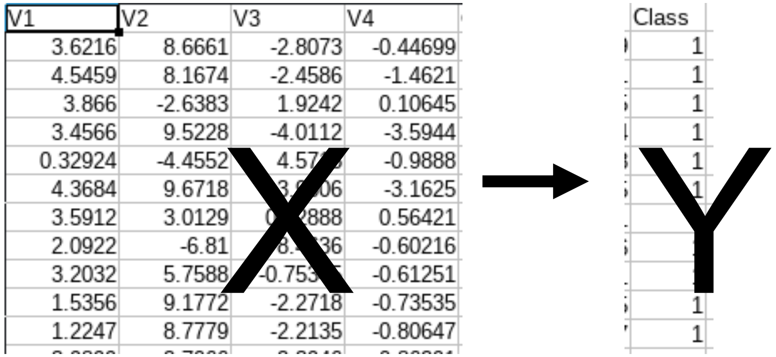
Decision Tree classifier

e.g.

Consider V2, is it > 5.1?

  if yes: Consider V1, is it < 2.4?

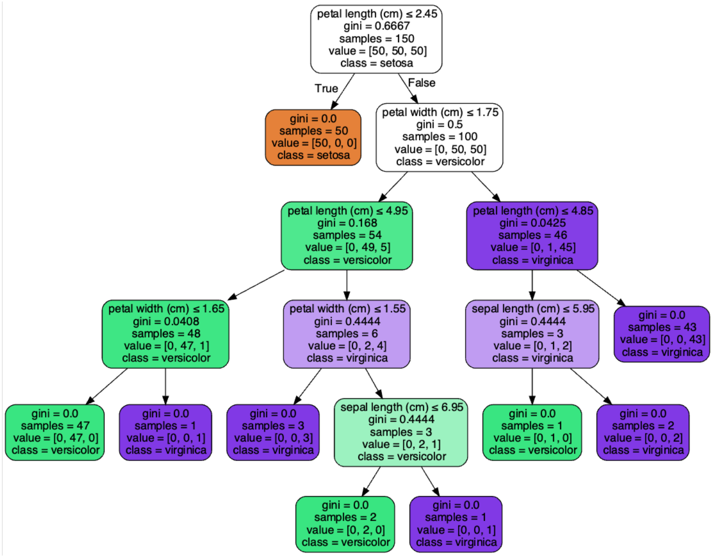
  if no: Consider V4, is it > 0.3?



And recurse… → tree of decisions.

At the leaves of this tree,

arrive at a classification



We can learn the tree, namely:

- which variables to consider, in order
- the thresholds

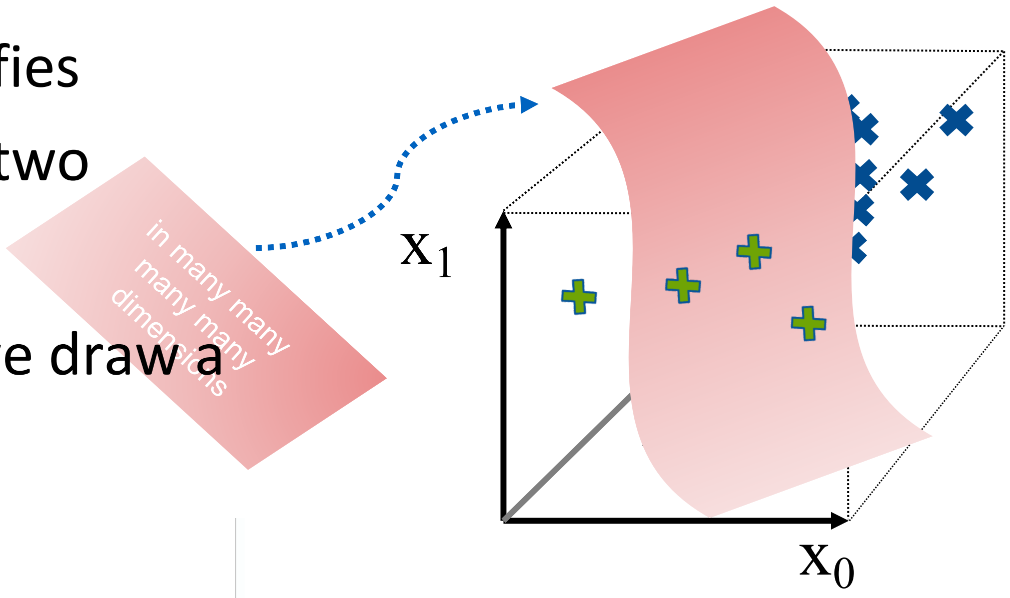
complexity control: scikit-learn suggests the max_depth

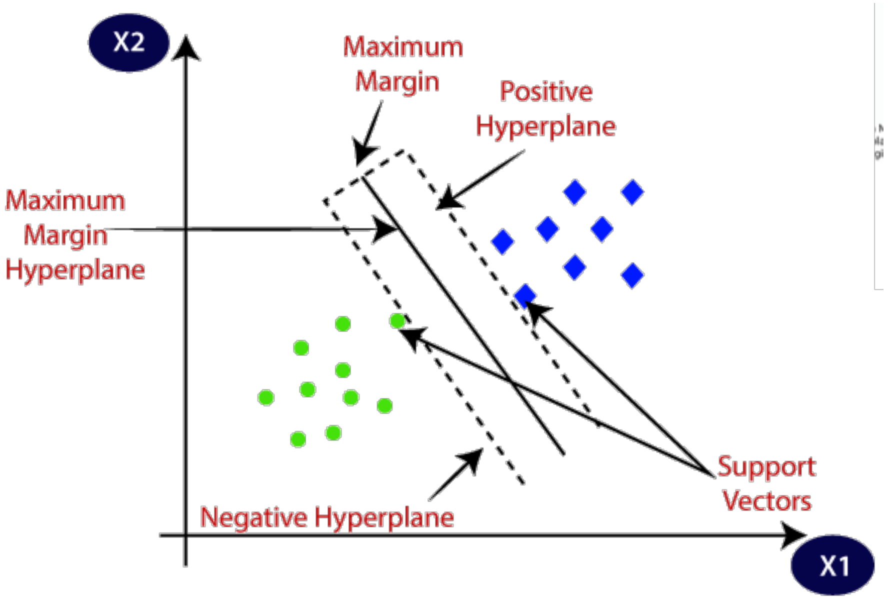
https://docs.daft-pgm.org

Support Vector Machines / Kernel machines

Two ways to think about SVMs:

- like k-NN, we have a function ("kernel") that quantifies "closeness" between two example X's
- like the Perceptron, we draw a hyperplane in a space

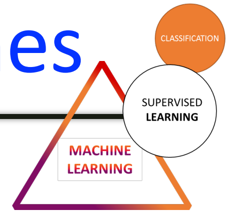




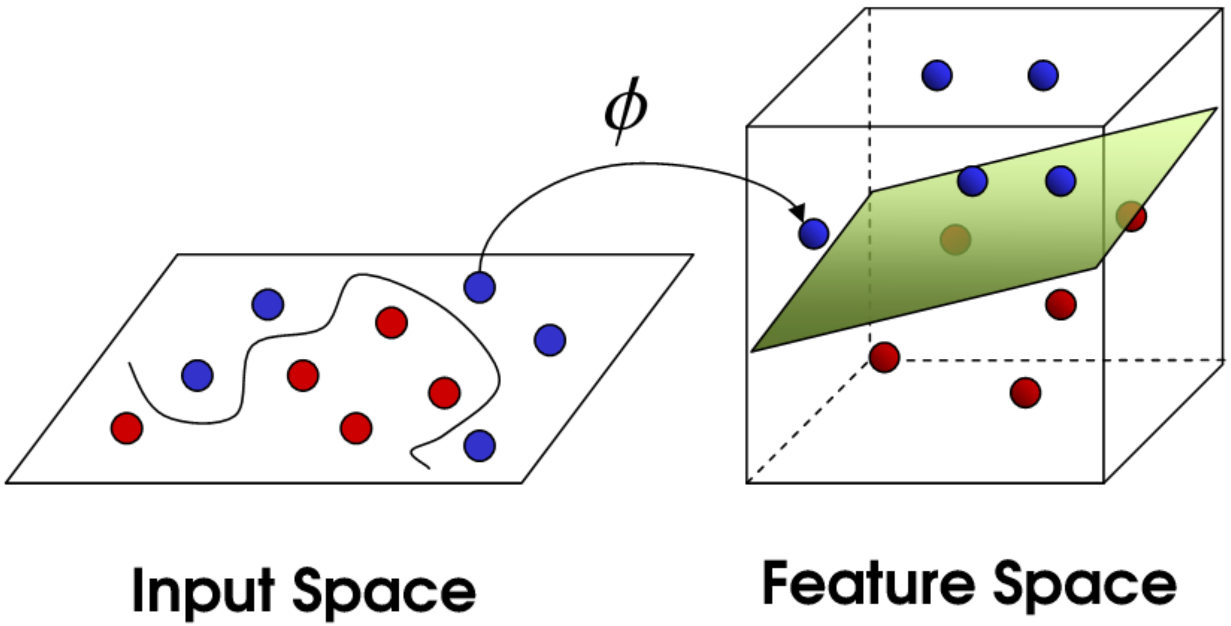
But using math trickery, this very-high-dim "space" is *never explicitly realised*, and the hyperplane is never *explicitly* computed!

https://static.javatpoint.com/tutorial/machine-learning/images/support-vector-machine-algorithm.png

Support Vector Machines / Kernel machines



- Very successful
- Largely robust to choice of kernel function
- Kernel trick - decision boundary can be *linear* in high-dimensional space (nice math properties), & *non-linear* in the input space



Ensembles of classifiers

Several of the most popular and successful classifiers are based on *ensembles* of simple classifiers.

For example:

- Boosting (e.g. GradBoost)
- Random Forests (e.g. of trees / stumps)

- take hundreds of "base" classifiers which might each be very weak (e.g. "decision stumps", i.e. one-level decision trees)
- pull these simple predictions together to obtain an ensemble prediction.

complexity control: as with Decision Trees – the max_depth

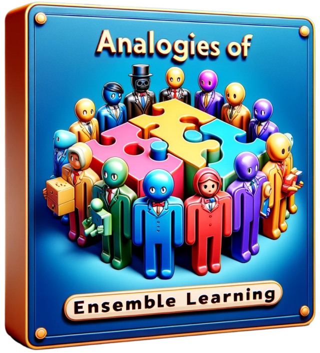
Ensemble Learning

- Sometimes each learning technique yields a different hypothesis
- But no perfect hypothesis…
- Could we combine several imperfect hypotheses into a better hypothesis?

Analogies of Ensemble Learning

- Analogies:
  - Elections combine voters' choices to pick a good candidate
  - Committees combine experts' opinions to make better decisions

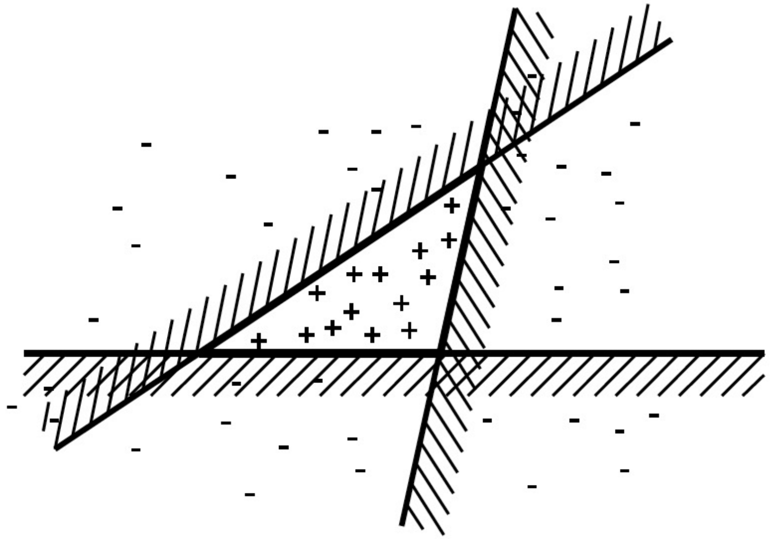
- Intuitions:
  - Individuals often make mistakes, but the "majority" is less likely to make mistakes.
  - Individuals often have partial knowledge, but a committee can pool expertise to make better decisions.



Definition of Ensemble Learning

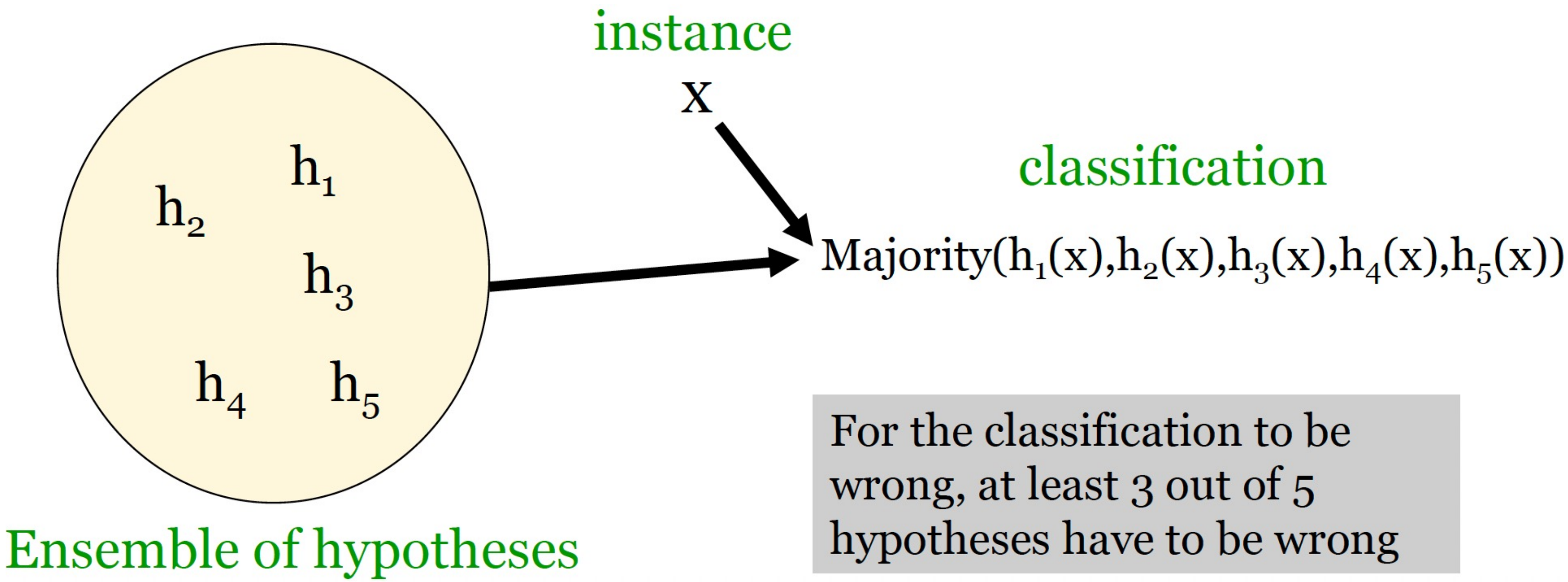
- Definition: method to select and combine an ensemble of classifiers into a (hopefully) better classifier

