Decision Trees

CS/DSA 5970: Machine Learning Practice
Regression / Classification So Far…

• Input features must be numeric
  – Categorical variables required us to first transform them into numeral variables (via one-hot-encoding)

• Models have been based on continuous functions

• When we have many input features, it is hard to understand how the model works by just looking at the model parameters
Decision Trees

A new way of looking at model representation:
• Ask a recursive set of binary (or N-ary) questions
• Questions refer to specific input features
  – Input features can be numerical or categorical (enumerated data type)
• After the questions: produce a prediction
Decision Tree Example

- If more than 5 legs?
  - Yes: Is hiding under your bed?
    - Yes: Star of Charlotte's Web?
      - Yes: Mosquito!
      - No: Honeybee!
    - No: Makes honey?
      - Yes: Bed bug!
      - No: Star of Charlotte's Web?
    - No: Star of Charlotte's Web?
      - Yes: Spider!
  - No: Delicious?
    - Yes: On back of Australian 5-cent coin?
      - Yes: Star of Charlotte's Web?
        - Yes: Honeybee!
        - No: Mosquito!
      - No: Pig!
    - No: Bison!
- No: Delicious?
Decision Trees

Tree structure:

- A query starts at the root of the tree (the root node)
- A question node asks something specific about a feature in the query
- Depending on the answer, the query “falls down” one of the branches from the question
  - Often binary trees: we have “Yes” or “No” branches
- Each branch can contain additional questions (and branches)
- All paths end in leaf nodes, where the predictions are made
Partition a feature space
Example with 2 DOF
Show tree
Show sorting of queries
Example: animal tree
Decision Trees

• Each question cuts the feature subspace into two pieces
  – These cuts are axis-aligned

• Sequences of questions along a path to a leaf node are Boolean AND operators

• Branches are OR operators

• A tree sorts a set of queries into different leaf nodes
Types of Leaf Node Predictions

Different types of trees make different types of predictions:

• Standard tree: predict a single class
• Probability tree: predict a probability distribution over possible classes
• Regression tree: predict a continuous value
Tree Learning

CS/DSA 5970: Machine Learning Practice
Tree Learning

Incremental, greedy algorithm

• Start with an empty tree
• Grow tree by adding one new question + leaves
• Recompute the predictions
• Repeat
Generic Tree Learning Algorithm

- Training set: feature vectors + correct answer
- Initialize tree with root node and a leaf
  - All training samples are sorted into this leaf
  - Make a prediction that yields the “best” performance
Generic Tree Learning Algorithm

Until an adequate tree is learned:
• Pick best leaf node to replace with
  – A question
  – A pair of leaves (or more)
• Pick the best question to ask
  – Which feature?
  – Which value (or threshold value?)
• Pick the best prediction for each of the new leaves
Choosing the best next question

• With a small, finite set of discrete features, each with a small number of possible values, it is possible to consider all possible questions when evaluating the “best”

• More generally, we will sample from the possible question set
  – Training process becomes a stochastic one!
Probability Tree Learning

• Desired output: class
• Leaf nodes: probability distribution over the possible classes
• General idea: want each of the leaf nodes to contain a “pure” set of training examples
Probability Tree Learning (Intuition)

• Want leaf nodes to be as pure as possible
• Greedy algorithm:
  – Pick the leaf node with the highest impurity to expand
  – Pick the feature and dividing line that best distinguishes the classes
• The greedy algorithm can keep going to an extreme
  – This is the overfitting problem again!
Formalizing Probability Tree Learning

CS/DSC 5970: Machine Learning Practice
Formalizing Probability Tree Learning

Measuring purity of leaf nodes:
- Information content
- Gini Impurity
Very similar metrics

- Really is an empirical question as to which one to use

- Gini:
  - Less computation
  - Tries to place most frequent class into one of the main branches

- Information:
  - Better balance of trees
Combatting Overfitting

CS/DSA 5970: Machine Learning Practice
Model Parameters

• In our modeling work so far, each model had a fixed number of parameters

• We have also discussed the fact that as the complexity of a model increases, we often need more training data to effectively “tie down” the parameters
  – Otherwise, we run the risk of overfitting
Decision Tree Learning

• With trees, we need a distinct set of parameters for:
  – Each question (which feature & which threshold)
  – Each leaf node (e.g., the probability function)
• This means that, as the tree grows, the number of parameters increases
• In fact, a tree can grow beyond the structure that a training set can provide…
Tree Learning

We can allow our tree learning algorithm to execute until all leaves have zero entropy

- Most extreme case: one leaf per training set sample
- Effectively have a custom rule for every training sample

- Very brittle: the regions defined by the tree do not necessarily generalize to independent queries
Combatting Overfitting in Trees

Fundamental idea: constrain the complexity of the tree
• But: what is the most effective way to do this?
Regularization in Trees

Different tree learning algorithms make different choices. Key ideas:

• Limit the maximum depth of the tree
  – At the limit, forces balanced trees
• Limit the number of leaf nodes
  – Allows more unbalanced trees
Sample-driven decisions for splitting a leaf node:

- Require a minimum number of samples in a leaf node before allowing it to be expanded
- Proposed split must result in a measurable improvement in performance. Possibilities:
  - Entropy change; Gini Score
  - Likelihood Ratio test
  - Crisp classification: Chi-squared test
Example: Probability Tree Learning

CS/DSA 5970: Machine Learning Practice
Example: Probability Tree Learning

• 2D classification problem (from SVM data)
• Baby action recognition
  – Adjust the positive examples:
    • Samples leading up to event are now considered positive
    • Dropping samples immediately after events
Example: Probability Tree Learning

Live example
Regression Trees

CS/DSA 5970: Machine Learning Practice
Regression Trees

Leaf nodes output a continuous value:

• Simple (most common) form: samples that fall into a particular leaf node are assigned the same value (i.e., constant function)
  – Yields a piecewise-constant function

• More general case: output is some function of the full feature vector
  – Each leaf node has its own function
  – So, more expressive than using the function over the entire training set
Example: Regression Trees

CS/DSA 5970: Machine Learning Practice
Regression Trees

Live example