Logistic Regression

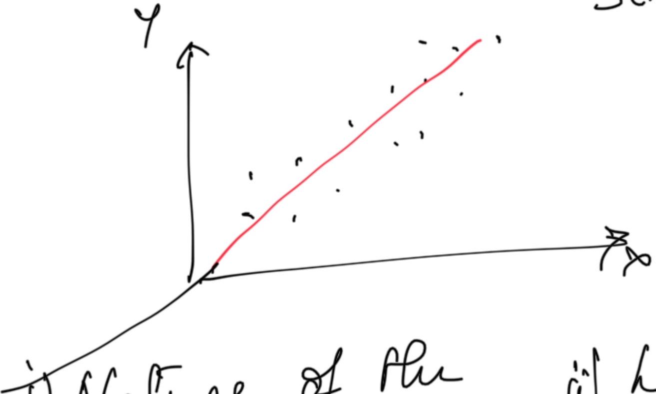
X variables — Independent

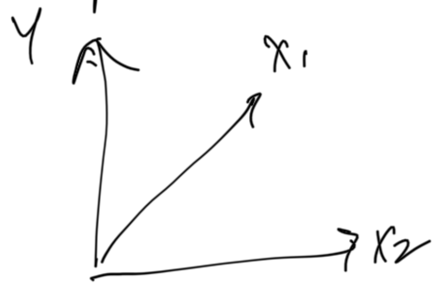
Y variables — Target

⇒ ① Simple linear regression.

② Multiple regression

Simple | Multiple





i) Nature of the algorithm

ii) Loss function.

⇒ Linear Regression — Y $(-\infty, \infty)$

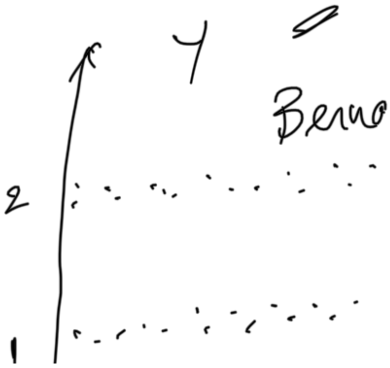
↓

Independent from

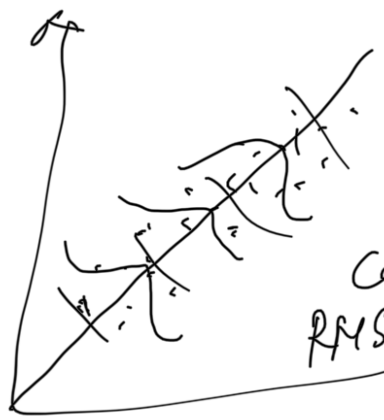
abs( )

Linear Regression
↓ Line
↓ Relationship

* Cases where Linear Reg. fail — Normal



Bernoulli distribution.



Performance can be ga

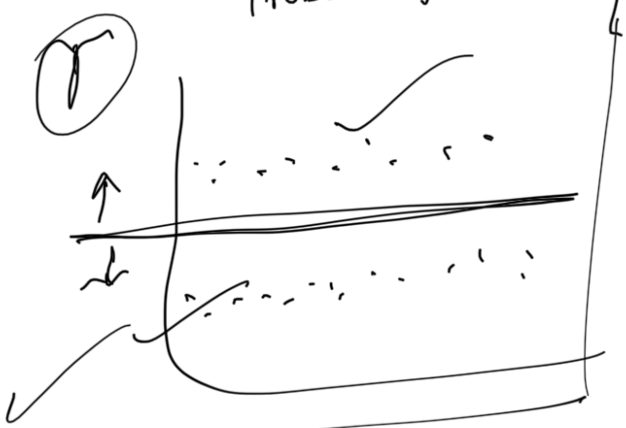
RMSE/MSE // R^2

$Y(0,1) \rightarrow$ (Spam/Ham) $\qquad Y(-\infty, \infty) \Downarrow$ Reg.

