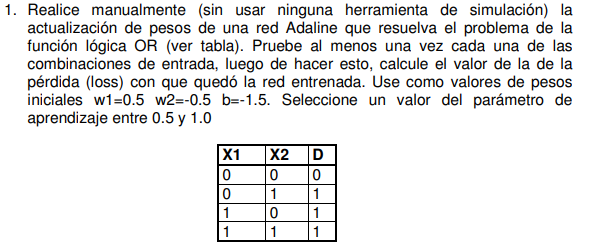
Respuestas de Trabajando con el gradiente descendente

Diego Iván Perea Montealegre (2185751) [diego.perea@uao.edu.co](mailto:diego.perea@uao.edu.co)

Facultad de Ingeniería, Universidad Autónoma de Occidente

Cali, Valle del Cauca



**1.**

Ecuación de peso

Ecuación de bias

Ecuación de error

Ecuación de salida

Ecuación de perdida

Primera combinación

Salida

Error

Peso

Bias

Segunda combinación

Salida

Error

Peso

Bias

Tercera combinación

Salida

Error

Peso

Bias

Cuarta combinación

Salida

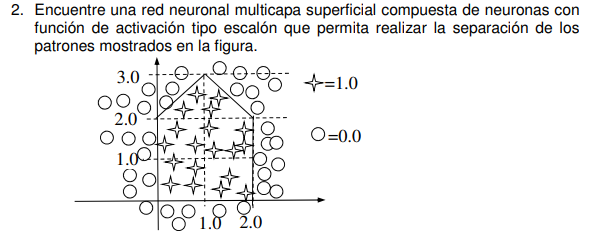
Error

Peso

Bias

Total de perdida

}



Se realiza la selección del trazado para estimar la clasificación de las estrellas frente a los círculos dado una línea de forma de pendiente

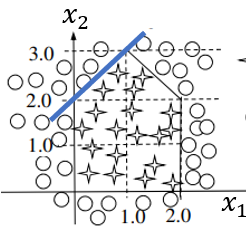


Figura 1. Trazo de primera línea

Darle valor de

Remplazar en

Pesos Resultantes

Con se dirige a los ceros del sistema intuyendo para así en dirección

Resultando ->

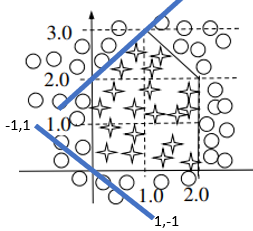


Figura 2. Trazo de pesos primera línea

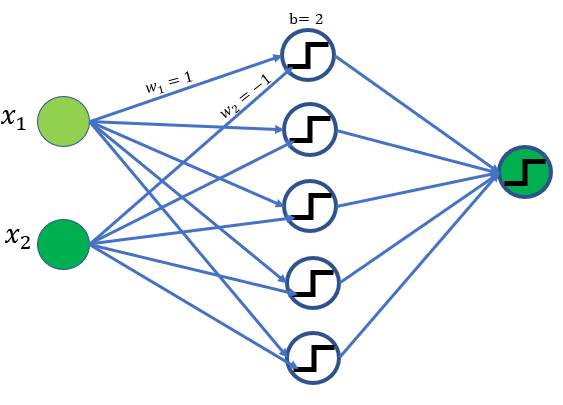


Figura 3. Red neuronal multicapa con primera línea

Dado a este trazo se procede a realizar el segundo trazo

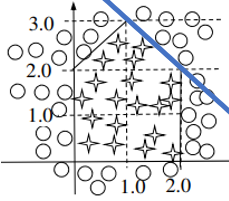


Figura 4. Trazo de segunda línea

Darle valor de

Pesos Resultantes

Con se dirige a los ceros del sistema intuyendo para así en dirección

Resultando ->

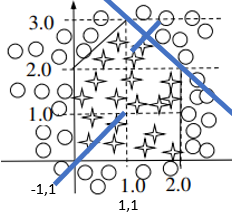


Figura 5. Trazo de pesos segunda línea

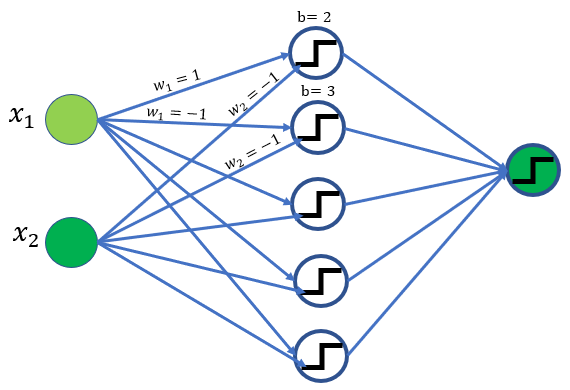


Figura 6. Red neuronal multicapa con segunda línea

Dado a este trazo se procede a realizar el tercer trazo

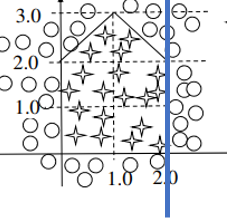


Figura 7. Trazo de tercera línea

Darle valor de

Pesos Resultantes

Con se dirige a los ceros del sistema intuyendo para así en dirección

Resultando ->

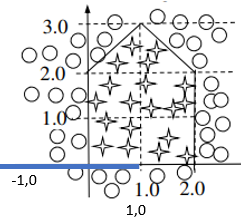


Figura 8. Trazo de pesos tercera línea

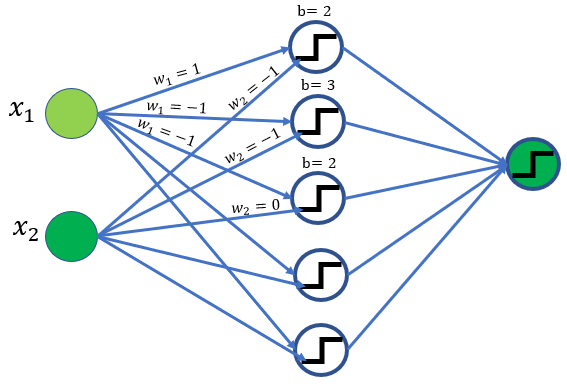


Figura 9. Red neuronal multicapa con tercera línea

Dado a este trazo se procede a realizar el cuarto trazo

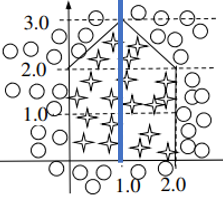


Figura 10. Trazo de cuarta línea

Darle valor de

Pesos Resultantes

Con se dirige a los unos del sistema

Resultando ->

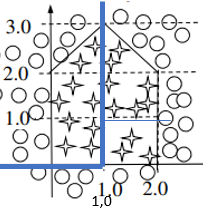


Figura 11. Trazo de pesos cuarta línea

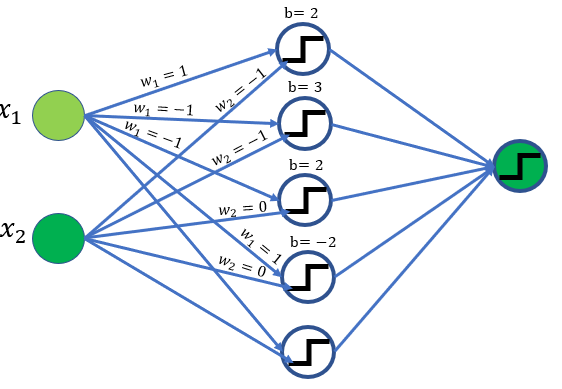


Figura 12. Red neuronal multicapa con cuarta línea

Dado a este trazo se procede a realizar el quinto trazo

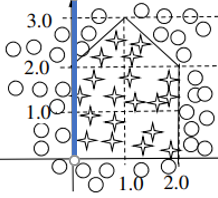


Figura 13. Trazo de quinta línea

Darle valor de

Pesos Resultantes

Con se dirige a los unos del sistema

Resultando ->

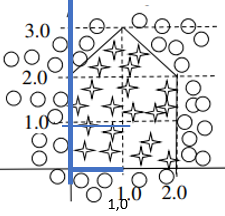


Figura 14. Trazo de pesos quinta línea

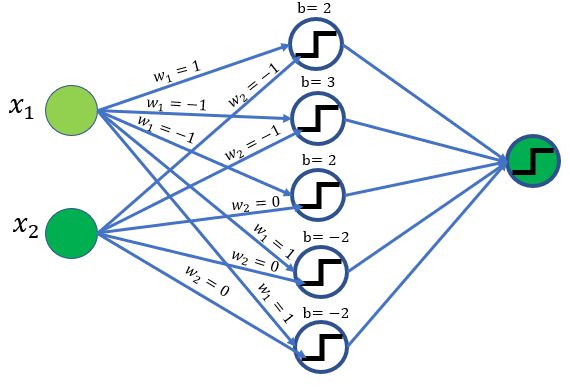


Figura 15. Red neuronal multicapa step con quinta línea

Desarrollo de las salidas según el trazo :

Primer trazo neurona

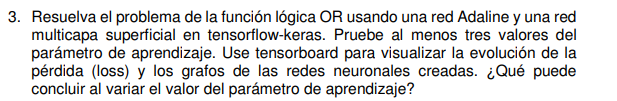
Segundo trazo neurona

Tercer trazo neurona

Cuarto trazo neurona

Quinto trazo neurona

Salida de la neurona



Se procede a realizar la Red Adaline para la compuerta OR , en el que se toma en cuenta las siguientes librerías

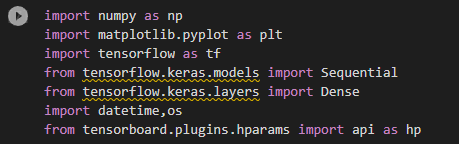


Figura 16. importación de librería necesarias para la ejecución de la Red Adaline OR

Para crear la Red Adaline se declara las variable x\_train como las entradas y de salida y\_train , para que realmente sea una Red Adaline se tuvo que realizar una función en el que se denominó como model\_or , en el que esta se configuraba la arquitectura , de tipo secuencial (una después de otra) , dos neuronas de entrada y una neurona de salida.

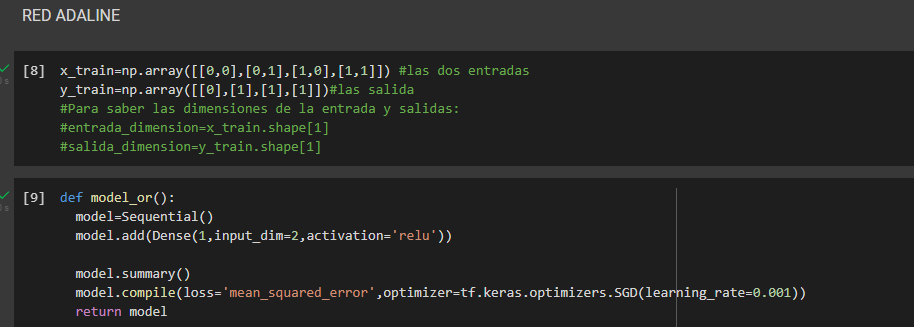


Figura 17. Configuración de arquitectura de Red Adaline OR

Se llama la función model\_or en la que se guerda en la variable modelo para asi mejorar el entendimiento del código y procesos a ejecutar, posteriormente se configura el entrenamiento en el que tensoboard se visualizará las épocas y los , para observar la eficiencia y la predicción dicha de la red Adaline creado con los parámetros anteriores mencionados.

