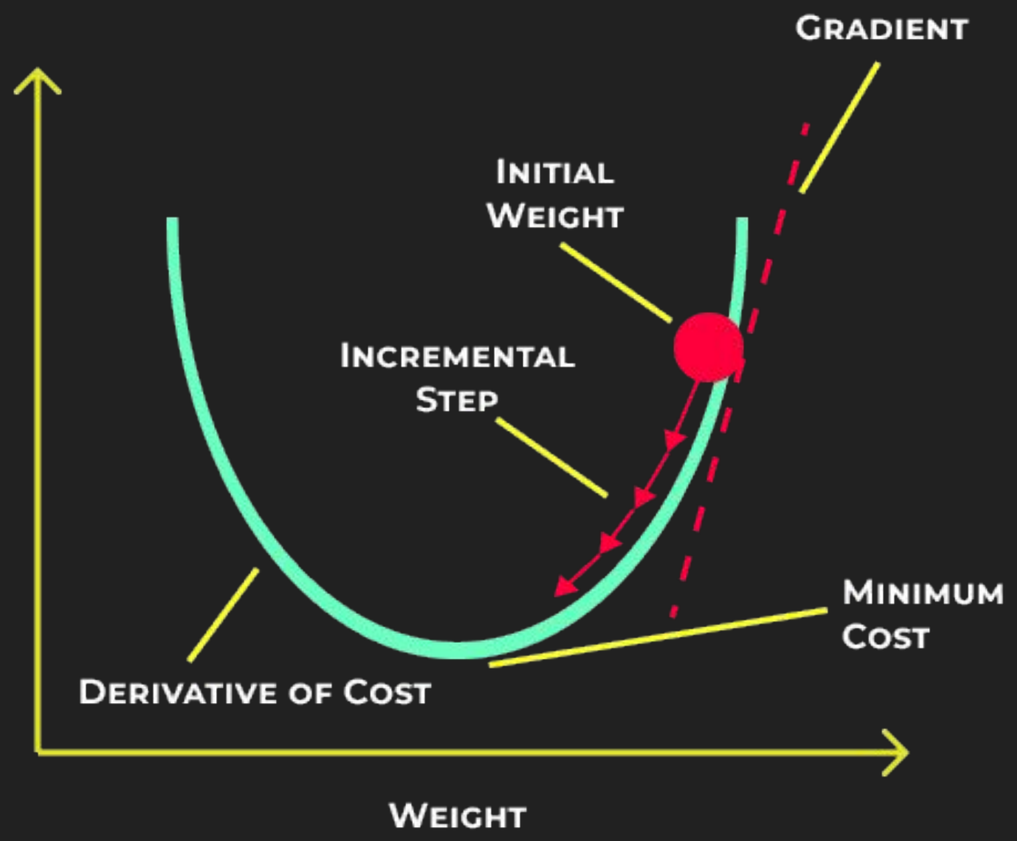
Gradient Descent

- Betty had a dream of perfect pizza but she did not have a recipe.

- She decided to find the best combination of ingredients and adjust their quantities to make the best pizza.

- She tried random combinations and after each attempt she would rate the cake based on deliciousness.

Let’s call each attempt an iteration, so she needs a systematic way to adjust ingredients.



Similarly, in Machine Learning model we need to adjust weights & bias to achieve perfect model. We use optimization technique called

GRADIENT DESCENT.

Let’s recall Gradient Descent from Linear Regression

$$w_j^{new} = w_j^{old} - \eta \times \frac{\partial L}{\partial w_j^{old}}$$

$$\frac{2}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i) \times x_{ij}$$

- iterates over all points

NOTE :

If we are using whole data for a single weight updation is called **Batch Gradient**

Betty realised she could treat her pizza making process as an optimization problem & tired GD to guide herself in right direction.

- She started with **BATCH GRADIENT** she baked several pizzas at once & calculated average loss across all.

Loss = desired deliciousness - actual deliciousness

- She then computed gradients to find direction & magnitude she needs to change to achieve perfect pizza.

But she got tired making this much pizzas.

Let's assume if the data contains 1 million data points (n≈1million)

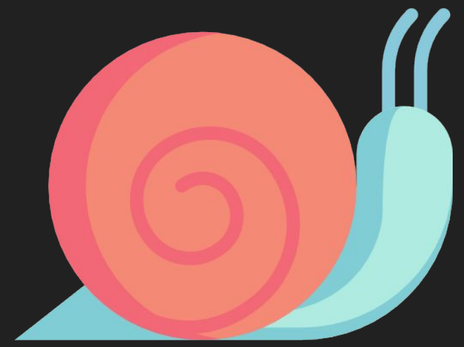
Q. What can be the potential problem while doing Gradient Descent ?

Ans: Due to the

$$\sum_{i=1}^{n}(\hat{y}_i - y_i) \times x_{ij}$$

in the Gradient calculation:

- It causes batch gradient to iterate over all the 1 million data points just to compute a single weight updation
- This makes Batch Gradient Descent **time intensive**



Tired baking pizzas, she decided to repeat the same process but with **small batches of pizza.**

This method of calculating gradients using small batches is called Mini - Batch GD.

Similarly, How do you think we can reduce time taken by batch Gradient Descent in our data ?

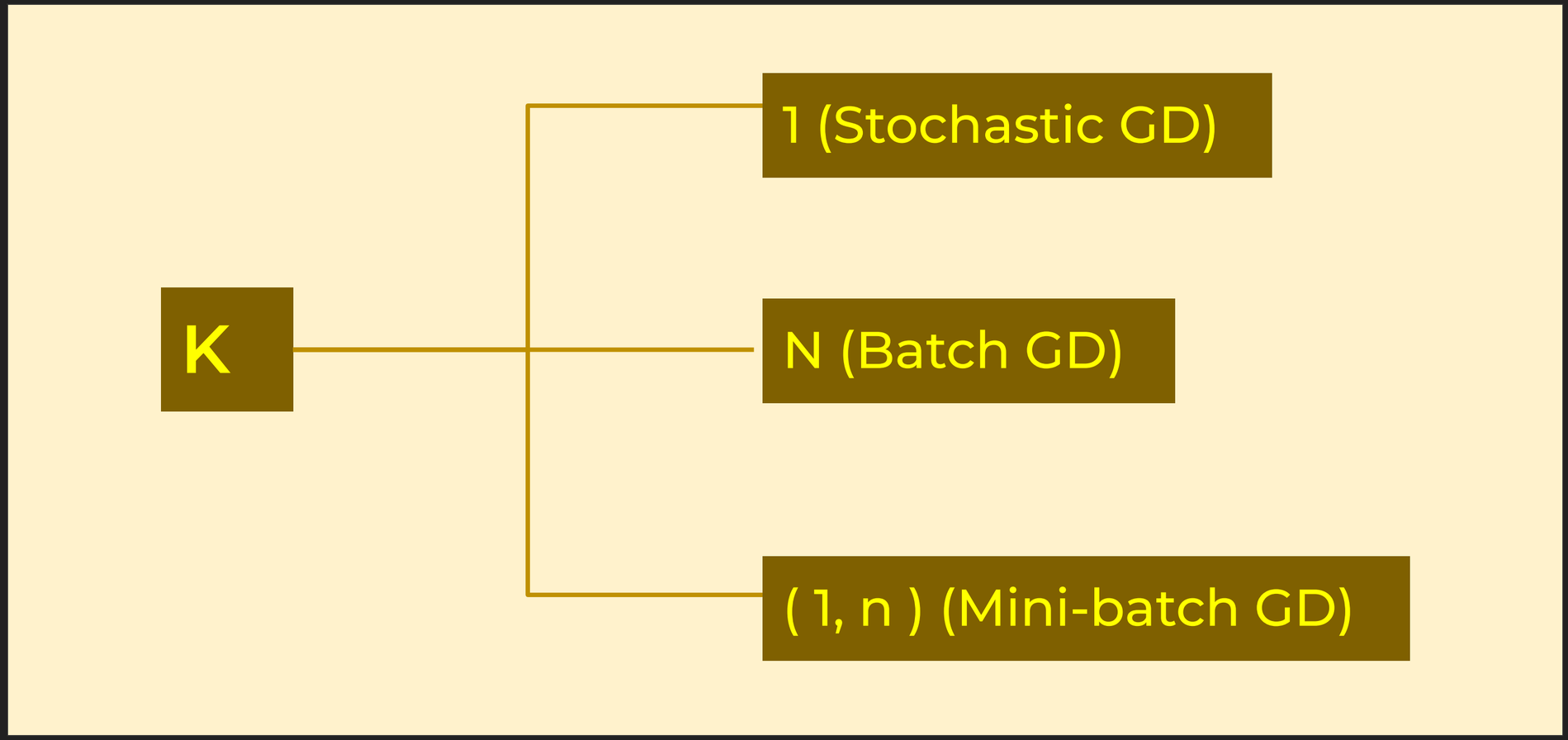
- Randomly choose K data point from dataset and find gradients using them.

$$w_j^{new} = w_j^{old} - \eta \times \frac{2}{n} \sum_{i=1}^{k} (\hat{y}_i - y_i) \times x_{ij}$$

Where K - Batch Size

What can be the possible values of K ?

