**---（1）经典算法KNN实现**

学生姓名： 学 号： 专业班级：

实验类型：□ 验证 ■ 综合 □ 设计 □ 创新 实验日期： 11.03 实验成绩：

**一、实验目的**

通过KNN算法了解消极学习方法，即不需要事先对训练数据建立分类模型的方法。

**二、实验内容**

根据KNN算法原理实现程序，请编译、运行并观察程序的输出，并分析实验结果，写出实验报告。

**三、实验要求**

1. 分析实验代码；

2. 分析运行结果；

3. 画出程序流程图。

**四、实验步骤**

1. 编写实验代码；

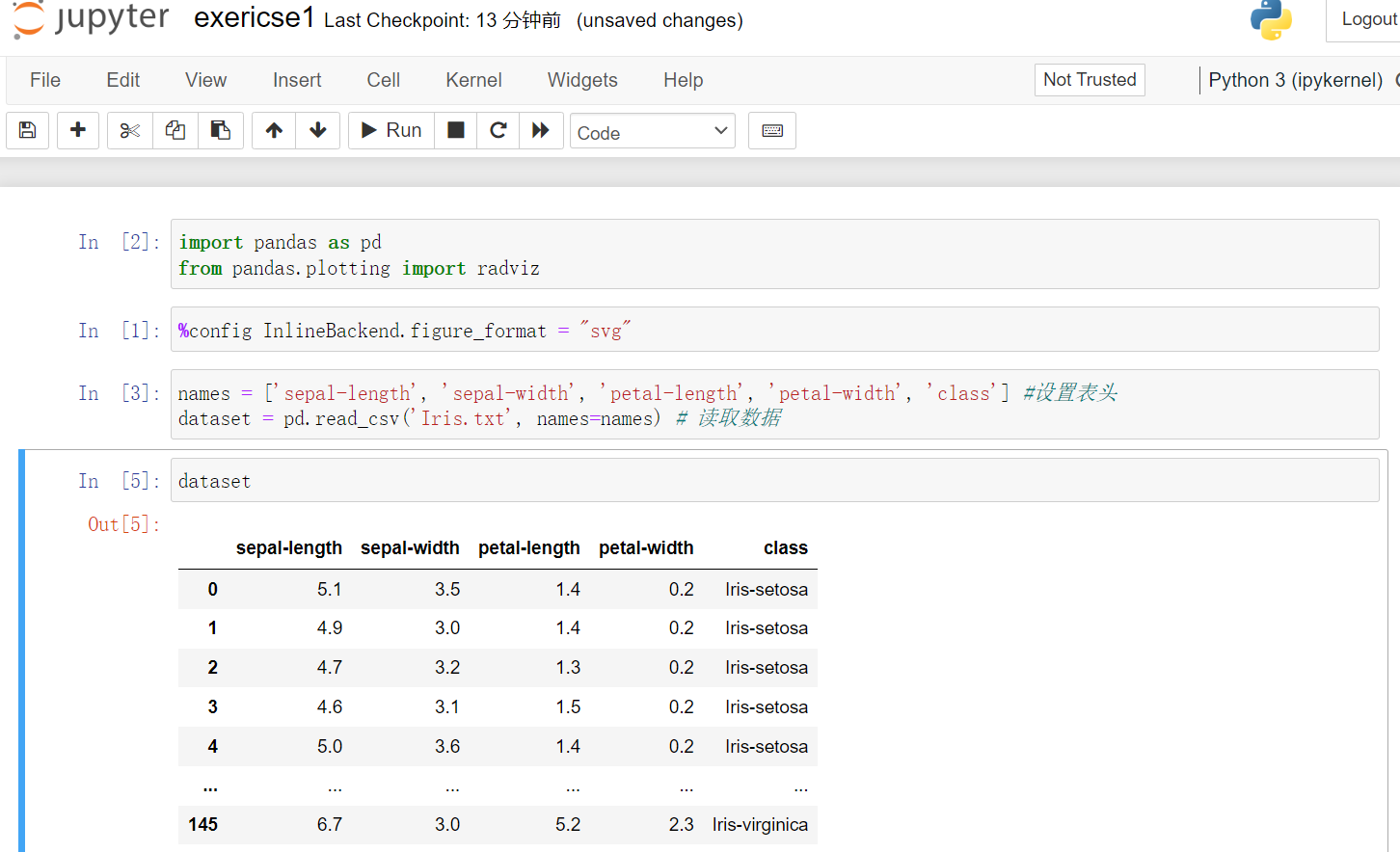
2. 编译、运行程序；

3. 观察程序的输出并分析实验结果。

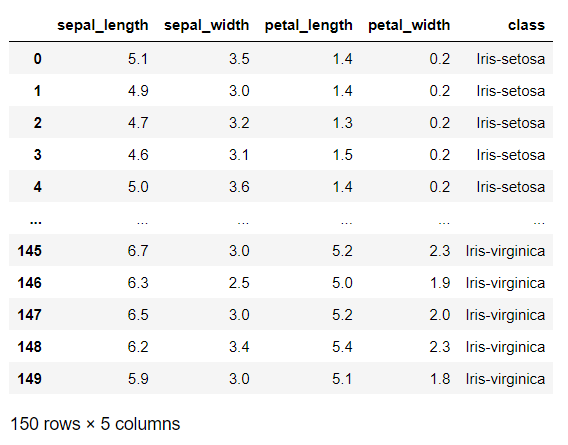
