DM825

Introduction to Machine Learning

Lecture 14

Tree-based Methods
Principal Components Analysis

Marco Chiarandini

Department of Mathematics & Computer Science
University of Southern Denmark

Outline

Outline

Learning Decision Trees

A decision tree of a pair $(\mathbf{x}, y)$ represents a function that takes the input attribute $\mathbf{x}$ (Boolean, discrete, continuous) and outputs a simple Boolean y.

E.g., situations where I will/won't wait for a table. Training set:

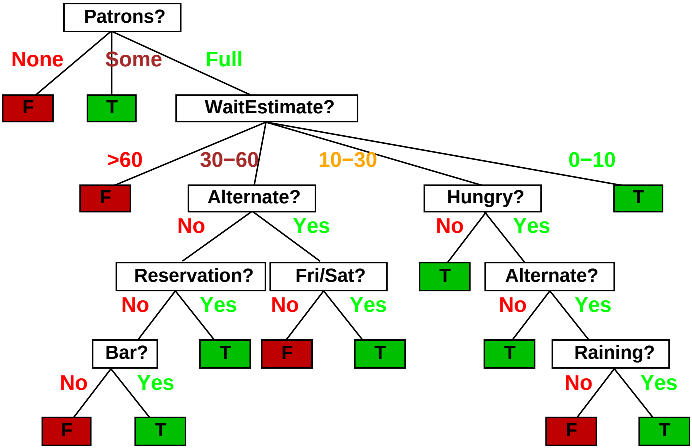
Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	$>$60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	$>$60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

Classification of examples positive (T) or negative (F)

Key property: readily interpretable by humans

Decision trees

One possible representation for hypotheses
E.g., here is the "true" tree for deciding whether to wait:



Example

NO.	RISK	CREDIT HISTORY	DEBT	COLLATERAL	INCOME
1.	high	bad	high	none	\$0 to \$15k
2.	high	unknown	high	none	\$15 to \$35k
3.	moderate	unknown	low	none	\$15 to \$35k
4.	high	unknown	low	none	\$0 to \$15k
5.	low	unknown	low	none	over \$35k
6.	low	unknown	low	adequate	over \$35k
7.	high	bad	low	none	\$0 to \$15k
8.	moderate	bad	low	adequate	over \$35k
9.	low	good	low	none	over \$35k
10.	low	good	high	adequate	over \$35k
11.	high	good	high	none	\$0 to \$15k
12.	moderate	good	high	none	\$15 to \$35k
13.	low	good	high	none	over \$35k
14.	high	bad	high	none	\$15 to \$35k

Table 10.1 Data from credit history of loan applications

Example

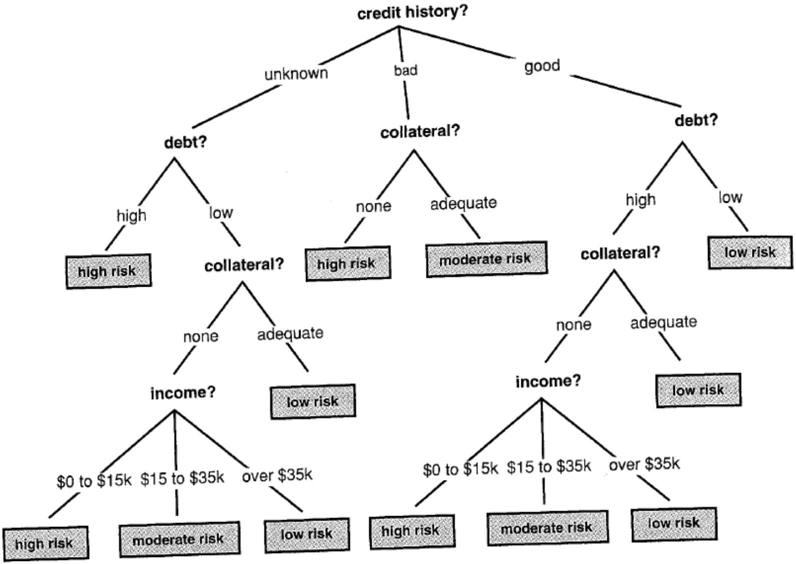
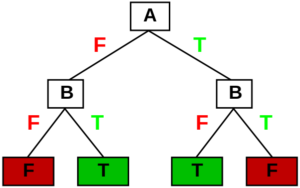


Figure 10.13 A decision tree for credit risk assessment.

