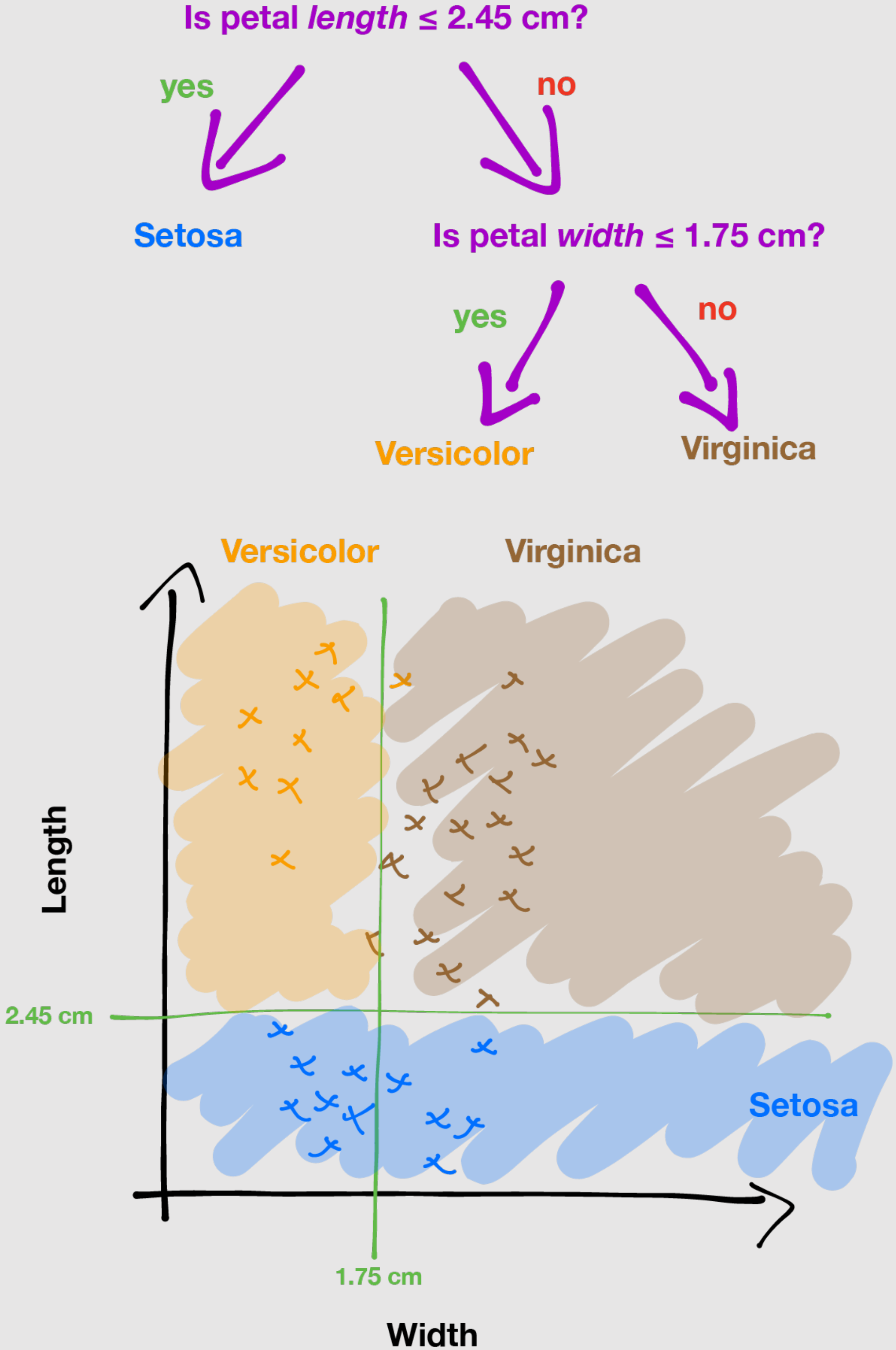


Supervised Learning: Decision Tree Models

Roadmap

- Classification vs Prediction
  - Classification Process
- Decision Tree Model
  - Model
  - ID3 Algorithm
- More Thoughts
- Deeper Perspectives
  - Decision boundary
  - Algorithmic implementation
  - More complicated decision boundary
  - Extensions of DT
- Take-home Messages!

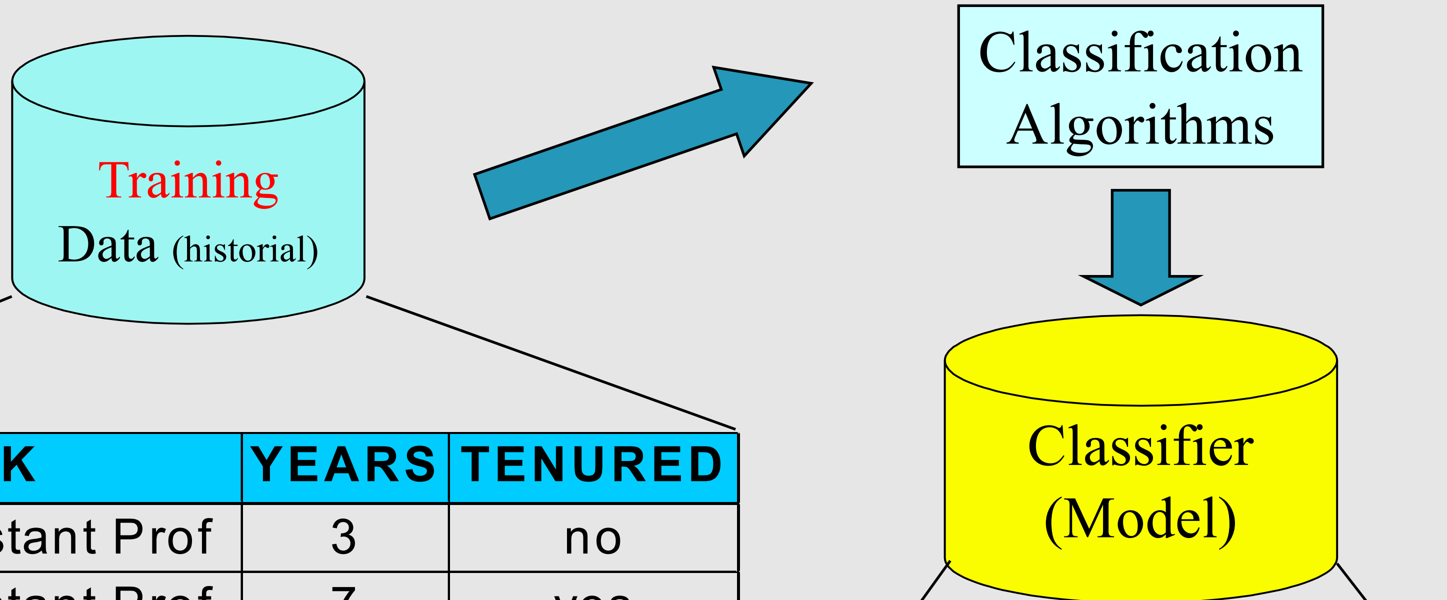
Classification vs. Prediction

- Classification:
  - predicts categorical class labels
  - classifies data (constructs a model) based on the training set and the values (class labels) in a classifying attribute and uses it in classifying new data
- Prediction:
  - models continuous-valued functions, i.e., predicts unknown or missing values
- Typical Applications
  - credit approval
  - target marketing
  - medical diagnosis

