DATA MINING
Introductory and Advanced Topics
Part II

Margaret H. Dunham
Department of Computer Science and Engineering
Southern Methodist University

Companion slides for the text by Dr. M.H.Dunham, *Data Mining, Introductory and Advanced Topics*, Prentice Hall, 2002.

Classification Outline

Goal: Provide an overview of the classification problem and introduce some of the basic algorithms

- Classification Problem Overview
- Classification Techniques
  - Rules
  - Regression
  - Decision Trees
  - Distance
  - Neural Networks

Classification Problem

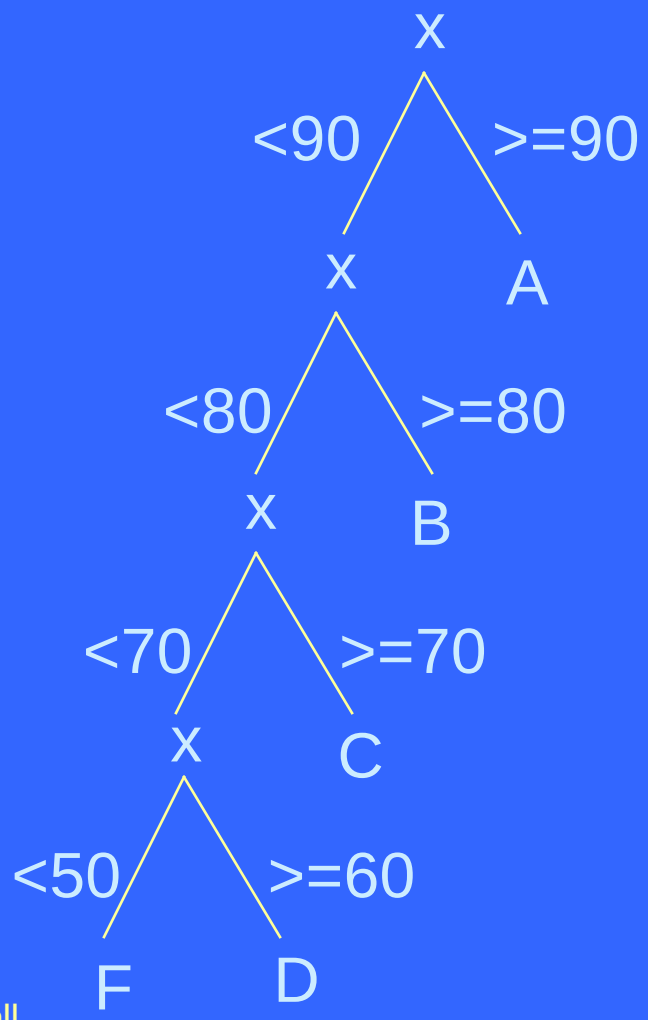
- Given a database $D=\{t_1,t_2,\ldots,t_n\}$ and a set of classes $C=\{C_1,\ldots,C_m\}$, the ***Classification Problem*** is to define a mapping $f:D\rightarrow C$ where each t_i is assigned to one class.
- Actually divides D into ***equivalence classes***.
- ***Prediction*** is similar, but may be viewed as having infinite number of classes.

Classification Examples

- Teachers classify students' grades as A, B, C, D, or F.
- Identify mushrooms as poisonous or edible.
- Predict when a river will flood.
- Identify individuals with credit risks.
- Speech recognition
- Pattern recognition

Classification Ex: Grading

- If x >= 90 then grade =A.
- If 80<=x<90 then grade =B.
- If 70<=x<80 then grade =C.
- If 60<=x<70 then grade =D.
- If x<50 then grade =F.



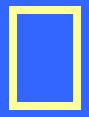
Classification Ex: Letter Recognition

View letters as constructed from 5 components:

 Letter A

 Letter **B**

 Letter C

 Letter **D**

 Letter E

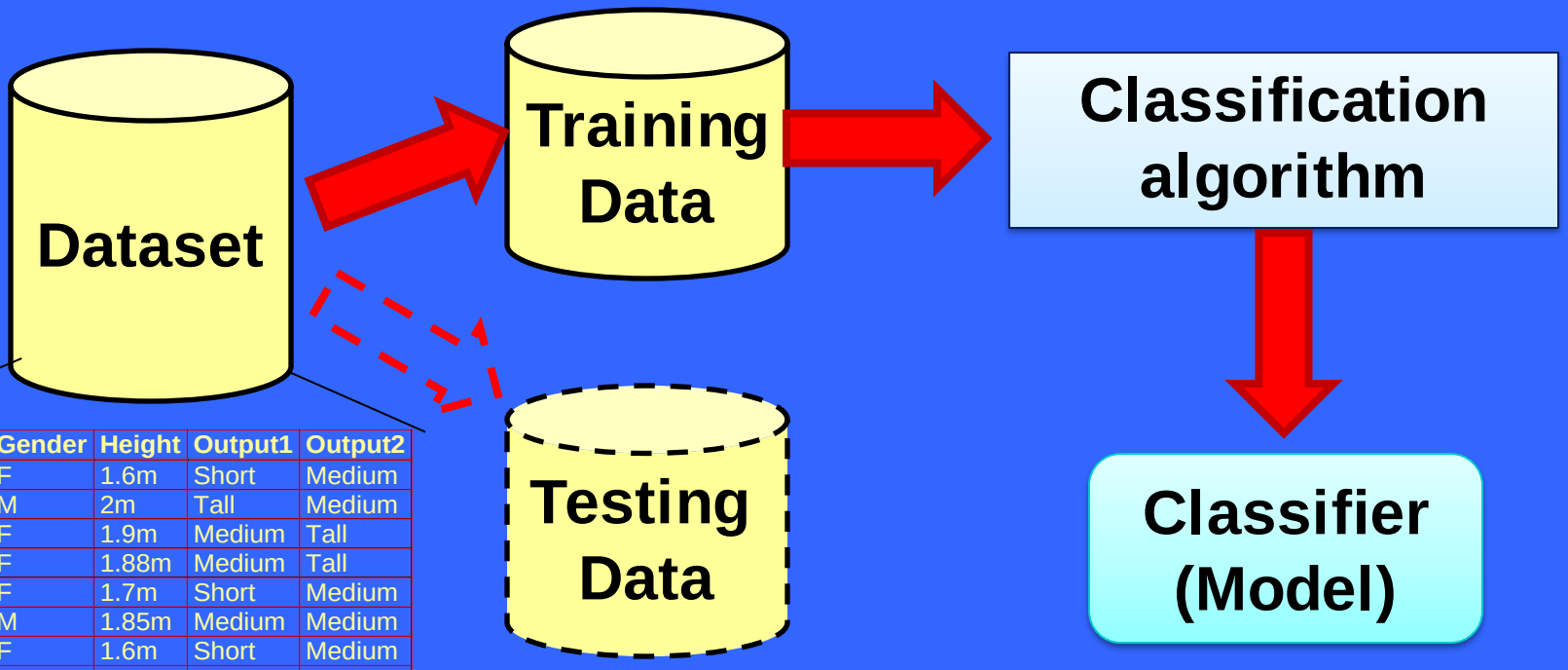
 Letter F

Classification Techniques

- Approach:
  1. Create specific model by evaluating training data (or using domain experts' knowledge).
  2. Apply model developed to new data.
- Classes must be predefined
- Most common techniques use DTs, NNs, or are based on distances or statistical methods.

