Machine Learning

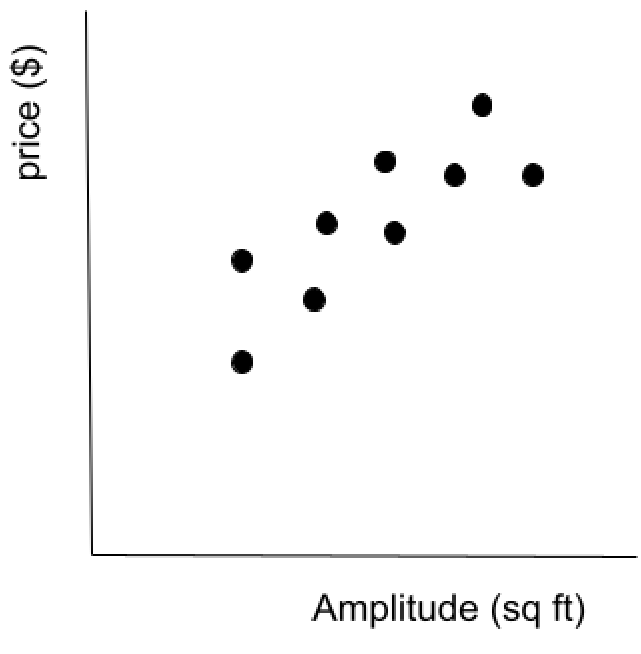
"Field of study that gives computers the ability to learn without being explicitly programmed" - Arthur Samuel

1. Supervised learning

Let's start from an example with a dataset

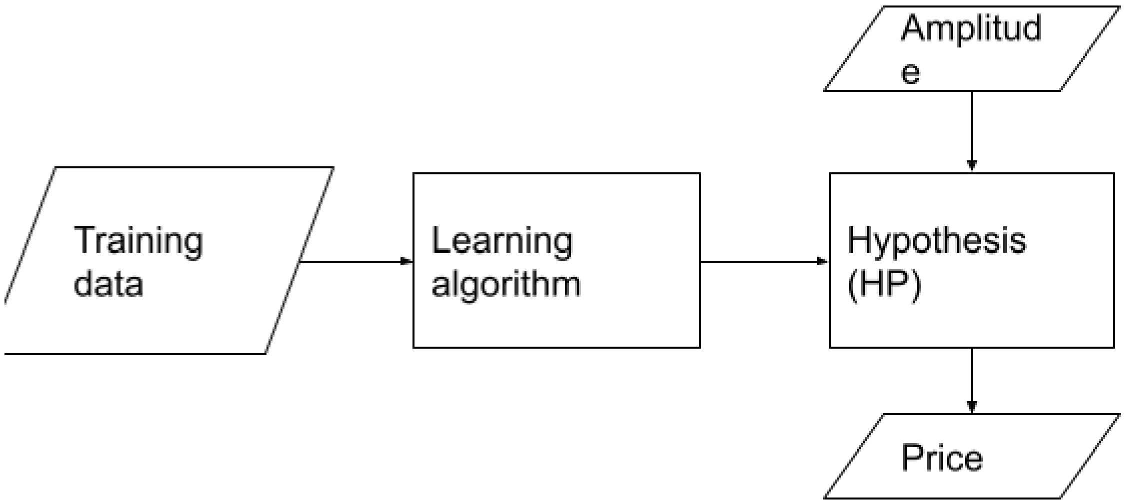
Living Area (sq ft)	Price ($)
2104	400
1600	330
2400	232

We can represent the data as follows



The common approach to this kind of problems is finding a *function* that operates with the data

In machine learning, the mapping is learnt automatically, and this leads to a different approach that is *data dependent*

