CS412: Introduction to Data Mining **Fall 2015**

Assignment 4

Due: 11/19/2015 11:59pm

General Instruction

- Errata: After the assignment is released, any further corrections of errors or clarifications will be posted at the Errata page at Piazza. Please watch it.
- Feel free to talk to other members of the class while doing the homework. We are more concerned that you learn how to solve the problem than that you solve it entirely on your own. You should, however, write the solution yourself.
- Please use Piazza first if you have questions about the homework. Also feel free to send us e-mails and come to office hours.
- For each question, you should show the necessary calculation steps and reasoning—not only final results. Keep the solution brief and clear.
- For a good balance of cognitive activities, we label each question with an activity type:
  - **L1 (Knowledge)** Definitions, propositions, basic concepts.
  - **L2 (Practice)** Repeating and practicing algorithms/procedures.
  - **L3 (Application)** Critical thinking to apply, analyze, and assess.

Assignment Submission

- Please submit your work before the due time. **We do NOT accept late submission!**
- Please submit your answers electronically via Compass. Contact CITES/TAs if you have technical difficulties in submitting the assignment.
- For this assignment, **typeset** your answers and submit it in a **single PDF file**. **Handwritten answers or hand-drawn pictures are not acceptable**.

1 Constraint pattern mining (5 points)

For the following constraints, write down your answer if they are (strongly convertible, convertible) Anti-monotone, Monotone or Succinct.

Purpose

- Have a better understanding of constraint pattern mining.

Requirements

- A short explanation to justify if the constraint belongs to **EACH** type is required. To show a constraint is not of one type, use a simple example.

a. (1', L1) for a set of values S, and a value v, constraint $v \in S$

b. (1', L1) for a set of values S, and a value v, constraint $max(S) \geq v$

c. (1', L1) for a set of values S, and a value v, constraint $max(S) \leq v$

d. (2', L2) for a set of values S, and a value v, constraints $avg(S) \geq v$ and $avg(S) \leq v$

2 Advanced pattern mining (10 points)

For each question below, indicate if the statement is **true** or **false**.

Purpose

- Have a better understanding of advanced pattern mining.

Requirements

- For each sub-question, select **true (T)** or **false (F)** and also write down a brief explanation. You will **NOT** get credit without an explanation.

a. (2', L2) **T/F.** Convertible constraint porperty cannot be exploited in an Apriori-based mining algorithm.

b. (2', L2) **T/F.** In *PrefixSpan*, physical project is not used because of its slow performance.

c. (2', L1) **T/F.** Mining closed frequent graphs only is lossless compression of the graph database.

d. (2', L3) **T/F.** In sequential pattern mining, the number of length-2 candidates generated from x frequent length-1 patterns is $\frac{3}{2}x^2 - \frac{1}{2}x$

e. (2', L3) **T/F.** For nontrivial constraints (trivial constraints are those that are satisfied by every possible pattern), it cannot be monotone and anti-monotone at the same time.

3 Sequential pattern mining (20 points)

Use a toy dataset to perform sequential pattern mining algorithms.

Purpose

- Work on sequential pattern mining using *GSP* and *PrefixSpan*

Requirements

- Write down each step as detail as possible.

Suppose a toy sequence database D contains three sequences as follows. Let the minimum support be 3.

customer_id	shopping sequence
1	$(bc)(de)f$
2	$bcdef$
3	$(bc)dbegf$

The following questions require you to perform GSP algorithm.

(1) (2', L2) Scan database once, list length-1 sequential pattern candidates C_1 and the result L_1 after pruning.

(2) (3', L2) Following, generate C_2, and L_2. *Hint: do not miss any candidates in* C_2

(3) (3', L3) Now, generate C_3, L_3, C_4, L_4 and longer candidates/results until the algorithm terminates.

Now, Using the same DB, apply PrefixSpan to get the same results:

(4) (2', L2) Scan database once, and get the length-1 prefix list *P1*.

(5) (3', L2) For each prefix in *P1*, generate its projected database.

(6) (3', L3) complete the PrefixSpan algorithm from the result above. For each step, list the prefix and its corresponding projected DB.

Comparing the two algorithms, answer the following questions:

(7) (2', L3) What is the major difference between the two algorithms?

(8) (2', L3) Based on the above analysis, can you tell which one will outperform the other? And in what contidition/situation particularly?

