Logistic Regression In PYTHON

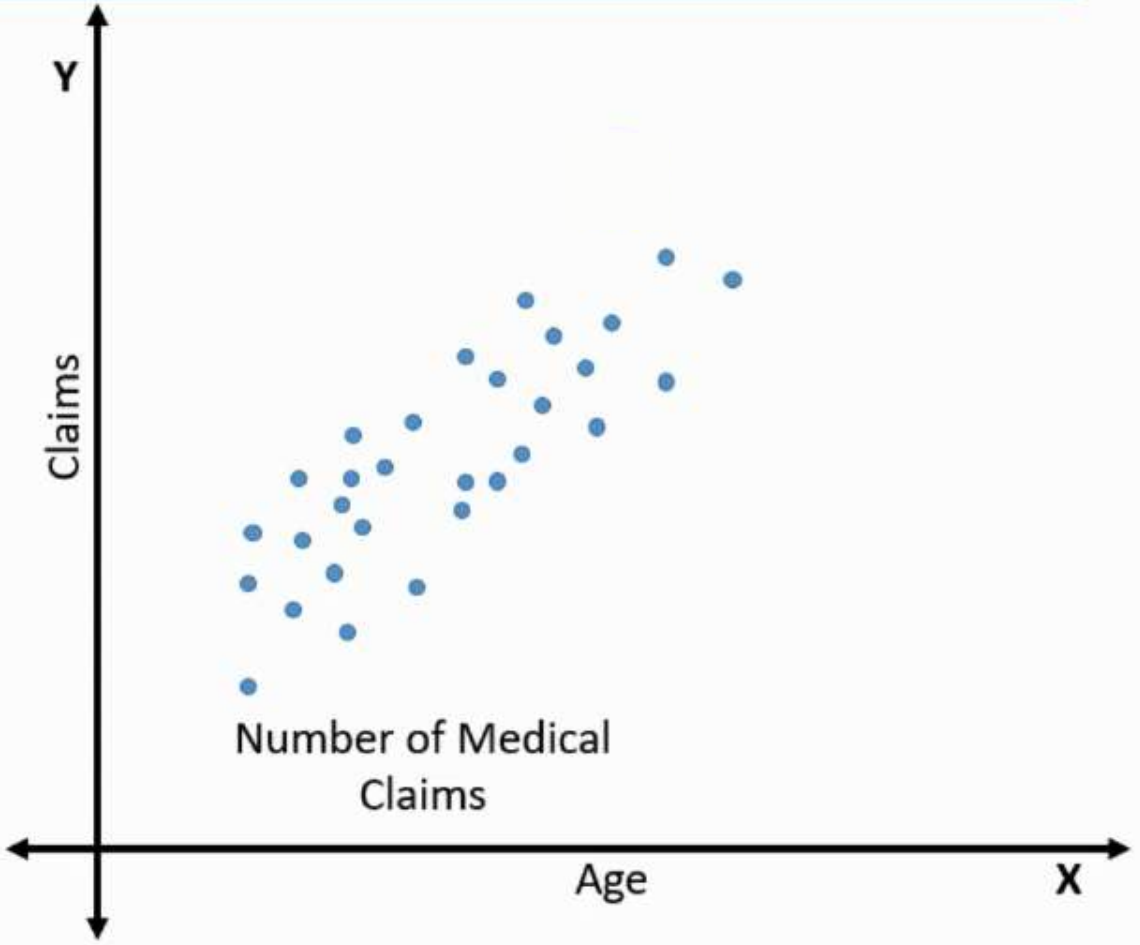
What is Logistics Regression?

- Used to predict the probability of an outcome
- Can be binary – Yes/No or Multiple
- Supervised learning method
- Must provide a dataset that already contains the outcomes to train the model.

