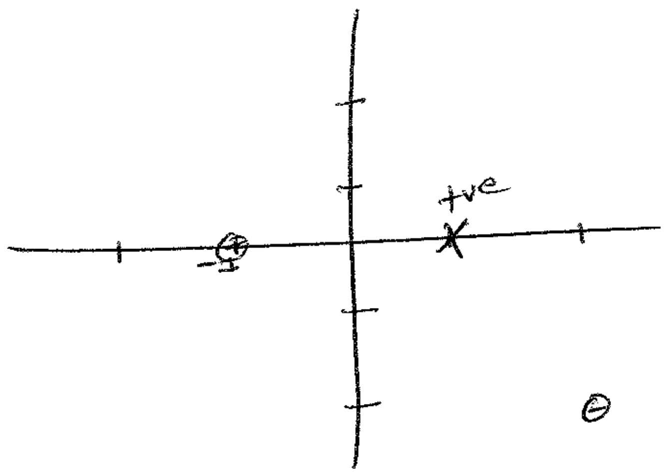
perceptron update



-1,0 -ve
2,-2 -ve

1,0 → +ve

final w = 0 1 2
let's say new point
$x = 2, -1.01$
$x_{aug} = 1\ 2\ -1.01$
$w x_{aug} = 0 + 2 - 2.02 < 0$
∴ -ve label

Augmented data

X_{aug}			t
1	-1	0	-1
1	2	-2	-1
1	1	0	1

initial wt $\vec{w} = (0\ 0\ 0)$

note we can also use $\eta\, \vec{w} \cdot \vec{x} + b$ original unaugmented x etc
$\eta = 0.9$
$b = 1$

	Test point	hypothesis	correct classified?	Updated weights	correct example
eg1	−: (1 −1 0)	$wx = 0+0+0 \geq 0$	false	$w = w - x = (-1\ 1\ 0)$	
eg2	−: (1 2 −2)	$wx = -1+2+0<0$	false	$w = w - x = (-2\ -1\ 2)$	
eg3	+: (1 1 0)	$wx = -2-1+0 \geq 0$	false	$w = w + x = (-1\ 0\ 2)$	
eg1	−: (1 −1 0)	$wx = -1+0+0<0$	true	$w = w = (-1, 0\ 2)$	
eg2	−: (1 2 −2)	$wx = -1+0+(-4)<0$	true	$w = w = (-1, 0, 2)$	
eg3	+: (1 1 0)	$wx = -1+0+0 \geq 0$	false	$w = w + x = (0\ 1\ 2)$	
eg1	−: (1 −1 0)	$wx = 0+(-1)+0<0$	true	$w = w = (0\ 1\ 2)$	
eg2	−: (1 2 −2)	$wx = 0+2+(-4)<0$	true	$w = w = (0\ 1\ 2)$	
eg3	+: (1 1 0)	$wx = 0+1+0 \geq 0$	true	$w = w = 0, 1, 2$	

final weight

confusion matrix

Actual class

		cat	not cat	
predicted class	cat	TP	FP	$\tilde{P}$
	not cat	FN	TN	$\tilde{N}$
		P	N	

True

00 = TN
01 = FP
10 = FN
11 = TP

	1	0
1	TP	FP
0	FN	TN
	P	N

actual

	0	1	
0	TN	FN	$\tilde{N}$
1	FP	TP	$\tilde{P}$
	N	P	

→ precision $= \frac{TP}{TP + FP}$

recall $= \frac{TP}{P} = \frac{TP}{TP + FN}$

(hit rate)

$\frac{\text{predicted true}}{\text{actual true}}$

F1 score is the harmonic mean of precision and recall.

$$F1 = \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}}$$

$$= \frac{2\ \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

* prove that the number of elements in x and y is also a kernel

$\Rightarrow k(x,y) = |x \cap y|$ is a kernel.

— x — x — x — x —

contd (Nov 11)

$$w^T \phi(x) = \sum_{n=1}^{N} \alpha_n t_n K(x_n, x)$$

$$= \sum_{m \in S} \alpha_m t_m K(x_m, x)$$

$$+ \sum_{m \notin S} \alpha_m t_m K(x_m, x)$$

but if $x_m \notin S$, then $\alpha_m = 0$

since $S = n$ $\alpha_n > 0$ such that

$1, 2, \ldots N$

$n \notin S \Rightarrow \alpha_n = 0$

* package libsvm or svm lite

input $\rightarrow \{(x_n, t_n)\}$ $1 \le n \le N$

output $\rightarrow \{\alpha_m, t_m, x_m\}$ $m \in S$

model.txt $\rightarrow$ $\boxed{\begin{matrix} b \\ \alpha_m t_m \cup x_m \end{matrix}}$

pg 3 here $n = 1, 2, \ldots, N$

Suppose, out of N samples, there are m support vectors then ~~N~~-m samples will have ~~lagrange~~ lagrange parameter $\alpha = 0$ and m examples will have non-zero Lagrange parameters.

$$1 - t_m w^T \phi(x_m) - t_m b = 0$$

or, $$1 - t_m \phi(x_m) \sum_n \alpha_n t_n \phi(x_n) - t_m b = 0$$

or, $$t_m b = 1 - t_m \phi(x_m) \sum_n \alpha_n t_n \phi(x_n)$$

$$= 1 - t_m \sum_n \alpha_n t_n \phi(x_n) \phi(x_m)$$

$$b t_m = 1 - t_m \sum_n \alpha_n t_n \phi(x_n) \phi(x_m)$$

$$\therefore b = \frac{1}{t_m} - \sum_n \alpha_n t_n \phi(x_n) \phi(x_m) = t_m - \sum_n \alpha_n t_n \phi(x_n) \cdot \phi(x_m)$$

$$\boxed{b = t_m - w \cdot \phi(x_m)}$$

$$\boxed{b = t_m - \sum_n \alpha_n t_n k(x_n, x_m)} \quad \because \frac{1}{t_m} = t_m$$

$$1 = \frac{1}{1} \text{ and } -1 = \frac{1}{-1}$$

this is true for all the m examples which have non zero Lagrange parameter α.

For numerical stability we choose value of b as the mean of all b-values, then,

$$\boxed{b = \frac{1}{|S|} \sum_{m \in S} \left[t_m - \sum_{n \in S} \alpha_n t_n k(x_n, x_m) \right]}$$

where S is the subset of all the examples where lagrange parameter α is non zero.

$$S \subseteq D$$

$$S = \{ n \mid 1 - t_n w^T \phi(x_n) - t_n b = 0 \}$$

then, Linear discriminant function is,

$$\boxed{y(x) = w^T \phi(x) + b = \sum_{m \in S} \alpha_m t_m k(x, x_m) + \frac{1}{|S|} \sum_{m \in S} [t_m - \sum_{n \in S} \alpha_n t_n k(x_n, x_m)]}$$

Then Dual Lagrangian is,

$$L_D(\alpha) = \frac{1}{2}\sum_{n,m} \alpha_m\alpha_n t_m t_n \phi_m \phi_n + \sum_n \alpha_n - \sum_n \alpha_n t_n \phi_n \cdot \sum_m \alpha_m t_m \phi_m$$

$$- \cancel{\sum_n \alpha_n t_n b}^{\to 0}$$

$$\boxed{L_D(\alpha) = \sum_n \alpha_n - \frac{1}{2}\sum_{n,m} \alpha_m \alpha_n t_m t_n K(x_n, x_m)}$$

$$K(x,y) = \vec{\phi}(x) \cdot \vec{\phi}(y) = \phi^T(x)\,\phi(y)$$

Then the optimization problem in dual space is,

maximize $L_D(\alpha) = \sum_n \alpha_n - \frac{1}{2}\sum_n \sum_m \alpha_m \alpha_n t_m t_n K(x_m, x_n)$

s.t. $\alpha_n \geq 0 \quad \forall n \in \{1 \cdots N\}$

$$\sum_{n=1}^{N} \alpha_n t_n = 0$$

Look here

Now, KKT conditions are,

note: we can write $y(x_n) = w^T\phi(x_n) + b$

① Primal constraints $\quad 1 - t_n(w^T\phi(x_n) + b) \leq 0$ (convex constraint equation) $t_n(w^T\phi(x_n)+b) \geq 1$

② dual constraint $\quad \alpha_n \geq 0 \quad \forall n \in \{1, \cdots N\}$

③ commementary ~~slackness~~ $\quad \alpha_n \{1 - t_n w^T\phi(x_n) - t_n b\} = 0$

for any data point, either $\alpha_n = 0$

$1 - t_n w^T\phi(x_n) - t_n b = 0$

these x_n are called support vectors

① Q① solve the SVM problem without slack using lagrange multiplier method

The optimization problem is

minimize $J(w,b) = \frac{1}{2}\|w\|^2$

s.t. $t_n(w^T\phi(x_n)+b) \geq 1 \quad \forall n \in \{1,\dots,N\}$

$1 \leq t_n(w^T\phi(x_n)+b)$

$1 \leq t_n w^T\phi(x_n) + t_n b$

$1 - t_n w^T\phi(x_n) - t_n b \leq 0$ (convex fn constraint)

compare $f_i(x) \leq 0 \quad i = 1,\dots,m$

primal Lagrangian,

$$L_p(w,b,\alpha) = \frac{1}{2}\|w\|^2 + \sum_{n=1}^{N}\alpha_n(1 - t_n w^T\phi_n - t_n b)$$

where $\alpha_n \geq 0$ are Lagrange multipliers.

Dual Lagrangian,

$$L_D(\alpha) = \inf_{w,b} L_p(w,b,\alpha)$$

first find the infimum of L_p w.r.t w, b:

$\frac{\partial}{\partial w}L_p = 0 = w + \sum_n \alpha_n(-)t_n\phi_n \Rightarrow \boxed{w = \sum_n \alpha_n t_n \phi_n}$ — ①

$\frac{\partial}{\partial b}L_p = 0 = \sum_n (-)\alpha_n t_n \Rightarrow \boxed{\sum_n \alpha_n t_n = 0}$ — ②

LOOK LHS

① constrained optimization
(Lagrange multipliers)

maximize $5-(x_1-2)^2-2(x_2-1)^2$

s.t. $x_1+4x_2=3$

Sol^n: If we ignore constraint we get $x_1=2, x_2=1$
then $x_1+4x_2 = 2+4\cdot 1 = 6$ is too large
for the constraint.
consider
$L = L(x_1,x_2,\lambda) = 5-(x_1-2)^2-2(x_2-1)^2+\lambda(3-x_1-4x_2)$

Look λ:
$\lambda=0$ $\quad 2,1$
$\lambda=1$ $\quad 3/2, 0$
$\lambda=\frac{2}{3}$ $\quad \left(\frac{5}{3}, \frac{1}{3}\right)$ $\quad \frac{5}{3}+4\cdot\frac{1}{3}=\frac{5}{3}+\frac{4}{3}=\frac{9}{3}=3$ ✓

formal sol^n

$\frac{\partial L}{\partial x_1} = -2(x_1-2) - \lambda = 0$
$= -2x_1+4-\lambda=0 \Rightarrow 2x_1 = 4-\lambda$

$2x_1 = 4-\lambda$
$= 4-\frac{2}{3}$
$= \frac{4\cdot 3-2}{3} = \frac{10}{3}$
$x_1 = \frac{5}{3}$

$\frac{\partial L}{\partial x_2} = -4(x_2-1)-4\lambda = 0$
$= -4x_2+4-4\lambda=0$
$\Rightarrow 4x_2 = 4-4\lambda$
$x_2 = 1-\lambda$

$\frac{\partial L}{\partial \lambda} = 3-x_1-4x_2=0$
$\Rightarrow 6-2x_1-8x_2=0$
$\Rightarrow 6-4+\lambda-8(1-\lambda)=0$
$\Rightarrow (6-4+\lambda)(-8+8\lambda)=0$

$-6+9\lambda=0$
$9\lambda=6$
$\lambda=2/3$
~~$6+17\lambda=0$~~
~~$17\lambda=6$~~
~~$\lambda=\frac{6}{17}$~~

① why kernels are symmetric?

