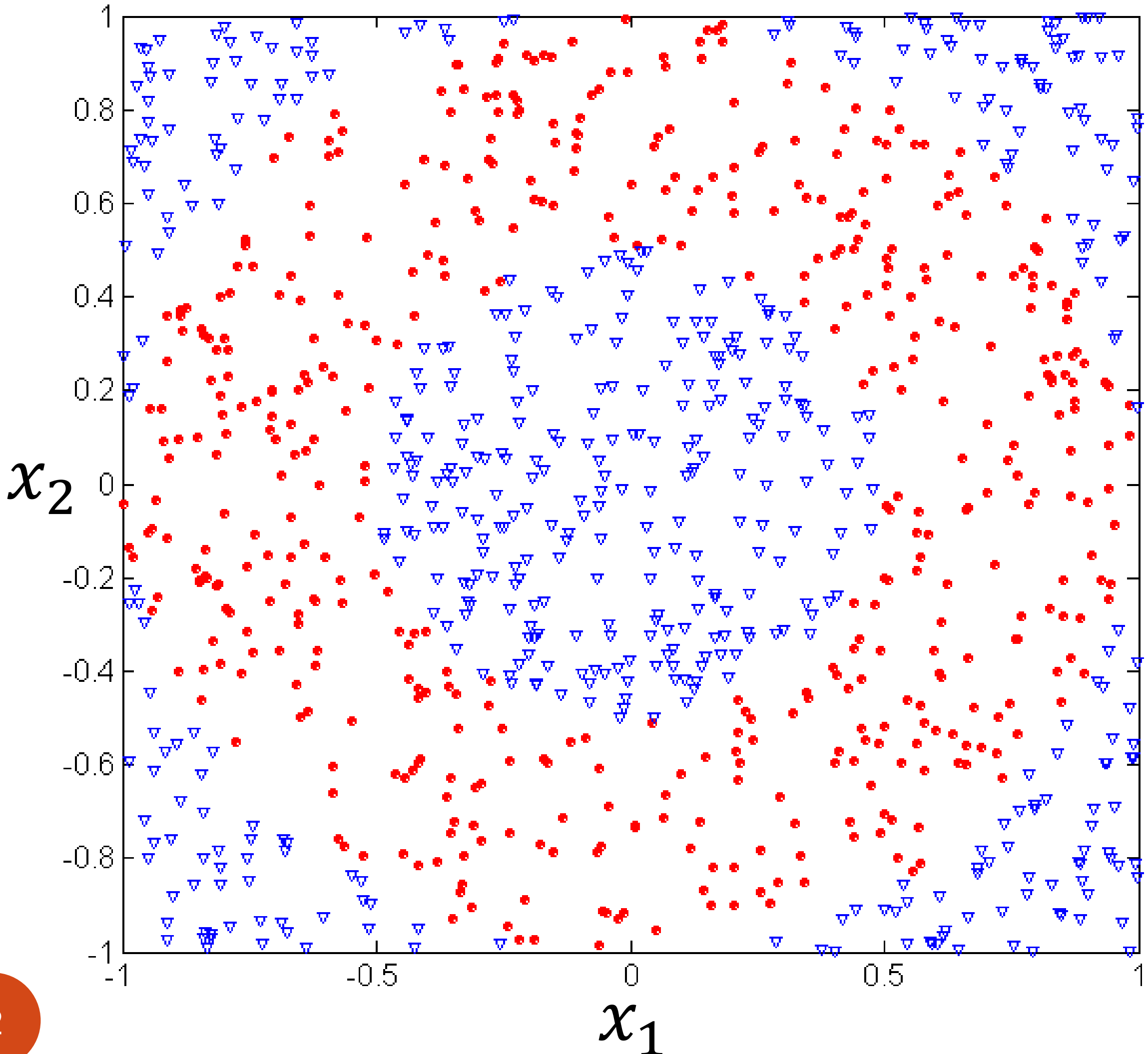
CZ4041/CE4041: Machine Learning

Lesson 6a: Generalization

Sinno Jialin PAN

School of Computer Science and Engineering, NTU, Singapore

Underfitting and Overfitting (Example)



500 circular and 500 triangular data points.

Circular points:

$$0.5 \leq \sqrt{x_1^2 + x_2^2} \leq 1$$

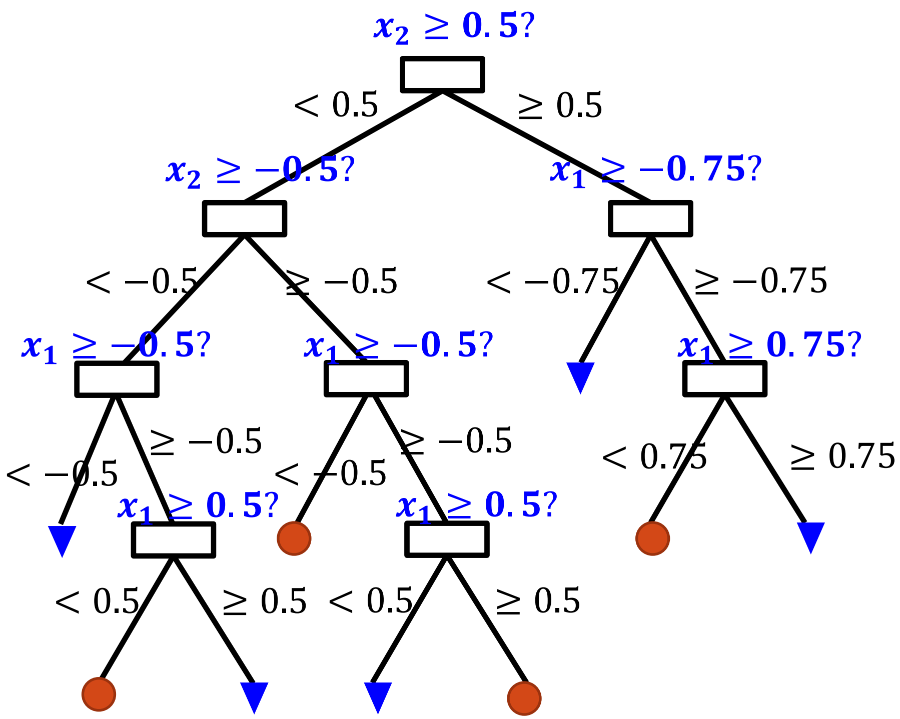
Triangular points:

