《智能信息网络》课程

案例分析与实践报告

(第三次)

题目：基于决策树算法的网络数据智能分类探究**实践报告**

场景：网络智能部署

姓名：

班级：

学号：

日期：2022 年 6 月

1、研究背景与问题描述

1.1 研究背景

随着互联网技术的不断发展，网络规模逐渐增大，涌现出各种新的网络应用。这些应用所产生的流量增长迅速，对网络带宽及网络安全的管理形成严重挑战。互联网中各种应用类型的流量都有自己的统计特征，通过分析这些特征可以区分其应用类型，帮助网络管理员对网络进行合理的优化。

1.2 问题描述

有来自同一数据集的 2898 条带分类的数据，且数据分布不平衡（长尾分布），和 400 条未分类的数据，需要利用已分类的 2898 条数据对未分类的 400 条数据进行分类。每条数据有九个属性，具体的属性如表1所示，所提供的原始训练集中每个类的数量如表2所示。

序号	英文描述	中文描述
1	Server Port	服务器端口号
2	Client Port	客户端端口号
3	q1_IAT	IP 数据包间隔到达时间的较小四分位数
4	Med_IAT	IP 数据包间隔时间到达时间的中位数
5	Min_IAT	IP 数据包到达时间的最小值
6	q3_IAT	IP 数据包间隔到达时间的较大四分位数
7	Max_IAT	IP 数据包间隔到达时间的最大值
8	Var_IAT	IP 数据包间隔到达时间的方差
9	Mean_IAT	IP 数据包间隔到达时间的平均值
10	Classes	流量所属类型

表 1　数据属性的介绍

类别	训练集中样本数量
FTP-DATA	878
MAIL	919
WWW	801
P2P	300
整体	2898

表 2　原始训练集中每个类的数量

所有数据共有 P2P、FTP-DATA、MAIL、WWW 四种流量类型。在不依靠端口号属性的前提下，为网络流量提供高效分类算法，并在 400 条未分类数据中测试分类性能。

2、总体思路与方案论证

2.1 流量分类

2.1.1 神经网络

策略：建立前馈神经网络网络，输入神经元数量为属性数，输出神经元数量为类别数，利用数据集训练神经元权重。

优势：预测准确度较高。

劣势：需要大量计算资源长时间训练。

2.1.2 决策树

策略：计算数据集中每个属性的信息增益率，逐代选出具有最大的信息增益率的属性分裂创造分支，生成决策树并剪枝。

优势：训练速度快，可解释性高。

劣势：容易过拟合。

2.2 样本不均衡（长尾分布）

样本（类别）样本不平衡（class-imbalance）指的是分类任务中不同类别的训练样例数目差别很大的情况，也被称为长尾分布（Long-Tailed Distribution）。样本不均衡带来的根本影响是模型会学习到训练集中样本比例的这种先验性信息，以以至于实际预测时就会对多数类别有侧重（可能导致多数类精度更好，而少数类比较差）。

在类别不均衡情况下，分类边界会偏向“侵占”少数类的区域。更重要的一点，这会影响模型学习更本质的特征，影响模型的鲁棒性。解决不均衡解决方法可以归结为：通过某种方法使得不同类别的样本对于模型学习中的损失函数（或梯度）贡献是比较均衡的。以消除模型对不同类别的偏向性，学习到更为本质的特征。从数据样本、模型算法、目标（损失）函数、评估指标等方面有相应的解决方法。

2.2.1 欠采样与过采样

最直接的处理方式就是样本数量的调整，常用的方法有：

- 欠采样：减少多数类的数量（如随机欠采样、NearMiss、ENN[1] 等）。
- 过采样：尽量多地增加少数类的的样本数量（如随机过采样等），以达到类别间数目均衡。
- 结合两者做混合采样（如 SMOTE[2] +ENN）。

2.2.2 损失函数的层面

损失函数层面主流的方法也就是常用的代价敏感学习（cost-sensitive），为不同的分类错误给予不同惩罚力度（权重），在调节类别平衡的同时，也不会增加计算复杂度。

