303981355 VL5B 2050

Euclidean norm $\|x\| = \sqrt{x_1^2 + \dots + x_n^2}$

normalize 商

$H: w \cdot x + \alpha = 0$. $w \perp$ hyperplane. α 平移.

$\frac{1}{\|w\|}(w \cdot x + \alpha)$ is signed distance. +ve on w's side of H

$\frac{\alpha}{\|w\|}$ is distance to origin

loss → cost/risk/objective

decision fn/model

batch GD → 全部

$Var[X] = E[X^2] - (E[X])^2$

with replacement 放回

range $(X^T X)$ = range (X^T)

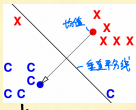
select feature

forward. 向前遍历. top-down 自顶

backward. 倒 扫

Centroid method

$f(x) = \underbrace{(\mu_C - \mu_X)}_{\text{normal vector}} \cdot x - (\mu_C - \mu_X) \cdot \underbrace{\frac{\mu_C + \mu_X}{2}}_{\text{midpoint between } \mu_C, \mu_X}$



$X \to n \times d$. ① centering $X \to \dot{X}$. 每一行 $X_i^T - \mu^T$

② decorrelate $\dot{X} \to Z$. $Z = \dot{X} \cdot V$. $\Sigma = V \Lambda V^T$

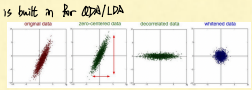
$X \sim N(\mu, \Sigma) \to Z \sim N(0, \Lambda)$

③ whitening $X \to W$ $W = \dot{X} \cdot \Sigma^{-\frac{1}{2}}$

sphering $\dot{X} \to W$

$X \sim N(\mu, \Sigma) \to W \sim N(0, I)$

whitening is built in for QDA/LDA



Perceptrons + SGD. linear. binary.

X_i. 矩阵第一行 ⇒ $y = \pm 1$

全 DB 此 最正 $z = \begin{cases} X_i \cdot w \geq 0 \Rightarrow \text{predict } 1 \\ X_i \cdot w \leq 0 \Rightarrow \text{predict } -1 \end{cases}$ (无α)

signed distance

⇒ constant $y_i X_i \cdot w \geq 0$

online (一边到一边边走)

must succeed for lin-seperable in $O(\frac{r^2}{\tau^2})$

$r \to$ longest feature

$\tau \to$ max margin

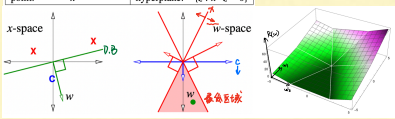
loss $L(z, y_i) = \begin{cases} 0 & \text{if } y_i z \geq 0, \text{ and} \\ -y_i z & \text{otherwise.} \end{cases}$

risk $R(w) = \sum_{i=1}^{n} L(X_i \cdot w, y_i)$,

$= \sum_{i \in V} -y_i X_i \cdot w$

find hyperplane in x-space ⇒ find optimal pt in w-space

x-space	w-space
hyperplane: $\{z : w \cdot z = 0\}$	point: w
point: x	hyperplane: $\{z : x \cdot z = 0\}$



$\nabla_w R = \begin{bmatrix} \frac{\partial R}{\partial w_1} \\ \frac{\partial R}{\partial w_2} \\ \vdots \\ \frac{\partial R}{\partial w_d} \end{bmatrix}$ $\nabla_w (z \cdot w) = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_d \end{bmatrix} = z$ $\nabla_w R = -\sum_{i \in V} y_i X_i$

上升方向

$w = w - \varepsilon \cdot \nabla R$

$w \leftarrow$ arbitrary nonzero starting point (good choice is any $y_i X_i$)

while $R(w) > 0$

$V \leftarrow$ set of indices i for which $y_i X_i \cdot w < 0$

$w \leftarrow w + \epsilon \sum_{i \in V} y_i X_i$

return w

slow $O(nd)$/step

SGD. while some $y_i X_i \cdot w < 0$

$w \leftarrow w + \epsilon y_i X_i$

return w

GD on loss fn

$O(d)$/step

SGD must succeed for perceptron

有 d $[w_1, w_2, \alpha] \cdot \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix}$.

put pt at z=1 plane

⇒ hyperplane through origin in 3D can cut the 2D plane x3=1 at non-origin place

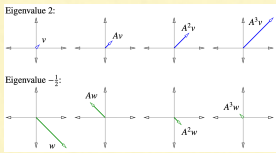
linear algebra

For square matrix A. $A v = \lambda v$

eigenvector eigenvalue

v is eigenvector of A^k with eigenvalue λ^k

A^{-1} (if invertible) λ^{-1}



Spectral: symmetric square $n \times n$ matrix has n mutually orthogonal eigenvector. $v_i^T \cdot v_j = 0$

若 2 个 eigenvector 共 value → 任一

⇒ 任一 linear 都是 eigenvector with that value

Choose n mutually orthogonal **unit** n-vectors $v_1, \dots, v_n$

Let $V = [v_1 \quad v_2 \quad \dots \quad v_n]$ ⇐ $n \times n$ matrix

Observe: $V^T V = I$ $V^T = V^{-1}$ $\Lambda = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix}$

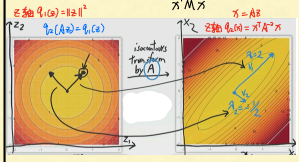
$A = V \Lambda V^T = \sum_{i=1}^{n} \lambda_i v_i v_i^T$ has chosen eigenvectors/values

symmetric outer product: $n \times n$ matrix, rank 1

$A = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 0 & -1/2 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}$

Isocontour of quadratic form $x^T A^2 x$ determined by eigenvector/value of A

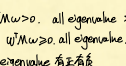
$x^T M x$ $y = Ax$ $V \Lambda^{-\frac{1}{2}} V^T = M^{-\frac{1}{2}}$



A/M is diagonal

eigenvectors are coordinate axes

ellipsoids are axis-aligned



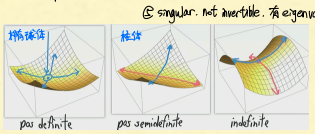
For symmetric M. ① pos-def. $w^T M w > 0$. all eigenvalue > 0

② pos-semidef. $w^T M w \geq 0$. all eigenvalue ≥ 0

$\begin{cases} A^2 \text{ is pos-semidef.} \\ A^2 \text{ is pos-def} \end{cases}$ ③ indefinite. eigenvalue 有正有负

④ invertible. eigenvalue $\neq 0$

⑤ singular. not invertible. 有 eigenvalue $= 0$



hard SVM

$\begin{cases} X_i \cdot w + \alpha \geq 0 \Rightarrow y = 1 \\ X_i \cdot w + \alpha \leq 0 \Rightarrow y = -1 \end{cases}$

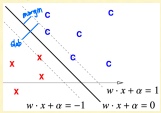
⇒ constraint. $y_i (X_i w + \alpha) \geq 1$

margin $= \min_i \frac{1}{\|w\|} |w \cdot X_i + \alpha| \geq \frac{1}{\|w\|}$

signed distance ≥ 1

max margin ⇒ min $\|w\|$ ⇒ min $\|w\|^2$

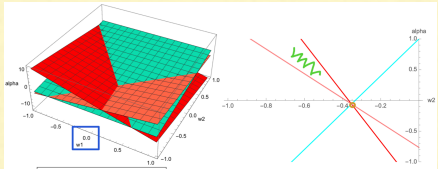
unique/no solution



⇒ Quadratic program (d+1-dim & n constraint)

Find w and α that minimize $\|w\|^2$

subject to $y_i(X_i \cdot w + \alpha) \geq 1$ for all $i \in [1, n]$



weight3d.pdf, weightcross.pdf [This is an example of what the linear constraints look like in the 3D weight space (w_1, w_2, α) for the SVM we've been studying with three training points. The SVM is looking for the point nearest the α-axis that lies above the blue plane (representing an in-class training point) but below the red and pink planes (representing out-of-class training points). In this example, that optimal point lies where the three planes intersect. At right we see a 2D cross section $w_1 = 1/17$ of the 3D space, because the optimal solution lies in this cross section. The constraints say that the solution must lie in the leftmost pizza slice, while being as close to the origin as possible, so the optimal solution is where the three lines meet.]

Prob theory

10% population has cancer 90% not

job	(X)	miner	farmer	other
cancer	(Y = 1)	20%	50%	30%
no cancer	(Y = −1)	1%	10%	89%

$P(X) = P(X|Y=1)\,P(Y=1) + P(X|Y=-1)\,P(Y=-1)$

$P(X = \text{farmer}) = 0.5 \times 0.1 + 0.1 \times 0.9 = 0.14$ 14%人是 farmer

↓ posterior probability 为 ↓ prior prob. ↓ if X = farmer

$P(Y=1|X) = \frac{P(X|Y=1)P(Y=1)}{P(X)}$ 10% $= \frac{0.05}{0.14}$

$P(Y=-1|X) = \frac{P(X|Y=-1)P(Y=-1)}{P(X)} = \frac{0.09}{0.14}$

$P(\text{cancer} \mid \text{farmer}) = 5/14 \approx 36\%$. farmer has cancer

asymmetric loss

$L(z, y) = \begin{cases} 1 & \text{if } z = 1, y = -1, \quad \text{false positive is bad} \\ 5 & \text{if } z = -1, y = 1, \quad \text{false negative is BAAAAAD} \\ 0 & \text{if } z = y. \end{cases}$

symmetric: if $z \geq y$ ⇒ pick class with highest post prob

$r^*(x) = \begin{cases} 1 & \text{if } L(-1, 1)\,P(Y=1|X=x) > L(1, -1)\,P(Y=-1|X=x) \\ -1 & \text{otherwise} \end{cases}$

cancer / no cancer

best decision rule

for asymmetric loss

r *(miner) = 1, r *(farmer) = 1, and r *(other) = -1.

Bayes risk/**optimal** risk (No decision rule gives a lower risk)

$R(r^*) = 0.1(5 \times 0.3) + 0.9(1 \times 0.01 + 1 \times 0.1) = 0.249$

SOFT SVM

slack var

Constraint: $y_i(X_i \cdot w + \alpha) \geq 1 - \xi_i$ $\xi_i \geq 0$

margin $= \frac{1}{\|w\|}$

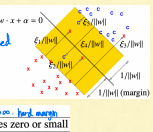
⇒ Quadratic program (d+1-dim & n constraint)

must ≥ 0 or option unbounded

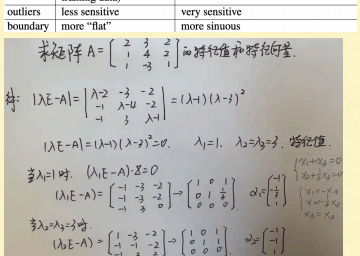
Find w, α, and ξ_i that minimize $\|w\|^2 + C \sum_{i=1}^{n} \xi_i$

subject to $y_i(X_i \cdot w + \alpha) \geq 1 - \xi_i$ for all $i \in [1, n]$

$\xi_i \geq 0$ for all $i \in [1, n]$



	small C 宽margin	big C slow. C→∞. hard margin
desire	maximize margin $1/\|\|w\|\|$	keep most slack variables zero or small
danger	underfitting (misclassifies much training data)	overfitting (awesome training, awful test)
outliers	less sensitive	very sensitive
boundary	more "flat"	more sinuous



求矩阵 $A = \begin{bmatrix} 2 & 3 & 2 \\ 3 & 4 & 2 \\ 1 & -3 & 1 \end{bmatrix}$ 的特征值和特征向量.

解: $|\lambda E - A| = \begin{vmatrix} \lambda-2 & -3 & -2 \\ -1 & \lambda-4 & -2 \\ -1 & 3 & \lambda-1 \end{vmatrix} = (\lambda-1)(\lambda-3)^2$

$|\lambda E - A| = (\lambda-1)(\lambda-3)^2 = 0$. $\lambda_1 = 1$. $\lambda_2 = \lambda_3 = 3$. 特征值.

当 $\lambda_1 = 1$ 时. $(\lambda_1 E - A) \cdot \vec{x} = 0$

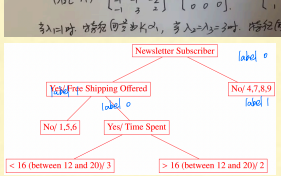
$(\lambda_1 E - A) = \begin{bmatrix} -1 & -3 & -2 \\ -1 & -3 & -2 \\ -1 & 3 & 0 \end{bmatrix} \to \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & \frac{1}{3} \\ 0 & 0 & 0 \end{bmatrix}$. $\alpha_1 = \begin{bmatrix} -1 \\ -\frac{1}{3} \\ 1 \end{bmatrix}$

$\begin{cases} x_1 = -x_3 \\ x_2 = -\frac{1}{3}x_3 \\ x_3 = x_3 \end{cases}$

当 $\lambda_2 = \lambda_3 = 3$ 时.

