PRACTICA 1

VECTORES SOPORTE

Ángel Igualada Moraga

2018-2019

EJERCICIO 3

En este ejercicio, calculamos los vectores soporte, los multiplicadores de Lagrange, la frontera de decisión y el margen para las muestras de dos conjuntos de entrenamiento en los que había dos clases en cada uno, siendo un conjunto linealmente separable y el otro no.

En la implementación, usamos el script esquemaMini para llamar a SVM, en él hemos utilizado en primer lugar el método svmtrain con los argumentos "-t 0 -c *valor*" para seleccionar el tipo de kernel y el valor de C.

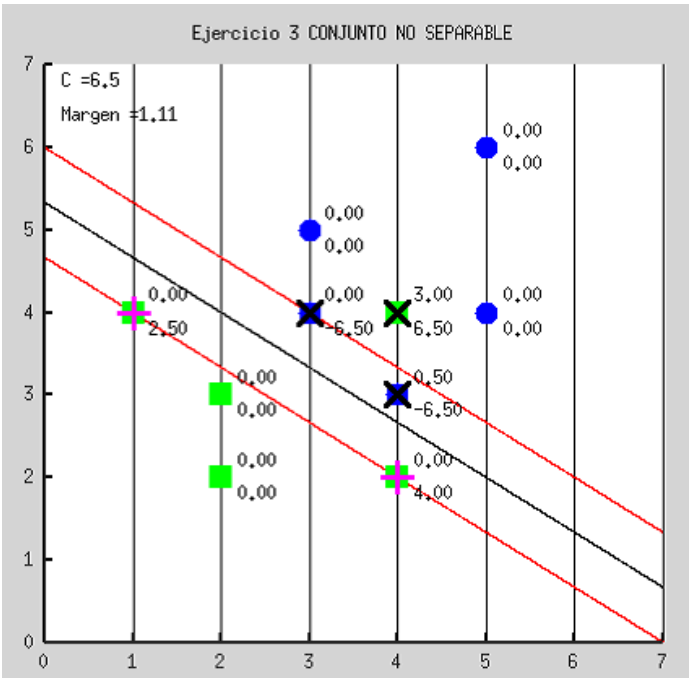
Esta función nos proporciona los Alphas (atributo sv_coef), los índices en los que en el conjunto de datos están los vectores soporte y el w0.

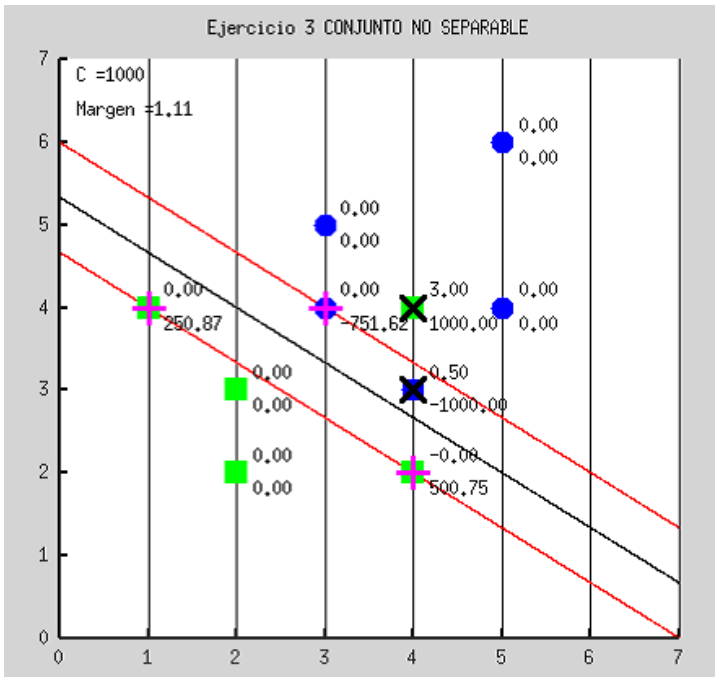
Con los índices de los vectores soporte, hemos obtenido los vectores soporte y con estos y los multiplicadores de Lagrange obtuvimos el vector de pesos. Teniendo el vector de pesos, dividimos 2 entre su modulo y obtuvimos el margen.

