Comparative Analysis of Skyline Query Execution using Imputation Techniques on Partially Complete Data

S. Deepa Kanmani[1], E. Kirubakaran[2] and Elijah Blessing Rajsingh[3]

deepakanmanisampath@gmail.com[1]
ekirubakaran@gmail.com[2]
elijablessing@gmail.com[3]

Research Scholar, Karunya Institute of Technology and Sciences[1]
Professor, Karunya Institute of Technology and Sciences [2]
Professor, Karunya Institute of Technology and Sciences[3]

Abstract. In this era, the Database community depends on preference queries to satisfy user needs according to their given preferences. Skyline query is one of the preference-based queries. The skyline proceeds with contradictory preferences given by the user. Skyline query derived from the maximum vector problem which deals with Pareto dominance. Skyline query always leads to promising results in the complete data environment. Due to the dynamic data setup, this leads to unknown values or noisy data in the database. This type of data leads partially complete data environment and this affects the performance of skyline queries. This paper gives an analysis of complete and partially complete data using skyline queries with imputation techniques. Two different imputation techniques are used namely Random forest and Amelia to execute the Skyline query on partially complete data. The experimental study gives the solemnity of partially complete data using the skyline query and its influence on the result of the query.

Keywords: Database, skyline query, preferences, pareto dominance, imputation techniques.

1 Introduction

Recently, several preference queries have been proposed which are expected to find the best items able to meet the preferences given by the user. There are several queries were proposed under this principle. They are top-k, skyline, ranked skylines, k-representative dominance, k-dominance, top-k dominating, and k-frequency.Skyline queries are a powerful tool in the current database management system. Implementing skyline in the database-oriented applications empowers users to submit more than one objective in the query which gives more precise results compared to the traditional queries. Skyline is often used in Multi-criteria decision-making applications. Here conflicting criteria contribute towards selecting the most promising results. Assume an example a customer wants to buy a mobile. Consider further that the personal attention on two significant attributes of mobile, they are cost and lightweight. Unfortunately, these attributes are contradictory and accordingly the number of results should be cautiously selected.

Skyline queries are extensively used in different domains such as multi-criteria decision-making applications where many different attributes are involved in the query to select the most relevant results. Another domain is a decision support system which aids to associate various interests in recommending a strategic decision. Other domains include e-commerce which helps

ICASISET 2020, May 16-17, Chennai, India

DOI 10.4108/eai.16-5-2020.2303973

the user to select the most interesting products according to their personal preferences. Skyline queries works under the principle of pareto dominance. This query can extract one data item p over the other data item q if and only if p is better than q with respect to all dimensions (attributes) and not worse than q from at least one dimension (attribute). This scenario is depicted in Fig 1.

Fig.1. User Prefernces

2 Background of Skyline query

In 1975, the skyline query for the current processing method was studied[1].The phrase "sky-line points" used in the skyline query is originally called the word "Pareto optimal" typically used in economics and engineering, skyline explored in 2001[2] and named the word "skyline points," the skyline query was extensively researched in database literature and the primary focus is to learn how to efficiently calculate skyline points on a very large relational database [3][4]. There is a humpty number of approaches are pro-posed to handle the skyline query. Initially skyline query only addressed towards complete data in a centralized environment. Data management and storage have been progressively decentralized over recent years. Progressive query operators, like skyline queries, are important to help users access the tremendous amount of data available by determining a collection of interesting data objects. The processing of skyline queries in highly distributed environments introduces implicit challenges and requirements and needs non-traditional techniques due to content delivery and lack of global expertise.

2.1 Partially Complete Data

Processing of skyline queries in the incomplete database is considered one of the most significant issues in database applications. Most of the existing approaches assume that all data items are available. This assumption is not true always due to larger real-world database applications. Applying existing algorithms on this incomplete setup leads to more pairwise comparisons among the data items. Existing approaches fail to save the transitivity property of the skyline. Losing this property leads to cyclic dominance in the skyline query. Failing to meet these properties gives a huge result set.

Definition for Skyline or Dominant Queries:
Retrieve a data or object a_i from the Dataset D if a_i is as good as a_j (where i and j are different data items) with respect to all dimensions and firmly better than a_j in at least one dimension. Finally, the result contains a set of skyline data, result = $\{a_i, a_j \in D, a_i \text{ dominates } a_j\}$.

