Supervised Learning

Sachin Tripathi

IIT(ISM), Dhanbad

Decision Tree

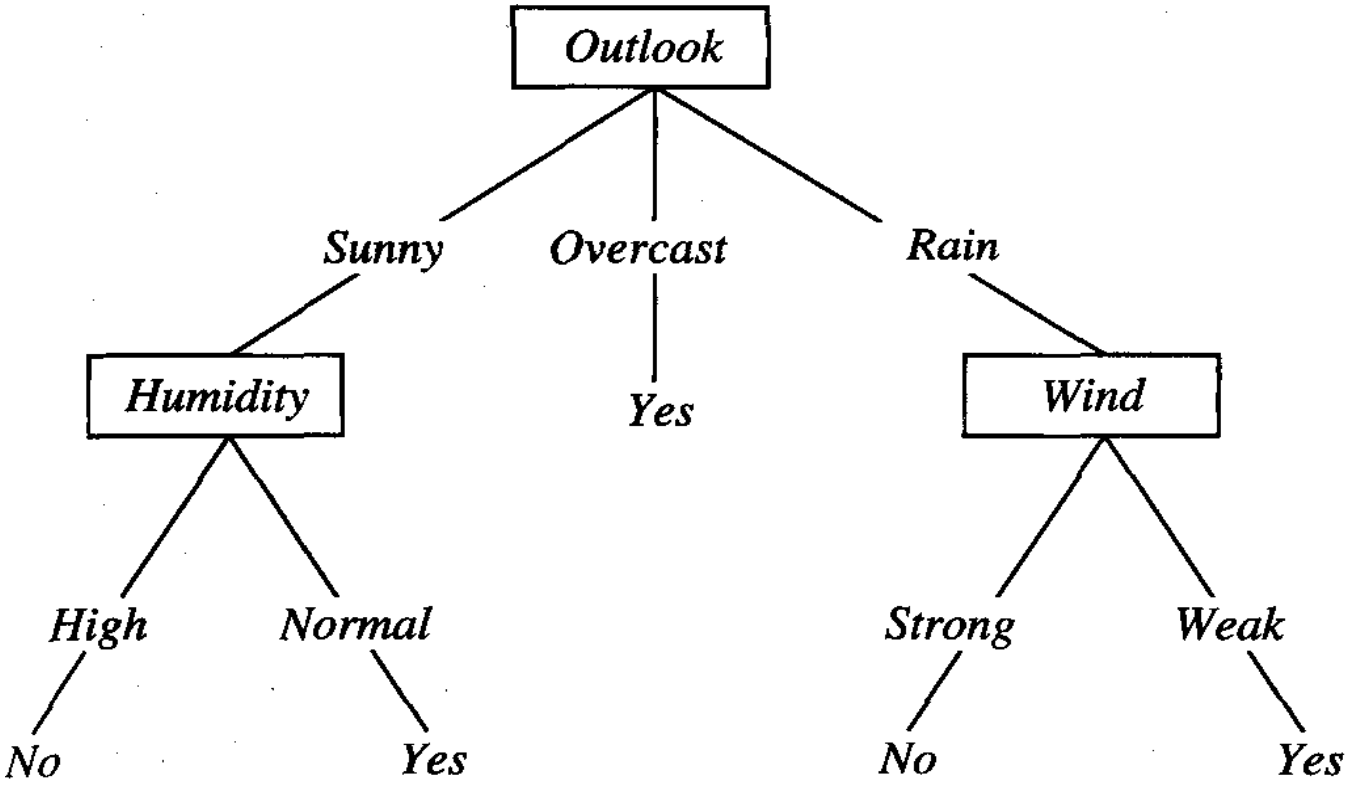
- Decision tree is a powerful tool for prediction and classification.
- A given set of objects and their attributes determine the decision of new objects.
- It represents rules which can be understood by humans and used in knowledge system such as database.

Key requirements

- **Attribute value** : object must be expressible in terms of properties or attributes.
- **Predefined classes :** the target function has a discrete output values.
- **Sufficient data** : enough training objects should be provided to learn the model.

Structure

- Decision tree is a classifier in the form of a tree.
- Decision node: specifies a test on a single attribute.
- Leaf node: indicates the value of the target attribute.
- Arc/edge: split of one attribute.
- Path: a disjunction of test to make the final decision.


Outlook
Sunny
Overcast
Rain
Humidity
Yes
Wind
High
Normal
Strong
Weak
No
Yes
No
Yes

Decision system

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Entropy computation

$$Entropy(S) \equiv \sum_{i=1}^{c} -p_i \log_2 p_i$$

$$Entropy(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

$$Entropy([9+, 5-]) = -(9/14) \log_2(9/14) - (5/14) \log_2(5/14)$$
$$= 0.940$$

Information Gain computation

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

Information Gain

continue..

$$
\begin{aligned}
Values(Wind) &= Weak, Strong \\
S &= [9+, 5-] \\
S_{Weak} &\leftarrow [6+, 2-] \\
S_{Strong} &\leftarrow [3+, 3-] \\
Gain(S, Wind) &= Entropy(S) - \sum_{v \in \{Weak, Strong\}} \frac{|S_v|}{|S|} Entropy(S_v) \\
&= Entropy(S) - (8/14) Entropy(S_{Weak}) \\
&\quad - (6/14) Entropy(S_{Strong}) \\
&= 0.940 - (8/14)0.811 - (6/14)1.00 \\
&= 0.048
\end{aligned}
$$

- Entropy(S_{Weak}) = -6/8 $\log_2$ (6/8) -2/8 $\log_2$ (2/8)
  = - 0.75 * (- 0. 41503749927) - 0.25 (-2)
  = 0.311278124 + 0.50
  = 0.811

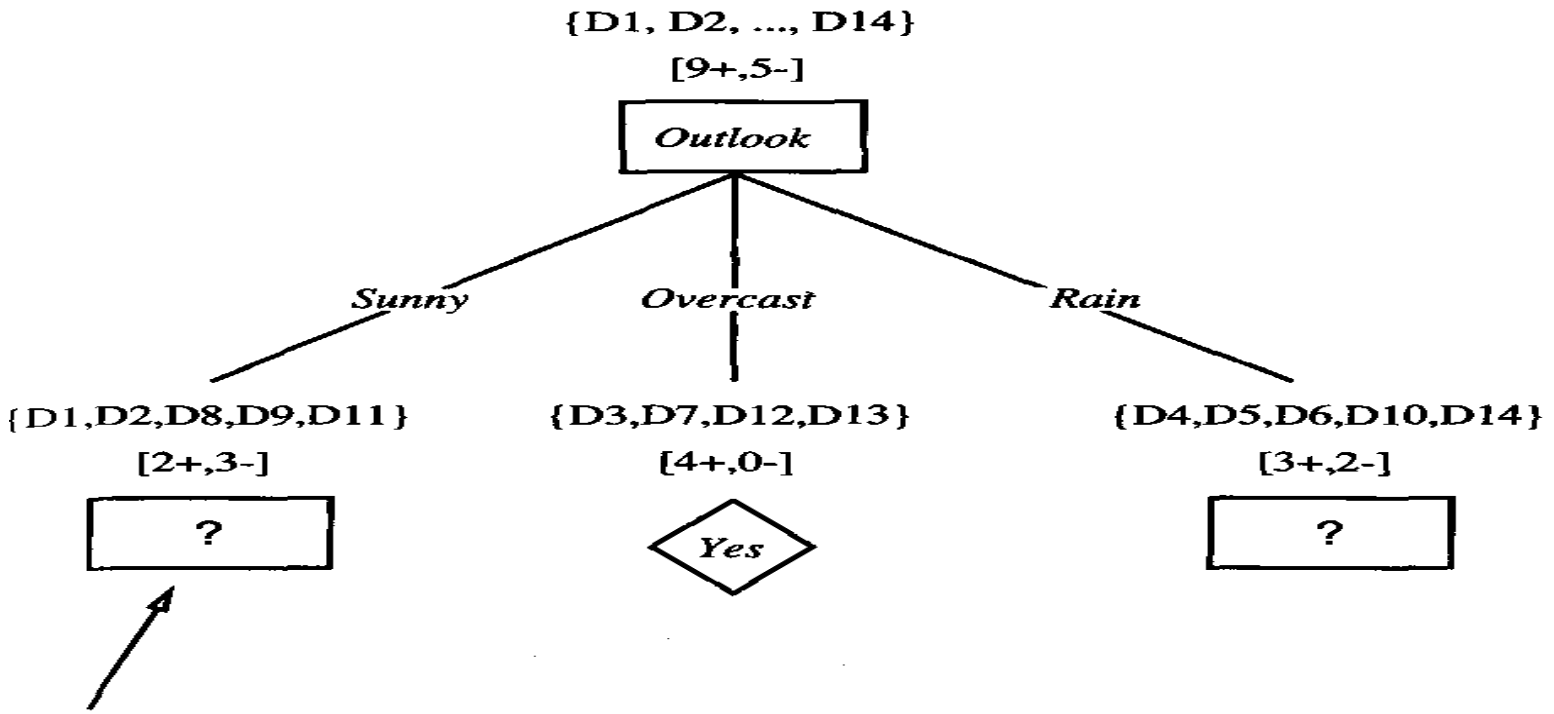
Entropy(S_{Strong}) = -3/6 $\log_2$ (3/6) – 3/6 $\log_2$(3/6)
= -0.5 * (-1) - 0.5 * (-1)
= 1

