CS249: ADVANCED DATA MINING

Decision Trees, Regression Trees, and Random Forest

Instructor: Yizhou Sun

yzsun@cs.ucla.edu

April 12, 2017

Announcements

- Course Project
  - Team formation due today
- Homework 1
  - Be out around tomorrow
- PTE will be decided by this weekend
- Office hour for TA (Jae LEE):
  - 3-5 PM on Tuesdays @BH 2432

Methods to Learn: Last Lecture

	Vector Data	Text Data	Recommender System	Graph & Network
Classification	Decision Tree; Naïve Bayes; **Logistic Regression** SVM; NN			Label Propagation
Clustering	K-means; hierarchical clustering; DBSCAN; Mixture Models; kernel k-means	PLSA; LDA	Matrix Factorization	SCAN; Spectral Clustering
Prediction	**Linear Regression** **GLM**		Collaborative Filtering	
Ranking				PageRank
Feature Representation		Word embedding		Network embedding

Methods to Learn

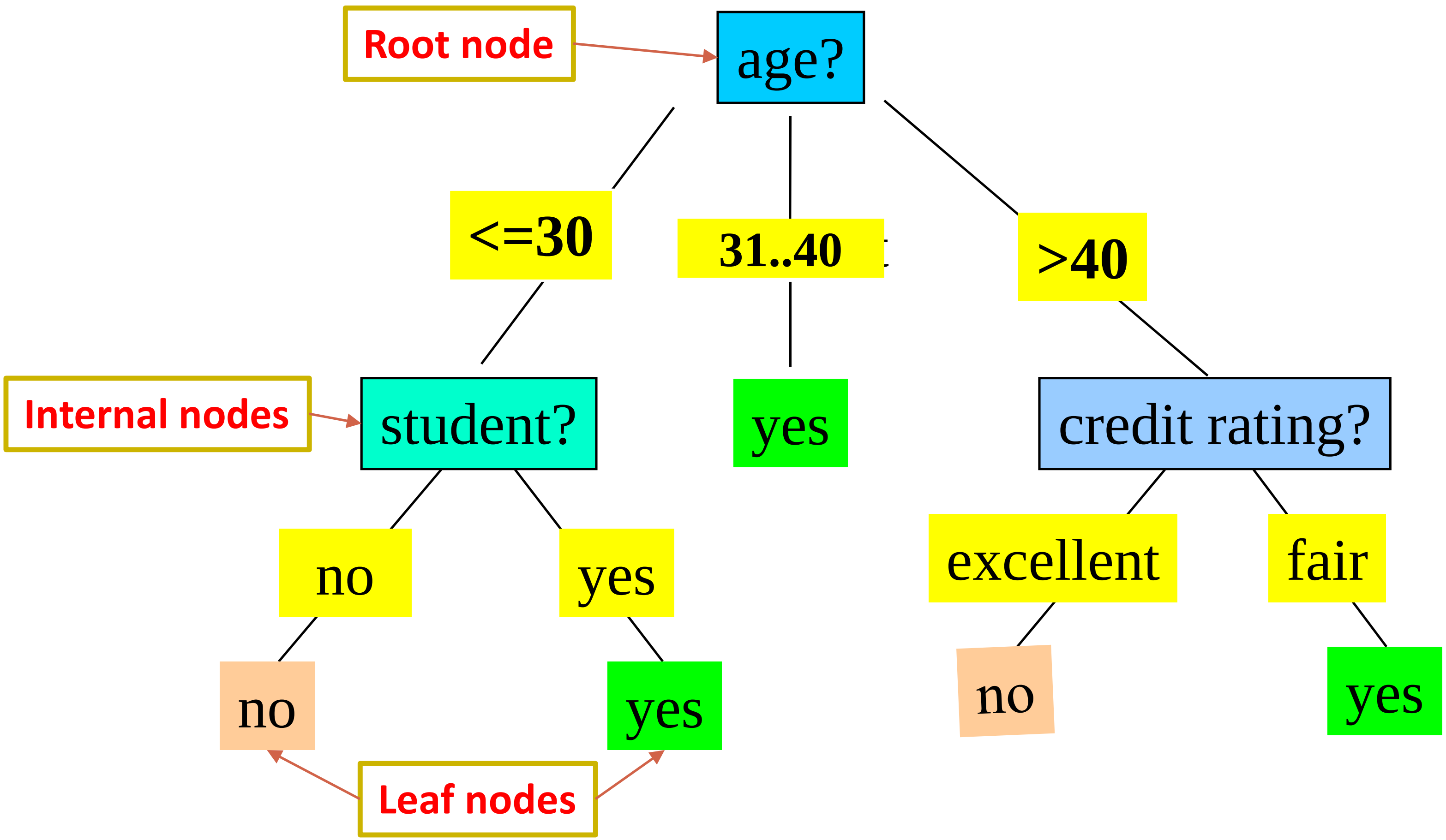
	Vector Data	Text Data	Recommender System	Graph & Network
Classification	**Decision Tree**; Naïve Bayes; **Logistic Regression** SVM; NN			Label Propagation
Clustering	K-means; hierarchical clustering; DBSCAN; Mixture Models; kernel k-means	PLSA; LDA	Matrix Factorization	SCAN; Spectral Clustering
Prediction	**Linear Regression** **GLM**		Collaborative Filtering	
Ranking				PageRank
Feature Representation		Word embedding		Network embedding

Vector Data: Trees

- Tree-based Prediction and Classification
- Classification Trees
- Regression Trees
- Random Forest
- Summary

