ISSN No. 0976-5697

Volume 4, No. 9, July-August 2013

International Journal of Advanced Research in Computer Science

REVIEW ARTICLE

Available Online at www.ijarcs.info

Approximate Nearest Neighbour Using Data Mining

[1*]Deepika Verma [2]Namita Kakkar
M.Tech Student (CSE)*, Asstt. Professor in Dept. of CSE
Rayat and Bahra Institute of Engg. & Biotechnology,
Sahauran, Punjab.
Er.deepikaverma@gmail.com*, namita.kakkar@gmail.com

Abstract: Data mining may be viewed as the extraction of the hidden predictive information from large databases, is a powerful new technology with great potential to analyze important information in the data warehouse. Nearest neighbor search (NNS), also known as proximity search, similarity search or closest point search, is an optimization problem for finding closest points in metric spaces. This paper presents an extensive study of existing techniques of the approximate nearest neighbour in data mining and a new algorithm is proposed for nearest neighbour. In this paper, we studied and compared k-d tree algorithm and brute force algorithm on various levels. The major contribution achieved by this research is the detection of flaws in both k-d tree and brute-force algorithms which helps to propose a new algorithm.

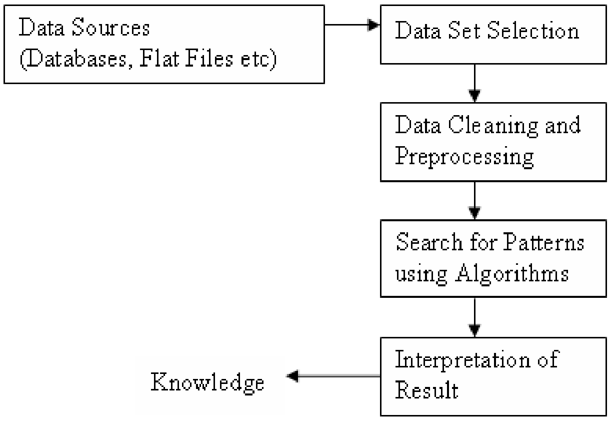
Keywords: Data Mining, Nearest neighbour, Approximate K-NN, K-d tree, Brute-force.

I. INTRODUCTION

Data mining is an interdisciplinary subfield of computer science which involves computational process of large data sets patterns discovery. The goal of this advanced analysis process is to extract information from a data set and transform it into an understandable structure for further use. The methods used are at the juncture of artificial intelligence, machine learning, statistics, database systems and business intelligence. Data Mining is about solving problems by analyzing data already present in databases [1].

Data mining is the process of analyzing data from different perspectives and summarizing it into useful information. Data mining is also known as Knowledge Discovery in Data (KDD). Data mining uses mathematical algorithms to part the data and evaluate the probability of future events. It automatically searches large volume of data to discover pattern and trend. Data mining software is one of a number of analytical tools for analyzing data. Technically, data mining is the process of finding correlations or patterns among dozens of fields in large relational databases.

Data mining [3] is a powerful tool that can help to find patterns and relationships within our data. Data mining discovers hidden information from large databases. To ensure meaningful data mining results, we must understand our data. Figure 1 shows the Process of Data Mining.


Data Sources
(Databases, Flat Files etc)
Data Set Selection
Data Cleaning and Preprocessing
Search for Patterns using Algorithms
Interpretation of Result
Knowledge

Following can be done using Data mining process.

a. It take out, change, and load data onto the data warehouse system.
b. It store and manage the data in a database system.
c. It provides data access to business analysts, information technology professionals and other persons.
d. Analysis of the data can be done by application software.
e. It represents the data in an understandable format, such as a graph or table.

A. *Nearest Neighbour:*

Nearest neighbour search (NNS) also known as proximity search, similarity search or closest point search, is an optimization problem for finding closest points in metric spaces. The problem is: given a set S of points in a metric space M and a query point q ∈ M, find the closest point in S to q. In many cases, M is taken to be d-dimensional Euclidean space and distance is measured by Euclidean distance or Manhattan distance [4]. For some applications we may have N data-points and wish to know which is the nearest neighbor for every one of those N points. This could of course be achieved by running a nearest-neighbor search once for every point, but an improved strategy would be an algorithm that exploits the information redundancy between these N queries to produce a more efficient search. Nearest neighbor is a technique that is quite similar to clustering - its essence is that in order to predict what a prediction value is in one record look for records with similar predictor values in the historical database and use the prediction value from the record that it “nearest” to the unclassified record.

B. *K-nearest neighbors:*

K-nearest neighbour uses the training set directly to classify an input when an input is given. When using a k-nearest neighbor algorithm on an input with d attributes the input is classified by taking a majority vote of the k (where k is some user specified constant) closest training records across all d attributes. "Closest" as used means the distance an attribute is away from the same attribute of the training set, using some specified similarity metric. k-nearest neighbour (kNN) in which nearest neighbour is calculated on the basis of value of *k*, that specifies how many nearest neighbours are to be considered to define class of a sample data point. The training points are assigned weights according to their distances from sample data point. But still, the computational complexity and memory requirements remain the main concern always [5]. The general model of a KNN query is that the user gives many query types such a point query in multidimensional space and a distance metric for measuring distances between points in this space. The system is then tried to find the *K* closest or nearest answers in the database from the submitted query (i.e. query point). Generally distance metrics may include: Euclidean distance, Manhattan distance, etc [6].

