CS454 Topics in Advanced Computer Science
More Data Mining Algorithms

Chengyu Sun
California State University, Los Angeles

Mining Sequential Patterns

- <{computer},{printer},{printer cartridge}>
- <{bread,milk},{bread,milk},{bread,milk}…>
- <{home.jsp},{search.jsp},{product.jsp},{product.jsp},{search.jsp}…>

Terminology and Notations

- Item, itemset
- Event = itemset
- A sequence is an ordered list of events
  - $<e_1e_2e_3…e_l>$
  - E.g. <(a)(abc)(bc)(d)(ac)(f)>
- The length of a sequence is the number of items in the sequence, i.e. *not the number of events*

Sequences vs. Itemsets

- {a,b,c}
  - # of 3-itemset(s)??
  - # of 3-sequence(s)??

Subsequence

- $A=<a_1a_2a_3…a_n>$
- $B=<b_1b_2b_3…b_m>$
- A is a *subsequence* of B if there exists $1 \le j_1 < j_2 < … < j_n \le m$ such that $a_1 \subseteq b_{j1}, a_2 \subseteq b_{j2}, …, a_n \subseteq b_{jn}$

Subsequence Example

- `s=<(abc)(de)(f)>`
- What are the subsequences of `s`??

Sequential Pattern

- If A is a subsequence of B, we say B *contains* A
- The support count of A is the number of sequences that contain A
- A is *frequent* if `support_count(A)≥min_sup`
- A frequent sequence is called a sequential pattern

Apriori Property Again

- Every nonempty subsequence of a frequent sequence is frequent

GSP Algorithm

- *Generalized Sequential Patterns*
- An extension of the Apriori algorithm for mining sequential patterns

GSP Example

SID	Sequence
1	<(a)(ab)(a)>
2	<(a)(c)(bc)>
3	<(ab)(c)(b)>
4	<(a)(c)(c)>

min_sup=2

L_1

C_1	support_count	L_1
a	4	<(a)>
b	3	<(b)>
c	3	<(c)>

L_2

C_2	support_count	L_2
<(a)(a)>	1	
<(a)(b)>	3	<(a)(b)>
<(a)(c)>	3	<(a)(c)>
<(b)(a)>	1	
<(b)(b)>	1	
<(b)(c)>	1	
<(c)(a)>	0	
<(c)(b)>	2	<(c)(b)>
<(c)(c)>	2	<(c)(c)>
<(ab)>	2	<(ab)>
<(ac)>	0	
<(bc)>	1	

From L_{k-1} to C_k

- Two sequences s_1 and s_2 are joinable if the subsequence obtained by dropping the first item in s_1 is the same as the subsequence obtained by dropping the last item in s_2
- The joined sequence is s_1 concatenated with the last item i of s_2
  - If the last two items in s_2 are in the same event, i is merged into the last event of s_1;
  - Otherwise i becomes a separate event

L3

C_3	support_count	L_3
<(a)(c)(b)>	2	<(a)(c)(b)>
<(a)(c)(c)>	2	<(a)(c)(c)>

Candidate Pruning

- A k-sequence can be pruned if one of its (k-1)-subsequence is not frequent

L_3	Candidate generation → C_4	Pruning → C_4
<(1)(2)(3)>	<(1)(2)(3)(4)>	<(1)(2 5)(3)>
<(1)(2 5)>	<(1)(2 5)(3)>	
<(1)(5)(3)>	<(1)(5)(3 4)>	
<(2)(3)(4)>	<(2)(3)(4)(5)>	
<(2 5)(3)>	<(2 5)(3 4)>	
<(3)(4)(5)>		
<(5)(3 4)>		

Probabilistic Relationship between Attributes and Class

- Ten middle-aged, divorced, male borrowers have defaulted on their loans, but would the 11th one default as well?

Bayes' Theorem

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

- Prior and posterior probabilities
  - P(A) and P(A|B)
  - P(B) and P(B|A)

Bayesian Classification

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

- **x** is a given record with attribute values $(x_1, x_2, \ldots, x_n)$, and C_i is a class
- $P(C_i|\mathbf{x})$ is the probability of **x** belonging to class C_i given **x**'s attribute values
- We predict that **x** belong to C_i if $P(C_i|X) > P(C_j|X)$ for $j \neq i$

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

- $P(\mathbf{X})$ does not need to be calculated
  - *Why??*
- $P(C_i)$??
- $P(\mathbf{X}|C_i)$??

