Notes

- No class on Thursday, Mar 18
- Takehome MT due Mar 25
- Paper "Kernels and Ensembles" by M. Zhu, is posted under March 9
- "Quick R":
  http://www.statmethods.net/index.html
- CRAN Task Views: http://cran.r-project.org/web/views/MachineLearning.html
Classification and Regression Trees

South African heart data

```plaintext
<table>
<thead>
<tr>
<th>age&lt; 50.5</th>
<th>age&lt; 30.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>typea&lt; 68.5</td>
<td>famhist=a</td>
</tr>
<tr>
<td>tobacco&lt; 7.605</td>
<td>ldl&lt; 4.99</td>
</tr>
<tr>
<td>adiposity&gt;=27.98</td>
<td></td>
</tr>
<tr>
<td>tobacco&lt; 4.15</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
```

... heart data

```r
> data(SAheart)
> names(SAheart)

[1] "sbp"   "tobacco" "ldl"     "adiposity" "famhist"
[6] "typea" "obesity" "alcohol" "age"     "chd"

> (heartree = rpart(chd ~ ., data = SAheart, method="class"))

## output follows
##
n= 462

node), split, n, loss, yval, (yprob)
 * denotes terminal node

1) root 462 160 0 (0.653680 0.346320)
   2) age< 50.5 290 64 0 (0.779310 0.220690)
      4) age< 30.5 108 8 0 (0.925926 0.074074) *
      5) age>=30.5 182 56 0 (0.692308 0.307692)
         10) typea< 68.5 170 46 0 (0.729412 0.270588) *
         11) typea>=68.5 12 2 1 (0.166667 0.833333) *
   3) age>=50.5 172 76 1 (0.441860 0.558140)
      6) famhist=Absent 82 33 0 (0.597561 0.402439)
         12) tobacco< 7.605 58 16 0 (0.724138 0.275862) *
         13) tobacco>=7.605 24 7 1 (0.291667 0.708333) *
      7) famhist=Present 90 27 1 (0.300000 0.700000)
         14) ldl< 4.99 39 18 1 (0.461538 0.538462)
            28) adiposity>=27.985 20 7 0 (0.650000 0.350000)
               56) tobacco< 4.15 10 1 0 (0.900000 0.100000) *
               57) tobacco>=4.15 10 4 1 (0.400000 0.600000) *
            29) adiposity< 27.985 19 5 1 (0.263158 0.736842) *
         15) ldl>=4.99 51 9 1 (0.176471 0.823529) *
```
> plot(heartree, margin = .10)
> text(heartree) # depth of branches proportional to reduction in error
> plot(heartree, margin = .10, compress = T, uniform = T, branch = 0.4)
> text(heartree, use.n = T) # depth of branches is uniform
> post(heartree) # makes a file called heartree.ps in the local directory

> printcp(heartree)

Classification tree:
rpart(formula = chd ~ ., data = SAheart, method = "class")

Variables actually used in tree construction:
[1] adiposity age famhist ldl tobacco typea

Root node error: 160/462 = 0.346

n= 462

<table>
<thead>
<tr>
<th>CP</th>
<th>nsplit</th>
<th>rel error</th>
<th>xerror</th>
<th>xstd</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.1250</td>
<td>0</td>
<td>1.000</td>
<td>1.000</td>
</tr>
<tr>
<td>2</td>
<td>0.1000</td>
<td>1</td>
<td>0.875</td>
<td>1.056</td>
</tr>
<tr>
<td>3</td>
<td>0.0625</td>
<td>2</td>
<td>0.775</td>
<td>1.000</td>
</tr>
<tr>
<td>4</td>
<td>0.0250</td>
<td>3</td>
<td>0.713</td>
<td>0.863</td>
</tr>
<tr>
<td>5</td>
<td>0.0188</td>
<td>5</td>
<td>0.663</td>
<td>0.831</td>
</tr>
<tr>
<td>6</td>
<td>0.0125</td>
<td>7</td>
<td>0.625</td>
<td>0.875</td>
</tr>
<tr>
<td>7</td>
<td>0.0100</td>
<td>8</td>
<td>0.613</td>
<td>0.931</td>
</tr>
</tbody>
</table>

> table(actual=SAheart$chd, predicted=predict(heartree, type="class"))

<table>
<thead>
<tr>
<th>predicted</th>
<th>actual</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>275</td>
</tr>
<tr>
<td>1</td>
<td>71</td>
</tr>
</tbody>
</table>

1 - sum(diag(.Last.value))/sum(.Last.value)

[1] 0.21212

## this is on the training data, not new test data, so is overly optimistic

> heartlogreg = glm(chd ~ sbp+tobacco+ldl+famhist+obesity+alcohol+age, data=SAheart, family=binomial)

