Supervised Learning

- Teaching a computer how to do something by showing it Examples.
- model learns from labeled training data.
- Eg: E-mail spam detection

  Emails would be labeled as spam/ not-spam

Classification

KNN, DT, SVM

NN, Logistic Reg.,

Random Forest, Naive Bias

Assign data points to pre-defined categories

Regression

Linear Reg.,

Polynomial Reg.,

Lasso Reg., Ridge Reg.

Predict numeric values based on input features.

Unsupervised Learning

- unlabeled data is used to train the model. ↓
  input without labels
- Algorithms: PCA
  K-means clustering
  Hierarchical clustering (BIRCH)
  DBSCAN clustering

Regression

- Help in understanding & predicting relationship b/w multiple variables

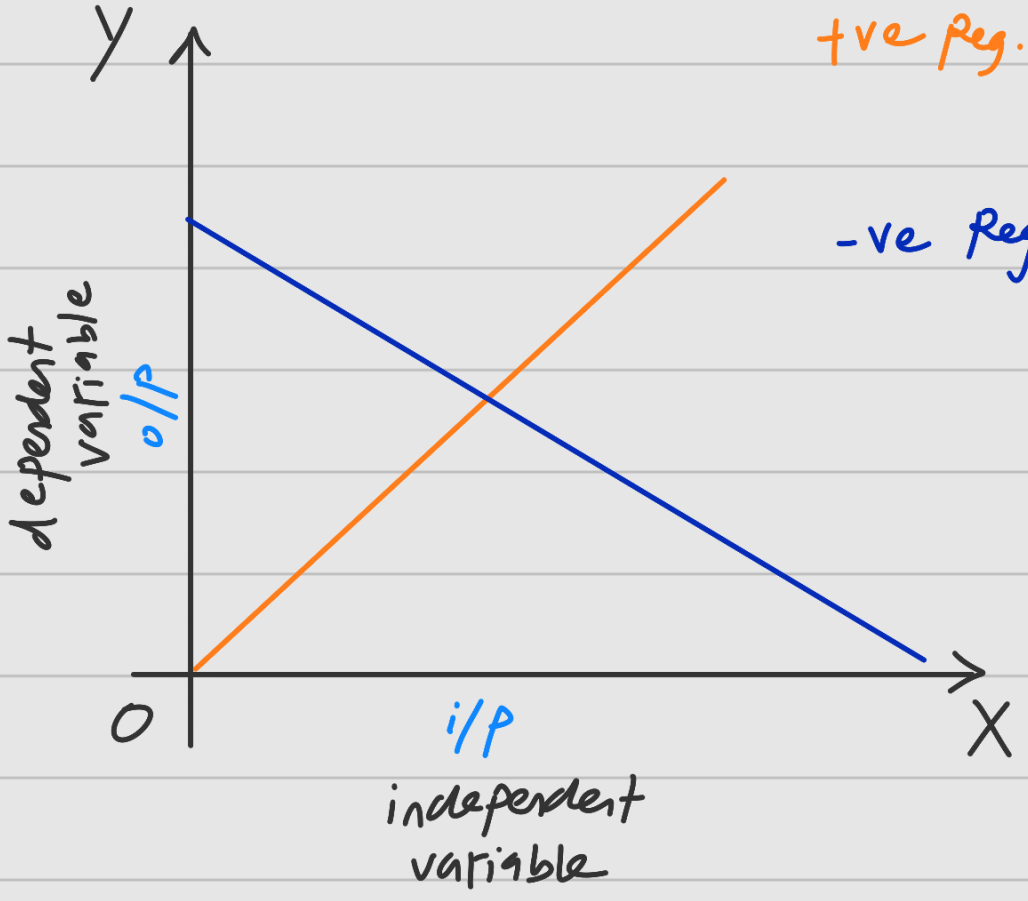
(input) independent
The basis for prediction.

dependent (output)
to be predicted

Eg: Predicting salary based on years of XP.