Classification Pipeline

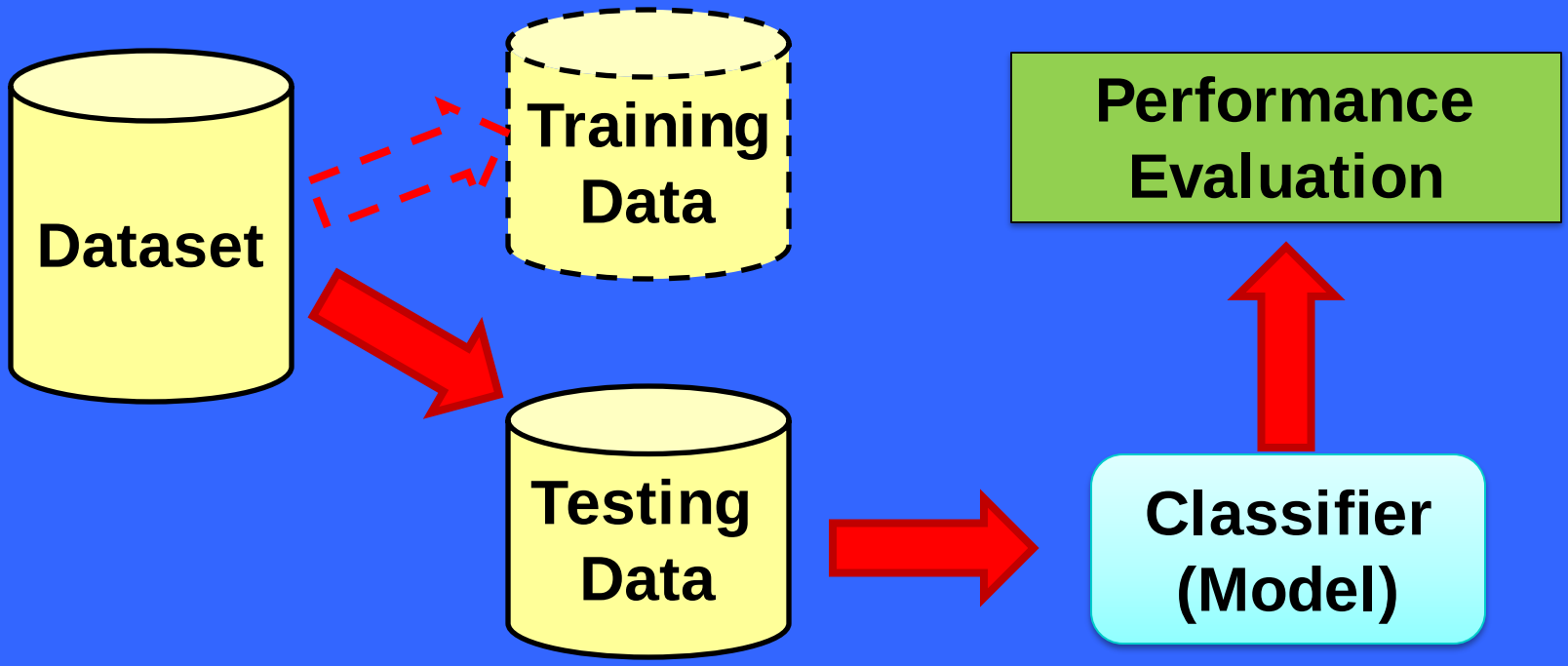
- Model construction



Name	Gender	Height	Output1	Output2
Kristina	F	1.6m	Short	Medium
Jim	M	2m	Tall	Medium
Maggie	F	1.9m	Medium	Tall
Martha	F	1.88m	Medium	Tall
Stephanie	F	1.7m	Short	Medium
Bob	M	1.85m	Medium	Medium
Kathy	F	1.6m	Short	Medium
Dave	M	1.7m	Short	Medium
Worth	M	2.2m	Tall	Tall
Steven	M	2.1m	Tall	Tall
Debbie	F	1.8m	Medium	Medium
Todd	M	1.95m	Medium	Medium
Kim	F	1.9m	Medium	Tall
Amy	F	1.8m	Medium	Medium
Wynette	F	1.75m	Medium	Medium

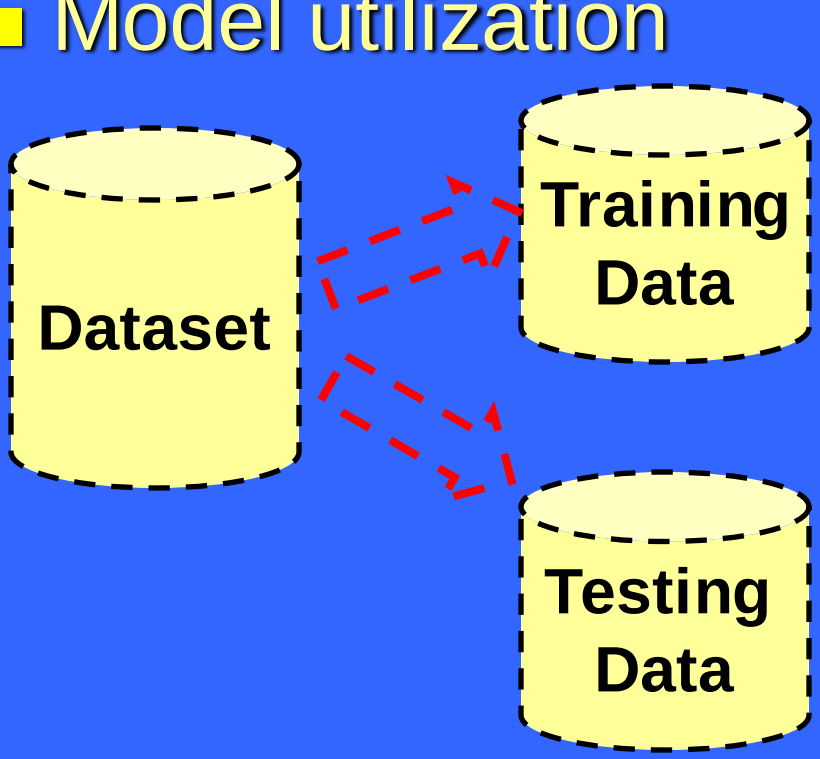
Classification Pipeline

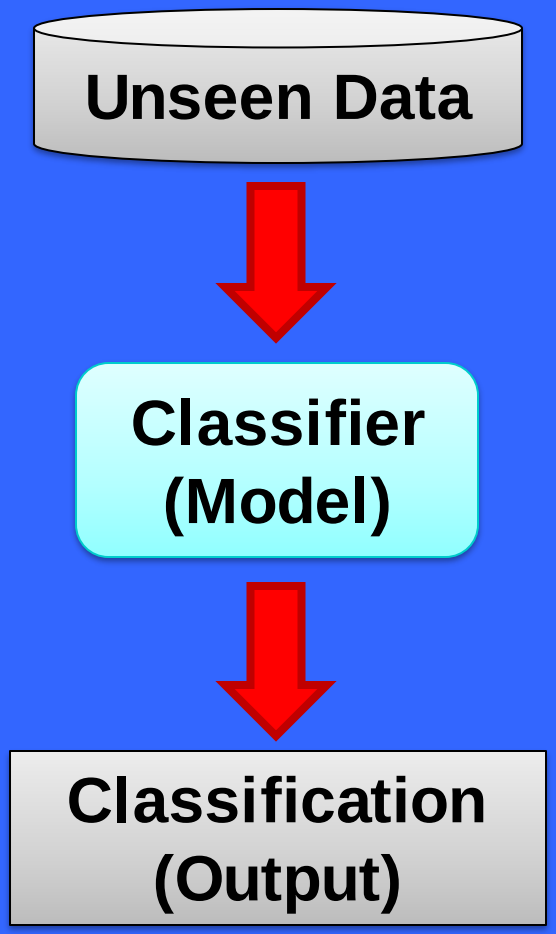
- Model evaluation



Classification Pipeline

- Model utilization

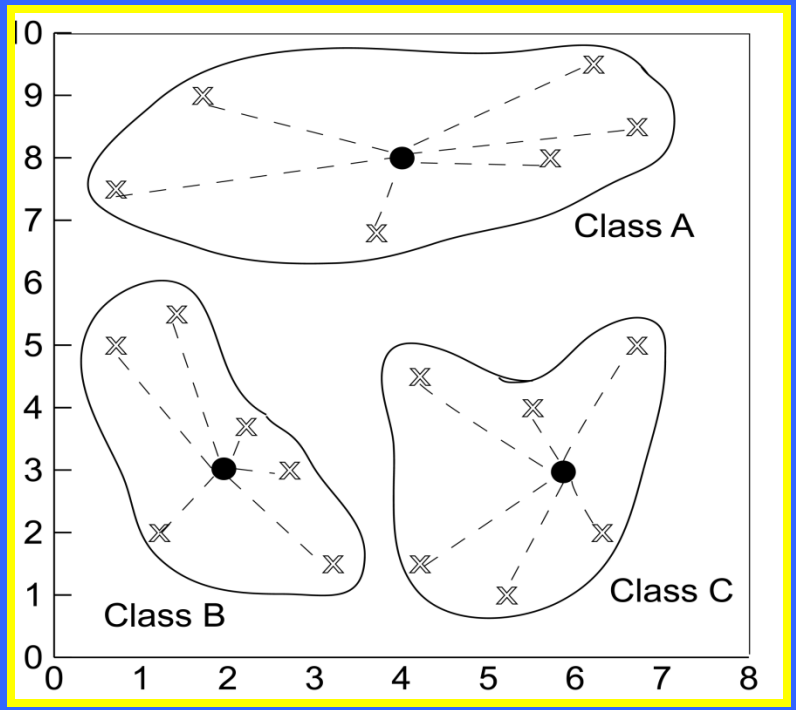




Defining Classes

Partitioning Based

Distance Based



Issues in Classification

- Missing Data
  - Ignore
  - Replace with assumed value
- Measuring Performance
  - Classification accuracy on test data
  - Confusion matrix
  - OC Curve

