Universidad Simón Bolívar

IA 2 - CI-5438

Miguel Perez 15-11126

Gabriel Chaurio 17-10126

Repositorio de Github

Proyecto 2: Redes neuronales

PROBLEMA

Las redes neuronales artificiales (ANN) son modelos matemáticos inspirados en el funcionamiento del sistema nervioso humano. Están formadas por una serie de neuronas interconectadas, cada una de las cuales realiza una función matemática simple. Al interactuar entre sí, las neuronas de una ANN pueden aprender a realizar tareas complejas, como el reconocimiento de patrones o la clasificación de datos.

El algoritmo de backpropagation es un algoritmo de aprendizaje supervisado que se utiliza para entrenar ANN. Este algoritmo funciona propagándose el error a través de la red, ajustando los pesos de las conexiones entre las neuronas para minimizar el error.

En este informe se presenta el desarrollo de una red neuronal con algoritmo de backpropagation para clasificación. El objetivo de este proyecto es demostrar la capacidad de las ANN para resolver problemas de categorización binaria y multi-clase.

IMPLEMENTACIÓN

El proyecto fue desarrollado con Python. La implementación de la red neuronal se encuentra en los archivos “network.py” y “neuron.py”. La red tiene la capacidad de recibir la cantidad de capas ocultas así como la cantidad de neuronas en cada capa. En el entrenamiento recibe la cantidad de iteraciones que hará así como la tasa de

aprendizaje. La función de test implementada para network, se utiliza luego de entrenar el modelo. Esta utiliza el test set guardado al hacer train_test_split cuando se entrena a la red, para probar el funcionamiento de los valores a los cuales la red converge, y verificar su correctitud; además de graficar el error y generar dataframes con la información utilizada en el análisis.

***Todos los gráficos generados para los errores en el entrenamiento por iteración, se encuentran en la carpeta graficos, asi como en testing.ipynb y spam.ipynb**

DESCRIPCIÓN DE LOS DATOS

- **Iris Dataset:** El archivo “iris.csv”, contiene información de 3 clases. Cada clase es un tipo de planta del género Iris. Las columnas contienen las características de cada muestra y el nombre de la especie. Se agregaron dos columnas al dataset y se modificó la columna de especies de forma que las últimas 3 columnas eran para las 3 especies y sus valores eran 0 o 1 dependiendo de sí la muestra pertenecía o no a la especie de la columna. Para este dataset, se hace un train_test donde se utiliza un 80% del dataset para entrenar y un 20% para testear.

- **Spam Dataset**: Este dataset está dividido en 3 archivos. Uno de nombres de columnas, uno con la data y otro con la documentación. El archivo de la data tiene 4601 filas de información relevante sobre la estructura de los correos electrónicos. La última columna indica si el correo electrónico fue considerado spam (1) o no (0), es decir, correo electrónico comercial no solicitado. La mayoría de los atributos indican si una palabra o carácter particular aparece con frecuencia en el correo electrónico. Los atributos de longitud de ejecución (55-57) miden la longitud de secuencias de letras mayúsculas consecutivas.

  Para este dataset, tambien se hace un train_test donde se utiliza un 80% del dataset para entrenar y un 20% para testear.

  A este conjunto de datos se le aplicó la siguiente normalización:

$$X = \frac{X - X_{min}}{X_{max} - X_{min}}$$

RESULTADOS DE LOS EXPERIMENTOS

- **Clasificadores Binarios:**

Species Classifier	Iters	Learning Rate	Mean Error	Max Error	Min Error	False Negative	False Positive	Positive	Negative
Iris-setosa	3000	0.1	0.00006763	0.0526894325	7.003e-7	0	0	10	20
Iris-setosa	6000	0.1	0.0000321524	0.045918465	2.131e-7	0	0	10	20
Iris-setosa	3000	0.01	0.0034606568	0.9989263672	0.0000305904	0	0	10	20
Iris-setosa	6000	0.01	0.0038081605	0.9997085864	0.0000103708	0	0	10	20
Iris-setosa-1CapOcul-2Neur	6000	0.01	0.328405674	0.9177222968	0.3269684806	10	0	0	20
Iris-setosa-1CapOcul-4Neur	6000	0.01	0.2808687383	0.9000302189	0.2396395667	8	0	2	20
Iris-setosa-1CapOcul-6Neur	6000	0.01	0.3198720789	0.9703187216	0.3157624448	10	0	0	20
Iris-setosa-2CapOcul-2_4Neur	6000	0.01	0.3281079578	0.9676997141	0.3267337082	10	0	0	20
Iris-setosa-2CapOcul-4_6Neur	6000	0.01	0.3317684104	0.9974323921	0.3011513953	10	0	0	20
Iris-versicolor	3000	0.1	0.5430917931	0.9998743705	0.3729742608	3	2	6	19
Iris-versicolor	6000	0.1	0.5886860153	0.9997917863	0.3784948179	3	3	6	18
Iris-versicolor	3000	0.01	0.4374144921	0.4578272732	0.3290970259	3	2	6	19
Iris-versicolor	3000	0.01	0.4374144921	0.4578272732	0.3290970259	3	2	6	19
Iris-versicolor-1CapOcul-2Neur	6000	0.01	0.3473636772	0.8723730092	0.3413559965	9	0	0	21
Iris-versicolor-1CapOcul-4Neur	6000	0.01	0.3501297489	0.7110661482	0.3396177465	9	0	0	21
Iris-versicolor-1CapOcul-6Neur	6000	0.01	0.3559359333	0.9939467928	0.3444043008	9	0	0	21
Iris-versicolor-2CapOcul-2_4Neur	6000	0.01	0.3462775501	0.986975301	0.3433132162	9	0	0	21
Iris-versicolor-2CapOcul-4_6Neur	6000	0.01	0.3448901612	0.8979914255	0.3445161302	9	0	0	21
Iris-virginica	3000	0.1	0.0059198497	0.7067408504	0.0003441864	0	1	11	18
Iris-virginica	6000	0.1	0.0029130803	0.5297151478	0.0000070689	0	1	11	18
Iris-virginica	3000	0.01	0.0625628117	0.4700075084	0.0000050438	0	1	11	18
Iris-virginica	6000	0.01	0.0388645272	0.4895184685	0.000033232	0	1	11	18
Iris-virginica-1CapOcul-2Neur	6000	0.01	0.6714460403	0.6727969866	0.2090545893	11	0	0	19
Iris-virginica-1CapOcul-4Neur	6000	0.01	0.6650536344	0.6711564925	0.0849863101	11	0	0	19
Iris-virginica-1CapOcul-6Neur	6000	0.01	0.6107608724	0.6684162767	0.0033441115	11	0	0	19
Iris-virginica-2CapOcul-2_4Neur	6000	0.01	0.670280293	0.6719330194	0.0277723969	11	0	0	19
Iris-virginica-2CapOcul-4_6Neur	6000	0.01	0.6694949635	0.6702302342	0.0476242646	11	0	0	19

Se crearon 3 clasificadores binarios (hablando en términos generales), uno para cada posible resultado, siendo estos:

- Iris Setosa
- Iris Versicolor
- Iris Virginica

Cada uno de estos clasificadores binarios fue entrenado bajo diversas condiciones referentes a cantidad de iteraciones, capas, neuronas por capa y learning rate.

Para cada modelo, se entrenó con variaciones de los siguientes valores:

- Iteraciones: 3000 y 6000 iteraciones.
- Learning Rate: 0.1 y 0.01
- Capas ocultas: 0, 1 y 2 capas ocultas.
- Neuronas: 2, 4, 6 y 8 neuronas.

El objetivo fue variar entre combinaciones de estos valores, para generar diversos casos de prueba y comprobar cuáles eran las mejores ante el problema planteado.