2.2 Properties of Skyline query

Transitivity: Given a_i, a_j, $a_k \in$ D(Database), if $a_i \succ$(dominates) a_j, and $a_j \succ$ (dominates) a_k, according to transitivity property ai $\succ$ (dominates) a_k holds for complete data. However, it may not true for incomplete data.
Cyclic dominance: Given a_i, a_j, $a_k \in D$, $a_i \succ a_j$, $a_j \succ a_k$, and, $a_k \succ a_i$ may happen over incomplete data.

This section detailed review of the approaches and algorithms available for handling a partially complete dataset.

Definition for Incomplete Database:
Given a database D where D is said to be incomplete if it contains at least a data item pj with missing values in one or more dimensions dk (attributes); otherwise, it is complete.

The first skyline algorithm for incomplete data[5]. Initially, they introduced two new algorithms namely replacement and bucket algorithms.Another algorithm for skyline computation [6][11] Replacement Based Sets Skyline Queries (RBSSQ) method which contains two phases: i) data preprocessing phase and ii) skyline sets the computation phase. In the first preprocessing phase, examine the database for finding missing values of the tuple. Then, replace the missing values of an attribute with a value.

An efficient Sort-based Incomplete Data Skyline (SIDS) algorithm for evaluating the skyline query over incomplete data discussed in [7].[12] This algorithm is an improvement of an existing bucket algorithm. Sorting based Cluster Skyline Algorithm(SCSA) for retrieving skyline from incomplete data proposed in [9-10].

3 Imputation Techniques

This section presents the imputation techniques on the skyline query to execute in partially complete data. There are two different imputation techniques are proposed to execute the skyline query. Since the missingness of the dataset affects the performance of the query, missing values are estimated using Amelia and Random forest.

3.1 Amelia

The working principle of Amelia which is depicted in Fig. 2. This imputation belongs to the principle of multiple imputations.

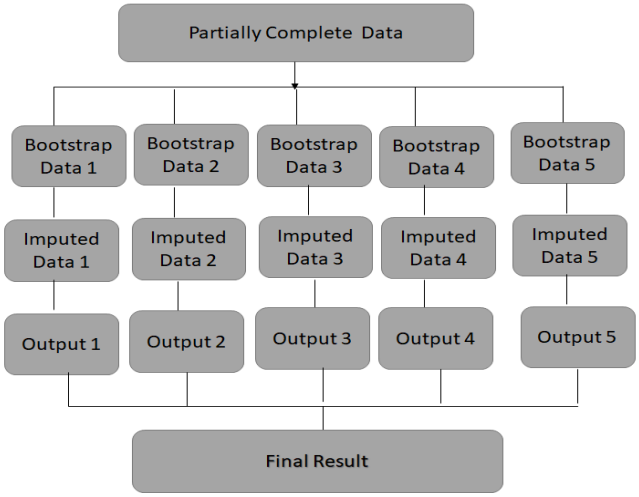

Fig.2. The workflow of the Amelia

Algorithm Steps

1. Using Bootstrapping: Sampling the dataset with 'n' number of repetition.
2. Consider a set of starting parameters and for all copies of incomplete data sets
3. Using Expectation step: Using the observed data available from the dataset, predict the missing values.
4. Using Maximization Step: Complete data generated after the expectation step and used to update the parameters.
5. Repeat steps 2 and 3 until the convergence takes place. (for all copies of the dataset)
6. Combine them into a single dataset.

3.2 Random Forest Imputation

Random forest is an attractive approach to solve missing data. This can accommodate mixed forms of missing data, and have the ability to scale to large data settings.
The work flow of Random forest is depicted in Fig 3.

