Estudio de los métodos de Gradiente Descendiente y Gradiente Descendente Estocástico en problemas de optimización

Flores Vásquez, Abraham Alejandro
Universidad Centroamericana "Jose Simeón Cañas"
00067323@uca.edu.sv

Morales Pineda, Alexander Efraín
Universidad Centroamericana "Jose Simeón Cañas"
00024123@uca.edu.sv

Iraheta Monterrosa, Diego Alejandro
Universidad Centroamericana "Jose Simeón Cañas"
00041923@uca.edu.sv

Tovar Jovel, Cesar Isaac
Universidad Centroamericana "Jose Simeón Cañas"
00016023@uca.edu.sv

RESUMEN

***Abstract*—El presente trabajo tiene como objetivo analizar y comparar dos técnicas fundamentales del aprendizaje supervisado: la regresión lineal multivariada y la regresión logística multiclase. Para el caso de la regresión lineal, se implementa el método de Gradiente Descendente con el fin de ajustar un modelo predictivo sobre variables continuas. Por otra parte, para la regresión logística multiclase, se utiliza el método de Gradiente Descendente Estocástico (SGD), dada su eficiencia computacional al trabajar con grandes volúmenes de datos. Ambas implementaciones incluyen procesos de preprocesamiento, entrenamiento, evaluación y visualización de resultados, los cuales son documentados con herramientas computacionales adecuadas. El estudio busca no solo comprender el funcionamiento numérico de ambos algoritmos, sino también evaluar su desempeño en distintos conjuntos de datos reales.**

PALABRAS CLAVE

***Index Terms*—Regresión lineal multivariada, regresión logística multiclase, Gradiente Descendente, Gradiente Descendente Estocástico, aprendizaje automático, optimización numérica.**

I. INTRODUCCIÓN

En el contexto del aprendizaje automático supervisado, los métodos de regresión y clasificación representan herramientas fundamentales para la predicción y la toma de decisiones basada en datos. La regresión lineal multivariada permite modelar relaciones entre múltiples variables independientes y una variable dependiente continua, siendo ampliamente utilizada en problemas de predicción cuantitativa.[4] Para el ajuste de los parámetros del modelo, se implementa el método de Gradiente Descendente, el cual optimiza una función de costo iterativamente a través de pasos proporcionales al gradiente negativo.

Por otro lado, cuando el objetivo consiste en asignar entradas a categorías discretas, se recurre a técnicas de clasificación, como la regresión logística. En particular, para problemas con múltiples clases, la regresión logística multiclase se presenta como una extensión natural del modelo binario.[5] En este trabajo, se opta por implementar este modelo utilizando el método de Gradiente Descendente Estocástico (SGD), el cual ofrece ventajas computacionales significativas al actualizar los parámetros del modelo a partir de muestras individuales o pequeños subconjuntos de datos.

Ambos métodos comparten fundamentos numéricos comunes, pero difieren en sus objetivos, funciones de costo y estrategias de optimización. El propósito de este estudio es explorar teórica y

computacionalmente estas metodologías, analizando su comportamiento, convergencia, ventajas y limitaciones, así como su aplicabilidad en distintos escenarios reales.

II. Preliminares Matemáticos

A. Expansión de Taylor para funciones multivariables

La expansión de Taylor permite aproximar una función diferenciable en un entorno cercano a un punto dado. [6] Para una función $L : \mathbb{R}^n \to \mathbb{R}$, la expansión de primer orden alrededor de un punto θ se expresa como:

$$L(\theta + \Delta\theta) \approx L(\theta) + \nabla L(\theta)^T \Delta\theta$$

donde:

- $\nabla L(\theta)$ es el vector gradiente, que contiene las derivadas parciales de L respecto a cada componente de θ.
- $\Delta\theta$ es el vector que representa el pequeño cambio en los parámetros.
- La transpuesta ^{T} se usa para realizar el producto escalar entre los vectores.

Esta aproximación es fundamental en algoritmos de optimización, ya que describe cómo cambia la función alrededor de un punto y permite decidir la dirección del siguiente paso.

