Improving the Usability of gaKnn Framework

Thimal Kempitiya

Department of Computer Science & Engineering, University of Moratuwa
Sri Lanka
thimal.10@cse.mrt.ac.lk

Abstract— **K nearest neighbour classification (KNN) is a popular non parametric and lazy algorithm for classification. gaKnn framework is a implementation of the KNN algorithm combine with genetic algorithm. It provides genetic algorithm optimization for KNN algorithm which will optimize the weight values for each attribute and k value. In this paper, I proposed improvements for the current implementation of the gaKnn framework to improve its usability and performance using kd tree to improve the KNN algorithm, different data and file type usage and regression algorithm based on k nearest neighbour. Mainly it introduce three modules for the current implementation of the gaKnn framework namely csv file reader and writer module, large dataset module and KNN regression module.**

Keywords*— *k nearest neighbour, regression, kd tree

I. Introduction

K nearest neighbour classifier used a data point's k nearest neighbours according to all of its attributes to select classification for it. KNN is a lazy type classification, so it will not take long time to predict and train. Performance of this algorithm depends on the k value select and the weights give to attribute when selecting k neighbours [1].

gaKnn [2] framework is one of the significant implementation of the this feature using genetic algorithm, but current implementation has issues in usability and performance.

Current implementation supports only the arff data format which has a lot of metadata at beginning of the dataset. But this is not a popular file format. So before used with this framework data need to convert to this data format. Csv (comma separated values) file format is good solution to this problem. But it will not provide any prior details about dataset need to process data to find them.

KNN classifier can be implementing in three different approaches brute force approach, kd tree and ball tree. First method gives good performance with the small datasets and second two give best performance with large datasets. Current implementation of gaKnn use brute force implementation of KNN which checks each and every data element to find k nearest neighbours. When classifying it need to deal efficiently with large dataset same as small datasets. Kd tree implementation of KNN will reduce the time about 50% from datasets which has over 20,000 data and attributes less than 20.

KNN has been popular as a classification algorithm, but researches are carried out to use it as a regression algorithm [3].Genetic algorithms can be used to optimize the KNN regression similar to the classifier which is not in the current implementation of gaKnn.

My implementation provides these usability and performance improvement to the current implementation of the gaKnn framework along with the refactoring of the current code

II. Literature review

KNN is a widely used classification problem where selection of the optimum K as the number of neighbours can be considered as optimization problem. Optimization parameters are k value, weight vector, voting power of neighbours, attribute selection and instance selection. Finding these values is search problem with large search space. Genetic algorithm provides optimum solution for search problems with large search space. gaknn framework provides this solution.

According to the research "Approximate K-Nearest Neighbour Based Spatial Clustering Using K-d Tree" [4] by Dr. Mohommed Otair spatial clustering using KNN gives better performance with the kd tree compared to the brute force method. Spatial data include spatial information and non-spatial attribute data kd tree data structure and its variants is the most popular data structure for searches in multidimensional space. For the clustering of the spatial data it can be used brute force method and but with large dataset of spatial data it will significantly slow the computation. It will show significant time reduction when using kd tree with increase of the k value and number of data elements.

The research unsupervised k-nearest neighbour regression proposed KNN regression as a good heuristic for dimensionality reduction. It tests two variants of unsupervised k nearest neighbour regression (UNN) with locally linear embedding and two UNN variations give lower data space reconstruction error. It can conclude that k nearest neighbour is also an important algorithm for regression.

III. Design and Implementation

This section proposed usability improvements for the current implementation of the gaKnn framework. It has three main sections. The three main sections introduced are csv reader/writer module, large dataset module and the regression module. These three modules are integrated to the current framework, so user applications built using this can have advantage of these modules.

A. *Csv Reader and Writer Module*

Csv (comma separated values) is commonly used data representation format for machine learning and data mining. Many of the data mining and machine learning algorithms need metadata of data file apart from the core data, but csv files does not provide the meta data need to process. Compared csv files arff file format provide most of the necessary metadata need to process. To create a csv reader and writer module it needs to support the framework's existing data structure. The existing data structure needed the metadata of data set at the creation time of the data structure.

Solution proposed here is to create data structure without specifying metadata and when reading the csv data set whenever new metadata of the data set found, add it to the data structure and updates data structure.

B. *Large Dataset Module*

Brute force method of KNN will work efficiently with the small datasets and for large data sets introduces the kd tree implementation of KNN.

Dataset is put into a kd tree data structure and find the k nearest neighbours.

