Parte 1: Análisis de la Base de Datos

Objetivo

Realizar un análisis exhaustivo de una base de datos antes de utilizarla para entrenar una red neuronal de clasificación.

Consigna:

Selección de la Base de Datos:

Elige una base de datos adecuada para un problema de clasificación. Asegúrate de que contenga al menos una columna objetivo y varias columnas con características.

Análisis general de los Datos:

1. Describe cada columna del conjunto de datos:
    - ¿Qué representa?
    - ¿Es una variable categórica, continua, discreta, etc.?
2. Análisis de Correlaciones:
   Usa correlaciones para evaluar si las características son relevantes para la tarea de clasificación. Identifica las relaciones entre las características y la columna objetivo. ¿Cuáles parecen ser las más influyentes?
3. Análisis de Factibilidad:
    - ¿Es esta base de datos adecuada para entrenar una red neuronal de clasificación? Justifica tu respuesta en base a los datos.
    - Explica el propósito de entrenar la red. ¿Qué intentará predecir? ¿Cuál es el objetivo del modelo?
4. Datos Atípicos y Limpieza de Datos:
    - Identifica datos atípicos (outliers) en las columnas relevantes.
    - ¿Es necesario limpiarlos? Si decides hacerlo, justifica tu decisión y describe cómo lo harías. Si no, explica por qué preferís dejarlos.
5. Transformaciones Preliminares:
    - Realiza las transformaciones necesarias a los datos, por ejemplo, normalización, conversión de categorías a valores numéricos, etc.
    - Explica por qué estas transformaciones son necesarias.

Parte 2: Desarrollo de la Red Neuronal

Objetivo:

Implementar una red neuronal desde cero utilizando numpy, y luego analizar su rendimiento.

Consigna:

1. Arquitectura de la Red:
    - Dibuja la arquitectura de la red neuronal.
    - ¿Cuántas capas tendrá la red?
    - ¿Cuántas neuronas habrá en cada capa?
    - ¿Qué función de activación utilizará cada capa? Explica por qué elegiste esas funciones de activación.
2. Implementación en numpy:
   Implementa tu red neuronal utilizando solo la librería numpy. No está permitido el uso de librerías de machine learning para esta parte.
   Asegúrate de incluir:
    - Inicialización de los pesos.
    - Cálculo de las activaciones para cada capa.
    - Función de costo.
    - Algoritmo de retropropagación (backpropagation) y ajuste de los pesos mediante descenso por gradiente estocástico.
3. Entrenamiento y Evaluación:
    - Entrena tu red con la base de datos seleccionada.
    - Traza las curvas de precisión (accuracy) y la función de pérdida para el set de entrenamiento y el set de validación a lo largo de las iteraciones (epochs).
4. Análisis de Overfitting:
    - Analiza si tu modelo presenta overfitting.
    - ¿Cómo se puede identificar el punto en el que el modelo empieza a sobreajustarse a los datos de entrenamiento? ¿Cómo lo manejarías?

Parte 3: Comparación con scikit-learn

Objetivo:

Implementar la misma red neuronal utilizando scikit-learn y comparar los resultados con la implementación en numpy.

Consigna:

1. Implementación en scikit-learn:
    - Utiliza scikit-learn para implementar una red neuronal similar a la que desarrollaste manualmente. Asegúrate de mantener la misma cantidad de capas y neuronas, y utilizar las mismas funciones de activación.
2. Comparación de Rendimiento:
   Compara el rendimiento de la red neuronal implementada en numpy con la que implementaste en scikit-learn.
    - ¿En qué aspectos los resultados son similares?
    - ¿En qué aspectos son diferentes?
      Compara las curvas de entrenamiento y validación, el tiempo de ejecución, y cualquier otro aspecto relevante.

Parte 4: Conclusión Final

Incluí una sección final en tu reporte donde reflexiones sobre todo el proceso. ¿Qué aprendiste sobre la construcción de redes neuronales desde cero? ¿Cuáles son las ventajas y desventajas de hacerlo manualmente versus usar una librería como scikit-learn?