B. Gradiente: definición e interpretación

El gradiente de una función $L : \mathbb{R}^n \to \mathbb{R}$ está definido como:

$$\nabla L(\theta) = \left(\frac{\partial L}{\partial \theta_1}, \frac{\partial L}{\partial \theta_2}, \ldots, \frac{\partial L}{\partial \theta_n} \right)^T$$

Cada componente corresponde a la derivada parcial respecto a uno de los parámetros. Geométricamente, el gradiente apunta hacia la dirección de máximo crecimiento de la función en un punto dado. Esta propiedad es clave para los métodos de descenso, ya que para minimizar la función se avanza en la dirección opuesta al gradiente. [7]

C. Concepto de óptimos: mínimos y máximos

En optimización, un punto θ^* es un *punto crítico* si cumple que:

$$\nabla L(\theta^*) = 0$$

Este punto puede ser:

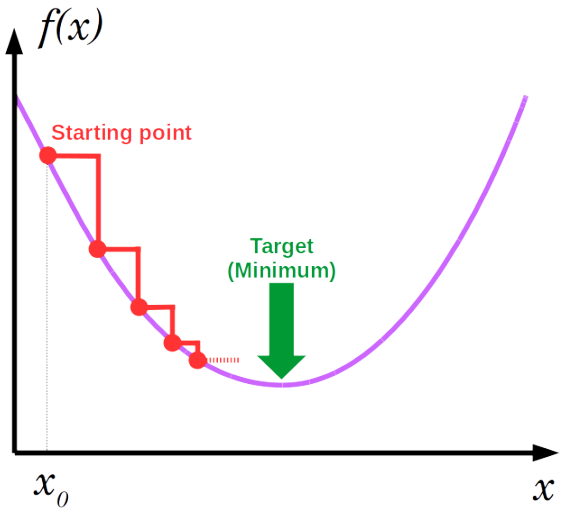
- Un mínimo local, si en un entorno de θ^* la función toma valores mayores o iguales.
- Un máximo local, si la función toma valores menores o iguales en un entorno cercano.
- Un punto de silla, si no es ni máximo ni mínimo.

En el caso de funciones convexas, cualquier mínimo local es también un mínimo global, lo cual simplifica considerablemente los problemas de optimizacion al descartar la existencia de algún otro posible mínimo en la función.

III. Método del Gradiente Descendiente

El Gradiente Descendiente es un método esencial ya que es la base de todos los algoritmos de entrenamiento de modelos Machine Learning (como las Redes Neuronales, Convolucionales, Recurrentes y Transformer).

Este algoritmo de optimización permite encontrar de forma automática el mínimo de una función. Para ello hace uso de el gradiente (o derivada) de dicha función que permite guiar al algoritmo para, de manera progresiva, acercarse al mínimo ideal de dicha función.[9]



A. Deducción del método

Sea $L(\theta) : \mathbb{R}^n \to \mathbb{R}$ una función de costo diferenciable que deseamos minimizar, donde $\theta \in \mathbb{R}^n$ representa el vector de parámetros del modelo. La función $L(\theta)$ mide el error o la diferencia entre las predicciones del modelo y los valores reales.

El método de Gradiente Descendente se basa en una aproximación de primer orden mediante la expansión de Taylor para funciones multivariables:

$$L(\theta + \Delta\theta) \approx L(\theta) + \nabla L(\theta)^T \Delta\theta$$

Aquí, $\nabla L(\theta)$ es el vector gradiente de la función de costo, que contiene las derivadas parciales respecto a los parámetros θ, y $\Delta\theta$ es el cambio aplicado a dichos parámetros. El símbolo T indica la **transpuesta** del vector gradiente. Dado que tanto $\nabla L(\theta)$ como $\Delta\theta$ son vectores columna, se utiliza la transpuesta para convertir el gradiente en un vector fila, lo que permite realizar un producto escalar entre ambos:

$$\nabla L(\theta)^T \Delta\theta \in \mathbb{R}$$

Para minimizar la función, se elige $\Delta\theta$ como la dirección opuesta al gradiente, ya que el gradiente apunta hacia la dirección de máximo crecimiento de la función. Por tanto, se define:

$$\Delta\theta = -\alpha \nabla L(\theta)$$

donde $\alpha > 0$ es la **tasa de aprendizaje** o *learning rate*, que determina el tamaño del paso.