- 最常用是 sklearn 中的`'class weight'`方法，`'class weight'`可以为不同类别的样本提供不同的权重（少数类有更高的权重），从而模型可以平衡各类别的学习。如下面的代码通过为少数类做更高的权重，以避免决策偏重多数类的现象（类别权重除了设定为 balanced，还可以作为一个超参搜索）。

代码 1: sklearn 中的损失函数重加权方法

```
clf2 = LogisticRegression(class_weight={0:1,1:10})  # 代价敏感学习
```

- OHEM[3]（Online Hard Example Mining）算法的核心是选择一些 hard examples（多样性和高损失的样本）作为训练的样本，针对性地改善模型学习效果。对于数据的类别不平衡问题，OHEM 的针对性更强。

- Focal Loss[4]的核心思想是在交叉熵损失函数（CE）的基础上增加了类别的不同权重以及困难（高损失）样本的权重以改善模型学习效果。

2.2.3 模型层面

模型方面主要是选择一些对不均衡比较不敏感的模型，比如，对比逻辑回归模型（逻辑回归学习的是全量训练样本的最小损失，自然会比较偏向去减少多数类样本造成的损失），决策树在不平衡数据上面表现相对好一些[5]，树模型是按照增益递归地划分数据，划分过程考虑的是局部的增益，全局样本是不均衡，局部空间就不一定，所以比较不敏感一些（但还是会有偏向性）。

3、关键技术研究与实现

3.1 数据处理

3.1.1 抑制样本不均衡

针对样本数量不均衡问题：采用欠采样方法，随机抛弃样本数量多的类别，使各类别样本数量相近，在 sklearn 中有相应的 imblearn 库可以实现相关方法。

随机过采样是在少数类 S_{min} 中随机选择一些样本，然后通过复制所选择的样本生成样本集 E，将它们添加到 S_{min} 中来扩大原始数据集从而得到新的少数类集合 $S_{new-min}$。新的数据集 $S_{new-min} = S_{min} + E$，具体代码实现如下：

代码 2: 随机过采样代码

```
ros = RandomOverSampler(random_state = 10)
X_train, y = ros.fit_resample(X_train, y)
```

SMOTE 算法是对随机过采样方法的一个改进算法，由于随机过采样方法是直接对少数类进行重采样，会使训练集中有很多重复的样本，容易造成产生模型过拟合问题。而 SMOTE 算法的基本思想是对每个少数类样本 x_i，从它的最近邻中随机选择一个样本 $\hat{x}_i$（$\hat{x}_i$ 是少数类中的一个样本），然后在 x_i 和 $\hat{x}_i$ 之间的连线上随机选择一点作为新合成的少数类样本。

SMOTE 算法合成新少数类样本的算法描述如下：

- 对于少数类中的每一个样本 x_i，以欧氏距离为标准计算它到少数类样本集 S_{min} 中所有样本的距离，得到其 k 近邻。

- 根据样本不平衡比例设置一个采样比例以确定采样倍率 N，对于每一个少数类样本 x_i，从其 k 近邻中随机选择若干个样本，假设选择的是 $\hat{x}_i$。

- 对于每一个随机选出来的近邻 $\hat{x}_i$，分别与 x_i 按照 $x_{new} = x_i + \text{rand}(0,1) \times (\hat{x}_i - x_i)$ 构建新的样本。

