**机器学习及应用实验报告**

**——WEKA API 的熟悉与应用**

实验编号 : 二

实验名称 : WEKA API的熟悉与应用

实验室 : 电光学院105

姓名 : 刘嘉洋

学号 : 1412620

班级 : 软件学院14级二班

实验日期 : 2017年3月7日

至 : 2017年3月14日

评分教师 实验报告成绩

评分日期 年 月 日

**一、实验目的**

* 在Eclipse中进行weka.jar的配置
* 结合weka-src.jar及说明文档进行API的学习
* 回顾WEKA基本模块及应用
* 完成Instances,Filters,Classifier,Evaluation类的编程操作
* 对比分析常见分类器的表现结果
* 掌握便捷获取.arff格式文件的技巧

**二、实验环境**

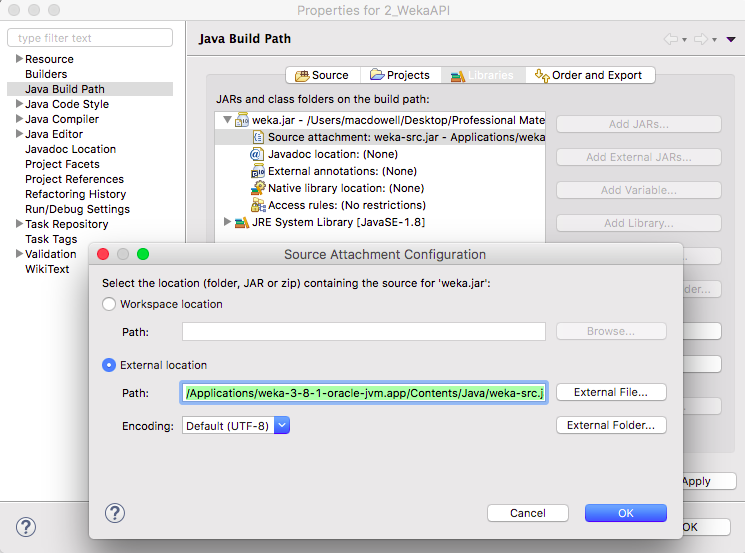
* macOS Sierra Version 10.12
* weka-3-8-1
* Eclipse Mars Release 4.5.0

**三、实验报告**

1. Instance类 • 数据读取与统计输出

step1: 在Eclipse中新建Java Project

step2: 导入weka.jar并配置源码路径指向weka-src.jar，以便API查阅



step3: 新建类，调用getFileInstance方法指向bank.arff数据集，获取数据实例

../../../../../Screen%20Shot%202017-03-07%20at%2011.49.00%20PM.png

step4: [重要] 调用实例对象的setClassIndex方法，将最后一列设置为类别

../../../../../Screen%20Shot%202017-03-07%20at%2011.49.18%20PM.png

step5: 调用对应方法，分别获取数据集样例数、属性数、类别数，并打印输出

../../../../../Screen%20Shot%202017-03-07%20at%2011.52.39%20PM.png

step6: InstanceAssign类控制台运行结果

../../../../../Screen%20Shot%202017-03-07%20at%2011.54.06%20PM.png

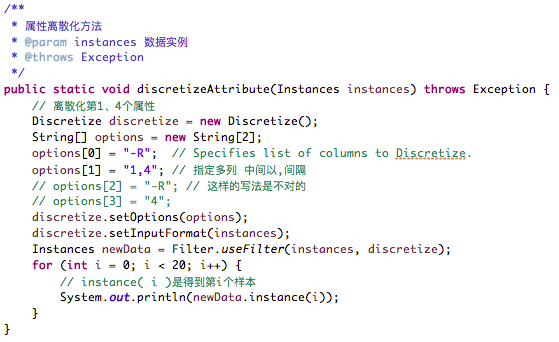
2. Filters类 • 属性操作

2.1 Discretize类 • 离散化

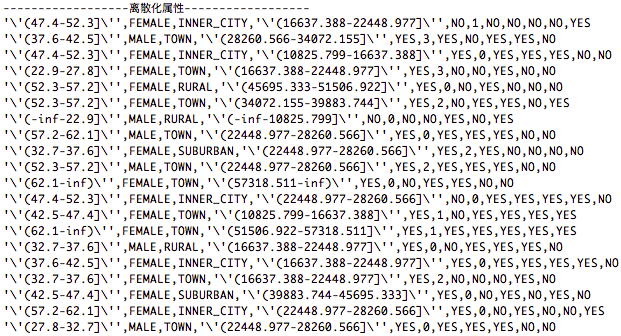
step1: 新建类，重复上述操作完成数据实例初始化 (bank.arff)

