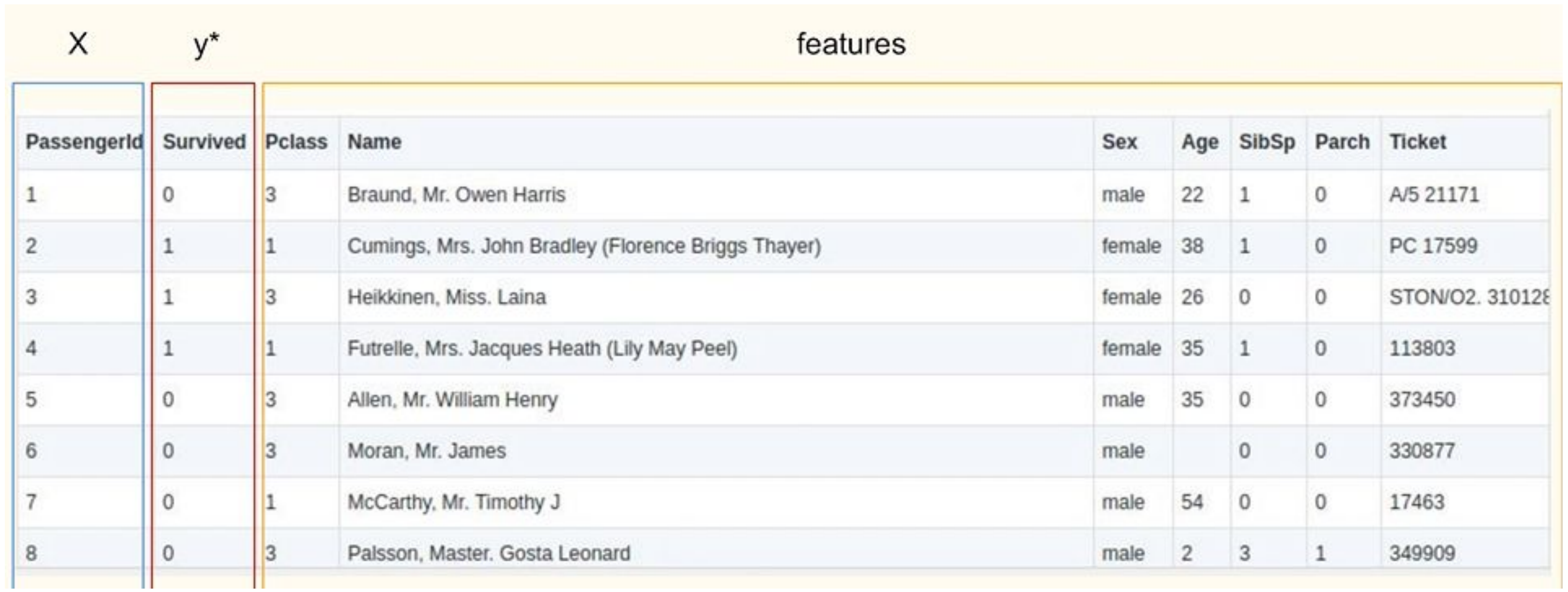
Logistic regression

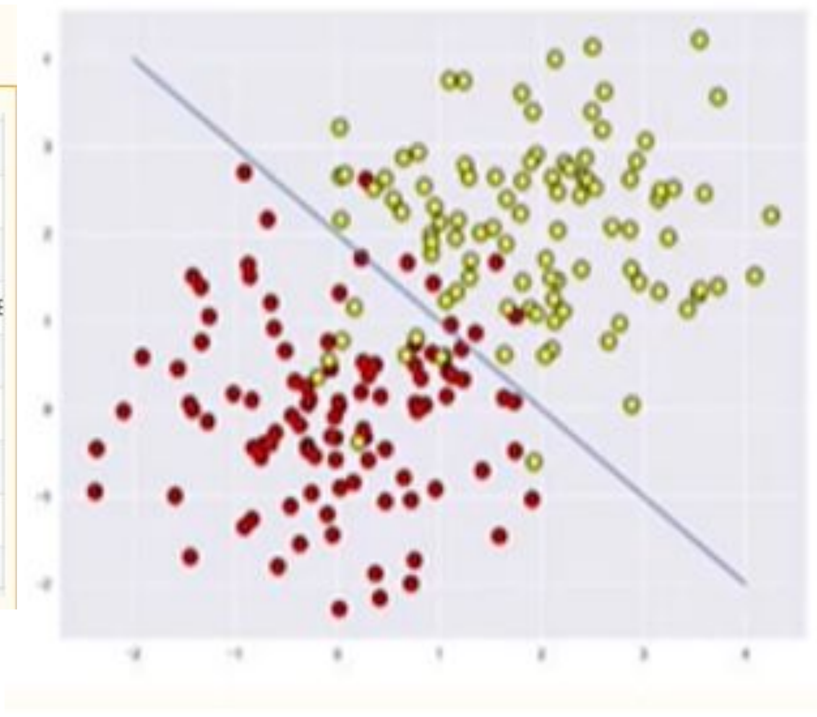
- Logistic Regression is a statistical method of classification of objects.
- In this tutorial, we will focus on solving binary classification problem using logistic regression technique.
- This tutorial also presents a case study that will let youlearn how to code and apply Logistic RegressioninPython.
-

