Lecture 23 EE 559 Apr 15, 2024

Machine Learning I: Supervised Methods

B. Keith Jenkins

Announcements

- Slido event code: 53959875
- Homework 8 will be posted today after lecture (due Mon. 4/22)
- Project Report is due Friday 4/26
  - Templates have been posted
- Reminder: Final exam is Wed. 5/1/2024
  - 4:30 - 6:30 PM PDT
  - We will have a larger room! (SGM)

Today's lecture

- Density estimation techniques for machine learning (non-parametric)
  - Preliminaries
  - Convergence
  - 2 approaches (KDE, kNN)
- Kernel density estimation (KDE)
- k-nearest neighbors density estimation (kNN)
- Discriminative and Generative approaches to ML
- Classification based on density estimation
  - Generative (using KDE or kNN)
  - Discriminative (using kNN or binary-window KDE)

Density Estimation Techniques for Machine Learning

Suppose we want to use a Bayes min. error classifier:

$$g_i(\underline{x}) = \ln\left[p(\underline{x}|S_i)P(S_i)\right], \quad i = 1, 2, \cdots, C$$

or $\tilde{g}_i(\underline{x}) = \ln P(S_i|\underline{x})$

but we don't know $p(\underline{x}|S_i)$, $P(S_i|\underline{x})$, maybe $P(S_i)$.

$\Rightarrow$ Estimate $p(\underline{x}|S_i)$ (or $P(S_i|\underline{x})$) from the data.

Or in regression problems, we will typically want to know $p(\underline{x}, y)$, $p(y|\underline{x})$, and/or $p(\underline{x})$. If they are not known:

$\Rightarrow$ Estimate $p(\underline{x}, y)$, $p(y|\underline{x})$, and/or $p(\underline{x})$ from the data.

Intro. & Assumptions

Suppose we want to est. $p(\underline{x}|S_1)$, $p(\underline{x}|S_2)$, $\cdots$, $p(\underline{x}|S_C)$

Assume data pts. of S_j are not useful in estimating $p(\underline{x}|S_i)$, $i \neq j$.

$\Rightarrow$ Estimate $p(\underline{x}|S_i)$ using only data pts. labeled S_i, for $i = 1, 2, \cdots, C$.

Notation: when considering $p(\underline{x}|S_i)$ only, drop $|S_i$ and class subscripts & superscripts for now.

$\Rightarrow$ Estimate $p(\underline{x})$, given a set of data pts. (drawn i.i.d. from $p(\underline{x})$): $\underline{x}_1, \underline{x}_2, \cdots, \underline{x}_N$.

Or, e.g., estimate $p(\underline{x}, y)$ from the data: $(\underline{x}_i, y_i)$, $i=1,\cdots,N$.

General Approach to Estimate $p(\underline{x})$ from data

Vector $\underline{x}$: $P\{\underline{x} \text{ lies in } R\} = P = \int_R p(\underline{x}')\, d\underline{x}'$

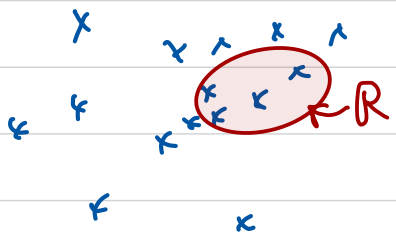
Draw N vectors $\underline{x}_1, \underline{x}_2, \cdots, \underline{x}_N$ i.i.d. from $p(\underline{x})$:

$P_k = P\{k \text{ vectors lie in } R\} = ?$ ↖ Binomial

$$P_k = \binom{N}{k} P^k (1-P)^{N-k}$$

mean: $E\{k\} = NP = \bar{k} \Rightarrow P = \frac{\bar{k}}{N}$

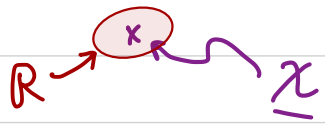
$\Rightarrow$ reasonable est. for P is $\hat{P} = \frac{k}{N}$



k in R
N-k outside R.

$\hat{P} = ? = \frac{4}{15}$

Assume R is small, has volume V, has $\underline{x}$ at its center. Also assume $p(\underline{x})$ is continuous at $\underline{x}$.



$$P = \int_R p(\underline{x}')\,d\underline{x}' \approx p(\underline{x}) \int_R d\underline{x}' = p(\underline{x})V$$

approx. $p(\underline{x}') \approx p(\underline{x})$ = const. in R.