$(\lambda_2 E - A) = \begin{bmatrix} 1 & -3 & -2 \\ -1 & -1 & -2 \\ -1 & 3 & -2 \end{bmatrix} \to \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$. $\alpha_2 = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$

当 $\lambda = 1$ 时. 对应的向量为 $k_1\alpha_1$, 当 $\lambda_2 = \lambda_3 = 3$ 时. 特征向量为 $k_2\alpha_2$



$P(n, k) = \frac{n!}{(n-k)!}$ $\binom{n}{k} = \frac{n!}{k!(n-k)!} = \frac{1}{k!} \cdot n \cdot (n-1) \cdots (n-k+1)$

Isotropic. σ

each class

$X \sim N(\mu, \sigma^2)$: $f(x) = \frac{1}{(\sqrt{2\pi}\sigma)^d} \exp\left(-\frac{\|x - \mu\|^2}{2\sigma^2}\right)$

[μ & x = vectors; σ = scalar; d = dimension]

post = 予 · prior = $\mathbb{P}_{X|Y=C}(x) \cdot \mathbb{P}(Y=C)$ (π_C)

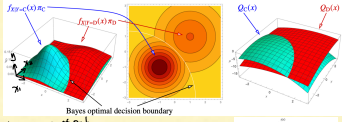
QDA quadratic decision boundary

max

$Q_C(x) = \ln\left((\sqrt{2\pi})^d f_{X|Y=C}(x)\pi_C\right) = -\frac{\|x - \mu_C\|^2}{2\sigma_C^2} - d \ln \sigma_C + \ln \pi_C$

Quadratic

$r^*(x) = \begin{cases} C & \text{if } Q_C(x) - Q_D(x) > 0, \\ D & \text{otherwise.} \end{cases}$ decision boundary: $Q_C(x) - Q_D(x) = 0$



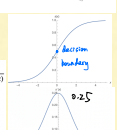
recover post prob

$P(Y = C|X) = \frac{f_{X|Y=C}\pi_C}{f_{X|Y=C}\pi_C + f_{X|Y=D}\pi_D}$

$P(Y = C|X = x) = \frac{e^{Q_C(x)}}{e^{Q_C(x)} + e^{Q_D(x)}} = \frac{1}{1 + e^{Q_D(x) - Q_C(x)}}$

$= s(Q_C(x) - Q_D(x))$, where

$s(y) = \frac{1}{1 + e^{-y}}$ ⇐ logistic fn aka sigmoid fn



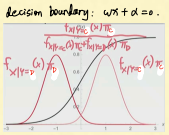
LDA linear boundary. same σ / Σ

$Q_C(x) - Q_D(x) = \underbrace{\frac{(\mu_C - \mu_D) \cdot x}{\sigma^2}}_{w \cdot x} \underbrace{- \frac{\|\mu_C\|^2 - \|\mu_D\|^2}{2\sigma^2} + \ln \pi_C - \ln \pi_D}_{+\alpha}$.

decision boundary: $w x + \alpha = 0$.

post prob: $s(wx + \alpha)$

multi-class



max $\frac{\mu_C \cdot x}{\sigma^2} - \frac{\|\mu_C\|^2}{2\sigma^2} + \ln \pi_C$

$\pi_D = \pi_C$. LDA → centroid

MLE

① prior. 10 抛 8 正. 抛 3 打 P. $X \sim B(n, p)$

$L(p) = \mathbb{P}[X = 8] = \binom{n}{8} p^8 (1-p)^{n-8}$

(likelihood) $= \binom{10}{8} p^8 \cdot (1-p)^2$

$\frac{d}{dp} = 0 \Leftrightarrow p = 0.8$ [$p = 8/10$ = 抛多 co. max]

$\hat{\pi}_C = \#\text{ 对 } \% / n$

② mean & variance

$L(\mu, \sigma; x_1, x_2 \cdots x_n) = f(x_1) \cdots f(x_n)$

(likelihood)

$\ell(\mu, \sigma; X_1, \dots, X_n) = \ln f(X_1) + \ln f(X_2) + \dots + \ln f(X_n)$

$= \sum_{i=1}^{n} \underbrace{\left(-\frac{\|X_i - \mu\|^2}{2\sigma^2} - d \ln \sqrt{2\pi} - d \ln \sigma\right)}_{\text{ln of normal PDF}}$

Set $\nabla_\mu \ell = 0$, $\frac{\partial \ell}{\partial \sigma} = 0$

$\nabla_\mu \ell = \sum_{i=1}^{n} \frac{X_i - \mu}{\sigma^2} = 0 \Rightarrow \hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} X_i$

$\frac{\partial \ell}{\partial \sigma} = \sum_{i=1}^{n} \frac{\|X_i - \mu\|^2 - d\sigma^2}{\sigma^3} = 0 \Rightarrow \hat{\sigma}^2 = \frac{1}{dn} \sum_{i=1}^{n} \|X_i - \mu\|^2$

LDA: $\hat{\sigma}^2 = \frac{1}{dn} \sum_C \sum_{\{i: y_i = C\}} \|X_i - \hat{\mu}_C\|^2$

$\hat{\sigma}^2 = \frac{1}{N} \sum_{i=1}^{N} (X_i - \hat{\mu})^2$

Aisotropic. Σ

$X \sim N(\mu, \Sigma)$

d×1 d×d

$f(x) = \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)\right)$

↑ determinant of Σ

precision matrix

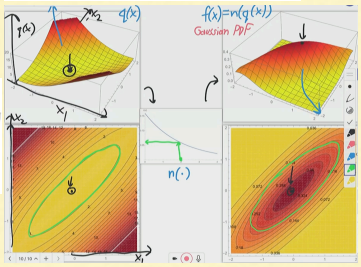
Write $f(x) = n(q(x))$, where $q(x) = (x - \mu)^T \Sigma^{-1} (x - \mu)$

quadratic gaussian↑ ↑

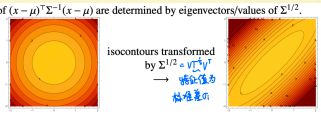
$\mathbb{R} \to \mathbb{R}$, exponential $\mathbb{R}^d \to \mathbb{R}$, quadratic

$k = 2$

$\nabla_x f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_d} \end{bmatrix} = \left(\frac{\partial f}{\partial x}\right)^T$



The isocontours of $(x - \mu)^T \Sigma^{-1} (x - \mu)$ are determined by eigenvectors/values of $\Sigma^{1/2}$.



Aside: $q(x)$ is the squared distance from $\Sigma^{-1/2} x$ to $\Sigma^{-1/2} \mu$. Consider the metric

$d(x, \mu) = \|\Sigma^{-1/2} x - \Sigma^{-1/2} \mu\| = \sqrt{(x - \mu)^T \Sigma^{-1} (x - \mu)} = \sqrt{q(x)}$.

[So we think of the precision matrix as a "metric tensor" which defines a metric, a sort of warped distance from x to the mean μ.]

$\text{Cov}(R, S) = E[(R - E[R])(S - E[S])^T] = E[RS^T] - \mu_R \mu_S^T$

$\text{Var}(R) = \text{Cov}(R, R)$

$\Sigma = \text{Var}(R) = \begin{bmatrix} \text{Var}(R_1) & \text{Cov}(R_1, R_2) & \dots & \text{Cov}(R_1, R_d) \\ \text{Cov}(R_2, R_1) & \text{Var}(R_2) & & \text{Cov}(R_2, R_d) \\ \vdots & & \ddots & \vdots \\ \text{Cov}(R_d, R_1) & \text{Cov}(R_2, R_1) & \dots & \text{Var}(R_d) \end{bmatrix} = V \Gamma V^T$

R_i is vector

R_i is scalar

symmetric

特征值为 variance = σ_i^2

R_i 两两独立 → $\text{Cov} = 0 \Rightarrow \Sigma$ is diagonal

$f(x) = f(x) \cdots f(x) \Leftrightarrow$ joint normal

→ 椭圆轴的坐标

半径与对角线元素的平方根

MLE

QDA $\hat{\Sigma}_C = \frac{1}{n_C} \sum_{i: y_i = C} \frac{(X_i - \hat{\mu}_C)(X_i - \hat{\mu}_C)^T}{\text{outer product matrix, } d \times d}$ pos-semidef. may not work

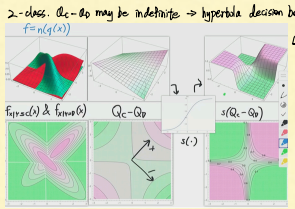
LDA $\hat{\Sigma} = \frac{1}{n} \sum_C \sum_{i: y_i = C} (X_i - \hat{\mu}_C)(X_i - \hat{\mu}_C)^T$

QDA

max

$Q_C(x) = \ln\left((\sqrt{2\pi})^d f_{X|Y=C}(x)\pi_C\right) = -\frac{1}{2}(x - \mu_C)^T \Sigma_C^{-1} (x - \mu_C) - \frac{1}{2} \ln|\Sigma_C| + \ln \pi_C$

2-class. $Q_C - Q_D$ may be indefinite → hyperbola decision boundary (在 isotropic 不可能出现)



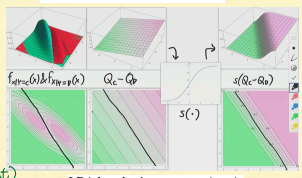
LDA

$Q_C(x) - Q_D(x) = \underbrace{(\mu_C - \mu_D)^T \Sigma^{-1}}_{w^T} x \underbrace{- \frac{\mu_C^T \Sigma^{-1} \mu_C - \mu_D^T \Sigma^{-1} \mu_D}{2}}_{+\alpha} + \ln \pi_C - \ln \pi_D$

Decision boundary is $w^T x + \alpha = 0$.

Posterior is $P(Y = C|X = x) = s(w^T x + \alpha)$

max $\mu_C^T \Sigma^{-1} x - \frac{\mu_C^T \Sigma^{-1} \mu_C}{2} + \ln \pi_C$



2 class ⇒ LDA has $d + 1$ parameters (w, α);

QDA has $\frac{d(d+3)}{2} + 1$ params;

PCA feature are linear comb of input feature

Kernel matrix pos-semidef, symmetric

k-means. 全垂平分线 linear boundary

k=n. stop at iteration 1

may not converge

deep tree increase difference

kernel perceptron + Gaussian filter = smoothed kNN

decision tree cost fn $\begin{cases} -p \log p - (1-p) \log(1-p) \\ p(1-p) \end{cases}$ Gini impurity

bagging is same algo

k-means. k↑ cost must ↓

adaboost can give post prob (continuous output)

shatter fn for linear classifier must be poly

- conic section 圆锥曲面
- Ellipsoid 椭球体
- hyperboloid 双曲面
- paraboloid 抛物面
- cylinder 柱体
- double cones 双锥体

adaboost Godel

random forest Netflix

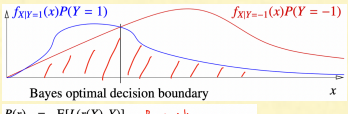
CNN Turing

1对 1对+1错 2

2 2对+1错

3 2对

4 3对



$R(r) = E[L(r(X), Y)]$ Bayes risk

$= P(Y=1) \int L(r(x), 1) f_{X|Y=1}(x)\,dx +$

$P(Y=-1) \int L(r(x), -1) f_{X|Y=-1}(x)\,dx.$

0-1 loss + 0.5 prior ⇒ Bayes DB: $P(Y=1|X=x) = P(Y=-1|X=x) = 0.5$

$R(r) = E[L(r(X), Y)]$ r: $\mathbb{R}^d \to \pm 1$. R: risk fn = expected loss

$= \sum_x \left(L(r(x), 1)\,P(Y=1|X=x) + L(r(x), -1)\,P(Y=-1|X=x)\right) P(X=x)$

$= P(Y=1) \sum_x L(r(x), 1)\,P(X=x|Y=1) + P(Y=-1) \sum_x L(r(x), -1)\,P(X=x|Y=-1)$

$r^*(x) = \begin{cases} 1 & \text{if } L(-1, 1)\,P(Y=1|X=x) > L(1, -1)\,P(Y=-1|X=x) \\ -1 & \text{otherwise} \end{cases}$

(a) $\nabla_x(\mathbf{w}^T \mathbf{x}) = \left(\frac{\partial f}{\partial x}\right)^T = (w^T)^T = w$. first principle: $w^T(x+\Delta) = w^T x + w^T \Delta = f(x) + w^T \Delta$

$\frac{\partial f}{\partial x} = w^T$ $\nabla_x f = w$

(b) $\nabla_x(\mathbf{w}^T \mathbf{A} \mathbf{x})$ $w^T A = V^T$ $\nabla_x (V^T x) = V = A^T w$

(c) $\nabla_A(\mathbf{w}^T \mathbf{A} \mathbf{x})$ $\sum_i \sum_j w_i A_{ij} x_j$ $\frac{\partial}{\partial A_{ij}} \sum_i \sum_j w_i A_{ij} x_j = w_i x_j = [w x^T]_{ij}$

only A_{ij} depend on A_{ij}: derivative on

$k = i, l = j$

outer product ⇒ matrix

(d) $\nabla_x(\mathbf{x}^T \mathbf{A} \mathbf{x})$ first principle: $(x + \Delta)^T A (x + \Delta) = x^T A x + \Delta^T A x + x^T A \Delta + \Delta^T A \Delta$

$= f(x) + x^T A^T \Delta + x^T A \Delta + o(\|\Delta\|^2)$

$\frac{\partial f}{\partial x} = x^T(A^T + A)$. $\nabla_x f = (A^T + A)^T x = (A + A^T) x$

(e) $\nabla_x^2(\mathbf{x}^T \mathbf{A} \mathbf{x})$ $= \nabla_x(\nabla_x(x^T A x)) = \nabla_x((A + A^T)x)$

Hessian $f(x) = (A + A^T)x$ $f(x+\Delta) = (A + A^T)(x + \Delta) = (A + A^T)x + (A + A^T)\Delta$

