Nithish S
BTech AI & DS

Unit - II

Supervised Learning

2.1 Least Squares Linear Regression

⇒ This is said to be the most common form of Linear Regression.

⇒ It aims to minimize the sum of the Squared errors between actual and predicted values.

⇒ Goal is to find the best fit line, that can explain Relation Ship between dependent and Independent variable.

2.2 Simple Linear Regression:

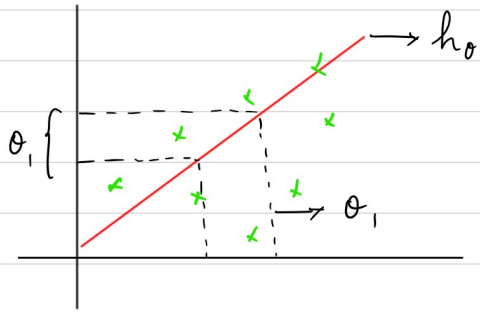
⇒ Involves "only one independent variable".

⇒ Goal is to model the relation between the dependent and independent variable.

⇒ 'y' is a linear function of 'x'.

Line Equation, $\left[h_\theta(x) = \theta_0 + \theta_1 \cdot x \right]$ (or) $\left[y = \beta_0 + \beta_1 x \right]$

⇒ For creating the straight line we use the equation.



θ_0 → Intercept

θ_1 → Slope

$\theta_0 \rightarrow$ Intercept, the point where we meet X-axis.

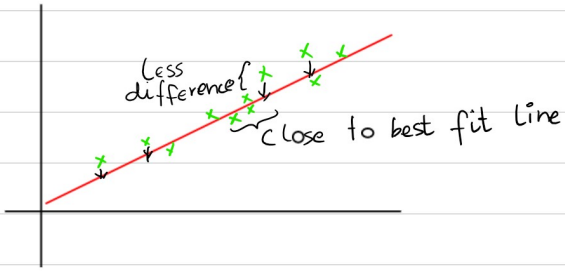
$\theta_1 \rightarrow$ One unit movement towards x-axis, Y-axis

$X_i \rightarrow$ Data Points

Aim of Linear Regression

- Find the best fit line such that difference between predict vs actual is very less.

-



- "Summation of all these points must be minimal" then only it is said to be best fit.

- Start at a point then lead to best fit line.
- For this purpose we use "Cost Function"

$$\text{Predicted} - \text{Actual} = \left(h_\theta(x) - y\right)^2$$

Cost Function

- Used for calculating error between actual and predicted.

m → no. of datapoints

$$RMS = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^i) - y^{(i)}\right)^2$$

sum of all data points.

How data is used in the Equation ?

data

$x = [1, 2, 3, 4, 5]$ $\quad Y = [2, 4, 5, 4, 5]$

equation

$$h_\theta(x) = \theta_0 + \theta_1 X$$

θ_0 x θ_1 are slopes and intercept , to find it we first

Step 1 Calculate means of X and Y.

$$\bar{X} = (1+2+3+4+5)/5 = 3$$

$$\bar{x} = (2+4+5+4+5)/5 = 4$$

Step 2 Calculate covariance and variance

$$\text{Covariance} = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{N-1}$$

$$(x,y) \Rightarrow [(1-3)\times(2-4) + (2-3)\times(4-4) + (3-3)\times(5-4) + (4-3)\times(4-4) + (5-3)\times(5-4)]/4 = 2.5$$

$$\text{Variance} = \frac{\sum (x_i - \bar{x})^2}{n-1}$$

$$\text{Var}(X) = \frac{[(1-3)^2 + (2-3)^2 + (3-3)^2 + (4-3)^2 + (5-3)^2]}{4}$$

$$= 2$$

Step 3 With this we calculate 'θ_1' & 'θ_0'

$$\theta_1 = \frac{Cov(x,y)}{Var(x)} = \frac{2.5}{2} \Rightarrow 1.25$$

$$\theta_0 = (\bar{y} - \theta_1) \times (\bar{x})$$
$$= (4 - 1.25) \times 3$$
$$= 0.25$$

(i.e) Linear Equation of our data is

$$y_i = 0.25 + (1.25 \times x_i)$$

x_i can be any data point

Ex:- If we substitute $x = 2$

then $y_i = 0.25 + (1.25 \times 2)$

$= 0.25 + 1.5$

$= 2.75$

X	Y	X_Pred	Residual
1	2	2.15	-0.15
2	4	4.18	-0.18
3	5	5.87	-0.87
4	4	4.76	-0.76
5	5	5.59	-0.59

$MSE = (0.15^2 + 0.18^2 + 0.87^2 + 0.76^2 + 0.59^2)/5$

$= 0.0225 + 0.0324 + 0.7569 + 0.5776 + 0.3481$

$= 1.7375 / 5 = 0.3475$

$RMSE = \sqrt{0.3475} = 0.5894$

Which means that our predicted values are 0.5894 units away from actual Points.

2.A Gradient Descent

- Aim of Linear Regression → "Find best fit line"
- Parameters defining best fit line → θ_1, θ_0 (slope & intercept)
- Aim of Gradient Descent → Find best "θ_1, θ_0" values
- By finding it we adjust the position of $(h_\theta(x))$ line.

- Hypothesis : $h_\theta(x) = \theta_0 + \theta_1 x$
- Parameters : θ_0, θ_1
- Cost Function : $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$
- Goal : minimize $J(\theta_0, \theta_1)$ → by adjusting (θ_0, θ_1)

Problem Setup :

- Assume we have the function $J(\theta_0, \theta_1)$
- We have to minimize θ_0, θ_1 using Gradient Descent.
- We initially start with some random guesses for θ_0, θ_1
- But it is recommended that θ_0 to be set to 0, and θ_1 to 1.
- We keep changing θ_0 and θ_1 such that we try to reduce $J(\theta_0, \theta_1)$, Until we wind at a minimum

What to assume ?

