Programación en Python: Manejo de DataFrame

y Descenso del Gradiente

Alexander Oviedo Fadul

Ingeniería de Sistemas, Corporación Universitaria Minuto de Dios – UNIMINUTO

NRC 198: Deep Learning

Profesor: Juan Carlos Valencia

30 de marzo de 2026

Programación en Python: Manejo de DataFrame y Descenso del Gradiente

Introducción

El presente documento consolida los entregables de la actividad de evaluación correspondiente a la Semana 3 de la asignatura Deep Learning (NRC 198). La actividad se centra en dos ejes temáticos fundamentales para el aprendizaje automático: el descenso del gradiente como algoritmo de optimización para redes neuronales, y la manipulación de DataFrames como estructura esencial para el preprocesamiento de datos.

Según Fandango (2018), el descenso del gradiente "se utiliza para alcanzar el mínimo de la función de error, para actualizar los pesos y el bias o sesgo en el entrenamiento de la red neuronal profunda artificial" (p. 78). Por otro lado, Galea y Capelo (2018) subrayan la importancia del manejo de DataFrames como paso previo a la conversión de datos en tensores, que son las estructuras numéricas que las redes neuronales procesan.

Desarrollo de la Actividad

La actividad se desarrolló en dos notebooks de Jupyter ejecutados en Google Colab, respondiendo a las dos preguntas orientadoras planteadas. A continuación se describen los resultados obtenidos en cada uno.

Notebook 1: Descenso del Gradiente

En este notebook se implementó el descenso del gradiente para la función $f(x) = x^2 + 1$ (ejemplo base del docente) y se resolvió el ejercicio propuesto: implementar el descenso del gradiente para la función $f(x) = x^2 + 5x + 6$, cuya derivada es $f'(x) = 2x + 5$ y cuyo mínimo teórico se encuentra en $x = -2.5$, donde $f(-2.5) = -0.25$.

Se realizaron tres pruebas con diferentes combinaciones de parámetros (punto inicial, precisión y learning rate), confirmando en todos los casos la convergencia al mínimo teórico. Los gráficos se presentaron con colores distintos a los del ejemplo (verde para la función, rojo para los pasos del gradiente), como fue solicitado. La visualización muestra claramente cómo el algoritmo desciende paso a paso por la superficie de la función hasta alcanzar el valle mínimo.

Tabla 1
Resultados de las pruebas del descenso del gradiente para $f(x) = x^2 + 5x + 6$

Prueba	**x_inicio**	**Precisión**	**Learning Rate**	**x mínimo encontrado**
1	8	0.0001	0.05	≈ -2.5000
2	-8	0.01	0.03	≈ -2.5000
3	0	0.001	0.1	≈ -2.5000

Nota. Elaboración propia. El mínimo teórico es x = -2.5, f(-2.5) = -0.25.

Enlace Google Colab — Notebook 1 (Descenso del Gradiente):

https://colab.research.google.com/drive/1RgXmsTE9y8aTDWwlrT9qSEKLw2SPK1gn

Notebook 2: Manejo de DataFrame

En este notebook se trabajó la manipulación de DataFrames utilizando la librería Pandas de Python. Se ejecutaron los ejemplos base del docente (lectura del archivo california_housing_test.csv, head, tail, shape, columns, describe) y se respondieron las cuatro preguntas propuestas.

Tabla 2
Resumen de preguntas respondidas en el notebook de DataFrame

Pregunta	**Descripción**	**Comandos principales**
1	Cargar california_housing_train.csv,	pd.read_csv(), head(10), tail(10)

	mostrar primeras y últimas 10 filas	
2	Cargar mnist_test.csv, mostrar primeras y últimas 10 filas	pd.read_csv(), head(10), tail(10)
3	Dimensión, columnas y estadísticos de median_house_value	shape, columns, describe()
4	Eliminar fila 2999, columna population, renombrar columnas a español	drop(index=), drop(columns=), rename()

Nota. Elaboración propia.