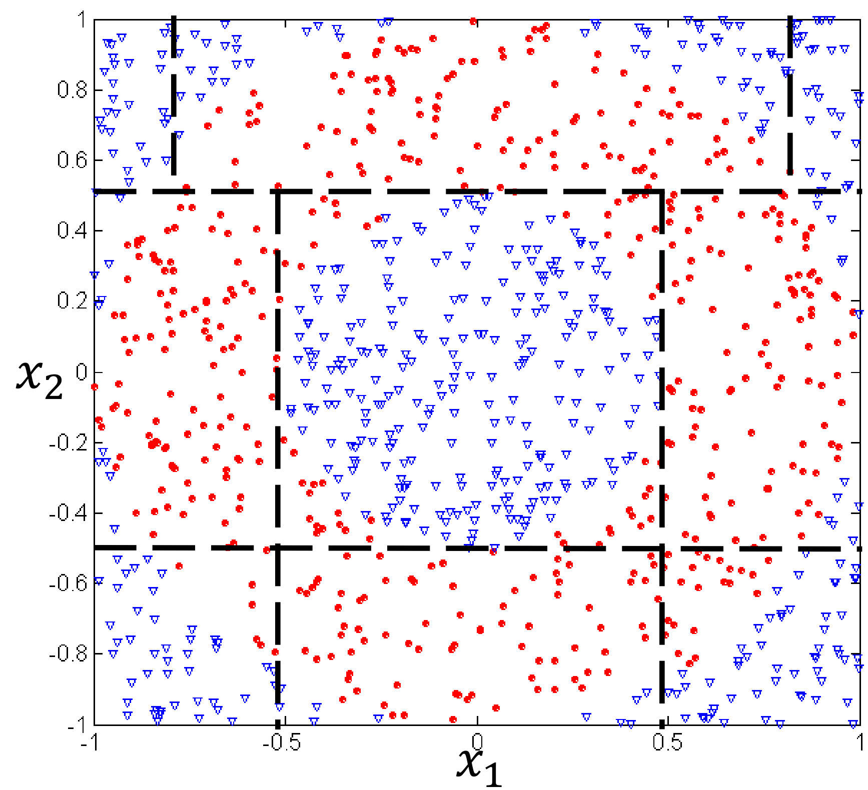
$$\sqrt{x_1^2 + x_2^2} > 1$$

or

$$\sqrt{x_1^2 + x_2^2} < 0.5$$

Overfitting v.s. Model Complexity

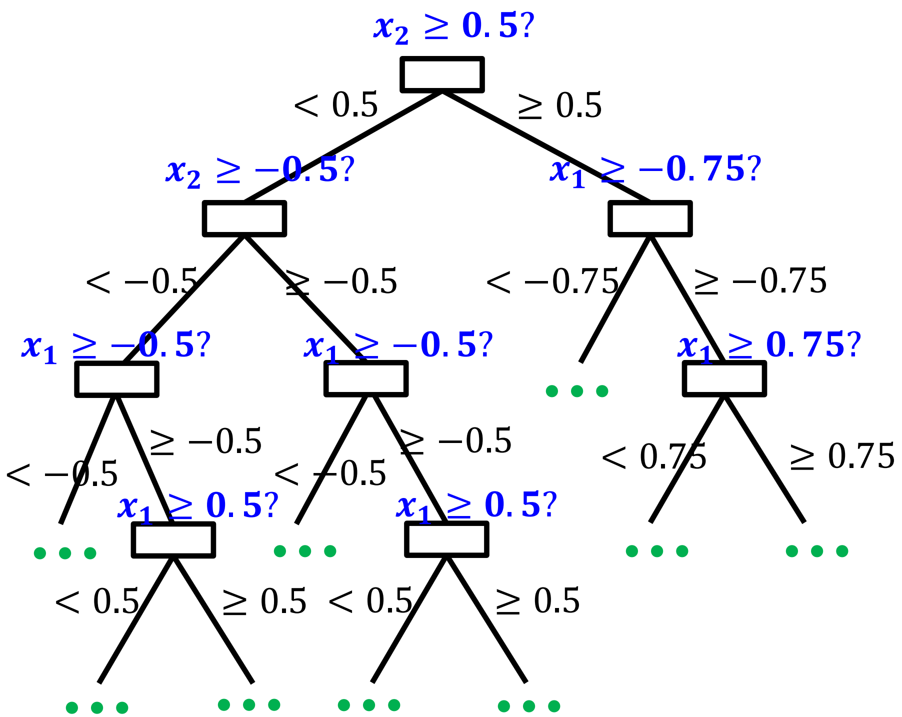


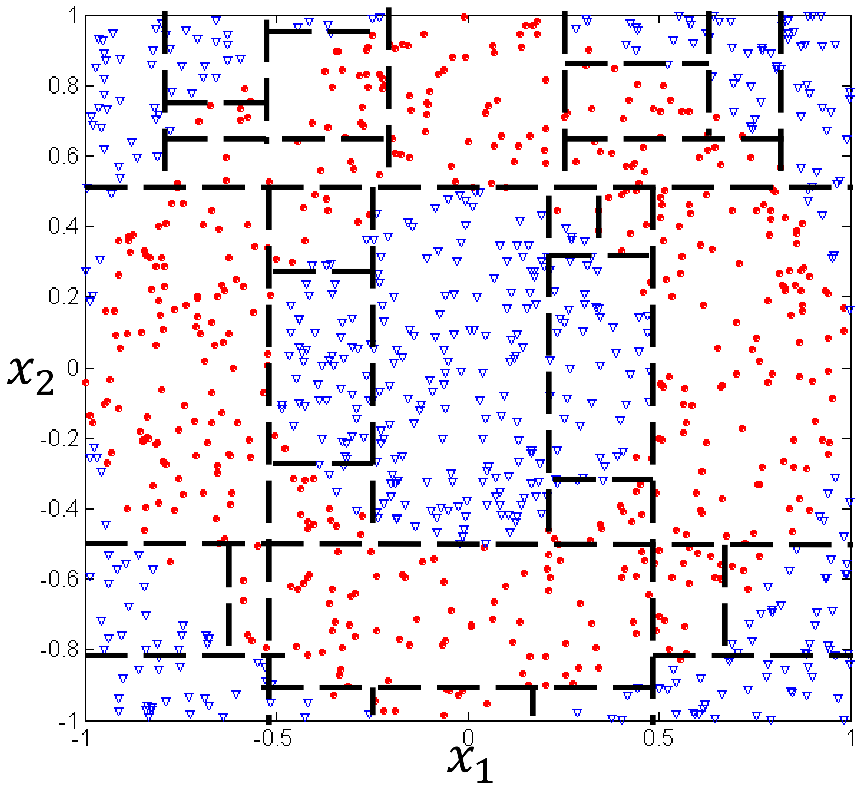


Training errors (#misclassified training data) = 100 +

Decision tree with 9 leaf nodes

Overfitting v.s. Model Complexity

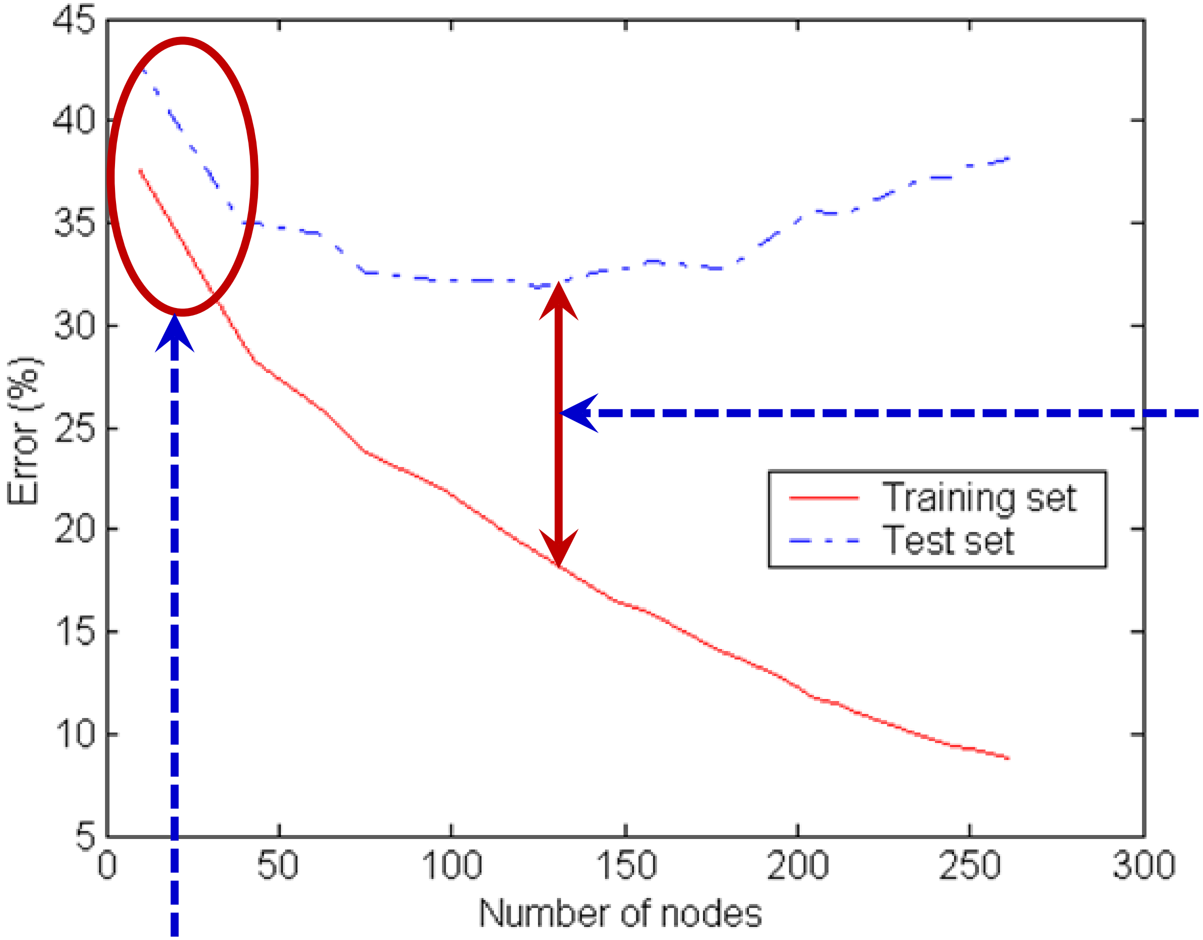




Training errors = 20

Decision tree with 30 + leaf nodes

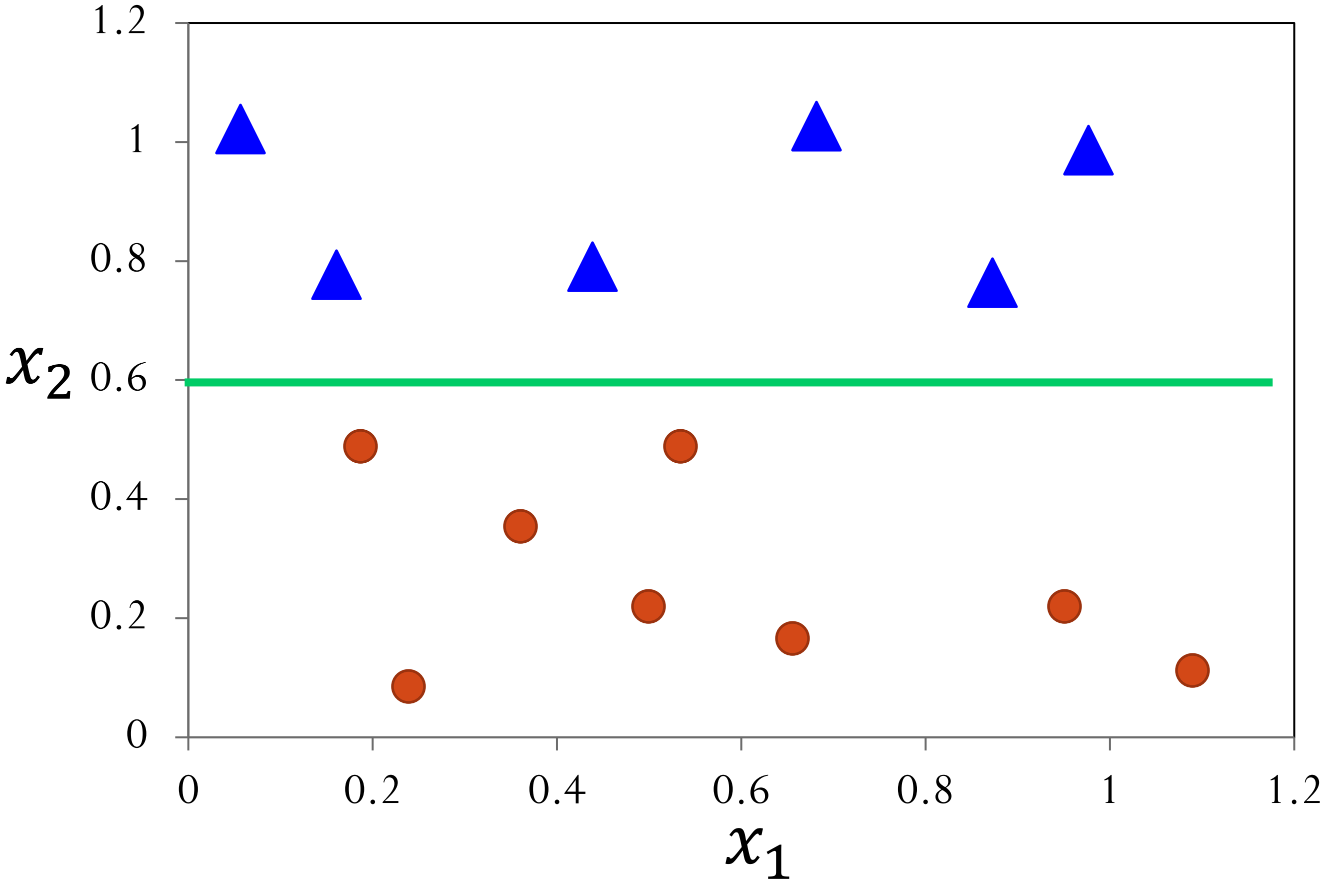
