Introduction to Machine Learning

Machine Learning (ML) is the subset of artificial intelligence that focuses on building systems that can learn from and make decisions or predictions based on the data.

In simple terms, ML enables computers to improve their performance on tasks by learning from experience (data) rather than relying on hard coded rules.

⊛ Key Components of Machine Learning:

1. **Data** : the foundation of ML ; used for training & testing models.
2. **Algorithms** : Mathematical methods that process data to identify patterns or make predictions
3. **Model** : the representation of knowledge learned by an ML algorithm.
4. **Features** : Input variables used for prediction or decision making.
5. **Labels** (In Supervised learning) : Known outcomes used to train the model.

⊛ Categories of Machine Learning :

① **Supervised Learning** : The models learns from labeled data (eg predicting house prices based on historical data)

2. **Unsupervised Learning** : the model identifies Patterns in unlabeled data (eg clustering customer segments).

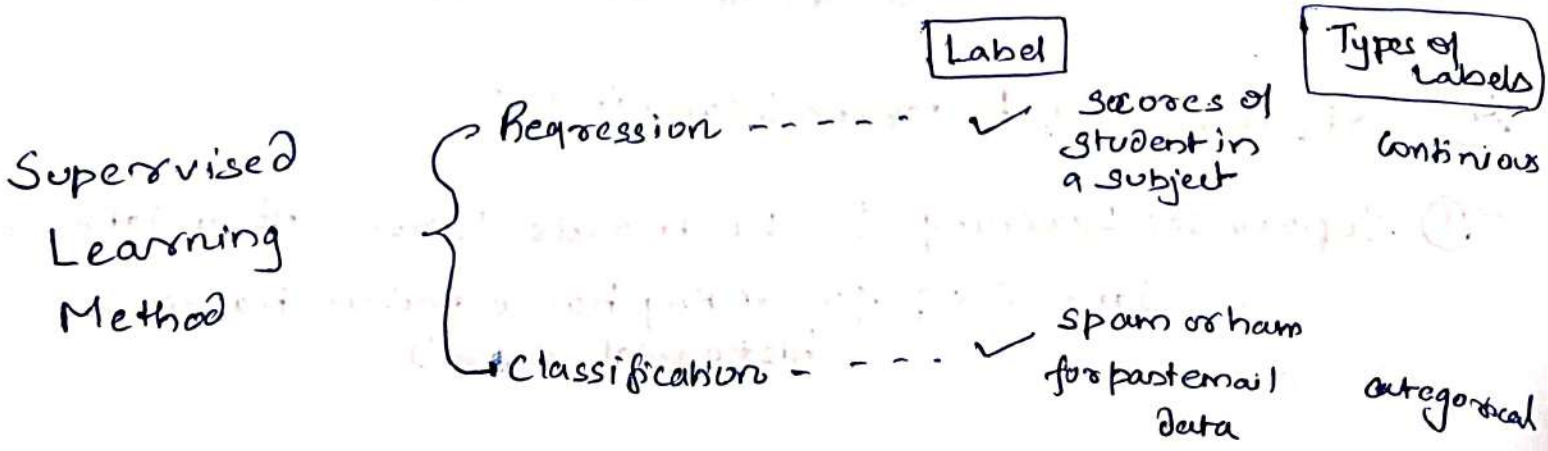
3. **Reinforcement Learning** : the model learns through trial and error by recieving rewards or penalties (eg training a robot to walk)

✳ **Machine Learning models can be classified into the following three types based on the task performed and the nature of the output**

① **Regression** : the output variable to be predicted is a continious variable eg scores of student

② **Classification** : the output variable to be predicted is a categorical variable eg classifying incoming emails as spam or ham.

③ **Clustering** : No predefined notion of a label is allocated to groups / clusters formed eg customer segmentation.



(*) Supervised Learning Methods

① Past data with labels is used for building the model.

② Regression and classification algorithms falls under this category.

(*) Unsupervised Learning methods

① No pre-defined labels are assigned to input data

② Clustering algorithms falls under this category.

⇒ Supervised Learning examples

→ supervised learning involves labeled datasets where the outcome or target variable are already known.

① Spam email classification

→ Input: features like email text, sender's address and subject line.

→ output: Labels like 'spam' or 'not spam'

② House Price Prediction

→ Input: features such as bedrooms, location & size.

→ Output: Predicted house price

Unsupervised Learning Examples:

It deals with unlabeled data, finding patterns or structures within data.

① Anomaly Detection

→ Input: Network traffic data, sensor data etc

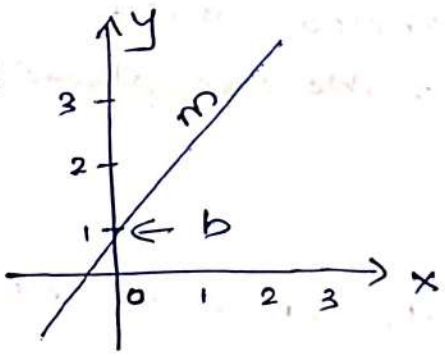
→ Output: Identifying unusual patterns that could indicate security threats.

✳ Regression LINE

First let's understand about the eqvn of a straight line

the eqvn of a straight line is usually written as $y = mx + b$ or $y = mx + c$

What does it stand for?



$y = mx + b$

- slope
- y value, when $x = 0$

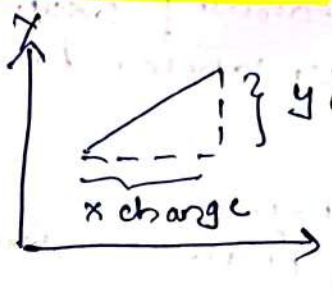
⇒ y = how far up
x = how far along
m = slope or gradient (how steep the line is)
b ⇒ value of y when x = 0

How do we find "m" and "b"?

→ b is easy: just see where the line crosses Y axis.

→ m (the slope) needs some calculation

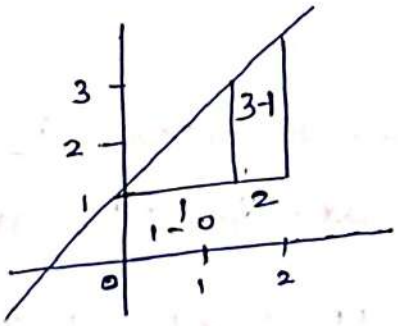
$$m = \frac{\text{change in } Y}{\text{change in } X}$$



Knowing this we can work out the equn of the straight line :

$$m = \frac{3-1}{1-0} = 2/1 = 2$$

$$b = 1$$



⇒ equn of straight line ⇒ $y = mx + c$ ⇒ $y = 2x + 1$

∴ Slope = $m = \frac{\Delta y}{\Delta x} = \frac{y_2 - y_1}{x_2 - x_1}$

Positive Slope

→ A +ve slope means that as x increases, y also increases

→ In a graph a line goes upwards from left to right.

eg : A line passing through points (1, 2) and (2, 4)

$$\Rightarrow m = \frac{4-2}{2-1} = 2/1 = 2$$

Negative Slope

→ A -ve slope means that as x increases, y decreases.

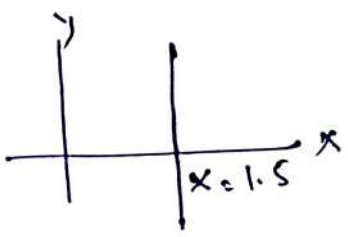
→ In a graph the line goes downwards from left to right

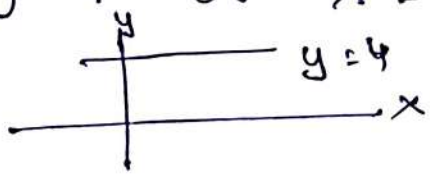
eg : A line passing through points (1, 4) and (2, 2)

$$m = \frac{2-4}{2-1} = \frac{-2}{1} = -2$$

Zero Slope : If the line is horizontal (y doesn't change) as x changes ⇒ the slope is zero

eg $y = 4$ or $x = 1.5$





Regression Line

A regression line is a straight line that best fits the data points on a scatter plot, it represents the relationship b/w the independent variable (x) and dependent variable (y). In statistical terms, it minimizes the sum of squared differences between the actual data points and the predicted data points on the line.

*Equn of a regression line

$$y = mx + b$$

where

y : Dependent variable (Predicted value)

x : Independent variable (input value)

m = slope of the line

$$m = \frac{\Delta y}{\Delta x}$$

b = intercept (the value of y when x = 0)

(Eg1) Predicting House Price

Data : x : Size of the house in square feet

y : Price of the house in $1000s

Size (sq ft, x)	Price (y)
1000	150
1500	200
2000	250
2500	300

$$m = \frac{\Delta y}{\Delta x} = \frac{200-150}{1500-1000} = \frac{50}{500} = \frac{1}{10} = 0.1$$

$$y = 0.1x + 50$$

Interpretion :

→ $m = 0.1$ for every additional savaare foot, the prius increases by $100

→ $b = 50$; A house of 0 saft would hypothetically cost $50000

Best fit line

In regression there is a notation of the best fit line the line which fits the given scatterplot in the best way. So...

the Best-fit line in regression is the line that most accurately represents the relationship blw an independent variable (x) and the dependent var (Y) It is determined using Ordinary Least Savaares (OLS) method, which minimizes the Residual Sum of Savaare (RSS), the sum of savared differences blw the actual and predicted values.

Process of finding the Best fit line

① Start with scatter plot ; usefull to visualize the relationship blw independent & dipendent variable.

2. Define Residuals

Residuals (e_i) represents the error b/w the actual values $\underline{y_i}$ and predicted values $\underline{\hat{y_i}}$:

$$e_i = y_i - \hat{y_i}$$

the smaller the residuals, the better the fit of the line.