Classification—A Two-Step Process

- *Model construction*: describing a set of predetermined classes
  - Each tuple/sample is assumed to belong to a predefined class, as determined by the class label attribute
  - The set of tuples used for model construction: training set
  - The model is represented as classification rules, decision trees, layered neural networks or mathematical formulae
- *Model usage*: for classifying future or unknown objects
  - The known label of test sample is compared with the classified result from the model
  - Accuracy is the percentage of test set samples that are correctly classified by the model
    - Need to estimate the accuracy of model
  - Test set is independent of training set, otherwise over-fitting will occur

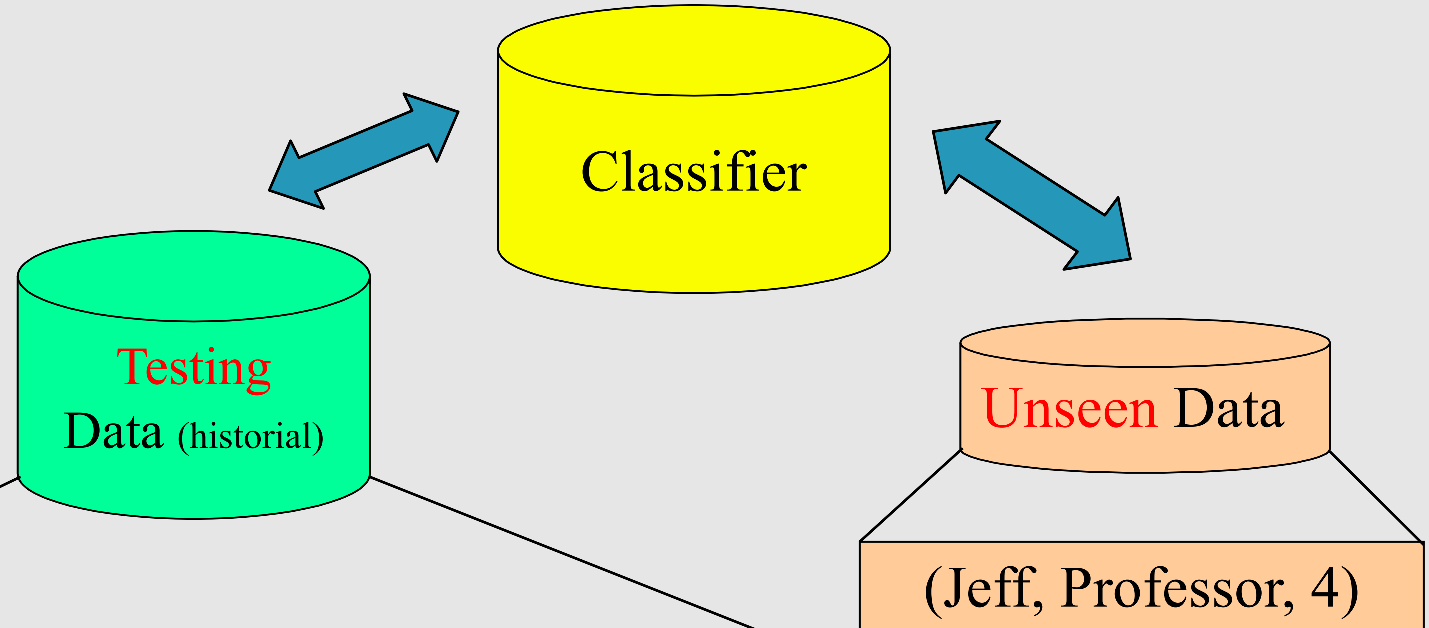
Classification Process: Model Construction



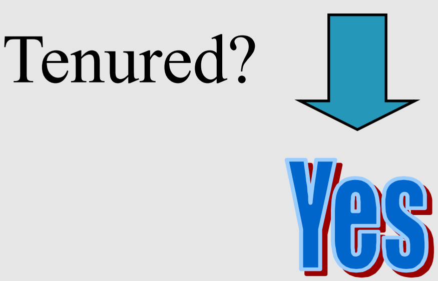
NAME	RANK	YEARS	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no

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

Classification Process: Model Usage



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



Different Types of Data in Classification

- Seen Data = Historical Data
- Unseen Data = Current and Future Data
- Seen Data:
  - Training Data – for constructing the model
  - Testing Data – for validating the model (It is unseen during the model construction process)
- Unseen Data:
  - Application (actual usage) of the constructed model

*** We will later come back to data for validation ***

Supervised vs. Unsupervised Learning

- **Supervised learning (classification)**
  - Supervision: The training data (observations, measurements, etc.) are accompanied by labels indicating the class of the observations
  - New 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

Issues regarding classification and prediction: Data preparation

- Data cleaning
  - Preprocess data in order to reduce noise and handle missing values
- Relevance analysis (feature selection)
  - Remove the irrelevant or redundant attributes
- Data transformation
  - Generalize and/or normalize data

Issues regarding classification and prediction: Evaluating classification methods

- Predictive accuracy
- Speed and scalability
  - time to construct the model
  - time to use the model
- Robustness
  - handling noise and missing values
- Scalability
  - efficiency in disk-resident databases
- Interpretability:
  - understanding and insight provided by the model
- Goodness of rules
  - decision tree size
  - compactness of classification rules

Decision Tree

A Data Analytics Perspective + Feature Engineering

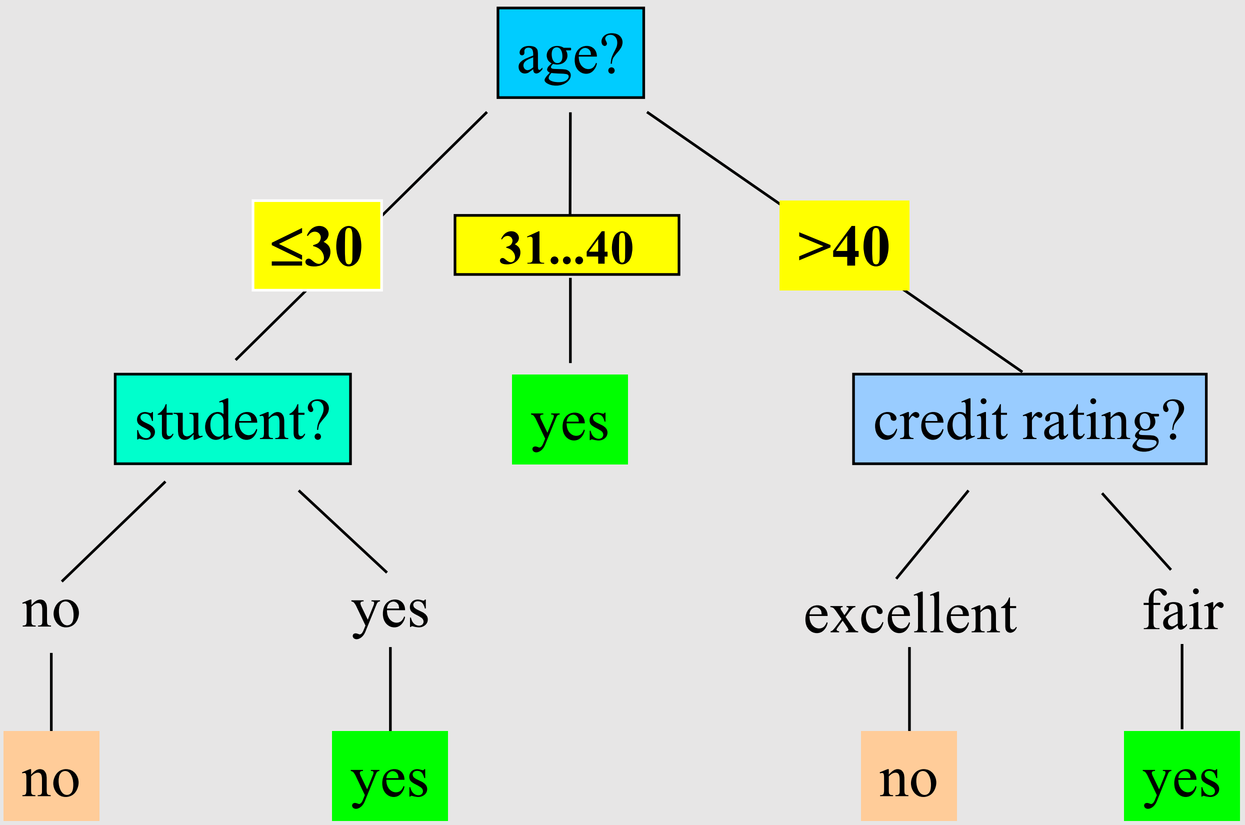
Classification by Decision Tree Induction

