Supervised Learning

Session 1 :
Theory

Abolfazl Madani

Classification vs Regression

- Classification :
  - Labels are discrete
  - Tomorrow is rainy or not?
- Regression:
  - The Number are real noy a boolean
  - Gives a actual number of rains or stock price

Just A look

- Generalization and Overfitting :

  - If we Train a data with 100% accuracy, and our Test data accuaracy going to 55% this is

Overfitting

NonParametric, Parametric, Aggregation

- Parametric:
- Nonparametric:
- Aggregation:
    - Reduce Variance, Avoid Overfitting, Statistical

Task: http://www.hcdi.net/machine-learning-in-neuroscience-a-primer/

K-Nearest Neighbor algorithm

- Most basic instance-based method
- Data are represented in a vector space
- Supervised learning

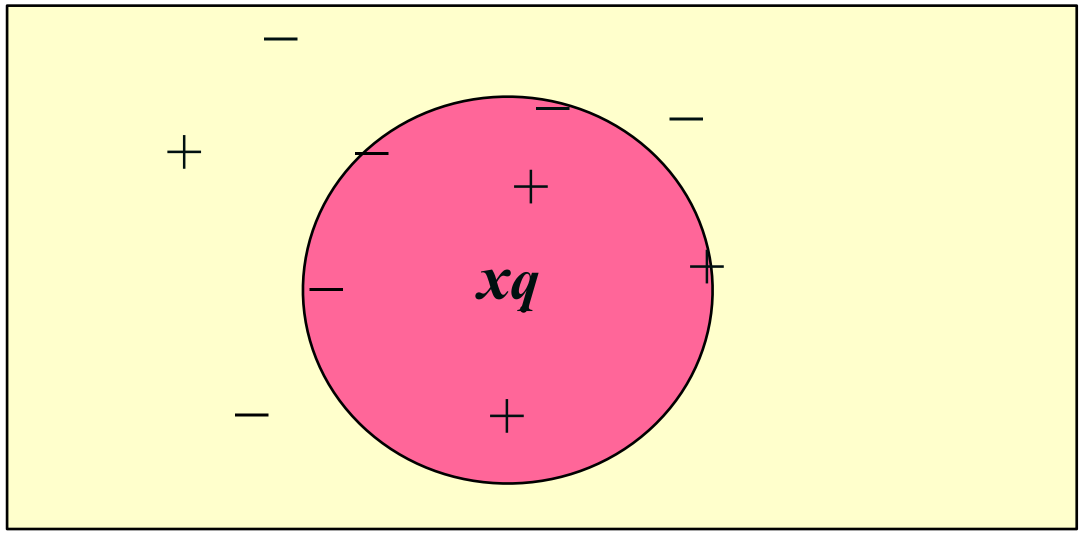
Explanation

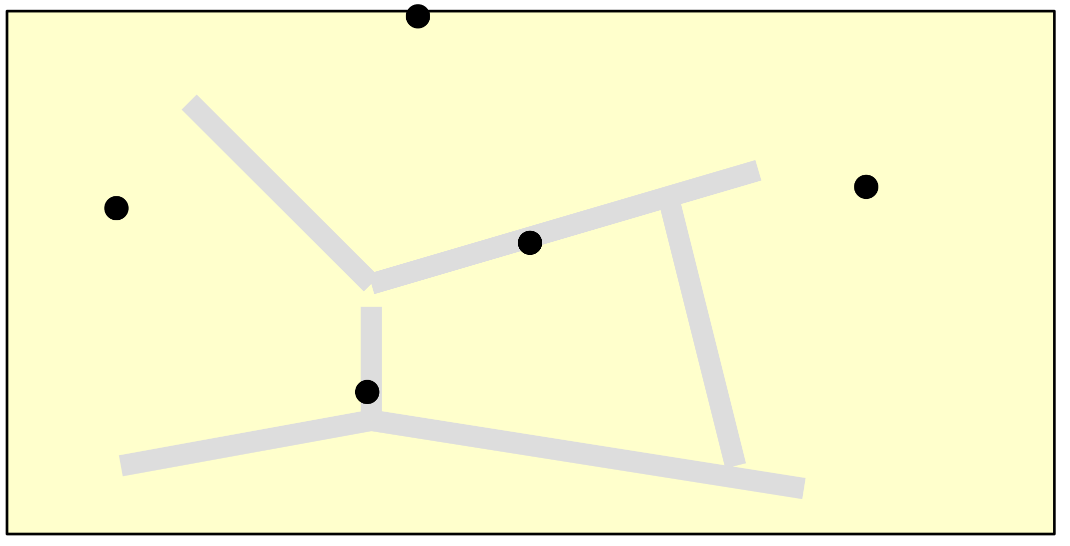
- Training algorithm
  - For each training example $<x,f(x)>$ add the example to the list
- Classification algorithm
  - Given a query instance x_q to be classified
    - Let $x_1,..,x_k$ k instances which are nearest to x_q