*) $\theta_0 = 0$

*) Prediction = Actual

When $\theta_0 = 0$, $h_\theta(x) = \theta_1 x$

$h_\theta(x) = \theta_0 + \theta_1 x$

$\theta_1 \rightarrow$ slope

$\theta_0 \rightarrow$ intercept

Case 1

$\theta_1 = 1$

(1,1) (2,2) (3,3) → data points

↓

perfect linear relationship.

$x = y \Rightarrow$ slope $= 1$

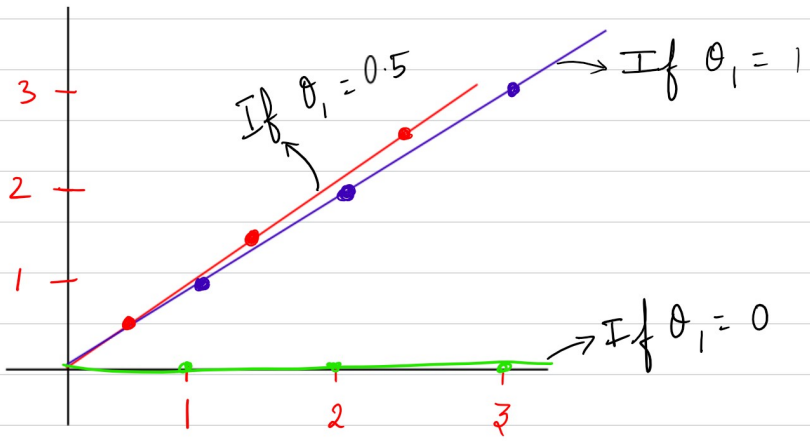
$$J(\theta_1) = \frac{1}{2m} \sum_{i=1}^{3} \left(h_\theta(x^i) - y^{(i)}\right)^2$$

$$J(\theta_1) = \frac{1}{2m} \left[(1-1)^2 + (2-2)^2 + (3-3)^2\right]$$

when $\theta_1 = 1$, $\boxed{J(\theta_1) = 0}$

Case 2

$\theta = 0.5$



$$J(\theta_1) = \frac{1}{2m} \sum_{i=1}^{3} \left[h_\theta(x^{(i)}) - y^{(i)}\right]^2$$

$$= \frac{1}{2m} \left[(0.5-1)^2 - (1-2)^2 - (1.5-3)^2\right]$$

$$J(\theta_1) = \frac{1}{2(3)} \left[3.5\right] \approx 0.58$$

when $\theta_1 = 0.5$, $\boxed{J(\theta_1) = 0.58}$

Case 3
$\theta = 0$

$$J(\theta_1) = \frac{1}{2m} \sum_{i=1}^{3} \left[h_\theta(x^{(i)}) - y^{(i)} \right]^2$$

$$= \frac{1}{2m} \left[(0-1)^2 + (0-2)^2 + (0-3)^2 \right]$$

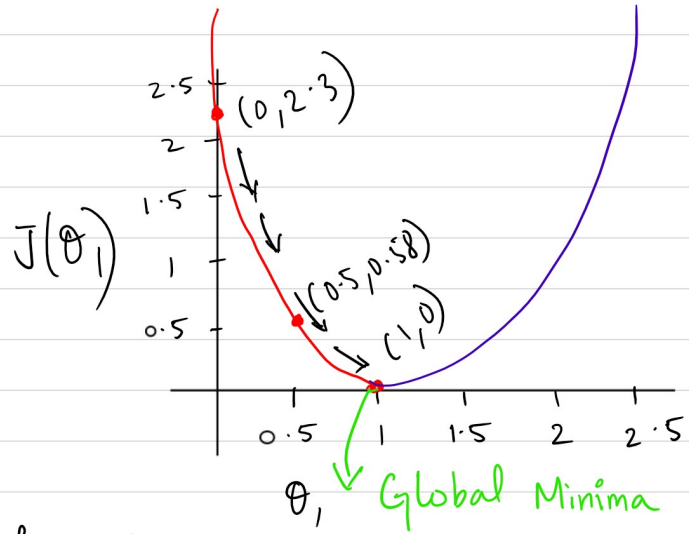
$$= \frac{1}{2\times 3} \left[1+4+9 \right] \Rightarrow \frac{1}{6} \left[14 \right]$$

when $\theta_1 = 0$

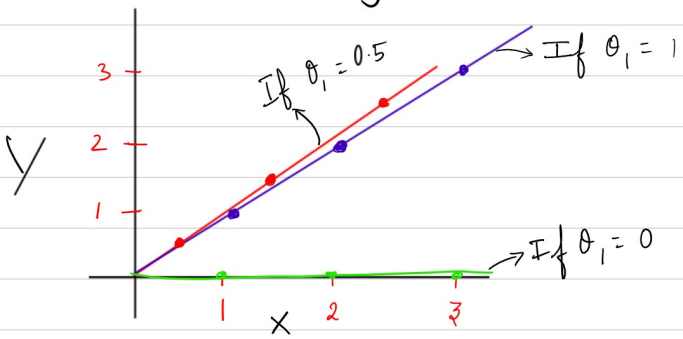
$$J(\theta_1) \simeq 2.3$$

$$\{ (1,0) , (0.5, 0.58) , (0, 2.3) \}$$

Case 1 Case 2 Case 3



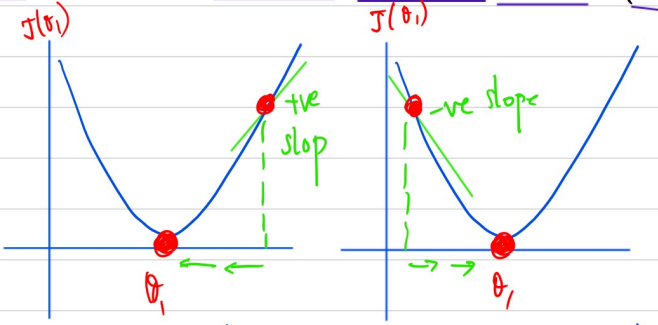
⇒ The idea is, θ_1 value is generated randomly and the algorithm automatically chooses the best fit.