- Can enlarge classification capability:
  - Perceptrons, logistic regression, support vector machines:
    - linear separators
  - Ensemble of linear seperators:
    - polytope



Bagging

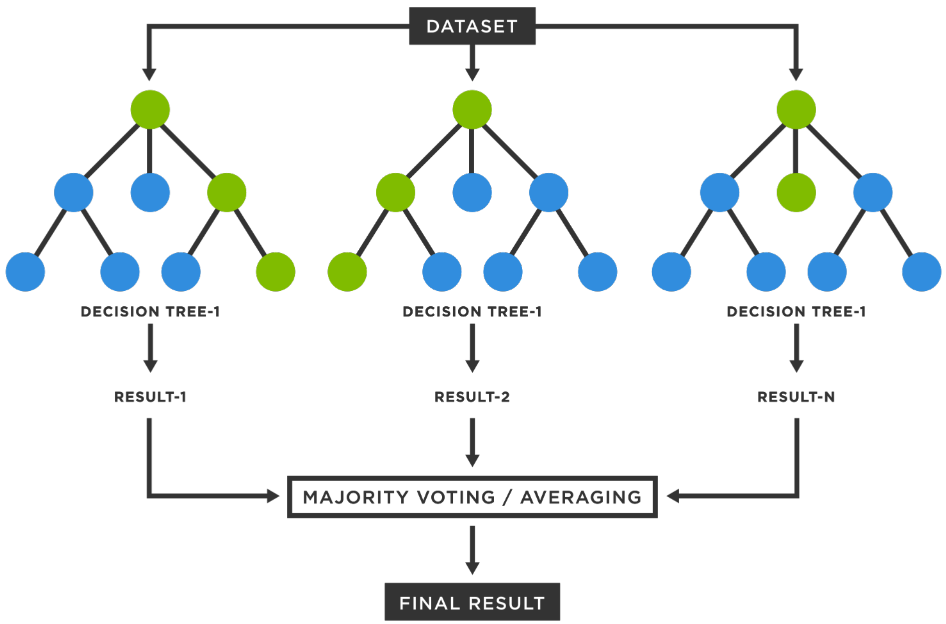
- Majority voting



For the classification to be wrong, at least 3 out of 5 hypotheses have to be wrong

Random Forests

- Each tree is built from a bootstrap sample (i.e. a sample drawn with replacement) from the training set
- When splitting each node during the construction of a tree, the best split is found either from all input features or a random subset of features

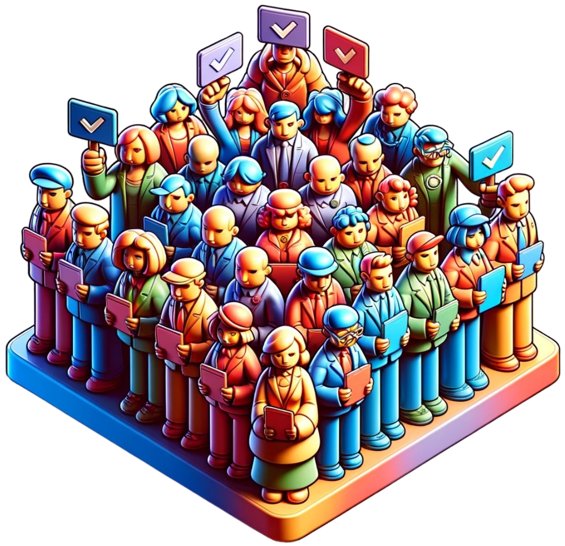


https://www.tibco.com/reference-center/what-is-a-random-forest

Weighted Majority

- In practice
  - Hypotheses are rarely independent
  - Some hypotheses have less errors than others

- Let's take a weighted majority
- Intuition:
  - Decrease weight of bad/correlated classifiers in the ensemble
  - Increase weight of good classifiers in the ensemble



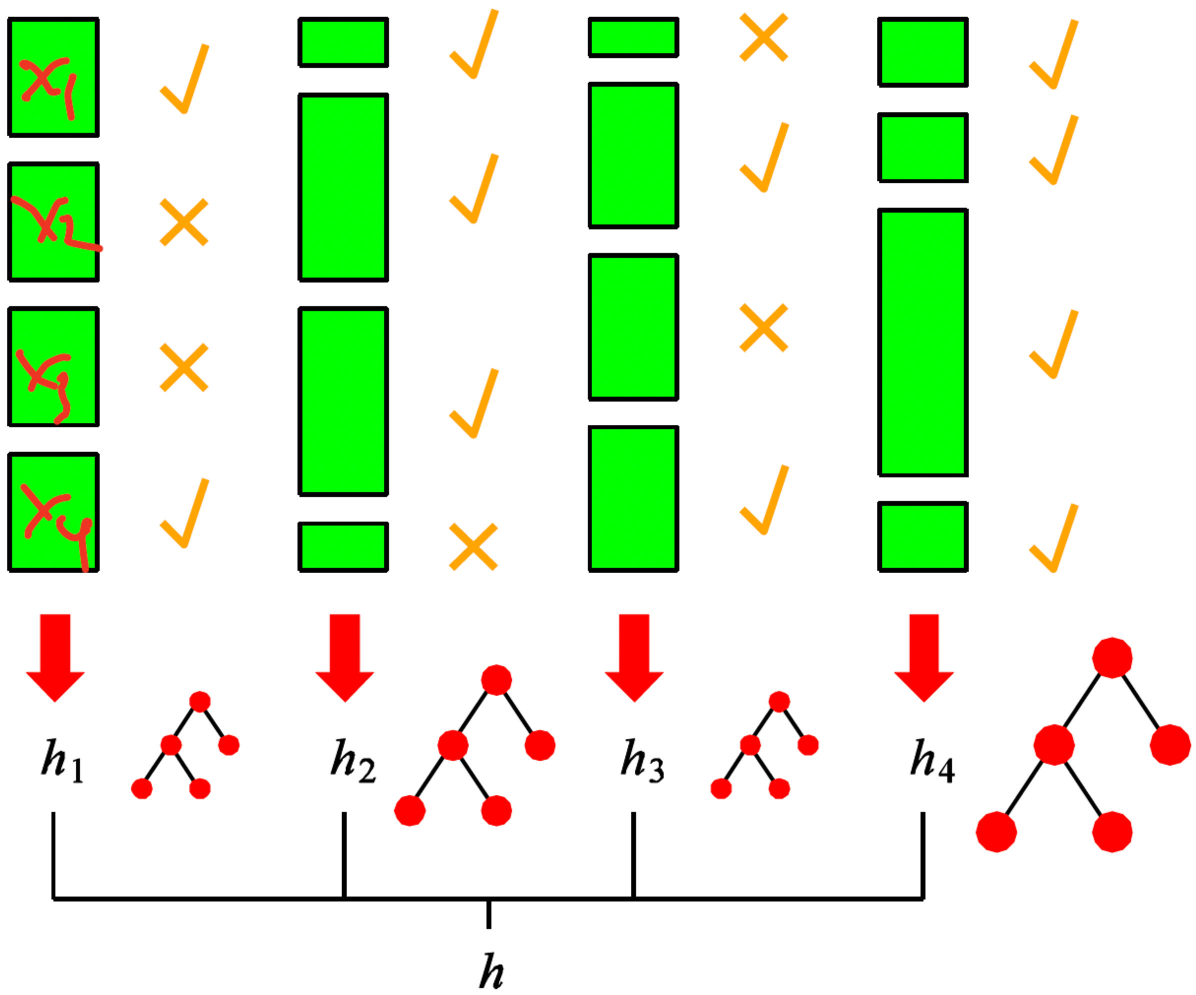
Boosting

- Very popular ensemble technique
- Computes a weighted majority
- Can “boost” a “weak learner”
- Operates on a weighted training set

Boosting Framework

- Set all instance weights w_j to 1
- Repeat
  - $h_i \leftarrow learn(dataset, instance\ weights)$
  - Increase weight w_j of misclassified instances x_j
- Until sufficient number of hypotheses
- Ensemble hypothesis is the weighted majority of h_i's with weights c_i proportional to the accuracy of h_i

Example of the Boosting Framework



Basic Performance Evaluation Metrics

Classification Accuracy

- Accuracy – the number of objects which are correctly detected/classified as a percentage of the total number of desired objects in the data set.

$$Accuracy = \frac{N_{CorrClassified}}{N_{total}} * 100\%$$

- $N_{CorrClassified}$: the number of objects correctly detected or classified
- N_{total}: the number of desired objects in the data set.

- Error Rate: the number of objects incorrectly classified as a percentage of the number of objects.

- Question: how to know the relative frequencies of false positive and the false negative errors?

TPR, TNR, FPR and FNR

		Actual Situation	
		Disease	Non-Disease
Diagnosed Situation	Disease	True *Positive* (**TP**)	False Positive (**FP**)
	Non-Disease	False Negative (**FN**)	True Negative (**TN**)

- TPR – True Positive Rate (Fraction) – **sensitivity**
  - The fraction of desired objects (actual Disease) in a database that are correctly classified/detected by a classifier/detector.
  - TPR = *TP/(TP+FN)*
- TNR – True Negative Rate (Fraction) – **specificity**
  - The fraction of non-objects (actual non-Disease) in a database that are correctly classified/detected as non-objects/background.
  - TNR = *TN/(FP+TN)*

TPR, TNR, FPR and FNR

- FPR – False Positive Rate (Fraction)
  - The fraction of non-objects in a database that are incorrectly classified/detected as objects.
  - FPR = *FP/(FP+TN)*

- FNR – False Negative Rate (Fraction)
  - The fraction of objects in a database that are incorrectly classified/detected as non-objects (background).
  - FNR = *FN/(TP+FN)*

True Class

Predicted Class	A	B	C
A	TP_A	E_{BA}	E_{CA}
B	E_{AB}	TP_B	E_{CB}
C	E_{AC}	E_{BC}	TP_C

TPR, TNR, FPR and FNR

- FNR + TPR = 1
- FPR + TNR = 1
- The bigger the TPR and TNR, the better the classifier.
- Because of the interrelationships among these measures, it is necessary only to indicate a single pair, either TPR and TNR or FNR and FPR are employed.

- TPR, TNR, FPR and FNR

		Actual Situation	
		Disease	Non-Disease
Diagnosed Situation	Disease	True *Positive* Rate (**TPR**)	False Positive Rate (**FPR**)
	Non-Disease	False Negative Rate (**FNR**)	True Negative Rate (**TNR**)

Example Question

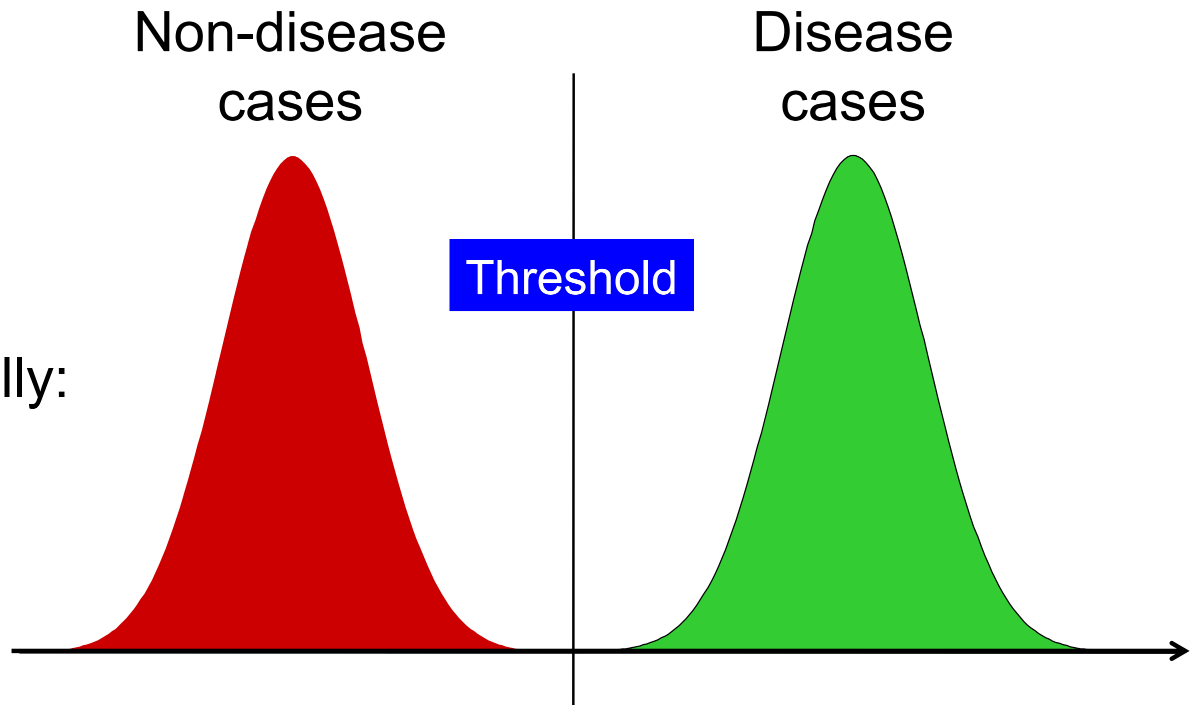
- Suppose a classifier is applied to a two-class object classification problem: class1 and class2.
- 200 objects for class1 and 300 objects for class2
- With some threshold, the classifier correctly classified 160 objects for class1 and 210 objects for class2 .
  - What is the accuracy?
  - What is the error rate?
  - What is the TPR?
  - What is the FPR?
  - What is the TNR?
  - What is the FNR?