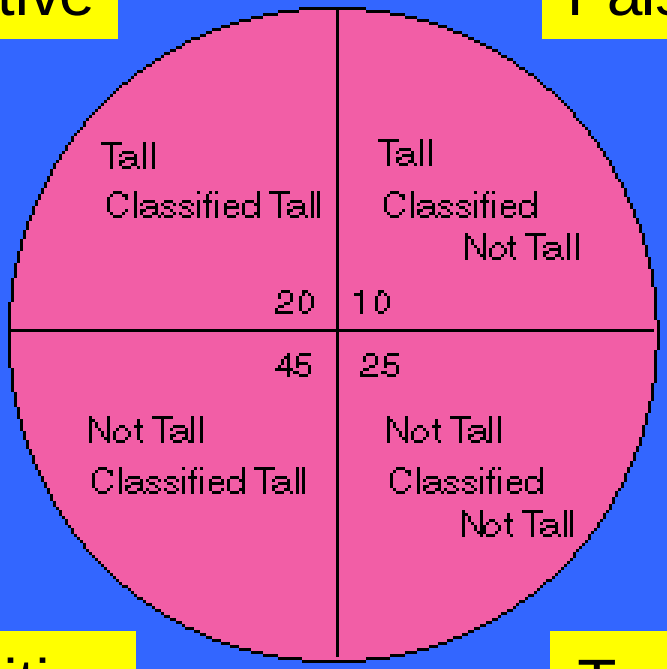
Height Example Data

Name	Gender	Height	Output1	Output2
Kristina	F	1.6m	Short	Medium
Jim	M	2m	Tall	Medium
Maggie	F	1.9m	Medium	Tall
Martha	F	1.88m	Medium	Tall
Stephanie	F	1.7m	Short	Medium
Bob	M	1.85m	Medium	Medium
Kathy	F	1.6m	Short	Medium
Dave	M	1.7m	Short	Medium
Worth	M	2.2m	Tall	Tall
Steven	M	2.1m	Tall	Tall
Debbie	F	1.8m	Medium	Medium
Todd	M	1.95m	Medium	Medium
Kim	F	1.9m	Medium	Tall
Amy	F	1.8m	Medium	Medium
Wynette	F	1.75m	Medium	Medium

Classification Performance

True Positive

False Negative



False Positive

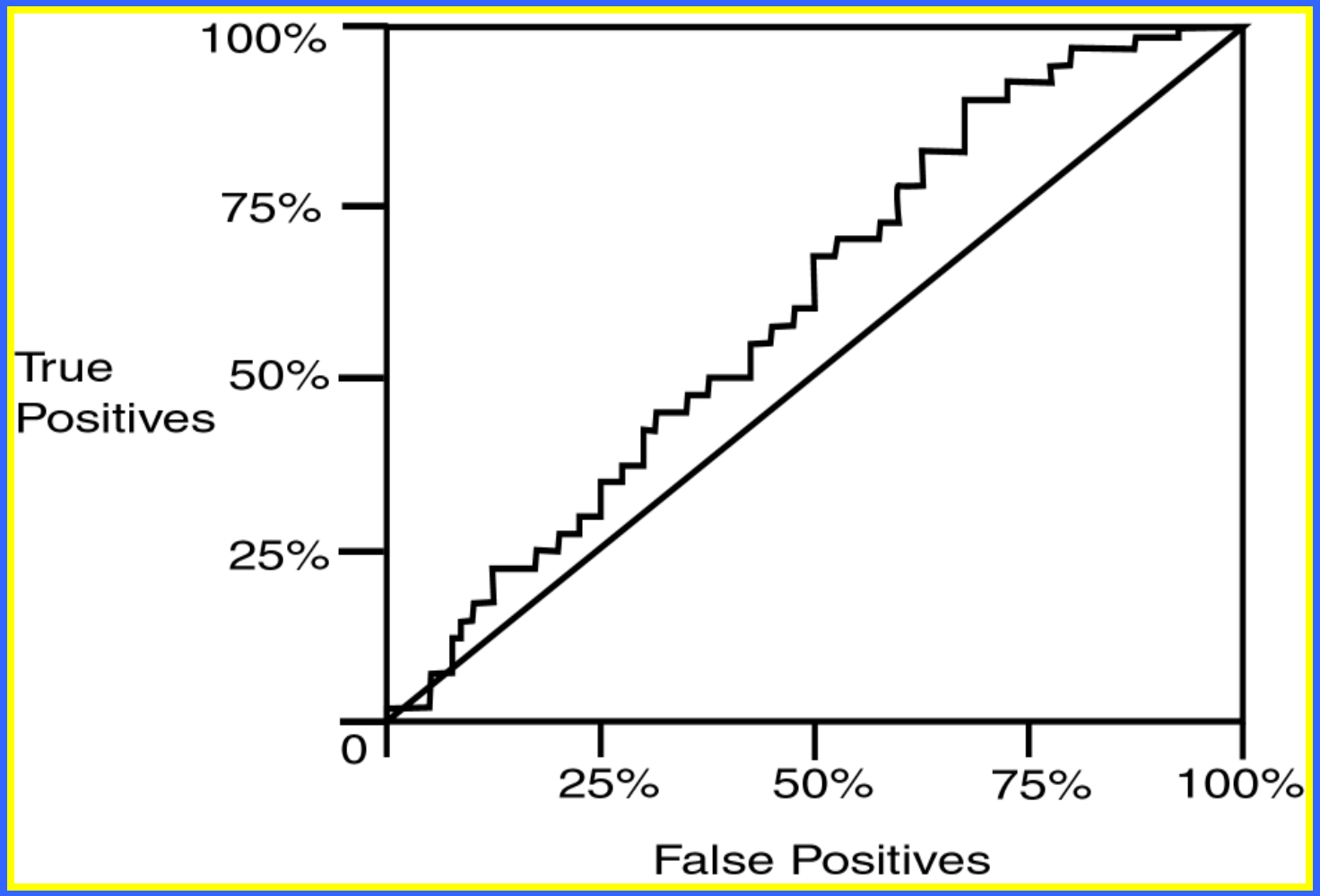
True Negative

Confusion Matrix Example

Using height data example with Output1 as assigned (prediction) and Output2 actual (true class)

Assigned Membership	**Actual Membership**		
	Short	Medium	Tall
Short	0	4	0
Medium	0	5	3
Tall	0	1	2

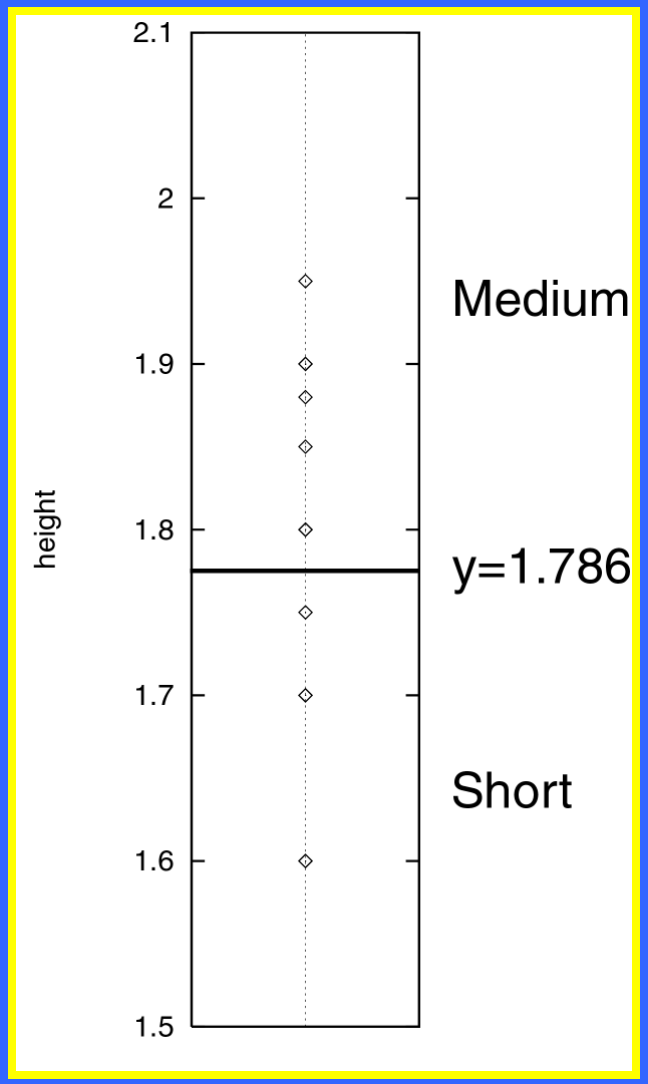
Operating Characteristic Curve



Classification Using Regression

- ***Division:*** Use regression function to divide area into regions.
- ***Prediction***: Use regression function to predict a class membership function. Input includes desired class.

