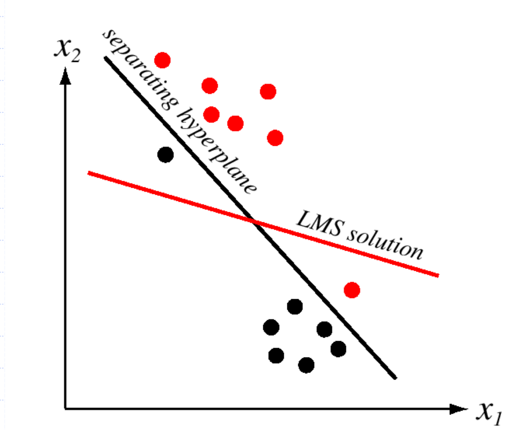
Local methods and simple comparisons

Randy Julian
Lilly Research Laboratories

Remember the bad news…

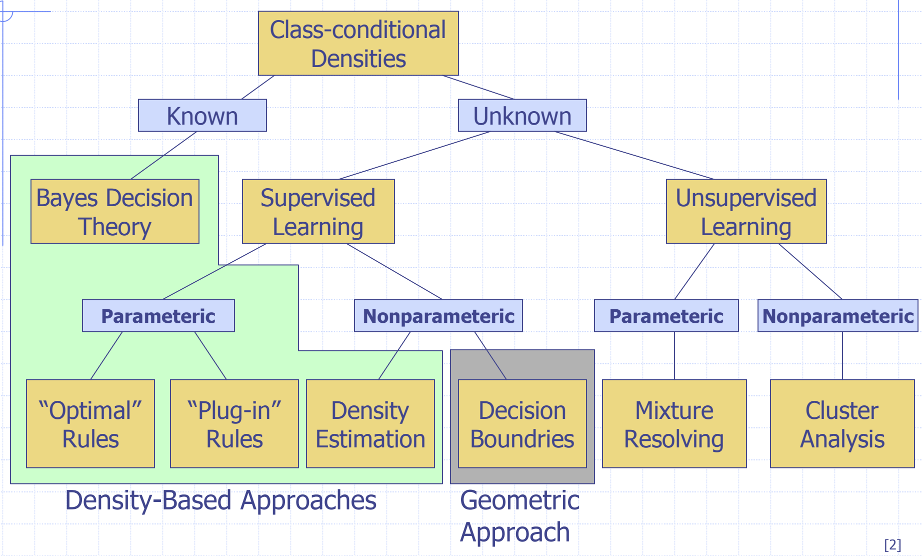


Sometimes local effects need to be taken into account.

[1]

3

Various approaches



[2]

4

Nonparametric Density Estimates

- Parametric forms involve a functional form of the density, and fitting parameters of the function via estimation from the sample. Typically, a Gaussian is used:

$$p(x) = \frac{1}{\sqrt{2\pi|\Sigma|}} \exp-\tfrac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)$$

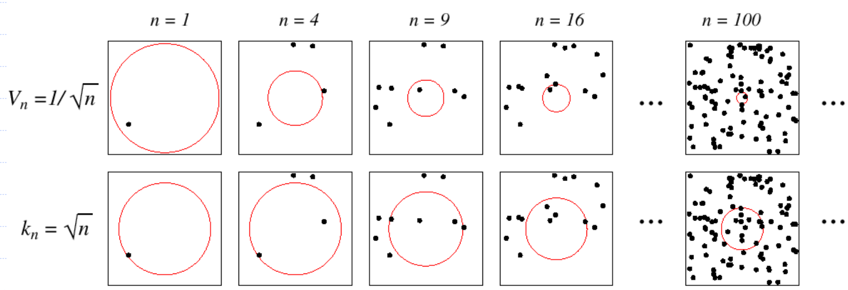
- Non-parametric methods do not use a parametric functional form of the density. Popular versions:
  - Parzen windows or kernel estimate
  - k-nearest neighbor estimate

[4]

5

Leading methods for estimating density at a point

Parzen-windows



k-nearest neighbors

[1]

6

Bias and Variance of KNN & Parzen

- Fukunaga gives the bias and variance of

Parzen $\hat{p}(x) = \frac{\mathbf{k}(x)}{Nv}$ $\quad \mathrm{var}[\hat{p}(x)] \approx \frac{p^2(x)}{k}$

