Instituto Tecnologico de Costa Rica (ITCR)
Sede Interuniversitaria de Alajuela
Escuela de Computacion
Curso: Inteligencia Artificial
Profesora: Maria Auxiliadora Mora

Entrega: en el TecDigital de un archivo .zip con un cuaderno jupyter en Pytorch, **debidamente documentado**, con una funcion definida por ejercicio.

Modo de trabajo: Grupos de una o dos personas.

Introduccion

En este trabajo practico se aplicaran conceptos basicos de optimizacion, minimos cuadrados y aprendizaje automatico utilizando redes neuronales convolucionales para resolver problemas de clasificacion de imagenes todo utilizando el lenguaje Python, con la libreria Pytorch.

El objetivo del trabajo es poner en practica el conocimiento adquirido en clase sobre estos temas por medio de ejercicios que permitan al estudiante experimentar con ejemplos de uso.

A. Optimizacion (8 puntos)

1. Optimizacion de funciones por el descenso de gradiente: Recuerde que el descenso del gradiente es un algoritmo de optimizacion que permite converger hacia el valor minimo de una funcion mediante un proceso iterativo. En aprendizaje automatico este metodo se utiliza para minimizar una funcion que mide el error de prediccion del modelo en el conjunto de datos.

   Para las siguientes funciones:

   $f_1(x, y) = (x - 0{,}7)^2 + (y - 0{,}5)^2 con\ x, y \in [-4, 4]$

   $f_2(x, y) = xe^{(-x^2-y^2)}$ con $x, y \in [-2, 2]$

Realice lo siguiente:

1. (1 punto) Grafique las funciones y distinga si las funciones son convexas o no, y los puntos minimos y regiones o puntos silla.
2. (3 puntos) Implemente el algoritmo del descenso del gradiente con tensores de Pytorch
3. (4 puntos) Para cada funcion:
   a) Escoja un coeficiente de aprendizaje ? que permita la convergencia y reporte los resultados para 10 corridas, debe incluir:
      1) La cantidad de iteraciones necesarias para converge.
      2) El punto de convergencia.

3) Reporte si convergio al punto correcto.

B. Minimos cuadrados con tensores de Pytorch (6 puntos)

El ejercicio se va a realizar utilizando el conjunto de datos «Default of Credit Card Clients Dataset. Default Payments of Credit Card Clients in Taiwan from 2005». Se adjunta un archivo con los datos (defaultofcredit). Una descripcion completa del conjunto de datos esta disponible en https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset.

1. Cargue el conjunto de datos defaultofcredit.csv, los targets (b) corresponden a la columna: default_payment_next_month.

2. (1 punto) Explore el conjunto de datos, visualice algunas estadisticas y verifique que no existan valores faltantes.

3. (2 punto) Implemente en python la funcion wopt = estimateOptimunW(X,b) la cual recibe los datos y sus respuestas (X y b) , y estima el vector de pesos optimo w_opt usando minimos cuadrados. Debe programar en python a nivel de operaciones matriciales y/o vectoriales, no debe invocar a una funcion pre-construida que calcule minimos cuadrados. Utilice la funcion estimateOptimumW para calcular el wopt para los datos.

4. (2 punto) Implemente funcion forward, la cual estima las salidas del modelo al hacer T=f(Xw opt). Donde la funcion f(x) se refiere a la funcion de activacion, que decide a cual clase pertenece cada muestra, segun el resultado del producto punto de las muestra y los pesos optimos, en este caso simplemente usando la funcion signo o escalon, es decir:

$$\text{f(x)} = 1, \text{si(x} > 0)$$

$$\text{f(x)} = \text{-1, si(x} <= 0)$$

5. (1 punto) Evalue el error de prediccion utilizando la distancia euclideana.

C. Perceptron de una capa con tensores de Pytorch (12 puntos)

1. (10 puntos) Implemente el algoritmo del perceptron de una capa rescindiendo al maximo de estructuras de tipo for, usando en su lugar operaciones matriciales. Debe implementarlo sin utilizar ninguna biblioteca, es decir en Pytorch no se puede usar ninguna clase o funcionalidad desarrollada por Pytorch o alguna otra biblioteca.

