Week 8-1

Machine Learning 1: Fundamentals, Decision Trees

Big Data

Prof. Hwanjo Yu
POSTECH

Supervised vs. Unsupervised learning

- Supervised learning (classification)
  - The training data (observations, measurements, etc.) are accompanied by labels indicating the class of the observations
  - New data (unlabeled data) is classified based on the training set
- Unsupervised learning (clustering)
  - The class labels of training data is unknown
  - Given a set of measurements, observations, etc. with the aim of establishing the existence of classes or clusters in the data
  - Group the data based on some similarity or distance measure

Classification vs. Regression vs. Ranking

- They have the similar purpose
  - Constructs a model based on the training dataset (labeled data), and use the model to classify or predict new data (unlabeled data)
- Difference
  - Classification: The target (class) is categorical (or nominal)
  - Regression: The target (value) is continuous (or real)
  - Ranking: input label is discrete ranking or relative ordering, and output label is ranking score (real)
- Applications
  - Credit/loan approval:
  - Medical diagnosis: if a tumor is cancerous or benign
  - Fraud detection: if a transaction is fraudulent
  - …

Prediction?

- Classification, regression, ranking can be also used for prediction problems
  - Whether forecast
  - Disease prognosis
  - Stock price prediction
  - …

Classification — Multi-Step process

- Data preprocessing
  - Data cleaning, integration, reduction, transformation (normalization, discretization)
- Training (learning model)
  - Construct a model describing a set of predetermined classes
  - Each tuple/sample belongs to a predefined class
  - The set of tuples used for model construction is training set
  - The model is represented as classification rules, decision trees, or mathematical formula
- Validation (tuning model)
  - Evaluate the accuracy of the model
  - Tune the parameters of the model
  - Validation set: labeled data that are <u>excluded</u> from the training set
- Deploy
  - Retrain the model with the best parameter values on the entire data
  - Classify future or unknown objects using the model

Learning model

Training Data

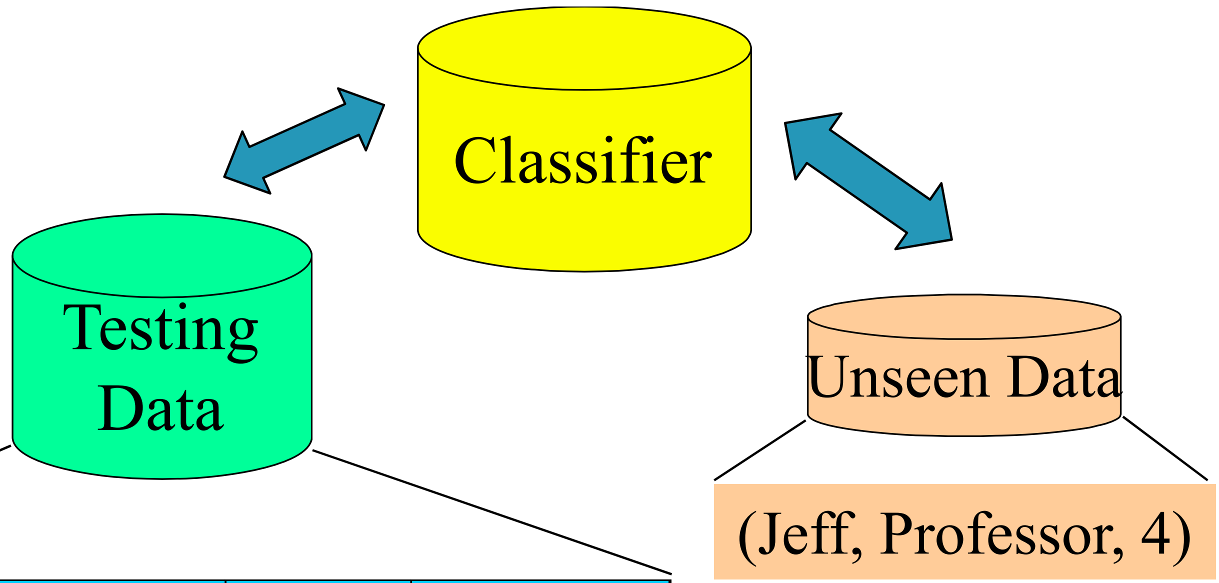
NAME	RANK	YEARS	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no

Classification Algorithms

Classifier (Model)

IF rank = ‘professor’
OR years > 6
THEN tenured = ‘yes’

Predicting using the model



NAME	RANK	YEARS	TENURED
Tom	Assistant Prof	2	no
Merlisa	Associate Prof	7	no
George	Professor	5	yes
Joseph	Assistant Prof	7	yes

Tenured?

