Reference: Leo Breiman,
http://www.stat.berkeley.edu/~breiman/RandomForests

1. Decision trees

Example (Guerts, Fillet, et al., Bioinformatics 2005):

Patients to be classified: normal vs. diseased
Decision trees
Classification of biomarker data: large number of values (e.g., microarray or mass spectrometry analysis of biological sample)
Decision trees
Mass spectrometry (m/z) parameters or gene expression parameters (around 15k values)

<table>
<thead>
<tr>
<th>Patients (10-500)</th>
<th>m/z values or gene expression (±15000)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A1</td>
</tr>
<tr>
<td></td>
<td>0.3</td>
</tr>
<tr>
<td></td>
<td>-123</td>
</tr>
<tr>
<td></td>
<td>56</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>89</td>
</tr>
</tbody>
</table>
Decision trees
Given new patient with biomarker data, is s/he normal or ill?

Needed: selection of relevant variables from many variables

Number $n$ of known examples in $D = \{ (x_i, y_i) \}_{i=1}^{n}$ is small (characteristic of machine learning/data mining problems)

Assume we have for each biological sample a feature vector $x$, and will classify it:
Decision trees

diseased: \( y = 1; \)  normal: \( y = -1. \)

Goal: find function \( f(x) \approx y \) which predicts \( y \) from \( x \).

How to estimate error of \( f(x) \) and avoid over-fitting the small dataset \( D \)?

Use cross-validation, i.e., test predictor \( f(x) \) in an unexamined part of sample set \( D \).
Decision trees

For biological sample, feature vector $\mathbf{x} = (x_1, \ldots, x_d)$ consists of features (or biomarkers or attributes) $x_i = A_i$ describing the biological sample from which $\mathbf{x}$ is obtained.
The decision tree approach
Decision tree approach to finding predictor $f(x) = y$ from data set $D$:

⊕ form a tree whose nodes are features (attributes)

$x_i = A_i$ in $x$

⊕ decide which features $A_i$ to consider first in predicting $y$ from $x$

i.e., find features $A_i$ with highest information gain - place these at top of tree
The decision tree approach then use recursion - form sub-trees based on attributes not used in the higher nodes:

Advantages: interpretable, easy to use, scalable, robust
Decision tree example

Example 1 (Moore): UCI data repository
(http://www.ics.uci.edu/~mlearn/MLRepository.html)

MPG (miles per gallon) ratings of cars:
Goal: predict MPG rating of a car from a set of features/attributes $A_i$
Decision tree example
Examples (each row is feature set for a sample car):

<table>
<thead>
<tr>
<th>mpg</th>
<th>cylinders</th>
<th>displacement</th>
<th>horsepower</th>
<th>weight</th>
<th>acceleration</th>
<th>model year</th>
<th>maker</th>
</tr>
</thead>
<tbody>
<tr>
<td>good</td>
<td>4</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>high</td>
<td>00to03</td>
<td>asia</td>
</tr>
<tr>
<td>bad</td>
<td>6</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>95to99</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>4</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>low</td>
<td>00to03</td>
<td>europe</td>
</tr>
<tr>
<td>bad</td>
<td>8</td>
<td>high</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>95to99</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>6</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>95to99</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>4</td>
<td>low</td>
<td>medium</td>
<td>low</td>
<td>medium</td>
<td>95to99</td>
<td>asia</td>
</tr>
<tr>
<td>bad</td>
<td>4</td>
<td>low</td>
<td>medium</td>
<td>low</td>
<td>medium</td>
<td>95to99</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>6</td>
<td>high</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>00to03</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>6</td>
<td>high</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>00to03</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>8</td>
<td>high</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>95to99</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>4</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>medium</td>
<td>00to03</td>
<td>europe</td>
</tr>
<tr>
<td>bad</td>
<td>5</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>00to03</td>
<td>europe</td>
</tr>
</tbody>
</table>

