Instituto Tecnológico y de Estudios Superiores de Monterrey

Monterrey Campus

School of Engineering and Sciences

Master of Science in Computer Science

Advanced Topics on Machine Learning
Assignment 2: Finding the best-two clusters in the Best 200 Universities in the QS World University Rankings 2019

By

Kevin Brian Kwan Chong Loo

ID A0119217

1. Introduction

This assignment is a continuation of the obtainment of a data set containing information of the Top 200 Universities according to the QS World University Ranking of 2019. For this task, binary partitions of the data set were made where given a rank value, one cluster is formed with the Universities with a higher rank and the other with the ones with lower rank. In order to evaluate the clusters, different classifiers were used. An implementation of the VIC algorithm was implemented in Python which consists of evaluating each partition with 10 Fold Cross Validation with different classifiers and obtaining the highest value.

2. Development

In order to achieve this assignment, a set of steps were required to be made. The data set had to be partitioned 50 times, each with a different rank value. Then, the classifiers were used from the "sklearn" library in Python and "h2o4gpu". Eight different classifiers were tested and implemented with the VIC algorithm. Finally, each of the results were compared among each other.

2.1. Partitions

In this task, 50 different partitions were made with the data set. In other words, 50 rank values were chosen to separate the data set into two classes. The fifty partition values chosen were from 75 to 125. Technically, both limits were considered, therefore 51 partitions were made. As an example, in the partition 75, the Universities with a rank lower than 76 (which in this area of focus, a lower value means higher ranking since being #1 is the best), were classified as class 0 and the ones with 76 or higher were classified as 1. For each partition, both the format in "arff" and "csv" were generated in order to be easily read in Python and in Weka.

2.2. VIC Algorithm

The validity index is a performance measurement which evaluates how well the clusters are separated or in a sense, different which implies that there is indeed discrepancies among clusters. For this task, the validity index chosen is VIC. The VIC algorithm was implemented in Python in order to evaluate a set of different classifiers. This algorithm consists of doing Cross Validation with different classifiers and evaluating their performance, in this case, the metric used was ROC Area Under the Curve since for partitions, the classes are somewhat imbalanced. By having all the performance values of each classifier applied to a partition, the highest value is extracted and repeated among all the partitions. In the end, the best partition is determined by the highest validity index which in this case is VIC.

2.3. Python Implementation

The evaluation of the different 51 partitions was implemented in Python with the assistance of the "sklearn" library. This package by itself includes many classifiers and has built in functions to apply it with k-fold cross validation. A default version run is implemented where it applies eight different classifiers and using 10 fold cross validation. The classifiers are Bayesian Networks, Multi-Layer Perceptron, AdaBoost, K-Nearest Neighbor, Random Forest, Support Vector Machines, Naive Bayes and Linear Discriminant Analysis. For this implementation, all the classifiers were available using the “sklearn” library except Bayesian Networks. For this specific classifier, Weka was used with Python.

To run it, the folder path with the partitions in “csv” format must be specified. The code checks that path folder and reads each file no matter how many there are. As an output, it generates a text file where the first line shows the order in which each classifier was used. In the following lines, it presents the file name of the partition and its AUC result per classifier and the highest one among them at the end.

The library of “sklearn” uses CPU by itself and normally it does not do any parallelization, making it somewhat slow. Therefore, an alternative library was also implemented which is “h2o4gpu”. This library is a variation of “sklearn” which contains almost all the classifiers of “sklearn” but with the advantage of using a GPU. For the implemented code, both imports of the libraries were implemented and can be used just by uncommenting them. In the case of “h2o4gpu”, it requires to be run using Python 3.6 specifically.

2.4. Weka Implementation

As mentioned before, not all classifiers are available in “sklearn” and in “h2o4gpu”. In this implementation, Bayesian Networks was implemented with Weka. Likewise, Support Vector Machines was implemented with Weka since the “sklearn” version took a lot of time. For this task, another package for Python was installed which is able to access the classifiers of Weka with an API in Python. In other words, by using Python, calls to Weka were made and it returned the information of interest. The advantage of the Weka Python wrapper installed is that by itself, it does parallelization which uses the available CPU’s efficiently, giving a better performance in execution time.