$\frac{\partial f}{\partial x} = A + A^T$ $\nabla_x^2(x^T A x) = (A + A^T)^T = A^T + A$

(f) Now let's apply our identities derived above to a practical problem. Given a design matrix $X \in \mathbb{R}^{n \times d}$ and a label vector $Y \in \mathbb{R}^n$, the ordinary least squares regression problem is

$$w^* = \arg\min_w \frac{1}{2} \|Xw - Y\|_2^2 \quad \text{norm}$$

Using parts (a)–(e), derive a necessary condition for w^*. *Note: We do not necessarily assume X is full rank! Hint: A necessary condition for a minimum point of a function is that it is a critical point, i.e. where the gradient is 0.*

$\nabla_w \left(\frac{1}{2} \|Xw - Y\|_2^2\right)$

$f(w) = \frac{1}{2} \|Xw - Y\|_2^2 = \frac{1}{2}(Xw - Y)^T (Xw - Y) = \frac{1}{2}(w^T X^T X w - w^T X^T Y - Y^T X w + Y^T Y) = \frac{1}{2}(w^T X^T X w - 2 Y^T X w + Y^T Y)$

$\nabla_w f(w) = \nabla_w \left(\frac{1}{2} w^T X^T X w - Y^T X w + \frac{1}{2} Y^T Y\right) = \left(\frac{1}{2} X^T X + \frac{1}{2}(X^T X)^T\right) w - X^T Y = X^T X w - X^T Y$

(2) $x^T A x$ ⇒ $(A + A^T)x$ (1) $\sqrt{A} x$ ⇒ $A^T w$ not depend on w $= X^T X$

$\Rightarrow X^T X w^* - X^T Y = 0$ $X^T (X w^* - Y) = 0$

Some regression fns: / Lasso fn $h(x;w)$

(1) linear: $h(x;w,\alpha) = w \cdot x + \alpha$

(2) polynomial [equivalent to linear regression with added polynomial features]

(3) logistic: $h(x;w,\alpha) = s(w \cdot x + \alpha)$ recall: logistic fn $s(y) = \frac{1}{1+e^{-y}}$

Some loss fns: let z be prediction $h(x)$; y be true label

(A) $L(z,y) = (z-y)^2$ squared error

(B) $L(z,y) = |z-y|$ absolute error

(C) $L(z,y) = -y \ln z - (1-y)\ln(1-z)$ logistic loss, aka cross-entropy: $y \in [0,1]$, $z \in (0,1)$

Some cost fns to minimize:

(a) $J(h) = \frac{1}{n}\sum_{i=1}^{n} L(h(X_i), y_i)$ mean loss [you can leave out the "$\frac{1}{n}$"]

(b) $J(h) = \max_{i=1}^{n} L(h(X_i), y_i)$ maximum loss

(c) $J(h) = \sum_{i=1}^{n} \omega_i L(h(X_i), y_i)$ weighted sum [some points are more important than others]

(d) $J(h) = (a), (b)$, or $(c) + \lambda\|w\|^2$ ℓ_2 penalized/regularized

(e) $J(h) = (a), (b)$, or $(c) + \lambda\|w\|_{\ell_1}$ ℓ_1 penalized/regularized

Some famous regression methods:

Least-squares linear regr.: (1) + (A) + (a)

Weighted least-squ. linear: (1) + (A) + (c)

Ridge regression: (1) + (A) + (a) + (d)

Lasso: (1) + (A) + (a) + (e)

} quadratic cost; minimize w/calculus

Logistic regr.: (3) + (C) + (a) quadratic program

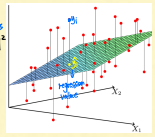
Least absolute deviations: (1) + (B) + (a) convex cost; minimize w/gradient descent

Chebyshev criterion: (1) + (B) + (b)

} linear program

Least square linear reg

Find w, α that minimizes $\sum_{i=1}^{n}(X_i \cdot w + \alpha - y_i)^2$ sensitive to outliers $\Rightarrow RSS(w) = \|Xw - y\|^2$



$$X = \begin{bmatrix} X_{11} & X_{12} & \dots & X_{1j} & \dots & X_{1d} & 1 \\ X_{21} & X_{22} & & X_{2j} & & X_{2d} & 1 \\ \vdots & & & & & & \\ X_{i1} & X_{i2} & & X_{ij} & & X_{id} & 1 \\ \vdots & & & & & & \\ X_{n1} & X_{n2} & & X_{nj} & & X_{nd} & 1 \end{bmatrix} \quad w = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \\ \alpha \end{bmatrix}$$

$n \times (d+1)$ (d+1)

$X_{2j} \leftarrow X^T \rightarrow X_{2d}$ 1 pad

feature column X_{ij}

$\hat{y} = X \cdot w$

$$\begin{aligned} \text{minimize } RSS(w) &= w^T X^T X w - 2y^T X w + y^T y \\ \nabla RSS &= 2X^T X w - 2X^T y = 0 \\ \Rightarrow \underbrace{X^T X}_{(d+1)\times(d+1)} w &= X^T y \quad \Leftarrow \text{the } \boxed{\text{normal equations}} \text{ [} w \text{ unknown; } X \text{ \& } y \text{ known]} \end{aligned}$$

solve

pos-semidef @ pos-def, sol w is unique. rank$(X) = d+1$

(at least one sol) colspace(X) = w-space ($\mathbb{R}^{n}$)

datapoints must span all of w-space ($\mathbb{R}^{d}$)

need $\Rightarrow$ @ signular, sol w is a subspace. under-constrained $\Rightarrow$ fail

Moore-Penrose pseudoinverse (Dis6, Q1)

$\hat{w} = X^+ y$

$\nabla^2 RSS(w) = 2X^T X$

- All eigenvalues are non-negative. Notice for any v in $\mathbb{R}^{d+1}$: $v^T X^T X v = (Xv)^2 \geq 0$.
- Notice 0 eigenvalue if datapoints do not span all of $\mathbb{R}^{d}$.
- Thus, the function is **convex**.

$w = (X^T X)^{-1} X^T y$. w is linear-fnns of y

X^+, the pseudoinverse of X, $(d+1)\times n$

left inverse. $X^+ \cdot X = I$

$\hat{y} = Xw = XX^+ y = Hy$

$H = X \cdot X^+$ ($n \times n$, hat matrix)

Statistical justification

model of reality: $y_i = g(x_i) + \epsilon_i$

$\uparrow$ $\uparrow$ $\uparrow$

fit D $y' = N(\epsilon, \sigma^2)$

Suppose $\epsilon_i \sim N(0, \sigma^2)$; then $y_i \sim N(g(X_i), \sigma^2)$.

Recall that log of normal PDF is

$$\ln f(y_i) = -\frac{(y_i - \mu_i)^2}{2\sigma^2} - \text{constant} \qquad \Leftarrow \mu_i = g(X_i)$$

& log likelihood is

$$\ell(g; X, y) = \ln(f(y_1) f(y_2) \cdots f(y_n)) = \ln f(y_1) + \ldots + \ln f(y_n) = -\frac{1}{2\sigma^2}\sum (y_i - g(X_i))^2 - \text{constant}.$$

Takeaway: Max likelihood on "parameter" $g \Rightarrow$ choose a g that minimizes $\sum(y_i - g(X_i))^2$.

Ridge reg L_2

$h(z) = w^T \cdot z$

Find w that minimizes $\|Xw - y\|^2 + \lambda\|w'\|^2 = J(w)$ w' is w with $\alpha = 0$

$\nabla J = 0 \Rightarrow$ normal eq: $(X^T X + \lambda I')\, w = X^T y$. I' 左下角 $= 0$

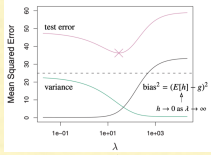
① guarantee pos-def $\rightarrow$ unique sol w always pos-def

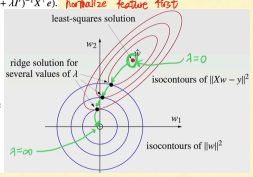
(det) $\cdot (\lambda \neq)$

② reduce overfit especially when nearly ill-posed

Variance of ridge regr. at test pt z is $\mathrm{Var}(z^T(X^T X + \lambda I')^{-1} X^T \epsilon)$. normalize feature first

As $\lambda \to \infty$, variance $\to 0$, but bias increases.





Statistical justification MAP

trust small w' more than big w'

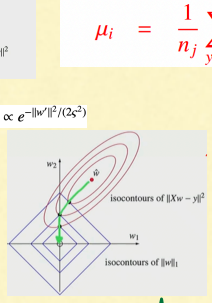
Assign a prior probability on w': $w' \sim N(0, \varsigma^2)$, with PDF $f(w') \propto e^{-\|w'\|^2/(2\varsigma^2)}$

Bayes' Theorem: posterior $f_{W|X,Y}(w) = \frac{f_{Y|X,W}(y)\, f(w')}{f_{Y|X}(y)}$ find

Maximize log posterior $= \ln f_{Y|X,W}(y) + \ln f(w') - \text{const}$

$= -\text{const}\,\|Xw - y\|^2 - \text{const}\,\|w'\|^2 - \text{const}$

$\Rightarrow$ Minimize $\|Xw - y\|^2 + \lambda\|w'\|^2$



LASSO L_1

naturally set some $w_i = 0$ quadratic program

Find w that minimizes $\|Xw - y\|^2 + \lambda\|w'\|_1$ where $\|w'\|_1 = \sum_{i=1}^{d}|w_i|$ (Don't penalize α.)

$\|w\|_p$, $p = 0.5$. pv. 更可能出现 $w_i = 0$

Bias-Variance decomposition

bias: error due to inability of hypothesis h to fit g perfectly

$E[h(x)] - g(x)$ e.g., fitting quadratic g with a linear h

variance: error due to fitting random noise in data

e.g., we fit linear g with a linear h, yet $h \neq g$.

$y_i = g(X_i) + \epsilon_i$ add feature always ↑ variance, sometimes ↓ bias

$Y = g(z) + \epsilon$. $E[Z] = g(z)$. $Var(Y) = Var(\epsilon)$

$R(h) = E[L(h(z), y)]$

↑ take expectation over possible training sets X, y & values of y

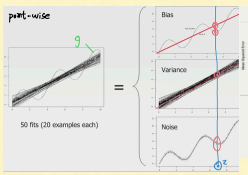
$$\begin{aligned} &= E[(h(z) - y)^2] \\ &= E[(h(z)^2] + E[y^2] - 2E[y\,h(z)] \\ &= \mathrm{Var}(h(z)) + E[(h(z))^2] + \mathrm{Var}(y) + E[y]^2 - 2E[y]E[h(z)] \\ &= (E[h(z)] - E[y])^2 + \mathrm{Var}(h(z)) + \mathrm{Var}(y) \\ &= \underbrace{(E[h(z)] - g(z))^2}_{\text{bias}^2 \text{ of method}} + \underbrace{\mathrm{Var}(h(z))}_{\text{variance of method}} + \underbrace{\mathrm{Var}(\epsilon)}_{\text{irreducible error}} \end{aligned}$$

random from ϵ_i random from (X, y) weight

[Observe that y and $h(z)$ are independent]

find best pt

- Underfitting = too much bias high bias 欠拟合
- Most overfitting caused by too much variance add data ↓ variance
- Training error reflects bias but not much variance; test error reflects both
- For many distributions, variance->0 as n->∞ (训练样本增多)
- If h can fit g exactly, for many distributions bias->0 as n->∞ (h要有足够的复杂度)
- If h cannot fit g well, bias is large at "most" points
- Adding a good feature reduce bias; adding a bad feature rarely increase it
- Adding a feature increases variance never increase training error
- Noise in test set affects only Var(ε); noise in training set affects only bias & Var(h)



eg: Least square linear reg

$g(z) = v^T \cdot z$

training label: $y = X \cdot v + e$, $e \in N(0, \sigma^2)$

$\Rightarrow w' = X^+ y = X^+(Xv + e) = v + X^+ e$. noise in $y \Rightarrow$ noise in w

$$\text{BIAS is } E[h(z)] - g(z) = E[w^T z] - v^T z = z^T E[w - v] = z^T E[X^+ e] = z^T E[X^+] E[e] = 0$$

bias can be zero when h on fit g

$$\begin{aligned} \text{VARIANCE is } \mathrm{Var}(h(z)) &= \mathrm{Var}(w^T z) = \mathrm{Var}(v^T z + (X^+ e)^T z) = \mathrm{Var}(z^T X^+ e) \\ &= \sigma^2 \|z^T X^+\|^2 = \sigma^2 z^T (X^T X)^{-1} X^T X (X^T X)^{-1} z \\ &= \sigma^2 z^T (X^T X)^{-1} z = \sigma^2 \cdot \frac{d}{n} \end{aligned}$$

k-mean. fix y. update μ

$$\frac{\partial J}{\partial \mu_i} = \sum_{y_j = i}(\mu_i - X_j).$$

$$\sum_{y_j = i} \mu_i = \sum_{y_j = i} X_j;$$

$$n_j \mu_i = \sum_{y_j = i} X_j;$$

$$\mu_i = \frac{1}{n_j}\sum_{y_j = i} X_j$$

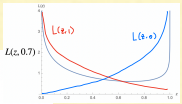
Logistic reg

input 0/1. or prob ∈ [0,1]