Tree-based Models

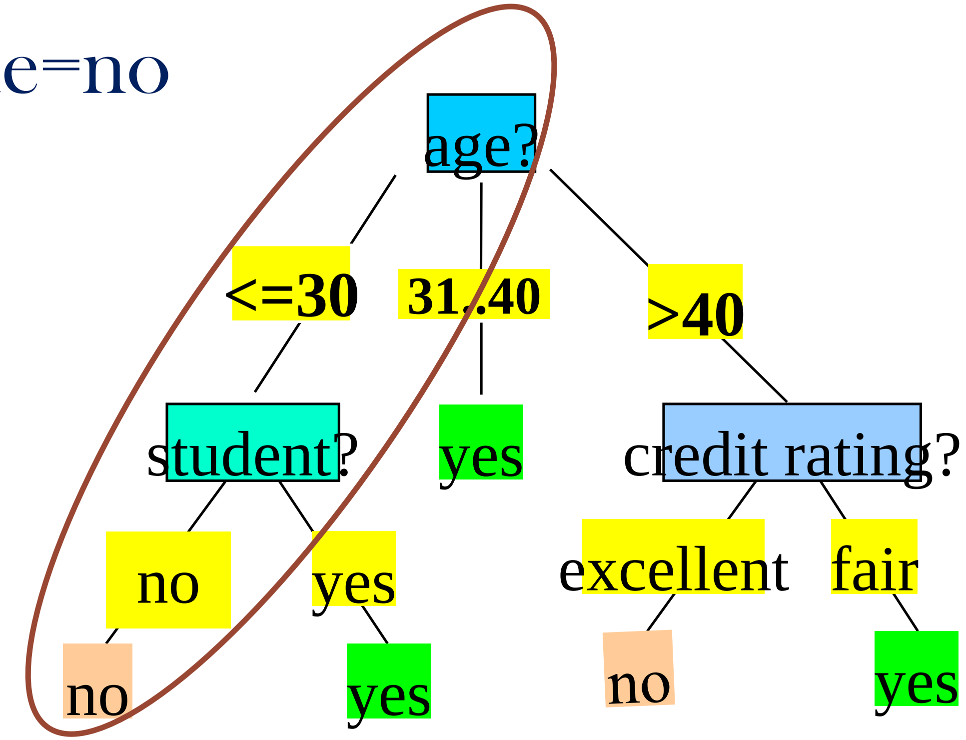
- Use trees to partition the data into different regions and make predictions



Easy to Interpret

- A path from root to a leaf node corresponds to a rule
  - E.g., **if** age<=30 and student=no

    then target value=no

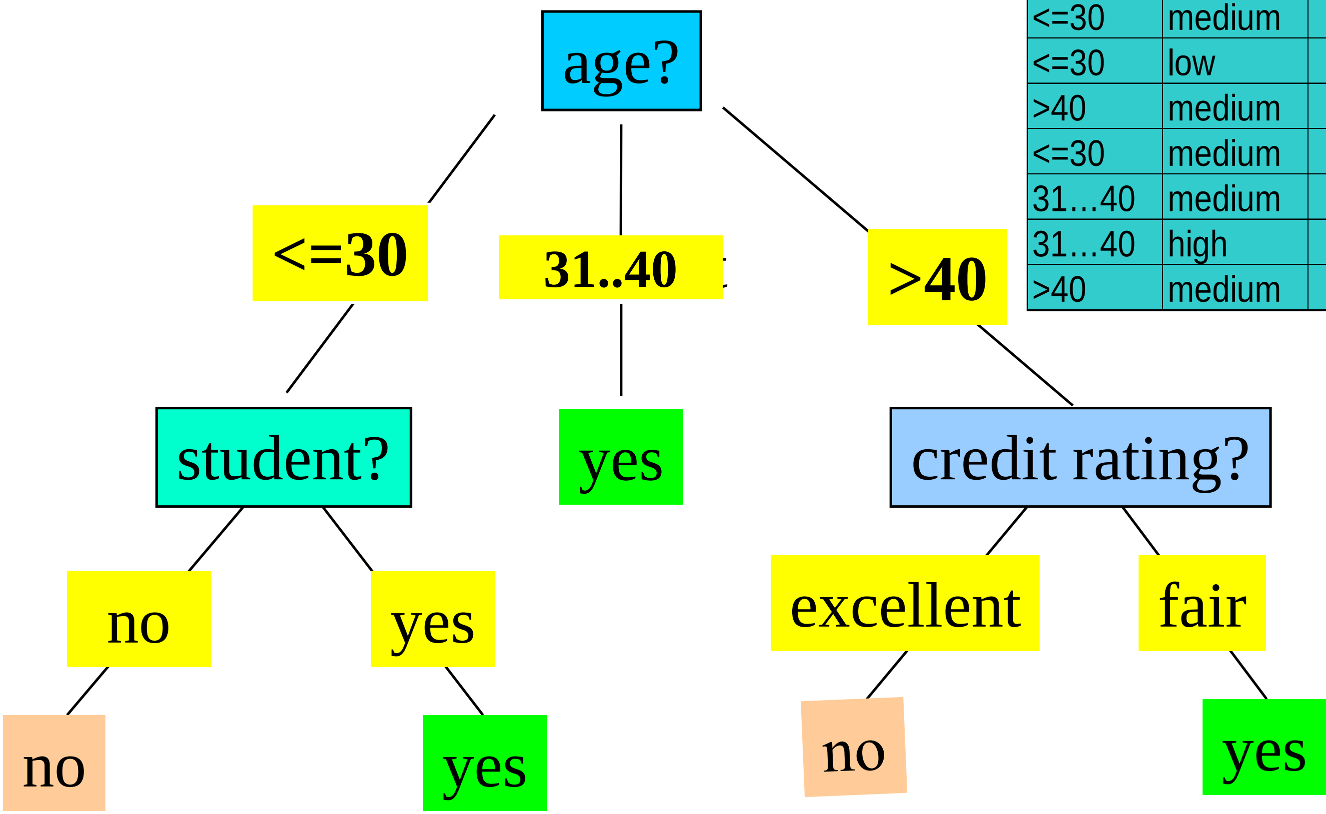


Vector Data: Trees

- Tree-based Prediction and Classification
- Classification Trees
- Regression Trees
- Random Forest
- Summary