Casos Unineuronales, 0 Capas Ocultas:

	i	l	Species Classifier	Iters	Learning Rate	Mean Error	Max Error	Min Error	False Negative	False Positive	Positive	Negative
0	0	0	Iris-setosa	3000	0.1	0.00006763	0.0526894325	7.003e-7	0	0	10	20
1	1	0	Iris-setosa	6000	0.1	0.0000321524	0.045918465	2.131e-7	0	0	10	20
2	2	0	Iris-setosa	3000	0.01	0.0034606568	0.9989263672	0.0000305904	0	0	10	20
3	3	0	Iris-setosa	6000	0.01	0.0038081605	0.9997085864	0.0000103708	0	0	10	20
4	4	0	Iris-versicolor	3000	0.1	0.5430917931	0.9998743705	0.3729742608	3	2	6	19
5	5	0	Iris-versicolor	6000	0.1	0.5886860153	0.9997917863	0.3784948179	3	3	6	18
6	6	0	Iris-versicolor	3000	0.01	0.4374144921	0.4578272732	0.3290970259	3	2	6	19
7	7	0	Iris-versicolor	3000	0.01	0.4374144921	0.4578272732	0.3290970259	3	2	6	19
8	8	0	Iris-virginica	3000	0.1	0.0059198497	0.7067408504	0.0003441864	0	1	11	18
9	9	0	Iris-virginica	6000	0.1	0.0029130803	0.5297151478	0.0000070689	0	1	11	18
10	10	0	Iris-virginica	3000	0.01	0.0625628117	0.4700075084	0.0000050438	0	1	11	18

Para los casos de una neurona, podemos observar resultados muy positivos, para todos los clasificadores, teniendo en el peor de los casos solo un 20%(1 caso) de error para los 30 casos de prueba, el resto de los resultados expresa una tasa de error de 16.7%(3 casos), 3.3%(3) y 0%(3 casos) para los 30 casos respectivos en su train_test. Puesto que el peor valor de correctitud es del 80% para los casos de prueba, podemos concluir que estos clasificadores hacen un buen trabajo.

En base a los resultados, parece ser que en términos generales, es indiferente el learning rate y la tasa de iteración siempre y cuando se encuentren en estos rangos, esto es debido a que los clasificadores binarios están especializados en decir si un ejemplo pertenece o no a una sola categoría y debido a la buena distribución del conjunto de datos con respecto a los posibles resultados, el proceso de aprendizaje es ameno y converge rápidamente a un buen clasificador.

De hecho, podemos destacar que aquel con mayor tasa de error, es el clasificador de Iris-versicolor, y esto puede ser atribuido a características del dataset, no al modelo de aprendizaje.

Casos Unineuronales, 1 Capa Oculta:

Species Classifier	Iters	Learni...	Mean Error	Max Error	Min Error	False Negative	False Positive	Positive	Negative
Iris-setosa-1CapOcul-2Neur	6000	0.01	0.328405674	0.9177222968	0.3269684806	10	0	0	20
Iris-setosa-1CapOcul-4Neur	6000	0.01	0.2808687383	0.9000302189	0.2396395667	8	0	2	20
Iris-setosa-1CapOcul-6Neur	6000	0.01	0.3198720789	0.9703187216	0.3157624448	10	0	0	20
Iris-versicolor-1CapOcul-2Neur	6000	0.01	0.3473636772	0.8723730092	0.3413559965	9	0	0	21
Iris-versicolor-1CapOcul-4Neur	6000	0.01	0.3501297489	0.7110661482	0.3396177465	9	0	0	21
Iris-versicolor-1CapOcul-6Neur	6000	0.01	0.3559359333	0.9939467928	0.3444043008	9	0	0	21
Iris-virginica-1CapOcul-2Neur	6000	0.01	0.6714460403	0.6727969866	0.2090545893	11	0	0	19
Iris-virginica-1CapOcul-4Neur	6000	0.01	0.6650536344	0.6711564925	0.0849863101	11	0	0	19
Iris-virginica-1CapOcul-6Neur	6000	0.01	0.6107608724	0.6684162767	0.0033441115	11	0	0	19

Para el caso de una capa oculta, tenemos que el mejor de los los resultados clasifica correctamente el 73.3% de los casos de prueba, y el peor clasifica correctamente el 63.3% de los casos de prueba.

