Machine Learning
DSE 2254

Rohini R. Rao & Manjunath Hegde
Department of Data Science and Computer Applications
MIT, Manipal
March 2022
Slide - Set 4 – Bayesian Learning , Decision Trees

Probability Basics

- Probability
  - is the study of randomness and uncertainty
  - Framework for representing uncertainty
- Random Experiment
  - A process whose outcome is uncertain
  - Examples
    - Toss a coin, outcome can be H or T
    - Toss 10 coins , outcome is how many Heads
    - Toss a coin repeatedly until we get Head
- Sample Space
  - denoted by Omega
  - Set of all possible outcomes
- Event
  - Subset of sample space
  - Example – Tossing 1 dice and outcome is even.

Probability Function

- Properties of Probability
  - Prob(A) or Pr(A) or P(A)
  - Probability is associated with an event A
  - $P(A^c) = 1 – P(A)$
- If there are 2 events A, B
  - P(AUB) = P(A) + P(B) if A and B are disjoint
  - $P(AUB) = P(A \cup B) = P(A) + P(B) - P(A \cap B)$ if A & B are not disjoint

Axioms of Probability

- The probability of an event is a non-negative real number:
  - $0 <= P(A) <= 1$
- that the probability that at least one of the elementary events in the entire sample space will occur is 1
  - $P(\Omega) = 1$
  - $P(\phi) = 0$
- Any countable sequence of disjoint sets (synonymous with mutually exclusive events) E_1, E_2, E_3 …. $E_n$satisfies

  $P(E_1 \cup E_2 \cup \ldots \cup E_n) = P(E_1) + P(E_2) + \ldots + P(E_n)$

Random Variables

- Random Variable is a function defined on sample space Ω
- Maps outcome of an random event into real scalar values.
- For example
  - Experiment We roll 2 dice and
  - Random variable is the sum of values
- Random variable may be discrete or continuous
  - If discrete use PMF
  - If continuous use PDF

Probability Mass Function

- **probability mass function** (**PMF**)
  - Distribution of discrete random variable
  - is a function that gives the probability that a discrete random variable is exactly equal to some value

- Standard distributions are
  - Uniform Distribution X can be 1,2,3,…N and P(X=i) = 1/N
  - Binomial Distributions

$$p_X(k) = P(X = k) = \begin{cases} \binom{n}{k} p^k q^{n-k} & \text{if } k = 0, 1, \ldots, n; \\ 0 & \text{otherwise.} \end{cases}$$

Probability Density Function

- If $f(x)$ is a **probability density function** for a continuous random variable X then

$$1)\ F(b) = \Pr(X \leq x) = \int_{-\infty}^{x} f(t)dt$$

$$2)\ f(x) \geq 0 \text{ for any value of } x$$

$$3) \int_{-\infty}^{\infty} f(t)dt = 1$$

- Normal Distribution
- A normal distribution in a variate X with mean μ and variance σ^2 is a statistic distribution with probability density function

$$f_X(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right], \quad -\infty < x < \infty.$$

Probability

- Joint Probability

$$p(A, B) = p(A \wedge B) = p(A|B)p(B)$$

- Marginal Distribution

$$p(A) = \sum_b p(A, B) = \sum_b p(A|B = b)p(B = b)$$

- Chain Rule of Probability

$$p(X_{1:D}) = p(X_1)p(X_2|X_1)p(X_3|X_2, X_1)p(X_4|X_1, X_2, X_3)\ldots p(X_D|X_{1:D-1})$$

- Conditional Probability
  - of event *A*, given that event *B* is true, as follows:

$$p(A|B) = \frac{p(A, B)}{p(B)} \text{ if } p(B) > 0$$

Maximum A Posteriori (MAP) Hypothesis

Deals with how to find the probability of a hypothesis given the data you have different possible competing hypothesis

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

The Goal of Bayesian Learning:
the most probably hypothesis , given the training data is

$$\begin{aligned} h_{MAP} &= \arg\max_{h \in H} P(h|D) \\ &= \arg\max_{h \in H} \frac{P(D|h)P(h)}{P(D)} \\ &= \arg\max_{h \in H} P(D|h)P(h) \end{aligned}$$

Bayes Rule

$$p(X=x|Y=y)=\frac{p(X=x,Y=y)}{p(Y=y)}=\frac{p(X=x)p(Y=y|X=x)}{\sum_{x'}p(X=x')p(Y=y|X=x')}$$

- Example - medical diagnosis problem.
  - Suppose has taken a mammogram test. If the test is positive, what is the probability the patient has cancer? That obviously depends on how reliable the test is. Suppose the test has a sensitivity of 80%. The probability of having breast cancer is 0.004. False positives are 0.1.
  - where $x = 1$ is the event the mammogram is positive,
  - And $y = 1$ is the event of having breast cancer

$$p(y=1|x=1) = \frac{p(x=1|y=1)p(y=1)}{p(x=1|y=1)p(y=1)+p(x=1|y=0)p(y=0)}$$

Naïve Bayesian Classifier

Bayes rule:

$$P(Y = y_k | X_1 \dots X_n) = \frac{P(Y = y_k) P(X_1 \dots X_n | Y = y_k)}{\sum_j P(Y = y_j) P(X_1 \dots X_n | Y = y_j)}$$

Assuming conditional independence among X_i's:

$$P(Y = y_k | X_1 \dots X_n) = \frac{P(Y = y_k) \prod_i P(X_i | Y = y_k)}{\sum_j P(Y = y_j) \prod_i P(X_i | Y = y_j)}$$

So, classification rule for $X^{new} = < X_1, \dots, X_n >$ is:

$$Y^{new} \leftarrow \arg\max_{y_k} P(Y = y_k) \prod_i P(X_i^{new} | Y = y_k)$$

Bayes Optimal Classifier

Question: Given new instance x, what is its most probable classification?

- $h_{MAP}(x)$ is not the most probable classification!

Example: Let P(h1|D) = .4, P(h2|D) = .3, P(h3 |D) =.3

Given new data x, we have h1(x)=+, h2(x) = -, h3(x) = -

What is the most probable classification of x ?

Bayes optimal classification:

$$\arg\max_{v_j \in V} \sum_{h_i \in H} P(v_j \mid h_i) P(h_i \mid D)$$

where *V* is the set of all the values a classification can take and v_j is one possible such classification.

