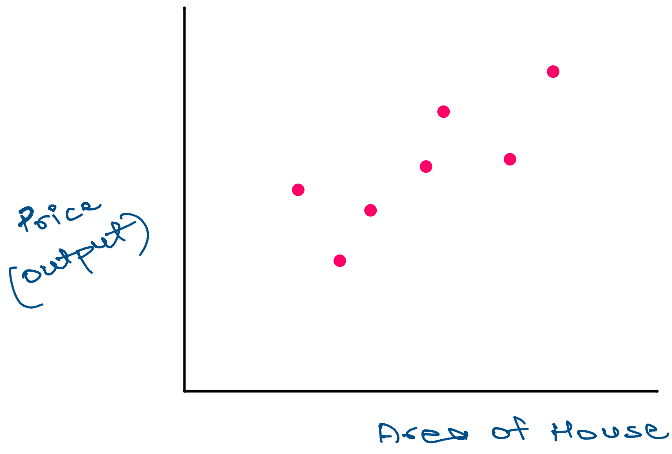
Linear Regression Algorithm

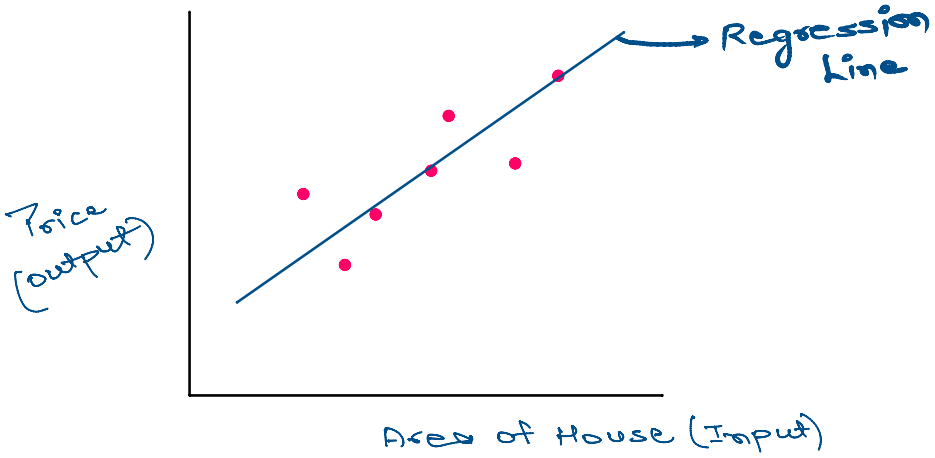
21 November 2024 06:57



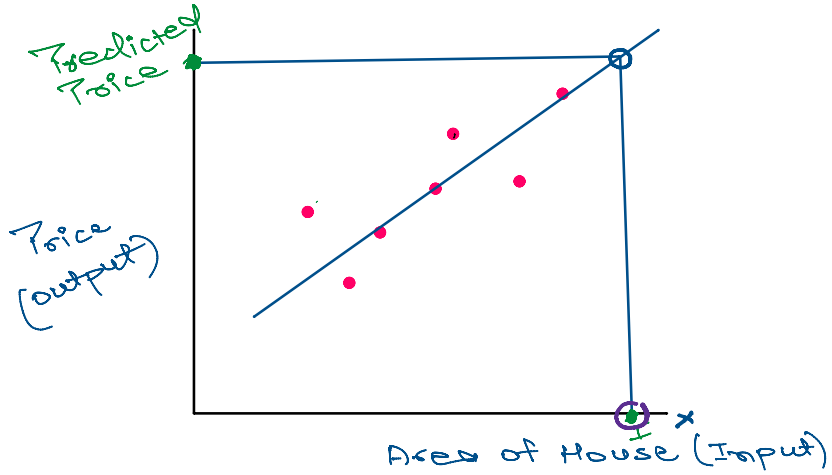
Past Data

Input: Area of House	Output: Price
2133	$800K
3022	$977K
1675	$698K

In Linear Regression Algorithm, we need to pass a line through the data such that it is closest to all the datapoints.

