CAP 5610: Machine Learning

Lecture 11:
Decision Trees and Random Forests

Instructor: Dr. Gita Sukthankar
Email: gitars@eecs.ucf.edu

Reference Materials From: GuoJun Qi, Predrag Radenković

Reading

Chapter 12: Marsland

Chapter 16.2 Murphy

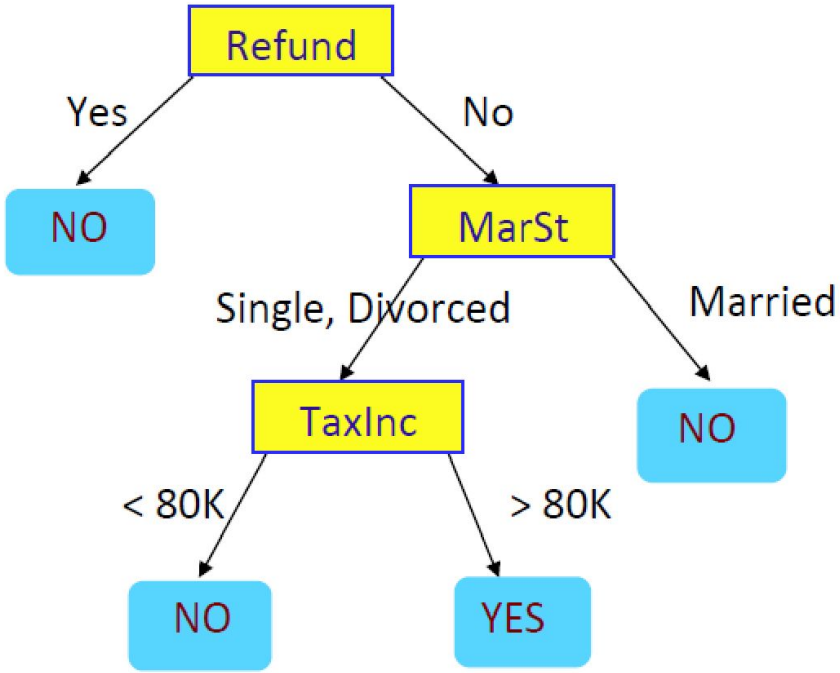
Chapter 14.4 Bishop

Decision Trees

- Decision trees are one of the most common benchmarks researchers use when evaluating classification approaches along with MLP, SVM, kNN, Naive Bayes.
- Explainable: easy to understand why the algorithm is making a decision
- Easy to combine (random forest classifiers)
- O(logN) classification time where N is the number of datapoints
- Limited to creating axis-aligned decision boundaries---no feature discovery process

Decision Trees

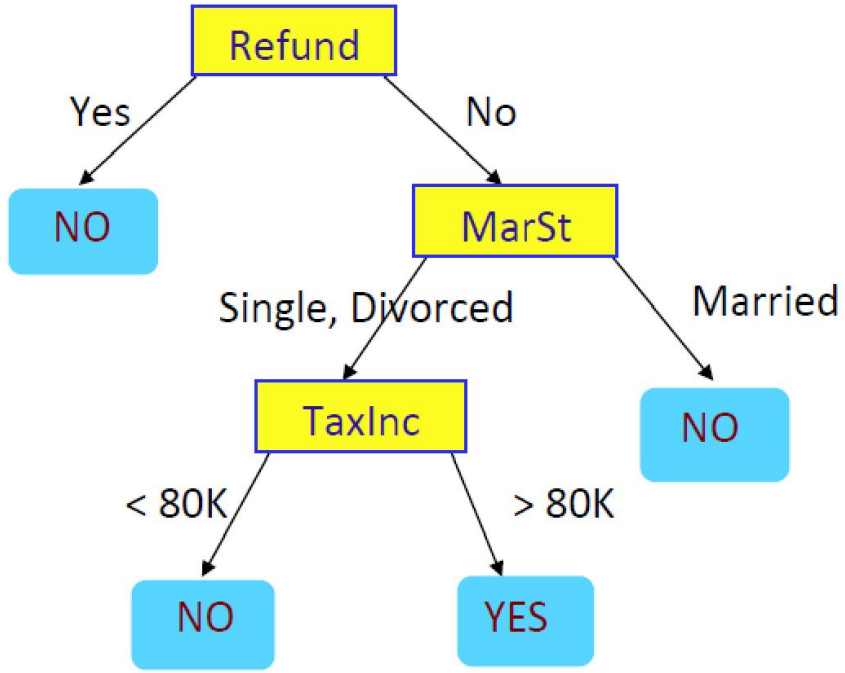
- Are a machine learning structure which is composed of a sequence of decisions to predict on an input vector of variables $X=(X_1,X_2, \ldots, X_n)$
  - An example of decision tree for tax fraud detection
  - X=(Refund, Marriage Status, Taxable income)
  - The output predicts whether an individual may cheat on tax.



