Title: College Predictor

Author: Ninad Subhash Ingale

Abstract:

To decide whether to carry on for higher studies is a herculean task. Once a student and their family make up their mind about attending the school another task which remains to be taken care of is to select which set of schools should they apply to in order to maximize their chances of getting in. Also when families are investing so much money and time to this college, it becomes really important to choose a college which would be offering the similar coursework and learning environment suitable for the student in order to maximize the outcome of attending that college. If this analysis is not done students might end up applying to colleges which are either too easy for them to get in to or they might get a lot of rejects and the efforts and the money involved in the application process would be wasted. It would be extremely helpful for all students to have access to a central data repository about the colleges and some data mining algorithm to help them make this decision a better one. This paper tries to find out patterns from the existing data about colleges and then suggest a group of schools a student can apply to. Once a student knows which group of colleges they can apply to, they can research more about the particular college they are interested in. The major accomplishment of this project is to narrow down the research area (related to colleges they want to apply) for students and help them make a better choice.

Introduction:

This paper helps us to compare various approaches used in order to predict which colleges to apply to. This was achieved using clustering algorithms. We used 3 different clustering methods in order to cluster colleges into different groups. We will be discussing more about the clustering methods in the experiments section of this paper. Once the colleges were divided into different clusters, we found out which colleges belonged to which particular group. Using this group assignment, now we have a basis to do some classification using the cluster assigned as the classification variable.

As with almost all data mining project, the biggest challenge in this case was to have clean data. The data set for this project consisted of data for 7805 institutions which had 1729 attributes. Major challenge was to decide which attributes to consider in order to cluster the colleges into different groups. In this paper we will discuss a couple of algorithms which were used in order to clean the data (Please refer section for Experiments). Also some of the attributes were removed because they did not fit the context of this project. We are trying to group the colleges based on factors like student's exam scores , number of people getting the degree in a specified stream (in this particular vase Computer Science) from a college , the admit rate of the college , the cost of attendance. The reason for choosing these factors is these are the things students and their families are first going to think about when applying to a college.

The representation of data in a 3 dimensional space is shown in Figure 1, we have the SAT_AVG scores on X-Axis, Cost of attendance on Y-axis and the admission rate for colleges on Z-axis. Reason for having the data represented as a collection of multiple single lines is because we have created equal (almost) sized bins for the SAT_AVG and Cost of attendance. This was done as a part of data cleaning (Please refer section for Experiments). Also we have discarded the data for colleges which had no SAT_AVG values specified.

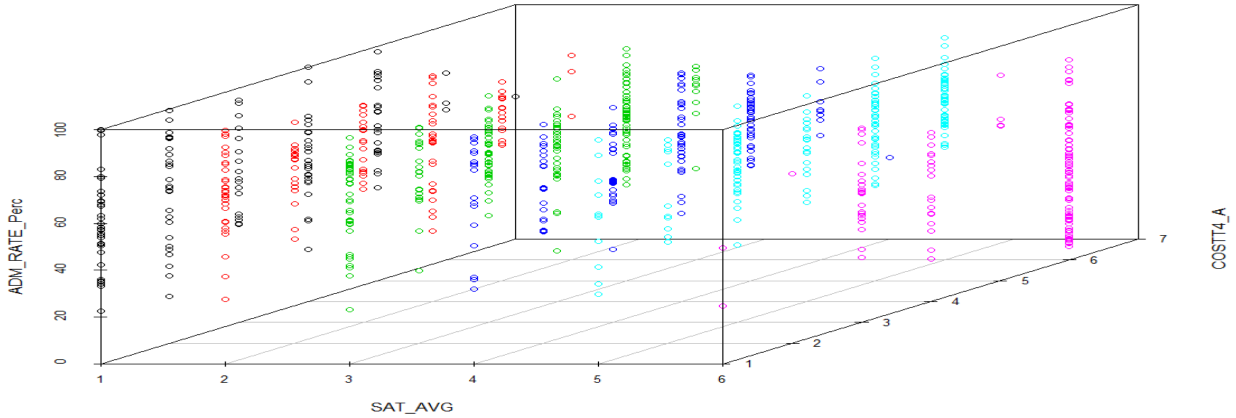


Figure 1: Representation of Data in 3 dimensional space. [3]

We have also removed colleges which had not admission rate mentioned, as that is one of the important factors when considering admission. For the sake of keeping the data specific to the objective of suggesting the graduate schools for student willing to pursue Computer Science, we are only considering the data for colleges related to graduate studies and for the Computer Science department. The same methods can be extended to the entire set of streams and we can get similar results for other streams.

