Decision Tree
Pruning

M. R. Hasan
Readings

- Bishop: 14.4
- Murphy: 16, 16.1, 16.2.1, 16.2.2, 16.2.3, 16.2.4
- Geron: chapter 6
What We Will Cover

• Growing Decision Tree (Stopping Criteria)
• Overfitting Problem
• Hyperparameter Tuning
• Tree pruning
• Chi-Square Test
• Computational Complexity
• Limitations
• Decision Tree for Regression
Training Decision Tree

• Next question that we need to resolve is how do we decide **when to stop** growing the tree?

• Growing is based on a **recursive algorithm**.
• Therefore, to stop the recursion we need to define stopping criteria.
• What is the **Stopping Criterion**?
Training Decision Tree

• Stopping Criteria:
• In a recursive algorithm, the stopping criteria is defined by the base cases that stops recursion.
• There are 3 Base Cases.
• **Base Case 1**: If all records in current data subset has the same output, then don’t recurse
• **Base Case 2**: If all records have exactly the same set of input attributes, then don’t recurse
• **Base Case 3**: If all attributes have zero information gain then don’t recurse.
Stopping Criteria: Base Case 1

Don’t split a node if all matching records have the same output value
Stopping Criteria: Base Case 2 (No Attribute can distinguish)

If all records have exactly the same set of input attributes, then don’t recurse.
Training Decision Tree: Base Case 3

- **Base Case 3:** If all attributes have zero information gain then don’t recurse.

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ y = a \text{ XOR } b \]

![The resulting decision tree:](image)

The information gains:

<table>
<thead>
<tr>
<th>Information gains using the training set (4 records)</th>
</tr>
</thead>
<tbody>
<tr>
<td>y values: 0 1</td>
</tr>
<tr>
<td>Input  Value Distribution Info Gain</td>
</tr>
<tr>
<td>a 0</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>b 0</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>

However, this base case may not produce an optimal tree!
Training Decision Tree: Base Case 3

- If we **omit base case 3**, then the resulting decision tree would be:

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

\[
y = a \text{ XOR } b
\]
Training Decision Tree: Base Case 3

- The **problem of base case 3** is:
- It stops us from recognizing situations where there is **no one good attribute**, but there are **combinations of attributes that are informative** (e.g., $a = 0 \& b = 0$).

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

$y = a \text{ XOR } b$
Training Decision Tree: Base Case 3

- If there are roughly equal number of examples for all four combinations of input values, then neither attribute will be informative.
- Yet the correct thing to do is to split on one of the attributes (it doesn’t matter which one).
- And then at the second level we will get splits that are informative.

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ y = a \text{ XOR } b \]
Training Decision Tree: Base Case 3

• That’s why instead of early stopping it’s better to grow the tree fully and then prune.

Before we discuss pruning technique, let’s see how to optimize the tree training (art of growing tree)
Decision Tree: Hyperparameters to Handle Overfitting
Training Decision Tree: Overfitting

- Decision tree is based on a **greedy** algorithm.
- It can easily **overfit** the data.
- How do we prevent overfitting?
- We need to **tune the hyperparameters**.
Training Decision Tree: Overfitting

• What are the possible hyperparameters for controlling overfitting?

• Let’s look at the variation of one hyperparameter to see how it influences the overfitting issue.
Training Decision Tree: Hyperparameters

• Hyperparameter: Minimum number of samples a leaf must have
Training Decision Tree: Hyperparameters

- There are several regularization hyperparameters that can be used to control overfitting.
- Maximum depth of the tree
  - Reducing the depth will reduce the overfitting.
- Minimum number of samples a node must have before split
- Minimum number of samples a leaf must have
- Maximum number of leaf nodes
- Maximum number of features that are evaluated for splitting at each node
Decision Tree: Training Issues
Training Decision Tree

• So far we have discussed the following two aspects of growing/training a decision tree using the CART algorithm:
  • How to choose the best feature/attribute?
  • How to determine when to stop growing the tree?
However, we have argued that often **none of the available splits produces a significant reduction in error**, and yet **after several more splits** a substantial error reduction is found.

