Apprentissage Artificiel

Logistic Regression

Ewa Kijak

ESIR/Univ. Rennes

Outline

Sommaire

Logistic regression is classification

- Logistic regression $\neq$ Linear regression
- Logistic regression is a Generalized Linear Model (GLM)

Univariate linear regression

Example : Housing Prices

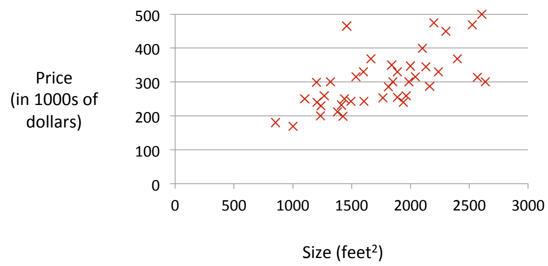
- m = Number of training examples
- $x^{(i)}$ = "input" variable / features of the i-th example
- $y^{(i)}$ = "output" variable / "target" variable of the i-th example

Size in feet² (x)	Price ($) in 1000's (y)
2104	460
1416	232
1534	315
852	178
...	...

Introductive examples borrowed from Andrew Ng

Univariate linear regression

- **Supervised learning** : Give the "right answer" for each example in the data.
- **Regression problem** : Predict real-valued output
- $\neq$ **Classification problem** : Discrete-valued output



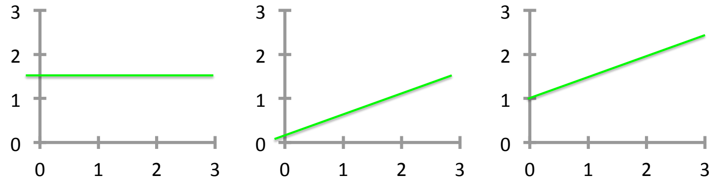
Univariate linear regression

- Univariate linear regression = Linear regression with one variable (x).
- Hypothesis : $h_\theta(x) = \theta_0 + \theta_1 x$
- Parameters : θ_0, θ_1

How to choose θ_i ?

Univariate linear regression

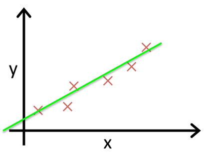
$h_\theta(x) = \theta_0 + \theta_1 x$



$\theta_0 = 1.5$ $\theta_1 = 0$ | $\theta_0 = 0$ $\theta_1 = 0.5$ | $\theta_0 = 1$ $\theta_1 = 0.5$

Univariate linear regression

Idea : Choose θ_0, θ_1 so that $h_\theta(x)$ is close to y for the training examples $(x^{(i)}, y^{(i)})$



- Cost function :

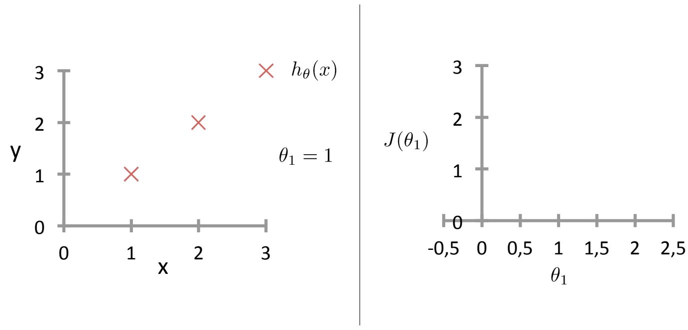
$$J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

- Goal :

$$\underset{\theta_0,\theta_1}{\text{minimize}}\, J(\theta_0, \theta_1)$$

Univariate linear regression

Simplified case (example) : $h_\theta(x) = \theta_1 x$



Multivariate linear regression

- Multivariate linear regression = Linear regression with multiple features (x_j).
  - d = number of features
  - $x^{(i)}$ = "input" variable / features of the i-th example
  - $x_j^{(i)}$ = value of feature j in the i-th example

Example : Housing Prices

Size (feet²)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178
...	...	...	...	...

Multivariate linear regression

- Hypothesis : $h_{\boldsymbol\theta}(\mathbf{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_d x_d$
- For convenience of notation, we define $x_0 = 1$.