Information Gain

continue..

Similarly for all attributes

Gain(S, Outlook) = 0.246 Gain(S, Humidity) = 0.151 Gain(S, Wind) = 0.048 Gain(S, Temperature) = 0.029



Which attribute should be tested here?

S_{sunny} = {D1,D2,D8,D9,D11}

Gain (S_{sunny}, *Humidity*) = .970 - (3/5) 0.0 - (2/5) 0.0 = .970

Gain (S_{sunny}, *Temperature*) = .970 - (2/5) 0.0 - (2/5) 1.0 - (1/5) 0.0 = .570

Gain (S_{sunny}, *Wind*) = .970 - (2/5) 1.0 - (3/5) .918 = .019

- Entropy (S_{sunny}) = $-2/5 \log_2 (2/5) – 3/5 \log_2 (3/5)$
  $=-2/5 * (-1.32192) -3/5 (0.73696)$
  $= 0.5287 + 0.442$
  $=0.97…$

Humidity :

Entropy($S_{sunny,High}$) = $- 0/3 \log_2 (0/3) – 3/3 \log_2 (3/3)$
$= 0$

Entropy($S_{sunny,Normal}$) = $2/2 \log_2 (2/2) - 0/2 \log_2 (0/2)$
$= 0$

Temperature:

- Entropy ($S_{Sunny, Hot}$) =0

  Entropy ($S_{Sunny, Mild}$) $= -1/2\log_2(1/2) - 1/2\log_2(1/2)$

  $= 1/2 + 1/2 = 1$

  Entropy ($S_{Sunny, Cool}$) = 0

Wind :

Entropy ($S_{Sunny,weak}$) $= -1/3\log_2(1/3) - 2/3\log_2(2/3)$

$= -1/3 * (-1.58496250072) - 2/3 * (-0.58496250072)$

$= 0.5230376252 + 0.3860752505$

$= 0.90$

- Entropy($S_{Sunny, Strong}$) = -1/2 $\log_2$ (1/2) – 1/2 $\log_2$ (1/2)
  = 1

Weakness

- Decision tree perform poorly with many class and small data.
- It computationally expensive to train.
- Overfitting (High Variance)
- Unstable (Small Changes in Data → Big Changes in Tree)
- Bias Toward Features with More Levels
- High dimensionality makes decision trees prone to Overfitting, computationally heavy, and less interpretable, which is why in practice, ensembles (Random Forest, XGBoost) are preferred.

Tutorial

You are a robot in a lumber yard, and must learn to discriminate Oak wood from Pine wood. You choose to learn a Decision Tree classifier. You are given the following training set examples:

Example	Density	Grain	Hardness	Class
Example #1	Heavy	Small	Hard	Oak
Example #2	Heavy	Large	Hard	Oak
Example #3	Heavy	Small	Hard	Oak
Example #4	Light	Large	Soft	Oak
Example #5	Light	Large	Hard	Pine
Example #6	Heavy	Small	Soft	Pine
Example #7	Heavy	Large	Soft	Pine
Example #8	Heavy	Small	Soft	Pine

(a) Which attribute wold information gain choose as the root of tree?

Oak 4, Pine 4

$$Entropy(S) = -\frac{4}{8}\log_2\frac{4}{8} - \frac{4}{8}\log_2\frac{4}{8} = 1$$

Density:

Heavy-6 (Oak-3, Pine-3), Linght-2 (Oak-1, Pine-1)

$$Entropy(Heavy) = -\frac{3}{6}\log_2\frac{3}{6} - \frac{3}{6}\log_2\frac{3}{6} = 1$$

$$Entropy(Light) = -\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2} = 1$$

$$Information\ Gain(S, Density) = 1 - \frac{6}{8} * 1 - \frac{2}{8} * 1 = 0$$

Grain:

Small-4 (Oak-2, Pine-2), Large-4 (Oak-2, Pine-2)

Entropy(Small)=1, Entropy(Large)=1

$$Information\ Gain(S, Grain) = 1 - \frac{4}{8} * 1 - \frac{4}{8} * 1 = 0$$

Hardness:

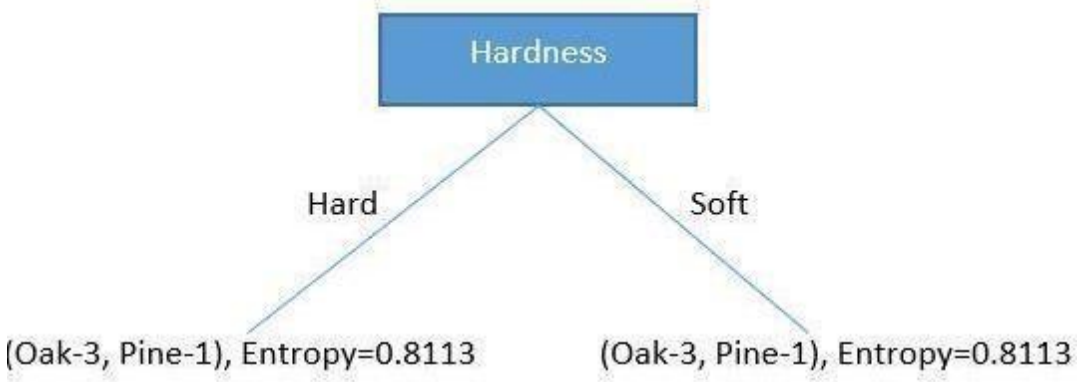
Hard-4 (Oak-3, Pine-1), Soft-4 (Oak-1, Pine-3)

Entropy(Hard)=0.8113, Entropy(Soft)=0.8113

$$Information\ Gain(S, Hardness) = 1 - \frac{4}{8} * 0.8113 - \frac{4}{8} * 0.8113 = 0.1887$$

Hence root node is '**Hardness**'

(b) Draw the decision tree that would be constructed by recursively applying information gain to select roots of subtrees, as in decision tree learning algorithm



Hard->Density (Heavy-> Oak-3, Light-> Pine-1), Entropy (Heavy)=0, Entropy (Light)=0

Soft->Density (Heavy-> Pine-3, Light-> Oak-1), Entropy (Heavy)=0, Entropy (Light)=0

$$Information\ Gain(Hard, Density) = 0.8113 - \frac{3}{4} * 0 - \frac{1}{4} * 0 = 0.8113$$

