School of Computer Science and Software Engineering
University of Wollongong
CSCI316 – Big Data Implementation and Techniques - 2018

Assignment 1 – 10 Marks
Due: 07/September/2018 before 23:55 AEST (Australian Eastern Standard Time).

Submission must be done online by using the submission link associated with assignment 1 for this subject on MOODLE. One PDF document and three python files (one for each of the three tasks) are to be submitted. The PDF must contain typed text of your answers (do not submit a scan of a handwritten document). The document should include computer generated graphics and illustrations (hand drawn graphics and illustrations will be ignored). The size limit for submitted material is 20MB. All questions are to be answered. A clear and complete explanation and analysis needs to be provided with each answer.
Submissions made after the due time will be assessed as late submissions. Late submissions are counted in full day increments (i.e. 1 minute late counts as a 1 day late submission). There is a 25% penalty for each day after the due date. The submission site closes four days after the due date. No submission will be accepted after the submission site has closed.
This is an individual assignment. Plagiarism of any part of the assignment will result in having 0 marks for the assignment and for all students involved.

You may need to do some research on background information for this assignment. For example, you may need to develop a deeper understanding of writing code in Python.

What you need:
From the subjects Moodle site download the file:
`kddcup.data_10_percent.csv`

For Task 1, Task 2, and Task 3 you may use the following python libraries: argparse, cairocffi, math, numpy, pandas, platform, time, and matplotlib. Do not use any other libraries for the tasks in this assignment.

Preface:
Self-Organizing Maps are very useful to Big Data. They can be used to

- reduce the dimensionality of a learning problem, and
- obtain a deeper insight into the problem domain, and
- visalize high-dimensional data, and
- serve as a pre-processor for cluster analysis, and
- enrich data through augmentation for further processing.

In this assignment you will implement the online version of the Self-Organizing Map algorithm and apply it to the kddcup.data_10_percent dataset.

Task1

Task specification: (3 marks)
Describe pre-processing steps that would suitably prepare the data in kddcup.data_10_percent for a processing by a Self-Organizing Map (SOM). It is important that you take into account that the SOM training algorithm requires numerical data and that the algorithm is to operate in Euclidean space (the distance function for finding the best matching unit is the Euclidean distance).

Task 1.1: Explain and justify the pre-processing steps that you would recommend.

Task 1.2: Create a file task1.py which contains the fully functional implementation of all your pre-processing methods (from Task 1.1) in python. Your code should write the pre-processed data to a file called "task1.csv" using a comma delimited CSV format.

Task2

Task specification: (5 marks)

Create a file called task2.py to contain your implementation of the online version of the SOM algorithm in Python 3.4. The objective is that you implement the SOM algorithm yourself using Python. Your code is to read the data file created in Task1, implement the online training algorithm as described in the lecture notes, and offer visualization routines as specified below. One pass through all samples in the training set is called an *iteration*. At every iteration, your code is to print the average quantization error and time (in seconds) taken to the screen. At the end of the network training your code is to create a data file called "mappings.dat" which contains the final mapping and quantization error of each of the training samples. A key challenge in this task is to write code that is efficient, correct, and fully functional. Your code can assume that we will only train 2-dimensional SOMs.

Your code should accept command line parameters as follows:

Argument	Type	Accepted values	Required?	Default value	Explanation
-alpha	float	Values greater than zero	Optional	0.8	Defines the initial learning rate at iteration zero.
-data	string	Any non-empty string	Required	NIL	Defines the name of the data file (CSV file)
-iter	integer	Any positive integer including zero.	Optional	10	Defines the number of training iterations.
-sigma	float	Values greater than zero.	Optional	0.3 times the maximum dimension of the map.	Defines the std.dev. Of the gaussian neighborhood function.
-mapsize	int,int	Two integers each of which must be positive (excluding zero)	Optional	8,16	Defines the size of the (2D-)map of the SOM.

For example, your code when started as follows:

```
python3.4 task2.py –mapsize 12,18 –iter 64 –data task1.csv –alpha 0.9
```

should train a SOM of size 12 x 18 for 64 iterations on the preprocessed kddcup dataset using alpha(0)=0.9, sigma(0)=5.4[1]. At every iteration your code would print the mean quantization error and time taken as in the following example:

```
Iter 0: 2.0354 130.43
Iter 1: 1.8320 128.23
...
Iter 63: 0.3421 131.41
```

at the end of a training session your code would create the file "mappings.dat" which will contain for each sample in the training set one line which gives the final mapping (the coordinates of the BMU at the conclusion of the training session) and the corresponding quantization error as in:

```
9 12 0.0026421
3 17 0.0102460
0 12 0.0000241
....
```

note that the number of lines in "mappings.dat" is equal to the number of training samples. Note also that the n-th line in mappings.dat must correspond to the n-th sample in the input file (i.e. the n-th entry in mappings.dat corresponds to the coordinate of the BMU for the n-th entry in the training set).