By A. Singh

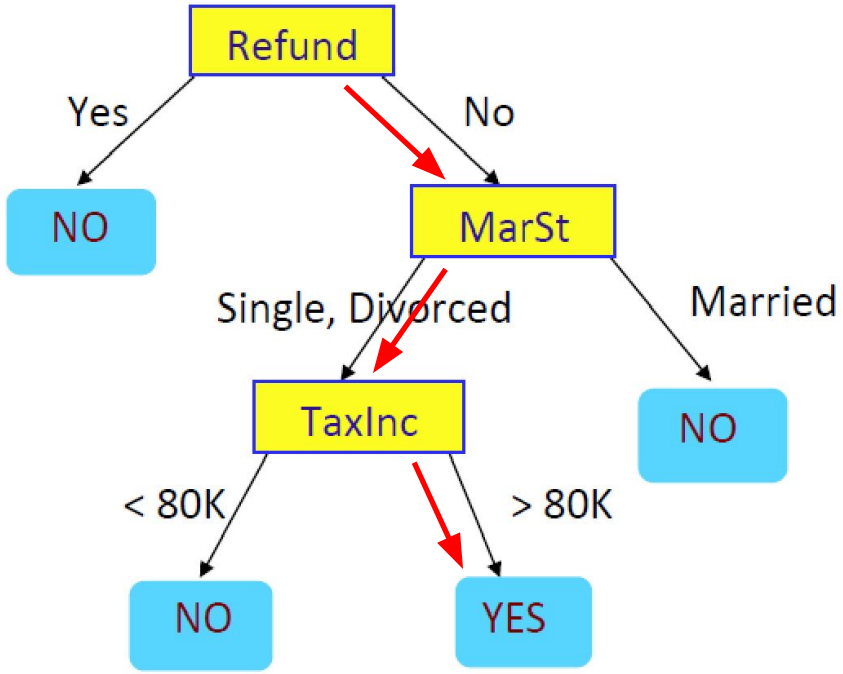
Structure

- Each non-leaf internal node corresponds to a decision made based on one of attributes X_i
- Each internal node generates several branches (usually two) depending on the decision made.
  - For example, depending on whether a tax refund is requested, we will choose to go down one of two branches.
- A leaf node will make a prediction on the input example X
  - E.g., if the individual is cheating.



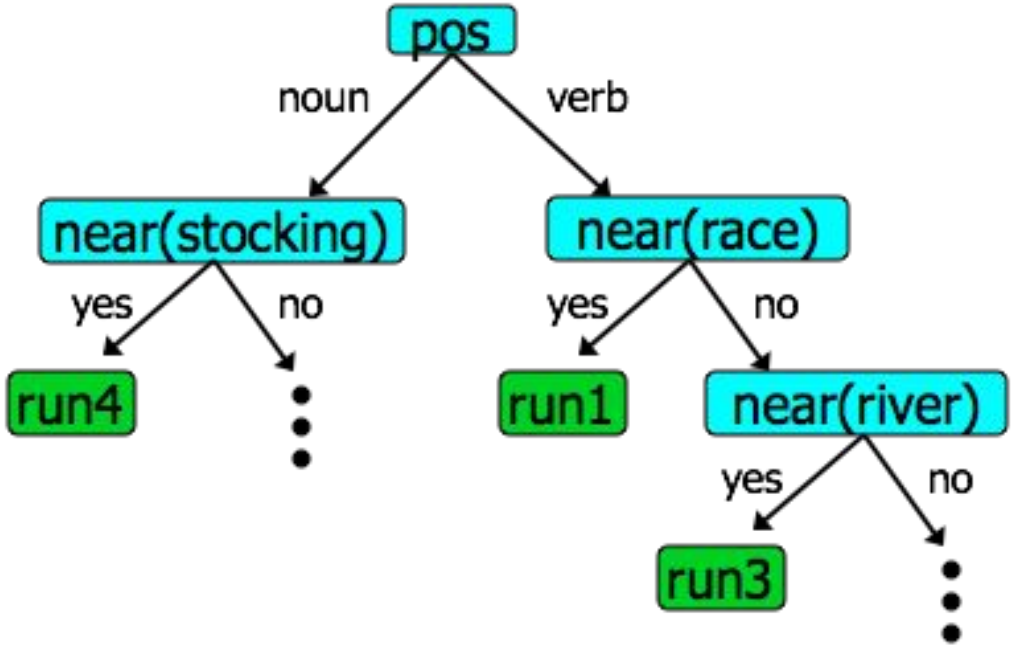
Making a prediction with a decision tree

- A test example X=(Refund: No, Marital Status: Single, Taxable Income: 85K)



Word Sense Disambiguation

- Example decision tree:



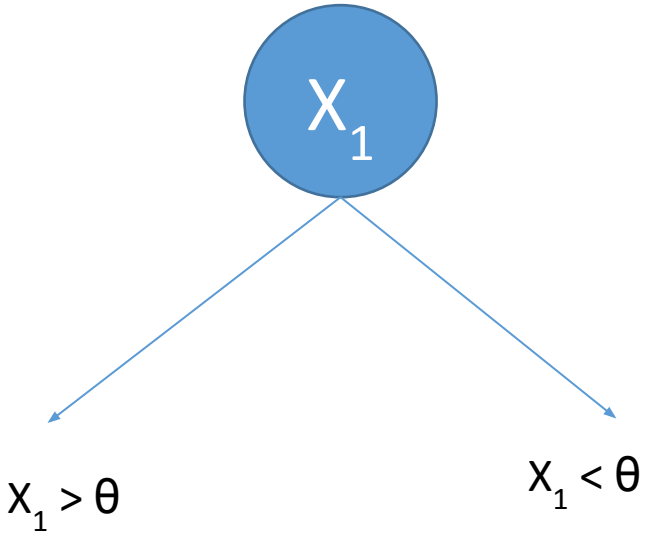
(Note: Decision trees for WSD tend to be quite large)

WSD: Training Data

Features				Word Sense
pos	near(race)	near(river)	near(stockings)	
noun	no	no	no	run4
verb	no	no	no	run1
verb	no	yes	no	run3
noun	yes	yes	yes	run4
verb	no	no	yes	run1
verb	yes	yes	no	run2
verb	no	yes	yes	run3

Types of feature attributes

- Discrete attributes
  - Each branch often contains a subset of possible values for a chosen attribute at a node.
- Continuous attributes
  - Each branch consists of a region decided by a threshold on a chosen dimension of attribute at a node
- Mixture attributes



Learning Decision Trees (Quinlan's ID3)

Top-down induction

Main loop:

1. Choose an attribute for next **node**
2. Branch the **node** based on the chosen attribute
3. Sort training examples to the descendants of the **node** for each branch
4. Stop if all the training examples have been perfectly classified, or the training error has fallen below a certain value

Later algorithms such as C4.5, CART, and C5.0 modify and improve this basic framework.

Prediction on the leaf node

- For each leaf node, usually a model is created to predict on the samples falling into this node.
- Classification problem
  - Minimizing the classification error on a training set
  - A single label is assigned to each leaf node based on majority class among the training examples falling into this node
- Regression problem
  - Minimizing the prediction error on the training set
  - A single value is assigned to all the examples assigned to the node
  - This value is the average over all the training examples assigned to this node.

Learning problem

- Deciding the tree structure
  - At each internal node (non-leaf node), which attribute is chosen?
  - Once an attribute is chosen, how can a branch be created to sort the examples?
  - When can we stop creating new branches, i.e., arriving at a leaf node?

Branching on a continuous attribute

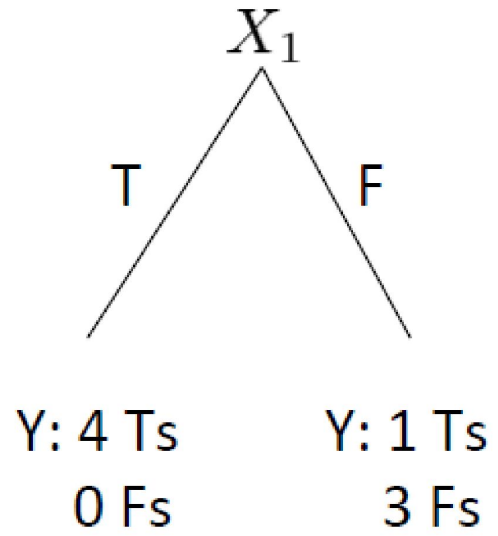
- Once have chosen a continuous attribute X_i on a node, we can fit class-conditional distributions over all the training examples sorted to this node
  - Model two Gaussian distributions $P(X_i|Y=+1)$ and $P(X_i|Y=-1)$ for a binary prediction
  - Based on Bayes classifier, a threshold can be chosen by the likelihood ratio

$$\frac{P(X_i|Y=1)P(Y=1)}{P(X_i|Y=-1)P(Y=-1)} <> \tau \leftrightarrow X_i <> \theta$$

Branching on a discrete attribute

- On a node with discrete attribute, the node can be branched with each value of this discrete attribute as an individual branch.

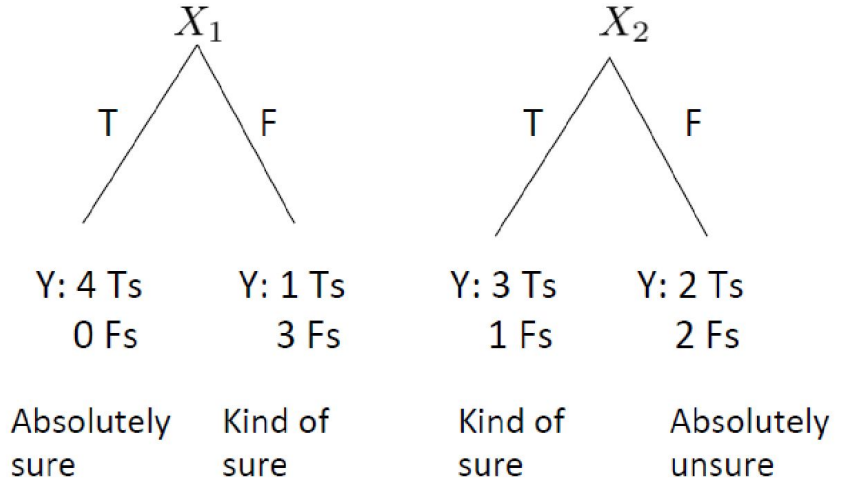
X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F
F	T	F
F	F	F



How to choose an attribute?