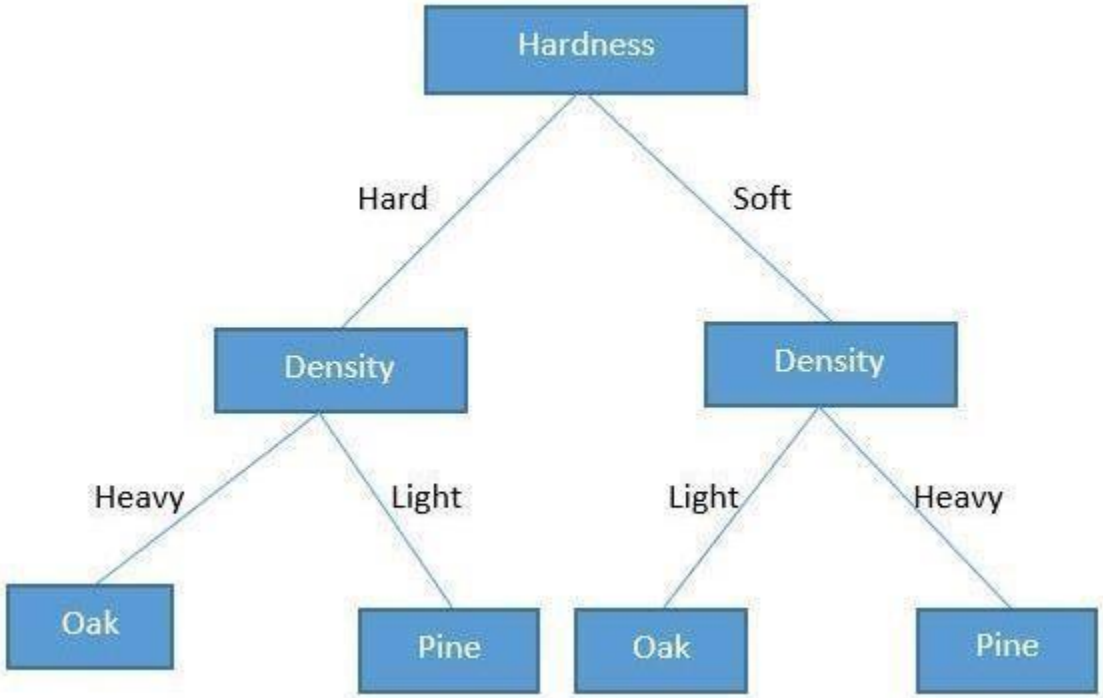
$$Information\ Gain(Soft, Density) = 0.8113 - \frac{3}{4} * 0 - \frac{1}{4} * 0 = 0.8113$$

Hard->Grain [Small-2 (Oak-2), Large-2 (Oak-1, Pine-1)], Entropy(Small)=0, Entropy(Large)=1

Soft->Grain [Small-2 (Pine-2), Large-2 (Oak-1, Pine-1)], Entropy(Small)=0, Entropy(Large)=1

$$Information\ Gain(Hard, Grain) = 0.8113 - \frac{2}{4} * 0 - \frac{2}{4} * 1 = 0.3113$$

$$Information\ Gain(Soft, Grain) = 0.8112 - \frac{2}{4} * 0 - \frac{2}{4} * 1 = 0.3113$$


Hardness
Hard
Soft
Density
Density
Heavy
Light
Light
Heavy
Oak
Pine
Oak
Pine

- Classify these new examples as Oak or Pine using your decision tree above.

Ans: **Pine**

Ans: **Oak**

  - i) [Density=Light, Grain=Small, Hardness=Hard]
  - ii)[Density=Light, Grain=Small, Hardness=Soft]

Linear Regression

- Linear regression is a type of supervised machine-learning algorithm that learns from the labelled datasets and maps the data points with most optimized linear functions which can be used for prediction on new datasets.

- It assumes that there is a linear relationship between the input and output, meaning the output changes at a constant rate as the input changes. This relationship is represented by a straight line.

- For example we want to predict a student's exam score based on how many hours they studied. We observe that as students study more hours, their scores go up. In the example of predicting exam scores based on hours studied. Here

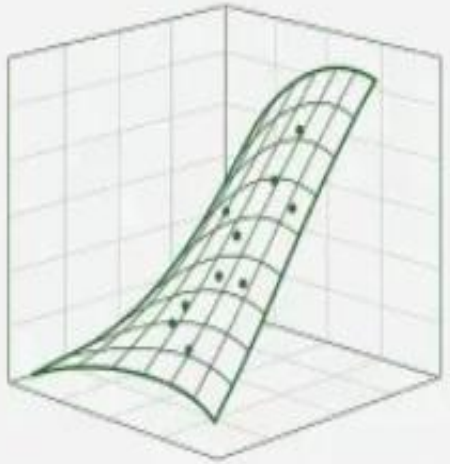
- Independent variable (input): Hours studied because it's the factor we control or observe.
- Dependent variable (output): Exam score because it depends on how many hours were studied.

Types of Linear Regression

Simple Linear Regression

Predicts the dependent variable using a single independent variable.

Multiple Linear Regression



Uses two or more independent variables to predict the dependent variable.

Why Linear Regression is Important?

- **Simplicity and Interpretability:** It's easy to understand and interpret, making it a starting point for learning about machine learning.
- **Predictive Ability**: Helps predict future outcomes based on past data, making it useful in various fields like finance, healthcare and marketing.
- **Basis for Other Models:** Many advanced algorithms, like logistic regression or neural networks, build on the concepts of linear regression.
- **Efficiency:** It's computationally efficient and works well for problems with a linear relationship.

- **Widely Used:** It’s one of the most widely used techniques in both statistics and machine learning for regression tasks.
- **Analysis:** It provides insights into relationships between variables (e.g., how much one variable influences another).

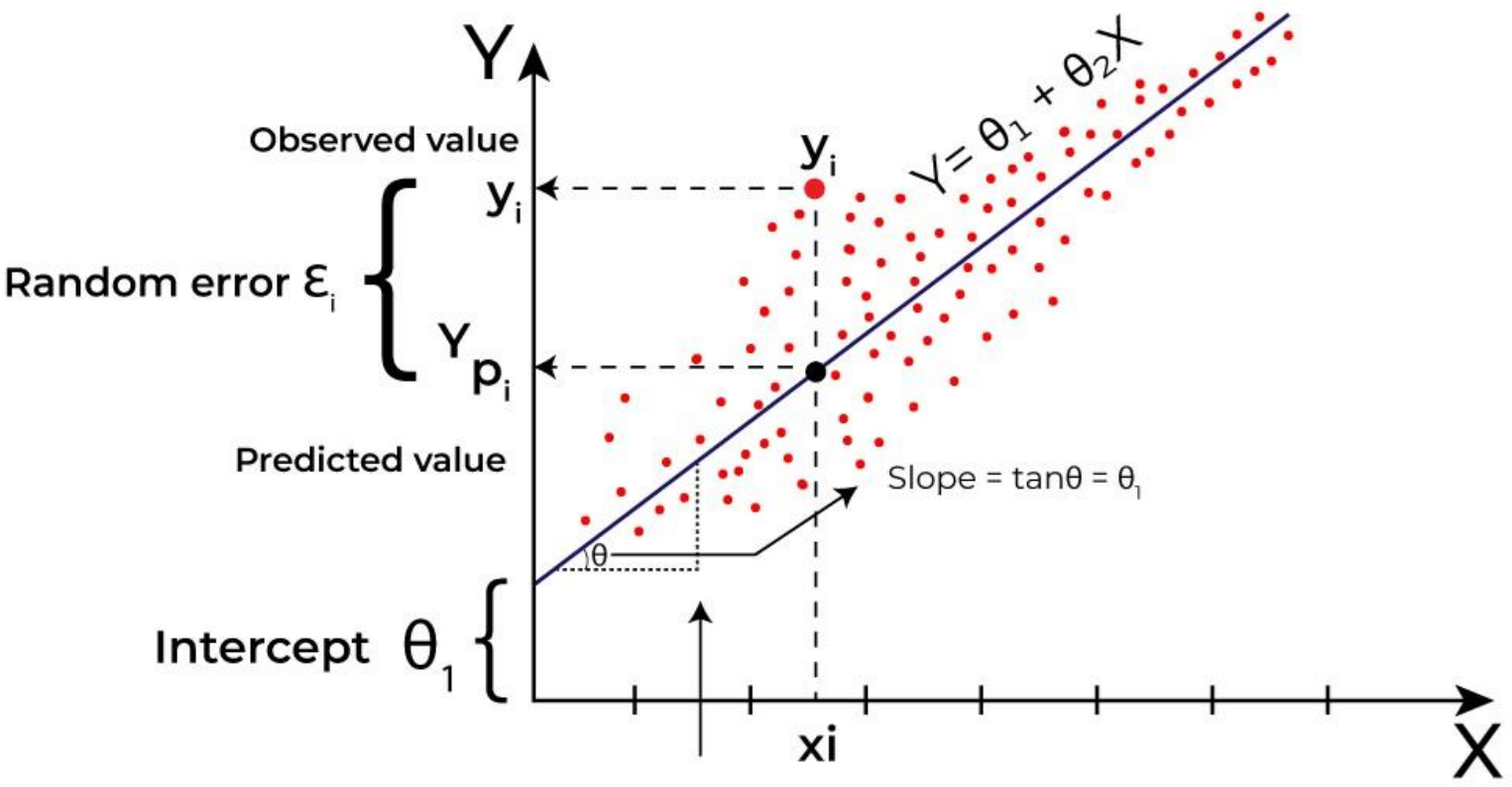
Best Fit Line in Linear Regression

- In linear regression, the best-fit line is the straight line that most accurately represents the relationship between the independent variable (input) and the dependent variable (output).
- It is the line that minimizes the difference between the actual data points and the predicted values from the model.

