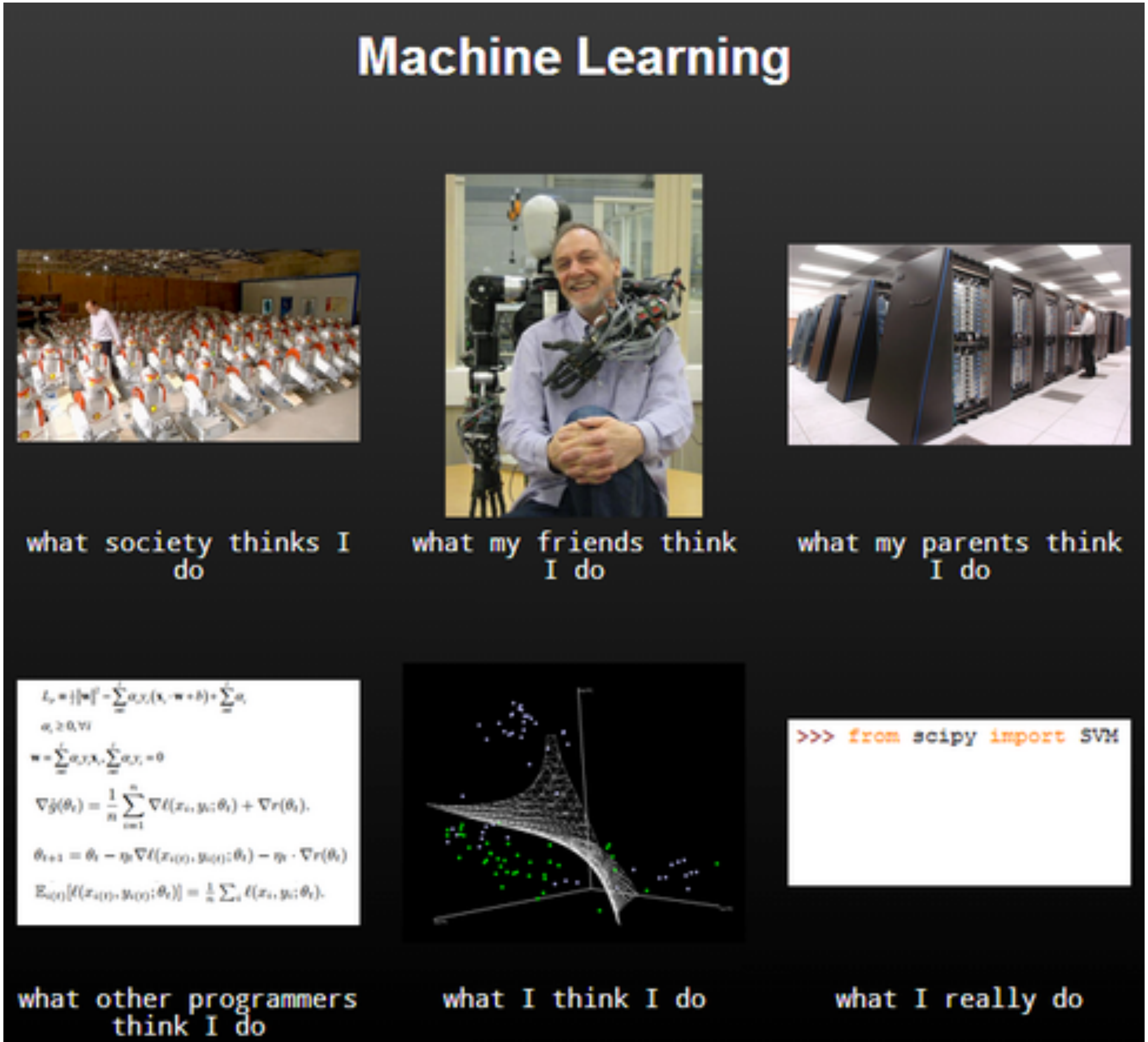
ICCS342 - Lecture 14

Decision Tree and Random Forest

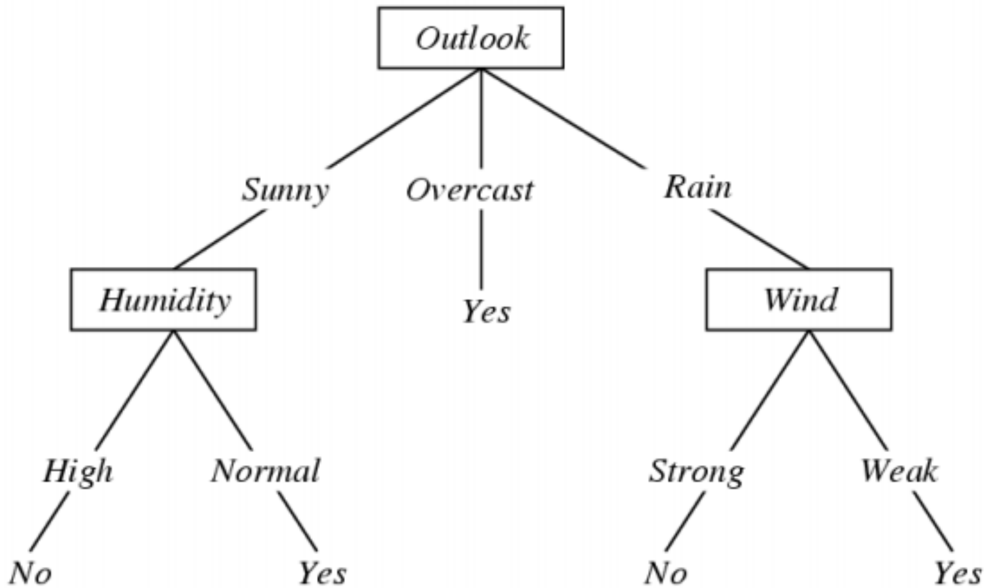
Sunsern Cheamanunkul

Slides adapted from lecture slides by Michael I. Jordan, UC — Berkeley,
and lecture slides by Tom Mitchell, CMU


Machine Learning
what society thinks I do
what my friends think I do
what my parents think I do
>>> from scipy import SVM
what other programmers think I do
what I think I do
what I really do

Decision Tree

- Example: Play tennis?



