DS 4400

Machine Learning and Data Mining I
Spring 2024

David Liu
Khoury College of Computer Science
Northeastern University

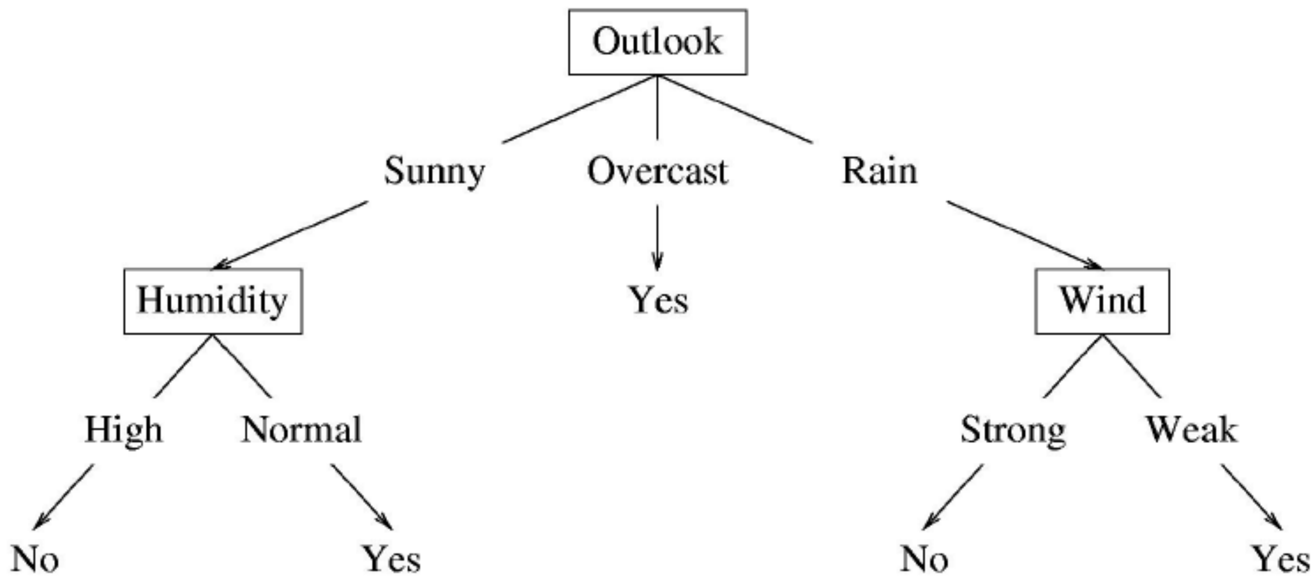
March 1 2024

Outline

- Decision trees
  - Information gain / entropy measures
  - Training algorithm
  - Example
- Ensemble models
  - Bagging
  - Boosting

Decision Tree

- A possible decision tree for the data:

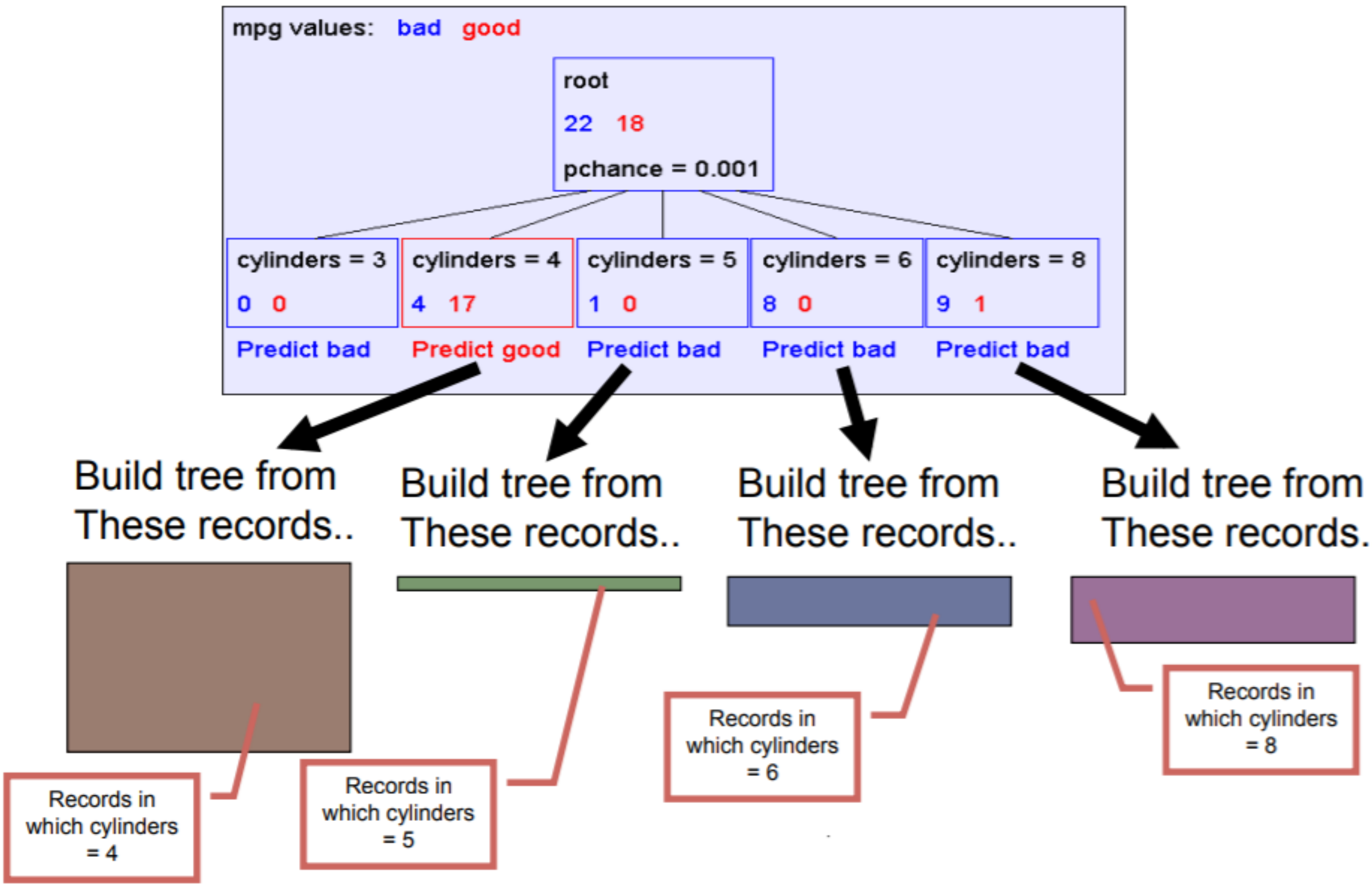


- Each internal node: test one attribute X_i
- Each branch from a node: selects one value for X_i
- Each leaf node: predict Y (or $p(Y \mid x \in \text{leaf})$)

Learning Decision Trees

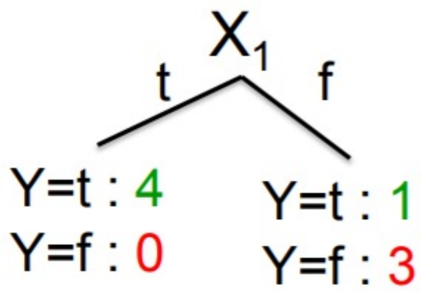
- Learning the simplest (smallest) decision tree is an NP-complete problem [Hyafil & Rivest ’76]
- Resort to a greedy heuristic:
  - Start from empty decision tree
  - Split on **next best attribute (feature)**
  - Recurse

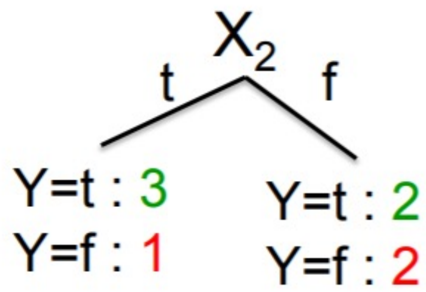
Key Idea: Use Recursion Greedily



Splitting

Would we prefer to split on X_1 or X_2?

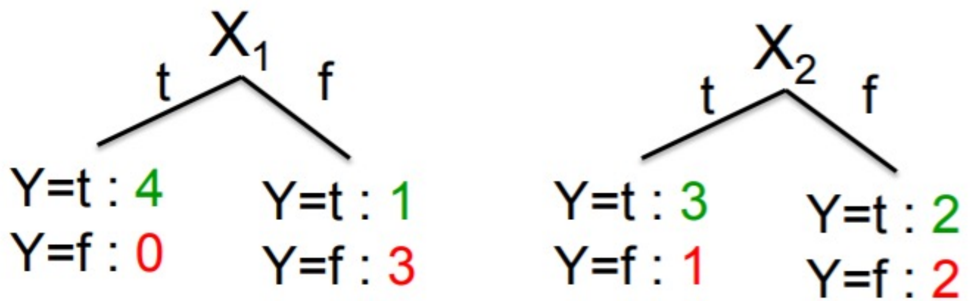




X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F
F	T	F
F	F	F

Splitting

Would we prefer to split on X_1 or X_2?



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

Idea: use counts at leaves to define probability distributions, so we can measure uncertainty!

Use entropy-based measure (Information Gain)

Entropy

Suppose X can have one of *m* values… $V_1, V_2, \ldots V_m$

$P(X=V_1) = p_1$	$P(X=V_2) = p_2$	….	$P(X=V_m) = p_m$

What's the smallest possible number of bits, on average, per symbol, needed to transmit a stream of symbols drawn from X's distribution? It's

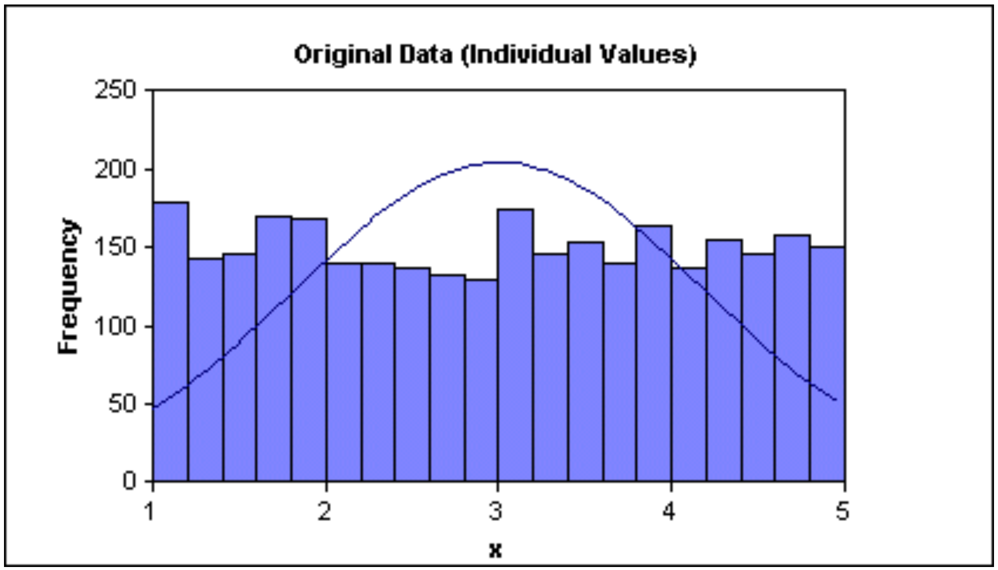
$$H(X) = -p_1 \log_2 p_1 - p_2 \log_2 p_2 - \ldots - p_m \log_2 p_m$$
$$= -\sum_{j=1}^{m} p_j \log_2 p_j$$

