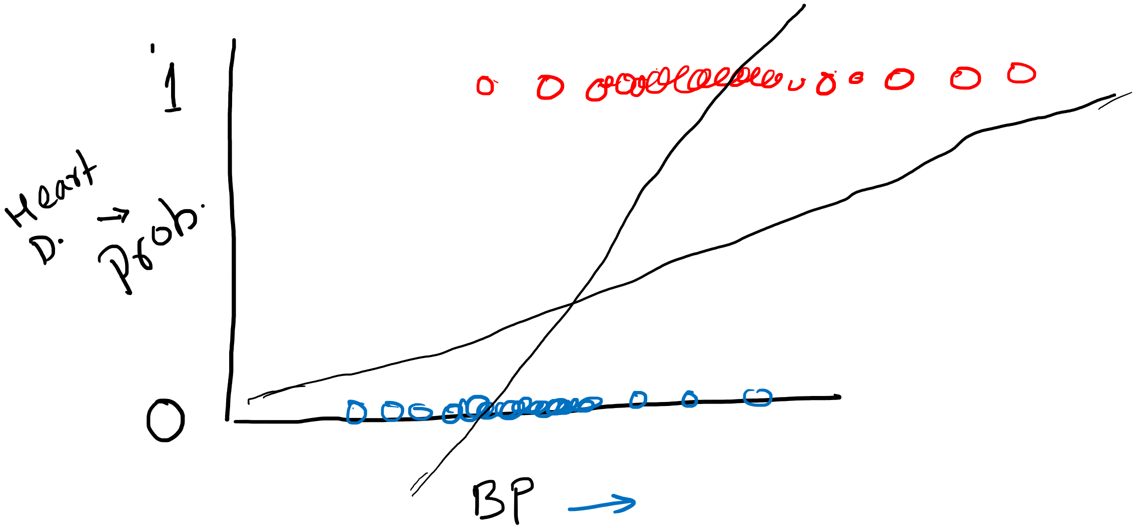
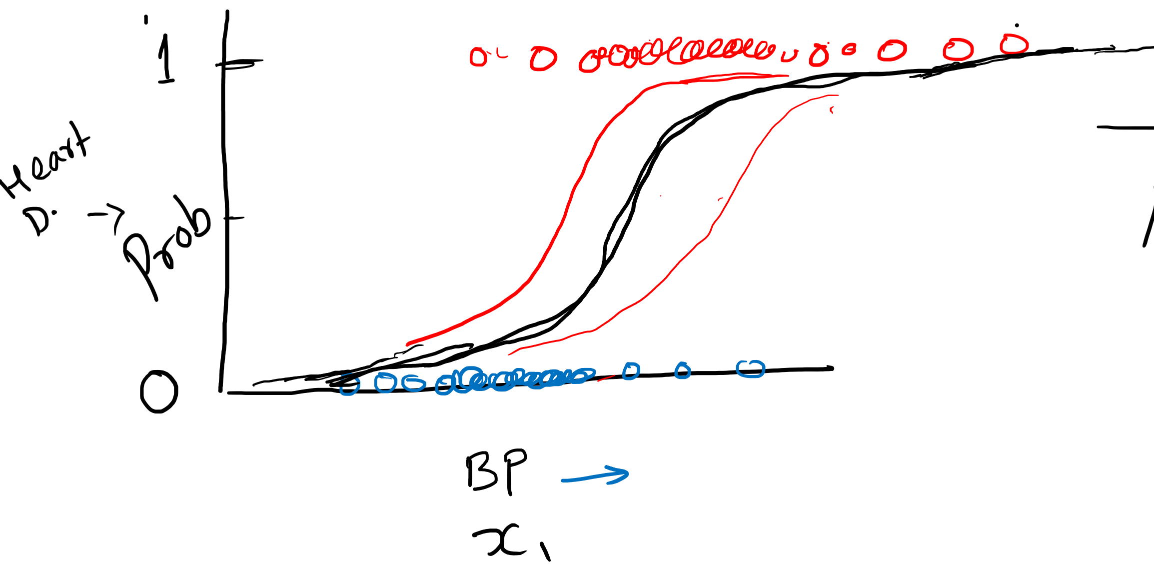
Machine Learning

