Report

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

Locally Adaptive Nearest Neighbours Algorithms

Monsoon'22

Team 19:

Likhith Asapu (2020114015)

Vikram Rao (2020101123)

Nitin Rajasekar (2020101117)

Joel Alex (2020101118)

Abstract

We introduce and compare four versions of a k-nearest neighbour algorithm with locally adaptive k to the basic k-nearest neighbour algorithm (kNN). Locally adaptive kNN algorithms determine the value of k to be used to classify a query by consulting the results of cross-validation computations in the query's local neighbourhood. In general, results from kNN are highly comparable to results from standard kNN. However, very encouraging results are observed in specific applications with uniquely constructed databases.

Contents

1. Introduction

1.1 Literature Survey

The paper on Locally Adaptive Nearest Neighbours [1] was thoroughly read and analysed. The disadvantages of the standard kNN algorithm was outlined and 4 locally adaptive methods to vary k was proposed:

- **localKNN$_{\textbf{k's unrestricted}}$**: All of the training examples are also saved by this algorithm. It stores a list of k values that correctly classify each training example under leave-one-out cross-validation alongside each training example. To classify a query q, the query's M nearest neighbours are computed, and the k that correctly classifies the majority of these M neighbours is determined. This is denoted as $k_{M,q}$. The class of the majority of the query q's $k_{M,q}$ nearest neighbours is then assigned to the query q. It should be noted that $k_{M,q}$ can be greater or less than M. The algorithm's only parameter is M, which can be determined through cross-validation.
- **localKNN$_{\textbf{k's pruned}}$**: The list of k values for each training example typically contains a large number of values. A global histogram of k values is generated, and k values that appear less than L times are removed from all lists (at least one k value must remain in each list). Cross-validation can be used to estimate the parameter L. The classification of queries is the same as with $localKNN_{k's\,unrestricted}$.
- **localKNN$_{\textbf{one k per class}}$**: Each individual class is assigned a 'k' value. The value is assigned based on how many members of a class are correctly classified for a certain 'k'. For a class 'C', which was assigned the value 'k', the percentage of the 'k' nearest neighbors to a given query 'q' that belong to 'C' is computed. The query data point is then assigned to the class with the highest such percentage.
- **localKNN$_{\textbf{one k per cluster}}$**:Firstly, clusters are determined within the data using an unsupervised clustering algorithm. Similar to the previous algorithm, a single value of k is determined for each cluster. Based on which cluster the given query 'q' is assigned to, the corresponding value of k is used for classification.

1.2 Implementation

1. Standard datasets are selected to test and compare the accuracies of the different classifiers. Some special datasets are also generated by combining datasets to test and compare effectiveness of locally adaptive algorithms.
2. Each algorithm is run on the datasets and optimal values for k are computed according to the algorithm. Accuracies are also calculated and compared for each algorithm for the different databases.

2. Dataset Selection

2.1 Databases and features

We have selected the same databases as the research papers, i.e. the datasets obtained from the UC-Irvine repository of machine learning databases, for initial performance comparision:

- **Wine** [5]: The result of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents(13 attributes) found in each of the three types of wines(3 classes). The dataset contains 178 tuples.
- **Voting** [4]: This data set includes votes for each of the U.S. House of Representatives Congressmen on the 16 key votes identified by the CQA. The dataset has 17 unique attributes, 2 classes (democrat or republican) and 435 tuples.
- **Glass** [3]: A Glass Identification Data Set from UCI. It contains 10 attributes including id. The response is glass type(discrete 7 values). The dataset contains 214 tuples
- **Iris** [2]: Used in R.A. Fisher's classic 1936 paper, The Use of Multiple Measurements in Taxonomic Problems, and can also be found on the UCI Machine Learning Repository. It includes three iris species with 50 samples each as well as some properties about each flower. One flower species is linearly separable from the other two, but the other two are not linearly separable from each other. The dataset contains 5 unique attributes.
- **Mobile Price** [6]: The data set consists of 2000 tuples and 20 features. The objects are classified into four different classes based on their price range.

