Machine Learning

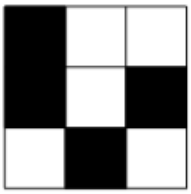
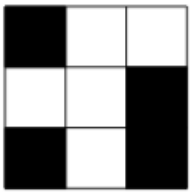
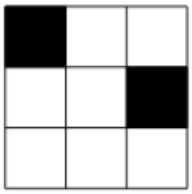
Master Degree in Computer Science - AIS Curriculum

Lesson 1

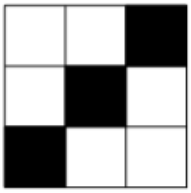
Marco Piangerelli
marco.piangerelli@unicam.it

5-6 Ottobre 2020

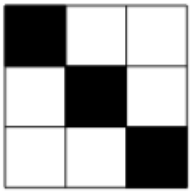
What are we learning?

$f = -1$



$f = +1$



$f = ?$

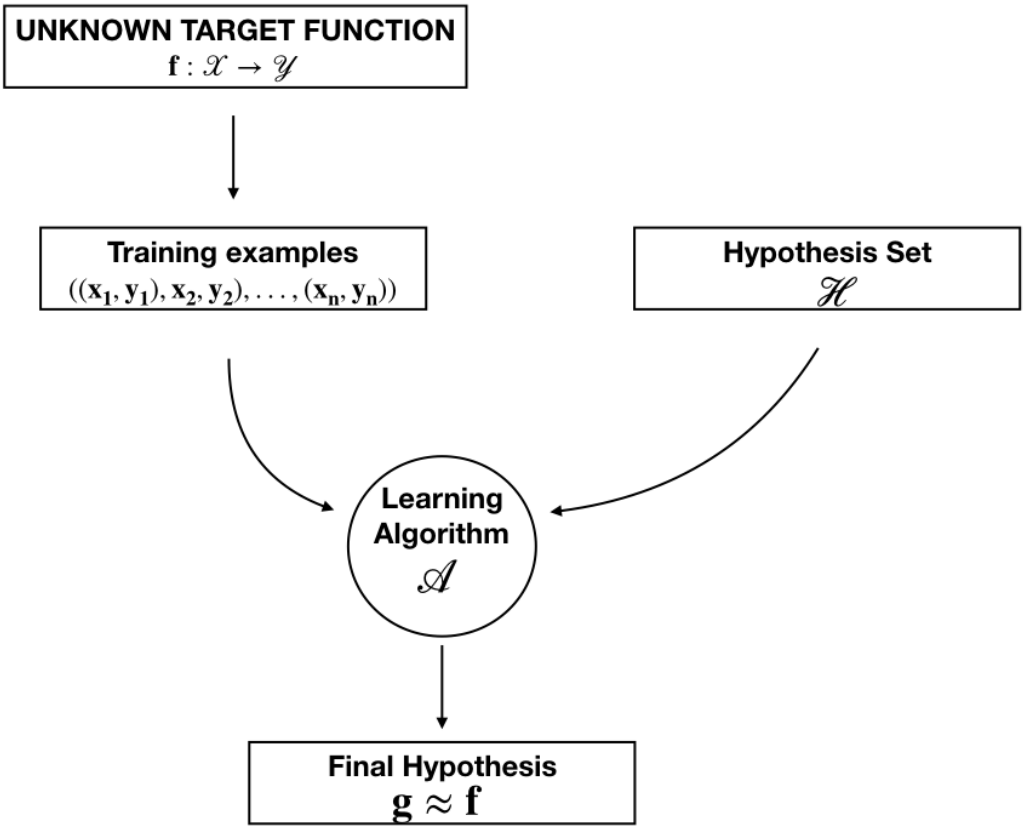
Learning VS Machine Learning

Definition [Mitchell (1997)]

" A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E"