Division



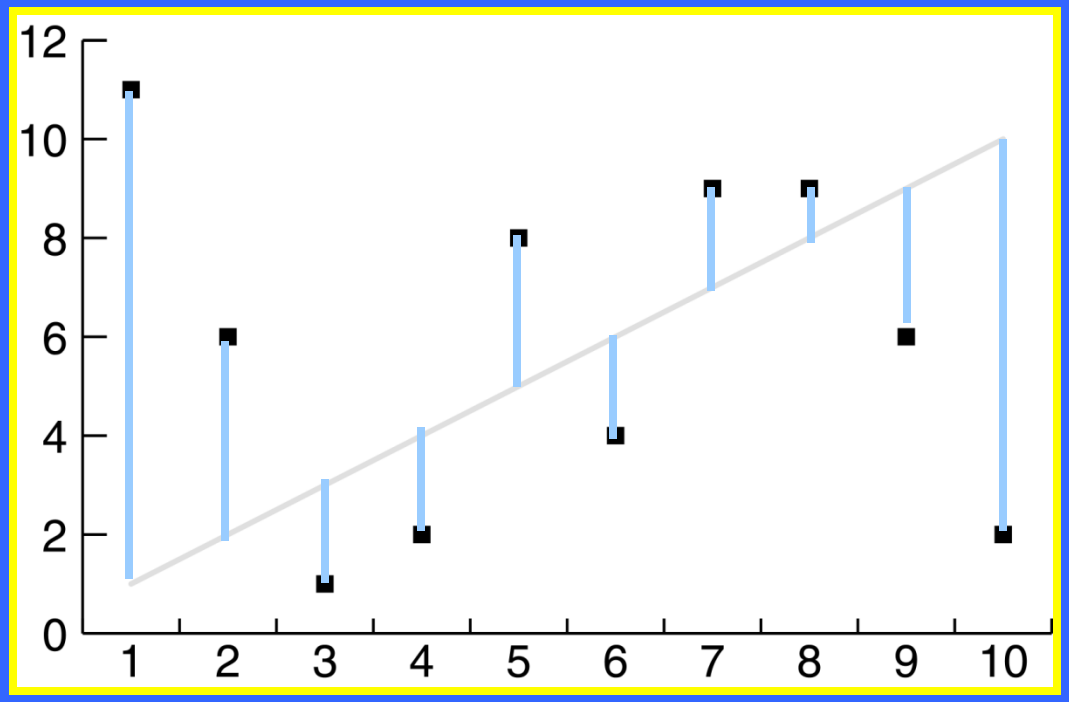
Regression

- Assume data fits a predefined function
- Determine best values for ***regression coefficients*** $c_0,c_1,\ldots,c_n$.
- Assume an error: $y = c_0+c_1x_1+\ldots+c_nx_n+\varepsilon$
- Estimate error using sum of squared error for training set:

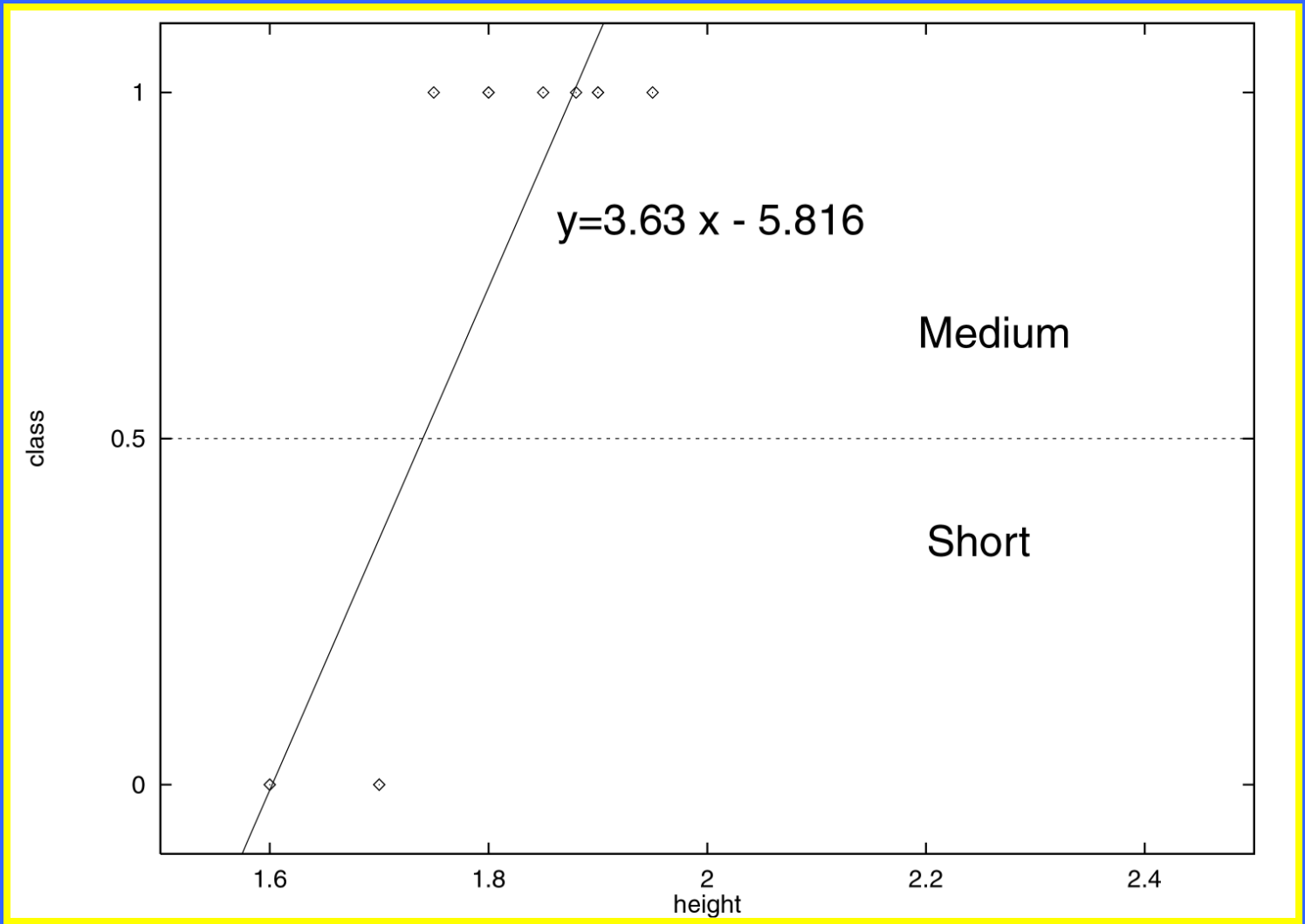
$$y_i = c_0 + c_1 x_{1i} + \epsilon_i, i = 1, ..., k$$

$$L = \sum_{i=1}^{k} \epsilon_i^2 = \sum_{i=1}^{k} (y_i - c_0 - c_1 x_{1i})^2$$

Linear Regression Poor Fit



Prediction



Classification Using Decision Trees

- ***Partitioning based:*** Divide search space into rectangular regions.
- Tuple placed into class based on the region within which it falls.
- DT approaches differ in how the tree is built: ***DT Induction***
- Internal nodes associated with attribute and arcs with values for that attribute.
- Algorithms: ID3, C4.5, CART

Decision Tree

Given:

- D = $\{t_1, \ldots, t_n\}$ where $t_i = \langle t_{i1}, \ldots, t_{ih} \rangle$
- Database schema contains $\{A_1, A_2, \ldots, A_h\}$
- Classes $C = \{C_1, \ldots, C_m\}$

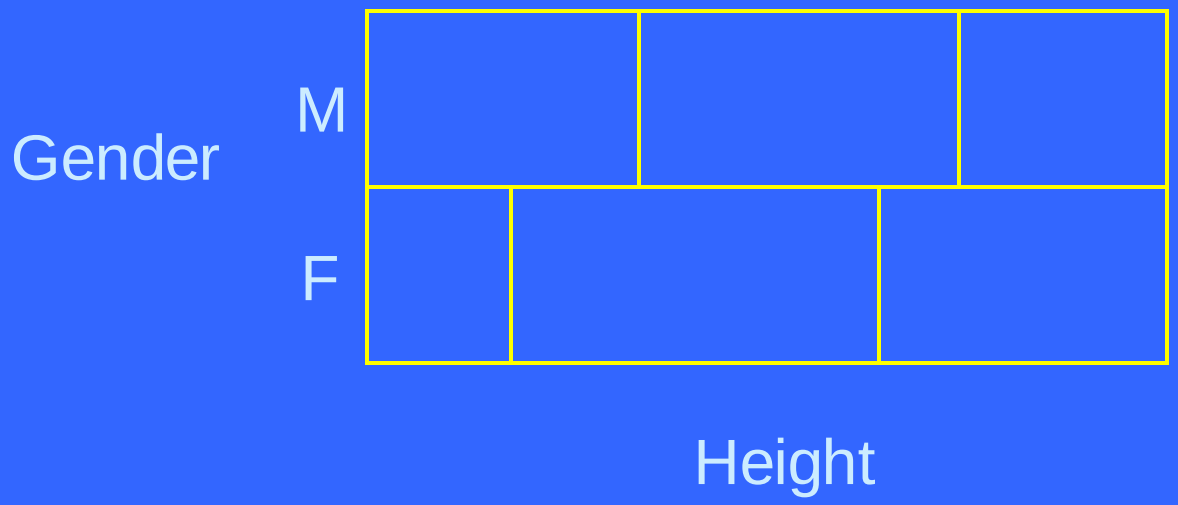
Decision or Classification Tree is a tree associated with D such that

