CS 171: Intro to ML and DM

Christian Shelton

UC Riverside

Slide Set 12: Decision Trees II

Slides from CS 171

- From UC Riverside
  - CS 171: Introduction to Machine Learning and Data Mining
  - Professor Christian Shelton
- DO NOT REDISTRIBUTE
  - These slides contain copyrighted material (used with permission) from
    - Elements of Statistical Learning (Hastie, et al.)
    - Pattern Recognition and Machine Learning (Bishop)
    - An Introduction to Machine Learning (Kubat)
    - Machine Learning: A Probabilistic Perspective (Murphy)
  - For use only by enrolled students in the course

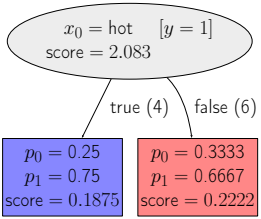
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1

$\boxed{y = 1}$

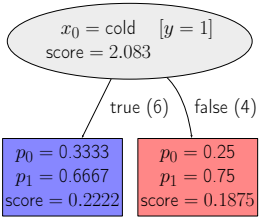
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



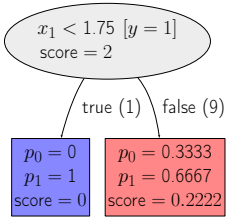
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



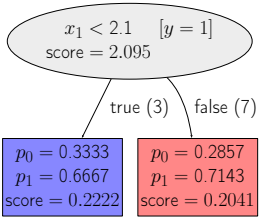
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



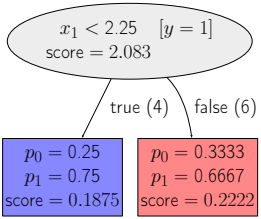
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



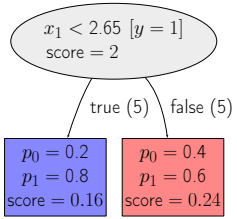
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



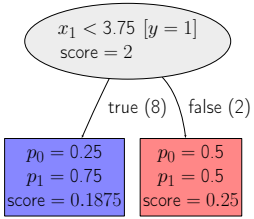
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



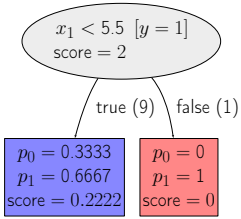
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



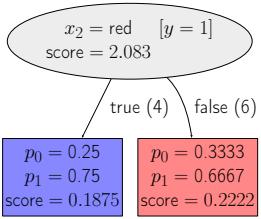
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



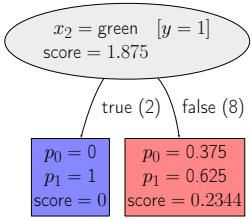
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



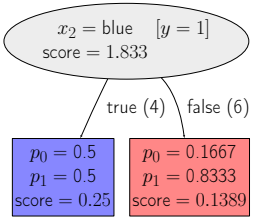
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



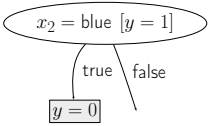
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



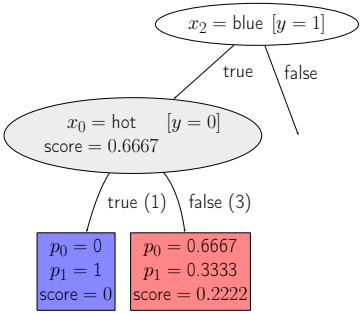
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



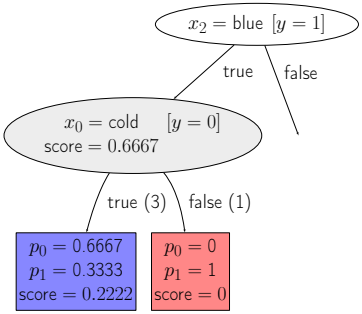
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



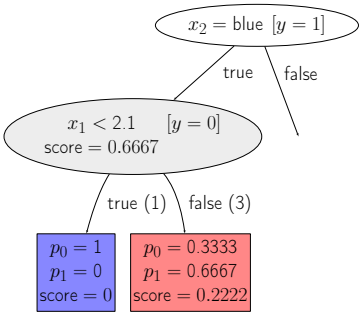
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



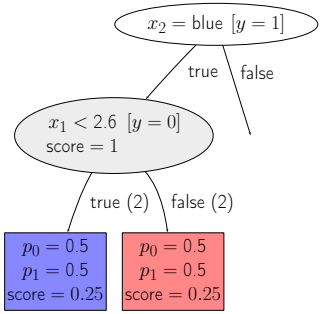
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



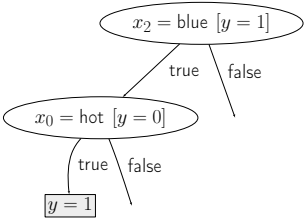
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



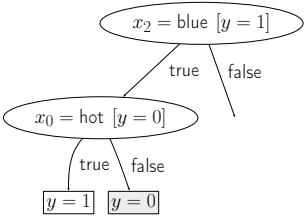
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



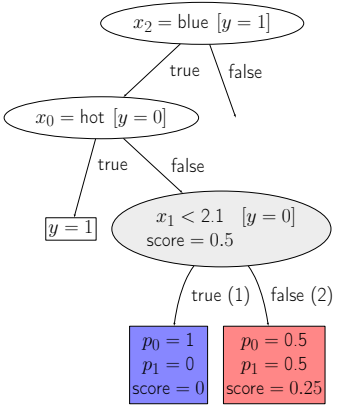
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



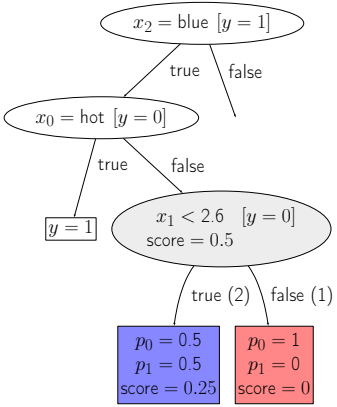
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