- Decision tree Structure
  - A flow-chart-like tree structure
  - Internal node denotes a test on an attribute
  - Branch represents an outcome of the test
  - Leaf nodes represent class labels or class distribution
- Decision tree generation consists of two phases
  - Tree construction
    - At start, all the training examples are at the root
    - Partition examples recursively based on selected attributes
  - Tree pruning
    - Identify and remove branches that reflect noise or outliers
- Use of decision tree: Classifying an unknown sample
  - Test the attribute values of the sample against the decision tree

Training Dataset:
Following an example from Quinlan's ID3

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*"



Extracting Classification Rules (Knowledge) from Decision Trees

- Represent the knowledge in the form of IF-THEN rules
- One rule is created for each path from the root to a leaf
- Each attribute-value pair along a path forms a conjunction
- The leaf node holds the class prediction
- Rules are easier for humans to understand
- Example

  IF *age* = "≤30" AND *student* = "*no*" THEN *buys_computer* = "*no*"

  IF *age* = "≤ 30" AND *student* = "*yes*" THEN *buys_computer* = "*yes*"

  IF *age* = "31…40" THEN *buys_computer* = "*yes*"

  IF *age* = ">40" AND *credit_rating* = "*excellent*" THEN *buys_computer* = "*no*"

  IF *age* = ">40" AND *credit_rating* = "*fair*" THEN *buys_computer* = "*yes*"

Algorithm for Decision Tree Construction

- 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

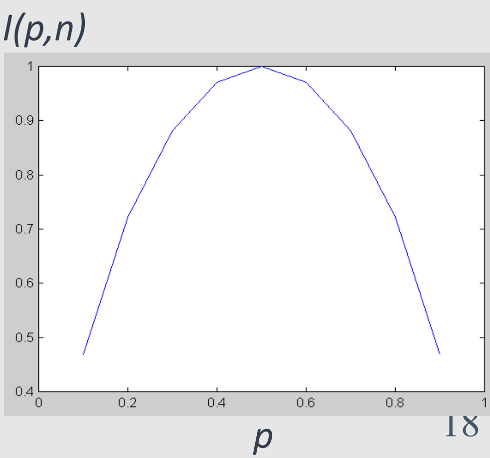
Which attribute to choose? Attribute Selection Measure

- Information gain (ID3/C4.5)
  - All attributes are assumed to be categorical
  - Can be modified for continuous-valued attributes
- Gini index (IBM IntelligentMiner)
  - All attributes are assumed continuous-valued
  - Assume there exist several possible split values for each attribute
  - May need other tools, such as clustering, to get the possible split values
  - Can be modified for categorical attributes

Information Gain (ID3/C4.5)

- Select the attribute with the highest information gain
- Assume there are two classes, *P* and *N*
  - Let the set of examples *S* contain *p* elements of class *P* and *n* elements of class *N*
  - The amount of information, needed to decide if an arbitrary example in *S* belongs to *P* or *N* is defined as

$$I(p,n) = -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$



Information Gain in Decision Tree Construction

- Assume that using attribute A a set S will be partitioned into sets $\{S_1, S_2, \ldots, S_v\}$
  - If S_i contains p_i examples of P and n_i examples of N, the entropy, or the expected information needed to classify objects in all subtrees S_i is

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I(p_i, n_i)$$

- The encoding information that would be gained by branching on A

$$Gain(A) = I(p, n) - E(A)$$

Attribute Selection by Information Gain Computation

- Class P: buys_computer = “yes”
- Class N: buys_computer =“no”
- $I(p, n) = I(9, 5) = 0.940$
- Compute the entropy for *age*:

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

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

Hence,

$$Gain(age) = I(p,n) - E(age) = 0.246$$

Similarly,

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

Thus, we should select “age” as the root node of the decision tree.

Avoid Overfitting in Classification

- The generated tree may overfit the training data
  - Too many branches, some may reflect anomalies due to noise or outliers
  - Result is in 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 a set of data different from the training data to decide which is the "best pruned tree"

Thinking time …

The model is simple enough to understand many machine learning and data analytics concepts.

Q1. Can you add one more record to the buys_computer dataset so that the classification accuracy on the training set must be <100%?

Thinking time …

Q2. Following Q.1, what can you do to potentially improve the training accuracy from <100% to 100%?

Fake Tips: Use a better supervised model!

What’s the implication?

With many rounds of thinking (research), we see Enhancements to basic decision tree induction