Previous Work:

U.S.Department of Education has put a lot of efforts in the direction to determine the quality of an institution (both Graduate and Undergraduate) on the basis of certain domains such as earnings, completion, cost, debt and repayment of the educational loans and academic [2]. Researchers and federal and state policy makers are really interested in assessing the quality of the educational institution because it would be very helpful if we learn a pattern and then allocate resources accordingly to the institutions in order to maximize the student benefits. Also learning these patterns will help institutions make any changes to their strategy in future in order to improve the institutional quality and in turn increase the number of prospective applicants to their institution. The availability of this central data repository will enable prospective students and their parents take a calculated decision about their future education plan and also might be helpful in predicting the outcome of that education. Of course this outcome will also depend upon student’s ability to complete the course for which he has enrolled.

The dataset available [4] is broadly categorized into 5 domains earnings, completion, cost, debt and repayment of the educational loans and academic as mentioned above [2]. Earnings domain gives us an overview of how students who graduated from an institution in past are doing financially currently. Completion domain helps us understand more about the status of completion of the degree by students attending the institution. Debt and repayment domain gives us information about the average time taken by the alumni of that particular institution by degree earned to repay the educational loan. Cost domain helps to compare cost of attending different institutions of same score (based on the domains mentioned above). Last but not the least, academic domain helps us find patterns of the students getting successfully admitted in the institution degree wise.

Challenges Faced in the implementation of the above mentioned solution:

One of the major data limitations mentioned is, we have the data available for students from National Student Load Data System (NSLDS) [5]. The dataset offered in this case contains undergraduate students who received financial aid. So here we are missing a considerable portion of data which contains student who have not received any financial aid but are still enrolled into the institution.

Another issue mentioned with the data is the institutions being covered and portion of students studying in them [2]. This dataset contains public and private schools offering 2 or 4 year courses. Students studying in public institution enrolled in 4 year courses constitute of little over one third of total number of students [2] but the number of institutions offering these courses are only 10 percent of the total institutions.

One of the major data limitations is, we have the data available for students from National Student Load Data System (NSLDS) [5]. The dataset offered in this case contains undergraduate students who received financial aid. So here we are missing a considerable portion of data which contains student who have not received any financial aid but are still enrolled into the institution.

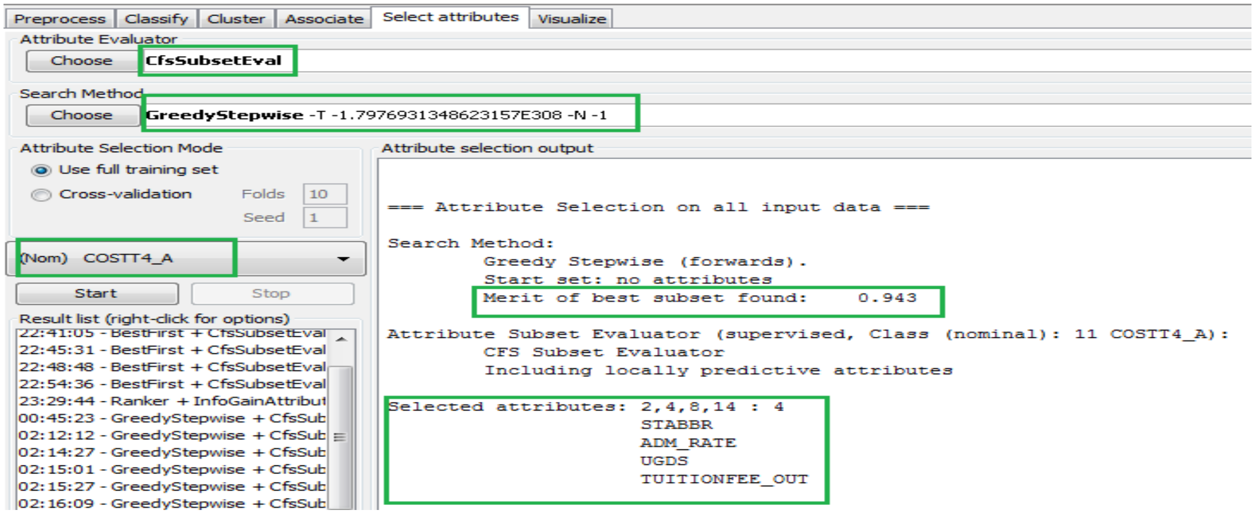
Another issue faced with the data is the institutions being covered and portion of students studying in them [2]. This dataset contains public and private schools offering 2 or 4 year courses. Students studying in public institution enrolled in 4 year courses constitute of little over one third of total number of students [1] but the number of institutions offering these courses are only 10 percent of the total institutions.