- An attribute will be selected if a good branch can be created.
- We are more certain about the classification after the examples are split by a good branch.

X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F
F	T	F
F	F	F



Mutual Information

- A good choice of attribute should reduce the uncertainty about the classification as much as possible.

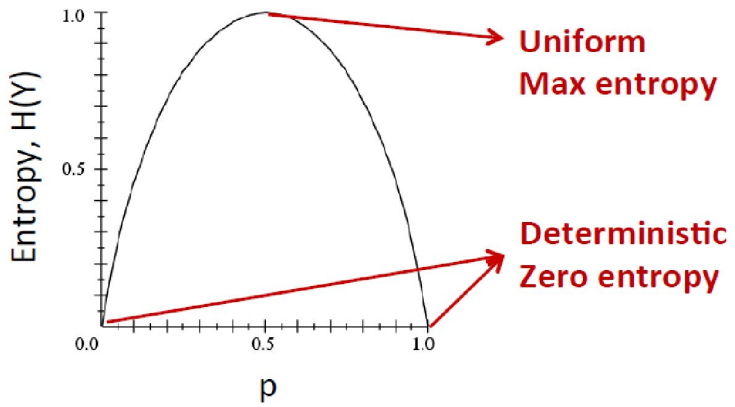
- Uncertainty can be measured by entropy according to information theory.

Choosing an attribute

- Entropy of Y before split the examples at an internal node

$$H(Y) = -\sum_{y} P(Y = y)\log \mathrm{P}(Y = y)$$

For example, when Y is binary label, and p=P(Y=1), Y is most uncertain when p=0.5, and it is least uncertain when p=0/1.



Conditional Entropy

- $H(Y|X_i)$ measures the uncertain after splitting the examples based on X_i

$$H(Y|X_i) = E_{X_i}[H(Y|X_i)]$$

$$= -\sum_x P(X_i = x) \sum_y P(Y = y|X_i = x) \log P(Y = y|X_i = x)$$

- Mathematically, we will choose the attribute that yields the maximum information gain:

$$\text{argmax}_i \; I(Y, X_i) = \text{argmax}_i [H(Y) - H(Y|X_i)] = \text{argmin}_i \; H(Y|X_i)$$

where $H(Y)$, $H(Y|X_i)$ – entropy of Y and conditional entropy of Y on X_i

Example

- Compute $H(Y|X_1)$
  - $P(X_1 = T) = P(X_1 = F) = 1/2$
  - Conditional distribution

$P(Y\|X_1)$	Y=T	Y=F
X_1=T	1	0
X_1=F	¼	¾

X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F
F	T	F
F	F	F

- $H(Y|X_1) = -\frac{1}{2}[1\log 1 + 0\log 0] - \frac{1}{2}\left[\frac{1}{4}\log\frac{1}{4} + \frac{3}{4}\log\frac{3}{4}\right]$

Example

- Compute $H(Y|X_2)$
  - $P(X_2 = T) = P(X_2 = F) = 1/2$
  - Conditional distribution

$P(Y|X_2)$	Y=T	Y=F
X_2=T	¾	¼
X_2=F	½	½

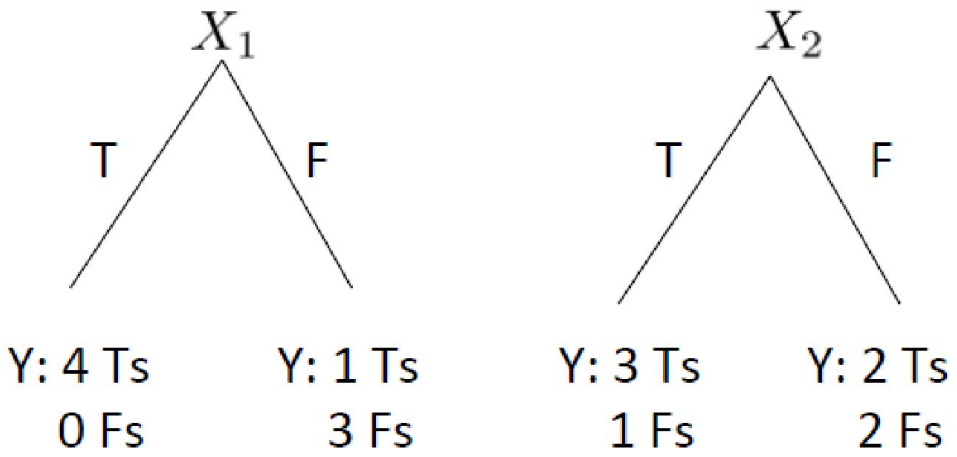
X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F
F	T	F
F	F	F