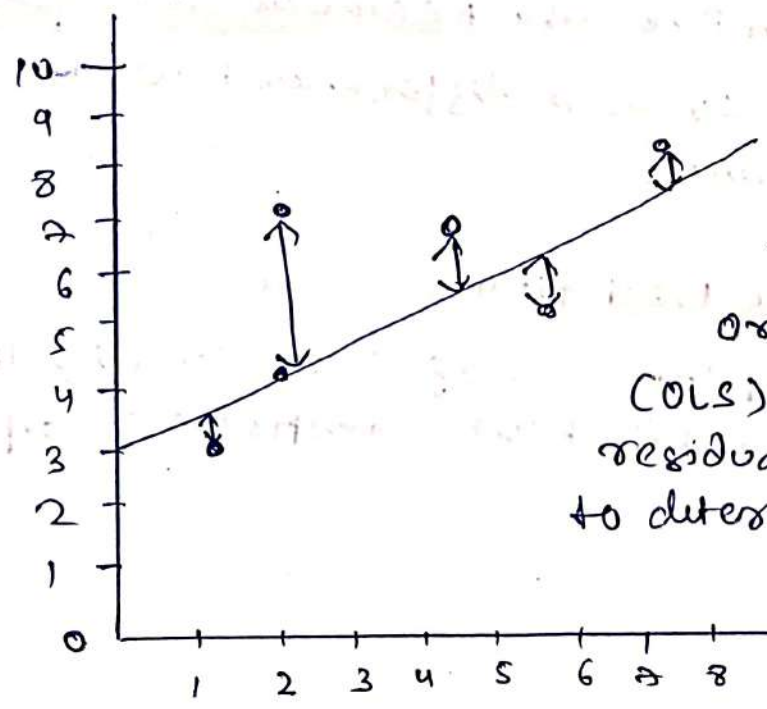
3. Residual Sum of Squares (RSS)

to evaluate how well a line fits the data calculate the RSS :

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

the goal is to find the line that minimizess RSS, the method for achieving the Ordinary Least Squares.

4. Ordinary Least Squares (OLS)



In this Img we see the concept of residuals and how the Ordinary least Squares (OLS) method minimized the residual sum of Squares to determine the best fit line

Key elements from the image:

① Eqn of the regression line:

$$y = B_0 + B_1 X$$

→ B_0 : Intercept (value of y when x = 0)

→ B_1 : Slope (change in y for one-unit change is X).

② Residuals (e_i)

→ the difference b/w the actual values (Y_i) and the predicted value ($\hat{Y}_i$)

$$e_i = Y_i - \hat{Y}_i$$

→ Residuals are shown as vertical lines b/w the observed points (dots) and the regression line.

③ Residual Sum of Squares (RSS)

→ the RSS formula calculates the total squared error:

$$RSS = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} (y_i - (B_0 + B_1 x_i))^2$$

$$\Rightarrow \sum_{i=1}^{n} (y_i - B_0 - B_1 x_i)^2$$

④ Goal of OLS:

Minimize the RSS to find the optimal values of B_0 and B_1, ensuring the regression line best fits the data.

Gradient Descent is an optimization algorithm used to find the values of the parameters (coefficients) of a function (f) that minimizes a given cost function (cost)

Unconstrained and Constrained Minimisation

es "Constrained Minimization" is a process in which we try to optimise an objective function with respect to some variables in the presence of the constraints on the variables or the function.

↳ In our case the objective func is a cost function.

ee "Unconstrained Minimization" on the other hand is a process in which we try to optimise the objective func with respect to some variables without any constraints on those variables.

Let's study them in Detail

Unconstrained minimization is an essential concept in linear regression and optimization where we aim to find the best fit line for a given dataset by minimizing the cost func usually the Mean Squared Error or MSE

In linear regression, we model the relation b/w the input features (x) and the target (y) as

$$\hat{y} = wx + b$$

where
$w \rightarrow$ weight or slope
$b \rightarrow$ bias (intercept)
$\hat{y} \Rightarrow$ predicted value

Goal: to find w and b that minimize the diff b/w the predicted values ($\hat{y}$) and actual value (y)

Objective: Minimize the cost func

the cost func measures how well the line fits the data, for linear regression we often use Mean Squared Error (MSE)

$$J(w,b) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \underbrace{(wx_i + b)}_{\hat{y}})^2$$

(J(w,b): cost func; depends on weight & bias)

Here:
$N \Rightarrow$ Nos of data points
$y_i \Rightarrow$ is the actual target value
$x_i \Rightarrow$ is the input feature value
$wx_i + b$ is the predicted value

In Unconstrained minimization, we aim to minimize $J(w,b)$ without any constraints on w or b this process involves

① Computing gradients : Derivatives of $J(w,b)$ with respect to w and b

② Setting gradients to zero : Solve for w and b where the gradients vanish (Stationary point)

③ Finding the minimum : Check if stationary point corresponds to minimum.

Simple Example

Dataset

X	y
1	2
2	3
3	4

Step1 : Write the cost function

$$J(w,b) = \frac{1}{3}\sum_{i=1}^{3}(y_i - (wx_i + b))^2$$

Step2 : expand the cost function

$$J(w,b) = \frac{1}{3}[(2-(w\cdot 1+b))^2 + (3-(w\cdot 2+b))^2 + (4-(w\cdot 3+b))^2]$$

the cost func $J(w,b)$ is the Mean Squared error

$$J(w,b) = \frac{1}{N}\sum_{i=1}^{N}(y_i - (wx_i + b))^2$$

to minimize $J(w,b)$ we need to compute its gradient ie the partial derivatives with respect to w and b.

* Gradient w.r.to w

the partial derivatives of $J(w,b)$ w.r.to w measures how $J(w,b)$ changes as the w changes

$$\frac{\partial J}{\partial w} = \frac{\partial}{\partial w}\left[\frac{1}{N}\sum_{i=1}^{N}(y_i-(wx_i+b))^2\right]$$

Bring the derivative inside the summation

$$\frac{\partial J}{\partial w} = \frac{1}{N}\sum_{i=1}^{N}\frac{\partial}{\partial w}(y_i-(wx_i+b))^2.$$

Use the chain rule for $(y_i-(wx_i+b))^2$, let $v = y_i-(wx_i+b)$ then

$$\frac{\partial}{\partial w}v^2 = 2v\frac{\partial v}{\partial w}$$

$$\Rightarrow \frac{\partial}{\partial w}(y_i-(wx_i+b))^2 = 2\cdot(y_i-(wx_i+b))\cdot(-x_i)$$

(Note: the derivative of $y_i-(wx_i+b)$ w.r.to w is $-x_i$)

$$\frac{\partial J}{\partial w} = \frac{1}{N}\sum_{i=1}^{N}[2\cdot(y_i-(wx_i+b))\cdot(-x_i)]$$

$$\frac{\partial J}{\partial w} = -\frac{2}{N}\sum_{i=1}^{N}x_i(y_i-(wx_i+b))$$

Gradient w.r.to b

$$\frac{\partial J}{\partial b} = \frac{\partial}{\partial b}\left[\frac{1}{N}\sum_{i=1}^{N}(y_i - (wx_i + b))^2\right]$$

Bring the derivative inside summation

$$\frac{\partial J}{\partial b} = \frac{1}{N}\sum_{i=1}^{N}\frac{\partial}{\partial b}(y_i - (wx_i + b))^2$$

using the chain rule

$$\Rightarrow 2\cdot(y_i - (wx_i + b))\cdot\frac{\partial}{\partial b}(y_i - (wx_i + b))$$

$$\frac{\partial}{\partial b}(y_i - (wx_i + b)) \Rightarrow -1$$

$$\Rightarrow \frac{\partial J}{\partial b} = \frac{1}{N}\sum_{i=1}^{N}[2\cdot(y_i - (wx_i + b))(-1)]$$

$$\Rightarrow -\frac{2}{N}\sum_{i=1}^{N}(y_i - (wx_i + b))$$

Setting $\frac{\partial J}{\partial w} = 0$ and $\frac{\partial J}{\partial b} = 0$ to get the optimal values of w, b

$$\Rightarrow w = 1, \quad b = 1$$

$$\Rightarrow \boxed{\hat{y} = x + 1}$$

Why unconstrained?

→ there are no restrictions on values of w and b can take.

⊛ What is Constrained Minimization?

In constrained minimization, we still aim to find the best values of the parameters (like w and b in regression) that minimize the error.

However we add rules or constraints to ensure the solution satisfies certain conditions.

eg:

Imagine you are choosing a path to minimize the travel time but are required to use certain roads only, this restriction is the constraint.

for regression, a constraint could be something like:

⇒ Keep the model's weight w small (to prevent overfitting),

⇒ Ensure the sum of weights equals a fixed value.

We know that the eqn is $y = wx + b$

where

y ⇒ predicted value

x ⇒ input feature

w ⇒ its the weight (slope)

b ⇒ bias (intercept)

the goal is to minimize the error:

Error (Cost function):

$$J(w,b) = \frac{1}{N}\sum_{i=1}^{N}(y_i - (wx_i + b))^2$$

this is called as unconstrained minimization, where we freely adjust w and b to minimize error.

Constrained Minimization Example:

let's add a constraint:

" The weight w must not exceed a certain value (eq $w^2 \leq \lambda$, where λ is a small no)"

Why add constraints?

① Prevent Overfitting: if w becomes very high, the model can overfit the training data, meaning it wont generalize well to new data.

② Real-world requirements: Sometimes weights must meet specific physical or financial limits.

How to solve Constrained Minimization?

to solve the constrained minimization problems, we slightly modify how we minimize the cost func,

Simple example:

Suppose we have:

→ cost function: $J(w) = (y - wx)^2$

constraint: $w \leq 2$

① Without constraint

If we ignore the constraint, we minimize $J(w)$ by taking the derivative and finding where it equals 0 let's say this gives us

$$\boxed{w = 5}$$

But $\boxed{w = 5}$ violates the constraint $w \leq 2$

② With constraint

Instead of freely choosing w, we must respect the constraint, in this case $\boxed{w = 2}$ ⇒ final soln

why? because $w = 2$ is the largest value that satisfies $w \leq 2$ and going beyond it would break the rule.

