第6章 统计学习方法

班级： 无61 姓名： 李溢 学号： 2016011235

1. **Fisher线性判别分析 ( Linear Discriminant Analysis)**

考虑以下6个训练样本点：



(a)试用Fisher线性判别分析获取投影向量和分类界面，并用图画出分类界面。

(b)假设新的测试样本为，试结合上述分类界面做出分类判决。

（a）均值

类内散度

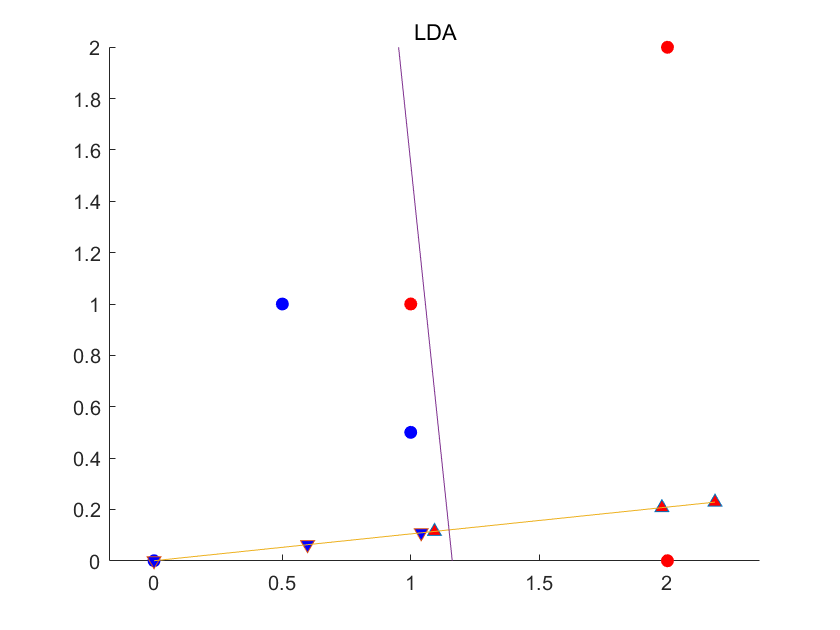
类间散度

投影向量

投影阈值

分类界面，即

如图所示，可以看出用两个类别均值的中心点作为分类阈值并不能准确的进行分类。更好的方法是分别计算两个类别中每个样本在投影向量方向的投影值，然后选择两个类别中最接近的两个投影点，可以认为他们刚好位于各自类别的“边界处”，然后取他们的中点作为判决阈值。



（b）投影值

因此被判决为**类别1**

1. **支持向量机 (Support Vector Machines)**

考虑以下6个训练样本点：



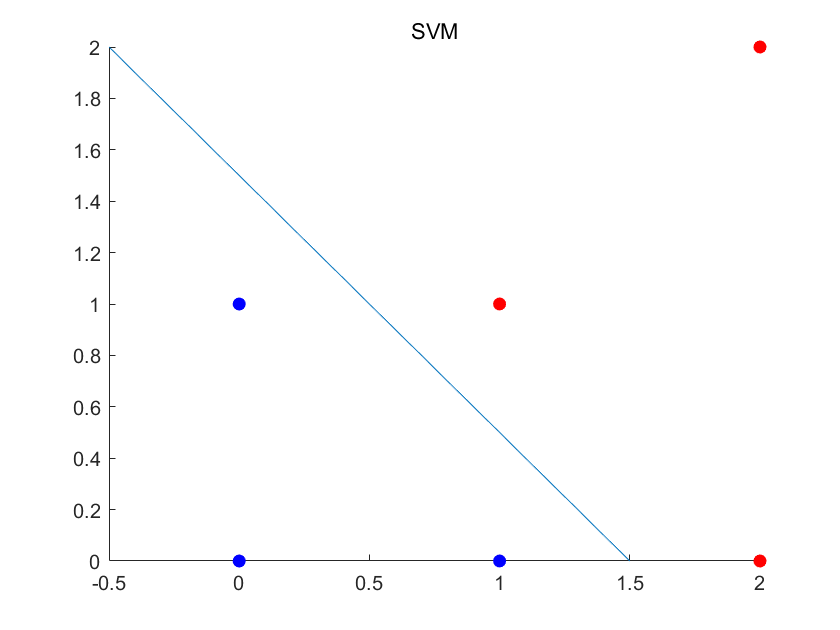
1. 在图中画出6个训练点，并构造最优间隔的权向量和对应的最优超平面。
2. 哪些是支持向量？
3. 假设在 加入一样本，试问在满足什么条件下样本线性可分，并给出在线性可分条件下支持向量机的分类界面(基于的表达式)。

（a，b）通过观察法，可以获得支持向量为

因此可以求解

得到

对应的最优超平面为



（c）若要线性可分，应该有g < 1

**当0 <= g < 1时**，可以得到支持向量为

因此类比前一问的求解方法，可以得到分类界面为

**当 g < 0时**，支持向量仍为

因此分类界面为

1. **决策树 (Decision Tree)**

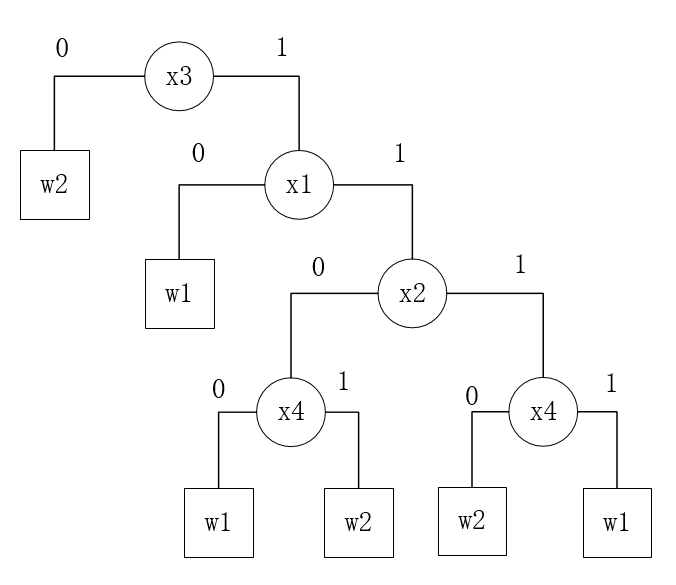
考虑两类分类问题，每一个样本点对应的数据为4个二进制值的属性

|  |  |
| --- | --- |
|  |  |
| 0110 | 1011 |
| 1010 | 0000 |
| 0011 | 0100 |
| 1111 | 1110 |