\[
y = a \text{ XOR } b
\]

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

![The information gains diagram](image)

![Decision tree diagram](image)
Training Decision Tree

- For this reason, it is common practice to **grow a large tree**, using a **stopping criterion** based on the number of data points associated with the leaf nodes, and then prune back the resulting tree.

\[
\begin{array}{ccc}
 a & b & y \\
 0 & 0 & 0 \\
 0 & 1 & 1 \\
 1 & 0 & 1 \\
 1 & 1 & 0 \\
\end{array}
\]

\[y = a \text{ XOR } b\]
Decision Tree: Pruning
Decision Tree: Pruning

- We start with a full tree.
- We then look at a test node that has only leaf nodes as descendants.
- A node whose children are all leaf nodes is considered unnecessary if the purity improvement it provides is not statistically significant.
Standard statistical tests, such as the $\chi^2$ test, are used to estimate the probability that the improvement is purely the result of chance (which is called the null hypothesis).
Decision Tree: Pruning

- If this probability (the p-value) is **higher than a given threshold** (typically 5%), then the **node is considered unnecessary** and is replaced by a leaf node.
- The pruning continues until **all unnecessary nodes have been pruned**.

If you don’t have background in the $\chi^2$ test, then review the next few slides (Title: The Chi-Square Test”).
The Chi-Square Test

- The $\chi^2$ test determines whether the deviations (differences between observed and expected) are the result of chance, or were they due to other factors.
- The chi-square test is always testing what scientists call the null hypothesis.
- The null hypothesis states that there is no significant difference between the expected and observed result.
The Chi-Square Test

• Chi-square is the sum of the squared difference between observed (o) and the expected (e) data (or the deviation, d), divided by the expected data in all possible categories.

• The formula for calculating $\chi^2$ value:

  $$(o - e)^2/e$$

• It is summed over all possible categories.
The Chi-Square Test: Step-by-Step

• **Step-by-Step Procedure** for Testing Your Hypothesis and Calculating Chi-Square:
  
  1. State the null hypothesis: “*There is no underlying pattern. Hence, the attribute is irrelevant and should be pruned*”.
  
  2. Determine the *expected numbers for each observational class*. Remember to use numbers, not percentages.
  
  3. Calculate the $\chi^2$ value using the formula. Complete all calculations to three significant digits. Round off your answer to two significant digits.

The formula for calculating $\chi^2$ value:

$$ (o - e)^2 / e $$
The Chi-Square Test: Step-by-Step

• 4. Use the chi-square distribution table to determine significance of the value.
• Determine degrees of freedom (one minus the number of values of the variable) and locate the value in the appropriate column.
• Locate the value closest to your calculated $\chi^2$ value on that degrees of freedom df row.
• Move up the column to determine the p value.
The Chi-Square Test: Step-by-Step

• 5. State your conclusion in terms of your hypothesis.
  - If the p value for the calculated $\chi^2$ is $p > 0.05$, accept your null hypothesis.
  - The deviation is small enough that chance alone accounts for it.
  - A p value of 0.6, for example, means that there is a 60% probability that any deviation from expected is due to chance only.
  - This is within the range of acceptable deviation.

In our problem, it means that there is no pattern in the data and that the attribute is irrelevant (hence, prune it).
The Chi-Square Test: Step-by-Step

• 5. State your conclusion in terms of your hypothesis.
  - If the p value for the calculated $\chi^2$ is $p < 0.05$, **reject your null hypothesis**.
  - Conclude that some factor other than chance is operating for the deviation to be so great.
  - For example, a p value of 0.01 means that there is only a 1% chance that this deviation is due to chance alone.
  - Therefore, **other factors** must be involved.

In our problem, it means that there is a pattern in the data and that the attribute is relevant for distinguishing the pattern (hence, **no pruning**).
Decision Tree: Pruning

• If the test node appears to be irrelevant (detecting only noise in the data) then we **eliminate the test**, replacing it with a leaf node.