step2: 自定义属性离散化方法discretizeAttribute

其中：option取”-R”表示对整列进行离散化，并在紧接着的option指定目标属性列号，列号从1开始，多列则标号间以’,’间隔



step3: main方法中调用discretizeAttribute，控制台运行结果



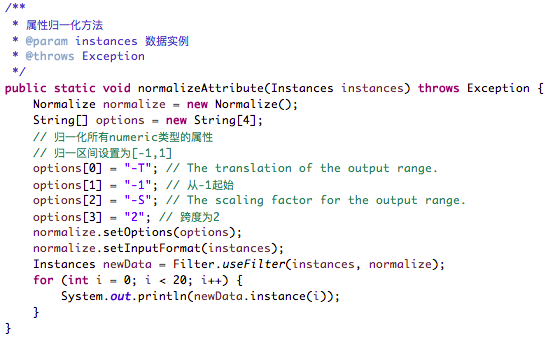
2.2 Normalize类 • 归一化

step4: 自定义属性归一化方法normalizeAttribute

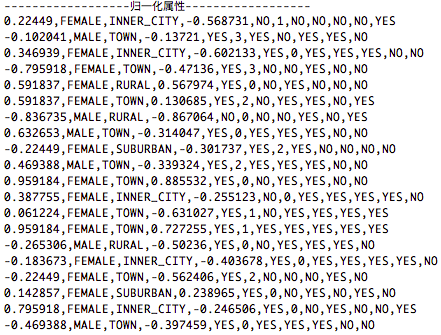
其中：option取”-T”：归一区间的起始坐标

取”-S”：归一区间的长度

[注] Normalize会自动归一化所有numeric类型的属性



step5: main方法中调用normalizeAttribute，控制台运行结果

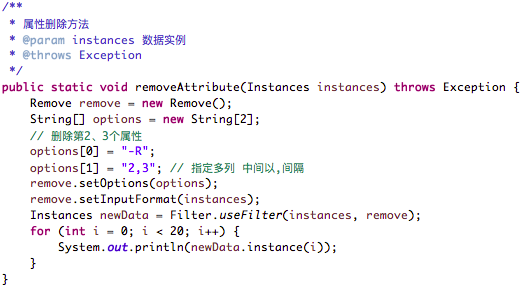


2.3 Remove类 • 删除

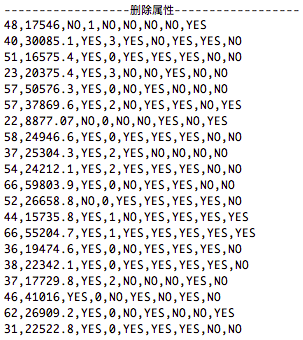
step6: 自定义属性删除方法removeAttribute

其中：option取”-R” 表示对整列进行删除，并在紧接着的option指定目标属性列号，列号从1开始，多列则标号间以’,’间隔

[注] Remove的删除只是将程序中处理的数据集进行过滤，并不会对.arff进行改变



step7: main方法中调用normalizeAttribute，控制台运行结果

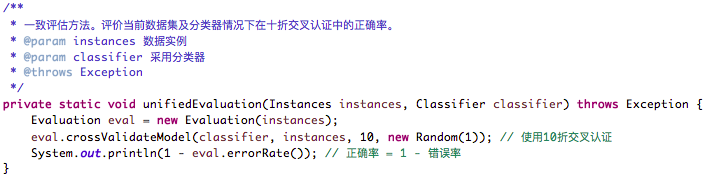


3. Classifier类 • 分类器的使用与评估

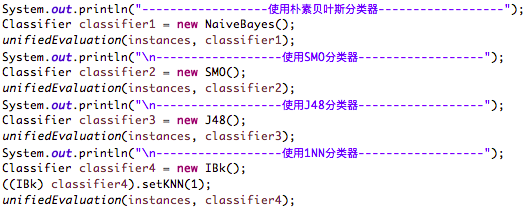
3.1 10折交叉验证的使用

step1: 新建类，重复上述操作完成数据实例初始化 (bank.arff)

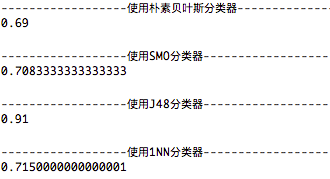
step2: 自定义一致评估方法unifiedEvaluation，提供给分类器对象统一的借口



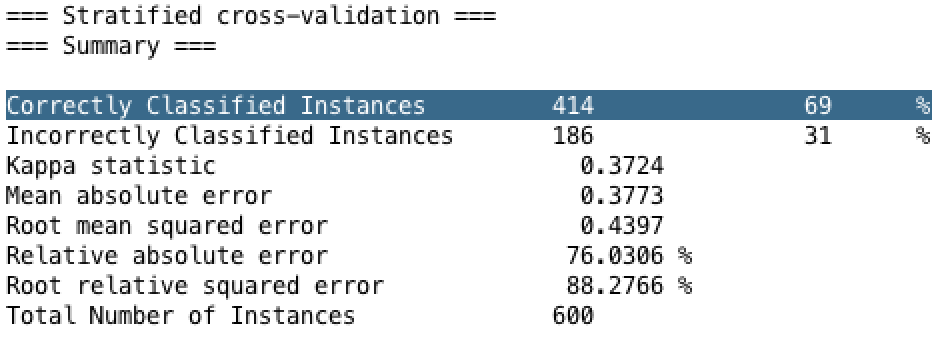
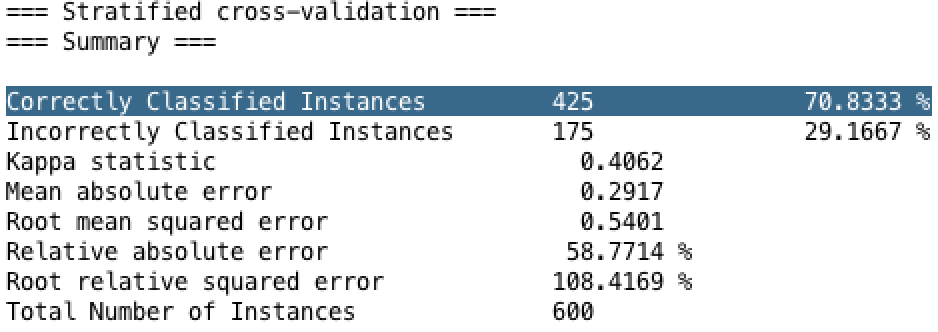
step3: 分别定义分类器并调用unifiedEvaluation方法