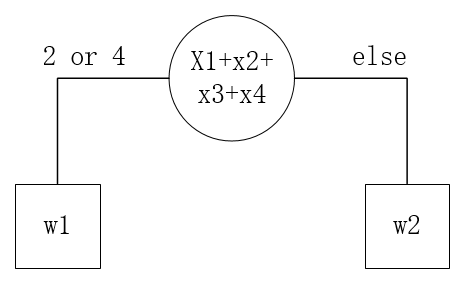
1. 以特征的每一位作为判别测试条件，结合Gini不纯度，生成决策分类树。
2. 新来测试样本0101，试给出其基于上述决策分类树的判决结果。

（a）记从左至右四个bit为分别为x1,x2,x3,x4

如果每次只能选择一个bit作为判决依据，则得到的决策树为



如果每次可以选择不只一个bit位作为判决依据，则得到的决策树为



（b）如果使用第一个决策树，由于x3=0，判决为w2

如果使用第二个决策树，由于0101有2个1，判决为w1

1. **(上机题) 支持向量机**

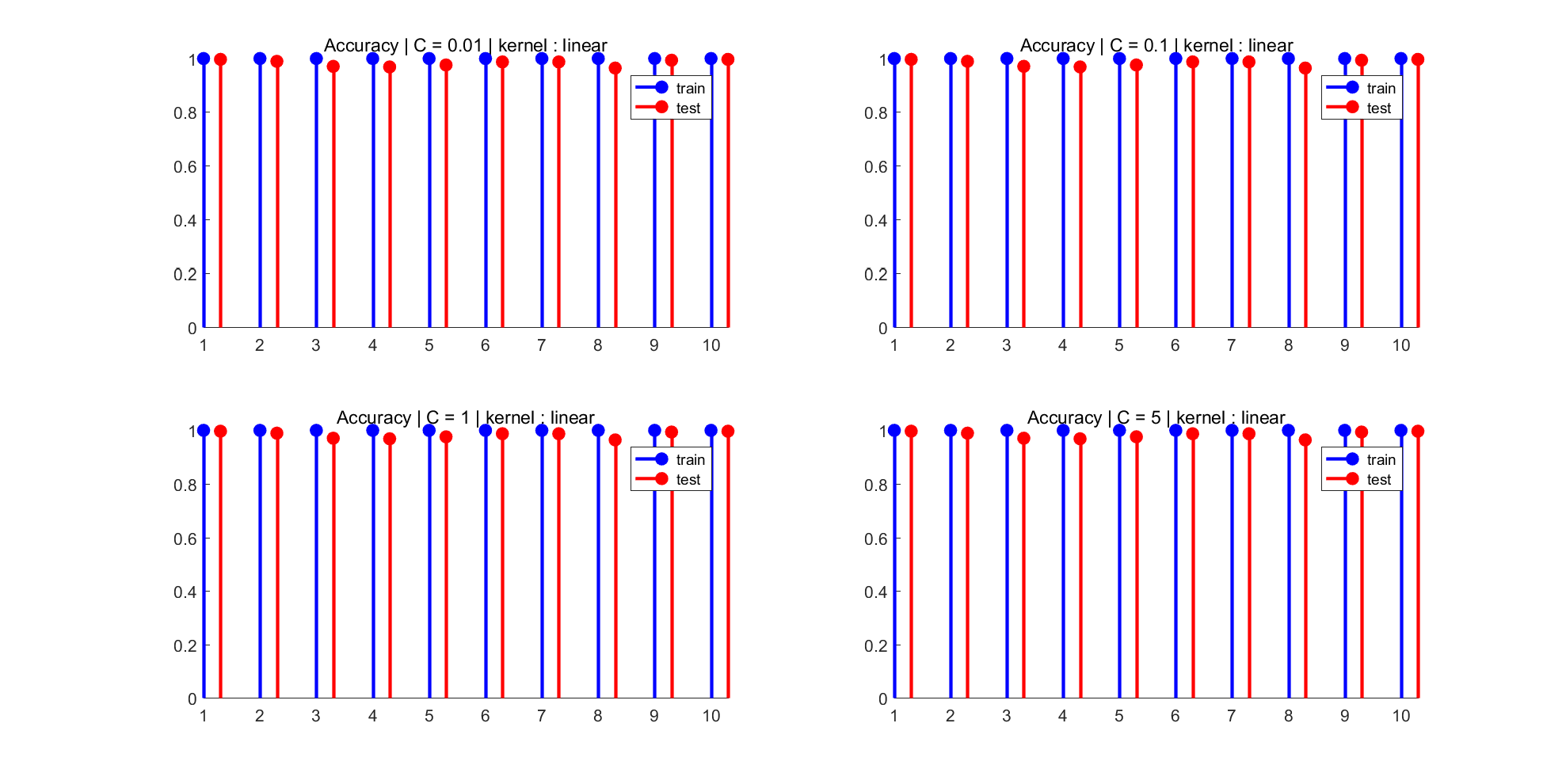
利用支持向量机实现图像分类。附件中是利用VGG深度神经网络提取到的Caltech256数据集中10类目标的特征，试基于SVM结合上述深度学习特征对图像进行分类。

1. 利用线性SVM在训练集上学习模型参数，变换两个以上C值，统计不同C值对应的训练集和测试集上的各类分类准确率；
2. 利用基于RBF高斯核函数在训练集上学习模型参数，变换两个以上C值，统计不同C值对应的训练集和测试集上的各类分类准确率；

（a）**理论分析**：C代表了对松弛因子的惩罚系数，C越大，则会使得松弛因子 越小，因此对噪声和异常值的容忍度也更小。因此理论上C较小时，SVM会有更大的可能把一些分界面附近的样本点当作噪声样本而误分类，因此准确率可能较低。当C增大后，对噪声容忍度减小，因此训练集分类准确率上升。

**实验结果**上，使用线性核函数时：**改变C的值对分类准确率基本没影响**，且整体分类效果较好，测试集上各个类别都可以达到96%以上，最好的可以达到99.7%。因此可以推断**提取的Caltech256数据集中10类目标的特征是大致线性可分的**。

