**实验报告**

周韧哲 181220076

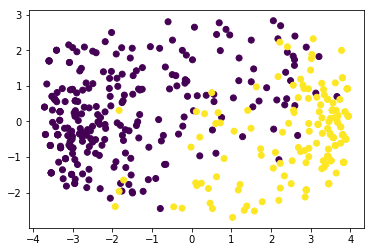
**任务一&任务三：**

首先读取文件，将Dataframe格式的数据的columns设为‘party’、’proposal1’~’proposal16’，为了方便接下来的处理，将数据中的’y’用1代替，‘?’用0代替，’n’用-1代替，‘republican’标记为1，’democrat’标记为0，然后将此 Dataframe按’party’列排序。结果如图（部分）：

图片包含 黑色

已生成高可信度的说明（图0）

先用包StandardScaler将投票矩阵X规范化。原数据有16维，为方便可视化，我们先用PCA将其降为两维，得出前两维的主成分比例为0.4626477与0.08919882，说明第一主成分占据了大部分。先将结果可视化一下：

（图1）紫色为标记为0的点，黄色为标记为1的点。

这里我自己实现了一个KNN算法my\_knn。在类my\_knn中，fit()的三个参数分别为训练集数据矩阵X、训练集数据标记矩阵Y与相邻点个数n\_neighbors。我采用的是欧几里得距离来计算点间距，在预测函数predict()中传入测试集数据矩阵X，返回的是预测的标记矩阵。我还定义了一个评估函数score()，评估标准是正确率，即：预测标记等于真实标记的个数/总个数。

主体函数如下：

图片包含 文字, 电话, 手机, 就坐

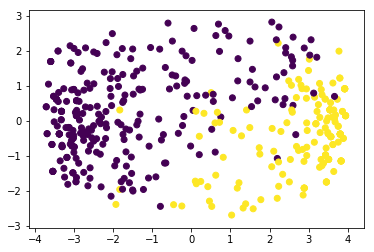
已生成极高可信度的说明（图2）

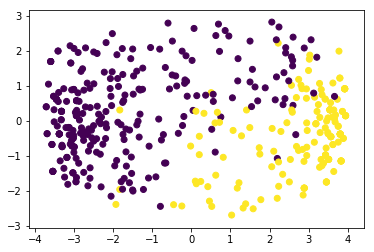
用train\_test\_split包划分训练集与测试集，测试集比例为0.3.这里我的n\_neighbors取了两个不同的值以观察不同初始情况对聚类结果的影响。

图片包含 文字

已生成高可信度的说明（图3）

打印出两者的正确率（泛化误差），score of knn\_1: 0.9236641221374046，score of knn\_2: 0.9389312977099237，即将n\_neighbors设置为15时比10的正确率要高，而我测试了几组不同的初始情况：5，20，30，40等，经过调参测试，n\_neighbors为14、15时正确率最大。0.9以上的正确率说明此模型的泛化效果还是不错的。可视化结果如下：

（图4）knn1

（图5）knn2

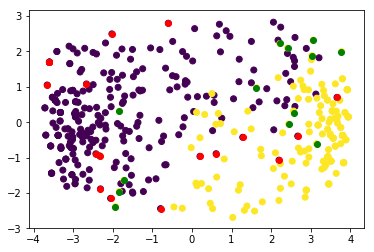
在图1中，即原数据的分布图，可以看出在左侧紫点聚集的中心有一颗黄点，在左下侧也有一颗孤立的黄点，在右上侧紫点分布密度不太均匀。所以可以初步判断数据中存在一些异常点。

用LocalOutlierFactor包来进行异常检测，其原理是k近邻。因为我前面已经对data按’party’列进行了排序，所以我可以取出其中所有标记为0的数据赋给矩阵X0，所有标记为1的数据赋给矩阵X1。将model中的k值取15，训练后得到预测值y0,y1。

图片包含 屏幕截图

已生成极高可信度的说明（图6）

可视化结果如下：

（图7）

图中紫点与黄点与图 1中的意义是一样的，红点表示原来属于标记为0（紫色）的点而被检测出异常的点，绿点表示原来属于标记为1（黄色）的点而被检测出异常的点。从图中可以看出红点大部分是属于“混进“黄点聚集区域中的紫点，而绿点大部分是属于”混进“紫点聚集区域中的黄点。在一定的误差范围内，我们可以判定这些点为异常点，其原数据是异常数据。

