CSE 4502/5717 Big Data Analytics

Notes by Yufan Zhang

Lecture 17

04/04/2018

LINEAR REGRESSION:

Consider a function $f: \mathbb{R}^n \longrightarrow \mathbb{R}$.

A simple model for f could be:

$f(x_1, x_2, \ldots, x_n) = w_1x_1 + w_2x_2 + \ldots + w_nx_n$. To learn this function, we will be supplied with a series of examples.

INPUT: Examples: $(x_i^1, x_i^2, x_i^3, \ldots, x_i^n;\ y^i)$, $1 \le i \le m$. Let $\boldsymbol{y} = (y_1, y_2, \cdots, y_m)^T$.

Let $\boldsymbol{X} = \begin{bmatrix} x_1^1 & x_1^2 & \cdots & x_1^n \\ x_2^1 & x_2^2 & & x_2^n \\ \vdots & & \ddots & \vdots \\ x_m^1 & x_m^2 & \cdots & x_m^n \end{bmatrix}$

Estimates from this model will be: $\hat{\boldsymbol{y}} = \begin{bmatrix} \widehat{y_1} \\ \widehat{y_2} \\ \vdots \\ \widehat{y_m} \end{bmatrix} = \boldsymbol{Xw}$, where $\boldsymbol{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}$

We want to minimize: $\frac{1}{m}\sum_{i=1}^{m}(y_i - \widehat{y_i})^2$ i.e., we want to minimize $\frac{1}{m}\|\boldsymbol{Xw} - \boldsymbol{y}\|_2^2$

$\|\boldsymbol{Xw} - \boldsymbol{y}\|_2^2 = (\boldsymbol{Xw} - \boldsymbol{y})^{\mathrm{T}}(\boldsymbol{Xw} - \boldsymbol{y})$

We want: $\nabla_{\boldsymbol{w}} \frac{1}{m}(\boldsymbol{Xw} - \boldsymbol{y})^{\mathrm{T}}(\boldsymbol{Xw} - \boldsymbol{y}) = 0$

$\Rightarrow \nabla_{\boldsymbol{w}}(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{X}^{\mathrm{T}} - \boldsymbol{y}^{\mathrm{T}})(\boldsymbol{Xw} - \boldsymbol{y}) = 0$

$\Rightarrow \nabla_{\boldsymbol{w}}(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{Xw} - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{X}^{\mathrm{T}}\mathbf{y} - \boldsymbol{y}^{\mathrm{T}}\mathbf{Xw} + \boldsymbol{y}^{\mathrm{T}}\mathbf{y}) = 0$

$\Rightarrow 2\boldsymbol{w}^{\mathrm{T}}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X} - \mathbf{2}\boldsymbol{y}^{\mathrm{T}}\boldsymbol{X} = 0$

$\Rightarrow \boldsymbol{X}^{\mathrm{T}}\boldsymbol{Xw} - \boldsymbol{X}^{\mathrm{T}}\boldsymbol{y} = 0$

$\Rightarrow \boldsymbol{X}^{\mathrm{T}}\boldsymbol{Xw} = \boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}$

$\Rightarrow \boldsymbol{w} = (\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})^{-\mathbf{1}}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}$

We have utilized the following fact: If $\alpha = \boldsymbol{x}^{\mathrm{T}}\boldsymbol{Ax}$, then $\frac{\partial\alpha}{\partial\boldsymbol{x}} = 2\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}$, if $\boldsymbol{A}$ is symmetric.

TIME Complexity:

$\boldsymbol{X}$ is $(m \times n)$; $\boldsymbol{y}$ is $(m \times 1)$

① to compute $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X} \rightarrow O(n^2m)$

② to compute $(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})^{-1} \rightarrow \mathrm{O}(n^3)$

③ to compute $(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})^{-1}\boldsymbol{X}^{\mathrm{T}} \rightarrow \mathrm{O}(n^2m)$

④ to compute $(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})^{-1}\boldsymbol{X}^{T}\boldsymbol{y} \rightarrow \mathrm{O}(mn)$

Total Run time = $\mathrm{O}(n^2m + n^3)$.

Example: Let $f: \mathbb{R}^2 \longrightarrow \mathbb{R}$. We could use the following model: $f(x_1, x_2) = w_1x_1 + w_2x_2$.

Let the input data (i.e., examples) be (0, 1; 2), (1, 0; 4), (1, 1; 4).