实验结果如下图所示：



准确值如下：

------------- C = 0.01 | kernel : linear -------------

train accuracy for class 1 is: 1.000000

test accuracy for class 1 is: 0.996859

train accuracy for class 2 is: 1.000000

test accuracy for class 2 is: 0.989529

train accuracy for class 3 is: 1.000000

test accuracy for class 3 is: 0.970681

train accuracy for class 4 is: 1.000000

test accuracy for class 4 is: 0.968586

train accuracy for class 5 is: 1.000000

test accuracy for class 5 is: 0.975916

train accuracy for class 6 is: 1.000000

test accuracy for class 6 is: 0.987435

train accuracy for class 7 is: 1.000000

test accuracy for class 7 is: 0.987435

train accuracy for class 8 is: 1.000000

test accuracy for class 8 is: 0.964398

train accuracy for class 9 is: 1.000000

test accuracy for class 9 is: 0.993717

train accuracy for class 10 is: 1.000000

test accuracy for class 10 is: 0.996859

------------- C = 0.1 | kernel : linear -------------

train accuracy for class 1 is: 1.000000

test accuracy for class 1 is: 0.996859

train accuracy for class 2 is: 1.000000

test accuracy for class 2 is: 0.989529

train accuracy for class 3 is: 1.000000

test accuracy for class 3 is: 0.970681

train accuracy for class 4 is: 1.000000

test accuracy for class 4 is: 0.968586

train accuracy for class 5 is: 1.000000

test accuracy for class 5 is: 0.975916

train accuracy for class 6 is: 1.000000

test accuracy for class 6 is: 0.987435

train accuracy for class 7 is: 1.000000

test accuracy for class 7 is: 0.987435

train accuracy for class 8 is: 1.000000

test accuracy for class 8 is: 0.964398

train accuracy for class 9 is: 1.000000

test accuracy for class 9 is: 0.993717

train accuracy for class 10 is: 1.000000

test accuracy for class 10 is: 0.996859

------------- C = 1 | kernel : linear -------------

train accuracy for class 1 is: 1.000000

test accuracy for class 1 is: 0.996859

train accuracy for class 2 is: 1.000000

test accuracy for class 2 is: 0.989529

train accuracy for class 3 is: 1.000000

test accuracy for class 3 is: 0.970681

train accuracy for class 4 is: 1.000000

test accuracy for class 4 is: 0.968586

train accuracy for class 5 is: 1.000000

test accuracy for class 5 is: 0.975916

train accuracy for class 6 is: 1.000000

test accuracy for class 6 is: 0.987435

train accuracy for class 7 is: 1.000000

test accuracy for class 7 is: 0.987435

train accuracy for class 8 is: 1.000000

test accuracy for class 8 is: 0.964398

train accuracy for class 9 is: 1.000000

test accuracy for class 9 is: 0.993717

train accuracy for class 10 is: 1.000000

test accuracy for class 10 is: 0.996859

------------- C = 5 | kernel : linear -------------

train accuracy for class 1 is: 1.000000

test accuracy for class 1 is: 0.996859

train accuracy for class 2 is: 1.000000

test accuracy for class 2 is: 0.989529

train accuracy for class 3 is: 1.000000

test accuracy for class 3 is: 0.970681

train accuracy for class 4 is: 1.000000

test accuracy for class 4 is: 0.968586

train accuracy for class 5 is: 1.000000

test accuracy for class 5 is: 0.975916

train accuracy for class 6 is: 1.000000

test accuracy for class 6 is: 0.987435

train accuracy for class 7 is: 1.000000

test accuracy for class 7 is: 0.987435

train accuracy for class 8 is: 1.000000

test accuracy for class 8 is: 0.964398

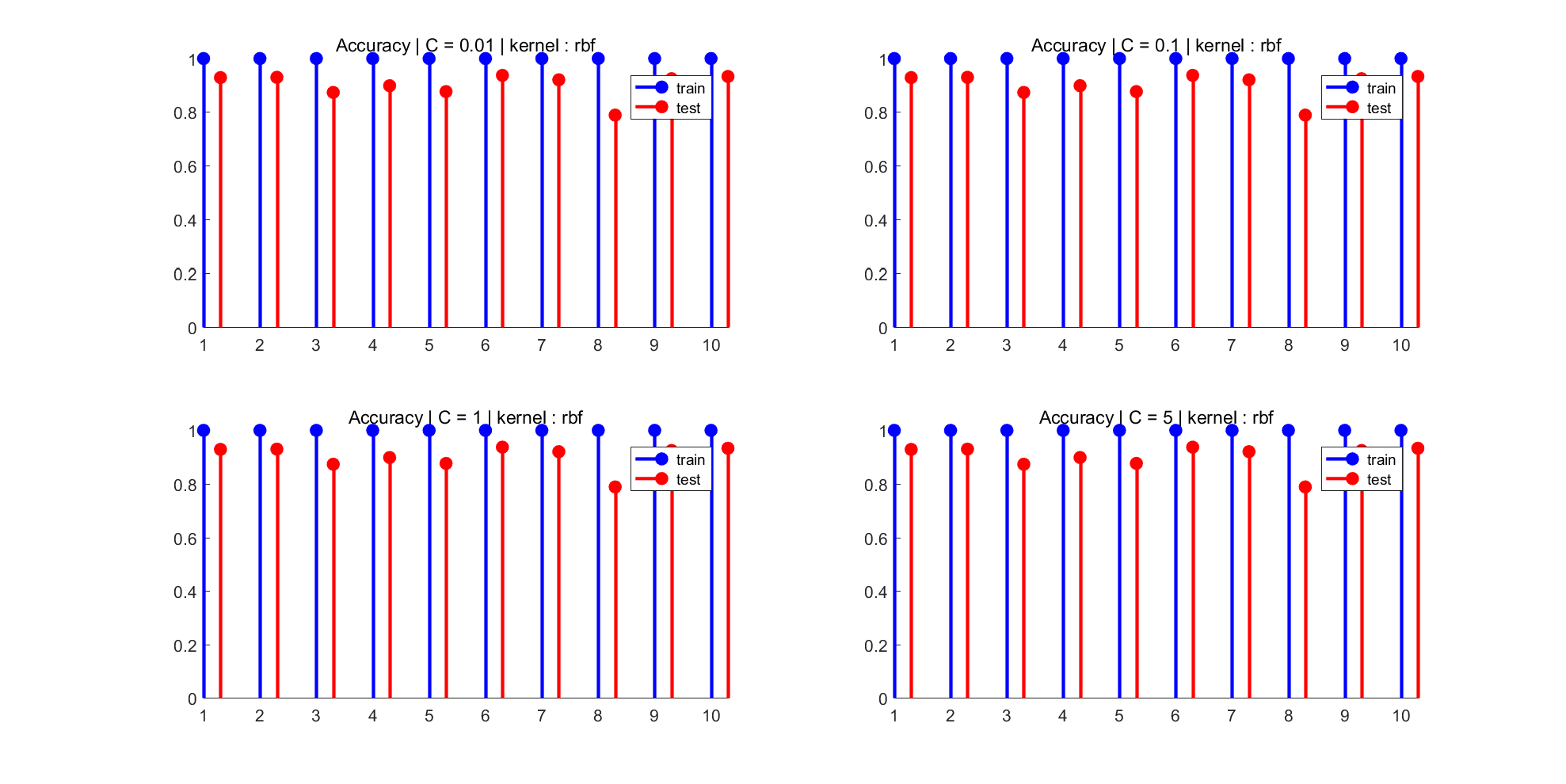
train accuracy for class 9 is: 1.000000

test accuracy for class 9 is: 0.993717

train accuracy for class 10 is: 1.000000

test accuracy for class 10 is: 0.996859

（b）理论分析与上面相同，实验中，使用基于RBF高斯核函数时：无论C怎么变化，SVM在测试集的准确率可达100%，但在训练集上的分类准确率最高可以达到93%，效果略差于线性SVM**。**



