用 Matlab 中的 Libsvm 对 Wine data 进行分类

1、 背景

我们拥有一个数据集，有 3 种不同类别的葡萄酒，每个葡萄酒有 13 个特征变量，总共有 187 个样本，我们选择其中样本得 50%作为训练样本另外的 50%作为测试样本，我们通过训练样本对 SVM 进行训练，然后用另一部分的数据对于训练结果进行验证，最后结果的准确率达 98.8764%。这是一个简单的多类分类问题，运用的软件为 matlab，需要下载 libsvm。

2、 流程图

数据集的 13 个属性为：*酒精 Alcohol*、*苹果酸 Malic acid*、*灰分的碱度 Alcalinity of ash*、*镁(元素)Magnesium*、*总酚含量 Total phenols*、*黄酮类化合物 flavanoids*、*酚 Nonflavanoid phenols*、*原花青素 proanthocyanins*、*色泽度 color intensitys*、*色调 hue*、*淡酒 OD280/OD315 of diluted wines*、*脯氨酸 proline*

3、 完整代码

```
%葡萄酒数据预处理
uiimport('wine.data');
%上述代码需要单独敲入，选择导入格式为矩阵
wine_label = wine(:, 1);
wine_data = wine(:, 2:end);
categories = {'Alcohol'; 'Malic acid'; 'Ash'; 'Alcalinity of ash'; 'Magnesium'; 'Total phenols'; 'Flavanoids'; 'Nonflavanoid phenols'; 'Proanthocyanins'; 'Color intensitys'; 'Hue'; 'OD280/OD315 of diluted wines'; 'Proline'};
classnumber = 3;
%上述过程目的是将表分开，利用其他的变量进行储存
save winedata.mat;
%储存为葡萄酒数据文件
load winedata;
%载入葡萄酒数据集
```

%画出测试数据框图%所用为 boxplot(x,g)，x 为输入数据，g 为分组变量，其余元素为修改一些修饰图，'horizontal'是 'orientation' 的属性值，'labels'表示框标签，代码中的 g 表示分类数组，下面的也类似，后面是前面的属性值

```
figure;
boxplot(wine_data, 'orientation', 'horizontal', 'labels', categories);
title('Wine Data Box Figure', 'FontSize',12);
xlabel('Attribute Value', 'FontSize', 12);
grid on;
```

%显示测试数据的分形维数图

%subplot(m,n,p) 将当前图窗划分为 m×n 网格，并在 p 指定的位置创建坐标区。MATLAB® 按行号对子图位置进行编号。第一个子图是第一行的第一列，第二个子图是第一行的第二列，依此类推。如果指定的位置已存在坐标区，则此命令会将该坐标区设为当前坐标区。

% plot(X,Y,LineSpec) 设置线型、标记符号和颜色

```
figure
subplot(3, 5, 1);
hold on
for run = 1:178
    plot(run,wine_label(run), '*');
end
xlabel('Sample', 'FontSize', 10);
ylabel('Label', 'FontSize', 10);
title('class', 'FontSize', 10);
for run = 2:14
    subplot(3, 5, run);
    hold on;
    str = ['attrib ', num2str(run-1)];
    for i = 1:178
        plot(i, wine(i, run-1), '*');
    end
    xlabel('Sample', 'FontSize', 10);
    ylabel('Attribute Value', 'FontSize', 10);
    title(str, 'FontSize', 10);
end
```

```
%选择第一品种的1-30，第二品种的60-95 和第三品种的131-153
作为训练集
train_wine_data = [wine_data(1:30, :); wine_data(60:95, :);
wine_data(131:153, :)];
%拆分标签
train_wine_labels = [wine_label(1:30); wine_label(60:95);
wine_label(131:153)];
%选择第一品种的31-59，第二品种的96-130，第三品种的154-178
作为试验集
test_wine_data = [wine_data(31:59, :); wine_data(96:130, :);
wine_data(154:178, :)];
%拆分标签
test_wine_labels = [wine_label(31:59); wine_label(96:130);
wine_label(154:178)];









%缩放（归一化）
```

```
[mtrain,ntrain] = size(train_wine_data);
[mtest,ntest] = size(test_wine_data);
% [sz1,...,szN] = size(A)  分别返回 A 的每个维度的长度
dataset = [train_wine_data; test_wine_data];
[dataset_scale, ps] = mapminmax(dataset', 0, 1);
% [Y,PS] = mapminmax(X,YMIN,YMAX)，通过将每行的最小值和最大值归一化为[ YMIN，YMAX] 来处理矩阵，但是归一化要使用列，所以进行转置，返回 Y 是一个矩阵
dataset_scale = dataset_scale';
train_wine_data = dataset_scale(1:mtrain, :);
test_wine_data = dataset_scale((mtrain+1):(mtrain+mtest), :);
%SVM 训练
model = svmtrain(train_wine_labels, train_wine_data, '-c 2 -g 1');
%预测
[predict_label,accuracy,decision_values]=svmpredict(test_wine_labels,
test_wine_data, model);




% 画出预测的和真实的对比图
```

```
figure;
hold on;
plot(test_wine_labels,'o');
plot(predict_label,'r*');
xlabel('Test Set Sample','FontSize',12);
ylabel('Label','FontSize',12);
legend('Real','Predicted');
title('Classifications Figure of Test Set','FontSize',12);
grid on;
```