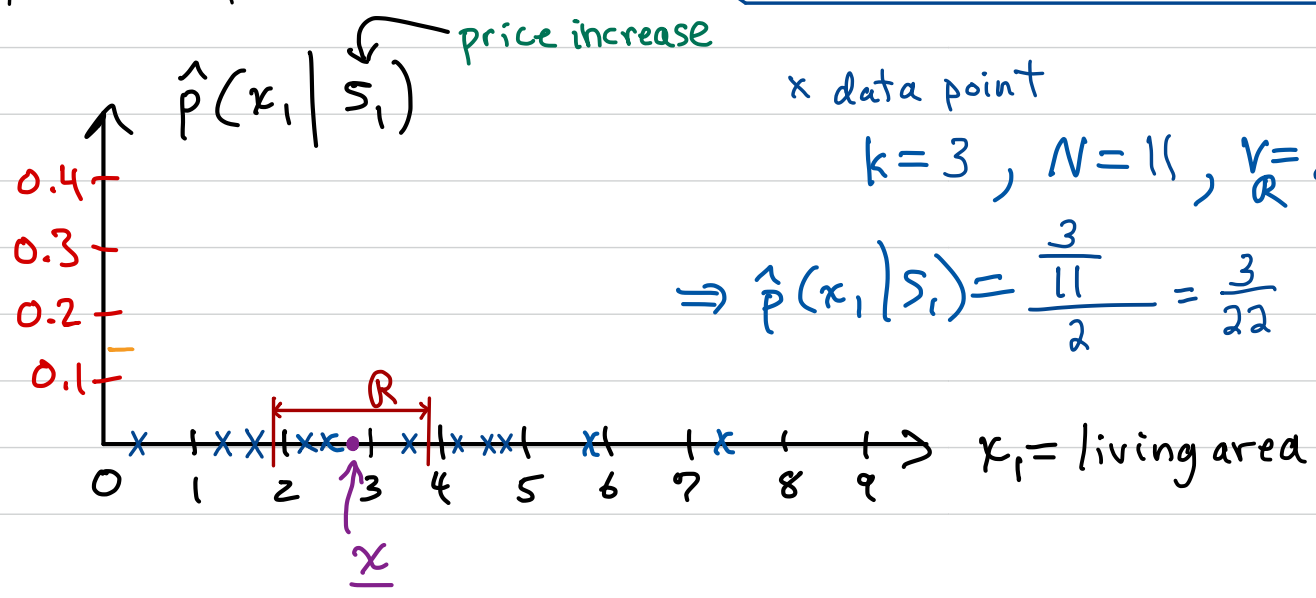
$$\Rightarrow \left[p(\underline{x}) \approx \frac{P}{V} = \frac{\int_R p(\underline{x}')\,d\underline{x}'}{\int_R d\underline{x}'} \right]$$

Use $P \approx \hat{P} = \frac{k}{N}$

$$\hat{p}(\underline{x}) = \frac{\hat{P}}{V} = \frac{\frac{k}{N}}{V}$$

- k: #(training) data pts. that lie in R
- N: Total # of (training) data pts.
- V: Volume of R.

Let $\hat{p}(\underline{x}) \rightarrow \hat{p}(x_1 | S_1)$:



x data point

$k = 3$, $N = 11$, $V_R = 2$

$$\Rightarrow \hat{p}(x_1 | S_1) = \frac{\frac{3}{11}}{2} = \frac{3}{22}$$

For theory, can construct a sequence

$p_n(\underline{x})$, $n = 1, 2, \cdots$

↑ Est. of $p(\underline{x})$ based on n data points.

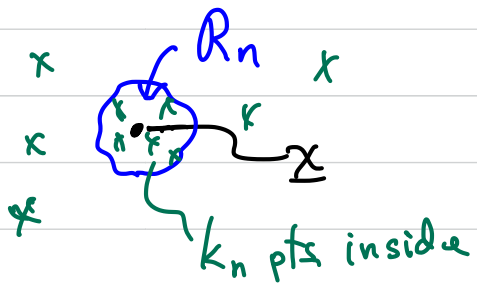
For each est. $p_n(\underline{x})$:

n data pts.

Region R_n containing $\underline{x}$, volume V_n

k_n = # pts. in R_n

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



For the sequence $p_n(\underline{x})$, $n = 1, 2, \cdots$, to converge to $p(\underline{x})$ as $n \to \infty$, we need to satisfy:

1. $\lim_{n\to\infty} V_n = 0$ (spatial resolution $\to \infty$)
2. $\lim_{n\to\infty} k_n = \infty$ (if $p(\underline{x}) \neq 0$) (quantization error $\to 0$)

and 3. $\lim_{n\to\infty} \frac{k_n}{n} = 0$ (% of data pts. in $R_n \to 0$)

2 PRIMARY APPROACHES TO ESTIMATING $p(\underline{x})$

1. Specify $V_n = f(n)$
   (perhaps also specify shape or profile of R_n)
   e.g., $V_n = \frac{1}{\sqrt{n}}$

   $\Rightarrow$ Kernel density estimation (KDE) (also: Parzen Windows) (PW)

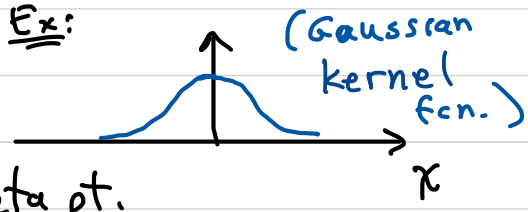
2. Specify $k_n = f(n)$
   e.g., $k_n = \sqrt{n}$

