International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 10 Issue 6 ǁ 2022 ǁ PP. 259-261

K-Dimensional Tree using coresets for KNN based Classification

Bannela Siva Kumar[1]
PG Scholar
Department of Computer Applications
Madanapalle Institute of Technology & Science, India

Dr.R.Maruthamuthu[2]
Assistance Professor
Department of Computer applications
[1,2]Master of Computer Applications, Madanapalle Institute of Technology and Science, India.

Abstract:
KNN (K Nearest-neighbor Classification) is a lazy learning classification technique that only remembers the training dataset and does not provide a defined discriminative function. KNN's prediction phase takes a long time since it searches for a target's nearest neighbor(s) over the whole training set. The KD-tree (K Dimensional-tree) is a multi-dimensional binary tree that is used to efficiently represent training data as a storage structure. As a result, the research combines the advantages of KNN and KD-tree to present KNN-KD-tree, a novel classification technique. Eleven datasets were used to conduct the tests. Experiments have demonstrated that the suggested KNN-KD-tree technique can greatly improve search performance while reducing time complexity.
Keyword— *Lazy Learning Classification Algorithm, KNN, KDtree, KNN-KD-tree.*

Date of Submission: 25-05-2022 Date of acceptance: 05-06-2022

I. INTRODUCTION:

Data classification is an essential subject in data mining [1] because it can categorise fresh samples using previously collected data. KNN is a statistical approach for classification and regression proposed by Cover and Hart [2], and it is one of the most basic methods in data mining classification technology. KNN, on the other hand, is a slow learning strategy. Each test sample necessitates a significant amount of calculation, as well as a high memory cost. Meanwhile, the distance formula used by the KNN algorithm has an impact on the final classification accuracy. Two significant factors have affected KNN's performance. The first is the influence of high dimensions on distance measurement, which reveals that as the number of variables increases, so does the discrimination ability of distance measurement. The enhanced KNN algorithm may be grouped into three categories, according to a survey of the literature: change of distance measurement method, optimization of K parameter method [3, 4], and data sample pre-processing. [5] The literature suggests that weighting features based on their importance enhances classification accuracy. significance. To limit the influence of the K parameter on classification accuracy, the literature classifies the dataset with many alternative K values and trains the sample datasets using BP neural networks. However, KNN's temporal complexity grew as a result of these new methods. Furthermore, the KD-tree [6] is a data structure for organising points in a k-dimensional space [7]. KD-trees are a helpful data structure. a structure that can be used for a multitude of purposes KD-tree is a hybrid point cloud data storage structure that can successfully explore the fast neighbourhood in a vast data environment, according to literature [8].

Based on the aforesaid research, this paper presents a KNN-KDtree classification technique. The algorithm's core ideas and implementation procedures are discussed in Section III. In parts IV and V, the classification accuracy and temporal complexity of KNN and KNN-KD-tree under numerous data sets are explored. A conclusion finishes Section VI.

II. LITERATURE REVIEW

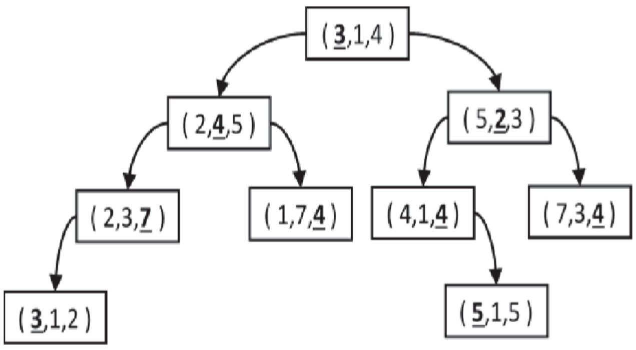
A. **Classification based on KNN**:

The KNN algorithm uses the distance metric function to calculate the distance between the sample to be categorised x and each sample in the training set, sort the calculated distance, and choose the k training samples closest to x as the k nearest neighbours of x.

