CS6220: DATA MINING TECHNIQUES

Chapter 8&9: Classification: Part 3

Instructor: Yizhou Sun

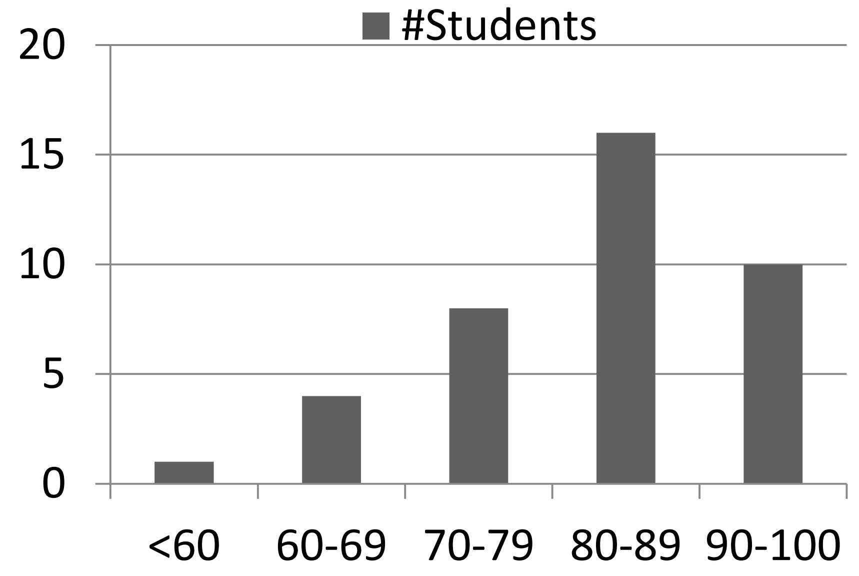
yzsun@ccs.neu.edu

March 12, 2013

Midterm Report

Grade Distribution

90 - 100	10
80 - 89	16
70 - 79	8
60 - 69	4
<60	1



Statistics

Count	39
Minimum Value	55.00
Maximum Value	98.00
Average	82.54
Median	84.00
Standard Deviation	9.18

Announcement

- Midterm Solution
  - https://blackboard.neu.edu/bbcswebdav/pid-12532-dt-wiki-rid-8320466_1/courses/CS6220.32435.201330/mid_term.pdf

- Course Project:
  - Midterm report due next week
    - A draft for final report
      - Don’t forget your project title
    - Main purpose
      - Check the progress and make sure you can finish it by the deadline

Chapter 8&9. Classification: Part 3

- Bayesian Learning
  - Naïve Bayes
  - Bayesian Belief Network
- Instance-Based Learning
- Summary

Bayesian Classification: Why?

- A statistical classifier: performs *probabilistic prediction, i.e.,* predicts class membership probabilities
- Foundation: Based on Bayes' Theorem.
- Performance: A simple Bayesian classifier, *naïve Bayesian classifier*, has comparable performance with decision tree and selected neural network classifiers
- Incremental: Each training example can incrementally increase/decrease the probability that a hypothesis is correct — prior knowledge can be combined with observed data
- Standard: Even when Bayesian methods are computationally intractable, they can provide a standard of optimal decision making against which other methods can be measured

Basic Probability Review

- Have two dices h_1 and h_2
- The probability of rolling an *i* given die h_1 is denoted $P(i|h_1)$. This is a ***conditional probability***
- Pick a die at random with probability $P(h_j)$, j=1 or 2. The probability for picking die h_j and rolling an i with it is called ***joint probability*** and is $P(i, h_j)=P(h_j)P(i|\ h_j)$.
- For any events X and Y, $P(X,Y)=P(X|Y)P(Y)$
- If we know $P(X,Y)$, then the so-called ***marginal probability*** $P(X)$ can be computed as $P(X)=\sum_{Y} P(X,Y)$

