Documentación: Perceptrón aplicado a comportamiento en páginas web

# 1. Introducción

Se implementa un ejemplo de un modelo de red neuronal simple conocido como Perceptrón, aplicado a un caso relacionado con Ingeniería de Sistemas: predecir si un usuario abandona una página web basándose en su comportamiento (tiempo de navegación, interacción y desplazamiento).

# 2. Objetivo

Diseñar e implementar un modelo Perceptrón que sea capaz de clasificar a los usuarios entre aquellos que permanecen en la página web y aquellos que la abandonan, utilizando un conjunto de datos simulados.

# 3. Variables del modelo

- Tiempo en página (segundos)  
- Scroll al 80% o más (1: sí, 0: no)  
- Clic en algún botón (1: sí, 0: no)  
  
Salida esperada:  
- 1: Usuario se quedó  
- 0: Usuario abandonó

# 4. Código Fuente

|  |
| --- |
| import numpy as np  def step\_function(z):      return 1 if z >= 0 else 0  class Perceptron:      def \_\_init\_\_(self, input\_size, learning\_rate=0.01):          self.weights = np.zeros(input\_size + 1)          self.learning\_rate = learning\_rate      def predict(self, x):          z = np.dot(x, self.weights[1:]) + self.weights[0]          return step\_function(z)      def train(self, X, y, epochs=50):          for epoch in range(epochs):              for inputs, label in zip(X, y):                  prediction = self.predict(inputs)                  error = label - prediction                  self.weights[1:] += self.learning\_rate \* error \* inputs                  self.weights[0] += self.learning\_rate \* error              print(f"Época {epoch + 1}: Pesos = {self.weights}")  # Datos de entrenamiento  X = np.array([      [5, 0, 0],    # Poco tiempo, sin scroll, sin clic = abandono      [20, 1, 1],   # Más tiempo, scroll y clic = se quedó      [10, 0, 1],   # Algo de tiempo, hizo clic = se quedó      [2, 0, 0],    # Muy poco tiempo = abandono      [25, 1, 0],   # Buen tiempo y scroll = se quedó      [3, 0, 0]     # Muy poco tiempo = abandono  ])  y = np.array([0, 1, 1, 0, 1, 0])  # Etiquetas (0 = se fue, 1 = se quedó)  # Entrenar perceptrón  perceptron = Perceptron(input\_size=3)  perceptron.train(X, y, epochs=20)  # Pruebas  test\_samples = np.array([      [15, 1, 1],   # Usuario muy activo      [2, 0, 0],    # Usuario pasivo  ])  print("\nResultados de prueba:")  for inputs in test\_samples:      result = perceptron.predict(inputs)      print(f"Entrada: {inputs} → {'Se quedó' if result == 1 else 'Abandonó'}") |

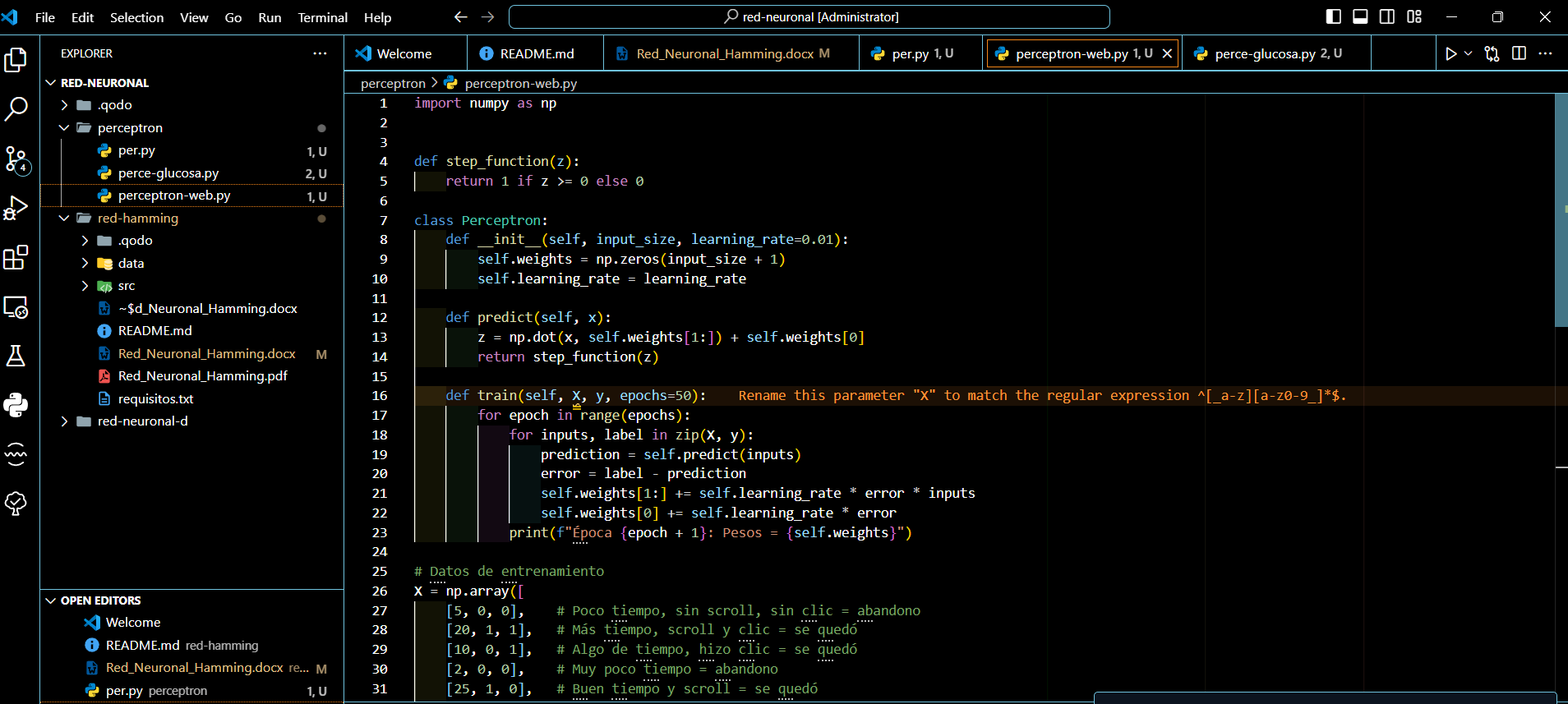
# 5. Conclusiones

El modelo Perceptrón fue capaz de aprender patrones simples de comportamiento en usuarios de páginas web. Aunque es un modelo sencillo, demuestra cómo técnicas básicas de redes neuronales pueden ser aplicadas en proyectos reales de Ingeniería de Sistemas, como análisis de comportamiento, optimización de experiencia de usuario y sistemas de recomendación.

# 6. Aplicaciones en Ingeniería de Sistemas

- Análisis de comportamiento de usuarios.  
- Optimización de interfaces web.  
- Sistemas de recomendación personalizados.  
- Predicción de tasa de abandono.  
- Desarrollo de dashboards de inteligencia web.

# 7. Anexos

IMPLEMENTACION  
  
  
RESULTADOS  
