第二次大作业—分类与聚类

数据仓库与数据挖掘

清华大学软件学院

彭程 2018214152 原之宇 2018214168

2018 年 12 月 13 日

目 录

1 任务一

1.1 数据清洗

1.1.1 缺失值处理

原数据共 41188 条记录，有 20 个特征，其中如下表1中 6 个类别特征存在 unknown 缺失值。可注意到 default 特征的缺失值比例较高，超过了 20%，其余属性的缺失值比例比较低。

表 1: 数据集中 6 个存在 unknown 缺失值的类别属性状况

Attribute	#Unknown	Unknown Ratio
default	8597	20.873%
education	1731	4.203%
housing	990	2.404%
loan	990	2.404%
job	330	0.801%
marital	80	0.194%

对于有缺失值的数据，可以采用删除记录或属性列，将缺失值视为属性类别，填补缺失值这三种处理方式。其中，删除记录或属性列会损失信息；将缺失值 unknown 直接视为属性类别不会对后续分类有明显帮助。填补缺失值常见方法包括用已有记录的众数来直接填补和用分类器来预测缺失值两种方式，用众数填补的效果差于用分类器来预测填补的效果。

对于真正的应用场景，采集到的记录中存在缺失值是难免会发生的情况，对于本问题的数据背景场景，当需要预测某个客户在电话销售中认购银行产品的概率时，我们不能因为客户的信息中有 unknown 就跳过这个客户。所以不能直接删除有缺失值的记录，而必须提供有效地对 unknown 值的处理机制。

本实验中用自己实现的随机森林来预测 unknown 的值进行缺失值填补。具体来讲，我们将数据集中 30488 条不存在缺失值的记录用作训练集，然后依次将上述 6 个有缺失值的属性用作预测类别，训练了 6 个随机森林来分别预测填补了这些属性处有缺失值的记录中的 unknown 值。

1.1.2 属性编码

数据集中除了 y 标签外还有 20 个属性，其中类别属性占 10 个，连续数值属性占 10 个。原数据集中类别属性的取值都是可枚举的字符串，我们将所有的类别属性的取值转化成离散的数值表示来方便后续处理流程。

表 2: 对类别属性的取值数值映射

Attribute	Value Map
job	{'entrepreneur': 0, 'admin.': 1, 'management': 2, 'blue-collar': 3, 'technician': 4, 'self-employed': 5, 'services': 6, 'housemaid': 7, 'retired': 8, 'student': 9, 'unemployed': 10}
marital	{'divorced': 0, 'married': 1, 'single': 2}
education	{'illiterate': 0, 'basic.4y': 1, 'basic.6y': 2, 'basic.9y': 3, 'high.school': 4, 'professional.course': 5, 'university.degree': 6}
default	{'no': 0, 'yes': 1}
housing	{'no': 0, 'yes': 1}
loan	{'no': 0, 'yes': 1}
contact	{'cellular': 0, 'telephone': 1}
month	{'jan': 0, 'feb': 1, 'mar': 2, 'apr': 3, 'may': 4, 'jun': 5, 'jul': 6, 'aug': 7, 'sep': 8, 'oct': 9, 'nov': 10, 'dec': 11}
day_of_week	{'mon': 0, 'tue': 1, 'wed': 2, 'thu': 3, 'fri': 4}
poutcome	{'nonexistent': 0, 'failure': 1, 'success': 2}
y	{'yes': 0, 'no': 1}

1.1.3 特征筛选

首先我们根据数据集说明文件删除了特征属性 duration，然后对剩下的 19 个特征进行分析。常见的特征筛选方法大致可以分为三类：过滤式 (filter)、包裹式 (wrapper) 和嵌入式 (embedding) [1]。

过滤式特征筛选算法直接用一些统计的方式来对特征进行排名，并根据排名来取靠前的特征，在特征的筛选过程中不涉及到学习模型，因此计算速度很快。常见的过滤式方法有 T 检验 (*t*-test) [2], 卡方检验 (χ^2 test) [3], 互信息 (mutual information) [4], 最大信息系数 (maximal information coefficient, MIC) [5] 等。包裹式的方法用分类学习模型的效果来筛选不同的特征组合，通常使用遗传算法 (GA) [6]，粒子群算法 (PSO) [7] 等群智能算法来迭代优化生成的特征子集，但时间开销较大。嵌入式特征筛选方式将分类模型的训练过程与特征筛选过程融为一体，在模型的训练过程中自动进行了不同特征的重要性比较已经特征的选择。

本实验采用了两种过滤式方法来对特征进行排名：*t*-test 和 MIC，被考察了特征的 *p*-value 取值。T 检验是常用的有效判断两组样本在统计上是否有显著差异的方式，被广泛用于高维数据的特征筛选问题 [8, 9]。T 检验算出的显著性水平以 p=0.05 为广泛采用的阈值 [10]，特征的 p 值越低，则其从统计意义来讲在正负两组样本中的差异性越显著，即对于分类任务越重要。MIC 是 2011 年发表在 science 上的一个基于信息论的计算不同变量间关联关系强弱的方法，可以检测出两个变量间各种线性或者非线性关系，在很多大型数据集上都有优秀表现 [11, 12]。

本实验还采用了两种嵌入式的方法来对特征进行排名：递归特征消除 (Recursive Feature Elimination, RFE) 和随机森林 (Random Forest, RF)。RFE 方法是 2002 年由 Guyon

等人 [13] 提出的递归式的特征子集删选方式，最初被用在有高维基因特征的基因微阵列数据集上来筛选显著差异表达的基因特征，后来被广泛应用到各种特征筛选问题上。此外，随机森林也可以输出特征的重要性排名。

用 *t*-test，最大互信息系数，递归特征消除与随机森林这四种方式对特征进行排名的结果见表3，我们还在表中列出了每个特征在 *t*-test 中的 *p*-value 值。

