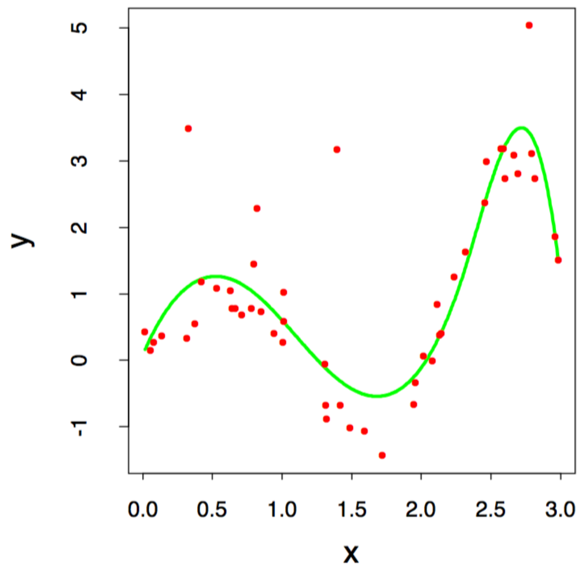
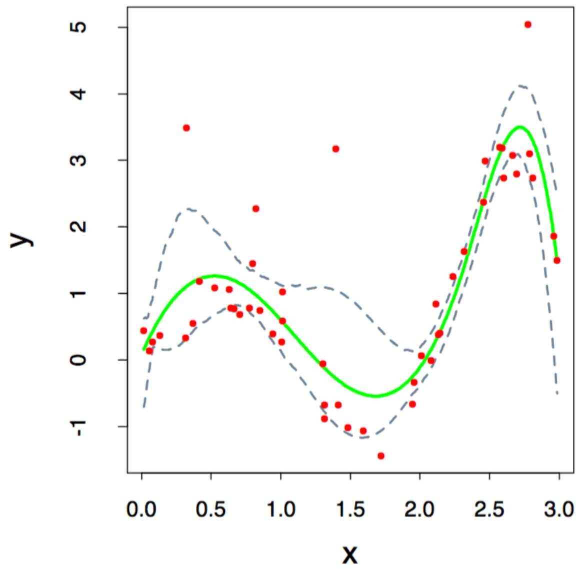
$\mathcal{D}$ $\hat{f}(\mathscr{D})$

