College Recommender system using student' preferences/voting: A system development with empirical study

Mr. Y. Subba Reddy and Prof. P. Govindarajulu

Department of Computer Science
S.V. University, Tirupati, AP, INDIA

Abstract
Recommender Systems are an ever trending research that can be applied in various domains. The college recommendation systems for undergraduate students are a challenging area that needs to be explored thoroughly. A college recommendation system provides the means to undergraduate students in their college selection process with a good number of suggestions. In this paper an effective weighted clustering process WCLUSTER is implemented using R-tree data structure. Instead of traditional data clustering approaches, an improved approach using top-k queries is applied for clustering the college data, based on students' preferences/voting. A new technique was proposed for finding similarity measures between objects by using both values of attributes and their corresponding voting / preferences / ratings for attributes. Traditional methods use distance measures for finding similarity between objects. Proposed method uses voting / preferences / ratings for finding similarity between objects by using top-k query ranking of objects. The preferences were obtained through a well structured questionnaire using which the responses from college students were gathered. Based on the sets of responses as preferences the proposed algorithm was executed. To speed up the query execution process a multidimensional indexing structure called R-Tree was used. Pruning techniques were applied for scalability purpose.
This paper introduced a recommendation system for college/course selection. The experimental results showed that applying WCLUSTER in this domain is superior to traditional and previous approaches.

Key words:
recommender system, voting, weighted cluster, top-k query, reverse top-k query, multi-dimensional index tree

1. Introduction

There are more than 750 engineering colleges in Andhra Pradesh and Telangana admission to which is based on web counseling. It is a challenging thing to students to opt a right college to join. There are many things to consider while deciding a college for admission. Infrastructure, faculty, facilities, placements and other related things of a college influence the admission decision. Students have their own preferences while joining a college. They need the list of colleges that meet their preferences. As the number of colleges available in the state is big, students required to analyze and get the information needed for their decision making. It is a very tiresome job for a student to exercise the college profile list (like the national and regional rank of the college, placements, pass percentage, staff quality, infrastructure and particularly the fee details). There is a need of a system that considers and analyzes the profile attributes of a college along with the preferences of the students towards the college profile attributes.

2. Recommendation systems

2.1 The evolution of Recommendation systems

The recommendation systems collect the information regarding items. They gather preferences and profiles and analyze the same to advise the user to make right decisions regarding products, people, policies, and services Subba Reddy.Y and Prof. P. Govindarajulu, [19]. As day after day, the availability of electronic and web content is growing fast, researchers are relying more on content to extract the vital information for better recommendations. So, recommendation systems became popular in assisting numerous decision-making contexts.

The basic idea of recommender systems is to utilize sources of web content about customers and to infer customer interests C.C. Aggarwal [3]. Here the user is an entity to which the recommendation is provided, and an item is a product/service being is recommended. Recommendation analysis predicts the future preferences by analyzing the previous interaction between users and items because past behaviors are often good indicators of future choices.

It is the toughest job to design and implement a large-scale online service that can find what is to be recommended to the customers based on the past purchase history. For example, Amazon gives product recommendation, yahoo makes web page recommendation. The process of constructing an efficient and effective recommendations system is a challenging task. The underlying reason is the

Manuscript received January 5, 2018
Manuscript revised January 20, 2018

large size of the product (object) space and context space. The main goal of recommender systems is to assist its users in finding their preferred objects from the large set of available objects. The voting of a particular customer on a particular object is learned through a random payoff and this payoff is received by the recommender system based on the response details of the customer to the recommendation system. For example, in a course recommendation, the payoffs are the ratings on the scale of 1 to 10, where the ratings are given by the students. In the case of web page recommendations, the payoffs are counted by customers 'clicks, where the Boolean value 1 denotes a click and 0 denotes no-click.

It is trivial that web mining is an important technique for finding the frequent data patterns from the Internet, data warehouse, data mart, and data set and so on. World Wide Web (www) is a powerful platform and it is considered to be the ultimate provider of information super high way used to store and retrieve information and also to mine useful knowledge and then use the same for predicting the interests /requirements of customers. Web data size is huge, unstructured and dynamic in nature. Hence, recommendation systems are the potentially desired information systems used for predicting the feature values according to the requirement of the customer. Web recommendation information systems are very useful for navigating through web pages and getting the desired information quickly.

Nowadays recommendation systems are popular and they try to suggest different types of items to different users. The items may be books, chairs, tables, pens, movies, music, washing machines, computers, printers, plotters and so on. For example, Amazon.com recommends various items to various users based on the knowledge – previously visited, purchased, ordered, enquired, referred, booked and so on.

Zhibo Wang, et al. [23] proposed a unique similarity based metric to find the similarity details of users in terms of their lifestyles and they have constructed a Friend book system to recommend friends based on their lifestyles.

Recommendation systems have developed in parallel with the web technology J. Bobadilla et al. [15]. At the initial time of their existence, they were based on demographic, content-based and collaborative filtering. Now they are in a position to incorporate social information also. A knowledge-based recommendation system considers user-centric requirements rather than his/her past history in order to make recommendations.

Hector Nunez, et al. [12] discussed the comparison of different similarity measures for improving the classification process. Authors said that automatic knowledge acquisition and management methods are needed to build consistent, robust, reliable, fault-tolerant, and effective decision support systems.

2.2 Recommendation systems for college selection

Fazeli Soude, et al. [7] said that recommender systems are being used (have been using) in many real-world applications such as e-commerce based applications – Amazon and eBay. Recommender systems must be accurate and useful to as many numbers of users as possible. The fundamental goal of the educational recommender systems is to satisfy many quality features such as accuracy, usefulness, effectiveness, novelty, completeness, and diversity. Recommender systems must satisfy user-centric requirements. User-centric based recommender systems are more useful than data-centric recommender systems.

Recommender systems were developed for various domains associated with daily life of people such as product recommendation, service recommendations, and people recommendations and so on. This kind of recommendations increases both user convenience and purchase transactions of products and/or services. Course/college recommendation for students is a challenging domain that has not reached the target community thoroughly.