此外，我们在图1中画出来每个特征在正反两类样本上的小提琴图，即箱形图和密度图的结合，可以用来观察数据的分布形状。小提琴图中间的黑色粗条表示四分位数范围，从其延伸的幼细黑线代表 95% 置信区间，而白点则为中位数。

首先查看表3中各个特征在 *t*-test 中算出的 *p*-value，可看到只有两个特征 default 和 loan 的 *p*-value 分别是 0.4256 和 0.4692，远远大于 p=0.05 的统计差异阈值，也就是说这两个特征在正反两类样本上差异性非常小。对于 default 特征，我们进一步观察发现在全部的 41188 条记录中，只有 3 个 'yes' 取值，并且共有 8597 个 (20.873%) 的 'unknown' 缺失值，缺失比例远远高于其他属性，而且 *t*-test、MIC、RFE、RF 这四种方法对 default 的排名分别是 18、18、19、19，是所有属性中排名最差的。因此属性 default 不仅区分能力弱，而且有超过 1/5 的缺失值记录，我们决定删除这个特征属性。对于 load 特征，虽然 *t*-test 和 MIC 分别将其排名为最低的 19 和 19，但在 RFE 和 RF 的排名中居于 15 和 13，也就是说该属性在分类过程中还是有一些意义的，我们选择保留。对于其他特征，除了 housing 和 day_of_week 的 *p*-value 分别是 0.0012 和 0.0414(低于 0.05) 外，剩余的 15 个属性的 *p*-value 在保留 4 位小数情况下都是 0.0000，说明在正负两组样本上有统计上明显的差异性，这些特征我们都保留。

观察图1，可以看到多数特征在正负两组样本的数据分布是有比较明显的差别的。比如我们观察特征 euribor3m，其在 *t*-test、MIC、RFE、RF 方法中分别排名 5、1、2、2，整体来看是所有特征中最高的，其小提琴图中正负两种样本的数据分布图是形状明显不同的，而且我们可以注意到其在正负两组样本中中位数分别是大于 4 和小于 2(即两个白点)，位置相差明显。整体来看，在四种特征排名中位置越靠前的特征，其小提琴图中正负两类样本的数据分布图形状越明显。

也就是说，最终我们只删除了原数据中的特征 duration 和 default，保留了其它所有特征。

1.1.4 不平衡样本处理

原始数据集中共正负样本数分别是 4640 和 36548，存在数据不均衡分布问题。我们将所有样本的 30% 作测试集，然后将剩余的共 28831 个样本用作训练集，在得到的训练集中共 3231 个正样本，25600 个负样本。训练数据的不均衡会影响分类器的效果，导致分类器的预测结果更倾向于数量多的类别。

我们使用了 SMOTE 算法 [14] 来对样本进行重采样来平衡数据不均衡现象。简单的通过直接“过抽样”来补齐类别较少的类会导致较少样本大量重复出现，使得分类模型过拟合，但是 SMOTE 算法基于插值的思路来为分析少数类的已有样本来合成新的样本，效果明显好于直接过采样的方法。

表 3: 使用 *t*-test，最大互信息系数，递归特征消除与随机森林来对所有特征进行排名

Attribute	*t*-test 排名	MIC 排名	RFE 排名	RF 排名	*p*-value
age	15	10	16	1	0.0000
job	10	12	11	3	0.0000
marital	12	15	5	9	0.0000
education	13	14	9	5	0.0000
default	18	18	19	19	0.4256
housing	16	17	18	10	0.0012
loan	19	19	15	13	0.4692
contact	7	11	3	18	0.0000
month	14	8	14	14	0.0000
day_of_week	17	16	12	7	0.0414
campaign	9	13	8	4	0.0000
pdays	1	6	17	8	0.0000
previous	2	9	4	17	0.0000
poutcome	3	7	1	11	0.0000
emp.var.rate	4	5	6	16	0.0000
cons.price.idx	8	2	10	15	0.0000
cons.conf.idx	11	3	7	12	0.0000
euribor3m	5	1	2	2	0.0000
nr.employed	6	4	13	6	0.0000