4 Decision Trees (18 points)

ID3 is a simple algorithm for decision tree construction using information gain. The steps of the ID3 algorithm are similar to those introduced in the lecture. In particular, ID3 uses information gain to select decision attributes, and each attribute is used at most once in any root-to-leaf path in the decision tree. You will use ID3 to build a decision tree that predicts whether a candidate will be accepted to the PhD program of some University X, given the student's information about GPA, university, publications, and recommedation.

Purpose

- Understand and practice basic decision tree construction, calculation of information gain measures, and classifier evaluation.

Requirements

- Show the calculations for selecting the decision tree attributes and the labels for each leaf.

a. (6', L2) Using the ID3 algorithm to construct a decision tree using the training data in Table 1. When multiple attributes has best information gain, choose the one whose name appears earliest in alphabetical order. When there is a tie for the majority labels, choose `no`. Show the final decision tree, and the calculations to derive that tree.

id	GPA	univ	published	recommendation	accepted
1	4.0	top-10	yes	good	yes
2	4.0	top-10	no	good	yes
3	4.0	top-20	no	normal	yes
4	3.7	top-10	yes	good	yes
5	3.7	top-20	no	good	yes
6	3.7	top-30	yes	good	yes
7	3.7	top-30	no	good	no
8	3.7	top-10	no	good	no
9	3.5	top-20	yes	normal	no
10	3.5	top-10	no	normal	no
11	3.5	top-30	yes	normal	no
12	3.5	top-30	no	good	no

Table 1: Training Data for the decision tree problem

id	GPA	univ	published	recommendation	accepted
1	4.0	top-10	yes	good	yes
2	3.7	top-30	yes	good	yes
3	3.5	top-30	yes	good	yes
4	3.7	top-10	no	good	no
5	3.5	top-30	no	good	no

Table 2: Testing Data for the decision tree problem

b. (4', L2) Evaluate your constructed decision tree using the testing data in Table 2 in terms of the precision and recall for the class `yes`. Show your calculations.

c. (4', L2) What is the worst case time complexity of training a decision tree using ID3 on a dataset with n data records and m attributes each having p possible values? Show your analysis.

d. (4', L3) Each root-to-leaf path in any decision tree can be converted into a rule, such as a path $A_1 \xrightarrow{=True} A_2 \xrightarrow{=False} class = +1$ can be converted to the rule "If attribute A_i is true and attribute A_2 is false, then the instance has class +1". Please do/answer the following:

1. Generate the rules for each leaf of your constructed decision tree.
2. Is it possible to construct a decision tree from a set of rules? Explain your answer.

5 AdaBoost (16 points)

You will be guided through the steps of building an ensemble classifier using AdaBoost. The data points to be classified are given in Table 3. Each classifier in the ensemble will have *one* of the following forms:

- If $x > a$, label +1, else label -1
- If $x \geq a$, label +1, else label -1
- If $y < b$, label +1, else label -1
- If $y \leq b$, label +1, else label -1

where a and b are constants for you to figure out. That is, the hypothesis of the classifier can be represented by a line parallel to the y-axis or the x-axis. While the original AdaBoost algorithm trains each base classifier on *sampled* data points, you will simulate AdaBoost (deterministically) by picking each base classifier given *all* the data points such that the base classifier minimizes the weighted error rate.

id	x	y	**label**
1	1.0	0.5	+1
2	2.2	1.0	+1
3	2.7	2.0	+1
4	0.5	1.5	-1
5	1.2	2.3	-1
6	1.5	2.7	-1

Table 3: Data points for the AdaBoost problem

Purpose

- Understand and practice AdaBoost algorithm by walking through the steps.

Requirements

- Show all the steps and calculations needed to derive each classifier.
- In case of ties when selecting classifiers, pick one that corresponds to a line parallel to the y-axis; if the tie still exists, pick one with minimum a or b.

a. (2', L2) Assume that data weight distribution D_1 in Round 1 is uniform. Find classifier h_1 that has minimum weighted error with data weight distribution D_1. (*Note: see requirements for breaking ties when choosing from equally good classifiers*).

b. (2', L2) What is the weighted error rate of classifier h_1 with data weights D_1?

c. (2', L2) After re-weighting the data according to the results from Round 1, what is the updated data weight distribution D_2 for Round 2? Normalize the weights so that they sum to 1.

d. (2', L2) Find classifier h_2 for Round 2 that have the minimum weighted error rate for the data weight distribution D_2.

e. (4', L2) Similar to (c) and (d), compute the weight distribution D_3 and find a classifier h_3 for Round 3.

f. (4', L3) What is the ensemble classifier h' that combines h_1, h_2, and h_3? Show h' by plotting the given data points in a 2-D plane and highlighting the regions where points would be classified as $+1$ by h'.

