

# Artificial Intelligence Introduction

## Logistic Regressor

Ph.D. Gerardo Marx Chávez-Campos

Instituto Tecnológico de Morelia



# Logistic Regression

As we discussed in Unit 1, some regression algorithms can also be used for classification (and vice versa). Logistic Regression (also called *Logit Regression*) is commonly used to estimate the probability that an instance belongs to a particular class.

If the estimated probability is greater than 50%, then the model predicts that the instance belongs to that class (called the positive class, labeled “1”), or else it predicts that it does not (i.e., it belongs to the negative class, labeled “0”).

**This makes it a binary classifier.**



# Estimating Probabilities

**So how does it work?** Just like a Linear Regression (LR) model, a Logistic Regression model computes a weighted sum of the input features (plus a bias term). However, instead of outputting the result directly as the LR model does, it outputs the logistic of this result:

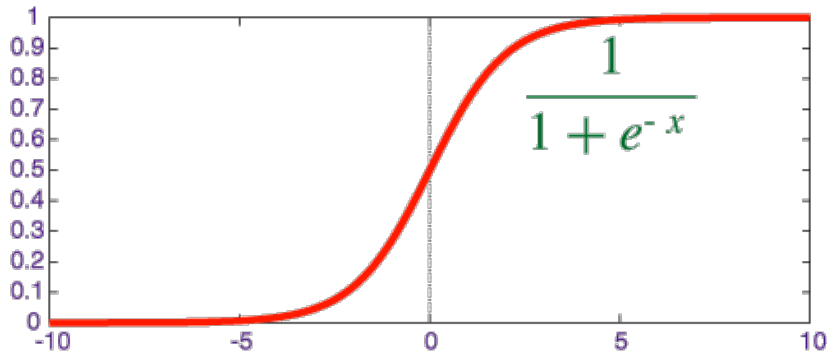
$$\hat{p} = h_{\theta}(\mathbf{x}) = \sigma(\mathbf{x}^T \boldsymbol{\theta}) \quad (1)$$

here  $\sigma$  is the sigmoid function defined as follows:

$$\sigma(t) = \frac{1}{1 + e^{-t}}$$



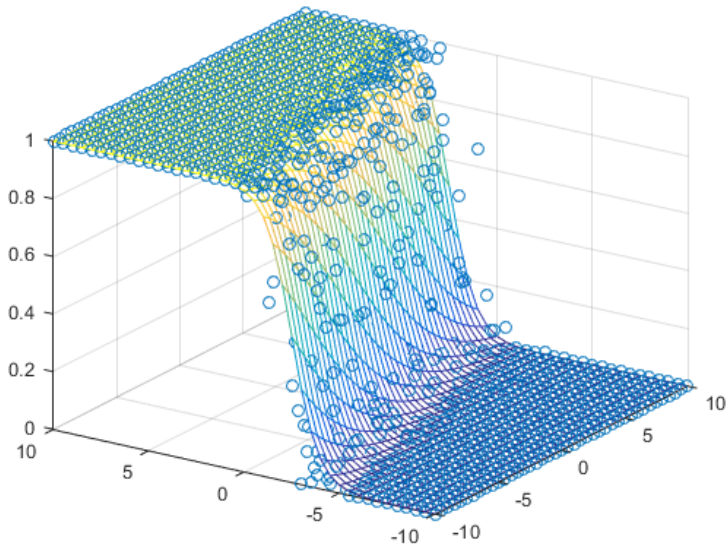
# One parameter Logistic Regressor



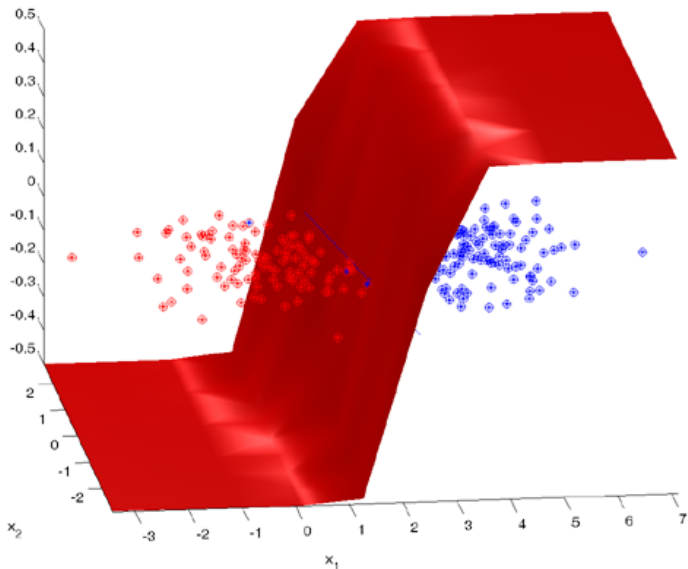
# Two parameters Logistic Regressor



# Two parameters Logistic Regressor 3D



# Two parameters Logistic Regressor 3D



Once the Logistic Regression model has estimated the probability  $\hat{p} = h_{\theta}(x)$  that an instance  $x$  belongs to the positive class, it can make its prediction  $\hat{y}$  easily:

$$\hat{y} = \begin{cases} 0 & \text{if } \hat{p} < 0.5 \\ 1 & \text{if } \hat{p} \geq 0.5 \end{cases} \quad (3)$$





# The Cost Function

The objective of the training is to set the parameter vector  $\theta$  so that the model estimates high probabilities for positive instances ( $y = 1$ ) and low probabilities for negative instances ( $y = 0$ ).

$$c(\theta) = \begin{cases} -\log(\hat{p}) & \text{if } y = 1 \\ -\log(1 - \hat{p}) & \text{if } y = 0 \end{cases} \quad (4)$$

This idea is captured by the cost function for a single training instance  $x$ .



# The Cost Function

The cost function over the whole training set is simply the average cost over all training instances. It can be written in a single expression (as you can verify easily), called the log loss, shown in Equation :

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m \left[ y^{(i)} \log(\hat{p}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{p}^{(i)}) \right] \quad (5)$$




# Gradient

$$\frac{\partial J(\theta)}{\partial \theta} = \sum_{i=1}^m \frac{1}{m} (\sigma(\theta^T x^i) - y^i) x_j^i \quad (6)$$



# Referencias

-  Géron, Aurélien. "Hands-on machine learning with scikit-learn and tensorflow: Concepts." Tools, and Techniques to build intelligent systems (2017).

