Universidad de Chile
Facultad de Ciencias Físicas y Matemáticas
Departamento de Ciencias de la Computación

Tarea 1

Entrenando una red neuronal

Alumna: Romina Romero Oropesa

Profesor: Alexandre Bergel

Auxiliares: Juan Pablo Silva

Ayudantes: Alonso Reyes Feris
Gabriel Chandía

Fecha de entrega: 9 de noviembre de 2018
Santiago, Chile

Índice de Contenidos

Lista de Figuras

Lista de Códigos

Implementación

El código de la implementación se encuentra en el repositorio de github https://github.com/romina-romero/redes_neuronales_2018_2. El lenguaje utilizado es python. La red neuronal se encuentra implementada en la carpeta trabajo_incremental. Aquí se incluye además una serie de tests que muestran gráficas de ejemplo.

Las clases usadas para procesar el dataset elegido son **SigmoidNeuron**, **NeuronLayer** y **Neuralnetwork**. NeuralNetwork y SigmoidNeuron incluyen unittest que validan su funcionamiento.

Para poder ejecutar los tests y hacer uso de esta implementación, se debe instalar el paquete numpy y matplotlib de python con una terminal:

Código 1: Instalación de dependencias.

```
pip install numpy matplotlib scipy
```

En https://pip.pypa.io/en/stable/installing/ se explica como instalar pip.

Para ejecutar los unittest:

Código 2: Ejecución de unittest.

```
cd trabajo_incremental
python SigmoidNeuron.py
python NeuralNetwork.py
```

Se incluye una colección de pruebas que se ha hecho durante el curso, a modo de ejemplo, los cuales entregan gráficos. Estos se encuentran en trabajo_incremental.

Las pruebas oficiales con el dataset elegido se encuentran en la carpeta pruebas_tarea_1. Para ejecutarlo, en la terminal:

Código 3: Ejecución de unittest.

```
cd pruebas_tarea_1
python efecto_lr.py
python efecto_shuffle.py
python efecto_numero_capas.py
python cambio_neuronas.py
```

Spambase Data Set

El dataset elegido es el Spambase Data Set (https://archive.ics.uci.edu/ml/datasets/Spambase). Es una colección de 4601 emails, clasificados como spam o no spam, caracterizados en un vector de largo 57.

Cada email se describe de la siguiente forma:

- 48 porcentajes de aparición de palabras claves sobre el total de palabras del email. Una palabra en este caso es cualquier conjunto de caracteres alfanuméricos delimitados por caracteres no alfanuméricos.
- 6 porcentajes de aparición de caracteres claves sobre el total de caracteres del email.
- Promedio de los largos de las secuencias ininterrumpidas de mayúsculas, en el email.
- Largo de la secuencia ininterrumpida de mayúsculas más larga.
- Total de letras mayúsculas del email.

El vector incluye además un ítem número 58, donde se indica si es (1) o no (0) es spam.

El dataset será dividida en dos partes, 2302 para **entrenamiento** y 2299 para **test**.

Tests

Las pruebas se encuentran en la carpeta pruebas_tarea_1. La red tiene 57 entradas y una salida, y cada capa oculta tiene 38 neuronas. Se usó una sola capa oculta en todos los tests excepto en la prueba de variación de capas ocultas

Dataset ordenado

Todos los emails clasificados como spam se encuentran en la primera parte del dataset, y los que no, al final.

