Linear Regression

Dr. Víctor Uc Cetina

Universität Hamburg

Content

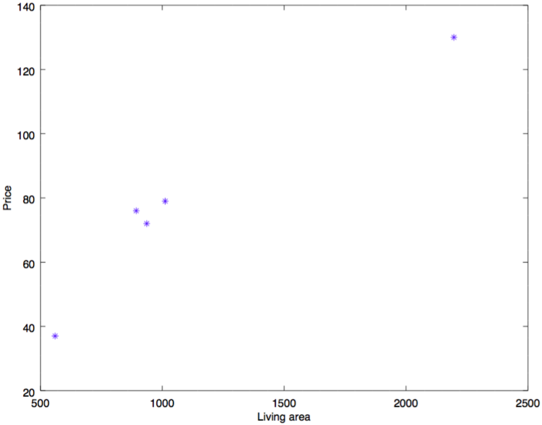
1. Problem 1 - Housing Data
2. Least Mean Square
3. The Normal Equations
4. A Probabilistic Interpretation
5. Locally Weighted Linear Regression

Housing Data

Suppose we have the following housing data:

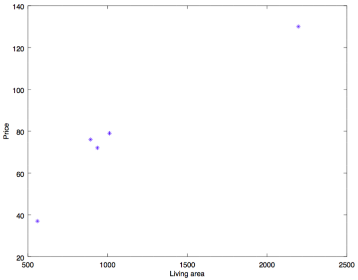
Living area (feet square)	Price (USD)
560	37
1012	79
893	76
2196	130
⋮	⋮
936	72

Housing Data



One Dimensional Regression Problem

Living area (x_1)	Price (y)
560	37
1012	79
893	76
2196	130
⋮	⋮
936	72



We are looking for something like: $h_\theta(\mathbf{x}) = \theta_0 + \theta_1 x_1$

Two Dimensional Regression Problem

Living area (x_1)	Bedrooms (x_2)	Price (y)
560	2	37
1012	3	79
893	3	76
2196	4	130
⋮	⋮	⋮
936	3	72

Now, we are looking for something like: $h_\theta(\mathbf{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$

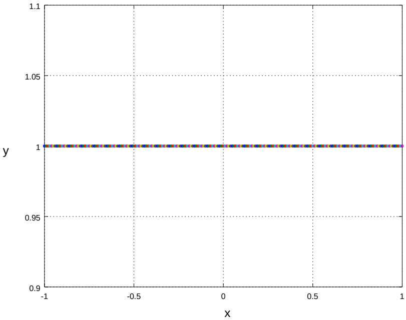
Letting $x_0 = 1$ we have: $h(\mathbf{x}) = \sum_{j=0}^{n} \theta_j x_j$