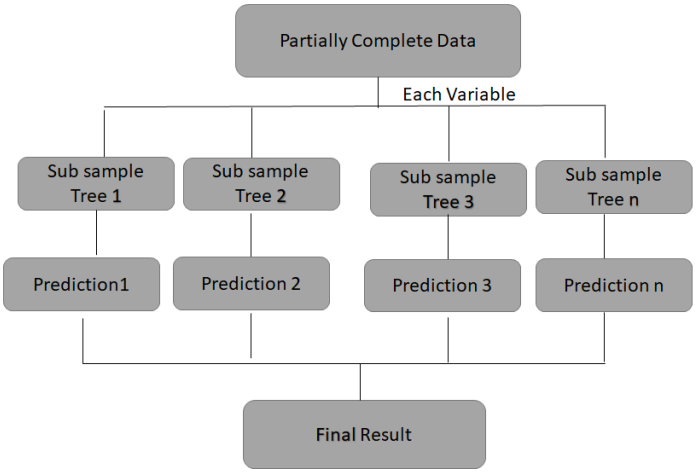

Fig. 3. The workflow of Random forest

Algorithm Steps

1 Considering a number of instances in the training set is N.Samples of such N cases are then taken at random, but with replacements. This sample is the training set for tree growth.
2 When there are x input variables and x< X where x is a random value and x variables are selected at each node. The best split of this x is used to split the node. Expand the tree to grow the forest
3 Each tree is developed to the greatest extent possible and there is no pruning.
4 Estimate new data by aggregating tree predictions.

4 Experimental Analysis

The skyline query is the most valuable tool for data mining and decision making applications. However, dealing with the massive amounts of data on the web or internet are frequently partially complete and uncertain due to data randomness, transmission errors, and many other reasons. This section presents the analysis between complete data and partially complete data.

In this comparative analysis, real-world diabetic datasets have been taken. This dataset contains 9 attributes and 768 records. This dataset analyzed using parameters like dataset size, execution time and dimensions. This dataset dominant records are identified using the skyline query by varying the incomplete data from 10 to 40%. Due to dynamic and data over the network which gives more amount of missing values may be generated. This yields huge number of pairwise comparisons among the records in a given dataset.Due to this incompleteness some times it may lead to an empty result or huge result to the user. Therefore, a user may get noninteresting items that may fail to satisfy the user's needs.

Partial data affects the performance and execution of the skyline query. The experimental analysis of the skyline query in both complete and partially complete data setup has been done.

The comparison of the execution time of incomplete data with complete data, it is increased by 49.4% concerning different data size. The result size of incomplete data with complete data which is increased by 60.4% concerning different data size.

The partial data imputed by Amelia and Random forest approaches. Then, this data is taken as complete data for further analysis using the skyline query. The execution of the skyline query is done on both approaches. Fig 4 shows the result size of the skyline query using both the methods. Amelia yields a better result than the Random forest.

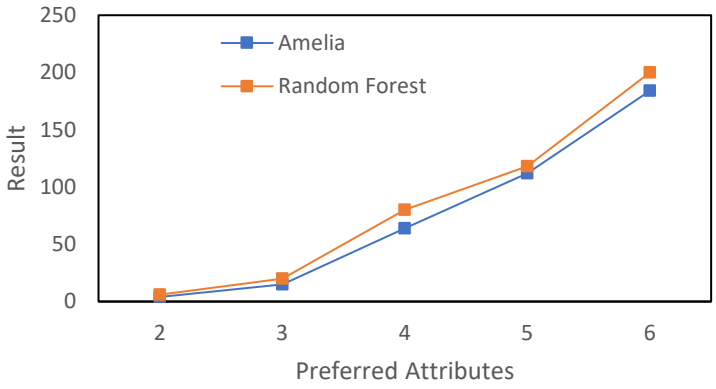

Fig. 4. Result size analysis

Fig 5 gives a comparative analysis of both the methods ofthe execution time parameter. From the analysis, Amelia can reduce the skyline query execution time.

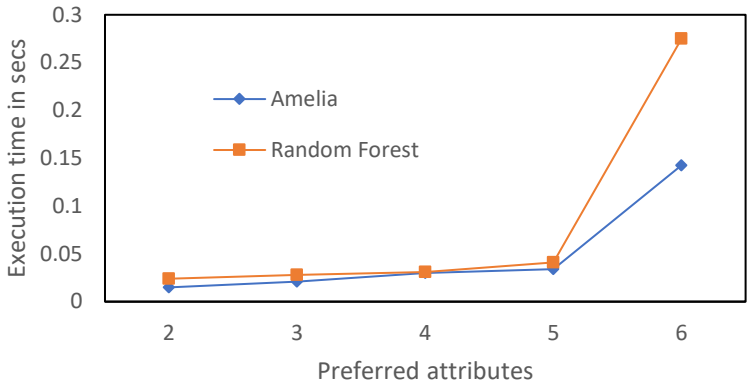

Fig.5.execution time

Fig 6 shows the accuracy measure of the skyline query using both methods. Using Skyline query, dominant records are identified. Based on the result, the Amelia imputation method using skyline query can give good accuracy comparatively with Random Forest.