Experiments:

Data Cleaning Methods Used:

Attribute Selection

We have used Weka and R tools for data cleaning purpose. In weka we have used *CfSubsetEval* [1] algorithm in order to determine which attributes are the most important ones to use. This method was used because of the large number of attributes to select from. The basis on which this method works is it evaluates different sets of attributes and calculates the set of attributes which give us the maximum merit.



Attribute Evaluator: CfsSubsetEval

Search Method: GreedyStepwise -T -1.7976931348623157E308 -N -1

Attribute Selection Mode:
● Use full training set
○ Cross-validation Folds 10 Seed 1

(Nom) COSTT4_A

Result list (right-click for options)

```
22:41:05 - BestFirst + CfsSubsetEval
22:45:31 - BestFirst + CfsSubsetEval
22:48:48 - BestFirst + CfsSubsetEval
22:54:36 - BestFirst + CfsSubsetEval
23:29:44 - Ranker + InfoGainAttribut
00:45:23 - GreedyStepwise + CfsSub
02:12:12 - GreedyStepwise + CfsSub
02:14:27 - GreedyStepwise + CfsSub
02:15:01 - GreedyStepwise + CfsSub
02:15:27 - GreedyStepwise + CfsSub
02:16:09 - GreedyStepwise + CfsSub
```

Attribute selection output

```
=== Attribute Selection on all input data ===

Search Method:
        Greedy Stepwise (forwards).
        Start set: no attributes
        Merit of best subset found:    0.943

Attribute Subset Evaluator (supervised, Class (nominal): 11 COSTT4_A):
        CFS Subset Evaluator
        Including locally predictive attributes

Selected attributes: 2,4,8,14 : 4
                     STABBR
                     ADM_RATE
                     UGDS
                     TUITIONFEE_OUT
```

Figure 2: Attribute selection using cfsetEval and GreedyStepwise

CfSubsetEval method starts with no attributes and builds the attribute set at each step in order to provide the set with most merit [1]. It can also proceed in backward direction (start with all reduce to some).We also used *InfoGainAttributeEval* method from Weka for attribute selection and it produced similar results. Reason for choosing *CfSubsetEval* over *InfoGainAttributeEval* was the number of attributes to be handled. The above attributes are not the only one which are considered for forming the clusters.

Binning

In order to create equal (almost) sized bins, following binning criteria was used

SAT_AVG		Cost Of Attendance	
Actual Values	Bin	Actual Values	Bin
759-950	1	7000-18000	1
950-1000	2	18000-22000	2
1000-1050	3	22000-25000	3
1050-1080	4	25000-35000	4
1080-1200	5	35000-45000	5
>1200	6	>45000	6

