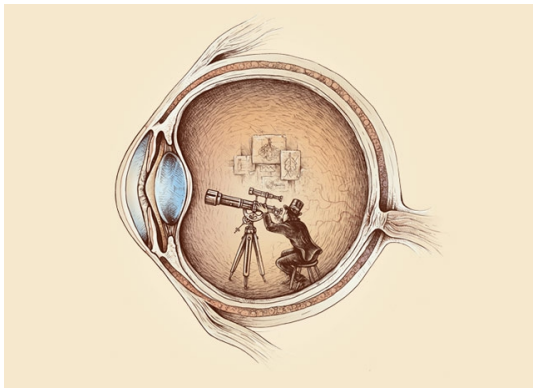


Computer Vision

CS 776 Fall 2018

Machine Learning 1

Prof. Alex Berg

(Slide credits to many folks on individual slides)

Today's lecture

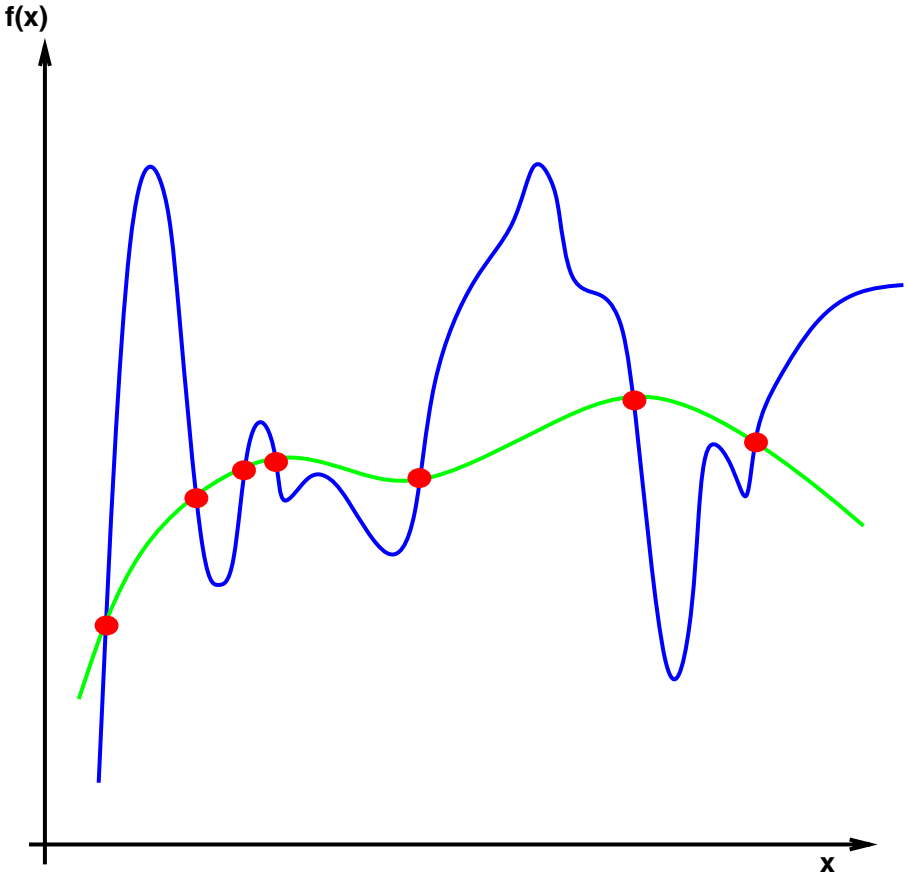
- Machine learning background
- Learning and Loss
- Optimization
- Parameterization

Very general take on Machine Learning

- At a high level, this is like the scientific method:
  - observe some data
  - make some hypotheses (choose model)
  - perform experiments (fit and evaluate model)
  - (profit!)

- Machine learning focuses on the mathematical formulations and computation

Problems with learning a function from data



From Poggio & Smale 2003

(training) data in red

Problem: predict y=f(x)

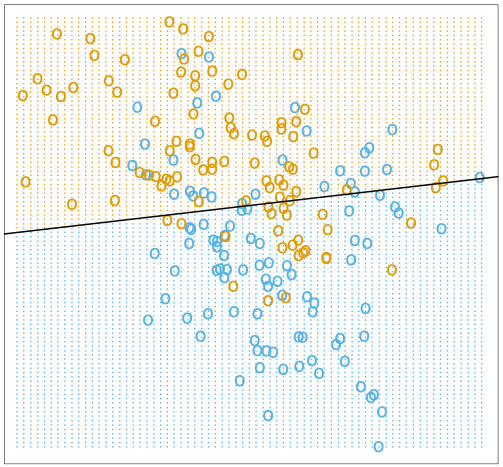
Possible f's shown in blue and green

Need something to limit possibilities!