Common Ways to Handle Constraints

① Hard Constraint (as in the above eq)

forcing the soln to strictly obey the rule
eg $w \leq 2$ so the final weight is capped at 2.

② Soft Constraint (Regularization)

Instead of forcing a strict rule, we penalize large weights for eg modify the cost func to

$$J(w,b) = \frac{1}{N}\sum_{i=1}^{N}(y_i - (wx_i+b))^2 + \alpha w^2$$

Here α is a small no that controls how much we penalize large weights.

⋆ Optimal Weight formula Derivation

OPT

Key points:

① Objective: Minimize the cost func to find the best weight w.

$$\text{Cost func}: J(w) = \frac{1}{N}\sum_{i=1}^{N}(y_i - wx_i)^2$$

bias : 0

this measures how far the predicted values wx_i are from the actual values y_i

Step2 : Expanding the cost function

→ Expand $(y_i - wx_i)^2$

$$J(w) = \frac{1}{N} \sum_{i=1}^{N} (y_i^2 - 2wx_i y_i + w^2 x_i^2)$$

→ Separate the terms

$$J(w) = \frac{1}{N} \left[\sum_{i=1}^{N} y_i^2 - 2w \sum_{j=1}^{N} x_i y_i + w^2 \sum_{j=1}^{N} x_i^2 \right]$$

Step3 : differentiating with respect to w:
take the derivative to minimize J(w):

$$\frac{\partial J(w)}{\partial w} = \frac{1}{N} \left[-2 \sum_{j=1}^{N} x_i y_i + 2w \sum_{i=1}^{N} x_i^2 \right]$$

Simplifying the term

$$\frac{\partial J(w)}{\partial w} = \frac{-2}{N} \sum_{i=1}^{N} x_i y_i + \frac{2w}{N} \sum_{i=1}^{N} x_i^2$$

Step4 : setting the derivative to 0, ie to find the minimum set $\frac{\partial J(w)}{\partial w} = 0$

$$- \sum_{j=1}^{N} x_i y_i + w \sum_{i=1}^{N} x_i^2 = 0$$

$$w \sum_{j=1}^{N} x_i^2 = \sum_{i=1}^{N} x_i y_i \Rightarrow w = \frac{\sum_{i=1}^{N} x_i y_i}{\sum_{i=1}^{N} x_i^2}$$

Final formula : $w = \frac{\Sigma(x \cdot y)}{\Sigma x^2}$

Interpretation

⇒ $\Sigma(x \cdot y)$ ⇒ Measures how x and y vary together

$\Sigma(x^2)$ ⇒ Measures the total spread of x

(eg) Given $x = [1, 2, 3]$, $y = [2, 4, 6]$

Compute $\Sigma(x \cdot y) = 2 + 8 + 18 = 28$

Compute $\Sigma x^2 = 1^2 + 2^2 + 3^2 = 1 + 4 + 9 = 14$

$w = 28/14 = 2$ ⇒ optimal line ⇒ $y = 2x$

Note : the value of RSS would change because the units are changing because :

the RSS for any regression line is given by the expression $\Sigma(y_i - y_{i\,pred})^2$, RSS is the sum of squared difference b/w the actual value and the predicted value and it's value will change if the units change since it has units of y^2

for eq (140 rss − 70 rss)^2 = 4900 rss whereas (2 USD − 1 USD)^2 = 1 so value of RSS is diff in both cases.

(*) TSS (Total Sum of Squares)

the total sum of squares is a way to measure how much the data values vary in the dataset

It helps us to understand the total amount of spread or variability in the dependent variable (y), which is the variable you are trying to predict or explain.

Why is TSS important?

When you build a regression model, the goal is to explain or predict this variability, TSS gives us a starting point to see how much total variation exists before any model is applied.

Breaking It Down:

Imagine you have some data points representing the exam scores

$$y = [50, 60, 70, 80, 90]$$

step 1 : find the mean ($\bar{y}$) : the mean is the avg of all the data points :

$$\bar{y} = \frac{\text{Sum of all } y}{\text{total number of } y} \Rightarrow \bar{y} = \frac{50+60+70+80+90}{5}$$

$$\Rightarrow \bar{y} = 70$$

Step 2 : Calculate how far each value is from the mean

Subtract the mean from each data point

$$(y_i - \bar{y})$$

for each y :

$[50-70,\ 60-70,\ 70-70,\ 80-70,\ 90-70]$

$\Rightarrow [-20, -10, 0, 10, 20]$

Step3 : Square each difference : squaring removes -ve signs and gives more weight to larger differences:

$[-20^2, -10^2, 0^2, 10^2, 20^2] \Rightarrow [400, 100, 0, 100, 400]$

Step 4 : Add them up : Add these squared differences

$TSS = 400+100+0+100+400 = 1000$

$$TSS = \sum_{i=1}^{N} (y_i - \bar{y})^2$$

$y_i \Rightarrow$ Actual Data values

$\bar{y} \Rightarrow$ Mean of the data

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

$\Rightarrow \hat{y}_i$ Predicted values from the regression model.

R-Squared (R^2) :

R^2 is a measure of how well the regression line explains the variability in the dependent variable y; It is calculated as :

$$R^2 = 1 - \frac{RSS}{TSS}$$

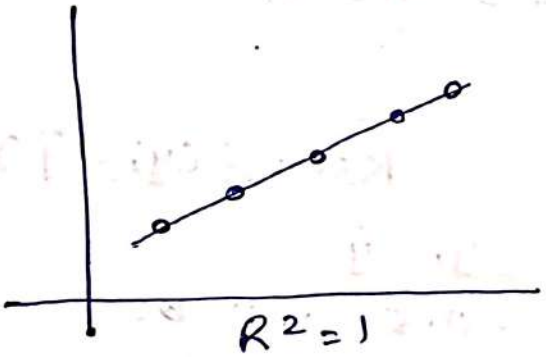
R squared = $(MSS/TSS) = 1 - (RSS/TSS)$

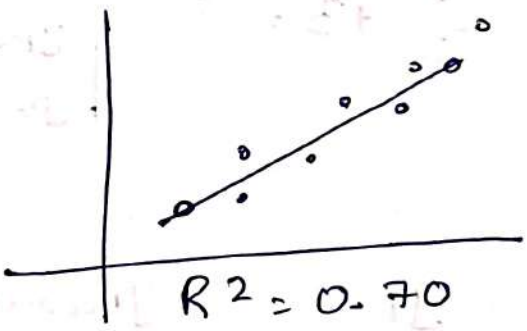
R^2 ⇒ Proportion of the total variability explained by the model.

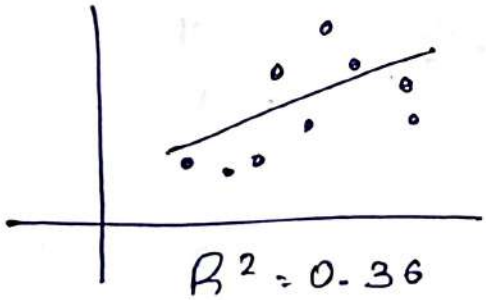
RSS/TSS : Proportion of variability not explained by the model.

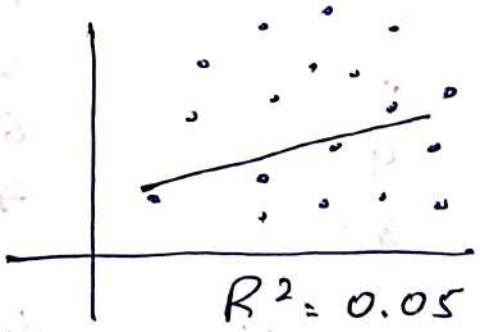
If $R^2 = 0.6$, it means the model explains 60% of the variability on the data, while remaining 40% is unexplained.

Physical Significance Of R^2



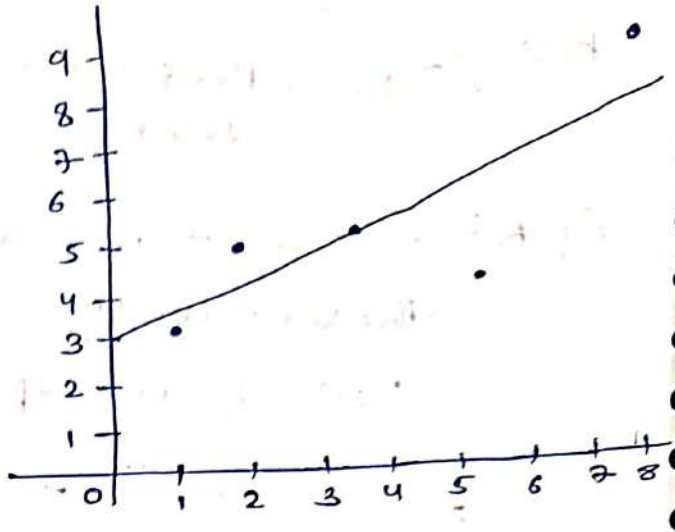






Q Given the equn of line

$$y = \frac{x}{2} + 3$$



Q1 to find the value of Rss

$y_{Pred1} = \frac{1}{2} + 3 = 3.5$ at $x = 1$

$y_1 = 3$

$y_{Pred2} = \frac{2}{2} + 3 = 4$

$y_2 = 5$

$y_{Pred4} = \frac{4}{2} + 3 = 5$

$y_4 = 5$

$y_{Pred6} = \frac{6}{2} + 3 = 6$

$y_6 = 4$

$y_{Pred8} = \frac{8}{2} + 3 = 4 + 3 = 7$

$y_8 = 8$

$$Rss = \Sigma (y_i - \hat{y})^2$$

⇒ table

y_i	$y_{Predicted\ i}$	$y_i - \hat{y}$
3	3.5	$-0.5 \Rightarrow 0.25$
5	4	$1 \Rightarrow 1$
5	5	$0 = 0$
4	6	$-2 = 4$
8	7	$1 = 1$
Rss	6.025	

