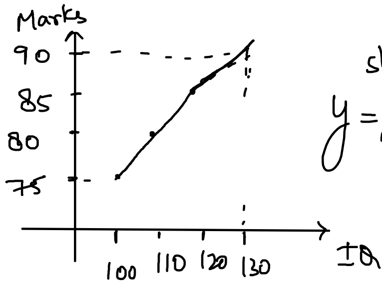
Sunday, May 12, 2024 9:34 AM

Linear Regression



$y = \underbrace{m}_{\text{slope}} x$

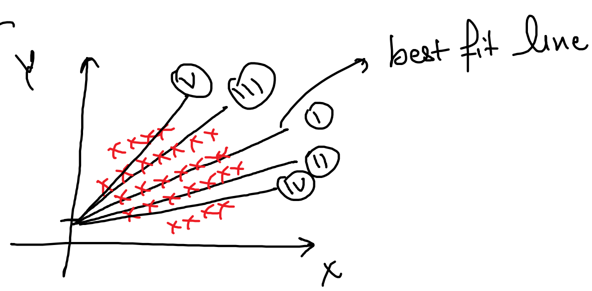
$m = \dfrac{y_2 - y_1}{x_2 - x_1} = \tan\theta = \dfrac{dy}{dx}$

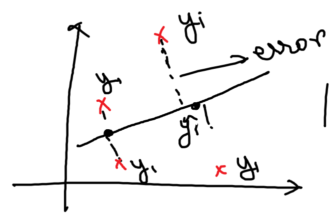
normal ↑ ; $\tan\theta$ ↓ trigonometrical ; $\dfrac{dy}{dx}$ ↳ calculas

$\boxed{\text{Marks} = m \times IQ}$

$\boxed{\text{Marks} = m_1 \times IQ + m_2 \times \text{Hours}}$

Reality





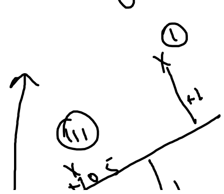
$|\hat{y}_i - y_i|$

Linear Regression

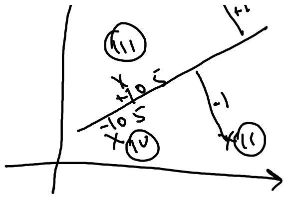
- OLS ↓ Ordinary Least Squares ↓ Direct plot best fit
- Gradient Descent ↓ min error.

OLS

$y = w_1 x_1 + w_2 x_2 + w_3 x_3 + - - - + w_n x_n \rightarrow$ eqⁿ of hyperplane



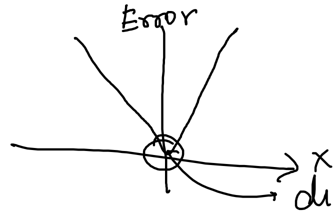
Absolute error $= \Sigma(y_i - \hat{y}_i)$



Absolute error $= \sum |y_i - \hat{y}_i|$

$= |+1| + |-1| + |-0.5| + |+0.5|$

Absolute Error $= 3$



discontinous → can't apply calculus

Squared Error $= \sum (y_i - \hat{y}_i)^2$

$\hat{y_i} = mx + b$

$$E(m, b) = (y_i - \hat{y_i})^2 = 0$$

$$E(m,b) = [y_i - (mx_i + b)]^2 = 0$$

b (intercept)

$$\frac{dE}{db} = \frac{d \sum (y_i - (mx_i + b))^2}{db} = 0 \qquad \boxed{\frac{dx^n}{dx} = nx^{n-1}}$$

$$\Rightarrow 2\sum\left(\cancel{\frac{dy_i}{db}}^{\,0} - \cancel{\frac{d\,mx_i}{dx}}^{\,0} - \frac{db}{db}\right)\underbrace{(y_i - mx_i - b)}$$

$$\Rightarrow -2\sum (y_i - mx_i - b) = 0$$

$$\sum (y_i - mx_i - b) = \frac{0}{-2} = 0$$

$$\sum y_i - \sum mx_i - \sum b = 0$$

Divide both sides by n

$y = b + mx_i$

Divde both sides by n

$$\frac{\sum y_i}{n} - m\frac{\sum x_i}{n} - \frac{\sum b}{n} = \frac{0}{n} = 0$$

$$\bar{y_i} - m\bar{x_i} - \frac{\cancel{n}b}{\cancel{n}} = 0$$

best value of b for best fit line $\leftarrow$ $\boxed{b = \bar{y_i} - m\bar{x_i}}$

