Laboratorio de aprendizaje de máquinas
Práctica 2: Redes Neuronales Algoritmo de backpropagation
Universidad Tecnológica de Pereira, Facultad de ingenierías, Maestría en Ingeniería de Sistema

Msc Ing Carlos Alberto Henao Baena
caralbhenao@utp.edu.co

Objetivo

Al finalizar esta práctica el estudiante conocerá e identificará los elementos básicos en una red neuronal full conected (FC). También explorará el algoritmo de backpropagation como regla de aprendizaje en una red FC. Finalmente se examina aplicaciones de la FC en tareas de clasificación y regresión evaluando cómo los hiperparámetros afectan el desempeño del modelo.

Parte 1.0 (Propagación)

A continuación, desarrolle los siguientes pasos en un *Google Colab Notebook*

1. Genere la siguiente base de datos:

Tabla1: Base de datos para una compuerta lógica XOR

X1	X2	Y
0	0	0
0	1	1
1	0	1
0	0	0

Donde X1 y X2 son las entradas y Y la salida.

2. Implemente una arquitectura FC que cumpla los siguientes requerimientos

   - El número capas de neuronas es modificable
   - El número de neuronas por cada es modificable
   - El tipo de función de activación por capa de neuronas es modificable y estás podrán ser $tanh(a)$, $\sigma(a)$ o lineal.
   - Inicialice los pesos y los bias de la red FC utilizando un método de muestreo estadístico.

3. Cálculo de $f(W, X_i)$ (ecuación 1). Desarrolle una función que permita propagar cada dato [X1, X2] desde la entrada hacia la salida almacenando el valor z^l , realice un seguimiento de los resultados evaluando la salida en cada una de las neuronas que conforman la red FC.

Parte 1.1 (backpropagation)

1. Establezca un valor *LearningRate = 0.3*
2. Conocido el valor de *LearningRate* y el vector z^l implemente el algoritmo de backpropagation (ver anexo 1) asumiendo las siguientes funciones de perdida:

   a. $LSE = \frac{2}{1}\sum_{i=1}^{n}\left(y_i - f(W, X_i)\right)^2$

   b. $BCE = -\sum_{i=1}^{n} y_i \ln f(W, X_i) + (1 - y_i)\ln\left(1 - f(W, X_i)\right)$

   Donde $f(W, X_i)$ es la representación matemática de la red (modelo **Feed-Foward**)

3. Grafique $LSE\ vs\ \text{Épocas}$ y $BCE\ vs\ \text{Épocas}$

Parte 1.2 (validación)

Una vez finalizado el proceso de optimización continué con lo siguiente:

4. Realice la predicción sobre la base de datos de la tabla 1 utilizando la ecuación 1

$$\hat{\boldsymbol{y}} = f(W, X_i) \tag{1}$$

5. ¿Varié las funciones de activación por capa de neurona hasta obtener un buen desempeño?
6. ¿En este caso con cuál función de pérdida se obtiene un mejor desempeño?
7. Realice la predicción $\hat{\boldsymbol{y}}$ definiendo en la capa de salida de la red FC una función $\sigma(a)$ (definiendo los siguientes umbrales 0.7, 0,8, 0.9 y 0.95) sobre el conjunto de datos de la tabla 1.
8. Conocidas las predicciones (en cada umbral) sobre el conjunto de datos de *Test* evalué las siguientes métricas (en cada clase):
   - Matriz de confusión
   - La sensibilidad
   - Especificidad
   - Precisión
   - Exactitud
   - ROC
   - UUROC

Parte 1.3 (Pruebas con bases de datos adicionales, caso clasificador biclase)

1. Considere el dataset *DataLeastSquareClasificationTwoClass.csv*
2. Cargue la base utilizando la instancia: ***!git clone dirección_del_dataset***
3. Transfiera el dataset a un frame panda utilizando el método *pd.read.csv('dirección_dataset_cargado.csv')*
4. Divida el dataset de la siguiente forma: *75%* para entrenamiento el restante para validación
5. Construya la matriz $\boldsymbol{X}$ y el vector $\boldsymbol{y}$

6. Construya un modelo FC repitiendo desde el paso 2 del literal **1.0** hasta el paso 8 del paso de sesión **1.2,** asumiendo como función de BCE, la predicciones y las métricas de desempeño debe calcularse utilizando el conjunto de datos de validación.
7. Grafique en un mismo gráfico lo siguiente
    - $X1_{Train}$ vs $X2_{Train}$
    - $X1_{Test}$ vs $X2_{Test}$
    - El plano de decisión y_0 que para un caso lineal viene dado por la curva $\frac{1}{2} = f(W, X_i)$, donde $f(W, X_i)$ corresponde al modelo **Feed-Foward**

Parte 1.3 (Pruebas con bases de datos adicionales, caso regresión)

1. Considere el dataset *DataXSquareNeuronalNetwork.csv*
2. Cargue la base utilizando la instancia: ***!git clone dirección_del_dataset***
3. Transfiera el dataset a un frame panda utilizando el método *pd.read.csv('dirección_dataset_cargado.csv')*
4. Divida el dataset de la siguiente forma: *75%* para entrenamiento el restante para validación
5. Construya la matriz $\boldsymbol{X}$ y el vector $\boldsymbol{y}$
6. Construya un modelo FC repitiendo desde el paso 2 del literal **1.0** hasta el paso 8 del paso de sesión **1.2,** asumiendo como función de LSE.
7. Calcule las predicciones sobre el conjunto de datos de validación y evalué las siguiente métricas de desempeño
    - $RMSE$
    - $\rho_{Y_{Test}, \hat{Y}_{Test}}$
    - En un mismo gráfico muestre lo siguiente: Y_{Test} vs Y_{Test} y : Y_{Test} vs $\hat{Y}_{Test}$. Qué puede concluir de este gráfico
8. Grafique en un mismo gráfico lo siguiente:
    - Gráfico 1: X_{Test} vs Y_{Test} y X_{Test} vs $\widehat{\boldsymbol{y}_{Test}}$
    - Gráfico 2: X_{Train} vs Y_{Train} y X_{Train} vs $\widehat{\boldsymbol{y}_{Trai}}$

9. Repita los resultados el procedimiento **Parte 1.3** con los siguientes datos *DataHeavsideNeuronalNetwork.csv y DataAbsoluteNeuronalNetwork.csv*