

Learning to play 20 questions: Decision Trees

Hal Daumé III

CS5350: Machine Learning

28 August 2008

1. The decision tree model
2. Entropy and information theory
3. Building a decision tree
4. Overfitting with a decision tree
5. Extensions...

The decision tree model

We're going to represent classification programmatically:

Class	Outlook	Temperature	Windy?
Play	Sunny	Low	Yes
No play	Sunny	High	Yes
No play	Sunny	High	No
Play	Overcast	Low	Yes
Play	Overcast	High	No
Play	Overcast	Low	No
No play	Rainy	Low	Yes
Play	Rainy	Low	No

```

if Outlook=Sunny
then if Temperature=High
then return No play
else return Play
else if Temperature=Low
then return Play
else return No play
    
```

Here, we get 7/8 classification accuracy on the training data.

Building a decision tree: Be greedy!

We build a tree *top-down* in a greedy manner.

```

function BuildDT(data, labels)
if all the labels are the same
then return a tree that always assigns that label
else
let f be the feature that is best to split on
let l = BuildDT(data with f=0, labels with f=0)
let r = BuildDT(data with f=1, labels with f=1)
return tree(f,l,r)
    
```

Open questions:

1. Will this always terminate?
2. Which feature is best to split on?

Which features do you like?

Class	Outlook	Temperature	Windy?
Play	Sunny	Low	Yes
Play	Rainy	Low	No
Play	Overcast	Low	Yes
Play	Overcast	High	No
Play	Overcast	Low	No
No play	Sunny	High	Yes
No play	Sunny	High	No
No play	Rainy	Low	Yes

Foray into information theory...

see information theory notes on the course web site...

Entropy on our example

Class	Outlook	Temperature	Windy?
Play	Sunny	Low	Yes
No play	Sunny	High	Yes
No play	Sunny	High	No
Play	Overcast	Low	Yes
Play	Overcast	High	No
Play	Overcast	Low	No
No play	Rainy	Low	Yes
Play	Rainy	Low	No

$$H(\text{Class}) = -\frac{5}{8} \log_2 \frac{5}{8} - \frac{3}{8} \log_2 \frac{3}{8} = 0.95$$

$$H(\text{Outlook}) = -\frac{3}{8} \log_2 \frac{3}{8} - \frac{3}{8} \log_2 \frac{3}{8} - \frac{2}{8} \log_2 \frac{2}{8} = 1.41$$

Conditional entropy on our example

Class	Outlook	Temperature	Windy?
Play	Sunny	Low	Yes
No play	Sunny	High	Yes
No play	Sunny	High	No
Play	Overcast	Low	Yes
Play	Overcast	High	No
Play	Overcast	Low	No
No play	Rainy	Low	Yes
Play	Rainy	Low	No

$$H(\text{Class} \mid \text{Outlook}=\text{Sunny}) = -\frac{1}{3} \log_2 \frac{1}{3} - \frac{2}{3} \log_2 \frac{2}{3} = 0.14$$

$$H(\text{Class} \mid \text{Outlook}=\text{Overcast}) = -1 \log_2 1 - 0 \log_2 0 = 0$$

$$H(\text{Class} \mid \text{Outlook}=\text{Rainy}) = -\frac{1}{2} \log_2 \frac{1}{2} - \frac{1}{2} \log_2 \frac{1}{2} = 1$$

$$H(\text{Class} \mid \text{Outlook}) = \frac{3}{8} \times 0.14 + \frac{3}{8} \times 0 + \frac{2}{8} \times 1 = 0.303$$

Information gain on our example

Class	Outlook	Temperature	Windy?
Play	Sunny	Low	Yes
No play	Sunny	High	Yes
No play	Sunny	High	No
Play	Overcast	Low	Yes
Play	Overcast	High	No
Play	Overcast	Low	No
No play	Rainy	Low	Yes
Play	Rainy	Low	No

$$IG(\text{Class} \mid \text{Outlook}) = 0.95 - 0.303 = 0.648$$

$$IG(\text{Class} \mid \text{Temperature}) = 0.95 - 0.796 = 0.154$$

$$IG(\text{Class} \mid \text{Windy?}) = 0.95 - 0.906 = 0.044$$

Back to decision trees...

Building a decision tree: Be greedy!

Recall our old algorithm:

```

function BuildDT(data, labels)
  if all the labels are the same if some-base-case-occurs(data, labels)
    then return a tree that always assigns that label
  then return a tree that assigns the mode label
  else
    let f be the feature that is best to split on
    let f be the feature with highest IG
    let l = BuildDT(data with f=0, labels with f=0)
    let r = BuildDT(data with f=1, labels with f=1)
    return tree(f,l,r)
  
```

What are good base cases?

1. All labels are the same? **Okay!**
2. All data points (for unused features) are the same? **Okay!**
3. All features have $IG=0$? **Not okay!**

Why is $IG=0$ not a good base case?

Consider the following data:

0	0	0
1	0	1
1	1	0
0	1	1

One-level gives $IG=0$, but two levels will give perfect classification!

Overfitting in decision trees

- ▶ x_0 is a perfect classification feature
- ▶ Suppose we flip each Y with probability $\frac{1}{4}$
- ▶ Building a full tree will first split on x_0 and then recurse deeply
- ▶ Training error = 0, test error will be $\frac{3}{8}$
 - ▶ Doubly corrupted (1/16) and doubly uncorrupted (9/16) are good
 - ▶ All others (6/16) are bad
- ▶ This is **worse** than just predicting on the basis of x_0 alone!

Y	x_0	x_1	x_2	x_3	x_4
0	0	0	0	0	0
1	1	0	0	0	0
0	0	1	0	0	0
1	1	1	0	0	0
0	0	0	1	0	0
1	1	0	1	0	0
0	0	1	1	0	0
1	1	1	1	0	0
0	0	0	0	1	0
1	1	0	0	1	0
⋮	⋮	⋮	⋮	⋮	⋮

32 examples total

Decision tree pruning

Basic idea: introduce new “base cases” that prevent further growth

Several options:

- ▶ Stop when # of examples is too low – # is parameter
- ▶ Stop when my classification error is not much more than the average of my children – difference in error is parameter
- ▶ Stop when the remainder of the tree is no more likely than chance (use χ^2 test) – chance is parameter
- ▶ More...

Why can't we just estimate these parameters on the training data?

Decision tree extensions

1. Real valued inputs can be handled through *thresholding*
2. Real valued outputs can be handled by redefining entropy (easy)
3. Alternatives to information gain: Gini index, misclassification rate
4. Alternatives to greedy building?