The loss function: $L(w_1, w_2) = (w_2 - 2)^2 + (w_1 - 4)^2 + (w_1 + w_2 - 4)^2$

To get optimal values for w_1 and w_2, we'll set $\frac{\partial L}{\partial w_1} = 0$; and $\frac{\partial L}{\partial w_2} = 0$.

$L = w_2^2 + 4 - 4w_2 + w_1^2 + 16 - 8w_1 + w_1^2 + w_2^2 + 16 + 2w_1w_2 - 8w_1 - 8w_2$

$= 2w_1^2 + 2w_2^2 + 2w_1w_2 - 16w_1 - 12w_2 + 36.$

$\frac{\partial L}{\partial w_1} = 4w_1 + 2w_2 - 16 = 0$ ——①

$\frac{\partial L}{\partial w_2} = 4w_2 + 2w_1 - 12 = 0$ ——②

① – 2② $\Rightarrow -6w_2 + 8 = 0 \Rightarrow w_2 = \frac{4}{3}$

$2w_1 = 12 - 4 \cdot \frac{4}{3} = 12 - \frac{16}{3} = \frac{20}{3} \Rightarrow w_1 = \frac{10}{3}$

We normally use a more general regressor:

$f(x_1, x_2, \ldots, x_n) = w_1x_1 + w_2x_2 + \ldots + w_nx_n + b.$

We can handle this generalization by extending $\boldsymbol{x}$ with $\boldsymbol{x}' = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \\ 1 \end{bmatrix}$.

A model that can represent more complex functions is the ARTIFICAL NEURAL NETWORK (ANN). ANNs have been employed since a long time ago. They were known with different names:

Cybernetics

⇓

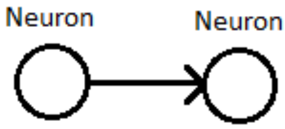
Connectionist models

⇓

DEEP NEURAL NETWORK

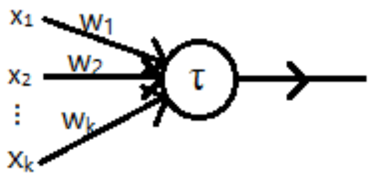
A neural network (NN) is a weighted directed graph $G(V, E)$.

Each node in V corresponds to a neuron. If there is a directed edge from a neuron u to another neuron v, a signal passes from u to v. I.e., v gets an input from u.



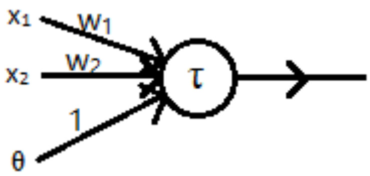
In any NN there are some input nodes and some output nodes. Input flows through the other nodes in the network, gets transformed, and finally reaches the output nodes.

Example: A PERCEPTRON:



output is 1 if $\sum_{i=1}^{k} w_i x_i \geq \tau$; It is zero otherwise.

A PERCEPTRON can be thought of as a BINARY CLASSIFIER. Consider the following perceptron:



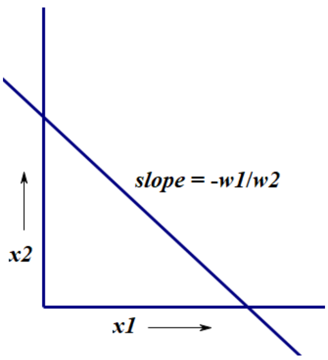
output is 1 if $w_1x_1 + w_2x_2 + \theta \geq \tau$

$\Rightarrow w_1x_1 + w_2x_2 \geq u$

$u \rightarrow$ a constant

$\Rightarrow w_2x_2 \geq u - w_1x_1$

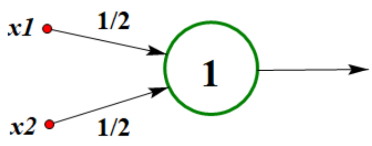
$\Rightarrow x_2 \geq \frac{-w_1}{w_2} x_1 + \frac{u}{w_2}$

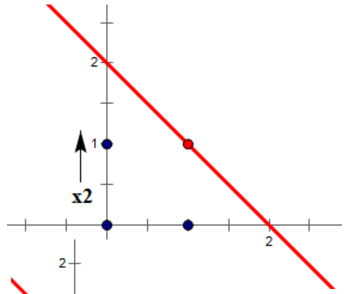


A perceptron is a binary classifier when the two classes can be separated by a straight line.

