Introduction to Big Data and Machine Learning
Nonparametric methods

Dr. Mihail

October 1, 2019

Nonparametric

Idea

- So far we focused on models (probabilistic or deterministic) that are **governed** by a small number of parameters. That is called a *parametric* approach.
- An important limitation of this approach is that the density model might a poor approximation of a distribution that generates the data
- For example: if the process that generates the data is multimodal, a Gaussian will never capture this aspect, since Gaussians are necessarily unimodal

Histogram approach

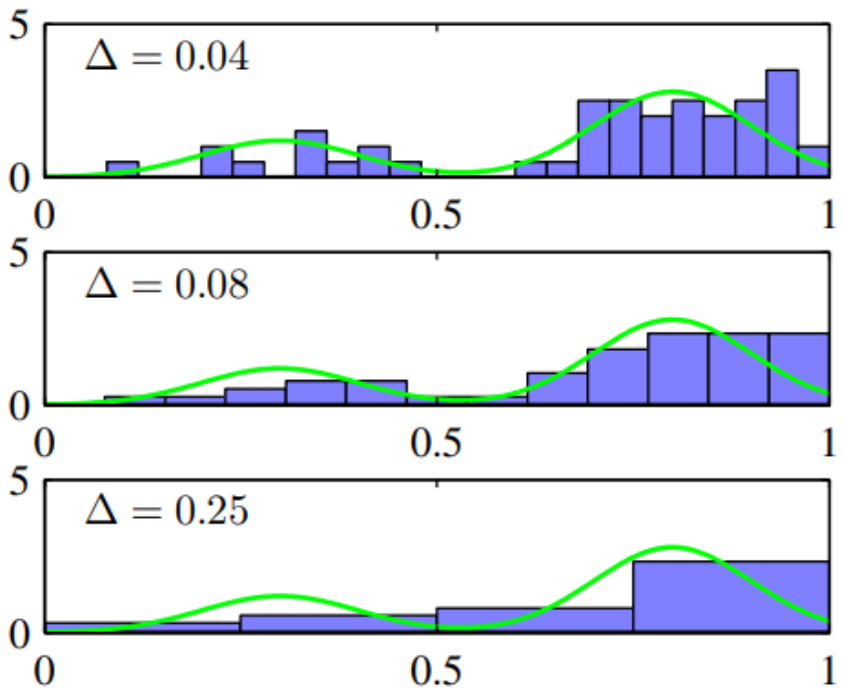
To illustrate

- Density estimation using histograms
- Standard histograms partition x into distinct bins of Δ_i and then count the number n_i of observations of x falling in bin i
- In order to turn this into a probability density (sum to 1) we simply divide by N and by the width of Δ_i of the bins to obtain the probability values for each bin given by:

$$p_i = \frac{n_i}{N\Delta_i} \tag{1}$$

Illustration

An illustration of the histogram approach to density estimation, in which a data set of 50 data points is generated from the distribution shown by the green curve. Histogram density estimates, based on (2.241), with a common bin width Δ are shown for various values of Δ.



Histrogram approach

- Benefit of histogram: once histogram has been computed, data can be discarded, useful when dataset is large
- Easy to update if data comes sequentially

Lessons

- To estimate the probability density at a particular location , we should consider the data points that lie within some local neighborhood of that points
- Note: concept of locality involves a distance metric
- The value of the smoothing parameter should neither be too large or too small

Kernel density estimators

- Suppose observations are being drawn from an unknown density $p(x)$ in some $D-$dimensional space, which we will assume to be Euclidean, and we wish to estimate $p(x)$
- Let us consider some small region $\mathcal{R}$ containing x. The probability mass associated with that region is

$$P = \int_{\mathcal{R}} p(x)dx \tag{2}$$

- Now suppose we have collected a dataset containing N observations drawn from $p(x)$. Each point has a probability P of falling within $\mathcal{R}$, the total number K of points that lie inside $\mathcal{R}$ will be distributed according to a binomial distribution:

$$Bin(K|N,P) = \frac{N!}{K!(N-K)!}P^K(1-P)^{1-K} \tag{3}$$

Statistics

- Using some insights from statistics we can see that the fraction of points falling inside the region is P from $\mathbb{E}[K/N] = P$, and similarly the variance around the mean is $var[K/N] = P(1-P)/N$
- For a large N, this distribution will sharply peak around the mean so

$$K \simeq NP \tag{4}$$

- If we also assume the region $\mathcal{R}$ is sufficiently small that the probability density $p(x)$ is roughly constant in that region, then we have

$$P \simeq p(x)V \tag{5}$$

where V is the volume of $\mathcal{R}$. Combining the above, we have:

$$p(x) = \frac{K}{NV} \tag{6}$$

The rise of two ideas

- The validity of Equation 6 depends on two contradictory assumptions, namely the region $\mathcal{R}$ is sufficiently small that the density is approximately constant over the region and yet sufficiently large (in relation to the value of that density) that the number K points falling inside the region is sufficiently for the binomial to be sharply peaked

Exploiting the result

- We can either fix K and determine the value V from the data, which gives rise to the K-**nearest-neighbor** technique or
- We can fix V and determine K from the data, giving rise to the **kernel** approach

