

# mentary (optional 6.86x notes)

What tools can we use to automatically extract, extrapolate, and explain patterns in data? These notes review the main tools we develop in 6.86x. As binocular vision deepens and disambiguates, the differences between how our lectures and how these notes organize concepts may also enrich understanding.

You do not need to read these notes at all to get an A in this course; conversely, you may not cite these notes when justifying work in homework or exams.

## A. Prologue

A WHIMSICAL STORY ABOUT AUTOMATION — We automate when we re-cast the previous generation's *methods* as mere *data* — data to produce and manipulate.

We once sought safety from floods. Long ago, if we lived near safe riverbanks it was by luck; we then found we could make unsafe rivers safe by erecting floodwalls. This was dangerous work, so we built brick laying robots to do the hard part. The machines needed as input some flood-risk estimates: the location and mud-softnesses of the river banks to work on.

So we sought flood-risk estimates. At first, if our city's flood-risk estimates were correct it was by luck; we then found we could translate topographic maps to flood-risk estimates by applying geology ideas. This was tedious work, so we built computers to do the hard part. The computers needed as input a risk-estimation *algorithm*: a recipe of the geology rules to calculate through.

We thus sought risk-estimation algorithms. At first, if our research center's risk-estimation algorithms were robust to new landscapes it was by luck; we then found we could calibrate our algorithms to historical flood data from many cities. To find the better of two candidate algorithms, we had to aggregate all their many small success and errors. This was subtle work, so we developed statistical theories to do the hard part. The theories needed as input a *machine learning model*: a computationally tractable class of candidate algorithms.

We thus sought machine learning models. At first, if our machine learning models were highly predictive it was by luck; we then found general principles to guide model design and selection from domain knowledge.

It is these principles that we study in 6.86x.

Looking back, these are principles that produce machine learning models that produce risk-estimation algorithms that produce flood-risk estimates that guide the building of floodwalls.

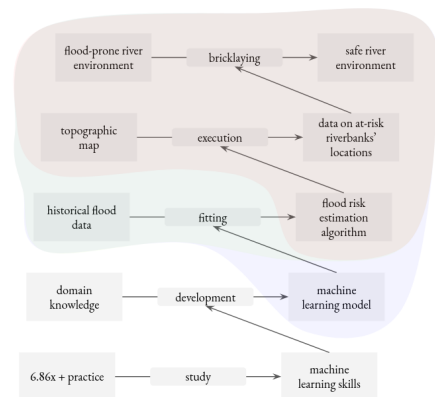
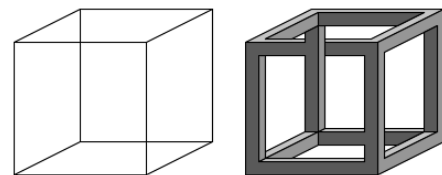


Figure 1: Machine learning continues our human tradition toward the systematic and automatic. What for one generation is a luckily discovered idea or recipe (bottom tip of red), the next generation refines by careful human thought (green). What to one generation is a task (green) requiring human thought is to a future generation merely a routine parameterized by some newly discovered recipe (bottom tip of blue). And the cycle repeats.

**KINDS OF LEARNING** — How do we communicate patterns of desired behavior? We can teach:

**by instruction:** “to tell whether a mushroom is poisonous, first look at its gills...”

**by example:** “here are six poisonous fungi; here, six safe ones. see a pattern?”

**by reinforcement:** “eat foraged mushrooms for a month; learn from getting sick.”

Machine learning is the art of programming computers to learn from such sources.

We’ll focus on the most important case: learning from examples. Given a list of  $N$  examples of properly answered prompts, we seek a pattern: a map from prompts in  $\mathcal{X}$  to answers in  $\mathcal{Y}$ . So a program that learns from examples has type

$$\mathcal{L} : (\mathcal{X} \times \mathcal{Y})^N \rightarrow (\mathcal{X} \rightarrow \mathcal{Y})$$

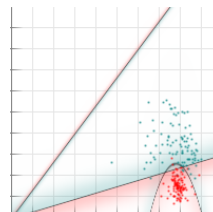
**LEARNING ERROR** — Draw examples  $\mathcal{S} : (\mathcal{X} \times \mathcal{Y})^N$  from a distribution  $\mathcal{D}$  on  $\mathcal{X} \times \mathcal{Y}$ . A pattern  $f : \mathcal{X} \rightarrow \mathcal{Y}$  has **training error**  $\text{trn}_{\mathcal{S}}(f) = \mathbb{P}_{(x,y) \sim \mathcal{S}}[f(x) \neq y]$  and **testing error**  $\text{tst}(f) = \mathbb{P}_{(x,y) \sim \mathcal{D}}[f(x) \neq y]$ . We want low  $\text{tst}(\mathcal{L}(\mathcal{S}))$ . We often set  $\mathcal{L}$  to minimize  $\text{trn}_{\mathcal{S}}$  in a candidates set  $\mathcal{H} \subseteq (\mathcal{X} \rightarrow \mathcal{Y})$ . Then  $\text{tst}$  decomposes into the failures of  $\text{trn}_{\mathcal{S}}$  to estimate  $\text{tst}$ ,  $\mathcal{L}$  to minimize  $\text{trn}_{\mathcal{S}}$ , and  $\mathcal{H}$  to contain “the” truth:

