Logistic Regression

- Algorithm used for classification problems

Classification Problems

- Email: spam / not spam
- Bank Transactions: fraudulent / not fraudulent
- Tumors: Malignant / benignant
  - Malignant: 1 (pos.)
  - benignant: 0 (neg.)

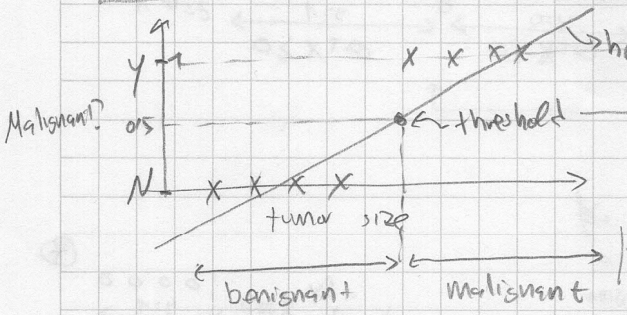
→ arbitrarily, the negative class for absence (0), the positive class for presence (1)

$y \in \{0,1\}$
- 0: Negative class
- 1: Positive class

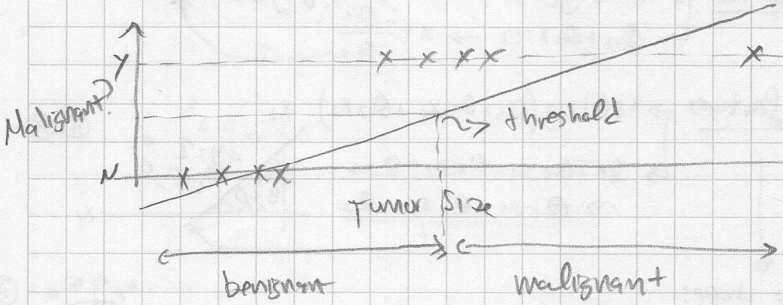
← we want to predict the variable y that takes two values {0, 1 / true, false}

(there are multi-class classification problems where $y \in \{0, 1, 2, 3, \dots\}$)

Classifying using Linear Regression? → not the best option...



linear LR seems to work with this dataset !



linear LR does not work with this dataset !

Another problem with using LR:

$y = 0$ or 1

$h_\theta(x)$ can be > 1 or < 0 → confusing!

we will use:

Logistic Regression: $0 \leq h_\theta(x) \leq 1$

↓

it is confusing to have 'regression' in the name of a classification algorithm

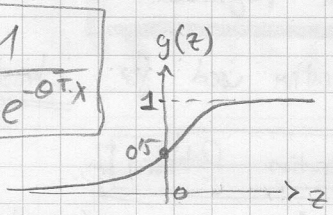
Hypothesis in Logistic Regression

Model

We want $0 \leq h_\theta(x) \leq 1$

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

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



Sigmoid function or Logistic function

- When looking for best θ, this $h_\theta(x)$ with the $g(z)$ will be used

Interpretation

- If $h_\theta(x) = 0'7$, means that we have a 70% (0'7) probability that the output is 1 ($y=1$)
  - formally: $h_\theta(x) = P(y=1 \mid x; \theta)$ → probability that $y=1$, given x, and parametrized by θ

Decision Boundary

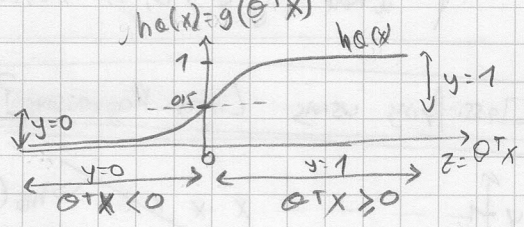
$\{$ we predict $y=1$ if $h_\theta(x) \geq 0'5$

" " $y=0$ if $h_\theta(x) < 0'5$

$\Downarrow$

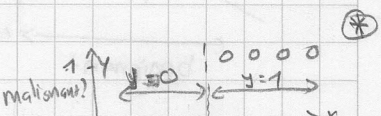
$\{$ we predict $y=1$ if $\theta^T \cdot X \geq 0$

" " $y=0$ if $\theta^T \cdot X < 0$

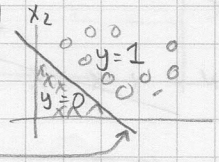


↳ depending on the number of features, this boundary has a different number of dimension