$\underline{m}$ (slope)

$$\frac{dE}{dm} = \frac{d\sum(y_i - mx_i - b)^2}{dm} = 0$$

↓ Assignment

value of m for best fit line $\leftarrow$ $\boxed{m = \frac{\sum(y_i - \bar{y_i})(\bar{x_i} - x_i)}{\sum(x_i - \bar{x_i})^2}} = 0$

⇒ best fit line parameters

$\boxed{b = \bar{y_i} - m\bar{x_i}}$ & $m =$

Model

$$x_i \longrightarrow \boxed{mx_i + b} \longrightarrow \hat{y_i}$$

with help of m and b, you can calculate best fit line (eq^n) directly.

Evaluation

Metrics

Fitness based Metrics ← → Error based Metrics

Error based Metrics

(1) Mean Absolute Error $\Rightarrow \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$

Advantages:

→ same unit as that of data

→ less sensitive to outliers

Disadvantages:

→ can't be differentiated

(2) Mean Squared Error $\Rightarrow \frac{1}{n}\sum(y_i - \hat{y}_i)^2$

Advantages

⇒ Can be used as loss f^n

Disadvantages

⇒ very sensitive to outliers

⇒ not intuitive

(3) RMS Error (Root Mean Squared) $\Rightarrow \sqrt{\frac{1}{n}\sum(y_i - \hat{y}_i)^2}$

MAPE (Mean absolute percentage error):

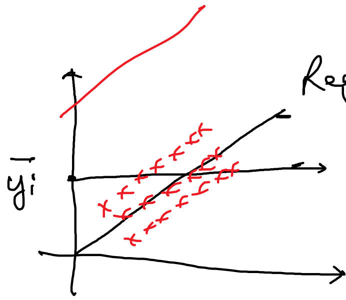
↳ "homework"

Fitness based

Metrics

R^2 ← Metrics → Adjusted R^2

R^2:



Regression line ($\overset{\text{error}}{SSR}$)

Mean line ($\overset{\text{error}}{SST}$) $(y_i - \bar{y}_i) = SST$

$$R^2 = 1 - \frac{SSR}{SST}$$

Case 1: $SSR = 0$, $SST = SST$

$$R^2 = 1 - \frac{0}{SST} = 1 - 0 = 1$$ (overfitting)

Case 2: $SSR = SST$

$$R^2 = 1 - \frac{SST}{SST} = 1 - 1 = 0$$ (underfitting)

Case 3: $SSR > SST$

$$R^2 = 1 - \left(\frac{SSR}{SST}\right) = -ve$$

greater than 1

Problem with R^2 $\uparrow$ $\Rightarrow$ As #columns $\uparrow$, $R^2 \uparrow$

Adjusted R^2 $\Rightarrow$ $\text{Adj } R = 1 - \left[\frac{(1-R^2)\downarrow (n-1)}{(n-p-1)\downarrow} \right]$

$n \rightarrow$ # datapoints

$p \rightarrow$ # no. of columns

↓

Not present in Sklearn

0.720 → 0.002
0.722 =

Adj $R^2 \downarrow \propto p \uparrow$

Multicollinearity

	f_1	f_2	f_3	
w	1	2	3	⇐ Right
w	0	3.5	3	

↳ faulty calculations

$y = \underbrace{1 f_1} + 2 f_2 + 3 f_3$

$f_1 = 1.5 f_2$

$y = 1.5 f_2 + 2 f_2 + 3 f_3$

$y = \underbrace{0 f_1} + 3.5 f_2 + 3 f_3$

↳ faulty model

Detect of Multicollinearity

Correlation Matrix
↓
(corr > 0.7 &
corr < -0.7)