salary → dependent

years of XP → independent



+ve Reg. X↑ Y↑
X↓ Y↓

-ve Reg X↑ Y↓
X↓ Y↑

Linear Regression

Eqn. of linear Regression:

$$y = mx + b$$

$y \rightarrow$ dependent variable

$x \rightarrow$ independent variable

$m \rightarrow$ slope of line

(change in y for unit change in x)

$b \rightarrow$ intercept

(value of y, when x is 0)

Least Square

Eg: Predicting Pizza prices based on diameter (least square method)

Diameter (X)	Price (Y)
8	10
10	13
12	16

Soln $y = mx + b$

calculating m & b first

X_i	Y_i	$\bar{X}-X_i$	$\bar{Y}-Y_i$	$(\bar{X}-X_i)^2$	$(\bar{X}-X_i)(\bar{Y}-Y_i)$
8	10	2	3	4	6
10	13	0	0	0	0
12	16	-2	-3	4	6
$\Sigma X = 30$	$\Sigma Y = 39$	$\Sigma = 0$	$\Sigma = 0$	$\Sigma = 8$	$\Sigma = 12$
$\bar{X} = 10$	$\bar{Y} = 13$				

$$m = \frac{\Sigma(\bar{X}-X_i)(\bar{Y}-Y_i)}{\Sigma(\bar{X}-X_i)^2} = \frac{+12}{8} = \boxed{\frac{3}{2}}$$

$\bar{y} = m\bar{x} + b$

$b = \bar{y} - m\bar{x}$

$b = 13 - (1.5)(10)$

$b = 13 - 15$

$\boxed{b = -2}$

Best fit line

$\boxed{y = 1.5x - 2}$

Error / loss → MSE

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - y_p)^2$$

y_i ↓ Actual

y_p ↘ predicted

Gradient Descent

works for any differential model

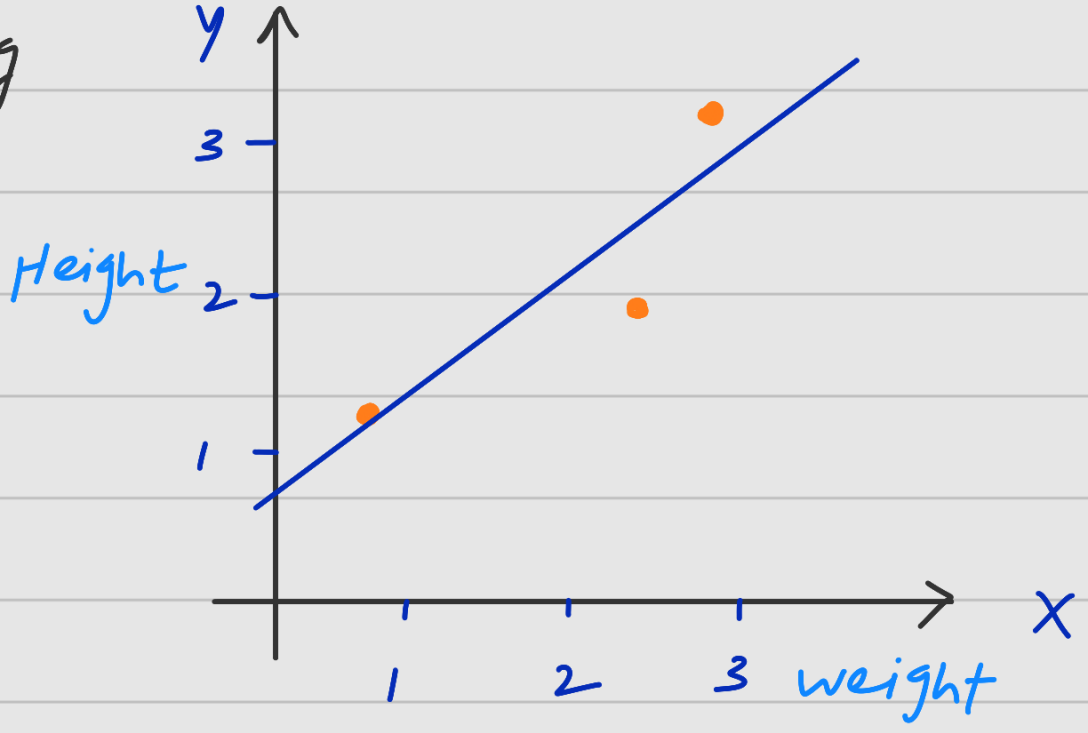
Suitable for large datasets

"used for optimization"