**五、实验数据及处理结果**

**1. 查看数据集**

使用Jupyter Notebook查看数据集。代码如下如下：



将输出结果展示完整，如下图。由此可知数据集共包含150个数据样本，每个样本共有花萼长度、花萼宽度、花瓣长度、花瓣宽度4个属性。本次实验正是根据这4个属性预测给定样本属于哪种类别。



1. **数据集可视化**

上述展示的表格仅是查看数据集，各个属性之间以及属性与分类之间的关系并没有直观展示出来。考虑到每个样本的属性为多维，所以此处选择绘制数据集的Radviz图进行可视化。

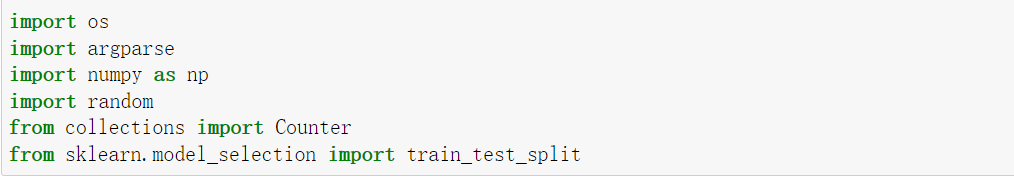
Radviz可视化原理是将一系列多维空间的点通过非线性方法映射到二维空间的可视化技术，是基于圆形平行坐标系的设计思想而提出的多维可视化方法。圆形的m条半径表示m维空间，使用坐标系中的一点代表多维信息对象，其实现原理参照物理学中物体受力平衡定理。绘制结果如下：