• We repeat this process, **considering each test with only leaf descendants**, until each one has either been pruned or accepted as is.
Decision Tree: Pruning

• The question is, how do we detect that a node is testing an irrelevant attribute?

• Suppose we are at a node consisting of $p$ positive and $n$ negative examples.

• If the attribute is irrelevant, we would expect that it would split the examples into subsets that each have roughly the same proportion of positive examples as the whole set, $p/(p + n)$.

• So the information gain will be close to zero.
Decision Tree: Pruning

• Thus, the information gain is a **good clue to irrelevance**.

• Now the question is, **how large a gain** should we require in order to split on a particular attribute?
Decision Tree: Pruning

- This is where the **chi-square statistical significance test** proves useful.
- This test begins by assuming that there is no underlying pattern (the so-called **null hypothesis**).
- Then the actual data are analyzed to calculate the **extent to which they deviate** from a perfect absence of pattern.
In this case, the null hypothesis is that the attribute is irrelevant and, hence, that the information gain for an infinitely large sample would be zero.

We need to calculate the probability that, under the null hypothesis, a sample of size \( v = n + p \) would exhibit the observed deviation from the expected distribution of positive and negative examples.
Decision Tree: Pruning

• We can measure the deviation by comparing the actual numbers of positive and negative examples in each subset, \( p_k \) and \( n_k \), with the expected numbers, \( \hat{p}_k \) and \( \hat{n}_k \), assuming true irrelevance:

\[
\hat{p}_k = p \times \frac{p_k + n_k}{p + n} \quad \hat{n}_k = n \times \frac{p_k + n_k}{p + n}.
\]

Light boxes: WillWait = Yes
Dark boxes: WillWait = No
Decision Tree: Pruning

- Example:
- Attribute “Type” has 4 values (3 degrees of freedom).
- Calculate the expected numbers for each category of 4 values.

\[
\hat{p}_k = p \times \frac{p_k + n_k}{p + n} \quad \hat{n}_k = n \times \frac{p_k + n_k}{p + n}.
\]

<table>
<thead>
<tr>
<th>Type</th>
<th>French</th>
<th>Italian</th>
<th>Thai</th>
<th>Burger</th>
</tr>
</thead>
<tbody>
<tr>
<td>p(_k)(hat)</td>
<td>([6*(1 + 1)]/12 = 1)</td>
<td>([6*(1 + 1)]/12 = 1)</td>
<td>([6*(2 + 2)]/12 = 2)</td>
<td>([6*(2 + 2)]/12 = 2)</td>
</tr>
<tr>
<td>n(_k)(hat)</td>
<td>([6*(1 + 1)]/12 = 1)</td>
<td>([6*(1 + 1)]/12 = 1)</td>
<td>([6*(2 + 2)]/12 = 2)</td>
<td>([6*(2 + 2)]/12 = 2)</td>
</tr>
</tbody>
</table>
Decision Tree: Pruning

- We measure of the $\chi^2$ value or the total deviation as follows:

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

- Example:
- **Total deviation** for the attribute “Type” that has 4 values (3 degrees of freedom).

\[
\Delta = \sum_{k=1}^{d} \left( \frac{(p_k - \hat{p}_k)^2}{\hat{p}_k} + \frac{(n_k - \hat{n}_k)^2}{\hat{n}_k} \right).
\]

<table>
<thead>
<tr>
<th></th>
<th>French</th>
<th>Italian</th>
<th>Thai</th>
<th>Burger</th>
</tr>
</thead>
<tbody>
<tr>
<td>((p_k - \hat{p}_k)^2/p_k)</td>
<td>((1 - 1)^2/1 = 0)</td>
<td>((1 - 1)^2/1 = 0)</td>
<td>((2 - 2)^2/2 = 0)</td>
<td>((2 - 2)^2/2 = 0)</td>
</tr>
<tr>
<td>((n_k - \hat{n}_k)^2/n_k)</td>
<td>((1 - 1)^2/1 = 0)</td>
<td>((1 - 1)^2/1 = 0)</td>
<td>((2 - 2)^2/2 = 0)</td>
<td>((2 - 2)^2/2 = 0)</td>
</tr>
</tbody>
</table>

\(\Delta = 0\)
### Decision Tree: Pruning

- Under the null hypothesis, the value of $\Delta$ is distributed according to the $\chi^2$ (chi-squared) distribution with $v - 1$ degrees of freedom.

