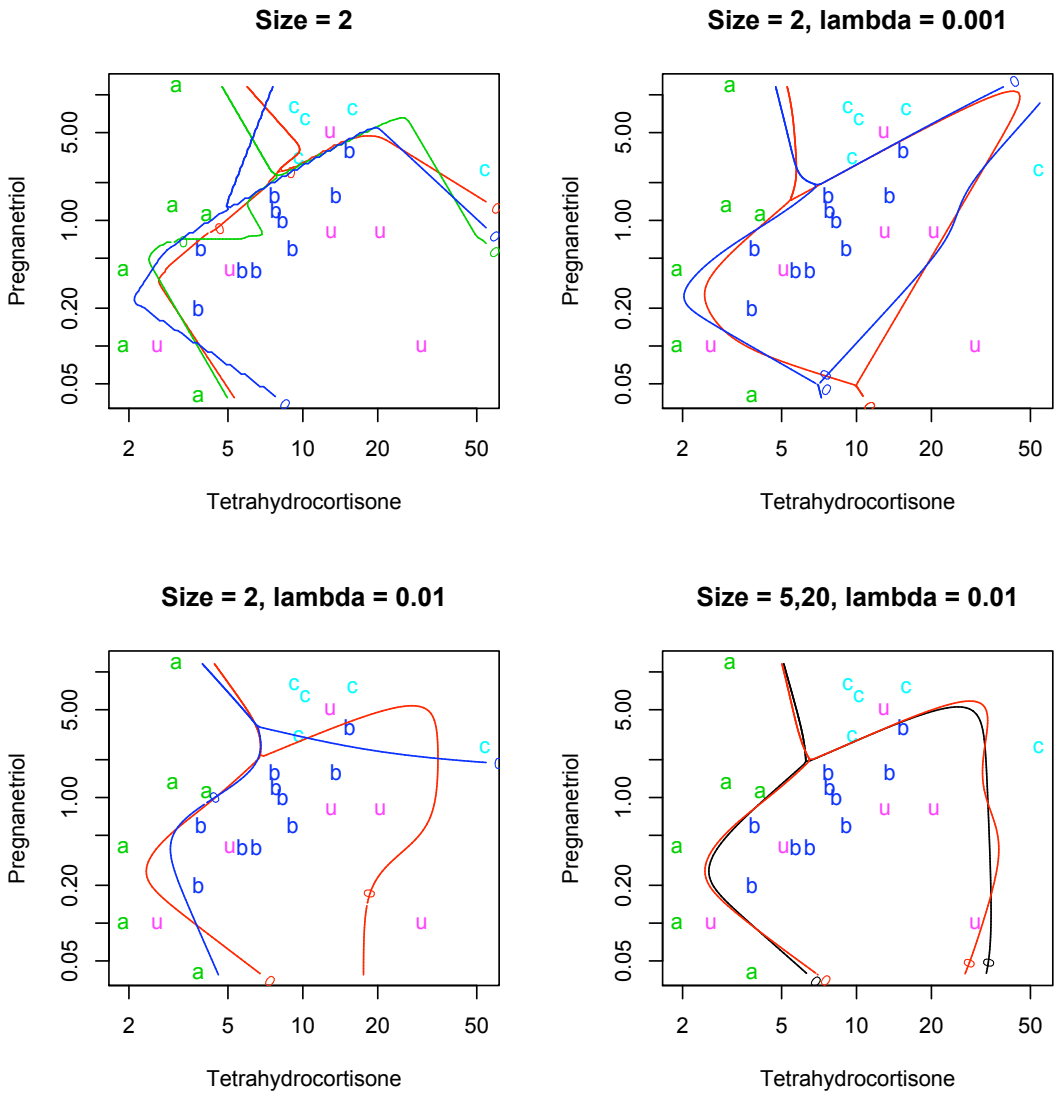
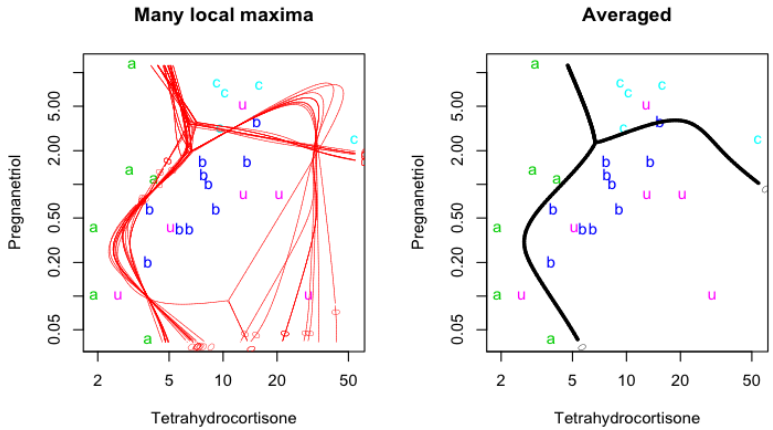
Notes

