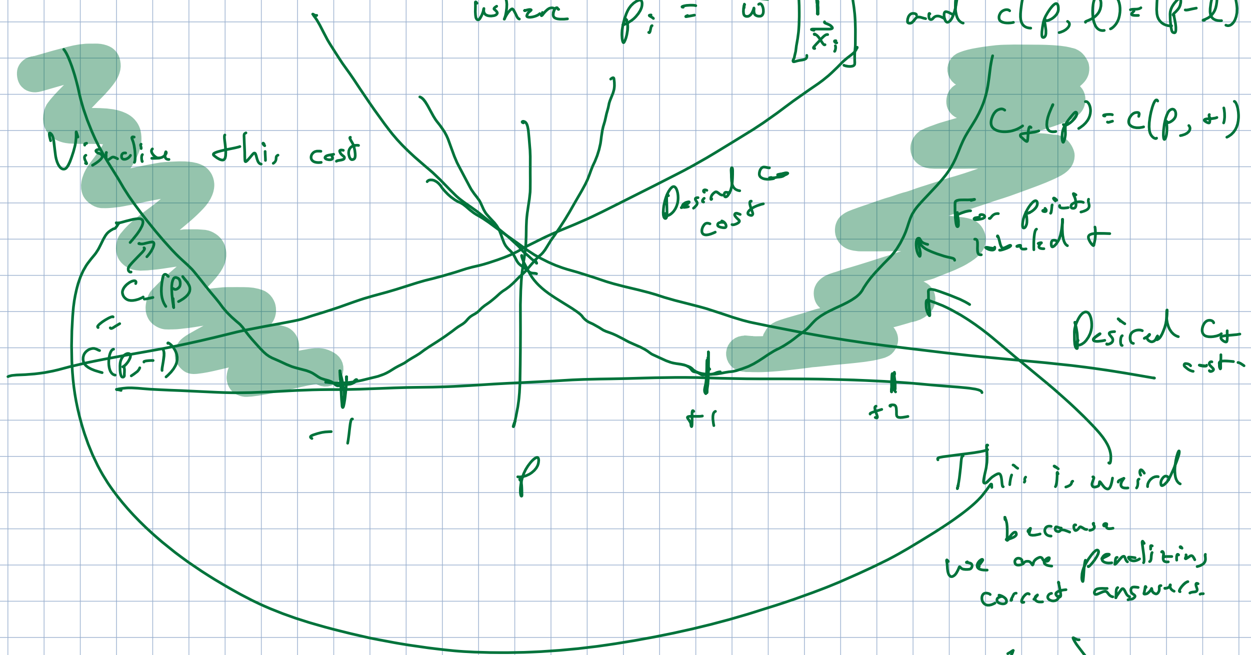
16B Prof. Anant Sahai

Today: Finish classification-inspired optimization story
Loose ends: Complex Inner Products

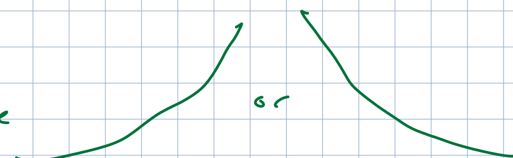
Announce: Happy Thanksgiving Week
No Dis W
No Lec Th
HW 13 due next Fri (Not this Fri)
HW 14 will release before Wed & Due Wed RRR week.

Recall: Have labeled points: $(\vec{x}_i, l_i)$ $i=1,\dots,m$
where $l_i = \pm 1$

What went wrong?! Least-squares is minimizing $\sum_{i=1}^{m} c(p_i, l_i)$
where $p_i = \vec{w}^T \begin{bmatrix} 1 \\ \vec{x}_i \end{bmatrix}$ and $c(p, l) = (p - l)^2$



This is weird because we are penalizing correct answers.

What kind of cost looks like 

e.g. exponential. Can we use a cost $C_+(p) = e^{-p}$ & $C_-(p) = e^{+p}$?

$$c(p, l) = e^{-lp}$$

We want $\underset{\vec{w}}{\arg\min} \sum_{i=1}^{m} \exp\left(-l_i \vec{w}^T \begin{bmatrix} 1 \\ \vec{x}_i \end{bmatrix}\right)$

From now on assume $\vec{x}_i$ has 0^{th} component as 1

$\arg\min \sum^{m} \exp(-l_i \vec{x}_i^T \vec{w})$

$$\underbrace{\qquad \underbrace{\qquad}_{C_i(\vec{x}_i^T\vec{w})}}_{C(\vec{w})}$$

How do I optimize this??

