3.GradientDescent

Tuesday, March 2, 2021 11:42 PM

→ Gradient Descent

It is a general optimization technique

L2 penalty

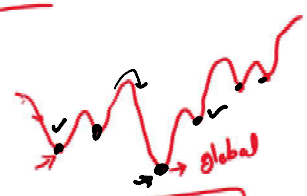
MSE → optimization

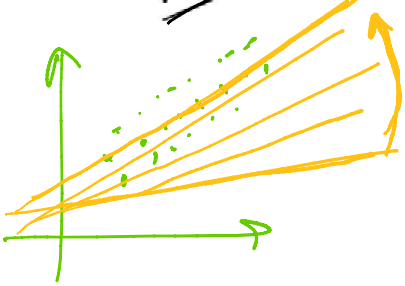
→ RMSE, MAE, R2 → Evaluation

$J(\theta) = MSE(\theta) = \text{Squared}$

→ Linear Regression ✓ → Convex optimization problem

non-Convex functions




$$\boxed{f'(x) = \frac{d\,f(x)}{dx} = 0}$$

Closed form Notation $\boxed{\hat{\theta} = (X^T \cdot X)^{-1} X^T \cdot y}$

Linear Regression → Closed form Notation

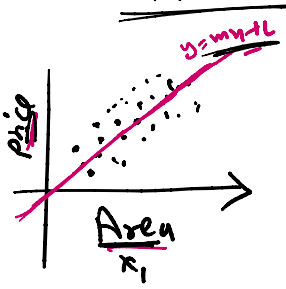
→ Gradient Descent Approach

Hypothesis function

$x_1$ → feature

$y$ → target

one variable regression




$$H\theta(x) = \theta_0 + \theta_1 x_1$$

$$\boxed{H(x) = b + wx}$$

Parameters

$\theta_0, \theta_1$ → Learning parameter

$f(x) = (x + x^2 + 2)^2$

$\boxed{f'(x) = 2 \cdot (x + x^2 + 2) \cdot (1 + 2x)}$

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

$\theta_0$ → bias / Intercept (c)

$\theta_1$ → coefficient (m) / weight

$MSE = \frac{1}{m} \sum_{i=1}^{} \boxed{(y_i - \hat{y}_i)}$

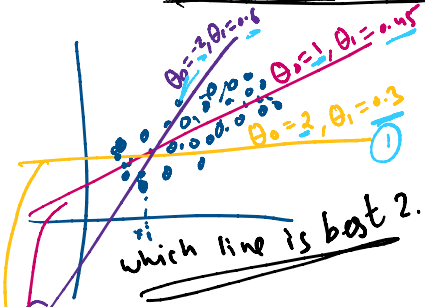
$\theta_1 \rightarrow$ coefficient

(m) / weg

# Cost function / loss function

$\boxed{\hat{y}_i = \hat{\theta}_0 + \hat{\theta}_1 x_i}$

Risidual $= y_i - \hat{y}_i$
(Error)




$$\boxed{J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (y_i - \hat{y})^2}$$

$$RSS = \sum_{i=1}^{m} (y_i - \hat{y})^2$$

m observations

$$\boxed{MSE(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (y_i - \hat{y})^2}$$

MSE = 400

MSE = 250

$\boxed{MSE = 150}$ → minimum

$\rightarrow \boxed{H\theta(X), \; J(\theta)}$

$H(\theta) = \theta^T X$

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

convex

Gradient → derivative $\nabla$

Descent → down hill

→ Gradient Descent

$\boxed{f'(x) = 0}$ — minimum

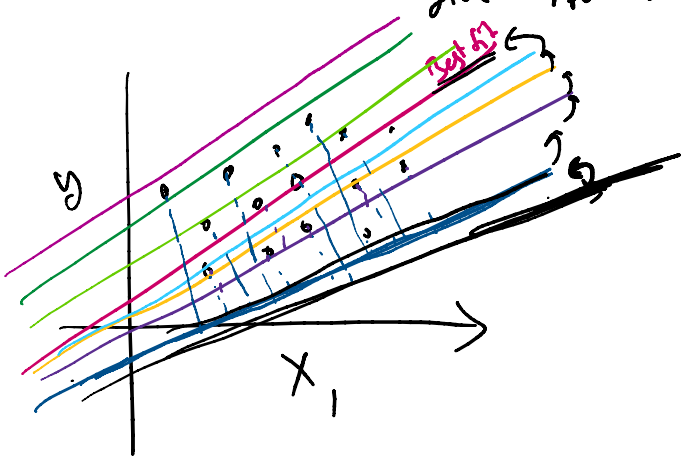
goal is minimise the cost function $J(\theta)$ or MSE