Notation

$\mathbf{x}$ the input $\mathbf{x} \in \mathcal{X}$. Often a column vector $\mathbf{x} \in \mathbb{R}^d$ or $\mathbf{x} \in \{1\} \times \mathbb{R}^d$. x is used if input is scalar. $\mathbf{y}$ the output $\mathbf{y} \in \mathcal{Y}$.
$\mathcal{X}$ input space whose elements are $\mathbf{x} \in \mathcal{X}$, $\mathcal{Y}$ output space whose elements are $\mathbf{y} \in \mathcal{Y}$
Data, $\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2) ... (\mathbf{x}_n, y_n)\}$
Unknown function to be learned $f : \mathcal{X} \rightarrow \mathcal{Y}$
Approximation of the **Unknown** function $g : \mathcal{X} \rightarrow \mathcal{Y}$
$\mathcal{A}$ learning algorithm, $\mathcal{H}$ set of candidates formulas for g


UNKNOWN TARGET FUNCTION
$\mathbf{f} : \mathcal{X} \rightarrow \mathcal{Y}$
Training examples
$((\mathbf{x}_1, \mathbf{y}_1), \mathbf{x}_2, \mathbf{y}_2), \ldots, (\mathbf{x}_n, \mathbf{y}_n))$
Hypothesis Set
$\mathscr{H}$
Learning Algorithm
$\mathscr{A}$
Final Hypothesis
$\mathbf{g} \approx \mathbf{f}$

Classification

The Task T

Compute $f : \mathbb{R}^n \to 1, ..., k$ which maps data $x \in \mathbb{R}^n$ to a category in $1, ..., k$. Alternative: Compute $f : \mathbb{R}^n \to \mathbb{R}^k$ which maps data $x \in \mathbb{R}^n$ to a histogram with respect to k categories.

Regression

Task T

Predict a numerical value $f : \mathbb{R}^n \rightarrow \mathbb{R}$.

1. energy consumption forecasting
2. market trade modeling
3. ...

Density Estimation

The Task T

Estimate a probability density $p : \mathbb{R} \to \mathbb{R}_+$ which can be interpreted as a probability distribution on the space that the examples were drawn from.

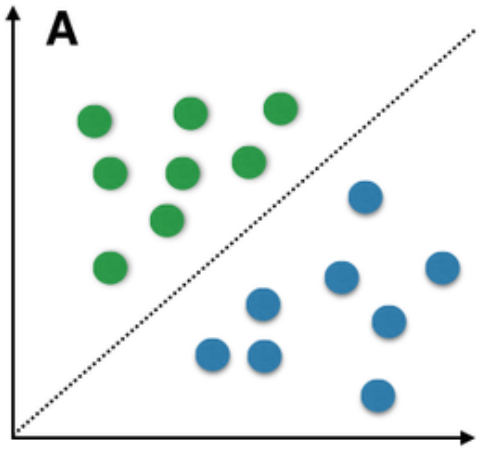
1. Useful for many tasks in data processing, for example if we observe corrupted data $\tilde{x}$ we may estimate the original x as the *argmax* $p(\tilde{x}|x)$.

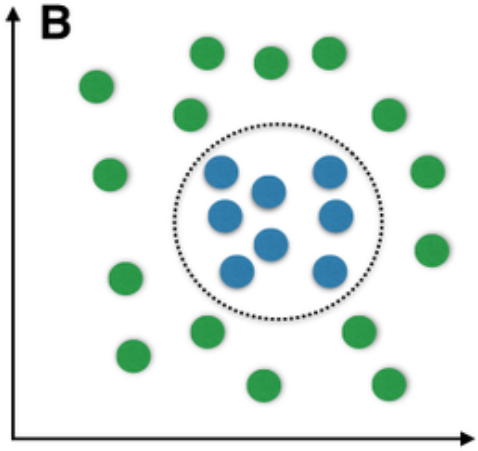
Supervised Learning

The experience E

The experience typically consists of a dataset which consists of many examples (aka data points).

If these data points are labeled (for example in the classification problem, if we know the classifier of our given data points) we speak of supervised learning.