Bayes' Theorem: Basics

- Bayes' Theorem: $P(h|\mathbf{X}) = \frac{P(\mathbf{X}|h)P(h)}{P(\mathbf{X})}$
  - Let **X** be a data sample ("*evidence*")
  - Let h be a *hypothesis* that X belongs to class C
  - P(h) (*prior probability*): the initial probability
    - E.g., **X** will buy computer, regardless of age, income, …
  - P(**X**|h) (likelihood): the probability of observing the sample **X**, given that the hypothesis holds
    - E.g., Given that **X** will buy computer, the prob. that X is 31..40, medium income
  - P(**X**): marginal probability that sample data is observed
    - $P(X) = \sum_h P(X|h)\, P(h)$
  - P(h|**X**), (i.e., *posteriori probability):* the probability that the hypothesis holds given the observed data sample **X**

Classification: Choosing Hypotheses

- *Maximum Likelihood* (maximize the likelihood):

$$h_{ML} = \arg\max_{h \in H} P(D \mid h)$$

- *Maximum a posteriori* (maximize the posterior):
  - Useful observation: it does not depend on the denominator P(D)

$$h_{MAP} = \arg\max_{h \in H} P(h \mid D) = \arg\max_{h \in H} P(D \mid h) P(h)$$

***D:* the whole training data set**

Classification by Maximum A Posteriori

- Let D be a training set of tuples and their associated class labels, and each tuple is represented by an n-D attribute vector **X** = $(x_1, x_2, \ldots, x_n)$
- Suppose there are *m* classes $C_1, C_2, \ldots, C_m$.
- Classification is to derive the maximum posteriori, i.e., the maximal $P(C_i|\mathbf{X})$
- This can be derived from Bayes' theorem

$$P(C_i|\mathbf{X}) = \frac{P(\mathbf{X}|C_i)P(C_i)}{P(\mathbf{X})}$$

- Since P(X) is constant for all classes, only $P(C_i, \mathbf{X}) = P(\mathbf{X}|C_i)P(C_i)$ needs to be maximized

Example: Cancer Diagnosis

- A patient takes a lab test with two possible results (+ve, -ve), and the result comes back positive. It is known that the test returns
  - a correct positive result in only 98% of the cases (true positive); and
  - a correct negative result in only 97% of the cases (true negative).
  - Furthermore, only 0.008 of the entire population has this disease.

1. What is the probability that this patient has cancer?
2. What is the probability that he does not have cancer?
3. What is the diagnosis?

Solution

$P(cancer) = .008$ $P(\neg cancer) = .992$

$P(+ve|cancer) = .98$ $P(-ve|cancer) = .02$

$P(+ve|\neg cancer) = .03$ $P(-ve|\neg cancer) = .97$

Using Bayes Formula:

$P(cancer|+ve) = P(+ve|cancer) x P(cancer) / P(+ve)$

$= 0.98 \times 0.008 / P(+ve) = .00784 / P(+ve)$

$P(\neg cancer|+ve) = P(+ve|\neg cancer) x P(\neg cancer) / P(+ve)$

$= 0.03 \times 0.992 / P(+ve) = .0298 / P(+ve)$

So, the patient most likely does not have cancer.

Chapter 8&9. Classification: Part 3

- Bayesian Learning
  - Naïve Bayes
  - Bayesian Belief Network
- Instance-Based Learning
- Summary

Naïve Bayes Classifier

- A simplified assumption: attributes are conditionally independent given the class (class conditional independency):

$$P(\mathbf{X}|C_i) = \prod_{k=1}^{n} P(x_k|C_i) = P(x_1|C_i) \times P(x_2|C_i) \times \ldots \times P(x_n|C_i)$$