    - Where $\delta(a,b)=1$ if $a=b$, else $\delta(a,b)=0$ (Kronecker function)

Definition of Voronoi diagram

- the decision surface induced by 1-NN for a typical set of training examples.





When to Consider Nearest Neighbors

- Instances map to points in R^d
- Less than 20 features (attributes) per instance, typically normalized
- Lots of training data
- Advantages:
- Training is very fast
- Learn complex target functions
- Do not loose information
- Disadvantages:
- Slow at query time
    - Presorting and indexing training samples into search trees reduces time
- Easily fooled by irrelevant features (attributes)

KNN

	Study Hours	Pass
Reza	5	Y
Maryam	4	Y
Nastaran	2	N
Omid	1	N

Hyperparameters

- A parameter whose value is set before the learning process begins. By contrast, the values of other parameters are derived via training.

- Different model training algorithms require different hyperparameters

KNN

- Lazy Classifier :
  - Train is just save data
  - Train just follow the exact place
- s

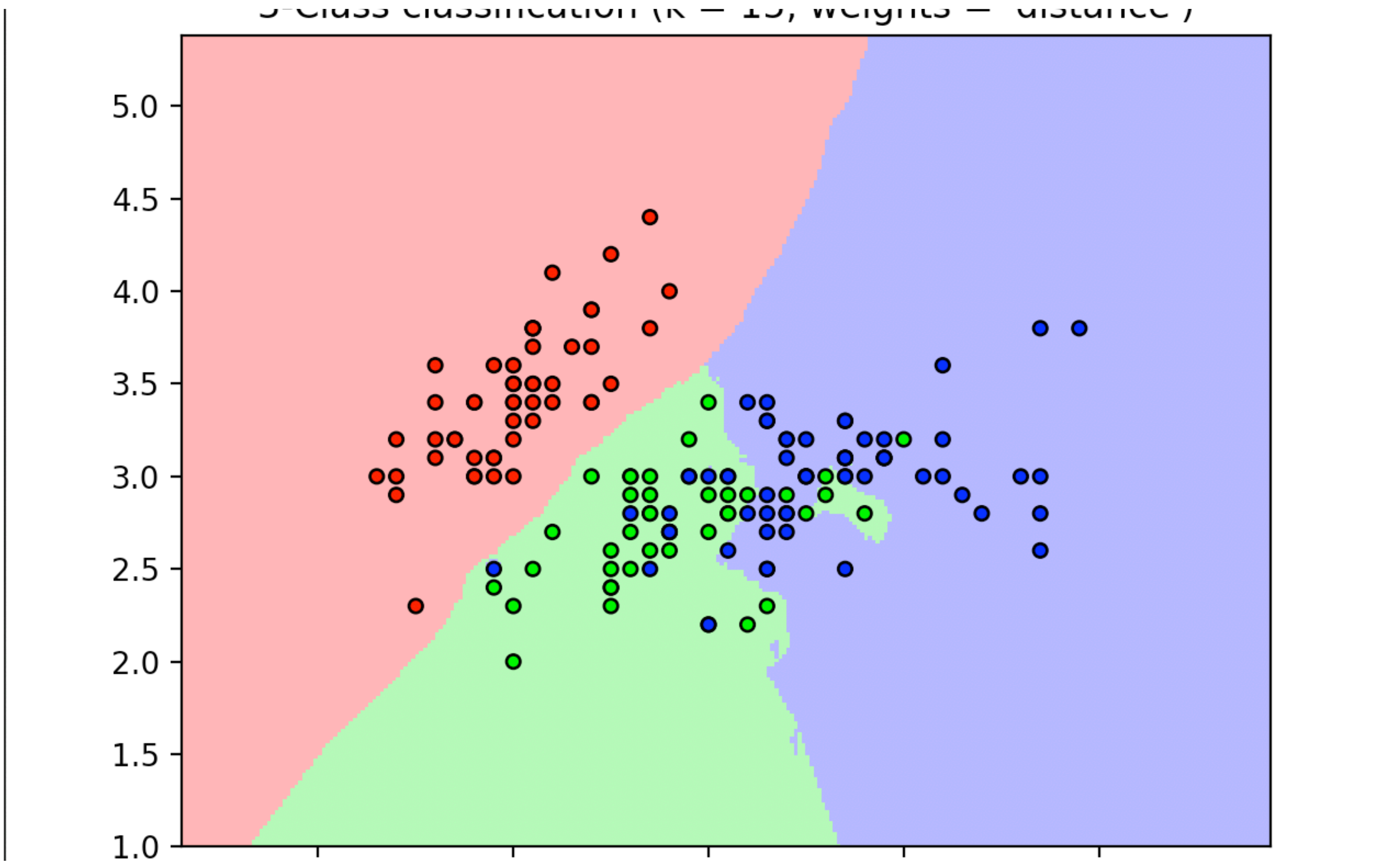
Code

- A lot of times when dealing with iterators, we also get a need to keep a count of iterations.

- Python eases the programmers' task by providing a built-in function enumerate() for this task.

- Enumerate() method adds a counter to an iterable and returns it in a form of enumerate object.

- This enumerate object can then be used directly in for loops or be converted into a list of tuples using list() method.