Example:

P(h1| D) =.4,	P(-|h1)=0,	P(+|h1)=1	$\sum_{h_i \in H} P(+ \mid h_i) P(h_i \mid D) = .4$
P(h2|D) =.3,	P(-|h2)=1,	P(+|h2)=0	
P(h3|D)=.3,	P(-|h3)=1,	P(+|h3)=0	$\sum_{h_i \in H} P(- \mid h_i) P(h_i \mid D) = .6$

Play Tennis Example

Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

X= {rain,hot,high,false}

Learning Phase

Learning Phase

Outlook	Play=Yes	Play=No
Sunny	2/9	3/5
Overcast	4/9	0/5
Rain	3/9	2/5

Temperature	Play=Yes	Play=No
Hot	2/9	2/5
Mild	4/9	2/5
Cool	3/9	1/5

Humidity	Play=Yes	Play=No
High	3/9	4/5
Normal	6/9	1/5

Wind	Play=Yes	Play=No
Strong	3/9	3/5
Weak	6/9	2/5

$P(\text{Play=Yes}) = 9/14$ $P(\text{Play=No}) = 5/14$

Test Phase

- Given a new instance, predict its label

  X'=(Outlook=Sunny, Temperature=Cool, Humidity=High, Wind=Strong)

- Look up tables achieved in the learning phrase

  P(Outlook=Sunny|Play=Yes) = 2/9
  P(Temperature=Cool|Play=Yes) = 3/9
  P(Huminity=High|Play=Yes) = 3/9
  P(Wind=Strong|Play=Yes) = 3/9
  P(Play=Yes) = 9/14

  P(Outlook=Sunny|Play=No) = 3/5
  P(Temperature=Cool|Play=No) = 1/5
  P(Huminity=High|Play=No) = 4/5
  P(Wind=Strong|Play=No) = 3/5
  P(Play=No) = 5/14

- Decision making with the MAP rule

$P(Yes|x') = [P(Sunny|Yes)P(Cool|Yes)P(High|Yes)P(Strong|Yes)]P(Play=Yes) = 0.0053$

$P(No|x') = [P(Sunny|No)P(Cool|No)P(High|No)P(Strong|No)]P(Play=No) = 0.0206$

Given the fact $P(Yes|x') < P(No|x')$, we label x' to be "No".

Gaussian Naïve Bayes

- **Algorithm: Continuous-valued Features**
  - **Conditional probability often modeled with the normal distribution**

$$P(X_i = x \mid Y = y_k) = \frac{1}{\sigma_{ik}\sqrt{2\pi}} \; e^{\frac{-(x-\mu_{ik})^2}{2\sigma_{ik}^2}}$$

Sometimes assume variance

- **is independent of Y (i.e., σ_i),**
- **or independent of X_i (i.e., σ_k)**
- **or both (i.e., σ)**

Naïve Bayes - Continuous valued features

- Temperature is naturally of continuous value.

  Yes: 25.2, 19.3, 18.5, 21.7, 20.1, 24.3, 22.8, 23.1, 19.8

  No: 27.3, 30.1, 17.4, 29.5, 15.1

- Estimate mean and variance for each class

$$\mu = \frac{1}{N}\sum_{n=1}^{N} x_n, \quad \sigma^2 = \frac{1}{N}\sum_{n=1}^{N}(x_n - \mu)^2$$

$$\mu_{Yes} = 21.64, \ \sigma_{Yes} = 2.35$$
$$\mu_{No} = 23.88, \ \sigma_{No} = 7.09$$

- **Learning Phase**: output two Gaussian models for P(temp|C)

$$\hat{P}(x|Yes) = \frac{1}{2.35\sqrt{2\pi}} \exp\left(-\frac{(x-21.64)^2}{2\times 2.35^2}\right) = \frac{1}{2.35\sqrt{2\pi}} \exp\left(-\frac{(x-21.64)^2}{11.09}\right)$$

$$\hat{P}(x|No) = \frac{1}{7.09\sqrt{2\pi}} \exp\left(-\frac{(x-23.88)^2}{2\times 7.09^2}\right) = \frac{1}{7.09\sqrt{2\pi}} \exp\left(-\frac{(x-23.88)^2}{50.25}\right)$$

15

Example – Estimation of probability from data

Tid	Refund	Marital Status	Taxable Income	Evade
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

- Normal distribution:

$$P(X_i \mid Y_j) = \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} e^{\frac{(X_i-\mu_{ij})^2}{2\sigma_{ij}^2}}$$

  - One for each (X_i, Y_i) pair
- For (Income, Class=No):
  - If Class=No
    - sample mean = 110
    - sample variance = 2975

Example – Estimation of probability from data

Tid	Refund	Marital Status	Taxable Income	Evade
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

Given a Test Record:

$$X = (\text{Refund} = \text{No}, \text{Divorced}, \text{Income} = 120\text{K})$$

For Taxable Income:
If class = No: sample mean = 110
sample variance = 2975
If class = Yes: sample mean = 90
sample variance = 25

- P(X | No) = P(Refund=No | No)
  × P(Divorced | No)
  × P(Income=120K | No)
  = 4/7 × 1/7 × 0.0072 = 0.0006

- P(X | Yes) = P(Refund=No | Yes)
  × P(Divorced | Yes)
  × P(Income=120K | Yes)
  = 1 × 1/3 × 1.2 × 10^{-9} = 4 × 10^{-10}

Zero Probability Problem

Consider the table with Tid = 7 deleted

Tid	Refund	Marital Status	Taxable Income	Evade
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
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Naïve Bayes Classifier:

P(Refund = Yes | No) = 2/6
P(Refund = No | No) = 4/6
P(Refund = Yes | Yes) = 0
P(Refund = No | Yes) = 1
P(Marital Status = Single | No) = 2/6
P(Marital Status = Divorced | No) = 0
P(Marital Status = Married | No) = 4/6
P(Marital Status = Single | Yes) = 2/3
P(Marital Status = Divorced | Yes) = 1/3
P(Marital Status = Married | Yes) = 0/3
For Taxable Income:
If class = No: sample mean = 91
sample variance = 685
If class = No: sample mean = 90
sample variance = 25

Given X = (Refund = Yes, Divorced, 120K)

$P(X \mid No) = 2/6 \times 0 \times 0.0083 = 0$

$P(X \mid Yes) = 0 \times 1/3 \times 1.2 \times 10^{-9} = 0$

Naïve Bayes will not be able to classify X as Yes or No!