Q2 to find TSS

y_i	$\bar{y}$
3	5
5	5
5	5
4	5
8	5

$\bar{y} \Rightarrow \frac{3+5+5+4+8}{5} = \frac{25}{5} = 5$

$TSS = \Sigma (y_i - \bar{y})^2 = (4 + 0 + 0 + 1 + 9)$

14

Q find $R^2 = 1 - \frac{RSS}{TSS} = 1 - \frac{6.25}{14} = \frac{14 - 6.25}{14} = \frac{7.75}{14}$

(#) Apart from R^2 there is one more quantity named RSE (Residual Square Error)

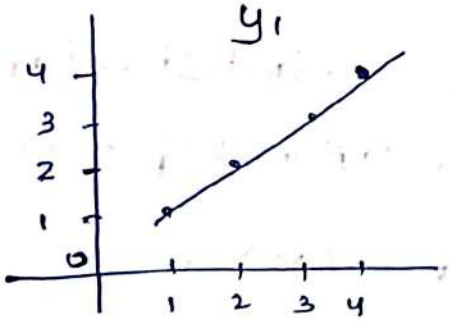
$RSE = \sqrt{\frac{RSS}{df}}$ Here $df = n - 2$ where n = nos of data points

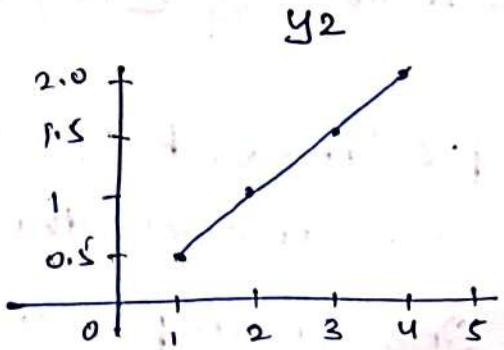
Note: the value of correlation coefficient always lies b/w -1 and +1, where a negative value implies a negative correlation and a +ve value shows a positive correlation and 0 means no correlation.

∴ $R^2 = 1 - \frac{RSS}{TSS}$ ⇒ the value of R^2 lies b/w 0 and 1 where 1 implies that the variance in the data is being explained by the model

0 means none of the variance values is being explained by the model, obviously its difficult to achieve either of the extremes.

Note: following two straight lines fitted for two set of datapoints





Q value of correlation coefficient for both 1's 1 and 1 these slopes are 1 and 0.5 but ∵ all the points lie on the straight line correlation coefficient becomes equal to 1 ∴ Slope of straight line don't represent correlation.

Q Suppose that we run a linear regression on the data where X → Independent var and Y is the target var, you find that the correlation b/w y and x is -0.92, what can you say here?

Ans the absolute value of the correlated coefficient is very high, -ve sign just implies that x and y are -voly correlated, Hence they have very strong -ve correlation.

Q The independent var X from a linear regression is measured in miles, if we convert it to kms keeping the unit of the dependent var Y the same, how will the slope coefficient change?

Note 1 mile = 1.61 km

Ans In the linear regression eqn, x gets multiplied by 1.6 with no change in y So it means slope will be divided by 1.6.

Summary of Simple linear Regression

Simple Linear Regression is a statistical method used to model the linear relationship between:

→ One Independent variable (X) and
→ One dependent variable (Y)

the equation of a simple linear regression model is:

$$Y = B_0 + B_1 X + \epsilon$$

Where:

y ⇒ Dependent Variable (Predicted value)
x ⇒ Independent variable (Predictor)
B_0 ⇒ Intercept (value of y when $x=0$)
B_1 ⇒ Slope (rate of change of y for a 1unit change in x)
ϵ ⇒ Error term (Unexplained variability)

② Assumptions of Simple Linear Regression

for the model to be valid, the following assumptions must hold:

① Linearity: the relationship b/w x and y is linear
② Independence: Observations are Independent of each other.
③ Homoscedasticity: the variance of the residuals (errors) in constant across all values of x.
④ Normality of Residuals: Residuals are normally distributed.

③ Interpretation of Coefficients

① Intercept (B_0): the predicted value of y when $x=0$
② Slope (B_1): the change in y for one unit Increase in x

4. Goodness of fit (R^2)

⇒ R^2 (Coefficient of Determination) measures the proportion of variation in Y (dependent) Explained by X (Independent)

⇒ Value Ranges from 0 to 1

⇒ $R^2 = 0$ (X explains none of the variability in Y)

⇒ $R^2 = 1$ (X explains all the variability in Y)

Higher R^2 indicates better fit.

5. Correlation Vs Regression

Correlation measures the strength and direction of the linear relationship b/w X and Y, range (-1 to +1)

Regression quantifies the exact relationship (eg slope and intercept) and allows for prediction.

6. Residual Analysis

→ Residuals ($Y_{actual} - Y_{Predicted}$) represent the Error in predictions.

→ A good regression model will have Residuals centered around 0

But if 0 can lead to overfit

7. Limitations

→ Sensitive to outliers

→ Only models linear relationships (non linear relationship requires other methods)

✳ Simple Linear Regression In Python

#Assumptions of Simple Linear Regression

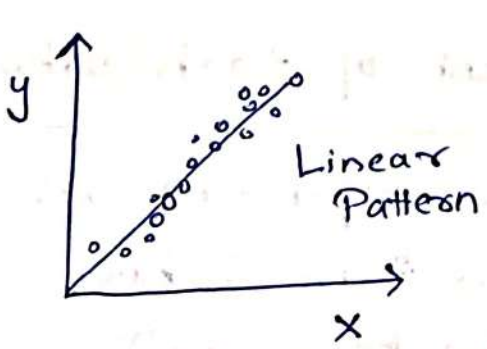
You are making conclusions on the "population" using a "sample", the assumptions that variables are linearly dependent is not enough to generalise the results you obtain on a sample to the population which is much larger in size than the sample.

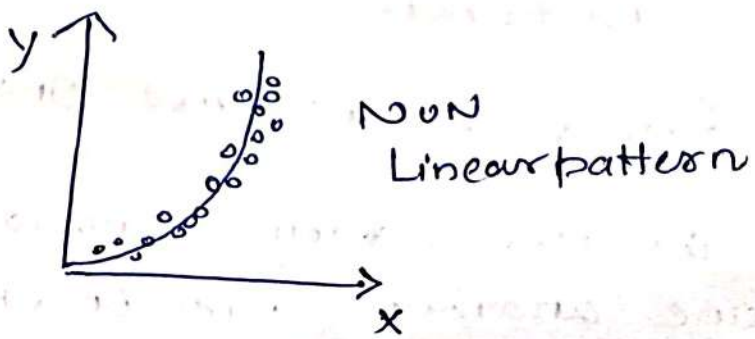
thus we need to have certain assumptions in place in order to make conclusions.

Let's understand the importance of each assumption one by one:

* There is a linear relationship b/w X and Y

① X and Y should display some sort of linear relationship otherwise there is no use of fitting a linear model b/w them.





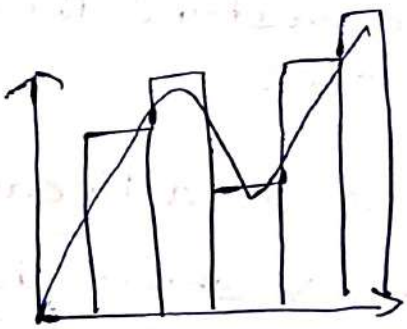
*Error terms are normally distributed with mean zero (not x, y)

→ There is no problem if the errors terms are not normally distributed if you just wish to fit a line and not make any further interpretations

→ But if we are willing to make some conclusions on the model that we build then we need to have a notion of the distribution of the error terms

→ the assumption of normality is made, as it has been observed that the error terms generally follow a normal distribution with the mean equal to zero in most cases.

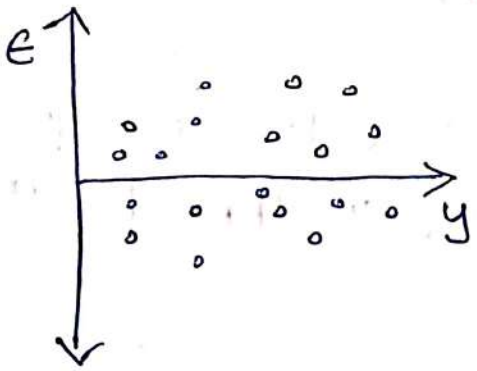
Error terms Normally Distributed



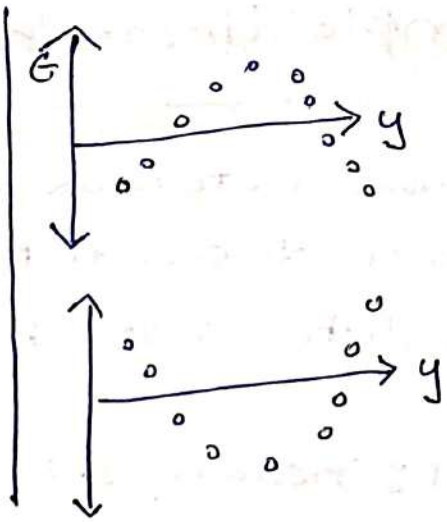
Error terms not normally distributed

*Error terms are independent of each other

the error terms should not be dependent on one another (like in time series data where the next value is dependent on the prev one)



No visible patterns
Error terms are
Independent

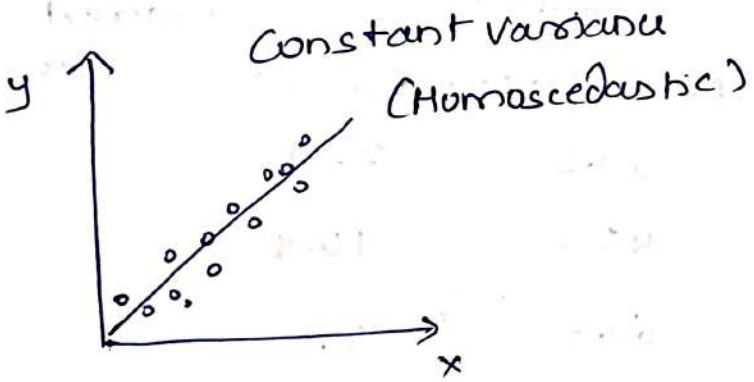


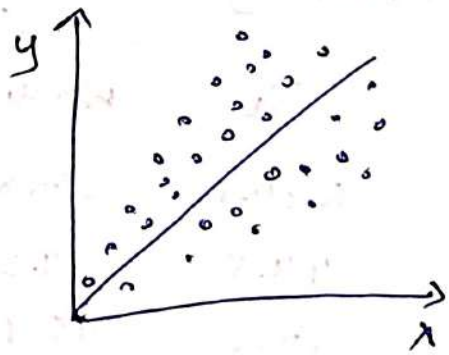
Visible pattern - Error terms are dependent

✳ Error terms have constant variance (homoscedasticity)

→ the variance should not increase (or decrease) as the error value changes

→ Also the variance should not follow any pattern as the error terms change.





changing variance
(Heteroscedastic)

Simple Linear Regression

In this notebook we will build a linear regression model to predict Sales using an appropriate Predictor variable.