This is the dot product: $\theta^\top \mathbf{x}$

Polynomial Functions

$$y = 1$$

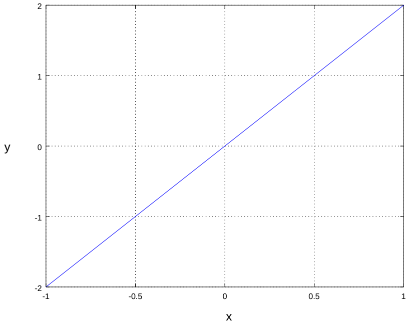
$$y = \theta_0$$



Polynomial Functions

$$y = 2x$$

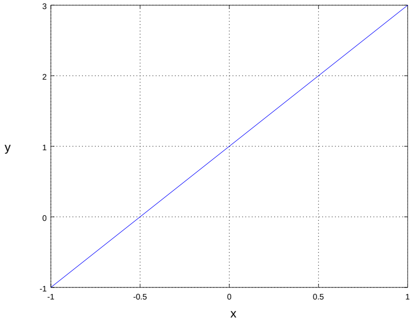
$$y = \theta_1 x$$



Polynomial Functions

$$y = 1 + 2x$$

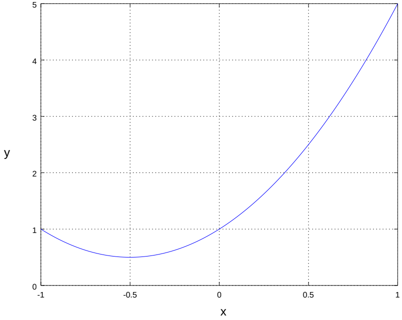
$$y = \theta_0 + \theta_1 x$$



Polynomial Functions

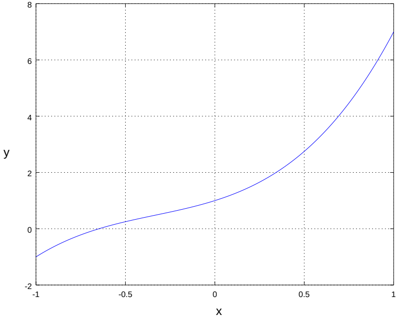
$$y = 1 + 2x + 2x^2$$

$$y = \theta_0 + \theta_1 x + \theta_2 x^2$$



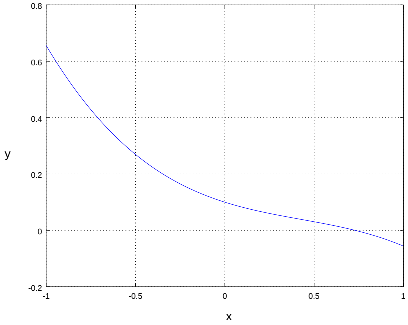
Polynomial Functions

$$y = 1 + 2x + 2x^2 + 2x^3$$
$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$$



Polynomial Functions

$$y = 0.1 - 0.2x + 0.2x^2 - 0.156x^3$$
$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$$

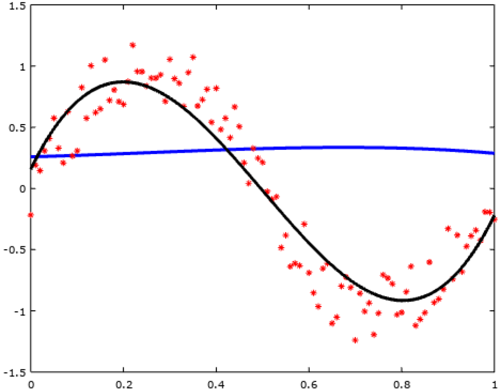


How do we pick θ?

- One reasonable method is to pick θ such that $h(x)$ is close to y, at least for our m training examples.
- We define the cost function $J(\theta) = \frac{1}{2}\sum_{i=1}^{m}\left[h_\theta(x_i) - y_i\right]^2$.
- We can initialize randomly θ and use the gradient descent algorithm to find the θ that minimizes $J(\theta)$.
- $\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta)$.

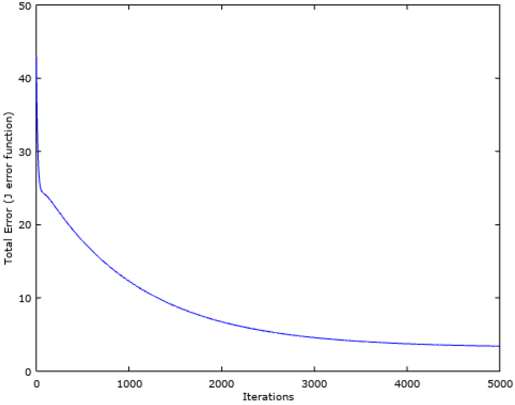
Estimating parameters

In blue, the initial $h_\theta(x)$ function, with randomly generated θ's. In black, the final $h_\theta(x)$ function.

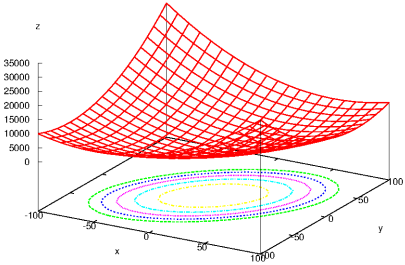


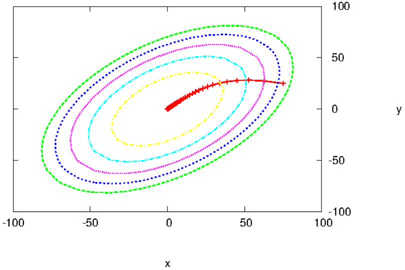
Graph of the error

Plot of the error $J(\theta) = \frac{1}{2}\sum_{i=1}^{m}\left[h_\theta(x_i) - y_i\right]^2$, after each iteration of stochastic gradient descent.