The Zero Probability Problem

- If one of the conditional probabilities is zero, then the entire expression becomes zero
- Need to use other estimates of conditional probabilities than simple fractions
- Probability estimation:

$$\text{Original}: P(A_i \mid C) = \frac{N_{ic}}{N_c}$$

$$\text{Laplace}: P(A_i \mid C) = \frac{N_{ic} + 1}{N_c + c}$$

c: number of classes

p: prior probability of the class

m: parameter

N_c: number of instances in the class

Zero Probability Problem

Pros and Cons of Bayesian Classifier

- Robust to isolated noise points
- Handle missing values by ignoring the instance during probability estimate calculations
- Robust to irrelevant attributes
- Independence assumption may not hold for some attributes
  - Use other techniques such as Bayesian Belief Networks (BBN)

Bayesian Network

- Structure of the graph ⇔ Conditional independence relations

In general,

$$p(X_1, X_2, \ldots X_N) = \Pi\, p(X_i \mid \text{parents}(X_i))$$

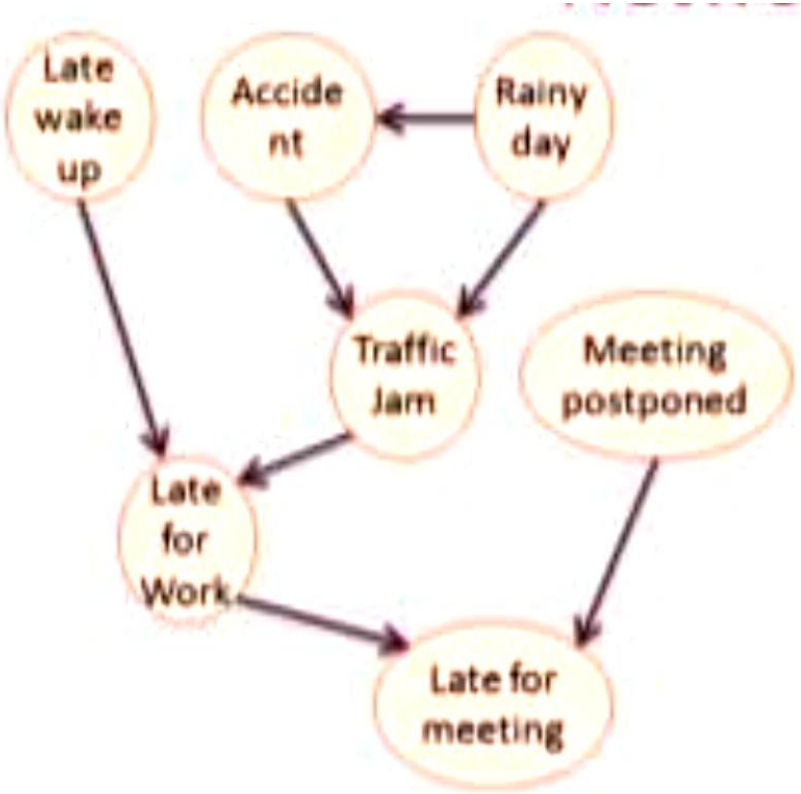
The full joint distribution

The graph-structured approximation

- Requires that graph is acyclic (no directed cycles)

- 2 components to a Bayesian network
  - The graph structure (conditional independence assumptions)
  - The numerical probabilities (for each variable given its parents)

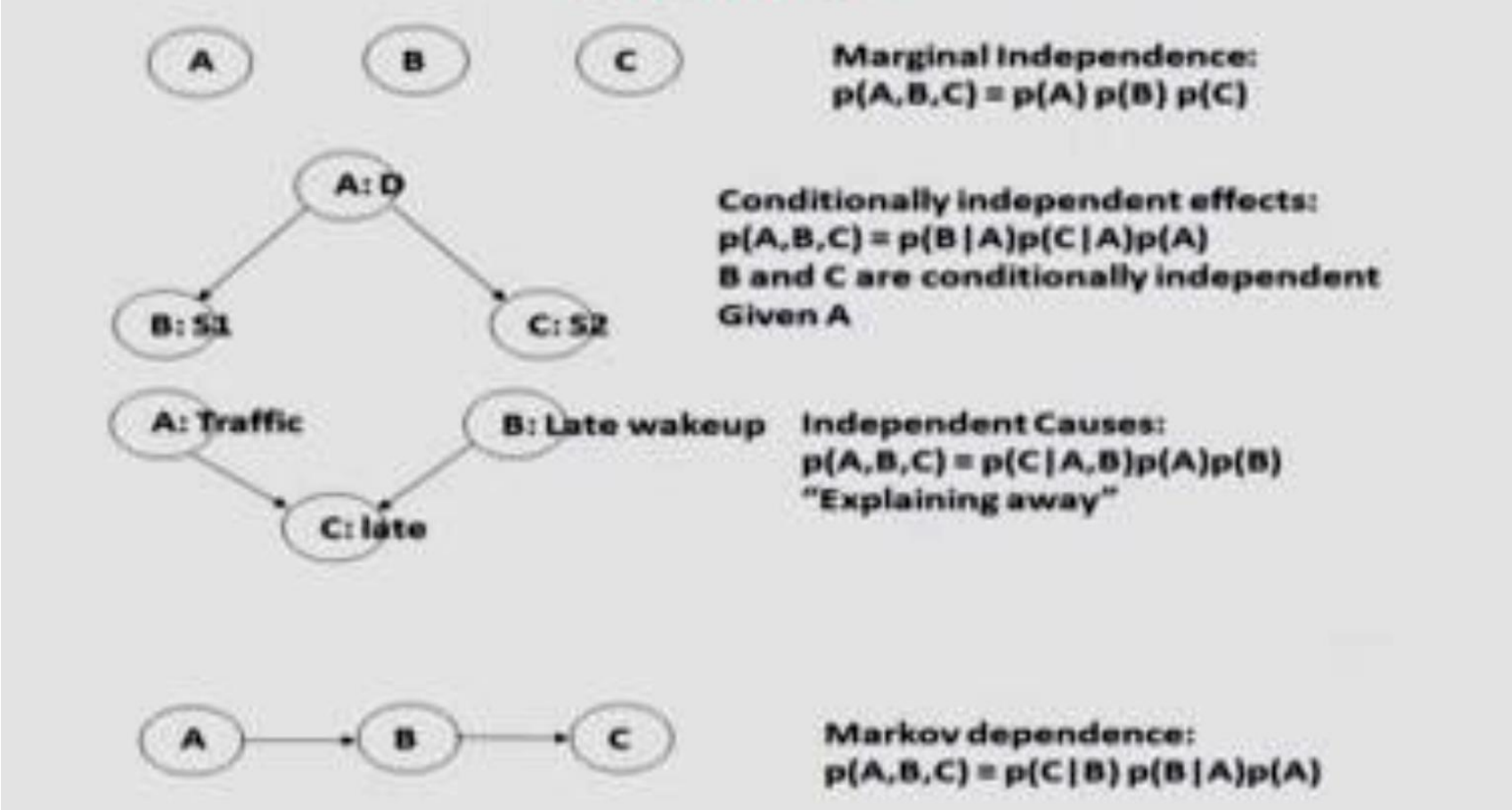
Representation of Bayesian Network



Conditional probability table associated with each node specifies the conditional distribution for the variable given its immediate parents in the graph

Each node is asserted to be conditionally independent of its non-descendants, given its immediate parents

