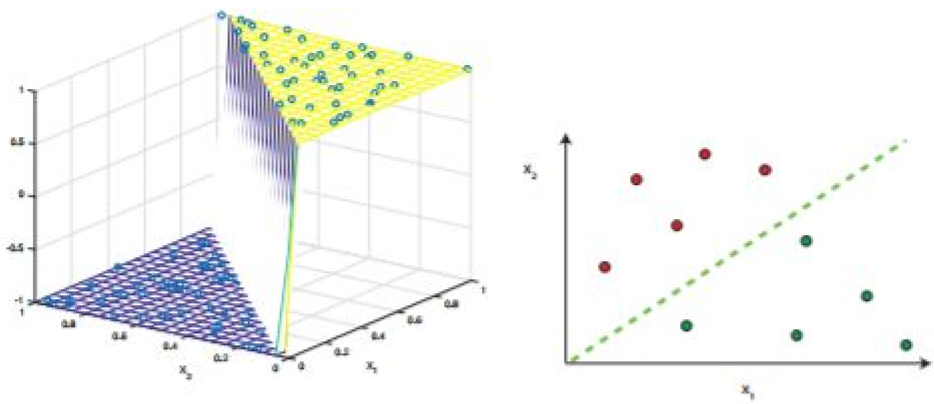
Classification

- **Logistic Regression with Two Classes**
    - **Two classes**
        - Output/target y is the **label** and takes a value of -1 or 1
        - **Classifier predicts the label of a new object**
            - E.g. if an email is spam or not spam
        - **Hypothesis**
            - Fitting a straight line to the data doesn't seem appropriate so not using same as lin reg
            - **Predict 1 when** $\theta^T x \geq 0$ **and -1 when** $\theta^T x < 0$
            - **So** $h_\theta(x) = sign(\theta^T x)$
        - Logistic regression is essentially trying to fit a plane that separates Y = 1 data from the Y = 0 data
        - $\theta^T x$ **defines:**
            - **A point in one dimension** $1 + 0.5x_1 = 0 \rightarrow x_1 = -2$
            - **A line in two dimensions** $2 + x_1 + 2x_2 = 0 \Rightarrow x_2 = \frac{-x_1}{2} - 1$
            - **…**
            - **A plane in higher dimensions**

• Example: suppose x is vector $x = [1, x_1, x_2]^T$ e.g. x_1 might be tumour size and x_2 patient age.



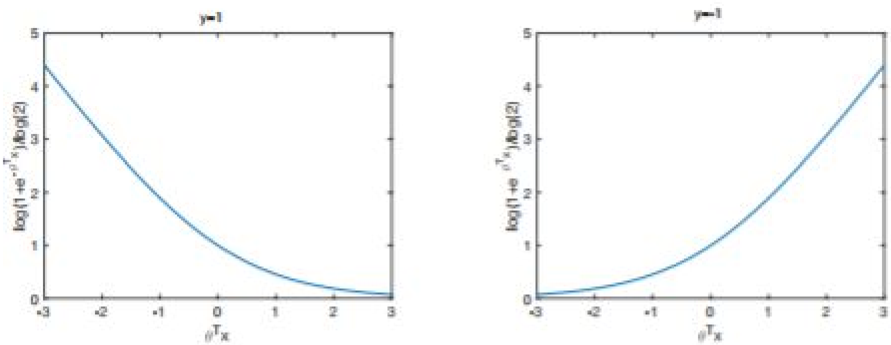
• $\theta_0 = 0$, $\theta_1 = 0.5$, $\theta_2 = -0.5$.

• $h_\theta(x) \geq 0$ when $0.5x_1 - 0.5x_2 \geq 0$ i.e. when $x_1 \geq x_2$.

• When data can be separated in this way we say that it is "linearly separable".