Since there are many options for colleges/courses students have to spend a lot of time for exploring the details and they may not do it in a proper way. Students need a system that accepts the students' preferences and recommends the right college/course. college selection is one of the issues that the students' community tends to solve. Recommender systems help the students decide in what college they should study. The methods existing for the recommendation are content-based filtering, collaborative filtering, and rule mining approaches. Content-based filtering approach recommends an item to a user by clustering the items and the user pairs into groups. This clustering is used to gain similarity between user and item. Personal information of the user is not considered here. Queen Esther Booker creates a prototype of a system for course recommendations [18]. The system accepts user requirements as keywords and recommends courses for students.

Collaborative filtering (CF) approach recommends an item to a user by grouping similar users based on user profiles and predicts the user interests towards the items. Hana introduces a system based on CF approach to recommend courses for a student by analyzing and matching the student's academic records [11]. Then the system analyses

and recommends a course that meets the student's profile. Elham S.Khorasani et al. proposed a Collaborative Filtering model based on Markov Chain to recommend courses based on historical data [7].

Rule mining approach focuses on recommending a series of items to a user by discovering the association rules. Itmazi and Megias developed a recommendation system based on rule mining to recommend learning objects [14].

3. The need for improvements in college recommendation systems

Majority of the students make mistakes in their preference list due to lack of knowledge, inappropriate and inaccurate analysis of colleges and anxious predictions. Hence, they become unhappy and repent after admission. An automated system to do all the work will help the students a lot. Today there are no such systems that consider the student preferences and recommend the right alternatives. A few systems are there in the field that can make predictions based on the rank obtained by the students. Therefore an improved intelligent system is needed to assist students in their college selection process which considers the college profile attributes and the students' preferences for each of the attributes. This weighted approach can provide better information by an efficient grouping of related items (colleges). Using this weighted groups of colleges with related profile attributes, one can suggest a better list of colleges that meet the preferences given by the students.

4. Weighted clustering with r-tree and top-k queries

4.1 Top–k Queries

Customer voting/ preferences play a major role in market data analysis. The database is the backbone of any modern organization. Different types of queries are available for effective database operations. Almost all business tasks need the results of different types of queries such as k–nearest neighbor query, range query, aggregate query, outlier detection query, group query, top–k query, reverse top-k query and so on. The query called top–k query is the one important database query that can be used for finding ranks of database objects. Top–k queries are frequently used in the database and information retrieval systems and applications. Top–k queries retrieve k-most objects from the given set of objects by using a linear score function and customer preferences. The main purpose of top-k query is to find ranking details of objects based on the score function value which is based on voting/ preferences to value of attributes. Score function is a linear function that gives sum of the products of attribute values and their corresponding voting/ preferences. Mathematically the linear score function is denoted as

$$f(object) = \sum_{i=1}^{n} voting(i) * object(i)$$

Here n is the number of attributes of the object and i runs from 1 to n. Voting (i) represents preference value of i^{th} attribute , object(i) represents a value of the i^{th} attribute. All the objects are represented in the multidimensional space. Different data sets needed to represent these computations are O, C and V where O is the set of objects, C is the set of customers and V is the set of voting/preferences of customers. $O = \{O_{ij}\}$, $C = \{C_{ij}\}$ and V = {Vij}, where, i represents rows and j represents columns.

HristidisVagelis, et al. [13] said that database systems cannot efficiently produce the top results of a given preference query because of the reason that they need to test and evaluate the special weight function over all the tuples of the selected relation whereas the developed PREFER system answers preference queries efficiently and effectively by using special materialized views that have been pre-processed and stored.

4.2 Reverse top–k Queries

The reverse top-k query is directly associated with the top-k query. Top–k queries retrieve k–number of products whereas reverse top–k queries retrieve customers who preferred their desired products to the corresponding top–k result sets. Top–k queries are frequently used in the database and information retrieval systems and applications. Top–k queries retrieve k-most objects from the given set of objects by using a linear score function and customer preferences. The main advantage of the reverse top–k query is that it identifies sets of products influenced by various customers and the influence of the reverse top-k set is defined as the cardinality of the reverse top–k result set. With the help of reverse top–k query it is possible to find influence details of products and it is used in market data analysis. The reverse top-k result is directly related to the number of customers who prefer or value a particular product. Many top–k queries are consolidated into one reverse top-k query. That is, there exist one-to-many relationships from reverse top-k query to top-k queries.

Akrivi Vlachou, et al. [1] said that finding the most influential database tuples from a given database of tuples is very useful in real-world applications such as market

data analysis and decision making. Authors proposed two algorithms for finding most influential database objects. The first one uses properties of the sky-band (SB) set for limiting the maximum number of resultant candidate objects and the second one follows branch and bound (BB) algorithm paradigm and it uses upper bound on influence score

Many techniques are available for evaluating reverse top-k queries but only thing is that they are costly in terms of overhead and hence they require significant processing which results in the execution of multiple top-k queries for finding the total number of customers who prefer the queried object. The reverse top-k query produces sets of customers based on object preferences. These sets represent a number of customers who prefer to include the object in their favorite lists. The reverse top-k query is one type of tool for estimating impact or demand of the object in the market.

Vlachou Akrivi et al. [21] proposed a reverse top-k query with two versions – monochromatic and bichromatic reverse top-k queries. Authors proposed an efficient threshold based algorithm for finding bichromatic reverse top-k queries.

Amit Singh, et al. [2] proposed an approximate solution to answer reverse nearest neighbor queries in high dimensional spaces. Authors said that the approach is mainly based on a feature called strong co-relation between k-nearest neighbor (k-NN) and reverse the nearest neighbor (RNN) in connection with Boolean range query (BRQ).