- Each internal node is labeled with attribute, A_i
- Each arc is labeled with predicate which can be applied to attribute at parent
- Each leaf node is labeled with a class, C_j

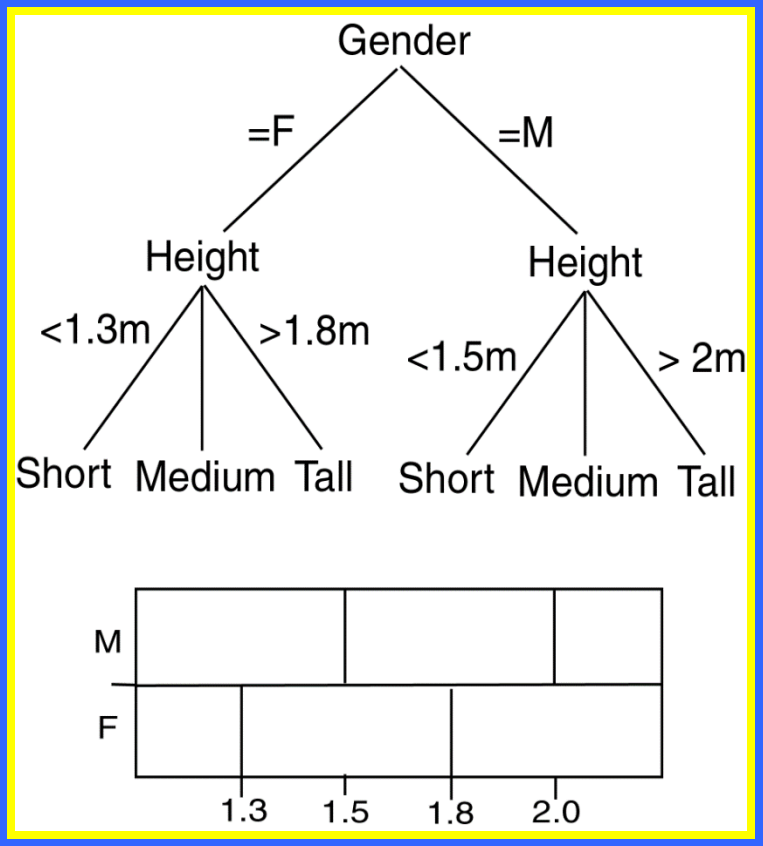
DT Induction

```
Input:
    D       //Training data
Output:
    T       //Decision Tree
DTBuild Algorithm:
            //Simplistic algorithm to illustrate naive approach to building DT
    T = ∅;
    Determine best splitting criterion;
    T = Create root node node and label with splitting attribute;
    T = Add arc to root node for each split predicate and label;
    for each arc do
        D = Database created by applying splitting predicate to D;
        if stopping point reached for this path then
            T' = Create leaf node and label with appropriate class;
        else
            T' = DTBuild(D);
        T = Add T' to arc;
```

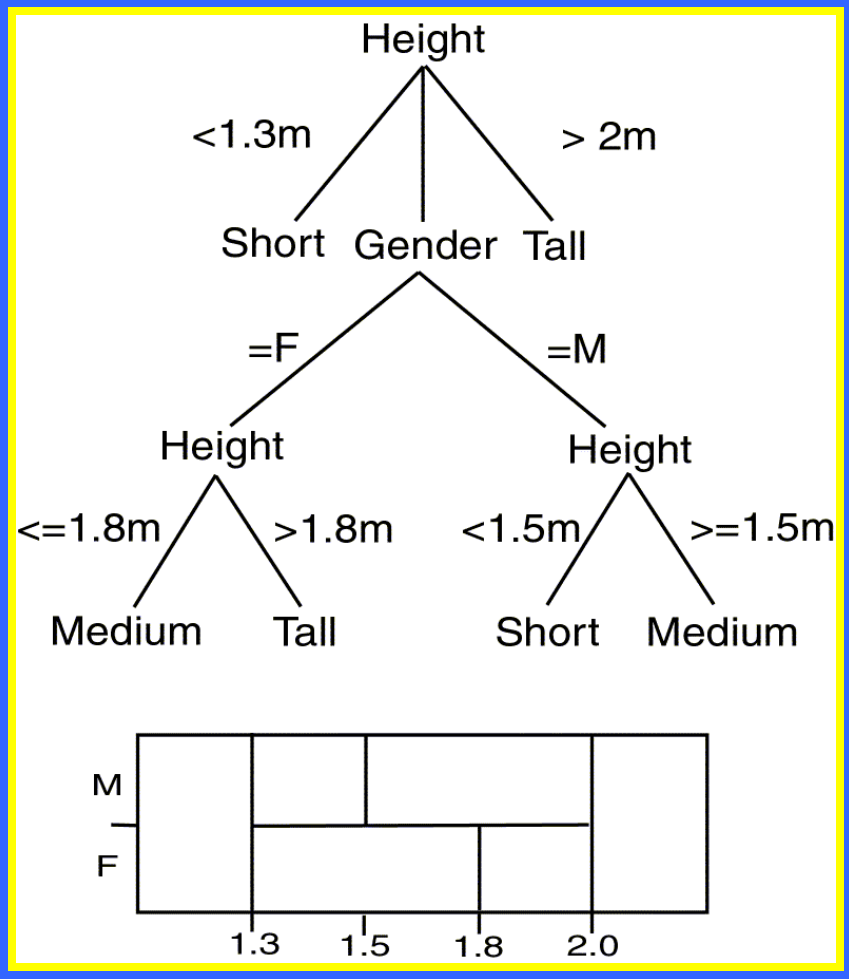
DT Splits Area



Comparing DTs



Balanced



Deep

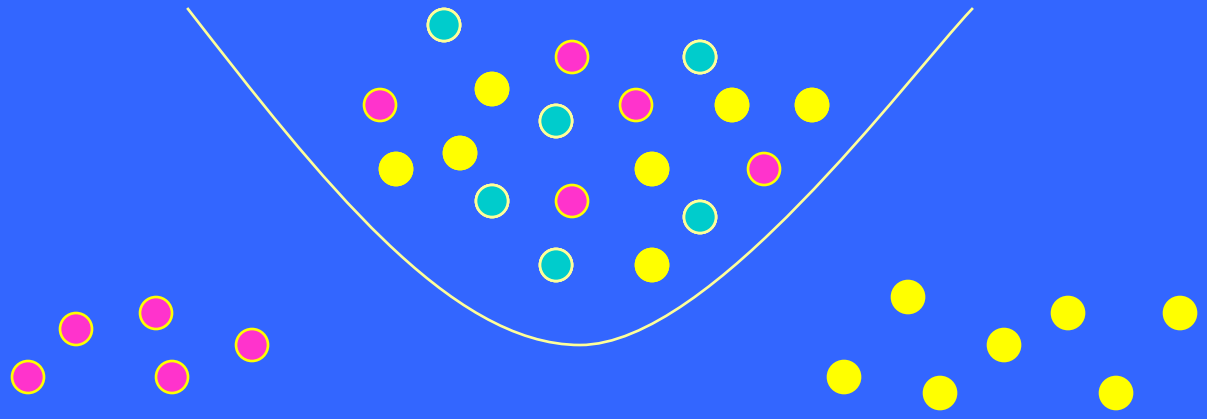
DT Issues

- Choosing Splitting Attributes
- Ordering of Splitting Attributes
- Splits
- Tree Structure
- Stopping Criteria
- Training Data
- Pruning

Decision Tree Induction is often based on Information Theory

So

Information



DT Induction

- When all the marbles in the bowl are mixed up, little information is given.
- When the marbles in the bowl are all from one class and those in the other two classes are on either side, more information is given.

Use this approach with DT Induction !

Information/Entropy

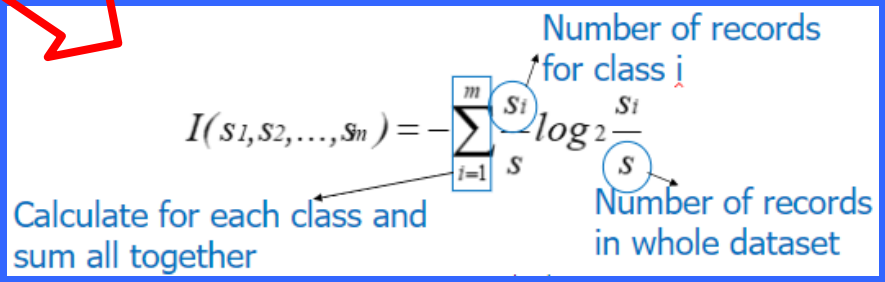
- Given probabilitites p_1, p_2, .., p_s whose sum is 1, ***Entropy*** is defined as:

$$H(p_1, p_2, ..., p_s) = \sum_{i=1}^{s} (p_i log(1/p_i))$$

Equivalent

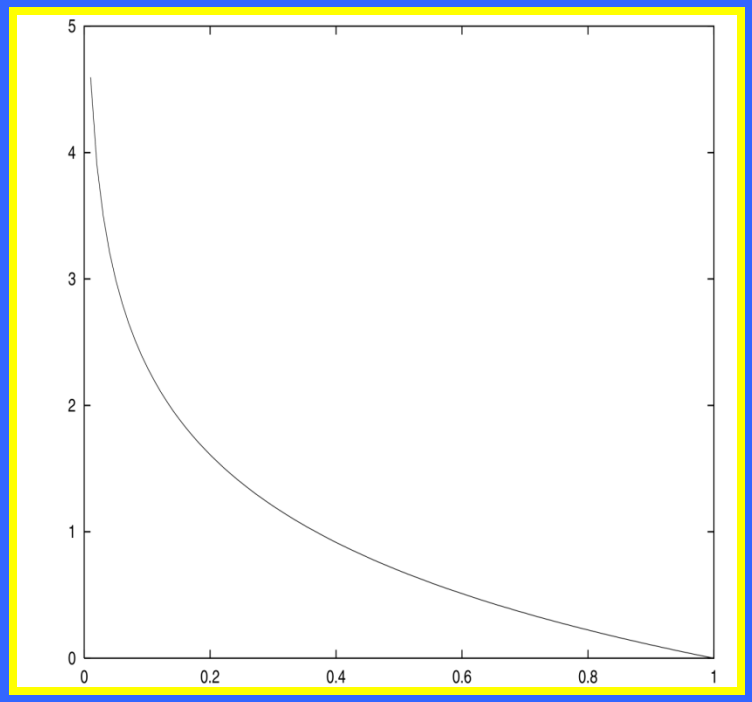
$$p_i = \frac{s_i}{s}$$

Previously:



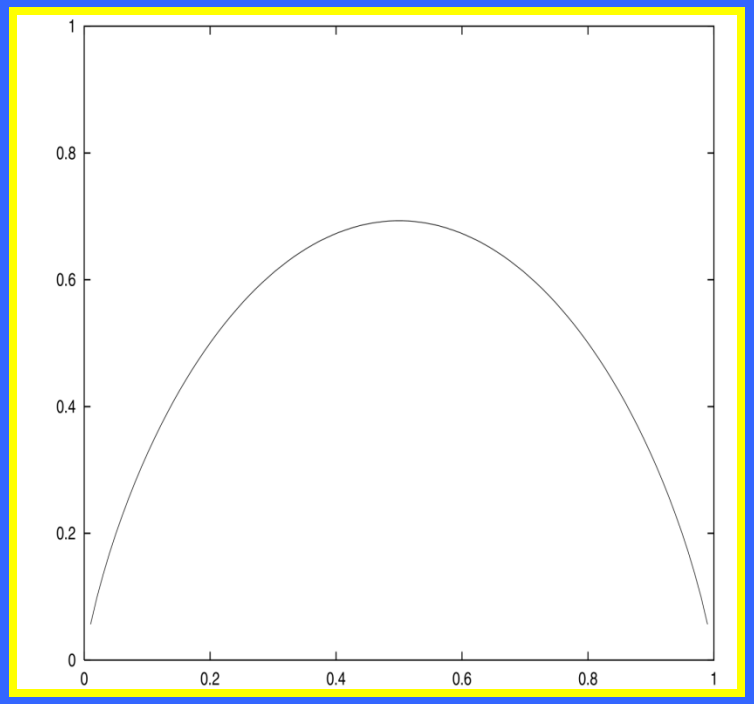
- Entropy measures the amount of randomness or surprise or uncertainty.
- Goal in classification
  - no surprise
  - entropy = 0

Entropy



log (1/p)

H



P

H(p,1-p)

ID3

- Creates tree using information theory concepts and tries to reduce expected number of comparison.
- ID3 chooses split attribute with the highest information gain:

$$Gain(D,S) = H(D) - \sum_{i=1}^{s} P(D_i)H(D_i)$$

Height Example Data

Name	Gender	Height	Output1	Output2
Kristina	F	1.6m	Short	Medium
Jim	M	2m	Tall	Medium
Maggie	F	1.9m	Medium	Tall
Martha	F	1.88m	Medium	Tall
Stephanie	F	1.7m	Short	Medium
Bob	M	1.85m	Medium	Medium
Kathy	F	1.6m	Short	Medium
Dave	M	1.7m	Short	Medium
Worth	M	2.2m	Tall	Tall
Steven	M	2.1m	Tall	Tall
Debbie	F	1.8m	Medium	Medium
Todd	M	1.95m	Medium	Medium
Kim	F	1.9m	Medium	Tall
Amy	F	1.8m	Medium	Medium
Wynette	F	1.75m	Medium	Medium

ID3 Example (Output1)

- Starting state entropy: 4 short, 8 medium, 3 tall

4/15 log(15/4) + 8/15 log(15/8) + 3/15 log(15/3) = 0.4384

- Gain using gender:
  - 9 females: 3 short, 6 medium
  - 6 males: 1 short, 2 medium, 3 tall
  - – Female: 3/9 log(9/3)+6/9 log(9/6)=0.2764
  - – Male: 1/6 (log 6/1) + 2/6 log(6/2) + 3/6 log(6/3) = 0.4392
  - – Weighted sum: (9/15)(0.2764) + (6/15)(0.4392) = 0.34152
  - – Gain: 0.4384 – 0.34152 = 0.09688

ID3 Example (Output1)

- Looking at the height attribute (discretize)
  - (0. 1.6] : 2
  - (1.6, 1.7] : 2
  - (1.7, 1.8] : 3
  - (1.8, 1.9] : 4
  - (1.9, 2.0] : 2
  - (2.0, ∞) : 2

[ ] : inclusive
( ) : exclusive

Name	Gender	Height
Kristina	F	1.6m
Jim	M	2m
Maggie	F	1.9m
Martha	F	1.88m
Stephanie	F	1.7m
Bob	M	1.85m
Kathy	F	1.6m
Dave	M	1.7m
Worth	M	2.2m
Steven	M	2.1m
Debbie	F	1.8m
Todd	M	1.95m
Kim	F	1.9m
Amy	F	1.8m
Wynette	F	1.75m

ID3 Example (Output1)

- Entropy of each height "value"
  - (0. 1.6] : 2
    2/2(0) + 0 + 0 = 0
  - (1.6, 1.7] : 2
  - (2/2(0) + 0 + 0) = 0
  - (1.7, 1.8] : 3
    (0 + 3/3(0) + 0) = 0
  - (1.8, 1.9] : 4
    (0 + 4/4(0) + 0) = 0
  - (1.9, 2.0] : 2
    (0 + ½(0.301) + ½ (0.301) = 0.301
  - (2.0, ∞) : 2
    (0 + 0 + 2/2(0)) = 0

Name	Gender	Height	Output1
Kristina	F	1.6m	Short
Jim	M	2m	Tall
Maggie	F	1.9m	Medium
Martha	F	1.88m	Medium
Stephanie	F	1.7m	Short
Bob	M	1.85m	Medium
Kathy	F	1.6m	Short
Dave	M	1.7m	Short
Worth	M	2.2m	Tall
Steven	M	2.1m	Tall
Debbie	F	1.8m	Medium
Todd	M	1.95m	Medium
Kim	F	1.9m	Medium
Amy	F	1.8m	Medium
Wynette	F	1.75m	Medium

ID3 Example (Output1)

