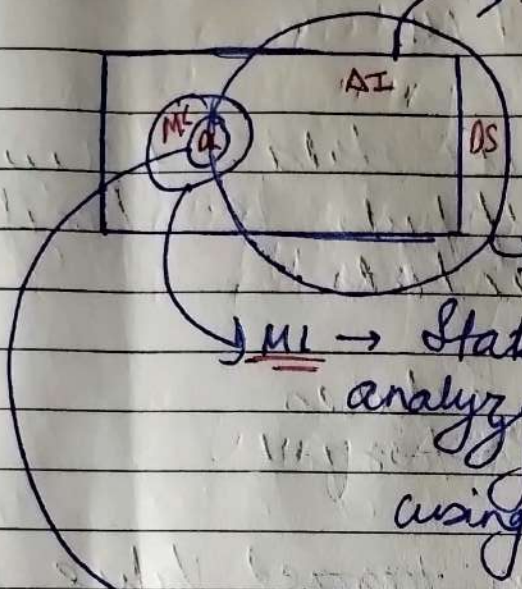
Date : Page :

{AI ML}

- Introduction
- AI VS ML VS DL VS DS



AI → An application that can do it's task without any human intervention.
Ex: (Netflix recommender system)
: (Self driving cars)

ML → Stats tool to analyze, visualize, forecast, predict using data

DL (Deep learning)
↳ trying to make a machine learn like human.
or simply to mimic human brain

Data Scientist DS : a person who analysis data then use any of the above to create a solution for a problem

Date: Page:

Machine Learning

- Supervised ML
- Unsupervised ML

In machine learning we are usually given a set of inputs called "features" and it's labeled outputs called "target values".

~~Supervised ML algo's are those which~~

We used Supervised ML algos ~~were~~ where we have access to both of these values. but, say if we only have input value & no labeled target Values, then we use Unsupervised ML algos.

Ex: → Student pass or Not { Input → Hours studied
Output → Passed / Fail

On the basis of given inputs, we can create and train a model to predict on new data if a student passe or not.

Date : Page :

Ex: ~~email classification (Spam or Not)~~

how ~~we need to~~ do an relation that makes an email ~~spam or not spam~~ & then predict on a new email that ~~we~~ might find.

ex: recommendation made by Differen app. Say Amazon:

Input : Recent purchases items in cart, wishlist

on basis of this input we need to recommend a use-s new products they would like to buy.

(Here we don't have target values as we don't know what a user might buy next).

Date : Page :

(Supervised ML)

This consist of mainly 2 type of problems :-

Regression
(output is a continuous
ex House price

Classification
(Yes/No or 0/1)
or 1/2/3/
(output is
(Categorical Value)
(Choices
Ex: Student pass

• Regression : It is a statistical method w attempts to find a relationship between a dependent variable & one more dependent variable.

• Classification : It is simply the allocatio of objects to certain pre-exis classes or categories. Specifically in M it use to categorise a given set o inputs.

Algorithms generally used :-

Regression : ① Linear Regression
② Lasso & Ridge
③ Elastic Net
④ Decision trees

Date: Page

5. Random Forest
6. Xgboost
7. ANN (Artificial Neural Networks)
8. RNN (Recurrent neural Networks)
9. ~~CNN~~

Classification : 1. Logistic
2. Lasso, Ridge
3. Decision Tree Classification
4. Random Forest Classification
5. Xgboost Classification
6. ANN
7. CNN → (Convolutional Neural Networks)

~~Most~~ algos used in Regression can also be used in Classification.
(many more algos exist and are used in Supervised ML)

(Still learning ...)

Date: Page:

Unsupervised ML

↳ Clustering Algos.
- → K-Means
- → DBSCAN
- → Hierarchical

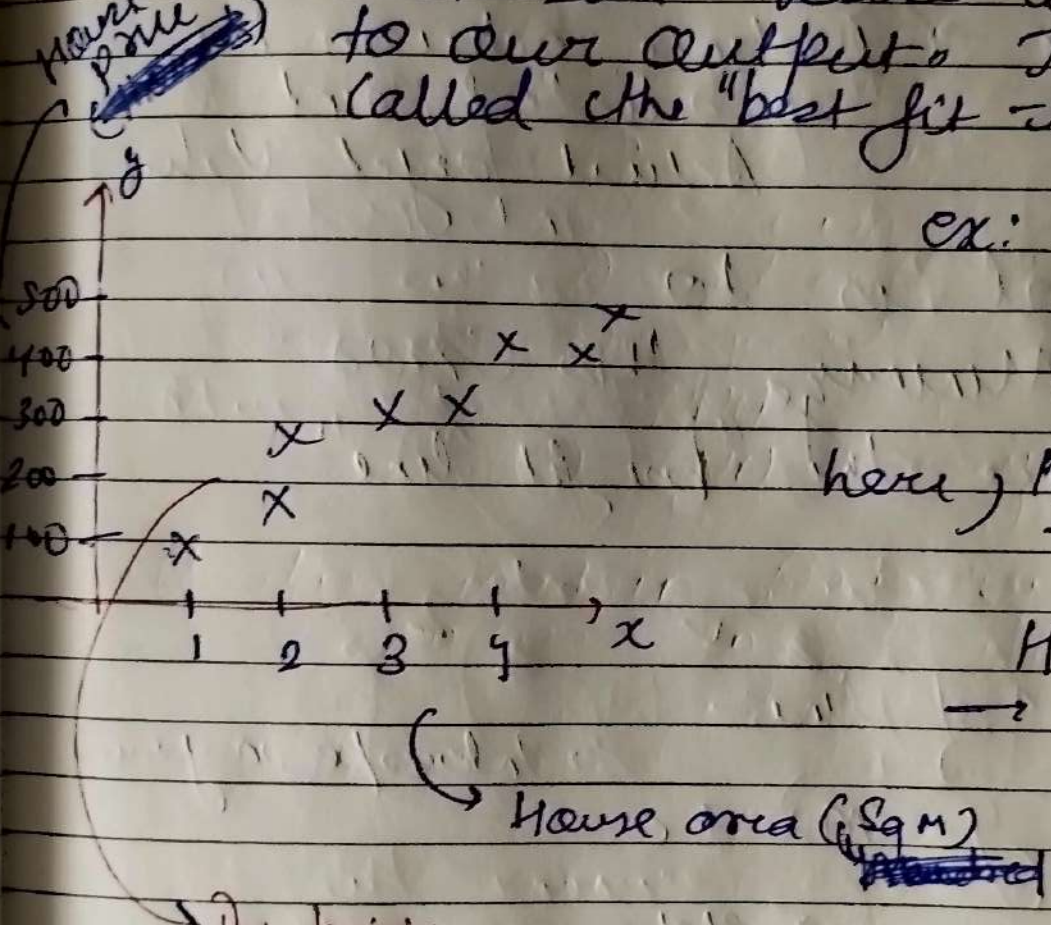
↳ Anomaly Detection
- Isolation Forest
- One Class SVM