C. Approximate Nearest Neighbour (ANN):

In some applications it may be acceptable to retrieve a "good guess" of the nearest neighbor. In those cases, we can use an algorithm which doesn't guarantee to return the actual nearest neighbor in every case, in return for improved speed or memory savings. Often such an algorithm will find the nearest neighbor in a majority of cases, but this depends strongly on the dataset being queried [7].

D. K-d Tree

A k-d tree, or k-dimensional tree, is a data structure used for organizing some number of points in a space with k dimensions. It is a binary search tree with other constraints imposed on it. K-d trees are very useful for range and nearest neighbour searches. The root-cell of this tree represents the entire simulation volume. The other cells represent rectangular sub-volumes that contain the mass, center-of-mass, and quadrupole moment of their enclosed regions. It was one of the early structures used for indexing in multiple dimensions. Each level of K-d tree partitions the space into two partitions, the partitioning is done along one dimension of the node at the top level of the tree, along another dimension in nodes at the next level, and so on, iterating through the dimensions. The partitioning proceeds in such a way that, at each node, approximately one half of the points stored in the subtree fall on one side, and one half fall on the other. Partitioning stops when a node has less than a given maximum number of points [8]. Their purpose is always to hierarchically decompose space into a relatively small number of cells such that no cell contains too many input objects. This provides a fast way to access any input object by position. We traverse down the hierarchy until we find the cell containing the object. Typical algorithms construct k-d trees by partitioning point sets recursively along with different dimensions. Each node in the tree is defined by a plane through one of the dimensions that partitions the set of points into left/right (or up/down) sets, each with half the points of the parent node. These children are again partitioned into equal halves, using planes through a different dimension. Partitioning stops after log n levels, with each point in its own leaf cell.

E. Brute Force:

Approximate k-nearestneighbour (kNN) search using a brute force approach as well as with the help of the kd-tree will be used to reach of the main objective of this research (i.e. to speed-up K-nearest neighbour searches). Brute-force search or exhaustive search is a very general problem-solving technique that consists of systematically enumerating all possible candidates for the solution and checking whether each candidate satisfies the problem's statement.

II. LITERATURE REVIEW

R. Panigrahy [10]: In this paper, a simple modification to the kd-tree search algorithm for nearest neighbor search resulting in an improved performance. The Kd-tree data structure seems to work well in finding nearest neighbors in low dimensions but its performance degrades even if the number of dimensions increases to more than three. Since the exact nearest neighbor search problem suffers from the curse of dimensionality we focus on approximate solutions; a c-approximate nearest neighbor is any neighbor within distance at most c times the distance to the nearest neighbor. One of the earliest data structures proposed for this problem that is still the most commonly used is the kd-tree that is essentially a hierarchical decomposition of space a long different dimensions. For low dimensions this structure can be used for answering nearest neighbor queries in logarithmic time and linear space. Their purpose is always to hierarchically decompose space into a relatively small number of cells such that no cell contains too many input objects. This provides a fast way to access any input object by position. We traverse down the hierarchy until we find the cell containing the object.

Böhm C., Braunmüller B., Krebs F., Kriegel H.-P [11]: In this paper, have analyzed the similarity join has become an important database primitive to support similarity search and data mining. A similarity join combines two sets of complex objects such that the result contains all pairs of similar objects. Well-known are two types of the similarity join, the distance range join where the user defines a distance threshold for the join, and the closest point query or k-distance join which retrieves the k most similar pairs. In this paper, we investigate an important, third similarity join operation called k-nearest neighbor join which combines each point of one point set with its k nearest neighbors in the other set. It has been shown that many standard algorithms of Knowledge Discovery in Databases (KDD) such as k-means and k-medoid clustering, nearest neighbor classification, data cleansing, post processing of sampling-based data mining etc. can be implemented on top of the k-nn join operation to achieve performance improvements without affecting the quality of the result of these algorithms. We propose a new algorithm to compute the k-nearest neighbor join using the multipage index (MuX), a specialized index structure for the similarity join. To reduce both CPU and I/O cost, we develop optimal loading and processing strategies.

JOHN PIETER [12]: In this paper, Multistep processing is commonly used for nearest neighbor (NN) and similarity search in applications involving high-dimensional data and costly distance computations. Today, many such applications require a proof of result correctness. In this setting, clients issue NN queries to a server that maintains a database signed by a trusted authority. The server returns the NN set along with supplementary information that permits result verification using the data set signature. An adaptation of the multistep NN algorithm incurs prohibitive network overhead due to the transmission of false hits, i.e., records that are not in the NN set, but are nevertheless necessary for its verification. In order to alleviate this problem, This paper present a novel technique that reduces the size of each false hit and generalize solution for a distributed setting, where the database is horizontally partitioned over several servers.