⇒ The point at global minima $\{\theta_1 = 1, J(\theta_1) = 0\}$ is the best fit as it reached global minima.

⇒ So, for our examples $\{\theta_1 = 1\}$ was the best value when $\{\theta_0 = 0\}$

Convergence Algorithm



$\theta_1 := \theta_1 - \alpha$ (+ve)

$\theta_1 := \theta_1 - \alpha$ (-ve)

$\theta_1 := \theta_1 + \alpha$

Repeat Until Convergence

{

$$\theta_j := \theta_j - \alpha \boxed{\frac{\partial}{\partial_j} J(\theta_0, \theta_1)}$$

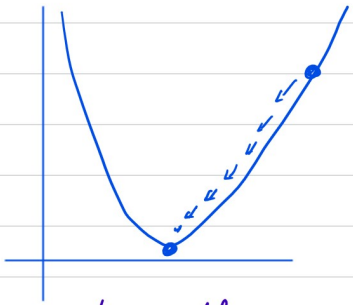
(α: learning rate)

→ derivative (slope)

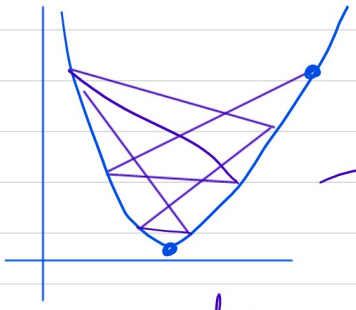
}

*) Find Slope of that particular point

*) Update $\theta_1 := \theta_1 - \alpha$ (+ve)



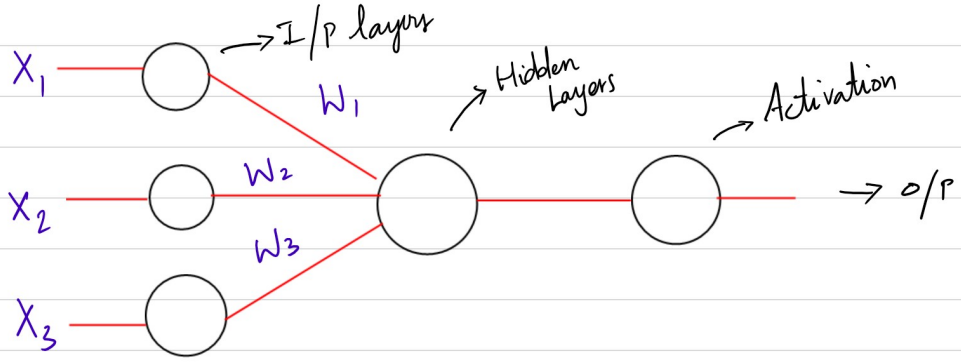
α = small



→ will never converge

α = large

2.5 PERCEPTRON {Single Layered Neural Networks}



a) It is a supervised learning algorithm that can be used as binary classifier.

b) It is a single layered neural network with four important parameters (i.e) input values, weights & bias, net sum, activation functions.

i) Weights → {how much the neuron should get activated}

ii) Activation Func. → {decides which neuron to activate}

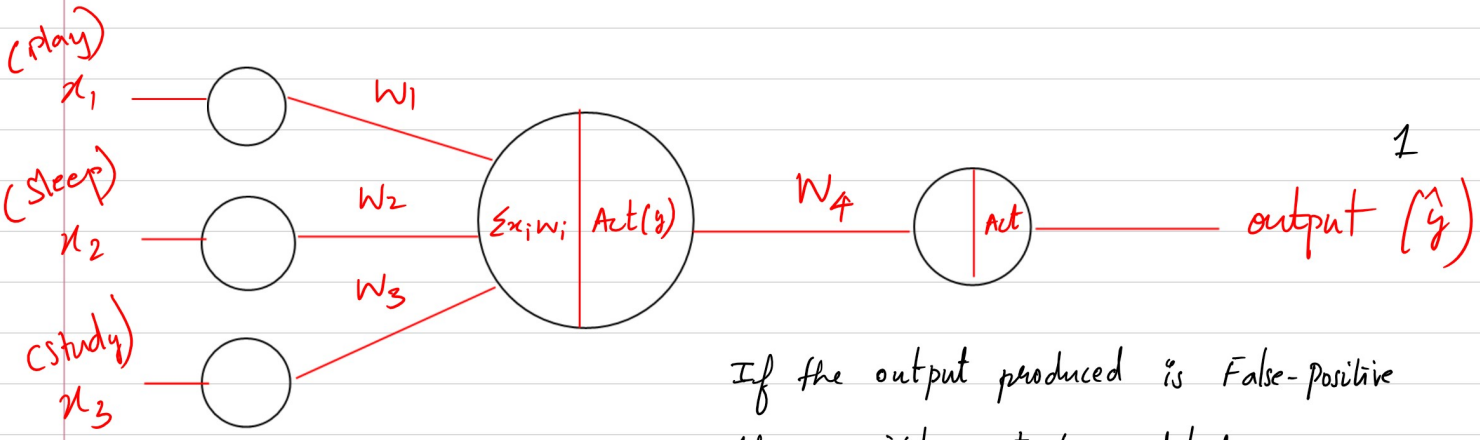
iii) Neuron → {Mathematical function that collects and classifies information}