- A doctor classifies the tumor as malignant or benign.
- A bank transaction may be fraudulent or genuine.
- For many years, humans have been performing such tasks -albeit they are error-prone. The question is can we train machines to do these tasks for us with a better accuracy?
- One such example of machine doing the classification is the email Clienton your machine that classifies every incoming mail as “spam” or “not spam” and it does it with a fairly large accuracy.
- The statistical technique of logistic regression has been successfully applied in email client. In this case, we have trained our machine to solve a classification problem

- Logistic Regression is just one part of machine learning used for solving this kind of binary classification problem. There are several other machine learning techniques that are already developed and are in practice for solving other kinds of problems.
- Here the outcome of the predication has only two values -Yes or No.
- We call these as classes -so as to say we say that our classifier classifies the objects **in two classes.** In technical terms, we can say that the outcome or target variable is **dichotomous in nature.**
-


X
y*
features

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket
1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	female	38	1	0	PC 17599
3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2. 310128
4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0	113803
5	0	3	Allen, Mr. William Henry	male	35	0	0	373450
6	0	3	Moran, Mr. James	male		0	0	330877
7	0	1	McCarthy, Mr. Timothy J	male	54	0	0	17463
8	0	3	Palsson, Master. Gosta Leonard	male	2	3	1	349909



- There are other classification problems in which the output may be classified into more than two classes. For example, given abasket full of fruits, you are asked to separate fruits of different kinds. Now, the basket may contain Oranges, Apples, Mangoes, and so on. So when you separate out the fruits, you separate them out in more than two classes. This is a **multivariate classification problem**

Classification

- Binary classification

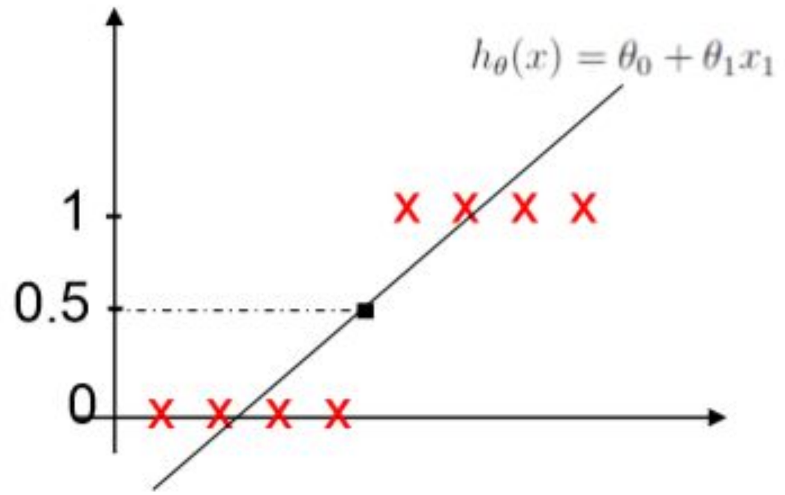
$y \in \{0, 1\}$ 0: "Negative Class" (e.g., benign tumor)
1: "Positive Class" (e.g., malignant tumor)

- Multi-class classification
  - $y \in \{0,1,2,3,\dots\}$

❑ A threshold is defined to classify

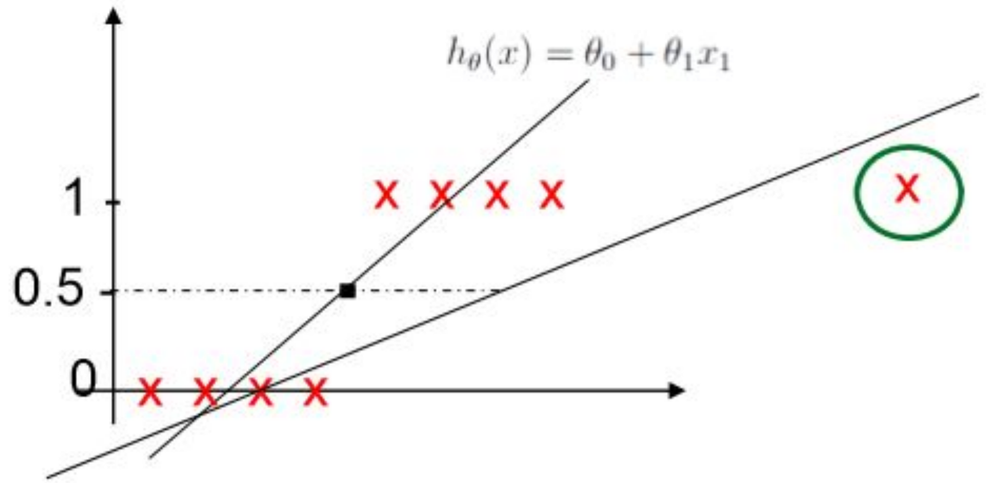
If $h_\theta(x) \geq 0.5$, predict "y = 1"

If $h_\theta(x) < 0.5$, predict "y = 0"



Linear regression for classification

- Applying linear regression for classification is often not useful



- $h_\theta(x)$ can be a large positive or negative value while y is 0 or 1
- Logistic regression : $0 \leq h_\theta(x) \leq 1$
  - A classification problem not regression despite the name