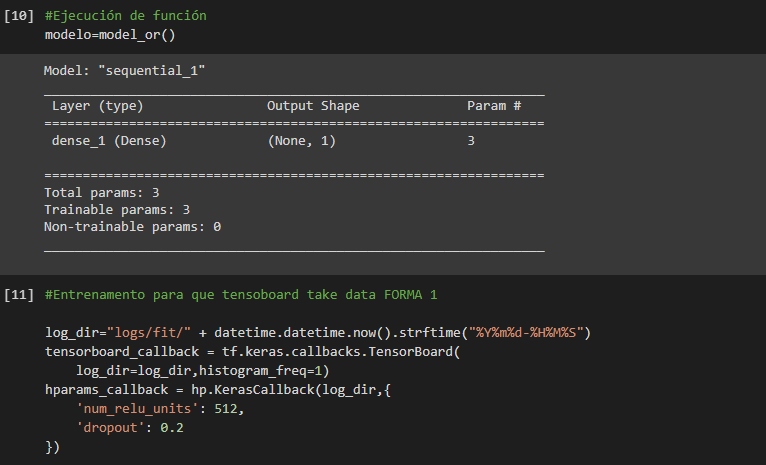


Figura 18. Configuraciones de Tensorboard OR

Además se realiza el entrenamiento del modelo, en el que tiene épocas de 500 y un batch\_size de 1

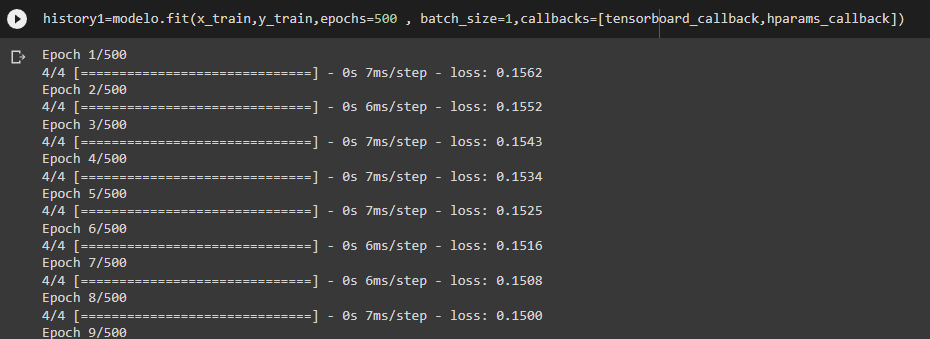


Figura 19. Entrenamiento de la red Adaline OR

Se llama a la funcion predict ara que prdiga los resultados frente a las entradas

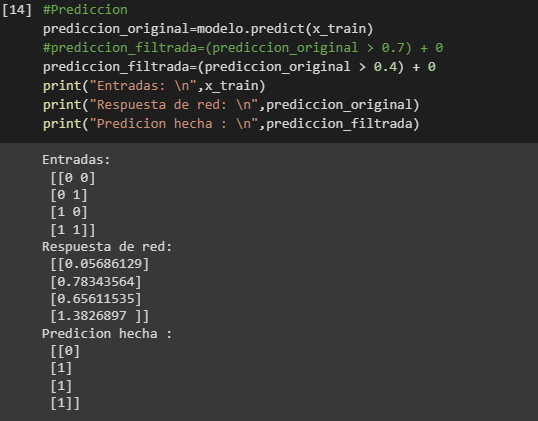


Figura 20. Predicción de la Red Adaline OR

Al observar las predicciones hechas se puede ver que son correctas , pero para visualizar cuanto loss frente a las épocas que se dieron , y presentar confiabilidad se muestra la comparación a traves de tensorboard .

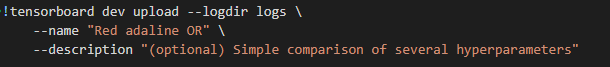


Figura 21. Visualización en Tensorboard

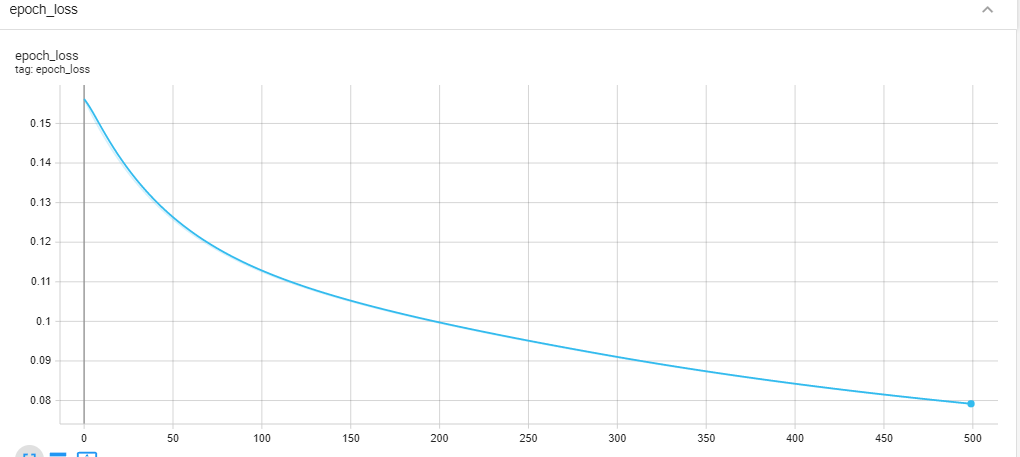


Figura 22. Loss vs epochs

En la anterior figura se observa como al no tener una capa oculta la red Adaline pudo realizar la predicción de la compuerta OR , dando resultados satisfactorios .

Red multicapa superficial , para la realización de esta red se debe poner una capa oculta en la cual se evidencia el cambio de capas, a diferencia de la adaline que no tenia capa oculta.

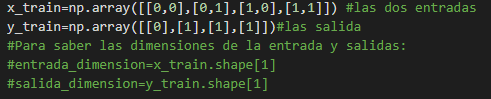
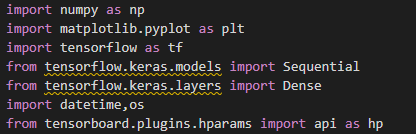


Figura 23. importación de librería y configuración red

Se procede a poner dos neuronas de entrada , con 16 neuronas en la capa oculta con función de activación relu y una neurona de salida con función de activación sigmoid, con una tasa de aprendizaje (learning\_rate) de 0.1

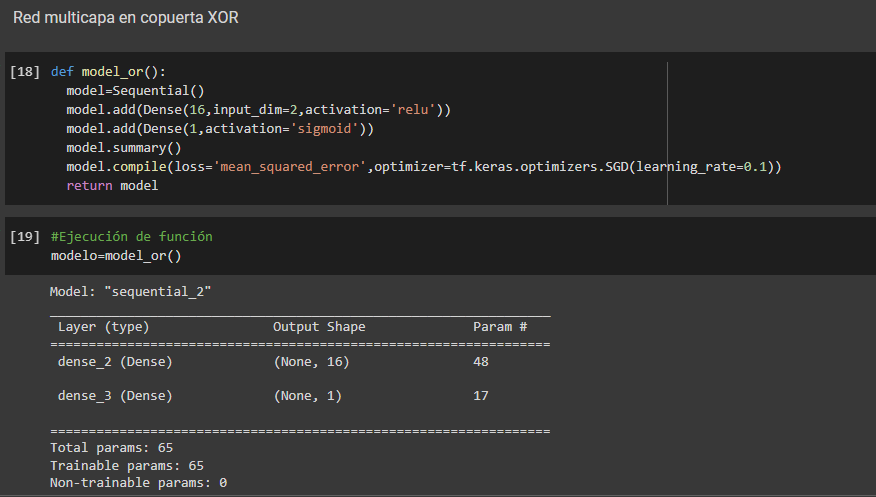


Figura 24. Configuración de arquitectura de Red multicapa superficial OR

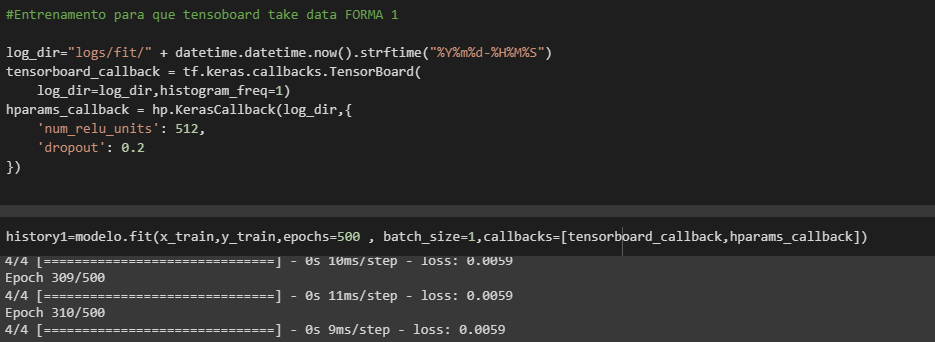


Figura 25. Configuración de tensoboard y entrenamiendo de modelo

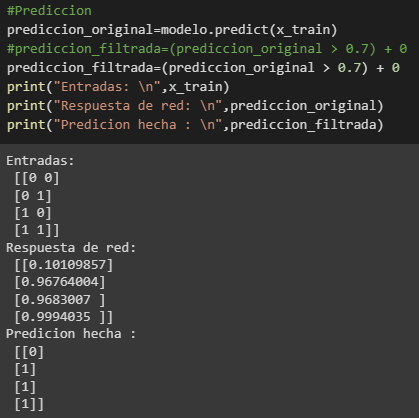


Figura 26. Predicción de la Red multicapa superficial OR con parámetro de aprendizaje de 0.1

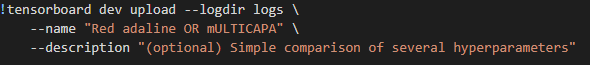


Figura 27. Visualización en Tensorboard de Red multicapa superficial OR

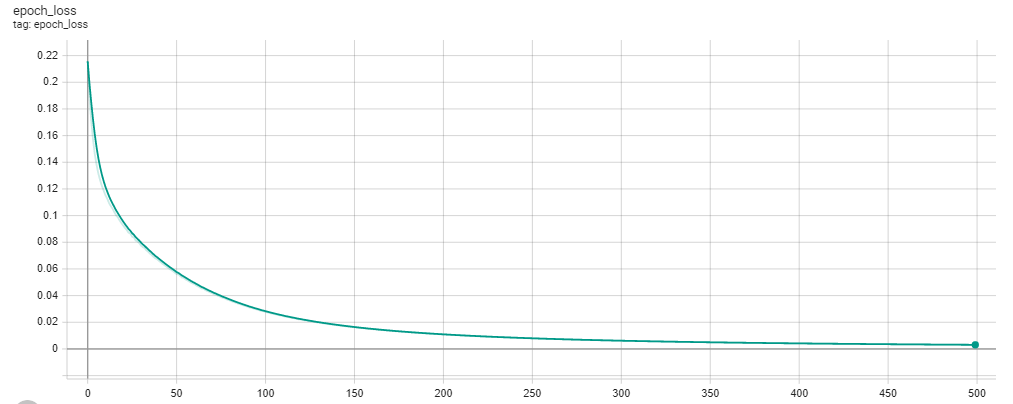


Figura 28. Loss vs epochs con parámetro de aprendizaje de 0.1

Ahora se comienza a modificar solo el learning\_rate del codigo con diferentes valores, en este caso con 0.01.