Three more databases from the UC-Irvine repository of machine learning databases are used to calculate accuracies seperately as well as combined in order to determine any improvements of the locally adaptive nearest neighbour algorithms over the standard algorithms:

- **LED** [7]: This simple domain contains 7 Boolean attributes and 10 concepts, the set of decimal digits. In this case, each attribute value has the 10% probability of having its value inverted.
- **Waveform** [8]: The dataset contains 3 classes of waves and 21 attributes, all of which include noise. Each class is generated from a combination of 2 of 3 "base" waves.
- **Letter Recognition** [9]: Black-and-white rectangular pixel displays needs to be classified as one of the 26 capital letters in the English alphabet. The character images were converted into 16 primitive numerical attributes (statistical moments and edge counts) which were then scaled to fit into a range of integer values from 0 through 15. It contains around 20000 items in the database.

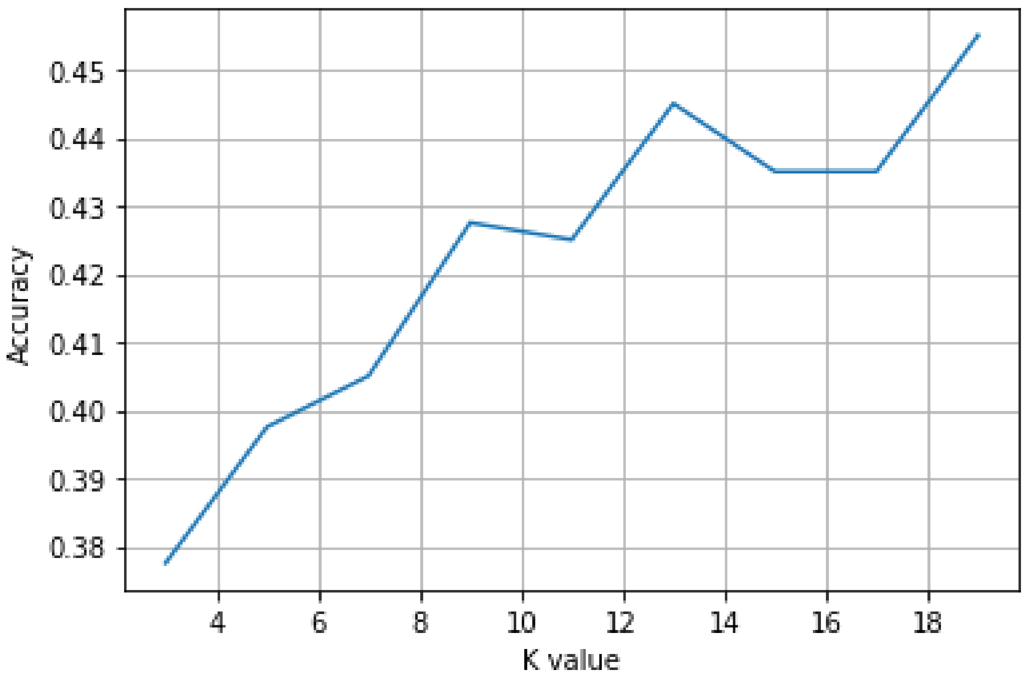
2.2 Database Preprocessing

1. **Normalization**: All the attributes of the databases are normalized (Min-max normalization) in order to prevent skew of data that might effect accuracy of classifiers.

2. **Train-Test Split**: We split each of the datasets into 80% training data and 20% test data. The training data is used to train each of the Nearest Neighbour classifiers and the Test Data is used to calculate the accuracy of each trained classifier.

3. K Nearest Neighbours

K Nearest Neighbours is implemented from scratch and code is hosted on Github [10]. Simultaneously, built-in KNN algorithm from Sklearn [11] is also run and accuracies of both classifiers are compared.
The values of K are varied and accuracies are calculated to determine optimal value of K for the mobile price database.



Observations:

- The accuracy of the classifier shows an increase with increase in value of k. From the graph we see that optimal value of k is around 20.
- The average accuracy of the Constructed KNN classifier comes out to be around 59% which matches the Sklearn kNN classifier.

