3/12/2022

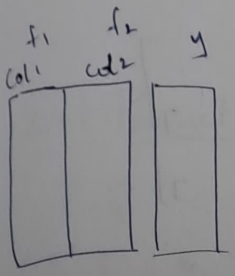
Maths

→ dimensions / features / Columns

→ datapoint / querypoint / Row

Data
- Quantitative
  - Numerical (discrete)
  - continuous (IR)
- Qualitative

It is impossible to visualize more dimensions. This is the reason we go with Mathematical functions



to represent it in math

$$D = \begin{bmatrix} f_1, f_2, \ldots, y_1 \\ f_1, f_2, y_2 \end{bmatrix}$$

no. of rows

$n \times d$ → no. of col's

$$D_n = \{(x_i, y_i)_{i=1}^{n} \mid x_i \in \mathbb{R}^{d-1}, \; y \in \{M, F\}\}$$

Matrix

↑ such that

↳ Classification

$$D_n = \{(x_i, y_i)_{i=1}^{n} \mid x_i \in \mathbb{R}^{d-1}, \; y \in \mathbb{R}^1\} \rightarrow \text{Regression}$$

In maths

$y = mx + c$ → line Equation

3D → plane

4D

:

D

} hyperplane

Circle — Square

2D → ○ □

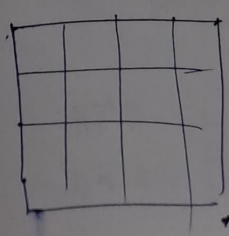
Sphere — Cube

3D → ◯ ▢

≥ 4D → Hyper sphere — Hyper cube

5/12/22

- Representation of a Dataframe (Using matrix)

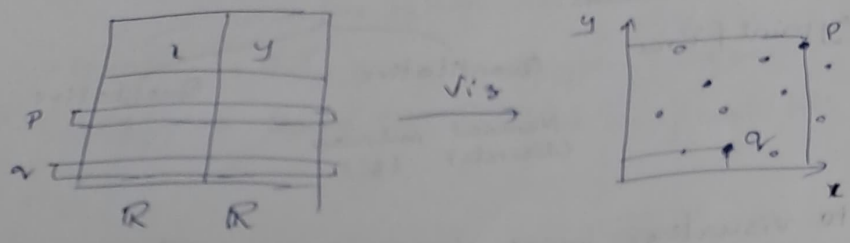


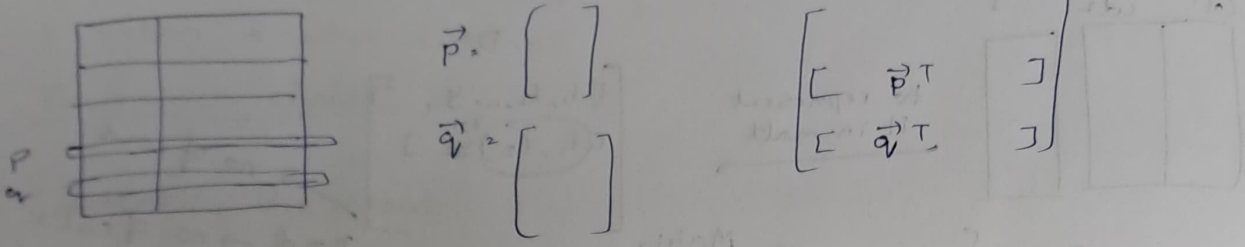
$n \times d$

$$= \begin{bmatrix} \cdot \\ \cdot \\ \cdot \end{bmatrix}$$

$n \times d$

• Representation of a Row/ data point (Using column vector)





→ DF in mathr → Row in mathr

↓ ↓

Matrix Column Matrix (Vectors)

↓

Then converted into row vectors in matrix.

Mathematical Operations on Vectors (i.e., Column Vectors)

$$\vec{P} = \begin{bmatrix} P_1 \\ P_2 \end{bmatrix}_{2*1} \qquad \vec{q} = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix}_{2*1}$$

⇒ Addition

$$\vec{P} + \vec{q} = \begin{bmatrix} P_1 + q_1 \\ P_2 + q_2 \end{bmatrix}$$

* $\vec{P} = \begin{bmatrix} 150 \\ 50 \end{bmatrix}$ $\vec{a} = \begin{bmatrix} 180 \\ 80 \end{bmatrix}$

$$\vec{P} + \vec{q} = \begin{bmatrix} 330 \\ 130 \end{bmatrix}_{2*1}$$

⇒ Multiplication (dot product only)

$$\vec{P} = \begin{bmatrix} P_1 \\ P_2 \end{bmatrix} \cdot \vec{q} = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix}$$

$$\vec{P} \cdot \vec{q} = P_1 \times q_1 + P_2 \times q_2$$

Vector Algebra

$$\vec{p} = \begin{bmatrix} p_1 \\ p_1 \end{bmatrix}_{2\times 1} \qquad \vec{q} = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix}_{2\times 1}$$

Addition

$\vec{p} + \vec{q}$

$= \begin{bmatrix} p_1 + q_1 \\ p_2 + q_2 \end{bmatrix}$

Vector

Subtraction

$\vec{p} - \vec{q}$

$= \begin{bmatrix} p_1 - q_1 \\ p_2 - q_2 \end{bmatrix}$

Vector

Multiplication

$\vec{p} \cdot \vec{q}$

$= p_1 * q_1 + p_2 * q_2$

Scalar.

Dot Product

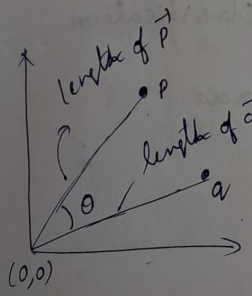
$\vec{p} \cdot \vec{q} =$

$p_1 \times q_1 + p_2 \times q_2 + \cdots\cdots p_d \times q_d$

$$\begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_d \end{bmatrix} \cdot \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_d \end{bmatrix}$$

$$\vec{p} \cdot \vec{q} = \sum_{i=1}^{d} p_i \times q_i \longrightarrow (1)$$

$$[p_1, p_2 \cdots\cdots p_d] \cdot \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_d \end{bmatrix} \qquad \vec{p} \cdot \vec{q} = \vec{p}^{\,T} \times \vec{q} \longrightarrow (2)$$



$$\vec{p} \cdot \vec{q} = \|\vec{p}\| * \|\vec{q}\| * \cos\theta_{\vec{p}\cdot\vec{q}} \longrightarrow (3)$$

$\|\vec{p}\|$: length of vector, i.e., magnitude of vector / norm of vector

$\theta_{\vec{p}\cdot\vec{q}}$: Angle b/w $\vec{p}$ & $\vec{q}$

Question:

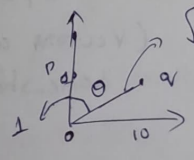
$\vec{p} \cdot \vec{q} = 1$

$$\vec{p} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \qquad \vec{q} = \begin{bmatrix} 10 \\ 1 \end{bmatrix}$$

$$\therefore \vec{p} \cdot \vec{q} = \|\vec{p}\| * \|\vec{q}\| * \cos\theta_{\vec{p}\cdot\vec{q}}$$

$$1 = \|\vec{p}\| * \|\vec{q}\| * \cos\theta_{\vec{p}\cdot\vec{q}}$$

$$\cos\theta_{\vec{p}\cdot\vec{q}} = \frac{1}{\|\vec{p}\| * \|\vec{q}\|}$$



$\|\vec{p}\| = 1$

$\|\vec{q}\| = \sqrt{1+100} = \sqrt{101}$

$$\Rightarrow \theta = \cos^{-1}\left[\frac{1}{\sqrt{101}}\right]$$

$= \cos^{-1}(0.099) = 1.471$

$= 84°$

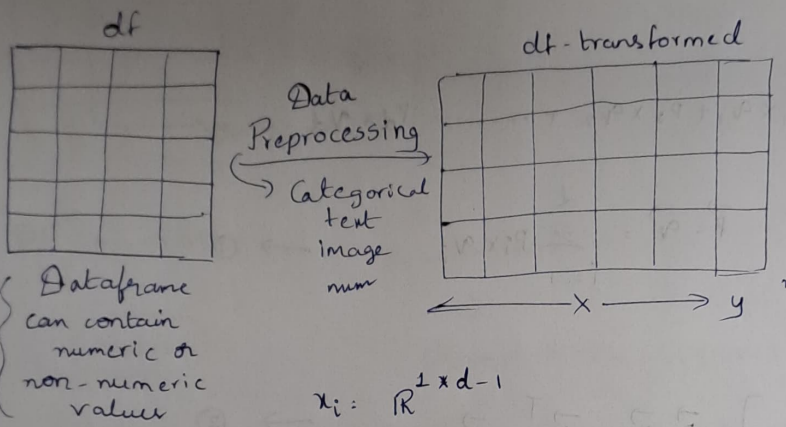
$$\|\vec{p}\| = \sqrt{\sum_{i=1}^{d} p_i^2}$$

$$\|\vec{q}\| = \sqrt{\sum_{i=1}^{d} q_i^2}$$

** To get inverse of cos from numpy it is np.arccos( )

$$\Rightarrow \theta_{\vec{p},\vec{q}} = \cos^{-1}\left[\frac{\sum_{i=1}^{d} p_i * q_i}{\sqrt{\sum p_i^2} * \sqrt{\sum q_i^2}}\right]$$

$$\Rightarrow$$



Dataframe can contain numeric or non-numeric values

$D_n = \mathbb{R}^{n*d} = \{(x_i, y_i)_{i=1}^{n} \mid x_i \in \mathbb{R}^d, y_i \in \mathbb{R}$ or $y_i \in \{0,1\}\}$

$n * d$

$x_i = \mathbb{R}^{1*d-1}$

$y_i = \mathbb{R}^1$ or $\{0,1\}$ or $\{0,1,2,3.....9\}$

Regression | binary Classification | Multiclass Classification

Column Vector Representation of a Datapoint or row

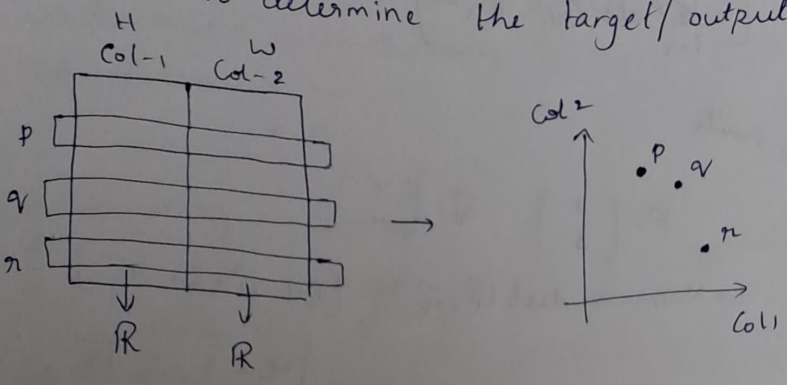
$$x_i = \begin{bmatrix} & \\ & \end{bmatrix}$$

Column Vector (By Default)

Vector Operation (Vectors are of same shape)
- Addition → component wise operation → Vector
- Subtraction → " → Vector
- Dot product → " → Scalar

** To get dot product from numpy
np.dot (a, b)

understand the similarities b/w datapoints are important to determine the target/output variable.