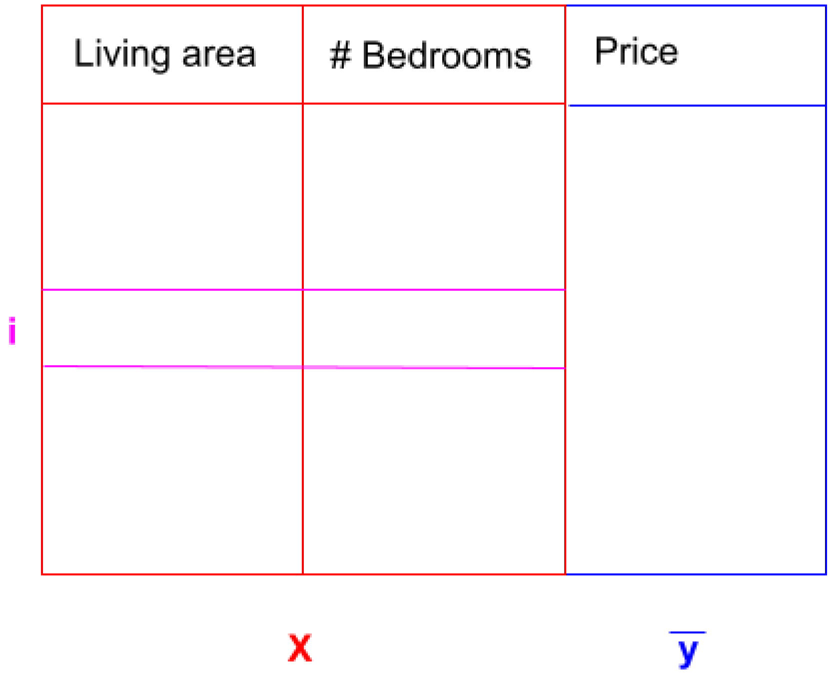


If the output data is real valued, we are considering a *regression*, if it's binary we call it *classification*. This distinction is implicated in the trained data.

<u>Example</u>

- Linear regression: regression
- Logistic regression: classification

2. Linear regression



N: number of examples (rows)
D: numbers of features (records)
$\vec{x}^i$: i[th] input $\in \mathbb{R}^d$
y^i: i[th] input $\in \mathbb{R}$

Let's define a function H related to the *ordinary least squares assumption*

$$H_\theta(\vec{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

θ: parameters, $\theta \in \mathbb{R}$
$\theta_1, \ldots, \theta_d$: weights
θ_0: bias

In general

$$H_\theta(x) = \sum_{i=0}^{d} \theta_i x_i$$

and the input of this function is one row at a time.

Our mission is to determine θ with the learning algorithm. To do this, we use a *cost function* J that associates greater thetas to a minor cost and vice versa.

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{n}\left[H_\theta\left(x^i\right) - y^i\right]^2 \in \mathbb{R}$$

What is happening here is that we are *comparing* our cost function with the right answer from y vector.

Now what we want is to *optimize* θ and, in linear regressions, there are two main algorithms to do that:

- LMS algorithm
- Normal equation

I. LMS algorithm

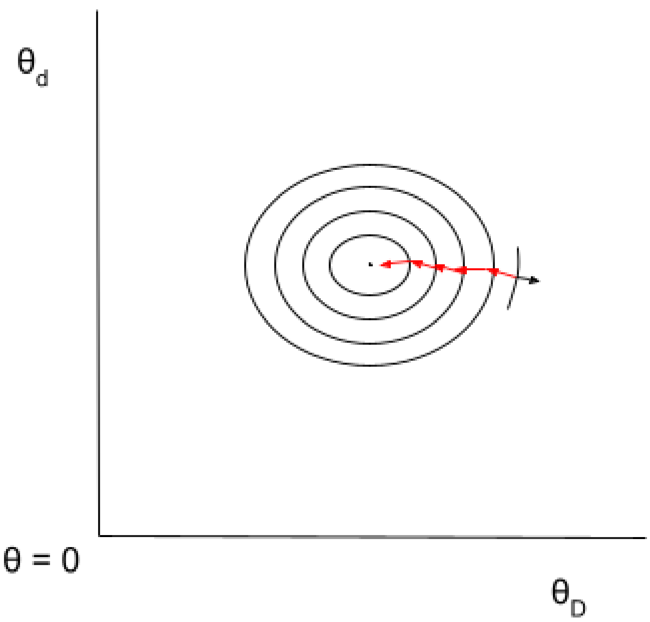
It is an iterative algorithm. Let's set an initial value for θ (any).

$$\theta := 0$$

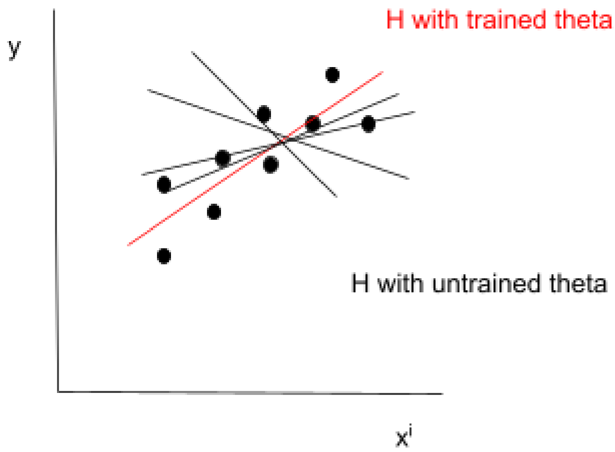
We need to repeat the following operation until *convergence* (not necessarily happening), for every element of theta (vector) and obtain a *new corresponding* θ.

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

With θ_j being the j^th of the *gradient*



The sign is flipped, α is the *learning grade* and scales the vector (a fixed value in this case), step by step it will eventually converge. The circles (ellipsis) represent the set of points of theta having the same cost.
Now let's take the new θ and use it in our function H.



