Machine Learning

Chapter 18.1-18.3

Today's Class

- Machine learning
    - What is ML?
    - Inductive learning
        - Supervised
        - Unsupervised
    - Decision trees
- Later we'll cover Bayesian learning, naïve Bayes, and BN learning

Why Learn?

- Understand and improve efficiency of human learning
  - Use to improve methods for teaching and tutoring people (e.g., better computer-aided instruction)
- Discover new things or structure that were previously unknown to humans
  - Examples: data mining, scientific discovery
- Fill in skeletal or incomplete specifications about a domain
  - Large, complex AI systems cannot be completely derived by hand and require dynamic updating to incorporate new information.
  - Learning new characteristics expands the domain or expertise and lessens the "brittleness" of the system
- Build software agents that can adapt to their users or to other software agents

Major Paradigms of Machine Learning

- **Rote learning** – One-to-one mapping from inputs to stored representation. "Learning by memorization." Association-based storage and retrieval.
- **Induction** – Use specific examples to reach general conclusions
- **Clustering** – Unsupervised identification of natural groups in data
- **Analogy** – Determine correspondence between two different representations
- **Discovery** – Unsupervised, specific goal not given
- **Genetic algorithms** – "Evolutionary" search techniques, based on an analogy to "survival of the fittest"
- **Reinforcement** – Feedback (positive or negative reward) given at the end of a sequence of steps
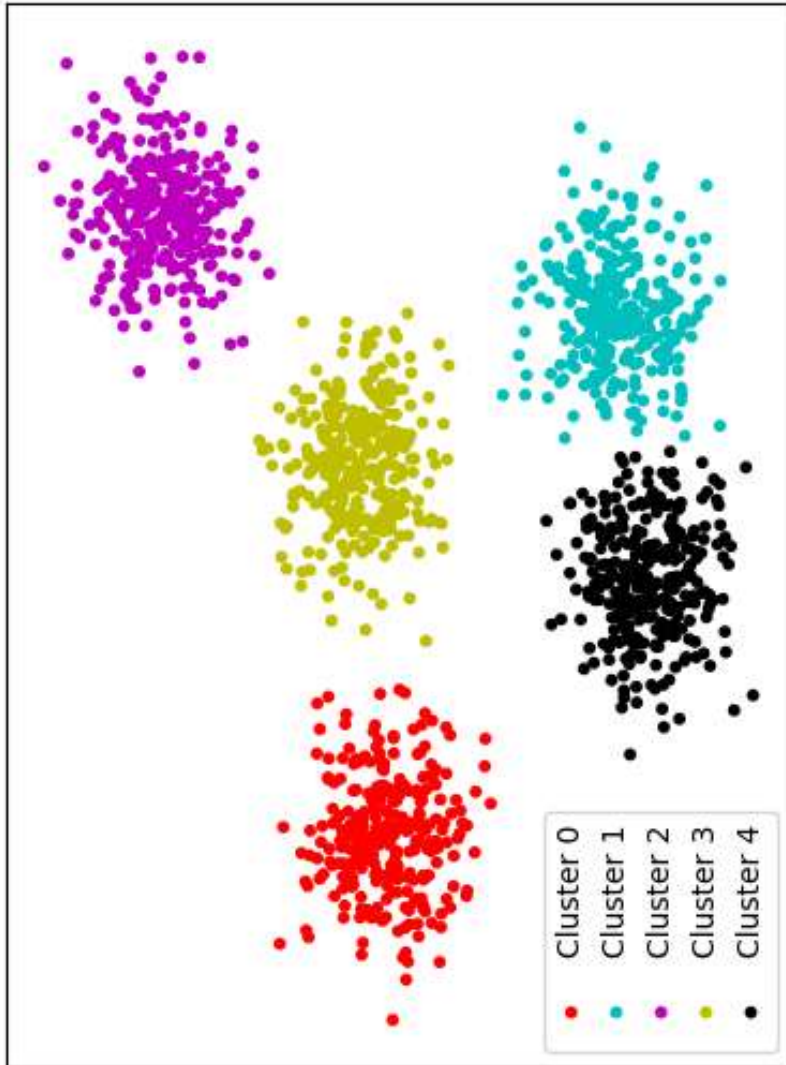
Regression
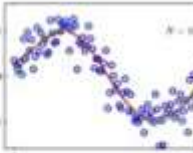
What is the temperature going to be tomorrow?
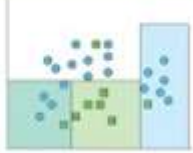
PREDICTION

84°

Fahrenheit
°F

Classification

Will it be Cold or Hot tomorrow?