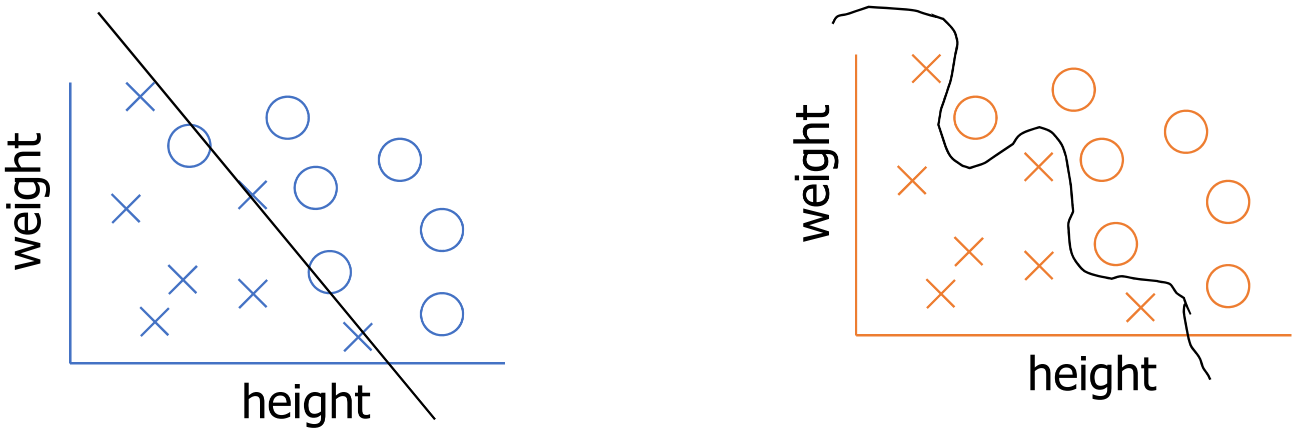
Yes

Issues: Evaluating learning methods

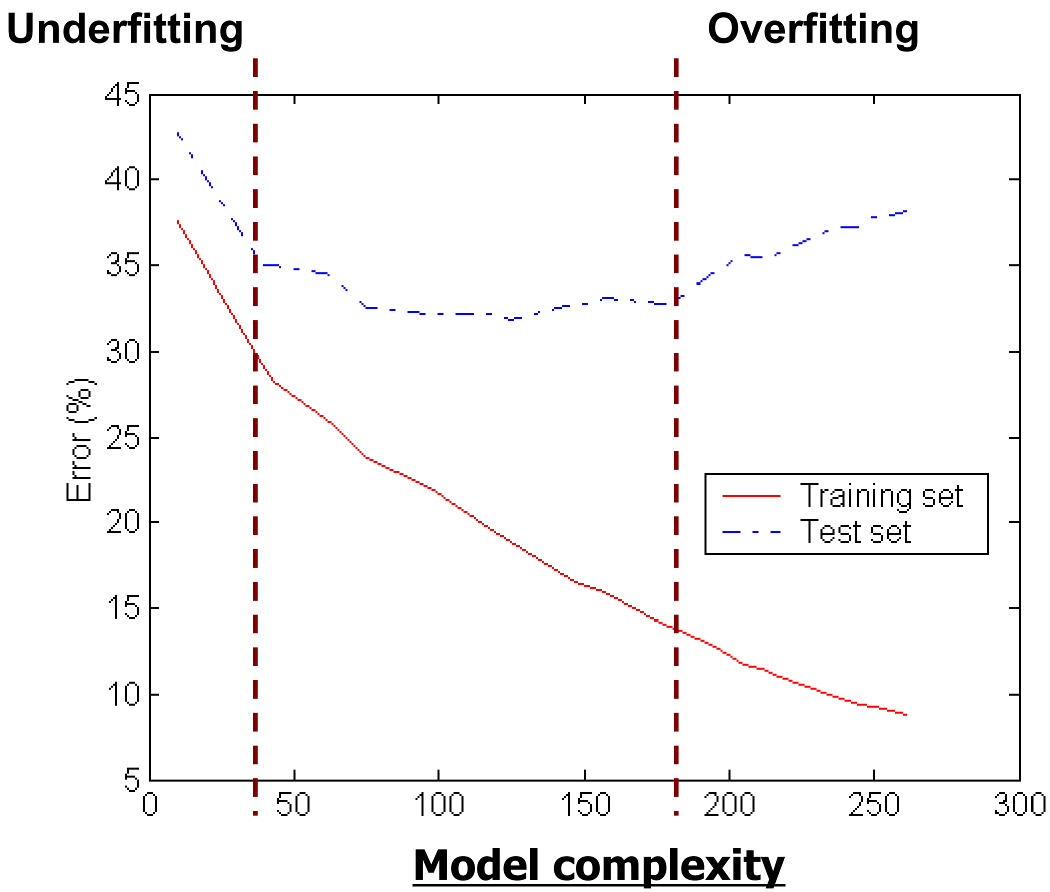
- Accuracy
  - How much accurately the model classifies?
  - Avoid overfitting => improve generalization
- Speed
  - Training time: Time to construct the model
  - Testing time: Time to classify a new data
- Robustness
  - Handling noise and missing values
- Interpretability
  - The model is understandable or interpretable?