Reemplazando en la regla de actualización, se obtiene:

$$\theta^{(k+1)} = \theta^{(k)} - \alpha \nabla L(\theta^{(k)})$$

Este procedimiento se repite iterativamente hasta que el gradiente sea cercano a cero o se alcance un número máximo de iteraciones, con el objetivo de aproximarse a un mínimo local o global de la función de costo.

B. Demostración de orden de convergencia

Supongamos que la función de costo $L : \mathbb{R}^n \to \mathbb{R}$ es convexa y que su gradiente es L_g-Lipschitz continuo, condición la cual implica que los cambios en el gradiente estén acotados por un factor proporcional a los cambios en los parámetros, es decir:

$$\|\nabla L(\theta_1) - \nabla L(\theta_2)\| \leq L_g \|\theta_1 - \theta_2\| \quad \forall \theta_1, \theta_2 \in \mathbb{R}^n.$$

La razón para requerir esta condición es que garantiza que la función no varía de manera demasiado abrupta y que los pasos del método de Gradiente Descendiente son controlados y estables. Sin esta propiedad, no es posible asegurar una tasa de convergencia teórica, y el algoritmo podría divergir o no comportarse de manera predecible. En términos prácticos, la continuidad Lipschitz del gradiente está asociada a que la función objetivo tenga una curvatura bien comportada en todo el dominio de optimización.

Bajo esta condición, se puede demostrar que el método de Gradiente Descendiente con tasa de aprendizaje $0 < \alpha < \frac{2}{L_g}$ satisface la siguiente desigualdad en cada iteración:

$$L(\theta^{(k+1)}) - L(\theta^*) \leq (1 - \alpha\mu)[L(\theta^{(k)}) - L(\theta^*)]$$

donde:

- θ^* es el punto óptimo global.
- μ es la constante de convexidad fuerte (si aplica).

En el caso general (sin convexidad fuerte), la convergencia es de tipo sublineal y se obtiene que:

$$L(\theta^{(k)}) - L(\theta^*) = \mathcal{O}\left(\frac{1}{k}\right).$$

Sin embargo, cuando L es convexa y fuertemente convexa, la convergencia es lineal:

$$\|\theta^{(k)} - \theta^*\| \leq C\rho^k$$

con $0 < \rho < 1$ y C una constante positiva dependiente de la condición inicial.

Esto significa que la distancia al óptimo disminuye aproximadamente en proporción constante en cada iteración, lo que caracteriza una **convergencia lineal**. Por esta razón, el método de Gradiente

Descendiente es considerado eficiente pero de orden de convergencia bajo en comparación con métodos como Newton-Raphson, que alcanzan convergencia cuadrática cerca del óptimo.

C. Condiciones de estabilidad

La estabilidad del método de Gradiente Descendiente está determinada principalmente por la elección de la tasa de aprendizaje α. Para que el algoritmo sea estable y converja hacia un óptimo, es necesario que la función de costo $L(\theta)$ sea convexa y que su gradiente sea L_g-Lipschitz continuo, como se discutió anteriormente.

La condición clásica de estabilidad establece que:

$$0 < \alpha < \frac{2}{L_g}$$

Este rango asegura que cada actualización reduce la función de costo. Si α es demasiado grande, el algoritmo puede oscilar alrededor del mínimo o incluso divergir, mientras que un α demasiado pequeño ralentiza significativamente la convergencia.

En la práctica, la elección adecuada de α puede requerir experimentación o el uso de técnicas adaptativas, como reducción progresiva de la tasa de aprendizaje o algoritmos más avanzados que ajusten dinámicamente este valor.

D. Ventajas y desventajas

Ventajas:

- **Simplicidad:** El método es sencillo de entender e implementar.
- **Escalabilidad:** Funciona bien con conjuntos de datos grandes, especialmente si se combina con variantes estocásticas (SGD).
- **Versatilidad:** Puede aplicarse a una amplia variedad de problemas de optimización convexa y no convexa.
- **Poca demanda computacional:** Solo requiere el cálculo del gradiente, lo que lo hace menos costoso que métodos de segunda orden.