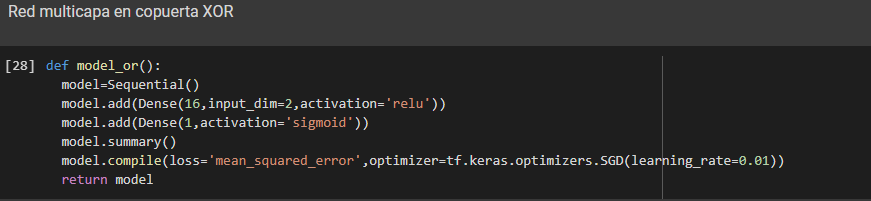


Figura 29. Modificación de parámetro de aprendizaje a 0.01

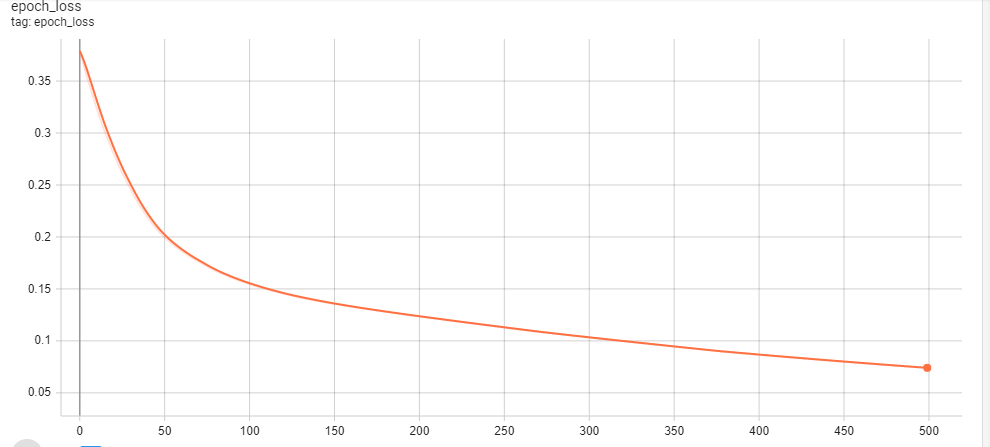


Figura 30. Loss vs epochs con parámetro de aprendizaje de 0.01

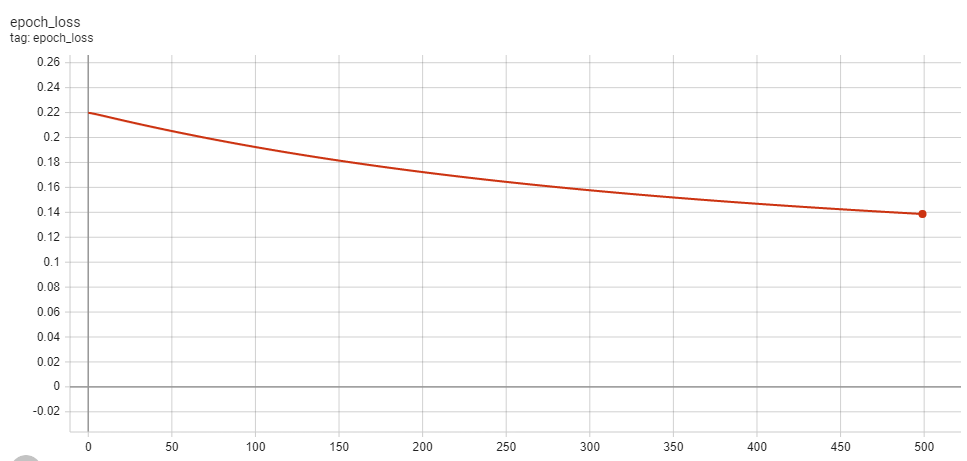


Figura 31. Loss vs epochs con parámetro de aprendizaje de 0.001

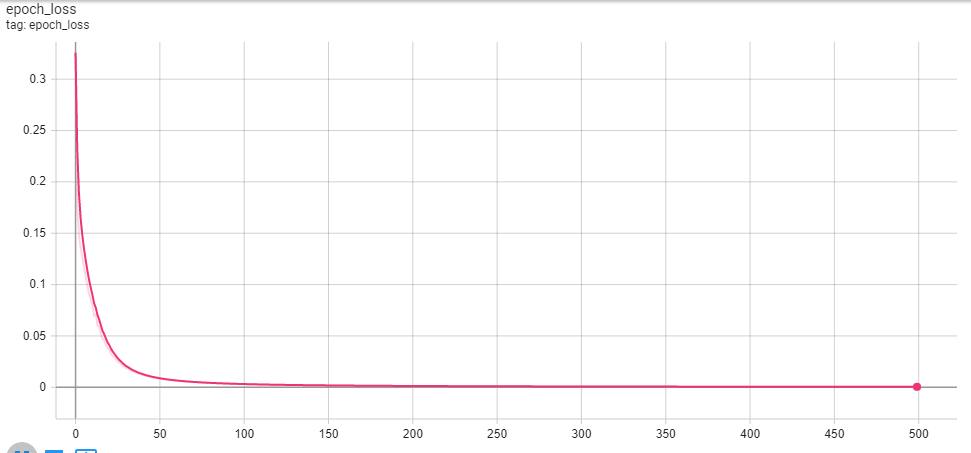
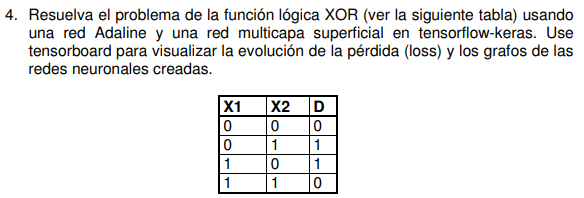


Figura 32 . Loss vs epochs con parámetro de aprendizaje de 0.5

Al observar la figuras de las modificaciones del parámetro de aprendizaje , entre mas pequeño sea el parámetro de aprendizaje genera mayor error , pero si este es demasiado grande puede ocurrir también errores grandes , por lo que se debe tener prioridad a valores cercanos como lo fueron el 0.5 , 0,1 y 0,01 , además comparando la red adaline y la red multicapa superficial se puede decir para este dicho problema de las compuertas OR no es necesario utilizar capas oculta si no la mejor opción de solución de desarrollo es la adaline ,



Red Adaline en XOR , se empieza a poner la librerías necesarias para desarrollar dicha red

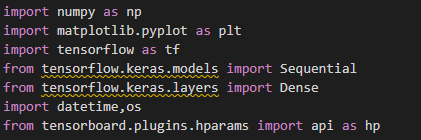


Figura 33. importación de librería necesarias para la ejecución de la Red Adaline XOR

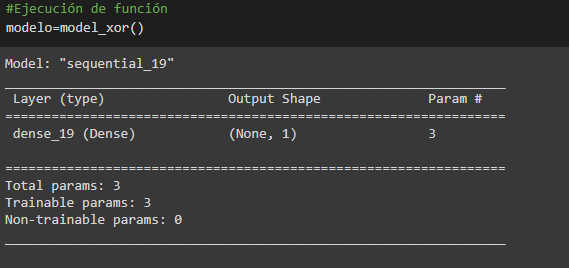
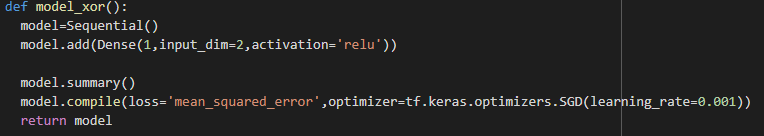
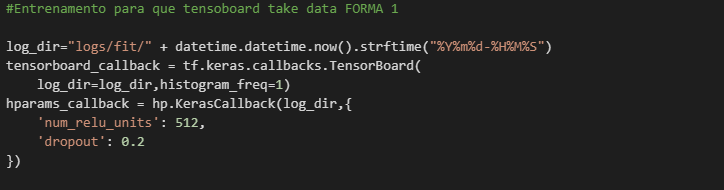


