**Práctica 3. Redes Neuronales**

**Maravilla Pérez Vianey**

**Mondolla Cervantes Erin**

**Ramirez Mendez Kevin**

**6AM1**

Utilizando el DEMO (<https://playground.tensorflow.org/>) que vimos en la clase pasada, realicen los siguientes puntos, contesten las preguntas, justificando su respuesta y agreguen evidencia de las redes neuronales (imagen).

***Ejercicio 1. Lineales***

1. El modelo de la Figura 1 combina dos características de entrada en una sola neurona.

* ***¿Aprenderá este modelo alguna no linealidad?*** Ejecútenlo para confirmar su suposición.

Tomando en cuenta la Figura 1 vemos un comportamiento de los datos muy diferente y disperso para poder clasificarlo de una manera lineal, su estructura no nos lo permite, si tratamos de clasificarlos de manera lineal, podemos notar que su clasificación es mala.

La figura 1 al clasificarla de manera lineal nos muestra la siguiente figura:

Interfaz de usuario gráfica

Descripción generada automáticamente

Como sabemos si un modelo es malo, significa que la relación entre las variables de entrada y la salida no es lineal.

En este caso, el modelo puede utilizar funciones no lineales para transformar los modelos de clasificación para que esto nos sea útil cuando los datos no son fácilmente separables mediante una línea recta

1. Prueben aumentar el número de neuronas en la capa oculta de 1 a 2, y prueben también cambiar de una activación lineal a una activación no lineal como ReLU.

**Interfaz de usuario gráfica

Descripción generada automáticamente**

* ***¿Se puede crear un modelo que aprenda no linealidades?***

ReLU nos ayuda porque es útil ya que introduce no linealidad en el modelo y esto permite que la red neuronal aprenda relaciones más complejas entre las entradas y las salidas.

En este caso vemos que ReLU nos ayuda, pero no mucho, ya que hace nuestra clasificación a medias, ya que 2/4 de nuestro modelo se queda “fuera” y eso no nos sirve.

* ***¿Puede modelar los datos con eficacia?***

***Como vemos en la imagen, no se podría hacer de manera muy eficiente por lo ya dicho entonces podemos encontrar más soluciones a este problema, como por ejemplo la siguiente asignación***

1. Intenten aumentar el número de neuronas de la capa oculta de 2 a 3, utilizando una activación no lineal como ReLU.

**Interfaz de usuario gráfica, Diagrama

Descripción generada automáticamente**

* **¿Puede modelar los datos eficazmente?**

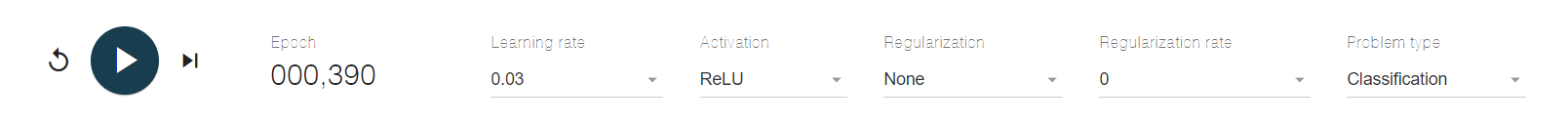
Como podemos notar en la imágen, agregando más neuronas en la capa oculta mejora el rendimiento, vemos más una no linealidad, esto con ayuda de ReLU y vemos mejor la clasificación de los datos, no de manera perfecta pero mucho más clara y precisa que solo ocupando 2 neuronas, ReLU es muy eficiente ya que solo implica una comparación y una asignación.

* **¿Cómo varía la calidad del modelo de una ejecución a otra?**

Se logra ver un modelo de mejor calidad, ya que vemos que agregando una neurona más a las capa oculta, entonces le da mejor flujo y le ayuda a dar más no linealidad a nuestro modelo.

1. Continúen experimentando, añadiendo o eliminando capas ocultas y neuronas por capa. También pueden cambiar los ritmos de aprendizaje, la regularización y otros parámetros de aprendizaje.