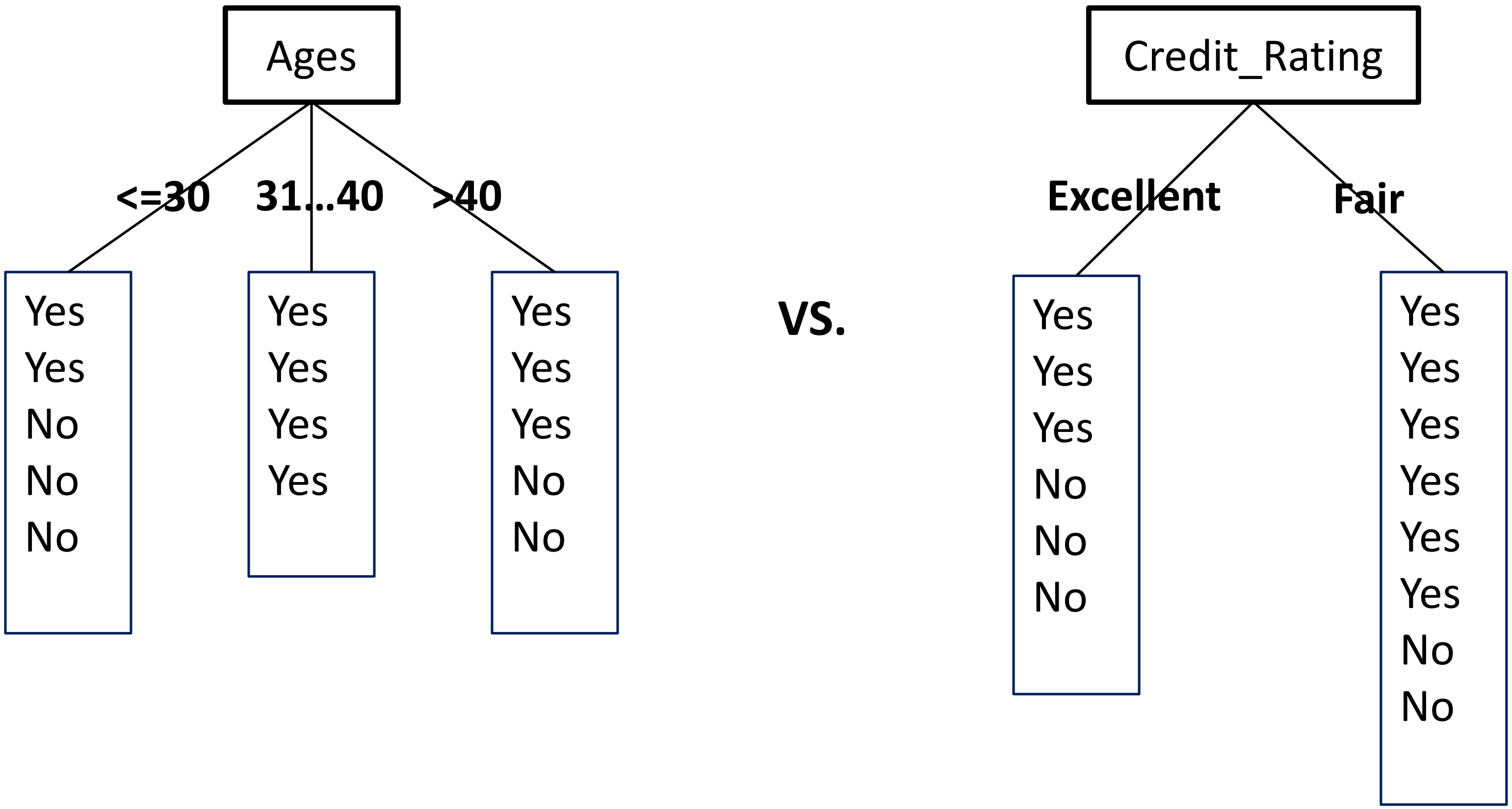
Decision Tree Induction: An Example

- Training data set: Buys_xbox
- The data set follows an example of Quinlan's ID3 (Playing Tennis)
- Resulting tree:



age	income	student	credit_rating	buys_Xbox
<=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

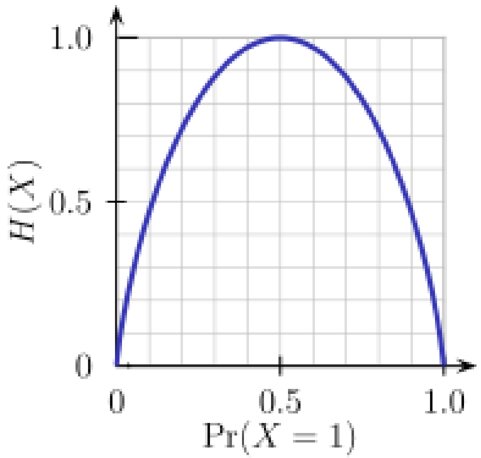
How to choose attributes?



Q: Which attribute is better for the task?

Brief Review of Entropy

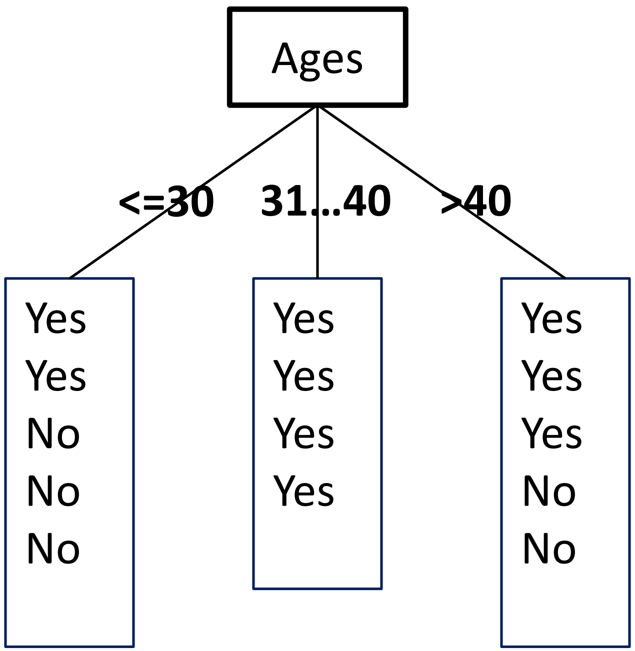
- Entropy (Information Theory)
  - A measure of uncertainty (impurity) associated with a random variable
  - Calculation: For a discrete random variable Y taking m distinct values $\{y_1, \dots, y_m\}$,
    - $H(Y) = -\sum_{i=1}^{m} p_i \log(p_i)$, where $p_i = P(Y = y_i)$
  - Interpretation:
    - Higher entropy => higher uncertainty
    - Lower entropy => lower uncertainty



m = 2

Conditional Entropy

- How much uncertainty of Y if we know an attribute X?
  - $H(Y|X) = \sum_x p(x) H(Y|X = x)$



Weighted average of entropy at each branch!