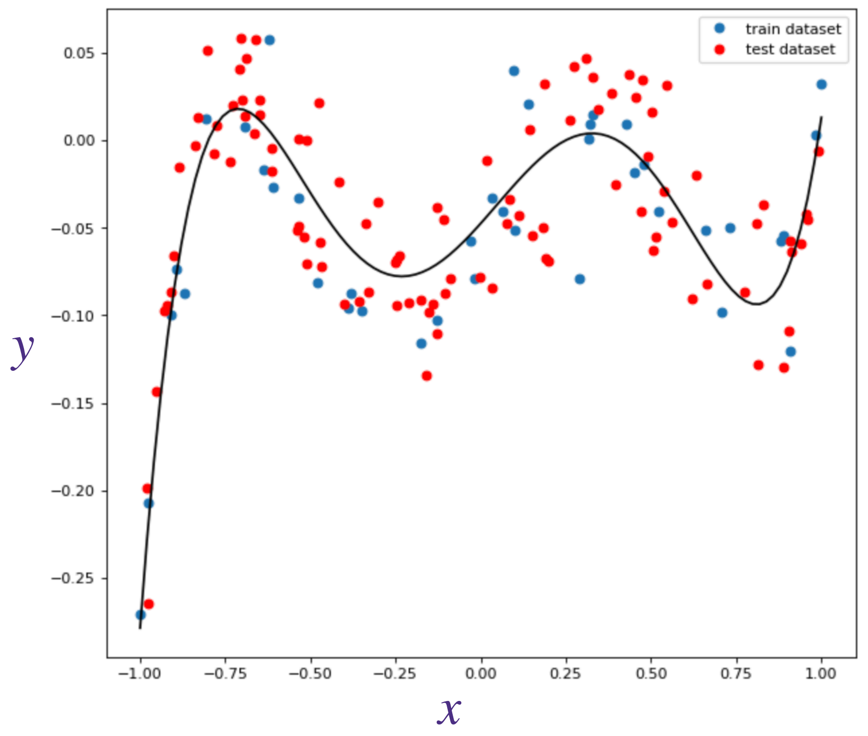


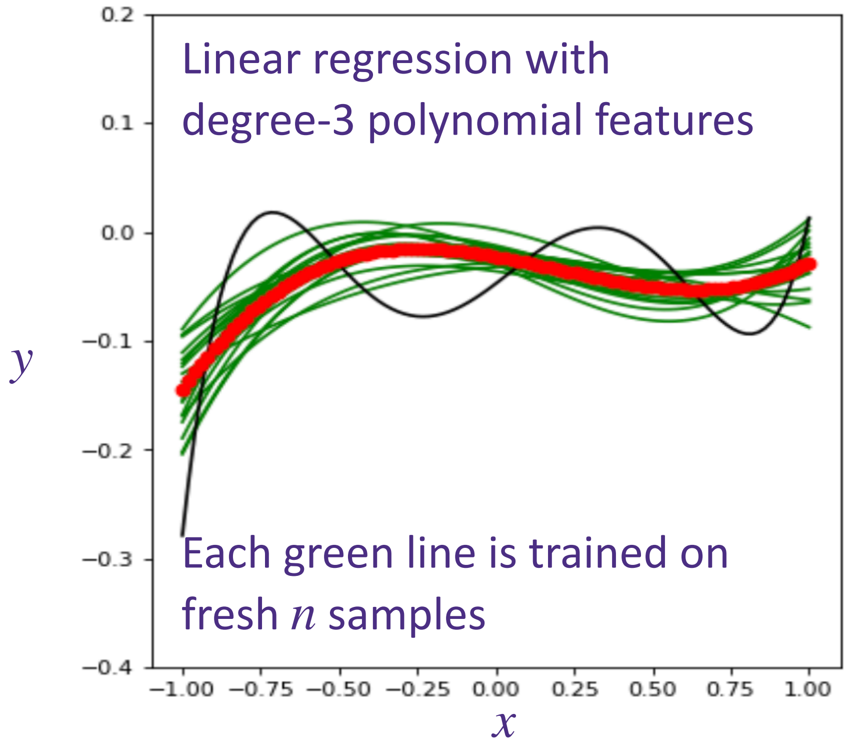
90% confidence interval



Bootstrap

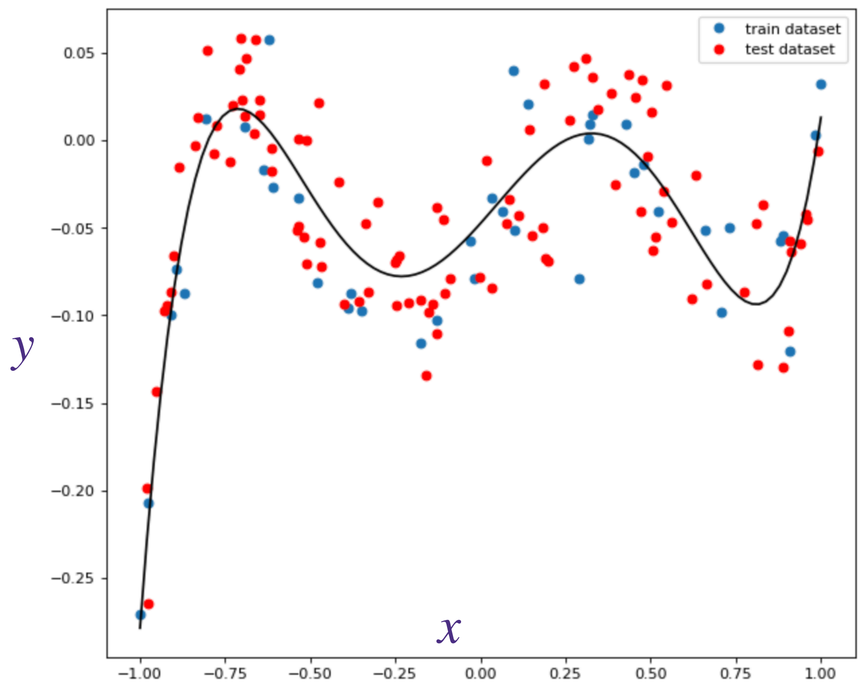
Recall bias-variance tradeoff

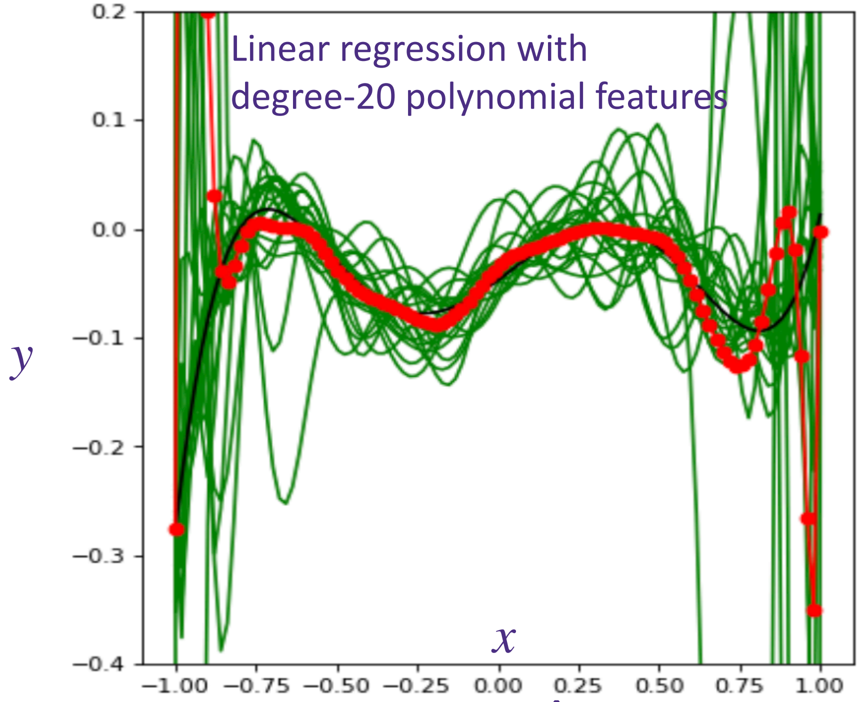




- When we want to evaluate the quality of our estimated $\hat{w}$, we would like to be able to have (many) **fresh samples** of size n , i.i.d. sampled from the ground truths distribution $(x, y) \sim P_{X,Y}$
- Then, we can draw the conclusion that, say, this model has **small variance**

Recall bias-variance tradeoff





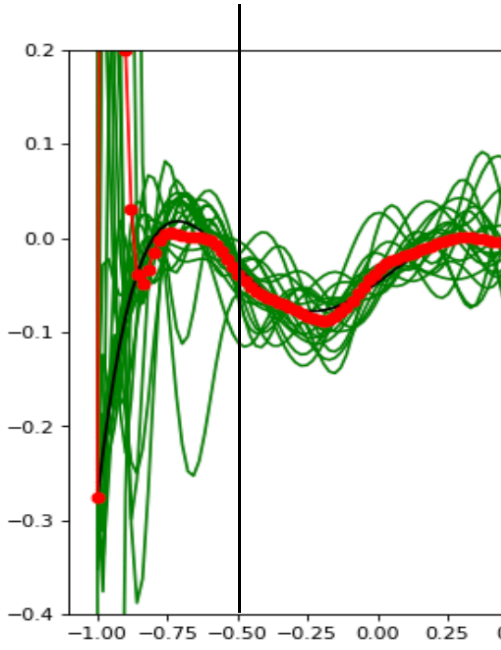
- When we want to evaluate the quality of our estimated $\hat{w}$, we would like to be able to have (many) **fresh samples** of size n , i.i.d. sampled from the ground truths distribution $(x, y) \sim P_{X,Y}$
- Then, we can draw the conclusion that, say, this model has **large variance** (and much more, e.g., variance is larger when $x \simeq 1.0$)

Motivation for Bootstrap methods

being able to draw fresh samples
from the ground truths distribution $P_{X,Y}(x, y)$
is quite useful in analyzing the quality of our estimation

As we cannot get fresh samples in practice, we resorted to Cross-validation

- Cross validation estimates the test error $\mathbb{E}[(\hat{w}^T x - y)^2]$, averaged over $(x, y) \sim P_{X,Y}$, but has limitations
  - Test error is informative, but how accurate is this number? (e.g., 3/5 heads vs. 30/50)
  - How do I get confidence intervals on statistics like the median or variance of a distribution?
  - Instead of the error for the entire dataset, what if I want to study the error for a *particular example* x?



The Bootstrap: Developed by Bradley Efron in 1979.

- The name is from “pull oneself up by one’s bootstraps”
- Bootstrap can estimate, for example, $\mathbb{P}_{y,\mathcal{D}_n}[y > \hat{w}_{\mathrm{LS}}^T x + 0.01 \mid x]$

(Non-parametric) Bootstrap method

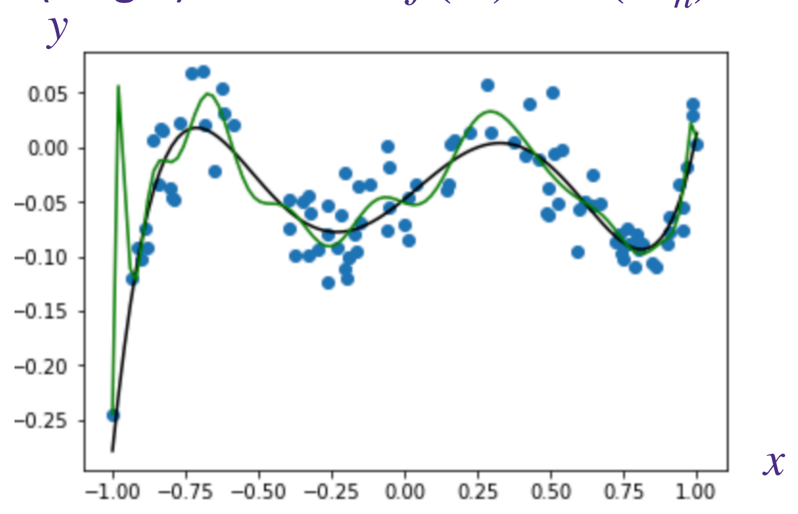
Real World

- (Unknown) true distribution $P_{X,Y}(x, y)$

- (Single) dataset i.i.d. from $P_{X,Y}$

  $\mathscr{D}_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$

- (Single) Estimator $\hat{f}(\,\cdot\,) = h(\mathscr{D}_n)$



Bootstrap World

- (Known) “true” distribution is empirical dist. $\mathscr{D}_n$

  $$\hat{P}_n(x, y) = \frac{1}{n}\sum_{i=1}^{n} \delta_{(x_i, y_i)}$$

- (Multiple resampling) dataset i.i.d. from $\hat{P}_n$

  $\mathscr{D}_n^{(b)} = \{(x_1^{(b)}, y_1^{(b)}), \ldots, (x_n^{(b)}, y_n^{(b)})\}$

  for $b = 1, 2, \ldots, B$

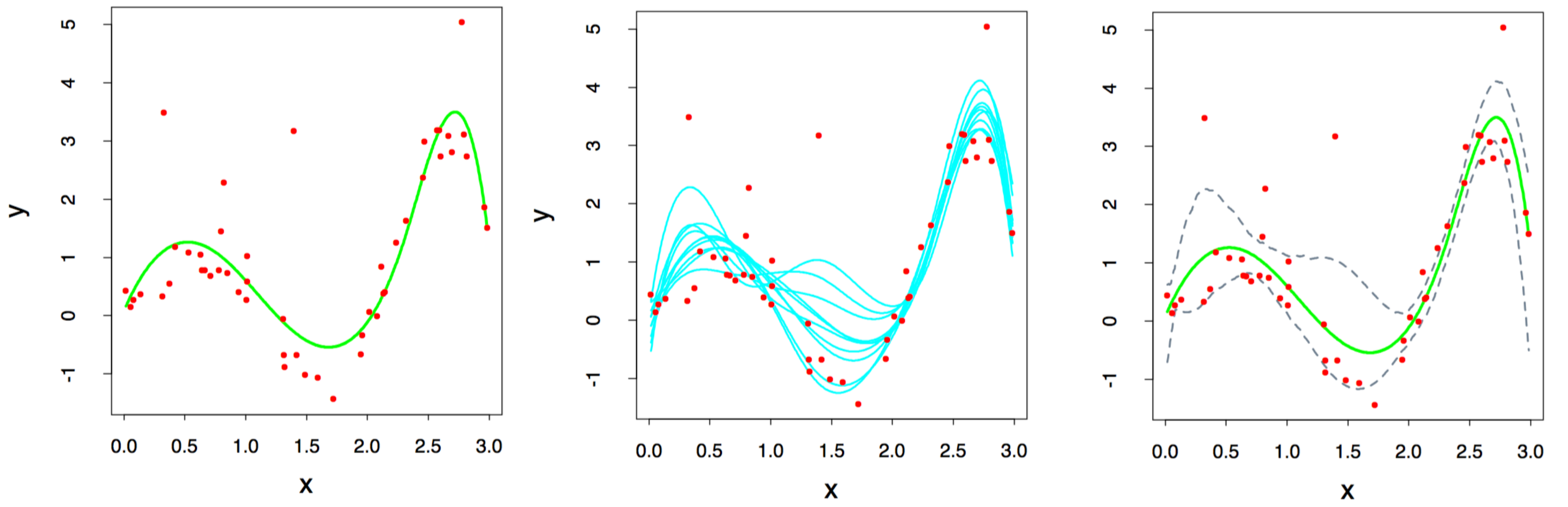
- (Multiple) Estimator $\hat{f}^{(b)}(\,\cdot\,) = h(\mathscr{D}_n^{(b)})$

Applications of Bootstrap

Common applications of the bootstrap:

- **Estimate parameters that escape simple analysis like the variance or median of an estimate**
- **Confidence intervals**
- **Estimates of error for a particular example x**

