《机器学习》读书笔记

1. 算法概述（算法名称及原理）

支持向量机（support vector machines）

svm本身是一个二类分类器，通过分离超平面对数据点进行分类，训练分离超平面。

原理：最大化支持向量到分离超平面的距离，用数学公式表达为其中支持向量为离分离超平面最近的点

三种数据集分布情况：

（1）完全线性可分（硬间隔）

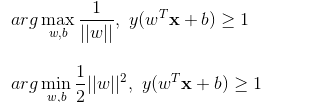
分离超平面：w^Tx + b。（训练中更新w和b，或alpha，使得分离超平面分类效果最佳）

某点到分离超平面的函数距离：

某点到分离超平面的几何距离：

点集到分离超平面的几何距离 => 距离超平面最近的点到其的距离：

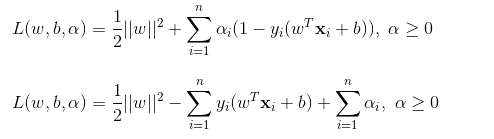
引入约束条件：假设所有点到超平面的距离都大于等于1，其中里分离超平面距离为1的点称为‘支持向量’（即等号成立时的点）。则问题转化为：



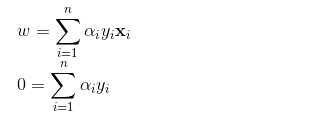
对偶问题

求解：以求得分离超平面，为凸二次规划问题。

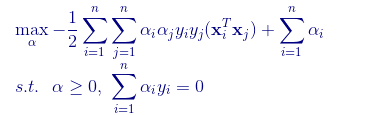
引入拉格朗日乘子alpha\_i >=0，则拉格朗日函数为：



分别对w，b求偏导，使导数为零：



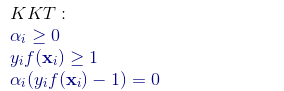
将w和b由alpha表示，并添加约束条件，最后使得问题转化为：



最后得到：



上述过程满足KKT条件：



上述KKT条件说明，当alpha为0时，对应的数据点不参与w的计算，而当alpha不为0时，，说明这些点到分离超平面距离为1，为支持向量。这说明：训练完成后，只需要保留支持向量的样本即可。

SMO算法

SMO：sequential minimal optimization 序列最小优化。优化求解alpha。

基本思路：固定alpha\_i之外的所有参数，然后求alpha\_i上的极值。由于约束条件：则固定alpha\_i之外的其他变量后，alpha\_i可以直接求解。

实验步骤：

step1：选取一对需要更新的alpha\_i，alpha\_j；

step2：固定alpha\_i，alpha\_j以外的参数，求解上述蓝色部分的对偶问题，获得更新后的alpha\_i，alpha\_j

alpha\_i的选取规则：第一个变量选取违背KKT条件程度最大的变量（KKT越违背，则更新alpha后目标函数的增值越大）。

alpha\_j的选取规则：第二个变量选取使目标数值增长最快的变量。

SMO中启发式方法：选取的两个变量alpha\_i，alpha\_j对应的两个样本之间的间隔最大，因为对差别大的两个变量进行更新，会带给目标函数值更大的变化。

当计算得到alpha后，因为对于所有支持向量, 其中S为所有支持向量的集合。因此可以将一个支持向量带入求得b值，或者求所有支持向量的b值然后取平均。

总结：

1. 初始化alpha为全1向量。

2. 第一轮两个alpha只能随机选择，因为都为1.

3. 以后几轮：更新alpha时，外循环遍历所有数据集或非边界alpha（不等于0或C的alpha），选取第一个alpha\_i。然后根据最大化误差步长来选取第二个alpha\_j, 根据目标函数及其约束条件，更新这两个alpha。

4. 重复上述步骤知道最大迭代步数，或所有alpha都不违背KKT条件。

（2）线性不可分

当数据线性不可分时，利用核函数将低维度下线性不可分的数据映射到高维度下线性可分的数据。即将数据从旧特征空间映射到新的特征空间。因为SVM中的运算可写成内积的形式（内积后得数为一个标量or数值），所以可以直接用核函数来代替原内积。