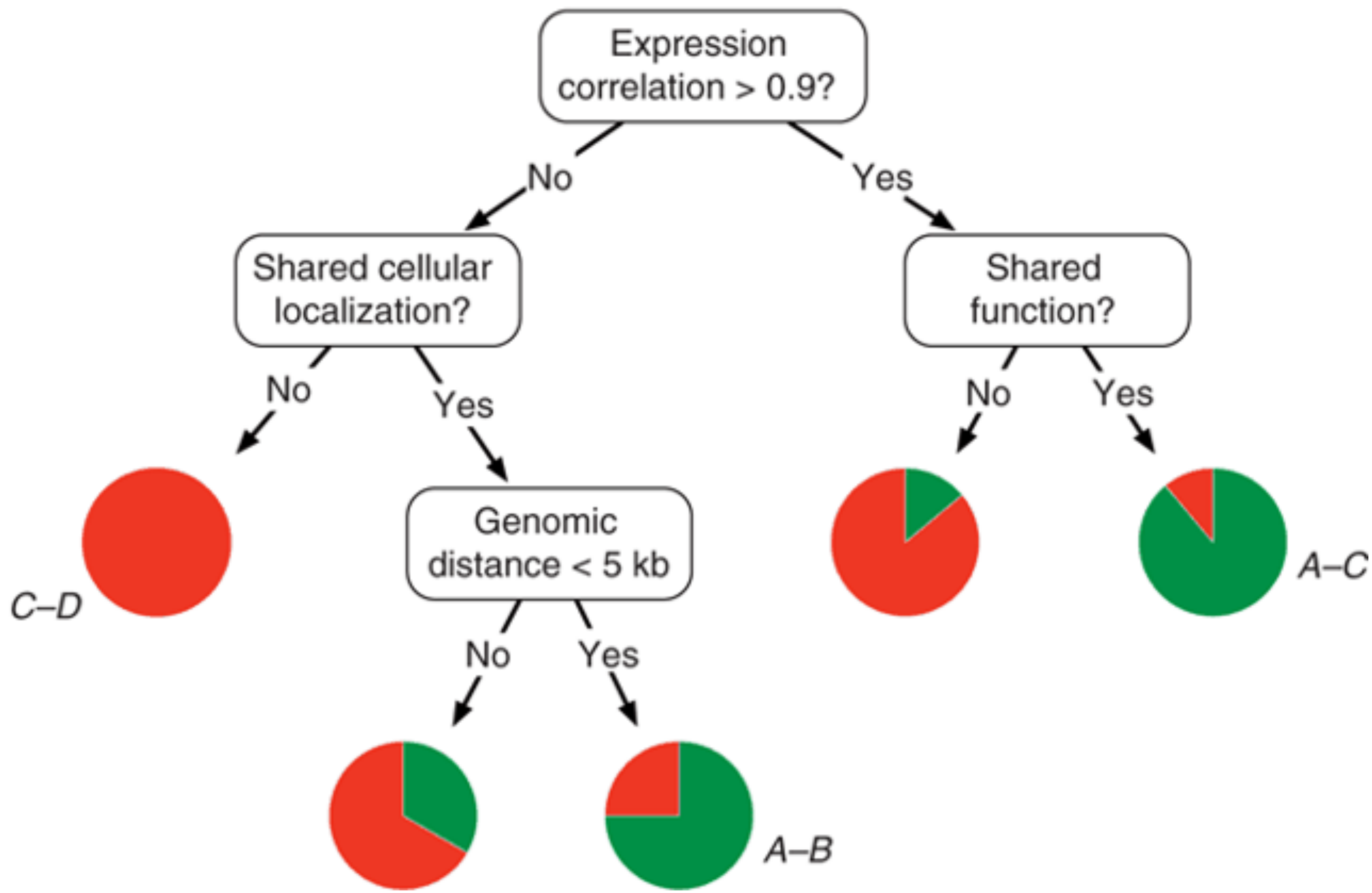


Code Result

a

Gene Pair	Interact?	Expression correlation	Shared localization?	Shared function?	Genomic distance
A-B	Yes	0.77	Yes	No	1 kb
A-C	Yes	0.91	Yes	Yes	10 kb
C-D	No	0.1	No	No	1 Mb
⋮					

b



Sample Experience Table

Example	Attributes				Target
	Hour	Weather	Accident	Stall	Commute
D1	8 AM	Sunny	No	No	Long
D2	8 AM	Cloudy	No	Yes	Long
D3	10 AM	Sunny	No	No	Short
D4	9 AM	Rainy	Yes	No	Long
D5	9 AM	Sunny	Yes	No	Long
D6	10 AM	Sunny	No	No	Short
D7	10 AM	Cloudy	No	No	Short
D8	9 AM	Rainy	No	No	Medium
D9	9 AM	Sunny	Yes	No	Long
D10	11 AM	Rainy	Yes	Yes	Long
D11	10 AM	Rainy	No	No	Short
D12	8 AM	Cloudy	Yes	No	Long
D13	9 AM	Sunny	No	No	Medium

What is a Decision Tree?

- An *inductive learning task*
  - Use particular facts to make more generalized conclusions
- A predictive model based on a branching series of Boolean tests
  - These smaller Boolean tests are less complex than a one-stage classifier
- Let’s look at a sample decision tree…

How to Create a Decision Tree

- We first make a list of attributes that we can measure

  - These attributes (for now) must be discrete

- We then choose a *target attribute* that we want to predict

- Then create an *experience table* that lists what we have seen in the past

When to use Decision Trees

- Problem characteristics:

  - Instances can be described by attribute value pairs
  - Target function is discrete valued
  - Possibly noisy training data samples
    - Robust to errors in training data
    - Missing attribute values

- Different classification problems:

  - Equipment or medical diagnosis
  - Credit risk analysis
  - Several tasks in natural language processing

