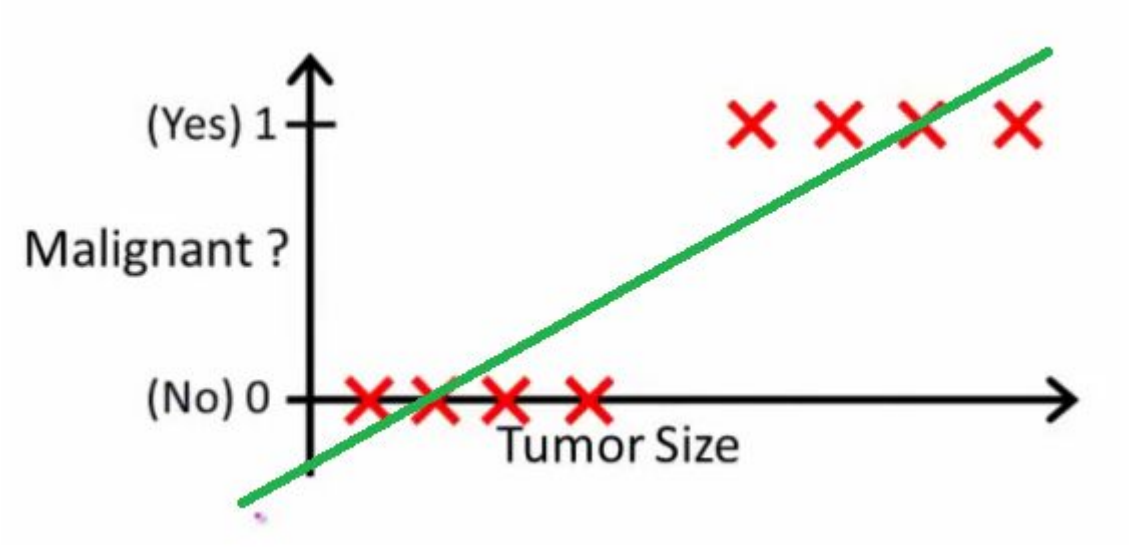
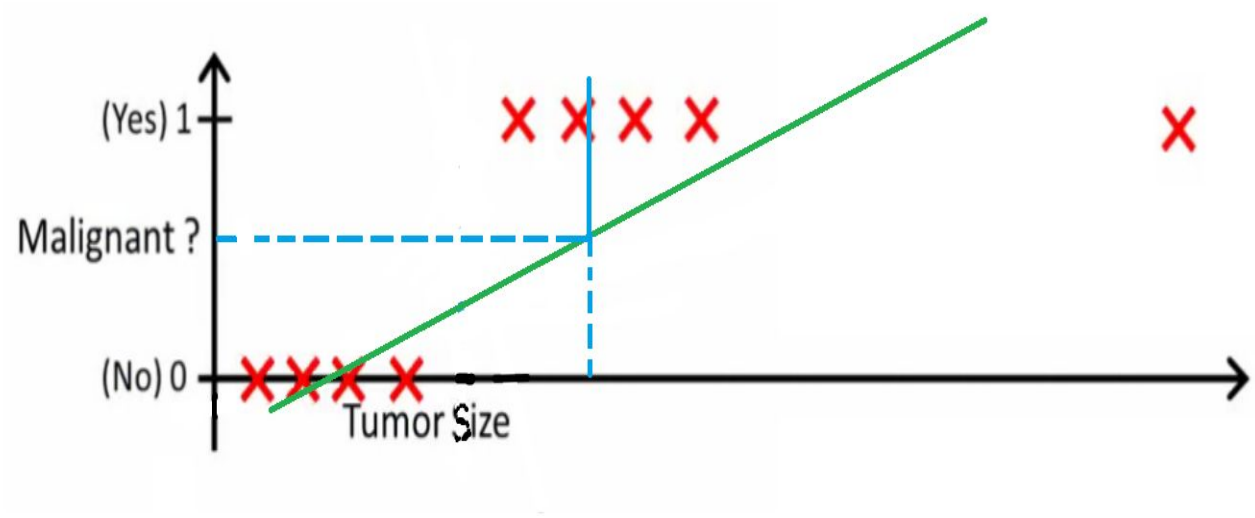
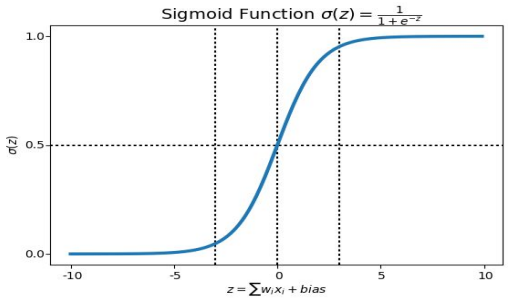
Logistic Regression