Overfitting

- Fitting the model exactly to the data is usually not a good idea.
- The resulting model may not generalize well to unseen data.



Generalization error, variance-bias trade-off



$E(y - f(x))^2 = Var(f) + Bias(f)^2 + Var(\varepsilon)$

Generalization error = Variance + Bias^2

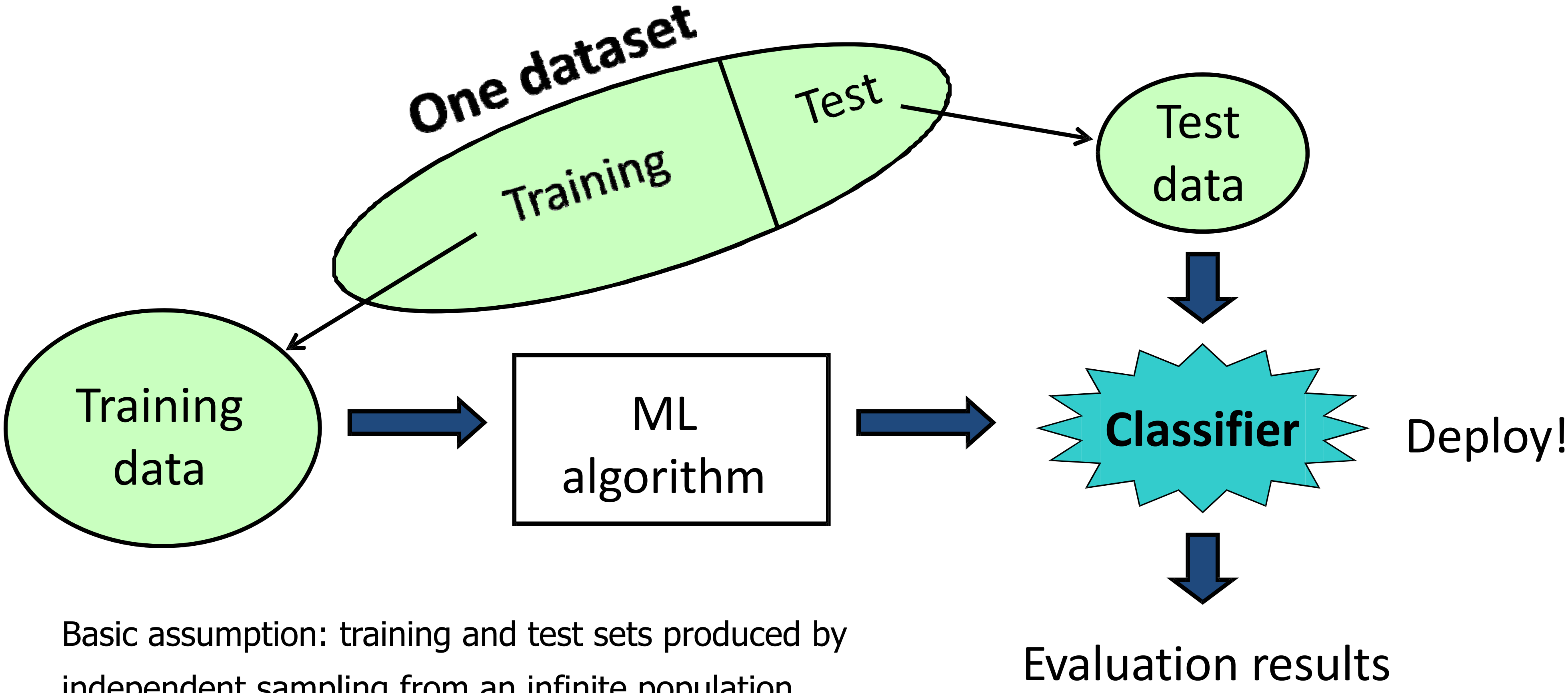
Variance : the amount by which f would change if we estimated it using a different training data set

Bias : the error that is introduced by approximating a real-life problem, which may be extremely complicated, by a much simpler model.

Occam's razor

- Given two models (or solutions) of similar performance, one should prefer the simpler model (or solution) over the more complex model (or solution)
- For complex models, there is a greater chance of overfitting.
- Therefore, one should include model complexity when evaluating a model

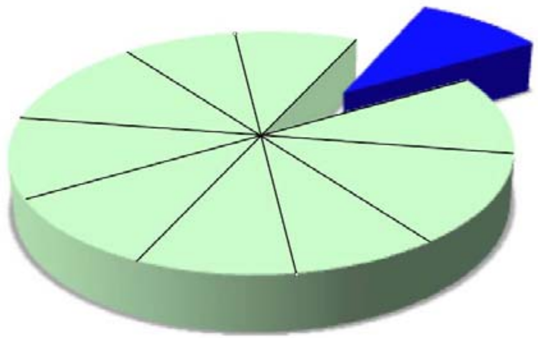
How to estimate generalization performance?



Basic assumption: training and test sets produced by independent sampling from an infinite population

How to estimate generalization performance?