{Best weights can be found using Gradient Descent}

Consider the following

Play	Sleep	Study	Result (y)
2	6	3	1
3	4	5	1
4	6	2	0

Perceptron Equation ⇒ $\sum x_i w_i = x_1 w_1 + x_2 w_2 + \cdots + x_n w_n$



If the output produced is False-positive then weights must be updated

I FWD Propagation

1. Multiply input with weights
2. Add bias
3. Apply Activation.
4. Produce 'y'

2. Until getting desired 'y'
1. Update the weights

II BWD Propagation

Apply Loss function $(y - \hat{y})$ = {output must be closen to 0} ← else

Aim: Compare predicted and actual values

) Weights will help the neurons to act whether it should be activated, and at what level it should get activated.

) Initially the weights passed is Zero, which is of no use, to overcome this we use something called bias.

Sigmoid Activation

) Decides which neuron to activate.

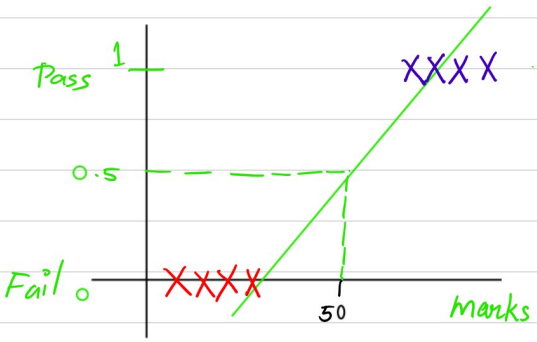
$$\text{Sigmoid} = \frac{1}{1+e^{-y}} \Rightarrow \frac{1}{1+e^{-(\Sigma x_i w_i + b)}} \Rightarrow \text{Outputs } \{0/1\}$$

< 0.5 will be considered 0, > 0.5 will be considered 1.

2.b Logistic Regression

Why Linear Regression can't be used for classification?

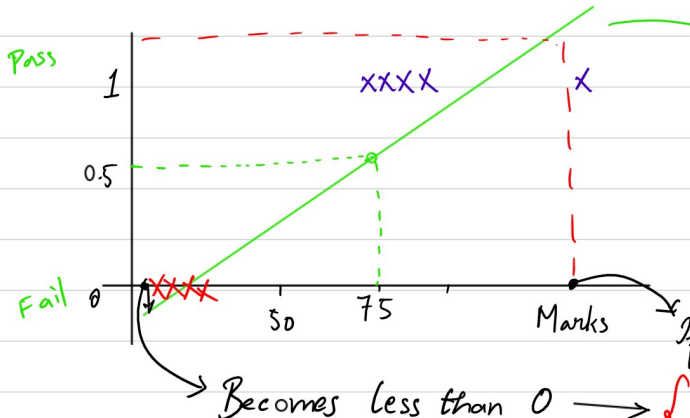
Consider the following



$$\left.\begin{array}{l} 1 \text{ if } h_\theta(x) \geq 0.5 \\ 0 \text{ if } h_\theta(x) < 0.5 \end{array}\right\}$$

*) From the above linear regression seems to be working.

*) But, the problem is it is prone to outliers, consider the below case.



→ The best fit line gets transformed due to the outlier.

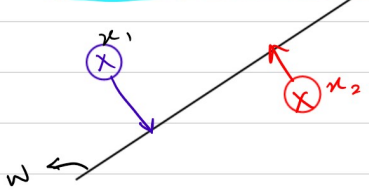
*) Due to the outliers we make wrong predictions.

If we try to predict this point goes above 1.

{which is not possible in Logistic Regression}

In logistic regression, it is told that the points must be linearly seperable

A plane in Linear Algebra.



$$\frac{\theta_1^T x + \theta_0}{\|w\|} \Rightarrow \theta_1^T x$$

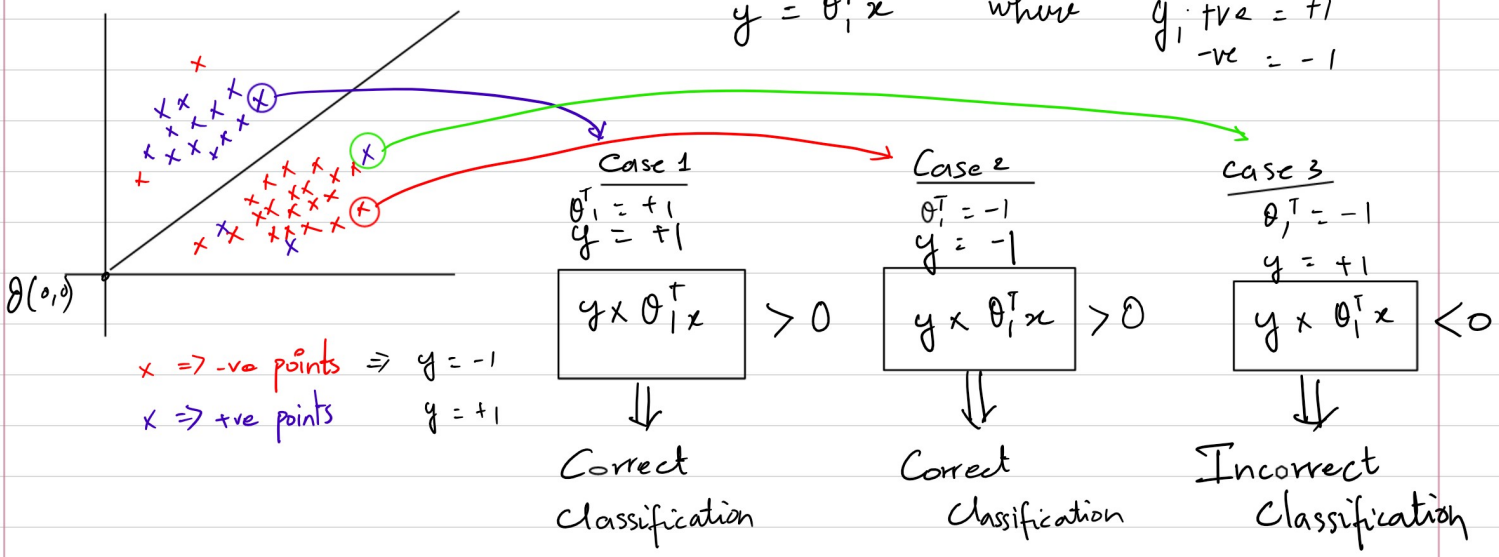
($\theta_1^T x + \theta_0 \to 0$, $\|w\| \to 1$)

