Date : / /

ML

* Example with Data:

House Size (x)	Rent (y)
800	1.5
1200	2.0
1500	2.5
1800	3.0

Ans:

Step-1 : Min-Max Normalization:

formula:

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

$$y' = \frac{y - \min(y)}{\max(y) - \min(y)}$$

∴ Applying Normalization:

Original x	Normalized x'
800	$\frac{800-800}{1800-800} = 0$
1200	$\frac{1200-800}{1800-1200} = 0.4$
1500	$\frac{1500-800}{1800-800} = 0.7$
1800	$\frac{1800-800}{1800-800} = 1$

Original y	Normalized y'
1.5	$\frac{1.5-1.5}{3.0-1.5} = 0$
2.0	$\frac{2.0-1.5}{3.0-1.5} = 0.3333$
2.5	$\frac{2.5-1.5}{3.0-1.5} = 0.6667$
3.0	$\frac{3.0-1.5}{3.0-1.5} = 1$

Date : / /

Step-2

* Initialize Parameters:

We start with: $w=0, b=0, \alpha=0.3$

Step3: Compute Predictions:

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

Initially all predictions are,

$$\hat{y} = 0 \times x + 0 = 0$$

Step4: Compute Loss (MSE):- mean Squared Error

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

$$= \frac{1}{4} [(0-0)^2 + (0-0.3333)^2 + (0-0.6667)^2 + (0-1)^2]$$

$$= 0.3889$$

Step-5: Compute Gradients:

$$\frac{\partial J}{\partial w} = -\frac{2}{n} \sum_{i=1}^{n} x_i (y_i - \hat{y}_i)$$

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

Evaluating the gradient with respect to w:

$$\frac{\partial J}{\partial w} = -\frac{2}{4} [0(0-0) + 0.4(0.3333-0) + 0.7(0.6667-0) + 1(1-0)]$$

$$= -0.8$$

evaluating the gradient with respect to b:

$$\frac{\partial J}{\partial b} = -\frac{2}{4} [(0-0) + (0.3333-0) + (0.6667-0) + (1-0)]$$

$$= -1.$$

Date : / /

Step-6 : Update Parameters

$$w = w - \alpha \frac{\partial J}{\partial w}$$

$$= 0 - (0.3 \times -0.8) = 0.24$$

$$b = b - \alpha \frac{\partial J}{\partial b}$$

$$= 0 - (0.3 \times -1) = 0.3$$

Step:7: Compute New Predictions

The new predictions are:

$$\hat{y} = 0.24 \times x' + 0.3$$

x'	y'
0	0.3
0.4	0.396
0.7	0.468
1.0	0.54

নতুন মান বসানোর পর $\hat{y}$ পাবো

Step8:

The new loss is:

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

$$= \frac{1}{4} [(0.3 - 0)^2 + (0.396 - 0.3333)^2 + (0.468 - 0.6667)^2 + (0.54 - 1)^2]$$

$$= 0.0863$$

Date :

Lecture 5

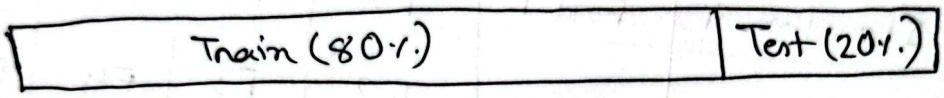
Assessing Model Performance

Holdout Validation

* Split the Dataset into two parts - Training set & a test set

- Training Data: Data the model was trained on.
- Test Data : Data the model has not seen before.

Typically, with a split ratio of 70:30 or 80:20



K-Fold Cross Validation

K-Fold cross Validation is a method to check how well a machine performs. Instead of using just one train test split, it divides the dataset into K parts (folds).

How it works:

1. The dataset is split into equal K parts (folds).
2. The model is trained K times, each time using K-1 fold for training and 1 fold for testing.
3. Each fold gets a chance to be the test set once.
4. The final performance is the average of all K test results.

AI, ML, OS, LAW, ENG

Date : / /

5 Fold Cross Validation

Iteration 1	Fold 1	F2	F3	F4	F5
Iteration 2	Fold 1	F2	F3	F4	F5
Iteration 3	Fold 1	F2	F3	F4	F5
Iteration 4	Fold 1	F2	F3	F4	F5
Iteration 5	Fold 1	F2	F3	F4	F5

Confusion Matrix

- TP (True Positive): Connectly Predicted positive case.
- TN (True Negative): " " Negative Case.
- FP (False Positive): Inconnectly " Positive case
- FN (False Negative): " " Negative ".