Then :

- $\mathbf{x} = [x_0, x_1, x_2, \cdots, x_d]^T \in \mathbb{R}^{d+1}$
- $\boldsymbol\theta = [\theta_0, \theta_1, \theta_2, \cdots, \theta_d]^T \in \mathbb{R}^{d+1}$

$$h_{\boldsymbol\theta}(\mathbf{x}) = \boldsymbol\theta^T \mathbf{x}$$

Sommaire

Logistic regression

- Linear **classification** model
- Let $\boldsymbol{x} \in \mathbb{R}^d$
- Linear predictor :

$$l = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_d x_d$$

- For convenience of notation, we define $x_0 = 1$ (then $\boldsymbol{x} \in \mathbb{R}^{d+1}$) :

$$l = \boldsymbol{\theta}^T \boldsymbol{x}$$

where $\boldsymbol{\theta}^T = [\theta_0, \theta_1, \cdots, \theta_d] \in \mathbb{R}^{d+1}$ are the parameters of the model
- Binary classification task : $y \in \{0, 1\}$

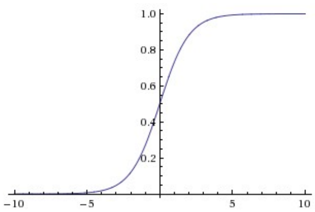
Logistic regression model

- Hypothesis : $h_\theta(\boldsymbol{x}) = g(\boldsymbol{\theta}^T \boldsymbol{x})$

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

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

$$0 \leq h_\theta(\boldsymbol{x}) \leq 1$$



Sigmoid function = Logistic function

Interpretation of hypothesis output

$h_\theta(\boldsymbol{x})$ = estimated probability that y=1, given $\boldsymbol{x}$, parametrized by $\boldsymbol{\theta}$

$$h_\theta(\boldsymbol{x}) = P(y = 1|\boldsymbol{x}, \boldsymbol{\theta}) = \frac{1}{1+\exp(-\boldsymbol{\theta}^T \boldsymbol{x})},$$

which implies that :

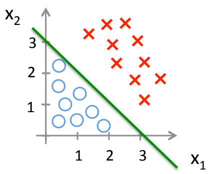
$$P(y = 0|\boldsymbol{x}, \boldsymbol{\theta}) = 1 - P(y = 1|\boldsymbol{x}, \boldsymbol{\theta}) = \frac{1}{1+\exp(\boldsymbol{\theta}^T \boldsymbol{x})}$$

Decision boundary
- if $h_\theta(\boldsymbol{x}) \geq 0.5$, predict "y=1"
- if $h_\theta(\boldsymbol{x}) < 0.5$, predict "y=0"

Because $g(Z) \geq 0.5$ when $z \geq 0$,

$$\boxed{y = 1 \Leftrightarrow \boldsymbol{\theta}^T \boldsymbol{x} \geq 0}$$

Decision boundary



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

Predict "$y = 1$" if $-3 + x_1 + x_2 \geq 0$

- if $x_1 + x_2 \geq 3$, predict "y=1"
- if $x_1 + x_2 < 3$, predict "y=0"

How to choose parameters θ ?

Sommaire

Find parameters $\boldsymbol{\theta}$

Minimization of a cost function

- We could define :

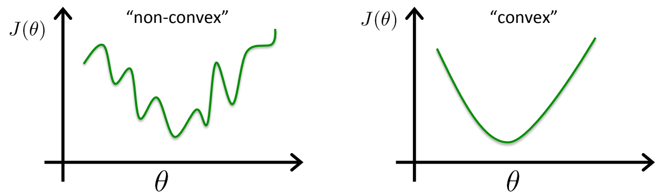
$$J(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m} \text{loss}(h_\theta(\mathbf{x}^{(i)}), y^{(i)})$$

with :

$$\text{loss}(h_\theta(\mathbf{x}), y) = \frac{1}{2}(h_\theta(\mathbf{x}) - y)^2$$

- But : $J(\boldsymbol{\theta})$ non-convex function of the parameters $\boldsymbol{\theta}$! (because $h_\theta(\mathbf{x})$ non-linear function)
- Solution : change the cost function to a convex one.