Note that the performance of the reverse top k-query mainly depends on the number top-k query execution for each object and top k-query execution in turn depends on voting/ preferences of customers. Reverse top-k query retrieves the set of customers to whom the object belongs to their top-k result sets. Reverse top-k sets are frequently used for finding the potential demand of the objects in the market. Reverse top-k query executions are costly. Hence there is a need for approximate reverse top-k query executions both for increased scalability and for speedup of the overall execution. Also, effective planning techniques are required. The performance of the R- tree index Data Structure decreases as the dimensionality of the data sets increases and the performance of all the algorithms that are based on R-tree will deteriorate. In such cases, alternative efficiency and effective indexing techniques and algorithms are needed.

Elke Achtert, et al. [6] said that all the existing generalized reverse k-nearest neighbor (RkNN) search methods are only applicable to Euclidian distances but not for general metric objects. As a result, authors proposed first approach for efficient reverse k-nearest neighbor search in arbitrary metric spaces (RkNNSAMS) and k value will be given at query run time.

4.3 R-tree

R-tree is a multidimensional indexing tree data structure. R-tree is a most popular, frequently used, multidimensional, height-balanced special indexing tree data structure and very useful for efficient management of very large training datasets particularly in many real-time applications involving data critical operations. Multidimensional R-tree indexing data structure is very useful and efficient for customer voting based similarity the data structure. In customer voting based similarity data search R-tree multidimensional indexing Data Structure is used with slight modifications and a finite set of constants applied on the bounds similarity values of the query points in inserting indexing entries.

In general, for efficient and fast access to the very large datasets, a multidimensional data access technique is needed for many real-time tasks. The R-tree multidimensional indexing tree data structure organizes data records in the form of hyper-rectangles and these hyper-rectangles usually called minimum bounding rectangles (MBRs) organized in the form of a tree hierarchy. R-tree multidimensional indexing tree data structure is height balanced and all data of objects are stored in leaves. Small rectangles are included at the bottom level and when the R-tree is transferred from bottom to the top a specific set of lower level small rectangles are grouped into one big high-level rectangle. Lee Ken C. K., et al. [16] said that R-tree and its variants, R+-trees, R*-trees, and aR-trees are data partitioning index techniques useful for clustering data objects in terms of minimum bounding boxes with an abstract mechanism. They proposed a variant of reverse nearest neighbor query called ranked reverse nearest neighbor query for searching and then proposed two algorithms for executing proposed query efficiently. These two algorithms are – k-counting and k-Browsing.

Each MBR is defined by two points, lower left corner and upper right corner and is represented as M (lower x1, y1, upper x2, y2). In general, the points lower x and y, and upper x and y may not be part of the actual data set. For efficient query processing of customer voting based similarity data search, index creation is inevitable for large data and R-tree multidimensional indexing tree data structure is mandatory for index creation.

Duc Thang et al. [5] said that fast, usability, simplicity and with reasonably good performance features are always

better than the best performing algorithm only in some cases and rare usage of the algorithm because of high complexity. Data clustering is one of the most important topics in data mining. Clustering is a method of arranging data objects into convenient and meaningful subgroups for further analysis, study, use, and application for effective data management. At present, the position of k-means algorithm is in the top-10 list of most important data mining algorithms. The main advantages of the k-means algorithm are – scalability, simplicity, robustness, understandability, fast, and ease of use. The main disadvantages of it are – selecting initial starting number of cluster centers is difficult and its time complexity is O(n2).

Charif haydar and Anne Boyer [3] proposed a clustering algorithm called mutual vote (MV) based on a statistical model. Authors said that their proposed clustering algorithm adjusts automatically to the data set and requires minimum parameters.

DINO IENCO, et al. [4] said that the process of clustering data objects containing only categorical attributes is a tedious task because defining a distance value between pairs of categorical attributes is difficult. Authors proposed a framework to find a distance measure between categorical attributes. Madhavi et al. [15] formulated measures on the data containing categorical attributes. They categorized existing measures as context-free and context-sensitive measures for categorical data. Usue Mori et al. [20] said that the most famous Euclidian distance and the common measures used for non-temporal data are not always the best methods for finding similarity between time series data because they do not deal with noise and misalignments in the time series data. Authors said that Euclidian distance suffers from noise and outliers problem.

Yung-Shen Lin et al. [22] said that similarity measures are being used extensively in text classification and clustering. In the literature, various methods used for similarity comparison are - Euclidian distance, Manhattan distance, taxicab distance, cosine similarity measure, city-block distance, Bray-Curties measure, Jaccard coefficient, extended Jaccard coefficient, Hamming distance, Dice coefficient, IT-Sim and so on. Authors have proposed a new measure for computing the similarity between two documents and they have extended to measure the similarity between two sets of documents. The proposed measure is applied in many real applications such as k-means like clustering, classification, and hierarchical clustering.

5. Proposed algorithm

ALGORITHM WCLUSTER (Threshold, Root, D)
INPUT
Threshold: user-specified similarity limit
Root: indexed tree
D: the dataset
OUTPUT
Set of clusters

1. Initialize cluster number i = 1
2. While D is not empty do
3. Object = first object in the D
4. Cluster set c_i = Theta-Similarity-Query (Root, Threshold, Object)
5. i = i + 1
6. update input dataset D = D - c_i
7. End-While

ALGORITHM Theta-Similarity-Query (Root, theta, q)
Input
Root: root node of the R-tree
theta: is the similarity measure threshold value
q: is the query object
Output
result-set: is the set of similar objects

1. node = create a new tree node
2. node = Root
3. if (minimum-similarity(node ,q) ≥ theta) then
4. result-set = result-set UNION p for every sub-tree (node)
5. end-if
6. if (node.type = leaf-node) then
7. for every p_i in the node do
8. reverse p_i vector = execute reverse top-k (p_i)
9. if (minimum-similarity(p_i, q) ≥ theta) then
10. result-set = result-set UNION p_i
11. end-if
12. end-for
13. else
14. for every sub-tree of node do
15. if (maximum–similarity(sub-tree , q)) ≥ theta then
16. node = sub-tree(node)
17. end-if
18. end-for
19. end-if
20. if (node is not empty) then
21. Theta-similarity-Query (node, theta, q)
22. end-if
23. return (result-set)

