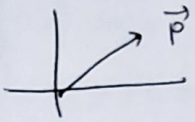
* Essential Maths for ML *

Scaler → only magnitude

vector → magnitude with direction

↳ column vector



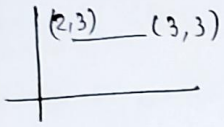
Position vector

$\begin{bmatrix} \\ \end{bmatrix}$ ↓ 1 column (for n dimension → n row) $X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$

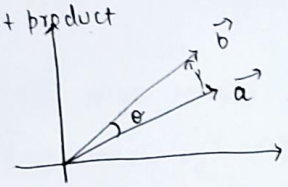
→ $X = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$ $a = 1$ $X + a$ X not in math

in Python (broadcast) → convert in same dim

translation → (2,3) → (3,3) $\begin{bmatrix} 2 \\ 3 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$

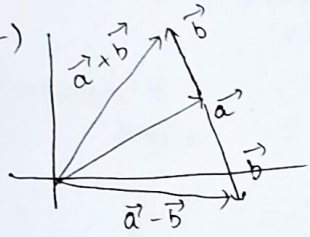


→ Dot product



$\vec{a} \cdot \vec{b} = ab\cos\theta = b(a\cos\theta) = a(b\cos\theta)$

→ (+), (−)



→ equation

$$\begin{bmatrix} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 = b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = b_3 \end{bmatrix} \Rightarrow A(X) = (b)$$

↳ vector → vector

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$ → transformation

Dimension ⇒ $\mathbb{R}^n$

$y = f(x)$

↳ $b = A(x)$ y → assume it transforming vector A

$A: \mathbb{R}^m \to \mathbb{R}^n$

[1 x 3] x [3 x 1] = [1 x 1]

$w: \mathbb{R}^n \to \mathbb{R}$ for Linear classifier

$$\hat{y} = f(x) = w_0 + w_1 x_1 + w_2 x_2 \dots w_n x_n$$

$$y \approx f(x) = w^T x \quad (x^T w) \to \text{not followed}$$

$$\begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{bmatrix}_{4\times 1} \qquad X = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ \end{bmatrix}_{4\times 1500}$$

$$y = \begin{bmatrix} - & - & - & - \end{bmatrix}_{1\times 1500}$$

$$(y - \hat{y}_i)^2 = (y - w^T x)(y - w^T x)$$

$$\Delta w_i = -2 (y - \hat{y}_i)^T \quad = \quad 1\times 1500 \quad 1\times 1500 \quad (1\times 1500)^T = (1500\times 1)$$

$$\Delta w_i = -2 (y - \hat{y})\, x_i^T$$

$$(1\times 1500)\ (500\times 1) \to 1\times 1 \ \checkmark$$

Normalisation

1) Remove Duplicate (Min max scaling)
2) Handle missing value (put 0 or other technique)

Min max Scaling $= \dfrac{x_i - \max}{\max - \min}$ $R \in (-1, 0)$

another way $\dfrac{x_i - \min}{\max - \min}$ $R \in (0, 1)$ ✓

standard $\to \dfrac{x - u}{\sigma}$ usually $(-2, +2)$

↳ Replace with mean value (ignore the missing value) (or make them 0) if n → increase

Preprocessing step in ML:-

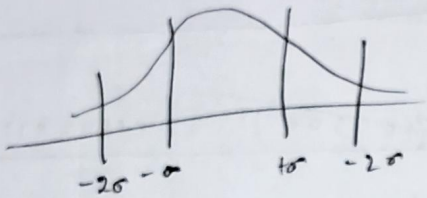
2) Normalisation
   - Min-Max scaling (on feature) → $\frac{x - min}{max - min}$
   - Standard scaling → μ, σ

$$\frac{x - \mu}{\sigma} \rightarrow (-3, 3)$$

2 error we have to compute → both check
- original error → $(y - (\sigma \times \hat{y} + \gamma))^2$
- normalized error → $(y_{norm} - \hat{y}_{norm})^2$

→ we have to handle outlier

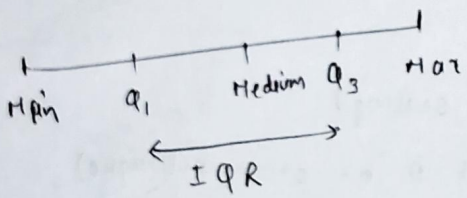
(1)



$$-3 > \frac{x - \mu}{\sigma} > 3$$

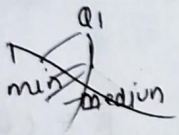
↳ consider outlier

(2)



Inter quantile range = $Q_3 - Q_1$

if value < $Q_1 - 1.5\,IQR$
or value > $Q_3 + 1.5\,IQR$



Percentile → $P[X \leq x)$
How many point is less than x

$$\frac{n}{N} \times 100$$

→ How to handle string.