- Each node tests an attribute Xi
- Each branch from a node selects a value for Xi
- Each **leaf node** predicts y

Top-Down Induction of Decision Trees

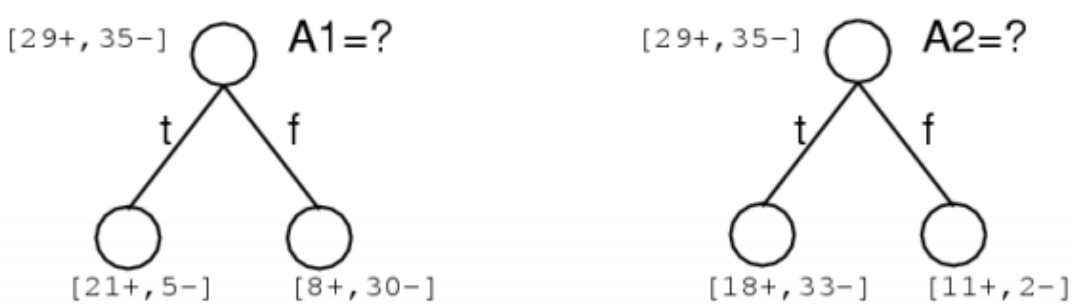
[ID3, C4.5, Quinlan]

node = Root

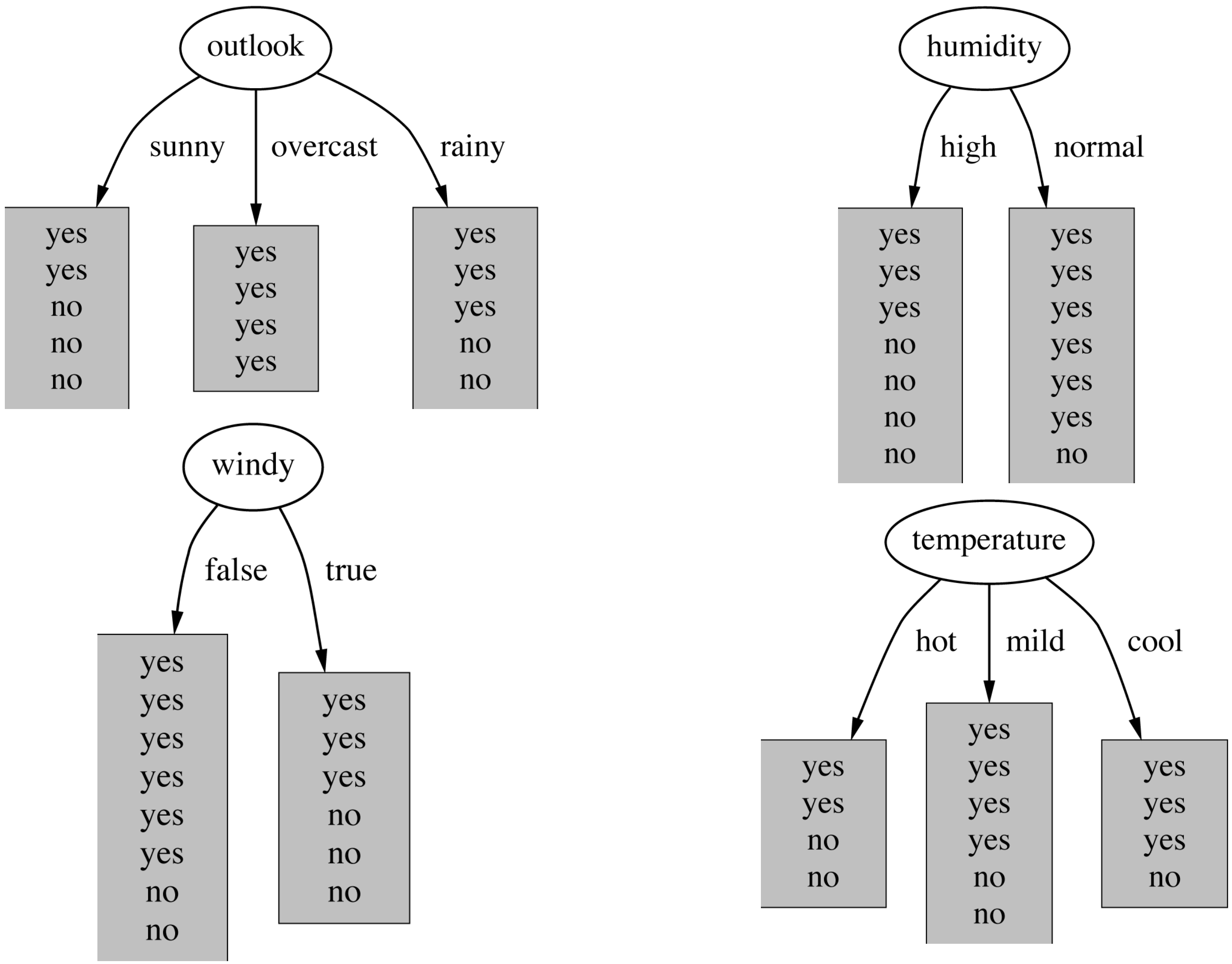
Main loop:

1. $A \leftarrow$ the "best" decision attribute for next *node*
2. Assign A as decision attribute for *node*
3. For each value of A, create new descendant of *node*
4. Sort training examples to leaf nodes
5. If training examples perfectly classified, Then STOP, Else iterate over new leaf nodes

Which attribute is best?



Which attribute to select?