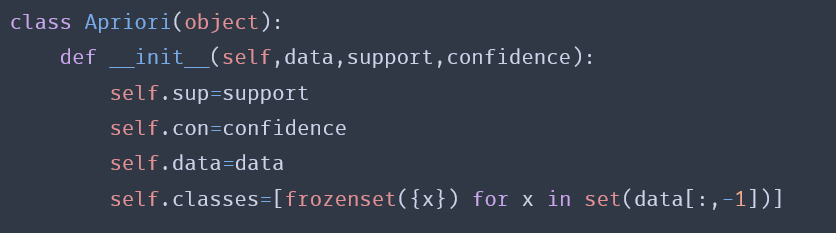
**任务二：**

首先读取文件，因为每一个属性的值是有相同的，为了区分它们，我给每一个属性的值前加上此属性首字母：

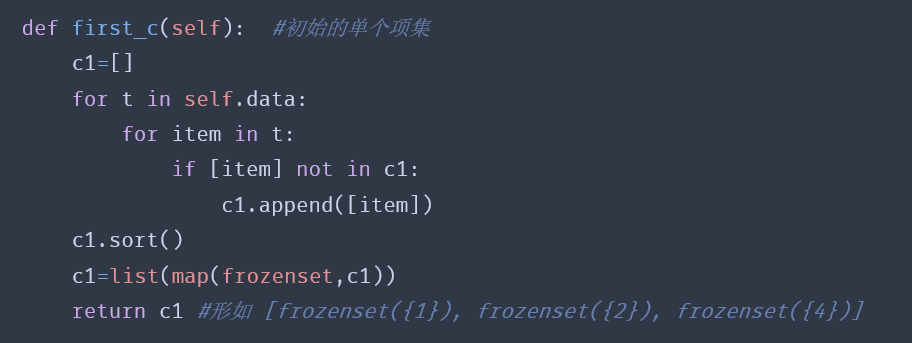
图片包含 文字, 屏幕截图

已生成高可信度的说明（图1）

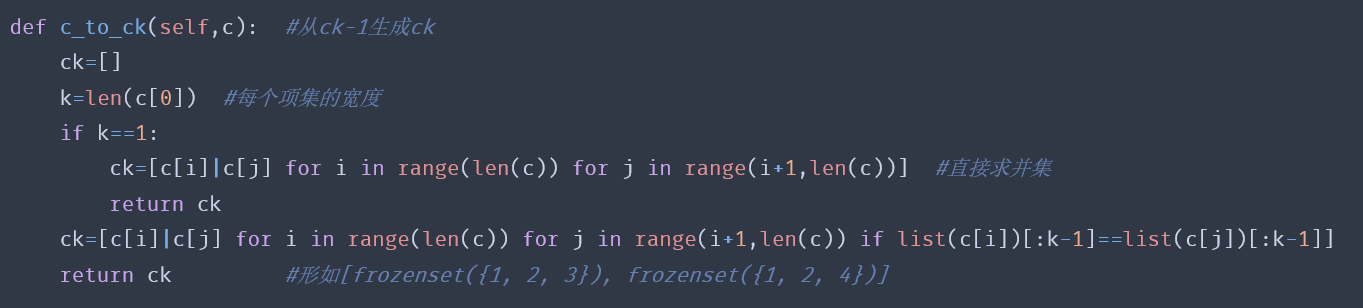
我自己实现的一个Apriori算法如下：类Apriori中，self.sup是支持度，self.con是置信度，self.data是传入的数据矩阵，self.classes是数据的分类结果，分类结果数据类型为forzenset。

（图2）

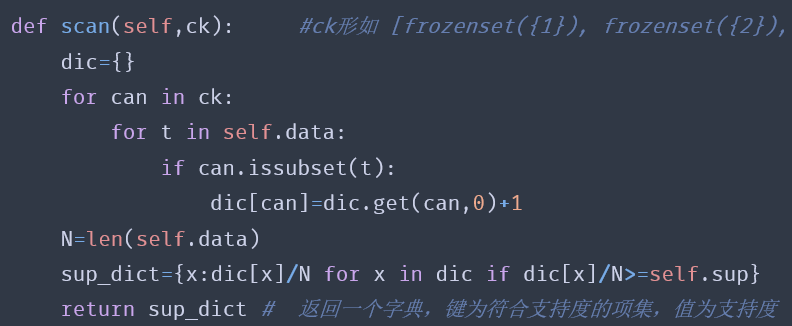
函数first\_c是用来生成初始项集的，每个初始项集中只有一个项，返回的是一个储存项集的列表，项集的数据类型是frozenset，以便之后可以作为字典的键值。