loss f^ → Sum of squared Residuals

Σ (observed − predicted)^2

Eg



Predicted Height = (slope x weight) + intercept

$[y = mx + b]$

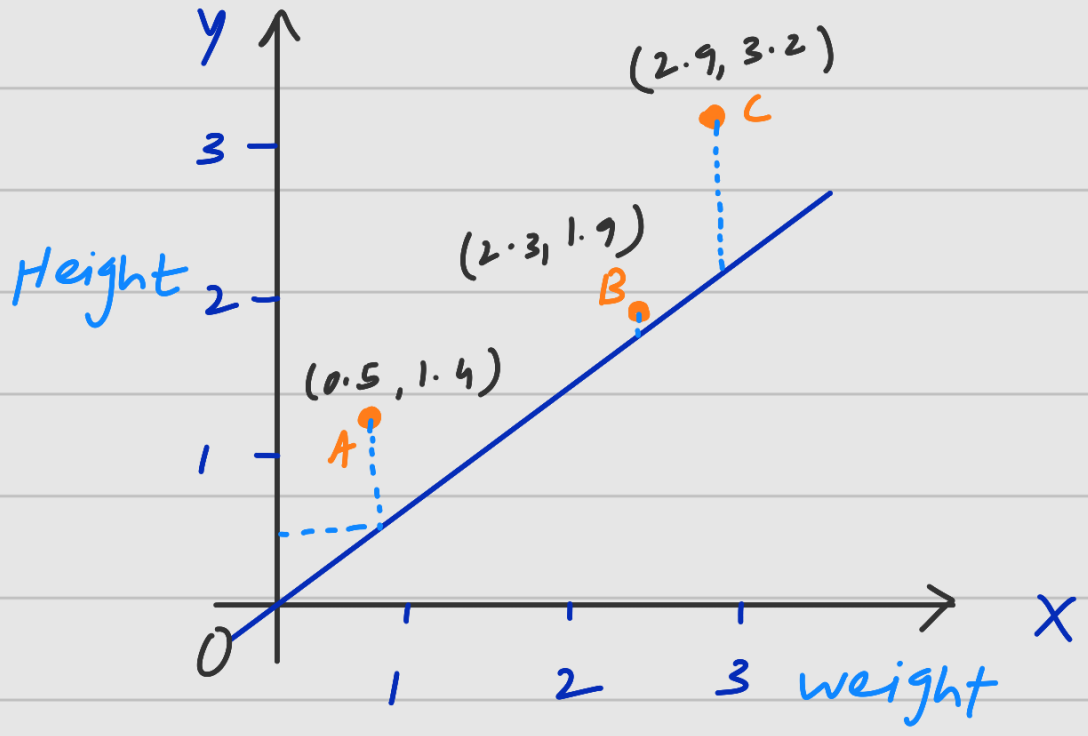
let's add least square estimate of

slope = 0.64

Predicted Height = 0.64 x weight + intercept

- we'll use gradient descent to find optimal value for intercept
- Firstly we pick random value for intercept.
- This is just initial guess that gives Gradient Descent to improve upon.