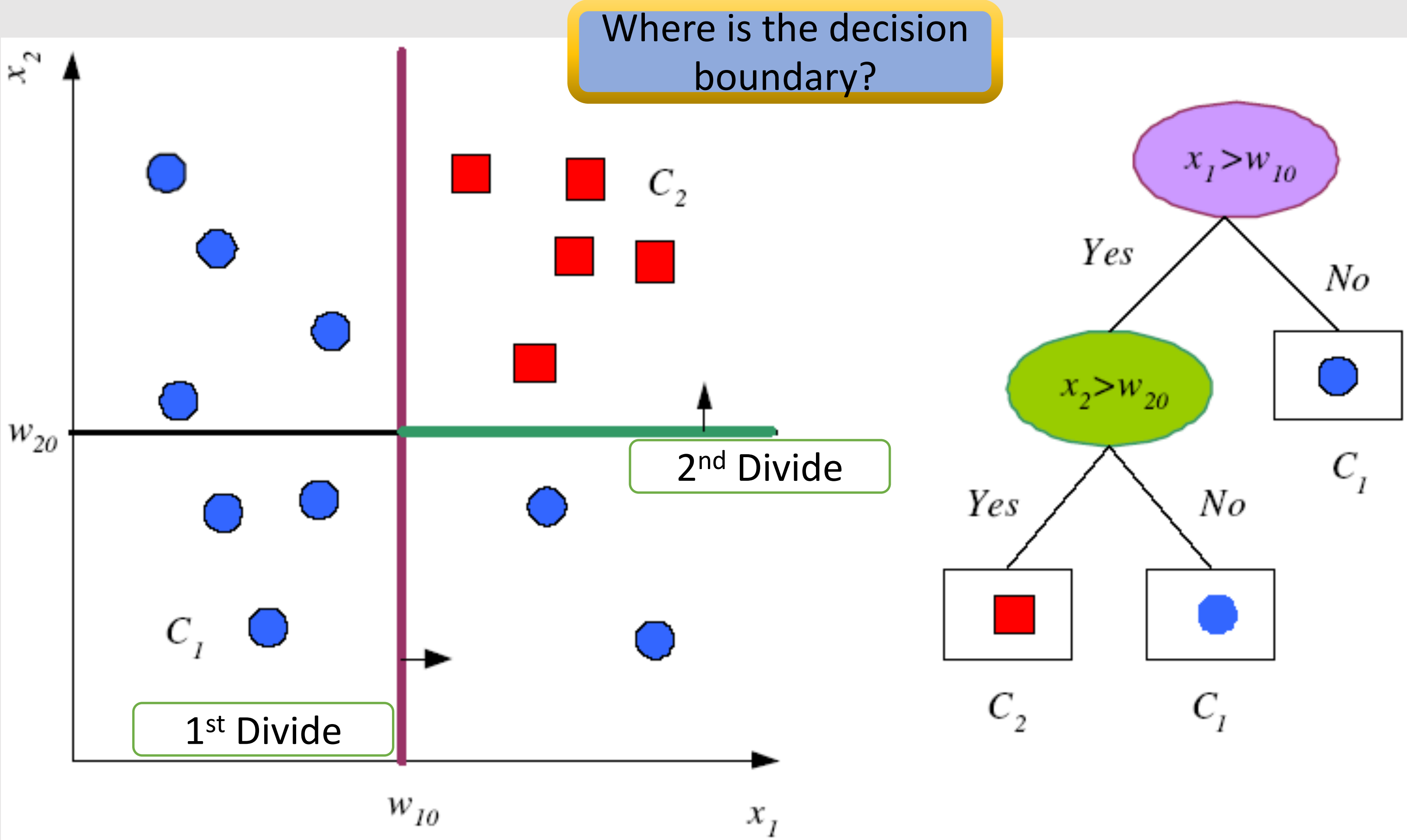
- Allow for continuous-valued attributes
  - Dynamically define new discrete-valued attributes that partition the continuous attribute value into a discrete set of intervals
- Handle missing attribute values
  - Assign the most common value of the attribute
  - Assign probability to each of the possible values
- Attribute construction
  - Create new attributes based on existing ones that are sparsely represented
  - This reduces fragmentation, repetition, and replication

Classification in Large Databases

- Classification—a classical problem extensively studied by statisticians and machine learning researchers
- Scalability: Classifying data sets with millions of examples and hundreds of attributes with reasonable speed
- Why decision tree induction in data mining?
  - relatively faster learning speed (than other classification methods)
  - convertible to simple and easy to understand classification rules
  - can use SQL queries for accessing databases
  - comparable classification accuracy with other methods

Let’s take a deeper look at Decision Trees

Nodes and Leaves of DT



Divide and Conquer Concept

- Internal decision nodes
  - Univariate: Uses a single attribute, x_i
    - Numeric x_i : Binary split : $x_i > w_m$
    - Discrete x_i : n-way split for n possible values (e.g. High, Low, Medium)
  - Multivariate **(Not so common!)**: Uses all attributes, $\boldsymbol{x}$
- Leaves
  - Classification: Class labels, or proportions
  - Regression: Numeric; r average, or local fit
- **Learning is greedy**; find the best split recursively (Breiman et al, 1984; Quinlan, 1986, 1993)

How to divide? - Classification Trees (ID3, CART, C4.5)

- For node *m*, N_m instances reach *m*, N^i_m belong to C_i

$$\hat{P}(C_i \mid \mathbf{x}, m) \equiv p^i_m = \frac{N^i_m}{N_m}$$

- Node *m* is pure if p^i_m is 0 or 1
- Measure of impurity is entropy

$$\mathcal{I}_m = -\sum_{i=1}^{K} p^i_m \log_2 p^i_m$$

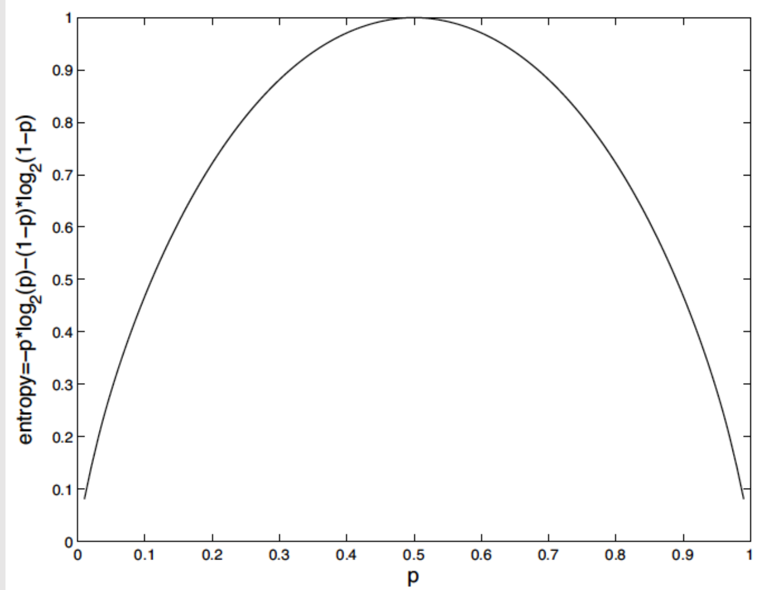


Figure 9.2 Entropy function for a two-class problem.

Best Split (Divide)

- If node *m* is pure, generate a leaf and stop, otherwise split and continue recursively
- Impurity after split: N_{mj} of N_m take branch *j*. N^i_{mj} belong to C_i