Inner products are symmetric by definitions, so, therefore if the kernel function represent an inner product in some Hilbert space, then the kernel function must be symmetric as well.

$$K(x, y) = \langle \varphi(x), \varphi(y) \rangle$$
$$= \langle \varphi(y), \varphi(x) \rangle \quad (\because \text{ property of inner product})$$
$$= K(y, x)$$

② show $K(x, z) = x^T A^T A z$ is a valid kernel.

let $\phi(x) = Ax$,

then,

$$\langle \phi(x), \phi(z) \rangle = \phi(x)^T \phi(z)$$
$$= (Ax)^T (Az)$$
$$= x^T A^T A z$$
$$= K(x, z)$$

$\therefore$ $K(x, z)$ is an inner product in some Hilbert space.

(3) AA^T is PSD matrix (indefinite)

proof Let, $A \in \mathbb{R}^{m \times n}$

$\lambda =$ eigen value of AA^T

$q =$ eigen vector of λ

$(AA^T) q = \lambda q$ (premultiply by q^T)

$\Rightarrow q^T AA^T q = q^T \lambda q$ $\because AA^T = A^T A$

$\Rightarrow \lambda = \dfrac{q^T AA^T q}{q^T q} = \dfrac{q^T A^T A q}{q^T q}$

$= (Aq)^T Aq / q^T q$

$= \dfrac{z^T z}{q^T q}$ where $z = A^T q$

here $z^T z \geq 0$

$q^T q \geq 0$

$\therefore \lambda \geq 0$ so AA^T is PSD.

multivariate Linear

bias	X floor size	bed	t
1	100	2	2K
1	200	3	3K
1	5000	5	5K

t

m features and one bias

$$X = \begin{bmatrix} x_0^{(1)}=1 & x_1^{(1)} & \cdots & x_m^{(1)} \\ x_0^{(2)} & x_1^{(2)} & \cdots & x_m^{(2)} \\ \vdots & \vdots & & \\ x_0^{N} & x_1^{(N)} & & x_m^{(N)} \end{bmatrix}_{N \times m+1} \qquad \begin{bmatrix} t_1 \\ t_2 \\ t_3 \\ \vdots \\ t_n \end{bmatrix}_{N \times 1}$$

There are m features and one bias total m+1
There are N samples for each feature

$$J = \frac{1}{2N}|Xw - t|^2 = \frac{1}{2N}(Xw-t)^T(Xw-t)$$

$$\nabla_w J = \frac{1}{\cancel{2}} \underbrace{\frac{d}{dx}(Xw-t)} \cdot \cancel{2}(Xw-t) \qquad \frac{d}{dx}(x^T x) = 2x$$

x is a matrix

$$0 = X^T(Xw-t)$$

$$\boxed{\frac{d}{dx}(Ax+b)^T(Ax+b) = 2A^T(Ax+b)}$$

$$X^T X w = X^T t$$

weights $\boxed{w = (X^T X)^{-1}(X^T t)}$

Derivatives of matrix products

L1 vs L2 norm
more used

L2 → more penalty on large weights, but doesn't drive small weights to zero.

L1 → Less penalty for large wt, but leads many weights ~~towards~~ to (or very very close) to zero. leading to weight vector to be sparse.

Bias-variance Tradeoff

(a) Ridge: $\sum_{i=1}^{N}\left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij}\right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 = RSS + \lambda \sum_{j=1}^{p} \beta_j^2$

shrinkage parameter which has to be tuned via validation set → regularization strength

Ridge
$\lambda\uparrow \Rightarrow$ var$\downarrow$ bias$\uparrow$
meaning: small change in training data big change in parameter estimates
effect will increase with no of parameters

(b) Lasso β_j^2 is $|\beta_j|$

Ridge shrinks w_j but do not make $\emptyset$
but Lasso makes them zero, makes less no of wts.

① Add ones column

= np.c_[np.ones(X.shape[0]).reshape(-1,1), X]

X = np.c_[np.ones(X.shape[0])[:,np.newaxis].T, X]

② correct = np.sum(y_pred == y_test)

accuracy = correct / len(y_pred)

⑥ Gradient Descent (GD or BGD)

vanilla GD

$$v^{\tau+1} = \eta \nabla J(w^{\tau})$$

$$w^{\tau+1} = w^{\tau} - v^{\tau+1}$$

GD with momentum

$$v^{\tau+1} = \gamma v^{\tau} + \eta \nabla J(w^{\tau})$$

$$w^{\tau+1} = w^{\tau} - v^{\tau+1}$$

Batch Gradient Descent (Lect01, p.24)

$$J(w) = \frac{1}{2N} \sum_{n=1}^{N} (h_n - t_n)^2$$

$$w^{\tau+1} = w^{\tau} - \eta \nabla J(w^{\tau})$$

$$= w^{\tau} - \eta \sum_{n=1}^{N} (h_n - t_n) x_n$$

$$= w^{\tau} - \text{learninRate } (h-t) @ X1$$

(4) X = design matrix $[\ ]_{N\times M}$ N = samples, m = features

$X1$ = biased design matrix $\begin{bmatrix} 1 & \\ 1 & \end{bmatrix}_{N, M+1}$

$t = [\ ]_{N\times 1}$ column vector

$w = [w_0 \ w_1 \ w_2 \ w_M]_{1, M+1}$ row vecter (2d array)

w = np.array(w).reshape(1, X1.shape[1]) $= (1, M+1)$

linear regression

$h = X1 \,@\, w.T = (N, M+1)\,(M+1, 1) = (N, 1)$ same as t

$e = h - t = N, 1 = 50, 1$

$SSE = \sum_{n=1}^{N} (h-t)^2$

$MSE = \frac{1}{N} \sum_{n=1}^{N} (h-t)^2$

$J = \frac{1}{2N} \sum_{n=1}^{N} (h-t)^2$ = np.sum((h-t)**2) / 2 / float(N)

$RMSE = E_{RMS} = \sqrt{MSE}$

Normal eqn: $w = \underbrace{(X^T X)^{-1} @ X^T}_{\text{moore penrose pseudo inverse of } X} @\, t$

np.linalg.pinv(X)

① cost for multivariate linear regression

$$J = \frac{1}{2N} \sum_{n=1}^{N} (h_n - t_n)^2 \qquad h = w^T X \qquad h = X1 @ w.T$$

$$J = \frac{1}{2N} \sum_{n=1}^{N} \left(\sum_{j=0}^{M} w_j x_n^j - t_n \right)^2 \qquad h_n = \sum_{j=0}^{M} w_j x_n^j$$

$h_1 = w_0 + w_1 x_1^{(1)} + w_2 x_1^{(2)} + w_M x_1^{(M)}$ first row of sample

$w = [w_0 \; w_1 \; w_2 \; w_3]_{1\times 4}$ bias

X1

$$\begin{bmatrix} 1 & 10 & 100 \\ 1 & 20 & 200 \\ 1 & 30 & 300 \\ 1 & 50 & 500 \end{bmatrix}_{50\times 4} \qquad t = \begin{bmatrix} 1k \\ 2k \\ 3k \\ 50k \end{bmatrix}_{50\times 1}$$

2 features age, floorsize

$h = (50,4) \cdot (4,1) = (50,1)$

$$e = h - t = \begin{bmatrix} h_1 - t_1 \\ h_2 - t_2 \\ \vdots \\ h_{50} - t_{50} \end{bmatrix}_{50\times 1}$$

SSE = (h-t) ** 2

$$MSE = \frac{(h-t) ** 2}{N}$$

MSE = np.mean(SSE)

RMSE = np.sqrt(MSE)

J = np.sum((h-t)**2) / 2/N

J = 0.5 * np.mean(np.sum(h-t)**2)

$J = \frac{1}{2} \times MSE$ (single float)

$$\boxed{J = \frac{0.5}{\text{len}(t)} (h-t)^T (h-t)}$$

$$(h-t)^T (h-t) = [h_1 - t_1 \; h_2 - t_2 \; \cdots]_{1\times 50} \begin{bmatrix} h_1 - t_1 \\ h_2 - t_2 \\ \vdots \\ h_n - t_n \end{bmatrix}_{50\times 1} = \left[\sum_n (h_n - t_n)^2 \right]_{1\times 1}$$

② gradient of J

$$\nabla_w J = \nabla_w \frac{1}{2N} \sum_n (h-t)^2 = \left[\frac{\partial J}{\partial w_0} \; \frac{\partial J}{\partial w_1} \; \frac{\partial J}{\partial w_2} \; \frac{\partial J}{\partial w_M} \right]_{4\times 1}$$

= (h-t).T @ X1 / N

= (1,50) · (50,4)

grad = $\nabla_w J$ = (1,4)

$$\boxed{\text{grad\_ols} = (h-t).T @ X1}$$

$$\frac{\partial J}{\partial w_j} \frac{1}{2N} \sum_{n=1}^{N} (x w_n - t)^2 = \frac{1}{2N} \sum (x_n w_0 + x_n w_1 + \cdots + x_n w_m)$$

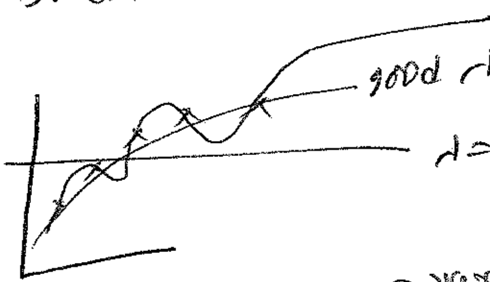
$$= \frac{1}{2N} \sum_{n=1}^{N} 2 (x w_j - t) \cdot X$$

summation vanishes b/c we use matrix

$$\nabla_w J = \frac{1}{N} (h-t) \cdot X$$

① L2 → more penalty larger weights
does not drive weights to zero
weight vector is <u>NOT</u> sparse.