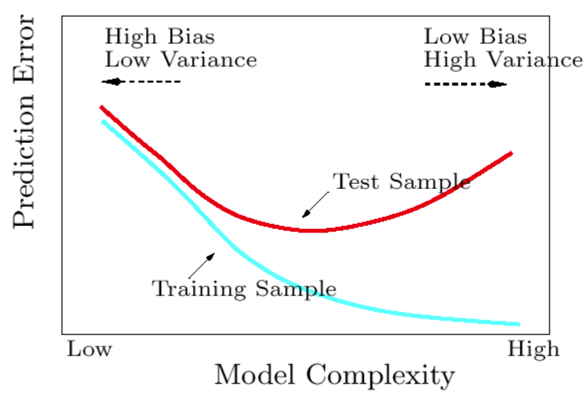
knn $\hat{\mathbf{p}}(x) = \frac{(k-1)}{N\mathbf{v}(x)}$ $\quad \mathrm{var}[\hat{p}(x)] \approx \frac{p^2(x)}{k}$

In Parzen windows, k(x) is a random variable, and v is held fixed. In knn, v(x) is a random variable, and k is held fixed.

In both cases, increasing the number of nearest neighbors *k* used to reduce variance. But this increases **v,** giving a coarser estimate of **p***(x)* and increasing bias.

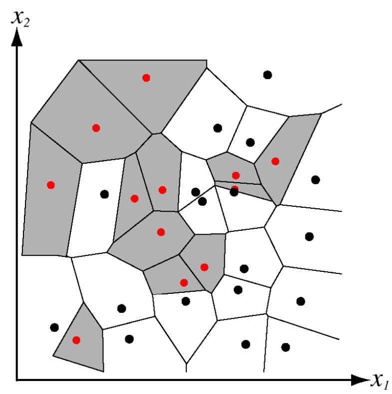
[4]

Bias-Variance Trade Off

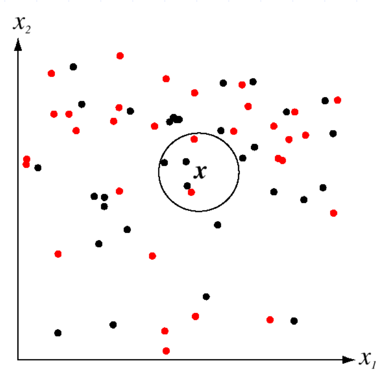


[3]

k=1 vs. k-nn



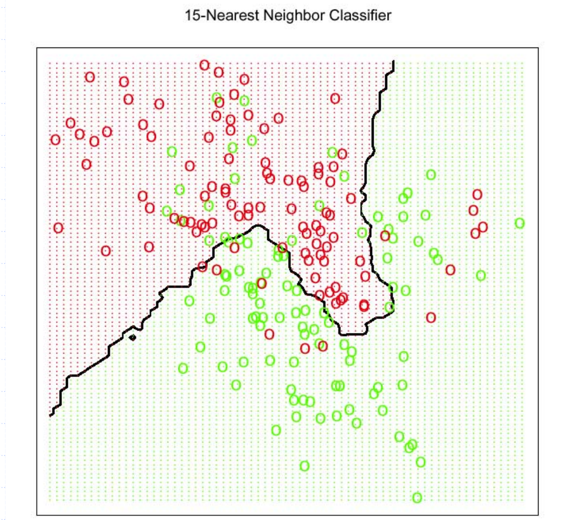
nearest neighbor



k-nearest neighbor

[1]

Nearest Neighbor Example (k=15)



[3]

Finding an f(x): Nearest Neighbor

- Nearest Neighbor tries to find f(x) directly from the training data:

$$\hat{f}(x) = \mathrm{Ave}\left(y_i \mid x_i \in N_k(x)\right)$$

- Where $N_k(x)$ is the neighborhood containing the k points in T closest to x.
- Approximations:
  - expectation is approximated by averaging over sample data
  - conditioning at a point is relaxed to conditioning on some region 'close' to the target point.
- Assumes f(x) is approximated by a locally constant function.

