1. Generación del Conjunto de Datos

scilab

```
nct=20; // Tamaño del conjunto de datos
x=2*rand(2,nct)-1; // Genera puntos aleatorios entre [-1, 1] en x e y
x1=x(1,:); // Coordenadas x
y1=x(2,:); // Coordenadas y
plot(x1,y1,'*'); // Grafica los puntos
```

Análisis del Código del Perceptrón en SCILAB

Análisis del Código del Perceptrón en SCILAB

Este código implementa un perceptrón simple para clasificar puntos en un plano 2D, generando datos aleatorios y aprendiendo a separarlos según una recta de referencia.

Definición de la Recta de Referencia

- Define la recta y = 2x como frontera de decisión verdadera
- Grafica esta recta en color rojo para visualización

```scilab
F=[1;-2]; // Coeficientes de la recta y-2x=0 (equivale a y=2x)
w=[0;0]; // Pesos iniciales del perceptrón (0,0)

// Grafica la recta de referencia (roja)
x2=linspace(-1,1,100);
for i=1:100
    y2(i)=2*x2(i);
end
plot(x2,y2,'r')
```

Clasificación de los Puntos

- Clasifica los puntos según su posición respecto a la recta y=2x
- Puntos arriba de la recta (verde): clase 1
- Puntos abajo de la recta (azul): clase -1

```
for i=1:nct
    l(i)=-F(2)*x1(i)-y1(i); // Calcula posición relativa respecto a la recta
    class_F(i)=sign(l(i));  // Clasificación: 1 (arriba de la recta), -1 (abajo)
end

// Grafica puntos según su clase
for i=1:nct
    if class_F(i)==1 then
        plot(x1(i),y1(i),'gre*'); // Verde para clase 1
    else
        plot(x1(i),y1(i),'blu*'); // Azul para clase -1
    end
end
```

```
// Inicialización
for i=1:nct
    g(i)=sign(w(2)*y1(i)+w(1)*x1(i)); // Predicción inicial con pesos [0,0]
end

ind = find(g ~= class_F); // Encuentra predicciones incorrectas
iter=1;

// Ciclo de entrenamiento
while ~isempty(ind)
    // Actualiza pesos usando el primer punto mal clasificado
    w(1) = w(1) + (class_F(ind(1))-g(ind(1)))*x1(ind(1));
    w(2) = w(2) + (class_F(ind(1))-g(ind(1)))*y1(ind(1));

    // Recalcula predicciones
    for i=1:nct
        temp(i)=sign(w(2)*y1(i)+w(1)*x1(i));
    end

    ind = find(temp ~= class_F); // Encuentra nuevos errores
    iter = iter + 1;
    g=temp;
end
```

Algoritmo del Perceptrón

- Inicia con pesos [0,0] (no tiene capacidad de clasificación)
- En cada iteración:
- Identifica puntos mal clasificados
- Ajusta pesos usando el primer punto mal clasificado
- Recalcula todas las predicciones
- Termina cuando todos los puntos están bien clasificados

Visualización del Resultado

scilab

```
// Grafica la recta aprendida (verde)
for i=1:100
    y3(i)=(-w(1)*x2(i))/w(2);
end
plot(x2,y3,'g');

// Muestra pesos finales
mfprintf(6,'- - Al final los pesos son:\nvector w
disp(w);
```

- Grafica la recta definida por los pesos aprendidos (en verde)
- Muestra los pesos finales en la consola

Flujo General del Programa

- Genera datos aleatorios en 2D
- Define una recta de separación verdadera (y=2x)
- Clasifica los puntos según su posición respecto a la recta verdadera
- Entrena un perceptrón para aprender esta clasificación
- Muestra la recta aprendida por el perceptrón

Limitaciones

- Solo funciona para problemas linealmente separables
- El conjunto de datos es pequeño (20 puntos)
- Usa una sola actualización por iteración (el primer punto mal clasificado)
- Este código es un excelente ejemplo didáctico del perceptrón, mostrando claramente cómo aprende una frontera de decisión lineal a partir de ejemplos.