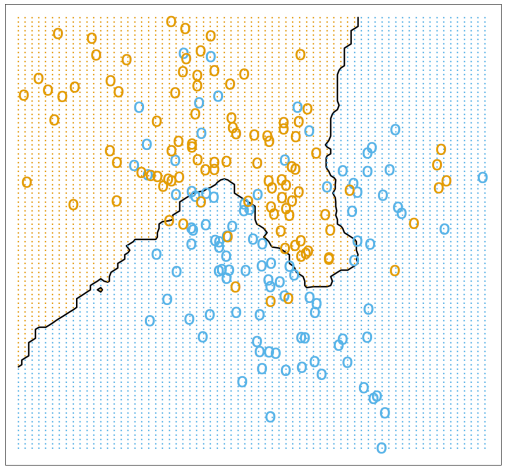
(possibilities are the hypothesis space)

Linear

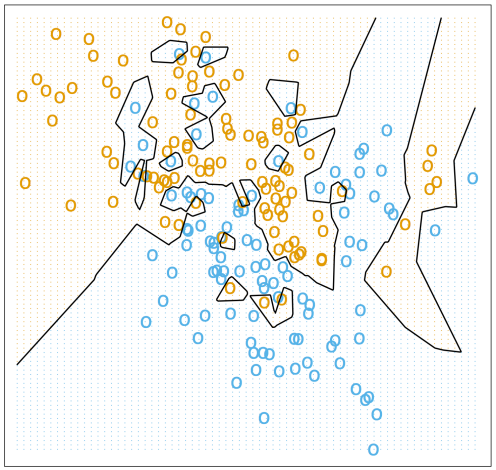
Linear Classifier



15 Nearest Neighbor

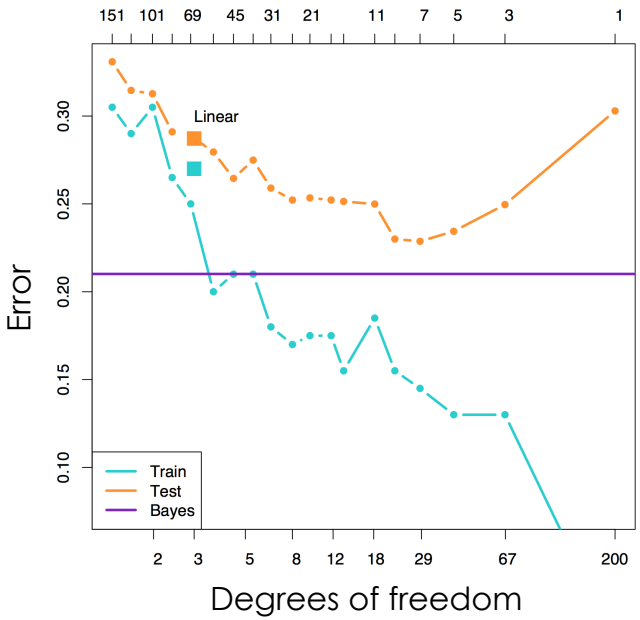


1 Nearest Neighbor



From Hastie, Tibshirani, Friedman Book

K – Number of nearest neighbors



(training) data in wheat/blue

Problem: predict $y=f(x_1,x_2)>0$

Possibilities for $f(x_1,x_2)=0$ shown in black

Learning/fitting is a process…

Estimating the probability that a tossed coin comes up heads…

$X_i \in \{0,1\}$ The i'th coin toss

$$\widehat{\theta}_n(X^n) = \frac{1}{n}\sum_{i=1}^{n} X_i$$

Computation

Estimator based on n tosses

$$\mathsf{G}_{n,\varepsilon} \triangleq \left\{x^n \in \{0,1\}^n : |\widehat{\theta}_n(x^n) - \theta| \leq \varepsilon\right\}$$ Estimate is within epsilon

$$\mathsf{B}_{n,\varepsilon} \triangleq \left\{x^n \in \{0,1\}^n : |\widehat{\theta}_n(x^n) - \theta| > \varepsilon\right\}$$ Estimate is not within epsilon

$$\mathbb{P}_\theta^n(\mathsf{B}_{n,\varepsilon}) \equiv \mathbb{P}_\theta^n\left(|\widehat{\theta}_n(X^n) - \theta| > \varepsilon\right) \leq 2e^{-2n\varepsilon^2}$$