Examples

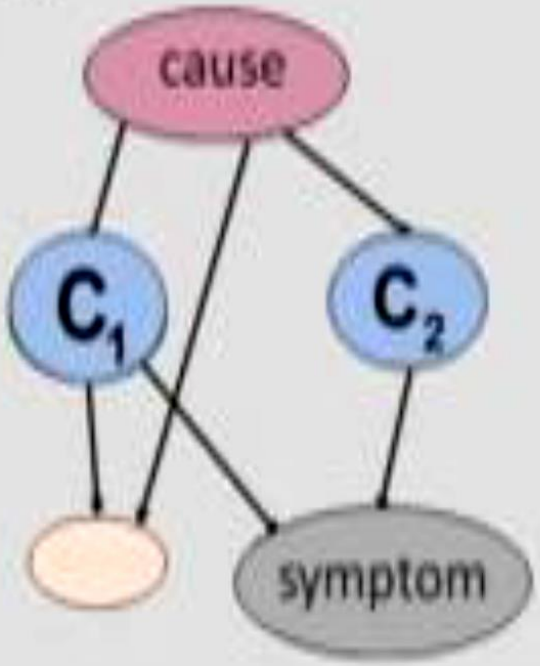

A
B
C
Marginal Independence:
p(A,B,C) = p(A) p(B) p(C)
A: D
B: S1
C: S2
Conditionally independent effects:
p(A,B,C) = p(B|A)p(C|A)p(A)
B and C are conditionally independent
Given A
A: Traffic
B: Late wakeup
C: late
Independent Causes:
p(A,B,C) = p(C|A,B)p(A)p(B)
"Explaining away"
A
B
C
Markov dependence:
p(A,B,C) = p(C|B) p(B|A)p(A)

Learning Bayesian Belief Networks

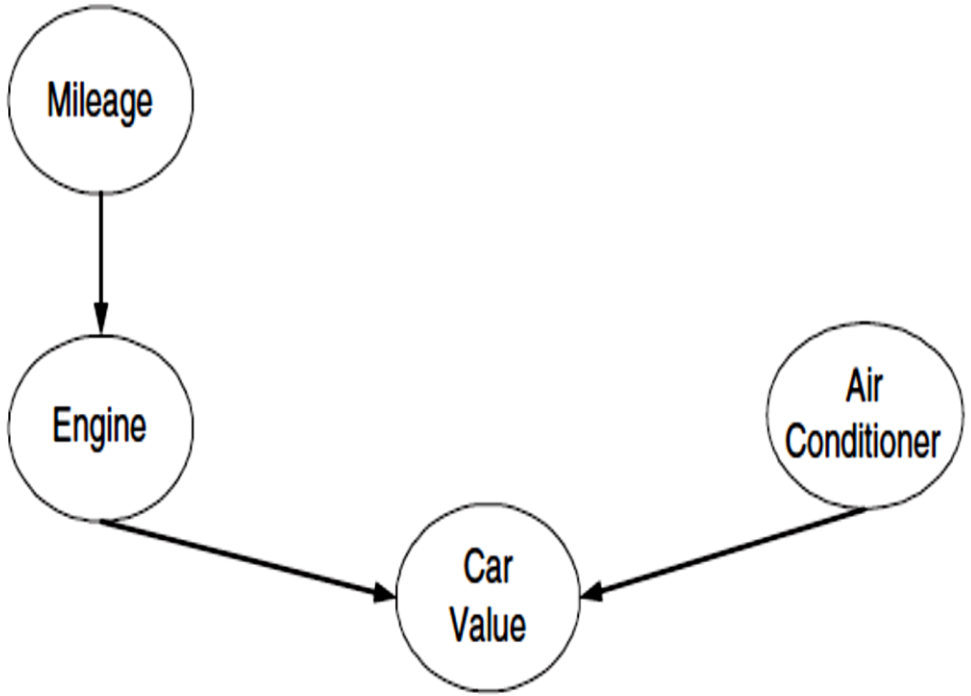
1. The network structure is given in advance and all the variables are fully observable in the training examples.
   - estimate the conditional probabilities.
2. The network structure is given in advance but only some of the variables are observable in the training data.
   - Similar to learning the weights for the hidden units of a Neural Net: Gradient Ascent Procedure
3. The network structure is not known in advance.
   - Use a heuristic search or constraint-based technique to search through potential structures.

Applications of Bayesian Network

- Diagnosis: P(cause|symptom)=?
- Prediction: P(symptom|cause)=?
- Classification: P(class|data)
- Decision-making
  (given a cost function)



Example



Mileage	Engine	Air Conditioner	Number of Records with Car Value=Hi	Number of Records with Car Value=Lo
Hi	Good	Working	3	4
Hi	Good	Broken	1	2
Hi	Bad	Working	1	5
Hi	Bad	Broken	0	4
Lo	Good	Working	9	0
Lo	Good	Broken	5	1
Lo	Bad	Working	1	2
Lo	Bad	Broken	0	2

Draw the probability table for each node in the network.