Enlace Google Colab — Notebook 2 (DataFrame):

https://colab.research.google.com/drive/1l9g9Q7vaEMDZ-Dr5gCvuNFEb34Ho0ZEL

Respuestas a las Preguntas Orientadoras

¿Cuál es el proceso para aplicar el descenso del gradiente para minimizar el error de aprendizaje de las redes neuronales?

El proceso del descenso del gradiente consta de los siguientes pasos: (1) se define una función de error o pérdida que mide la diferencia entre la predicción del modelo y el valor real; (2) se calculan los pesos iniciales de forma aleatoria; (3) se computa la derivada parcial de la función de error respecto a cada peso, lo que indica la dirección y magnitud del cambio necesario; (4) se actualizan los pesos en la dirección opuesta al gradiente, multiplicando por un factor llamado learning rate; (5) se repite este proceso iterativamente hasta que la diferencia entre actualizaciones consecutivas sea menor que un umbral de precisión definido. Como se demostró en el notebook, este proceso converge al mínimo de la función, que en el caso de redes neuronales corresponde al punto donde el error de predicción es mínimo.

¿Cuál es el proceso para interpretar y manipular un Data Frame?

Un DataFrame es una estructura bidimensional de datos organizada en filas y columnas, similar a una hoja de cálculo. El proceso de manipulación incluye: (1) carga de datos desde archivos CSV usando pd.read_csv(); (2) exploración inicial con head() y tail() para visualizar las primeras y últimas filas; (3) inspección de la dimensión con .shape y de los nombres de columnas con .columns; (4) análisis estadístico descriptivo con .describe(); (5) eliminación de filas y columnas no deseadas con .drop(); y (6) renombrado de columnas con .rename(). En el contexto de Deep Learning, los DataFrames son el primer paso del pipeline de datos: la información se carga, limpia y transforma antes de convertirse en tensores que las redes neuronales pueden procesar.

Conclusiones

El descenso del gradiente es el algoritmo fundamental mediante el cual las redes neuronales aprenden de los datos. Al implementarlo manualmente para la función $f(x) = x^2 + 5x + 6$, se pudo observar cómo el algoritmo converge al mínimo teórico ($x = -2.5$) independientemente del punto de partida, lo cual demuestra su robustez como método de optimización.

Los tres parámetros del descenso del gradiente (punto inicial, precisión y learning rate) tienen un impacto directo en el rendimiento del entrenamiento. Un learning rate demasiado alto puede causar divergencia, mientras que uno muy bajo hace que la convergencia sea excesivamente lenta. Esta comprensión es esencial para el ajuste de hiperparámetros en modelos reales de Deep Learning.

Los DataFrames de Pandas constituyen una herramienta indispensable en el pipeline de datos de cualquier proyecto de aprendizaje profundo. Los comandos de exploración (head, tail, shape, describe) permiten comprender rápidamente la estructura y distribución de los datos antes de transformarlos en tensores para el entrenamiento.

La conexión entre DataFrames y tensores es directa: los datos se cargan y preparan en DataFrames, y luego se convierten en las estructuras numéricas multidimensionales (tensores) que las redes neuronales procesan. Comprender ambos conceptos es requisito para desarrollar modelos de Deep Learning funcionales.

Referencias

Fandango, A. (2018). Mastering TensorFlow 1.x. Advanced Machine Learning and Deep Learning Concepts Using TensorFlow 1.x and Keras (pp. 73-111). Packt Publishing.

Galea, A., y Capelo, L. (2018). Applied Deep Learning with Python: Use Scikit-Learn, TensorFlow, and Keras to Create Intelligent Systems and Machine Learning Solutions (pp. 6-61, 186-194). Packt Publishing.

Hodnett, M., y Wiley, J. (2018). R Deep Learning Essentials: A Step-by-Step Guide to Building Deep Learning Models Using TensorFlow, Keras, and MXNet (pp. 60-96). Packt Publishing.

McKinney, W. (2017). Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython (2ª ed.). O'Reilly Media.