Understanding the Logistic Regression

$$y = b_0 + \ b_1 \, x$$

$$No\ of\ Claims = 18 + b_1 * (age)$$

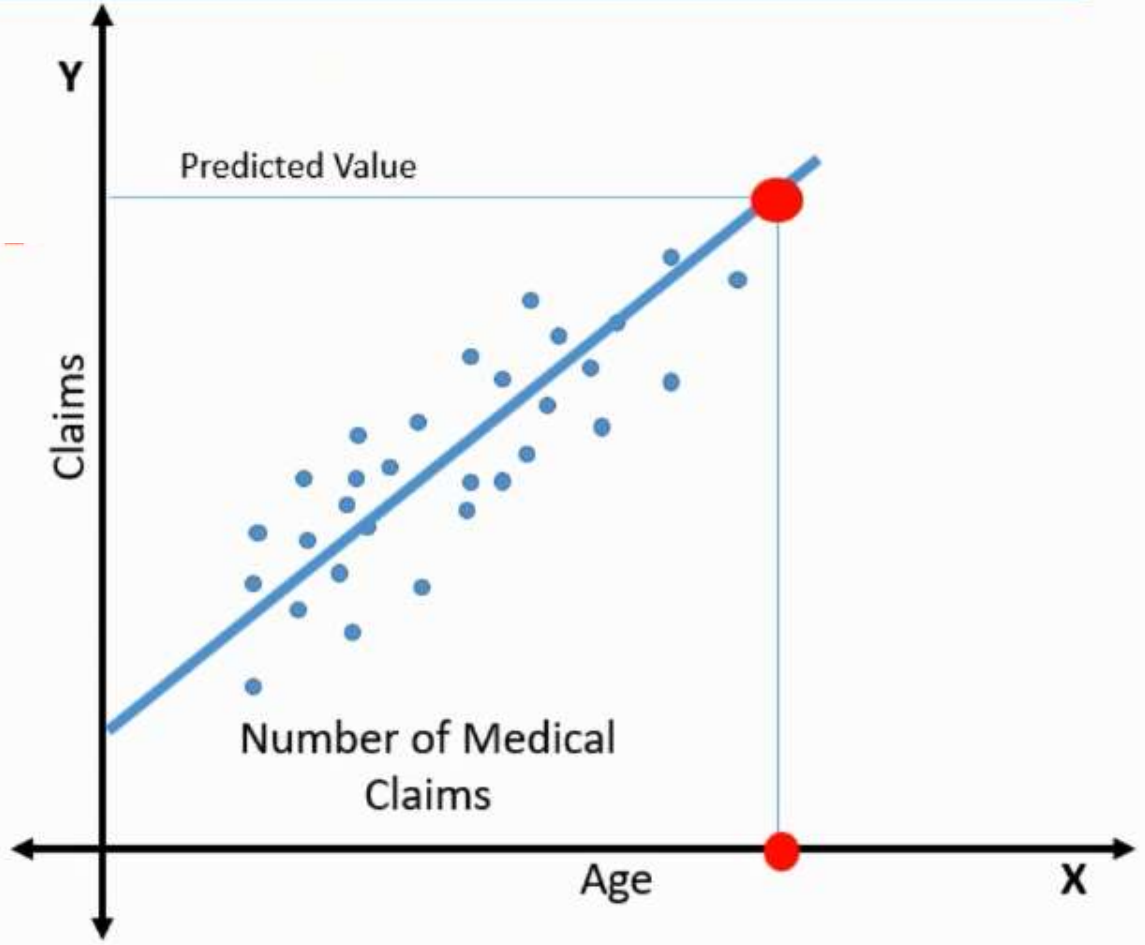


Simple Linear Regression

Understanding the Logistic Regression

$$y = b_0 + \ b_1 \, x$$

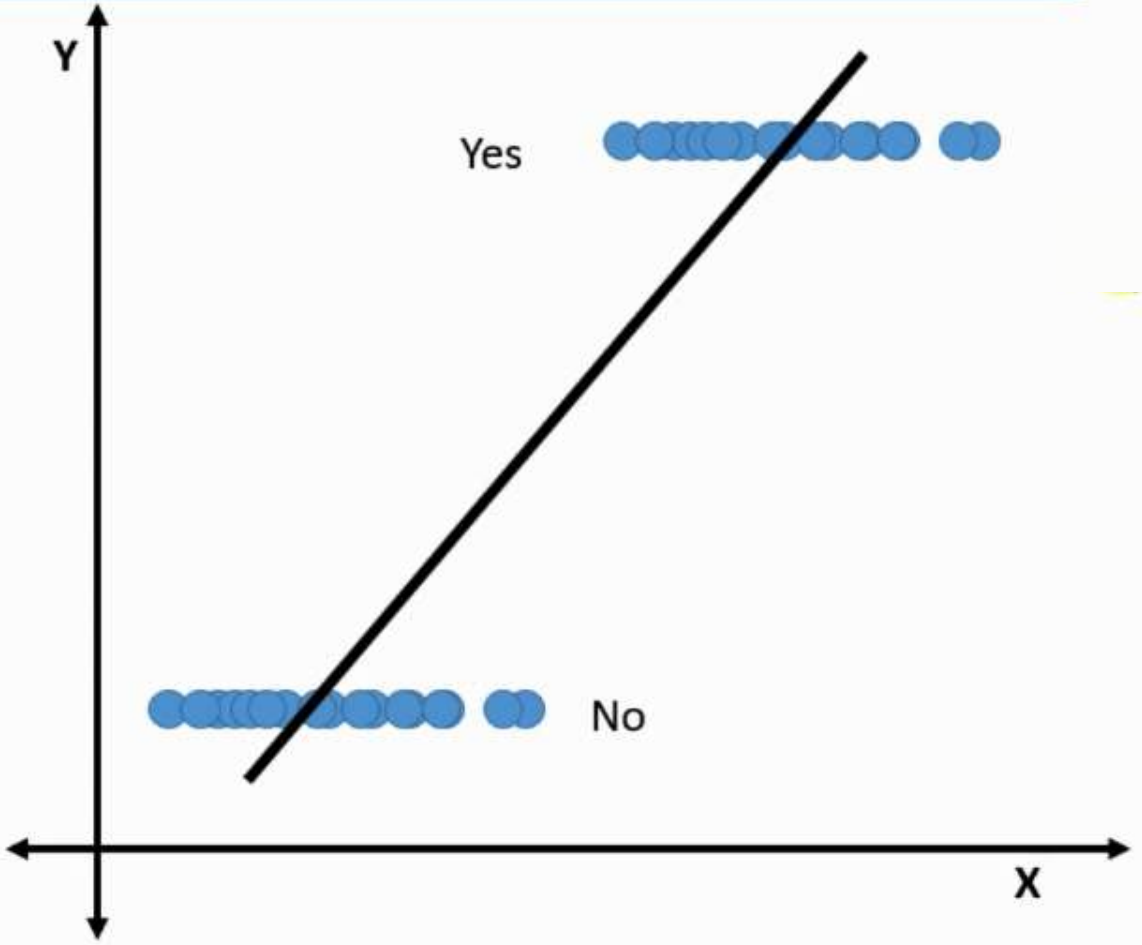
$$No\ of\ Claims = 18 + b_1 * (age)$$



Simple Linear Regression

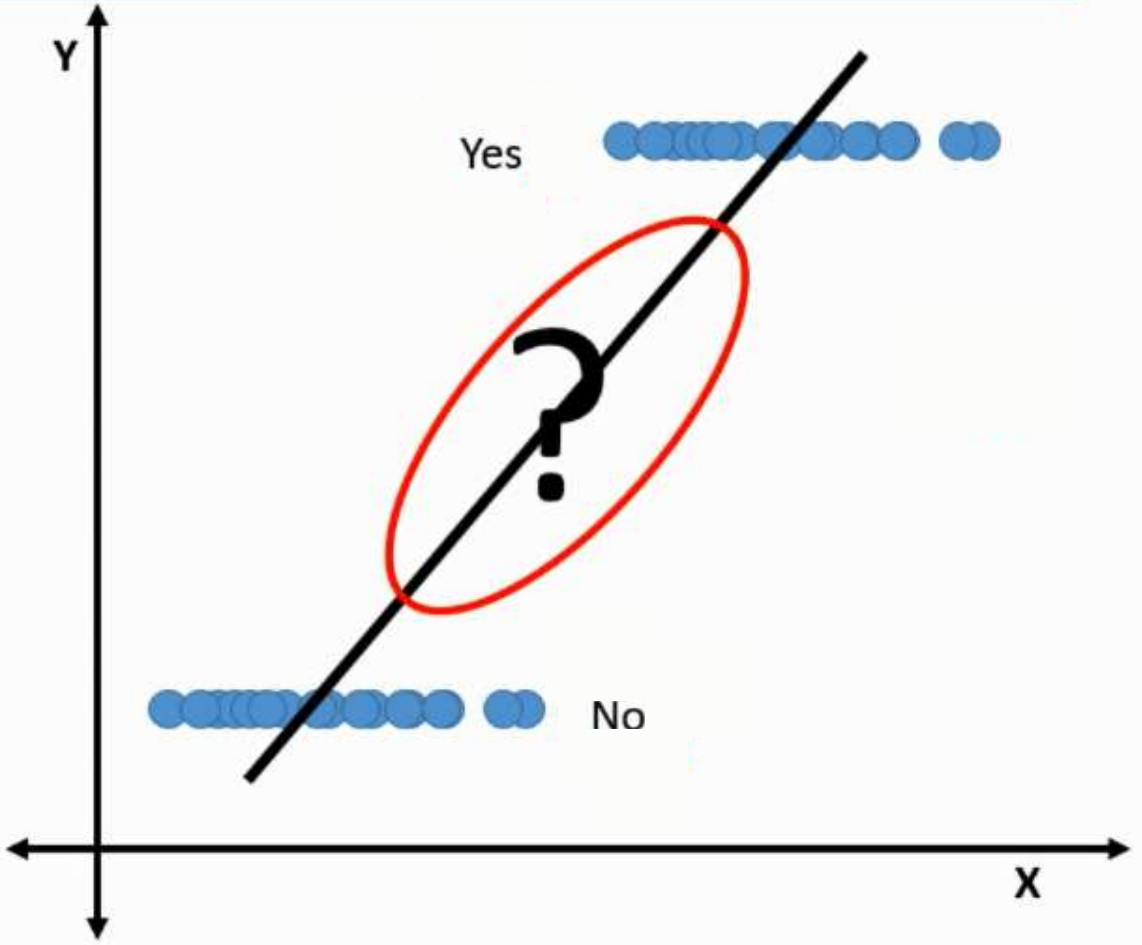
Logistic Regression?

- Outcome is categorical
- Will this customer buy my product?



Logistic Regression?

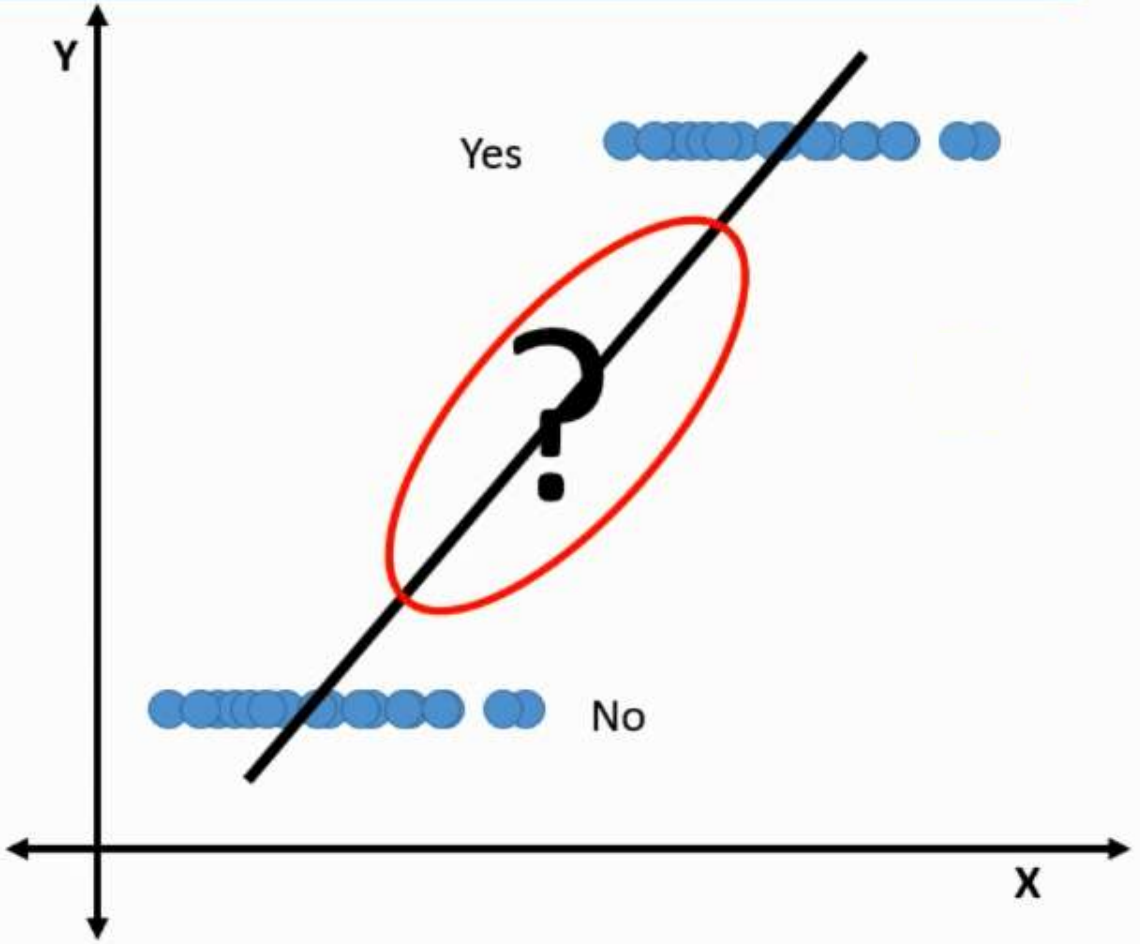
- Outcome is categorical
- Will this customer buy my product?



Logistic Regression?

- Outcome is categorical

- What is the probability of this customer buying this product?



Logistic Regression?

- Probability needs to satisfy two basic conditions
  - Always positive i.e. > 0
  - Always less than or equal to 1

$$y = b_0 + b_1 x \xrightarrow{\text{Always Positive}} e^{y} \xrightarrow{\text{Make it less than 1}} \frac{e^{y}}{e^{y} + 1}$$

Logistic Regression?

$$P = \frac{e^{y}}{e^{y} + 1}$$

$$Q = 1 - P = 1 - \frac{e^{y}}{e^{y} + 1} = \frac{e^{y} + 1 - e^{y}}{e^{y} + 1} = \frac{1}{e^{y} + 1}$$

Probability of Failure

Logistic Regression?

$$P = \frac{e^{y}}{e^{y} + 1}$$

$$1 - P = \frac{1}{e^{y} + 1}$$

$$\text{Odds} = \frac{P}{1 - P} = \frac{\frac{e^{y}}{e^{y} + 1}}{\frac{1}{e^{y} + 1}} = e^{y}$$

Logistic Regression?

$$P = \frac{e^{y}}{e^{y} + 1}$$

$$1 - P = \frac{1}{e^{y} + 1}$$

$$\frac{P}{1 - P} = e^{y}$$

Logistic Regression?

$$P = \frac{e^{y}}{e^{y} + 1}$$

$$1 - P = \frac{1}{e^{y} + 1}$$

$$\log\left(\frac{P}{1 - P}\right) = y$$

Logistic Regression?

