15/15

6.825

Techniques in Artificial Intelligence

2005/12/08

Dan Ports

drkp@mit.edu

Project 3 Exercise 3

Using SVMlight and a test wrapper script[1], we evaluated different kernels and classification error penalties on two different test sets. Specifically, we considered linear kernels, polynomial kernels with degrees 2, 3, and 10, and RBF kernels with $\sigma = 1/2$, $1/10$, and $1/50$.

Table 1 shows the results for the first example training/test set, and Table 2 shows the results for the second example training/test set.

+1 +1 +2 +2 In general, increasing C causes the number of support vectors used to decrease. Increasing the degree of the polynomial kernel or decreasing the value of σ in the RDF kernel increases the number of support vectors, presumably causing a danger of overfitting. Accordingly, increasing C generally caused the test accuracy to increase in Example 1; in Example 2, the test accuracy was usually 100% regardless of C.

+2 For the first example, the classification boundary is fairly simple, so all choices of kernel and C provide high accuracy (97-99%). The best results are obtained from polynomial kernels; a degree-3 kernel provides 98.90% accuracy with $C = 0.1$, and 98.87% accuracy with $C = 1.0$ or $C = 20$, and a degree-2 kernel provides 98.88% accuracy with $C = 20$. All provide essentially the same accuracy; though the degree-3, $C = 0.1$ case provides slightly more accuracy than the others, the others might also be considered a bit more preferable because they use less support vectors (3 rather than 6).

+2 For the second example, all kernels except for the linear kernel provide 100% accuracy and no training misclassifications, and the value of C does not have much of an effect. It isn't surprising that the linear kernel performs poorly, since there's no way to fit a straight line to the circular data set. Of the others, since the accuracy is perfect for each, we would presumably want to choose the simplest model, i.e. the one with the least support vectors. This is the degree-2 polynomial kernel, with any of the values of C.

[1] Python source code is available at `http://www.ambulatoryclam.net/svn/classes/6.825/proj3exercises/ex3`

Table 1: Training/Test Set 1

+2.5

Kernel	**C**	**# SVs**	**# Training Misclass.**	**% Test Accuracy**
linear	0.1	100	3	97.49
linear	1	45	0	97.85
linear	20	15	0	98.63
poly (2)	0.1	22	0	97.90
poly (2)	1	11	0	98.65
poly (2)	20	3	0	98.88
poly (3)	0.1	6	0	98.90
poly (3)	1	3	0	98.87
poly (3)	20	3	0	98.87
poly (10)	0.1	3	0	98.75
poly (10)	1	3	0	98.75
poly (10)	20	3	0	98.75
RB ($\sigma = 1/2$)	0.1	89	0	98.00
RB ($\sigma = 1/2$)	1	32	0	98.05
RB ($\sigma = 1/2$)	20	9	0	98.66
RB ($\sigma = 1/10$)	0.1	125	0	98.32
RB ($\sigma = 1/10$)	1	45	0	98.62
RB ($\sigma = 1/10$)	20	28	0	98.70
RB ($\sigma = 1/50$)	0.1	200	1	98.26
RB ($\sigma = 1/50$)	1	120	0	98.66
RB ($\sigma = 1/50$)	20	111	0	98.85

Table 2: Training/Test Set 2

Kernel	C	# SVs	# Training Misclass.	% Test Accuracy
linear	0.1	38	11	68.23
linear	1	38	11	68.23
linear	20	38	11	68.23
poly (2)	0.1	13	0	100.00
poly (2)	1	13	0	100.00
poly (2)	20	13	0	100.00
poly (3)	0.1	22	0	100.00
poly (3)	1	22	0	100.00
poly (3)	20	22	0	100.00
poly (10)	0.1	27	0	100.00
poly (10)	1	27	0	100.00
poly (10)	20	27	0	100.00
RB ($\sigma = 1/2$)	0.1	38	0	100.00
RB ($\sigma = 1/2$)	1	25	0	100.00
RB ($\sigma = 1/2$)	20	26	0	100.00
RB ($\sigma = 1/10$)	0.1	38	0	100.00
RB ($\sigma = 1/10$)	1	32	0	100.00
RB ($\sigma = 1/10$)	20	32	0	100.00
RB ($\sigma = 1/50$)	0.1	38	0	100.00
RB ($\sigma = 1/50$)	1	36	0	100.00
RB ($\sigma = 1/50$)	20	36	0	100.00

+ 2.5