UNIVERSIDAD DE SANTIAGO DE CHILE
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA INFORMÁTICA
REDES NEURONALES
Ayudante Ignacio Ibáñez Aliaga
ignacio.ibanez@usach.cl

Laboratorio 1

Fundamento Teórico

El machine learning es la ciencia que permite a los computadores actuar de manera inteligente, sin necesidad de haber sido programados explícitamente. Un proceso de machine learning consta de 5 grandes etapas:

- **Recolección de datos:** esta etapa es fundamental y corresponde al fenómeno que se desee estudiar. Los datos pueden provenir de diversas fuentes, como por ejemplo: señales (biológicas, audio, imágenes, geofísicas, etc.), texto (tweets, mails, diálogos de películas, etc.), imágenes (Instagram, flickr, mapas, etc.), entre otras.
- **Pre-procesamiento de datos:** en esta etapa se debe transformar los datos a un formato que el algoritmo de aprendizaje entienda, generalmente se deben presentar como vectores numéricos.
- **Exploración de datos:** esta etapa consiste en analizar los datos, para lo cual la estadística será una herramienta esencial. Entrenamiento del modelo: aquí es donde aparecen los diferentes algoritmos de aprendizaje. Estos algoritmos de manera general buscan patrones en los datos, los que permiten al modelo aprender el comportamiento implícito de los datos para tomar decisiones, estas decisiones son clasificar (escoger una clase 0 o 1) o predecir en un valor (determinar el valor real en un siguiente instante de tiempo).
- **Evaluación del modelo:** cuando ya se tiene un modelo entrenado es momento de ponerlo a prueba, para esto se le presenta un conjunto de datos diferentes y evalúa en base a diferentes métricas (precisión, sensibilidad, especificidad, entre otras), en caso que los resultados no sean buenos se puede volver a entrenar el modelo de aprendizaje con diferentes hiper-parámetros (a esto se le denomina “tuning the hyperparameters of a model”).

- **Modelo en producción:** finalmente se utiliza el modelo enfrentando la realidad, esto puede hacer que se deba volver a alguna etapa anterior para mejorar la calidad de los resultados en base a la métrica que se desee.

En la actualidad existen diferentes herramientas que son utilizadas en el machine learning una de estas corresponde a las redes neuronales que su primer acercamiento ocurrió en 1958 cuando el psicólogo Frank Ronsenblant desarrolló un modelo simple de neurona basado en el modelo de McCulloch y Pitts, con una regla de aprendizaje en base a la corrección del error, su modelo fue nombrado como perceptrón simple que tenia la capacidad de aprender a reconocer patrones.

El perceptrón simple está constituido por un conjunto de sensores de entrada que recibe los patrones de entrada a reconocer o clasificar y una neurona de salida que se utiliza para clasificar a los patrones de entrada en dos clases.

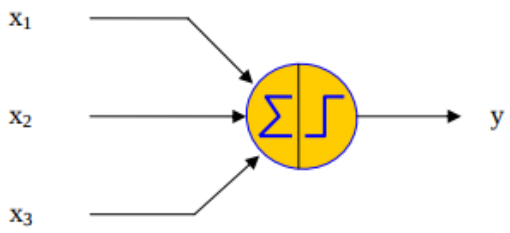

Figura 1: Perceptrón simple.

Con la evolución de las redes neuronales se han generado múltiples variaciones en su estructura, en donde se varia la cantidad de neuronas en la capa oculta, cantidad de capas ocultas y métodos de capacitación para lograr que la red neuronal aprenda, entre otros hiper-parámetros.

Pero todos estos tipos de redes adolecen a un proceso de capacitación costoso en el tiempo que generalmente adopta algoritmos de retro-propagación de errores basados en gradiente, y consecuentemente es propenso a atascarse en mínimos locales. El enfoque de las ELM o Extreme Learning Machine tiene como objetivo evitar un procedimiento de entrenamiento iterativo que consume mucho tiempo y al mismo tiempo mejorar el rendimiento de generalización. La estructura de este tipo de red es apoyado por la idea de que debe haber algunas partes del cerebro donde las configuraciones de las neuronas no depende del entorno, el algoritmo ELM aprovecha este argumento biológico y emplea neuronas sin ajuste entre la capa de entrada y la capa oculta, para resolver los problemas

adversos por los algoritmos de retro-propagación son obtenidos los pesos de la capa oculta y de salida por medio de mínimos cuadrados. Algoritmo de ELM:

1. Asigne aleatoriamente los pesos de conexión entre la capa de entrada y la capa oculta W_1 y los bias B.
2. Calcula la matriz de salida de la capa oculta $H = g(W_1X + B)$.
3. Obtener los pesos entre la capa oculta y la capa de salida usando el método de mínimos cuadrados $W_2 = H^+T$ donde $H^+ = H^T(HH^T)^{-1}$ que corresponde al inverso generalizado de Moore-Penrose de la matrix H, obteniendo todos los pesos de la red.
4. Usar la red con el conjunto de test, usando para obtener la salida $H = g(W_1X_2 + B)$ con el conjunto de test y la salida de la red por $T^* = g_2(HW_2)$.
   - **X:** Datos de entrada al modelo.
   - **T:** Datos de salida esperados.
   - T^***:** Datos de salida obtenidos por el modelo.
   - **B:** Bias de la capa oculta.
   - W_1**:** Pesos entre la capa de entrada y oculta.
   - W_2**:** Pesos entre la cada oculta y de salida.
   - **g(x):** Función de activación en la capa oculta.
   - $g_2(x)$**:** Función de activación en la capa salida.
   - X_2**:** conjunto de test.
5. Link de referencias
   a) Extreme learning machine: Theory and applications
   b) Two-hidden-layer extreme learning machine for regression and classification

Actividades

1. Implementar un perceptrón simple en Python utilizando la librería numpy y matplotlib (para graficar). Probar el perceptrón para las compuertas lógicas AND, OR y XOR.
2. Implementar una Extreme Learning Machine en Python utilizando la librería numpy y matplotlib (para graficar). Probar la red para las mismas compuertas lógicas que el perceptrón simple.

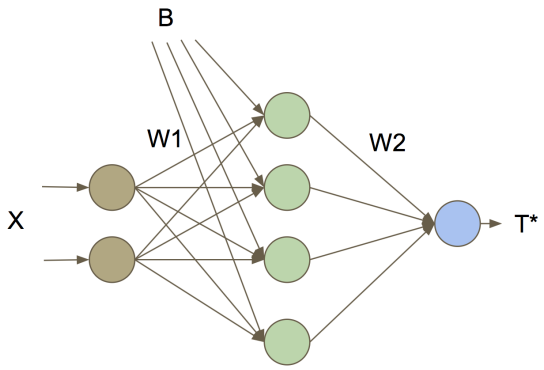

Figura 2: Extreme Learning Machine.

3. Seleccionar un dataset de clasificación, normalizar los datos y graficar todos los atributos con todos, colocando en color las diferentes clases.

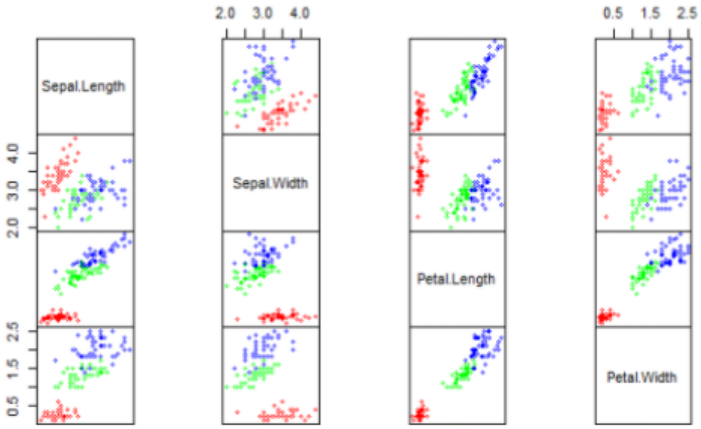

Figura 3: Ejemplo de gráfico con dataset iris con 4 atributos y 3 clases distintas.

4. Dividir el dataset en 70 % train y 30 % test, usar la ELM con 15 diferentes configuraciones variando cantidad de neuronas en la capa oculta. Presentar el gráfico de número de neuronas vs error en el conjunto de test y la matriz de confusión para el mejor modelo generado.

Presentación del entregable

Todas las actividades que fueron listadas en la sección anterior deben ser realizadas en jupyter notebook con python 3.*, en donde la presentación del archivo debe ser de la siguiente manera:

1. Marco teórico del perceptrón simple.
2. Código de la primera actividad.
3. Resultado gráfico de las diferentes compuertas lógicas.
4. Análisis de los resultados obtenidos.
5. Marco teórico de la ELM.
6. Código de la segunda actividad.
7. Resultado gráfico de las diferentes compuertas lógicas.
8. Análisis de los resultados obtenidos comparando con el perceptrón simple.
9. Descripción del dataset escogido.
10. Código de lectura del dataset y normalización.
11. Presentación gráfica del dataset, es decir, actividad 3.
12. Código de separación del dataset en conjunto de test y train.
13. Código de diferentes configuraciones de la ELM.
14. Presentación gráfica de número de neuronas vs error en el conjunto de test y matriz de confusión del mejor modelo.
15. Análisis de los resultados obtenidos.
16. Conclusiones.

Fecha de entrega: 4 de Mayo del 2018