② $\lambda \uparrow$ ⇒ high bias, low variance, underfit, simpler model
drives weights closer to zero.
small change in training data ⇒ big change in estimate



N - examples
M - features

cost function for L2-regularized linear regression

$$X = \begin{bmatrix} x_1^1 & x_1^2 & x_1^3 & x_1^M \\ x_2^1 & x_2^2 & x_2^3 & x_2^M \\ x_N^1 & x_N^2 & x_N^3 & x_N^M \end{bmatrix}$$

③ $$J = \frac{1}{2N} \sum_{n=1}^{N} (h_n - t_n)^2 + \frac{\lambda}{2} \|w\|^2$$

$$J(w) = \frac{1}{2N} \sum_{n=1}^{N} \left(w_0 + \sum_{j=1}^{M} w_j x_n^j - t_n\right)^2 + \frac{\lambda}{2} \sum_{j=1}^{M} w_j^2$$

(we omit w_0 in regularization)

regularizing w_0 = shifting origin of target
⇒ same change in all target values ⇏ similar change in estimates

④ Batch GD

for num-iters:

$$w^{\tau+1} = w^\tau - \eta \nabla J$$

learning rate

stochastic GD

for num-iters:
for sample in data:
$$w^{\tau+1} = w^\tau - \eta \nabla J$$

momentum GD

$$v^{\tau+1} = \eta \nabla J(w^\tau)$$
$$w^{\tau+1} = w^\tau - v^{\tau+1}$$

$\gamma \approx 0.9$

Nesterov Accelerated gradient

$$w^{\tau+1} = w^\tau - \eta \nabla J(w^\tau - \gamma v^\tau) - \gamma v^\tau$$

5 Gradient Descent

η = learning rate
γ = momentum hyperparameter
w^{τ} = weight vector at time stamp τ

$$v^{\tau+1} = \begin{cases} \eta \nabla J(w^{\tau}) & \text{vanilla} \\ \eta \nabla J(w^{\tau}) + \gamma v^{\tau} & \text{with momentum} \\ \eta \nabla J(w^{\tau} - \gamma v^{\tau}) + \gamma v^{\tau} & \text{Nesterov Accelerated gradient NAG} \end{cases}$$

$$w^{\tau+1} = w^{\tau} - v^{\tau+1}$$

$\gamma \approx 0.9$

5b Types of GD based on data used:

Batch GD $\rightarrow$ use all data

Stochastic GD $\rightarrow$ use only one example and update w after each iteration

minibatch GD $\rightarrow$ use constant number of examples and update w after each

$$J = \frac{1}{2N} \sum_{n=1}^{N} (h_n - t_n)^2$$

$$\nabla J = \frac{1}{N} \sum_{n=1}^{N} (h_n - t_n) \cdot x_n = \frac{1}{N} \cdot \text{np.sum}((h-t) \cdot x)$$

$$w = w - \eta \, \text{grad}$$

5c GD vs Normal eqn

$$w = w - \eta \, \text{grad}$$

$$w = \underbrace{(X^T X)^{-1} X^T}_{\text{pseudoinverse (pinv)}} t$$

$$w = \underbrace{(X^T X + \lambda N I)^{-1}}_{\text{for L2 regularized}} X^T t$$

I is identity matrix of shape $X^T X$

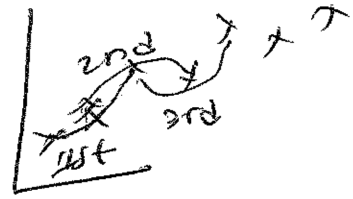
$I[0][0] = 0$ for not to regularize bias term

⑥ cubic spline smoothing

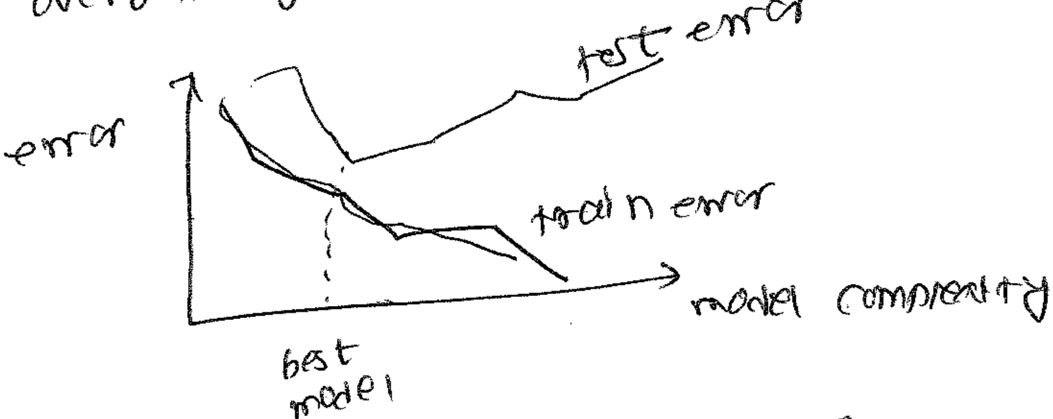
Take 3 points x $x-x_i$ $x+x_i$

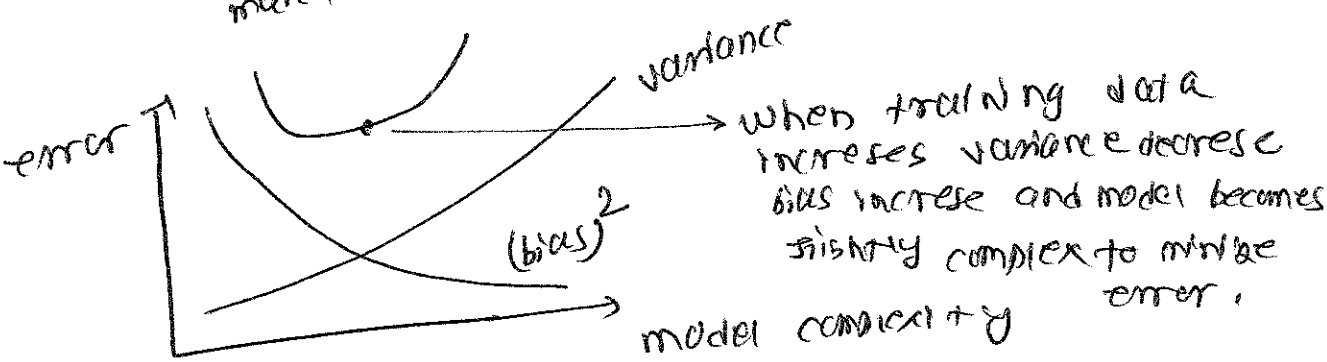
$$s_i(x) = a_i(x-x_i)^3 + b_i(x-x_i)^2 + c_i(x-x_i) + d_i$$

$$\forall x \in [x_i, x_{i+1}]$$



⑦ overfitting





① Likelihood function for logistic Regression

$$p(t_n|w) = h_n^{t_n} \cdot (1-h_n)^{1-t_n}$$

$$p(t|w) = \prod_{n=1}^{N} h_n^{t_n} (1-h_n)^{1-t_n}$$

$$\text{cost } E = \frac{1}{N} \cdot (-\ln p) = -\frac{1}{N} \sum_{n=1}^{N} \left[t_n \ln h_n + (1-t_n) \ln(1-h_n) \right]$$

$$\text{grad} = \nabla_w J = \frac{1}{N} \cdot \sum_{n=1}^{N} (h_n - t_n) \cdot x_n$$

② perceptron criterion

minimize $E_p(w) = - \sum_{n \in \text{Misclassified}} t_n w^T x_n$

initialize $w = 0$, $\bar{w} = 0$, $\tau = 1$

for $i = 1, \dots, N$

$h_n = \text{sgn}(w^T x_n)$

if $h_n \neq t_n$ then

$w = w + t_n x_n$

$\tau = \tau + 1$

$\bar{w} = \bar{w} + w$

$\tau = \tau + 1$

return $\bar{w} / \tau$

Test: $\text{sgn}(w^T x)$

① SVM with slack (soft margin SVM)

$$\min J(\omega,b,\xi) = \frac{1}{2}\|\omega\|^2 + C\sum_{n=1}^{N}\xi_n$$

with constraint

$$t_n(\omega^T\phi(x_n)+b) \geq 1-\xi_n$$

$$\xi_n \geq 0 \quad \text{and} \quad \sum_{i=1}^{N}\xi_i \leq Z \qquad \forall n\in\{1,\ldots,N\}$$

i.e. $1-\xi_n - t_n\omega^T\phi(x_n) - bt_n \leq 0$ —— ① (convex fn)

$-\xi_n \leq 0$ —— ② (convex fn constraint)

$\forall n = 1,2,\ldots,N$

primal Lagrangian

$$L_p = L(\omega,b,\xi,\alpha,r) = \frac{1}{2}\|\omega\|^2 + C\sum_{n=1}^{N}\xi_n + \sum_{n=1}^{N}\alpha_n(1-\xi_n-t_n\omega^T\phi_n-bt_n) - \sum_{n=1}^{N}r_n\xi_n$$

Dual Lagrangian

$$L_D(\alpha,r) = \inf_{\omega,b,\xi} L_p(\omega,b,\xi,\alpha,r)$$

Now, $0 = \frac{\partial L_p(\omega,b,\xi,\alpha,r)}{\partial\omega} = \omega - \sum_n\alpha_n t_n\phi(x_n) \Rightarrow \boxed{\omega = \sum_n\alpha_n t_n\phi(x_n)}$

$0 = \frac{\partial L_p}{\partial b} = -\sum_n\alpha_n t_n \Rightarrow \boxed{\sum_n\alpha_n t_n = 0}$

NO summation!

& $0 = \frac{\partial L_p}{\partial\xi_n} = C - \alpha_n - r_n \Rightarrow \boxed{C = \alpha_n + r_n} \quad \forall n\in\{1,2,\ldots,N\}$

Then,

$$L_D(\alpha, r) = \frac{1}{2}\|w\|^2 + C \sum_n \xi_n + \sum_n \alpha_n (1 - \xi_n - t_n w^T \phi_n - t_n b) - \sum_n r_n \xi_n$$

$$L_D = \sum_n \alpha_n - \frac{1}{2} \sum_{m,n=1}^{N} \alpha_m \alpha_n t_m t_n \, k(x_m, x_n)$$

Then the optimization problem in dual space is,

$$\underset{\alpha}{\text{maximize}} \quad L_D(\alpha) := \sum_n \alpha_n - \frac{1}{2} \sum_{m,n} \alpha_m \alpha_n t_m t_n \, k(x_m, x_n)$$

with constraints $\quad 0 \leq \alpha_n \leq C \quad \forall n = 1, 2, \cdots, N$

