ARISA Learning Material

Educational Profile and EQF level: DATA SCIENTIST – EQF 6

PLO: 1, 2, 3, 4, 5

Learning Unit (LU): MACHINE LEARNING: SUPERVISED

Topic: LINEAR MODELS AND LOGISTIC REGRESSION

Co-funded by the European Union

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

ARISA Learning Material 2024

This material is a draft version and is subject to change after review coordinated by the European Education and Culture Executive Agency (EACEA).

Authors: Universidad Internacional de La Rioja (UNIR)

Disclaimer: This learning material has been developed under the Erasmus+ project ARISA (Artificial Intelligence Skills Alliance) which aims to skill, upskill, and reskill individuals into high-demand software roles across the EU.

Co-funded by the European Union

This project has been funded with support from the European Commission. The material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

- About ARISA

- The Artificial Intelligence Skills Alliance (ARISA) is a four-year transnational project funded under the EU's Erasmus+ programme. It delivers a strategic approach to sectoral cooperation on the development of Artificial Intelligence (AI) skills in Europe.
- ARISA fast-tracks the upskilling and reskilling of employees, job seekers, business leaders, and policymakers into AI-related professions to open Europe to new business opportunities.
- ARISA regroups leading ICT representative bodies, education and training providers, qualification regulatory bodies, and a broad selection of stakeholders and social partners across the industry.

ARISA Partners & Associated Partners | LinkedIn | Twitter

- CODE

- EN AMAZON CLOUD ARISA 1 – ANN
- EN LOCAL TUTORIAL_DEEP_LEARNING_basics

ARISA Partners & Associated Partners | LinkedIn | Twitter

ARISA Learning Material

Educational Profile and EQF level: DATA SCIENTIST – EQF 6

PLO: 1, 2, 3, 4, 5

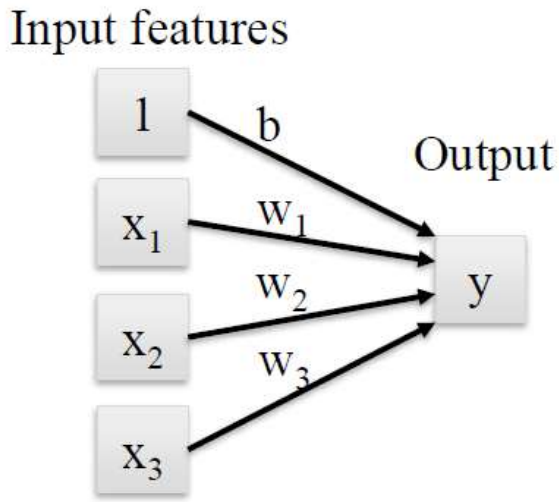
Learning Unit (LU): MACHINE LEARNING: SUPERVISED

Topic: LINEAR MODELS AND LOGISTIC REGRESSION

Co-funded by the European Union

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

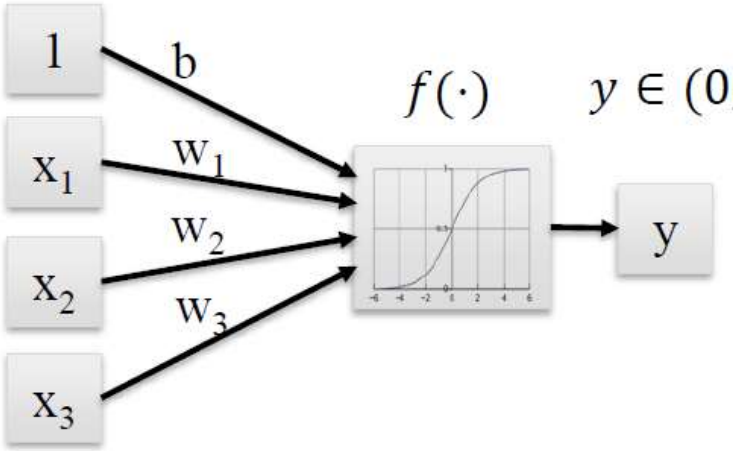
Regresión lineal

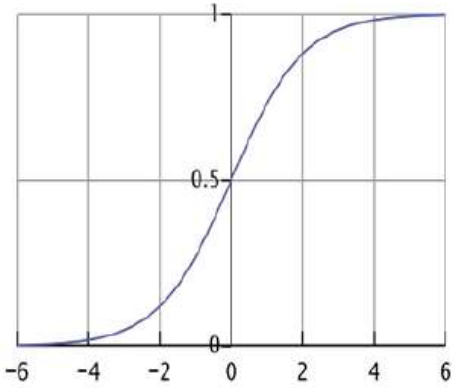


$$\hat{y} = \hat{b} + \hat{w}_1 \cdot x_1 + \cdots \hat{w}_n \cdot x_n$$

Linear models for classification: Logistic Regression

Input features





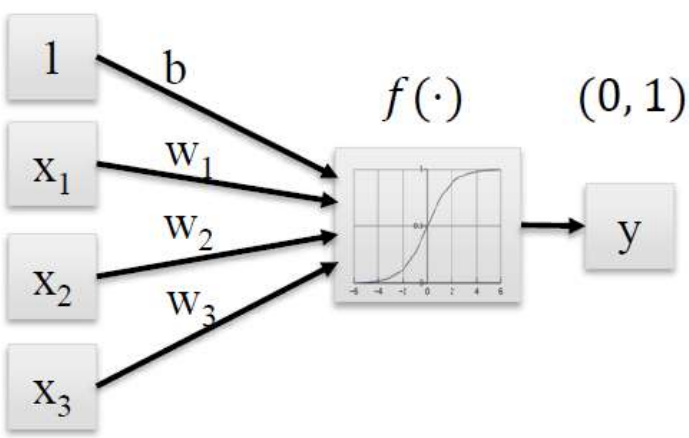
$$\hat{y} = \text{logistic}(\hat{b} + \hat{w}_1 \cdot x_1 + \cdots \hat{w}_n \cdot x_n)$$

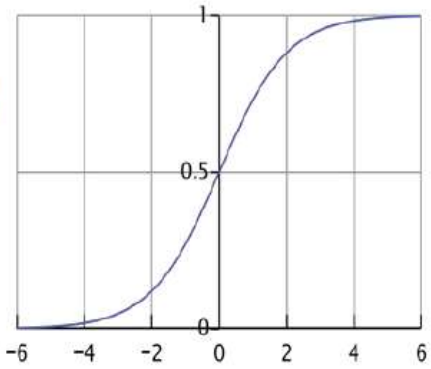
$$= \frac{1}{1 + \exp\left[-(\hat{b} + \hat{w}_1 \cdot x_1 + \cdots \hat{w}_n \cdot x_n)\right]}$$