若数据完全线性不可分，则引入核函数，即对输入空间进行转换到特征空间：

，将原式中的x的内积转换成核函数。

较常用的核函数有

线性核函数：



多项式核函数：



高斯核（径向基核函数）：



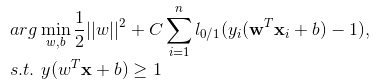
是用户自定义的用于确定到达率或者说函数值跌落到0的速度参数

高斯核是比较常用且效果较好的一个模型

（3）非完全线性可分（软间隔）

若数据非完全线性可分，则引入松弛变量,，允许少部分数据点处于分割面错误一侧。

优化目标：



其中l\_{0/1}为“0/1损失函数”：即当某点到分离超平面距离小于1时，考虑常数C，否则还按照线性可分的硬间隔来做。



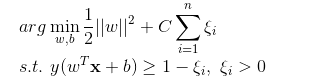
一般使用一些替代损失来代替上述“非凸，非连续的0/1损失”，一般选择有：

hinge损失：

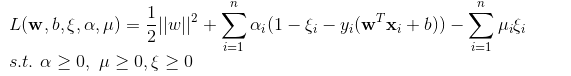
指数损失：

对数损失：

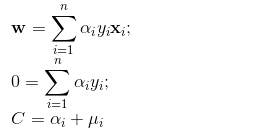
若引入松弛变量，目标优化函数为：



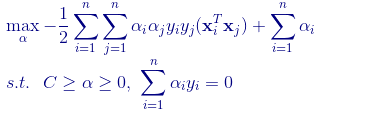
因此对应的拉格朗日函数：



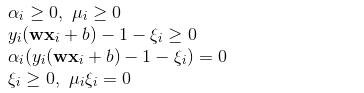
对w,b,xi求偏导为0，求得：



因此原目标函数的对偶问题为：

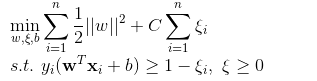


软间隔支持向量机的KKT条件为：



合页损失函数形式的目标函数

原目标函数：



新目标函数：



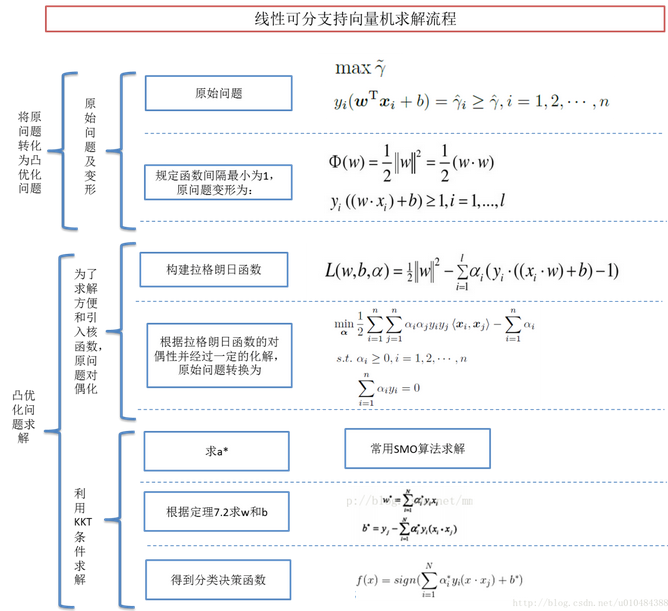
其中：

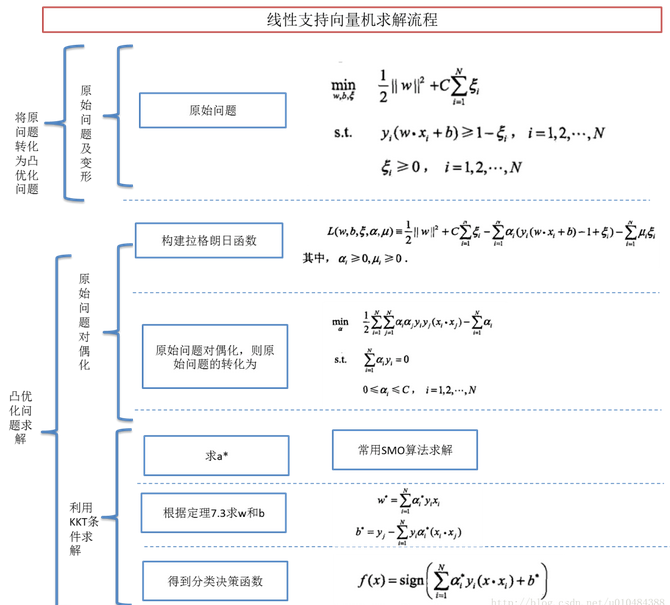


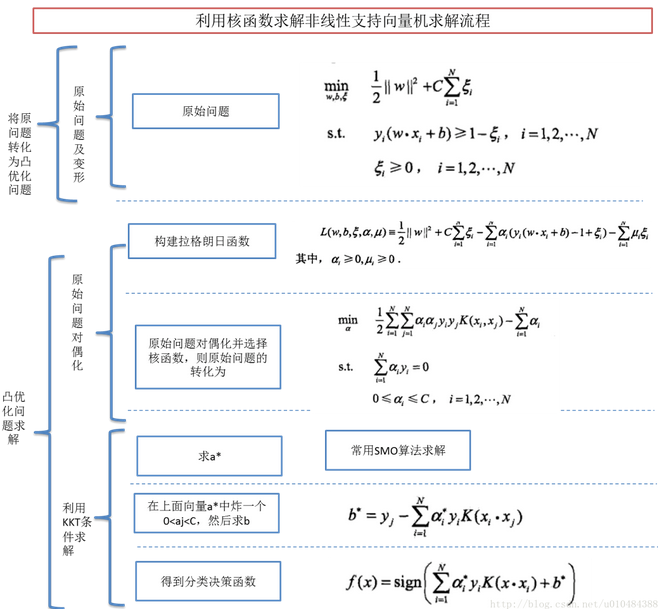
合页损失函数是指：