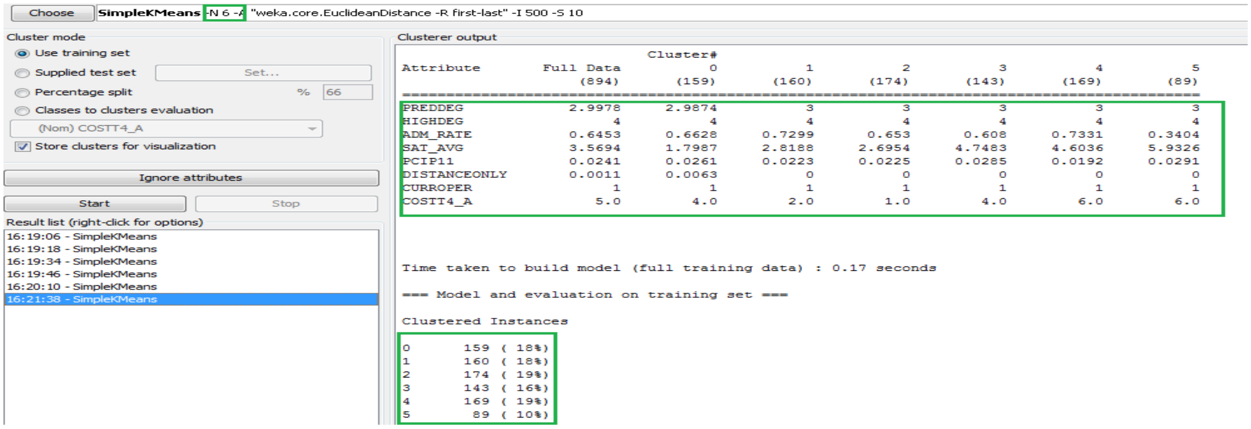
Clustering

K-means Algorithm:

When it comes to choosing clustering algorithm, the first choice is K-means algorithm as it is easy and effective [6][7]. It represents the centroid of the cluster as the central tendency of the entire group. K-means clustering uses the Sum of Squared error as the measure of bestness.The goal is to minimize this sum of squared error [6]. It is an iterative method of clustering where each time a centroid is calculated (shifted) and we stop when the centroid of the group stops moving. This is the point where the sum of squared error is minimum. The disadvantage of this clustering method is that the sum of squared error is calculated locally and not globally i.e. we can group colleges into completely different clusters if we choose different initial centroids every time we run K-means. In our case, below is the sum of squared error statistics

Number of Clusters 2	686
Number of Clusters 3	589
Number of Clusters 4	420
Number of Clusters 5	411
Number of Clusters 6	395

From the table we should select k=4, but for k=6 the data is more evenly distributed.



```
SimpleKMeans -N 6 -A "weka.core.EuclideanDistance -R first-last" -I 500 -S 10
```

Cluster mode

- ● Use training set
- ○ Supplied test set
- ○ Percentage split % 66
- ○ Classes to clusters evaluation (Nom) COSTT4_A
- ☑ Store clusters for visualization

Result list (right-click for options)

16:19:06 - SimpleKMeans
16:19:18 - SimpleKMeans
16:19:34 - SimpleKMeans
16:19:46 - SimpleKMeans
16:20:10 - SimpleKMeans
16:21:38 - SimpleKMeans

Clusterer output

```
                               Cluster#
Attribute          Full Data          0          1          2          3          4          5
                       (894)      (159)      (160)      (174)      (143)      (169)       (89)
==============================================================================================
PREDDEG               2.9978     2.9874          3          3          3          3          3
HIGHDEG                    4          4          4          4          4          4          4
ADM_RATE              0.6453     0.6628     0.7299      0.653      0.608     0.7331     0.3404
SAT_AVG               3.5694     1.7987     2.8188     2.6954     4.7483     4.6036     5.9326
PCIP11                0.0241     0.0261     0.0223     0.0225     0.0285     0.0192     0.0291
DISTANCEONLY          0.0011     0.0063          0          0          0          0          0
CURROPER                   1          1          1          1          1          1          1
COSTT4_A                 5.0        4.0        2.0        1.0        4.0        6.0        6.0



Time taken to build model (full training data) : 0.17 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       159 ( 18%)
1       160 ( 18%)
2       174 ( 19%)
3       143 ( 16%)
4       169 ( 19%)
5        89 ( 10%)
```

