recitation 02 (optional 6.86x notes)

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 solving homework or exams.

Recitation will have more coding practice and much less detail than these notes. We'll skip whole sections of these notes during recitation. Footprints — — in these notes indicate roughly which points we will linger on during recitation. Each footprint represents perhaps three to seven minutes of recitation time.

B. linear models: the basics

linear approximations

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

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. There, 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 that of "representative" photos 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.

TODO: mention one-hot, etc TODO: mention LOWRANK for multiregression? (relevant to futrure intermezzo? 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. TODO: graphic of separability; and how projection can reduce it 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).

geometry of feature-space —

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 $A(x), B(x), y$ are *dependent*!

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

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

TODO: projectivization (say this foreshadows kernel discussion?)

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.

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

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

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 a length-l list of real numbers. 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!

TODO: talk about structured (trees/sequences/etc) output!

iterative optimization

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, ...
— paul mccartney, john lennon

(STOCHASTIC) GRADIENT DESCENT — We have a collection $\mathcal{H}$ of candidate patterns together with a function $1 - \text{trn}_{\mathcal{S}}$ that tells us how good a candidate is. In §A we found a nearly best candidate by brute-force search over all of $\mathcal{H}$; this doesn't scale to the general case, where $\mathcal{H}$ is intractably large. So: *what's a faster algorithm to find a nearly best candidate?*

← We view $1 - \text{trn}_{\mathcal{S}}$ as an estimate of our actual notion of "good": $1 - \text{tst}$.

A common idea is to start arbitrarily with some $h_0 \in \mathcal{H}$ and repeatedly improve to get $h_1, h_2, \cdots$. 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_{10000}.

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

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

LOGISTIC MODELS —

mention convexity and convergence?

We get a historically crucial and intuition-pumping algorithm when we do logistic regression in the limit of low temperatures.

mention convergence? show trajectory in weight space over time – see how certainty degree of freedom is no longer redundant? ("markov")

HUMBLE MODELS — Let's generalize logistic classification to allow for *unknown unknowns*. We'll do this by allowing a classifier to distribute probability mass not only among the labels $\mathcal{Y}$ but also to a special class $\star$ that means "no comment" or "alien input". A logistic classifier always sets $\mathbb{P}_{y|x}[\star|x] = 0$. But we could use other probability models that put nonzero mass on "no comment"; different models give different learning programs. Here are four models we might try:

	LOGISTIC	PERCEPTRON	SVM	GAUSS
$\mathbb{P}_{y\|x}[-1\|x] \propto$	$e^{-d/2}$	$\min(1, e^{-d})/2$	$\min(1, e^{-d-1})/2$	$\epsilon e^{-(d+1)^2/4}$
$\mathbb{P}_{y\|x}[+1\|x] \propto$	$e^{+d/2}$	$\min(1, e^{+d})/2$	$\min(1, e^{+d-1})/2$	$\epsilon e^{-(d-1)^2/4}$
$\mathbb{P}_{y\|x}[\star\|x] \propto$	0	$1 -$ the above	$1 -$ the above	$1 -$ the above
loss name	softplus(z)	srelu(z)	hinge(z)	parab(z)
formula	$\log(1+\exp(z))$	$\max(1,z)+1$	$\max(1,z+1)$	$(z+1)^2$
to outliers	vulnerable	robust	robust	vulnerable
to inliers	sensitive	blind	sensitive	blind
humility	low	medium	high	high
acc bound	good	bad	good	medium

Table 1: **Four popular models for binary classification. Top rows:** Each model's asserted probability that a given x is paired with $y = +1$ or -1 or $\star$. We used the shorthand $d = \vec{w} \cdot \vec{x}$ and $p(z) = \exp(zd)$. And ϵ is any tiny fixed positive number, say, $1/100$. **Middle rows:** loss functions that arise when maximize likelihood using these models. Read further in the notes to see interpretation. **Bottom rows:** qualitative aspects: Are they robust or vulnerable to outliers? Are they sensitive or blind to inliers? Read further in the notes to see interpretation.

Given a list of training examples, a probability model, and a hypothesis $\vec{w}$, we can compute $\vec{w}$'s asserted probability that the training xs correspond to the training ys. It's reasonable to choose $\vec{w}$ so that this probability is maximal. This method is called **maximum likelihood estimation (MLE)**.

The probability of a bunch of independent observations is the same as the product of probabilities of the observations. Taking logarithms turns products into more manageable sums. And — this is a historical convention — we further flip signs $\pm$ to turn maximization to minimization. After this translation, MLE with the logistic model means finding $\vec{w}$ that minimizes

$$\sum_i \log(1+\exp(-y_i\vec{w}\cdot\vec{x}_i)) = \sum_i \text{softplus}(-y_i\vec{w}\cdot\vec{x}_i)$$

Here, softplus is our name for the function that sends z to $\log(1+\exp(z))$.

MLE with the perceptron model, svm model, or gauss model minimizes the same thing, but with $\text{srelu}(z) = \max(0,z)+1$, $\text{hinge}(z) = \max(0,z+1)$, or $\text{parab}(z) = (z+1)^2$ instead of $\text{softplus}(z)$.◦

← Other models we might try will induce other substitutes for softplus. E.g. $\text{shinge}(z) = \text{hinge}(z)^2$ or $\text{hyper}(z) = \text{avg}(z, \sqrt{z^2+4})$.

Two essential properties of softplus are that: (a) it is convex and (b) it upper bounds the step function. Note that srelu, hinge, and parab also enjoy these properties. Property (a) ensures that the optimization problem is relatively easy — under mild conditions, gradient descent is guaranteed to find a global minimum. By property (b), the total loss on a training set upper bounds the rate of erroneous classification on that training set. So loss is a *surrogate* for (in)accuracy: if the minimized loss is nearly zero, then the training accuracy is nearly 100%.

MARGINS —

training behavior!!

probabilistic interpretations; tails response to outliers support vectors

priors and generalization

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

ON OVERFITTING —

LOG PRIORS AND BAYES — FILL IN COMPUTATION AND BASES VISUAL ILLUSTRATION OF HOW CHOICE OF DOT PRODUCT MATTERS

ℓ^p REGULARIZATION; SPARSITY —

ESTIMATING GENERALIZATION —

C. linear models: getting bang for your buck

selecting models that generalize

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

GRID/RANDOM SEARCH —

SELECTING PRIOR STRENGTH —

RULES OF THUMB — dimension generalization bound margin generalization bound

OVERFITTING ON A VALIDATION SET —

ideas in optimization

premature optimization is the root of all evil
— donald knuth

LOCAL MINIMA —

IMPLICIT REGULARIZATION —

LEARNING RATE SCHEDULE —

LEARNING RATES AS DOT PRODUCTS —

kernels enrich approximations

... 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