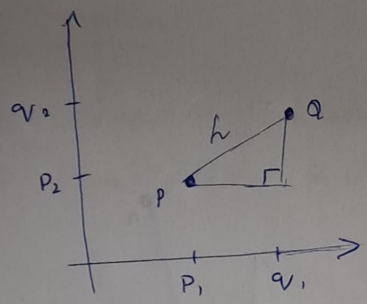


$$Sim(\vec{p}, \vec{q}) > Sim(\vec{p}, \vec{r})$$

Similarity b/w two points/vectors can be computed by looking into the distance between the two points

$$\uparrow\downarrow \text{Similarity} \propto \frac{1}{\text{Distance}} \downarrow\uparrow$$

Distance between the two vectors:



$$h = \sqrt{(side-1)^2 + (side-2)^2}$$

$$dist(\vec{p}, \vec{q}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2}$$

$$\vec{p} = \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_d \end{bmatrix} \qquad \vec{q} = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_d \end{bmatrix}$$

$$dist(\vec{p}, \vec{q}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{d} (q_i - p_i)^2}$$

$$\boxed{dist(\vec{p}, \vec{q}) = \left[\sum_{i=1}^{d} (q_i - p_i)^2\right]^{1/2}}$$

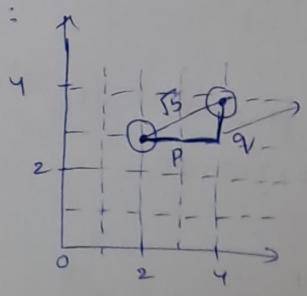
↑ Euclidean Distance (i.e., Shortest distance b/w two vectors)

or

L2 Norm.

$$dist(\vec{p}, \vec{q}) = \left[\sum_{i=1}^{d} abs(q_i - p_i)^1 \right]^{1/1}$$ ← absolute

Eg:



$$\vec{p} = \begin{bmatrix} 2 \\ 3 \end{bmatrix} \quad \vec{q} = \begin{bmatrix} 4 \\ 4 \end{bmatrix}$$

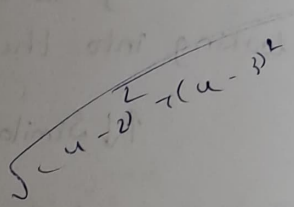
L2 Norm → $dist(\vec{p}, \vec{q}) = \left[(4-2)^2 + (4-3)^2 \right]^{1/2}$

$= \sqrt{4+1} = \sqrt{5}$

L1 Norm → $dist(\vec{p}, \vec{q}) = \left[abs(4-2)^1 + abs(4-3)^1 \right]^{1/1}$

Manhattan distance $= 2 + 1 = 3$

Lp Norm:



if p = 1, L_1 norm / Manhatten distance

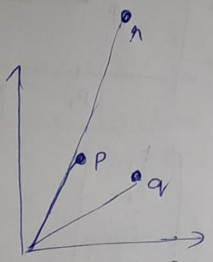
if p = 2, L_2 norm / Eucledian distance

L_p Norm $\left[\sum_{i=1}^{d} abs(a_i - b_i)^p \right]^{1/p}$

↓

Minkowski distance

Angular distance:



Distance ↑↓ ⇒ Sim ↓↑

similarity $(\vec{p}, \vec{q})$ < sim $(\vec{p}, \vec{r})$

Angular distance $(\vec{p}, \vec{q})$ > Angular distance < $(\vec{p}, \vec{r})$

Cosine Similarity:

$$\vec{p} \cdot \vec{q} = \|\vec{p}\| \|\vec{q}\| \cdot \cos\theta_{\vec{p}, \vec{q}}$$

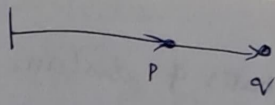
$$\cos\theta_{\vec{p}, \vec{q}} = \frac{\vec{p} \cdot \vec{q}}{\|\vec{p}\| \|\vec{q}\|}$$ ← Cosine Similarity

$-1 \leq \cos\theta \leq +1$

Cosine distance = 1 − Cosine Similarity

⟹ $\cos\theta(\vec{p}, \vec{q}) = -1$

$\cos\theta$ { max = 1, min = −1, 0



angular_distance $(\vec{p}, \vec{q}) \downarrow \Rightarrow sim(\vec{p}, \vec{q}) \uparrow$

Cos distance = 1 − Cos Similarity

Cos_distance = 1 − 1

= 0

Cos_Similarity : Cos θ

= Cos 180°

= −1

Cos distance = 1 − (−1) = 2

Cos_dist	Cos_sim	
0	1	→ 0° angle or 360° angle
2	−1	→ 180°
1	0	→ 90° & 270° angle

K → No. of nearest neighbours

KNN Algorithm:

Type - Supervised Learning

Task - Classification & Regression.

① Given → $D_n = \{ (x_i, y_i)_{i=1}^{n} \mid x_i \in \mathbb{R}^{d-1}, y_i \in \{0, 1\} \}$

$x_q \in \mathbb{R}^{d-1}$

② Decide - Value of k

- Distance Metrics → Euclidian
  → Manhattan
  → Minkowski
  → Angular

distances = [ ]

③ for datapoint in D_n :

- Calculate the distance b/w x_q and data point
- distances.append ((datapoint, distance))

distances = [(dp1, d1), (dp2, d2), ... (dpn, dn)]

④ Sort the distances list on the basis of distance d_i i.e., i from 1 to n

distances : [(dP1s, d1s), (dP3, d3) .....]

⑤ KNNPts : distances [: K]

⑥
```
count_pos = 0
count_neg = 0
for pt in KNNpts :
    if pt['y'] == 1 :
        count_pos += 1
    else :
        count_neg += 1
```

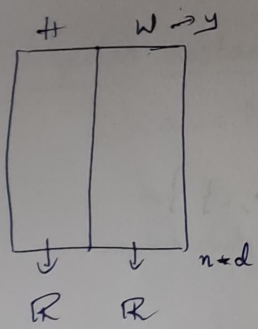
⑦
```
if count_pos > count_neg :
    print ['y_q = 1']
else :
    print ['y_q = 0']
```

Issue with KNN Algorithm:

Time & Space complexity for KNN is very high.

⇒ Mode.

KNN for Regression:



① Given

$D_n = \mathbb{R}^{n*d}$

$D_n = \{(x_i, y_i)_{i=1}^{n} \mid x_i \in \mathbb{R}^{d-1}, y \in \mathbb{R}\}$

Know the query point $x_q \in \mathbb{R}^{d-1}$

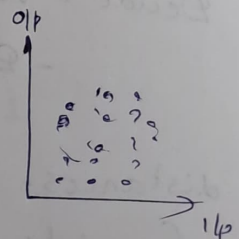
② Decide → k (by default = 5)

Distance (by default = Euclidean

③ distances = [ ]

④ for x_i in D_n :

- Calculate the distance b/w x_i & x_q
- distances.append((d(x_i, x_q), y_i))



⑤ Sort the distances list based on $d(x_i, x_q)$

⑥ KNN_pts . distances [: K]

Mean/Median

(2)

```
sum = 0
for pt in KNNpts :
    sum = sum + pt[y_i]
avg = sum / K
return avg ← y_q
```

⇒ Hyperparameter in KNN ?

K & distance

↑

# of nearest neighbours

⇒ Explain Overfitting & Underfitting in KNN ?

K ↓ → Overfitting / high variance / Low Bias

K ↑ → Underfitting / high bias / low variance

In KNN underfit & Overfit can be controlled by changing the value of K.

High bias — High variance

B U V O

↳ Overfit

↳ Underfit

⇒ How do you detect Overfitting / Underfitting ?

⇒ Can you explain the impact of outliers on KNN models ?

⇒ Can you explain the impact of K on KNN models ?

Same solution i.e., Overfit & Underfit

K ↓ K ↑

Min value of K = 1

Max value of K = n / total no. of datapoints in given dataset.

⇒ What is the space & Complexity of KNN ?

⇒ Why do we call KNN as lazy learner ?

Since there is no learning in KNN algorithm. It simply remembers the data

-At prediction time it performs the distance calculation, computes the nearest neighbours & do the prediction for a query point.

Data → [Algo] → Model

x_q → [Model] → $\hat{y}_q$

KNN

Data → [Algorithm] → Nothing but remember the data

x_q → [Algo] → $\hat{y}_q$

↳ distance Calculation

↳ Sorting

⇒ Is K-NN a parametric Algo or a Non-parametric Algorithm

Non-Parametric

⇒ What is the difference b/w parameter & hyperparameters ?

DT Algo → ID3 Algorithm (10%)

DT Algo → C4.5 Algo

Threshold - Acceptable value

↓

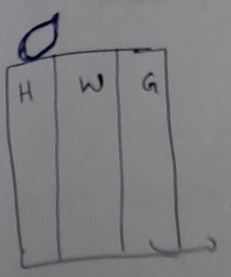
(10-20% larger than max value)

Naive bayes Algo → MAP (Max A Posteriori)

KNN Algo → lazy learner.