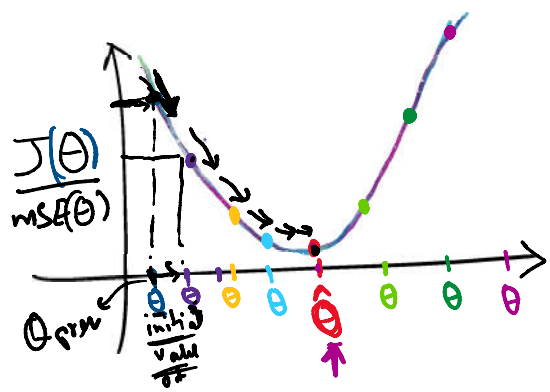
give hypothesis $h\theta(x)$

descent

$\rightarrow \boxed{\theta_{next} = \theta_{prev} - \nabla_\theta MSE(\theta)}$

gradient

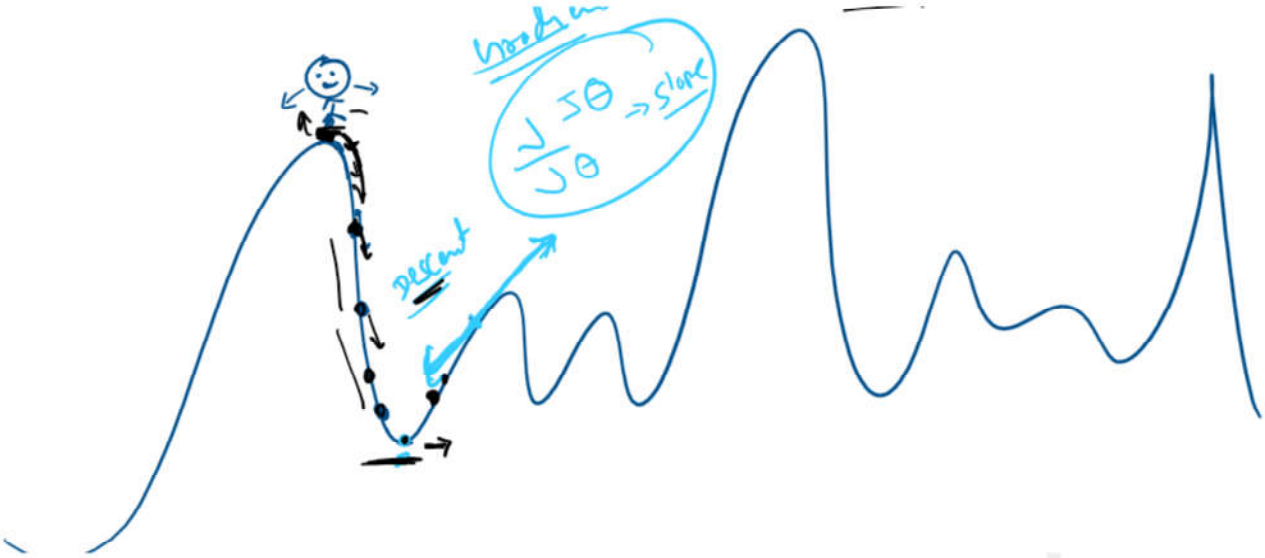


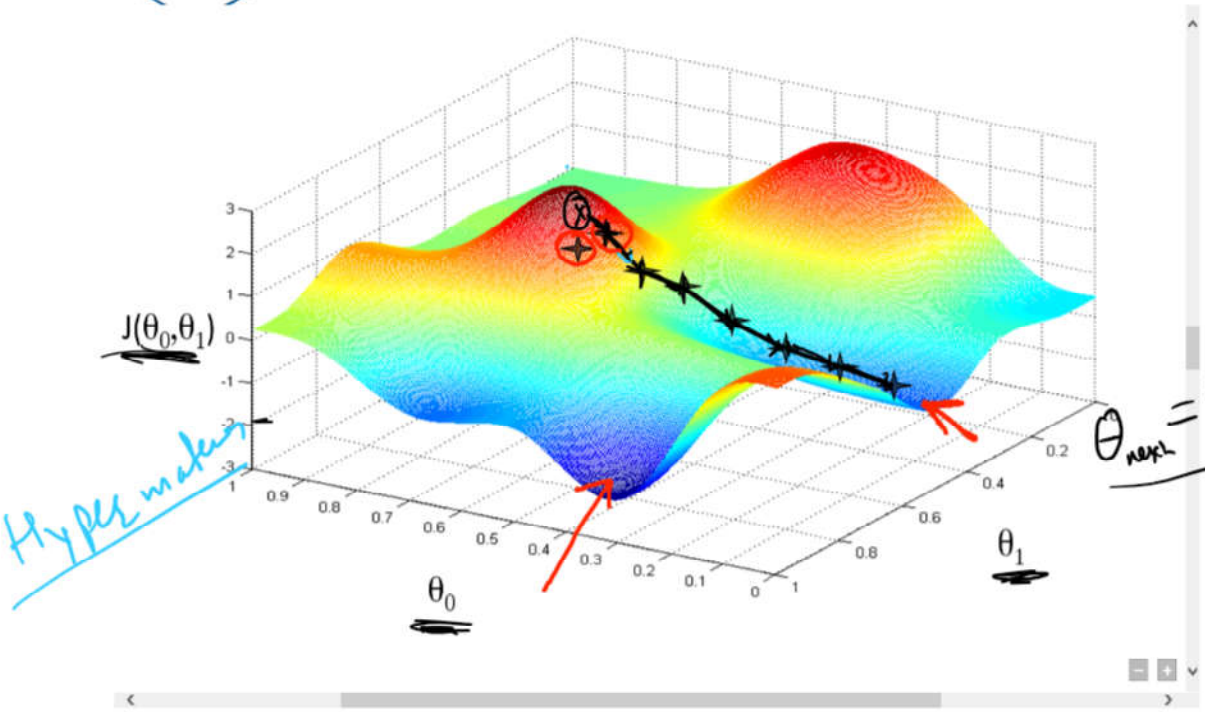




Random initialization

gradient $\nabla\theta$

slope




$$\nabla_0 = \frac{\partial\, MSE(\theta)}{\partial \theta_i}$$

$$\theta_{next} = \theta_{prev} - \nabla_0 MSE(\theta)$$

GD → sensitive to feature scale

$-3 \quad +3$

## Steps of Gradient Algorithm

1. Initlise intial value of $\theta_i$ parameter randomly