Sub Algorithm Minimum_Similarity(p,q)
Input

P:is the object (college) presents in the leaf node of the R-Tree

q:is the queried object(college)

Output

a numeric value representing the similarity measure between two objects

1. a= total list of students referenced the college object p
2. b= total list of students referenced the college object q
3. similarity = $\frac{a \cap b}{a \cup b}$
4. return similarity

Reverse Top-k computation Algorithm Reverse Top-k Full()

```
Reverse Topk[][]=new int [college][students]
for  i=1 to number of colleges do
{
   Col=0
  for j=0 to number of elements in each rows in top-k resultsset
  {
    for k=0 to number of elements in row
    {
        if  (topkresultset == I  ) then
            reverseTopk[i-1][col++]= j+1
    }
 }
}
```

Algorithm reverseTopklist(obj)

Input

Obj:collegeObject

Output

List of students

```
for i=1 to number of colleges do
{
     if (collegelist[i][1]=obj) then
        return i^th row list in reverseTopk[i]
    endif
}
endfor
```

The WCLUSTER algorithm makes use of the above similarity search algorithm. WCLUSTER provides the exhaustive set of clusters. For each step of the iterative process, a cluster is separated from the whole dataset and the remaining dataset is the candidate for the next iteration. The process ends when all the elements of the master dataset have been clustered.

The algorithm, Theta-Similarity-Query, returns all the similar objects of the given object q. The object may be any one of the items such as a tuple, product, book, patient, medicine, profile, mobile, wine and so on. R-tree index structure is mainly used for a fast searching purpose. During each of the search operation in each iteration a node is examined and if the node satisfies the maximum-similarity value greater than or equal to the theta value, then all the nodes within the sub-tree of the node are recursively searched and all the tuples of each node are processed based on the minimum similarity condition some tuples or objects are added to the result set. Whenever a leaf node is referenced Jaccard similarity measure is applied to all the objects of the leaf node by executing reverse top-k query for each object and at the sometimes similarity measure, similar (p, q) greater than or equal is also tested and the corresponding object is added to the result set during the computation of the similarity measure different types of pruning techniques are applied.

6. Comparison of proposed algorithm with traditional methods

The data grouping in recommender systems traditionally follows k-means approach. This k-means approach treats each attribute alike and does not consider weights with respect to priority attributes. In addition, the traditional approach needs high computational effort. The proposed approach using R-Tree saves a significant amount of computation time. The traditional approach needs comparatively more iterations for clustering than the proposed R-tree based method. The time complexity of the proposed approach is sub-linear, whereas the traditional methods like k-means algorithm need O (n^2) of time.

Time complexity of search operation in R-Tree is O (log n) in the best case when all the colleges belong to a single cluster and the R-Tree is called once. Hence best case time complexity is O (log n). In the worst case when no two engineering colleges have same profile of attributes then the R-Tree is called n times where n is the number of engineering colleges. Hence worst case time complexity of proposed algorithm is O (n log n). The average case time complexity of the algorithm may be anywhere between O(log n) and O(n log n)and it can be computed in best way as

$$\frac{(O(\log n)+O(n \log n))}{2} \approx \frac{O(\log n)}{2} + \frac{O(n \log n)}{2} \approx \frac{O(n \log n)}{2} \approx O(n \log n)$$

Hence, best case, average case and worst case time complexities of proposed algorithm respectively are O (log n), O (n log n) and O (n log n). In many real time cases average time complexity is considered to be the best estimator for algorithm time complexity.

Hence in terms of time complexity proposed algorithm is superior than many of the traditional clustering algorithms.

Georgoulas Konstantinos, et al. [10] introduced a new user-centric approach for finding object's similarities. New approach considers not only values of attributes of objects but also preferences of attributes of objects are used in finding similarities between objects. Authors said that proposed technique is very much useful for business organizations in finding business status details of a particular product/object and a more efficient, effective, optimal marketing business policy can be established and products can be clustered based on the preferences of customers.

Table 1: Existing K-means clustering algorithm execution times

Sno	Number of colleges	Execution time in seconds	Clusters
1	50	9	6
2	100	23	9
3	150	53	13
4	200	94	14
5	296	170	18

Table2: Execution times of proposed W-clustering algorithm with R-tree

Sno	Number of colleges	Number of students	Execution time in seconds	Clusters
1	50	50	1	9
2	100	100	4	13
3	150	150	12	14
4	200	200	31	17
5	296	300	46	19

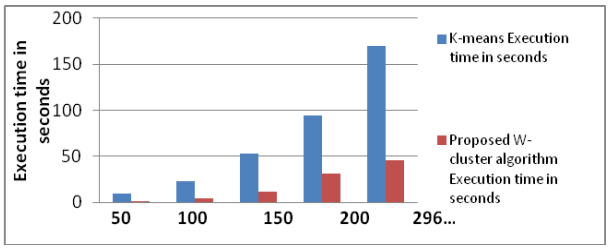

Fig. 1 Execution times of k-means and proposed algorithms

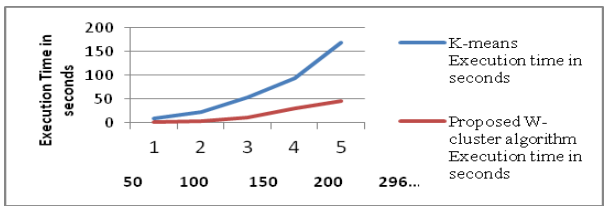

Fig. 2 Execution times of k-means and proposed algorithms

Experimentally obtained execution time details of both existing K-means clustering algorithm and proposed W-clustering algorithm with R-tree are respectively shown in the tables TABLE-1 and TABLE-2. Two different graphs, column chart and line chart, are drawn in Fihure-1 and Figure-2 respectively for the experimentally obtained data shown in TABLE-1 and TABLE-2 respectively. After observing the two graphs shown in Figure-1 and Figure-2 it is clear that for the datasets with small sizes the difference between execution times of existing k-means algorithm and proposed W-clustering algorithm is very small and the difference in execution times will increase rapidly as the sizes of datasets increase. For very large datasets the algorithm k-means is not scalable whereas the proposed W-cluster algorithm is scalable to the maximum extent and it is suitable for many real world applications because of the possible large data indexing capability of the R-tree indexing technique power. Figure-3 shows that number of clusters in the proposed W-cluster technique increases gradually as the size of the dataset increases