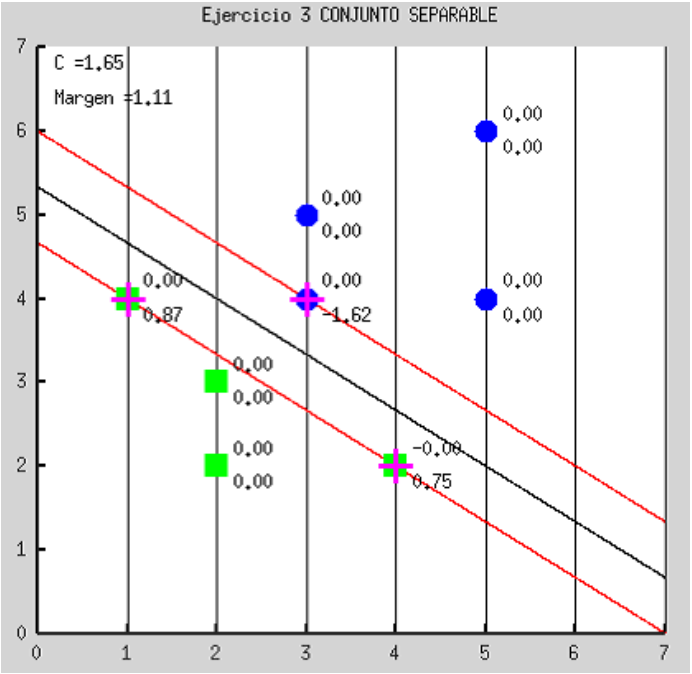
Por último obtuvimos la pendiente de la frontera y su ecuación. Con esta misma pendiente obtuvimos las rectas de los márgenes.

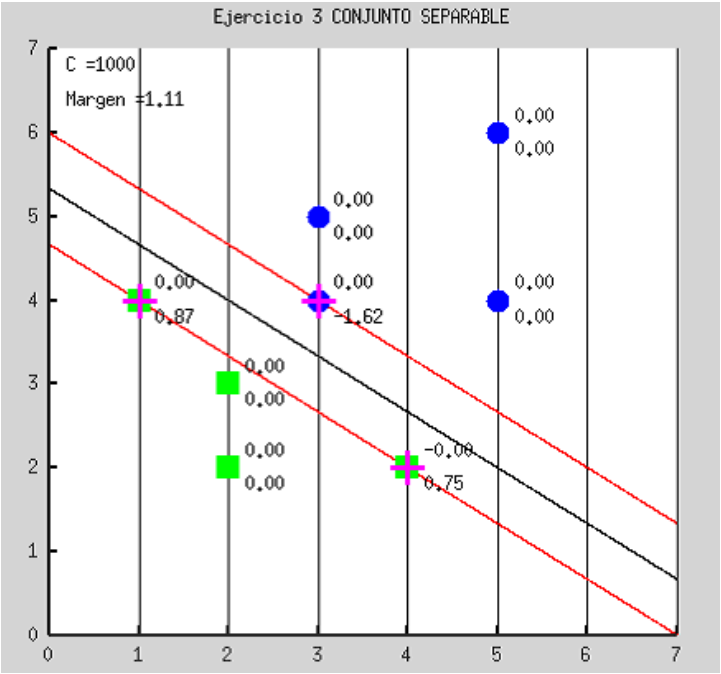
Resultados

En el caso separable observamos que a partir de C=1.65, el margen llega a su valor mínimo (1.109) mientras que en el caso no separable, necesita un C=6,5 para poder obtener su margen mínimo (1.109). Como era de esperar, en el caso no separable tenemos vectores soporte malos llegado el margen mínimo.