3. Results

For the execution of the Python implementation, the partitions in both arff and csv format were imported. Before evaluating the partition with the classifiers, the attribute of ranking was discarded due to the fact that this attribute by its nature has a unique value which increments per University. If it was not removed, the classifier would learn to simply differentiate the clusters per its ranking value which in a sense, it’s the partition value. For this reason, it was removed in order to avoid erroneous results.

By running the implementation in Python, with the 8 classifiers, using 10 fold cross validation and in all the partitions, the following results were obtained:

Partition Number	Best ROC-AUC	Partition Number	Best ROC-AUC
75	0.85378	101	0.83676
76	0.83565	102	0.83055
77	0.83400	103	0.81000
78	0.85173	104	0.80200
79	0.84200	105	0.80475
80	0.84479	**106**	**0.78700**
81	0.85500	107	0.79200
82	0.84200	108	0.80400
83	0.85680	109	0.79900
84	**0.86324**	110	0.80505
85	0.84215	111	0.80800
86	0.84356	112	0.78800
87	0.83939	113	0.82037
88	0.82100	114	0.81197
89	0.82200	115	0.80200
90	0.83333	116	0.80418
91	0.82600	117	0.80164
92	0.84980	118	0.81000
93	0.84348	119	0.81300
94	0.82500	120	0.81000
95	0.84616	121	0.80700
96	0.84317	122	0.81600
97	0.82825	123	0.80800
98	0.83763	124	0.81500
99	0.83302	125	0.81000
100	0.82050		

Table 1: VIC Validity Index using AUC implemented for all 51 Partitions.