Gini Impurity

- Given a set of items with J classes, suppose $i \in \{1,2,3,..,J\}$, and let p_i be the fraction of items labeled with class i in the set:

$$Gini(p) = \sum_{i=1}^{J} p_i(1 - p_i)$$

Gini Impurity

Example: $S = \{1,1,2,2,3\}$

$p_1 = 2/5$, $p_2 = 2/5$, $p_3 = 1/5$

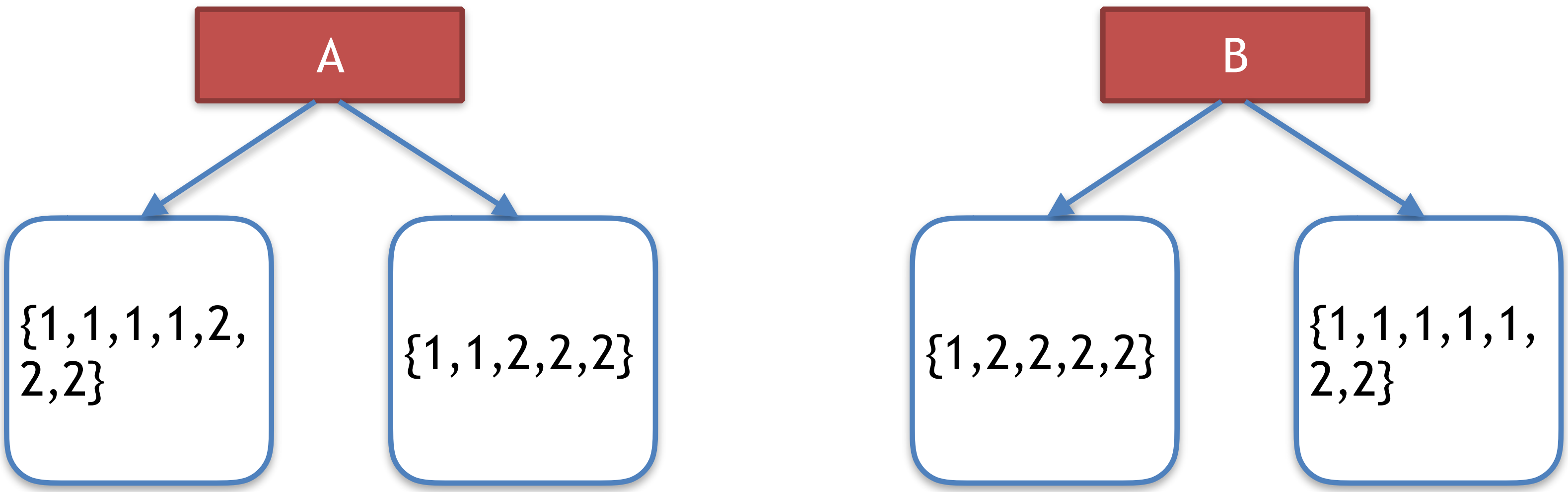
Gini Impurity of S

$= 2/5(1-2/5) + 2/5(1-2/5) + 1/5(1-1/5)$

$= 6/25 + 6/25 + 4/25$

$= 16/25 = 0.64$

Which split is better?



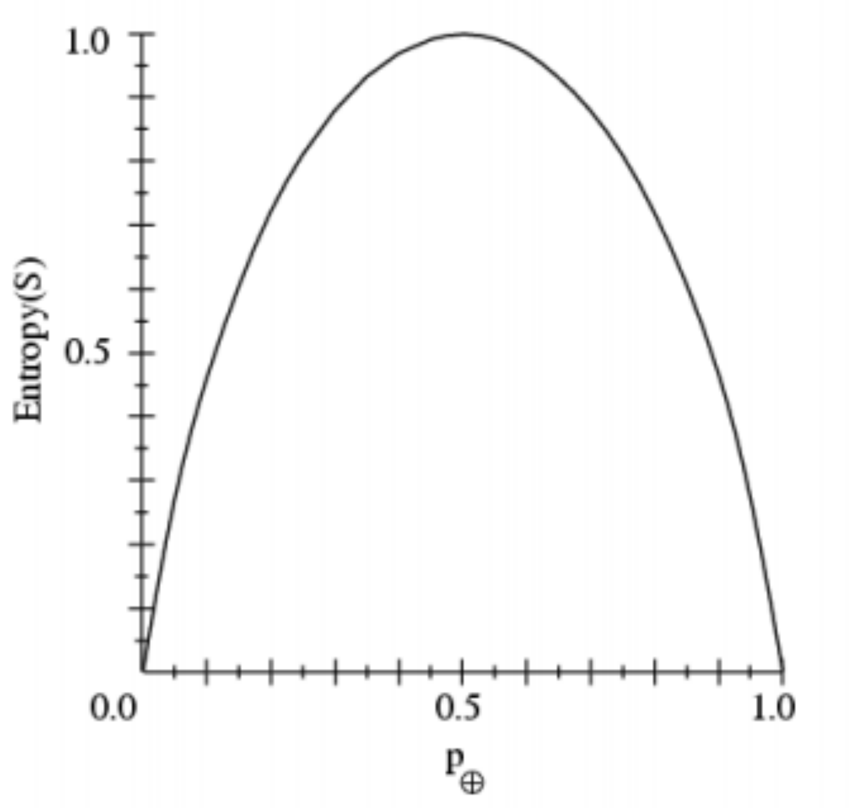
Information Entropy

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_b p(x_i)$$

- Quantifies “randomness”
- High entropy —> more random
- Low entropy —> less random
- When b=2, H(X) is the expected number of bits to encode the random variable X

Sample Entropy

- S is a sample of training examples.
- $p_\oplus$ is the proportion of positive examples in S.
- $p_\ominus$ is the proportion of negative examples in S.
- Entropy measures the impurity of S:
- $H(S) = -p_\oplus \log_2 p_\oplus - p_\ominus \log_2 p_\ominus$



Information Gain

- Based on Shannon Entropy
- IG calculates effective change in entropy after making a decision based on the value of an attribute.
- For decision trees, it's ideal to base decisions on the attribute that provides the largest change in entropy, the attribute with the highest gain.

