Machine Learning (CS60050)

Instructor : Prof. Aritra Hazra
scribed by Prasanta Dutta (20CS91F02)

January 30, 2021

1 K Nearest Neighbour (KNN)

1.1 KNN Algorithm (From any point q)

1. Compute the distance of q from all other points.
2. Find top k points according to the distance found in step 1.
3. Get the majority label of the k points and assign it to q.

1.2 Time Complexity Analysis

Let there are N points, each of dimension D.

1. Calculating the distance between any two points takes $O(D)$ computation. Hence for calculating the distance from q to all other points, takes $O(D) * O(N) = O(ND)$ time complexity.
2. Finding the nearest point, takes $O(N)$ time complexity. For finding top k points takes $O(Nk)$ time complexity.

   This complexity can be reduced to $O(Nlog_2k)$ by using a binary min-heap. For that a min-heap is created using k points which takes $O(k)$ time complexity. Then a point is deleted from the heap and a point from remaining $(N-k)$ points is inserted into the heap, which takes $O(log_2k)$ time complexity. The last step is repeated for $(N-k)$ times. Hence total time complexity for calculating 2^{nd} step of the algorithm is $O(k) + O((N-k)log_2k) = O(Nlog_2k)$[since $k \leq N$].

3. Calculating the majority label among k points, takes $O(k)$ time complexity.

Hence the total time complexity for KNN algorithm is $O(ND + Nlog_2k + k)$.

1.3 Observation

From the above analysis, it is clear that time complexity of KNN, depends on k, N and D. If these parameters are large then KNN is infeasible for real world applications. Generally k is taken as $\sqrt{N}$ or $N/10$ (if N is small). Or it could be decided from the experimental results. Now we will see how to reduce effective training example (N) and how to represent the dimensionality (D) successfully.

1.4 Procedure to reduce effective training examples

1.4.1 Decision Boundary Consistent Set

In this method, the decision boundary is fixed first and then points are chosen such that the boundary is not shifted.

Figure 1: Decision Boundary Consistent Set

1.4.2 Minimum Consistent Set

In this method, points are chosen iteratively and decision boundary is modified accordingly. This method stops when minimum set of points fix the decision boundary in such a way that training examples are properly classified. This method is also known as *Condensing*.

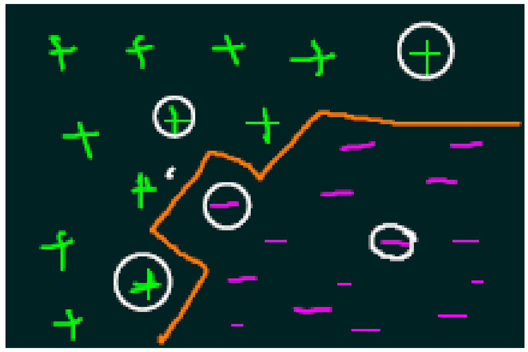

Figure 2: Minimum Consistent Set

Algorithm for Condensing

1. Choose a misclassified point.
2. Recompute the decision boundary.
3. Repeat the above two steps until all the points are correctly classified.

The worst case time complexity for the Condensing algorithm is $O(N^3)$ but in practical scenario it works quite well (far from the worst case).

1.5 Procedure to represent the dimensionality of attributes

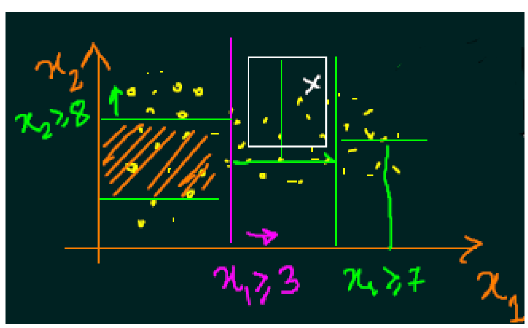

Figure 3: Process of splitting of points in 2-d space

We need to select an attribute window where exactly k points resides. For that *k-dimensional tree or k-d tree* data structure is used in order to store the training points efficiently. It is analogous to the indexing technique which is used in DBMS, where a large file is broken into several smaller files. Then the files are indexed for searching efficiently. k-d tree is a space-partitioning data structure for organizing points in a k-dimensional space. In this context the number of dimension means the number of attributes, a data point has.

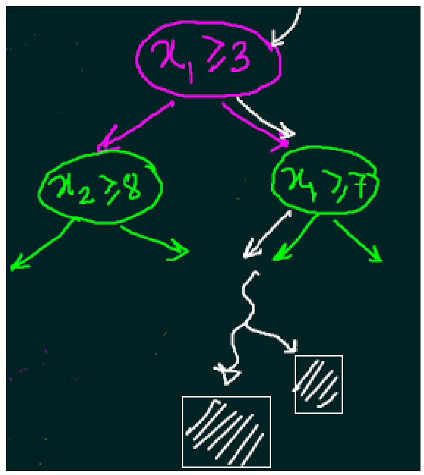

Figure 4: Process of forming a k-d tree

1.5.1 Procedure for K-d tree formation

1. a : Find the axis to which data is aligned to.
2. m : Find the median points by a.
3. H_l, H_r : Split the data into two halves based on m.
4. Apply steps 1-3 until each half contains $\leq k$ number of points.

k-d tree searching is much efficient. Even searching neighbouring block is easy (flipping the last condition). The dimensionality is actually the number of index nodes in the k-d tree. Fig. 3 shows the split of points based on two attributes x_1 and x_2. Here the yellow points are the points of training data. Green and purple lines show the partition of the points. Fig. 4 shows the k-d tree so formed based on the two attributes x_1 and x_2.

1.6 Practical application of KNN algorithm

- **Medical Domain / Diagnostic :** Here based on the similarity with the previous instances of medical data, a new case is diagnosed. Hence this type of learning is called *Instance based learning*.

- **Judgement / Law verdict :** Here based on the similarity with the previous cases, action is taken on a new case. Hence this type of learning is called *Case based learning*.

As KNN algorithm greedily finds out the nearest neighbours, hence this type of learning is known as *Lazy learning*.