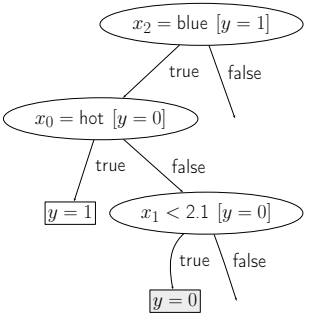
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



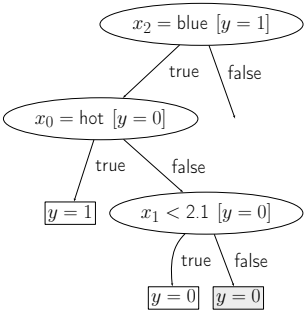
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



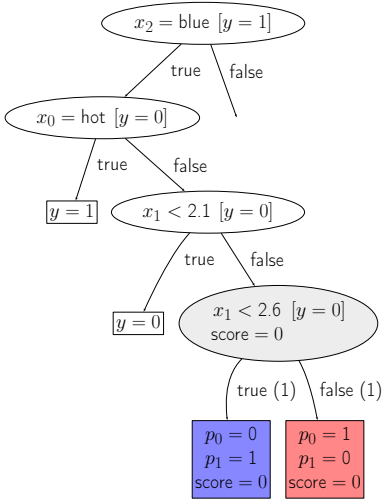
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



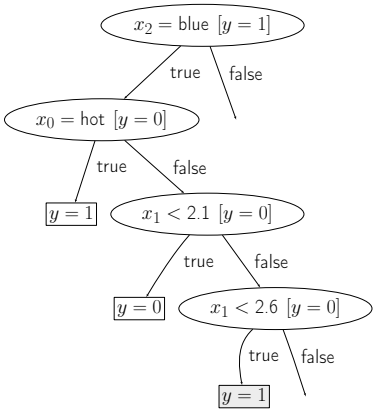
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



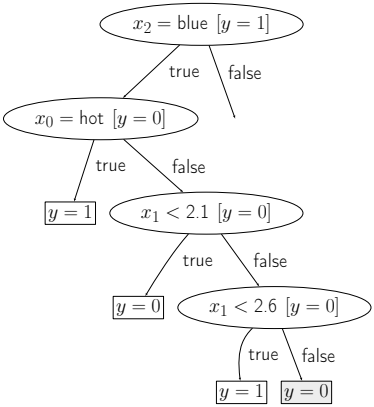
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



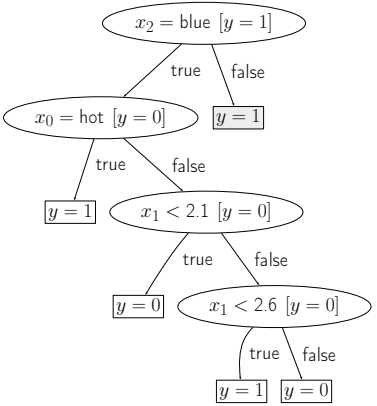
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



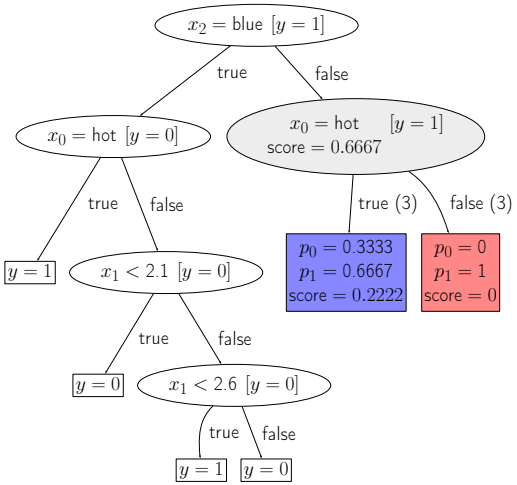
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



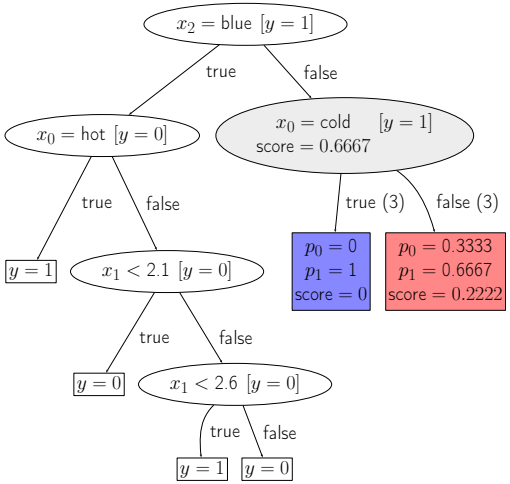
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



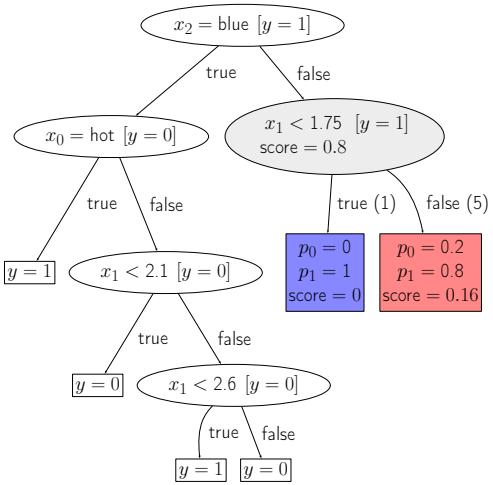
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



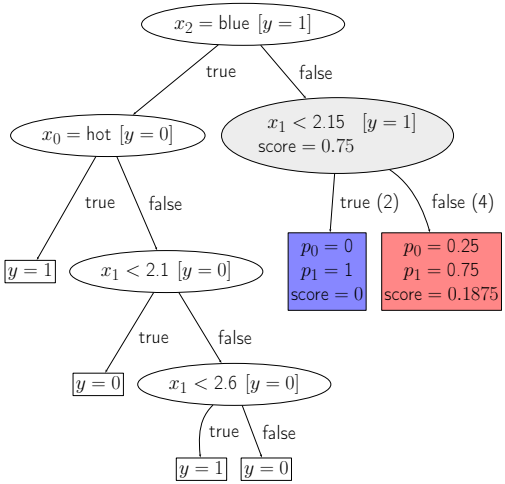
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