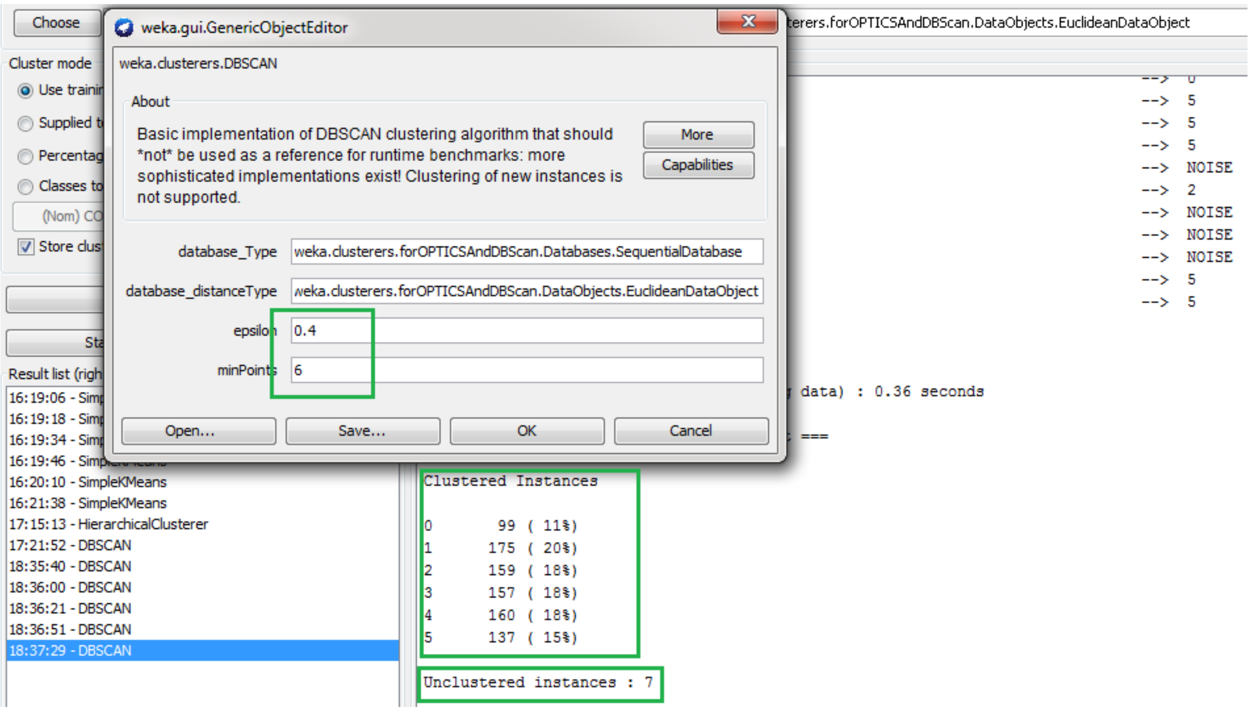
Figure 3 Uniform distribution of data into clusters

DBScan:

DBScan is a density based clustering algorithm [8]. It finds the number of clusters based on the density of the points located in the data. The clusters are formed on the basis of epsilon distance from the central point (arbitrarily chosen at first) and then as and when the iterations proceed, the cluster centroid is shifted. Below is the statistics about the noise points and the clusters formed for each epsilon distance.

Epsilon distance	Number of clusters formed	Number of Min points in a cluster	Unclustered points
0.2	31	6	29
0.3	6	6	10
0.4	6	6	7

From the above table, we can say that as epsilon distance crosses 0.2, the number of clusters formed is reduced drastically and it does not change from that point onwards. Hence for this particular dataset the value of epsilon to be chosen should be 0.3 to 0.4. Below is the specification from Weka for the same.



weka.gui.GenericObjectEditor

weka.clusterers.DBSCAN

About

Basic implementation of DBSCAN clustering algorithm that should *not* be used as a reference for runtime benchmarks: more sophisticated implementations exist! Clustering of new instances is not supported.