Change of hypothesis

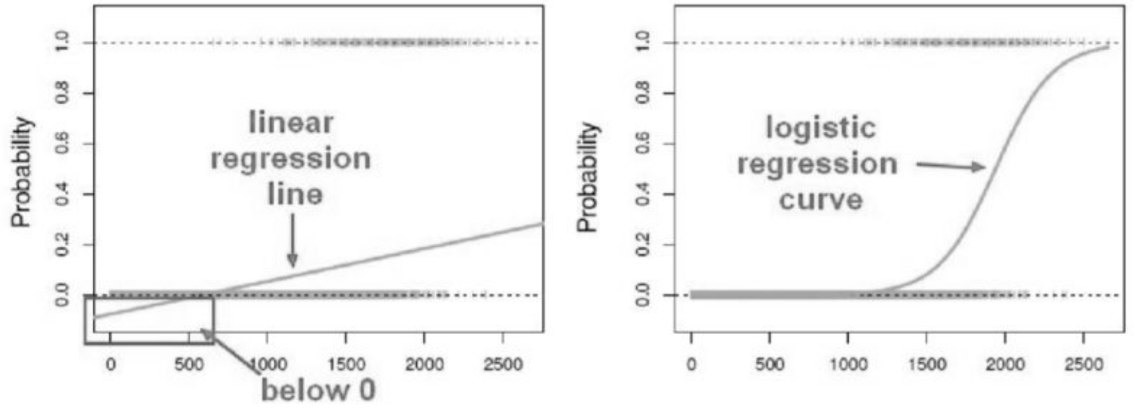
- Logistic or sigmoid function

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1+e^{-\theta^T x}}$$

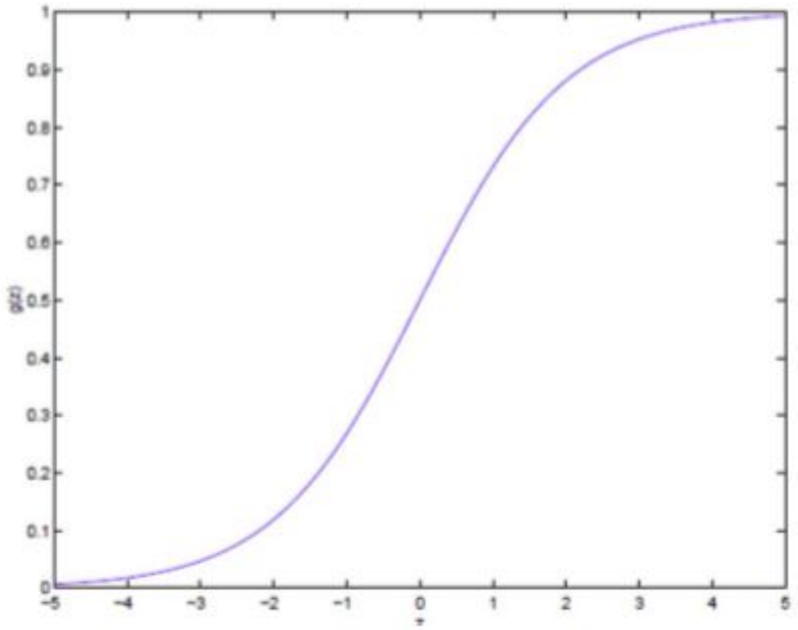
- Or $g(z) = \dfrac{1}{1+e^{-z}}$

- Derivative of sigmoid:

$$\begin{aligned} g'(z) &= \frac{d}{dz}\frac{1}{1+e^{-z}} \\ &= \frac{1}{(1+e^{-z})^2}\left(e^{-z}\right) \\ &= \frac{1}{(1+e^{-z})}\cdot\left(1-\frac{1}{(1+e^{-z})}\right) \\ &= g(z)(1-g(z)). \end{aligned}$$