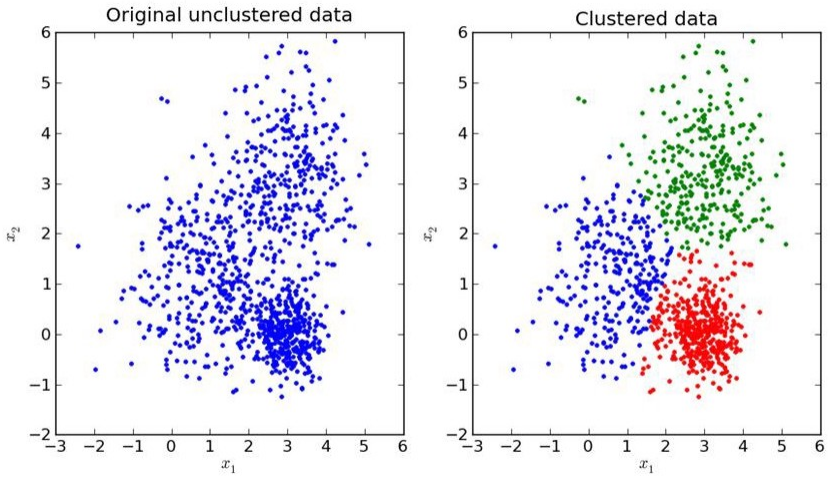


Unsupervised Learning

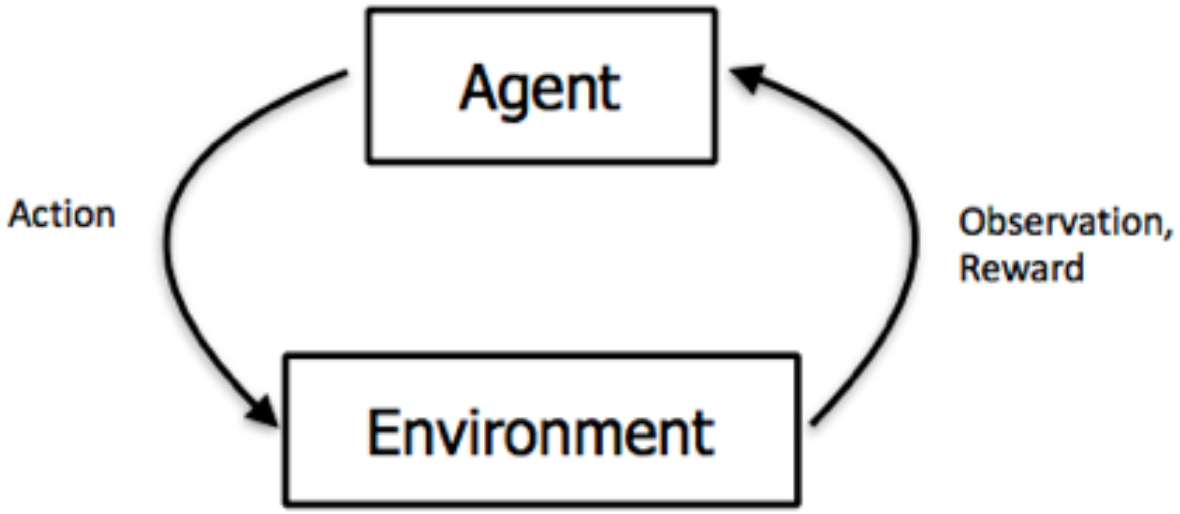
The experience E

The experience typically consists of a dataset which consists of many examples (aka data points).

If these data points are not labeled (for example in the classification problem, the algorithm would have to find the clusters itself from the given dataset) we speak of unsupervised learning.



Reinforcement Learning



The Performance Measure P

In classification problems this is typically the accuracy, i.e., the proportion of examples for which the model produces the correct output.

In regression problems this is typically the cost function or energy function, i.e., the error with respect to ground truth ,i.e. the real value

Often the given dataset is split into a training set on which the algorithm operates and a test set on which its performance is measured.

Examples...

The Task

Binary Classification: Predict $f : \mathbb{R}^n \to \{-1-, 1\}$.