(& many more categories & algorithms.)

(Still learning :-) :-)

Supervised ML

- Linear Regression
  - ↳ here we try finding an eqn that best relates our inputs to our output. This line is called the "best fit line".



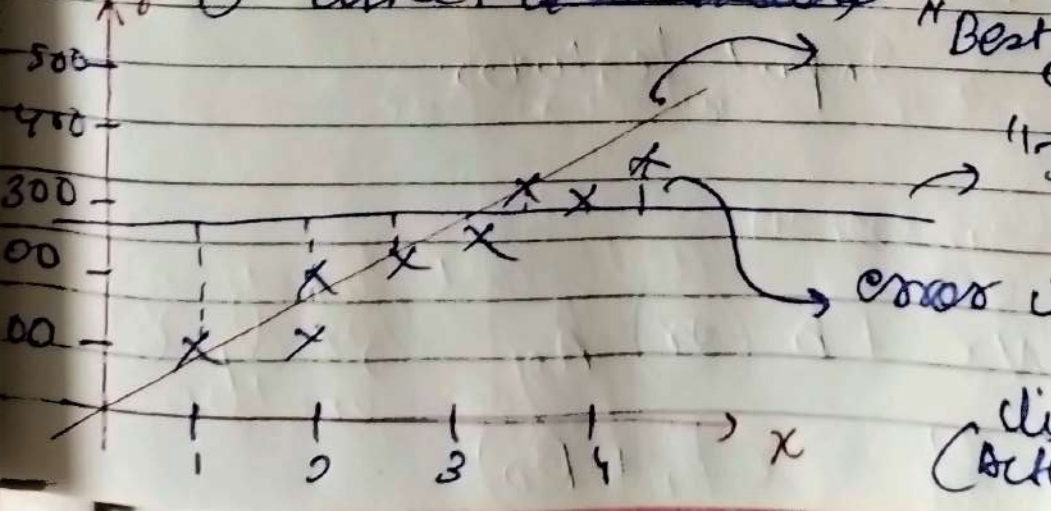
ex: House price vs House area.

here, House price → dependent var.

House area → Independent

Data points

through this data we need to fit an ~~best~~ line:- ~~the best line~~



Date: Page:

"Just a line": Doesn't accurately predict a relation between input (x) and output (y). So, it's prone to have a ton of error.

error → (Distance between the predicted point & Actual target value)

Our goal is to "Minimize this error" to produce the "best fit line"