- Take-home Midterm: March 16 – March 25
- One question from ”Kernels and Ensembles” by M. Zhu
- Classification and regression trees §9.2
- Ensemble methods and random forests Zhu + §15.1-3
- *k*-means and *k*-nearest neighbour methods §13.1-3
- unsupervised learning §14.1-3

VR code for neural networks



...code for nn




```
plt.bndry <- function(size=0, decay=0, ...)
{
   cush.nn <- nnet(cush, tpi, skip=T, softmax=T, size=size,
      decay=decay, maxit=1000)
   invisible(b1(predict(cush.nn, cushT), ...))
}

b1 <- function(Z, ...)
{
   zp <- Z[,3] - pmax(Z[,2], Z[,1])
   contour(exp(xp), exp(yp), matrix(zp, np),
      add=T, levels=0, labex=0, ...)
   zp <- Z[,1] - pmax(Z[,3], Z[,2])
   contour(exp(xp), exp(yp), matrix(zp, np),
      add=T, levels=0, labex=0, ...)
}
```

...code for nn

- `Z = predict(cush.nn, cushT)`
  - uses fitted neural network to predict on a 100×100 grid of (x_1, x_2) called `cushT`
  - the grid points are called `xp` and `yp`
  - this gives a $10,000 \times 3$ matrix of probabilities
- `zp = Z[,3] - pmax(Z[,2], Z[,1] )`
  - $\Pr(3) - \max\{\Pr(2), \Pr(1)\}$
  - positive if class 3 is the highest probability, else negative
  - `matrix(zp, np)`: `np`= 100, the number of columns in the matrix `zp`
- `contour(exp(xp), exp(yp), matrix(zp, np), add=T, levels = 0)`
  - the boundary where Pr(3) is largest
- and similarly
  - `zp = Z[,1] - pmax(Z[,3], Z[,2] )`
    $\Pr(1) - \max\{\Pr(3), \Pr(2)\}$
  - positive if class 1 is the highest probability, else negative
  - `contour(exp(xp), exp(yp), matrix(zp, np), add=T, levels = 0`: the boundary where Pr(1) is largest

Support Vector Machines

- $$\min_{\beta_0,\beta} \frac{1}{2}||\beta||^2 + \gamma \sum_{i=1}^{N} \xi_i \quad s.t. \quad \xi_i \geq 0$$
- under constraints
$$\xi_i \geq 0, \quad y_i(x_i^T \beta + \beta_0) \geq 1 - \xi_i$$
- Lagrange multipliers:
$$L_P = \frac{1}{2}||\beta||^2 + \gamma \sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha_i \{y_i(x_i^T \beta + \beta_0) - (1 - \xi_i)\} - \sum_{i=1}^{N} \mu_i \xi_i$$
- minimize over β, β_0, ξ_i
- $$\hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i x_i y_i$$
- $\{x_i : \hat{\alpha}_i \neq 0\}$ support vectors

Code for Support Vector Machines
Jean-François Plante, U de M

```
library(e1071)
library(rgl)

## similar boundary drawing functions for
## neural networks etc are in Chapter 12 scripts for MASS

boundaries=function(y,b,n=100){
# y is the data, Êb the svm object
grid=expand.grid(seq(min(y[,1]),max(y[,1]),length=n),seq(min(y[,2]),max(y[,2]),length=n))
points(grid,pch=4,cex=.15,col=paste(as.character(predict(b,grid))))
}

supports=function(y,b){
# y is the data, Êb is the svm object
points(y[b$index,],cex=2)
}

# Examples of SVM
# ---------------

y=matrix(rnorm(200),ncol=2)+rep(c(0,5),each=50)
lab=rep(c("Red","Blue"),each=50)
plot(y,col=lab,pch=20)

b=svm(y,as.factor(lab),kernel="linear")
summary(b)
supports(y,b)

boundaries(y,b)
```

Example with misclassification

```
y=matrix(rnorm(200),ncol=2)+rep(c(0,2),each=50)
lab=rep(c("Red","Blue"),each=50)
plot(y,col=lab,pch=20)

b=svm(y,as.factor(lab),kernel="linear")
supports(y,b)
summary(b)
fit=as.character(b$fitted)
plot(y,col=lab,pch=20)
points(y[fit!=lab,],col=fit[fit!=lab],pch=5,cex=1.5)

boundaries(y,b)
```