- K can range from [1, n]

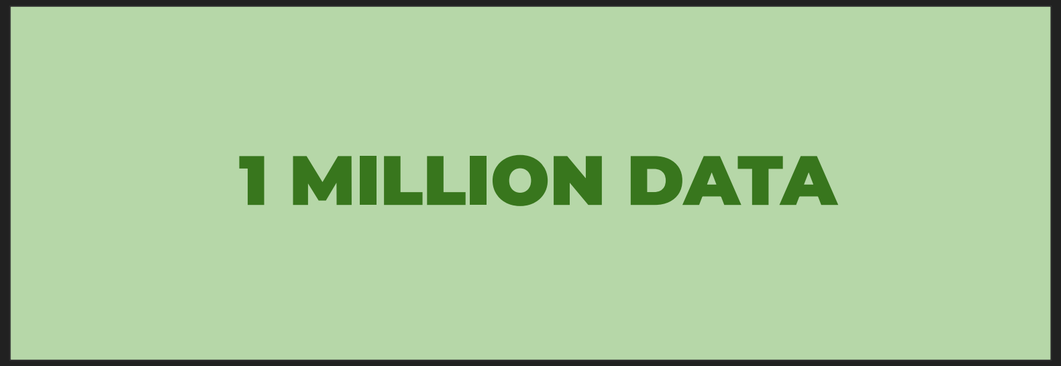


MINI - BATCH GD

Let’s take a quick example:

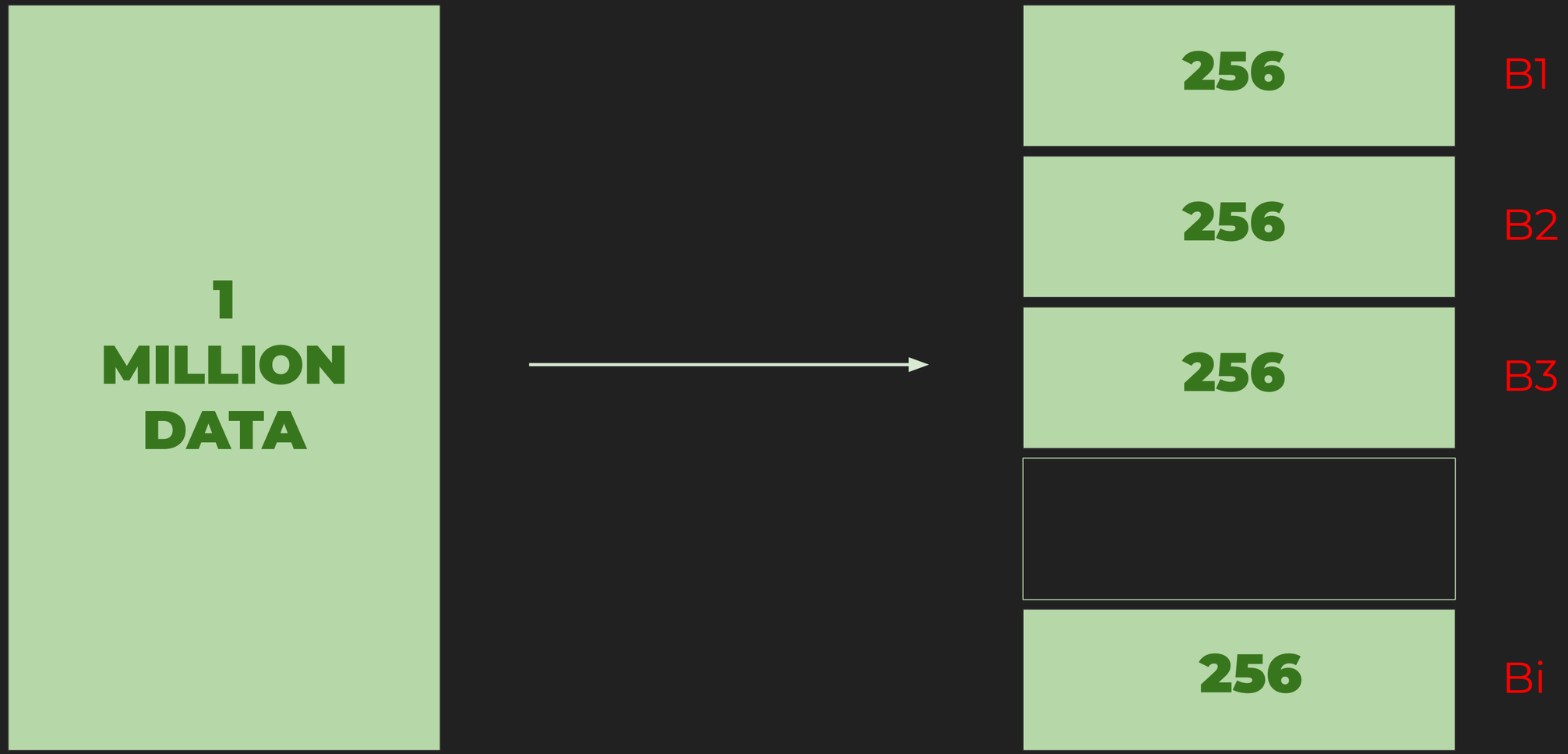
Suppose we have 1 million data (n=1million) and we take k=256,

1. Step1: will be to shuffle the entire 1 million data

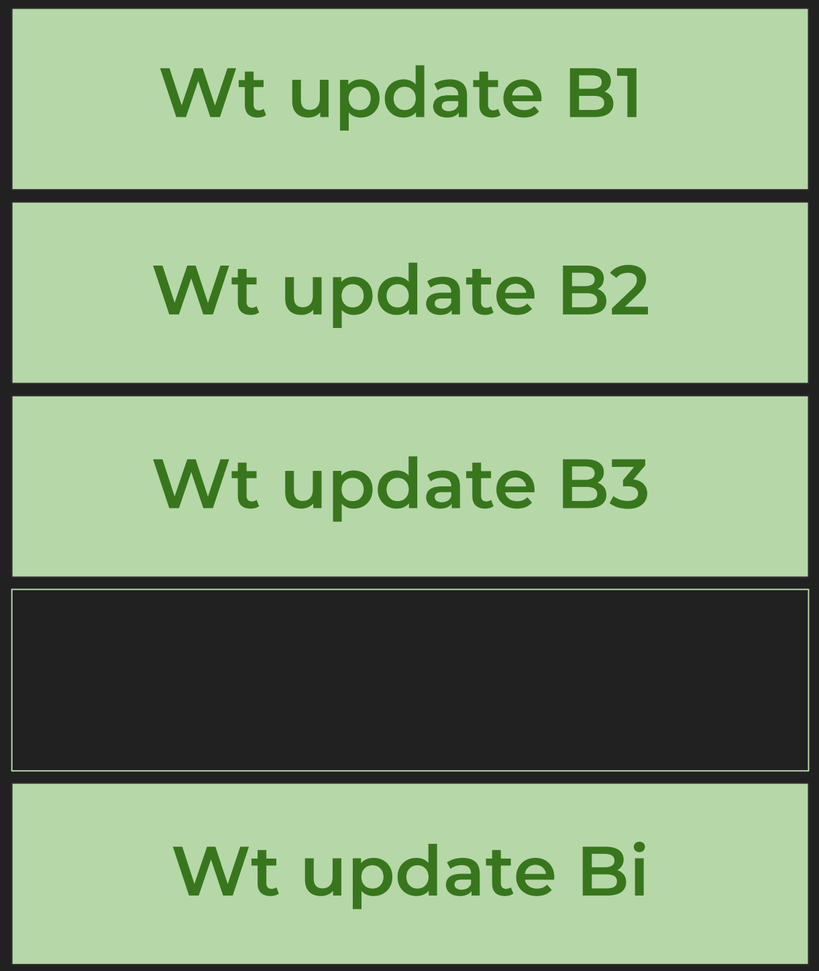


Shuffle the data

STEP - 2 : Create sets of 256 (K) data points and call them batches i.e [B1, B2 —-Bi]



STEP - 3 : Perform weight updates on each set [B1, B2 —-Bi] and lets call them as iteration [iter_1 , iter_2 , —- iter_i]



When Iteration is performed on every set such that one cycle on the whole data is completed, we call it an EPOCH.

In batch gradient, we calculate gradients on 1 million data points,

while in Mini-batch Gradient, we calculate gradients on 256 data points.

Will the gradient value be accurate for mini batch with k = 256 ?

Ans: No, we assume that gradient of 256 data points approximates/close to the gradient of 1 million datapoint.

But this is not always the case,

- **there are times when the gradient value of mini-batch shoots up**
- **or goes down from the optimal value for some set of 256 data points**

Hence, the weight values fluctuates a lot

Batch gradient descent

Mini-batch gradient Descent

Stochastic gradient descent

If in this contour plot, where:

- **outer circle:** denotes weight values which have high loss value,
- and as we go inwards we reach to the optimal weights having zero loss value.

For Batch GD,

The weight reaches its optimal value more in a straight line with least number of fluctuations

Mini-Batch GD,

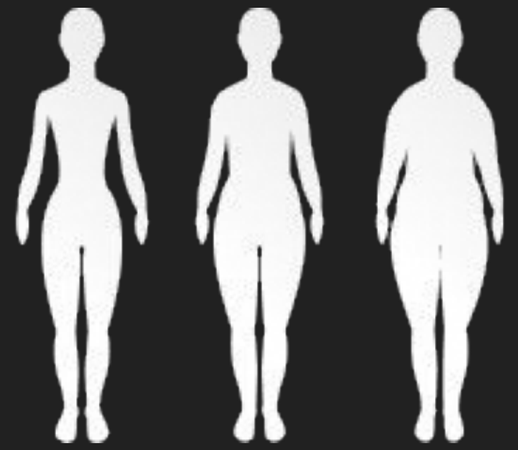
Since it approximates the gradient value, there are fluctuations in the weight value but Mini-Batch GD reaches optimal weight value the fastest.