Underfitting and Overfitting



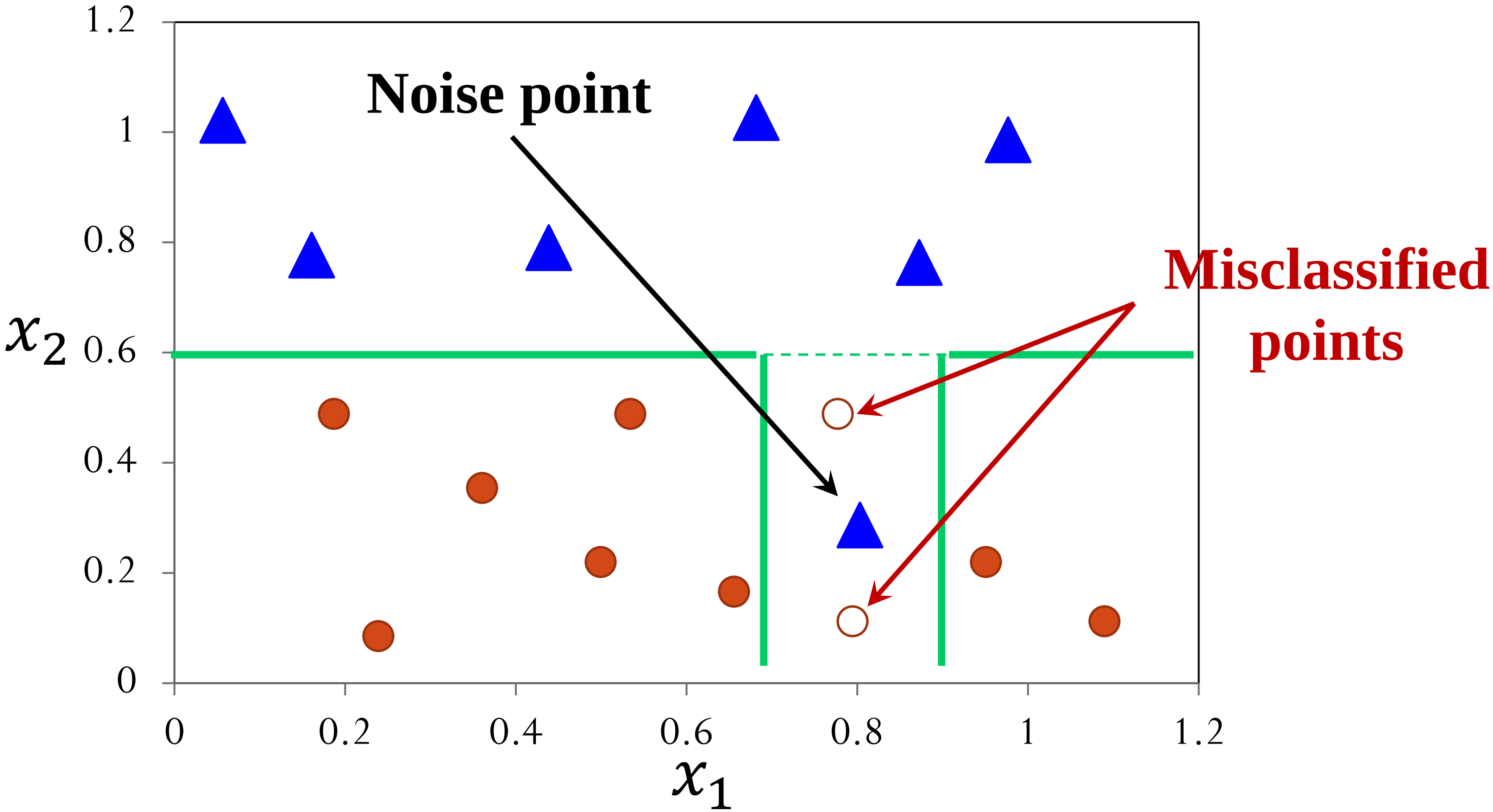
Overfitting: when test error rate begins to increase even though training error rate continues to decrease

Underfitting: when model is too simple, both training and test error rates are large

Overfitting due to Noise



Overfitting due to Noise (cont.)



Decision boundary is distorted by noise point

Notes on Overfitting

- Overfitting results in decision trees that are more complex than necessary