Goal of the Best-Fit Line

- The goal of linear regression is to find a straight line that minimizes the error (the difference) between the observed data points and the predicted values. This line helps us predict the dependent variable for new, unseen data.

Linear Regression



Linear Regression

- Here Y is called a dependent or target variable and X is called an independent variable also known as the predictor of Y.
- There are many types of functions or modules that can be used for regression.
- A linear function is the simplest type of function.
- Here, X may be a single feature or multiple features representing the problem.

Equation of the Best-Fit Line

- For simple linear regression (with one independent variable), the best-fit line is represented by the equation

$$Y = mx + by = mx + b$$

Where:

- y is the predicted value (dependent variable)
- x is the input (independent variable)
- m is the slope of the line (how much y changes when x changes)
- b is the intercept (the value of y when x = 0)

- The best-fit line will be the one that optimizes the values of m (slope) and b (intercept) so that the predicted y values are as close as possible to the actual data points.

Minimizing the Error

- The Least Squares Method
- To find the best-fit line, we use a method called Least Squares.
- The idea behind this method is to minimize the sum of squared differences between the actual values (data points) and the predicted values from the line. These differences are called residuals.
- The formula for residuals is: $Residual = y_i - \hat{y}_i$

Where: y_i is the actual observed value
$\hat{y}_i$ is the predicted value from the line for that x_i

- The least squares method minimizes the sum of the squared residuals:

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

This method ensures that the line best represents the data where the sum of the squared differences between the predicted values and actual values is as small as possible.

Interpretation of the Best-Fit Line

- **Slope (m):** The slope of the best-fit line indicates how much the dependent variable (y) changes with each unit change in the independent variable (x).

  - For example if the slope is 5, it means that for every 1-unit increase in x, the value of y increases by 5 units.
- **Intercept (b):** The intercept represents the predicted value of y when x = 0. It's the point where the line crosses the y-axis.

In linear regression some hypothesis are made to ensure reliability of the model's results.

Limitations

- Assumes Linearity: The method assumes the relationship between the variables is linear. If the relationship is non-linear, linear regression might not work well.
- Sensitivity to Outliers: Outliers can significantly affect the slope and intercept, skewing the best-fit line.

Hypothesis in Linear Regression

- In linear regression, the hypothesis function is the equation used to make predictions about the dependent variable based on the independent variables.
- It represents the relationship between the input features and the target output.
- For a simple case with one independent variable, the hypothesis function is:

  $h(x)=\beta0+\beta1xh(x)=\beta0+\beta1x$

- h(x)(ory^) is the predicted value of the dependent variable (y).
- x is the independent variable.
- β0 is the intercept, representing the value of y when x is 0.
- β1 is the slope, indicating how much y changes for each unit change in x.

Multiple Linear Regression

- For multiple linear regression (with more than one independent variable), the hypothesis function expands to:

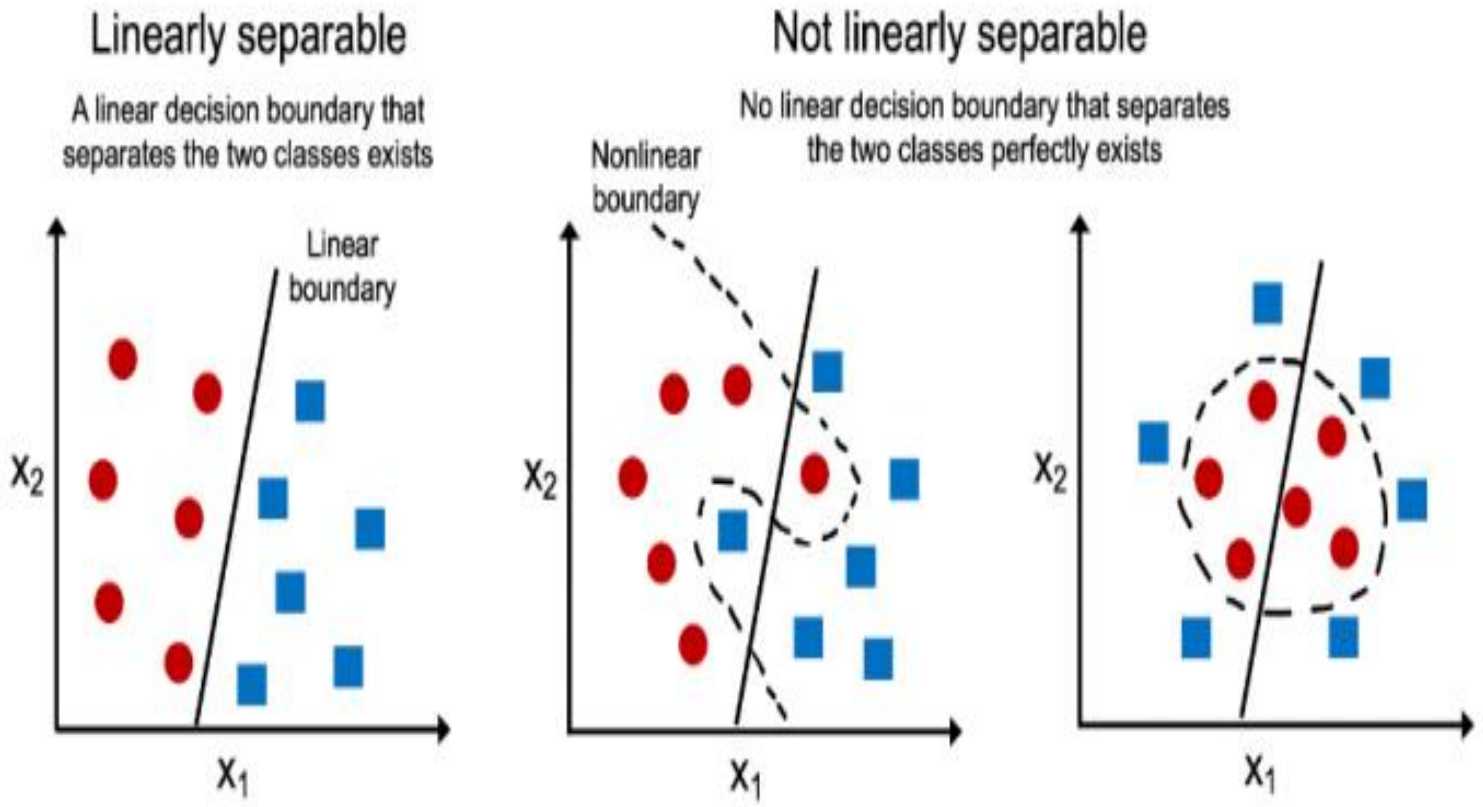
$$h(x_1, x_2, \ldots\ldots, x_k) = \beta 0 + \beta 1 x_1 + \beta 2 x_2 + \ldots + \beta k x_k$$

