

kmeans算法实验报告

**机器学习实验报告**

课程名称 机器学习

专业班级 计算机少61

姓 名 熊兴宇

学 号 2140506094

目录

[一、 实验要求](#_Toc14415_WPSOffice_Level1) [3](#_Toc14415_WPSOffice_Level1)

[1. 实现SVM方法](#_Toc20798_WPSOffice_Level2) [3](#_Toc20798_WPSOffice_Level2)

[2. 在数据集上尝试SVM方法](#_Toc21451_WPSOffice_Level2) [3](#_Toc21451_WPSOffice_Level2)

[二、 实验原理](#_Toc20798_WPSOffice_Level1) [3](#_Toc20798_WPSOffice_Level1)

[三、 实验过程](#_Toc21451_WPSOffice_Level1) [6](#_Toc21451_WPSOffice_Level1)

[1. 读取iris数据集](#_Toc13747_WPSOffice_Level2) [6](#_Toc13747_WPSOffice_Level2)

[2. 生成SVM实例](#_Toc6913_WPSOffice_Level2) [6](#_Toc6913_WPSOffice_Level2)

[3. 初始化接入sklearn的SVM接口，打印结果](#_Toc13045_WPSOffice_Level2) [7](#_Toc13045_WPSOffice_Level2)

[四、 实验分析](#_Toc13747_WPSOffice_Level1) [8](#_Toc13747_WPSOffice_Level1)

[五、 实验总结](#_Toc6913_WPSOffice_Level1) [10](#_Toc6913_WPSOffice_Level1)

1. 实验要求
2. 实现SVM方法
3. 在数据集上尝试SVM方法
4. 实验原理

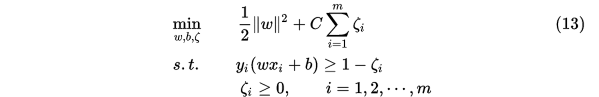
转自<https://www.cnblogs.com/xujianqing/p/7083605.html>

软间隔的svm需要优化的目标函数就是：

IMG_256

这里的C就是权衡一下坏点所占有的比重，当IMG_257时，那时候就强制要求所有的样例点都要满足约束条件，于是上面的问题就转化为一个硬间隔的svm。对式(12)进行优化，代表着两层含义：使得间隔尽量大，同时使得分类点的个数尽量少。

于是软间隔的SVM的原始问题可以描述成下式



式（13）是一个凸二次规划问题，在这里面可以证明得到IMG_259的解是唯一的，而IMG_260的解是不唯一的，IMG_261的解存在于一个区间中，具体证明见书《数据挖掘中的新方法——支持向量机》，当然为了求解式（13）的问题，我们同样要通过求解其对偶问题，从而得到该问题的解。  
同样引入拉格朗日函数

IMG_262

对偶问题是拉格朗日的极大极小问题，同样先要求关于 IMG_263的极小

IMG_264 IMG_265 IMG_266

可得到

IMG_267

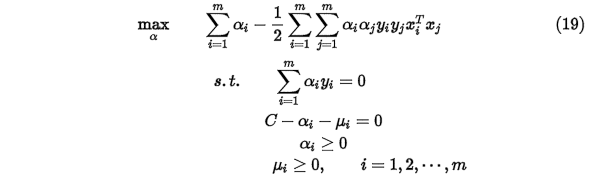
IMG_268

IMG_269

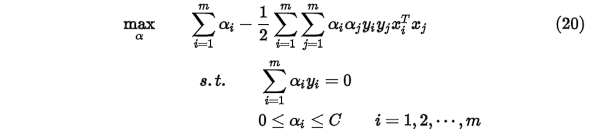
上面三个式子代入（14）可得到

IMG_270

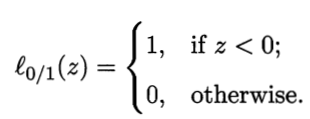
再对IMG_271求IMG_272的极大，即可得到对偶问题



对约束条件进行转换，消去IMG_274从而只留下IMG_275,即后面三个约束条件变为  
IMG_276  
对偶问题就转为：



损失函数是评价预测准确度的一种度量，预测:某个假设推断出的结果，损失函数是与假设密切相关的。  
常见的损失函数：

0-1损失函数：  
此时软间隔的优化函数为：

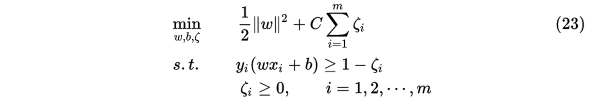
IMG_257

IMG_258不敏感损失函数  
hinge损失函数 ：IMG_259,此时的优化函数为：

IMG_260

指数损失函数：IMG_261  
对率损失函数：IMG_262

在这里证明一下如果在软间隔的SVM中，该原始问题采用的是式(23)的话，



那么其实他的损失函数就是用hinge函数，该函数的作用后面会提，先来证明一下：参考《统计学习方法》李航P114

证明：线性支持向量机原始最优化问题式（23）等价于最优化问题

IMG_264

证明：令IMG_265  
IMG_266 当IMG_267时  
IMG_268  
IMG_269 当IMG_270时  
IMG_271  
IMG_272  
IMG_273  
故满足式（23）的约束条件:  
IMG_274  
IMG_275 最优化问题IMG_276 可以写成：