$\mathcal{D}$ $\hat{f}(\mathcal{D})$ $\hat{f}^{(b)} = \hat{f}(\mathcal{D}^{(b)})$ for $b \in \{1,\ldots,10\}$ 95% confidence interval



Figures from Hastie et al.

the largest value ν such that $\frac{1}{B}\sum_{b=1}^{B} \mathbf{1}\{\widehat{f}_b(x) \le \nu\} \le .05$,

Takeaways

Advantages:

- Bootstrap is very generally applicable.
  Build a confidence interval around *anything*
- Very simple to use
- Appears to give meaningful results even when the amount of data is very small

Takeaways

Advantages:

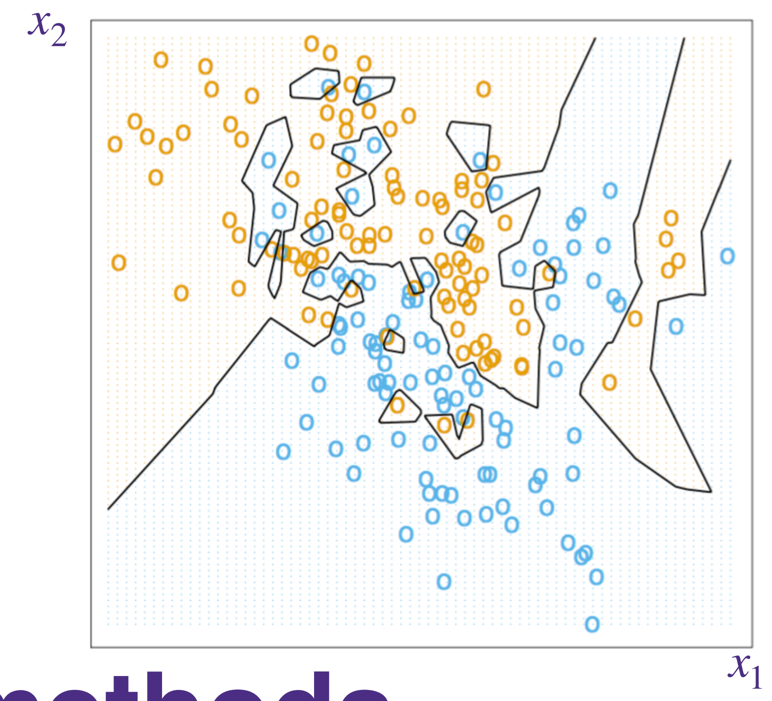
- Bootstrap is very generally applicable.
  Build a confidence interval around *anything*
- Very simple to use
- Appears to give meaningful results even when the amount of data is very small

Disadvantages

- Potentially computationally intensive
- Reliability relies on test statistic and rate of convergence of empirical CDF to true CDF, which is unknown (so we do not know how good Bootstrap is)
- Poor performance on “extreme statistics” (e.g., the max)

Further reading

- “Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning”, Yarin Gal, Zoubin Ghahramani, ICML 2016



Nearest neighbor methods

One way to approximate optimal classifier = local statistics

- Consider an example of binary classification on 1-dimensional $x \in \mathbb{R}$
- The problem is fully specified by the ground truths $P_{X,Y}(x,y)$
- Suppose for simplicity that $P_Y(y=+1) = P_Y(y=-1) = 1/2$

- What is the Bayes optimal classifier that minimizes $P(\hat{y} \neq y \mid x)$?

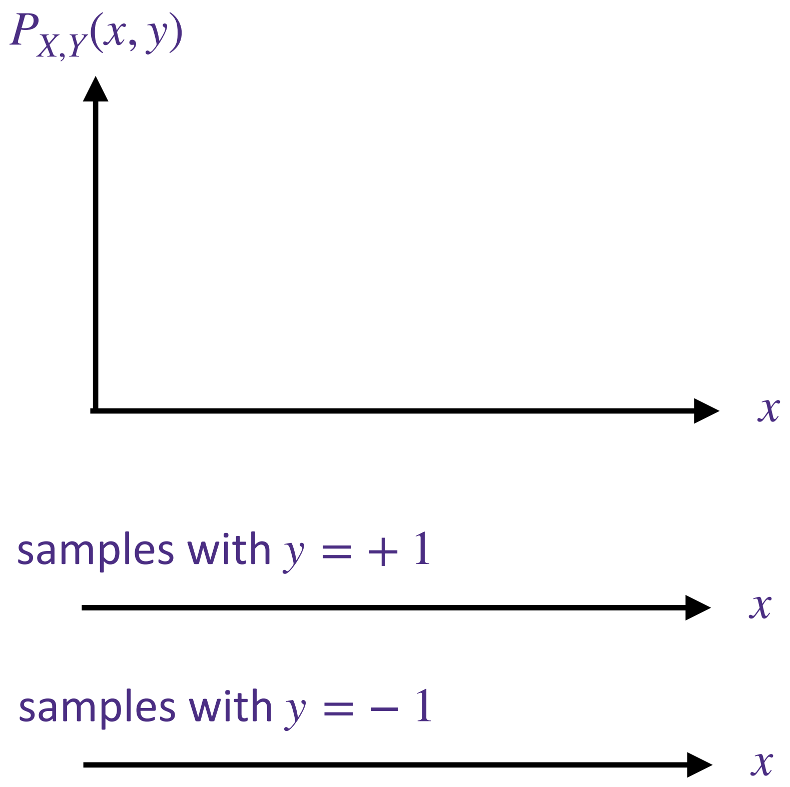
$$\hat{y} = +1 \text{ if } P(x,+1) > P(x,-1)$$
$$-1 \text{ if } P(x,+1) < P(x,-1)$$

samples with $y = +1$

- How do we compare $P(y=+1 \mid x)$ and $P(y=-1 \mid x)$ from samples?

samples with $y = -1$

One way to approximate Bayes Classifier = local statistics



- What is the Bayes optimal classifier that minimizes $P(\hat{y} \neq y \,|\, x)$?

$$\hat{y} = \begin{cases} +1 \text{ if } P(x,+1) > P(x,-1) \\ -1 \text{ if } P(x,+1) < P(x,-1) \end{cases}$$

- k-nearest neighbors classifier considers the k-nearest neighbors and takes a majority vote

$$\hat{y} = \begin{cases} +1, & \text{if (\# of +1 samples)} > \text{(\# of -1 samples)} \\ -1, & \text{if (\# of +1 samples)} < \text{(\# of -1 samples)} \end{cases}$$

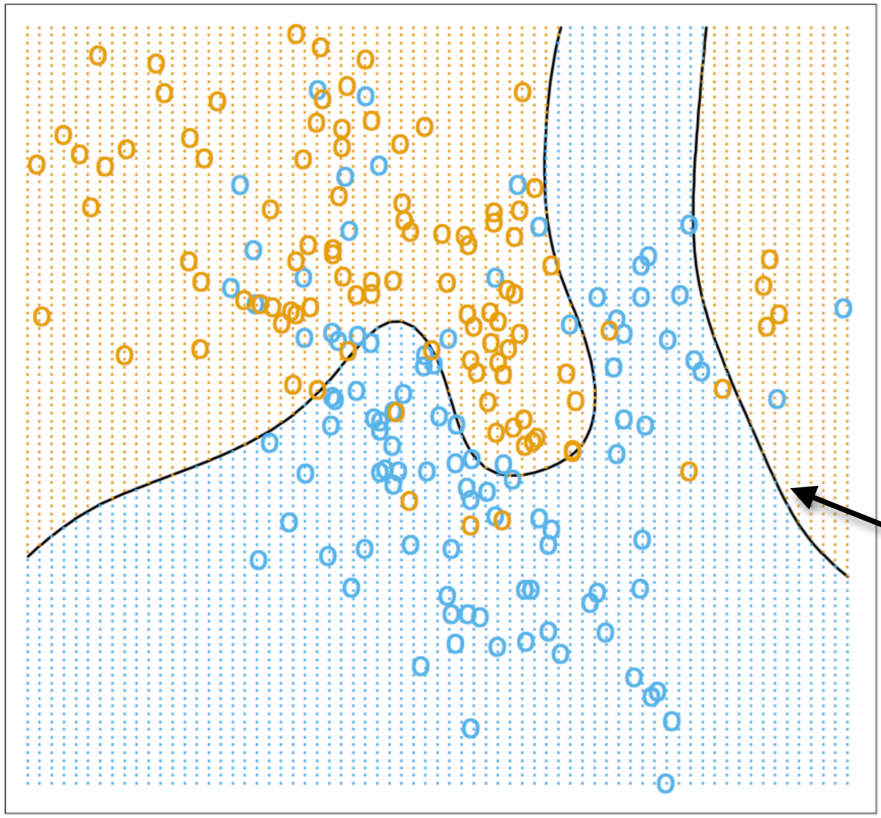
- Denote the k_r^+ as the number of samples within distance r from x with label $+1$, then

$$\frac{k_r^+}{n} \longrightarrow 2r \times P(x \,|\, y = +1)$$

as we increase n and decrease r.