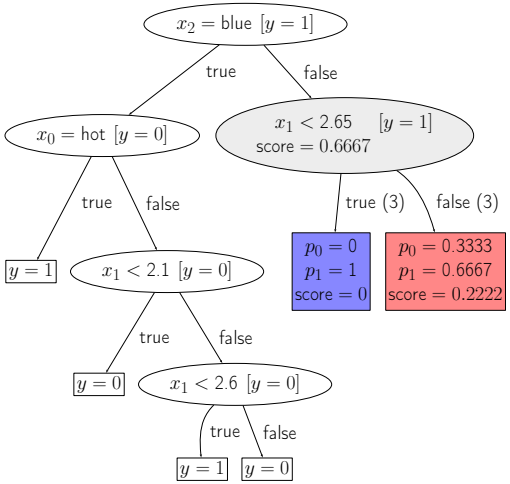
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



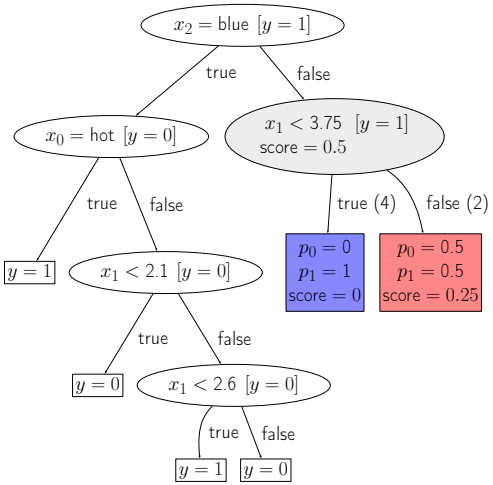
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



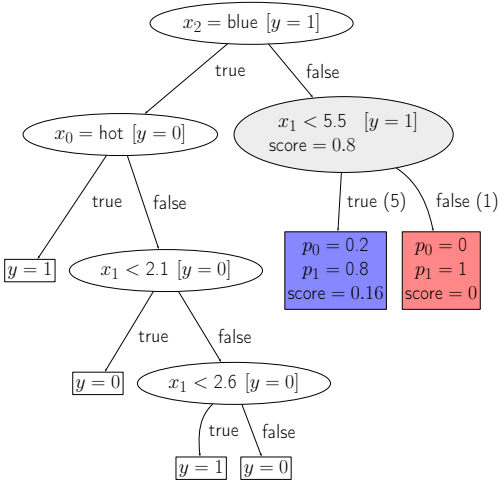
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



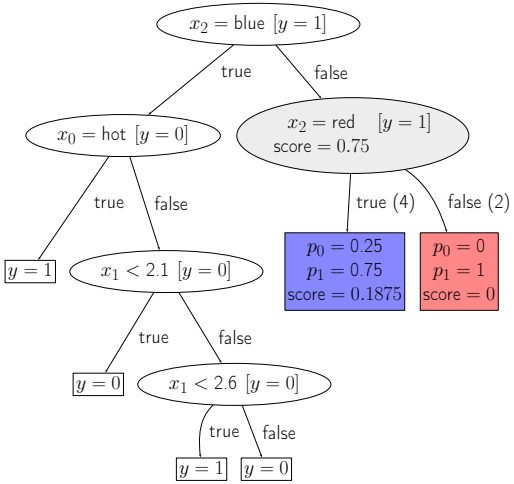
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



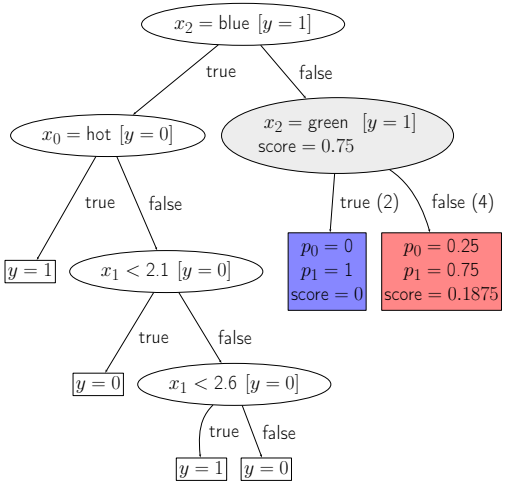
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



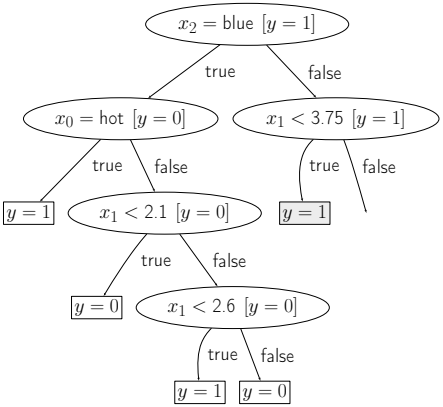
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



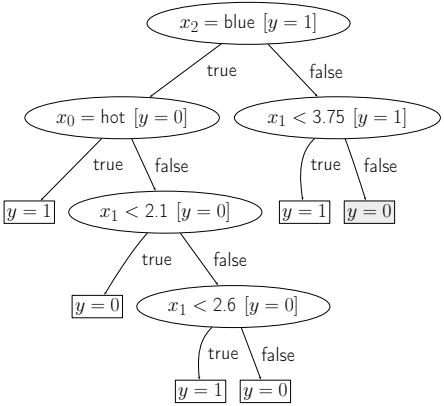
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



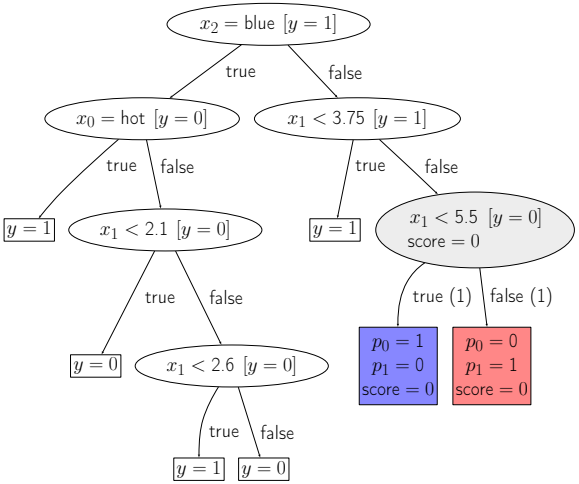
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