2. take partial Derivative of $J(\theta)$ to calculate gradient $\nabla_\theta$

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

$\sim \theta_1$

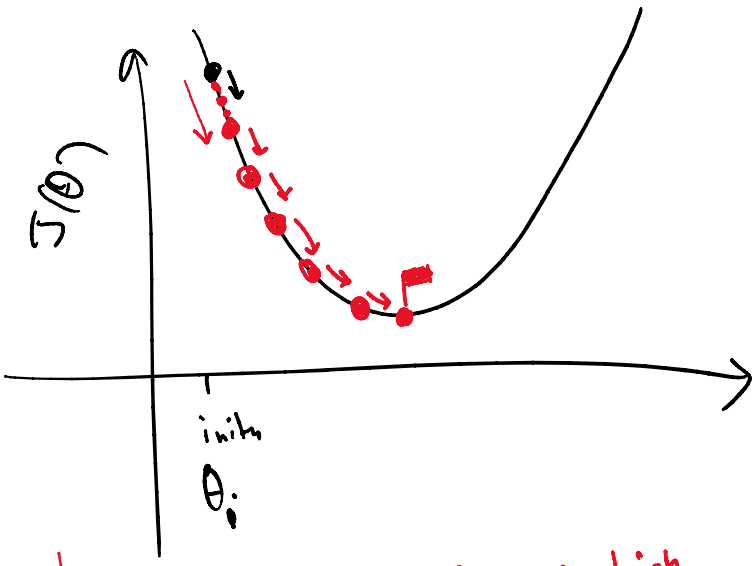
gradient $\Rightarrow$

$$\boxed{\begin{aligned} \nabla_{\theta_0} MSE &= \frac{1}{m} \sum_{i=1}^{m} (\hat{y} - y_i) \\ \nabla_{\theta_1} MSE &= \frac{1}{m} \sum_{i=1}^{m} (\hat{y}_i - y_i)\, x_i \end{aligned}}$$

(3) We want to go down hill means we want to Descent the value of $\theta$ in next such that it takes us to minimum of $J(\theta)$