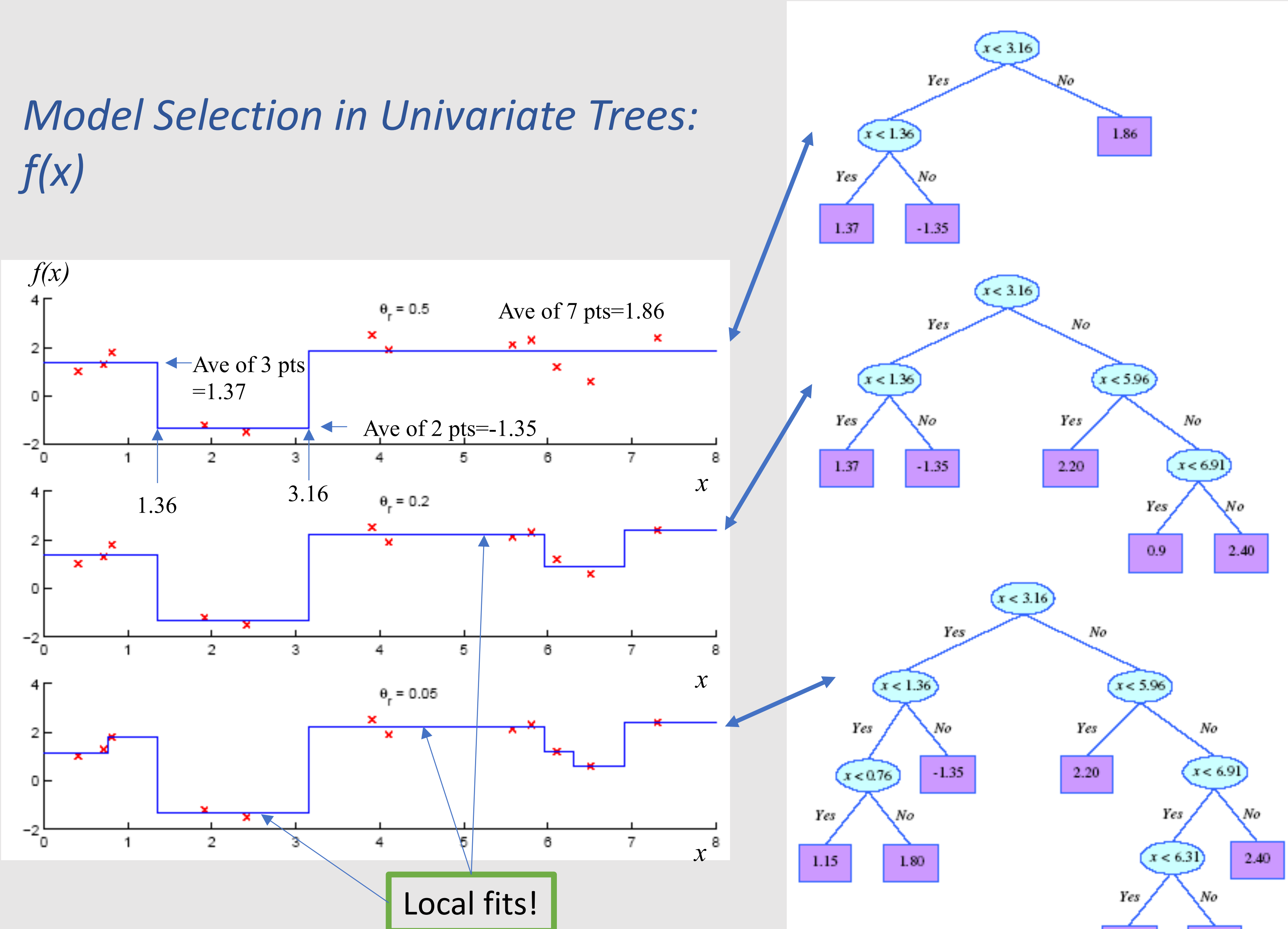
$$\hat{P}(C_i \mid \mathbf{x}, m, j) \equiv p^i_{mj} = \frac{N^i_{mj}}{N_{mj}}$$

$$\mathcal{I}'_m = -\sum_{j=1}^{n} \frac{N_{mj}}{N_m} \sum_{i=1}^{K} p^i_{mj} \log_2 p^i_{mj}$$

- Find the variable and split that minimize impurity (among all variables -- and split positions for numeric variables) → many possible methods!

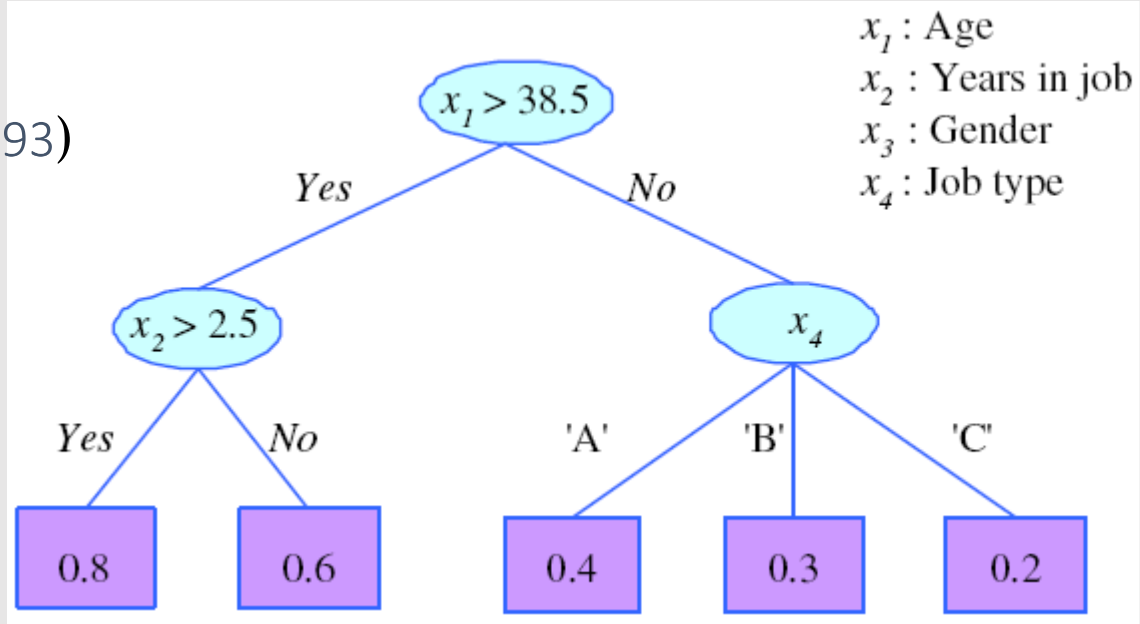
```
GenerateTree(𝒳)
    If NodeEntropy(𝒳)< θ_I /*  (See I_m on slide 29)
      Create leaf labelled by majority class in 𝒳
      Return
    i ← SplitAttribute(𝒳)
    For each branch of x_i
      Find 𝒳_i falling in branch
      GenerateTree(𝒳_i)                          Recursive implementation!
SplitAttribute(𝒳)
    MinEnt← MAX
    For all attributes i = 1, . . . , d
        If x_i is discrete with n values
          Split 𝒳 into 𝒳_1, . . . , 𝒳_n by x_i
          e ← SplitEntropy(𝒳_1, . . . , 𝒳_n) /  (See I'_m on slide 30)
          If e<MinEnt MinEnt ← e; bestf ← i
        Else /* x_i is numeric */
          For all possible splits
              Split 𝒳 into 𝒳_1, 𝒳_2 on x_i
              e←SplitEntropy(𝒳_1, 𝒳_2)
              If e<MinEnt MinEnt ← e; bestf ← i
    Return bestf
```

Model Selection in Univariate Trees: f(x)