For stochastic GD,

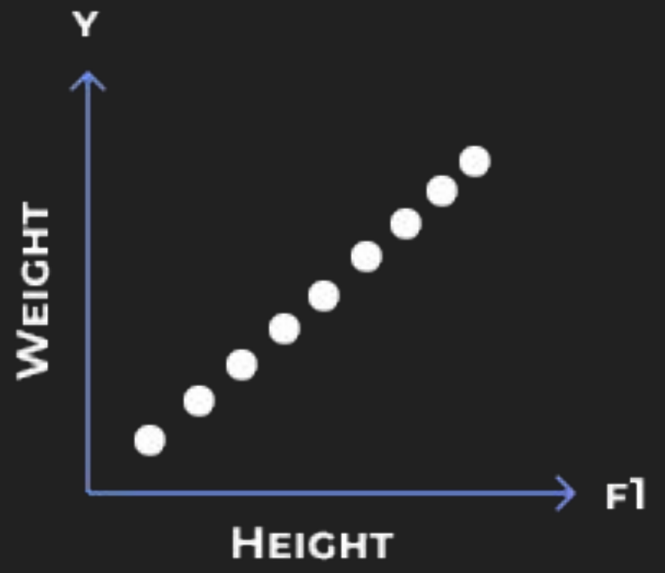
Lots of fluctuations in the weight value since it approximately gradients value of Batch GD, using single data point.

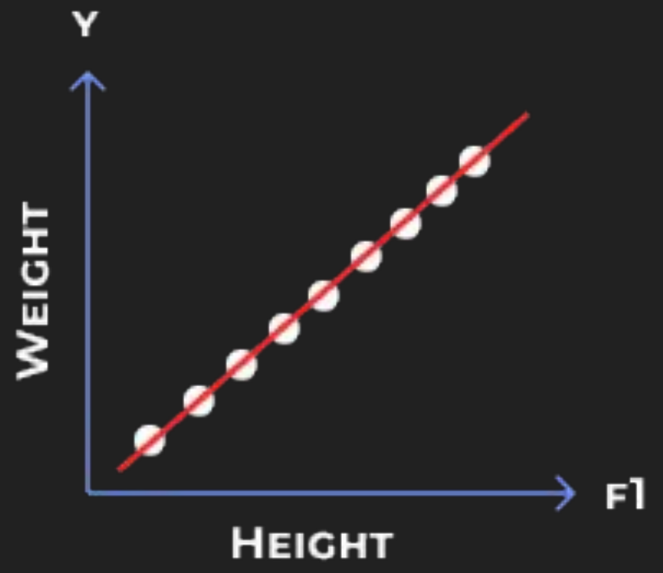
Meet Jenny:

Jenny is a growth spurt she is quickly growing with time



Will linear regression work here ?

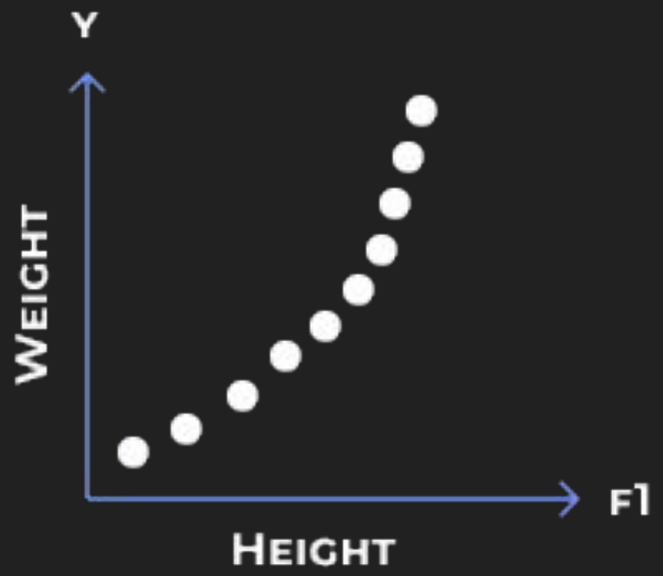




It will work since data is linear.

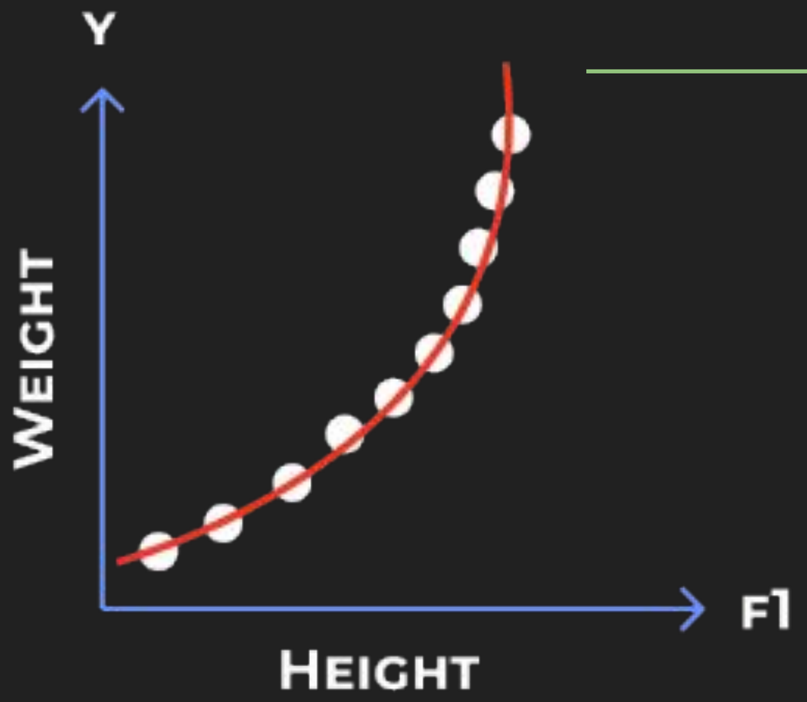
But she started eating lots of fast food resulting which she started gaining fat such that with same height she started gaining weight

Will linear regression work here ?



High loss value since line doesn't fit well

What model will perfectly fit here ?



QUADRATIC MODEL

We need a Quadratic curve to fit such data.

Q. How can we build a LR line Quadratic ?

- We can make linear regression quadratic by adding new feature i.e

$$Quadfunc\ \hat{y}_i \approx w_0 + w_1 f_1 + w_2 {f_1}^2$$

f_1	$f_2 = f_1^2$	y
x_{11}	x_{11}^2	y_1
x_{21}	x_{21}^2	y_2
x_{31}	x_{31}^2	y_3
x_{n1}	x_{n1}^2	y_n

- This is how data will look.