- [R-D Reiss. Approximate distributions of order statistics: with applications to nonparametric statistics. Springer Science & Business Media, 2012.]

Some data, Bayes Classifier



Training data:

● True label: +1

● True label: -1

Optimal "Bayes" classifier:

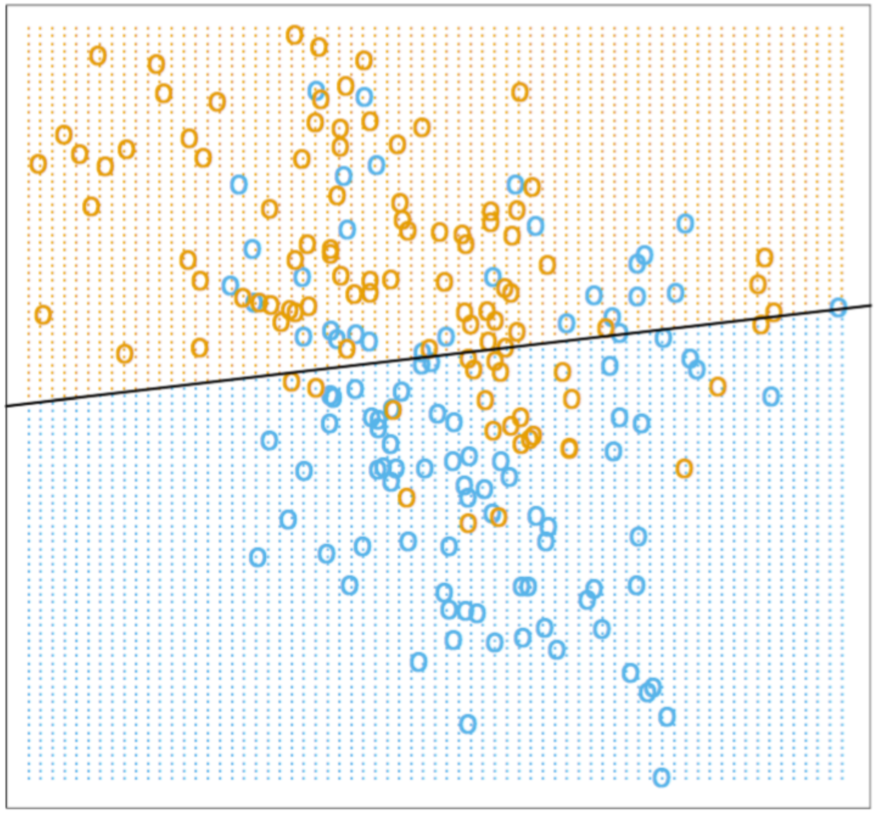
$$\mathbb{P}(Y = 1 | X = x) = \frac{1}{2}$$

▨ Predicted label: +1

▨ Predicted label: -1

Figures from Hastie et al.

Linear Decision Boundary



Training data:

● True label: +1

● True label: -1

Learned:

Linear Decision boundary

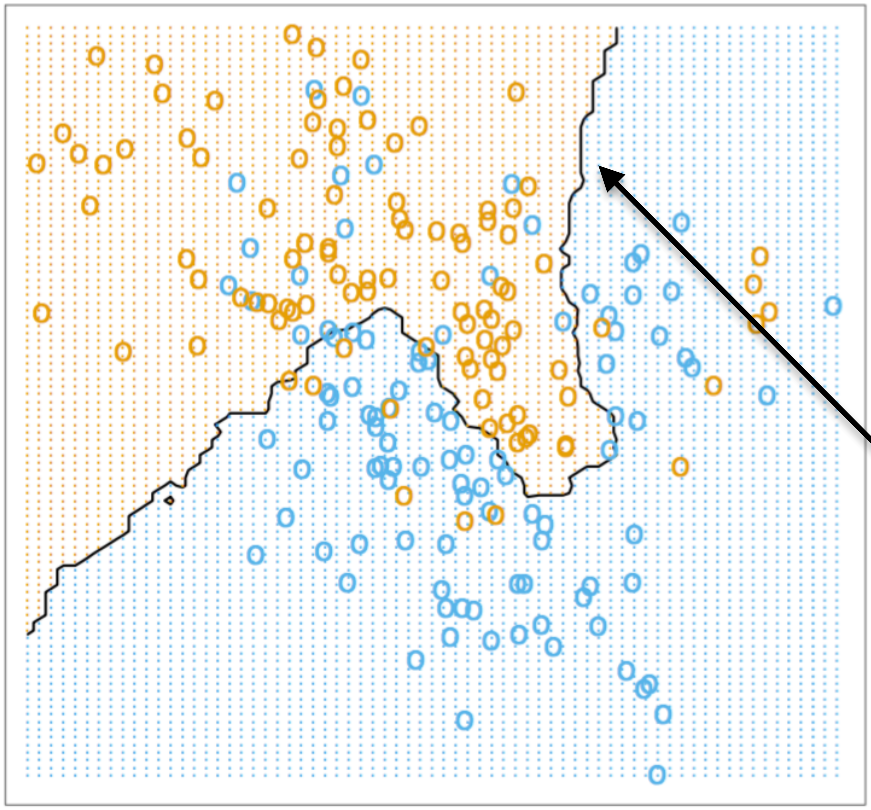
$$x^T w + b = 0$$

▨ Predicted label: +1

▨ Predicted label: -1

Figures from Hastie et al

15 Nearest Neighbor Boundary



Training data:

○ True label: +1

○ True label: -1

Learned:

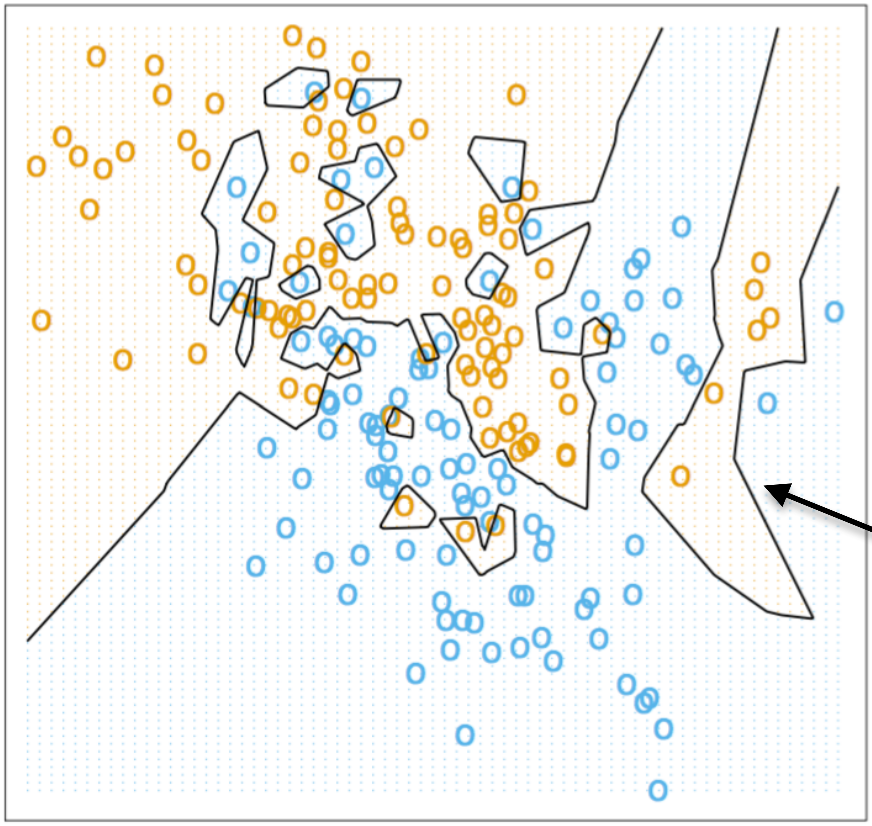
15 nearest neighbor decision boundary (majority vote)

▨ Predicted label: +1

▨ Predicted label: -1

Figures from Hastie et al

1 Nearest Neighbor Boundary



Training data:

● True label: +1

● True label: -1

Learned:

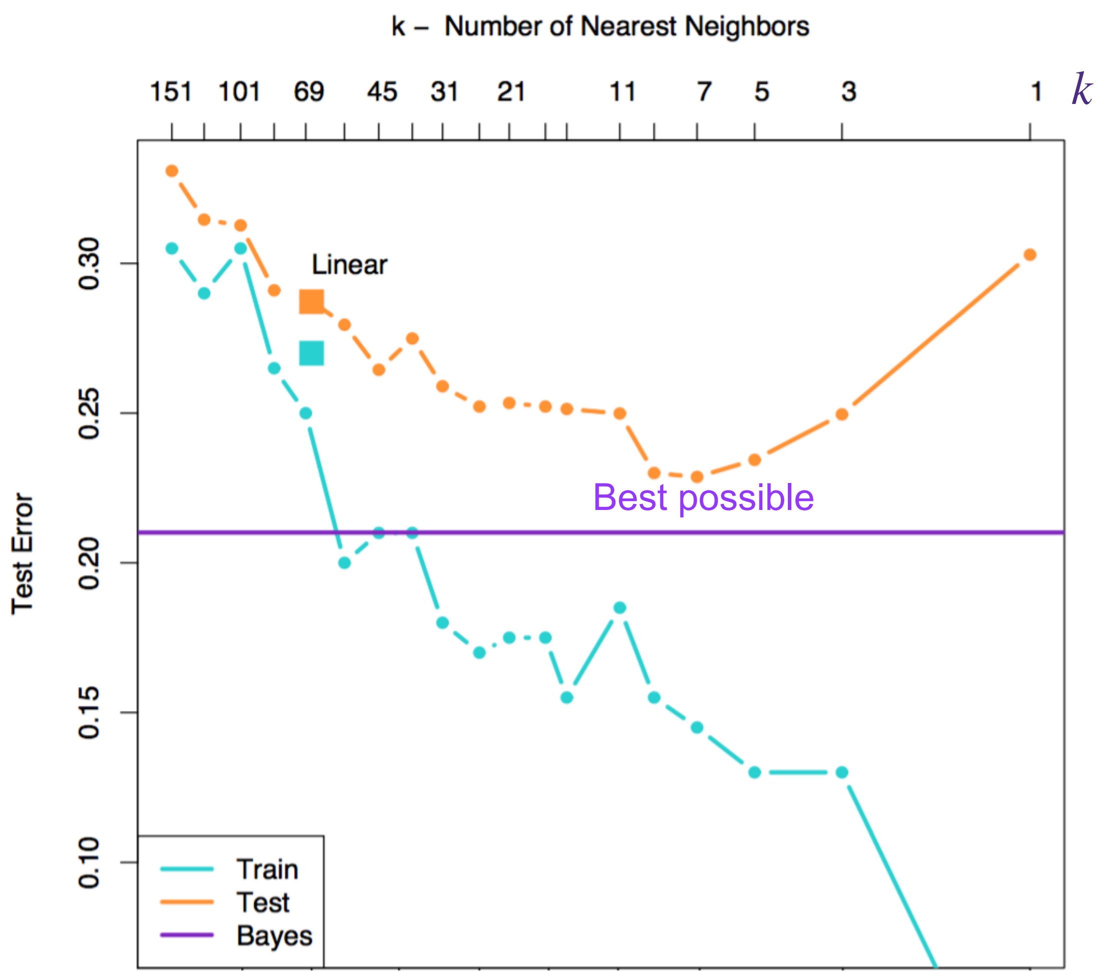
1 nearest neighbor decision boundary (majority vote)

▨ Predicted label: +1

▨ Predicted label: -1

Figures from Hastie et al

k-Nearest Neighbor Error



Bias-Variance tradeoff

As k->infinity?

Bias:

Variance:

As k->1?

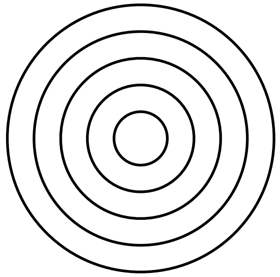
Bias:

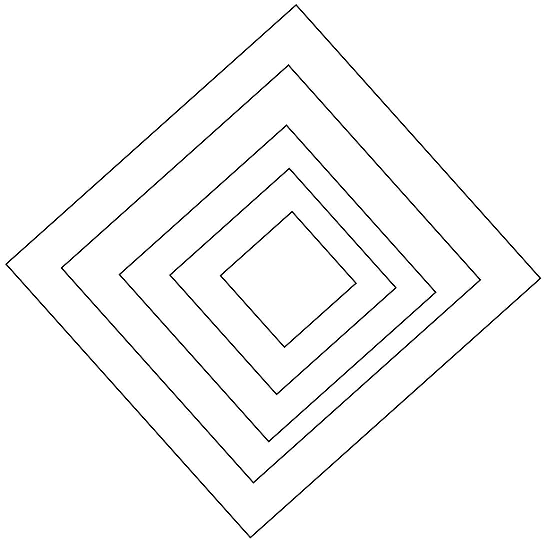
Variance:

Figures from Hastie et al

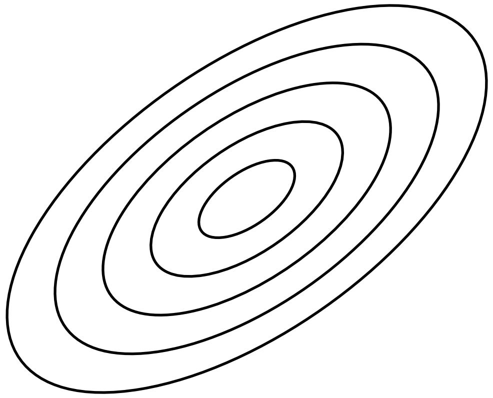
Notable distance metrics (and their level sets)

L_2 norm : $d(x, y) = \|x - y\|_2$

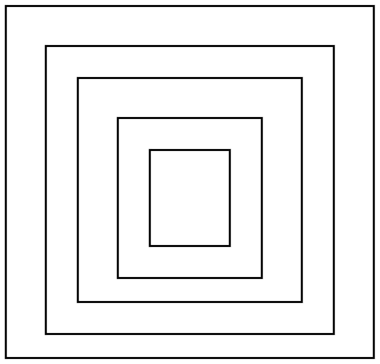




L_1 norm (taxi-cab)



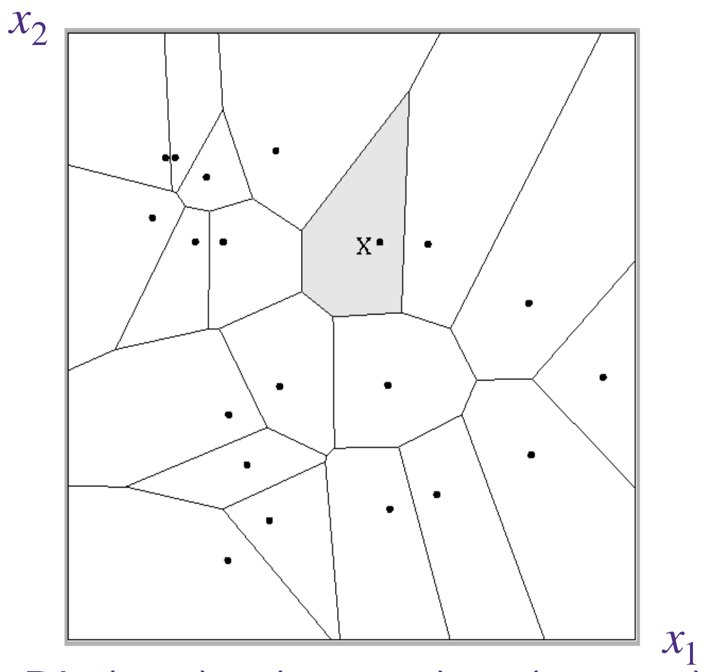
Mahalanobis norm: $d(x, y) = (x - y)^T M (x - y)$



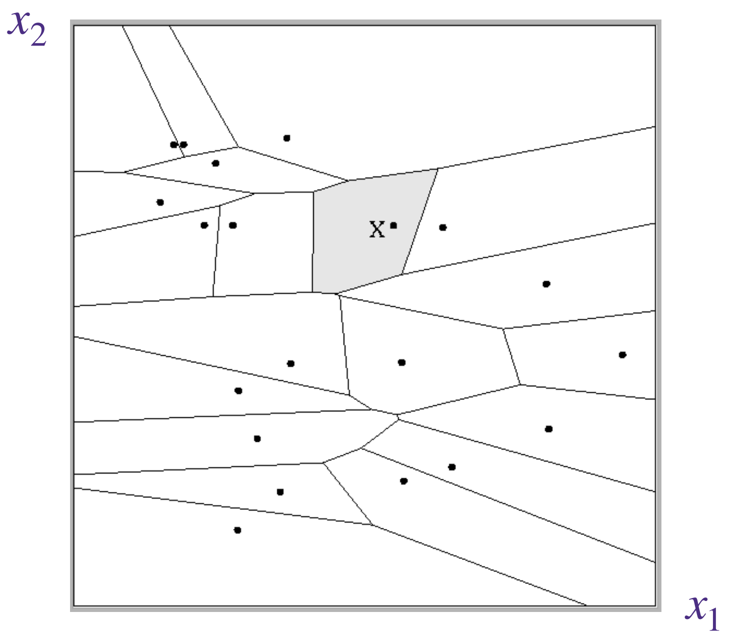
L-infinity *(max) norm*

1 nearest neighbor

One can draw the nearest-neighbor regions in input space.



$Dist(\mathbf{x}^i,\mathbf{x}^j) = (x^i_1 - x^j_1)^2 + (x^i_2 - x^j_2)^2$



$Dist(\mathbf{x}^i,\mathbf{x}^j) =(x^i_1 - x^j_1)^2+(3x^i_2 - 3x^j_2)^2$

The relative scalings in the distance metric affect region shapes

1 nearest neighbor guarantee - classification

$$\{(x_i, y_i)\})_{i=1}^n \qquad x_i \in \mathbb{R}^d, \quad y_i \in \{0,1\} \qquad (x_i, y_i) \overset{iid}{\sim} P_{XY}$$

Theorem[Cover, Hart, 1967] If P_X is supported everywhere in $\mathbb{R}^d$ and $P(Y = 1|X = x)$ is smooth everywhere, then as $n \to \infty$ the 1-NN classification rule has error at most twice the Bayes error rate.