SMOTE 算法摒弃了随机过采样复制样本的做法，可以防止随机过采样中容易过拟合的问题，可以提高分类器的性能。

代码 3: SMOTE 过采样代码

```
X_train, y = SMOTE().fit_resample(X_train, y)
```

3.1.2 标准化与归一化

针对均值方差较大问题：将样本各属性进行特征缩放，提升模型收敛速度，提升精度。

（1）数据标准化：将各属性数值转化为正态分布。

$$\hat{x} = \frac{x - \mu}{\sigma} \tag{1}$$

其中 x 为原始样本，$\hat{x}$ 为标准化样本，μ 为样本均值，σ 为样本标准差。

（2）数据归一化：将各属性数值转为 $[0,1]$ 区间。

$$\hat{x} = \frac{x - \min}{\max - \min} \tag{2}$$

其中 x 为原始样本，$\hat{x}$ 归一化样本，min 为样本最小值，max 为样本最大值，标准化与归一化的具体代码如下所示。

代码 4: 标准化与归一化代码

```
# 数据标准化
X_train = preprocessing.scale(X_train)

# 数据归一化
scaler = MinMaxScaler()
X_train = scaler.fit_transform(X_train)
```

3.2 数据集划分

已知类别的 2898 条数据分割为训练集和验证集，如 7：3，9：1 等。

3.2.1 k 折交叉验证

- 交叉验证目的：尝试利用不同的训练集/验证集划分来对模型进行训练和验证，解决单独测试结果过于片面以及训练数据不足的问题。
- k 折交叉验证方案：将数据集分为均等不相交的 k 份，依次取其中一份作为验证集，其余为训练集。最终结果为: 对所有划分下的结果取均值。

3.2.2 决策树剪枝

在不加限制的情况下，决策树往往会过拟合。为提升决策树泛化能力，需对生成的决策树剪枝。

sklearn 中的剪枝策略：(可多次尝试不同剪枝策略超参数取最优)

- `max_depth`：限制树的最大深度，超过设定深度的树枝全部剪掉。
- `min_samples_leaf`：分枝会朝着满足每个子节点都包含`min_samples_leaf`个样本的方向去发生。
- `min_samples_split`：一个节点必须要包含至少`min_samples_split`个训练样本，这个节点才允许被分枝。

3.3 决策树分类算法

决策树模型的特点在于可解释性，它可以看作将数据自顶向下进行划分，然后通过回答一系列问题来做出决策的方法。换句话说，基于训练数据集的特征，决策树模型通过一系列的问题来推断样本的类标。

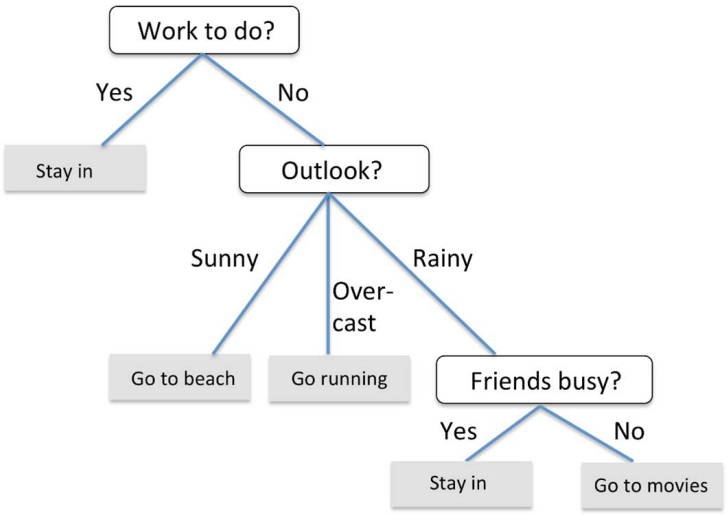

图 1　决策树示意图

使用决策树算法，需要从树根开始，基于可获得最大信息增益（Information Gain，IG）的特征来对数据进行划分。通过迭代处理，在每个子节点上重复此划分过程，直到叶子节点。这意味着在每一个叶子节点处，所有样本都属于同一类别。在实际应用中，这可能会导致生成一颗深度很大且拥有众多节点的树，容易产生过拟合问题，一般通过“剪枝”来限定树的最大深度。

为了在可获得最大信息增益的特征处进行节点划分，需要定义一个可通过树学算法进行优化的目标函数。该目标函数能够在每次划分时实现对信息增益的最大化，定义为：

$$IG\left(D_p, f\right) = I\left(D_p\right) - \sum_{j=1}^{m} \frac{N_j}{N_p} I\left(D_j\right) \tag{3}$$