Negative → 0

Positive → 1

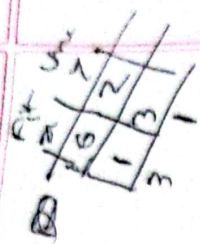
1 - Positive
0 - Negative

Date : / /

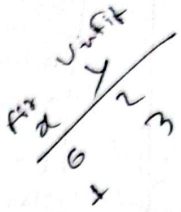
Instance	1	2	3	4	5	6	7	8	9	10	11	12
Actual Classification	1	1	1	1	1	1	1	1	0	0	0	0
Predicted Classification	0	0	1	1	1	1	1	1	1	0	0	0
Result.	FN	FN	TP	TP	TP	TP	TP	TP	FP	TN	TN	TN

True
False
~~False~~
Positive
Negative

Date :



		Predicted Condition	
	Total Population P+N	Positive (PP)	Negative (PN)
Actual Condition	Positive (P)	True Positive	FN
	Negative (N)	FP	TN



		Prediction Condition	
	Total 8+4 = 12	Fit = 7	Unfit = 5
Actual Condition	Fit = 8	6	2
	Unfit = 4	1	3

* Accuracy = $\frac{TP+TN}{TP+TN+FP+FN} = \frac{6+3}{6+3+1+2} = 0.75 = 75\%$.

* Precision = $\frac{TP}{TP+FP} = \frac{6}{6+1} = 0.857 = 85.7\%$.

* Sensivity / Recall = $\frac{TP}{TP+FN} = \frac{6}{6+2} = 0.75 = 75\%$.

* Specificity = $\frac{TN}{TN+FP} = \frac{3}{3+1} = 0.75 = 75\%$.

∴ Fmeasure / F-score,

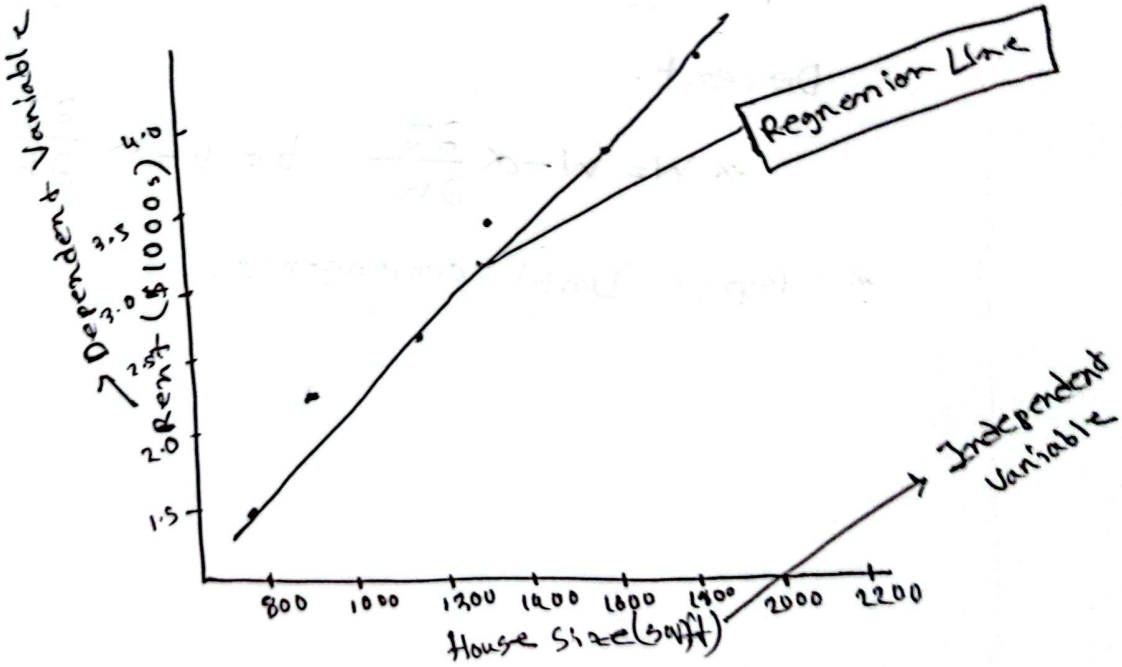
$$F1 = \frac{2 \times (Precision \times Recall)}{(precision + recall)}$$

$$= \frac{2(0.85\% \times 0.75)}{(0.857 + 0.75)} = 0.80 = 80\%.$$

Date :

Supervised Learning
Linear Regression

- **Linear Regression** is a basic technique in machine learning used for predicting continuous values. (price, temperature on sales).
- It helps understand the relationship between input and output variables using straight line.
- **Dependent Variable (Label)**: The value we want to predic (e.g, house rent)
- **Independent Variable (Feature)**: The Input used for prediction.