General eq^n of line:-

$y = mx + c$

- c → Intercept (or y axis)
- m → slope
- prediction
- x → Input features

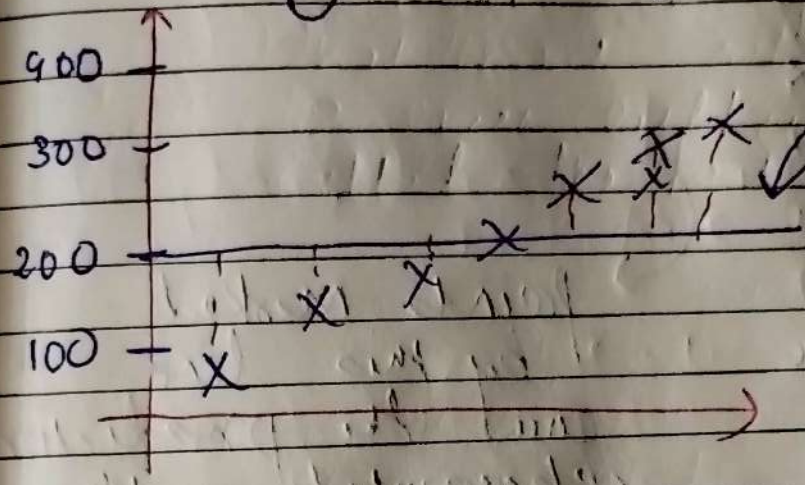
In linear Regression:-

$y_{pred} = wx + b$ (or) $h_\theta = \theta_0 + \theta_1 x$

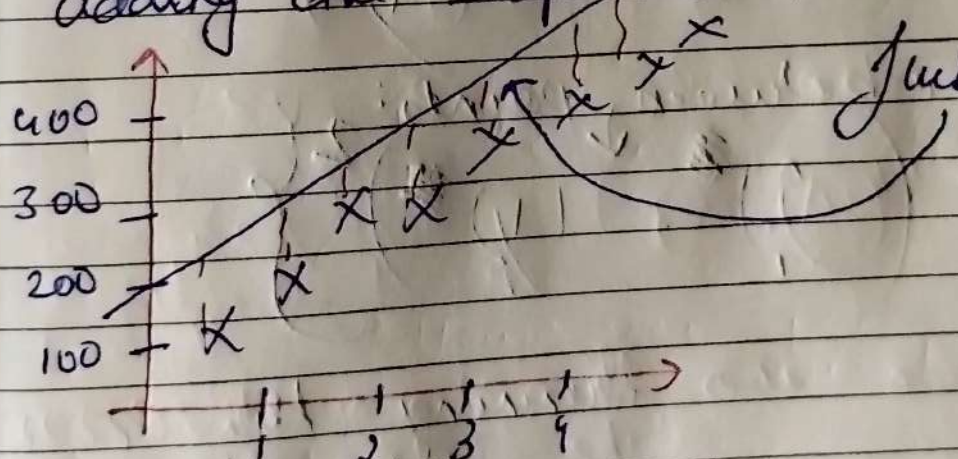
(or) $(y_i = x\beta_1 + \beta_0)$

9 use : $(f_{wb} = Wx + b)$

So, say $b = 200$, $W = 0$ $\{f_{wb} = 200\}$



adding the slope term :- $b = 200$, $W = 5$



So, to minimize this error we need to find correct values of W & b.

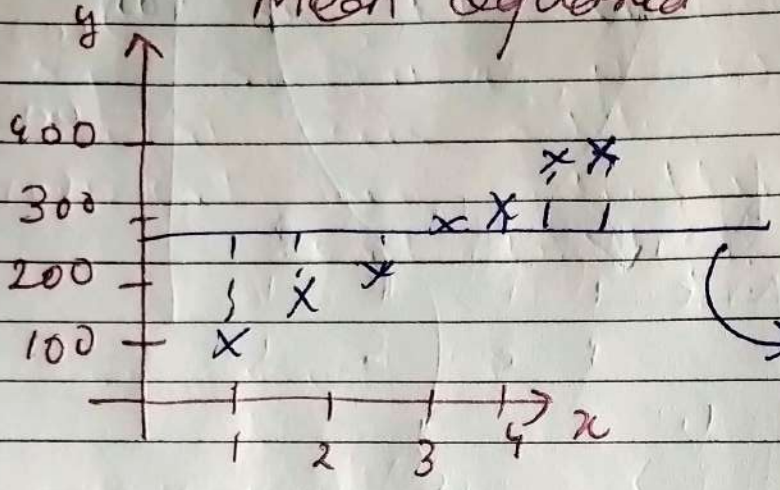
• More general eqn :-

$$\left(f_{Wb} = b + W_1 x_1 + W_2 x_2 + \cdots + W_n x_n \right)$$

Date : Page :

Computing the error :

Mean Squared Error :



points marked by this line and the prediction represented with $(\hat{y})$

Distance between points :

$$\frac{1}{2m} \sum_{i=1}^{m} \left\{ \left(\hat{y}_i - y_i \right)^2 \right\}$$

$\frac{1}{2m}$ → taking mean

$\sum_{i=1}^{m}$ → Summing this error over all the data points

$\left(\hat{y}_i - y_i \right)^2$ → here for Calculation ease

The above eq^n is called "Cost function" or "Loss function"

Date: Page:

Represented as :-

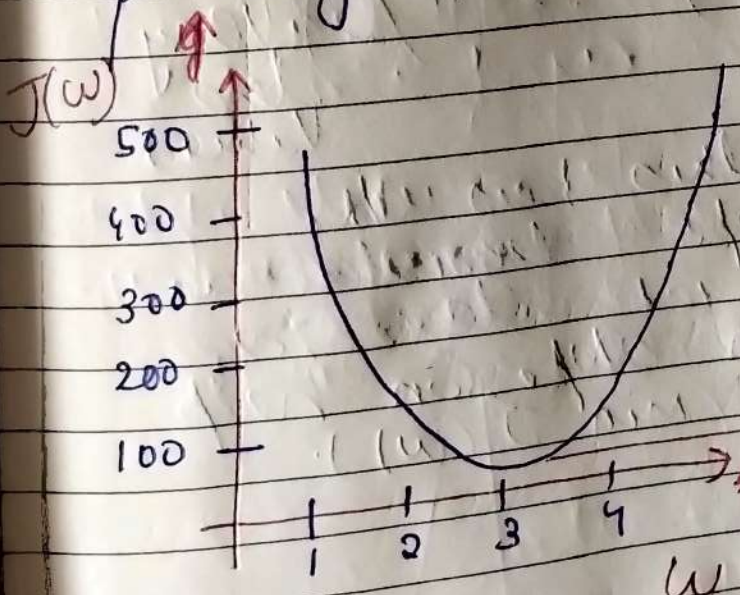
$$J(w,b) = \frac{1}{2m} \sum_{i=1}^{m} \left((\hat{y}_i) - (y_i) \right)^2$$

→ we need to minimize this cost.