er size → large → 0
medium → 1
Small → 2

} label encoding (Cordinal data)

(medium - small = large - medi

why not → $(100 - 0)^2$ } → Difference is large

one hot encoding → Sparse Representation

$[1\ 0\ 0]$
$[0\ 1\ 0]$
$[0\ 0\ 1]$

Maths → x^2 $\quad \frac{d}{dx}x^2 = 2x$ ∀ scalar

$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$ $\quad X^TX = [x_1\ x_2\ x_3]\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = x_1^2 + x_2^2 + x_3^2$

$y = X^TX$ ∀ $X \rightarrow$ vector

vector derivative with scaler → vector

$\frac{dy}{dx} = \begin{bmatrix} \frac{\partial y}{\partial x_1} \\ \frac{\partial y}{\partial x_2} \\ \frac{\partial y}{\partial x_3} \end{bmatrix} = \begin{bmatrix} 2x_1 \\ 2x_2 \\ 2x_3 \end{bmatrix} = 2\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = 2X$

(vector → 3 component)

→ $X^TAX = y$

$1\times2\ \ 2\times2\ \ 2\times1$ $\quad y \rightarrow 1\times1$ ∀ → scaler

$X = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix}$ $\quad A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$

$\frac{dy}{dx} = (A + A^T)X$

↳ if A is not symmetric

$2AX \rightarrow$ A is symmetric

→ X^TAX

↳ $\begin{bmatrix} a_{11}x_1 + a_{12}x_2 \\ a_{21}x_1 + a_{22}x_2 \end{bmatrix}$ $\quad 2\times1$

$[x_1\ x_2]$ $\quad 1\times2$ $\quad = a_{11}x_1^2 + a_{12}x_1x_2 + a_{21}x_1x_2 + a_{22}x_2^2$

$$y = a_{11}x_1^2 + (a_{12} + a_{21})x_1x_2 + a_{22}x_2^2$$

$$\frac{\partial y}{\partial x_1} = 2a_{11}x_1 + (a_{12} + a_{21})x_2$$

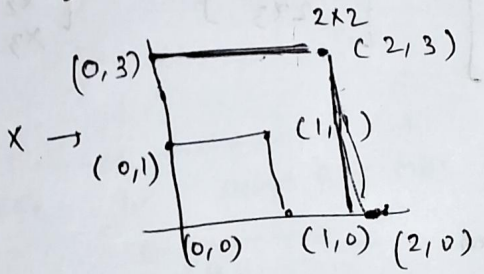
$$\frac{\partial y}{\partial x_2} = (a_{12} + a_{21})x_1 + 2a_{22}x_2$$

$$\begin{bmatrix} \frac{dy}{\partial x_1} \\ \frac{\partial y}{\partial x_2} \end{bmatrix} = 2 \begin{bmatrix} \quad \end{bmatrix} = 2AX$$

Essential math in ML

$$\underset{A}{\begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix}} \underset{X}{\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}} = \underset{2\times4}{\begin{bmatrix} 0 & 0 & 2 & 2 \\ 0 & 3 & 0 & 3 \end{bmatrix}}$$

2×2

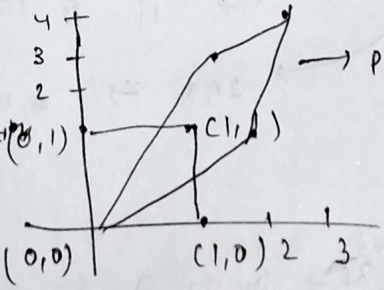


square transformed to rectangle

AX → Translation (if non-dignol is 0)

$$\begin{bmatrix} 2 & 0 \\ 1 & 3 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 2 & 3 \\ 0 & 3 & 1 & 4 \end{bmatrix}$$

Geometric Representation



→ PARALLELogram

The above operation is shear / Rotation operation

The basis (axes change)

Transformation for linear regression

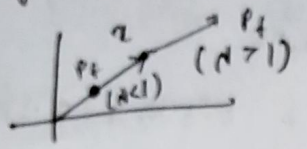
→ Gradient Descent is slow

A transformation

x → eigen vector

λ → eigen value

$Ax = \lambda x$



How to solve → $(A - \lambda I)x = 0$ or $(\lambda I - A)x = 0$

$X = 0$ (Trivial case)

$\det(A - \lambda I)$ → represent Area

$$A = \begin{bmatrix} 1 & 3 \\ 2 & 2 \end{bmatrix} \quad A - \lambda I = \begin{bmatrix} 1-\lambda & 3 \\ 2 & 2-\lambda \end{bmatrix}_{2\times 2} \text{ (so } 2\lambda)$$

$$(1-\lambda)(2-\lambda) - 6 = 0$$

No, for rectangular matrix

$$2 - \lambda - 2\lambda + \lambda^2 - 6 = 0 \Rightarrow \lambda^2 - 3\lambda - 4 = 0$$

$$\lambda = 4, -1$$

$$\begin{bmatrix} 1 & 3 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$x + 3y = -x$