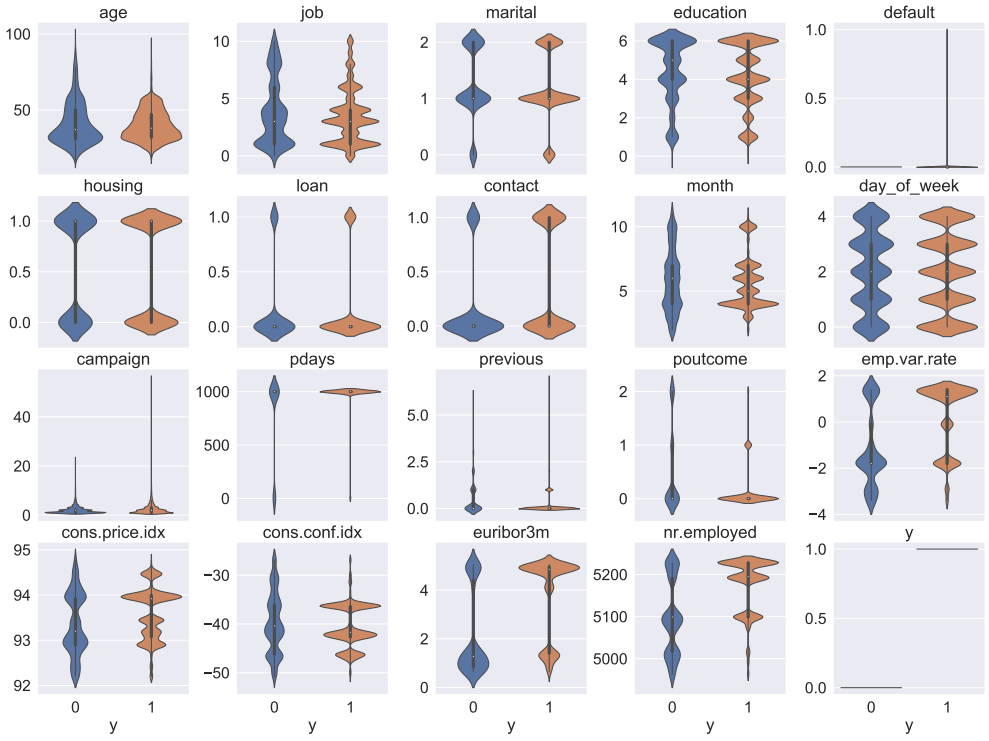


图 1: 用小提琴图来直观展示所有属性特征在正反两类样本上的数据分布情况

由于正样本数量明显较少，大量生成正样本可能会改变数据的原始分布，而且过量的训练样本也会导致后端训练模型的更大的时间开销，我们最后通过 SMOTE 算法将正样本的比例调整到负样本的 1/2，并且实验测试发现负样本数量再增大不会导致测试集中正样本上预测准确率的明显提升反而可能导致负样本上预测准确率的下降。

1.2 分类器设计

我们实现了三种分类器：朴素贝叶斯，决策树 C4.5，以及随机森林，参考了周志华老师的《机器学习》[1]。实现的三种分类器都可以同时处理离散属性和数值属性，分别对应数据集中 categorical 和 numerical 两类属性。此外，我们在决策树中还实现了后剪枝过程，在随机森林的 Bagging 过程中添加了针对不平衡数据集的采样方式。

1.2.1 朴素贝叶斯

朴素贝叶斯模型基于贝叶斯公式，然后用所有属性相互独立的假设来简化计算过程 (公式1)。该模型直接基于训练集来估计先验概率 $P(c)$，并为每个属性估计条件概率 $P(x_i|c)$。

$$P(c|x) = \frac{P(c)P(x|c)}{P(x)} = \frac{P(c)}{P(x)} \prod_{i=1}^{d} P(x_i|c) \tag{1}$$

实现的分类器同时支持离散属性和数值属性的处理。在计算 $P(x_i|c)$ 时对于离散属性直接统计训练集中类别 c 下属性 x 中取值 x_i 出现的概率即可；对于连续属性我们使用高斯分布来求估计值 $p(x_i|c)$(公式2)，其中 $\mu_{c,i}$ 和 $\sigma_{c,i}$ 分别是第 c 类样本在第 i 个属性上取值的均值和方差。

$$p(x_i|c) = \frac{1}{\sqrt{2\pi}\sigma_{c,i}} \exp\left(-\frac{(x_i - \mu_{c,i})^2}{2\sigma_{c,i}^2}\right) \tag{2}$$

此外，算法实现时我们对离散属性使用了拉普拉斯平滑 (Laplacian Smoothing)；对连续属性高斯分布计算时会处于分母位置的方差 $\sigma_{c,i}$ 也设置了默认平滑值 1e-9。

1.2.2 决策树 C4.5

实现了决策树 C4.5。算法递归地用当前最优划分属性来分割训练集，直到当前树结点中所有样本都是同一类，或者达到最大树高阈值，或者结点分裂允许的最小样本数阈值等条件后可以设置叶结点，让其对应的分类结果是当前结点中样本数量最多的类。属性划分时采用的度量方式是信息增益。对于连续属性采用二分法的机制，会从该属性的取值中等间隔地选取候选值来计算最优的分割点。

为防止“过拟合”，还实现了“后剪枝”的策略，即在树完全地生长完成后，用独立的验证集来自底向上地对树中结点进行考察，根据验证集上的效果来决定是否剪枝。“后剪枝”的效果一般优于贪心的“预剪枝”策略，不过算法的时间开销也会明显上升 [1]。

实现的决策树在创建的时候有三个可设置的属性，最大树高 (max_depth)，允许结点分裂的最少样本数 (min_samples_split)，以及对于连续属性二分划分时候选取值的数量 (max_continuous_attr_splits)。前两个可防止“过拟合”，实验中经测试设置成了 35 和 2，最后一个候选划分位置的数量直接影响到算法的运行时间，当样本数量较大时，该值应设置得稍小一些，实验中我们经测试比较后在简单决策树中取 50，在随机森林中取 10。

更多细节可参见代码，有较多注释。其实决策树的实现花了三个分类中最长时间，实现得比较仔细。

1.2.3 随机森林

随机森林是集成学习模型，其核心是 Bagging 策略结合引入了随机属性选择的决策树。训练每颗决策树时，采用自助采样法从 N 个样本中有放回地进行 N 次采样形成训练集，然后在决策树训练的过程中每次随机选择部分属性在其中找最优的划分属性来完成树的生成，最终预测时采用投票法 (少数服从多数) 来整合所有树的预测结果来预测一个样本的类别。决策树每次找最优划分属性时，我们随机从当前结点的属性集合中 (假设有 d 个属性) 选择 $\log_2^d$ 个候选属性来从中找最优划分属性 [15]。

由于我们训练数据中正负样本是不平衡的 (正负比 1:2)，因此我们提供了 Bagging 策略的稍微修改版来实现正负两种样本的平衡采样，即自助采样时在正负两类样本中分别采样 N/2 个样本来形成训练集。实际使用时原始的 Bagging 策略和稍调整后的 Bagging 策略可以用参数来进行选择。

1.3 结果与分析

1.3.1 评价指标

我们使用了常见的大多数二分类指标来评价分类模型效果，包括：Sensitivity, Specificity, Accuracy, Precision, Recall, F1-Score, MCC(Matthews Correlation Coefficient), AUC。敏感度 Sensitivity 和特异性 Specificity 分别衡量模型在正负两类样本上的分类准确比例，准确率 Accuracy 衡量模型在两类样本上整体的分类准确比例。精度 Precision 和召回率 Recall 分别关注分类模型的查准率和查全率，F1-Score 是精度和召回率的调和平均。马氏相关系数 MCC 是一个综合型的比较全面均衡的评价指标。AUC 是 ROC 曲线与坐标轴围成的面积，衡量分类模型把正样本排在负样本前面的概率。