Optimization Algorithm:

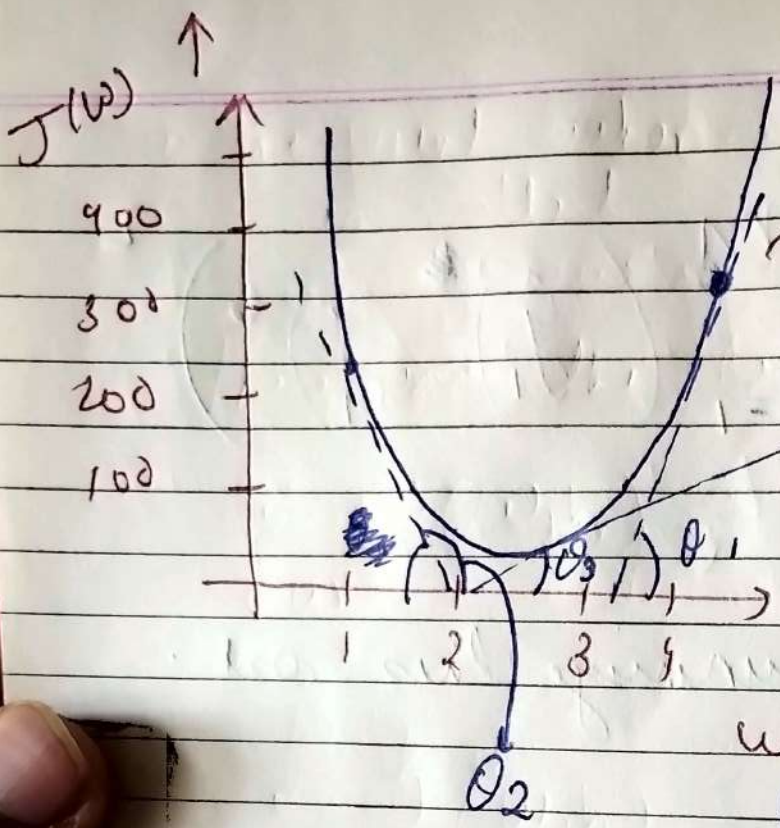
① "Gradient Descent Algo"

Plotting this $J(w,b)$ for different w values & $b=0$.

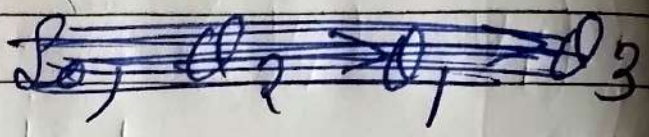


→ parabolic curve with a global minimum

→ at this global minimum we get the least value $J(w)$. we need to come how reach this point.



→ taking a random point & calculating it's slope



(slope)

now, $\tan(\theta_1) \rightarrow$ +ve

$\tan(\theta_2) \rightarrow$ -ve

$\tan(\theta_3) \rightarrow (+/-)$ (very less value)

→ we can use this, as the ~~slope decreases we get closer to~~ variation in slope can help get to (min J(w)).

We will simply :—

$$W_{new} = \left(W - \frac{2}{2W} J(W)\right)$$

to min J(w),

if slope is +ve ↦ $W \uparrow_+$ → getting closer to min J(w).

(if we are already near min J(w), ~~we will make the update~~ the value of derivative will be small, so no significant ↑+ or ↓-)

going from right to left → $W \downarrow_-$

↳ if we encounter min J(w)

↳ similarly left to right → $W \uparrow_+$

↳ to Quantify How much to move we use the α value of slope.

↳ Repeated updates to the parameters result in optimal W value

This algorithm is "gradient descent algo."

representation :-

"learning rate"

$$W_{new} = W_{old} - \alpha \frac{\partial}{\partial w} J(w,b)$$

$$b_{new} = b_{old} - \alpha \frac{\partial}{\partial b} J(w,b)$$

The point when it reaches min J(w,b) is called "Convergence" & we repeate our algo till then.

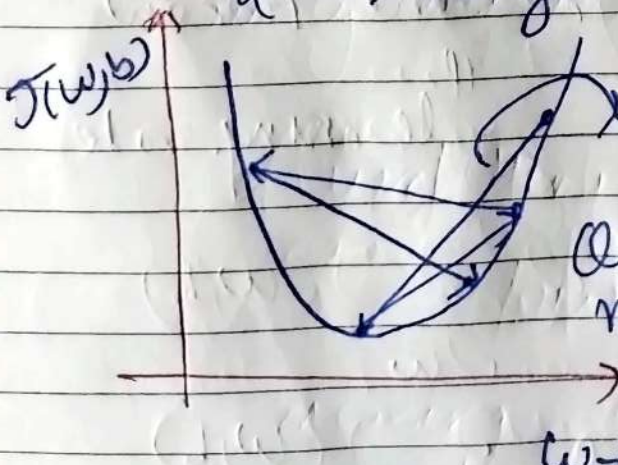
these are the derivative terms:-

$$\frac{\partial}{\partial w} J(w,b) = \frac{1}{m} \sum_{i=1}^{m} ((\hat{y}_i) - y_i) \cdot x_i$$

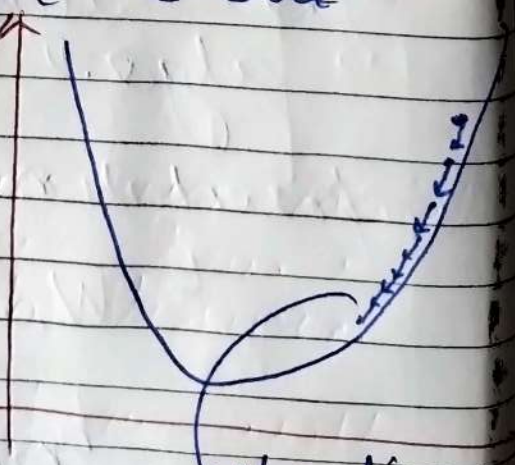
$$\frac{\partial}{\partial b} J(w,b) = \frac{1}{m} \sum_{i=1}^{m} ((\hat{y}_i) - y_i)$$

Learning Rate (α) : It decides how big of a step we take while updating of parameters (w,b)

α >> large



α <<< small



we need the pick alpha Dance where in the middle of this.

