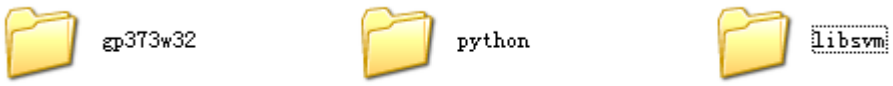
软件： libsvm , python ,gnuplot



gnuplot (解压，不需安装)
Libsvm（解压，不需安装）
Python(从 75 /share/软件 下载，安装)



文档目录：
一、基本概念
二、libsvm 操作简介
三、步骤总结

一 基本概念：

训练集，测试集，SVM，svmtrain，svmpredict，svmscale，数据格式，
参数，交叉验证 cross validation

训练集、测试集：训练集用于建立模型，测试集评估模型的预测等能力。
（好吧，通俗易懂不知道准不准确）

SVM：（support vector machine）现在最常拿来做分类，也就是说有一堆已经分好类的东西（分类依据是未知的），当收到新的东西，SVM 可以预测新的数据分到哪一类。

数据格式：[label] [Index1]:[value1] [index2]:[value2] [index3]:[value3]

[label]:类别（通常是整数）[index n]: 有顺序的索引 [value n]: 用来 train 的数据

为什么用这种格式？

Support vector machine ，vector（向量）也就是一排的 x1,x2,x3…。x[n]中的 n 即为 index n，值为 value n。这种格式的最大好处就是可以使用 spare matrix（稀疏矩阵），即有些 data 的属性值可以不存在。

Svmtrain：train 会接受特定格式的输入，产生一个 model 文件，这个 model 可以想象成是 svm 的内部文件，因为 predict 要 model 才能预测，而不能直接使用原始文件。

Svmtrain 语法：svmtrain [option] train_file [model_file]

Svmpredict：依照已经 train 好的 model ，输入新的数据，并输出预测新数据的类别。

Svmpredict 语法：svmpredict test_file mode_file output_file
要预测的数据 规则文件 输出结果

predict 完后，会顺便拿 predict 出来的值跟 test_file 里面写的值去做比对，这就是说，假设当 test_file 中的 label 是真正的分类结果，拿出来跟 predict 的结果比对，就可以知道 predict 有没有猜对，即显示预测的准确率。

Svmscale: 因为原始数据特征可能范围过大或过小，svmscale 可以先将数据重新 scale（缩放）

到适当范围。适当的 scale 有助于参数的选择，还有解 svm 的速度。

Svmscale 会对每个 attribute 做 scale ，通常是范围[0,1] 或[-1,1]，并且测试数据和训练数据要一起 scale。

交叉验证（cross validation）

1、把数据集分开轮流做训练集和测试集；

2、用某组参数去 train 并 predict 别组看正确率；

3、正确率不够的话，换参数再重复 train/predict；

4、找到一组不错的参数后，拿这组参数建 model 并用来对最后知数据的 predict；

n 重交叉验证法(n-fold cross validation)：可用 svmtrain 内置的 cross validation 功能 -v n n 是拆分的著述：比如 n=3，首先拿 1,2 来 train 并 predict 3；

其次拿 2,3 来 train 并 predict 1；

最后拿 1,3 来 train 并 predict 2；

参数：比较重要的参数是 gamma（-g）和 cost (-c) ，cost 的预设值是 1，gamma 预设值是 1/k , k 是输入数据的数据的记录数（？?）

参数的获取：自己 try 出来？？!!! 这样啊!!

Try 参数过程常用指数变化的方式来增加与减少参数的数值，因为有两组参数所以等于要 try n*n=n^2 次，这个过程是离散的，可以想象成假设在 x,y 平面上指定的范围内有一群格点（grid），每个格点的 x,y 经过换算（2^x , 2^y）就拿去当 cost 和 gamma 的值来 cross validation 。