REALIZING Boolean AND:

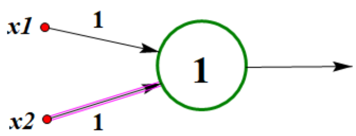
x_2	x_1	$x_1 \wedge x_2$
0	0	0
0	1	0
1	0	0
1	1	1

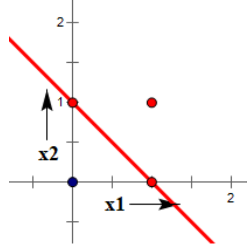




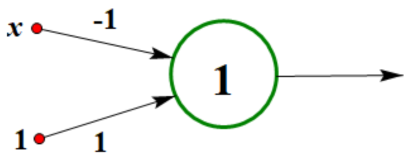
BOOLEAN OR:

x_2	x_1	$x_1 \vee x_2$
0	0	0
0	1	1
1	0	1
1	1	1



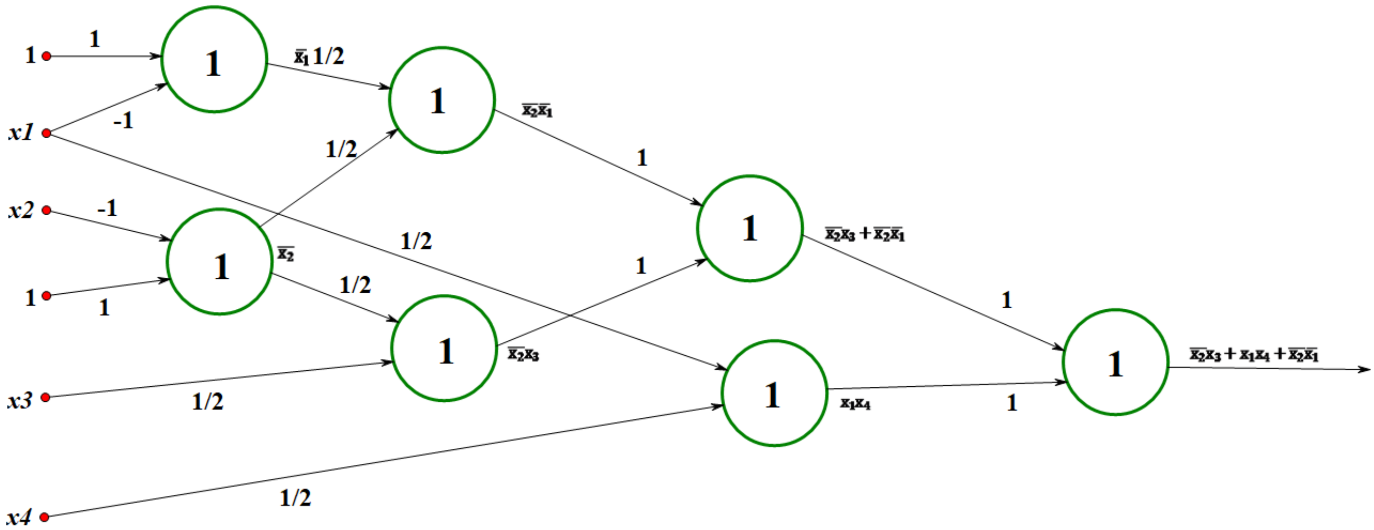


Boolean NOT:

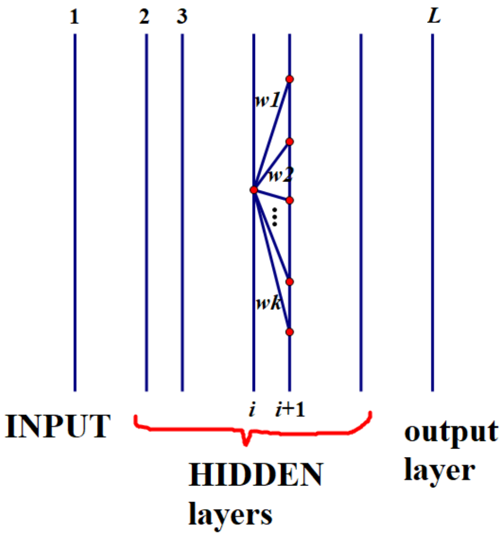


COROLLARY: ANY Boolean function can be realized with a neural network using perceptrons.

Example: F = $\overline{x_2}x_3 + x_1x_4 + \overline{x_2}\ \overline{x_1}$



A General NN looks like:



To use GRADIENT DESCENT, we have to make sure that the output from every node is continuous.

Therefore, we apply an “ACTIVATION FUNCTION” at each node.