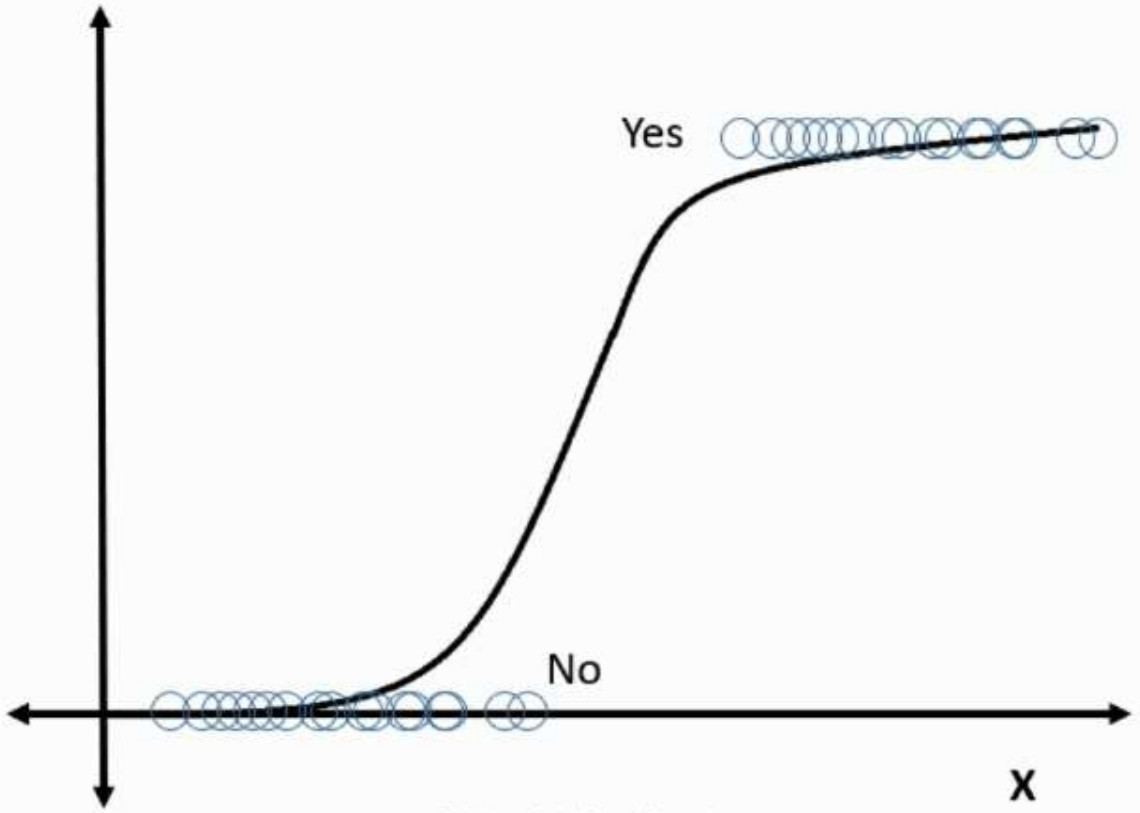
$$P = \frac{e^{y}}{e^{y} + 1}$$

$$1 - P = \frac{1}{e^{y} + 1}$$

$$\log\left(\frac{P}{1-P}\right) = y = (b_0 + b_1 x)$$

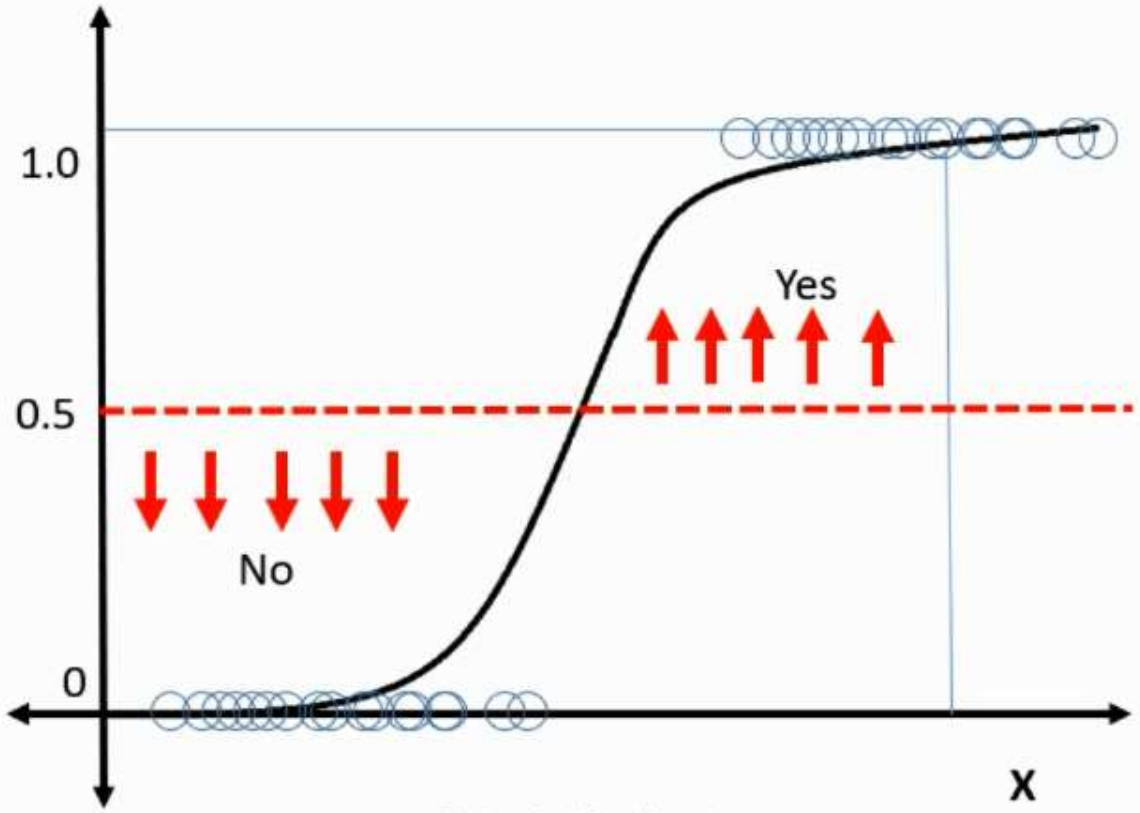
Plotting Logistic Regression

$$\log\left(\frac{P}{1-P}\right) = (b_0 + b_1 x)$$



Plotting Logistic Regression

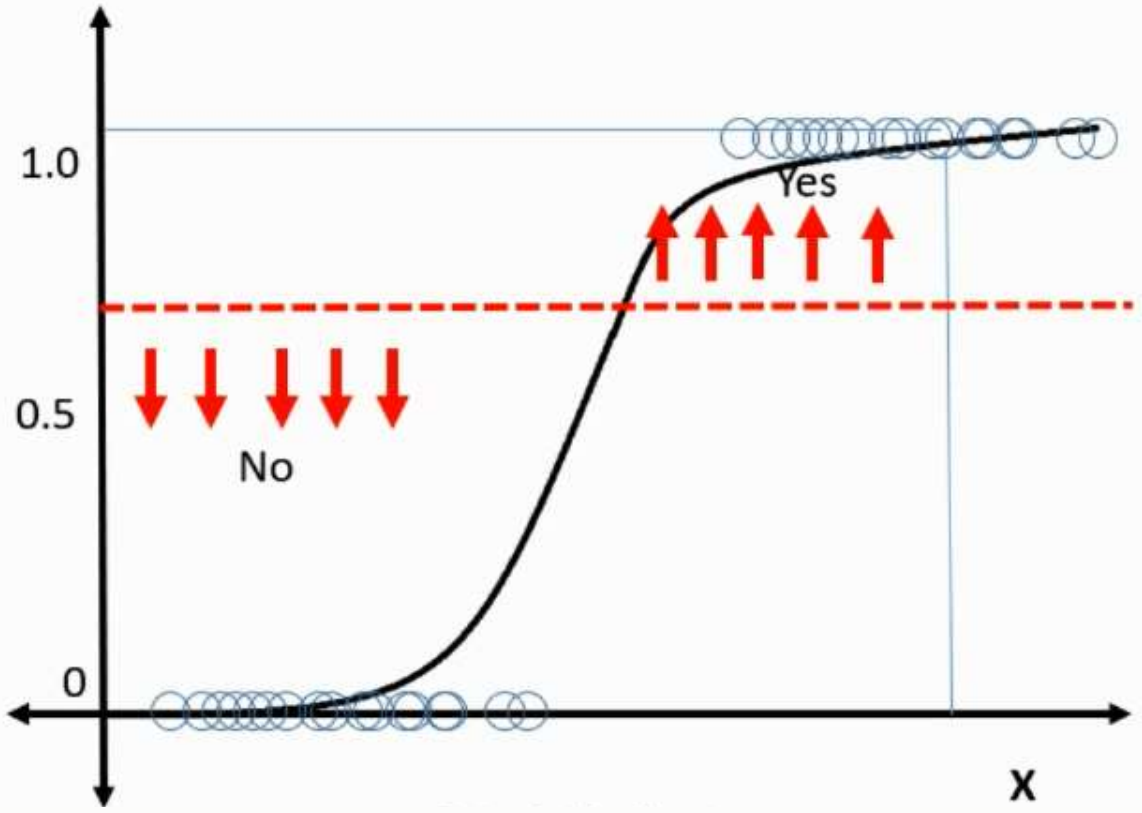
$$\log\left(\frac{P}{1-P}\right) = (b_0 + b_1 x)$$



Threshold Value of 50% probability
> 0.5 Success
< 0.5 Failure

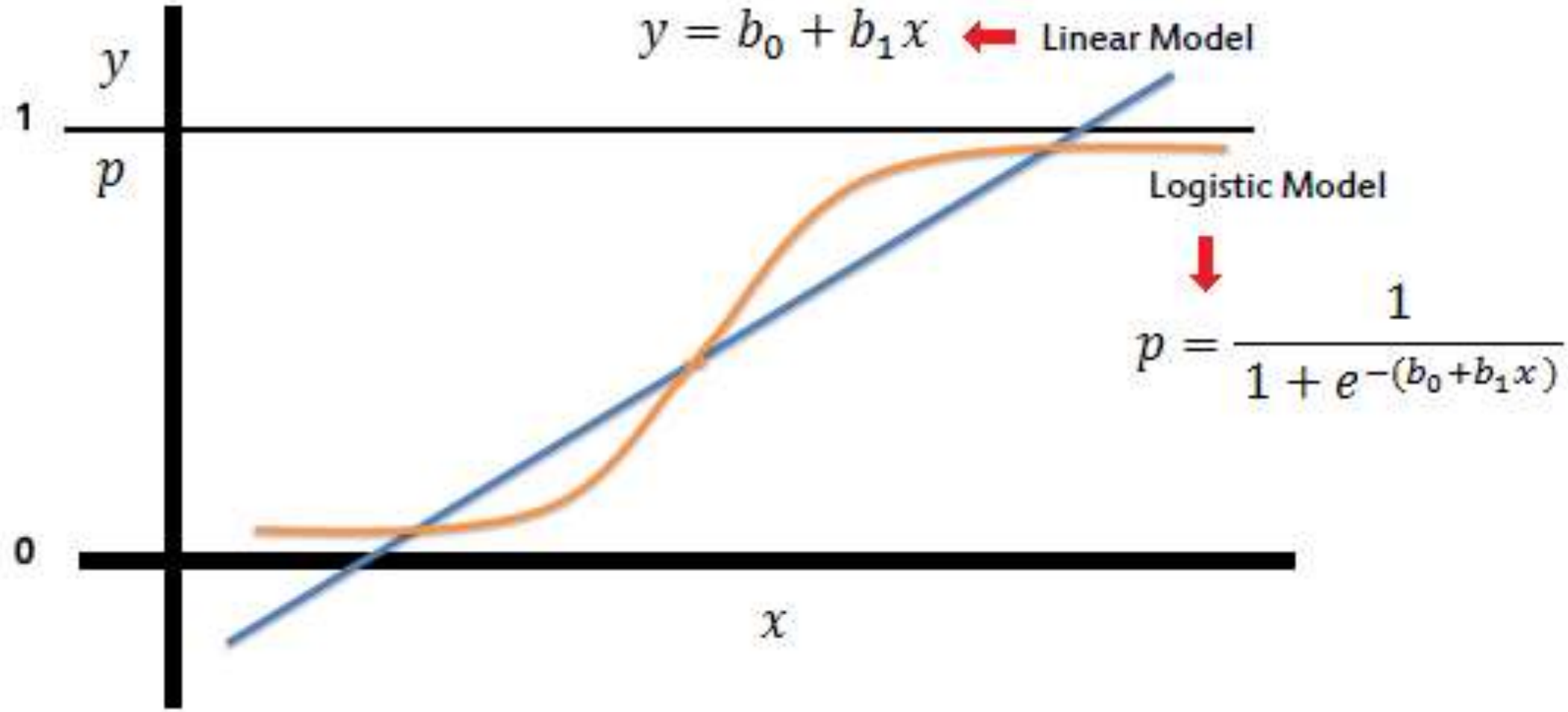
Plotting Logistic Regression

$$\log\left(\frac{P}{1-P}\right) = (b_0 + b_1 x)$$



Threshold Value of 70% probability
> 0.7 Success
< 0.7 Failure

Logistic Regression Equation



Demo : Create ML for predicting if the customer will buy the product or not ?

Demo 2: Create ML for predicting Bank Churn .

Assignment : Create ML for automating Loan Approval process.

