NTITUTO POLITECNICO NACIONAL

ESCUELA SUPERIOR DE COMPUTO

Nombre de los alumnos: Sandoval Muñoz Emanuel, Valerio López José

Eduardo, Trejo Sierra Héctor.

Matriculas: 2022670104, 2022670002, 2021630065.

Maestr@: CAMACHO VAZQUEZ VANESSA ALEJANDRA

Materia: MACHINE LEARNING

Grupo: 6CV1

Trabajo: Práctica 7 Clasificador por el método del perceptrón simple

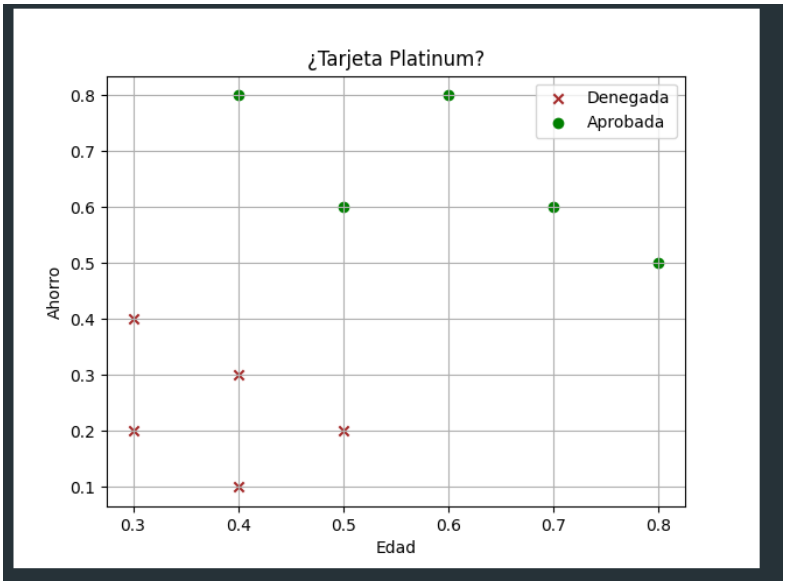
Código completo del clasificador del perceptrón con comentarios

```python
import numpy as np
import matplotlib.pyplot as plt

# Datos de 10 personas representados como una matriz de características [edad, ahorro]
personas = np.array([[0.3, 0.4], [0.4, 0.3],
                     [0.3, 0.2], [0.4, 0.1],
                     [0.5, 0.2], [0.4, 0.8],
                     [0.6, 0.8], [0.5, 0.6],
                     [0.7, 0.6], [0.8, 0.5]])

# Clases que indican si una tarjeta es aprobada (1) o denegada (0) para cada persona
clases = np.array([0, 0, 0, 0, 0, 1, 1, 1, 1, 1])

# Crear la gráfica de dispersión para visualizar los datos
for i, clase in enumerate(clases):
    if clase == 1:
        plt.scatter(personas[i, 0], personas[i, 1], color='green', marker='o',
                    label='Aprobada' if i == clases.tolist().index(1) else "")
    else:
        plt.scatter(personas[i, 0], personas[i, 1], color='brown', marker='x',
                    label='Denegada' if i == clases.tolist().index(0) else "")

# Etiquetas y título del gráfico
plt.xlabel('Edad')
plt.ylabel('Ahorro')
plt.title('¿Tarjeta Platinum?')
plt.legend()
plt.grid(True)

# Mostrar la gráfica
plt.show()
```

```python
# Función de activación para el perceptrón
4 usages  new *
def activacion(peso, x, b):
    z = peso * x
    if z.sum() + b > 0:
        return 1
    else:
        return 0


# Inicialización de pesos y bias aleatorios
pesos = np.random.uniform(-1,  high: 1, size=2)
b = np.random.uniform(-1,  high: 1)


# Prueba de la función de activación con dos puntos
print(f'Funcion de activacion [0.1,0.7]: {pesos, b, activacion(pesos,  x: [0.1, 0.7], b)}')
print(f'Funcion de activacion [0.6,0.8]: {pesos, b, activacion(pesos,  x: [0.6, 0.8], b)}')


# Entrenamiento del perceptrón
pesos = np.random.uniform(-1,  high: 1, size=2)
b = np.random.uniform(-1,  high: 1)
tasa_de_aprendizaje = 0.01
epocas = 100

print("\nEntrenamiento del perceptron del [0.5, 0.5]")
#print("\nEntrenamiento del perceptron del [0.1, 0.7]")
#print("\nEntrenamiento del perceptron del [0.6, 0.8]")
```

```python
# Iterar a través de las épocas de entrenamiento
for epoca in range(epocas):
    error_total = 0
    for i in range(len(personas)):
        # Predicción utilizando la función de activación
        prediccion = activacion(pesos, personas[i], b)
        # Calcular el error
        error = clases[i] - prediccion
        # Sumar el error cuadrático total
        error_total += error ** 2
        # Actualización de los pesos y el bias
        pesos[0] += tasa_de_aprendizaje * personas[i][0] * error
        pesos[1] += tasa_de_aprendizaje * personas[i][1] * error
        b += tasa_de_aprendizaje * error
    # Imprimir el error total por época
    print(error_total, end=" ")


# Realizar una predicción final después del entrenamiento
print(f'\nActivacion para [0.5, 0.5]: {activacion(pesos, x: [0.5, 0.5], b)}')
#print(f'Activacion para [0.5, 0.5]: {activacion(pesos, [0.1, 0.7], b)}')
#print(f'Activacion para [0.5, 0.5]: {activacion(pesos, [0.6, 0.8], b)}')
```