Train your som on the data in task1.csv by using parameters -alpha 0.8 -iter 300 -sigma 40 and using a the largest mapsize that would allow your code to complete training within 60 minutes. Use a mapsize that defines an aspect ratio of approximately 1.414 to 1. For example, if you found that the largest SOM that your

1 Since sigma was not specified in this example it will take a default value computed as 0.3 * max(12,18) = 5.4

code could train within 60 minutes (using the parameters as specified before) is 368 then your mapsize would be "-mapsize 23,16" because 23x16=368 and 23/16 is approx 1.414[2]. Report the mapsize you used and meassure the total training time. Also compute and report the mean quantization-error. Also add to your report (the PDF document):

1. Visualize the mappings of the training data by class using the symbol '+' for samples that belong to class normal and using the symbol 'X' for classes that belong to one of the attack classes,
2. visualize the mapping density by using a heatmap plot
3. visualize the umatrix of your map
4. propose, implement, and use a visualization technique that offers a great insight into the property of the data and/or a great insight into the way SOM mapped the data.
5. Fully explain your approach to implementing SOM in python. Fully explain all of the illustrations and observations that can be made from these illustrations. What insights can you obtain from your illustrations (what can you say about the property of the problem domain)?

Task 3

Task specification: (2 marks)

Create a file called task3.py to implement the DBScan algorithm. Use your code to apply DBScan to the mappings in mappings.dat as follows:

1. Read data from mappings.dat
2. Remove the last column from the data in mappings.dat
3. Remove all duplicate entries from the data
4. Apply DBScan (using appropriately chosen parameters for Eps and minPts)
5. Visualize the results using a location plot in which symbols are color coded according to cluster membership (i.e. similar to as shown in the sample Illustration A on page 4.)
6. Document and explain your results.

Marking guidelines:

Code: Your python code will be assessed on a lab computer using python 3.4 under Ubuntu as is available in 3.124. Thus, **the computers in room 3.124 define the standard environment for code development and code execution.** Note that CSCI316 is not a programming subject (it does not teach programming skills) and hence the correctness, completeness, efficiency, and results of your executed code will be assessed. Thus, code that produces no useful outputs will receive zero marks. This also means that code that does not run on a computer in room 3.124 would be awarded zero marks or code where none of the core functions produce correct results would be awarded zero marks.

Your answers in the PDF document: The correctness and completeness of your answers will be assessed.

2 Computed as follows: Given number of neurons x, then ydim=round(sqrt(x/1.414)), xdim=round(x/ydim)

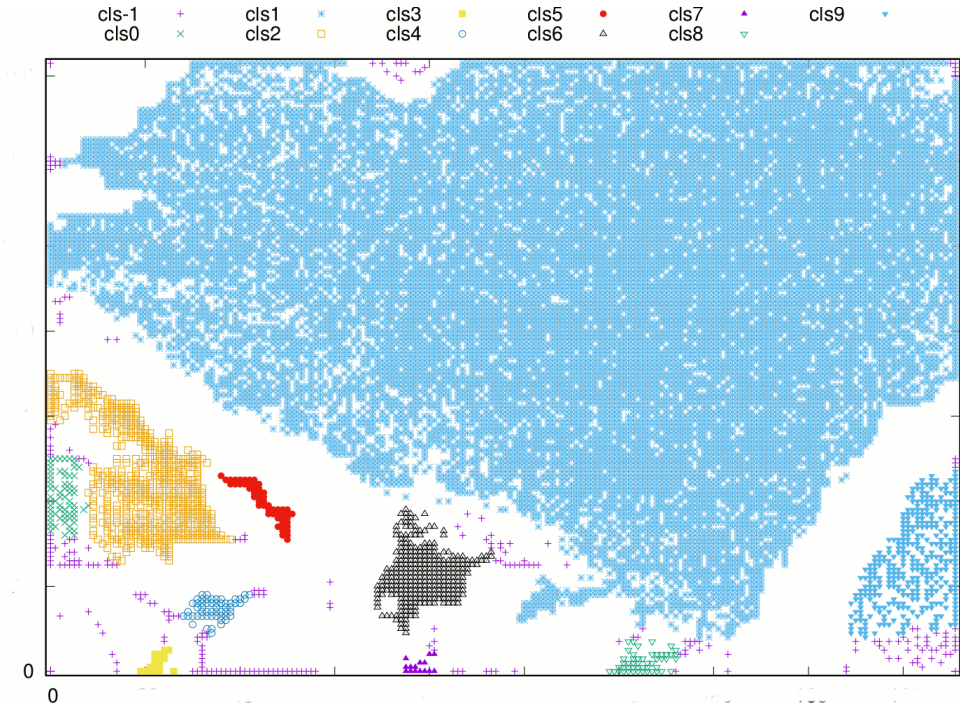

Illustration A: DBScan sclustering result