- Title

Linear models for classification: Logistic Regression

Input features

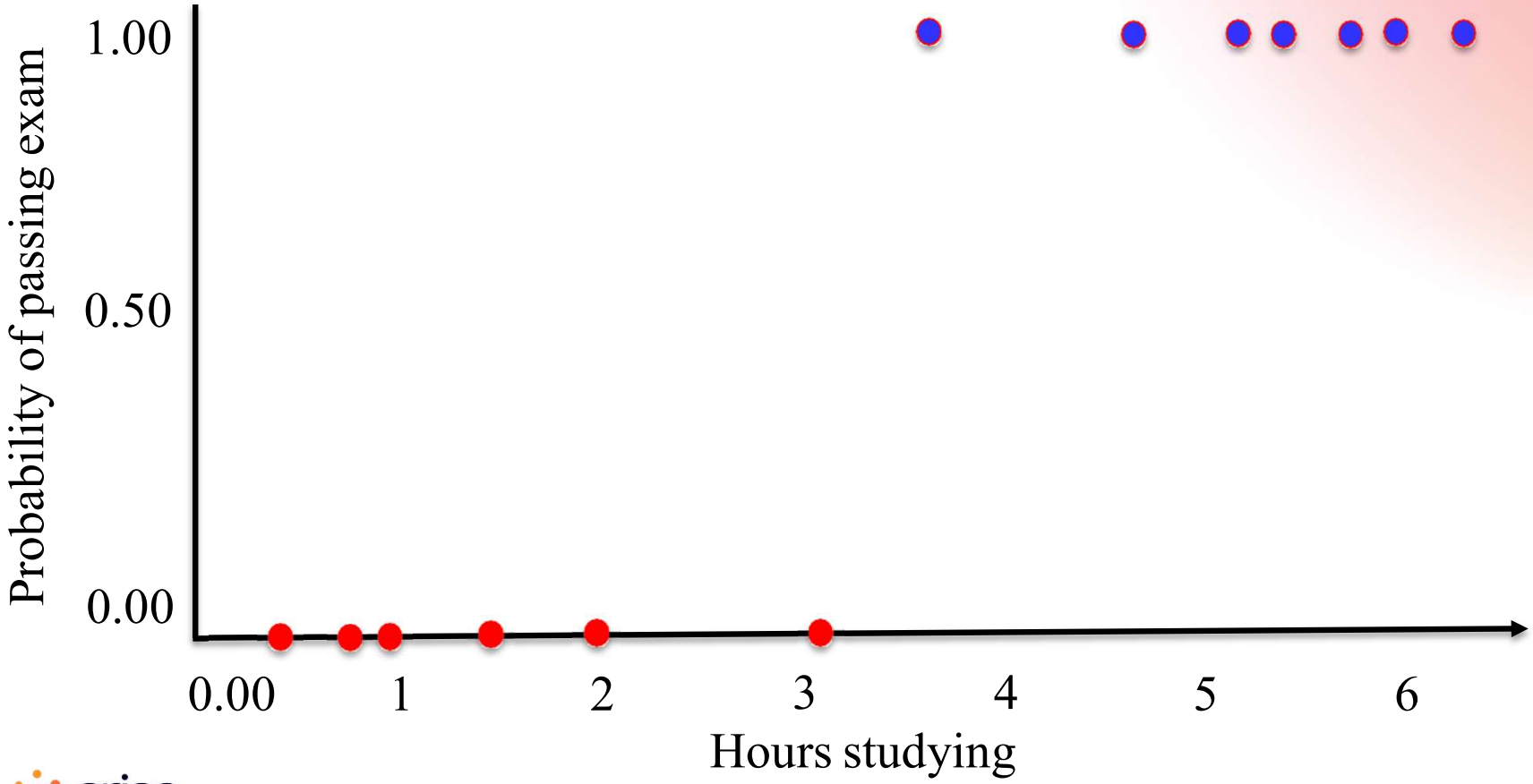




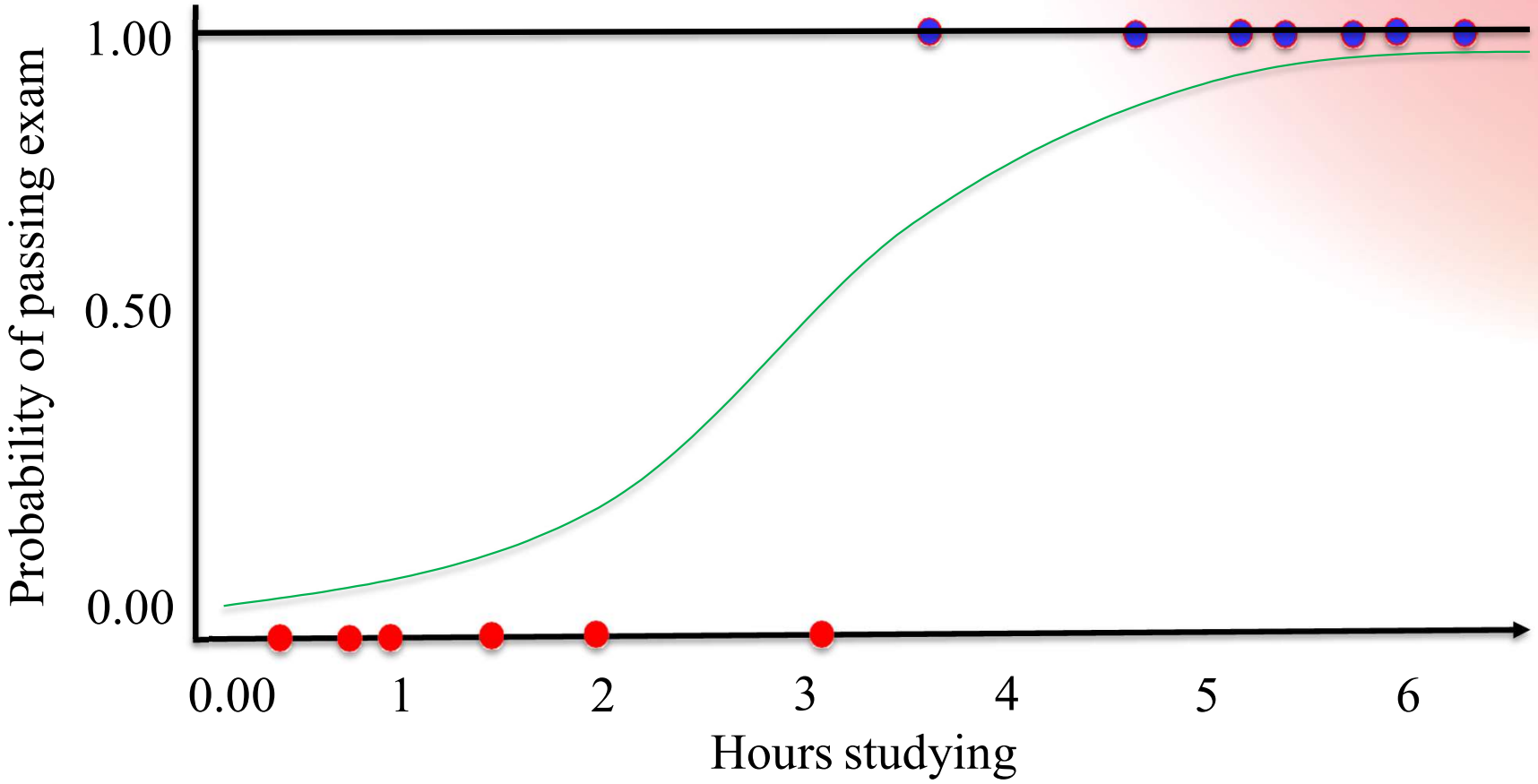
The logistic function transforms real-valued input to an output number y between 0 and 1, interpreted as the probability the input object belongs to the positive class, given its input features $(x_0, x_1, \ldots, x_n)$

$$\hat{y} = \text{logistic}(\hat{b} + \hat{w}_1 \cdot x_1 + \cdots \hat{w}_n \cdot x_n)$$
$$= \frac{1}{1 + \exp\left[-(\hat{b} + \hat{w}_1 \cdot x_1 + \cdots \hat{w}_n \cdot x_n)\right]}$$

