Instance based learning

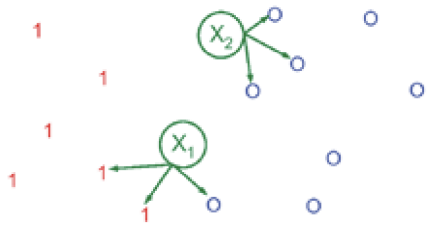
Introduction

Here we introduce non-parametric models. The core idea of instance based learning is to make predictions without an explicit formulation of the output function we are trying to guess.

Fundamentally, we will not work with some models, but with algorithms that directly compute predictions given a dataset and a new instance.

K-Nearest-Neighbours (K-NN)

KNN is one of the most popular algorithms. The idea is simple: given a datasetd D, a parameter k and a new instance x', we individuate the k nearest points to that new instance. The output prediction will correspond to the most frequent class present in those k neighbours.



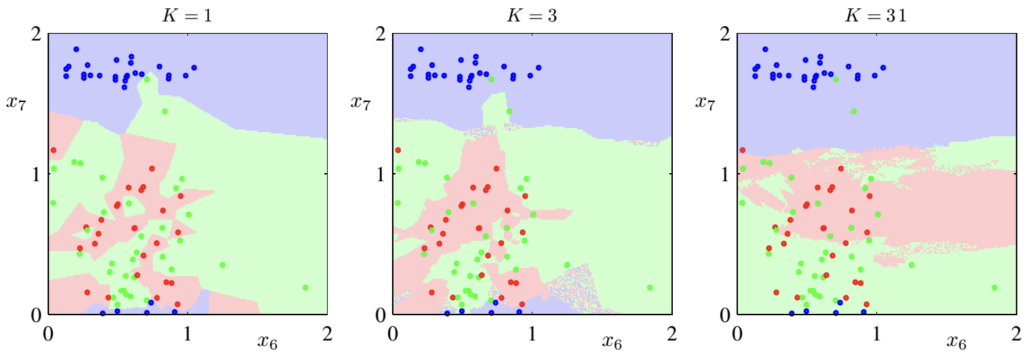
$K = 3$

Given a new instance, the likelihood for class C is:

$$P(C|x', D, k) = \frac{1}{k} \sum_{x_n \in N_k(x_n, D)} I(t_n = C)$$

With $N_k(x_n, D)$ being the k nearest points to x' and $I(t_n = C)$ a function that outputs one if the condition is true, 0 otherwise.

In general, the bigger is k, the smoother the separation surface will be. This will imply that for bigger number of k we will tend to reduce the probability of overfitting.



Despite being so intuitive, KNN has two main problems. The first one is storage: in fact in order to make new predictions, we will always have to consider the whole dataset. Another problem might be finding a proper distance function, that is not always easy to define.

Kernelized KNN

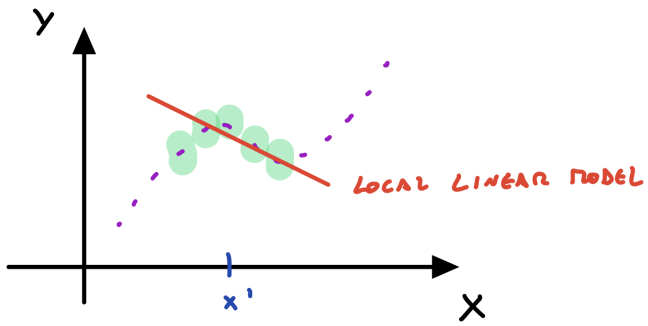
When computing $N_k(x_n, D)$ we can define the distance function as follows:

$$||x - x_n||^2 = x^T x + x_n^T x_n + 2x^T x_n$$

Notice how this form allows the application of the kernel trick.

Locally weighted regression

The concept of KNN can be extended also to the problem of regression. Given a dataset and a new instance, we can try to guess it by creating a local linear model that considers only the k-nearest points and make a prediction:



For example with k=5, we consider the 5 closest points to the new instance x' and fit a linear model on these 5 neighbours.

The actual steps are:

1. Compute $N_k(x', D)$
2. Fit a regression model $y(x; w)$ on $N_k(x', D)$
3. Return $y(x', w)$.