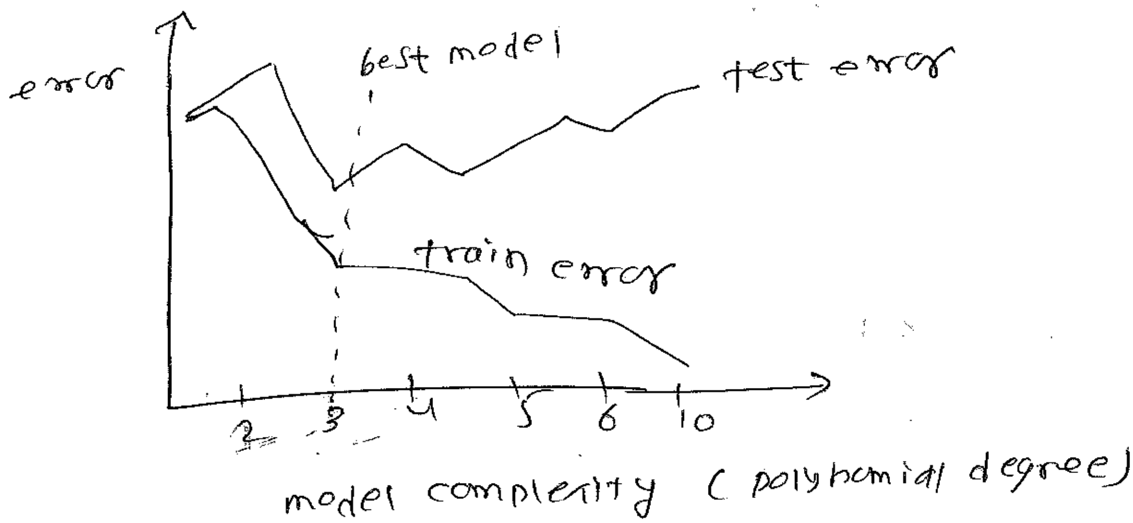
$$\sum_{n=1}^{N} \alpha_n t_n = 0$$

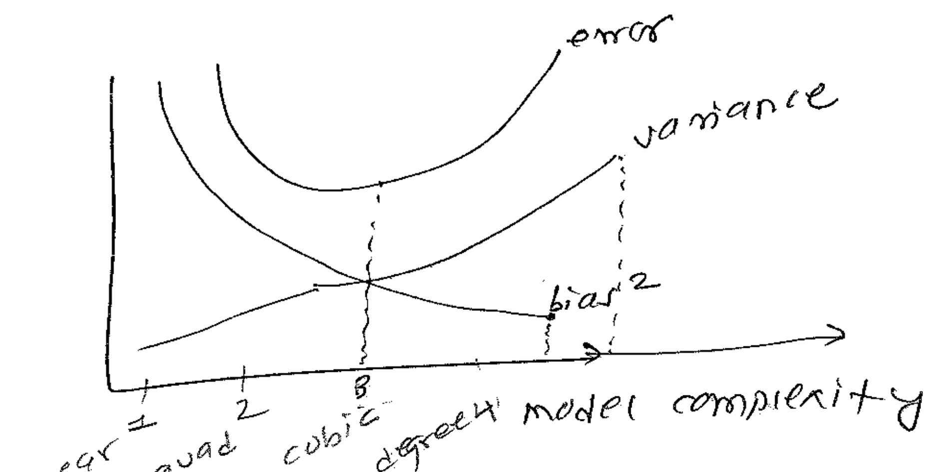
note: $\quad C \sum_n \xi_n - \sum_n \alpha_n \xi_n - \sum_n r_n \xi_n = 0$

since, $\quad C \sum_n \xi_n = \sum_n \alpha_n \xi_n + \sum_n r_n \xi_n$

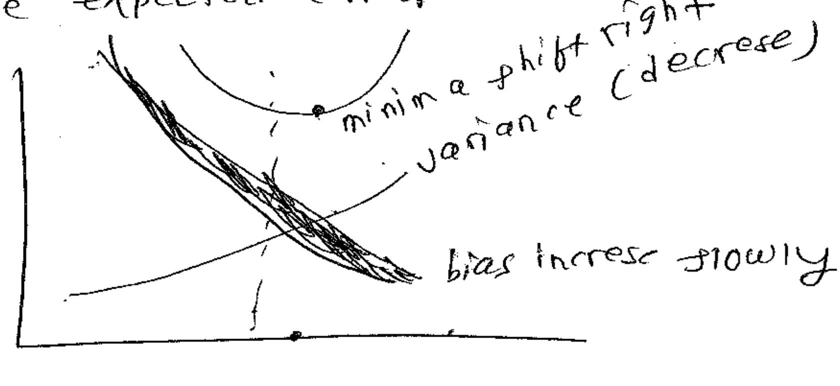
$$= \sum_n (\alpha_n + r_n) \xi_n$$

overfitting





when training data increase, variance is reduced and slightly more complicated model minimizes the expected error.



① Show that $x^T A y$ is a valid kernel if A is sym. p.s.d.

$\Rightarrow$ $A = Q^T \Lambda Q$ where $\Lambda = \mathrm{diag}(\lambda_1, \lambda_2, \cdots, \lambda_n)$ and $\lambda_1, \lambda_2, \cdots, \lambda_n \geq 0$

define $F = \mathrm{diag}(\sqrt{\lambda_1}, \sqrt{\lambda_2}, \cdots, \sqrt{\lambda_n})$ then $\Lambda = F^T F$

$\Rightarrow A = Q^T F^T F Q$

$\Rightarrow x^T A y = x^T Q^T F^T F Q y$

$= (F Q x)^T (F Q y)$

$= \phi(x)^T \phi(y)$

where $\phi(x) = F Q x$

Aside:

$(AB)^T = B^T A^T$

$(ABC)^T = C^T B^T A^T$

$((AB)C)^T = C^T (AB)^T$

$= C^T B^T A^T$

KNN = memory-based (no model to fit)

① $\begin{cases} \text{find K nearest examples } x_1, x_2 \dots x_K \text{ from test } x, \\ y(x) = \underset{t \in T}{\arg\max} \sum_{i=1}^{K} \delta_t(t_i) \end{cases}$

w_i gives distance-weighted KNN

$w_i = \frac{1}{\|x - x_i\|^2}$

KNN usage

① Satellite image classification

② digit recognition

② mahalanobis dis

$d(x,y) = \sqrt{(x-y)^T S^{-1} (x-y)}$

sample covariance matrix

if $S = I$ = Euclidean distance

$S = \text{diag}(\sigma_1^{-2}, \sigma_2^{-2}, \dots, \sigma_K^{-2})$

normalized Euclidean

③ cosine similarity

$d(x,y) = 1 - \frac{x^T y}{\|x\| \|y\|}$

④ Kernel based distance weighted NN

→ binary classification $T \in \{+1, -1\}$

→ $y(x) = \text{sign}\left(\sum_{i=1}^{N} K(x, x_i) \cdot t_i\right)$

Wrapper method

① greedy
Forward selection
F = all features
S = subset of features
start S = { } empty
for each feature f in F−S
find best performing feature f* and add to S.
Repeat until performance does not incrrese
or performance good enough

② Recursive
Backword Elimination
F = {1, 2, … K} is set of features
S = [ ] ranked set of features
Repeat until F−S is empty
train $\vec{w}$ using linear SVM and F−S
find feature f with minimum $|w_f|$
append f to S

Return S

④ distance-weighted KNN for regression

1. find k nearest points על פי מרחק

2. $$y(x) = \frac{\sum_{i=1}^{k} w_i t_i}{\sum_i w_i}$$ where $w_i = \frac{1}{\|x - x_i\|^2}$

KNN → Shepard's method

$$y(x) = \sum_{i=1}^{N} K(x, x_i)\, t_i \Big/ \sum_{i=1}^{N} K(x, x_i)$$ (kernel-based dist weighted)

⑤ Regression with KNN

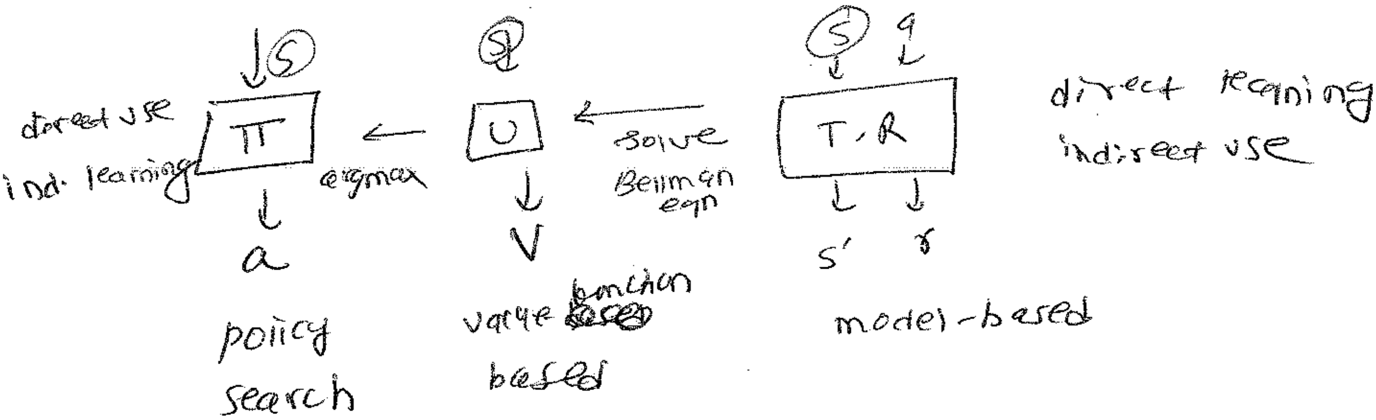
$$y(x) = \frac{1}{K} \sum_{i=1}^{k} t_i$$ (t_i are values, not classes)

⑥ distance-weighted KNN (regression)

primal: $$y(x) = \sum_{i=1}^{k} w_i t_i \Big/ \sum_{i=1}^{k} w_i$$ $w_i = \frac{1}{\|x - x_i\|^2}$

kernel based: $$y(x) = \sum_{i=1}^{N} K(x, x_i)\, t_i \Big/ \sum_{i=1}^{N} K(x, x_i)$$ (t_i are values not classes)

3 Approaches to RL



Filter method of feature selection

① Mutual Information

$$MI(X,Y) = \sum_{x}\sum_{y} p(x,y) \ln \frac{p(x,y)}{p(x)\,p(y)}$$

= 0 when x, y indep
max when x = y

Let there are K examples with l features.

O_{ij} = observed value for $x = i$, $y = j$

②

$$\begin{array}{c|c} & 1 \;\cdots\; j \;\cdots\; l \\ \hline 1 & \\ \vdots & \\ i & O_{ij} \quad \} N_{x=i} \\ \vdots & \\ K & \underbrace{\qquad}_{N_{y=j}} \end{array}$$

$$E_{ij} = \frac{N_{x=i}\, N_{y=j}}{N}$$

$$\chi_e^2 = \sum_{i=1}^{K}\sum_{j=1}^{l} \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$$

③ pearson corr coeff

$$\rho(X,Y) = \frac{E[(X-\mu_x)(Y-\mu_y)]}{\sigma_x \, \sigma_y} \quad \text{(population)}$$

(sample)

$$\rho(x,y) = \frac{\sum_{i=1}^{n} (x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n} (x_i-\bar{x})^2} \cdot \sqrt{\sum_{i=1}^{n} (y_i-\bar{y})^2}}$$

~~Example~~ $-1 \le \rho(x,y) \le 1$

④ SNR

binary classes $= \{+,-\}$
mean $= \{\mu_+, \mu_-\}$
examples are binary
$y \in \{+1,-1\}$

$$\mu(X,Y) = \frac{|\mu_+ - \mu_-|}{\sigma_+ + \sigma_-}$$

μ_+, σ_+ = mean & std for +ve class samples

⑤ T-test

$$T(X,Y) = \frac{|\mu_+ - \mu_-|}{\sqrt{\frac{\sigma_+^2}{\mu_+} + \frac{\sigma_-^2}{\mu_-}}}$$

