**Practica 7: Detección de spam**

Alberto Muñoz Fernández

Óscar García Castro

Parte A: Support Vector Machines

import numpy as np

import matplotlib.pyplot as plt

import sklearn.svm

import scipy.io as sc

def get\_data(path):

    data = sc.loadmat(path, squeeze\_me=True)

    X = data['X']

    y = data['y']

    return X, y

def draw(X, y, x1, x2, yp):

    plt.figure()

    positive = y == 1

    negative = y == 0

    plt.plot(X[positive, 0], X[positive, 1], 'k+')

    plt.plot(X[negative, 0], X[negative, 1], 'yo')

    plt.contour(x1, x2, yp)

    plt.show()

def kernel\_lineal(X, y):

    svm = sklearn.svm.SVC(kernel='linear', C=1.0)

    svm.fit(X, y)

    x1 = np.linspace(X[:, 0].min(), X[:, 0].max(), 100)

    x2 = np.linspace(X[:, 1].min(), X[:, 1].max(), 100)

    x1, x2 = np.meshgrid(x1, x2)

    yp = svm.predict(np.array([x1.ravel(),

x2.ravel()]).T).reshape(x1.shape)

    draw(X, y, x1, x2, yp)

def kernell\_gausiano(X, y, c = 1, sigma = 0.1):

    svm = sklearn.svm.SVC(kernel='rbf', C = c, gamma=1 / (2 \* sigma\*\*2))

    svm.fit(X, y)

    x1 = np.linspace(X[:, 0].min(), X[:, 0].max(), 100)

    x2 = np.linspace(X[:, 1].min(), X[:, 1].max(), 100)

    x1, x2 = np.meshgrid(x1, x2)

    #da error aqui (X has 2 features, but SVC is expecting 1899 features as input)

    yp = svm.predict(np.array([x1.ravel(), x2.ravel()]).T).reshape(x1.shape)

    draw(X, y, x1, x2, yp)

    return svm

def elec\_params(X, y, Xval, yval, c, sigma):

    svm = sklearn.svm.SVC(kernel='rbf', C = c, gamma=1 / (2 \* sigma\*\*2))

    svm.fit(X, y)

    yp = svm.predict(Xval)

    cont = 0

    for i in range (len(yval)):

        if (yval[i] == yp[i]):

            cont += 1

    return cont/len(yval) \* 100

def bestCandSigma():

    data = sc.loadmat('data/ex6data3.mat', squeeze\_me=True)

    X = data['X']

    y = data['y']

    Xval = data['Xval']

    yval = data['yval']

    vals = [0.01, 0.03, 0.1, 0.3, 1, 3, 10, 30]

    acierto = -1

    bestC = -1

    bestSigma = -1

    for c in vals:

        for sig in vals:

            aux = elec\_params(X, y, Xval, yval, c, sig)

            if (acierto == -1 or aux > acierto):

                acierto = aux

                bestSigma = sig

                bestC = c

    kernell\_gausiano(X, y, bestC, bestSigma)

def calcSVM(X, y, Xval, yval):

    vals = [0.01, 0.03, 0.1, 0.3, 1, 3, 10, 30]

    acierto = -1

    bestC = -1

    bestSigma = -1

    for c in vals:

        for sig in vals:

            aux = elec\_params(X, y, Xval, yval, c, sig)

            if (acierto == -1 or aux > acierto):

                acierto = aux

                bestSigma = sig

                bestC = c

    svm = sklearn.svm.SVC(kernel='rbf', C = bestC, gamma=1 / (2 \*

bestSigma\*\*2))

    svm.fit(X, y)

    return svm

Parte B: Detección de spam