Classification

julien.brajard@upmc.fr

UPMC

1-5 August 2016

Classification problems

Definition

A classification problems is a prediction problem where the value to predict y is categorical.

Classification problems

Definition

A classification problems is a prediction problem where the value to predict y is categorical.

Examples :

- Identify manuscript digits (10 classes)
- Identify a label on a image (multiclass)

We will begin with binary classification where there is only two classes ($y \in \{0, 1\}$) :

- Healthy (0) / Sick (1)
- Not Spam / Spam (for emails)
- ...

Why linear regression is not applicable ?

We could imagine using the linear regression :

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

and then apply a threshold to the output of the linear regression :

- If $h_\theta(x) \geq 0.5$, predict 1
- If $h_\theta(x) < 0.5$, predict 0.

Why linear regression is not applicable ?

We could imagine using the linear regression :

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

and then apply a threshold to the output of the linear regression :

- If $h_\theta(x) \geq 0.5$, predict 1
- If $h_\theta(x) < 0.5$, predict 0.

Why is it not satifactory ?

- Not robust (unexpected behaviors)
- linear regression allow $h_\theta(x) < 0$ or $h_\theta(x) > 1$.

The Logistic regression Model

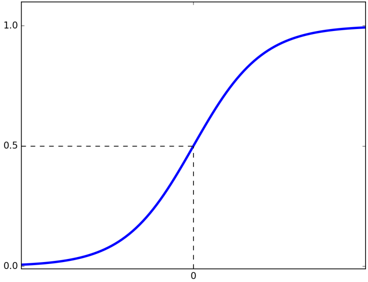
We modify the hypothesis function :

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

where

$$g(z) = \frac{1}{1 + \exp(-z)}$$

is the so-called sigmoid function or logistic function.



Probabilistic interpretation

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

$h_\theta(x)$ can be interpreted as the probability that $y = 1$ knowing x :

$$h_\theta(x) = P(Y = 1|x; \theta)$$

Probabilistic interpretation

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

$h_\theta(x)$ can be interpreted as the probability that $y = 1$ knowing x :

$$h_\theta(x) = P(Y = 1|x; \theta)$$

Example :

If $x = \begin{pmatrix} 1 \\ \text{body temperature} \end{pmatrix}$, $h_\theta(x) = 0.6$ means that the probability to be sick given the temperature (and the paramter θ) is 0.6.

Note : Consequently, we have, $P(Y = 0|x; \theta) = 1 - h_\theta(x)$

Predicting the output y

In order to predict an ouptut $y \in \{0; 1\}$, the following rule is used :

- if $h_\theta(x) \geq 0.5$, we predict 1.
- if $h_\theta(x) < 0.5$, we predict 0.

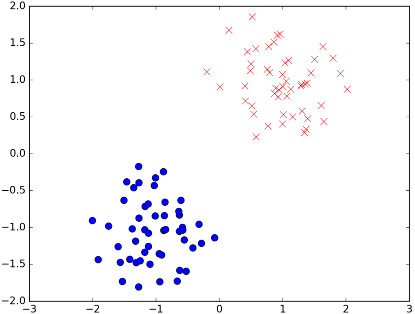
Predicting the output y

In order to predict an ouptut $y \in \{0; 1\}$, the following rule is used :

- if $h_\theta(x) \geq 0.5$, we predict 1.
- if $h_\theta(x) < 0.5$, we predict 0.

Note : $h_\theta(x) \geq 0.5 \Leftrightarrow \theta^T x \geq 0$

Decision boundary

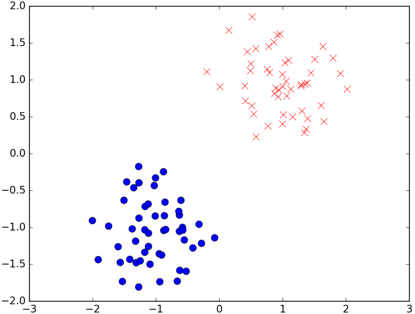


$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

with $\boldsymbol{\theta} = (0, 1, 1)^T$

The system predicts $y = 1$ if $x_1 + x_2 \geq 0$

Decision boundary



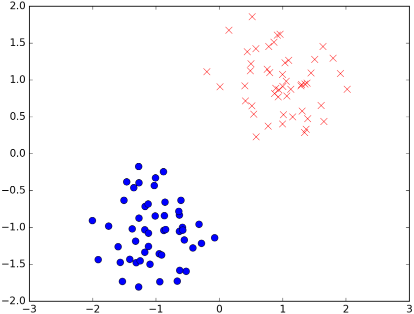
$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

with $\boldsymbol{\theta} = (0, 1, 1)^T$

The system predicts $y = 1$ if $x_1 + x_2 \geq 0$

The Decision boundary is the line defined by the equation $x_1 + x_2 = 0$

Decision boundary



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

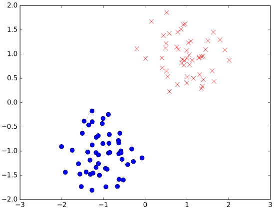
with $\boldsymbol{\theta} = (0, 1, 1)^T$