- We can use a $\chi^2$ table or a standard statistical library routine to see if a particular $\Delta$ value confirms or rejects the null hypothesis.

\[
\Delta = \sum_{k=1}^{d} \frac{(p_k - \hat{p}_k)^2}{\hat{p}_k} + \frac{(n_k - \hat{n}_k)^2}{\hat{n}_k}.
\]
Decision Tree: Pruning

• For example, consider the restaurant “Type” attribute, with four values and thus 3 degrees of freedom.
• For 0 deviation, p-value is greater than 99.5%.
• A value of $\Delta = 7.82$ or more would reject the null hypothesis at the 5% level.
• Hence, our null hypothesis is confirmed.
• The attribute is irrelevant and should be pruned.

Light boxes: WillWait = Yes
Dark boxes: WillWait = No

$\Delta = 0$
Let’s look at another example.

Attribute “Patrons” has 3 values (2 degrees of freedom).

Calculate the expected numbers for each category of 3 values.

\[
\hat{p}_k = p \times \frac{p_k + n_k}{p + n}, \quad \hat{n}_k = n \times \frac{p_k + n_k}{p + n}.
\]
Decision Tree: Pruning

- Example:
- **Total deviation** for the attribute “Patrons” that has 3 values (2 degrees of freedom).

\[
\Delta = \sum_{k=1}^{d} \left( \frac{(p_k - \hat{p}_k)^2}{\hat{p}_k} + \frac{(n_k - \hat{n}_k)^2}{\hat{n}_k} \right).
\]

<table>
<thead>
<tr>
<th></th>
<th>None</th>
<th>Some</th>
<th>Full</th>
</tr>
</thead>
<tbody>
<tr>
<td>((p_k - p^\wedge_k)^2/p^\wedge_k)</td>
<td>((0 - 2/3)^2/2/3 = 2/3)</td>
<td>((4 - 2)^2/2 = 2)</td>
<td>((2 - 3)^2/3 = 1/3)</td>
</tr>
<tr>
<td>((n_k - n^\wedge_k)^2/n^\wedge_k)</td>
<td>((2 - 2/3)^2/2/3 = 8/3)</td>
<td>((0 - 2)^2/2 = 2)</td>
<td>((4 - 3)^2/3 = 1/3)</td>
</tr>
</tbody>
</table>

\[\Delta = 2/3 + 8/3 + 2 + 2 + 1/3 + 1/3 = 8\]
Decision Tree: Pruning

• For the “Patrons” attribute, with 3 values (2 degrees of freedom) $\Delta = 8$, p-value < 5%
• A value of $\Delta = 5.991$ or more would reject the null hypothesis at the 5% level.
• Hence, we **reject** the null hypothesis.
• The attribute is not irrelevant and should be retained.
Decision Tree: Pruning

• Comparison between the “Type” and “Patrons” attribute.
• “Type”: $\Delta = 0$ [reject null if $\Delta \geq 7.82$ at the 5% level].
• “Patrons”: $\Delta = 8$ [reject null if $\Delta \geq 5.991$ at the 5% level].
• Conclusion: “Type” attribute is irrelevant, hence pruned.
Decision Tree: Pruning

- **Pruning demo.**
- Consider two of the four features from the **3-class iris dataset.**
Decision Tree: Pruning

- Pruning demo.
- The resulting tree and the decision boundaries are shown.
- We see that the tree is quite complex, as are the resulting decision boundaries.
Decision Tree: Pruning

• Pruning demo (misclassification rate is used in cross-validation).
• Following we show that the CV estimate of the error is much higher than the training set error, indicating overfitting.
• Hence we need to do pruning!
Therefore, the **standard approach** is:

- To grow a “full” tree, and then
- To **perform pruning**.