PREDICTION

COLD   HOT

Fahrenheit
°F
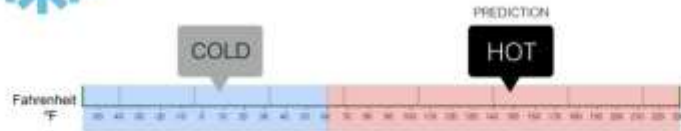
Predictive methods	Descriptive methods
Classification	**Clustering**
Learns a method for predicting the instance class from pre-labeled (classified) instances	Finds "natural" grouping of instances given un-labeled data
Regression	**Association Rules**
An attempt to predict a continuous attribute	Method for discovering interesting relations between variables in large DBs

Classification
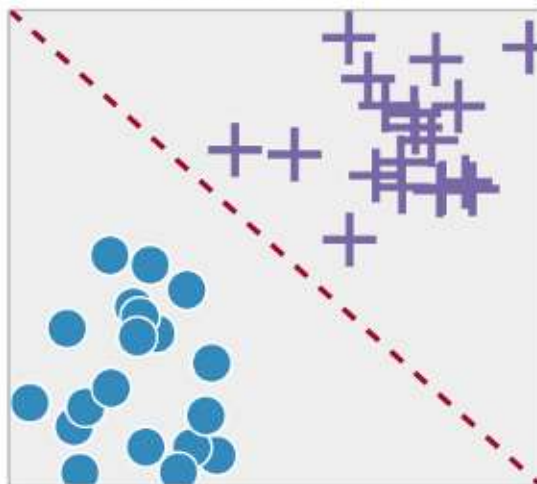
Regression
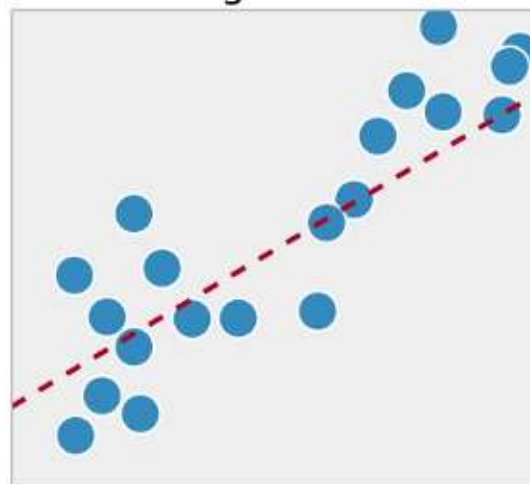
Classification Learning: Definition

- Given a collection of records (*training set*)
  - Each record contains a set of *attributes*, one of the attributes is the *class*
- Find a *model* for the class attribute as a function of the values of the other attributes
- Goal: <u>previously unseen</u> records should be assigned a class as accurately as possible
  - Use *test set* to estimate the accuracy of the model
  - Often, the given data set is divided into training and test sets, with training set used to build the model and test set used to validate it

Illustrating Classification Learning

Examples of Classification Task

- Predicting tumor cells as benign or malignant

- Classifying credit card transactions as legitimate or fraudulent

- Classifying secondary structures of protein as alpha-helix, beta-sheet, or random coil

- Categorizing news stories as finance, weather, entertainment, sports, etc.
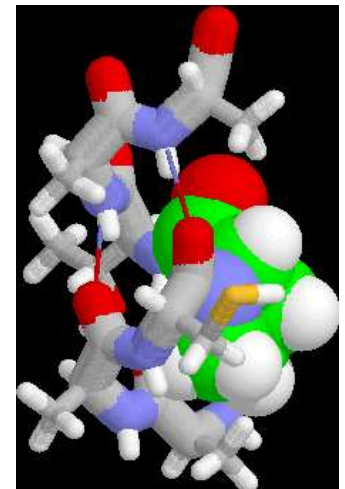
Inductive Learning and Bias

(a)   (b)   (c)   (d)
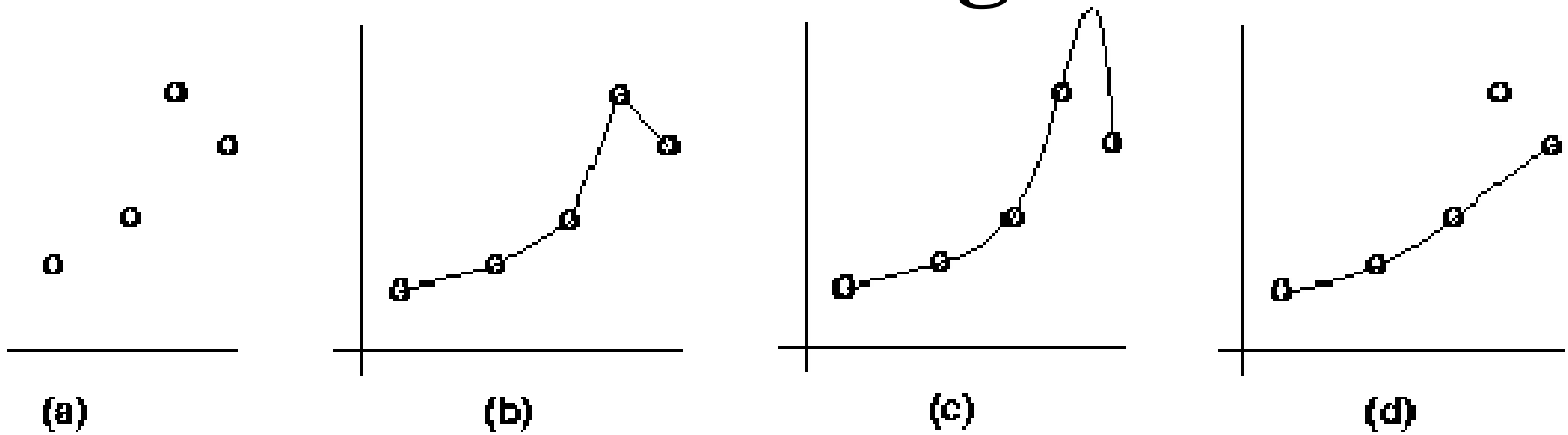
- Suppose that we want to learn a function f(x) = y and we are given some sample (x,y) pairs, as in figure (a)