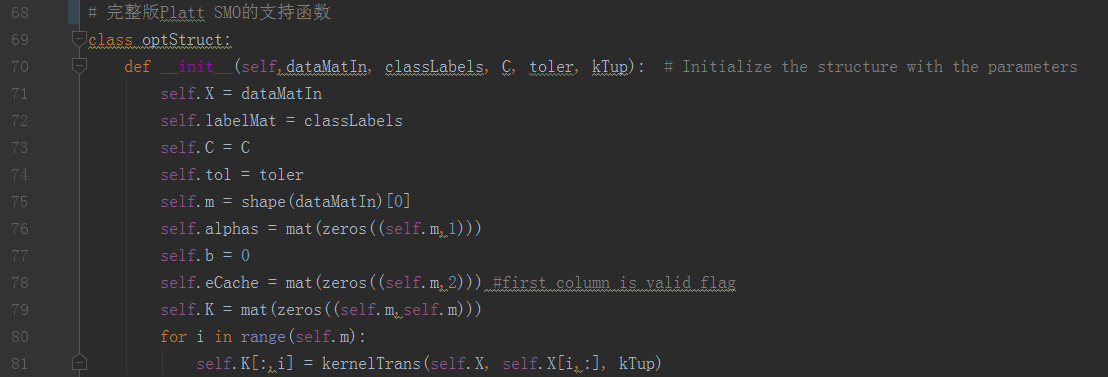
1. 算法设计（流程图及主要分段代码，附详细代码注释）

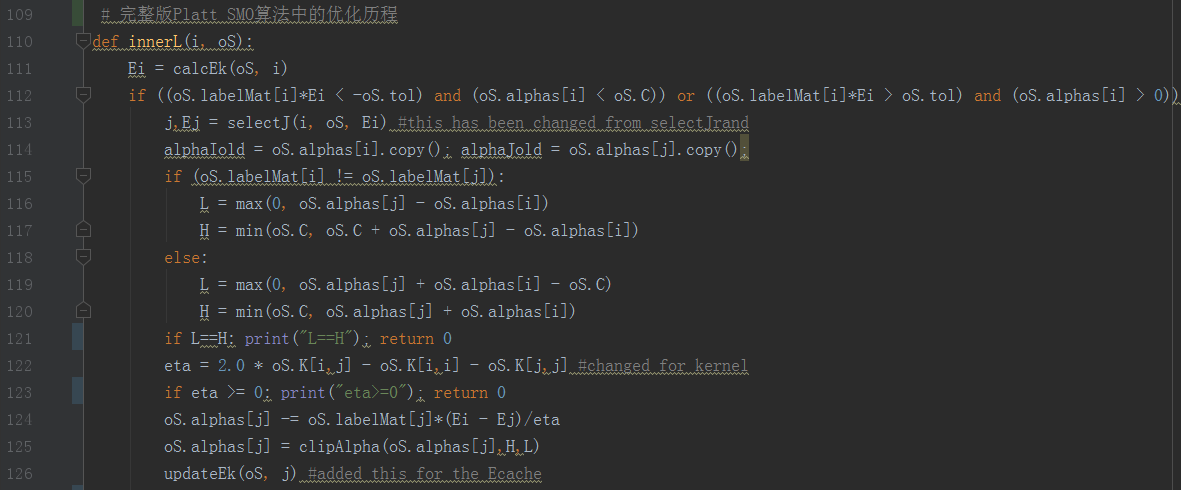


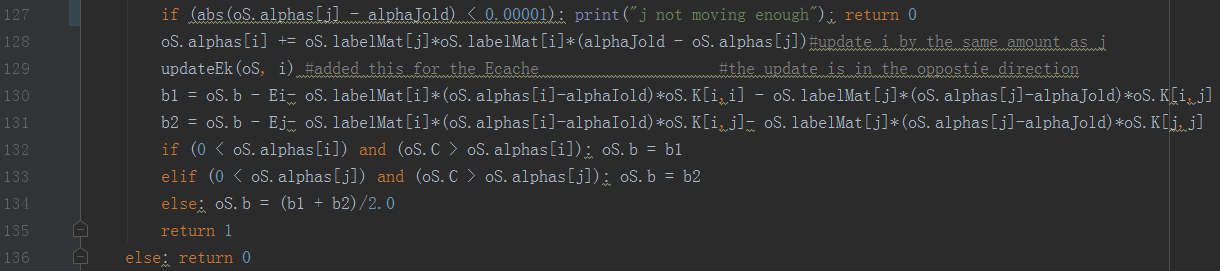


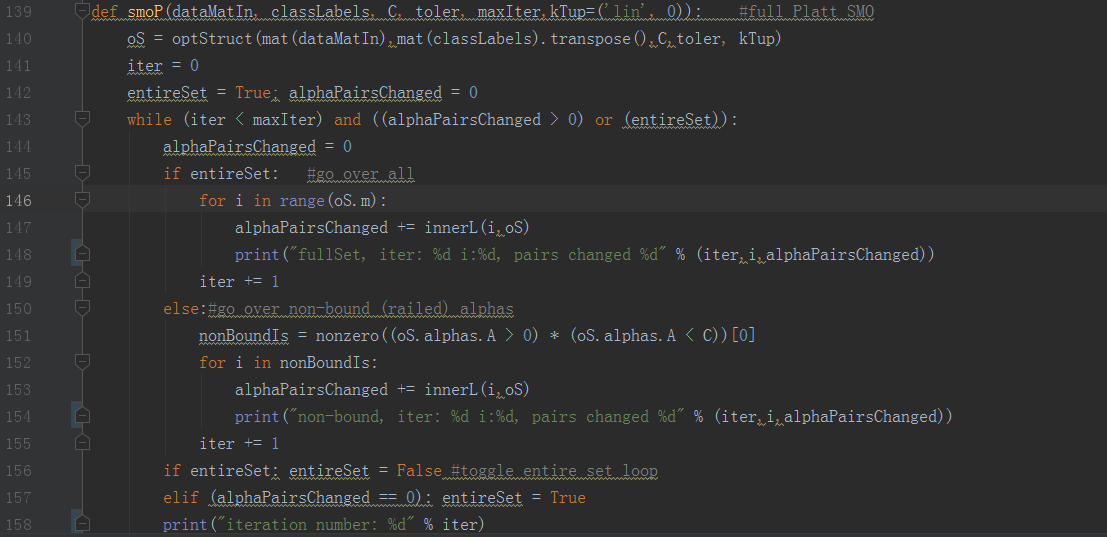


寻找最优的b，alpha代码部分（基于SMO函数实现）

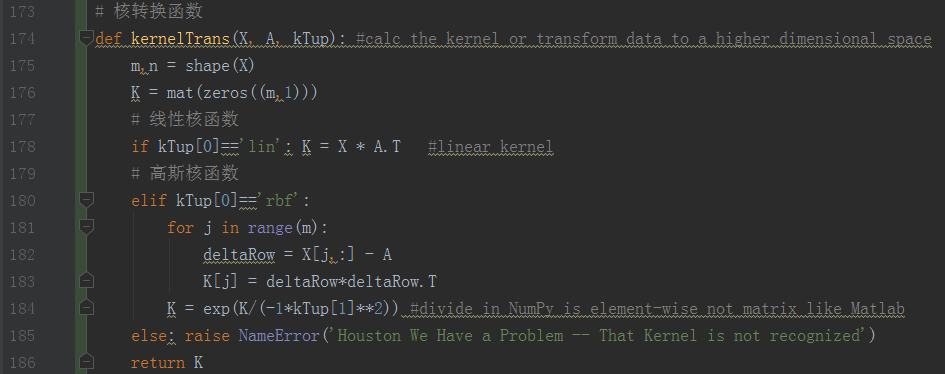




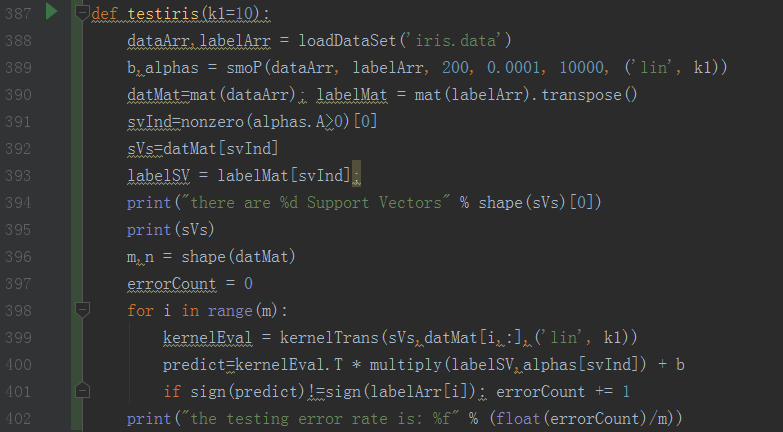




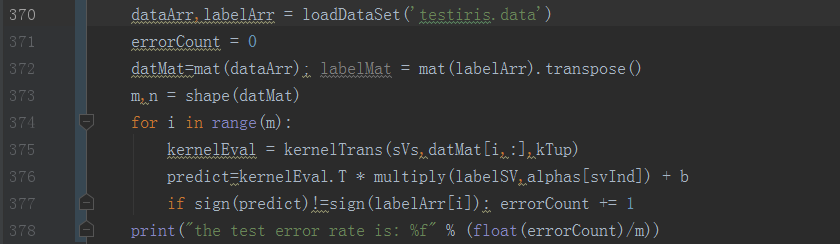
当数据集线性不可分时选用核函数的代码



训练数据集部分（以iris数据集为例）



测试集部分



1. 选用数据（数据集描述，包括来源，行数，列数，格式等）

数据集均来源于UCI

数据集一：

iris鸢尾花，行数：150，列数：4

列属性：

1. sepal length in cm
2. sepal width in cm
3. petal length in cm
4. petal width in cm

类别：  
Iris Setosa ，Iris Versicolour ，Iris Virginica

数据集二：

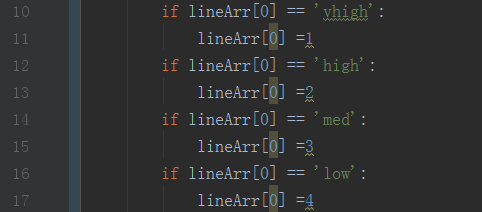
car，行数：1728，列数：6

列属性及取值：  
1) buying: vhigh, high, med, low.   
2) maint: vhigh, high, med, low.   
3) doors: 2, 3, 4, 5more.   
4) persons: 2, 4, more.   
5) lug\_boot: small, med, big.   
6) safety: low, med, high.