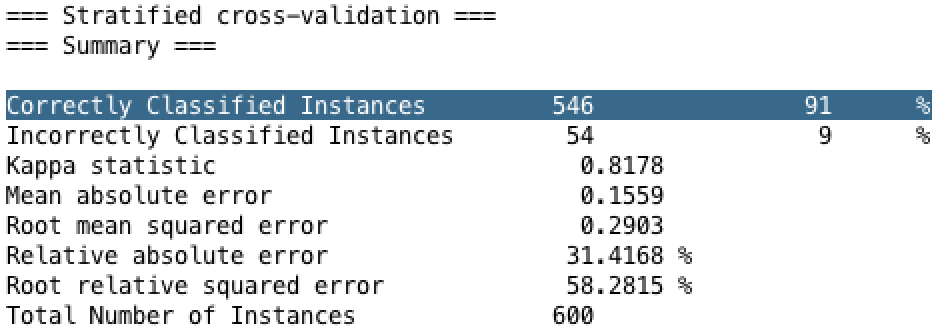
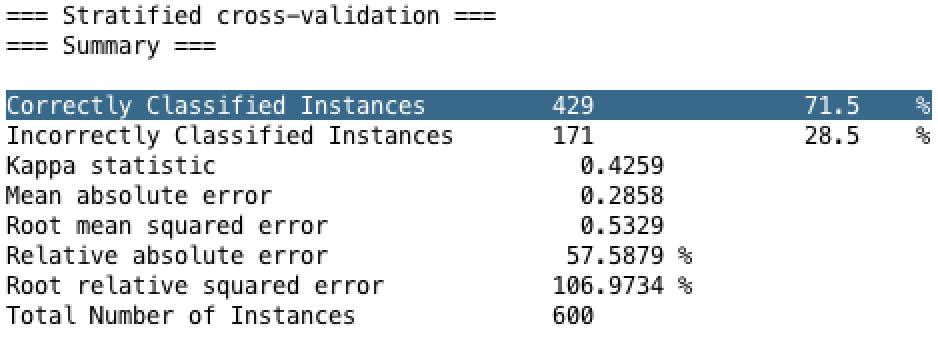
step4: ClassifierAssign1类控制台运行结果



step5: 在WEKA打开bank.arff数据集依次使用四种分类器，核对输出验证程序正确性

a. 朴素贝叶斯 b. SMO

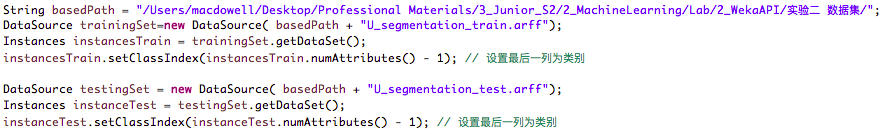
 

c. J48 d. 1NN

图1 WEKA中四种分类器对bank.arff的表现结果

3.2 训练集与测试集的使用

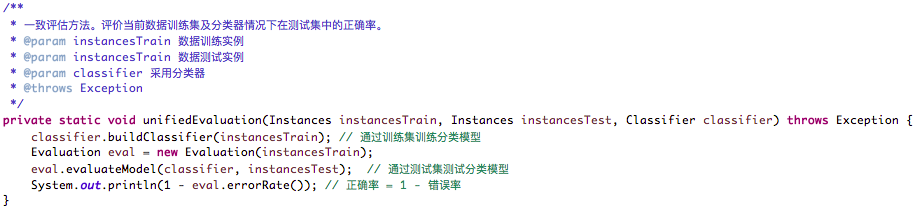
step1: 新建类，分别对U\_segmentation\_train.arff训练集与U\_segmentation\_tetst.arff测试集进行数据实例初始化以及类别属性设置