------------- C = 0.01 | kernel : rbf -------------

train accuracy for class 1 is: 1.000000

test accuracy for class 1 is: 0.928796

train accuracy for class 2 is: 1.000000

test accuracy for class 2 is: 0.929843

train accuracy for class 3 is: 1.000000

test accuracy for class 3 is: 0.873298

train accuracy for class 4 is: 1.000000

test accuracy for class 4 is: 0.898429

train accuracy for class 5 is: 1.000000

test accuracy for class 5 is: 0.876440

train accuracy for class 6 is: 1.000000

test accuracy for class 6 is: 0.937173

train accuracy for class 7 is: 1.000000

test accuracy for class 7 is: 0.920419

train accuracy for class 8 is: 1.000000

test accuracy for class 8 is: 0.788482

train accuracy for class 9 is: 1.000000

test accuracy for class 9 is: 0.924607

train accuracy for class 10 is: 1.000000

test accuracy for class 10 is: 0.932984

------------- C = 0.1 | kernel : rbf -------------

train accuracy for class 1 is: 1.000000

test accuracy for class 1 is: 0.928796

train accuracy for class 2 is: 1.000000

test accuracy for class 2 is: 0.929843

train accuracy for class 3 is: 1.000000

test accuracy for class 3 is: 0.873298

train accuracy for class 4 is: 1.000000

test accuracy for class 4 is: 0.898429