lr=0.1

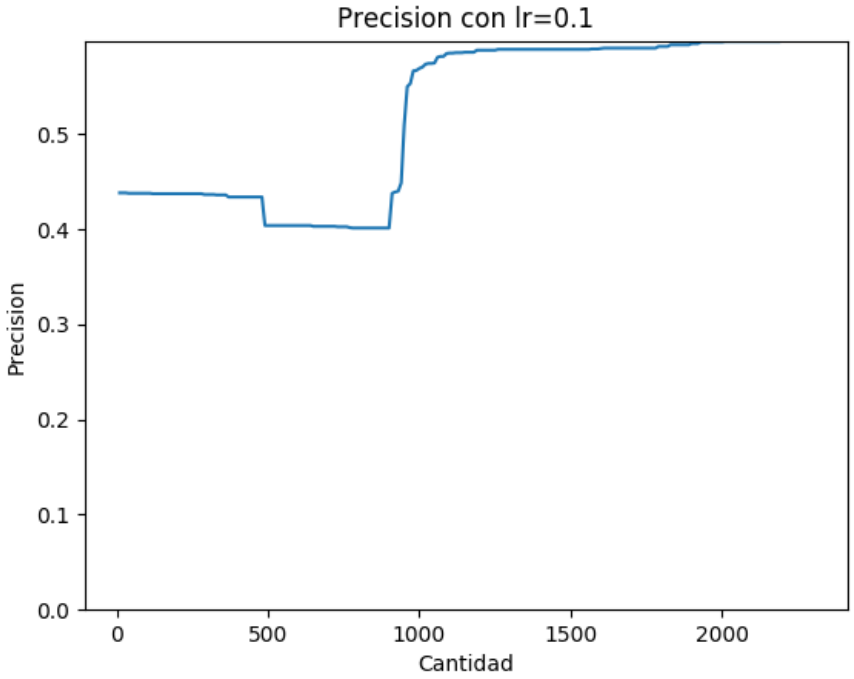

Figura 1: Precisión con learning rate=0.1, dataset ordenado

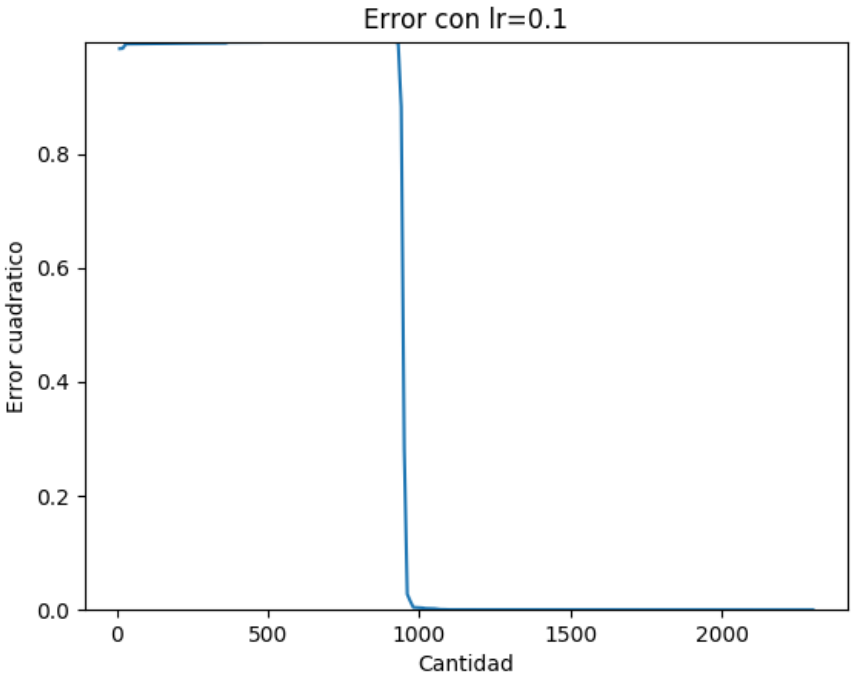

Figura 2: Error con learning rate=0.1, dataset ordenado

lr=0.5

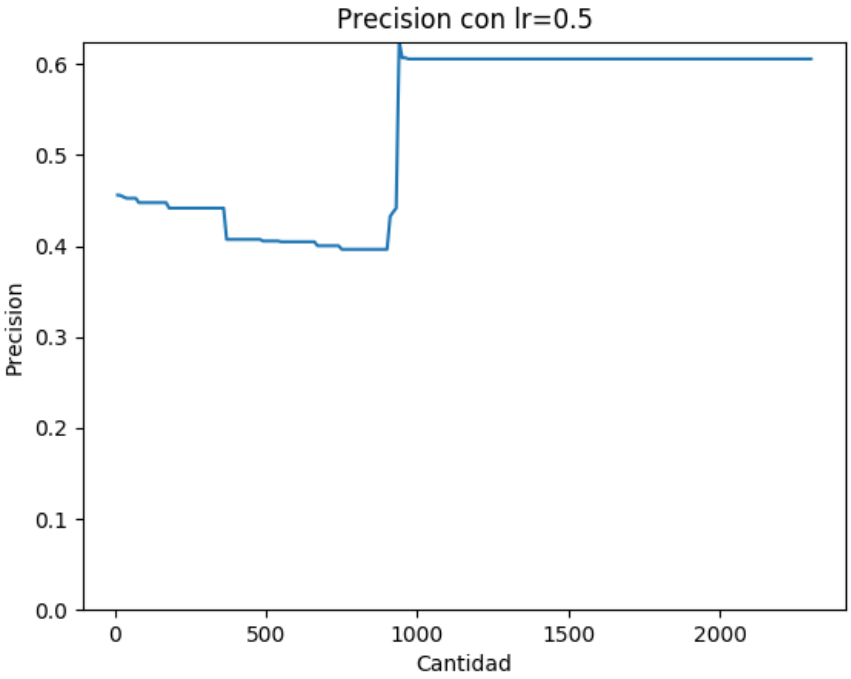

Figura 3: Precisión con learning rate=0.5, dataset ordenado

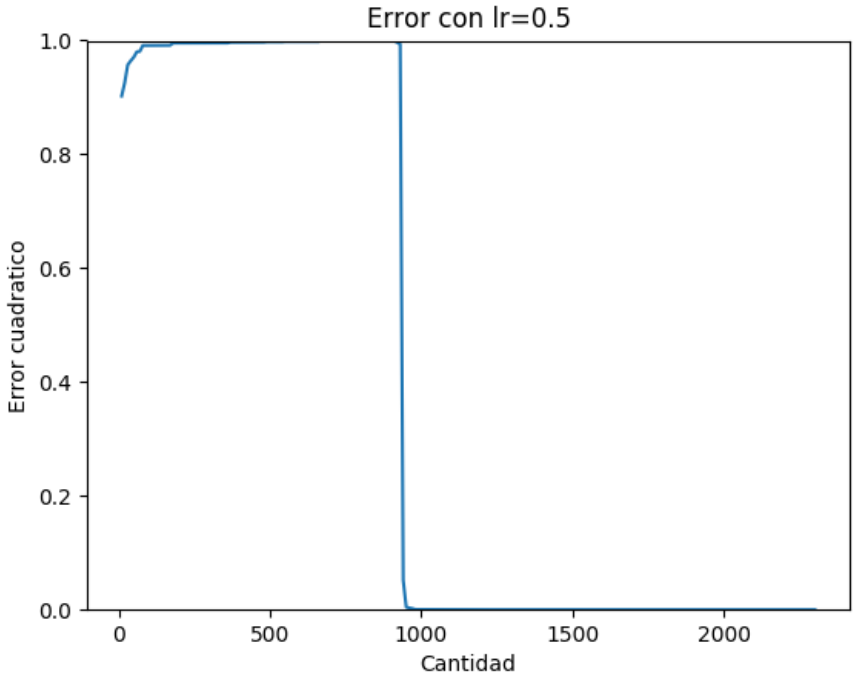

Figura 4: Error con learning rate=0.5, dataset ordenado

lr=1.0

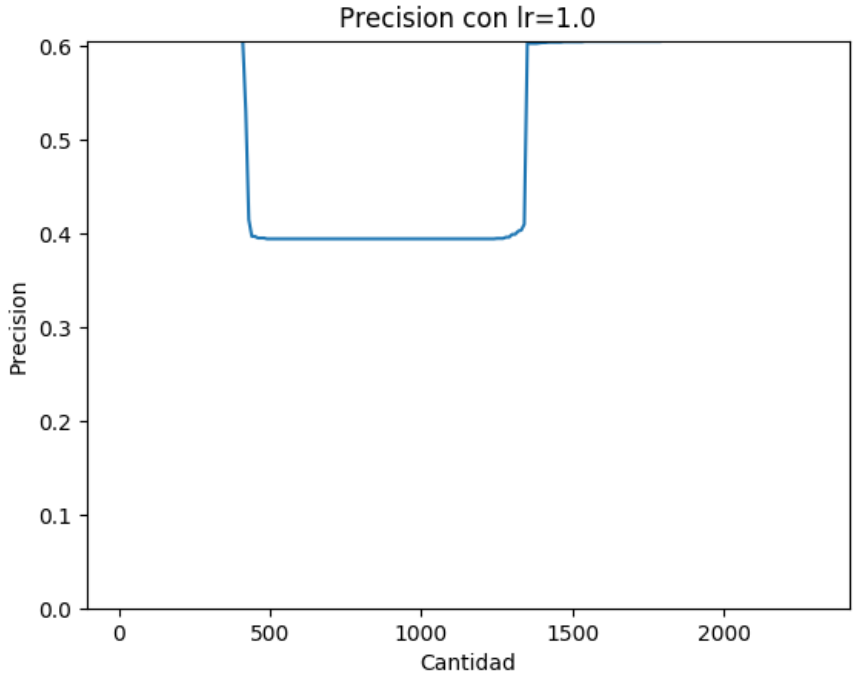