Step1 : Reading & Understanding the Data

Let's start with the following steps :

① Import data using Pandas library

② Understanding the structure of the data

```
> import numpy as np
> import pandas as pd
# reading the dataset
> advertising = Pd.read_csv("advertising.csv")
> advertising.head()
```

Predictive variables → Tv, Radio, Newspaper

Sales → target var

	Tv	Radio	Newspaper	Sales
	230.1	37.8	69.2	22.1
	44.5	39.3	45.1	10.4
	17.2	45.9	69.3	12.0
	151.5	41.3	58.5	16.5
	180.8	10.8	58.4	17.9

Here each row represents a diff market

Goal is to build the regression model which Predict sales using cols Tv, Radio, Newspaper

Checking the shape of dataset

> advertising.shape

↳ (200, 4)

checking for the missing values

> advertising.info()

↳ All 4 cols have 200 non null float64 values

Checking for statistics value

> advertising.describe()

↳ It will show mean, median values

Visualise the dataset

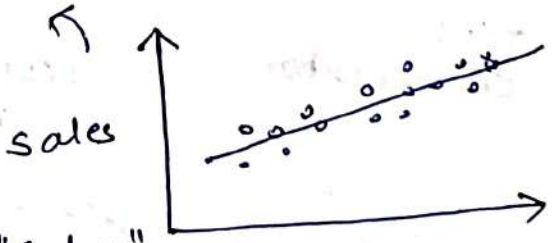
> import matplotlib.pyplot as plt

> import seaborn as sns

(#) Comparing "Tv" & "Sales" columns

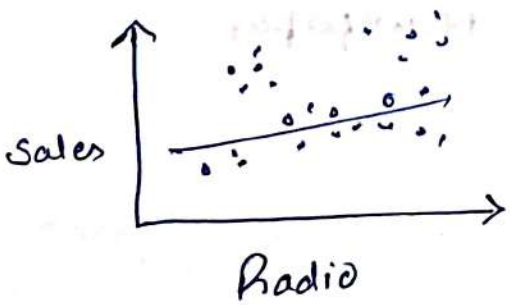
> sns.regplot(X="TV", y="Sales", data=advertising)

Good Relation



(#) Similarly we can Plot

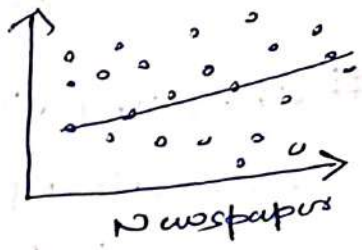
> sns.regplot(X="Radio", y="Sales", data = advertising)



∴ the data is scattered hence Not to smooth Relation b/w the two as compared to Tv Vs Sales

Similarly for Newspaper vs Sales column

```
sns.regplot(x="Newspaper", y="Sales",
            data = advertising)
```

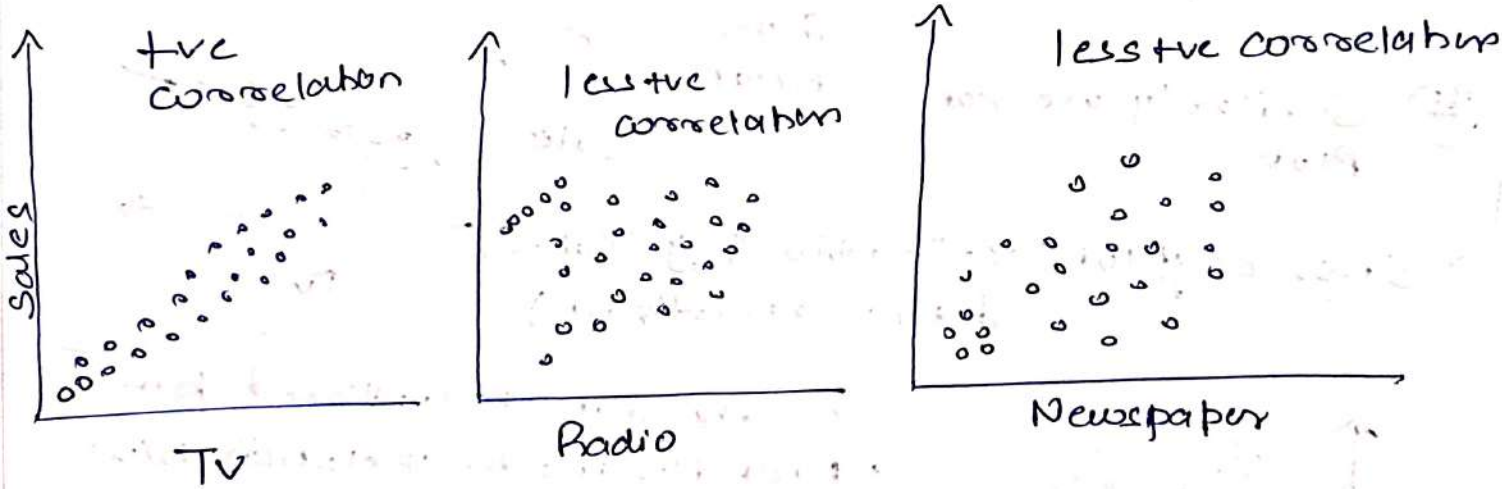


Data scattered around the sales line → Bad relation b/w the two.

For making a good comparison b/w the plots

↳ Pairplots

```
> sns.pairplot(data = advertising, x_vars =
              ['Tv', 'Radio', 'Newspaper'],
              y_vars = 'Sales')
```



Another way → heatmap

Using the heatmap we want to plot the correlation

> advertising.corr()

Corr Matrix

	Tv	Radio	Newspaper	sales
Tv	1.00	0.054	0.056	0.901
Radio	0.05	1.00	0.354	0.349
Newspaper	0.05	0.354	1.000	0.157
Sales	0.901	0.349	0.157	1.000

> sns.heatmap (advertising.corr(), annot = True)

So we read the data and visualized it using seaborn, we also looked at the correlation b/w the target variables 'sales' and diff predictor variables

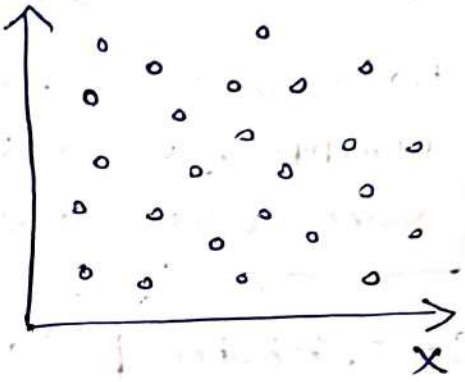
Hypothesis testing in Linear Model

When we fit the straight line through the data, we will obviously get the two parameters of the straight line ie the Intercept (B_0) and the slope (B_1)

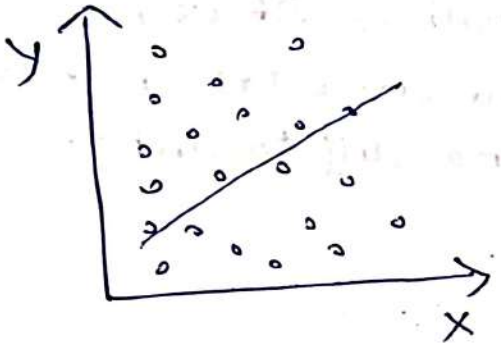
Now while B_0 is not of much importance right now, but there are few aspects surrounding B_1 which is needed to be checked and verified.

the first ques we ask is " Is the beta coeffcien significant? " what does this mean?

⇒ Suppose we have a dataset for which the scatterplot looks like the following



Now if we run a linear regression on this dataset in Python, it will fit a line on the data which say looks like the following:



Now, we can clearly see that the data is randomly scattered and doesn't follow any linear trend or any trend in general

But Python will anyway fit a line through the data using the least squared method, but we can see that the fitted line is of no use here

Hence everytime we perform a linear regression we need to test whether the fitted line is a significant one or not or to simply put it we need to test whether B_1 is significant or not

cc

Here comes the idea of Hypothesis testing on B_1 ,,

We start by saying that B_1 is not significant ie there is no relationship b/w X and Y.

So in order to perform the hypothesis test, we first propose the null hypothesis that B_1 is 0 and the alternate hypothesis that B_1 is not zero.

- Null Hypothesis (H_0) : $B_1 = 0$
- Alternate Hypothesis (H_A) : $B_1 \neq 0$

if we fail to reject the null hypothesis that would mean that B_1 is zero which would simply mean that B_1 is insignificant and of no use in the model.

Similarly if we reject the null hypothesis B_1 is not zero & the line fitted is a significant one.

How do we perform the hypothesis test?