Nuevamente observamos poca variación con respecto a los resultados, siendo que cada clasificador se comporta de manera similar sin importar la cantidad de neuronas en su capa. En estos casos se utilizó un 0.01 de learning rate y 6000 iteraciones, puesto que se quiere ver el desempeño más que debido al impacto de las neuronas.

index	Species Classifier	Iters	Learni...	Mean ...	Max E...	Min Er...	False Negat...	False Positive	Positive	Negative
0	Iris-setosa-1CapOcul-2Neur	6000	0.1	0.190876...	0.398928...	0.113889...	1	0	9	20

Por ejemplo, si variamos para el caso de 2 neuronas y 1 capa oculta el learning rate a ser de 0.1 y usamos el mismo número de iteraciones, para el modelo de iris setosa, podemos ver que solo comete un error en su clasificación, en lugar de 10, con un learning rate de 0.01.

Podemos concluir de estos experimentos, que para un learning rate menor, es necesario que tenga más iteraciones para converger a una mejor solución, o utilizar un learning rate mayor. Pero más allá de esto, el desempeño del clasificador binario es generalmente bueno con una capa oculta y con un entrenamiento adecuado, este

presenta tan buenos resultados como aquel de una sola neurona y sin capas ocultas.

Casos Unineuronales, 2 Capas Ocultas:

Species Classifier	Iters	Learni...	Mean Error	Max Error	Min Error	False Negative	False Positive	Positive	Negative
Iris-setosa-2CapOcul-2_4Neur	6000	0.01	0.3281079578	0.9676997141	0.3267337082	10	0	0	20
Iris-setosa-2CapOcul-4_6Neur	6000	0.01	0.3317684104	0.9974323921	0.3011513953	10	0	0	20
Iris-versicolor-2CapOcul-2_4Neur	6000	0.01	0.3462775501	0.986975301	0.3433132162	9	0	0	21
Iris-versicolor-2CapOcul-4_6Neur	6000	0.01	0.3448901612	0.8979914255	0.3445161302	9	0	0	21
Iris-virginica-2CapOcul-2_4Neur	6000	0.01	0.670280293	0.6719330194	0.0277723969	11	0	0	19
Iris-virginica-2CapOcul-4_6Neur	6000	0.01	0.6694949635	0.6702302342	0.0476242646	11	0	0	19

Para estos casos, observamos resultados sumamente similares al caso de 1 capa oculta, independientemente de la cantidad de neuronas. Esto sucede porque el conjunto de funciones que aprende las redes de una sola capa, es prácticamente el mismo para el caso de dos capas ante el universo de datos presentados.

Análisis de resultados:

- Tasa de aprendizaje: Para estos casos, las tasas de aprendizaje usadas proveen variaciones muy pequeñas en los resultados, por lo que es indiferente a menos que manejen capas, donde tiende a ser favorable un valor mayor (el desempeño de 0,1 fue levemente mejor que el de 0.01) que siga considerándose pequeño.
- Iteraciones: Ante un learning rate menor, es recomendable utilizar una cantidad mayor de iteraciones, para permitir al modelo a que llegue a una convergencia. Mientras más capas y neuronas tenga el modelo, y mayor sea el learning rate, se necesitarán más iteraciones .
- Número de capas: El número de capas tiende a ser irrelevante ante el caso de clasificación binaria en el universo probado (iris.csv), siempre y cuando se le den suficientes iteraciones dependiendo del learning rate, por ejemplo, para un learning rate de 0.01, es recomendable usar más de 6000, pero estas son suficientes para un learning rate de 0.1. Además, puesto que el universo es pequeño, una sola capa tiene el mismo desempeño que dos capas, debido

a que las funciones aprendidas para la estimación de valores son pocas y estas pueden manejarse de forma correcta en una sola capa.

- Neuronas por capa: Las neuronas por capa tienden a ser irrelevantes en un clasificador binario para el universo presentado. Aunque debido al funcionamiento de las neuronas, si el universo fuera más grande, los valores a converger son más complicados y un mayor número de neuronas ayuda a que las capas pueden tener más alimentación de información.

