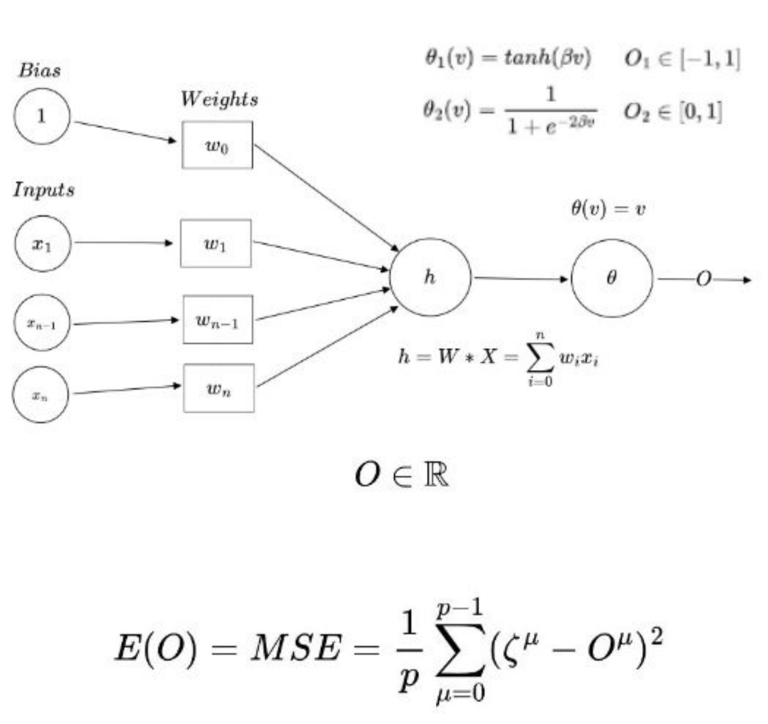
Trabajo Práctico N°3
Perceptrón Simple y Multicapa

Grupo 7:

- Valentin Ye Li
- Nicolás Birsa
- Andres Podgorny
- Ouss Slaitane
- Hugo Lichtenberger

Ej2. Perceptrón Simple

1. Arquitectura Lineal vs No lineal

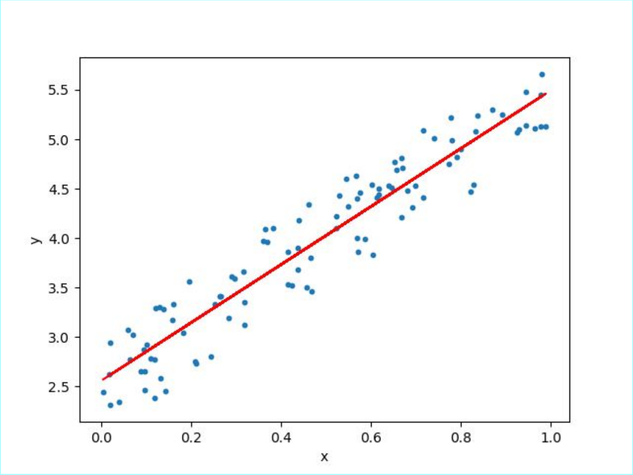


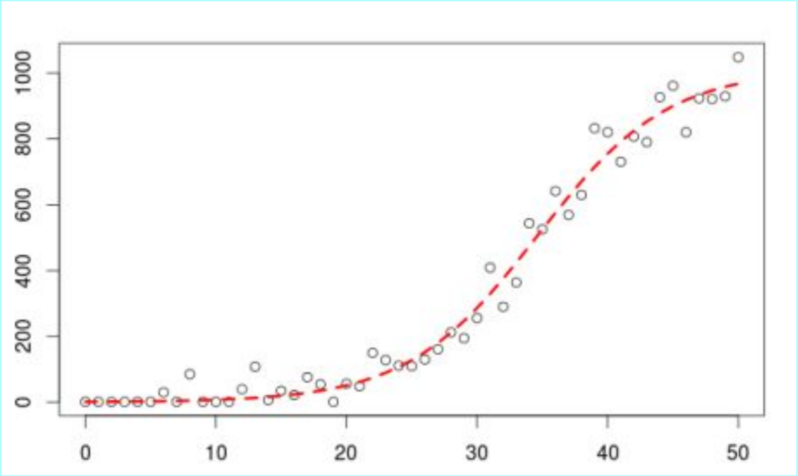
Arquitectura

Entradas: x1 x2 x3

Funciones de activación:
-Lineal: Identidad
-No Lineal: Tanh y Función Logística

Perceptrón Lineal y No lineal





Lineal: Resuelve problemas donde haya una relación cuasilineal entre inputs y outputs