Gradient Descent





Deriving the LMS Learning Rule

$$
\begin{aligned}
\frac{\partial}{\partial \theta_j} J(\theta) &= \frac{\partial}{\partial \theta_j} \frac{1}{2}(h_\theta(x) - y)^2 \\
&= 2 \cdot \frac{1}{2}(h_\theta(x) - y) \cdot \frac{\partial}{\partial \theta_j}(h_\theta(x) - y) \\
&= (h_\theta(x) - y) \cdot \frac{\partial}{\partial \theta_j}\Big(\sum_{k=0}^{K} \theta_k x_k - y \Big) \\
&= (h_\theta(x) - y) x_j
\end{aligned}
$$

For a single example i, the rule is:

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

$$\theta_j := \theta_j + \alpha \big[y_i - h_\theta(x_i) \big] (x_i)_j$$

Applying the update rule

The rule is:

$$\theta_j := \theta_j + \alpha\big[y_i - h_\theta(x_i)\big](x_i)_j$$

Consider that your third example is $x_3 = 2$, $y_3 = 20$ and your learning rate is $\alpha = 0.1$.

Consider also that you are using a polynomial of degree 3:
$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$.

The update rule is applied as follows:

$$\theta_0 := \theta_0 + 0.1\big[20 - h_\theta(2)\big]2^0.$$
$$\theta_1 := \theta_1 + 0.1\big[20 - h_\theta(2)\big]2^1.$$
$$\theta_2 := \theta_2 + 0.1\big[20 - h_\theta(2)\big]2^2.$$
$$\theta_3 := \theta_3 + 0.1\big[20 - h_\theta(2)\big]2^3.$$

LMS Algorithms

Batch Gradient Descent

Repeat until convergence {

$$\theta_j := \theta_j + \alpha \sum_{i=1}^{m} \left[y_i - h_\theta(x_i) \right] (x_i)_j \qquad \text{(for every } j\text{).}$$

}

Stochastic Gradient Descent

Loop {

for $i = 1$ to m {

$$\theta_j := \theta_j + \alpha \left[y_i - h_\theta(x_i) \right] (x_i)_j \qquad \text{(for every } j\text{).}$$

}

}

LMS Algorithms

Mini-Batch Gradient Descent

Repeat until convergence {

$$\theta_j := \theta_j + \alpha \sum_{i=1}^{k} \left[y_i - h_\theta(x_i) \right] (x_i)_j \qquad \text{(for every } j\text{)}.$$

}

Here we use mini-batches containing 10 to 1000 examples. This is $k \in [10, 1000]$.

Matrix of Training Examples

Given a training set of m examples, with each example consisting of n variables, then we can construct a $m \times (n+1)$ matrix:

$$\mathbf{X} = \begin{bmatrix} x_{1,0} & x_{1,1} & \cdots & x_{1,n} \\ x_{2,0} & x_{2,1} & \cdots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m,0} & x_{m,1} & \cdots & x_{m,n} \end{bmatrix} = \begin{bmatrix} [\mathbf{x}_1]^\top \\ [\mathbf{x}_2]^\top \\ \vdots \\ [\mathbf{x}_m]^\top \end{bmatrix}$$

Vector of Training Target Values

Let $\mathbf{y}$ be the m-dimensional vector containing the target values from the training set:

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

Cost Function $J(\theta)$

We can write the $J(\theta)$ cost function as follows:

$$
\begin{aligned}
J(\theta) &= \tfrac{1}{2}\textstyle\sum_{i=1}^{m}\left[h_\theta(x_i) - y_i\right]^2 \\
& \tfrac{1}{2}(\mathbf{X}\theta - \mathbf{y})^\top(\mathbf{X}\theta - \mathbf{y})
\end{aligned}
$$

and the $\nabla_\theta J(\theta)$ can be written as:

$$
\begin{aligned}
\nabla_\theta J(\theta) &= \nabla_\theta \tfrac{1}{2}(\mathbf{X}\theta - \mathbf{y})^\top(\mathbf{X}\theta - \mathbf{y}) \\
\nabla_\theta J(\theta) &= \mathbf{X}^\top\mathbf{X}\theta - \mathbf{X}^\top\mathbf{y} \\
0 &= \mathbf{X}^\top\mathbf{X}\theta - \mathbf{X}^\top\mathbf{y} \\
\mathbf{X}^\top\mathbf{X}\theta &= \mathbf{X}^\top\mathbf{y} \\
\theta &= (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}.
\end{aligned}
$$