Sigmoid function



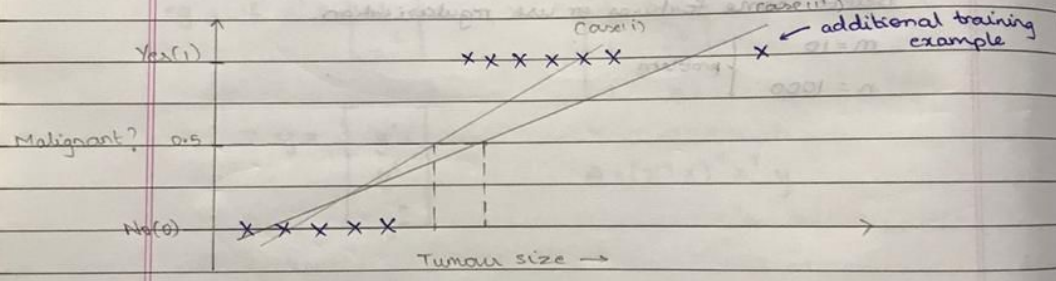
logistic regression

Classification
→ Email : Spam / Not spam?
→ Online Transactions : Fraudulent (Yes/No)?
→ Tumor : Malignant / Benign ?

$y \in \{0, 1\}$ 0 : " Negative Class"
1 : " positive class"

This is binary classification with two values of y.

let's take an example



$h_\theta(x) = \theta^T x$

Case(i) Threshold classifier output $h_\theta(x)$ at 0.5 :
→ $h_\theta(x) \geq 0.5$, predict " $y = 1$ "
→ $h_\theta(x) \leq 0.5$, predict " $y = 0$ "

case(ii) linear regression fails at threshold 0.5 and hence linear regression in classification problem is not suitable
Also $h_\theta(x)$ can be >1 or <0 but here $y = 0$ or 1

So we use logistic regression : $0 \leq h_\theta(x) \leq 1$
Classification

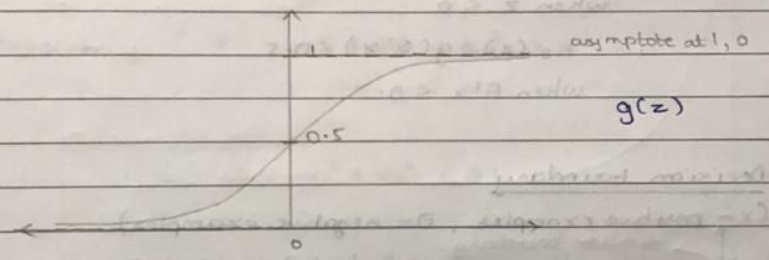
Logistic regression - hypothesis representation

Want $0 \le h_\theta(x) \le 1$

$\therefore \; h_\theta(x) = g(\theta^T x)$

$g(z) = \dfrac{1}{1+e^{-z}}$ ... sigmoid / logistic function

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



Interpretation of hypothesis output

$h_\theta(x)$ = estimated probability that $y=1$ on input x

Example: if $x = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \text{tumour size} \end{bmatrix}$