        - Not all data is linearly separable
    - **Choosing cost function**
        - Given training data
            - $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(m)}, y^{(m)})\}$

- $$x \in \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}, x_0 = 1, y \in \{-1, 1\}$$
- Create x_0 and set it to 1 so we can have compatible dimensions for $\theta^T x$
- And our hypothesis:
  - $h_\theta(x) = sign(\theta^T x)$
- How do we choose θ? We **need a cost function**
  - **Maybe the 0-1 loss function**

    $$\frac{1}{m}\sum_{i=1}^{m} \mathbb{I}(h_\theta(x^{(i)}) \neq y^{(i)})$$
    - where indicator function $\mathbb{I} = 1$ if $h_\theta(x^{(i)}) \neq y^{(i)}$ and $\mathbb{I} = 0$ otherwise.
    - **Essentially a count of the number of incorrect predictions, divided by the number of predictions**
    - **Hard to work with**
  - **For logistic regression we use:**
    - $$\frac{1}{m}\sum_{i=1}^{m} \log(1 + e^{-y^{(i)}\theta^T x^{(i)}})/log(2)$$
    - **Noting:** y = -1 or y = 1
    - Scaling by log(2) is optional **but makes loss = 1 when** $y^{(i)}\theta^T x^{(i)} = 0$
    - 
      - Left is y=1, right is y=-1
      - Large penalties occur when prediction is much less than 0 and y = 1 and when prediction is >> 0 and y = -1
      - Minimal when prediction almost equals label
- **So Logistic regression:**

Hypothesis: $h_\theta(x) = sign(\theta^T x)$

Parameters: θ

Cost Function: $J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \log(1 + e^{-y^{(i)}\theta^T x^{(i)}})$

Goal: Select θ that minimises $J(\theta)$

- **Gradient Descent**
  - General concept: Start with some θ, iterate to vector of θ which minimises J(θ)
  - Gradient descent approach:

    Start with some θ

    Repeat:

    for j=0 to n $\{tempj := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)\}$

    for j=0 to n $\{\theta_j := tempj\}$

  - **The gradient descent maths**

    For $J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \log(1 + e^{-y^{(i)}\theta^T x^{(i)}})$:

    - $\frac{\partial}{\partial \theta_j} J(\theta) = \frac{1}{m}\sum_{i=1}^{m} -y^{(i)} x_j^{(i)} \frac{e^{-y^{(i)}\theta^T x^{(i)}}}{1+e^{-y^{(i)}\theta^T x^{(i)}}}$
    - (Remember $\frac{d\log(x)}{dx} = \frac{1}{x}$, $\frac{d\exp(x)}{dx} = exp(x)$ and chain rule $\frac{df(z(x))}{dx} = \frac{df}{dz}\frac{dz}{dx}$)

    So gradient descent algorithm is:

    - Start with some θ
    - Repeat:

      for j=0 to n $\{tempj := \theta_j + \frac{\alpha}{m}\sum_{i=1}^{m} y^{(i)} x_j^{(i)} \frac{e^{-y^{(i)}\theta^T x^{(i)}}}{1+e^{-y^{(i)}\theta^T x^{(i)}}}\}$

      for j=0 to n $\{\theta_j := tempj\}$
    - $J(\theta)$ is convex, has a single minimum. Iteration moves downhill until it reaches the minimum

  - So using Logistic regression in practice
    - **Similar to linear regression**
    - **Select learning rate α to ensure convergence in reasonable time**
    - **Use cross-validation** to select what features to include/exclude from x
    - **Use regularisation to incorporate prior knowledge of constraints** on parameters θ e.g. add penalty on θ^2 by changing cost function to:

      $$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \log(1 + e^{-y^{(i)}\theta^T x^{(i)}}) + \lambda \sum_{j=1}^{n} \theta_j^2$$

- **Logistic Regression with Multiple Classes**
  - Examples
    - Filtering emails into folders, classifying weather
  - Now our y output takes values 0,1,2…
    - 0 if it's sunny, 1 if it's cloudy…

- **Make it into a binary problem**
    - Train a classifier $h_\theta^{(i)}(x)$ for each class i
    - Predicts the probability that y = i
    - Training data: re-label data as y = -1 when y $\neq$ i and y = 1 when y = i