   $\Rightarrow$ k-Nearest Neighbors (k-NN)

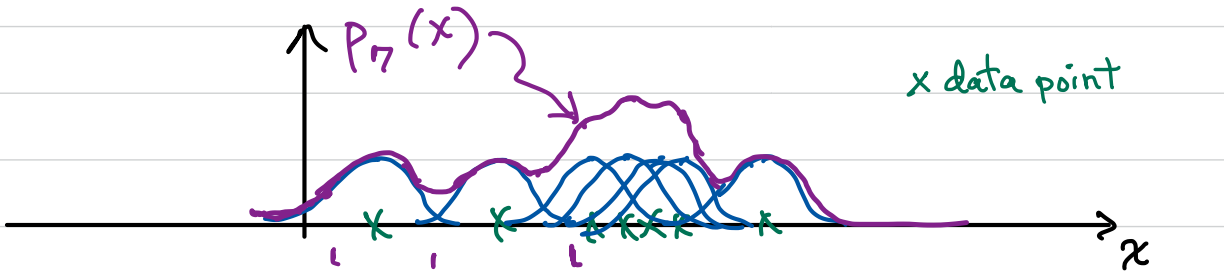
Both can converge to $p(\underline{x})$ as $n \to \infty$ if V_n or k_n is chosen appropriately.

Kernel Density Estimation (KDE)

Ex:



- Create a window or kernel function
- Center the kernel fcn. at each data pt.



- Take sum of all kernel (window) fcns.

Window fcn: $\Delta(\underline{u})$

$\Delta(\underline{x}-\underline{x}_i)$ centered at $\underline{x}_i$.

$$p_n(\underline{x}) = \frac{1}{n}\sum_{i=1}^{n} \Delta(\underline{x}-\underline{x}_i)$$ ← KDE estimate

For $p_n(\underline{x})$ to be a density, require:

$$(*) \begin{cases} \int \Delta(\underline{u})\, d\underline{u} = 1 \\ \Delta(\underline{u}) \geq 0 \end{cases}$$

If $\Phi(\underline{x})$ is an unnormalized window fcn., then we can define:

$$\Delta_n(\underline{x}) = \frac{1}{V_n}\Phi\left(\frac{\underline{x}}{h_n}\right)$$

← h_n = window (kernel) width parameter

Choose V_n to ensure $\int \Delta_n(\underline{x})\, d\underline{x} = 1$.

Convergence of PW/KDE

Is $p_n(\underline{x})$ random or deterministic?

> $p_n(\underline{x})$ is random, because $\underline{x}_i$ are drawn at random.

Does sequence of $p_n(\underline{x})$ converge to $p(\underline{x})$ as $n \to \infty$?

Convergence in mean square:

Def: $\lim_{n\to\infty} \overline{p_n}(\underline{x}) = p(\underline{x})$

and: $\lim_{n\to\infty} \sigma_n^2(\underline{x}) = 0$

$\sigma_n^2(\underline{x})$ is the variance of $p_n(\underline{x})$.

If $\Delta_n(\underline{x}) = \frac{1}{V_n}\,\Phi\left(\frac{\underline{x}}{h_n}\right)$

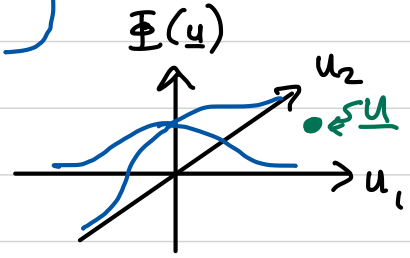
then the following conditions ensure convergence:

In addition to (*):

(1) $\sup \Phi(\underline{u}) < \infty$

(2) $\lim_{\|\underline{u}\| \to \infty} \left[\Phi(\underline{u}) \prod_{d=1}^{D} u_d\right] = 0$

} $\Phi(\underline{u})$ must be well behaved.