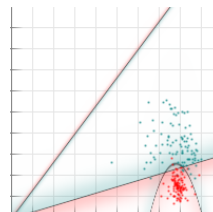
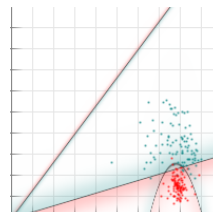
$$\begin{aligned} \text{tst}(\mathcal{L}(\mathcal{S})) &= \text{tst}(\mathcal{L}(\mathcal{S})) && - \text{trn}_{\mathcal{S}}(\mathcal{L}(\mathcal{S})) && \} \text{generalization error} \\ &+ \text{trn}_{\mathcal{S}}(\mathcal{L}(\mathcal{S})) && - \inf_{\mathcal{H}}(\text{trn}_{\mathcal{S}}(f)) && \} \text{optimization error} \\ &+ \inf_{\mathcal{H}}(\text{trn}_{\mathcal{S}}(f)) && && \} \text{approximation error} \end{aligned}$$

These terms are in tension. For example, as  $\mathcal{H}$  grows, the approx. error may decrease while the gen. error may increase — this is the “**bias-variance tradeoff**”.

**TINY EXAMPLE** —  $\mathcal{X} = \{\text{grayscale } 28 \times 28\text{-pixel images}\}$ ;  $\mathcal{Y} = \{0, 1\}$ .  $\mathcal{D}$ ’s samples are photos  $x$  of handwriting of the digit  $y$ .  $\mathcal{H}$  contains all “linear hypotheses”  $f_{a,b}$  defined by:

$$f_{a,b}(x) = 0 \text{ if } a \cdot \text{width}(x) + b \cdot \text{darkness}(x) < 0 \text{ else } 1$$

Our program  $\mathcal{L}$  loops over all integer pairs  $(a, b)$  in  $[-99, +99]$  to minimize  $\text{trn}_{\mathcal{S}}$ , giving  $(a, b) = (-20, 6)$  with  $\text{trn}_{\mathcal{S}} \approx 5\%$ . We got optimization error  $\approx 0$  (albeit by *unscalable brute-force*). So approximation error  $\approx \text{trn}_{\mathcal{S}} \approx 5\%$ . Indeed, our straight lines are *too simple*: width and darkness lose useful information and the “true” boundary looks curved (see ’s curve). The testing error  $\text{tst} \approx 8\%$  exceeds  $\text{trn}_{\mathcal{S}}$ : we suffer *generalization error*. In 6.86x we’ll address all three italicized issues.

**Exercise:** how do ’s two straight lines and ’s two marked points correspond?  
**Exercise:** visualize  $\text{trn}_{\mathcal{S}}(f_{a,b})$  in the  $(a, b)$  plane for  $N = 1$  training example.  
**Exercise:** why is generalization error usually positive?

**TODO:** five labeled pictures of mushrooms and poison levels (say, “safe”, “stomach upsetting”, “very dangerous”). A sixth, un-labeled picture with a question mark for its label. Make explicit that we view mushrooms as prompts/inputs and poison levels as answers/outputs. And talk about patterns/hypotheses.

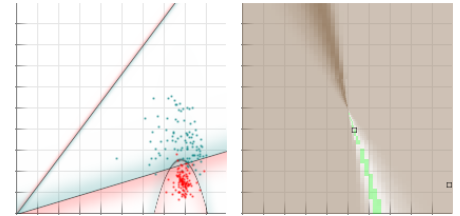
← We’ll see in §E that learning by example is key to the other modes of learning.