Desventajas:

- **Sensibilidad a la tasa de aprendizaje:** La elección de α es crítica y puede afectar significativamente la convergencia.
- **Convergencia lenta:** La velocidad de convergencia es lineal, más lenta que la de métodos de segunda orden como Newton-Raphson.
- **Mínimos locales:** En problemas no convexos, puede quedar atrapado en óptimos locales o puntos de silla.

E. Ejemplo de uso

Consideremos la función cuadrática:

$$L(\theta) = (\theta - 3)^2$$

Esta función tiene un único mínimo global en $\theta = 3$. El gradiente es:

$$\nabla L(\theta) = 2(\theta - 3)$$

Aplicando el método de Gradiente Descendiente con una tasa de aprendizaje $\alpha = 0.1$ y un valor inicial $\theta^{(0)} = 0$, obtenemos las siguientes iteraciones:

$$\theta^{(1)} = 0 - 0.1 \times (-6) = 0.6,$$

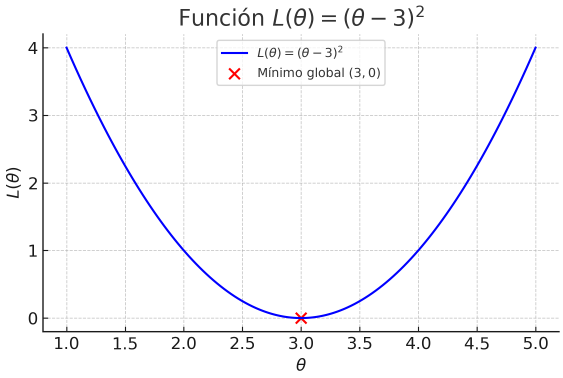
$$\theta^{(2)} = 0.6 - 0.1 \times (-4.8) = 1.08,$$

$$\theta^{(3)} = 1.08 - 0.1 \times (-3.84) = 1.464,$$

$$\theta^{(4)} = 1.464 - 0.1 \times (-3.072) = 1.771,$$

$$\theta^{(5)} = 1.771 - 0.1 \times (-2.458) = 2.016.$$

Se observa cómo θ se aproxima progresivamente al valor óptimo $\theta = 3$ a medida que avanzan las iteraciones. Este ejemplo simple ilustra cómo el método ajusta los parámetros en la dirección que minimiza la función de costo.

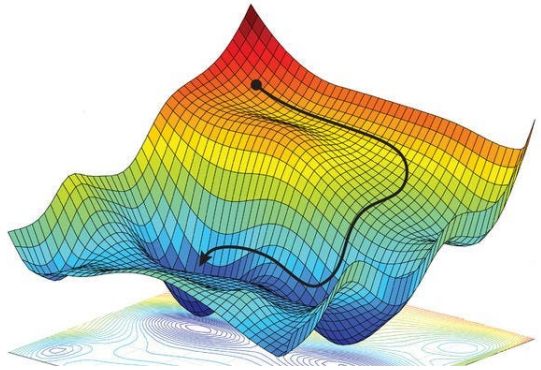


IV. Gradiente Descendiente Estocástico (SGD)

El método de Gradiente Descendiente Estocástico (Stochastic Gradient Descent, SGD) es una variante del Gradiente Descendiente clásico diseñada para mejorar la eficiencia computacional, especialmente cuando se trabaja con grandes conjuntos de datos. A diferencia del método tradicional, que calcula el gradiente utilizando la totalidad de las muestras en cada iteración, el SGD actualiza los parámetros utilizando solo una muestra (o un pequeño minibatch) por vez.[10]

Esta característica hace que el SGD sea mucho más rápido en cada iteración, aunque introduce ruido en la trayectoria de convergencia, ya que cada paso puede fluctuar debido a la variabilidad inherente de los datos individuales.

En la práctica, el SGD es ampliamente utilizado para entrenar modelos de clasificación, como la regresión logística multiclase, redes neuronales y otros algoritmos de aprendizaje automático.



A. Deducción del método

Sea $\{(x_i, y_i)\}_{i=1}^{m}$ un conjunto de entrenamiento, donde:

- $x_i \in \mathbb{R}^n$ es la entrada (vector de características),
- y_i es la etiqueta de clase correspondiente,
- m es el número total de muestras.