Find w that minimizes 各自 α. 独立

$$J = \sum_{i=1}^{n} \underbrace{L(s(X_i \cdot w), y_i)}_{\text{convex}} = -\sum_{i=1}^{n}\left(y_i \ln s(X_i \cdot w) + (1 - y_i)\ln(1 - s(X_i \cdot w))\right)$$

$$\begin{aligned} s'(\gamma) &= \frac{d}{d\gamma}\frac{1}{1 + e^{-\gamma}} = \frac{e^{-\gamma}}{(1 + e^{-\gamma})^2} \\ &= s(\gamma)(1 - s(\gamma)) \end{aligned}$$



Let $s_i = s(X_i \cdot w)$

$$\begin{aligned} \nabla_w J &= -\sum\left(\frac{y_i}{s_i}\nabla s_i - \frac{1 - y_i}{1 - s_i}\nabla s_i\right) \\ &= -\sum\left(\frac{y_i}{s_i} - \frac{1 - y_i}{1 - s_i}\right) s_i(1 - s_i) X_i \\ &= -\sum (y_i - s_i) X_i \\ &= -X^T(y - s(Xw)) \qquad \text{where } s(Xw) = \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_n \end{bmatrix} \end{aligned}$$

Gradient descent rule: $w \leftarrow w + \epsilon X^T(y - s(Xw))$

Stochastic gradient descent: $w \leftarrow w + \epsilon(y_i - s(X_i \cdot w)) X_i$ slow

linearly seperable $\Rightarrow X_i \cdot w \to \infty$. $s(X_i \cdot w) \to 1$. $J(w) \to 0$

$X_i \cdot w \to 0$. $s(X_i \cdot w) \to 0$ $w \to \infty$

Newton's method

smooth. find nearest critical pt

can tell max/min/saddle

Taylor series about v:

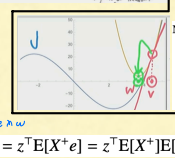
$$\nabla J(w) = \nabla J(v) + (\nabla^2 J(v))(w - v) + O(\|w - v\|^2)$$

where $\nabla^2 J(v)$ is the Hessian matrix of J at v.

Find critical pt w by setting $\nabla J(w) = 0$:

$$w = v - (\nabla^2 J(v))^{-1} \nabla J(v)$$

converge one-step for quadratic strict convex smooth



Newton's method:

pick starting point w

repeat until convergence

$e \leftarrow$ solution to linear system $(\nabla^2 J(w))\, e = -\nabla J(w)$

$w \leftarrow w + e$

优势. 选择合适 step size

选择 steepest 更好的方向.

在 quadratic 处收敛快

logistic reg with Newton

Recall: $s'(\gamma) = s(\gamma)(1 - s(\gamma))$, $\quad s_i = s(X_i \cdot w)$, $\quad s = \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_n \end{bmatrix}$

$$\nabla_w J(w) = -\sum_{i=1}^{n}(y_i - s_i) X_i = -X^T(y - s)$$

Hessian

$$\nabla_w^2 J(w) = \sum_{i=1}^{n} s_i(1 - s_i) X_i X_i^T = X^T \Omega X \quad \text{where } \Omega = \begin{bmatrix} s_1(1 - s_1) & 0 & \dots & 0 \\ 0 & s_2(1 - s_2) & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \dots & s_n(1 - s_n) \end{bmatrix}$$

$s \in [0,1]$

Ω pos-def

$X^T \Omega X$ pos-semidef

J convex

Newton's method:

$w \leftarrow 0$

repeat until convergence

$e \leftarrow$ solution to normal equations $(X^T \Omega X)\, e = X^T(y - s)$

$w \leftarrow w + e$

Ω, s are fn of w

Statistical justification

$y_i \in [0,1]$

label y_i, predict $h(X_i)$. 的 y_i 的 class C. $(1 - y_i)$ 的非 class C

Likelihood is $\mathcal{L}(h; X, y) = \prod_{i=1}^{n} h(X_i)^{y_i\beta}(1 - h(X_i))^{(1-y_i)\beta}$ fixed order

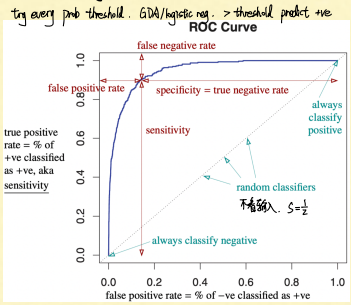
Log likelihood is $\ell(h) = \ln \mathcal{L}(h)$

$$= \beta\sum_i \left(y_i \ln h(X_i) + (1 - y_i)\ln(1 - h(X_i))\right)$$

$$\max = -\beta \sum \underbrace{\text{logistic loss fn } L(h(X_i), y_i)}_{\min}$$

ROC curve (test)

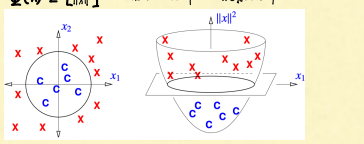
try every prob threshold. GDA/logistic reg. > threshold predict +ve



new feature

$$\Phi(x) = \begin{bmatrix} x \\ \|x\|^2 \end{bmatrix}$$

linear in Φ-space 投影回 x-space



$$\Phi(x) = [x_1^2 \quad x_2^2 \quad x_3^2 \quad x_1x_2 \quad x_2x_3 \quad x_3x_1 \quad x_1 \quad x_2 \quad x_3]^T$$

多项 $O(d^2)$. 部分, $O(d)$ $\Rightarrow$ axis-aligned 椭圆/双曲面

Decision function is $\underbrace{[A \quad B \quad C \quad D \quad E \quad F \quad G \quad H \quad I]}_{w^T} \cdot \Phi(x) + \alpha$

$$\Phi(x) = [x_1^3 \quad x_1^2x_2 \quad x_1x_2^2 \quad x_2^3 \quad x_1^2 \quad x_1x_2 \quad x_2^2 \quad x_1 \quad x_2]^T$$

$$\Phi(x) : \mathbb{R}^d \to \mathbb{R}^{O(d^p)}$$

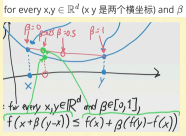
4 level: application. model. optim problem. optim algo

3 optim problem

① unconstrained. ∇f continuous $\Rightarrow$ smooth

convex $\Rightarrow$ for every x, y ∈ $\mathbb{R}^d$, the line segment connecting (x, f(x)) to (y, f(y)) does not go below f(·).

for every x,y ∈ $\mathbb{R}^d$ (x y 是两个横坐标) and $\beta \in [0,1]$, $f(x + \beta(y - x)) \leq f(x) + \beta(f(y) - f(x))$



: for every x,y ∈ $\mathbb{R}^d$ and $\beta \in [0,1]$,

$f(x + \beta(y - x)) \leq f(x) + \beta(f(y) - f(x))$

perceptron $\Rightarrow$ convex. not smooth

convex $\Rightarrow$ no min / one min / a set of min

Hessian ill-conditioned. 椭圆

$\Rightarrow$ largest eigenvalue >> smallest

Least-square poly reg

$$\Phi(X_i) = [X_{i1}^2 \quad X_{i1}X_{i2} \quad X_{i2}^2 \quad X_{i1} \quad X_{i2} \quad 1]^T$$

Weighted Least-square

Assign each sample pt a weight ω_i; collect ω_i's in $n \times n$ diagonal matrix Ω.

Greater $\omega_i \to$ work harder to minimize $(\hat{y}_i - y_i)^2$ recall: $\hat{y} = Xw$ [$\hat{y}_i$ is predicted label for X_i]

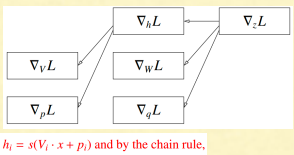
Find w that minimizes $(Xw - y)^T \Omega (Xw - y) = \sum_{i=1}^{n} \omega_i (X_i \cdot w - y_i)^2$

$\nabla J = 0 \Rightarrow$ normal eq: $X^T \Omega X w = X^T \Omega y$

auto encoder

$h = s(Vx + p)$. $z = s(Wh + q)$. $L = \|z - x\|_2^4$

$v = \|z - x\|^2$. $L = v^2$. $\nabla_z L = 2v \cdot 2(z - x) = 4\|z - x\|^2 (z - x)$



$$\frac{\partial z_j}{\partial h_i} = s'(W_j \cdot h + q_j)\frac{\partial}{\partial h_i}(W_j \cdot h + q_j) = z_j(1 - z_j)W_{ij}, \quad \text{row } j$$

$$\frac{\partial L}{\partial x_k} = \frac{dL}{dx_k} + \sum_{i=1}^{m}\frac{\partial L}{\partial h_i}\frac{\partial h_i}{\partial x_k}.$$

which could alternatively be written $\frac{dL}{dx_k} + (\nabla_h L) \cdot \frac{\partial h}{\partial x_k}$. Recall that $h = s(Vx + p)$, hence $h_i = s(V_i \cdot x + p_i)$ and by the chain rule,

$$\frac{\partial h_i}{\partial x_k} = s'(V_i \cdot x + p_i)\frac{\partial}{\partial x_k}(V_i \cdot x + p_i) = h_i(1 - h_i)V_{ik},$$

Therefore,

$$\frac{\partial L}{\partial x_k} = 4\|x - z\|^2(x_k - z_k) + \sum_{i=1}^{m}\frac{\partial L}{\partial h_i} h_i(1 - h_i)V_{ik}.$$

kernel PCA

$\Phi: d \to D$

covariance matrix

$C = \frac{1}{n}\Phi(x)^T \cdot \Phi(x)$

$k(X_i, X_j) = \Phi(X_i)^T \cdot \Phi(X_j)$

① eigenvector v. $Cv = \lambda v$

$\lambda v = \frac{1}{n}\Phi(x)^T \Phi(x)\, v$

$v = \Phi(x)^T \cdot \frac{1}{\lambda n} \cdot \Phi(x) \cdot v$

$v = \Phi(x)^T \cdot a$

② $K = \Phi(x)\Phi(x)^T$.

prove $K^2 a = \lambda K a$

$Cv = \lambda v$

$\frac{1}{n}\Phi(x)^T \cdot \Phi(x)\, \Phi(x)^T \cdot a = \lambda \Phi(x)^T \cdot a$

左乘 $\Phi(x)$ $K^2 a = \lambda n K a$

$\Rightarrow$ 化 ω

③ a is eigenvector of K with eigenvalue Ka

$\Rightarrow v = \Phi(x)^T a$ is eigenvector of C with eigenvalue $\frac{\lambda a}{n}$

$\lambda a \cdot a = K \cdot a = \Phi(x)\Phi(x)^T \cdot a$

$Cv = \frac{1}{n}\Phi(x)^T \cdot \Phi(x)\, \Phi(x)^T \cdot a$

$= \frac{1}{n}\Phi(x)^T \cdot K \cdot a$

$= \frac{1}{n}\Phi(x)^T \cdot \lambda a \cdot a$

$= \frac{\lambda a}{n} \cdot v$

④ test. $v = \sum_{j=1}^{n} a_j \Phi(x_j)$

principal coordinate: $\Phi(z)^T \cdot v$

$\Phi(z)^T \cdot \sum_{j=1}^{n} a_j \Phi(x_j) = \sum_{j=1}^{n} a_j K(z, x_j)$

梯度

$$\begin{aligned} \nabla_W J(W) &= \sum_{i=1}^{n} \nabla_W (X_i^T W X_i - y_i)^2 \\ &= \sum_{i=1}^{n} 2(X_i^T W X_i - y_i) \cdot \nabla_W (X_i^T W X_i - y_i) \\ &= 2\sum_{i=1}^{n}(X_i^T W X_i - y_i) X_i X_i^T. \end{aligned}$$

adaboost

initial $w_{i=8} = \frac{1}{8}$.

$err = \frac{1}{8}$ $\quad \sqrt{\frac{1 - err}{err}} = \sqrt{7}$

$w_{i=7} = \frac{1}{8\sqrt{7}}$ $\quad w_k = \frac{\sqrt{7}}{8}$

likelihood

$X_1 \cdots X_n \sim N(\mu, \Sigma)$. d features. $\quad$ 1x1 $\quad$ dxd

$$\mathcal{L}(\mu, \Sigma) = \frac{1}{(2\pi)^{nd/2}|\Sigma|^{n/2}}\prod_{i=1}^{n}\exp\left(-\frac{1}{2}(X_i - \mu)^T \Sigma^{-1}(X_i - \mu)\right)$$

$$\ell(\mu, \Sigma) = \ln \mathcal{L}(\mu, \Sigma) = -\ln\left((2\pi)^{nd/2}|\Sigma|^{n/2}\right) - \frac{1}{2}\sum_{i=1}^{n}(X_i - \mu)^T \Sigma^{-1}(X_i - \mu).$$

$\nabla_\mu l = 0 \Rightarrow \Sigma^{-1}\left(\sum_{i=1}^{n} X_i - n\mu\right) = 0.$

左乘 Σ $\quad \hat{\mu} = \frac{1}{n}\sum X_i$

(b) [4 pts] Suppose that X has rank n. (That is, the training points are linearly independent and $n \leq d$.) Let I_i be the **row vector** that is row i of the $n \times n$ identity matrix; hence, its entries are zero except a 1 at position i. Observe that we can write training point i as the row vector $X_i^T = I_i X$. Let X^+ be the Moore–Penrose pseudoinverse of X. As a prelude to part (c), show that $X_i^T X^+ = I_i$. (*Hint: What do you know about pseudoinverses?*)

By postmultiplying both sides of $X_i^T = I_i X$ by X^+, we have $X_i^T X^+ = I_i X X^+ = I_i$. (As X is an $n \times d$ matrix with rank n, $XX^+ = I$.)