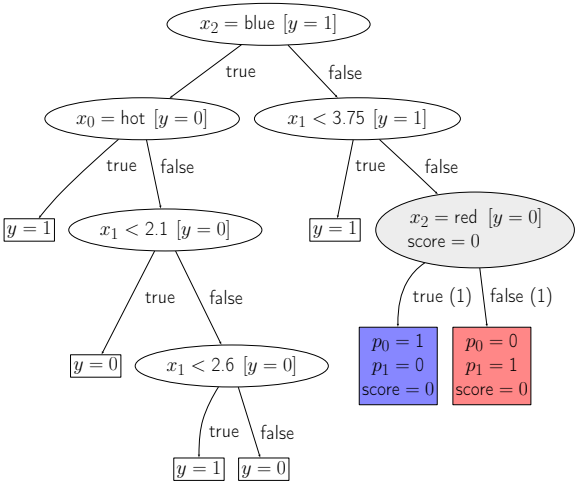
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



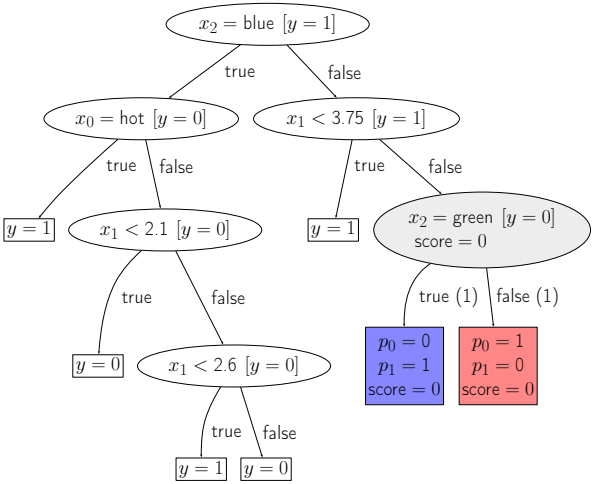
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



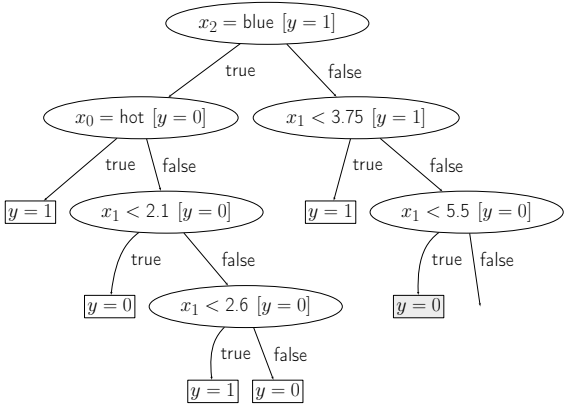
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



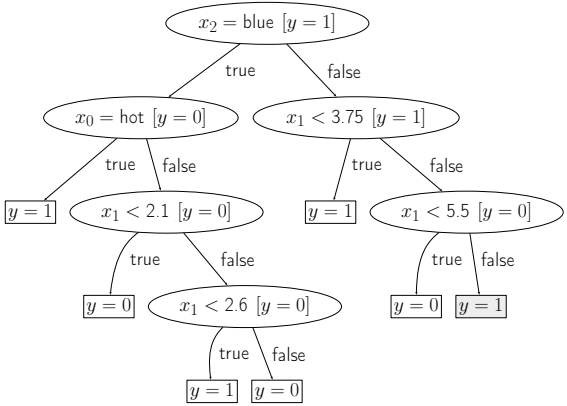
Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



Building a toy decision tree

x_0	x_1	x_2	y
cold	2	red	1
cold	2.3	green	1
hot	4.5	red	0
hot	6.5	green	1
cold	3	blue	0
hot	3	blue	1
cold	2	blue	0
cold	2.2	blue	1
hot	3	red	1
cold	1.5	red	1



Decision Tree Pruning

Consider (recursively, bottom-up) replacing each subtree with a leaf

- Cost-complexity pruning: Prune if it improves $L + \alpha\|\mathcal{T}\|$ (α chosen by cross-validation)
- Chi-squared pruning: use statistical test to check if test is correlated with label
- Reduced error pruning: Prune if doing so does not change or improves **pruning set** (like a validation set) **error**

Decision Tree Pruning

Consider (recursively, bottom-up) replacing each subtree with a leaf

- Cost-complexity pruning: Prune if it improves $L + \alpha\|\mathcal{T}\|$ (α chosen by cross-validation)
- Chi-squared pruning: use statistical test to check if test is correlated with label
- Reduced error pruning: Prune if doing so does not change or improves **pruning set** (like a validation set) **error**
  - How is pruned leaf's y values chosen? *from training data*
  - Bottom-up or iterative?

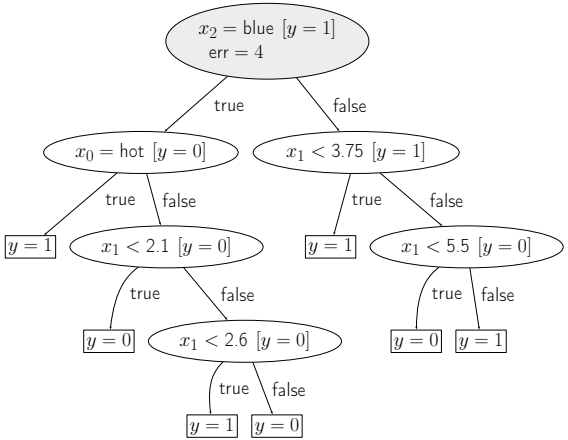
Decision Tree Pruning

Consider (recursively, bottom-up) replacing each subtree with a leaf

- Cost-complexity pruning: Prune if it improves $L + \alpha\|\mathcal{T}\|$ (α chosen by cross-validation)
- Chi-squared pruning: use statistical test to check if test is correlated with label
- Reduced error pruning: Prune if doing so does not change or improves **pruning set** (like a validation set) **error**
  - How is pruned leaf's y values chosen? *from training data*
  - Bottom-up or iterative? *bottom-up*