update state $\rightarrow$

$$\boxed{\theta_{next} = \theta_{prev} - \nabla_\theta J(\theta)}$$

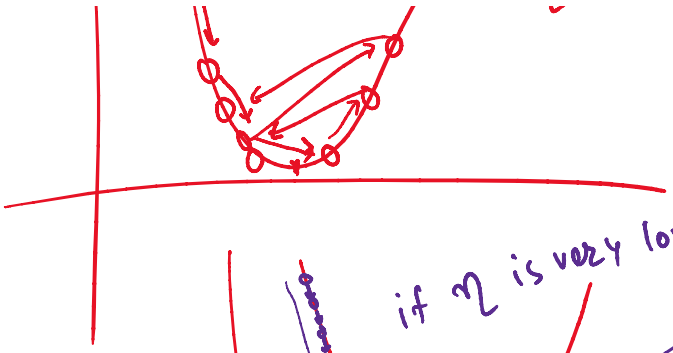


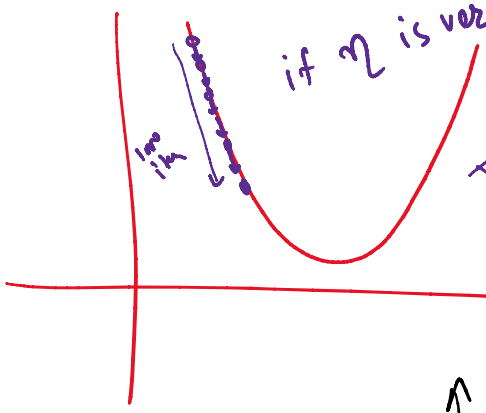

# to speed up the steps of down hill we multiply gradient with learning rate known as eta ($\eta$)

$$\boxed{\theta_{next} = \theta_{prev} - \eta \nabla_\theta MSE}$$

Hyperparameter learning rate

if $\eta$ is high






if $\eta$ is very low

So use gridsearchCV or cross validation to select best value of $\eta$

Hyperparameter learning rate

$\eta = 0.01$

n_iteration $= 100$

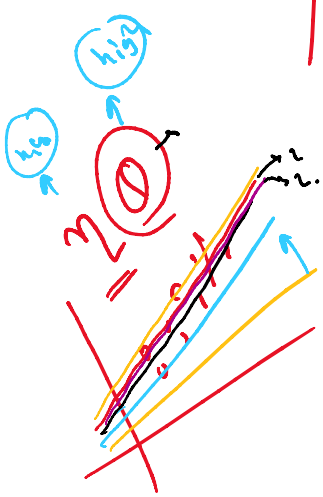
tolerance $= 0.001$

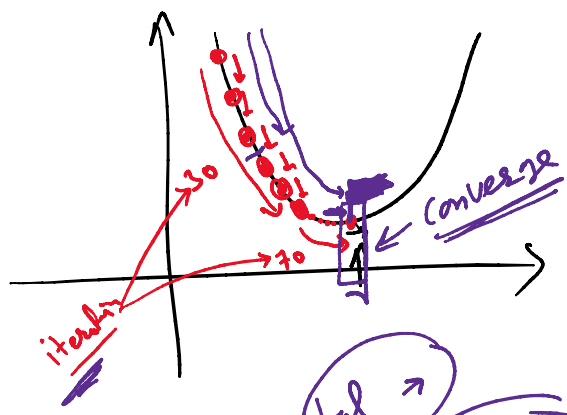
Early Stopping

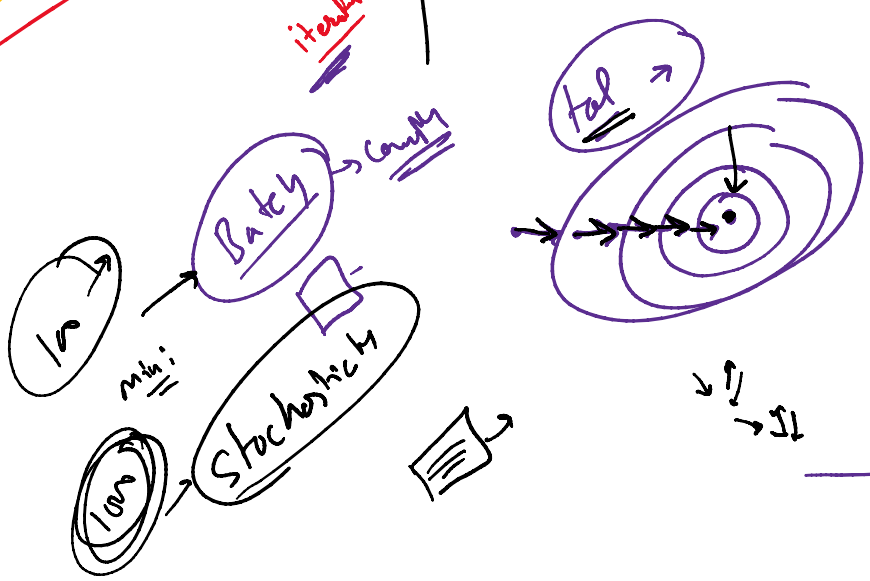
