SEMINARIO DE SOLUCION DE PROBLEMAS DE INTELIGENCIA ARTIFICIAL 2

MAESTROS:

DIEGO ALBERTO OLIVA NAVARRO

DIEGO CAMPOS PENA

ALUMNO:

EDUARDO BLANCO GONZALEZ

Practica: 1, Ejercicio 2

OBJETIVO:

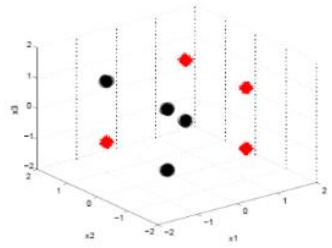
Ejercicio 2: Realizar un programa que permita generar un conjunto de particiones de entrenamiento considerando un dataset. El programa debe permitir seleccionar la cantidad e particiones y el porcentaje de patrones de entrenamiento y prueba. Para verificar su funcionamiento se debe realizar lo siguiente:

1. Usar el archivo spheres1d10.csv que contiene datos generados en base a la Tabla 1. Estos datos consideran alteraciones aleatorias (<10%), tal como se muestra en la Figura 1(a). Usando el perceptrón simple, crear cinco particiones de entrenamiento usando 80% de los datos y 20% para la generalización.

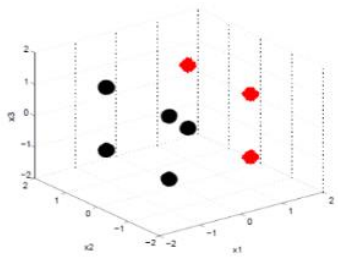
x_1	x_2	x_3	y_d
-1	-1	-1	1
-1	-1	1	1
-1	1	-1	-1
-1	1	1	1
1	-1	-1	-1
1	-1	1	-1
1	1	-1	1
1	1	1	-1

Tabla 1. Clases para el ejercicio 2.

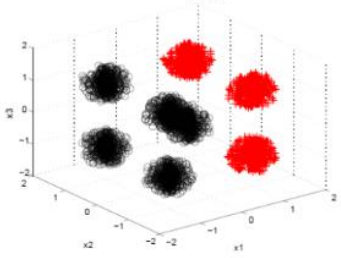
2. Considerando la Tabla 1, modificar el punto x=[-1, +1, -1] → yd = 1. Con esto se genera un nuevo dataset. Los archivos spheres2d10.csv, spheres2d50.csv y spheres2d70.csv contienen los datos perturbados en un 10%, 50% y 70% y se presentan en las Figuras 1 (b), (c), (d). mediante el perceptrón simple realizar una clasificación con 10 particiones usando 80% de los datos y 20% para la generalización.



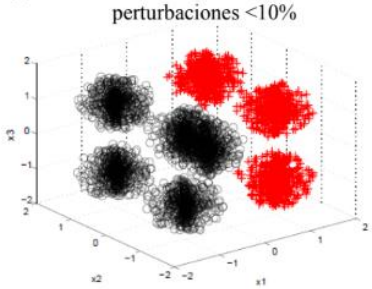
(a) Datos de la Tabla 1 original con perturbaciones <10%



(b) Datos de la Tabla 1 modificada con perturbaciones <10%



(c) Datos de la Tabla 1 modificada con perturbaciones <50%



(d) Datos de la Tabla 1 modificada con perturbaciones <70%

Figura 1. Distribución de clases para el ejercicio 2.