step2: 自定义一致评估方法unifiedEvaluation，提供给分类器对象统一的借口

其中：buildClassifier方法通过训练集训练出分类模型

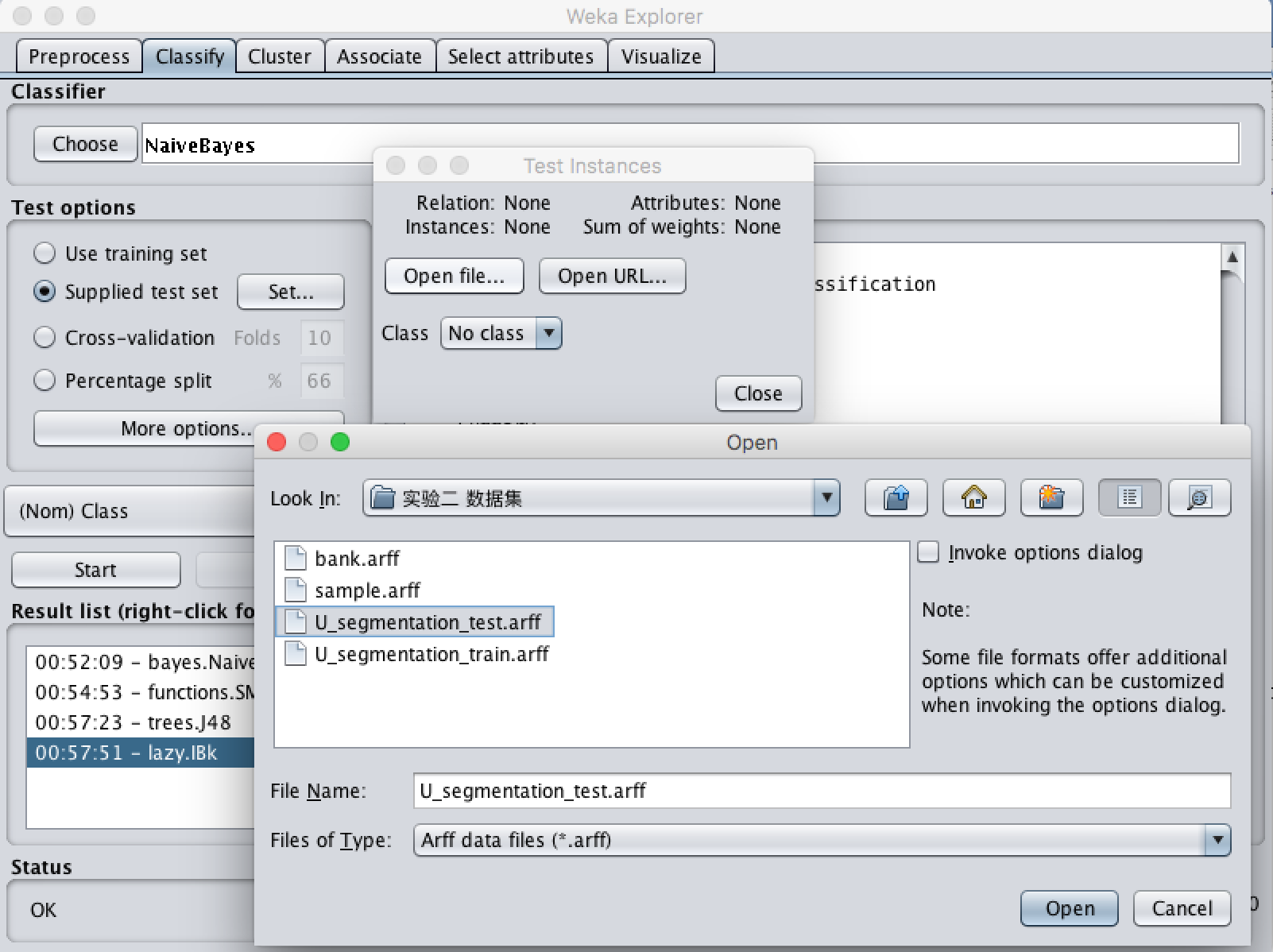
evaluateModel方法通过测试集测试所得分类模型

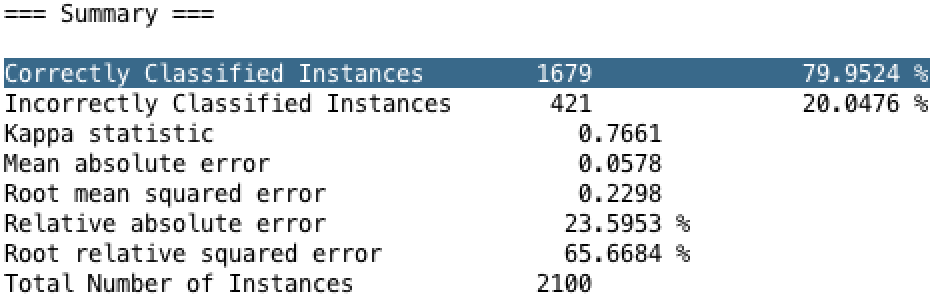
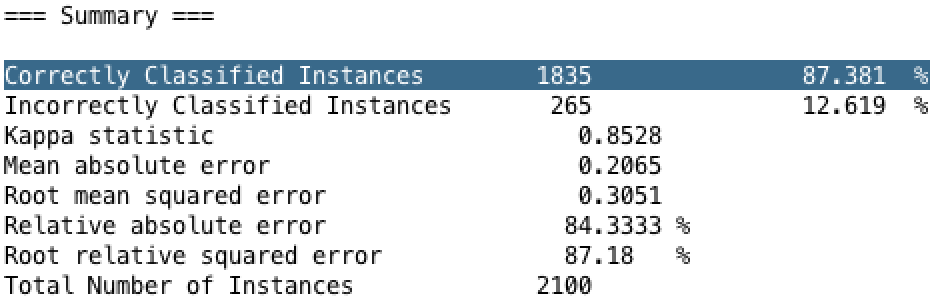