* 3 parametric approaches to

① Discriminant function

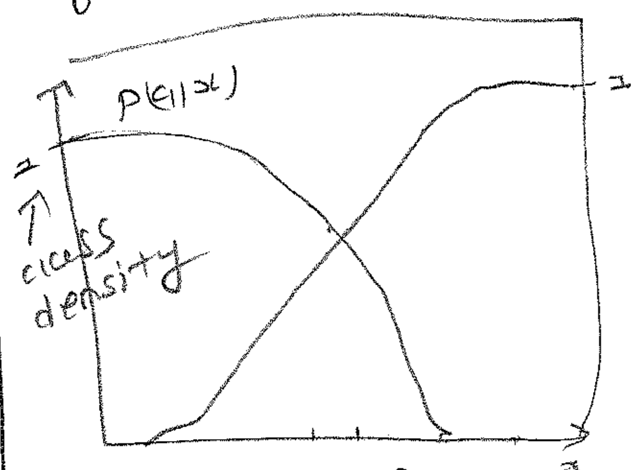
- Fisher's linear disc
- perceptron
- SVM

inference and decision are combined as single learning problem

② probabilistic discriminative models

- Logistic regression
- conditional random fields

- inf, dec → separate
- less data need to compute $p(C_k|x)$ than $p(x|C_k)$
- can accommodate many overlapping features



discriminative

classification
③ probabilistic Generative models

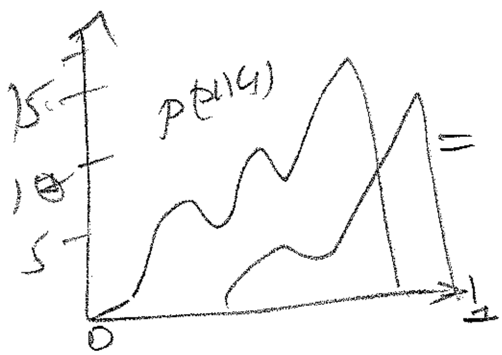
{ Naive Bias
Hidden Markov models

- inf. dec → separate
- can use $p(x)$ for outlier or novelty detection
- need to model dependencies between features

NaiveBias → resilient to noise

- text classification with NB → look HW
- multiple class
posterior prob of class c_k given data x is,

$$p(c_k|x) = \frac{p(x|c_k) \cdot p(c_k)}{\sum_j p(x|c_j)\, p(c_j)}$$



$$= \frac{\exp(a_k(x))}{\sum_j \exp(a_j(x))}$$

normalized exponentials
(softmax fn)

generative where $a_k(x) = \ln p(x|c_k) \cdot p(c_k)$

① SVM for ranking

optimization problem

minimize $J(w,b) = \frac{1}{2}\|w\|^2 + C\sum \xi_{k,i,j}$

s.t. $w^T\phi(q_k, d_i) \geq w^T\phi(q_k, d_j) + 1 - \xi_{k,i,j}$

$\xi_{k,i,j} \geq 0$

(for a query q_k we want document d_i be ranked higher than document q_j)

Logistic Regression (classification (binary))

linear regression $h = w^T x$

logistic regression $h = \sigma(x) = \frac{1}{1+e^{-w^T x}} = h(x) = y(x)$

likelihood function $p(t|w) = \prod_n h_n^{t_n} (1-h_n)^{1-t_n}$

-ve log likelihood, E or $J = -\ln p(t|w)$

$$= -\sum_{n=1}^{N} [t_n \ln h_n + (1-t_n)\ln(1-h_n)]$$

∴ cost or Error or Loss function

$$\boxed{E_D = -\sum_n [t_n \ln h_n + (1-t_n)\ln(1-h_n)]}$$

where $t_n \in \{0,1\}$ NOT $\{-1,1\}$

add regularizer,

$$E_w = \frac{\lambda}{2} w^T w$$

then L2-regularized logistic regression cost function

cost ~~##~~

$$\boxed{E = -\frac{1}{N}\sum_n [t_n \ln h_n + (1-t_n)\ln(1-h_n)] + \frac{\lambda}{2} w^T w}$$

λ is regularizer

$$h(x) = \sigma(x) = \frac{1}{1+e^{-w^T x}}$$

$$h_n = h(x_n) = \sigma(x_n) = \frac{1}{1+e^{-w^T x_n}}$$ is sigmoid f^n

...classification (multiclass)

Softmax Regression

Training set: $(x_1, t_1), (x_2, t_2), \ldots (x_n, t_n)$

$X = [1, x_1, x_2, \ldots, x_n]$

$t_1, t_2, \ldots, t_n \in \{1, 2, \ldots, K\}$

$w_k = [w_{k0}, w_{k1}, w_{k2}, \ldots]^T$

one weight vector per class,

$$p(C_k|x) = \frac{e^{w_k^T x}}{\sum_j e^{w_j^T x}}$$

prob that nth example x_n has class t_n

$$p(t_n|x_n) = \frac{e^{w_{t_n}^T x_n}}{\sum_j e^{w_{t_j}^T x_n}}$$

likelihood is the joint probability of all classes

$$L(w) = \prod_{n=1}^{N} p(t_n|x_n)$$

MLE

sample: $x_1, x_2, \ldots, x_N$

prob of x is $f(x_n, w)$

joint prob of $x_1, x_2, \ldots x_N$ is called likelihood $L(w)$

$L(w) = p(X_1 = x_1, \ldots X_N = x_N)$

$= f(x_1, w) \ldots$

$\prod f(x_n)$

$$L(w) = \prod_{n=1}^{N} f(x_n, w)$$

cost is the -ve log likelihood,

$$E_D(w) = -\frac{1}{N} \ln L(w)$$

$$= -\frac{1}{N} \ln \prod_{n=1}^{N} p(t_n|x_n)$$

$$= -\frac{1}{N} \sum_n \ln p(t_n|x_n)$$

$$= -\frac{1}{N} \sum_n \ln \frac{e^{w_{t_n}^T x_n}}{Z(x_n)}$$

w_{t_n} is weight vector for nth example

w_k is weight vector for all the examples belonging to class k.

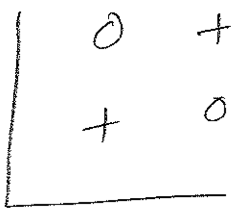
$$\boxed{E_D(w) = -\frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{K} \delta_k(t_n) \ln\left(\frac{e^{w_k^T x_n}}{Z(x_n)}\right)}$$

python

① $X = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$ append ones column to data X.

```
X = np.array([ [1,2,3], [4,5,6] ]) = np.arange(1,7).reshape(2,3)
ones = np.ones( X.shape[0] ).reshape(-1,1)
X1 = np.append( ones, X, axis=1 ).astype(np.int)
     concatenate
X1 = np.c_[ np.ones( X.shape[0] ).reshape(-1,1) , X]
X1 = np.column_stack(( " ))
X1 = np.c_[np.ones( X.shape[0] )[:, np.newaxis] , X ]
                                 .reshape(-1,1)
   = np.c_[np.atleast_2d( np.ones( X.shape[0] ) ).T , X]
   = np.c_[np. expand_dims( np.ones( X.shape[0], axis=1) , X ]
```

①



- SVM (quad kernel) can acheive zero training error
- Logis Regr, 3-NN cannot

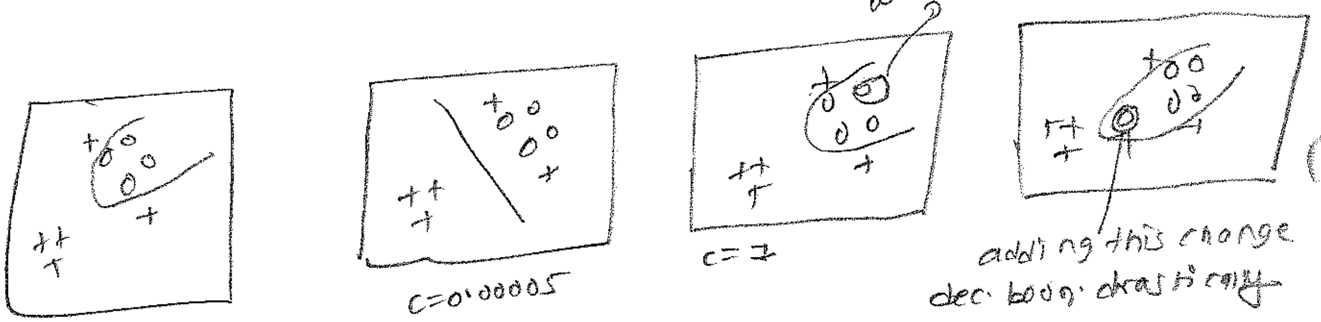
② If examples are iid, increase training examples → may increase train error
↳ but decrease test error

③ SVM effect of 'C'

$$\min J(w,b) = \frac{1}{2}\|w\|^2 + C\sum_{n=1}^{N}\xi_n$$

$$s.t. \quad w^T\phi(x_n)+b \geq 1-\xi_n \quad \forall n \in \{0,N\}$$

will not change decision bond.



C= 10,000

C=0.00005

C=1

adding this change dec. boun. drastically

between C>>1 and C≈0 choose C≈0 because it maximizes the margin between dominant cloud of points and we can not depend on only few data points which can be noise.

④ Bias variance Tradeoff

	Bias	var
linear regr	high	low
d=2 poly	low	low
d=10 poly	low	high

(5) Given $\phi(x) = [1, x_1, x_2, x_1 x_2]$

find the kernel $k(x, x')$.

$\Rightarrow$ $k(x, x') = 1 + x_1 x_1' + x_2 x_2' + x_1 x_2 x_1' x_2'$

(6) L1 vs L2 loss

(a) False: L2 is more robust to outlier than L1

gradient of L2 loss can grow without bounds, but, gradient of L1 loss is bounded, hence influence of outlier is limited.

(Note: L2 gives more values to misclassification than L1)

(b) L1 gives sparse solution & used in feature selection.

(c) Logistic loss is better than L2 loss in classification task.

(7) SVM small C,

For linearly separable data, small C can affect training accuracy.

A small C can allow large slacks, thus, the resulting classifier will have smaller w^2 and can have non-zero training error.

Distance metrics

① Euclidean $d(x,y) = \|x-y\|_2 = \sqrt{(x-y)^T (x-y)}$

② Hamming $d(x,y) =$ # of different values in fixed length strings

③ Mahalanobis distance $d(x,y) = \sqrt{(x-y)^T S^{-1} (x-y)}$

S is sample cov. matrix

$S = I \rightarrow$ Euclidean

$S = diag(\sigma_1^{-2}, \sigma_2^{-2}, \ldots) \rightarrow$ normalized Euclidean dist.