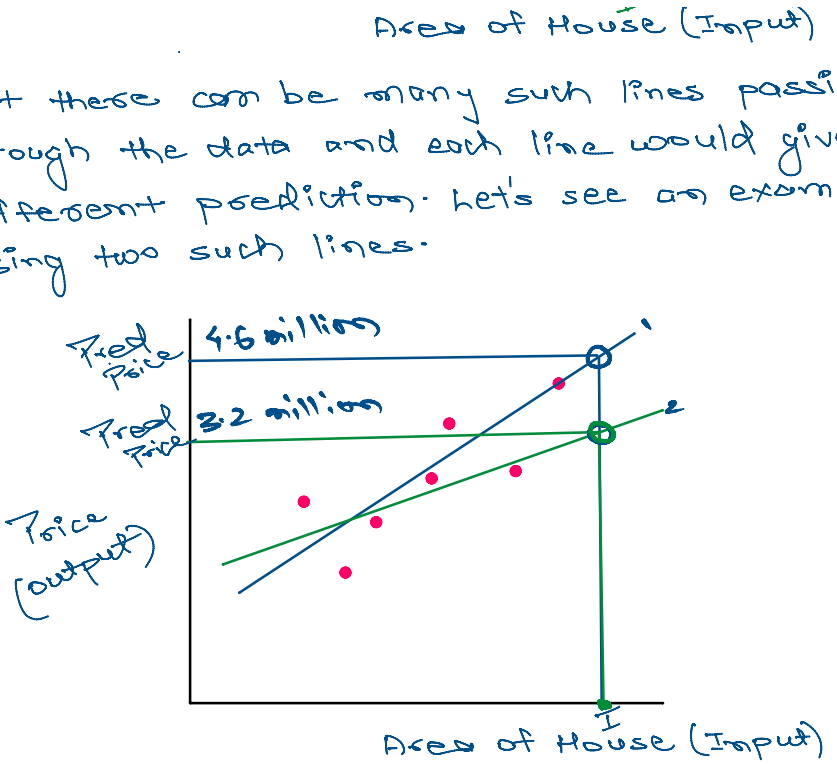


Once we have the regression line passing through the data, we can use the line to make predictions.



Real Estate

But there can be many such lines passing

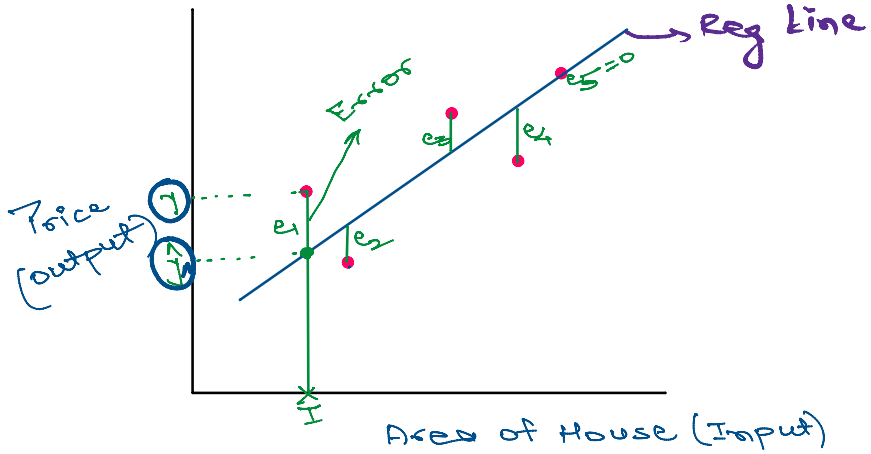


As we can observe above, each line gives different predicted output for the same input.

So which line to use then for final prediction?

The regression line we use finally is the one which is closest to the datapoint since that is the line which has learned the relationship b/w input & output with least errors. This line is called 'Best Fit Line'.

