Lecture 18: Non-parametric learning with K-nn

K-Nearest Neighbor (K-nn) method

1. K-nn density estimation --- estimate the prior and class models
2. K-nn classification --- estimate the posterior directly
3. K-nn decision rule
4. k-nn error analysis

Summary

Last lecture has introduced the non-parametric methods for density estimation. Given n samples $\{\mathrm{x}_1, \mathrm{x}_2, ..., \mathrm{x}_n\}$, the density at point x is estimated as,

$$p_n(\mathrm{x}) = \frac{k_n/n}{V_n} = \frac{1}{n}\sum_{i=1}^{n} \delta_n(\mathrm{x} - \mathrm{x}_i)$$

There are two methods for designing the window sequence $V_1, V_2, ..., V_n$ at point x:

- the Parzen windows,
- the k_n-Nearest-Neighbor.

The Parzen window method is an unbiased estimate of the underlying density p(x)

$$\boxed{\lim_{n\to\infty} p_n(\mathrm{x}) = p(\mathrm{x}), \quad \lim_{n\to\infty} \sigma^2(p_n(\mathrm{x})) = 0.}$$

Parzen window vs K-NN--- a comparison

Given n samples, in a non-parametric method, we estimate the density at x by

$$\boxed{p_n(x) = \frac{k_n / n}{V_n}}$$

Design I: Parzen window.

Fix the window size $V_n = \frac{V_1}{\sqrt{n}}$ and $k_n = k_n(x)$ is a function of x.

$$\boxed{p_n(x) = \frac{k_n / n}{V_n} = \frac{k_n(x)}{V_1\sqrt{n}}}$$

Design II: K-nn method.

Fix the number of sample inside window $K_n = \sqrt{n}$ and $V_n = V_n(x)$ --- a function of x.

$$\boxed{p_n(x) = \frac{k_n / n}{V_n} = \frac{1}{V_n(x)\sqrt{n}}}$$

Parzen window vs K-NN—a comparison

Parzen window

$$V_n = \frac{V_1}{\sqrt{n}}$$

$$\boxed{p_n(x) = \frac{k_n / n}{V_n} = \frac{k_n(x)}{V_1\sqrt{n}}}$$

1. Here the number of samples falling in a window can be counted explicitly

$$K_n(x) = \sum_{i=1}^{n} \varphi\left(\frac{x - x_i}{h_n}\right)$$

2. We can prove the convergence

K-nn

$$K_n = \sqrt{n}$$

$$\boxed{p_n(x) = \frac{k_n / n}{V_n} = \frac{1}{V_n(x)\sqrt{n}}}$$

1. The window size does not have an explicitly form. Very irregular plot, the diameter of the window is

$$\phi(V_n(x)) = 2\,|x - x_K^*|, \quad x_K^* \in \{x_1, ..., x_n\}$$

2. The integration of $p_n(x)$ is infinity

$$\int p_n(x)\,dx = \infty$$

Examples of K-nn density

$$\boxed{p_n(x) = \frac{k_n / n}{V_n} = \frac{1}{V_n(x)\sqrt{n}}}$$

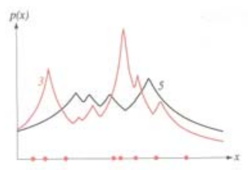


FIGURE 4.10. Eight points in one dimension and the *k*-nearest-neighbor density estimates, for $k = 3$ and 5. Note especially that the discontinuities in the slopes in the estimates generally lie *away* from the positions of the prototype points.

$$\boxed{p_1(x) = \frac{1}{2\,|\,x - x_1\,|}}$$

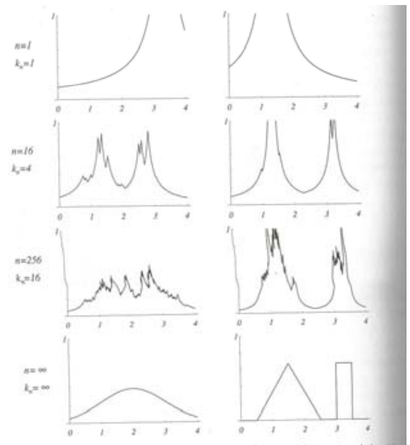


FIGURE 4.12. Several *k*-nearest-neighbor estimates of two unidimensional densities
Gaussian and a bimodal distribution. Notice how the finite *n* estimates can be qu
"spiky."