train accuracy for class 5 is: 1.000000

test accuracy for class 5 is: 0.876440

train accuracy for class 6 is: 1.000000

test accuracy for class 6 is: 0.937173

train accuracy for class 7 is: 1.000000

test accuracy for class 7 is: 0.920419

train accuracy for class 8 is: 1.000000

test accuracy for class 8 is: 0.788482

train accuracy for class 9 is: 1.000000

test accuracy for class 9 is: 0.924607

train accuracy for class 10 is: 1.000000

test accuracy for class 10 is: 0.932984

------------- C = 1 | kernel : rbf -------------

train accuracy for class 1 is: 1.000000

test accuracy for class 1 is: 0.928796

train accuracy for class 2 is: 1.000000

test accuracy for class 2 is: 0.929843

train accuracy for class 3 is: 1.000000

test accuracy for class 3 is: 0.873298

train accuracy for class 4 is: 1.000000

test accuracy for class 4 is: 0.898429

train accuracy for class 5 is: 1.000000

test accuracy for class 5 is: 0.876440

train accuracy for class 6 is: 1.000000

test accuracy for class 6 is: 0.937173

train accuracy for class 7 is: 1.000000

test accuracy for class 7 is: 0.920419

train accuracy for class 8 is: 1.000000

test accuracy for class 8 is: 0.788482

train accuracy for class 9 is: 1.000000

test accuracy for class 9 is: 0.924607