3 approaches:

Ⓐ Pretend this is locally (around $\vec{w}_0$) quadratic and massage it into least-squares form to solve.

$$C(\vec{w}) \approx C(\vec{w}_0) + \frac{d}{d\vec{w}} C\Big]_{\vec{w}_0} (\vec{w} - \vec{w}_0) + \frac{1}{2}(\vec{w} - \vec{w}_0)^T H_w(C)\Big|_{\vec{w}_0} (\vec{w} - \vec{w}_0)$$

Ⓑ Look for a local min/max by seeing when the system of nonlinear equations $\frac{\partial}{\partial \vec{w}} C(\vec{w}) = \vec{0}^T$ is solved.

(Use Newton's Method)

It turns out that these two give the same algorithm.

Ⓒ Pretend it is linear and just iterate $\Longrightarrow$ go down the slope a little.

SUPER IMPORTANT

Computers can take derivatives automatically, e.g. Pytorch. (also tensorflow, JAX, etc...)

$$C(\vec{w}) \approx C(\vec{w}_0) + \frac{d}{d\vec{w}} C\Big]_{\vec{w}_0} (\vec{w} - \vec{w}_0) + \frac{1}{2}(\vec{w} - \vec{w}_0)^T H_w(C)\Big|_{\vec{w}_0} (\vec{w} - \vec{w}_0)$$

Where

$$\frac{d}{d\vec{w}} C = \begin{bmatrix} \frac{\partial C}{\partial w[0]} & \frac{\partial C}{\partial w[1]} & \cdots & \frac{\partial C}{\partial w[n]} \end{bmatrix}$$

$$H_{\vec{w}} C = \begin{bmatrix} \frac{\partial^2 C}{\partial w[0]^2} & \frac{\partial^2 C}{\partial w[1]\partial w[0]} & \frac{\partial^2 C}{\partial w[2]\partial w[0]} & \cdots \\ \vdots & & \ddots & \\ & & & \frac{\partial^2 C}{\partial w[n]^2} \end{bmatrix}$$

How to minimize something like this? (exploit locally quadratic)

1) Choose some initial $\vec{w}_0$ (e.g. $\vec{w}_0 = \vec{0}$)
Set $i = 0$

2) Take a local quadratic approximation around $\vec{w}_i$

3) Find $\vec{\tilde{w}}_{i+1}$ = minimizer of this quadratic

Set $\vec{w}_{i+1} = (1-\eta)\,\vec{w}_i + \eta\,\vec{\tilde{w}}_{i+1}$ of $0 \le \eta \le 1$

"etc"

4) See if "done" If so stop
else goto ②.

A) Is the $c(\vec{w})$ not changing by much.
A') Is w_{i+1} very close to w_i?
B) Have you run out of time.

How to minimize this quadratic?

$$\underbrace{\frac{1}{2}\vec{w}^T H \vec{w}}_{\substack{n\times n \text{ matrix} \\ \text{Hessian}}} + \underbrace{\vec{b}^T \vec{w}}_{\substack{n\text{-dim row} \\ \frac{d}{d\vec{w}} c(\vec{w})\big|_{\vec{w}_i}}} + d$$

HOPE: Can we just match terms to least squares?

Do we care about constant term d?

No. It doesn't matter for argmin.

Hope $\exists A$ so that
$\frac{1}{2} H = A^T A \Rightarrow H = 2A^T A$

What do we know?
Least Squares!

$A\vec{w} \approx \vec{y}$

Find $\vec{w}$ that minimizes $\|A\vec{w} - \vec{y}\|^2$

$\|A\vec{w} - \vec{y}\|^2 = (\vec{w}^T A^T - \vec{y}^T)(A\vec{w} - \vec{y})$
$= \vec{w}^T A^T A \vec{w} - 2\vec{y}^T A \vec{w} + \vec{y}^T \vec{y}$

Solved by $\hat{\vec{w}} = (A^T A)^{-1} A^T \vec{y}$

Also could use SVD

Not worried about invertibility
Can threshold to zero very small singular values

If $A = U_1 \Sigma_c V_1^T$

Then $(A^T A)^{-1} A^T = V_1 \Sigma_c^{-1} U_1^T$

Hope $\exists \vec{y}$ so that
$$\vec{b}^T = -2\vec{y}^T A$$
$\leftarrow$ I need A to be invertible.
$$\vec{y} = -\frac{1}{2} A^{-T} \vec{b}$$

With these hopes, I will get the solution:
$$\begin{aligned}\hat{\vec{w}} &= (A^T A)^{-1} A^T \vec{y} \\ &= \left(\tfrac{1}{2} H\right)^{-1} \left(-\tfrac{1}{2}\right) \cancel{A^T A^{-T}} \vec{b} \\ &= -H^{-1} \vec{b} \\ &= -H^{-1} \underbrace{\left(\frac{d}{d\vec{w}} c\right)^T}_{\text{Called the Gradient } \nabla_{\vec{w}} c}\end{aligned}$$

Still hanging on hope....

