Machine Learning and Data Mining

Decision Trees

Prof. Alexander Ihler
Decision trees

- “Split” input into cases
  - Usually based on a single variable
  - Recurse down until we reach a decision
  - Continuous vars: choose split point
Decision trees

- Categorical variables
  - Could have a child per value
  - Binary tree: split values into two sets

The discrete variable will not appear again below here…

Could appear again multiple times…

(This ^^^ is easy to implement using a 1-of-K representation…)
Decision trees

- “Complexity” of function depends on the depth

- A depth-1 decision tree is called a decision “stump”
  - Simpler than a linear classifier!

\[ X_1 > 0.5? \]
Decision trees

- “Complexity” of function depends on the depth

Depth $d = \text{up to } 2^d \text{ regions & predictions}
Decision trees for regression

- Exactly the same
- Predict real valued numbers at leaf nodes

Examples on a single scalar feature:

Depth 1 = 2 regions & predictions

Depth 2 = 4 regions & predictions

...
Machine Learning and Data Mining

Learning Decision Trees

Prof. Alexander Ihler
Learning decision trees

• Break into two parts
  – Should this be a leaf node?
  – If so: what should we predict?
  – If not: how should we further split the data?

• Leaf nodes: best prediction given this data subset
  – Classify: pick majority class; Regress: predict average value

• Non-leaf nodes: pick a feature and a split
  – Greedy: “score” all possible features and splits
  – Score function measures “purity” of data after split
    • How much easier is our prediction task after we divide the data?

• When to make a leaf node?
  – All training examples the same class (correct), or indistinguishable
  – Fixed depth (fixed complexity decision boundary)
  – Others …

Example algorithms: ID3, C4.5
See e.g. wikipedia, “Classification and regression tree”
Scoring decision tree splits

- Suppose we are considering splitting feature 1
  - How can we score any particular split?
  - “Impurity” – how easy is the prediction problem in the leaves?
- “Greedy” – could choose split with the best accuracy
  - Assume we have to predict a value next
  - MSE (regression)
  - 0/1 loss (classification)
- But: “soft” score can work better

![Graph showing decision tree split](image)
Entropy and Information

• “Entropy” is a measure of randomness
  – How hard is it to communicate a result to you?
  – Depends on the probability of the outcomes

• Communicating fair coin tosses
  – Output: H H T H T T H H H H T …
  – Sequence takes n bits – each outcome totally unpredictable

• Communicating my daily lottery results
  – Output: 0 0 0 0 0 0 …
  – Most likely to take one bit – I lost every day.
  – Small chance I’ll have to send more bits (won & when)

• Takes less work to communicate because it’s less random
  – Use a few bits for the most likely outcome, more for less likely ones

Lost: 0
Won 1: 1(…)0
Won 2: 1(…)1(…)0
Entropy and Information

- Entropy $H(x) \equiv E[\ log\ 1/p(x)\ ] = \sum p(x) \log\ 1/p(x)$
  - Log base two, units of entropy are “bits”
- Examples:

$$H(x) = .25 \ log\ 4 + .25 \ log\ 4 + .25 \ log\ 4 + .25 \ log\ 4$$

$$= \ log\ 4 = 2\ bits$$
Entropy and Information

- Entropy $H(x) \equiv \mathbb{E}[\log 1/p(x)] = \sum p(x) \log 1/p(x)$
  - Log base two, units of entropy are “bits”
- Examples:

  $H(x) = \frac{1}{2} \log 4 + \frac{1}{2} \log 4 + \frac{1}{2} \log 4 + \frac{1}{2} \log 4$  
  $= \log 4 = 2 \text{ bits}$

  $H(x) = \frac{3}{4} \log \frac{4}{3} + \frac{1}{4} \log 4$  
  $\approx .8133 \text{ bits}$
Entropy and Information

- Entropy $H(x) \equiv \mathbb{E}[ \log \frac{1}{p(x)} ] = \sum p(x) \log \frac{1}{p(x)}$
  - Log base two, units of entropy are “bits”

- Examples:

  \[
  H(x) = .25 \log 4 + .25 \log 4 + .25 \log 4 + .25 \log 4
  = \log 4 = 2 \text{ bits}
  \]

  \[
  H(x) = .75 \log \frac{4}{3} + .25 \log 4
  \approx .8133 \text{ bits}
  \]

  \[
  H(x) = 1 \log 1
  = 0 \text{ bits}
  \]