（libsvm 的 python 目录下有个 grid.py 既是自动执行上述过程）。

二、libsvm 操作简介

过程：写程序输出正确格式的数据 -> 归一化（可选）-> 训练 -> 预测 -> 获得结果

这里的软件都安装解压在 E 盘（习惯了）

1、获得数据（没有现成数据可以先这样玩玩）

点击 E:\libsvm\libsvm-2.86\windows\svm-toy.exe （也可以在 cmd 中进入 E:\libsvm\libsvm-2.86\windows>svm-toy.exe 运行过程如图 5）跳出如下图 1，点击左下方的 change 改变点的颜色，然后随便乱点啦，图 2 .点击 run 获得 图 3（根据点的颜色不同分类）

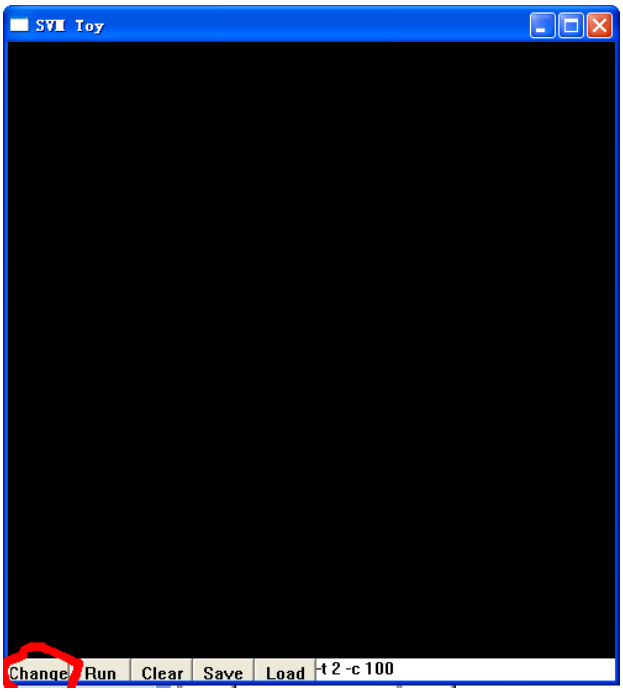


图 1

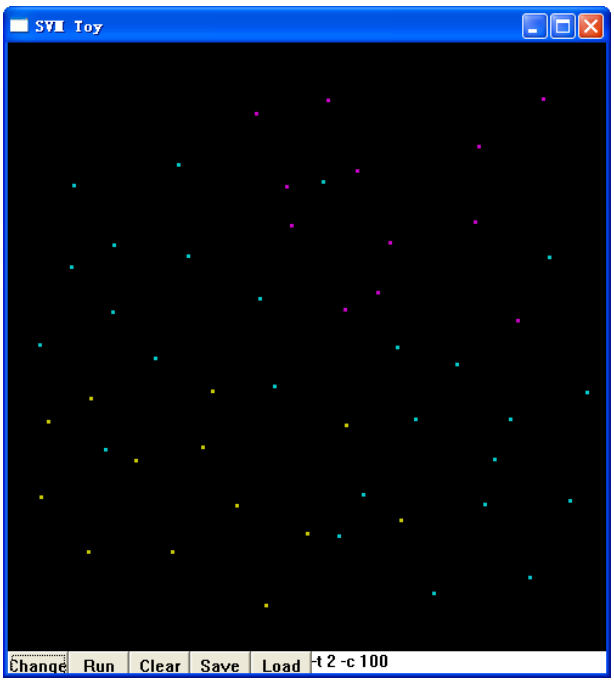


图 2

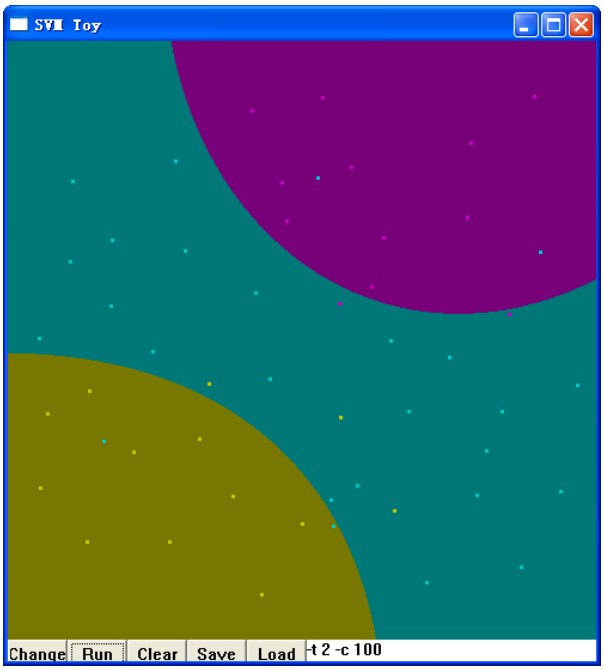


图 3

点击 sava 把数据导出 存为 test ， 获得如下图所示数据

```
1  1:0.170000 2:0.440000
2  1:0.172000 2:0.330000
3  1:0.050000 2:0.494000
4  1:0.102000 2:0.366000