the points above the plane will be +ve
points below plane will be -ve

x_1 → will give a +ve value
x_2 → will give a -ve value

⇒ Above the plane the point 'x' is considered positive.

⇒ Below the plane it is -ve



*) If the value is > 0 then the points are classified correctly.

Cost function Max $\sum_{i=1}^{n} y_i \theta_i^T x_i$

:) If we want to create the best fit line which linearly seperates the points, the summation of all the points and distance must be as maximum as possible.

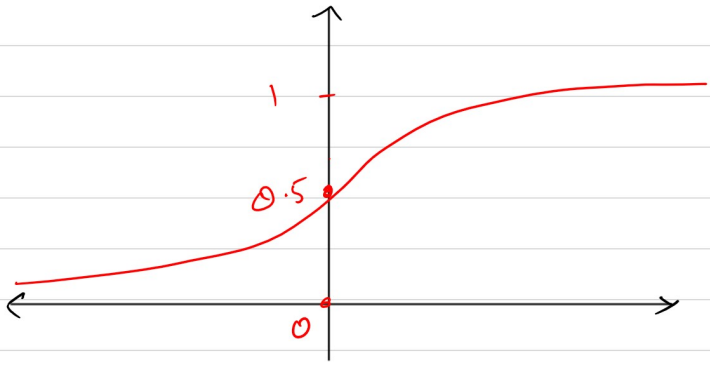
:) We have to update θ_i^T in such way that we get the maximum. Such that the line created will linearly classify the points

:) Again here we can have multiple best fit lines, but one with the maximum value will be selected.

Aim of Logistic Regression :→

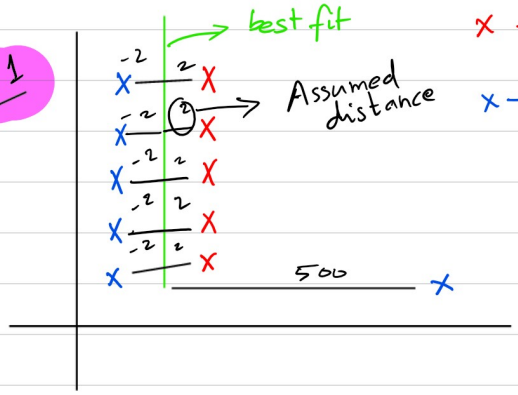
Find the $\max\left(\sum_{i=1}^{n} y_i \times \theta_i^T x_i\right)$

How Sigmoid is used with Logistic Regression?



Consider the following,

Case 1



$x \rightarrow +ve \Rightarrow \boxed{y_i = 1}$ Since above the plane.

$x \rightarrow -ve \Rightarrow \boxed{y_i = -1}$ Since below the plane

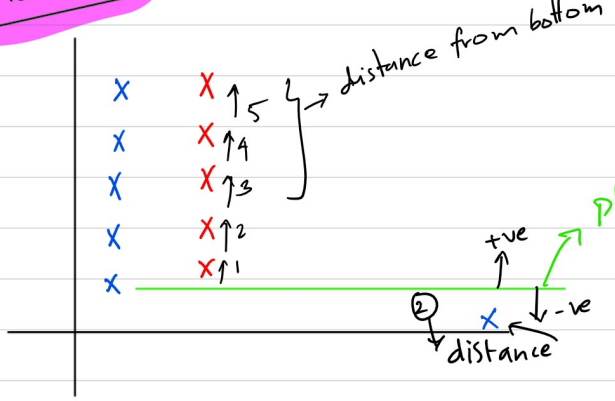
Calculating values using cost function.

$$\sum_{i=1}^{n}\left(y_i \times \theta_i^T x_i\right) \Rightarrow 2+2+2+2+2 + (-1(-2)+2+2+2+2) - 1(500) = -480$$

Since the value is -480 which means $\boxed{\sum_{i=1}^{n}\left(y_i \times \theta_i^T x_i\right) < 0}$

So, the line is not correctly classified.

