

Introduction to kNN Classification and CNN Data Reduction

Oliver Sutton

February, 2012

- 1 The Classification Problem
 - Examples
 - The Problem
- 2 The k Nearest Neighbours Algorithm
 - The Nearest Neighbours Approach
 - The Idea
 - The Algorithm
 - Applet Example
 - Problems With the Algorithm
- 3 Condensed Nearest Neighbour Data Reduction
 - The Idea
 - The Algorithm
 - Example Using the Applet
- 4 Summary

The Classification Problem: Example 1

- ▶ Suppose we have a database of the characteristics of lots of different people, and their credit rating
 - ▶ e.g. how much they earn, whether they own their house, how old they are, etc.
- ▶ Want to be able to use this database to give a new person a credit rating
- ▶ Intuitively, we want to give similar credit ratings to similar people

The Classification Problem: Example 2

- ▶ We have a database of characteristic measurements from lots of different flowers, along with the type of flower
 - ▶ e.g. their height, their colour, their stamen size, etc.
- ▶ Want to be able to use this database to work out what type a new flower is, based on its measurements
- ▶ Again, we want to classify it with the type of flower it is most similar to

The Classification Problem

- ▶ In general, we start with a database of objects whose classification we already know
 - ▶ Known as the training database, since it trains us to know what the different types of things look like
- ▶ We take a new sample, and want to know what classification it should be
- ▶ The classification is based on which items of the training database the new sample is similar to

The Problem

What makes two items count as similar, and how do we measure similarity?

One way of solving these problems is with the nearest neighbours approach...

The Nearest Neighbours Idea: Measuring Similarity

- ▶ The nearest neighbours approach involves interpreting each entry in the database as a point in space
- ▶ Each of the characteristics is a different dimension, and the entry's value for that characteristic is its coordinate in that dimension
- ▶ Then, the similarity of two points is measured by the distance between them
 - ▶ Different metrics can be used, although in general they give different results

The Nearest Neighbours Idea: Sufficiently Similar

- ▶ The nearest neighbours approach then classifies the new sample by looking at the classifications of those closest to it
- ▶ In the k Nearest Neighbours (kNN), this is achieved by selecting the k entries which are closest to the new point
 - ▶ An alternative method might be to use all those points within a certain range of the new point
- ▶ The most common classification of these points is then given to the new point

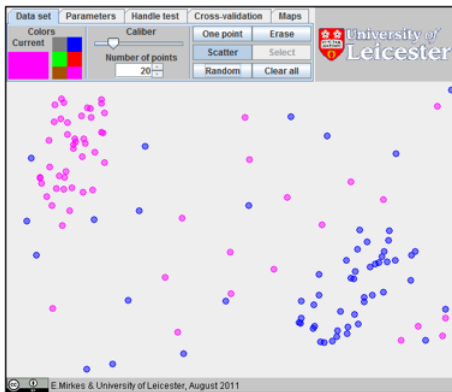
The k Nearest Neighbours Algorithm

The algorithm (as described in [1] and [2]) can be summarised as:

1. A positive integer k is specified, along with a new sample
2. We select the k entries in our database which are closest to the new sample
3. We find the most common classification of these entries
4. This is the classification we give to the new sample

Example Using the Applet

- ▶ First we set up the applet [1] like this:

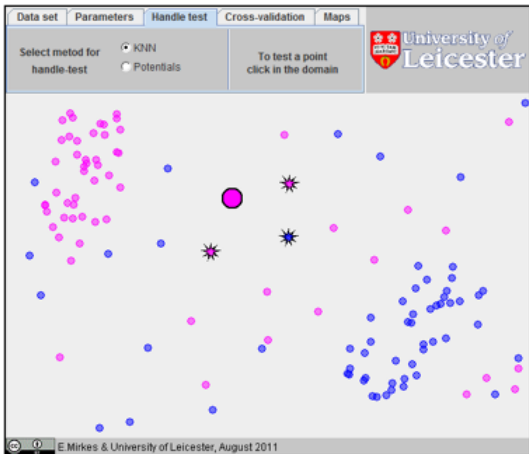


- ▶ i.e. with a cluster of 40 points of different colours in opposite corners, and 20 points of random noise from each colour