- 10 fold cross-validation
  - Divide dataset into 10 parts (folds)
  - Hold out each part in turn: testing on 1 fold and training on 9 folds
  - Repeat 10 times, each with different fold for testing
  - Average the results
  - Each data point used once for testing, 9 times for training



- Stratified cross-validation
  - Ensure that each fold has the right(?) proportion of each class value

Validation method: Estimating generalization performance

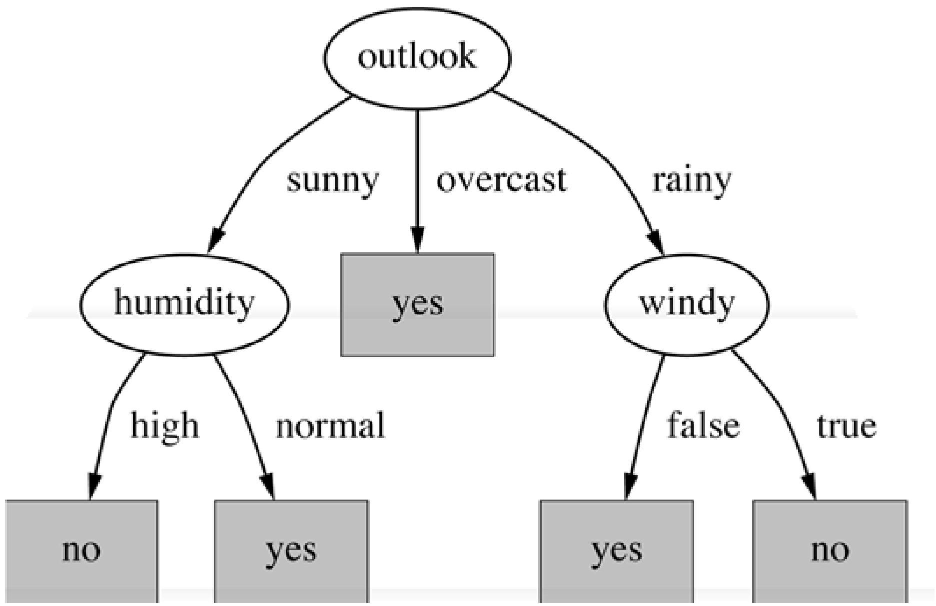
- Holdout method
  - Given data is randomly partitioned into two independent sets
    - Training set (e.g. 2/3) for model construction
    - Test set (e.g. 1/3) for accuracy estimation
- k-fold cross-validation (e.g. k = 10)
  - Randomly partition the data into k *mutually exclusive* subsets, each approximately equal size
  - At i-th iteration, use D_i as test set and others as training set
  - Repeat k times, each with different D_i for test set
  - Stratified cross-validation: folds are stratified so that class distribution in each fold is approximately the same as that in the entire data
- Leave-one-out test: k folds where k = # of tuples
  - Most stable but most inefficient

Decision Tree

Big Data

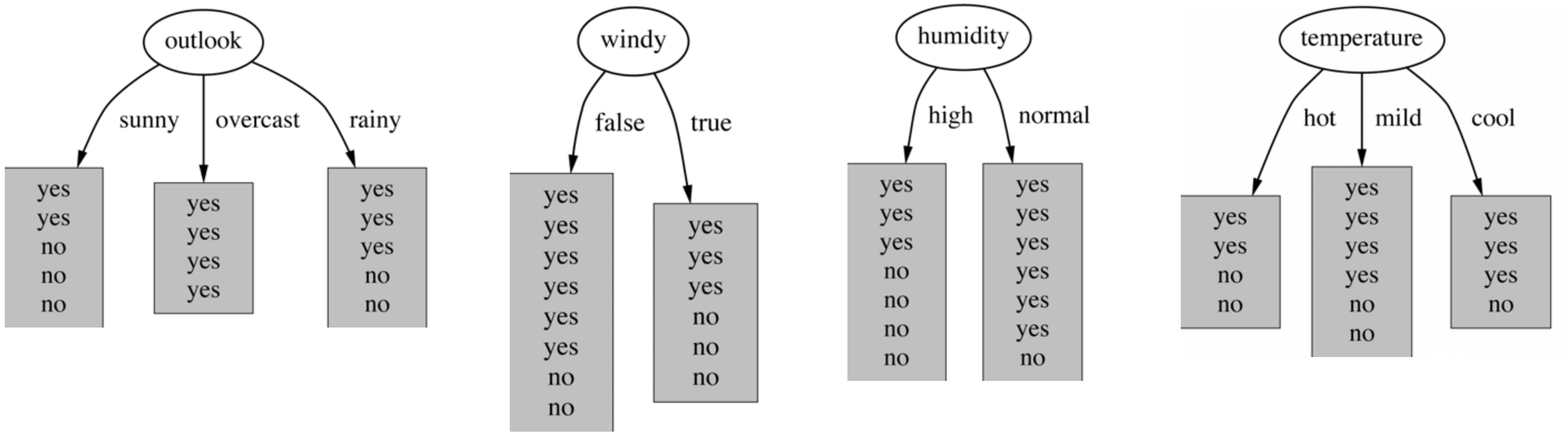
Decision trees

Outlook	Temp	Humidity	Wind	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No