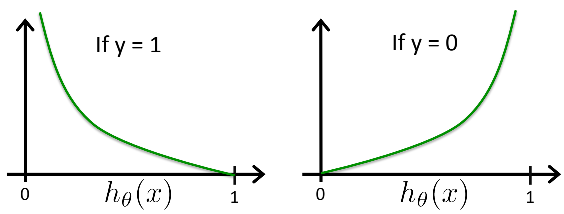
Convexity



- Convex functions : can be solved quickly and reliably up to very large scale
  - e.g. gradient descent, Lagrange method

Find parameters $\boldsymbol{\theta}$

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



- If $h_\theta(\mathbf{x}) = P(y = 1|\mathbf{x}, \boldsymbol{\theta}) = 0$, but $y = 1$, the learning algorithm will be penalized with a very large cost

Logistic regression cost function

To fit parameters $\boldsymbol{\theta}$: $\min_\theta J(\boldsymbol{\theta})$

$$J(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m} \text{loss}(h_\theta(\mathbf{x}^{(i)}), y^{(i)})$$
$$= -\frac{1}{m}\sum_{i=1}^{m} y^{(i)}\log(h_\theta(\mathbf{x}^{(i)})) + (1 - y^{(i)})\log(1 - h_\theta(\mathbf{x}^{(i)}))$$

The cost function is :

- convex
- derived from maximum likelihood estimation
- also called **cross-entropy** error function

Logistic regression cost function

$$\min_\theta J(\boldsymbol{\theta}) = -\frac{1}{m}\sum_{i=1}^{m} y^{(i)}\log(h_\theta(\mathbf{x}^{(i)})) + (1 - y^{(i)})\log(1 - h_\theta(\mathbf{x}^{(i)}))$$

Solution : gradient descent

- $\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\boldsymbol{\theta})$
- $\frac{\partial}{\partial\theta_j}J(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(\mathbf{x}^{(i)}) - y^{(i)})x_j^{(i)}$
- simultaneously update all θ_j

Sommaire

Going further

- Gradient descent parameters and properties
- Underfitting and adding features to get non linear classifier
- Overfitting and regularization
- Multinomial logistic regression

Problem of underfitting/overfitting

- **High bias** or **underfitting** is when the form of our hypothesis maps poorly to the trend of the data. It is usually caused by a function that is too simple or uses too few features.
- At the other extreme, **overfitting** or **high variance** is caused by a hypothesis function that fits the available data but does not generalize well to predict new data. It is usually caused by a complicated function that creates a lot of unnecessary curves and angles unrelated to the data.

Basis expansion 1

- Data is likely to be non-linearly separable
- What if we still wanted to use a linear regression ?
- How to marry non-linear data to a linear method ?

The trick is to **transform the data** : Map the data onto another features space, such that the data is linear in that space.

$\rightarrow$ Including higher order terms **increases the capacity/complexity of the model** : it allows to learn decision boundaries that would be unreachable using simply the original features. This is because a linear decision boundary (which is what logistic regression fits) learned on nonlinear transformations of features will ultimately be nonlinear in terms of the original features.

Basis expansion 2

- Denote this transformation $\varphi : \mathbb{R}^d \rightarrow \mathbb{R}^n$.
- If $\boldsymbol{x}$ is the original set of features $\varphi(\boldsymbol{x})$ denotes the new set of features

Example : Polynomial regression

- suppose there is just one feature x.
- define $\varphi : \mathbb{R} \rightarrow \mathbb{R}^2$ such that $\varphi_1(x) = x$ and $\varphi_2(x) = x^2$
- the linear predictor becomes
  $\boldsymbol{\theta}^T\varphi(x) = \theta_0 + \theta_1\varphi_1(x) + \theta_2\varphi_2(x) = \theta_0 + \theta_1 x + \theta_2 x^2$

More generally, a **polynomial basis** is the set of attributes that are powers of $\boldsymbol{x}$.