1 nearest neighbor guarantee - classification

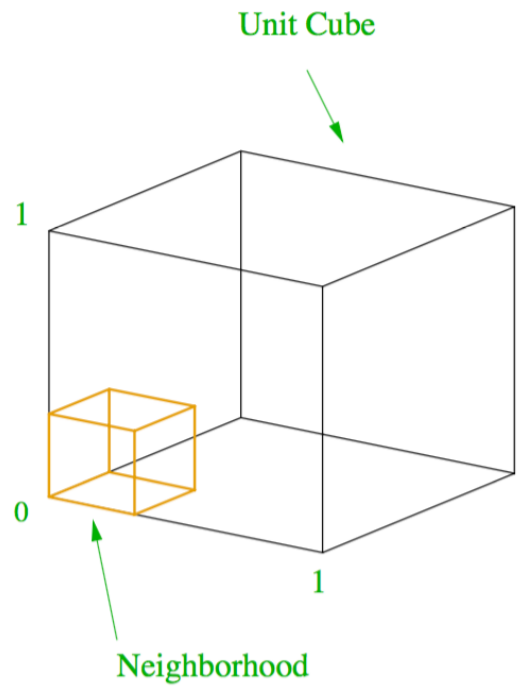
$$\{(x_i, y_i)\})_{i=1}^n \qquad x_i \in \mathbb{R}^d, \quad y_i \in \{0,1\} \qquad (x_i, y_i) \overset{iid}{\sim} P_{XY}$$

Theorem[Cover, Hart, 1967] If P_X is supported everywhere in $\mathbb{R}^d$ and $P(Y = 1|X = x)$ is smooth everywhere, then as $n \to \infty$ the 1-NN classification rule has error at most twice the Bayes error rate.

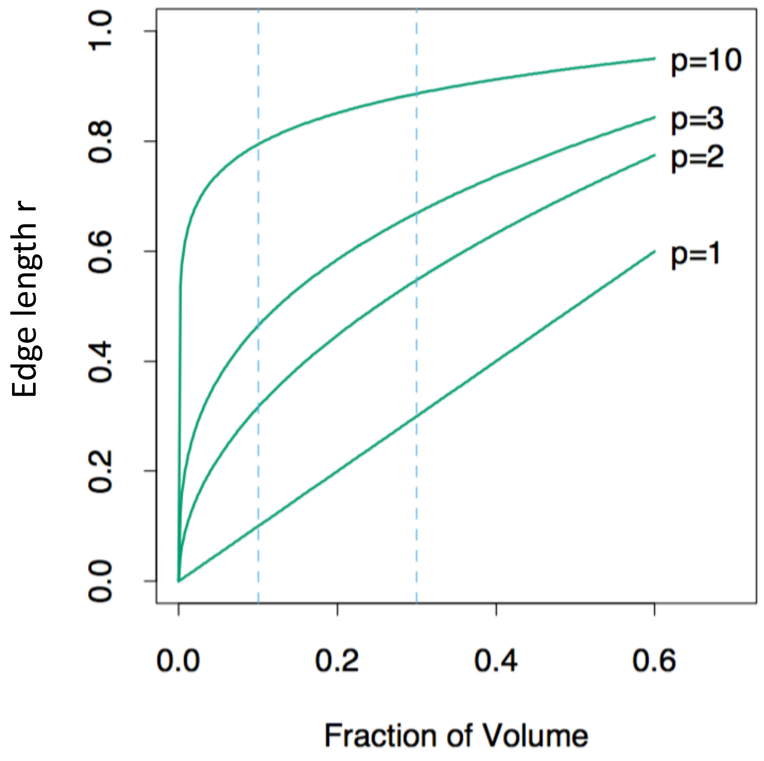
- Let x_{NN} denote the nearest neighbor at a point x
- First note that as $n \to \infty$, $P(y = +1 \,|\, x_{NN}) \to P(y = +1 \,|\, x)$
- Let $p^* = \min\{P(y = +1 \,|\, x), P(y = -1 \,|\, x)\}$denote the Bayes error rate
- At a point x,
    - Case 1: nearest neighbor is $+1$, which happens with $P(y = +1 \,|\, x)$ and the error rate is $P(y = -1 \,|\, x)$
    - Case 2: nearest neighbor is $+1$, which happens with $P(y = -1 \,|\, x)$ and the error rate is $P(y = +1 \,|\, x)$
- The average error of a 1-NN is

$$P(y = +1 \,|\, x)\, P(y = -1 \,|\, x) + P(y = -1 \,|\, x)\, P(y = +1 \,|\, x) = 2p^*(1 - p^*)$$

Curse of dimensionality Ex. 1



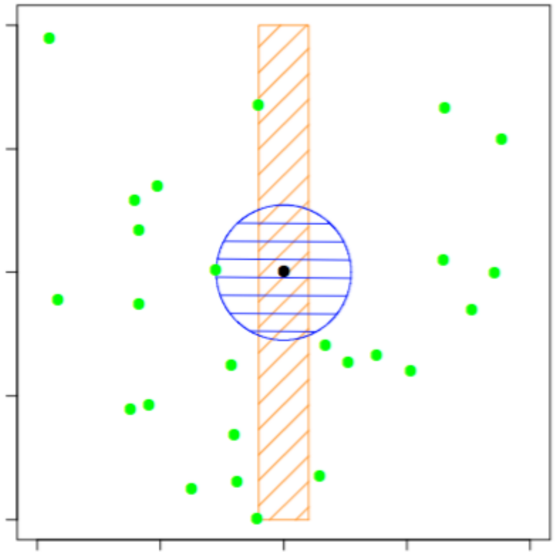
side length r

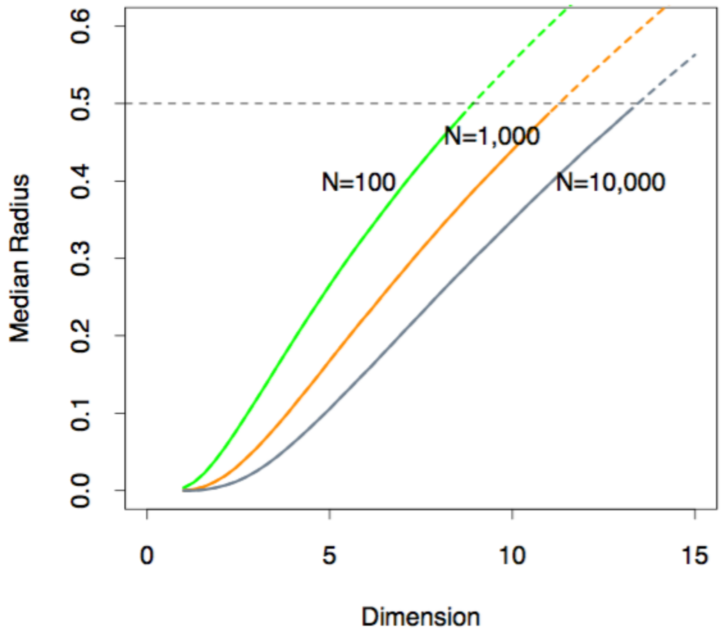


X is uniformly distributed over $[0,1]^p$. What is $\mathbb{P}(X \in [0,r]^p)$?

Curse of dimensionality Ex. 2

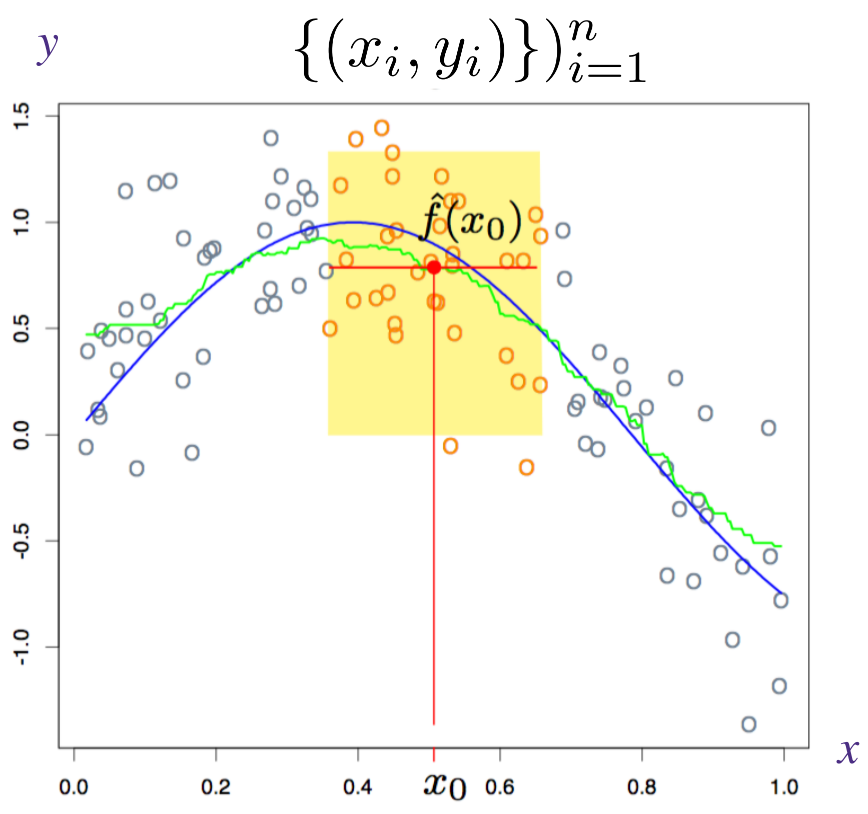
$\{X_i\}_{i=1}^n$ are uniformly distributed over $[-.5, .5]^p$.