用 P 和 N 分别表示正负两类样本的数量；TP 和 TN 分别表示被正确分类的正样本数和负样本数；FP 和 FN 分别表示被错误分类的正样本数和负样本数。上述指标的

计算方式如公式3所示 (Recall 和 Sensitivity 计算方式相同)：

$$\begin{aligned}
&\text{Sensitivity} = \frac{TP}{TP+FN}, \quad \text{Specificity} = \frac{TN}{TN+FP}, \quad \text{Accuracy} = \frac{TP+TN}{P+N}, \\
&\text{Precision} = \frac{TP}{TP+FP}, \quad \text{Recall} = \frac{TP}{TP+FN}, \quad \text{F1-Score} = \frac{2\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \\
&\text{MCC} = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}
\end{aligned} \tag{3}$$

1.3.2 结果对比与分析

图4展示了不同分类模型在测试数据集上的性能表现，粗体数值为某个指标行中的最高值。可以看到不同模型在不同指标上表现情况有差异，没有一个模型在所有指标上都高过其余模型。决策树 (后剪枝) 模型相比于决策树模型增加了后剪枝过程，即把训练数据集的一部分留出来做验证集，然后等树模型生长完成后用验证集来对树进行剪枝。

朴素贝叶斯模型在正样本中的识别准确率最高，其 Sensitivity/Recall 值达到 0.6375，远远超过别的模型，而且其 AUC 值也是所有模型中最高的，该模型的不足之处在于负样本上的准确率 Specificity 和整体准确率 Accuracy 明显低于决策树模型和随机森林模型。

决策树模型在进行了后剪枝之后在 7 个指标上都超过了未剪枝的模型，比如 Precision,F1-Score 和 MCC 都上升了约 6 个百分点，说明后剪枝策略效果非常明显。我们也记录了进行后剪枝的模型中结点的数量的变化情况，在剪枝前树中有共 8522 个结点，其中 5612 个是叶节点；在剪枝后模型中只剩了 2393 个结点，其中 1428 个是叶结点，即剪枝后模型大小大约变成了原来的 1/3，但是性能却有明显的上升。

随机森林模型中的 Bagging 过程我们给出了两种实现，第一种是不考虑训练解的不平衡问题，每次直接在所有训练样本中有放回的采样 N 个样本来训练一棵树；第二种是在训练样本中对正负两类分别单独采样，正负样本中都放回采样 N/2 次，组成正负样本 1:1 的重采样集合来训练一颗树，即对应“平衡采样”模型。第一种原始 Bagging 采样的随机森林得到了所有模型中最高的 Specificity，Accuracy 和 Precision 指标，即在负样本中的识别准确率最高，并且在两类样本中整体的分类准确率最高。第二种进行了“平衡采样”的随机森林在正样本中的识别准确率大大上升，相比于第一种随机森林在 Sensitivity/Recall 上提高了 10 个百分点 (0.4577/0.3573)，是这两个属性上数值第二高的模型，不过在负样本上的识别准确率也下滑了 3 个百分点 (0.9448/0.9101)。同时，进行了“平衡采样的”随机森林在两个综合性的评价指标 MCC 和 F1-Score 上取得了最高值。

整体来看，两个随机森林模型的性能超过了朴素贝叶斯模型和决策树模型。此外，关于 AUC 值，需要模型输出一个测试样本是正样本的概率值而不是分类标签才可以计算。对于朴素贝叶斯模型，其直接得到的不同类别上的概率值进行比例缩放后就可以得到不同类别上的预测概率；对于随机森林，我们统计一个测试样本在森林中所有树的投票结果的比例也可以输出预测概率；但是用决策树进行预测时，我们对一个测试

样本从根节点开始往下走，走到叶节点后叶节点对应的类别标签就是该样本预测结果，所以暂时不支持概率预测；因此，我们计算了朴素贝叶斯模型和随机森林模型的 AUC 值，没有计算决策树模型的 AUC 值。

表 4: 数据集上不同分类模型在不同评价指标下的性能对比

评价指标	朴素贝叶斯	决策树	决策树(后剪枝)	随机森林	随机森林(平衡采样)
Sensitivity/Recall	**0.6375**	0.3754	0.3973	0.3573	0.4577
Specificity	0.8028	0.9129	0.9300	**0.9448**	0.9101
Accuracy	0.7851	0.8553	0.8729	**0.8818**	0.8616
Precision	0.2795	0.3409	0.4052	**0.4372**	0.3792
F1-Score	0.3886	0.3573	0.4012	0.3932	**0.4148**
MCC	0.3169	0.2764	0.3302	0.3305	**0.3390**
AUC	**0.7790**	-	-	0.7555	0.7572

粗体的数值表示所有分类器在某个指标下取到的最高值。

图2显示了朴素贝叶斯模型和两种随机森林模型的 ROC 曲线以及 AUC 值，可以看到朴素贝叶斯模型的 ROC 曲线稍明显地略高于两种随机森林模型，AUC 值相应的有 2 个百分点的差距，两种随机森林模型的 ROC 曲线基本重合，AUC 值也很接近。

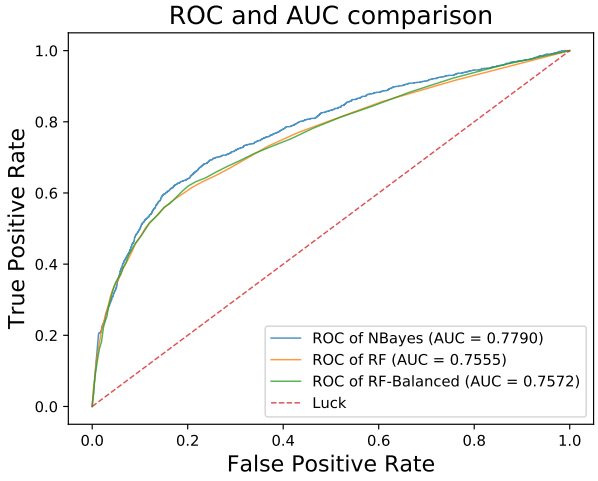


图 2: 朴素贝叶斯 (NBayes) 与随机森林 (RF) 及平衡采样的随机森林 (RF-Balanced) 的 ROC 与 AUC 对比

图3展示了两种随机森林模型随着森林中树的数量从 1 增长到 50 的的变化过程中，所有评价指标数值的变化情况。可以发现当随机森林中树的数量在 10 以内时，各个评价指标的数值存在上升的弧度，树的数量超过 10 以后各个指标的值仍然会上下抖动，但是总体趋势是这不断趋于稳定，在树的数量达到 50 左右时，各个指标的数值已经比