（图3）

函数c\_to\_ck是从含k-1个项的项集生成含k个项的项集，生成规则是若两个项集的前k-2个项相同，则合并。这一合并的前提是传入函数的项集是排好序的。

（图4）

函数scan是扫描项集，将不满足支持度要求的项/项集去掉，返回一个字典，键为符合支持度的项集，值为支持度。

（图5）

fit函数data\_lst储存满足支持度的项集，最后返回的data\_lst是一个字典，键为项集，值为支持度，并将其赋给self.freq以便在关联分析时调用。

图片包含 文字

已生成高可信度的说明（图6）

函数association用来关联分析，通过挖掘频繁项集self.freq来找出满足置信度要求的值-分类结果对，返回的dict\_a是一个字典，键为字符串‘值-分类结果’，值为置信度。



（图7）

先来看评估结果为unacc(unacceptable)的关联挖掘。由于评估结果为unacc的二手车比例为0.7，占据大多数，所以其支持度与置信度应要最高，我将支持度设为0.15，置信度设为0.6，结果如下：

图片包含 文字

已生成极高可信度的说明（图8）

可以看出low safety和2 persons与 unacc关联最强，置信度为1，即只要是low safety和2 persons都被评估为unacc的，而关联次强的是vhigh paint和vhigh buying，置信度约为0.83，此外high buying、2 doors、small lug\_boot、high paint等也与评估为unacc关联较强。

再来看看评估结果为acc(acceptable)的关联挖掘。由于评估结果为unacc的二手车比例约为0.22，较之unacc很小，所以将支持度与置信度设置低一些才能挖掘出一些关联结果，我将支持度设为0.05，置信度设为0.3，结果如下：

图片包含 屏幕截图, 文字

已生成高可信度的说明（图9）

可以看到，high safety+4 persons对acc的关联程度程度最高为0.5625，其次是med safety+more persons，置信度为0.5，med safety+4 persons 和med safety+more persons对acc的关联程度也较高。

再来看看评估结果为vgood(very good)的关联挖掘。由于评估结果为vgood的二手车比例约为0.04，较之unacc、acc很小，所以将支持度设置更低一些才能挖掘出一些关联结果，我将支持度设为0.01，置信度设为0.3，结果如下：

图片包含 文字

已生成极高可信度的说明（图10）

可以看到，low buying+high safety+big lug\_boot对vgood的关联程度最高为0.5，其次是low buying+high safety+more persons，置信度约为0.44。

再来看看评估结果为good(good)的关联挖掘。由于评估结果为good的二手车比例约为0.038，较之unacc、acc很小，所以将支持度设置更低一些才能挖掘出一些关联结果，我将支持度设为0.005，置信度设为0.3，结果如下：

图片包含 文字

已生成极高可信度的说明（图11）

可以看出，各种属性的值对good的置信度较为均匀，处于0.3~0.4之间，其中low buying+low paint+med safety、low buying+med paint+med safety和med buying+low paint+med safety对good的关联程度最高，约为0.36。

为了对数据进行聚类，我将数据映射到数域上：

图片包含 户外

已生成极高可信度的说明

（图12）

我尝试用knn来进行聚类，导入KNeighborsClassifier来训练，将数据用StandardScaler规范化，用train\_test\_split划分训练集与测试集，测试集比例为0.3，knn距离为欧几里得距离，k值为10，可计算出测试集的正确率(泛化误差)为0.9691714836223507。

图片包含 文字

已生成高可信度的说明（图13）

knn聚类是根据距离最近的k个点中的主要类别来判定此点属于哪个类别，而某一些值对类别的关联越大，它们就越会聚集在一起，故关联分析中对某一类别置信度高的点，在聚类中被判定为此类别的概率就越大。