- There are several hypotheses we could make about this function, e.g.: (b), (c) and (d)

- A preference for one over the others reveals the **bias** of our learning technique, e.g.:
  – prefer piece-wise functions (b)
  – prefer a smooth function (c)
  – prefer a simple function and treat outliers as noise (d)

Inductive Learning as Search

- Instance space I defines the language for the training and test instances
  - Typically, but not always, each instance i $\in$ I is a feature vector
  - Features are also sometimes called attributes or variables
  - I: V_1 x V_2 x … x V_k, i = (v_1, v_2, …, v_k)
- Class variable C gives an instance's class (to be predicted)
- Model space M defines the possible classifiers
  - M: I $\rightarrow$ C, M = {m_1, … m_n} (possibly infinite)
  - Model space is sometimes, but not always, defined in terms of the same features as the instance space
- Training data can be used to direct the search for a good (consistent, complete, simple) hypothesis in the model space

Model Spaces

- **Decision trees**
  - Partition the instance space into axis-parallel regions, labeled with class value
- Nearest-neighbor classifiers
  - Partition the instance space into regions defined by the centroid instances (or cluster of k instances)
- Bayesian networks (probabilistic dependencies of class on attributes)
  - Naïve Bayes: special case of BNs where class → each attribute
- Neural networks
  - Nonlinear feed-forward functions of attribute values
- Support vector machines
  - Find a separating plane in a high-dimensional feature space
- Associative rules (feature values → class)
- First-order logical rules

Learning Decision Trees

- Goal: Build a **decision tree** to classify examples as positive or negative instances of a concept using supervised learning from a training set

- A **decision tree** is a tree where
  - each non-leaf node has associated with it an attribute (feature)
  - each leaf node has associated with it a classification (+ or -)
  - each arc has associated with it one of the possible values of the attribute at the node from which the arc is directed

- Generalization: allow for >2 classes
  - e.g., {sell, hold, buy}
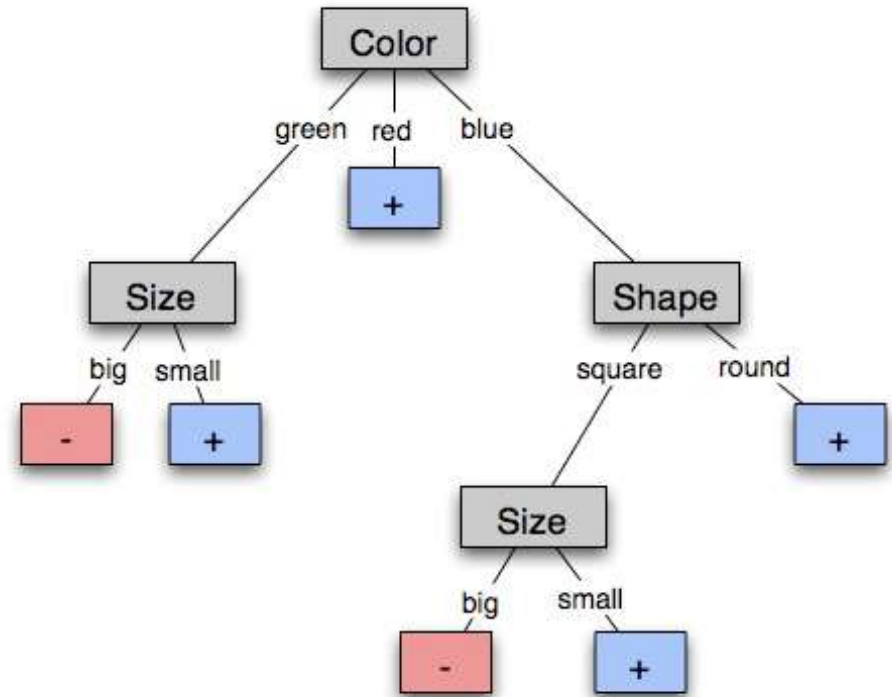
Example of a Decision Tree