Now, what do we mean by an error?



$\hat{J}$ - cap

Here,

J = Actual Price

$\hat{J}$ = Predicted Price for the given input (I)

e_i = Errors in prediction.

Error = Predicted − Actual

$= \hat{y} - y$

To get the total error made by the line, we add all the errors:

Total errors $= e_1 + e_2 + e_3 + e_4 + e_5$

But here some errors would be positive and some negative and might cancel out each other, so to avoid this issue, we square the errors before adding them.

$$\text{Total Error} = e_1^2 + e_2^2 + e_3^2 + e_4^2 + e_5^2$$

We call it sum squared Error (SSE).

But there is one small problem. If there is a large training data, then SSE will be a very large value and can't be saved in the memory. So we take average of SSE.

$$\frac{1}{n} \times SSE$$

where SSE can be written as,

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

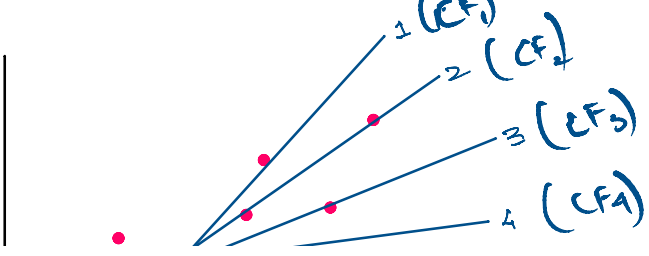
Hence,

$$\frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 \longrightarrow \text{Mean Square Error.}$$

and finally,

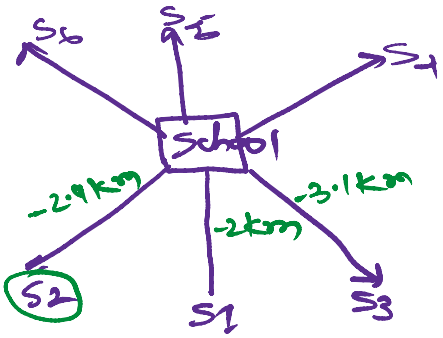
$$\text{Cost Function } (J(\theta)) = \frac{1}{2n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2$$