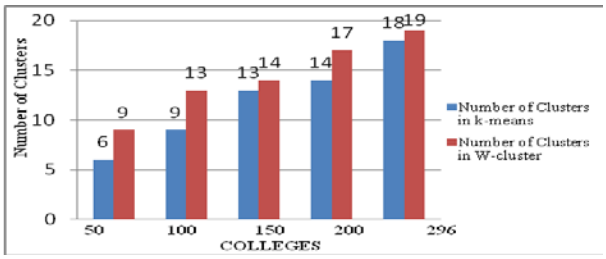

Fig. 3 Number of clusters in k-means and W-cluster

7. Data preparation

The data set is collected from the students of intermediate and B.Tech students with a sample of 2,000 students from various colleges. Through a structured online questionnaire, the data is gathered which consists of the student's opinions and preferences towards the engineering colleges they would like to join. The actual attribute information is also collected from about 500 engineering colleges. These two data sets were used to apply the proposed methodology.

8. The application

College recommender system is implemented in java and its main application is to take an optimal decision in selecting the best college for EAMCET admissions. The proposed algorithm was applied to a college data set having 296 records in which each record contains 7 attributes. The present system also uses student data set which contains their individual preferences of various attributes pertaining to various colleges. During the process of college clustering, both the above data sets are used. The execution process is applied by dividing the data sets into different cases using both fixed and variable parameters. Experimentally obtained results are placed in the form of tables and figures.

9. Results

The developed system is experimentally verified by taking two real-world data sets namely, colleges and students. Different output parameter values are noted and their relationships are plotted on graphs and charts.

Table 3: fixed variables set

Total number of colleges	296
Total number of students	50
Similarity between colleges	0.2
Maximum number of attributes	7

Table 4: execution results for various values of k in top-k

Case No	K in top-k	Execution time	Number of Clusters	Clusters data
1	5	3 min 38 sec	4	[1, 104, 105, 106, 108, 112, 113, 114, 115, 117, 118, 120, 121, 122, 124, 125, 136, 139, 14, 140, 142, 143, 144, 145, 146, 147, 148, 149, 150, 153, 154, 161, 167, 168, 169, 170, 172, 173, 179, 182, 185, 186, 187, 189, 190, 192, 193, 196, 197, 199, 20, 201, 203, 204, 207, 208, 209, 213, 214, 216, 217, 218, 22, 221, 222, 223, 224, 229, 231, 234, 241, 242, 245, 247, 249, 25, 255, 26, 260, 265, 268, 270, 275, 279, 28, 282, 287, 29, 31, 33, 34, 35, 36, 37, 39, 40, 46, 48, 49, 51, 52, 56, 6, 60, 63, 64, 7, 71, 73, 82, 83, 84, 89, 94, 97] [228, 230, 235, 236, 239, 240, 251, 254, 257, 259, 267, 276, 288, 290, 291, 292, 293, 294, 30, 41, 57, 65, 68, 70, 85] [112, 139, 142, 143, 144, 145, 147, 167, 172, 173, 187, 192, 193, 196, 197, 20, 201, 204, 207, 208, 214, 22, 222, 224, 255, 260, 265, 275, 28, 29, 31, 33, 35, 36, 37, 39, 56, 60, 63, 64, 7, 71, 73, 82, 83, 84, 97] [206, 228, 230, 235, 236, 239, 240, 251, 254, 259, 276, 280, 290, 291, 292, 293, 294, 41, 57, 68, 70]
2	10	3 min 55 sec	6	[1, 104, 105, 106, 108, 112, 113, 114, 115, 117, 118, 120, 121, 122, 124, 125, 136, 139, 14, 140, 142, 143, 144, 145, 146, 147, 148, 149, 150, 153, 154, 161, 167, 168, 169, 170, 172, 173, 179, 182, 185, 186, 282, 284, 285, 287, 29, 291, 292, 294, 30, 31, 33, 34, 35, 36, 37, 39, 40] [41, 46, 48, 49, 51, 52, 56, 57, 6, 60, 63, 64, 65, 66, 67, 68, 7, 70, 71, 73, 82, 83, 84, 85, 89, 94, 97,187, 189, 190, 192, 193, 196, 197, 199, 20, 201, 202, 203, 204, 206, 207, 208, 209, 213, 214, 216, 217, 218, 22, 221, 222, 223, 224] [228, 229, 230, 231, 234, 235, 236, 239, 240, 241, 242, 245, 246, 247, 249, 25, 251, 254, 255, 257, 26, 260, 265, 268, 270, 275, 276, 279, 28, 280, [259, 267, 288, 290, 293] [112, 172, 173, 204, 22, 251, 254, 255, 265, 275, 279, 280, 282, 284, 285, 291, 292, 294, 37, 39, 57, 64, 68, 7] [259, 272, 274, 290, 293] [22, 240, 251, 254, 276, 291, 292, 294, 37, 41, 57, 68, 7]
3	15	3 min 52 sec	6	[1, 104, 105, 106, 108, 112, 113, 114, 115, 117, 118, 120, 121, 122, 124, 125, 136, 139, 14, 140, 142, 143, 144, 145, 146, 147, 148, 149, 150, 153, 154, 161, 167, 168, 169, 170, 172, 173, 179, 182, 185, 186, 25, 251, 254, 255, 257, 26, 260, 265, 268, 270, 275, 276, 279, 28, 280, 282, 284, 285, 287, 29, 291, 292, 294, 30, 31, 33, 34, 35, 36, 37, 39, 40] [187, 189, 190, 192, 193, 196, 197, 199, 20, 201, 202, 203, 204, 206, 207, 208, 209, 213, 214, 216, 217, 218, 22, 221, 222, 223, 224, 228, 229, 230, 231, 234, 235, 236] [41, 46, 48, 49, 51, 52, 56, 57, 6, 60, 63, 64, 65, 66, 67, 68, 7, 70, 71, 73, 82, 83, 84, 85, 89, 94, 97, 239, 240, 241, 242, 245, 246, 247, 249, 259, 267, 288, 290, 293] [112, 172, 173, 204, 22, 251, 254, 255, 265, 275, 279, 280, 282, 284, 285, 291, 292, 294, 37, 39, 57, 64, 68, 7] [259, 272, 274, 290, 293] [22, 240, 251, 254, 276, 291, 292, 294, 37, 41, 57, 68, 7]

