Decision Trees and Random Forests


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 parameters or
gene expression parameters (around 15k values)

Given new patient with biomarker data, is s/he normal or ill?

<table>
<thead>
<tr>
<th>Patients (10-500)</th>
<th>A1</th>
<th>A2</th>
<th>...</th>
<th>An</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.3</td>
<td>28.34</td>
<td>...</td>
<td></td>
<td>123</td>
<td>Normal</td>
</tr>
<tr>
<td>-123</td>
<td>0</td>
<td>...</td>
<td></td>
<td>17</td>
<td>...</td>
</tr>
<tr>
<td>56</td>
<td>-123</td>
<td>...</td>
<td></td>
<td>-23</td>
<td>Normal</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td></td>
<td>...</td>
<td>Disease</td>
</tr>
<tr>
<td>89</td>
<td>-123</td>
<td>...</td>
<td></td>
<td>12</td>
<td>Disease</td>
</tr>
</tbody>
</table>
Decision trees

Needed: selection of relevant variables from many

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

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

diseased: $y = 1$; normal: $y = -1$.

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

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

Use cross-validation to test predictor $f(x)$ in an unexamined part of the sample $D$.

For biological sample, feature vector $x = (x_1, \ldots, x_d)$ consists of features (or biomarkers or attributes) $x_i = A_i$ describing the biological sample from which $x$ is obtained.
The decision tree approach

Decision tree approach to finding predictor $f(x) = y$ based on data set $D$:

- Form a tree whose nodes are attributes $x_i = A_i$ in $x$
- Decide which attributes $A_i$ to look at first in predicting $y$ from $x$ find those with highest information gain - place these at top of tree
- Then use recursion to form sub-trees based on attributes not used in the higher nodes:
The decision tree approach

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 ratings of cars:
Goal: predict MPG rating of a car from a set of attributes $A_i$
Decision tree example

Examples (each row is attribute 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>modelyear</th>
<th>maker</th>
</tr>
</thead>
<tbody>
<tr>
<td>good</td>
<td>4</td>
<td>low</td>
<td>low</td>
<td>high</td>
<td>high</td>
<td>75to78</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>70to74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>4</td>
<td>medium</td>
<td>medium</td>
<td>low</td>
<td>low</td>
<td>75to78</td>
<td>europe</td>
</tr>
<tr>
<td>bad</td>
<td>8</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>low</td>
<td>70to74</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>70to74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>4</td>
<td>low</td>
<td>medium</td>
<td>low</td>
<td>low</td>
<td>70to74</td>
<td>asia</td>
</tr>
<tr>
<td>bad</td>
<td>8</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>low</td>
<td>75to78</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>8</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>low</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>8</td>
<td>high</td>
<td>medium</td>
<td>high</td>
<td>high</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>8</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>low</td>
<td>75to78</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>4</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>6</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>high</td>
<td>75to78</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>4</td>
<td>medium</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>4</td>
<td>low</td>
<td>medium</td>
<td>high</td>
<td>high</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>8</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>low</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>4</td>
<td>low</td>
<td>medium</td>
<td>low</td>
<td>medium</td>
<td>75to78</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>75to78</td>
<td>europe</td>
</tr>
</tbody>
</table>

R. Quinlan
Decision tree example

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

Begin the decision tree: start with most informative

<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></td>
<td>0.505731</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></td>
<td>0.223114</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></td>
<td>0.367695</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></td>
<td>0.304018</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></td>
<td>0.0642089</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>model_year</td>
<td>70-74</td>
<td></td>
<td>0.267004</td>
</tr>
<tr>
<td></td>
<td>75-79</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>79-83</td>
<td></td>
<td></td>
</tr>
<tr>
<td>maker</td>
<td>america</td>
<td></td>
<td>0.0437265</td>
</tr>
<tr>
<td></td>
<td>asia</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Decision tree example

criterion, cylinders:

mpg values: bad good

root
22 18
pchance = 0.001

cylinders = 3
0 0
Predict bad

cylinders = 4
4 17
Predict good

cylinders = 5
1 0
Predict bad

cylinders = 6
8 0
Predict bad

cylinders = 8
9 1
Predict bad
Decision tree example

Recursion: build next level of tree. Initially have:

Now build sub-trees: split each set of cylinder numbers into
Decision tree example

Further groups:

Resulting next level:
Decision tree example

mpg values: bad good

Final tree:

Recursively build a tree from the seven records in which there are four cylinders and the maker was based in Asia

(Similar recursion in the other cases)
Decision tree example

mpg values: bad, good

root
22 10
pchance = 0.001

cylinders = 3
cylinders = 4
cylinders = 5
cylinders = 6
cylinders = 8

Predict bad
pchance = 0.135
Predict bad
Predict bad
pchance = 0.085

maker = america
maker = asia
maker = europe
horsepower = low
horsepower = medium
horsepower = high

Predict good
pchance = 0.317
Predict bad
pchance = 0.717
Predict good
Predict bad
pchance = 0.717

horsepower = low
horsepower = medium
horsepower = high
acceleration = low
acceleration = medium
acceleration = high

Predict good
pchance = 0.854
Predict bad
Predict bad
Predict good
pchance = 0.717

acceleration = low
acceleration = medium
acceleration = high
modelyear = 70to74
modelyear = 75to78
modelyear = 79to83

Predict bad
Predict bad
Predict good
Predict bad
Predict bad
Predict bad

Predict bad
(unexpandable)
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 structures of molecules.
Mass spectroscopy

How does it work?