Basis expansion 3

- Data transformation, also known as basis expansion, is a general technique
- There are many possible choices of φ

Example 2 : Radial basis function

A **radial basis function** is a function of the form $\varphi(\boldsymbol{x}) = \Phi(\| \boldsymbol{x} - \boldsymbol{z} \|)$ where $\boldsymbol{z}$ is a constant

- e.g. $\varphi(\boldsymbol{x}) = \| \boldsymbol{x} - \boldsymbol{z} \|$ or $\varphi(\boldsymbol{x}) = \exp(-\frac{1}{\sigma} \| \boldsymbol{x} - \boldsymbol{z} \|^2)$

Basis expansion 4

- Basis expansion can significantly increase the utility of methods, especially, linear methods
- In the above examples, one limitation is that the transformation needs to be defined beforehand
- One idea is to *learn* the transformation φ from data (e.g., Artificial Neural Networks)
- Another powerful extension is the use of the *kernel trick* (e.g. SVM)

Regularization

Problem of underfitting/overfitting

There are two main options to address the issue of overfitting :

1. Reduce the number of features.
   - Manually select which features to keep.
   - Use a model selection algorithm.
2. Regularization
   - Keep all the features, but reduce the parameters.

Regularized logistic regression 1

The principle of regularization is to limit the overfitting by simultaneously controlling the model error on the learning set and the values of the model coefficients.

Intuition

Controlling these coefficients is a way to control the complexity of the model.

This control consists in constraining the coefficients to belong to a subset of $\mathbb{R}^{d+1}$ rather than being able to take any value in this space. This restricts the set of possible solutions.

Regularization works well when we have a lot of slightly useful features.

Regularized logistic regression 2

Cost function

$$J(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m} \text{loss}(h_\theta(\mathbf{x}^{(i)}), y^{(i)}) + \lambda \ \text{reg}(\boldsymbol{\theta})$$

- $\text{loss}(h_\theta(\mathbf{x}^{(i)}), y^{(i)}) = y^{(i)} \log(h_\theta(\mathbf{x}^{(i)})) + (1 - y^{(i)}) \log(1 - h_\theta(\mathbf{x}^{(i)}))$ for logistic regression.
- λ is the **regularization parameter**
  - $\text{reg}(\boldsymbol{\theta})$ is a constraint term on the model coefficients $\boldsymbol{\theta}$
  - λ is an **hyperparameter** of the logistic regression.
  - If λ is chosen to too large, it may smooth out the function too much and cause underfitting.

Regularized logistic regression 3

Ridge regularization (clustering)

$$\text{reg}_r(\boldsymbol{\theta}) = \| \boldsymbol{\theta} \|_2^2 = \sum_{j=1}^{d} \theta_j^2$$

- $\sum_{j=1}^{d} \theta_j^2$ excludes the bias term θ_0
- ridge regression uses the l_2 norm of $\boldsymbol{\theta}$ as a regularizer
- it has a clustering effect on correlated variables, as correlated variables will have similar coefficients
- it is a convex optimization problem (quadratic form) that always admits an explicit (analytical) single solution

Regularized logistic regression 4

Lasso regularization (sparsity)

$$\text{reg}_l(\boldsymbol{\theta}) = \| \boldsymbol{\theta} \|_1 = \sum_{j=1}^{d} | \theta_j |$$

- $\sum_{j=1}^{d} | \theta_j |$ excludes the bias term θ_0
- lasso regression uses the l_1 norm of $\boldsymbol{\theta}$ as a regularizer
- it acts as feature selection as it creates a sparse model : some coefficients will be null, leading the corresponding variables to be removed from the model
- it has nor analytical solution, neither always a unique solution, gradient descent should be used.

Regularized logistic regression 5

Elastic Net regularization

$$\text{reg}_{el}(\boldsymbol{\theta}) = ((1 - \alpha) \| \boldsymbol{\theta} \|_2^2 + \alpha \| \boldsymbol{\theta} \|_1)$$

- elastic net regression combines both the l_1 and l_2 norm of $\boldsymbol{\theta}$ in the regularizer
  - l_1 norm allows to obtain a more easily interpretable model
  - while l_2 norm avoids the overfitting
- it is parametrized by $\alpha \in [0, 1]$

Higher level view

Logistic regression is a specific type of Generalized Linear Models (GLM).