train accuracy for class 10 is: 1.000000

test accuracy for class 10 is: 0.932984

------------- C = 5 | kernel : rbf -------------

train accuracy for class 1 is: 1.000000

test accuracy for class 1 is: 0.928796

train accuracy for class 2 is: 1.000000

test accuracy for class 2 is: 0.929843

train accuracy for class 3 is: 1.000000

test accuracy for class 3 is: 0.873298

train accuracy for class 4 is: 1.000000

test accuracy for class 4 is: 0.898429

train accuracy for class 5 is: 1.000000

test accuracy for class 5 is: 0.876440

train accuracy for class 6 is: 1.000000

test accuracy for class 6 is: 0.937173

train accuracy for class 7 is: 1.000000

test accuracy for class 7 is: 0.920419

train accuracy for class 8 is: 1.000000

test accuracy for class 8 is: 0.788482

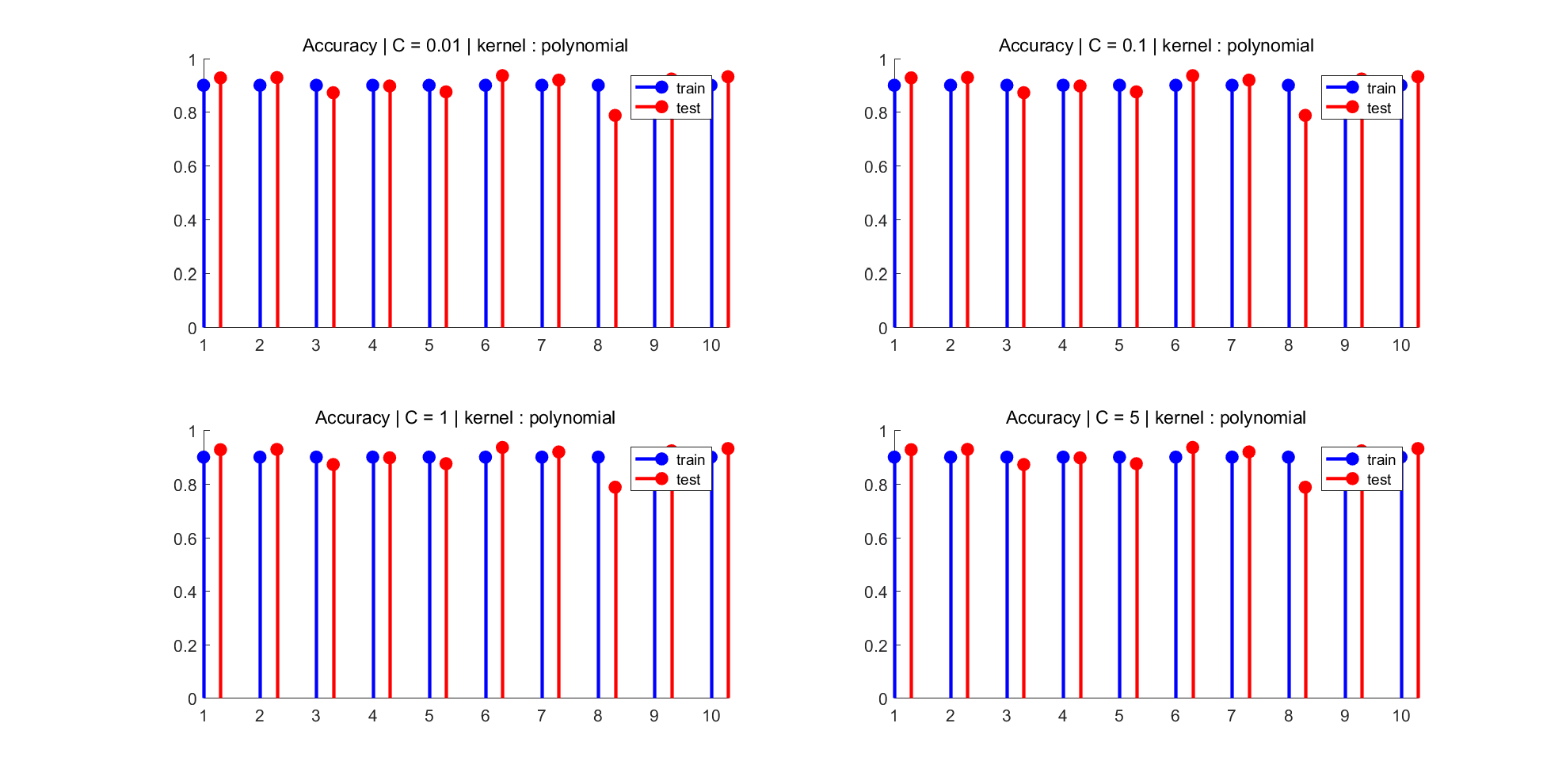
train accuracy for class 9 is: 1.000000

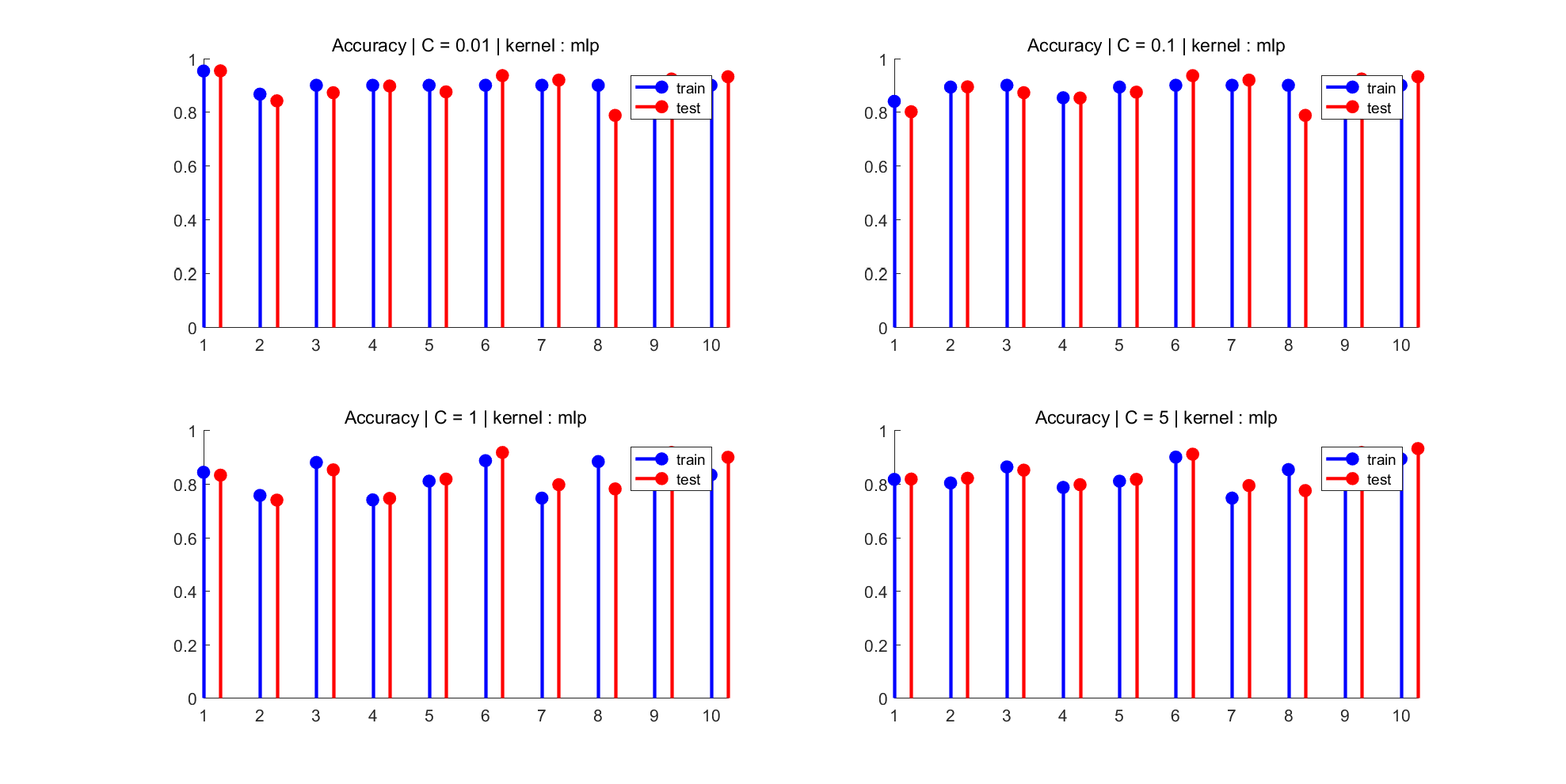
test accuracy for class 9 is: 0.924607

train accuracy for class 10 is: 1.000000

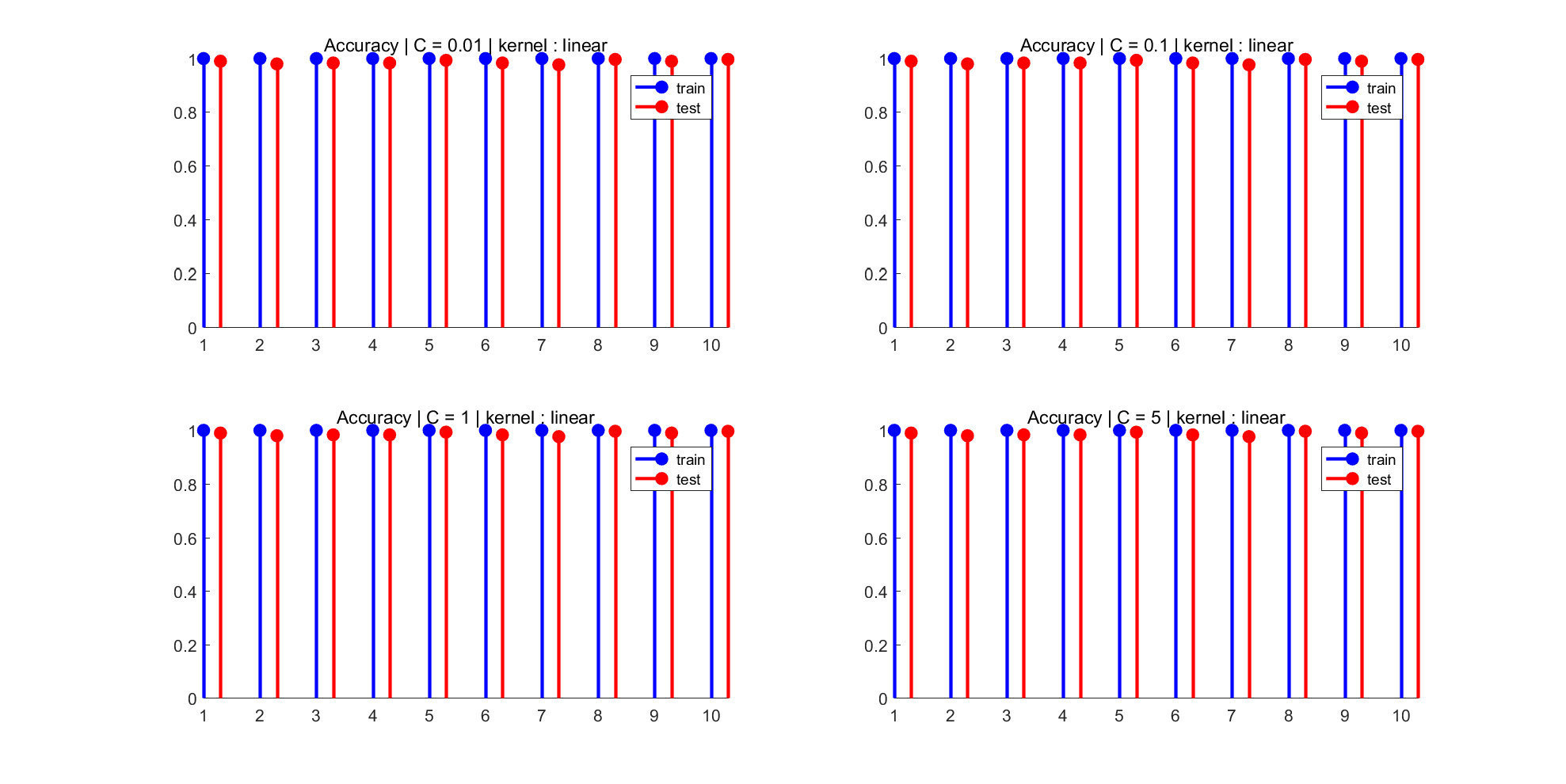
test accuracy for class 10 is: 0.932984

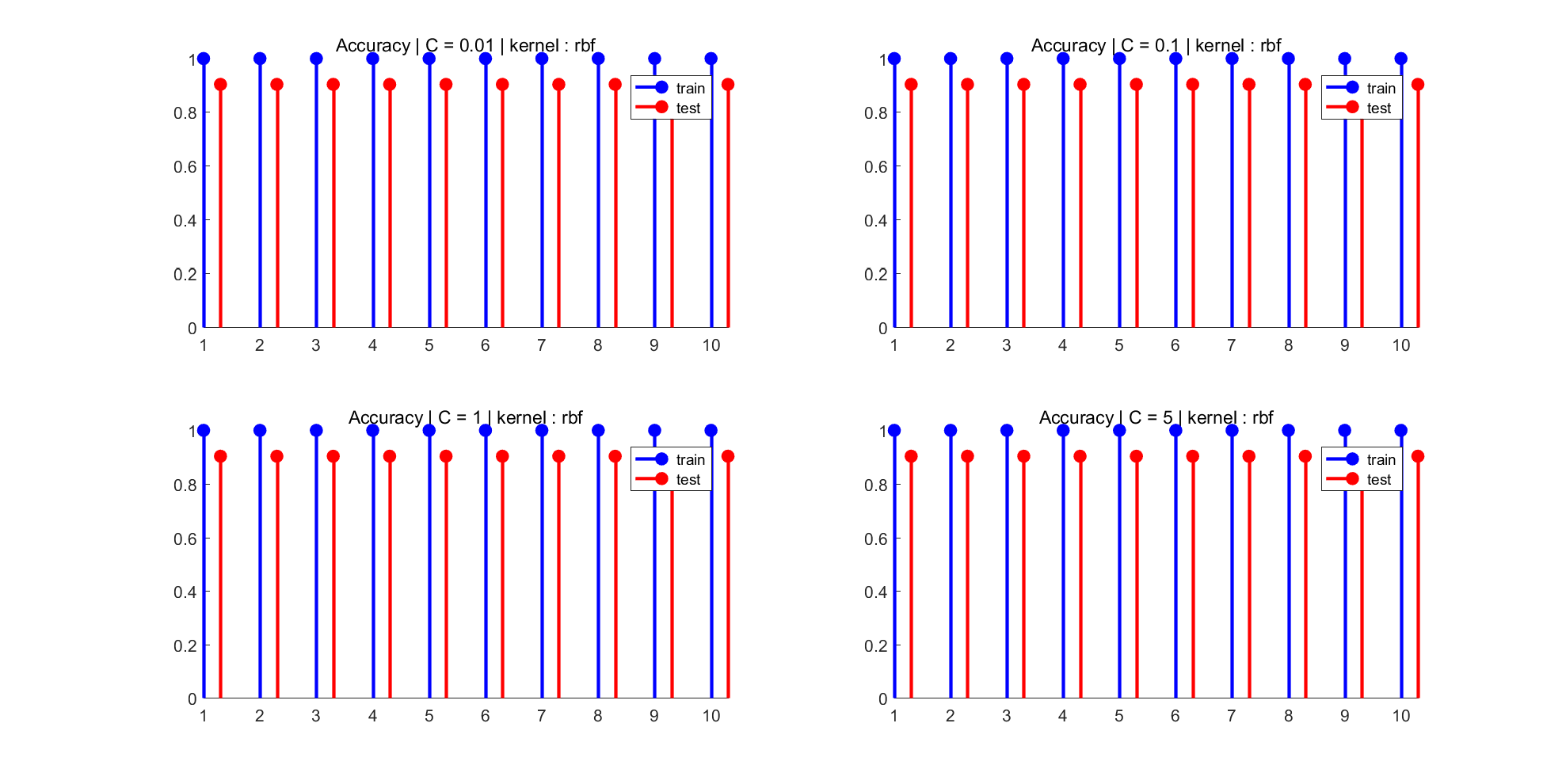
注：除了上面两种方法，我还尝试了另外两种核函数： polynomial、mlp，并取不同的C值，得到的结果如图所示。可以发现他们的效果都不如线性SVM，甚至在训练集上都不能达到准确率100%，可见**Caltech256数据集中10类目标的特征近似为线性可分的。**