Is there an A s.t. $\frac{1}{2}H = A^T A$??

$$\begin{aligned}\frac{d}{d\vec{w}} c(\vec{w}) &= \sum_{i=1}^{m} \frac{\partial}{\partial \vec{w}} c_i(\vec{x}_i^T \vec{w}) \\ &= \sum_{i=1}^{m} c_i'(\vec{x}_i^T \vec{w}) \underbrace{\left[x_i[0]\ x_i[1] \ - - - \ \cdots\ x_i[n]\right]}\end{aligned}$$

$$\frac{\partial}{\partial \vec{w}} [\vec{x}_i^T \vec{w}] = \frac{\partial}{\partial \vec{w}} \sum_{k=0}^{n} x_i[k]\, w[k]$$

$$= \sum_{i=1}^{m} c_i'(\vec{x}_i^T \vec{w})\ \vec{x}_i^T$$

$$\begin{aligned}H_{\vec{w}}\, c &= \frac{d}{d\vec{w}} \left[\frac{d}{d\vec{w}} c(\vec{w})\right]^T \\ &= \sum_{i=1}^{m} \frac{d}{d\vec{w}} c'(\vec{x}_i^T \vec{w})\ \vec{x}_i \\ &= \sum_{i=1}^{m} \vec{x}_i\ \frac{d}{d\vec{w}} c'(\vec{x}_i^T \vec{w}) \quad \text{By same logic as} \\ &= \sum_{i=1}^{m} \vec{x}_i\ c''(\vec{x}_i^T \vec{w})\ \vec{x}_i^T \\ &\overset{?}{=} 2A^T A \quad \text{for some } A??\end{aligned}$$

Notice that if $c_i''(\vec{x}_i^T \vec{w}_0) \geq 0$, then

$$H_{\vec{w}}\, c = \sum_{i=1}^{m} \vec{x}_i \sqrt{c_i''} \cdot \sqrt{c_i''}\ \vec{x}_i^T \implies$$

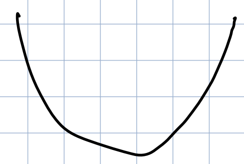
$$= 2\, A^T A$$

So
$$A = \frac{1}{\sqrt{2}} \begin{bmatrix} \sqrt{c_1''} & \vec{x}_1^T \\ \sqrt{c_2''} & \vec{x}_2^T \\ \vdots & \\ \sqrt{c_m''} & \vec{x}_m^T \end{bmatrix}$$

i^{th} position

$$\left(\sqrt{c_i''(\vec{x}_i^T \vec{w}_0)}\right) \vec{x}_i^T$$

Aside: What are $\overset{\text{Scalar}}{\text{functions}}$ whose second derivative is always ≥ 0 called?



<u>Convex</u> $\leftarrow$ From calculus.

Note: e^x and e^{-x} are both convex ☺

This is why convexity is something people can care about.

See 127 for more....

<u>Let's check to make sure things make sense.</u>

Let's use squared error:

$$c_i(p) = (p - l_i)^2$$

$$c_i'(p) = 2(p - l_i)$$

$$c_i''(p) = 2$$

$$\frac{1}{2}H = \sum_{i=1}^{m} \vec{x}_i \vec{x}_i^T = A^T A \quad \text{if} \quad A = \begin{bmatrix} \vec{x}_1^T \\ \vdots \\ \vec{x}_m^T \end{bmatrix} \quad \checkmark$$

$$\frac{d}{d\vec{w}} c = \sum_{i=1}^{m} 2(\vec{x}_i^T \vec{w} - l_i)\vec{x}_i^T$$

$$= 2(A\vec{w} - \vec{l})^T A$$

Recall: In a least-squares problem $A = \begin{bmatrix} \vec{x}_1^T \\ \vdots \\ \vec{x}_m^T \end{bmatrix}$ since $\|A\vec{w} - \vec{l}\|^2 = \sum_{i=1}^{m} (\vec{x}_i^T \vec{w} - l_i)^2$

Put everything together.

$$\|A(\vec{w}_0 + \vec{\delta w}) - \vec{l}\|^2 = ((A\vec{w}_0 - \vec{l}) + A\vec{\delta w})^T ((A\vec{w}_0 - \vec{l}) + A\vec{\delta w})$$

$$= \|A\vec{w}_0 - \vec{l}\|^2 + \vec{\delta w}^T A^T A \vec{\delta w} + 2(A\vec{w}_0 - \vec{l})^T A \vec{\delta w}$$

$$\approx \|A\vec{w}_0 - \vec{l}\|^2 + 2(A\vec{w}_0 - \vec{l})^T A \vec{\delta w} + \vec{\delta w}^T A^T A \vec{\delta w}$$

<u>Passes</u> For something actually quadratic, the local quadratic approximation is exact.