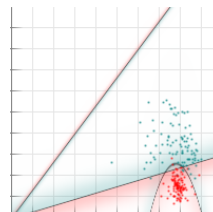
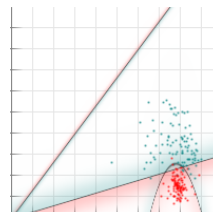
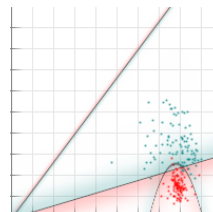
← Actually, we’ll soon account for uncertainty by letting patterns map to *probability distributions over answers* (§B.1); then a program that learns from examples has type

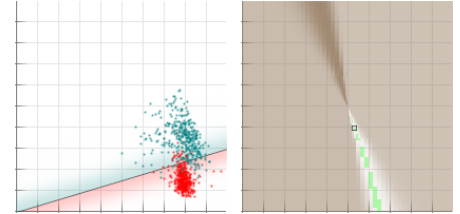
$$\mathcal{L} : (\mathcal{X} \times \mathcal{Y})^N \rightarrow (\mathcal{X} \rightarrow \text{DistributionsOn}(\mathcal{Y}))$$

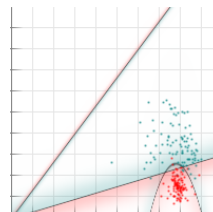
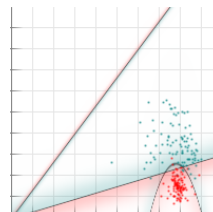
Sometimes, the prompt is always the same — say, “produce a beautiful melody” — and we seek (e.g.) to generate many answers. So-called **unsupervised learning** thusly focuses on output structure. **Supervised learning** focuses on the input-output relation.

Six example  $(x, y)$ s. An  $x$ ’s *darkness* is its average pixel darkness; its *width* is the stddev column index weighted by column darkness. We normalize both to have max value 1.0:



3 hypotheses classify  $x$ s in the darkness-width plane (). The  $(a, b)$  plane () parameterizes linear hypotheses; darker means larger  $\text{trn}_{\mathcal{S}}$ .  $(a, b) = (-20, 6)$  minimizes  $\text{trn}_{\mathcal{S}}$  to 0.05 but on fresh test samples () suffers  $\text{tst} = 0.08$ .



’s axes range  $[0, 0.5]$ . ’s axes range  $[-99, +99]$ ; green indicates error  $< 0.1$ .

## B. Linear models

### 0. linear approximations

**FEATURIZATION** — As in the prologue, we represent our input  $x$  as a fixed-length list of numbers so that we can treat  $x$  with math. For example, in the prologue we represented each photograph by 2 numbers: width and darkness. We could instead have represented each photograph by 784 numbers, one for the brightness at each of the  $28 \cdot 28 = 784$  many pixels. Or by 10 numbers, each measuring the overlap of  $x$ 's ink with a “standard” photo of the digits 0 through 9.

When we choose how to represent  $x$  by a list of numbers, we're choosing a **featurization**. We call each number a “feature”. For example, width and darkness are two features.

**Exercise:** Beyond width and darkness, what features do you think might help us to separate digits 0 from 1 by a line through the origin? How about 3 from 8?

There are lots of interesting featurizations, each making different patterns easier to learn. So we judge a featurization with respect to the kinds of patterns we use it to learn. Learning usually happens more accurately, robustly, and interpretably when our featurization is abstract (no irrelevant details) but complete (all relevant details), compressed (hard to predict one feature from the others) but accessible (easy to compute interesting properties).

Caution: a feature  $A(x)$  that is statistically independent from  $y$  may still be relevant for predicting  $y$ . For example, if  $A, B$  are two features, it is possible that  $A(x), y$  are independent and that  $B(x), y$  are independent and yet  $(A(x), B(y)), y$  are *dependent*!

**TODO:** example featurization (e.g. MNIST again?)

**SOME FUN GEOMETRY** — Now say we've decided on a **featurization** of our input data  $x$ .

$$f_{a,b}(x) = 0 \text{ if } a \cdot \text{width}(x) + b \cdot \text{darkness}(x) < 0 \text{ else } 1$$

Illustrate ‘averaging’ of good features vs ‘correction’ of one feature by another (how much a feature correlates with error)

**LINEAR ALGEBRA** — Linear algebra is the part of geometry that focuses on when a point is the origin, when a ‘line’ is a straight, and when two straight lines are parallel. Linear algebra thus helps us deal with the preceding pictures mathematically. The concept of ‘straight lines’ gives a simple, flexible model for extrapolation from known points to unknown points. That is intuitively why linear algebra will be crucial at every stage of 6.86x.