Expressiveness

Decision trees can express any function of the input attributes.
E.g., for Boolean functions, truth table row $\rightarrow$ path to leaf:

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F



Trivially, there is a consistent decision tree for any training set
w/ one path to leaf for each example (unless f nondeterministic in x)
but it probably won't generalize to new examples
Prefer to find more **compact** decision trees

Hypothesis spaces

How many distinct decision trees with n Boolean attributes??
= number of Boolean functions
= number of distinct truth tables with 2^n rows = 2^{2^n} functions
E.g., with 6 Boolean attributes, there are 18,446,744,073,709,551,616 trees

More expressive hypothesis space
– increases chance that target function can be expressed
– increases number of hypotheses consistent w/ training set
$\Longrightarrow$ may get worse predictions

There is no way to search the smallest consistent tree among 2^{2^n}.

Heuristic approach

Greedy divide-and-conquer:

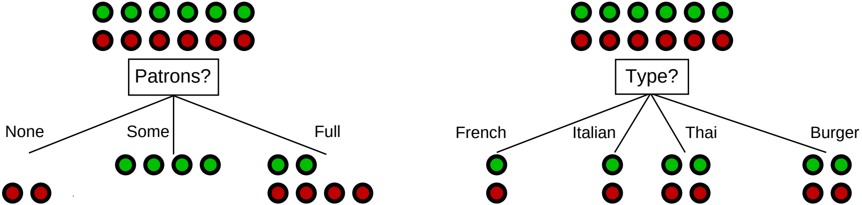
- test the most important attribute first
- divide the problem up into smaller subproblems that can be solved recursively

```
function DTL(examples, attributes, default) returns a decision tree

  if examples is empty then return default
  else if all examples have the same classification then return the classification
  else if attributes is empty then return Plurality_Value(examples)
  else
      best ← Choose-Attribute(attributes, examples)
      tree ← a new decision tree with root test best
      for each value v_i of best do
          examples_i ← {elements of examples with best = v_i}
          subtree ← DTL(examples_i, attributes − best, Mode(examples))
          add a branch to tree with label v_i and subtree subtree
      return tree
```

Choosing an attribute

Idea: a good attribute splits the examples into subsets that are (ideally) "all positive" or "all negative"



Patrons? is a better choice—gives **information** about the classification

Information

The more clueless I am about the answer initially, the more information is contained in the answer

0 bits to answer a query on a coin with only head
1 bit to answer query to a Boolean question with prior $\langle 0.5, 0.5 \rangle$
2 bits to answer a query on a fair die with 4 faces a query on a coin with 99% probability of returing head brings less information than the query on a fair coin.

Shannon formalized this concept with the concept of entropy.
For a random variable X with values x_k and probability $\Pr(x_k)$ has entropy:

$$H(X) = -\sum_k \Pr(x_k) \log_2 \Pr(x_k)$$

- Suppose we have p positive and n negative examples is a training set, then the entropy is $H(\langle p/(p+n), n/(p+n)\rangle)$
  E.g., for 12 restaurant examples, $p=n=6$ so we need 1 bit to classify a new example information of the table
- An attribute A splits the training set E into subsets $E_1, \ldots, E_d$, each of which (we hope) needs less information to complete the classification
- Let E_i have p_i positive and n_i negative examples
  $\leadsto$ $H(\langle p_i/(p_i+n_i), n_i/(p_i+n_i)\rangle)$ bits needed to classify a new example on that branch
  $\leadsto$ **expected** entropy after branching is

$$Remainder(A) = \sum_i \frac{p_i+n_i}{p+n} H(\langle p_i/(p_i+n_i), n_i/(p_i+n_i)\rangle)$$

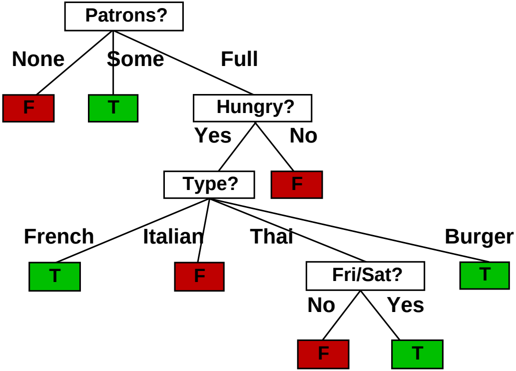
- The **information gain** from attribute A is

$$Gain(A) = H(\langle p/(p+n), n/(p+n)\rangle) - Remainder(A)$$

$\Longrightarrow$ choose the attribute that maximizes the gain

Example contd.