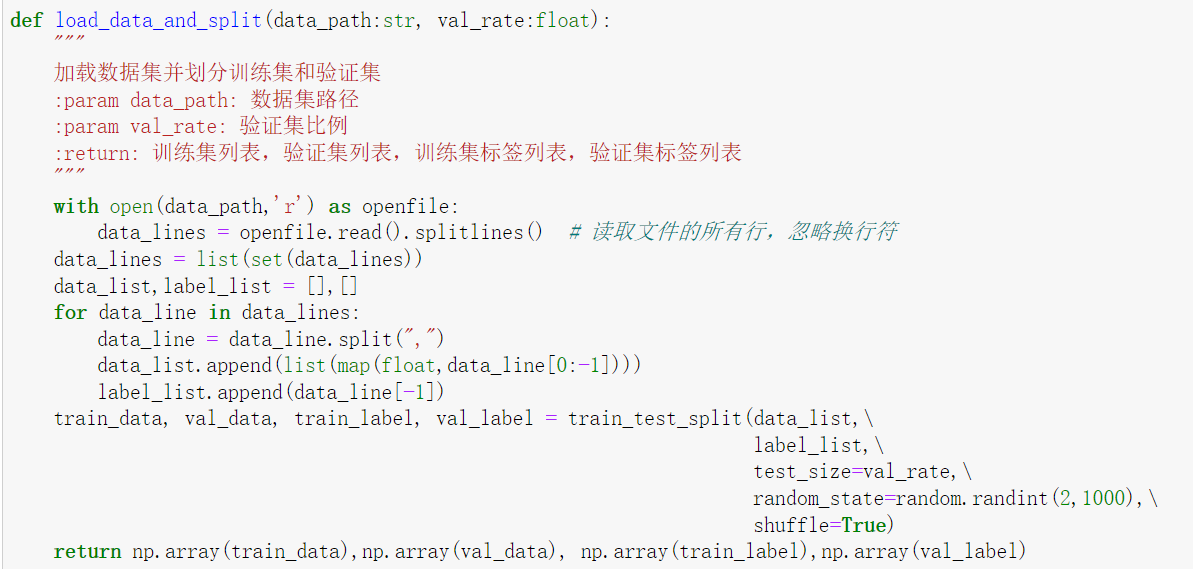


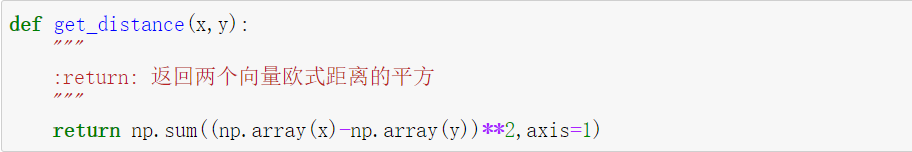
由上图可知，数据集共三个类别。样本的四个属性中，lris-setosa的sepal-width较其他类别有明显的差异。不同类别中的其他三个属性仅有微小的差异。这也考验算法挖掘微小差异并将其于分类的能力。