Understanding "transpose trick" i.e. Gradient Descent.
Just expand out to first derivative:

$$c(\vec{\omega}_0 + \vec{\delta\omega}) \simeq c(\vec{\omega}_0) + \underbrace{\left.\frac{dc}{d\vec{\omega}}\right|_{\vec{\omega}_0}}_{\text{row vector}} \vec{\delta\omega}$$

Want to minimize or maximize? What direction should $\vec{\delta\omega}$ be?

We know by Cauchy-Schwartz $|\langle \vec{a}, \vec{b} \rangle| \le \|\vec{a}\| \cdot \|\vec{b}\|$

with equality only when $\vec{a} = \alpha \vec{b}$

$\vec{b}^T \vec{a}$

So $$\vec{\delta\omega} = \left[\left.\frac{dc}{d\vec{\omega}}\right|_{\vec{\omega}}\right]^T$$ is the direction that maximizes the change in c

$$\vec{\delta\omega} = \underbrace{-\left[\left.\frac{dc}{d\vec{\omega}}\right|_{\vec{\omega}_0}\right]^T}$$ is the direction that minimizes

$\nabla_{\vec{\omega}} c(\vec{\omega})$: Gradient of cost w.r.t. $\vec{\omega}$

scalar step size aka learning rate

For Gradient Descent: $\vec{\omega}_{i+1} = \vec{\omega}_i - \eta \nabla_{\vec{\omega}} c(\vec{\omega}_i)$

For squared error (Least-Squares) $$\nabla_{\vec{\omega}} C(\vec{\omega}_i) = \left[\left.\frac{d}{d\vec{\omega}} c(\vec{\omega})\right|_{\vec{\omega}_i}\right]^T$$

$$= 2A^T(A\vec{\omega}_i - \vec{l})$$

$$\vec{\omega}_{i+1} = \vec{\omega}_i + \eta \cdot 2 \cdot A^T \underbrace{(\vec{l} - A\omega_i)}_{\text{residual.}}$$

Source of A^T in HW Problem

Remember: η can't be too big or dynamics go unstable.

For generic losses. $\nabla_{\vec{w}} c(\vec{w}) = \sum_{i=1}^{m} c_i'(\vec{x}_i^T \vec{w})\vec{x}_i$

$$\vec{w}_{i+1} = \vec{w}_i + \eta A^T \underbrace{\begin{bmatrix} -c_1'(\vec{x}_1^T \vec{w}_i) \\ \vdots \\ -c_m'(\vec{x}_m^T \vec{w}_i) \end{bmatrix}}_{\text{negative}}$$

These derivatives of the losses take the place of the residual.

<u>Loose End</u>: When we did upper-triangularization, we restricted to real eigenvalues / eigenvectors.

$\Longrightarrow$ We need a complex inner-product.

For real vectors $\langle \vec{a}, \vec{b} \rangle = \vec{b}^T \vec{a} = \vec{a}^T \vec{b} = \sum_{i=1}^{n} a_i b_i$

Doesn't work for complex vectors...

Want $\underbrace{\|\vec{a}\| = \sqrt{\sum_{i=1}^{n} |a_i|^2}}_{\text{makes sense}} = \langle \vec{a}, \vec{a} \rangle$

For complex vectors makes sense but squaring entries won't give us magnitude squared

$\langle \vec{a}, \vec{b} \rangle \overset{?}{=} \overline{\vec{b}}^T \vec{a} = \sum_{i=1}^{n} a_i \overline{b_i}$ $\Leftarrow$ We choose this one.

$\overset{?}{=} \overline{\vec{a}}^T \vec{b} = \sum_{i=1}^{n} \overline{a_i}\, b_i$

i.e. $\langle \vec{a}, \vec{b} \rangle = \vec{b}^* \vec{a}$

when $\vec{b}^* = (\overline{\vec{b}})^T$

called conjugate transpose.

Next time: finish up complex inner products. (Also discussions next week)
& Review.