Probability of being bad is inversely proportional to the number of samples…
(the underlying computation is an example of a tail bound)

From Raginsky notes

Bad news for more dimensions

- Estimating a single variable (e.g. bias of a coin) within a couple of percent might take ~100 samples…
- Estimating a function (e.g. the probability of being in class 1) for an *n-dimensional* binary variable requires estimating ~2^n variables. $100x2^n$ can be large.
- *In most cases our game will be finding ways to restrict the possibilities for the function and to focus on the decision boundary (where f(x)=0) instead of f(x) itself.*

Reading

Maxim Raginksy's introduction notes for statistical machine learning:
http://maxim.ece.illinois.edu/teaching/fall14/notes/intro.pdf

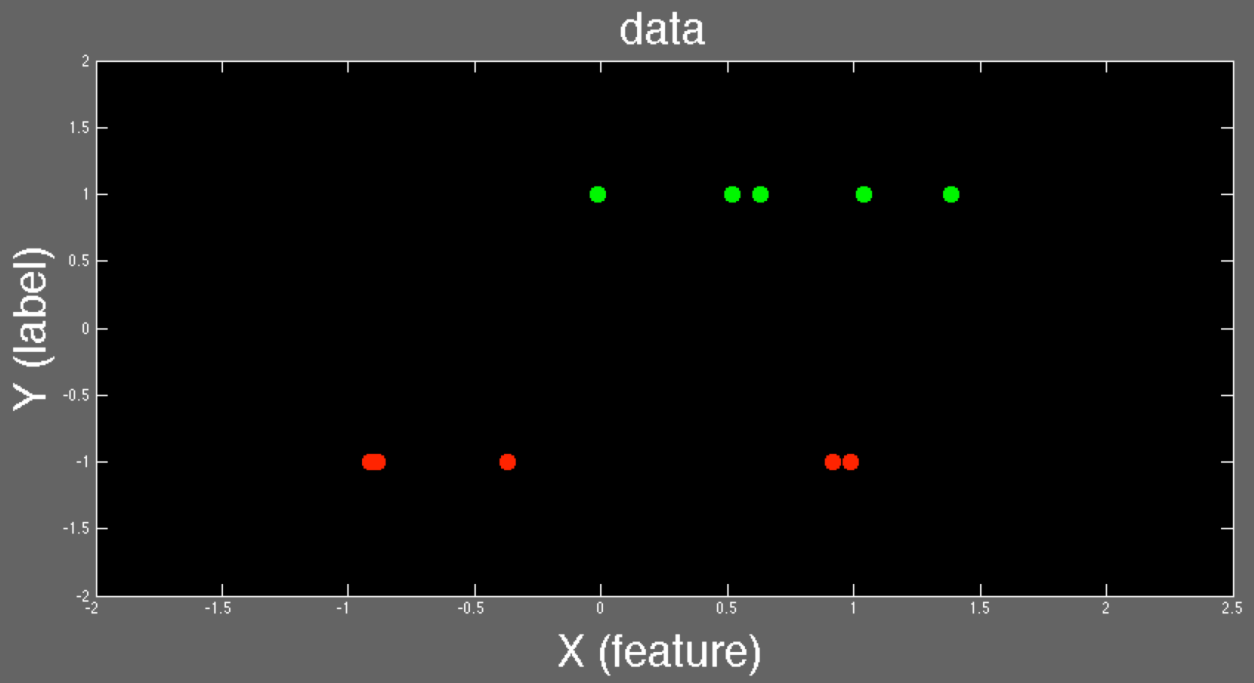
Poggio & Smale "The mathematics of learning: dealing with data", Notices of the American Mathematical Society, vol. 50, no. 5, pp. 537-544, 2003.
http://cbcl.mit.edu/projects/cbcl/publications/ps/notices-ams2003refs.pdf

Hastie, Tibshirani, Friedman Elements of Statistical Learning (the course textbook)
Chapters 1 and 2
http://statweb.stanford.edu/~tibs/ElemStatLearn/