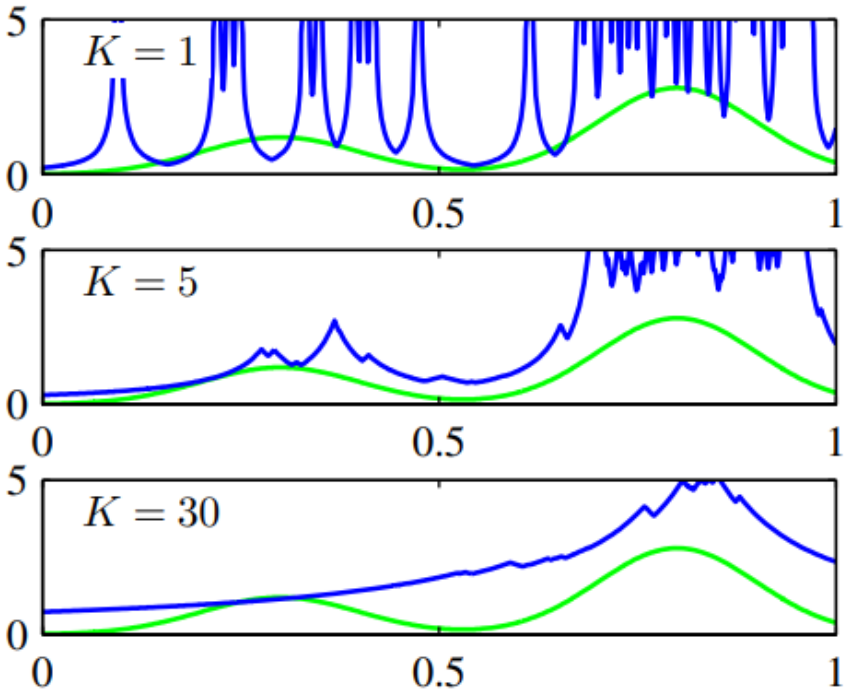
Nearest neighbor

Fixing K

- We fix K and determine the value of V from the data
- To do this, we consider a small sphere centered on the point x at which we wish to estimate the density $p(x)$, and allow the radius of the sphere to grow until it contains exactly K data points.
- The estimate of the density $p(x)$ is then given by Equation 6, with V set to the volume of the resulting sphere.
- This technique is known as K-nearest-neighbor

K-nearest-neighbor

Illustration of K-nearest-neighbour density estimation using the same data set as in Figures 2.25 and 2.24. We see that the parameter K governs the degree of smoothing, so that a small value of K leads to a very noisy density model (top panel), whereas a large value (bottom panel) smoothes out the bimodal nature of the true distribution (shown by the green curve) from which the data set was generated.



Classification with KNN

- K-nearest-neighbor technique can be used for classification using Bayes' theorem.
- To do this, we apply KNN separately to each class, then make use of Bayes' theorem.

KNN classification

- Suppose we have a dataset of N_k points in class $\mathcal{C}_k$ with N points in total, so that $\sum_k N_k = N$.
- If we wish to classify a new point x we draw a sphere centered on x containing precisely K points irrespective of their class. Suppose this sphere has a volume V and contains K_k points from class $\mathcal{C}_k$
- Then, using Equation 6, estimate a density associated with each class:

$$p(x|\mathcal{C}_k) = \frac{K_k}{N_k V} \tag{7}$$

KNN classification

- Similarly, the unconditional density is given by:

$$p(x) = \frac{K}{NV} \tag{8}$$

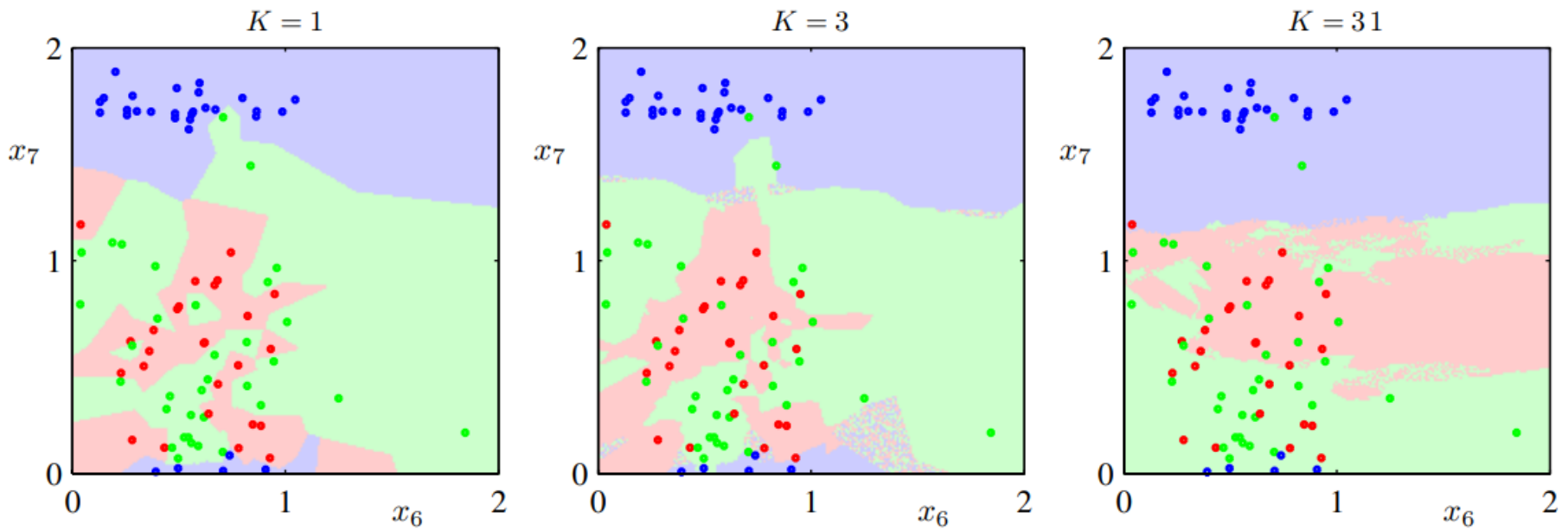
while the class priors are given by

$$p(\mathcal{C}_k) = \frac{N_k}{N} \tag{9}$$

and by using Bayes' theorem, we can get the posterior:

$$p(\mathcal{C}_k|x) = \frac{p(x|\mathcal{C}_k)p(\mathcal{C}_k)}{p(x)} = \frac{K_k}{K} \tag{10}$$

KNN Example



Memory based methods

Extending parametric models

- Linear parametric models seen so far estimate a few parameters from the training set and discard the training data for predictions
- We can combine the two approaches by casting parametric model into an equivalent "dual representation" where the predictions are also based on linear combinations of a "kernel" function evaluated at training data points
- For models which are based on a fixed nonlinear feature space mapping $\phi(x)$, the kernel is given by the relation

$$k(x, x') = \phi(x)^T \phi(x') \tag{11}$$

- The kernel is a symmetric function of its arguments so that $k(x, x') = k(x', x)$

Dual representations

- The simplest example of a kernel function is obtained by considering the identity: $\phi(x) = x$ so that $k(x, x') = x^T x'$. We will refer to this as the linear kernel.
- The concept of a kernel formulated as an inner product in a feature space allows us to build interesting extensions of well-known algorithms by making use of the "kernel trick" or "kernel substitution"
- The general idea is that if some algorithm is formulated in such a way that input vector x enters only in the form of a scalar products, we can replace that scalar product with some other choice of kernels

Kernel examples

- Many kernels have the property of being only a function of the difference between arguments, so that $k(x, x') = k(x - x')$, known as stationary because are invariant to translations in feature space
- Homogeneous kernels (also known as radial basis functions) depend only on the distance (typically Euclidean), such that $k(x, x') = k(||x - x'||)$

Dual representations

- Consider a linear regression model, whose parameters are determined by minimizing a regularized sum-of-squares error function given by

$$J(w) = \frac{1}{2}\sum_{n=1}^{N}\{w^T\phi(x_n) - t_n\}^2 + \frac{\lambda}{2} \tag{12}$$

where $\lambda \geq 0$. Setting the gradient of $J(w)$ to zero with respect to w we obtain:

$$w = -\frac{1}{\lambda}\sum_{n=1}^{N}\{w^T\phi(x_n) - t_n)\phi(x_n) = \sum_{n=1}^{N} a_n\phi(x_n) = \Phi^T a \tag{13}$$

where Φ is the design matrix whose n^{th} row is given by $\phi(x_n)^T$.

Dual representations

- The vector $a = (a_1, \ldots, a_N)^T$:

$$a_n = -\frac{1}{\lambda}\{w^T\phi(x_n) - t_n\} \tag{14}$$

- Instead of working with parameter vector w, we can now reformulate the least squares algorithm in terms of the parameter vector a giving rise to a dual representation. If we substitute $w = \Phi^T a$ into $J(w)$ we obtain:

$$J(a) = \frac{1}{2}a^T\Phi\Phi^T\Phi\Phi^T a - a^T\Phi\Phi^T t + \frac{1}{2}t^T t + \frac{\lambda}{2}a^T\Phi\Phi^T a \tag{15}$$

where $t = (t_1, \ldots, t_N)^T$. We can now define the Gram matrix $K = \Phi\Phi^T$ which is NxN symmetric matrix with elements

$$K_{nm} = \phi(x_n)^T\phi(x_m) = k(x_n, x_m) \tag{16}$$

Dual representation

- In terms of the Gram matrix, the sum-of-squares error function can be written as:

$$J(a) = \frac{1}{2}a^T KKa - a^T Kt + \frac{1}{2}t^T t + \frac{\lambda}{2}a^T Ka \tag{17}$$

setting the gradient of $J(a)$ with respect to a to zero, we get:

$$a = (K + \lambda I_N)^{-1} t \tag{18}$$

and substituting this back into a linear regression model, we obtain the following prediction for a new input x

$$y(x) = w^T \phi(x) = a^T \Phi \phi(x) = k(x)^T (K + \lambda I_N)^{-1} t \tag{19}$$