Naive Bayes Algorithm: (Maximum A posteriori)

(Mainly for Classification task)



Supervised learning
Target variable
Classification

Different approaches
→ Distance Based Approach
→ Boundary Based Approach
→ Probabilistic " "
→ Rule Based Approach
→ Ensemble " "
→ Deep learning " "

Task: Given x_q find y_q

$P(y_q = \text{'M'} \mid x_q(h,w))$, $P(y_q = \text{'F'} \mid x_q(h,w))$

① Identify target variable
② Task
③ Evaluation Matrix
④ Algorithms

* Given $D_n = \{(x_i, y_i)_{i=1}^{n} \mid x_i \in \mathbb{R}^{d-1}, y_i \in \{C_1, C_2, \ldots C_k\}\}$

$x_q =$

Task → find y_q

How? → Given x_q, compute the prob of x_q belonging to each of the classes whichever is maximum, assign that class to x_q

i.e., $P(C_1|x_q)$, $P(C_2|x_q)$ ---- $P(C_k|x_q)$

$y_q = \arg\max \{P(C_i|x_q)\}$ ~~$P(C_2|x_q) \ldots P(C_k|x_q)$~~

$C_1, C_2, \ldots C_k$ ↳ Posterior probability $\frac{P(C_i)}{P(x_q)}$

From Bayes theorem: likelyhood, prior

$$P(C_i|x_q) = \frac{P(x_q|C_i) * P(C_i)}{P(x_q)}$$

↳ Marginal

Probability → Study of Uncertainity

Probability:

@. Random Experiment →

an experiment for which exact outcome is not known

for Eg: Tossing a Coin, Rolling a dice

$\{H, T\}$ $\{1, 2, 3, 4, 5, 6\}$

⇒ (b) {Sample Space} → It is a set of all the possible outcomes for a Random Experiment

(c) {Event} → Subset of a Sample space

⇒

For Eg:

R.E → Uncertain experiment

- Rolling a dice

S.S → {1, 2, 3, 4, 5, 6}

↑ set of all the possible outcome

Event → Subset of a Sample Space

E_1 - a prime no. {2, 3, 5}

$P(E) = \frac{3}{6} = 50\%$ (or) 0.5

Conditional Probability:

⇒ $P(A/B) = \frac{P(A \cap B)}{P(B)}$

Compulsory Event is the subset of Sample Space that why probability is $0 \leq P(E) \leq 1$

⇒ Axioms of probability

$P(SS) = 1$

Marginal Probability:

$P(A)$ ↓ A

$P(B)$ ↓ B

Frequency table:

Temperature:	Yes	No	
Hot	2	2	4
Mild	4	2	6
Cool	3	1	4
	9	5	

Joint Probability:

$P(A \cap B)$

Outlook	Yes	No	
Sunny			
Rain	3	2	5

Wind:

	Yes	No	
Weak	2	6	8
Strong	3	3	6
	5	9	

P(Playing Tennis & wind strong)

$\frac{3/14}{5/14} = \frac{3}{14}$

P(Play tennis = 'No' | Outlook = Rainy)

$$= \frac{P(\text{'No'} \cap \text{'Rainy'})}{P(\text{Rainy})} = \frac{2/14}{5/14} = \frac{2}{5}$$

R.E: tossing a coin & Rolling a dice

S.S: {(H,1), (H,2), (H,3), (H,4), (H,5), (H,6)
(T,1), (T2), (T3), (T4), (T5), (T6)}

Event: What is probability of getting an even number on the dice

= {(H 2), (H 4), (H 6), (T 2), (T4), (T6)}

Probability: $\frac{6}{12} = \frac{1}{2} = 0.5$ or 50% (chance)

$$P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{3/12}{6/12} = \frac{3}{6} = 50\% \text{ chance (or) } 0.5$$

Given that 'B' has already happen

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

$$P(A \cap B) = P(B|A) P(A)$$

Bayes Theorem:

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

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

Independent Events:

$$P(A|B) = P(A)$$

$$P(B|A) = P(B)$$

$$P(A \cap B) = P(B|A) \cdot P(A)$$

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$

Similarly,

$$P(A \cap B) = P(A|B) * P(B)$$
$$= P(A) * P(B)$$

More than 1 event (A & B)

- Marginal: P(A), P(B)
- Joint: P(A∩B) or P(B∩A) } Both are same
- Conditional: P(A|B), P(B|A)

$\Rightarrow$ $\Rightarrow$ $P(C_1 = 0 | x_q)$ $\quad$ $P(C_2 = 1 | x_q)$

max

$\Rightarrow$ $\max \{ P(C_1 | x_q) , P(C_2 | x_q) \}$

$\max \{ P(x_q | C_1) * P(C_1) , P(x_q | C_2) * P(C_2) \}$

$\max \{ P(f_1 \cap \ldots f_{d-1} \cap C_1) , P(f_1 \cap f_2 \ldots f_{d-1} \cap C_2) \}$

$\hat{y}_q = a$

For Tennis data set:

$P(C_1 = Yes | x_q) =$

$\Rightarrow$

$\Rightarrow$

$\Rightarrow$ Colour = White

Texture = Smooth

Sound = Woof

$P(C_1 = Dog | x_q)$

$\downarrow$

$P(x_q | Dog) * P(Dog)$

$\downarrow$

$P(x_q \cap Dog)$

$\downarrow$

$P(White \cap Smooth \cap Woof \cap Dog)$

$P(White | Smooth \cap Wolf \cap Dog) *$

$P(Smooth | Woof \cap Dog) *$

$P(Woof | Dog) * P(Dog)$

$P(C_2 = Cat | x_q)$

$\downarrow$

$P(x_q | Cat) * P(Cat)$

$\downarrow$

$P(x_q \cap Cat)$

$\downarrow$

2q contains 3 independent features

$P(\text{White}|\text{Dog}) * P(\text{Smooth}|\text{Dog}) * P(\text{Wolf}|\text{Dog}) * P(\text{Dog})$ | $P(\text{White}|\text{Cat}) * P(\text{Smooth}|\text{Cat}) * P(\text{Wolf}|\text{Cat}) * P(\text{Cat})$

Sound	Dog	Cat	
Woof	3	1	4
Purr	1	3	4
	4	4	

Texture	Dog	Cat	
Furry	2	3	5
Smooth	2	1	3
	4	4	

Colour	Dog	Cat	
White	2	1	3
Black	2	3	5
	4	4	

$$\Rightarrow \frac{P(\text{White} \cap \text{Dog})}{P(\text{Dog})} * \frac{P(S \cap D)}{P(D)} * \frac{P(W \cap D)}{P(D)} * P(D)$$

$$\Rightarrow \frac{2/8}{4/8} * \frac{2/8}{4/8} * \frac{3/8}{4/8} * \frac{4}{8}$$

$$\Rightarrow \frac{1}{2} * \frac{1}{2} * \frac{3}{4} * \frac{1}{2}$$

$$\Rightarrow \frac{3}{32} = 0.0937 \quad > \quad 0.0078$$

$$\Rightarrow \frac{1/8}{4/8} * \frac{1/8}{4/8} * \frac{1/8}{4/8} * \frac{4}{8}$$

$$\Rightarrow \frac{1}{4} * \frac{1}{4} * \frac{1}{4} * \frac{1}{2}$$

$$\Rightarrow \frac{1}{128} = 0.0078$$

the predicted class is Dog

Issue → Zero probability Issue

If a feature in query point contain an unseen value {in historical data}

In this scenario NB fails provide the output

* float point precision → Numerical Instability

To bring out from Numerical Instability,

log transformation

$\log(a * b) = \log a + \log b$

$$\hat{y}_q = \underset{i=\{1,\dots,k\}}{\text{argmax}} \left\{ \log\left(P(C_i) * \prod_{j=1}^{d} P(f_j|C_i)\right)\right\}$$

$$\hat{y}_{qv} = \underset{i=[1,\dots,k]}{\arg\max} \left\{ \log(P(C_i)) + \log\left[\prod_{j=1}^{d} (P(f_j)|C_i) \right] \right\}$$

$$\log \prod_{j=1}^{d} b_j = \log(b_1 * b_2 * b_3 * \dots * b_d)$$

- $x, y \rightarrow$ NB Algo $\rightarrow$ Model
  - freq table of f_i vs y ← if f_i is categorical
  - $f \rightleftarrows \mu_y, \sigma_y$ ← if f_i is num & Normal dis

$x_q \rightarrow y_q = \text{argmax}\{\dots\} \rightarrow \hat{y}_{qv}$

→ Zero Probability

Sol$_n$: Apply Regularization → Laplace Smoothing

for Eg : $P(\text{Brown}|\text{Dog}) = 0 + \text{some v.v.v small value}$

$P(\text{Brown}|\text{cat}) = 0 + \text{"}$

$$\Rightarrow P(\underset{A}{B}|\underset{B}{D}) = \frac{0 + \alpha}{n+k+\alpha} \qquad P(f_j|C_i)$$

$n = P(\text{Dog})$

$k = \#$ of unique value in A

i.e., $k = 2$

$$\Rightarrow P(B|C) = \frac{0+\alpha}{n+k+\alpha} \qquad m = P(\text{Cat}), \quad k = 2$$

$\alpha \rightarrow$ hyperparameter (we have to decide what to take)

Questions:

1) Is there any Hyperparameter in Vanilla N.B ?

Ans No. It will be having zero probability value

2) How do you resolve zero prob Issue ?

Ans By appling Regularization

3) Can you explain Outfitting & Underfitting in N.B ?

$\alpha \rightarrow \uparrow \rightarrow$ Underfitting

$\alpha \rightarrow \downarrow \rightarrow$ Overfit

4) Is there any impact of outliers on N.B ?

Not affected by outliers that much

No impact

5) Why we called N.B as MAP (Maximum

Because x_q → Posteriori

We try to find $P(c_1|x_q) = P(x_q|c_1) * P(c_1) / P(x_q)$

$P(c_2|x_q)$

$P(c_k|x_q)$

Max of these probability is taken as Class

From Sklearn → Naive Bayes

- Gaussian NB
  - Data → Numerical features ↑ few categorical features
- Multinomial NB
  - Data → Text data
- Bernoulli NB
  - lot of Categorical features.

6) Assumption in NB Algorithm?

* features should be conditionally independent
* Another assumption for Gaussian NB
  - All num features should be gaussian distributed.

Decision Tree Algorithm

highly used Algorithms

→ Decision Tree Model (ID3, C4.5, C5.0, SLIQ, SPRINT) → commercialized (not available freely)

SPRINT → where data is distributed on CART differ at machines

Algorithms to create DT Models

* ID3 → Iterative Dichotomiser 3 (Old Algorithm only works for the data which contains categorical features → Works only for classification)
  - ↳ Division
* Rule Based Algorithm
* C4.5 → Solves both Classification as well as Regression
  - Data → Categorical / Numerical

Rules: for example Iris data

2. for Setosa
   - PL < 2.5
   - PW < 0.8

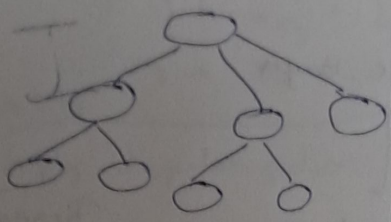
For Versicolor
- PL < 5.0
- PW < 1.6

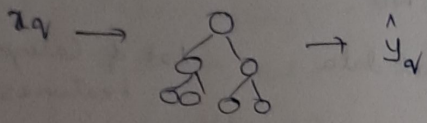
For Virginica
- PL ≥ 5.0
- PW ≥ 1.6

shape, column index } properties

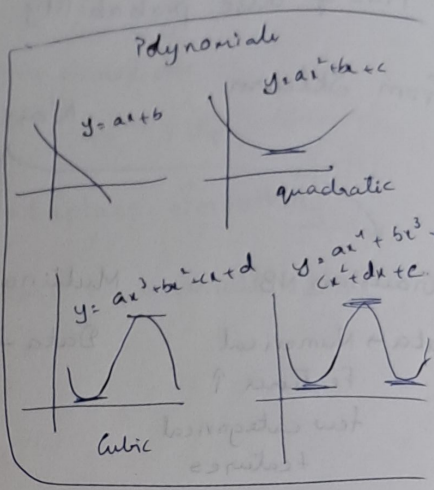
describe } function

Tree Structure :





(Check rules/ conditions)



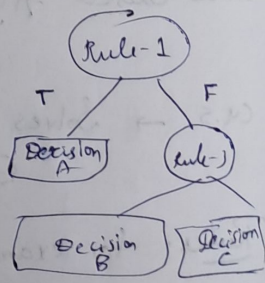
- For DT Algorithms we don't need to apply Standardization/ Normalization technique & on categorical columns only apply label encoder.