Figura 34. Configuraciones de arquitectura de Red Adaline XOR



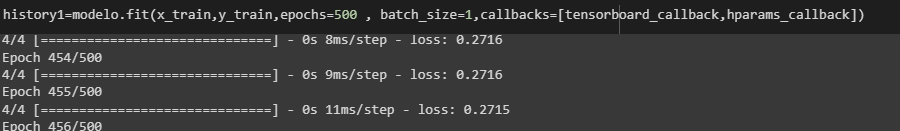


Figura 35. Configuración de tensoboard y entrenamiento de modelo para XOR

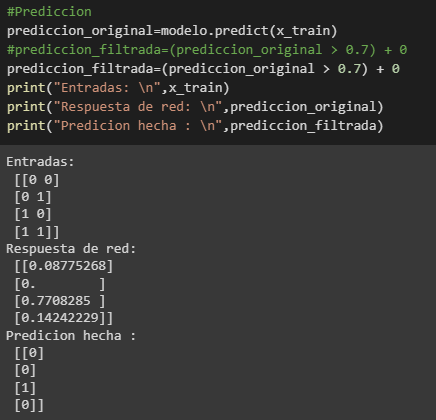


Figura 36. Predicción de la Red Adaline XOR

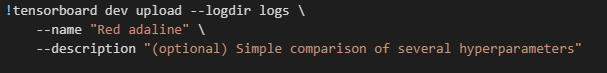


Figura 37. Visualización en Tensorboard de Red Adaline XOR

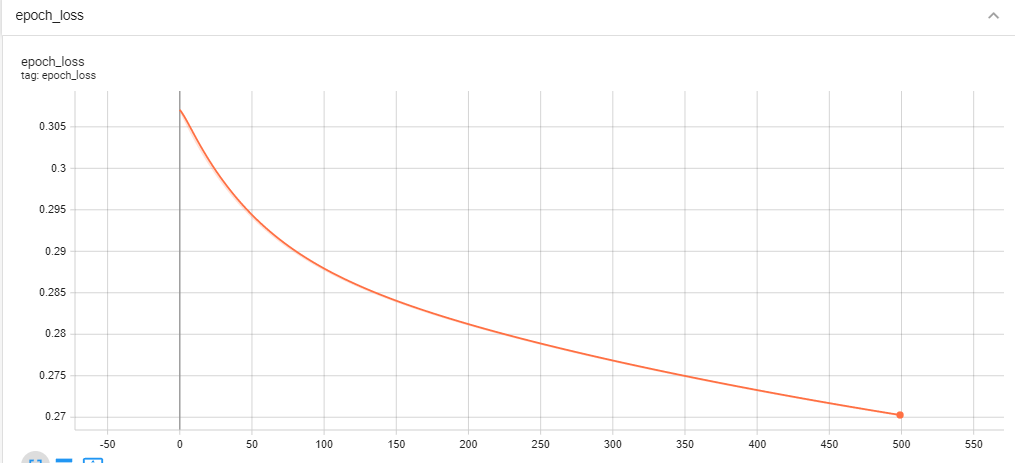


Figura 38. Loss vs epochs de Red Adaline XOR

Al observar la anterior figura se evidencia que al desarrollar una red adaline para la compuerta XOR es muy complejo dado una perdida mayor y haciendo que la predicción no sea exacta; Dejando que la aplicación de la red adaline en este tipo de problema no es debida.

Red multicapa superficial para la compuerta XOR

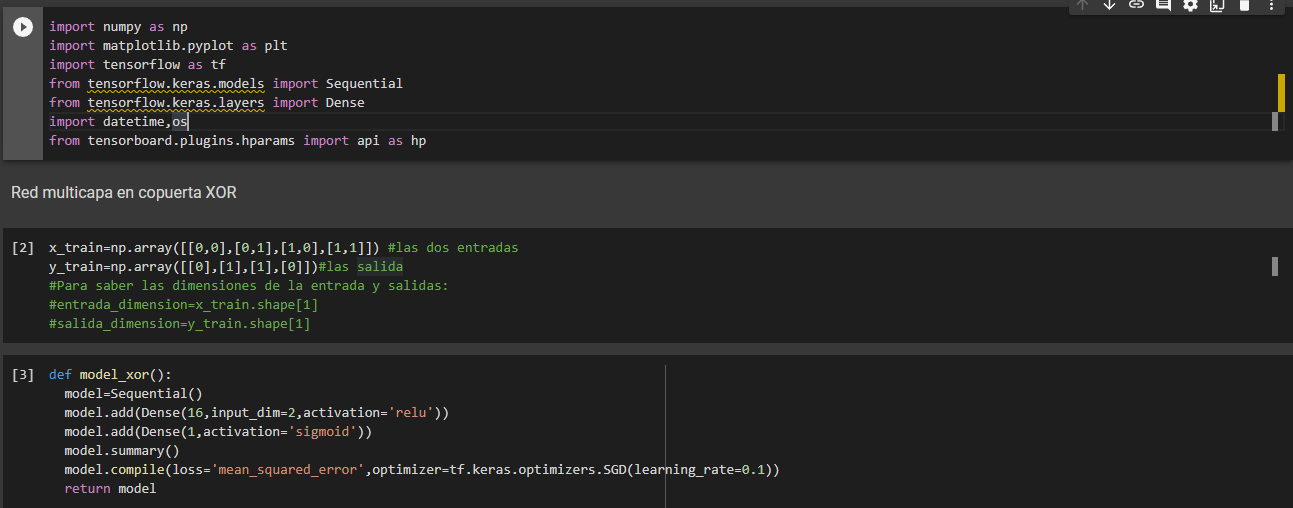


Figura 39. Importación de librería necesarias y Configuraciones de arquitectura de Red multicapa XOR



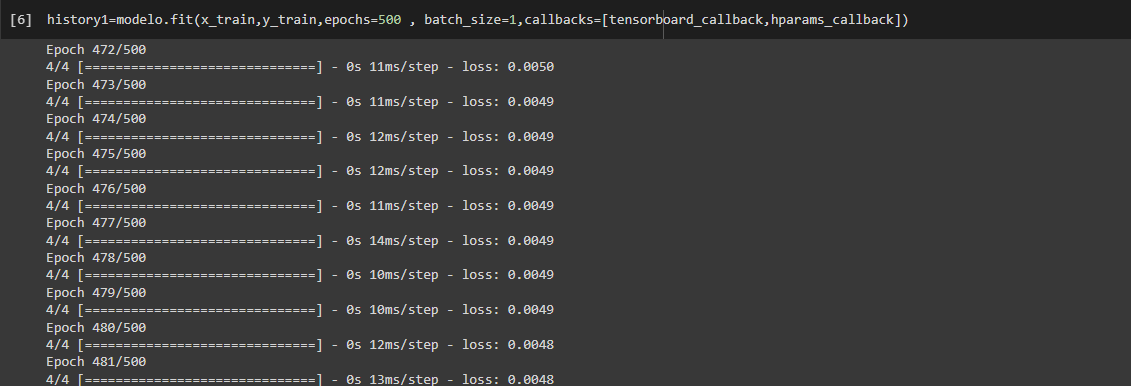


Figura 40. Configuración de tensoboard y entrenamiento de modelo multicapa para XOR

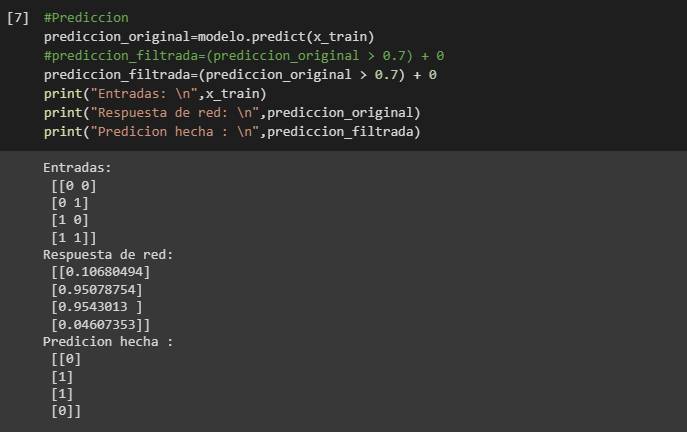


Figura 41. Predicción de la Red multicapa XOR

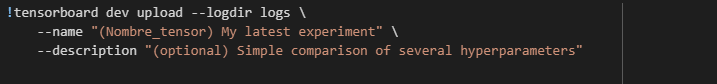


Figura 42. Visualización en Tensorboard de Red multicapa XOR

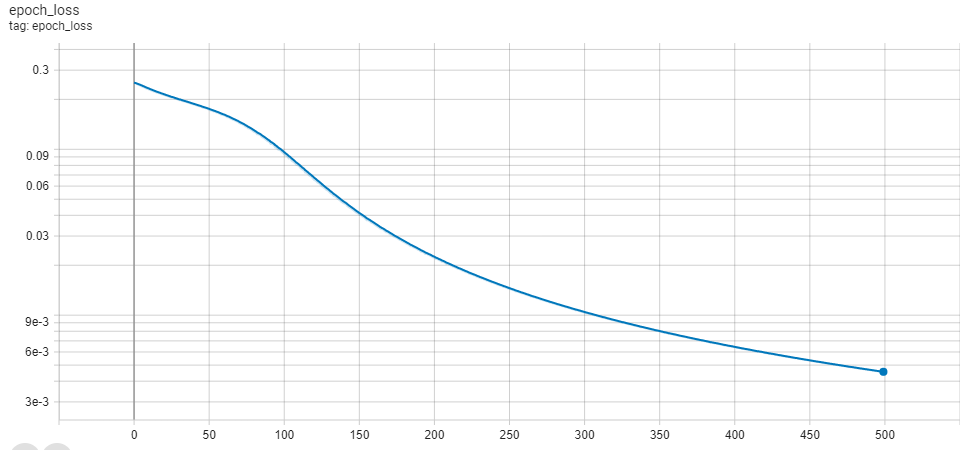
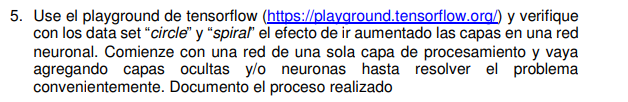


Figura 43. Loss vs epochs de Red multicapa XOR

Al observar la anterior figura demuestra la predicción y exactitud de esta red multicapa , prediciendo perfectamente los resultados y dando a tener en cuenta que este tipo de problema como lo es la compuerta XOR es de un grado de complejidad mas alto que la compuerta OR por lo que a utilizar capas ocultas disminuye inmensamente el error dando el resultado peregrinamente.



En la figura 44 se observa que hay 2 entradas , con dos neuronas en dos capas oculta , en el cual se esta utilizando una tasa de aprendizaje del 0.03 y función de activación tangencial hiperbólica simbolizada con (tanh) , en tipos de problemas de clasificación .

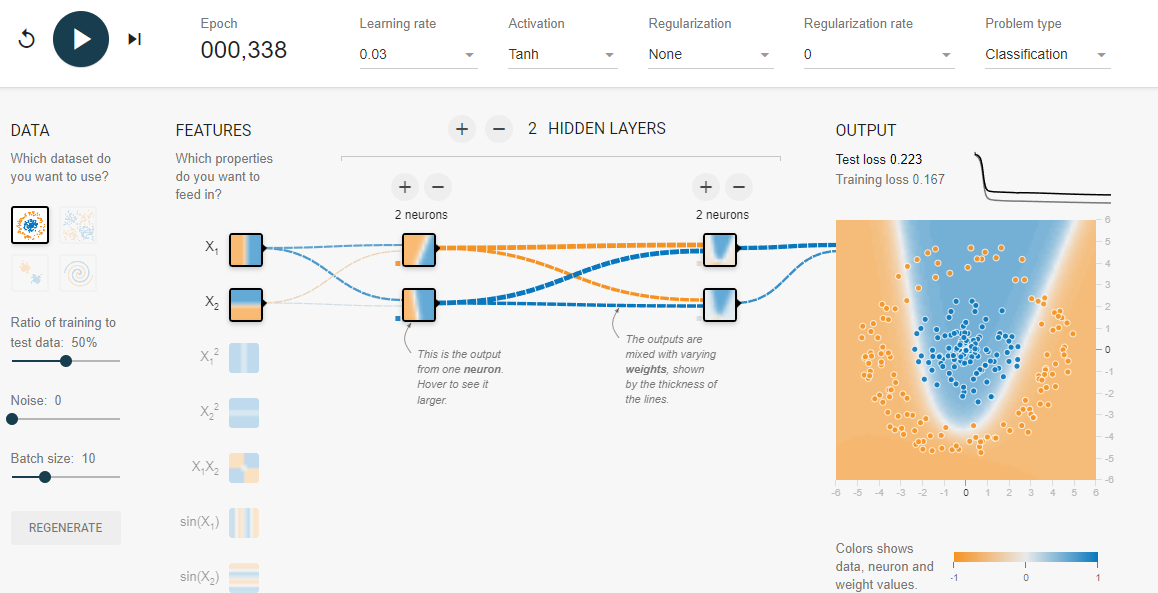


Figura 44. Data set circle con dos capas oculta.