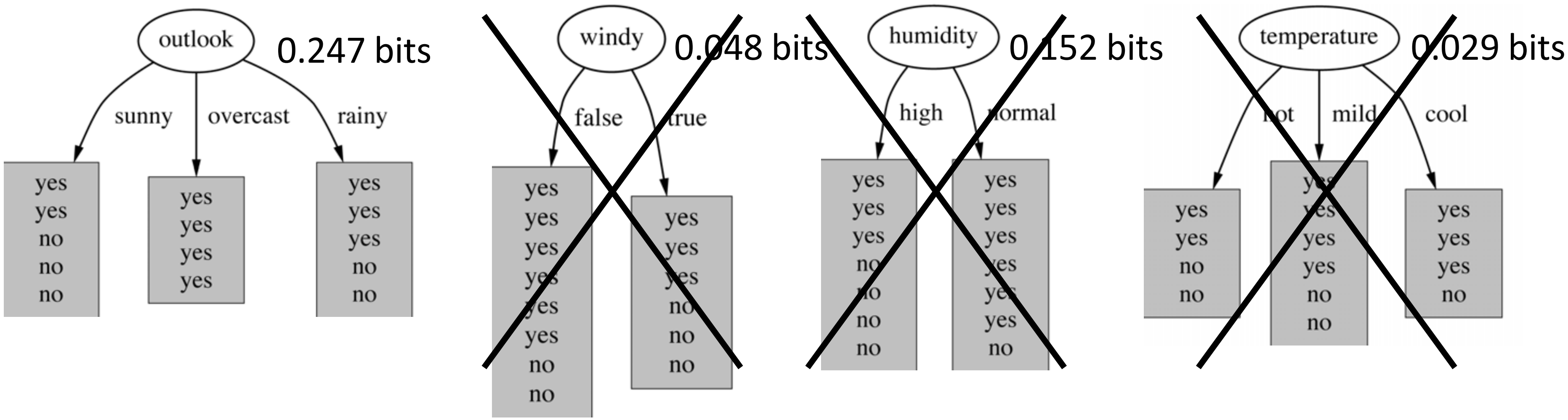
Decision trees

Which attribute to select? => Entropy difference => Info. gain



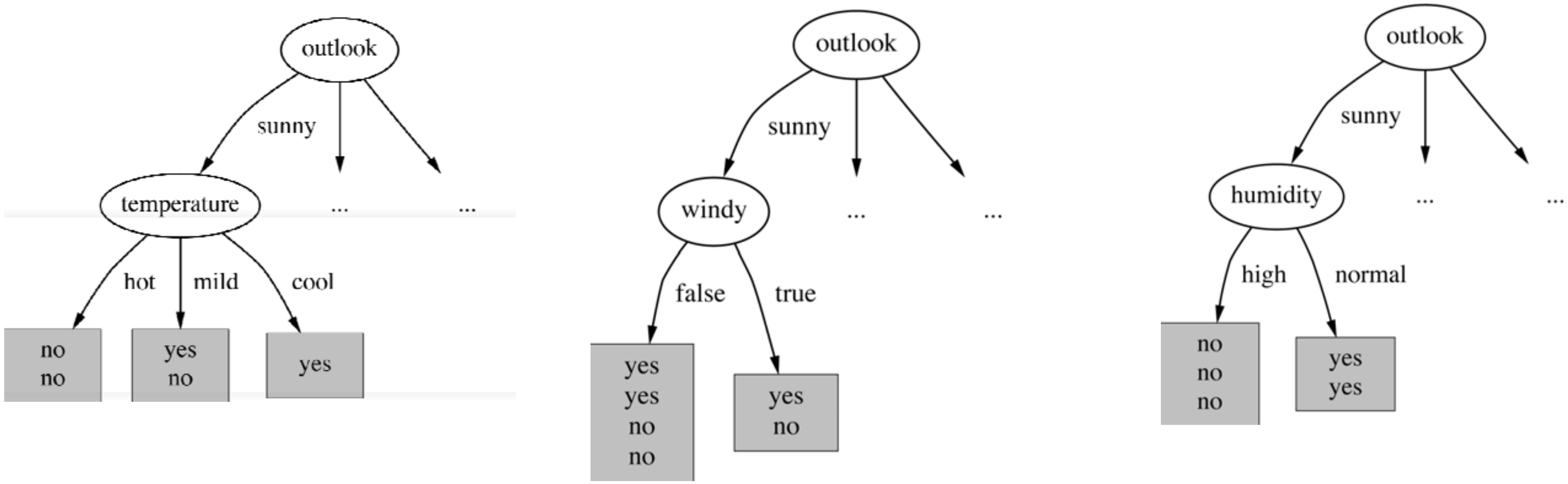
Decision trees

Which attribute to select? => Entropy difference => Info. gain



Decision trees

Continue to split …



gain(*temperature*) = 0.571 bits

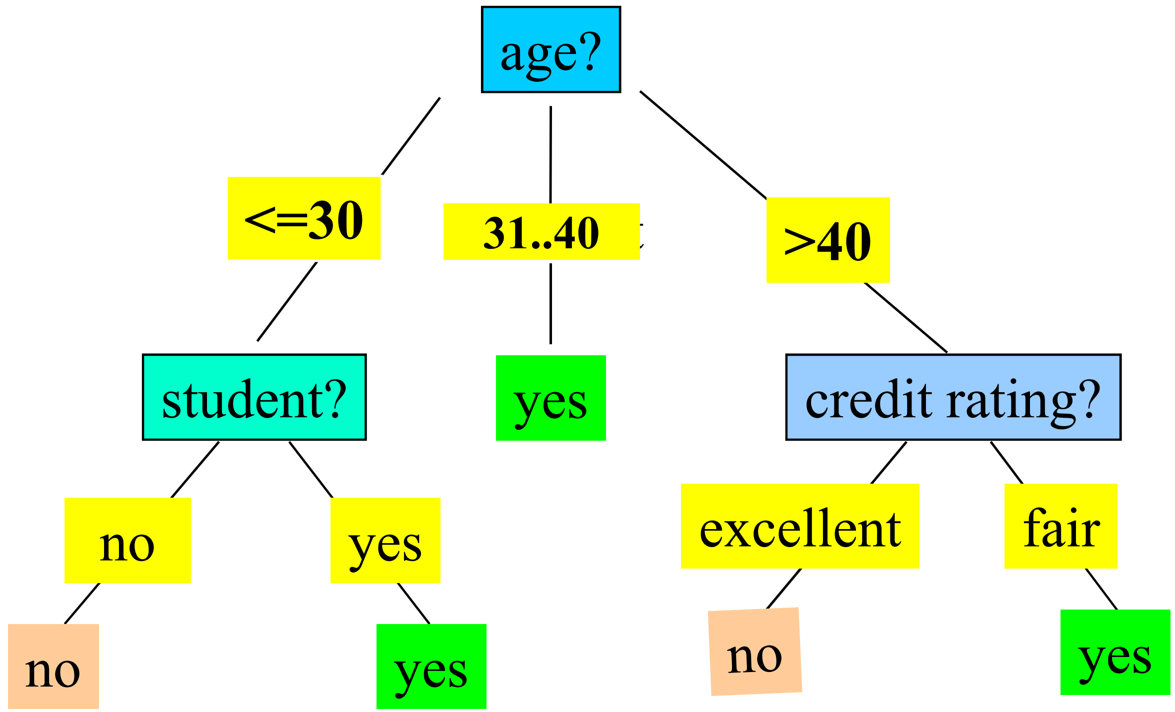
gain(*windy*) = 0.020 bits

gain(*humidity*) = 0.971 bits

Decision tree induction: Training dataset

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Output: A decision tree for “buys_computer”



Algorithm for decision tree induction

- Basic algorithm (a greedy algorithm)
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - At start, all the training examples are at the root
  - Attributes are categorical (if continuous-valued, they are discretized in advance)
  - Examples are partitioned recursively based on selected attributes
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g. information gain)
- Conditions for stopping partitioning
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf
  - There are no samples left

Attribute selection measure: Information gain (ID3/C4.5)

- Select the attribute with the highest information gain
- Let p_i be the probability that an arbitrary tuple in D belongs to class C_i, estimated by $|C_{i,D}|/|D|$
- Expected information (entropy) needed to classify a tuple in D:

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

- Information needed (after using A to split D into v partitions) to classify D:

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$$

- Information gained by branching on attribute A: $Gain(A) = Info(D) - Info_A(D)$

Attribute selection: Information gain

- Class P: buys_computer = "yes"
- Class N: buys_computer = "no"

$$Info(D) = I(9,5) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.940$$

$$Info_{age}(D) = \frac{5}{14}I(2,3) + \frac{4}{14}I(4,0) + \frac{5}{14}I(3,2)$$
$$= 0.694$$

age	p_i	n_i	$I(p_i, n_i)$
<=30	2	3	0.971
31…40	4	0	0
>40	3	2	0.971

$\frac{5}{14}I(2,3)$ means "age <=30" has 5 out of 14 samples, with 2 yes'es and 3 no's.