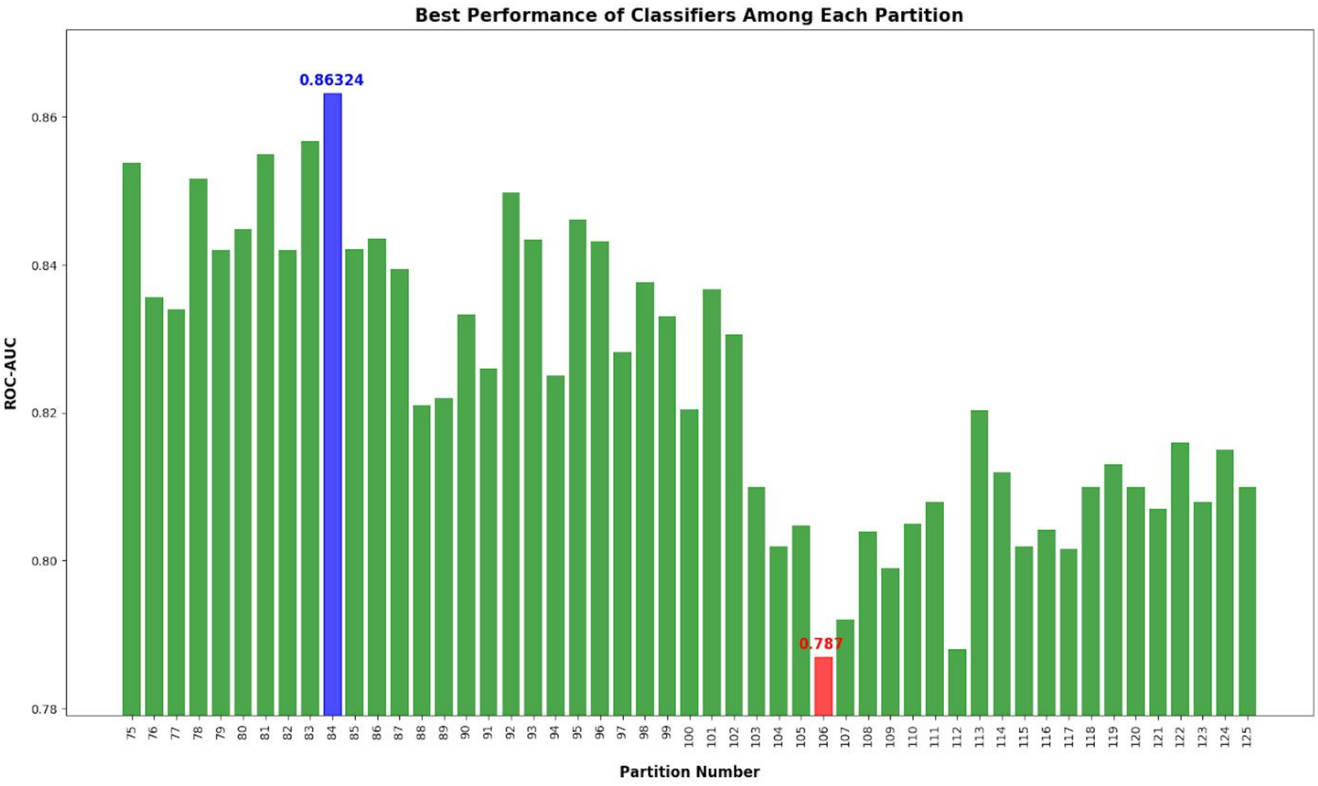

Figure 1: Bar graph of the VIC Validity Index per Partition

Based on the results presented above, it can be seen that the best performance was achieved by the Partition 84 meaning that the Universities with a ranking better than 84 were put in one cluster and the ones with lower ranking than 84 were put in the other cluster. The VIC validity index value for this partition was of 0.86324. As for the lowest performance with the VIC validity index is Partition 106 with a value of 0.787. Frankly, the difference in performance is actually small since it is a difference of 0.076 which could mean that the classifiers presented a similar behaviour among the 51 partitions.

By looking at the bar graph above, it can be seen that there is an apparent positive skewness among the validity index. This could mean that the higher ranking partitions generate better clusters. In a sense, translated to the context, the similarity among higher ranking Universities is greater than the similarity among the lower ranking Universities. This would mean that the characteristics of the best Universities are more similar than the ones of the lower ranking Universities.