$h_\theta(x) = 0.7$

Tell the patient that 70% chance of tumour being malignant

$h_\theta(x) = p(y=1 \mid x; \theta)$ → "probability that $y=1$, given x, parametrized by θ)

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

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

Logistic regression - decision boundary

- suppose predict "y=1" if $h_\theta(x) \geq 0.5$

  >> $g(z) \geq 0.5$

  when $z \geq 0$

  $\therefore h_\theta(x) = g(\theta^T x) \geq 0.5$

  when $\theta^T x \geq 0$

- predict "y=0" if $h_\theta(x) < 0.5$

  >> $g(z) \leq 0.5$

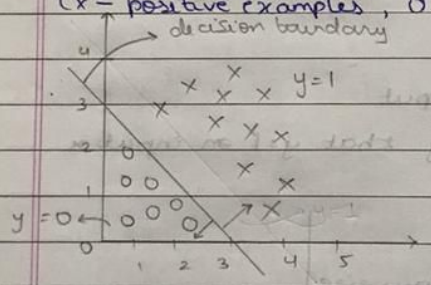
  when $z < 0$

  $\therefore h_\theta(x) = g(\theta^T x) < 0.5$

  when $\theta^T x < 0$.

Decision boundary

(x - positive examples, O - negative examples)



$$h_\theta(x) = g(\underset{-3}{\theta_0} + \underset{1}{\theta_1} x_1 + \underset{1}{\theta_2} x_2)$$