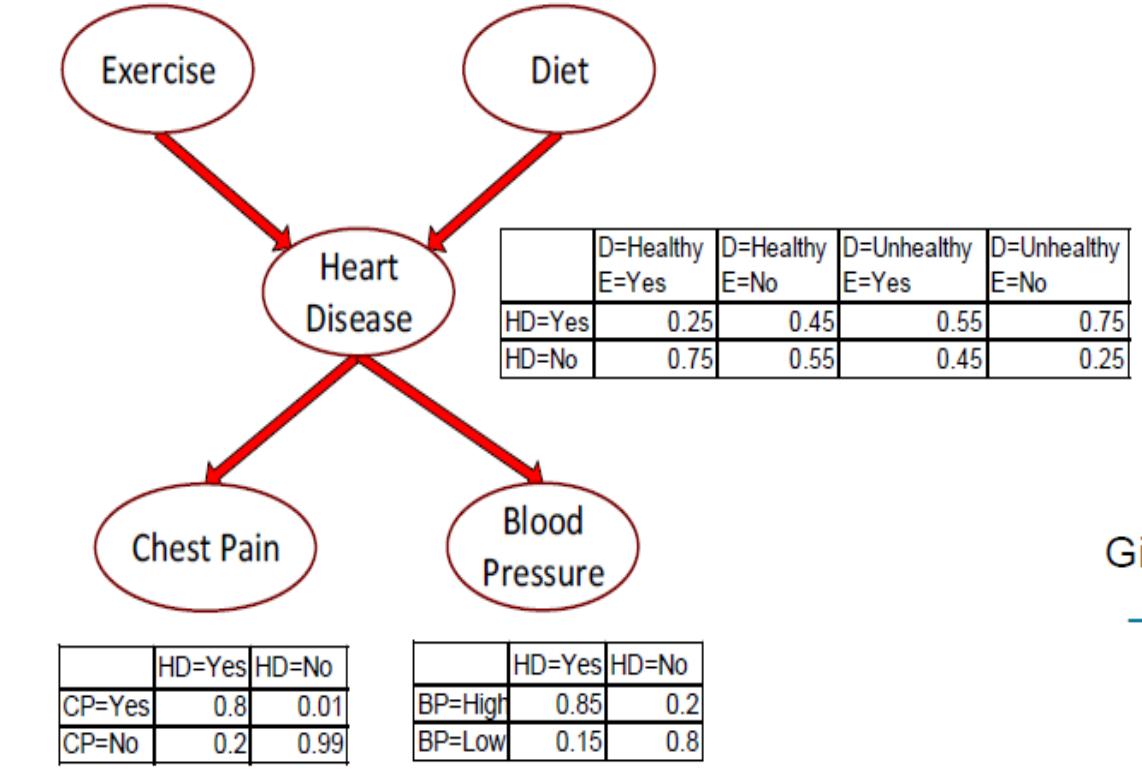
P(Mileage=Hi) = 0.5

P(Air Cond=Working) = 0.625

P(Engine=Good|Mileage=Hi) = 0.5

P(Engine=Good|Mileage=Lo) = 0.75

Example



Exercise=Yes	0.7
Exercise=No	0.3

Diet=Healthy	0.25
Diet=Unhealthy	0.75

	D=Healthy E=Yes	D=Healthy E=No	D=Unhealthy E=Yes	D=Unhealthy E=No
HD=Yes	0.25	0.45	0.55	0.75
HD=No	0.75	0.55	0.45	0.25

	HD=Yes	HD=No
CP=Yes	0.8	0.01
CP=No	0.2	0.99

	HD=Yes	HD=No
BP=High	0.85	0.2
BP=Low	0.15	0.8

Given: X = (E=No, D=Yes, CP=Yes, BP=High)

- Compute P(HD|E,D,CP,BP)?

Example

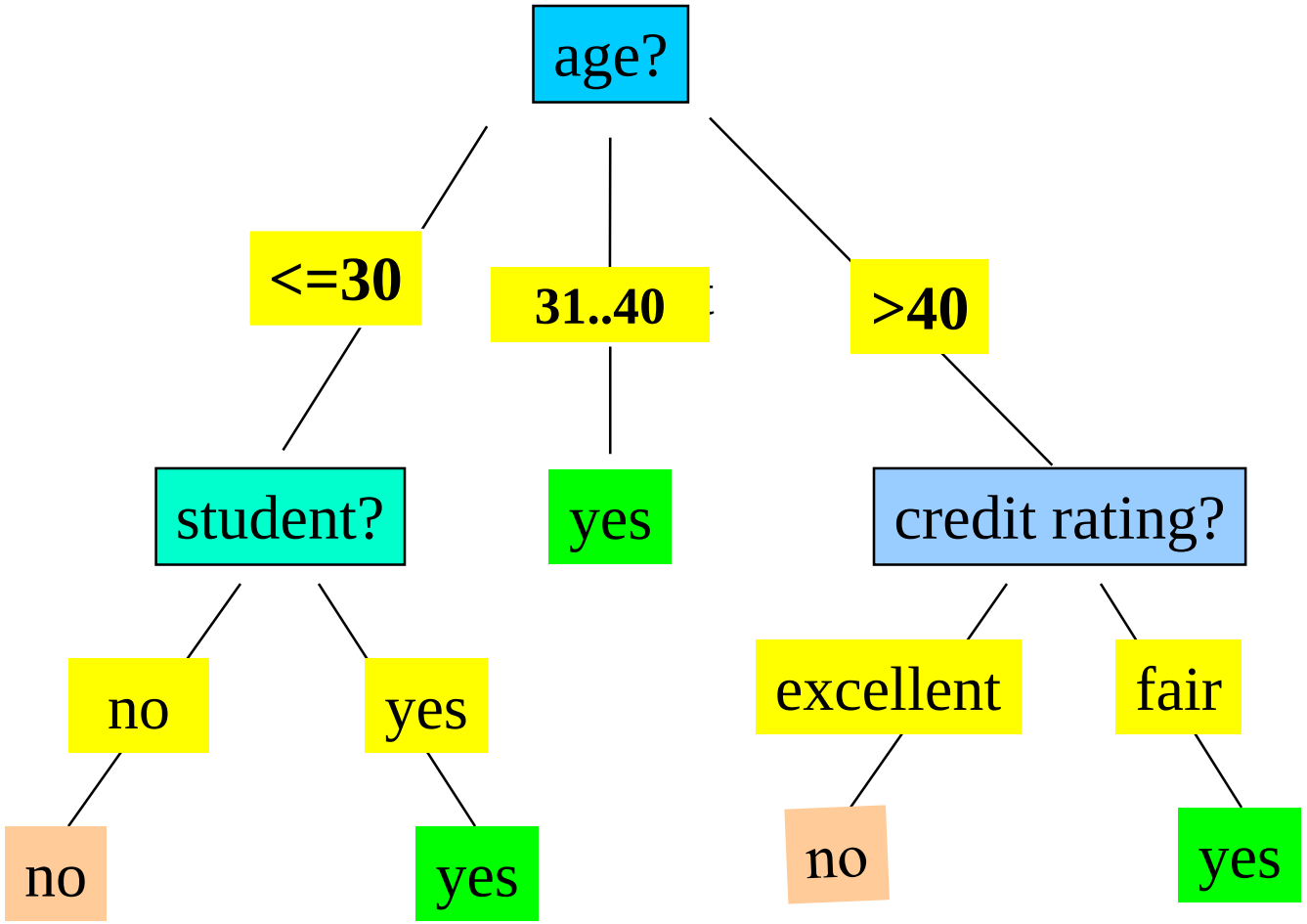
- P(HD=Yes| E=No,D=Yes) = 0.55
  P(CP=Yes| HD=Yes) = 0.8
  P(BP=High| HD=Yes) = 0.85
  - P(HD=Yes|E=No,D=Yes,CP=Yes,BP=High) $\propto 0.55 \times 0.8 \times 0.85 = 0.374$
- P(HD=No| E=No,D=Yes) = 0.45
  P(CP=Yes| HD=No) = 0.01
  P(BP=High| HD=No) = 0.2
  - P(HD=No|E=No,D=Yes,CP=Yes,BP=High) $\propto 0.45 \times 0.01 \times 0.2 = 0.0009$