- $H(Y|X_2) = -\frac{1}{2}\left[\frac{3}{4}\log\frac{3}{4} + \frac{1}{4}\log\frac{1}{4}\right] - \frac{1}{2}\left[\frac{1}{2}\log\frac{1}{2} + \frac{1}{2}\log\frac{1}{2}\right]$

Example

- $H(Y|X_1) = -\frac{1}{2}[1\log 1 + 0\log 0] - \frac{1}{2}\left[\frac{1}{4}\log\frac{1}{4} + \frac{3}{4}\log\frac{3}{4}\right]$
- $H(Y|X_2) = -\frac{1}{2}\left[\frac{3}{4}\log\frac{3}{4} + \frac{1}{4}\log\frac{1}{4}\right] - \frac{1}{2}\left[\frac{1}{2}\log\frac{1}{2} + \frac{1}{2}\log\frac{1}{2}\right]$

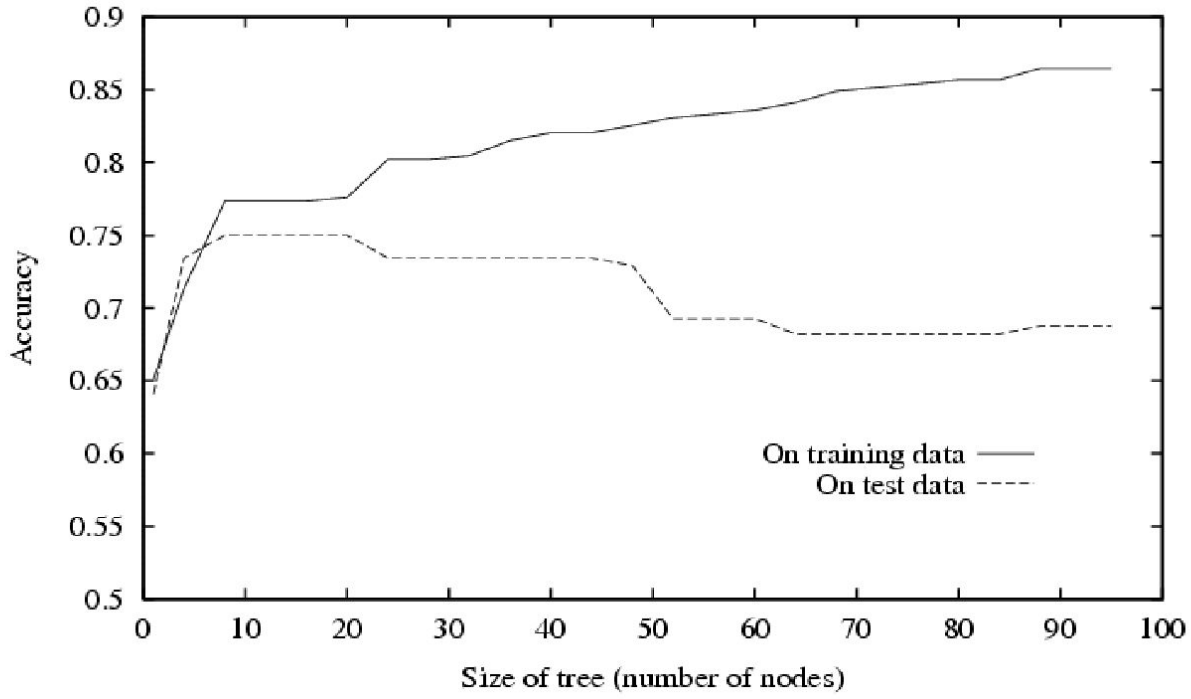
- We have $H(Y|X_1) < H(Y|X_2)$, so X_1 should be chosen.



X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F
F	T	F
F	F	F

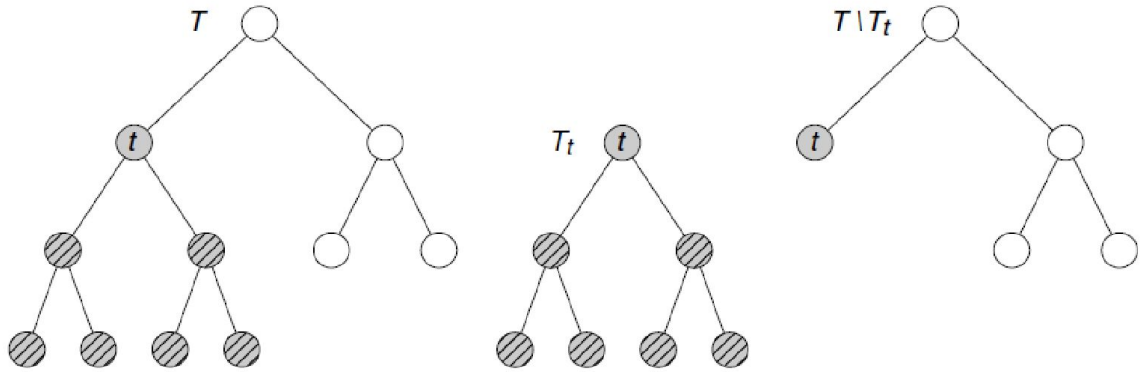
Dangers of Overfitting

- A decision tree of larger size (usually in terms of # of nodes) tends to over fit on the training set.