Dtree

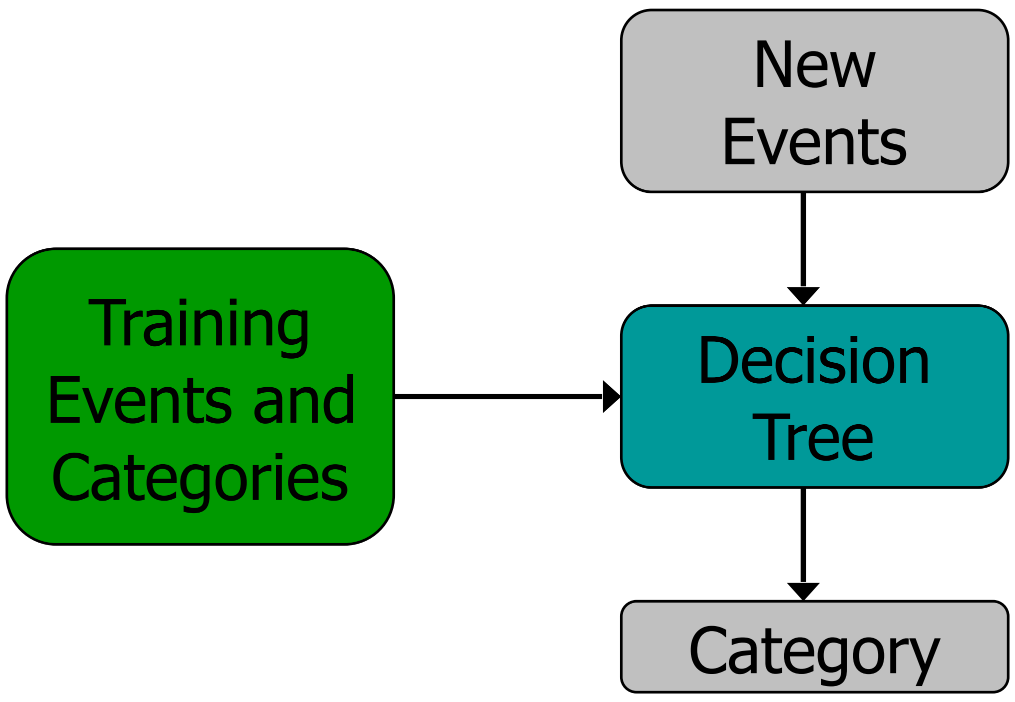
- If our data is not Complex it could be great to using Simple Classification.
- But if it’s more complex and all columns and rows related to each other.

Goal : Categorization

- Given an event, predict is category. Examples:
  - Who won a given ball game?
  - How should we file a given email?
  - What word sense was intended for a given occurrence of a word?
- Event = list of features.
- Examples:
  - Ball game: Which players were on offense?
  - Email: Who sent the email?
  - Disambiguation: What was the preceding word?

Introduction

- Use a decision tree to predict categories for new events.
- Use training data to build the decision tree.



WSD: Sample Training Data

Features				Word Sense
pos	near(race)	near(river)	near(stockings)	
noun	no	no	no	Task1
verb	no	no	no	Task2
verb	no	yes	no	Task3
noun	yes	yes	yes	Task4
verb	no	no	yes	Task5
verb	yes	yes	no	Task6
verb	no	yes	yes	Task7

Entropy

- Entropy is minimized when all values of the target attribute are the same.
    - If we know that commute time will always be *short*, then entropy = 0
- Entropy is maximized when there is an equal chance of all values for the target attribute (i.e. the result is random)
    - If commute time = short in 3 instances, medium in 3 instances and long in 3 instances, entropy is maximized

Entropy in general

- Entropy measures the amount of information in a random variable

  $H(X) = -p_+ \, log_2 \, p_+ - p_- \, log_2 \, p_- \qquad X = \{+, -\}$

  for binary classification [two-valued random variable]

  $$H(X) = -\sum_{i=1}^{c} p_i \, log_2 \, p_i = \sum_{i=1}^{c} p_i \, log_2 \, 1/p_i \qquad X = \{i, \ldots, c\}$$

  for classification in c classes

  Example: rolling a die with 8, equally probable, sides

  $$H(X) = -\sum_{i=1}^{8} 1/8 \, log_2 \, 1/8 = - \, log_2 \, 1/8 = log_2 \, 8 = 3$$

