**Guía 3 – Aprendiza automático “no neuronal”**

**Anotaciones**

**¿Qué es Scikit-Learn?**

Es una biblioteca de Python que proporciona acceso a versiones eficaces de muchos algoritmos comunes. También proporciona una API propia y estandarizada. Por tanto, una de las grandes ventajas de Scikit-Learn es que una vez que se entiende el uso básico y su sintaxis para un tipo de modelo, cambiar a un nuevo modelo o algoritmo es muy sencillo. La biblioteca no solo permite hacer el modelado, sino que también puede garantizar los pasos de preprocesamiento que veremos en el siguiente artículo.

Tiene distintos módulos que iremos viendo en los imports, por ejemplo, en model\_selection vamos a tener los métodos train\_test\_split t KFold que vamos a usar.

**Página oficial**: <https://scikit-learn.org/stable/>

La idea es que podamos ir probando cada clasificador cambiando los parámetros y viendo cómo va variando cada cosa, para lograr el mejor resultado que podamos.

**Tutoriales** que nos pueden servir para aprender sobre la librería:

<https://www.youtube.com/playlist?list=PLzKF7sAvC2S9wrGq0UPqzbypOzwXagE2J>

<https://www.youtube.com/playlist?list=PLJjOveEiVE4Dk48EI7I-67PEleEC5nxc3>

En el Word de teoría también deje algunos videos donde se explican los métodos.

**Importante**:

*Ir leyendo en la documentación sobre cada método o clasificador que utilicemos, para entender que son los parámetros que se utilizan en cada uno y qué es lo que hace.*

**EJERCICIO 1**

**Leer la teoría** sobre el tema de validación cruzada, k-fold y demás.

Esto está dentro de “model\_selection” en la biblioteca porque justamente estas cosas de validación cruzada las hacemos para seleccionar un modelo, y queremos que todos los métodos estén entrenados y probados de la misma forma y con los mismos datos y particiones.

**train\_test\_split**

<https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html>

Este método sería una partición simple, divide una parta para entrenamiento y otra para test.

Otra explicación del train\_test\_split:

<https://keepcoding.io/blog/para-que-sirve-el-train-test-split/>

Tutorial sencillo en video sobre train\_test\_split (usa el problema Iris que vimos en la guía 2):

<https://www.youtube.com/watch?v=8Jfc1w8wwII&list=PLzKF7sAvC2S9wrGq0UPqzbypOzwXagE2J&index=3>

**KFold**

<https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html>

Validación cruzada usando k-fold (leer de teoría).

**load\_digists**

<https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_digits.html>

Es un dataset con números manuscritos. En el código expliqué dos formar de leerlo y como acceder a la información que nos interesa.

**MLPClasifier**

<https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html>

Es un perceptrón multicapa que vamos a usar para probarlo y evaluarlo con distinta cantidad de particiones como indica el enunciado, y calcular la media y varianza cuando tenemos varias particiones.

En la documentación se puede ver todos los parámetros que se pueden usar y qué significa cada uno.

Accuracy y matriz de confusión:

<https://www.youtube.com/watch?v=qCI2zV5XbvE&list=PLzKF7sAvC2S9wrGq0UPqzbypOzwXagE2J&index=4>

**EJERCICIO 2**

**Naive Bayes**

<https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html>

Más teoría: <https://scikit-learn.org/stable/modules/naive_bayes.html>

**Análisis discriminante lineal**

<https://scikit-learn.org/stable/modules/generated/sklearn.discriminant_analysis.LinearDiscriminantAnalysis.html>

**K vecinos más cercanos**

<https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html>

Más teoría: <https://scikit-learn.org/stable/modules/neighbors.html>

En la lista de reproducción que deje con tutoriales hay dos videos sobre este método.

**Árbol de decisión**

<https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html>

Más teoría: <https://scikit-learn.org/stable/modules/tree.html>

**Máquina de soporte vectorial (support vector machine)**

Viendo en la documentación entiendo que hay distintos tipos:

<https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html>

Más teoría: <https://scikit-learn.org/stable/modules/svm.html>

**EJERCICIO 3**

El conjunto de datos “Wine” de Sklearn es un dataset de vinos para utilizar en clasificación.

<https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_wine.html>

Teoría sobre **ensambladores**:

<https://scikit-learn.org/stable/modules/ensemble.html>

**Bagging classifier**:

<https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.BaggingClassifier.html>

**AdaBoost classifier**:

<https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html>