KDE convergence

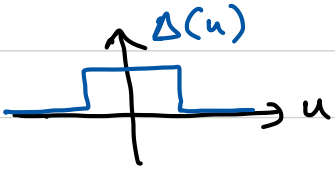
(3) $\lim_{n\to\infty} V_n = 0$

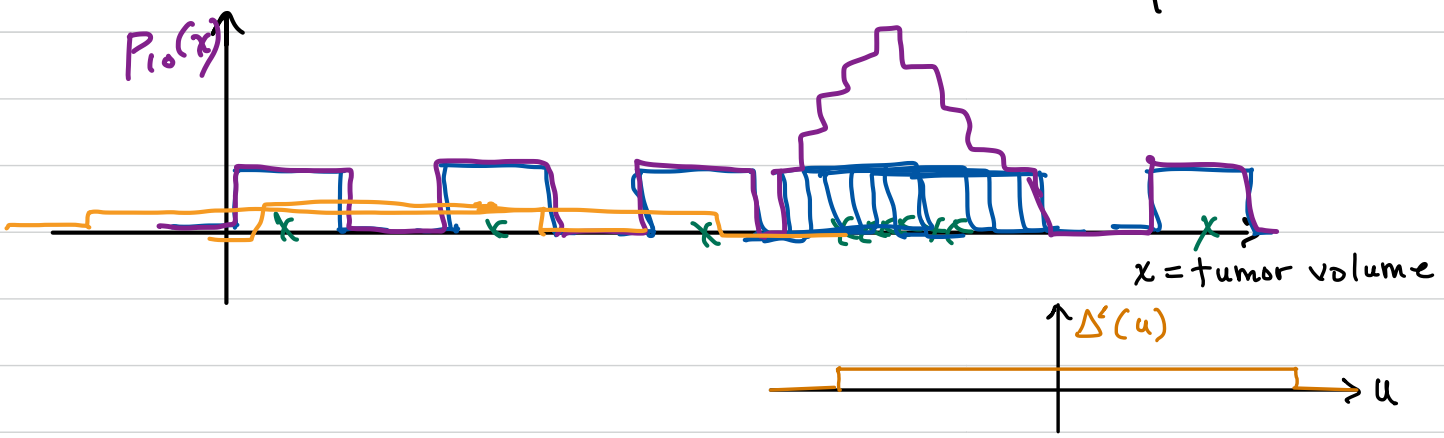
and (4) $\lim_{n\to\infty} n V_n = \infty$

} $V_n \to 0$, but not too fast.

? $V_n = \frac{1}{\sqrt{n}}$, $n V_n = \frac{n}{\sqrt{n}} = \sqrt{n}$

(S) Ex: KDE with rectangular window fcn.





Ideally, would like window width to vary, dependent on local density of data points.

$\Rightarrow$ k-NN

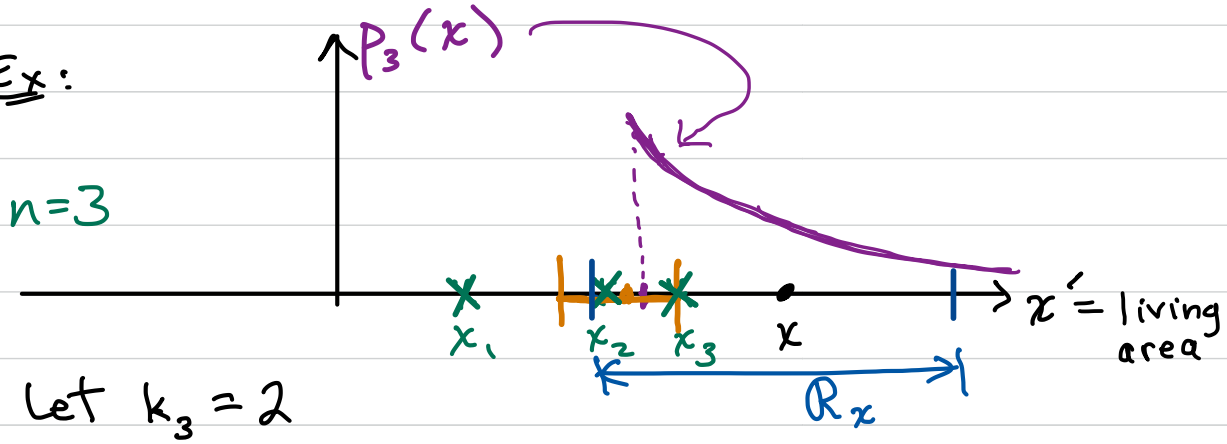
k-Nearest Neighbors (k-NN) for Density Estimation

No width parameter (h_n).
No choice of window or kernel shape (in k-NN standard form)
→ Let region width vary according to local density of data points.

Center R_n at $\underline{x}$. Make R_n just large enough to capture k_n data pts. Specify $k_n = f(n)$.

Still: $$p_n(\underline{x}) = \frac{\frac{k_n}{n}}{V_n}$$ ← k_n: specify; n: const.; V_n: calculated based on k_n.