Example Question

- Suppose a classifier is applied to a two-class object classification problem: class1 and class2.
- 200 objects for class1 and 300 objects for class2
- With some threshold, the classifier correctly classified 160 objects for class1 and 210 objects for class2 .
  - What is the accuracy? (160+210)/500 = 74%
  - What is the error rate? 26%
  - What is the TPR? 160/200=80%
  - What is the FPR? (300-210)/300 = 30%
  - What is the TNR? 70%
  - What is the FNR? 20%

Receiver Operating Characteristic (ROC) Curve

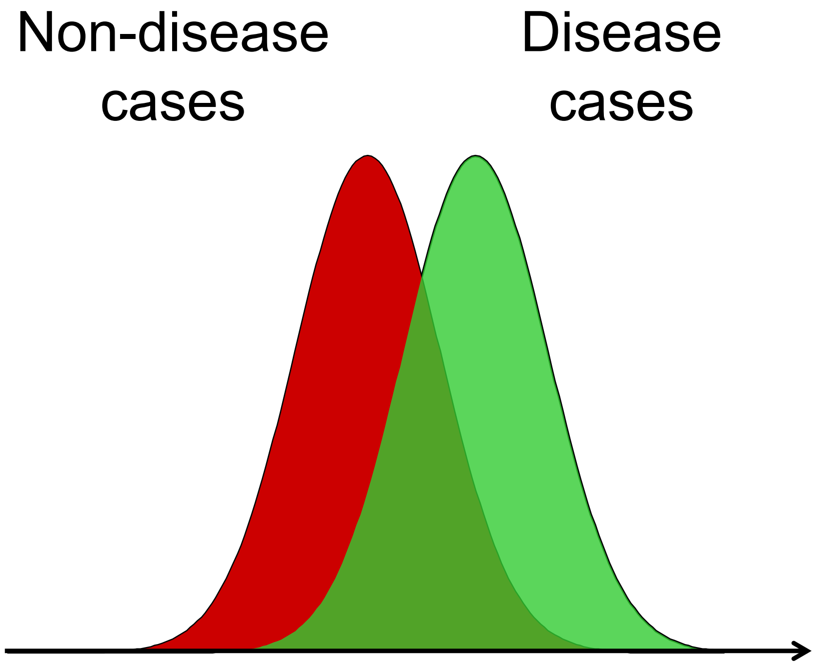
Ideally:



Test result value
or
subjective judgement of likelihood that case has disease

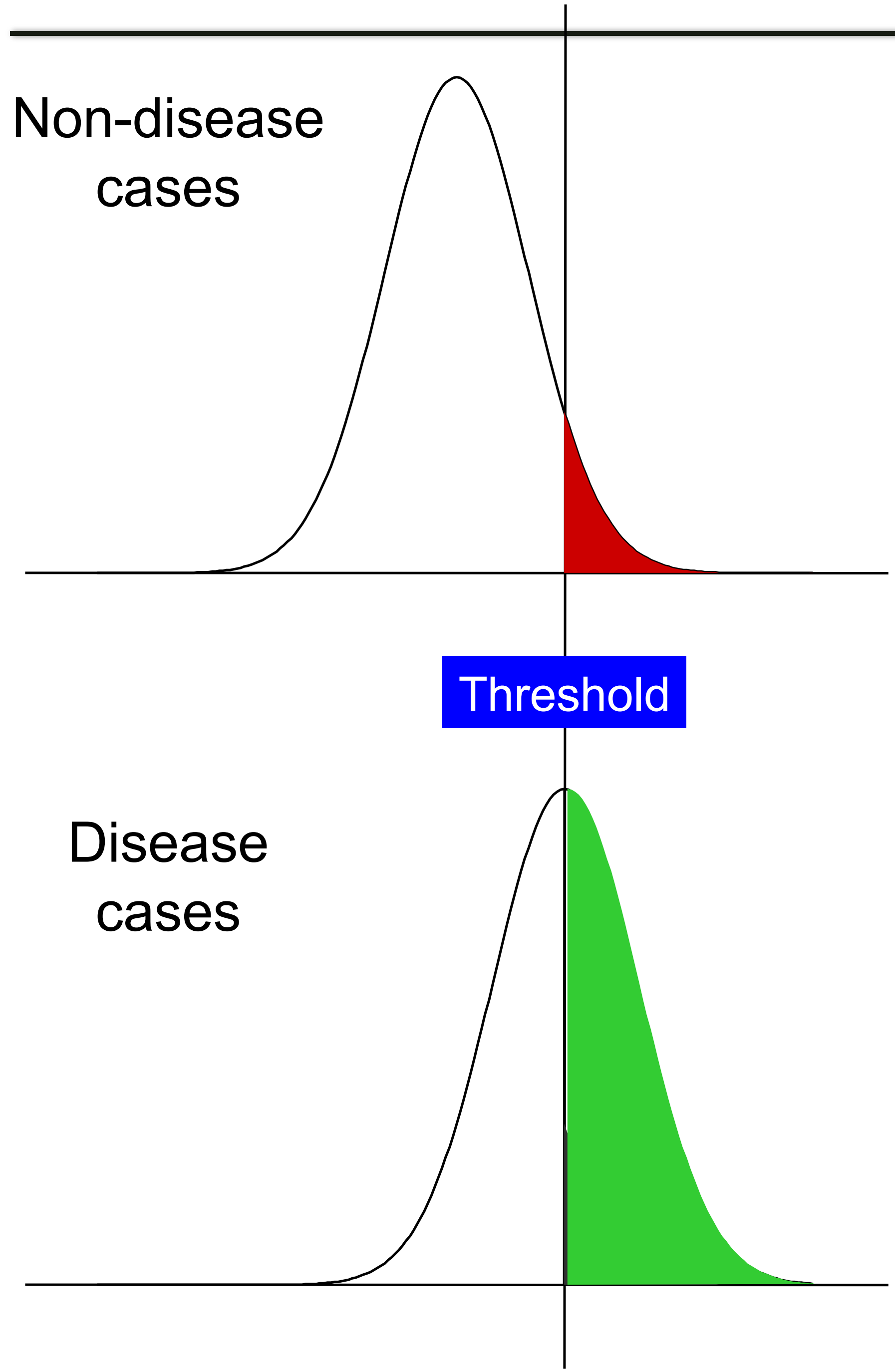
ROC Curve

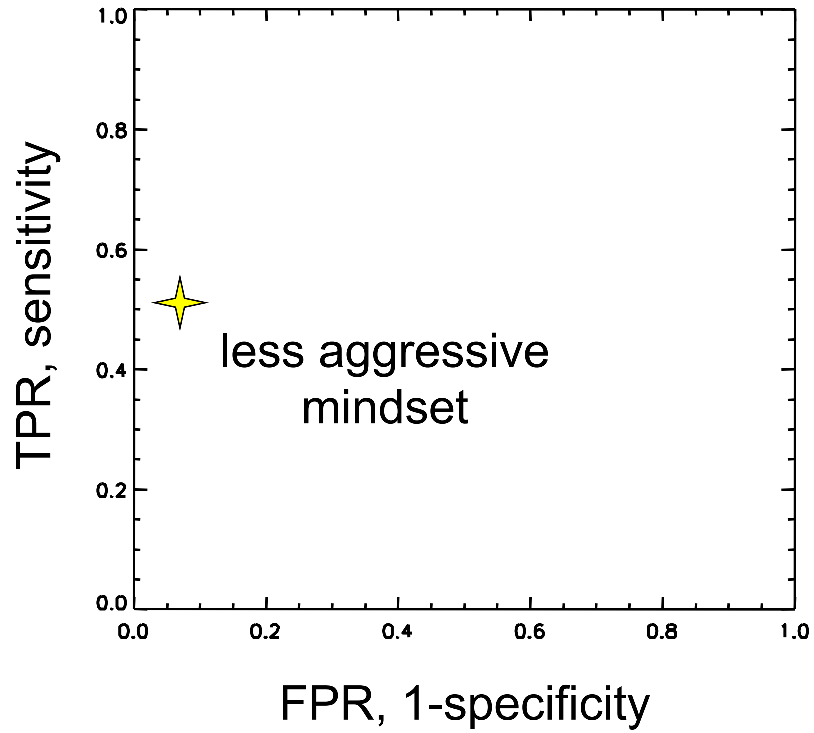
more typically:



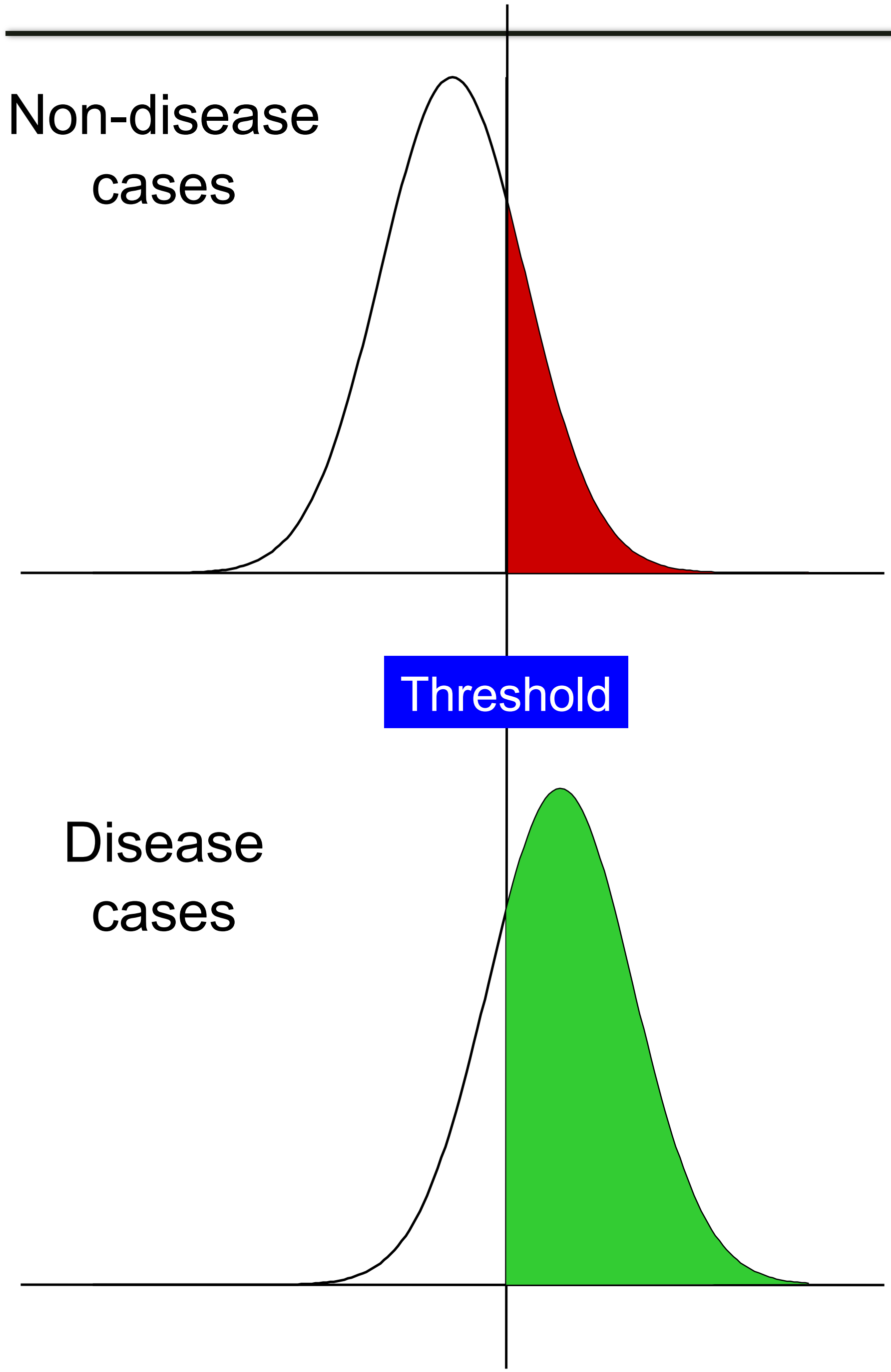
Test result value
or
subjective judgement of likelihood that case has disease