Figura 5: Precisión con learning rate=1.0, dataset ordenado

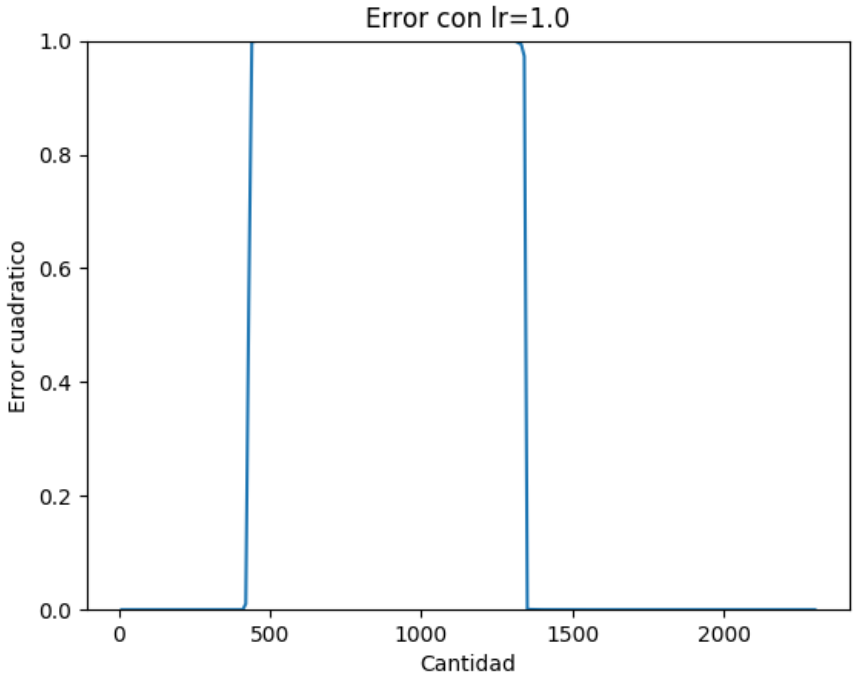

Figura 6: Error con learning rate=1.0, dataset ordenado

lr=1.5

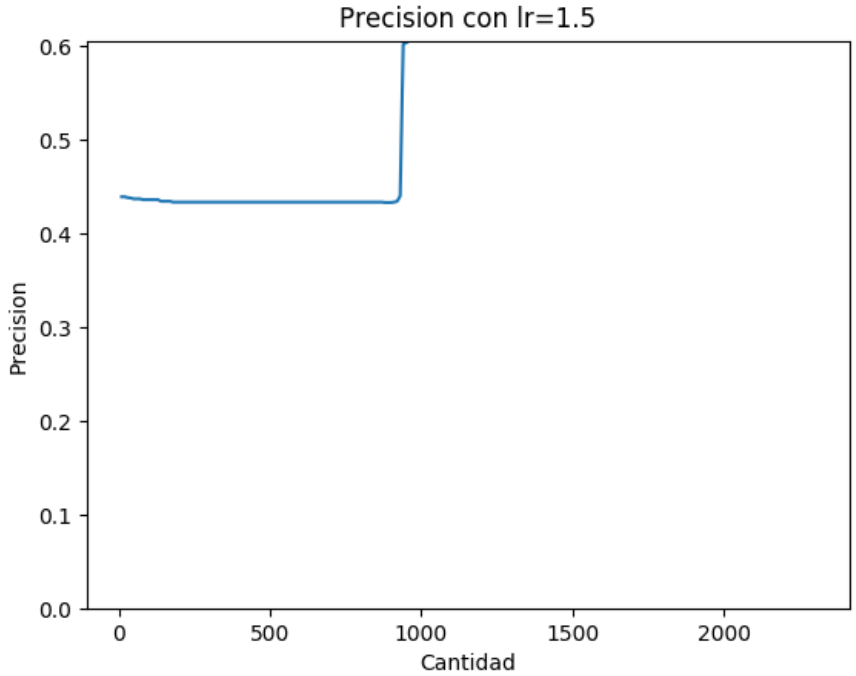

Figura 7: Precisión con learning rate=1.5, dataset ordenado

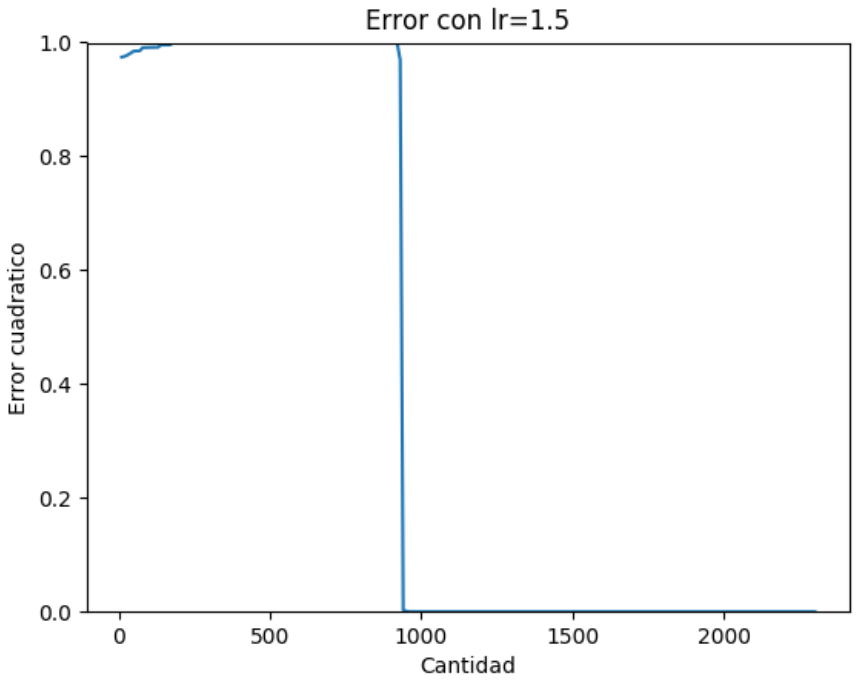

Figura 8: Error con learning rate=1.5, dataset ordenado

Dataset desordenado

Al dataset se le aplicó la función *shuffle* que desordena la lista.