Pruning Strategies

- Pre-pruning: stop adding new nodes when a certain level of complexity has been reached.
  - Fixed depth
  - Fixed number of leaf nodes
- Post-pruning
  - Reduced-error pruning
    - An internal node will be converted to a leaf node
    - This leaf node will be assigned to the most common label among the training examples sorted to this node.



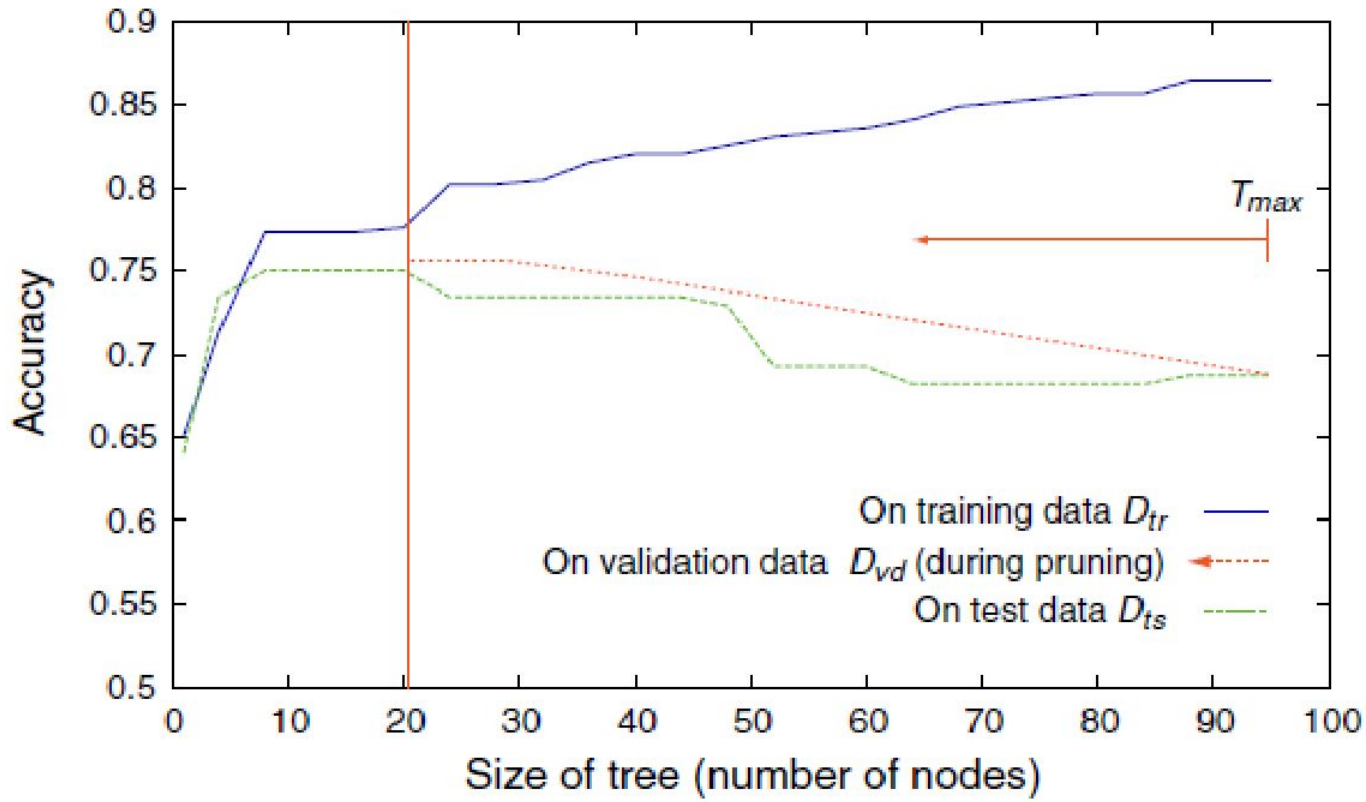
Decision Tree Pruning Methods

- Validation set – withhold a subset (~1/3) of training data to use for pruning
  - Note: randomized order of training examples