The elements of linear algebra are **column vectors** and **row vectors**. **FILL IN** Though we represent the two similarly in a computer's memory, they have different geometric meanings. We save much anguish by remembering the difference. **FILL IN**

**FILL IN LINEAR DECISION BOUNDARY!** (remark on featurization and argmax nonlinearities)

We may **evaluate** a row vector on a column vector. **FILL IN** A **dot product** is

*He had bought a large map representing the sea,  
Without the least vestige of land:  
And the crew were much pleased when they found it to be  
A map they could all understand.  
— charles dodgson*

Example. Consider the uniform distribution on the four corners of a tetrahedron embedded within the corners of a cube **TODO: graphic**. The three spatial coordinates give three bit-valued random variables. Any two of these variables are independent. But the three together are dependent. **TODO: also do a decision boundary (simpsons style) graph illustrating this phenomenon**

← It is important to **thoroughly understand these basics** of linear algebra. Please see §G.2 for further discussion of these basics.

a way of translating between row and column vectors. **FILL IN: DISCUSS GENERALIZATION; (DISCUSS ANGLE, TOO)**

**FILL IN COMPUTATION AND BASES**

**VISUAL ILLUSTRATION OF HOW CHOICE OF DOT PRODUCT MATTERS**

**RICHER OUTPUTS** — We’ve learned how to construct a set  $\mathcal{H}$  of candidate patterns

$$f_{\vec{w}}(\vec{x}) = \text{threshold}(\vec{w} \cdot \vec{x})$$

that map (a featurization of) a prompt  $\vec{x}$  to a binary answer  $y = 0$  or  $y = 1$ .

What if we’re interested in predicting a richer kind of  $y$ ? For example, maybe there are  $k$  many possible values for  $y$  instead of just 2. Or maybe there are infinitely many possible values — say, if  $y$  is a real number. Or maybe we want the added nuance of predicting probabilities, so that  $f$  might output “20% chance of label  $y = 0$  and 80% chance of label  $y = 1$ ” instead of just “ $y = 1$ ”.

I’ll write formulas and then explain.

$$f_{\vec{w}_i: 0 \leq i < k}(\vec{x}) = \text{argmax}_i(\vec{w}_i \cdot \vec{x})$$

$$f_{\vec{w}}(\vec{x}) = \vec{w} \cdot \vec{x}$$

**TODO: add multi-output regression?**

$$f_{\vec{w}_i: 0 \leq i < k}(\vec{x}) = \text{normalize}(\exp(\vec{w}_i \cdot \vec{x}) : 0 \leq i < k)$$

**TODO: interpret**

**TODO: discuss measures of goodness!**

## 1. iterative optimization

You can **read the passages on “perceptrons” and on “logistic models” in either order**; it depends on your personality. Logistic models and perceptrons are two sides of the same coin. The former is continuous; the latter, discrete. I prefer to read “logistic models” first.

*Hey Jude, don't make it bad  
Take a sad song and make it better  
Remember to let her under your skin  
Then you'll begin to make it  
Better, better, better, better, better, better, ...  
— paul mccartney, john lennon*

**(STOCHASTIC) GRADIENT DESCENT** — We have a collection  $\mathcal{H}$  of candidate patterns together with a function  $1 - \text{trn}_S$  that tells us how good a candidate is. In §A we found a best candidate by brute-force search over all of  $\mathcal{H}$ ; this doesn’t scale to our linear models, since now  $\mathcal{H}$  is intractably large. So: *what’s a faster algorithm to find (or approximate) a best candidate?*

A common idea is to start arbitrarily with some  $h_0 \in \mathcal{H}$  and repeatedly improve to get  $h_1, h_2, \dots$ . Two questions are: *how do we select  $h_{t+1}$  in terms of  $h_t$ ?* And *how do we know when to stop?* We’ll discuss termination conditions later — for now, let’s agree to stop at  $h_{1000}$ .

As for selecting a next candidate, we’d like to use more detailed information on  $h_t$ ’s inadequacies to inform our proposal  $h_{t+1}$ . Intuitively, if  $h_t$  misclassifies a particular  $(x_n, y_n) \in \mathcal{S}$ , then we’d like  $h_{t+1}$  to be like  $h_t$  but nudged a bit in the direction of accurately classifying  $(x_n, y_n)$ .

**FILL IN FOR PROBABILITIES (LOGISTIC) MODEL**

← We view  $1 - \text{trn}_S$  as an estimate of our actual notion of “good”:  $1 - \text{tst}$ .

This is the idea of **gradient descent**. **MOTIVATE AND GIVE PSEUDOCODE FOR STOCHASTIC GD**

LOGISTIC MODELS —

PERCEPTRONS —

HINGE LOSS AND SVMs —

## 2. priors and generalization

ON OVERFITTING —

LOG PRIORS AND BAYES —

$\ell^p$  REGULARIZATION; SPARSITY —

ESTIMATING GENERALIZATION —

## 3. model selection

TAKING STOCK SO FAR —

GRID/RANDOM SEARCH —

SELECTING PRIOR STRENGTH —

OVERFITTING ON A VALIDATION SET —

## 4. generalization bounds

DOT PRODUCTS AND GENERALIZATION —

PERCEPTRON BOUND —

DIMENSION BOUND —

MARGIN BOUND —

## 5. ideas in optimization

LOCAL MINIMA —

IMPLICIT REGULARIZATION —

LEARNING RATE SCHEDULE —

LEARNING RATES AS DOT PRODUCTS —

## 6. kernels enrich approximations

FEATURES AS PRE-PROCESSING —

ABSTRACTING TO DOT PRODUCTS —

KERNELIZED PERCEPTRON AND SVM —

KERNELIZED LOGISTIC REGRESSION —

*I believe that either Jupiter has life or it doesn't. But I  
neither believe that it does, nor do I believe that it doesn't.*  
— raymond smullyan