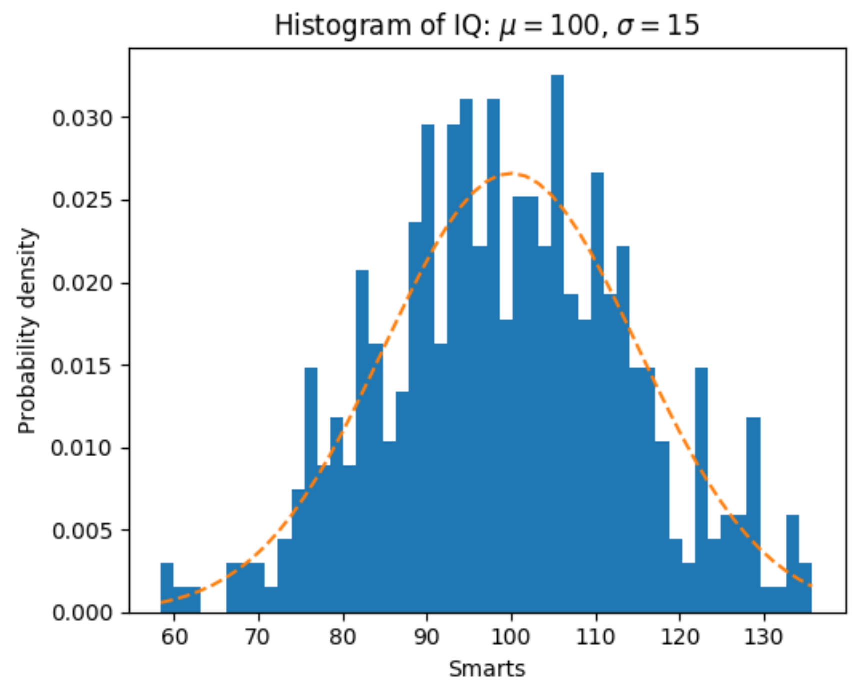
H(X) = The entropy of X

- "High Entropy" means X is from a uniform (boring) distribution
- "Low Entropy" means X is from varied (peaks and valleys) distribution

High/Low Entropy

Which distribution has high entropy?





Conditional Entropy

Suppose I'm trying to predict output Y and I have input X

X = College Major

Y = Likes "Gladiator"

Let's assume this reflects the true probabilities

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Conditional Entropy

Suppose I'm trying to predict output Y and I have input X

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Let's assume this reflects the true probabilities

E.G. From this data we estimate

- *P(LikeG = Yes) = 0.5*
- *P(Major = Math & LikeG = No) = 0.25*
- *P(Major = Math) = 0.5*
- *P(LikeG = Yes | Major = History) = 0*

Note:

- $H(X) = 1.5$
- $H(Y) = 1$

Conditional Entropy

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Definition of Specific Conditional Entropy:

$H(Y|X=v)$ = **The entropy of** Y **among only those records in which** X **has value** v

Example:

- $H(Y|X=Math) =$
- $H(Y|X=History) =$
- $H(Y|X=CS) =$

Conditional Entropy

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Definition of Specific Conditional Entropy:

$H(Y|X=v)$ = **The entropy of** Y **among only those records in which** X **has value** v

Example:

- $H(Y|X=Math) = 1$
- $H(Y|X=History) = 0$
- $H(Y|X=CS) = 0$

Conditional Entropy

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Definition of Conditional Entropy:

$H(Y|X)$ = The average specific conditional entropy of Y

= if you choose a record at random what will be the conditional entropy of Y, conditioned on that row's value of X

= Expected number of bits to transmit Y if both sides will know the value of X

$= \Sigma_j \, Prob(X=v_j) \, H(Y \mid X = v_j)$

Conditional Entropy

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Definition of Conditional Entropy:

$H(Y|X)$ = The average conditional entropy of Y

$= \Sigma_j Prob(X=v_j)\, H(Y \mid X = v_j)$

Example:

v_j	$Prob(X=v_j)$	$H(Y \mid X = v_j)$
Math		
History		
CS		

Conditional Entropy

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Definition of Conditional Entropy:

$H(Y|X)$ = The average conditional entropy of Y

$= \Sigma_j Prob(X=v_j)\ H(Y \mid X = v_j)$

Example:

v_j	$Prob(X=v_j)$	$H(Y \mid X = v_j)$
Math	0.5	1
History	0.25	0
CS	0.25	0

$H(Y|X) = 0.5 * 1 + 0.25 * 0 + 0.25 * 0 = 0.5$

Information Gain

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Definition of Information Gain:

$IG(Y|X)$ = **I must transmit** Y. **How many bits on average would it save me if both ends of the line knew** X**?**

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

Example:

- **H(Y) =**
- **H(Y|X) =**
- **Thus IG(Y|X) =**

Information Gain

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Definition of Information Gain:

$IG(Y|X)$ = **I must transmit** Y. **How many bits on average would it save me if both ends of the line knew** X**?**

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

Example:

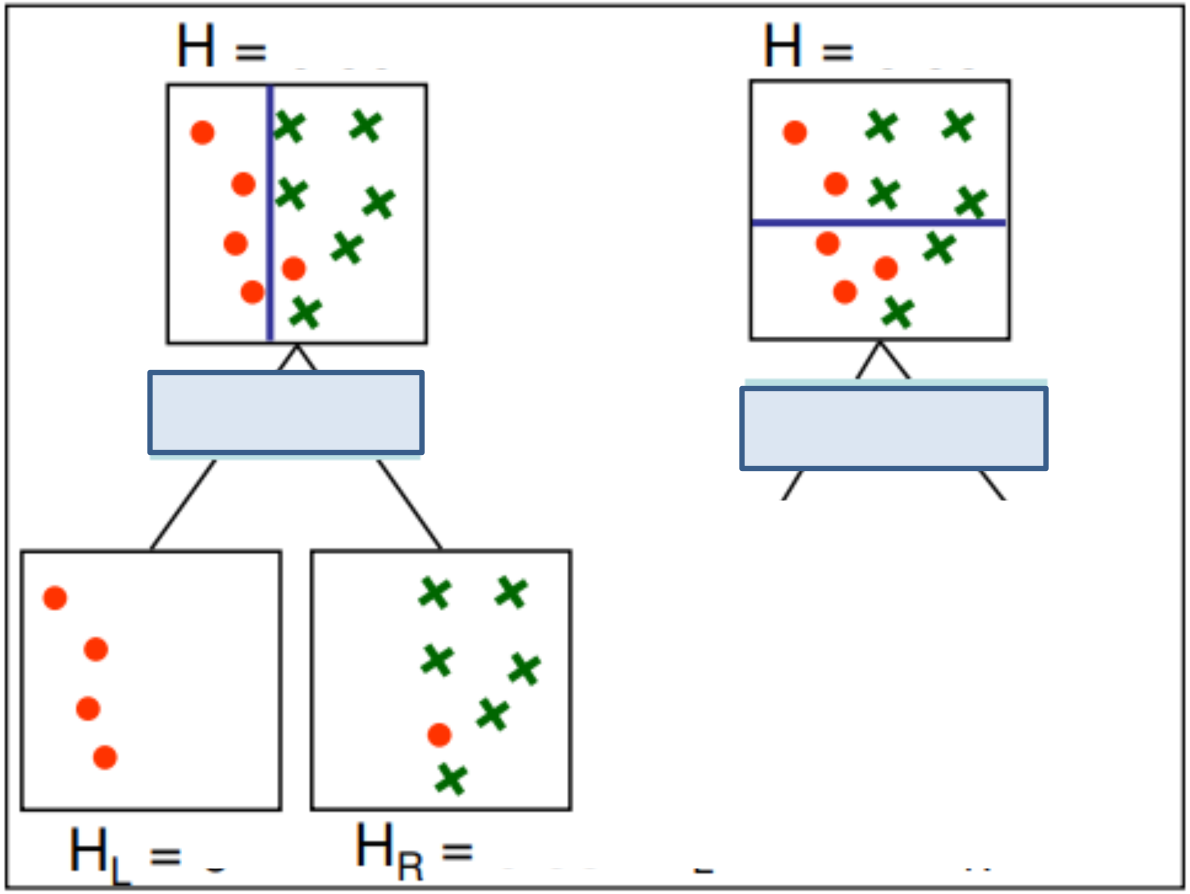
- **H(Y) = 1**
- **H(Y|X) = 0.5**
- **Thus IG(Y|X) = 1 – 0.5 = 0.5**

Relevance for decision trees

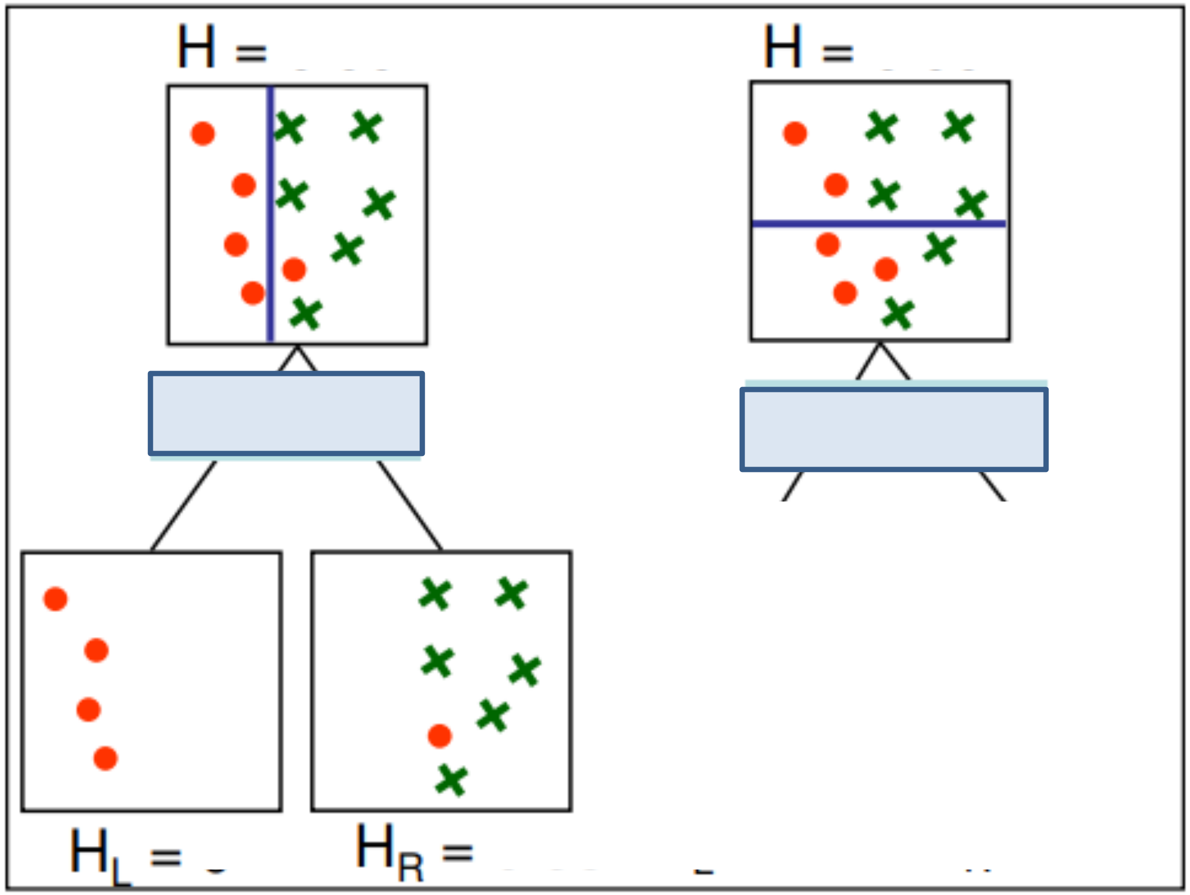
- Multiple features $X_1, \ldots, X_d$
- Label Y: Initial entropy $H(Y)$
- How much each feature X_i helps explain uncertainty in Y
  - Compute Information gain