$2x + 2y = -4$

$2x = -3y$

→ $(-3, 2)$

→ collection of vector

$X \Big]_{\mathbb{R}^{m\times n}}$

$$S = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

S_1 → column vector

$[S_1 \; S_2 \; S_3 \; \cdots \; S_m]$

$\mathbb{R}^{n\times m}$ (column as vector)

$\mathbb{R}^{m\times n}$ (row as vector)

$$X^T X \rightarrow \sum_{i=1}^{m} \sum_{j=1}^{n} x_i x_j = x_1 x_2 + x_1 x_3 + x_1 x_3 \cdots x_1 x_n$$

→ Identify the eigen value & vector for this

Reduce vector:

$X^T X$ → Eigen vector are ~~actually~~ corresponding to top eigen value are actually encoding maximum information

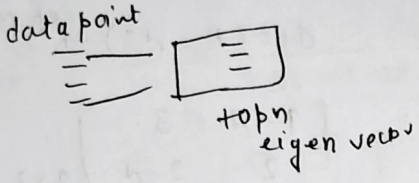
→ Sort d_i value Sort

ex → take top 5 value

$0.4\,d_1$ $0.3\,d_2$ $0.2\,d_3$ $0.2\,d_4$ $0.1\,d_5$

$$X_{new} = 0.4\,d_1 + 0.3\,d_2 + 0.2\,d_3 + 0.1\,d_4$$

compute the loss



Vector Norms → for a vector of n dimension

1's Norm ←

$$L_1 = \sum_{i=1}^{n} |x_i|$$

$$L_2 = \left(\sum_{i=1}^{n} |x_i|^2\right)^{1/2}$$

$$L_\infty = \max_{1 \le i \le n} |x_i|$$

$$L_p = \left(\sum_{i=1}^{n} |x_i|^p\right)$$

pth norm

Norm ≥ 0

$$|x+y| \le |x| + |y|$$

⇒ $A \in \mathbb{R}^{m \times n}$

Vector norm

1) Positive $|x| \ge 0$ (non-negative)

2) Homogeneous $|\alpha x| = \alpha |x|$

3) Triangle Inequality $|x+y| \le |x| + |y|$

Matrix form:

1'st norm

$$\|A\|_1 = \left(\max_{j=1 \dots n} \sum_{i=1}^{m} a_{ij} \right)$$

max from column sum

$$\|A\|_\infty = \left(\max_{j=i \dots m} \sum_{i=1}^{n} a_{ij} \right)$$ → max from row sum.

Frobenium norm of f-norm $\|A\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2}$

$\|A\| \geq 0$ $\quad \alpha \|A\| = \|\alpha A\|$ $\quad \|A+B\| \leq \|A\| + \|B\|$

⇒ X

$u \rightarrow$ mean $\quad A = X - u$ $\quad A^T A \rightarrow$ find eigen value & vector

→ Sort & find top eigen val.

↳ $x_1, x_2 - - x_{10}$

$a_i = x_i - u$

Find the projection of a_i on eign vector?

→ Dot product

K → eigen vector

$$a_{reconst_i} = \sum_{i=1}^{K} w_i x_i$$

$a_{reconst_i}^{update} = a_{reconst_i} + u$

Now find

$$\left\| (x_i - a_{recon_i, update})^2 \right\|$$ ← Make Norm

EIGEN FACES

(Improved) Gradient descent:

→ Momentum based GD
(we accumulate the gradient)

previos formula

$w_{t+1} = w_t - \eta \nabla w_t$

$\eta \rightarrow$ less (slow & stable)

$\eta \rightarrow$ high (fast & unstable)

$v_i \rightarrow$ velocity

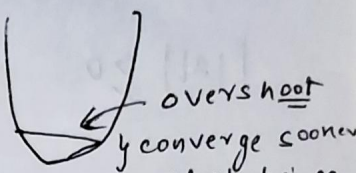
$v_0 = 0$

$v_{t+1} = \gamma v_t + \eta \nabla w_t$

$w_{t+1} = w_t - v_{t+1}$

normally $\gamma = 0.9$

$\Rightarrow (w_{prev} - w_{curr}) << 0.001$ stop



→ Another improved version

NAG → (Nestov Accelerated Gradient)

so, they check if it overshoot if overshoot make some change & don't overshoot

Deep learning
by Dr. Hitesh Khapra
IIT Madras

$v_{t+1} = \gamma v_t + \eta \nabla (w_t - \gamma v_t)$

↓

lookahead position gradient

$\gamma = 0.9$ | $(0.7, 0.8, 0.9 \ldots 1.0)$

$\eta = [0.01, 0.001 \ldots 0.005]$

$w_{t+1} = w_t - v_{t+1}$

$\Rightarrow$

Probability & statics for ~~HE~~ ML

Random variable:

↳ showing result of random experiment

outcome/sample space → all possible outcome

$x = 1 \ldots x = 6$

$P_r(E) = \frac{|E|}{|S|} = \frac{n(E)}{n(S)}$

$\sum_{i=1}^{N} P_r(x_i) = 1$

→ Probability mass fn

$P(X=1) = 1/6$
$P(X=2) = 1/6$

↳ equally likely event

→ N, Z, Q, R, C

$N \subseteq Z \subseteq Q \subseteq R \subseteq C$ → non countable inf...

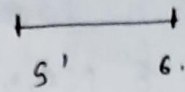
$f(x) = \{ x+1 \mid x \in N \}$ → countable infiwite

→ Let have height

→ continos random experiment.

→ we cannot ennumerate

→ so, use probability density function



$f(x)dx$ → interval

↓ Probability when $dx \to 0$

PDF

function determine the height of person

for → $\int_a^b f(x)\,dx$

CDF = $Pr(X \le x)$

cummulative

$\int_{-\infty}^{\infty} f(x)\,dx = 1$

for discrete

$CDF = \int_{-\infty}^{x} f(x)\,dx$

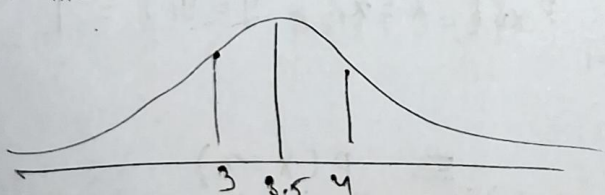
$P_r(X < x_c) = \sum_{i=1}^{i=c} P_r(x_i)$

Excepectation

$E[x] = \sum_{i=N}^{N} x_i \cdot P(x_i) = \frac{1}{6} + \frac{2}{6} + \frac{3}{6} + \frac{4}{6} + \frac{5}{6} + \frac{6}{6} = 3.5$ (mean)

Law of large Number : → if a experiment infinite time the $E[x]$ is happend

centre limit theorem:



$E[(X-\bar{X})]^2 = E[X^2] - (E[X])^2$

↓ variance

$P(A|B) = \frac{P(A \cap B)}{P(B)}$ $\qquad P(B|A) = \frac{P(A|B)\,P(B)}{P(A)}$

$P(A) = \sum_B P(A,B)$ → Marginal prob

$P(B) = \sum_A P(A,B)$

entropy, cross entropy / information

⇒ Joint, conditional and Marginal Distribution

$P_{x,y}(X=x, Y=y)$ $\qquad P_{X,Y}$ → Joint Distribution

↓ PMF

Rule of multiplication A then B $P(A,B) = P(A)\,P(B|A)$

B then A $P(B,A) = P(B)\,P(A|B)$

Rule of addition $\sum_{i=1}^{\infty} P(x_i) = 1$ $\qquad P(A \cap B) = P(A) + P(B)$ − $P(A \text{ and } B)$

↓ OR

$P(A \text{ or } B) = P(A) + P(B)$ A & B are mutually exclusive

⇒ $P(X_1 X_2 X_3) = P(X_1)\,P(X_2|X_1)\,P(X_3|X_1 X_2)$

$= P(X_1)\,\frac{P(X_1 X_2)}{P(X_1)}\,\frac{P(X_1 X_2 X_3)}{P(X_1 X_2)} = P(X_1|X_2 X_3)$

→ $P_{x,y}(X=x, Y=y)$

$\sum_{i=1}^{N} \sum_{j=1}^{M} P_{XY}(X=x_i, Y=y_j) = 1$

$x \sim P(x) \equiv P(X=x)$

↓ sample ↳ probability distribution

$\mu = 0$
$\sigma = 1$

$$P(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$$ PDF

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} P_{x,y}(x,y)\,dx\,dy = 1$$

X \ Y	Y=0	Y=1	Y=2	
X=0	0.2	0.1	0.2	0.5
X=1	0	0.2	0.1	0.3
X=2	0.1	0	0.1	0.2
	0.3	0.3	0.4	1

P(Y=0) P(Y=1) P(Y=2)

$P(0,0) = 0.2$

$P((X=0, Y=0) = 0.2$

$P(Y=0) = P(X=0, Y=0) + P(1,0) + P(2,0)$
$= 0.3$

$$P(Y=y_j) = \sum_{x_i \in X} P(X=x_i,\ Y=y_j)$$

$$P(X=x_j) = \sum_{y_i \in Y} P(X=x_j,\ Y=y_i)$$

conditional in two:

$P(Y \mid X=0)$ → represent row $x=0$

$P(X \mid Y=1)$ → represent column $y=1$

conditional Distribution

$$= \frac{P(X, Y=1)}{P(Y)}$$ → depend on x

$$P(X=1 \mid Y=1) = \frac{P(X=1, Y=1)}{P(Y=1)} = \frac{0.2}{0.3} = \frac{2}{3}$$

for continuous

$P(X+Y=2)$ $P(X \mid Y=1)$

⇒ Information theory → information encoded in distribution
↳ mesure in bits

Suprise $= \log\left(\frac{1}{p}\right)$ $\quad p \approx 0$

$= -\log p$

Entropy = E[suprise)

$= \sum_{x_i \in X}$ Suprise$_i$ P.(surprise$_i$)

$= \sum -\log p_i \times P(-\log p_i)$

entropy $= \sum_{i=1}^{n} -\log p_i \; p_i$ $\quad \} \; E[f(x)] = f[x]$

cross entropy
KL-Divergence →

Jagrachand Jakhar

Entropy:-

$I(x) = \log_2\left(\frac{1}{P(x)}\right)$ → Represent no. of bit

$H(P) = E[I(x)]$

$= E[-\log(P(x))$

$= \sum -\log(P(x))\, P(x)$

$= -\sum_{x \in X} P(x) \log P(x)$

$E[f(x)] \leq f[E(x)]$

Jensen's Inequality

coin Toss

	H	T
P	1/2	1/2
	1	0
	0	1
	↓	
	0	

$\log 0 = 0$ →

Entropy $= -\sum \frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}$

$= 1$

Entropy $= -\sum 0 \log 0 + 1 \log_2 1$

$= 0$

Entropy → calculate randomness

$P(H) = 0.2 \quad P(T) = 0.8$

$\text{Entropy} = -[\,0.2 \log 0.2 + 0.8 \log 0.8\,]$

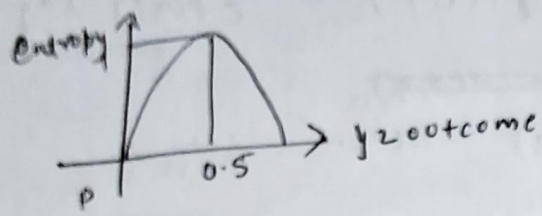
$= -(-2.3 \times 0.2 + 0.8 \times 0.3)$

$= 0.46 + 0.24 = 0.7$

More random $P(H) = 0.4, 0.6$

$1.3 \times 0.4 + 0.7 \times 0.6 = 0.52 + 0.42$

$= 0.94$



→ Maximum entropy is possible when the distribution is uniform

For dice → 1/6

Joint Entropy:- $H(x,y) = -\sum_{x,y} P(x,y) \log P(x,y)$ $\quad H(X|Y)$

$H(X|Y) = \sum_{x,y} P(x,y) \log\left(\frac{P(x,y)}{P(y)}\right)$ ← $\log P(X|Y)$

$H \geq 0$ (non-negative)

→ Cross Entropy

→ Relative Entropy (KL Divergence)

→ $CE = -\sum P(x) \log q(x)$

$P(x)$: True distribution

P, q → distribution

$q(x)$: Estimated / Modeled Distribution

$x, y, \hat{y} \quad \hat{y} = f(x)$ → Loss fn dependent on CE

Logistic Regression

true distribution → y estimated → $\hat{y}$

$CE = -(y \log \hat{y} + (1-y) \log(1-\hat{y}))$

$\frac{1}{1+e^{-x}}$

$x \rightarrow w^T x$

KL Divergence → P, q show different 2 distribution are)

$$D(P||q) = \int P(x) \log \frac{P(x)}{q(x)} dx \quad \text{for continuous}$$

P → true

q → estimated

$$= \sum_{x \in X} P(x_i) \log \frac{P(x_i)}{q(x_i)}$$

$$\Rightarrow -\left(-\sum P(x_i) \log P(x_i)\right) - \sum P(x_i) \log q(x_i)$$

$$= \cancel{H(P) = H(P,q)}$$

$$CE(P,q) - H(P)$$

Mutual Information

$$I(X;Y) = \sum_{Y} \sum_{X} P(X,Y) \log \frac{P(X,Y)}{P(X)\,P(Y)}$$

when independent $I(X;Y) = 0$

(encode mutual Info)

Vocabulary of ML

T, P, E $P_1 \rightarrow E t_1$ $P_2 > P_1 \rightarrow E t_2$ $t_2 > t_1$

(training experience)

Testing phase

Two phase
- Training
- Test

ML algo
- Paramatric
- Non-Paramatric → KNN

Types of ML(training style)
- Supervised → Semi's-supervised
- Unsupervised → Reinforcement learning
  - clustering
    - Kmeans
    - Agglomerative Clustering
    - DBScan (Density base