R. Quinlan
Decision tree example

Simple assessment of information gain: how much does a particular feature $A_i$ help to classify a car with respect to MPG?
### Decision tree example

<table>
<thead>
<tr>
<th>Input</th>
<th>Value</th>
<th>Distribution</th>
<th>Info Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>cylinders</td>
<td>3</td>
<td>0.505731</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>displacement</td>
<td>low</td>
<td>0.222114</td>
<td></td>
</tr>
<tr>
<td></td>
<td>medium</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>high</td>
<td></td>
<td></td>
</tr>
<tr>
<td>horsepower</td>
<td>low</td>
<td>0.367605</td>
<td></td>
</tr>
<tr>
<td></td>
<td>medium</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>high</td>
<td></td>
<td></td>
</tr>
<tr>
<td>weight</td>
<td>low</td>
<td>0.304018</td>
<td></td>
</tr>
<tr>
<td></td>
<td>medium</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>high</td>
<td></td>
<td></td>
</tr>
<tr>
<td>acceleration</td>
<td>low</td>
<td>0.0842988</td>
<td></td>
</tr>
<tr>
<td></td>
<td>medium</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>high</td>
<td></td>
<td></td>
</tr>
<tr>
<td>modelyear</td>
<td>70s74</td>
<td>0.267004</td>
<td></td>
</tr>
<tr>
<td></td>
<td>78s78</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>79s83</td>
<td></td>
<td></td>
</tr>
<tr>
<td>maker</td>
<td>america</td>
<td>0.0437266</td>
<td></td>
</tr>
<tr>
<td></td>
<td>asia</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Decision tree example

Begin the decision tree: start with most informative criterion, cylinders:

```
mpg values: bad good

root
22 18
pchange = 0.001

<table>
<thead>
<tr>
<th>cylinders = 3</th>
<th>cylinders = 4</th>
<th>cylinders = 5</th>
<th>cylinders = 6</th>
<th>cylinders = 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0</td>
<td>4 17</td>
<td>1 0</td>
<td>8 0</td>
<td>9 1</td>
</tr>
</tbody>
</table>

Predict bad    Predict good    Predict bad    Predict bad    Predict bad
```
Decision tree example

Recursion: build next level of tree. Initially have:
Decision tree example

Now build sub-trees: split each set of cylinder numbers into further groups-

Resulting next level:
Recursively build a tree from the seven records in which there are four cylinders and the maker was based in Asia.
Decision tree example

Final tree:
Decision tree example

Points:

- Don't split node if all records have same value (e.g. cylinders = 6)

- Don't split node if can't have more than 1 child (e.g. acceleration = medium)
Pseudocode:

Program Tree(Input, Output)

If all output values are the same, then return leaf (terminal) node which predicts the unique output
If input values are balanced in a leaf node (e.g. 1 good, 1 bad in acceleration) then return leaf predicting majority of outputs on same level (e.g. bad in this case)
Else find attribute $A_i$ with highest information gain

If attribute $A_i$ at current node has $m$ values
then Return internal (non-leaf) node with $m$ children
Build child $i$ by calling Tree(NewIn, NewOut), where
NewIn = values in
dataset consistent with value $A_i$ and all
values above this node
Another decision tree: prediction of wealth from census data (Moore):
Prediction of age from census:
Prediction of gender from census:

A. Moore
2. Important point: always cross-validate

It is important to test your model on new data (test data) which are different from the data used to train the model (training data).

This is cross-validation.

Cross-validation error – 2% is good; 40% is poor.
3. Background: mass spectroscopy

What does a mass spectrometer do?

1. It measures masses of molecules better than any other technique.

2. It can give information about chemical (in particular protein) compositions of tissue samples.
Mass spectroscopy

How does it work?