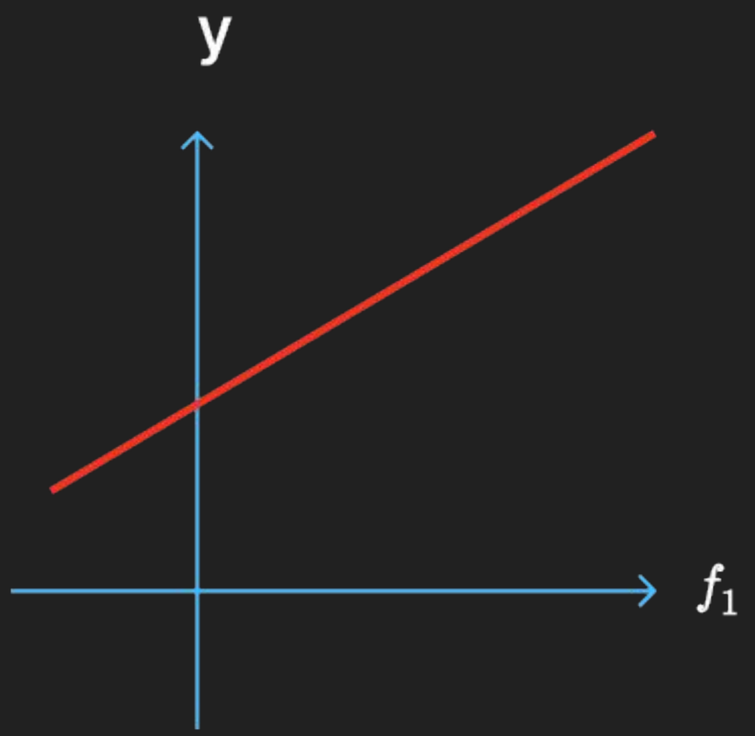
New Feature

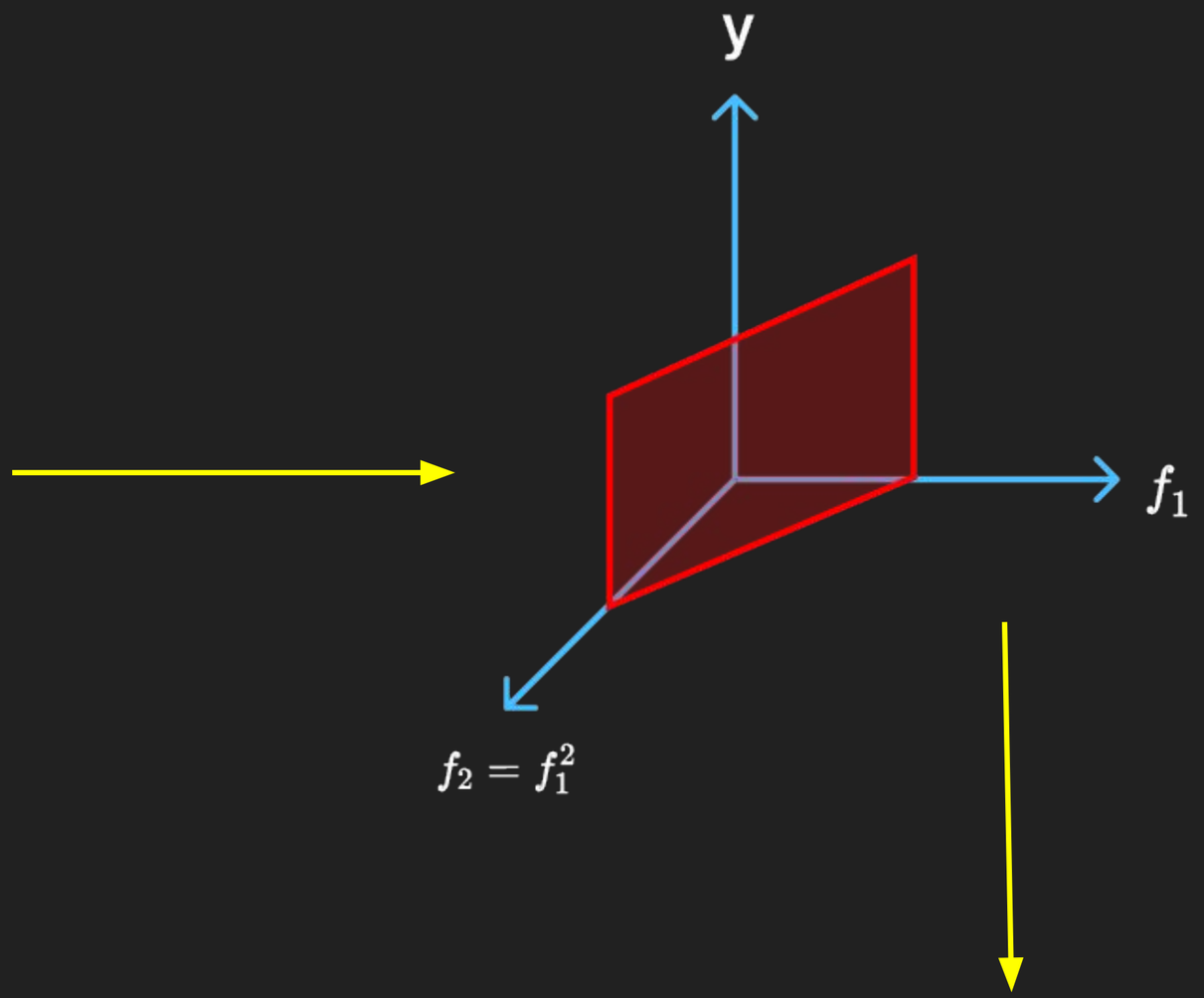
$f_2 = f_1^2$

$$Quadfunc\ \hat{y}_i \approx w_0 + w_1 f_1 + w_2 {f_1}^2$$

$$f_2 = f_1^2 = [x_{11}^2, x_{21}^2, x_{31}^2 \ldots, x_{n1}^2]$$

How does data with $f_2 = f_1^2$ look ?





- Linear Regression model is a plane here.

Does $f_2 = f_1^2$ cause multicollinearity ?

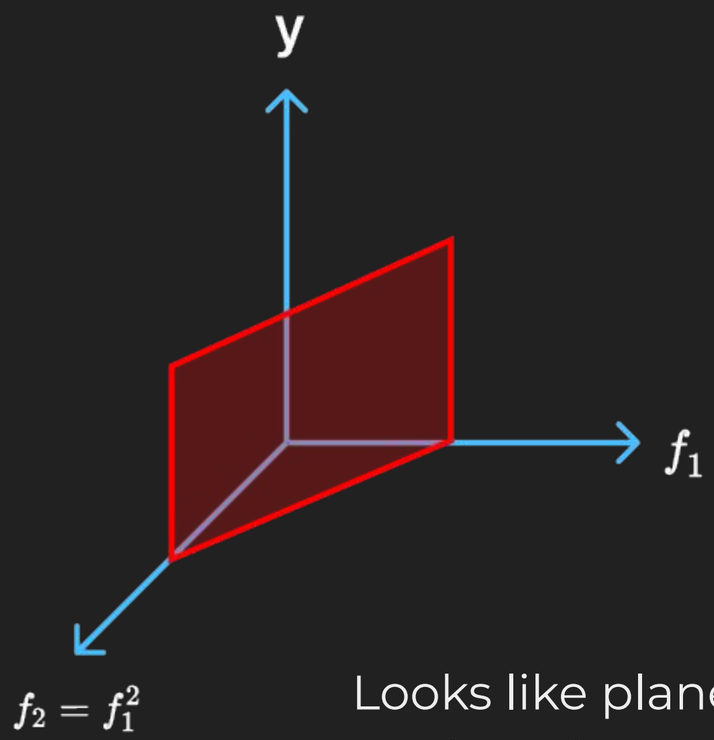
- NO, since it's a nonlinear relationship.
- Multicollinearity will occur only if its a linear relationship

$$f_2 = \alpha f_1 + \beta$$

$$f_2 = \alpha f_1 + \beta \longrightarrow \text{Multicollinearity}$$

$$f_2 = f_1 * f_1 \longrightarrow \text{Not Multicollinearity}$$

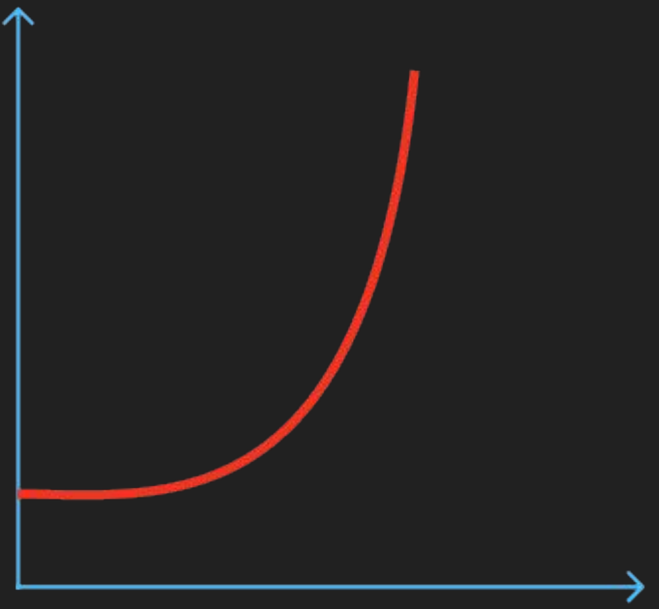
How does adding $f_2 = f_1^2$ makes linear Regression a Quadratic Mode ?



Looks like plane due to high dimension.

Projected

in low dim

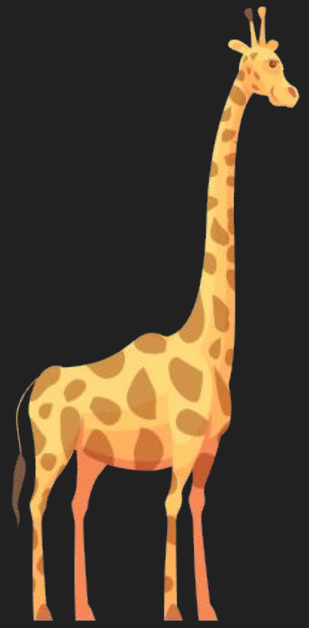


Linear regression is curved due to $f_2 = f_1^2$

Such addition of non-linear feature which make linear Regression model non- linear is called POLYNOMIAL REGRESSION

Now that we learned about Polynomial regression and Quadratic curves

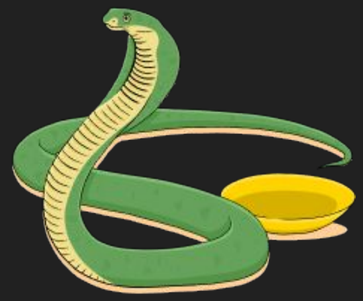
- Let’s understand another concept of Underfitting & Overfitting using our same example of weight & height.