Naive Bayesian Classification

- $\mathbf{X}=(x_1,x_2,\ldots,x_n)$
- Assume the attribute values are conditionally independent of one another (the *naive* assumption)

$$P(\mathbf{X} \mid C_i) = \prod_{i=1}^{n} P(x_i \mid C_i)$$
$$= P(x_1 \mid C_i) \times P(x_2 \mid C_i) \times \cdots \times P(x_n \mid C_i)$$

Sample Data

TID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
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
11	No	Married	120K	??

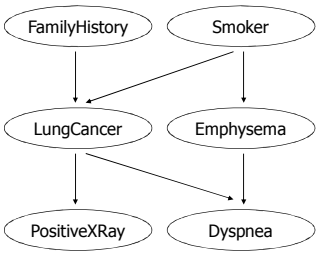
Bayesian Belief Network (BBN)

- A *directed acyclic graph* (dag) encoding the dependencies among a set of variables
- A *conditional probability table* (CPT) for each node given its immediate parent nodes

A BBN Example

CPT for LungCancer

	Yes	No
FH,S	0.8	0.2
FH,!S	0.5	0.5
!FH,S	0.7	0.3
!FH,!S	0.1	0.9



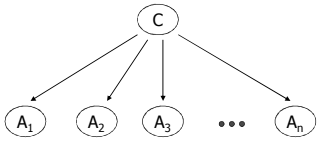
BBN Terminology

- If there is a directed *arc* from X to Y
  - X is a parent of Y
  - Y is a child of X
- If there is a directed *path* from X to Y
  - X is an ancestor of Y
  - Y is a descendent of X

Conditional Independence in BBN

- A node in a Bayesian network is conditionally independent of its non-descendants if its parents are known

Naive Bayesian is a Special Case of BBN



Construct a BBN

- Create the structure of the network
  - From domain knowledge
  - From training data
- Calculate the CPT for each node X
  - P(X) if X does not have any parent
  - P(X|Y) if X has one parent Y
  - $P(X|Y_1,Y_2,...,Y_k)$ if X has multiple parents $\{Y_1,Y_2,...,Y_k\}$

Another BBN Example

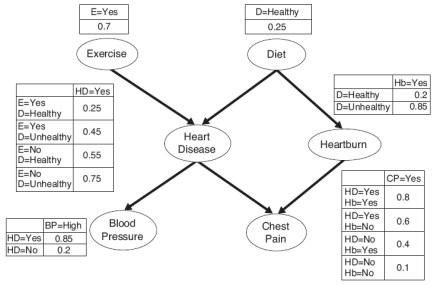


Figure 5.13. A Bayesian belief network for detecting heart disease and heartburn in patients.

Bayesian Classification Examples

- Output node – Heart Disease
- Testing data
  - ()
  - (BP=high)
  - (BP=high,D=Healthy,E=Yes)

Bayesian Classification Examples – 1

$$P(HD=Yes)=\sum_{i=1}^{n}\sum_{j=1}^{m}P(HD=Yes\mid E=a_i,D=b_j)P(E=a_i,D=b_j)$$

$$=\sum_{i=1}^{n}\sum_{j=1}^{m}P(HD=Yes\mid E=a_i,D=b_j)P(E=a_i)P(D=b_j)$$

$$=0.49$$

Bayesian Classification Examples – 2

$$
\begin{aligned}
&P(HD = Yes \mid BP = High) \\
&= \frac{P(BP = High \mid HD = Yes)P(HD = Yes)}{P(BP = High)} \\
&= \frac{P(BP = High \mid HD = Yes)P(HD = Yes)}{\sum_{i=1}^{n} P(BP = High \mid HD = a_i)P(HD = a_i)} \\
&= 0.80
\end{aligned}
$$

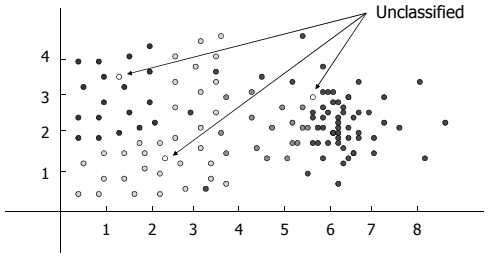
Bayesian Classification Examples – 3

$$
\begin{aligned}
&P(HD = Yes \mid BP = High, D = Healthy, E = Yes) \\
&= \frac{P(BP = High \mid HD = Yes, D = Healthy, E = Yes)P(HD = Yes \mid D = Healthy, E = Yes)}{P(BP = High \mid D = Healthy, E = Yes)} \\
&= \frac{P(BP = High \mid HD = Yes)P(HD = Yes \mid D = Healthy, E = Yes)}{\sum_{i=1}^{n} P(BP = High \mid HD = a_i)P(HD = a_i \mid D = Healthy, E = Yes)} \\
&= 0.59
\end{aligned}
$$

