Project 2 实验报告

邱圆辉
清华大学软件学院
2017013591
qiuyh0924@163.com

概要

本文为人工智能导论 project 2 实验报告。

关键词

有监督学习，无监督学习，聚类分析

1. 简介

本次实验分为两个子实验。第一个实验实现了一个银行精准营销解决方案，可以根据给定的银行营销数据，构建出三种不同的分类模型，预测客户是否会购买银行产品。第二个实验实现了青蛙叫声的聚类分析，可以根据给定的青蛙叫声的 MFCC 特征数据，对不同科的青蛙进行聚类分析。

2. 银行精准营销解决方案

2.1 模型

此次实验中共实现了三种不同的分类器，分别为：

2.1.1 朴素贝叶斯分类器

基于 sklearn 的 *CalibratedClassifierCV* 实现

2.1.2 决策树分类器

基于 sklearn 的 *DecisionTreeClassifier* 实现

2.1.3 k 近邻分类器

自主实现的 k 近邻算法，仅调用了 numpy 库中关于矩阵计算的一些接口，未调用任何第三方 knn 的库

2.2 数据

本次实验中，对于给定的银行营销数据，共选择了两种特征组合进行模型的训练，分别为：

2.2.1 经预处理的原始特征组合

此特征组合在原始特征数据的基础上，进行了以下的预处理：

1. 去掉了原始数据中的 ID 字段，因其对实验结果不会产生任何影响。

2. 修改了原始数据中的 pdays 字段，因为-1 代表从未联系或很久未联系客户，因此将其中所有的-1 值均替换为了 999。

3. 修改了 job, education, contact, poutcome 这四个字段。将这四个字段中所有的 unknown 替换成了对应特征的众数，即该特征中出现次数最多的值。

4. 将经上述三个步骤修改后的数据进行了 *LabelEncode*，即将所有的类别型特征转换为数值型特征。比如，假设有一个 color 字段，其取值有“red”,“green”,“blue”，则将这三种不同的取值分别编码替换为 1, 2, 3。

2.2.2 SelectKBest 后的特征组合

实验中采取的第二种特征组合在第一种特征组合的基础上，调用了 sklearn.feature_selection 中的 *SelectKBest*，采用计算特征的卡方均值的计算方法，来选取部分最重要的特征进行模型的训练。

2.3 评价

本次实验中，由于只给定了训练数据集，未给定测试数据集，因此为了进行模型预测结果的评价，必须将给定训练数据集划分为训练集和测试集。

在每个分类器进行完对原始数据集的预处理后，都会调用 sklearn.model_selection 包中的 *train_test_split* 接口对数据集进行划分，随机将其中 6%的数据划分为测试集以便对模型的预测结果进行评估。具体评估方法包括以下两种：

2.3.1 accuracy_score

准确度是最直观，也是最简单的一种评价指标，其计算方式为分类正确的样本数占所有样本数的百分比，取值在 0-1 之间，越接近 0 代表模型分类结果越不准确，越接近 1 代表分类结果越准确。

然而其缺点在于无法反馈出分类器犯错的类型，无法提供改善分类器的经验和方向。

2.3.2 roc_auc_score

为了避免上述评估方法的缺点，在第二种评估方法中我采用了 sklearn.metrics 包中的 *roc_auc_score* 评估方法。此方法的评估核心依据为 *ROC* 曲线，即受试者工作特征曲线(*receiver operating characteristic*)，该曲线以 *TPR*(*True Positive Rate*)即真正率为纵坐标，以 *FPR*(*False Positive Rate*)即假正率为横坐标进行绘制。

这两个指标的计算公式分别为：

$$TPR = \frac{TP}{TP + FN} \quad （正样本预测数/正样本实际数）$$

$$TPR = \frac{TP}{TP + FN} \quad （被预测为正的负样本数/负样本实际数）$$

在 *ROC* 曲线中，纵坐标 *TPR* 代表了模型的敏感性，横坐标 *FPR* 代表了模型的特异性。

roc_auc_score 的计算方式为直接根据真实值和预测值，计算出 *AUC(Area Under Roccurve)*值，即 *ROC* 曲线下的面积，意义即为模型准确度。

2.4 结果

四种分类器在两种特征组合下的预测评估结果如下表，其中第二三列为第一种特征组合的评估结果，第四五列为第二种特征组合的评估结果：

分类器/指标	accuracy1	roc1	accuracy2	roc2
NaiveBayes	0.8822	0.8264	0.8822	0.8261
DecisionTree	0.8678	0.7096	0.8493	0.6580
KNN	0.8737	0.6065	0.8743	0.6093