(KNN → reverse KNN)

$[x_1, x_2 \; - \; - \; x_n]\,[y]$ $\qquad$ $y = f(x_1, x_2 \; - \; x_n)$

L_1 Loss $= \left[\sum y_i - \hat{y}_i \right]$ Minimize over all loss

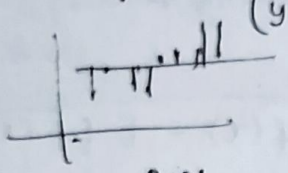
3 Data → Training, Validation → testing → Unseen

→ come from the same distribution

why validation →

→ overfitting

→ underfitting

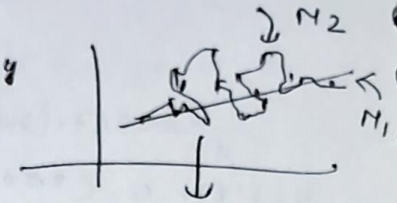


n epoch

= 1m sample m x n sample

=

overfitting

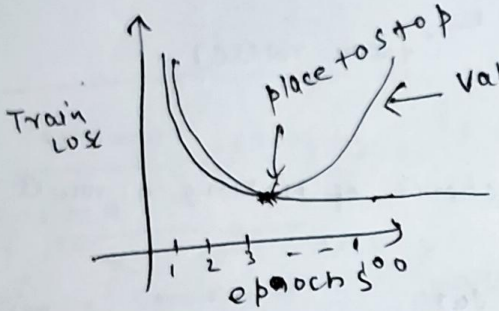


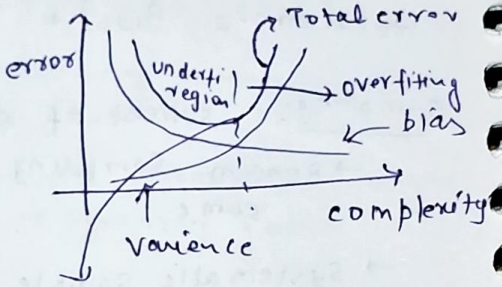
testing Accuracy

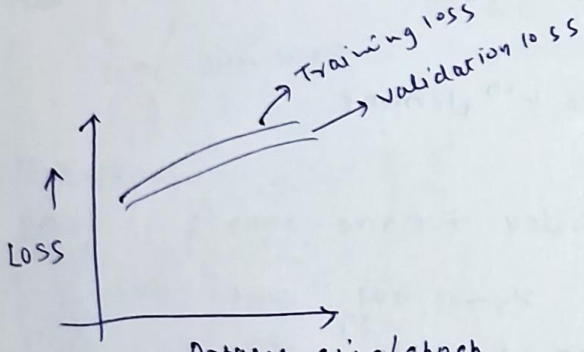
overfitting

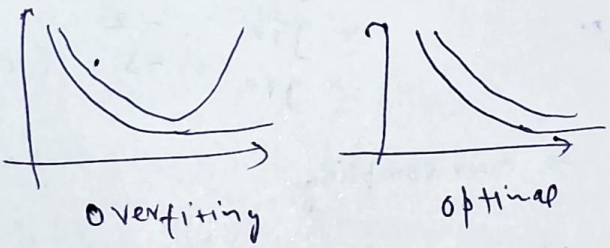
$(N_1) >$ testing Acc (N_2)

$$\text{MSE} = \frac{\sum (y_i - \hat{y}_i)^2}{N}$$









Low- bias & Low variance

Low bias
Low variance

high bias low variance

Low bias High variance

High bias & high variance

Bias vs variance Trade off

Inverse proportional

Total error = Bias error + variance error + irreducible error

↓ does not capture pattern

↓ very sensitive to training data

↓ noise

$y = f(x) + \epsilon$ (True Distribution; ϵ: noise, $\epsilon \approx 0$)

$\Rightarrow \epsilon \rightarrow \mathcal{N}(\mu, \sigma)$

0 1

$$\text{Bias} = f(x) - E[\hat{f}(x)]$$

$$\text{variance} = E[(x-\mu)^2] = E[x^2] - (E(x))^2$$

$$\text{Variance} = E[(f(x) - E[\hat{f}(x)])^2]$$

$$\text{total error} = \text{Bias}^2 + \text{variance} + \sigma^2 \text{ (from noise)}$$

sampling: subset of dataset.