Matrix Notation of Linear Regression

$$y = XW + \varepsilon$$ (residuals)

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

$$X = \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_n^T \end{bmatrix} = \begin{bmatrix} 1 & x_{11} & \cdots & x_{1p} \\ 1 & x_{21} & \cdots & x_{2p} \\ \vdots & & & \\ 1 & x_{n1} & \cdots & x_{np} \end{bmatrix}$$

(for bias/intercept term)

no of features in each example

↳ total no of examples

$$W = \begin{bmatrix} b \\ w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_p \end{bmatrix}$$

→ intercept term

coefficients / slopes

another way to find the parameters

$$\vec{w} = (X^T X)^{-1} X^T y$$

{ Still dessing as to How this works } }.

Problem of Overfitting and Underfitting.

- Bias: 'error' in the prediction made by model.
- Variance: Specifies the amount our model when a different portion of data (or no unseen data) is used.

We, want our model to have low-Bias, & low Variance (though both simultaneously are not possible)

Overfitting occurs when: our model has ~~high Bias & low~~ low bias & high Variance.

meaning our model performs well on training data but fails on unseen new data

Date: Page:

It occurs when our model is trained only on training data & the optimization algo end up with over accurate parameters that work only to model real training data.

There are multiple way to fix this ex: (Splitting data to set (into train, test, cross valid)

(using regularization methods)

Underfitting: occurs when our model has high Bias & low Variance
i.e. it doesn't perform well on training data & there is no change in its performance on switching data.

Causes are poor training of model, wrong I/P etc.

(Ridge & Lasso Regression)

Solution to the problem of overfitting.
This is a regularization method.

Date: Page

Overfitting occurs when low bias & high Variance.
So, here we intentionally add some error to our model which results in more error on training data but also reduces Variance.

- Ridge Regression. (also called L2 Norm)
we add this term to our (MSE) or Cost function.

$$\left\{ + \frac{1}{2m} \left(\sum_{i=1}^{n} (W_i)^2 \right) \right\}$$

(n = no of features) ← → Regularization parameter.

& it's derivative wrt to W becomes

$$\left\{ + \frac{1}{m} \left(\sum_{i=1}^{m} W_i \right) \right\}$$

Date : Page :

- In addition to adding an bit of error, to more generalize the model.

Lasso Regression also helps in Reducing the dimensionality of our data.

i.e

it eliminates irrelevant features.

We add this term in Lasso: (also called L1 norm)

$$+ \frac{1}{p} \lambda \sum_{i=1}^{p} |w_i|$$

and it's Derivative cost w.

(now as mod doesn't have a proper Derivative we use *Subderivative*.

$$+ \lambda \sum_{j=1}^{p} \text{sign}(w_j)$$

- $\text{sign}(w_j) = \begin{cases} 1, & \text{if } w_j > 0 \\ -1, & \text{if } w_j < 0 \\ 0, & \text{if } w_j = 0 \end{cases}$

Date : Page :

Difference between the 2 is
→ Ridge introduce more errors,
→ Lasso introduce less error than Ridge but helps in Reducing dimension.

Now, The Combination of the above 2 becomes :-

(Elastic net regularization
↳ having good point of both)

this is the term added to cost function :-

$$+ \frac{1}{2p} \lambda_1 \sum_{j=1}^{p} (W_j)^2 + \frac{1}{p} \lambda_2 \sum_{i=1}^{p} |W_i|$$

and derivative cost to w :-

$$+ \frac{1}{p} \lambda_1 \sum_{j=1}^{p} W_j + \frac{1}{p} \lambda_2 \sum_{i=1}^{p} \text{sign}(W_i)$$

Date: Page:

The above is also used in this form:-

$$\frac{1-\alpha}{2p}\sum_{j=1}^{p}(w_j)^2 + \frac{\alpha}{p}\sum_{j=1}^{p}|w_j|$$

→ where α is "mixing parameter".

(if $\alpha = 1$, Elastic Net performs Lasso

$\alpha = 0$, Elastic Net performs Ridge

$0 < \alpha < 1$, (combination of both))

Date: Page:

- In Ridge Regression, the penalty reduces the values of large coefficient but it never goes to zero.
- but, in Lasso this goes to zero resulting in zero parameters reducing to 0

(I still am not entirely clear as to why Lasso goes to 0 ...)