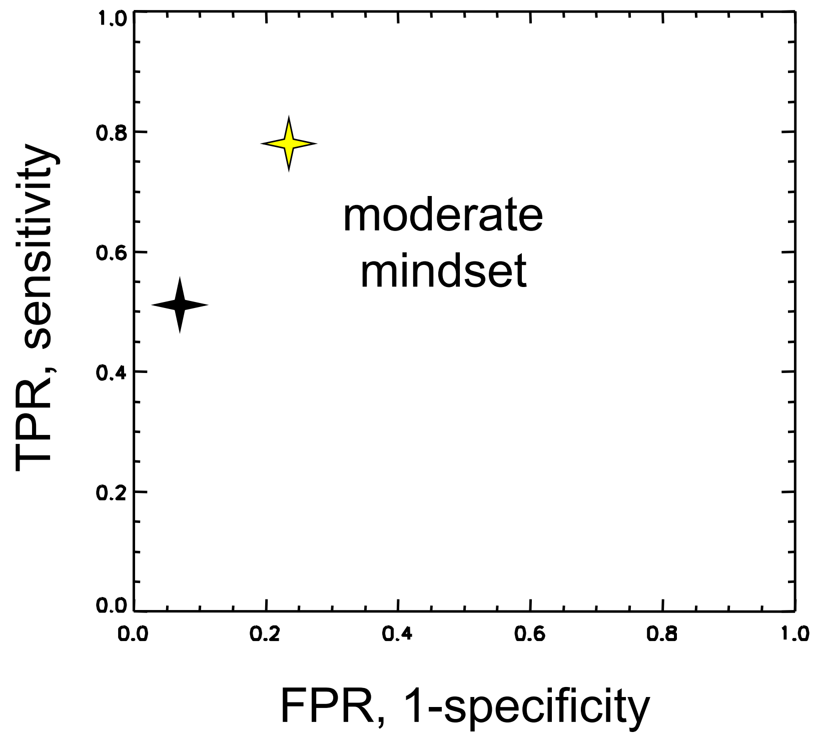
ROC Curve


Non-disease cases
Threshold
Disease cases

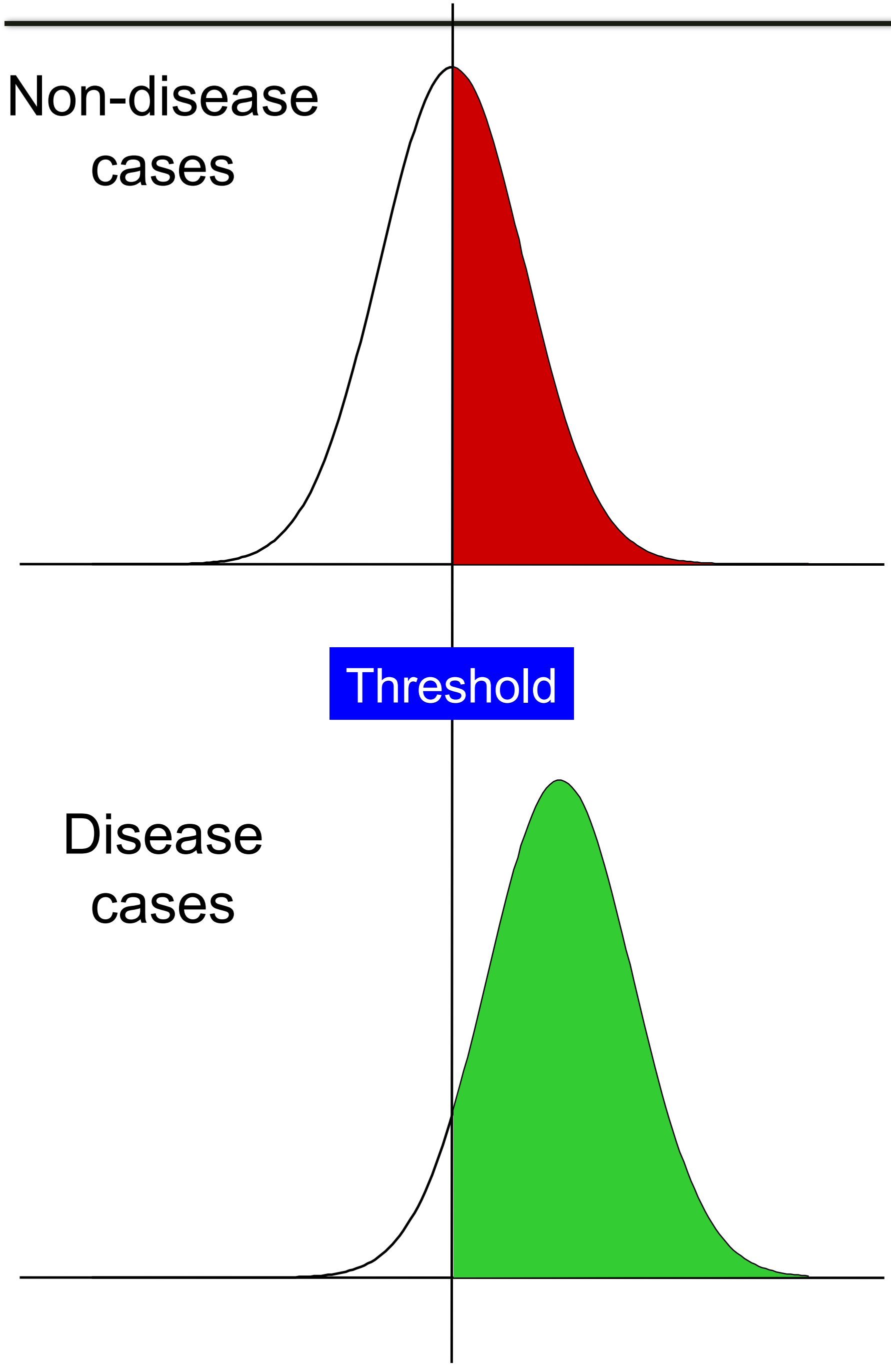

less aggressive mindset
TPR, sensitivity
FPR, 1-specificity
0.0
0.2
0.4
0.6
0.8
1.0

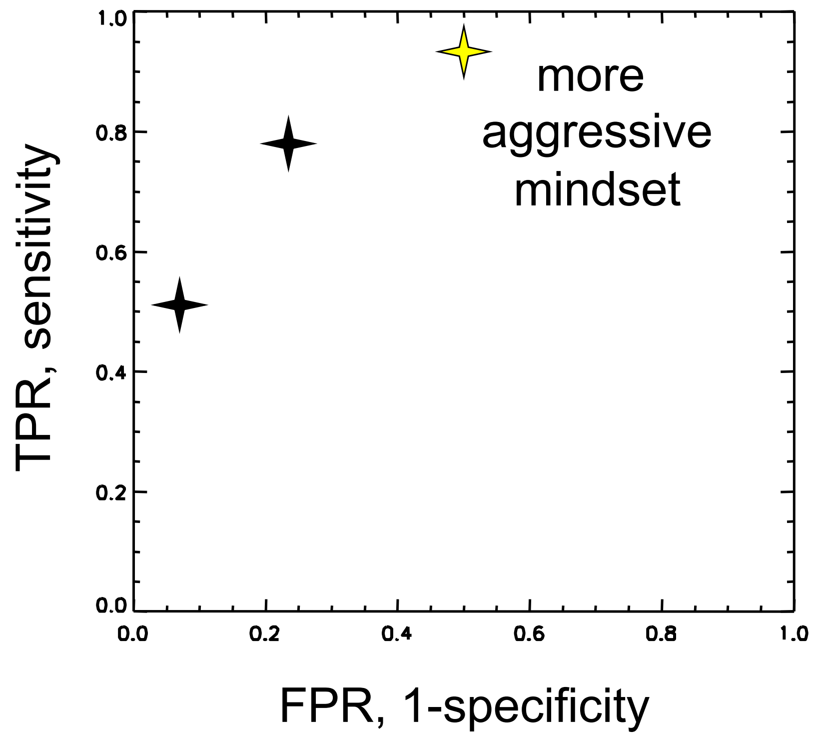
ROC Curve


Non-disease
cases
Threshold
Disease
cases

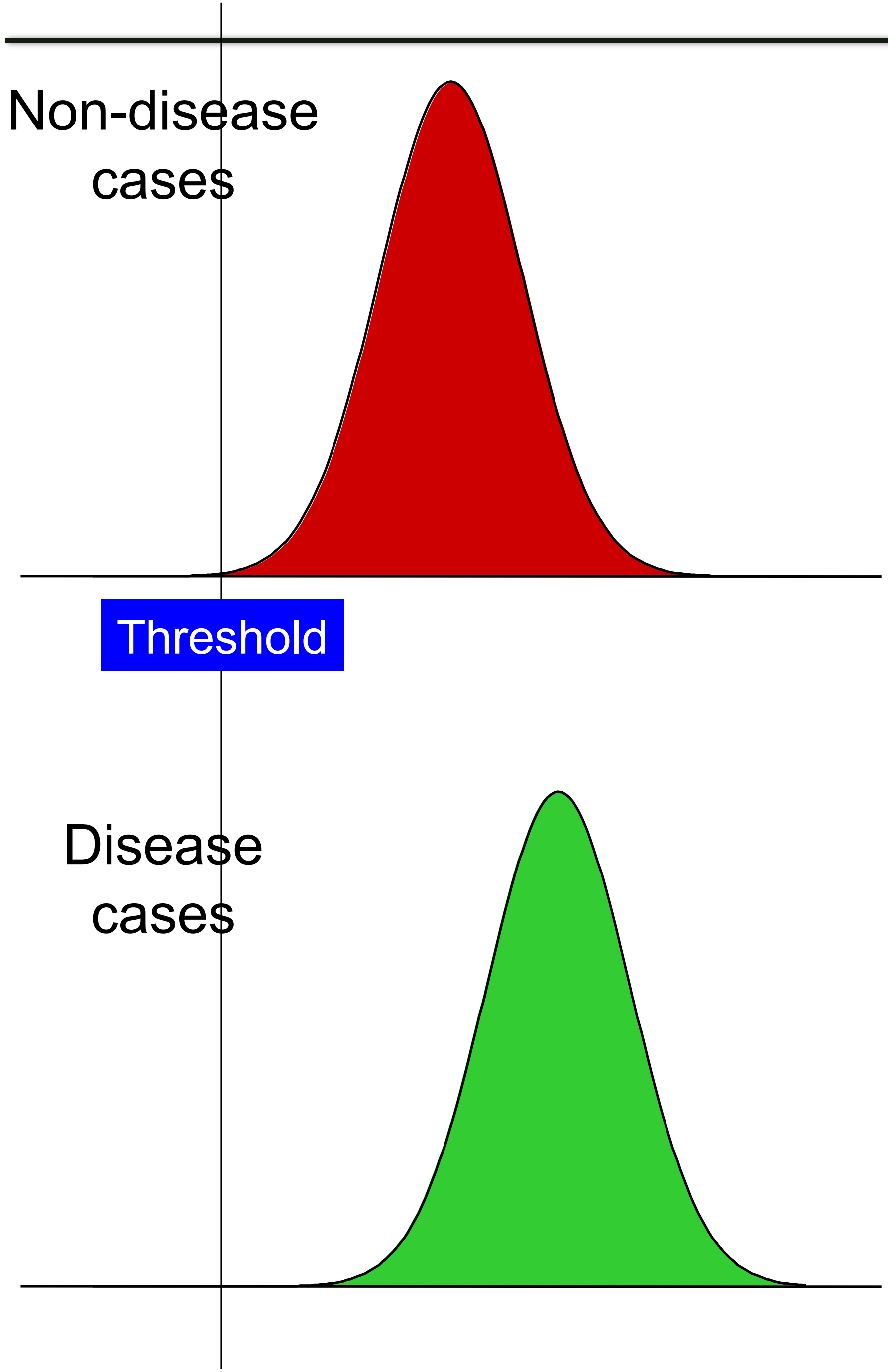

TPR, sensitivity
FPR, 1-specificity
moderate
mindset
0.0
0.2
0.4
0.6
0.8
1.0

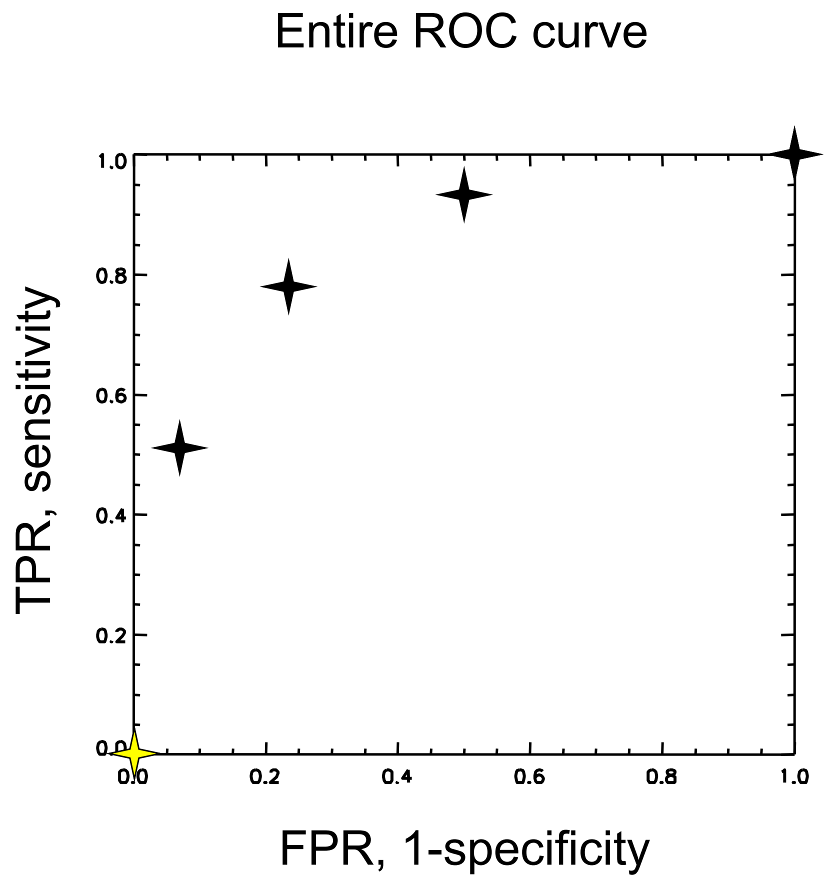
ROC Curve


Non-disease
cases
Threshold
Disease
cases


1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
TPR, sensitivity
FPR, 1-specificity
more
aggressive
mindset

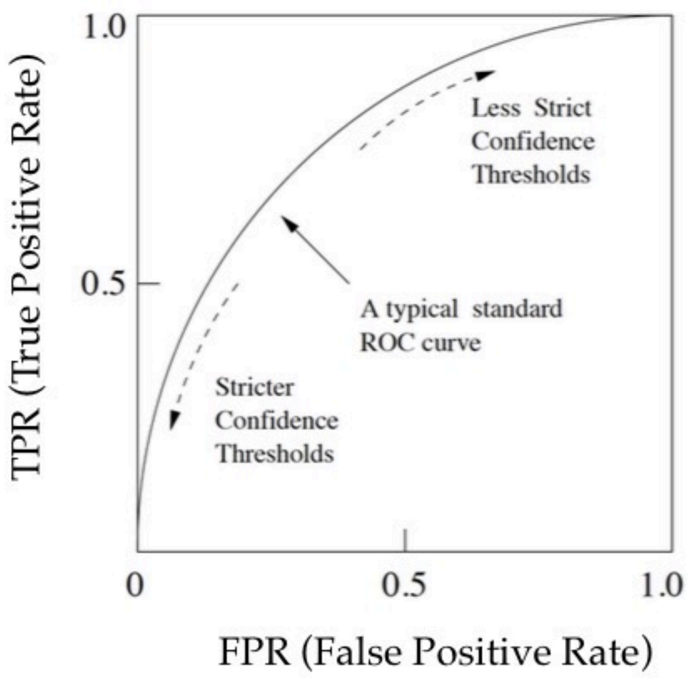
ROC Curve


Non-disease
cases
Threshold
Disease
cases

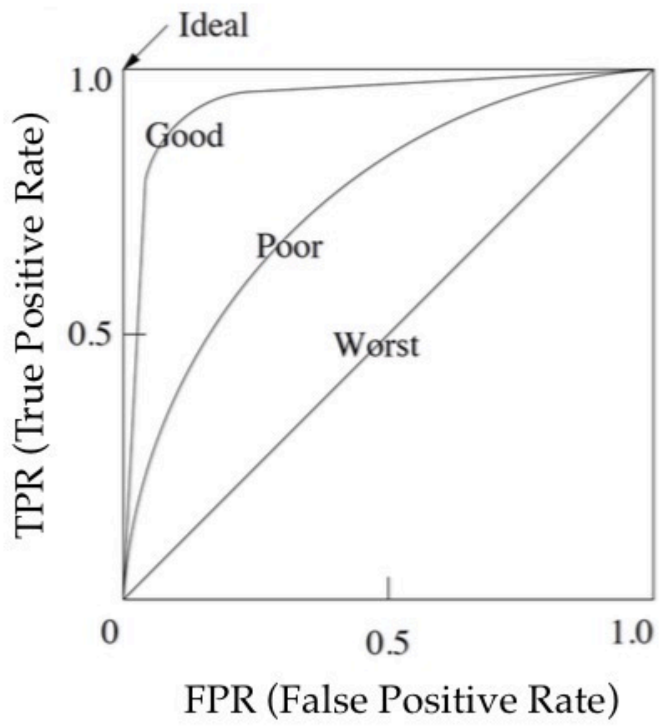

Entire ROC curve
TPR, sensitivity
FPR, 1-specificity
0.0
0.2
0.4
0.6
0.8
1.0

ROC Curve

- **ROC** curve: Receiver Operating Characteristic curve or Relative Operating Characteristic curve
- Standard ROC curves conventionally take the FPR as the x axis, and the TPR as the y axis.
- A typical standard ROC curve:



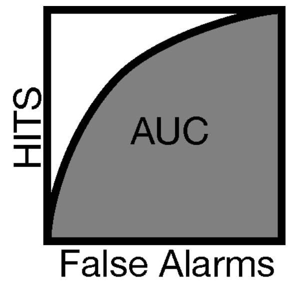
(a)



(b)

ROC Curve

- Different confidence thresholds correspond to different points, which represent different pairs of TPR and FPR.
- A higher ROC curve indicates greater discrimination capacity
  - (the “good” case in the figure).
- A lower ROC curve indicates weaker classification capacity
  - (the “poor” case in the figure).
- The worst classification or diagnostic system
  - Usually, which has no discrimination between positives and negatives (the “worst case” in the figure).
- The ideal system represents perfect interpretation, and the ideal point is TPR = 1.0 and FPR = 0.
  - (the “ideal” case in the figure).