Case 2



Applying $\sum_{i=1}^{n}\left(y_i \times \theta_i^T x_i\right)$

$$\Rightarrow \underbrace{-1-2-3-4-5}_{} + \underbrace{1+2+3+4+5}_{\text{+ve points}} + \underbrace{-1(-2)}_{}$$

$$\Rightarrow 0+2 = \boxed{2}$$

In case 2 we got $\max\left(\sum_{i=1}^{n} y_i \times \theta_i^T x_i\right) > 0$ but, not best fitting

So, we consider (case 1) where $\sum_{i=1}^{n} y_i \times \theta_i^T x_i = -500$

For this if we apply sigmoid $\sigma(z) = \frac{1}{1+e^{-z}}$

Our value would get reduced between 0 to 1.

$$\max \sum_{i=1}^{n} f\left(y_i \times \theta_i^T x_i\right) \rightarrow Z$$

2.7 Naïve Bayes

:) It is based on Bayer Theorem

Baye's Theorem

$$\underbrace{P(A/B)}_{\text{Conditional Prob.}} = \frac{P(A \cap B)}{P(B)} \quad ; \quad P(B/A) = \frac{P(B \cap A)}{P(A)}$$

$$P(A/B) \cdot P(B) = P(A \cap B) \quad ; \quad P(B/A) \cdot P(A) = P(B \cap A)$$

$$\Downarrow$$

$$P(A/B) \cdot P(B) = P(B/A) \cdot P(A)$$

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

$P(A/B)$ → Posterior Prob.

$P(B/A)$ → Likelihood prob

$P(A)$ → Prior Prob

$P(B)$ → Marginal Prob.

We have a dataset $x = \{x_1, x_2 \dots x_n\}$ $\{y\}$

When we apply naive bayes to our dataset

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

$$\Downarrow$$

$$P(y/x_1, x_2, \dots x_n) = \frac{P(x_1/y) \cdot P(x_2/y) \cdot P(x_3/y) \cdots P(x_n/y) \times P(y)}{P(x_1) \cdot P(x_2) \cdots P(x_n)}$$

$$= \frac{P(y) \prod_{i=1}^{n} P(x_i/y)}{P(x_1) P(x_2) \dots P(x_n)}$$

$$P(y/x_1, x_2 \dots x_n) \propto P(y) \prod_{i=1}^{n} P(x_i/y)$$

$$y = \text{argmax}_y \; P(y) \prod_{i=1}^{n} P(x_i/y)$$

The maximum value given will be chosen

if 'Yes = 0.7', 'No = 0.3'

↓ will be chosen

Example

OUTLOOK

	Yes	No	P(Y)	P(N)
Sunny	2	3	2/9	3/5
Overca	4	0	4/9	0/5
Rainy	3	2	3/9	2/5
tot	9	5	100%	100%

TEMPERATURE

	Yes	No	P(Y)	P(N)
HOT	2	2	2/9	2/5
MILD	4	2	4/9	2/5
COLD	3	1	3/9	1/5
tot	9	5	100%	100%

$P(Y) = 9/14$ $P(N) = 5/14$

Find Today (SUNNY, HOT)

$P(\text{Yes}/\text{Today}) = P(S/Y)\, P(H/Y) \cdot P(Y)$

$= 2/9 \times 2/9 \times 9/14$

$= 0.031$

$P(\text{NO}/\text{Today}) = 3/5 \times 2/5 \times 5/14$

$= 0.085$

Normalizing

$$P(\text{Yes}) = \frac{0.031}{0.031 + 0.095} \approx 0.27$$

$$P(\text{NO}) = \frac{0.085}{0.031 + 0.095} \approx 0.73$$

2.8 Support Vector Machines (SVM)

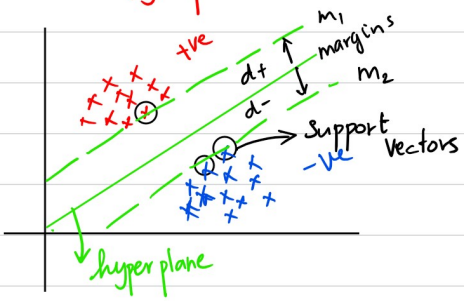
*) Supervised ML algorithm, useful for both Regr. & classification.

Geometrical Intuition

Geometric In

Aim w.r.t classification ⇒

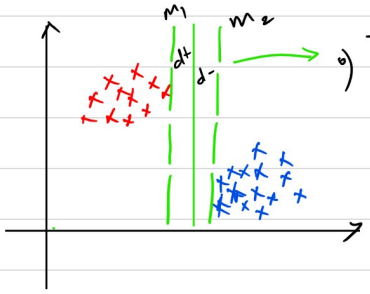
Classify points with a hyperplane with highest marginal distance.



*) A margin is used because it becomes easily linearly seperable.

*) After creating the hyperplane, we create two margins m_1 & m_2 which passes very through nearest points.

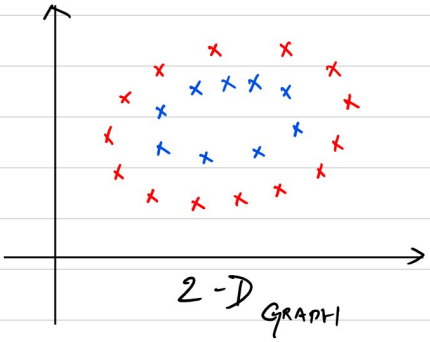
*) We can create multiple hyperplanes, but we have to choose the one that has the highest marginal distance.



*) The hyperplane can be drawn like this but the marginal distance is lower compared to previous example.

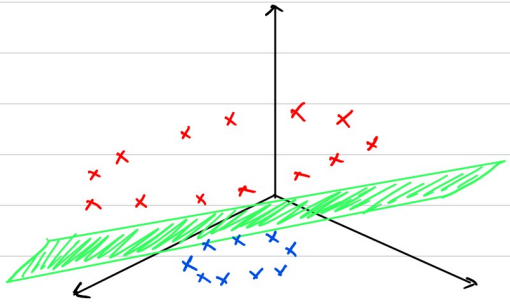
*) Which leads to a un-generalized model, which may not perform well with test data.