$$IG(Y|X_i) = H(Y) - H(Y|X_i)$$

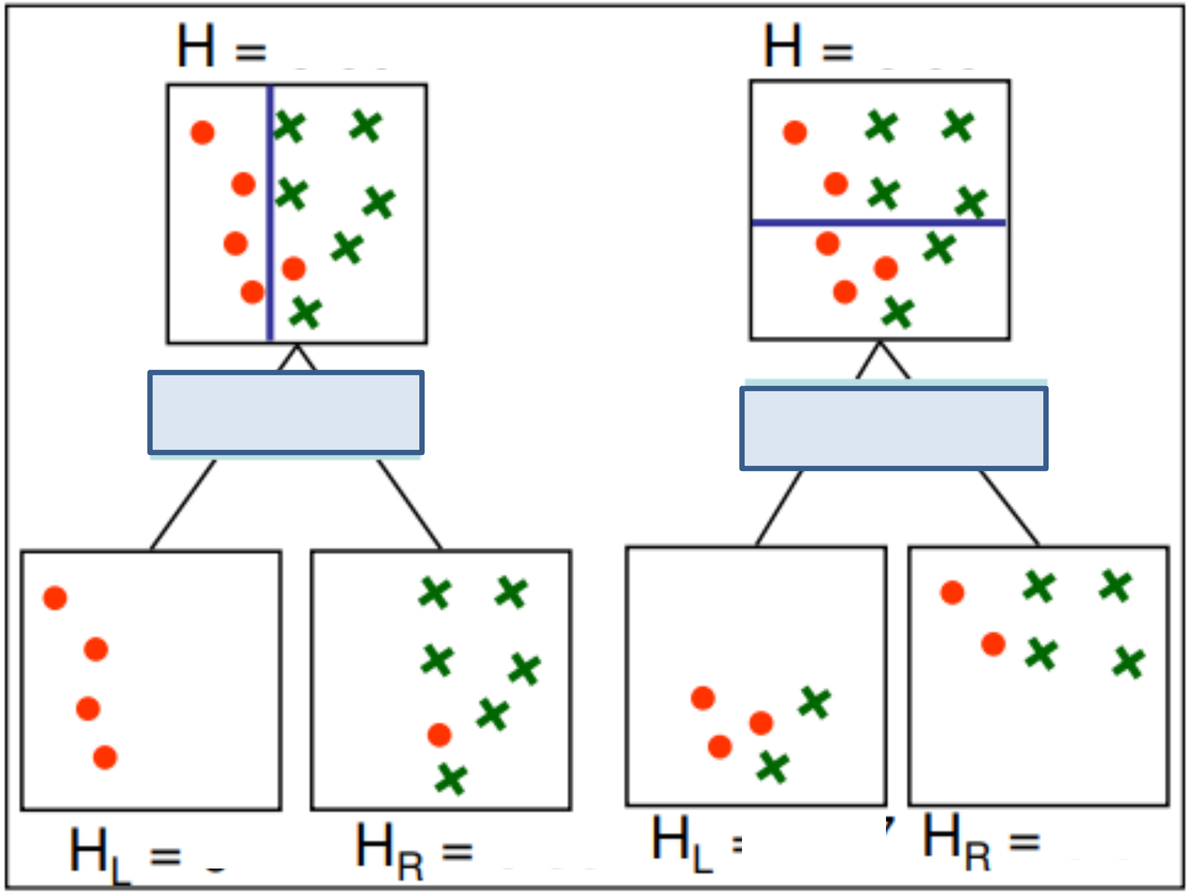
Example Information Gain



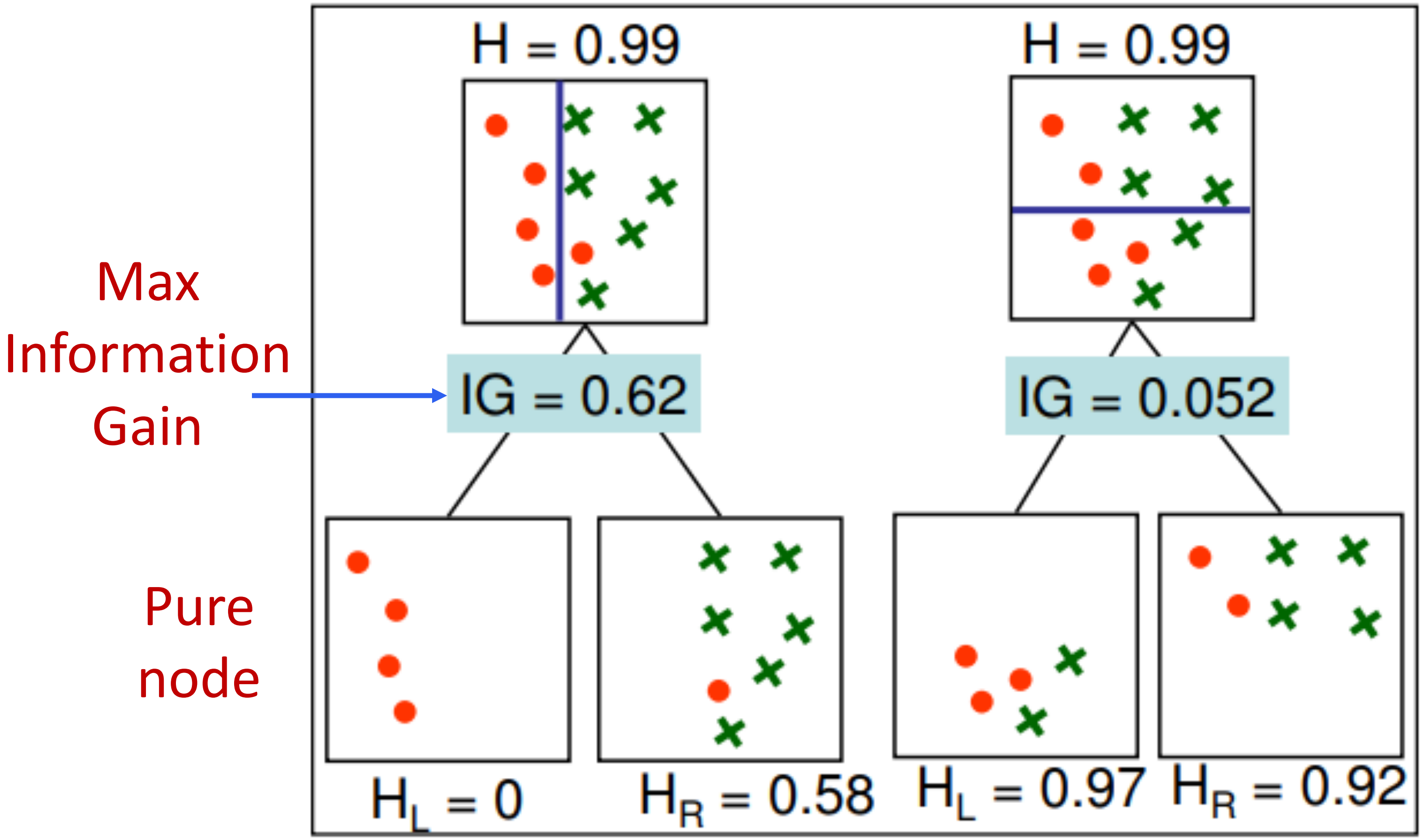
Example Information Gain



Example Information Gain



Example Information Gain



Learning 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

ID3 algorithm uses Information Gain
Information Gain reduces uncertainty on Y

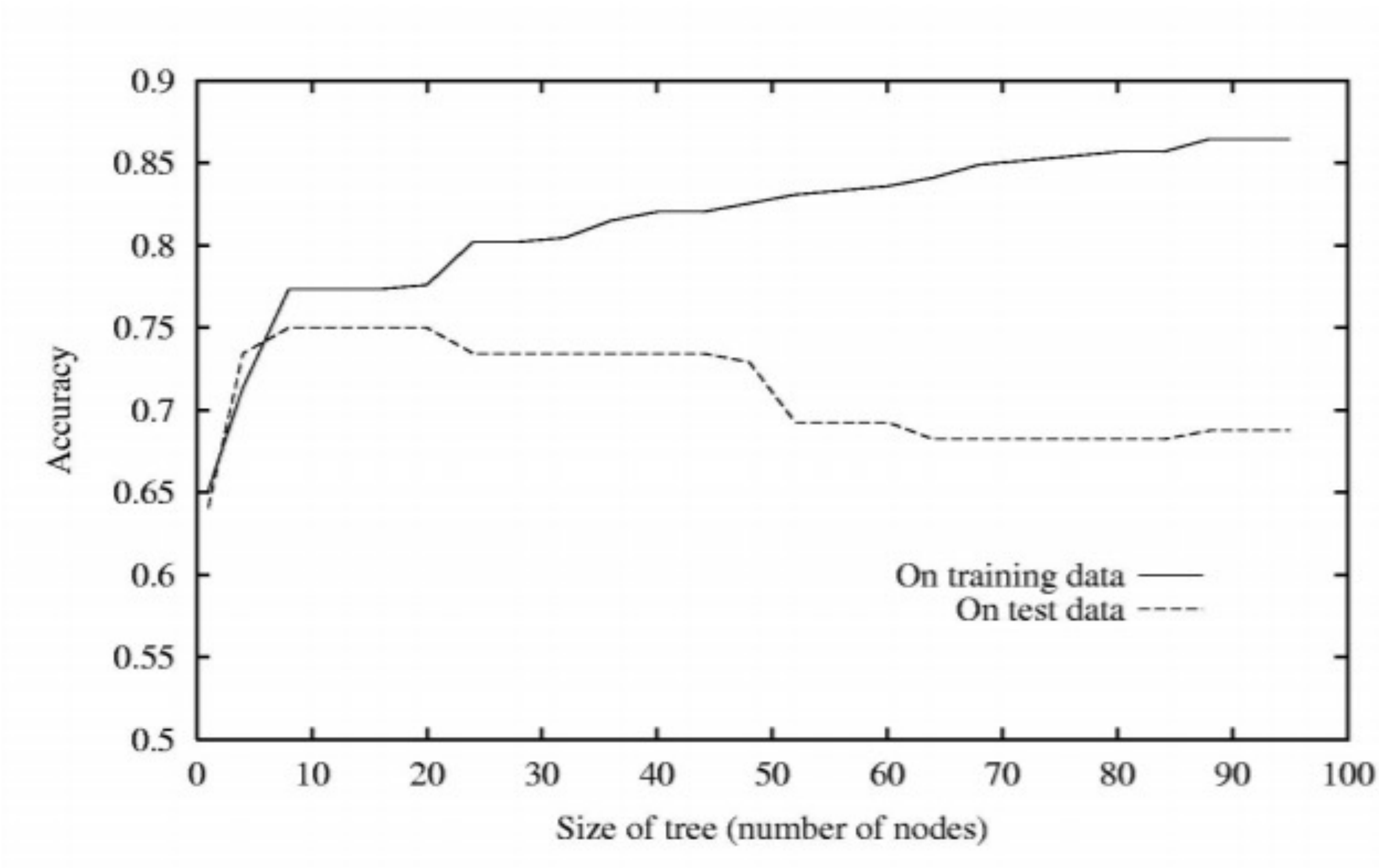
Impurity Metrics

Split a node according to max reduction of impurity

1. Entropy
2. Gini Index
   - For binary case with prob p_0, p_1:
     $$I(p_0, p_1) = 2p_0 p_1 = 2p_0(1 - p_0)$$
   - For multi-class with prob $p_1, \ldots, p_K$:
     $$I(p_1, \ldots p_K) = \sum_{i=1}^{K} p_i \ (1 - p_i)$$

- Properties
  - Impurity metrics have value 0 for pure nodes
  - Impurity metrics are maximized for uniform distribution (nodes with most uncertainty)