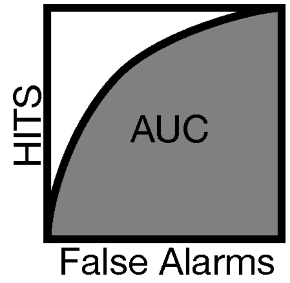


AUC

Conversion of ROC curve to a single measure is necessary sometimes.

One commonly used measure is:

- *Area under the ROC curve* (*AUC*): the area that lies beneath the entire ROC curve.
  - 0.5 < AUC <1.0
  - AUC = 0.5 corresponds to the worst case
  - AUC = 1.0 corresponds to the ideal case
  - Difficult to calculate/estimate, especially when the points are not well spread across the ROC space, depends a lot on the non-interesting part of the ROC curve (both TPR and FPR tend to 1.0).



Example Question

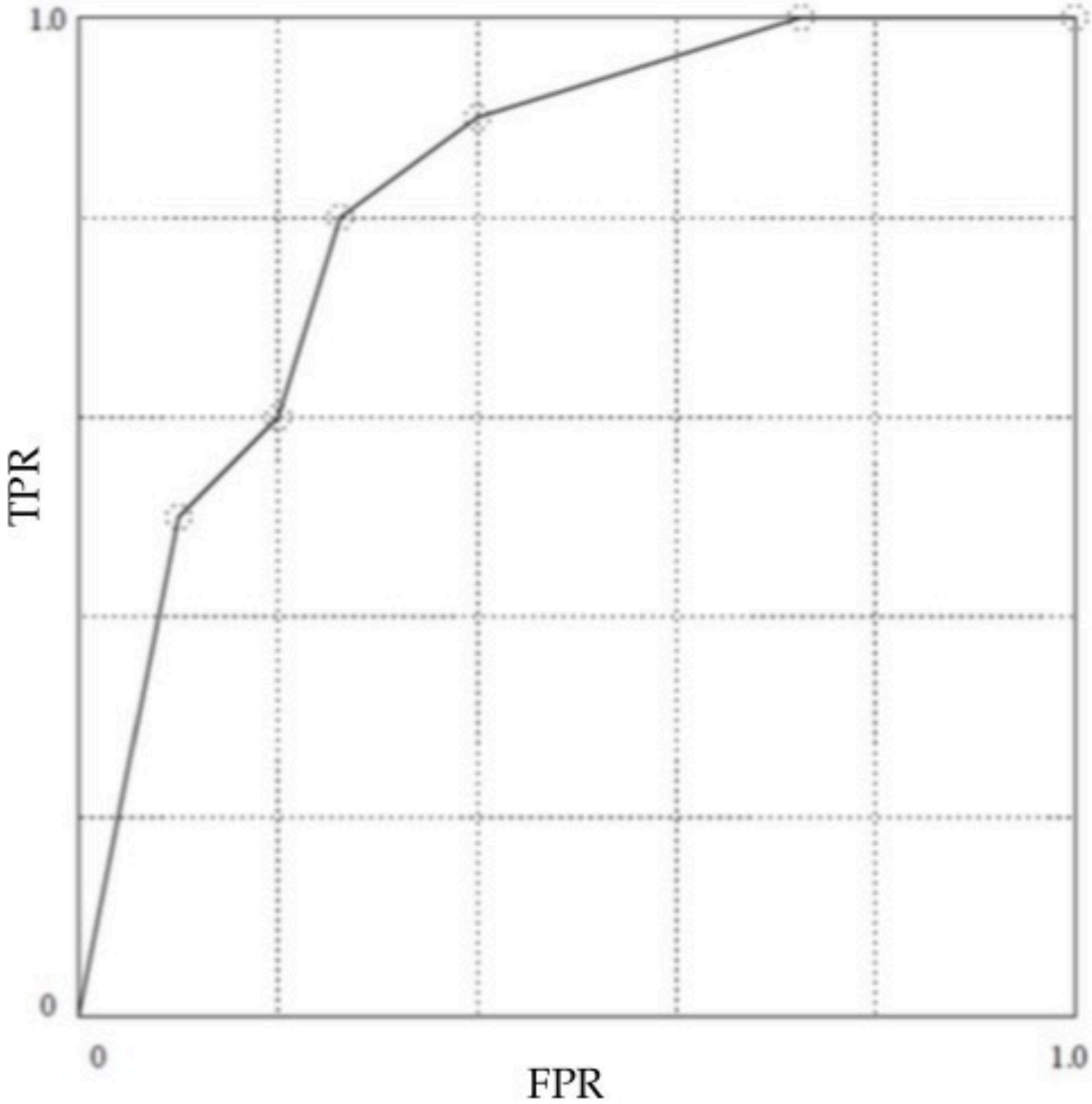
- For a two-class classification task, if the classifier correctly achieved the following results under the corresponding thresholds,

	class1					
Threshold	0.40	0.50	0.60	0.70	0.80	0.90
N-O-C	200	200	180	160	120	100
N-O-Tot	500	400	300	240	180	130
N-O-IC	300	200	120	80	60	30
TPR						
FPR						

- Present the ROC curve for class1. Assume we have 200 class 1 cases and 300 class 2 cases.

	class1					
Threshold	0.40	0.50	0.60	0.70	0.80	0.90
N-O-C	200	200	180	160	120	100
N-O-Tot	500	400	300	240	180	130
N-O-IC	300	200	120	80	60	30
TPR	100%	100%	90%	80%	60%	50%
FPR	100%	66.6%	40%	26.7%	20%	10%

Example two



Precision & Recall

- Precision and recall measure:
  - widely used in the area of *information retrieval*.
  - also used in object recognition/detection sometimes.

- Precision as known in many pattern recognition
  - the measure of *how 'good' the information retrieved by a system* is.
  - Originally, precision refers to the number of relevant documents retrieved by a system as a percentage of the number of documents retrieved, such as:

$$precision = \frac{NO.Relevant\ Documents\ Retrieved}{Total\ NO.Documents\ Retrieved} * 100\%$$

Precision & Recall

- In object detection/recognition --- the number of objects correctly reported by a detection system as a percentage of the total number of objects reported.

- As in the detection rate/false alarm rate,
  - For a single class,
  - $Precision_i = \frac{\sum_{j=1}^{n} N_{true}(i,j)}{\sum_{j=1}^{n} N_{reported}(i,j)} * 100\%$
  - The overall precision for a detection system,
  - $Precision \;=\; \frac{\sum_{j=1}^{n}\sum_{i=1}^{m} N_{true}(i,j)}{\sum_{j=1}^{n}\sum_{i=1}^{m} N_{reported}(i,j)} * 100\%$

Precision & Recall

- Recall or sensitivity
  - a measure of how much of the relevant information was retrieved by a system.
- In information retrieval
  - the number of relevant documents retrieved by a system as a percentage of of the total number of documents in a database.
- $recall = \frac{NO.Relevant\ Documents\ Retrieved}{Total\ NO.\ Relevant\ Documents\ in\ Database} * 100\%$

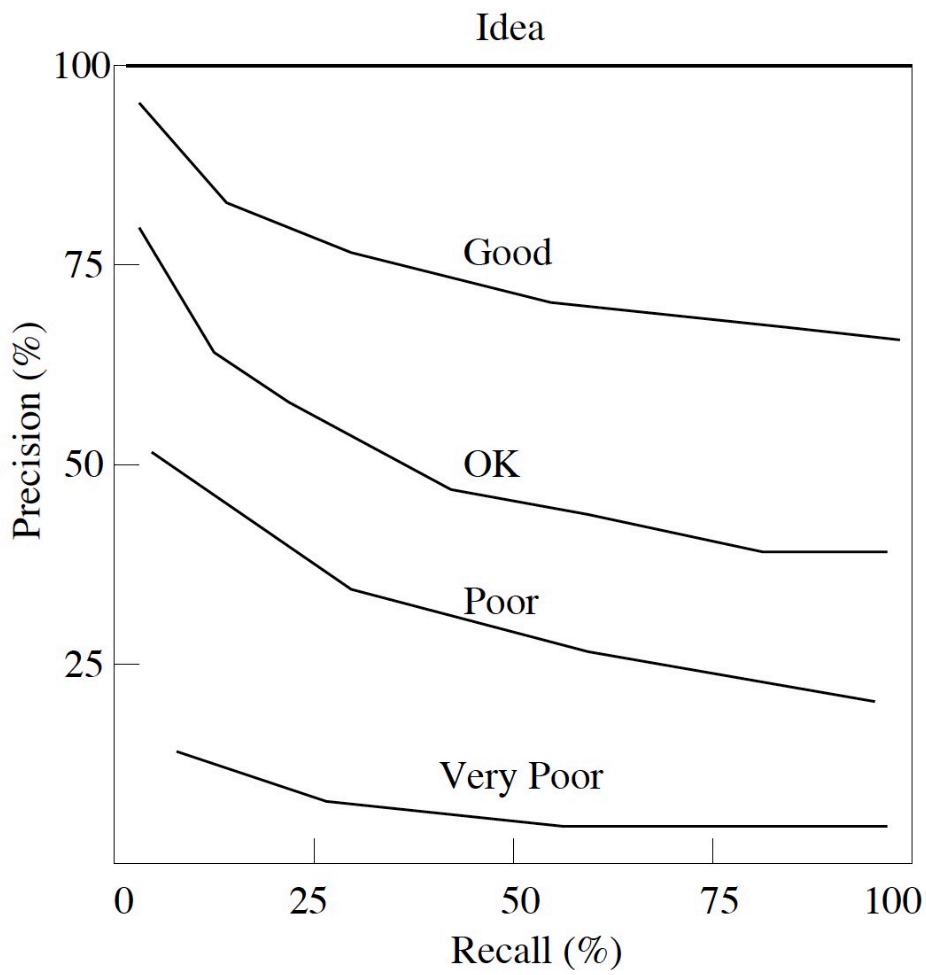
- In object recognition/detection
  - the number of objects correctly reported by a detection system as a percentage of the total number of desired known objects in a database.

Precision & Recall

- For a single class,
  - $$Recall_i = \frac{\sum_{j=1}^{n} N_{true}(i,j)}{\sum_{j=1}^{n} N_{known}(i,j)} * 100\%$$
- The overall recall for a detection system,
  - $$Recall = \frac{\sum_{j=1}^{n}\sum_{i=1}^{m} N_{true}(i,j)}{\sum_{j=1}^{n}\sum_{i=1}^{m} N_{known}(i,j)} * 100\%$$
- Recall is identical to detection rate, accuracy, or TPR.
- The range of recall and precision is [0, 1] or [0, 100%].

Precision vs. Recall

- Here are some sample precision/recall plots:



$$\textbf{F1 Score} = \frac{2}{\left(\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}\right)}$$

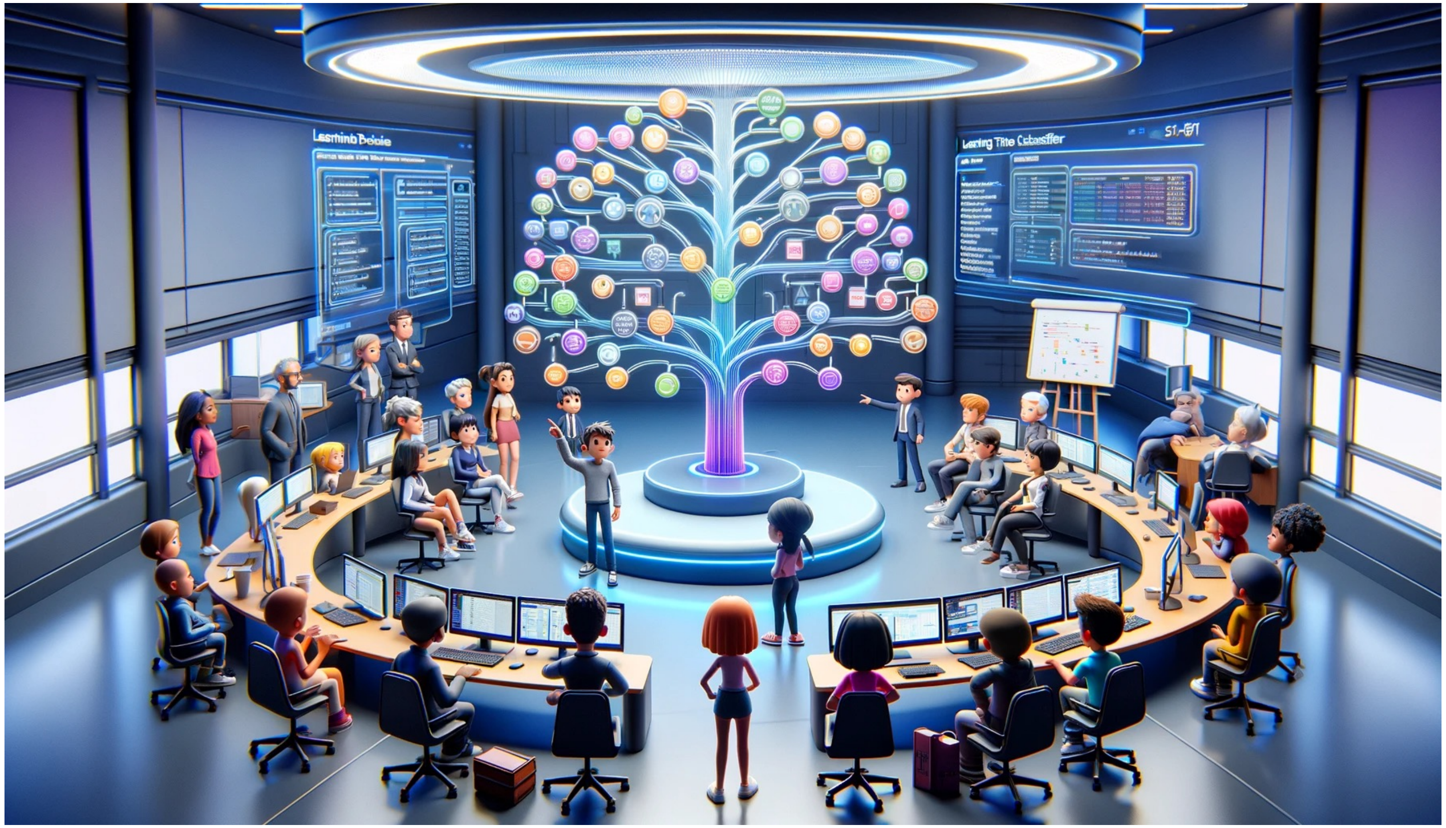
$$\textbf{F1 Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Example Question

- Consider a disease screening test where a classifier is used to predict whether individuals have a disease (positive) or not (negative). After testing 100 individuals, the results are as follows:
  - 30 people actually have the disease (positives), and 70 do not (negatives).
  - The classifier identified 25 individuals as having the disease, of whom 20 actually have the disease (TP = 20).
  - The classifier incorrectly identified 5 healthy individuals as having the disease (FP = 5).
  - The classifier correctly identified 65 individuals as not having the disease (TN = 65).
  - The classifier missed 10 cases of the disease (FN = 10).