The system predicts $y = 1$ if $x_1 + x_2 \geq 0$

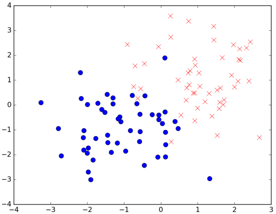
The Decision boundary is the line defined by the equation $x_1 + x_2 = 0$

Using logistic regression, the boundary is always linear (an hyperplane of the space)

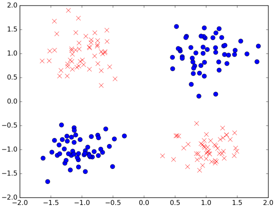
Limits of the linear decision boundary



Linear decision boundary perfectly discrimates the two classes



Linear decision boundary is still relevant but some mis-classification can occur.



Linear decision boundary is irrelevant, the logistic regression will fail.

Choosing the parameters θ

- The training set $\{(x_1, y_1), \dots, (x_n, y_n)\}$
- The hypothesis representation : $\hat{y} = h_\theta(x) = g(\theta^T x)$, $\theta \in \mathbb{R}^p$

The determination of the "optimal" θ is made by minimizing a cost function :

$$J(\theta) = \frac{1}{n} \sum_{i=1}^{n} D(h_\theta(x_i), y_i)$$

where D is a misfit function between the target output y and the output of the classifier $h_\theta(x)$

Choosing the parameters θ

- The training set $\{(x_1, y_1), \ldots, (x_n, y_n)\}$
- The hypothesis representation : $\hat{y} = h_\theta(x) = g(\theta^T x)$, $\theta \in \mathbb{R}^p$

The determination of the "optimal" θ is made by minimizing a cost function :

$$J(\theta) = \frac{1}{n} \sum_{i=1}^{n} D(h_\theta(x_i), y_i)$$

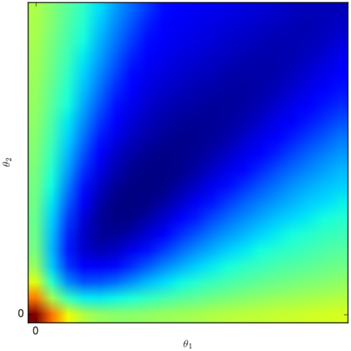
where D is a misfit function between the target output y and the output of the classifier $h_\theta(x)$

Note, if we take the least-square cost function : $D(\hat{y}, y) = \frac{1}{2}(\hat{y} - y)^2$, the cost function is equivalent to the linear regression cost function.

Is it relevant to take the least-square function ?

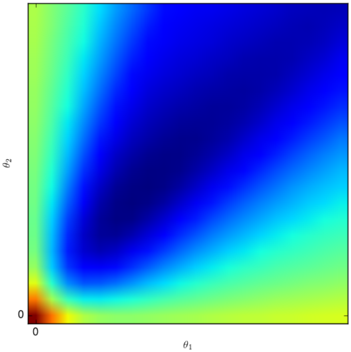
Problem with the least-square cost function

- First problem (practical) : the function is non-convex and the minimum is unclear



Problem with the least-square cost function

- First problem (practical) : the function is non-convex and the minimum is unclear



- Second problem (theorical) : the mimimum of this cost function is not the maximum likekihood estimator

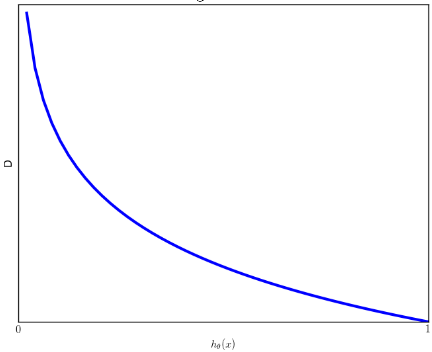
Logistic regression cost function

A intuitive definition

For logistic regression, the cost function is contructed as follow :