4、 代码分析

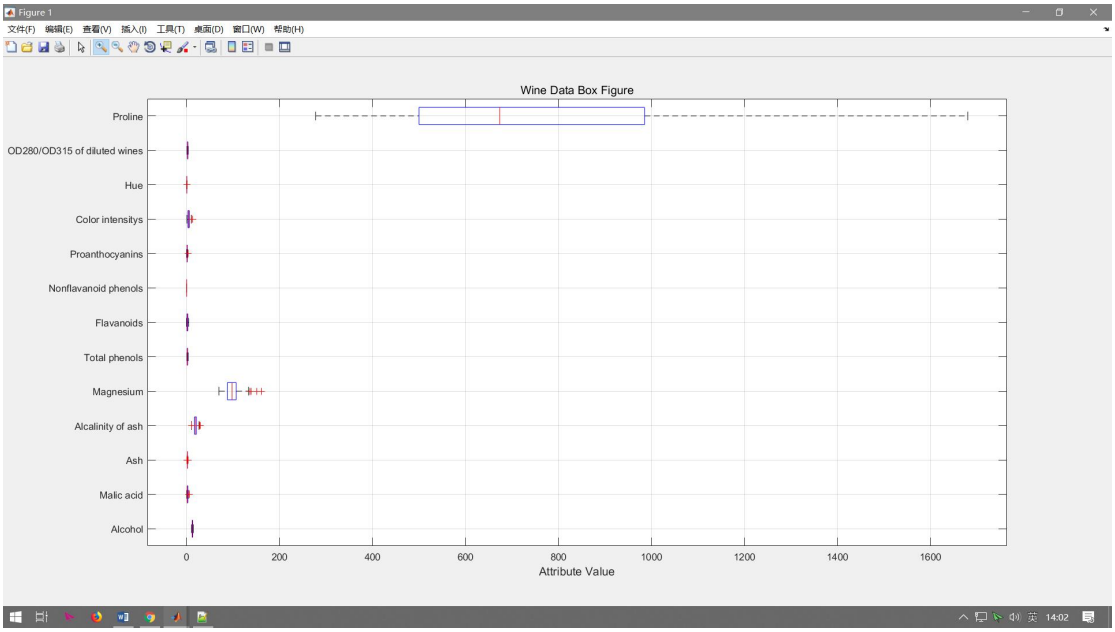
1) 加载葡萄酒数据，将其保存到 winedata

数据集的下载地址为：

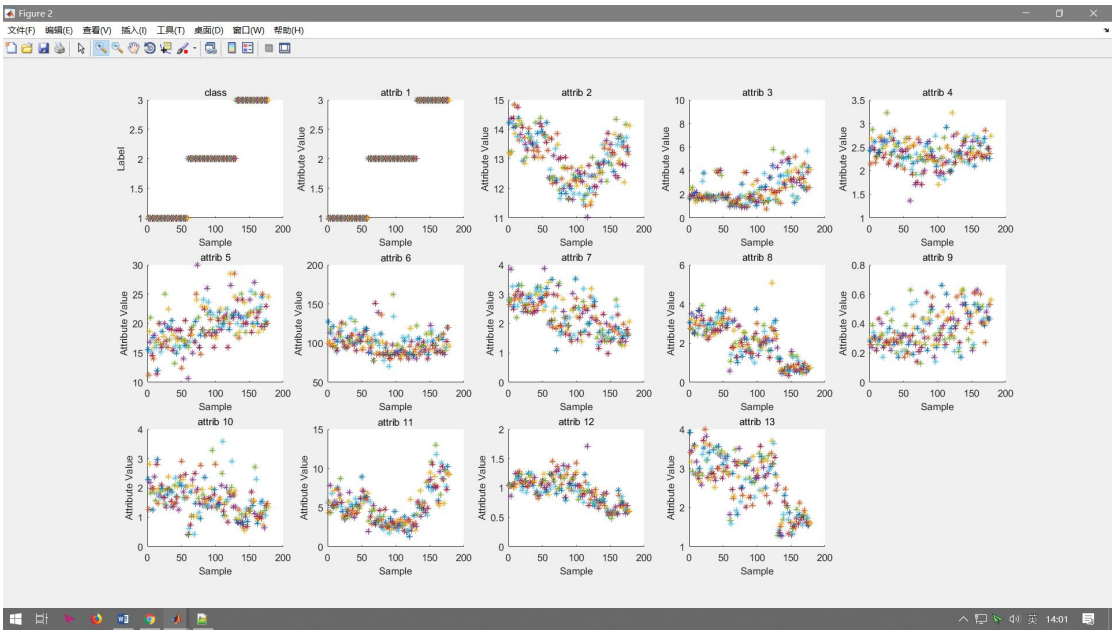
https://archive.ics.uci.edu/ml/datasets/Wine+Quality

数据文件只有一个，而我们的代码中需要将数据中的葡萄酒类别进行分开，用一个新的变量来储存 wine_label，而 wine_data 为葡萄酒的 13 个属性值的数据，categories 显示的 13 属性的具体名称。

2) 显示测试数据框图



3）显示测试数据的分形维数图



4）选择训练样本和测试样本

到了这一步，我有一个疑问，我们如果有一个数据集我们该如何去选择训练样本和测试样本的数量了,也就是该如何去划分数据集？于是我上网查询了一下

如果数据没有那么多，可以采用两种方法：一是交叉验证的方法，

具体可以分为简单交叉验证、S 折交叉验证和留一交叉验证，二是自助法，可以作为一个选项，一般不推荐使用

回到交叉验证，根据切分的方法不同，交叉验证分为下面三种：