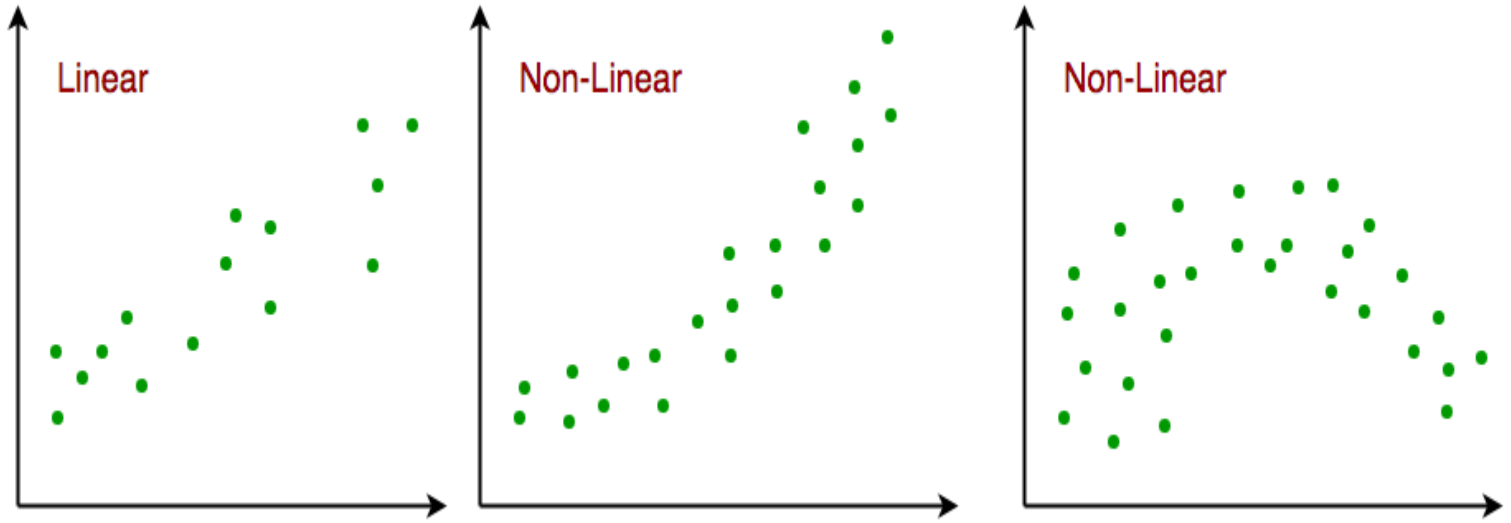
Where:

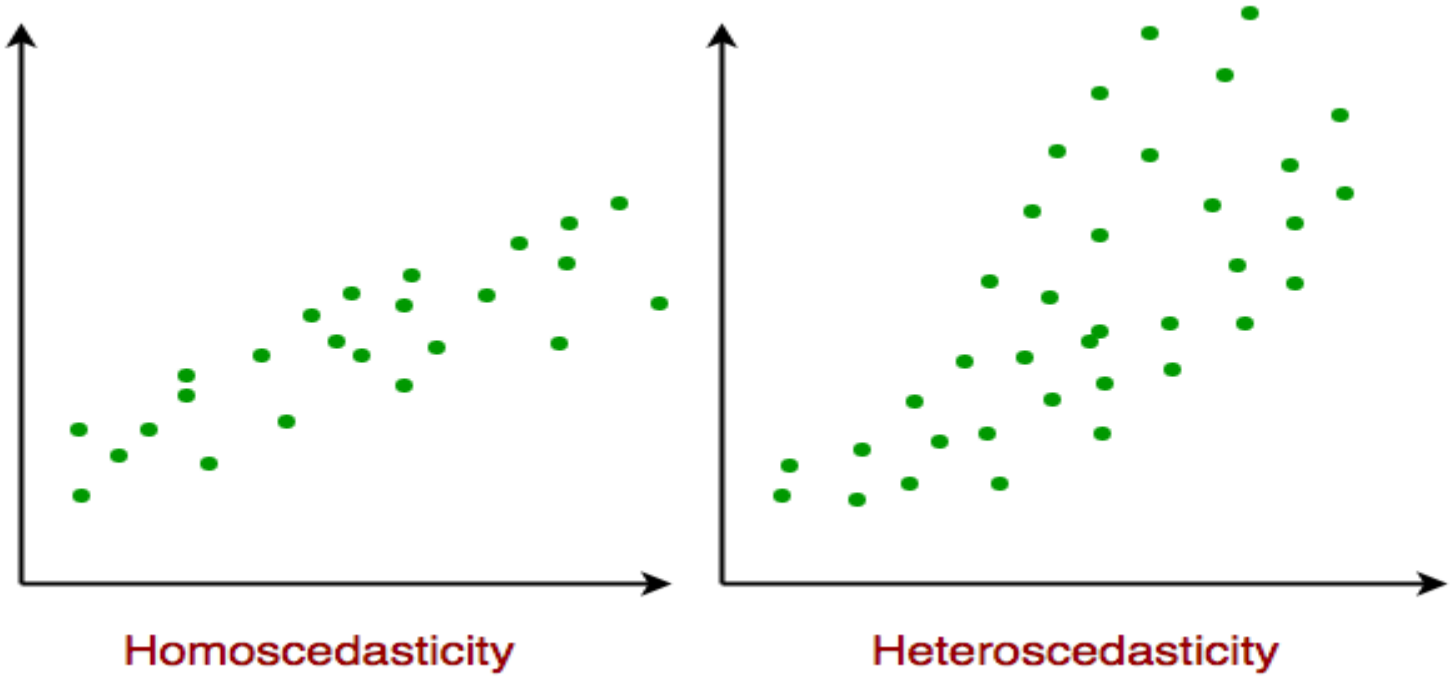
- $x_1, x_2, \ldots, x_k$ are the independent variables.
- $\beta 0$ is the intercept.
- $\beta 1, \beta 2, \ldots, \beta k$ are the coefficients, representing the influence of each respective independent variable on the predicted output.

Assumptions of the Linear Regression

- **Linearity:** The relationship between inputs (X) and the output (Y) is a straight line.
- **Independence of Errors:** The errors in predictions should not affect each other.
- **Constant Variance (Homoscedasticity):** The errors should have equal spread across all values of the input.
  - If the spread changes (like fans out or shrinks), it's called heteroscedasticity and it's a problem for the model.


Linearly separable
A linear decision boundary that separates the two classes exists
Linear boundary
x_2
x_1
Not linearly separable
No linear decision boundary that separates the two classes perfectly exists
Nonlinear boundary
x_2
x_1
x_2
x_1


Linear
Non-Linear
Non-Linear


Homoscedasticity
Heteroscedasticity

- **Normality of Errors:** The errors should follow a normal (bell-shaped) distribution.
- **No Multicollinearity(for multiple regression):** Input variables shouldn’t be too closely related to each other.
- **No Autocorrelation:** Errors shouldn't show repeating patterns, especially in time-based data.
- **Additivity:** The total effect on Y is just the sum of effects from each X, no mixing or interaction between them.'

Types of Regression

- Simple Linear Regression

  Example:
  Predicting a person's salary (y) based on their years of experience (x).

- Multiple Linear Regression

  Use Case of Multiple Linear Regression

  - Real Estate Pricing: In real estate MLR is used to predict property prices based on multiple factors such as location, size, number of bedrooms, etc.

- Financial Forecasting: Financial analysts use MLR to predict stock prices or economic indicators based on multiple influencing factors such as interest rates, inflation rates and market trends. This enables better investment strategies and risk management24.

- Agricultural Yield Prediction: Farmers can use MLR to estimate crop yields based on several variables like rainfall, temperature, soil quality and fertilizer usage. This information helps in planning agricultural practices for optimal productivity

- E-commerce Sales Analysis: An e-commerce company can utilize MLR to assess how various factors such as product price, marketing promotions and seasonal trends impact sales.

Cost function for Linear Regression

- The difference between the predicted value Y^ and the true value Y is called cost function or the loss function.