This cost function is used to find the "Best Fit Line". How? Let's see →



Modulus $|-10| = 10$

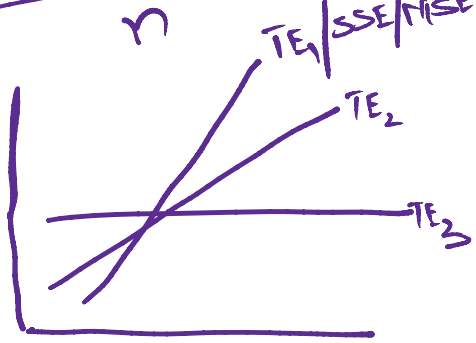
$(-10)^2 = 100$

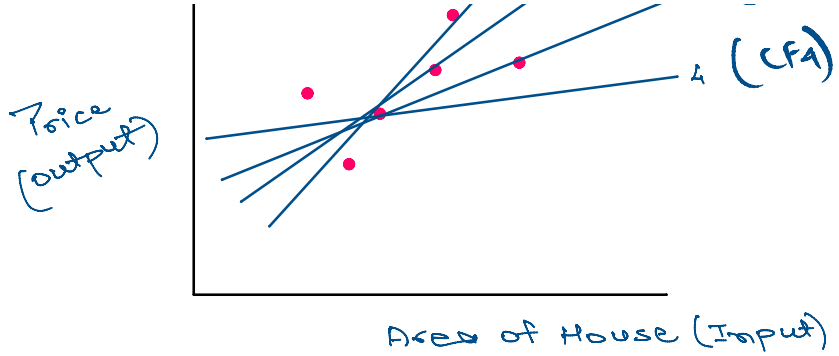


$(-15)^2 = 225$

4876251395135

n

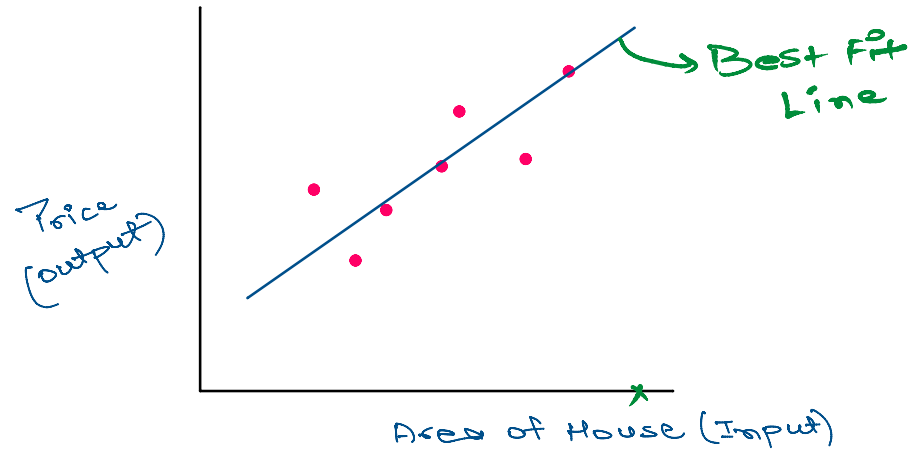




To find the Best Fit Line, the algorithm draws many lines passing through the datapoints and then calculate "Cost Function for each line" which is nothing but the total error made by each line. Then it compares the Cost Functions →

CF(line1) vs CF(line2) vs CF(line3) vs .....

Now, whichever line has least cost function will be our "Best Fit Line".



This line can be represented as an equation which is in this form →

$$y = mx + c$$

that we write as:

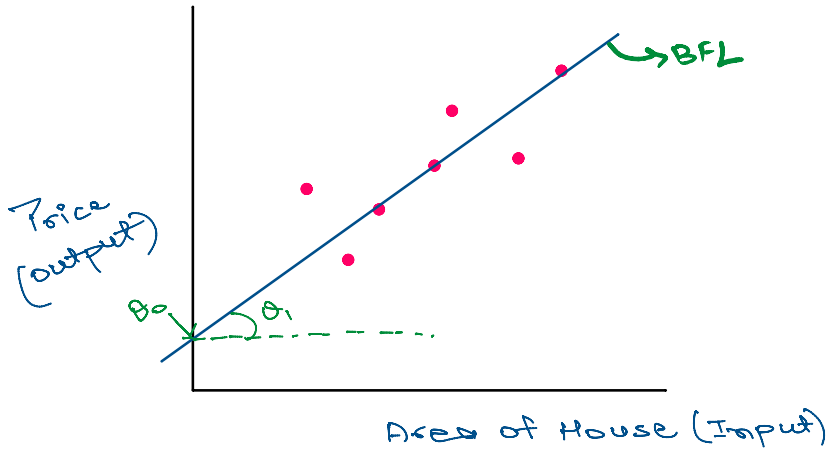
$$\hat{y} = \theta_0 + \theta_1 x$$

and in final form it will look like →

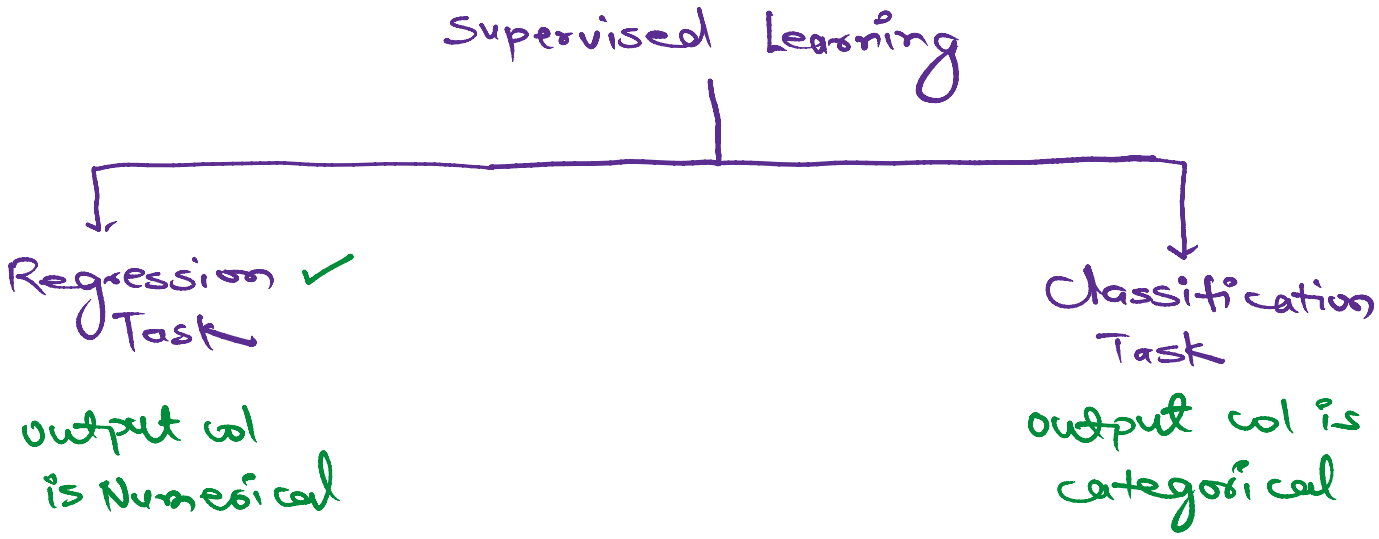
$$\text{Predicted Price } (\hat{y}) = 1.8 + 0.9x \quad \text{(suppose)}$$

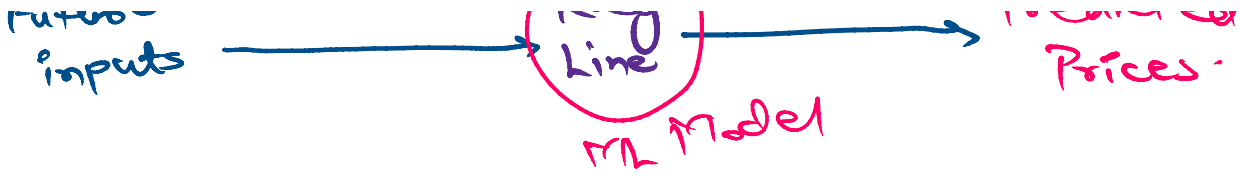
where the value of θ_0 and θ_1 will be taken from the best fit line and x is the future

where the value of θ_0 and θ_1 will be taken from the best fit line and x is the future input.



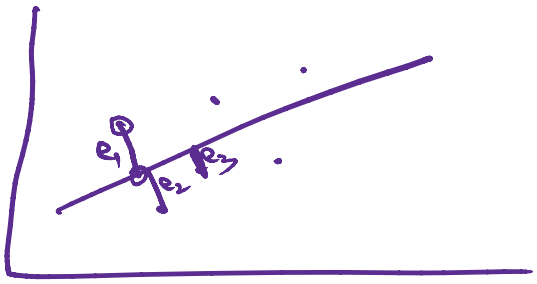
In this process, the algorithm has to try 100s of combination of θ_0 & θ_1 to find the ones that minimize the cost function.