1) *Kd Tree:*

K-Dimensional tree (kd-tree) [6] is a data structure for storing special data in k-dimensional space which was invented by Jon Bentley in 1975. This is very useful data structure for operations on multi-dimensional data. Kd tree mainly use for range searching and nearest neighbour searching

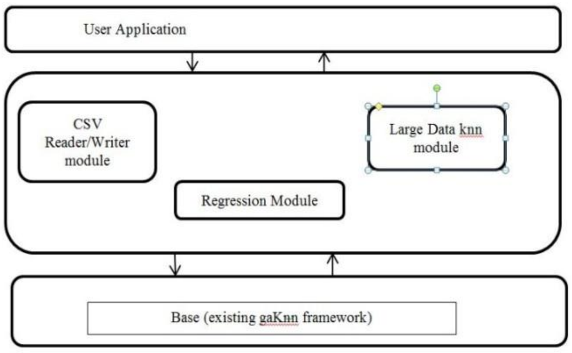

Fig. 1 High Level structure of the Improved Framework.

2) *Kd Tree construction:*

Kd tree is a binary tree which has k dimensional nodes. Each node needs to keep information regarding split which is an integer to specify dimensions, left child and right child which are nodes to keep left and right sub trees respectively and domain values (vector) to keep the attribute values.

Each non-leaf node is split into sub-spaces by a splitting hyper plane according to the split. This hyper plane goes through the selected dimension of node and perpendicular to the direction specified by the split field of node. Splitting plane will change alternatively and we can get the value of splitting plane by N mod k, where N is the depth of tree and k is the number of dimension. Points which have less value for dimension specified by splitting field compare to dimension specified by splitting field of node there add to left sub-tree and the points added to right sub-tree.

Here is an example to understand a kd tree of dimension 2 we have points which have x and y values (3, 6), (2, 8), (6, 7), (1, 6), (4, 5), (5, 10) we put (3,6) as root node and start splitting with x values (2,8) and (1,6) have less values than (3,6) x value (3) these two points are in the left sub-tree of root node and for point (2,8) is split the space by y value (1,6) come to the left sub-tree of (2,8) node and there are no right sub-tree. (6,7), (4,5), (5,10) points have greater x value than 3 so there are at the right sub-tree of root. Next point (6,7) become the root of right sub-tree and it split space according to the y value. (4,5) has less y value than the (6,7) so it added to left sub-tree and (5,10) has greater y value than the (6,7) it added to right sub-tree.

At point (3,6) 2d plane is divided to sub-spaces at x=2, left subspace have points (2,8) and (1,6). Right subspace have points (6,7), (4,5) and (5,10). Next at point (2,8) by y value 8 its space is divided to sub-spaces upper sup-space which is empty and lower subspace which has point (1,6). Next right sup-space is divided at point (6,7) by its y value 7. Upper sub-space has point (5,10) and lower sub-space has point (4,5).

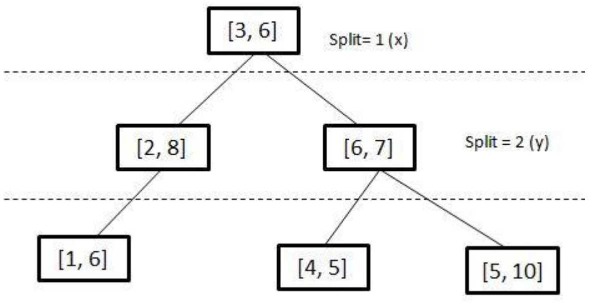

Fig. 2 2-D Kd Tree Structure.

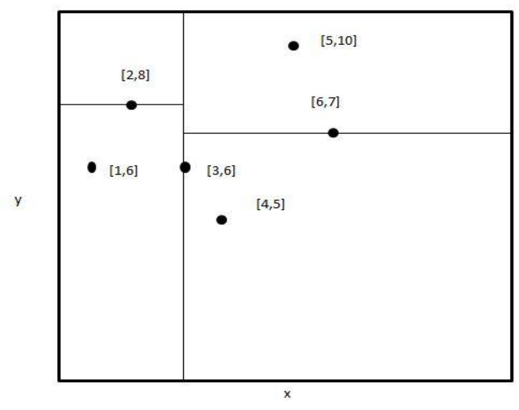

Fig. 3 kd tree decomposition for 2-D data elements

3) Nearest Neighbour Search:

In kd tree nearest neighbour search can be done in O(logn) average time and in worst case O(n). (n is the number of data elements) In nearest neighbour search find element that is nearest to the searching element, what it means is it search element that has least distance. Mostly we use Euclidean distance in this.