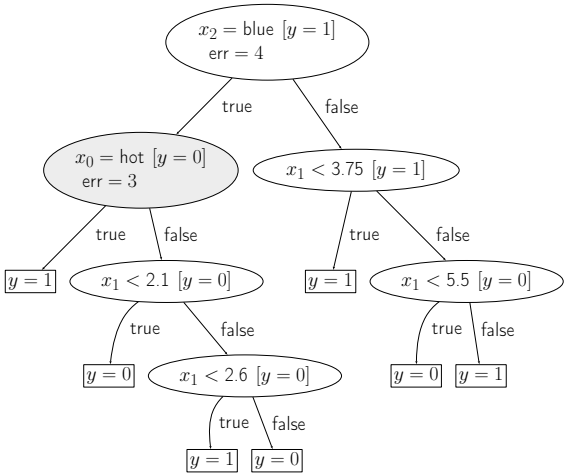
Building a toy decision tree

x_0	x_1	x_2	y
cold	3.5	blue	1
cold	1.5	blue	0
cold	2	blue	0
hot	3.5	red	0
cold	5.9	green	1
cold	2.2	blue	0
hot	3	blue	1
cold	3	blue	1



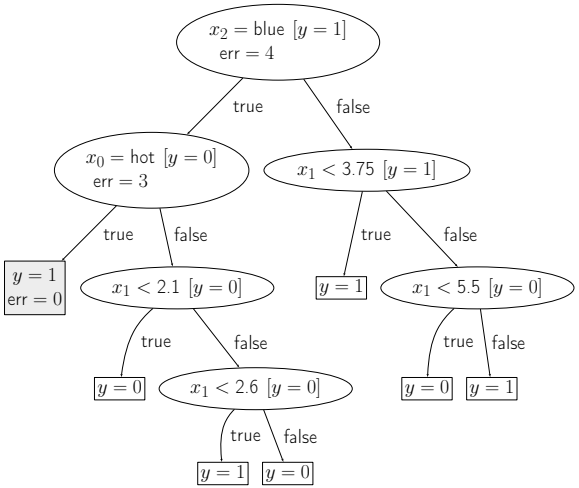
Building a toy decision tree

x_0	x_1	x_2	y
cold	3.5	blue	1
cold	1.5	blue	0
cold	2	blue	0
hot	3.5	red	0
cold	5.9	green	1
cold	2.2	blue	0
hot	3	blue	1
cold	3	blue	1



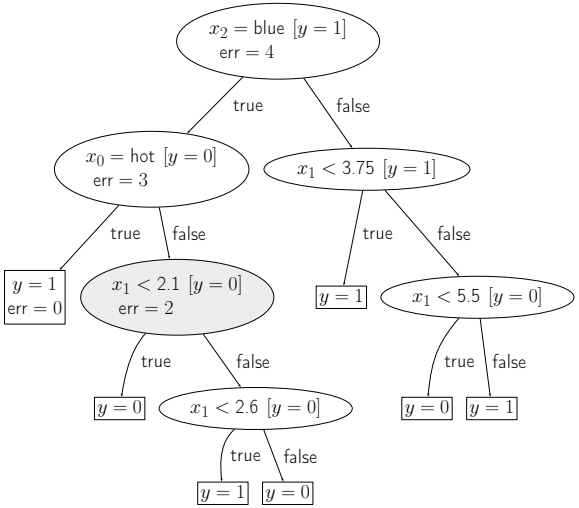
Building a toy decision tree

x_0	x_1	x_2	y
cold	3.5	blue	1
cold	1.5	blue	0
cold	2	blue	0
hot	3.5	red	0
cold	5.9	green	1
cold	2.2	blue	0
hot	3	blue	1
cold	3	blue	1



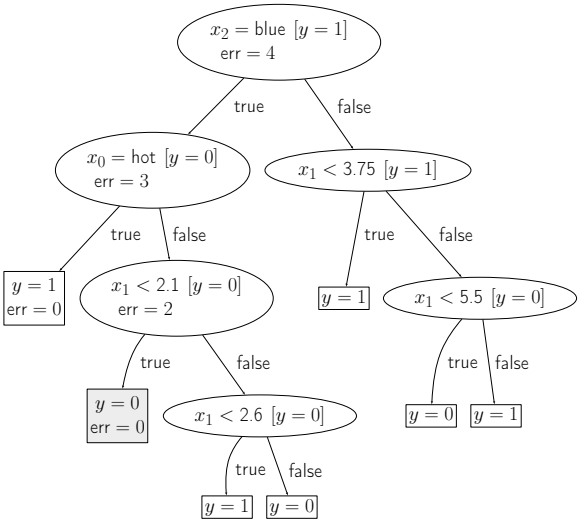
Building a toy decision tree

x_0	x_1	x_2	y
cold	3.5	blue	1
cold	1.5	blue	0
cold	2	blue	0
hot	3.5	red	0
cold	5.9	green	1
cold	2.2	blue	0
hot	3	blue	1
cold	3	blue	1



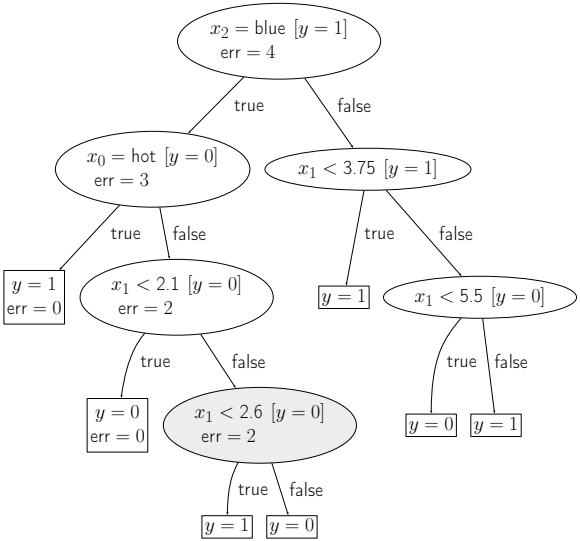
Building a toy decision tree

x_0	x_1	x_2	y
cold	3.5	blue	1
cold	1.5	blue	0
cold	2	blue	0
hot	3.5	red	0
cold	5.9	green	1
cold	2.2	blue	0
hot	3	blue	1
cold	3	blue	1