Logistic Regression



$y = mx + c$

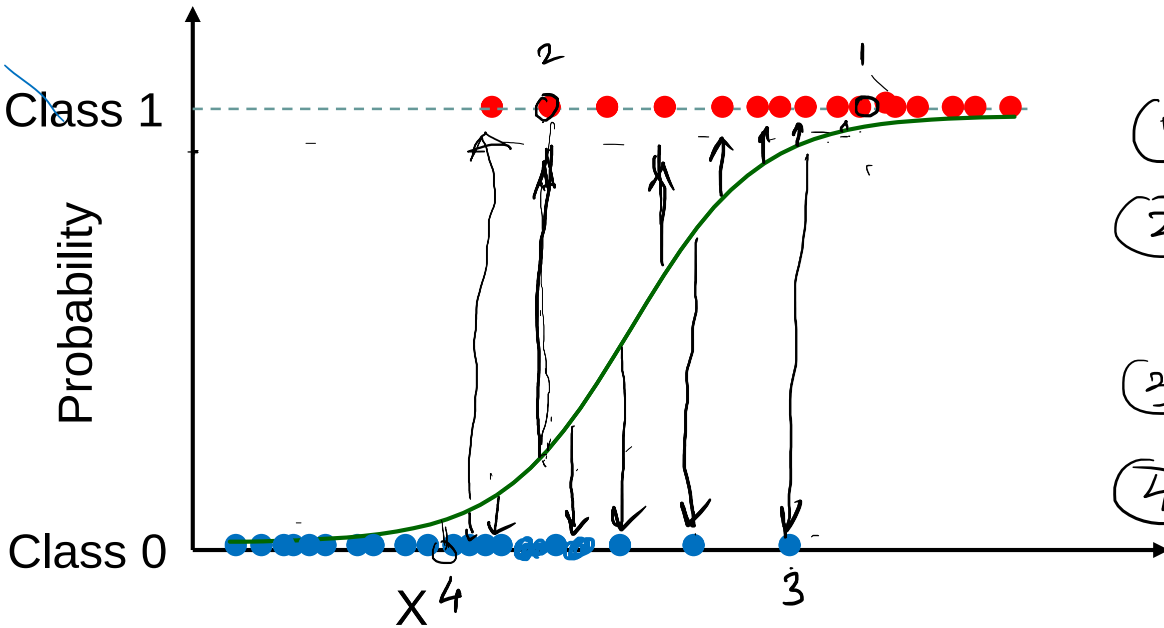
CLASSIFICATION



Sigmoid f^n

$$\frac{1}{1 + e^{-(b_0 + b_1 x_1)}}$$

Maximise Likelihood: L = $\prod P^{y_i} * (1-P)^{(1-y_i)}$



① $p^1 \cdot (1-p)^0 = p$

② $p^1 (1-p)^0 = p$

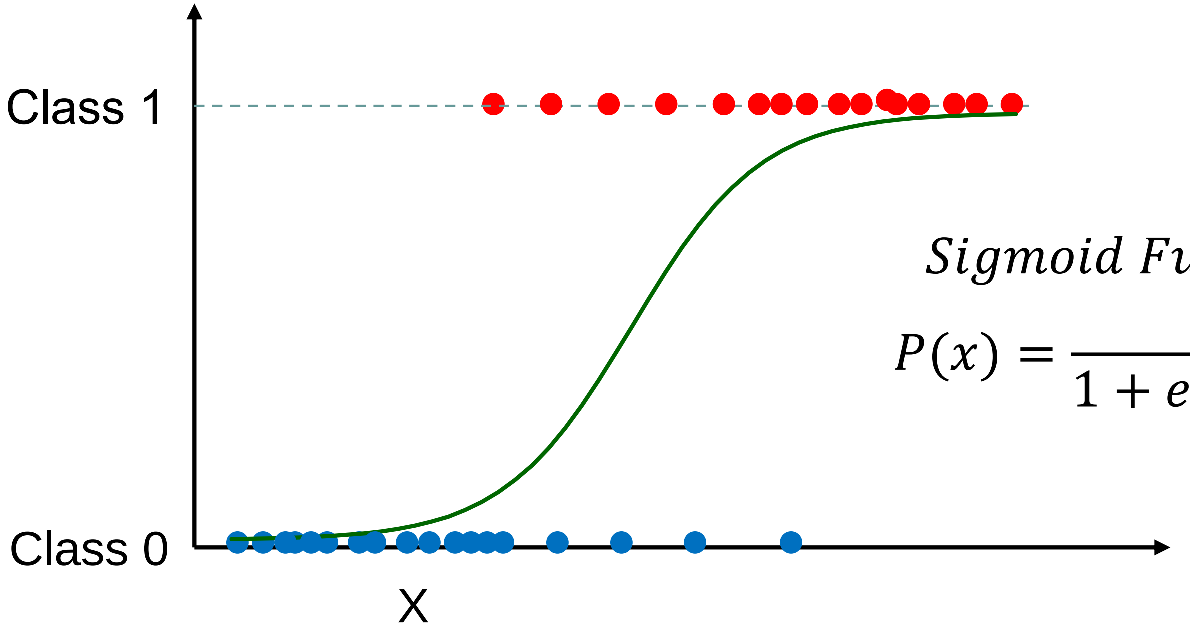
③ $p^0 (1-p)^1 = 1-p$

④ $p^0 (1-p)^1 = 1-p$

Y
1
0
1
?
0

Linear Regression and Logistic Regression

- Logistic regression uses Sigmoid function to make probabilistic prediction. Probability P that a data point belongs to a class for a given value of x