The Experience

Training data $(x_i^{train}, y_i^{train})_{i=1}^m$

The Performance Measure

Given test data $(x_i^{test}, y_i^{test})_{i=1}^n$ we evaluate the performance of an estimator $\hat{f} : \mathbb{R}^n \to \{-1, 1\}$ as the accuracy

$$Accuracy = \frac{\#k|k = True}{m}$$

A "simple" model

$\mathcal{X}$ is the set of data, $\mathbf{x}$, namely the information about the clients that requested a bank loan
$\mathcal{Y}$ is the binary set $\{-1, 1\}$ (yes or no)

A "simple" model

$\mathcal{X}$ is the set of data, **x**, namely the information about the clients that requested a bank loan
$\mathcal{Y}$ is the binary set $\{-1, 1\}$ (yes or no)
A simple model could be a "thresholded" model:

- $\sum_{i=1}^{k} w_i x_i > threshold \rightarrow +1 \rightarrow$ YES
- $\sum_{i=1}^{k} w_i x_i < threshold \rightarrow$ -1 $\rightarrow$ NO

A "simple" model

$\mathcal{X}$ is the set of data, $\mathbf{x}$, namely the information about the clients that requested a bank loan
$\mathcal{Y}$ is the binary set $\{-1, 1\}$ (yes or no)
A simple model could be a "thresholded" model:

$$\mathcal{H} = g(\mathbf{w}^T \phi(\mathbf{x}))$$
$$\mathcal{H} = g(\mathbf{w}^T \mathbf{x}) \text{ (original model)}$$

$$g(a) = \begin{cases} +1 & a \geq 0 \\ -1 & a < 0 \end{cases}$$

A "simple" model

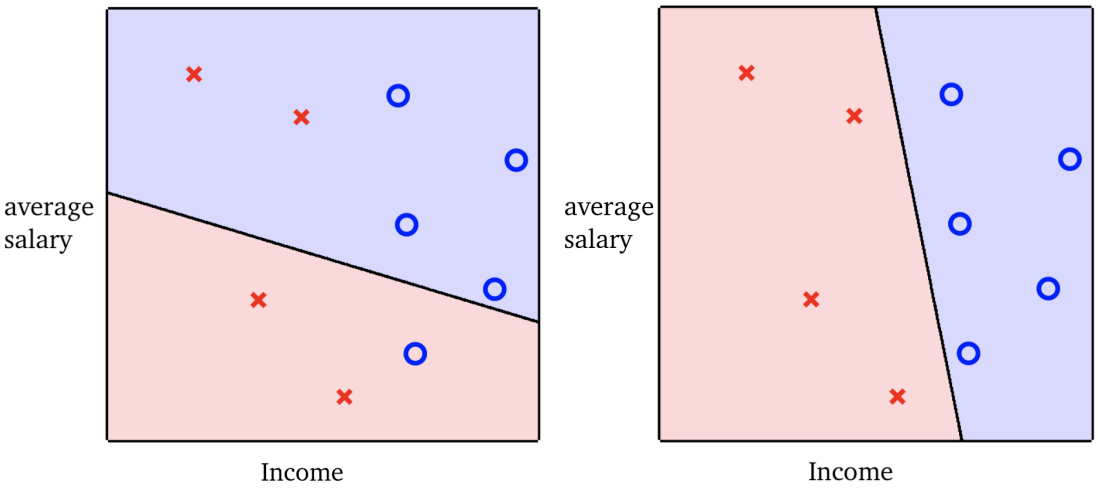
$\mathcal{X}$ is the set of data, $\mathbf{x}$, namely the information about the clients that requested a bank loan
$\mathcal{Y}$ is the binary set $\{-1, 1\}$ (yes or no)
A simple model could be a "thresholded" model:

$$\mathcal{H} = g(\mathbf{w}^T \phi(\mathbf{x}))$$
$$\mathcal{H} = g(\mathbf{w}^T \mathbf{x}) \text{ (original model)}$$

$$g(a) = \begin{cases} +1 & a \geq 0 \\ -1 & a < 0 \end{cases}$$

$$g(\mathbf{w}^T \phi(\mathbf{x})) = y(\mathbf{x}) = \{-1, +1\}$$

The Perceptron (Rosenblatt 1958)



Perceptron Learning Algorithm (PLA)