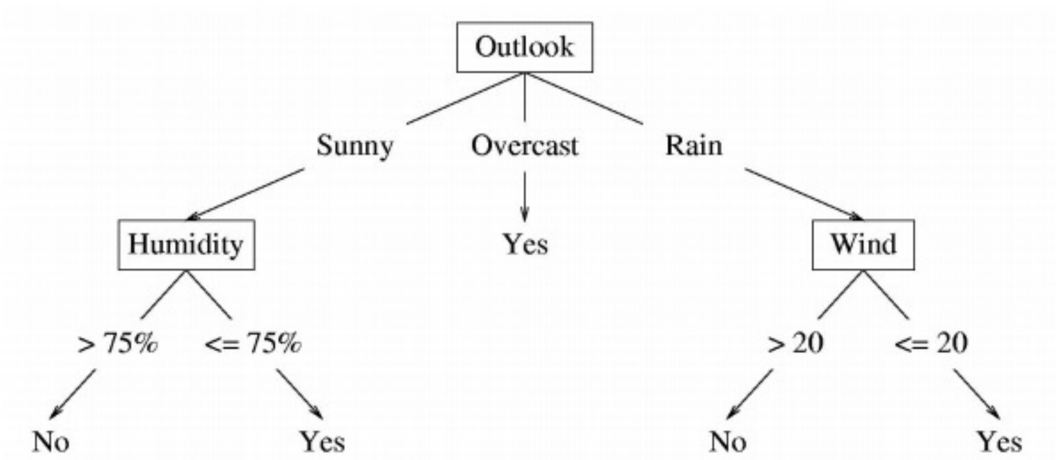
Overfitting



Solutions against Overfitting

- Standard decision trees have no learning bias
  - Training set error is always zero!
    - (If there is no label noise)
  - Lots of variance
  - Must introduce some bias towards simpler trees
- Many strategies for picking simpler trees
  - Fixed depth
  - Minimum number of samples per leaf
- Pruning
  - Remove branches of the tree that increase error using cross-validation

Real-valued Features



- Change to binary splits by choosing a threshold
- One method:
  - Sort instances by value, identify adjacencies with different classes

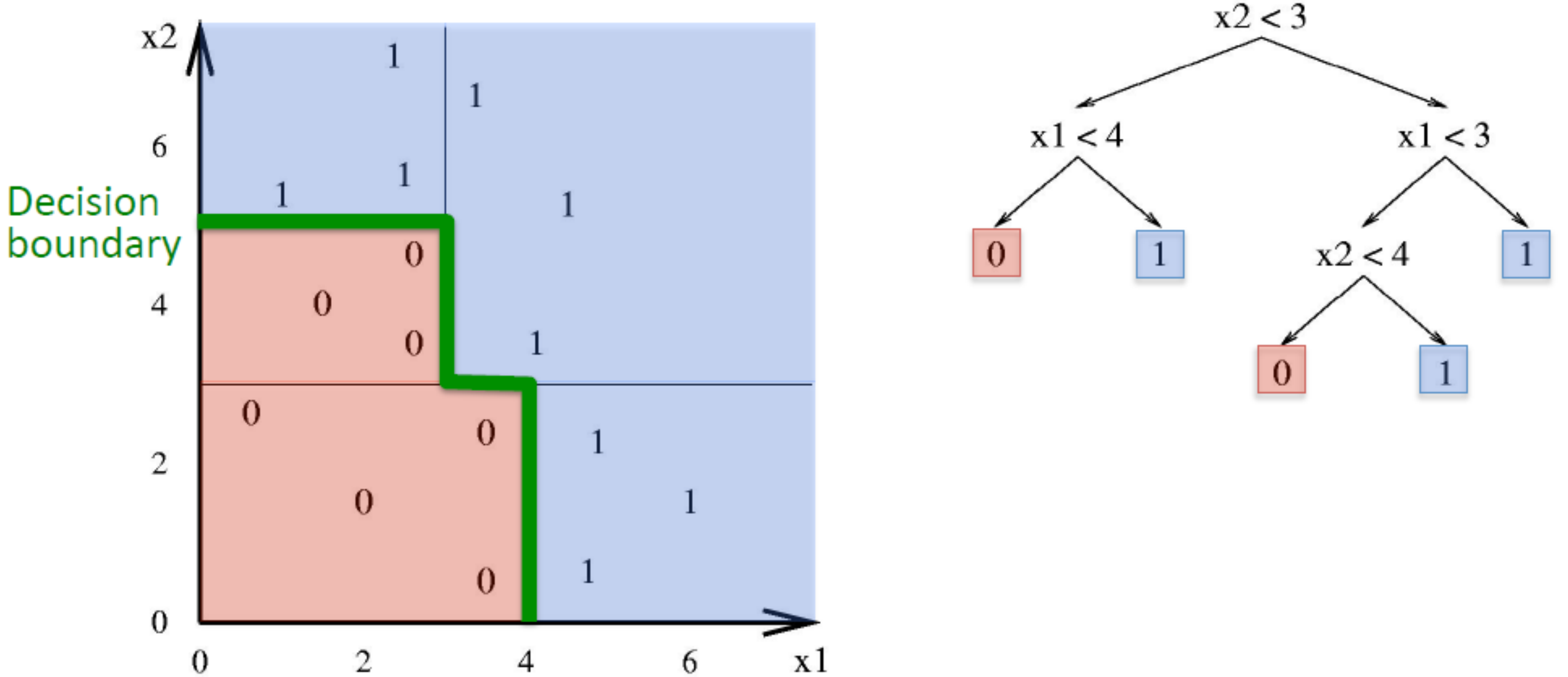
Humidity	40	48	60	72	80	90
PlayTennis:	No	No	Yes	Yes	Yes	No

candidate splits

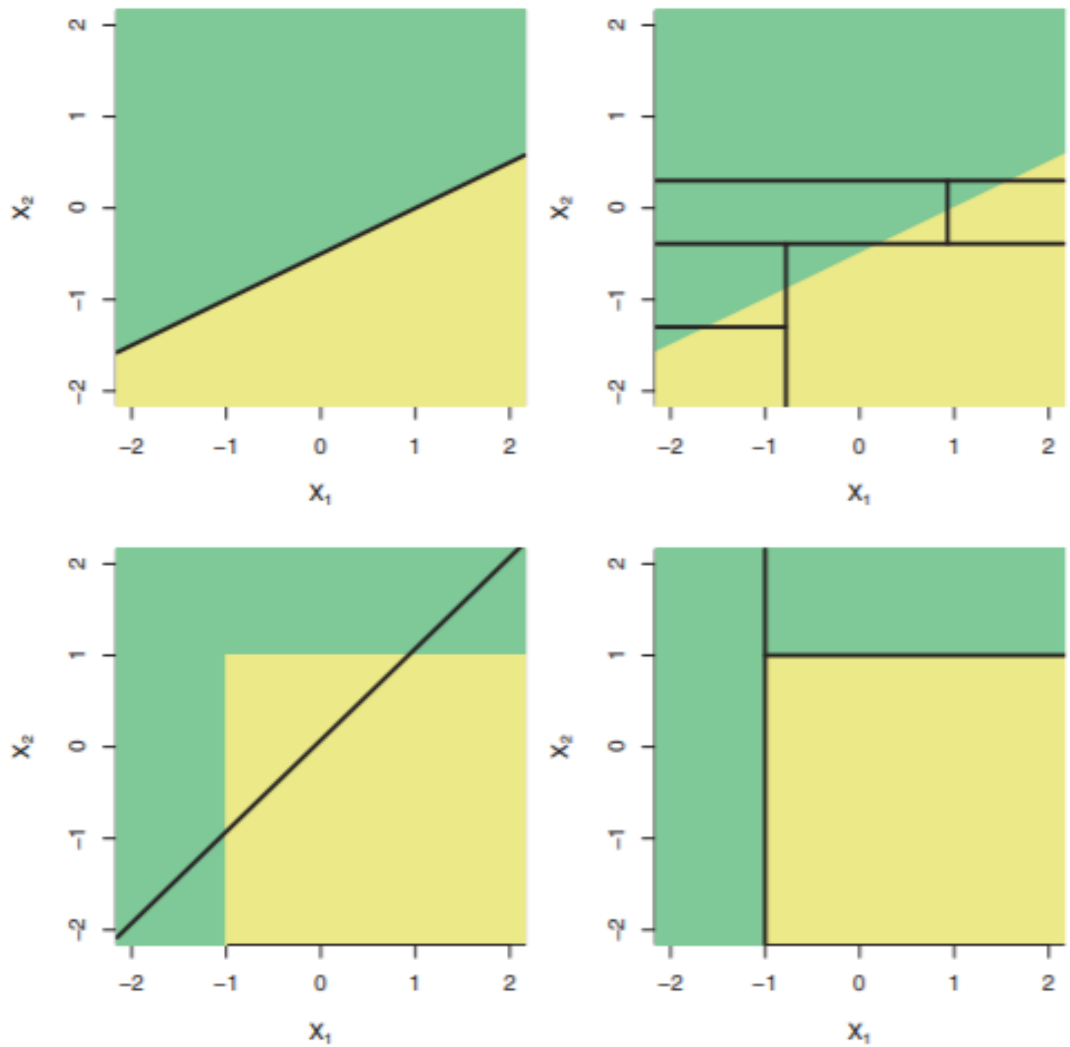
  - Choose among splits by InfoGain()

