Machine Learning and applications in AI

Martin Palazzo

Universite de Technologie de Troyes
Universidad Tecnologica Nacional Buenos Aires

mpalazzo@frba.utn.edu.ar

December 5, 2022

Overview

Supervised Learning: Train, Validation and Test sets

Train, Validation and Test sets

Given a data set $S = (x_1, y_1)...(x_n, y_n)$ we never will fit a classifier using the full set.

Train set

The set of samples $X_{tr} \in S$ used to build the model (classifier, regression function).

Validation set

The set of samples $X_{val} \in S$ used to measure the prediction performance of the classifier. Once it is validated, train and validation sets are merged to define a final model.

Test set

The independent set of samples $X_{te} \in S$ used to measure the prediction performance of the final classifier. These samples are never used during the training and fit process.

Cross Validation

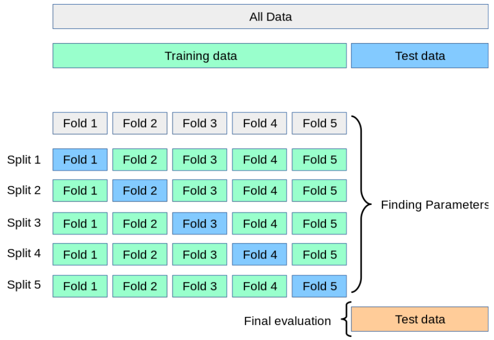


Figure: Cross validation process

5 fold cross validation

For cross validation first we can split the full dataset in 80% for train and 20% for test. Then make CV on training set.

Train, Validation and Test sets

Exercise 03

Given the breast cancer wisconsin dataset divide the data into train and test sets. Then using the training set train a KNN model by 5 fold cross validation. Compute the accuracy and empirical error of each fold. Select the best parameter and re-train using the full training set. Finally classify the rest of the test labels using the trained model.

Cost functions

Cost functions

We can measure the performance of our model by computing the Empirical Error defined as

$$J_{emp} = \frac{1}{n} \sum_{i=1}^{n} Q(d(x_i, w), y_i)$$

The error to minimize can be designed in many ways. There are two alternatives (among many)

$$Q(x, y) = (y - d(x, w))^2 \quad \text{The quadratic cost}$$

$$Q(x, y) = |y - d(x, w)| \quad \text{The absolute cost}$$

Statistical Learning as an optimization approach

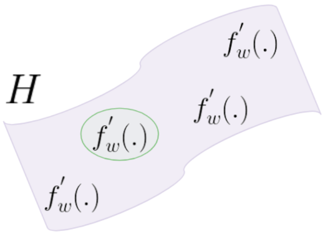
Loss function optimization

$$\min_{w} L(y, \hat{y}) = \min_{w} L(y, \hat{f}(x))$$

The optimization problem is structured as the Loss function as objective function to be minimized and the function parameters w as the decision variables.

Classification with hyper-planes

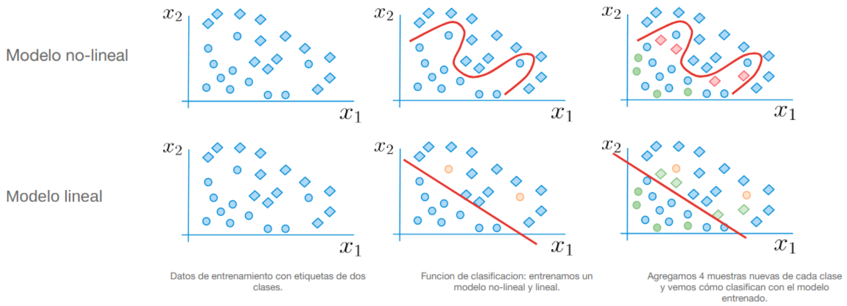
Hypothesis space



$$H = \left\{ f_w^1(.), f_w^2(.), \dots, f_w^{'}(.), \dots \right\}$$

Figure: Space of possible funtion defined by the parameters *w*

Complexity of the decision boundary



Modelo no-lineal
Modelo lineal
x_1
x_2
Datos de entrenamiento con etiquetas de dos clases.
Funcion de clasificacion: entrenamos un modelo no-lineal y lineal.
Agregamos 4 muestras nuevas de cada clase y vemos cómo clasifican con el modelo entrenado.