How does PLA work? $\rightarrow$ it is an iterative procedure

- $\mathbf{w}(1) = 0$

Perceptron Learning Algorithm (PLA)

How does PLA work? $\rightarrow$ it is an iterative procedure

- $\mathbf{w}(1) = 0$
- for each iteration $t = 0, 1, 2, ...$

Perceptron Learning Algorithm (PLA)

How does PLA work? $\rightarrow$ it is an iterative procedure

- $\mathbf{w}(1) = 0$
- for each iteration $t = 0, 1, 2, ...$
- the weight vector is $\mathbf{w}(t)$

Perceptron Learning Algorithm (PLA)

How does PLA work? $\rightarrow$ it is an iterative procedure

- $\mathbf{w}(1) = 0$
- for each iteration $t = 0, 1, 2, ...$
- the weight vector is $\mathbf{w}(t)$
- from $(\mathbf{x}_1, y_1)$,...,$(\mathbf{x}_N, y_N)$ pick one misclassified example.

Perceptron Learning Algorithm (PLA)

How does PLA work? $\rightarrow$ it is an iterative procedure

- $\mathbf{w}(1) = 0$
- for each iteration $t = 0, 1, 2, ...$
- the weight vector is $\mathbf{w}(t)$
- from $(\mathbf{x}_1, y_1)$,...,$(\mathbf{x}_N, y_N)$ pick one misclassified example.
- Call this example $(\mathbf{x}_\mu, y_\mu)$
  $g(\mathbf{w}^T(\mathbf{x}_\mu)) \neq y_\mu$

Perceptron Learning Algorithm (PLA)

How does PLA work? $\rightarrow$ it is an iterative procedure

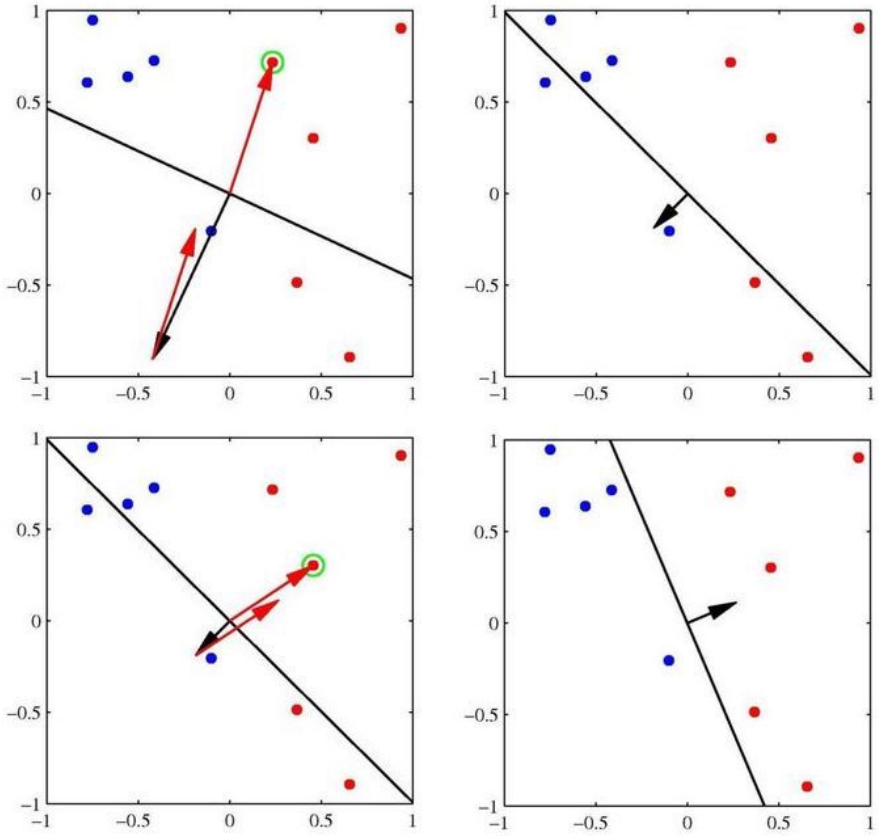
- $\mathbf{w}(1) = 0$
- for each iteration $t = 0, 1, 2, ...$
- the weight vector is $\mathbf{w}(t)$
- from $(\mathbf{x}_1, y_1)$,...,$(\mathbf{x}_N, y_N)$ pick one misclassified example.
- Call this example $(\mathbf{x}_\mu, y_\mu)$
  $g(\mathbf{w}^T(\mathbf{x}_\mu)) \neq y_\mu$
- update the weight vector is $\mathbf{w}(t+1) = \mathbf{w}(t) + (\mathbf{x}_\mu * y_\mu)$