lr=0.1

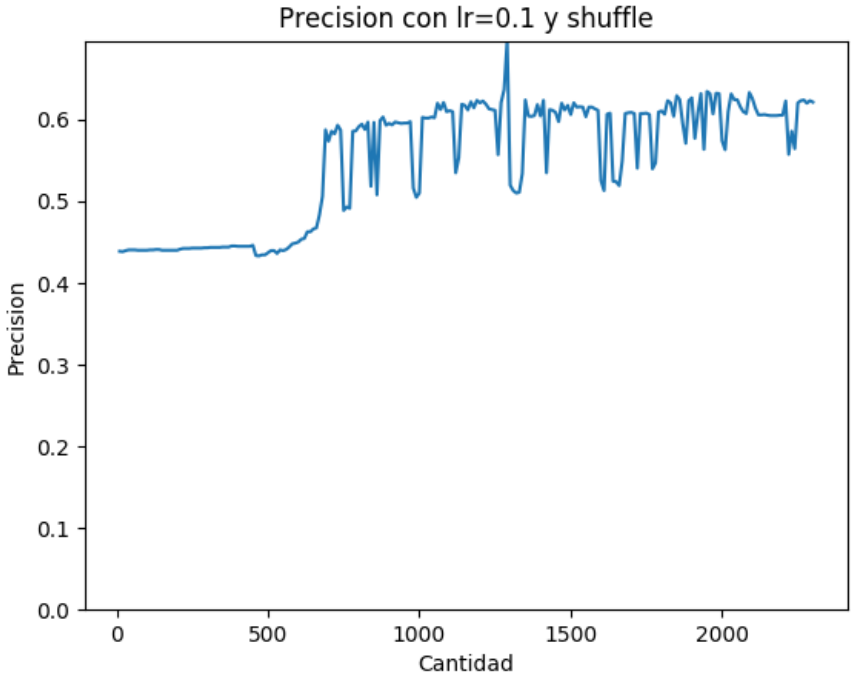

Figura 9: Precisión con learning rate=0.1, dataset desordenado

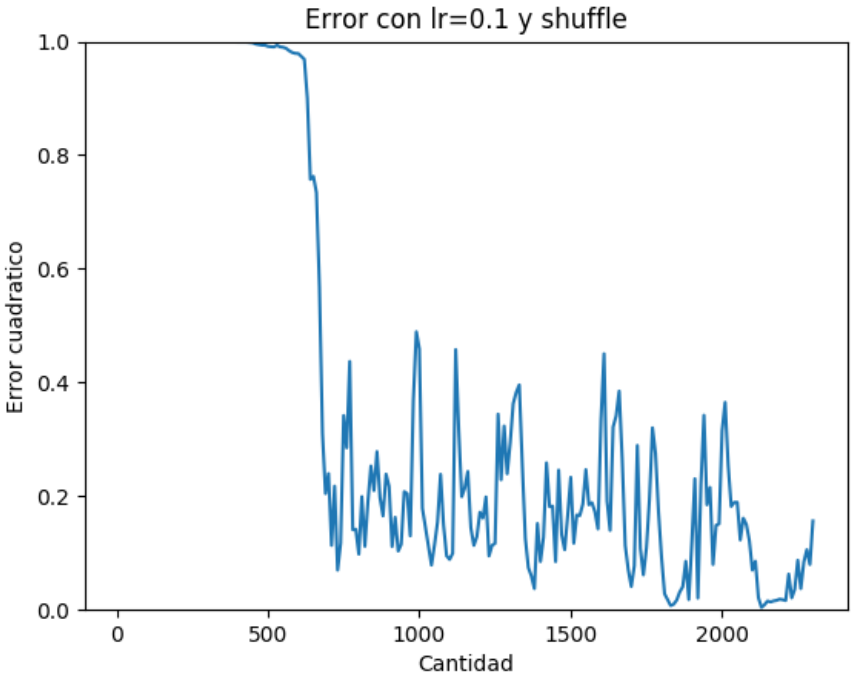

Figura 10: Error con learning rate=0.1, dataset desordenado

lr=0.5

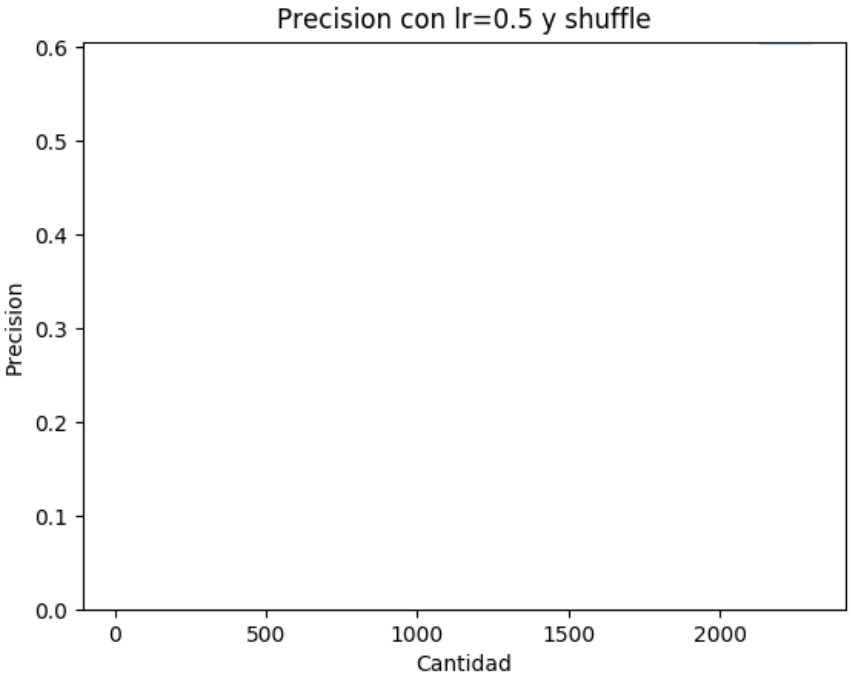

Figura 11: Precisión con learning rate=0.5, dataset desordenado