Conclusiones Clasificadores Binarios:

Definitivamente estamos ante un conjunto linealmente separable y esto puede ser corroborado por los resultados, ya que al haber conseguido tantos aciertos ante sólo dos posibles respuestas -si y no-, podemos inequívocamente concluir que este conjunto puede ser linealmente separado en dos conjuntos para cada clasificador binario.

Clasificadores Multivariables

Hidden Layers	Neuron Per Layers	Iters	Learning Rate	Mean Error	Max Error	Min Error	Correct Classifications	Average Certainty	Incorrect Classifications	Average Incorrect Activation
1	[2]	1000	0.1	0.0029278143	0.6536039919	0.3518208059	17	0.3811287502	13	0.3689900822
1	[2]	1000	0.01	0.0064677036	0.6712740539	0.3407296057	14	0.3438312902	16	0.3429813895
1	[2]	5000	0.1	0.0286694628	0.6539183931	0.3494847439	23	0.604207732	7	0.4274643798
1	[2]	5000	0.01	0.0017111263	0.6725683305	0.3429586961	9	0.3424541019	21	0.3424719819
1	[2]	10000	0.1	0.030651576	0.6550089958	0.3518002273	23	0.6103767175	7	0.422562532
1	[2]	10000	0.01	0.0006400446	0.6727160022	0.3429177584	9	0.342689112	21	0.3425699261
1	[4]	1000	0.1	0.0026650654	0.6389293223	0.3582865418	11	0.3799809665	19	0.3785983938
1	[4]	1000	0.01	0.0056447982	0.6710836827	0.3431331608	9	0.3429745238	21	0.3434370458
1	[4]	5000	0.1	0.002497043	0.6387587775	0.3584393501	11	0.3796948234	19	0.3784902679
1	[4]	5000	0.01	0.0057854985	0.6716280293	0.3428287784	21	0.3899337252	9	0.3561350191
1	[4]	10000	0.1	0.0122186977	0.638822213	0.358288228	21	0.4703637032	9	0.4304078694
1	[4]	10000	0.01	0.002066984	0.6709729162	0.3436914852	18	0.3467626501	12	0.346911941
1	[6]	1000	0.1	0.0065690187	0.6296906989	0.3614297748	12	0.4264040587	18	0.3905542685
1	[6]	1000	0.01	0.0093408306	0.6694860273	0.3445695325	9	0.3434680255	21	0.3435340919
1	[6]	5000	0.1	0.0174137288	0.6306653018	0.3610742212	21	0.5359919395	9	0.451640285
1	[6]	5000	0.01	0.0061848477	0.6709193122	0.3433688403	21	0.403941737	9	0.3621650696
1	[4]	10000	0.1	0.0122186977	0.638822213	0.358288228	21	0.4703637032	9	0.4304078694
1	[6]	10000	0.01	0.0169412422	0.6676923136	0.3441162224	21	0.4882624976	9	0.3879194351
1	[8]	1000	0.1	0.0263584065	0.6248903392	0.3639733209	20	0.4679374924	10	0.4308851174
1	[8]	1000	0.01	0.0143728785	0.6687398063	0.3437885102	23	0.3519618287	7	0.3471837135
1	[8]	5000	0.1	0.0165489535	0.6259731391	0.3636539286	21	0.5088886555	9	0.4402589083
1	[8]	5000	0.01	0.0133368074	0.6671797589	0.3453617241	24	0.3639256661	6	0.3538931907
1	[8]	10000	0.1	0.0559110631	0.6244536297	0.3623333496	25	0.6866207854	5	0.4718247066
1	[8]	10000	0.01	0.0046349495	0.6684601926	0.344750926	21	0.3838915482	9	0.3592470674
2	[2, 4]	5000	0.1	0.0678411857	0.6604012368	0.3441045435	28	0.7280048669	2	0.5393301775
2	[2, 4]	5000	0.01	0.0012544475	0.6724846047	0.3430268316	9	0.3424964401	21	0.3424976666
2	[2, 4]	10000	0.1	0.032251271	0.6620579166	0.3487985149	28	0.704939454	2	0.5365776778
2	[2, 4]	10000	0.01	0.0007024582	0.672247595	0.3431426907	9	0.3425594755	21	0.3425633296
2	[4, 6]	5000	0.1	0.0594041036	0.6357904347	0.3592030517	28	0.7140639058	2	0.5149137003
2	[2, 4]	5000	0.01	0.0012544475	0.6724846047	0.3430268316	9	0.3424964401	21	0.3424976666
2	[4, 6]	10000	0.1	0.0515827366	0.6451286771	0.086059153	28	0.7794862448	2	0.5800662749
2	[4, 6]	10000	0.01	0.0028250653	0.6713298364	0.3421348902	9	0.3428178562	21	0.3428284844
2	[6, 8]	5000	0.1	0.0479051624	0.6514766994	0.3541179329	28	0.7092502211	2	0.5144451407
2	[6, 8]	5000	0.01	0.0097273943	0.6679207586	0.3452282581	9	0.3437645784	21	0.343787794
2	[6, 8]	10000	0.1	0.0434265209	0.6431547635	0.0155489129	28	0.8243294638	2	0.6438346134
2	[6, 8]	10000	0.01	0.012943398	0.6690131645	0.3447649373	9	0.3435157528	21	0.3435186349
2	[5, 5]	5000	0.1	0.0427545812	0.6525123865	0.3514291456	28	0.7081957885	2	0.5292599069
2	[5, 5]	5000	0.01	0.0012453805	0.6706815334	0.3439275342	10	0.3446817	20	0.3439520442
2	[5, 5]	10000	0.1	0.0358737895	0.6430317656	0.0097673276	28	0.8383067793	2	0.6786746101
2	[5, 5]	10000	0.01	0.0018602174	0.6714250347	0.3434549993	9	0.3427414751	21	0.3427778675
2	[7, 7]	5000	0.1	0.065641208	0.6439509748	0.3561066667	28	0.725971122	2	0.5163834858
2	[7, 7]	5000	0.01	0.0219849174	0.6695922669	0.3444870503	9	0.3433258709	21	0.3433279627
2	[7, 7]	10000	0.1	0.0601094546	0.6436853939	0.3562571852	28	0.7339763121	2	0.5614965726
2	[7, 7]	10000	0.01	0.0079314045	0.6695707104	0.3445270324	9	0.3436207009	21	0.343539746