[3]

11

knn - library function in R

k-nearest neighbor classification for test set from training set. For each row of the test set, the `k' nearest (in Euclidean distance) training set vectors are found, and the classification is decided by majority vote, with ties broken at random. If there are ties for the `k'th nearest vector, all candidates are included in the vote.

Usage: knn(train, test, cl, k = 1, l = 0, prob = FALSE, use.all = TRUE)

train: matrix or data frame of training set cases.

test: matrix or data frame of test set cases. A vector will be interpreted as a row vector for a single case.

cl: factor of true classifications of training set

k: number of neighbors considered.

l: minimum vote for definite decision, otherwise `doubt'. (More precisely, less than `k-l' dissenting votes are allowed, even if `k' is increased by ties.)

prob: If this is true, the proportion of the votes for the winning class are returned as attribute `prob'.

use.all: controls handling of ties. If true, all distances equal to the `k'th largest are included. If false, a random selection of distances equal to the `k'th is chosen to use exactly `k' neighbors.

Value:

factor of classifications of test set. `doubt' will be returned as `NA'.

[6]

12

k-nearest neighbors in R

```
p <- as.matrix(Pop[, -3])
tp <- Pop$y

xp <- seq(-6.0, 10.0, length = 10); np <- length(xp)
yp <- seq(-6.0, 10.0, length = 10)

pt <- expand.grid(x1 = xp, x2 = yp) # from library(nnet)

par(mfcol=c(1,2))

Z <- knn(p, pt, tp, k = 1) #from library(class)
decplot(xp, yp, class.ind(Z),"k=1")
```

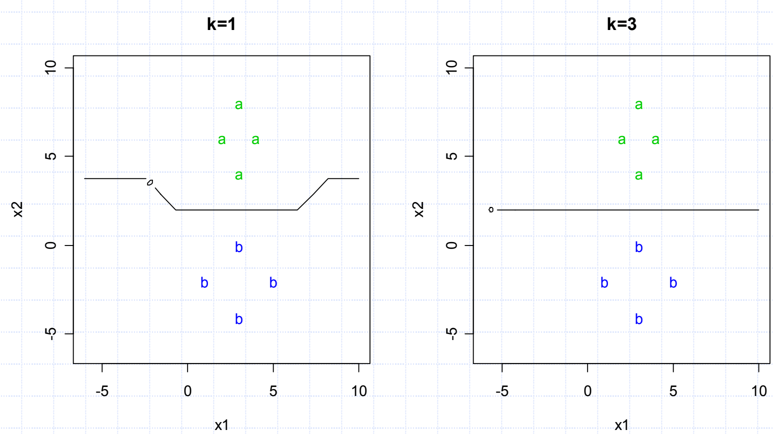
[5]

Functions in R

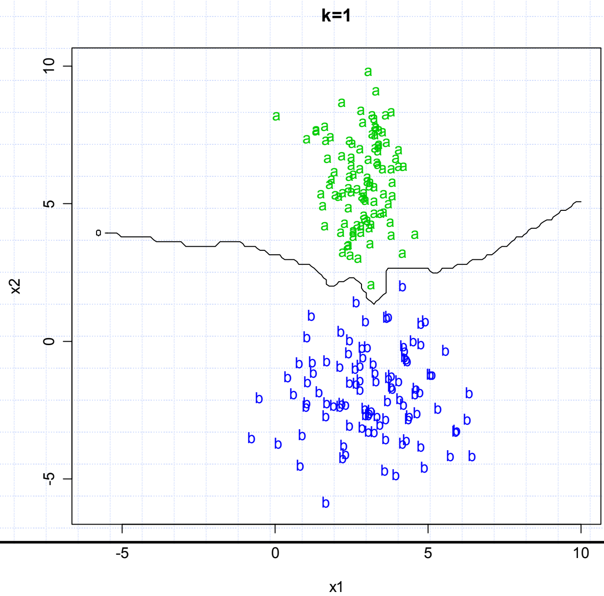
```
decplot <- function(xp, yp, Z, t)
{
    plot(Pop[, 1], Pop[, 2], xlim=c(-6.0,10), ylim=c(-6,10),
                 type = "n", xlab = "x1", ylab = "x2")
    title(t)
    for(il in 1:2) {
         set <- Pop$y==levels(Pop$y)[il]
         text(Pop[set, 1], Pop[set, 2],
                 labels = as.character(Pop$y[set]), col = 2 + il) }
    zp <- Z[, 1] - Z[, 2]
    contour(xp, yp, matrix(zp, np), add = T, levels = 0, labex = 0)
    invisible()
}
```