第一种是**简单交叉验证**，所谓的简单，是和其他交叉验证方法相对而言的。首先，我们随机的将样本数据分为两部分（比如： 70%的训练集，30%的测试集），然后用训练集来训练模型，在测试集上验证模型及参数。接着，我们再把样本打乱，重新选择训练集和测试集，继续训练数据和检验模型。最后我们选择损失函数评估最优的模型和参数。

第二种是**S折交叉验证**（S-Folder Cross Validation）。和第一种方法不同，S折交叉验证会把样本数据随机的分成S份，每次随机的选择S-1份作为训练集，剩下的1份做测试集。当这一轮完成后，重新随机选择S-1份来训练数据。若干轮（小于S）之后，选择损失函数评估最优的模型和参数。

第三种是**留一交叉验证**（Leave-one-out Cross Validation），它是第二种情况的特例，此时S等于样本数N，这样对于N个样本，每次选择N-1个样本来训练数据，留一个样本来验证模型预测的好坏。此方法主要用于样本量非常少的情况，比如对于普通适中问题，N小于50时，我一般采用留一交叉验证。

通过反复的交叉验证，用损失函数来度量得到的模型的好坏，最终我们可以得到一个较好的模型。那这三种情况，到底我们应该选择哪一种方法呢？一句话总结，如果我们只是对数据做一个初步的模型建立，不是要做深入分析的话，简单交叉验证就可以了。否则就用S折交叉验证。在样本量少的时候，使用S折交叉验证的特例留一交叉验证。