$$tol = \theta_{next} - \theta_{prev}$$

tol is very low mean we are at minimum

$$tol = |Error_{prev} - Error_{curr}|$$



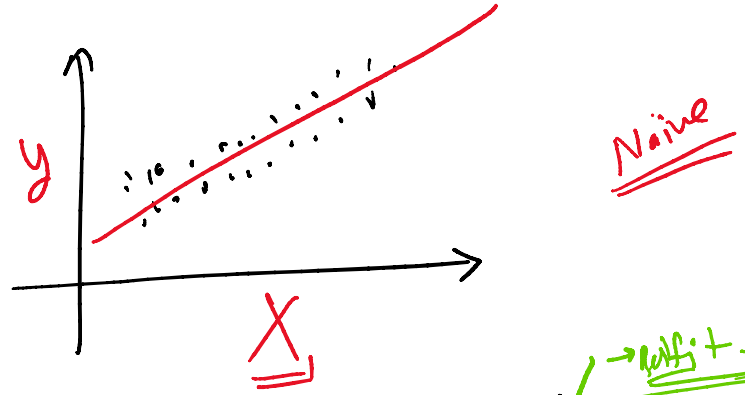



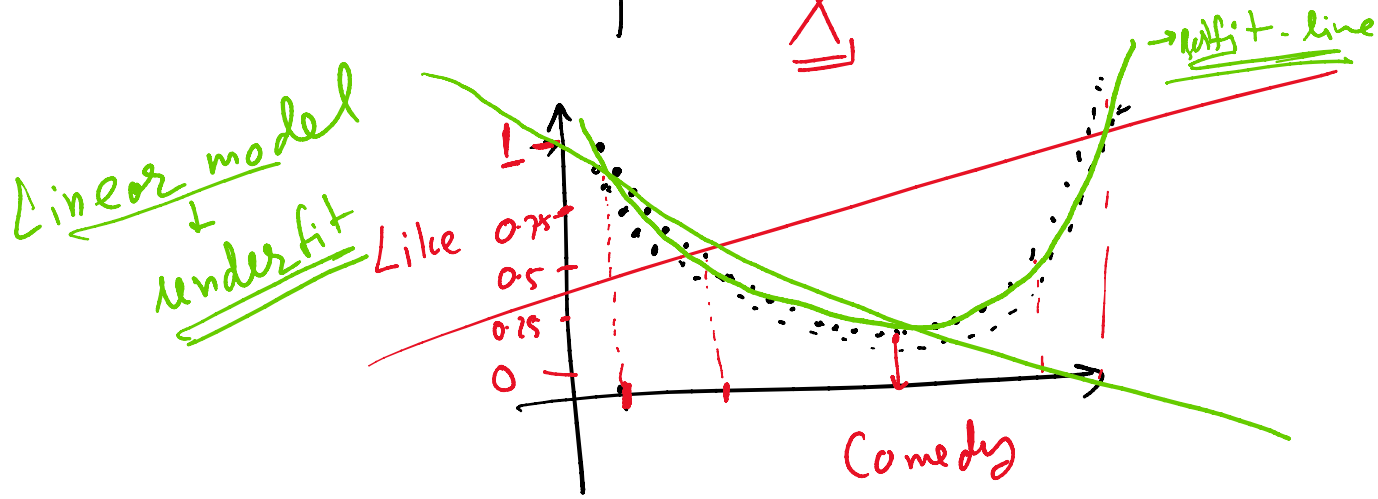
$\eta \to 2.$

eta0 →

n_iter →

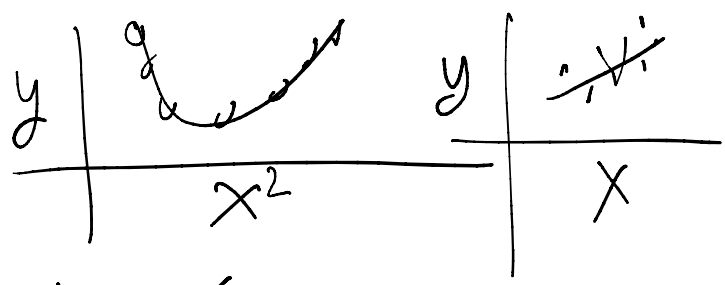
# Liner Regression




$X, y \rightarrow$ linear

$X_1 \rightarrow$



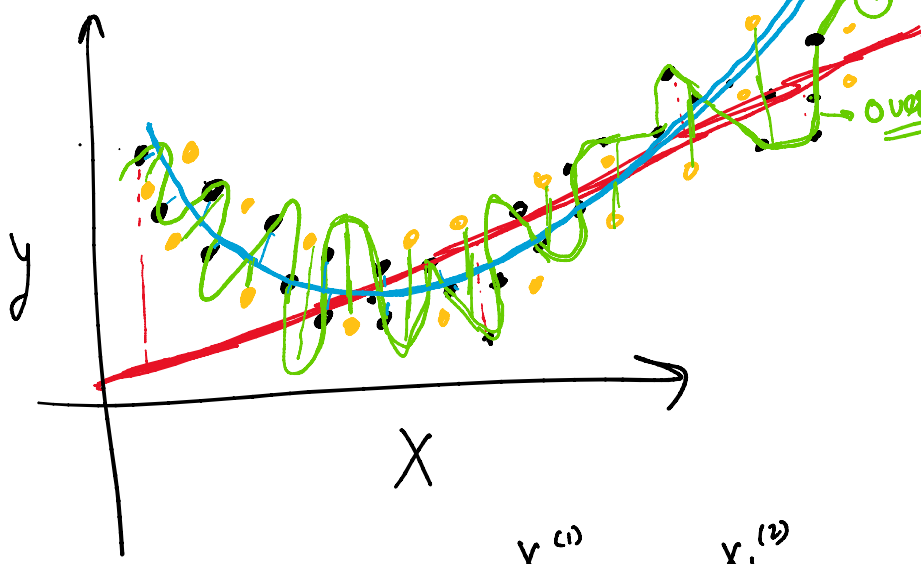

$2^\circ$ degree $= a X_1^2 + b X_1 + c$

$2^\circ$ degree $= a x_1^2 + b x_1 x_2 + c x_2^2 + d$

$X \rightarrow$ $2^\circ$ $\boxed{a X \mid b X^2}$ $\boxed{c}$

$3^\circ$ $\boxed{a x \mid b x^2 \mid c x^3 \mid d}$

$x \xrightarrow{2^0} \boxed{ax} \; \boxed{bx^2} \; \boxed{cx^3} \; \boxed{d}$




2nd fit / Best

