Práctica 4

Multiperceptrón

Objetivos

El objetivo de esta práctica es comprender el funcionamiento del multiperceptrón

Temas

- Métricas: precision, recall, F1-Score
- Multiperceptrón

Lectura

Material de Lectura: Capítulos 2 y 3 del libro Neural Networks and Deep Learning.

Parte 1 - Multiperceptrón con SciKit-Learn

Ejercicio 1

Se entrenó una red neuronal multiperceptrón para resolver un problema de clasificación y al medir su desempeño sobre el conjunto de datos de entrenamiento se obtuvo la siguiente matriz de confusión:

clases 1 2 3 4 5

17	***0***	***1***	***0***	***1***
0	***12***	***0***	***0***	***0***
0	***0***	***12***	***0***	***0***
2	***0***	***0***	***38***	***0***
0	***8***	***0***	***0***	***61***

17 clases se clasificacon correctamente como clase 1. 1 clase se clasifico incorrectamente como otra clase

12 clases se clasificaron correctamete como clase 2

a) En base a esta información, indique:
   - Cuántos ejemplos se utilizaron en el entrenamiento.
   - Cuántas clases puede reconocer este multiperceptrón.
   - Cuál es la precisión (***accuracy***) de la red sobre el conjunto de ejemplos completo.
   - Cuáles son los valores de precisión de la red al responder por cada uno de los valores de clase (***precision***).
   - Cuáles son los valores de sensibilidad de la red al responder por cada uno de los valores de clase (***recall***).

b) Identifique la clase con el mejor valor de F1-score.

Ejercicio 2

Se desea utilizar una red multiperceptrón para reconocer muestras de tres variedades diferentes de trigo: Kama, Rosa y Canadiense. Para entrenarla se utilizará una parte de los ejemplos del archivo ***SEMILLAS.CSV***. Estos datos fueron utilizados en el ejercicio 3 la práctica 2.

Fuente de datos: ***Seeds Data Set*** - https://archive.ics.uci.edu/ml/datasets/seeds

a) Con respecto a la arquitectura, indique: 1 neurona de entrada por cada atributo
   - La cantidad de neuronas de la capa de entrada. 3 porq es la cant de clases
   - La cantidad de neuronas de la capa de salida.
   - La cantidad de pesos (arcos) que tiene la red si se utiliza una única capa oculta formada por 4 neuronas.

b) La arquitectura del multiperceptrón utilizado para predecir los 3 tipos de semillas está formada por 3 capas: la capa de entrada, una única capa oculta de 4 neuronas y la capa de salida. Las funciones de activación para las capas oculta y de salida son “tanh” y “sigmoid” respectivamente. Indique cuáles de los siguientes factores inciden en la dirección de cambio (signo de la modificación) de los pesos de la red:
   - El error cometido en la predicción.
   - El valor de la derivada de la función de activación.
   - Los valores anteriores de los pesos de la red.

c) Luego de ingresar una muestra de semilla a la red se obtiene como salida (0.78, 0, 0). Utilizando la arquitectura descripta en b), indique cuántos pesos de la red serán modificados sabiendo que la respuesta esperada es (1, 0, 0).

Ejercicio 3

El archivo ***Vinos.csv*** tiene información referida a 13 características químicas y/o visuales de varias muestras de vinos pertenecientes a 3 clases distintas.

Utilice el 80% de los ejemplos del archivo Vinos.csv para entrenar un multiperceptrón que sea capaz que distinguir entre las 3 clases de vinos. Observe la tasa de acierto obtenida sobre el 20% restante.

Fuente de datos: ***Wine Data Set*** - https://archive.ics.uci.edu/ml/datasets/wine

Ejercicio 4

El archivo ***Balance.csv*** tiene información sobre un experimento psicológico realizado para evaluar el aprendizaje en los niños. Cada fila de la tabla tiene las características de una balanza, referidas a la longitud de los brazos izquierdo y derecho de la balanza y al peso que hay en cada brazo, y un atributo que indica si la balanza se inclina al lado izquierdo (**L**), derecho (**R**), o está balanceada (**B**).

Utilice una parte de los ejemplos para entrenar un multiperceptrón que sea capaz que predecir si la balanza está inclinada a derecha, a izquierda o si está balanceada. Analice la precisión de la red sobre los ejemplos de entrenamiento y sobre los de testeo.