David Hand, Heikki Mannila, and Padhraic Smyth [13]: In this paper, the first algorithm we shall investigate is the k-nearest neighbor algorithm, which is most often used for classification, although it can also be used for estimation and prediction. K-Nearest neighbor is an example of instance-based learning, in which the training data set is stored, so that a classification for a new unclassified record may be found simply by comparing it to the most similar records in the training set.

Weinberger, K.Q., Saul, L.K. [14]: In this paper, a distance based classification is one of the popular methods for classifying instances using a point-to-point distance based on the nearest neighbour or k -NEAREST NEIGHBOUR (k-NN). The representation of distance measure can be one of the various measures available (e.g. Euclidean distance, Manhattan distance or other specific distance measures). In this paper, we propose a modified nearest neighbour method called Nearest Neighbour Distance Matrix (NNDM) for classification based on unsupervised and supervised distance matrix. In the proposed NNDM method, an Euclidean distance method coupled with a distance loss function is used to create a distance matrix. In our approach, distances of each instance to the rest of the training instances data will be used to create the training distance matrix(TADM). Then, the TADM will be used to classify a new instance. In supervised NNDM, two instances that belong to different classes will be pushed apart from each other. This is to ensure that the instances that are located next to each other belong to the same class.

A.N.Pathak [15]: In this paper, Data mining is a field of database application that searches for unknown patterns in data that can be used to predict future behavior. Basically data mining is a technique not to change the presentation but to discover unknown relationships between the data. Data mining is termed as software, which is used to describe data in a new way, which is not true.

III. CONCLUSION

The use of the approximate k-nearest neighbour with k-d Tree data structure and comparing its performance to the brute-force approach. In the future, comparison between the performance of both, k-d tree and brute-force approach. In approximate nearest neighbour to evaluate and compare the efficiency of the data structure when applied on a particular data set size, distance and execution time. The proposed work will concentrate on comparison between two techniques and select the best one.

IV. ACKNOWLEDGEMENT

I would like to thank my guide Ms. Namita Kakkar, Assistant Professor in Department of Computer Science and Engineering at R.B.I.E.B.T. (Kharar), Punjab, India for motivating me to work on Approximate Nearest Neighbour using Data Mining. I would also like to thank my mother, my father and my siblings for their continuous support, cooperation, and guidance throughout my M.Tech. work. Moreover I would also like to thank my Professors who were always there at the need of the hour and provided with all the help and facilities, which I required, for my thesis work.

V. REFERENCES

[1]. Ian H. Witten and Eibe Frank, *Data Mining- Practical Machine Learning Tools and Techniques*- second Edition.

[2]. (1996) Technology Note prepared for Management 274A "at Anderson Graduate School of Management at UCLA, Bill Placce, Retrieved from "http://www.anderson.ucla.edu/faculty/jason.frand/teacher/technologies/palace/datamining.html" .

[3]. Oracle® Data Mining Concepts|11g Release1(11.1) Retrieved from "http://docs.oracle.com/ cd/B28359_01/ datamine.11b /b28129 /process.html".

[4]. en.wikipedia.org/wiki/Nearest_neighbor_search

[5]. Nitin Bhatia, Vandana, (2010) "Survey of Nearest Neighbor Techniques", International Journal of Computer Science and Information Security, Vol. 8, No. 2.

[6]. Anoop Jain , Parag Sarda , & Jayant R. Haritsa, (2003) "Providing Diversity in K-Nearest Neighbour Query", Tech. Report TR-2003-04.

[7]. S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman and A. Wu, (1998) "An optimal algorithm for approximate nearest neighbor searching", Journal of the ACM, 45(6):891-923.

[8]. William R. Mark, Gordon Stoll, (2006) "Fast kd-tree Construction with an Adaptive Error-Bounded Heuristic", Warren Hunt, IEEE Symposium on Interactive Ray Tracing.

[9]. Steven S. Skiena, (2010) "The Algorithm Design Manual" , 2nd Edition, Stony Brook, NY 11794-4400.

[10]. R. Panigrahy. Entropy based nearest neighbor search in high dimensions. In SODA '06: Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm (Miami, FL, 2006), pages 1186–1195. ACM Press, New York, NY, 2006.

[11]. Böhm C., Braunmüller B., Krebs F., Kriegel H.-P.: Epsilon Grid Order: An Algorithm for the Similarity Join on Massive High-Dimensional Data, ACM SIGMOD Int. Conf. on Management of Data, 2001.

[12]. JOHN PIETER Authenticated Multistep Nearest Neighbour Search in Data mining International Journal of Communications and Engineering Volume 01– No., Issue: 02 March2012.

[13]. David Hand, Heikki Mannila, and Padhraic Smyth, 2001, Principles of Data Mining, MIT Press, Cambridge, MA, 2001.

[14]. Weinberger, K.Q., Saul, L.K.: Distance Metric Learning for Large Margin NearestNeighbor Classification. Journal of Machine Learning Research 10, 207–244 (2009).

[15]. A.N.Pathak, Manu Sehgal, Divya Christopher: A Study on Selective Data Mining Algorithms, IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 2, March 2011.