let, intercept = 0



Point A → Actual height = 1.4

Actual weight = 0.5

pred. Height = 0.64 x weight + 0

= 0.64 x 0.5 = 0.32

Residual(A) = Actual Height − Predicted Height

= 1.4 − 0.32 = 1.1

Point B → Actual height = 1.9

Actual weight = 2.3

Pred. Height = 0.64 × weight + 0

= 0.64 × 2.3 = 1.472

Residual(B) = Actual − Pred. Height

= 1.9 − 1.472 ≈ 0.4

Point C → Actual Height = 3.2

Actual weight = 2.9

Pred. Height = 0.64 × weight + 0

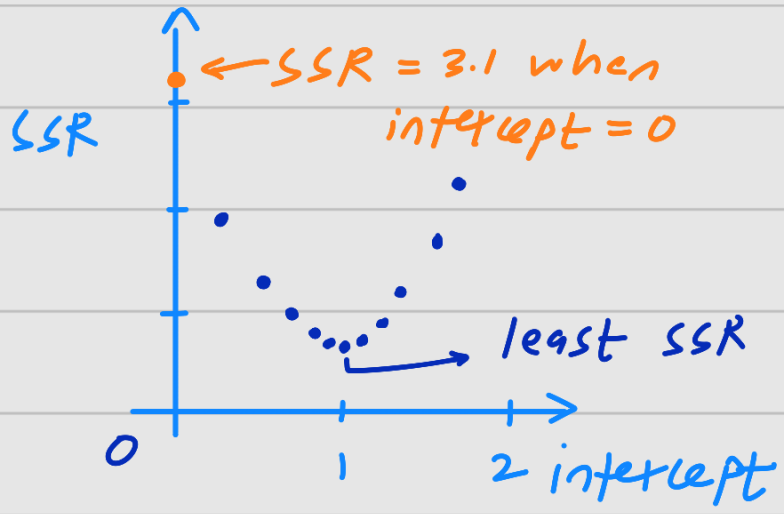
= 0.64 × 2.9 = 1.856

Residual(C) = Actual − Pred. Height

= 3.2 − 1.856 ≈ 1.3

Sum of Squared Residuals = $1.1^2 + 0.4^2 + 1.3^2$

≅ 3.1



For increasing values of intercept we get diff points

Notice Big steps when its far away

Sum of Squared Residuals $= \sum (\text{Actual} - \text{Predicted Height})^2$

$$= (1.4 - (\text{intercept} + 0.64 \times 0.5))^2 + (1.9 - (\text{intercept} + 0.64 \times 2.3))^2 + (3.2 - (\text{intercept} + 0.64 \times 2.9))^2$$

Now we can put any value for intercept & get sum of squared Residuals

We can take derivative of this fn & take find slope at any value of intercept.

$$\frac{d(SSR)}{d\,intercept} = \frac{d[(1.4-(intercept+0.64\times0.5))^2]}{d\,intercept} + \frac{d[(1.9-(intercept+0.64\times0.23))^2]}{d\,intercept} + \frac{d[(3.2-(intercept+0.64\times2.9))^2]}{d\,intercept}$$

$$\frac{d[(1.4-(intercept+0.64\times0.5))^2]}{d\,intercept}$$

$$2(1.4-(intercept+0.64\times0.5))(-1)$$

$$\frac{d(1.4-intercept-K)}{d\,intercept}$$

$$0-1-0$$

$$(-1)$$

$$-2(1.4-(intercept+0.64\times0.5))$$

$$-2(1.4-(intercept+0.64\times2.3))$$

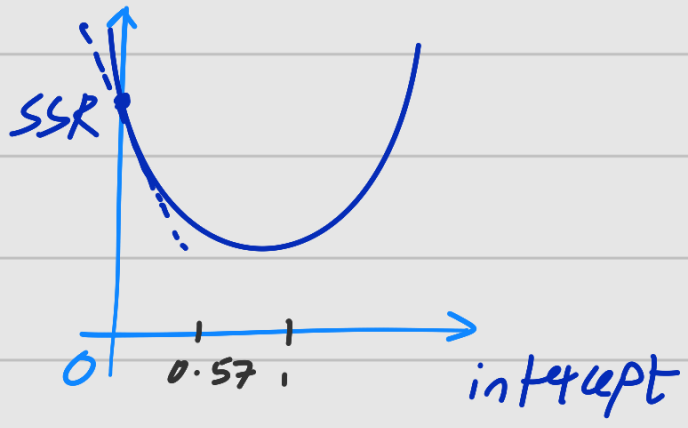
$$-2(3.2-(intercept+0.64\times2.9))$$

Grad. descent will use it to find where the SSR is lowest.