Consider the following case,



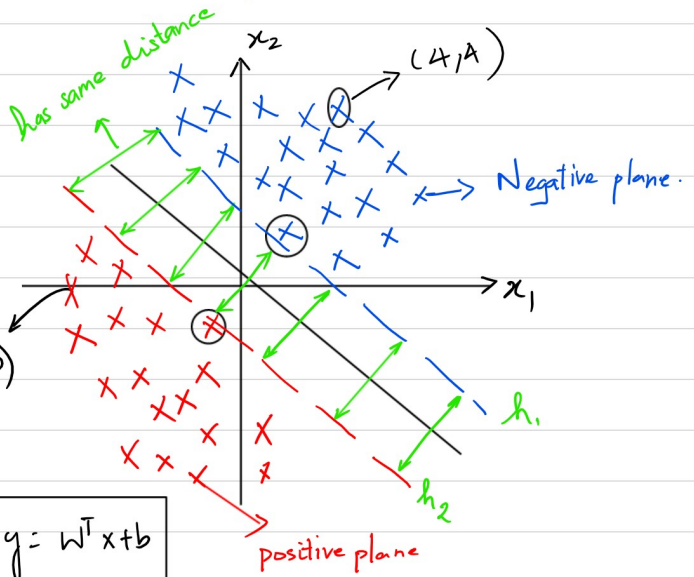
*) In this case the data is not linearly separable.

*) For this case we a SVM kernel, which tries to convert lower dimension to higher dimension.



Math behind SVM

Step 1



applying $y = w^T x + b$

Step 2

For Negative Plane,

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

$$= -4$$

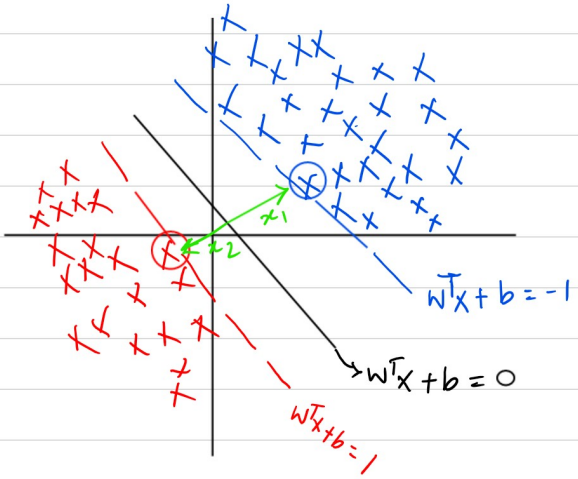
So, all other points in -ve plane will be considered negative

Step 3

Consider $b = 0$

$$= \begin{bmatrix} -1 \\ 0 \end{bmatrix} \begin{bmatrix} -4 & 0 \end{bmatrix} \Rightarrow 4$$

Since we get a positive value, for a point at the positive plane, we can consider that all other points will also be positive.



Step 4

$$w^T x_1 + b = -1$$
$$w^T x_2 + b = 1$$

(-) (-) (-)

$$w^T (x_2 - x_1) = 2$$

$$\frac{w^T}{\|w\|} (x_2 - x_1) = \frac{2}{\|w\|}$$

Since, we have to remove the slope transpose.

The magnitude of 'w^T' will be removed.

This is our optimization function, which we need to maximize.

Step 5

Update

$$(w,b) \max \frac{2}{\|w\|}$$ ← Optimization function.

y_i such that $\begin{cases} +1 & w^T x \geq 1 \\ -1 & w^T x \leq -1 \end{cases}$

→ But in real-life scenario its not possible

Step 6

Since, our previous method was overfitting we update the optimization function.

$$(w^*, b^*) = \min \frac{\|w\|}{2} + C \sum_{i=1}^{n} \xi_i$$

Regularization.

How many errors

Value of the error.

errors basically means the misclassified data points.

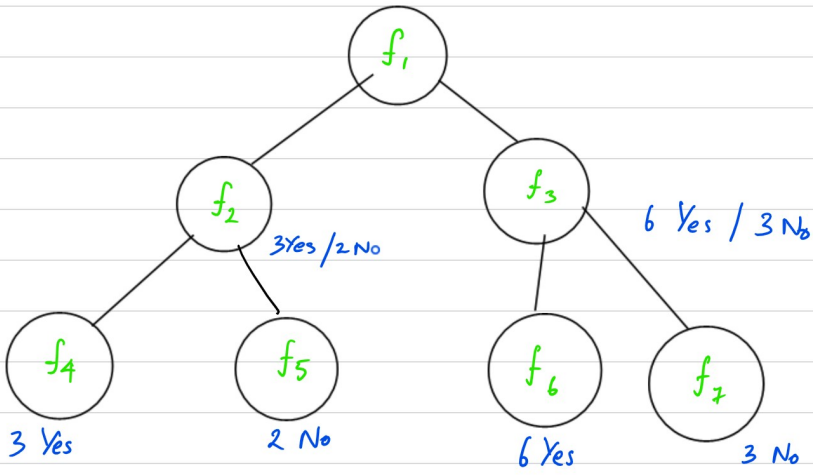
2.9 DECISION TREE

- There are some parameters $f_1, f_2, f_3 \ldots$.
- We have to construct a tree based on these attributes.
- Follows ID3 algorithm in which the first step itself is to choose the right attribute for the tree.
- And, that is when entropy comes into action.
- In order to consider which feature to select for splitting the node we use "entropy".
- Why its important is because if we choose the right node we can achieve the output in less number of steps.
- Entropy :- "Measures the purity of the split".
- Aim :- Reach the Leaf node Quickly.
- We have to calculate the purity of each split