Figure: The complexity of the decision function may change the solution

Linear Classifier: hyperplanes

$$f(x) = b + w^T x = 0$$

$$D(x) = \text{sign}[w^T x + b]$$

There are many types of decision functions. The best-known family of functions is the linear ones. These functions are hyper-planes characterized by parameters w (vector w = [w1... .wd]) that will determine how the decision boundary is positioned in the hyper-space of dimension d. In binary classification, the decision function will assign a value of y = 1 or y = -1 depending on which side of the hyperplane the x samples are positioned.

Linear Classifier: toy exercise

$$\mathbf{x}_{train} = \begin{bmatrix} 1 & 1 \\ 3 & 0.5 \\ 2 & 3 \\ 2.5 & 1.5 \\ 1.5 & 2 \\ 2.5 & 4 \\ 3.5 & 1.5 \\ 4 & 3 \end{bmatrix} \quad \mathbf{y}_{train} = \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Given the following dataset compute manually the values of **w** and **b** that minimize the empirical error.

Perceptron

Perceptron classifier

The Perceptron algorithm is a linear classifier proposed in the 60s. Despite is not the best option to use nowadays, it represents the building block of neural networks and also the first successful linear discriminator in the modern history of pattern recognition.

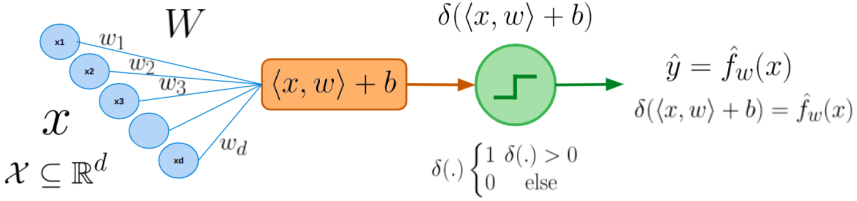
$$y(x) = f(w^T x)$$

Perceptron is designed for binary classification and the idea is to output the following considering class 1 $C_1 = 1$ and class 2 $C_2 = -1$

$$f(x^T \phi(x)) = \begin{cases} +1 & \text{if } (w^T x) \geq 0 \\ -1 & \text{if } (w^T x) < 0 \end{cases}$$

So finally we want a function that $(w^T x)y > 0$

Perceptron classifier



W
w_1
w_2
w_3
w_d
x1
x2
x3
xd
x
$\mathcal{X} \subseteq \mathbb{R}^d$
$\langle x, w \rangle + b$
$\delta(\langle x, w \rangle + b)$
$\delta(.) \begin{cases} 1 & \delta(.) > 0 \\ 0 & \text{else} \end{cases}$
$\hat{y} = \hat{f}_w(x)$
$\delta(\langle x, w \rangle + b) = \hat{f}_w(x)$

Figure: The perceptron pipeline

The perceptron model

The model get d-dimensional input vectors where each input variable is linearly combined to a weight vector w. After the linear combination $\langle x, w \rangle$ an activation function is used to discretize the output value.

Perceptron classifier

Perceptron training

The perceptron model start with random weight initialization. The idea is to make iterations while computing the classification error. At each iteration if the error does not decreases, then the weights are updated by gradient descent.

Perceptron optimization

At each iteration, the error should decrease until convergence. This case can only occur if classes are completely separable. In case of not separable classes then the Perceptron optimization does not never converge to cero error.