Loan Approval Prediction

Loan_ID	Gender	Married	Dependents	Self_Employed	Income	LoanAmt	Term	CreditHistory	Property_Area	Status
LP001002	Male	No	0	No	$5,849.00		60	1	Urban	Y
LP001003	Male	Yes	1	No	$4,583.00	$128.00	120	1	Rural	N
LP001005	Male	Yes	0	Yes	$3,000.00	$66.00	60	1	Urban	Y
LP001006	Male	Yes	2	No	$2,583.00	$120.00	60	1	Urban	Y

- Automate loan eligibility process
- Identify customers whose loan will be approved

Loan Approval Prediction

Loan_ID	Gender	Married	Dependents	Self_Employed	Income	LoanAmt	Term	CreditHistory	Property_Area	Status
LP001002	Male	No	0	No	$5,849.00		60	1	Urban	Y
LP001003	Male	Yes	1	No	$4,583.00	$128.00	120	1	Rural	N
LP001005	Male	Yes	0	Yes	$3,000.00	$66.00	60	1	Urban	Y
LP001006	Male	Yes	2	No	$2,583.00	$120.00	60	1	Urban	Y

- Automate loan eligibility process
- Identify customers whose loan will be approved

gender	married	ch	income	loanamt	status
Male	No	1	5849		Y
Male	Yes	0	4583	128	N
Male	Yes	1	3000	66	Y
Female	Yes	1	2583	120	Y
Male	No	1	6000	141	Y
Male	Yes	1	5417	267	Y

Solution Steps...

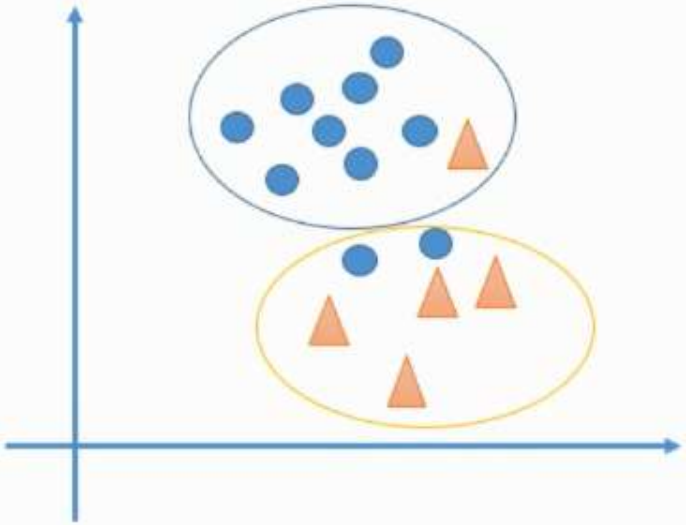
- Import Libraries and read data
- Identify and Deal with Missing Values
- Create Dummy Variables
- Normalise the Data
- Select relevant columns
- Split dataset in Training and Test datasets
- Train and Evaluate the model

gender	married	ch	income	loanamt	status
Male	No	1	5849		Y
Male	Yes	0	4583	128	N
Male	Yes	1	3000	66	Y
Female	Yes	1	2583	120	Y
Male	No	1	6000	141	Y
Male	Yes	1	5417	267	Y

Prediction Outcome

	Predicted Negatives	Predicted Positives
Actual Negatives	True Negatives	False Positives
Actual Positives	False Negatives	True Positives

	Predicted Negatives	Predicted Positives	
Actual Negatives	4	1	5
Actual Positives	2	8	10
	6	9	



= Negative

= Positive

Assignment 3: Create ML for Titanic dataset to predict who survived ?

Quiz

Question 1:

Logistic Regression is a linear classifier

- True
- False

Question 2:

Logistic Regression returns probabilities

- True
- False

Question 3:

In Python, what is the class used to create a logistic regression classifier ?

- ○ GLM
- ○ LogisticRegression
- ○ Logit
- ○ LogReg

Evaluate Classification Models

Model Evaluation

	0	1
0	29	20
1	2	108

Accuracy = 0.86

	0	1
0	3814	559
1	800	764

Accuracy = 0.77

Loan Approval Prediction

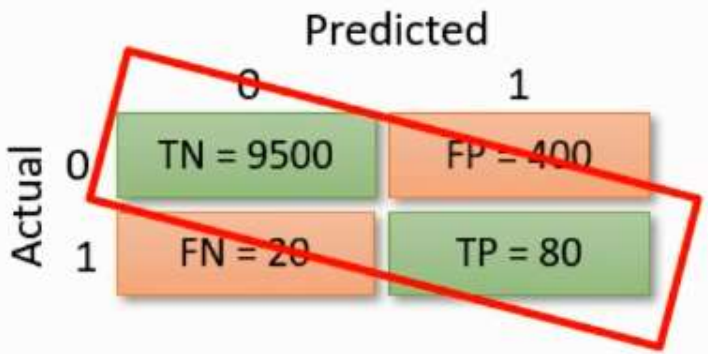
Loan_ID	Gender	Married	Dependents	Self_Employed	Income	LoanAmt	Term	CreditHistory	Property_Area	Status
LP001002	Male	No	0	No	$5,849.00		60	1	Urban	Y
LP001003	Male	Yes	1	No	$4,583.00	$128.00	120	1	Rural	N
LP001005	Male	Yes	0	Yes	$3,000.00	$66.00	60	1	Urban	Y
LP001006	Male	Yes	2	No	$2,583.00	$120.00	60	1	Urban	Y

- Automate loan eligibility process
- Identify customers whose loan will be approved

gender	married	ch	income	loanamt	status
Male	No	1	5849		Y
Male	Yes	0	4583	128	N
Male	Yes	1	3000	66	Y
Female	Yes	1	2583	120	Y
Male	No	1	6000	141	Y
Male	Yes	1	5417	267	Y

Accuracy

Fraud Detection – 0.958



Accuracy – Proportions of total number of correct results

$$Accuracy = \frac{TN + TP}{Total\ Observations} = \frac{9500 + 80}{10000} = 0.958$$

Null – 0.99

$$Accuracy = \frac{TN + TP}{Total\ Observations} = \frac{9900 + 1}{10000} = 0.99$$

Precision and Recall

Fraud Detection – 0.958

Actual \ Predicted	0	1
0	TN = 9500	FP = 400
1	FN = 20	TP = 80

Null – 0.99

Actual \ Predicted	0	1
0	TN = 9900	FP = 0
1	FN = 99	TP = 1

Precision – Proportion of correct positive results out of all predicted positive results.

Recall/Sensitivity – Proportion of actual positive cases

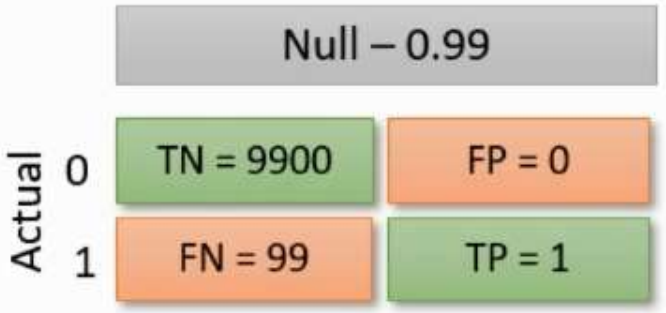
$$Precision = \frac{TP}{TP + FP} = \frac{80}{80 + 400} = 0.167$$

$$Recall = \frac{TP}{TP + FN} = \frac{80}{80 + 20} = 0.8$$

$$Precision = \frac{1}{1 + 0} = 1$$

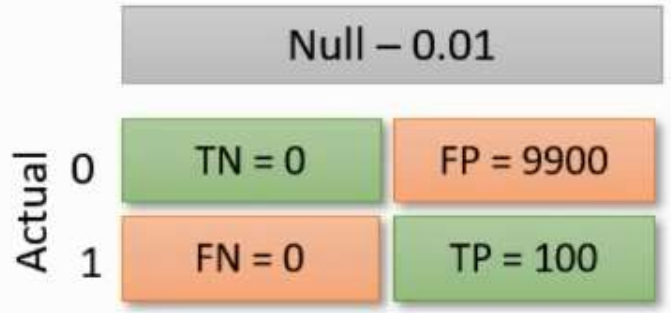
$$Recall = \frac{1}{1 + 99} = 0.01$$

Precision and Recall



$$Precision = \frac{1}{1+0} = 1$$

$$Recall = \frac{1}{1+99} = 0.01$$



$$Precision = \frac{100}{100+9900} = 0.01$$

$$Recall = \frac{100}{100+0} = 1$$

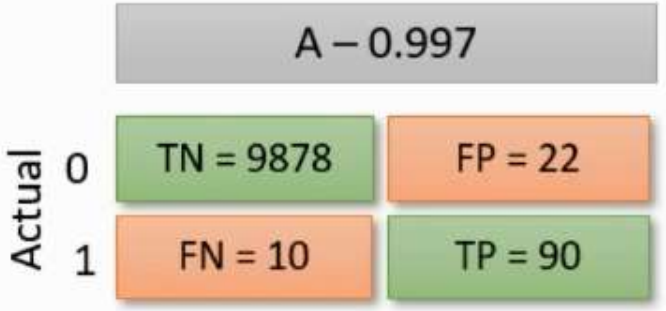
Best Case Scenario

$$Precision = \frac{100}{100 + 0} = 1$$

$$Recall = \frac{100}{100 + 0} = 1$$

$$Accuracy = \frac{9900 + 100}{10000} = 1$$

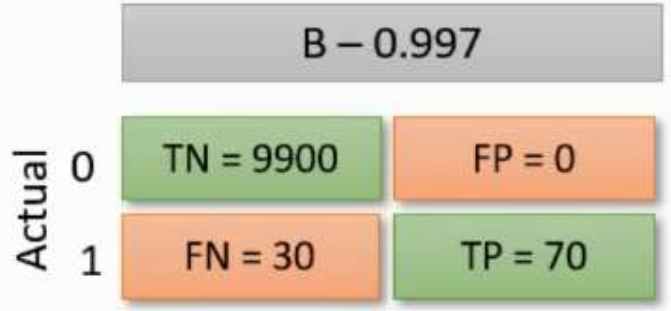
F1Score



$$Precision = \frac{90}{90 + 22} = 0.8035$$

$$Recall = \frac{90}{90 + 10} = 0.9$$

$$F1Score = \frac{2 * 0.8035 * 0.9}{0.8035 + 0.9} = \mathbf{0.849}$$



$$Precision = \frac{70}{70 + 0} = 1$$

$$Recall = \frac{70}{70 + 30} = 0.7$$

$$F1Score = \frac{2 * 1 * 0.7}{1 + 0.7} = \mathbf{0.823}$$

Recap

$$Accuracy = \frac{TN + TP}{Total\ Observations}$$

$$Precision = \frac{TP}{TP + FP}$$

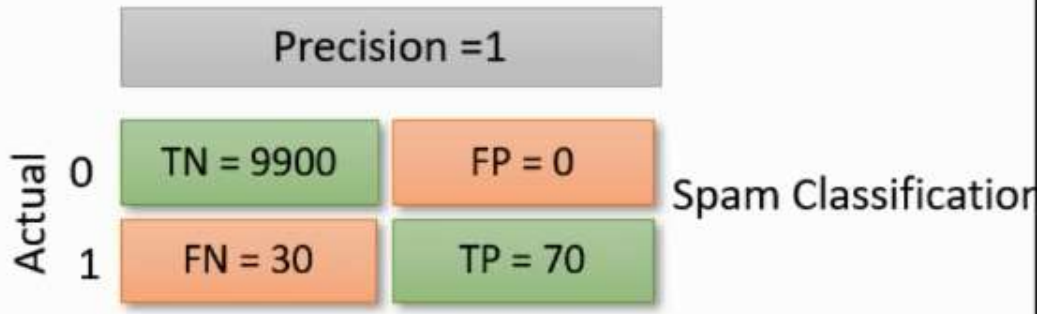
$$F1Score = \frac{2 * Precision * Recall}{Precision + Recall}$$

$$Recall\ or\ Sensitivity = \frac{TP}{TP + FN}$$

True Positive Rate

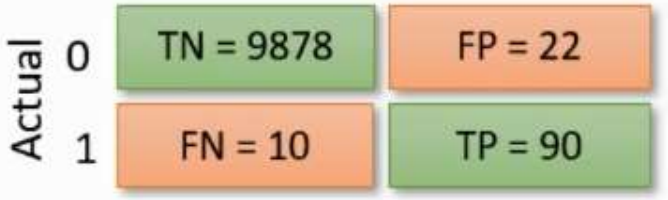
Which metric to use?