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Hence

$$Gain(age) = Info(D) - Info_{age}(D) = 0.246$$

Similarly,

$$Gain(income) = 0.029$$
$$Gain(student) = 0.151$$
$$Gain(credit\_rating) = 0.048$$

Gain ratio for attribute selection (C4.5)

- Information gain measure is biased towards attributes with a large number of values
- C4.5 (a successor of ID3) uses gain ratio to overcome the problem (normalization to information gain)

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times log_2(\frac{|D_j|}{|D|})$$

  - $\mathrm{GainRatio(A) = Gain(A)/SplitInfo(A)}$

$$SplitInfo_A(D) = -\frac{4}{14} \times log_2\left(\frac{4}{14}\right) - \frac{6}{14} \times log_2\left(\frac{6}{14}\right) - \frac{4}{14} \times log_2\left(\frac{4}{14}\right) = 0.926$$

- Example
  - gain_ratio(income) = 0.029/0.926 = 0.031
- The attribute with the maximum gain ratio is selected as the splitting attribute

Gini index (CART)

- If a data set D contains examples from n classes, gini index, *gini* (D) is defined as

$$gini(D) = 1 - \sum_{j=1}^{n} p_j^2$$

where p_j is the relative frequency of class j in D

- If a data set D is split on A into two subsets D_1 and D_2, the gini index *gini* (D) is defined as

$$gini_A(D) = \frac{|D_1|}{|D|} gini(D_1) + \frac{|D_2|}{|D|} gini(D_2)$$

- Reduction in Impurity:

$$\Delta gini(A) = gini(D) - gini_A(D)$$

- The attribute provides the smallest $gini_{split}$(D) (or the largest reduction in impurity) is chosen to split the node (need to enumerate all the possible splitting points for each attribute)

Comparing attribute selection measures

- The three measures, in general, return good results but
- Information gain:
  - biased towards multivalued attributes
- Gain ratio:
  - tends to prefer unbalanced splits in which one partition is much smaller than the others
- Gini index:
  - biased to multivalued attributes
  - has difficulty when # of classes is large
  - tends to favor tests that result in equal-sized partitions and purity in both partitions

Other attribute selection measures

- CHAID: a popular decision tree algorithm, measure based on χ^2 test for independence
- C-SEP: performs better than info. gain and gini index in certain cases
- G-statistics: has a close approximation to χ^2 distribution
- MDL (Minimal Description Length) principle (i.e. the simplest solution is preferred):
  - The best tree as the one that requires the fewest # of bits to encode the tree
- Which attribute selection measure is the best?
  - Most give good results, <u>none is significantly superior than others</u>

Overfitting and tree pruning

- Overfitting: An induced tree may overfit the training data
  - Too many branches, some may reflect anomalies due to noise or outliers
  - Poor accuracy for unseen samples => poor generalization
- Two approaches to avoid overfitting
  - Prepruning:
    - Halt tree construction early—do not split a node if this would result in the goodness measure falling below a threshold
    - Difficult to choose an appropriate threshold
  - Postpruning:
    - Remove branches from a "fully grown" tree—get a sequence of progressively pruned trees
    - Use a set of data different from the training data to decide which is the "best pruned tree"