Aumentando la capas ocultas y numero de neuronas se clasifica de manera mas acertada que con dos capas , es decir aumenta la precisión de la clasificación , tal como se observa en la figura 45.

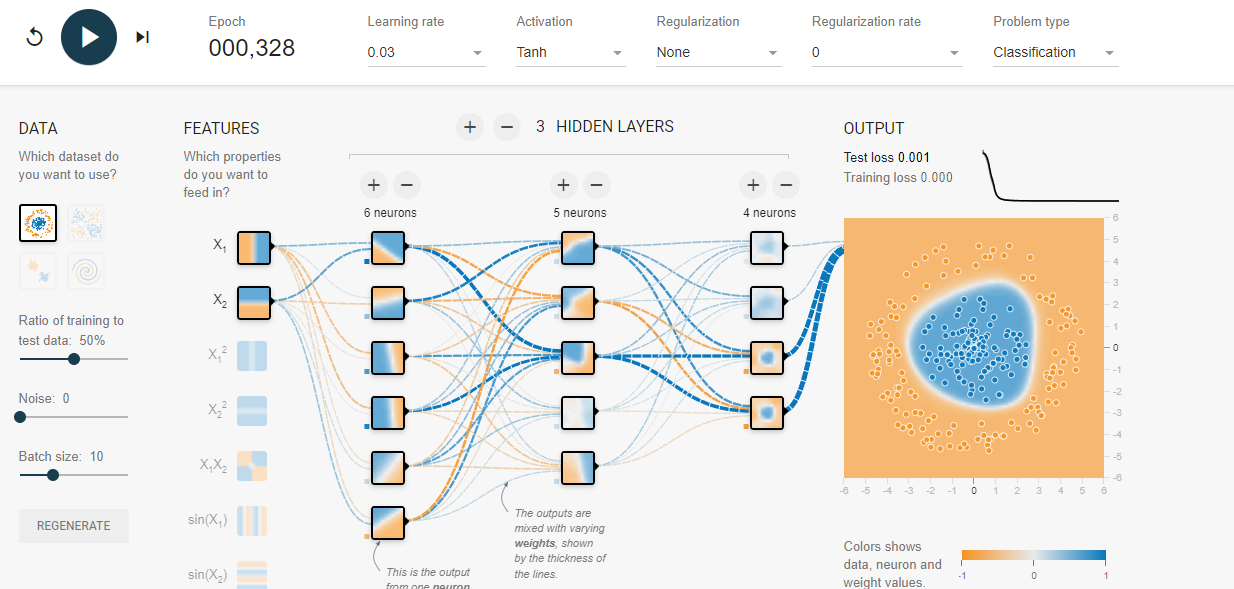


Figura 45. Data set circle con cuatro neuronas capa oculta.

Al mostrar el data set del spiral los puntos azules y amarillos esta deforma dispersa muy diferente como lo estaba en el data set de circle , por que las capas y numero de neuronas deben cambiar debido a la diferente aplicación. Se realizó el mismo tasa de aprendizaje y función de activación que con la hecha en l data set de circle.

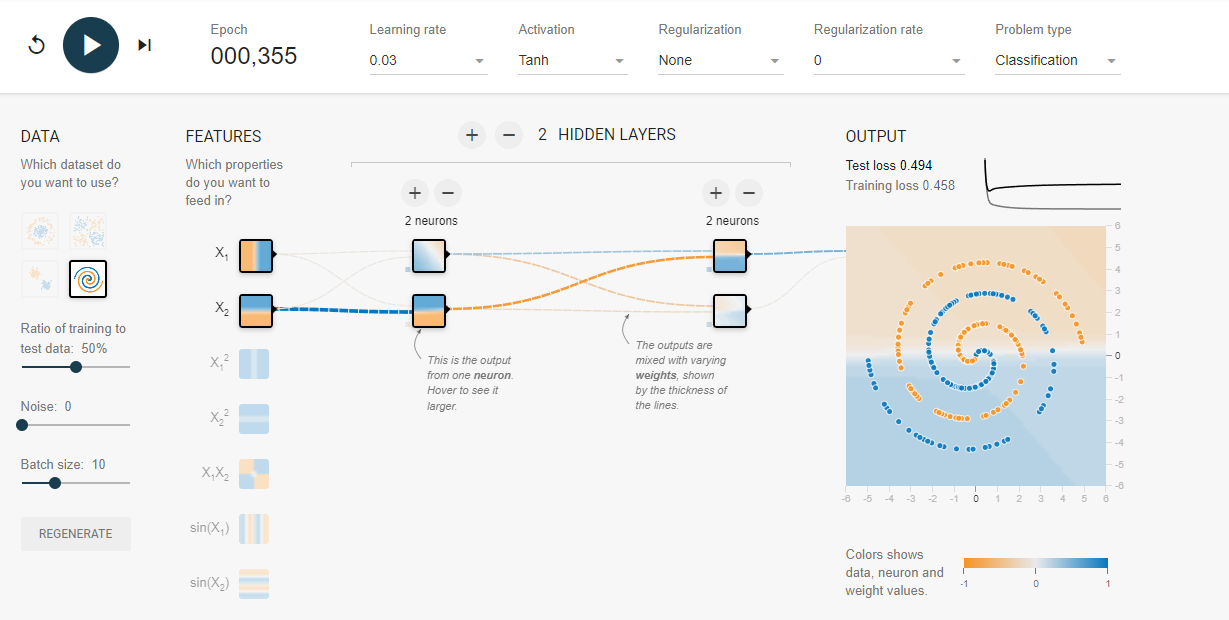


Figura 46. Data set spiral con dos neuronas en dos capas ocultas

Al aumentar la capas ocultas y el numero de neuronas en esas capas , en el mismo tiempo de acción no alcanza para clasificar los elementos, se necesita mas tiempo debido a que las entradas son lineales para conseguir dicha clasificación.

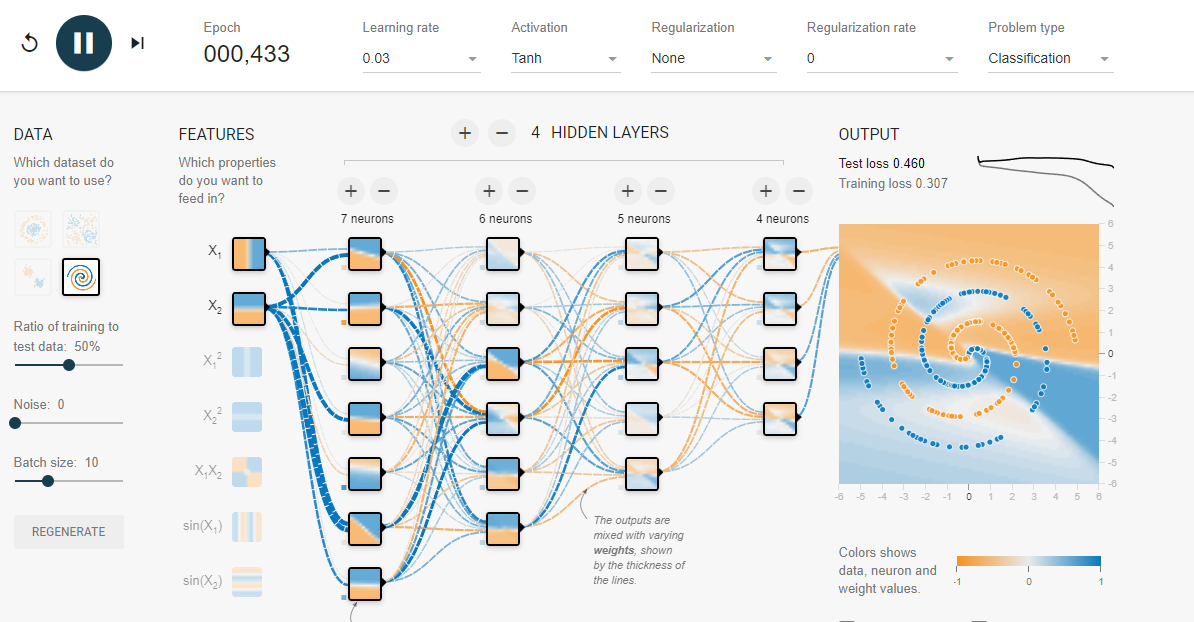


Figura 47. Data set spiral con aumento de capas ocultas y numero de neuronas.

Con usar entradas no lineales se puede mejorar la clasificación en un menor tiempo y menor numero de neuronas , también hay que tener en cuenta la función de activación de aplicar debido a que esto mejorara la clasificación del data set , como se observa en la figura 49 .

