

### Assignment3

葛云皓 (116020910017)

#### Task1

Implement traditional one-versus-one and one-versus-rest task decomposition methods to solve a multi-class problem mentioned below.

##### 1、 one versus one

We use matlab circumstance to operate the libsvm. First to compile 4 most significant function

Libsvmtrain    libsvmpredict    Libsvmread    Libsvmwrite

(1)Load the train.txt and test.txt

(2)Devide the training data into 12 groups

(3)Reordering the data to prepare the training data

(4)Train the data with libsvmtrain function, and get  $12*(12-1)/2=66$  models.

(5)Predict the test data with each models and get the final predict by voting.

More details are in the attachment 'hw3\_onevsone.m'

##### 2、 one versus rest

We use matlab circumstance to operate the libsvm. First to compile 4 most significant function

Libsvmtrain    libsvmpredict    Libsvmread    Libsvmwrite

(1)Load the train.txt and test.txt

(2)Reordering the data to prepare the training data

(3)Change the entrance parameter of libsvmtrain function to set one versus rest structure by logic calculate.

(4)Train the data with libsvmtrain function, and get 12 models.

(5) Predict the test data with each models and get the final predict by maximum probably.

More details are in the attachment 'hw3\_onevsrest.m'

#### Task2

Implement part-versus-part task decomposition method to solve the same multi-class problem  
part-versus-part

(1) Load the train.txt and test.txt

(2) Devide the training data into 12 groups as the 'one'

(3) Use min-max methort devide the single group into different part.

(4) Train the combine of part to get the models

(5) Use min-max methord to predict the test data

More details are in the attachment 'hw3\_onevsrest.m'

#### Task3

Use two different kernel functions, namely linear and RBF, in all in all your classifiers.

To change the kernel function, we change the interface parameters of libsvmtrain

-t means the type of kernel function

0 - linear:  $u \cdot v$

1 - polynomial





















2 - RBFfunction

3 - sigmoid:

#### Task4

Compare the advantages and disadvantages of these three task decomposition methods.

Compare of the these three task decomposition methods with parameter of libsvmtrain(-c 1 -g 0.2 -t 0 -b 1)

Method	Kernel function	Time					Accuracy	Complexity
one versus one	linear	Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)	0.6063	middle
		hw3_onesone	1	16.103 s	1.199 s			
		libsvmtrain (MEX-file)	66	12.775 s	12.775 s			
		libsvmpredict (MEX-file)	66	2.008 s	2.008 s			
		libsvmread (MEX-file)	2	0.108 s	0.108 s			
		mode	1	0.014 s	0.013 s			
		ipermute	1	0.001 s	0.001 s			
one versus one	RBF	Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)	0.5991	middle
		hw3_onesone	1	21.886 s	1.242 s			
		libsvmtrain (MEX-file)	66	18.011 s	18.011 s			
		libsvmpredict (MEX-file)	66	2.513 s	2.513 s			
		libsvmread (MEX-file)	2	0.104 s	0.104 s			
		mode	1	0.015 s	0.015 s			
		ipermute	1	0.001 s	0.001 s			
one versus rest	linear	Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)	0.5707	concise
		hw3_onesall	1	24.173 s	0.048 s			
		libsvmtrain (MEX-file)	12	23.185 s	23.185 s			
		libsvmpredict (MEX-file)	12	0.834 s	0.834 s			
		libsvmread (MEX-file)	2	0.107 s	0.107 s			
one versus rest	RBF	Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)	0.5779	concise
		hw3_onesrest	1	34.842 s	0.042 s			
		libsvmtrain (MEX-file)	12	33.515 s	33.515 s			
		libsvmpredict (MEX-file)	12	1.183 s	1.183 s			
		libsvmread (MEX-file)	2	0.103 s	0.103 s			
part versus part	linear	Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)	0.6030	complex
		hw3_partvspart	1	23.036 s	2.902 s			
		libsvmtrain (MEX-file)	264	14.095 s	14.095 s			
		libsvmpredict (MEX-file)	330	5.904 s	5.904 s			
		libsvmread (MEX-file)	2	0.121 s	0.121 s			
		mode	1	0.014 s	0.014 s			
		ipermute	1	0.001 s	0.001 s			
part versus part	RBF	Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)	0.5680	complex
		hw3_partvspart	1	26.444 s	2.607 s			
		libsvmtrain (MEX-file)	264	17.064 s	17.064 s			
		libsvmpredict (MEX-file)	330	6.660 s	6.660 s			
		libsvmread (MEX-file)	2	0.099 s	0.099 s			
		mode	1	0.014 s	0.013 s			
		ipermute	1	0.001 s	0.001 s			

#### Conclusion

- 1 Without adjust parameter, the accuracy is not high enough to get the optimism prediction.
- 2 The Linear kernel may get a higher accuracy compared with RBF kernel in same parameter of libsvmtrain.
- 3 One versus one method may get the highest accuracy with the middle complexity.