此外还有一种比较特殊的交叉验证方式，也是用于样本量少的时候。叫做**自助法**(bootstrapping)。比如我们有m个样本（m较小），每次在这m个样本中随机采集一个样本，放入训练集，采样完后把样本放回。这样重复采集m次，我们得到m个样本组成的训练集。当然，这m个样本中很有可能有重复的样本数据。同时，用没有被采样到的样本做测试集。这样接着进行交叉验证。由于我们的训练集有重复数据，这会改变数据的分布，因而训练结果会有估计偏差，因此，此种方法不是很常用，除非数据量真的很少，比如小于20个。

5）**[0,1]缩放**

数据的归一化：那么我们为什么要对数据进行归一化？

The main advantage of scaling is to avoid attributes in greater numeric ranges dominating those in smaller numeric ranges. We have linearly scalable each attribute to the range [-1; +1] or [0; 1] 摘自 libsvm 上的 guide

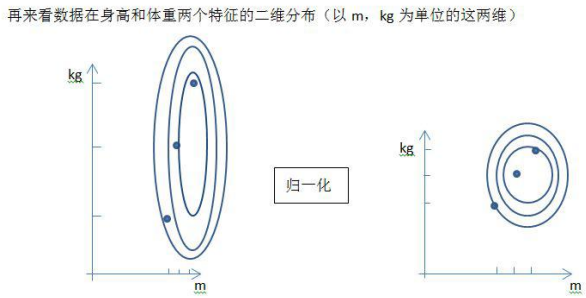
我的理解就是根据吴恩达预测房价的那个例子中，假如房价的特征值为两个，而且两个的差值特别大的话，我们画出的图形大约是一个很扁很扁的椭圆形，不利于梯度下降，归一化有利于求出结果，而我们这个例子中具有 13 个特征变量，为了使那个多维图形更有利于求解，因此需要数据归一化预处理，总结，归一化是为了增大求解速

度。

提示：通过搜索发现有两个概念比较容易混淆“归一化”和“标准化”，发现很多的 csdn 博主似乎也并没有很好的区分这个概念，通过在网上搜索我找到如下图，归一化和标准化都属于特征缩放：

归一化：

特点：①将训练集中某一列数值特征的值缩放到 0 和 1 之间



标准化：

特点：①将训练集中某一列数值特征的值缩放成均值为 0 方差为 1

6）SVM 训练和预测

SVM 训练和预测，分类有很多种不同的分类器，本次代码中采用的是 libsvm 分类器，这种需分类器的需要首先对数据进行归一化的处理，为了了解为什么要选择这种分类器我在网上进行查询得知有以下优点：可以解决非线性问题、可以很好处理高维数据、解决小样本机器学习问题等。同样我也查询一些其他分类器的优缺点进行对比。

（https://blog.csdn.net/July_sun/article/details/53088673）

在简单说下libsvm的安装，下载地址（https://www.csie.ntu.edu.tw/~cjlin/libsvm/）

网上有很多说需要编译，其实64位电脑不需要编译，只需要在matlab中添加路径就ok了，并且改名为libsvm应为matlab中已经自带了svm不可以同名。

model = libsvmtrain(train_wine_labels, train_wine_data, '-c 2 -g 1');

- `-s` svm类型：SVM设置类型（默认0，我就用默认的）
  - 0 – C-SVC;
  - 1 – v-SVC;
  - 2 – one-class SVM;
  - 3 – e-SVR;
  - 4 – v-SVR
- `-t` 核函数类型：核函数设置类型（默认2，据说2的效果也比较好，所以继续用2）
  - 0 – 线性核函数：`u'v`
  - 1 – 多项式核函数：`(r*u'v + coef0)^degree`
  - 2 – RBF(径向基)核函数：`exp(-r|u-v|^2)`
  - 3 – sigmoid核函数：`tanh(r*u'v + coef0)`
- `-h` shrinking：是否使用启发式，0或1（默认1）
  - 用0的话训练会快一些，
- `-d` degree：核函数中的degree设置（针对多项式核函数）（默认3）
- `-g` r(gamma)：核函数中的gamma函数设置（针对多项式/rbf/sigmoid核函数）（默认1/k，k为总类别数）
- `-r` coef0：核函数中的coef0设置（针对多项式/sigmoid核函数）（（默认0）
- `-c` cost：设置C-SVC，e-SVR和v-SVR的参数（损失函数）（默认1）
- `-n` nu：设置v-SVC，一类SVM和v- SVR的参数（默认0.5）
- `-p` p：设置e-SVR中损失函数p的值（默认0.1）
- `-m` cachesize：设置cache内存大小，以MB为单位（默认40）
- `-e` eps：设置允许的终止判据（默认0.001）
- `-wi` weight：设置第几类的参数C为weight*C (C-SVC中的C)（默认1）
- `-v` n: n-fold交互检验模式，n为fold的个数，必须大于等于2