Perceptron classifier

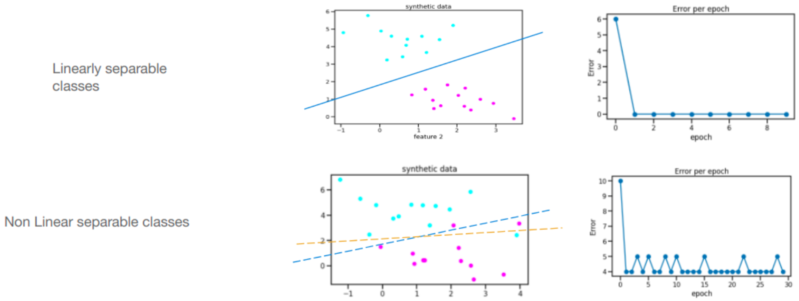
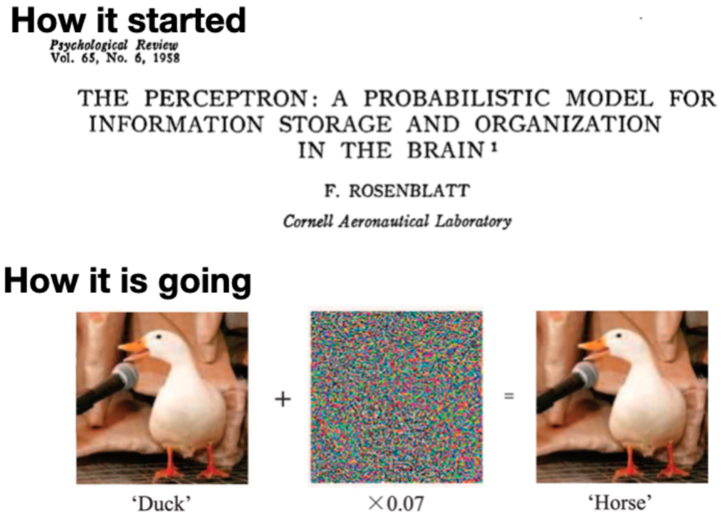


Figure: Perceptron architecture

Some limitations of the Perceptron lies in it can not converge when classes are not separable. In addition, for separable classes many solutions can occur. It has not any margin optimization!!

Perceptron classifier: toy exercise


How it started

Psychological Review
Vol. 65, No. 6, 1958

THE PERCEPTRON: A PROBABILISTIC MODEL FOR INFORMATION STORAGE AND ORGANIZATION IN THE BRAIN[1]

F. ROSENBLATT

Cornell Aeronautical Laboratory

How it is going

Figure: Perceptron dream

Perceptron classifier: toy exercise

Exercise 04

Design a two dimensional dataset where the perceptron will never find a solution and classes are completely separable.

Perceptron classifier

Exercise 05

Given the Wisconsin Breast dataset divide the data into train and test sets (equal to exercise 07). Then using the training set train a Perceptron model by 5 fold cross validation. Compute the accuracy and empirical error of each fold. Select the best parameter and re-train using the full training set. Finally classify the rest of the test labels using the trained model. Compare the results with the KNN.

Logistic Regression

Logistic Regression Classifier

The logistic regression classifier is a linear model embedded with a sigmoid function

$$p(y \mid x) = \sigma(w^\top x)$$

where the sigmoid function is

$$\sigma(x) = \frac{1}{1 + \exp(-x)}$$

therefore the final model is

$$\sigma(w^\top x) = \frac{1}{1 + \exp(-w^\top x)}$$

Sigmoid Function

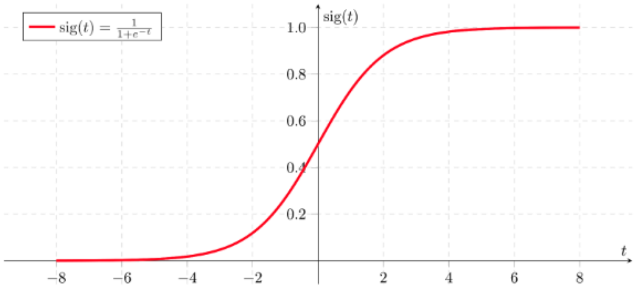


Figure: Sigmoid function

Logistic Regression weight solution

Loss function

The cross entropy loss function is used to find the logistic regression weights

$$L_{CE}(y,\hat{y}) = -[y \log \hat{y} + (1-y) \log (1-\hat{y})]$$

Weight solution to LR

$$\hat{w} = \underset{n}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} L_{\mathrm{CE}} \left(f \left(x^{(i)}; w \right), y^{(i)} \right)$$