As intercept = 0, push it in derivative

$$\left.\begin{aligned} &-2(1.4-(0+0.64\times 0.5)) \\ &-2(1.4-(0+0.64\times 2.3)) \\ &-2(3.2-(0+0.64\times 2.9)) \end{aligned}\right\} = -5.7$$

for intercept = 0 ; slope = -5.7



When slope is far from zero take big steps.
When slope is near to zero take small steps

[Step size = Slope × learning Rate]

When intercept = 0

Slope = -5.7

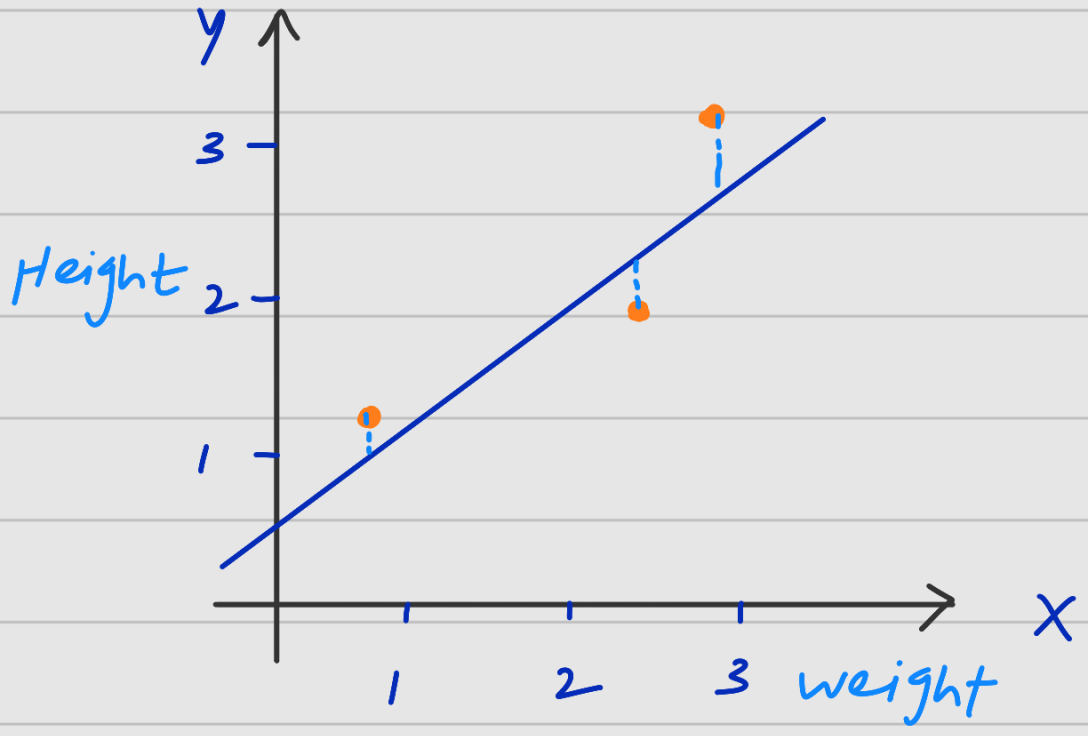
Step size = -5.7 × 0.1

= -0.57

[New intercept = old intercept - step size]

" = 0 - (-0.57)

New intercept = 0.57



At intercept = 0.57, the residuals shrinked

Push new intercept = 0.57 in derivative

$$\left.\begin{aligned} &-2(1.4-(0.57+0.64\times 0.5)) \\ &-2(1.4-(0.57+0.64\times 2.3)) \\ &-2(3.2-(0.57+0.64\times 2.9)) \end{aligned}\right\} = -2.3$$