TALL

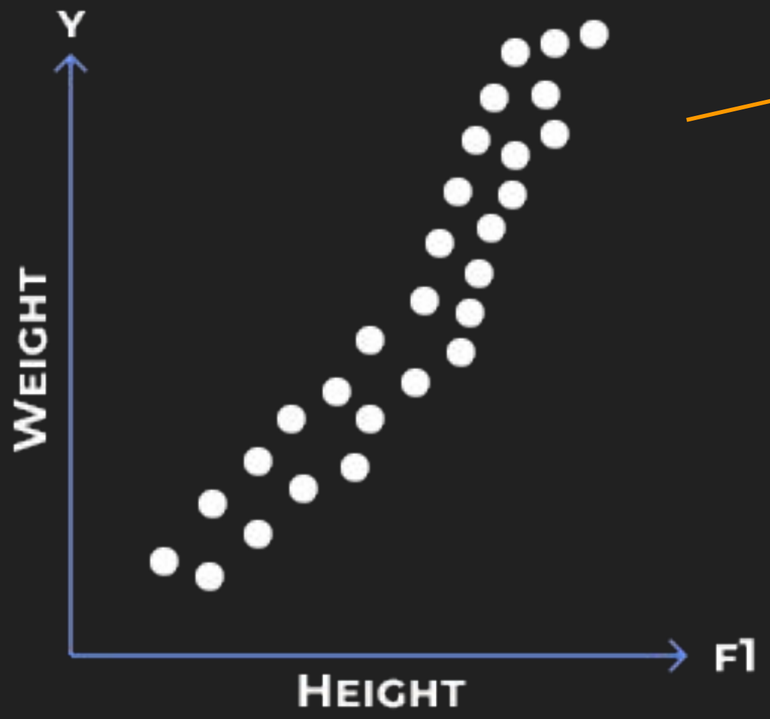
FAT

SMALL



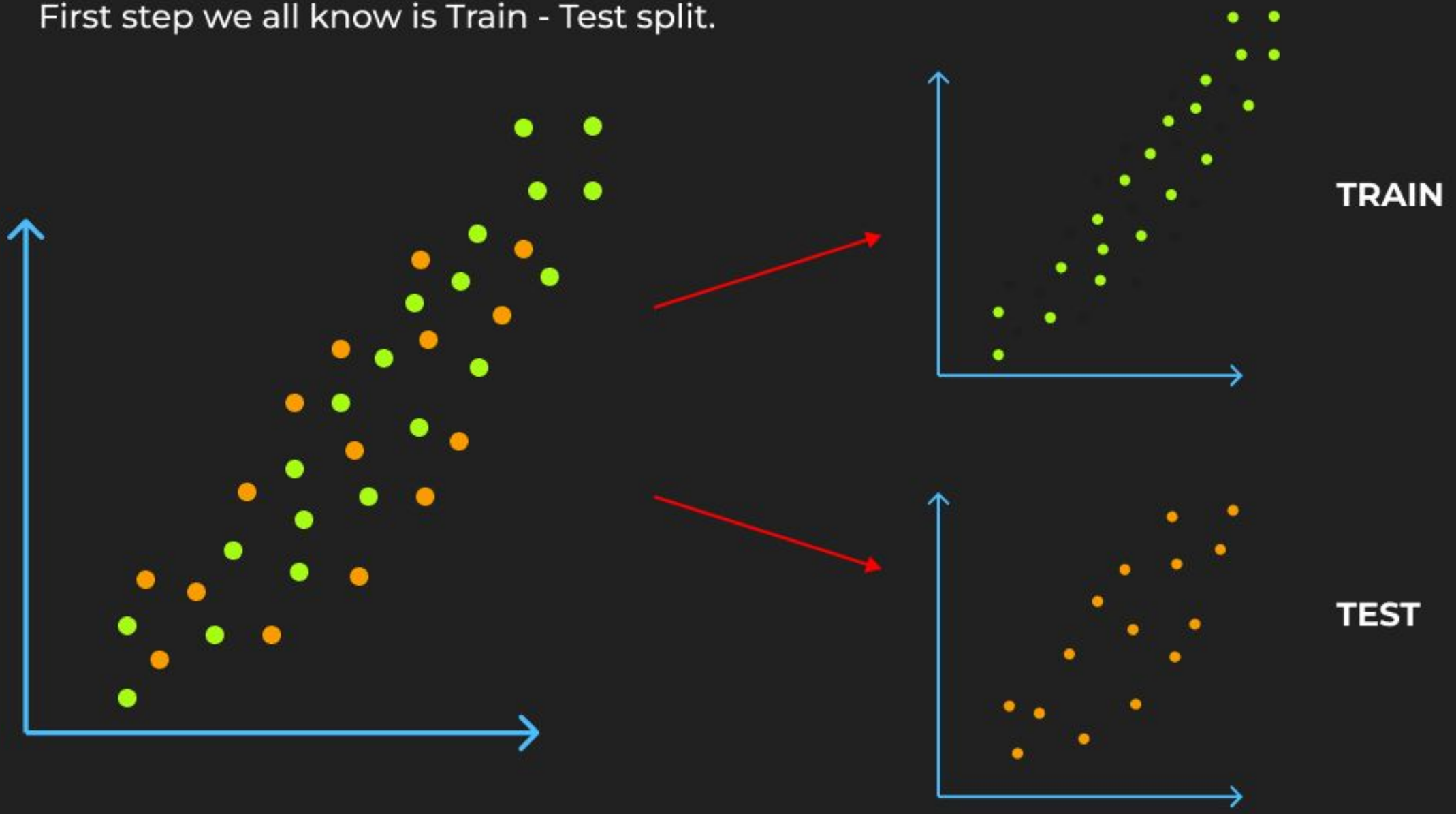
SKINNY

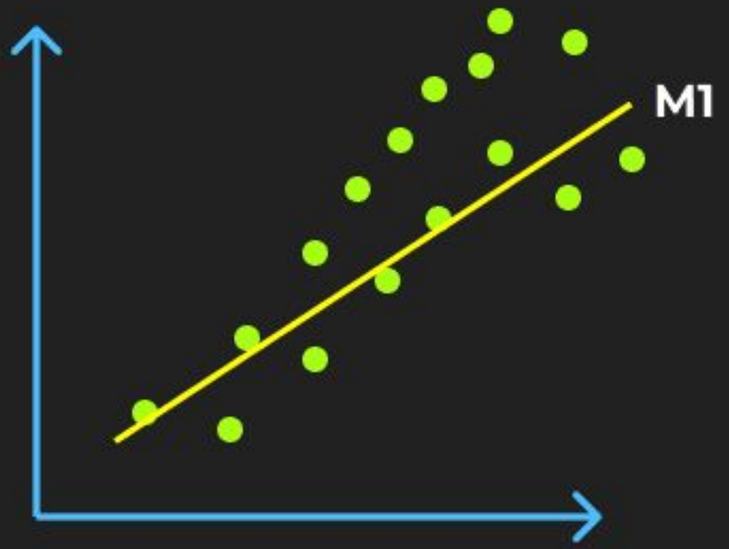
Suppose we are given dataset if height & weight of several people and you are asked to train a ML model to predict weight given height.

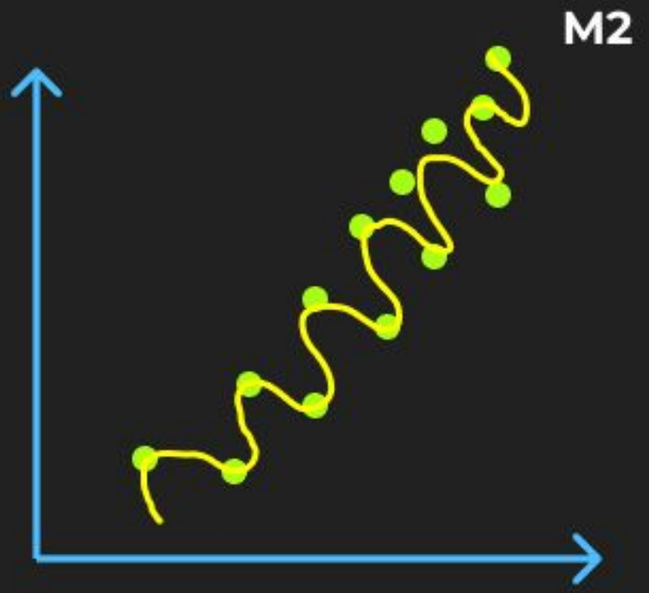


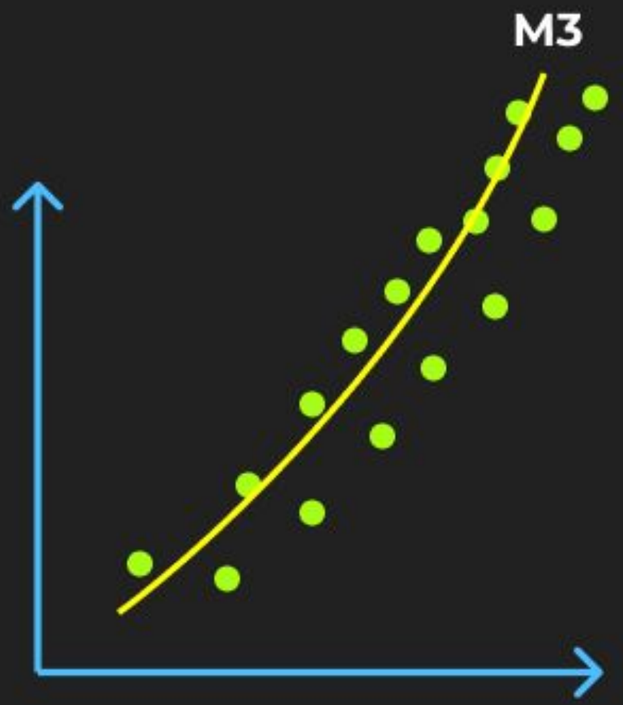
After a certain time person don’t get taller, just more obese.

First step we all know is Train - Test split.



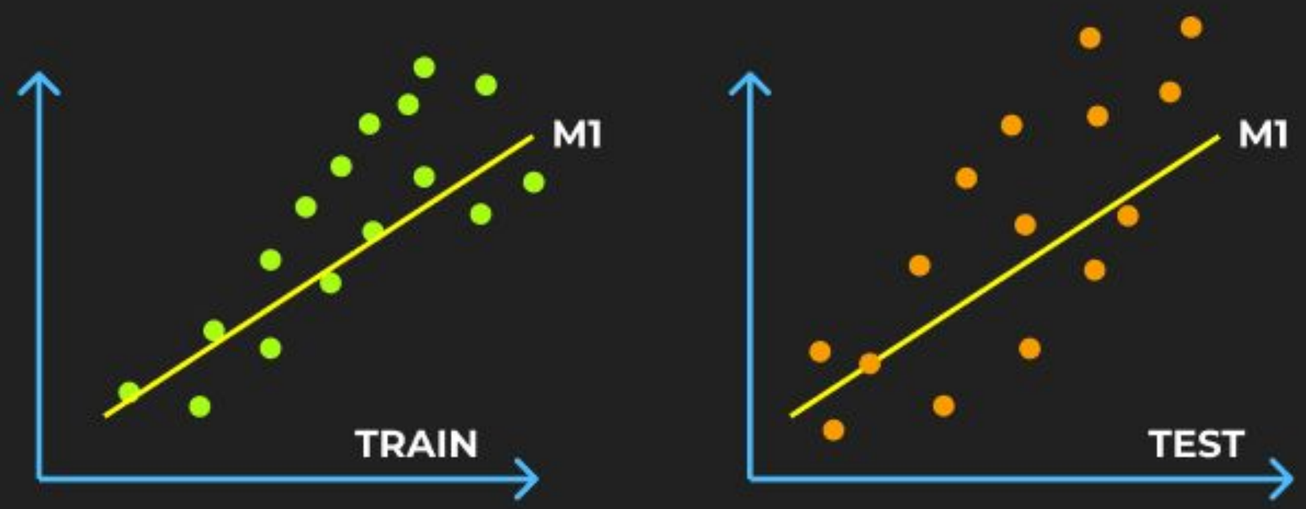






Which three models out of M1, M2, M3 do you think is best ?