- High Accuracy is nice to have
- High Precision when its OK to have false negatives

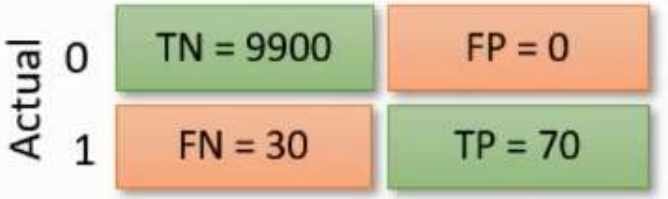


Which metric to use?

- High Accuracy is nice to have
- High Precision when its OK to have false negatives
- High recall or sensitivity when cost of false negative is very high



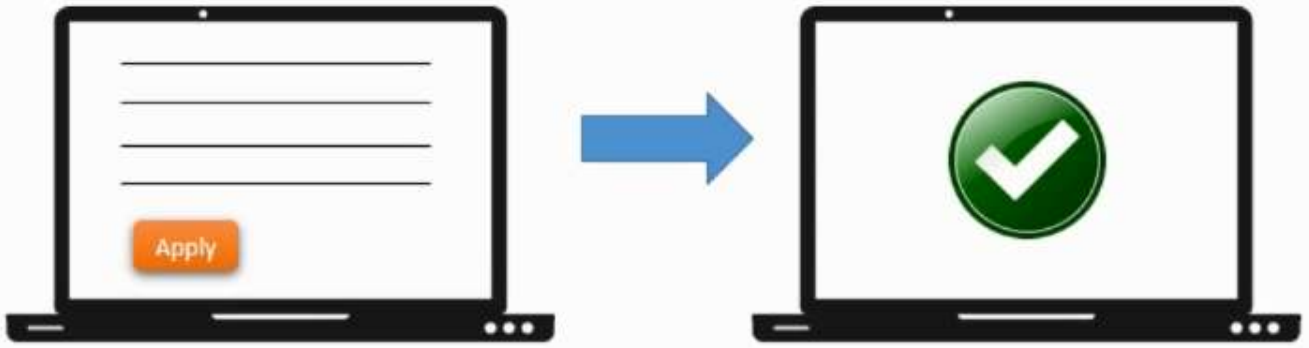
Loan Default



Recommendation

Loan Approval Prediction

Actual \ Predicted	0	1
0	TN = 29	FP = 20
1	FN = 2	TP = 108



$Accuracy = 0.8616$

$Precision = 0.84375$

$Recall\ or\ Sensitivity = 0.9818$

16% Bad Loans

Loan Approval Prediction

Actual \ Predicted	0	1
0	TN = 29	FP = 20
1	FN = 2	TP = 108

$$Accuracy = 0.8616$$

$$Precision = 0.84375$$

$$Recall\ or\ Sensitivity = 0.9818$$

Loan Approval Prediction

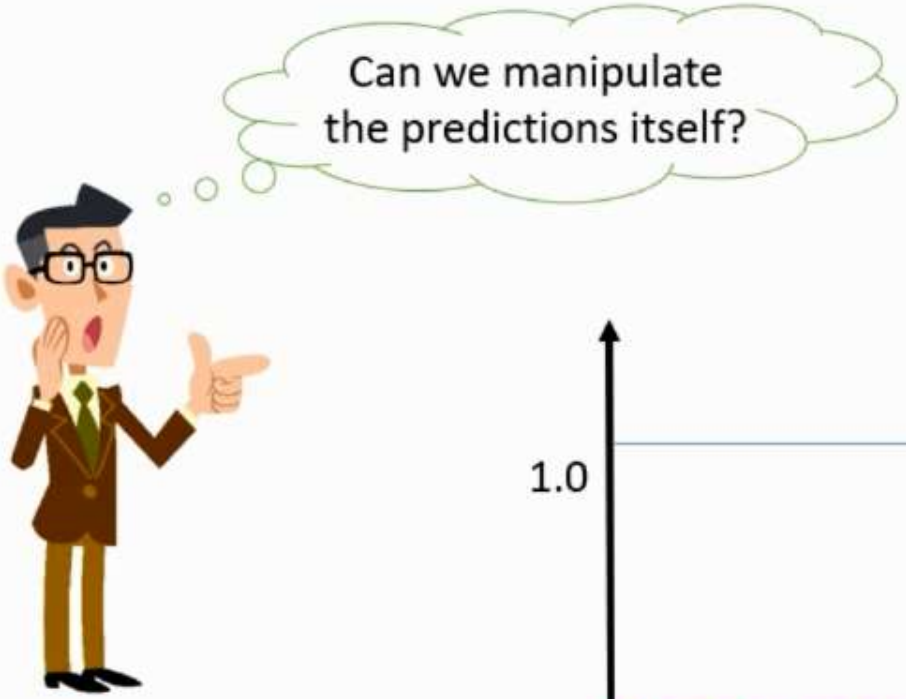
Actual \ Predicted	0	1
0	TN = 29	FP = 20
1	FN = 2	TP = 108