In Linear Regression, the Mean Squared Error (MSE) cost function is employed, which calculates the average of the squared errors between the predicted values y^ and the actual values yi

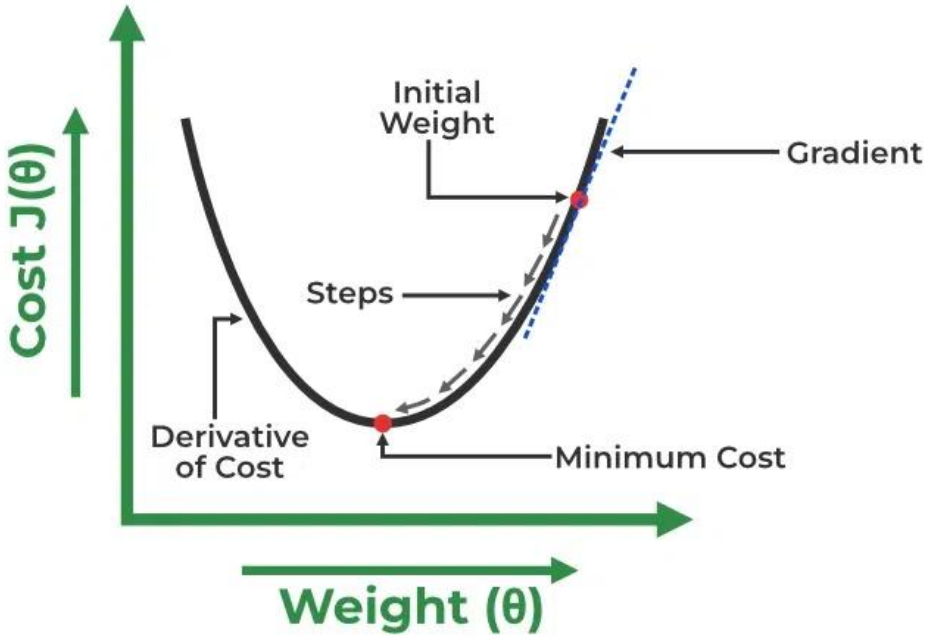
MSE function can be calculated as:

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

- Utilizing the MSE function, the iterative process of gradient descent is applied to update the values of θ1&θ2.
- This ensures that the MSE value converges to the global minima, signifying the most accurate fit of the linear regression line to the dataset.

Gradient Descent for Linear Regression

- Gradient descent is an optimization technique used to train a linear regression model by minimizing the prediction error.
- It works by starting with random model parameters and repeatedly adjusting them to reduce the difference between predicted and actual values.


Cost J(θ)
Initial
Weight
Gradient
Steps
Derivative
of Cost
Minimum Cost
Weight (θ)

How It Works?

- Start with random values for slope and intercept.
- Calculate the error between predicted and actual values.
- Find how much each parameter contributes to the error (gradient).
- Update the parameters in the direction that reduces the error.
- Repeat until the error is as small as possible.
- This helps the model find the best-fit line for the data.

Example

We have a dataset of study hours vs exam scores:

Hours (x)	Score (y)
1	2
2	4
3	6

We want to fit a linear regression model:$y=mx+c$y=mx+c

where: m = slope c = intercept

1. Loss Function

We use **Mean Squared Error (MSE)**:

$$J(m,c) = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - (mx_i + c)\right)^2$$

Gradient Descent Update Rule

To minimize $J(m, c)$, we update m and c using:

$$m_{\text{new}} = m - \alpha \cdot \frac{\partial J}{\partial m}$$

$$c_{\text{new}} = c - \alpha \cdot \frac{\partial J}{\partial c}$$

Where:

- α = learning rate
- Gradients are:

$$\frac{\partial J}{\partial m} = -\frac{2}{n} \sum_{i=1}^{n} x_i \left(y_i - (mx_i + c)\right)$$

$$\frac{\partial J}{\partial c} = -\frac{2}{n} \sum_{i=1}^{n} \left(y_i - (mx_i + c)\right)$$

Step-by-Step Calculation

- **Initial values:**
  m=0, c=0, α=0.01, n=3

Iteration 1

Predictions:

$$\hat{y} = [0, 0, 0]$$

Errors:

$$y - \hat{y} = [2, 4, 6]$$

Gradients:

$$\frac{\partial J}{\partial m} = -\frac{2}{3}\left[(1)(2) + (2)(4) + (3)(6)\right] = -\frac{2}{3}(2 + 8 + 18) = -\frac{2}{3}(28) = -18.67$$

$$\frac{\partial J}{\partial c} = -\frac{2}{3}(2 + 4 + 6) = -\frac{2}{3}(12) = -8$$

Update:

$$m = 0 - 0.01 \cdot (-18.67) = 0.1867$$

$$c = 0 - 0.01 \cdot (-8) = 0.08$$

Iteration 2

Predictions:

$$\hat{y} = [0.1867 + 0.08,\ 0.3734 + 0.08,\ 0.5601 + 0.08] = [0.2667, 0.4534, 0.6401]$$

Errors:

$$[2 - 0.2667, 4 - 0.4534, 6 - 0.6401] = [1.7333, 3.5466, 5.3599]$$

Gradients:

$$\frac{\partial J}{\partial m} = -\frac{2}{3}\left[(1)(1.7333) + (2)(3.5466) + (3)(5.3599)\right] = -\frac{2}{3}(1.7333 + 7.0932 + 16.0797) = -\frac{2}{3}(24.9062) = -16.6041$$

$$\frac{\partial J}{\partial c} = -\frac{2}{3}(1.7333 + 3.5466 + 5.3599) = -\frac{2}{3}(10.6398) = -7.0932$$

Update:

$$m = 0.1867 - 0.01 \cdot (-16.6041) = 0.3527$$

$$c = 0.08 - 0.01 \cdot (-7.0932) = 0.151$$

- We keep repeating until m and c converge.

- If we run enough iterations, we'll get: $m \approx 2, c \approx 0$

which matches the perfect relationship $y=2xy$.

Multivariable Example

- Suppose we have data about house prices. We want to predict House Price (y) using two features:

  x_1 = Size of house (in 100 sq.ft)

  x_2 = Number of bedrooms

Training dataset (3 houses):

House	x_1 (size)	x_2 (bedrooms)	y (price in lakhs)
1	1	1	1
2	2	2	2
3	3	2	2.5

We want the regression model:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

- General equation: $y = X\beta + \epsilon$

  Here,

$$X = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 2 \\ 1 & 3 & 2 \end{bmatrix}, \quad y = \begin{bmatrix} 1 \\ 2 \\ 2.5 \end{bmatrix}, \quad \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix}$$

(The first column of 1's is for the intercept.)

The solution is:

$$\hat{\beta} = (X^T X)^{-1} X^T y$$

- Compute $X^T X$ and $X^T y$

$$X^T X = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 1 & 2 & 2 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 2 \\ 1 & 3 & 2 \end{bmatrix} = \begin{bmatrix} 3 & 6 & 5 \\ 6 & 14 & 11 \\ 5 & 11 & 9 \end{bmatrix}$$

$$X^T y = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 1 & 2 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 2.5 \end{bmatrix} = \begin{bmatrix} 5.5 \\ 12.5 \\ 9.5 \end{bmatrix}$$

- We need: $\hat{\beta} = (X^T X)^{-1}(X^T y)$

$$\begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix} = \begin{bmatrix} 3 & 6 & 5 \\ 6 & 14 & 11 \\ 5 & 11 & 9 \end{bmatrix}^{-1} \begin{bmatrix} 5.5 \\ 12.5 \\ 9.5 \end{bmatrix}$$

- Using matrix algebra solution comes out to:

$$\hat{\beta}_0 = 0.0, \quad \hat{\beta}_1 = 0.5, \quad \hat{\beta}_2 = 0.5$$

- We want to minimize Mean Squared Error (MSE):

$$J(\beta_0, \beta_1, \beta_2) = \frac{1}{2m} \sum_{i=1}^{m} \left(\hat{y}^{(i)} - y^{(i)}\right)^2$$

where

$$\hat{y}^{(i)} = \beta_0 + \beta_1 x_1^{(i)} + \beta_2 x_2^{(i)}$$

- We compute partial derivatives:

$$\frac{\partial J}{\partial \beta_0} = \frac{1}{m}\sum_{i=1}^{m}(\hat{y}^{(i)} - y^{(i)})$$

$$\frac{\partial J}{\partial \beta_1} = \frac{1}{m}\sum_{i=1}^{m}(\hat{y}^{(i)} - y^{(i)}) \cdot x_1^{(i)}$$

$$\frac{\partial J}{\partial \beta_2} = \frac{1}{m}\sum_{i=1}^{m}(\hat{y}^{(i)} - y^{(i)}) \cdot x_2^{(i)}$$

For each iteration:

$$\beta_j := \beta_j - \alpha \cdot \frac{\partial J}{\partial \beta_j}$$

where α = learning rate.