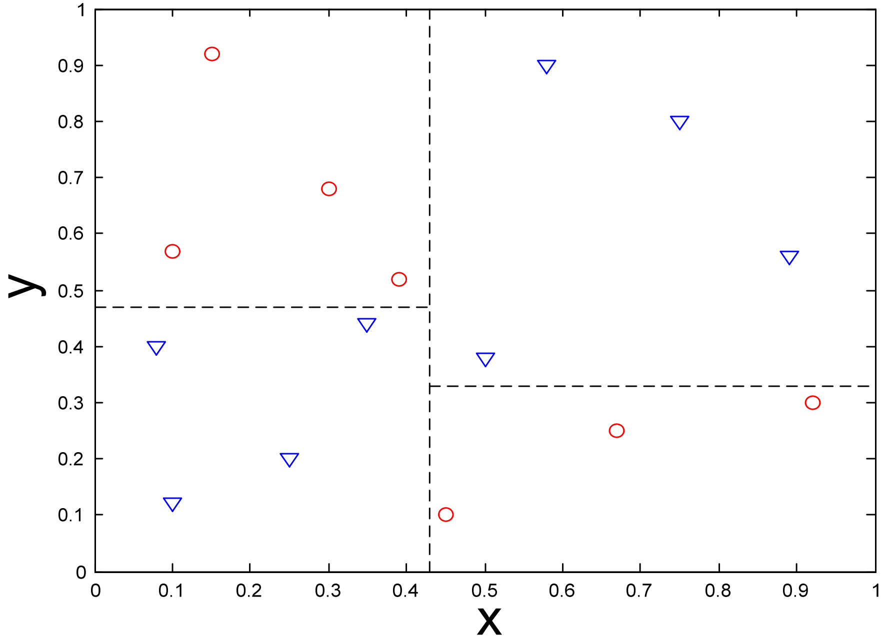
Postpruning

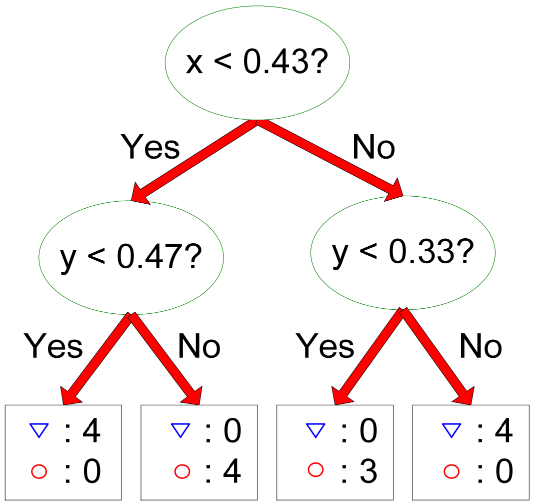
- Postpruning:
  - Remove branches from a “fully grown” tree—get a sequence of progressively pruned trees
  - Subtree Replacement: Merge from the leaves to the root
  - Subtree Raising: Raise the subtree of the most popular branch to replace its parent
- Two postpruning methods
  - Reduced-error pruning:
    - Use a set of data different from the training data to decide which is the “best pruned tree”
    - Pros: more accurate estimate than C4.5 pruning.
    - Cons: growing tree based on less training data
  - C4.5 pruning:
    - Use some estimate of error based on the training data itself

Decision tree based classification

- Advantages:
  - Fast learning
  - Fast prediction
  - Easy to interpret the model
  - Comparable accuracy
- Disadvantages:
  - Attributes must be discretized (information loss)
  - Usually not as accurate as other advanced classification methods such as SVM, boosting, etc.

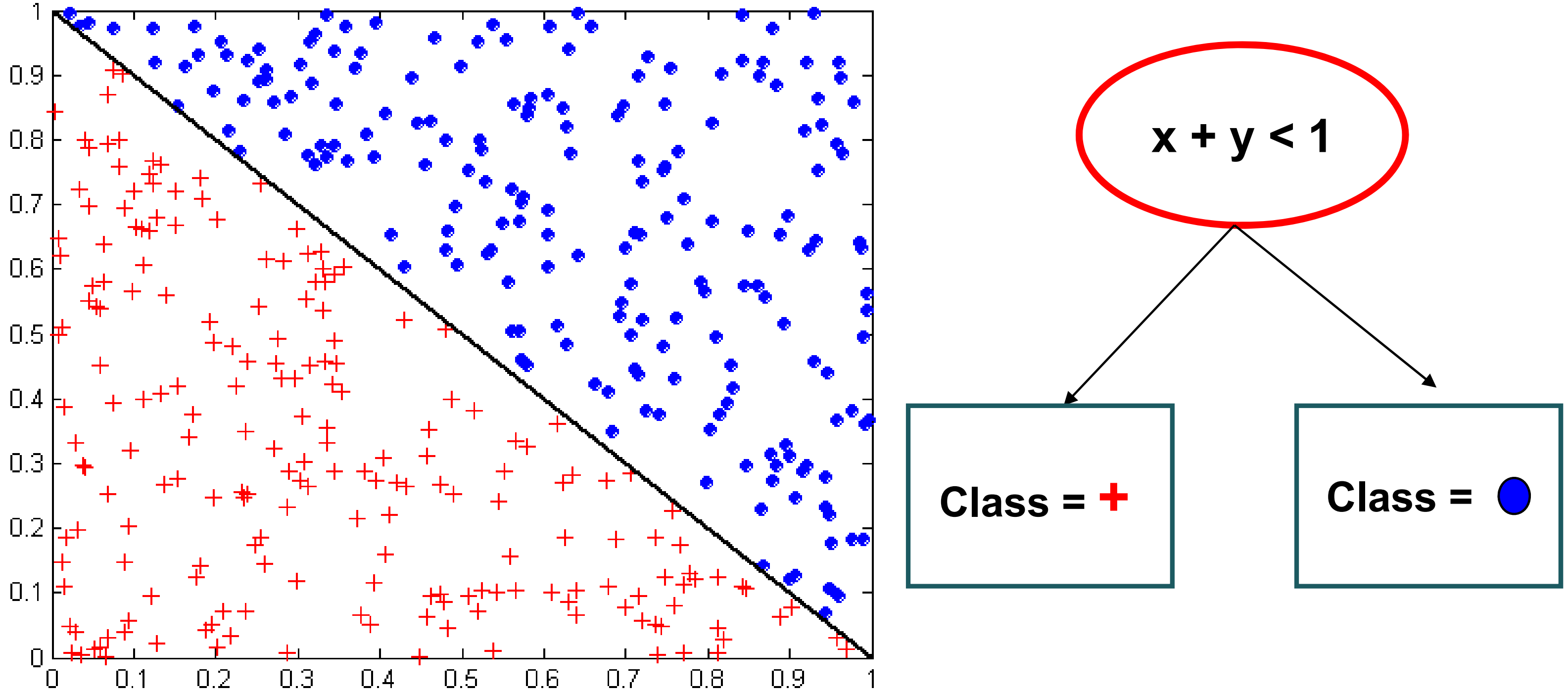
Decision boundary





- Border line between two neighboring regions of different classes is known as decision boundary
- Decision boundary is parallel to axes because test condition involves a single attribute at-a-time

Oblique decision trees



- Test condition may involve multiple attributes => More expressive representation
- Finding optimal test condition is computationally expensive