Data Mining

CIS 467

Dr. Qasem Al-Radaideh

qradaideh@yahoo.com

Yarmouk University

Department of Computer Information Systems

Bayesian Classification

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

Bayesian Theorem

- Given training data **X**, posteriori probability of a hypothesis H, P(H|**X**), follows the Bayes theorem:

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

- Informally, this can be written as

  posteriori = likelihood x prior/evidence

- Predicts **X** belongs to C_2 iff the probability $P(C_i|\mathbf{X})$ is the highest among all the $P(C_k|X)$ for all the *k* classes
- Practical difficulty: require initial knowledge of many probabilities, significant computational cost

Bayesian Theorem

- **Purpose** of the Bayesian Theorem : To Predict the class label for a given tuple .
- P(C/X) **denotes** ….. The probability that tuple X belongs to the class C according some feature of X .
- P(C) **denotes** ….. The Probability that any given tuple belong to this class . **EX**: any customer will buy a computer .
- P(X/C) **denotes** …..The probability that customer X , 35years old and earns $40.000 , given, that we know the customer will buy a computer.
- P(X) **denotes** ….. The probability that a person from our set of customers is 35 years old and earns $40.000 .
- Since P(X) is **constant** for all classes, only we need :

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

Play-tennis example 2 : estimating $P(x_i|C)$

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

P(p) = 9/14
P(n) = 5/14

Naive Bayesian Classifier (II)

- Given a training set, we can compute the probabilities

Outlook	P	N		Humidity	P	N
sunny	2/9	3/5		high	3/9	4/5
overcast	4/9	0		normal	6/9	1/5
rain	3/9	2/5				
Tempreature				Windy		
hot	2/9	2/5		true	3/9	3/5
mild	4/9	2/5		false	6/9	2/5
cool	3/9	1/5				

Estimating $P(x_i | C)$

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

P(p) = 9/14	
P(n) = 5/14	

outlook	
P(sunny|p) = 2/9	**P(sunny|n) = 3/5**
P(overcast|p) = 4/9	**P(overcast|n) = 0**
P(rain|p) = 3/9	**P(rain|n) = 2/5**
temperature	
P(hot|p) = 2/9	**P(hot|n) = 2/5**
P(mild|p) = 4/9	**P(mild|n) = 2/5**
P(cool|p) = 3/9	**P(cool|n) = 1/5**
humidity	
P(high|p) = 3/9	**P(high|n) = 4/5**
P(normal|p) = 6/9	**P(normal|n) = 2/5**
windy	
P(true|p) = 3/9	**P(true|n) = 3/5**
P(false|p) = 6/9	**P(false|n) = 2/5**

Play-tennis example: classifying X

- An unseen sample

  X = <rain, hot, high, false>

- P(X|p)·P(p) =
  P(rain|p)·P(hot|p)·P(high|p)·P(false|p)·P(p) =
  3/9·2/9·3/9·6/9·9/14 = 0.010582

- P(X|n)·P(n) =
  P(rain|n)·P(hot|n)·P(high|n)·P(false|n)·P(n) =
  2/5·2/5·4/5·2/5·5/14 = 0.018286

- Sample X is classified in class n (don't play)

Example:The Buys Computer Dataset

RID	*age*	*income*	*student*	*credit_rating*	*Class: buys_computer*
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Example

- **Predicting a class label for this tuple** :

  X= (age=Youth ,Income=Medium ,Student=Yes , Credit rating=Fair)

First Step

Find Classes:

C1: buys_computer = 'yes'

C2: buys_computer = 'no'

Second Step

Find P(C1) and P(C2)

P(buys_computer = "yes") = 9/14 = 0.643

P(buys_computer = "no") = 5/14= 0.357

Example (Con't)

Third Step

Compute P (X|C) for each class :

P (age = "Youth" / buys_computer = "yes") = 2/9 = 0.222

P (age = "Youth" / 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

Example (Con't)

Fourth Step .

Find P(X|C_i) :

For C1: Multiply all the probabilities that belong to class "yes"

P(X|buys_computer = "yes") = 0.222 x 0.444 x 0.667 x 0.667=.044

Similarly for "class = no"

P(X|buys_computer = "no") = 0.6 x 0.4 x 0.2 x 0.4 = 0.019

Example

Finally ,

find …... P(X|C_i)*P(C_i) ……. for each class

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 0-Probability Problem

- This problem about if one of the probability of one feature = 0
- **EX :**

  P(student =yes / Buy_computer = no) = 0

  what we do ??????
- We use (**Laplacian correction Method**) to avoid this problem

  By adding one to each count that we need

EX: Suppose that for class buy_computer =yes in some training data set containing 1000 tuples .

we have 0 tuple with income = low

have 990 tuple with income = medium

have 10 tuple with income = high

Without Laplacian correction the probability for these tuple

income(low)=0 , income(medium)=990/1000=.99 , income (high)=10/1000=.01

Avoiding the 0-Probability Problem

- But when using **Laplacian correction**
- income(low)=1/1003 , income(medium)=991/1003=.988 ,

  and income (high) = 11/1003 =.011