database_Type: weka.clusterers.forOPTICSAndDBScan.Databases.SequentialDatabase

database_distanceType: weka.clusterers.forOPTICSAndDBScan.DataObjects.EuclideanDataObject

epsilon: 0.4

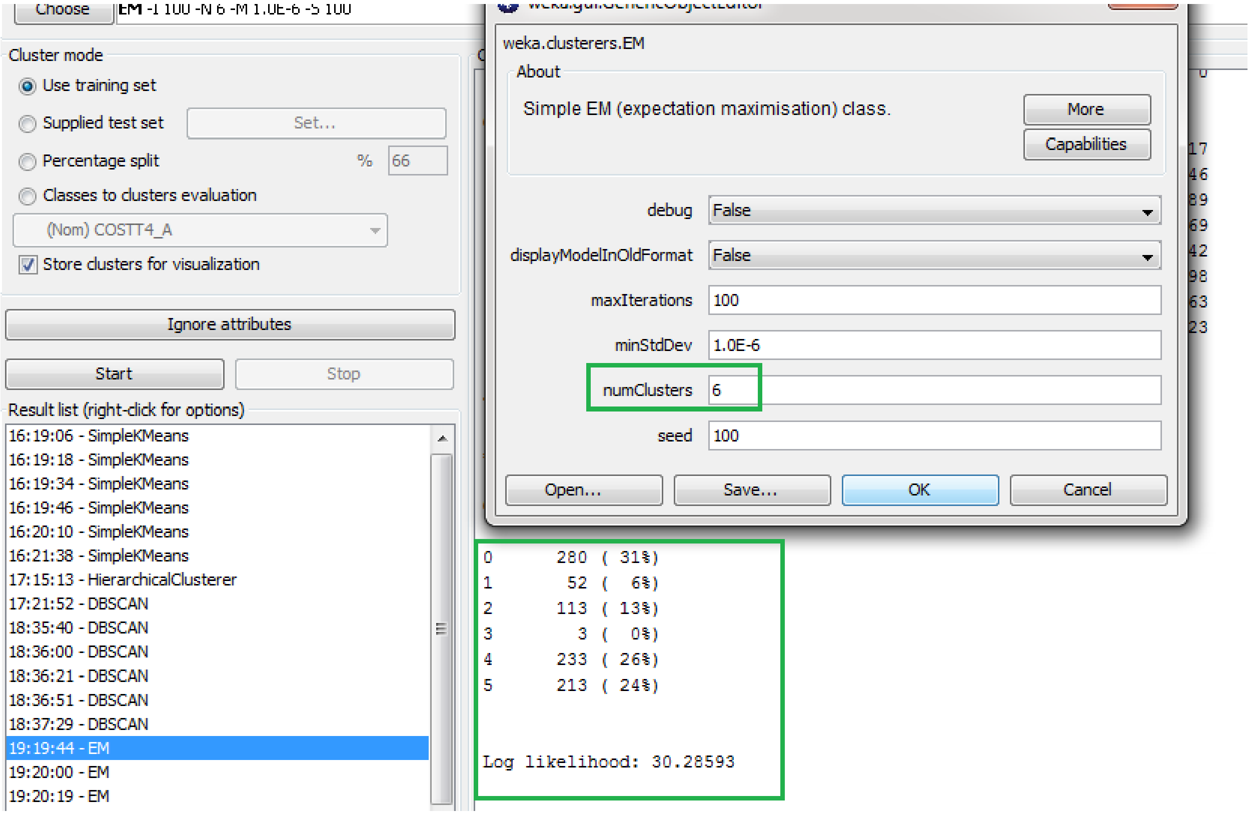
minPoints: 6

```
-->  5
-->  5
-->  5
-->  NOISE
-->  2
-->  NOISE
-->  NOISE
-->  NOISE
-->  5
-->  5

data) : 0.36 seconds

===

Clustered Instances

0       99 ( 11%)
1      175 ( 20%)
2      159 ( 18%)
3      157 ( 18%)
4      160 ( 18%)
5      137 ( 15%)

Unclustered instances : 7
```

Figure 4: DBScan algorithm specifications in Weka

Expectation Maximization:

Expectation maximization is a probabilistic algorithm which works on the basis of likelihood of the point being present in particular cluster [9]. This algorithm is generally used when we are not satisfied with the results from Kmeans algorithm. Below is the implementation of EM algorithm in Weka.