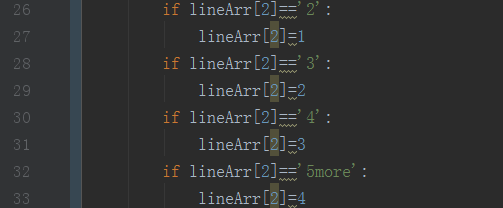
类别：

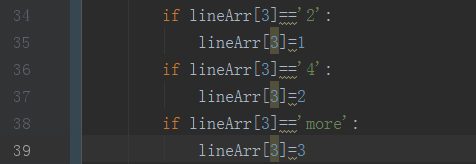
unacc, acc, good, vgood

遇到取值为字符串型属性时，将其取值数值化

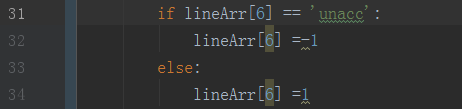


对于属性doors和persons的值，转化的代码如下





将类别数值化



数据集三：

wine，行数：178，列数：13

属性：

1. Alcohol
2. Malic acid
3. Ash
4. Alcalinity of ash
5. Magnesium
6. Total phenols
7. Flavanoids
8. Nonflavanoid phenols
9. Proanthocyanins
10. Color intensity
11. Hue
12. OD280/OD315 of diluted wines
13. Proline

类别：

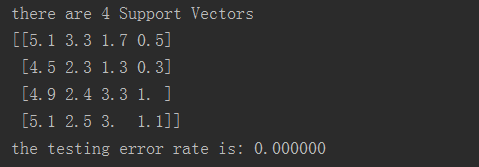
Alcohol 1，2，3

1. 评价方法（说明训练集和测试集分配方法及评价指标）

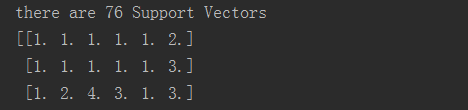
分配方法：十次十折交叉验证

评价指标：将数据打乱分成十份，取其中一份为测试集，剩下九份为训练集，进行十次试验之后再将数据集打乱，共打乱十次，以此类推进行一百次试验，每次都得到一个错误率，最后取平均数