1. Takes unknown molecule $M$, adds $i$ protons to it giving it charge $+i$ (forming $MH_i^+$)

2. Accelerates ion $MH_i^+$ in *known* electric field $E$.

3. Measures time of flight along a *known* distance $D$.

4. Time $T$ of flight is inversely proportional to electric charge $i$ and proportional to mass $m$ of ion.
Mass spectrometry

Thus

\[ T \propto m/i \]

So mass spectrometer measures ratio of mass \( m \) and charge \( i \) (also known as \( z \)), i.e., \( m/i = m/z \).

With a large number of molecules in a biosample, this gives a spectrum of \( z/m \) values, which allows identification of molecules in sample (below \( \text{IgG} = \text{immunoglobulin G} \) )
Mass spectroscopy

MALDI TOF spectrum of IgG

Relative Abundance

(M+2H)^2+

(M+3H)^3+

MH^+

Mass (m/z)

500000

1000000

1500000

2000000
Mass spectroscopy
Mass spectrum shows the results:
Mass spectrometry
ESI Spectrum of Trypsinogen (MW 23983)
Mass spectroscopy

4. Dimensional reduction (G. Izmirlian):

Sometimes we perform a *dimension reduction* by reducing mass spectrum information of (human) subject $i$ to store only peaks:
Mass spectrometry
Mass spectroscopy

Then have (compressed) peak information in feature vector

\[ \mathbf{x} = (x_1, \ldots, x_d), \]

with \( x_k = \) location of \( k^{th} \) mass spectrum peak (above a fixed threshold).

Compressed or not, outcome value to feature vector \( \mathbf{x}_i \) for subject \( i \) is \( y_i = \pm 1 \).
5. Random forest example

Example (Guerts, et al.):

Normal/sick dichotomy for RA and for IBD based on blood sample protein markers (above - Geurts, et al.):

We now build a forest of decision trees based on differing attributes in the nodes:
Random forest application

Note: different trees have access to a different random sub-collection of the feature set $\{A_i\}_{i=1}^n$, or to a different random subcollection of the data.
Random forest application

For example: Could use mass spectroscopy data as above to determine disease state

Mass Spec segregates proteins through spectrum of $m/z$ ratios (again $m = \text{mass}; \ z = \text{charge}$).
Random forest application

Geurts, et al.
Random Forests:

**Advantages:** accurate, easy to use (Breiman software), fast, robust

**Disadvantages:** difficult to interpret

**More generally:** How to combine results of different predictors (e.g. decision trees)?

Random forests are examples of *ensemble methods,* which combine predictions of weak classifiers $p_i(x)$. 
Ensemble methods: observations

1. **Boosting**: As seen earlier, take linear combination of predictions $p_i(x)$ by classifiers $i$ (assume these are decision trees)

$$f(x) = \sum_{i} a_i p_i(x),$$

(1)

where $p_i(x) = \begin{cases} 1 & \text{if } i^{th} \text{ tree predicts illness} \\ -1 & \text{otherwise} \end{cases}$,

and predict $y = 1$ if $f(x) \geq 0$ and $y = -1$ if $f(x) < 0$. 
Ensemble methods: observations

2. **Bagging**: Take a vote: majority rules (equivalent in this case to setting $a_i = 1$ for all $i$ in (1) above).

Example of a Bagging algorithm is *random forest*, where a forest of decision trees takes a vote.
General features of a random forest:
If original feature vector $\mathbf{x} \in \mathbb{R}^d$ has $d$ features $A_1, \ldots, A_d$,

♦ Each tree uses a random selection of $m \approx \sqrt{d}$ features $\{A_{i,j}\}_{j=1}^m$ chosen from all features $A_1, A_2, \ldots, A_d$; the associated feature space is different (but fixed) for each tree and denoted by $F_k$, $1 \leq k \leq K = \#$ trees.

(Often $K = \#$ trees is large; e.g., $K = 500$).