Various Definitions for Machine Learning

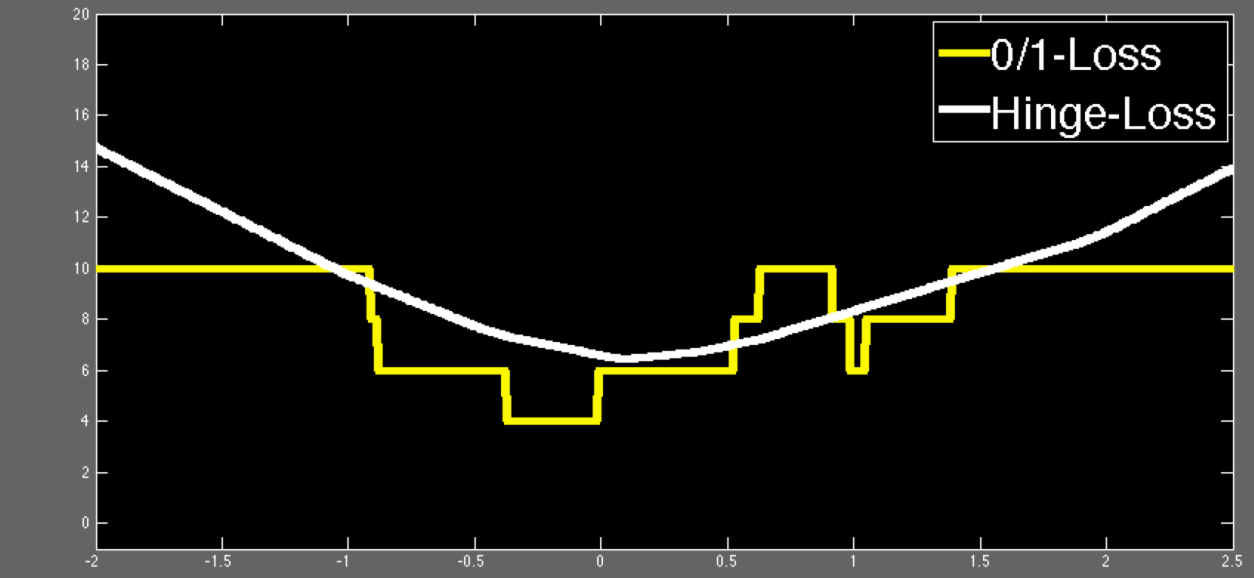
- [Tom Mitchell] Study of algorithms that improve their performance, P, at some task, T, with experience, E. <P,T,E>
- [Wikipedia] a scientific discipline that explores the construction and study of algorithms that can learn from data.
- [Course] Study of how to build/learn functions (programs) to predict something.

  Data, Formulations, Computation



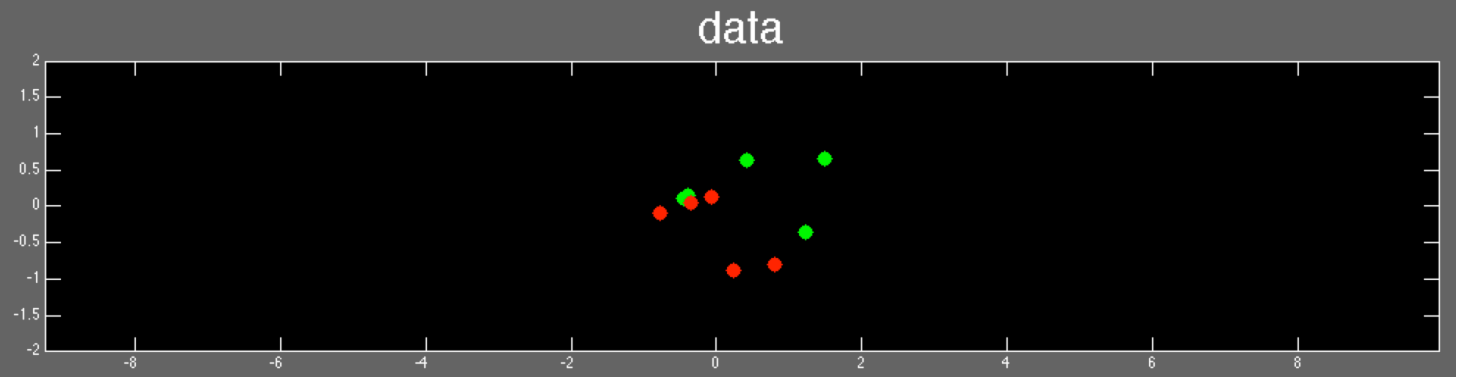
1 D Data with labels -1,1

(indicated by color and y-axis position)