Ex:



Let $k_3 = 2$

$$p_3(x) = \frac{\frac{k_3}{3}}{V_3} = \frac{\frac{2}{3}}{V_3(x)}$$

$$V_3(x) = 2|x - x_2|$$

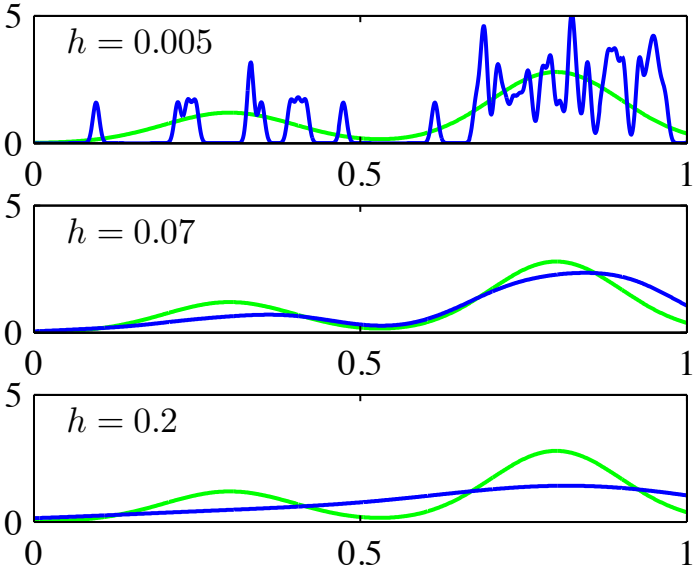
$$p_3(x) = \frac{\frac{2}{3}}{2|x-x_2|} = \frac{1}{3(x-x_2)}, \quad x \geq \frac{x_2+x_3}{2}$$

If we choose $k_n = k_1\sqrt{n}$, $k_1 > 0$, k_1 is a parameter
Then $p_n(\underline{x})$ converges to $p(\underline{x})$ as $n \to \infty$.

Examples

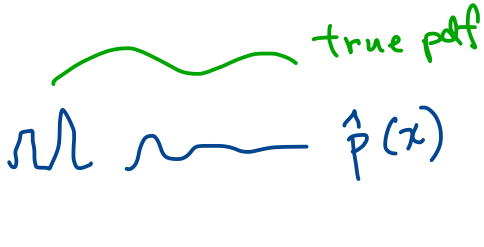
Dataset : 50 points drawn from pdf of green curve.

KDE



← under-smoothed.

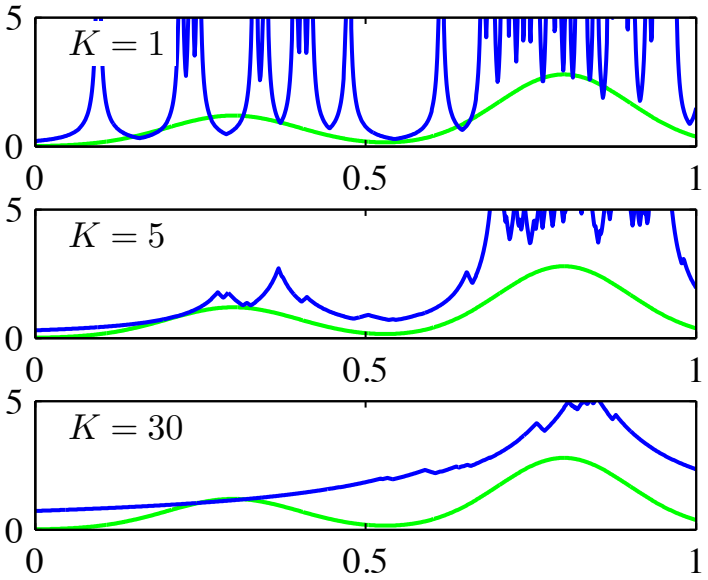
← oversmoothed



h = width parameter

Bishop Fig. 2.25

KNN



$K = k_n$

Bishop Fig. 2.26

Generative and Discriminative Approaches for ML

1. Generative

→ Estimate $p(\underline{x}, y)$. (or, $p(\underline{x}|y)$ and $p(y)$). ($p(\underline{x}|S_i)$, $P(S_i)$)

From that, can get: $P(y|\underline{x}) = \frac{p(\underline{x}, y)}{p(\underline{x})}$

Note: from $p(\underline{x}, y)$ we can synthetically generate more data.

2. Discriminative

→ Estimate $p(y|\underline{x})$ directly.

(Can't generate more data synthetically, unless also estimate $p(\underline{x})$.)

Classification Based on Density Estimation

> **Option 1: Generative Approach**

Apply above techniques to each class separately:

1. Estimate $p(\underline{x} | S_k)$ from data pts. $\underline{x}_i^{(k)}$ of S_k, separately for each class.

2. Use or estimate priors $P(S_k)$, $k = 1, 2, \cdots, C$

   e.g., $\hat{P}(S_k) = \frac{n^{(k)}}{N}$ (frequency of occurence)

   $n^{(k)}$ = # data pts. labeled S_k (in training set)

   N = # data pts. in training set.