↳ first we need to compute t-score which is similar to the Z score

$$\text{Z score} = \frac{X - \mu}{S/\sqrt{n}}$$

or

t score

where μ → Population mean

S → Sample standard deviation

$S/\sqrt{n}$ → Standard error

Using this, the t-score for $\hat{B}_1$ comes out to be (Since the null hypothesis is that B_1 is equal to 0)

$$\frac{\hat{B}_1 - 0}{SE(\hat{B}_1)}$$

Now, In order to perform the hypothesis test, we need to derive the P value for the given Beta,

Let's see how Pvalue is calculated?

→ Calculate the value of t-score for the mean point (in this case 0, according to the null hypothesis that we have started) on the distribution.

→ Calculate the P-value from the cumulative prob for the given t score using t table.

→ Make the decision on the basis of the pvalue w.r.to the given value of B (significance level)

Now If Pvalue < 0.05 → reject null hypothesis and state that B_1 is indeed significant.

Q Suppose for a linear model we got B_1 as 0.5 and Standard error of B_1 was found out to be 0.02 Hence t score?

$$t\text{ score} = \frac{\hat{B}_1 - 0}{SE(\hat{B}_1)} = \frac{0.50}{0.02} = \frac{50}{2} = 25$$

Since "Tv" is very strongly correlated to "Sales" let's first build a simple linear regression model with "Tv" as the predictor variable.

Building a Linear Model

Building a linear model for Tv as the predictor and sales as the dependent value.

Step2 : Performing Simple Linear Regression

Eqn of linear regression

$$y = c + m_1 x_1 + m_2 x_2 + \ldots + m_n x_n$$

↳ y ⇒ is the response var
c ⇒ is the intercept
m_1 ⇒ is the coefficient for the first feature
m_n ⇒ is the coefficient for the nth feature

In our case :

$$y = c + m_1 \times (TV)$$ ← column

the m values are called the model coefficients or model parameters

library being used "statsmodels"

```
> import statsmodels
> import statsmodels.api as sm
> import sklearn
```

We can build the model using "statsmodels" or "Sklearn"

Steps in Model Building

→ Create X and Y

→ Create train and test sets (70:30 or 80:20) ratio

→ Train model on the training set (i.e to learn the coefficients)

→ Evaluate the model (training set, test set)

Creating X and Y

> X = advertising['TV']

> Y = advertising['sales']

train - test split

↳ Sklearn comes with a built in method for train-test split

> from sklearn.model_selection import train_test_split

> X_train, X_test, Y_train, Y_test = train_test_split(X, Y, train_size = 0.70, random_state = 100)

train_size = 70% test_size = 30%

random_state → 100 (reshuffel data)

> X_train.shape

↳ (140,)

↳ X_test.shape

↳ (60, 1)

So values got split into train test

(#) training the Model (Statsmodel)

So when we are using the statsmodel library for creating a model, there is a catch there and the catch is by default the statsmodel library doesn't include "C" constant term.

ee So to add the constant term explicitly we need to tell the library about it ,,

```
> X_train.head(3)
     74      213.4
      3      151.5
    185      205.0
```

We will create the new version of the Xtrain for the statsmodel library

```
> X_train_sm = sm.add_constant(X_train)
```

```
> X_train_sm.head(3)
      const    TV
  1.0   1.0   213.4
  :     1.0   151.5
        1.0   205.0
```

} It added a col called as const with all the values as 1.0 for the constant part

So our eqvn became

$y = C + m_1 \cdot x_1$ (Original)

$y = C.\text{const} + m_1 \cdot TV_{\text{col}}$

So now our training set looks like

> X_train_sm.head(3)

↳

	const	TV
	1.0	213.4
	1.0	151.5
	1.0	205.0

fitting the model (ordinary least squares)

> lr = sm.OLS(y_train, X_train_sm)

↳ It creates a linear regression object

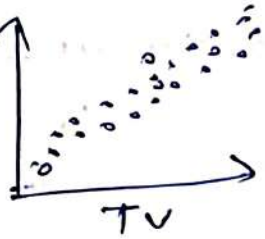
> lr_model = lr.fit()

> lr_model.params # to see Model Parameters

↳ const 6.94 } Coefficients
TV 0.05

↳ # Sales = 6.94 + 0.05 × TV

Coefficient of TV ⇒ +ve



We saw that Sales ↑ as TV ↑

also const is 6.94 ⇒ It means

if we spend 0$ on TV ⇒ Sales will be

6.94

other details

> lr_model.summary()

↳ Only given by statsmodels

In Short

After we import the statsmodel.api we create a simple linear regression model in just few steps

```
ee import statsmodels.api as sm
X_train_sm = sm.add_constant(X_train)
lr = sm.OLS(y_train, X_train_sm)
lr_model = lr.fit()  99
```

Here OLS stands for ordinary least savuares, which is the method that statsmodels use to fit the line, we use command add_constant so that statsmodels also fits an intercept, if we don't use this it will ignore the const term.

```
> lr_model.summary()
```

↳ It has info like R^2 value, p value

Here the top 3 things that we referred here

① Coefficients and Pvalue

② R-savuared is 81.6% very high

③ PCF-statistic) is low ⇒ the fit is not by chance

In summary statics values

R-savuared : 0.816

F-statistic : 611.2

Prob (F statistic) : 1.52e-52

Coef std error & P>|t|

↳ for cols Tv and Const

F-Statistic and Model Significance:

The f-statistic tests whether the overall regression model is significant, unlike testing the individual coefficients ($B_1, B_2, \ldots$) for their significance, the f-statistic tests if the combination of all predictors explain the variability in dependent variable significantly.

① Purpose of f-statistic
- → Determines if the model as a whole fits better than a model with no predictors (i.e all coefficients = 0)
- → Ensures that the model's overall significance isn't just due to chance.

② Prob (f-statistic)
- → this is the Pvalue associated with f statistic
- → if Pvalue < 0.05 ⇒ the model is significant
- if Pvalue > 0.05 ⇒ Model might have fit the data purely by chance.

In the eg P-value is 1.52e-52 (Practically 0)
- ↳ Means model highly significant

Why its useful?
- ↳ Particularly useful in multiple linear regression, where there are multiple predictors, the f-statistic helps determine if all the predictor combined are significant

⊛ R-squared

→ Measures how much variance in the dependent variable is explained by the model.

eg An Rsquared value is 0.86 means model explains 86% of the variance, which is good.

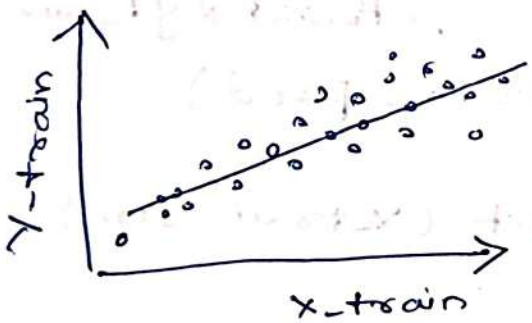
⊛ Now we have built the Model but the Model evaluation is still left

Plotting the line

```
> plt.scatter(X_train, y_train)
  plt.plot(X_train, 6.948 + 0.054 * X_train, 'r')
```

(6.948 + 0.054 * X_train → Predicted value)

Coefficients taken from summary()



Model Building Process

Residual Analysis and Predictions

Recall that one of the assumptions that you studied was the error terms should be normally distributed with mean equal to 0, so once we have built the model, you would need to verify if the model is not violating the assumption.

for this we can plot a "histogram of the error terms" to check whether they are normally distributed.

(#) So the steps that were involved in the exercise end to end were

Step1 : Reading & Understanding the Data

Step2 : Training the model

Step3 : Residual Analysis (specific to linear regression)

A fundamental assumption here is that these residuals should be normally distributed.

Step4: Predicting and evaluating on the test set

Step3: Residual Analysis

error = func (y_train, y_train_pred) ← Predicted y train

> y_train_pred = lr_model.predict (X_train_sm)

↳ we got the predicted values

(#) Now we can do error analysis

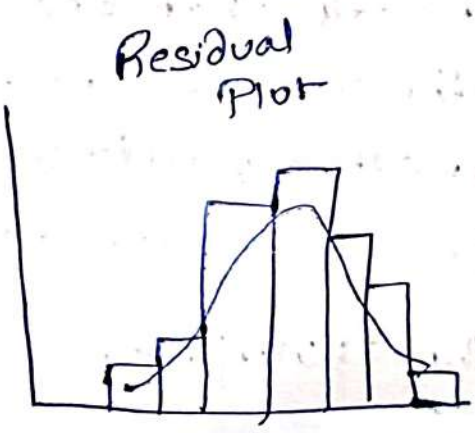
> res = y_train - y_train_pred

Plot the residuals (Histogram

> sns.distplot(res)

> plt.title ("ResidualPlot")

#Normally Distributed



So In the first case we saw that the errors are normally distributed.

2nd thing is we look for patterns in residuals (we shouldn't be able to identify any patterns)

```
> plt.scatter(X_train, res)
  plt.show()
```

(No Patterns, or clusters)

Conclusion

⇒ Hence the residuals are normally distributed and we couldn't identify any patterns here. So this model fit looks good.

Step 4 : Predictions and Evaluation of test_set

↳ # Predictions on the test_set (y_test_pred)

evaluate the model, r-squared on the test

```
>
>
# y_test_pred ke liye we need to add the const term
> X_test_sm = sm.add_constant(X_test)
> y_test_pred = lr_model.predict(X_test_sm)
# Now we will evaluate the model using Sklearn
> from sklearn.metrics import mean_squared_error
> from sklearn.metrics import r2_score
```

Motivation - When One variable isn't enough ⇒ Multiple Linear Regression

---

The term "multiple" in multiple linear regression gives us a fair idea in itself, it represents the relationship b/w two or more independent input variables and a response variable.

Multiple Linear Regression is needed when one variable might not be sufficient to create a good model and make accurate predictions

We know that R squared measures how well the independent variables (like TV, Radio, Newspaper) explain the variance of the dependent variable

→ Range : 0 to 1 (or 0% to 100%)

Higher values of R^2 indicate that larger proportion of the variance in the dependent variable is explained by the independent variables.