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 (conditional entropy):

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

- Information gained by branching on attribute A

$$Gain(A) = Info(D) - Info_A(D)$$

Attribute Selection: Information Gain

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

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

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

age	income	student	credit_rating	buys_xbox
<=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

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

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

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

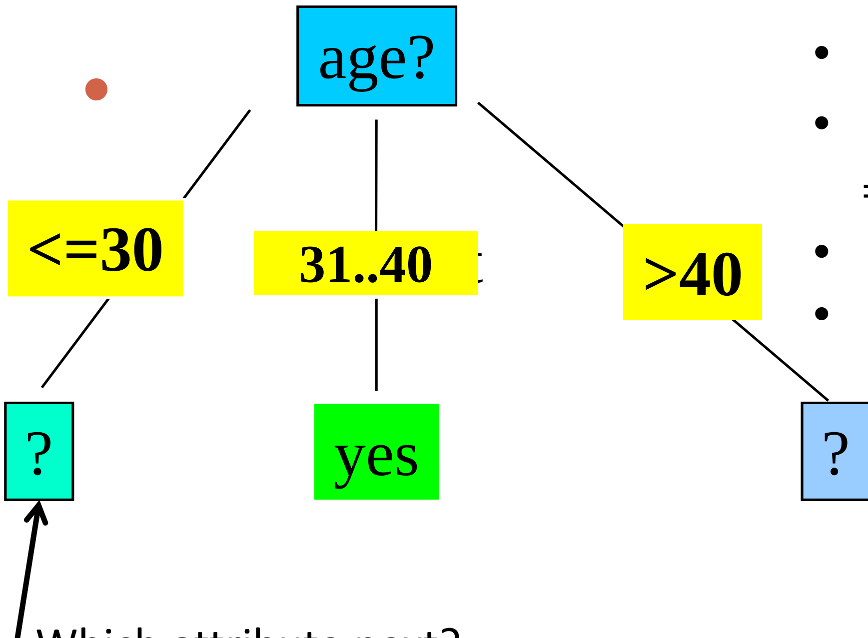
Similarly,

$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

$$Gain(credit\_rating) = 0.048$$

Attribute Selection for a Branch

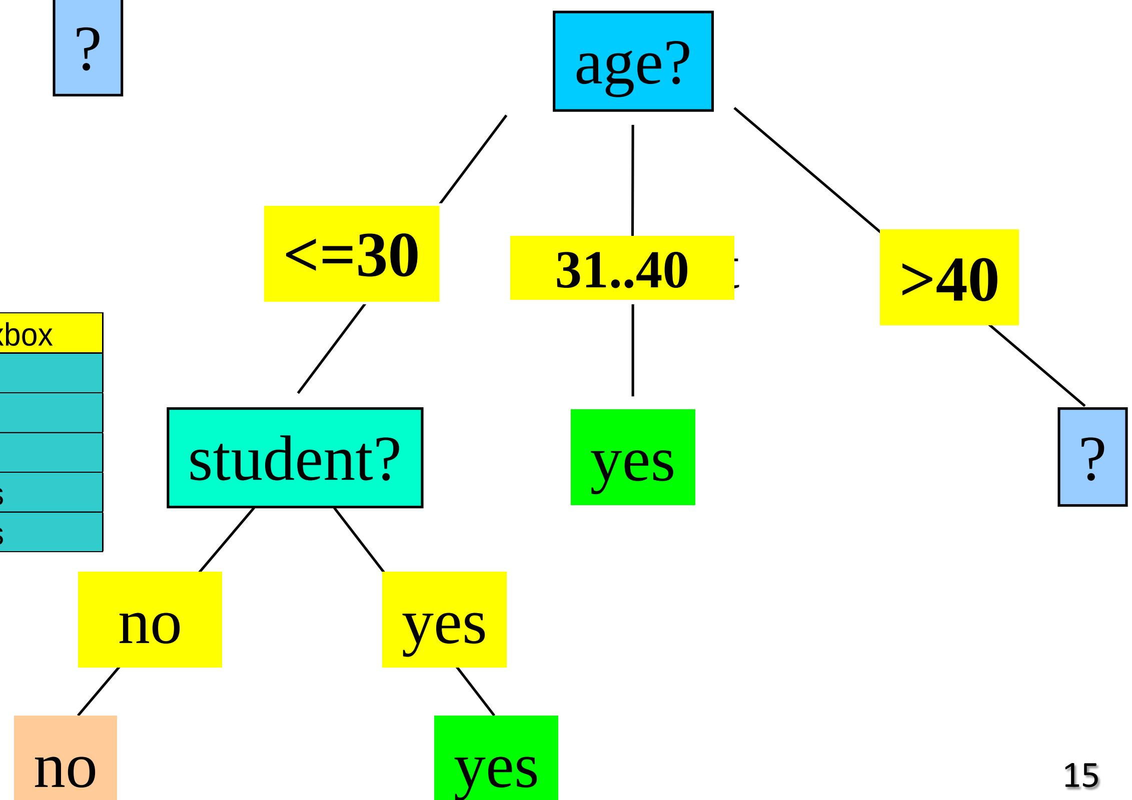


Which attribute next?