♦ For each split in a tree node based on a given variable choose the variable $A_i$ to be used from its information content.
Information content in a node

To compute information content of a node:
Information content in a node

Assume input set to node is \( S \): then information content of node \( N \) is

\[
I(N) = |S| H(S) - |S_L| H(S_L) - |S_R| H(S_R),
\]

where

\[
|S| = \text{input sample size}; \quad |S_{L,R}| = \text{size of left, right subclasses of } S
\]

\[
H(S) = \text{Shannon entropy of } S = - \sum_{i=\pm1} p_i \log_2 p_i
\]
Information content in a node with

\[ p_i = \hat{P}(C_i|S) = \text{proportion of class } C_i \text{ in sample } S. \]

[later we will use \textit{Gini index}, another criterion]

Thus \( H(S) \) = "variability" or "lack of full information" in the probabilities \( p_i \) forming sample \( S \) input into current node \( N \).

\[ I(N) = \text{"information from node } N\". \]
Information content in a node

For each variable $A_i$, average over all nodes $N$ in all trees involving this variable to find average information content $H_{av}(A_i)$ of $A_i$. 

![Bar chart showing information content for variables A6 to A18]
Information content in a node

(a) Rank all variables $A_i$ according to information content

(b) For each fixed $n_1 < n$ rebuild the Random Forest using only the first $n_1$ variables. Select $n_1$ which minimizes prediction error.
Information content in a node

Geurts, et al.
Random forests: application

Application to:

- early diagnosis of Rehumatoid arthritis
- rapid diagnosis of inflammatory bowel diseases (IBD)
Random forests: application

3 patient groups (University Hospital of Liege):

<table>
<thead>
<tr>
<th></th>
<th>RA</th>
<th>IBD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disease patients</td>
<td>34</td>
<td>60</td>
</tr>
<tr>
<td>Negative controls</td>
<td>29</td>
<td>30</td>
</tr>
<tr>
<td>Inflammatory controls</td>
<td>40</td>
<td>30</td>
</tr>
<tr>
<td>Total</td>
<td>103</td>
<td>120</td>
</tr>
</tbody>
</table>

Mass spectra obtained by SELDI-TOF mass spectrometry on protein chip array proteins:
Random forests: application

- Hydrophobic (H4) chips
- weak cation-exchange (CM10) chips
- strong cation-exchange (Q10) chips
Random forests: application

Feature vectors for tissue classification: $\mathbf{x} \in F$ consists of about 15,000 Mass Spectometry values in each case.
Random forests: application

Effective dimension reduction method: Discretize horizontally and vertically to go from 15,000 to 300 variables
Random forests: application
Random forests: application

Sensitivity and specificity:

Guerts, et al.
Random forests: application
Above: accuracy measures for
   DT=Decision tree;
   RF=random forest;
   $\kappa$NN = $\kappa$-nearest neighbors;
   BA = bagging (bootstrapped resampled tree ensembles);
   BO = Boosting;
   ET = Extra trees (variation on RF)
Random forests: application

Note on sensitivity and specificity: use confusion matrix

<table>
<thead>
<tr>
<th>Actual Condition</th>
<th>True</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>TP</td>
<td>FP</td>
</tr>
<tr>
<td>Negative</td>
<td>FN</td>
<td>TN</td>
</tr>
</tbody>
</table>

Test outcome

Sensitivity = \( \frac{TP}{TP + FN} = \frac{TP}{\text{Total positives}} \)
Random forests: application

Specificity = \frac{TN}{TN + FP} = \frac{TN}{\text{Total negatives}}

Positive predictive value = \frac{TP}{TP + FP} = \frac{TP}{\text{Total predicted positives}}
Random forests: application

Variable ranking on the IBD dataset:

10 most important variables in spectrum:
Random forests: application

RF-based (tree ensemble) - based variable ranking vs. variable ranking by individual variable $p$ values:
Random forests: application

RA

IBD
6. RF software:

Spider:  
http://www.kyb.tuebingen.mpg.de/bs/people/spider/whatisit.html

Leo Breiman:  
http://www.stat.berkeley.edu/~breiman/RandomForest/cc_software.htm

WEKA machine learning software  
http://www.cs.waikato.ac.nz/ml/weka/  