→ Max depth refers to length of longest path from root to a leaf / limit to stop further splitting of nodes.

To picturize DT model
» from sklearn.tree import plot_tree
» filled = True (Colour filling)
» rounded = True (for edge roundness)

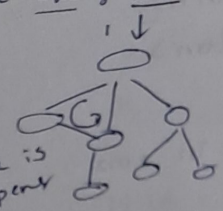
Learning :

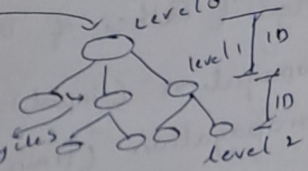
x-train, y-train → [ID3/ C4.5] → Model (set of rules arranged in tree structure)



Tree Structure norms :
+ Nodes (Root, Intermediate, leaf nodes)
+ Branches (No cycles)

difference b/w graphs & trees:





SQL → indexing for easy & quick retrieval of data
↳ indexing is stored using B trees & B+ trees format.

ID3 Algorithm : (Entropy, Information Gain, Gini Impurity)

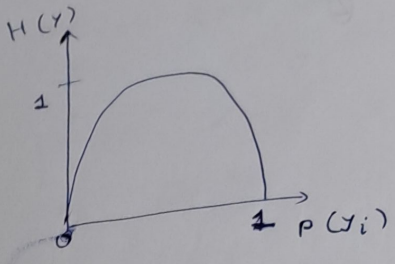
1. Entropy → Information theory
→ Bivariant

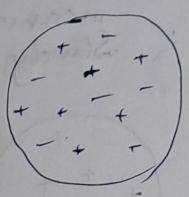
→ H (Target)

→ $H(Y) = -\sum_{i=1}^{k} P(y_i) \cdot \log_2 P(y_i)$; k → no. of classes.

$0 \leq H(Y) \leq 1$ ← Max Entropy

↙ No imp



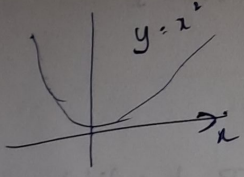


$\Rightarrow H(Y) = -\{P(yes) * \log_2(P(yes)) + P(No) * \log_2(P(No))\}$

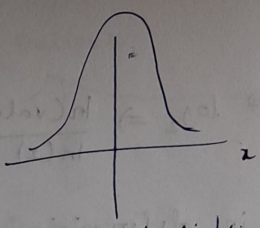
$= -\left\{\frac{9}{14} * \log_2\left(\frac{9}{14}\right) + \frac{5}{14} * \log\left(\frac{5}{14}\right)\right\}$

$= 0.60$

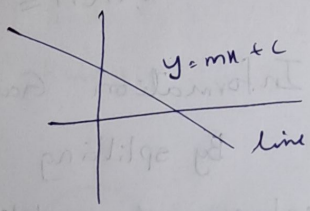
log (0 to 1)
↓
-ve value



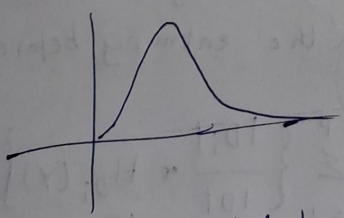
Parabola (function)



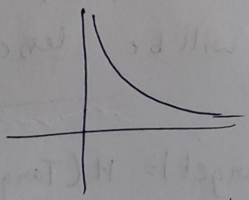
Normal distri



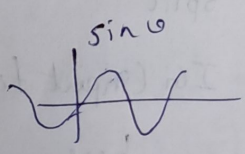
function

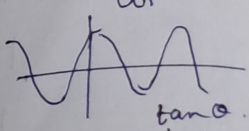


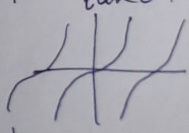
Right skewed dist
(Log Normal)

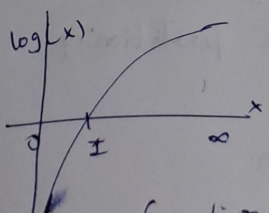


Pareto dist
Power law distribution

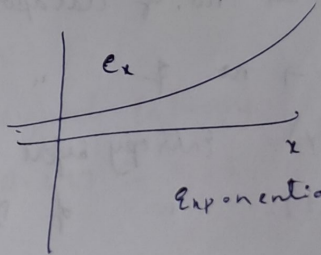








function.



Exponential graph.

In deep learning (father of DL - Geoffrey Hinton)

- forward propagation (input)
- backward propagation (train)

} used in 2012

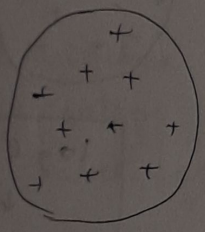
- Make a split of impure data such that the new partitions should have lesser impurity → Entropy ↓

Ways to create Decision Trees?

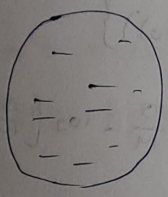
① ID3 (Iterative Dichotomiser 3) - Only Classification task for categorical variables

②

Entropy: measure of randomness or impurity



Entropy = 0
∴ No impurity value



H(Y) = Entropy = 0

of -ve = # of +ve
→ H(Y) = 1

⇒ $0 \leq H(Y) \leq 1$

⇒ $\log_2 \Rightarrow \frac{\ln(\text{value})}{\ln(2)}$

2. Information Gain (helps in determining the Best split)

By splitting into multiple patterns the weighted entropy of partitions will be lesser than the entropy before split.

$$IG(\text{input feature}, \text{target}) = H(\text{Target}) - \sum_{i=1}^{p} \left\{ \frac{|D_i|}{|D|} * H_{D_i}(Y) \right\}$$

here, p → Total no of partitions

$H_{D_i}(Y)$ → Weighted entropy.

$|D_i|$ → no. of datapoints in each partition

$|D|$ → no. of " before partitioning

$H(Y)$ = Entropy before partition

Data:

Outlook	Temp	Humidity	Wind	PlayTennis
S	Hot	High	Weak	N
S	H.	H	strong	N
O	H	H	W	Y
R	Mild	H	W	Y
R	Cool	Normal	W	Y
R	C	N	S	N
O	C	N	S	Y
S	M	H	W	N
S	C	N	W	Y
R	M	N	W	Y
S	M	N	S	Y
O	M	N	S	Y
O	H	H	S	Y
R	M	N	W	Y
		H	S	N

$$H(\text{Target}) = -\sum_{i=1}^{K} P(x) \log_2 (P(x))$$

$$= -\left[(P(Y) \cdot \log_2 P(Y)) + (P(N) \cdot \log_2 P(N))\right]$$

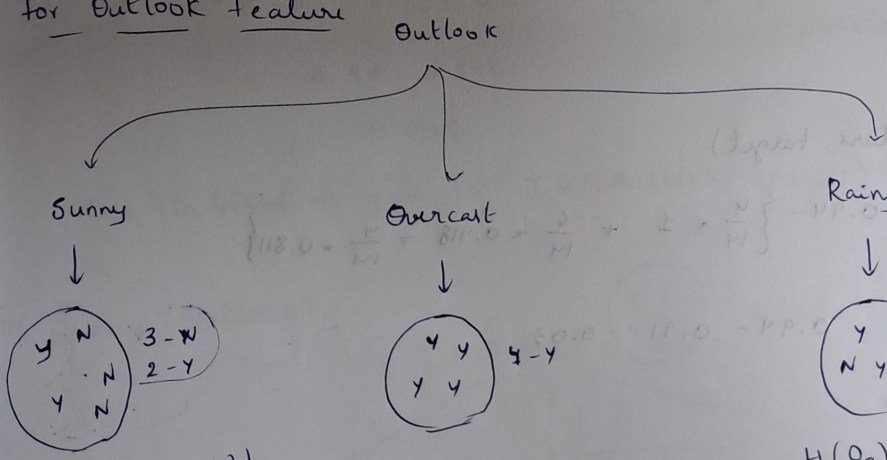
$$= -\left[\frac{9}{14} \cdot \log_2\left(\frac{9}{14}\right) + \frac{5}{14} \cdot \log_2\left(\frac{5}{14}\right)\right]$$

$$= -\left[-0.6374 * (9/14) + (5/14) * (-1.4854)\right]$$

$$= -\left[-0.409 + (-0.5305)\right]$$

$$= -(-0.9394) = 0.94$$

For Outlook feature



$$H(O_S) = -\left[\frac{3}{5}\log_2\left(\frac{3}{5}\right) + \frac{2}{5}\log_2\left(\frac{2}{5}\right)\right]$$

$$= -(-0.4422 + (-0.5288))$$

$$= 0.97$$

$$H(O_O) = 0$$

$$H(O_R) = -\left[\frac{3}{5}\log_2\left(\frac{3}{5}\right) + \frac{2}{5}\log_2\left(\frac{2}{5}\right)\right]$$

$$= 0.97$$

Avg = Weighed Entropy

$$= 0.647 \simeq 0.65$$

$$IG(Outlook, target) = H(target) - \sum_{i=1}^{P} \left\{ \frac{|D_i|}{|D|} * H_{D_i}(y) \right\}$$