ENTROPY :- $H(S) = -P_{(+)} \log_2(P+) - P_{(-)} \log_2(P-)$

P+ / P- ⇒ % of +ve class / % of -ve class

S : Subset of training example



Calculating Entropy for the above tree node "f_2"

$$= -\left(\frac{3}{5}\right) \log_2\left(\frac{3}{5}\right) - \left(\frac{2}{5}\right) \log_2\left(\frac{2}{5}\right)$$

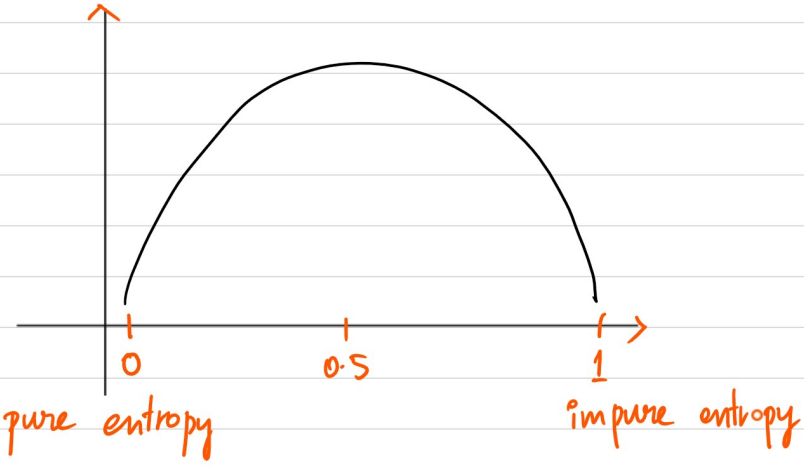
+ve class (Yes) — -ve class (No)

$= 0.72$

Completely impure subset, ex:- (3 Yes / 3 No) → Entropy will always be 1.

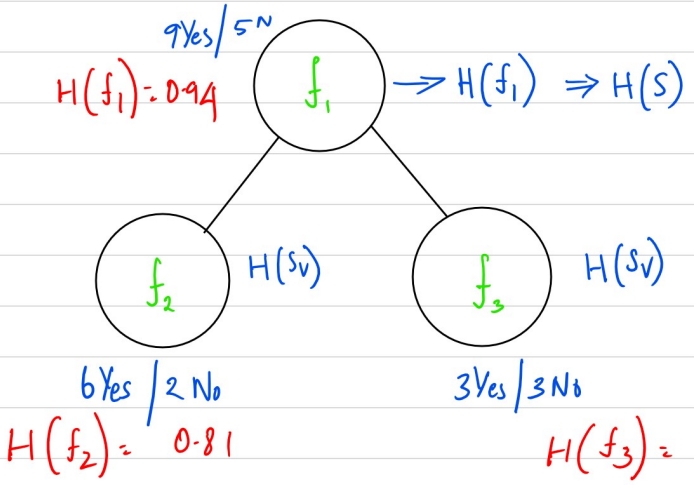
Which denotes that the tree split is very worst.

∴ The entropy value lies between 0 to 1. At choosing the nodes, the node with least entropy will be chosen



2.91 # Information Gain in Decision Tree

- It can simply said as the average of all the entropy value.
- So, that it can be used for finding a pattern in the tree which will take us to the leaf node as quickly as possible.
- $Gain(S,A) = H(S) - \sum_{v \in val} \frac{|S_v|}{|S|} H(S_v)$



Finding the information gain for the above tree.

$$Gain(S, f_1) = H(S) - \left(\frac{8}{14}\right) \cdot H(f_2) - \left(\frac{6}{14}\right) H(f_3)$$

Entropy of node. Sub node. total of parent node.

$$= 0.91 - \left(\frac{8}{14}\right) \times 0.81 - \left(\frac{6}{14}\right) \times 1$$

$$= 0.049$$

- The structure / path / pattern with highest 'Gain(S,A)' will be used.

Nithish S

BTech AI & DS

953621243039@ritrjpm.ac.in

2.92 Gini Impurity / Index

$$GI = 1 - \sum_{i=1}^{n} (P)^2$$

here $P \Rightarrow$ Prob of both +ve & -ve.

EX:- Consider our previous case, "f_2" which had 3 Yes / 2 No.

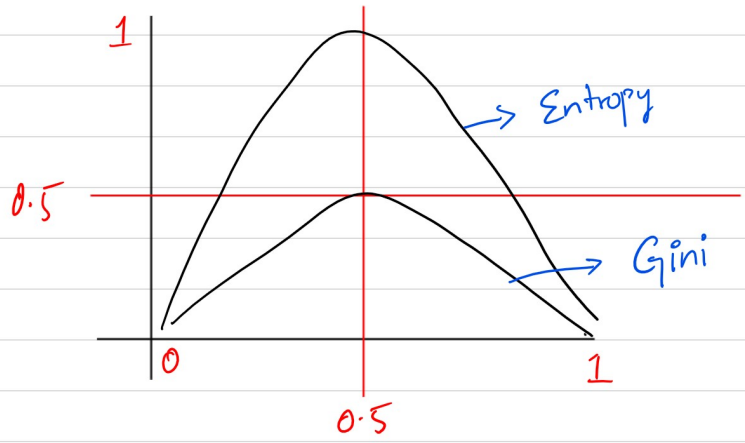
$$= 1 - \left((3/5)^2 + (2/5)^2 \right)$$

$$= 1 - (0.36 + 0.16)$$

$$= 1 - 0.52$$

$$\Rightarrow 0.48.$$

) Our "Entropy value was 0.72", Gini value = 0.48



Why to use Gini over Entropy?

) Computationally Efficient than entropy.

) Less prone to overfitting.

) Robust to outliers.

) Robust to irrelevant attributes.