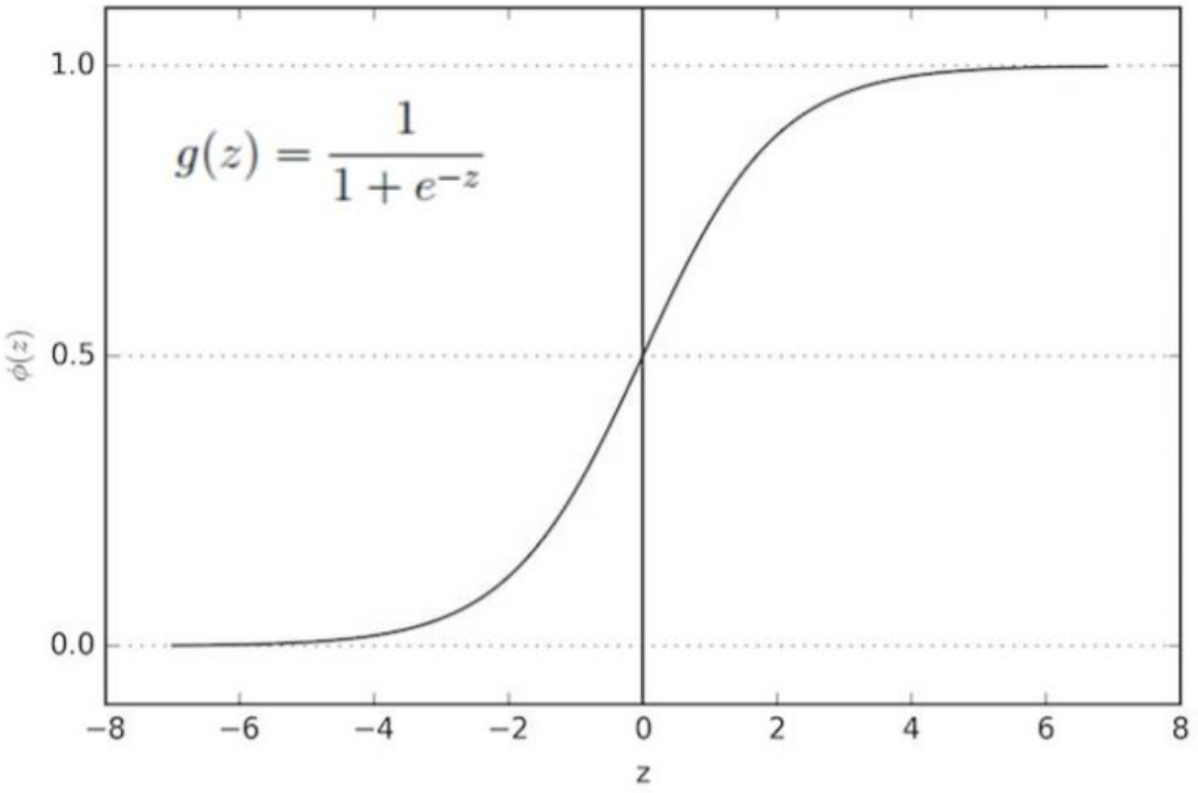


Conversion Of Linear To Logistic Regression Curve



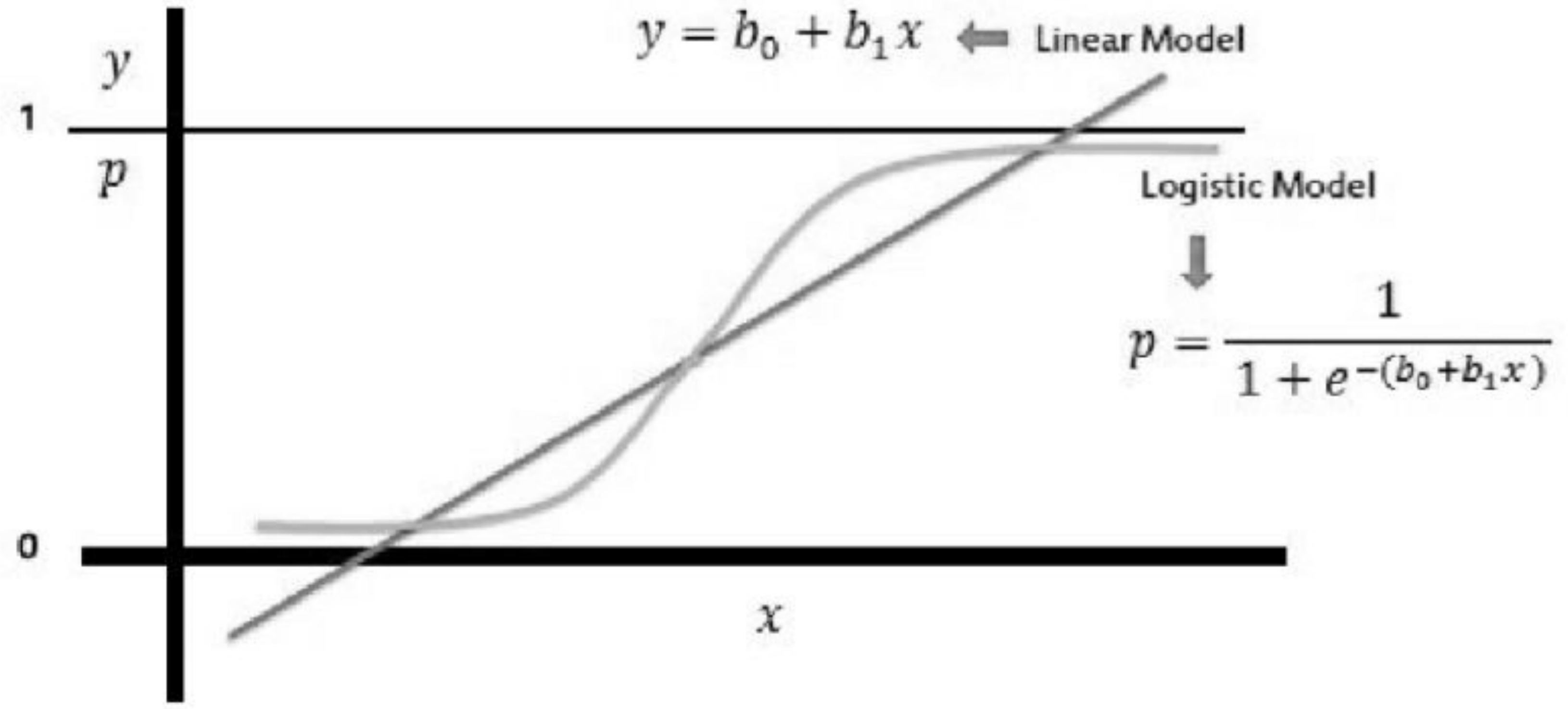
The **sigmoid function** also known as the **logistic function** is going to be the key to using **logistic regression** to perform **classification**.
The **sigmoid function** takes in any value and outputs it to be between **0**and **1**.

The **key** thing to notice here is that it doesn’t matter what value of **z** you put into the **logistics** or the **sigmoid function** you’ll always get a value between 0 and 1.