What is the median distance from a point at origin to its 1NN?

Nearest neighbor regression



- What is the optimal classifier that minimizes MSE$\mathbb{E}[(\hat{y} - y)^2]$?
$$\hat{y} = \mathbb{E}[y \mid x]$$

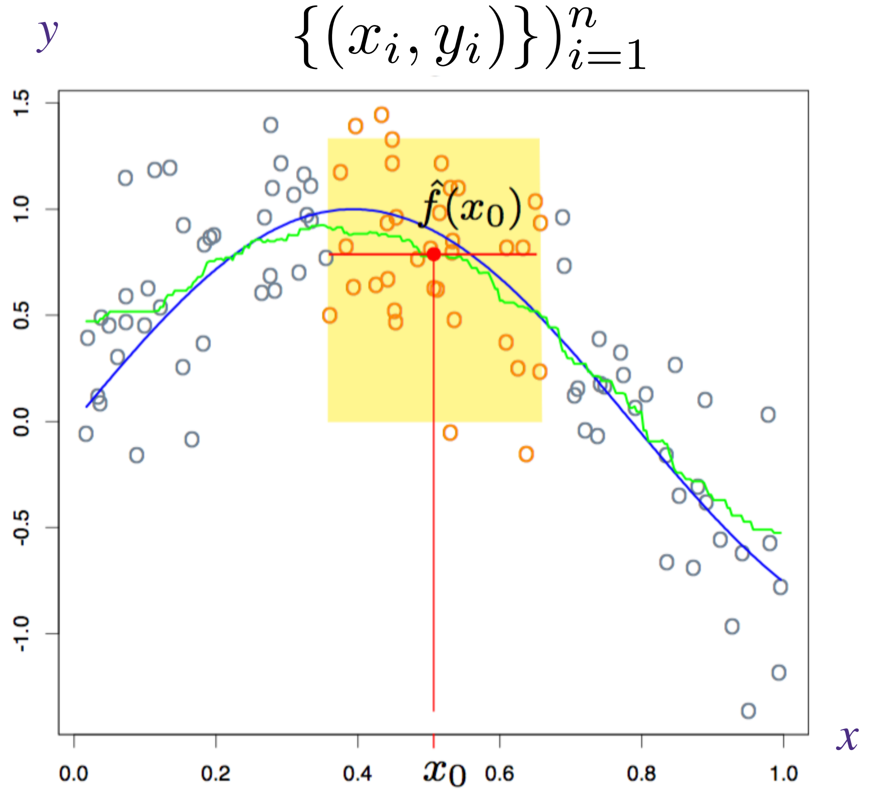
- Recall that
$$\frac{k_r^+}{n} \longrightarrow 2r \times P(x \mid y = +1)$$

- k-nearest neighbor regressor is
$$\hat{f}(x) = \frac{1}{k} \sum_{j \in \text{nearest neighbor}} y_j$$

$$= \frac{\sum_{i=1}^n y_i \times \mathrm{Ind}(x_i \text{ is a } k \text{ nearest neighbor})}{\sum_{i=1}^n \mathrm{Ind}(x_i \text{ is a } k \text{ nearest neighbor})}$$

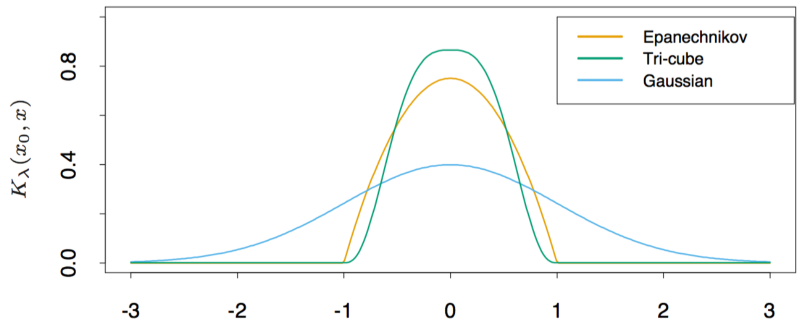
$\longrightarrow$

Nearest neighbor regression



Why are far-away neighbors weighted same as close neighbors!

smoothing: $K(x, y)$

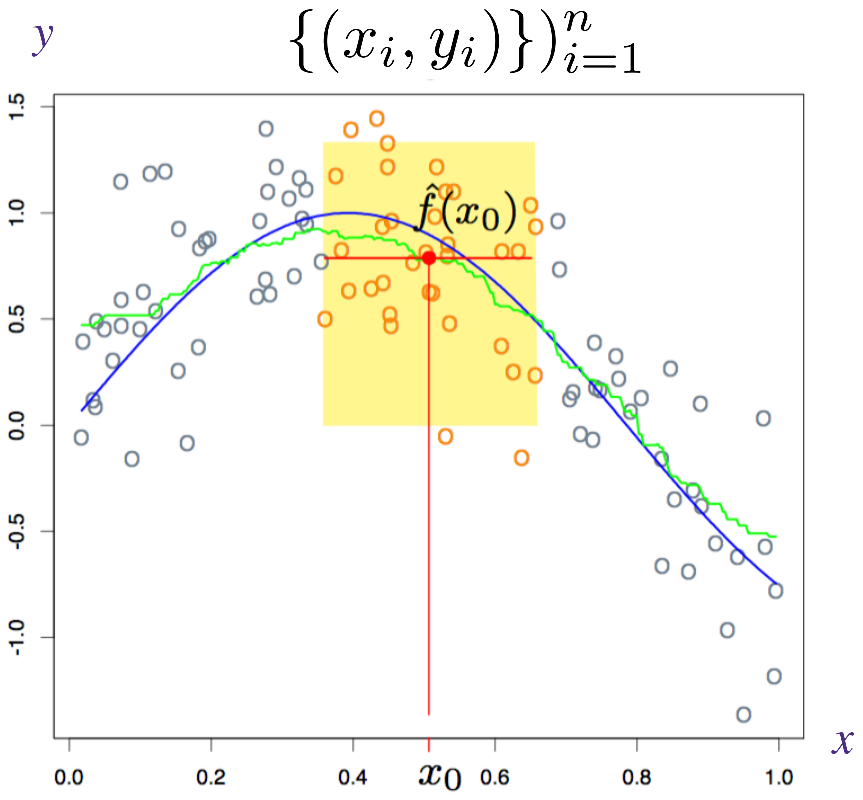


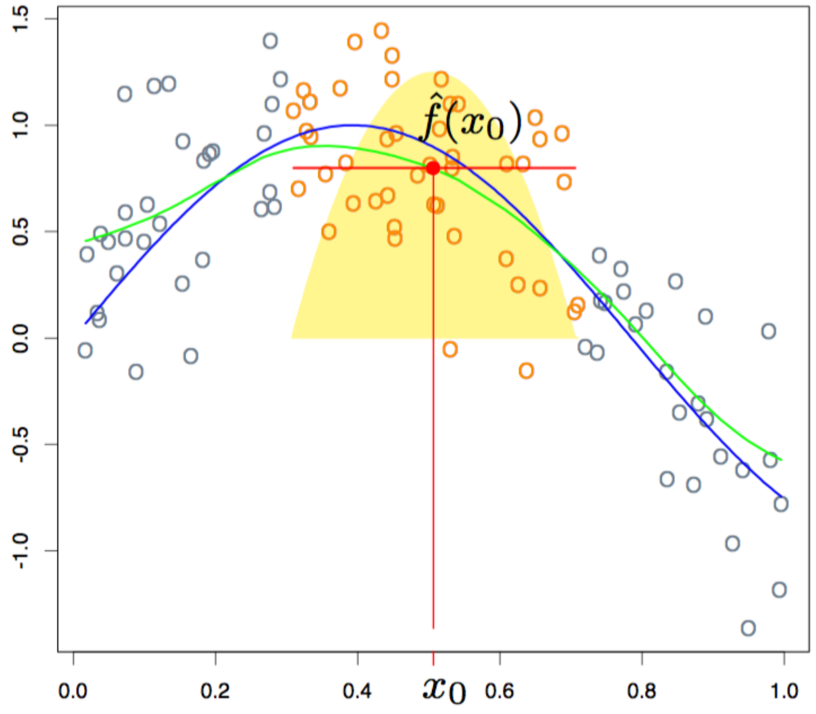
- k-nearest neighbor regressor is

$$\hat{f}(x_0) = \frac{1}{k} \sum_{j \in \text{nearest neighbor}} y_j$$

$$\widehat{f}(x_0) = \frac{\sum_{i=1}^n K(x_0, x_i) y_i}{\sum_{i=1}^n K(x_0, x_i)}$$