- This greatly reduces the computation cost: Only counts the class distribution
- $P(C_i) = |C_{i,D}|/|D|$ ($|C_{i,D}|$= # of tuples of C_i in D)
- If A_k is categorical, $P(x_k|C_i)$ is the # of tuples in C_i having value x_k for A_k divided by $|C_{i,D}|$
- If A_k is continuous-valued, $P(x_k|C_i)$ is usually computed based on Gaussian distribution with a mean μ and standard deviation σ

$$g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

and $P(x_k|C_i)$ is

$$P(\mathbf{X}|C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i})$$

Naïve Bayes Classifier: Training Dataset

Class:

C1:buys_computer = 'yes'

C2:buys_computer = 'no'

Data to be classified:

X = (age <=30,

Income = medium,

Student = yes

Credit_rating = Fair)

age	income	student	credit_rating	_comp
<=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

Naïve Bayes Classifier: An Example

age	income	student	credit_rating	_comp
<=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

- $P(C_i)$: P(buys_computer = “yes”) = 9/14 = 0.643
  P(buys_computer = “no”) = 5/14= 0.357
- Compute $P(X|C_i)$ for each class
  P(age = “<=30” | buys_computer = “yes”) = 2/9 = 0.222
  P(age = “<= 30” | buys_computer = “no”) = 3/5 = 0.6
  P(income = “medium” | buys_computer = “yes”) = 4/9 = 0.444
  P(income = “medium” | buys_computer = “no”) = 2/5 = 0.4
  P(student = “yes” | buys_computer = “yes) = 6/9 = 0.667
  P(student = “yes” | buys_computer = “no”) = 1/5 = 0.2
  P(credit_rating = “fair” | buys_computer = “yes”) = 6/9 = 0.667
  P(credit_rating = “fair” | buys_computer = “no”) = 2/5 = 0.4
- **X = (age <= 30 , income = medium, student = yes, credit_rating = fair)**

$P(X|C_i)$: P(X|buys_computer = “yes”) = 0.222 x 0.444 x 0.667 x 0.667 = 0.044
P(X|buys_computer = “no”) = 0.6 x 0.4 x 0.2 x 0.4 = 0.019

$P(X|C_i)*P(C_i)$: P(X|buys_computer = “yes”) * P(buys_computer = “yes”) = 0.028
P(X|buys_computer = “no”) * P(buys_computer = “no”) = 0.007

Therefore, X belongs to class (“buys_computer = yes”)

Avoiding the Zero-Probability Problem

- Naïve Bayesian prediction requires each conditional prob. be **non-zero**. Otherwise, the predicted prob. will be zero

$$P(X \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i)$$

- Use **Laplacian correction** (or Laplacian smoothing)
  - *Adding 1 to each case*
    - $P(x_k = j|C_i) = \frac{n_{ik,j}+1}{\sum_{j'}(n_{ik,j'}+1)}$ where $n_{ik,j}$ is # of tuples in C_i having value $x_k = j$
    - Ex. Suppose a dataset with 1000 tuples, income=low (0), income= medium (990), and income = high (10)

      Prob(income = low) = 1/1003

      Prob(income = medium) = 991/1003

      Prob(income = high) = 11/1003

  - The “corrected” prob. estimates are close to their “uncorrected” counterparts

*Notes on Parameter Learning

- Why the probability of $P(X_k|C_i)$ is estimated in this way?
  - http://www.cs.columbia.edu/~mcollins/em.pdf
  - http://www.cs.ubc.ca/~murphyk/Teaching/CS340-Fall06/reading/NB.pdf

Naïve Bayes Classifier: Comments

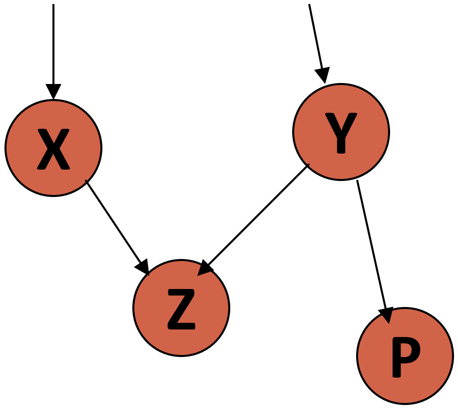
- Advantages
  - Easy to implement
  - Good results obtained in most of the cases
- Disadvantages
  - Assumption: class conditional independence, therefore loss of accuracy
  - Practically, dependencies exist among variables
    - E.g., hospitals: patients: Profile: age, family history, etc.
      Symptoms: fever, cough etc., Disease: lung cancer, diabetes, etc.
    - Dependencies among these cannot be modeled by Naïve Bayes Classifier
- How to deal with these dependencies? Bayesian Belief Networks

