# Implementacion de clasificadores de Scikit-learn en la base de datos Glass.

Echeverry Luis (2420161006)
2420161006@estudiantesunibague.edu.co
Inteligencia Artificial
Programa de Ingeniería Electrónica
Facultad de Ingeniería
Universidad de Ibagué
Ibagué - Tolima
Semestre 2020A
Presentado al ingeniero Jose Armando Fernández Gallego

## 1. Resumen

Haciendo uso de los conocimientos adquiridos a lo largo de todo el curso, enfocados específicamente en la utilización de la librería Scikit-learn de Python; se desea implementar una serie de algoritmos de clasificación los cuales nos permitirán separar, de forma adecuada, al menos tres de las 6 clases de vidrios presentes en la base de datos galss. Los clasificadores que se desean implementar son Perceptron, Perceptron+regresión logística, SVM+linear, SVM+rbf, DT, RF. Los cuales están contenidos en la librería Scikit-learn mencionada previamente.

## 2. Desarrollo

Como primera medida, se hace indispensable describir la base de datos seleccionada. Glass Identification Data Set es una recopilación de las características de fragmentos de vidrios obtenidos en escenas de crímenes, realizada por el departamento de ciencias forenses de los Estados Unidos. Está compuesta por un número total de 214 muestras de 6 tipos diferentes de vidrios. Las cuales, a su vez, tienen los valores correspondientes del contenido de óxido de Sodio (Na), Magnesio (Mg), Aluminio (Al), Bario (Ba), Silicón (Si), Potasio (K), Calcio (Ca) y Hierro (Fe). Además del índice de refracción y un número único para cada muestra tomada, parámetros que fueron evaluados al momento de realizar las pruebas necesarias para crear cada una de las muestras.

Como segunda, tenemos que los algoritmos que se desean aplicar son el perceptrón, el perceptrón más la regresión logística, máquinas de soporte vectorial (SVM) de forma lineal, máquinas de soporte vectorial (SVM) con kernel de función de base radial, los árboles de decisión y los random forests. Cada uno de estos se encuentra dentro de la librería Scikit-learn de Python, facilitando asi su implementación.

## 3. Diagrama de flujo

A continuación, se presenta el diagrama de flujo que corresponde a la implementación de cada uno de los clasificadores mencionados anteriormente, especificando sus correspondientes salidas y entradas.