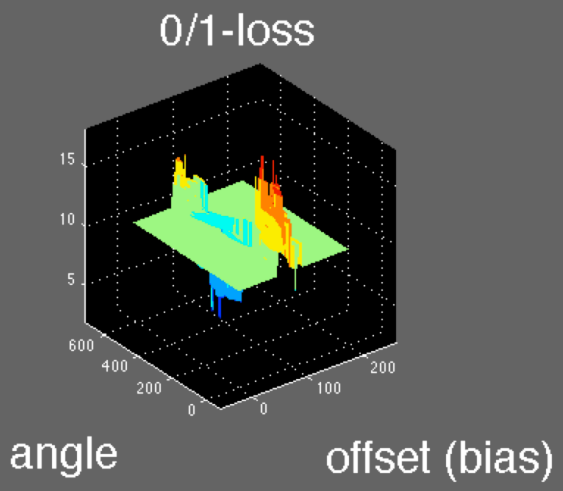


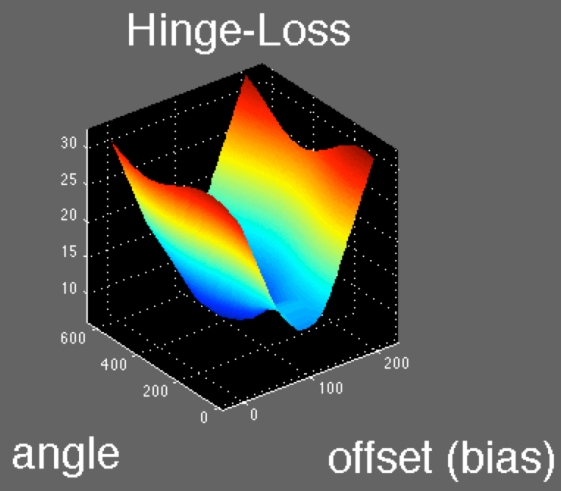
0/1 Loss and Hinge Loss

(y-axis is loss,
x-axis is decision boundary)

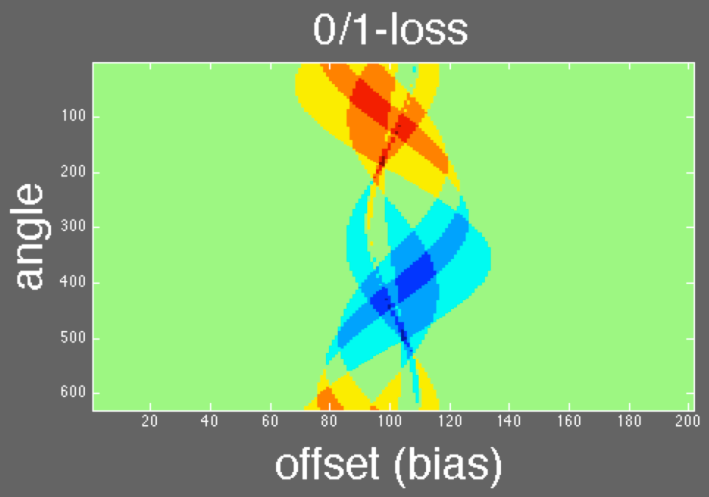


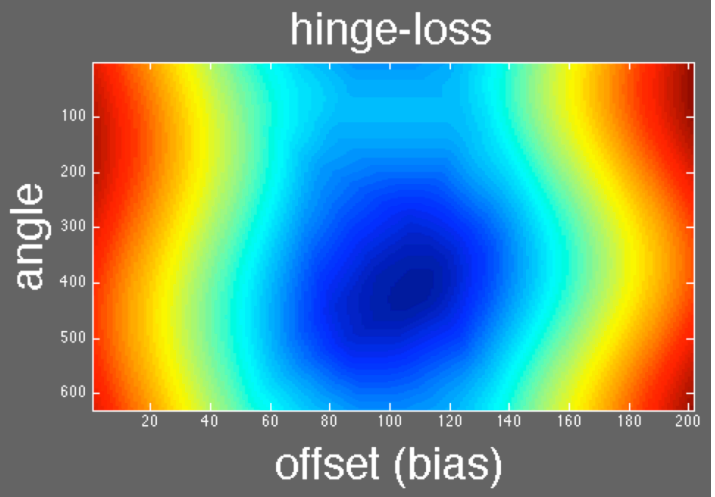
2 D Data with labels -1,1 (indicated by color)





0/1 Loss and Hinge Loss Indicated by color





y-location is angle of decision boundary

x-location is Offset of decision boundary

Quick messages

- 0/1-Loss counts actual errors, no partial credit
- Hinge-Loss is not the same as 0/1 Loss
- Hinge-Loss is greater when farther from correct the decision
- 0/1-Loss has discontinuous first derivative
- 0/1-Loss is non-convex, Hinge-Loss is convex (*)
- * This may depend on parameters/parameterization of the decision functions.