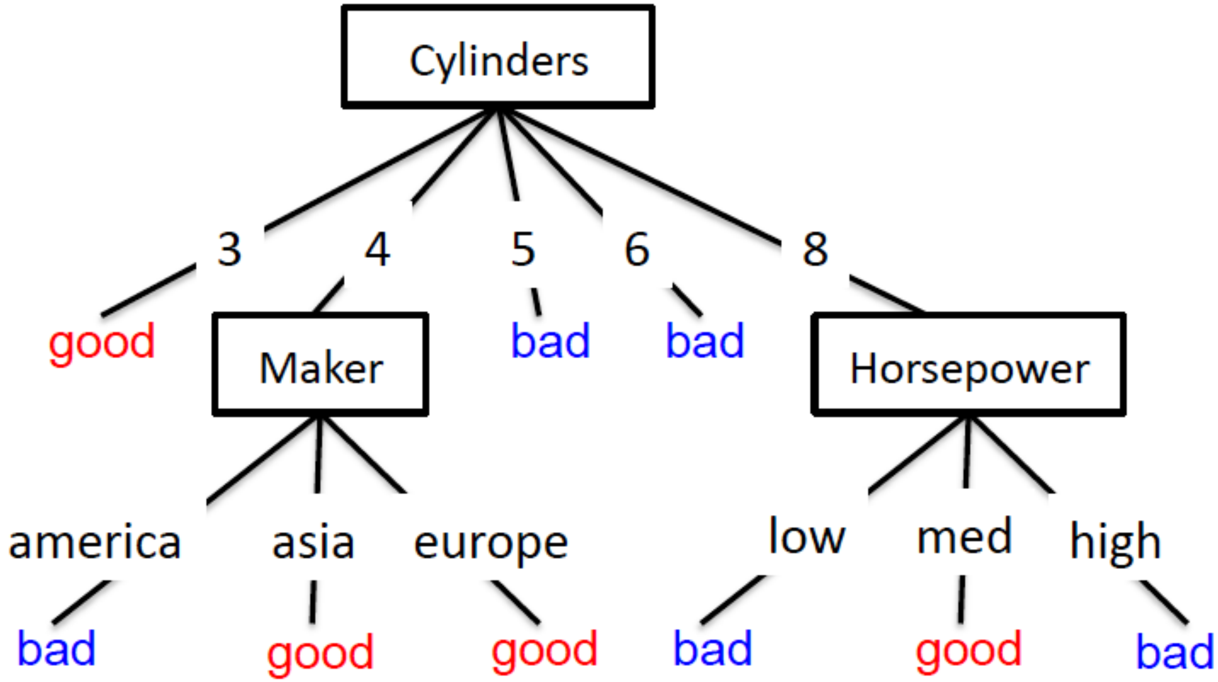
- Question: What are the precision, recall and F1 score of this classifier?

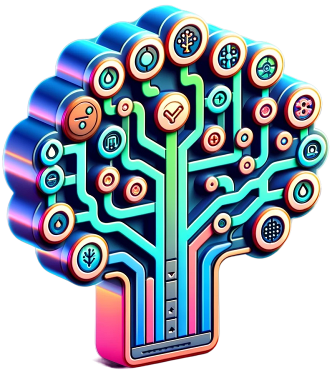
Learning Decision Tree Classifiers



Decision Tree is Human Interpretable

- Each internal node tests an attribute x_i
- One branch for each possible attribute value $x_i = v$
- Each leaf assigns a class y
- To classify input x: traverse the tree from root to leaf, output the label y of the leaf finally reached



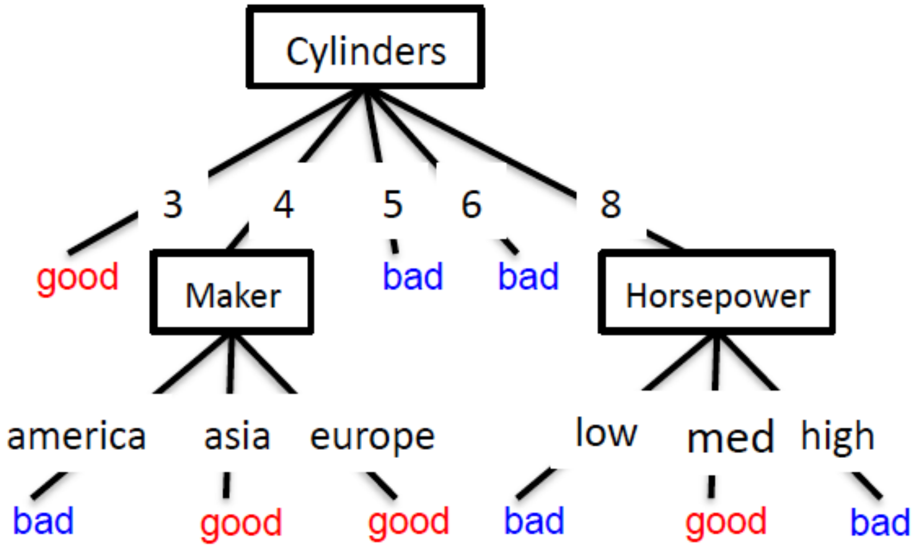


Human interpretable!

Learning Capability of Decision Tree

- Question: How many different classification functions can we build using decision trees on a training dataset?

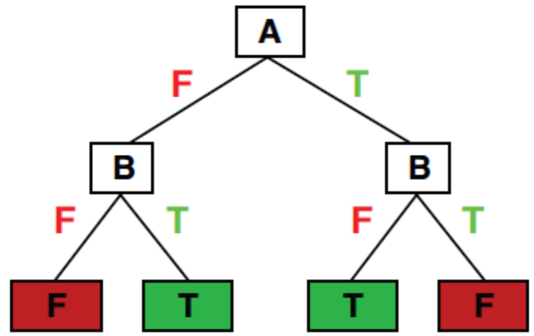
mpg	cylinders	displacement	horsepower	weight	acceleration	modelyear	maker
good	4	low	low	low	high	75to78	asia
bad	6	medium	medium	medium	medium	70to74	america
bad	4	medium	medium	medium	low	75to78	europe
bad	8	high	high	high	low	70to74	america
bad	6	medium	medium	medium	medium	70to74	america
bad	4	low	medium	low	medium	70to74	asia
bad	4	low	medium	low	low	70to74	asia
bad	8	high	high	high	low	75to78	america
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
bad	8	high	high	high	low	70to74	america
good	8	high	medium	high	high	79to83	america
bad	8	high	high	high	low	75to78	america
good	4	low	low	low	low	79to83	america
bad	6	medium	medium	medium	high	75to78	america
good	4	medium	low	low	low	79to83	america
good	4	low	low	medium	high	79to83	america
bad	8	high	high	high	low	70to74	america
good	4	low	medium	low	medium	75to78	europe
bad	5	medium	medium	medium	medium	75to78	europe

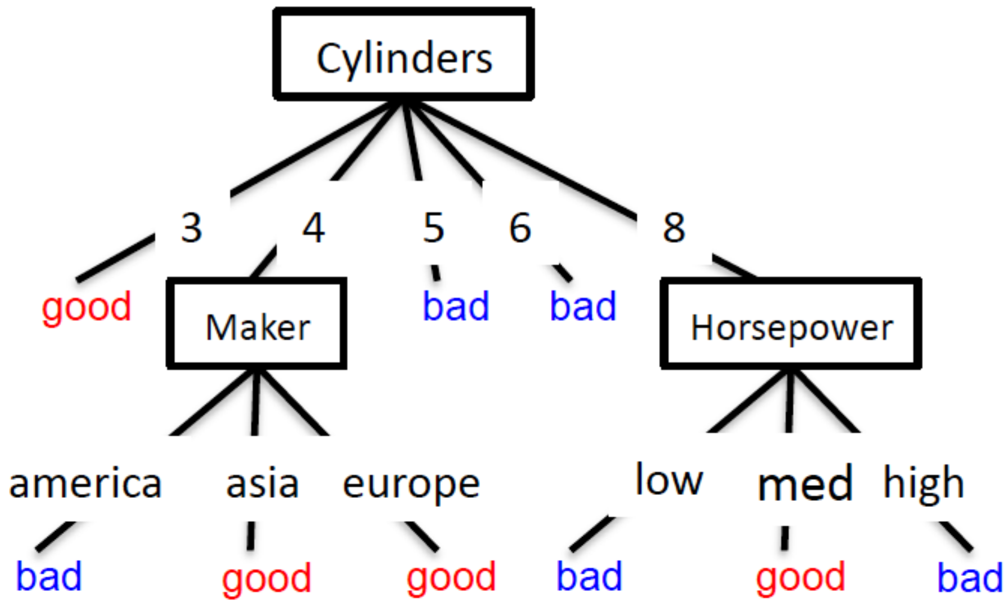


What functions can be represented?

- Decision trees can represent any function of the input features.
- For Boolean functions, each path from root to leaf in a decision tree corresponds to one row of the truth table.
- A decision tree may include an exponential number of nodes.