Análisis de resultados:

- Tasa de aprendizaje: Se hicieron pruebas con valores de 0.1 y 0.01 y se obtuvieron mejores resultados utilizando una tasa de aprendizaje de 0.1, puesto que estas requieren menos iteraciones para converger a una solución, pero de haber muchas más iteraciones, pudiera tenerse una mejor rendimiento el modelo para una tasa de 0.01, puesto que este requiere más iteraciones para converger pero lo hace de manera más precisa .
- Valores de activación: Se ve una fuerte relación entre el valor promedio de activación de las neuronas y la precisión de la red. En los datos se puede

evidenciar que un mayor valor de activación promedio para los casos correctos trae consigo un incremento en la precisión del modelo. Estos valores de activación, dependen de la cantidad de neuronas y capas. Mientras más haya, más iteraciones tiene que tener el modelos para que este vaya aumentando en la certeza de sus resultados.

- Iteraciones: El número de iteraciones afecta de forma interesante la precisión del modelo. Sí bien queda evidenciado que las redes con mayor precisión fueron entrenadas con 10000 iteraciones, también se encuentran en el tope múltiples de redes entrenadas con 5000 iteraciones, esto es debido a los valores de aprendizaje rate. Como se ha visto en el modelo binario, mientras más pequeño el aprendizaje rate, más iteraciones requiere para converger a las soluciones correctas.
- Número de capas: Los modelos con mejor precisión fueron los modelos con 2 capas ocultas, gracias a la presencia de más neuronas que permiten comunicación entre sí por lo que entrena de mejor manera las funciones aprendidas que convergen resultados. Sin embargo, los modelos con una sola capa oculta se comportaron mejor que un grupo de modelos de 2 capas ocultas ante la búsqueda de aciertos, porque como en el caso anterior, las funciones que aprende una sola capa son suficientes para clasificar de forma satisfactoria.
- Neuronas por capa: Un mayor número de neuronas genera mejores resultados pero requiere un entrenamiento mayor. Hay que conseguir un balance entre el número de neuronas, el número de iteraciones y el aprendizaje para obtener la mayor precisión y mayor número de aciertos.