$$\text{Errors} = P - A$$

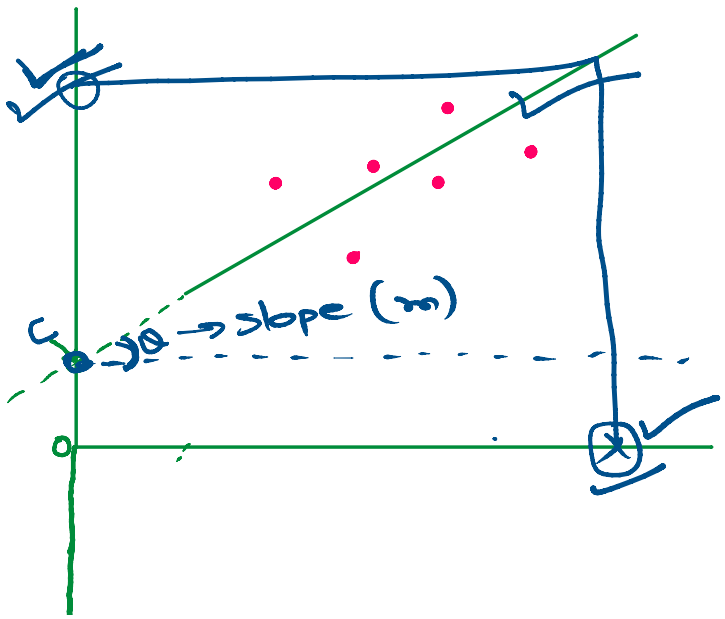
$$= \hat{y} - y$$

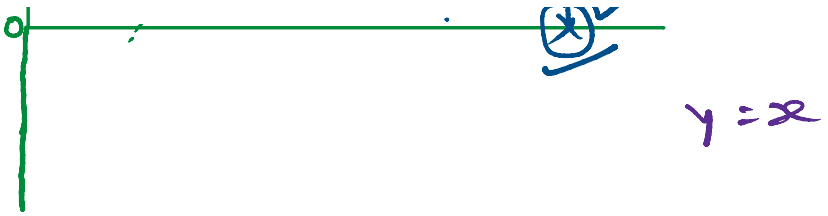


$$\text{Tot. Error} = e_1^2 + e_2^2 + e_3^2 + e_4^2$$

$$= (\hat{y}_1 - y_1)^2 + (\hat{y}_2 - y_2)^2 + (\hat{y}_3 - y_3)^2 + \dots$$

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





FI → $\hat{y} = mx + c$ → PO

↓ (4)

$\hat{y} = 1.1x + 0.9$ → BFL

$\hat{y} = 1.1 \times 4 + 0.9$

$\hat{y} = 5.3$

↓

Simple Linear Regression

Multiple Linear Regression →

Input cols

x_1 No of bedrooms	x_2 Age	x_3 No of floors	x_4 Area of House	Output col ↓ Price

(5D)

$$\hat{y} = m_1x_1 + m_2x_2 + m_3x_3 + m_4x_4 + C$$

$$= 1.1 \times \text{No of Bedrooms} + (-1.26) \times \text{Age of House} + 1.8 \times \text{No of Floors} +$$

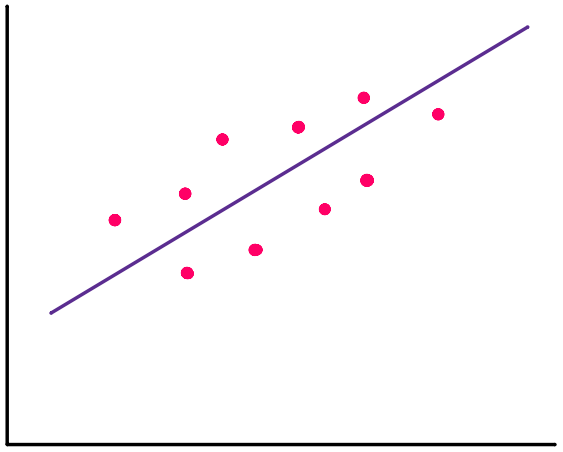
$$2.1 \times \text{Area of House} + 1.19$$