*All human beings have three lives: public, private, and  
secret.*  
— gabriel garcia marquez

*A foreign philosopher rides a train in Scotland. Looking  
out the window, they see a black sheep; they exclaim:  
“wow! at least one side of one sheep is black in Scotland!”*  
— unknown

*premature optimization is the root of all evil*  
— donald knuth

*... animals are divided into (a) those belonging to the  
emperor; (b) embalmed ones; (c) trained ones; (d) suckling  
pigs; (e) mermaids; (f) fabled ones; (g) stray dogs; (h) those  
included in this classification; (i) those that tremble as if  
they were mad; (j) innumerable ones; (k) those drawn with  
a very fine camel hair brush; (l) et cetera; (m) those that  
have just broken the vase; and (n) those that from afar look  
like flies.*  
— jorge luis borges

## C. Nonlinearities

### 0. fixed featurization

*Doing ensembles and shows is one thing, but being able to front a feature is totally different. ... there's something about ... a feature that's unique.*  
— michael b. jordan

### 1. learned featurizations

### 2. differentiation

### 3. architecture and symmetry

*About to speak at [conference]. Spilled Coke on left leg of jeans, so poured some water on right leg so looks like the denim fade.*  
— tony hsieh

### 4. feature hierarchies

### 5. stochastic gradient descent

*The key to success is failure.*  
— michael j. jordan

### 6. loss landscape shape

*The virtue of maps, they show what can be done with limited space, they foresee that everything can happen therein.*  
— josé saramago

## D. Structured inference

### 0. graphical generative models

#### 1. inferring conditional marginals

#### 2. learning the parameters

#### 3. hierarchy and mixtures

#### 4. hierarchy and transfer

#### 5. variational and sampling methods

#### 6. amortized inference

*... [to treat] complicated systems in simple ways[,] probability ... implements two principles[:] [the approximation of parts as independent] and [the abstraction of aspects as their averages].*

— michael i. jordan

*Reminding myself that I have a tailbone keeps me in check.*  
— tig notaro

## E. Reductions to supervision

0. to build a tool, use it

1. distributions as maps

2. self-supervised: downstream tasks

3. self-supervised: autoregression

4. reinforcement: exploration-exploitation

5. reinforcement: states and q-values

*We are what we pretend to be, so we must be careful about  
what we pretend to be.*  
— kurt vonnegut

*Possession of anything new or expensive only reflected a  
person's lack of theology and geometry; it could even cast  
doubts upon one's soul.*  
— john toole

*Be a pattern to others and then all will go well.*  
— marcus cicero

*When we learn to speak, we learn to translate.*  
— octavio paz

*We're all curious about what might hurt us.*  
— frederico garcia lorca

*There's a good reason why nobody studies history: it just  
teaches you too much.*  
— noam chomsky

ON-POLICY LEARNING —

OFF-POLICY LEARNING —

DEEP CURRICULA: RE-PLAY —

DEEP CURRICULA: SELF PLAY —

THE DEADLY TRIAD —

6. beyond the i.i.d. hypothesis

OUT-OF-DISTRIBUTION TESTS —

DEPENDENT SAMPLES —

CAUSALITY —

*When I was a boy of 14, my father was ignorant; I could  
hardly stand to have the old man around. But when I got  
to be 21, I was astonished at how much the old man had  
learned in seven years.*  
— mark twain