[5]

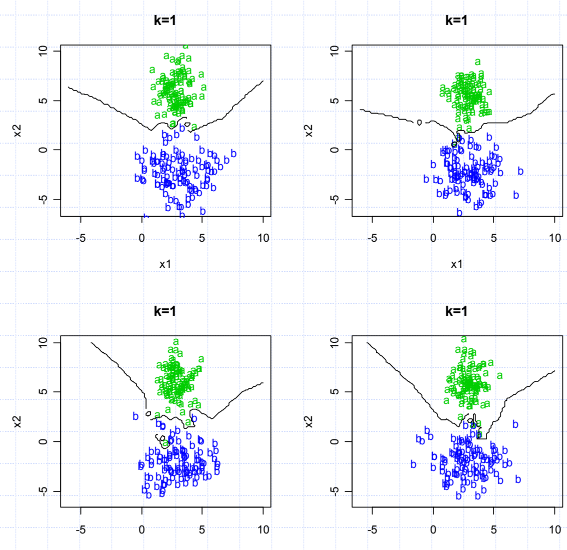
k-nearest neighbors



k=1 : maximize local effects



Variance in k = 1



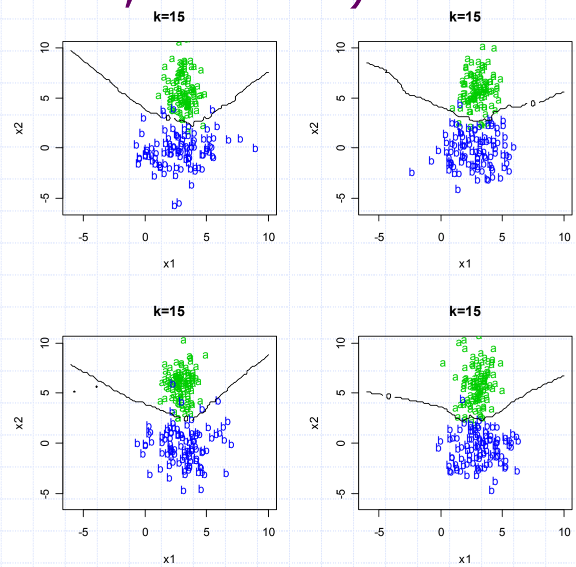
k=15 (ex10a.dat,ex10b.dat)