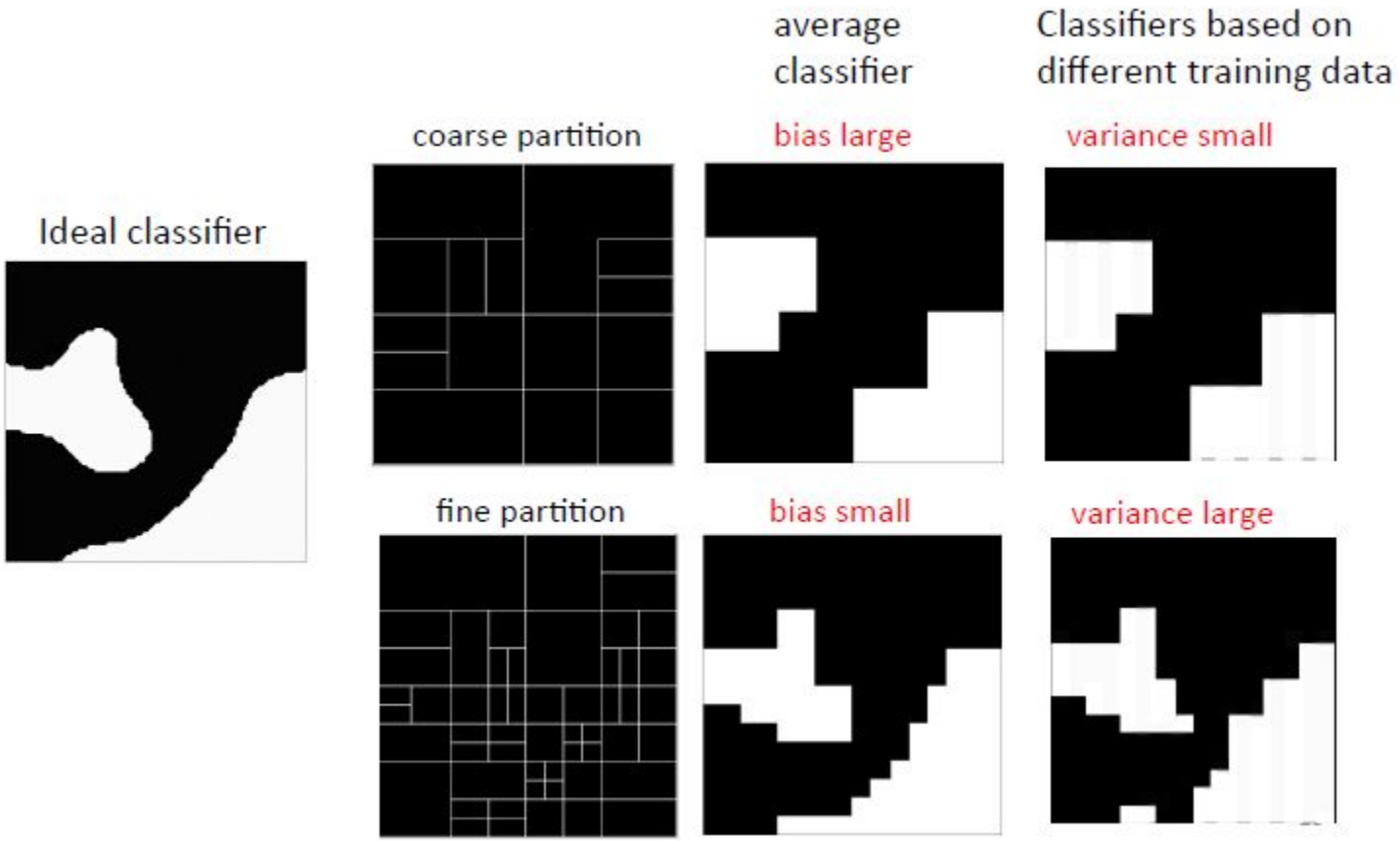
Reduced-Error Pruning

- Classify examples in validation set – some might be errors
- For each node:
  - Sum the errors over entire sub-tree of this node
  - Calculate error on same example if converted to a leaf with majority class label
- Prune node with highest reduction of error
- Repeat until error no longer reduced