5  1:0.106000 2:0.232000
6  1:0.278000 2:0.198000
7  1:0.294000 2:0.348000
8  1:0.240000 2:0.516000
9  1:0.158000 2:0.666000
10 1:0.436000 2:0.562000
11 1:0.412000 2:0.418000
12 1:0.516000 2:0.226000
13 2 1:0.268000 2:0.834000
14 2 1:0.490000 2:0.804000
15 2 1:0.318000 2:0.662000
16 2 1:0.554000 2:0.626000
17 2 1:0.334000 2:0.570000
18 2 1:0.134000 2:0.582000
19 2 1:0.052000 2:0.744000
20 2 1:0.208000 2:0.684000
21 2 1:0.064000 2:0.620000
22 2 1:0.130000 2:0.834000
```

图 4

Cmd 中运行 svm-toy.exe （从 C 盘到 E 盘直接 > E: 就 ok 了,开始我还真不懂，心虚 ing）

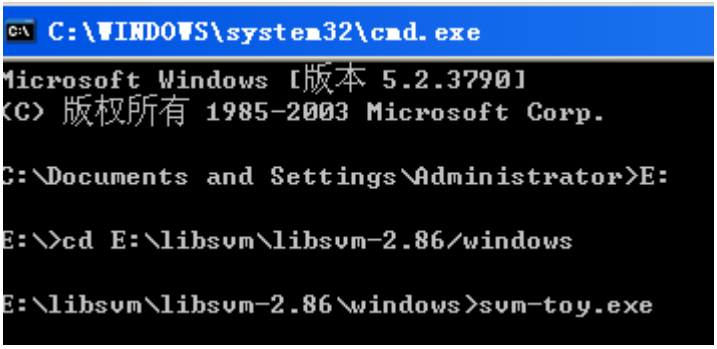
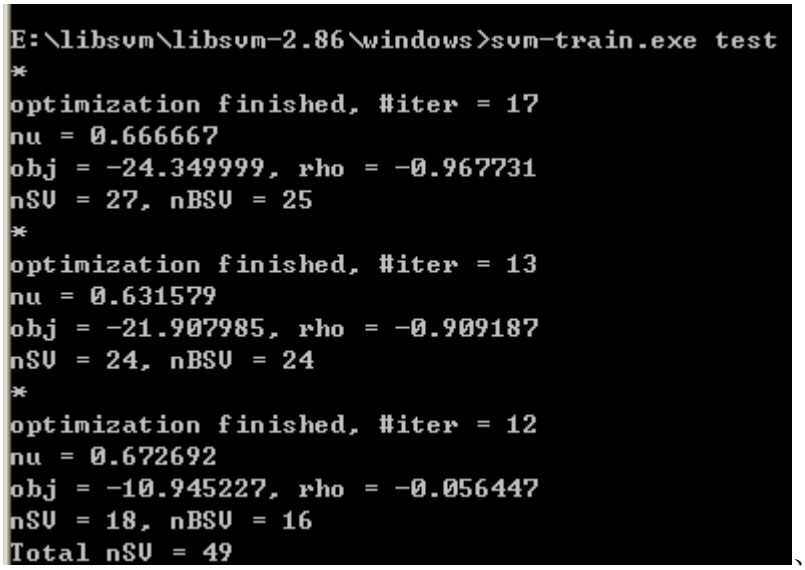