II. The normal equation

Uses the dataset from the beginning, instead of iterating gradually.

$X = \begin{bmatrix} & \end{bmatrix}$: (d+1 x n)-matrix

$\vec{y} = \begin{bmatrix} \end{bmatrix}$: n-vector

$\theta = \begin{bmatrix} \end{bmatrix}$: d+1-vector

$$X\theta - \vec{y} = \begin{bmatrix} \theta^{x^i} - y^i \\ \dots \end{bmatrix}$$

Starting from the definition of J, we can work our way out to another model

$$J(\theta) = \frac{1}{2} \sum_{i=1}^{n} \left[H_\theta\left(x^i\right) - y^i\right]^2 \in \mathbb{R}$$

$$J(\theta) = \frac{1}{2}(X\theta - \vec{y})^T(X\theta - \vec{y})$$

$$\nabla_\theta J(\theta) = 0$$

$$\nabla_\theta \frac{1}{2}(X\theta - \vec{y})^T (X\theta - \vec{y}) = 0$$

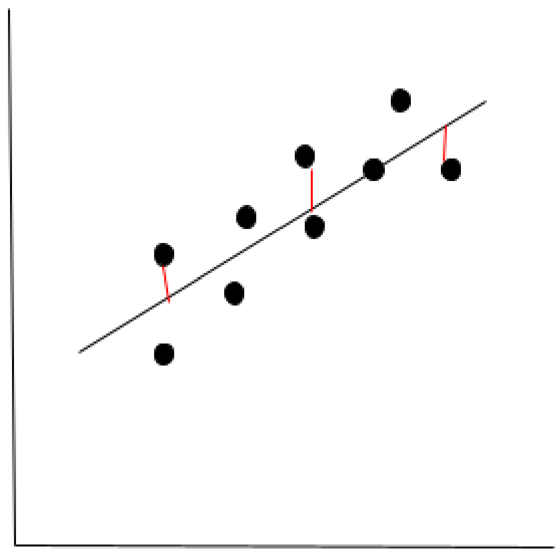
$$\nabla_\theta \frac{1}{2}\left[2X^T X\theta - 2X^T \vec{y}\right] = 0$$

$$X^T X\theta - X^T \vec{y} = 0$$

We come to the final conclusion, which is the solution for θ for a few “lucky” ML models.

$$\theta = \left(X^T X\right)^{-1} X^T \vec{y}$$

Now, let’s take a statistical approach, and analyze the *noise* related to this kind of regression.



We can define y^i as follows

$$y^i = \theta^T\left(x^i\right) + \varepsilon^i$$

Where $\theta^T\left(x^i\right)$ is the *HP function* and ε^i is the *noise randomness* and $\varepsilon^i \sim N\left(0, \sigma^i\right)$.

Let’s better define epsilon

$$\varepsilon^i = \vec{y}^i - \varepsilon^i\left(x^i\right)$$

And the associated *gaussian density distribution function* P

$$P\left(\varepsilon^i\right) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left\{-\frac{1}{2}\frac{\left(\varepsilon^i - d^2\right)}{\sigma^2}\right\}$$

$$P\left(\vec{y}^i||x^i i\theta\right) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left\{\frac{-\left[y^i - \theta^T\left(x^i\right)\right]^2}{2\sigma^2}\right\}$$

$$P(y||xi\theta) = \prod_{i=1}^{n} P\left(y^i||x^i i\theta\right)$$

Let’s define L function of likelihood, in order to *maximize* the result

$$L(\theta) = \prod_{i=1}^{n} P\left(y^i||x^i i\theta\right)$$

$$\log L(\theta) = \log\prod_{i=1}^{n} P\left(y^i||x^i i\theta\right) = \sum_{i=1}^{n}\log P\left(y^i||x^i i\theta\right)$$

$$\sum_{i=1}^{n}\log\left[\frac{1}{\sqrt{2\pi}\sigma}\exp\left\{\frac{-\left[y^i - \theta^T\left(x^i\right)\right]^2}{2\sigma^2}\right\}\right]$$

$$\sum_{i=1}^{n}\left(\log\frac{1}{\sqrt{2\pi}\sigma}\right) + \left\{\frac{-\left(y^i - \theta\left(x^i\right)\right)^2)}{2\sigma^2}\right\}$$

$$\sum_{i=1}^{n}\left\{K + \frac{-\left(y^i - \theta\left(x^i\right)\right)^2)}{2\sigma^2}\right\}$$

Coming to the conclusion

$$\log L(\theta) = nK - \frac{1}{2\sigma^2}\sum_{i=1}^{n}\left(y^i - \theta^T\left(x^i\right)\right)^2$$