$$\theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$$

predict "y=1" if $\theta^T x \geq 0$

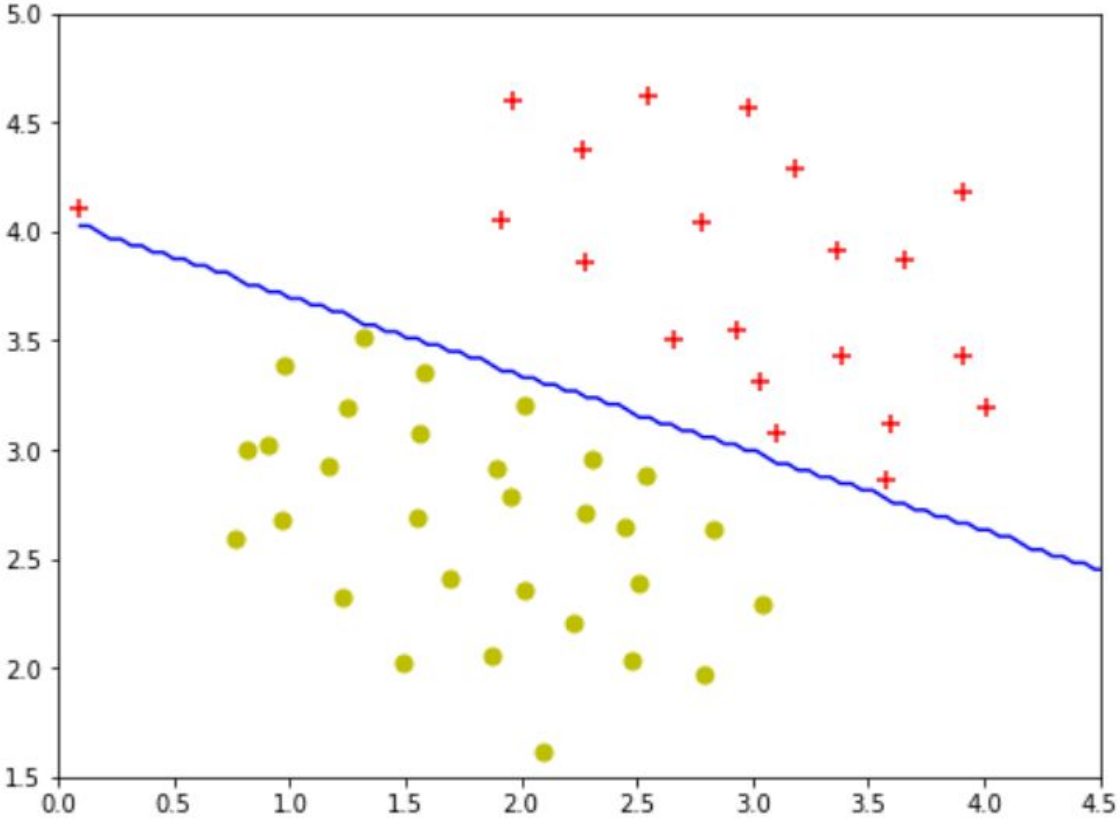
$-3 + x_1 + x_2 \geq 0$

$x_1 + x_2 \geq 0$

"y=0" if $x_1 + x_2 < 0$

* Decision boundary is a property of the hypothesis, θ, it is not a function / property of the data set.

Linear decision boundary



Logistic regression - decision boundary

- suppose predict "y=1" if $h_\theta(x) \geq 0.5$

  >> $g(z) \geq 0.5$

  when $z \geq 0$

  $\therefore h_\theta(x) = g(\theta^T x) \geq 0.5$

  when $\theta^T x \geq 0$

- predict "y=0" if $h_\theta(x) < 0.5$

  >> $g(z) \leq 0.5$

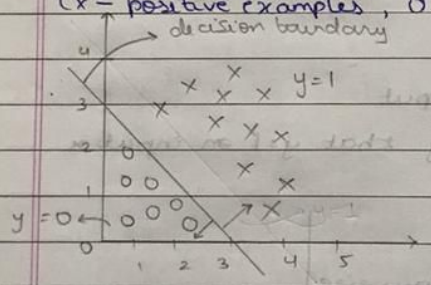
  when $z < 0$

  $\therefore h_\theta(x) = g(\theta^T x) < 0.5$

  when $\theta^T x < 0$.

Decision boundary

(x - positive examples, O - negative examples)



$$h_\theta(x) = g(\underset{-3}{\theta_0} + \underset{1}{\theta_1} x_1 + \underset{1}{\theta_2} x_2)$$

$$\theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$$

predict "y=1" if $\theta^T x \geq 0$

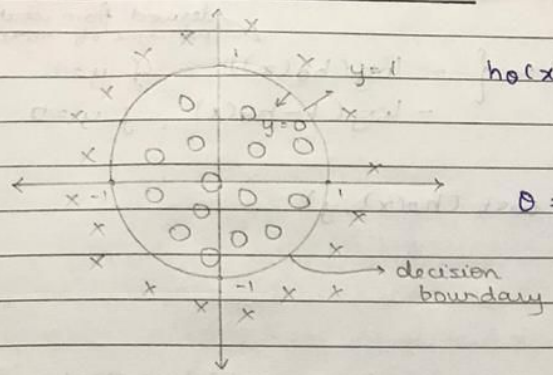
$-3 + x_1 + x_2 \geq 0$

$x_1 + x_2 \geq 0$

"y=0" if $x_1 + x_2 < 0$

* Decision boundary is a property of the hypothesis, θ, it is not a function / property of the data set.

Non-linear decision boundaries



$$h_\theta(x) = g(\overset{-1}{\theta_0} + \overset{0}{\theta_1} x_1 + \overset{0}{\theta_2} x_2 + \underset{1}{\theta_3} x_1^2 + \underset{1}{\theta_4} x_2^2)$$

$$\theta = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

Predict "y=1" if $-1 + x_1^2 + x_2^2 \geq 0$

$x_1^2 + x_2^2 \geq 1$

"y=0" if $x_1^2 + x_2^2 < 1$

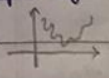
* Training set may be used for fitting the values of θ
* Decision boundary does not depend on the training set

With higher order polynomial we could have figures which are more complex (ellipse,  , ) as our decision boundaries

Cost function

We cannot use $J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \frac{1}{2} (h_\theta(x^i) - y^i)^2$