- $Info(D_{age\leq30}) = -\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5} = 0.971$
- $Gain_{age\leq30}(income)$
  $= Info(D_{age\leq30}) - Info_{income}(D_{age\leq30}) = 0.571$
- $Gain_{age\leq30}(student) = 0.971$
- $Gain_{age\leq30}(credit\_rating) = 0.02$

age	income	student	credit_rating	buys_xbox
<=30	high	no	fair	no
<=30	high	no	excellent	no
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
<=30	medium	yes	excellent	yes

$D_{age\leq30}$



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 – use majority voting in the parent partition

Computing Information-Gain for Continuous-Valued Attributes

- Let attribute A be a continuous-valued attribute
- Must determine the *best split point* for A
  - Sort the value A in increasing order
  - Typically, the midpoint between each pair of adjacent values is considered as a possible *split point*
    - $(a_i+a_{i+1})/2$ is the midpoint between the values of a_i and a_{i+1}
  - The point with the *minimum expected information requirement* for A is selected as the split-point for A
- Split:
  - D1 is the set of tuples in D satisfying A ≤ split-point, and D2 is the set of tuples in D satisfying A > split-point

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)

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

  - GainRatio(A) = Gain(A)/SplitInfo(A)
- Ex.

$$SplitInfo_{income}(D) = -\frac{4}{14} \times \log_2\left(\frac{4}{14}\right) - \frac{6}{14} \times \log_2\left(\frac{6}{14}\right) - \frac{4}{14} \times \log_2\left(\frac{4}{14}\right) = 1.557$$

  - gain_ratio(income) = 0.029/1.557 = 0.019
- The attribute with the maximum gain ratio is selected as the splitting attribute

Gini Index (CART, IBM IntelligentMiner)

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

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

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*)

Computation of Gini Index

- Ex. D has 9 tuples in buys_computer = "yes" and 5 in "no"

$$gini(D) = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right)^2 = 0.459$$

- Suppose the attribute income partitions D into 10 in D_1: {low, medium} and 4 in D_2: {high}

$$gini_{income \in \{low, medium\}}(D) = \left(\frac{10}{14}\right) Gini(D_1) + \left(\frac{4}{14}\right) Gini(D_2)$$

$$= \frac{10}{14}\left(1 - \left(\frac{7}{10}\right)^2 - \left(\frac{3}{10}\right)^2\right) + \frac{4}{14}\left(1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2\right)$$

$$= 0.443$$

$$= Gini_{income \in \{high\}}(D).$$

$Gini_{\{low,high\}}$ is 0.458; $Gini_{\{medium,high\}}$ is 0.450. Thus, split on the {low,medium} (and {high}) since it has the lowest Gini index

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 (why?)
  - **Gini index:**
    - biased to multivalued attributes

*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-statistic: 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 both (1) encode the tree, and (2) encode the exceptions to the tree
- Multivariate splits (partition based on multiple variable combinations)
  - CART: finds multivariate splits based on a linear comb. of attrs.
- Which attribute selection measure is the best?
  - Most give good results, none is significantly superior than others

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
- 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 validation dataset to decide which is the "best pruned tree"

Vector Data: Trees

- Tree-based Prediction and Classification
- Classification Trees
- Regression Trees
- Random Forest
- Summary

From Classification to Regression

- Target variable
  - From categorical variable to continuous variable
- Attribute selection criterion
  - Measure the purity of continuous target variable in each partition
- Leaf node
  - A simple model for that partition, e.g., average

Attribute Selection

- Reduction of Variance
- For attribute A, weighted average variance

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

$$Var(D_j) = \sum_{y \in D_j} (y - \bar{y})^2 / |D_j|,$$

$$\text{where } \bar{y} = \sum_{y \in D_j} y \,/ |D_j|$$

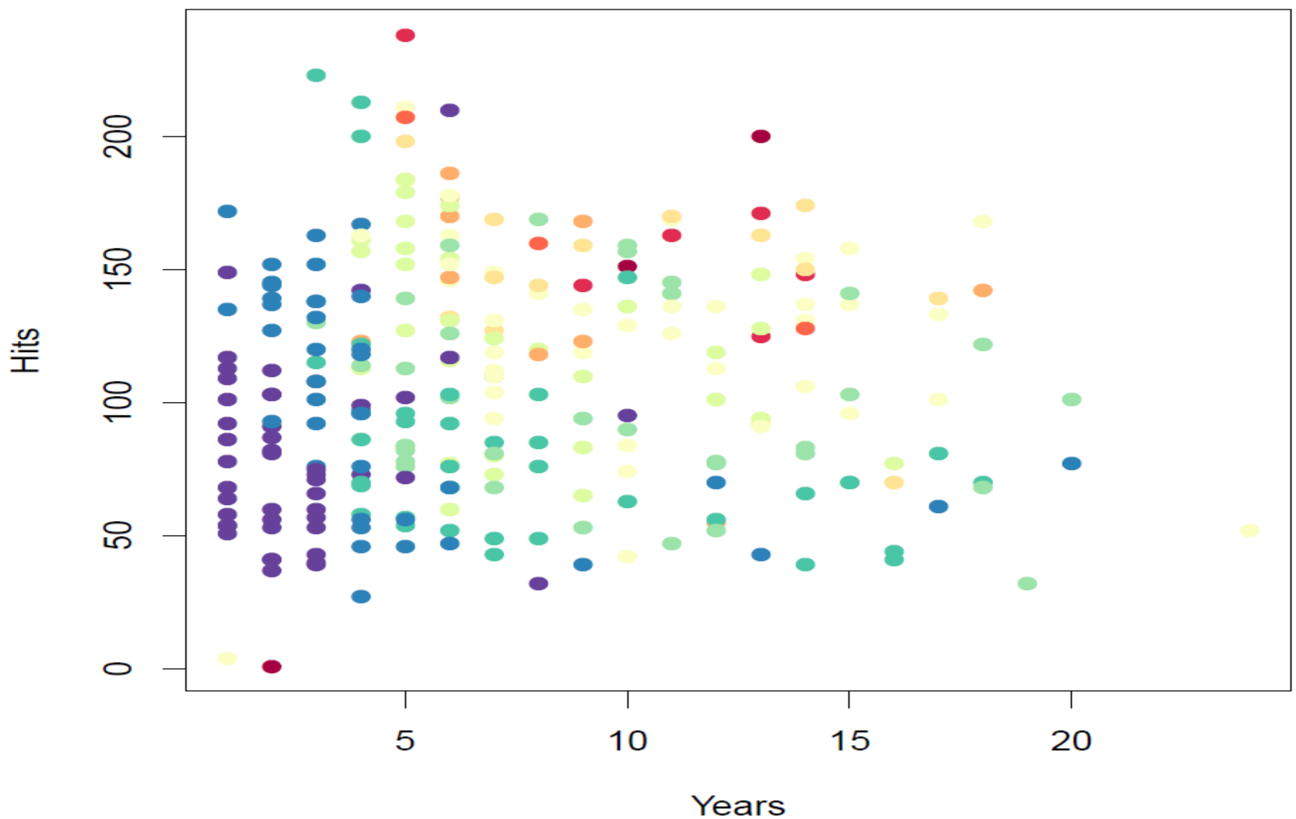
- Pick the attribute with the lowest weighted average variance