(c) [5 pts] Again suppose that X has rank n. Write the labels as a diagonal $n \times n$ matrix

$$Y = \mathrm{diag}(y_i) = \begin{bmatrix} y_1 & 0 & \dots & 0 \\ 0 & y_2 & & 0 \\ \vdots & & \ddots & \\ 0 & \dots & & y_n \end{bmatrix}.$$

Using (b), show that the cost function $J(W)$ is minimized by $W^* = X^+ Y X^{+T}$, where X^{+T} denotes the transpose of X^+.

For each $i \in [1, n]$,

$$X_i^T W^* X_i = X_i^T X^+ Y X^{+T} X_i = I_i Y I_i^T = y_i.$$

Therefore, $J(W^*) = \sum_{i=1}^{n}(X_i^T W^* X_i - y_i)^2 = 0$. As $J(W)$ is nonnegative for all W, no matrix has a lower cost than W^*, so W^* is optimal.

(d) [5 pts] Suppose the ground truth for the labels is $g(z) = z^T A z$, where $A \in \mathbb{R}^{d \times d}$ is a fixed, symmetric matrix. Each training label is drawn from this ground truth with random Gaussian noise; that is, $y_i = g(X_i) + \epsilon_i$ where $\epsilon_i \sim N(0, 1)$. Assume that the regression function we learn is $h(z) = z^T W^* z$ where W^* is given in part (c). Show that the **bias** of this regression method at a test point z is

$$\mathrm{bias}(z) = |z^T(X^+ \mathrm{diag}(X_i^T A X_i) X^{+T} - A)z|.$$

where $\mathrm{diag}(X_i^T A X_i)$ is a diagonal $n \times n$ matrix whose diagonal values are $X_i^T A X_i$.

$$\mathrm{bias}(z) = |E[h(z)] - g(z)| = |E[z^T X^+ Y X^{+T} z] - z^T A z| = |z^T(X^+ E[Y] X^{+T} - A)z|.$$

$E[Y]$ is a diagonal matrix whose diagonal values are $E[y_i] = E[g(X_i) + \epsilon_i] = g(X_i) = X_i^T A X_i$, so

$$\mathrm{bias}(z) = |z^T(X^+ \mathrm{diag}(X_i^T A X_i) X^{+T} - A)z|.$$

Adaboost 0-1 loss

$$w_i^{(t+1)} = \omega\, L(\sum_{k=1}^{t}\beta_k G_k(X_i), y_i)$$

Begin from $w_i^{(t+1)} = \frac{w_i^{(t)}}{Z_t} e^{-\beta_t y_i G_t(X_i)}$. Apply recursion: $w_i^{(t+1)} = \frac{w_i^{(t)}}{Z_t} e^{-\beta_t y_i G_t(X_i)} = \frac{w_{i-1}(i)}{Z_t Z_{t-1}} e^{-y_i \beta_t G_t(X_i)} e^{-y_i \beta_{t-1} G_{t-1}(X_i)}$. Complete recursion: $w_i^{(t+1)} = (n\prod_{k=1}^{t} Z_k)^{-1} e^{-y_i \sum_{k=1}^{t}\beta_k G_k(X_i)}$ after using $w_i^{(1)} = 1/n$. Therefore $\omega = (n\prod_{k=1}^{t} Z_k)^{-1}$.

$$R(\sum_{k=1}^{t}\beta_k G_k) = \prod_{k=1}^{t} Z_k$$

Substitute into risk expression $R(\sum_{k=1}^{t}\beta_k G_k) = \frac{1}{n}\sum_{i=1}^{n} e^{-y_i \sum_{k=1}^{t}\beta_k G_k} = \frac{1}{n}\sum_{i=1}^{n}(n\prod_{k=1}^{t} Z_k) w_i^{(t+1)} = (n\prod_{k=1}^{t} Z_k)\frac{1}{n}\sum_{i=1}^{n} w_i^{(t+1)} = \prod_{k=1}^{t} Z_k$ where in the last step $\sum_i w_i^{(t+1)} = 1$ is used.

$$R(\sum_{k=1}^{t}\beta_k G_k) = \prod_{k=1}^{t} 2\sqrt{\mathrm{err}_k(1 - \mathrm{err}_k)}.$$

$Z_t = \sum_{i=1}^{n} w_i^{(t)} e^{-\beta_t y_i G_t(X_i)} = \sum_{i: y_i = G_t(X_i)} w_i^{(t)} e^{-\beta_t} + \sum_{i: y_i \neq G_t(X_i)} w_i^{(t)} e^{\beta_t} = (1 - \mathrm{err}_t)e^{-\beta_t} + \mathrm{err}_t e^{\beta_t} = (1 - \mathrm{err}_t)\sqrt{\frac{\mathrm{err}_t}{1 - \mathrm{err}_t}} + \mathrm{err}_t\sqrt{\frac{1 - \mathrm{err}_t}{\mathrm{err}_t}} = 2\sqrt{\mathrm{err}_t(1 - \mathrm{err}_t)}$. Thus from (b) we have $R(\sum_{k=1}^{t}\beta_k G_k) = \prod_{k=1}^{t} Z_k = \prod_{k=1}^{t} 2\sqrt{\mathrm{err}_k(1 - \mathrm{err}_k)}$.

0-1 loss function $L_{01}(\rho, \ell) = \mathbf{1}[\rho \neq \ell]$

0-1 risk $R_{01}(f) = \frac{1}{n}\sum_{i=1}^{n} L_{01}(f(X_i), y_i)$ is less than or equal to the *exponential risk* $R(f)$ for any classifier $f(\cdot)$.

See that $x \leq 0 \implies e^{-x} \geq 1$ and $x > 0 \implies e^{-x} > 0$, thus $\mathbf{1}[x \leq 0] \leq e^{-x}$. Use this on $R_{01}(f) = \frac{1}{n}\sum_{i=1}^{n}\mathbf{1}(f(X_i) \neq y_i) = \frac{1}{n}\sum_{i=1}^{n}\mathbf{1}(y_i f(X_i) \leq 0) \leq \frac{1}{n}\sum_{i=1}^{n} e^{-y_i f(X_i)} = R(f)$.

square loss

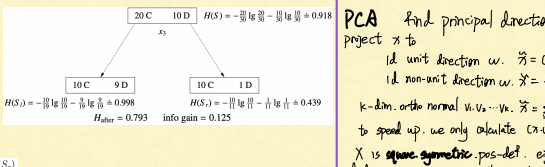
empirical Risk $\frac{1}{n}\sum_{i=1}^{n} L(M_T(X_i), y_i)$. risk $= \frac{1}{n}\sum_{i=1}^{n}\left(\sum_{i=1}^{T}\beta_i G_i(X_i) - y_i\right)^2$ choose β_T 有 Σ 不可约分

$$\beta_T = \frac{\sum_{i=1}^{n} G_T(X_i)(y_i - M_{T-1}(X_i))}{\sum_{i=1}^{n} G_T^2(X_i)}$$

decision tree high variance

GrowTree(S) leaf node
if $y_i = C$ for all $i \in S$ and some class C) then {
  return new leaf(C) [We say the leaves are pure]
} else { internal node
  choose best splitting feature j and splitting value β (*)
  $S_l = \{i \in S : X_{ij} < \beta\}$ [Or you could use ≤ and >]
  $S_r = \{i \in S : X_{ij} \geq \beta\}$
  return new node(j, β, GrowTree(S_l), GrowTree(S_r))
}

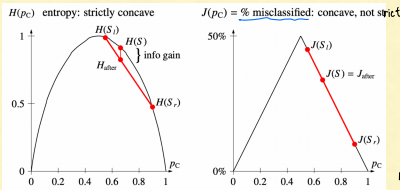
subset



choose split minimizes weighted average $H_{after} = \frac{|S_l|H(S_l) + |S_r|H(S_r)}{|S_l| + |S_r|}$
maximize 的 gain = $H(S) - H_{after} \geq 0$

$P(Y=c) = p_c$. surprise $= -\log_2 p_c > 0$. entropy $H(S) = -\sum_c p_c \log_2 p_c$
的 gain = 0 { one child empty
  ≥0 10D → 10c 5D + 10c 5D

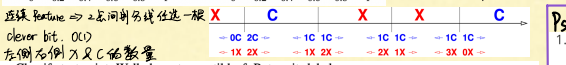
noise/overlap distribution ⇒ pure leaves overfit
return posterior prob instead of majority class
stop early 略



for regression
label $\mu_S = \frac{1}{|S|}\sum_{i \in S} y_i$
Cost $J(S) = \frac{1}{|S|}\sum_{i \in S}(y_i - \mu_S)^2$

Pruning: greedily remove split who improve valid
multivariate split 略
multivariate split 略

选择 feature ⇒ 2点间的分线性选一般
clever bit. O(1)
左(右)为例 X 是 C 的数量



– Classify test point: Walk down tree until leaf. Return its label.
  Worst-case time is O(tree depth).
  For binary features, that's $\leq d$. [Quantitative features may go deeper.]
  Usually (not always) $\leq O(\log n)$.
– Training: For binary features, try $O(d)$ splits at each node.
  For quantitative features, try $O(n'd)$ splits; n' = points in node
  Either way ⇒ $O(n'd)$ time at this node
  [Quantitative features are asymptotically just as fast as binary features because of our clever way of computing the entropy for each split.]
  Each point participates in O(depth) nodes, costs $O(d)$ time in each node.
  [This is an amortized analysis: we are charging $O(d$ depth) time to each sample point.]
  Running time $\leq O(nd$ depth).
  [As nd is the size of the design matrix X, and the depth is often logarithmic, this is a surprisingly reasonable running time.]

PCA

find principal direction/feature with most info (reduce variance)
project x to
  1d unit direction w. $\tilde{x} = (x \cdot w) \cdot w$
  1d non-unit direction w. $\tilde{x} = \frac{x \cdot w}{\|w\|^2} \cdot w$
k-dim. ortho normal $v_1, v_2 \cdots v_k$. $\tilde{x} = \sum_{i=1}^{k}(x \cdot v_i) \cdot v_i$
to speed up. we only calculate (x·vi)

% of variability $= \frac{\sum_{i=d-k+1}^{d}\lambda_i}{\sum_{i=1}^{d}\lambda_i}$

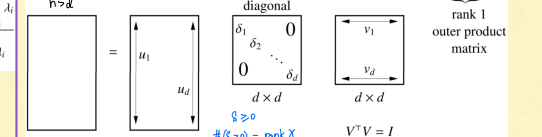
X is **square-symmetric**. pos-def. eigenvalue $0 \leq \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_d$
d×d ortho normal eigenvector v_1 v_2 v_d

PCA alg:
- must Center X (mean = 0)
- Optional: Normalize X
  Units of measurement different, Normalize; same: don't
- Compute unit eigenvectors/values of $X^\top X$
- Choose k principal direction with largest eigenvalue
- Compute the k principal coordinates x · vi for each training/test point

SVD non-square. Non-symmetric

get k huge singular value in $O(nd^2)$ [or $O(nd \log k)$]
compute $X^\top X$ take $O(nd^2)$. max eigenvalue / min eigenvalue $X \Rightarrow$ eigenvector inaccurate/slow

$X = U D V^\top$, X: n×d, U: n×d, D: diagonal, $V^\top$: d×d



$= \sum_{i=1}^{d}\delta_i u_i v_i^\top$ rank 1 outer product matrix

$\delta \geq 0$
$\#(\delta > 0) = \text{rank } X$

$n \times d$ $n \times d$ $d \times d$ $d \times d$

$U^\top U = I$
orthonormal u_i's are left singular vectors of X

$V^\top V = I$
orthonormal v_i's are right singular vectors of X

Fact: vi (right sigular vector) is an eigenvector of $X^\top X$ w/eigenvalue δ_i^2 max singular value / min singular value → ⇒ stable

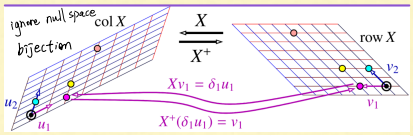
Proof: $X^\top X = V D U^\top U D V^\top = V D^2 V^\top$

Find w that maximizes $\text{Var}(\tilde{X}_1, \tilde{X}_2, \ldots, \tilde{X}_n) = \frac{1}{n}\sum_{i=1}^{n}\left(X_i \cdot \frac{w}{\|w\|}\right)^2 = \frac{1}{n}\frac{\|Xw\|^2}{\|w\|^2} = \frac{1}{n}\frac{w^\top X^\top X w}{w^\top w}$ variance ≥ 0, $\leq \sigma^2_{\max}$

w = eigenvector v_i. Rayleigh quotient of $X^\top X$ and w λ_i

Find w that minimizes $\sum_{i=1}^{n}\|X_i - \tilde{X}_i\|^2 = \sum_{i=1}^{n}\left\|X_i - \frac{X_i \cdot w}{\|w\|^2} w\right\|^2 = \sum_{i=1}^{n}\left(\|X_i\|^2 - \left(X_i \cdot \frac{w}{\|w\|}\right)^2\right)$ mean squared projection
$= \text{constant} - n(\text{variance from derivation 2}).$

