# Aprendizaje de Máquina

## ITAM

Semestre agosto-diciembre 2017

# Menu

- In this session we will see how to use what we know about doing linear regression to perform classification

    - We are going to see the perceptron model

# How to convert a regressor into a classifier

- Suppose we have the following data

**Moroso**
**Regresión lineal**

# How to convert a regressor into a classifier



**Moroso**
**Regresión lineal**

- Moroso
- Linear (Moroso)
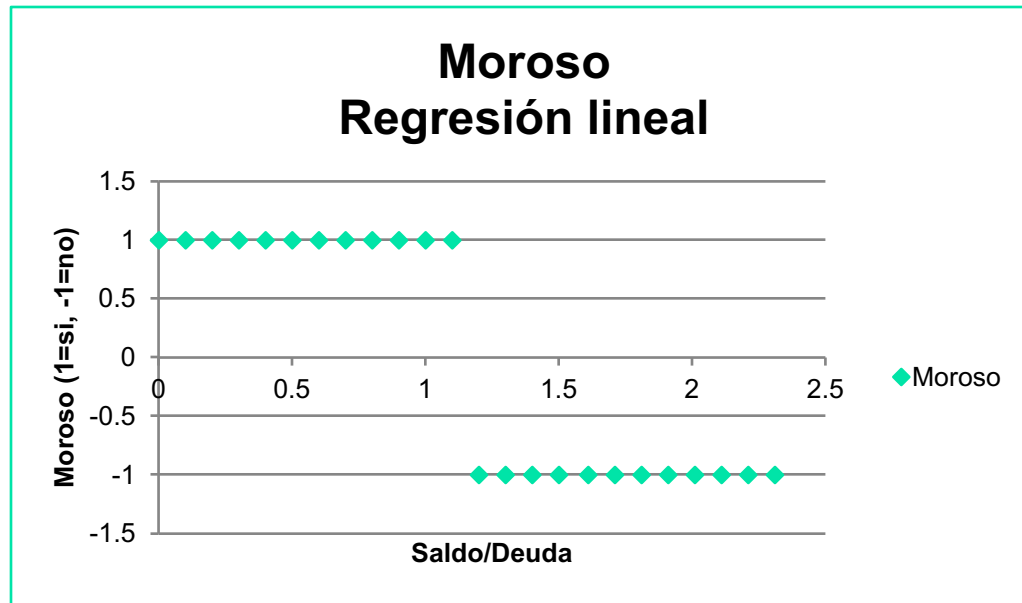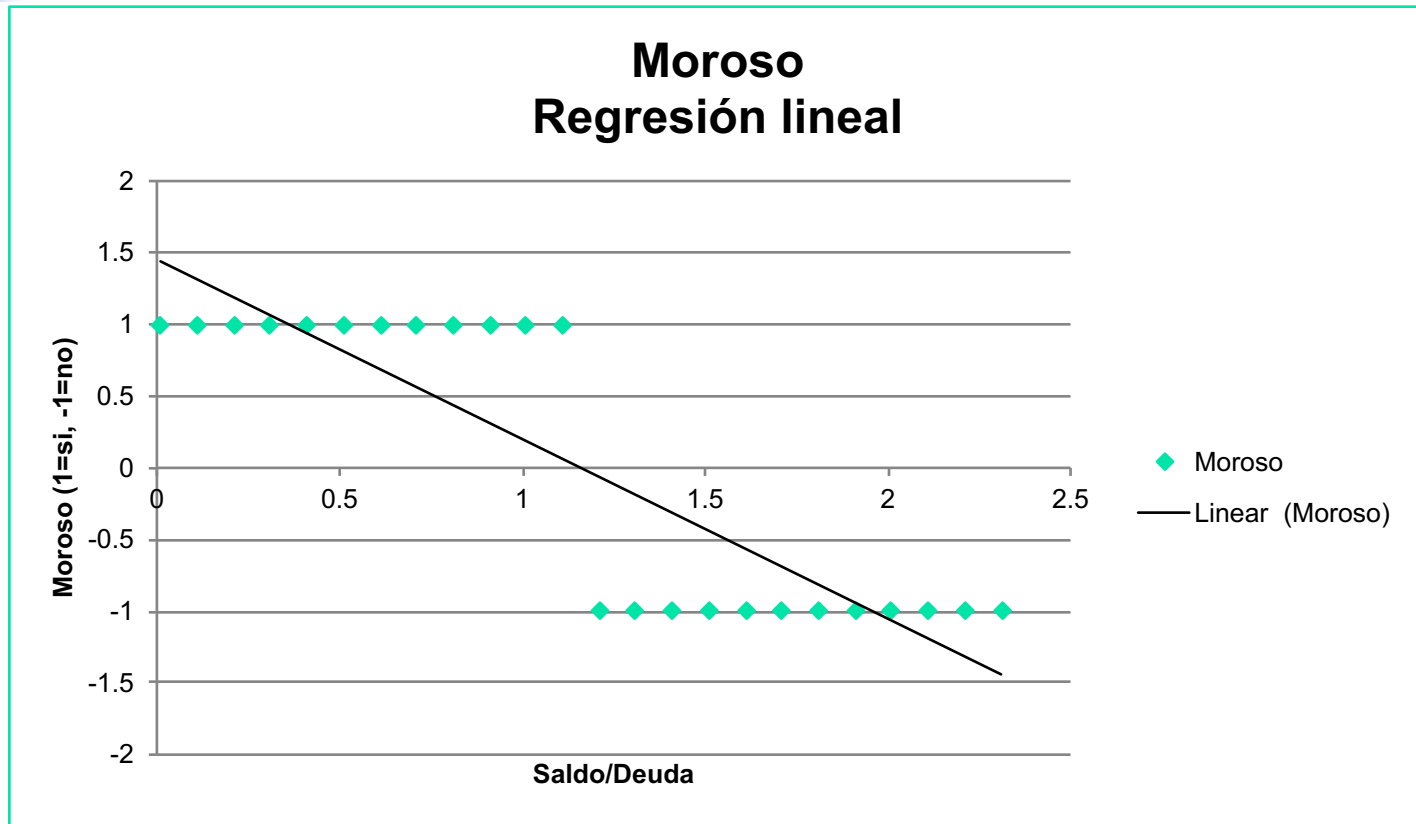
# How to convert a regressor into a classifier