Decision Boundary

- Decision trees divide the feature space into axis-parallel (hyper-)rectangles
- Each rectangular region is labeled with one label

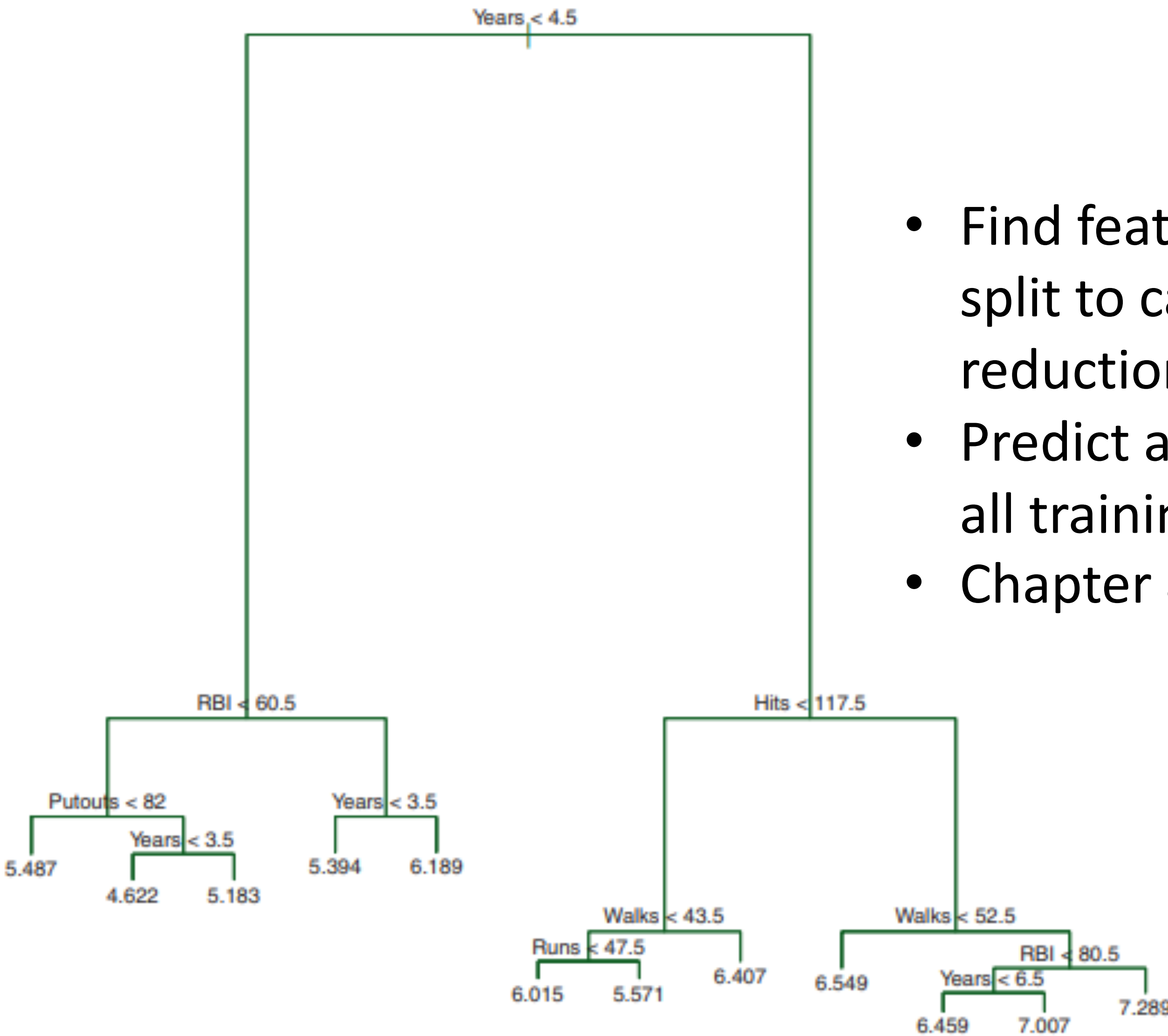


Decision Trees vs Linear Models



Linear model　　　　Decision tree

Regression Trees

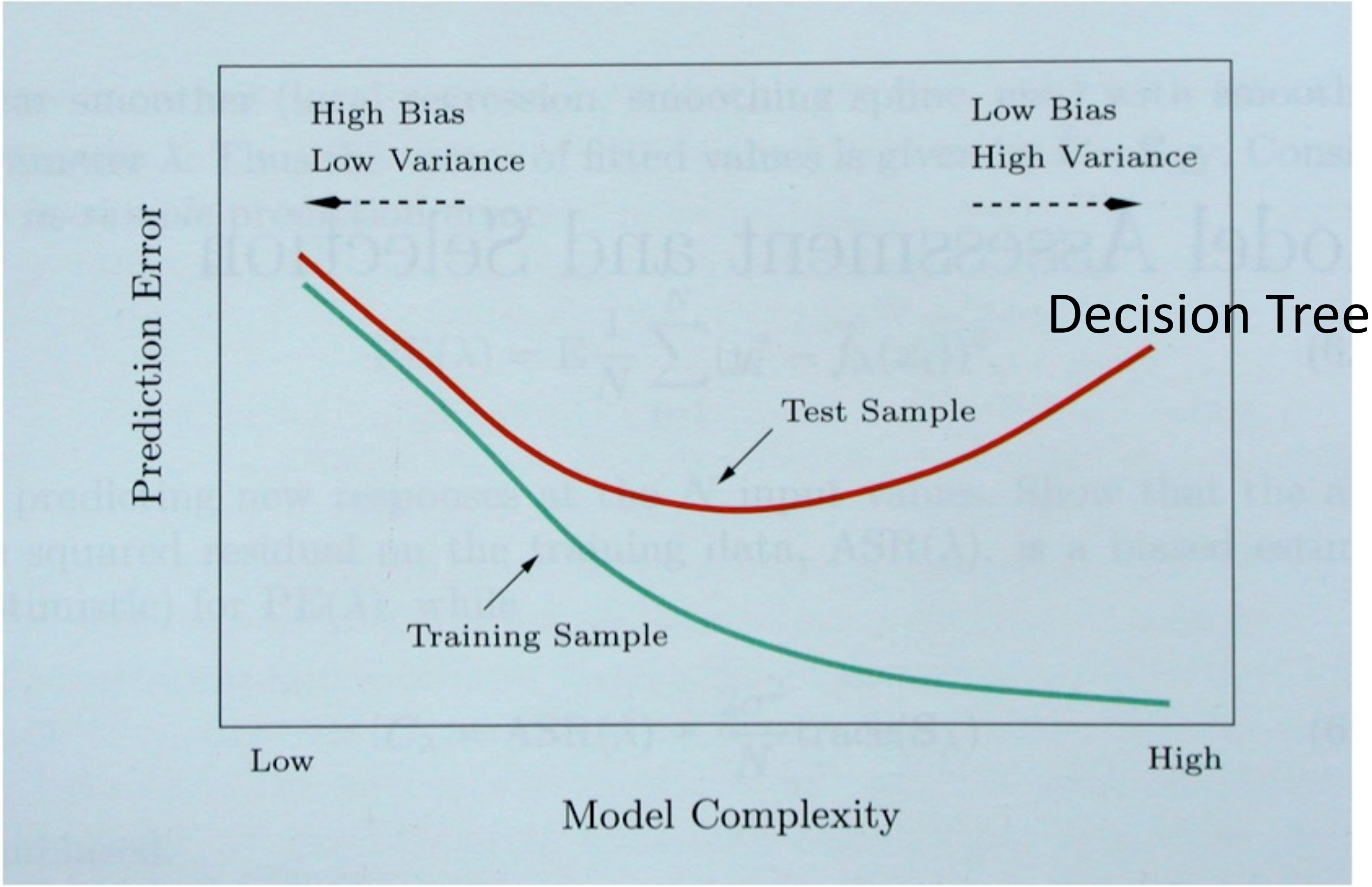


- Find feature and value to split to cause the maximum reduction in MSE
- Predict average response of all training data at each leaf
- Chapter 8.1 from textbook

Summary Decision Trees

- Greedy method for training
  - Not based on optimization or probabilities
- Uses impurity metric (e.g., information gain or Gini index) for splitting
- Advantages
  - Interpretability of decisions
- Limitations
  - Decision trees are prone to overfitting
  - Can be addressed by pruning or using ensembles of decision trees

Bias/Variance Tradeoff



Decision Trees

Hastie, Tibshirani, Friedman "Elements of Statistical Learning" 2001

How to reduce variance of single decision tree?

