

For Monday

- Finish chapter 19
- Homework
 - Chapter 18, exercise 3, 4, and 7.
 - Note that 7 is primarily a logic question.

Program 3

- Any questions?

Exam 2

- Next Friday
- Covers material through chapter 18
- Take home due at the exam, handed out Monday

Basic Decision Tree Algorithm

DTree(examples, attributes)

If all examples are in one category, return a leaf node with this category as a label.

Else if attributes are empty then return a leaf node labelled with the category which is most common in examples.

Else Pick an attribute, A, for the root.

For each possible value v_i for A

Let examples i be the subset of examples that have value v_i for A.

Add a branch out of the root for the test $A=v_i$.

If examples i is empty then

 Create a leaf node labelled with the category which is most common in examples

Else recursively create a subtree by calling

 DTree(examples i , attributes - {A})

Picking an Attribute to Split On

- Goal is to have the resulting decision tree be as small as possible, following Occam's Razor.
- Finding a minimal decision tree consistent with a set of data is NP-hard.
- Simple recursive algorithm does a greedy heuristic search for a fairly simple tree but cannot guarantee optimality.

What Is a Good Test?

- Want a test which creates subsets which are relatively “pure” in one class so that they are closer to being leaf nodes.
- There are various heuristics for picking a good test, the most popular one based on information gain (mutual information) originated with ID3 system of Quinlan (1979)

Entropy

- Entropy (impurity, disorder) of a set of examples, S , relative to a binary classification is:

$$\text{Entropy}(S) = -p_+ \log_2(p_+) - p_- \log_2(p_-)$$

where p_+ is the proportion of positive examples in S and p_- is the proportion of negatives.

- If all examples belong to the same category, entropy is 0 (by definition $0 \log(0)$ is defined to be 0).
- If examples are equally mixed ($p_+ = p_- = 0.5$) then entropy is a maximum at 1.0.

- Entropy can be viewed as the number of bits required on average to encode the class of an example in S , where data compression (e.g Huffman coding) is used to give shorter codes to more likely cases.
- For multiple-category problems with c categories, entropy generalizes to:

$$\text{Entropy}(S) = \sum -p_i \log_2(p_i)$$

where p_i is proportion of category i examples in S .

Information Gain

- The information gain of an attribute is the expected reduction in entropy caused by partitioning on this attribute:

$$\text{Gain}(S,A) = \text{Entropy}(S) - \sum (|S_v|/|S|) \text{Entropy}(S_v)$$

where S_v is the subset of S for which attribute A has value v and the entropy of the partitioned data is calculated by weighting the entropy of each partition by its size relative to the original set.

Information Gain Example

- Example:
 - big, red, circle: +
 - small, red, circle: +
 - small, red, square: -
 - big, blue, circle: -
- Split on size:
 - big: 1+, 1-, $E = 1$
 - small: 1+, 1-, $E = 1$
 - gain = $1 - ((.5)1 + (.5)1) = 0$
- Split on color:
 - red: 2+, 1-, $E = 0.918$
 - blue: 0+, 1-, $E = 0$
 - gain = $1 - ((.75)0.918 + (.25)0) = 0.311$
- Split on shape:
 - circle: 2+, 1-, $E = 0.918$
 - square: 0+, 1-, $E = 0$
 - gain = $1 - ((.75)0.918 + (.25)0) = 0.311$

Hypothesis Space in Decision Tree Induction

- Conducts a search of the space of decision trees which can represent all possible discrete functions.
- Creates a single discrete hypothesis consistent with the data, so there is no way to provide confidences or create useful queries.

Algorithm Characteristics

- Performs hill-climbing search so may find a locally optimal solution. Guaranteed to find a tree that fits any noise-free training set, but it may not be the smallest.
- Performs batch learning. Bases each decision on a batch of examples and can terminate early to avoid fitting noisy data.

Bias

- Bias is for trees of minimal depth; however, greedy search introduces a complication that it may not find the minimal tree and positions features with high information gain high in the tree.
- Implements a preference bias (search bias) as opposed to a restriction bias (language bias) like candidate-elimination.

Simplicity

- Occam's razor can be defended on the basis that there are relatively few simple hypotheses compared to complex ones, therefore, a simple hypothesis that is consistent with the data is less likely to be a statistical coincidence than finding a complex, consistent hypothesis.
- However,
 - Simplicity is relative to the hypothesis language used.
 - This is an argument for any small hypothesis space and holds equally well for a small space of arcane complex hypotheses, e.g. decision trees with exactly 133 nodes where attributes along every branch are ordered alphabetically from root to leaf.

Overfitting

- Learning a tree that classifies the training data perfectly may not lead to the tree with the best generalization performance since
 - There may be noise in the training data that the tree is fitting.
 - The algorithm might be making some decisions toward the leaves of the tree that are based on very little data and may not reflect reliable trends in the data.
- A hypothesis, h , is said to **overfit** the training data if there exists another hypothesis, h' , such that h has smaller error than h' on the training data but h' has smaller error on the test data than h .

Overfitting and Noise

- Category or attribute noise can cause overfitting.
- Add noisy instance:
 - `<<medium, green, circle>, +>` (really -)
- Noise can also cause directly conflicting examples with same description and different class.
Impossible to fit this data and must label leaf with majority category.
 - `<<big, red, circle>, ->` (really +)
- Conflicting examples can also arise if attributes are incomplete and inadequate to discriminate the categories.

Avoiding Overfitting

- Two basic approaches
 - **Prepruning**: Stop growing the tree at some point during construction when it is determined that there is not enough data to make reliable choices.
 - **Postpruning**: Grow the full tree and then remove nodes that seem to not have sufficient evidence.

Evaluating Subtrees to Prune

- Cross-validation:
 - Reserve some of the training data as a hold-out set (validation set, tuning set) to evaluate utility of subtrees.
- Statistical testing:
 - Perform some statistical test on the training data to determine if any observed regularity can be dismissed as likely to be random chance.
- Minimum Description Length (MDL):
 - Determine if the additional complexity of the hypothesis is less complex than just explicitly remembering any exceptions.

Beyond a Single Learner

- Ensembles of learners work better than individual learning algorithms
- Several possible ensemble approaches:
 - Ensembles created by using different learning methods and voting
 - Bagging
 - Boosting

Bagging

- Random selections of examples to learn the various members of the ensemble.
- Seems to work fairly well, but no real guarantees.

Boosting

- Most used ensemble method
- Based on the concept of a **weighted** training set.
- Works especially well with **weak** learners.
- Start with all weights at 1.
- Learn a hypothesis from the weights.
- Increase the weights of all misclassified examples and decrease the weights of all correctly classified examples.
- Learn a new hypothesis.
- Repeat