Fuente: ***Balance Scale Data Set*** **-** https://archive.ics.uci.edu/ml/datasets/Balance+Scale

Ejercicio 5

El archivo **ZOO.csv** contiene información de 101 animales caracterizados por los siguientes atributos

1. Nombre del animal	7. Acuático	13. Aletas
2. Tiene Pelo	8. Depredador	14. Patas
3. Plumas	9. Dentado	15. Cola
4. Huevos	10. Vertebrado	16. Domestico
5. Leche	11. Branquias	17. Tamaño gato
6. Vuela	12. Venenoso	18. Clase

Salvo los atributos 1 y 18 que contienen texto y el 14 que contiene el número de patas del animal, el resto toma el valor 1 si el animal posee la característica y 0 si no. Hay 7 valores de clase posible (atributo 18): mamífero, ave, pez, invertebrado, insecto, reptil y anfibio.

Entrene un multiperceptrón que sea capaz de clasificar un animal en una de las 7 clases. Utilice el 70% de los ejemplos para entrenar y el 30% para realizar el testeo. Realice al menos 10 ejecuciones independientes de la configuración seleccionada para respaldar sus afirmaciones referidas a la performance del modelo.

Fuente de Datos: ***Zoo Data Set*** **-** https://archive.ics.uci.edu/ml/datasets/zoo

Ejercicio 6

Los archivos ***Segment_Train.csv*** y ***Segment_Test.csv*** contienen información referida a regiones de 3x3 pixeles pertenecientes a 7 imágenes distintas. Cada una corresponde a uno de los siguientes tipos de superficie: ladrillo, cielo, follaje, cemento, ventana, camino y pasto.

Cada región de 3x3 ha sido caracterizada por 19 atributos numéricos:

1. ***region-centroid-col:*** la columna del pixel central de la región.
2. ***region-centroid-row:*** la fila del pixel central de la región.
3. ***region-pixel-count:*** el número de pixeles de la región = 9.
4. ***short-line-density-5:*** el resultado de un algoritmo de extracción de líneas que cuenta la cantidad de líneas de bajo contraste que atraviesan la región.
5. ***short-line-density-2:*** ídem anterior para líneas de alto contraste.
6. ***vedge-mean:*** medida del contraste entre pixeles adyacentes. Este atributo contiene el valor promedio y el siguiente la desviación. Estas medidas sirven para detectar la presencia de un eje vertical.
7. ***vegde-sd:*** (ver 6)
8. ***hedge-mean:*** ídem 6 para eje horizontal. Contiene el valor medio y el siguiente la desviación.
9. ***hedge-sd:*** (ver 8).
10. ***intensity-mean:*** El promedio calculado sobre la región de la forma (R + G + B)/3
11. ***rawred-mean:*** el promedio sobre la región de los valores R.
12. ***rawblue-mean:*** el promedio sobre la región de los valores B.
13. ***rawgreen-mean:*** el promedio sobre la región de los valores G.
14. ***exred-mean:*** Medida de exceso de color rojo: (2R - (G + B))
15. ***exblue-mean:*** Medida de exceso de color azul: (2B - (G + R))
16. ***exgreen-mean:*** Medida de exceso de color verde: (2G - (R + B))
17. ***value-mean:*** Transformación no lineal 3D de RGB.
18. ***saturatoin-mean:*** (ver 17)
19. ***hue-mean:*** ver 17)

El atributo 20 corresponde al número de imagen de la cual fue extraída la región de 3x3. Sus valores son: 1 (ladrillo), 2 (cemento), 3(follaje), 4 (pasto), 5 (camino), 6 (cielo), 7 (ventana).
Entrene una red neuronal multiperceptrón para que dada una región de 3x3, representada a través de los 19 atributos indicados anteriormente, sea capaz de identificar a cuál de las 7 imágenes corresponde. Utilice los ejemplos del archivo ***Segment_Train.csv*** para entrenar y los del archivo ***Segment_Test.csv*** para realizar el testeo.
Realice al menos 10 ejecuciones independientes de la configuración seleccionada para respaldar sus afirmaciones referidas a la performance del modelo.

Fuente: **Image Segmentation Data Set:** https://archive.ics.uci.edu/ml/datasets/Image+Segmentation

Parte 2 - Multiperceptrón con Tensorflow/Keras

Ejercicio 7

Re-implemente el modelo del ejercicio 4 (Balance.csv) utilizando un Multiperceptrón Tensorflow/Keras.