其中，f 将要进行划分的特征，D_p 与 D_j 分别为父节点和第 j 个子节点，I 为不纯度衡量标准，N_p 为父节点中样本的数量，N_j 为子结点中样本的数量。可见信息增益是父节点的不纯度与所有子节点不纯度总和之差，即子节点的不纯度越低，信息增益越大。

出于简化和缩小组合搜索空间的考虑，大多数库中实现的都是二叉决策树，即每个父节点被划分为两个子节点：D_{left} 和 D_{right}，目标函数改写为：

$$IG\left(D_p, f\right) = 1\left(D_p\right) - \frac{N_{\text{left}}}{N_p} I\left(D_{\text{left}}\right) - \frac{N_{\text{right}}}{N_p} I\left(D_{\text{right}}\right) \tag{4}$$

4、结果分析与结论

4.1 实验设置

我们的实验在 Ubuntu 16.04.7 LTS 环境中进行，显卡 GPU 配置为 NVIDIA TITAN Xp×4, 编程语言 Python, 代码框架为 Sklearn。具体的代码将在 GitHub 上开源[1]。本实验中识别性能的评价方法采用准确率（分类正确的数据/测试集总数据）的指标。设 $P_i(1 \leq i \leq 4)$ 是单类型分类准确度，P_s 为整体分类准确度。T_i 为属于 i 类且被分类为 i 类的数量，R_i 为实际类型为 i 的类别的总数目，则有：

- 单类型分类准确度计算公式：

$$P_i = \frac{T_i}{R_i} \tag{5}$$

- 整体分类准确度计算公式：

$$P_S = \frac{\sum_{i=1}^{4} T_i}{\sum_{j=1}^{4} R_j} \tag{6}$$

4.2 实验结果

baseline 的参数设置如下，这里采用了 k 折交叉验证，超参数 k 设置为 15，并且将训练集和验证集划分为 9:1 的比例，决策树的参数设置如下所示。

代码 5: baseline 参数设置

```
1 tree_model = tree.DecisionTreeClassifier(criterion='gini')
```

baseline 的决策树模型在原始数据集经过 k 折交叉验证划分成的训练集上测试的结果如表3所示，我们发现，样本不均衡带来的根本影响是模型会学习到训练集中样本比例的这种先验性信息，以至于实际预测时多数类精度更好，而少数类比较差，如数量较多的 FTP-DATA 类别分类的准确率有 97.33%，而数量较少的 P2P 类别分类的准确率仅仅有 78.95%。

类别	训练集中样本数量	分类准确率（%）
FTP-DATA	878	97.33
MAIL	919	90.18
WWW	801	84.00
P2P	300	78.95
整体	2898	90.25

表 3　原始测试集测试结果

于是为了抑制长尾分布，我们在数据读入后对数据进行了随机过采样与 SMOTE 算法的方法增加样本数量，平衡每一个类样本的分布。可以看到经过过采样操作后训练集中每个类的数量相对平衡，并且分类准确率上从 90.25% 提升到了 92.25%，具体每一个类的准确率也有大幅度提升，以 P2P 类别为例，分类准确率从原来的 78.95% 提升到了 84.21%。

[1] https://github.com/realmadridchenwentao/Intelligent-Network

类别	训练集中样本数量（过采样后/划分训练集验证集后）	分类准确率（%）
FTP-DATA	919/815	96.67
MAIL	919/837	93.75
WWW	919/837	87.00
P2P	919/819	84.21
整体	2898/3308	92.25

表 4　数据过采样后测试集测试结果（按照 9:1 划分训练集与验证集后）

我们对决策树方法的参数进行消融实验，这里我们根据实验平台的数据制作了测试集的标签，方便进行消融实验，寻找最优的参数，具体的测试集标签将会附在代码中。改变超参数并进行测试，结果如表5所示：