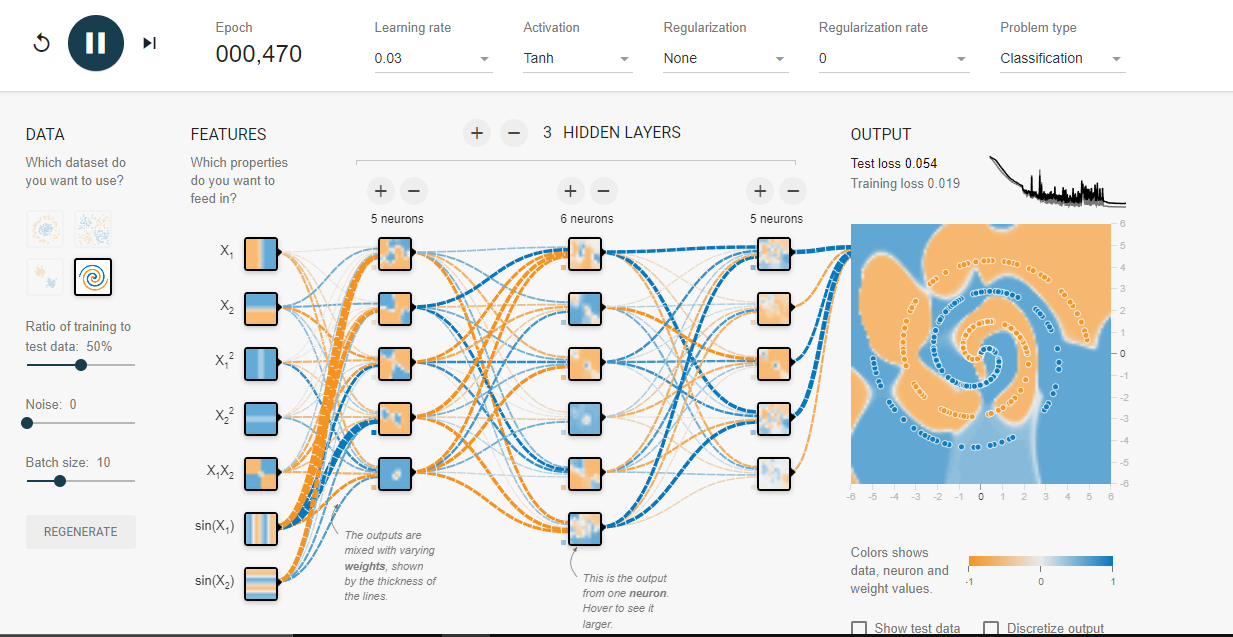


Figura 48. Data set spiral con entradas no lineales y lineales

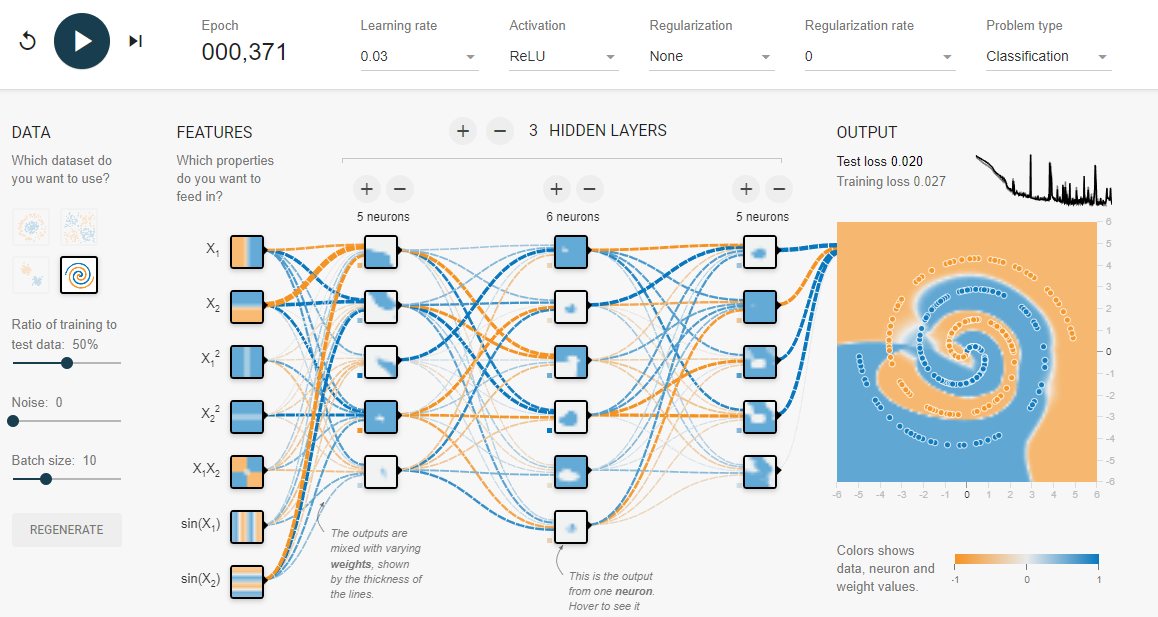
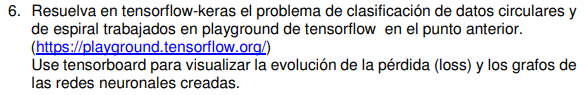


Figura 49. Data set spiral con función activación RELU



Se inició con el dataset del circle

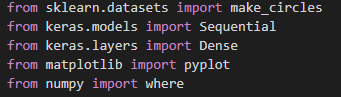


Figura 50. Importaciones de librerías necesarias dataset circle

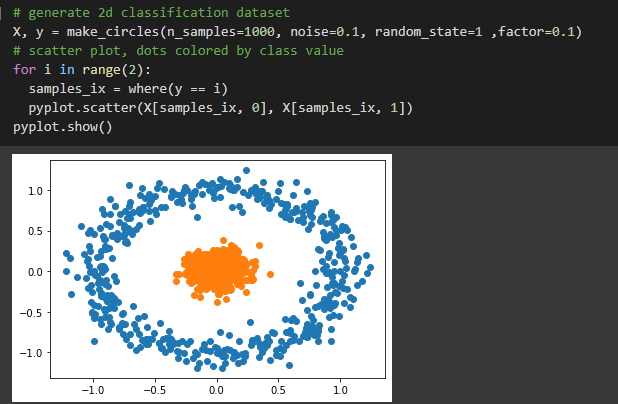


Figura 51. Configuraciones y generación de puntos de circulo 2d, dataset circle

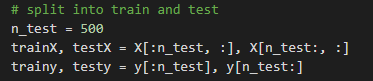


Figura 52. Separación de datos para entrenamiento y testeo , dataset circle

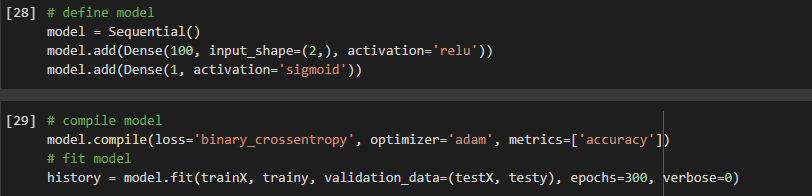


Figura 53. Definición del modelo y ajuste para entrenamiento, dataset circle

Se evalúa el modelo, informando la precisión de clasificación en el tren y conjuntos de prueba de alrededor en donde se observa que dio 100% en ambos.

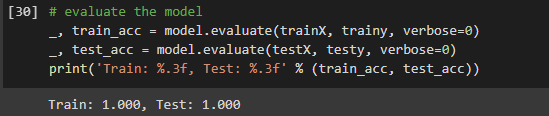


Figura 54. Evaluación del modelo, dataset circle

Después de la evaluación del modelo indica el buen porcentaje de satisfacción para el aprendizaje , para observar el los y la acuracy se grafican

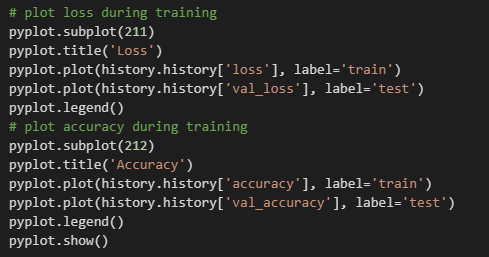


Figura 55. Código de graficar los y acurracy, dataset circle

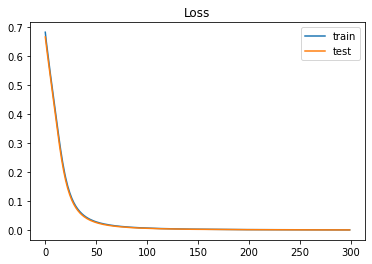


Figura 56. Grafica los vs epochs, dataset circle

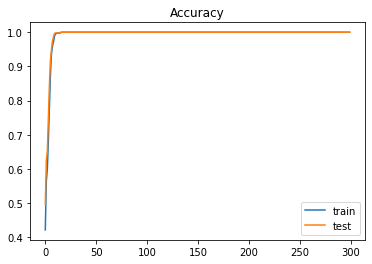


Figura 57. Grafica accuracy vs epochs, dataset circle

Al observar los anteriores gráficos se precisa que la elección de la arquitectura del modelo fue la ideal para resolver el problema dando que el valor del aprendizaje sea muy bueno en el transcurso de las épocas y no desarrollarse un overfitting que puede ser probable en dichos problemas o al momento de realizar aprendizajes.

Ahora se realiza el proceso con el spiral dataset

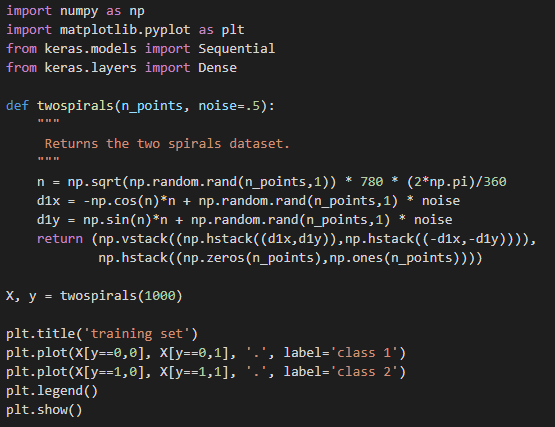


Figura 58. Importe de librerías y creación de spiral dataset

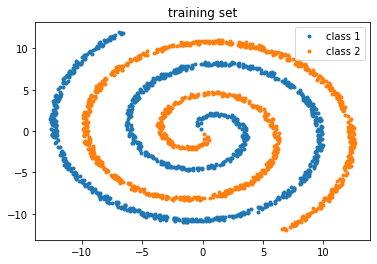


Figura 59. Visualización de spiral data set

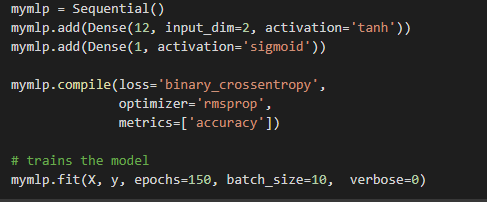


Figura 60. Creación de arquitectura y entrenamiento del modelo

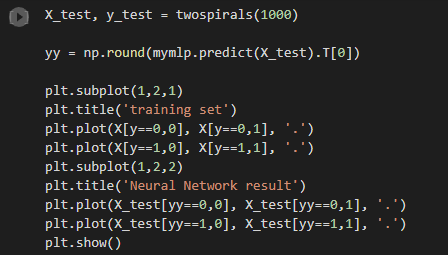


Figura 61. Código de visualización de comparación del training set y el resultado de la red neuronal

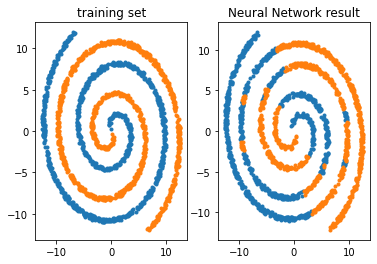


Figura 62. Resultado de la red neuronal entrenada con una capa oculta de 12 neuronas