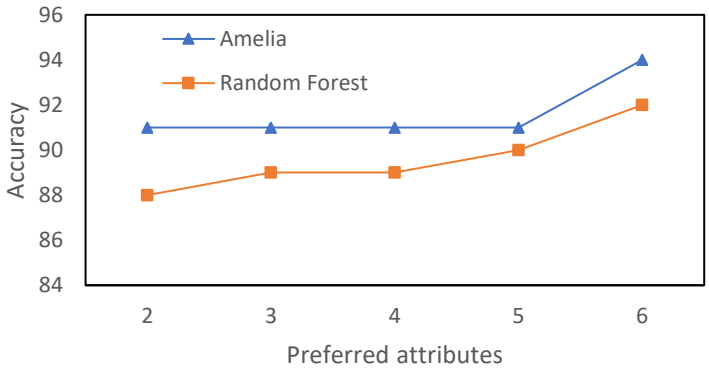

Fig.6. Accuracy

5 Conclusion

Skyline query always good when it deals with a complete data environment. This may not be true always. Since data-driven applications are dynamic and flow over the network. This leads to errors or unknown values to the dataset. Handling such type of data is challenging one using skyline query. In big data analytics using such types of queries is a major one to be addressed. This paper providesa detailed study and analysis of the skyline query in both complete and partially complete data environments. This paper addressed the issue with imputation techniques to make complete data to provide interesting results to the user. This will enable to improve the decision-making process. In the future, Efficient techniques are much needed to handle such type of preference-based queries.

References

1. Alwan, Ali, Ibrahim, Hamidah, Udzir, Nur, Sidi, Fatimah. Preference Evaluation Techniques of Preference Queries in Database. . *International Journal of Advancements in Computing Technology* 2013; 5(5): 756-766.
2. Börzsönyi, Donald Kossmann, and Konrad Stocker. The skyline operator. In Proceedings of the 17th International Conference on Data Engineering, pages 421–430, Washington, DC, USA, 2001. IEEE Computer Society
3. EvangelosDellis, Bernhard Seeger. Efficient Computation of Reverse Skyline Queries. *The International Journal on Very Large Data Bases* 2007; (1): 291-302.
4. Cuiping Li, Beng Chin Ooi, Anthony K. H. Tung, and Shan Wang. Dada: a data cube for dominant relationship analysis. In SIGMOD Conference, pages 659–670, 2006
5. Khalefa, Mohamed E., Mohamed F. Mokbel, and Justin J. Levandoski. "Skyline query processing for incomplete data." *2008 IEEE 24th International Conference on Data Engineering*. IEEE, 2008.
6. Arefin MS, Morimoto Y. Skyline Sets Queries for Incomplete Data. *International Journal of Computer Science & Information Technology* 2012; 4(5): 67-80.
7. Bharuka R, Kumar PS (2013) Finding skylines for incomplete data. Paper presented at the Proceedings of the 24th Australasian Database Conference - Volume 137, Adelaide

8. Gulzar, Yonis . SCSA: Evaluating Skyline Queries in Incomplete Data. *Applied Intelligence* 2018; 49(5): 1636-1657.
9. Gulzar, Yonis, Alwan, Ali, Alshaikhli. Processing Skyline Queries in Incomplete Data: Issues, Challenges and Future Trends. *Journal of Computer Science* 2017; 13(5): 647-658.
10. Han, Xixian. Efficient skylinecomputation on big data. *IEEE Transactions on Knowledge and Data Engineering*; 2012;25(11): 2521-2535.
11. K. Vijayakumar, Chokkalingam Arun, "Integrated cloud-based risk assessment model for continuous integration",Int. J. Reasoning-based Intelligent Systems", Vol. 10, Nos. 3/4, 2018.
12. Vijayakumar, S. Suchitra and P. Swathi Shri, "A secured cloud storage auditing with empirical outsourcing of key updates", Int. J. Reasoning-based Intelligent Systems, Vol. 11, No. 2, 2019.