Model M1 Clearly M1 model is not fitting well on both training and test data.

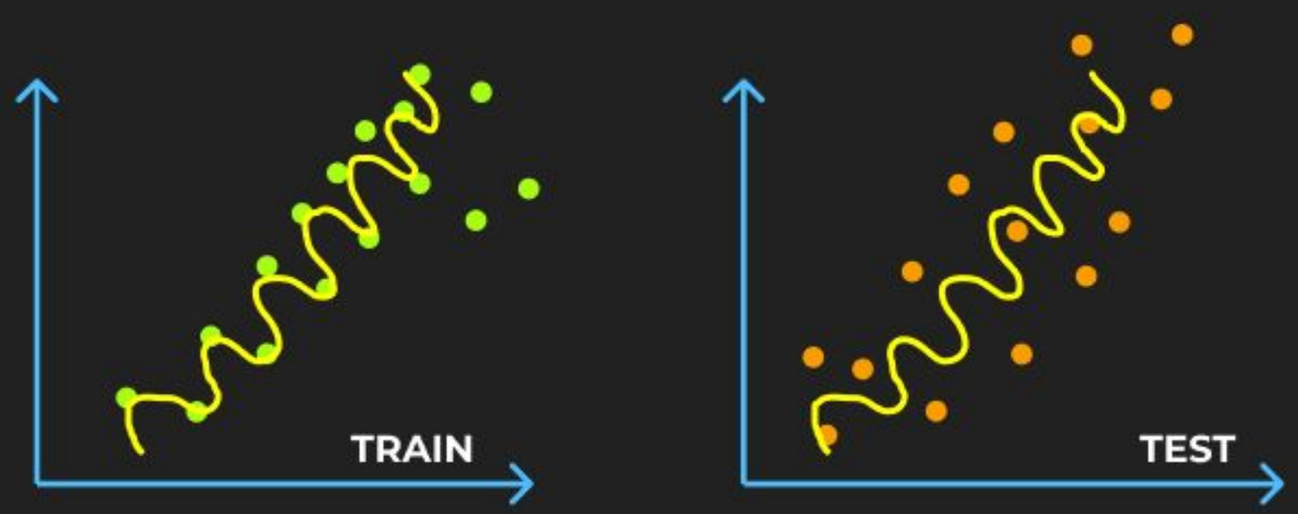


Failed to capture patterns.

This situation is called as Underfitting.

- Imagine M1 as a student who did not study at all for exam, so bound to fail.

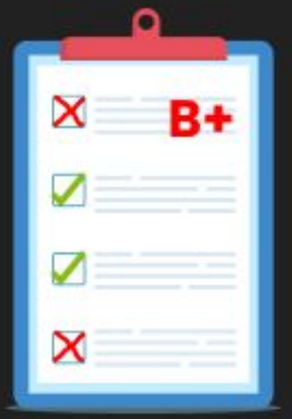
MODEL M2 M2 is performing great on training but only decent on testing.



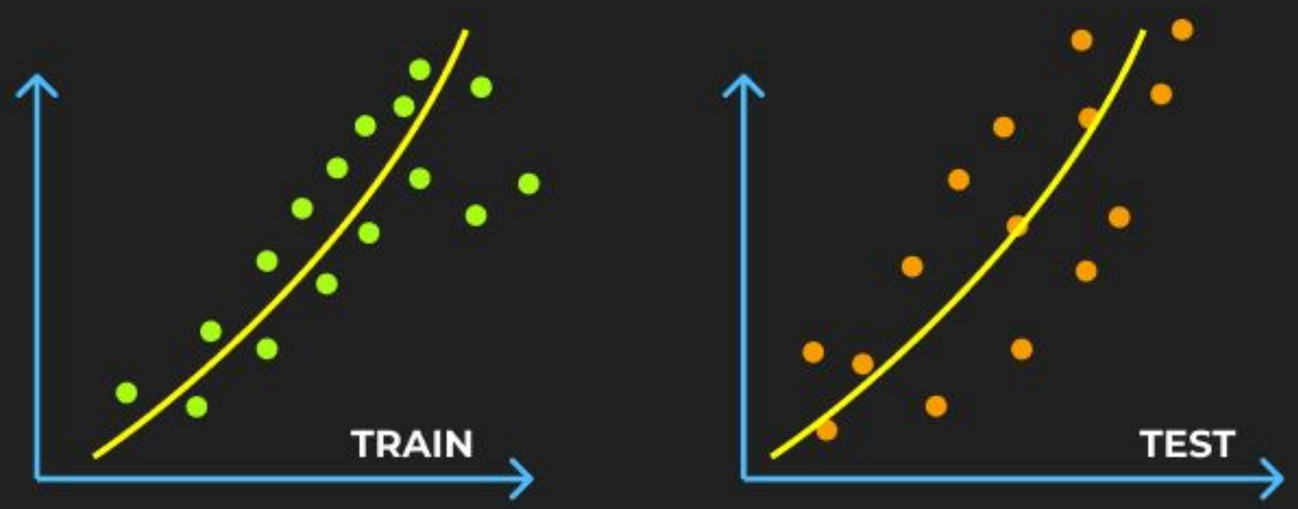
Overcaptured the pattern due to noise.

This is an example of Overfitting.

- Imagine M2 as a student who studied hard bt instead of understanding better, just cramped the Q/A



MODEL M3 M2 is performing great on training but only decent on testing.



Able to see the generic

Balanced between overfitting & underfitting & underfitting

- Imagine M3 as a student who studied smartly and understood the concept.

To summarize,

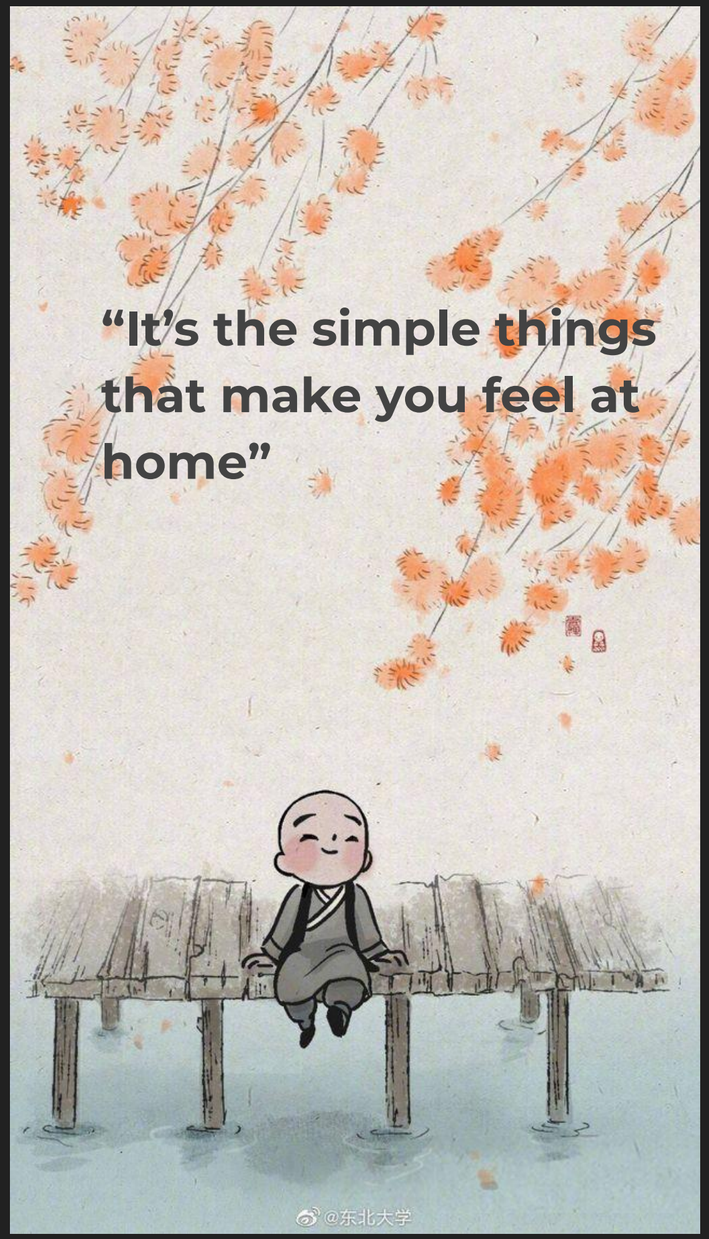
	TRAINING	TESTING
UNDERFIT	POOR	POOR
PERFECTLY FIT	GREAT	GREAT
OVERFIT	BEST	DECENT

Comparison of these three models M1, M2, M3 gave very important concept of :

Generalization-

If there are several ML model, Generalization states that:

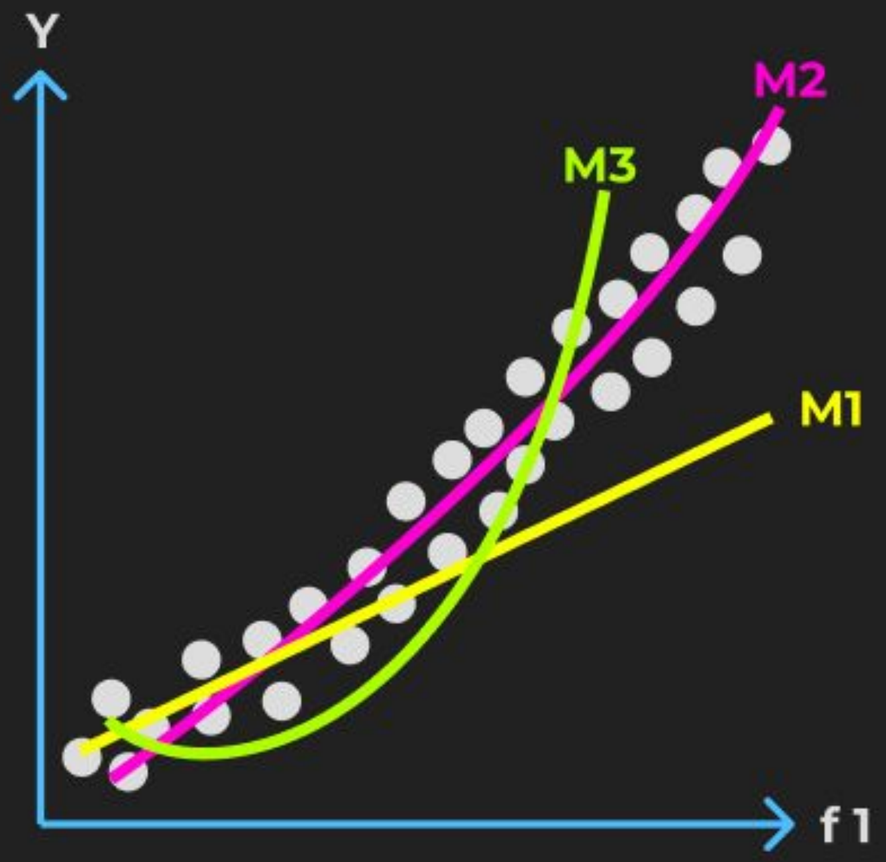
- Always choose the models that learns and understand the data inside out so that it can make good predictions on unseen data.



OCCAM’s RAZOR

There is another rule that we follow while choosing best model.

“There are many solutions to the problem,
always choose the simpler one”



Now suppose given these 3 models :

M1 Linear: f_1

M2 Quad: f_1, f_1^2

M3 Cubic: f_1, f_1^2, f_1^3

Which one should we choose ?

Test Performance

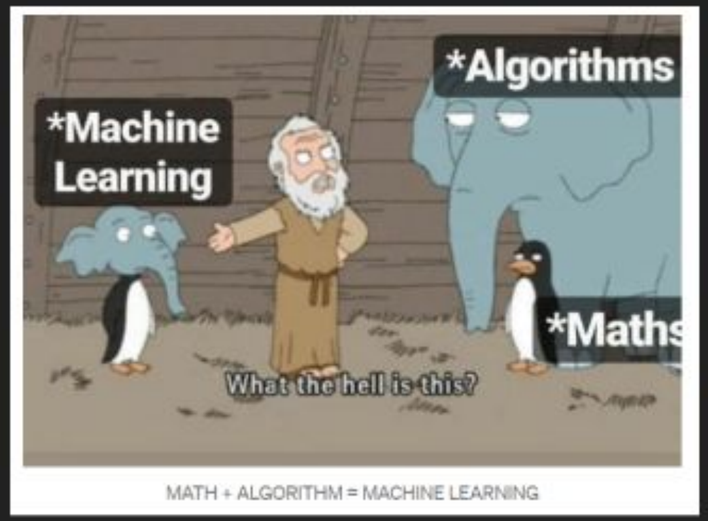
M1	M2	M3
HIGH	LOW	LOW

Apply Generalization & Occam's Razor :

- Clearly, M2 is best

    1. It satisfies generalization since M2 has low test loss values.
    2. Among M3, M2 - M2 is a simpler model as

Number of parameter in M2 i.e 3 **<** **Number of parameter in M3 i.e 4**



Let's understand the maths behind same

Given:

$$LR = w_0 + w_1 f_1 + w_2 f_1^2 + w_3 f_1^3 + w_4 f_1^4$$

f_1 → Original features

f_1^2, f_1^3, f_1^4 → Transformed features

Three different model scenarios :

M1	M2	M3
$w_1, w_2, w_3, w_4 \neq 0$	$w_3 = w_4 = 0$ $w_1 \neq w_2 \neq 0$	$w_2 = w_3 = w_4 = 0$ $w_1 \neq 0$

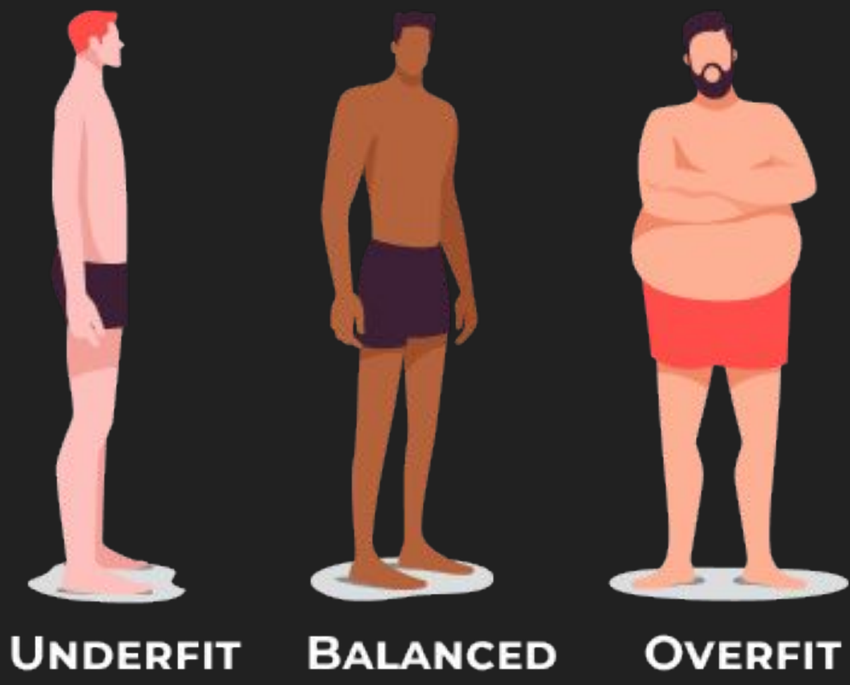
Can you tell which out of M1, M2,M3 is overfitting & underfitting ?

- M1 is overfitting because higher degree weights = ∅, meaning a complex curve.
- M3 underfits as only w1 =/0, meaning a straight line

NOTE :

If weights on lower order polynomial = ∅, more chances of underfitting.

- **M2 is perfectly fit.**



Underfitting & overfitting can also be understood with bias and variance.

Let’s understand Bias & Variance using this game of Archery.

Betty is learning game of archery.

Since she started learning she is very chaotic and just hitting arrow anywhere randomly.

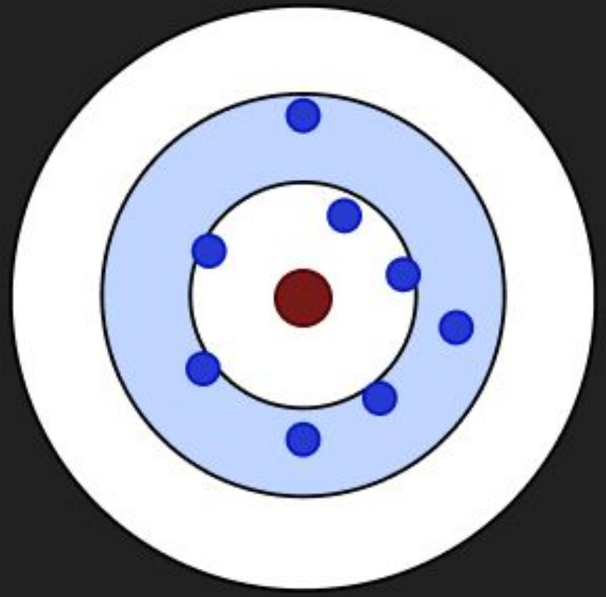
CASE I

This is the case of,

High Bias	High loss, since target is missed everytime.
High Variance	Because she is not consistent, just hitting ramdomly

Betty tried hard to learn, finally she can now hit near the bull's eye. But she lacks consistency

CASE II

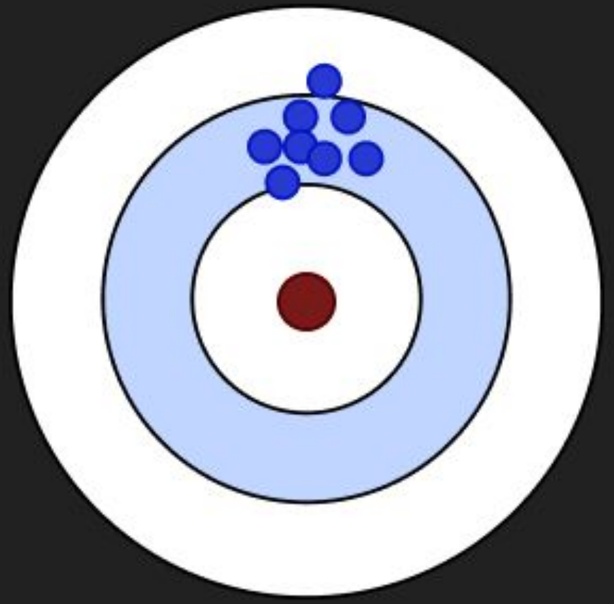


This is the case of,

LOW BIAS	Because she is hitting near target.
HIGH VARIANCE	Due to lack of consistency.

So she decided to work on consistency with time, she learnt the art of consistency but was still missing target.

