K-Nearest Neighbor Classifier

Dr. Deepak Ranjan Nayak

Department of CSE
IIITDM Kancheepuram

Table of contents

Do we need hundreds of classifiers?

Fernández-Delgado, Manuel, Eva Cernadas, Senén Barro, and Dinani Amorim. "Do we need hundreds of classifiers to solve real world classification problems?." The Journal of Machine Learning Research 15, no. 1 (2014): 3133-3181. **cited 1649 times**

- 179 classifiers arising from 17 families (discriminant analysis, Bayesian, neural networks, support vector machines, decision trees, rule-based classifiers, boosting, bagging, stacking, random forests and other ensembles, nearest neighbors, etc.)
- 121 data sets, which represent the whole UCI
- Better classifier: random forest, SVM with Gaussian and polynomial kernels, extreme learning machine

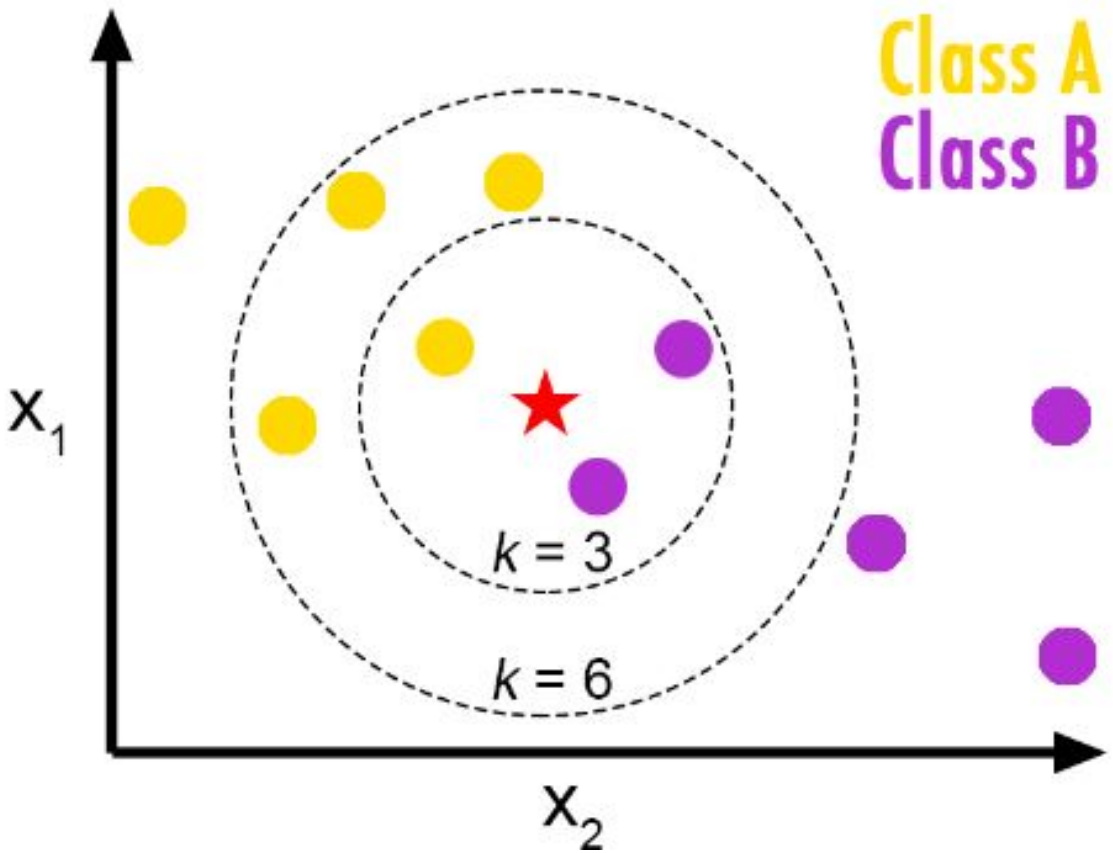
k-NN: Introduction

- Non-parametric based algorithm
- Neither probability distribution nor discriminant function is known (all we have is labeled data)
- It is also called as a *lazy learner* as no generalized model is constructed using training set for predicting a class label.
- Classification is performed on the basis of its nearest neighbors.
- It requires no preprocessing of the labeled sample set prior to their use.
- It is important and good for datasets which has noise and variability.