EM -I 100 -N 6 -M 1.0E-6 -S 100

Cluster mode

- ● Use training set
- ○ Supplied test set
- ○ Percentage split % 66
- ○ Classes to clusters evaluation
- (Nom) COSTT4_A
- ☑ Store clusters for visualization

Result list (right-click for options)

```
16:19:06 - SimpleKMeans
16:19:18 - SimpleKMeans
16:19:34 - SimpleKMeans
16:19:46 - SimpleKMeans
16:20:10 - SimpleKMeans
16:21:38 - SimpleKMeans
17:15:13 - HierarchicalClusterer
17:21:52 - DBSCAN
18:35:40 - DBSCAN
18:36:00 - DBSCAN
18:36:21 - DBSCAN
18:36:51 - DBSCAN
18:37:29 - DBSCAN
19:19:44 - EM
19:20:00 - EM
19:20:19 - EM
```

weka.clusterers.EM

About

Simple EM (expectation maximisation) class.

debug	False
displayModelInOldFormat	False
maxIterations	100
minStdDev	1.0E-6
numClusters	6
seed	100

```
0      280 ( 31%)
1       52 (  6%)
2      113 ( 13%)
3        3 (  0%)
4      233 ( 26%)
5      213 ( 24%)


Log likelihood: 30.28593
```

Figure 5: EM implementation in Weka

Decision Tree:

Once we have the colleges divided into cluster, we can form a decision tree on the basis of cluster assignment by one of the algorithms used above. After the tree is formed we can understand which were the most important attributes used (from the information gain) and much to my surprise, the cost of attendance had less information gain than the Admission rate of the college specified. We observed, the attribute which had the maximum information gain (root node of the tree) is as expected SAT score required by the college in order to be admitted to that particular college. From this tree it would be very easy for students to get to know which group of universities to target and maximize their chances of getting more number of admits.

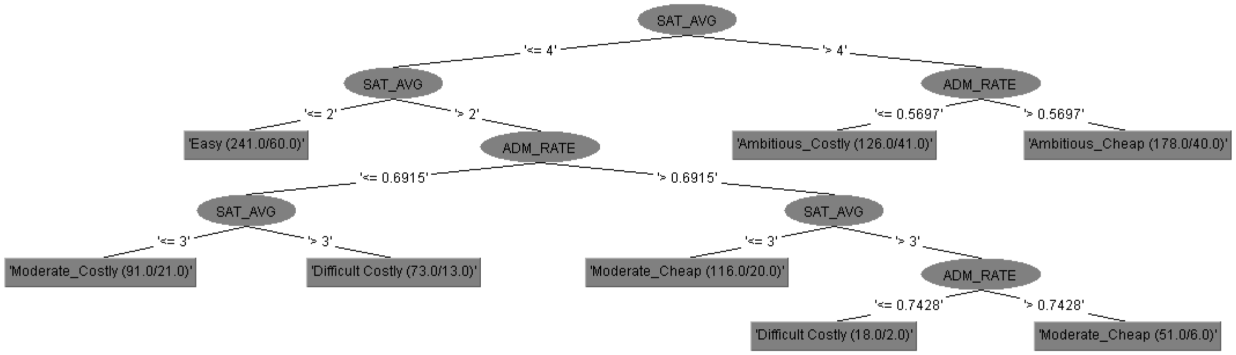

SAT_AVG
'<= 4'
'> 4'
SAT_AVG
ADM_RATE
'<= 2'
'> 2'
'<= 0.5697'
'> 0.5697'
'Easy (241.0/60.0)'
ADM_RATE
'Ambitious_Costly (126.0/41.0)'
'Ambitious_Cheap (178.0/40.0)'
'<= 0.6915'
'> 0.6915'
SAT_AVG
SAT_AVG
'<= 3'
'> 3'
'<= 3'
'> 3'
'Moderate_Costly (91.0/21.0)'
'Difficult Costly (73.0/13.0)'
'Moderate_Cheap (116.0/20.0)'
ADM_RATE
'<= 0.7428'
'> 0.7428'
'Difficult Costly (18.0/2.0)'
'Moderate_Cheap (51.0/6.0)'

Figure 6: Decision tree to help students determine group of universities to apply to.