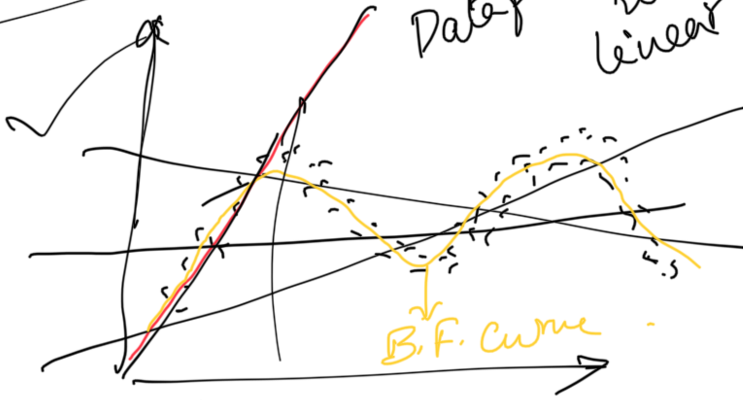
(Classification) (Give credit/NOT)

Classify: Binary. 1 0

Probability



Case II — Relationship between the Datapoints are not linear



but Quadratic

In these cases: ① My LR is incapable & I will need a more Complex Model.

In order fit non-linear data pts we will add more 'Complexity in the model.

$y = mx + c$ { Slope / intercept }

$y = (\uparrow \text{Complexity}) \rightarrow x \in$ independ.

Mapping function to be more complex

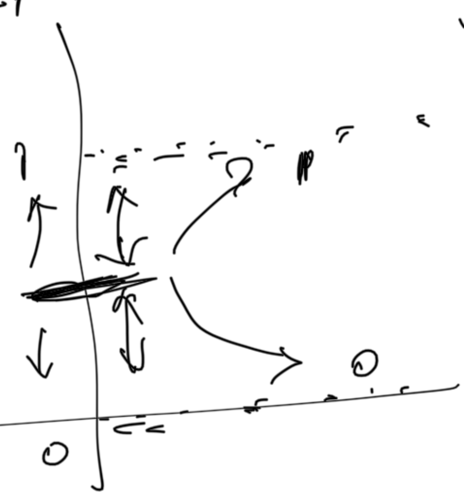
Classification

Not see today

Linear Regression $-\infty$ ∞ $\to$ 0 — 1

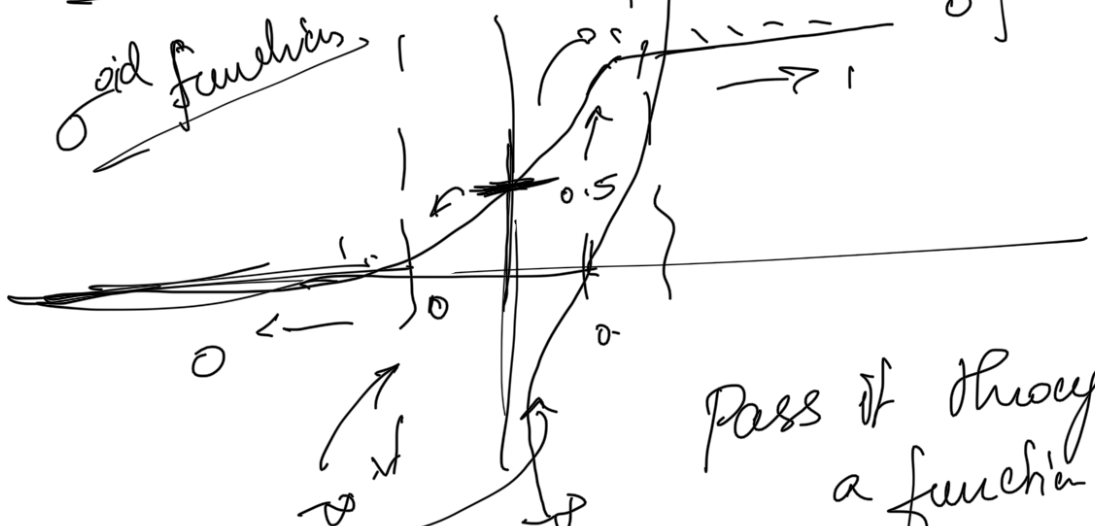
Squish

0 ⟷ 1 (Between)