In a similar way for different values k in top-k experiments are executed and the obtained results are shown in the TABLE-4. Execution times are noted tabulated against different values of k in top-k value.

Table 5: top-k versus execution time

Serial No.	K in top-k	Execution time in sec
1	5	3 min 38 sec = 218
2	10	3 min 55 sec = 235
3	15	3 min 52 sec = 232
4	20	3 min 43 sec = 223
5	25	3 min 52 sec = 232
6	30	3 min 27 sec = 207
7	35	3 min 43 sec = 223
8	40	3 min 30 sec = 210
9	45	4 min 22 sec = 262
10	50	3 min 54 sec = 234
11	55	3 min 49 sec = 239
12	60	3 min 54 sec = 234

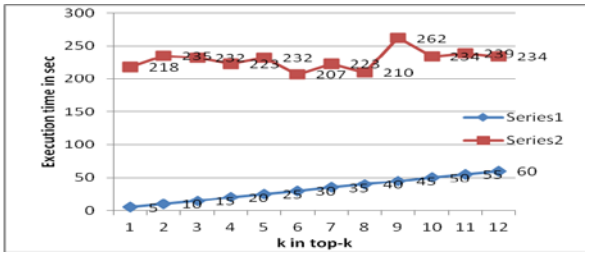

Fig. 4 Relationship between k value in Top-k and execution Time

Figure-4 shows the relationship between k value in top-k and the corresponding execution time. Here maximum college data size and student preferences size are kept constant. Figure-4 shows that there will not be drastic ups

and downs in execution times for various values because data size is kept constant. The range of execution times is approximately fixed. Here execution time is mainly based on size of the data set. Execution time increases as the data set size increases.

Table 6: top-k versus number of clusters

Serial No.	K in top-k	Number of Clusters
1	5	4
2	10	6
3	15	7
4	20	10
5	25	12
6	30	13
7	35	15
8	40	17
9	45	19
10	50	21

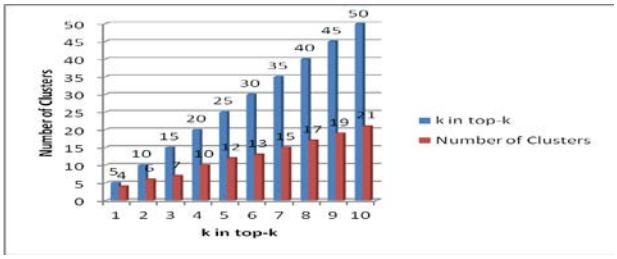

Fig. 5 top-k versus number of clusters

Figure-5 depicts that the number of clusters will increase when there is increase in the value of k in top-k list. This is certainly true because when k value in top-k increases, the same object appears in many preference lists and consequently preference groups (clusters) will increase in a natural manner. Hence figure-5 shows that the number of clusters will progressively increase with the increase of k values.

Table 7: fixed parameter list

Total tuples =	296
K in top-k =	50
Theta similarity =	0.2
Maximum attributes =	7

Table 8: total data set size versus variable sizes of weights and execution times

Serial No.	Maximum Students	Execution Time	Number of Clusters
1	100	5 min 9 sec = 309	20
2	200	8 min 41 sec = 521	17
3	300	11 min 29 sec = 689	20
4	400	14 min 5 sec = 845	21
5	500	17 min 4 sec = 1024	21
6	600	20 min 21 sec = 1221	21
7	700	23 min 56 sec = 1436	20
8	800	27 min 12 sec = 1632	21
9	900	31 min 55 sec = 1925	21
10	1000	35 min 49 sec = 2149	21

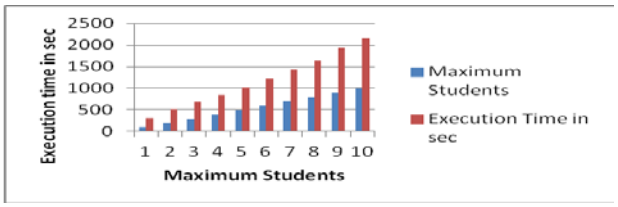

Fig. 6 Relationship between maximum preferences and execution times

FIGURE-6 shows that there exist a linear relationship between preferences and execution times. It depicts a natural phenomenon that execution time increases as the data seize increases. For smaller preference sets the execution time follows linear relationship and for larger preference sets the execution time follows sub-linear relationship. That is scalability is linear for simple data sets where as scalability is sub-linear when the number tuples in the data set increases gradually. Also it is true that the scalability will decrease as the dimensionality of the data set increases.