- 1 feature → $h_\theta(x) = g(\theta_0 + \theta_1 \cdot x)$
  ↳ $\theta_0 + \theta_1 x = 0 \rightarrow x = \frac{-\theta_0}{\theta_1}$

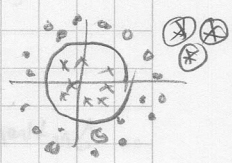


- 2 features → $h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$
  ↳ $\theta_0 + \theta_1 x_1 + \theta_2 x_2 = 0 \Rightarrow \theta_1 x_1 + \theta_2 x_2 = -\theta_0$

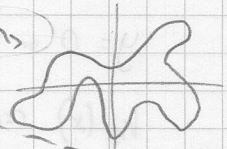


any shape:

$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2) \rightarrow$



$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \dots) \rightarrow$

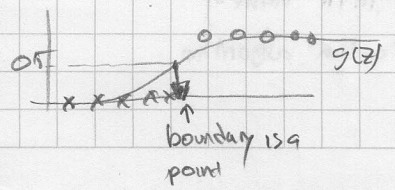


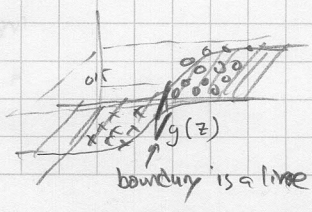
Note that the decision boundary depends on a set of parameters θ, it has nothing to do with the data-set

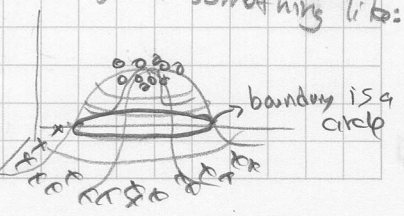
In order to select the best set of parameters θ, we will use the data-set though...

⊛, ⊛⊛, ⊛⊛⊛

I think working with $g(z)$, when looking for the best θ, we are looking at something like:







Cost function - How to choose θ

Training set $\begin{cases} (x^{(1)}, y^{(1)}) \\ (x^{(2)}, y^{(2)}) \\ \vdots \\ (x^{(m)}, y^{(m)}) \end{cases}$ m examples

$x = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}$ $x_0 = 1$ (always), $n+1$ (n features) — feature vector

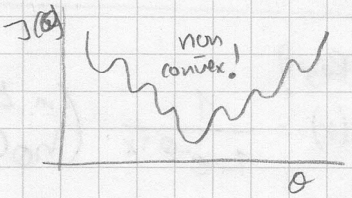
$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix}$ parameter vector

$y \in \{0, 1\}$ output

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

How to choose θ?

- We still want to minimize a $J(\theta)$ function $J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \text{Cost}(h_\theta(x), y)$ (average cost of all samples)
- If we use the same cost function as in LinReg ($\text{Cost}(h_\theta(x), y) = \frac{1}{2}(h_\theta(x) - y)^2$), it won't work properly because $J(\theta)$ becomes non-convex:

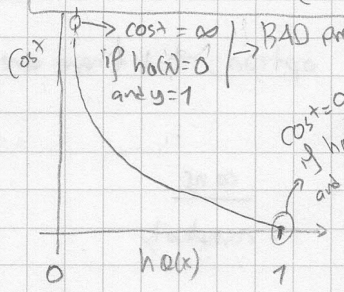


← note that in LogReg $h_\theta(x) = g(\theta^T x) = \frac{1}{1+e^{-\theta^T x}}$, this fact makes it non-convex

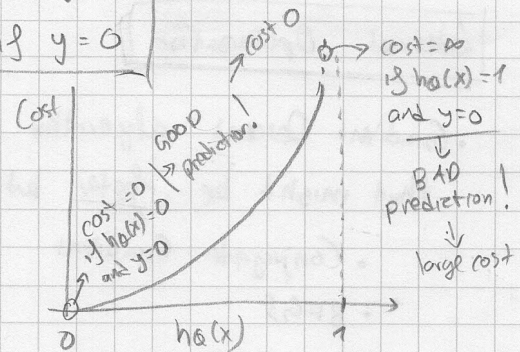
← Gradient-descent might end up in a local minimum

- We are going to use this cost function:

$$\text{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$ ← cost of a single sample



if $y = 1$



if $y = 0$

- $J(\theta) = \frac{1}{2} \sum_{i=1}^{m} \text{Cost}(h_\theta(x), y)$ (J is average cost of all samples), with this cost function, becomes convex! :) (not demonstrated, beyond objective of the course)

- Reformulation:

(single sample →) $\text{Cost}(h_\theta(x), y) = -y \cdot \log(h_\theta(x)) - (1-y) \log(1 - h_\theta(x))$ ← these terms are either 0 or 1.