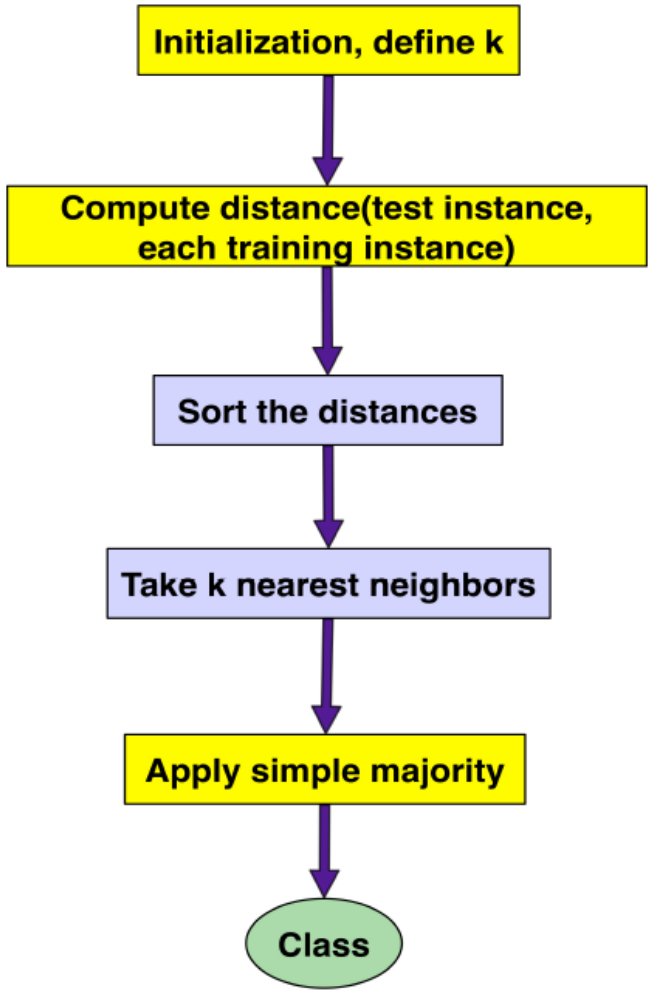
Lazy vs. Eager Learning

- Lazy Learner (e.g., instance-based learning):
  - Simply stores training data (or only minor processing) and waits until it is given a test tuple
  - Less time in training but more time in predicting
- Eager learning (eg. Decision trees, SVM, NN):
  - Given a set of training set, constructs a classification model before receiving new (e.g., test) data to classify
  - More time in training but less time in predicting

Key Idea



Basic steps


Initialization, define k
Compute distance(test instance, each training instance)
Sort the distances
Take k nearest neighbors
Apply simple majority
Class

Algorithm

Given c classes, $C_i, i = 1, 2, \ldots, c$, a test sample $y \in \Re^d$ (whose class label is unknown), and N training samples $x_j \in \Re^d, j = 1, 2, \ldots, N$ with the corresponding class labels.

Task: Classify y in one of the c classes.

1. Initialize the value of k
2. Calculate the distance between y and all training samples x_j
3. Find the k-closest neighbors and identify the number n_i of the samples that belong to c_i, $\sum_{i=1}^{c} n_i = k$.
4. Assign y to class c_i for which $n_i > n_j \forall j \neq i$, i.e., y is assigned to the class in which the majority of the k-closest neighbors belongs to.

How to handle tie cases?

```
1: if (a tie exists) then
2:     Compute sum of distances of neighbors in each class tied
3:     if (no tie occurs) then
4:         Classify y in the class of minimum sum
5:     else
6:         Classify y in the class of last minimum found (arbitrary)
7:     end if
8: else
9:     Classify y in the majority class (as found in step 4)
10: end if
```

Choosing k value

Note

- k value should not be a multiple of c.
- For two-class problem, k value should be an odd value (no tie will occur). But this may not work when the classes are more than two.

The value of k has to be adjusted (crossvalidation)

- We can overfit (k too low)
- We can underfit (k too high)

Distance Metrics

Euclidean Distance $$d(x_p, x_q) = \sqrt{\sum_{i=1}^{d}(x_{p,i} - x_{q,i})^2}$$

Manhattan Distance $$d(x_p, x_q) = \sum_{i=1}^{d}|x_{p,i} - x_{q,i}|$$

Example

Data

Name	Acid durability	Strength	Class
Type-1	7	7	Bad
Type-2	7	4	Bad
Type-3	3	4	Good
Type-4	1	4	Good

Assign a class label for test sample= [3, 7]. Assume $k = 3$.

Example

Data

Name	Acid durability	Strength	Class
Type-1	7	7	Bad
Type-2	7	4	Bad
Type-3	3	4	Good
Type-4	1	4	Good

Assign a class label for test sample= [3, 7]. Assume $k = 3$.

Answer: Good

Pros and Cons

Pros

- simple to implement and use
- cost of the learning process is zero
- robust to noisy data by averaging k-nearest neighbours
- k-NN classification is based solely on local information
- the decision boundaries can be of arbitrary shapes

Pros and Cons

Cons

- O(n) for each instance to be classified
- more expensive to classify a new instance than with a model
- computationally expensive to find the k nearest neighbours when the dataset is very large
- model can not be interpreted (there is no description of the learned concepts)


Thank You!