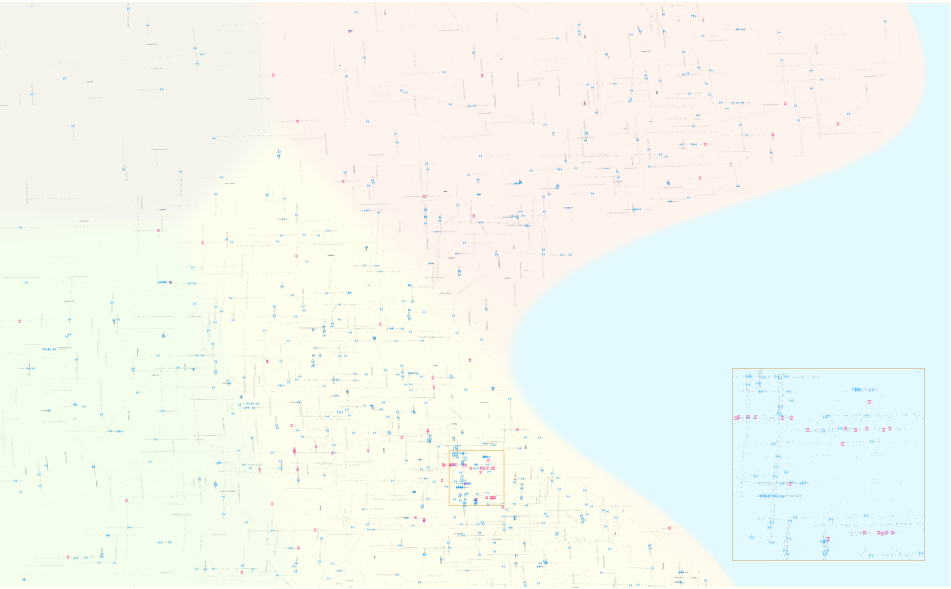


F. Three brief example projects

Now comes a hurried tour of the kinds of analysis these notes discuss. I recommend that you skim these three example projects without trying to understand each step.

1. flooding plains

VISUALIZATION —



*I shall have to accept the fact, I'm afraid, that [their] mind is a very thin soil, laid an inch or two deep upon very barren rock.*  
— virginia woolf

Figure 2: In light yellow and blue are land and sea: 6.4km × 10.4km around the city of Zembla. Gray dots: untested faucets. Red bowties: faucets tested positive. Blue I-beams: faucets tested negative. The small yellow box surrounds a neighborhood that enjoyed more thorough testing; the large yellow box in the lower right magnifies that extra-tested region. TODO: number of gray dots, among labels: training set vs hidden

FITTING —

MODEL SELECTION —

PREDICTION —

OVERFITTING —

1. ancient tablets

VISUALIZATION —

TRANSCRIPTION —

CLEANING AND IMPUTATION —

MODELING AND FITTING —

GENERATION —

A WHIMSICAL STORY ABOUT MODELING —

2. pairing flavors

VISUALIZATION —

MODELING AND FITTING —

*I am so clever that sometimes I don't understand a single word of what I am saying.*  
— oscar wilde

*Bread that must be sliced with an ax is bread that is too nourishing.*  
— fran lebowitz

REGULARIZATION —

EXPLANATION —

DOMAIN ADAPTATION —

A WHIMSICAL STORY ABOUT INDUCTION —

## Appendices

Here are “refreshers” on programming and on math, especially on the math of high dimensions and of bayes’ law. **They can help jog your memory of the math and programming** you hopefully have encountered prior to taking 6.86x. They also help establish our naming conventions.

### python programming refresher

*If I have not seen as far as others, it is because giants were standing on my shoulders.*  
— hal abelson

SETUP —

STATE AND CONTROL FLOW —

INPUT/OUTPUT —

NUMPY —

### probability refresher

We’ve tried to use

sans serif for the names of random variables,

*italics* for the values they may take, and

*CURLY CAPS* for sets of such values.

For example, we write  $p_{y|h}(y|h)$  for the probability that the random variable  $y$  takes the value  $y$  conditioned on the event that the random variable  $h$  takes the value  $h$ . Likewise, our notation  $p_{\hat{h}|h}(h|h)$  indicates the probability that the random variables  $\hat{h}$  and  $h$  agree in value given that  $h$  takes a value  $h \in \mathcal{H}$ .

```
def get_random_number():
    return 4 # chosen by a fair dice roll
            # guaranteed to be random
```

— randall munroe, translated by sam to Python

### linear algebra refresher

*Stand firm in your refusal to remain conscious during algebra. In real life, I assure you, there is no such thing as algebra.*  
— fran lebowitz

### calculus refresher

### high dimensional geometry

VISUALIZING HIGHER DIMENSIONS —

CONCENTRATION OF MEASURE —

**Lemma (Chernoff).** *The fraction of heads among  $N$  i.i.d. flips of a biased coin exceeds its mean  $p$  by  $g$  with chance at most  $\exp(-Ng^2)$ , for  $0 \leq p < p + g \leq 1$ .*

*Can I just say Chris for one moment that I have a new theory about the brontosaurus. ... This theory goes as follows and begins now. All brontosaurus are thin at one end, much, much thicker in the middle and then thin again at the far end. That is my theory, it is mine, and belongs to me and I own it, and what it is too.*  
— john cleese

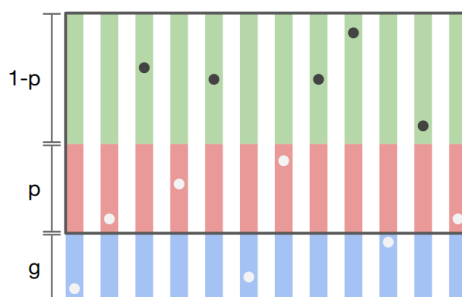


Figure 3: We sample points uniformly at random on  $N$  sticks, each with three parts: **green** with length  $1 - p$ , **red** with length  $p$ , and **blue** with length  $g$ . We call non-blue points **boxed** and non-green points **hollow**.