Logistic regression loss

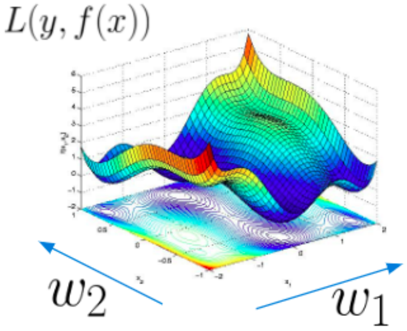


Figure: Non convex optimization problem

Linear Regression

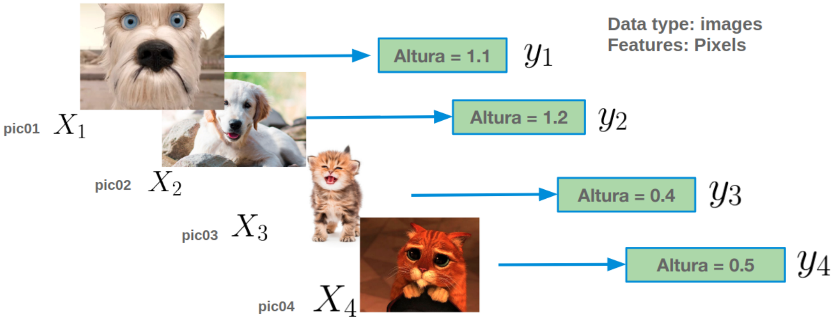


Figure: Regression

Regression

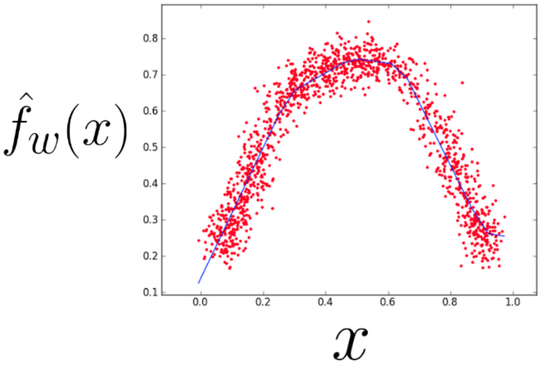


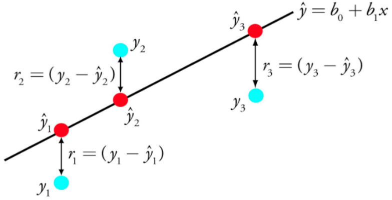
Figure: Regression function

Instead of learning a separating function now a regression function is learned.

Regression

$$e_i = y_i - \hat{y}_i$$

$$\text{RSS} = e_1^2 + e_2^2 + \cdots + e_n^2$$



$\hat{y} = b_0 + b_1 x$
$\hat{y}_3$
y_2
$r_2 = (y_2 - \hat{y}_2)$
$r_3 = (y_3 - \hat{y}_3)$
y_3
$\hat{y}_2$
$\hat{y}_1$
$r_1 = (y_1 - \hat{y}_1)$
y_1

Figure: Residuals

The idea is to minimize the residual sum of squares.

Regression

Regression problem

$$\mathrm{RSS}(w) = \sum_i^n (y_i - f(x_i))^2$$

$$\mathrm{RSS}(w) = \sum_i^n \left(y_i - w_0 - \sum_j^d x_{ij} w_j\right)^2$$

Linear regression

$$\min_w \|Xw - y\|^2$$

The regression problem aims to find a vector w that minimizes the error between the predicted labels and the true ones.

Regression

Linear regression solution: Ordinary Least Squares (OLS)

$$\hat{w} = (\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}^T y$$

The regression problem aims to find a vector w that minimizes the error between the predicted labels and the true ones.