Information Gain

- Also known as Mutual Information:

$$I(X, A) = H(X) - H(X|A)$$

where

$$H(X|A) = \sum_{a} P(A = a) H(X|A = a)$$

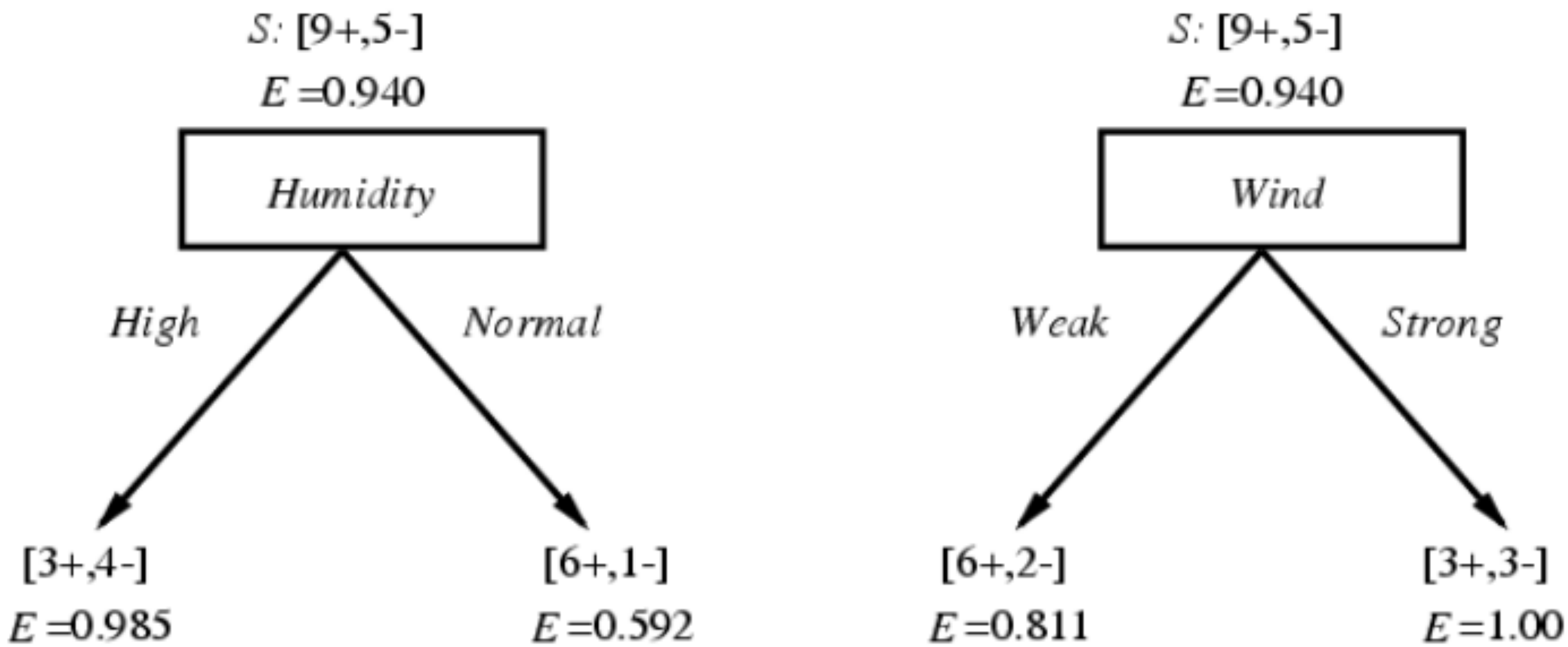
- IG is the expected reduction in entropy of the target variable X, due to sorting on variable A

Exercise: Play tennis?

Outlook	Tem	Humid	Windy	Play
Sunny	Hot	High	FALSE	NO
Sunny	Hot	High	TRUE	NO
Overcast	Hot	High	FALSE	YES
Rainy	Mild	High	FALSE	YES
Rainy	Cool	Norm	FALSE	YES
Rainy	Cool	Norm	TRUE	NO
Overcast	Cool	Norm	TRUE	YES

Outlook	Tem	Humid	Windy	Play
Sunny	Mild	High	FALSE	NO
Sunny	Cool	Norm	FALSE	YES
Rainy	Mild	Norm	FALSE	YES
Sunny	Mild	Norm	TRUE	YES
Overcast	Mild	High	TRUE	YES
Overcast	Hot	Norm	FALSE	YES
Rainy	Mild	High	TRUE	NO

Which attribute is the best classifier?