2.4 Statistical Decision Theory

In this section we develop a small amount of theory that provides a framework for developing models such as those discussed informally so far. We first consider the case of a quantitative output, and place ourselves in the world of random variables and probability spaces. Let $X \in \mathbb{R}^p$ denote a real valued random input vector, and $Y \in \mathbb{R}$ a real valued random output variable, with joint distribution $\Pr(X, Y)$. We seek a function $f(X)$ for predicting Y given values of the input X. This theory requires a *loss function* $L(Y, f(X))$ for penalizing errors in prediction, and by far the most common and convenient is *squared error loss*: $L(Y, f(X)) = (Y - f(X))^2$. This leads us to a criterion for choosing f,

$$\begin{aligned} \mathrm{EPE}(f) &= \mathrm{E}(Y - f(X))^2 && (2.9) \\ &= \int [y - f(x)]^2 \Pr(dx, dy), && (2.10) \end{aligned}$$

the expected (squared) prediction error . By conditioning[1] on X, we can write EPE as

$$\mathrm{EPE}(f) = \mathrm{E}_X \mathrm{E}_{Y|X} \left([Y - f(X)]^2 | X \right) \quad (2.11)$$

and we see that it suffices to minimize EPE pointwise:

$$f(x) = \mathrm{argmin}_c \mathrm{E}_{Y|X} \left([Y - c]^2 | X = x \right). \quad (2.12)$$

The solution is

$$f(x) = \mathrm{E}(Y | X = x), \quad (2.13)$$

the conditional expectation, also known as the *regression* function. Thus the best prediction of Y at any point $X = x$ is the conditional mean, when best is measured by average squared error.

The nearest-neighbor methods attempt to directly implement this recipe using the training data. At each point x, we might ask for the average of all those y_is with input $x_i = x$. Since there is typically at most one observation at any point x, we settle for

$$\hat{f}(x) = \mathrm{Ave}(y_i | x_i \in N_k(x)), \quad (2.14)$$

where "Ave" denotes average, and $N_k(x)$ is the neighborhood containing the k points in $\mathcal{T}$ closest to x. Two approximations are happening here:

- expectation is approximated by averaging over sample data;
- conditioning at a point is relaxed to conditioning on some region "close" to the target point.

For large training sample size N, the points in the neighborhood are likely to be close to x, and as k gets large the average will get more stable. In fact, under mild regularity conditions on the joint probability distribution $\Pr(X, Y)$, one can show that as $N, k \to \infty$ such that $k/N \to 0$, $\hat{f}(x) \to \mathrm{E}(Y | X = x)$. In light of this, why look further, since it seems we have a universal approximator? We often do not have very large samples. If the linear or some more structured model is appropriate, then we can usually get a more stable estimate than k-nearest neighbors, although such knowledge has to be learned from the data as well. There are other problems though, sometimes disastrous. In Section 2.5 we see that as the dimension p gets large, so does the metric size of the k-nearest neighborhood. So settling for nearest neighborhood as a surrogate for conditioning will fail us miserably. The convergence above still holds, but the *rate* of convergence decreases as the dimension increases.

How does linear regression fit into this framework? The simplest explanation is that one assumes that the regression function $f(x)$ is approximately linear in its arguments:

$$f(x) \approx x^T \beta. \quad (2.15)$$

This is a model-based approach—we specify a model for the regression function. Plugging this linear model for $f(x)$ into EPE (2.9) and differentiating we can solve for β theoretically:

$$\beta = [\mathrm{E}(X X^T)]^{-1} \mathrm{E}(XY). \quad (2.16)$$

Note we have *not* conditioned on X; rather we have used our knowledge of the functional relationship to *pool* over values of X. The least squares solution (2.6) amounts to replacing the expectation in (2.16) by averages over the training data.