Rule Extraction from Trees

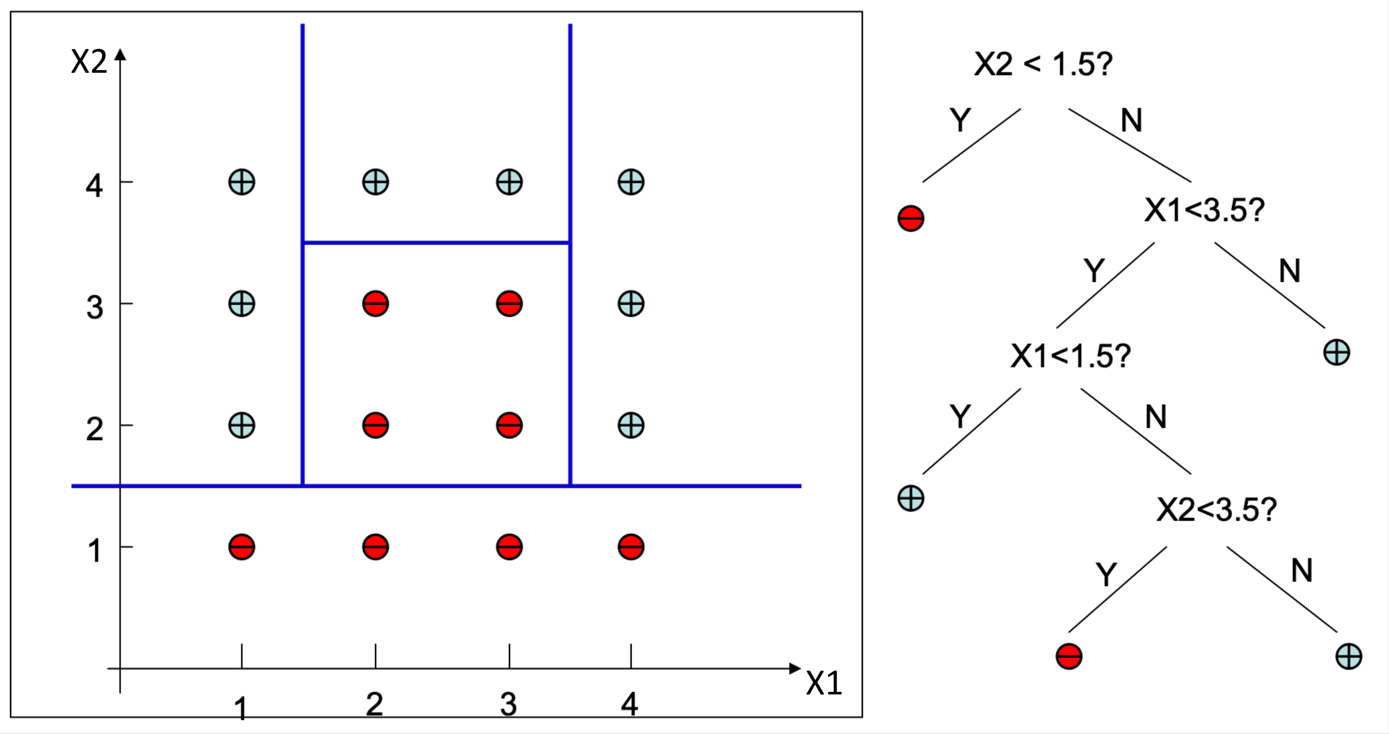
C4.5Rules
(Quinlan, 1993)



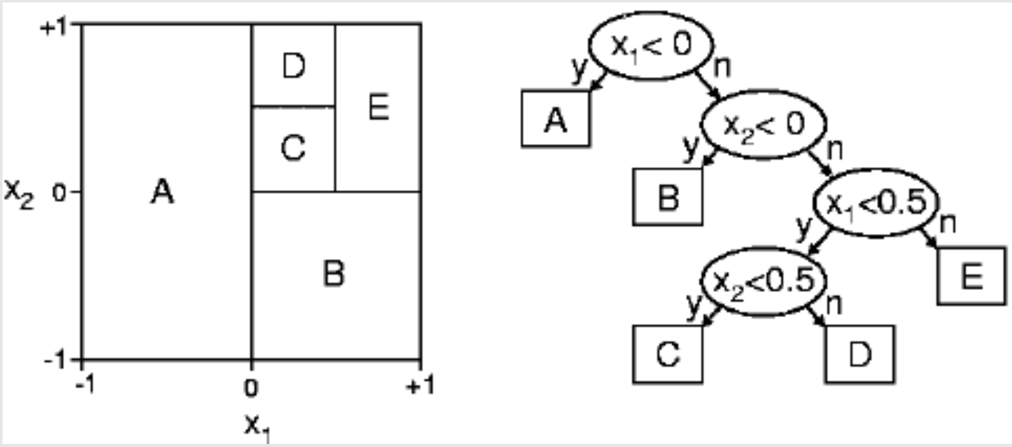
R1: IF (age>38.5) AND (years-in-job>2.5) THEN y =0.8
R2: IF (age>38.5) AND (years-in-job≤2.5) THEN y =0.6
R3: IF (age≤38.5) AND (job-type=‘A’) THEN y =0.4
R4: IF (age≤38.5) AND (job-type=‘B’) THEN y =0.3
R5: IF (age≤38.5) AND (job-type=‘C’) THEN y =0.2

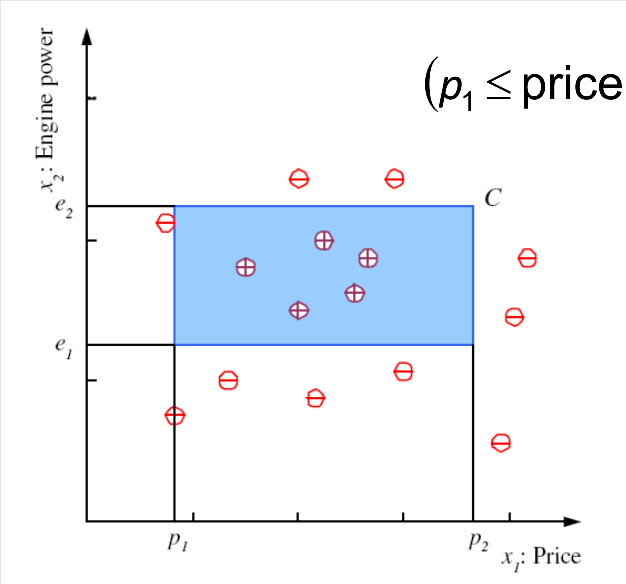
Decision Surface of Decision Tree

- Decision Trees divide the input space into axis-parallel rectangles and label each rectangle with one of the K classes