Non-parametric Classification

Given n labeled samples $\chi = \{\mathrm{x}_1, \mathrm{x}_2, ..., \mathrm{x}_n\}$, and we have

classes	$\omega_1,\ \omega_2,\ ...,\ \omega_c$	
#samples	$n_1,\ n_2,\ ...,\ n_c$	$\sum_{i=1}^{c} n_i = n.$
$p(\omega_i\|\chi)$	$\frac{n_1}{n},\ \frac{n_2}{n},\ ...,\ \frac{n_c}{n},$	the prior probabilities.

At each point x, a cell of volume V_n covers total k samples, with k_i samples in each class ω_i, $i = 1, 2, ..., c$. The likelihood is estimated as

$$p(\mathrm{x}|\omega_i, \chi) = \frac{k_i/n_i}{V_n}, \qquad i = 1, 2, ..., c.$$

Note that the above class models are estimated by the Parzen windows – all c classes share the same window V_n at x, and it is possible that $k_i = 0$ for some classes. In general, One may use different windows V_n for different classes.

Non-parametric Classification

Given the learned prior model and class models, we compute the posterior probability,

$$P(\omega_i|\mathrm{x},\chi) = \frac{p(\mathrm{x}|\omega_i,\chi)p(\omega_i|\chi)}{\Sigma_{j=1}^{c} p(\mathrm{x}|\omega_j,\chi)p(\omega_j|\chi)} = \frac{k_i}{k}.$$

– it is the fraction of the samples within the cell that are labeled ω_i.

The Bayes rule chooses

$$\omega^*(\mathrm{x}) = \arg\max_i \{k_1, k_2, ..., k_c\}.$$

It turns out that the K-nn classification is also a special case of Bayes decision !

Nearest Neighbor Decision Rule

The Bayes decision rule in the non-parametric case indicates that we can go directly for a decision, and thus bypass the steps of estimating $p(\omega_i|\chi)$ and $p(\mathrm{x}|\omega_i,\chi)$. This leads to two methods:

- the nearest neighbor decision rule,
- the k_n-nearest neighbor decision rule.

Intuitively, both methods partition the feature space into c disjoint subspaces,

$$\Omega = \cup_{i=1}^{n}\Omega_i, \quad \Omega_i \cap \Omega_j = \emptyset, i \neq j.$$

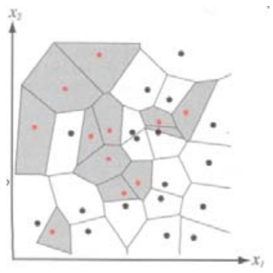
It is possible that Ω_i is not connected for some i.

Nearest Neighbor Decision Rule

Let $\{(\mathrm{x}_1, \omega(\mathrm{x}_i)), ..., (\mathrm{x}_n, \omega(\mathrm{x})_n)\}$ be the labeled samples, for any $\mathrm{x} \in \Omega$,

$$\omega_{NN}(\mathrm{x}) = \omega(\mathrm{x}^*), \quad \mathrm{x}^* = \arg\min_{\mathrm{x}_j}\{||\mathrm{x} - \mathrm{x}_j||, \ j = 1, 2, ..., n\}$$

Remark I: this simply partitions the feature space by a Voronoi diagram, with each sample x_i occupies a cell.



Nearest Neighbor Decision Rule

Asymptotic Behavior of the NN decision rule.