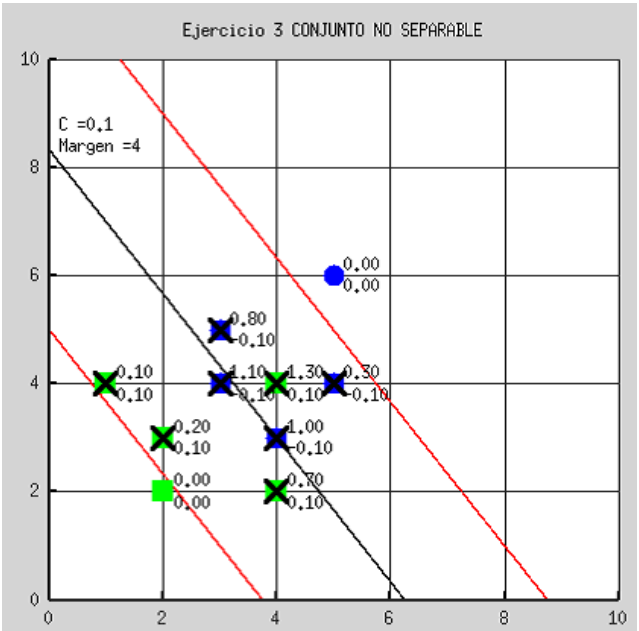




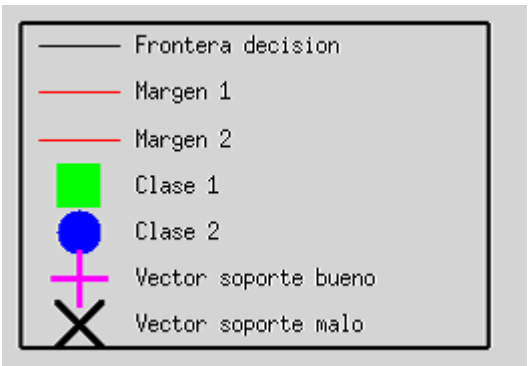




Para valores de C muy pequeños el margen es grande y tenemos muchos vectores soporte malos:



LEYENDA



EJERCICIO 4

En este ejercicio se ha realizado un script que itera por distintos valores de C en los kenels 0 (lineal),1(polinomial (con grado 1,2,3 y 4)),2(radial) y 3(sigmoide).

Cabe destacar que para varias ejecuciones se alcanza el numero máximo de iteraciones establecidas por la librería, en algunas de estas, se obtiene un 100% de acierto por lo que no es segura su correctitud.

Se adjuntará en un anexo una tabla con las ejecuciones que alcanzan el máximo de iteraciones.

Sin tener en cuenta las ejecuciones que alcanzan el máximo de iteraciones, los mejores resultados, se obtienen con el kernel polinomial con grado 2, 3 y 4 en las que se llega hasta a un 99,78 % de exactitud con valores muy variados de C con un intervalo de confianza de (99,50 – 99,99), siendo el kernel sigmoide el peor.

En la siguiente tabla se muestran los mejores resultados, remarcando en amarillo aquellos que se obtuvieron en una ejecución que alcanzó el máximo número de iteraciones.

C	Kernel	grado	exactitud	Intervalo confianza	tiempo_ejecución (segs)
0,01	0-lineal	1	1	±0	53,8732
1	1-polinomial	1	1	±0	65,6894
10	1-polinomial	1	1	±0	27,7900
0,1	0-lineal	1	0,9993	±0,0014	44,3995
100	1-polinomial	1	0,9993	±0,0014	15,9357
1000	1-polinomial	1	0,9993	±0,0014	16,4489
10	1-polinomial	2	0,9978	±0,0025	2,1579
100	1-polinomial	2	0,9978	±0,0025	0,3154
100	1-polinomial	3	0,9978	±0,0025	7,3737
1000	1-polinomial	2	0,9978	±0,0025	0,3089
1000	1-polinomial	3	0,9978	±0,0025	7,3062
1000	1-polinomial	4	0,9978	±0,0025	1,9796