The representative classification sample x is categorised into a category if the majority of the k nearest neighbours belong to that class. The Euclidean distance between a and b is defined as follows: x x a(a1, a2, xan), x x b(b1, b2, xbn), where a and b are two n-dimensional vectors.

KD-tree:

A hyper-plane can divide each non-leaf node of a KD-tree into two subspaces, and each subspace can be recursively divided in the same way. Left subspace and right subspace, or upper subspace and lower subspace, are the two components of every subspace. A partition of the K dimensional space created by a K dimensional data collection is represented by constructing a KD-tree on it. Each node in the tree, in other words, corresponds to a K-dimensional hyper rectangular area.



A three-dimensional binary tree is shown in Figure 1. It should be a space division. For ease of observation, Figure 1 illustrates the tree in planar depiction. The dimension of the bold number in each node represents the split axis. In one dimension, it can be observed that it is consistent with the binary tree properties:
It has property 1 if its left subtree is not empty. The value of none of the nodes on the left subtree is greater than the root node.

Property 2: If the right subtree isn't empty, then all nodes on the right subtree have values greater than the root node's value.

Property 3 is also included, as well as its left and right subtrees.

III. PROPOSED METHOD

Insert, remove, and search are all supported by KD-tree as a binary storage structure, but it lacks the capacity to recognise characteristics and labels. Each node has a tag value associated with it, but the tag is not involved in the axis selection or division. Make a classification algorithm out of it and save it as a KNN-KD tree.
ADVANTAGES: • Increased Accuracy
• Characterization has a low degree of volatility.
• There are few, if any, biases due to assumptions about the dataset, and the execution time is fast.

IV.CONCLUSION

When employed with coresets, each dataset displayed competitive or, in some cases, higher accuracy for at least one value of m, as shown in the tables above (default credit card and HTRU2). The coreset size is substantially smaller than the original dataset size for larger datasets like bio train and MiniBooNE. Despite this, they produce virtually identical accuracy results as the original dataset. In both the offline (indexing) and online (query) stages, KD-Tree constructed on the coresets of these datasets enjoys a significant speed gain. We can also see that when the dataset size lowers, the time difference between indexing and querying shrinks. This could be true even for smaller datasets (spambase and HTRU2).

REFERENCES

[1]. Wenfeng Hou, Daiwei Li, Chao Xu, Haiqing Zhang and Tianrui Li: An Advanced k Nearest Neighbor Classiffication Algorithm Based on KD-tree. IEEE International Conference of Safety Produce Informatization (IICSPI), 2018.

[2]. Chi-Chun Huang and Hsin-Yun Chang: A Novel SVM-based Reduced NN Classiffication Method. 11th International Conference on Computational Intelligence and Security, 2015.

[3]. Songrit Maneewongvatana and David M. Mount: Its okay to be skinny, if your friends are fat. 4th Annual CGC Workshop on Comptutational Geometry, 1999.

[4]. Mario Lucic, Olivier Bachem and Andreas Krause. :Strong Coresets for Hard and Soft Bregman Clustering with Applications to Exponential Family Mixtures. https://arxiv.org/abs/1508.05243

[5]. O. Bachem, M. Lucic, A. Krause.: Scalable k-Means Clustering via Lightweight Coresets. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), 2018.

[6]. V. Hyvnen and T. Pitknen and S. Tasoulis and E. Jsaari and R. Tuomainen and L. Wang and J. Corander and T. Roos.: Fast nearest neighbor search through sparse random projections and voting. 2016 IEEE International Conference on Big Data (Big Data)
Spatial KDTree Class, https://docs.scipy.org/doc/scipy-0.14.0/reference/generated/scipy.spatial.KDTree.html

[7]. Datases,https://archive.ics.uci.edu/ml/datasets.php,osmot.cs.cornell.edu./kddcup/datasets.html