Contemplando la anterior figura la red neuronal entrenada no es lo suficiente buena para resolver le problema de clasificar los puntos , esto es debido a que las capa ocultas y numero de neuronas no son lo suficientes para abordarlo , por eso se procede aumentar el numero de capas ocultas y numero de neuronas

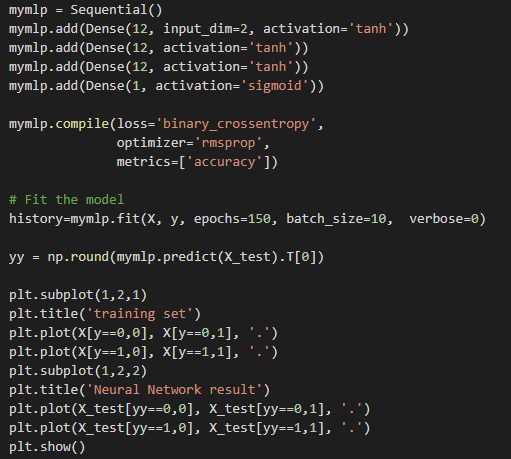


Figura 63. Aumento de numero de capas ocultas y numero de neuronas

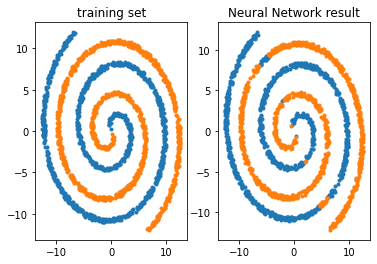
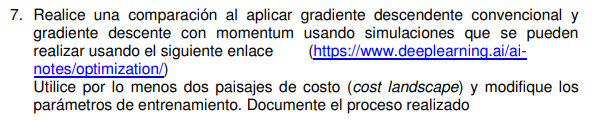


Figura 64. Resultado de la red neuronal entrenada con tres capas ocultas de 12 neuronas



Figura 65. Grafica de los vs epochs

La anterior figura se denota la solución abarcada al data set spiral de forma satisfactoria debido a que el los frente a 150 épocas realizadas presento un loss muy bajo , dando un entrenamiento ideal para la solución del entrenamiento. Por lo que se puede fijar como es por dentro de forma de keras lo que se realizaba en el punto 5 ya realizado



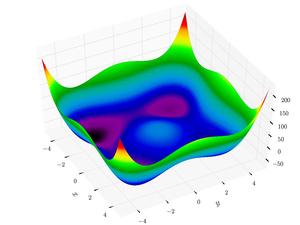


Figura 66. Paisaje seleccionado de costo 1

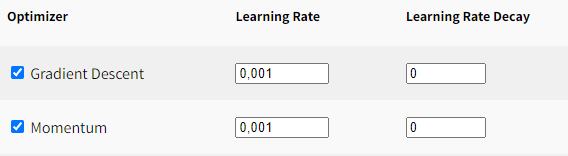


Figura 67. Tasa de aprendizaje de 0.001 en descenso de gradiente y momentum, paisaje costo 1

Para el momentum (impulso) y el descenso de gradiente se usó una tasa de aprendizaje de 0.001 , debido al momento se dieron menos épocas para que se estabilice la red y entrenarla, como se observa en la siguiente figura

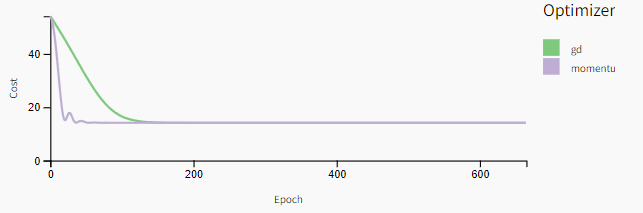


Figura 68. Cost vs epoch de descenso de gradiente y momentum, paisaje de costo 1

Para visualizar una mejor forma el comportamiento de la acción del desplazamiento frente a las diferentes parámetros de w1,w2 , está presente en la consecutiva figura

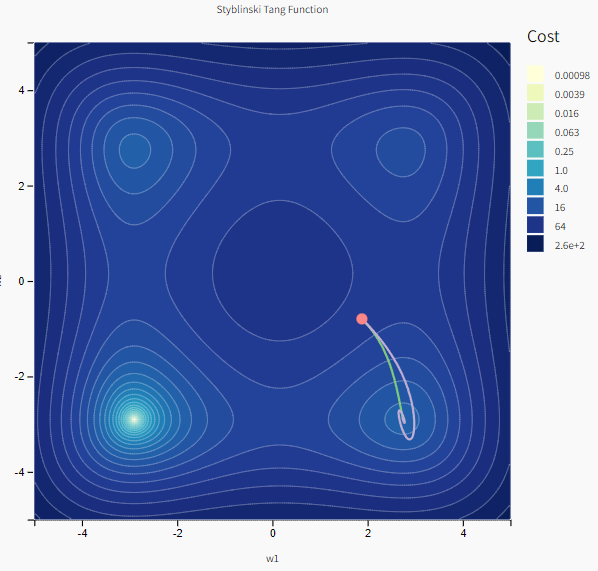


Figura 69. Simulación 2D de Costos según parámetros (w1,w2) , paisaje costo 1

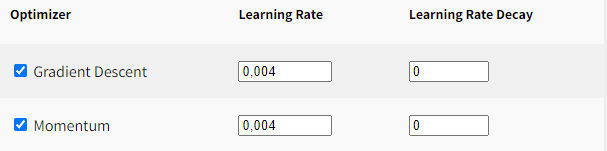


Figura 70. Tasa de aprendizaje de 0.004 en descenso de gradiente y momentum, paisaje costo 1

Al tener las diferentes figuras, para analizar el comportamiento del paisaje de costo 1 se aumenta la tasa de aprendizaje a 0.004 en el descenso de gradiente y el momentum , en el que el cambio no es evidenciable por lo que proporcionaron las mismas épocas para que se estabilizara cada uno de los parámetros , esto se refleja en la siguiente figura.

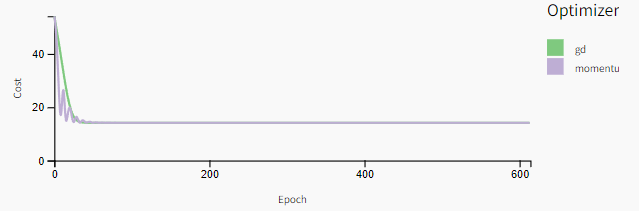


Figura 71. Cost vs epoch de descenso de gradiente y momentum, paisaje de costo 1 , tasa de aprendizaje de 0.004

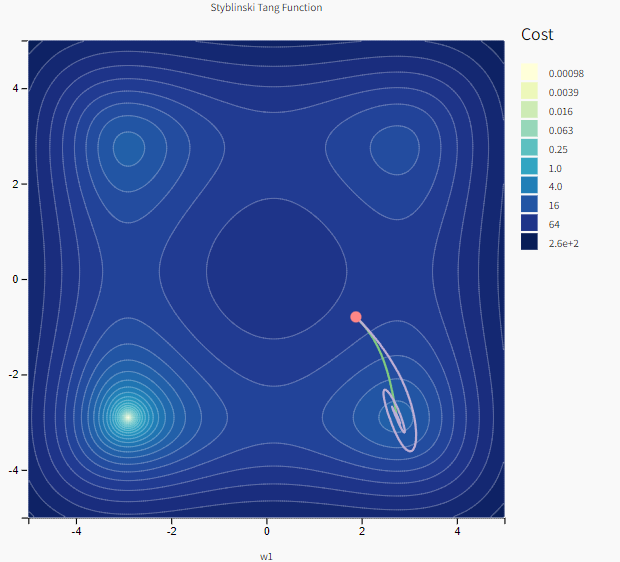


Figura 72. Simulación 2D de Costos según parámetros (w1,w2) con tasa de aprendizaje de 0.004 , paisaje costo 1

De forma de análisis se dispone dar selección de un diferente paisaje

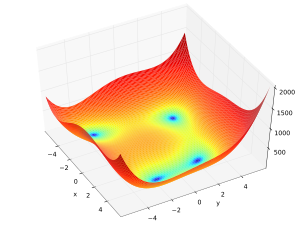


Figura 73. Paisaje seleccionado de costo 2

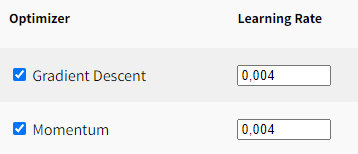


Figura 74. Tasa de aprendizaje de 0.004 en descenso de gradiente y momentum, paisaje costo 2

Al proporcionar una tasa de aprendizaje de 0.004 ubicada para el descenso de gradiente y el momentum(impulso) se demandaron menos épocas para que se estabilizara la red y que se entrenara, pero esto es debido a un paisaje diferente, por lo que proporcionado esa tasa de aprendizaje de 0.004 fueron parejas, y tuvo menos tiempo el momentum para que se estabilizara en comparación con el descenso de gradiente.

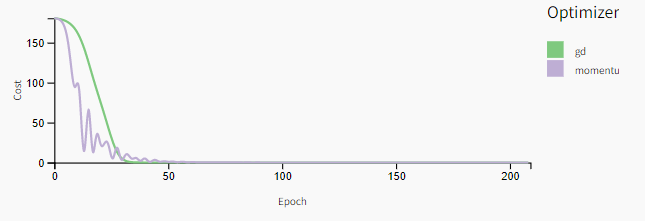


Figura 75. Cost vs epoch de descenso de gradiente y momentum, paisaje de costo 2 , tasa de aprendizaje de 0.004

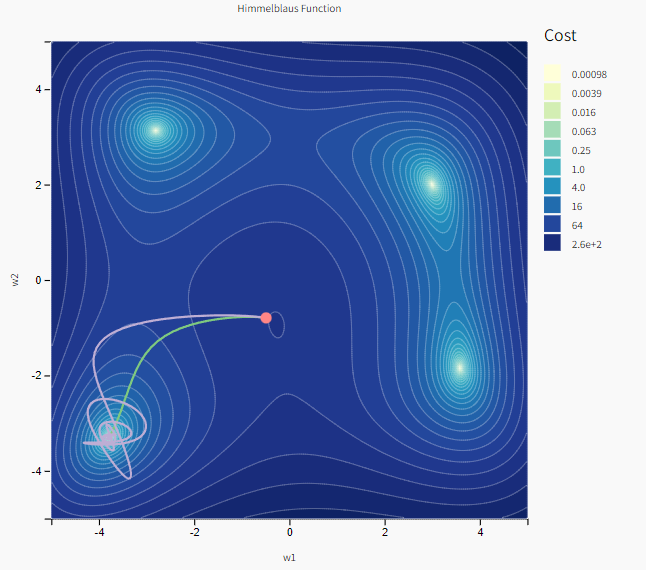


Figura 76. Simulación 2D de Costos según parámetros (w1,w2) con tasa de aprendizaje de 0.004 , paisaje costo 2

Al tener las diferentes figuras, para analizar el comportamiento del paisaje de costo 2 se aumenta la tasa de aprendizaje a 0.01 en el descenso de gradiente y el momentum.