**3. KNN算法实现**

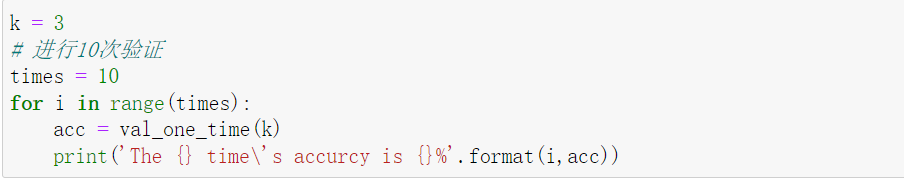
代码实现如下：











上述程序的流程如下图所示：

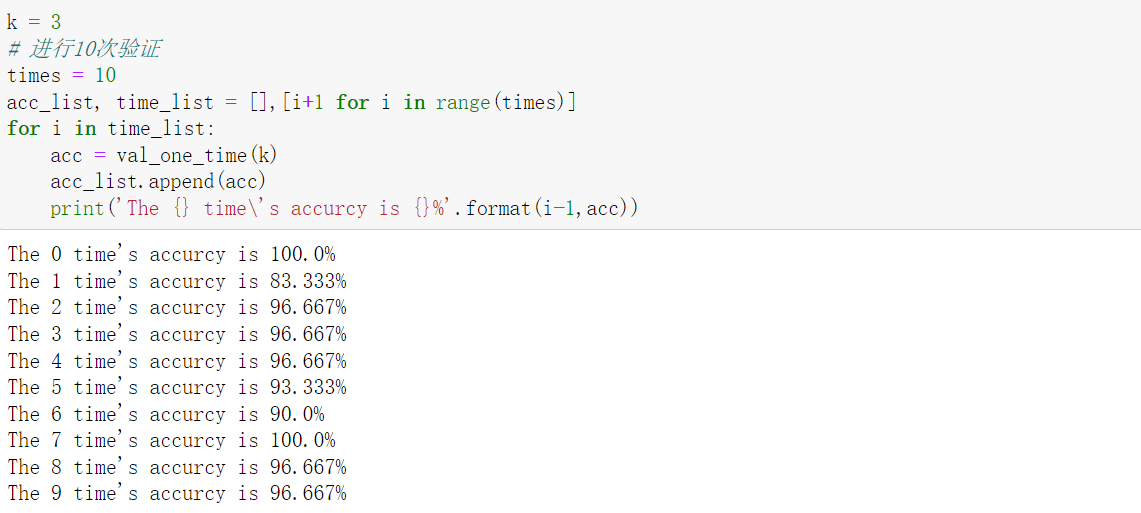
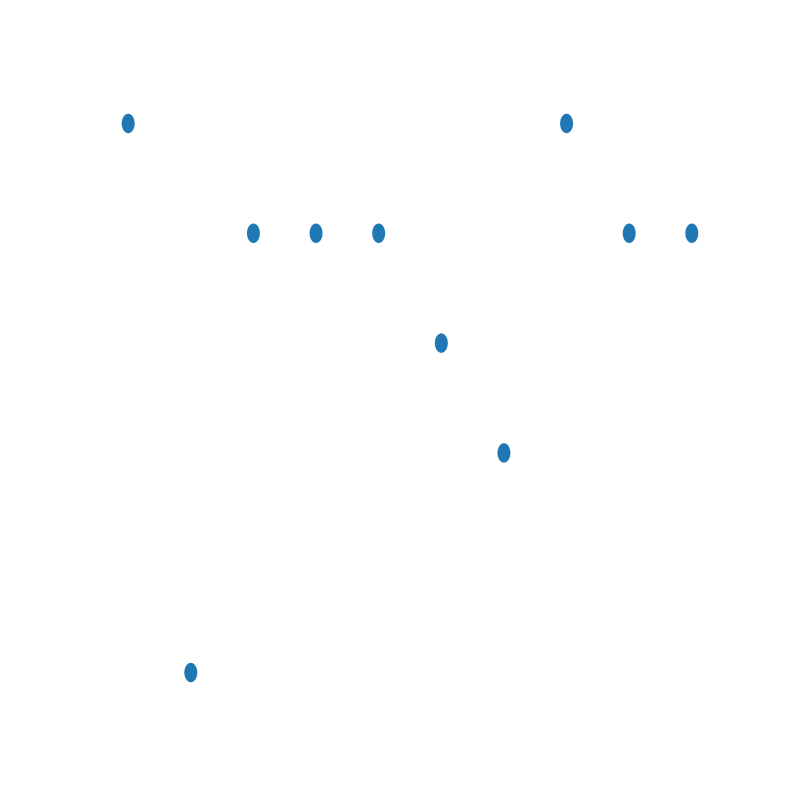


其中，KNN算法的流程图如下：

****

**4. 运行结果**

程序运行结果如下所示。

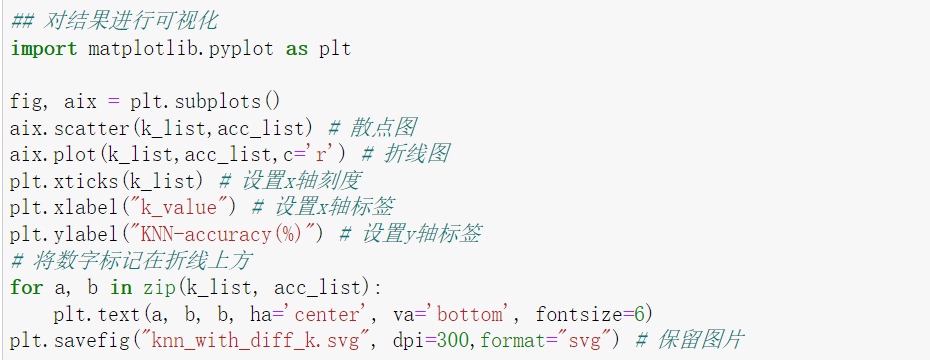
 

注意到划分训练集和验证集时，程序会将数据集打乱并进行随机采样，所以每次运行结果不尽相同。但是观察运行结果可知，10次预测准确率大部分在90%以上，甚至存在准确率为100%的运行结果，说明该模型在鸢尾花数据集上有较好的分类效果。由此可见，KNN模型不仅简单易懂，而且高效，仅用少数代码便能获得较为理想的效果。