No lineal: Misma estructura que el Lineal, pero su función de activación ***sigmoidea*** le permite resolver problemas no lineales

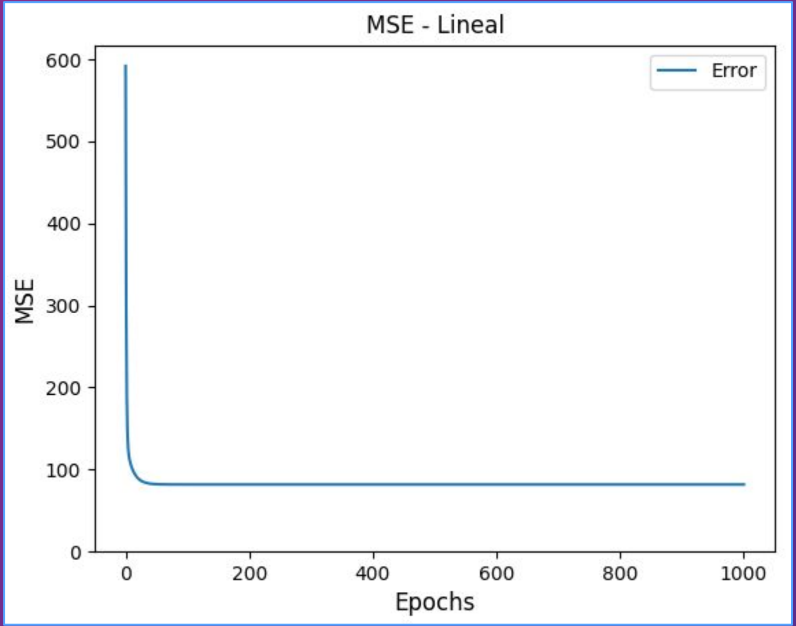
Escalado de valores

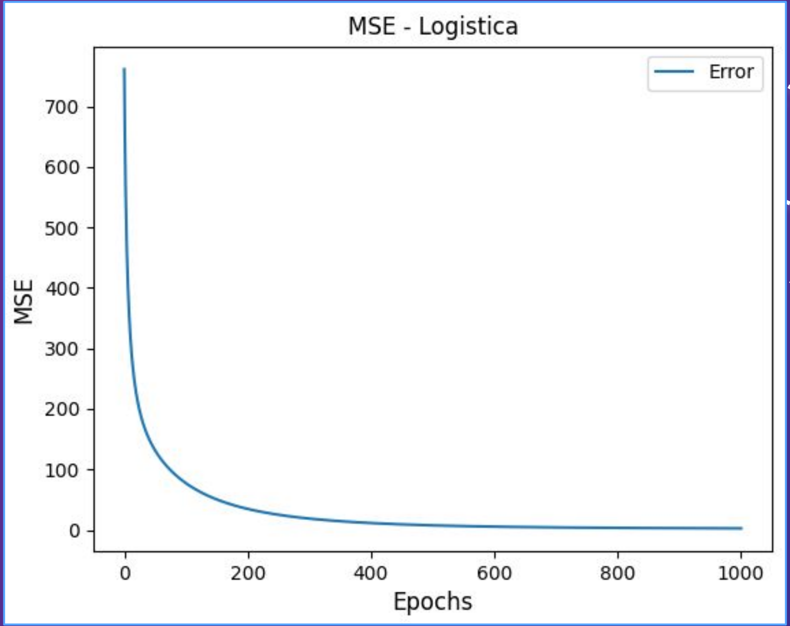
Internamente se escalan los valores de la columna "y" para poder hacer el correcto cálculo del error con las funciones de activación sigmoideas.

Igualmente, en los gráficos figura el error **MSE** de los valores sin escalar.

$$x' = a + \frac{(x - \min(x))(b - a)}{\max(x) - \min(x)}$$

Diferencia entre Lineal y No lineal





Epochs: 1000
Learning rate: 0.005

Accepted error: 0.01
Beta: 1

El MSE del Lineal se "estanca" cerca de 100, mostrando una muy mala aproximación.
El del No lineal es mucho menor, acercándose a cero con el paso de épocas.

2. Separación en Test y Train

Selección de los conjuntos train y test

Se pensaron dos tipos de separación para los datos de entrada:

1. Random, dando el **mismo valor a cada dato**. Puede recaer en que haya datos "repetidos" presentes en los conjuntos

2. Separando en "clases" con datos repetidos en cada una, y destinando sólo un valor de cada clase al train set. Así se toman más valores únicos en el train set y el modelo generaliza mejor.