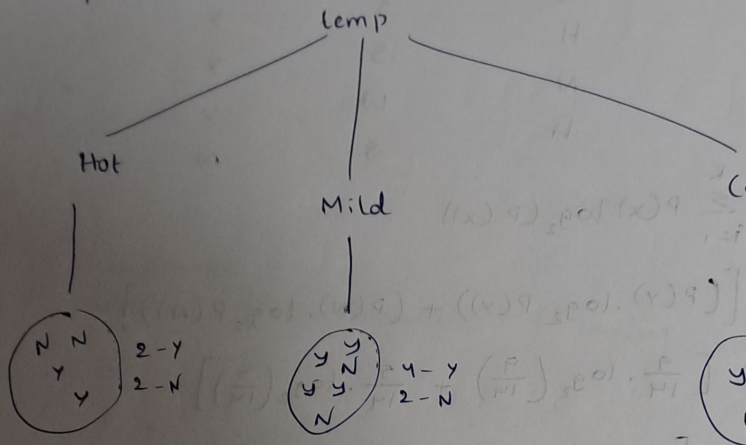
$$= 0.94 - \left\{ \frac{5}{14} * 0.97 + \left(\frac{4}{14} * 0\right) + \left[\frac{5}{14} * 0.97\right] \right\}$$

$$= 0.94 - \left\{ 2\left[\frac{5}{14} * 0.97\right] \right\}$$

$$= 0.94 - 0.6714$$

$$= 0.2686$$

For temp feature



$$H(T_h) = -\left[\frac{2}{4}\log_2\left(\frac{2}{4}\right) + \frac{2}{4}\log_2\left(\frac{2}{4}\right)\right] = 1$$

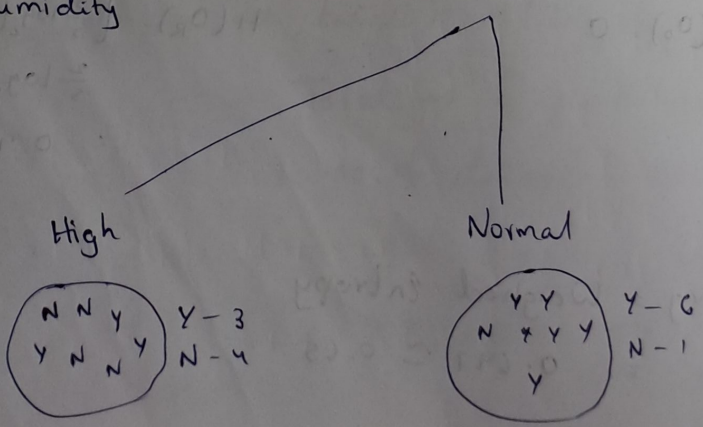
$$H(T_m) = -\left[\frac{4}{6}\log_2\left(\frac{4}{6}\right) + \frac{2}{6}\log_2\left(\frac{2}{6}\right)\right] = 0.918$$

$$H(T_c) = -\left[\frac{3}{4}\log_2\left(\frac{3}{4}\right) + \left(\frac{1}{4}\right)\log_2\left(\frac{1}{4}\right)\right] = 0.811$$

$$IG(Temperature, target) = 0.94 - \left\{ \frac{4}{14} * 1 + \frac{6}{14} * 0.918 + \frac{4}{14} * 0.811 \right\}$$

$$= 0.94 - 0.91 = 0.03$$

Humidity



$H(H_h) = -\left[\frac{4}{7}\log_2\left(\frac{4}{7}\right) + \frac{3}{7}\log_2\left(\frac{3}{7}\right)\right]$ $H(H_N) = -\left[\frac{6}{7}\log_2\left(\frac{6}{7}\right) + \frac{1}{7}\log_2\left(\frac{1}{7}\right)\right]$

$= 0.985$ $= 0.591$

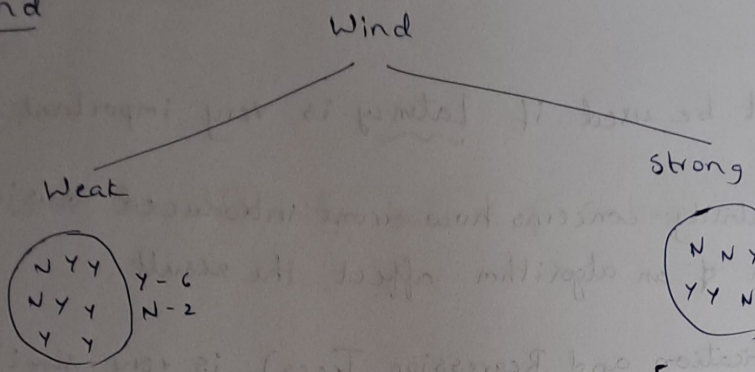
$IG(\text{Humidity, Target})$

$= 0.94 - \left\{\frac{7}{14} * 0.985 + \frac{7}{14} * 0.591\right\}$

$= 0.94 - 0.788$

$= 0.152$

Wind



$H(W_w) = -\left[\frac{6}{8}\log_2\left(\frac{6}{8}\right) + \frac{2}{8}\log_2\left(\frac{2}{8}\right)\right]$

$= 0.811$

$H(W_s) = -\left[\frac{3}{6}\log_2\left(\frac{3}{6}\right) + \frac{3}{6}\log_2\left(\frac{3}{6}\right)\right]$

$= 1$

$IG(\text{Wind, Target})$

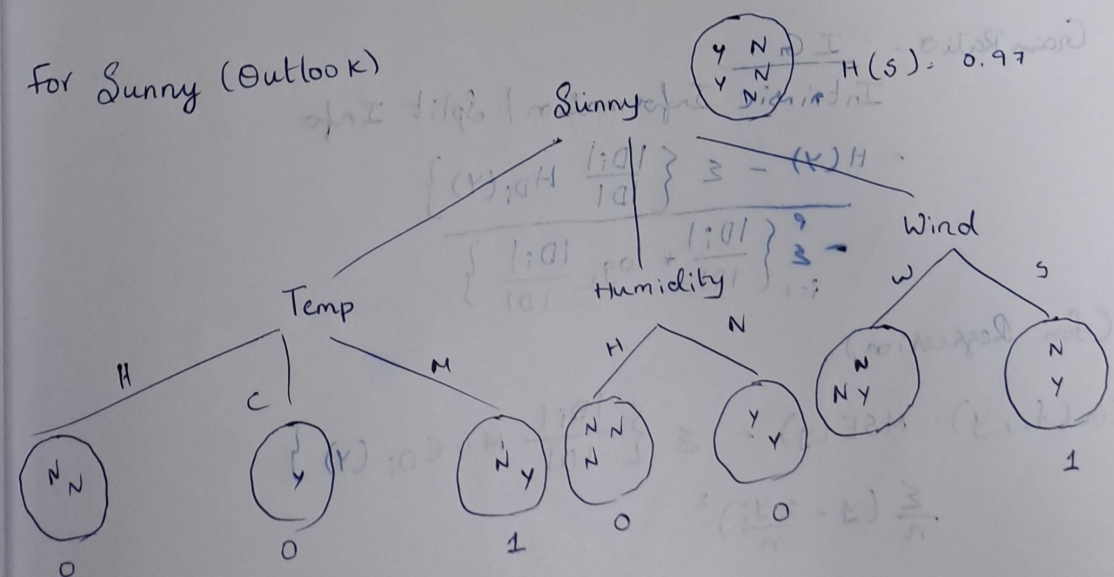
$= 0.94 - \left\{\frac{8}{14} * 0.811 + \frac{6}{14} * 1\right\}$

$= 0.94 - 0.892$

$= 0.048$

Max information gain is $IG(\text{outlook, target})$

For Sunny (Outlook)



Take a decision on which feature to choose for splitting

		Classification	Regression
	* KNN	✓ (mode)	✓ (mean
Supervised Learning	* Naive Bayes (Max a posterior)	✓	X
	* Decision Tree		

* KNN cannot be used if Latency is very important

* Numerical Stability concerns how errors introduced during the execution of an algorithm affect the result.

⇒ CART (Classification and Regression Trees) is very similar to C4.5 and this algo is used by sklearn library

⇒ C4.5 (Entropy + Information Gain Ratio)

⇒ ID3 (" + Information Gain)

⇒ CART (Gini Impurity / Gini Index + Weighed Sum of Entropy)

Information Gain Ratio:

Information gain favours high branching features, lets look at IG Ratio

$$\text{Gain Ratio} = \frac{IG}{\text{Intrinsic Information / Split Info}}$$

$$= \frac{H(Y) - \sum \left\{ \frac{|D_i|}{|D|} H_{D_i}(Y) \right\}}{-\sum_{i=1}^{p} \left\{ \frac{|D_i|}{|D|} * \log_2 \frac{|D_i|}{|D|} \right\}}$$

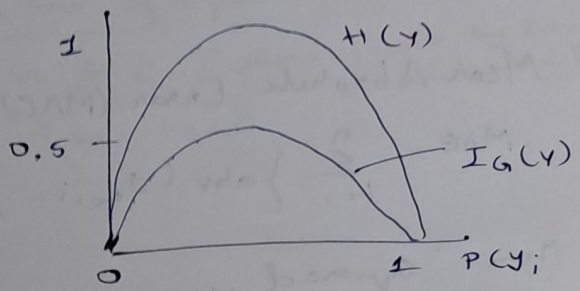
IG (for Regression)

$$IG(f, y) = MSE(y) - \sum \left\{ \frac{|D_i|}{|D|} MSE_{D_i}(Y) \right\}$$

$$\frac{\sum}{n} (y - \frac{\sum y_i}{n})^2$$

Gini Index (Another form of measure of impurity)

Gini Impurity



→ $I_G(\text{Target}) = 1 - \sum_{i=1}^{C} (P(y_i))^2$

$$0 \leq I_G(Y) \leq 1 - \frac{1}{C}$$

→ $1 - \{ P(y = +ve)^2 + P(y = -ve)^2 \}$

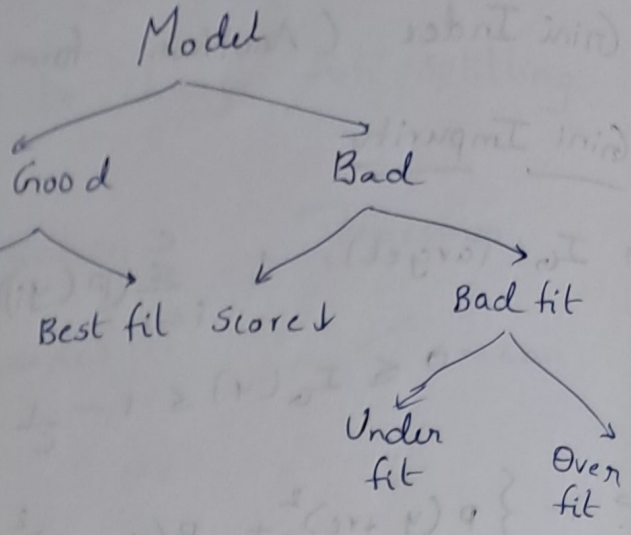
Measure of Randomness :

- $H(Y)$
- $I_G(Y)$

C4.5 with Gini Impurity instead of Entropy.