较平稳了。此外，可以注意到右边 RF-Balanced 图中蓝色的 Sensitivity 曲线比较明显地高过左边图中相同的蓝色曲线。

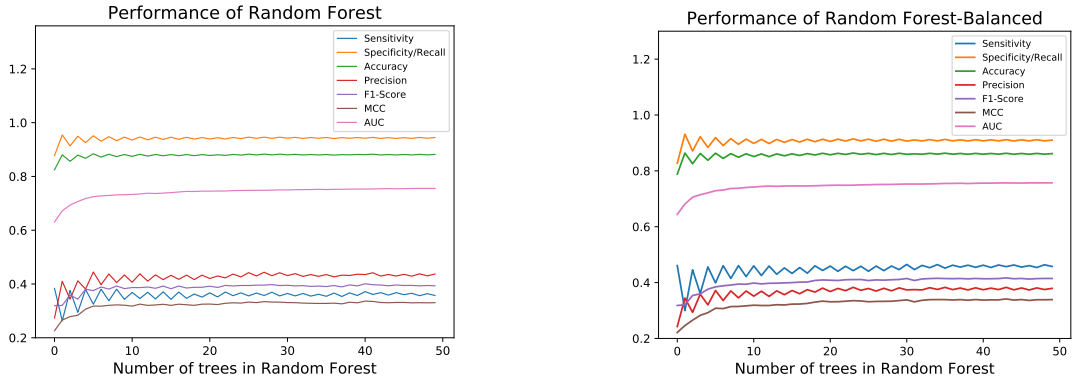


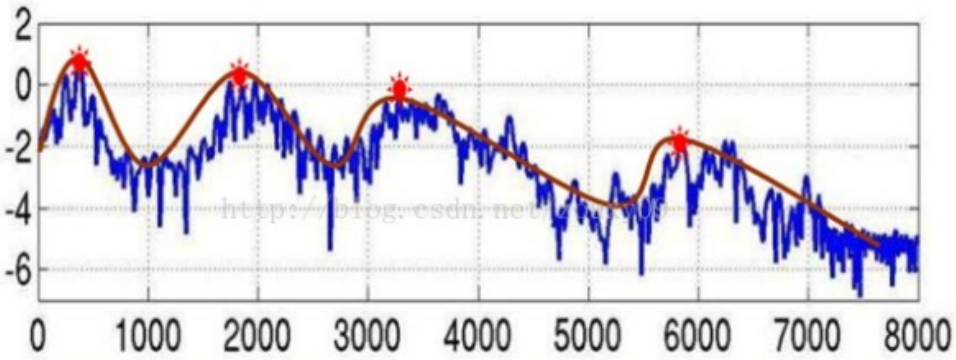
图 3: 随机森林 (RF) 及平衡采样的随机森林 (RF-Balanced) 随着子树数量增加的性能变化情况

2 任务二

聚类任务所使用的数据集是 $AnuranCalls$，来源于 $UCIMachineLearningRepository$。数据集使用 Mel 频率倒谱系数 $MFCCs$ 来描述不同科、属、种青蛙的叫声，在本数据集中 $MFCCs$ 是 22 维的特征向量。

2.1 音频特征提取

先简单描述一下 Mel 频率倒谱系数 $MFCC$ 是如何生成的。人声道的形状决定了能发出什么样的声音，而能够描述声道形状的是在语音的短时功率谱中的包络，而 $MFCCs$ 就是描述包络的一种特征。



上图包含了频谱图中的两种信息：包络（红色线条）和频谱的细节信息。我们需要把这两部分信息分开，就可以得到对包络的描述了。频谱信息使用 $logX[k]$ 描

述，包络和频谱的细节信息分别用 $logH[k]$ 和 $logE[k]$ 描述。现在是已知 $logX[k]$，求得 $logH[k]$ 和 $logE[k]$ 以满足 $logX[k] = logH[k] + logE[k]$。经过以下过程：

1. 将原语音信号经过傅里叶变换得到频谱：$|X[k]| = |H[k]||E[k]|$；
2. 两边取对数：$log|X[k]| = log|H[k]| + log|E[k]|$；
3. 再在两边取逆傅里叶变换得到：$x[k] = h[k] + e[k]$。

通过以上操作可以得到倒谱。而如果先将频谱通过一组 Mel 滤波器就得到 Mel 频谱，再将 Mel 频谱进行上述操作后，得到的是 Mel 频率倒谱系数，简称 $MFCC$。所以每个 $MFCC$ 向量是每一个小段语言的特征描述，我们可以利用这些倒谱向量对语音进行聚类了。

由于对 $MFCCs$ 没有特别深入的理解，所以没有对 $MFCCs$ 进行特征筛选，而只使用了 PCA 降维的方法与不降维的方法进行了对比。

2.2 聚类算法实现

以下两种聚类算法都是在 Jupyter Notebook 环境下编辑完成，代码文件为 k-means.ipynb 和 k-medoids.ipynb，Python 版本为 3.6。