step3: ClassifierAssign2类控制台运行结果

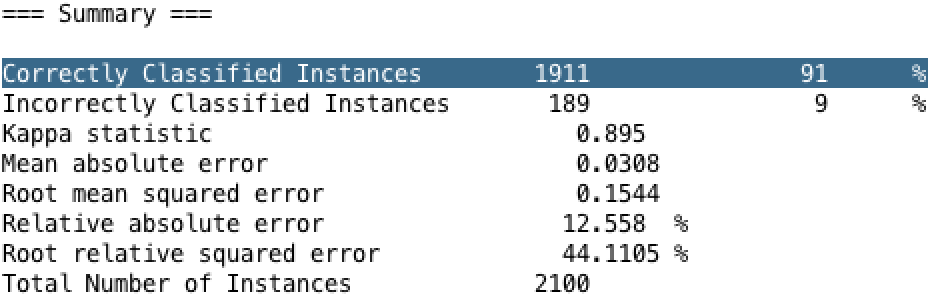
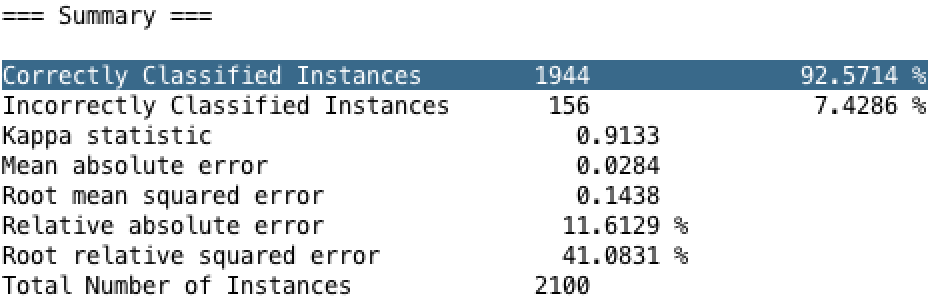


step4: 在WEKA输入同样的训练集和测试集使用四种分类器，核对输出验证程序正确性



a. 朴素贝叶斯 b. SMO

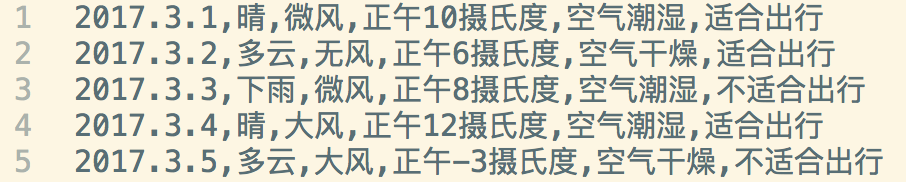
 

c. J48 d. 1NN

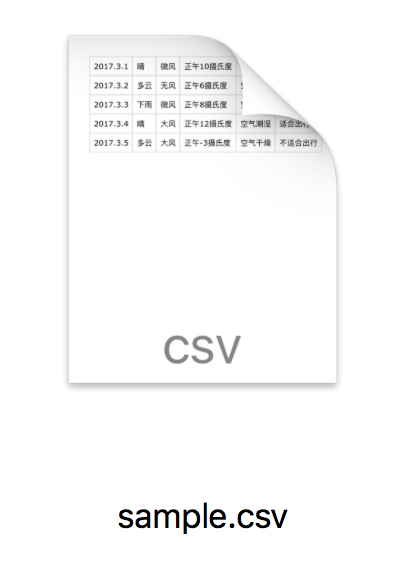
图2 WEKA中四种分类器对U\_segmentation训练集与测试集的表现结果