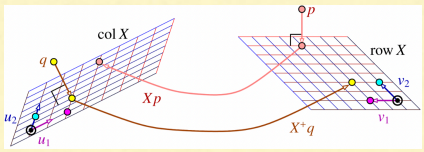
Row i of UD give principal coordinate of sample pt Xi
$[\forall i, \forall j, X_i \cdot v_j = \delta_j U_{ij}]$
[Proof: $XV = UDV^\top V = UD$, so$(XV)_{ij} = (UD)_{ij}$]



r = rank X = dim row space = dim col space, $Xv_i = \delta_i u_i$ $X^+ u_i = \frac{1}{\delta_i} v_i$

- The r right singular vectors vi with $\delta_i \neq 0$ are an orthonormal basis for row X (图中v1, v2)
- The r left singular vectors ui with $\delta_i \neq 0$ are an orthonormal basis for col X (图中u1, u2)

X is a 4*3 matrix, rank = 2
X first project pt in R^3 to the R^2 row X, then map pt in row X to pt in col X
X^+ first project pt in R^4 to the R^2 col X, then map pt in col X to pt in row X



Pseudoinverse

1. psuedoinverse of diagonal matrix
   pseudoinverse D^+: transpose D and replace every nonzero entry with reciprocal if very small. treat as 0

$$D = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1/3 \\ 0 & 0 & 0 \end{pmatrix}, \quad D^+ = \begin{pmatrix} 1/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 \end{pmatrix}, \quad DD^+ = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad D^+D = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$DD^+D = D$ and $D^+DD^+ = D^+$ If D is square, $D^2D^+ = D$

If D is square & no zero on diagonal, D+ is the inverse of D, $DD^+ = D^+D = I$

2. pseudoinverse of a matrix in general

$X = UDV^\top$, rank D = rank X, Moore–Penrose pseudoinverse: $X^+ = VD^+U^\top$

(1) $XX^+ = UDV^\top VD^+U^\top = U(DD^+)U^\top$ is symmetric & positive semidefinite.
  [Observe that this is an eigendecomposition of XX^+, and all the eigenvalues are 1 or 0.]
(2) $X^+X = VD^+U^\top UDV^\top = V(D^+D)V^\top$ is symmetric & PSD too.
(3) All have the same rank: $D, D^+, DD^+, D^+D, X, X^+, XX^+, X^+X$.
(4) If X has rank n, then $XX^+ = I_{n\times n}$ and X^+ is a right inverse.
(5) If X has rank d, then $X^+X = I_{d\times d}$ and X^+ is a left inverse.
(6) $XX^+X = X$. Proof: $XX^+X = U(DD^+)U^\top UDV^\top = U(DD^+D)V^\top = UDV^\top = X$.
(7) $X^+XX^+ = X^+$. [The proof is symmetric to the previous one.]

for linear regression
X^+y is least-norm sol (best)
minimize $\|w\|$

Theorem: A solution to the normal equations $X^\top Xw = X^\top y$ is $w = X^+y$.
Proof: $X^\top Xw = X^\top XX^+y = VDU^\top U(DD^+)U^\top y = V(D^2D^+)U^\top y = VDU^\top y = X^\top y$

ensemble learning ↓ variance

use low bias son-learner

① bagging. random subsample n' from n with replacement
  choose a pt j times (n'≈n. 63.2% are used)
  Decision trees: j-time point has $j \times$ weight in entropy.
  SVMs: j-time point incurs $j \times$ penalty to violate margin.
  Regression: j-time point incurs $j \times$ loss.

② random forest. randomized tree + bagging
  At each treenode, take random sample of m features (out of d).
  Choose best split from m features.
  [We're not allowed to split on the other $d - m$ features!]
  Different random sample for each treenode.
  $m \approx \sqrt{d}$ works well for classification; $m \approx d/3$ for regression.
  m↓. more decorrelate ⇒ less variance after ensemble
  more bias

Suppose we train n *independent* models on the same dataset. Given some test point, z we let X_i be the prediction of the ith model for this point, and $F(z)$ be the average of these n iid predictions. We thus have

$$\mathbb{E}[F(z)] = \mathbb{E}\frac{1}{n}\sum_{i=1}^{n} X_i = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[X_i] = \mathbb{E}[X_1] \quad \text{same bias}$$

$$Var[F(z)] = Var\frac{1}{n}\sum_{i=1}^{n} X_i = \frac{1}{n^2}\sum_{i=1}^{n} Var[X_i] = \frac{Var[X_1]}{n} \quad \text{variance}\downarrow$$

high dim

Vector almost same length. orthogonal
In high dim, 1NN and 1000NN don't differ much
k-means and kNN are less effective for large d

kNN Nearest neighbor

Given query point q, find the k training pts nearest q
- Regression: Return average label train time : 0
- Classification: Return class with the most votes OR return histogram of class prob

large k -> underfit; small k -> overfit Extnustive test time: O(nd + n·log k)
data get denser, best k increase random order expected: O(nd + k·log n·log k) (slow) but more cache miss

train & test pt IID

Theorem (Cover & Hart, 1967):
As $n \to \infty$, the 1-NN error rate is $< 2B - B^2$ where B = Bayes risk
  if only 2 classes, $\leq 2B - 2B^2$

Theorem (Fix & Hodges, 1951):
As $n \to \infty$, $k \to \infty$, $k/n \to 0$, k-NN error rate converges to B.

random projection

PCA preserve variance · random prj preserve distance
reduce dim, then for cluster/classify/regression kNN. k-means

Pick a small ϵ, a small δ, and a random subspace $S \subset \mathbb{R}^d$ of dimension k, where $k = \left\lceil \frac{2\ln(1/\delta)}{\epsilon^2/2 - \epsilon^3/3} \right\rceil$ ceiling

For any pt q, let $\hat{q}$ be orthogonal projection of q onto S, multiplied by $\sqrt{\frac{d}{k}}$. compensate for throw-away dim

Johnson–Lindenstrauss Lemma (modified): input dim doesn't matter
For any two pts $q, w \in \mathbb{R}^d$, $(1-\epsilon)\|q - w\|^2 \leq \|\hat{q} - \hat{w}\|^2 \leq (1+\epsilon)\|q - w\|^2$ with probability $\geq 1 - 2\delta$.
Typical values: $\epsilon \in [0.02, 0.5]$, $\delta \in [1/n^3, 0.05]$. [You choose ϵ and δ according to your needs.]

Adaboost ↓ bias

use with short trees (underfit)
more weight on misclassified pt & accurate learner

Train T classifiers G1,...,GT iteration 1 iteration T
For test point z, $M(z) = \sum_{t=1}^{T}\beta_t G_t(z)$ (better for non-linear model)
Each Gt is ±1, Return sign of M(z) X : n×d

Find G_T & β_T that minimize in iteration T. $G_1 \cdots G_{T-1}$ are already trained

$$\text{Risk} = \frac{1}{n}\sum_{i=1}^{n} L(M(X_i), y_i), \qquad M(X_i) = \sum_{t=1}^{T}\beta_t G_t(X_i)$$

just loss for metalearner not for son-learner G_t

$$L(\rho, \ell) = \begin{cases} e^{-\rho} & \ell = +1 \\ e^{\rho} & \ell = -1 \end{cases}$$

not robust to outlier

Important: label ℓ is binary, G_t is binary, but $\rho = M(X_i)$ is continuous!

① G_T

$$n \cdot \text{Risk} = \sum_{i=1}^{n} L(M(X_i), y_i) = \sum_{i=1}^{n} e^{-y_i M(X_i)}$$

$$= \sum_{i=1}^{n}\exp\left(-y_i\sum_{t=1}^{T}\beta_t G_t(X_i)\right) = \sum_{i=1}^{n}\prod_{t=1}^{T} e^{-\beta_t y_i G_t(X_i)} \quad \Leftarrow \quad y_i G_t(X_i) = \pm 1,\ -1 \to G_t \text{ misclassifies } X_i$$

$$= \sum_{i=1}^{n} w_i^{(T)} e^{-\beta_T \overset{\pm 1}{y_i G_T(X_i)}}, \quad \text{where } w_i^{(T)} = \prod_{t=1}^{T-1} e^{-\beta_t y_i G_t(X_i)}$$

1~T-1 已经确定

$$= e^{-\beta_T}\sum_{y_i = G_T(X_i)} w_i^{(T)} + e^{\beta_T}\sum_{y_i \neq G_T(X_i)} w_i^{(T)} \quad \text{[correctly classified and misclassified]}$$

$$= \cancel{e^{-\beta_T}\sum_{i=1}^{n} w_i^{(T)}} + (e^{\beta_T} - \cancel{e^{-\beta_T}})\sum_{y_i \neq G_T(X_i)} w_i^{(T)}.$$

all pts >0 misclassified pts (optimize)

GT that minimize the sum of $w_i^{(T)}$ over all misclassified pts Xi -> minimize the risk

speed up weight for each train pt X_i

$$w_i^{(T+1)} = w_i^{(T)} e^{-\beta_T y_i G_T(X_i)} = \begin{cases} w_i^{(T)} e^{-\beta_T} & y_i = G_T(X_i), \\ w_i^{(T)} e^{\beta_T} & y_i \neq G_T(X_i). \end{cases}$$

② β_T

To choose β_T, set $\frac{d}{d\beta_T}\text{Risk} = 0$.

$$0 = -e^{-\beta_T}\sum_{i=1}^{n} w_i^{(T)} + (e^{\beta_T} \triangle e^{-\beta_T})\sum_{y_i \neq G_T(X_i)} w_i^{(T)}; \quad \text{[now divide both sides by the first term]}$$

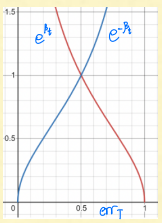
$$0 = -1 + (e^{2\beta_T} + 1)\,\text{err}_T, \quad \text{where } \text{err}_T = \frac{\sum_{y_i \neq G_T(X_i)} w_i^{(T)}}{\sum_{i=1}^{n} w_i^{(T)}}; \quad \Leftarrow G_T\text{'s weighted error rate}$$

$$\beta_T = \frac{1}{2}\ln\left(\frac{1 - \text{err}_T}{\text{err}_T}\right).$$

T=∞ $\text{err}_T = 0$
T=0 $\text{err}_T \doteq \frac{1}{2}$

AdaBoost alg:
1. Initialize weights $w_i \leftarrow \frac{1}{n}, \forall i \in [1, n]$.
2. for $t \leftarrow 1$ to T
   a. Train G_t with weights w_i.
   b. Compute weighted error rate err $\leftarrow \frac{\sum_{\text{misclassified}} w_i}{\sum_{\text{all}} w_i}$; coefficient $\beta_t \leftarrow \frac{1}{2}\ln\left(\frac{1 - \text{err}}{\text{err}}\right)$
   c. Reweight pts: $w_i \leftarrow w_i \cdot \begin{cases} e^{\beta_t}, & G_t \text{ misclassifies } X_i \\ e^{-\beta_t}, & \text{otherwise} \end{cases} = w_i \cdot \begin{cases} \sqrt{\frac{1-\text{err}}{\text{err}}}, \\ \sqrt{\frac{\text{err}}{1-\text{err}}}. \end{cases}$
3. return metalearner $h(z) = \text{sign}\left(\sum_{t=1}^{T}\beta_t G_t(z)\right)$.

Posterior prob can be approximated: $P(Y = 1|x) \approx 1/(1 + e^{-2M(x)})$
If every learner beats error μ for μ < 50%, metalearner training error will eventually be zero [hw7]



k-means cluster

Goal: Partition n points into k disjoint clusters.
  Assign each input point X_i a cluster label $y_i \in [1, k]$.
  Cluster i's mean is $\mu_i = \frac{1}{n_i}\sum_{y_j = i} X_j$, given n_i points in cluster i.

Find y that minimizes $\sum_{i=1}^{k}\sum_{y_j = i}\|X_j - \mu_i\|^2$ y_1

[Sum of the squared distances from points to their cluster means.]