→ Random Sampling / uniform → chance of picking a row is same

→ Systematic Sample → for sorted data

pick k^{th} element

=] 10 category-1

=] 10 − 2

=] 10 − 3

=] 10

→ over sampling

→ repeat minority class to make 990

1000 → sample let have 2 classes

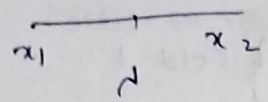
990 10 → bias toward category 1

majority class minority class

Undersampling : undersample the majority class

SMOTE:- Synthetic Minority Oversampling Technique

$$\lambda x_1 + (1-\lambda) x_2 = 0$$



→ Take point from minority class & create new point

process:

take a point & check nearest and find new point using different λ

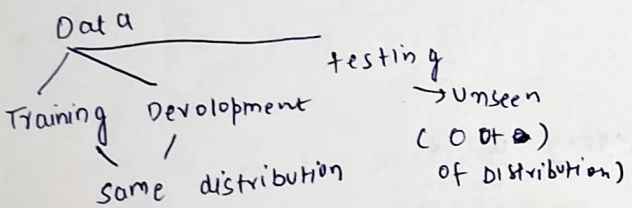
⇒ Label has 3 class

c1
c2
c3

how many model take:

1st Model (binary) → c_1 (not c_1)
2nd Model → c_2, Not c_2
3rd Model → c_3, Not c_3

⇒ Validation Strategies ⇒ to validate model

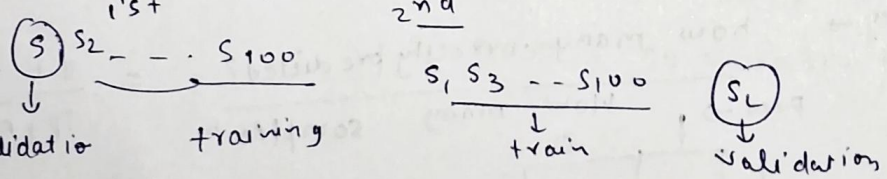


Data augumentation in text
→ synonym replace

Technique:

→ LOOV :- (leave one out validation)

Let have 100 sample

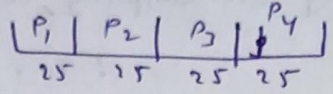


→ 100 train & test and take average

K-fold cross validation:-

too sample → divide in k part

Let's 4-fold →



First time → P_1 → test → 25
→ $P_2 P_3 P_4$ → train → 75
} Fold-1

Fold two → P_2 → test
P_1, P_3, P_4 → train

fold 3 → P_3 → test
$P_1 P_2 P_4$ → train

fold 4 → _____

→ Take average of all accuracy

dis

—] c_1 N/K
—] c_2 N/K
—] c_3
—
c_k

gt data is in order

fold → c_1
training → c_2 - - c_K
⇒ accuracy = 0

Stratified Validation:

→ Select sample from category wise

c_1 [ ] → strata

c_2 [ ] strata - - c_k [ ] → strata

Evaluation Technique:

→ Accuracy → how many correctly predicted? / how many samples? = $\frac{TP + TN}{TP + FP + FN + TN}$

Gold (actual) \ Pred →	1	0
1	TP	FN
0	FP	TN

True positive → TP
False Negative → FN
False Positive → FP
True Negative → TN

Total positive sample = $TP + FN$

Total negative sample = $FP + TN$

Total model predict positive = $TP + FP$

$$\text{Precision } (P) = \frac{TP}{TP + FP} \qquad \text{Recall } (R) = \frac{TP}{TP + FN}$$

$$F_1 \text{ score} = \frac{2}{\frac{1}{P} + \frac{1}{R}} = \frac{2PR}{P+R}$$

$$F_\beta \text{ score} = (\beta+1)\frac{PR}{P + \beta^2 R}$$ } weight for recall more

For Regression

MSE → $\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$ } ⇒ Mean Residual score

Sum of squared err = $\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$

Mean Absolute error = $\frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$

→ $R^2 = 1 - \frac{SSE}{SST}$

$= 1 - \frac{\text{residual}}{\text{sum of average error}}$

$\bar{y} = \frac{1}{n}\sum_{i=1}^{n}\hat{y}_i$

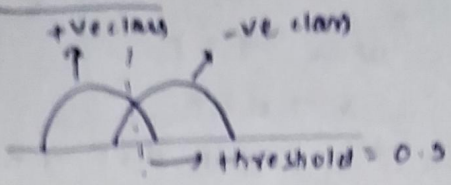
$SST = \sum_{i=1}^{n}(y_i - \bar{y})^2$

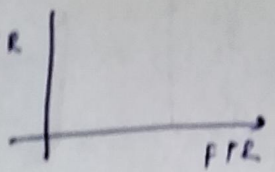
$0 \le R^2 \le 1$ (very close to 1 is good)

$R^2 \approx 1$ → variability

$R^2 \approx 0$ → no variability

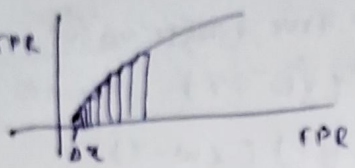
ROC curve





(for different threshold)