About BBN

- Does not assume attribute independence
- Provides a way to encode domain knowledge
  - Robust to model overfitting
- Any node can be used an output node

k Nearest Neighbor (kNN) Classification Example



- *What is the class of each unclassified sample??*

kNN Classification

- Find the `k` nearest neighbors of the test sample
- Classify the test sample with the majority class of its `k` nearest neighbors

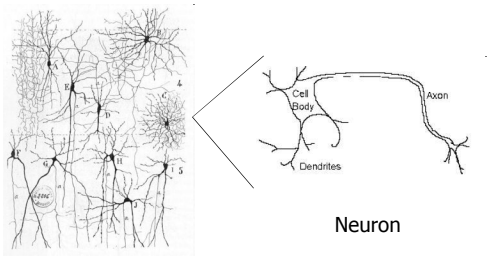
About kNN

- Similarity/distance measures
- Index structures
- Local decision – susceptible to noise

Artificial Neural Network (ANN)

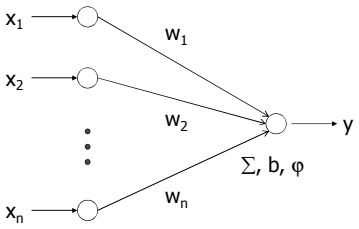
- Simulates the learning process of biological neural network

Neural Network



Neuron

Perceptron



About Perception

- Input nodes x_i
- Output node y
- Weights w_i
- Bias b
- Activation function φ
- Can be applied to linearly separatable classification problems

$$y = \varphi(\sum_{i=1}^{n} w_i x_i + b)$$

Common Activation Functions

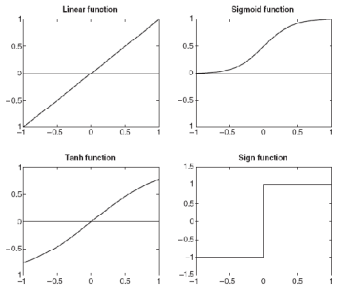


Figure 5.18. Types of activation functions in artificial neural networks.

Learning with a Perceptron

- Initialize weights with random values
- For each training record
  - Compute the predicted output y'
  - For each weight w_i
    - Update the weight $w_i = w_i + \lambda(y-y')x_i$

y is the correct output, λ is the learning rate.

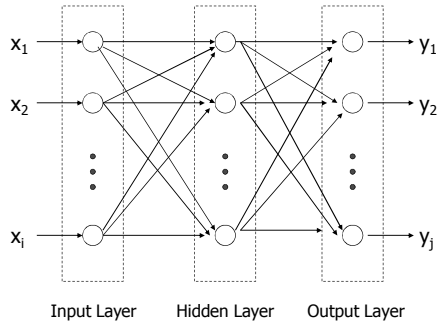
Perceptron Example: Data

X1	X2	X2	y	
1	0	0	-1	
1	0	1	1	Training set
0	1	0	-1	
0	1	1	1	
1	1	0	1	
0	0	1	-1	Testing set
1	1	1	1	
0	0	0	-1	

Perceptron Example: Parameters

- Input nodes: x_1, x_2, x_3
- Output node: y
- Weights: 0.1, 0.1, 0.1
- Bias: -0.5
- Learning rate: 0.25
- Activation function: sign function

A Feed-Forward ANN



Adjust Weights

- Minimize the total sum of squared error

$$E(\mathbf{w}) = \sum (y_i - y_i')^2 \qquad w_j \leftarrow w_j - \lambda \frac{\partial E(\mathbf{w})}{\partial w_j}$$

- Back-propagation
  - The true output of hidden layers are known
  - Propagate errors backwards

Design Issues in ANN

- Number of input nodes
- Number of output nodes
- Network topology
- Weights and biases
- Learning rate

About ANN

- ANN with at least on hidden layer are *universal approximators*
- It's difficult to decide the topology of a network
  - Number of hidden layers
  - Number of hidden nodes
  - Feed-forward or recurrent
- Training can be expensive, but classification is very efficient