So both k-nearest neighbors and least squares end up approximating conditional expectations by averages. But they differ dramatically in terms of model assumptions:

[1] Conditioning here amounts to factoring the joint density $\Pr(X, Y) = \Pr(Y|X)\Pr(X)$ where $\Pr(Y|X) = \Pr(Y, X)/\Pr(X)$, and splitting up the bivariate integral accordingly.

So both k-nearest neighbors and least squares end up approximating conditional expectations by averages. But they differ dramatically in terms of model assumptions:

- Least squares assumes $f(x)$ is well approximated by a globally linear function.
- k-nearest neighbors assumes $f(x)$ is well approximated by a locally constant function.

Although the latter seems more palatable, we have already seen that we may pay a price for this flexibility.

Many of the more modern techniques described in this book are model based, although far more flexible than the rigid linear model. For example, additive models assume that

$$f(X) = \sum_{j=1}^{p} f_j(X_j). \tag{2.17}$$

This retains the additivity of the linear model, but each coordinate function f_j is arbitrary. It turns out that the optimal estimate for the additive model uses techniques such as k-nearest neighbors to approximate *univariate* conditional expectations *simultaneously* for each of the coordinate functions. Thus the problems of estimating a conditional expectation in high dimensions are swept away in this case by imposing some (often unrealistic) model assumptions, in this case additivity.

Here the data $(x_i, y_i)_{i=1}^m$ is supposed random, so that there is an unknown probability measure ρ on the product space $X \times Y$ from which the data is drawn.

This measure ρ defines a function

$$f_\rho : X \to Y \tag{8}$$

satisfying $f_\rho(x) = \int y d\rho_x$, where ρ_x is the conditional measure on $x \times Y$.

From this construction f_ρ can be said to be the true input-output function reflecting the environment which produces the data. Thus a measurement of the error of f is

$$\int_X (f - f_\rho)^2 d\rho_X \tag{9}$$

where ρ_X is the measure on X induced by ρ (sometimes called the marginal measure).

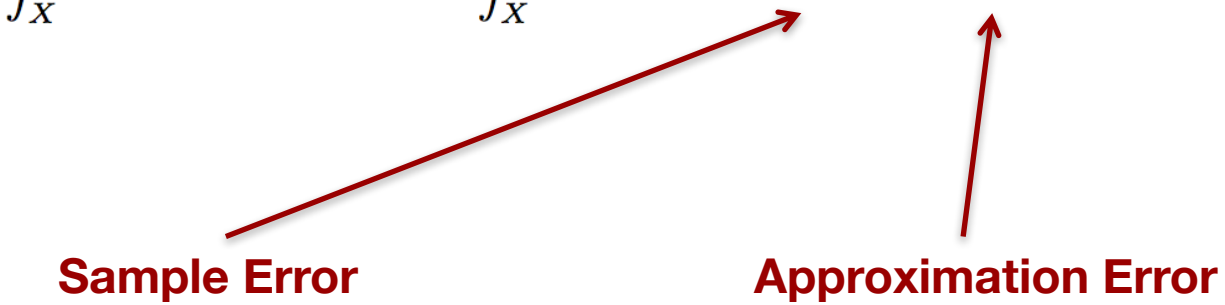
The goal of learning theory might be said to "find" f minimizing this error.

Starting from the data $(x_i, y_i)_{i=1}^m = z$ one may minimize $\frac{1}{m}\sum_{i=1}^m (f(x_i) - y_i)^2$ over $f \in \mathcal{H}$ to obtain a unique hypothesis $f_z : X \to Y$. This f_z is called the empirical optimum and we may focus on the problem of estimating

$$\int_X (f_z - f_\rho)^2 d\rho_X \tag{11}$$

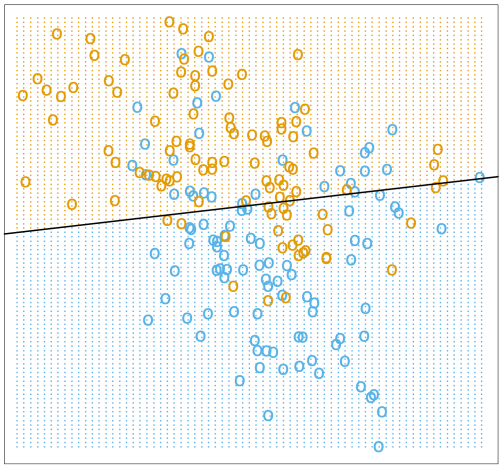
It is useful towards this end to break the problem into steps by defining a "true optimum" $f_{\mathcal{H}}$ relative to $\mathcal{H}$, by taking the minimum over $\mathcal{H}$ of $\int_X (f - f_\rho)^2$. Thus we may exhibit

$$\int_X (f_z - f_\rho)^2 = S(z, \mathcal{H}) + \int_X (f_{\mathcal{H}} - f_\rho)^2 = S(z, \mathcal{H}) + A(\mathcal{H}) \tag{12}$$