Equivalent objective fn: the within-cluster variation
Find y that minimizes $\sum_{i=1}^{k}\frac{1}{n_i}\sum_{y_j = i}\sum_{y_m = i}\|X_j - X_m\|^2$ y_1 $y_2 = \mathcal{E}_3 \cdot y_1$
[training pt with index j, m both in cluster i]

k-means heuristic:
- (1) yj are fixed; update μi = mean of the pts in cluster i O(nd)
- (2) μi are fixed; update yj = closest center μi O(nkd)

[If there's a tie平局, stay in previous cluster]

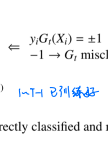
Getting started:
- Forgy method: choose k random sample points to be initial μi; go to (2)
- Random partition: randomly assign each sample point to a cluster; go to (1)
- k-means++: like Forgy, but biased distribution [center far from each other]

For best results, run k-means multiple times with times random starts

k-NN

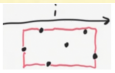
preprocess train pts to obtain sublinear query time only 1-NN
- 2–5 dimensions: Voronoi diagrams (BSP trees)
- Medium dim (up to ~30): k-d trees (easier to implement than Voronoi diagrams)
- Large dim: exhaustive k-NN, but can use PCA or random projection locality sensitive hashing

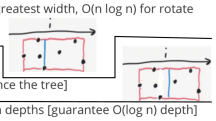
① Voronoi diagram
convex polyhedron/polytope Let X be a point set.
Voronoi cell of $w \in X$ is: $\text{Vor } w = \{p \in \mathbb{R}^d : \|p - w\| \leq \|p - v\| \quad \forall v \in X\}$



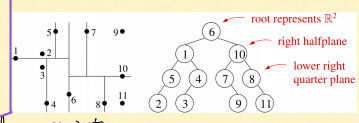
Pt location: Given query pt $q \in \mathbb{R}^d$, find the pt $w \in X$ for which $q \in \text{Vor } w$
- 2D: **trapezoidal map** for pt location
  O(n log n) preprocess: compute V.d. and trapezoidal map
  O(log n) query time
- dD (d>3): **binary space partition tree (BSP tree)** for pt location

② k-d tree
- Choose splitting feature
  - slow: choose greatest width: feature i in $\max_{i,j,k}(X_{ji} - X_{ki})$
  - Fast: rotate through features O(d)
    [2 feature: split on feature1 at depth 1, feature2 at depth 2, feature1 at depth 3...]
- choose splitting value:
  - median point for feature i; O(nd log n) for greatest width, O(n log n) for rotate
    [guarantee short tree: $\lfloor \log_2 n \rfloor$ tree depth]
  - midpoint $\frac{X_{ji} + X_{ki}}{2}$
    [guarantee nicely shaped box, may unbalance the tree]
  - medians at odd depths and centers at even depths [guarantee O(log n) depth]
- Each internal node stores a training pt, Usually the splitting point





speed up. approximate
given query pt q, find a training pt w such that $\|q - w\| \leq (1+\epsilon)\|q - s\|$, where s is the closest training pt.
$\epsilon = 0 \Rightarrow$ exact NN; $\epsilon > 0 \Rightarrow$ approximate NN.



Hierarchical Clustering
- Bottom-up/agglomerative: start with each point a cluster; repeatedly fuse融合 pair [point set]
- Top-down/divisive: start with all pts in one cluster; repeatedly split it [graph]

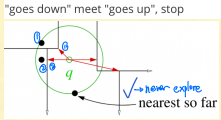
complete linkage: $d(A, B) = \max\{d(w, x) : w \in A, x \in B\}$ best balanced
single linkage: $d(A, B) = \min\{d(w, x) : w \in A, x \in B\}$ very unbalanced.
average linkage: $d(A, B) = \frac{1}{|A||B|}\sum_{w \in A}\sum_{x \in B} d(w, x)$
centroid linkage: $d(A, B) = d(\mu_A, \mu_B)$ where μ_S is mean of S parent cluster may fuse at lower height than children
only for Euclidean distance

Greedy agglomerative alg: Repeatedly fuse the two clusters that minimize d(A, B), $O(n^3)$

Dendrogram: y-axis is linkage distance, leaf node is each pt

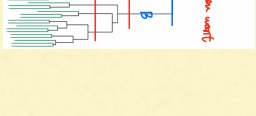
- nearest neighbor found so far: start at ∞ and **goes down**
- unexplored subtree: start from closest box and **goes up**
  [Binary min-heap, keyed by distance
  distance is not q to a train pt, but q to vertice of box]

"goes down" meet "goes up", stop



Alg. for 1-NN query:
  $Q \leftarrow$ heap containing root node with key zero
  $r \leftarrow \infty$
  while Q not empty and $(1+\epsilon) \cdot \text{minkey}(Q) < r$
    $B \leftarrow \text{removemin}(Q)$
    $w \leftarrow B$'s training point
    $r \leftarrow \min\{r, \text{dist}(q, w)\}$
    $B', B'' \leftarrow$ child boxes of B key for B'
    if $(1+\epsilon) \cdot \text{dist}(q, B') < r$ then insert($Q, B', \text{dist}(q, B')$)
    if $(1+\epsilon) \cdot \text{dist}(q, B'') < r$ then insert($Q, B'', \text{dist}(q, B'')$)
  return point that determined r

For k-NN, replace "r" with a max-heap holding the k nearest neighbors

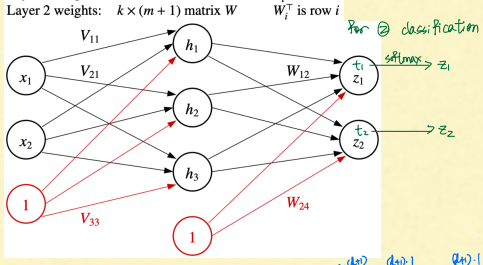


neural network

Input layer: $x_1, \dots, x_d$; $x_{d+1} = 1$
Hidden units: $h_1, \dots, h_m$; $h_{m+1} = 1$
Output layer: $z_1, \dots, z_k$

Layer 1 weights: $m \times (d+1)$ matrix V — $V_i^\top$ is row i
Layer 2 weights: $k \times (m+1)$ matrix W — $W_i^\top$ is row i

For ② classification



For vector v, $s(v) = \begin{bmatrix} s(v_1) \\ s(v_2) \\ \vdots \\ 1 \end{bmatrix}$, $s_1(v) = \begin{bmatrix} s(v_1) \\ s(v_2) \\ \vdots \\ 1 \end{bmatrix}$

$$\overset{(m+1)\cdot 1}{h} = s_1(\overset{m\cdot(d+1)}{V}\overset{(d+1)\cdot 1}{x}) \qquad h_i = s(V_i^\top x)$$
$$\overset{k\cdot 1}{z} = s(\underset{k\cdot(m+1)}{W}\underset{(m+1)}{h}) = s(W s_1(Vx))$$

② regularized classification softmax (sigmoid) at output layer. cross-entropy loss

Let $y \in \mathbb{R}^k$ be vector of labels [for one training point, indicating its membership in the k classes].
In the final layer, compute $t = Wh \in \mathbb{R}^k$ and apply softmax activation to obtain prediction $z \in \mathbb{R}^k$.

Softmax output is $z_i(t) = \frac{e^{t_i}}{\sum_{j=1}^k e^{t_j}}$. Each $z_i \in (0,1)$; $\sum_{i=1}^k z_i = 1$.

[Interpret z_i as an estimate of the posterior probability that the input belongs to class i.]

For k-class softmax output, cross-entropy is $L(z, y) = -\sum_{i=1}^k y_i \ln z_i$.

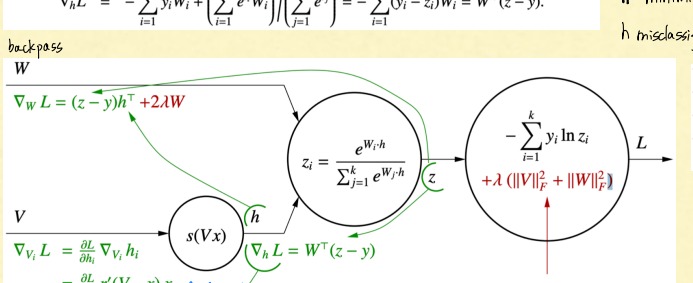
↑ true labels
↑ prediction
} k-vectors

Forward pass
W_i is row of W

Derivatives for k-class softmax backprop: [correct only if $\sum_{i=1}^k y_i = 1$ for each training point]

$$L(z,y) = -\sum_{i=1}^k y_i \ln z_i = -\sum_{i=1}^k y_i\left(t_i - \ln\sum_{j=1}^k e^{t_j}\right) = -\sum_{i=1}^k y_i t_i + \ln\sum_{j=1}^k e^{t_j},$$
$$t_i = \overset{(m+1)\cdot 1}{W_i}\cdot\overset{(m+1)\cdot 1}{h}, \quad \nabla_{W_i} t_i = h, \quad \nabla_h t_i = W_i, \qquad = -\sum_{j=1}^k y_j \cdot w_j \cdot h + \ln\sum_{j=1}^k e^{w_j\cdot h}$$
$$\overset{(m+1)\cdot 1}{\nabla_{W_i} L} = -y_i h + \left(e^{t_i}\Big/\sum_{j=1}^k e^{t_j}\right) h = (z_i - y_i)\, h,$$
$$\overset{k\cdot(m+1)}{\nabla_W L} = \overset{k\cdot 1}{(z-y)}\,\overset{1\cdot(m+1)}{h^\top},$$
$$\overset{(m+1)\cdot 1}{\nabla_h L} = -\sum_{i=1}^k y_i W_i + \left(\sum_{i=1}^k e^{t_i} W_i\right)\Big/\left(\sum_{j=1}^k e^{t_j}\right) = -\sum_{i=1}^k (y_i - z_i)\overset{(m+1)\cdot 1}{W_i} = \overset{(m+1)\cdot k}{W^\top}\overset{k\cdot 1}{(z-y)}.$$

backpass



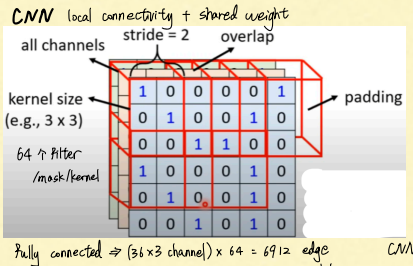
$k=2$. → sigmoid

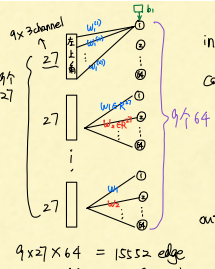
$z_1 = s(t_1 - t_2)$; $\quad z_2 = 1 - z_1 = s(t_2 - t_1)$.

Cross-entropy loss is $L(z, y) = -y_1 \ln z_1 - y_2 \ln z_2 = -y_1 \ln z_1 - (1 - y_1)\ln(1 - z_1)$

CNN

local connectivity + shared weight





(batch, 3, 32, 32)
input (3, 32, 32)
Conv2d (in_channel = 3
out_channel = 64
kernel_size = 3
stride = 2
padding = 1)
output $\boxed{(64, 3, 3)}$
(batch, 64, 3, 3)

Fully connected ⇒ (36×3 channel) × 64 = 6912 edge
6912 weight

CNN 9×27×64 = 15552 edge
27×64 = 1728 weight
#patch #filter

Kernel trick

$\Phi: d \to D$

$w = X^\top a = \sum a_i x_i$ ($\underset{D\times n}{X^\top}\ \underset{n\times 1}{a}$; weight, train pt, linear comb)

w is linear comb of train pt X

ridge reg.
center X, y (keep. $X_{d+1} = 1$)
Find a minimize
$$\|XX^\top a - y\|^2 + \lambda\|X^\top a\|^2$$
$(X^\top X + \lambda I)w = X^\top y$ $D\times D$
Training: Solve $(X^\top X + \lambda I)a = y$ for a.
Testing: Regression fn is $n\times n$
$$h(z) = w^\top z = a^\top X z = \sum_{i=1}^n a_i (X_i^\top z)$$
kernel fn poly kernel
$k(x,z) = x^\top z$ / $(x^\top z+1)^p = \Phi(x)^\top\Phi(z)$
$K = XX^\top$. $K_{ij} = k(X_i, X_j)$

(Dual/kernel ridge regr. alg:
kernel matrix
$\forall i, j$, $K_{ij} \leftarrow k(X_i, X_j)$ $O(n^2 d)$
Solve $(K + \lambda I)a = y$ for a $O(n^3)$
for each test pt z
$h(z) \leftarrow \sum_{i=1}^n a_i k(X_i, z)$ $O(nd)$

logistic reg
Let $\Phi(X)$ be $n \times D$ matrix with rows $\Phi(X_i)^\top$.
Featurized logi. regr. with batch grad. descent:
$w \leftarrow 0$
repeat until convergence
$w \leftarrow w + \epsilon\,\Phi(X)^\top(y - s(\Phi(X)w))$
for each test pt z
$h(z) \leftarrow s(w^\top\Phi(z))$

$w = \Phi(X)^\top a$ $\quad k = \Phi(X)\cdot\Phi(X)^\top$ $\quad k_a = \Phi(X)\cdot w$

Dual/kernel logistic regression:
$a \leftarrow 0$
$\forall i, j$, $K_{ij} \leftarrow k(X_i, X_j)$ $O(n^2 d)$
repeat until convergence
$a \leftarrow a + \epsilon(y - s(Ka))$ $O(n^2)$
for each test pt z
$h(z) \leftarrow s\left(\sum_{i=1}^n a_i k(X_i, z)\right)$ $O(nd)$

j iteration

SGD
$w \leftarrow w + \epsilon\left(y_i - s(\Phi(X_i)^\top w)\right)\Phi(X_i)$

$q_i = Ka$. $q_i = (\Phi(X)\cdot w)_i = \Phi(X_i)^\top\cdot w$
Let K_{*i} denote column i of K.
$a \leftarrow 0$; $q \leftarrow 0$; $\forall i, j, K_{ij} \leftarrow k(X_i, X_j)$
repeat until convergence
choose random $i \in [1, n]$
$a_i \leftarrow a_i + \epsilon(y_i - s(q_i))$ $O(1)$
$q \leftarrow q + \epsilon(y_i - s(q_i))K_{*i}$ $O(n)$

Primal: $O(Dj)$ time
Dual (no kernel trick): $O(n^2 D + nj)$ time
Kernel: $O(n^2 d + nj)$ time

Running times for 3 ridge algs:

	primal	dual (no kernel trick)	kernel
train	$O(D^3 + D^2 n)$	$O(n^3 + n^2 D)$	$O(n^3 + n^2 d)$
test (per test pt)	$O(D)$	$O(nD)$	$O(nd)$

Primal: $O(nDj)$ time Dual (no kernel trick): $O(n^2 D + n^2 j)$ time Kernel: $O(n^2 d + n^2 j)$ time

poly kernel $k(x,z) = (x^\top z + 1)^p$

Theorem: $(x^\top z + 1)^p = \Phi(x)^\top\Phi(z)$ for some $\Phi(x)$ containing every monomial in x of degree $0 \dots p$.