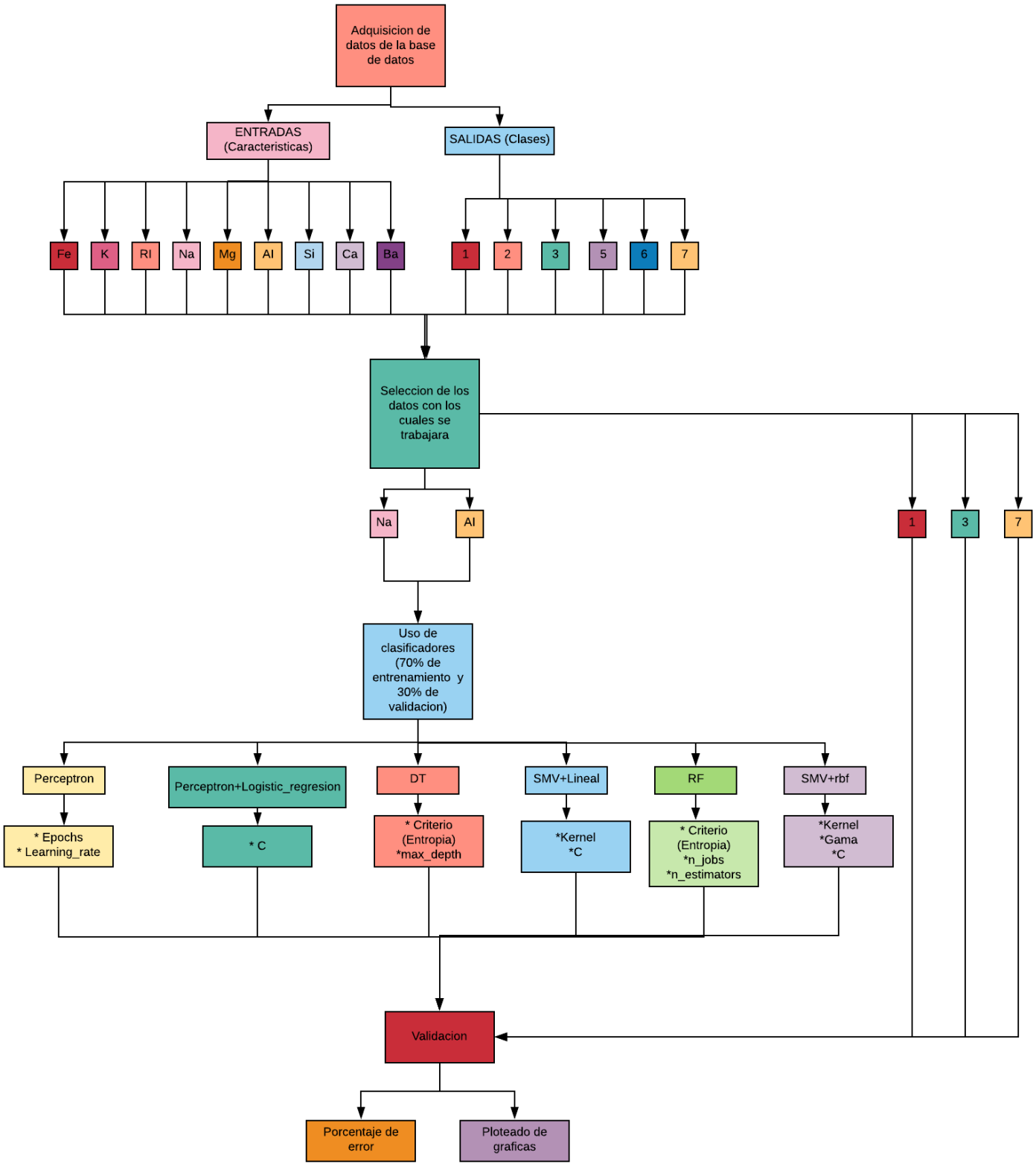


**Figura 1.** Diagrama de flujo del código a implementar.

## 4. Resultados esperados

**Perceptrón.**

Para el caso particular del Perceptrón, podemos recordar que en entregas anteriores este algoritmo fue implementado de forma practica junto con el ADALINE. De esta manera podemos esperar que los resultados que serán obtenidos van a ser muy similares a los presentados en la entrega anterior, logrando separar, de forma adecuada, cada una de las regiones de decisión como se muestra en la figura 2.

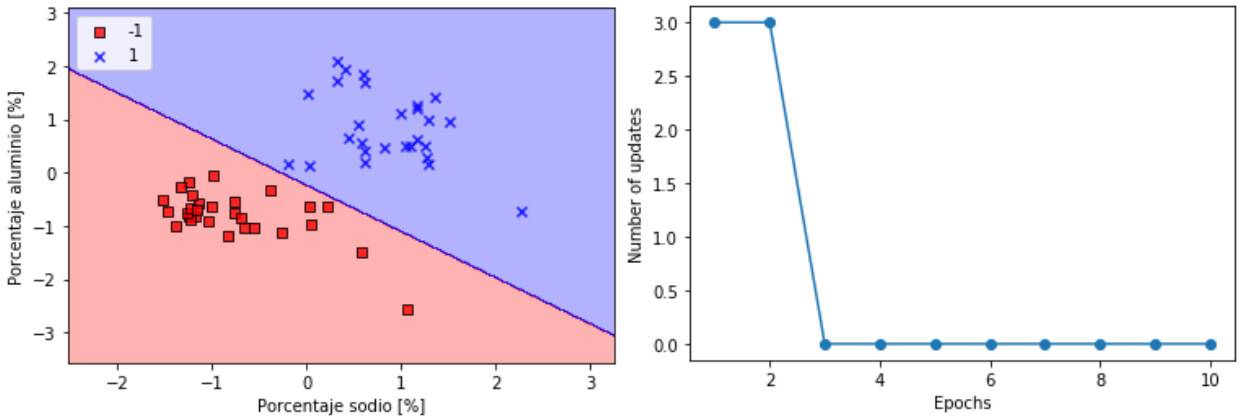


**Figura 2.** Clasificación por Perceptrón.

**Perceptrón más la regresión logística.**

Para este caso, se espera que, al tener un cambio en la función de activación, remplazando a la función escalón unitario del perceptrón por una sigmoide, el resultado de las regiones de decisión sea un poco más acertado, acercándose más a una clasificación completamente acertada. Esto no implica que los datos vayan a ser separados en su totalidad de forma correcta, ya que existen algunos que tienen unas características bastante diferentes al resto de datos pertenecientes a esa misma clase.

**Máquinas de soporte vectorial.**

Para este clasificador en particular se espera que la separación se en forma de franjas, muy similar a la que se presenta en el caso del libro con la base de datos de iris. Ya que este algoritmo lo que permite es una reducción en el margen de clasificación con el cual se separan cada una de las clases escogidas.

**Máquinas de soporte vectorial con Kernel de base radial.**

En este caso especifico tenemos que la clasificación se podrá hacer de una forma un poco más adecuada, ya que esta implementación nos permite una clasificación no lineal, separando de forma mas adecuada las clases seleccionadas.

**Arboles de decisión.**

A pesar de la simplicidad de este algoritmo, es de los mas potentes para la separación de clases. Para este caso en particular se espera que el árbol de decisión realice una clasificación correcta de todos los datos.

**Random forests.**

Dentro de todos los clasificadores, este es el que mejor se adecua a todos los procesos de separación. Como se pudo evidenciar en el ejemplo del libro, la separación realizada por este algoritmo fue correcta y no se tenia el riesgo latente de un sobre entrenamiento. Las regiones fueron halladas de forma correcta separando las tres clases.