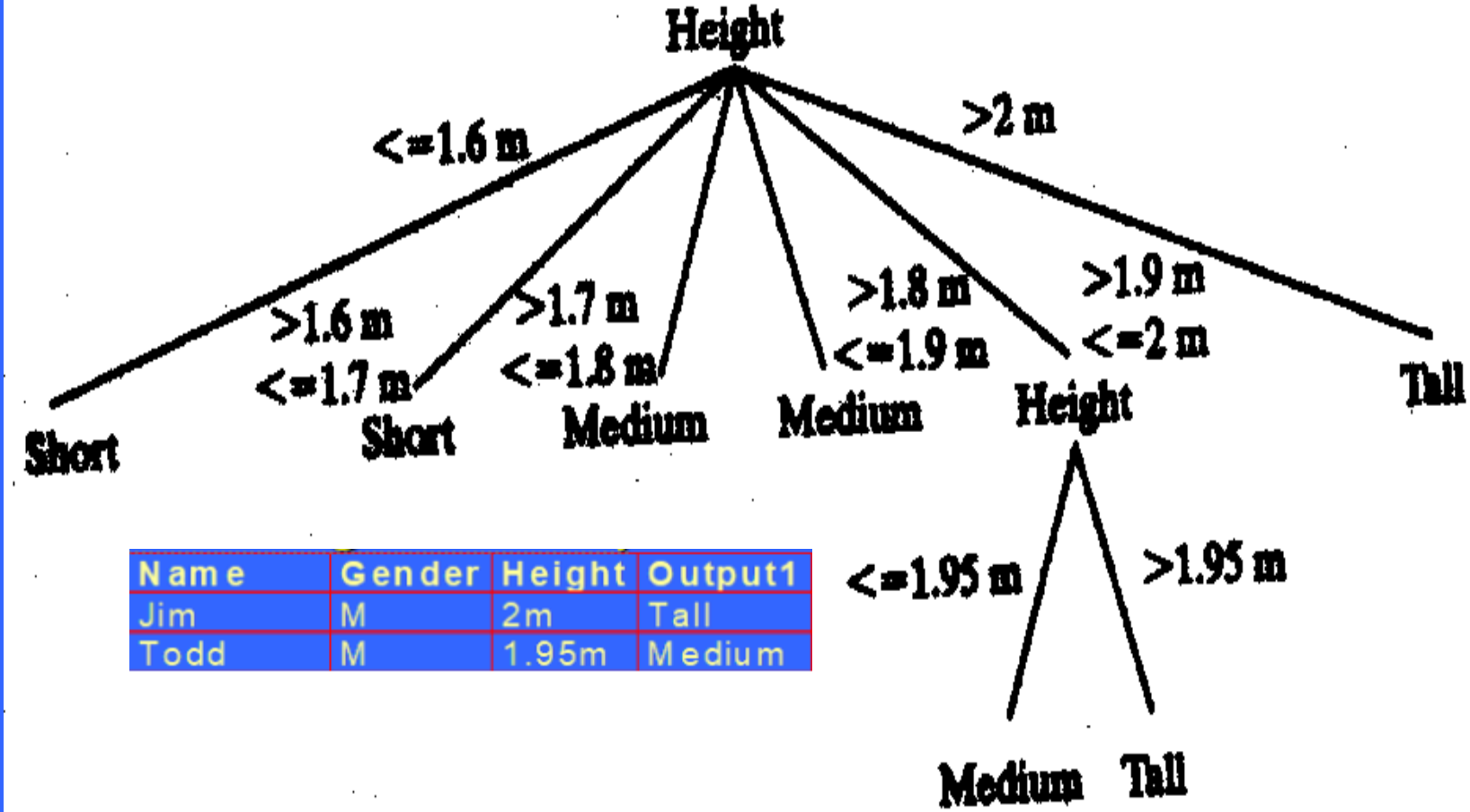
- The gain in entropy by using the height attribute is thus
- 0.4384 – 2/15 (0.301) = 0.3983

Note:

Starting state entropy: 4 short, 8 medium, 3 tall

4/15 log(15/4) + 8/15 log(15/8) + 3/15 log(15/3) = 0.4384

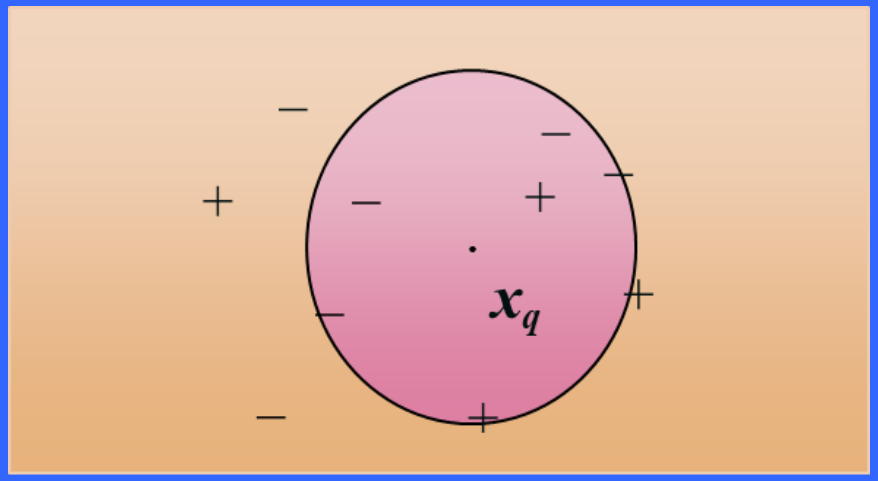
Decision Tree



Name	Gender	Height	Output1
Jim	M	2m	Tall
Todd	M	1.95m	Medium

k-Nearest Neighbors

- Non-parametric, supervised learning classifier
  - uses proximity to make classifications or predictions about the grouping of an individual data point.
  - Technically, does not need any training on data
- Target function could be discrete- or real- valued
- For discrete-valued, k-NN returns the most common value/class among the k training examples nearest to xq



Visual look at k-NN

- All instances correspond to points in the n-D space
- The nearest neighbor are defined in terms of Euclidean distance, dist(A, B)