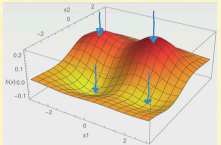
Example for $d = 2$, $p = 2$:

$$(x^\top z + 1)^2 = x_1^2 z_1^2 + x_2^2 z_2^2 + 2x_1 z_1 x_2 z_2 + 2x_1 z_1 + 2x_2 z_2 + 1$$
$$= [x_1^2 \ \ x_2^2 \ \ \sqrt{2}x_1 x_2 \ \ \sqrt{2}x_1 \ \ \sqrt{2}x_2 \ \ 1]\,[z_1^2 \ \ z_2^2 \ \ \sqrt{2}z_1 z_2 \ \ \sqrt{2}z_1 \ \ \sqrt{2}z_2 \ \ 1]^\top$$
$$= \Phi(x)^\top\Phi(z) \qquad \text{[This is how we're defining } \Phi.]$$

Gaussian kernel
$$k(x,z) = \exp\left(-\frac{\|x-z\|^2}{2\sigma^2}\right) \quad \Phi(x) = \exp\left(-\frac{x^2}{2\sigma^2}\right)\left[1, \frac{x}{\sigma\sqrt{1!}}, \frac{x^2}{\sigma^2\sqrt{2!}}, \frac{x^3}{\sigma^3\sqrt{3!}}, \dots\right]^\top$$

hypothesis $h(z) = \sum_{j=1}^n a_j k(X_j, z)$ is a linear combo of Gaussians centered at training pts

large σ ⇒ wide Gaussian & smooth h ⇒ more bias, less variance



① regression

no activation fn at output layer. squared loss

Pick loss fn $L(z, y)$ e.g., $L(z, y) = \|z - y\|^2$
↑ ↑
predictions true labels (could be vectors)

Cost fn is $J(V, W) = \frac{1}{n}\sum_{i=1}^n L(z(X_i), Y_i)$

Let w be a vector containing all the weights in V & W. Batch gradient descent:
$w = 0$ cannot break symmetry
$w \leftarrow$ vector of random weights
repeat
$w \leftarrow w - \epsilon\nabla J(w)$

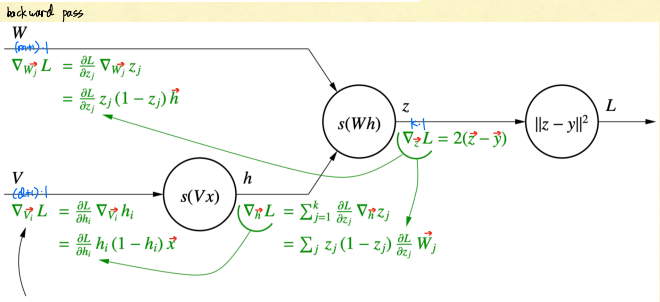
Forward pass
$V_i^\top$ is a row in matrix. V_i is transpose it to get a column vector

Recall $s'(\gamma) = s(\gamma)(1 - s(\gamma))$

$$\overset{(d+1)\cdot 1}{h_i} = s(\overset{(d+1)\cdot 1}{V_i}\cdot x), \text{ so } \overset{(d+1)\cdot 1}{\nabla_{V_i} h_i} = s'(V_i\cdot x)\,x = h_i(1 - h_i)\,x$$
$$\underset{(m+1)\cdot 1}{z_j} = s(\underset{(m+1)\cdot 1}{W_j}\cdot h), \text{ so } \underset{(m+1)\cdot 1}{\nabla_{W_j} z_j} = s'(W_j\cdot h)\,h = z_j(1 - z_j)\,h$$
$$\underset{(m+1)\cdot 1}{\nabla_h z_j} = s'(W_j\cdot h)\,W_j = z_j(1 - z_j)\,W_j$$

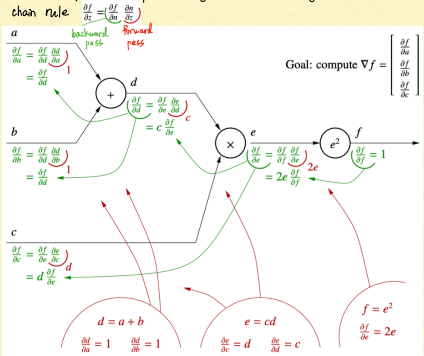
$\nabla_W(x^\top W) = (x^\top)^\top = x$ (1×n n×1)
$\nabla_W(W^\top x) = x$

backward pass



Compute $\nabla_V L$, $\nabla_W L$ one row at a time.

SGD (one pt each step). $O(\text{edge}^2)$ backprop → $O(\text{edge})$
chain rule $\frac{\partial f}{\partial c} = \left(\frac{\partial f}{\partial a}\cdot\frac{\partial a}{\partial c}\right)$



$Y = XW$. $\frac{\partial L}{\partial X} = \frac{\partial L}{\partial Y}\frac{\partial Y}{\partial X}$

$$\frac{\partial L}{\partial X_{ij}} = \sum_{l,m}\frac{\partial L}{\partial Y_{lm}}\cdot\frac{\partial Y_{lm}}{\partial X_{ij}}.$$

$$Y_{lm} = \sum_k X_{lk}\cdot W_{km} \Rightarrow \frac{\partial Y_{lm}}{\partial X_{ij}} = 0 \text{ when } l \neq i \text{ or } k \neq j$$
$$\frac{\partial Y_{lm}}{\partial X_{ij}} \neq 0 \text{ when } l = i \text{ and } k = j$$

when $l = i$ & $k = j$ $\quad \frac{\partial L}{\partial X_{ij}} = \sum_{l=i,m}\frac{\partial L}{\partial Y_{lm}}\frac{\partial Y_{lm}}{\partial X_{ij}}$, $\quad Y_{lm} = \sum_{k=j} X_{lj}\cdot W_{jm}$

$$= \sum_m \frac{\partial L}{\partial Y_{im}}\cdot W_{jm} = \sum_m \frac{\partial L}{\partial Y_{im}}\cdot W_{mj}^\top$$

$$\frac{\partial L}{\partial X} = \frac{\partial L}{\partial Y}\cdot W^\top$$

$$\frac{\partial L}{\partial X} = \begin{bmatrix} \frac{\partial L}{\partial X_{11}} = \sum_m \frac{\partial L}{\partial Y_{1m}} W_{m1}^\top \cdots \\ \frac{\partial L}{\partial X_{21}} = \sum_m \frac{\partial L}{\partial Y_{2m}} W_{m1}^\top \cdots \\ \vdots \end{bmatrix} \quad \frac{\partial L}{\partial Y} = \begin{bmatrix} \frac{\partial L}{\partial Y_{11}} & \frac{\partial L}{\partial Y_{12}} & \frac{\partial L}{\partial Y_{13}} & \cdots \\ \frac{\partial L}{\partial Y_{21}} & \frac{\partial L}{\partial Y_{22}} & \frac{\partial L}{\partial Y_{23}} & \cdots \\ \vdots \end{bmatrix} \quad W^\top = \begin{bmatrix} W_{11} & W_{21} & \cdots \\ W_{12} & W_{22} & \\ W_{13} & W_{23} & \\ \vdots & \vdots & \end{bmatrix}$$

learning theory

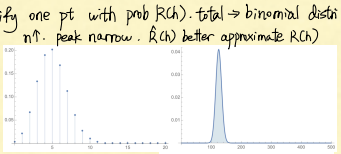
central idea: exclude lucky h : low $\hat{R}(h)$, high $R(h)$
less dichotomy

range space (P, H)
P: set of all train/test pts (eg. $\mathbb{R}^d$). $\sim D$
$h \subseteq P$. one hypothesis. all pts predecided in class C (infinite)
H: set of all hypothesis. set of (sets of pts)
Power set: $|P| = k$, contain all 2^k subsets of P
$P = \{1, 2\}, H = \{\emptyset, \{1\}, \{2\}, \{1, 2\}\}$

$R(h)$ true risk/generalization err: h misclassify random $x \sim D$
[mean test err. test err on ∞ data]

h^* minimize $R(h)$. best

h misclassify one pt with prob $R(h)$. total → binomial distri
n↑. peak narrow. $\hat{R}(h)$ better approximate $R(h)$



Hoeffding's inequality: $\Pr(\underbrace{|\hat{R}(h) - R(h)| > \epsilon}_{\text{lucky } h}) \le 2e^{-2\epsilon^2 n}$

X: train set
Dichotomy: $X \cap h$. all train pts predecided in class C (finite)
二分法

$\hat{R}(h)$ empirical risk/train err: h misclassify % train set X

$\hat{h}$: minimize $\hat{R}(h)$.

Π_H: H has Π dichotomy $\Pr\left(\text{at least one dichotomy has } |\hat{R} - R| > \epsilon\right) \le \delta$, where $\delta = 2\Pi e^{-2\epsilon^2 n}$
Π↓. n↑. better

fix δ, with prob $\ge 1 - \delta$, for every $h \in H$: $|\hat{R}(h) - R(h)| \le \epsilon = \sqrt{\frac{1}{2n}\ln\frac{2\Pi}{\delta}}$

$\to R(\hat{h}) \le \hat{R}(\hat{h}) + \epsilon$; $\hat{R}(h^*) \le R(h^*) + \epsilon$; $[\hat{R}(\hat{h}) \le \hat{R}(h^*)]$

$\to R(\hat{h}) \le \hat{R}(\hat{h}) + \epsilon \le \hat{R}(h^*) + \epsilon \le R(h^*) + 2\epsilon$, $\quad \epsilon = \sqrt{\frac{1}{2n}\ln\frac{2\Pi}{\delta}}$

Sample complexity: Fix ϵ & δ. # of train pt needed $n \ge \frac{1}{2\epsilon^2}\ln\frac{2\Pi}{\delta}$

shatter fn For linear classifier (# way divide pt into 2 class)
$\Pi_H(X) = 2^n \Rightarrow H$ shatters a set of n pt
for one X with size n $\prod_H(X) = |\{X \cap h : h \in H\}| \in [1, 2^n]$
for all X with size n $\prod_H(n) = \max_{|X|=n, X\subseteq P}\prod_H(X)$

Surprising fact: for all range spaces, either $\Pi_H(n)$ is polynomial in n, or $\Pi_H(n) = 2^n$

Cover's Theorem [1965]: linear classifiers in $\mathbb{R}^d$ allow up to $\Pi_H(n) = 2\sum_{i=0}^d \binom{n-1}{i}$ dichotomies of n pts.

For $n \le d + 1$, $\Pi_H(n) = 2^n$.
For $n \ge d + 1$, $\boxed{\Pi_H(n) \le 2\left(\frac{e(n-1)}{d}\right)^d}$ [Observe that this is polynomial in n! With exponent d.]
and the sample complexity needed to achieve $R(\hat{h}) \le \hat{R}(\hat{h}) + \epsilon \le R(h^*) + 2\epsilon$ with prob. $\ge 1 - \delta$ is

$$n \ge \frac{1}{2\epsilon^2}\left(d\ln\frac{n-1}{d} + d + \ln\frac{4}{\delta}\right). \quad \text{[Observe that the logarithm turned the exponent } d \text{ into a factor!]}$$

Corollary: linear classifiers need only $n \in \Theta(d)$ train pts

VC dim the threshold after which you cannot shatter $\mathrm{VC}(H) = \max\{n : \Pi_H(n) = 2^n\}$ (can be ∞)

(easy but loose)
Theorem: $\Pi_H(n) \le \sum_{i=0}^{\mathrm{VC}(H)}\binom{n}{i}$. Hence for $n \ge \mathrm{VC}(H)$, $\boxed{\Pi_H(n) \le \left(\frac{en}{\mathrm{VC}(H)}\right)^{\mathrm{VC}(H)}}$

usually $\mathrm{VC}(H) = d+1$
Cover's theorem

sample complexity $n \ge \frac{1}{2\epsilon^2}\left(\mathrm{VC}\cdot\ln\frac{n}{\mathrm{VC}} + \mathrm{VC} + \ln\frac{2}{\delta}\right)$

sample complexity $n \ge \frac{1}{2\epsilon^2}\ln\frac{2\Pi}{\delta}$
exp → constant

Corollary: O(VC(H)) train pt

eg. linear classifier in $\mathbb{R}^2$. $\Pi_H(3) = 8$. $\Pi_H(4) = 14$
$\mathrm{VC}(H) = 3$ $\quad \Pi_H(n) \le \left(\frac{en}{3}\right)^3$. sample complexity $O(1)$

VC dim for linear classifier $f(x) = \mathrm{sign}(w^\top x)$. $w \in \mathbb{R}^d$ is d (DIS 13)

$\epsilon = 0.05$. $\delta = 0.01$ $\quad \mathrm{VC} = 3$ $\quad \Pi \le \left(\frac{en}{3}\right)^3$ $\quad n \ge \frac{1}{2\epsilon^2}\ln\frac{2\Pi}{\delta}$

pt. $O(1)$