\> c1.m

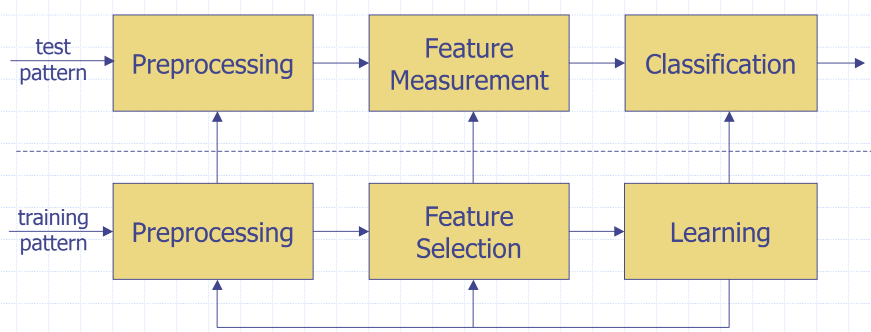
	[,1]	[,2]
[1,]	3	0
[2,]	0	6

\> c2.m

	[,1]	[,2]
[1,]	3	0
[2,]	0	0

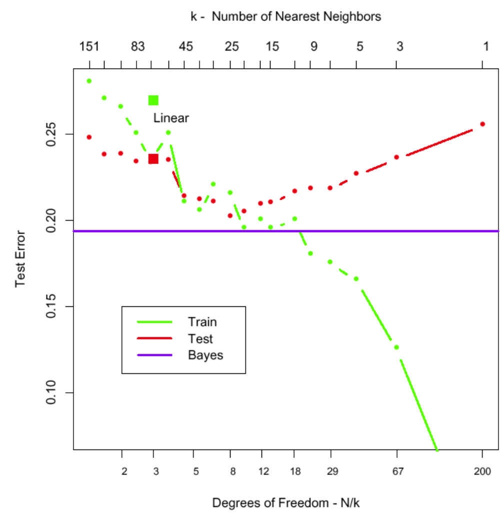


Model for machine learning



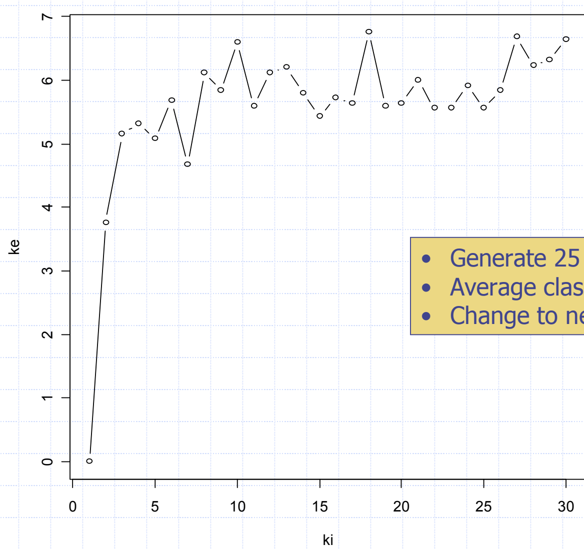
[2]

Classification Error



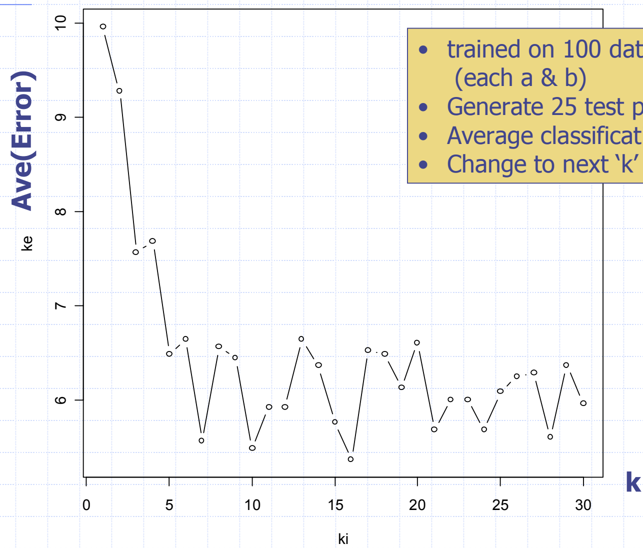
[3]

Bias on training set



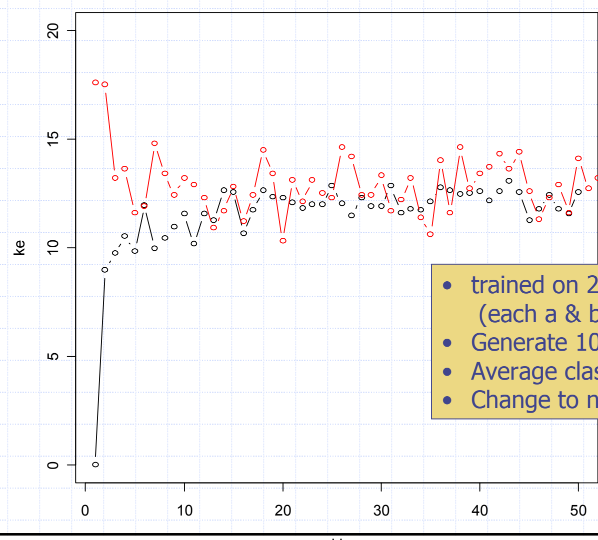
- Generate 25 test populations
- Average classification error
- Change to next 'k' (1-30)