AUC



$$= \lim_{\Delta x \to 0} h\,\Delta x$$

TPR & FPR ⟹ atleast 100 points (bins)

Supervised Learning

→ require a dataset — feature / data
label / classes / target / value
classification — Regression

i^{th} sample → $(\vec{x}_i, y_i)$

vector / collection of feature

Linear Regression :-

⇒ Mini Batch : medium update

Batch size $K = 16$ (normally take in in power of 2)

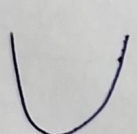
for batch : error is

$$L = \frac{1}{K} \sum_{i=1}^{K} (y - \hat{y})^2$$

$$\Delta w = 2 \times \frac{1}{2} \sum_{i=1}^{K} (y_i - \hat{y}) x_i$$

Another way:

$$\frac{\partial L}{\partial w} = 0$$



Normal equation

$$\hat{y} = w^T x$$

$$\hat{y}_1 = w^T x_1$$

$$\hat{y}_2 = w^T x_2$$

$$\hat{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

$$\hat{y} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_n \end{bmatrix} = \begin{bmatrix} - w^T x_1 - \\ - w^T x_2 - \\ \vdots \\ - w^T x_n - \end{bmatrix}$$

$\rightarrow w_0 x_0 + w_1 x_1 + w_2 x_2 - - w_m x_m$

$w \rightarrow d \times 1$

$X \rightarrow d \times n$

$n \times d \quad d \times 1 \rightarrow n \times 1$

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

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

$\frac{\partial}{\partial w} w^T a = \frac{\partial a^T w}{\partial w} = a$

$\frac{\partial L}{\partial w} = 0$

$\hat{y} = Xw$

$L = (Xw - Y)^2$ → true target value

$L = (Xw - Y)^T (Xw - Y)$

$= ((Xw)^T - Y^T)(Xw - Y)$

$= (w^T X^T - Y^T)(Xw - Y)$

$= w^T X^T X w - w^T X^T Y - Y^T X w + Y^T Y$

$\frac{\partial L}{\partial w} = 0$

$w = (X^T X)^{-1} X^T Y$

$\Rightarrow \frac{\partial}{\partial w}(w^T X^T X w - w^T X^T Y - Y^T X w + Y^T Y) = 0$

$\Rightarrow \frac{\partial}{\partial w}(Y^T X w) = \frac{\partial}{\partial w}(X^T Y)^T w$

$= X^T Y$

$\Rightarrow \frac{\partial}{\partial w} w^T X^T X w$

$= w^T \frac{\partial}{\partial w} X^T X w + X^T X w \frac{\partial}{\partial w} w^T$

$= w^T X^T X + X^T X w I$

$= w^T X^T X + X^T X w$

If w & a has same dimension)

$w = [w_1, w_2, w_3]$

$a = [a_1, a_2, a_3]$

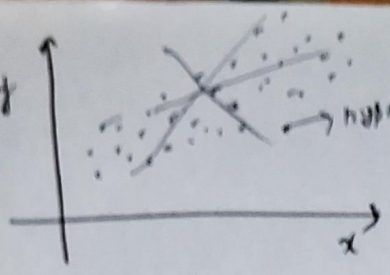
$w^T a = w_1 a_1 + w_2 a_2 + w_3 a_3$

$\frac{\partial x}{\partial w} = \begin{bmatrix} \frac{\partial x}{\partial w_1} \\ \frac{\partial x}{\partial w_2} \\ \vdots \\ \frac{\partial x}{\partial w_d} \end{bmatrix}$

$\frac{\partial w^T a}{\partial w} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = a^T$

$\frac{\partial a^T w}{\partial w} = a^T$

$\frac{\partial}{\partial w} w^T X^T Y = X^T Y$



$\hat{y} = w_0 + w_1 x$

↑ parameter (infinite)

hypothesis

2d → line
3d → plane
>3 ⇒ hyperplane

⇒ hypothesis is a function which your algo consider while creating a model

training data → Algo → h(x) → Prediction

unseen sample → Prediction

⇒ Hypothesis space :-

→ Best fit line (Mean squared error) → Regression

⇒ $\hat{y} = w_0 + w_1 x$ (x, y) ⇒ $L = (y - \hat{y})^2$

⇒ (L_1, y_1, L_1) ⇒ if plot eclispoid (parabola)

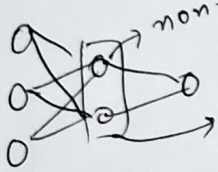
⇒ if in 2D it get elipse

→ contours

~ in contour line gradient are same

Inductive Bias :

Bias → set of assumption



Inductive Bias → generalise → predicting lable discrete o/p on unseen ex.

⇒ occam's razor → choose the simplest hypothesis