* **¿Cuál es el menor número de neuronas y capas que puedes utilizar para obtener una pérdida en la prueba de 0,177 o inferior?**

****

**Los parámetros utilizados fueron los de la imagen de arriba para obtener un modelo y obtener una pérdida en la prueba menor a 0,177**

Interfaz de usuario gráfica, Gráfico

Descripción generada automáticamente

* **¿El aumento del tamaño del modelo mejora el ajuste o la rapidez de convergencia? ¿Cambia la frecuencia con la que converge a un buen modelo? Por ejemplo, prueben la siguiente arquitectura:**
  + **Primera capa oculta con 3 neuronas**
  + **Segunda capa oculta con 3 neuronas**
  + **Tercera capa oculta con 2 neuronas.**

Interfaz de usuario gráfica, Gráfico, Gráfico de cajas y bigotes

Descripción generada automáticamente

Esa arquitectura nos funciona después de darle tiempo al modelo para que funcione de manera optima con RELU, al menos en la rapidez no nos ayuda, debemos dejar que actué con lentitud a comparación de otras arquitectura.

Interfaz de usuario gráfica, Diagrama

Descripción generada automáticamente

Figura 1.

***Ejercicio 2. Inicialización***

Este ejercicio utiliza de nuevo los datos XOR, pero examina la repetibilidad del entrenamiento de Redes Neuronales y la importancia de la inicialización.

1. Ejecuten el modelo cuatro o cinco veces como se muestra en la Figura 2. Antes de cada prueba, pulsen el botón *Reiniciar* la red para obtener una nueva inicialización aleatoria. (El botón Reiniciar la red es la flecha circular de reinicio que está justo a la izquierda del botón Reproducir). Dejen que cada prueba se ejecute durante al menos *500 pasos* para asegurar la convergencia.

* **¿A qué forma converge la salida de cada modelo?**

**Ejecuciones:**

**1.-**

**Interfaz de usuario gráfica

Descripción generada automáticamente**

**Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente2.-**

**3.-**

**Interfaz de usuario gráfica

Descripción generada automáticamente**

**4.-**

**Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente**

**Interfaz de usuario gráfica

Descripción generada automáticamente5.-**

* **¿Qué nos dice esto sobre el papel de la inicialización en la optimización no convexa?**

Nos salen diferentes salidas, lo cual nos indica que tal vez no es muy optima nuestra arquitectura, por lo cual es un poco optimo, lo adecuado es optimizarlo y ver si funciona con los reiniciados del primer paso.

1. Intenten hacer el modelo ligeramente más complejo añadiendo una capa y un par de nodos extra. Repitan las pruebas del punto 1.

**Ejecuciones:**

1.-

Interfaz de usuario gráfica

Descripción generada automáticamente2.- Interfaz de usuario gráfica, Diagrama

Descripción generada automáticamente

Diagrama

Descripción generada automáticamente3.-

Diagrama

Descripción generada automáticamente con confianza media4.-

5.-

Diagrama

Descripción generada automáticamente

* **¿Añade esto alguna estabilidad adicional a los resultados?**

**Ahora vemos un modelo más eficaz y sobre todo con una baja perdida tanto de prueba como de entrenamiento, esto es muy importante ya que obtenemos un modelo más correcto para lo que tenemos que hacer o interpretar.**

Interfaz de usuario gráfica, Diagrama

Descripción generada automáticamente

Figura 2.

***Ejercicio 3. Red neuronal en Espiral***

1. Entrenen el mejor modelo que puedan, utilizando sólo X1 y X2. Siéntanse libres de añadir o eliminar capas y neuronas, cambiar los parámetros de aprendizaje como la tasa de aprendizaje, la tasa de regularización y el tamaño del conjunto.

Diagrama

Descripción generada automáticamente

**¿Cuál es la mejor pérdida de prueba que puede obtener?**

**Como podemos ver en la imagen anterior, la manera más acertada y la que nos dío una perdida tanto en prueba como en entrenamiento con los parámetros que vemos en la imagen.**

1. Incluso con las redes neuronales, a menudo es necesaria cierta ingeniería de características para lograr el mejor rendimiento. Prueben añadir funciones adicionales de producto cruzado u otras transformaciones comoy

**¿Obtiene un modelo mejor?**

**Se obtiene un modelo más optimo para la red**