Entropy

- Calculation of entropy

- Entropy(S) = $\sum_{(i=1 \text{ to } l)} -|S_i|/|S| * \log_2(|S_i|/|S|)$

  - S = set of examples
  - S_i = subset of S with value v_i under the target attribute
  - l = size of the range of the target attribute

When to consider Decision Trees

- Instances describable by attribute-value pairs
- Target function is discrete valued
- Disjunctive hypothesis may be required
- Possibly noisy training data
- Missing attribute values
- Examples:
  - **Medical diagnosis**
  - Credit risk analysis
  - Object classification for robot manipulator (Tan 1993)

Entropy in binary classification

- Entropy measures the *impurity* of a collection of examples. It depends from the distribution of the random variable *p*.
  - S is a collection of training examples
  - p_+ the proportion of positive examples in S
  - p_- the proportion of negative examples in S

$$Entropy\ (S) \equiv -p_+ \, log_2 \, p_+ - p_- log_2 \, p_- \quad [0 \; log_2 0 = 0]$$

Exercise:

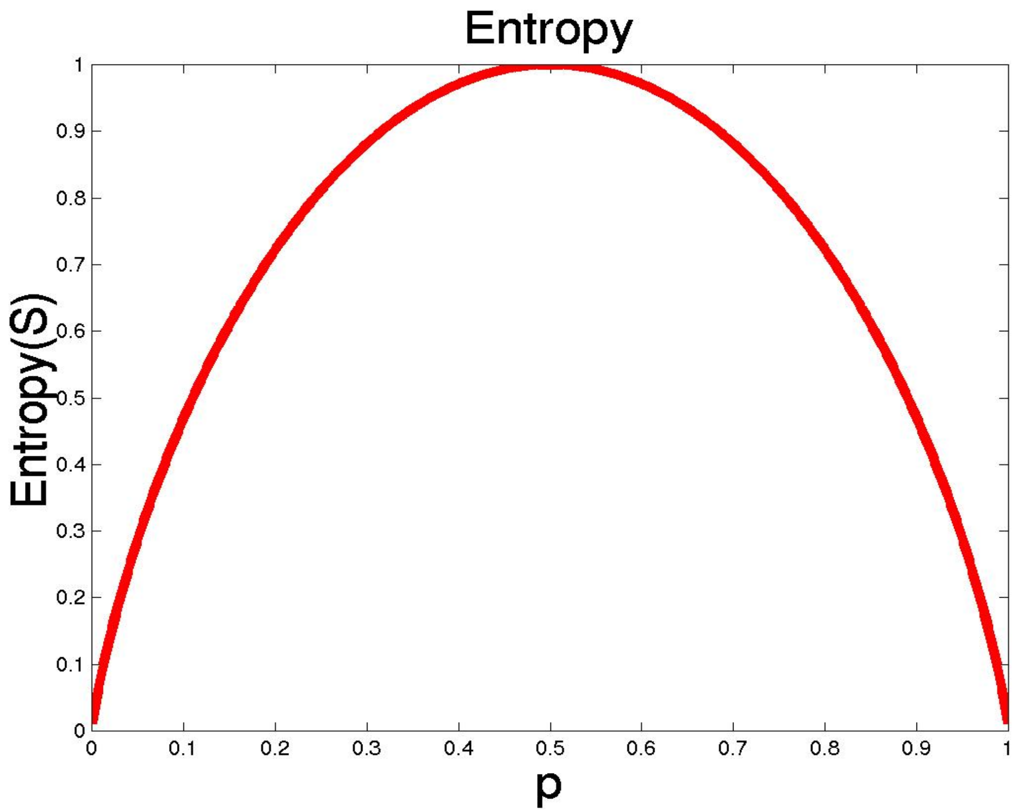
$$Entropy\ ([14+, 0-]) =$$

$$Entropy\ ([9+, 5-]) =$$

$$Entropy\ ([7+, 7-]) =$$

- Note: the log of a number < 1 is negative, $0 \le p \le 1$, $0 \le entropy \le 1$

