Overview

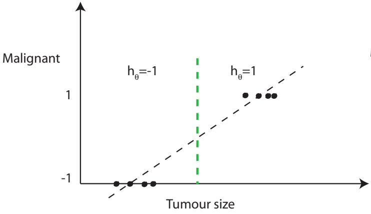
- Classification
- Logistic Regression with Two Classes
- Gradient Descent
- Logistic Regression with Multiple Classes
- Probabilistic Interpretation

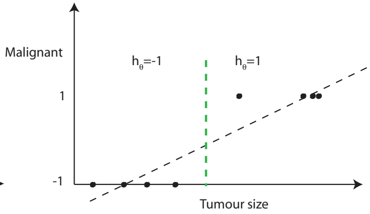
Classification with Two Classes

- Examples:
  - Email: spam or not ?
  - Online transactions: fraudulent or not ?
  - Tumor: malignant or benign ?
- As before x= "input" variable/features e.g. text of email, location, nationality
- Now y= "output" variable/ "target" variable only takes values -1 or 1 (with linear regression y was real valued). In classification y often referred to as the **label**[1].
- We want to build a **classifier** that predicts the label of a new object e.g whether a new email is spam or not.

[1] Note could also use values 0 and 1 rather than -1 and 1, leave this as an exercise

Logistic Regression: Choice of Hypothesis



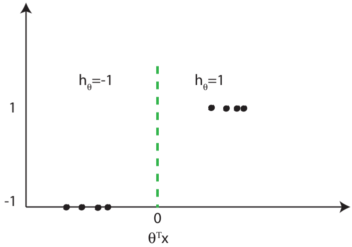


- Fitting data with a straight line $\theta^T x$ doesn't look appropriate (prone to misclassification)
- Predict output 1 when $\theta^T x \geq 0$ and output -1 when $\theta^T x < 0$ i.e.

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

Logistic Regression: Decision Boundary

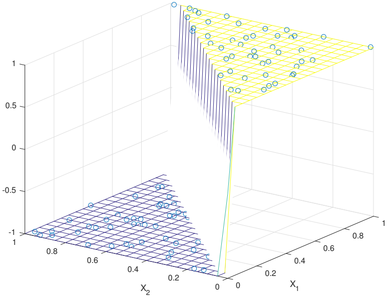
We can think of logistic regression as trying to fit a plane that separates the $Y = 1$ data from the $Y = 0$ data.

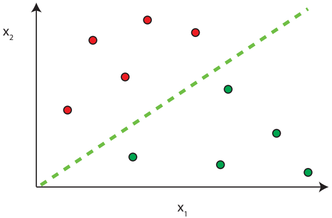


- $\theta^T x = 0$ defines a point in one dimension e.g. $1 + 0.5x_1 = 0 \rightarrow x_1 = -2$ ...
- ... a line in two dimensions e.g. $2 + x_1 + 2x_2 = 0 \Rightarrow x_2 = -x_1/2 - 1$ ...
- .. and a plane in higher dimensions

Logistic Regression: Decision Boundary

- 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.

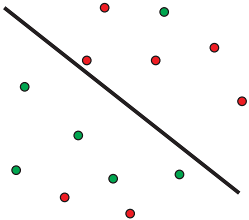




- $\theta_0 = 0$, $\theta_1 = 0.5$, $\theta_2 = -0.5$.
- $h_\theta(x) \geq 0$ when $0.5x_1 - 0.5x_2 \geq 0$ i.e. when $x_1 \geq x_2$.
- When data can be separated in this way we say that it is "linearly separable".

Logistic Regression: Decision Boundary

- Not all data is linearly separable e.g.



Logistic Regression: Choice of Cost Function

- Training data: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(m)}, y^{(m)})\}$
- $x \in \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}$, $x_0 = 1$, $y \in \{-1, 1\}$
- Hypothesis: $h_\theta(x) = sign(-\theta^T x)$
- How to choose parameters θ ?

Logistic Regression: Choice of Cost Function

- We might consider 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. But hard to work with.

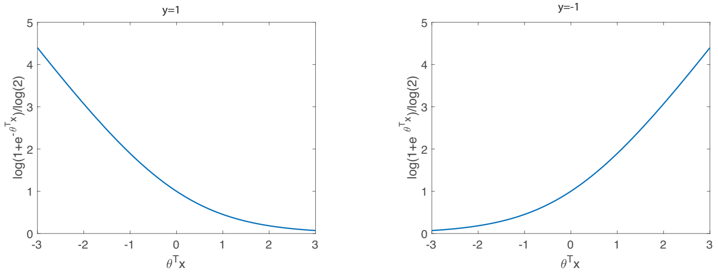
- For logistic regression we use:

$$\frac{1}{m}\sum_{i=1}^{m}\log(1 + e^{-y^{(i)}\theta^T x^{(i)}})/log(2)$$

  noting that $y = -1$ or $y = 1$. Scaling by $\log(2)$ is optional, but makes the loss 1 when $y^{(i)}\theta^T x^{(i)} = 0$.