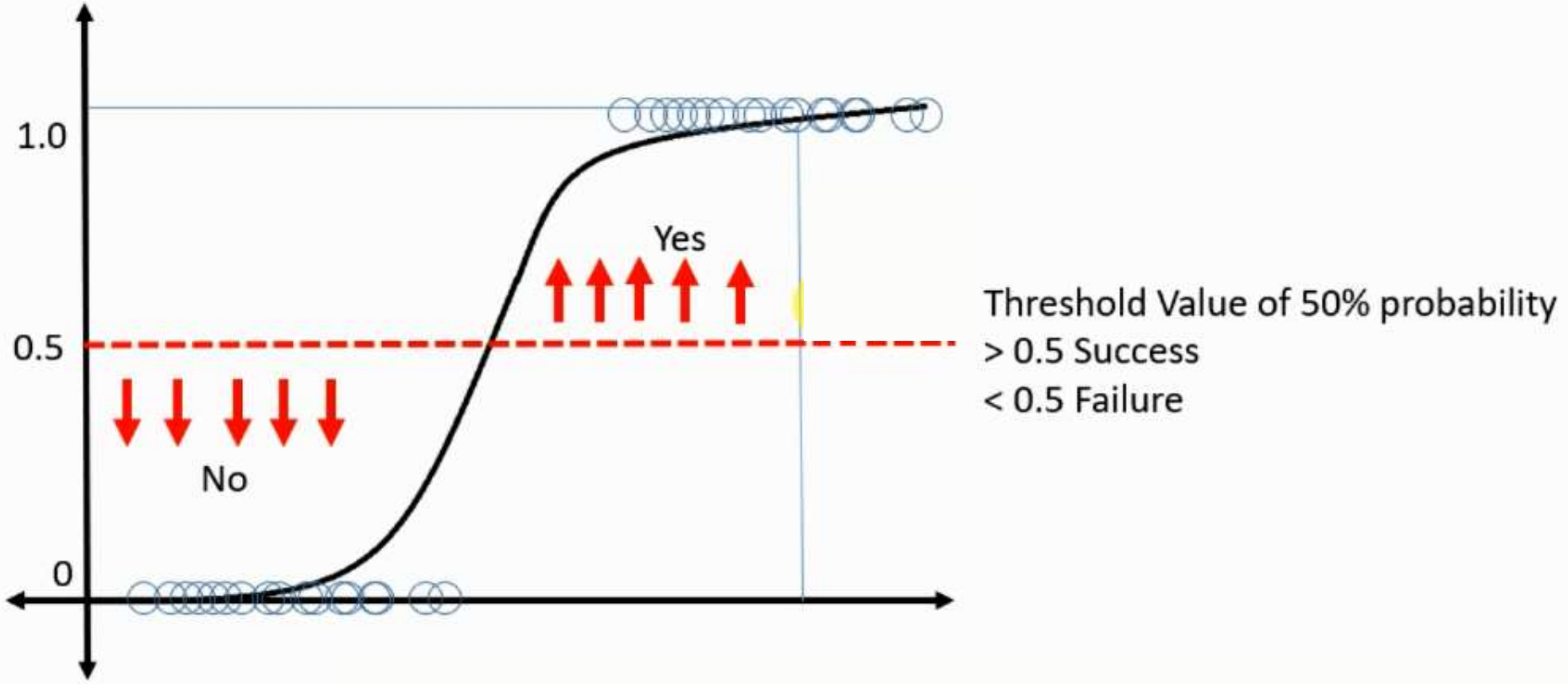
$$Accuracy = 0.8616$$

$$Precision = 0.84375$$

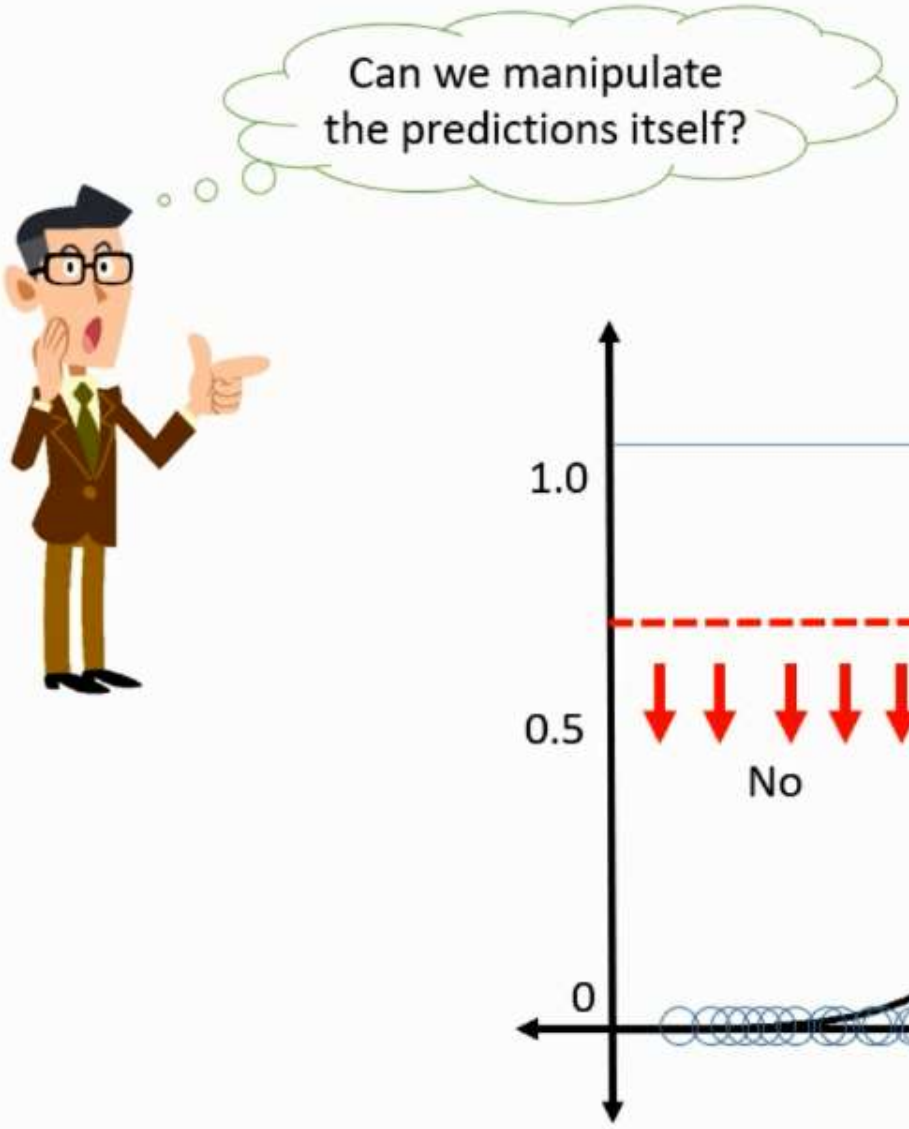
$$Recall\ or\ Sensitivity = 0.9818$$

Customise the metrics

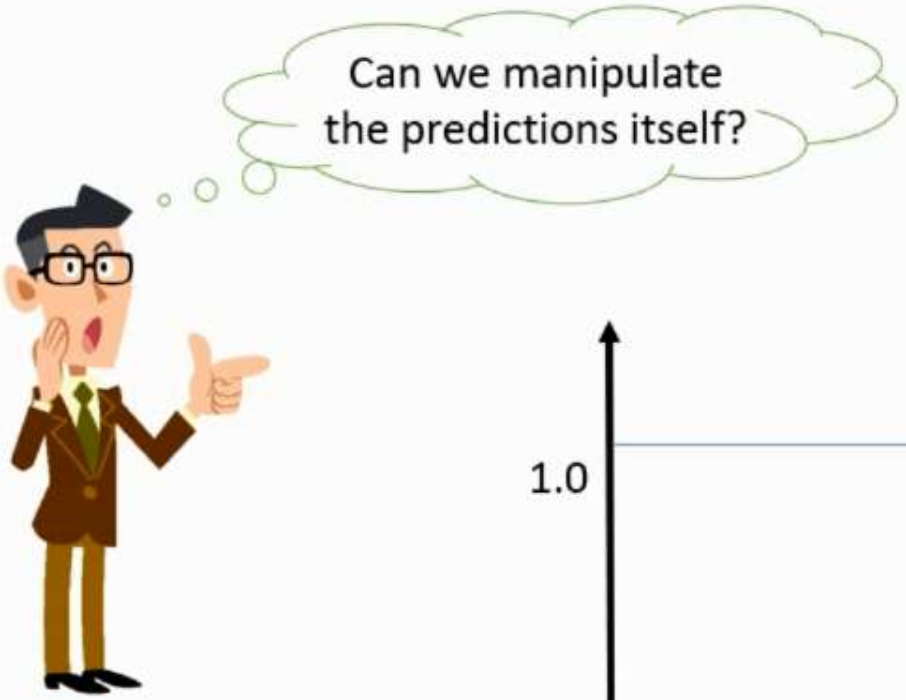


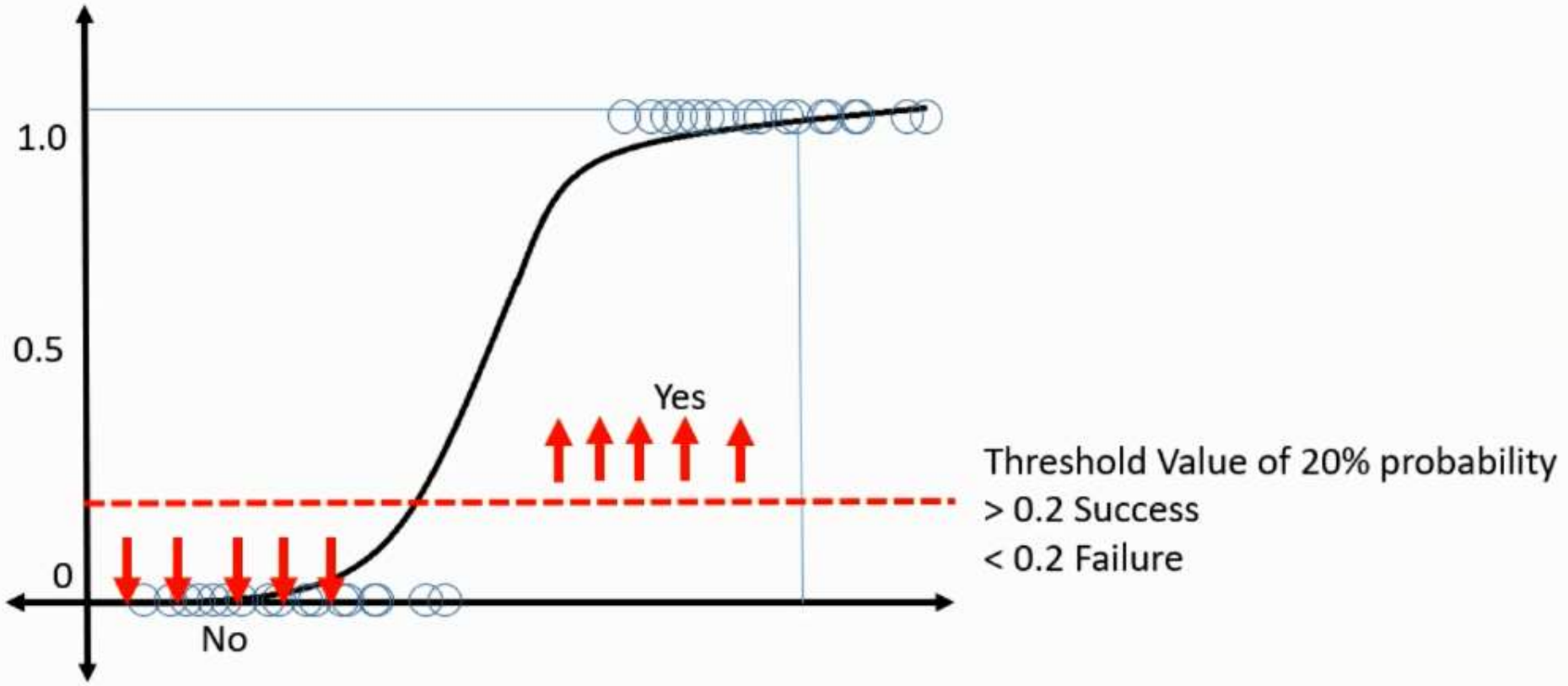


Customise the metrics



Customise the metrics

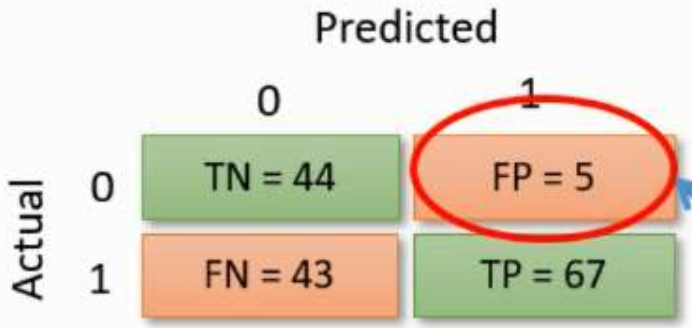




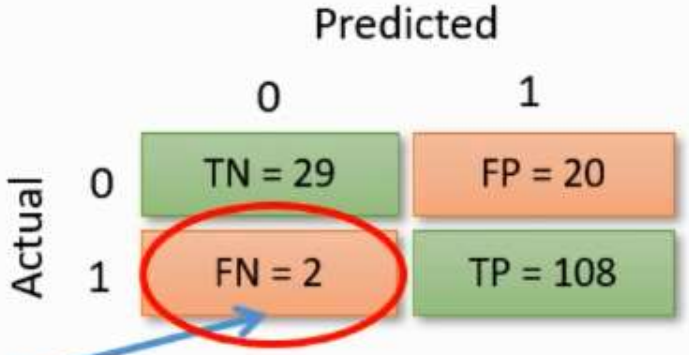
Demo: Evaluation Metrics for Loan Prediction projects

Demo: Adjusting Thresholds

Final 78.5% probability



With 60% probability



2 loss of opportunity and
5 bad loans

1. Anything above 78.5% probability – Approve — 72 Confirmed approvals
2. Anything below 60% probability – Reject — 31 Confirmed rejections
3. Anything between 60-78.5% – On Hold — 56 on Hold with manual checks

Final 78.5% probability

Actual \ Predicted	0	1
0	TN = 44	FP = 5
1	FN = 43	TP = 67

Adjusted Probability Threshold

2 loss o unity and
5 had l

With 60% probability

Actual \ Predicted	0	1
0	TN = 29	FP = 20
1	FN = 2	TP = 108

1. Anything above 78.5% probability – Approve pprovals
2. Anything below 60% probability – Reject ne ections
3. Anything between 60-78.5% – On Hold d with manual checks

KNN Algorithm

using Python

What is a K-Nearest Neighbor Algorithm?

KNN falls in the supervised learning family of an algorithm.

Informally this means that we are given a labelled dataset consisting of (x,y)

kNN is a simple algorithm that stores all available cases and classifies new cases based on a similarity measure (eg distance function).

How it Work?