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Training Data
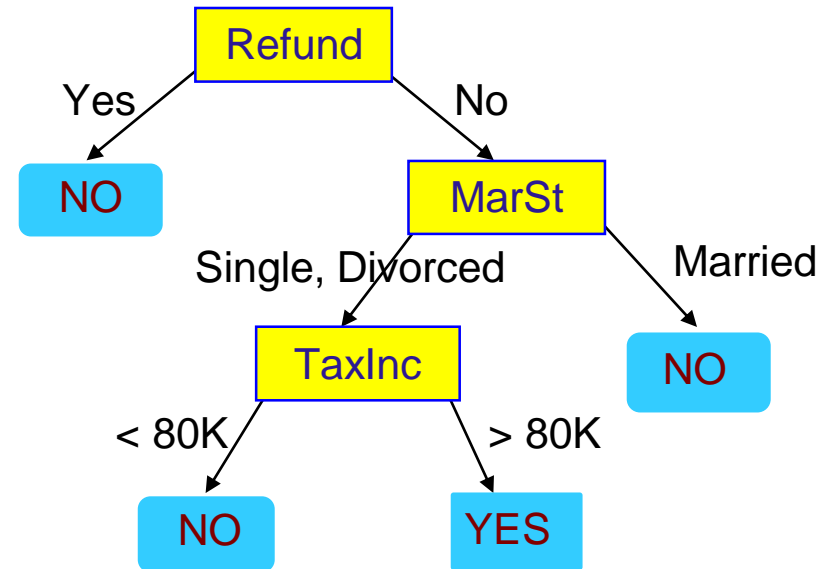
Model: Decision Tree

Refund
Yes → NO
No → MarSt
Single, Divorced → TaxInc
Married → NO
TaxInc < 80K → NO
TaxInc > 80K → YES

Apply Model to Test Data

Start at the root of tree

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Refund

Yes → NO

No → MarSt

Single, Divorced → TaxInc

Married → NO

< 80K → NO

> 80K → YES

Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

```
          Refund
        /        \
      Yes         No
      /            \
    NO            MarSt
              /          \
   Single, Divorced      Married
          /                 \
       TaxInc              NO
      /      \
   < 80K    > 80K
    /          \
   NO         YES
```

Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?
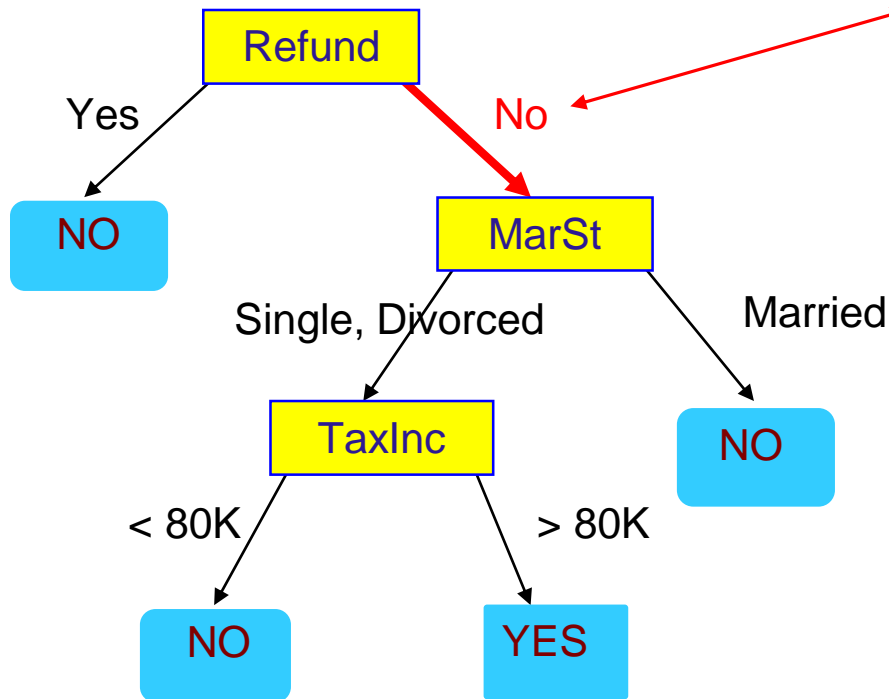
Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Assign Cheat to "No"