Sigmoid Function Curve

- We can take our linear regression solution and place it into the sigmoid function and it looks something like this:



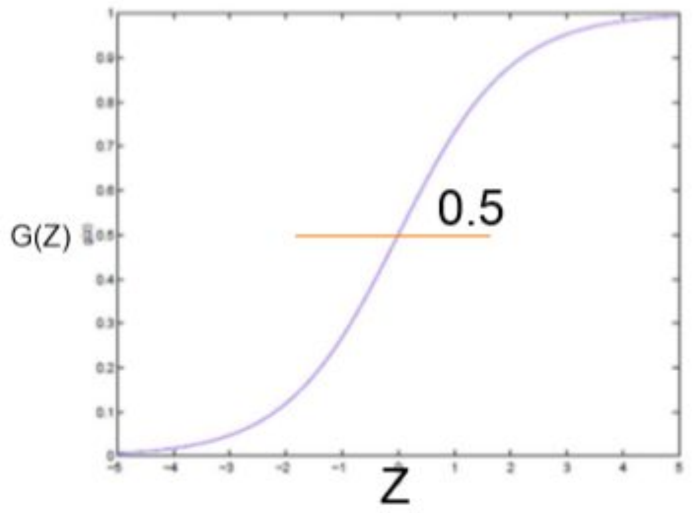
Logistic regression

- Assume:

$$P(y=1 \mid x;\theta) = h_\theta(x)$$
$$P(y=0 \mid x;\theta) = 1 - h_\theta(x)$$

- It can be written as:

$$p(y \mid x;\theta) = (h_\theta(x))^y \, (1 - h_\theta(x))^{1-y}$$



- We can see: g(z)= $h_\theta(x) = g(\theta^T x) \geq 0.5$ if $Z \geq 0$

And $g(z) < 0.5$ if $Z < 0$

- We can set a **cutoff point** at **0.5**and we can say anything below **0.5** results in **class 0** and anything above **0.5** belongs to **class 1**.

Cost Function and optimization

- Linear regression cost function was convex

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

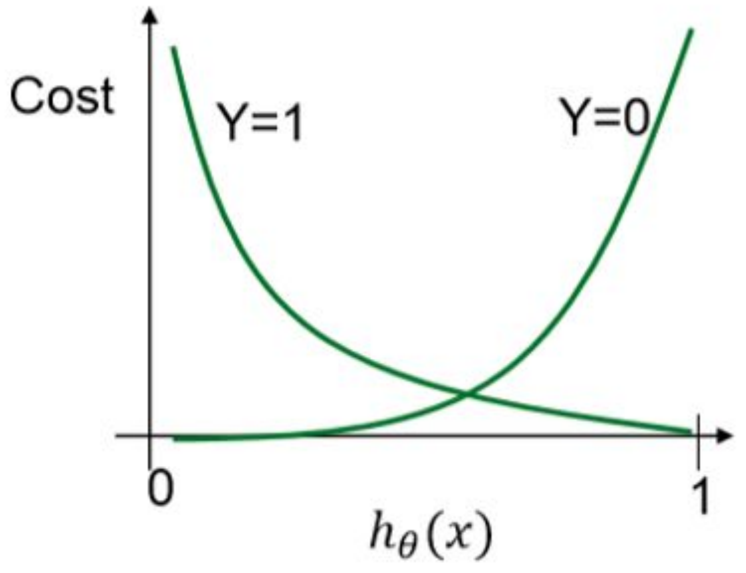
- The same cost function for logestic regression is non-convex because of nonlinear sigmoid function

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}}$$

- We define logistic regression cost function as :