Date :

Mathematics Behind Linear Regression

- Hypothesis function : $\hat{y} = WX + b$
- Cost function : Mean Squared Error $= J(w,b) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$

Linear Regression Algorithm

* Initialize weight(W), bias (b), and learning rate (α)
* Compute the predicted value. $\hat{y}$
* Compute the loss using Mean Squared Error (MSE).
* Update weight and ~~bias~~ bias using Gradient Descent.
  * $W = W - \alpha \frac{\partial J}{\partial W}$, $b = b - \alpha \frac{\partial J}{\partial b}$
* Repeat Until convergence.

Date :

Multiple Linear Regression

It uses two or more independent variables (predicting rent based on house size and location).

* The formulation: $\hat{y} = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_p x_p + b$

* Gradient Descent for Multiple Linear Regression

$$w_i = w_i - \alpha \frac{\partial J}{\partial w_i}, \quad b = b - \alpha \frac{\partial J}{\partial b}$$

Pros:

* Simple model
* Computationally efficient
* Interpretability of the Output

Cons:

* Overly-Simplistic
* Linearity Assumption
* Severely affected by Outliers.

Date :

Polynomial Regression

Polynomial Regression is used when the ~~relat~~ relationship between variables is not a straight line but a curve.

- It fits an nth-degree polynomial equation to the data.

Example: Instead of a straight line, we fit a curve to better match the data points.

Types of Polynomial Regression:

- Linear (degree = 1) → straight line
- Quadratic (degree = 2) → U-shaped curve
- Cubic (degree = 3) → S-shaped curve

Pros:

→ Can model more complex ~~relat~~ relationships.

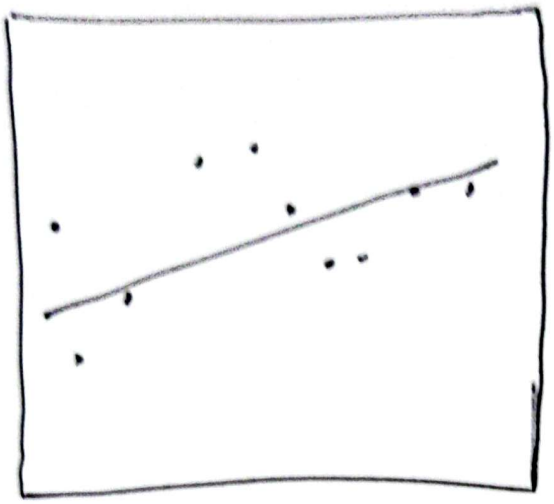
→ Works well when a straight line doesn't fit.

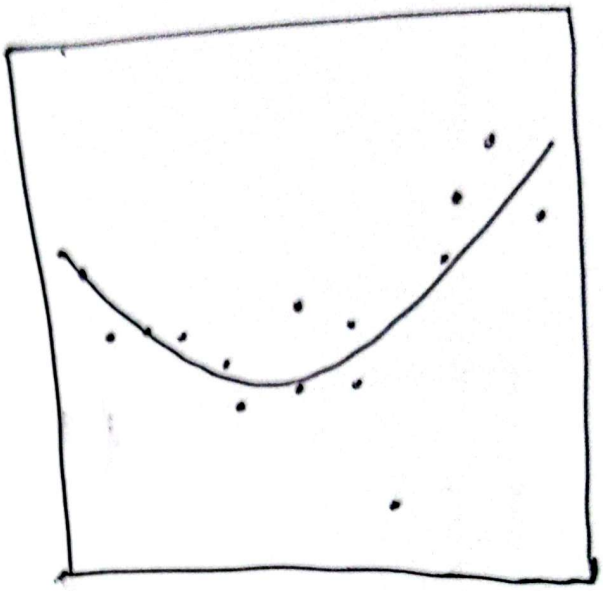
Cons:

→ Very sensitive to Outliers (one bad value can effect the curve)

Date

Why Polynomial Regression





Date : / /

Linear Regression

* Continuous Values নিয়ে কাজ করে।
* এটি independent (y) এবং dependent variable (x) নিয়ে সম্পর্ক স্থাপন করে।
* It can be multivariate linear regression

Mathematic Behind Linear Regression

linear equation হল, $y = mx + c$

ML এ লেখা হয় : $\hat{y} = wx + b$ = Hypothesis function.