Information Theory

- Information is measured in bits
- Information conveyed by a message depends on its probability
- With n equally probable possible *messages*, the probability p of each is *1/n*
- Information conveyed by message is $\log_2(n) = -\log_2(p)$
  - e.g., with 16 messages, then $\log_2 (16) = 4$ and we need 4 bits to identify/send each message
- Given probability distribution for n messages $P = (p_1, p_2 \ldots p_n)$, the information conveyed by distribution (aka *entropy* of P) is:

$$I(P) = -(p_1 * \log_2 (p_1) + p_2 * \log_2 (p_2) + .. + p_n * \log_2 (p_n))$$

probability of msg 2   info in msg 2

Entropy

- Entropy: $H(S) = - p_{(+)} \log_2 p_{(+)} - p_{(-)} \log_2 p_{(-)}$ bits
  - S ... subset of training examples
  - $p_{(+)}$ / $p_{(-)}$ ... % of positive / negative examples in S
- Interpretation: assume item X belongs to S
  - how many bits need to tell if X positive or negative
- impure (3 yes / 3 no):

$$H(S) = -\frac{3}{6}\log_2 \frac{3}{6} - \frac{3}{6}\log_2 \frac{3}{6} = 1 \text{ bits}$$

- pure set (4 yes / 0 no):

$$H(S) = -\frac{4}{4}\log_2 \frac{4}{4} - \frac{0}{4}\log_2 \frac{0}{4} = 0 \text{ bits}$$
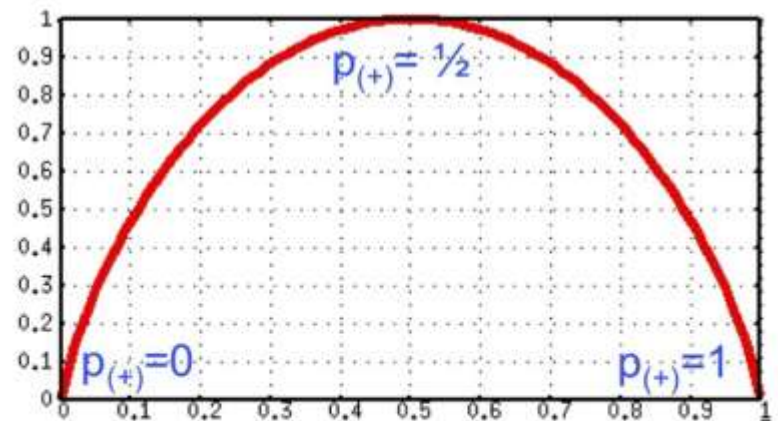
Information Gain

Which test is more informative?

Split over whether Balance exceeds 50K
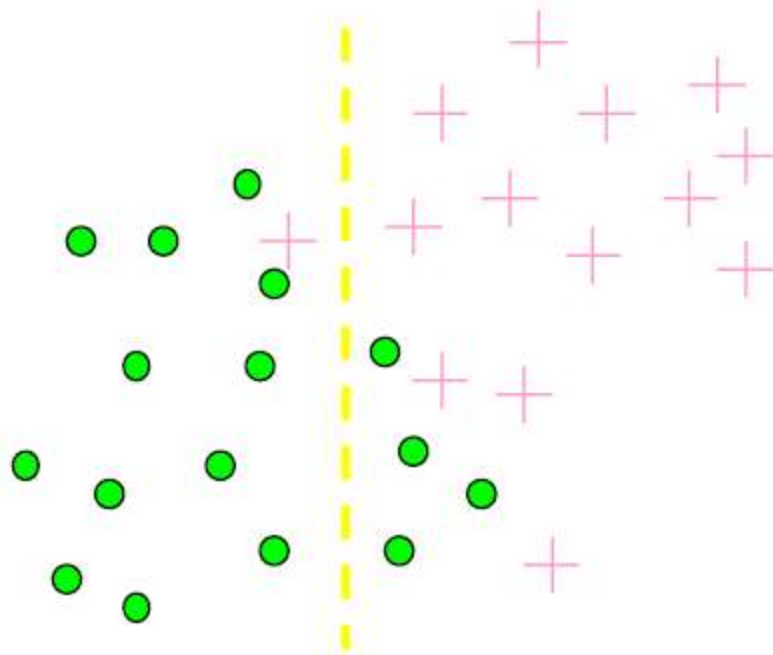
Split over whether applicant is employed

ess or equal 50K   Over 50K
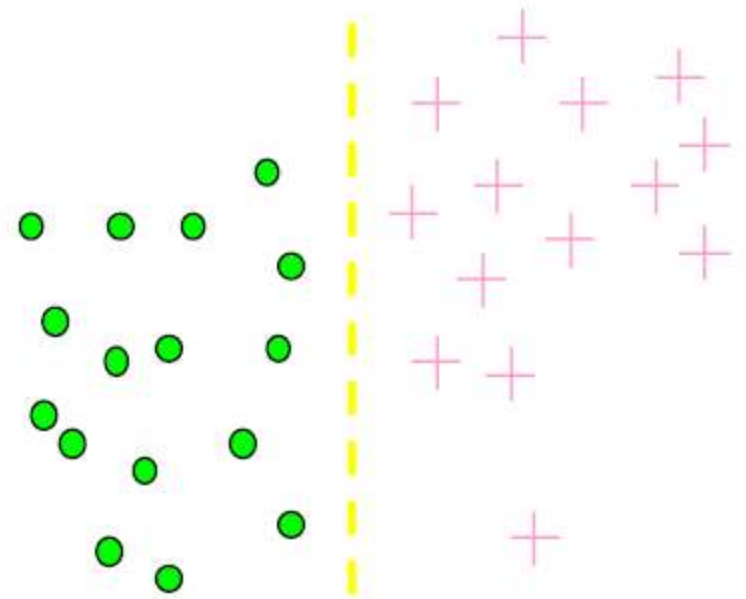
Unemployed   Employed