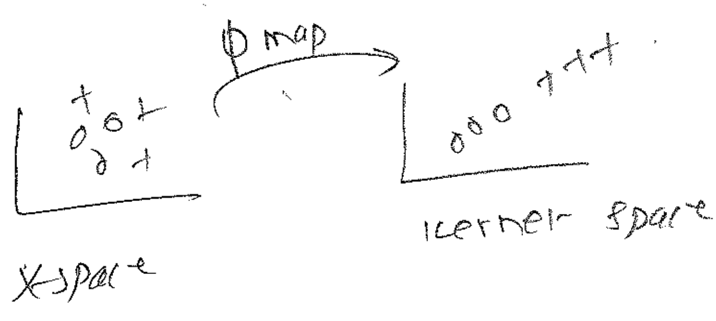
③ cosine similarity

$d(x,y) = 1 - \cos(x,y) = 1 - \frac{x^T y}{\|x\| \|y\|}$

④ Levenstein distance (edit distance)

min # of basic operations (del, insert, juxtapose) betn two strings

$x =$ 'athens' $y =$ 'hints' $d(x,y) = 4$



① mutual information

$$MI(X,Y) = \sum_X \sum_Y p(x,y) \cdot \ln\left(\frac{p(x,y)}{p(x)\,p(y)}\right) \quad \begin{array}{l} = 0 \text{ when } x, y \text{ indp} \\ = \max \text{ when } x = y \end{array}$$

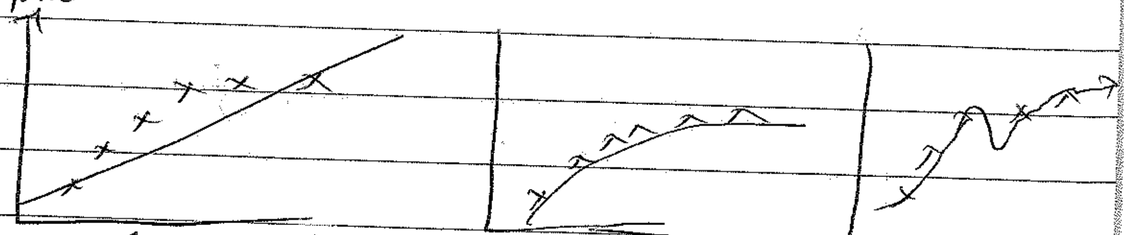
$$= KL\left[p(x,y) \,\|\, p(x)\,p(y)\right]$$

bad → biased towards high arity features

bad → may choose redundant feature

bad → works only with nominal features & labels

Bias-variance Tradeoff



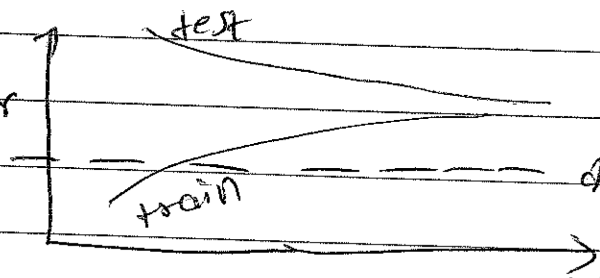
deg = 1 | deg = 2 | deg = 4

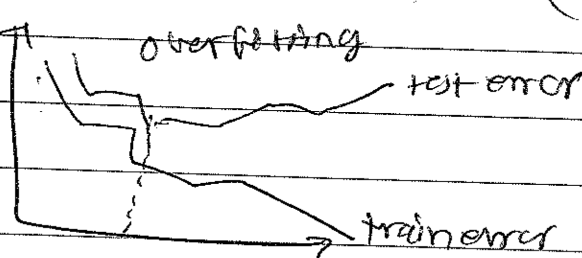
$w_0 + w_1 x$ | $w_0 + w_1 x + w_2 x^2$ | $w_0 + w_1 x + w_2 x^2 + w_3 x^3 + w_4 x^4$

high bias (underfit) (simple model) | just right | high var. (overfit)

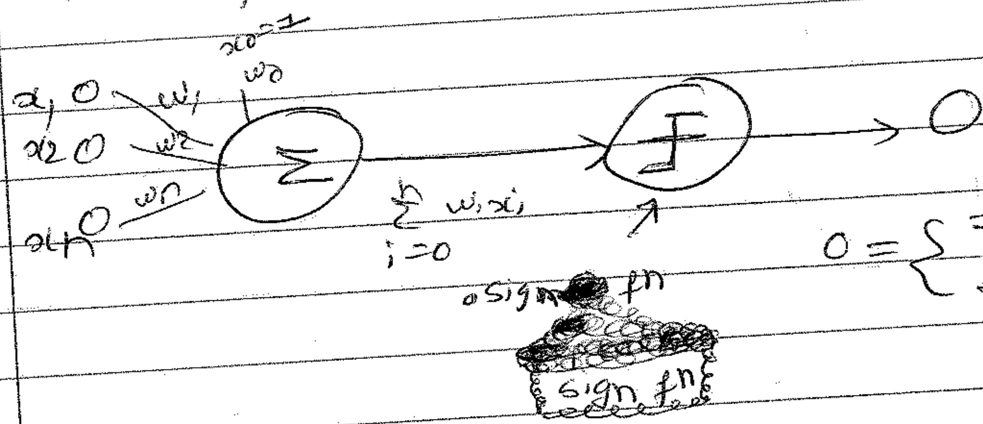
more training eg → fix high σ^2 (overfitting)
smaller feature set → fix high σ^2 (")
larger feature set → fix high bias (underfitting)

Learning curve for high bias (underfit)





perceptron



$$O = \begin{cases} 1 & \text{if } \sum_{i=0}^{n} w_i x_i > 0 \\ -1 & \text{otherwise} \end{cases}$$

$$o(x_1, x_2, \ldots, x_n) = \begin{cases} 1 & \text{if } w_0 + w_1 x_1 + \cdots + w_n x_n > 0 \\ -1 & \text{otherwise} \end{cases}$$

output

$$= \begin{cases} 1 & \text{if } \vec{w} \cdot \vec{x} > 0 \\ -1 & \text{otherwise} \end{cases}$$

Filter	vs wrapper
1 much faster since no need to train the model	• computationally expen
2 use ~~statistical~~ method of evaluation	• uses cross-validation
3 might fail to find best subset	• finds best subset
4 less prone to overfitting	• more prone to overfitting

① SVM for regression

$$\min J(w,b) = \frac{1}{2}\|w\|^2 + C\sum_{n=1}^{N}(\xi_n + \hat{\xi}_n)$$

s.t. $t_n \le t_n(w^T\phi(x_n)+b) + \epsilon + \xi_n$

$t_n \ge t_n(w^T\phi(x_n)+b) - \epsilon - \hat{\xi}_n$

$\xi_n, \hat{\xi}_n \ge 0 \quad \forall\ 1 \le n \le N$

② SVM for ranking

$$\min J(w,b) = \frac{1}{2}\|w\|^2 + C\sum \xi_{k,i,j}$$

s.t. $w^T\phi(q_k, d_i) \ge w^T(\phi\, q_k, d_j) + 1 - \xi_{k,i,j}$

$\xi_{k,i,j} \ge 0$

$$MI(X,Y) = \sum_x \sum_y p(x,y) \cdot \ln \frac{p(x,y)}{p(x)\cdot p(y)} = \begin{cases} 0 & \text{when } x,y \text{ indep} \\ \max & \text{when } x=y \end{cases}$$

5 mutual information
- → works with nominal
- → biased towards high arity feature
- → may choose redundant feature

② chi square

$$E_{ij} = \frac{N_{x=i} \cdot N_{y=j}}{N}$$

$$x_0^2 = \sum_{i,j} \frac{(O_{ij}-E_{ij})^2}{E_{ij}}$$

l features →

$$\begin{array}{c|ccccc} & 1 & \cdots & j & \cdots & l \\ & i & \cdots & O_{ij} & \cdots & \} N_{x=i} \\ K \text{ examples} & K & & \vdots & & \\ & & & N_{y=j} & & \end{array}$$

③ pearson corr. coeff

$$\rho(x,y) = \frac{E[(x-\mu_x)(y-\mu_y)]}{\sigma_x \sigma_y} = \frac{\mathrm{cov}(x,y)}{\sigma_x \sigma_y}$$

④ SNR

$$\mu(x, \overset{\text{binary}}{y}) = \frac{|\mu_+ - \mu_-|}{\sigma_+ + \sigma_-}$$

⑤ FTest

$$T(x,y) = \frac{|\mu_+ - \mu_-|}{\sqrt{\frac{\sigma_+^2}{N_+} + \frac{\sigma_-^2}{N_-}}}$$

* A kernel will be valid if there exist a space such that

$$K(x_1, x_2) = \phi(x_1)^T \phi(x_2) = \sum_{n=1}^{N} \phi_n(x_1) \phi_n(x_2)$$

* Consider a quadratic kernel with $D = 2$,

$$K(x, z) = (x^T z)^2 = (x_1 z_1 + x_2 z_2)^2$$

$$= (x_1^2 z_1^2 + 2 x_1 z_1 x_2 z_2 + x_2^2 z_2^2)$$

$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$

$z = \begin{pmatrix} z_1 \\ z_2 \end{pmatrix}$

$x^T = (x_1\ x_2)$

$x^T z = (x_1\ x_2) \begin{pmatrix} z_1 \\ z_2 \end{pmatrix}$

This can be expressed as an inner product of space where,

$$\phi(x) = x_1^2 + \sqrt{2} x_1 x_2 + x_2^2$$

$$\phi(z) = z_1^2 + \sqrt{2} z_1 z_2 + z_2^2$$

this gives, $\phi(x)\phi(z) = x_1^2 z_1^2 + 2 x_1 x_2 z_1 z_2 + x_2^2 z_2^2$

$$K(x, z) = \phi(x)^T \phi(z)$$

* A necessary and sufficient condition for a kernel function to be "valid" is that the gram matrix be positive and semidefinite for all choices of $\{\vec{x}_m\}$.

A gram matrix of $\vec{x}$ is $x^T x$.

The linear vector x is projected into a quadratic surface.

If all the points in this surface are non-zero, then our kernel is valid.

* Fisher's discriminant cost

$$J(w) = Tr\{(w S_w w^T)^{-1}\} \cdot (w S_B w^T)$$

— × — × — × — × Tue Oct 31

~~least square~~ perceptron

DOES NOT WORK

$$E(w) = \frac{1}{2} \sum_{n=1}^{N} (h_n - t_n)^2$$

N-1 mistake N mistakes

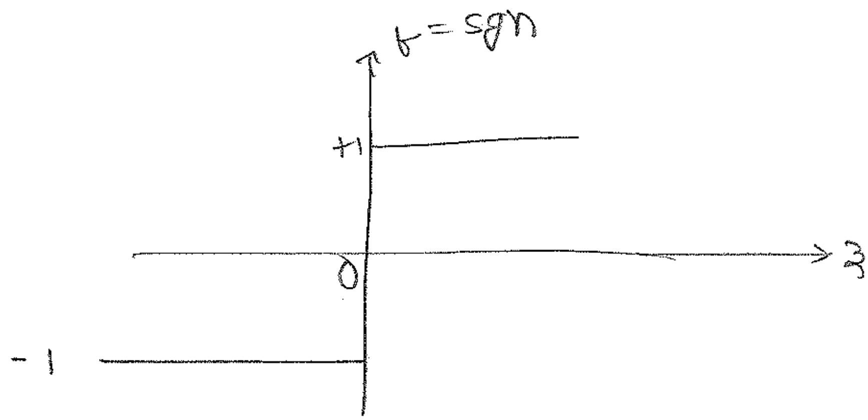
$0 \quad \frac{1}{2} \cdots \frac{N-2}{2} \quad \frac{N}{2}$ N+1 values

min $E = 0$
max $E = \frac{N}{2}$

we cannot compute gradient, since function is discrete and not continuous.