Max entropy for 4 outcomes

Min entropy
Entropy and Information

• Information gain
  – How much is entropy reduced by measurement?
• Information: expected information gain

\[ H = .99 \text{ bits} \]
Entropy and Information

- Information gain
  - How much is entropy reduced by measurement?
- Information: expected information gain

$H = 0.99 \text{ bits}$

$H = 0.77 \text{ bits}$

$\text{Prob} = 13/18$

$H = 0$

$\text{Prob} = 5/18$
Entropy and Information

- Information gain
  - How much is entropy reduced by measurement?
- Information: expected information gain

Information gain

Information = \( \frac{13}{18} \times (0.99 - 0.77) + \frac{5}{18} \times (0.99 - 0) \)

Equivalent:

\[
\sum p(s,c) \log \left( \frac{p(s,c)}{p(s)p(c)} \right) = \frac{10}{18} \log \left( \frac{10/18}{13/18 \times 10/18} \right) + \frac{3}{18} \log \left( \frac{3/18}{13/18 \times 8/18} \right) + \ldots
\]
Entropy and Information

- Information gain
  - How much is entropy reduced by measurement?

- Information: expected information gain
Entropy and Information

- Information gain
  - How much is entropy reduced by measurement?
- Information: expected information gain

Information gain: $\text{Information} = \frac{17}{18} \times (0.99 - 0.97) + \frac{1}{18} \times (0.99 - 0)$

Less information reduction – a less desirable split of the data
Gini index / impurity

- An alternative to information gain
  - Measures variance in the allocation (instead of entropy)

- \( H_{\text{gini}} = \sum_c p(c) (1-p(c)) \) vs. \( H_{\text{ent}} = - \sum_c p(c) \log p(c) \)

\[
\begin{align*}
H_g &= 0 \\
\text{Prob} &= 5/18 \\
H_g &= .355 \\
\text{Prob} &= 13/18 \\
H_g &= .494
\end{align*}
\]

Gini Index = \( 13/18 \times (.494-.355) + 5/18 \times (.494 - 0) \)
Entropy vs Gini index

- The two are nearly the same...
  - Pick whichever one you like
For regression

- Most common is to measure variance reduction
  - Equivalent to “information gain” in a Gaussian model…

\[
\text{Var reduction} = 4/10 \times (.25 - .1) + 6/10 \times (.25 - .2)
\]
Building a decision tree

- Pseudo-code

```python
decisionTreeSplitData(X, Y)
    if (stopping condition) return decision for this node
    For each possible feature
        For each possible split
            (for cts features: sort & compute split points)
                Score the split (e.g. information gain)
            Pick the feature & split with the best score
            Split the data at that point
            Recurse on each subset
```

Stopping conditions:
- # of data < K
- Depth > D
- All data indistinguishable (discrete features)
- Prediction sufficiently accurate
Building a decision tree

• Pseudo-code

```python
def decisionTreeSplitData(X, Y):
    if (stopping condition) return decision for this node
    For each possible feature
        For each possible split
            (for cts features: sort & compute split points)
            Score the split (e.g. information gain)
            Pick the feature & split with the best score
            Split the data at that point
            Recurse on each subset
```

Stopping criteria:
• Information gain threshold? Often not a good idea…

No single split improves performance, but two splits together is accurate

*Instead: grow a large tree and prune back, using training or validation data*
Controlling complexity

- Maximum depth cutoff
Controlling complexity

- Minimum # parent data

\[ \text{minParent 1} \]

\[ \text{minParent 3} \]

\[ \text{minParent 5} \]

\[ \text{minParent 10} \]
Decision trees in Python

- Many implementations
- Class implementation:
  - real-valued features (can use 1-of-k for discrete)
  - Uses entropy (easy to extend)

```python
T = dt.treeClassify()
T.train(X,Y,maxDepth=2)
print T

if x[0] < 5.602476:
    if x[1] < 3.009747:
        Predict 1.0  # green
    else:
        Predict 0.0  # blue
else:
    if x[0] < 6.186588:
        Predict 1.0  # green
    else:
        Predict 2.0  # red

ml.plotClassify2D(T, X,Y)
```
Summary

• Decision trees
  – Flexible functional form
  – At each level, pick a variable and split condition
  – At leaves, predict a value

• Learning decision trees
  – Score all splits & pick best
    • Classification: Information gain, Gini index
    • Regression: Expected variance reduction
  – Stopping criteria

• Complexity depends on depth
  – Decision stumps: very simple classifiers