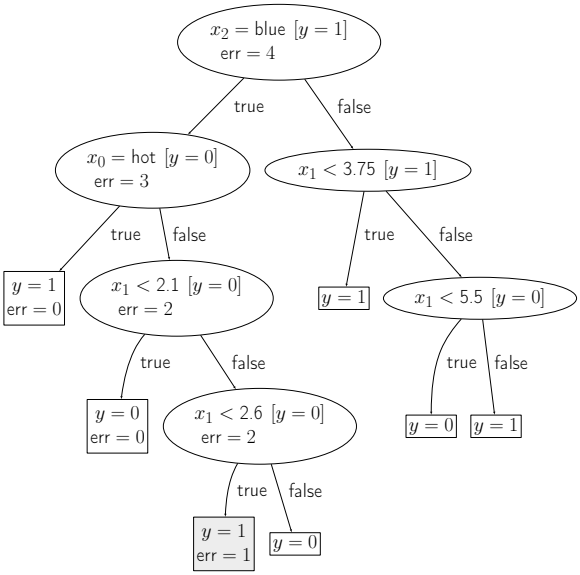
Building a toy decision tree

x_0	x_1	x_2	y
cold	3.5	blue	1
cold	1.5	blue	0
cold	2	blue	0
hot	3.5	red	0
cold	5.9	green	1
cold	2.2	blue	0
hot	3	blue	1
cold	3	blue	1



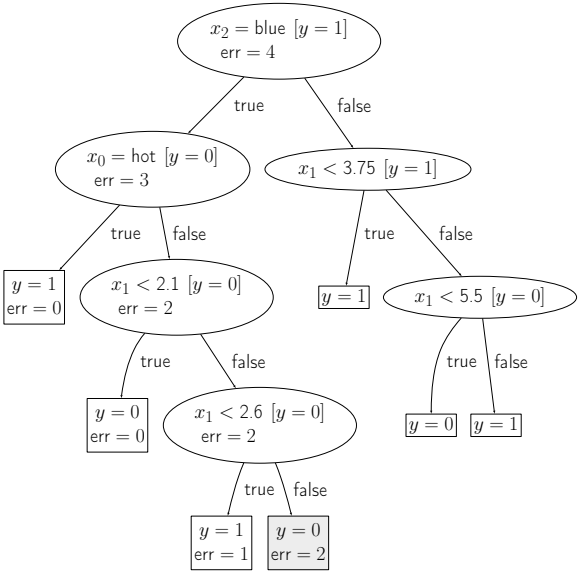
Building a toy decision tree

x_0	x_1	x_2	y
cold	3.5	blue	1
cold	1.5	blue	0
cold	2	blue	0
hot	3.5	red	0
cold	5.9	green	1
cold	2.2	blue	0
hot	3	blue	1
cold	3	blue	1



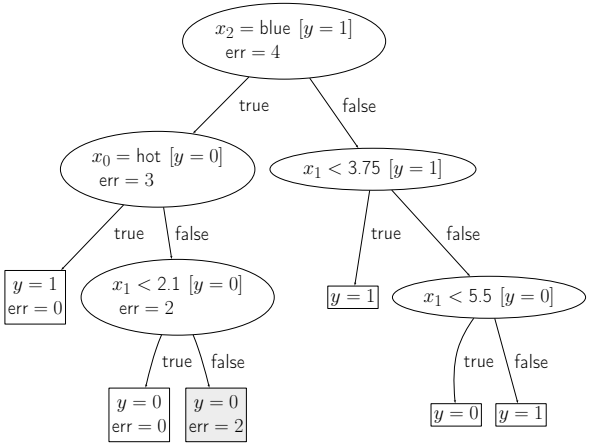
Building a toy decision tree

x_0	x_1	x_2	y
cold	3.5	blue	1
cold	1.5	blue	0
cold	2	blue	0
hot	3.5	red	0
cold	5.9	green	1
cold	2.2	blue	0
hot	3	blue	1
cold	3	blue	1



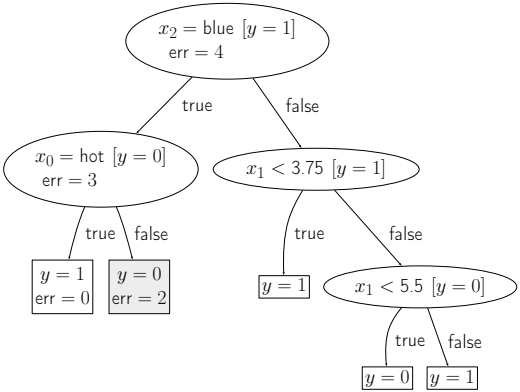
Building a toy decision tree

x_0	x_1	x_2	y
cold	3.5	blue	1
cold	1.5	blue	0
cold	2	blue	0
hot	3.5	red	0
cold	5.9	green	1
cold	2.2	blue	0
hot	3	blue	1
cold	3	blue	1



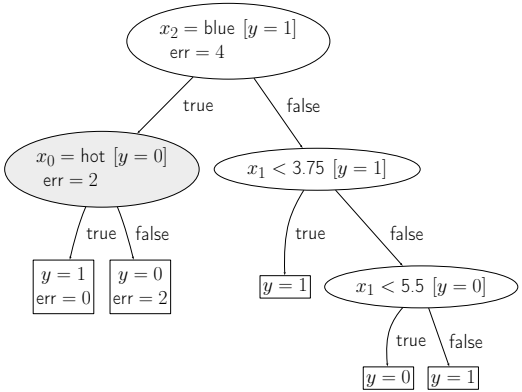
Building a toy decision tree

x_0	x_1	x_2	y
cold	3.5	blue	1
cold	1.5	blue	0
cold	2	blue	0
hot	3.5	red	0
cold	5.9	green	1
cold	2.2	blue	0
hot	3	blue	1
cold	3	blue	1