Regresión logística

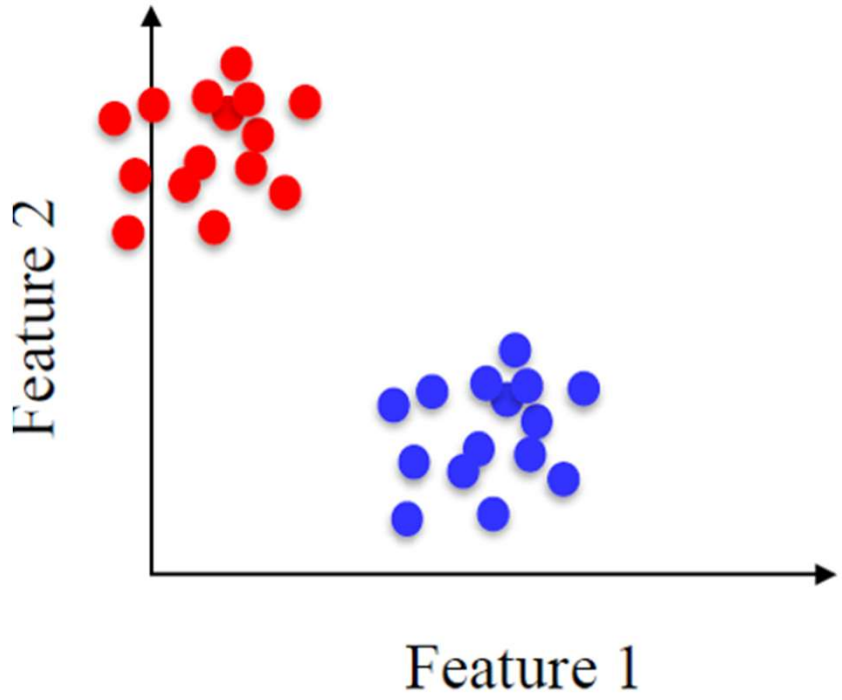


Regresión logística



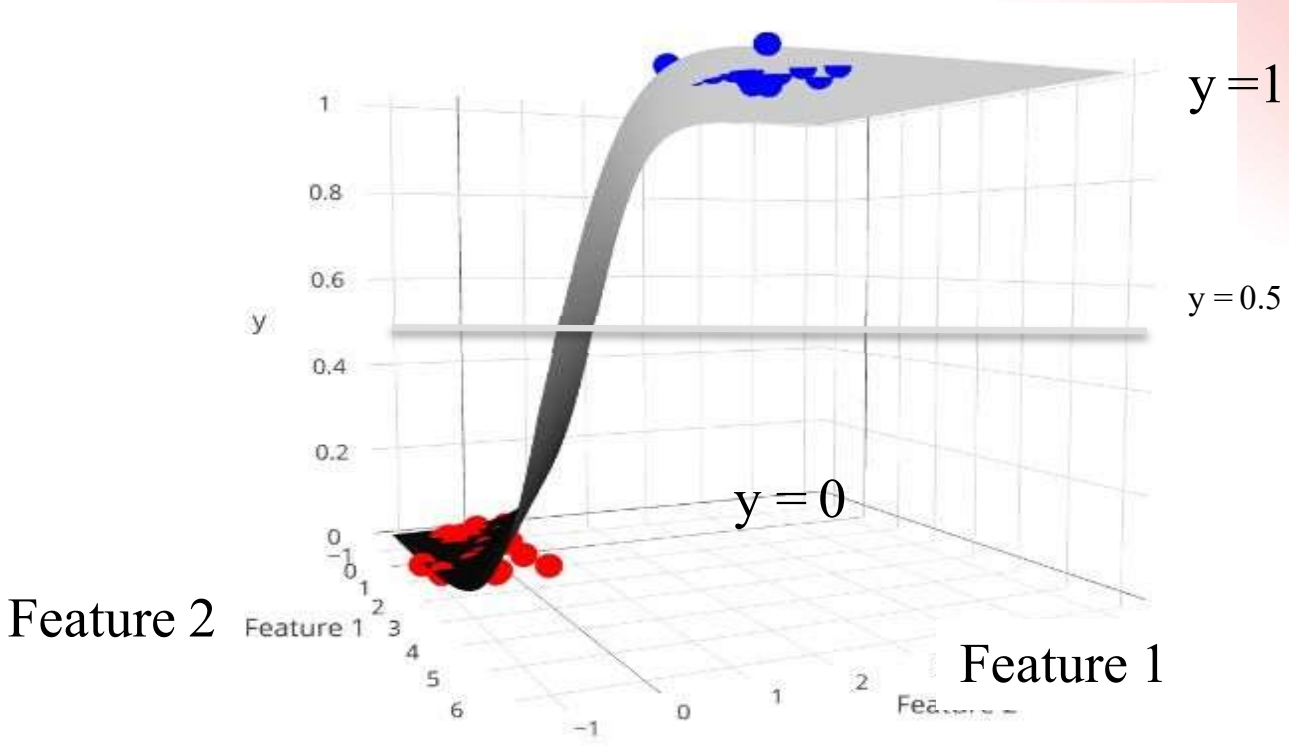
arisa

Regresión logística para la clasificación binaria

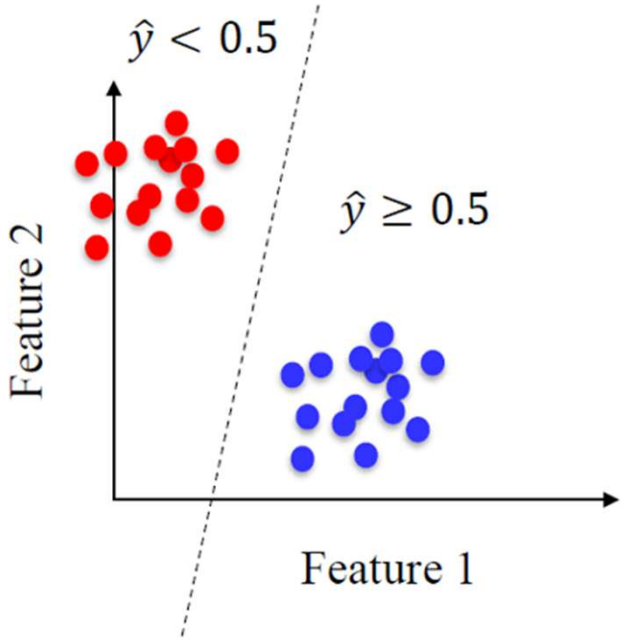


arisa

Regresión logística para la clasificación binaria

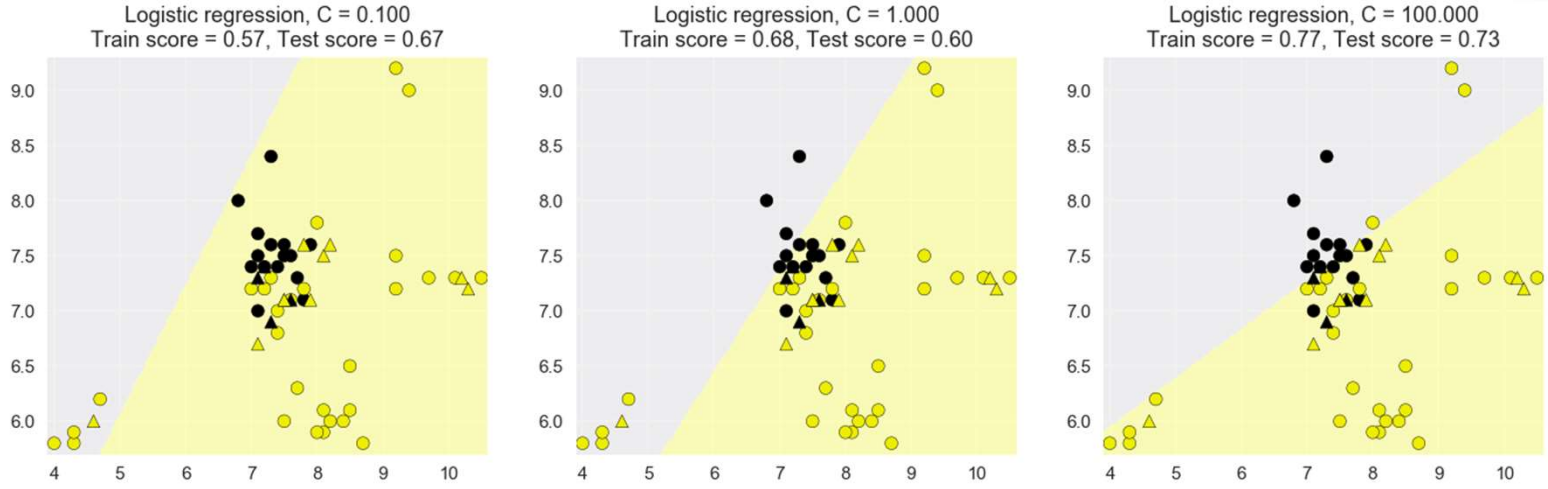


Regresión logística para la clasificación binaria



Regresión logística: Regularización

- La regularización L2 está 'activada' de forma predeterminada (como la regresión de cresta)
- El parámetro C controla la cantidad de regularización (por defecto 1.0)
- Al igual que con la regresión lineal regularizada, puede ser importante normalizar todas las entidades para que estén en la misma escala.



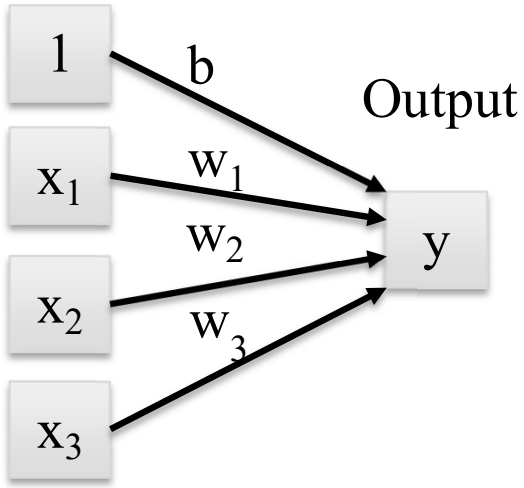
Regresión logística: Regularización

- C pequeña (regularización fuerte): más regularización - modelo más simple - mayor sesgo, menor varianza - mejor para evitar el sobreajuste
- C grande (regularización débil): - Menos regularización - Modelo más complejo - Menor sesgo, mayor varianza - Puede conducir a un sobreajuste
- Cuándo ajustar C:
  - Conjunto de datos pequeño → Considere C más pequeño
  - Conjunto de datos de gran tamaño → Puede usar C más grande

  - Consejos prácticos:
  - - Usar la búsqueda de cuadrícula o la búsqueda aleatoria para el ajuste
  - Rango común: 0,001 a 1000 -

Linear regression

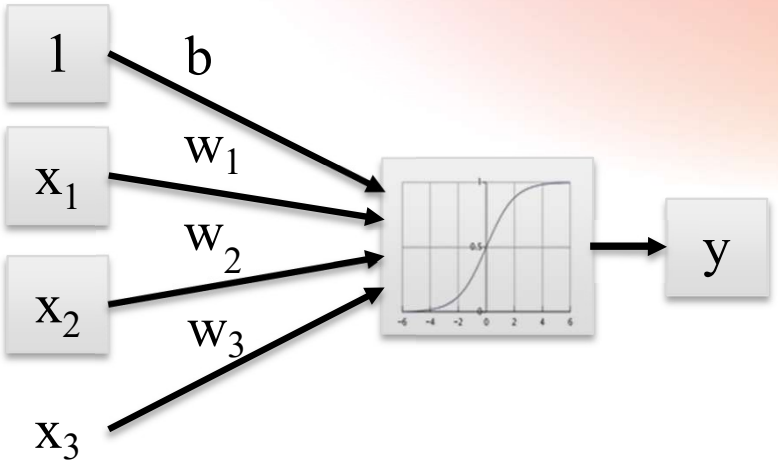
Input features



$$\hat{y} = \hat{b} + \hat{w}_1 \cdot x_1 + \cdots \hat{w}_n \cdot x_n$$

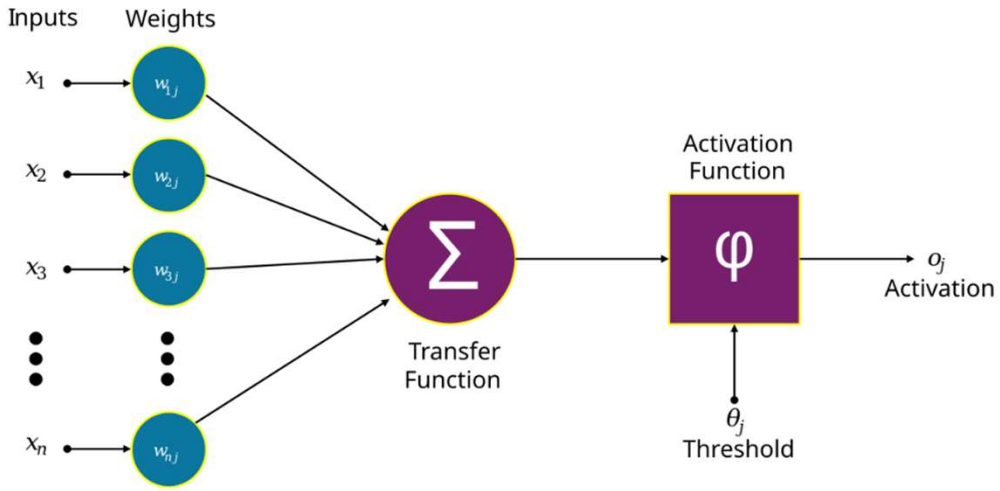
Logistic regression

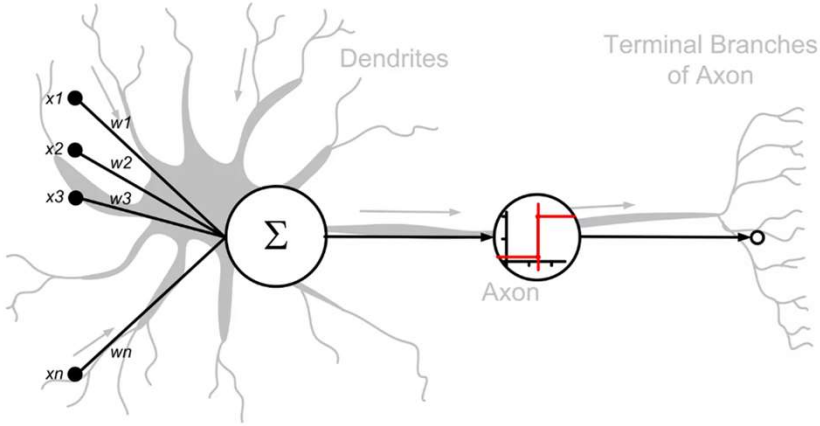
Input features



$$\hat{y} = \text{logistic}(\hat{b} + \hat{w}_1 \cdot x_1 + \cdots \hat{w}_n \cdot x_n)$$