图 5

2、归一化(跳过)

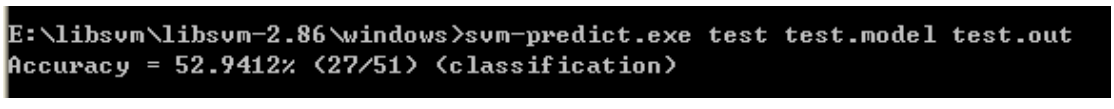
3、训练：E:\libsvm\libsvm-2.86\windows\svm-train.exe　test ，目录下生成.moedel 文件



、



4、预测：E:\libsvm\libsvm-2.86\windows>svm-predict.exe test test.model test.out，目录下生成，test.out 文件就是预测结果





打开，就是分类后的结果

```
1
1
1
1
1
3
1
1
1
1
1
1
1
1
1
1
1
1
```

貌似效果很烂

4、用 grid 优化 libsvm

<1>、把 E:\libsvm\libsvm-2.86\tools 目录下的 grid.py 复制到 E:\libsvm\libsvm-2.86\windows

<2>、编辑 gird.py 在 else 的地方

```
else:
        # example for windows
        svmtrain_exe = r"..\windows\svm-train.exe"
        gnuplot_exe = r"c:\tmp\gnuplot\bin\pgnuplot.exe"
```

修改路径名为绝对路径。

```
else:
        # example for windows
        svmtrain_exe = r"E:\libsvm\libsvm-2.86\windows\svm-train.exe"
        gnuplot_exe = r"E:\gp373w32\gp373w32\pgnuplot.exe"
```

<3>、进入 pythoon 的安装目录

cd E:\python

```
E:\python>python E:\libsvm\libsvm-2.86\windows\grid.py E:\libsvm\libsvm-2.86\windows\test
```

获得一堆数据：

```
[local] 13 -1 62.7451  (best c=0.5, g=8.0, rate=82.3529)
[local] 13 -13 58.8235  (best c=0.5, g=8.0, rate=82.3529)
[local] 13 1 56.8627  (best c=0.5, g=8.0, rate=82.3529)
[local] 13 -11 72.549  (best c=0.5, g=8.0, rate=82.3529)
[local] 13 -5 72.549  (best c=0.5, g=8.0, rate=82.3529)
[local] 13 -15 50.9804  (best c=0.5, g=8.0, rate=82.3529)
[local] 13 3 56.8627  (best c=0.5, g=8.0, rate=82.3529)
[local] 13 -9 72.549  (best c=0.5, g=8.0, rate=82.3529)
[local] 13 -3 74.5098  (best c=0.5, g=8.0, rate=82.3529)
0.5 8.0 82.3529
```

前面两个分别是 c 跟 g ， 再用这组参数 train 一下看看效果：

回到 libsvm 的 windows 目录

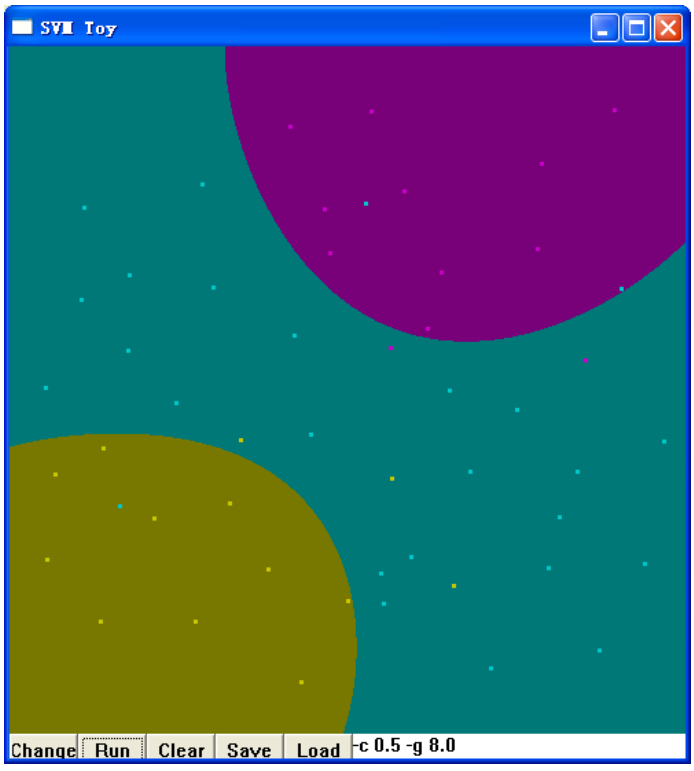
```
E:\libsvm\libsvm-2.86\windows>
```

```
E:\libsvm\libsvm-2.86\windows>svm-train.exe -c 0.5 -g 8.0 test
*
optimization finished, #iter = 29
nu = 0.591205
obj = -9.512685, rho = -0.528736
nSV = 28, nBSV = 20
*
optimization finished, #iter = 33
nu = 0.537714
obj = -7.969102, rho = -0.455895
nSV = 26, nBSV = 18
*
optimization finished, #iter = 22
nu = 0.425744
obj = -3.044524, rho = 0.056979
nSV = 13, nBSV = 7
Total nSV = 46
```