(w = Weight, b = biased)

x	y
1	2
3	4
2	

→ True Value যেটা data set এ already আছে।

Suppose, $y = 2x + 4$

তাহলে,

$y = 2 \times 2 + 4$

$= 8$

$\therefore y = 8$ → এই value কে $\hat{y}$ — Predicted Value

$\therefore$ Mean Squared Error $(MSE) = J(w, b) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$

(y_i = real value, $\hat{y}_i$ = Predicted value)

Date :

Linear Regression Algorithm

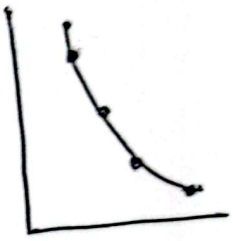
Here, $y = Wx + b$

* Gradient Descent এর সাহায্যে সবচেয়ে Cost কমানোর কাজটা করা হয়,

* Cost কমানোর step: $W = W - \alpha \frac{\partial J}{\partial W}$, $b = b - \alpha \frac{\partial J}{\partial b}$

  $\frac{\partial J}{\partial W}$ ← Cost function

  Cost কমানোর step কতটুকু হবে তা



* যতক্ষণ পর্যন্ত Gradient Descent এর সাহায্যে Cost কমানো যাবে যতক্ষণ পর্যন্ত convergence হয়।

Multiple linear Regression

formula: $\hat{y} = W_0 + W_1 x_1 + W_2 x_2 + \cdots + W_p x_p + b$

∴ Gradient descent for multiple L.R:

$$W_i = W_i - \alpha \frac{\partial J}{\partial W}, \quad b = b - \alpha \frac{\partial J}{\partial b}$$

Date

Pros and Cons of Linear Regression

Pros:

- Simple model.
- Computationally efficient
- Interpretability of the Output

Cons

- Overly- Simplistic
- Linearity Assumption
- Severely affected by Outliers.

Gradient of vector

$$\vec{\nabla} f(x) = \vec{g} = g_1 \hat{i} + g_2 \hat{j}$$

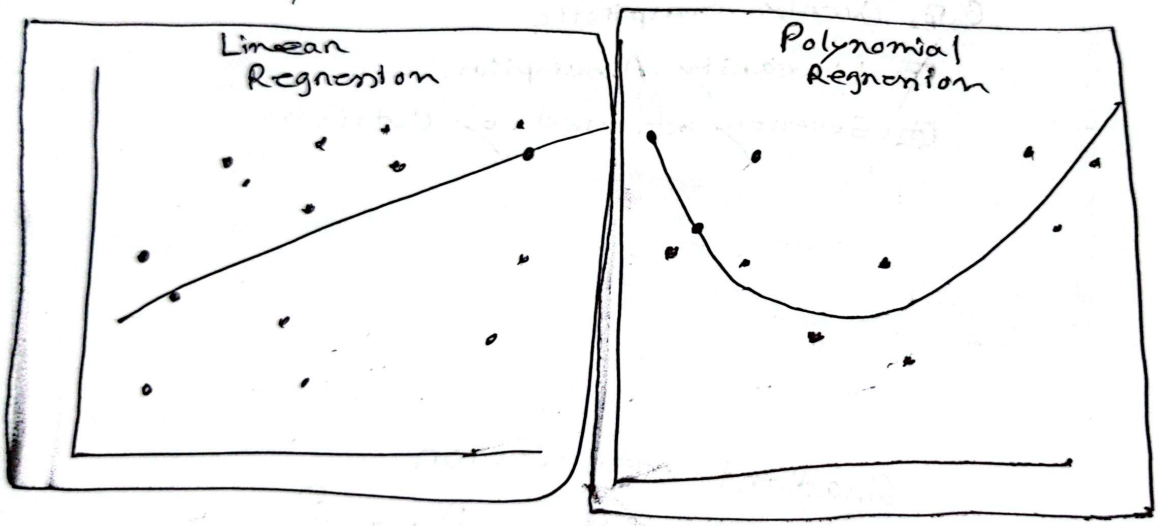
$$\|g\| = \sqrt{g_1^2 + g_2^2}$$

Date : / /

Polynomial Regression

- It describes the relationship between the independent variable x and the dependent variable y using an nth degree polynomial in x.

Here, polynomial equation is: $a_0x_0 + a_1x_1^2 + a_2x_2^3$



Why Polynomial Regression:
কারণ error কমে যায়