$\text{cost}(h_\theta(x), y) = \frac{1}{2} (h_\theta(x) - y)^2$ because for logistic

regression $J(\theta)$ is non ~~complex~~ convex. ie the graph of $J(\theta)$ vs θ is not  but insted may be like 

So we do not global minimum by gradient descent

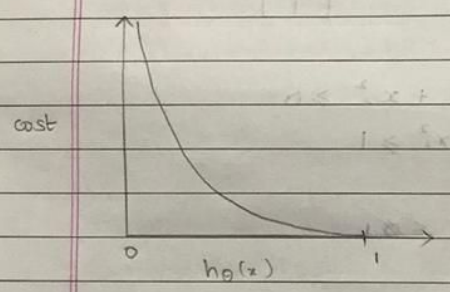
∴ logistic regression - cost function

derived from statistics principle of maximum likelihood estimation

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

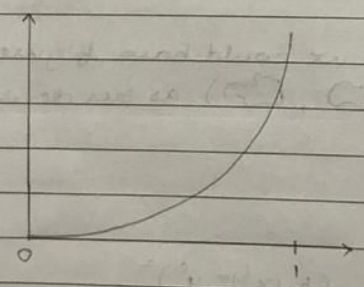
$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \text{cost}(h_\theta(x), y)$$

→) If $y=1$



Cost = 0 if $h_\theta(x) = 1$
But as $h_\theta(x) \to 0$
cost $\to \infty$
$h_\theta(x) = 0$ but $y = 1$
so the algorithm is penalized with a very high cost

→) If $y = 0$



Cost = 0 if $h_\theta(x) = 0$
But as $h_\theta(x) \to 1$
cost $\to \infty$
above explanation.

Simplified cost function

$$\text{Cost}(h_\theta(x), y) = -y(\log(h_\theta(x))) - (1-y)\log(1-h_\theta(x))$$

If $y=1$: $\text{cost}(h_\theta(x), y) = -\log(h_\theta(x))$

If $y=0$: $\text{cost}(h_\theta(x), y) = -\log(1-h_\theta(x))$

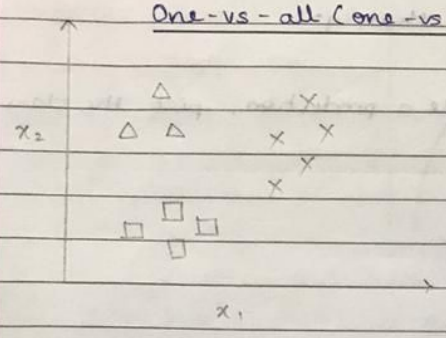
→ L-BFGS

Multiclass classification

- Email foldering: Work, friends, family, Hobby
  - Work → $y=1$, friends → $y=2$, family → $y=3$, Hobby → $y=4$
- Medical diagrams: Not ill, cold, Flu
  - Not ill → $y=1$, cold → $y=2$, Flu → $y=3$
- Weather: sunny, cloudy, Rain, Snow
  - sunny → $y=1$, cloudy → $y=2$, Rain → $y=3$, Snow → $y=4$

Multi class classification

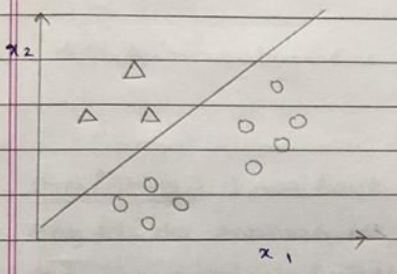
One-vs-all (one-vs-rest):



Class 1 = △ , $y=1$
Class 2 = □ , $y=2$
Class 3 = X , $y=3$

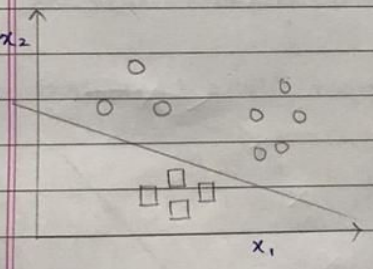
we take this and turn it to seperate binary classification

case(i)