When large samples $\chi = \{\mathrm{x}_1, \mathrm{x}_2, ..., \mathrm{x}_n\}$ are available, for any point x, its nearest sample x^* in χ *approaches* x, therefore

$$p(\omega_i|\mathrm{x}^*) \approx p(\omega_i|\mathrm{x}).$$

Therefore, the chance of $\omega(\mathrm{x}^*) = \omega_i$ is simply $p(\omega_i|\mathrm{x})$, that is,

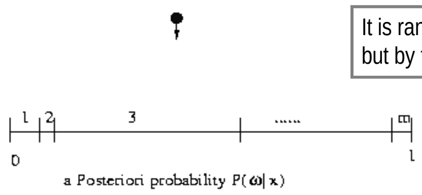
$$\omega_{NN}(\mathrm{x}) = \omega(x^*) \sim p(\omega|\mathrm{x})$$

Randomized decision

Asymptotic Behavior of the NN decision rule.

The Nearest-Neighbor decision rule is a randomized decision procedure, draw a random number $s \in [0, 1]$, and decide $\omega_{NN}(\mathrm{x})$ according to the posterior probability $p(\omega|x)$.

It is randomized not by our decision,
but by the random nature of the samples !



If n is large enough, $P(\omega|\mathrm{x})$ is close to the true posterior probability.

In fact, we are assuming that the samples are drawn from the true posterior probability.

Two Special Cases

Recall that we have discussed this randomized decision before in comparison with the Bayes decision rule. We have proven that the randomized decision is not as good as the Bayes decision rule.

Two special cases,

1. when $P(\omega_i|\mathrm{x}) \approx 1.0$ for a one class, then in most of the time, $\omega_{NN}(\mathrm{x}) = \omega_{Bayes}(\mathrm{x})$.

2. when $P(\omega_i|\mathrm{x}) \approx \frac{1}{c}$, i.e. a uniform distribution, then $\omega_{NN}(\mathrm{x}) = \omega_{Bayes}(\mathrm{x})$ with one out of c times.

In both cases, the performance of the Nearest-Neighbor decision rule matches that of the Bayes rule.

Now we analyze the error rate of the NN rule quntitatively.

NN error analysis

Error rate of the NN-decision rule at x with n samples.

$$P_n(e|\mathrm{x}) = \int p_n(e, \mathrm{x}^*|\mathrm{x}) d\mathrm{x}^* = \int P_n(e|\mathrm{x}^*, \mathrm{x}) p(\mathrm{x}^*|\mathrm{x}) d\mathrm{x}^*$$

as we know,

$$p(\mathrm{x}^*|\mathrm{x}) = \delta(\mathrm{x} - \mathrm{x}^*) \quad \text{as} \quad n \to \infty,$$

where $\delta()$ is the Dirac delta function, and

$$\begin{aligned} P_n(e|\mathrm{x}^*, \mathrm{x}) &= 1 - \sum_{i=1}^{c} P(\omega(\mathrm{x}) = \omega_i, \omega(\mathrm{x}^*) = \omega_i|\mathrm{x}^*, \mathrm{x}) \\ &= 1 - \sum_{i=1}^{c} P(\omega_i|\mathrm{x}^*) P(\omega_i|\mathrm{x}) \end{aligned}$$

NN error analysis

Error rate of the NN-decision rule at x with n samples.

$$\begin{aligned} \lim_{n\to\infty} P_n(e|\mathrm{x}) &= \int [1 - \sum_{i=1}^{c} P(\omega_i|\mathrm{x}^*) P(\omega_i|\mathrm{x})] \delta(\mathrm{x} - \mathrm{x}^*) d\mathrm{x}^* \\ &= 1 - \sum_{i=1}^{c} P^2(\omega_i|\mathrm{x}) \end{aligned}$$

and the expected error rate is

$$P = \lim_{n\to\infty} \int P_n(e|\mathrm{x}) p(\mathrm{x}) d\mathrm{x} = \int [1 - \sum_{i=1}^{c} P^2(\omega_i|\mathrm{x})] p(\mathrm{x}) d\mathrm{x}$$

In comparison, the optimal Bayes rate is

$$P^* = \int [1 - max\{P(\omega_{Bayes}(\mathrm{x})|\mathrm{x})\}] p(\mathrm{x}) d\mathrm{x}$$

The Bayes rate is obtained by assuming we know the true densities.