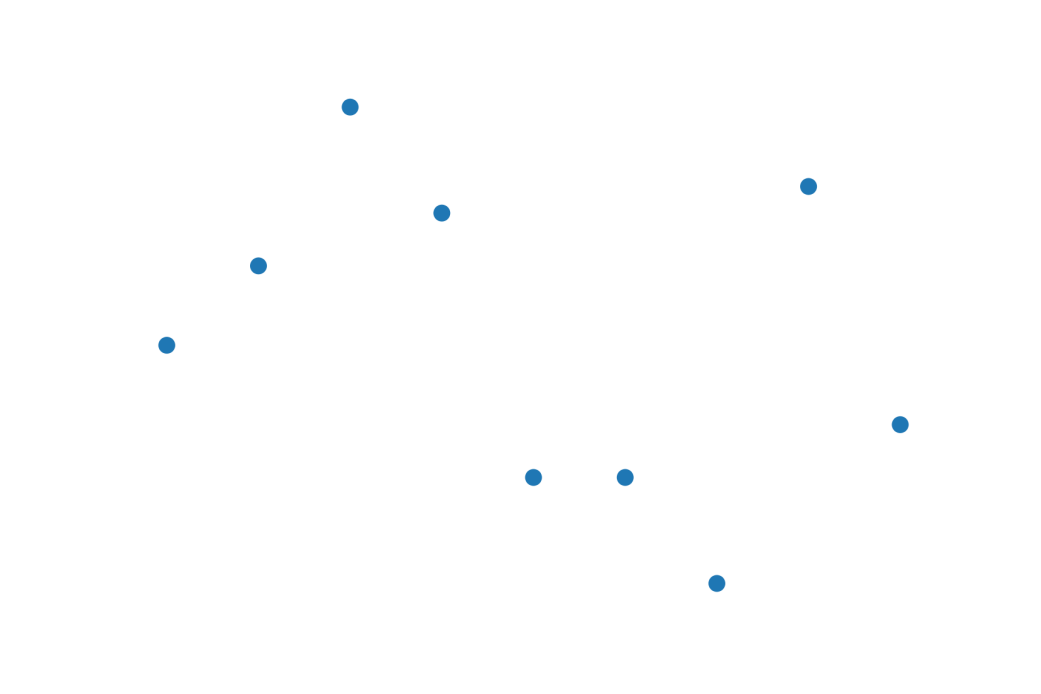
1. **调整参数**

KNN算法中存在一个超参数k,即选择与给定样本的邻居的数量。k值的选取对k近邻算法的结果会产生重大影响。如果选择较大的k值，可以减少学习的估计误差，但同时学习的近似误差会增大。而选择较小的k值则相反。为了使模型更好地拟合数据，下面对比不同k值下模型预测的准确率。代码如下：





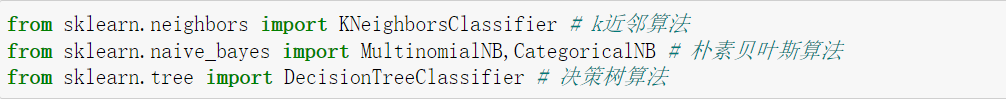
程序对比k从2到10的预测能力。对于每个k，进行50次验证计算准确率的平均值作为最终准确率，有效地减少了由于随机取样而带来的抖动。程序运行结果如下：



由此可以发现，在本次运行结果中，选择k=4将会获得更高的准确率，k=8时效果最差。当然，由于数据集仅有150个样本，训练集为数据集的80%，数量更少，所以无法直接下定结论k=4一定为最佳选择。同时观察到，由不同的k预测获取的准确率相差不大，所以k选择3、4、5、9均是合理的。