Reduced-Error Pruning



Bias-Variance Tradeoff



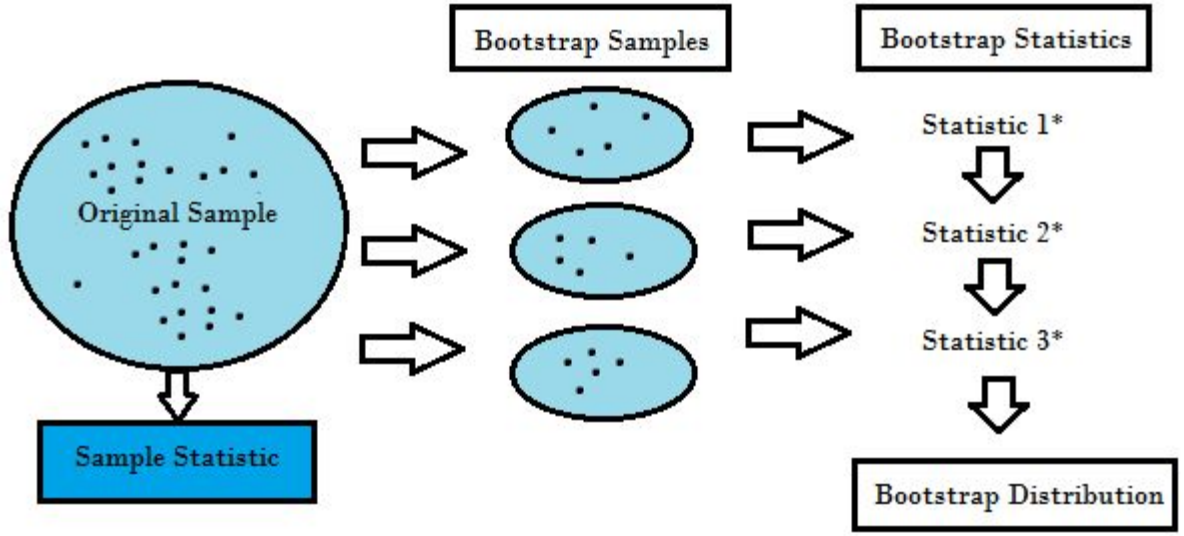
Drawn by A. Singh

A special decision tree: stump

- Stump is a decision tree with only one internal node
- Split the whole feature space by a threshold along a dimension.
- Application:
  - Due to its simplicity and fast prediction, it is used to detect faces in an image at varying locations and scales
  - But a stump is too weak as a classifier
  - But it is possible to combine them to make a better classifier (ensemble learning)

Bootstrap Sampling

- Construct classifiers using only a subset of the data
- Bootstrap sampling is a variance reduction technique used in statistics.



Ensemble learning: train classifier with different subsets of the data and then combine the classifiers.

Random Forest Algorithm

1.Let the number of training cases be N, and the number of variables in the classifier be M.

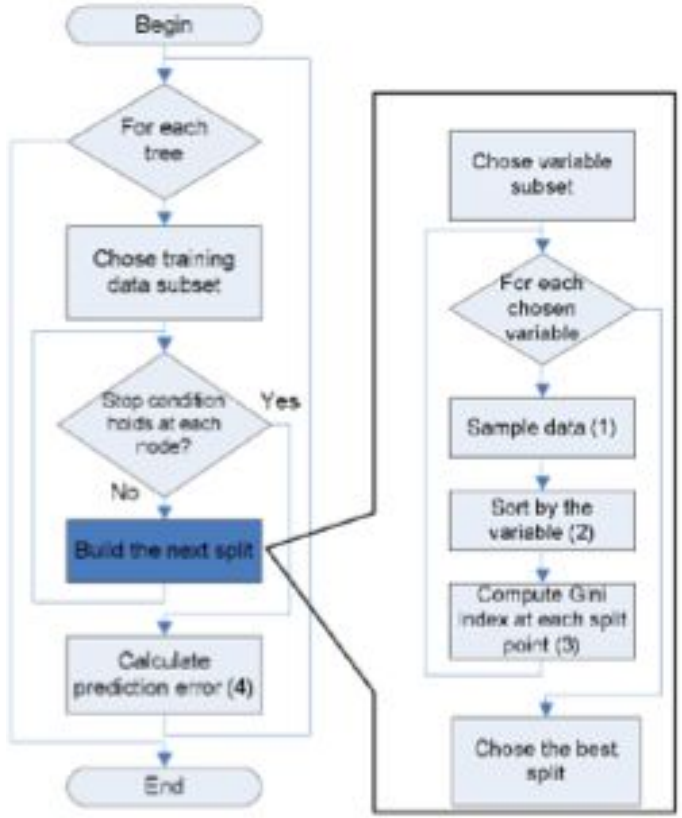
2.We are told the number m of input variables to be used to determine the decision at a node of the tree; m should be much less than M.

3.Choose a training set for this tree by choosing n times with replacement from all N available training cases (i.e. take a bootstrap sample). Use the rest of the cases to estimate the error of the tree, by predicting their classes.

4.For each node of the tree, randomly choose m variables on which to base the decision at that node. Calculate the best split based on these m variables in the training set.

5.Each tree is fully grown and not pruned (as may be done in constructing a normal tree classifier).

For prediction a new sample is pushed down the tree. It is assigned the label of the training sample in the terminal node it ends up in. This procedure is iterated over all trees in the ensemble, and the average vote of all trees is reported as random forest prediction.



Random Forest Video

https://youtu.be/LIPtRVDmj1M

Advantages

- It is one of the most accurate learning algorithms available. For many data sets, it produces a highly accurate classifier.
- It runs efficiently on large databases.
- It can handle thousands of input variables without variable deletion.
- It gives estimates of what variables are important in the classification.
- It generates an internal unbiased estimate of the generalization error as the forest building progresses.
- It has an effective method for estimating missing data and maintains accuracy when a large proportion of the data are missing.

Disadvantages

- Random forests have been observed to overfit for some datasets with noisy classification/regression tasks.
- For data including categorical variables with different number of levels, random forests are biased in favor of those attributes with more levels. Therefore, the variable importance scores from random forest are not reliable for this type of data.

Next time

Ensemble learning: general methods for combining multiple classifiers