$$\text{Euclidean Distance} = |X - Y| = \sqrt{\sum_{i=1}^{i=n} (x_i - y_i)^2}$$

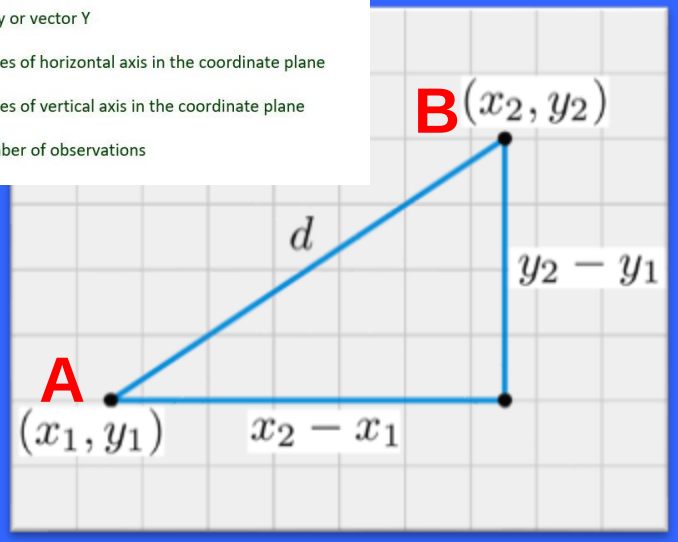
X: Array or vector X

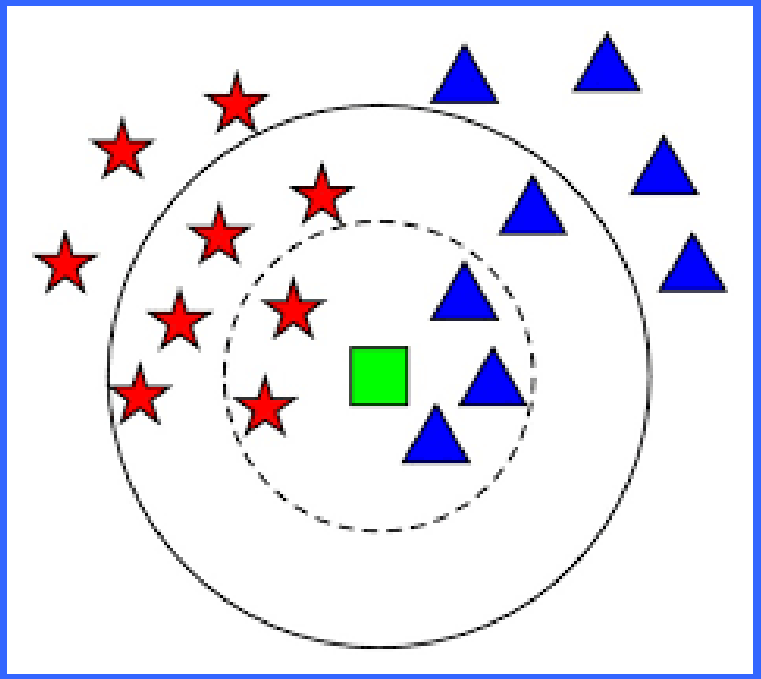
Y: Array or vector Y

x_i: Values of horizontal axis in the coordinate plane

y_i: Values of vertical axis in the coordinate plane

n: Number of observations



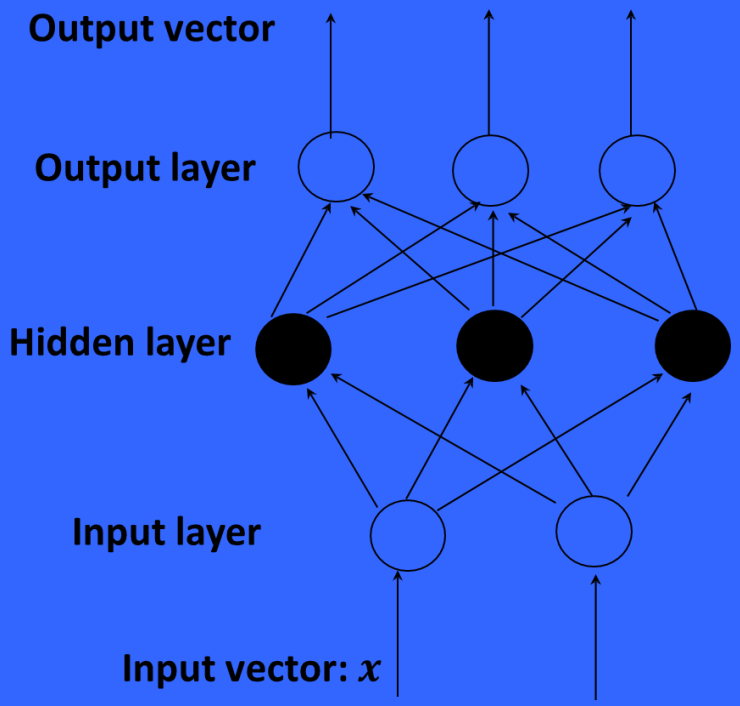


KNN Discussion

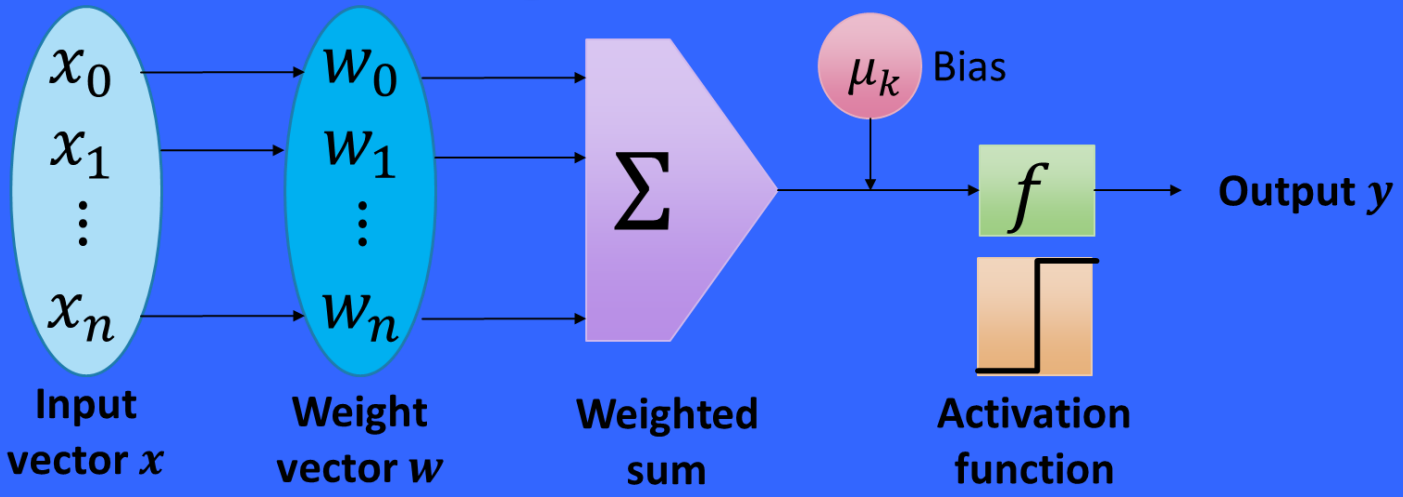
- k-NN for real-valued prediction for a given unknown tuple
  - Returns the mean values of the k nearest neighbors
- Distance-weighted nearest neighbor algorithm
  - Weight the contribution of each of the k neighbors according to their distance to the query xq
    - » Give greater weight to closer neighbors
- Robust to noisy data by averaging k-nearest neighbors
- Curse of dimensionality: distance between neighbors could be dominated by irrelevant attributes
  - To overcome it, axes stretch or elimination of the least relevant attributes

Neural Networks

- A set of connected input/output units where each connection has a weight associated with it
- During the learning phase, the network learns by adjusting the weights so as to be able to predict the correct class label of the input tuples
- Also referred to as connectionist learning due to the connections between units
- **Backpropagation**: A neural network learning algorithm



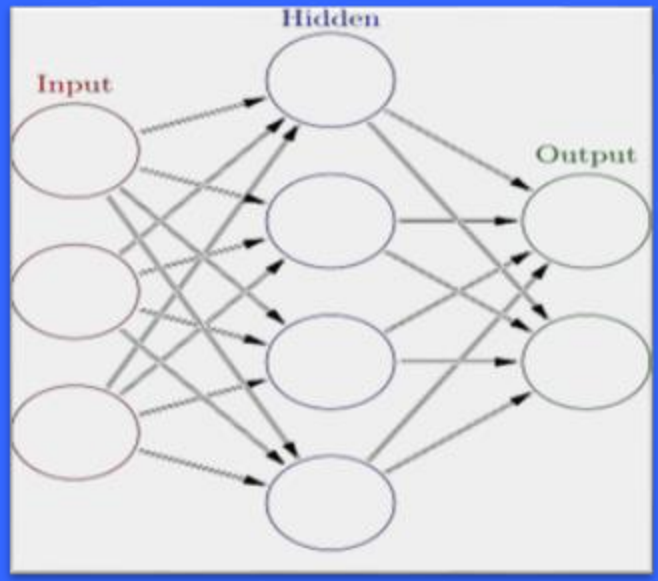
Neuron: A Hidden/Output Layer Unit



- An n-dimensional input vector x is mapped into variable y by means of the scalar product (multiplication of vectors) and a nonlinear function mapping (for complex problems)
- The inputs to unit are outputs from the previous layer. They are multiplied by their corresponding weights to form a weighted sum, which is added to the bias associated with unit. Then a nonlinear activation function is applied to it.

Multi-Layer Perceptron (MLP)

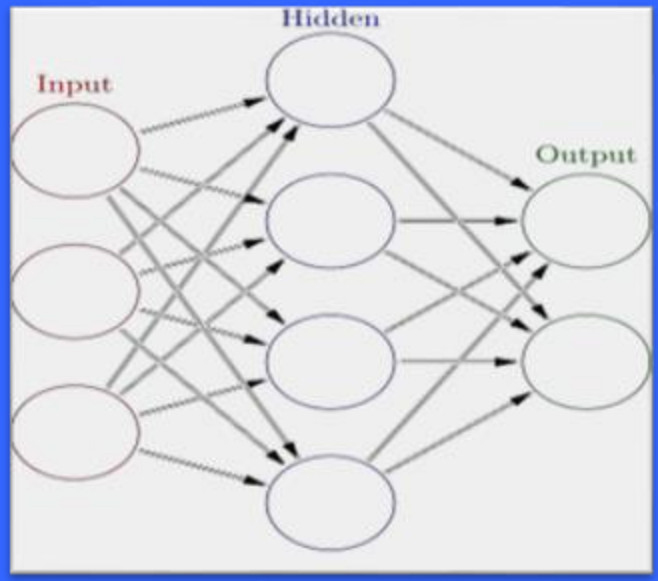
- The inputs to the network correspond to the attributes measured for each training tuple
- Inputs are fed simultaneously into the units making up the input layer
- They are then weighted and fed simultaneously to a hidden layer
- The number of hidden layers is arbitrary, although usually only one (a shallow net)



- The weighted outputs of the last hidden layer are input to units making up the output layer, which emits the network's prediction
- The network is feed-forward: None of the weights cycles back to an input unit or to an output unit of a previous layer

Training a MLP

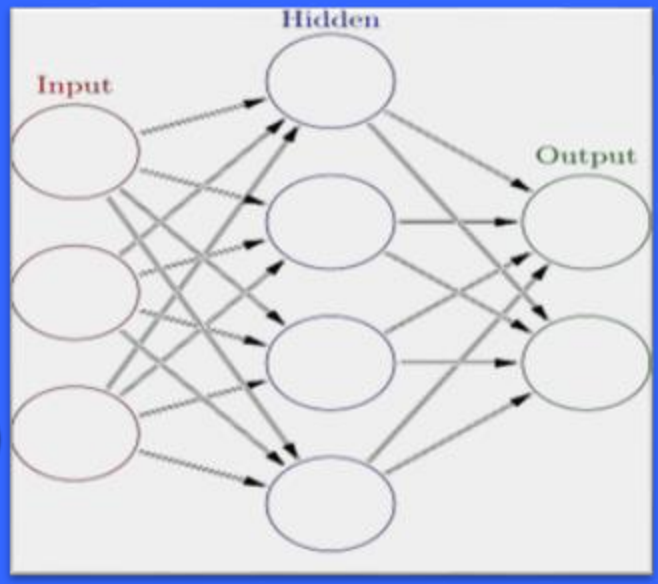
- Back propagation: Reset weights on the "front" neural units and this is sometimes done in combination with training where the correct result is known
- Iteratively process a set of training tuples & compare the network's prediction with the actual known target value
- For each training tuple, the weights are modified to minimize the mean squared error between the network's prediction and the actual target value



- Modifications are made in the “backwards” direction: from the output layer, through each hidden layer down to the first hidden layer, hence “backpropagation”

Training Steps

- Overall steps:
  - Initialize weights to small random numbers, associated with biases
  - Propagate the inputs forward (by applying activation function)
  - Backpropagate the error (by updating weights and biases)
  - Terminating condition (when error is very small, etc.)



Strengths and Weaknesses

- Advantages
  - prediction accuracy is generally high
  - robust, works when training examples contain errors
  - Can process numeric and categorical data
  - fast evaluation of the learned target function
- Criticism
  - long training time
  - difficult to understand the learned function (weights)
  - not easy to incorporate domain knowledge
  - Lack explanation capabilities.
  - NN is like a black box.