(all samples →) $J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} \log(h_\theta(x)) + (1-y^{(i)}) \log(1 - h_\theta(x)) \right]$

- To fit parameters θ: $\min_\theta J(\theta)$ → get θ
- To predict given new x: $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}} = p(y=1 | x; \theta)$

Logistic Regression & Gradient Descent

We will use Gradient Descent to minimize $J(\theta)$ and obtain the best θ parameters.

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} y \cdot \log\left(h_\theta(x)\right) + (1-y)\log\left(1-h_\theta(x)\right)\right]$$

$\min_\theta J(\theta)$: Repeat {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

(simultaneously update all θ_j)

}

$$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x) - y\right)x_j$$

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

$\min_\theta J(\theta)$: Repeat {

$$\theta_j := \theta_j - \alpha \cdot \sum_{i=1}^{m}\left(h_\theta(x) - y\right)\cdot x_j$$

(simultaneously update all θ_j)

}

‼ same algorithm as in LinReg !

The difference is that now $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$ (in LinReg it was $h_\theta(x) = \theta^T x$)

- Since we apply Gradient Descent, feature scaling is also useful here.

Advanced Optimization

- Gradient Descent algorithm is the simplest optron, but there are others that might be faster but more complex:
  - Conjugate Gradient
  - BFGS
  - L-BFGS

pros	cons
• faster than GD • no need to choose α	• complexity

- Can be used as a 'black box' without understanding what they really do.
  - you just need to pass a cost function to these algorithms...

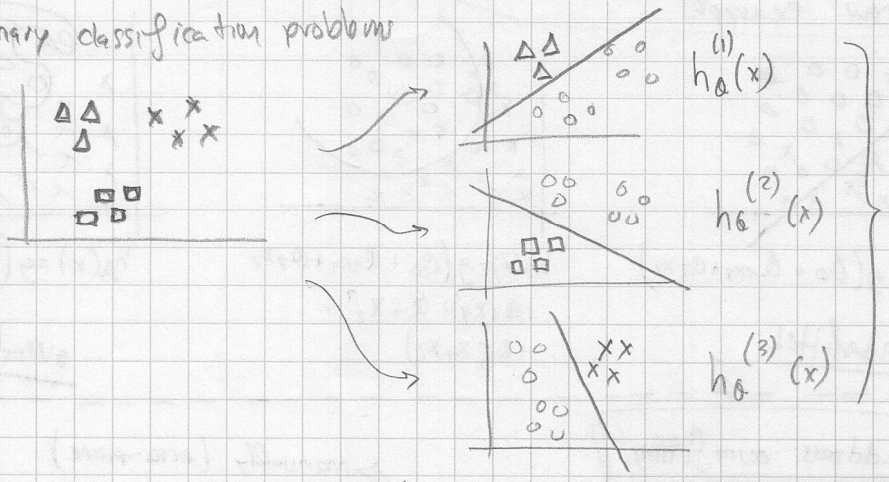
Multiclass classification

- Email folders: work, friends, family, hobby,
  $y=1$ $y=2$ $y=3$ $y=4$
- Medical diagrams: not ill, cold, flu
  $y=0$ $y=1$ $y=2$

⋮

One-vs-all (or One-vs-rest)

- The idea is to convert the multiclass classification problem to several binary classification problems



- So we separate and try to predict the probability that an example belongs to a particular class; and we compute this probability for each class.
- When we predict, we predict the class that gives the maximum probability.

Youtube Lecture 4 →

Newton's Method for Log Reg

- An alternative algorithm to minimize $J(\theta)$ that works well with Logistic Regression
- Quite fast and technical description in the video...