从实验结果可以看出：

1. 两种特征组合对实验结果的影响不大。我觉得原因在于特征工程做的不到位。在网上查阅相关资料时看到有将几个原始特征进行组合的做法，以及将年龄模三十的做法。但由于时间原因我没有采取，希望之后能够进行尝试。

2. 在这三种分类器中，表现最好的是朴素贝叶斯分类器，k 近邻分类器次之，决策树分类器表现最差。个人认为原因在于朴素贝叶斯的方法是从统计学上进行相关计算的，有一定的数学依据，更加贴近于银行营销这个实际情况。而决策树只是简单的依次对各个特征进行分段的判断，但在这个案例中，客户是否会购买银行产品并不是简单的由单个或几个因素决定的，因此决策树分类器在分类的过程中会可能会出现欠拟合的情况，导致预测结果不理想。

3. 青蛙叫声聚类分析

3.1 模型

项目总共实现了两种不同的聚类器，分别为：

3.1.1 层次聚类器

基于 sklearn 的 *AgglomerativeClustering* 实现

3.1.2 k-means 算法聚类器

自主实现的 *k-means* 算法，未调用任何第三方的 *k-means* 接口，只在涉及矩阵计算时调用了 numpy 相关的接口。

为了尽量避免初始类中心的选择给聚类结果带来的不稳定性，在初始化 *k* 个类中心点时我采用了 *k-means++*的选择算法，具体思路为：

1. 先随机选择一个样本作为中心点之一

2. 计算每个样本与所有已选中心点之间的距离，计算这个距离与所有距离之和相除的结果，这个结果即为选择这个样本作为下一个中心点的概率

3. 不断重复 2 直至已选中心点的个数为 *k*

3.2 数据

实验中主要选择了两种特征组合进行效果对比，分别为：

3.2.1 经预处理的原始特征

此特征组合在原始特征的基础上，进行了如下的预处理：去除了 Family, Genus, Species, RecordID 四个字段，同时将 Family 字段作为一列单独保存了下来，便于后续的有标评估

3.2.2 SelectKBest 后的特征组合

此特征组合在第一种特征组合的基础上，调用了 sklearn.feature_selection 中的 *SelectKBest*，采用计算特征的卡方均值的计算方法，来选取部分最重要的特征进行模型的训练。

3.3 评价

无监督学习聚类结果的评价主要分为有标评价和无标评价，即是否有样本的真实类的数据。在对聚类结果进行评价时我选择了三种不同的评估算法，其中有一种无标评价方法和两种有标评价方法，分别为：

3.3.1 v_measure_score

有标评估。这种评价方法是聚类结果的同质性与完整性的调和平均，两者定义如下：

同质性(homogeneity)：每个集群只包含单个类的成员的程度

完整性(completeness)：一个真实的类中的所有成员都被聚类到同一个集群的程度

3.3.2 adjusted_rand_score

有标评估。调整兰德系数。假定实际的类别信息为 *C*，聚类结果为 *K*，*a* 表示在 *C* 与 *K* 中都是同类别的元素对数，*b* 为在 *C* 与 *K* 中都是不同类别的元素对数，则兰德指数定义为：

$$RI = \frac{a+b}{C_n^2}$$

其中，C_n^2表示数据集中可以组成的总元素对数。兰德指数的取值范围为[0, 1]，越接近 1 意味着越接近真实情况。然而当分类完全随机时，*RI* 并不一定会趋近于零，因此为了实现“在聚类完全随即时，指标应该接近零”，调整兰德指数被提出，定义为：