cost = No. of misclassified patterns
$= 0$ for no mistake
$\frac{N}{2}$ for N mistakes

cost $= [0 \ \frac{1}{2} \ 2 \cdots \frac{N-2}{2} \ \frac{N}{2}]$ discrete set.



op1

$$\min_{w,b} \frac{1}{2}\|w\|^2$$

$$\text{s.t. } t_n(w^T\phi(x_n)+b) \geq 1$$

solution $= w_1^*, b_1^*$

op2

$$\min_{w,b} \frac{1}{2}\|w\|^2$$

$$\text{s.t. } t_n(w^T\phi(x_n)+b) \geq r$$

$\Updownarrow$ op3

$$\min_{w,b} \frac{1}{2}\left\|\frac{w}{r}\right\|^2$$

$$\text{s.t. } t_n\left(\left(\frac{w}{r}\right)^T\phi(x_n) + \frac{b}{r}\right) \geq 1$$

op3 has solution

$w_2^* = w_1^* r$

$b_2^* = b_1^* r$

$\Updownarrow$ op3 rename

$w' = w/r$

$b' = b/r$

$$\min_{w,b} \frac{1}{2}\|w'\|^2$$

$$\text{s.t. } t_n(w'\phi(x_n)+b') \geq 1$$

Now, decision hyperplanes are,

$$H1 = \{x \mid w_1^{*T}\phi(x) + b_1^* = 0\}$$

$$H2 = \{x \mid w_2^*\phi(x) + b_2^* = 0\}$$

$$= \{x \mid r w_1^*\phi(x) + r b_1^* = 0\}$$

$$= \{x \mid w_1^*\phi(x) + b_1^* = 0\}$$

$H2 = H1$ q.e.d

3 KMCP Kernel multi-class perceptron

1 define $f(x) = \sum_{i,j} \alpha_{ij} \left[\phi(x_i, t_i)^T \phi(x, t) - \phi(x_i, c_j)^T \phi(x, t) \right]$

$f(x) = w^T x = \sum_n \alpha_n t_n x_n^T x = \sum_n \alpha_n t_n K(x_n, x)$

2 initialize dual params $\alpha_{ij} = 0$

3 for $i = 1, \dots n$

4 $\quad c_j = \arg\max_{t \in T} f(x_i, t)$ $\quad$ ($h_n = \text{sgn}(f(x))$)

5 $\quad$ if $c_j \neq t_i$ then $\quad$ ($h_n \neq t_n$)

6 $\quad\quad \alpha_{ij} = \alpha_{ij} + 1$ $\quad$ ($\alpha_n = \alpha_n + 1$)

Repeat

Testing: $t^* = \arg\max_{t \in T} f(x, t)$ $\quad$ ($h(x) = \text{sgn}(f(x))$)

2 mcp (concept of kernel comes from here)

initialize parameters $w = 0$

for $i = 1 \cdot N$

$\quad c_j = \arg\max_{t \in T} w^T \phi(x_i, t)$

$\quad$ if $c_j \neq t_i$ then

$\quad\quad w = w + \phi(x_i, t_i) - \phi(x_i, c_j)$

w is loop invariant and is the weighted average

$$w = \sum_{i,j} \alpha_{ij} \left(\phi(x_i, t_i) - \phi(x_i, c_j) \right)$$

$$f(x) = w^T \phi(x, t) = \sum_{ij} \alpha_{ij} \left(\phi(x_i, t_i)^T \phi(x, t) - \phi(x_i, c_j)^T \phi(x, t) \right)$$

1 KP

define $f(x) = w^T x = \sum_n \alpha_n t_n K(x_n, x)$

initialize dual parameter $\alpha_n = 0$

for $i = 1 \cdots N$

$\quad h_n = \text{sign}(f(x))$

$\quad$ if $h_n \neq t_i$ then

$\quad\quad \alpha_n = \alpha_n + 1$

Repeat

TEST: $y(x) = \text{sign}(f(x))$

MDP Markov decision process

① policy

argmax

$$\pi^* = \underset{\pi}{\operatorname{argmax}} \; E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) \mid \pi \right]$$

(policy expectation)

$$\pi^* = \underset{a}{\operatorname{argmax}} \sum_{s'} T(s,a,s') \, U(s')$$

(best action policy)

R = Reward

$\gamma^t R$ = discounted Reward

② utility

expectation & max only

$$U(s) = E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) \mid \pi, \; s_0 = s \right]$$

(expectation)

$$U(s) = R(s) + \gamma \max_{a} \sum_{s'} T(s,a,s') \, U(s')$$ ← Bellman eqn

* Naive Bias

3 Boolean input vectors $x_1 \, x_2 \, x_3$ and output y

$p(y=1)$

$P(x_1=1 \mid y=0)$ $P(x_1=1 \mid y=1)$

$P(x_2=1 \mid y=0)$ $P(x_2=1 \mid y=1)$

$P(x_3=1 \mid y=0)$ $P(x_3=1 \mid y=1)$

- # of parameters $= 2m+1 = 2\times3+1 = 7$
- # of parameters if no conditional independence $= 1+2(2^m-1) = 1+2(2^3-1) = 1+2(8-1) = 1+14 = 15$

① MLE VS MAP

MLE = maximize p(data|params) by searching over parameters

MAP = maximize p(param|data) by searching over params and accounting for prior over params.

MLE → finds ω by maximizing likelihood fn p(D|ω)

MAP → maximizes the posterior prob p(ω|D)

② classification maps inputs to discrete outputs

Regression " continuous " .

③ PCA & feature selection

similarity : reduce the dimension of data

difference : feature selection finds a subset of features, PCA produces a smaller new set

perceptron

perceptron criterion: $w^T x_n \geq 0$ for $t_n = +1$

$w^T x_n < 0$ for $t_n = -1$

want: $t_n w^T x_n \geq 0$ for all patterns

minimize: $-w^T x_n t_n$ for all missclassified patterns M

$$\boxed{\underbrace{E_p}_{\text{perceptron}}(w) = -\sum_{\substack{n \in M \\ \uparrow \\ \text{mistakes (missclassified)}}} w^T x_n t_n}$$

Binary perceptron

initialize $\vec{w} = 0$

for $n = 1, \dots, N$

$h_n = \text{sgn}(w^T x_n)$

if $h_n \neq t_n$ then

$w = w + t_n x_n$

repeat until convergence or given number of epochs

softmax Regression

$$P(C_k|x) = \frac{e^{w_k^T x}}{\sum_j e^{w_j^T x}}$$

note:

$$h(w) = \frac{1}{\sum_{k=1}^{K} e^{w_k^T x}} \begin{bmatrix} e^{w_1^T x} \\ e^{w_2^T x} \\ \vdots \\ e^{w_K^T x} \end{bmatrix}$$

$$p(t_n|x_n) = \frac{e^{w_{t_n}^T x_n}}{\sum_{j=1}^{N} e^{w_{t_n}^T x_n}}$$

$$E_D(w) = \frac{1}{N} \cdot -\ln \prod_n p(t_n|x_n)$$

$$= -\frac{1}{N} \ln \prod_n \frac{e^{w_{t_n}^T x_n}}{\sum_j e^{w_{t_n}^T x_n}}$$

$$= -\frac{1}{N} \sum_n \ln \frac{e^{w_{t_n}^T x_n}}{\sum_j e^{w_{t_n}^T x_n}}$$

$$\boxed{E_D(w) = -\frac{1}{N} \sum_n \sum_k \delta_k(t_n) \ln\left(\frac{e^{w_k^T x_n}}{\sum_k e^{w_k^T x_n}}\right)}$$

regularizer

$$+ \frac{\alpha}{2} \sum_{k=1}^{K} w_k^T w_k$$

$$\frac{\partial E_w}{\partial w_k} = \alpha w_k$$

(no summation)

Let, $E_n = \sum_k \delta_k(t_n) w_k^T x_n - \sum_k \delta_k(t_n) \ln\left(\sum_k e^{w_k^T x_n}\right)$

$$\frac{\partial E_n}{\partial w_j} = \delta_j(t_n) x_n - \delta_j(t_n) \cdot \frac{1}{\sum_j e^{w_j^T x_n}} \cdot e^{w_j^T x_n} \cdot \delta_j(t_n) x_n$$

$$\frac{\partial E_n}{\partial w_k} = \delta_k(t_n) x_n - \frac{e^{w_k^T x_n}}{\sum_k e^{w_k^T x_n}} \cdot x_n$$