Leaf Node Model

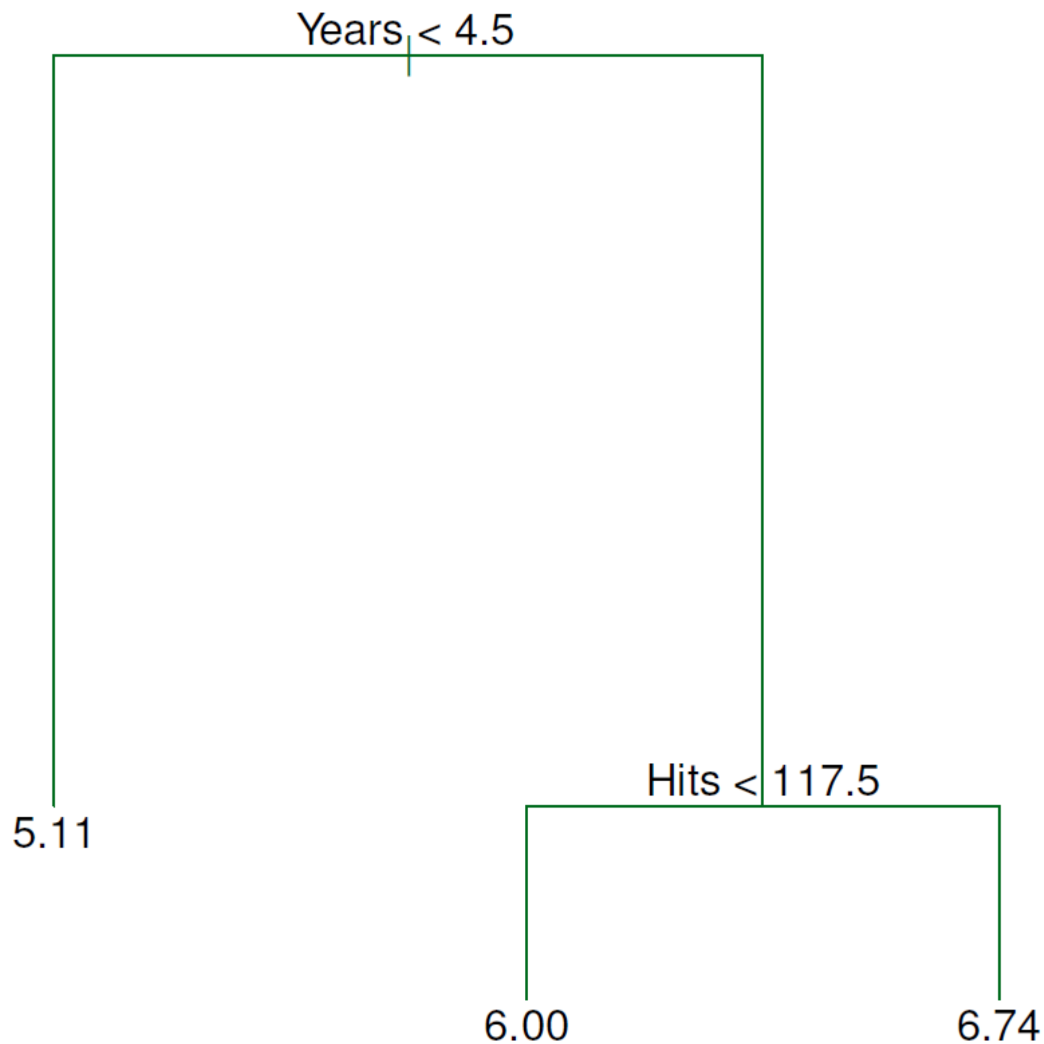
- Take the average of the partition for leave node *l*
  - $\widehat{y_l} = \sum_{y \in D_l} y \,/ |D_l|$

Example: Predict Baseball Player Salary

- Dataset: (years, hits)=>Salary
  - Colors indicate value of salary (blue: low, red: high)