1. 实验结果截图

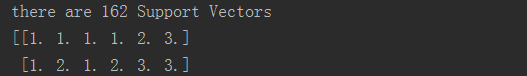
iris实验结果

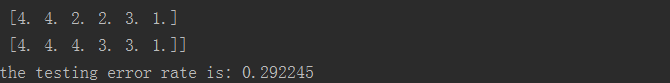
car实验结果（k1=30）



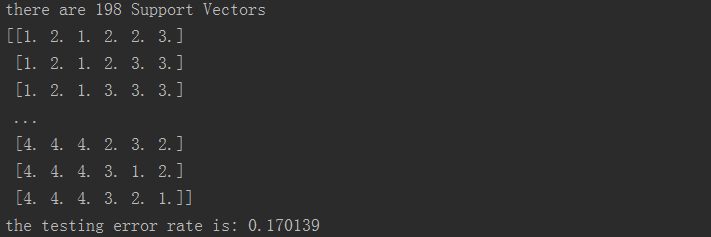


修改k1=20

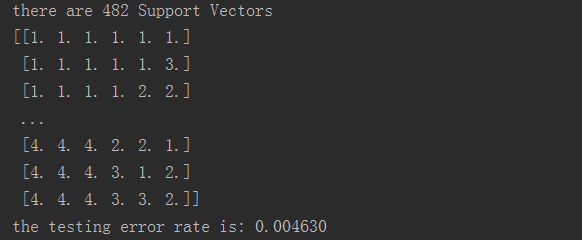




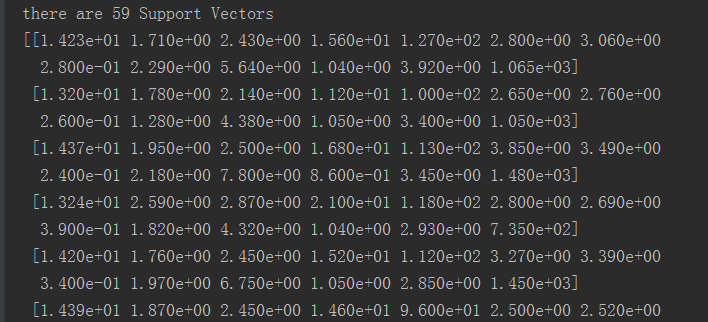
修改k1=10

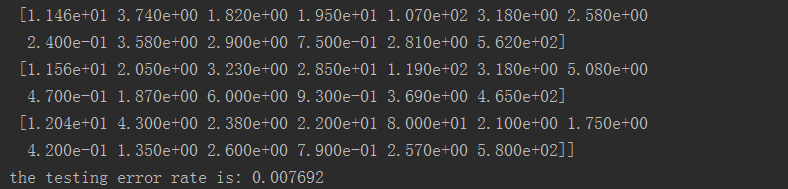


修改k1=1

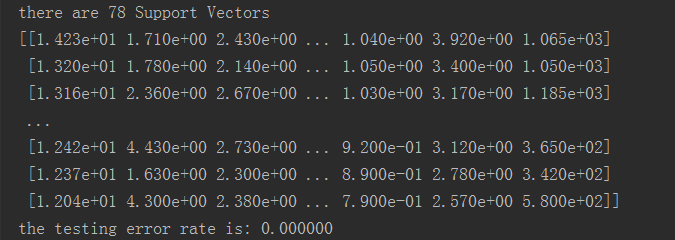


wine实验结果(k1取值30)





修改k1值为20



1. 实验结果分析及比较

关于支持向量的个数以及错误率的变化

通过修改径向基函数中的k1值会发现，每次运行时支持向量的个数都不相同，k1的值是用户自定义的用于确定到达率或者说函数值跌落到0的速度参数

实验证明支持向量的个数存在一个最优值，SVM的优点在于它能对数据进行高效分类，如果支持向量太少，就可能会得到一个很差的决策边界，如果支持向量太多，也就相当于每次都利用整个数据集进行分类（即K近邻），knn算法也的确是一个十分高效的算法

1. 遇到的问题及解决方法，实践心得

问题一：关于python版本不兼容的问题

将python2.x的代码转化成3.x的代码，方法如下：

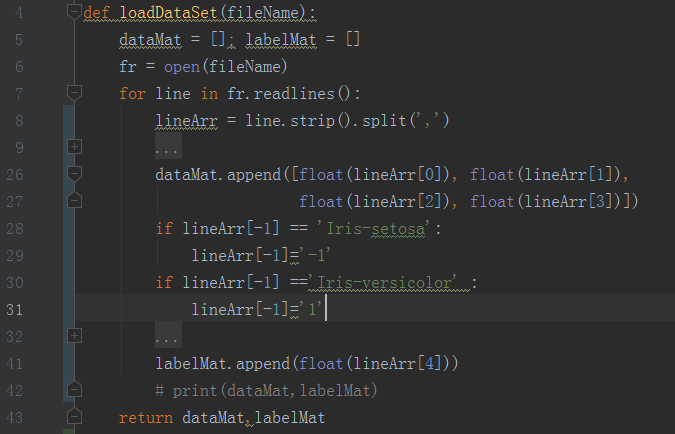
借助python安装目录下的工具包里面的2to3.py进行转化，具体操作步骤见以下链接

<https://blog.csdn.net/u012211419/article/details/51136232>

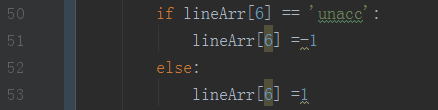
问题二：关于数据集的预处理问题

SVM本身是一个二类分类器，当分类目标不止两个时就需要对代码或者是数据集进行修改

解决办法：选取数据集中两个类别进行实验，同时将类别改为-1和1，修改代码如下



或者是将类别划分成两类，比如上面的数据集car，它有四个类别分别是unacc, acc, good, vgood,可以把它划分成两类，好和不好,修改代码如下



对于car数据集中doors和persons的预处理问题

因为doors只有四种取值2，3，4，5more，所以应该数值化的结果是1，2，3，4而不是被表面所蒙蔽取值2，3，4，5，同理，persons只有三种取值，所以它数值化的取值为1，2，3

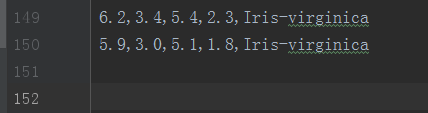
问题三：关于运行过程中出现的问题

1：IndexError:list index out of range

错误原因1；数组下标越界

错误原因2：出现了空字符串

仔细查看数据集发现是因为在数据集的末尾多加了几行，所以在读取数据的时候151和152行即为空字符串



解决办法：删除151和152行即可

建议：

对于使用核函数的问题，选择不同的核函数进行实验，选择最好的核函数

对于数据的预处理问题，可以使用降维的方式选择其中几个属性进行实验，对于降维的方法有成分分析法，前面学到的特征树进行降维