Computing Directly θ

For an n by n square matrix A, the trace of A is defined to be the sum of its diagonal entries

$$\operatorname{tr} A = \sum_{i=1}^{n} A_{ii}$$

If a is a real number, then

$$\operatorname{tr} a = a$$

Computing Directly θ

For matrices A, B, C and D, we have that

$$\text{tr } AB = \text{tr } BA$$

$$\text{tr } ABC = \text{tr } CAB = \text{tr } BCA$$

$$\text{tr } ABCD = \text{tr } DABC = \text{tr } CDAB = \text{tr } BCDA$$

Computing Directly θ

For matrices A and B, and real number a, we have that

$$\mathrm{tr}\ A = \mathrm{tr}\ A^\top$$

$$\mathrm{tr}\ A + B = \mathrm{tr}\ A + \mathrm{tr}\ B$$

$$\mathrm{tr}\ aA = a\ \mathrm{tr}\ A$$

$$\nabla_A\ \mathrm{tr}\ AB = B^\top$$

$$\nabla_{A^\top} f(A) = (\nabla_A f(A))^\top$$

$$\nabla_{A^\top}\ \mathrm{tr}\ ABA^\top C = B^\top A^\top C^\top + BA^\top C$$

Computing Directly θ

$$
\begin{aligned}
\nabla_\theta J(\theta) &= \nabla_\theta \tfrac{1}{2}(\mathbf{X}\theta - \mathbf{y})^\top(\mathbf{X}\theta - \mathbf{y}). \\
&= \nabla_\theta \tfrac{1}{2}(\theta^\top \mathbf{X}^\top - \mathbf{y}^\top)(\mathbf{X}\theta - \mathbf{y}). \\
&= \tfrac{1}{2}\nabla_\theta(\theta^\top \mathbf{X}^\top \mathbf{X}\theta - \theta^\top \mathbf{X}^\top \mathbf{y} - \mathbf{y}^\top \mathbf{X}\theta + \mathbf{y}^\top \mathbf{y}). \\
&= \tfrac{1}{2}\nabla_\theta \operatorname{tr}(\theta^\top \mathbf{X}^\top \mathbf{X}\theta - \theta^\top \mathbf{X}^\top \mathbf{y} - \mathbf{y}^\top \mathbf{X}\theta + \mathbf{y}^\top \mathbf{y}). \\
&= \tfrac{1}{2}\nabla_\theta \operatorname{tr}\theta^\top \mathbf{X}^\top \mathbf{X}\theta - \operatorname{tr}\theta^\top \mathbf{X}^\top \mathbf{y} - \operatorname{tr}\mathbf{y}^\top \mathbf{X}\theta + \operatorname{tr}\mathbf{y}^\top \mathbf{y}.
\end{aligned}
$$

Using $\operatorname{tr} A = \operatorname{tr} A^\top$ and $(ABC)^\top = C^\top B^\top A^\top$,
we have $\operatorname{tr}\theta^\top \mathbf{X}^\top \mathbf{y} = \operatorname{tr}(\theta^\top \mathbf{X}^\top \mathbf{y})^\top = \operatorname{tr}\mathbf{y}^\top \mathbf{X}\theta$.

$$
= \tfrac{1}{2}\nabla_\theta \operatorname{tr}\theta^\top \mathbf{X}^\top \mathbf{X}\theta - 2\operatorname{tr}\mathbf{y}^\top \mathbf{X}\theta.
$$

Computing Directly θ

$$
\begin{aligned}
\nabla_\theta J(\theta) &= \tfrac{1}{2}\nabla_\theta \,\mathrm{tr}\, \theta^\top \mathbf{X}^\top \mathbf{X}\theta - 2\,\mathrm{tr}\,\mathbf{y}^\top \mathbf{X}\theta. \\
& \quad \tfrac{1}{2}\nabla_\theta \,\mathrm{tr}\, \theta^\top \mathbf{X}^\top \mathbf{X}\theta - 2\nabla_\theta\,\mathrm{tr}\,\mathbf{y}^\top \mathbf{X}\theta.
\end{aligned}
$$