- with **binomial** conditional distribution of the response (Y)
- parameter $p = P(Y = 1|X = x)$
- linear predictor $\boldsymbol{\theta}^T X$
- the **logit** function is used to map the linear predictor $\boldsymbol{\theta}^T X$ to a probability p :

$$\text{logit(p)} = \log\left[\frac{p}{1-p}\right] = \boldsymbol{\theta}^T X \Leftrightarrow p = \frac{1}{1+e^{-\boldsymbol{\theta}^T X}}$$

Because logistic regression predicts *probabilities*, it can be fitted using likelihood.

Multinomial logistic regression

- generalizes logistic regression to multiclass problems
- generalization of the logistic sigmoid : *normalized exponential*, *softmax function*

$$P(Y = k|\boldsymbol{x}; \boldsymbol{\theta}) = \frac{\exp(\boldsymbol{\theta}_k^\top \boldsymbol{x})}{\sum_{c=1}^{C} \exp(\boldsymbol{\theta}_c^\top \boldsymbol{x})} .$$

- also known as : multiclass LR, softmax regression, multinomial logit, maximum entropy (MaxEnt) classifier, conditional maximum entropy model

Terminology

Loss function

$\mathcal{L}(y, h_{\boldsymbol{\theta}}(\boldsymbol{x}))$ computes the error for a single training example

- example : 0/1 loss, hinge loss, cross-entropy loss, exponential loss

Cost function

usually more general : average of the loss function over the entire training set (empirical risk)

$$C(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}(y^{(i)}, h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}))$$

Objective function

- Most general term for any function optimized during the training
- weighted sum of **cost function** and **regularization**

Maximum likelihood estimation

Logistic regression assumes a **Bernoulli distribution** defined as :

$$P(Y = 1) = p \text{ and } P(Y = 0) = 1 - p; \quad \text{with } p \in [0, 1]$$

equivalently :

$$P(Y = y) = p^y (1-p)^{(1-y)}; \quad \text{for } y \in \{0, 1\}$$

For the logistic regression model :

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

Assuming independence of examples in the training set, the likelihood is :

$$L(\theta) = P(y^{(1)}, y^{(2)}, \cdots y^{(m)} | \boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \cdots \boldsymbol{x}^{(m)}) = \prod_{i=1}^{m} P(y^{(i)}|\boldsymbol{x}^{(i)})$$

Maximum likelihood estimation

$$\begin{aligned} L(\theta) &= \prod_{i=1}^{m} P(y^{(i)}|\boldsymbol{x}^{(i)}) \\ &= \prod_{i=1}^{m} p^{y^{(i)}} (1-p)^{(1-y^{(i)})} \end{aligned}$$

"Log trick" : Instead of maximizing the likelihood, maximise its logarithm (log-likelihood)

$$\begin{aligned} \log L(\theta) = \log\left(\prod_{i=1}^{m} P(y^{(i)}|\boldsymbol{x}^{(i)})\right) &= \sum_{i=1}^{m} \log(P(y^{(i)}|\boldsymbol{x}^{(i)})) \\ &= \sum_{i=1}^{m} y^{(i)} \log(p) + (1 - y^{(i)}) \log(1-p) \end{aligned}$$

Cross-entropy

The **negative** log-likelihood for a **single data point** is the **cross-entropy** :

$$-\log(P(y|\boldsymbol{x})) = -y\log(p) - (1-y)\log(1-p) = H(y, p)$$

$$\boxed{\text{Maximise the log-likelihood} \Leftrightarrow \text{Minimise the binary cross-entropy}}$$

- same formula, but two different interpretations

Exercice - Calcul du gradient

Compute the gradient of the loss fonction (log-likelihood) for a given example $\boldsymbol{x} \in \mathbb{R}^n$:

$$L(\boldsymbol{\theta}) = L(\theta_0, \theta_1, \cdots, \theta_n) = y \log(p) + (1 - y) \log(1 - p)$$

with $p = h_\theta(\boldsymbol{x}) = \dfrac{1}{1 + \exp(-\boldsymbol{\theta}^T \boldsymbol{x})}$

and show that :

$$\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) = [y - h_\theta(\boldsymbol{x})]\boldsymbol{x}$$