4. .txt->.cvs->.arff的转换

step1: 打开sample.txt，将辅助说明删去，并使属性数值间以英文’,’间隔



step2: 直接将文件后缀由.txt改为.cvs

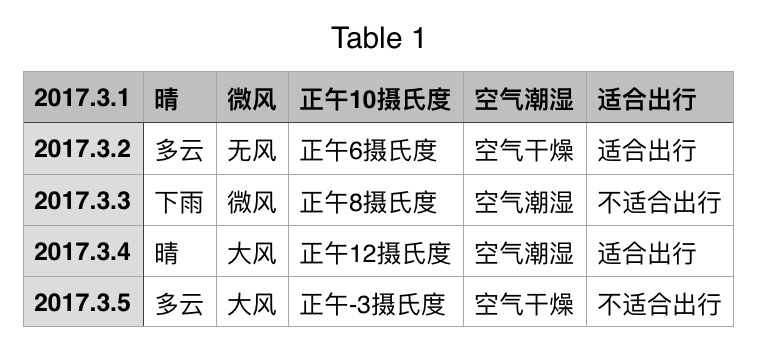


step3: 此时.cvs文件使用Excel或Numbers可直接打开

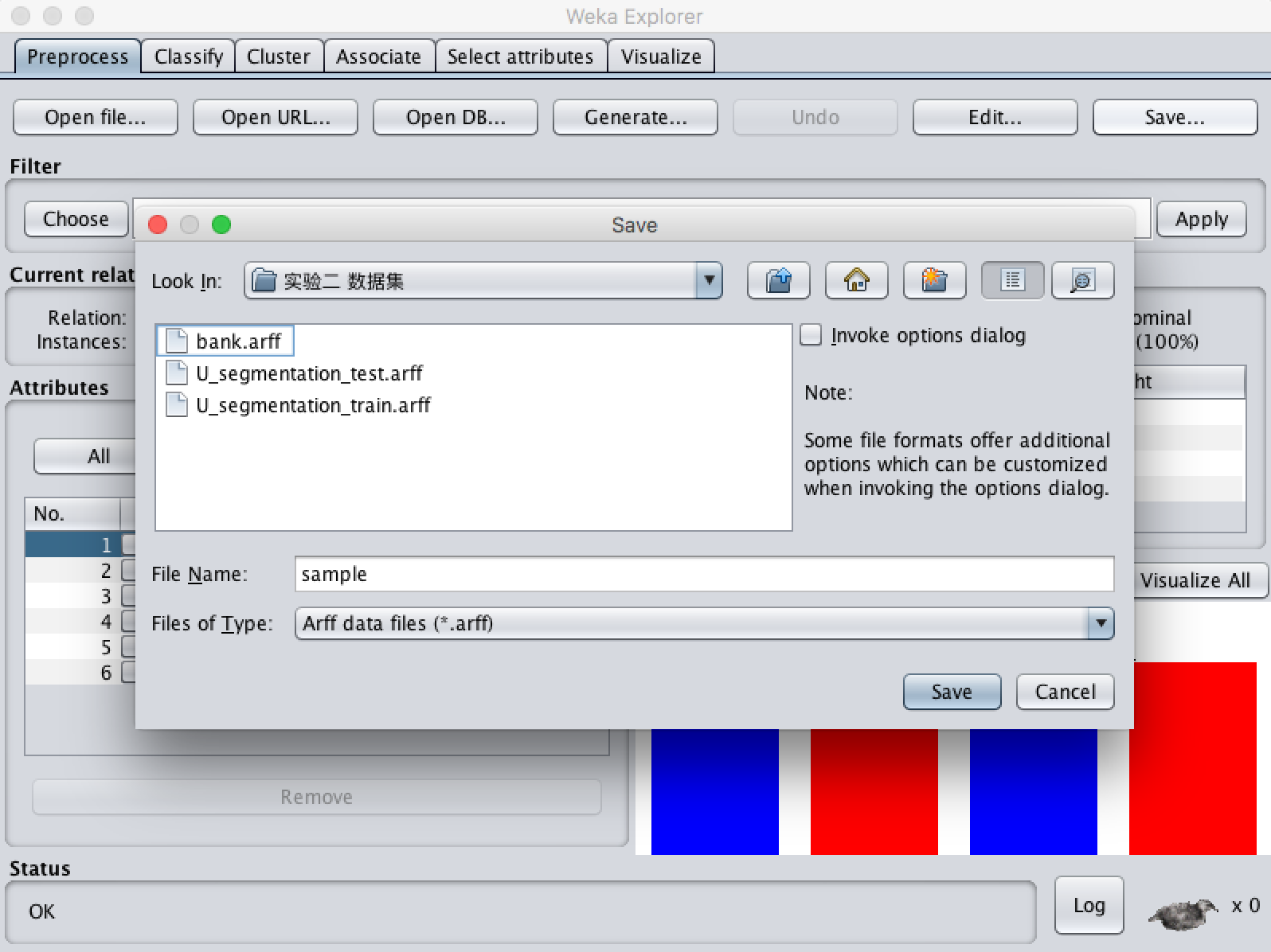
[注] Excel打开.cvs文件可能会出现中文乱码的情况

原因：.cvs文件默认为ANSI编码，如果.cvs文件的编码方式为utf-8、Unicode等方式时便会出现乱码

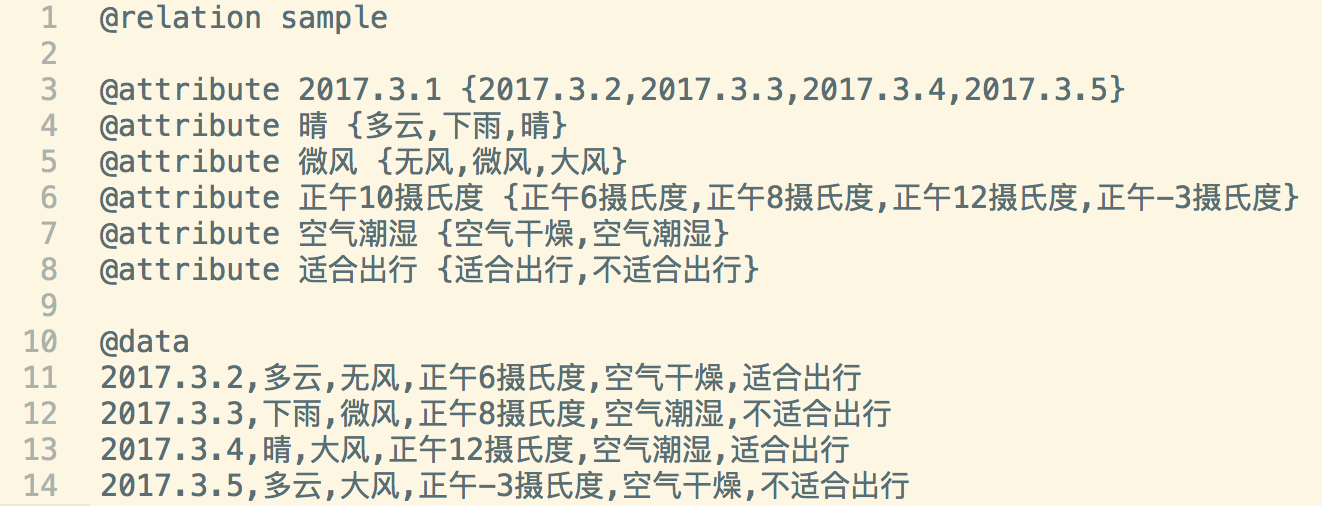
解决方法：用记事本等工具打开.cvs文件，将其另存为并选择ANSI编码方式即可



step4: 在WEKA中打开sample.cvs，单击”Save…”按钮，选择.arff保存格式