2.2.1 k-means 聚类

k-means 算法通过最小化簇内距离平方和来把数据分成不同的簇，算法要求一开始要指定所要分的簇数 k。每个簇都有一个簇中心（centroids），它是该簇内所有样本数据的平均值，但是该中心点不一定属于样本数据。它可以很好地扩展到大量样本，并且已经在许多不同领域的大范围应用领域中使用。

$$\sum_{i=0}^{n} \min_{\mu_j \in C}(||x_i - \mu_j||^2)$$

k-means 算法流程如下：

1. 选取 k 个初始质心（作为初始 cluster）；
2. 对每个样本点，计算得到距其最近的质心，将其类别标为该质心所对应的 cluster；
3. 重新计算 k 个 cluser 对应的质心；
4. 重复步骤（2）（3），直到质心不再发生变化。

对于欧式空间的样本数据，如上面公式所示，以平方误差和（sum of the squared error, SSE) 作为聚类的目标函数，同时也可以衡量不同聚类结果好坏的指标。

k-means 存在缺点：

1. k-means 是局部最优的，容易受到初始质心的影响；

2. 同时，k 值的选取也会直接影响聚类结果，最优聚类的 k 值应与样本数据本身的结构信息相吻合，而这种结构信息是很难去掌握，因此选取最优 k 值是非常困难的。

在实验过程中，这两个缺点也体现得十分明显。算法选取的初始点是选取的随机点，因此每次的实验结果都有较大的不同。而根据实验结果来看，当 k 值从 2 变化到 6 的时候，$purity$ 和 f_score 的变化细节不确定，但是总体趋势是上升的。

2.2.2 k-medoids 聚类

k-means 与 k-medoids 之间的差异就是可以理解为对于数据样本的平均值和中位数之间的差异：前者的取值范围可以是连续空间中的任意值，而后者的取值却只能是数据样本范围中的样本。这个原因就是 k-means 对于数据样本的要求太高了，要求所有数据样本处在一个欧式空间中，对于有很多噪声的数据就会造成极大的误差。同时对于非数值型数据样本，不能够计算平均值等实数型变量。

在 k-medoids 中，相对于 k-means 的目标函数，我们将目标函数（J）中的欧式距离改写成了一个任意的 dissimilarity measure 函数 ν：

$$J = \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}\nu(x_n, \mu_k)$$

其中 ν 函数表示样本点和当前参考点之间的差异值，这个值对于非实数型的数据样本是人为提前设定的，这样，又将问题转换为了最小化目标函数的问题。

算法步骤与 $k-means$ 类似，只是将平均值为参考点改变成用样本做参考点，要全部遍历一遍。

1. 在数据样本 D 中随机选择 k 个数据样本作为质点（参考点）O_j（j=1.2.3...k）；
2. 重复地将剩下的样本点分配到 k 个簇类当中；
3. 随机选择一个非质点样本 O_{random}；计算交换对象 O_{random} 和 O_1 参考点，重复 2 中的操作，产生新的一组簇类，计算目标函数 S，若 $S<0$ 则将 O_{random} 和 O_1 交换，保留新的簇类，否则，保留原中心点和聚类。重复此步骤直到 k 个中心点不再变化。

2.3 实验过程

2.3.1 特征处理

由于数据集中的特征是 22 维的，所以实验分别通过选择以下方法分别对特征进行降维处理，与不降维的数据进行结果比较和分析。

去除第一列特征 通过观察和计算，第一列特征的数据 97% 都是 1，而剩下 3% 的数据从 0 到 1 分布，因此可以判断第一列特征存在大量冗余，可以在聚类时将第一列去掉，提高实验结果的速度和精度。

皮尔森相关系数 皮尔森相关系数是一种线性相关系数。皮尔森相关系数是用来反映两个变量线性相关程度的统计量。相关系数用 r 表示，其中 n 为样本量，分别为两个变量的观测值和均值。r 描述的是两个变量间线性相关强弱的程度。r 的绝对值越大表明相关性越强。计算公式如下：

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}$$

数据集中 22 个特征和标签的皮尔森相关系数计算出来之后，第一、二、三、五、十、十八列特征与标签的相关系数绝对值小于 0.1，因此可以将它们从数据集中去除，作为另外一种情况来进行比较。

主成分分析 主成分分析（Principal Component Analysis）是一种在尽可能减少信息损失的情况下找到某种方式降低数据的维度的方法。通常来说，我们期望得到的结果，是把原始数据的特征空间（n 个 d 维样本）投影到一个小一点的子空间里去，并尽可能表达的很好（就是说损失信息最少）。并且为了方便地对结果进行可视化，所以需要使用 PCA 降维把 22 维特征数据降为 2 维特征数据。

2.3.2 距离度量

为了比较不同距离度量方式下，聚类结果有何不同，从而得到在该数据集下最优的距离度量方式，选择了以下两种方式进行比较：

欧几里得度量 欧几里得度量（Euclidean Metric）是一个通常采用的距离定义，指在 m 维空间中两个点之间的真实距离，或者向量的自然长度（即该点到原点的距离）。在二维和三维空间中的欧氏距离就是两点之间的实际距离。计算公式如下：

$$d(x,y) = \sqrt{(x_1-y_1)^2 + (x_2-y_2)^2 + (x_3-y_3)^2 + \cdots + (x_n-y_n)^2}$$

余弦相似度 余弦相似度通过测量两个向量的夹角的余弦值来度量它们之间的相似性。计算公式如下：

$$similarity = \cos(\theta) = \frac{A \cdot B}{||A|| ||B||}$$

2.4 结果与分析

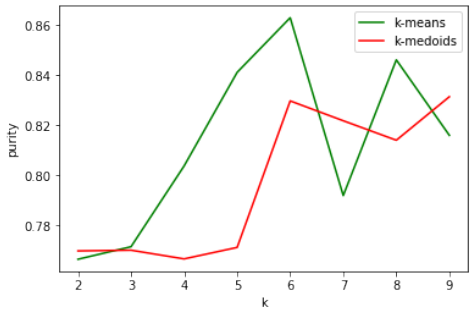
聚类所用的评价指标为 $purity$ 和 f_score，计算公式如下：

$$purity(\Omega, C) = \frac{1}{N} \sum_k max_j |\omega_k \cap c_j|$$

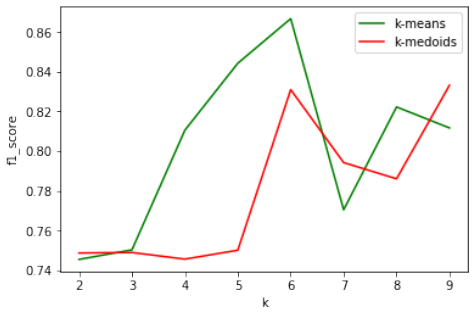
$$F_\beta = \frac{(\beta^2+1)PR}{\beta^2 P + R}, P = \frac{TP}{TP+FP}, R = \frac{TP}{TP+FN}$$