→ Linear Regression itself also has many variants. —

(1) Simple Linear Regression → (with 1 feature)

(2) Multivariable Linear Regression
↳ (with multiple feature)

(3) Multivariate
↳ predicting more than one variable on the basis of multiple input features
(ex: (predicting if student passes in English & Hindi, simultaneously)

Date: Page:

④ Polynomial Regression: Fitting a data with higher degree relation
ex: $(y = x^2)$

(many more still learning - 11)

→

Date: Page:

{Logistic Regression}

↳ Here we try classifying either a continious set of inputs or a set of binary inputs into one of 2 category usually 1/0 or (True, False)

We used the normal line's eqn to find the value of a prediction but as we have to classify this input in terms of 0 & 1 we use

Sigmoid function
(it a case of logistic function)

(logistic growth)

$$p(x) = \frac{L}{1 + e^{-(x - x_0)k}}$$

L → Supremum of the values of function.

x_0 → location parameter which means mid-point of curve where $(p(x_0) = 1/2)$

k → logistic growth rate

when the value of

$L=1, k=1, x_0=0$

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

↳ Sigmoid function

In our Code:

~~f(z)~~

$$f(w,b) = \frac{1}{1+e^{-(wx+b)}}$$

$(z = wx+b)$

↳ this function squashes value between (0 & 1).

Date: Page:

a more accurate way of writing sigmoid function, as we won't always have $(\mu = 0)$

$$p(x) = \frac{1}{1 + e^{-(x-\mu)/s}}$$

→ Scale parameter
↓
Controls the spread of the distribution

↓
(midpoint of curve, where $p(\mu) = 1/2$)

and when written with $f(w,b)$:-

$$f(w,b) = p(x) = \frac{1}{1 + e^{-(b + wx)}}$$

$$\begin{cases} b = -\mu/s \\ \text{Intercept} \end{cases}, \quad \underbrace{w = 1/s}_{\text{Slope}}$$

$$\left(\mu = -b/w, \; s = 1/w\right)$$

while, implementing we mostly care about.

$$f(w,b) = \frac{1}{1+e^{-(wx+b)}}$$

now, here also the initial sigmoid might not "fit" our data well. So, we have to reduce the error here as well.

But, Linear Regression's Cost function won't work here. we need to find something else:-

we define.

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m} \text{Cost}\left(f_{(w,b)}, y^{(i)}\right)$$

(new one)

$$\begin{cases} -\log(f(w,b)) & \text{if } y=1 \\ -\log(1-f(w,b)) & \text{if } y=0 \end{cases}$$

Combining we get this:

$$\Rightarrow -y \log(f(w,b)) - (1-y)\log(1-f(w,b))$$

while implementing we mostly care about.

$$f(w,b) = \frac{1}{1+e^{-(wx+b)}}$$

now, here also the initial sigmoid might not "fit" our data well. So we have to reduce the error here as well.

But, Linear Regression's Cost function won't work here. We need to find something else:-

we define.

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m} \text{cost}\left(f(w,b), y^{(i)}\right)$$

(new one)

$$\begin{cases} -\log(f(w,b)) & \text{if } y=1 \\ -\log(1-f(w,b)) & \text{if } y=0 \end{cases}$$

Combining we get this:

$$\Rightarrow -y\log(f(w,b)) - (1-y)\log(1-f(w,b))$$

also known as binary cross entropy

Finally:-

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m}\left(-y_i \log(\hat{y}_i) - (1-y_i)\log(1-\hat{y}_i)\right)$$

reduces to

$-\log(\hat{y}_i)$ for $y_i = 1$

& to

$-\log(1-\hat{y}_i)$ for $y_i = 0$

and it's total sum is also called (-ve log likelihood)

Now why this & how does the penalizes the input if it has a large error.

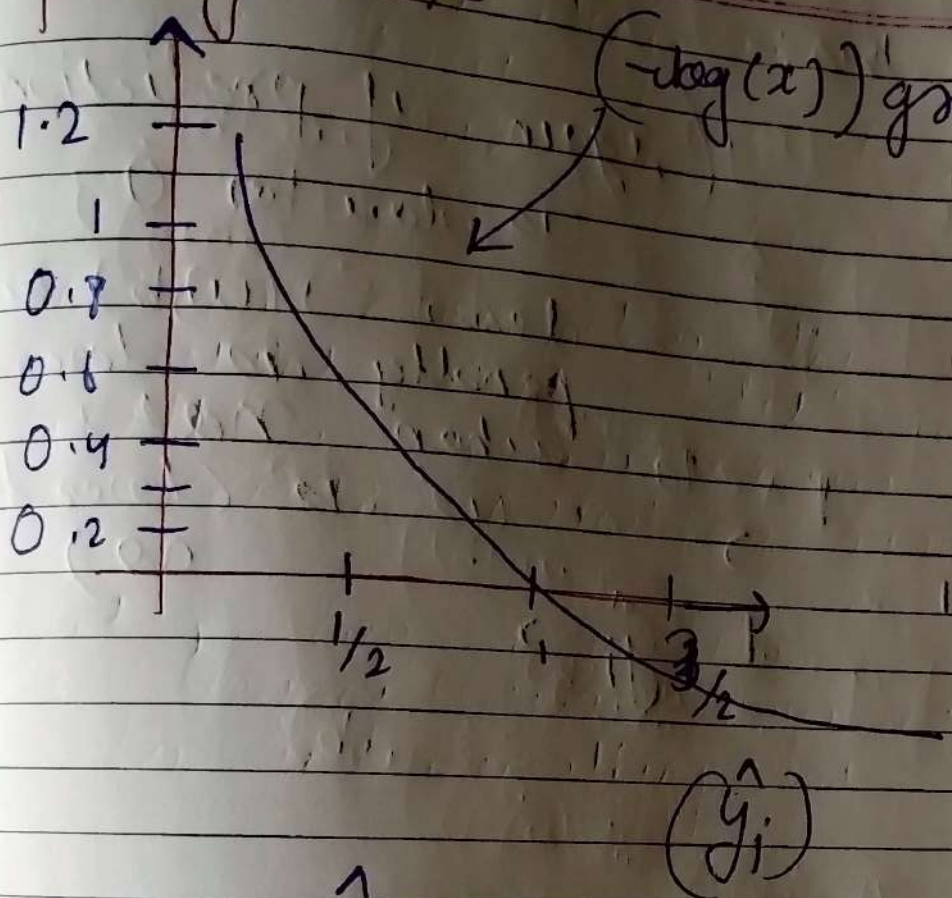
The function is in 2 parts for 4 reason:-

when $(y_i = 1)$, we want our value to be as close to true value as possible.