3. Use Bayes classifier

   (min. error or min. risk)

$$\hat{p}(\underline{x} | S_k) \hat{P}(S_k) > \hat{p}(\underline{x} | S_j) \hat{P}(S_j) \quad \forall j \neq k$$

$$\Rightarrow \underline{x} \in S_k .$$

> Option 2 : Discriminative Approach

For the case of k-NN or KDE estimates using a binary-valued window fcn., we can derive an expression in advance based on estimates of posterior probabilities:

$$P(S_k|\underline{x}) > P(S_j|\underline{x}) \;\forall\, j \neq k \Rightarrow \underline{x} \in S_k$$

as follows:

$$p_n(\underline{x}, S_i) = \frac{\frac{k_n^{(i)}}{n}}{V_n}$$

$k_n^{(i)}$ = # data pts. in R_n that belong to S_i.
n = total # data pts. over all classes
k_n = total # data pts. in R_n. ← k-NN est. for classifier specifies this k.
V_n = volume of R_n.

$$p_n(\underline{x}, S_i) = P_n(S_i|\underline{x})\, p_n(\underline{x})$$

$$p_n(\underline{x}) = \sum_{i=1}^{C} p_n(\underline{x}, S_i)$$

$$P_n(S_i|\underline{x}) = \frac{p_n(\underline{x}, S_i)}{\sum_{i'=1}^{C} p_n(\underline{x}, S_{i'})} = \frac{\frac{k_n^{(i)}}{n}\cancel{\frac{}{V_n}}}{\sum_{i'=1}^{C}\left(\frac{k_n^{(i')}}{n}\cancel{\frac{}{V_n}}\right)}$$

$$\boxed{P_n(S_i|\underline{x}) = \frac{k_n^{(i)}}{\sum_{i'=1}^{C} k_n^{(i')}} = \frac{k_n^{(i)}}{k_n}}$$

$k_n^{(i)}$ ← # data pts in S_i that lie in R_n

k_n ← # data pts. over all classes that lie R_n.

$\Rightarrow P_n(S_i|\underline{x}) =$ % of data pts. in R_n that belong to S_i.

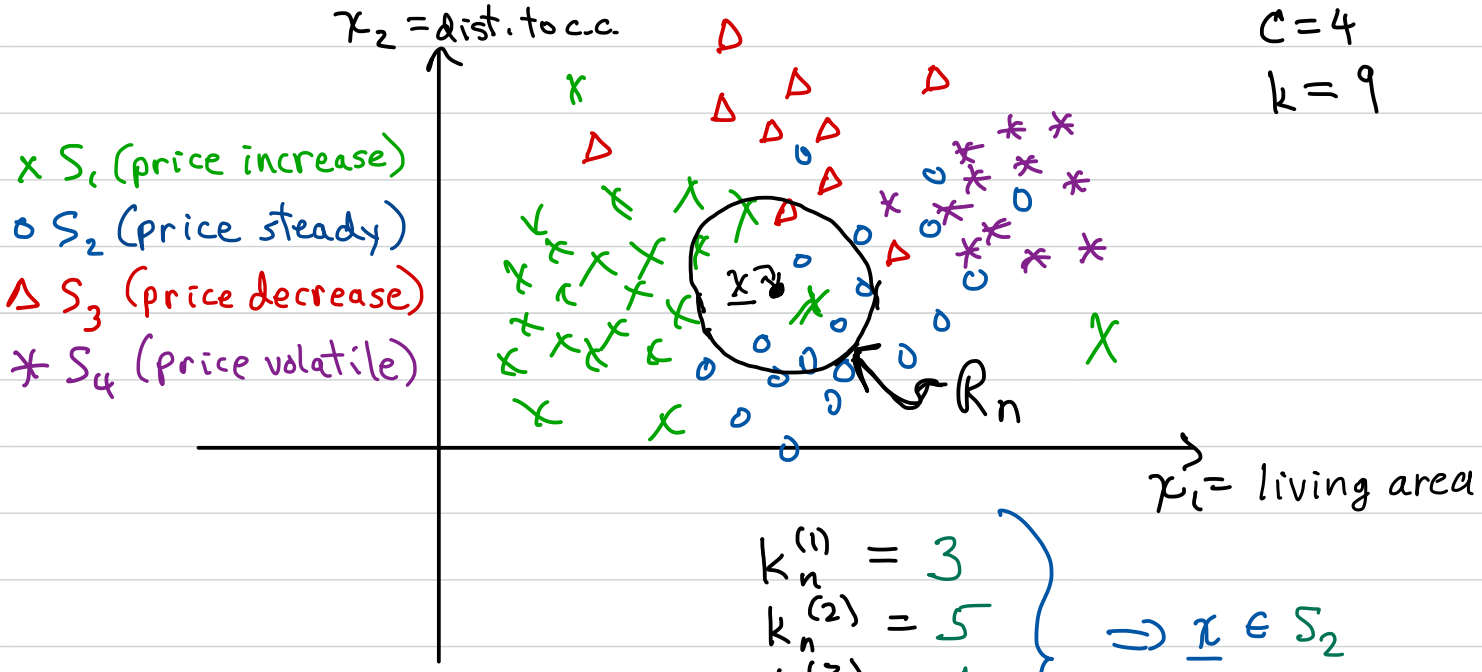
Use in a Bayes min. error classifier:

$$P(S_k|\underline{x}) > P(S_j|\underline{x}) \quad \forall j \neq k \Rightarrow \underline{x} \in S_k.$$

$$\frac{k_n^{(k)}}{k_n} > \frac{k_n^{(j)}}{k_n}$$

$\therefore$ if $k_n^{(k)} > k_n^{(j)} \quad \forall j \neq k \Rightarrow \underline{x} \in S_k$

↳ Bayes min. error dec. rule for estimates based on kNN or KDE with binary-valued window function.