Neurona Artificial (ANN) y Redes Neuronales



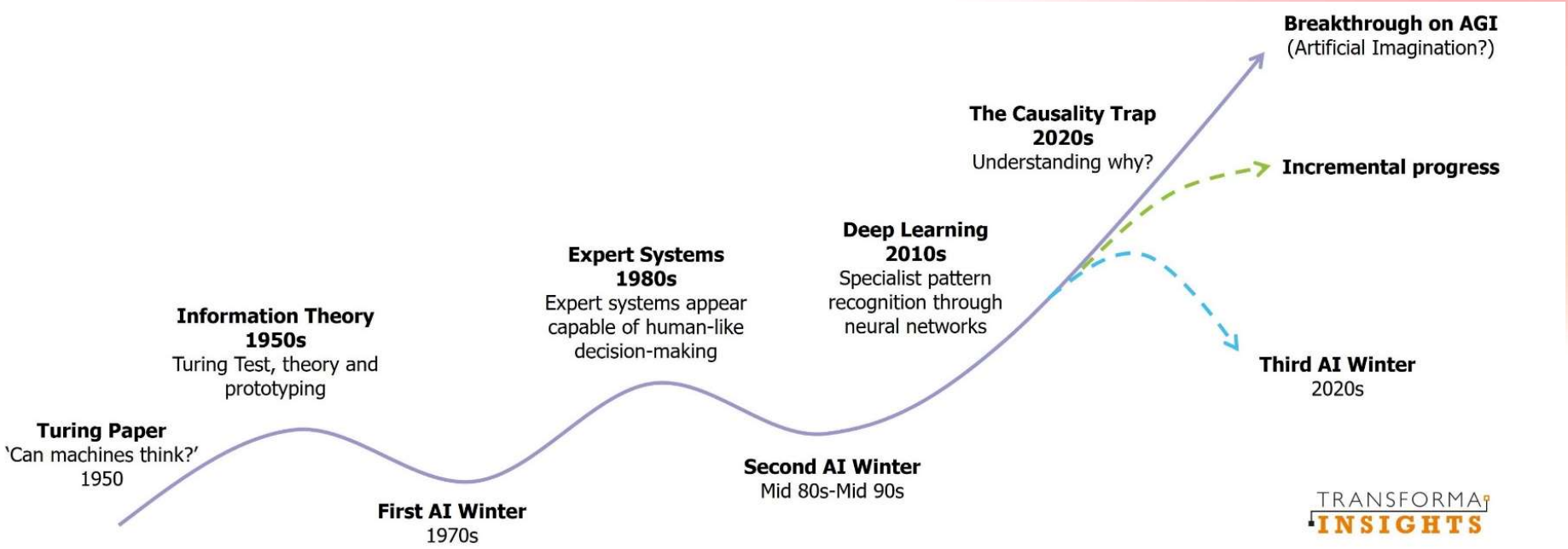


arisa

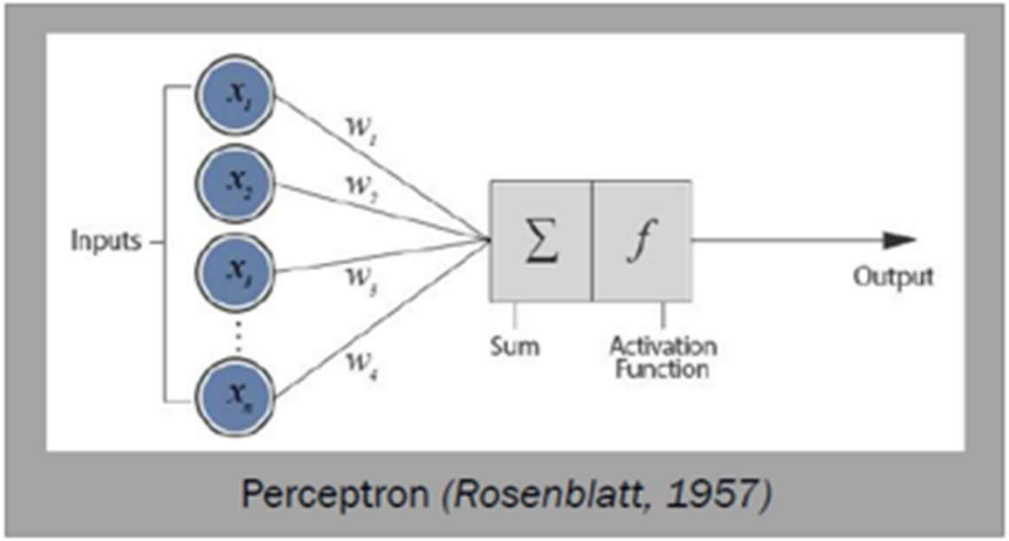
Neurona de McCullocg-Pitts

- 1943: Neurona de McCullocg-Pitts.
- Uno era neurocientífico y el otro matemático.
- "*El cerebro es un solucionador de problemas, así que copiemos al cerebro*".
- Acabó influyendo el diseño de puertas lógicas.
- 1957: Perceptron de Rosenblatt (clasificadore lineal)
- SI QUIERES SABER MÁS: https://es.wikipedia.org/wiki/Neurona_de_McCulloch-Pitts, https://en.wikipedia.org/wiki/Perceptron