Slope = -2.3 at intercept = 0.57

step size = learning rate × slope

= 0.1 × (-2.3)

= -0.23

new intercept = old intercept − step size

= 0.57 − (−0.23)

new intercept = 0.80

Residuals shrinked even more at intercept = 0.80

Push intercept = 0.8 in derivative

$$\left.\begin{aligned} &-2(1.4-(0.8+0.64\times 0.5)) \\ &-2(1.4-(0.8+0.64\times 2.3)) \\ &-2(3.2-(0.8+0.64\times 2.9)) \end{aligned}\right\} = -0.9$$

Slope = −0.9 ; at intercept = 0.8

Step size = Slope × L.rate

= −0.9 × 0.1 = −0.09

new intercept = old intercept − step size

= 0.8 − (−0.09)

= 0.8 + 0.09

new intercept = 0.89

⋮ going on

new intercept = 0.92
new intercept = 0.94
new intercept = 0.95

} After 6 steps Gradient descent estimate for intercept = 0.95

The least square estimate for intercept is also 0.95

[Gradient descent will stop when step size is very close to zero. i.e when slope is very close to zero]

In practice min step size = 0.001

Optimization for 2 or more variables

- We take derivative of SSR wrt intercept as earlier (slope as K)
- Then we take derivative of SSR wrt slope (considering intercept as K)

$$\frac{d(SSR)}{d(Slope)} = -2(0.5)(1.4-(intercept + Slope \times 0.5)) - 2(2.3)(1.9-(intercept + Slope \times 2.3)) - 2(2.9)(3.2-(intercept + Slope \times 2.9))$$

↓ weight of a point

$$\frac{d}{d\ intercept}\ \text{Sum of squared residuals} = -2(\mathbf{1.4} - (intercept + slope \times \mathbf{0.5})) + -2(\mathbf{1.9} - (intercept + slope \times \mathbf{2.3})) + -2(\mathbf{3.2} - (intercept + slope \times \mathbf{2.9}))$$

$$\frac{d}{d\ slope}\ \text{Sum of squared residuals} = -2 \times \mathbf{0.5}(\mathbf{1.4} - (intercept + slope \times \mathbf{0.5})) + -2 \times \mathbf{2.9}(\mathbf{3.2} - (intercept + slope \times \mathbf{2.9})) + -2 \times \mathbf{2.3}(\mathbf{1.9} - (intercept + slope \times \mathbf{2.3}))$$

NOTE: when we have 2 or more derivative of a same f"(SSR), they are called gradient

Again we'll pick intercept = 0 & slope = 1
& push it in derivative

$$\frac{d}{d\,\text{intercept}} \text{Sum of squared residuals} = -2(1.4 - (0 + 1 \times 0.5)) + -2(1.9 - (0 + 1 \times 2.3)) + -2(3.2 - (0 + 1 \times 2.9)) = -1.6$$

Slope

$$\frac{d}{d\,\text{slope}} \text{Sum of squared residuals} = -2 \times 0.5(1.4 - (0 + 1 \times 0.5)) + -2 \times 2.9(3.2 - (0 + 1 \times 2.9)) + -2 \times 2.3(1.9 - (0 + 1 \times 2.3)) = -0.8$$

Slope

LR = 0.01

step size (intercept) = LR × Slope
= 0.01 × (-1.6) = -0.016

step size (Slope) = LR × Slope
= 0.01 × (-0.8) = -0.008

new intercept = old - step size

$= 0 - (-0.016) = 0.016$

new slope = old - step size

$= 1 - (-0.008) = 1.008$

Repeat this until max no. of steps reached OR step sizes becomes 0.001

Here the best-fit line, with intercept = 0.95 slope = 0.64, the same values we get from least square method

Tip: Standardize the data before implementing Gradient descent otherwise it can cause huge/ unstable steps during training. This makes GD Explode (diverge) instead of converging

Converge: GD keeps improving model step by step until it reaches minimum loss (error)

Diverge: GD goes in wrong direction / takes too long steps, making error / loss worse