- Let's try with initial values:
  $\beta 0=0, \beta 1=0, \beta 2=0$, Learning rate: $\alpha=0.1 \alpha=0.1$

Predictions: all $\hat{y} = 0$

Errors: $(0-1), (0-2), (0-2.5) = -1, -2, -2.5$

Now compute gradients:

$$\frac{\partial J}{\partial \beta_0} = \frac{1}{3}((-1) + (-2) + (-2.5)) = \frac{-5.5}{3} \approx -1.83$$

$$\frac{\partial J}{\partial \beta_1} = \frac{1}{3}((-1)(1) + (-2)(2) + (-2.5)(3)) = \frac{-13.5}{3} = -4.5$$

$$\frac{\partial J}{\partial \beta_2} = \frac{1}{3}((-1)(1) + (-2)(2) + (-2.5)(2)) = \frac{-11}{3} \approx -3.67$$

❑ Update Parameters

$$\beta_0 = 0 - 0.1(-1.83) = 0.183$$

$$\beta_1 = 0 - 0.1(-4.5) = 0.45$$

$$\beta_2 = 0 - 0.1(-3.67) = 0.367$$

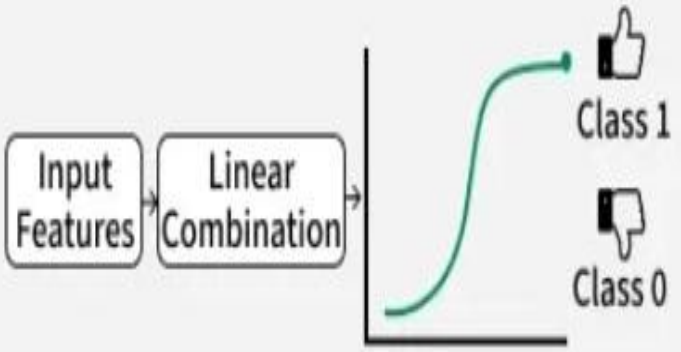
Now use new β to calculate predictions and update again.

The parameters converge to approximately after iterations to :

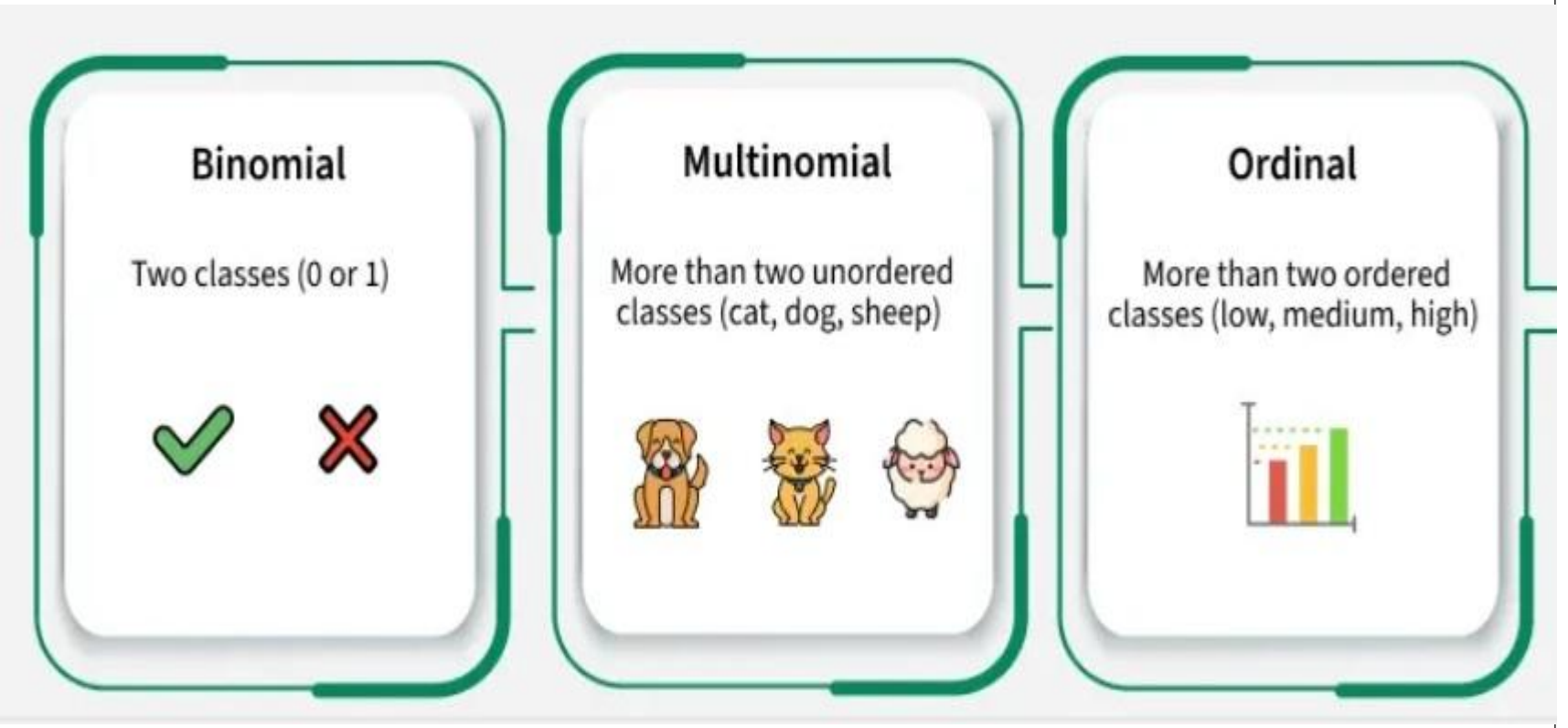
β0≈0, β1≈0.5, β2≈0.5

What is Logistic Regression

- Predicts the probability of a binary outcome (Yes/No, 0/1)
- Uses the sigmoid function to map inputs to probabilities (0 to 1)
- Ideal for classification tasks

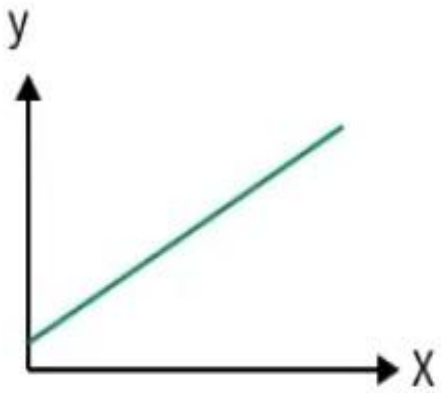


Types of Logistic Regression

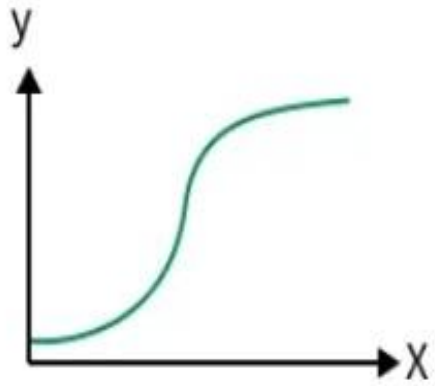


Linear Vs. Logistic Regression

- Predicts continuous values
- Uses best-fit line
- Solves regression problems



- Predicts categorical classes
- Uses sigmoid S-curve
- Solves classification problems



Assumptions

- Understanding the assumptions behind logistic regression is important to ensure the model is applied correctly, main assumptions are:

- **Independent observations:** Each data point is assumed to be independent of the others means there should be no correlation or dependence between the input samples.
- **Binary dependent variables:** It takes the assumption that the dependent variable must be binary, means it can take only two values. For more than two categories SoftMax functions are used.

- **Linearity relationship between independent variables and log odds:** The model assumes a linear relationship between the independent variables and the log odds of the dependent variable which means the predictors affect the log odds in a linear way.
- **No outliers:** The dataset should not contain extreme outliers as they can distort the estimation of the logistic regression coefficients.
- **Large sample size:** It requires a sufficiently large sample size to produce reliable and stable results.