- Probability value is between 0 and 1

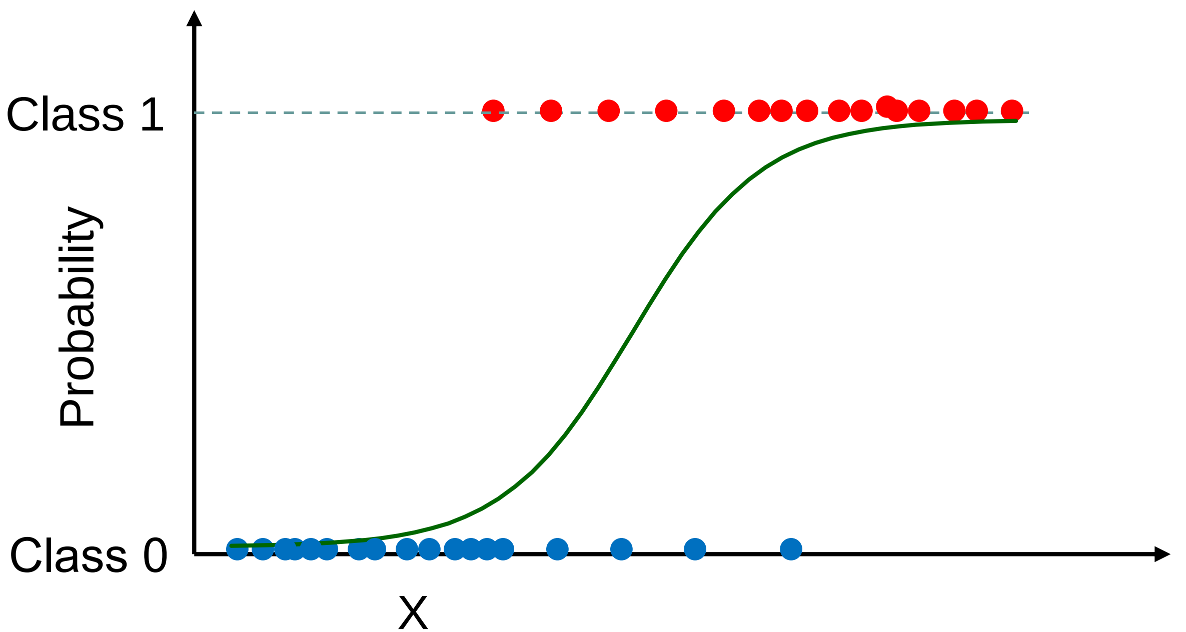


$$Sigmoid\ Function:$$

$$P(x) = \frac{1}{1 + e^{-(b_0 + b_1 * x)}}$$

Linear Regression and Logistic Regression

- As X increases, the probability value increases. As x tends to infinity, the probability becomes 1
- As value of X decreases, the probability decreases. As x tends to negative infinity, the probability becomes 0