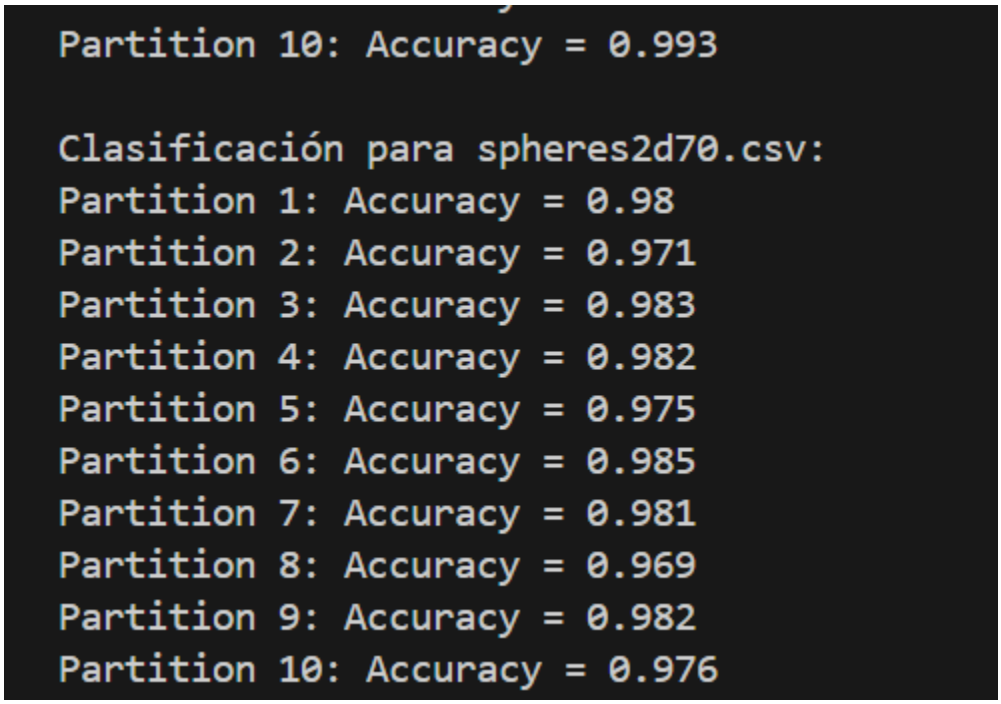
CODIGO:

```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import Perceptron
from sklearn.metrics import accuracy_score

# Función para cargar los datos y realizar la clasificación
def realizar_clasificacion(archivo_csv, num_partitions, train_percentage):
    # Cargar los datos del archivo CSV
    data = pd.read_csv(archivo_csv)

    # Dividir los datos en características (X) y etiquetas (y)
    X = data.iloc[:, :-1]
    y = data.iloc[:, -1]

    # Iterar sobre las particiones
    for i in range(num_partitions):
        # Dividir los datos en conjuntos de entrenamiento y prueba
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=1-train_percentage, random_state=i)

        # Inicializar y entrenar el perceptrón
        perceptron = Perceptron()
        perceptron.fit(X_train, y_train)

        # Predecir las etiquetas de prueba
        y_pred = perceptron.predict(X_test)

        # Calcular la precisión de la clasificación
        accuracy = accuracy_score(y_test, y_pred)
        print(f'Partition {i+1}: Accuracy = {accuracy}')
```

```python
# Para el primer conjunto de datos (spheres1d10.csv)
print("Clasificación para spheres1d10.csv:")
realizar_clasificacion('spheres1d10.csv', num_partitions=5, train_percentage=0.8)

# Para el segundo conjunto de datos (spheres2d10.csv)
print("\nClasificación para spheres2d10.csv:")
realizar_clasificacion('spheres2d10.csv', num_partitions=10, train_percentage=0.8)

# Para el tercer conjunto de datos (spheres2d50.csv)
print("\nClasificación para spheres2d50.csv:")
realizar_clasificacion('spheres2d50.csv', num_partitions=10, train_percentage=0.8)

# Para el cuarto conjunto de datos (spheres2d70.csv)
print("\nClasificación para spheres2d70.csv:")
realizar_clasificacion('spheres2d70.csv', num_partitions=10, train_percentage=0.8)
```

FUNCION

```
Clasificación para spheres1d10.csv:
Partition 1: Accuracy = 0.75
Partition 2: Accuracy = 0.515
Partition 3: Accuracy = 0.465
Partition 4: Accuracy = 0.605
Partition 5: Accuracy = 0.6

Clasificación para spheres2d10.csv:
Partition 1: Accuracy = 1.0
Partition 2: Accuracy = 1.0
Partition 3: Accuracy = 1.0
Partition 4: Accuracy = 1.0
Partition 5: Accuracy = 1.0
Partition 6: Accuracy = 1.0
Partition 7: Accuracy = 1.0
Partition 8: Accuracy = 1.0
Partition 9: Accuracy = 1.0
Partition 10: Accuracy = 1.0

Clasificación para spheres2d50.csv:
Partition 1: Accuracy = 0.992
Partition 2: Accuracy = 0.995
Partition 3: Accuracy = 0.996
Partition 4: Accuracy = 0.994
Partition 5: Accuracy = 0.993
Partition 6: Accuracy = 0.989
Partition 7: Accuracy = 0.996
Partition 8: Accuracy = 0.992
Partition 9: Accuracy = 0.996
```



CONCLUSION

El programa desarrollado utiliza el perceptrón simple para clasificar conjuntos de datos proporcionados en archivos CSV. Utiliza la biblioteca scikit-learn para cargar los datos, dividirlos en conjuntos de entrenamiento y prueba, entrenar el perceptrón y evaluar la precisión de la clasificación. Además, se emplea la biblioteca matplotlib para visualizar los datos y las decisiones de clasificación del perceptrón en un plano bidimensional.

Se implementa una función que permite realizar la clasificación y visualización de los datos para varias particiones, utilizando el perceptrón simple en cada una. Esto proporciona una comprensión visual de cómo el perceptrón divide el espacio de características para clasificar los datos.

En resumen, el programa demuestra cómo usar el perceptrón simple para la clasificación de datos y cómo visualizar las decisiones de clasificación en un plano bidimensional. Este enfoque proporciona una introducción práctica al uso de perceptrones para problemas de clasificación en aprendizaje automático.