Ejercicio 8

Utilizando el archivo **Iris.csv** que contiene información referida a la longitud y al ancho de sépalos y pétalos de tres especies de flores: *iris setosa*, *iris versicolor* e *iris virginica*.

a) Entrenar una multiperceptrón que aprenda a clasificar las 3 clases de flores.
b) Utilice Python para calcular la matriz de confusión y calcule de forma manual las métricas de ***precision***, ***recall***, ***accuracy*** y ***f1-score***. Luego utilice la función ***classification_report*** de SciKit-Learn para comparar los resultados.

Ejercicio 9

Utilizando los ejemplos del archivo **AUTOS.csv** genere un modelo utilizando un multiperceptrón para predecir el precio del auto (atributo ***price***) y la cantidad de millas por galón en ruta (***MPG-highway***) en función del resto de los atributos. Recuerde completar los valores faltantes, utilizar normalización y dividir el dataset en entrenamiento y validación.
Realice 30 ejecuciones independientes de la configuración seleccionada para respaldar sus afirmaciones referidas a la precisión obtenida tanto para el conjunto de entrenamiento como para el de testeo. Utilice un máximo de 1000 épocas e implemente una parada temprana

Complete la siguiente tabla y realice un análisis de los valores obtenidos

Optimizador	Función activación	Iteraciones promedio	Accuracy Promedio
SGD	tanh		
	sigmoid		
	ReLU		
	LeakyReLU		
RMSProp	tanh		
	sigmoid		
	ReLU		
	LeakyReLU		
Adam	tanh		
	sigmoid		
	ReLU		
	LeakyReLU		

Donde

- **Épocas Promedio** es el número de épocas promedio en el que se detuvo el entrenamiento
- **Accuracy Promedio** es el promedio de las 30 tasas de acierto obtenidas en cada entrenamiento.

Ejercicio 10

El conjunto de datos **"Fingers"** consiste en una serie de imágenes de 64x64 píxeles con fondo negro donde en su centro se encuentra un mano que muestra una cantidad de dedos que va desde 0 a 5. La versión original de este conjunto de imágenes se encuentra en https://www.kaggle.com/koryakinp/fingers

Muchas veces se utilizan técnicas de procesamiento de imágenes para obtener características representativas (features extraction) de los objetos dentro de una imagen con el objetivo de simplificar el problema, reducir la dimensionalidad y/o reducir el costo de procesamiento. Bibliotecas como ***OpenCV*** o ***SciKit-Learn*** proveen funciones que permiten procesar imágenes y obtener valores estadísticos que caracterizan los objetos.

a) Utilice la función ***regionprops*** de ***SciKitLearn*** con algunas imágenes de ejemplo del dataset para experimentar con las distintas características que extrae de la imagen de la mano.
b) Implemente un script que convierta las imágenes en las carpetas **"test"** y **"train"** del dataset **"Fingers"** en dos archivos (uno por carpeta) separados por comas (csv). De todas las características que provee ***regionprops***, tienen potencial aquellas que son independientes o se pueden independizar de las unidades (píxeles). Algunas de estas características pueden ser:
    - ***filled_area***: cantidad de píxeles que contiene región (podría interpretarse como píxeles cuadrados).
    - ***major_axis_length***: longitud (en píxeles) del eje mayor de la elipse que mejor se ajusta a la región.
    - ***minor_axis_length***: longitud (en píxeles) del eje menor de la elipse que mejor se ajusta a la región.
    - ***perimeter***: cantidad de píxeles que forman el contorne de la región.
    - **eccentricity**: excentricidad de la elipse de mejor ajuste, cerca de 0 es un círculo, mientras que cerca de 1 es una elipse más "larga".
    - ***solidity***: razón entre la cantidad de píxeles de la región original y de la región convexa. Para generar una región convexa se completan los pixeles de forma de eliminar regiones cóncavas de una figura. La región convexa de una estrella de 5 puntas se convertirá en un pentágono al completarla.
    - ***extent***: razón entre pixeles de la región original y el rectángulo que la contiene (bounding box).

c) Entrene un modelo de multiperceptrón a partir del archivo de entrenamiento generado en el punto anterior.
d) Reflexione acerca de las propiedades geométricas generadas a partir de un objeto/región de una imagen. ¿Cómo cree que afectaría la rotación, translación y escalado de un objeto/región dentro de la imagen?