C	0-lineal	1-polinomial grado-1	1-polinomial grado-2	1-polinomial grado-3	1-polinomial grado-4	2-radial	3-sigmoide
0,01	**1**	0,7951	0,9826	0,9877	0,987	0,6061	0,5445
0,1	0,9993	0,992	0,992	0,9906	0,4881	0,7371	0,4301
1	0,9971	**1**	0,9957	0,992	0,9913	0,8675	0,3353
10	0,9971	**1**	**0,9978**	0,9964	0,9949	**0,908**	0,3172
100	0,9971	0,9993	**0,9978**	**0,9978**	0,9957	0,9073	0,6792
1000	0,9971	0,9993	**0,9978**	**0,9978**	**0,9978**	0,9073	**0,714**

EJERCICIO 5

En este ejercicio se ha partido del script del ejercicio anterior que itera por distintos valores de C en los kenels 0 (lineal),1(polinomial (con grado 1,2,3 y 4)),2(radial) y 3(sigmoide).

En este caso, no tenemos ninguna ejecución que alcance el máximo número de iteraciones.

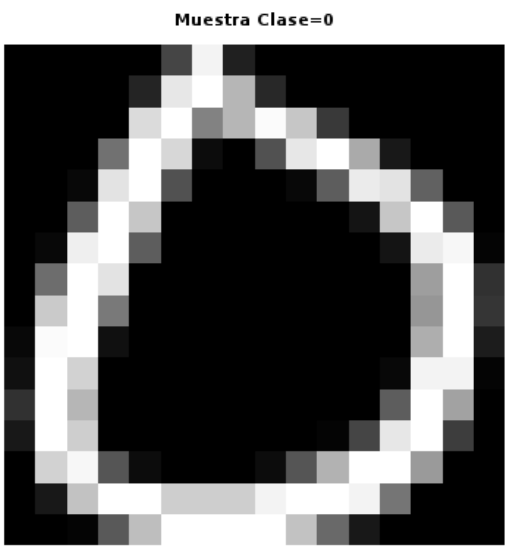
Los mejores valores, esta vez son obtenidos con un kernel radial con el que llegamos a conseguir un 95,02% de acierto con un intervalo de confianza de 95,02 ± 0,95 %.

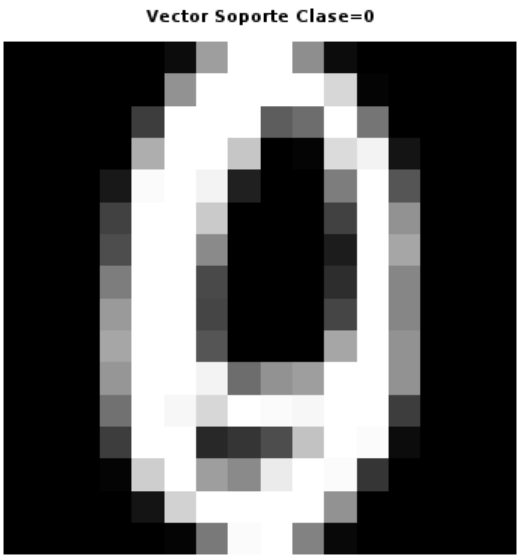
De nuevo, es posible conseguir los mejores resultados con un C bajo (2-3) hasta C's muy altos.