Nearest neighbor regression





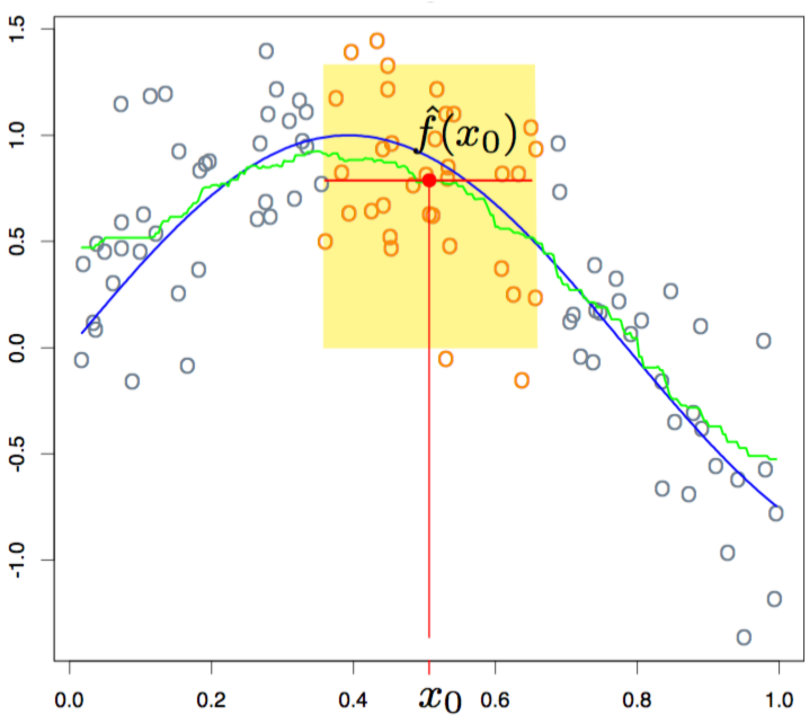
- k-nearest neighbor regressor is

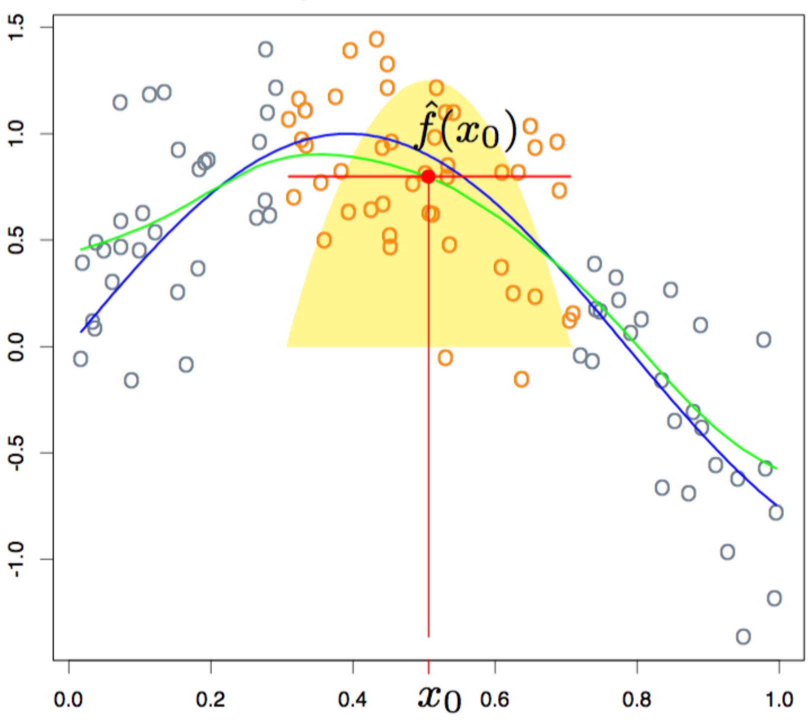
$$\hat{f}(x_0) = \frac{1}{k} \sum_{j \in \text{nearest neighbor}} y_j$$

$$\widehat{f}(x_0) = \frac{\sum_{i=1}^n K(x_0, x_i) y_i}{\sum_{i=1}^n K(x_0, x_i)}$$

Nearest neighbor regression

$$\{(x_i, y_i)\}_{i=1}^n$$





- k-nearest neighbor regressor is

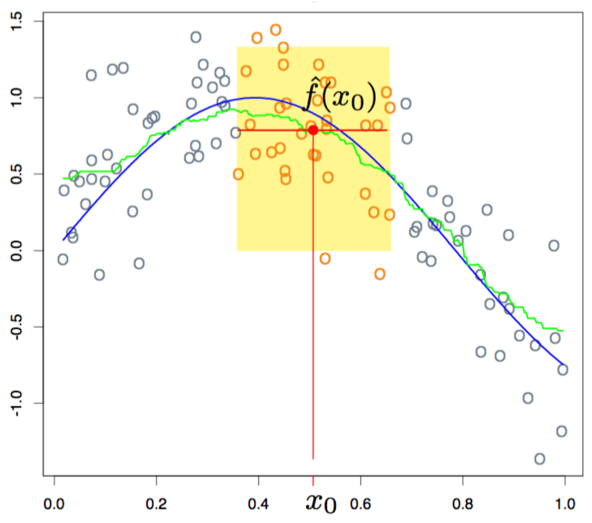
$$\hat{f}(x_0) = \frac{1}{k} \sum_{j \in \text{nearest neighbor}} y_j$$

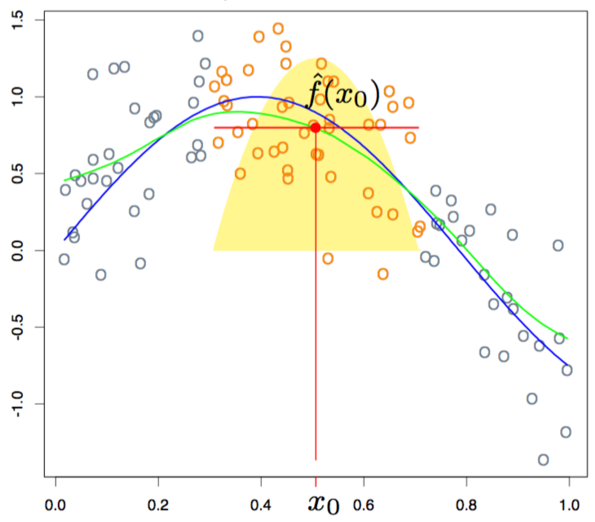
Why just average them?

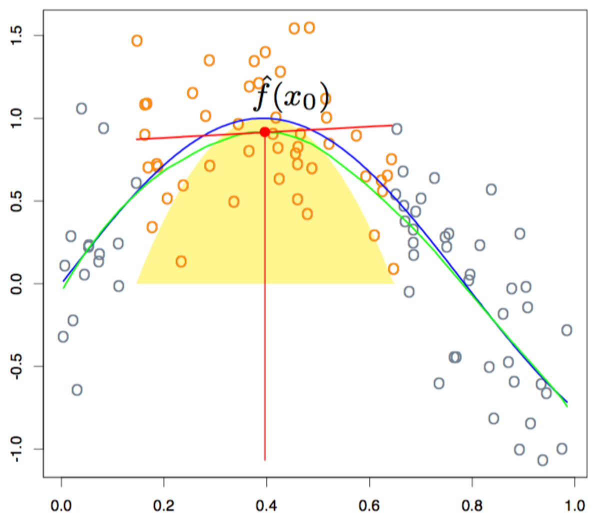
$$\widehat{f}(x_0) = \frac{\sum_{i=1}^n K(x_0, x_i) y_i}{\sum_{i=1}^n K(x_0, x_i)}$$

Nearest neighbor regression

$$\{(x_i, y_i)\}_{i=1}^n$$







$\mathcal{N}_k(x_0) = k$-nearest neighbors of x_0

$$\hat{f}(x_0) = \sum_{x_i \in \mathcal{N}_k(x_0)} \frac{1}{k} y_i$$

$$\hat{f}(x_0) = \frac{\sum_{i=1}^n K(x_0, x_i) y_i}{\sum_{i=1}^n K(x_0, x_i)}$$

$$\hat{f}(x_0) = b(x_0) + w(x_0)^T x_0$$

$$w(x_0), b(x_0) = \arg\min_{w,b} \sum_{i=1}^n K(x_0, x_i)(y_i - (b + w^T x_i))^2$$

Local Linear Regression

Nearest Neighbor Overview

- Very simple to explain and implement
- No training! But finding nearest neighbors in large dataset at test can be computationally demanding (KD-trees help)
- You can use other forms of distance (not just Euclidean)
- Smoothing and local linear regression can improve performance (at the cost of higher variance)
- With a lot of data, “local methods” have strong, simple theoretical guarantees.
- Without a lot of data, neighborhoods aren’t “local” and methods suffer (curse of dimensionality).

Questions?