Using $\mathrm{tr}\, AB = \mathrm{tr}\, BA$, with $A = \mathbf{y}^\top \mathbf{X}, B = \theta$.

$$\tfrac{1}{2}\nabla_\theta \,\mathrm{tr}\, \theta^\top \mathbf{X}^\top \mathbf{X}\theta - 2\nabla_\theta\,\mathrm{tr}\,\theta\mathbf{y}^\top \mathbf{X}.$$

Using $\nabla_{A^\top}\,\mathrm{tr}\, ABA^\top C = B^\top A^\top C^\top + BA^\top C$,
with $A^\top = \theta, B = \mathbf{X}^\top\mathbf{X}, C = I$,
and using $\nabla_A\,\mathrm{tr}\, AB = B^\top$, with $A = \theta, B = \mathbf{y}^\top\mathbf{X}$.

$$
\begin{aligned}
&= \tfrac{1}{2}(\mathbf{X}^\top\mathbf{X}\theta + \mathbf{X}^\top\mathbf{X}\theta - 2\mathbf{X}^\top\mathbf{y}). \\
&= \mathbf{X}^\top\mathbf{X}\theta - \mathbf{X}^\top\mathbf{y}.
\end{aligned}
$$

Why the Cost Function J is Reasonable?

Given a training example i, we may write

$$y_i = \theta^\top \mathbf{x}_i + \epsilon_i,$$

with the assumption

$$\epsilon_i \sim \mathcal{N}(0, \sigma^2).$$

Therefore, the density of ϵ_i is given by

$$p(\epsilon_i) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(- \frac{(\epsilon_i)^2}{2\sigma^2} \right).$$

This implies

$$p(y_i | \mathbf{x}_i; \theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(- \frac{(y_i - \theta^\top \mathbf{x}_i)^2}{2\sigma^2} \right).$$

Likelihood of θ

The likelihood of θ is:

$$L(\theta) = L(\theta; \mathbf{X}; \mathbf{y}) = p(\mathbf{y}|\mathbf{X}; \theta).$$

Given the independence assumption on the ϵ_i's, we can also write:

$$\begin{aligned} L(\theta) &= \prod_{i=1}^{m} p(y_i|\mathbf{x}_i; \theta) \\ &= \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(- \frac{(y_i - \theta^\top \mathbf{x}_i)^2}{2\sigma^2} \right). \end{aligned}$$

Maximum Likelihood of θ

$$
\begin{aligned}
\ell &= \log L(\theta) \\
&= \log \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(- \frac{(y_i - \theta^\top \mathbf{x}_i)^2}{2\sigma^2} \right) \\
&= \sum_{i=1}^{m} \log \frac{1}{\sqrt{2\pi}\sigma} \exp\left(- \frac{(y_i - \theta^\top \mathbf{x}_i)^2}{2\sigma^2} \right) \\
&= m \log \tfrac{1}{\sqrt{2\pi}\sigma} - \tfrac{1}{\sigma^2} \cdot \tfrac{1}{2} \sum_{i=1}^{m} (y_i - \theta^\top \mathbf{x}_i)^2
\end{aligned}
$$

Hence, maximizing $\ell(\theta)$ gives the same answer as minimizing

$$
\frac{1}{2} \sum_{i=1}^{m} (y_i - \theta^\top \mathbf{x}_i)^2.
$$

Locally Adjusting the Model

The algorithm works as follows:

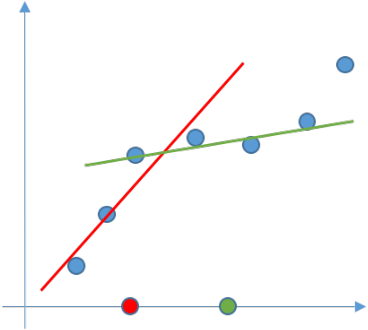
1. Fit θ to minimize $\sum_i w_i(y_i - \theta^\top x_i)^2$.
2. Output $\theta^\top x$.

Where w_i's are non-negative valued weights.

A good choice for the weights is:

$$w_i = \exp\left(-\frac{(x_i - x)^2}{2\tau^2}\right)$$

Locally Adjusting the Model



Thank you!

Dr. Víctor Uc Cetina
cetina@informatik.uni-hamburg.de