Evaluation Matrix:

For Regression :-



① Mean Absolute Error (MAE)

$$MAE = \sum_{i=1}^{n} \{abs(y_{act_i} - y_{pred_i})\}$$

② Mean Squared MSE (MSE)

Algorithm → Decision Tree (CART)

How ?

→ Computer impurity in your data using entropy / Gini impurity

→ Choose one feature & split your data into multiple partitions.

Representation of datapoint or row using maths

↳ Vector → By default representation → Column

Equation of a line:

$y = mx + c$

- m → Slope of the line w.r.t x.
- c → intercept w.r.t y

Parameters of a plane / line:

m & c [where y cuts by line i.e. at which point]

↳ $m = \tan\theta$

2D dim → $ax + by + c = 0$ (Not better approach) } Eq of line

$w_1x_1 + w_2x_2 + w_0 = 0$

3D dim → $w_1x_1 + w_2x_2 + w_3x_3 + w_0 = 0$ → Eq of plane

4D dim → $w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_0 = 0$ → Eq of hyperplane

100 dim → $w_1x_1 + \dots + w_{100}x_{100} + w_0 = 0$ → It is also Eq of hyperplane.

For 2D → $w_1x_1 + w_2x_2 + w_0 = 0$.

$w_1x_1 = -w_2x_2 - w_0$

$x_1 = -\left[\frac{w_2}{w_1}\right]x_2 - \left[\frac{w_0}{w_1}\right]$

Eq of ~~Par~~ Hyperplane

→ π_d →

# of slopes = $d-1$

# of intercepts = 1

Slope: $-\frac{w_2}{w_1}$ w.r.t x_2

Intercept: $-\frac{w_0}{w_1}$ w.r.t x_1

$\therefore w_1x_1 + w_2x_2 + \dots + w_{100}x_{100} + w_0 = 0$

$\vec{w} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$

→ $\pi_{100} = \sum_{i=1}^{100} w_ix_i + w_0 = 0$

$\vec{w} \in \mathbb{R}^2$ → Real values with 2 components

$\therefore$

Represent the Eq of Hyperplane as a dot Product

$\pi_d : \vec{w} \cdot \vec{x} + w_0 = 0$

π_d : $\sum_{i=1}^{d} w_ix_i + w_0 = 0$

π_d : $\vec{w}^T\vec{x} + w_0 = 0$

$\therefore w^Tx + w_0 = 0$

π_d : $\|\vec{w}\|\|\vec{x}\| \cos\theta_{\vec{w},\vec{x}} + w_0 = 0$

20/12/22

2D → Equation of Circle :

$$x^2 + y^2 = r^2$$

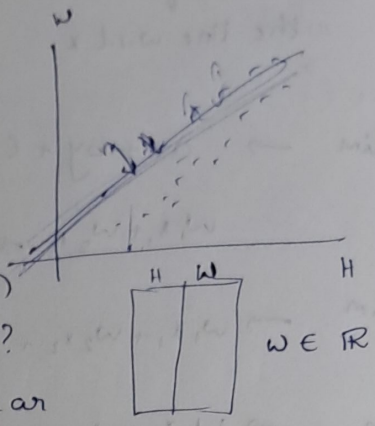
3D → Equation of Sphere :

$$x^2 + y^2 + z^2 = r^2$$

Regression task

Algorithm: Linear Regression

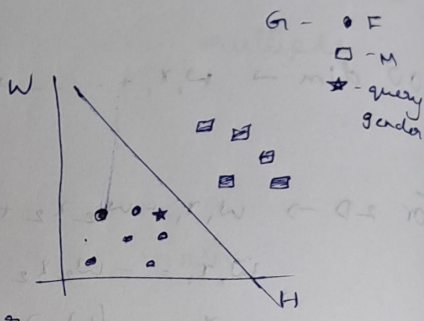
- How? Find a line that best fits the data (Role of linear Regression)
- Why it is named as linear Regression?
  Coefficient terms should be linear



Classification task

Algorithm: Logistic Regression

How? Find a line the best separator +ve from -ve.

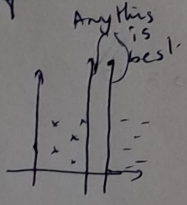


Logistic Regression only find the linear pattern in case of non-linear distribution it simply fails

* Gives only straight boundary.

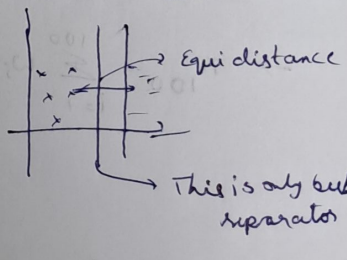
Logistic (Vs) Support Vectors

It takes any of the best fit separator for analyzing.



It only gives the best fit separator among all the best fit separators

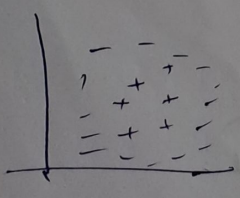
* It is only for Classification tasks



Feature Engineering:

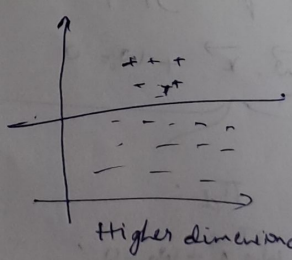
learns new features from the data.

By SVM



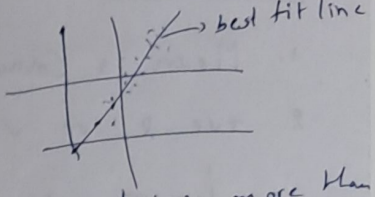
Lower dimensionality

Automatic feature Engg →



Higher dimensionality

How? Try to find HYPERPLANE that BEST FITS the given data



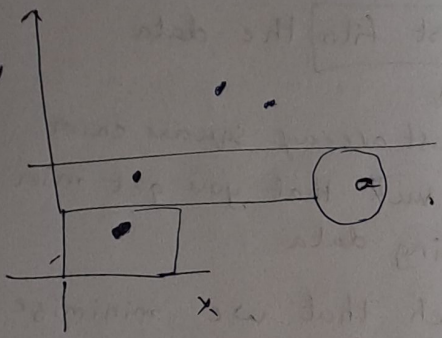
Types of Linear Regression:

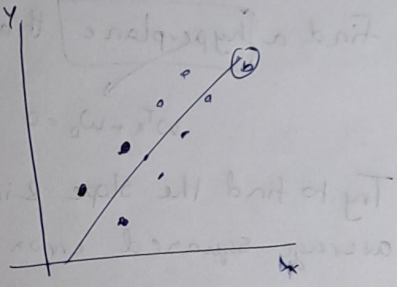
1. Simple Linear Regression (If your input contains one feature) $y = mx + c$
2. Multiple linear Regression (If your input contains more than one feature)

$$y = m_1x_1 + m_2x_2 + c$$

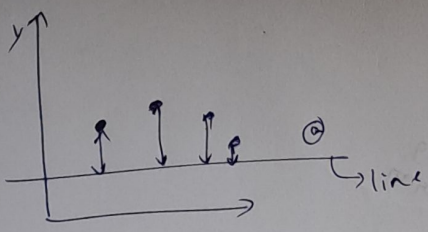
3. Linear Regression with feature Engg / Polynomial Regression

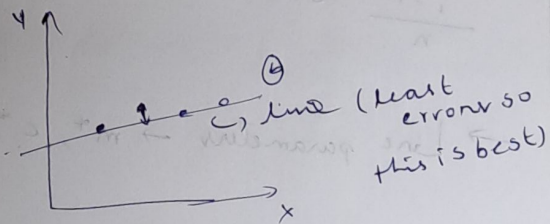
What do I mean by Predict BEST FIT?





The line which fits the best is having the / making least error.





Compute the error done by line ⓐ by line ⓑ Compute

Point 1 → $error_1 = y_1 - \hat{y}_1 = y_1 - (mx_1 + c)$

Point 1 → $error_1 = y_1 - \hat{y}_1$ (Actual, Predict)

2 → $error_2 = y_2 - \hat{y}_2$

3

4

5 → $y_5 - \hat{y}_5$

$\frac{\Sigma}{n}$ $error_i$ > $\frac{\Sigma}{n}$ $error_i$

ⓑ line best fits

⇒ Issues with Linear Regression:

1. Mean is Corrupted
2. +ve & -ve values

↳ To solve this problem.

for ② → $\frac{\sum_{i=1}^{n} (y_i - (mx_i + c))^2}{n}$

Actual — error — predicted

The best fit line is a line that has the least average squared error

→ find a hyperplane that has best fits the data

$w^T x + w_0 = 0$ — least average square error.

* Try to find the slope & intercept such that you get min average squared error on training data

* Try to find the $\vec{w}, w_0$ such that we minimize $\frac{\sum_{i=1}^{n} (y_i - (w^T x_i + w_0))^2}{n}$ on the training data.

→ Line parameters → m^*, c^*

X-train, y-train

minimizing the error

→ $\vec{w}^*, w_0^* = \text{argmin} \left\{ \frac{\sum_{i=1}^{n} (y_i - \{w(x_i) - w_0\})^2}{n} \right\}$ → Model

↳ loss function
Error function
Cost function

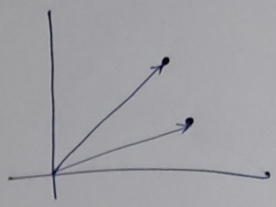
Optimization means minimizing a function i.e., minimizer for function.

Ordinary least Square (or) Equation.

** How do you Solve the Optimization Eqn's ?

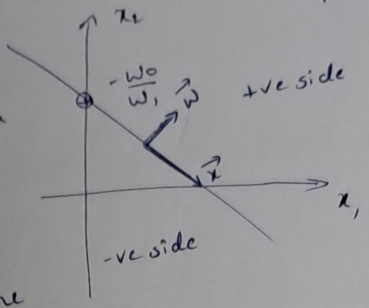
Ans Using Optimizers
1. Gradient Descent
2. Stochastic Gradient
3. Momentum Based Gradient Descent
4. RMS Prop

The dot product b/w the vectors is zero whenever these vectors and are perpendicular or orthogonal to each other



$$\vec{w} \perp \vec{x}$$

$$\vec{w} \cdot \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \\ w_0 \end{bmatrix}$$



⇒ $\vec{w}$ is called a norm of a hyperplane

$\vec{w} \perp$ to the hyperplane

* It helps us identify the direction in which the hyperplane is facing

If $\vec{w} \cdot \vec{p} > 0 \Rightarrow \vec{p}$ is on the +ve side

$\vec{w} \cdot \vec{p} < 0 \Rightarrow \vec{p}$ is on the -ve side

$\vec{w} \cdot \vec{p} = 0 \Rightarrow \vec{p}$ is on the line

$$\Rightarrow \frac{\sum_{i=1}^{n} [\mu_{f_1} - (obs_i)_{f_1}] * [\mu_{f_2} - (obs_i)_{f_2}]}{n}$$

$$= \text{Covariance}(f_1, f_2)$$

$$\beta = \frac{\text{Covariance}(f_1, f_2)}{\sigma(f_1) * \sigma(f_2)}$$

* Slope determines the rate of change

Logistic Regression:

Find a hyperplane that Best Separates +ve's from -ve's.