NN error analysis

The error bound of the NN-decision rule:

$$P^* \leq P \leq P^*(2 - \frac{c}{c-1}P^*)$$

[Proof] The lower bound is obvious, we only prove the upper bound. Let $\omega_m = \omega_{Bayes}(\mathrm{x})$, and $P^*(e|\mathrm{x}) = 1 - P(\omega_m|\mathrm{x})$, then we have,

$$\begin{aligned}\sum_{i=1}^{c} P^2(\omega_i|\mathrm{x}) &= P^2(\omega_m|\mathrm{x}) + \sum_{i \neq m} P^2(\omega_i|\mathrm{x}) \\ &\geq (1 - P^*(e|\mathrm{x}))^2 + \frac{{P^*}^2(e|\mathrm{x})}{c-1}.\end{aligned}$$

so

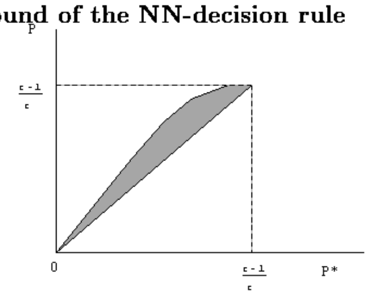
$$1 - \sum_{i=1}^{c} P^2(\omega_i|\mathrm{x}) \leq P^*(e|\mathrm{x})(2 - \frac{c}{c-1}P^*(e|\mathrm{x}))$$

Take an integral on both sides, the conclusion follows.

A lemma: for n positive real numbers $z_1, z_2, ..., z_n$ with constraint $\sum_{i=1}^{n} z_i = a$, then $\sum_{i=1} n z_i^2 \geq \frac{a^2}{n}$. Prove it by lagrange multipliers.

NN error analysis

The error bound of the NN-decision rule



***P* reaches *P** at two extreme cases:**

Case I: when no information is available, $P = P^* = \frac{c-1}{c}$.

Case II: when there is no uncertainty, $P = P^* = 0$.

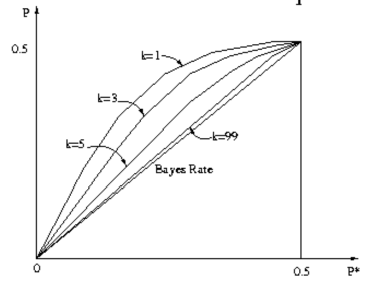
But this error bound is obtained as $n \to \infty$, for finite samples the convergence rate can be arbitrarily slow. The error rate $P_n(\epsilon)$ could be very bad.

Knn-Decision Rule

k-Nearest Neighbor Decision Rule:

At each point x, assign x the label that is most frequently represented among the k-nearest samples.

The error bound for a two-class classification problem.



The bigger k is, the more accurate the estimation of the class models. However, the larger k is, the bigger the window is, thus the less accurate the estimation is. Given finite training samples, the optimal k is a trade-off between these two aspects.

Discussion on distance measure

The K-nn method is quite popular due to its simplicity in learning and classification
This lecture studies some in-depth issues related to K-nn.