> table(SAheart$chd, predict(heartlogreg, type="response">0.5)

<table>
<thead>
<tr>
<th></th>
<th>FALSE</th>
<th>TRUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>actual</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>255</td>
<td>47</td>
</tr>
<tr>
<td>1</td>
<td>78</td>
<td>82</td>
</tr>
</tbody>
</table>

1 - sum(diag(.Last.value))/sum(.Last.value)

[1] 0.27056

## so we've done a bit better; but true test is on test data
Forensic glass

data(fgl)
dim(fgl)
# [1] 214 10

fgl[1:4,

# **********************************************************
# RI  Na  Mg  Al  Si  K  Ca  Ba  Fe  type
# 1   3.01 13.64 4.49 1.10 71.78 0.06 8.75 0  0  WinF
# 2  -0.39 13.89 3.60 1.36 72.73 0.48 7.83 0  0  WinF
# 3  -1.82 13.53 3.55 1.54 72.99 0.39 7.78 0  0  WinF
# 4  -0.34 13.21 3.69 1.29 72.61 0.57 8.22 0  0  WinF
# **********************************************************

levels(fgl$type)
# **********************************************************
# [1] "WinF" "WinNF" "Veh" "Con" "Tabl" "Head"
# **********************************************************
#
>
> fgltree = rpart(type ~ ., data = fgl, cp=0.001)
> plot(fgltree, unif = T); text(fgltree, use.n=T, cex=0.8)
> fgltree2 = prune(fgltree, cp=0.02)
> plot(fgltree, unif = T); text(fgltree, use.n=T, cex=0.8)
> plot(fgltree2, unif = T); text(fgltree2, use.n=T, cex=0.8)
> table(fgl$type,predict(fgltree2,type="class"))

<table>
<thead>
<tr>
<th></th>
<th>WinF</th>
<th>WinNF</th>
<th>Veh</th>
<th>Con</th>
<th>Tabl</th>
<th>Head</th>
</tr>
</thead>
<tbody>
<tr>
<td>WinF</td>
<td>59</td>
<td>11</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>WinNF</td>
<td>7</td>
<td>57</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Veh</td>
<td>3</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Con</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Tabl</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Head</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>26</td>
</tr>
</tbody>
</table>

> 1-sum(diag(.Last.value))/sum(.Last.value)
[1] 0.22897
Tree-based regression

- output $y$ is continuous
- model: $f(x) = \sum_{m=1}^{M} c_m 1\{x \in \mathcal{R}_m\}$
- response is constant in each region
- $\mathcal{R}_m$ is a subspace of $\mathbb{R}^p$ obtained by partitioning the feature space using binary splits
- for fixed set of regions, $\{\mathcal{R}_m\}$, $\hat{c}_m = \text{ave}(y_i \mid x_i \in \mathcal{R}_m)$
- fitting: find the ‘best’ split to minimize residual sum of squares

\[
\min_{j,s} \left\{ \min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2 \right\}
\]

- $R_1(j, s) = \{X \mid X_j \leq s\}$
- $R_2(j, s) = \{X \mid X_j > s\}$
...tree-based regression

- construct large tree
- ‘prune’ the tree using

\[ C_\alpha(T) = \sum_{m=1}^{\left| T \right|} \sum_{x_i \in R_m} (y_i - \hat{c}_m)^2 + \alpha \left| T \right| \]

- estimate \( \alpha \) using 5- or 10-fold cross-validation
- cost-complexity criterion

\[ C_\alpha(T) = \sum_{m=1}^{\left| T \right|} N_m Q_m(T) + \alpha \left| T \right| \]

\[ Q_m(T) = \frac{1}{N_m} \sum_{x_i \in R_m} (y_i - \hat{c}_m)^2 \]
... tree based regression: example

library(MASS)

# Implementation in RPART (dataset: cpus)

library(rpart)

data(cpus)

dim(cpus)
# [1] 209 9

# RPART differs from TREE function mainly in its handling of surrogate variables
# In most details it follows Breiman’s et al quite closely.

cpus.rp <- rpart(log10(perf) ~ ., cpus[,2:8], cp=1e-3)
post(cpus.rp, title="Plot of rpart object cpus.rp", filename="Cpus.tree.ps", horizontal=F, pointsize=8)
Classification trees