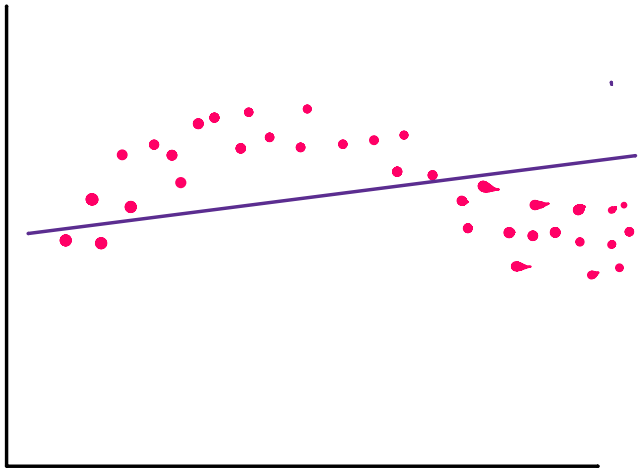
1

$2.1 \times$ Area of House $+$ 1.19

↓ Base Price

Assumptions →

① 

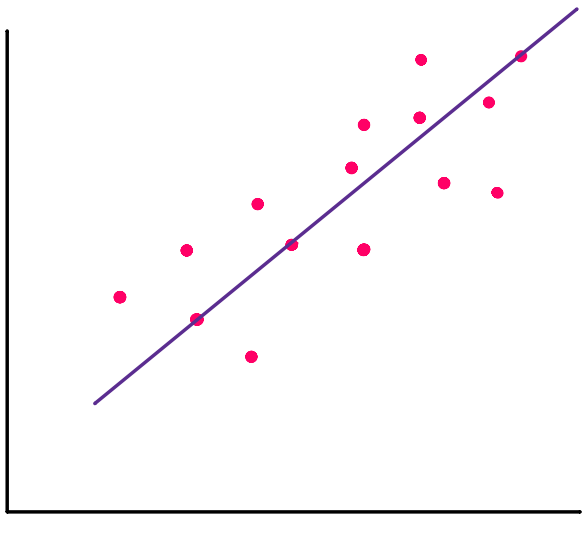


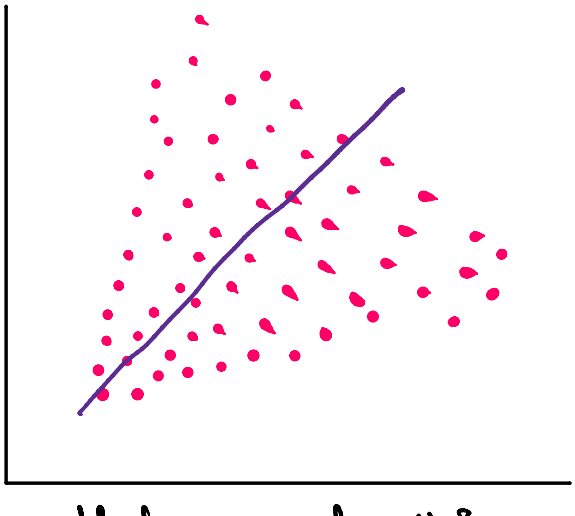
② $e_1 + e_2 + e_3 + e_4 + \cdots\cdots$

↓

zero or close zero

Then you can use Linear Regression on this data.

③ 



Heteroscedasticity

Homoscedasticity ✓

Heteroscedasticity ✗

(4) 'Multicollinearity' should not be present in the data.

When input cols are highly correlated to each other that is called as multicollinearity.

Features / input

Rooms | Area | Floors | Len | Breadth | Price

VIF (variance Inflation Factor) helps in identifying and removing multicollinearity.

If any col has VIF > 5

Delete that column.