STEP 1: Choose the number K of neighbors

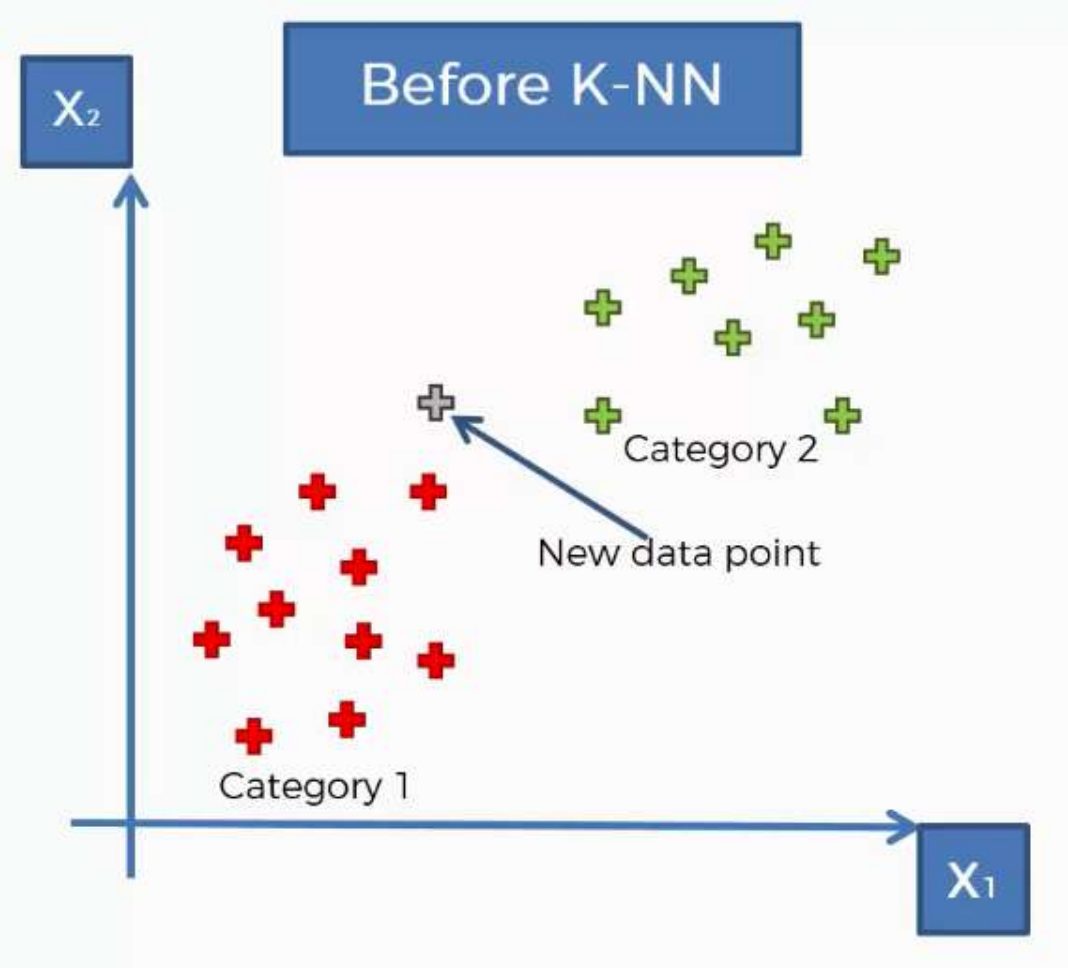
STEP 2: Take the K nearest neighbors of the new data point, according to the Euclidean distance

STEP 3: Among these K neighbors, count the number of data points in each category

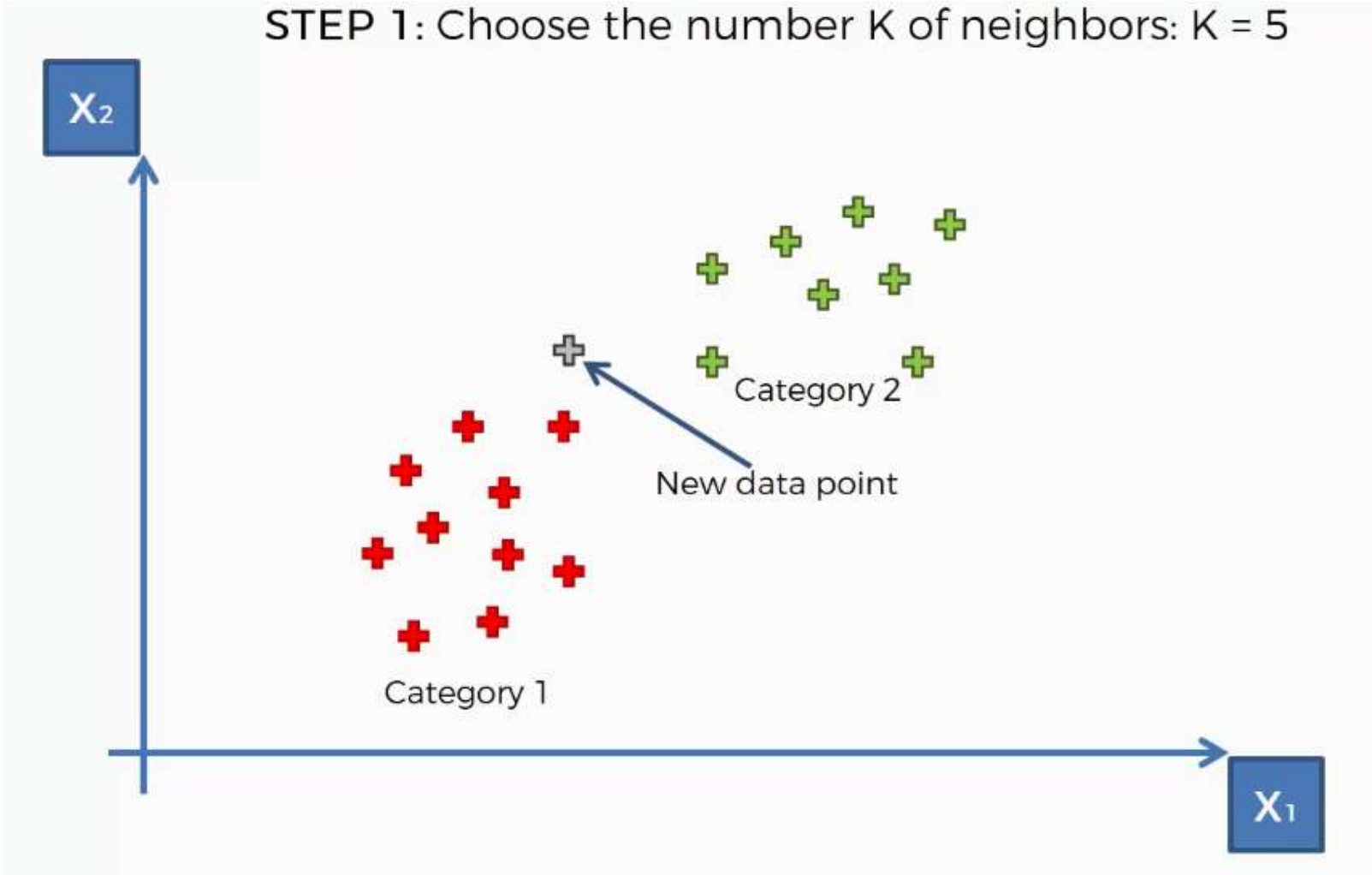
STEP 4: Assign the new data point to the category where you counted the most neighbors

Your Model is Ready

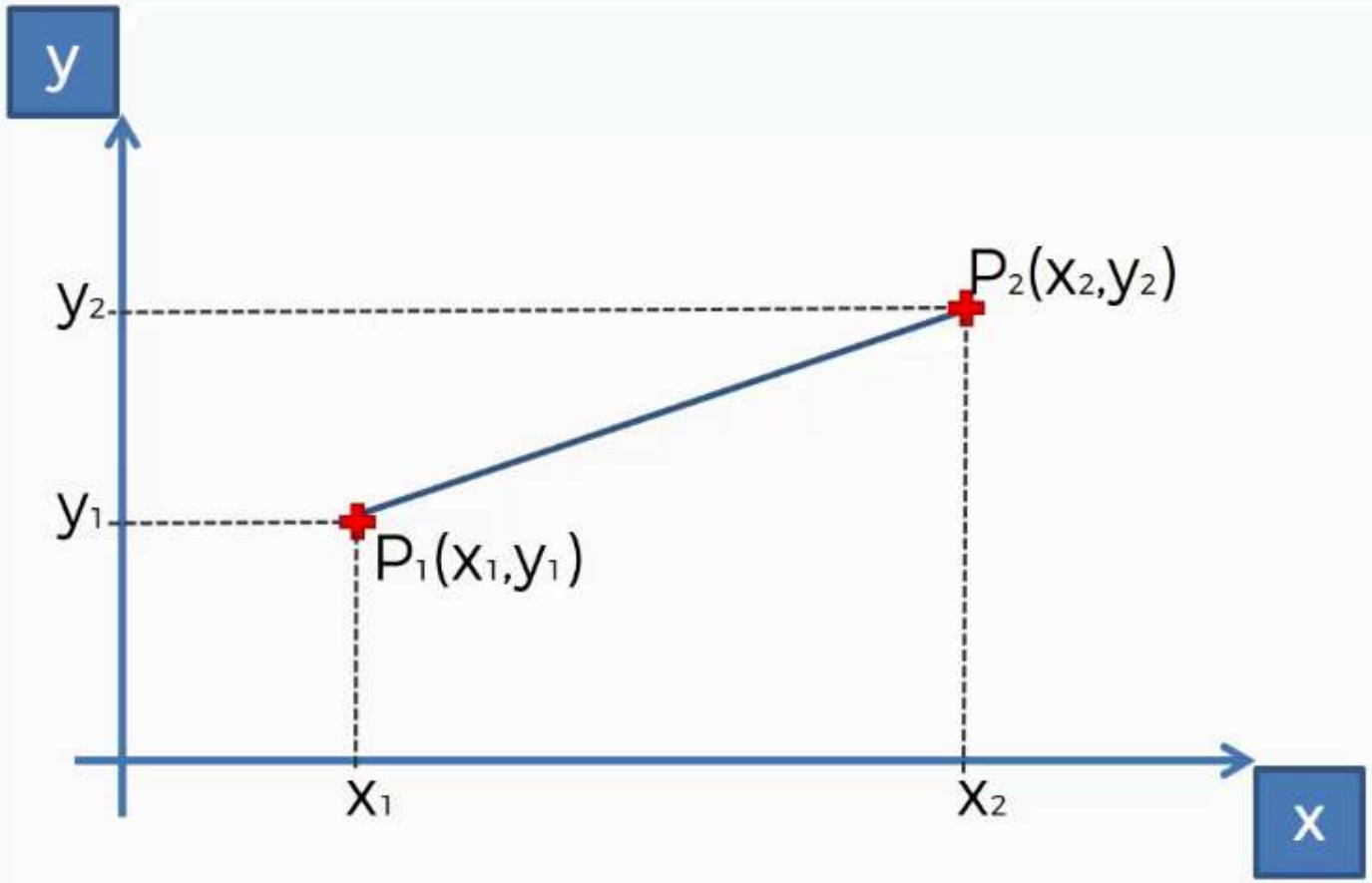
How it Work?



How it Work?