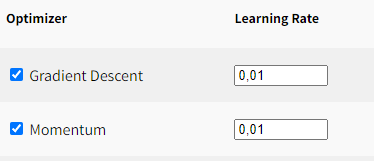


Figura 77. Tasa de aprendizaje de 0.01 en descenso de gradiente y momentum, paisaje costo 2

Prestando atención en la modificación el momentum adquirió más épocas para poder estabilizar la red y entrenarla , comparado con el descenso de gradiente; Se podría decir que en este tipo de paisaje al aumentar la tasa de aprendizaje en el descenso de gradiente se disminuye las épocas de entrenamiento.

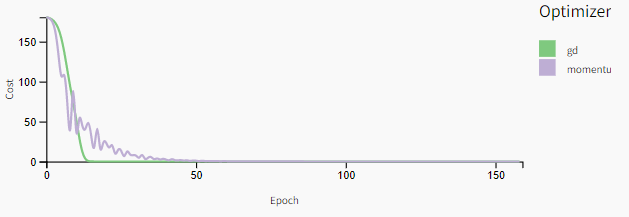


Figura 78. Cost vs epoch de descenso de gradiente y momentum, paisaje de costo 2 , tasa de aprendizaje de 0.01

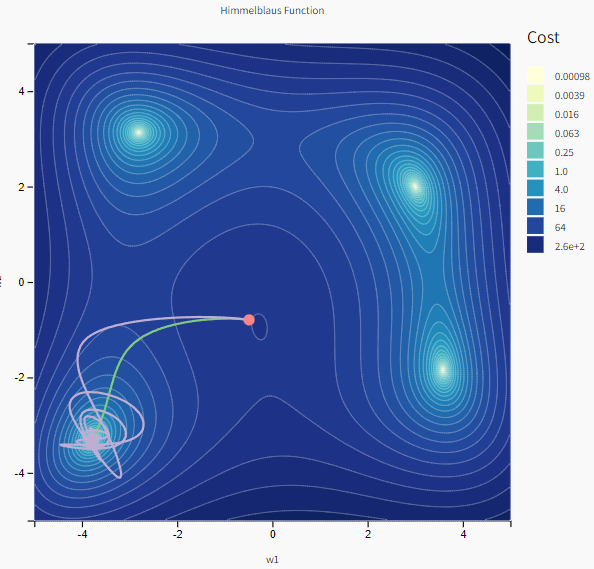
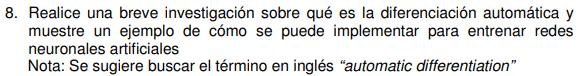


Figura 79. Simulación 2D de Costos según parámetros (w1,w2) con tasa de aprendizaje de 0.01 , paisaje costo 2



Diferenciación automática (AD) es un término usado en general describir las diversas técnicas numéricas para calcular las derivadas de una función de una o más variables. Mientras que las reglas para la diferenciación son sencillos de entender, su implementación a mano es a menudo requiere mucho tiempo y es propenso a errores. Las técnicas de AD ayudan a superar ambos estos inconvenientes y ofrecer al usuario involucrado en el modelado no lineal y optimización la promesa de información derivada precisa y correcta con un esfuerzo aparentemente mínimo.[9]

Por lo que la diferenciación automática es un método entre la diferenciación simbólica y la diferenciación numérica; La diferenciación numérica calculada una aproximación a la derivada de una función en un punto utilizando los valores y propiedades de la misma

La diferenciación simbólica esta dado en el algebra como resolución directamente y se sustituye en el valor del problema , por lo que la diferenciación automática aplica la diferenciación simbólica a los operadores como constantes , funciones de potencia , funciones exponenciales , funciones logarítmicas , funciones trigonométricas entre otros ; Siendo esta la diferentes foras mas básicas , después se sustituyen en valores numéricos manteniendo los resultados intermedios y finalmente se aplican a toda la función ; Por lo tanto su aplicación es bastante flexible y puede ocultar completamente a los usuarios el proceso de resolución diferencial. Puesto a que se aplica la ley de diferenciación simbólica a funciones basias o constantes , puede combinar de manera flexible la estructura condicional de l leguaje de programación , utilizando diferenciación automática y debido a que su calculo s grafico , se puede optimizar bastante siendo la razón por lo que se usa ampliamente en los sistemas modernos de aprendizaje profundo , una clara idea de saber sobre esta tema y de su funcionamiento.

Hay dos enfoques de programación diferentes para diferenciar código: sobrecarga de operadores y transformación fuente a fuente. En el enfoque de sobrecarga del operador, las operaciones aritméticas básicas y A las funciones se les asignan rutinas que calculan las derivadas de los operadores. salidas además del cálculo del valor de la función. el código fuente de la función se diferencia progresivamente llamando a estas rutinas al mismo tiempo que se realiza cada operación en la evaluación del programa. Existen dos enfoques para la diferenciación de programas: reenvío automático diferenciación automática (FAD) y diferenciación automática inversa (RAD).

La diferenciación automática inversa (RAD) está relacionada matemáticamente con el análisis de sensibilidad adjunto para ecuaciones diferenciales, cuyo uso también se remonta a la década de 1960. El método fue entonces ampliamente utilizado en ingeniería nuclear. [10] y predicción meteorológica [11], por ejemplo. El algoritmo RAD funciona en términos del gráfico computacional asociado con la evaluación de un función. Al igual que FAD, para implementar el algoritmo RAD requerimos la declaración de una nueva variable escalar conocida como la variable adjunta t¯ para cada nodo del gráfico computacional.

Ventajas de FAD

Como FAD, RAD proporciona derivadas matemáticamente precisas. El algoritmo RAD es eficiente en términos del número de pasos computacionales. Esto hace que RAD sea especialmente adecuado para problemas con un gran número de variables independientes. El método RAD también es eficiente con respecto a los requisitos de memoria. En particular, el requisito de memoria es independiente del número de variables.

Desventajas de FAD

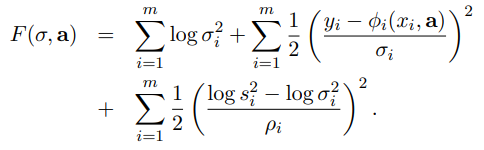
El algoritmo RAD requiere la modificación del código fuente por parte del usuario. De manera similar a FAD, la modificación solo implica cambiar la parte declarativa del código fuente, mientras que el resto del código permanece sustancialmente igual. Para implementar RAD, el usuario también debe declarar un nuevo tipo de variable y los operadores de sobrecarga asociados con este nuevo tipo. Las implementaciones del algoritmo RAD implican listas enlazadas, árboles binarios, punteros y una cuidadosa gestión de la memoria. Requiere un lenguaje que admita la sobrecarga de operadores, como Fortran 90.

Ejemplo de aplicación: estimación de máxima verosimilitud

Los problemas de mínimos cuadrados no lineales ponderados surgen cuando las incertidumbres Si en los datos de entrada se han caracterizado adecuadamente. Si solo tenemos estimaciones aproximadas si para σi , eso puede ser apropiado considerar las σi como incógnitas pero para las cuales tenemos algún modelo estadístico. Por ejemplo, supongamos que el modelo es:



donde ρi es conocida. La estimación de máxima verosimilitud de σ = (σ1, . . . , σm) T y a se determina minimizando una función logarítmica de verosimilitud de la forma



Software de optimización como el componente E04LBF en la biblioteca NAG [12] requiere que calculemos el vector gradiente de las primeras derivadas y el Matriz hessiana de segundas derivadas de F con respecto a σ ya. Nosotros han calculado estos utilizando el algoritmo RAD implementado en Fortran 90.

**Bibliografía**

[1]"Una sencilla Red Neuronal en Python con Keras y Tensorflow | Aprende Machine Learning", *Aprendemachinelearning.com*, 2022. [Online]. Available: https://www.aprendemachinelearning.com/una-sencilla-red-neuronal-en-python-con-keras-y-tensorflow/. [Accessed: 28- Aug- 2022]

[2]"Keras: mi primer modelo de clasificación binaria de redes neuronales - programador clic", *Programmerclick.com*, 2022. [Online]. Available: https://programmerclick.com/article/83031913685/. [Accessed: 28- Aug- 2022]

[3]"Deep Learning với Tensorflow Module 2 Phần 1: Xây dựng mô hình phân loại trong Neural network - Mô hình 2 class (binary class)", *Mvt-blog.com*, 2022. [Online]. Available: https://mvt-blog.com/posts/deep-learning-with-tensorflow-module-3-part-1-neural-network-classification. [Accessed: 28- Aug- 2022]

[4]2022. [Online]. Available: https://machinelearningmastery.com/impact-of-dataset-size-on-deep-learning-model-skill-and-performance-estimates/. [Accessed: 28- Aug- 2022]

[5]2022. [Online]. Available: https://machinelearningmastery.com/how-to-calculate-precision-recall-f1-and-more-for-deep-learning-models/. [Accessed: 28- Aug- 2022]

[6]*Youtube.com*, 2022. [Online]. Available: https://www.youtube.com/watch?v=HMsMUg1BzzY. [Accessed: 28- Aug- 2022]

[7]"Solving the Two Spirals problem with Keras", *Glowingpython.blogspot.com*, 2022. [Online]. Available: https://glowingpython.blogspot.com/2017/04/solving-two-spirals-problem-with-keras.html. [Accessed: 28- Aug- 2022]

[8]H. shape?, S. Nanda, F. Dernoncourt and H. Du, "How to classify data which is spiral in shape?", *Cross Validated*, 2022. [Online]. Available: https://stats.stackexchange.com/questions/235600/how-to-classify-data-which-is-spiral-in-shape. [Accessed: 28- Aug- 2022]

[9]*Eprintspublications.npl.co.uk*, 2022. [Online]. Available: https://eprintspublications.npl.co.uk/2828/1/cmsc26.pdf. [Accessed: 28- Aug- 2022]

[10]D. G. Cacuci. Sensitivity theory for nonlinear systems I: nonlinear functional analysis approach. Journal of Mathematical Physics, 22(12):2794–2802, 1981.

[11]I. M. Navon and U. Muller. FESW - a finite-element Fortran IV program for solving the shallow water equation. Advances in Engineering Software, 1:77–84, 1970.

[12]The Numerical Algorithms Group Limited, Wilkinson House, Jordan Hill Road, Oxford, OX2 8DR. The NAG Fortran Library, Mark 20, Introductory Guide, 2002. http://www.nag.co.uk/