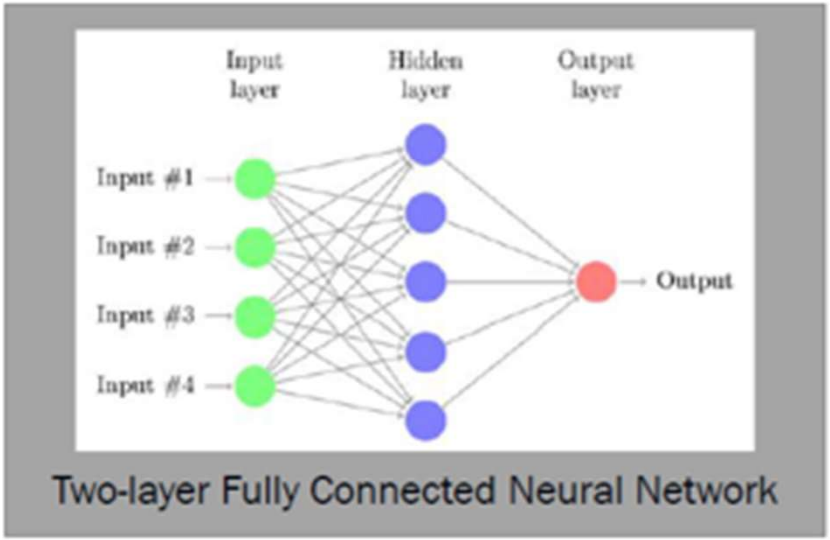
Inviernos de la IA



arisa

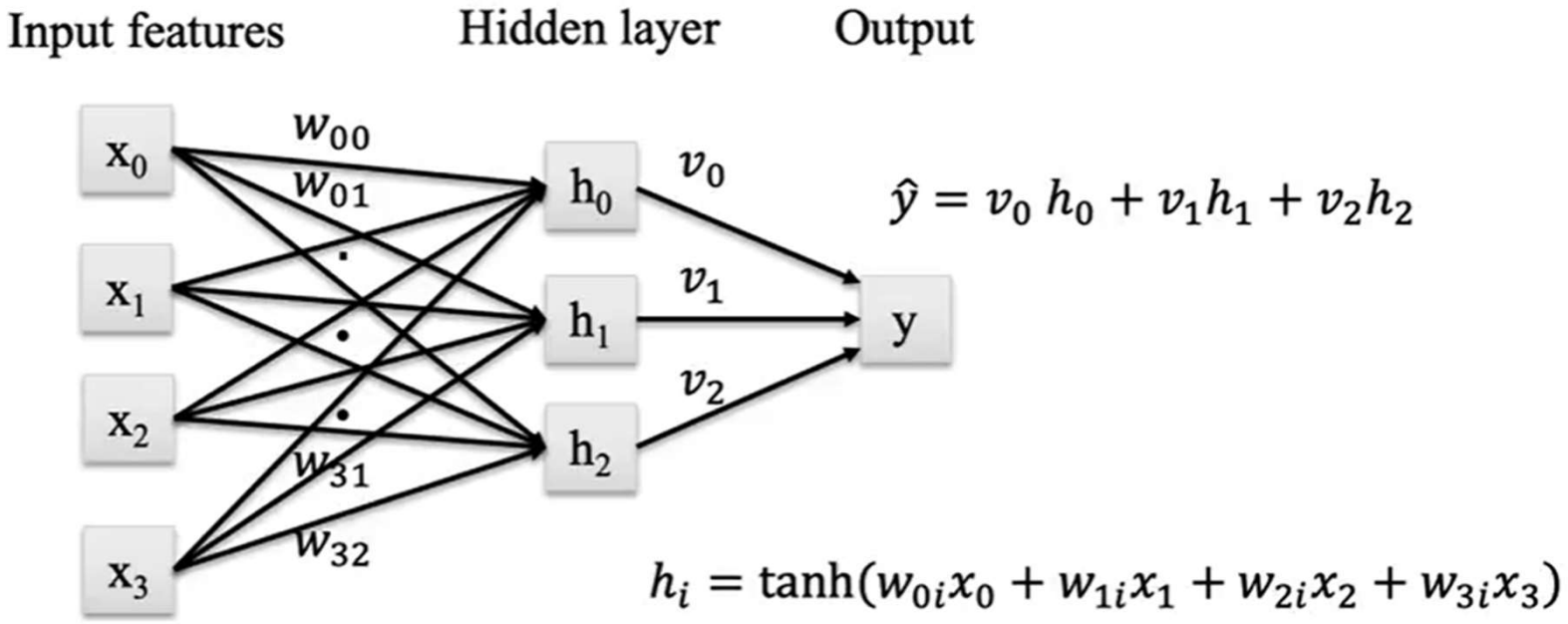


Perceptron *(Rosenblatt, 1957)*

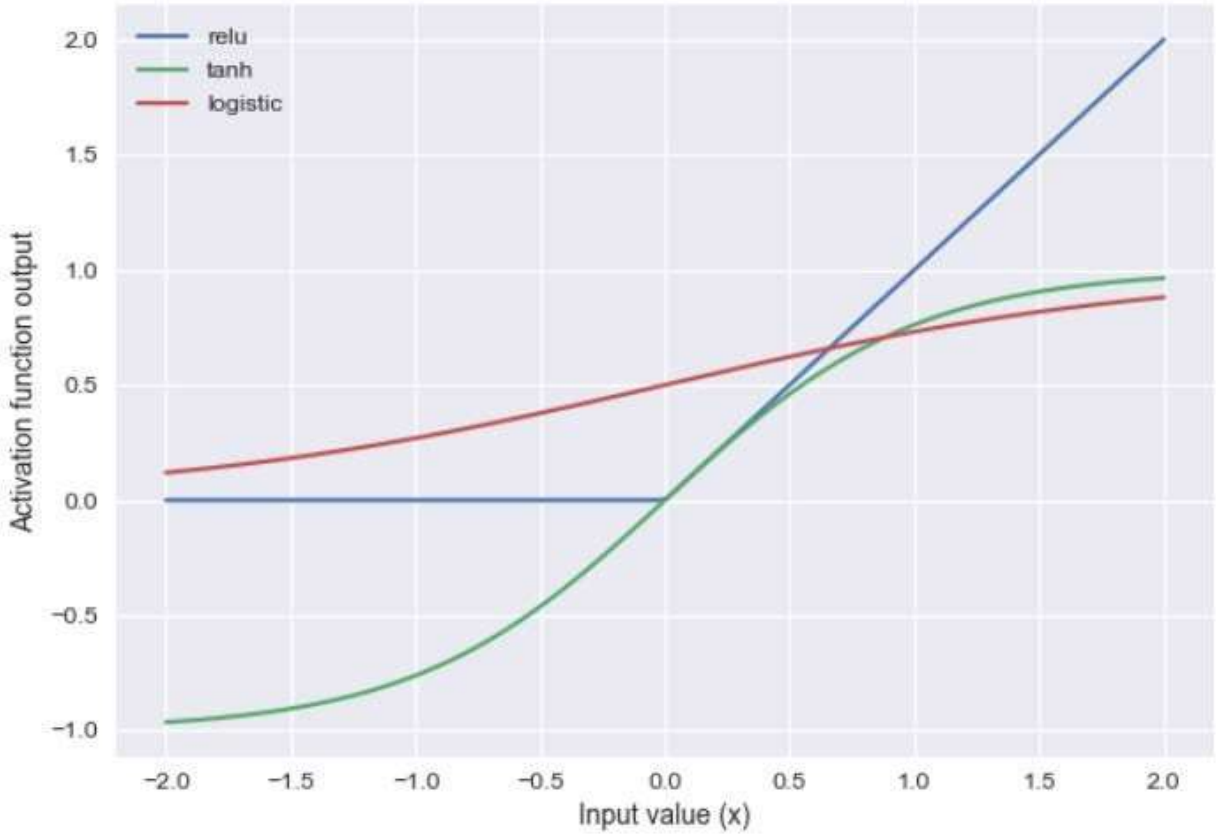


Two-layer Fully Connected Neural Network

MLP de una capa (con activación de función tanh)



Activation Functions

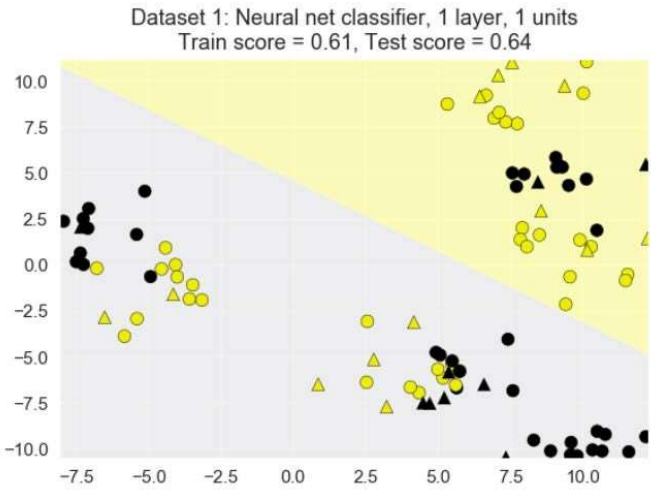


$$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$$

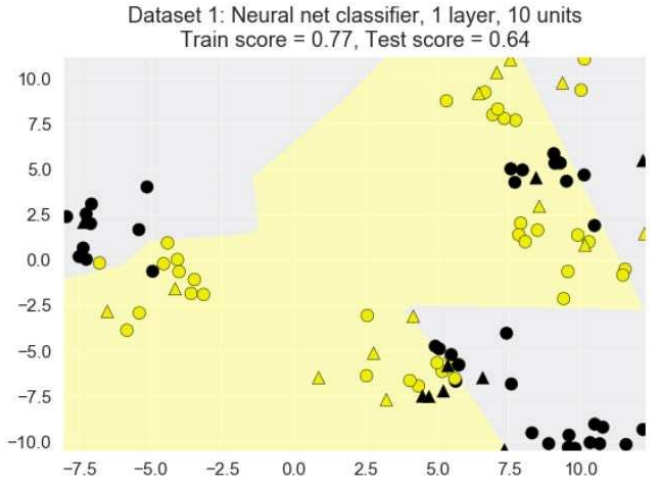
MLP

```
|: from sklearn.neural_network import MLPClassifier
   from adspy_shared_utilities import plot_class_regions_for_classifier_subplot

   X_train, X_test, y_train, y_test = train_test_split(X_D2, y_D2,
                                                       random_state=0)

   fig, subaxes = plt.subplots(3, 1, figsize=(6,18))

   for units, axis in zip([1, 10, 100], subaxes):
       nnclf = MLPClassifier(hidden_layer_sizes = [units], solver='lbfgs',
                             random_state = 0).fit(X_train, y_train)

       title = 'Dataset 1: Neural net classifier, 1 layer, {} \
   units'.format(units)

       plot_class_regions_for_classifier_subplot(nnclf, X_train, y_train,
                                                 X_test, y_test, title, axis)
       plt.tight_layout()
```