Classify X as Yes

Decision Tree Induction

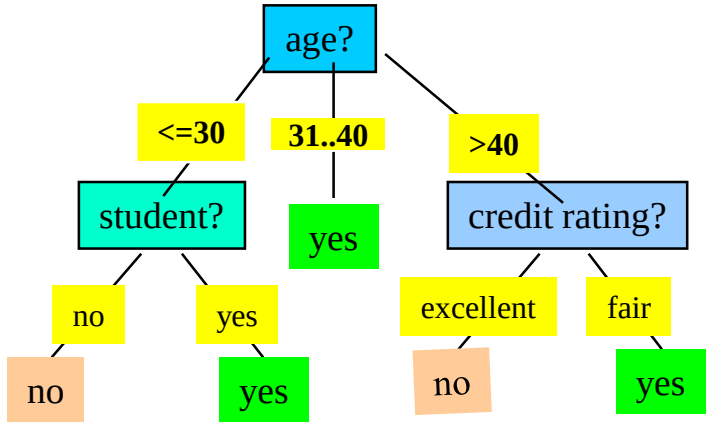
- In 70s and early 80s Ross Quinlan , a researcher in machine learning developed ID3 (Iterative Dichotomiser)
  - uses entropy as measure of how informative the node is.
- C4.5 which is a benchmark to which new supervised learning algorithms are often compared.
  - Extension of ID3. accounts for unavailable values, continuous attributes , pruning of decision trees and rule derivation
  - Does not generate a binary tree
- Classification and Regression Trees (CART)
  - uses gini index for determining best split
  - Build binary decision tree
  - Adopt a greedy ie. Non back tracking approach in which decision trees are constructed in a top down recursive divide and conquer manner.
- Training set s recursively partitioned into smaller sub sets as the tree is being built.

Output: A Decision Tree for "*buys_computer*"



Rule Extraction from a Decision Tree

- Rules are easier to understand than large trees
- 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 holds the class prediction
- Rules are mutually exclusive and exhaustive



- Example: Rule extraction from our *buys_computer* decision-tree

  IF *age* = young AND *student* = *no* THEN *buys_computer* = *no*

  IF *age* = young AND *student* = *yes* THEN *buys_computer* = *yes*

  IF *age* = mid-age THEN *buys_computer* = *yes*

  IF *age* = old AND *credit_rating* = *excellent* THEN *buys_computer* = *no*

  IF *age* = old AND *credit_rating* = *fair* THEN *buys_computer* = *yes*

Using IF-THEN Rules for Classification

- Represent the knowledge in the form of IF-THEN rules

  R: IF *age* = youth AND *student* = yes THEN *buys_computer* = yes

  - Rule antecedent/precondition vs. rule consequent

- Assessment of a rule: *coverage* and *accuracy*
  - n_{covers} = # of tuples covered by R
  - $n_{correct}$ = # of tuples correctly classified by R

  coverage(R) = n_{covers} /|D| /* D: training data set */

  accuracy(R) = $n_{correct}$ / n_{covers}

- If more than one rule is triggered, need **conflict resolution**
  - Size ordering: assign the highest priority to the triggering rules that has the "toughest" requirement (i.e., with the *most attribute test*)
  - Class-based ordering: decreasing order of *prevalence or misclassification cost per class*
  - Rule-based ordering (**decision list**): rules are organized into one long priority list, according to some measure of rule quality or by experts

Attribute Selection Measure

- Is a heuristic for selecting the splitting criterion that best separates a given data partition D
- Ideally each partition has to be pure.
- Information Gain
  - Used in ID3 and based on Claude Shannon's work on information theory, which studied the value or information content of messages.
  - Attribute with highest information gain is chosen as splitting attribute for node N
  - Minimizes the expected number of tests needed to classify a given tuple
  - A log function to the base 2 is used because information is encoded in bits

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:

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

- Information gained by branching on attribute A

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

Decision Tree Induction: Training Dataset

This follows an example of 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

Attribute Selection: Information Gain

- Class P: buys_computer = "yes"
- Class N: buys_computer = "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_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

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

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

Similarly,

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

$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

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

Computing Information-Gain for Continuous-Value 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

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
  - **Gini index:**
    - biased to multivalued attributes
    - has difficulty when # of classes is large
    - tends to favor tests that result in equal-sized partitions and purity in both partitions

Algorithm for Decision Tree Induction

- Tree is constructed in a top-down recursive divide-and-conquer manner
- At start, all the training examples are at the root
- Use attribute selection to pick splitting attribute.
- The splitting criterion specifies splitting attribute and split-point.
- Examples are partitioned recursively based on selected attributes
- A partition is pure if all the tuples belong to the same class
- Splitting variable A can be
  - Discrete , then the outcomes of the test are the known values of A
  - Continuous-valued, then the outcomes are <= split point and > split_point
  - Binary , then yes and no outcomes and binary tree has to be produced

- 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 tuples left , ie. Partition is empty