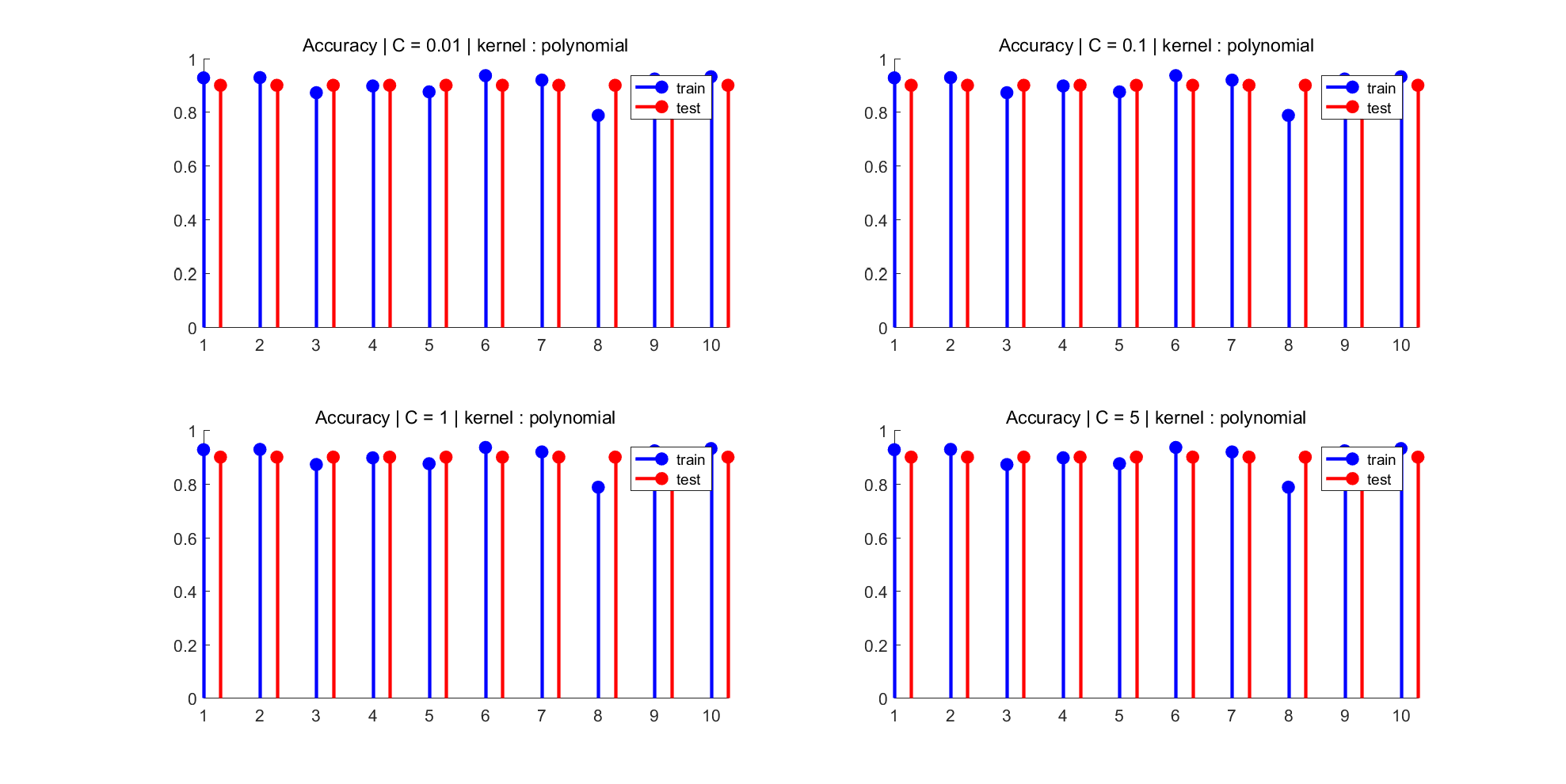
****

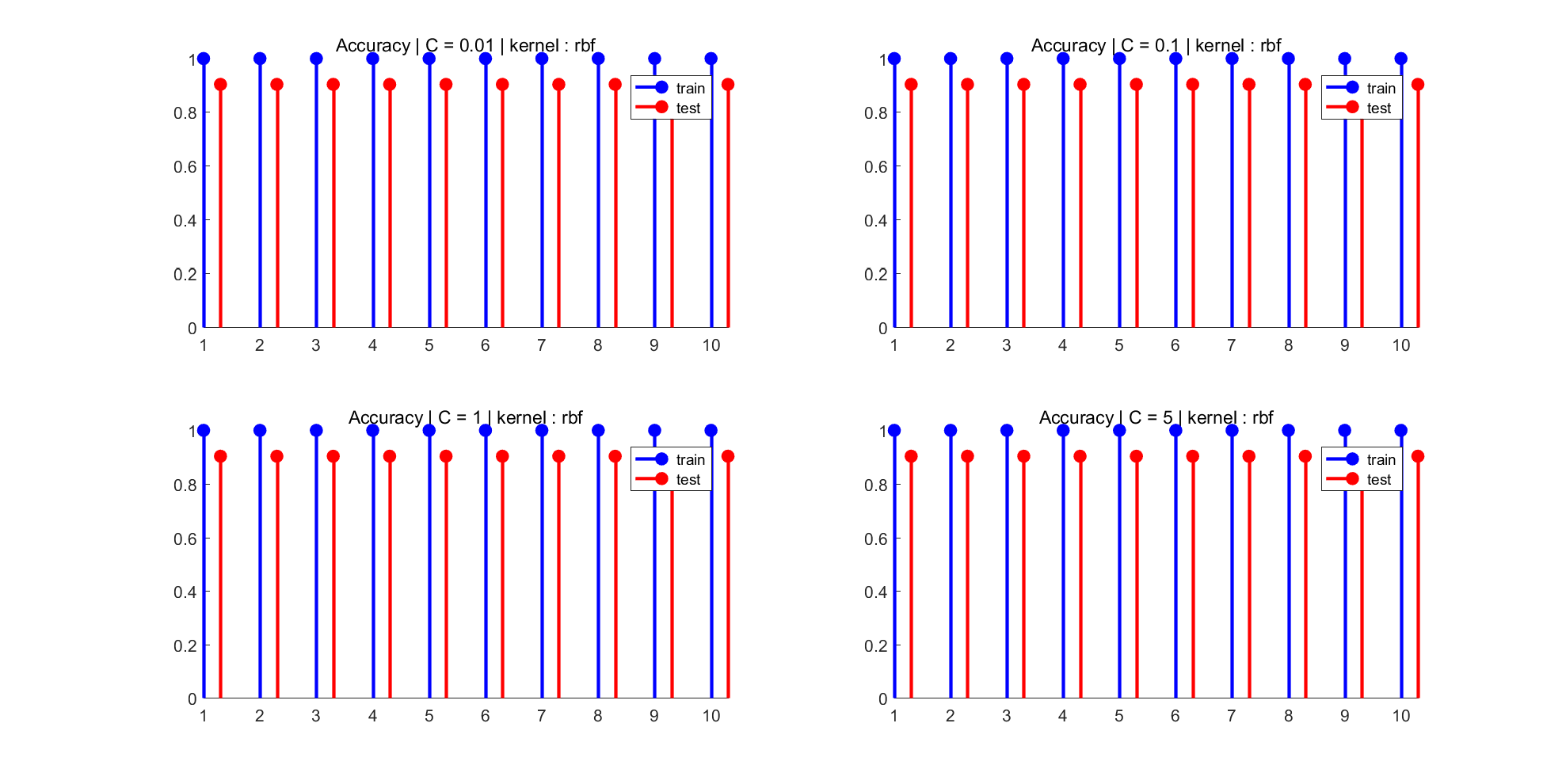


由于训练集只有300个样本，而测试集有955个样本，之后我交换了训练集和测试集重新进行训练，发现与前面的结论基本相同，各个方法效果都有一定程度提升。









注：上面所描述的方法中，我将测试结果数据都放在了accuracy\_matlab.txt文件中。

附：matlab源代码

%% svm image classification

clear all; close all; clc;

% load data

load Caltech-256\_VGG\_10classes.mat

x\_test = traindata.X;

y\_test = traindata.y;

x\_train = testdata.X;

y\_train = testdata.y;

train\_accr = zeros(1,10);

test\_accr = zeros(1,10);

%调用SVM训练，设置参数，包括svmtrain/svmclassify

X = 1:10;

C = [0.01 0.1 1 5];

kernel = {'linear', 'polynomial', 'rbf', 'mlp'};

for j = 1:length(kernel) %设置不同核函数

figure;

set(gcf,'outerposition',get(0,'screensize'));

hold on;

for i = 1:length(C) %设置不同C值

disp(['------------- C = ',num2str(C(i)),' | kernel : ',char(kernel(j)),' -------------']);

for k = 1 : 10

model = svmtrain(x\_train', (y\_train == k)',...

'boxconstraint', C(i),...

'kernel\_function', char(kernel(j)));

y\_pred = svmclassify(model, x\_train');

label = (y\_train == k);

train\_accr(k) = length(find(y\_pred' == label))/length(label);

fprintf( 'train accuracy for class %d is: %f\n',...

k, train\_accr(k));

y\_pred = svmclassify(model, x\_test');

label = (y\_test == k);

test\_accr(k) = length(find(y\_pred' == label))/length(label);

fprintf( 'test accuracy for class %d is: %f\n',...

k, test\_accr(k));

end

subplot(2,2,i);

hold on;

title(['Accuracy | C = ', num2str(C(i)),' | kernel : ',char(kernel(j))]);

stem(X,train\_accr,'b','filled','linewidth',2);

stem(X+0.3,test\_accr,'r','filled','linewidth',2);

legend('train','test');

%axis tight;

axis([1 10.3 0 1]);

saveas(gcf,['accur\_',char(kernel(j)),'\_2.png']);

end

end