6 Bayes Classifier (16 points)

Using the same training data as in Table 1, your will train a classifier using the Naive Bayes method.

Purpose

- Understand and practice the principles of Naive Bayes classifier and its training algorithm; compare the trained classification models to see their pros/cons.

Requirements

- Show the steps and calculations to derive the classifier.
- Show the formulas you used to calculate the results.

a. (1', L2) What is the prior probability of `accepted` being `yes`/`no` estimated from the data?

b. (3', L2) What is the conditional probability of attribute in `GPA` taking each of the values in $\{4.0, 3.7, 3.5\}$, given `accepted=yes`? Also calculate the conditional probabilities for each of attributes {`university, published, recommendation`} taking each of its possible values given that `accepted=yes`.

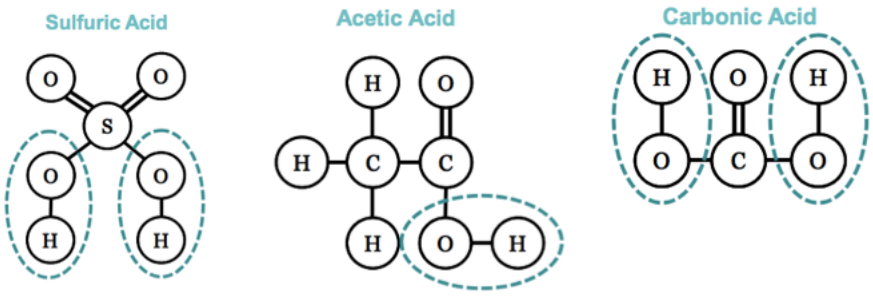
c. (3', L2) Calculate similiar conditional probabilities asked in (b) with the conditions replaced by `accepted=no`.

d. (3', L2) Based the results you got from (a)-(c), given a student with attributes (`GPA=3.7, university=top-20, published=yes, recommedation=good`), calculate the probability of the student being accepted. What will the probability become if the student has (`GPA=3.7, university=top-30, publication=no, recommendation=normal`)?

e. (2', L2) Consider a traning dataset with n tuples and m attributes, and assume each attributes can take k possible values. What is the time complexity of training a Naive Bayes classifier? Show your analysis.

f. (4', L3) Discuss the pros and cons of the classification models have trained using decision trees and Naive Bayes (Name one pro and one con for each model).

7 Frequent Subgraph Pattern Mining (15 points)



gSpan is an algorithm for mining frequent graph patterns. On the course page, download the gSpan package and complete the following questions.

Purpose

- Work on graph mining using *gSpan*
- Learn how to utilize gSpan package by converting graph and pattern into files of correct format.

Requirements

- For each question, draw all the pattern graphs as given from the gSpan algorithm. Like what we did in class, we start with a tree of empty root node and generate patterns by adding new nodes as children of the pattern tree.

Here is three molecular formulas of acids that we are interested in. Use gSpan to do the following:

(1) (5', L2) Practice the algorithm **by hand** to find **ALL** patterns that appear in all three molecular structures. Draw the pattern grow tree and list the pattern and its frequency of each node. **For this question, it is fine to draw the tree by hand and take a picture of it and paste in the answer.** (*Hint: the frequency is counted by the number of molecular structures it appears, not by its number of occurrences. E.g., the frequency of O-H is 3, though it appears four time.*)

(2) (5', L2) Now, use the gSpan **package** to find **ALL** patterns that appear in at least two molecules. Note that the executable binary we have is compiled on Linux. You will need to use a Linux machine (such as EWS) to run the algorithm (though you can hack a little bit to run Linux binary on Mac by emulating — do a Google search for related results.)

(3) (5', L2) Find the closed patterns from the result of (2). Considering that sulfuric acid is a typical inorganic acid while acetic acid is organic, what can you say that determins the acidity of organic/inorganic compound in general?