Example Using the Applet

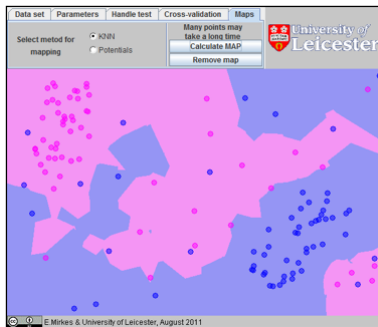
- ▶ These blue and pink points will be our training dataset, since we have specified their characteristics (x and y coordinates), and their classification (colour)
- ▶ We then select the “Handle Test” tab at the top, and click anywhere in the screen to simulate trying to classify a new point
- ▶ For the value of k specified in the parameters tab, the applet will find and highlight the k nearest data points to the test point
- ▶ The most common colour is then shown in the big circle where the test point was placed, as shown on the next slide

Example Using the Applet



Example Using the Applet

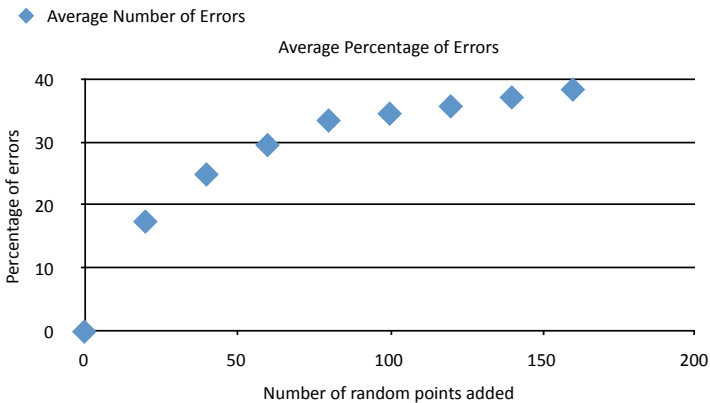
- ▶ Alternatively, we can use the applet to draw the map of the data
- ▶ The colour at each point indicates the result of classifying a test point at that location



Cross Validation

- ▶ In order to investigate how good our initial training set is, we can use a procedure called cross validation
- ▶ Essentially, this involves running the kNN algorithm on each of the points in the training set in order to determine whether they would be recognised as the correct type
- ▶ Clearly, this can very easily be corrupted by introducing random noise points into the training set
- ▶ To demonstrate this, the applet was used with increasing numbers of random data points in the training set, and the number of cross validation errors counted, as shown in the graph on the next slide

Cross Validation



Cross Validation

- ▶ It can clearly be seen that including more random noise points in the training set increases the number of cross validation errors
- ▶ As the number of random noise points becomes very large, the percentage of points which fail the cross validation tends to 50%
 - ▶ As would be expected for a completely random data set

Main Problems With the Algorithm

- ▶ Difficult to implement for some datatypes
 - ▶ e.g. colour, geographical location, favourite quotation, etc.
 - ▶ This is because it relies on being able to get a quantitative result from comparing two items
 - ▶ Can often be got around by converting the data to a numerical value, for example converting colour to an RGB value, or geographical location to latitude and longitude
- ▶ Slow for large databases
 - ▶ Since each new entry has to be compared to every other entry
 - ▶ Can be sped up using data reduction...

Condensed Nearest Neighbours Data Reduction

Data Reduction

The database is summarised by finding only the important data-points

Condensed Nearest Neighbours Data Reduction

Data Reduction

The database is summarised by finding only the important data-points

- ▶ Datapoints in the training set are divided into three types (as described in [1]):
 1. **Outliers:** points which would not be recognised as the correct type if added to the database later
 2. **Prototypes:** the minimum set of points required in the training set for all the other non-outlier points to be correctly recognised
 3. **Absorbed points:** points which are not outliers, and would be correctly recognised based just on the set of prototype points
- ▶ New samples only need to now be compared with the prototype points, rather than the whole database

CNN Data Reduction Algorithm

The algorithm (as described in [1]) is as follows:

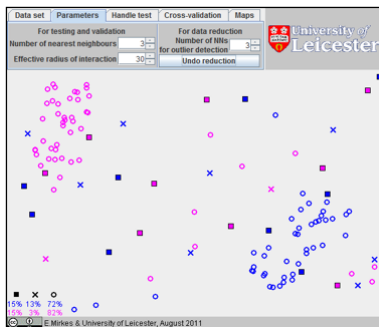
1. Go through the training set, removing each point in turn, and checking whether it is recognised as the correct class or not
 - ▶ If it is, then put it back in the set
 - ▶ If not, then it is an outlier, and should not be put back
2. Make a new database, and add a random point.
3. Pick any point from the original set, and see if it is recognised as the correct class based on the points in the new database, using kNN with $k = 1$
 - ▶ If it is, then it is an absorbed point, and can be left out of the new database
 - ▶ If not, then it should be removed from the original set, and added to the new database of prototypes
4. Proceed through the original set like this
5. Repeat steps 3 and 4 until no new prototypes are added