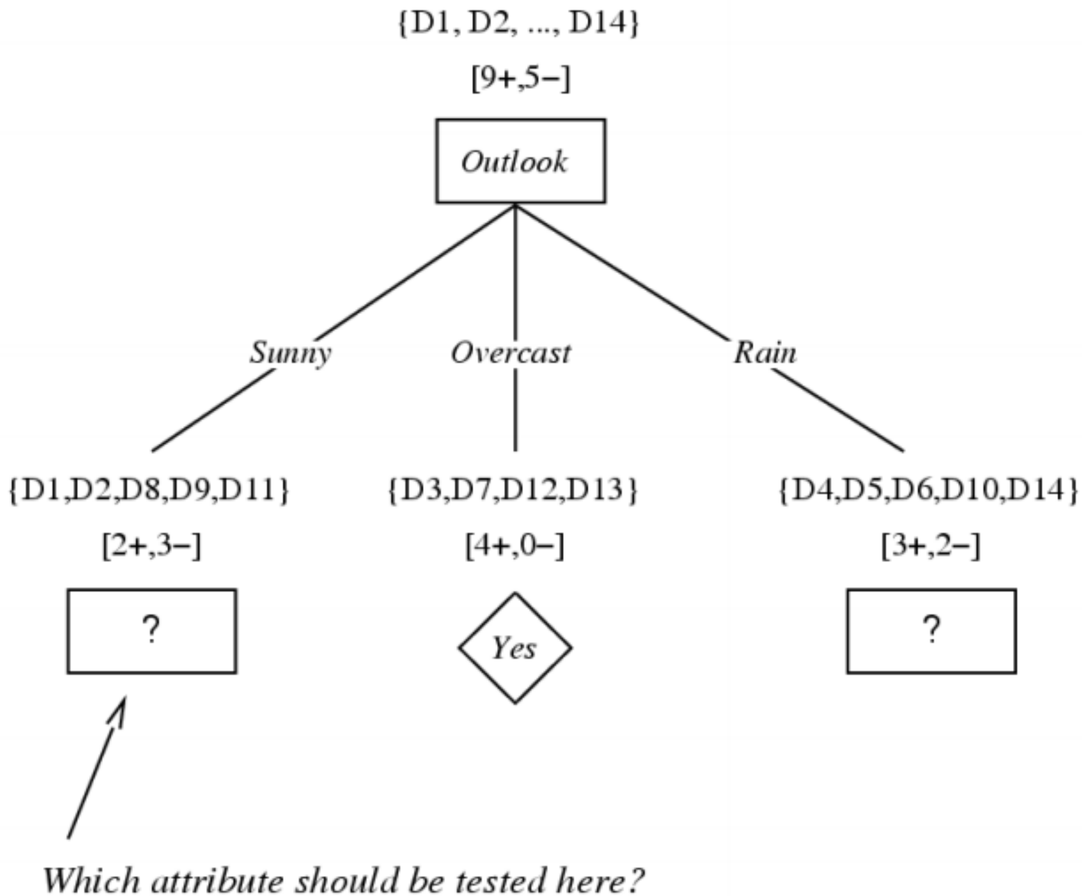


Gain (S, Humidity)

= .940 - (7/14).985 - (7/14).592
= .151

Gain (S, Wind)

= .940 - (8/14).811 - (6/14)1.0
= .048



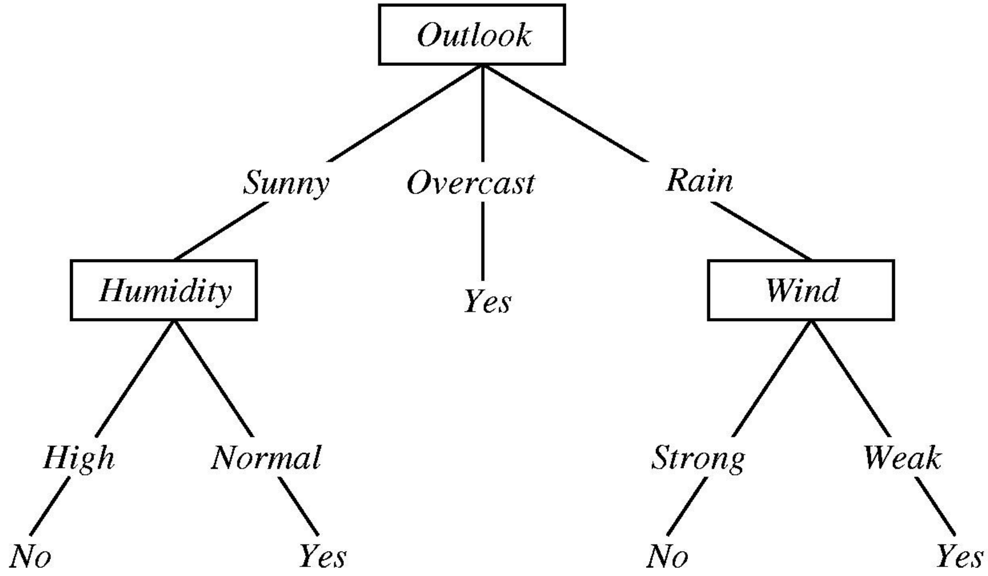
S_{sunny} = {D1,D2,D8,D9,D11}

$Gain\ (S_{sunny}, Humidity) = .970 - (3/5)\ 0.0 - (2/5)\ 0.0 = .970$

$Gain\ (S_{sunny}, Temperature) = .970 - (2/5)\ 0.0 - (2/5)\ 1.0 - (1/5)\ 0.0 = .570$

$Gain\ (S_{sunny}, Wind) = .970 - (2/5)\ 1.0 - (3/5)\ .918 = .019$

Overfitting in Decision Trees



Consider adding a noisy training example:

Sunny, Hot, Normal, Strong, PlayTennis=No

What effect on tree?

Overfitting

Consider error of hypothesis h over

- training data: $error_{train}(h)$
- entire distribution $\mathcal{D}$ of data: $error_{\mathcal{D}}(h)$

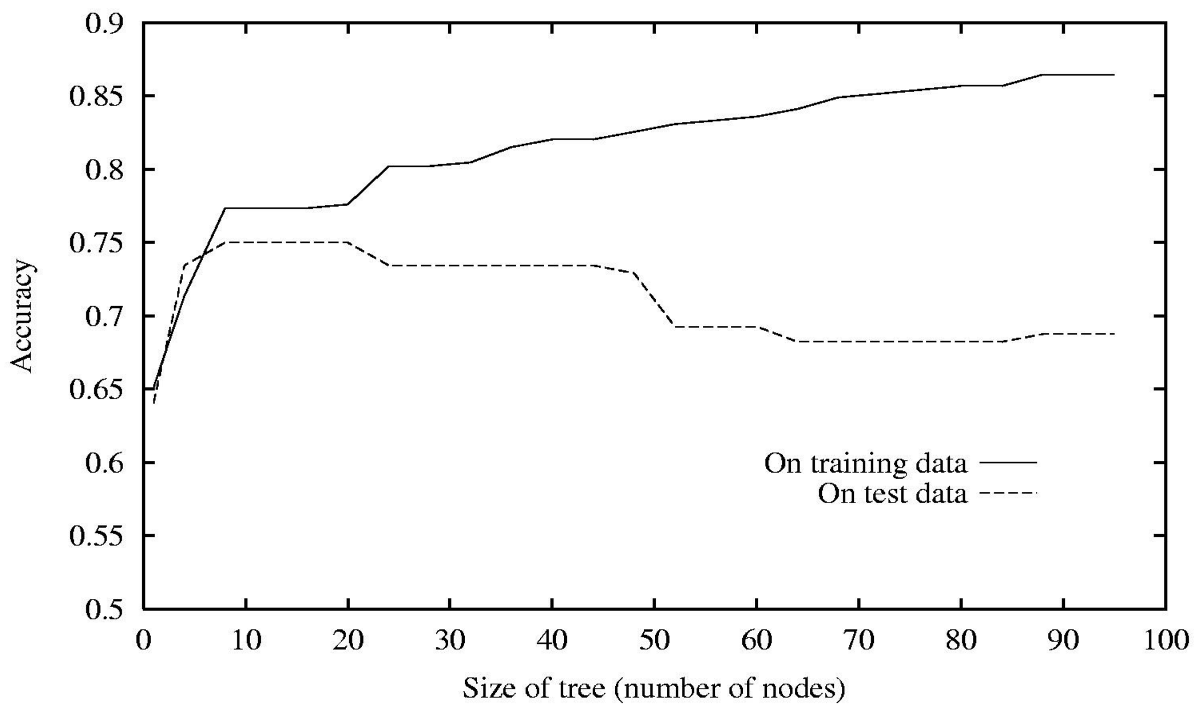
Hypothesis $h \in H$ **overfits** training data if there is an alternative hypothesis $h' \in H$ such that