① $d = 1$, Linear

under fitting

② $d = 2$, poly

overfit

③ $d = 100$, poly

$X_1 \rightarrow 2^0 \rightarrow$

| $X_1^{(1)}$ | $X_1^{(2)}$ |
|---|---|
| $X_1$ | $X_1^2$ |

$a, b = 1$

$ax^2 + bx + c$

$a = 1, b = 1, c = 0$

$x^2 + x$

$\boxed{X_1, X_2} \rightarrow 2^0 \rightarrow$

| ① | ② | ③ | ④ | ⑤ | ⑥ |
|---|---|---|---|---|---|
| $x_1^2$ | $x_2^2$ | $x_1^2 x_2$ | $x_2^2 x_1$ | $x_1 x_2$ | bias |
| | | | | | |
| | | | | | |
| | | | | | |

$ax_1^2 + bx_2^2 + x_1 x_2$

$x_1^2 x_2 + x_2^2 x_1$

degree $= d \rightarrow 2$

features $= n \rightarrow 2$

$$n_{new} = \frac{(n+d)!}{d!\, n!}$$

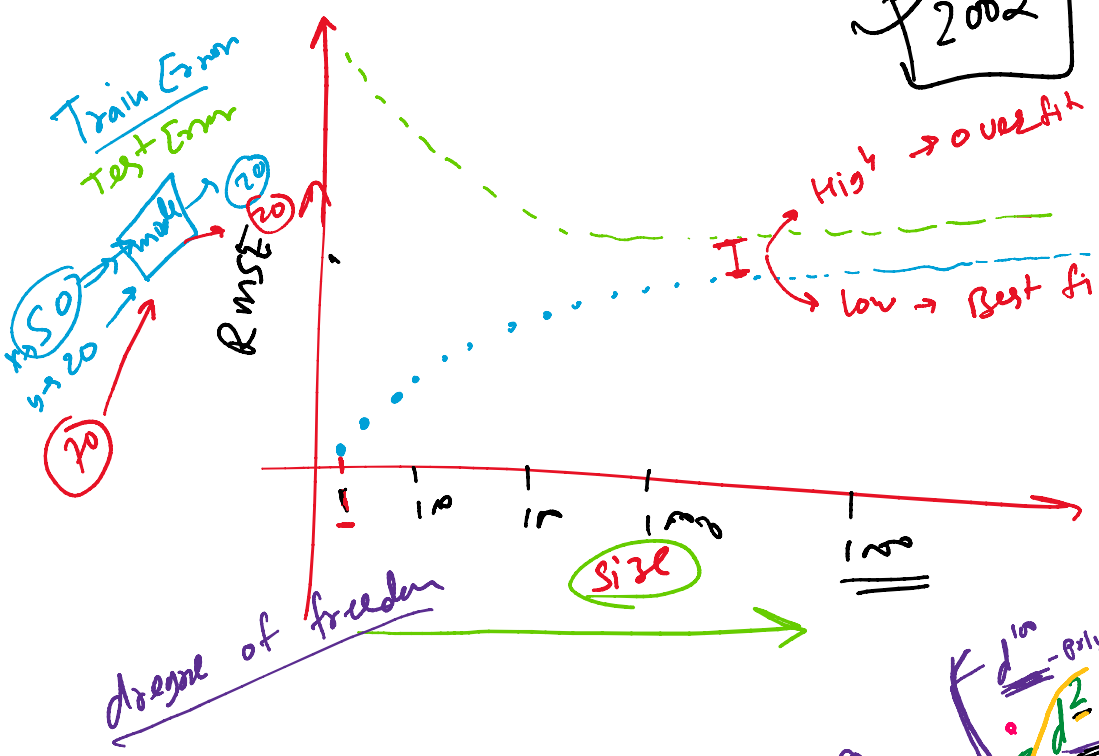
$$n_{new} = \frac{(4)!}{2!\,2!} = \frac{4\times3\times2}{2\times2} \Rightarrow 6$$