21

Information Gain

Impurity/Entropy (informal)

– Measures the level of **impurity** in a group of examples
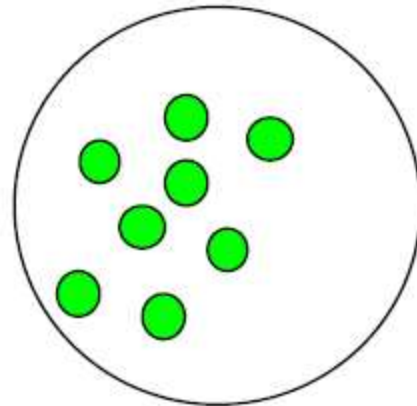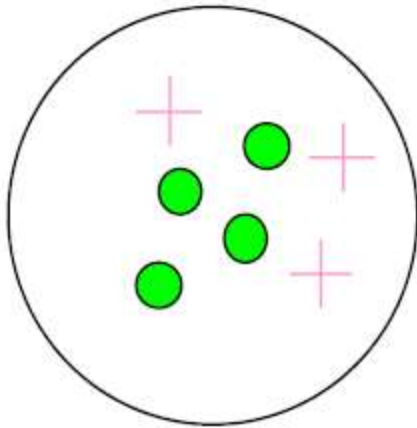
Impurity

Very impure group
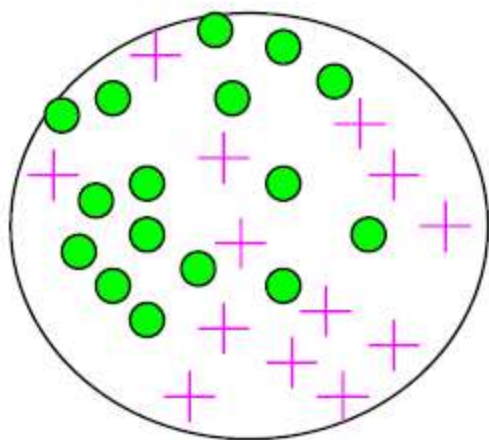
Less impure
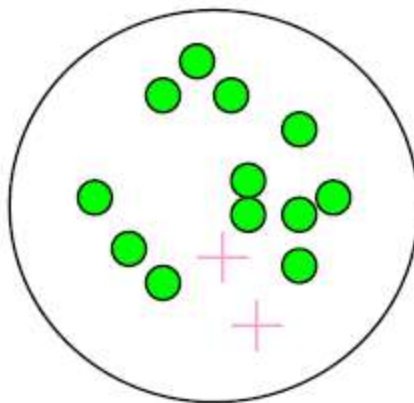
Minimum impurity
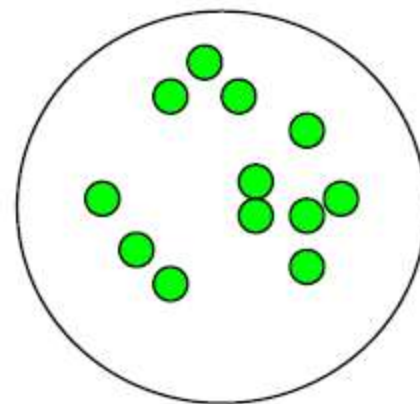
Entropy: a common way to measure impurity

- **Entropy =** $$\sum_i - p_i \log_2 p_i$$

  p_i is the probability of class i

  Compute it as the proportion of class i in the set.
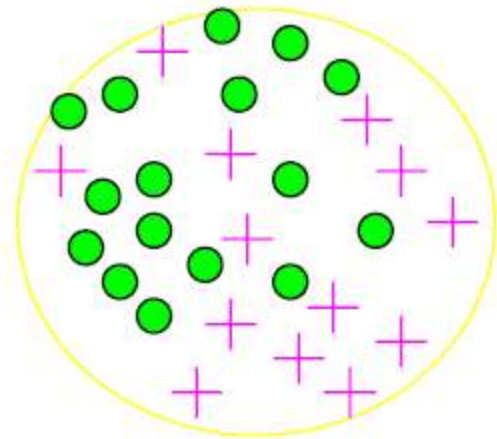
- **Entropy comes from information theory. The higher the entropy the more the information content.**

What does that mean for learning from examples?