参数设置	分类准确率（%）
gini+max_depth=None+min_samples_leaf=1+min_samples_split=2 (baseline)	92.25
gini+max_depth=None+min_samples_leaf=1+min_samples_split=10	91.50
gini+max_depth=None+min_samples_leaf=15+min_samples_split=10	90.25
gini+max_depth=None+min_samples_leaf=5+min_samples_split=10	92.25
gini+max_depth=10+min_samples_leaf=5+min_samples_split=10	91.25
gini+max_depth=100+min_samples_leaf=5+min_samples_split=10	**92.75**
gini+max_depth=50+min_samples_leaf=5+min_samples_split=10	**92.75**
entropy+max_depth=10+min_samples_leaf=5+min_samples_split=10	90.00

表 5　决策树参数的消融实验

我们发现在参数为 gini+max_depth=50+min_samples_leaf=5+min_samples_split=10 时，分类准确率较高达到了 92.75%，于是我们引入 graphviz 模块用来导出决策树结构图，具体代码如下：

代码 6: 使用 graphviz 模块用来导出决策树结构图

```
fn = ['q1_IAT', 'Med_IAT', 'Min_IAT', 'q3_IAT', 'Max_IAT', 'Var_IAT',
    'Mean_IAT']
cn = ['WWW', 'MAIL', 'P2P', 'FTP-DATA']

tree.export_graphviz(clf,
                     out_file="tree.dot",
                     feature_names = fn,
                     class_names = cn,
```

```
8                     filled = True)
```

导出的决策树示意图如图2所示，具体图片将附在提交的文件夹中：

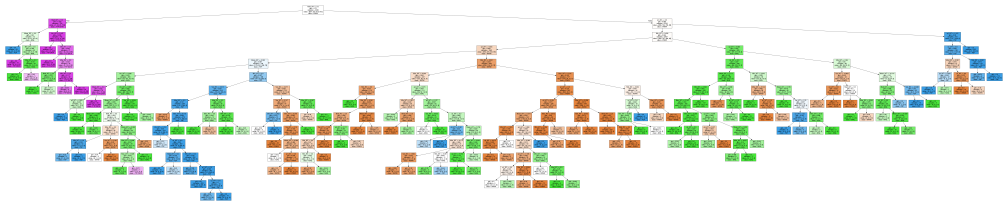

图 2　使用 graphviz 绘制的决策树结构图

这里我们也尝试了一些别的方法，随机森林[6]具备好的分类能力、可扩展性、易用性强等特点。随机森林可以视为多颗决策树的集成。集成学习的基本理念是将弱分类器集成为鲁棒性更强的强分类器，能力更强，有更好的泛化误差，不易产生过拟合现象。随机森林主要步骤有：

- 使用 bootstrap 抽样方式随机选择 n 个样本用于训练（从训练集中随机可重复地选择 n 个样本）。
- 使用第一步选定的样本构造一棵决策树，节点划分规则如下：
- 不重复地随机选择 d 个特征；
- 根据目标函数的要求，如最大化信息增益，使用选定的特征对节点进行划分；
- 重复上述过程 1 至 2000 次；
- 汇总每棵决策树的类标进行多数投票（majority vote）。

如表6所示，可以发现在如下参数设置时，结合过采样和 k 折交叉验证等操作，随机森林模型在测试集上的分类准确率达到了 95.25%，我们也发现在随机森林方法下，每个类的分类精度也有一定程度的提升，特别是 WWW 和 P2P 两个类的分类准确率相对决策树分类方法有相当大的提升，准确率都达到了 90% 以上（参数设置为 n_estimators=100+criterion=gini+max_depth=100）。

类别	训练集中样本数量（过采样后/划分训练集验证集后）	分类准确率（%）
FTP-DATA	919/815	96.67
MAIL	919/837	97.32
WWW	919/837	92.00
P2P	919/819	92.11
整体	2898/3308	95.25