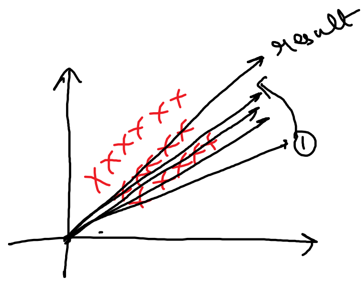
VIF (variance inflation factor)
↳ $\frac{1}{1-R^2}$
↳ $[1, \infty]$

$(corr > 0.7$ & $corr < -0.7)$ ↳ $[1, \infty]$

VIF > 5

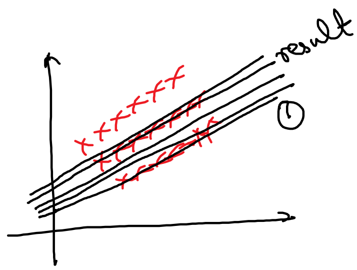
Gradient Descent

(I)



$y = mx + b$ (line rotation can be done by slope)

(II)



$y = mx + b$ (with help of intercept g can move line up & down)

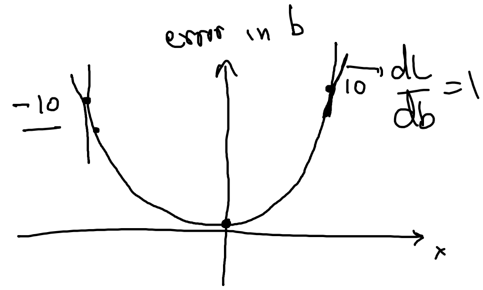
Steps: m = constant, b = variables

1> Choose any random value of b

2> $\frac{dL}{db} = \frac{d(y_i - mx_i - b)^2}{db} = -2\underbrace{(y_i - mx_i - b)}_{\text{slope}}$

3> update: $b_{next} = b_{old} - \text{slope}$

Slope $= \underbrace{-1}_{} = \frac{dL}{db}$ ↳ gradient (∂)



(1)

$b_{next} = -10 - (-1) = -10 + 1 = -9$

$b_{next} = 10 - 1 = 9$

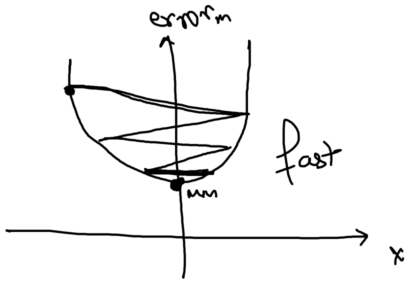
Actual steps:

1) Choose any random value of m & b

2) find $\frac{\partial L}{\partial m}$ & $\frac{\partial L}{\partial b}$

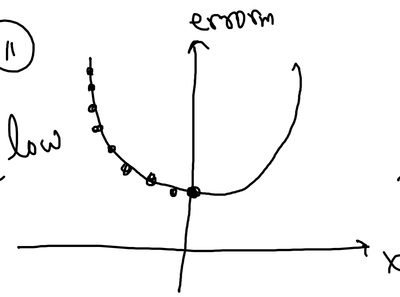
3) $m_{next} = m_{old} - \eta \frac{\partial L}{\partial m}$, $b_{next} = b_{old} - \eta \frac{\partial L}{\partial b}$

(I) η high



fast

(II) η low



slow

$\eta \rightarrow$ learning rate / step size

Hyperparameter $\rightarrow \lambda , \eta$

Log Reg $\Rightarrow$ $\overbrace{\text{Logistic loss}}^{\text{loss } f^n}$ + Regularizer

Lin Reg $\Rightarrow$ $\underbrace{\text{Squared Loss}}_{\text{loss } f^n}$ + Regularizer

GD

Stochastic GD | Batch GD | Mini Batch GD (DL)

Stochastic GD → faster

→ 100 rows, 100 iteration

1 iteration → (1) b, m
(2) b, m
⋮
(100) b, m

2 iteration → (1) b, m
⋮
(100) b, m

Calculation = 100×100 = 10,000

Batch GD → slow → not good for large dataset

→ 100 rows 100 iterations

1 iteration → 100 rows → b, m

2 " → 100 rows → bnext, mnext

⋮

100 " → 100 rows → b, m

calculations = 100