Nosotros **implementamos el segundo**, que se asemeja a la idea de **Stratify**, y determinamos si dos datos no son únicos mediante su diferencia en valores $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$ e $\mathbf{y}$.

¿Cuándo dos valores son “repetidos”?

Se toman dos valores como repetidos cuando:

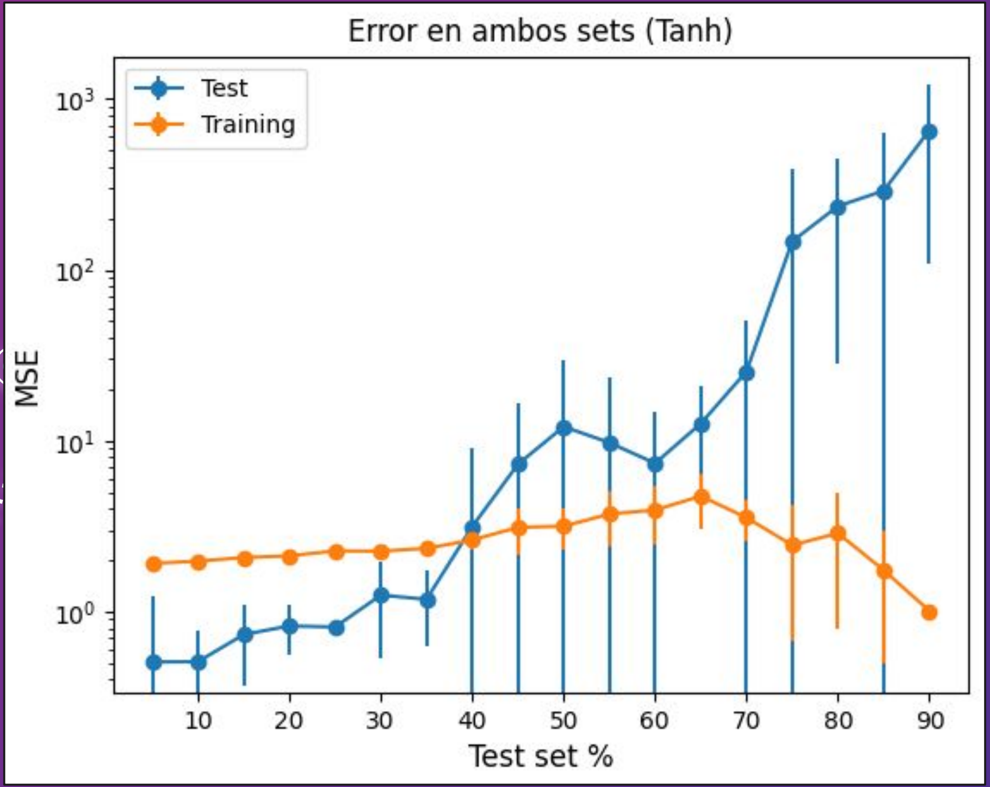
$$\sqrt{(x_{11} - x_{21})^2 + (x_{21} - x_{22})^2 + (x_{31} - x_{32})^2} \leq \delta_x = 1.5$$

y

$$|y_{11} - y_{21}| \leq \delta_y = 10$$

Ejemplo:
[0, 0.4, 2.7, 61.301] y **[0.4, 0, 2.7, 68.568]** son datos tomados como **repetidos**