Decision tree learned from the 12 examples:



Substantially simpler than "true" tree—a more complex hypothesis isn't justified by small amount of data

Overfitting and Pruning

Pruning by statistical testing
under the null hyothesis expected numbers, $\hat{p}_k$ and $\hat{n}_k$:

$$\hat{p}_k = p \cdot \frac{p_k + n_k}{p + n} \quad \hat{n}_k = n \cdot \frac{p_k + n_k}{p + n}$$

$$\Delta = \sum_{k=1}^{d} \frac{(p_k - \hat{p}_k)2}{\hat{p}_k} + \frac{(n_k - \hat{n}_k)2}{\hat{n}_k}$$

χ^2 distribution with $p + n - 1$ degrees of freedom
Early stopping misses combinations of attributes that are informative.

Further Issues

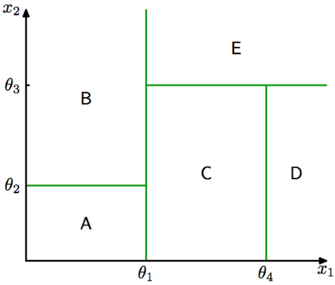
- Missing data
- Multivalued attributes
- Continuous input attributes
- Continuous-valued output attributes

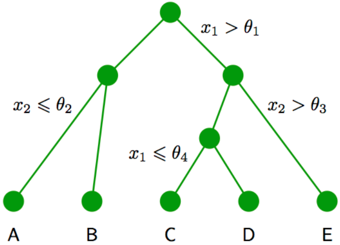
Decision Tree Types

- Classification tree analysis is when the predicted outcome is the class to which the data belongs. Iterative Dichotomiser 3 (ID3), C4.5, (Quinlan, 1986)
- Regression tree analysis is when the predicted outcome can be considered a real number (e.g. the price of a house, or a patient's length of stay in a hospital).
- Classification And Regression Tree (CART) analysis is used to refer to both of the above procedures, first introduced by (Breiman et al., 1984)
- CHi-squared Automatic Interaction Detector (CHAID). Performs multi-level splits when computing classification trees. (Kass, G. V. 1980).
- A Random Forest classifier uses a number of decision trees, in order to improve the classification rate.
- Boosting Trees can be used for regression-type and classification-type problems.

Used in data mining (most are included in R, see `rpart` and `party` packages, and in Weka, Waikato Environment for Knowledge Analysis)

Regression Trees





1. select variable
2. select threshold
3. for a given choice: the optimal choice of predictive variable is given by local average

Splitting the j attribute on θ

$$\mathcal{R}_1(j,\theta) = \{x \mid x_j \leq \theta\} \qquad \mathcal{R}_2(j,\theta) = \{x \mid x_j > \theta\}$$

$$\min_{j,\theta}\left[\min_{c_1}\sum_{x^i\in\mathcal{R}_1(j,\theta)}(y^i - c_1)^2 + \min_{c_2}\sum_{x^i\in\mathcal{R}_2(j,\theta)}(y^i - c_2)^2\right]$$

where $\min_{c_1}\sum_{x^i\in\mathcal{R}_1(j,\theta)}(y^i - c_1)^2$ is solved by

$$\hat{c_1} = \frac{1}{m}\sum_{i=1}^{m} y^i$$

Pruning

T_0 tree grown with stopping criterion the number of data points in the leaves.
$T \subseteq T_0$
$\tau = 1 \dots |T|$ number of leaf nodes

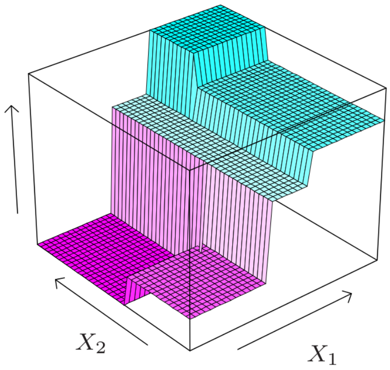
$$\hat{y}^i_\tau = \frac{1}{N_\tau} \sum_{x^i \in \mathcal{R}_\tau} y^i$$

$$Q_\tau(T) = \sum_{x^i \in \mathcal{R}_\tau} (y^i - \hat{y}^i)^2$$

pruning criterion: find T such that it minimizes:

$$C(T) = \sum_{\tau=1} |T| Q_\tau(T) + \lambda |T|$$

Disadvantage: piecewise-constant predictions with discontinuities at the split boundaries



Outline

To be written