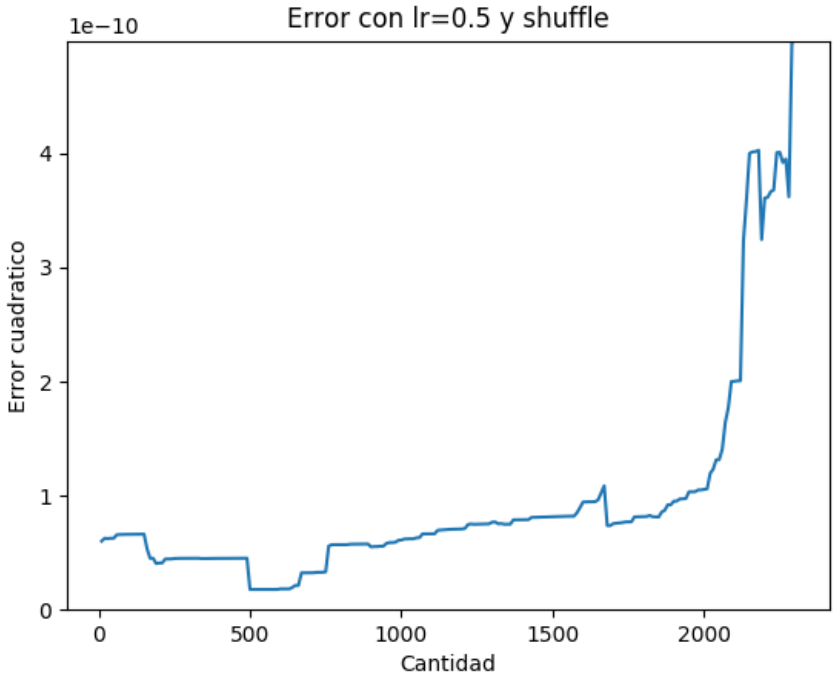

Figura 12: Error con learning rate=0.5, dataset desordenado

lr=1.0

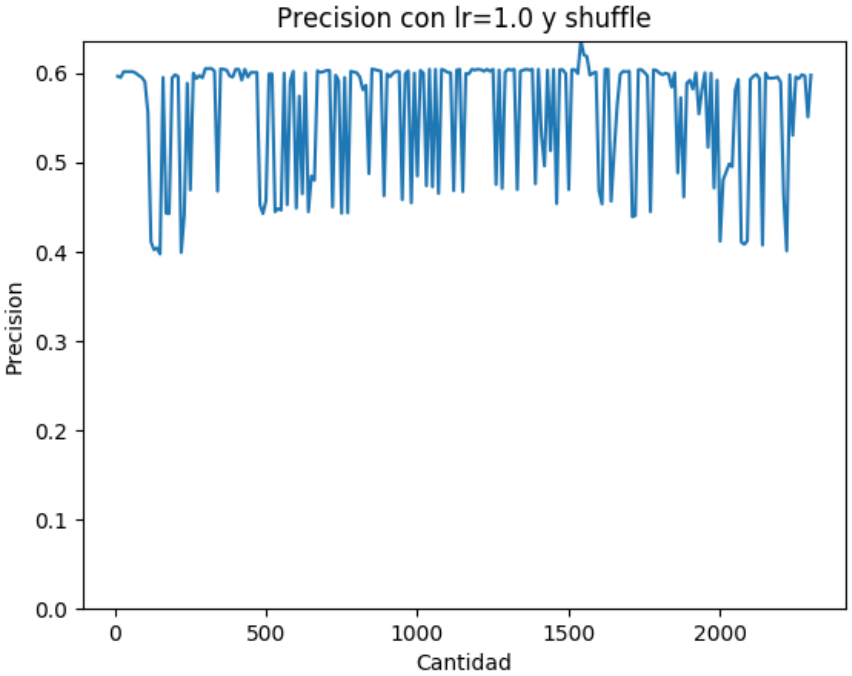

Figura 13: Precisión con learning rate=1.0, dataset desordenado

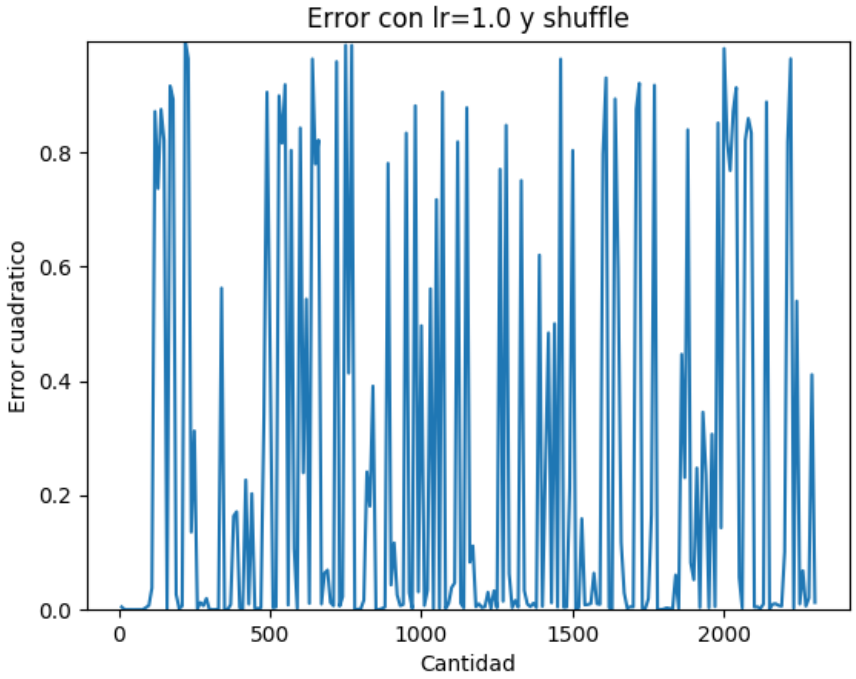

Figura 14: Error con learning rate=1.0, dataset desordenado

lr=1.5