step5: 使用Sublime等工具打开生成的sample.arff文件，验证格式正确性

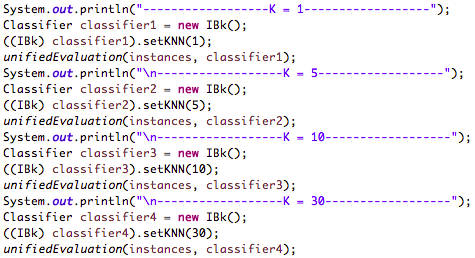


5. KNN分类器的比较与分析

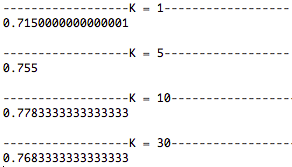
step1: 在Eclipse工程中新建类，重复上述操作完成数据实例初始化 (bank.arff)

step2: 同样采用在上述ClassifierAssign1类中自定义的一致评估方法unifiedEvaluation

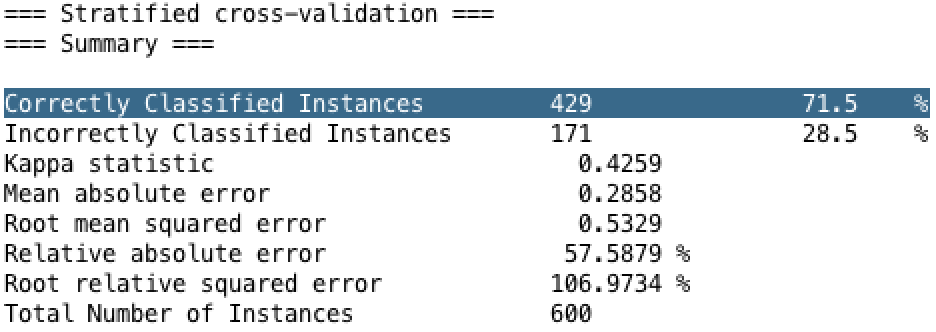
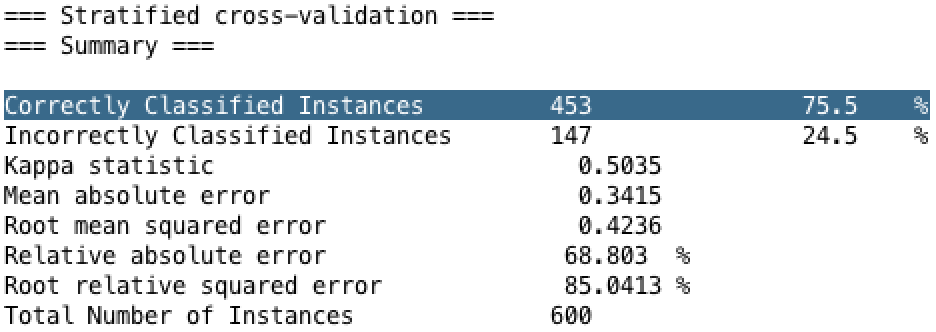
step3: 构建分类器，使K分别取值1、5、10、30



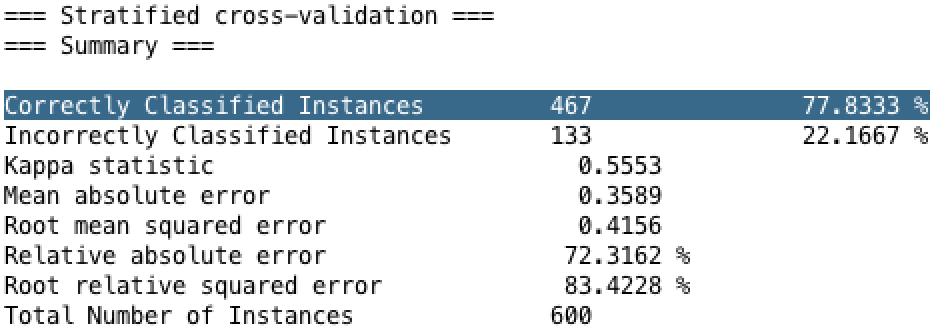
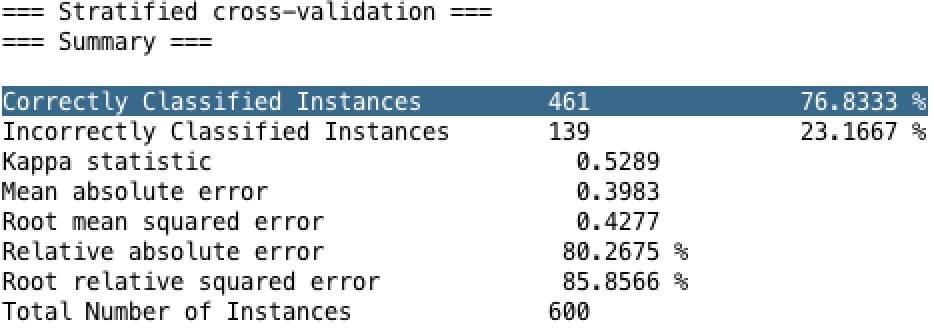
step4: KNNAssign类控制台运行结果



