SEMINARIO DE SOLUCION DE PROBLEMAS DE INTELIGENCIA ARTIFICIAL 2

MAESTROS:

DIEGO ALBERTO OLIVA NAVARRO

DIEGO CAMPOS PENA

ALUMNO:

EDUARDO BLANCO GONZALEZ

Tarea: Práctica 1, ejercicio 3.

OBJETIVO:

Crea un software que por medio del descenso del gradiente sea capaz de optimizar la función adjunta en la imagen en los limites -1 a 1. Tiene que ser capaz de cambiar el lr (learning rate). Los valores iniciales tienen que ser de forma aleatoria.

Reporte:
-Introducción (explicar el descenso del gradiente, su funcionamiento)
-Desarrollo
-Resultados: Valor de X1 y X2 como mejor solución, grafica de convergencia del error.
-Conclusiones

Descenso del Gradiente

El descenso del gradiente es un algoritmo de optimización utilizado en el campo de la inteligencia artificial, específicamente en el entrenamiento de modelos de aprendizaje automático, como redes neuronales. Su objetivo principal es minimizar una función de costo, que representa la diferencia entre las predicciones del modelo y los valores reales de los datos de entrenamiento.

El proceso del descenso del gradiente implica ajustar iterativamente los parámetros del modelo en la dirección que reduce la función de costo. Esto se logra calculando el gradiente de la función de costo con respecto a los parámetros del modelo y luego actualizando los parámetros en la dirección opuesta al gradiente. En otras palabras, se "desciende" por la pendiente de la función de costo hacia el mínimo.

El descenso del gradiente viene en varias variantes, como el descenso del gradiente estocástico (SGD), el descenso del gradiente mini batch y el descenso del gradiente por lotes (batch gradient descent). Cada variante tiene sus propias características y se adapta mejor a diferentes tipos de conjuntos de datos y problemas de optimización.

En resumen, el descenso del gradiente es una técnica fundamental en el entrenamiento de modelos de aprendizaje automático que permite ajustar

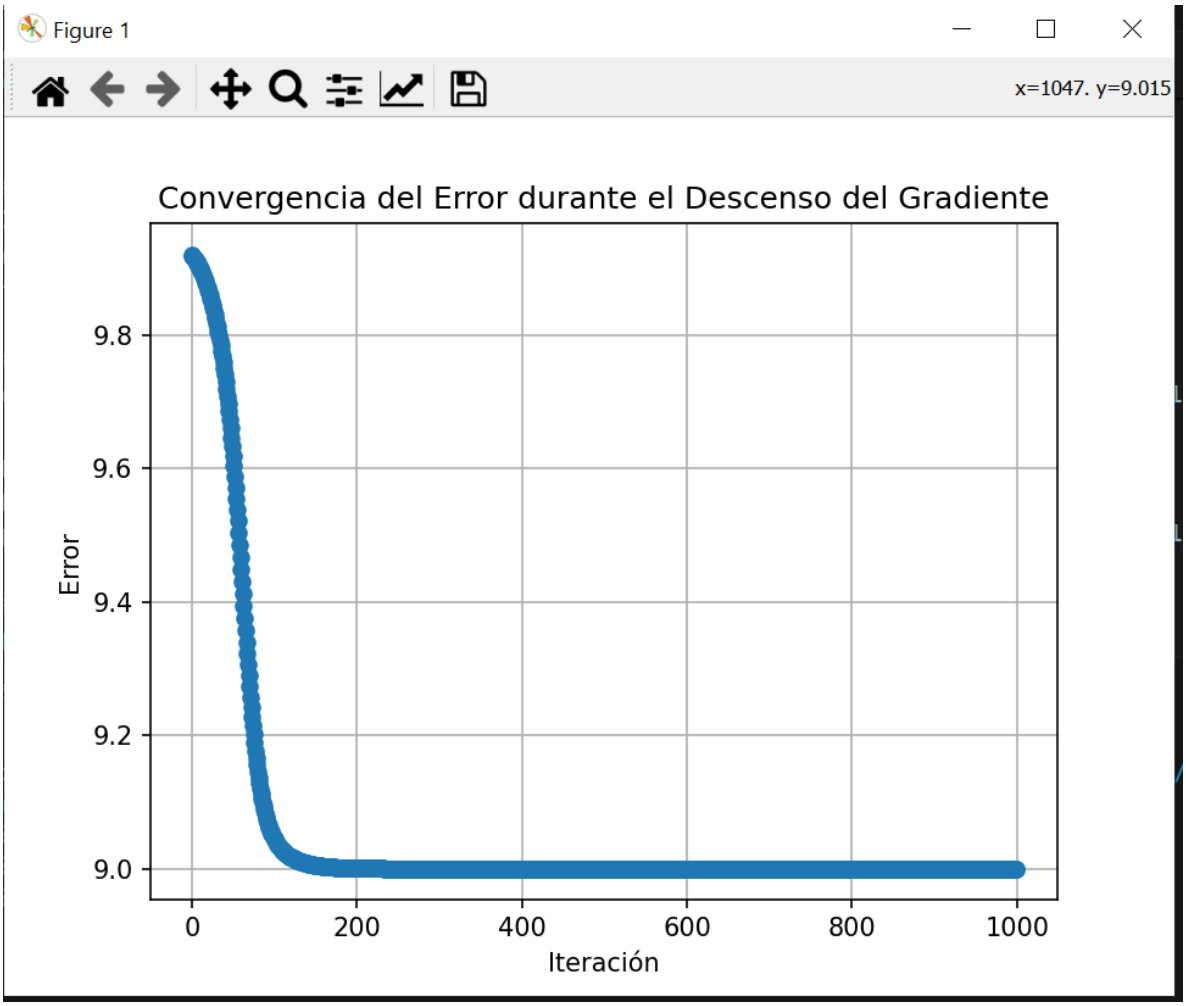
los parámetros del modelo para que se ajusten mejor a los datos de entrenamiento y, en última instancia, mejoren su capacidad para hacer predicciones precisas.