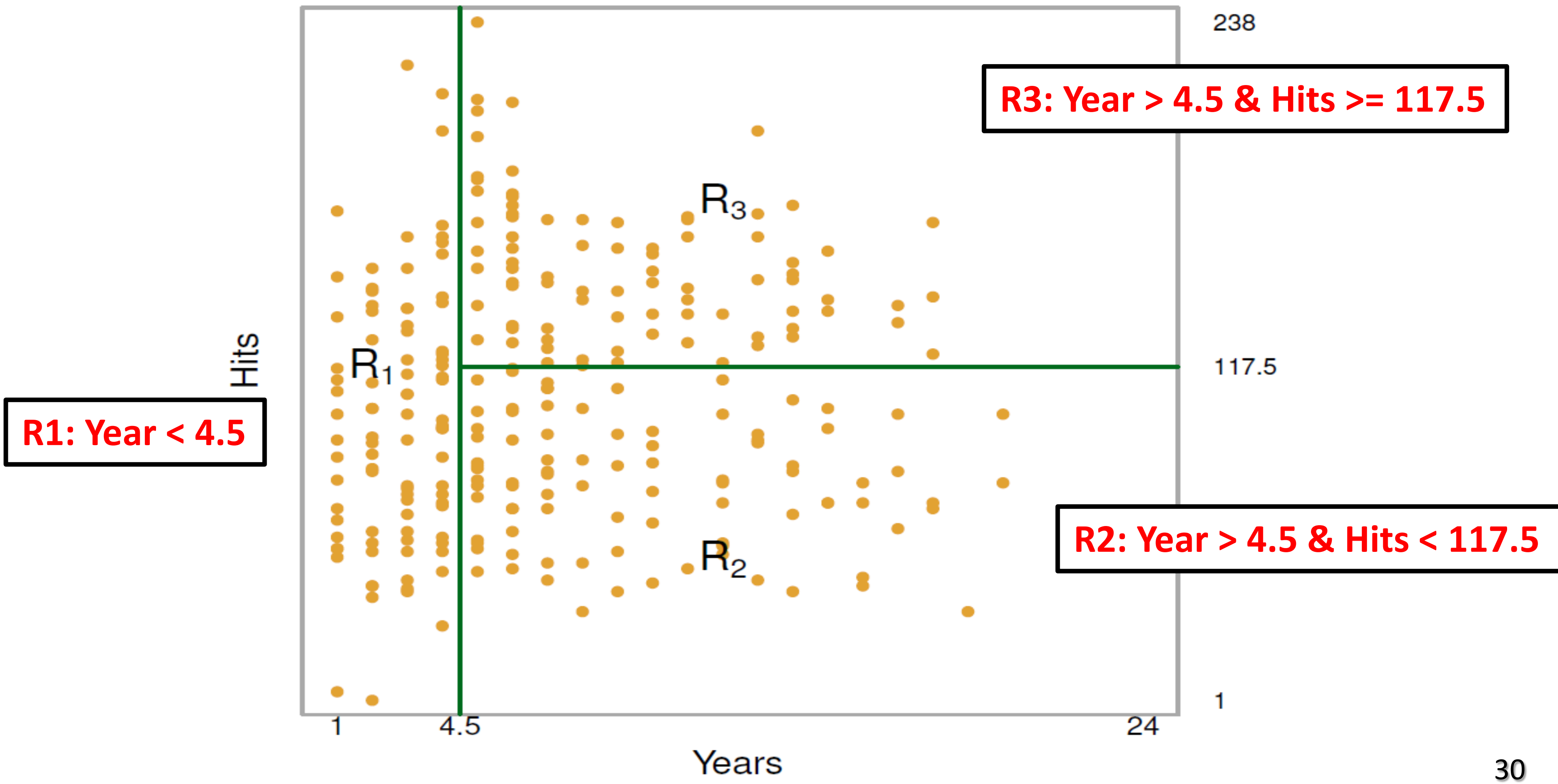


A Regression Tree Built



A Different Angle to View the Tree

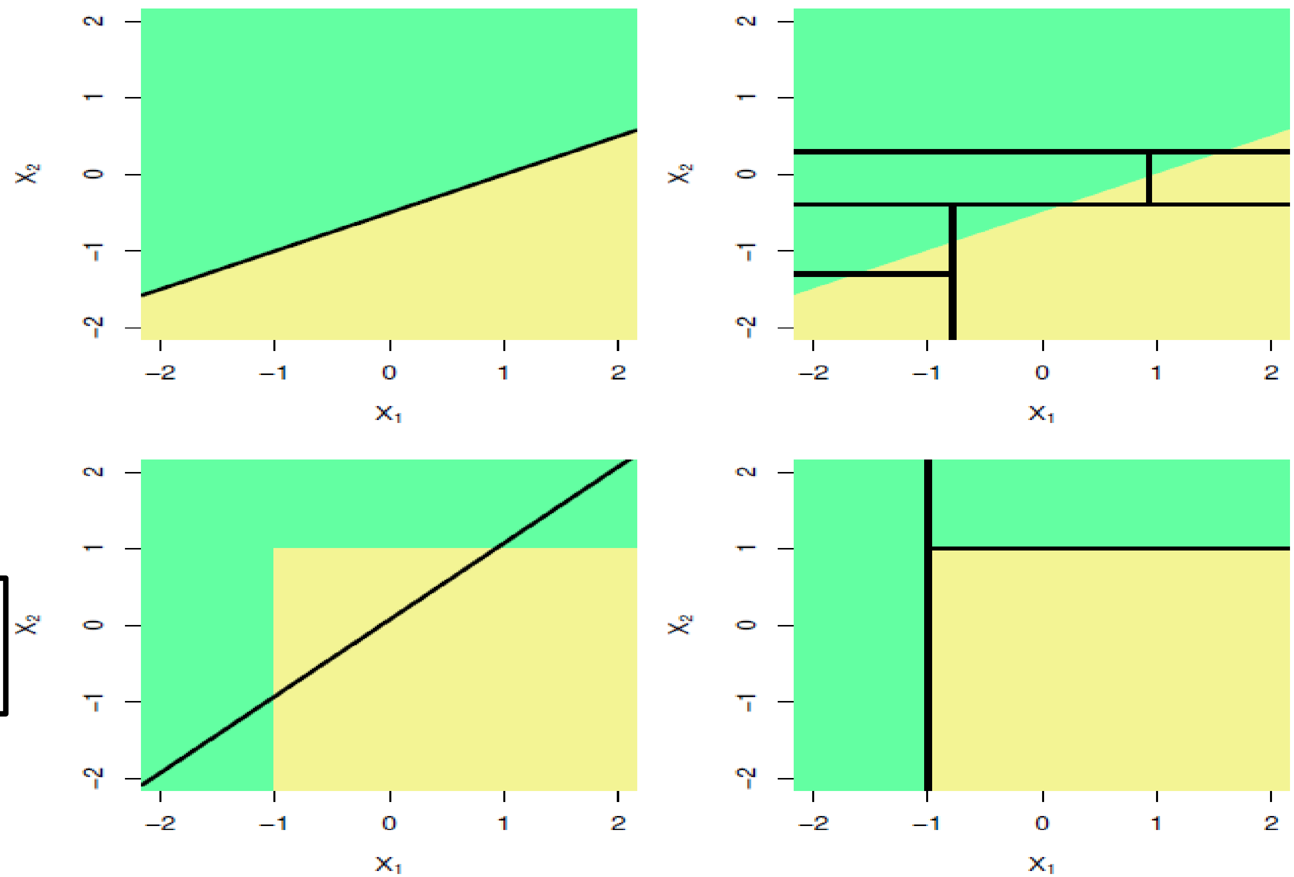
- A leaf is corresponding to a box in the plane



