import pandas as pd

import sklearn

import matplotlib.pyplot as plt

from sklearn.decomposition import KernelPCA

from sklearn.linear\_model import LogisticRegression

from sklearn.preprocessing import StandardScaler

from sklearn.model\_selection import train\_test\_split

#from sklearn.metrics import jaccard\_score

dt\_heart = pd.read\_csv('heart.csv')

X = dt\_heart.drop('target', axis =1)

Y = dt\_heart['target']

X = StandardScaler().fit\_transform(X)

X\_train,X\_test, y\_train, y\_test = train\_test\_split(X,Y,test\_size=0.3,random\_state=42)

# Default si no asigna n\_components = min(n\_columnas, n\_filas)

# Entrenaremos con 3 componentes de los 13 que tiene

# Que PCA evalue cuales son los mas relevantes

n= 3

#tipo de kernel

#normal : linear

#polinomio : poly

#gausiano : rbf

kpca = KernelPCA(n\_components=n, kernel='poly')

#ajustar datos de entrenamiento

kpca.fit(X\_train)

# aplicar PCA a los datos de entrenamiento y prueba

dt\_train = kpca.transform(X\_train)

dt\_test = kpca.transform(X\_test)

#regresion logistica

logistic = LogisticRegression(solver='lbfgs')

logistic.fit(dt\_train, y\_train)

print("Score PCA: ", logistic.score(dt\_test, y\_test))

**Comparación entre los tres tipos de kernel con este ejercicio en particular**

Graphical user interface, text

Description automatically generated with medium confidence