*Proof.* Let our coin flips arise from sampling points on sticks (Figure 3), where green means tails, red means heads, and we condition on the event that blues do not occur. To show that less than  $(p + g)N = p'N$  flips are heads is to show — given that all points are **boxed** — that less than  $p'N$  points are red. For any  $M$ :

$$\begin{aligned} & \mathbb{P}[M \text{ are red} \mid \text{all are boxed}] \\ &= \frac{\mathbb{P}[\text{all hollows are red} \mid M \text{ hollow}] \cdot \mathbb{P}[M \text{ are hollow}]}{\mathbb{P}[\text{all are boxed}]} \\ &= (1 - g/p')^M \cdot (1 + g)^N \cdot \mathbb{P}[M \text{ are hollow}] \end{aligned}$$

We sum over  $M \geq p'N$ , bound  $\mathbb{P}[\dots p'N \dots] \leq 1$ , then invoke  $(x \mapsto x^{p'})$ 's concavity and  $\exp$ 's convexity:

$$\begin{aligned} & \mathbb{P}[\text{at least } p'N \text{ are red} \mid \text{all are boxed}] \\ &\leq (1 - g/p')^{p'N} \cdot (1 + g)^N \cdot \mathbb{P}[\text{at least } p'N \text{ are hollow}] \\ &\leq (1 - g)^N \cdot (1 + g)^N = (1 - g^2)^N \leq \exp(-Ng^2) \quad \square \end{aligned}$$

The Chernoff bound gives us the control over tails we would expect from the Central Limit Theorem, but for finite instead of asymptotically large  $N$ . In particular, when we learn from much but finite data, the training error will **concentrate** near the testing error.

Indeed, for any  $f \in \mathcal{H}$ ,  $\text{trn}_S(f)$  is the average of  $N$  independent Bernoullis of mean  $\text{tst}(f)$ . So for finite  $\mathcal{H}$ , the gap is probably small:

$$\begin{aligned} & \mathbb{P}_{S \sim \mathcal{D}^N}[\text{gap}_S(\mathcal{L}) \geq g] \\ &\leq \sum_{f \in \mathcal{H}} \mathbb{P}_{S \sim \mathcal{D}^N}[\text{tst}(f) \geq \text{trn}_S(f) + g] \\ &\leq |\mathcal{H}| \cdot \exp(-Ng^2) \end{aligned}$$

For example, if  $\mathcal{H}$  is parameterized by  $P$  numbers, each represented on a computer by 32 bits, then  $|\mathcal{H}| \leq 2^{32P}$  and, with probability  $1 - \delta$ , the gap is less than

$$\sqrt{(\log(1/\delta) + 32P)/N}$$

This bound's sensitivity to the description length  $32P$  may seem artificial. Indeed, the various  $\mathcal{H}$  used in practice — e.g. linear models or neural networks — depend smoothly on their parameters, so the parameters' least significant bits barely affect the classifier. In other words,  $\mathcal{H}$ 's cardinality is not an apt measure of its size. The VC-dimension measures  $\mathcal{H}$  more subtly.

QUADRATIC FORMS —

COVARIANCE, CORRELATION, LEAST SQUARES REGRESSION —

bayesian inference

*So little of what could happen does happen.*  
— salvador dali

CONCEPTUAL FRAMEWORK — We're confronted with an observation or dataset  $o$  that comes from some unknown underlying pattern  $h$ . We know how each possible value  $h$  for  $h$  induces a distribution on  $o$  and we have a prior sense of which  $h$ s are probable. Bayes' law helps us update this sense to account for the dataset by relating two functions of  $h$ :

$$\underbrace{p_{h|o}(h|o)}_{\text{posterior}} \propto \underbrace{p_{o|h}(o|h)}_{\text{likelihood}} \cdot \underbrace{p_h(h)}_{\text{prior}}$$

Bayes' law underlies most of our analyses throughout these notes.