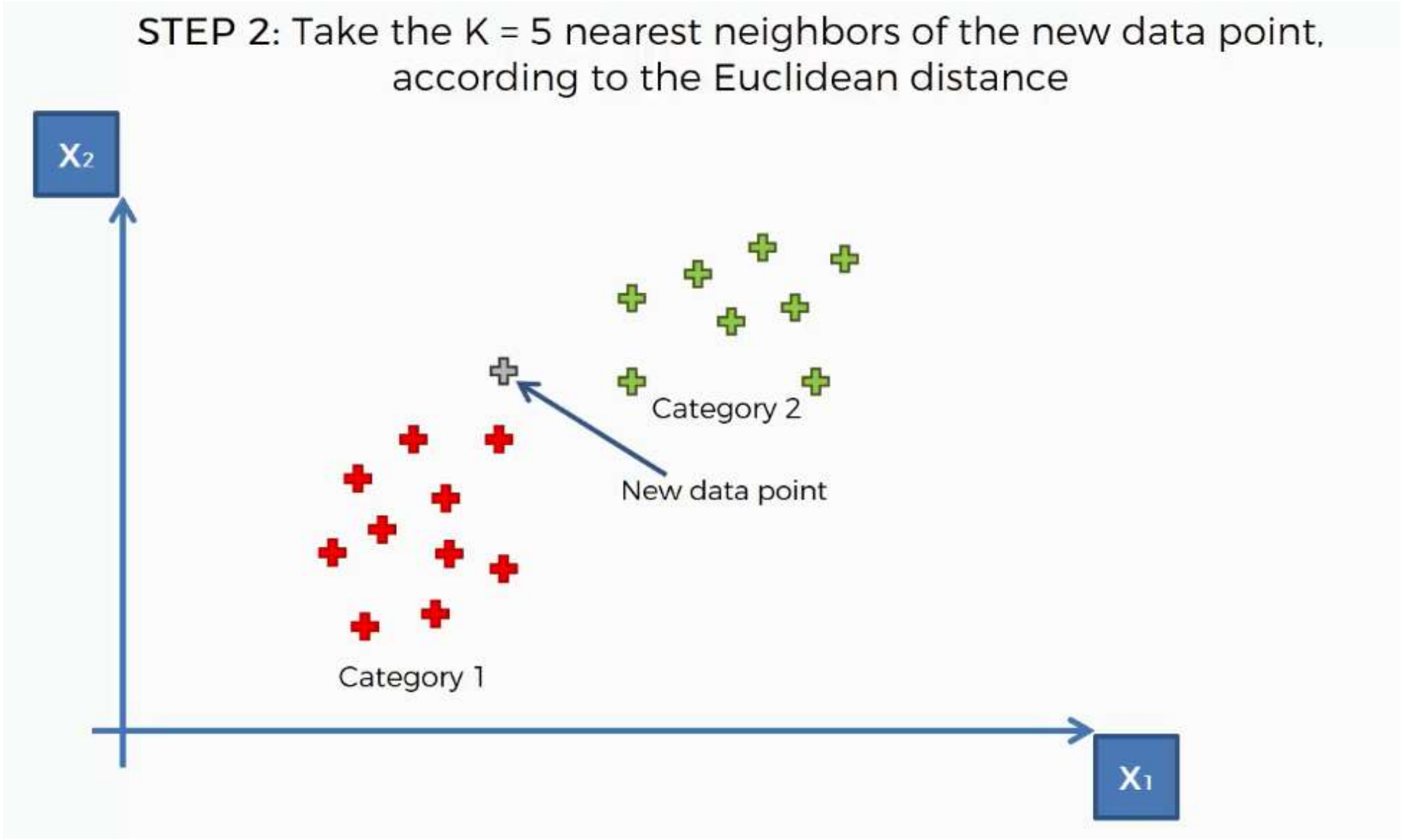

STEP 1: Choose the number K of neighbors: K = 5
X2
Category 2
New data point
Category 1
X1

Euclidean Distance



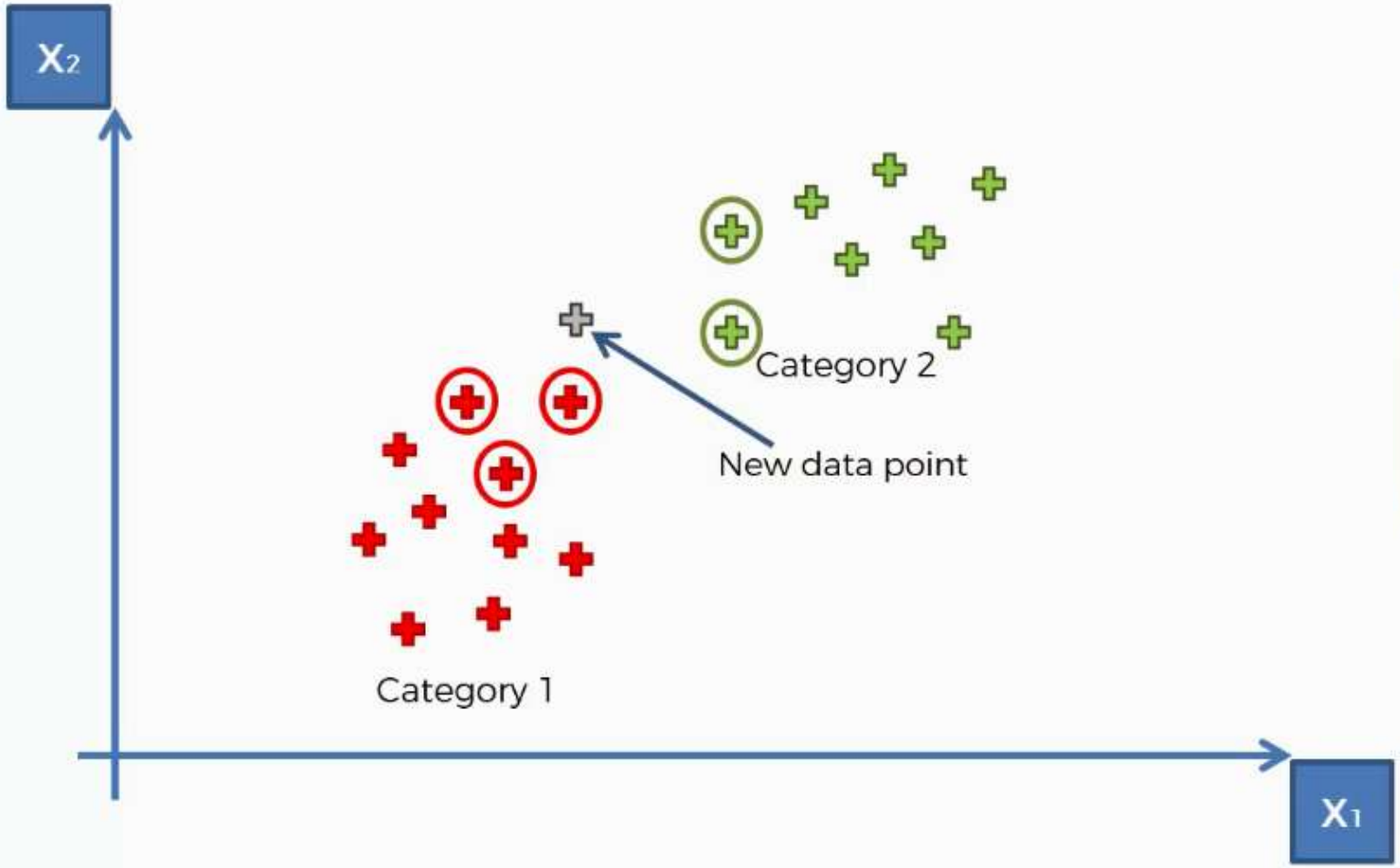
$$\text{Euclidean Distance between } P_1 \text{ and } P_2 = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

How it Work?


STEP 2: Take the K = 5 nearest neighbors of the new data point, according to the Euclidean distance
X2
Category 2
New data point
Category 1
X1

How it Work?

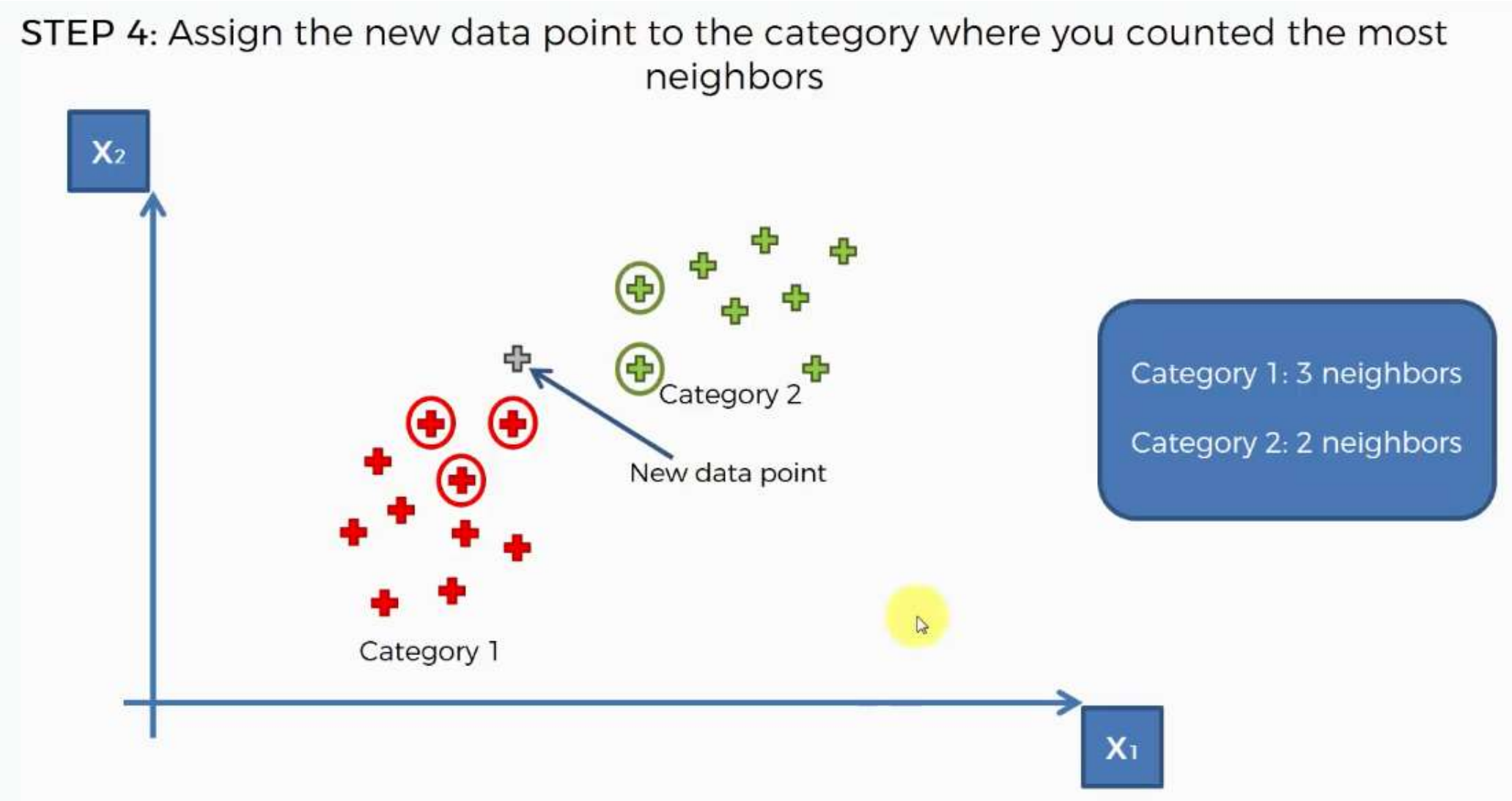
STEP 3: Among these K neighbors, count the number of data points in each category



Category 1: 3 neighbors

Category 2: 2 neighbors

How it Work?


STEP 4: Assign the new data point to the category where you counted the most neighbors
X2
X1
Category 2
New data point
Category 1
Category 1: 3 neighbors
Category 2: 2 neighbors

Demo: Creating ML model using KNN