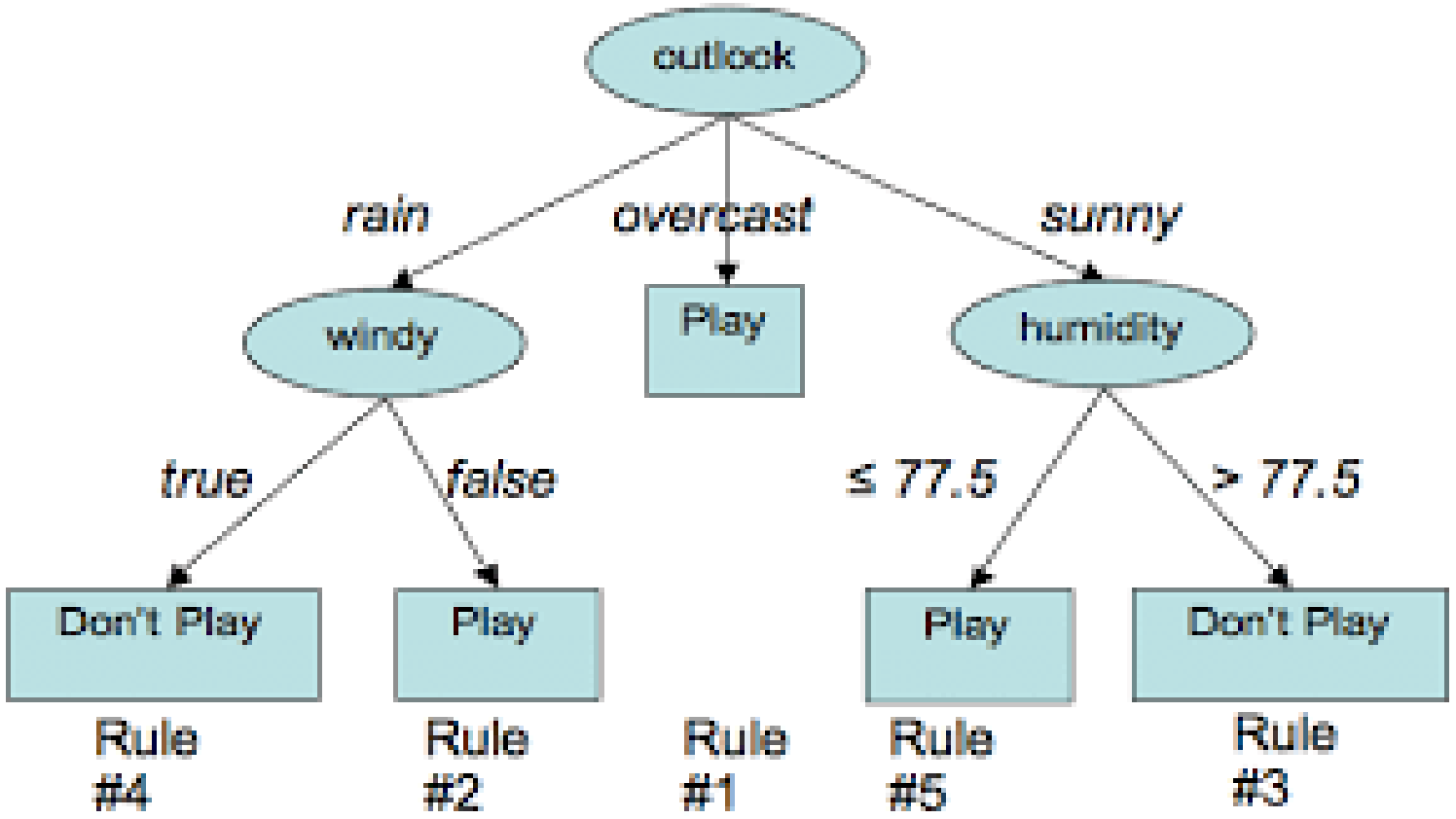
Play Golf - Training Data Set

Outlook	Temp	Humidity	Windy	Class
Sunny	79	90	True	No play
Sunny	56	70	False	Play
Sunny	79	75	True	Play
Sunny	60	90	True	No play
Overcast	88	88	False	No play
Overcast	63	75	True	Play
Overcast	88	95	False	Play
Rain	78	60	False	Play
Rain	66	70	False	No play
Rain	68	60	True	No Play

Play Golf- Test Data Set

Outlook	Temp	Humidity	Windy	Class
Sunny	79	90	True	Play
Sunny	56	70	False	Play
Sunny	79	75	True	No Play
Sunny	60	90	True	No play
Overcast	88	88	False	No play
Overcast	63	75	True	Play
Overcast	88	95	False	Play
Rain	78	60	False	Play
Rain	66	70	False	No play
Rain	68	60	True	Play

What is accuracy of model, metrics for rules?



Pros and Cons of Decision Tree Classification

- Advantages
  - Construction of Decision tree does not require any domain knowledge or parameter setting and so is appropriate for exploratory knowledge discovery
  - Can handle high dimensional data
  - are able to generate understandable rules
  - able to handle both numerical and categorical attributes
  - indicate which attributes are important for prediction or classification

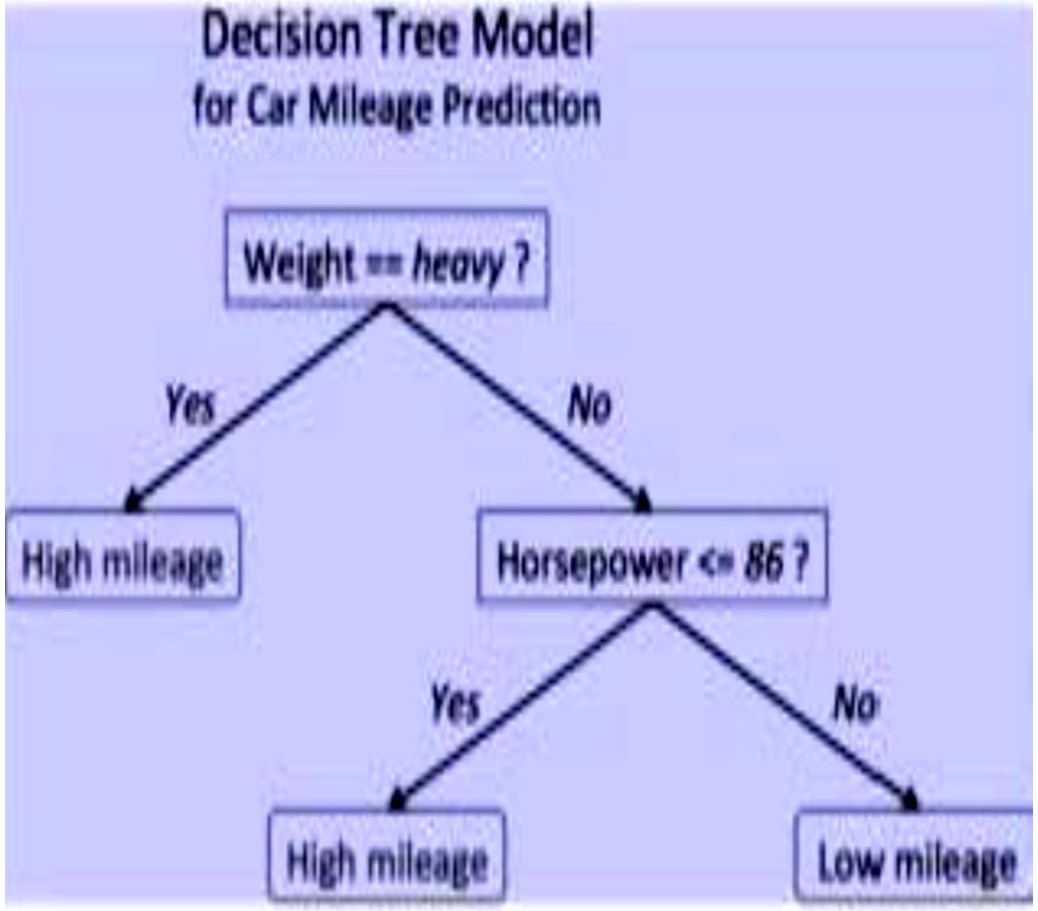
- Disadvantages
  - Error prone where training class is small
  - Can be computationally expensive.

Example 2

- Consider the following data set for a binary class problem. Calculate the information gain when splitting on attribute A and on attribute B. Which attribute would be selected for the root of the decision tree?

