



Decision Trees:



Overfitting

CS229: Machine Learning
Carlos Guestrin
Stanford University

Slides include content developed by and co-developed with Emily Fox

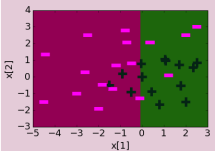
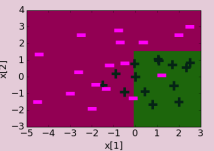
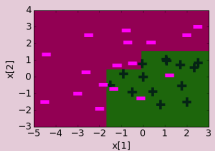
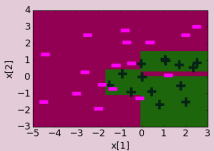
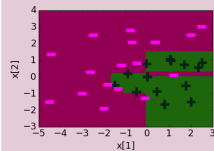
©2022 Carlos Guestrin

Overfitting in decision trees

What happens when we increase depth?

Training error reduces with depth



| Tree depth | depth = 1 | depth = 2 | depth = 3 | depth = 5 | depth = 10 |
|-------------------|--|---|--|--|--|
| Training error | 0.22 | 0.13 | 0.10 | 0.03 | 0.00 |
| Decision boundary |  |  |  |  |  |

Two approaches to picking simpler trees

1. **Early Stopping:**

Stop the learning algorithm **before** tree becomes too complex

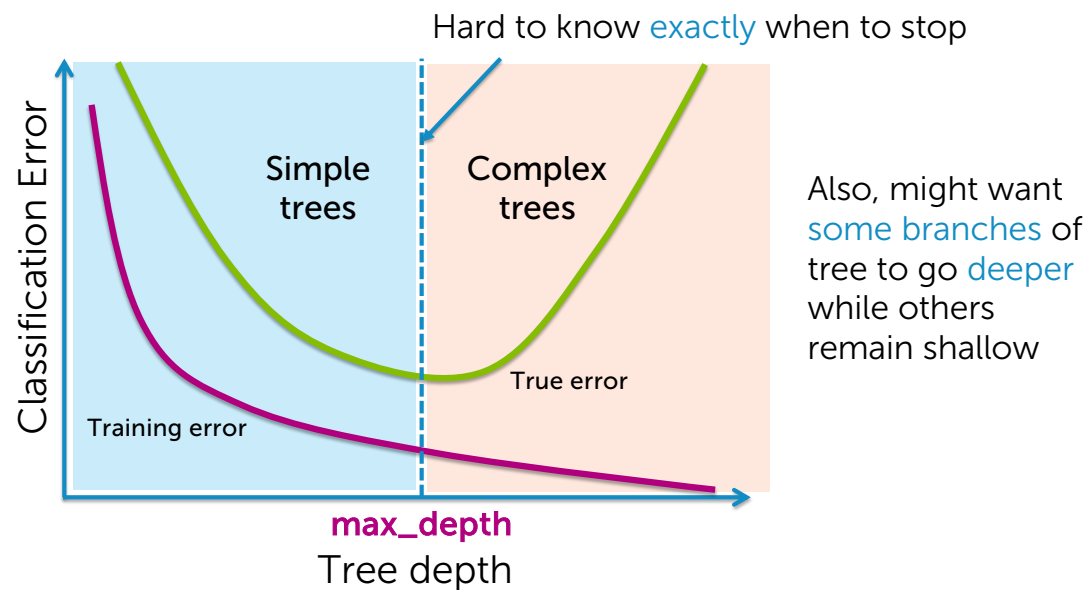
2. **Pruning:**

Simplify the tree **after** the learning algorithm terminates

Technique 1: Early stopping

- Stopping conditions (recap):
 1. All examples have the same target value
 2. No more features to split on
- Early stopping conditions:
 1. Limit tree depth (choose *max_depth* using validation set)
 2. Do not consider splits that do not cause a sufficient decrease in classification error
 3. Do not split an intermediate node which contains too few data points

Challenge with early stopping condition 1



Early stopping condition 2: Pros and Cons

- **Pros:**
 - A reasonable heuristic for early stopping to avoid useless splits
- **Cons:**
 - **Too short sighted:** We may miss out on “good” splits may occur right after “useless” splits
 - Saw this with “**xor**” example