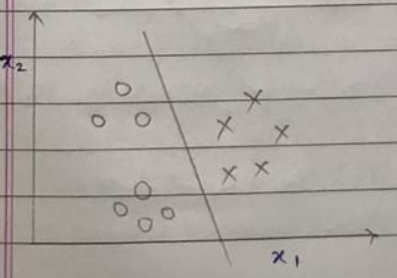
$h_\theta^{(1)}(x)$

case(ii)



$h_\theta^{(2)}(x)$

case(iii)

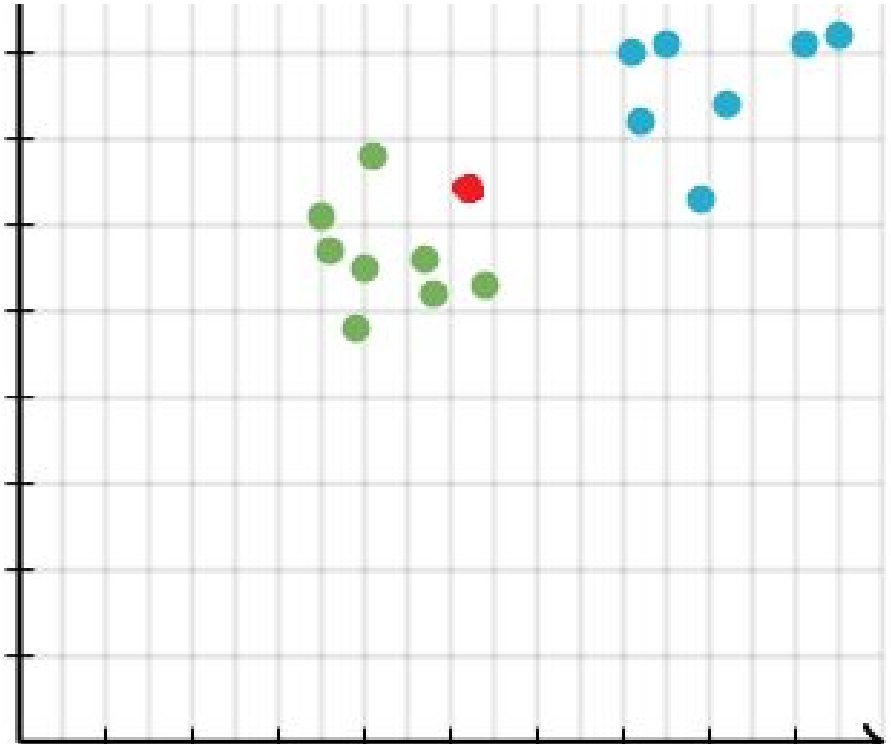


$h_\theta^{(3)}(x)$

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

K-nearest Neighbours

The k-nearest neighbors (KNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems.
KNN is a non-parametric, lazy learning algorithm. Its purpose is to use a database in which the data points are separated into several classes to predict the classification of a new sample point.
KNN Algorithm is based on **feature similarity.**



When we say a technique is non-parametric , it means that it does not make any assumptions on the underlying data distribution.

KNN is also a lazy algorithm.What this means is that it does not use the training data points to do any generalization.

The KNN Algorithm

1.Load the data
2.Initialize K to your chosen number of neighbors
3. For each example in the data
3.1 Calculate the distance between the query example and the current example from the data.
3.2 Add the distance and the index of the example to an ordered collection
4. Sort the ordered collection of distances and indices from smallest to largest (in ascending order) by the distances
5. Pick the first K entries from the sorted collection
6. Get the labels of the selected K entries
7. If regression, return the mean of the K labels
8. If classification, return the mode of the K labels

A few Applications and Examples of KNN

1.Should the bank give a loan to an individual? Would an individual default on his or her loan? Is that person closer in characteristics to people who defaulted or did not default on their loans?

2.In political science — classing a potential voter to a “will vote” or “will not vote”, or to “vote Democrat” or “vote Republican”.