Example with circles

```
norm=function(x){sqrt(sum(x^2))}
y=matrix(rnorm(200),ncol=2)
y[51:100,]=y[51:100,]+4*y[51:100,]/apply(y[51:100,],1,norm)
lab=rep(c("Red","Blue"),each=50)
plot(y,col=lab,pch=20)

b=svm(y,as.factor(lab),kernel="linear")
summary(b)
fit=as.character(b$fitted)
points(y[fit!=lab,],col=fit[fit!=lab],pch=5,cex=1.5)

boundaries(y,b)

b=svm(y,as.factor(lab),kernel="radial")
summary(b)
fit=as.character(b$fitted)
plot(y,col=lab,pch=20)
points(y[fit!=lab,],col=fit[fit!=lab],pch=5,cex=1.5)
boundaries(y,b)

plot(y,col=lab,pch=20)
points(y[fit!=lab,],col=fit[fit!=lab],pch=5,cex=1.5)
supports(y,b)

# Check new data

y2=matrix(rnorm(200),ncol=2)
y2[51:100,]=y2[51:100,]+4*y2[51:100,]/apply(y2[51:100,],1,norm)
lab=rep(c("Red","Blue"),each=50)
fit=as.character(predict(b,y2))
plot(y2,col=fit,pch=20)
points(matrix(y2[fit!=lab,],ncol=2),cex=1.2)
```

Example with many clusters

```
norm=function(x){sqrt(sum(x^2))}
y=matrix(rnorm(400),ncol=2)
y[51:100,]=y[51:100,]+3*y[51:100,]/apply(y[51:100,],1,norm)
y[101:150,]=y[101:150,]+c(4,4)
y[151:200,]=y[151:200,]+6*y[151:200,]/apply(y[151:200,],1,norm)
lab=rep(c("Red","Blue","Green","Black"),each=50)
plot(y,col=lab,pch=20)

chk=sample(1:200,20)

b=svm(y[-chk,],as.factor(lab[-chk]),kernel="radial")
summary(b)
boundaries(y,b)

supports(y,b)

fit=as.character(predict(b,y[chk,]))
plot(y[-chk,],col=lab[-chk],pch=20)
points(y[chk,],col=fit,pch=3)

# Points with wrong category
points(matrix(y[chk[fit!=lab[chk]],],ncol=2),col=lab[chk[fit!=lab[chk]]],pch=7)
```

Example with music files

```
# Real data. List of songs. Variables are based on the analysis
# of the signal of 30 seconds of music.
# Use that data to predict the type of songs.

a=read.table("music_data_17_10_05.exa",sep=" ")[,-52]

x=array(as.numeric(as.matrix(a[,2:50])),dim(a))

rem=apply(is.na(x),1,sum)
tit=a[!rem,1]
genre=a[!rem,51]
x=x[!rem,]
x=x[,apply(x,2,sd)!=0]

# Visualization

pairs(x[,sample(1:46,5)],col=lab,pch=20)
keep=sample(1:46,3); kp=paste("var",keep); plot3d(x[,keep],size=3,col=lab,xlab=kp[1],ylab=kp[

# Remove data for validation

chk=sample(1:1559,200)

# Different SVM

b=svm(x[-chk,],genre[-chk])
sum(predict(b,x[chk,])==genre[chk])

b1=svm(x[-chk,],genre[-chk],kernel="linear")
sum(predict(b1,x[chk,])==genre[chk])
```

Vary the value of C /γ

```
b1=svm(x[-chk,],genre[-chk],kernel="linear")
sum(predict(b1,x[chk,])==genre[chk])

b1=svm(x[-chk,],genre[-chk],kernel="linear",cost=10)
sum(predict(b1,x[chk,])==genre[chk])

b1=svm(x[-chk,],genre[-chk],kernel="linear",cost=.1)
sum(predict(b1,x[chk,])==genre[chk])

b1=svm(x[-chk,],genre[-chk],kernel="linear",cost=100)
sum(predict(b1,x[chk,])==genre[chk])


# Data from the book

x=matrix(scan("x.dat"),ncol=2)
y=scan("y.dat")
y=c("orange","blue")[2-y]

plot(x,col=y,pch=20)
b=svm(x,as.factor(y))
boundaries(x,b)

supports(x,b)

b=svm(x,as.factor(y),kernel="polynomial",degree=4)
boundaries(x,b)

supports(x,b)
```

Music files

```
The Garageband Feature Set
--------------------------

The data files contain features for most of the audio tracks in Garageband dataset.
These have been extracted using the approach described in [Mierswa/Morik/2005a].
More information about the Garageband dataset and the audio tracks can be
found in [Homburg/etal/2005a].

Format
------

The dataset is represented in the Yale format and can be read and processed with the
Yale package (http://yale.cs.uni-dortmund.de).

History
-------

17.10.2005
We have revised the former exampleset, which can be found under
"http://www-ai.cs.uni-dortmund.de/AUDIO/Music_features_old.zip".
We replaced ".mp3" with "_mp3" in the suffixes of the example ids, to make
them compatible with the user taxonomies.
Also 30 examples were removed to provide consistency with another exampleset containing
more features which will be available soon.

References
----------

[Homburg/etal/2005a]   Homburg, Helge and Mierswa,Ingo and Möller, Bülent and Morik, Katharina
A Benchmark Dataset for Audio Classification and Clustering. In Proc. of the International
Symposium on Music Information Retrieval 2005, 2005.
[Mierswa/Morik/2005a]  Mierswa, Ingo and Morik, Katharina. Automatic Feature Extraction for C
```