C	Kernel	grado	exactitud	intervalo	tiempo_ejecucion
10	2-radial	1	0,9502	±0,0095	8,1747
100	2-radial	1	0,9492	±0,0096	9,3479
1000	2-radial	1	0,9492	±0,0096	8,4036
10	1-polinomial	2	0,9457	±0,0099	7,8178
1000	1-polinomial	2	0,9447	±0,01	7,3825
100	1-polinomial	2	0,9437	±0,0101	7,4363
1	2-radial	1	0,9422	±0,0102	9,7255
100	1-polinomial	3	0,9412	±0,0103	7,8763
1000	1-polinomial	3	0,9407	±0,0103	6,4674

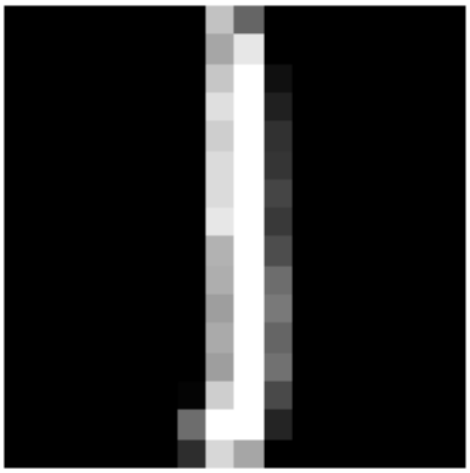
Partiendo de estos datos, se ha elegido el kernel a analizar, en este caso el radial y se ha realizado un script que calcula los vectores soportes para el kernel radial con C=10 y muestra una muestra y un vector soporte de cada clase.

Estos son algunos de los resultados obtenidos:

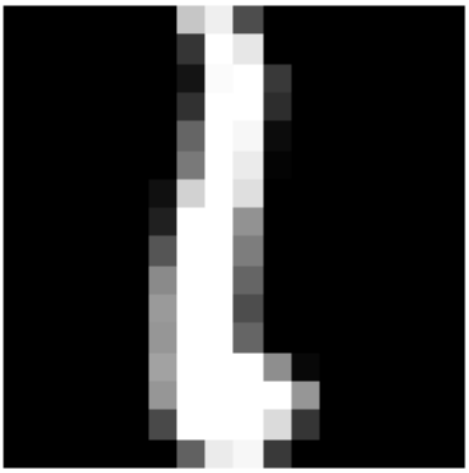




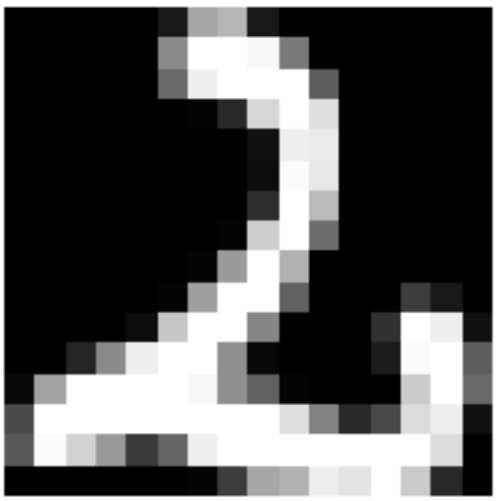
Muestra Clase=1



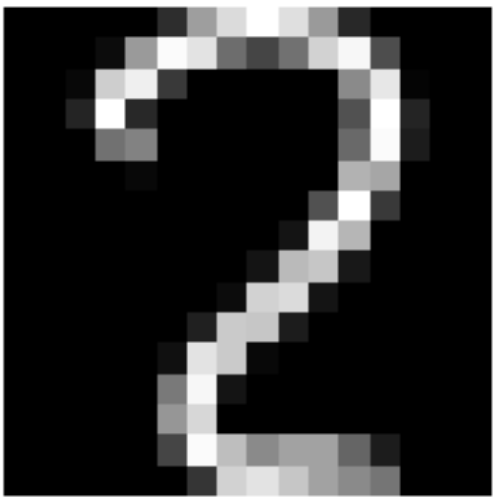
Vector Soporte Clase=1



Muestra Clase=2



Vector Soporte Clase=2



Muestra Clase=3

Vector Soporte Clase=3