之后再 predict 一下

```
E:\libsvm\libsvm-2.86\windows>svm-predict.exe test test.model test.out
Accuracy = 84.3137% (43/51) (classification)
```

在 grid 之前是 52.9412% 很明显有很大的提升哦!!!!!!!



从图上的效果看……也比之前的精确了一些些……

补充： scale 用法

Svmscale 用来调整特征的大小范围，如果的特征变化范围不大，尽量不要归一化。训练数

据和测试数据要一起归一化，

同时归一化 test data 和 train data 的方法（目前还没用到，没试过，心虚 ing）

E:\libsvm\libsvm-2.86\windows>svm-scale.exe –s scale train_data > train_data .scale

E:\libsvm\libsvm-2.86\windows>svm-scale.exe –r scale test_data > test_data.scale

三 步骤总结

获得特定格式的数据

xfz_test 为测试集，xfz_train 为训练集

E:\libsvm\libsvm-2.86\windows\svm-scale.exe -s scale xfz_test>xfz_test.scale

E:\libsvm\libsvm-2.86\windows\svm-scale.exe -r scale xfz_train>xfz_train.scale

E:\python>phthon E:\……\windows\grid.py E:\……\windows\xfz_test.scale

获得 c g

E:\libsvm\libsvm-2.86\windows>svm-train -c XXX -g XXX xfz_test.scale

E:\libsvm\libsvm-2.86\windows>svm-predict xfz_train.scale xfz_test.scale.model
xfz_test.scale.out

Accuracy=XXXXX.XXXXXXXXXX%

上述过程也可以直接用一行指令

E:\python>python (路径名)\easy.py (路径名)\xfz_test (路径名)\xfz_train

前提是：把 libsvm 中 tool 目录下的 easy.py 复制到指定目录，打开并修改参数：

```
else:
        # example for windows
        svmscale_exe = r"..\windows\svm-scale.exe"
        svmtrain_exe = r"..\windows\svm-train.exe"
        svmpredict_exe = r"..\windows\svm-predict.exe"
        gnuplot_exe = r"c:\tmp\gnuplot\bin\pgnuplot.exe"
        grid_py = r".\grid.py"
```

例如，改为：

```
else:
        # example for windows
        svmscale_exe = r"E:\libsvm\libsvm-2.86\windows\svm-scale.exe"
        svmtrain_exe = r"E:\libsvm\libsvm-2.86\windows\svm-train.exe"
        svmpredict_exe = r"E:\libsvm\libsvm-2.86\windows\svm-predict.exe"
        gnuplot_exe = r"E:\gp373w32\gp373w32\pgnuplot.exe"
        grid_py = r"E:\libsvm\libsvm-2.86\windows\grid.py"
```

```
E:\python>python E:\libsvm\libsvm-2.86\windows\easy.py E:\libsvm\libsvm-2.86\windows\test
Scaling training data...
Cross validation...
Best c=8.0, g=0.5 CV rate=80.3922
Training...
Output model: test.model
```

----------by MYZ xiaofengzi

2012/7/10