2-Class Cases:

$$\text{Entropy} \quad H(x) = -\sum_{i=1}^{n} P(x=i) \log_2 P(x=i)$$

- **What is the entropy of a group in which all examples belong to the same class?**
  - entropy = $- 1 \log_2 1 = 0$

  not a good training set for learning

Minimum impurity
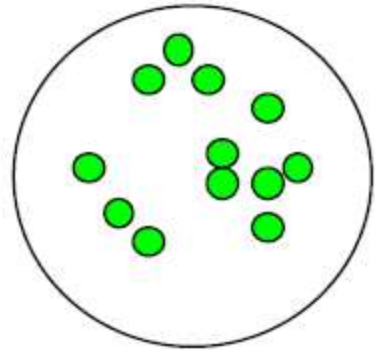
- **What is the entropy of a group with 50% in either class?**
  - entropy = $-0.5 \log_2 0.5 - 0.5 \log_2 0.5 = 1$

  good training set for learning

Maximum impurity

25
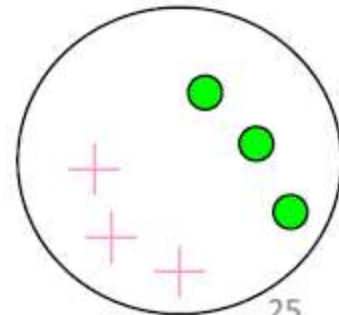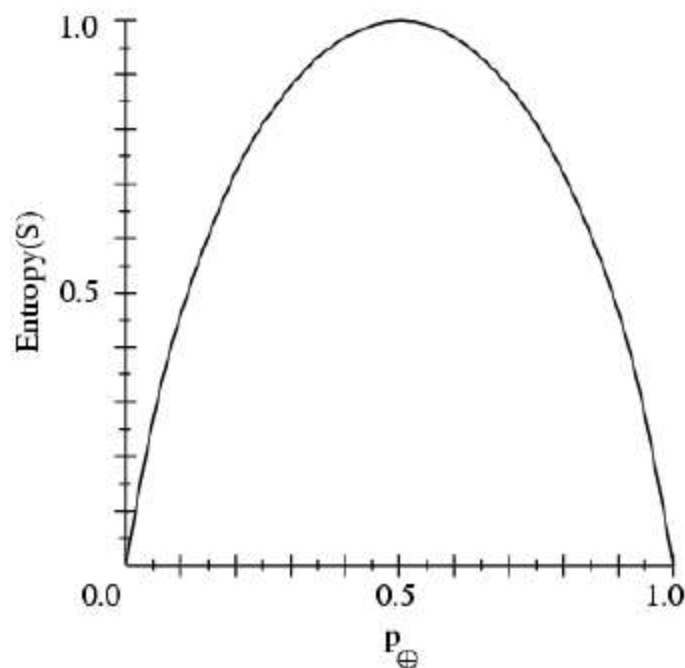
Sample Entropy

- S is a sample of training examples
- $p_\oplus$ is the proportion of positive examples in S
- $p_\ominus$ is the proportion of negative examples in S
- Entropy measures the impurity of S

$$H(S) \equiv -p_\oplus \log_2 p_\oplus - p_\ominus \log_2 p_\ominus$$

Information Gain

- We want to determine which attribute in a given set of training feature vectors is most useful for discriminating between the classes to be learned.

- Information gain tells us how important a given attribute of the feature vectors is.

- We will use it to decide the ordering of attributes in the nodes of a decision tree.

From Entropy to Information Gain

Entropy $H(X)$ of a random variable X

$$H(X) = -\sum_{i=1}^{n} P(X=i)\log_2 P(X=i)$$

Specific conditional entropy $H(X/Y=v)$ of X given $Y=v$:

$$H(X|Y=v) = -\sum_{i=1}^{n} P(X=i|Y=v)\log_2 P(X=i|Y=v)$$

Conditional entropy $H(X/Y)$ of X given Y :

$$H(X|Y) = \sum_{v \in values(Y)} P(Y=v)H(X|Y=v)$$

Mututal information (aka Information Gain) of X and Y :

$$I(X,Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$$

Decision Tree

Day	Outlook	Temp.	Humidity	Wind	Decision
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

Decision Tree

- We can summarize the ID3 algorithm as illustrated below

$$Entropy(S) = \sum - p(I) . log_2 p(I)$$

$$Gain(S, A) = Entropy(S) - \sum [p(S|A) . Entropy(S|A)]$$

Entropy

We need to calculate the entropy first. Decision column consists of 14 instances and includes two labels: yes and no. There are 9 decisions labeled yes, and 5 decisions labeled no.

$$Entropy(Decision) = - p(Yes) . log_2 p(Yes) - p(No) . log_2 p(No)$$

$$Entropy(Decision) = - (9/14) . log_2(9/14) - (5/14) . log_2(5/14) = 0.940$$

Now, we need to find the most dominant factor for decisioning.