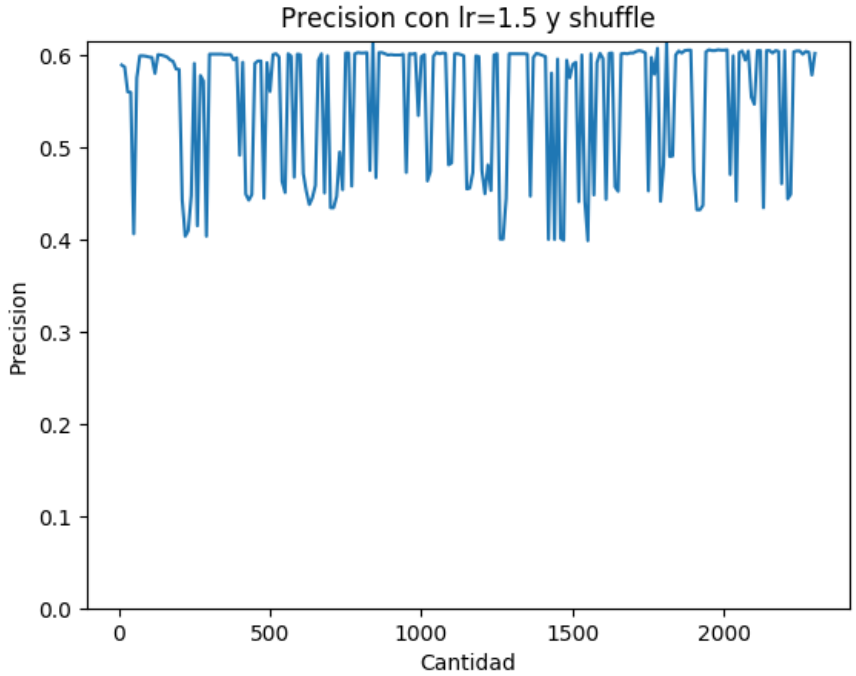

Figura 15: Precisión con learning rate=1.5, dataset desordenado

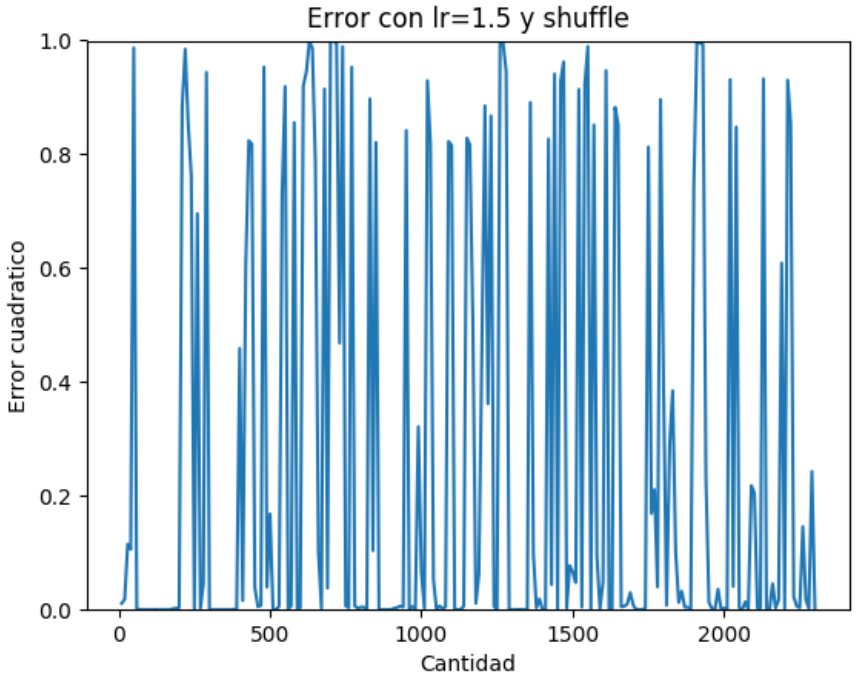

Figura 16: Error con learning rate=1.5, dataset desordenado

Distinto número de capas ocultas

Con learning rate 0.1 y 38 neutonas en cada capa oculta.

Una capa oculta

Tiempo total ejecución: 847 segundos.