$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

Convex cost function for logistic regression

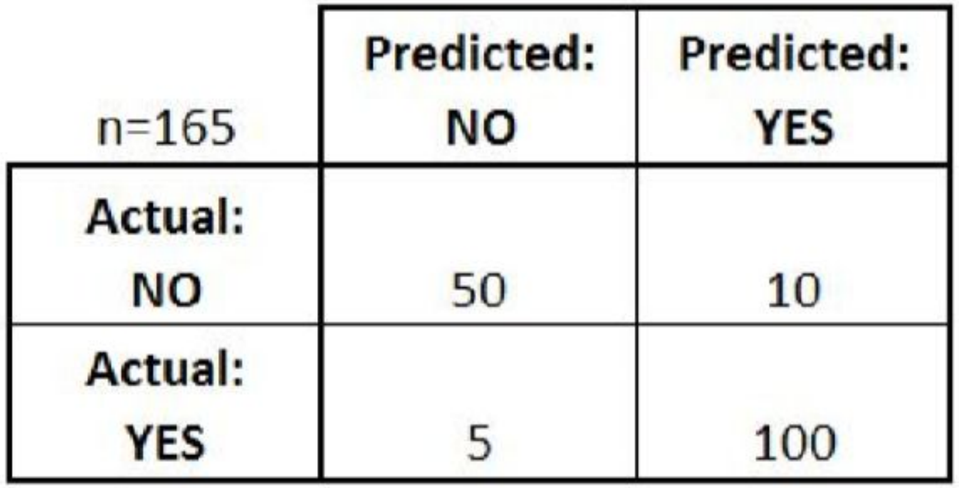


$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

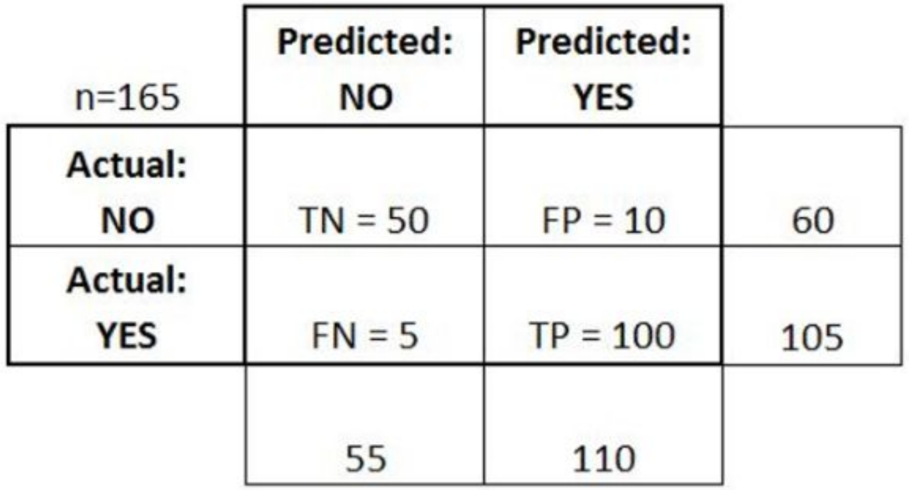
- •If h goes to zero and Cost also goes to zero, Class 0 is selected •If h goes to 1 and Cost goes to zero, class 1 is selected

Model evaluation

- After we have trained a **logistic regression** model on some **training** dataset we can evaluate the model's *performance* on some **test** dataset, we can use **confusion matrix** to *evaluate* **classification** models.
- **Confusion matrix:**
- The **confusion matrix** is a table test is often used to describe the *performance* of the **classification model** on the *test* data for which the *true* values are already known, so we can use a **confusion matrix** to evaluate a model.