Entropy



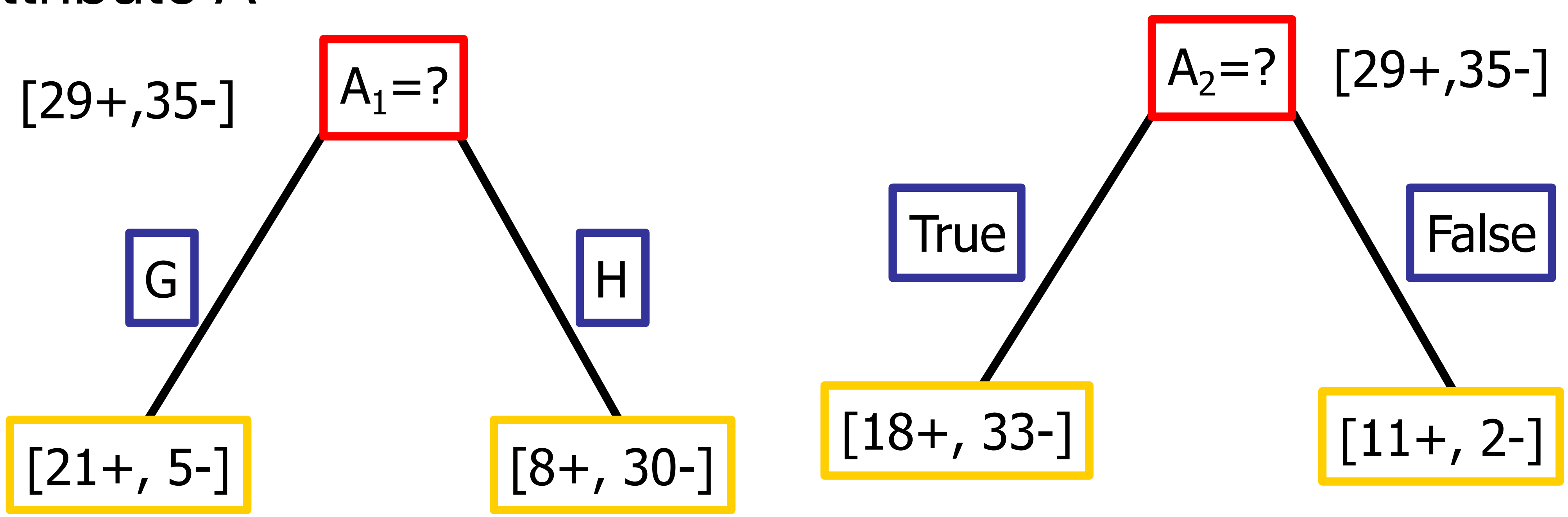
- S is a sample of training examples
- p_+ is the proportion of positive examples
- p_- is the proportion of negative examples
- Entropy measures the impurity of S

Information Gain (S=E)

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in D_A} \frac{|S_v|}{|S|} Entropy(S_v)$$

Entropy([29+,35-]) = -29/64 log2 29/64 – 35/64 log2 35/64 = 0.99

- Gain(S,A): expected reduction in entropy due to sorting S on attribute A



Overfitting

- Definition: If your machine learning algorithm fits noise (i.e. pays attention to parts of the data that are irrelevant) it is overfitting.

- Fact (theoretical and empirical): If your machine learning algorithm is overfitting then it may perform less well on test set data.