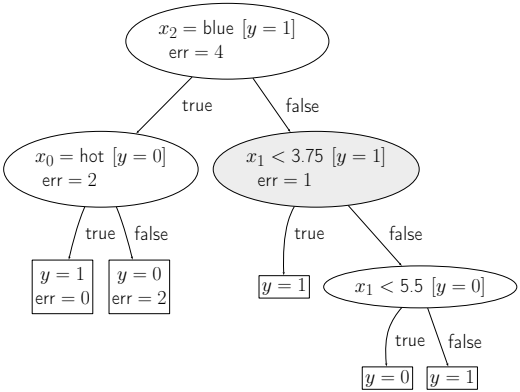
Building a toy decision tree

x_0	x_1	x_2	y
cold	3.5	blue	1
cold	1.5	blue	0
cold	2	blue	0
hot	3.5	red	0
cold	5.9	green	1
cold	2.2	blue	0
hot	3	blue	1
cold	3	blue	1



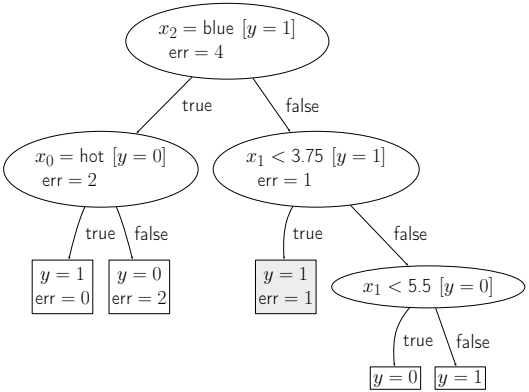
Building a toy decision tree

x_0	x_1	x_2	y
cold	3.5	blue	1
cold	1.5	blue	0
cold	2	blue	0
hot	3.5	red	0
cold	5.9	green	1
cold	2.2	blue	0
hot	3	blue	1
cold	3	blue	1



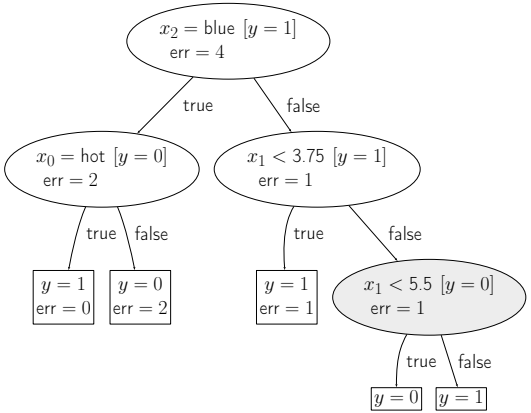
Building a toy decision tree

x_0	x_1	x_2	y
cold	3.5	blue	1
cold	1.5	blue	0
cold	2	blue	0
hot	3.5	red	0
cold	5.9	green	1
cold	2.2	blue	0
hot	3	blue	1
cold	3	blue	1



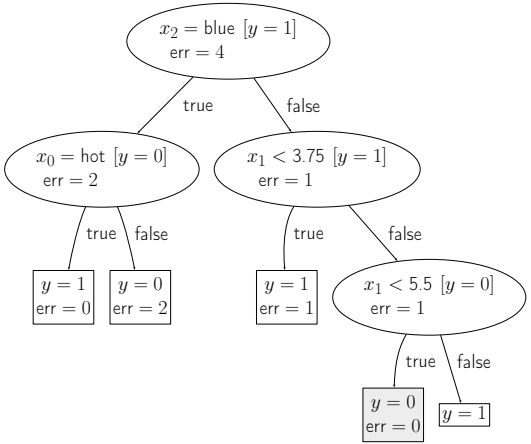
Building a toy decision tree

x_0	x_1	x_2	y
cold	3.5	blue	1
cold	1.5	blue	0
cold	2	blue	0
hot	3.5	red	0
cold	5.9	green	1
cold	2.2	blue	0
hot	3	blue	1
cold	3	blue	1



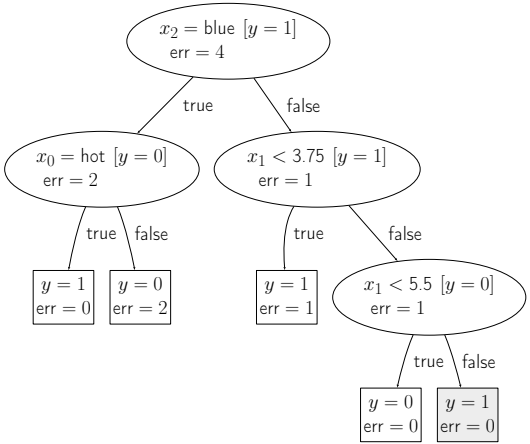
Building a toy decision tree

x_0	x_1	x_2	y
cold	3.5	blue	1
cold	1.5	blue	0
cold	2	blue	0
hot	3.5	red	0
cold	5.9	green	1
cold	2.2	blue	0
hot	3	blue	1
cold	3	blue	1



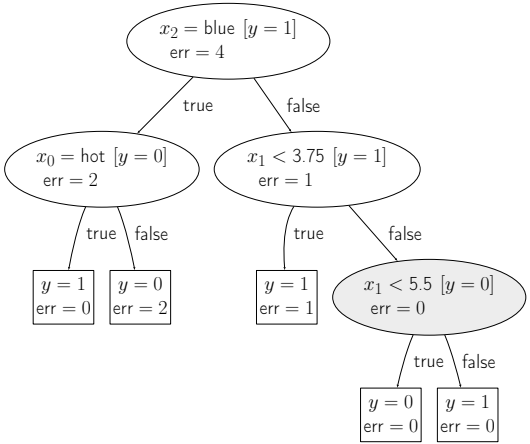
Building a toy decision tree

x_0	x_1	x_2	y
cold	3.5	blue	1
cold	1.5	blue	0
cold	2	blue	0
hot	3.5	red	0
cold	5.9	green	1
cold	2.2	blue	0
hot	3	blue	1
cold	3	blue	1



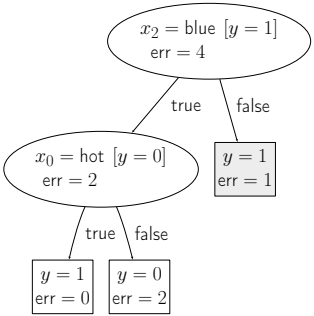
Building a toy decision tree

x_0	x_1	x_2	y
cold	3.5	blue	1
cold	1.5	blue	0
cold	2	blue	0
hot	3.5	red	0
cold	5.9	green	1
cold	2.2	blue	0
hot	3	blue	1
cold	3	blue	1



