Clasificación

Resolver los ejercicios teóricos de las diapos. Luego jugar con las siguientes datos:

parcialitos.txt

Esta base de datos cuenta con las notas de dos parcialitos (diagnósticos) de un curso de diferentes alumnos. La idea es predecir si el alumno va a aprobar finalmente el curso (1) o por el contrario va a desaprobar (0).

1. Usando "loss=tf.keras.losses.BinaryCrossentropy(from_logits=True)", programar con keras el GD. Indicar el learning rate, el patience del early stopping y el riesgo empírico finalmente alcanzado.

2. El comando compile puede recibir, a demás de optimizer y loss, métricás adicionales. Incorpore "metrics=['accuracy']" al mismo e indique el porcentaje de predicciones correctas con el conjunto de train (el único que hay) una vez finalizado el entrenamiento.

3. Grafique el riesgo empírico y el porcentaje de clasificaciones correctas en función de los epochs.

4. Predecir si un estudiante con notas de $(63, 55)$ va a aprobar.

5. Hacer un scatter plot con los datos, ploteando las dos clases con colores diferentes y denotanto las etiquetas en el gráfico. Superponer sobre el mismo la frontera de decisión.

glass.csv

En una escena del crimen, un vidrio dejado puede usarse como evidencia... ¡si se identifica correctamente! Clasificar a donde pertenecía el vidrio a partir de la concentración de los diferentes elementos químicos que posee. Las clases de vidrio posibles son:

1. building windows float processed
2. building windows non float processed
3. vehicle windows float processed
4. vehicle windows non float processed
5. containers
6. tableware
7. headlamps

A resolver:

1. *Curado de datos*: Observar bien los labels de este dataset y acomodarlo para que keras lo interprete correctamente.

2. Pensar cuantos parámetros se necesitarán para resolver esta tarea con una activación softmax. Crear el modelo y chequear con el comando summary que efectivamente se haya construido el modelo deseado.

3. Usando "loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True)", programar con keras el GD. Indicar el learning rate, el patience del early stopping, el riesgo empírico finalmente alcanzado y el correspondiente accuracy.

4. Crear una función que clasifique vidrios, devolviendo el string correspondiente del tipo de vidrio. Pruebelo para $(1{,}52; 13; 3{,}5; 1{,}2; 73; 0{,}6; 9; 0{,}1; 0{,}2)$

5. Mostrar la confusion matrix. Puede usar "sklearn.metrics.confusion_matrix".

prostata_data.csv y prostata_label.csv

Son mediciones de suero de pacientes de distintos grupos (370 pacientes en total): sanos (sin patologías), cáncer (enfermedad confirmada por un patólogo), post-cirugía (pacientes con enfermedad confirmada y el tumor removido por cirugía) y benignos (pacientes con una forma benigna de la enfermedad). Las mediciones fueron realizadas por cromatografía de ultra alta performance acoplada a espectrometría de masas de alta resolución. El valor asociado a cada feature en la matriz de datos es la abundancia de concentración de un determinado compuesto químico.

1. *Curado de datos*: Quitar todos los datos sin etiquetar.

2. Programar con keras el GD. Indicar el learning rate, el patience del early stopping, el riesgo empírico finalmente alcanzado y el correspondiente accuracy.

3. Mostrar la confusion matrix.

MNIST y FASHION-MNIST

MNIST y FASHION-MNIST son bases de datos de similares caracerísticas (imágenes de 28×28 en escala de grises, 10 clases, $60K$ muestras de entrenamiento y $10K$ de testeo), una de dígitos y la otra de ropa. En el caso de MNIST los labels representan el dígito, mientras que en la versión FASHION los labels son:

- 0: T-shirt/top
- 1: Trouser
- 2: Pullover
- 3: Dress
- 4: Coat
- 5: Sandal
- 6: Shirt
- 7: Sneaker
- 8: Bag
- 9: Ankle boot

A resolver:

1. Ambas bases de datos se encuentran en Keras. Pueden ser cargadas como

```
mnist = tf.keras.datasets.mnist
(train_images, train_labels), (test_images, test_labels) = mnist.load_data()
```

cambiando mnist por fashion_mnist cuando corresponda. Explore los datos de ambas dataset con el siguiente código:

```
plt.figure(figsize=(10,10))
for i in range(25):
        plt.subplot(5,5,i+1)
        plt.xticks([])
        plt.yticks([])
        plt.grid(False)
        img_index = np.random.randint(0, train_images.shape[0])
        plt.imshow(train_images[img_index], cmap="gray_r")
        plt.xlabel(text_labels[train_labels[img_index]])
```

donde text_labels es una lista de strings con las etiquetas correspondientes. Explique brevemente que hace el código.

2. Normalice los datos antes de entrenar y lleve las imágenes a un vector. ¿Que dificultades encontró?

3. Programar con keras el GD para cada base de datos (dos códigos independientes) usando GPU. Indicar el learning rate, el patience del early stopping. También indicar el riesgo y el accuracy alcanzado tanto para el conjunto de entrenamiento como el de testeo.

4. Mostrar la confusion matrix.