- $D(h_\theta(x), y = 1) = -\log(h_\theta(x))$

If $y = 1$



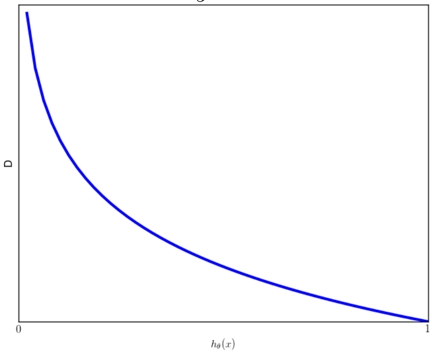
Logistic regression cost function

A intuitive definition

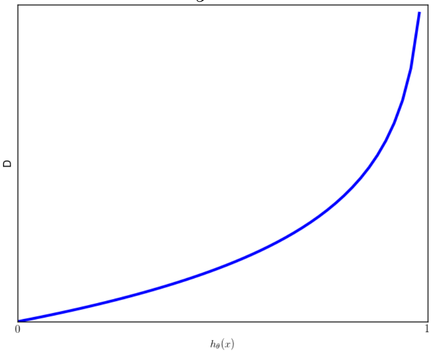
For logistic regression, the cost function is contructed as follow :

- $D(h_\theta(x), y = 1) = -\log(h_\theta(x))$
- $D(h_\theta(x), y = 0) = -\log(1 - h_\theta(x))$

If $y = 1$



If $y = 0$



Logistic cost function

A compact definition

The logistic cost function can be rewritten :

$$D(h_\theta(x), y) = -y \log(h_\theta(x)) - (1-y)\log(1 - h_\theta(x))$$

- The value θ that minimize this cost function is the maximum of likelkihood (maximum of $P(y|x, \theta)$)

Logistic cost function

A compact definition

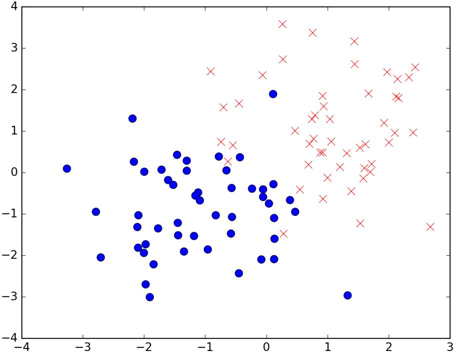
The logistic cost function can be rewritten :

$$D(h_\theta(x), y) = -y\log(h_\theta(x)) - (1-y)\log(1-h_\theta(x))$$

- The value θ that minimize this cost function is the maximum of likelkihood (maximum of $P(y|x,\theta)$)
- If we consider that $h_\theta(x)$ is a probability q_1 of the prediction $\hat{y}$ to be one. We have a "true" probability $p_{y=1} = y$. Minimizing D is equivalent to minimizing to cross-entropy between p and q.

The cost function

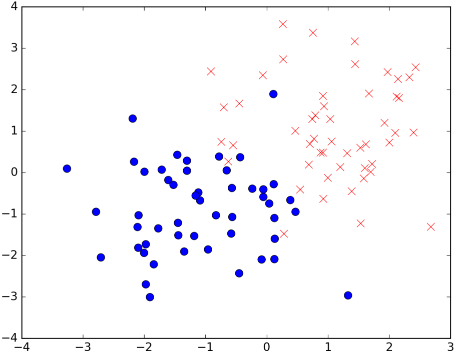
Points to classify :



model : $h_\theta(x) = \theta_1 x_1 + \theta_2 x_2$

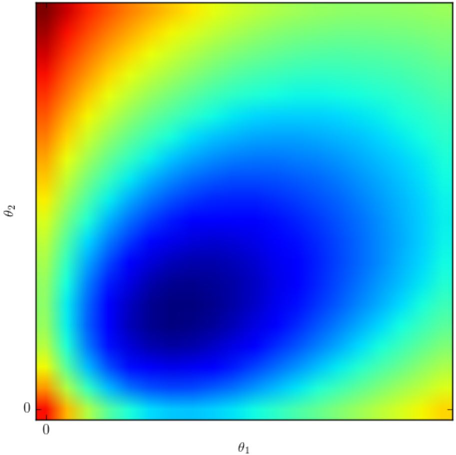
The cost function

Points to classify :



model : $h_\theta(x) = \theta_1 x_1 + \theta_2 x_2$

The cost function to minimize :



Gradient descent

To minimize the cost function, the gradient descent algorithm can be used in the same way as for the linear regression.
The gradient of the cost function $J(\theta)$:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \begin{pmatrix} \frac{\partial J}{\partial \theta_0} \\ \vdots \\ \frac{\partial J}{\partial \theta_p} \end{pmatrix}$$