Variance: error by ‘k’ on Test sets



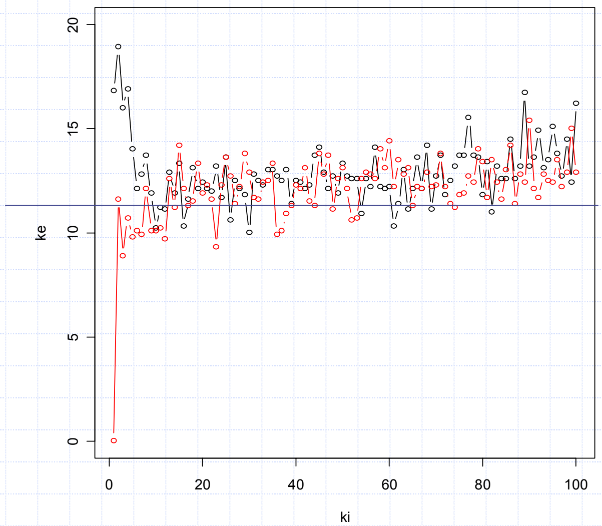
- trained on 100 data points (each a & b)
- Generate 25 test population
- Average classification error
- Change to next ‘k’ (1-30)

Mixture of Gaussians should converge on Bayes limit…



- trained on 200 data points (each a & b)
- Generate 10 test population
- Average classification error
- Change to next ‘k’ (1-50)

Diminishing returns



K-NN and Parzen Density Estimates

- In the Parzen window (uniform kernel) estimate, kernel volume is fixed and we counted the number of samples falling inside the volume to estimate p(x).
- In the K-nearest neighbor estimator, we choose a point x at which we wish to estimate the density, and construct the smallest region L(x) that contains k points. Then estimate the density at x as

$$\hat{p}(x) = \frac{k-1}{N\ V(x)}$$

where N is the total number of points, V(x) the volume of the minimal n-dim region containing *k* points, and the numerator is *k-1* so that the estimate is roughly unbiased.

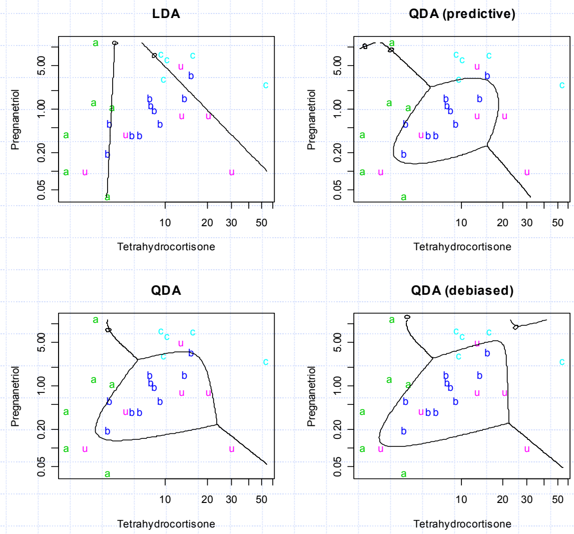
If *r* is the distance from *x* to the k^{th} nearest neighbor, then we can take *V(x)* to be the volume inside the n-sphere of radius *r*

$$V = \frac{\pi^{n/2}}{\Gamma\left(\frac{n+2}{2}\right)}\ r^n$$

where Γ is the Euler gamma function.

[4]

Next Time:
Multiple classes & Mixtures



[5]

Next Week:

- Exam-Project Starts Monday
- Dongmao Zhang (Ben-Amotz Group)

D. Zhang and D. Ben-Amotz, "Enhanced Chemical Classification of Raman Images with of Strong Fluorescence Interference", Appl. Spectrosc., 54 (2000) 1379-83.

References

[1] "Pattern Classification", Duda, Hart, Stork, John Wiley Sons, (2001).

[2] "Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings", Lipinski, C. A., Lombardo, F., Dominy, B. W., & Feeney, P. J. *Advanced Drug Delivery Reviews*, 23, 3-25 (1997).

[3] "The Elements of Statistical Learning: Data Mining, Inference and Prediction", Hastie, Tibshirani and Friedman, Springer-Verlag, (2001).

[4] "*Statistical Pattern Recognition*", K. Fukunaga, 2nd Ed. Academic Press, 1990 (Purdue Electrical Engineering)

[5] "Modern Applied Statistics in S", W.N. Venables, B.D. Ripley - 4th Ed, Springer-Verlag, 2002

[6] R-Documentation (www.r-project.org)