Trees vs. Linear Models

Ground Truth:
Linear Boundary

Ground Truth:
Non-Linear Boundary



Fitted Model:
Linear Model

Fitted Model:
Trees

Vector Data: Trees

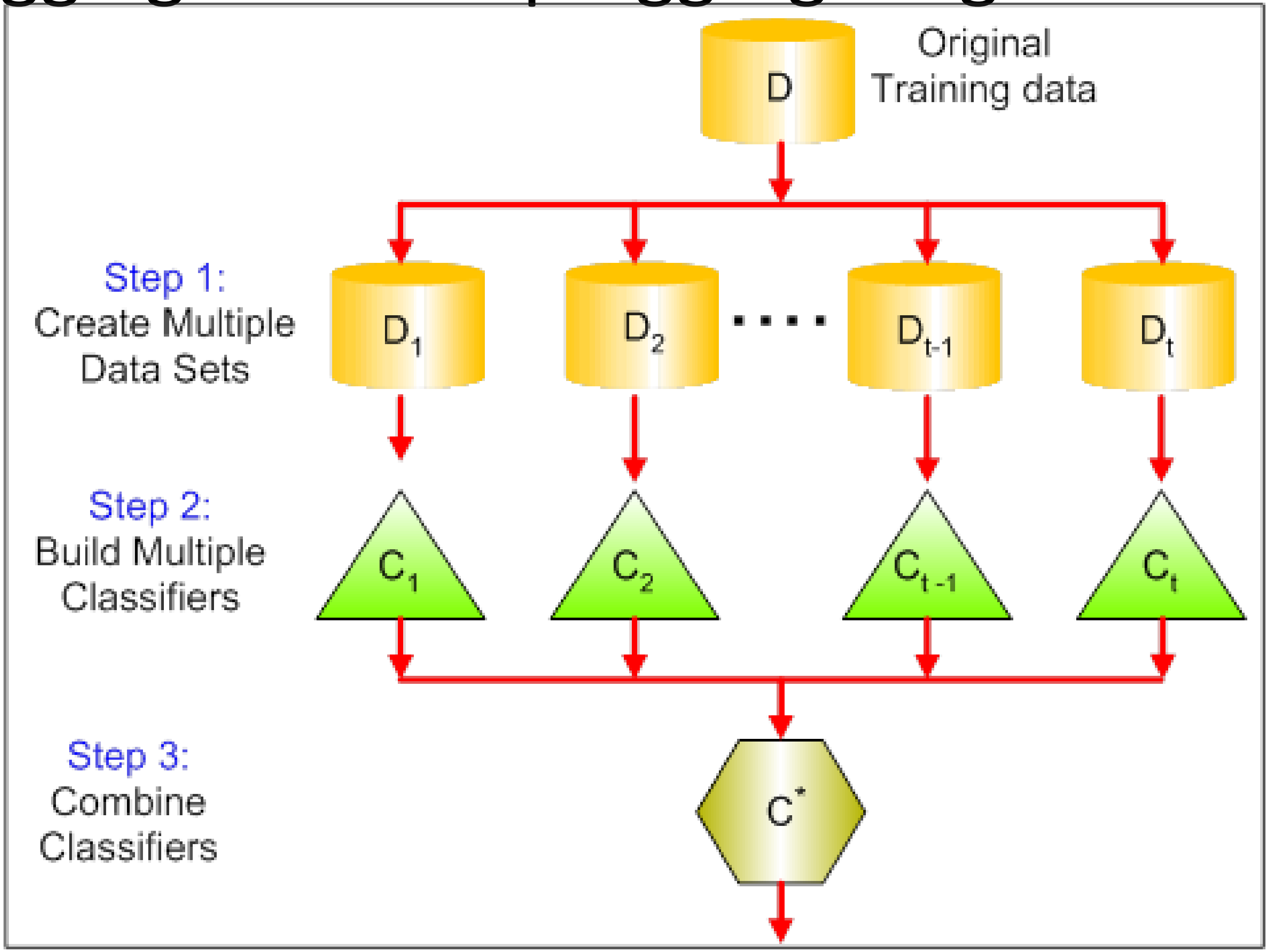
- Tree-based Prediction and Classification
- Classification Trees
- Regression Trees
- Random Forest
- Summary

A Single Tree or a Set of Trees?

- Limitation of single tree
  - Accuracy is not very high
  - Overfitting
- A set of trees
  - The idea of ensemble

The Idea of Bagging

- Bagging: Bootstrap Aggregating



Why It Works?

- Each classifier produces the prediction
  - $f_i(x)$

- The error will be reduced if we use the average of multiple classifiers
  - $var\left(\frac{\sum_i f_i(x)}{t}\right) = var(f_i(x))/t$

Random Forest

- **Sample t times data collection**: random sample with replacement for objects, $n' \leq n$
- **Sample p' variables**: Select a subset of variables for each data collection, e.g., $p' = \sqrt{p}$
- **Construct t trees** for each data collection using selected subset of variables

- **Aggregate the prediction** results for new data
  - Majority voting for classification
  - Average for prediction

Properties of Random Forest

- Strengths
  - Good accuracy for classification tasks
  - Can handle large-scale of dataset
  - Can handle missing data to some extent
- Weaknesses
  - Not so good for predictions tasks
  - Lack of interpretation

Vector Data: Trees

- Tree-based Prediction and Classification
- Classification Trees
- Regression Trees
- Random Forest
- Summary

Summary

- Classification Trees
  - Predict categorical labels, information gain, tree construction
- Regression Trees
  - Predict numerical variable, variance reduction
- Random Forest
  - A set of trees, bagging