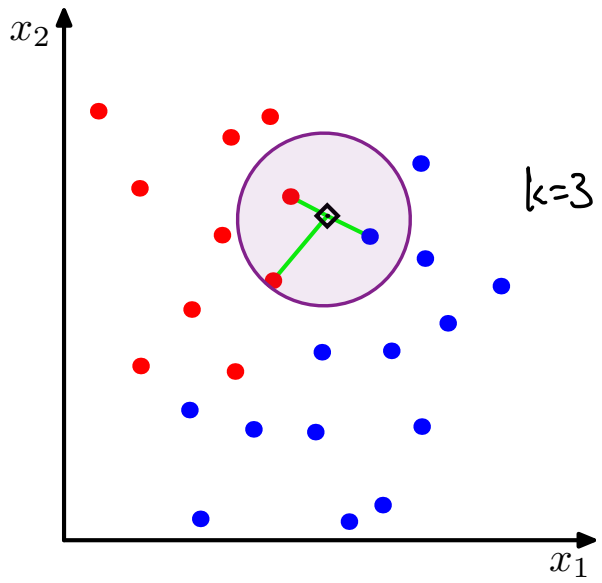
$k_n^{(1)} = 3$
$k_n^{(2)} = 5$
$k_n^{(3)} = 1$
$k_n^{(4)} = 0$