Optimization function for Logistic Regression

Maximise Likelihood: $L = \prod P^{y_i} * (1-P)^{(1-y_i)}$

Maximise Likelihood => Maximize Log(Likelihood)

Max

$\boxed{\text{Log(Likelihood)} = \sum (y_i \log(P) + (1-y_i) \log(1-P))}$

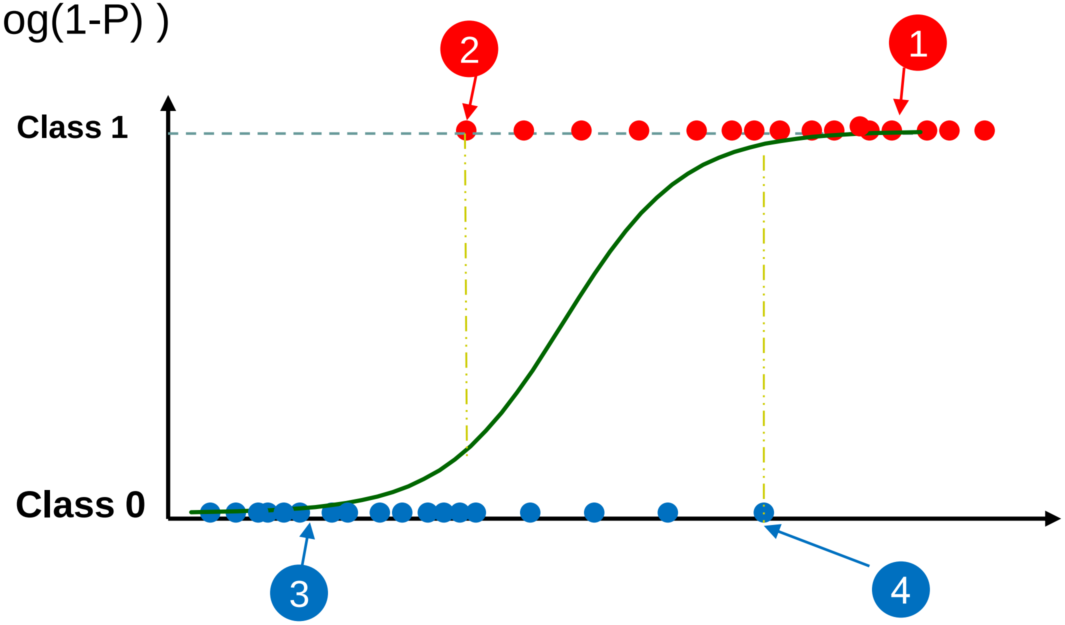
Max n ~ Max log n

Minimise LogLoss = $-\sum (\quad) (\quad)$

EXTRA SLIDE… Log Likelihood

$$\text{Log(Likelihood)} = \sum (y_i \log(P) + (1-y_i) \log(1-P))$$

p	Ln(p)	Change in value
0.0000001	-16.1181	Smaller
0.001	-6.9078	
0.1	-2.3026	
0.2	-1.6094	
0.3	-1.2040	
0.4	-0.9163	
0.5	-0.6931	
0.6	-0.5108	
0.7	-0.3567	
0.8	-0.2231	
0.9	-0.1054	
0.999	-0.0010	
0.9999999	-1E-07	Larger



	y_i	p_i	LOG(pi)	y_i * log(p_i)	1-y_i	1-p_i	LOG(1-pi)	(1-yi) *LOG(1-pi)	Log Likelihood
Case 1	1	Near 1	Larger	**Larger**	0	Near 0		**0**	**Larger**
Case 2	1	Near 0	Smaller	**Smaller**	0	Near 1		**0**	**Smaller**
Case 3	0	Near 0		**0**	1	Near 1	Larger	**Larger**	**Larger**
Case 4	0	Near 1		**0**	1	Near 0	Smaller	**Smaller**	**Smaller**

