This exercise revolves around the SD and EMM tool *SubDisc*, as a GUI-based tool, and as a Python package. We’ll start with the GUI. Information about SubDisc can be found here: <https://github.com/SubDisc/SubDisc>

Amongst the assignment downloads, you will find a file *subdisc-gui.jar*. SubDisc is written in Java (which you don’t need to be familiar with), and only runs on your computer when you have a Java Runtime Environment. Most computers already have such a JRE, or come with one readily installed. If yours doesn’t have one, you should download (from Oracle) one and install it. Run the SubDisc GUI to start your Subgroup Discovery exercise with the following command from within the correct folder (where the data files and the .jar reside):

> java -jar subdisc-gui.jar

# Section 1

The assignment starts with a UCI dataset called Adult, provided as adult.txt, which contains data about 1000 American citizens. Open the file in SubDisc and inspect its properties. You can select ‘Browse…’ to see the actual data. Adult has a single binary target called ‘target’, which you should select on the top right, and a target value ‘gr50K’, which should also be selected. This value indicates whether a person earns more than $ 50k on an annual basis.

The default quality measure selected under ‘single nominal’ is Cortana Quality. Descriptions of some quality measures can be found here:

<https://github.com/SubDisc/SubDisc/wiki/Quality-Measure>

Make sure that refinement depth is set to 1, which means subgroups of a single condition. Then, press ‘Subgroup Discovery’. Inspect the subgroups that you get. Press the ‘ROC’ button. What is the area under the curve?

# Section 2

Set the numeric strategy from ‘bins’ to ‘best’. ‘bins’ indicates that for each numeric attribute encountered, only some thresholds need to be considered (based on equal-width histograms). ‘best’ indicates that all thresholds need to be considered, which obviously takes longer. ‘best’ will continue searching with only the best subgroup. ‘bins’ continues with all (promising) thresholds.

Run SD again, and see if you get better subgroups for the numeric attributes. How is the result list different from the previous one?

# Section 3

Let’s look for slightly deeper subgroups by setting the refinement depth to 2. Because this potentially produces many results, let’s raise the bar. The default measure minimum is set to 0.1, which is not very restrictive. Set this to 0.25 (25% of the maximum) and run SD. How is this different?

Leave refinement depth at 2 for the remainder.

# Section 4

By default, SubDisc runs generalization-aware subgroup discovery (controlled by the Filter Subgroups checkbox). Try switching this off, and compare the number of results with and without filtering (Section 3). What spurious subgroups are found, and why should they be filtered?

Compute a Pattern Team of size 3. Inspect the contents of the two opened windows.

After completing this, turn Filter Subgroups back on.

# Section 5

Let’s try a different quality measure, Relative Lift, that doesn’t involve the size of the subgroup. Set the measure minimum to 0.0. Inspect the ROC plot, and note how the isometrics run for Relative Lift. Where are the small subgroups located?

# Section 6

Let’s look for some really small (and good) subgroups: set the minimum coverage to 5 (this means at least 5 members of each subgroup). Raise the minimum quality to 3. Where are the subgroups located in ROC space? What does it mean to have such a low False Positive Rate?

# Section 7

So much for the classification setting (target is nominal). Let’s switch to regression (target is numeric). Select ‘single numeric’ as target type, and ‘age’ as the target attribute (there is no target value now). What is the average age in the dataset? What’s the default quality measure for single numeric? Set measure minimum to 0.0. Set the minimum coverage back to 10% of the dataset.

How is the age distribution of the best subgroup compared to that of the entire dataset (Show Model)?

# Section 8

So far, we have been setting the measure minimum by hand. Ideally, we would set a measure minimum that is based on the significance of the results, given the extent of the search. This can be achieved by the Compute Threshold button. Pressing this starts a large number (default 100) SD runs with a swap-randomised target. The result of this (the Distribution of False Discoveries) produces a normal distribution that can be used to estimate the measure minimum corresponding to a p-value of 0.05. After the corresponding measure is computed, it is automatically filled in in the measure minimum field. Run (normal) SD with this setting.

# Section 9

In this final section, we’ll quickly try an EMM run. Open ameshousing.txt, and set the target type to double regression. The quality measure will be Sign. of Slope Diff. (complement). The Ames Housing dataset contains 79 explanatory variables describing (almost) every aspect of residential homes in Ames, Iowa. As the sale price could depend on the lot area, we will select these two attributes (‘Lot Area’ and ‘SalePrice’) as the primary and secondary target. What linear model represents their relationship on the entire dataset? How many dollars per square feet does this model use?

Now run SD/EMM with this target concept at depth = 1, and investigate the best subgroup found. How does its linear model differ from the base model? How significant is it. Analyse the plot.

# SubDisc in Python: pySubDisc

Most of the actions in the SubDisc tool can be reproduced in code, both in R and in Python. Here, we will experiment with the latter: the pySubDisc package. The pySubDisc GitHub repository can be found here: <https://github.com/SubDisc/pySubDisc>

The pySubDisc package is available from PyPI. To install it, run:

python3 -m pip install pysubdisc

The code for **Section 1** has already been filled in for you. Examine the output. Before the actual run is started, the default parameters are printed; examine these. The default setting asks for subgroups of depth = 1, for the nominal target ‘target’. Of the two possible target values, ‘gr50K’ is selected. The SD-run is executed with the default verbose=True, which causes the algorithm to print debug information[[1]](#footnote-1). In this case, you see which attributes are considered and what potential thresholds are found, along with the quality. Finally, a list of subgroups is printed. Check these, and try to understand their details.

Continue completing the implementation of Sections 2 to 9 by filling in code where-ever … appears. If in the description above, there is a question, this is mostly for you to investigate and understand what is going on. So no code is required to answer these questions. Also, we there is a description of drawing a plot, you can skip this in the code. The code should not open any plot windows.

Final tip: the pySubDisc repository contains some example code.

# What to submit

Your final submission should be the file PA4\_XXXXXXX.py (replaces Xs with your student number (no leading s)) with the missing code filled in correctly. The results of the GUI exercise do not need to be handed in. That just serves as a preparation for the pySubDisc part of the assignment, and to make you familiar with SubDisc. The submitted code is tested for correctness of the approach, not for the correctness of the SD/EMM results.

1. You can set verbose=False for the other runs, if you don’t want to be bothered by too many intermediate results. [↑](#footnote-ref-1)