CS 301 – Intro to Data Mining
Project
Programming Option

- This assignment is **due by noon on Friday, December 6, 2013**.
- This assignment will be worth **25%** of your course grade.
- You are to work on this assignment **with exactly one other person** who is enrolled in the class. If you are enrolled in the *distance ed* section of the course, contact Dr. Leopold at leopoldj@mst.edu
- You are to do **either** the *Programming Option* or the *Non-Programming Option* of the project, **not** both.

Basic Instructions:

You are to write a program to implement the **rule induction from coverings** algorithm that was discussed in class to produce **association rules**. The program can be written in **any programming language**, but must compile and execute on one of the **campus Linux machines** (such as rc*nn*ucs213.managed.mst.edu where ***nn*** is **01-08**); it **cannot** be dependent on any other machine!

Input File Format:

Your program should accept input files in the **Weka *arff*** file format. Data values may be numeric or nominal. Missing data values will be designated as ***?*** and should **not** be removed or altered (i.e., no data cleaning); ***?*** should be treated as a legitimate value. Duplicate rows are possible in the dataset, but your program does not to have handle cases of rows with multiple classifications (e.g., having both a row (***outlook*** = sunny, ***play*** = yes) and a row (***outlook*** = sunny, ***play*** = no) where ***play*** is specified as the decision attribute).

Input Parameters:

When executed, your program should prompt the user for the following:

- Name of the input file
- A selection of the attributes that are to be considered the decision attributes; an enumerated menu would be nice here so that the user doesn't have to type out the names of the attributes they want, and can't enter names of attributes that don't actually exist in the input file
- The maximum number of attributes to consider for a covering (e.g., maybe we only want to see coverings of size 3 or smaller)
- The minimum coverage required for a rule to be reported (e.g., maybe we only want to see rules that apply to at least 3 rows)

- Whether or not unnecessary conditions should be dropped in the rules that are reported

Output:

Your program should produce **neatly formatted** output that includes the following:

- Values for all of the input parameters
- Distribution of values for the decision attributes, both individually and combined (e.g., for the ***weather*** dataset if the user specifies that the decision attributes are ***outlook*** and ***play***, you should output the number of occurrences of each value for ***outlook*** (sunny, overcast, and rainy), the number of occurrences of each value for ***play*** (yes, no), and the number of occurrences of each possible combination of ***outlook*** and ***play*** values ((sunny, yes), (overcast, yes), (rainy, yes), (sunny, no), etc.))
- All coverings, and for each covering the rules, and for each rule the coverage of that rule

Because the datasets can contain many attributes, we do not want to report the rules in a wordy sentence format (e.g., "if … and … and …. then …"). Instead your output for the rules for each covering should look like the following:

Decision attributes: [a77]
Distribution of values for attribute a77:
  Value: 0 Occurrences: 23
  Value: 1 Occurrences: 2
  Value: ? Occurrences: 0
Rules for covering [a72, a75]:
[[[0, 0, 0], 6], [[0, 1, 0], 9], [[1, 0, 0], 2], [[1, 1, 1], 2], [[?, 0, 0], 3], [[?, 1, 0], 3]]

The tuple [[0, 1, 0], 9] is being reported for covering [a72, a75] for decision attribute set [a77]. This tuple would be interpreted as "if a72 = 0 and a75 = 1 then a77 = 0", with that rule applying to 9 instances in the dataset.

The user can specify that s/he wants unnecessary conditions dropped in the reported rules. If you drop a condition in a rule, that attribute should be output as _ in the tuple. For example, in the tuple [[?, 0, 0], 3], which would be interpreted as "if a72 = ? and a75 = 0 then a77 = 0", if we determine that the condition "a75 = 0" can be dropped (and the user wants us to drop unnecessary conditions), then we would report that rule as [[?, _, 0], 3].

Once you have dropped conditions in the rules (assuming the user wanted you to), you should go back through the rule set to see if any rules can be

"combined." For example, suppose we find that we can drop attribute a75 in both [[?, 0, 0], 3] and [[?, 1, 0], 3]], resulting in "new" rules [[?, _, 0], 3] and [[?, _, 0], 3]]. In that case, the output should be [[?, _, 0], 6], rather than showing two separate instances of the rule [?, _, 0].

Testing:

It is **your** responsibility to **rigorously** test your program. There are several *arff* files of varying sizes available with the Weka download and/or you can create some simple datasets of your own. You should be familiar enough with the rule induction from coverings algorithm to be able to check your program's results by hand for small datasets.

In particular, you **must** test your program on two data files posted on the Blackboard website: **table3_10_fg.arff** and **wilkinsonMatrix.arff**. One file is small enough that you should be able to test all possible combinations of sets of decision attributes and easily check the results by hand. The other file is much larger; however, **you are expected to perform a sufficient number and variety of tests to demonstrate the soundness and completeness of your program!**

Don't forget to test your program for **ALL** the input parameters including **maximum number of attributes to consider for a covering, minimum coverage required for a rule to be reported,** and **dropping unnecessary conditions from rules.**

What To Submit For Grading:

You will need to make a *zip* file containing the following:

- Well-commented source code files for your program.
- A text file containing simple instructions for how to compile and execute your program on one of the cslinux computers.
- A report in pdf format that includes the details of a variety of test cases (e.g., input file contents, input parameters, and output) to sufficiently show that your program correctly performs all the required functionality for the project. **If you do not submit this report, you will receive a zero on the project!**
- Any data files you tested (other than **table3_10_fg.arff** and **wilkinsonMatrix.arff**), the results of which are included in your pdf report.

Name this file using the combination of the last names of the two people who worked on the project (e.g., if John Smith and Jane Doe worked together, their submission should be named *smithdoe.zip* or *doesmith.zip*). Submit **only** this *zip* file via Blackboard. Below are instructions. You can submit multiple times before the deadline; only your last submission will be graded.

Submitting from Blackboard:

1. Go to **Assignments** -> **Project (Programming)** (although it may not appear that you can click on **Project (Programming)**, you actually can)
2. Scroll down to **Section 2. Assignment Materials**. Under the text box, click on “Browse My Computer” to attach a file. Find and attach your *zip* file.
3. Scroll to the top or the bottom, and click submit. Confirm your submission.