Ejecución:



Función de activación

```
Funcion de activacion [0.1,0.7]: (array([ 0.8805165 , -0.07609817]), -0.49601900603377813, 0)
Funcion de activacion [0.6,0.8]: (array([ 0.8805165 , -0.07609817]), -0.49601900603377813, 0)
```

Entrenamiento con 0.5, 0.5

```
Entrenamiento del perceptron del [0.5, 0.5]
5 5 5 5 4 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Activacion para [0.5, 0.5]: 1
```

Entrenamiento con 0.1, 0.7

```
Entrenamiento del perceptron del [0.1, 0.7]
5 5 5 5 5 5 4 5 4 4 4 4 5 4 5 4 4 5 4 5 4 4 4 5 4 4 5 4 4 4 5 3 2 4 3 2 2 2 3 2 2 2 2 3 2 2 2 2 3 2 2 2 2 2 3 2 2 2 2 3 2
  2 2 2 2 3 2 2 2 2 3 2 2 2 2 1 2 2 2 1 2 2 2 1 2 2 2 1 2 2 2 1 2 2 2 2 1 2 2 2
Activacion para [0.1, 0.7]: 1
```

Entrenamiento con 0.6, 0.8

```
Entrenamiento del perceptron del [0.6, 0.8]
5 5 5 4 3 3 3 2 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Activacion para [0.6, 0.8]: 1
```

¿Cómo está compuesta una neurona biológica y cómo funciona? ¿Cómo se modeló matemáticamente una neurona biológica para que pudiéramos llegar al perceptrón (señales de entrada, salida y nodos)?

Composición y funcionamiento

Una neurona biológica está compuesta por varias partes principales:

1. Cuerpo celular (soma): contiene el núcleo y otros organelos importantes. Es el centro metabólico de la neurona.
2. Dendritas: Extensiones ramificadas que reciben señales de otras neuronas.
3. Axón: Una extensión larga que transmite señales eléctricas desde el cuerpo celular hacia otras neuronas o músculos.
4. Terminales del axón: ramificaciones al final del axón que transmiten señales a otras neuronas mediante sinapsis.
5. Sinapsis: Estructuras donde se comunican dos neuronas, típicamente mediante la liberación de neurotransmisores.

Las neuronas funcionan transmitiendo señales eléctricas y químicas. El proceso se puede resumir en los siguientes pasos:

1. Recepción de señales: las dendritas reciben señales de otras neuronas.
2. Integración: El cuerpo celular integra las señales recibidas. Si la señal integrada supera un umbral, se genera un potencial de acción.
3. Transmisión: El potencial de acción viaja a lo largo del axón.
4. Comunicación: En los terminales del axón, la señal eléctrica se convierte en una señal química que se transmite a otras neuronas a través de la sinapsis.

Modelado Matemático del Perceptrón

El perceptrón es un modelo matemático simplificado de una neurona biológica. Fue introducido por Frank Rosenblatt en 1958, y se puede describir de la siguiente manera:

1. Señales de entrada: En el modelo del perceptrón, las entradas ($x_1, x_2, \ldots, x_n$) representan los estímulos que recibe la neurona.
2. Pesos: Cada entrada tiene un peso asociado ($w_1, w_2, \ldots, w_n$) que representa la importancia o influencia de esa entrada.
3. Sumador: Se calcula una suma ponderada de las entradas:

$$\sum_{i=1}^{n} w_i x_i$$

4: Función de activación: La salida ponderada se pasa a través de una función de activación, generalmente una función escalón, que determina si la neurona se activa o no.
5: Señales de salida: La salida del perceptrón (y) o la respuesta de la neurona, que puede ser 0 o 1 en el caso de la función escalón.