2. Reporte los resultados del clasificador: Para probar su clasificador utilice cumulos de datos generados con la funcion createData disponible en el cuaderno de Jupyter 003_LeastSquares_ejemploV2.ipynb (directorio 014 y 015

Minimos cuadrados) la cual implementa la generacion de datos aleatorios. Tales datos seran utilizados como datos de prueba. Parametrice la cantidad de muestras, matriz de covarianza y medias. La meta es lograr que el perceptron clasifique bien los datos generados. Es decir construis un separador lineal de las clases.

2.1. (3 puntos) Realice 2 pruebas con distintas distancias de separacion entre las muestras de las clases, con una prueba linealmente separable, y otra no, y documente el numero de muestras mal clasificadas y la cantidad de iteraciones para converger. Defina el conjunto de muestras de entrenamiento como el 70 % de las muestras aleatoriamente seleccionadas, y el resto utilicelas como muestras de prueba.

2.2. (2 puntos) Grafique el error o perdida de entrenamiento de al menos dos corridas, con todas las iteraciones de esas corridas. Ejemplo de grafica de error por epoca:

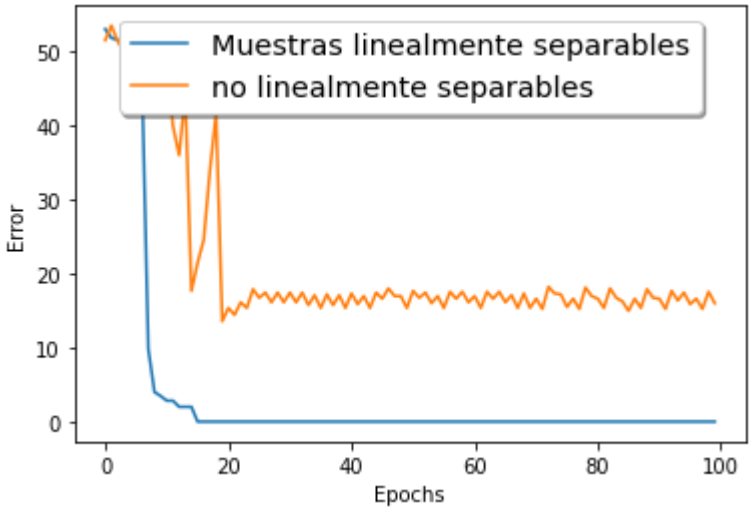

Figura 1. Error de entrenamiento para muestras linealmente separables generadas de forma aleatoria

D. Redes neuronales convolucionales (16 puntos)

Utilice PyTorch para implementar una red neuronal profunda para clasificar imagenes utilizando capas convolucionales. Realice las siguientes actividades vistas en clase:

1. (2 puntos) Cargar y normalizar los datos.

2. (2 puntos) Explorar los datos

3. (2 puntos) Definir la red convolucional

4. (2 puntos) Definir los hiperparametros, por ejemplo, funcion de perdida, el optimizador, entre otros.

5. (2 puntos) Entrenar la red

6. (3 puntos) Evaluar el modelo resultante Accuracy, Precision, Recall y F1-score (investigue como se utilizan estas medidas en clasificacion)

7. (3 puntos) Presentar al menos cuatro conclusiones.

Para el ejercicio **seleccione alguno de los siguientes conjuntos de datos:**

1. Natural Images: https://www.kaggle.com/prasunroy/natural-images
2. Dogs & Cats images: https://www.kaggle.com/chetankv/dogs-cats-images
3. The Oxford-IIIT Pet Dataset: https://www.kaggle.com/devdgohil/the-oxfordiiit-pet-dataset
4. Intel Image Classification: https://www.kaggle.com/puneet6060/intel-image-classification
5. Flower Color Images: https://www.kaggle.com/olgabelitskaya/flower-color-images