- Training error no longer provides a good estimate of how well the tree will perform on previously unseen records
- Need new ways for estimating errors

Overfitting v.s. Model Complexity (cont.)

- How do we determine the right model complexity?
- A model with ideal complexity is the one that produces the lowest generalization error
- No knowledge of the test data and how well the model will perform on unseen data
- The best it can do is to **estimate** the generalization error of the induced model

Estimation of Generalization Errors

- Training errors: error on the training set: $e(T)$
- Generalization errors: error on previously unseen testing set: $e'(T)$

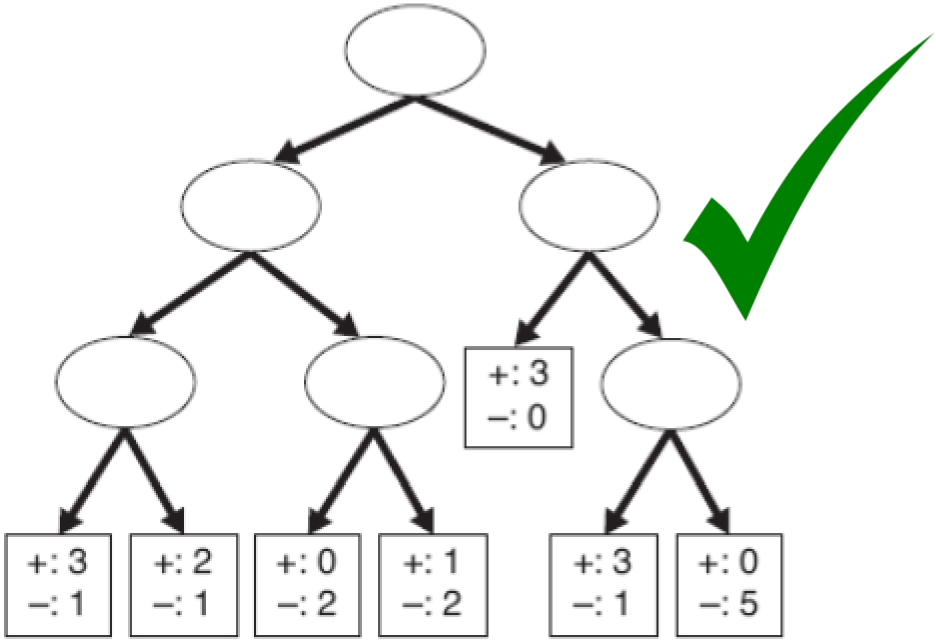
Estimation of Generalization Errors

- Approaches to estimating generalization errors:
  - Optimistic Estimate: $e'(T) = e(T)$
  - Incorporating model complexity
    - Occam's Razor
      - Pessimistic Error Estimate
  - Using Validation Set