En el método clásico, el gradiente de la función de costo total se calcula como:

$$\nabla L(\theta) = \frac{1}{m} \sum_{i=1}^{m} \nabla L_i(\theta)$$

donde $L_i(\theta)$ es la pérdida asociada a la muestra i.

En el método estocástico, en lugar de calcular la suma completa, se selecciona aleatoriamente una muestra (x_j, y_j) y se actualizan los parámetros usando solo su gradiente:

$$\theta^{(k+1)} = \theta^{(k)} - \alpha \nabla L_j(\theta^{(k)})$$

donde:

- α es la tasa de aprendizaje,
- $j \in \{1, 2, \ldots, m\}$ es el índice aleatorio de la muestra elegida en la iteración k.

Este esquema introduce ruido en cada paso, pero en promedio sigue aproximándose a la dirección del gradiente real, permitiendo la convergencia al mínimo esperado.

B. Demostración de orden de convergencia

El análisis de la convergencia del Gradiente Descendiente Estocástico (SGD) difiere del método clásico debido a la naturaleza ruidosa de las actualizaciones. Bajo ciertas condiciones, se puede demostrar que el SGD converge en promedio hacia un óptimo.

Supongamos que la función de costo es convexa y que el gradiente estocástico es un estimador insesgado del gradiente verdadero:

$$\mathbb{E}[\nabla L_j(\theta)] = \nabla L(\theta)$$

Además, se requiere que la varianza del gradiente esté acotada:

$$\mathbb{E}\|\nabla L_j(\theta) - \nabla L(\theta)\|^2 \leq \sigma^2$$

Bajo estas condiciones y utilizando una tasa de aprendizaje decreciente $\alpha_k = \frac{1}{k}$, se puede demostrar que:

$$\mathbb{E}[L(\theta^{(k)}) - L(\theta^*)] = \mathcal{O}\left(\frac{1}{k}\right)$$

Esto implica que la convergencia esperada del SGD es **sublineal**, es decir, la precisión mejora lentamente a medida que aumentan las iteraciones, pero sigue siendo efectiva para grandes conjuntos de datos debido a la rapidez de cada paso individual.

C. Condiciones de estabilidad

Para que el SGD sea estable y converja, se deben cumplir ciertas condiciones relacionadas con la tasa de aprendizaje. A diferencia del método clásico, donde la tasa de aprendizaje puede ser constante bajo ciertas condiciones, en el SGD es recomendable que la tasa decrezca con el número de iteraciones para mitigar el ruido estocástico.

Una condición típica es que la secuencia $\{\alpha_k\}$ de tasas de aprendizaje satisfaga:

- $\sum_{k=1}^{\infty} \alpha_k = \infty$ (suma infinita),
- $\sum_{k=1}^{\infty} \alpha_k^2 < \infty$ (suma finita de los cuadrados).

Por ejemplo, una elección común es $\alpha_k = \frac{1}{k}$. Estas condiciones aseguran que el método converge casi seguramente al óptimo bajo hipótesis estándar de convexidad y acotamiento del gradiente.

En la práctica, también es habitual usar una tasa de aprendizaje constante durante un número determinado de iteraciones y luego reducirla progresivamente para estabilizar la convergencia.

D. Ventajas y desventajas

Ventajas:

- **Eficiencia computacional:** Cada actualización requiere solo una muestra o un pequeño mini-batch, lo que reduce significativamente el costo computacional por iteración.
- **Escalabilidad:** Es ideal para conjuntos de datos muy grandes donde calcular el gradiente completo es inviable.
- **Capacidad de escape:** Al ser estocástico, puede escapar de mínimos locales poco profundos en problemas no convexos.
- **Simplicidad de implementación:** Su estructura es simple y fácil de implementar en la práctica.

Desventajas:

- **Convergencia ruidosa:** La naturaleza aleatoria introduce oscilaciones, dificultando una convergencia suave hacia el óptimo.
- **Dependencia de la tasa de aprendizaje:** Elegir la secuencia correcta de tasas de aprendizaje es crucial y puede requerir ajuste experimental.
- **Convergencia más lenta:** Aunque cada iteración es rápida, pueden ser necesarias muchas más iteraciones para alcanzar una precisión deseada en comparación con el método clásico.