$$ARI = \frac{RI - E[RI]}{\max(RI) - E[RI]}$$

ARI 的取值范围为[-1, 1]，值越大意味着越接近真实情况。

3.3.3 silhouette_score

轮廓系数，无标评估。对于每个样本来说，假设 *a* 是它与同类别中其他样本的平均距离，*b* 是它与距离最近的不同类别中的样本的平均距离，则轮廓系数定义为：

$$s = \frac{(b-a)}{\max(a,b)}$$

而对一个样本集合来说，它的轮廓系数定义为所有样本的轮廓系数的平均值。

轮廓系数的取值范围为[-1, 1]，分数越高，代表着同类别样本距离越近且不同类别间距离越远，即聚类效果越好。

3.4 结果

两种分类器在两种不同的特征组合下的聚类结果如下表：同样第二三列代表第一种特征组合的聚类结果，最后两列代表第二种特征组合的聚类结果：

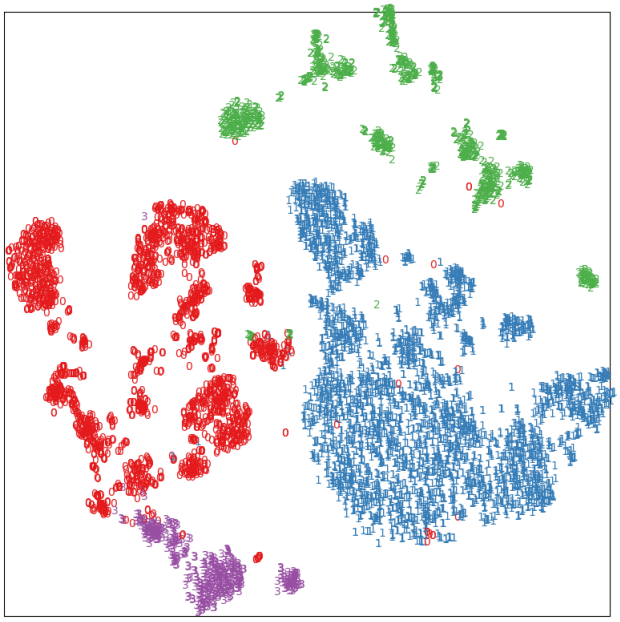
	vsc1	ari1	ss1	vsc2	ari2	ss2
Agglom	0.4533	0.4882	0.3767	0.4136	0.4723	0.4279
Kmeans	0.4278	0.5090	0.3233	0.4765	0.5131	0.4255

可以看出，当以调整兰德系数作为评估指标时，两种特征组合下聚类的结果差异较大，而当以其他两种指标为标准进行评估时，两者的差异不明显。

同时，整体上，sklearn 库自带的层次聚类方法的聚类效果要优于 *k-means* 的聚类效果，我认为这主要是 *k-means* 方法本身的缺陷（即初始中心点的选择会对聚类结果产生较大的影响）导致的，在之后会在初始中心点选择方面再尝试其他途径，如初始选择数倍于 *k* 的中心点，再通过类似层次聚类的方法不断合并类的方法等。

3.5 可视化

下图为层次聚类的可视化聚类效果图：



可以看到，被分到同一类的样本基本都集中在一起，只有个别样本远离了其类中心，整体的聚类效果还是不错的。

4. 其他说明

1. 由于我在网上查找 MFCC 相关资料的过程中，没有找到关于 MFCC 简单易懂的解释，要么解释的过于专业晦涩难以理解，要么对聚类实验中样本间距离的度量没有帮助，因此在第二题计算样本间距离时我只采用了简单的欧氏距离作为度量，但实验结果表明在本案例下欧氏距离的采用不失为一种合适的选择。

2. 银行程序的源代码见 *bank.py*，青蛙叫声的源代码见 *frog.py*，程序运行方式及相关命令行参数的说明可通过

python frog.py -h 或 python frog.py –help

命令进行查看（*bank.py* 同）

3. 程序在对聚类结果进行可视化时采用的是 t-SNE 方法，运行速度较慢，实测时间在 45 秒左右，烦请助教耐心等待。

4. 程序运行的所需依赖已经集成在 requriements.txt 文件中，可通过

pip install -r requirements.txt

命令一键安装所有依赖。

5. 实验收获及体会

此次实验，我的收获如下：

1. 了解了 sklearn, pandas, numpy 等第三方库的使用。具体包括上文介绍到的各种 sklearn 接口，pandas 中 *DataFrame* 和 *Series* 数据结构的使用以及 numpy 的各种科学计算方法等。

2. 对机器学习的相关方法有了进一步的理解，特别是 *knn* 算法和 *k-means* 算法。同时，在 sklearn 的官方文档上，我还了解了我用到的其他分类器和聚类器背后的实现原理，加深了自己对课程内容的理解。

然而不得不说，这次实验中调用的第三方库太多，主要是 sklearn 各种机器学习方法的调用，让我觉得自己并没有完全掌握这些学习方法，只是对其实现原理有了一个大概的了解，仍然只停留在理论层面上，而没有落到实际的代码实现层面。因此，在之后有关机器学习这部分内容的学习中，我会进一步尝试自己用实现其他各种分类器，巩固课上学到的知识，提高自己的 coding 能力。