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F

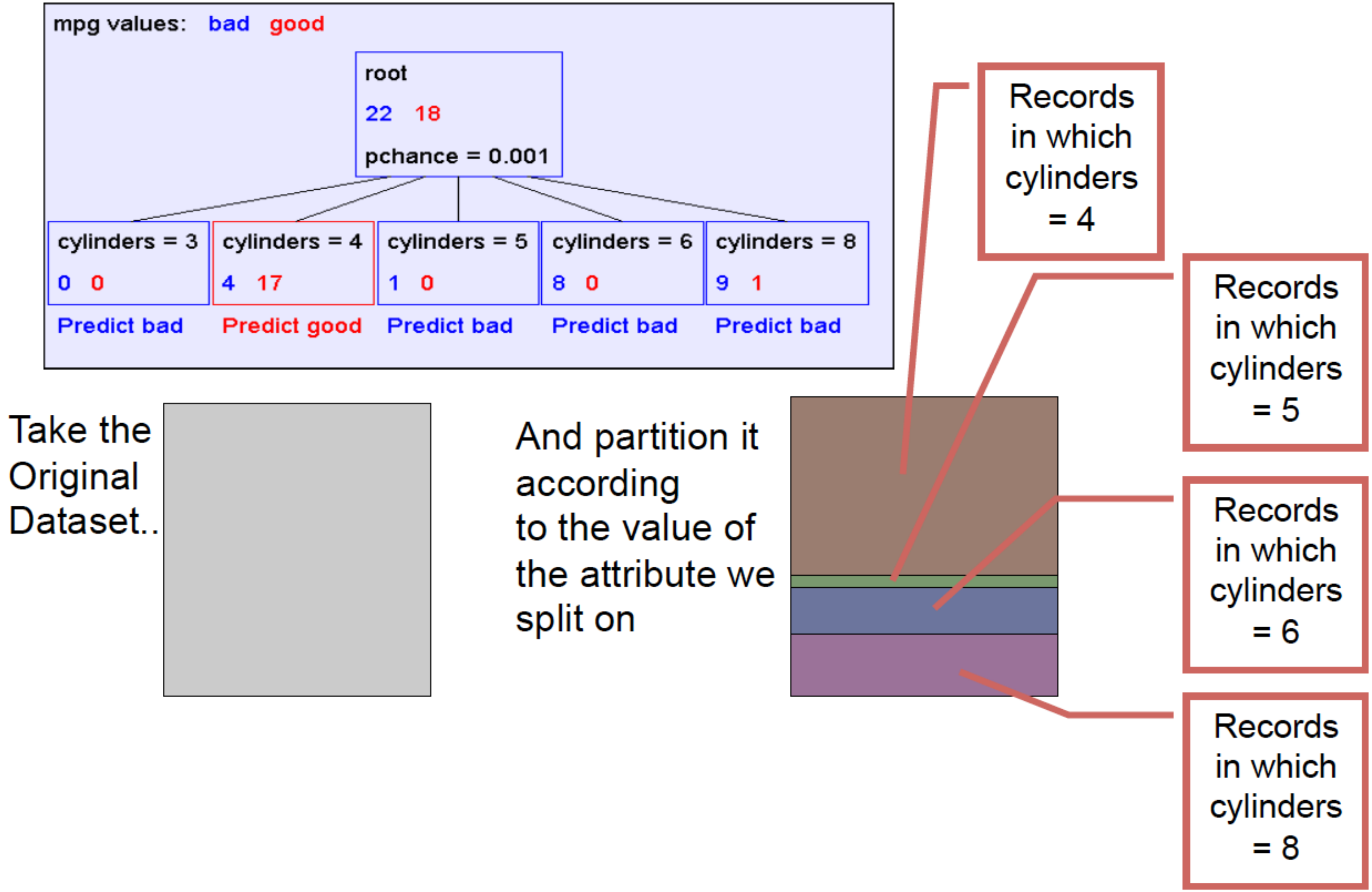




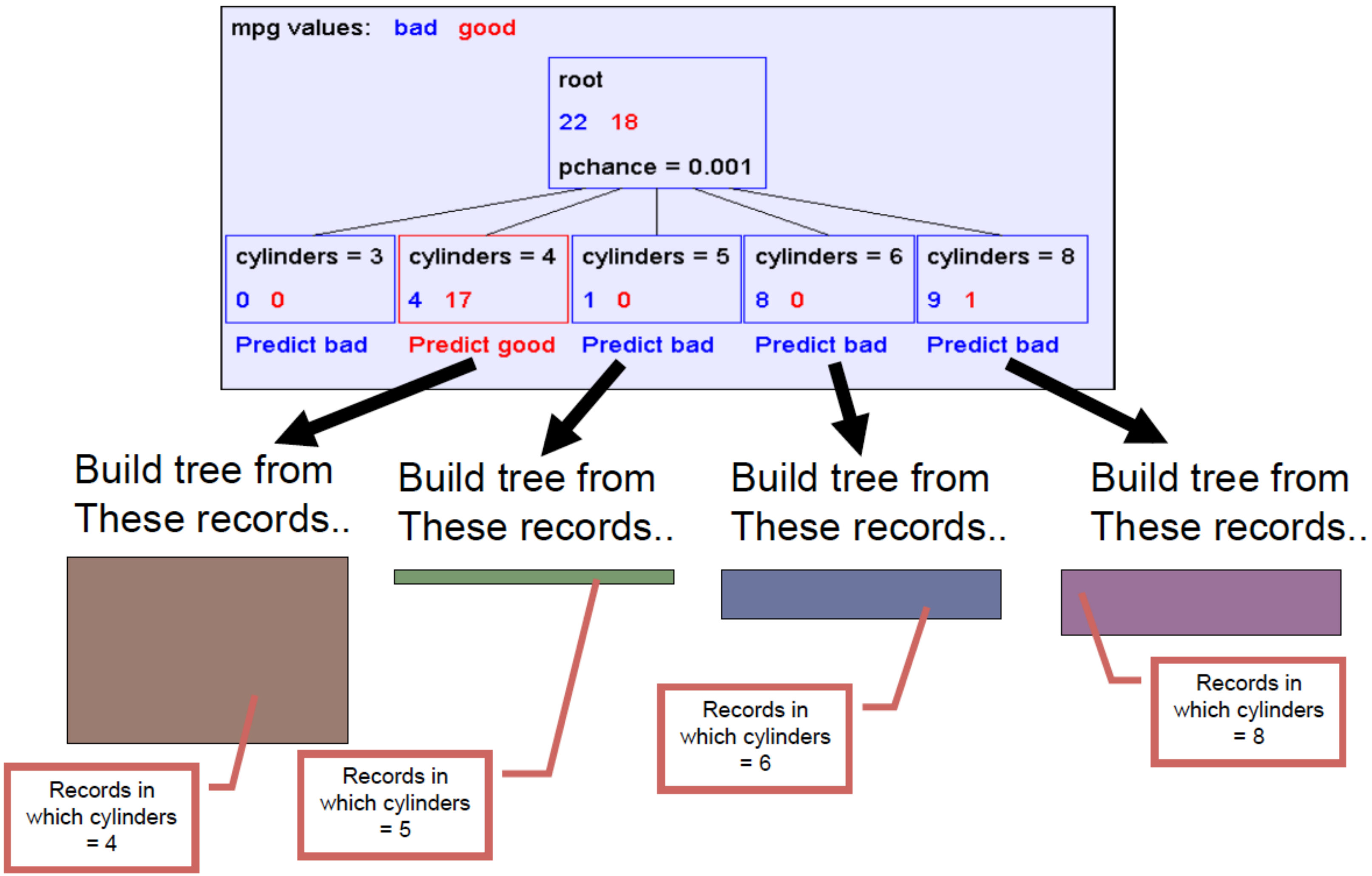
Complexity of Learning Decision Trees

- Learning the simplest (smallest) decision tree is an NP-complete problem
- Resort to a greedy heuristic:
  - Start from an empty decision tree
  - Iteratively split on the best next feature
  - Repeat

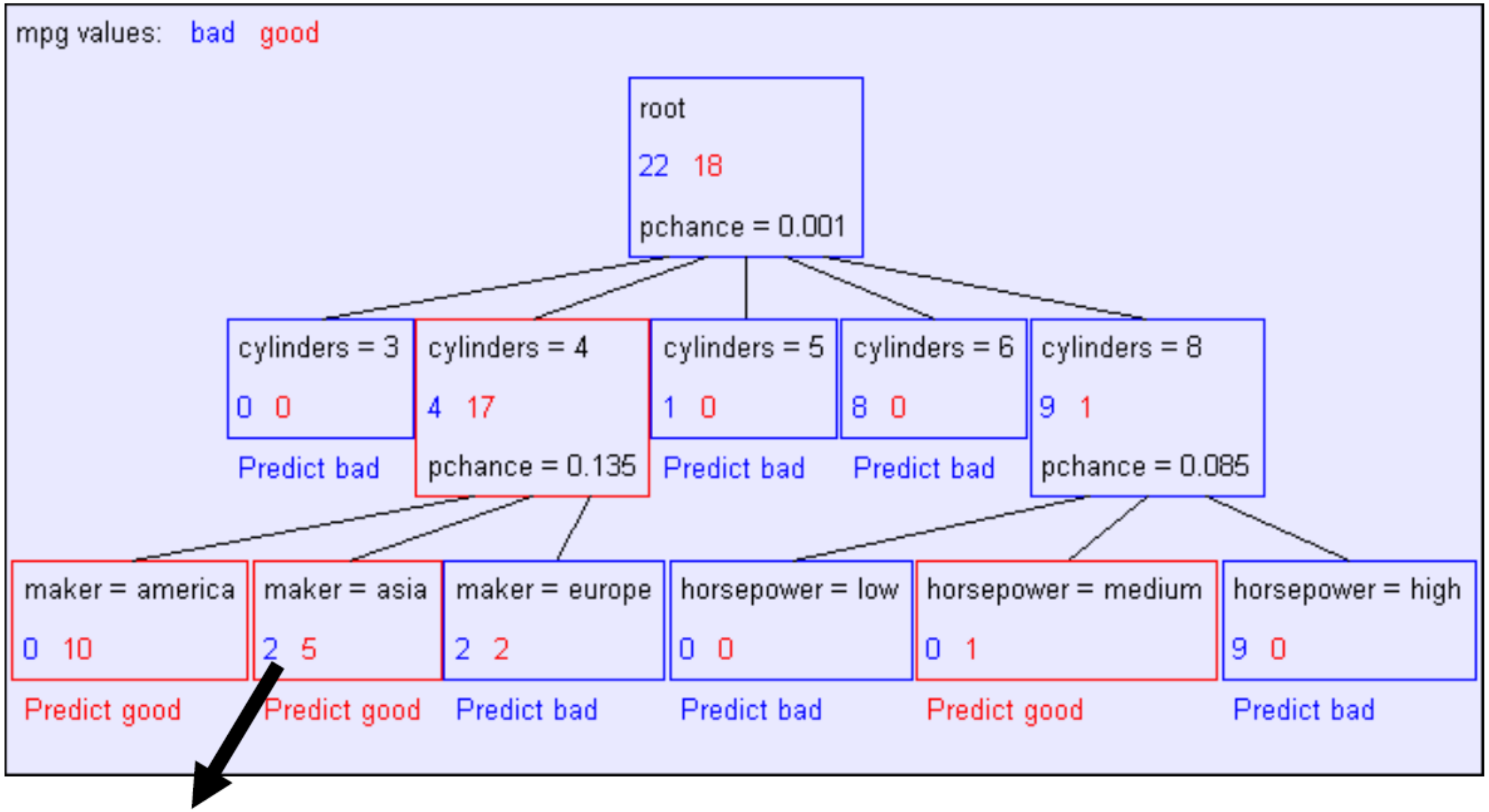
Greedily Learn a Decision Tree Repeatedly



Repeated Construction of Decision Tree



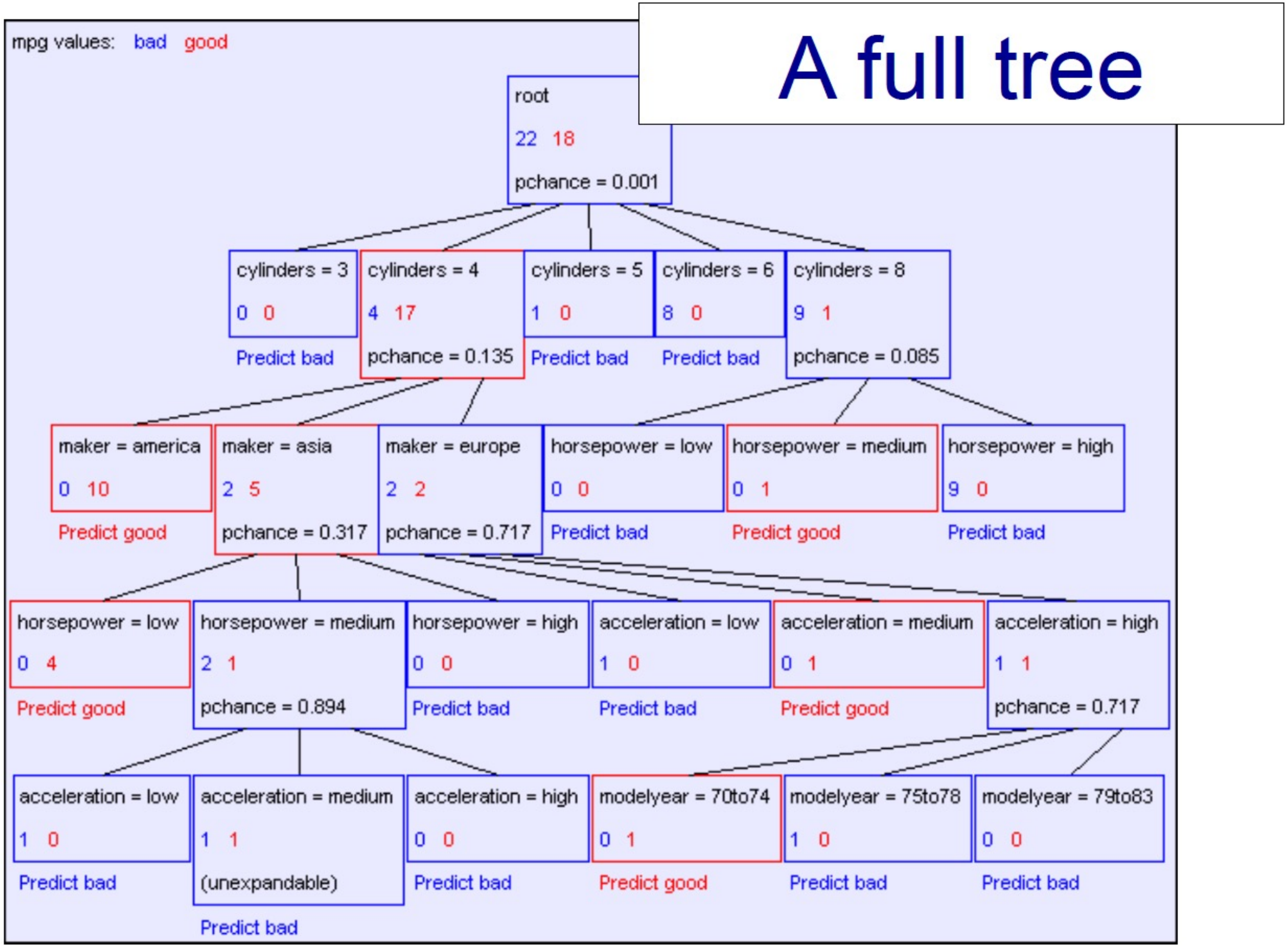
Second Level of the Decision Tree



Recursively build a tree from the seven records in which there are four cylinders and the maker was based in Asia

(Similar recursion in the other cases)

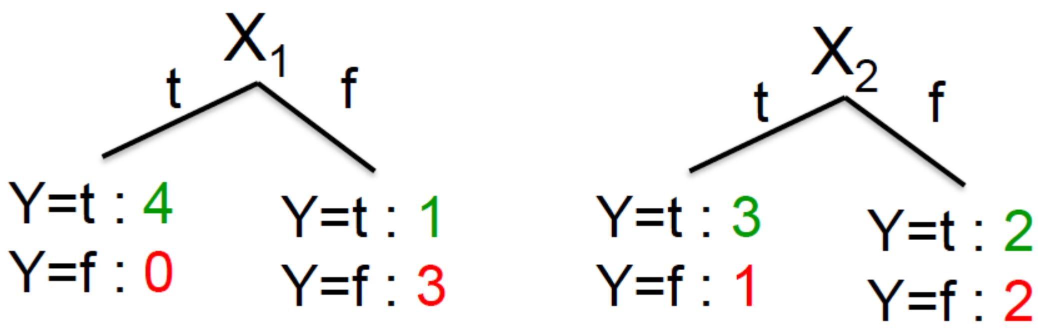
Create a Full Decision Tree



Choose the Suitable Feature to Split

- Should we choose x_1 or x_2 to split?
- Idea:
  - Any splitting on a feature should reduce our uncertainty of the class labels to be applied to each partition after splitting.
  - Use counts at leaves to define probability distributions, so we can measure uncertainty.

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



Measure Uncertainty

- Good split if we are more certain about classification after split
  - Deterministic good (all true or all false)
  - Uniform distribution bad

- Question: What about distributions in between?