E. Ejemplo práctico

Consideremos un problema de clasificación binaria con la función de costo logística para una sola muestra (x_j, y_j):

$$L_j(\theta) = -y_j \log(h_\theta(x_j)) - (1 - y_j) \log(1 - h_\theta(x_j))$$

donde:

- $h_\theta(x_j) = \frac{1}{1+e^{-\theta^T x_j}}$ es la función sigmoide,
- $y_j \in \{0, 1\}$ es la etiqueta de clase,
- $x_j \in \mathbb{R}^n$ es el vector de características.

El gradiente para una sola muestra es:

$$\nabla L_j(\theta) = (h_\theta(x_j) - y_j)x_j$$

Usando SGD, la actualización para una muestra seleccionada aleatoriamente sería:

$$\theta^{(k+1)} = \theta^{(k)} - \alpha(h_\theta(x_j) - y_j)x_j$$

Ejemplo numérico:

Supongamos:

- $x_j = (1, 2)$,
- $y_j = 1$,
- $\theta^{(0)} = (0, 0)$,
- $\alpha = 0.1$.

Cálculo:

1) $h_\theta(x_j) = \frac{1}{1+e^0} = 0.5$,
2) Gradiente: $(0.5 - 1)(1, 2) = (-0.5, -1.0)$,
3) Actualización: $\theta^{(1)} = (0, 0) - 0.1 \times (-0.5, -1.0) = (0.05, 0.1)$.

Este ejemplo muestra cómo, a partir de una sola muestra, se actualizan los parámetros acercándose al óptimo de manera incremental.

V. Comparación entre Gradiente Descendiente y Gradiente Descendiente Estocástico

A continuación, se presenta una comparación entre el método de Gradiente Descendiente (GD) y su variante estocástica (SGD), destacando sus principales características y diferencias clave:

- **Cálculo del gradiente:**

  - **GD:** Calcula el gradiente completo utilizando todas las muestras del conjunto de datos en cada iteración.
  - **SGD:** Calcula el gradiente utilizando solo una muestra (o un minibatch) seleccionada aleatoriamente en cada iteración.
- **Costo computacional:**
  - **GD:** Costoso por iteración para conjuntos de datos grandes, ya que requiere procesar todos los datos en cada paso.
  - **SGD:** Bajo costo por iteración, ideal para datasets muy grandes.
- **Convergencia:**
  - **GD:** Convergencia estable y suave, generalmente más rápida en número de iteraciones totales.
  - **SGD:** Convergencia más ruidosa debido a la variabilidad de las muestras, pero eficiente en términos de tiempo de cómputo.
- **Estabilidad:**
  - **GD:** Permite usar una tasa de aprendizaje constante bajo ciertas condiciones.
  - **SGD:** Requiere una tasa de aprendizaje decreciente o técnicas adaptativas para asegurar estabilidad.
- **Aplicaciones:**
  - **GD:** Adecuado para problemas donde el conjunto de datos es pequeño o moderado y el coste por iteración no es una limitación.
  - **SGD:** La opción preferida para problemas de aprendizaje automático con grandes volúmenes de datos, como en el entrenamiento de modelos de deep learning.

En resumen, aunque ambos métodos comparten la misma filosofía de optimización basada en el gradiente, sus diferencias en la implementación y comportamiento los hacen más o menos adecuados según el contexto y las necesidades computacionales.[11]

VI. Conclusiones

En este trabajo se han estudiado y comparado dos métodos fundamentales para la optimización numérica en aprendizaje automático: el Gradiente Descendiente y el Gradiente Descendiente Estocástico (SGD). A través del análisis teórico y ejemplos prácticos, se ha mostrado cómo ambos métodos utilizan la información del gradiente para minimizar funciones de costo, pero con enfoques distintos en cuanto a la cantidad de datos procesados por iteración.