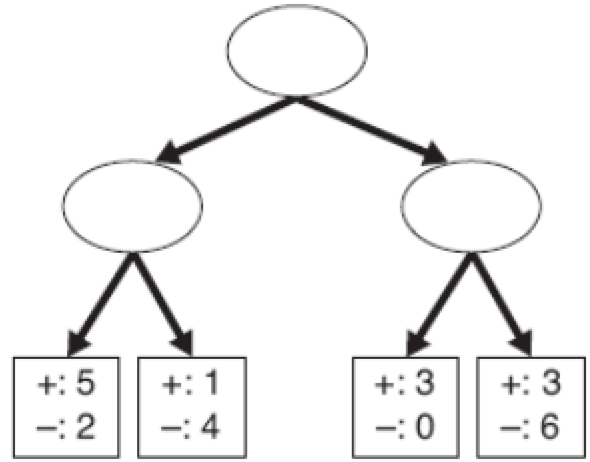
Optimistic Estimate

- Assume that the training set is a good representation of the overall data
- The training error can be used to provide an optimistic estimate for the generalization error
  - $e'(T) = e(T)$
- A decision tree induction algorithm selects the model that produces the lowest training error rate

An Example of Optimistic Estimate



Decision Tree, T_L



Decision Tree, T_R

$$e(T_L) = 4$$

$$e(T_R) = 6$$

$$e(T_L)\text{ rate} = \frac{4}{24} = 0.167$$

$$e(T_L)\text{ rate} = \frac{6}{24} = 0.25$$

Estimation of Generalization Errors

- Approaches to estimating generalization errors:
  - Optimistic Estimate: $e'(T) = e(T)$
  - Incorporating model complexity
    - Occam's Razor
      - Pessimistic Error Estimate
  - Using Validation Set

Occam's Razor

- Definition: Given two models of similar generalization errors, one should prefer the simpler model over the more complex model
- For complex models, there is a greater chance that it was fitted accidentally by errors in data
- Therefore, one should include model complexity when evaluating a model

"Everything should be made as simple as possible, but no simpler." – Albert Einstein

Pessimistic Error Estimate

- Idea: explicitly computes generalization error as the sum of training error and a penalty term for model complexity

$$e'(T) = e(T) + \Omega(T)$$

- In a decision tree, we can define a penalty term of $k > 0$ on each leaf node, i.e.,

$$e'(t) = e(t) + \Omega(t) = e(t) + k$$

- Then,

$$e'(T) = e(T) + \boxed{N} \times \boxed{k}$$

Total number of leaf nodes

$k > 0$, e.g., $k = 0.5$

Example

- For a tree with 30 leaf nodes and 10 errors on training (out of 1,000 instances):
  - Training errors $= 10$
  - Training error rate $= \frac{10}{1000} = 1\%$
  - Generalization errors $= 10 + 30 \times 0.5 = 25$
  - Generalization error rate $= \frac{25}{1000} = 2.5\%$

Estimation of Generalization Errors

- Approaches to estimating generalization errors:
  - Optimistic Estimate: $e'(T) = e(T)$
  - Incorporating model complexity
    - Occam's Razor
      - Pessimistic Error Estimate
  - Using Validation Set

Using a Validation Set

- Divide the original training data set into two smaller subsets
- One is for training, the other (known as the validation set) is for estimating the generalization error
- The complexity of the best model can be estimated based on the performance of the model on the validation set

How to Address Overfitting

- Pre-Pruning (Early Stopping Rule)
  - Stop the algorithm before it becomes a fully-grown tree
  - Typical stopping conditions for a node:
    - Stop if all instances belong to the same class
    - Stop if all the feature values are the same

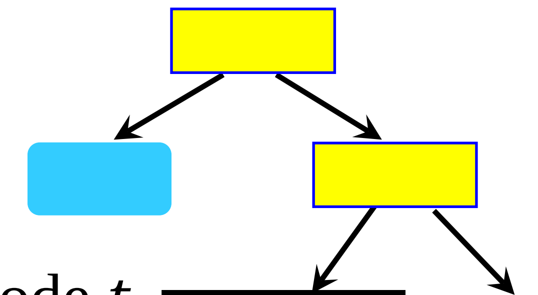
How to Address Overfitting (cont.)

- Pre-Pruning (Early Stopping Rule)
  - More restrictive conditions:
    - Stop if number of instances is less than some user-specified threshold
    - Stop if expanding the current node does not improve generalization errors

Pre-Pruning Example

Generalization errors: $e'(T) = e(T) + \Omega(T) = e(T) + N \times k$

e.g., $k = 0.5$



Node t

Expand or not?

$+1$	20
-1	10

$+$

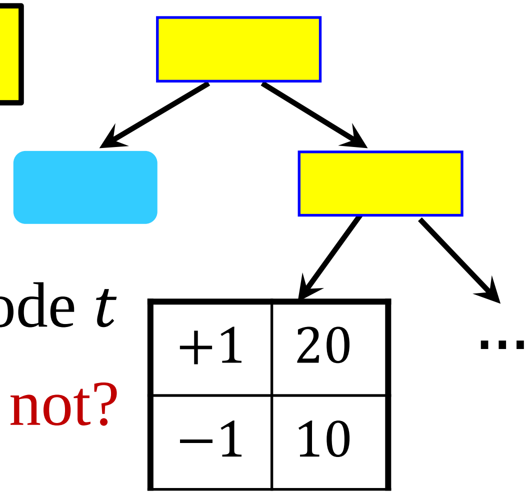
$$e'(\boxed{t}) = e(t) + \Omega(t)$$
$$= 10 + 1 \times 0.5$$
$$= 10.5$$

A subtree with node t as its root

Pre-Pruning Example (cont.)