4. localKNN$_{k's\,unrestricted}$

Locally adaptive K Nearest Neighbours with k value unrestricted is implemented from scratch and code is hosted on Github [10]. All of the training examples are saved by the algorithm and it stores a list of k values that correctly classify each training example under leave-one-out cross-validation alongside each training example. To classify a query q, the query's M nearest neighbours are computed, and the k that correctly classifies the majority of these M neighbours is determined. This is denoted as $k_{M,q}$. The class of the majority of the query q's $k_{M,q}$ nearest neighbours is then assigned to the query q. It should be noted that $k_{M,q}$ can be greater or less than M. The algorithm's only parameter is M, which can be determined through cross-validation.
Accuracies determined for the various databases are as follows:

- **Iris**: 0.9666666666666667 (With M=3)
- **Glass**: 0.9534883720930233 (With M=7)
- **Wine**: 0.9722222222222222 (With M=3)
- **Mobile Price**: 0.4475 (With M=17)
- **LED**: 0.75 (With M=3)
- **House Votes**: 0.896551724137931 (With M=5)

Two pairs of databases are also combined and accuracies are determined for them as well:

- **Glass-Mobile Combined**: 0.510158013544018 (With M=17)
- **LED-Letter Combined**: 0.375 (With M=13)
- **Mobile-Wine Combined**: 0.463302752293578 (With M=9)

5. localKNN$_{k's\,pruned}$

Locally adaptive K Nearest Neighbours with k values pruned is implemented from scratch and code is hosted on Github [10]. The list of k values for each training example typically contains a large number of values. A global histogram of k values is generated, and k values that appear less than L times are removed from all lists (at least one k value must remain in each list). Cross-validation can be used to estimate the parameter L. The classification of queries is the same as with $localKNN_{k's\,unrestricted}$.
Accuracies determined for the various databases are as follows:

- **Iris**: 0.9666666666666667 (With M=3)
- **Glass**: 0.9534883720930233 (With M=7)
- **Wine**: 0.9722222222222222 (With M=3)
- **Mobile Price**: 0.4675 (With M=17)

- **LED**: 0.7 (With M=3)
- **House Votes**: 0.896551724137931 (With M=5)

Two pairs of databases are also combined and accuracies are determined for them as well:

- **Glass-Mobile Combined**: 0.5079006772009029 (With M=17)
- **LED-Letter Combined**: 0.35 (With M=13)
- **Mobile-Wine Combined**: 0.4105504587155963 (With M=9)

6. localKNN$_{one\ k\ per\ class}$

Locally adaptive K Nearest Neighbours with one k assigned per class is implemented from scratch and code is hosted on Github [10]. Each individual class is assigned a 'k' value. The value is assigned based on how many members of a class are correctly classified for a certain 'k'. For a class 'C', which was assigned the value 'k', the percentage of the 'k' nearest neighbors to a given query 'q' that belong to 'C' is computed. The query data point is then assigned to the class with the highest such percentage.
Accuracies determined for the various databases are as follows:

- **Iris**: 0.9666666666666667
- **Glass**: 0.9666666666666667
- **Wine**: 0.9722222222222222
- **Mobile Price**: 0.4325
- **LED**: 0.725
- **House Votes**: 0.896551724137931

Two pairs of databases are also combined and accuracies are determined for them as well:

- **Glass-Mobile Combined**: 0.4966139954853273
- **LED-Letter Combined**: 0.3875
- **Mobile-Wine Combined**: 0.4380733944954128