Chapter 8&9. Classification: Part 3

- Bayesian Learning
  - Naïve Bayes
  - Bayesian Belief Network
- Instance-Based Learning
- Summary

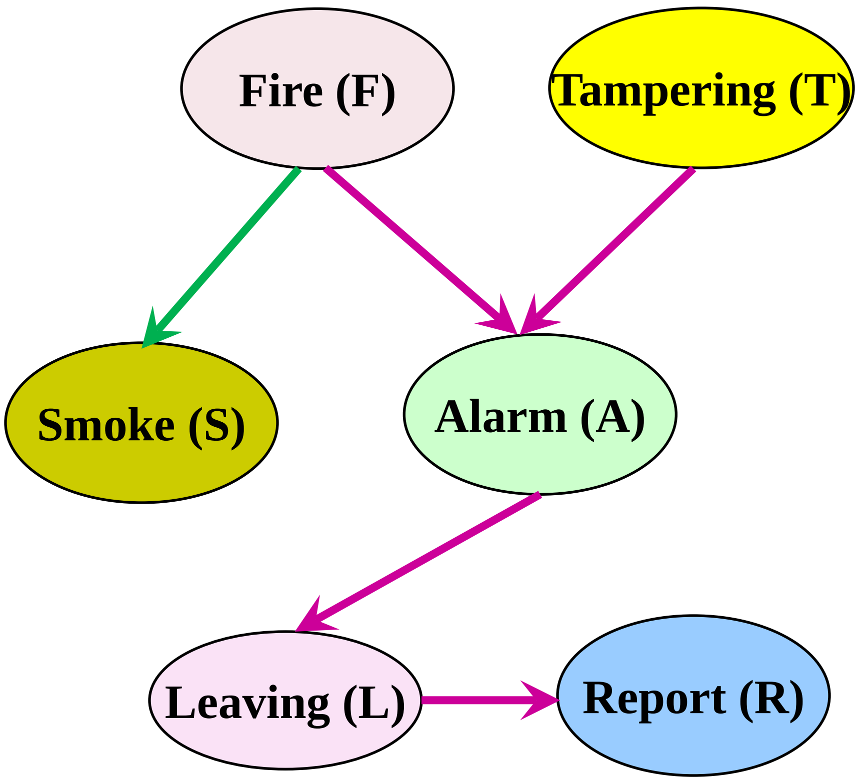
Bayesian Belief Networks (BNs)

- **Bayesian belief network** (also known as **Bayesian network**, **probabilistic network**): allows *class conditional independencies* between *subsets* of variables
- Two components: (1) A *directed acyclic graph* (called a structure) and (2) a set of *conditional probability tables* (CPTs)
- A (*directed acyclic*) graphical model of *causal influence* relationships
  - Represents <u>dependency</u> among the variables
  - Gives a specification of joint probability distribution



- ❑ Nodes: random variables
- ❑ Links: dependency
- ❑ X and Y are the parents of Z, and Y is the parent of P
- ❑ No dependency between Z and P conditional on Y
- ❑ Has no cycles

A Bayesian Network and Some of Its CPTs



CPT: Conditional Probability Tables

	F	¬F
S	.90	.01
¬S	.10	.99

	F, T	F, ¬T	¬F, T	¬F, ¬T
A	.5	.99	.85	.0001
¬A	.95	.01	.15	.9999

CPT shows the conditional probability for each possible combination of its parents