表 6　数据过采样后使用随机森林分类测试集测试结果

参数设置	分类准确率（%）
DecisionTreeClassifier (baseline)	**92.25**
KNeighborsClassifier	84.25
SVC（支持向量机）	45.25
LogisticRegression	63.00
SGDClassifier	22.25

表 7　数据过采样后使用其他机器学习方法（默认参数）分类测试集测试结果

如表7所示，与其他常用机器学习方法（默认参数）进行分类测试，我们发现我们采用的决策树算法具有非常高的准确率，同时相比卷积神经网络方法训练速度快，可解释性高。

未来本项目及相关方法在智能信息网络中有着非常广泛的应用，由于网络相关的数据数量庞大，对于数据的标注是及其昂贵并且耗时的[7]，如何考虑使用半监督/自监督学习的方法达到有监督学习一样的效果是一个很好的研究方向。此外，随着深度学习方法，特别是注意力机制和 Transformer[8] 的大范围应用，如何利用 Transformer 的全局时空特性在网络流量数据分类上也是未来的研究方向，由于 Transformer 与卷积神经网络相比没有归纳偏置性，在网络流量数据这种大规模数据集上一定会有非常好的效果。此外，实际模型部署优化等课题也是未来的研究方向，期待随着知识的不断完善在相关领域开展进一步的研究。

5、课程学习收获与体会建议

- 通过智能信息网络课程的学习，我们通过概念理解了光网络，移动网络，边缘计算等基础网络。此外在课程中我们通过方法培养了基础的思维，通过算法掌握基础智能，通过案例掌握典型的应用，对实际的案例有了自己独特的体会。课程通过探究式，启发式，案例驱动实现了从被动学习到主动求知的转变，通过研究探讨的过程掌握新的思路以及许多方法的本质。三次大作业分别提高了我们对于网络需求的理解，提高思维思辨能力，更锻炼了解决问题了能力。整个课程让我对于“通信-计算-协同”协同为基础，“大数据”分析为驱动，以“人工智能”赋能为特征的智能信息网络有了更加深入的理解，希望可以有更多的机会和老师交流讨论智能信息网络的相关应用，也感谢老师和助教这一个学期以来的辛苦付出！

- 建议：希望后面的课程中可以对第三次大作业的平台进行进一步的改进，参考学校的《C/C++ 程序设计》和《算法设计与分析》等课程采用的 Online Judge 平台，可以学生直接提交代码，平台的编译器运行后将学生提交的代码结果与真实数据标签进行比较直接给出结果，学生可以直接在平台上进行实验的全流程操作，现阶段也有 Jupyter Lab 等在线编程平台，有条件的情况下可以在课程大作业中引入。

参考文献

[1] BECKMANN M, EBECKEN N F, DE LIMA B S P, et al. A knn undersampling approach for data balancing[J]. Journal of Intelligent Learning Systems and Applications,

2015, 7(04): 104. 2.2.1

[2] CHAWLA N V, BOWYER K W, HALL L O, et al. Smote: synthetic minority oversampling technique[J]. Journal of artificial intelligence research, 2002, 16: 321-357. 2.2.1

[3] SHRIVASTAVA A, GUPTA A, GIRSHICK R. Training region-based object detectors with online hard example mining[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 761-769. 2.2.2

[4] LIN T Y, GOYAL P, GIRSHICK R, et al. Focal loss for dense object detection[C]// Proceedings of the IEEE international conference on computer vision. 2017: 2980-2988. 2.2.2

[5] YU L, ZHOU N. Survey of imbalanced data methodologies[J]. arXiv preprint arXiv:2104.02240, 2021. 2.2.3

[6] RIGATTI S J. Random forest[J]. Journal of Insurance Medicine, 2017, 47(1): 31-39. 4.2

[7] LI Z, CHEN W, WEI Z, et al. Semi-wtc: A practical semi-supervised framework for attack categorization through weight-task consistency[J]. arXiv preprint arXiv:2205.09669, 2022. 4.2

[8] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30. 4.2