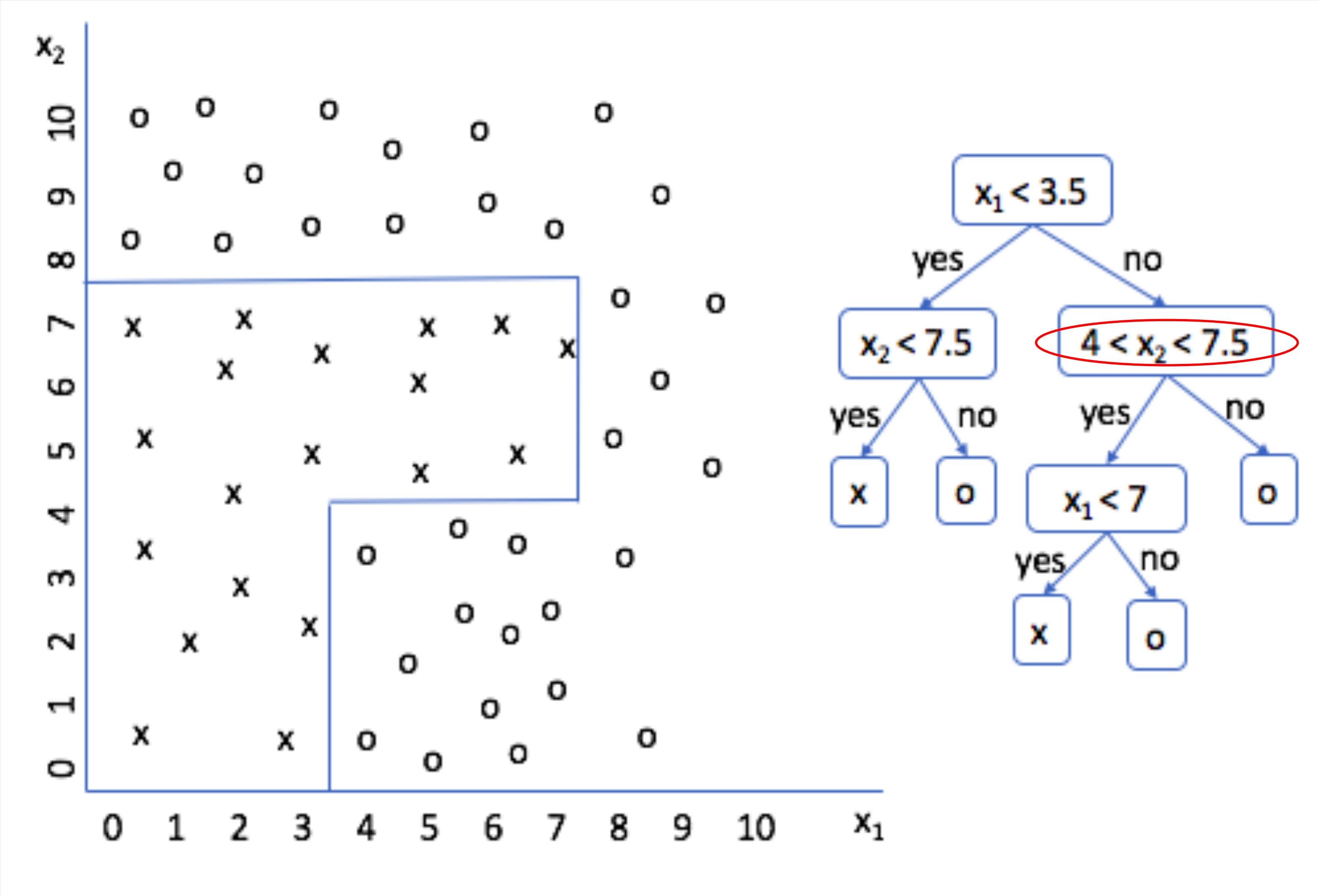
More examples





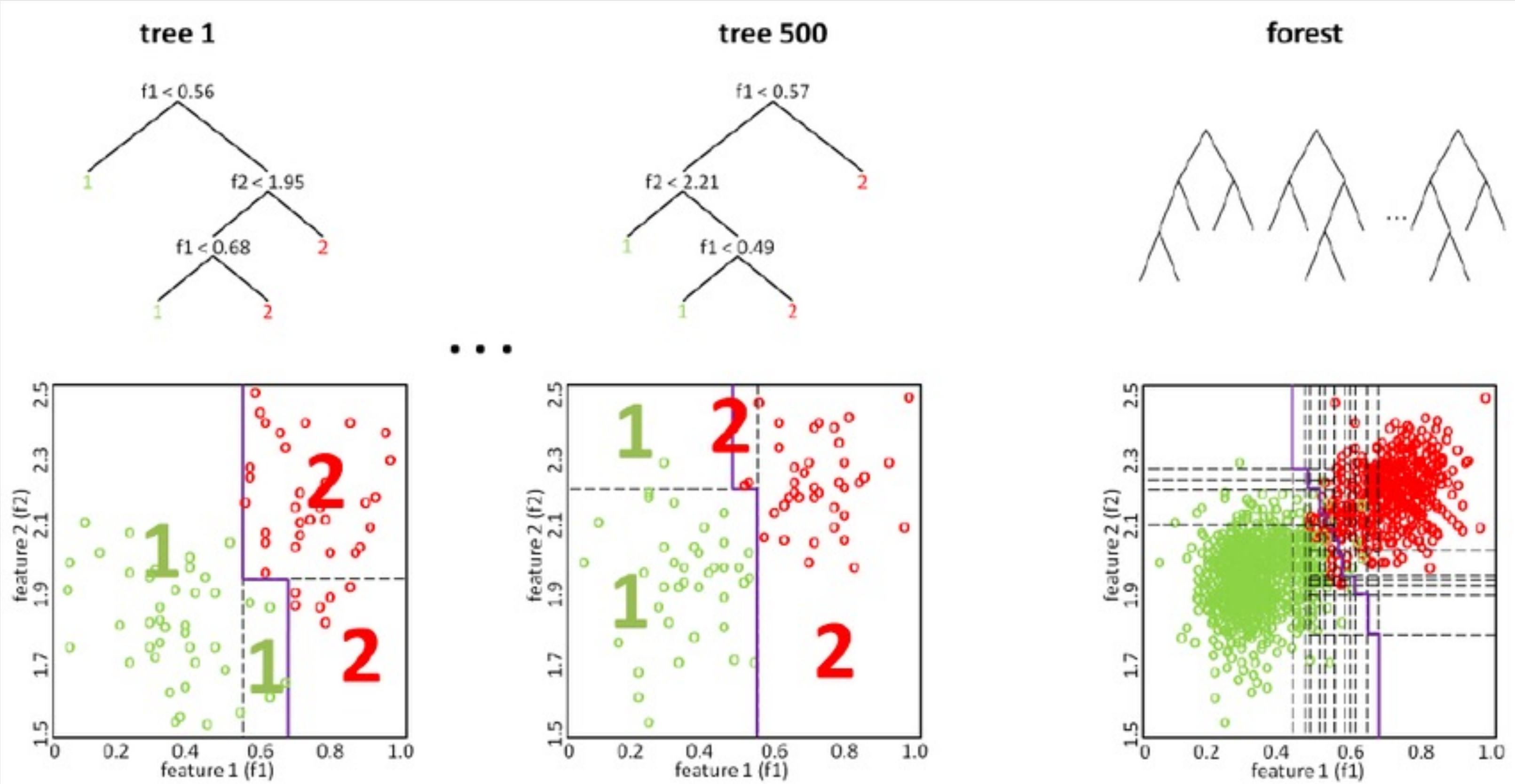
$$(p_1 \leq \text{price} \leq p_2) \text{ AND } (e_1 \leq \text{engine power} \leq e_2)$$

Extending Decision Tree with multiple splits



Another extension

- Decision Forest (boosting technique to be introduced later)