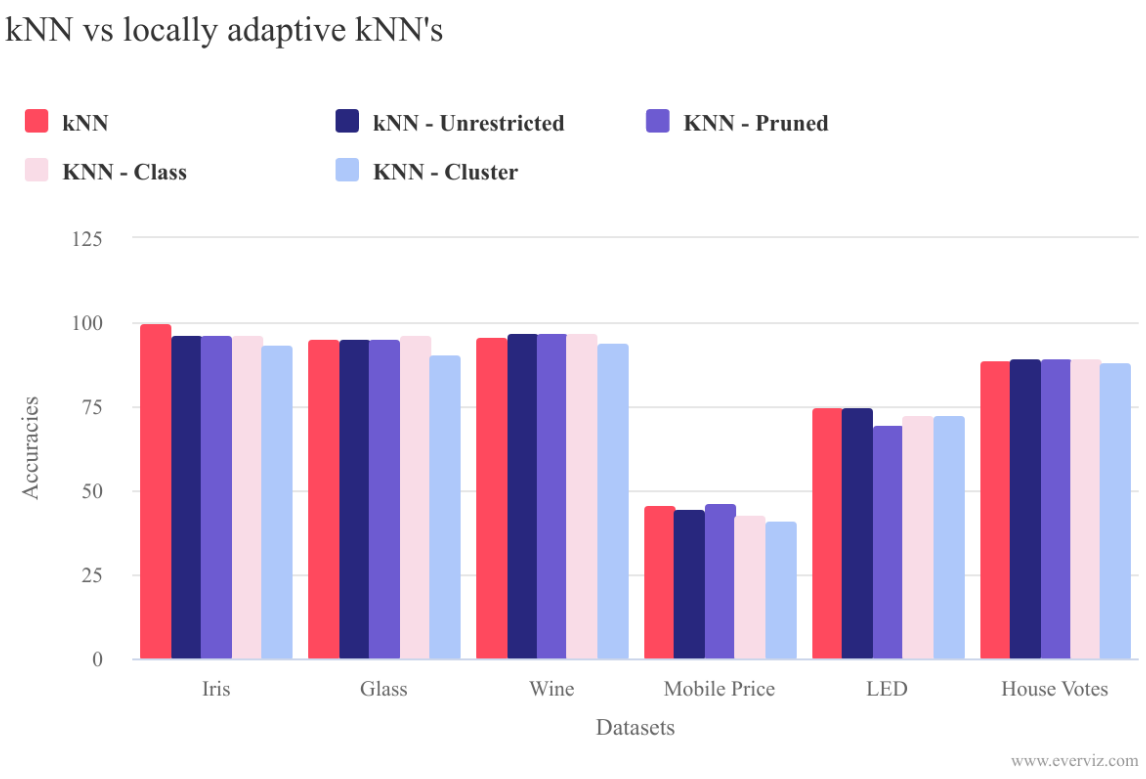
7. localKNN$_{one\ k\ per\ cluster}$

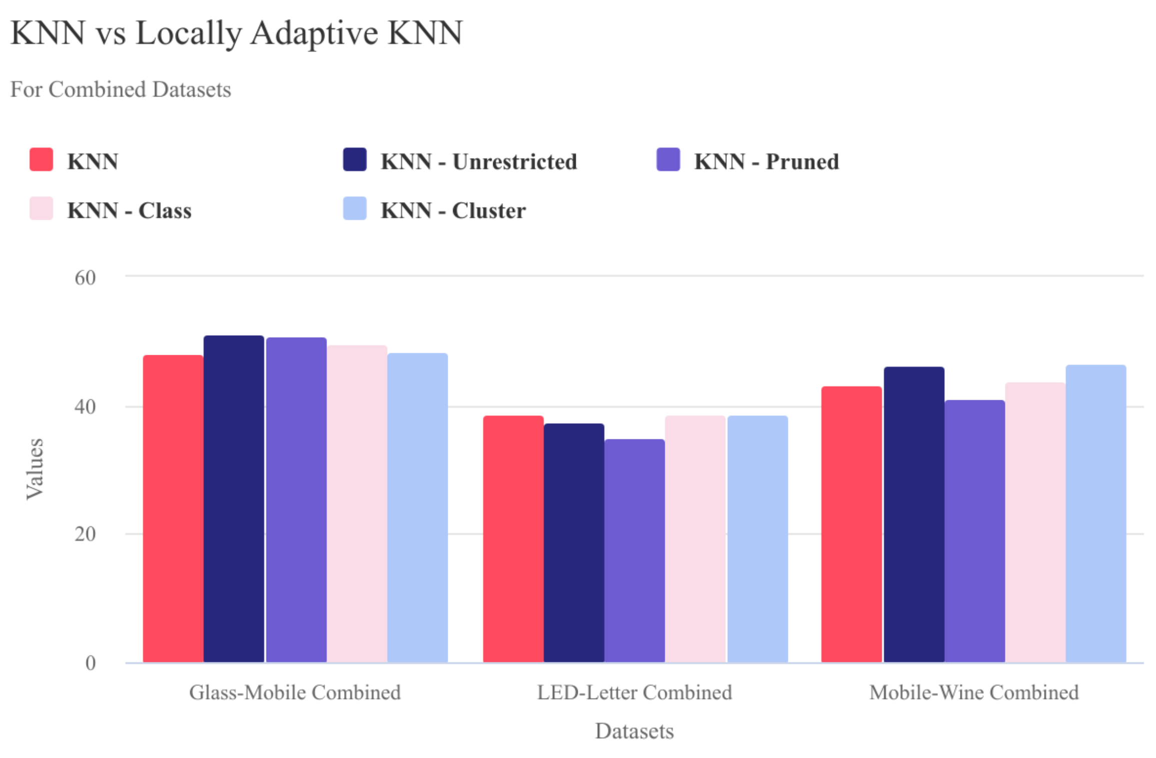
Locally adaptive K Nearest Neighbours with one k assigned per cluster is implemented from scratch and code is hosted on Github [10].Firstly, clusters are determined within the data using an unsupervised clustering algorithm. Similar to the previous algorithm, a single value of k is determined for each cluster. Based on which cluster the given query 'q' is assigned to, the corresponding value of k is used for classification.