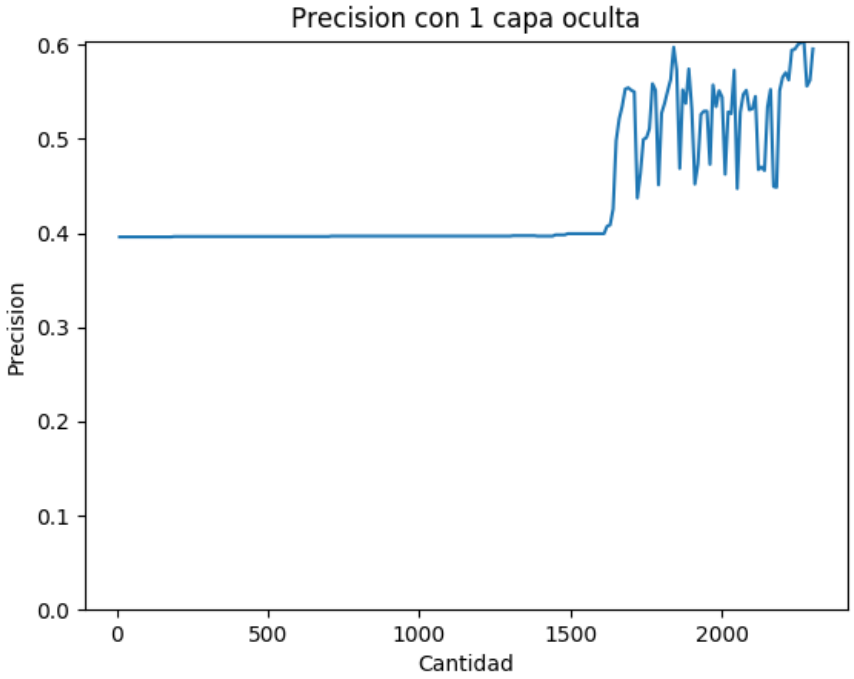

Figura 17: Precisión de red neuronal con una capa oculta

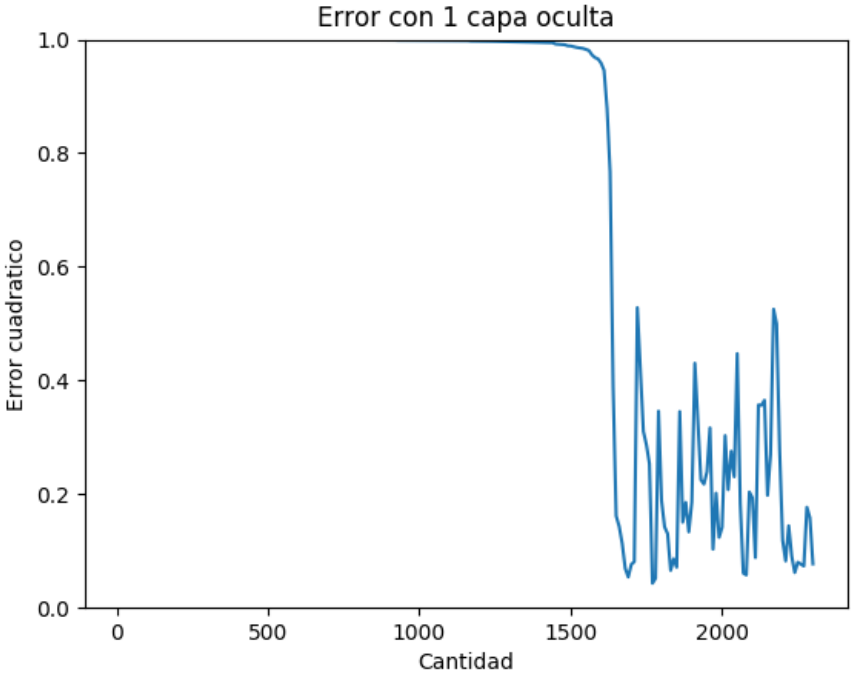

Figura 18: Error de red neuronal con una capa oculta

Dos capas ocultas

Tiempo total ejecución: 1607 segundos.