So this function works:-

$-\log(\hat{y}_i)$

plotting this:-



when, $\hat{y}_i$ is close to 1, $J(w,b) \to 0$

& $\hat{y}_i$ is close to 0, $J(w,b) \to \infty$

heavily penalizing the wrong predictions.

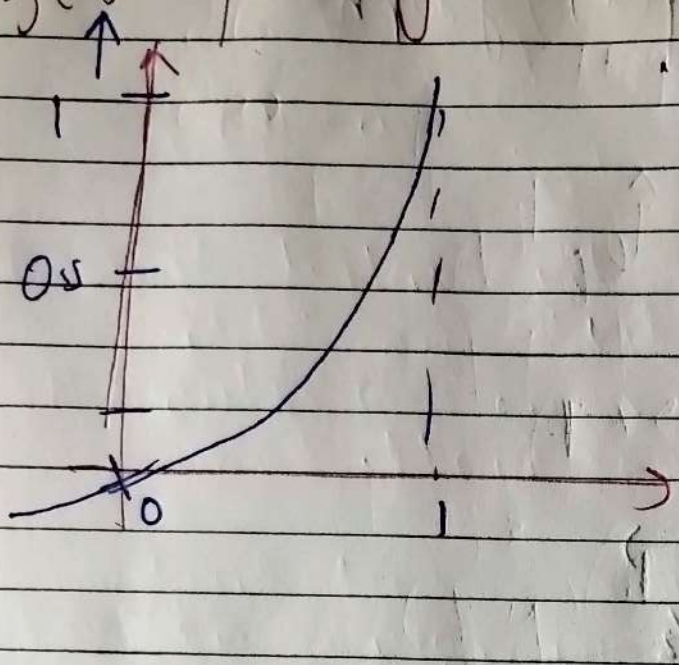
Similarly when $(y_i = 0)$,

this function works

$$-\log(1-\hat{y}_i)$$

Date : Page :

$J(w,b)$: plotting this :-



(again if prediction is close to 0, then loss $J(w,b)$ or penalty is closer to 0 & when not, it is closer to ∞)

Gradient Descent remain same as linear Regression.

as when we take derivate of above Cost function we get the same value as linear regression, only now definition of $f(w,b)$ is different

• Let's find the derivative :-

$$A = -y * \log(\hat{y})$$

$$= -y * \log\left(\frac{1}{1+e^{-(wx+b)}}\right)$$

$$\frac{\partial A}{\partial w} = -y * \frac{1}{1+e^{-(wx+b)}} \underbrace{\frac{\partial}{\partial w}\left(\frac{1}{1+e^{-(wx+b)}}\right)}$$

$$= -y * \left(1+e^{-(wx+b)}\right) + \frac{1}{1+e^{-(wx+b)}}\left(1 - \frac{1}{1+e^{-(wx+b)}}\right)$$

$$\frac{-1 \cdot e^{-(wx+b)} \cdot x}{\left(1+e^{-(wx+b)}\right)^2}$$

$$= -y \cdot \frac{e^{-(wx+b)}}{\left(1+e^{-(wx+b)}\right)} \cdot x$$

$$= -y\left(\frac{1+e^{-(wx+b)} - 1}{1+e^{-(wx+b)}}\right) x$$

$$= -yx\left(1 - f(wx+b)\right)$$

or

$$\hat{y}$$

Date : Page :

$$\frac{\partial B}{\partial w} = -(1-y) \cdot \log(1-\hat{y})$$

$$\frac{\partial B}{\partial w} = \frac{+(1-y)}{1-\hat{y}} \cdot +\frac{\partial}{\partial w} \frac{1}{1+e^{-(wx+b)}}$$

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

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

$$= (1-y) \cdot x \cdot f(wx+b)$$

now for $\frac{\partial (\cos L)}{\partial w} = \frac{\partial A}{\partial w} + \frac{\partial B}{\partial w}$

$$= -yx(1 - f(wx+b)) + (1-y) \cdot x \cdot f(wx+b)$$

$$= -yx + yx f(w,b) + x f(w,b)$$

$$- yx - f(w,b)$$

$$= (f(w,b) - y)\, x$$

$$= (\hat{y} - y) \cdot x$$

Equal to the term in Linear regression.