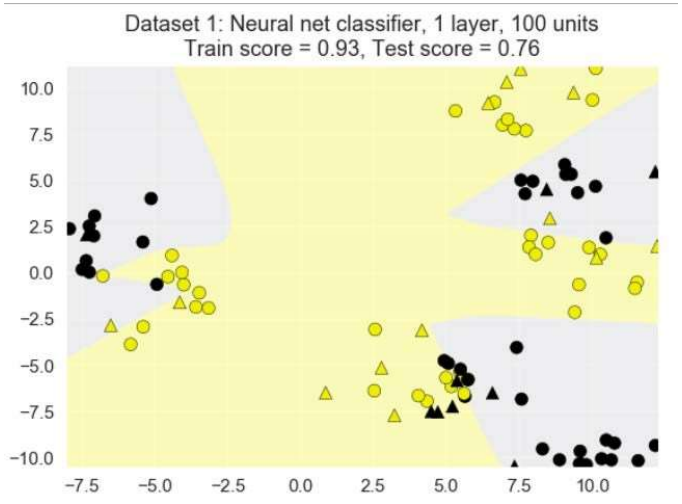
1 capa interna de 1, 10, or 100 units



1 layer, 1 unit

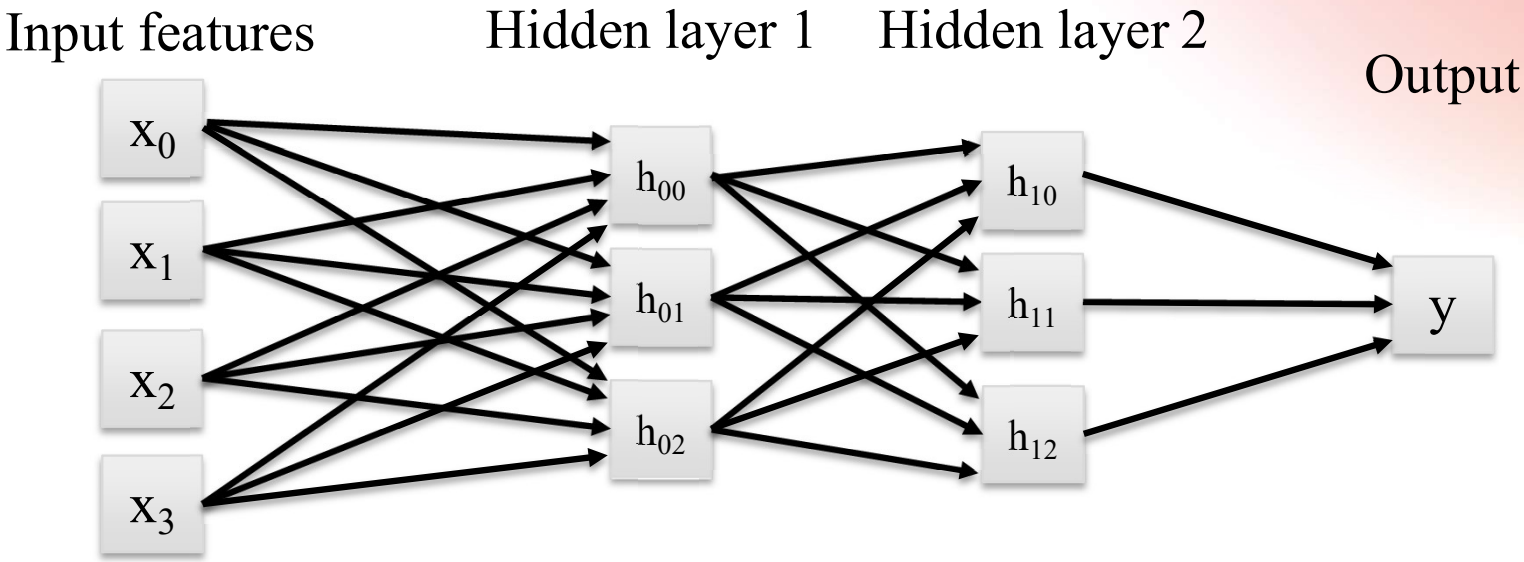


1 layer, 10 units

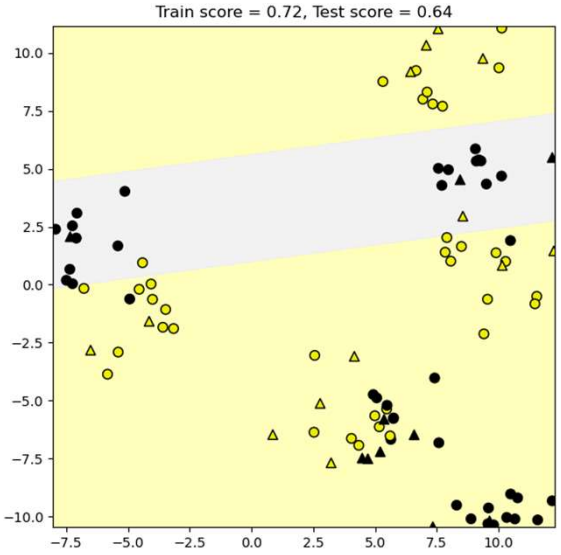


1 layer, 100 units

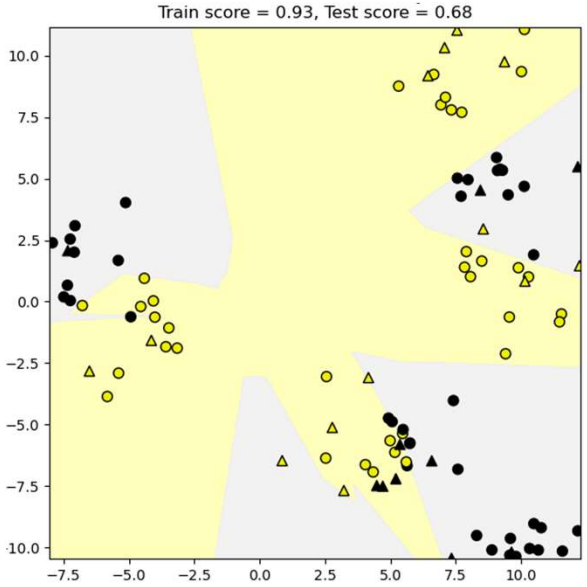
MLP 2 Capas


Input features
Hidden layer 1
Hidden layer 2
Output
x_0
x_1
x_2
x_3
h_{00}
h_{01}
h_{02}
h_{10}
h_{11}
h_{12}
y