Ensemble Learning

Consider a set of classifiers h_1, ..., $\mathrm{h_L}$

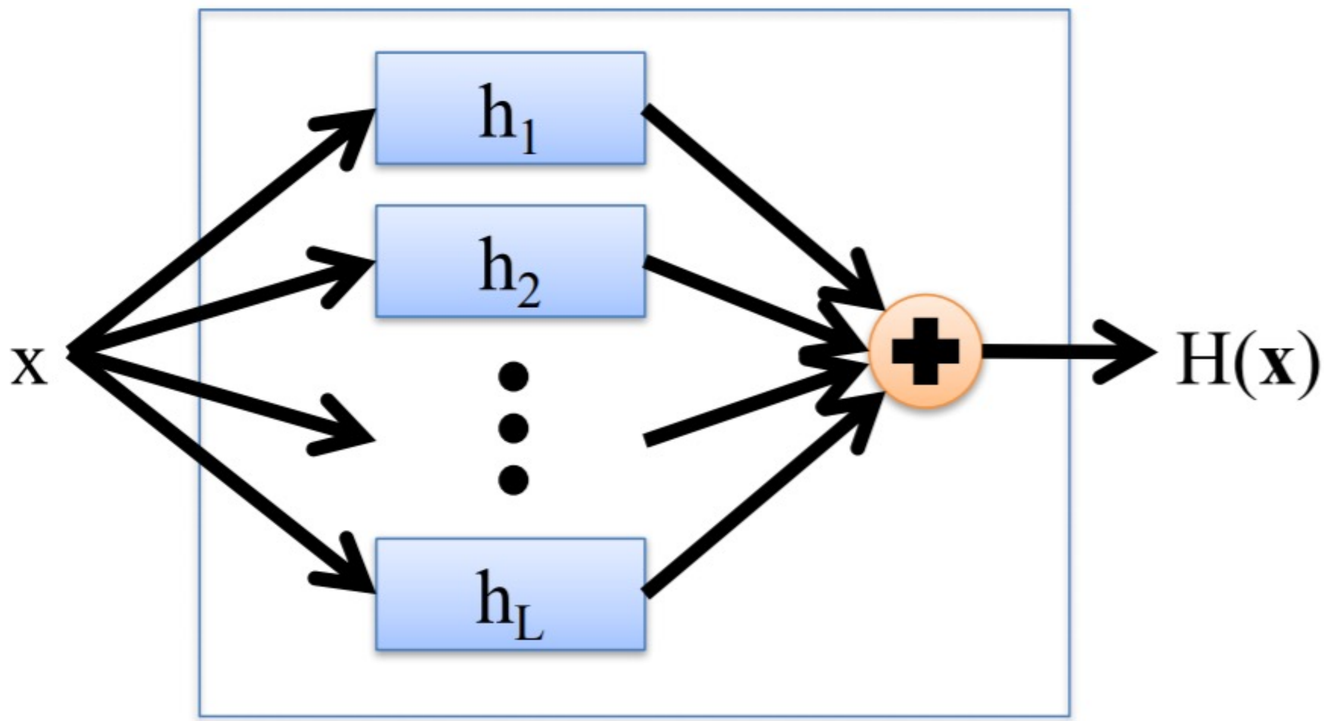
Idea: construct a classifier $H(\mathbf{x})$ that combines the individual decisions of h_1, ..., $\mathrm{h_L}$

- e.g., could have the member classifiers vote, or
- e.g., could use different members for different regions of the instance space

Successful ensembles require **diversity**

- Classifiers should make different mistakes
- Can have different types of base learners

Combining Classifiers: Averaging



- Final hypothesis is a simple vote of the members

Practical Applications

Goal: predict how a user will rate a movie

- Based on the user's ratings for other movies
- and other peoples' ratings
- with no other information about the movies

This application is called "collaborative filtering"

Netflix Prize: $1M to the first team to do 10% better then Netflix' system (2007-2009)

Winner: BellKor's Pragmatic Chaos – an ensemble of more than 800 rating systems

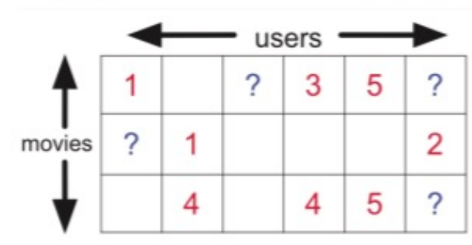
Netflix Prize

Machine learning competition with a $1 million prize


Leaderboard

Display top 20 leaders.

Rank	Team Name	Best Score	% Improvement	Last Submit Time
1	The Ensemble	0.8553	10.10	2009-07-26 18:38:22
2	[illegible]	[illegible]	10.09	2009-07-26 18:18:28
Grand Prize - RMSE <= 0.8563				
3	Grand Prize Team	0.8571	9.91	2009-07-24 13:07:49
4	Opera Solutions and Vandelay United	0.8573	9.89	2009-07-25 20:05:52
5	Vandelay Industries !	0.8579	9.83	2009-07-26 02:49:53
6	PragmaticTheory	0.8582	9.80	2009-07-12 15:09:53
7	BellKor in BigChaos	0.8590	9.71	2009-07-26 12:57:25
8	Dace_	0.8603	9.58	2009-07-24 17:18:43
9	Opera Solutions	0.8611	9.49	2009-07-26 18:02:08
10	BellKor	0.8612	9.48	2009-07-26 17:19:11
11	BigChaos	0.8613	9.47	2009-06-23 23:06:52
12	Feeds2	0.8613	9.47	2009-07-24 20:06:46
Progress Prize 2008 - RMSE = 0.8616 - Winning Team: BellKor in BigChaos				
13	xiangliang	0.8633	9.26	2009-07-21 02:04:40
14	Gravity	0.8634	9.25	2009-07-26 15:58:34
15	Ces	0.8642	9.17	2009-07-25 17:42:38
16	Invisible Ideas	0.8644	9.14	2009-07-20 03:26:12
17	Just a guy in a garage	0.8650	9.08	2009-07-22 14:10:42
18	Craig Carmichael	0.8656	9.02	2009-07-25 16:00:54
19	J Dennis Su	0.8658	9.00	2009-03-11 09:41:54
20	acmehill	0.8659	8.99	2009-04-16 06:29:35
Progress Prize 2007 - RMSE = 0.8712 - Winning Team: KorBell				
Cinematch score on quiz subset - RMSE = 0.9514				



Reduce error

- Suppose there are 25 base classifiers
- Each classifier has error rate, $\varepsilon = 0.35$
- Assume independence among classifiers
- Probability that the ensemble classifier makes a wrong prediction:

Reduce Variance

- Averaging reduces variance:

$$Var(\overline{X}) = \frac{Var(X)}{N}$$

(when predictions are **independent**)

Average models to reduce model variance

One problem:

only one training set

where do multiple models come from?

How to Achieve Diversity

How to Achieve Diversity

- Avoid overfitting
  - Vary the training data
- Features are noisy
  - Vary the set of features

Two main ensemble learning methods

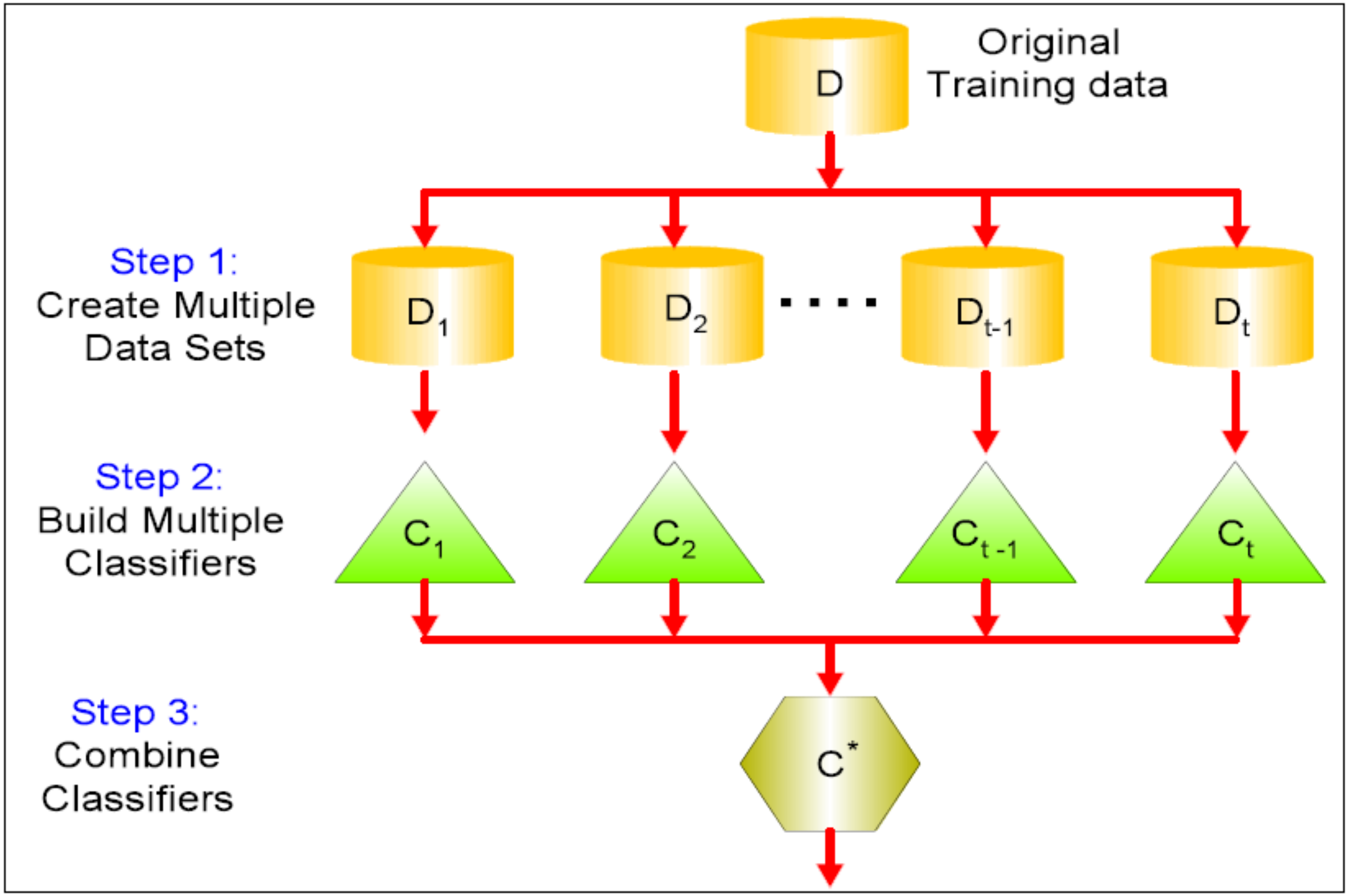
- Bagging (e.g., Random Forests)
- Boosting (e.g., AdaBoost)

Bagging

- Leo Breiman (1994)
- Take repeated bootstrap samples from training set D
- *Bootstrap sampling*: Given set D containing N training examples, create D' by drawing N examples at random with replacement from D.

- Bagging:
  - Create k bootstrap samples $D_1 \ldots D_k$.
  - Train distinct classifier on each D_i.
  - Classify new instance by majority vote / average.

