Homework 3　ZhiWeiYu

jordanyzw@mail.sjtu.edu.cn

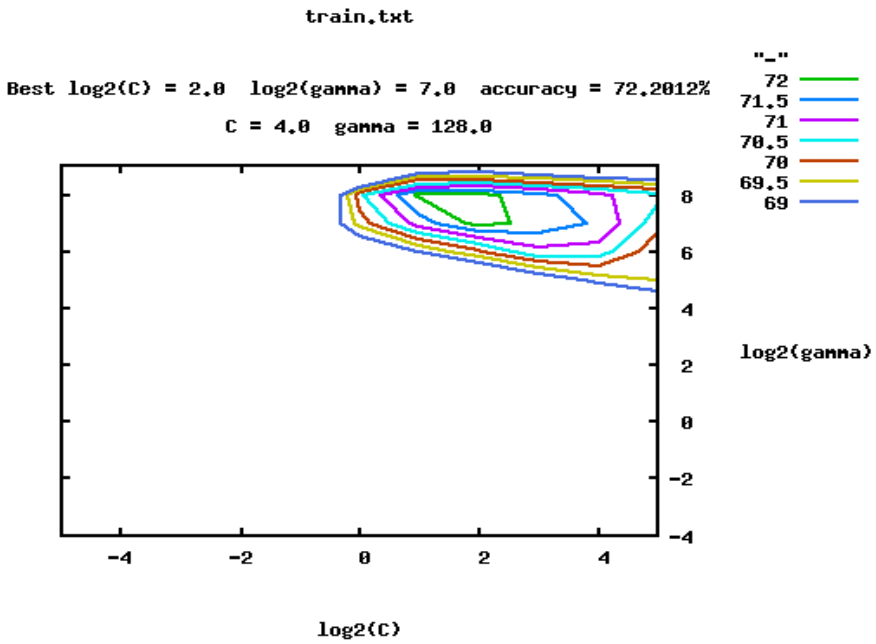
1 .preparetion:

To be convenient, I divide the large training data into different files which are named by their class labels.I put them into the trainfile directory.The python source file is named FileOperation.py .It worths saying that the source code was developed in Ubuntu python2.76.In order to run the source code .you have to intall numpy package.

2. OneVsOneSvm

The source file is named OneVsOneSvm.py. I build k(k-1)/2 two class models and store them in the OneVsOneModel directory. I choose the polynomial function as the kernel function because it is the best of three kernel functions.

(1) When use the RBF as kernel function ,the interface and the parameter is showed as svm_parameter('-t 2 -g 128 -c 4'),I use the cross validation method to get the best param gamma and C,And the following is the graph stating the relationship between training accuracy and the param gamma and C



As we choose the different param,we find that if we choose larger,the number of support vector is increasing and if we set small C,we can reduce the number

support vector.After using the cross validation method ,we find the best param is gamma=128,C=4. Finally,we get a test error rate of 0.0825627476882.

(2) when we use linear function as kernel function,the interface and parameter is svm_parameter('-t 0'),the training time is much faster,but the test result is really poor.we get a generalization error of 0.544253632761.It is because they are still not linear seperable after we use the linear function as the mapping function

(3) when we use polynomial function as the kernel function,the interface the parameter is svm_parameter('-t 1 –d 10 -r 1 –g 10'), the mapping function f(x)=(10*u'*v+1)^10.The training time does not cost much compared to the RBF， but get the best generalization performance,the test error rate is 0.051519154557.

Table for comparasion:

OneVsOneSvm Kernel Function	RBF	Linear	polynomial
Training time(s)	0.6599	0.5489	0.6140
Error rate	0.0825627476882	0.544253632761	0.0515191545575

3.OneVsRestSvm

The source file is named OneVsRestSvm.py.The k models are stored in the OneVsRestSvm directory and the kernel function is RBF

(1) when use the linear function as the kernel function, we get a test error rate 0.482826948481.The reason is also that the linear function does not map the original data into high dimension.Thus not increase the possibility for separating the two class with a hyperplane.

(2) When use the polynomial as the kernel function,I use the same parameter as OneVsOneSvm in order for cmparasion.We get a error rate of 0.0752972258917 for testing.It already performs good enough.

(3) When use RBF as the kernel function, I also use the same parameters as OneVsOneSvm.The generalization error is 0.0739762219287

Table for comparasion:

OneVsRestSvm Kernel Function	RBF	Linear	polynomial
Training time(s)	4.6121	3.3266	5.8704
Error rate	0.0739762219287	0.482826948481	0.0752972258917

4.PartVsPartSvm

The source file is named PartVsPartSvm.py.The 9*k(k-1)/2models are stored in the PartVsPartSvmModel directory and the kernel function is polynomial

(1) when use the linear function as the kernel function, we get a test error rate 0.554161162483.The reason is also that the linear function does not map the original data into high dimension.Thus not increase the possibility for separating the two class with a hyperplane.

(2) When use the polynomial as the kernel function,I use the same parameter as OneVsOneSvm in order for comparasion.We get a error rate 0.0449141347424 for testing.It already performs good enough.

(3) When use RBF as the kernel function, I also use the same parameters as OneVsOneSvm.The generalization error is 0.0832232496697

Table for comparasion:

PartVsPartSvm Kernel Function	RBF	Linear	polynomial
Training time(s)	3.2713	2.8516	3.2502
Error rate	0.0832232496697	0.554161162483	0.0449141347424

5.summary

For a K class problem ,the OneVsOneSvm has to construct k(k-1)/2 classifiers where each one is trained on data from two out of classes.It needs to construct more models than other methods.As for OneVsRestSvm needs to construc K classifiers.It may cause inbalance in some problems such that is may not predict a good result for the smaller class .When it comes to PartVsPartSvm,although it has to construct the most submodel , it can decompose the inbalance data into a comparatively balance data so that it will increase the possibility of good performance. We should also try many tips to find the best one ,such as use cross validation method to find the best gamma and C for RBF.