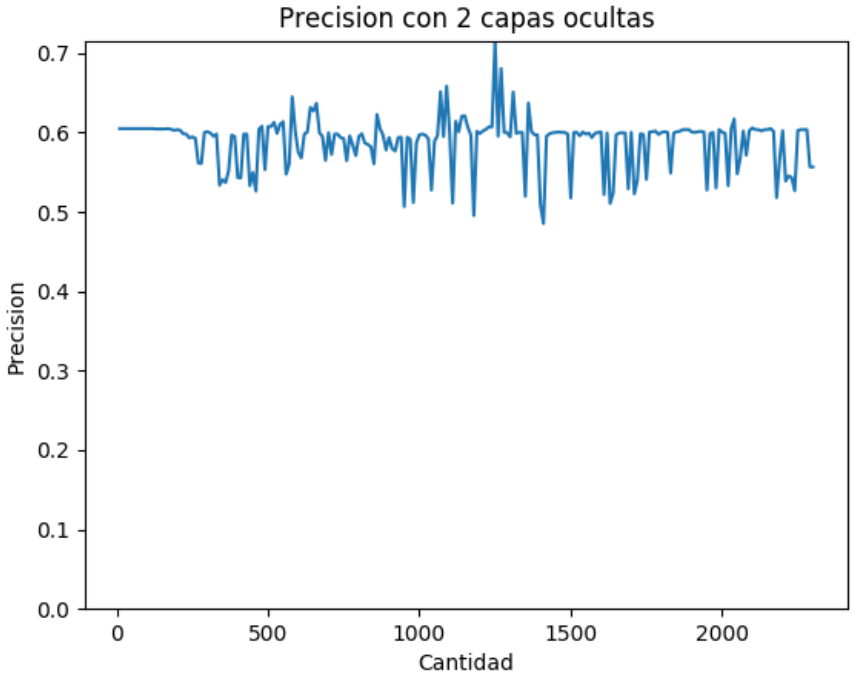

Figura 19: Precisión de red neuronal con dos capas ocultas

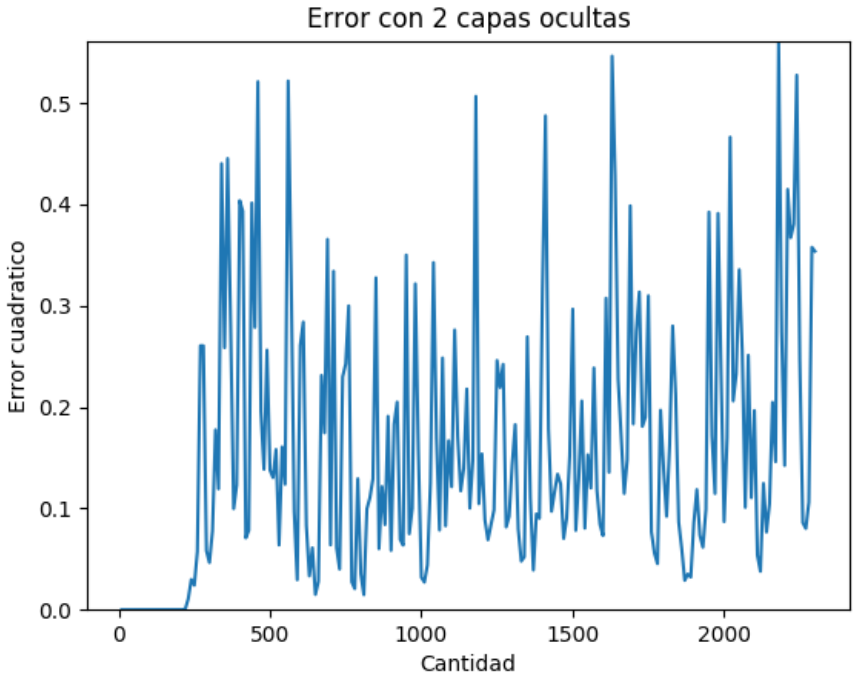

Figura 20: Error de red neuronal con dos capas ocultas

Tres capas ocultas

Tiempo total ejecución: 1883 segundos.

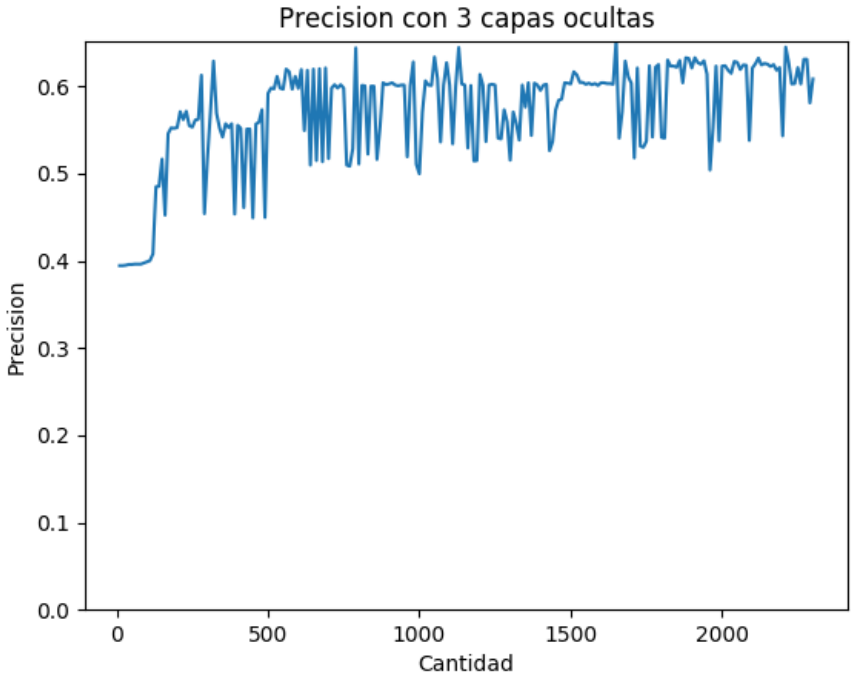

Figura 21: Precisión de red neuronal con tres capas ocultas

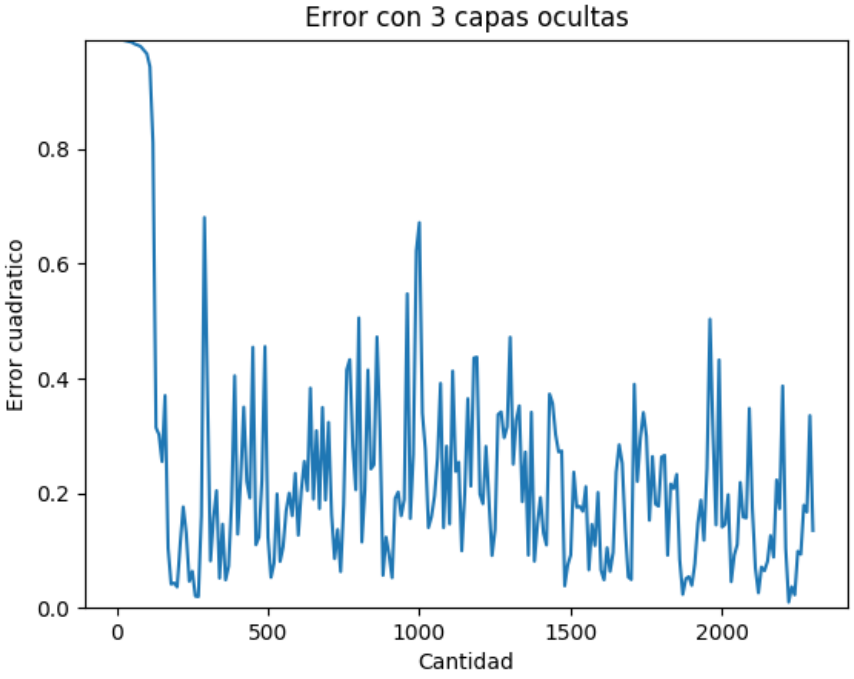

Figura 22: Error de red neuronal con tres capas ocultas

Cambios en neuronas

Las neuronas que más cambian son las más cercanas a las salidas, segun el experimento hecho en el archivo cambio_neuronas.py.

Resultados

- Para nuestro dataset, la precisión queda estancada en 0.6. Se cree que incrementando la cantidad de salidas pudiese mejorar. Queda propuesta esta prueba.
- Al desordenar el dataset, el resultado final empeora, pues es inestable. Para el caso de learning rate 0.1, se puede ver claramente como va decayendo la curva de error vs incremento de la presición.
- El learning rate puede funcionar bien hasta 0.5, pero después no permite que aprenda bien, el error se dispara y no converge.
- El resultado es similar al agregar una o dos capas adicionales, aunque tiene pequeños saltos hacia la presición 0.7. El tiempo de ejecución aumenta proporcionalmente.
- Las neuronas que más cambian son las cercanas a la salida.
- Se cree que con una arquitectura diferente se puede mejorar el resultado.