Derivation of the probability of a particular combination of values of **X**, from CPT (joint probability):

$$P(x_1,\ldots,x_n) = \prod_{i=1}^{n} P(x_i \mid Parents(x_i))$$

Inference in Bayesian Networks

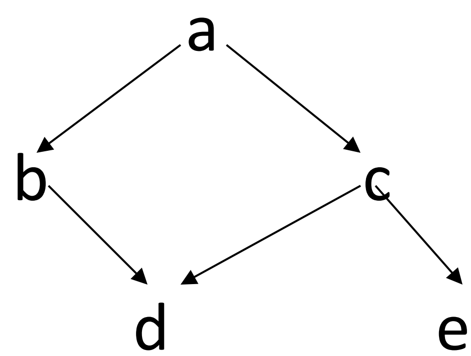
- Infer the probability of values of some variable given the observations of other variables
  - E.g., P(Fire = True | Report = True, Smoke = True)?
- Computation
  - Exact computation by enumeration
  - In general, the problem is NP hard
    - Approximation algorithms are needed

Inference by enumeration

- To compute posterior marginal $P(X_i \mid E=e)$
  - Add all of the terms (atomic event probabilities) from the full joint distribution
  - If **E** are the evidence (observed) variables and **Y** are the other (unobserved) variables, then:

    $P(X|\mathbf{e}) = \alpha\, P(X, \mathbf{E}) = \alpha \sum P(X, \mathbf{E}, \mathbf{Y})$
  - Each P(X, **E**, **Y**) term can be computed using the chain rule
- Computationally expensive!

Example: Enumeration



- P (d|e) = $\alpha\ \Sigma_{ABC}$P(a, b, c, d, e)
  = $\alpha\ \Sigma_{ABC}$P(a) P(b|a) P(c|a) P(d|b,c) P(e|c)
- With simple iteration to compute this expression, there's going to be a lot of repetition (e.g., P(e|c) has to be recomputed every time we iterate over C=true)
  - A solution: variable elimination

How Are Bayesian Networks Constructed?

- **Subjective construction**: Identification of (direct) causal structure
  - People are quite good at identifying direct causes from a given set of variables & whether the set contains all relevant direct causes
  - Markovian assumption: Each variable becomes independent of its non-effects once its direct causes are known
  - E.g., S ← F → A ← T, path S→A is blocked once we know F→A
- **Synthesis from other specifications**
  - E.g., from a formal system design: block diagrams & info flow
- **Learning from data**
  - E.g., from medical records or student admission record
  - Learn parameters give its structure or learn both structure and parms
  - Maximum likelihood principle: favors Bayesian networks that maximize the probability of observing the given data set

Learning Bayesian Networks: Several Scenarios

- Scenario 1: Given both the network structure and all variables observable: *compute only the CPT entries (Easiest case!)*
- Scenario 2: Network structure known, some variables hidden: *gradient descent* (greedy hill-climbing) method, i.e., search for a solution along the steepest descent of a criterion function
  - Weights are initialized to random probability values
  - At each iteration, it moves towards what appears to be the best solution at the moment, w.o. backtracking
  - Weights are updated at each iteration & converge to local optimum
- Scenario 3: Network structure unknown, all variables observable: search through the model space to *reconstruct network topology*
- Scenario 4: Unknown structure, all hidden variables: No good algorithms known for this purpose
- D. Heckerman. A Tutorial on Learning with Bayesian Networks. In *Learning in Graphical Models,* M. Jordan, ed. MIT Press, 1999.