General Idea



Majority Votes

Example of Bagging

- Sampling with replacement

Training Data

Data ID

Original Data	1	2	3	4	5	6	7	8	9	10
Bagging (Round 1)	7	8	10	8	2	5	10	10	5	9
Bagging (Round 2)	1	4	9	1	2	3	2	7	3	2
Bagging (Round 3)	1	8	5	10	5	5	9	6	3	7

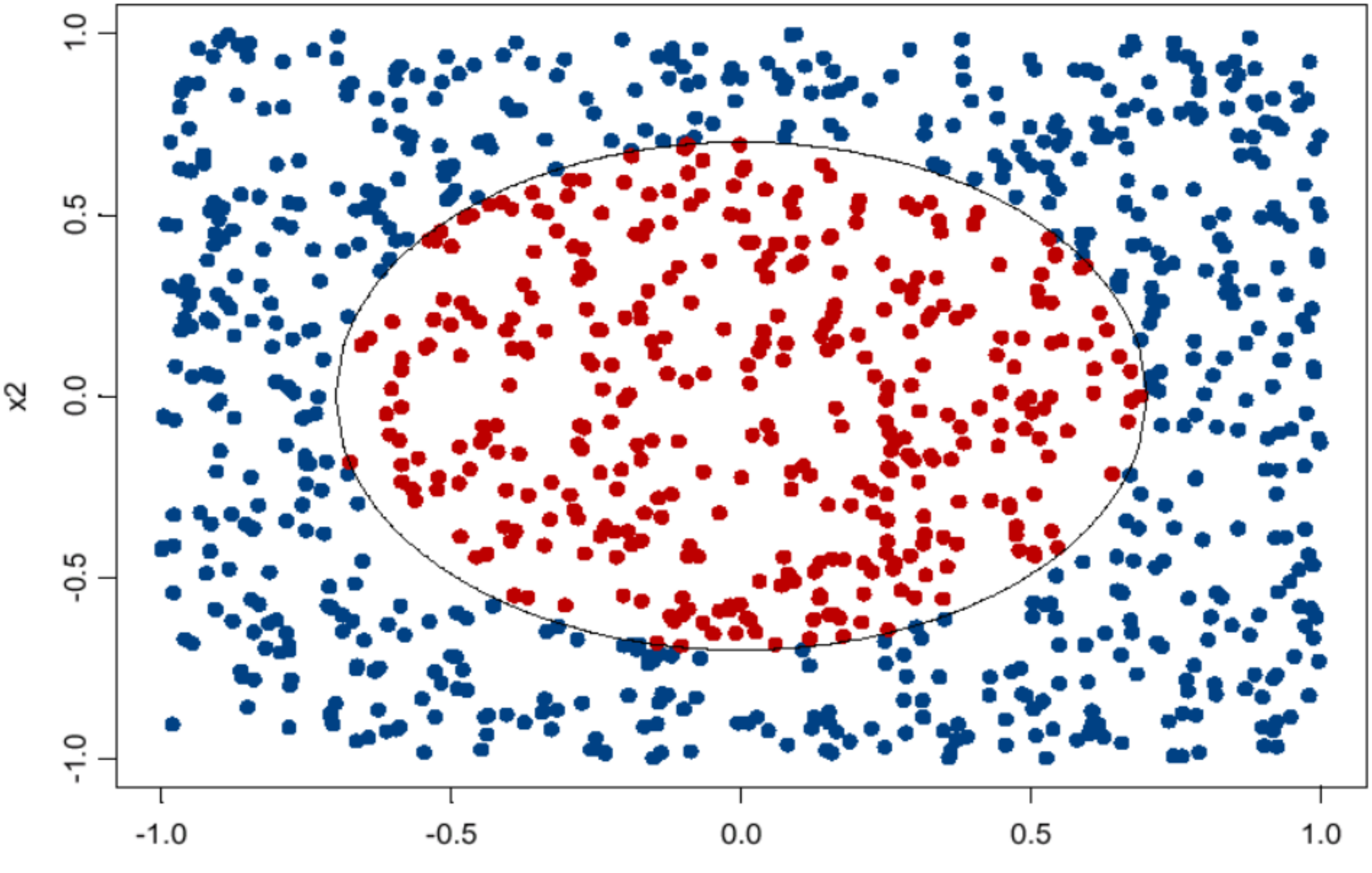
- Sample each training point with probability 1/n
- Out-Of-Bag (OOB) observation: point not in sample

Bootstrap Samples

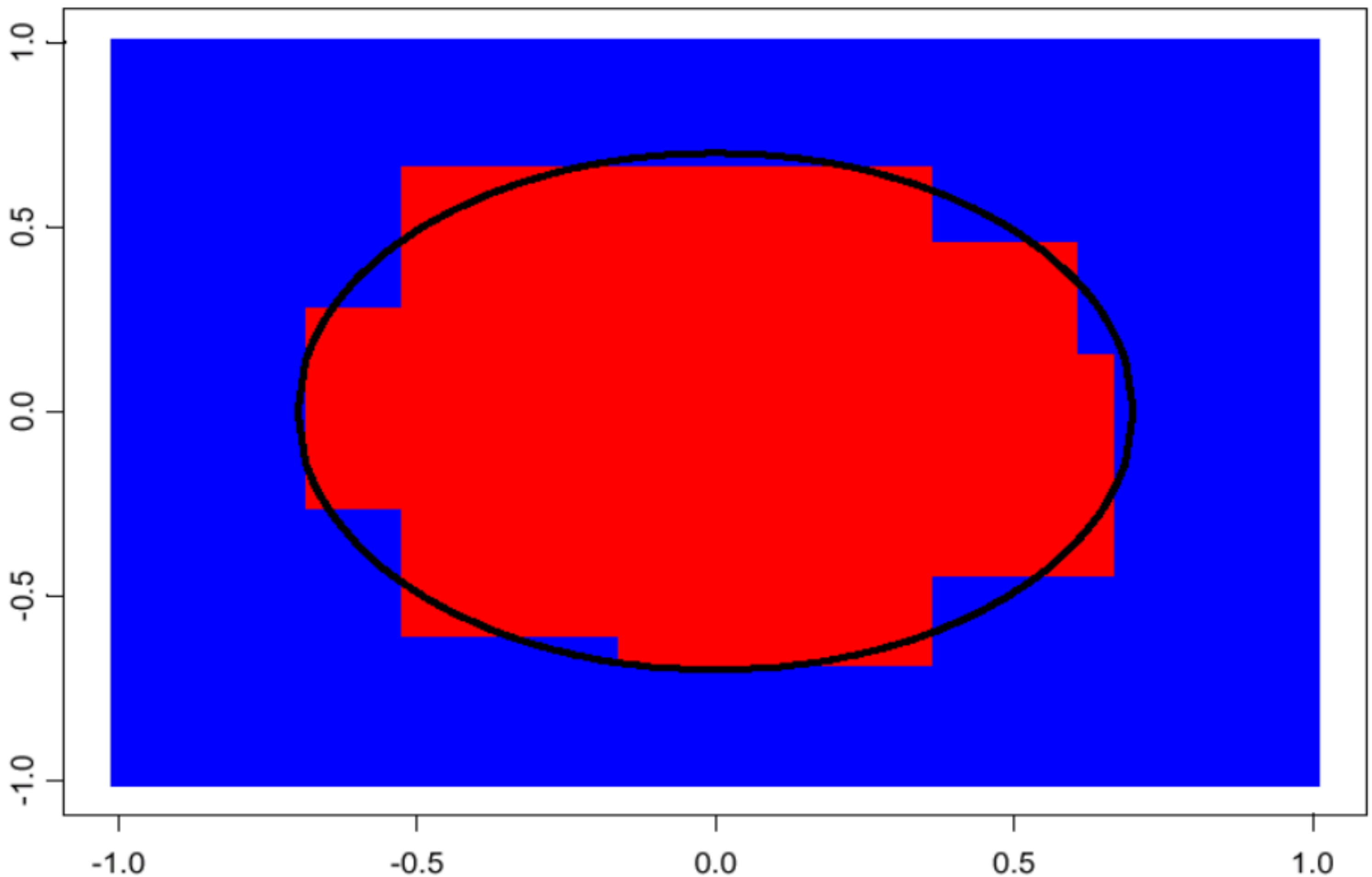
Bagging

- Can be applied to multiple classification models
- Very successful for decision trees
  - Decision trees have high variance
  - Don’t prune the individual trees, but grow trees to full extent
  - Precision accuracy of decision trees improved substantially
- OOB average error used instead of Cross Validation