Separación usando Tanh

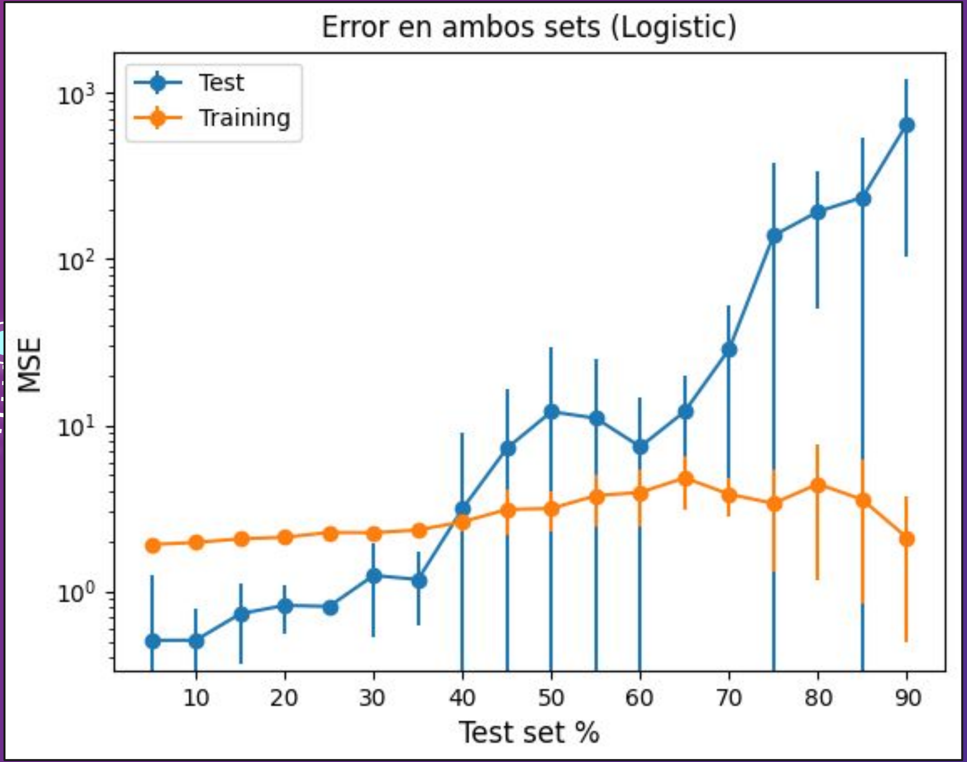


10 ejecuciones por valor
Epochs: 10000
Learning rate: 0.005
Accepted error (MSE): 1
Beta: 1

Para valores altos el *train set* tiene pocos elementos y se observa un **underfitting**.

Para valores bajos, como el *test set* tiene solo datos “repetidos” no se ve **overfitting**, pero sí una variación en el MSE.

Separación usando Logística



10 ejecuciones por valor
Epochs: 10000
Learning rate: 0.005
Accepted error (MSE): 1
Beta: 1

Se observa el mismo comportamiento que la Tanh.

En ambos casos el test set de 25% tiene **stderr=0**, indicando que el conjunto sólo tiene datos “repetidos”.

3. Capacidad de generalización (No lineal)

Error a través de las épocas

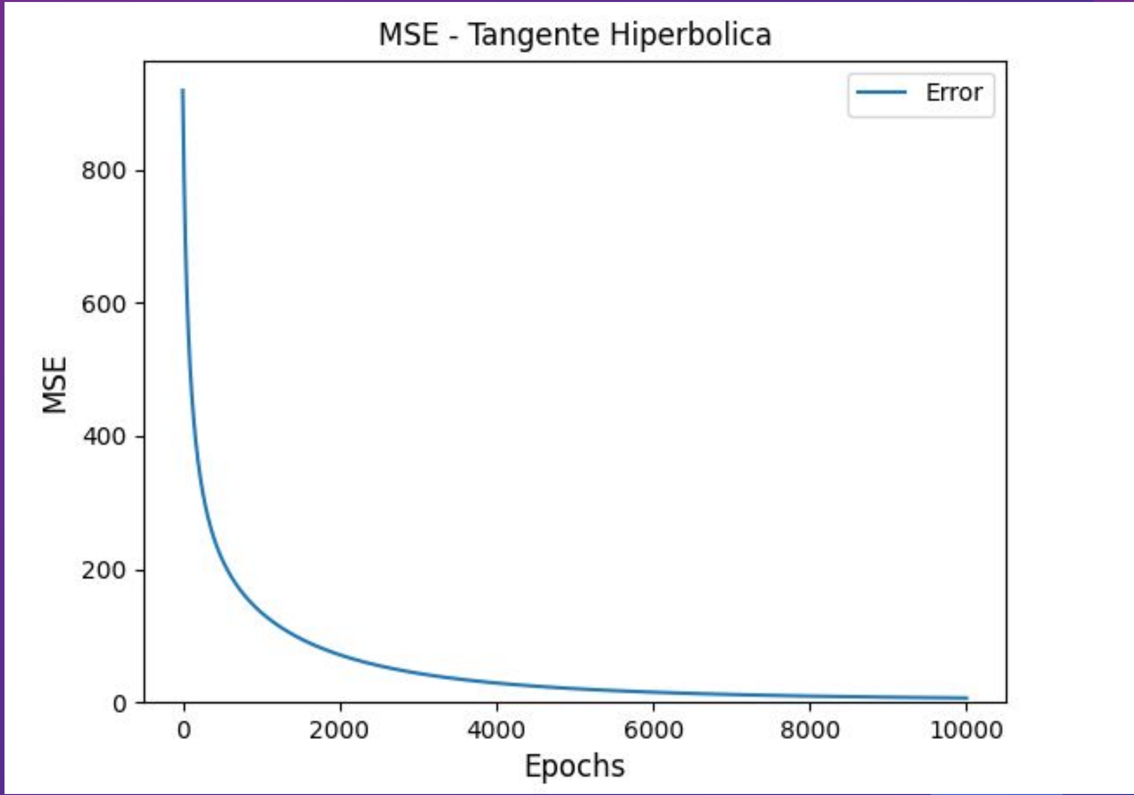
Al igual que en la comparación de Lineal vs No lineal, vemos el MSE a través de las épocas, pero esta vez usando la separación 25% test - 75% train vista en el ítem anterior.