prevent overflow

$$p(C_k|x_n) = \frac{e^{w_k^T x_n}}{\sum_k e^{w_k^T x_n}}$$

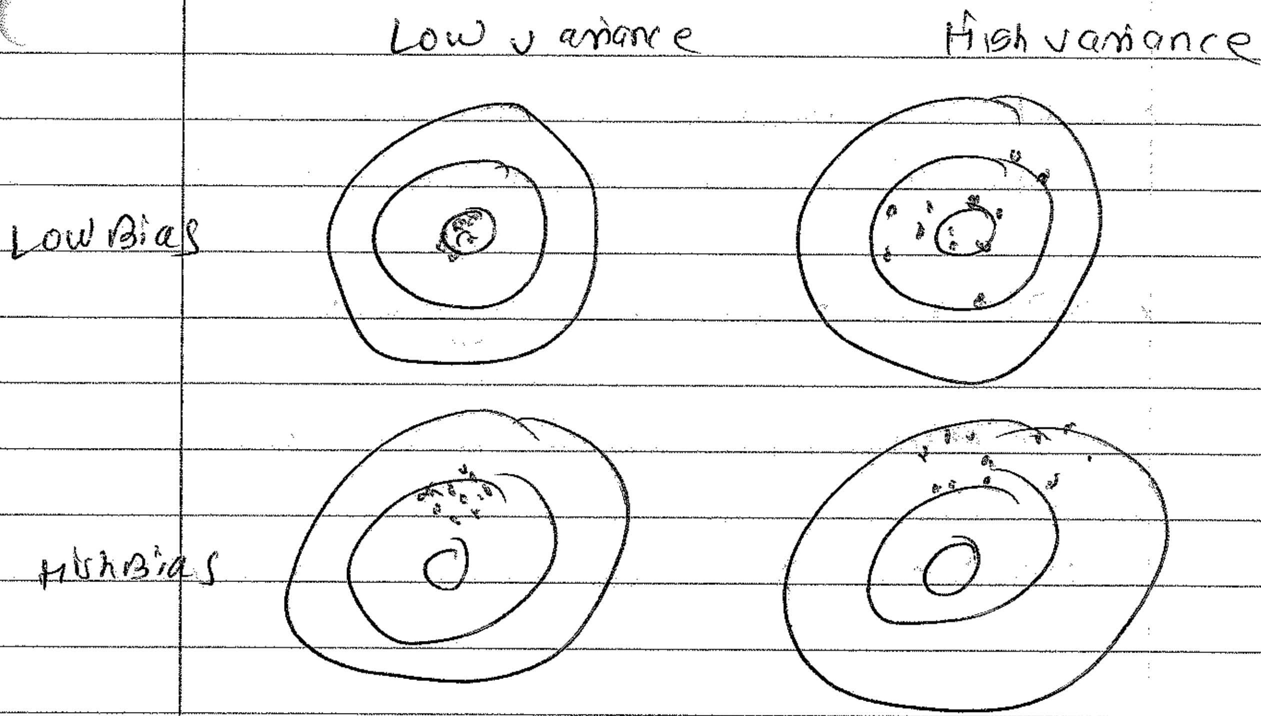
$$c = \max_{1 \le k \le K} w_k^T x_n$$

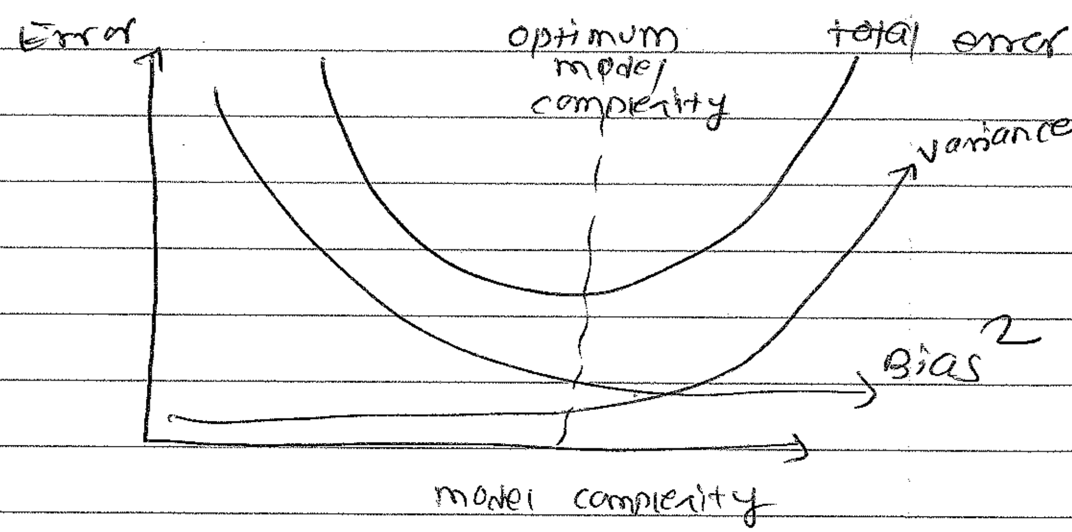
$$p(C_k|x_n) = \frac{e^{(w_k^T x_n - c)}}{\sum_k \exp(w_k^T x_n - c)}$$

$$\frac{\partial E_n}{\partial w_k} = \left[\delta_k(t_n) - p(C_k|x_n)\right] x_n$$

$$\Rightarrow \boxed{\frac{\partial E_D(w)}{\partial w_k} = -\frac{1}{N} \sum_n \left[\delta_k(t_n) - p(C_k|x_n)\right] x}$$

there are K-such equations for each classes.


Low variance
High variance
Low Bias
High Bias


Error
optimum model complexity
total error
variance
Bias^2
model complexity

maximum likelihood estimation   maximum a priori

MLE vs MAP

- MLE maximizes the $\ln p(D|w)$ likelihood of model (w) w.r.t. data D
- MAP maximizes the $\ln p(w|D)$ likelihood of data D w.r.t. model w

moreover, MAP additionally uses priors.

MAP solution for LR

$$p(t|w) = \prod_{n=1}^{N} h_n^{t_n} (1-h_n)^{1-t_n}$$

$$p(w) = \left(\frac{\alpha}{2\pi}\right)^{\frac{m+1}{2}} e^{-\frac{\alpha}{2} w^T w} \propto e^{-\frac{\alpha}{2} w^T w}$$

$$p(w|t) = \frac{p(t|w)\, p(w)}{p(t)} \propto p(t|w)\, p(w)$$

Now, $\vec{w}_{MAP} = \underset{w}{\arg\max}\ p(w|t)$

$$= \underset{w}{\arg\max}\ p(t|w)\, p(w)$$

$$= \underset{w}{\arg\min}\ -\ln p(t|w) - \ln p(w)$$

$$= \underset{w}{\arg\min}\ -\ln \prod_n h_n^{t_n} (1-h_n)^{1-t_n} - \ln e^{-\frac{\alpha}{2} w^T w}$$

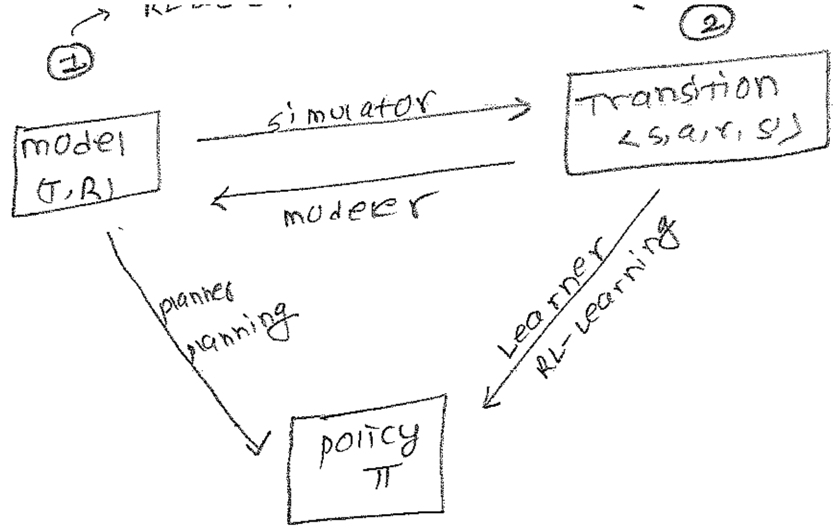
$$= \underset{w}{\arg\min}\ -\sum_n \ln\left(h_n^{t_n} (1-h_n)^{1-t_n}\right) + \frac{\alpha}{2} w^T w$$

$$= \underset{w}{\arg\min}\ \underbrace{-\sum_n \left[t_n \ln h_n + (1-t_n)\ln(1-h_n)\right]}_{E_D(w)} + \underbrace{\frac{\alpha}{2} w^T w}_{E_W(w)}$$

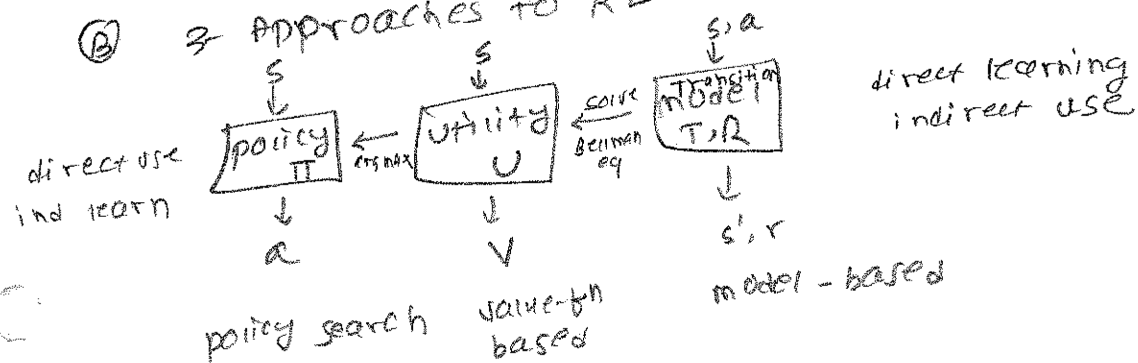
$$\boxed{\vec{w}_{MAP} = \underset{w}{\arg\min}\ E_D(w) + E_W(w)}$$

Ans

(A)



(B) 3 Approaches to RL



(C) Q function

$$U(s) = R(s) + \gamma \max_a \sum_{s'} T(s,a,s')\, U(s')$$ (Bellman eqn) utility is a scalar

change U to Q

$$\pi(s) = \arg\max_a \sum_{s'} T(s,a,s')\, U(s')$$ (policy gives an action)

$$Q(s,a) = R(s) + \gamma \sum_{s'} T(s,a,s') \max_{a'} Q(s',a')$$

quiz ⇒ $U(s) = \max_a Q(s,a)$

⇒ $\pi(s) = \arg\max_a Q(s,a)$

4NB

Example

label

	DOC	words (wi)	class	Total
Train				
3 documents for c1	1	Chinese Beiging Chinese	c1	n1=8
	2	Chinese Chinese Shanghai	c1	
	3	Chinese Macao	c1	
1 document for c2	4	Tokyo Japan Chinese	c2	n2= 3
			?	11 words 6 unique
Test	5			
Total	$\|D\| = \|D1\|+\|D2\| = 3+1 = 4$			

Chinese ++++|
2 Beiging 1
3 Shanghai 1
4 Macao 1
(2–4: ignore)
5 Tokyo 1
6 Japan 1

① vocabulary, V = { Chinese, Beiging, Shanghai, Macao, Tokyo, Japan }

$|V| = 6$

②

category $c1 = c$

- prior $P(c1) = \frac{|D1|}{|D|} = \frac{3}{4}$
- # of words in class c1, $n1 = 8$

conditional probabilities

$p(w_1|c_1) = p(\text{Chinese}|c) = \frac{5+1}{8+6} = \frac{6}{14} = \frac{3}{7}$

n_k: $5 + 1$ → $n_{ki} = n_{11} = 5$, → Laplace smoothing; $8 + 6$ → n_k, $|V|$

$p(w_2|c_1) = p(\text{Beiging}|c) = \frac{1+1}{14} = \frac{2}{7}$

$p(w_3|c_1) = p(\text{Shanghai}|c) = 2/7 = p$

$p(w_2|c_1) = p(\text{Tokyo}|c) = \frac{0+1}{8+6} = \frac{1}{14}$

$p(w_3|c_1) = p(\text{Japan}|c) = \frac{0+1}{8+6} = \frac{1}{14}$

category $c2 = j$ (japan)

- prior $p(c2) = \frac{|D2|}{|D|} = \frac{1}{4}$
- # of words in class c2, $n2 = 3$

$p(w_1|c_2) = p(\text{Chinese}|j) = \frac{1+1}{3+6} = \frac{2}{9}$

$p(w_2|c_2) = p(\text{Tokyo}|j) = \frac{1+1}{3+6} = \frac{2}{9}$

$p(w_3|c_2) = p(\text{Japan}|j) = \frac{1+1}{3+6} = \frac{2}{9}$

③ choosing a class

$$p(c1|x) \propto p(c1) \prod_i p(w_i|c_1) = \underbrace{\frac{3}{4}}_{\text{prior}} \cdot \underbrace{\left(\frac{3}{7}\right)^3}_{\text{Chinese}} \cdot \underbrace{\frac{1}{14}}_{\text{Japan}} \cdot \underbrace{\frac{1}{14}}_{\text{Tokyo}} \cong 0.0003$$

$$p(c2|x) \propto p(c2) \prod_i p(w_i|c_2) = \frac{1}{4} \cdot \left(\frac{2}{9}\right)^3 \cdot \frac{2}{9} \cdot \frac{2}{9} \approx 0.0001$$

$$c^* = \underset{c_k}{\operatorname{argmax}}\ p(c_k) \prod_{j=1}^{n} p(w_j|c_k) = \underset{c_k}{\operatorname{argmax}} \{0.0003, 0.0001\}$$

= Label of 0.0003 = C1