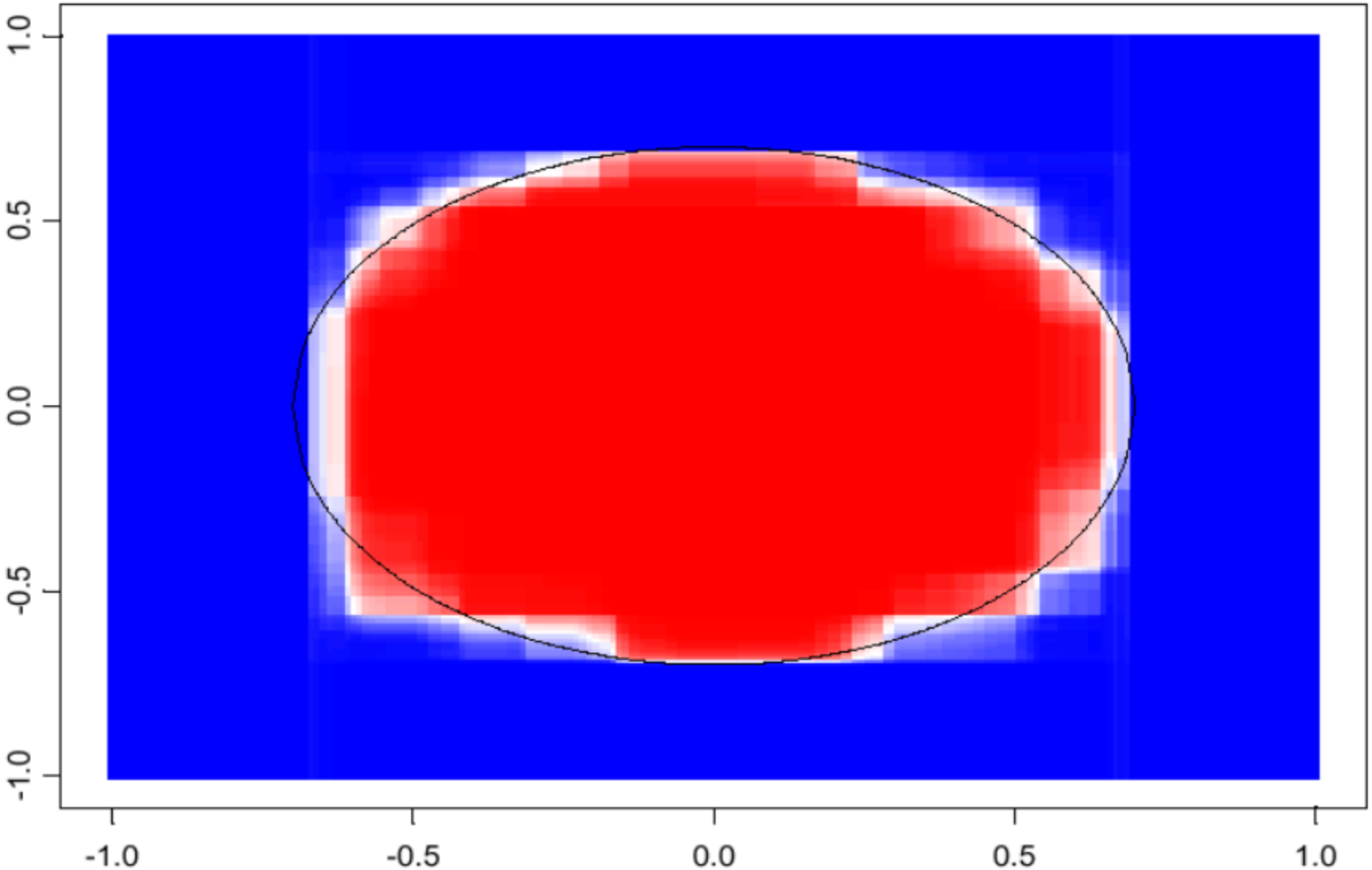
Example Distribution



Decision Tree Decision Boundary



100 Bagged Trees



shades of blue/red indicate strength of vote for particular classification

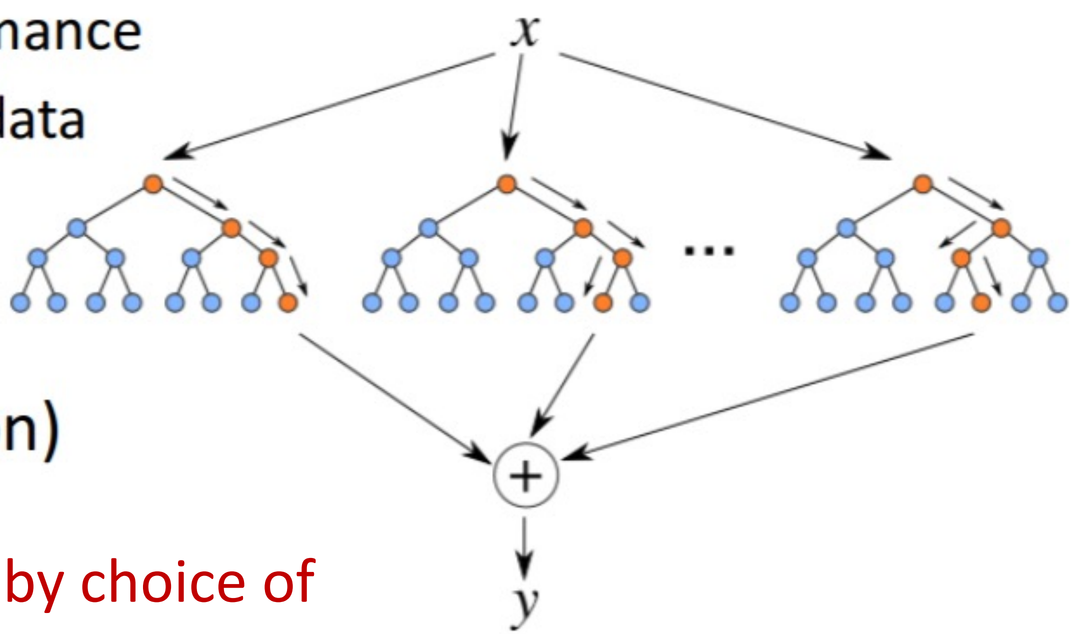
Random Forests

- Ensemble method specifically designed for decision tree classifiers

- Introduce two sources of randomness: "Bagging" and "Random input vectors"
  - Bagging method: each tree is grown using a bootstrap sample of training data
  - Random vector method: At each node, best split is chosen from a random sample of *m* attributes instead of all attributes

Random Forests

- Construct decision trees on bootstrap replicas
  - Restrict the node decisions to a small subset of features picked randomly for each node
- Do not prune the trees
  - Estimate tree performance on out-of-bootstrap data
- Average the output of all trees (or choose mode decision)



Trees are de-correlated by choice of random subset of features

Random Forest Algorithm

1. For $b = 1$ to B:

   (a) Draw a $\boxed{\text{bootstrap sample}}$ $\mathbf{Z}^*$ of size N from the training data.

   (b) Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached.

      i. Select $\boxed{m \text{ variables at random}}$ from the p variables.

      ii. Pick the best variable/split-point among the m.

      iii. Split the node into two daughter nodes.

2. Output the ensemble of trees $\{T_b\}_1^B$.

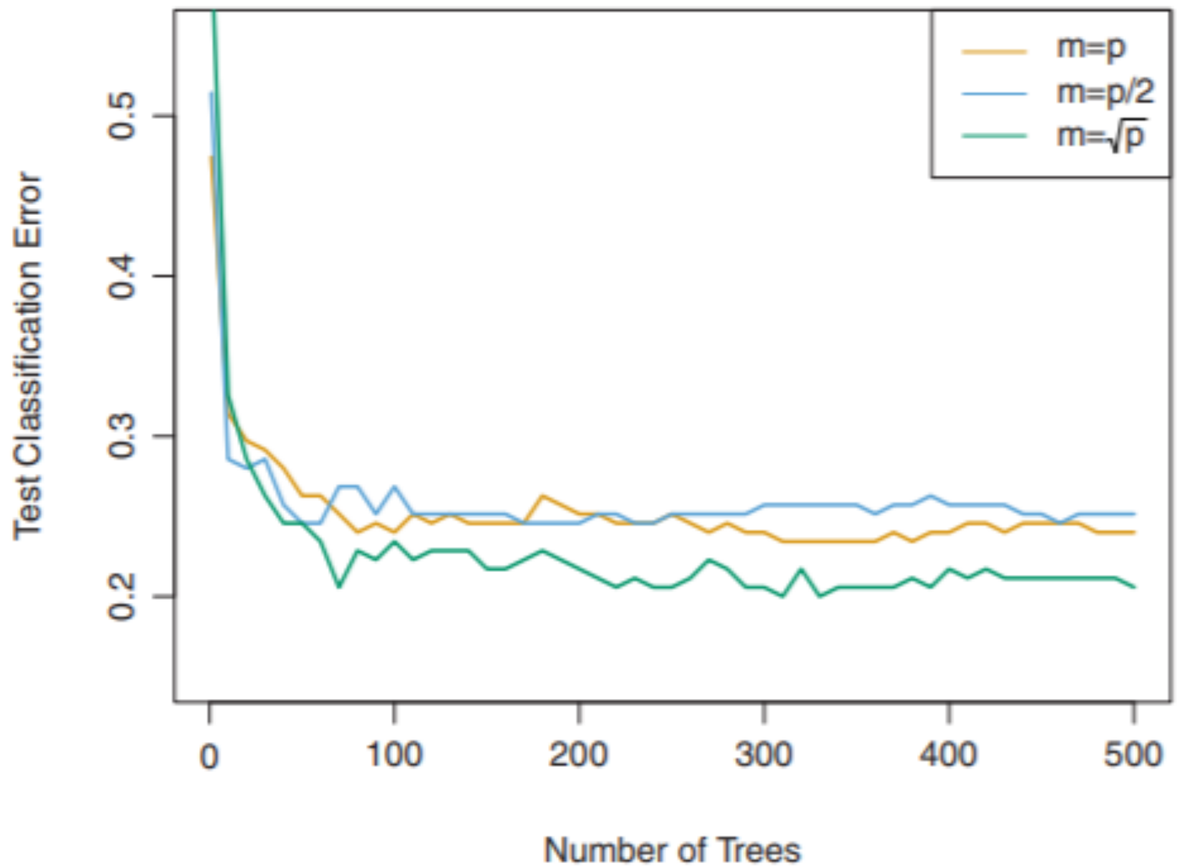
To make a prediction at a new point x:

Regression: $\hat{f}_{\mathrm{rf}}^B(x) = \frac{1}{B}\sum_{b=1}^B T_b(x)$.

Classification: Let $\hat{C}_b(x)$ be the class prediction of the bth random-forest tree. Then $\hat{C}_{\mathrm{rf}}^B(x) = \textit{majority vote}\ \{\hat{C}_b(x)\}_1^B$.

If m=p, this is equivalent to Bagging
with Decision Trees as base learner

Effect of Number of Predictors



- p = total number of predictors; m = predictors chosen in each split
- Random Forests uses $m = \sqrt{p}$

Variable Importance

- Ensemble of trees looses somewhat interpretability of decision trees
- Which variables contribute mostly to prediction?
- Random Forests computes a Variable Importance metric per feature
  - For each tree in the ensemble, consider the split by the particular feature
  - How much information gain / Gini index decreases after the split
  - Average over all trees

Variable Importance Plots

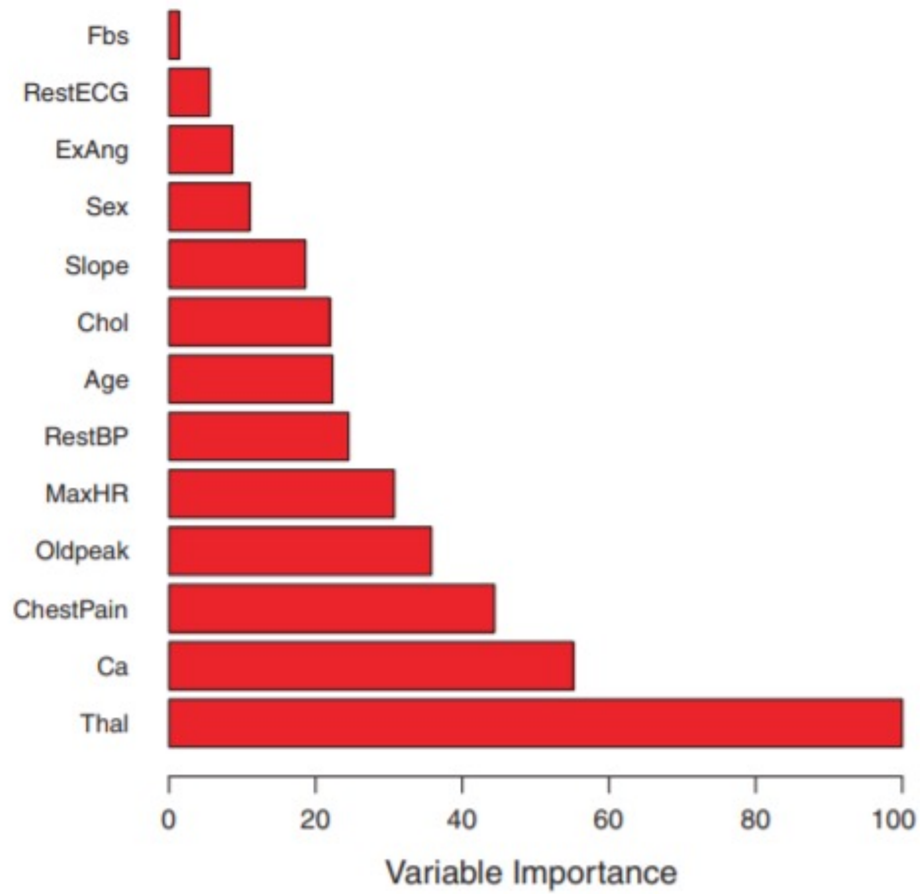


FIGURE 8.9. *A variable importance plot for the* `Heart` *data. Variable importance is computed using the mean decrease in Gini index, and expressed relative to the maximum.*

Variable Importance

- Ensembles of trees loose in interpretability
  – Variable importance helps with determining important features
- Can be used as a filter method for feature selection
  – Train Random Forest model
  – Compute variable importance
  – Select top k features by highest important
  – Train other models with the k features