Gradient descent

To minimize the cost function, the gradient descent algorithm can be used in the same way as for the linear regression.
The gradient of the cost function $J(\theta)$:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \begin{pmatrix} \frac{\partial J}{\partial \theta_0} \\ \vdots \\ \frac{\partial J}{\partial \theta_p} \end{pmatrix}$$

After calculation, we obtain :

$$\frac{\partial J}{\partial \theta_i} = -\frac{1}{m} \sum_{j}^{n} (y_j - h_\theta(x_j)) x_{ij}$$

Gradient algorithm (batch gradient)

1. Intialize $\theta = \theta^0$
2. Repeat N times the following instructions :
   1. Initiatialize the gradient $G = 0$
   2. Repeat n times
      - $G = G - (y_j - h_\theta(x_j)).x_{ij}$
   3. Update the parameter : $\theta = \theta - \nu G$
3. return θ

Problems with the gradient algorithm

- The method is sensitive to different order of magnitude for the components of x.

Problems with the gradient algorithm

- The method is sensitive to different order of magnitude for the components of x.
- There is an hyper-parameter to determine : the learning rate ν

Problems with the gradient algorithm

- The method is sensitive to different order of magnitude for the components of x.
- There is an hyper-parameter to determine : the learning rate ν
- A stopping criteria has to be found to confirm the convergence

Problems with the gradient algorithm

- The method is sensitive to different order of magnitude for the components of x.
- There is an hyper-parameter to determine : the learning rate ν
- A stopping criteria has to be found to confirm the convergence
- For large samples, the convergence is very long

Scaling of x

A easy way to avoid some problems of different order of magnitudes of $\boldsymbol{x}$ component, variables can be re-scaled before the regression process.

Scaling of x

A easy way to avoid some problems of different order of magnitudes of x component, variables can be re-scaled before the regression process.

Option 1 : reduction

Let's define $m_i = \frac{1}{n}\sum_{j=1}^{n} x_{ij}$ and $s_i = \frac{1}{n-1}\sum_{j=1}^{n}(x_{ij} - m_i)^2$.
Reduction :

$$\tilde{x}_i = \frac{x_i - m_i}{s_i}$$

Scaling of x

A easy way to avoid some problems of different order of magnitudes of x component, variables can be re-scaled before the regression process.

Option 1 : reduction

Let's define $m_i = \frac{1}{n}\sum_{j=1}^{n} x_{ij}$ and $s_i = \frac{1}{n-1}\sum_{j=1}^{n}(x_{ij} - m_i)^2$.
Reduction :

$$\tilde{x}_i = \frac{x_i - m_i}{s_i}$$

Option 2 : minmax

The scaled parameter is stricly comprised between 0 and 1 :

$$\tilde{x}_i = \frac{x_i - min(x_i)}{max(x_i) - min(x_i)}$$

The learning rate ν

It is possible to use more complex algorithms that doesn't need to specifiy a learning rate (or that optimizes the learning rate during the minimization)

- Conjugate gradient
- BFGS (Broyden-Fletcher-Goldfarb-Shanno)
- L-BFGS (Limited Memory BFGS)

Stochastic gradient algorithm

During the gradient process, the gradient is calculated n times by iterations, and the parameters are updated only 1 time by iteration.
To allow a more frequent update of the parameter, it is possible to update the parameters each time a regressor x_i is calculated

1. Intialize $\theta = \theta^0$
2. Repeat N times the following instructions :
    1. Repeat n times
        - $G = G - (y_j - h_\theta(x_j)).x_{ij}$
        - Update the parameter : $\theta = \theta - \nu G$
3. return θ

Remarks on the logistic gradient

Advantages

- It can converge faster
- It can make a natural use of GPU (Graphical Processor Unites) for the computation

Remarks on the logistic gradient

Advantages

- It can converge faster
- It can make a natural use of GPU (Graphical Processor Unites) for the computation

Drawbacks

It is unstable near the optimum

"mini-batch gradient"

Let's define the size of a "small" amount of data, call a "mini-batch" size, denoted m

1. Intialize $\theta = \theta^0$
2. Repeat N times the following instructions :
   1. Initilalize the gradient $G = 0$
   2. Repeat n times
      - ★ $G = G - (y_j - h_\theta(x_j)).x_{ij}$
      - ★ Every m iteration :
        Update the parameter : $\theta = \theta - \nu G$ and
        Set the gradient $G = 0$
3. return θ

Remark

- If $m = 1$, it is equivalent to the stochastic gradient algorithm.
- If $m = n$, it is equiivalent to the batch gradient algorithm

Demonstration

Large batch
Stochastic algorithm