Two approaches to picking simpler trees

1. **Early Stopping:**

Stop the learning algorithm **before** tree becomes too complex

2. **Pruning:**

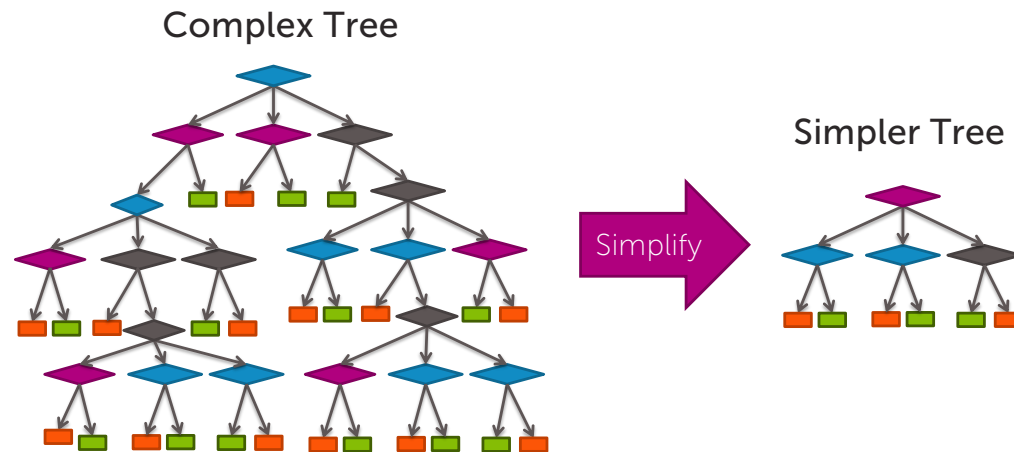
Simplify the tree **after** the learning algorithm terminates



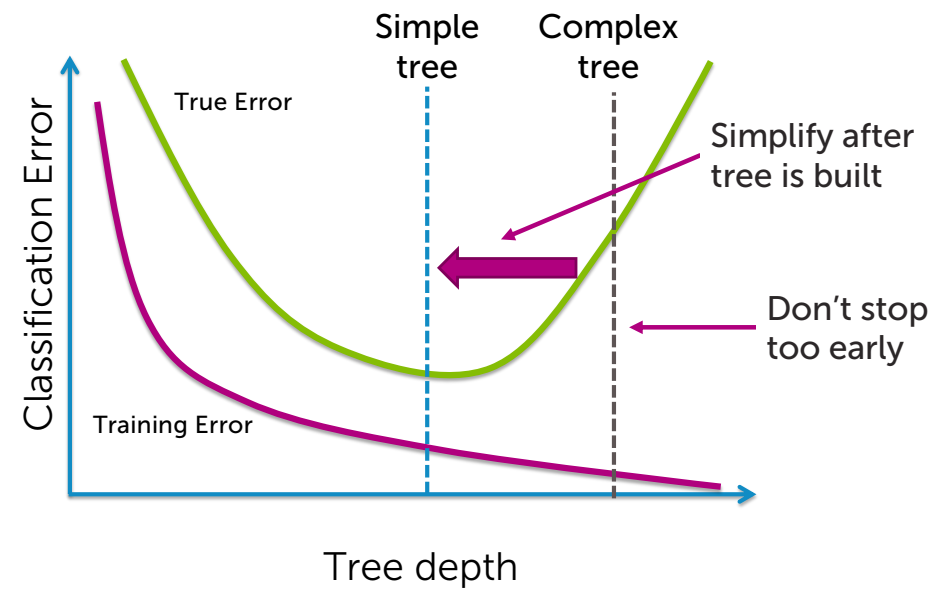
Complements early stopping

Pruning: *Intuition*

Train a complex tree, simplify later




Pruning motivation



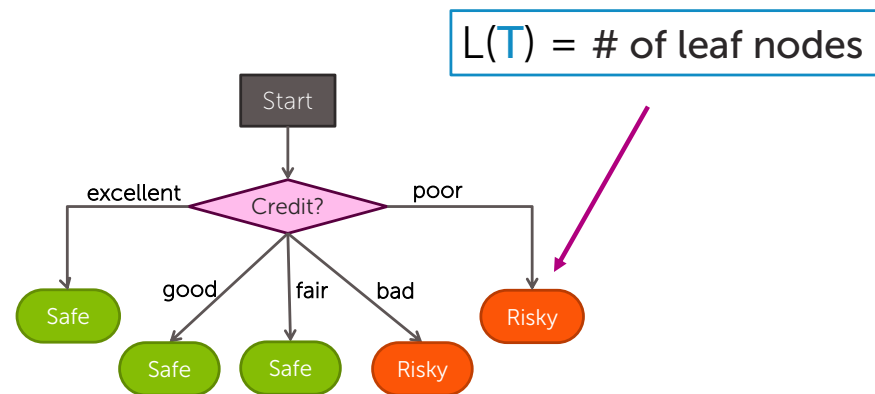
Scoring trees: Desired total quality format