结果：

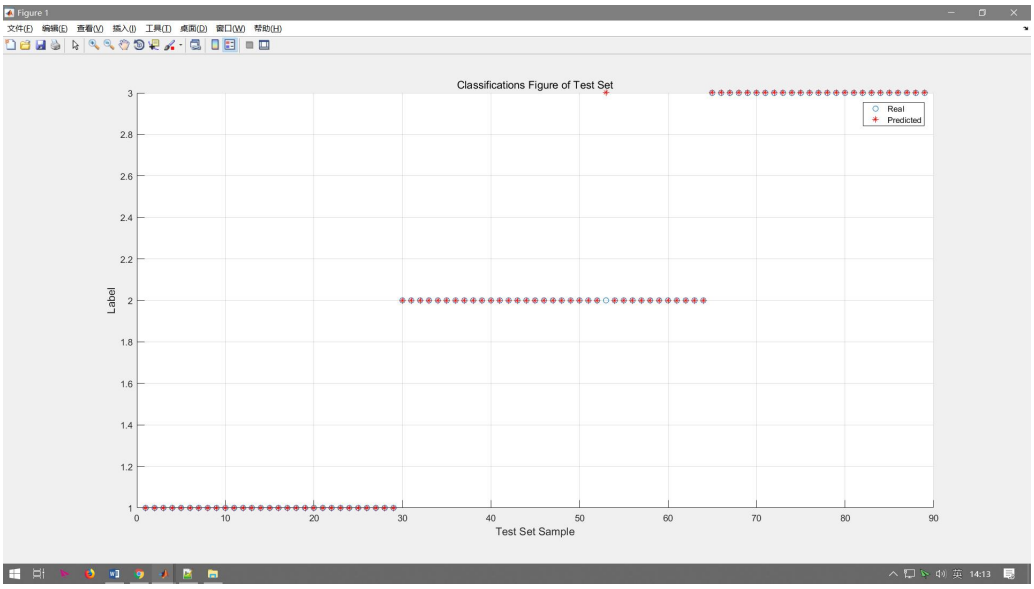
```
*
optimization finished, #iter = 47
nu = 0.178691
obj = -12.867480, rho = 0.692153
nSV = 21, nBSV = 7
*
optimization finished, #iter = 44
nu = 0.065614
obj = -3.476909, rho = 0.077960
nSV = 14, nBSV = 0
*
optimization finished, #iter = 46
nu = 0.209059
obj = -15.592942, rho = -0.187650
nSV = 20, nBSV = 7
Total nSV = 41
```

[predict_label,accuracy,decision_values]=libsvmpredict(test_wine_labels, test_wine_data, model);

结果：

```
Accuracy = 98.8764% (88/89) (classification)
```

8）



小结

此次我研究的是多类分类问题，在学习的过程中，遇到了许多的问题，确定的数据集的过程中我觉得研究葡萄酒这个项目还是挺有趣，想下载数据集发现中国 csdn 上面下载都是需要积分的，于是我只能选择去国外的网站 uci 中去找，结果很容易就找到了我的数据集，下一步就是寻找代码，也在 csdn 上面寻找，也发现一些问题，将代码复制下来后发现，代码出现了问题，我们首先需要下载 libsvm 而 matlab 中自带的 svm 不能够与其同名，代码总是会报错，所以我选择了国外的网站，也发现一些问题，csdn 中的代码似乎和国外的代码是相同的，并且是外国代码的缩略版。。。。。matlab 我看的是 matlab 官网的文档上面对于每一个函数的解释还是很清楚的，当然也可以选择 b’站上面一个台湾大学老师的视频。