What happens when we add more variables?

① simple linear regression (1 variable : TV)

→ R-squared = 0.816

this means 81.6% of the variability in the dependent variable (sales) is explained by TV alone

② Multiple Linear Regression (2 variables : Tv + newspaper)

→ R-squared = 0.836

Now by adding "newspaper" as a predictor, the model explains extra 2% of the variance.

→ So the new variable may not be as strong as Tv, but it adds some predictive power.

③ Multiple Linear Regression (2 variables : Tv + Radio)

→ R-squared = 0.910

→ Adding "Radio" increases the R squared significantly this means Radio has a strong relationship with the dependent var, helping explain 91% of variance

(*) Why does R-squared increase when you add vars?

→ each new variable provides additional info that helps to explain the target

→ R squared always increase or stay the same when you add new independent variables

The Multiple linear regression is just an extension of simple linear regression, hence the formulation is largely the same.

Equation of Multiple Linear Regression:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon$$

⇒ Y ⇒ Dependent variable (response / output)

⇒ β_0 ⇒ Intercept (value of Y when all predictors are 0)

⇒ $\beta_1, \beta_2 \ldots \beta_p$: Coefficient of predictors ($X_1, X_2 \ldots X_p$)

↳ they measure the change in Y per unit increase in the corresponding X, keeping all other predictors constant.

⇒ ϵ : Error term

~~Interpretation of Constants~~ Interpretation of Coefficients:

→ Each β represents the effect of its predictive power

Interpretion of coefficients:

Each β represents the effect of its respective Predictor X on Y, holding other variables constant

eg β_1 tells how Y changes for a unit increase in X_1 assuming other predictors remains fixed

Apart from formulation, these are some aspects that still remain the same:

→ Model now fits a hyperplane instead of a line

→ Coefficients are still obtained by minimising the sum of squared errors

→ Assumptions from single linear regression still hold like Zero mean, Independent and normally distributed Error terms with constant variance.

✱ While moving from Simple Linear Regression to Multiple Linear Regression, the new aspects to consider when moving from simple to multiple linear regression

① Overfitting

↳ As we keep adding the variables, the model may become far too complex.

↳ It may end up memorising the training data and may fail to generalize.

↳ ie high training accuracy → low test accuracy

② Multicollinearity

↳ Association b/w the predictor variables

③ Hence the feature selection becomes an important aspect.

Multicollinearity

What is Multicollinearity?

→ Multicollinearity occurs when two or more independent variables in a regression model are highly correlated, in simple terms:

→ the independent variables are not truly independent

→ one variable can be predicted to a significant extent by one or more of the other variables

this redundancy in variables leads to difficulties in interpreting the regression coefficients, which impacts the statistical inference of the model.

Why is multicollinearity a problem?

Multicollinearity affects the following:

1. Interpretation:
   → the fundamental principle of regression is the interpretation of each coefficient as the change in the dependent variable (Y) when a single independent variable (x) changes while holding others constant.
   → Multicollinearity makes the interpretation unreliable because changing one var also causes the change in the other variable.

Conclusion from this :

→ When multicollinearity is present, small changes in the data can cause large changes in the estimated coefficients.

→ Multicollinearity does not affect the goodness of fit metrics (like R^2) or predictions but reduces the reliability of your regression coefficients.

How to detect Multicollinearity?

1. Pairwise Correlation

↳ Calculate the correlation coefficients b/w pairs of independent variables.

↳ If two variables are highly correlated (eg correlation coefficient > 0.7) it may indicate multicollinearity.

2. Variance Inflation factor

↳ A more robust way to detect multicollinearity

↳ VIF measures how much the variance of estimated regression coefficient is inflated due to multicollinearity.

Formula : $VIF_i = \frac{1}{1 - R_i^2}$

⊙ $VIF_i = \frac{1}{1-R_i^2}$

Where R_i^2 is the R^2 obtained by regressing the i-th independent variable on all other independent variables.

↳ Heuristic :

VIF < 5 : Low multicollinearity (Good to use)

5 ≤ VIF ≤ 10 : Moderate multicollinearity (Inspect further)

VIF ≥ 10 : High Multicollinearity (Consider eliminating or transforming the variables)

Why VIF is preferred over pairwise correlation

→ Pairwise correlation only evaluates relationship b/w two variables at a time.

→ VIF accounts for multicollinearity caused by combination of variables providing a more comprehensive view.

Some methods that can be used to deal with Multicollinearity are

→ Dropping Variables → Drop vars which are highly correlated with others

→ Create new var using the combination of old vars

→ PCA

⊛ Dealing With Categorical Variables

When we have a categorical variable with say 'n' levels, the idea of dummy variable creation is to build 'n-1' variables indicating the levels, for a variable say, 'Relationship' with three levels namely - Single, In relationship, Married we would create a dummy table like the following:

Relationship status	Single	In a relation	Married
Single	1	0	0
In a relationship	0	1	0
Married	0	0	1

But you can clearly see that there is no need of defining three different levels, if you drop a level say "Single" you would still be able to explain the three levels. (Hence Dropping Dummy Var)

Relationship status	In a relationship	Married
Single	0	0
In a relationship	1	0
Married	0	1

How it impacts the eqn?

Value	Indicator variable
Gender	Female
Male	0
Female	1

$$x_i = \begin{cases} 1 & \text{if } i\text{th person is female} \\ 0 & \text{if } i\text{th person is male} \end{cases}$$

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i = \begin{cases} \beta_0 + \beta_1 + \epsilon_i & \text{if } i\text{th} = \text{femal} \\ \beta_0 + \epsilon_i & \text{if } i\text{th person is male} \end{cases}$$

In short : N - 1 dummy variables can be used to describe a categorical variable with N variables.

Feature Scaling

Why do we need to scale features?

↳ ① Ease of Interpretation

② Faster Convergence for gradient descent methods

✳ Scaling in Regression Models:

Why is Scaling Important?

Scaling ensures that variables with different units or magnitudes are on the same scale, which is particularly important in the following scenarios:

1. **Algorithms Sensitive to Magnitude**: Algorithms like gradient descent (used in linear regression, logistic regression, etc.) or regularization techniques (Lasso and Ridge etc.) converge faster when the variables are scaled.

2. **Interpretability of Coefficients**: While scaling changes the magnitude of the coefficients, it does not affect statistical measures like t-statistic, P-value, or R^2

3. **Multicollinearity**: Scaling does not eliminate the multicollinearity but helps normalize variables for regularization based solns.

Key Parameters Affected by Scaling

→ Coefficients : Scaling changes their magnitude, but not their relationship with the dependent variable.

→ Predictions remain unaffected.

→ Statistical Metrics such as t-statistic, f-statistic, P-value, R^2 remains unchanged regardless of the scaling methods.

Methods of Scaling

① Standardization

↳ transforms the data into standard normal distribution where

↳ Mean = 0
Standard deviation = 1

formula ⇒ $x' = \frac{x - \text{mean}(x)}{\text{std}(x)}$

use case :

↳ Preferred when data is normally distributed

② Min Max Scaling

↳ Rescales the data to lie within a fixed range usually [0, 1] formula

$$x' = \frac{x - \min(x)}{\text{Max}(x) - \min(x)}$$

useful when you need the data in a specific range

⊛ Model Assessment and Comparison

When building regression models, there is often a trade off b/w keeping the model simple (with fewer Predictor variables) and explaining the highest possible variance (which means including as many relevant predictors as possible). To strike this balance 2 additional metrics come into play

↳ Adjusted R^2 and Akaike Information Criterion (AIC)

Adjusted R^2

↳ It is a modified version of R^2 that adjusts for the number of predictors in the model.

Unlike R^2, which always increases when a new Predictor is added, Adjusted R^2 considers whether the added variable improves the model enough to justify its inclusion.

$$\text{Adjusted } R^2 = 1 - \left(\frac{(1-R^2)(N-1)}{N-P-1} \right)$$

N → Nos of rows
P → Nos of Predictor variables

Higher Adjusted R^2 indicates a better model
It ensures only meaningfull variables are included

Akaike Information Criterion (AIC)

→ AIC is a metric used to compare diff models considering both the goodness of fit (how well the model fits the data) and the complexity (number of predictors), Lower AIC value indicates better models.

$$AIC = n \times \log\left(\frac{RSS}{n}\right) + 2p$$

$n \rightarrow$ nos of observations
$RSS \rightarrow$ Residual sum of squares
$P \rightarrow$ Nos of predictors

Eqs

Suppose we are building a regression model to Predict housing prices and you have 3 candidate models

Model	Predictors	R^2	Adj R^2	AIC
M_1	Lotsize, beds	0.75	0.74	210
M_2	Lotsize, beds, garage	0.78	0.76	190
M_3	All cols	0.80	0.73	250

M2 Model Best

Q $n = 101$, $P = 10$, $R^2 = 0.7$

$$\text{Adjusted } R^2 = \frac{1 - (1-0.7)(101-1)}{101-10-1}$$

$$1 - \frac{(0.3)(10\cancel{0})}{9\cancel{0}} = 1 - \frac{30}{390} = 1 - \frac{1}{3} = \frac{3-1}{3} = 2/3$$

In the context of Adjusted R^2

↳ Adding More Predictors (Independent variables) always increases R^2, even if those predictors don't actually help the model.

→ Adjusted R^2 penalize (reduces its value) if we add the predictors that don't increase model's value

---

Key Difference b/w the R^2 and the Adjusted R^2

---

① R-squared (R^2)

formula : $R^2 = 1 - \frac{RSS}{TSS}$

where RSS ⇒ Residual sum of squares

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

TSS ⇒ total sum of squares $\sum_{i=1}^{n} (y_i - \bar{y})^2$

y_i ⇒ Actual value of dependent var

$\hat{y}_i$ ⇒ Predicted value of the dependent var

$\bar{y}_i$ ⇒ Mean of actual values

Keypoints:

→ R^2 tells how much of the total variation in the dependent variable is explained by the independent variables

→ Adding more variables to the model always increases or maintains the R^2 value, even if the new variable has no relationship with the dependent variable, because it ignores the number of predictors.