Want to balance:

- i. How well tree fits data
- ii. Complexity of tree

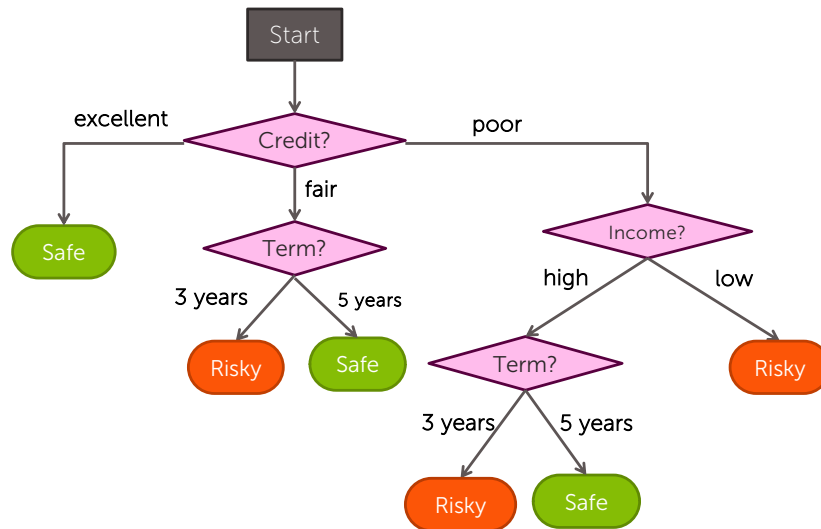
Total cost = 
measure of fit + measure of complexity

Simple measure of complexity of tree



Balance simplicity & predictive power

Too complex, risk of overfitting



Too simple, high classification error



Balancing fit and complexity

$$\text{Total cost } C(\mathbf{T}) = \text{Error}(\mathbf{T}) + \lambda L(\mathbf{T})$$


tuning parameter

If $\lambda=0$:

If $\lambda=\infty$:

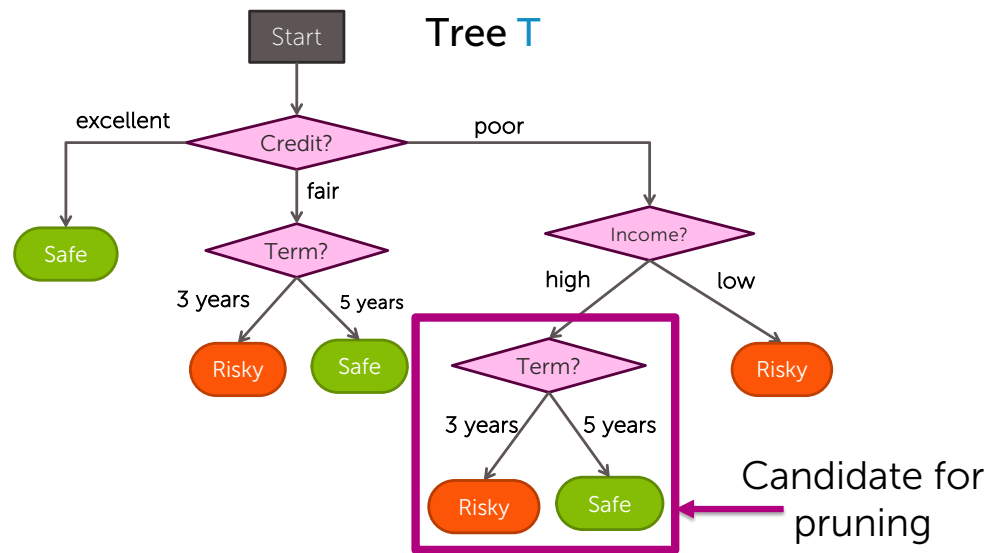
If λ in between:



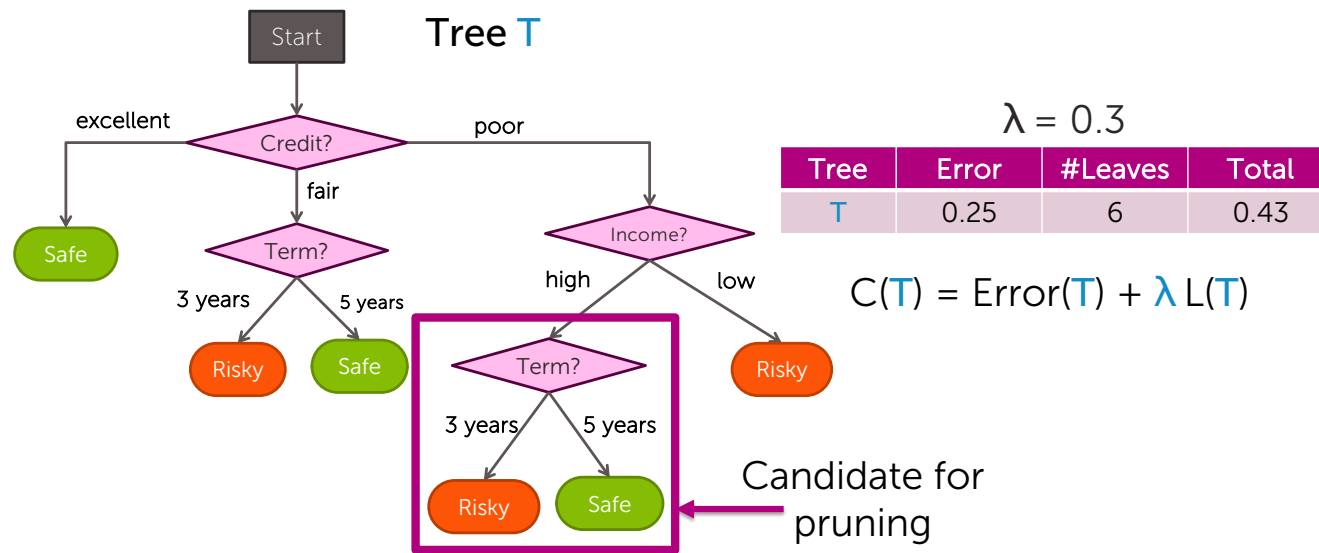
Tree pruning algorithm



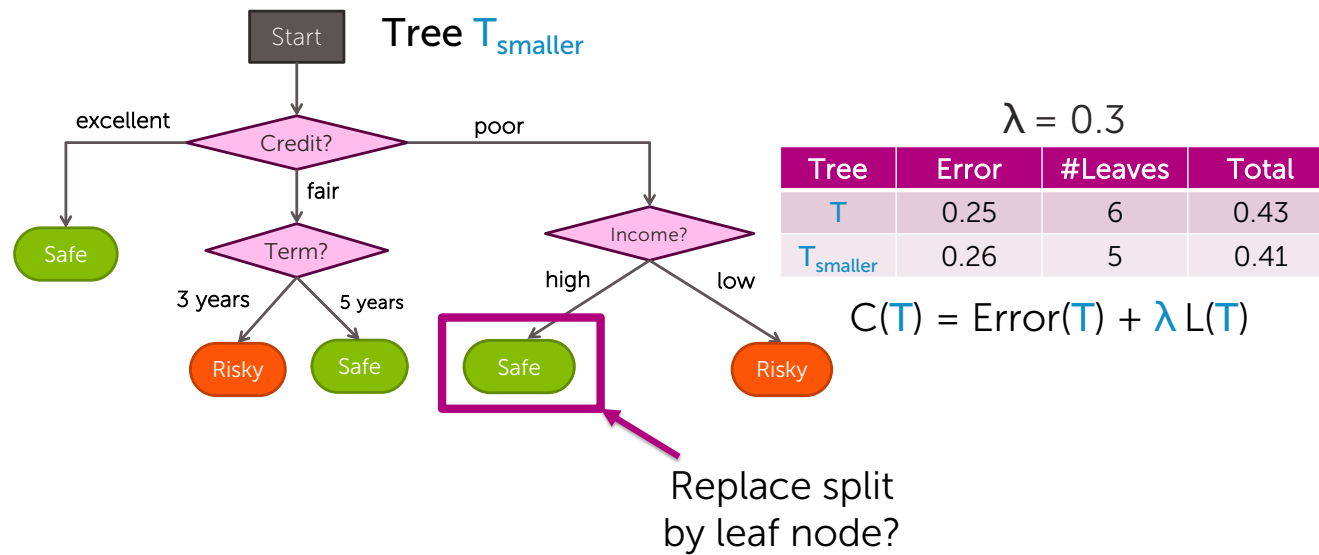
Step 1: Consider a split



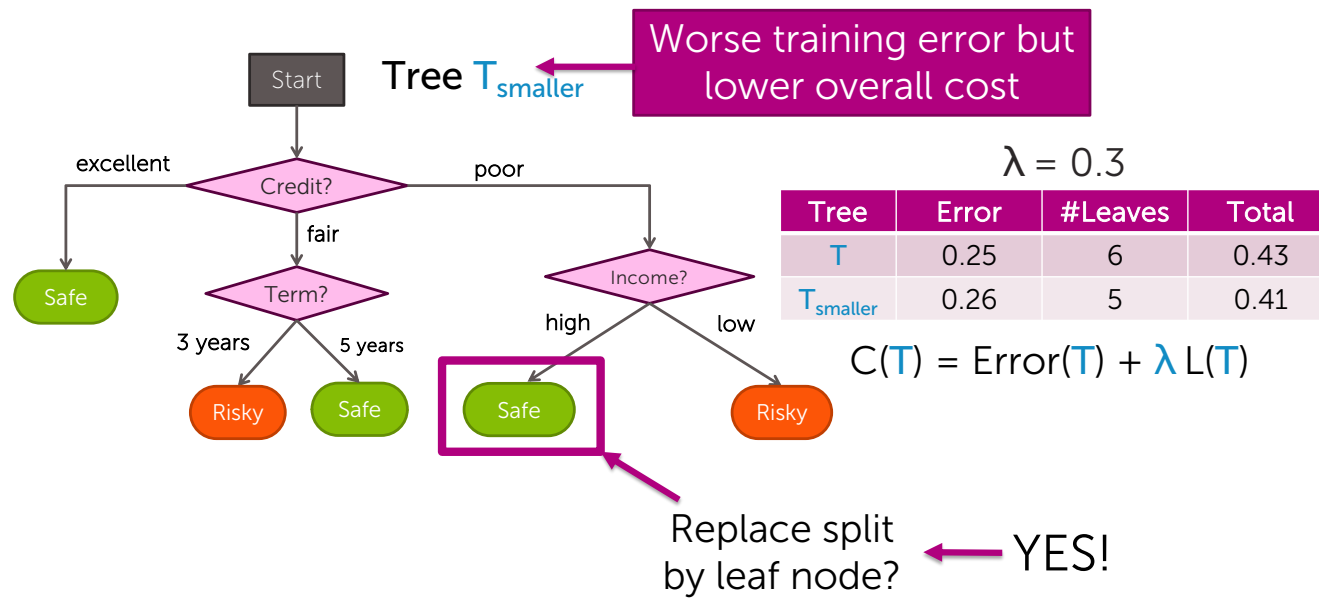
Step 2: Compute total cost $C(T)$ of split



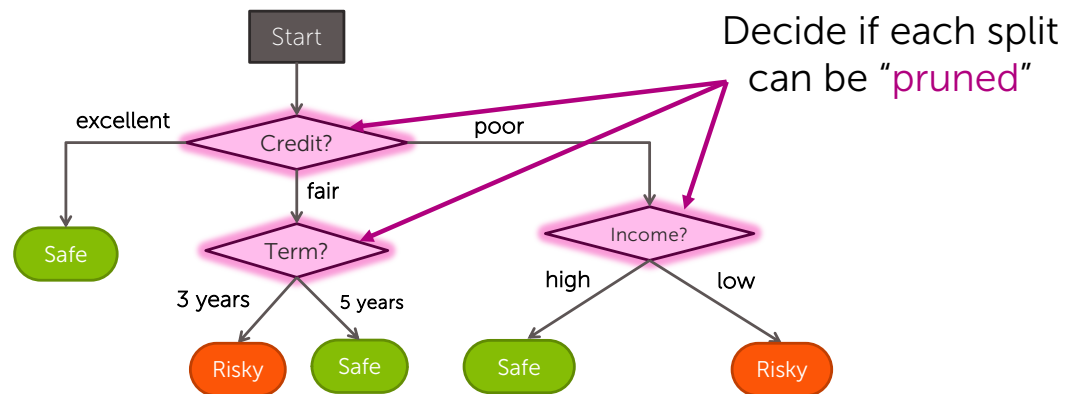
Step 2: “Undo” the splits on T_{smaller}



Prune if total cost is lower: $C(T_{\text{smaller}}) \leq C(T)$



Step 5: Repeat Steps 1-4 for every split



Summary of overfitting in decision trees

What you can do now...

- Identify when overfitting in decision trees
- Prevent overfitting with early stopping
 - Limit tree depth
 - Do not consider splits that do not reduce classification error
 - Do not split intermediate nodes with only few points
- Prevent overfitting by pruning complex trees
 - Use a total cost formula that balances classification error and tree complexity
 - Use total cost to merge potentially complex trees into simpler ones