Formally, we posit a set  $\mathcal{H}$  of *hypotheses*, a set  $\mathcal{O}$  of possible *observations*, and a set  $\mathcal{A}$  of permitted *actions*. We assume as given a joint probability measure  $p_{o,h}$  on  $\mathcal{O} \times \mathcal{H}$  and a *cost function*  $c : \mathcal{A} \times \mathcal{H} \rightarrow \mathbb{R}$ . That cost function says how much it hurts to take the action  $a \in \mathcal{A}$  when the truth is  $h \in \mathcal{H}$ . Our primary aim is to construct a map  $\pi : \mathcal{O} \rightarrow \mathcal{A}$  that makes the expected cost  $\mathbb{E}_{h,o} c(\pi(a); h)$  small.

Below are three examples. In each case, we're designing a robotic vacuum cleaner:  $\mathcal{H}$  contains possible floor plans;  $\mathcal{O}$ , possible readings from the robot's sensors. The examples differ in how they define and interpret  $\mathcal{A}$  and  $c$ .

A.  $\mathcal{A}$  consists of probability distributions over  $\mathcal{H}$ . We regard  $\pi(o)$  as giving a posterior distribution on  $\mathcal{H}$  upon observation  $o$ . Our cost  $c(a; h)$  measures the surprise of someone who believes  $a$  upon learning that  $h$  is true. Such *inference problems*, being in a precise sense universal, pose huge computational difficulties; we thus often collapse distributions to points, giving rise to the distinctive challenge of balancing estimation error with structural error.

B.  $\mathcal{A}$  consists of latitude-longitude pairs, interpreted as a guessed location of the robot's charging station. The cost  $c(a; h)$  measures how distant our guess is from the truth. Such *estimation problems* abound in science and engineering; they pose the distinctive challenge of balancing sensitivity-to-misleading-outliers against sensitivity-to-informative-datapoints.

C.  $\mathcal{A}$  consists of instructions we may send to the motors, instructions that induce motion through our partially-known room. The cost  $c(a; h)$  incentivizes motion into dusty spaces and penalizes bumping into walls. We often compose such *decision problems* sequentially; this gives rise to the distinctive challenge of balancing exploration with exploitation.

← Like Newton's  $F = ma$ , Bayes is by itself inert: to make predictions we'd have to specify our situation's forces or likelihoods. Continuing the metaphor, we will rarely solve our equations exactly; we'll instead make approximations good enough to build bridges and swingsets. Still, no one denies that  $F = ma$  orients us usefully in the world of physics. So it is with the law of Bayes.

EFFECT OF PRIOR CHOICE —

MIXTURE PRIORS AND HIERARCHY —

FREQUENTISM AND CHOICE OF PRIOR — Our engineering culture prizes not just *utility* but also *confidence*, since strong guarantees on our designs allow composition of our work into larger systems: equality, unlike similarity, is transitive. For example, we'd often prefer a 99% guarantee of adequate performance over a 90% guarantee of ideal performance. This asymmetry explains our pessimistic obsession with worst-case bounds over best-case bounds, cost functions over fitness functions, and simple models with moderate-but-estimatable errors over rich models with unknowable-but-often-small errors.

The *frequentist* or *distribution-free* style of statistics continues this risk-averse tradition. In the fullest instance of this style, we do inference as if the true unknown prior on  $\mathcal{H}$  is chosen adversarially. That is, we try to find  $\pi$  that makes the following error small:

$$\max_{\mathbf{p}_h} \mathbb{E}_{h \sim \mathbf{p}_h(\cdot)} \mathbb{E}_{\mathbf{o} \sim \mathbf{p}_o(\cdot|h)} c(\pi(\mathbf{o}); h)$$

Intuitively,

P-HACKING —

HIDDEN ASSUMPTIONS —

(MULTIPLE) HYPOTHESIS TESTING —

Let's now consider the case where  $\mathcal{H}$  is a small and finite. We

## TABLE OF CONTENTS

A. Prologue	
B. Linear classification	
linear approximations	
iterative optimization	
generalization bounds	
model selection	
priors and generalization	
optimization tricks	
kernels enrich approximations	
C. Nonlinearities	
fixed featurization	
learned featurizations	
differentiation	
architecture and symmetry	
feature hierarchies	
stochastic gradient descent	
loss landscape shape	
D. Structured inference	
graphical generative models	
inferring conditional marginals	
learning parameters	
hierarchy and mixtures	
hierarchy and transfer	
variational and sampling methods	
amortized inference	
E. Reductions to supervision	
to build a tool, use it	
distributions as maps	
self-supervised: downstream tasks	
self-supervised: autoregression	
reinforcement: exploration-exploitation	
reinforcement: states and q-values	
beyond i.i.d.	
F. Three brief example projects	
example: flooding plains	
example: ancient tablets	
example: pairing flavors	
G. Appendices	
python refresher	
probability refresher	
linear algebra refresher	
calculus refresher	
notes on high dimensions	
notes on bayes' law	