Table 9: fixed parameter list

Total Colleges	100
Total Students	100
K in top-k	50
Maximum attributes	7

Table 10: various theta values

Serial No.	Theta value	Execution time in sec	Number of clusters
1	0.1	23	18
2	0.2	25	17
3	0.3	25	14
4	0.4	26	11
5	0.5	23	9
6	0.6	23	7
7	0.7	21	6

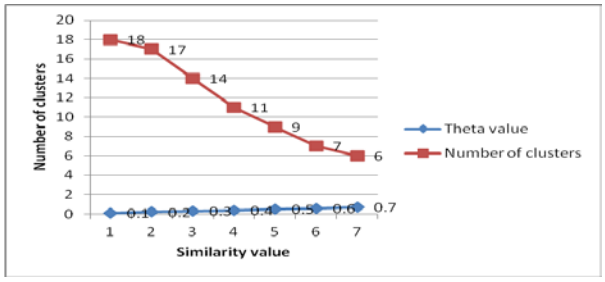

Fig. 7. Relation between similarity measure and number of clusters

FIGURE-7 shows that total number of clusters generated will be decreased smoothly in a continuous manner as the similarity between cluster objects increases and this is true because when the similarity threshold value set is very high

then many objects will not satisfy set threshold similarity value and consequently not included in any of the clusters. Many objects are excluded from the clustering process, as their similarity threshold value is very less, which results decrease in the number of clusters.

Table 11: represents the summarization of the results of the first three cases executed. The details of the rest of the cases resembling the first three cases and so were not summarized again.

S No.	Number of Colleges	Number of Students	Execution time in sec	Number of clusters	Actual clusters
1	50	50	12	2	[14, 20, 22, 25, 26, 28, 29, 30, 31, 33, 34, 35, 36, 37, 39, 40, 41, 46, 48, 49, 6, 7] [112, 167, 172, 173, 193, 204, 22, 228, 230, 235, 236, 239, 240, 255, 265, 275, 276, 282, 285, 33, 35, 36, 37, 39, 41, 56, 64, 7, 70, 84]
2	100	100	81	4	[1, 14, 20, 22, 31, 34, 37, 48, 49, 51, 52, 6, 63, 7, 73, 82, 83, 89, 94, 97] [28, 29, 30, 33, 35, 36, 39, 40, 41, 46, 56, 57, 60, 65, 66, 67, 68, 70, 71, 84, 85] [1, 22, 31, 37, 63, 7, 83, 89, 97] [25, 26, [64, 63]
3	150	150	145	5	[1, 104, 105, 106, 108, 112, 113, 114, 115, 117, 118, 120, 124, 125, 136, 14, 140, 146, 147, 150, 20, 22, 25, 26, 31, 34, 37, 48, 49, 51, 52, 6, 63, 64, 7, 73, 82, 83, 89, 94, 97] [121, 122, 139, 142, 143, 144, 145, 30, 33, 35, 36, 40, 41, 46, 56, 57, 60, 65, 66, 67, 68, 70, 71, 85] [1, 112, 147, 22, 31, 37, 52, 63, 64, 7, 73, 82, 83, 89, 97, [122, 29, 33, 35, 36, 39, 40, 41, 46, 56, 57, 67, 68, 70, 84] [147, 20, 6, 63, 148, 149, 28, 33, 35, 36, 41, 56, 57, 67, 68, 70] [150, 104, 83, 94, 97, 85, 33, 35, 36, 56, 84]

Table 12: relationships among colleges, students, execution time and clusters

SNo.	Number of Colleges	Number of Students	Execution time in sec	Number of clusters
1	50	50	12	2
2	100	100	81	4
3	150	150	145	5
4	200	200	211	5
5	296	300	836	18
6	296	400	836	20

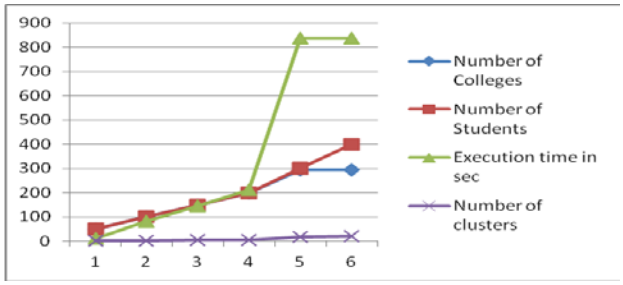

Fig. 8 relationships among colleges, students, execution time and clusters

FIGURE-8 shows the relationships among colleges, students, execution times and clusters formed after execution. Number of colleges and execution times increase linearly up to a certain point beyond that point execution time curve follows exponential growth rate as is the case with many real world large data sets. Number of clusters increases smoothly as the number of colleges and students increases.

10. Conclusions

A novel technique for college recommendation was presented. A well potent problem of college recommender system was undertaken to solve with the proposed grouping and recommendation technique. The proposed technique is mostly suitable for present trends of data available. Intelligent and time saving recommendation systems can be developed embedding the proposed R-Tree and top-k query approaches. The same was implemented and applied to develop a recommender system for college selection based on students' preferences. The results showed that the proposed technique is more reliable, more intelligent and faster than the existing approaches.

A novel technique for top engineering college recommendation is developed. A well potent problem of engineering college recommender system for students is undertaken to salve many of the problems that frequently occur during EAMCET admission process with respect to student voting/preference/rating/opinions. A new intelligent and time saving system is developed based on approaches R-Tree, Top-K query and voting/preference of the students. The developed system is tested on the data collected from various engineering colleges. The college data set represents all the profile attributes of engineering colleges. Also students voting/preferences are collected with respect to college attributes and used in the present recommendation system. Experimental results show that proposed system is reliable, faster, intelligent and more useful for aspirants of engineering college admissions. In

the feature the system can be extended for admissions like IITs, IIITs, and NITs and so on. In future the same setup can be extended for many more applications relating to recommender systems that can exhibit the same betterments.

Acknoledgement

To collect the related data, an online survey is conducted using a questionnaire. I am always thankful to all and sundry who participated and cooperated in data collection.