step5: 在WEKA打开bank.arff数据集使用IBk分类器将K依次取同样四种值，核对输出验证程序正确性

a. K = 1 b. K = 5

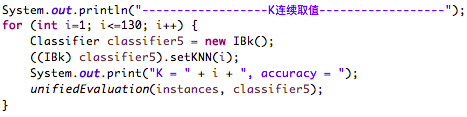
 

c. K = 10 d. K = 30

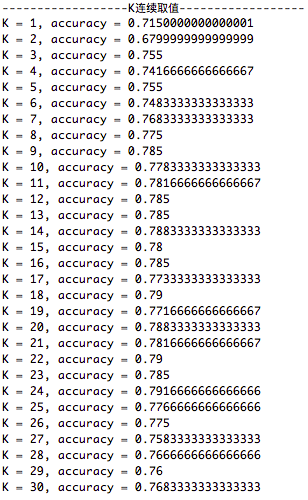
图3 WEKA中IBk分类器四种K取值时的表现结果

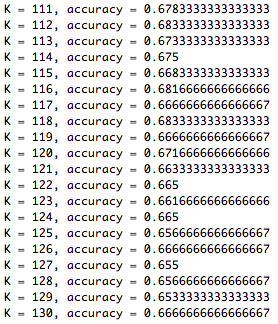
step6: 比较与分析

首先尝试在程序中将K以1为间隔在[1,130]连续取值找寻变化趋势



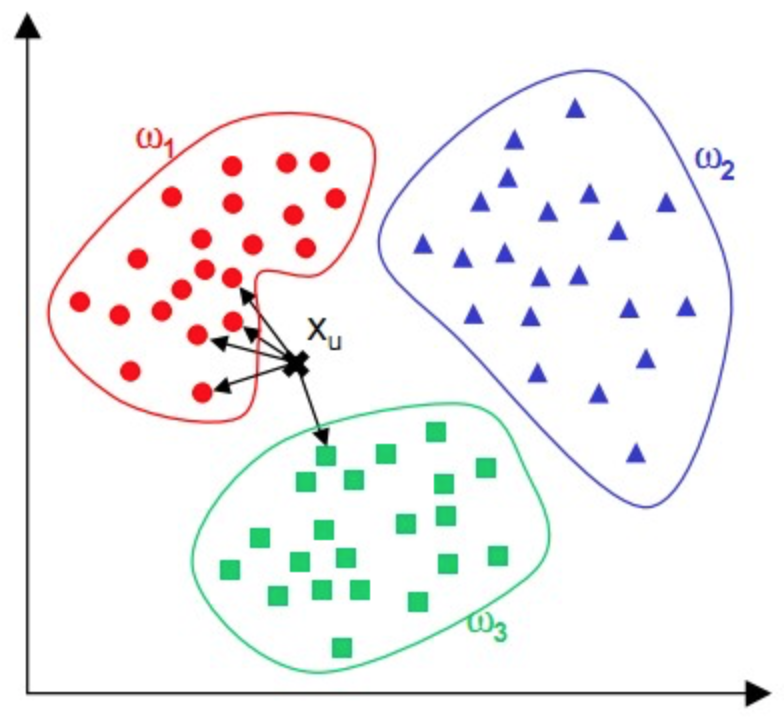
其在控制台运行结果 ([31,110]区间段没有贴出)





通过观察可以得知，其准确率并没有标准的单调性，在[0.665,0.79166]间波动，在K取[22,24]区间段达到峰值，并在其后逐渐减小 (后测试将K取值区间扩大至[1,230]可确认K值继续增大后准确率保持着波动下降的趋势)

回顾KNN算法



step1---初始化距离为最大值

step2---计算未知样本和每个训练样本的距离dist

step3---得到目前K个最临近样本中的最大距离maxdist

step4---如果dist小于maxdist，则将该训练样本作为K-最近邻样本

step5---重复步骤2、3、4，直到未知样本和所有训练样本的距离都算完

step6---统计K-最近邻样本中每个类标号出现的次数

step7---选择出现频率最大的类标号作为未知样本的类标号

个人理解

最优的K值是取决于数据集的，其取值一方面应尽量大以使得噪点不会干扰预测的准确率，同时也不应过大避免导致K值主导了KNN算法中的其他参数反而影响准确率。在有的参考资料上看到说是推荐的取值，但我觉得在数据集规模适宜的情况下最稳妥的方式仍然是通过遍历结合交叉验证找寻到使预测准确率最高的K值。

**四、写在最后**

感谢卫金茂老师以及助教学长学姐对本次实验的细致指导和悉心解答。

因初期学习机器学习知识，报告中出现的理解偏差也望指正。

刘嘉洋

NKCS, 1412620

[macdowellliu@163.com](mailto:macdowellliu@163.com)

2017.3.8