⊛ Adjusted R^2 (R^2_{adj})

formula: $$R^2_{adj} = 1 - \left(\frac{(1-R^2)(n-1)}{n-p-1}\right)$$

R^2 ⇒ R squared value
n ⇒ number of observations (sample size)
p ⇒ Number of Predictors (independent var) in the model.

Keypoints:

→ R^2_{adj} adjusts for the number of predictors in the model

→ If we add an independent var that does not significantly improve the model → R^2_{adj} will decrease

→ It is more reliable than R^2 for multiple linear Regression because it accounts for overfitting

⊛ Feature Selection

Feature selection is a crucial step in building an effective linear regression model. It involves identifying the most relevant independent vars (predictors) that contribute to the dependent variable (outcome) while avoiding overfitting or redundancy.

Why is feature selection important?

① It simplifies the model
   ↳ It reduces the nos of predictors, making the model easier to interpret.

→ Helps in avoiding overfitting by excluding irrelevant or redundant variables.

→ Reduces Multicollinearity
   ↳ Prevents correlated variables from distorting the result.

Approaches to feature selection

① Exhaustive Search:
   ↳ Test all the possible combinations of the Predictors (2^p models for P features).

computationally expensive, and impractical for large datasets.

2. Manual feature Elimination

↳ works for smaller nos of predictors (eg 10-20)

Steps :

① Build the model with all features

↳ Fit the regression model with all the Potential Predictors.

② Drop features with high P values :

↳ Remove features that are statistically insignificant (eg $P > 0.05$)

③ Remove Redundant features

→ use correlation analysis and variance Inflation factor (VIF) to identify multi-collinear variables and eliminate them.

→ VIF rule of thumb : Drop features with VIF > 5 or 10

④ Rebuild the Model and Repeat

↳ Iteratively refine the model until only significant and non redundant features remain.

③ Automated feature Selection :

→ Usefull when dealing with large nos of Predictors (eg 100 or more)

Techniques :

- Recursive feature Elimination (RFE)
  - Iteratively build the model and eliminate the least important features based on a scoring metric.
- Stepwise Selection :
  - Forward Selection :
    - Start with no predictors and add them one at a time based on their significance.
  - Backward Elimination
    - Start with all predictors and remove them one at a time based on their Pvalues or other metrics.
  - Bidirectional / Stepwise selection
    - combines forward and backward selection.

RFE In Detail + Graded Ques Study

RFE stands for Recursive feature elimination, is a feature technique used in machine learning to select the most important feature by recursively removing the least important ones, It helps in improving the model performance, reduce overfitting and speeding up the model training process by eliminating the irrelevant features.

Here's a breakdown of how RFE works

RFE works by fitting a model to the data and ranking features based on their importance, the model assign weights to features, indicating how influential each feature is in predicting target variable.

⇒ Recursive Elimination

After the Model training is completed, RFE Recursively eliminates the least important features, this process is repeated until the specified number of features remains, on each iteration the least imp feature is removed and the model is retrained.

⇒ Ranking features:

RFE uses model's performance to evaluate the imp of each feature, features with the least weights or those contributing the least to the model's prediction are eliminated first.

Example with Linear Regression:

↳ we want to predict a target variable using linear regression

let's say we have the dataset as follows:

feature 1 (X1)	Feature 2 (X2)	Feature 3 (X3)	Target (Y)
5	10	100	200
6	12	120	220
7	14	130	240
8	16	150	250

Step 2 : Train Initial Model

You first train a linear regression model using all the features (X_1, X_2, X_3), the model will assign weights to each feature based on how they affect the target variable Y.

Step 2 : Train Initial Model

You first train a linear regression model using all features (X_1, X_2, X_3) → the model will assign weights to each feature based on how they affect the target variable Y.

Step 3 : Evaluate Feature Importance

RFE uses the weights of the features assigned by the linear regression model to rank their importance. If X_1 has a weight of 0.5, X_2 has 1.2 and X_3 has 0.1, It means X_2 is the most important feature while X_3 is the least important.

Step: 4 Eliminate Least Important feature

Since X_3 is the least important, RFE will remove it from the dataset, now you're left with just X_1 and X_2.

Step 5: Retrain the Model

RFE then retrains the model with only the remaining features (X_1, X_2), the model will reassign weights to these features and repeat the process

Step 6: Repeat Until Desired Number of features remains

this process continues recursively, eliminating one feature at a time based on importance, until the specified nos of features is reached.

Advantages of RFE

① Improve model performance by removing unnecessary features

② It provides a ranking of features, helping to understand which ones are most impactful.

Disadvantages:

→ Computationally expensive, Requires a model.

Code example with Linear Regression

```
from sklearn.feature_selection import RFE
from sklearn.linear_model import LinearRegressio
import pandas as pd

# eg Dataset

data = pd.DataFrame({
        'X1': [5, 6, 7, 8],
        'X2': [10, 12, 14, 16],
        'X3': [100, 120, 130, 150],
        'y': [200, 220, 240, 250]
    })

# features and target variables
X = data[['X1', 'X2', 'X3']]
y = data['y']
# Initialize the model
model = LinearRegression()
# Initialize Model in RFE and add the nos of features
  to select
rfe = RFE(model, 2) # Select top 2 features
# fit RFE
rfe.fit(X, y)
```

```
# Get the features selected by RFE
Selected_features = X.columns[rfe.support_]
Print(selected_features)
```

Notes On Multiple Linear Regression

Summary

Multiple Linear Regression (MLR) is a statistical technique used to predict the values of a dependent var Y based on multiple independent variable $X_1, X_2 \ldots, X_n$, It extends simple linear regression which involves one independent variable.

$$Y = B_0 + B_1 X_1 + B_2 X_2 + \cdots + B_n X_n + \epsilon$$

where : $y \Rightarrow$ Dependent var (Target)

$X_1, X_2 \ldots X_n$: Independent var

$B_0 \Rightarrow$ Intercept

$B_1, B_2 \ldots B_n$: Coefficients of independent variables

$\epsilon \Rightarrow$ Error term (residual)

Assumptions of MLR :

1. Linearity : the relationship b/w dependent and independent var is linear.
2. errors should be independent
3. Homoscedasticity : Variance of residuals should be const across all levels

④ Independent variables should not be too highly correlated.

⑤ Residuals should be normally distributed.

✱ Solving MLR using an example

Problem statement : Predict the price of house (Y) using its size in sq feet (X_1) and number of bedrooms (X_2)

Step1 : Data

House Size (X_1)	Number of Bedroom (X_2)	Price (Y)
1500	3	40,00,000
2000	4	55,00,000
2500	4	62,00,000
1800	3	48,00,000
2300	5	65,00,000

Step2 : Model

$$y = B_0 + B_1X_1 + B_2X_2 + G$$

Assume $B_0 = 20,000$, $B_1 = 10$, $B_2 = 50,000$

Step3 : Prediction

to predict the price of a house with size $X_1 = 2100$ sqft and $X_2 = 4$ bed room

$$y = 20,000 + 10(2100) + 50000(4)$$

$$\rightarrow \boxed{y = 2,41,000}$$

Metrics to evaluate Model Performance

① R-squared (R^2) : Proportion of variance in Y explained by $X_1, X_2, \ldots X_n$

② Adjusted R squared : Penalizes for the addition of irrelevant predictors

③ Mean Squared Error (MSE) : Avg squared diff b/w predicted and actual values

④ Root Mean Squared Error (RMSE) : Square root of MSE, same units as Y.

✱ Model Performance Metrics

to evaluate how well a multiple linear regression model fits the data, we use the following metrics

① R-squared (R^2)

It measures the proportion of variance in the dependent var (Y) that is explained by the independent var ($X_1, X_2, \ldots, X_n$)

↳ Range from 0 to 1, Closer to 1 indicates a better model fit.

formula ⇒ $R^2 = 1 - RSS/TSS$

↳ Residual Sum of Squares

↳ Total Sum of Squares

eg

We have the following values for the dependent var y and it's corresponding predicted values ($\hat{Y}$)

Actual (Y)	Predicted ($\hat{Y}$)
50	48
60	63
55	54
70	69

$$RSS = \Sigma (y - \hat{y})^2 \quad | \quad TSS = \Sigma (y - \bar{y})^2$$

Here $\bar{y}$ is the mean of $y = (50+60+55+70)/4 = 58.75$

$$TSS = (50-58.75)^2 + (60-58.75)^2 + (55-58.75)^2 + (70-58.75)^2 = 219.75$$

$$RSS = (50-48)^2 + (60-63)^2 + (55-54)^2 + (70-69)^2 = 15$$

$$R^2 = 1 - \frac{RSS}{TSS} = 1 - \frac{15}{219.75} = 0.9318$$

So R^2 explains 93.18% of the variance in Y.

② Adjusted R-squared

Adding irrelevant predictors can artificially increase R^2, even if the predictors have no real influence, Adjusted R^2 corrects for this.

$$R_{adj}^2 = 1 - \left(\frac{(1-R^2)(n-1)}{n-p-1}\right)$$

where $n \Rightarrow$ nos of observations

$P \Rightarrow$ nos of predictors

eg

Suppose $R^2 = 0.9$, nos of predictors $= 2$ and the nos of observations $n = 10$

$$R^2_{adj} = 1 - \frac{((1-0.9)(10-1))}{10-2-1}$$

$$\Rightarrow R^2 adj = 1 - \frac{((0.1)(9))}{7} = 0.3714$$

(3) Mean Squared Error (MSE)

↳ Avg squared diff b/w actual and predicted values.

formula: $\Rightarrow MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y_i})^2$

Here $n \Rightarrow$ nos of observation

eg using the earlier table of Actual & Predicted val

$$MSE = \frac{(50-48)^2 + (60-63)^2 + (55-54)^2 + (70-69)^2}{4}$$

$$\frac{4+9+1+1}{4} = \frac{15}{4} = 3.75$$