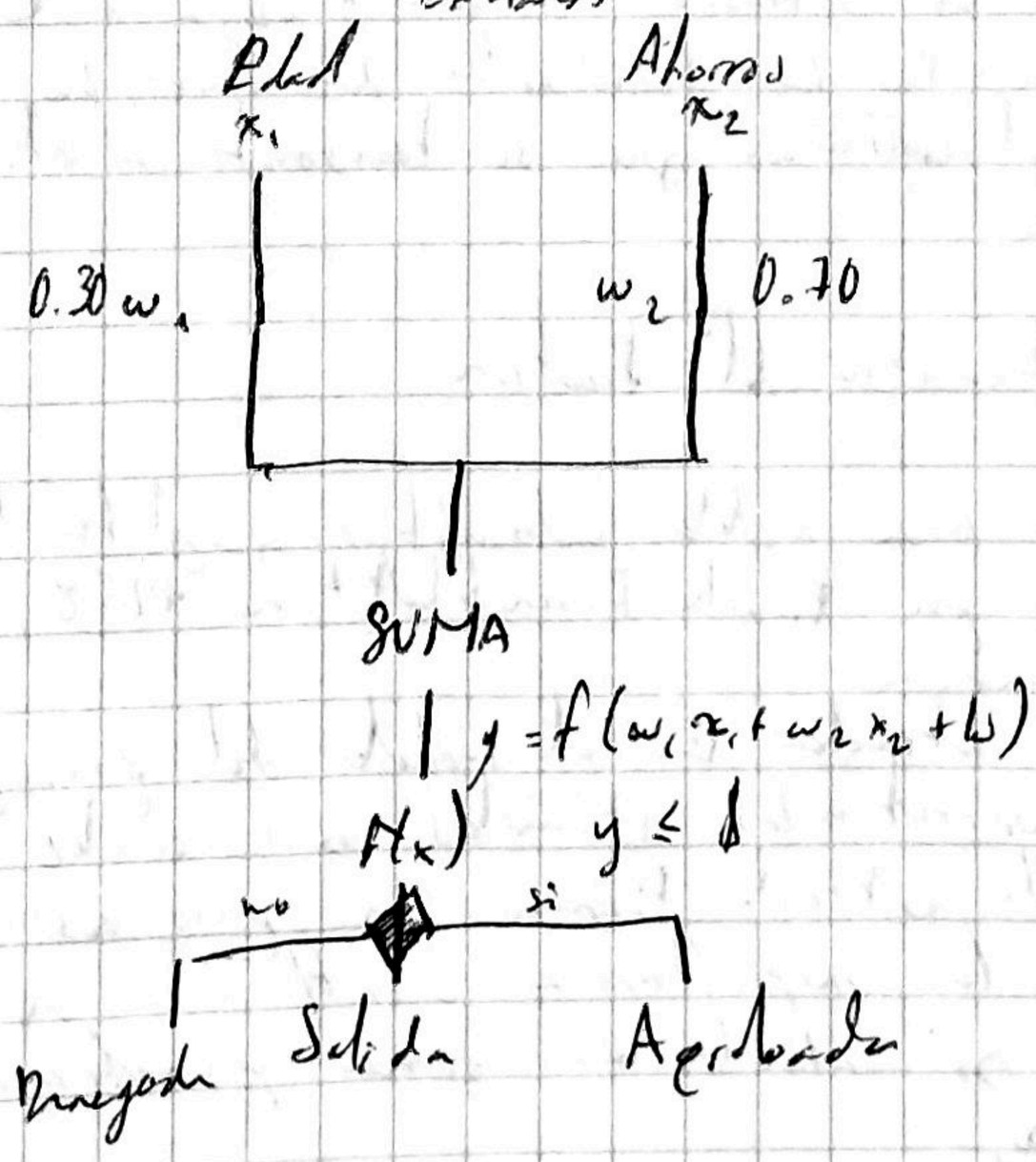
La ecuación del perceptrón puede ser representada como:

$$y = \sum_{i=1}^{n} w_i x_i + b$$

Donde b es el sesgo (bias) y f es la función de activación.
Esta es la base de las redes neuronales artificiales.