Similarly we can find b.

$$\frac{\partial A}{\partial b} = -y \cdot \log(\hat{y})$$

$$= -y \cdot \frac{1}{\hat{y}} \cdot \frac{\partial}{\partial b} \hat{y}$$

$$= -y \cdot \frac{1}{\hat{y}} \cdot \frac{e^{-(wx+b)}}{(1+e^{-(wx+b)})^2}$$

$$\frac{\partial A}{\partial b} = -y \cdot \frac{e^{-(wx+b)}}{(1+e^{-(wx+b)})^2}$$

Date : Page :

$$\frac{\partial B}{\partial b} = -(1-y) \log(1-\hat{y})$$

$$= +\frac{(1-y)}{1-\hat{y}} \cdot \frac{\partial}{\partial b}(\hat{y})$$

$$= \frac{(1-y)\,(1+e^{-(wx+b)})}{e^{-(wx+b)}} \cdot \frac{e^{-(wx+b)}}{(1+e^{-(wx+b)})^2}$$

$$\frac{\partial Cost}{\partial b} = \frac{\partial A}{\partial b} + \frac{\partial B}{\partial b}$$

$$= -y\,\frac{e^{-(wx+b)}}{1+e^{-(wx+b)}} + \frac{(1-y)}{(1+e^{-(wx+b)})}$$

$$= \frac{1 - y(1+e^{-(wx+b)})}{(1+e^{-(wx+b)})}$$

$$= (f(w,b) - y)$$

or $(\hat{y} - y)$

another way of minimizing loss is to maximize the inverse of (-ve log likelihood)

meaning (+ve log likelihood)

Still learning this—

Finally to convert the final squashed value to a probability 0 or 1 we use a "Decision Boundary"

or a threshold line, above which model predicts 1, below which 0. usually 0.5

usually we take ($y = 0.5$)

Date : Page :

Naive Bayes Classifier

"Based on Bayes' Theorem:

It describes the probability of an event, based on prior knowledge of condition that might be related to the event.

→ "likelihood"

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

(PRIOR)

→ Conditional probability → probability of A happening if B happened.

aka posterior probability.

+ (Similarly for $P(B/A)$)

$P(A)$ & $P(B)$ → probability of A & B respectively without any condition.

For Machine learning :-

Say we have to predict (y) given that $(x_1, x_2 \cdots x_n)$ happended.

So, $P\left(\frac{y}{x_1, \cdots x_n}\right) = \frac{P(x_1 \cdots x_j / y)\, P(y)}{P(x_1 \cdots x_j)}$

to understand how we use this, let's take the example of :-

"whether to play or not, on the basis of "Outlook", "Humidity", "Temp", "Windy".

Date : Page :

Outlook	Temp	Humidity	Windy	play
Rainy	H	Hi	F	no
R	H	Hi	T	no
OC	H	Hi	F	yes
S	M	Hi	F	yes
S	C	N	F	yes
S	C	N	T	no
OC	C	N	T	yes
R	M	Hi	F	no
R	C	N	F	yes
S	M	N	F	yes
R	M	N	T	yes
OC	M	Hi	T	yes
OC	H	N	F	yes
S	M	Hi	T	no

{ R → Rainy, H → hot, Hj → High
OC → Ques Cost, M → Mild, N → Normal
S → Sunny, C → Cool }

now we can further simplify the eqn.

$$P(y/x_1, x_2, x_3 \cdots x_n) \Rightarrow$$

$$P(y) * \frac{P(x_1 x_2 x_3 x_4 \cdots x_n / y)}{P(x_1, x_2, x_3, x_4 \cdots x_n)}$$

$$\Rightarrow P(y) * \frac{P(x_1/y) * P(x_2/y) * \cdots P(x_n/y)}{P(x_1) P(x_2) \cdots P(x_n)}$$

Say $y \rightarrow$ (T or F)

then we can write :-

$$P(T/x_1, x_2, \cdots x_n) \Rightarrow P(T) * P(x_1/T) \cdot P(x_2/T) \cdots P(x_n/T)$$

$$\frac{}{P(x_1) P(x_2) \cdots P(x_n)}$$

similarly for $P(N/x_1, x_2, \cdots x_n)$

(* We can ignore the denominator as it is constant for all)

So, now we compare these values:

(PLAY)		P(Yes)	P(NO)
Yes →	9	9/14	5/14
NO →	5		

(Outlook)

	Yes	NO	P(Yes)	P(NO)
Sunny	2	3	2/9	3/5
Overcast	4	0	4/9	0/5
Rainy	3	2	3/9	3/5
	9	5		

→ P(Sunny / Yes)

(Temperature)

	Yes	NO	P(Yes)	P(NO)
Hot	2	2	2/9	2/5
mild	4	2	4/9	2/5
Cool	3	1	3/9	1/5
	9	5		

(Humidity)

	(Yes)	(No)	P(Yes)	P(No)
High	3	4	3/9	4/5
Normal	6	1	6/9	1/5
	9	5		

(Wind)

	Yes	No	P(yes)	P(No)
Weak (F)	6	2	6/9	2/5
Strong (T)	3	3	3/9	3/5
	9	5		

(OR) $P(\text{Weak}/\text{Yes}) \text{ or } P(\text{Strong}/\text{Yes})$

Now, Say we need to predict if to play on a day with:-

$X = (\text{Sunny}, \text{Hot}, \text{High}, \text{Strong})$

$P(\text{Yes}/X) = 9/14 * 3/9 * 2/9 * 3/9 * 3/5$

$\Rightarrow 0.00529$

$P(\text{No}/X) = 5/14 * 3/5 * 2/5 * 4/5 * 3/5$

0.02742

$\Rightarrow$

Date: Page:

We can't really understand anything from this value so,

$$P(\text{Yes}/\bar{X}) = \frac{0.00589}{0.00589 + 0.02746}$$

$$= 0.1729 \times 100$$

$$= 17.29\ \%$$

$$P(\text{No}/\bar{X}) = 0.92678 \times 100$$

$$= 92.678\ \%$$

$$= 83.82\ \%$$

So, our prediction will be (No)