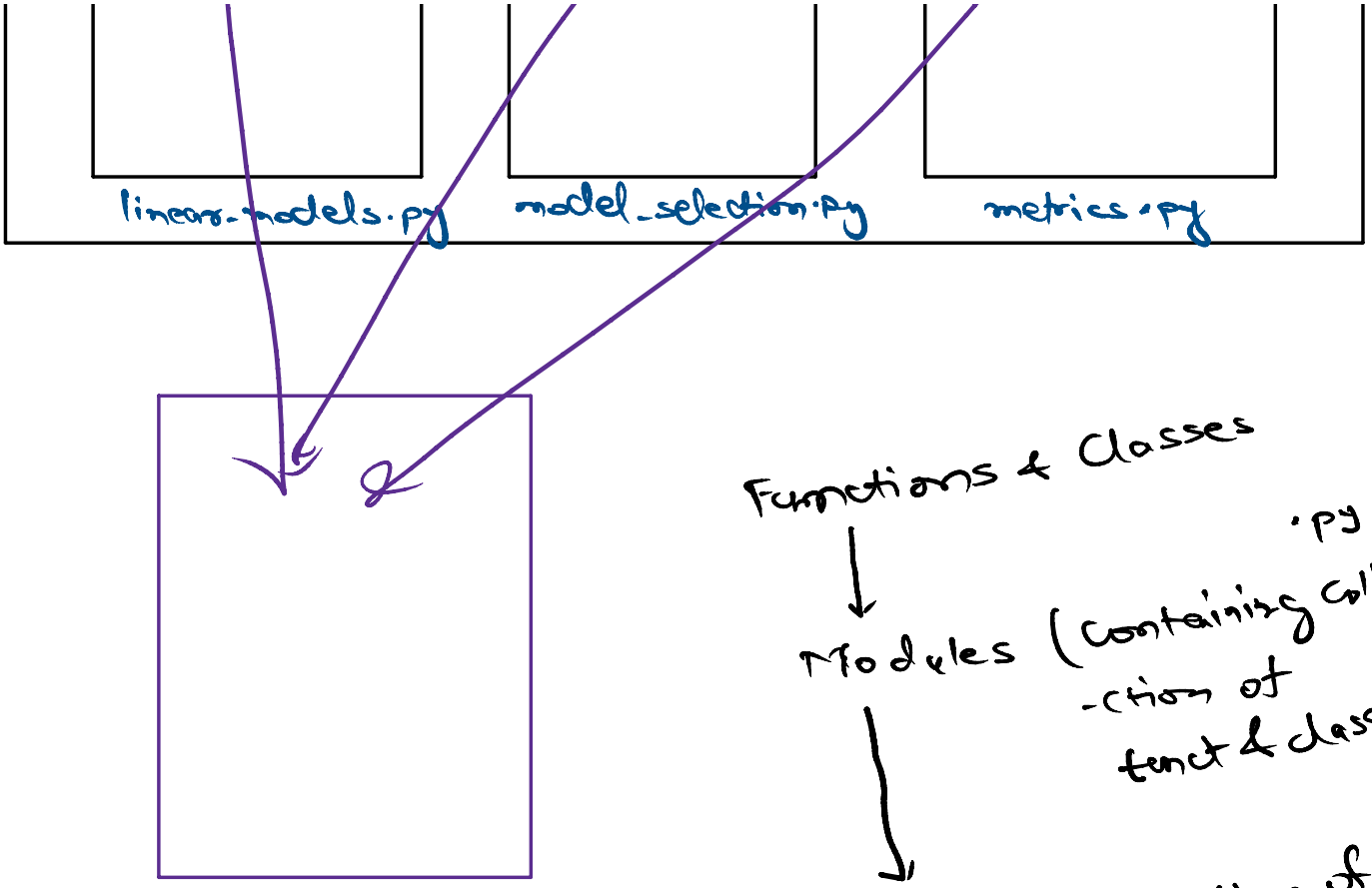
Sklearn → Library / Package

Modules

class LinearRegression:

def train_test_split():

def r2_score():



Functions & Classes

↓

Modules (containing collection of funct & classes) .py

↓

Package (collection of modules)