Generalization errors: $e'(T) = e(T) + N \times 0.5$

Not expand!

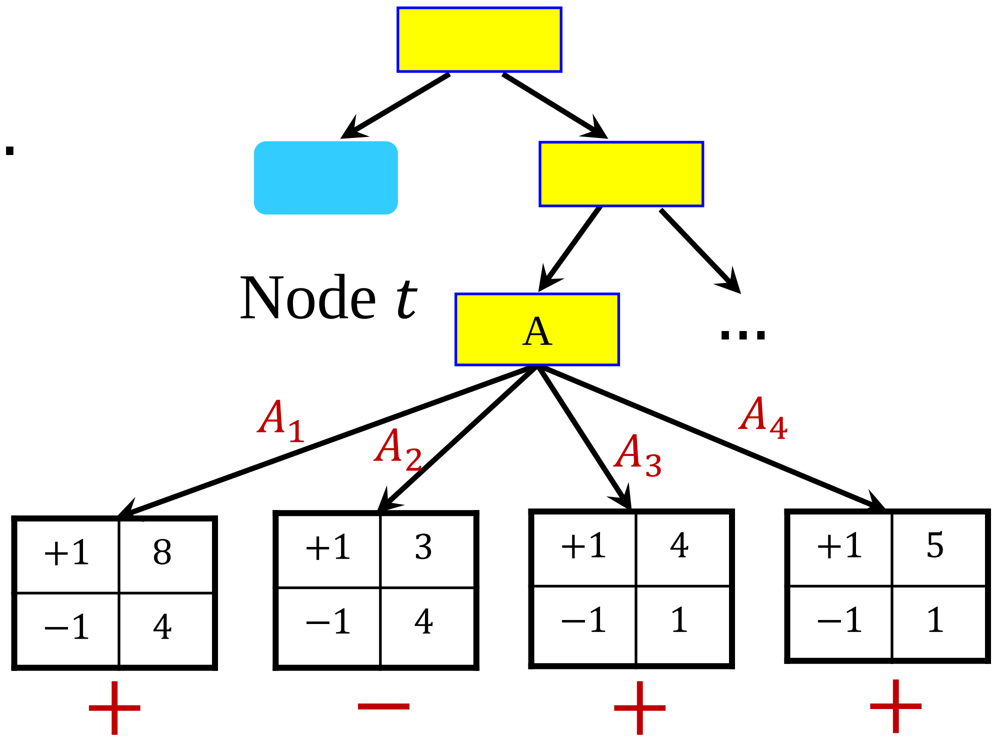


Expand or not?

A subtree with node t as its root

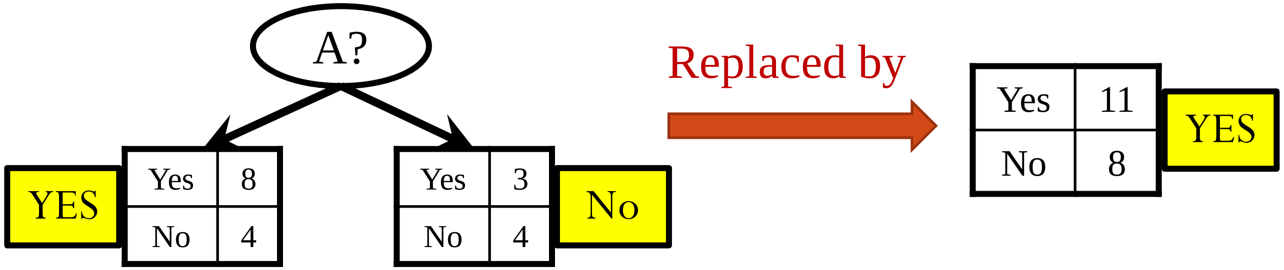
$$e'(\boxed{t}) = e(t) + \Omega(t)$$
$$= 9 + 0.5 \times 4$$
$$= 11$$

If yes, suppose A is best feature to conduct condition test



How to Address Overfitting (cont.)

- Post-pruning
  - Grow decision tree to its entirety
  - Trim the nodes of the decision tree in a bottom-up fashion
  - If generalization error improves after trimming, replace sub-tree by a new leaf node
  - Class label of leaf node is determined from majority class of instances in the sub-tree