$$\frac{P}{1-P} \qquad P = \frac{1}{1+e^{-(z)}} = \frac{\left(\frac{e^z}{1+e^z}\right)}{\left(\frac{1}{1+e^z}\right)}$$

$$\frac{P}{1-P} = e^z \qquad P = 0.8 \qquad 1-P = 0.2 \qquad \frac{P}{1-P} = \frac{4}{1} = 4 = \underline{4:1}$$

ODD

$$\log \frac{P}{1-P} = \log(\text{ODDs}) = z = b_0 + b_1 x_1 + b_2 x_2 \ldots + b_n x_n$$

EXTRA SLIDE…Odds

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

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

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

Odds = $\frac{P}{1 - P} = e^{z}$

P = $\frac{Odds}{1+Odds}$

Log(Odds) = z = $(b_0 + \ b_1 * x_1 + \ ...)$

EXTRA SLIDE… Log(Odds)

- Another way to interpret logistic regression

$$\log \frac{p(x)}{1-p(x)} = \beta_0 + x \cdot \beta$$

Value of p	Odds of p
0.9	9:1
0.8	4:1
0.6	1.5:1
0.5	1:1
0.4	0.67:1
0.2	0.25:1
0.1	0.11:1

EXTRA SLIDE… Odds Ratio

$\text{Odds}_{x1} = e^{(b_0 + b_1 * x_1 + \dots)}$

$\text{Odds}_{x1+1} = e^{(b_0 + b_1 * (x_1 + 1) + \dots .)}$

$\text{Odds ratio} = \frac{\text{Odds}_{x1+1}}{\text{Odds}_{x1}} = e^{b_1}$

If b1 = 1.5, then for every unit increase in x1 (having all other Xs unchanged), the odds will increase $e^{1.5}$ times

What will be the case when b1 is negative?

Logistic Regression

Advantages -

- Efficient algorithm, does not require too many computational resources
  - Fast in training
  - Very fast in classifying unknown records
- Probabilistic view used in the method is easy to understand
- No assumptions about distributions of variables are made
- Extended to multiple classes
- Good accuracy for many simple data sets
- Resistant to overfitting

0, 1, 2

P(0, 1&2)

P(1, 0,2)

P(2, 0 1

Disadvantages -

- Underperforms where there are more complex relationships requiring non-linear boundaries

Confusion Matrix

		Predicted		
		A	B	C
Actual	A	15	0	0
	B	0	19	2
	C	0	0	17

- Classification accuracy = correct predictions / total predictions

- Precision is the proportion of the predicted positive cases that were correct.
  - Precision of C = 17 / (17+2)

- Recall is the proportion of positive cases that were correctly identified
  - Recall for B = 19 / (19+2)

- F1 Score = 2*(Recall * Precision) / (Recall + Precision)

Confusion Matrix

		Predicted	
		Negative	Positive
Actual	Negative	TN	FP
	Positive	FN	TP

		Predicted	
		Positive	Negative
Actual	Positive	TP	FN
	Negative	FP	TN

$$\textbf{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\textbf{\textit{Recall}} = \frac{TP}{TP + FN} \qquad \textbf{\textit{Precision}} = \frac{TP}{TP + FP}$$

- True Positive (TP) : Observation is positive, and is predicted to be positive.
- False Negative (FN) : Observation is positive, but is predicted negative.
- True Negative (TN) : Observation is negative, and is predicted to be negative.
- False Positive (FP) : Observation is negative, but is predicted positive.

- Note that in binary classification, **recall of the positive** class is also known as "**sensitivity**"; and **recall of the negative** class is "**specificity**".
- High recall, low precision: This means that most of the positive examples are correctly recognized (low FN) but there are a lot of false positives.
- Low recall, high precision: This shows that we miss a lot of positive examples (high FN) but those we predict as positive are indeed positive (low FP)