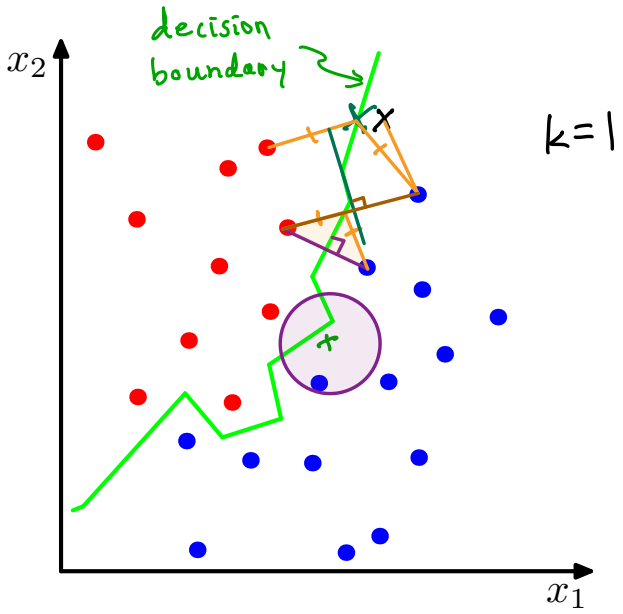
$\Rightarrow \underline{x} \in S_2$

Ex's – see below ↓

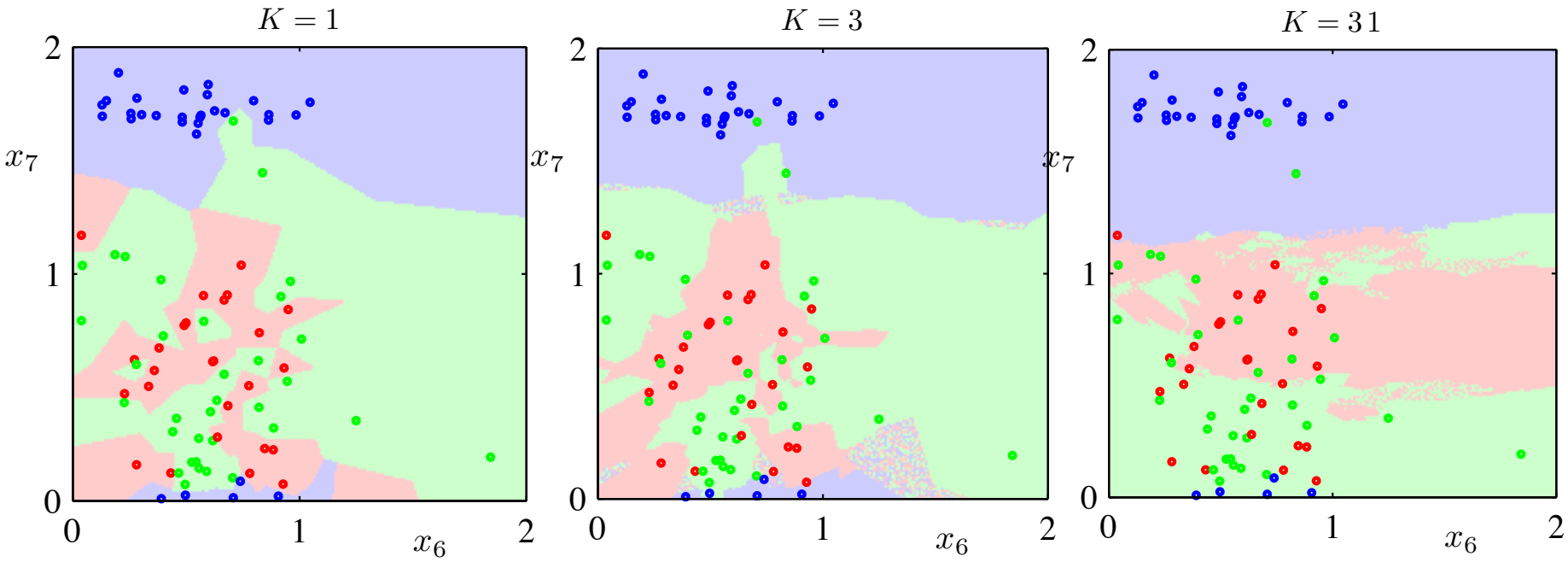
Ex

C = 2 classes





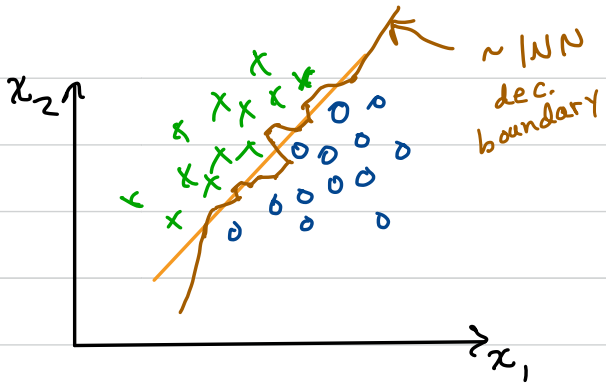
Bishop Figure 2.27



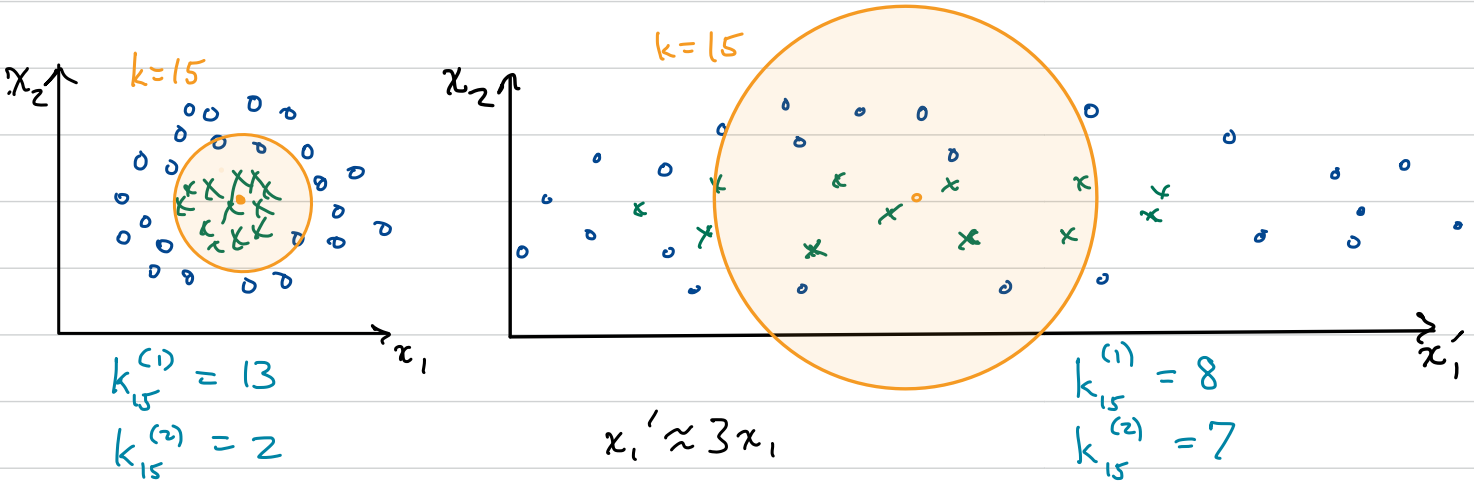
Bishop Figure 2.28

kNN classification on real dataset (oil data (Bishop)), C = 3.

Comments on kNN and KDE



1. Normalization matters!
   - Sensitive to differing scale sizes of different features



2. Choice of k (kNN) or kernel width h (KDE) affects amount of smoothing and resolution (i.e. underfit / overfit).

3. Can be computationally slow for large N or large D.
   - There are algorithms to speed up computation (e.g., tree search methods), usually giving an approximate result.