$d = 9$

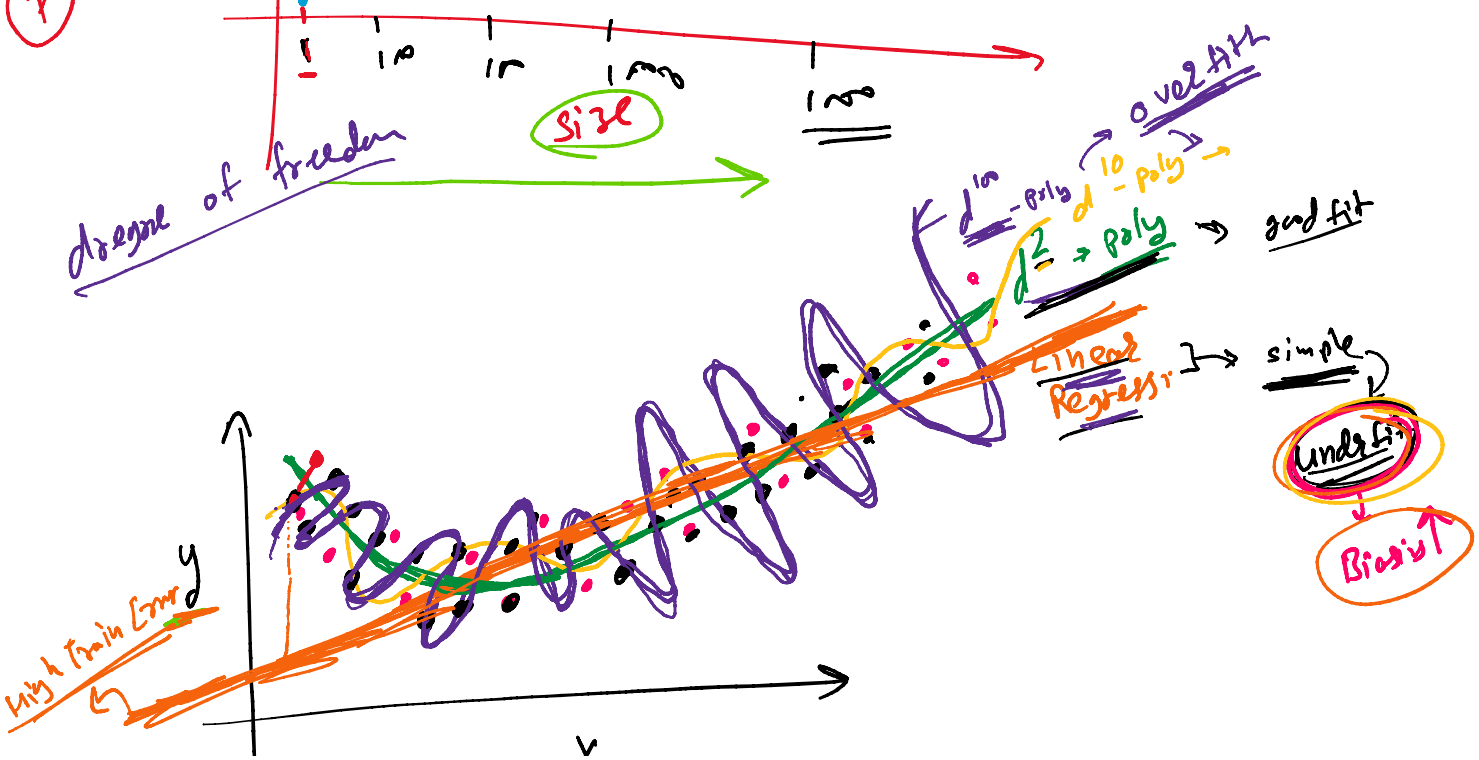
$d = 5$

$$n_{new} = \frac{14!}{9!\,5!} \Rightarrow \frac{14\times13\times12\times11\times10\times9!}{9!\times5\times4\times3\times2\times1} \Rightarrow \boxed{14\times13\times11}$$