References

[1] Akrivi Vlachou, Christos Doulkerids, Kjetil Norvag, and Yannis Kotidis, "Identifying the Most Influential Data Objects with Reverse Top-k Queries," Proceedings of the VLDB Endowment, Vol. 3, No. 1, Copy right 2010 VLDB Endowment 2150-8097/10/09

[2] Amit Singh, Hakan Ferhatosmanoglu, and Ali Saman Tosun, "High Dimensional Reverse Nearest Neighbor Queires," CIKM'03, November 3-8, 2003, New Orleans, Louisiana, USA, copyright 2003 ACM 1-58113-723-0/03/0011

[3] C.C. Aggarwal, Recommender Systems: The Textbook, DOI 10.1007/978-3-319-29659-3 1© Springer International Publishing Switzerland 2016

[4] Charif Haydar, Anne Boyer, "A New Statistical Density Clustering Algorithm based on Mutual Vote and Subjective Logic Applied to Recommender Systems", UMAP 2017 Full Paper UMAP'17, July 9- 12, 2017,Bratislava,Slovakia

[5] DINO IENCO, RUGGERO G. PENSA and ROSA MEO, "From Context to Distance: Learning Dissimilarity for categorical Data Clustering," ACM Journal Vol. X. 10 2009, pages 1- 0??

[6] Duc Thang Nguyen, Lihui Chen, Chee keong Chan, "Clustering with Multiviewpoint-Based Similarity Measure," IEEE Transactions on Knowledge and Data Engineering. Vol. 24. No. 6. June 2012

[7] Elham S.Khorasani, Zhao Zhenge, and John Champaign. AMarkov Chain Collaborative Filtering Model for Course Enrollment Recommendations: 2016, "IEEE International Conference on Big Data (Big Data)", P. 3484 – 3490

[8] Elke Achtert, Christian Bohm, Peer Kroger, Peeter Kunath, Alexy Pryakhin, Matthias, " Efficient Reverse k-Nearest Neighbor Search in Arbitrary Metric Spaces," SIGMOD 2006 June 27-29, 2006 Chicago, Illinois, USA.

[9] Fazeli Soude, Hendrik Drachsler, Marlies Bitter-Rijpkema, Francis Brouns, Wim van der Vegt, and Peter B. Sloep, "User-centric Evaluation of Recommender Systems in Social Learning Platforms: Accuracy is Just the Tip of the Iceberg", IEEE Transactions on Learning Technologies, August 26, 2015

[10] Georgoulas Konstantinos, Akrivi Vlachou, Christos Doulkeridis, and Yannis Kotidis, "User-Centric Similarity Search," IEEE Transactions on Knowledge and Data Engineering, Vol. 29, No. 1, January 2017

[11] Hana Bydžovská. Course Enrollment Recommender System: Proceeding of the 9th International Conference on Educational Data Mining, P. 312 – 317.

[12] Hector Nunez, Miquel sanchez-Marre, Ulises Cortes, Joaquim Comas, Montse Martinez, Ignasi Rodriguez-Roda, Manel Poch, "A Comaprative study on the use of similarity measure in case based reasoning to improve the classification of environmental system situations,", ELSEVIER, Environmental Modeling and Software XX (2003) xxx-xxx.

[13] HristidisVagelis, Nick Koudas, Yannis Papakonstantinou, "PREFER: A System for the Efficient Execution of Multiparametric Ranked Queries", ACM SIGMOD '2001 Santa Barbara, California,USA

[14] Jamil Itmazi and Miguel Megias (2008), Using recommendation Systems in Course Management Systems to Recommend Learning Objects, P. 234 - 240.

[15] J. Bobadilla et al. "Knowledge-Based System" 2013 Elsevier B.V.

[16] Lee Ken C. K., Baihua Zheng, Wang-Chien Lee, "Ranked Reverse Nearest Neighbor Search", IEEE Transactions on knowledge and Data Engineering. Vol. 20, No.7, July 2008

[17] Madhavi Alamuri, Bapi raju Surampudi and Atul Negi, "A Survey of Dustance / Similarity Measure for categorical Data," 2014 International Joint conference on Neural Networks (IJCNN), July 6-11, 2014, Beijing, china.

[18] Queen Esther Booker (2009). A Student Program Recommendation System Prototype: Issues in Information Systems, P. 544 - 551.

[19] Subba Reddy.Y and Prof. P. Govindarajulu," A survey on data mining and machine learning techniques for internet voting and product/service selection", IJCSNS International Journal of Computer Science and Network Security, VOL.17 No.9, September 2017

[20] Usue Mori, Alexander Mendiburu, and Jose A.Lozano, "Similarity Measure Selection for Clustering Time Series databases," IEEE Transactions on Knowledge and Data Engineering. Vol. 28. No. 1. January 2016

[21] Vlachou Akrivi, Charitos Doulkeridis, Yannis Kotidis, Kjetil Nrvag, "Reverse Top-k Queries", ICDE Conference 2010 978-1-4244-5446-4/10

[22] Yung-Shen Lin, Jung-Yi Jiang, and Shie-Jue Lee, "A Similarity Measure for Text Classification and Clustering," IEEE Transactions on Knowledge and Data Engineering. Vol. 26. No. 7. July 2014

[23] Zhibo Wang, Jilong Liao, Qing Cao, Hairong Qi, and Zhi Wang, "Friend book: A Semantic-based Friend Recommendation System for Social Networks", IEEE Transactions on Mobile Computing.

Y.Subba Reddy received M.Sc (Computer Science) degree from Bharathidasan University, Tiruchirapalli, TN and M.E degree in Computer Science & Engineering from Sathyabama University, Chennai, TN. He is a research scholar in the Department of Computer Science, Sri Venkateswara University, Tirupati, AP, India. His research focus is on Data Mining in Clustering and Similarity measures.

P.Govindarajulu, Professor, Department of Computer Science, Sri Venkateswara University, Tirupathi, AP, India. He received his M. Tech., from IIT Madras (Chennai), Ph. D from IIT Bombay (Mumbai). His area of research are Databases, Data Mining, Image processing, Intelligent Systems and Software Engineering