Effects of CNN Data Reduction

- ▶ After applying data reduction, we can classify new samples by using the kNN algorithm against the set of prototypes
 - ▶ Note that we now have to use $k = 1$, because of the way we found the prototypes
- ▶ Classifying a new sample point is now faster, since we don't have to compare it to so many other points
- ▶ Classifying new samples against the new reduced data set will sometimes lead to different results than comparing the new sample against the whole training set
 - ▶ This is the trade-off we have to make, between speed and accuracy

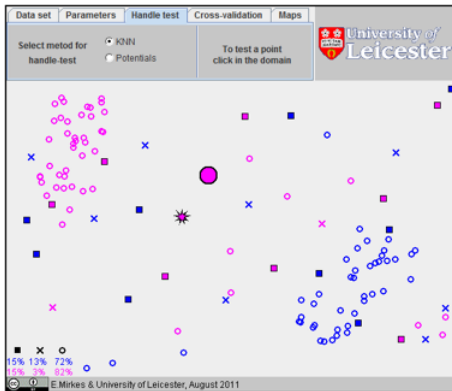
Example Using the Applet

- ▶ We set up a dataset as before
- ▶ Under the parameters tab, we select "Implement Reduction", and reduce the number of nearest neighbours for test points to 1
- ▶ Crosses mark outliers, squares mark prototypes, and circles mark absorbed points



Example Using the Applet

- ▶ Introducing a test point as before, we can see that it still has the same result:

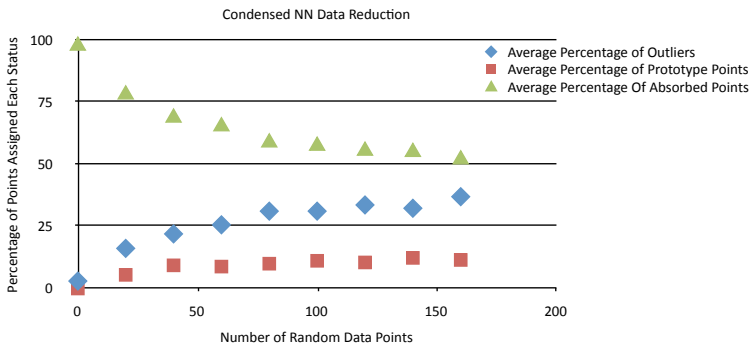


How Noise Affects CNN

- ▶ Again using the applet, we can investigate how well CNN manages to reduce the number of data points, in the presence of different quantities of random noise
- ▶ To do this, we record what percentage of points were assigned each classification on each of three different data sets as we increased the number of random noise points
- ▶ The results are presented in the graph on the next slide

How Noise Affects CNN

The effects of implementing CNN data reduction on increasingly noisy data sets:



How Noise Affects CNN

- ▶ As can be seen, increasing the number of noise points had the following effects:
 1. Percentage of points classed as outliers increased dramatically
 2. Percentage of points classed as absorbed decreased
 3. Percentage of points classed as prototypes increased slightly
- ▶ These are all as would be expected

Problems With CNN

- ▶ The CNN algorithm can take a long time to run, particularly on very large data sets, as described in [4] and [3].
 - ▶ There are alternative methods for reducing dimensionality, which are described in [3]
 - ▶ An alternative version of CNN which often performs faster than standard CNN is also given in [4]
- ▶ Using CNN can cause our system to give us different classifications than if we just used kNN on the raw data

Summary

- ▶ We can use the kNN algorithm to classify new points, by giving the same classification as those which the new point is close to
- ▶ For large datasets, we can apply CNN data reduction to get a smaller training set to work with
- ▶ Tests after CNN will be obtained more quickly, and will normally have the same result



E. Mirkes, 2011

KNN and Potential Energy (Applet),

University of Leicester. Available at: <http://www.math.le.ac.uk/people/ag153/homepage/KNN/KNN3.html>, 2011.



L. Kozma, 2008

k Nearest Neighbours Algorithm.

Helsinki University of Technology. Available at: <http://www.lkozma.net/knn2.pdf>, 2008.



N. Bhatia et al,

Survey of Nearest Neighbor Techniques.

International Journal of Computer Science and Information Security, Vol. 8, No. 2, 2010.



F. Anguilli,

Fast Condensed Nearest Neighbor Rule.

Proceedings of the 22nd International Conference on Machine Learning, Bonn, Germany, 2005.