$$\boxed{2002}$$




Unreasonable effectiveness of data



# Bias and Variance Trade Off

Hypothesis $H(\theta)$

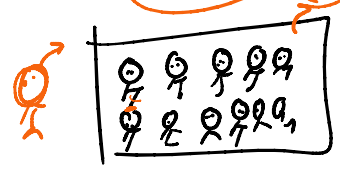
underfitting

**Bias** → Error in model due wrong Assumptions

(like you are fitting a line to a polynomial function.

→ Training Error ≈ Bias

Avg = 100



over fit

**Variance** → (Test Error)

$(\bar{x} - x_i) \rightarrow$ High

model's excessive sensitivity towards small variation in training set. A model with higher degree of freedom tends overfit training point, thus it will have high variance

simple ↓ more complex

| | | |
|---|---|---|
| Bias ↓ | Variance ↑ | → over fit |
| Bias ↑ | Variance ↓ | → underfit |

... and a balance b/w Bias & Variance

You need a balance b/w Bias & Variance

So that you can find a good fit model.

→ How to find a good fit model

Bias ↓ Variance ↑

Regulator

?

polynomial Regression

Regularization

(i) if model is over fitting, what steps you will take to reduce overfitting

we will to reduce no of degree

we will go towards simpler model

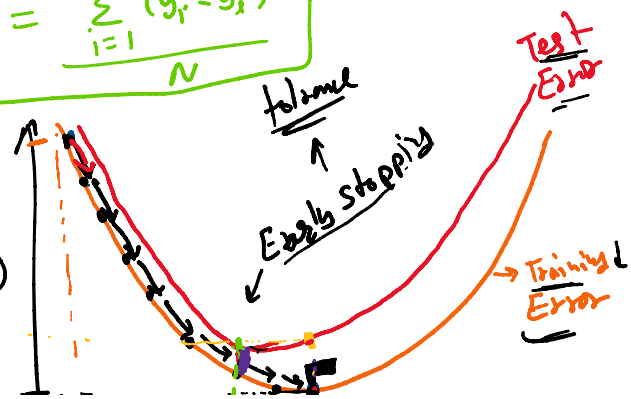
(ii) if model is underfitting → Bias ↑ var ↓

we will increase degree of freedom

# Ridge Regression

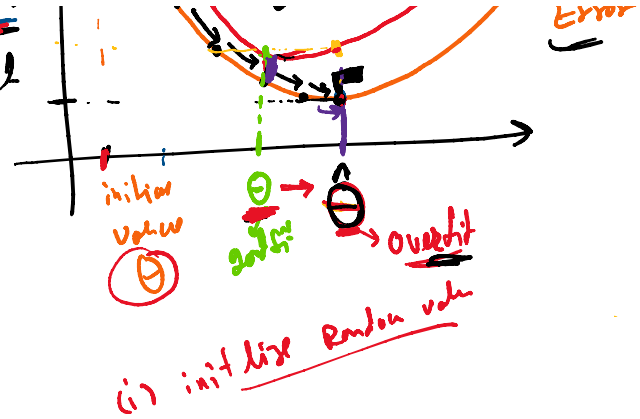
$$MSE\theta = \frac{\sum_{i=1}^{m} (\hat{y}_i - y_i)}{N}$$

$$J(\theta) = MSE(\theta) + \alpha \theta^2$$



$\frac{\partial J(\theta)}{\partial \theta}$

we minimize cost function using gradient descent to find best parameter value




if $\alpha = 0$

Then → $J\theta = MSE(\theta)$

it will same as LR, model can overfitt

if $\alpha$ is very high

your weight will end up close to zero, it will increase model bias then you will end up underfitting your model

Play with alpha $\alpha$ to find best fit

$$\boxed{\hat{\theta} = (X^T X)^{-1} X^T y}$$ → Normal eq$^n$

Ridge

$$\boxed{\hat{\theta} = (X^T X + \alpha A)^{-1} X^T y}$$ → SLR

$\frac{1}{\alpha \uparrow} \to J$

$A \to$ identity matrix

$A \to$ identity

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$