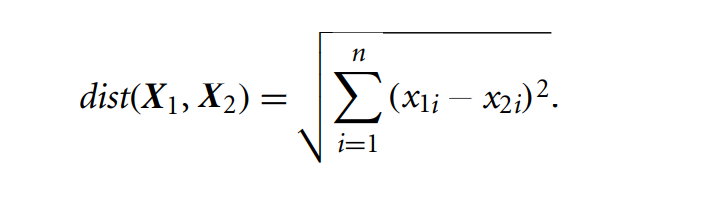
**K-nearest Neighbor**

The k-nearest neighbors algorithm (k-NN) is a non-parametric machine learning method first developed by Evelyn Fix and Joseph Hodges in 1951, and later expanded by Thomas Cover. It is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems. It basically assumes that similar things exist in close proximity. In other words, similar things are near to each other.

* In k-NN classification, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbor.
* In k-NN regression, the output is the property value for the object. This value is the average of the values of k nearest neighbors.

k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally, and all computation is deferred until function evaluation. Since this algorithm relies on distance for classification, if the features represent different physical units or come in vastly different scales then normalizing the training data can improve its accuracy dramatically.

“Closeness” is defined in terms of a distance metric, such as Euclidean distance. The Euclidean distance between two points or tuples, say, X1 = (x11, x12, ... , x1n) and X2 = (x21, x22, ... , x2n), is



Both for classification and regression, a useful technique can be to assign weights to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of 1/d, where d is the distance to the neighbor.

The neighbors are taken from a set of objects for which the class (for k-NN classification) or the object property value (for k-NN regression) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required.

A peculiarity of the k-NN algorithm is that it is sensitive to the local structure of the data.

**The KNN Algorithm**

1. Load the data.
2. Initialize K to your chosen number of neighbors.
3. For each example in the data.
4. Calculate the distance between the query example and the current example from the data.
5. Add the distance and the index of the example to an ordered collection.
6. Sort the ordered collection of distances and indices from smallest to largest (in ascending order) by the distances.
7. Pick the first K entries from the sorted collection.
8. Get the labels of the selected K entries.
9. If regression, return the mean of the K labels.
10. If classification, return the mode of the K labels.

**Choosing the right value for K**

To select the K that’s right for your data, we run the KNN algorithm several times with different values of K and choose the K that reduces the number of errors we encounter while maintaining the algorithm’s ability to accurately make predictions when it’s given data it hasn’t seen before.

Here are some things to keep in mind:

1. As we decrease the value of K to 1, our predictions become less stable. Just think for a minute, imagine K=1 and we have a query point surrounded by several reds and one green, but the green is the single nearest neighbor. Reasonably, we would think the query point is most likely red, but because K=1, KNN incorrectly predicts that the query point is green.
2. Inversely, as we increase the value of K, our predictions become more stable due to majority voting / averaging, and thus, more likely to make more accurate predictions (up to a certain point). Eventually, we begin to witness an increasing number of errors. It is at this point we know we have pushed the value of K too far.
3. In cases where we are taking a majority vote (e.g. picking the mode in a classification problem) among labels, we usually make K an odd number to have a tiebreaker.

**Advantages**

* The algorithm is simple and easy to implement.
* There is no need to build a model, tune several parameters, or make additional assumptions.
* This algorithm is versatile as it can be used for classification, regression, and search problems.

**Disadvantages**

* The algorithm gets significantly slower as the number of examples and/or predictors/independent variables increase.

**Real-world application of KNN**

* KNN can be used for Recommendation Systems. Although in the real world, more sophisticated algorithms are used for the recommendation system. KNN is not suitable for high dimensional data, but KNN is an excellent baseline approach for the systems. Many companies make a personalized recommendation for its consumers, such as Netflix, Amazon, YouTube, and many more.
* KNN can search for semantically similar documents. Each document is considered as a vector. If documents are close to each other, that means the documents contain identical topics.
* KNN can be effectively used in detecting outliers. One such example is Credit Card fraud detection.

**Conclusion**

K- Nearest Neighbors (KNN) identifies the nearest neighbors given the value of K. It is lazy learning and non-parametric algorithm. KNN works on low dimension dataset while faces problems when dealing with high dimensional data.

**References**

1. **Textbook:** Jiawei Han and Micheline Kamber: Data Mining - Concepts and Techniques, 3rd Edition, Morgan Kaufmann Publisher, 2014
2. **Textbook:** Pang-Ning Tan, Michael Steinbach, Vipin Kumar: Introduction to Data Mining (1st Edition)
3. **Paper:** Pádraig Cunningham, Sarah Jane Delany (29 Apr 2020) k-Nearest Neighbor Classifiers 2nd Edition (with Python examples).
4. **Wikipedia** - <https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm>
5. Article on K-Nearest Neighbors (KNN) algorithm (Towards Data Science) - <https://towardsdatascience.com/k-nearest-neighbors-knn-algorithm-23832490e3f4>
6. Article on K-Nearest Neighbors (KNN) algorithm - <https://www.geeksforgeeks.org/k-nearest-neighbours/>