Realizar un diagrama de la estructura básica del perceptrón
*Nota: El perceptrón es un clasificador binario, es decir, solamente puede clasificar las instancias o objetos entre 2 diferentes categorías

El clasificador por el método del perceptrón simple que se construye interpreta si la dieta dependiendo su edad y ahorros la será:
- Aceptada una entrada platinum (1)
- Denegada una entrada platinum (0)



Entradas (x_1, x_2, quizás otras): Estas son las señales de entrada que representan las características del cliente (ahorros y edad).

Pesos (w_1, w_2, ...): Cada entrada tiene un peso asociado que determina su importancia en la decisión.

Suma: calcula la suma ponderada de las entradas $\sum_{i=1}^{n} w_i x_i$

Función de activación: Esta función decide si la suma ponderada es suficiente para activar la neurona. Puede ser una función escalón que devuelve 0 o 1.

Salida: El resultado del perceptrón, que 1 es aprobado y 0 es denegado

Ejemplo de Clasificador

Para este ejercicio supongamos que tenemos dos características: edad y ahorros. Para obtener su perceptrón es necesario determinar si el cliente es aprobado o deniega una tarjeta platinum.
- Edad (x_1): Edad del cliente
- Ahorros (x_2): Cantidad de ahorros del cliente.

La salida será 1 si ya aprobada la tarjeta, y 0 si se deniega. Los pesos (w_1, w_2) y el sesgo (b) se ajustan durante el entrenamiento del perceptrón.

$$y = f(w_1 x_1 + w_2 x_2 + b)$$

Donde f es la función de activación, típicamente una función escalón definida por:

$$y \begin{cases} 1 & \text{si } x \geq 0 \\ 0 & \text{si } x < 0 \end{cases}$$

¿El perceptrón es un modelo de aprendizaje automático supervisado o no supervisado? Explica por qué.

El perceptrón es un modelo de aprendizaje supervisado.

Aprendizaje Supervisado

En el aprendizaje supervisado, el modelo se entrena utilizando un conjunto de datos etiquetados, es decir, cada ejemplo de entrenamiento viene acompañado de la respuesta correcta. El objetivo es que el modelo aprenda a mapear las entradas a las salidas correctas utilizando esta información.

Características del perceptrón

1- Datos Etiquetados: Durante el entrenamiento, el perceptrón proporciona ejemplos de entrada junto con sus correspondientes etiquetas de salida (también conocidas como clases). Por ejemplo, en el caso del clasificador de tarjetas platinum, las entradas pueden ser características como la edad y los ahorros del cliente, y las etiquetas son 0 (rechazadas) y 1 (aceptadas).

2- Función de pérdida: El perceptrón utiliza una función de pérdida para evaluar qué tan bien está haciendo las predicciones durante el entrenamiento. La función de pérdida calcula la diferencia entre la salida predicha por el modelo y la etiqueta real proporcionada en los datos de entrenamiento.

3- Actualización de pesos: Con base en la función de pérdida, el perceptrón ajusta los pesos de las conexiones de entrada para reducir el error en futuras predicciones. Este proceso se repite iterativamente hasta que el modelo converge o alcanza un número máximo de iteraciones.

4- Generalización: Una vez entrenado, el perceptrón puede generalizar a datos no vistos, es decir, puede predecir la etiqueta de nuevas ejemplos de entrada basándose en el aprendizaje obtenido durante el entrenamiento.

Parte más interesante:
La parte más interesante podría ser la programación del perceptrón desde cero y el proceso de ajuste iterativo de los pesos. Esto es fascinante porque permite comprender y visualizar cómo funciona el aprendizaje y cómo aprende a clasificar datos mediante la retroalimentación de errores. Programar el perceptrón desde una perspectiva práctica y tangible del aprendizaje automático, lo que puede ser muy gratificante para los estudiantes.

Parte más difícil
La parte más difícil podría ser la comprensión e implementación del ajuste iterativo de los pesos y el sesgo. Esta sección es compleja porque implica un entendimiento detallado de cómo ajustar los pesos en función del error de predicción, lo que requiere una buena base en matemáticas y lógica de programación. Además, encontrar la tasa de aprendizaje adecuada y el número óptimo de épocas puede ser un desafío, ya que requiere experimentación y ajustes finos para lograr un modelo bien entrenado.