SIGMOID

σ-oid function



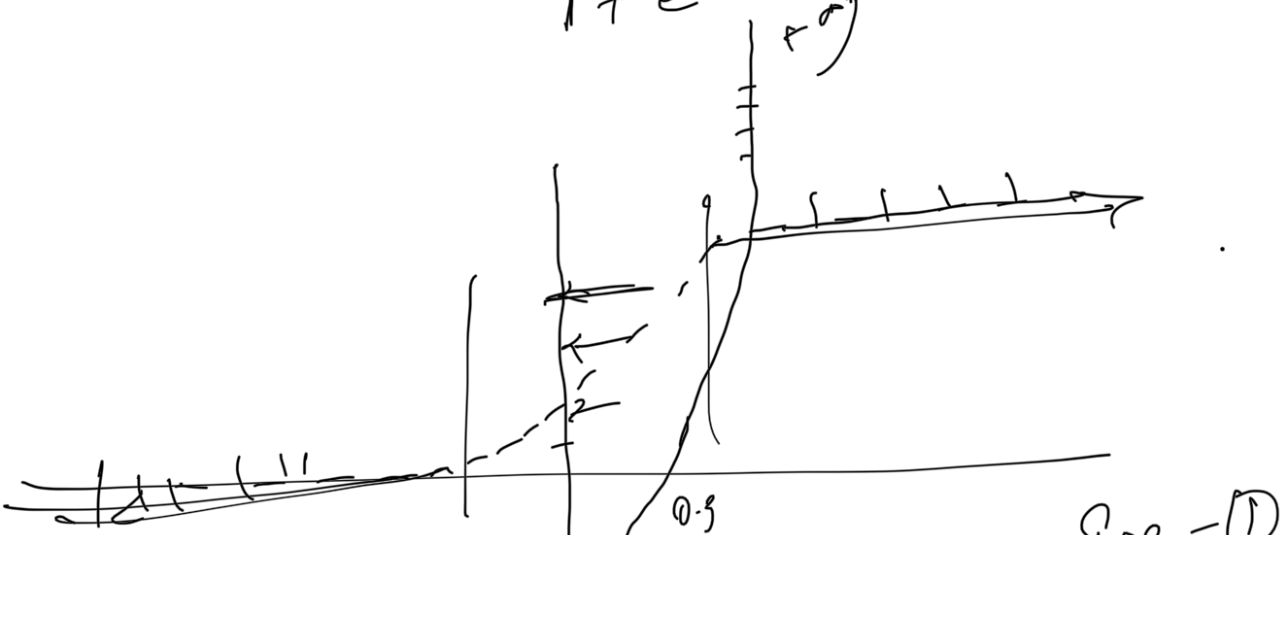
Pass it through a function called "σ Sigmoid fn"

$-\infty$ $+\infty$

0 — 1 probability value.

$$P = \frac{1}{1+e^{-y}}$$ where $y = (mx+c)$

$$\frac{1}{1+e^{-(mx+c)}}$$



Sure

$-\infty$ - $+\infty$ → $\boxed{\sigma}$ → 0 — 1

Compression

Y — datapoint ⟷ probability

P(Success) ⟶ derive probability

Continuous prediction

P(Failed) — 1

Stock price prediction

Y ⇒ Tomorrow stock price.

~~5806.23~~

Continuous values

Given the data should I buy tomorrow or not

Yes | No.

0.8

0.5

0 — 1

Logistic Regression.

0 — 1

Where

(I) Linear Regression would fail

(ii) Linear Regression would fail in places where I probability as o/p & not continuous o/p's

(iii) Sigmoid function which maps any L.R o/p from $-\infty \rightarrow +\infty$ ⟶ 0 & 1 probability

→ loss function ????

MSE | SSE → predict o/p 0.8 ↔ 1 Expect o/p

0.5^2 ⇒ → would make gradient descent think we are in optimal place already

I will need a loss fn. Blows up when prediction is wrong (∞) → 0↔1

Case 1 | actual = 0 | predicted value = 0 | Error be = 0

$\hat{y}$

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

$(1-0)\log(1-0)$

$-\log(1) = 0$

Case 2 | actual = 0 | $\hat{y} = 1$

$+(1-0)\log(1-1)$

$\Rightarrow -1\log(0) = \infty$

$+ (1-y)\log(1-\hat{y})$ ← $y = 1$ $\hat{y} = 0$

0

Part 1 is working fine for $y = 0$ case 0

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

Case 3 : $y(\log(\hat{y}))$ all $y = 1$

$y=1,\ \hat{y}=1 \rightarrow$

$1(\log(1)) = 0$

Case 4:

$\underline{y=1} \quad \hat{y}=0$

$1(\log(0)) \Rightarrow \infty$

$$\Rightarrow -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \cdot \log(p(\hat{y}_i)) + (1-y_i)\log(1-p(\hat{y}_i))\right]$$

probability

actual y	$\hat{y}$	Error
0	0	0
0	1	∞
1	1	0
1	0	∞

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

$(y)\log\hat{y} \Rightarrow 0$ (for $y=1$)

Indication of how close the predic probabil is to corresponding orig. value.

(0 or 1) Binary

↑ higher → ↑ deviation
↓ lower → ↓ deviation

x_1 x_2 x_3 x_4 x_4

0 | 1

0.8
0.4 } decision Boundary

log-loss !!!