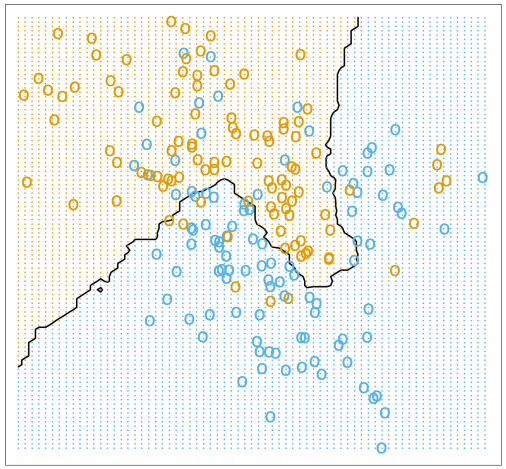


Identify Sample and Approximation Error in this setting…

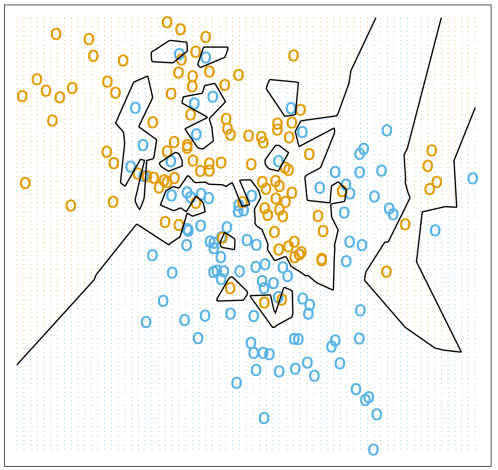
Linear Classifier



15 Nearest Neighbor

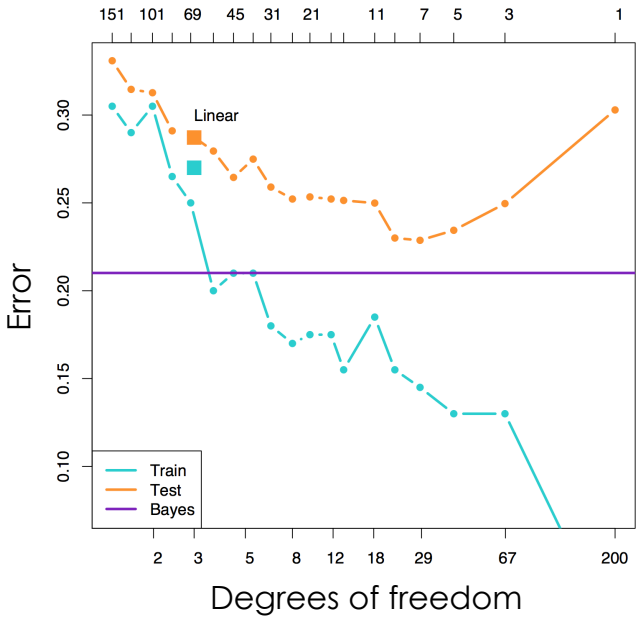


1 Nearest Neighbor



From Hastie, Tibshirani, Friedman Book

K – Number of nearest neighbors



(training) data in wheat/blue

Problem: predict $y=f(x_1,x_2)>0$

Possibilities for $f(x_1,x_2)=0$ shown in black

Related Reading

Messy code for the 2D loss example in Matlab (link).

Kearns and Vazirani Introduction to Computational Learning Theory pages 1-16 (link)

Maxim Raginksy's introduction notes for statistical machine learning:
http://maxim.ece.illinois.edu/teaching/fall14/notes/intro.pdf

Poggio & Smale "The mathematics of learning: dealing with data", Notices of the American Mathematical Society, vol. 50, no. 5, pp. 537-544, 2003.
http://cbcl.mit.edu/projects/cbcl/publications/ps/notices-ams2003refs.pdf

Hastie, Tibshirani, Friedman Elements of Statistical Learning (the course textbook) Chapters 1 and 2
http://statweb.stanford.edu/~tibs/ElemStatLearn/