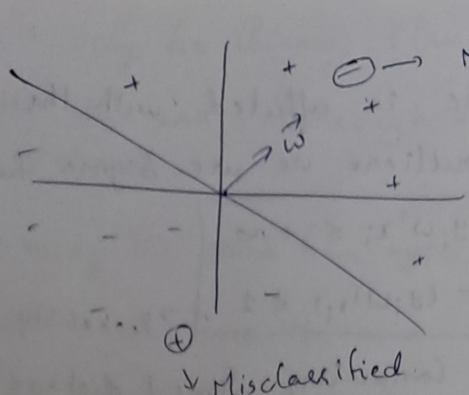
↓

max the correctly classified points

Signed distance → $act_i * predicted_i$

↳ -ve or +ve

Best model to be linear and correct rate of change is represented.



Total no. of Dpts = 13

Total no. of misclassified = 2

Total no. of correctly classified = 11

Case - 1:

if actual = +1 & predicted = +ve

actual * predicted = +ve Correctly classification

Case - 2:

if actual = +1 & predicted = -ve

actual * predicted = -ve Misclassification

Case - 3:

if actual = -1 & predicted = -ve

actual * predicted = +ve Correct Classification

Case - 4: if actual = -1 & predicted = +ve

actual * predicted = -ve Mis Classification

Distance of a point from hyperplane:

$$d = \frac{\vec{w} \cdot \vec{p} + w_0}{\sqrt{\sum w_i^2}}$$

Let assume the w_0 is 0 { TI is passing through Origin }

Assume the $\vec{w}$ is unit vector

$$\|\vec{w}\| = 1$$

$$w_0 = 0$$

$$d = \vec{w} \cdot \vec{p}$$

Signed distance is affected with the outliers. To minimize the outliers we use Sigma function then

	$-\infty \leq y_i w^T x_i \leq +\infty$
After Sigma fn	$0 \leq \sigma(y_i w^T x_i) \leq 1$ → Probability

Instead of Computing Signed distance for row, we will compute Sigmoid (Signed Distance)

$$w^* = \arg\max \left\{ \prod_{i=1}^{n} \sigma(\text{Signed dist}) \right\}$$

$$= \arg\max \left\{ \pi \left(\frac{1}{1+\exp\{-y_i w_\tau x_i\}} \right) \right\}$$

Problem → Numerical Instability Because the product b/w no. of values is zero.

Solution → log transformation.

$$w^* = \arg\max \left\{ \log \left(\prod_{i=1}^{n} \left(\frac{1}{1+\exp\{-y_i w_\tau x_i\}} \right) \right) \right\}$$

$$\log\left(\prod_{i=1}^{3} a_i\right) = \sum_{i=1}^{3} \log a_i$$

$$= \arg\max \left\{ \sum_{i=1}^{n} \log \left(\frac{1}{0+\exp\{-y_i w_\tau x_i\}} \right) \right\}$$

$$= \arg\max \left\{ \sum_{i=1}^{n} (\log(1) - \log(1+\exp(-y_i w^T x_i))) \right\}$$

$$w^* = \arg\max \left\{ -\sum_{i=1}^{n} \log(1+\exp\{-y_i w^T x_i\}) \right\}$$

Optimization of Logistic Regression.

Q) How do we solve the above Eq?

Ans Optimizer → Gradient Ascent.

Acc to Optimization Theory

$$\max(-f(x)) = \min(f(x))$$

$$\therefore w^* = \arg\min \left\{ \sum_{i=1}^{n} \log(1+\exp(-y_i w^T x_i)) \right\}$$

final Optimization Equation.

→ Loss function / Error fn / cost fn / Logistic Loss function.

⇒ How do we solve?

Gradient Descent.

4/12/22

Logistic Regression is only for Binary Classification

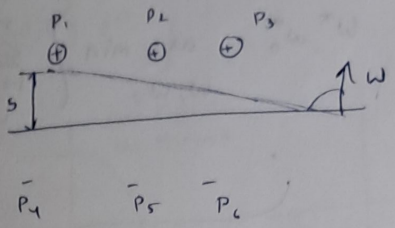
* We assume the line passes through origin i.e, there is no intercept value.

Magnitude: how far away the point from hyperplane

In Logistic regression we have the issue of outliers

Solution: Sigmoid function to treat outliers.

For Eg :- First hyperplane



# of Data points = 7
# of Correctly Classified = 6
# of Misclassified = 1

Signed distance:

$act_i * pred_i$

$\downarrow$

$y_i * w^T x_i$

$y_i * (\vec{w} \cdot \vec{x_i})$

$P_1 = +1 * (+ve\ value) = +ve\ mag = +5$

$P_2 = +1 * (+ve\ value) = +ve\ mag = +5$

P_3 - - - - - - - - $= +5$

$P_4 = -1 * (-ve\ value) = +ve\ mag = +5$

P_5 - - - - - - - $= +5$

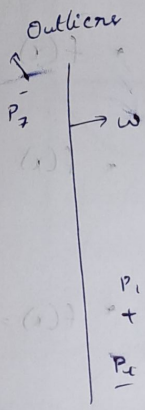
P_6 - - - - - - - - $+5$

P_7 $-1 * (+ve\ value) = -ve\ mag = -50$

$\sum (act_i * pred_i)$

$= [5+5+5+5+5+5+(-50)]$

$= -20$



Second hyperplane

$P_1 = +1 * (+ve\ value) = +1$

$P_2 = +1 * (\ \ "\ \ ) = +5$

$P_3 = \ " * (\ \ "\ \ ) = +10$

$P_4 = -1 * (+ve\ value) = -1$

$P_5 = -1 * (\ \ "\ \ ) = -5$

$P_6 = -1 * (\ \ "\ \ ) = -10$

$P_7 = -1 * (-ve\ value) = +1$

$\sum (act_i * pred_i)$

$= [5+10+1+(-1)+(-5)+(-10)+1]$

$= 1$

Support Vector Machine:

$\Rightarrow$ $D_{n \times d} \xrightarrow[\text{trick}]{\text{Kernel}} D_{n \times d'}$ i.e., $d' >> d$

(exposed to larger dimensions)

It project the data into higher dimensional space

Linear Regression:

$$w^*, w_0^* = \arg\min_{w, w_0} \left\{ \sum_{i=1}^{n} (y_i - (w^T x_i + w_0))^2 \right\}$$

SSE Loss function

Optimization fn for linear Regression.

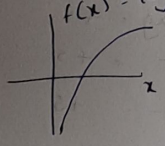
$$= \arg\min_{w, w_0} \left\{ \frac{\sum_{i=1}^{n} (y_i - (w^T x_i + w_0))^2}{n} \right\}$$

MSE loss function

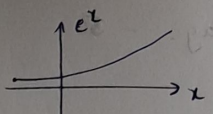
Gradient Descent:

$$x^* = \arg\min_{x} \{f(x)\}$$

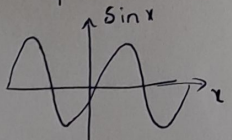
- $f(x) = \log x$



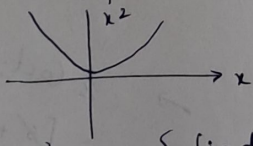
- $f(x) = e^x$



- $f(x) = \sin x$



- $f(x) = x^2$



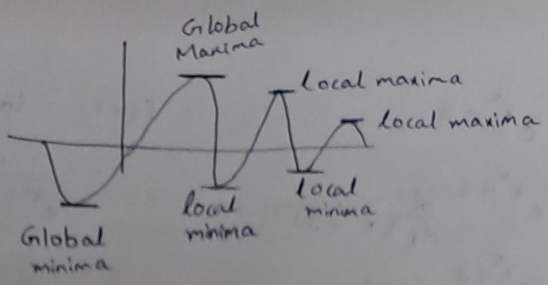
$x^* = \arg\min_{x} \{x^2\} \rightarrow$ { find that value of x which minimizes x^2 function }

Linear Reg:

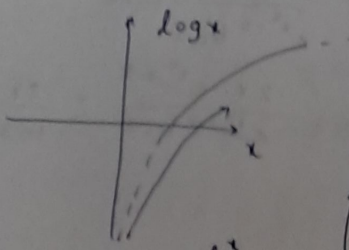
$$w^*, w_0^* = \arg\min_{w, w_0} \left\{ \frac{\sum}{n} (y_i - (w^T x_i + w_0))^2 \right\} \rightarrow$$

{ find the values of w_i, w_0 which minimizes the MSE loss function }

$f(x) \Rightarrow$

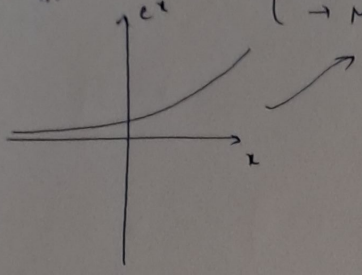


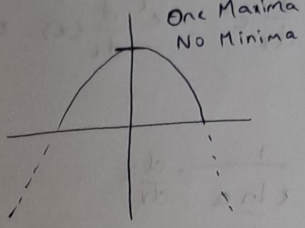
$\Rightarrow$



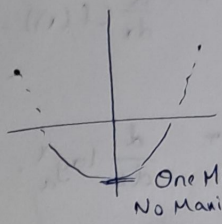
No Maxima
No Minima

{ It is a continuous & always increasing f^n → Monotonic f^n }





Concave f^n



Convex f^n

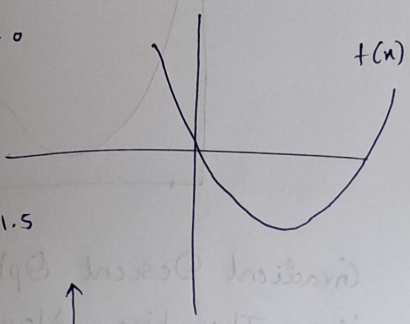
* Linear & Logistic Regression are Convex functions.