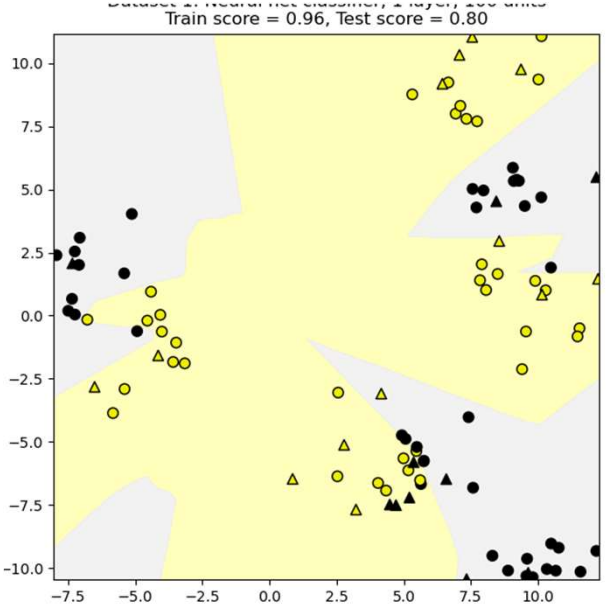
2 capa interna de 1, 10, or 100 units



2 layer, 1-10 unit

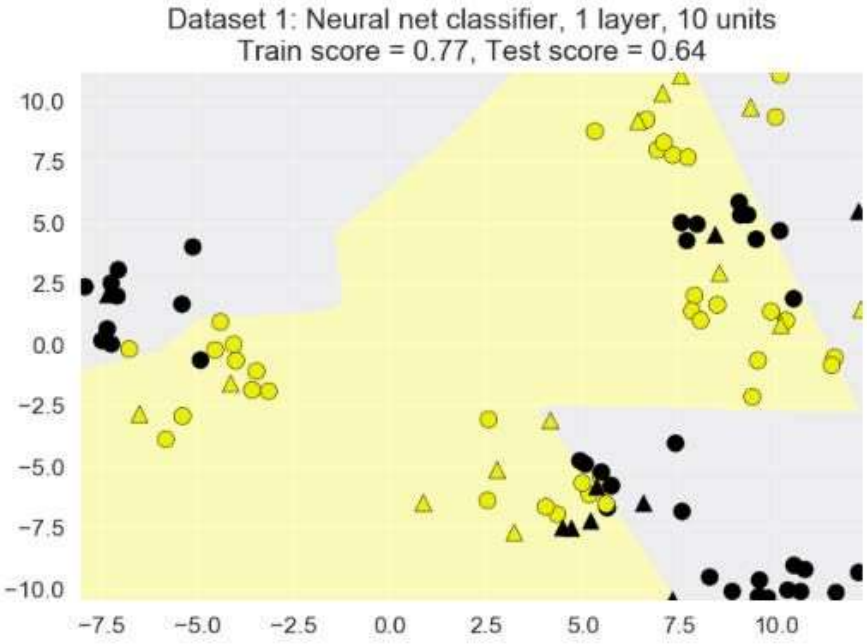


2 layer, 10-10 units

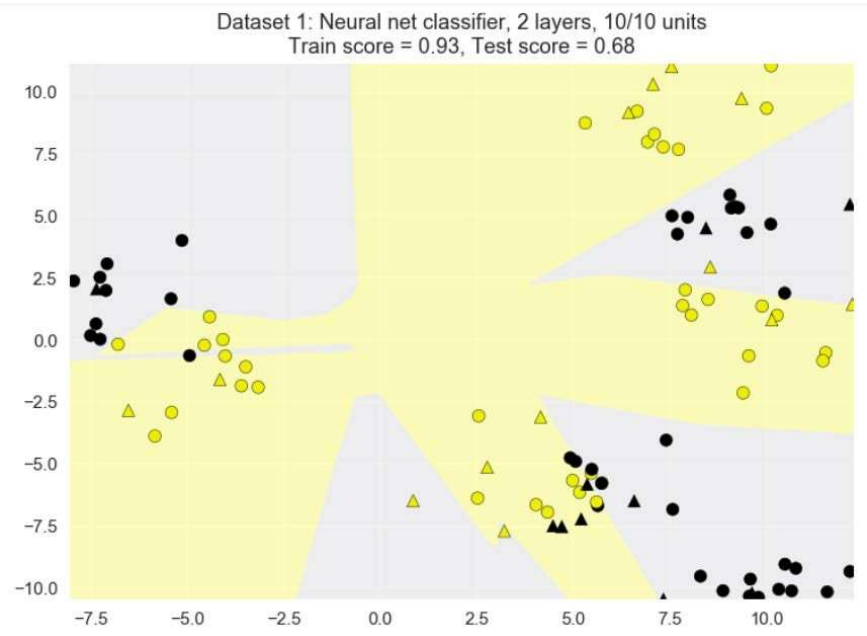


2 layer, 100-10 units

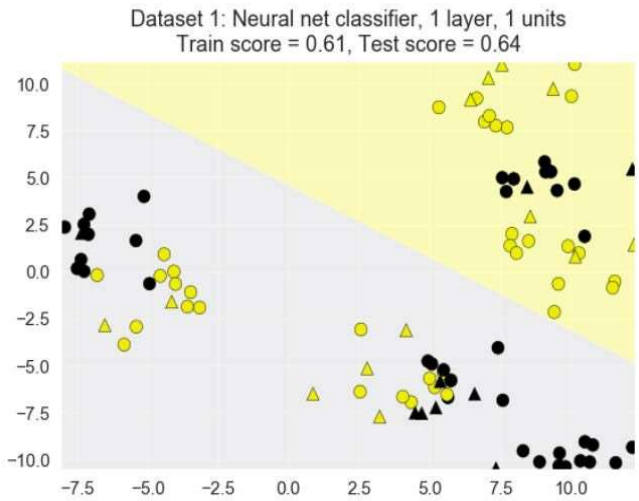
1vs 2 capas

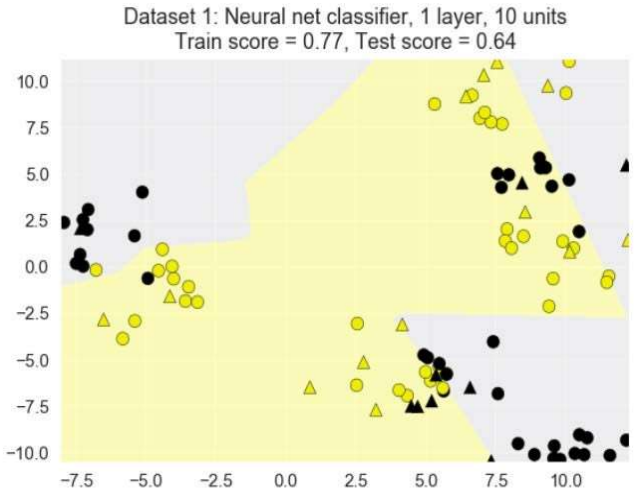


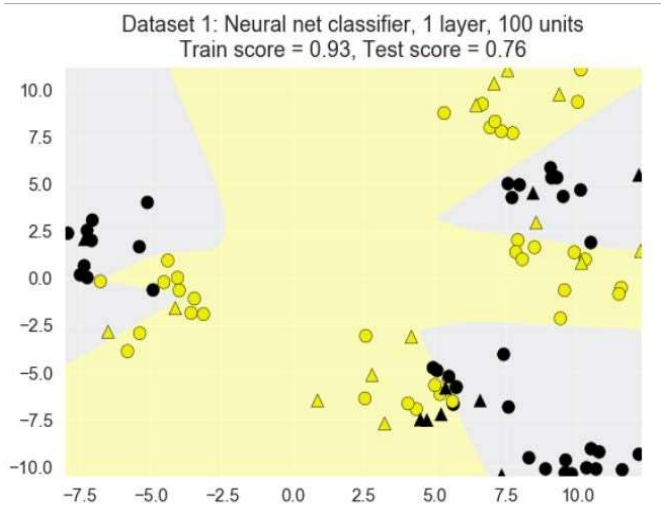
1 layer, 10 units

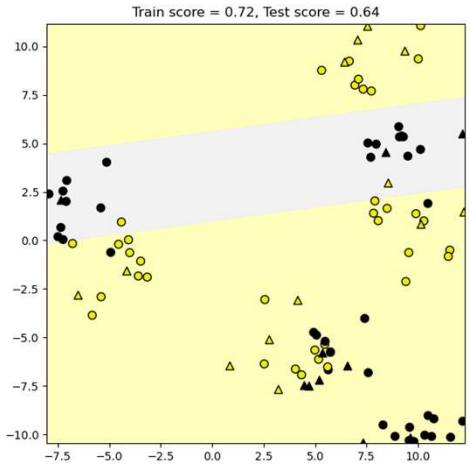


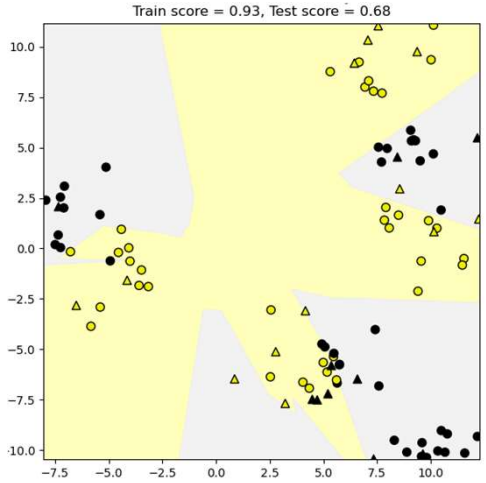
2 layers, (10, 10) units

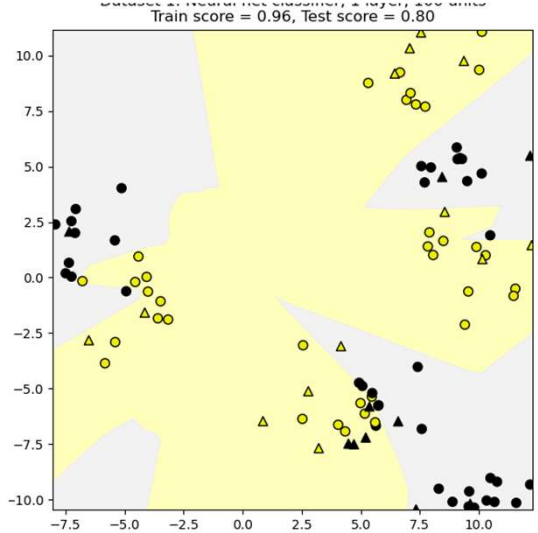












2 layer, 1-10 unit

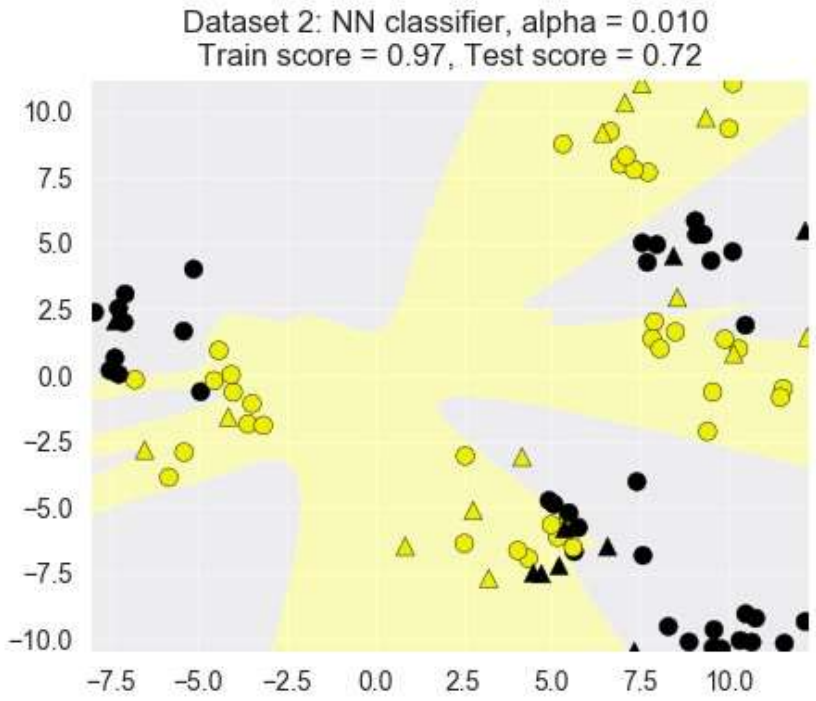
2 layer, 10-10 units

2 layer, 100-10 units

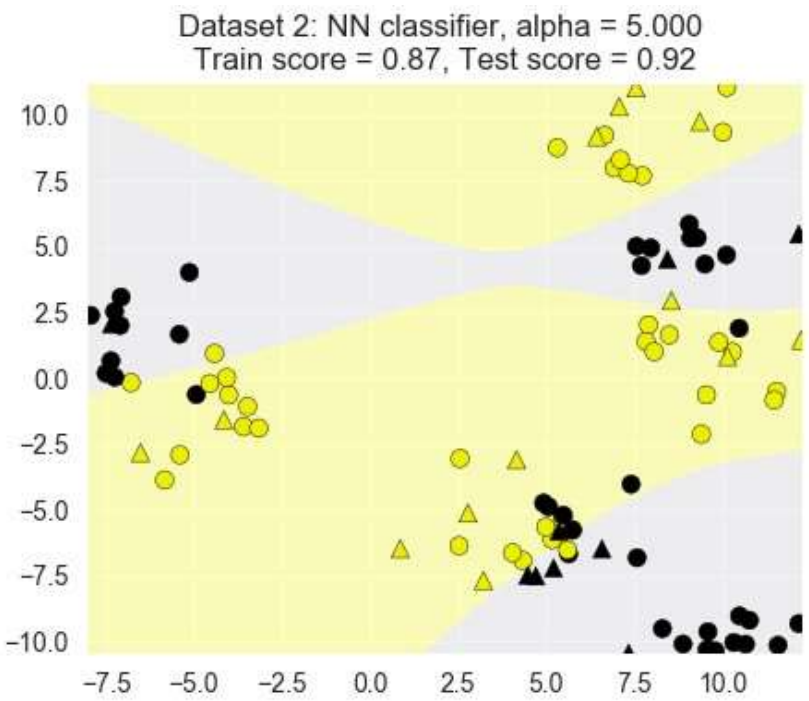
Como controlar la complejidad de NLP multicapas

- Al aumentar las capas y las neuronas aumenta la complejidad al tener que computer mas pesos
- EL modelo puede hacerse muy Complejo muy rapido por eso es muy relevante controlar la complejidad mediante regularizacion
- Al igual que en los modelos lineales la regularizacion es un peso sobre los pesos – cuando mas grande la regularizacion (alpha) mas grade es la corrección y los modelos que se generan son mas sencillos evitando por tanto el overfitting
- Normalizacion es clave para NN debido a que se basan en metricas de distancia y tiene un gran impacto para el resultado final