This can be done using a scheme that **prunes the branches giving the least increase in the error**.
Decision Tree: Pruning

• **To determine how far to prune back**, we can evaluate the cross-validated error on each such subtree, and then **pick the tree whose CV error is within 1 standard error of the minimum**.

• The point with the minimum CV error corresponds to the simple tree in the Figure.
Decision Tree: Pruning

- **Comparison**: unpruned vs. pruned tree.
Decision Tree: Key Limitations
Decision Tree: Limitations

• CART models are **popular for several reasons**.

• They are **easy to interpret**, they can easily handle mixed discrete and continuous inputs.

• They perform **automatic variable selection**.

• They are relatively **robust to outliers**.

• They scale well to large data sets.

• They can be modified to handle missing inputs.
Decision Tree: Limitations

• CART models also have **some limitations**.

• The primary one is that they **do not predict very accurately** compared to other kinds of model.

• This is in part due to the **greedy** nature of the tree construction algorithm.
A related problem is that trees are unstable:

Small changes to the input data can have large effects on the structure of the tree.
Decision Tree: Limitations

• In practice it is found that the particular tree structure that is learned is very sensitive to the details of the data set.

• Decision Trees are very sensitive to small variations in the training data.
Decision Tree: Limitations

- A small change to the training data can result in a very different set of splits.
Decision Tree: Limitations

- This is due to the hierarchical nature of the tree-growing process,
  causing errors at the top to affect the rest of the tree.
Decision Tree: Limitations

• In frequentist terminology, we say that trees are high variance estimators.

• This high variance problem is solved by growing forest (combining multiple trees)
Decision Tree: Computational Complexity
Decision Tree: Computational Complexity

• Making predictions requires **traversing the Decision Tree** from the root to a leaf.

• Decision Trees are generally **approximately balanced**.

• So **traversing** the Decision Tree requires going through roughly \( O(\log_2 n) \) nodes [number of samples is \( n \)].
Decision Tree: Computational Complexity

• Since each node only requires checking the value of one feature, the overall prediction complexity is just $O(\log_2 n)$.

• It is independent of the number of features.

• So predictions are very fast, even when dealing with large training sets.
Decision Tree: Computational Complexity

• However, the training algorithm compares all features on all samples at each node.

• This results in a training complexity of \( O(d \times n \log_2 n) \). [d represents the number of features]

• It slows down training considerably for larger training sets.
Decision Tree: Regression
Decision Tree: Regression

- A regression tree splits a complex nonlinear regression problem into a set of smaller problems which can be more easily handled by simpler models.
Decision Tree: Regression

- The first job of a decision tree is to decide **which branch to direct the incoming data to**.
- But when the data reaches a terminal node then that leaf needs to make a prediction.
Decision Tree: Regression

- The actual form of the prediction depends on the prediction model.
- We have a few alternatives.
- Different possible predictor models.
- (a) Constant. (b) Polynomial and linear.
Decision Tree: Regression

- For instance we could use a **polynomial function** of a subspace of the input $x$.
- In the low dimensional example in the figure a generic polynomial model corresponds to $y(x) = \sum w_i x^i$
- This simple model also captures the linear and constant model.
Decision Tree: Summary

- Decision trees are one of the most popular data mining tools
- Easy to understand
- Easy to implement
- Easy to use
- Computationally cheap (to solve heuristically)
Decision Tree: Summary

- Information gain/Gini index to select attributes (CART, ID3, C4.5, etc.)
- Presented for classification, can be used for regression and density estimation too.
- Decision trees will overfit!!!
- Zero bias classifier with lots of variance
- Must use tricks to find “simple trees”, e.g.,
  - Fixed depth/Early stopping
  - Pruning
  - Hypothesis testing (chi-square test)