SVM Experiment Report

Qiu Wei

March 16, 2017

Contents

Abstract

This document explains a experiment using libsvm to solve a k-class classification problem. This experiment used three methods to decompose a k-class classification problem into several two-class classification problems. A compare between RBF kernel function and linear kernal function is also done. This experiment can help us choose the right decompose method and kernal function when solving k-class classification problems.

1 Procedure

I tried three methods to decompose the origin problem: one versus one, one versus rest and part versus part.

For each two-class classification problem,The parameters are detected using grid.py which used cross validation and grid search to find the best

parameters. Each class are weighted by the number of instances in the other class.

1.1 One Versus One Method

The One Versus One Method(OVO) trains $\frac{k(k-1)}{2}$ models,each model seperates two single classes i and j. When classifying, the input vector will go through all the models,and each model will "vote" for their preferred classes. The class with the most votes will be the result.

1.2 One Versus Rest Method

The One Versus Rest Method(OVR) trains k models, each model seperates class i and the other classes.When classifying, the input vector go through all the models and if model i gave positive result, the input would be classified as class i. If multiple models tried to classify the input into their classes, the model with the best accuracy would be considered. If no model reacted, the input would be grouped in class -1.

1.3 Part Vesus Part Method

The Part Vesus Part Method(PVP) uses a technical inspired by binary search algorithm. The first model divides the classes into two parts with similar number of classes. After going through the first model, the chosen classes will be seperated by the next model. The possible classes will keep reducing until only one class is left. This method uses a total number of approximate n models for classifying.

2 Result

Method	Kernal Function	Accuracy
OVO	RBF	72.5891677675033 %
OVR	RBF	65.85204755614267 %
PVP	RBF	69.48480845442536 %
OVO	Linear	63.14398943196829 %
OVR	Linear	42.40422721268164 %
PVP	Linear	54.359313077939234 %

3 Analyse

3.1 Decomposing Method

From the table we can find the OVO method have the best accuracy,the next is PVP and the worst is OVR.

OVO method gets the best accuracy because it use $\frac{k(k-1)}{2}$ models to vote the most possible choise, and the input of each model is "balanced" : not like OVR. Whats more, the input of each model is simple so that it's even faster than other methods when k is not too large. However, if k is very large, OVO will take too much time to train the models and go through them.

OVR method does not get good accuracy because the input is hard to seperate: the single class may be too small than the rest classes. And I do not have a good method to deal with multiple positive results or no positive results. Though the number of models is less than OVO, each model will cost much more time to train as the input is the whole trainning set.

PVP method gets a good accuracy and takes less time than OVO.It may be a good choice when k is large.In this experiment I just divide the classes brutely: in their origin order. We may get a better result if a smarter dividing method is used.

3.2 Kernal Function

Linear kernal function get worse accuracy than RBF kernal. It's advantage is less training time.

4 Souce Code Explaination

All the source code are writen in python3.

To run the svm test:

```
$ cd src/
$ ./main.py
```

The program will read the svm models in models/ folder and use it to predict the test data. If one model is not exist, it will use tools/grid.py to do a grid search and generate a .model file in models/ folder.

Source code of different decomposing methods can be found in core/ , named ovo.py, ovr.py and pvp.py.

Util functions can be found in tools/util.py , I put many DSL functions in that file.