A	B	Class Label
T	F	+
T	T	+
T	T	+
T	F	--
T	T	+
F	F	--
F	F	--
F	F	--
T	T	--
T	F	--

Decision Tree



- Given Training examples, it is possible to build many decision trees to fit data
- If examples are noisy, there could be no decision tree which exactly fit the data.
- Bias to restrict the hypothesis could be :
  - Prefer smaller trees- smaller number of nodes
  - Trees with smaller depth
- Could be computationally hard problem
- So go for greedy algorithms, that search for good tree

Hypothesis Space Search in Decision Trees

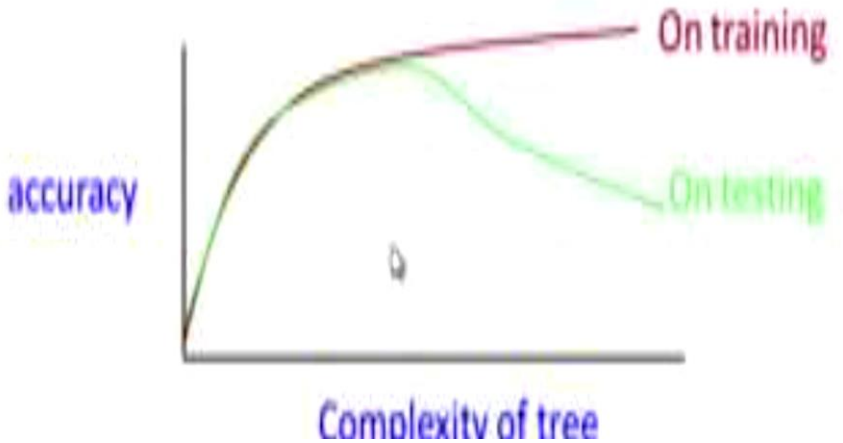
- Conduct a search of the space of decision trees which can represent all possible discrete functions.
- Goal: to find the best decision tree
- Finding a minimal decision tree consistent with a set of data is NP-hard.
- Perform a greedy heuristic search: hill climbing without backtracking
- Statistics-based decisions using all data

Practical Issues of Classification

- Underfitting and Overfitting
- Missing Values
- Costs of Classification

Underfitting & Overfitting

- Learning a tree that classifies the training data perfectly may not lead to the tree with the best generalization performance.
  - There may be noise in the training data
  - May be based on insufficient data
- A hypothesis *h* is said to overfit the training data if there is another hypothesis, h', such that *h* has smaller error than *h'* on the training data but *h* has larger error on the test data than *h'*.



Avoid Overfitting

- How can we avoid overfitting a decision tree?
  - Prepruning: Stop growing when data split not statistically significant
  - Postpruning: Grow full tree then remove nodes

- Methods for evaluating subtrees to prune:
  - Minimum description length (MDL):

Minimize: size(tree) + size(misclassifications(tree))

MDL –
principle describes a way to minimize models.
For instance, combining the length (or in our case, the tree depth) and the cost into a new and improved cost function.

Pre vs. Post pruning

Pre-Pruning (Early Stopping)

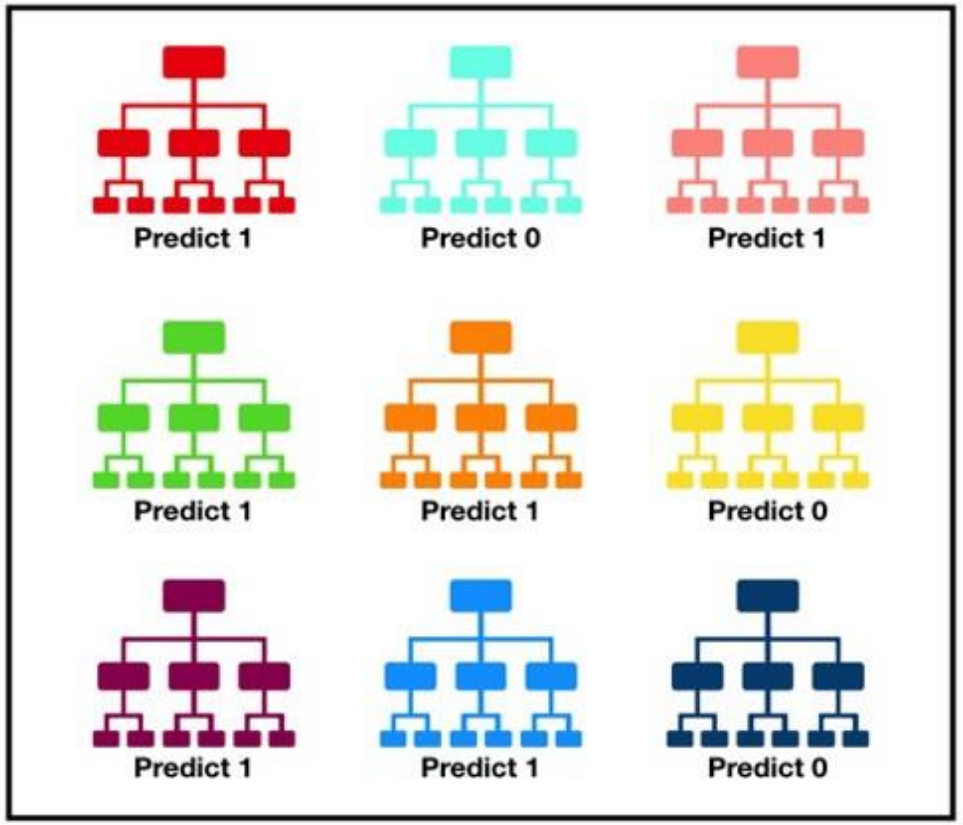
- Typical stopping conditions for a node:
  - Stop if all instances belong to the same class
  - Stop if all the attribute values are the same
- More restrictive conditions:
  - Stop if number of instances is less than some user-specified threshold
  - Stop if class distribution of instances are independent of the available features (e.g., using χ^2 test)
  - Stop if expanding the current node does not improve impurity measures (e.g., Gini or information gain).

Reduced-error Pruning

- A post-pruning, cross validation approach
  - Partition training data into "grow" set and "validation" set.
  - Build a complete tree for the "grow" data
  - Until accuracy on validation set decreases, do:
    For each non-leaf node in the tree
    Temporarily prune the tree below; replace it by majority vote
    Test the accuracy of the hypothesis on the validation set
    Permanently prune the node with the greatest increase in accuracy on the validation test.
- Problem: Uses less data to construct the tree
- Sometimes done at the rules level

General Strategy: Overfit and Simplify

Random forest



- A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent model
- Consists of a large number of individual decision trees that operate as an ensemble.
- Each individual tree in the random forest spits out a class prediction
- the class with the most votes becomes our model's prediction

Method of ensuring diversity in trees

- **For random forest to make accurate class predictions**
  - We need features that have at least some predictive power.
  - The trees of the forest and more importantly their predictions need to be uncorrelated (or at least have low correlations with each other).
- **Bagging**
  - allowing each individual tree to randomly sample from the dataset with replacement, resulting in different trees.
- **Feature Randomness**
  - each tree in a random forest can pick only from a random subset of features.