Perceptron Learning Algorithm (PLA)

How does PLA work? $\rightarrow$ it is an iterative procedure

- $\mathbf{w}(1) = 0$
- for each iteration $t = 0, 1, 2, ...$
- the weight vector is $\mathbf{w}(t)$
- from $(\mathbf{x}_1, y_1)$,...,$(\mathbf{x}_N, y_N)$ pick one misclassified example.
- Call this example $(\mathbf{x}_\mu, y_\mu)$
  $g(\mathbf{w}^T(\mathbf{x}_\mu)) \neq y_\mu$
- update the weight vector is $\mathbf{w}(t+1) = \mathbf{w}(t) + (\mathbf{x}_\mu * y_\mu)$
- $t \leftarrow t + 1$

Perceptron Learning Algorithm (PLA)



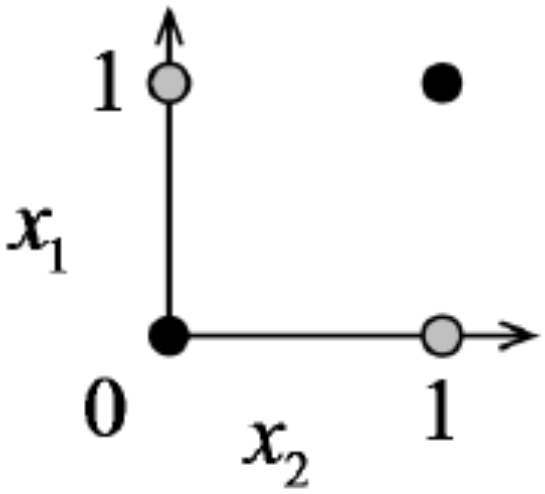
Figures from: Bishop, Christopher M. "Pattern recognition and machine learning." springer, 2006.

Perceptron Learning Algorithm (PLA)

The REAL question is: "It does really work"?

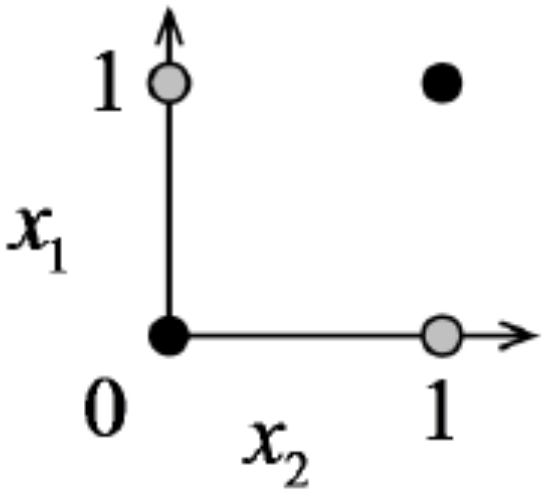
Perceptron Learning Algorithm (PLA)

The REAL question is: "It does really work"?



Perceptron Learning Algorithm (PLA)

The REAL question is: "It does really work"?



Theorem

If the data can be fit by a linear separator, then after some finite number of steps, PLA will find one

Linear Regression

The Task

Regression: Predict $f : \mathbb{R}^n \to \mathbb{R}$.

The Experience

Training data $(x_i^{train}, y_i^{train})_{i=1}^m$

The Performance Measure

Given test data $(x_i^{test}, y_i^{test})_{i=1}^n$ we evaluate the performance of an estimator $\hat{f} : \mathbb{R}^n \to \mathbb{R}$ as the mean squared error

$$\frac{1}{n}\sum_{i=1}^{n}(\hat{f}(x_i^{test}) - y_i^{test})^2$$

Let's see an example....