P(Y=A) = 1/2	P(Y=B) = 1/4	P(Y=C) = 1/8	P(Y=D) = 1/8

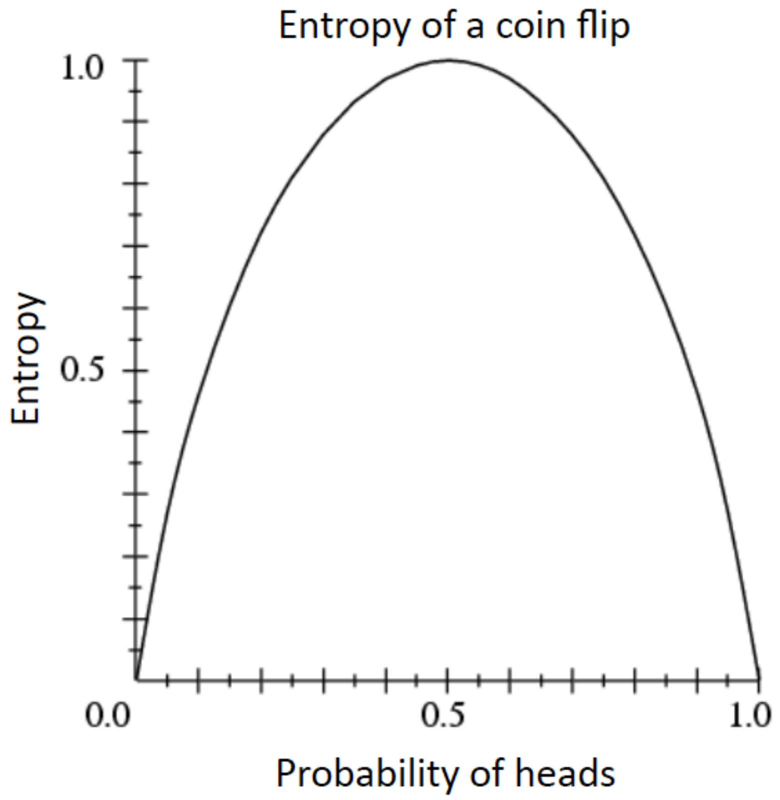
P(Y=A) = 1/4	P(Y=B) = 1/4	P(Y=C) = 1/4	P(Y=D) = 1/4

Entropy

- Entropy of a random variable Y

$$H(Y) = -\sum_{i=1}^{k} P(Y = y_i) \log_2 P(Y = y_i)$$

- Entropy measures the uncertainty over the value of a random variable
  - More uncertainty, more entropy!



High and Low Entropy

- High entropy
  - Y follows a uniform like (flat) distribution
  - Values sampled from Y are less predictable

- Low entropy
  - Y follows a varied (peaks and valleys) distribution
  - Values sampled from Y are more predictable

Entropy Calculation Example

$$H(Y) = -\sum_{i=1}^{k} P(Y = y_i) \log_2 P(Y = y_i)$$

P(Y=t) = 5/6

P(Y=f) = 1/6

$H(Y) = -\ 5/6 \log_2 5/6 - 1/6 \log_2 1/6$

$= 0.65$

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

Conditional Entropy

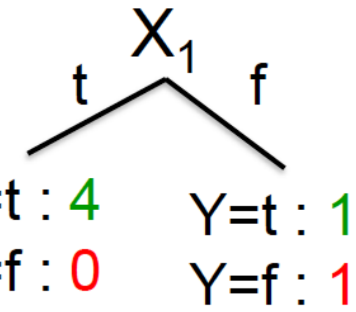
- Conditional entropy $H(Y|X)$ of a random variable Y conditioned on another random variable X

$$H(Y \mid X) = -\sum_{j=1}^{v} P(X = x_j) \sum_{i=1}^{k} P(Y = y_i \mid X = x_j) \log_2 P(Y = y_i \mid X = x_j)$$

Example:

$P(X_1=t) = 4/6$

$P(X_1=f) = 2/6$



$H(Y|X_1) = -\ 4/6\ (1 \log_2 1 + 0 \log_2 0)$

$\quad -\ 2/6\ (1/2 \log_2 1/2 + 1/2 \log_2 1/2)$

$= 2/6$

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

Information Gain

- Decrease in entropy (uncertainty) after using a feature for splitting

$$IG(X) = H(Y) - H(Y \mid X)$$

In our running example:

$IG(X_1) = H(Y) - H(Y|X_1)$
$= 0.65 - 0.33$

$IG(X_1) > 0 \rightarrow$ we prefer the split!

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

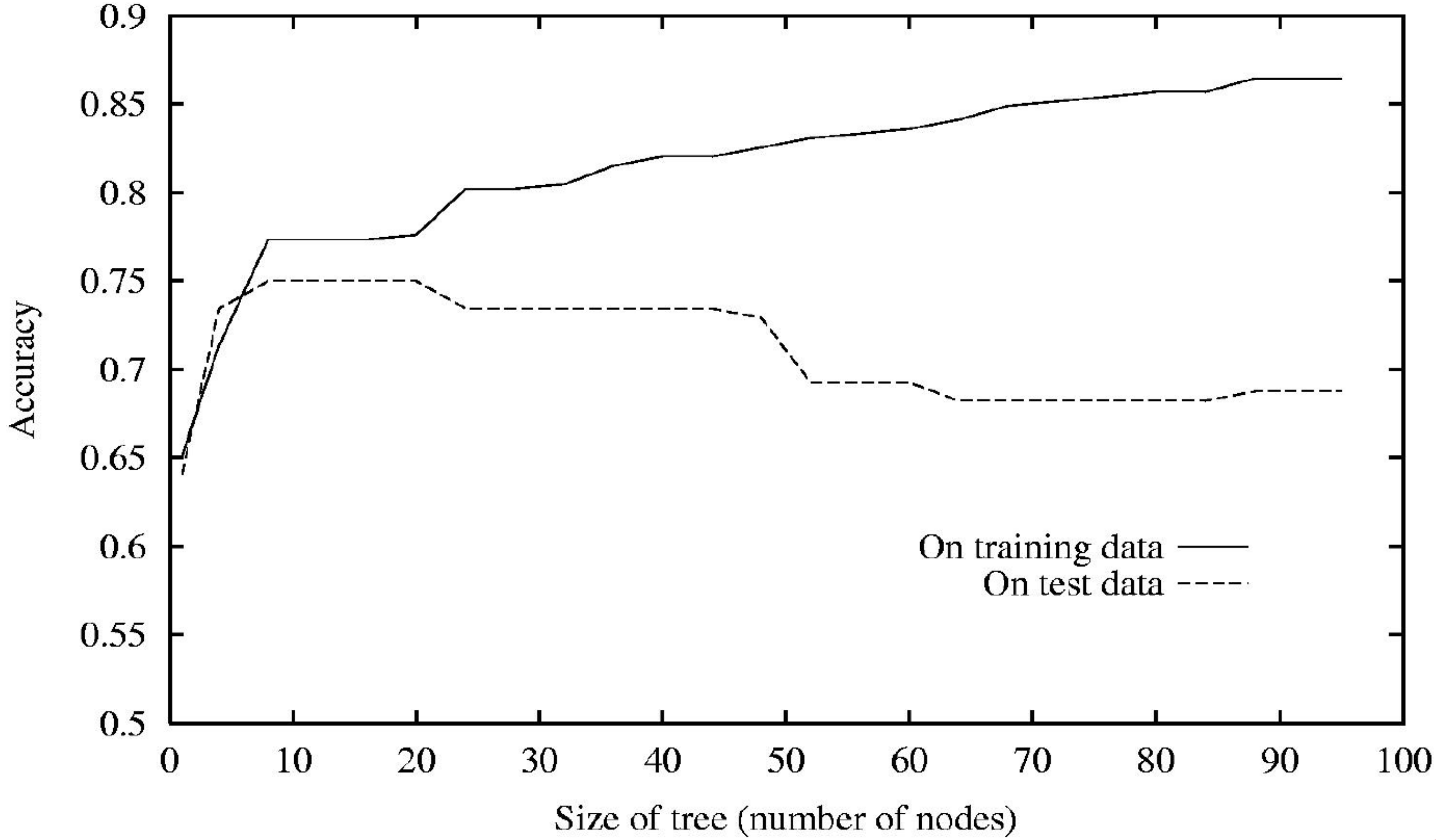
Algorithm to Learn Decision Trees

- Start from empty decision tree
- Split on **next best attribute (feature)**
  - Use, for example, information gain to select attribute:

$$\arg\max_i IG(X_i) = \arg\max_i H(Y) - H(Y \mid X_i)$$

- Recurse

Decision Trees can Overfit



Decision Trees can Overfit

- Standard decision trees have learning bias
  - The learned decision tree can be very large.
  - The accuracy on the training set can be high.
  - However, the accuracy on the test set can be poor.

- To address overfitting, we must introduce a learning bias towards simple trees
  - Set the maximum tree depth
  - Set the minimum partition size of each node (including the leaf node)

- Pruning
  - Pre-pruning: stops the tree from growing before it perfectly classifies the training data
  - Post-pruning: grows a full tree first and then simplify the tree.

Handle Real-Valued Features

- What should we do if some of the input features have real values?

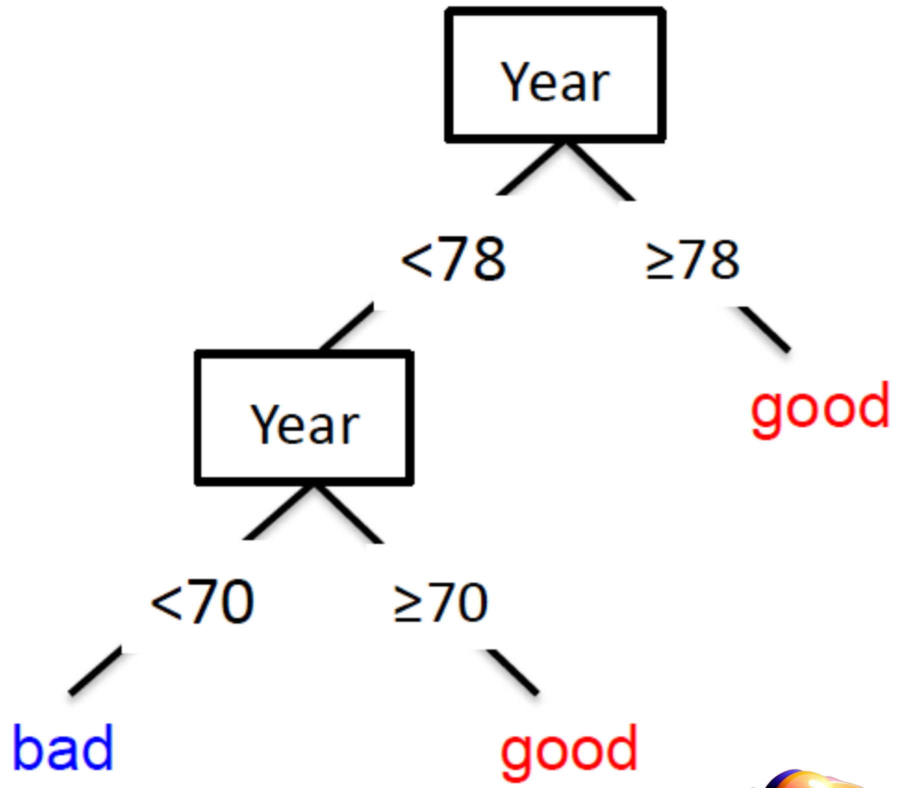
Infinite number of possible split values!!!

mpg	cylinders	displacemen	horsepower	weight	acceleration	modelyear	maker
good	4	97	75	2265	18.2	77	asia
bad	6	199	90	2648	15	70	america
bad	4	121	110	2600	12.8	77	europe
bad	8	350	175	4100	13	73	america
bad	6	198	95	3102	16.5	74	america
bad	4	108	94	2379	16.5	73	asia
bad	4	113	95	2228	14	71	asia
bad	8	302	139	3570	12.8	78	america
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
good	4	120	79	2625	18.6	82	america
bad	8	455	225	4425	10	70	america
good	4	107	86	2464	15.5	76	europe
bad	5	131	103	2830	15.9	78	europe

Threshold based Split

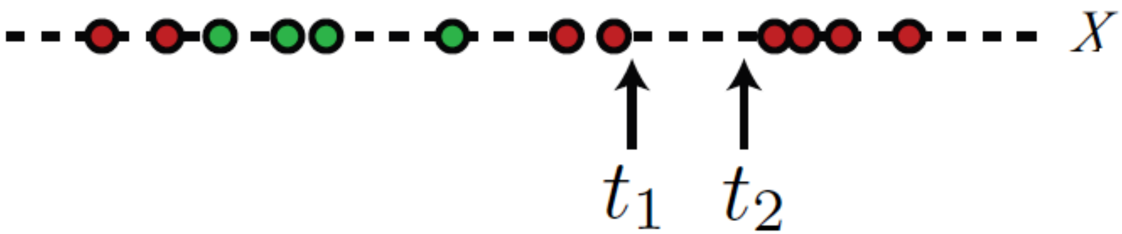
- Binary tree: split on any feature x based on threshold t
  - Branch 1: $x < t$
  - Branch 2: $x \geq t$

- Small change of the learning algorithm is required
  - Allow repeated split of the same input feature based on different thresholds



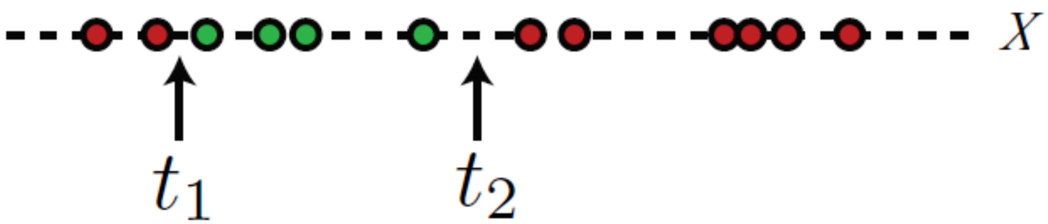
Determine All Possible Thresholds

- For any input feature X, only a finite number of thresholds are potentially important
- How to identify important thresholds?
  - Sort data according to X into $\{x_1, x_2, \cdots, x_n\}$



  - Consider the split threshold of the form $\frac{x_i + x_{i+1}}{2}$

  - Further simplification: only splits between data instances of different classes matter



Choose the Best Threshold

- Suppose *X* is real valued with threshold *t*
- Want **IG(Y | X:t)**, the information gain for Y when testing if *X* is greater than or less than *t*
- Define:
  - $H(Y|X{:}t) = p(X < t)\, H(Y|X < t) + p(X >= t)\, H(Y|X >= t)$
  - $IG(Y|X{:}t) = H(Y) - H(Y|X{:}t)$
  - $IG^*(Y|X) = \max_t IG(Y|X{:}t)$
- Use: $IG^*(Y|X)$ for continuous variables

Supervised Learning for Classification

- So far …
  - K-nearest neighbors
  - Perceptron
  - Logistic regression
  - SVMs
  - Decision trees
  - Neural networks

- Which technique should we pick?