$$error_{train}(h) < error_{train}(h')$$

and

$$error_{\mathcal{D}}(h) > error_{\mathcal{D}}(h')$$

Overfitting in Decision Tree Learning



Avoiding Overfitting

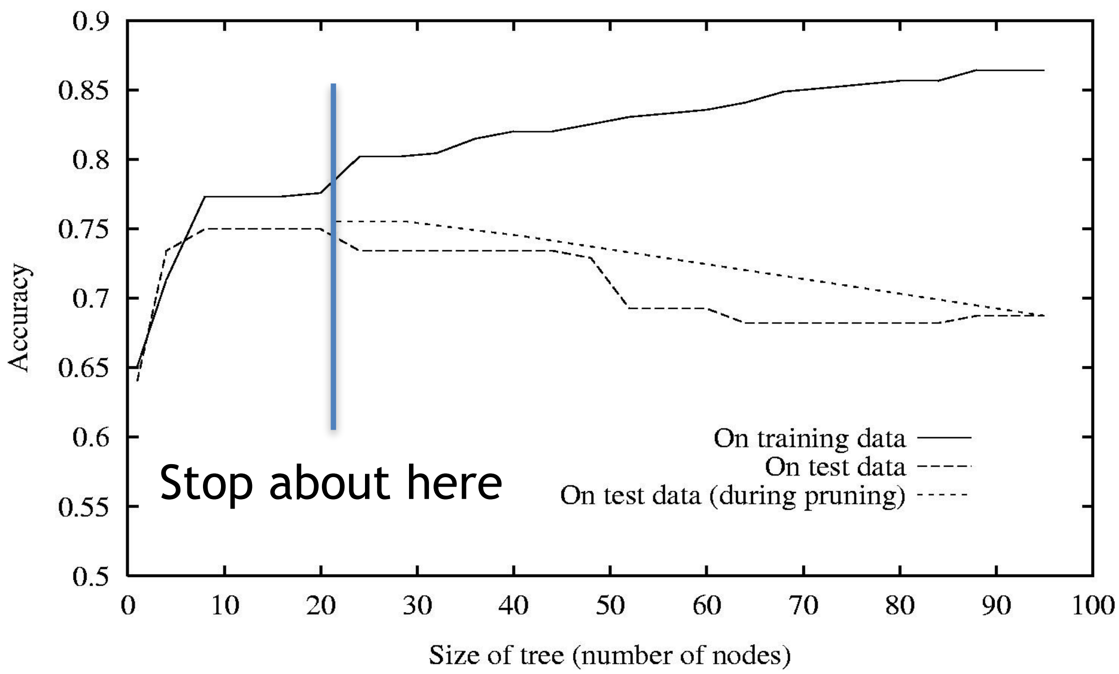
How can we avoid overfitting?

- Stop growing when data split not statistically significant
- Grow full tree, then post-prune

How to select "best" tree:

- Measure performance over training data
- Measure performance over separate validation data set
- Add complexity penalty to performance measure