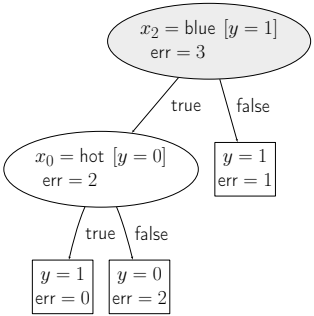
Building a toy decision tree

x_0	x_1	x_2	y
cold	3.5	blue	1
cold	1.5	blue	0
cold	2	blue	0
hot	3.5	red	0
cold	5.9	green	1
cold	2.2	blue	0
hot	3	blue	1
cold	3	blue	1



Building a toy decision tree

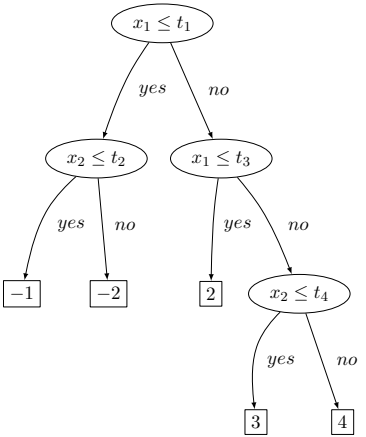
x_0	x_1	x_2	y
cold	3.5	blue	1
cold	1.5	blue	0
cold	2	blue	0
hot	3.5	red	0
cold	5.9	green	1
cold	2.2	blue	0
hot	3	blue	1
cold	3	blue	1

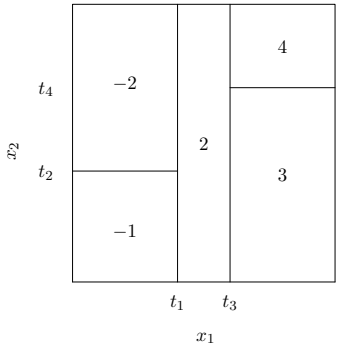


Regression Trees

Decision trees can be turned into **regression** trees:
$f(x)$ represented by a (binary) tree.

- Non-leaves are test
  - Usually uni-variate single-threshold tests
- Leaves are predicted values **(real valued scalar)**

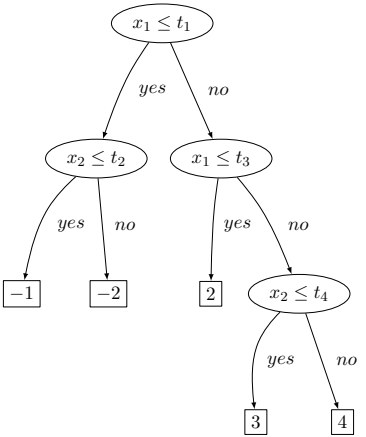


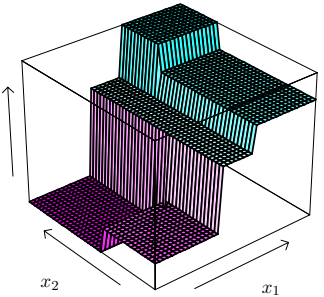


Regression Trees

Decision trees can be turned into **regression** trees:
$f(x)$ represented by a (binary) tree.

- Non-leaves are test
  - Usually uni-variate single-threshold tests
- Leaves are predicted values **(real valued scalar)**





Decision Tree Variants

- Multiclass output: adjust impurity measure for growing
- Non-binary splits: not usually done (insufficient data)

Decision Tree Variants

- Multiclass output: adjust impurity measure for growing
- Non-binary splits: not usually done (insufficient data)
- Categorical features test of the form $x_i \in \{v_1, v_2, \dots\}$:
  Instead of asking $x_i = v$, ask if one of a set of values.
  Many different splits, but can build up set incrementally.

Decision Tree Variants

- Multiclass output: adjust impurity measure for growing
- Non-binary splits: not usually done (insufficient data)
- Categorical features test of the form $x_i \in \{v_1, v_2, \dots\}$:
  Instead of asking $x_i = v$, ask if one of a set of values.
  Many different splits, but can build up set incrementally.
- Missing values:
  - Use ternary splits (extra "missing" branch)
  - Use surrogate splits (a second split that best mimics the primary split)

Decision Trees

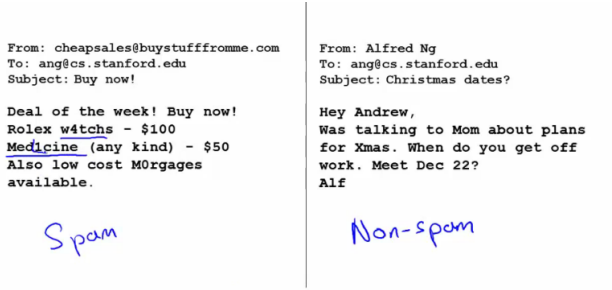
Benefits:

- Interpretable (?)
- Missing values
- Categorical features

Problems:

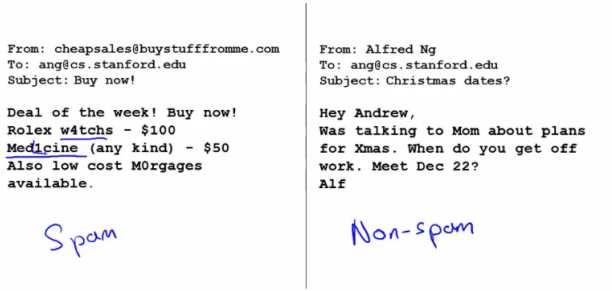
- High variance
- Not smooth
- Cannot produce linear model

Spam Example



Top image courtesy of Andrew Ng
Bottom example from George Forman of HP Labs

Spam Example



Convert e-mail into features:

- (1) CAPMAX: length of longest uninterrupted sequence of capitals
- (1) CAPAVE: average length of capitals sequences
- (1) CAPTOT: total number of capitals
- (48) % of words that match a particular word, ex: business, address, internet, free, george
- (6) % of characters that match ; ([$ #

Top image courtesy of Andrew Ng
Bottom example from George Forman of HP Labs

Spam Example