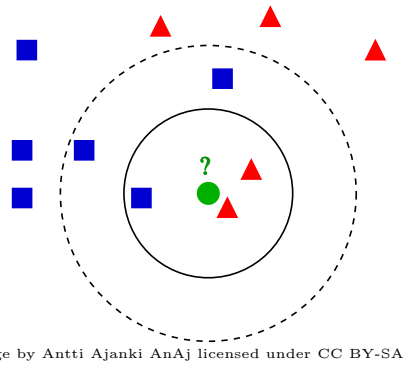
Map Reduce Design : k-Nearest Neighbor

1 k-Nearest Neighbor

This is a design activity. There is no programming involved.

The k-Nearest Neighbor algorithm is a machine learning algorithm to infer classification of a query based on known classification of a set of observations.

Mathematically, you are given a database of n points located a d-dimensional feature space, and each point is associated with a class (an integer) and queries, vectors in the same d dimensional space. The k-nearest neighbor algorithm will guess a class for each query by: identifying the k points in the database that are the closest to the query (by euclidean distance), computing which class appears the most frequently among the k nearest neighbors.



On the above example the database is composed of triangles and squares located in a 2d space. The query is the green dot. With $k = 3$, the three closest neighbors in the database are the ones inside the solid circles. Since there are 2 triangles and 1 square, the algorithm guesses the query is a triangle. If the algorithm is run with $k = 5$, then the closest neighbors in the database would be the ones within the dashed circle. There are 3 squares and 2 triangles, so the query would be classified as a square.

In practice the application is passed two files. The first one that contain the database, a CSV file where each line is a point in the database expressed as d values and the identifier of class. The second one contain the queries, a CSV file where each line is a query point expressed as a query identifier (an integer), and d values.

The application should be designed to be a single pipeline. And the application should be designed so that all map and reduce tasks require at most $O(k)$ memory. However, there is no restriction on how many key-value pairs they can generate or ingested by a task. The program should generate for each query identifier the predicted class of the query.

Question: Specify the application by describing each phase, whether it is a map phase, a reduce phase, or a cross product. Describe how the tasks are generated/parametrized. Describe what the computation is performed by the tasks in each phase. Describe in between phases the tuples that are generated and what the keys and values are.