1 Means for expedite the K-nn algorithm
  - a). Editing and pruning (redundant) training samples
  - b). Search tree
  - c). Computing partial distance (pruning samples over a certain bound value)

2. Metircs and distance

3. Invariance to transformations

4. Tangent distance

K-nn distance measure

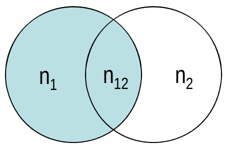
For any two points x and y in a d-space

- Minkowski metric $$D(x,y) = (\sum_{i=1}^{d} |x_i - y_i|^k)^{1/k}$$

2. Manhattan metric $$D(x,y) = \sum_{i=1}^{d} |x_i - y_i|$$

3. Mahalanobis metric $$D(x,y) = X' \Sigma^{-1} Y$$

4. Tanimoto metric for two sets $$D(s_1,s_2) = \frac{n_1 + n_2 - 2n_{12}}{n_1 + n_2 - n_{12}}$$



Distance is sensitive to transforms

The distance from “5” in the middle is closer to “8” on the left, in comparison with the distance to and slightly shifted “5” on the right.

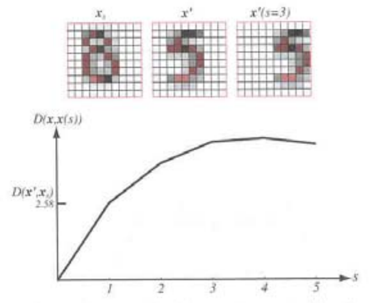


FIGURE 4.20. The uncritical use of Euclidean metric cannot address the problem of translation invariance. Pattern $\mathbf{x}'$ represents a handwritten 5, and $\mathbf{x}'(s=3)$ represents the same shape but shifted three pixels to the right. The Euclidean distance $D(\mathbf{x}', \mathbf{x}'(s=3))$ is much larger than $D(\mathbf{x}', \mathbf{x}_8)$, where $\mathbf{x}_8$ represents the handwritten 8. Nearest-neighbor classification based on the Euclidean distance in this way leads to very large errors. Instead, we seek a distance measure that would be insensitive to such translations, or indeed other known invariances, such as scale or rotation.

Invariant distance to transforms

There are two ways to make the nn-distance work under various transforms.

Method 1:

For exch training example x_i, we make many copies with transformations (translation, rotation and scaling).

Method 2:

We compute a distance on-line which is the minimum distance between y and a transformed x, by a transform function with parameter a in a transformation group G. For example, a is a vector for translations, rotation, scaling etc.

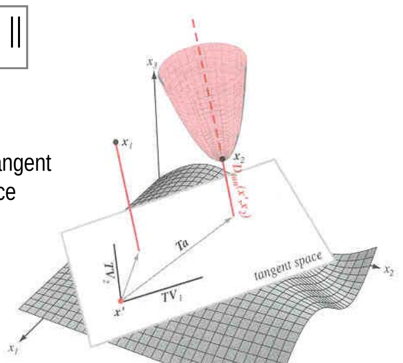
$$\boxed{D(x,y) \;=\; \min_{a\in G} \| f(x;\; a) - y \|}$$

Tangent distance

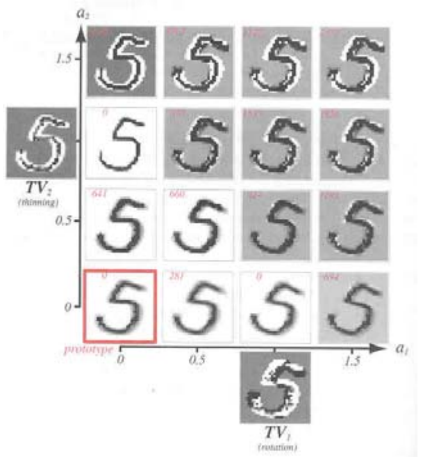
One may approximate the transform function by a first order Taylor expansion, thus we have a tangent distance

$$\boxed{D(x,y) \;=\; \min_{a\in G} \| (x + Ta) - y \|}$$

T is a matrix for the gradients. X+Ta spans a tangent plane at x. D(x,y) is the perpendicular distance from y to the tangent plane.



Example of the tangent plane



a_2
1.5
TV_2
(thinning)
0.5
0
prototype
0
0.5
1.5
a_1
TV_1
(rotation)