- It doesnt make much sense to allow our model to take values higher than 1 and lower than -1. There is no data with such values

- Solution: Limit the possible values to this range via a transfer function---a function that takes the output of the regressor and transforms it into something else

# The perceptron: a model neuron

- The function that represents the activation of the perceptron is

  - $g(x_1,x_2,..,x_n)= \begin{cases} 1 \text{ si } w_o+\sum_{i=1,n}w_ix_i >=0 \\ -1 \text{ otherwise} \end{cases}$

  - We can think of $w_o$ as a threshold value since it does not depende on an input variable.

  - We could say that the perceptron fires if there is enough stimulus in the input, if the weighed sum of the inputs is greater than $-w_o$.

# Perceptron
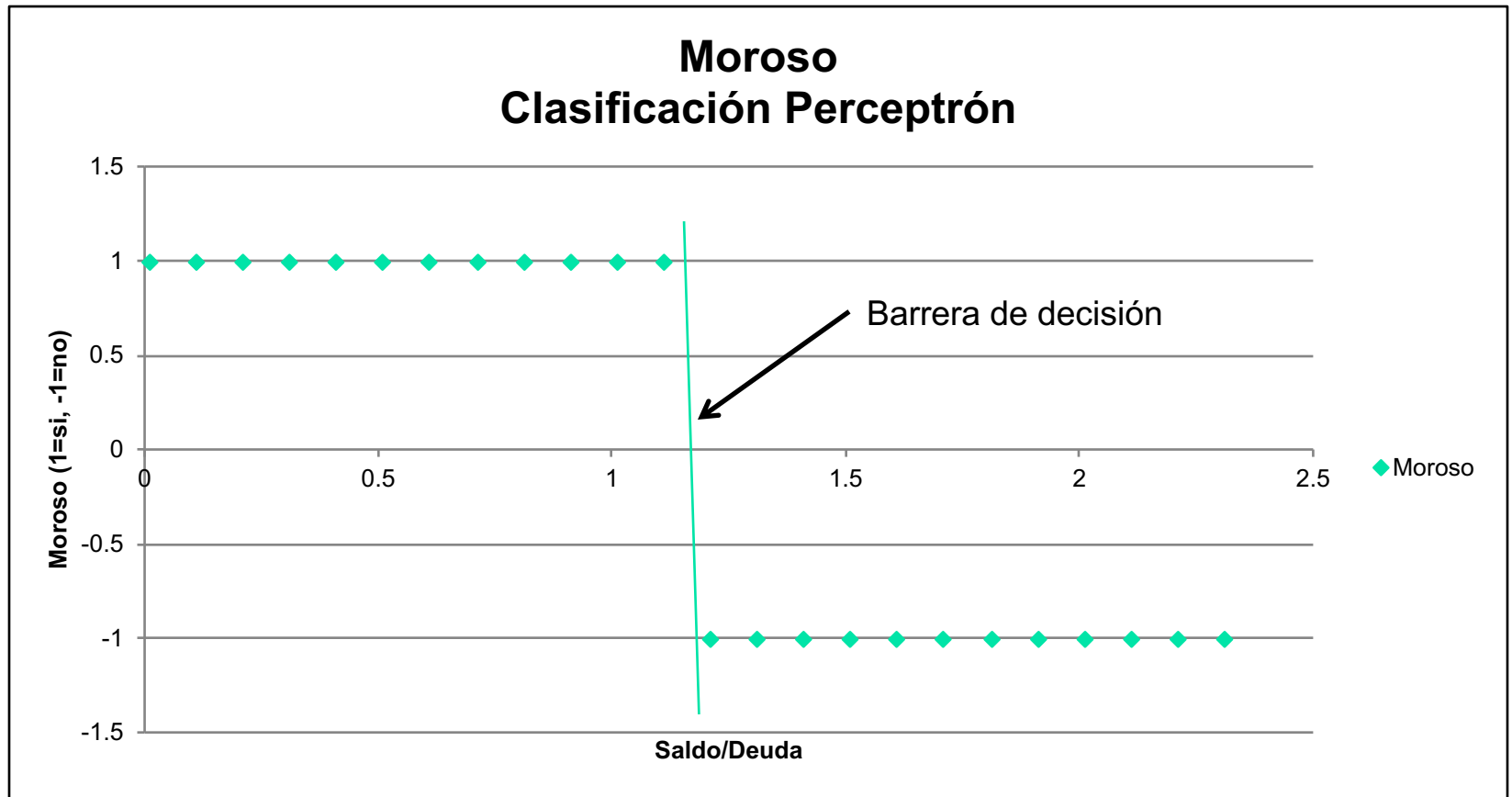# Transfer function: Step function
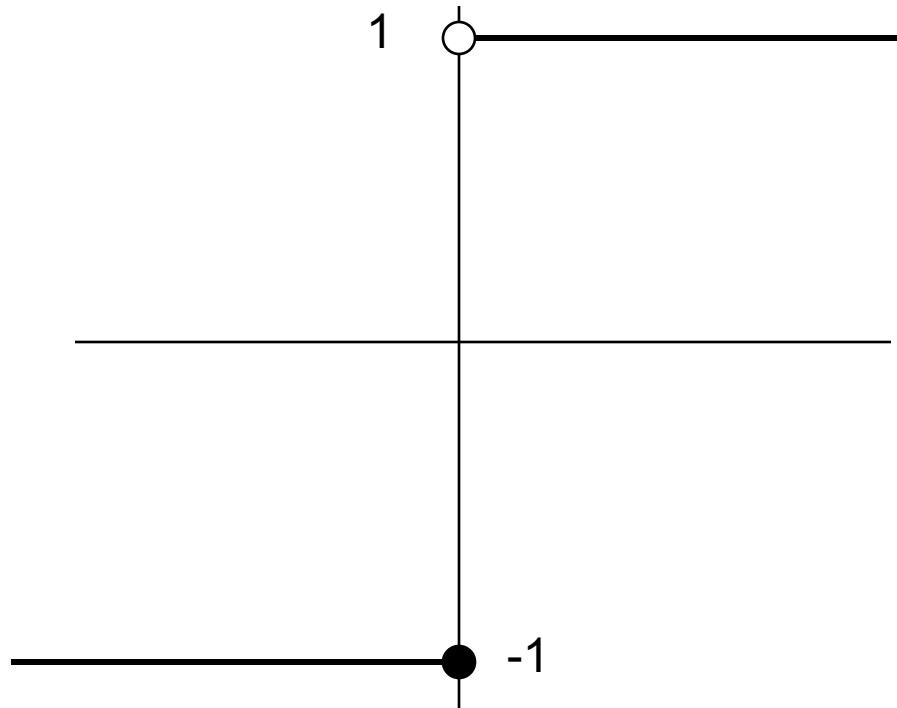
# Representational ability
# Perceptron

- To ilustrate the representation power of the perceptron we can plot the equiation
  $\sum_{i=0,n} w_i x_i = 0$

- Since when $\sum_{i=0,n} w_i x_i$ is greater that or equal to zero it classifies an input as 1 and -1 otherwise

  - $\sum_{i=0,n} w_i x_i = 0$ represents a decision bounday

# From the above example

**Moroso
Clasificación Perceptrón**

Barrera de decisión

Saldo/Deuda

Moroso (1=si, -1=no)