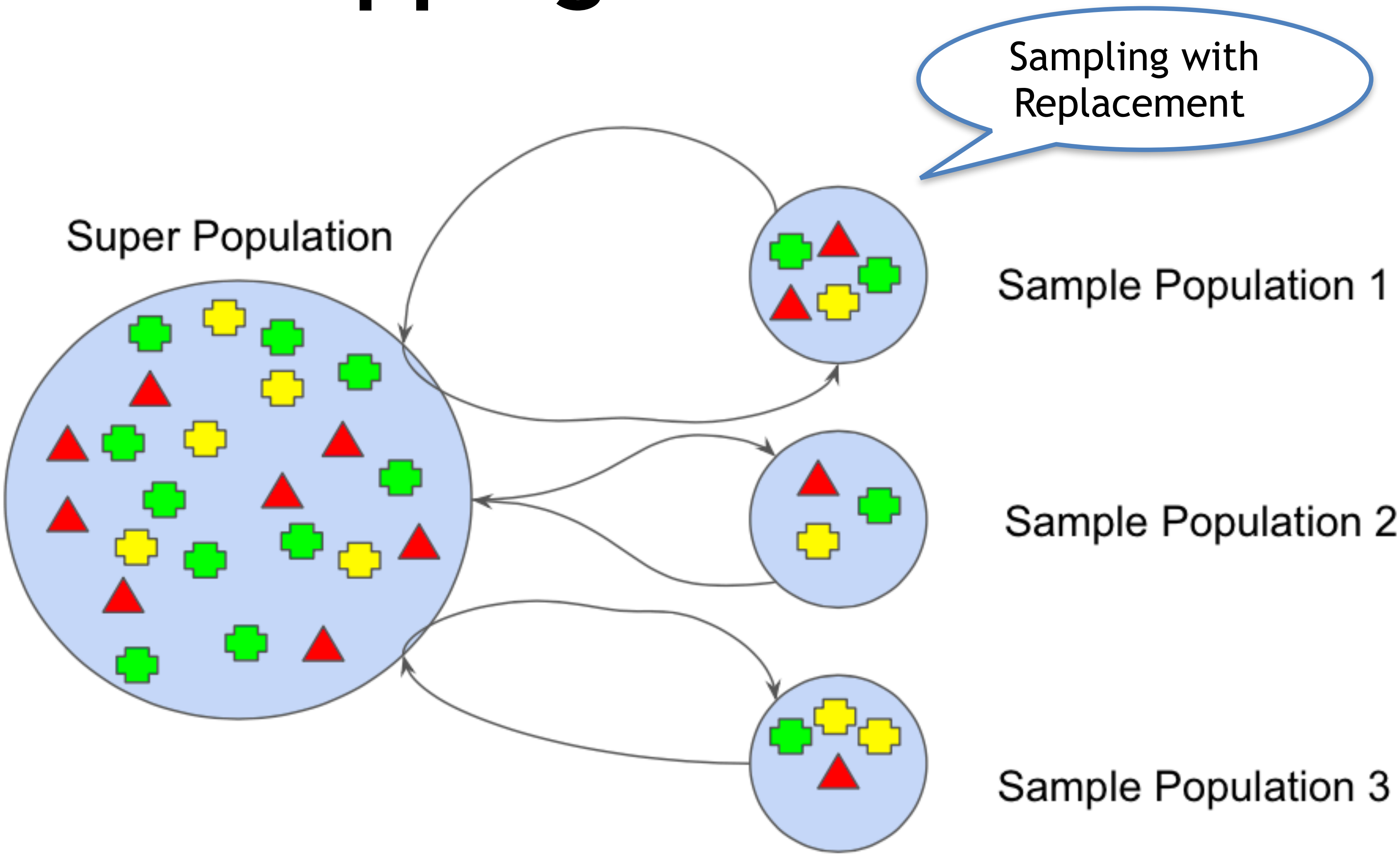
Effect of Reduced-Error Pruning



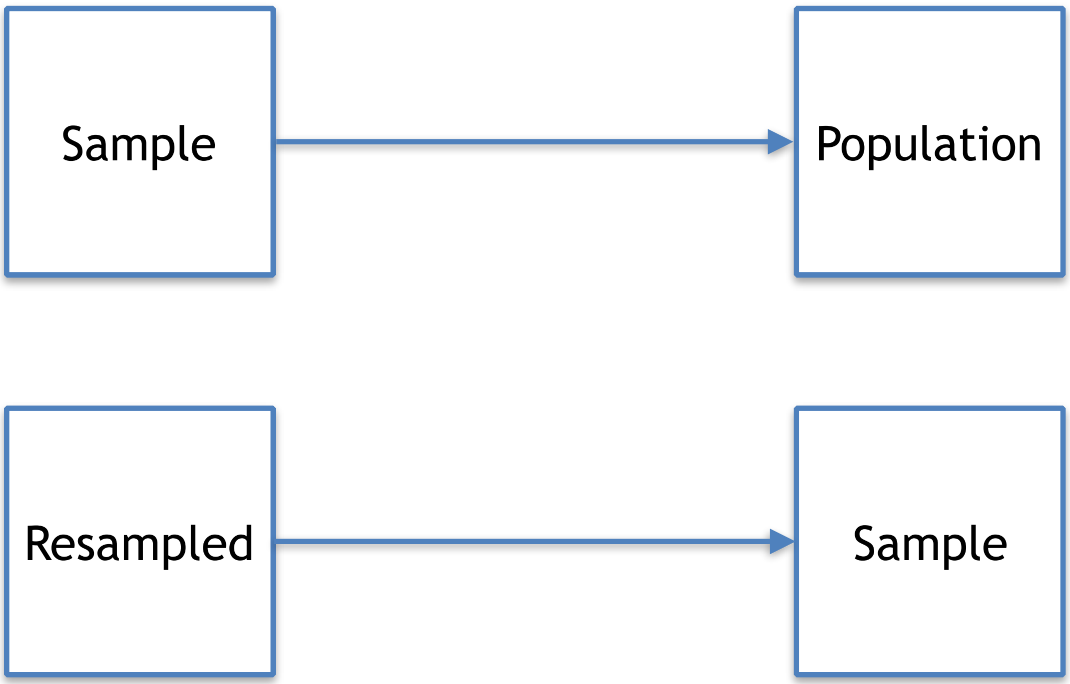
Ensemble Learning

- Combines a set of weak hypotheses (classifiers) to create a strong classifier that obtains better performance than a single one.