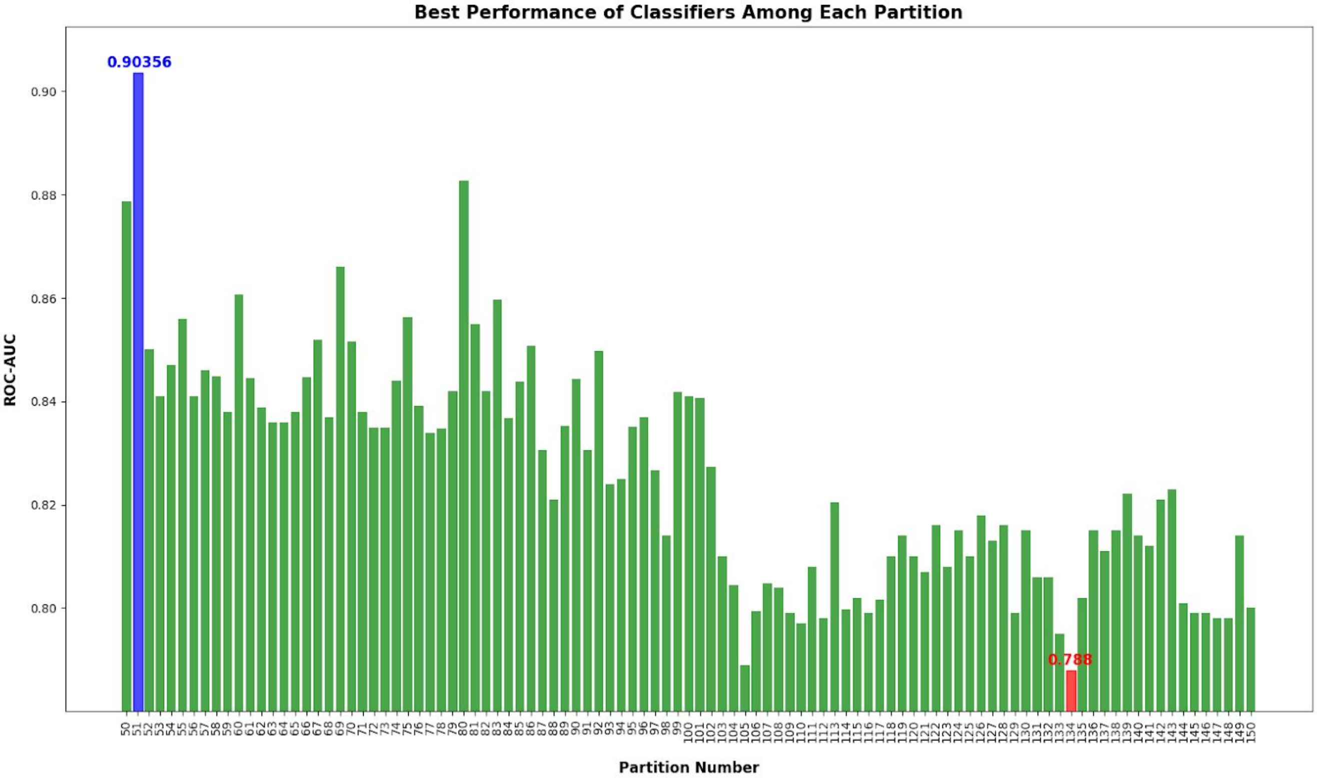

Figure 2: Bar graph of the VIC Validity Index per 101 Partitions.

In Figure 2, it can be seen that it was tested with 101 partitions. In this case, the Partition of rank 51 produced the best validity index. This means that the Top 51 Universities present the most similarity among each other and the rest of the 200 Universities present another behaviour but similar among themselves. In this graph, it can be seen that the positive skewness is still present but a particular behaviour is presented among the 70's and 80's partition rankings. It is clearly shown that the best partition is with rank 51 but the next higher values are among the 70's and 80's and not the partitions right after 51. This would imply that the better partitions are among these ranges. Additionally, it is important to take into account that for this assignment and the VIC algorithm implemented, the cross validation is done once per classifier per partition. A better evaluation should include "k" iterations of the k-fold cross validation. This fault is clearly seen by observing that the next best validity index is partition 81 rather than 84 as seen in the previous results of the 50 partitions.

Partition Number	Best ROC-AUC	Partition Number	Best ROC-AUC	Partition Number	Best ROC-AUC	Partition Number	Best ROC-AUC
50	0.87867	76	0.83100	102	0.81848	128	0.81600
51	**0.90356**	77	0.83791	103	0.81000	129	0.79900
52	0.85000	78	0.83872	104	0.80200	130	0.81500
53	0.84100	79	0.84200	105	0.79113	131	0.80600
54	0.84700	80	0.85255	106	0.79130	132	0.80600
55	0.85600	81	0.85500	107	0.79200	133	0.79500
56	0.84100	82	0.84620	108	0.80400	134	0.78800
57	0.84600	83	0.84980	109	0.79900	135	0.80200
58	0.84492	84	0.85787	110	0.79700	136	0.81500
59	0.83800	85	0.84303	111	0.80800	137	0.81100
60	0.86071	86	0.83551	**112**	**0.78800**	138	0.81500
61	0.83927	87	0.83481	113	0.82037	139	0.82215
62	0.83872	88	0.82731	114	0.79600	140	0.81400
63	0.83600	89	0.83997	115	0.80200	141	0.81200
64	0.83600	90	0.83202	116	0.79900	142	0.82100
65	0.83100	91	0.83076	117	0.80164	143	0.82300
66	0.84400	92	0.84939	118	0.81000	144	0.80100
67	0.85194	93	0.82400	119	0.81300	145	0.79900
68	0.84654	94	0.82840	120	0.81000	146	0.79900
69	0.84019	95	0.83505	121	0.80700	147	0.79800
70	0.85165	96	0.83817	122	0.81600	148	0.79800
71	0.83800	97	0.81903	123	0.80800	149	0.81400
72	0.83823	98	0.81747	124	0.81500	150	0.80000
73	0.83500	99	0.83092	125	0.81000		
74	0.84400	100	0.82135	126	0.81800		
75	0.83900	101	0.83444	127	0.81300		

Table 2: VIC Validity Index with K-Iterations of K-Fold Cross Validation using AUC implemented for all 101 Partitions.