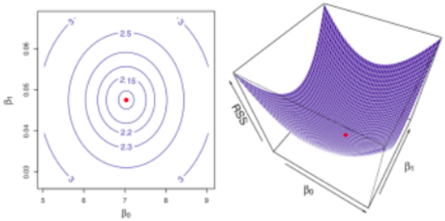


Figure: Optimization in Regression Loss

Regression

$$R^2 = \frac{\mathrm{TSS} - \mathrm{RSS}}{\mathrm{TSS}}$$

$$\mathrm{TSS} = \sum_{i=0}^{i=n} (y_i - \bar{y})^2 \qquad \mathrm{RSS} = e_1^2 + e_2^2 + \cdots + e_n^2$$

Figure: How to compute the R squared metric.

Regression

$$\boldsymbol{MSE} = \frac{\sum \left(\widehat{y}_t - y_t\right)^2}{n}$$

$$\boldsymbol{RMSE} = \sqrt{\frac{\sum \left(\widehat{y}_t - y_t\right)^2}{n}}$$

Figure: How to compute the MSE metric.

Regression

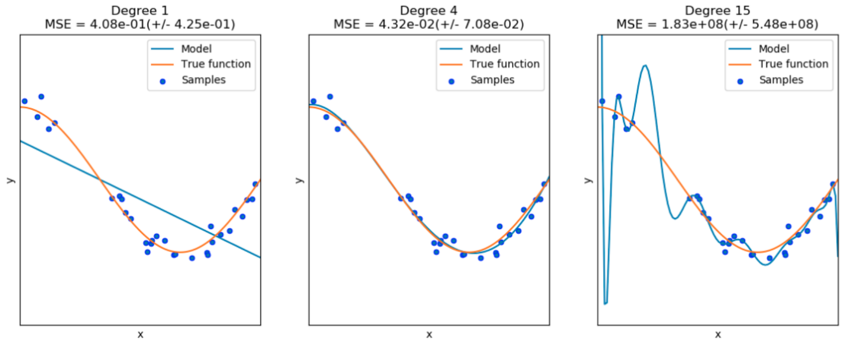


Figure: There exists a trade-off between complexity and simplicity of the learned function.

Instead of learning a separating function now a regression function is learned.

Regularization with ridge regression

Regularization can be used in any learning algorithm. Ridge regression shrinks the regression coefficients by imposing a penalty on their size [3]. The penalization term serves as a regularization policy that helps to make smoother functions.

$$w_{ridge} = argmin(\sum_{i=1}^{n}(y_i w_0 - \sum_{j=1}^{p} x_{ij} w_j)^2 + \lambda \sum_{j=1}^{p} w_j^2)$$

Ridge regression

By changing the λ value we change the **w** coefficients and thus the regression error. The higher the λ parameter, the higher the regularization and the simpler the model.

Regularization with ridge regression

Then, the solution of the beta coefficients will be

$$\beta_{ridge} = (XX^T + \lambda I)^{-1}.Xy$$

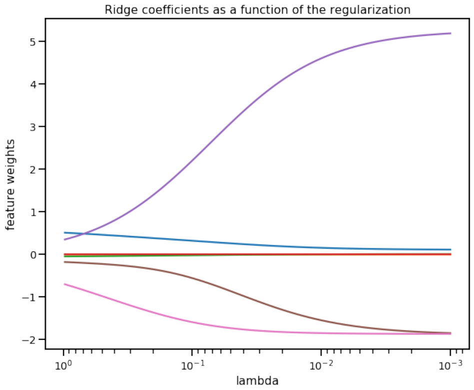


Figure: Ridge regression regularization for 6 features.

Regularization with ridge regression

Exercise 06

Use a grid of 1000 different values of the lambda hyper-parameter and compute the value of the weights and the RMSE of the prediction on validation data.

- Chollet, F. (2018). Keras: The python deep learning library. Astrophysics Source Code Library.
- Bishop, C. M. (2006). Pattern recognition and machine learning. springer.
- Friedman, J., Hastie, T., Tibshirani, R. (2001). The elements of statistical learning (Vol. 1, No. 10). New York: Springer series in statistics.