1. **模型对比**

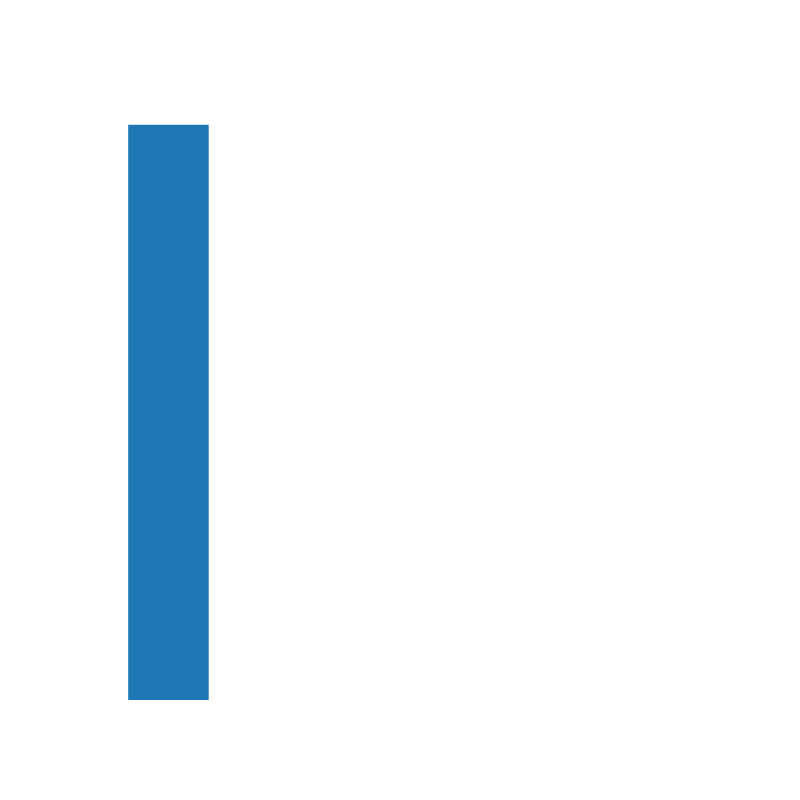
评价一个模型的好坏，不仅仅要看其准确率、稳定性等横向数据，还需要与其他算法进行纵向对比。本小节选择sklearn库中封装好的KNN算法、决策树、朴素贝叶斯三个算法与本次实验实现的KNN进行对比。代码如下：







程序运行结果如下：



可以看到四种算法进行50次预测的平均准确率相差不大，但是朴素贝叶斯算法和决策树算法50次预测的准确率的方差要比k近邻算法的略大。这是因为朴素贝叶斯模型需要知道先验概率，而先验概率很多时候取决于假设，假设的模型可以有很多种，因此在某些时候会由于假设的先验模型的原因导致预测效果不佳。本次实验假设样本各个属性独立且均符合某种分布，这或许是导致该模型不稳定的原因。而对于决策数算法而言，由于其太依赖于数据的稳定性，数据中的微小变化可能会导致完全不同的树生成。程序进行了50次测试，每次都是随机采样，这便有可能导致其稳定性下降。

综上，四种算法均有优劣。虽然k近邻算法较为稳定，但是预测准确率更低，而朴素贝叶斯算法和决策树算法虽然准确率较高，但其相较于前者又更不稳定。在实际应用中，还需要从更多的维度考察才能判断孰优孰劣。

**六、实验总结**

本次实验先是根据算法流程使用python搭建了一个自己的knn算法，在验证集上获得90%以上准确率。准确率较高，可以接受。然后通过调整knn算法中的超参数k,对比了不同k值下该模型在验证集上预测的准确率。发现在k取值从2变化到9的过程中，模型的准确率并无明显差异，但是k=3,4,5,9，模型预测的准确率更高。考虑应该是数据集太小，导致改变k后，模型的准确率波动不大。最后将本次实验中实现的knn算法与sklearn库中封装好的knn算法、朴素贝叶斯算法以及决策树算法进行了对比。发现knn算法比朴素贝叶斯以及决策树算法稳定性略好，但是准确率略差。同时，也发现本次实验中实现的knn算法比sklearn库中的knn算法准确率略低，稳定性一致。虽说差异不是特别明显，但是仍能感觉到sklearn库的强大。

同时，在完成实验后，我也切身地感受到了knn算法的优点：

1. 简单、易于理解和易于实现。本次实验仅了几行代码便实现了knn算法的核心功能。
2. 天然可以处理多分类问题，适合对稀有事件进行分类。本次实验所实现的knn算法在鸢尾花数据集获得了超过90%的分类准确率，准确率较高。
3. 和朴素贝叶斯之类的算法比，对数据没有假设，稳定性好，对异常点不敏感。

当然，其缺点也不可忽视：

1. 计算量大。当特征数非常多的时候，其计算所消耗的时间是巨大的。相比

之下朴素贝叶斯算法和决策树算法的时间复杂度更低。

2. 对样本的平衡性依赖较高。样本不平衡的时候，如一个类的样本容量很大，而其他类样本容量很小时，有可能导致当输入一个新样本时，该样本的K个邻居中大容量类的样本占多数，大大增加了犯错率。当然，本实验中的数据集平衡性较好，这也是knn算法获取的预测准确率较高的原因之一。

**七、参考资料**

《数据挖掘原理与实践 蒋盛益，李霞，郑琪编著》