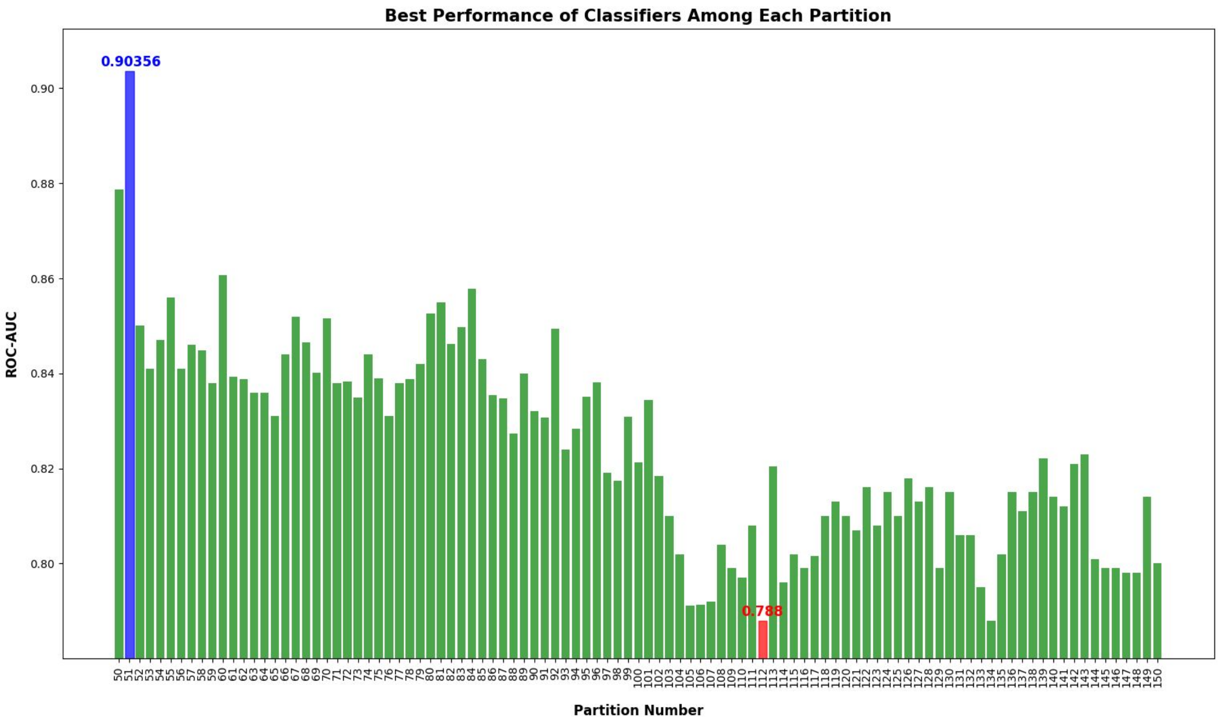

Figure 3: Bar graph of VIC Validity Index with K Iterations of K-Fold per 101 Partitions.

The modification of K iterations of the K-Fold cross validation was also implemented. Table 2 and Figure 3 show the results of this modification of the VIC algorithm. This is in fact the best possible evaluation considering 101 partitions. In this case, partition 51 present the best validity index with 0.90356 and partition 112 with the lowest validity index with 0.788. It can also be seen that for the original 51 partitions, partition 84 has the best performance. However, considering all the 101 partitions, number 51 presents the possible value with a relative large difference compared to the other partitions. This would in fact support the premise that the Top 51 Universities and the Lower 149 Universities present the most similarities or patterns among each other in their own classification clusters.

4. Conclusions

The focus of this assignment was to evaluate the partitions of the clusters. With the conducted experiments it was determined that among the 51 partitions, partition 84 presented the highest validity index with more than 0.857. However, by doing 101 partitions, it could be seen that the best partition rank was 51 with a VIC validity index of more than 0.903. These values were obtained by doing K iterations of the K-Fold Cross Validation per Classifier which is a more robust test result. The fact that partition 51 gave the highest validity index gives insight to how similar the Top 51 Universities are and by knowing that, it could help in determining what improvements need to be applied for the remaining Universities in order to have a higher ranking in the QS World Ranking.

5. Github Repository

https://github.com/Kranok94/VIC-Validity-Index-Python-Weka