- $Y = k$, $k = 1, \ldots, K$
- $\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} 1\{y_i = k\}$, $N_m = \#\{x_i \in R_m\}$
- proportion of observations in node $m$ that fall in class $k$
- assign class by maximum probability: $k(m) = \arg\max_k \hat{p}_{mk}$
- replace $Q_m$ by a measure of ‘node impurity’
  1. $\frac{1}{N_m} \sum_{x_i \in R_m} 1\{y_k \neq k(m)\} = 1 - \hat{p}_{mk(m)}$
     misclassification error
  2. $\sum_{k \neq k'} \hat{p}_{mk} \hat{p}_{mk'} = \sum_{k=1}^{K} \hat{p}_{mk} (1 - \hat{p}_{mk})$
     Gini index
  3. $- \sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$
     “cross-entropy”, deviance, negative log-likelihood

Figure 9.3
... classification trees

- usually use 1. to prune the tree and 2. or 3. to grow the tree

- 2. is the default in `rpart`: to get 3. need `method = "class", parms = list(split = "information")`

- `method = "class"` is default if response is a factor variable
Other issues §9.2.4

- Categorical predictors: $2^{q-1} - 1$ possible partitions, can be reduced to $q - 1$ by a trick, when $y = 0/1$
- Loss matrix $L_{kk'}$; loss for classifying a class $k$ observation as class $k'$
- Missing features: new category; construction of surrogate
- C5.0, C4.5 (Quinlan)
- Instability and lack of smoothness
- Figure 9.5
Bagging = bootstrap aggregation

- data $z = (z_1, \ldots, z_N) = ((x_1, y_1), \ldots, (x_N, y_N))$
- fit $\hat{f}(x)$ based on $z$
- bootstrap: $z^*_b$: resample $z$ with replacement
- fit $\hat{f}^*_b(x)$
- repeat $B$ times

$$\hat{f}_{bag}(x) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}^*_b(x)$$

Figure 8.2

- Example: classification tree (§8.7.1)
- 5 correlated inputs: response depends only on 1st input: $P(Y = 1 \mid x_1 \leq 0.5) = 0.2; P(Y = 1 \mid x_1 > 0.5) = 0.8$
- Figure 8.9, 8.10
... bagging

- aggregation reduces mean squared error for regression trees, see (8.52)
- for classification trees, aggregation simulates “wisdom of crowds”
  - \( G(x) = 1 \) true value
  - \( G^*_b(x) \) classification rule with error rate \( e < 0.5 \) say
  - consensus vote \( S_1(x) = \sum_{b=1}^{B} G^*_b(x) \sim Bin(B, 1 - e) \) if \( G^*_b \) independent
  - \( \Pr(S_1(x) > B/2) \to 1, \quad B \to \infty \) Figure 8.11
- BUT, \( \hat{f}_b^* \) for trees are in fact highly correlated Figure 8.12
Random Forests Ch. 15

- trees are highly interpretable, but also quite variable
- bagging (bootstrap aggregation) resamples from the data to build $B$ trees, then averages
- if $X_1, \ldots, X_N$ independent $(\mu, \sigma^2)$, then $\text{var}(\bar{X}) = \sigma^2 / B$
- if corr($X_i, X_j$) = $\rho > 0$, then
  
  $$\text{var}(\bar{X}) = \rho \sigma^2 + \frac{1 - \rho}{B} \sigma^2$$

  $\rightarrow \rho \sigma^2$ as $B \rightarrow \infty$; no benefit from aggregation

- average many trees as in bagging, but reduce correlation using a trick: use only a random sample of $m$ of the $p$ input variables each time a node is split
- $m = O(\sqrt{p})$, for example, or even smaller
... random forests

▶ See Algorithm 15.1
▶ email spam example in R
▶ Figures 15.1, 2, 4, 5

```r
> spam2 = spam
deleting columns.
> names(spam2)=c(spam.names,"spam")
> spam.rf = randomForest(x=as.matrix(spam2[spamtest==0,1:57]),
                        y=spam2[spamtest==0,58], importance=T)
> varImpPlot(spam.rf)
> table(predict(spam.rf, newdata = as.matrix(spam2[spamtest==1,])),spam2[spamtest==1,58])

                     email spam
email         908   38
spam           33  557

> .Last.value/sum(spamtest)

                     email spam
email  0.591146 0.024740
spam  0.021484 0.362630
> .0247+.02148
[1] 0.04618
```
... random forests

spam.rf

| ! | $ | remove | average | hp | longest | free | your | edu | george | total | 1999 | our | re | you | ( | hpl | business | 000 | meeting | money | will | internet | pm | 650 | email | receive | font | mail | ; | 0.35 | 0.40 | 0.45 | 0.50 | 0.55 | 0.60 | MeanDecreaseAccuracy |
|---|---|-------|--------|----|---------|------|------|-----|--------|-------|------|-----|----|-----|---|-----|----------|-----|---------|------|------|--------|---|---|------|---------|-------|------|MeanDecreaseGini|