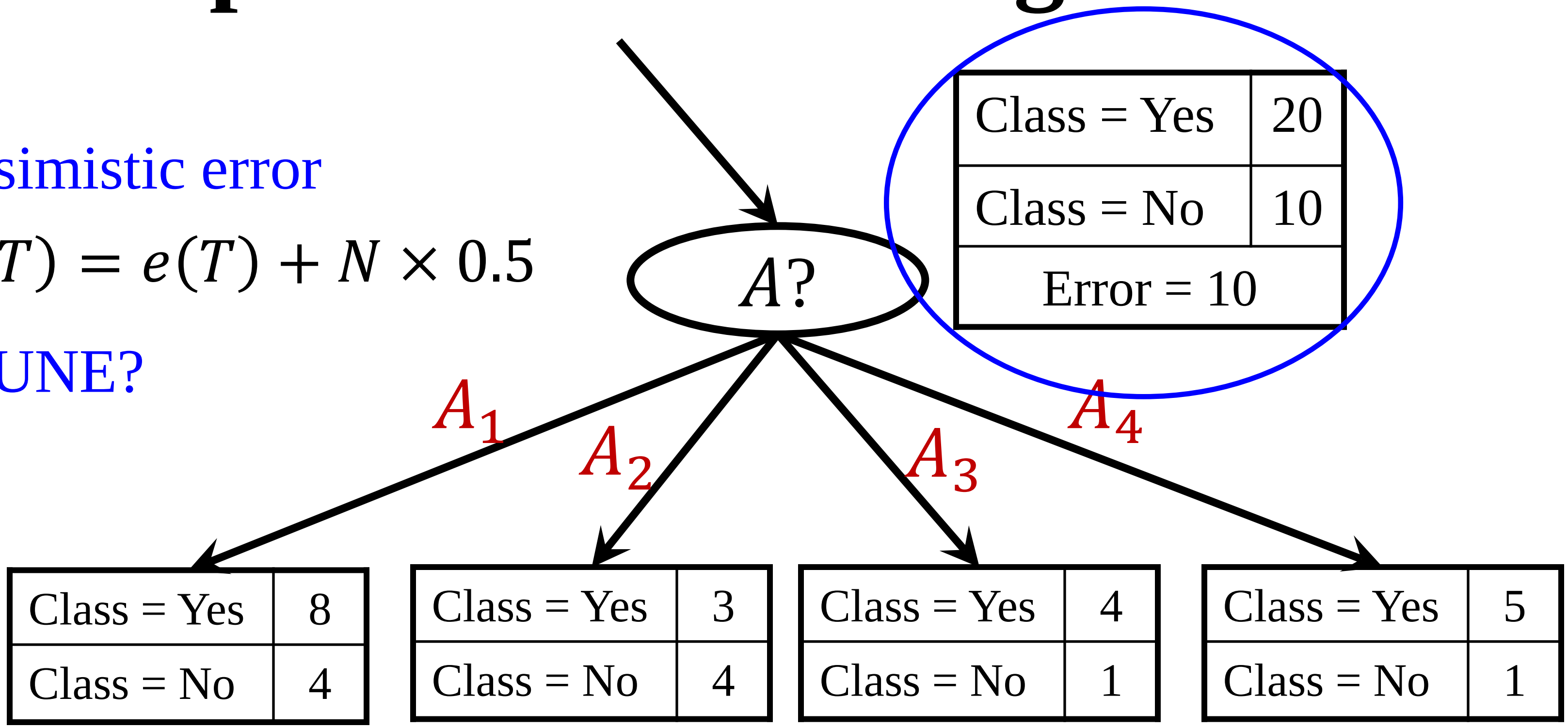


Example of Post-Pruning

Pessimistic error

$$e'(T) = e(T) + N \times 0.5$$

PRUNE?



Training errors (before pruning) $= 4 + 3 + 1 + 1 = 9$
Pessimistic errors (before pruning) $= 9 + 4 \times 0.5 = 11$

PRUNE!

Training errors (after pruning) $= 10$
Pessimistic errors (after pruning) $= 10 + 0.5 = 10.5$

Examples of Post-pruning

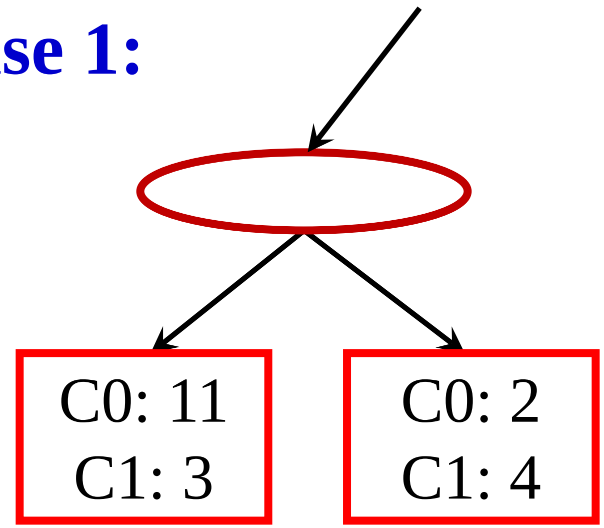
- Pessimistic error?

$$e'(T) = e(T) + N \times 0.5$$

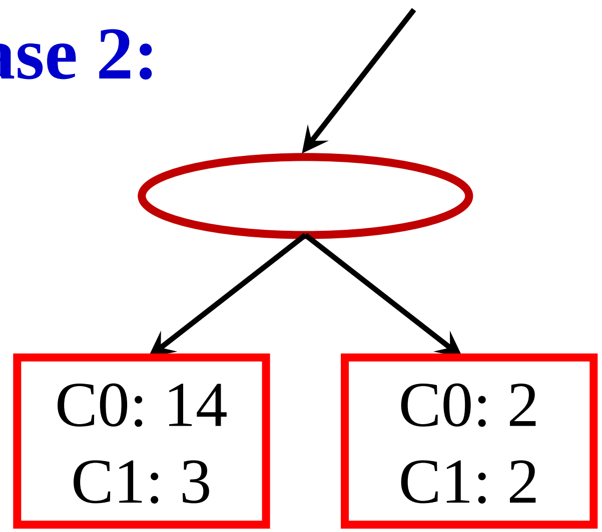
PRUNE?

Tutorial

Case 1:



Case 2:



In Practice

criterion : *{"gini", "entropy"}, default="gini"*

The function to measure the quality of a split. Supported criteria are "gini" for the Gini impurity and "entropy" for the information gain.

max_depth : *int, default=None*

The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

max_features : *int, float or {"auto", "sqrt", "log2"}, default=None*

The number of features to consider when looking for the best split:

- If int, then consider `max_features` features at each split.
- If float, then `max_features` is a fraction and `int(max_features * n_features)` features are considered at each split.
- If "auto", then `max_features=sqrt(n_features)`.
- If "sqrt", then `max_features=sqrt(n_features)`.
- If "log2", then `max_features=log2(n_features)`.
- If None, then `max_features=n_features`.

Thank you!