Avoid Overfitting

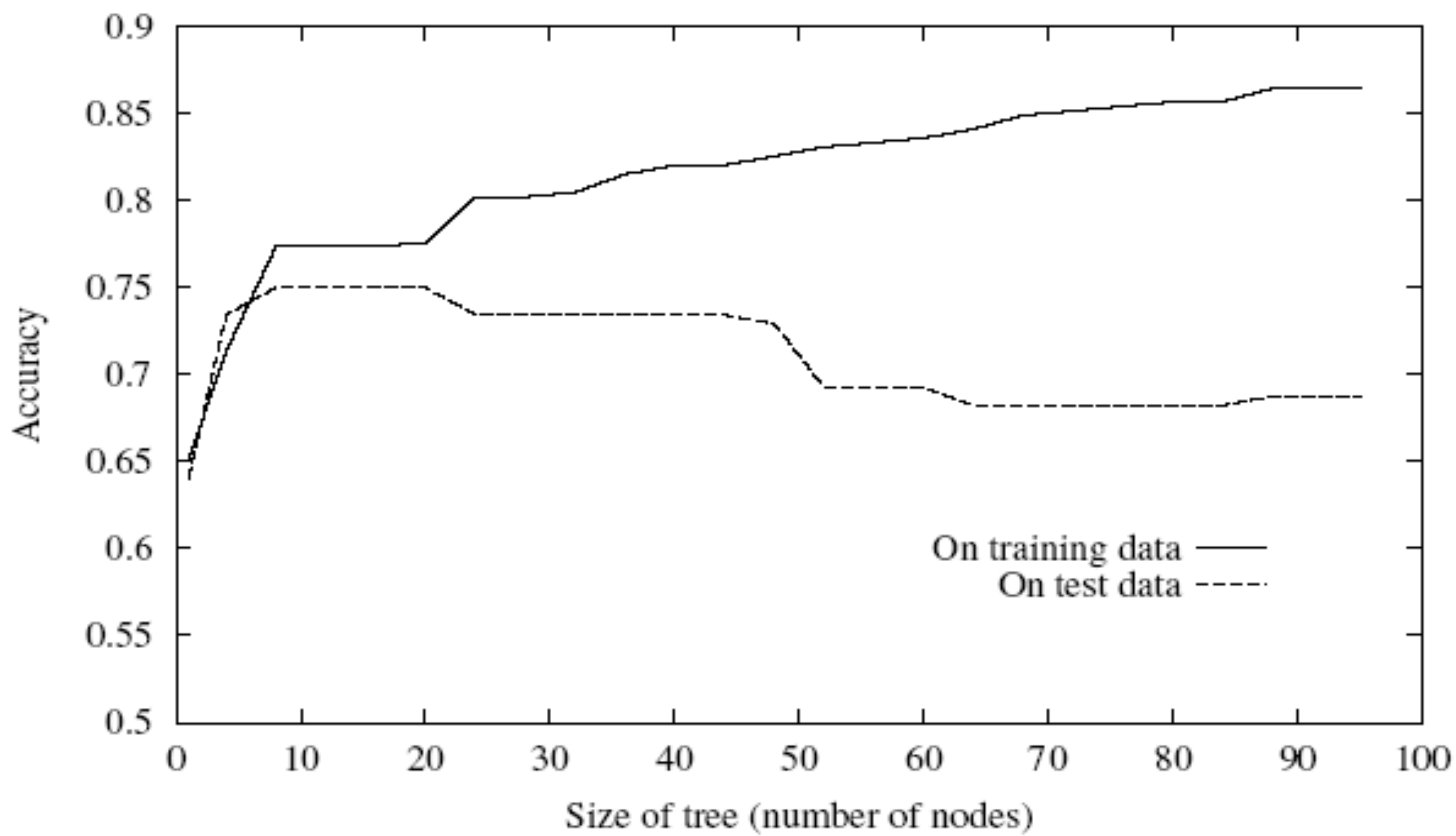
- stop growing when split not statistically significant
- grow full tree, then post-prune

Select “best” tree:

- measure performance over training data
- measure performance over separate validation data set
- min(|tree|+|misclassifications(tree)|)

Overfitting

- One of the biggest problems with decision trees is **Overfitting**



Continuous Valued Attributes

Create a discrete attribute to test continuous

- Temperature = 24.5^0C
- (Temperature > 20.0^0C) = {true, false}

Where to set the threshold?

Temperature	15^0C	18^0C	19^0C	22^0C	24^0C	27^0C
PlayTennis	No	No	Yes	Yes	Yes	No