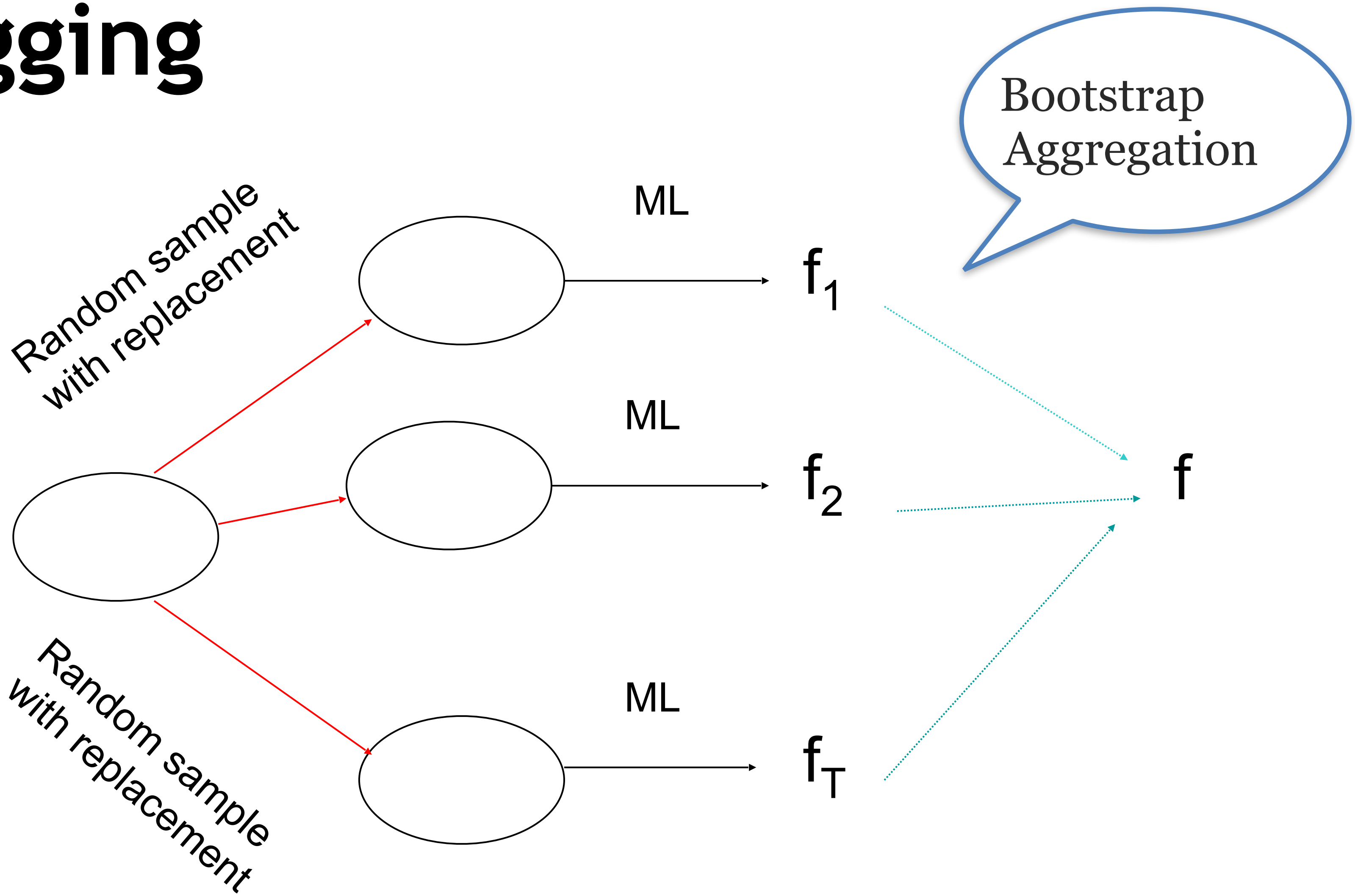
Bootstrapping



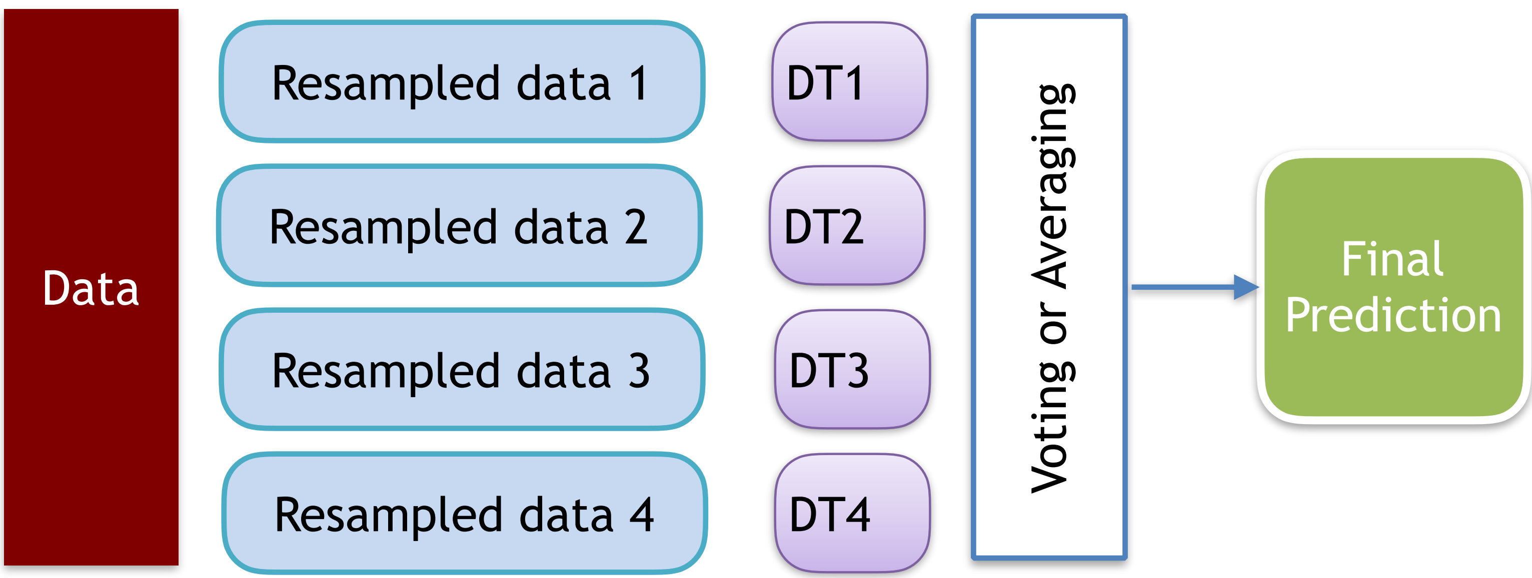
Learning about population from sample



Bagging


Bootstrap Aggregation
Random sample with replacement
Random sample with replacement
ML
ML
ML
f_1
f_2
f_T
f

Random Forest



More detailed in Python notebook