Conclusiones Clasificadores Binarios VS Clasificadores Multiclase:

Ante lo visto anteriormente, puesto que este es un conjunto linealmente separable, es mejor utilizar para el universo presentado en iris.csv un clasificador binario, ya que este es más correcto en evaluar para cada clase la pertenencia o no de un ejemplo. El clasificador multiclase no tiene un desempeño pobre ante un buen entrenamiento, pero definitivamente no posee la misma eficiencia para clasificar datos que el binario. Esto podemos atribuirlo a una falta de datos existente en el

universo, ya que no se necesitan redes tan complejas para converger a resultados tan simples como los de un conjunto linealmente separable. Ante una situación en la cual el conjunto de evaluación no posea tal característica, es posible que un modelo más robusto, donde se posea capas ocultas y neuronas con un evaluador multiclase sea más efectivo. De igual manera, creemos que las funciones que estima el clasificador, pueden ser evaluadas y mejoradas en el clasificador multiclase pero esto conlleva más entrenamiento con respecto a las iteraciones y el learning rate, por lo que sigue siendo mas optimo, para este caso, utilizar un clasificador binario.

Clasificador de spam:

Para el clasificador de spam, nos encontramos con un conjunto donde el resultado se clasifica como spam, o no, por lo que ante lo visto anteriormente, tiene sentido realizar un clasificador binario. Probaremos parámetros utilizados para el set de iris, para identificar cuales son las mejores configuraciones para dicho clasificador y ver como afectan las variaciones en los parámetros.

Species Classifier	Iters	Learning Rate	Mean Error	Max Error	Min Error	False Negative	False Positive	Positive	Negative
spam-class-1hidden-5Neur-001LR	1000	0.01	0.7056713932	0.7356343171	0.0294955398	249	82	141	449
spam-class-1hidden-5Neur-01LR	1000	0.1	0.7929236397	0.8021777488	0.6233855101	214	100	176	431
spam-class-1Neur	1000	0.1	0.160963056	0.6781082695	0.069035614	38	24	352	507
spam-class-2Hidd_5_5Neur-1000_001	1000	0.01	0.7245677765	0.7657007357	0.6188576436	224	94	166	437
spam-class-2Hidd_5_5Neur-1000_01	1000	0.1	0.7936853217	0.8039160188	0.6247749759	217	100	173	431

Para evaluar el spam, se utilizaron las mejores hipótesis obtenidas en el entrenamiento de iris.csv. Se mantuvo un mismo valor de iteraciones para todos los casos.

Claramente el mejor resultado es el último, el que representa al modelo sin capas ocultas, el cual solo contiene un error de 6.73% en los casos evaluados. Esto, al igual que en iris.csv, sucede porque es un clasificador binario, el cual no tiene tantas dificultades en llegar a una convergencia con una sola neurona. Con el learning rate y el número de iteraciones seleccionado, es más que suficiente para entrenar a una

sola neurona, la cual evalúa de forma satisfactoria los ejemplos proveídos. Para los demás clasificadores, la tasa de errores es mayor, tanto de entrenamiento como de evaluación, ya que el número de capas y neuronas no tiene el suficiente entrenamiento puesto que el dataset es muy grande. Por lo que es posible que un modelo con más neuronas y más capas, junto con más iteraciones puede llegar a predecir mejor por todo lo analizado a lo largo de los 2 conjuntos de pruebas de esta evaluación. Aun así, es evidente que ante un conjunto linealmente separable, un modelo de una sola neurona de salida y sin capas ocultas entrenado con un correcto número de iteraciones y tasa de aprendizaje, es capaz de evaluar satisfactoriamente y converger a una solución apropiada.