CASE III

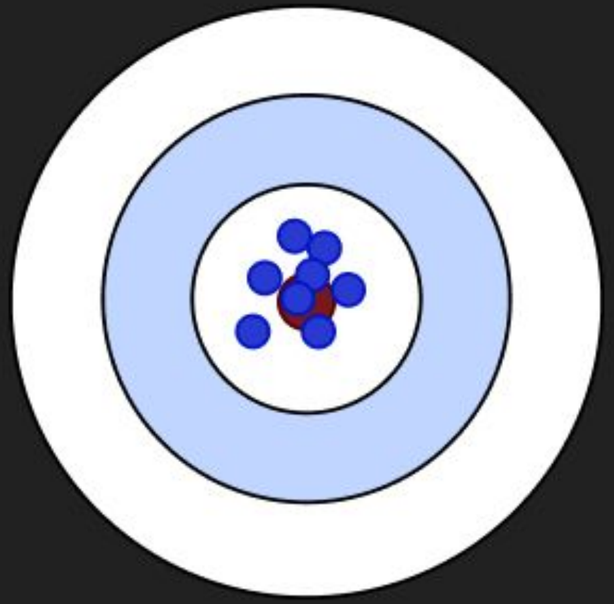


This is the case of,

High Bias	High loss, since target is missing
Low Variance	because shots are consistence

Finally, after years of practice she mastered the art of archery.

CASE IV



This is the case of,

LOW BIAS	To the bull's eye
LOW VARIANCE	Consistent

Q. What is Bias & Variance in ML ?

- They are kind of errors

1. High Bias leads to high error
2. Similarly, high variance leads to high error.

Q. Which among the 4 scenarios above have the least error ?

Ans: **CASE IV** ,

(Low variance & low bias)

But how is Bias & Variance related to overfitting & underfitting ?

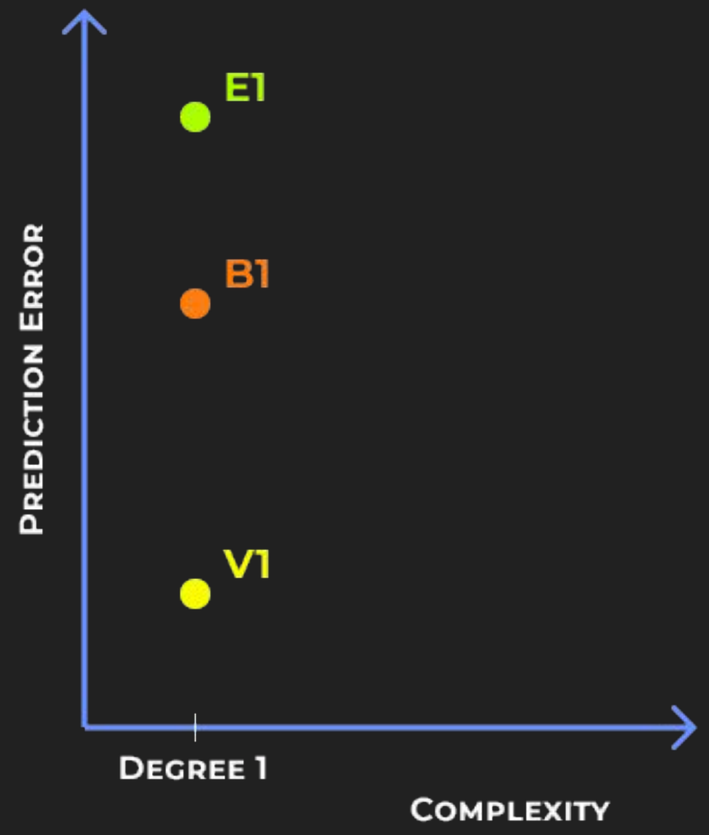
- We know if model is very simple (low degree low variance) , there are high chances of High test error which means High Bias.
  - **Implies underfitting**

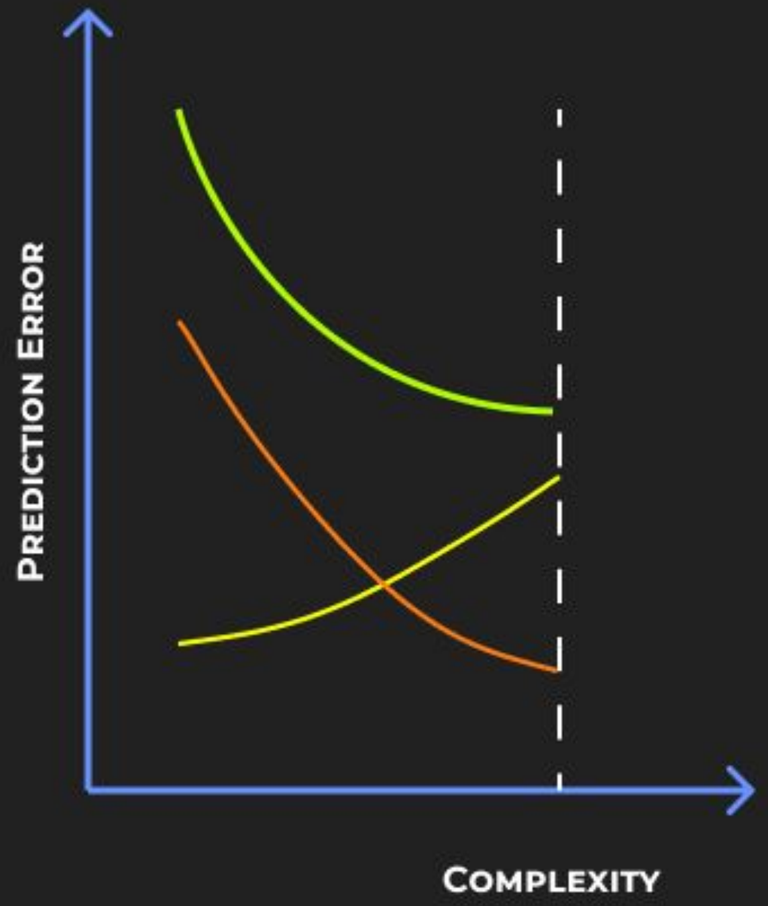
Let’s try to plot this

B1 - Bias for Degree 1

V1 - Variance for Degree 1

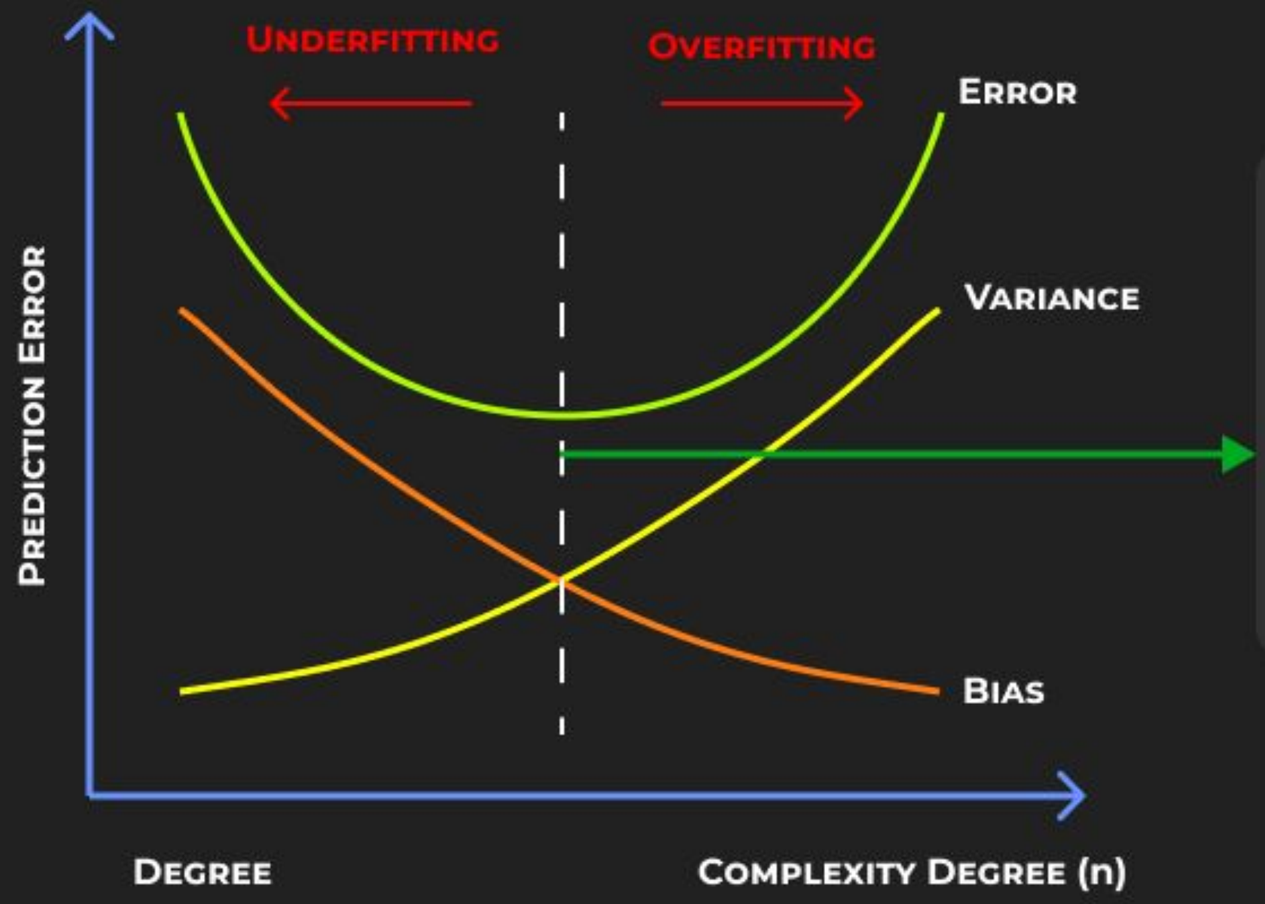
E1 - Error for Degree 1





- Now, if we start increasing degree i.e Increasing Variance Bias will start falling and so the error.

- But after a certain threshold when model become too complex Prediction error will start increasing because model will start overfitting.



A perfect model lies in between Bias & Variance. Hence, there is always trade off between **Bias & Variance.**