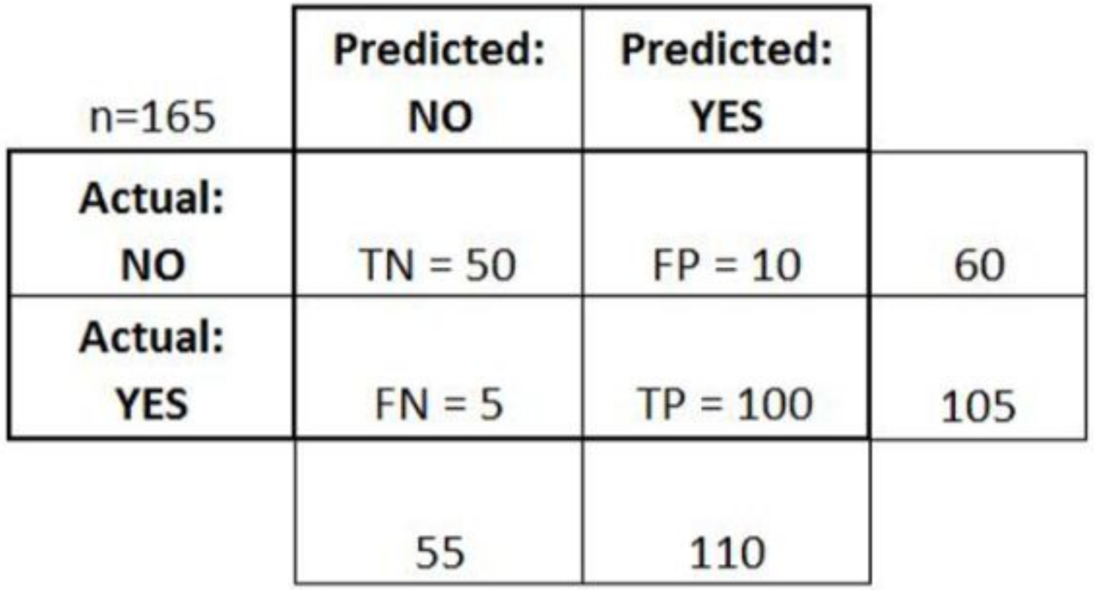


n=165	Predicted: NO	Predicted: YES
Actual: NO	50	10
Actual: YES	5	100



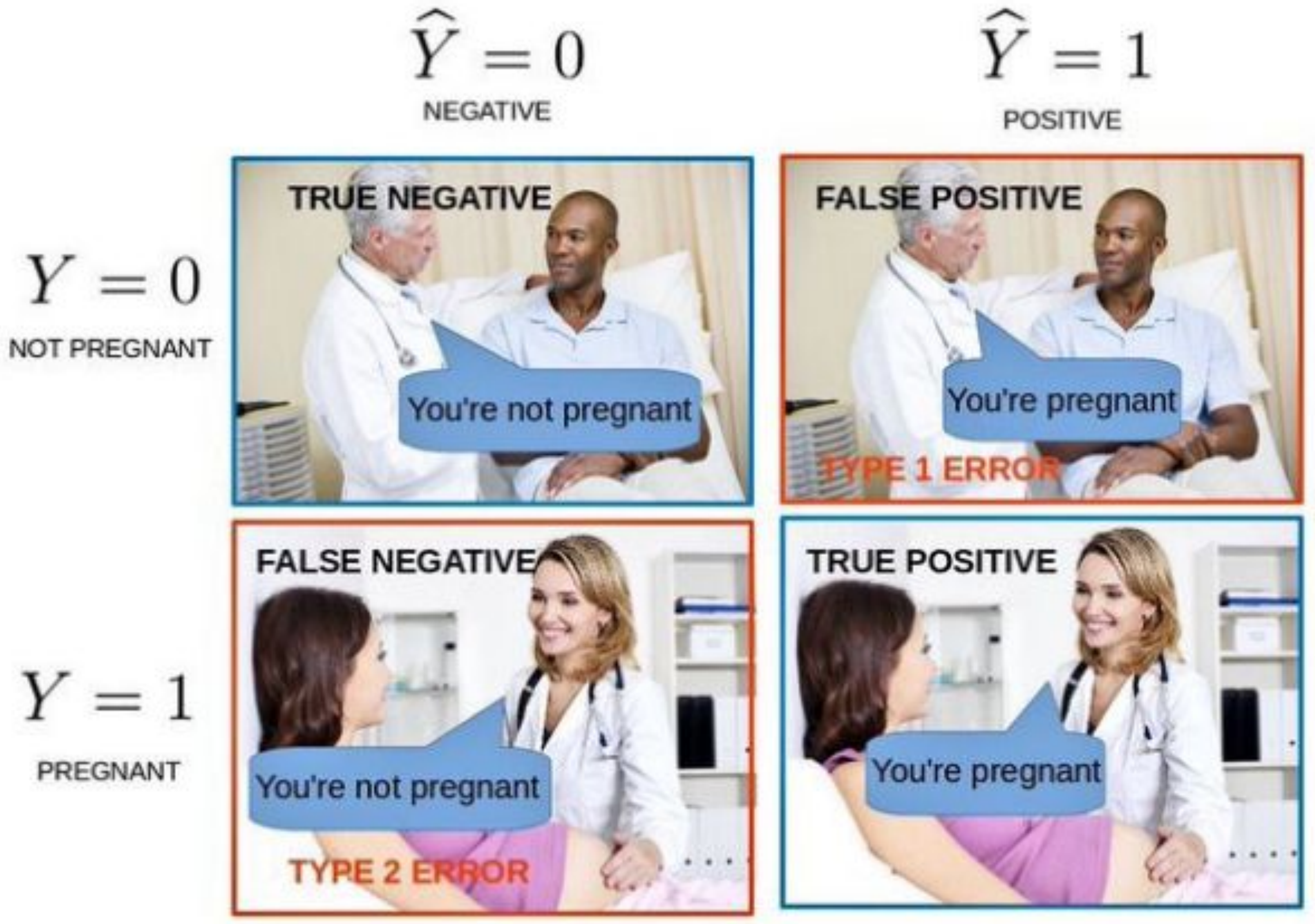
n=165	Predicted: NO	Predicted: YES	
Actual: NO	TN = 50	FP = 10	60
Actual: YES	FN = 5	TP = 100	105
	55	110	

- #**example**: testing the presence of a *disease*
- **NO = negative test = False = 0**
- **YES = positive test = True = 1**
- **Basic Terms:**
- **True Positives(TP)**= are the cases in which we *predicted yes* they have the disease and in reality, they *do have* the disease.
- **True Negative(TN)**= are the cases in which we *predicted no* they don’t have the disease and in reality, *they don’t* have the disease.
- **False Positive(FP)** = are the cases in which we *predicted yes* they have the disease and in reality, *they don’t* have the disease. This is also known as **Type 1 Error.**
- **False Negative(FN)**= are the cases in which we *predicted no* they don’t have the disease and in reality, *they do* have the disease. This is also known as the **Type 2 Error**.



n=165	Predicted: NO	Predicted: YES	
Actual: NO	TN = 50	FP = 10	60
Actual: YES	FN = 5	TP = 100	105
	55	110	

- **Misclassification Rate:**
- **how often is it wrong?**
- **MR = (FP+FN)/total**
- **MR = (10+5)/165 = 0.09**
- This is also called as the **Error Rate**