Understanding Sigmoid Function

- The sigmoid function is a important part of logistic regression which is used to convert the raw output of the model into a probability value between 0 and 1.
- This function takes any real number and maps it into the range 0 to 1 forming an "S" shaped curve called the sigmoid curve or logistic curve. Because probabilities must lie between 0 and 1, the sigmoid function is perfect for this purpose.
- In logistic regression, we use a threshold value usually 0.5 to decide the class label.
  - If the sigmoid output is same or above the threshold, the input is classified as Class 1.
  - If it is below the threshold, the input is classified as Class 0.

How does Logistic Regression work?

- Logistic regression model transforms the linear regression function continuous value output into categorical value output using a sigmoid function which maps any real-valued set of independent variables input into a value between 0 and 1. This function is known as the logistic function.
- Suppose we have input features represented as a matrix:

$$X = \begin{bmatrix} x_{11} & \cdots & x_{1m} \\ x_{21} & \cdots & x_{2m} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nm} \end{bmatrix}$$

- The dependent variable is Y having only binary value i.e 0 or 1.

$$Y = \begin{cases} 0 & \text{if } Class\ 1 \\ 1 & \text{if } Class\ 2 \end{cases}$$

then, apply the multi-linear function to the input variables X.

$$z = \left(\sum_{i=1}^{n} w_i x_i\right) + b$$

- Here x_i is the i^{th} observation of X, wi=[w1,w2,w3,⋯, is the weights or Coefficient and b is the bias term also known as intercept.

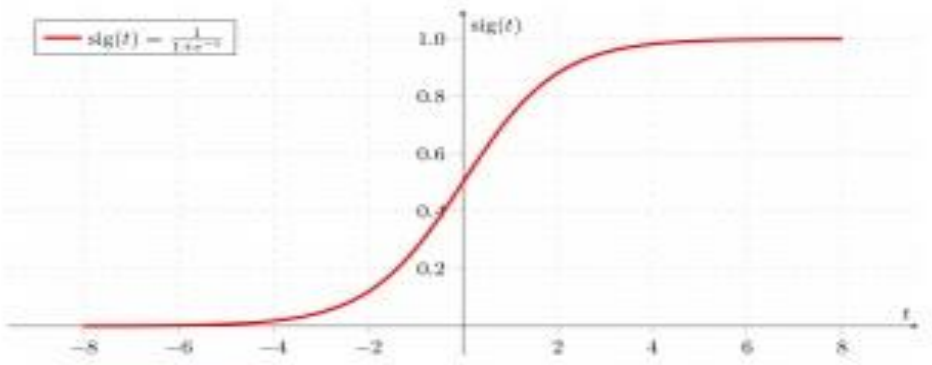
- Simply this can be represented as the dot product of weight and bias.

$$z = w \cdot X + b$$

- At this stage, z is a continuous value from the linear regression.
- Logistic regression then applies the sigmoid function to z to convert it into a probability between 0 and 1 which can be used to predict the class.

- Now we use the sigmoid function where the input will be z and we find the probability between 0 and 1. i.e. predicted y.

$$\sigma(z) = \frac{1}{1+e^{-z}}$$



Sigmoid function

- As shown in previous slide the sigmoid function converts the continuous variable data into the probability i.e. between 0 and 1.
  - $\sigma(z)$ tends towards 1 as $z \rightarrow \infty$
  - $\sigma(z)$ tends towards 0 as $z \rightarrow -\infty$
  - $\sigma(z)$ is always bounded between 0 and 1

where the probability of being a class can be measured as:

$P(y=1)=\sigma(z)$

$P(y=0)=1-\sigma(z)$

Logistic Regression Equation and Odds:

It models the odds of the dependent event occurring which is the ratio of the probability of the event to the probability of it not occurring:

$$\frac{p(x)}{1-p(x)} = e^z$$

Taking the natural logarithm of the odds gives the log-odds or logit:

$$\log\left[\frac{p(x)}{1-p(x)}\right] = z$$

$$\log\left[\frac{p(x)}{1-p(x)}\right] = w \cdot X + b$$

$$\frac{p(x)}{1-p(x)} = e^{w \cdot X + b} \quad \cdots \text{Exponentiate both sides}$$

$$p(x) = e^{w \cdot X + b} \cdot (1 - p(x))$$

$$p(x) = e^{w \cdot X + b} - e^{w \cdot X + b} \cdot p(x))$$

$$p(x) + e^{w \cdot X + b} \cdot p(x)) = e^{w \cdot X + b}$$

$$p(x)(1 + e^{w \cdot X + b}) = e^{w \cdot X + b}$$

$$p(x) = \frac{e^{w \cdot X + b}}{1 + e^{w \cdot X + b}}$$

- The final logistic regression equation will be:

$$p(X;b,w) = \frac{e^{w \cdot X + b}}{1 + e^{w \cdot X + b}} = \frac{1}{1 + e^{-w \cdot X + b}}$$

This formula represents the probability of the input belonging to Class 1.

Likelihood Function for Logistic Regressio

- The goal is to find weights w and bias b that maximize the likelihood of observing the data.

  For each data point i

  for y=1, predicted probabilities will be: p(X;b,w) =p(x)

  for y=0, predicted probabilities will be: 1-p(X;b,w) = 1−p(x)

$$L(b, w) = \prod_{i=1}^{n} p(x_i)^{y_i} (1 - p(x_i))^{1-y_i}$$

- Taking natural logs on both sides:

$$\log(L(b,w)) = \sum_{i=1}^{n} y_i \log p(x_i) \; + \; (1-y_i)\log(1-p(x_i))$$

$$= \sum_{i=1}^{n} y_i \log p(x_i) + \log(1-p(x_i)) - y_i \log(1-p(x_i))$$

$$= \sum_{i=1}^{n} \log(1-p(x_i)) + \sum_{i=1}^{n} y_i \log \frac{p(x_i)}{1-p(x_i}$$

$$= \sum_{i=1}^{n} -\log 1 - e^{-(w\cdot x_i+b)} + \sum_{i=1}^{n} y_i(w\cdot x_i+b)$$

$$= \sum_{i=1}^{n} -\log\left(1 + e^{w\cdot x_i+b}\right) + \sum_{i=1}^{n} y_i(w\cdot x_i+b)$$

  - This is known as the log-likelihood function.

Gradient of the log-likelihood function

- To find the best w and b we use gradient ascent on the log-likelihood function. The gradient with respect to each weight wj is:

$$\frac{\partial J(l(b,w)}{\partial w_j} = -\sum_{i=n}^{n} \frac{1}{1+e^{w\cdot x_i+b}} e^{w\cdot x_i+b} x_{ij} + \sum_{i=1}^{n} y_i x_{ij}$$

$$= -\sum_{i=n}^{n} p(x_i;b,w) x_{ij} + \sum_{i=1}^{n} y_i x_{ij}$$

$$= \sum_{i=n}^{n} (y_i - p(x_i;b,w)) x_{ij}$$

$$\frac{d}{dw}\left[-\log\left(1+e^{wx+b}\right)\right]$$

Step 1 — Outer function derivative (chain rule)

Let:

$$u = 1 + e^{wx+b}$$

We know:

$$\frac{d}{du}[-\log(u)] = -\frac{1}{u}$$

So applying chain rule:

$$\frac{d}{dw}[-\log(u)] = -\frac{1}{u} \cdot \frac{du}{dw}$$

Step 2 — Derivative of $u = 1 + e^{wx+b}$

The constant 1 vanishes when differentiating:

$$\frac{du}{dw} = e^{wx+b} \cdot \frac{d}{dw}(wx + b)$$

Derivative of $wx + b$ w.r.t. w is x:

$$\frac{du}{dw} = e^{wx+b} \cdot x$$

Step 3 — Combine results

Substitute back into the chain rule result:

$$\frac{d}{dw}\left[-\log(1+e^{wx+b})\right] = -\frac{1}{1+e^{wx+b}} \cdot e^{wx+b} \cdot x$$

Step 4 — Simplify

We can write:

$$\frac{d}{dw}\left[-\log(1+e^{wx+b})\right] = -\frac{e^{wx+b}}{1+e^{wx+b}} \cdot x$$

THANK YOU