Problem:

$f(x) = x^2 - 3x + 2$. Is it is Convex or Concave f^n

differentiation of a function @ a point x_0

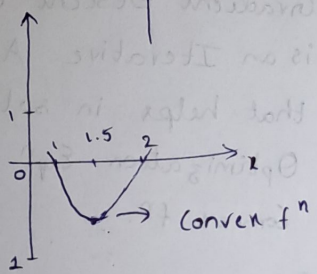
$\Rightarrow$ Slope of tangent @ point x_0



$$\frac{d}{dx} f(x) = \frac{d}{dx}(x^2 - 3x + 2) \Rightarrow 2x - 3 = 0 \Rightarrow x = 1.5$$

@ $x = 1.5$

$f(x) = (1.5)^2 - 3(1.5) + 2 = -0.25$



Then we check for prior and after ptr

@ $x = 1 \Rightarrow f(x) = 0$

@ $x = 2 \Rightarrow f(x) = 0$

So, here it is minima

Optimizer is a function / also that helps you solve Optimization function (with Convex f^n's).

Gradient Descent should be used for Convex f^n then only it gives proper results.

27/12/2022

$f(x) = e^{x^2+3}$

$= e^{x^2+3} \frac{d}{dx}(x^2+3)$

$= e^{x^2+3} \cdot (2x)$

$= 2x \cdot e^{x^2+3}$

$f(x) = \log_3 e^{x^2+3}$

$= \frac{1}{e^{x^2+3} \ln 3} \cdot \frac{d}{dx}(e^{x^2+3})$

$= \frac{1}{e^{x^2+3} \ln 3} \cdot e^{x^2+3} \cdot 2x$

$= \frac{2x \cdot e^{x^2+3}}{e^{x^2+3} \ln 3} = \frac{2x}{\ln 3}$

* $\frac{d}{dx} f(x) = \frac{d}{dx} ax = a \frac{dx}{dx} = a$

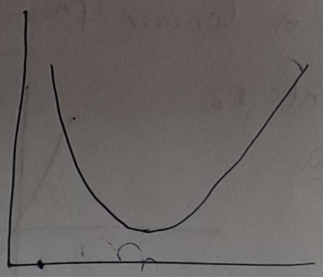
* $\frac{d}{dx}(f(x)) = \frac{d}{dx} e^x = e^x \frac{d}{dx}(x) = e^x$

* $\frac{d}{dx} \log_a^x = \frac{1}{x \ln a} \cdot \frac{dx}{dx}$

↳ natural logarithm

* $\frac{d}{dx} x^n = n \cdot x^{n-1}$

Gradient Descent:



Gradient Descent Optimizer is an Iterative Algorithm that helps in solving Optimization Eqn with Convex fn.

$x^* = \underset{x}{\text{argmin}} \{ f(x) \}$

⇒ Assuming $f(x)$ to be a convex function

Algorithm:

1. Randomly pick the value of x x_0 (minima if slope of tangent = 0)
2. Move forward mining & inorde to do that we have to mo either towards left or Right
3. Reiterate step 2 until you reach minima.

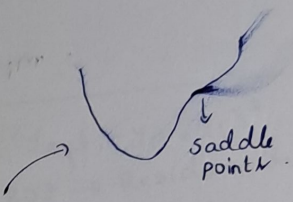
① Randomly initialize $\rightarrow x_i$

② Move towards minima $\rightarrow$ hyperparameter of Gradient Descent

$$x_{i+1} = x_i - \eta \left[\frac{d}{dx} f(x)\right]$$

η ↑ learning rate

[update Eqn of Gradient Descent]



Issue → Oscillation Problem

→ Gradient Descent can get stuck at Saddle points

Hyperparameter of GD

① η = Learning rate

② # of iteration.

$$m_{i+1} = m_i - \eta \left[- \frac{\sum_{i=1}^{n}}{n} \right]$$

$$m_{i+1} = m_i - \eta \left[\frac{\partial}{\partial m} f(m)\right]_{m_i}$$

$$= m_i - \eta \left[\frac{\partial}{\partial m}\left[+ \sum_{i=1}^{n} \log\left(1 + \exp\left\{-y_i w^T x_i\right\}\right)\right]\right]_{m_i}$$

$$= m_i - \eta \left[\frac{\partial}{\partial m}\left[+ \sum_{i=1}^{n} \log\left(1 + \exp\left\{-y_i m_i x_i\right\}\right)\right]\right]_{m_i}$$

$$= m_i - \eta \left[+ \sum_{i=1}^{n} \frac{\partial}{\partial m}\left[\log\left(1 + \exp\left\{-y_i m_i x_i\right\}\right)\right]\right]_{m_i}$$

$$= m_i - \eta \left[+ \sum_{i=1}^{n} \left(\left[\frac{\partial}{\partial m}(\log 1)\right] + \frac{\partial}{\partial m}\left[\log\left(\exp\left\{-y_i m_i x_i\right\}\right)\right]\right)\right]_{m_i}$$

$$= m_i - \eta \left[+ \sum_{i=1}^{n} \left[[0] + \frac{1}{\exp\{-y_i m_i x_i\} \ln a} \cdot \frac{\partial}{\partial m}\left(\exp\left\{-y_i m_i x_i\right\}\right)\right]\right]_{m_i}$$

$$= m_i - \eta \left[\sum_{i=1}^{n} \frac{\partial}{\partial m_i} \left[\log \overbrace{(1+\exp\{-y_i m_i x_i\})}^{} \right] \right]_{m_i}$$

$$= m_i - \eta \left[\frac{1}{(1+\exp\{-y_i m_i x_i\}) \log 1} \cdot \frac{\partial}{\partial m_i} (1+\exp\{-y_i m_i x_i\}) \right]$$

$$= m_i - \eta \left[\frac{1}{1+\exp\{-y_i m_i x_i\} \log 1} \left[\frac{\partial}{\partial m_i}(1) + \frac{\partial}{\partial m_i}(\exp\{-y_i m_i x_i\}) \right] \right]$$

$$= m_i - \eta \left[\frac{1}{1+\exp\{-y_i m_i x_i\} \log_1} \left[0 + \exp\{-y_i m_i x_i\} \frac{\partial}{\partial m_i}(-y_i m_i x_i) \right] \right]$$

$$= m_i - \eta \left[\frac{1}{1+\exp\{-y_i m_i x_i\} \log_1} \left[\exp\{-y_i m_i x_i\} \cdot (-y_i x_i) \right] \right]$$

↳ $(\log 1 = 0)$

$$= m_i - \eta \left[\frac{1}{1+0} \left[\exp\{y_i m_i x_i\} \cdot (y_i x_i) \right] \right]$$

$$= m_i - \eta \left[\exp\{y_i m_i x_i\} \cdot (y_i x_i) \right]$$

- $\log(ab) = \log a + \log b$
- $\log(a+b) = \log a + \log(1+b)$

Derivative of Sigmoid fn?

$$\frac{d}{dx} \sigma(x) = \frac{d}{dx}\left[\frac{1}{1+e^{-x}}\right]$$

$$= (1+e^{-x})^{-1}$$

$$= -1 \cdot (1+e^{-x})^{-1-1} \cdot \frac{d}{dx}(1+e^{-x})$$

$$= -1 (1+e^{-x})^{-2} \cdot (0+e^{-x})$$

$$= \frac{-1}{(1+e^{-x})^2} \cdot e^{-x} (-1)$$

$$= \frac{+e^{-x}}{(1+e^{-x})^2}$$

$$= \frac{e^{-x}}{(1+e^{-x})^2} = \frac{e^{-x}}{(1+e^{-x})(1+e^{-x})}$$

$$\left[\because \frac{e^{-x}}{1+2e^{-x}+(e^{-x})^2} = \frac{e^{-x}}{1+2e^{-x}+e^{-2x}} \right.$$

$$\frac{d}{dx}\sigma(x) = \frac{e^{-x}}{(1+e^{-x})} * \frac{1}{(1+e^{-x})}$$

$$= (1-\sigma(x)) * \sigma(x)$$

Linear Regression takes some time for training whereas for predicting it takes less time

for Logistic Regression.

$$D_n = \{(x_i, y_i)_{i=1}^{n} \mid x_i \in \mathbb{R}^{d-1}, y_i \in \{-1, +1\}\}$$

→ why because when we take 0, 1 then Signed distance becomes '0'.

Evaluation matrix: (for Regression)

MAE → $\frac{\Sigma}{n} abs(\underbrace{y_{act_i} - y_{pred_i}}_{error})$

* MSE → $\frac{\Sigma}{n} (y_{act_i} - y_{pred_i})^2$

* RMSE → $\sqrt{\frac{\Sigma}{n} (y_{act_i} - y_{pred_i})^2}$

$\Sigma (y_{act_i} - y_{pred_i})^2$

RSS → Residual Sum of squares

* R^2 → $1 - \frac{RSS}{TSS}$ $\quad [-\infty \leq R^2 \leq 1]$

* Adjusted R^2 → $\overline{R^2} = 1 - \left[\frac{(1-R^2)(n-1)}{n-p-1}\right]$

$$RSS = \Sigma (residual)^2$$

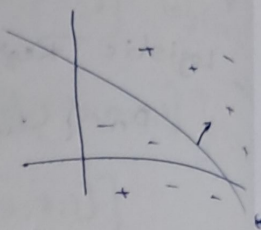
$$= \sum_{i=1}^{n} (y_{act_i} - y_{pred_i})^2$$ → Best Model

$$TSS = \Sigma (y_{act_i} - y_{pred_i})^2$$ → Dumb model.

⇒ Evaluation Matrix (for Classification) :

- Accuracy → $\frac{\#\text{ of Datapoints which were correctly classified}}{\text{Total \# of Datapoints}}$

→ $\frac{8}{10} = 0.8$



⇒

S.No	Actual	Predicted	So called
1	A	A	True Positive
2	Not A	Not A	~~Fals~~ True Negative
3.	A	Not A	False Negative
4.	Not A	A	False Positive

- Confusion Matrix → By this matrix we determine the accuracy of predicted & actual values

→ Accuracy : $\frac{TP + TN}{(TP + TN + FP + FN)}$

→ Precision : $\frac{TP}{(TP + FP)}$

→ Recall : $\frac{TP}{(TP + FN)}$

→ F1 Score : $\frac{2(p * r)}{(p + r)}$ ∴ P – precision, r – recall

from picture :

Actual \ Predicted	Positive	Negative
Positive	TP	FN
Negative	FP	TN