En particular, se ha implementado el Gradiente Descendiente para resolver un problema de Regresión Lineal Multivariada, donde la función objetivo es suave y convexa, lo que garantiza una convergencia estable hacia la solución óptima. Por otro lado, se ha utilizado el método SGD para la Regresión Logística Multiclase, un escenario más complejo y común en clasificación supervisada, donde el SGD demuestra su eficacia computacional al manejar grandes volúmenes de datos y actualizaciones rápidas.

La elección entre uno u otro método dependerá de las características específicas del problema: el Gradiente Descendiente es más adecuado para tareas de regresión sobre conjuntos de datos moderados, mientras que el SGD resulta ser la mejor opción para problemas de clasificación multiclase en entornos de big data o cuando la eficiencia computacional es crítica.

Ambos métodos siguen siendo pilares fundamentales en la optimización moderna y continúan evolucionando mediante variantes más sofisticadas que abordan sus limitaciones y mejoran su rendimiento.

Agradecimientos

Los autores desean expresar su agradecimiento a la Universidad Centroamericana "José Simeón Cañas" y al Departamento de Matemática por el apoyo y la formación recibida durante la realización de este proyecto académico. Asimismo, se reconoce la valiosa orientación del personal docente y los recursos proporcionados para el desarrollo de este trabajo.

References

[1] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.

[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.

[3] J. Nocedal and S. Wright, *Numerical Optimization*, 2nd ed., Springer, 2006.

[4] IBM, “Regresión lineal múltiple,” *IBM Documentation*, [En línea]. Disponible en: https://www.ibm.com/docs/es/cognos-analytics/11.1.x?topic=tests-multiple-linear-regression

[5] Amazon Web Services, “¿Qué es la regresión logística?,” *AWS*, [En línea]. Disponible en: https://aws.amazon.com/es/what-is/logistic-regression/

[6] OpenStax, “6.3 Series de Taylor y de Maclaurin,” *Cálculo, Volumen 2*, [En línea]. Disponible en: https://openstax.org/books/clculo-volumen-2/pages/6-3-series-de-taylor-y-maclaurin

[7] Khan Academy, “El gradiente,” *Khan Academy*, [En línea]. Disponible en: https://es.khanacademy.org/math/multivariable-calculus/multivariable-derivatives/partial-derivative-and-gradient-articles/a/the-gradient

[8] UNAM, “Series de Taylor,” *PAPIME 2020, Geofísica UNAM*, [En línea]. Disponible en: http://gmc.geofisica.unam.mx/papime2020/index.php/articulos/10-series-de-taylor

[9] Khan Academy, “¿Qué es el descenso del gradiente?,” *Khan Academy*, [En línea]. Disponible en: https://es.khanacademy.org/math/multivariable-calculus/applications-of-multivariable-derivatives/optimizing-multivariable-functions/a/what-is-gradient-descent

[10] Scikit-learn, “Descent gradient estocástico (SGD),” *Scikit-learn (traducción)*, [En línea]. Disponible en: https://scikit--learn-org.translate.goog/stable/modules/sgd.html?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=tc

[11] Toolify, “Descenso del Gradiente Estocástico vs Descenso del Gradiente: ¿Cuál es el mejor?,” *Toolify.ai*, 4 de marzo de 2024. [En línea]. Disponible en: https://www.toolify.ai/es/ai-news-es/descenso-del-gradiente-estocstico-vs-descenso-del-gradiente/-cual-es-el-mejor-2361431

[12] J. Stewart, *Cálculo de una variable: Trascendentes tempranas*, 8ª ed., México: Cengage Learning, 2016.

[13] G. Strang, *Introduction to Linear Algebra*, 5th ed., Wellesley-Cambridge Press, 2016.

[14] I. Goodfellow, Y. Bengio, y A. Courville, *Deep Learning*, MIT Press, 2016. [En línea]. Disponible en: https://www.deeplearningbook.org/

[15] S. Boyd y L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004. [En línea]. Disponible en: https://web.stanford.edu/~boyd/cvxbook/

[16] T. M. Apostol, *Calculus, Volume II: Multi-Variable Calculus and Linear Algebra with Applications*, 2nd ed., Wiley, 1969.

[17] CRAI UCA El Salvador, *Centro de Recursos para el Aprendizaje y la Investigación*, [En línea]. Disponible en: https://crai.uca.edu.sv