Chapter 8&9. Classification: Part 3

- Bayesian Learning
  - Naïve Bayes
  - Bayesian Belief Network
- Instance-Based Learning
- Summary

Lazy vs. Eager Learning

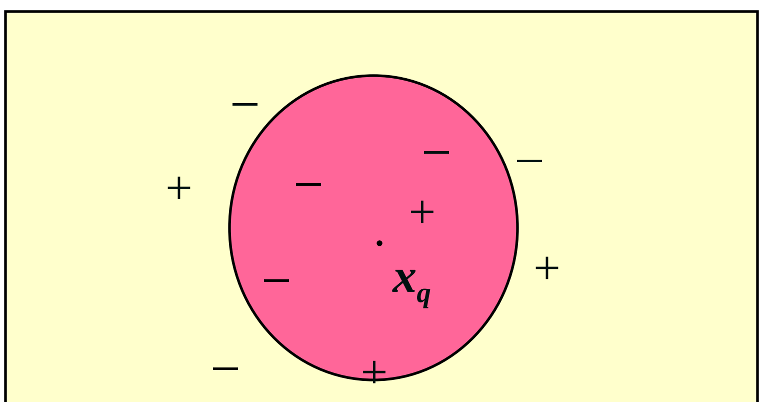
- Lazy vs. eager learning
  - **Lazy learning** (e.g., instance-based learning): Simply stores training data (or only minor processing) and waits until it is given a test tuple
  - **Eager learning** (the above discussed methods): Given a set of training tuples, constructs a classification model before receiving new (e.g., test) data to classify
- Lazy: less time in training but more time in predicting
- Accuracy
  - Lazy method effectively uses a richer hypothesis space since it uses many local linear functions to form an implicit global approximation to the target function
  - Eager: must commit to a single hypothesis that covers the entire instance space

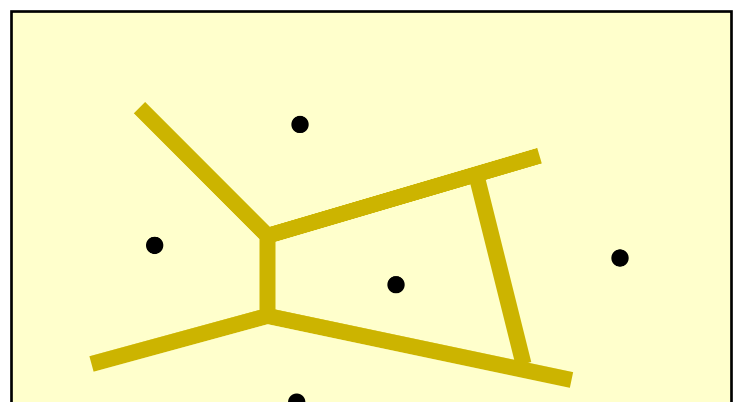
Lazy Learner: Instance-Based Methods

- Instance-based learning:
  - Store training examples and delay the processing ("lazy evaluation") until a new instance must be classified
- Typical approaches
  - *k*-nearest neighbor approach
    - Instances represented as points in a Euclidean space.
  - Locally weighted regression
    - Constructs local approximation

The *k*-Nearest Neighbor Algorithm

- All instances correspond to points in the n-D space
- The nearest neighbor are defined in terms of Euclidean distance, dist($\mathbf{X_1}$, $\mathbf{X_2}$)
- Target function could be discrete- or real- valued
- For discrete-valued, *k*-NN returns the most common value among the *k* training examples nearest to x_q
- Vonoroi diagram: the decision surface induced by 1-NN for a typical set of training examples





Discussion on the *k*-NN Algorithm

- *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 x_q
    - Give greater weight to closer neighbors
    - $y_q = \frac{\sum w_i y_i}{\sum w_i}$, where x_i's are x_q's nearest neighbors

$$w \equiv \frac{1}{d(x_q, x_i)^2}$$

- 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

Chapter 8&9. Classification: Part 3

- Bayesian Learning
  - Naïve Bayes
  - Bayesian Belief Network
- Instance-Based Learning
- Summary

Summary

- Bayesian Learning
  - Bayes theorem
  - Naïve Bayes, class conditional independence
  - Bayesian Belief Network, DAG, conditional probability table
- Instance-Based Learning
  - Lazy learning vs. eager learning
  - K-nearest neighbor algorithm