Decision Tree

Wind factor on decision

Gain(Decision, Wind) = Entropy(Decision) – ∑ [p(Decision|Wind) . Entropy (Decision|Wind)]

Wind attribute has two labels: weak and strong. We would reflect it to the formula.

Gain(Decision, Wind) = Entropy(Decision) – [p(Decision|Wind=Weak) . Entropy (Decision|Wind=Weak)] – [p(Decision|Wind=Strong) . Entropy (Decision|Wind=Strong)]

Now, we need to calculate (Decision|Wind=Weak) and (Decision|Wind=Strong) respectively.

Decision Tree

Weak wind factor on decision

Day	Outlook	Temp.	Humidity	Wind	Decision
1	Sunny	Hot	High	Weak	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
13	Overcast	Hot	Normal	Weak	Yes

There are 8 instances for weak wind. Decision of 2 items are no and 6 items are yes as illustrated below.

1- $Entropy(Decision|Wind=Weak) = - p(No) \cdot \log_2 p(No) - p(Yes) \cdot \log_2 p(Yes)$

2- $Entropy(Decision|Wind=Weak) = - (2/8) \cdot \log_2(2/8) - (6/8) \cdot \log_2(6/8) = 0.811$

Decision Tree

Strong wind factor on decision

Day	Outlook	Temp.	Humidity	Wind	Decision
2	Sunny	Hot	High	Strong	No
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
14	Rain	Mild	High	Strong	No

Here, there are 6 instances for strong wind. Decision is divided into two equal parts.

1- Entropy(Decision|Wind=Strong) = $-$ p(No) . $\log_2$p(No) $-$ p(Yes) . $\log_2$p(Yes)

2- Entropy(Decision|Wind=Strong) = $-$ (3/6) . $\log_2$(3/6) $-$ (3/6) . $\log_2$(3/6) = 1

Now, we can turn back to Gain(Decision, Wind) equation.

Gain(Decision, Wind) = Entropy(Decision) $-$ [p(Decision|Wind=Weak) . Entropy(Decision|Wind=Weak)] $-$ [p(Decision|Wind=Strong) . Entropy(Decision|Wind=Strong)] = 0.940 $-$ [(8/14) . 0.811] $-$ [(6/14). 1] = 0.048

Decision Tree

Other factors on decision

We have applied similar calculation on the other columns.

1- Gain(Decision, Outlook) = 0.246

2- Gain(Decision, Temperature) = 0.029

3- Gain(Decision, Humidity) = 0.151

As seen, outlook factor on decision produces the highest score. That's why, outlook decision will appear in the root node of the tree.

Decision Tree

Overcast outlook on decision

Basically, decision will always be yes if outlook were overcast.

Day	Outlook	Temp.	Humidity	Wind	Decision
3	Overcast	Hot	High	Weak	Yes
7	Overcast	Cool	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes

Decision Tree

Sunny outlook on decision

Day	Outlook	Temp.	Humidity	Wind	Decision
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes

Here, there are 5 instances for sunny outlook. Decision would be probably 3/5 percent no, 2/5 percent yes.

1- Gain(Outlook=Sunny|Temperature) = 0.570

2- Gain(Outlook=Sunny|Humidity) = 0.970

3- Gain(Outlook=Sunny|Wind) = 0.019

Now, humidity is the decision because it produces the highest score if outlook were sunny.

Decision Tree

At this point, decision will always be no if humidity were high.

Day	Outlook	Temp.	Humidity	Wind	Decision
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
8	Sunny	Mild	High	Weak	No

On the other hand, decision will always be yes if humidity were normal

Day	Outlook	Temp.	Humidity	Wind	Decision
9	Sunny	Cool	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes

Finally, it means that we need to check the humidity and decide if outlook were sunny.

Decision Tree

Rain outlook on decision

Day	Outlook	Temp.	Humidity	Wind	Decision
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
10	Rain	Mild	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

1- Gain(Outlook=Rain | Temperature)

2- Gain(Outlook=Rain | Humidity)

3- Gain(Outlook=Rain | Wind)

Here, wind produces the highest score if outlook were rain. That's why, we need to check wind attribute in 2nd level if outlook were rain.

Decision Tree

So, it is revealed that decision will always be yes if wind were weak and outlook were rain.

Day	Outlook	Temp.	Humidity	Wind	Decision
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes

What's more, decision will be always no if wind were strong and outlook were rain.

Day	Outlook	Temp.	Humidity	Wind	Decision
6	Rain	Cool	Normal	Strong	No
14	Rain	Mild	High	Strong	No

So, decision tree construction is over. We can use the following rules for decisioning.

Decision Tree

Final version of decision tree