1. Takes unknown molecule M, adds $i$ protons to it giving it charge $+i$ (forming $\text{MH}_i^+$)

2. Accelerates ion $\text{MH}_i^+$ 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 spectroscopy

Thus

\[ T \propto \frac{i}{m} \]

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

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

MALDI TOF spectrum of IgG

Relative Abundance

Mass (m/z)
Mass spectroscopy

What are the measurements good for?

To identify, verify, and quantify: metabolites, proteins, oligonucleotides, drug candidates, peptides, synthetic organic chemicals, polymers

Applications of Mass Spectrometry

Biomolecule characterization
Pharmaceutical analysis
Proteins and peptides
Oligonucleotides
Mass spectroscopy

How does a mass spectrometer work?

- **Sample**
- **Ion source:** makes ions
- **Mass analyzer:** separates ions
- **Mass spectrum:** presents information
Mass spectroscopy

Mass Spectrometer Block Diagram

[Source: Sandler Mass Spectroscopy]
Mass spectroscopy

Two types of ionization:

1. Electrospray ionization (ESI):
**Mass spectroscopy**

*Ion Sources make ions from sample molecules* (Ions are easier to detect than neutral molecules.)

**Electrospray ionization:**

- Pressure = 1 atm
- Inner tube diam. = 100 um
- Sample Inlet Nozzle (Lower Voltage)
- Partial vacuum

- Sample in solution
- $\text{N}_2$ gas
- High voltage applied to metal sheath (~4 kV)

- Charged droplets
  - $\text{MH}^+$
  - $\text{MH}_2^+$
  - $\text{MH}_3^+$
Mass spectroscopy

[above $MH_i^+$ denotes molecule with $i$ protons ($H^+$) attached]
2. MALDI:

MALDI: Matrix Assisted Laser Desorption Ionization

1. Sample is mixed with matrix (X) and dried on plate.
2. Laser flash ionizes matrix molecules.
3. Sample molecules (M) are ionized by proton transfer: 
   \[ XH^+ + M \rightarrow MH^+ + X. \]
Mass spectroscopy

Mass analyzers separate ions based on their mass-to-charge ratio (m/z)

- Operate under high vacuum
- Measure mass-to-charge ratio of ions (m/z)
Mass spectroscopy

**Components:**

1. **Quadrupole Mass Analyzer (filter)**

Uses a combination of RF and DC voltages to operate as a mass filter before masses are accelerated.

- Has four parallel metal rods.
- Lets one mass pass through at a time.
- Can scan through all masses or only allow one fixed mass.
Mass spectroscopy
Mass spectroscopy

2. Time-of-flight (TOF) Mass Analyzer

Accelerates ions with electric field, detects them, analyzes flight time.
Mass spectroscopy

Source

Drift region (flight tube)

V

detector
Mass spectroscopy

- Ions are formed in pulses.
- The drift region is field free.
- Measures the time for ions to reach the detector.
- Small ions reach the detector before large ones.

Ion trap mass analyzer:
Mass spectroscopy

Top View

Cut away side view
Mass spectroscopy

3. Detector: Ions are detected with a microchannel plate:
Mass spectroscopy
Mass spectrum shows the results:

ESI Spectrum of Trypsinogen (MW 23983)
Mass spectroscopy
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 spectroscopy
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 (above - Geurts, et al.): we now build a forest of decision trees based on differing attributes in the nodes:
Random forest application

For example: Could use mass spectroscopy data to determine disease state
Random forest application

Mass Spec segregates protein and other molecules through spectrum of $m/z$ ratios ($m = \text{mass}; \ z = \text{charge}$).
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)

\[
f(x) = \sum_i a_i p_i(x),
\]

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 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 $A_i$ from information content.
Information content in a node

To compute information content of a node:

Assume input set to node is $S$: then information content of node $N$ is
Information content in a node

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

where

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

\[ H(S) = \text{Shannon entropy of } S = - \sum_{i=\pm1} p_i \log_2 p_i \]

with

\[ p_i = \hat{P}(C_i|S) = \text{proportion of class } C_i \text{ in sample } S. \]
Information content in a node
[later we will use *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) = \) "information from node \( N \)".

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

(b) For each fixed $n_1 < n$ use 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 Rheumatoid arthritis
- rapid diagnosis of inflammatory bowel diseases (IBD)

3 patient groups (University Hospital of Liege):
Random forests: application

<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 chip arrays:
Random forests: application

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

Feature vectors: $\mathbf{x} \in F$ consists of about 15,000 values in each case.

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:
Random forests: application

Accuracy measures: DT=Decision tree; RF=random forest; \( k \)-NN = \( k \)-nearest neighbors;

**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

<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>
Random forests: application

Sensitivity = \( \frac{TP}{TP + FN} \) = \( \frac{TP}{\text{Total positives}} \)

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/RandomForests/cc_software.htm

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