Se observa el MSE del conjunto de test en cada época, incluido su valor final con la desviación estándar.

MSE - No lineal Tanh

Test set: 25%
Epochs: 10000
Learning rate: 0.0001
Accepted error: 0.001
Beta: 1

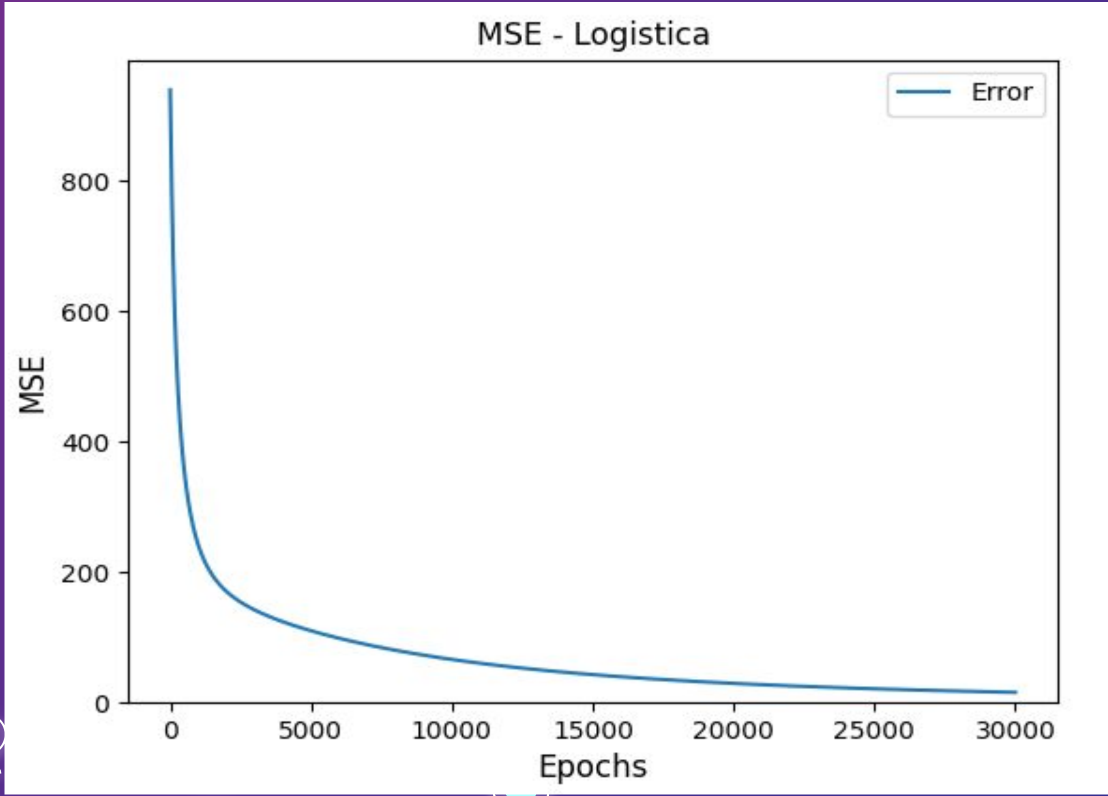
Non lineal MSE: 8.46
Std error: 0.0



MSE - No lineal Logística

Test set: 25%
Epochs: 30000
Learning rate: 0.0001
Accepted error: 0.001
Beta: 1

Non lineal MSE: 14.65
Std error: 0.0



Conclusiones

- Dado que el perceptrón lineal no puede aproximar una solución con bajo error, el problema se trata de uno no lineal.

- La mejor separación train-test se da con las proporciones de 25%-75%, donde todos los datos del train set son "únicos".

- Usando la separación mencionada, el modelo tiene buena capacidad de generalización.

¡Gracias!