2.4.1 聚类方法比较

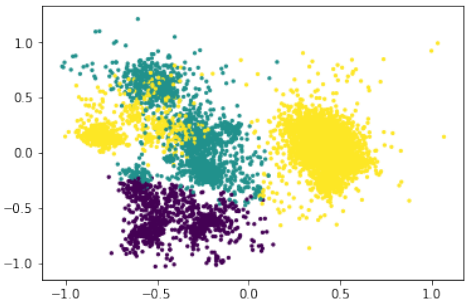
下图展示了利用 k-means 算法和 k-medoids 在不进行降维处理时，purity 和 f-score 的结果比较



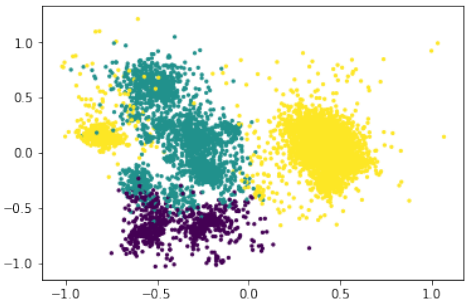
(a) purity



(b) f-score



(c) cluster1,k=6



(d) cluster2,k=6

从结果来看，k-means 算法表现效果稍微优于 k-medoids 算法的表现效果。当 k 从 2 变化到 9 时，k-means 算法的 purity 在 $[0.76, 0.86]$ 之间波动，f-score 在 $[0.74, 0.87]$ 之间波动；k-medoids 算法的 purity 在 $[0.76, 0.83]$ 之间波动，f-score 在 $[0.74, 0.83]$ 之间波动。当 $k < 6$ 时，k-means 的 purity 和 f-score 都高于 k-medoids 算法，但是当 k 继续增长时，指标会下降。除此之外，k-medoids 运行时间比 k-means 运行时间多了 30s，所以 k-means 计算效率更高。

使用 PCA 降维之后，可以观察到聚类的效果（选取了 k=6）。二者的聚类效果大体上相同，但是在细节上 k-means 可以将更多的黄色标签找到，符合原始数据分布。而 k-medoids 算法会将混在蓝色中的黄色标签判断成蓝色标签。

本次实验存在如下两个问题：

1. 取不同 k 值的时候，purity 和 f-score 波动较大；
2. 聚类结果只有三类，但是真实数据中有四种标签；

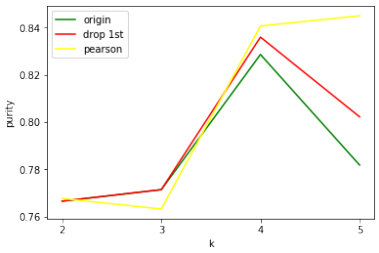
经分析，问题一的原因可能是 k-means 算法和 k-medoids 算法在选取初始点时是随机选取的，因此会像在算法描述时说的那样，每次跑的结果都会有较大的变动。解

决方法可以用在对一个 k 进行实验时，可以跑多次，将结果取平均值来降低初始点对聚类结果的扰动。

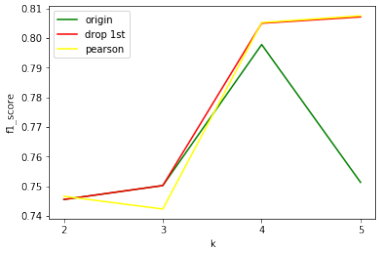
问题一的原因是 Bufonidae 类的数据只有 68 条，占所有数据的 0.9%，数据量实在是太少了，所以无法聚出 Bufonidae 类。解决方法可以是通过复制、再采样等方式，增多 Bufonidae 类的数量，在数据量占的比重稍微大一点的时候，结果会好很多。但是和同学讨论之后，认为聚类是无监督学习，在聚类时候是没有标签信息的，在实际生活的很多应用场景中我们也拿不到真实的标签数据，所以手动增添数据的方式我们认为不是特别科学。

2.4.2 特征处理方式比较

比较处理方式时，由于 k-means 算法效果较好，所以选取 k-means 算法，比较了去除第一列特征、使用皮尔森相关系数去除不相关的特征这两种特征处理方法和原始方法之间的对比。



(e) purity



(f) f-score

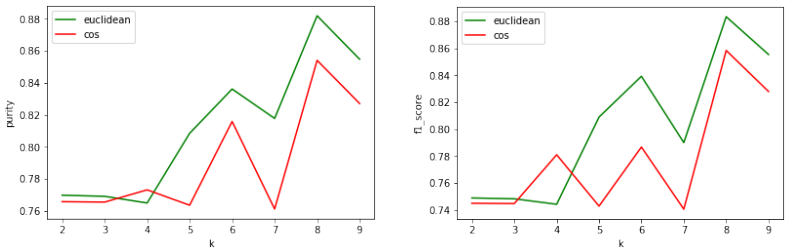
图中黄色折线是计算皮尔森相关系数之后，去除了第一、二、三、五、十、十八列特征后聚类的结果，红色折线是去除了第一列后聚类的结果，绿色折线是不经任何处理的聚类结果。