- General formulation of the regression problem (Least Squares Estimation)
- Normal equation for solving the LSE
- Probabilistic view of Linear Regression

Logistic Regression

The Task

Logistic Regression: Binary Classification $f : \mathbb{R}^n \to \{0, 1\}$.

The Experience

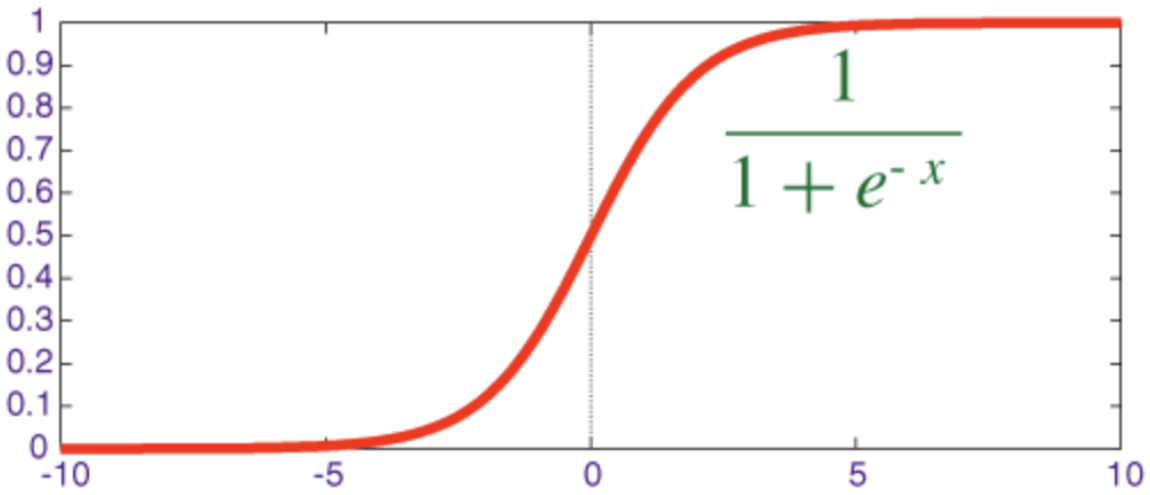
Training data $(x_i^{train}, y_i^{train})_{i=1}^m$

The Performance Measure

Given some data we have to infer the probability of an event to occur

Logistic Regression

Logistic function



Logistic Regression

The likelihood:

$$\mathcal{L}(\theta) = \sum_{i=1}^{m} y^{(i)} log(h_\theta(x^{(i)})) - (1 - y^{(i)}) log(1 - h_\theta(x^{(i)}))$$

Gradient Descent with fixed step

Theorem (Rate of convergence)

Suppose the function $f : \mathbb{R}^n \to \mathbb{R}$ is convex and differentiable and that its gradient is Lipschitz continuous with constant $L \geq 0$, i.e. $\|\nabla f(y) - \nabla f(x)\|_2 \leq L \|\boldsymbol{y} - \boldsymbol{x}\|$, for any $\boldsymbol{y}, \boldsymbol{x}$. Then if we run the gradient descent algorithm for k iterations with a fixed time step size $t \leq 1/L$ it will yield a solution $f^{(k)}$ which satisfies:

$$f(\boldsymbol{x}^{(k)}) - f(\boldsymbol{x}^*) = \frac{\left\|\boldsymbol{x}^{(0)} - \boldsymbol{x}^*\right\|}{2tk} \tag{1}$$

where $f(\boldsymbol{x}^*)$ is the optimal value. Intuitively, this means that gradient descent is guaranteed to converge and that it converges with rate $\mathcal{O}(1/k)$

Proof

Since ∇f is Lipschitz continuous with constant $L \geq 0$, $\nabla^2 f(x) \preceq LI$ or equivanlently, , $\nabla^2 f(x) - LI$ is a semidefinite positive matrix ...