IMG_277  
取IMG_278可以得到：  
IMG_279  
命题得证

1. 实验过程
2. 读取iris数据集

方法与K-means读取文件方法相同，保存在同一个类中

tests.py

1. **def** loadDataSet(fileName):
2. numFeat = len(open(fileName).readline().split(','))    # 计算有多少列
3. dataMat = []
4. labelMat = []
5. fr = open(fileName)
6. **for** line **in** fr.readlines():        #  遍历原始数据集每一行
7. **if**(line=='\n'):
8. **continue**
9. lineArr =[]
10. curLine = line.split(',')      # 是一列表类型,这里先采用iris数据集，以逗号分开
11. **for** i **in** range(numFeat-1):     # numFeat - 1的原因：因为原始数据的最后一列是类别，不是属性数据
12. lineArr.append(float(curLine[i]))  # 一个一个传进lineArr列表向量
13. dataMat.append(lineArr)     # 再传进dataMat列表向量
14. labelMat.append(str(curLine[-1]).strip('\n'))  # 写进标签列表，去除空格
15. **return** dataMat, labelMat
16. 生成SVM实例

tests.py

1. **def** test\_svm(self):
2. kernel\_class=['linear','rbf','poly','sigmoid']
3. choose\_kernel=2
4. svm=SVM(self.dataMat,self.labelMat,kernel\_class[choose\_kernel])

SVM核可以选取线性核，径向核，多项式核和sigmoid核

1. 初始化接入sklearn的SVM接口，打印结果

SVM.py

1. **import** numpy as np
2. **from** sklearn **import** svm
3. **from** sklearn.metrics **import** accuracy\_score
4. **from** sklearn.model\_selection **import** train\_test\_split
5. **import** pandas as pd
6. **import** matplotlib.pyplot as plt

9. SELECT\_X=0
10. SELECT\_Y=1
11. **class** SVM():
12. **def** \_\_init\_\_(self,args,label,kernel='linear'):
13. self.data=np.array(args)
14. x=self.data[:,SELECT\_X:SELECT\_Y+1]
15. y=pd.Categorical(label).codes
16. x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,random\_state=0,train\_size=0.8)
17. #C为惩罚因子，C越大，则会使模型更容易过拟合
18. svc=svm.SVC(C=0.1,kernel=kernel).fit(x\_train,y\_train.ravel())
19. y\_train\_pre=svc.predict(x\_train)
20. y\_test\_pre=svc.predict(x\_test)
21. **print**('Accuracy for train sets:',accuracy\_score(y\_train,y\_train\_pre))
22. **print**('Accuracy for test sets:',accuracy\_score(y\_test,y\_test\_pre))
24. fig=plt.figure('')
25. ax=fig.add\_subplot(311)
26. ax.scatter(x[:,0],x[:,1],c=y)
27. ax=fig.add\_subplot(312)
28. ax.scatter(x\_train[:,0],x\_train[:,1],c=y\_train\_pre)
29. ax=fig.add\_subplot(313)
30. ax.scatter(x\_test[:,0],x\_test[:,1],c=y\_test\_pre)
31. plt.show()

给定惩罚因子，调用sklearn的方法自动将训练集和测试集分开，并统计正确率，将分类结果画图

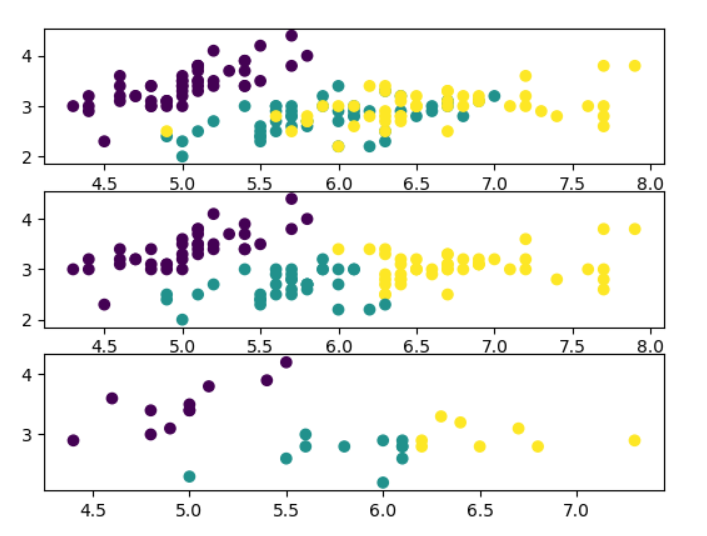
1. 实验分析

线性核 C=0.1 X=sepal length Y=sepal width

第一行为原数据及标签

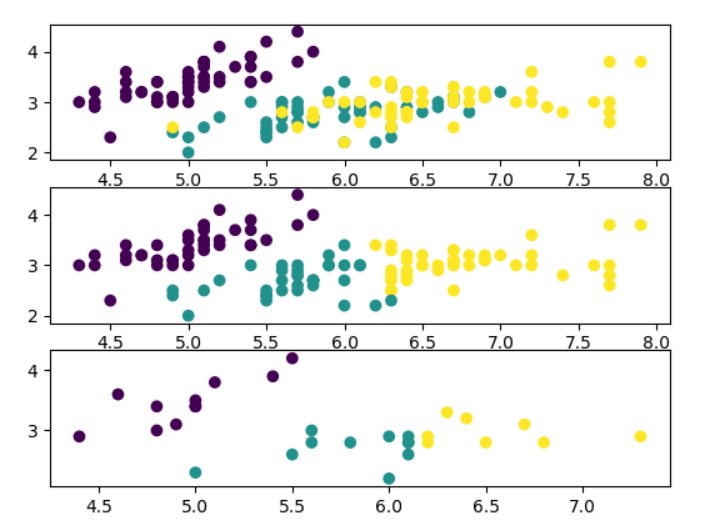
第二行为训练集及标签

第三行为测试集及标签





多项式核 其它项如上





可以看出，多项式核效果确实比线性核更好，正确率更高一点点。训练集与测试集不重叠，训练集的预测效果通常比测试集更好。

1. 实验总结

由于不需要自己实现直接调包，可以说是最简单的一个实验了。