Logistic Regression: Choice of Cost Function

Loss function: $\log(1 + e^{-y\theta^T x})/log(2)$



- So a small penalty when $\theta^T x \gg 0$ and $y = 1$, and when $\theta^T x \ll 0$ and $y = -1$.
- Minimising this thus gives preference to θ values that push $\theta^T x$ well away from the decision boundary $\theta^T x = 0$.

Summary

- 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

As before, can find θ using:

- Start with some θ
- Repeat:

  Update vector θ to new value which makes $J(\theta)$ smaller

e.g using gradient descent:

- 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 $\{\theta_j := tempj\}$

Gradient Descent

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

Probabilistic Interpretation: Logistic Regression

- Label Y only takes values -1 or 1.
- Assume

$$P(Y = y|\theta, x) = \frac{1}{1 + e^{-y\theta^T x}}$$

and recall $y = 1$ or $y = -1$ only.

- The **likelihood** $P(d|\theta)$ of the training data d is therefore:

$$P(d|\theta) = \prod_{i=1}^{m} \frac{1}{1 + e^{-y\theta^T x}}$$

- Taking logs:

$$\log P(d|\theta) = \sum_{i=1}^{m} \log \frac{1}{1 + e^{-y\theta^T x}}$$

- And the maximum likelihood estimate of θ minimises:

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

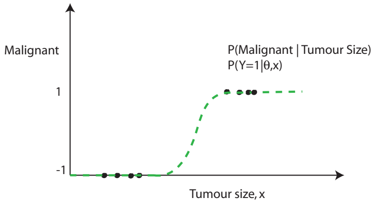
since $-\log(z) = \log(1/z)$.

Probabilistic Interpretation: Logistic Regression

- The probabilistic formulation of logistic regression provides us with a new insight:

$$P(Y = y|\theta, x) = \frac{1}{1 + e^{-y\theta^T x}}$$

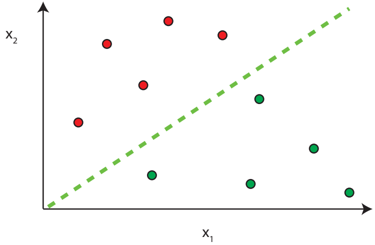
- So in addition to prediction $h_\theta(x) = sign(\theta^T x)$ we also have an estmate of our confidence in the prediction, namely $\frac{1}{1+e^{-y\theta^T x}}$.

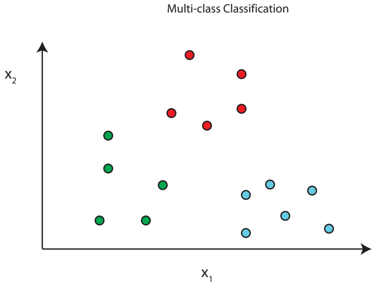


- When $\frac{1}{1+e^{-y\theta^T x}}$ is close to 1, then we are confident in our prediction but when $\frac{1}{1+e^{-y\theta^T x}}$ is small then we are less confident.

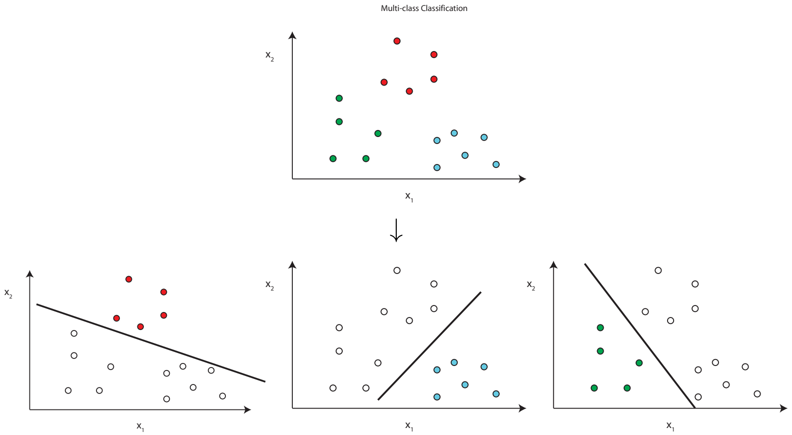
Logistic Regression With Multiple Classes

- Examples:
  - Email folder tagging: work, friends, family, hobby
  - Weather, sunny, cloudy, rain, snow
  - Given where I live in Dublin, predict which political party I'll vote for.
- Now y="output" variable/"target" variable takes values 0,1,2,. . . .
  E.g. $y = 0$ if sunny, $y = 1$ if cloudy, $y = 2$ if rain etc.





Logistic Regression With Multiple Classes



- Train a classifier $h_\theta^{(i)}(x)$ for each class i to predict the probability that $y = i$. Training data: re-label data as $y = -1$ when $y \neq i$ and as $y = 1$ when $y = i$, so we're back to a binary classification task.