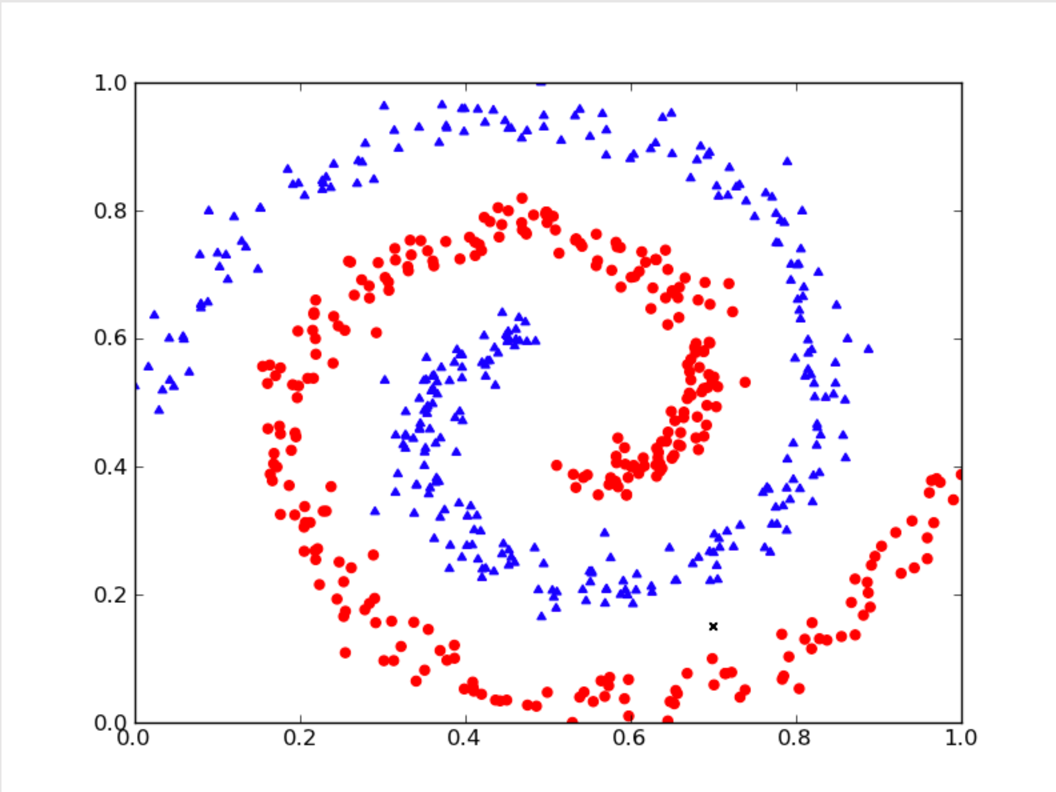


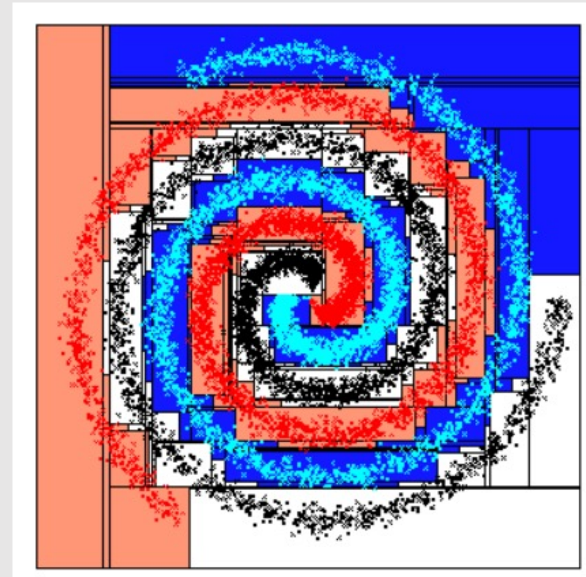
Data subset 1

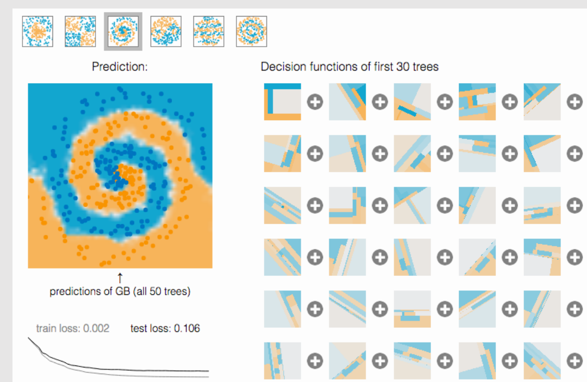
Data subset 500

Full dataset

How about such spiral data?

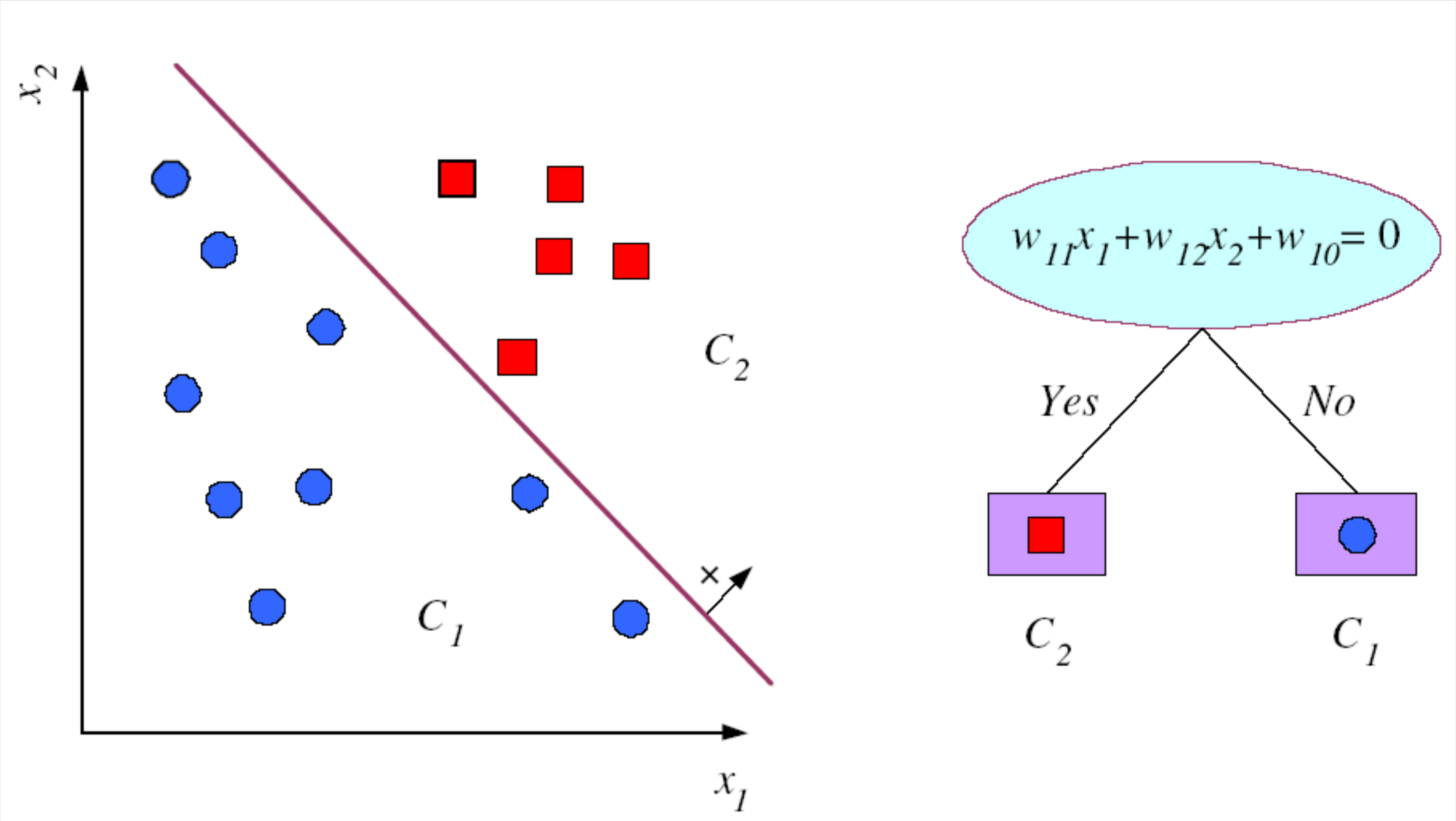






Check out this link:
http://arogozhnikov.github.io/2016/07/05/gradient_boosting_playground.html

Could it be non-axis parallel split?



Take-home messages

- Decision tree is a highly popular model in both data science and machine learning.
- The idea is very simple with tons of extensions (which involves many PhDs' hard works).
- If the feature engineering work is not good, the most sophisticated DT models like Decision Forest can only attain sub-optimal performance. As a data scientist/analyst, one may need to do a better feature engineering work. As a machine learning scientist/engineer, you may want to develop a more powerful supervised learning model.

Acknowledgement

- Slides/Materials of
  - [1] Jiawel Han and Micheline Kamber, Data Mining: Concepts and Techniques. 2nd Ed. Morgan Kaufmann, 2006.
  - [2] E. Alpaydin, Introduction to Machine Learning. 2nd Ed. MIT Press, 2010.
- Photos from Internet