Nearest neighbour search algorithm get one parameter one point which we need to find a point near to that in kd tree. Search start at the root and add the least distance is set to root node distance and search both left sub-tree and right sub-tree. When it found a node which has distance less than current distance it replace the current least distance and any sub-tree's shortest distance is greater than current least distance whole sub-tree is pruned.

Weka library is used for implementation of kd tree data structure with modifications to use weight values.

C. *KNN Regression Module*

This module gives implementation to the K nearest neighbour regression, which is a method some of the researched carried out. It proved to be successful for dimensionality reduction. This implementation is same as the KNN classifier optimized with the genetic algorithm and optimized the weight values for each attribute and k value to use.

KNN regression is a continuous function where the x Є R^n and y Є R. This function finds the mean of the k neighbours.

$$f_{knn}(x) = \sum_{i \in N_K(x)}^{n} y_i \qquad (1)$$

f_{knn} is the KNN Regression function and it take input as x vector of R^n, K is the number of neighbours to consider, $N_K()$ gives the k nearest neighbours of the x data tuple. And y_i is the output value of ith neighbour.
For some datasets this may not give better performances, but this is a fairly fast regression method, training is needed only for optimization of k value and weight values same as the KNN classifier. It can be used to get approximated values.

IV. Results

Testing of the usability improved gaKnn framework is done using Amazon company employee detail data set, it needs to predict whether grant or revoke the employees access. This dataset has 9 attributes. Experiments have been performed on 2.30 GHz PC with intel core i5 processer and 3Gb memory.

Training data set has 32769 data tuples and test data set has58921 data tuples. Two methods were test with first 200, 1000, 5000, 20000, 50000 data elements and results are given below. At first three point's execution time different are less than 2 times in brute force compare to kd tree, but after 5000 it become more than 2 times. So the with large data sets kd tree give better performances.

Csv file reader and writer and the KNN regression modules are test using unit test. Amazon data set is in csv file format and it gave successful results in using csv files.

KNN regression module is tested with the abalone dataset which have the 10 attributes and predict the age of abalone. The results from the KNN regression for this dataset less accurate compare to other methods regression trees. It needs to be test with the other datasets get better understand.

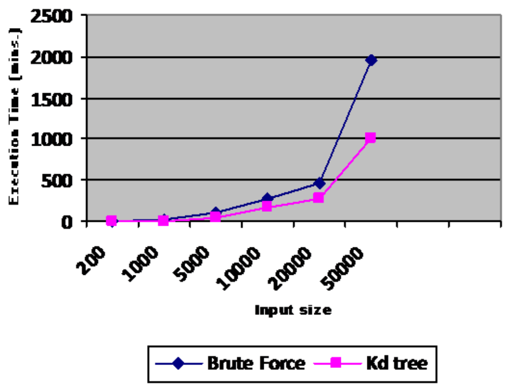

Fig. 4 kd tree vs. brute force performance with different data sizes

TABLE I
KD TREE VS BRUTE FORCE TEST RESULTS in MINUTES

	200	1000	5000	10000	20000	50000
BF	3.92	19.28	94.93	271.7	462.1	1953
KD tree	2.49	10.43	50.60	167.1	267.4	1013

V. Conclusions

The major contribution achieved by this research is the usability and performance improvement of the current implementation of the gaKnn framework. I introduced three major improvements namely csv file format reading, large dataset KNN implementation and KNN regression module. According to the results, csv file reading and big dataset module which were implemented using kd tree shows better improvements to the framework. The KNN regression module was successful as an implementation, but as a regression algorithm gives poor performance and needs to test with other datasets.

REFERENCES

[1] H. Altay Guvenir and A. Akkus, “Weighted k nearest neighbor classification on feature projections,” Ankara, Turkey.

[2] D.G.N Dayaratne, “Genetic algorithm optimized k nearest neighbor classification framework (gaKnn),” Degree of MSc in computer science thesis, University of Moratuwa, Sri Lanka, Nov. 2008.

[3] O. Krmer, “Unsupervised K nearest Neighbor Regression,” Oldenburg, Germany, Sep. 2011.

[4] Dr. M. Otair, “Approximate k- nearest neighbor based spatial clustering using k-d tree,” Int. Journal of Database Management Systems, Amman, Jordan, Feb. 2013 vol. 5, No. 1.K. Elissa, “Title of paper if known,” unpublished.

[5] L. Baoli, Y. Shiwen, and L. Qin, “An improved k-nearest neighbor algorithm for text categorization,” Proc. Of thhe 20th Int. conf. on Computer Processing of oriental Languages, Shenyang, China, 2003.

[6] (2013) Wikipedia kd tree page. [Online]. Available: http://en.wikipedia.org/wiki/K-d_tree