**Redes Neuronales Básicas** Parte 1 – Python y Numpy

**Oscar Fernando López Barrios - Carné 20679**

**Alejandro José Gómez Hernández - Carné 20347**

En clase vimos un modelo simple, utilizando solo Python y Numpy, para resolver Regresiones Lineales mediante redes neuronales. Utilizando el código desarrollado (o si lo desea uno propio), responda a las siguientes preguntas:

1. Cambie el número de observaciones a 100,000. Explique que es lo que ocurre en términos de:
   1. El tiempo de ejecución para resolver el problemas

R// En este caso el tiempo de ejecución aumenta de mayor manera debido a que al momento de realizar tantas observaciones más, los cálculos que se deben de realizar también aumentan. Por lo tanto el tiempo que se ejecuta para resolver problemas es mayor.

* 1. El resultado final vrs lo encontrado en clase: es igual, o diferente...¿porqué?

R// En general el resultado final se diferencia al obtenido durante la clase debido a que se cuenta con una mayor cantidad de tiempo de ejecución, además que se puede notar que cuando se realiza esta nueva prueba con la cantidad definida la pérdida aumenta con respecto a la cantidad de las 1000 observaciones que se tenía antes en clase.

* 1. Las graficas para representar los datos/resultados

R// Se puede observar a comparación de las gráficas anteriores que existen diferencias con respecto a como se mira la cantidad de datos que existen dispersos por toda la gráfica, en general se puede ver una cantidad de datos mayormente juntos y la gráfica se puede ver totalmente lineal.

1. Cambie el número de observaciones a 1,000,000. Explique que es lo que ocurre en términos de:
   1. El tiempo de ejecución para resolver el problemas

R// Así como pasó con el ajuste anterior el tiempo de ejecución fue mucho mayor al que brindaba

* 1. El resultado final vrs lo encontrado en clase: es igual, o diferente...¿porqué?
  2. Las graficas para representar los datos/resultados

1. “Juegue” un poco con el valor de la tasa de aprendizaje, por ejemplo 0.0001, 0.001, 0.1, 1.



CC3074 – Minería de Datos

Laboratorios 6 y 7

Semestre I - 2023

Para cada uno de estos indique:

* 1. ¿Qué ocurre con el tiempo de ejecución?
  2. ¿Qué ocurre con la minimización de la pérdida?
  3. ¿Qué ocurre con los pesos y los sesgos?
  4. ¿Qué ocurre con las iteraciones?
  5. ¿El problema queda resuelto o no?
  6. ¿Cuál es la apariencia de la última gráfica? ¿Se cumple con la condición de que sea de 45 grados?

1. Cambie la función de pérdida “L2-norm” a la misma pero sin dividir por 2. Explique lo que ocurre en términos de:
   1. El tiempo que se tarda el algoritmo en terminar, comparado a lo que vimos en clase
   2. Si la pérdida se minimiza igual que lo que vimos en clase
   3. Si los pesos y sesgos son parecidos a los vistos en clase
   4. Si el problema se resuelve como ocurrió en clase
   5. Si se obtiene un mejor resultado al hacer más iteraciones
2. Cambie la función de pérdida de la “L2-norm” a la “L1-norm”. Explique lo que ocurre en términos de:
   1. El tiempo que se tarda el algoritmo en terminar, comparado a lo que vimos en clase
   2. Si la pérdida se minimiza igual que lo que vimos en clase
   3. Si los pesos y sesgos son parecidos a los vistos en clase
   4. Si el problema se resuelve como ocurrió en clase 5. Si se obtiene un mejor resultado al hacer más iteraciones

6. ¿Tendrá una de estas más limitaciones que la otra?

1. Cree una función f(x1,x2) = 13 \* x1 + 7 \* x2 - 12.
   1. ¿Funciona el algoritmo de la misma forma?

# Parte 2 – Tensorflow2

En clase vimos un modelo simple de una red neuronal utilizando TensorFlow 2. Utilizando el código desarrollado (o si lo desea uno propio pero que funcione correctamente), responda a las siguientes preguntas:

1. Cambie el número de observaciones a 100,000. ¿Qué ocurre?



CC3074 – Minería de Datos

Laboratorios 6 y 7

Semestre I - 2023

1. “Juegue” un poco con la tasa de aprendizaje. Los valores como 0.0001, 0.001, 0.1, 1 son interesantes para observar ¿Qué diferencias se observan? ¿Se comporta bien el algoritmo?
2. Cambie la función de pérdida. Una función altenativa es la “Huber Loss”.

La función de pérdida Huber es más adecuada que la L2.norm cuando tenemos valores atípicos, ya que es menos sensitiva a los mismos (en nuestro ejemplo no tenemos valores atípicos, pero seguramente se topará con ellos en el futuro). La L2-norm eleva todas las diferencias al **cuadrado**, por lo que los valores atípicos tienen mucha influencia sobre los resultados. La sintáxis correcta de la función de pérdida Huber es “huber\_loss”.

¿Cómo se comparan los resultados al cambiar la función de pérdida?

Referencia:

https://www.tensorflow.org/versions/r1.15/api\_docs/python/tf/keras