- **Iris**: 0.9333333333333334
- **Glass**: 0.9069767441860465
- **Wine**: 0.9444444444444444
- **Mobile Price**: 0.4125
- **LED**: 0.725
- **House Votes**: 0.8850574712643677

Two pairs of databases are also combined and accuracies are determined for them as well:

- **Glass-Mobile Combined**: 0.4853273137697517
- **LED-Letter Combined**: 0.3875
- **Mobile-Wine Combined**: 0.4655963302752294





8. Work Distribution

- **Likhith**: Implemented KNN unrestricted, Documentation, Database Selection
- **Joel**: Implemented KNN pruned, Documentation, Database Selection
- **Nitin**: Implemented KNN one k per class and KNN, Documentation
- **Vikram**: Implemented KNN one k per cluster and KNN, Documentation

9. Conclusions

Upon observing the results of the locally adaptive Nearest Neighbours algorithms and comparing it with the standard kNN we arrive at the following conclusions:

1. For standard databases, the locally adaptive nearest neighbour algorithms perform approximately the same as normal K Nearest Neighbour algorithm with an optimal value of K selected.
2. For the combined database, LED-Letter recognition, the locally adaptive nearest neighbour algorithms perform approximately the same as normal K Nearest Neighbour algorithm with an optimal value of K selected since optimal value of k is very similar for each of the separate databases (LED-7 [7] and Letter Recognition [9]).

3. For the combined databases, Glass-Mobile Price and Mobile-Wine Combined, the locally adaptive nearest neighbour algorithms show a noticeable improvement over normal K Nearest Neighbour algorithm with an optimal value of K selected since there is significant variation in the input space and optimal value of k is very different for the separate databases (Glass [3] and Mobile Price [6] as well as Wine [5]).
4. Our observations align with the obseervations from the research paper tested over a wider range of databases.

10. Acknowledgements

Many thanks to the authors of the original research paper, Dietrich Wettschereck and Thomas G. Dietterich for a clear and concise presentation of novel variations of the Nearest Neighbour Algorithms. We also extend our thanks to the Course Professors and Teaching Assistants for helping us gain a clear understanding of all the Machine Learning Concepts used in the implementation of this project.

11. References

1. https://proceedings.neurips.cc/paper/1993/file/5f0f5e5f33945135b874349cfbed4fb9-pdf
2. https://www.kaggle.com/datasets/uciml/iris
3. https://www.kaggle.com/datasets/uciml/glass
4. https://www.kaggle.com/datasets/devvret/congressional-voting-records
5. https://www.kaggle.com/datasets/brynja/wineuci
6. https://www.kaggle.com/datasets/iabhishekofficial/mobile-price-classification
7. https://archive.ics.uci.edu/ml/datasets/LED+Display+Domain
8. https://archive.ics.uci.edu/ml/datasets/Waveform+Database+Generator+(Version+1)
9. https://archive.ics.uci.edu/ml/datasets/letter+recognition
10. https://github.com/viks01/SMAI-Project
11. https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html