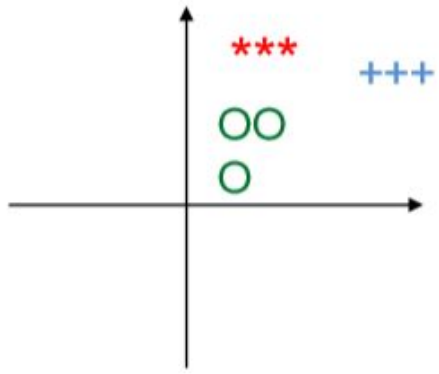


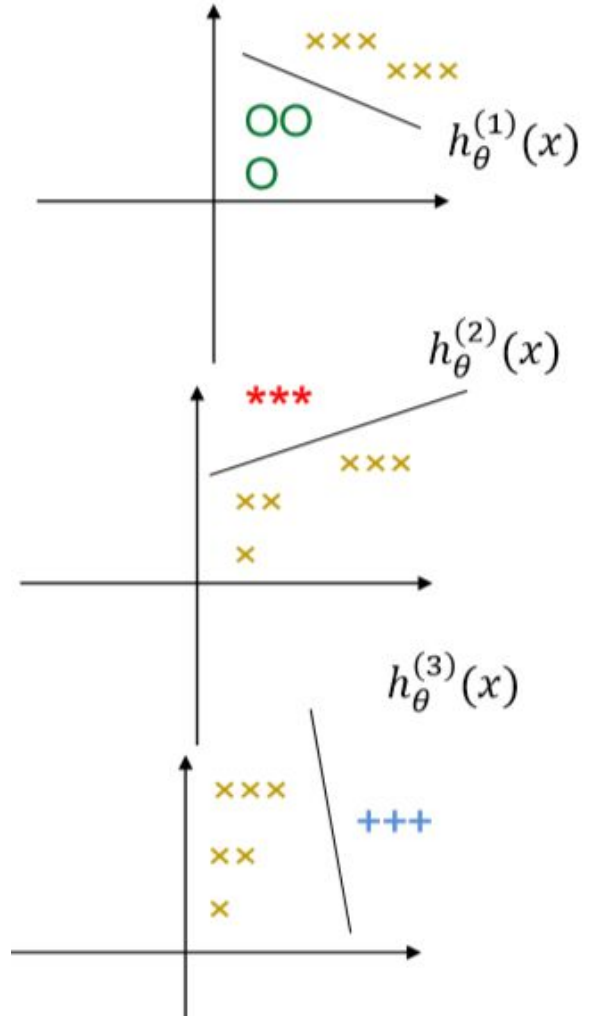
Type 1 and Type 2 error

Type of Errors:

1. Type 1 Error(False Positive)
2. Type 2 Error(False Negative)

Multi-class classification



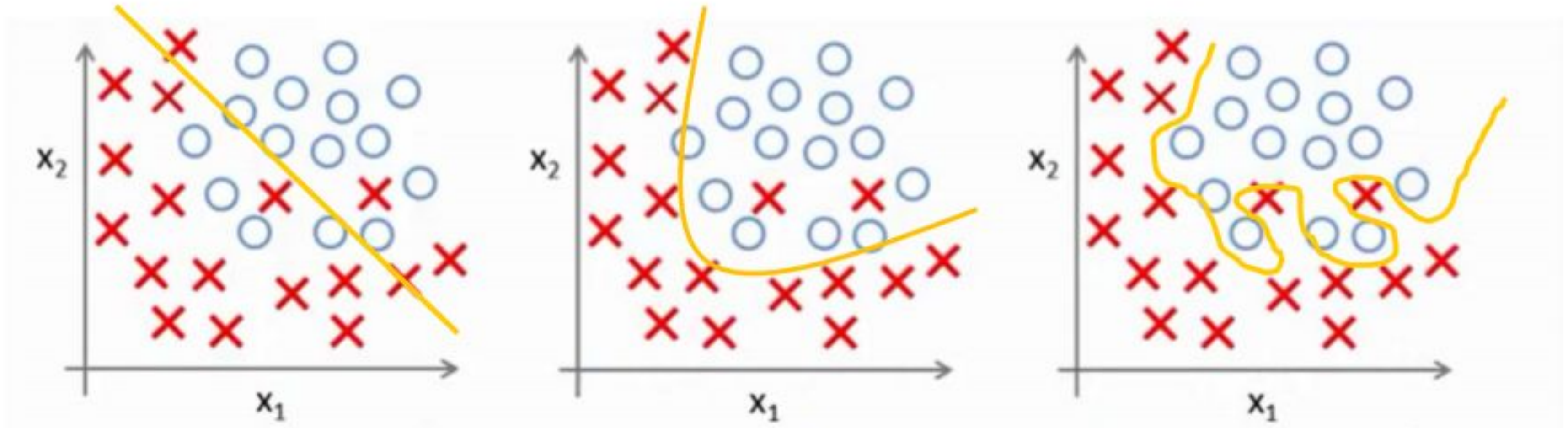


- One-vs-all strategy: working with multiple binary classifications
- We train one logistic regression classifier for each class i to predict the probability that y = i

$$h_\theta^{(i)}(x) = P(y = i|x;\theta) \qquad (i = 1, 2, 3)$$

For each x, pick the class having highest value of probability $\max_i h_\theta^{(i)}(x)$

Overfitting



$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$

(g = sigmoid function)

$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2)$

$g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \theta_6 x_1^3 x_2 + \ldots)$

How to deal with overfitting

- Seems having higher order of polynomials is good fit, but how to deal with overfitting?
- Reduce the number of features manually –Keep all the features, but apply regularization
- The most common variants in machine learning are $L_1$and L_2 regularization
- –Minimizing $E(X, Y) + \alpha\|w\|$, where w is the model's weight vector, $\|\cdot\|$ is either the L_1norm or the squared L_2norm, and α is a free parameter that needs to be tuned empirically
- –Regularization using L_2norm is called Tikhonov regularization (Ridge regression), using L_1norm is called Lasso regularization

Advantages:

- it doesn’t require high *computational power*
- is easily *interpretable*
- is used widely by the **data analyst** and **data scientists**.
- is very easy to ***implement***
- it doesn’t require *scaling* of *features*
- it provides a ***probability*** *score* for *observations*.

Disadvantages:

- while working with ***Logistic regression*** you are not able to handle a large number of *categorical features/variables*.
- it is **vulnerable** to overfitting
- it cant solve the *non-linear* problem with the *logistic regression model* that is why it requires a *transformation* of *non-linear* features
- **Logistic regression** will not perform well with *independent(**X**)* variables that are not **correlated** to the *target(**Y**)* variable.

https://www.youtube.com/watch?v=yIYKR4sgzI8

AT HOME

- https://www.youtube.com/watch?v=zAULhNrnuL4
- https://www.youtube.com/watch?v=ckkiG-SDuV8
- https://www.youtube.com/watch?v=NmjT1_nClzg
- https://www.youtube.com/watch?v=gcr3qy0SdGQ
- https://www.youtube.com/watch?v=gcr3qy0SdGQ
- https://www.youtube.com/watch?v=scVUuaLmb9o