数据说明：附件中的.mat文件来自对Caltech256数据集中的前10类提取VGG特征获取，其中训练集包含300个样本(traindata)，测试集包含955个样本(testdata)；

程序说明：SVM已被嵌入至多个机器学习工具包里面，以下供参考。

**附1： Python例程: 利用sklearn中的SVM**

from sklearn import svm

import numpy as np

import scipy.io as sio

# 装载数据

data = sio.loadmat('Caltech-256\_VGG\_10classes.mat')

traindata = data['traindata']

testdata = data['testdata']

x\_train = traindata[0][0][0].transpose()

y\_train = traindata[0][0][1].ravel()

x\_test = testdata[0][0][0].transpose()

y\_test = testdata[0][0][1].ravel()

# check if the data have been correctly loaded

print x\_train.shape

print y\_train.shape

# 调用SVM，设置参数，请查看SVC的用法

model = svm.SVC()

# 学习模型参数

model.fit(x\_train,y\_train)

# 输出识别准确率

print "SVM-training accuracy:", model.score(x\_train,y\_train)

y\_hat=model.predict(x\_train)

# 计算训练集各类别的识别准确率，请根据准确率定义写出计算公式

print "SVM-testing accuracy", model.score(x\_test,y\_test)

y\_hat=model.predict(x\_test)

# 计算测试集各类别的识别准确率，同上

**附2：Matlab例程：调用svmtrain svmclassify函数**

% 装载数据

load 'Caltech-256\_VGG\_10classes.mat';

x\_train = traindata.X;

y\_train = traindata.y;

x\_test = testdata.X;

y\_test = testdata.y;

%调用SVM训练，设置参数，包括svmtrain/svmclassify¨

for k = 1 : 10

model = svmtrain(x\_train, (y\_train == k) ); %此处需修改

y\_pred = svmclassify( ); %´此处需修改

label = (y\_train == k);

fprintf( 'training accuracy for class %d is: %f\n',…

k, length(find(y\_pred' == label))/length(label) );

y\_pred = svmclassify(model, x\_test' );

label = (y\_test == k);

fprintf( 'test accuracy for class %d is: %f\n', …

k, length(find(y\_pred' == label))/length(label) );

end

**附3：利用libsvm, 见** [**https://www.csie.ntu.edu.tw/~cjlin/libsvm/**](https://www.csie.ntu.edu.tw/~cjlin/libsvm/)

具体步骤同附2，函数调用形式不同