可以看出经过筛选掉与标签不太相关的列之后，聚类效果在总体上提高了，并且运行速度也稍微加快了一点。去除第一列之后聚类效果同样比不经任何处理时聚类效果好。

2.4.3 距离度量方式比较

使用 k-medoids 算法，不对特征进行降维处理，使用欧氏距离和余弦相似度的实验结果如下：

使用 k-medoids 算法在 k 从 2 取到 9 的时候，当距离度量方式选择欧氏距离时，算法的聚类效果比余弦相似度的结果更好。



(g) purity (h) f-score

3 总结

3.1 任务一

分类任务中第一步是数据预处理。我们首先对缺失值用自己实现的随机森林进行了填补；接下来对类别属性进行了数值化编码；然后是特征筛选部分，用 T 检验，MIC，RFE 和随机森林进行了特征排名与分析；最后用 SMOTE 算法来生成正样本处理数据不平衡问题。

第二步是分类器实现。我们实现了朴素贝叶斯，决策树 C4.5 和随机森林，三种算法都可以同时处理离散属性和连续属性。在决策树中我们还实现了后剪枝过程，可以有效地避免模型过拟合问题，在随机森林中我们也提供了对不平衡样本问题的改进的 Bagging 抽样方式。

最后是模型测试与结果分析部分。我们使用了常用的大多数的分类评价指标来分析模型的效果，包括准确率，敏感性，特异性，F1-Score，MCC 和 AUC 等。通过实验结果的对比，我们发现随机森林模型整体上效果超过朴素贝叶斯模型和决策树；决策树在后剪枝后树模型大小明显降低且各个分类指标值都有提升；进行平衡采样的随机森林在正样本上准确率明显提升了，且得到了所有模型中最高的 F1-Score 和 MCC；朴素贝叶斯模型虽然整体准确率较低，但是在正样本上有最高的识别准确率。

3.2 任务二

经过对 k-means 和 k-medoids 两种算法的实现和测试对比，我们比较深入地了解了两种算法的特点和各自的优缺点。k-means 的质心是各个样本点的平均，可能是样本点中不存在的点。k-medoids 的质心一定是某个样本点的值。虽然 k-medoids 的运行速度较慢，计算质心的步骤时间复杂度是 $O(n^2)$，因为必须计算任意两点之间的距离，而 k-means 只需平均即可。但 k-medoids 对噪声鲁棒性比较好。例：当一个 cluster 样本点只有少数几个，如 $(1,1),(1,2),(2,1),(100,100)$。其中 $(100,100)$ 是噪声。如果按照 k-means 质心大致会处在 $(1,1),(100,100)$ 中间，这显然不是我们想要的。这时 k-medoids 就可以避免这种情况，它会在 $(1,1),(1,2),(2,1),(100,100)$ 中选出一个样本点使 cluster 的绝对误差最小，计算可知一定会在前三个点中选取。

由于是第一次接触聚类的任务，所以一开始不太明白要怎么对特征进行处理，也

不太明白如何选择特征向量之间的距离度量方式。但是经过这次实验，我们对这些内容都有了一定的掌握。

总之，这次实验让我们比之前更深入地理解了常见的特征筛选方法，数据不平衡问题的影响与处理方式，以及数据缺失值的处理方式，而且在分类聚类算法的实现过程与效果测试对比中也对各个算法的特点有了更透彻的认识。

参考文献

[1] 周志华. 机器学习. Qing hua da xue chu ban she, 2016.

[2] Nina Zhou and Lipo Wang. A modified t-test feature selection method and its application on the hapmap genotype data. *Genomics, proteomics & bioinformatics*, 5(3):242–249, 2007.

[3] Huan Liu and Rudy Setiono. Chi2: Feature selection and discretization of numeric attributes. In *Tools with artificial intelligence, 1995. proceedings., seventh international conference on*, pages 388–391. IEEE, 1995.

[4] Hanchuan Peng, Fuhui Long, and Chris Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on pattern analysis and machine intelligence*, 27(8):1226–1238, 2005.

[5] Chen Lin, T Miller, D Dligach, RM Plenge, EW Karlson, and G Savova. Maximal information coefficient for feature selection for clinical document classification. In *ICML Workshop on Machine Learning for Clinical Data. Edingburgh, UK*, 2012.

[6] Stjepan Oreski and Goran Oreski. Genetic algorithm-based heuristic for feature selection in credit risk assessment. *Expert systems with applications*, 41(4):2052–2064, 2014.

[7] Cong Jin, Shu-Wei Jin, and Li-Na Qin. Attribute selection method based on a hybrid bpnn and pso algorithms. *Applied Soft Computing*, 12(8):2147–2155, 2012.

[8] C. Lazar, J. Taminau, S. Meganck, D. Steenhoff, A. Coletta, C. Molter, V. de Schaetzen, R. Duque, H. Bersini, and A. Nowe. A survey on filter techniques for feature selection in gene expression microarray analysis. *Ieee-Acm Transactions on Computational Biology and Bioinformatics*, 9(4):1106–1119, 2012.

[9] Yvan Saeys, Iñaki Inza, and Pedro Larrañaga. A review of feature selection techniques in bioinformatics. *bioinformatics*, 23(19):2507–2517, 2007.

[10] R. A. Fisher. *Statistical Methods for Research Workers*. Oliver and Boyd, 1958.

[11] David N Reshef, Yakir A Reshef, Hilary K Finucane, Sharon R Grossman, Gilean McVean, Peter J Turnbaugh, Eric S Lander, Michael Mitzenmacher, and Pardis C Sabeti. Detecting novel associations in large data sets. *science*, 334(6062):1518–1524, 2011.

[12] Ruiquan Ge, Manli Zhou, Youxi Luo, Qinghan Meng, Guoqin Mai, Dongli Ma, Guoqing Wang, and Fengfeng Zhou. Mctwo: a two-step feature selection algorithm based on maximal information coefficient. *BMC bioinformatics*, 17(1):142, 2016.

[13] Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1):389–422, 2002.

[14] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16(1):321–357, 2002.

[15] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.