**K-VECINOS MÁS PRÓXIMOS**

En una base de datos donde a los pacientes se les midió su nivel de depresión se quiere saber si van a volver a recaer en base a una serie de variables: “recuperación” (pacientes que se han recuperado y pacientes que no se han recuperado), “tratamiento” (pacientes que reciben tratamiento estándar y tratamiento combinado), “régimen” (1=interno, 2=externo, 3=domiciliario), “edad” (años que tienen), “años” (años que llevan en tratamiento), etc.

La variable dependiente (recaída) es categórica, el valor 1 indica que recaen el primer año de su recuperación, el valor 2 que recaen en el segundo año de su recuperación y el valor 3 que no recaen.

import pandas as pd

import matplotlib.pyplot as plt

import numpy as np

depresion = pd.read\_csv("C:/Users/alici/Desktop/Master/Aprendizaje Automático/Python/r\_logistica.csv", sep=";")

#comprobamos que se ha cargado viendo los primeros registros

print(depresion.head())

#quitamos el id y el sexo (no interesan), edad\_c, años\_c, años\_cat (redundantes)

depresion = depresion.drop(["id","sexo","edad\_c","años\_c","años\_cat"], axis=1)

#los vectores en el espacio serán solo las variables

#independientes. Será la matriz X. Por eso se quita la variable criterio.

X = depresion.drop("recaída", axis=1)

X = X.values

#También en el vector y se colocan los valores de la variable a predecir (recaidas)

y = depresion["recaída"]

y = y.values

print(X)

print(y)

print(depresion.head())

Una vez tratados los datos, queremos predecir si un nuevo paciente va a recaer en función del resto de los datos. El nuevo paciente tiene las siguientes características: se encuentra en el régimen 2 (externo), de 21 años, que lleva 4 años consumiendo, que recibe el tratamiento 1 (combinado).

#Se incluye un nuevo ejemplar del que se ha de estimar la recaída

new\_data\_point = np.array([2,21,4,1,1])

#se calculan las distancias entre el nuevo y todos los vectores de la matriz

#esto se hace calculando la norma a dichas restas

distances = np.linalg.norm(X-new\_data\_point, axis=1)  #distancia euclidea. La norma

print(distances)

#calculadas las distancias, se ha de coger un vecindario de 3

#para ello se ordenan las distancias y se toman los índices de las distancias más cortas

k = 3

nearest\_neighbor\_ids = distances.argsort()[:k]

print(nearest\_neighbor\_ids)

#ahora tomamos los valores en y de esos índices. y contiene los valores en "rings"

nearest\_neighbor\_rings = y[nearest\_neighbor\_ids]

print(nearest\_neighbor\_rings)

#la predicción será el promedio de los valores de "rings" de esos vecinos.

prediction = nearest\_neighbor\_rings.mean()

print(prediction)

Indicando k=3, el modelo ha predicho que nuestro sujeto que se encuentra en el régimen 2 (externo), de 21 años, que lleva 4 años consumiendo, que recibe el tratamiento 1 (combinado) y que se ha recuperado no va a recaer.

Pero este es el resultado obtenido para k=3, ¿qué predicciones hará el modelo si variamos el valor de k? ¿predecirá igual si el paciente recaerá?

from scipy.stats import mode

distances = np.linalg.norm(X-new\_data\_point, axis=1)  #distancia euclidea. La norma

print(distances)

results = []

for k in range(1, 11):

    nearest\_neighbor\_ids = distances.argsort()[:k]

    nearest\_neighbor\_rings = y[nearest\_neighbor\_ids]

    prediction = mode(nearest\_neighbor\_rings).mode[0]  # Usamos la moda para la predicción categórica

    # Agregar los resultados al listado

    results.append({

        'k': k,

        'nearest\_neighbor\_ids': nearest\_neighbor\_ids,

        'nearest\_neighbor\_rings': nearest\_neighbor\_rings,

        'prediction': prediction

    })

# Convertir el listado de resultados en un DataFrame

results\_df = pd.DataFrame(results)

# Mostrar el DataFrame

print(results\_df)

Salida:

0 3

1 3

2 3

3 3

4 3

5 3

6 3

7 3

8 3

9 3

Vemos que, con distintos valores de k, se sigue prediciendo que el nuevo sujeto no va a recaer. Esto indica que las características del sujeto son todas favorables a la recuperación, va a recuperarse sí o sí, y por eso, nuestro modelo de KNN no encuentra divergencias variando el valor de k.