Código

```python
import random
import math
import matplotlib.pyplot as plt

# Definir la función objetivo
def objective_function(x1, x2):
    return 10 - math.exp(-(x1**2 + 3*x2**2))

# Gradiente de la función objetivo
def gradient(x1, x2):
    grad_x1 = 2 * x1 * math.exp(-(x1**2 + 3*x2**2))
    grad_x2 = 6 * x2 * math.exp(-(x1**2 + 3*x2**2))
    return grad_x1, grad_x2

def gradient_descent(lr, max_iterations, tolerance):
    # Inicializar valores aleatorios para x1 y x2 en el rango [-1, 1]
    x1 = random.uniform(-1, 1)
    x2 = random.uniform(-1, 1)

    errors = []  # Lista para almacenar los errores en cada iteración

    iteration = 0
    while iteration < max_iterations:
        # Calcular el gradiente
        grad_x1, grad_x2 = gradient(x1, x2)

        # Actualizar los valores de x1 y x2 usando el descenso del gradiente
        x1 -= lr * grad_x1
        x2 -= lr * grad_x2

        # Calcular el cambio en la función objetivo
        current_value = objective_function(x1, x2)

        # Calcular el error y agregarlo a la lista
        error = abs(current_value)
        errors.append(error)

        # Verificar el criterio de parada
        if error < tolerance:
            break

        iteration += 1

    return x1, x2, errors

# Parámetros
learning_rate = 0.01
max_iterations = 1000
tolerance = 1e-5

# Optimizar la función objetivo utilizando el descenso del gradiente
optimal_x1, optimal_x2, errors = gradient_descent(learning_rate, max_iterations, tolerance)

print("Valor óptimo de x1:", optimal_x1)
print("Valor óptimo de x2:", optimal_x2)
print("Valor óptimo de la función objetivo:", objective_function(optimal_x1, optimal_x2))

# Graficar la convergencia del error
```

```
59    plt.plot(range(len(errors)), errors, linestyle='-', marker='o')
60    plt.xlabel('Iteración')
61    plt.ylabel('Error')
62    plt.title('Convergencia del Error durante el Descenso del Gradiente')
63    plt.grid(True)
64    plt.show()
```

Resultados




```
Valor óptimo de x1: 2.5506895068314666e-09
Valor óptimo de x2: 3.925411625626831e-26
Valor óptimo de la función objetivo: 9.0
```

Conclusión

El código muestra el valor óptimo de las variables de entrada, así como el valor óptimo de la función objetivo. Además, se visualiza la convergencia del error a lo largo de las iteraciones del algoritmo. Esto permite observar cómo el error disminuye con cada iteración, mostrando así el proceso de optimización.

En este código, hemos implementado el algoritmo del descenso del gradiente para minimizar una función objetivo dada. El descenso del gradiente es una técnica fundamental en el campo del aprendizaje automático y la optimización, que busca ajustar iterativamente los parámetros de un modelo en la dirección que reduce una función de costo.

Al ejecutar el código, obtenemos los valores óptimos de las variables de entrada que minimizan la función objetivo. La visualización de la convergencia del error durante las iteraciones nos proporciona una comprensión visual de cómo el algoritmo se acerca al óptimo.

El descenso del gradiente es una herramienta poderosa para encontrar soluciones óptimas en problemas de optimización, y su implementación nos permite entender cómo los modelos de aprendizaje automático pueden ajustarse para mejorar su rendimiento en la resolución de problemas del mundo real.