Moroso

# Learning Algorithm Perceptron

- For each training example($\boldsymbol{X}$,y)
    - Calculate g with the current w's
    - For each $w_i$ ,
        - $w_i$ <---$w_i$ + $\eta$(y-g($\boldsymbol{X}$)) $x_i$

- Where $\eta$ is a small constant lower than 1 (learning constant)

- The rule is applied iteratively a fixed number of times, or until the error reaches a desired value, or if no further decrease in the error is detected

- Note again that the difference with iterative regression is the function g

$$g(x_1,x_2,..,x_n)= \begin{cases} 1 \text{ si } \sum_{i=0,n}w_ix_i >0 \\ -1 \text{ de otra forma} \end{cases}$$

# Example

|       | X0 | X1  | X2 | X3  | X4 | X5 | X6 |
|-------|----|-----|----|-----|----|----|----|
| x´s   | 1  | 1   | 1  | 0   | 2  | 0  | 1  |
| w´s   | -1 | -0.5| 1  | 0.5 | 0  | 1  | 1  |
| $x_iw_i$ | -1 | -0.5| 1  | 0   | 0  | 0  | 1  |

**Before without g:** y=-1, V^(**X**)=0.5, Error=-1-0.5=-1.5, $\eta$ =0.1

**Now:** y=-1, V^(**X**)=g(**X**)=1, **Error=-1-1=-2.0**, $\eta$ =0.1

w0 = -1 + 0.1(-2.0)1=-1.2      w4 = 0 + 0.1(-2.0)2 = -0.4

w1 = -0.5 + 0.1(-2.0)1=-0.7     w6 =1 + 0.1(-2.0)1 = 0.8

w2 = 1 + 0.1(-2.0)1 = 0.8

# Exercise

- Modify the iterative regression algorithm to include the step function as a transfer function
- Generate data for the logical *and* function

| X1 | X2 | X1 and X2 |
|----|----|-----------|
| 0  | 0  | 0         |
| 0  | 1  | 0         |
| 1  | 0  | 0         |
| 1  | 1  | 1         |

- Train the perceptron this data set
  - Visualize the data
  - Plot the decision boundary
  - Calculate the classification error
    - Number of misses over number of examples

# Exercise

- Repeat for the XOR function

| X1 | X2 | X1 xor X2 |
|----|----|-----------|
| 0  | 0  | 0         |
| 0  | 1  | 1         |
| 1  | 0  | 1         |
| 1  | 1  | 0         |

- For those that finish early

  - Repeat the *and* exercise

  - How much does the separating hyperplane change from run to run?

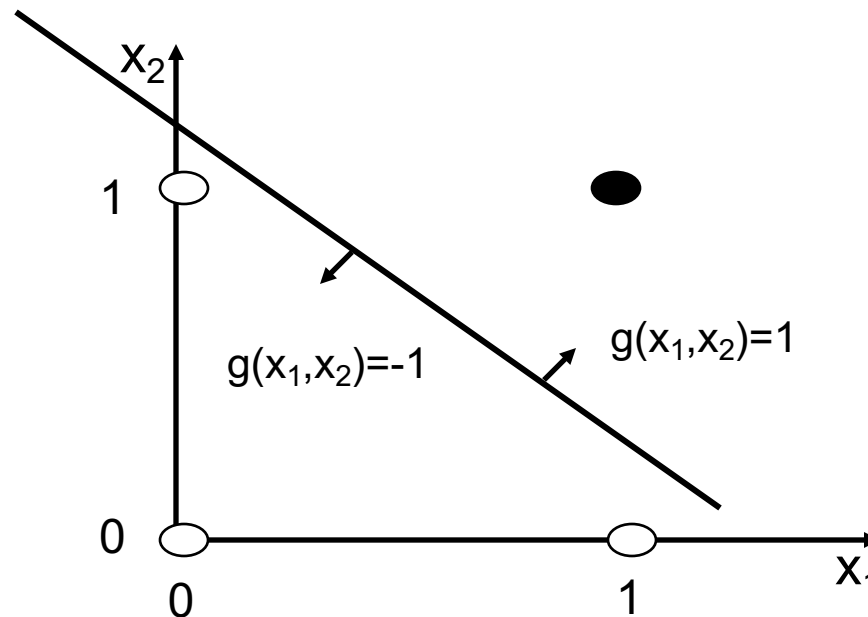    - Using or not using regularization (0.001)

# Assignment

- Program a perceptron in tensorflow
- Follow tensorflow's beginners tutorial

# Representation Power
## Perceptrón



• White and black circles belong to different categories