L2 Regularization



alpha = 0.01

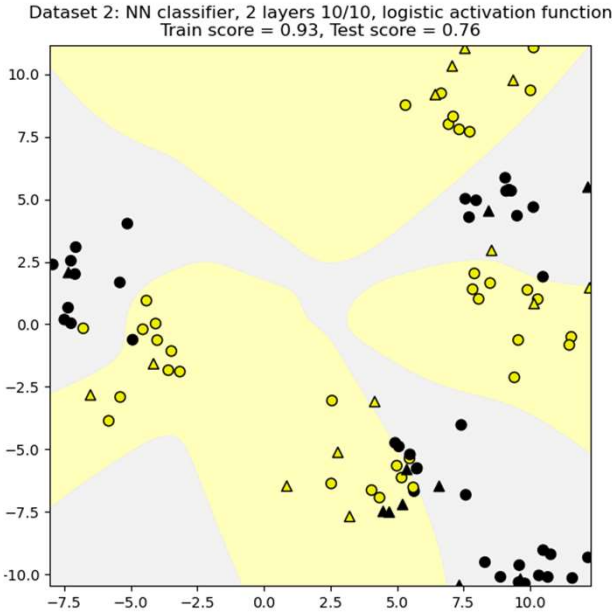


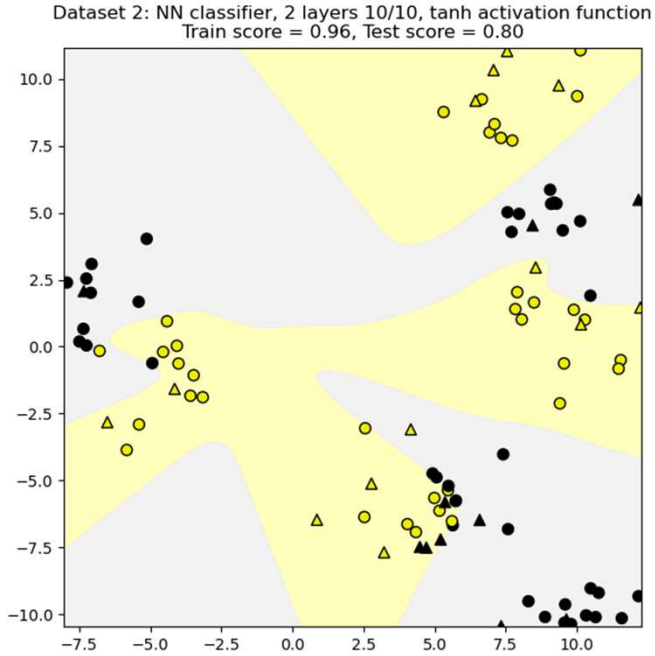
alpha = 5.0

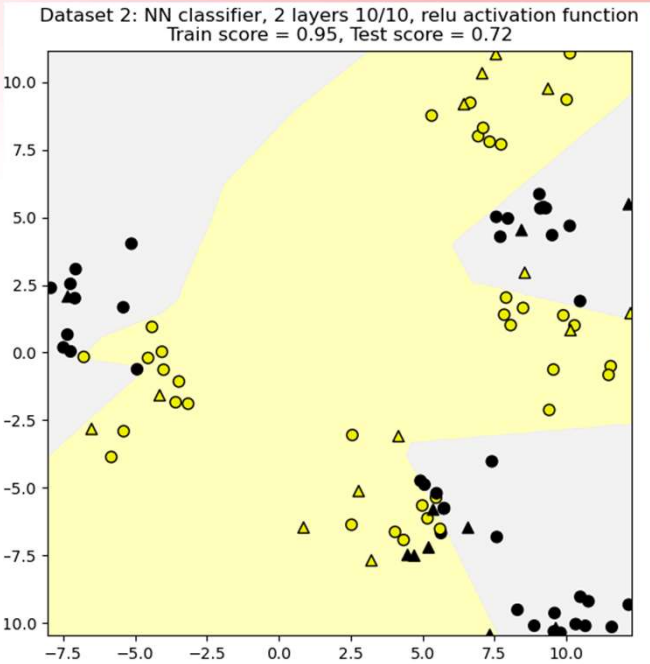
Relevancia de las funciones de activacion

```
X_train, X_test, y_train, y_test = train_test_split(X_D2, y_D2, random_state=0)

fig, subaxes = plt.subplots(3, 1, figsize=(6,18))

for this_activation, axis in zip(['logistic', 'tanh', 'relu'], subaxes):
    nnclf = MLPClassifier(solver='lbfgs', activation = this_activation,
                          alpha = 0.1, hidden_layer_sizes = [10, 10],
                          random_state = 0).fit(X_train, y_train)

    title = 'Dataset 2: NN classifier, 2 layers 10/10, {} \
activation function'.format(this_activation)

    plot_class_regions_for_classifier_subplot(nnclf, X_train, y_train,
                                              X_test, y_test, title, axis)
    plt.tight_layout()
```

Relevancia de las funciones de activacion







Importancia de la normalización para NN

```
from sklearn.neural_network import MLPClassifier
from sklearn.preprocessing import MinMaxScaler


scaler = MinMaxScaler()

X_train, X_test, y_train, y_test = train_test_split(X_cancer, y_cancer,
                                                    random_state = 0)
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

clf = MLPClassifier(hidden_layer_sizes = [100, 100], alpha = 5.0,
                    random_state = 0, solver='lbfgs').fit(X_train_scaled,
                                                          y_train)
```

Relevancia

```
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import train_test_split

# Split dataset into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X_cancer, y_cancer, random_state=0)

# Define and train a Multi-Layer Perceptron classifier WITHOUT normalization
clf_no_scaling = MLPClassifier(hidden_layer_sizes=[100, 100], alpha=5.0,
                               random_state=0, solver='lbfgs').fit(X_train, y_train)

# Print accuracy
print('Breast cancer dataset (No Normalization)')
print('Accuracy of NN classifier on training set: {:.2f}'.format(clf_no_scaling.score(X_train, y_train)))
print('Accuracy of NN classifier on test set: {:.2f}'.format(clf_no_scaling.score(X_test, y_test)))
```

```
Breast cancer dataset (No Normalization)
Accuracy of NN classifier on training set: 0.94
Accuracy of NN classifier on test set: 0.94
```

```
Breast cancer dataset
Accuracy of NN classifier on training set: 0.98
Accuracy of NN classifier on test set: 0.97
```

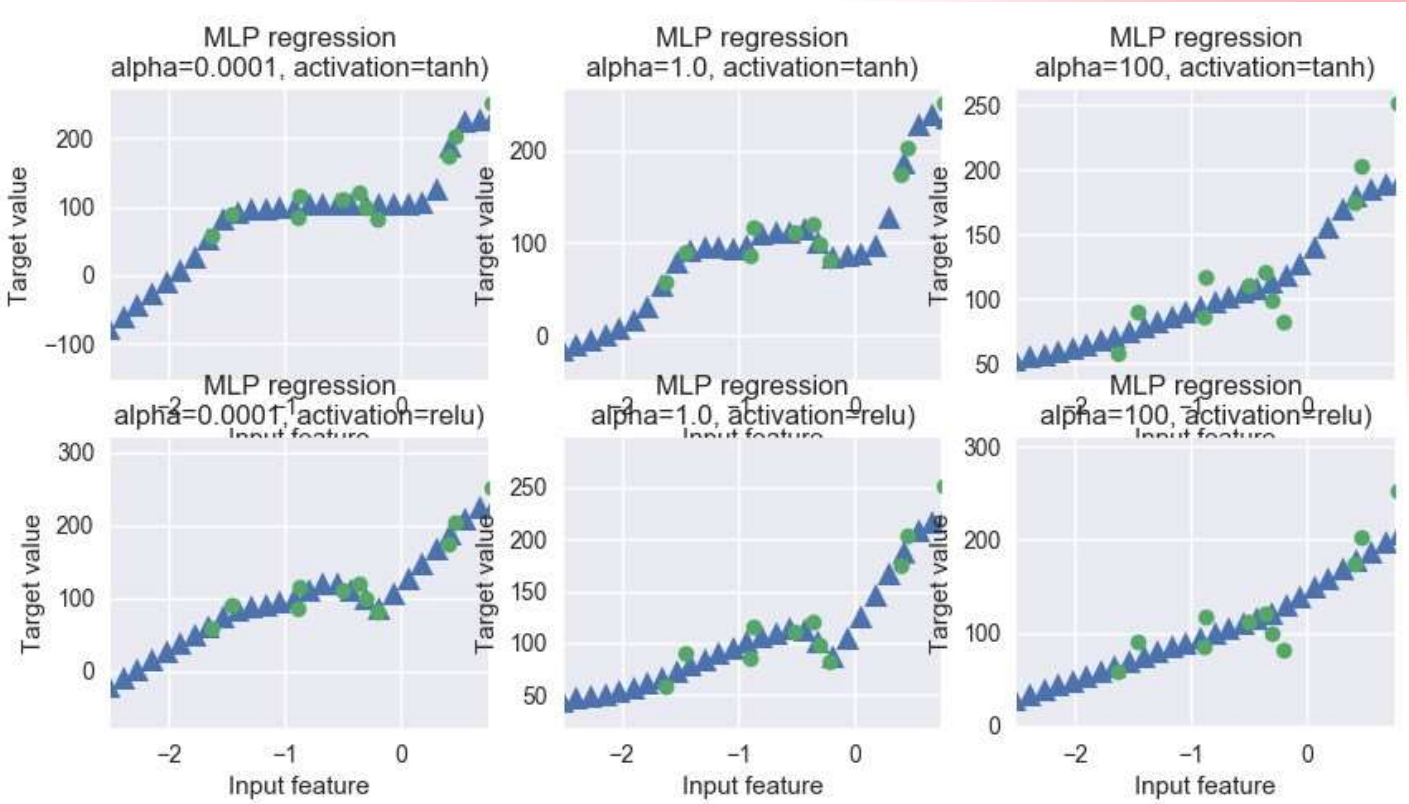
Regresión con MLP y Regularizacion (para bajar los pesos)

```
from sklearn.neural_network import MLPRegressor

fig, subaxes = plt.subplots(2, 3, figsize=(11,8), dpi=70)

X_predict_input = np.linspace(-3, 3, 50).reshape(-1,1)

X_train, X_test, y_train, y_test = train_test_split(X_R1[0::5], y_R1[0::5], random_state = 0)

for thisaxisrow, thisactivation in zip(subaxes, ['tanh', 'relu']):
    for thisalpha, thisaxis in zip([0.0001, 1.0, 100], thisaxisrow):
        mlpreg = MLPRegressor(hidden_layer_sizes = [100,100],
                              activation = thisactivation,
                              alpha = thisalpha,
                              solver = 'lbfgs').fit(X_train, y_train)
        y_predict_output = mlpreg.predict(X_predict_input)
```

Regresión con NN con MLPRegressor

activation = 'tanh'

activation = 'relu'



alpha = 0.0001 alpha = 1.0 alpha = 100.0

Increasing regularization

Parámetros

- **`hidden_layer_sizes`: sets the number of hidden layers (number of elements in list), and number of hidden units per layer (each list element).** *Default: (100).*

- **`alpha`: controls weight on the regularization penalty that shrinks weights to zero.** *Default: alpha = 0.0001.*

- **`activation`: controls the nonlinear function used for the activation function, including: `'relu'` (default), `'logistic'`, `'tanh'`.**

- https://playground.tensorflow.org/

arisa

#AIskills4all | @AIskillsEU | linkedin.com/AIskillsEU

Co-funded by the European Union

www.aiskills.eu

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.