Discussion:

In this paper we have tried to cluster colleges based on various features of students and colleges. For this purpose we have used 3 clustering algorithms. The most easy and effective to use is Kmeans algorithm which makes task of clustering easy. The only disadvantage of this algorithm is that the cluster centroids calculated are local and not global, hence if we choose different initial seed points, the results are bound to be different. Next algorithm used is DBScan which is density based iterative algorithm. This clusters points on the basis of density of points surrounding them. This algorithm gave us a result of 6 clusters as after the 6 clusters, centroid was not moving much and we had included almost all points from the data (~10 noise points). Next algorithm used was Expectation Maximization which is a probabilistic clustering algorithm. This algorithm is used when we are not satisfied by the results from Kmeans algorithm. This algorithm is complex to use and is used when we cannot form a simple boundary between the clusters. In our case that is not the scenario hence the first two algorithms used give us the best results.

This paper gives us insights about the data that SAT_AVG,ADM_RATE are the most important columns but we have also used the number of people receiving degree from that college , cost of attendance , distanceonly as the other important parameters in order to form clusters. Surprisingly ADM_RATE is even more important than the cost of attendance for any college from this dataset.

Future Work:

From this paper we have identified the group of colleges a particular student can apply to. This solution only caters to graduate students aspiring to study Computer Science in their higher education. This project can be extended to the undergraduate students as well. It needs to be developed for all the branches. Also this solution gives a student a group of universities but lacks to give those top 10 or top 20 from each and every group.

Conclusion:

In this paper we studied 3 clustering algorithms in order to cluster colleges based on certain features. The major takeaway from this project was, importance of attribute selection when it comes to a clustering and classification problem. It is extremely important to select correct attributes which give us maximum information before even thinking of which clustering algorithm to use. As the original dataset was regarding the analysis of quality of colleges, it contained a lot of attributes related to the repayment of loan and salary people were getting after their graduation. This did not consider the factor of capability of student to complete the course successfully.

After attribute selection, I learnt various algorithms to cluster colleges into different groups and compare the results of each of the clustering algorithms. This exercise gave me an insight about functionality of different clustering algorithms and the pros and cons of using each one in this case. Kmeans and DBScan work best in this case.

References:

[1] Tutorial on Machine Learning . Part 2 Descriptor Selection Bias

http://infochim.u-strasbg.fr/CS3/program/Tutorials/Tutorial2b.pdf

[2]

USING FEDERAL DATA TO MEASURE AND IMPROVE THE PERFORMANCE OF U.S. INSTITUTIONS OF HIGHER EDUCATION

https://collegescorecard.ed.gov/assets/UsingFederalDataToMeasureAndImprovePerformance.pdf

[3] Scatterplots

http://www.statmethods.net/graphs/scatterplot.html

[4] College Scorecard Data

https://collegescorecard.ed.gov/data/

[5]Federal Student Aid

https://www.nslds.ed.gov/nslds/nslds_SA/

[6] Perera D, Kay J, Koprinska I, Yacef K, Zaiane OR. Clustering and Sequential Pattern Mining of Online Collaborative Learning Data. IEEE Transactions on Knowledge and Data Engineering. 2009;21(6):759-72.

http://ieeexplore.ieee.org.ezproxy.rit.edu/xpls/abs_all.jsp?arnumber=4564464

[7] Witten IH, Hall MA, EBSCO Publishing (Firm). Data mining: practical machine learning tools and techniques. 3rd ed. Burlington, MA: Morgan Kaufmann; 2011.

http://web.b.ebscohost.com.ezproxy.rit.edu/ehost/detail/detail?sid=e4ee0719-ff0d-499a-9a4b-42a80afa4589%40sessionmgr103&vid=0&hid=106&bdata=JnNpdGU9ZWhvc3QtbGl2ZQ%3d%3d&preview=false#AN=351343&db=nlebk

[8] Sharma N., Bajpai A., Litoriya R. Comparison the various clustering algorithms of weka tools.

ISSN 2250-2459, Volume 2, Issue 5, May 2012

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.413.5080&rep=rep1&type=pdf

[9] Sehgal G., Dr. Garg K. Comparison the various clustering algorithms.

ISSN 0975-9646, (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (3), 2014, 3074-3076

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.661.4971&rep=rep1&type=pdf