目录

4 类别不平衡问题处理

4.1 类别不平衡问题

在现实的分类问题中, 可能存在因变量Y的分布很不均衡的情况. 例如, 天猫用户重复购买案例中, 因变量Y包含两个类别: 重复购买和未重复购买. 大部分新用户在双十一后的六个月内不会重复购买, 发生重复购买的比例约为6%. 这就形成了占比较高的多数类和占比较低的少数类, 成为类别不平衡.

然而, 类别不平衡现象并不必然导致类别不平衡问题. **只有当少数类发生误判具有比多数类发生误判更高的损失时, 才会产生类别不平衡问题.** 例如, 天猫用户重复购买案例中, 如果我们将实际上会发生重复购买的用户错误地预测为不会重复购买, 那么可能导致商家错失机会, 这样的损失可能远高于将实际不会重复购买的用户预测为会重复购买. 这就形成了类别不平衡问题.

注记4.1. 如果只是类别不平衡现象, 那么是不需要特别关注的. 只有当存在类别不平衡问题时, 才需要进行专门的处理.

处理类别不平衡问题, 一般需要考虑3个方面(Weiss, 2004):

(1) 损失矩阵(cost matrix), 即发生误判时的代价. 随着少数类发生误判相对损失的增大, 类别不平衡问题变得愈加严重, 需要有针对性地进行处理.

(2) 训练集规模(training set size) 和各类别样本数量(class prior). 例如, 当训练集规模大、少数类样本足够多时, 可以采用减少多数类样本的方法; 反之, 可以采用增加少数类样本的方法.

(3) 分类模型的决策边界(placement of decision boundary), 可理解为模型方法的特性. 例如, k近邻算法的决策边界仅受到局部数据点, 即近邻的影响, 从而对于类别不平衡问题具有较好的抵御能力, 因而当采用k近邻算法时, 可以不必作类别不平衡的数据预处理. 但是, 大部分的模型或算法(例如, 决策树中的CART算法以及以此为基础的集成学习方法)在建模过程中可能会受到类别不平衡问题的影响, 只是影响程度有所不同, 在数据预处理中可结合模型特性予以考虑.

对于类别不平衡问题, 我们可以在数据预处理阶段进行采样, 也可以在模型方法中予以专门的处理. 针对类别不平衡问题的采样方法包括3种:

(1) 欠采样(under-sampling或down-sampling): 从多数类样本中随机抽取一部分样本, 从而与少数类样本合并形成平衡的建模数据集.

(2) 过采样(over-sampling或up-sampling): 从少数类样本中抽取或合成比少数类样本数量更大的样本.

(3) 混合采样(hybrid sampling): 将欠采样与过采样混合使用.

4.2 欠采样: EasyEnsemble算法

欠采样方法是从多数类样本中随机抽取一部分样本, 从而与少数类样本合并形成平衡的建模数据集. 当训练集样本量较大、少数类样本已经充足时, 可以考虑使用欠采样. 欠采样的缺陷是会损失多数类样本的信息. 那么, 有什么办法尽量降低多数类样本信息的损失呢? 一些类别不平衡学习方法提供了很好的思路, 它们基于每一次欠采样的数据集建立模型, 并将这一过程重复多次, 最终将多次建模的结果进行集成. 这样, 可以在多次欠采样中使用不同的多数类样本信息, 从而减少信息损失.

本节介绍一种基于欠采样的类别不平衡学习方法——EasyEnsemble. 该方法由Liu et al. (2008) 提出, 其主要思路是: 首先进行欠采样, 即在多数类样本观测中进行随机采样, 样本量与少数类样本数量相同; 然后将入样的多数类样本观测与全体少数类样本观测合并构造为一个训练集, 在此基础上建立一个Adaboost模型. 独立地重复上述过程T次, 最终集成T个Adaboost模型.

EasyEnsemble是以Adaboost为个体学习器的Bagging集成, 综合利用了AdaBoost降低偏差和Bagging降低方差的能力. 由于Adaboost模型本身就

是集成学习模型, 因此EasyEnsemble可以看作为“集成的集成”. 它巧妙地将欠采样融入集成模型之中, 这种处理类别不平衡的思想值得借鉴.

4.3 过采样: SMOTE算法

过采样是从少数类样本中抽取或合成比少数类样本数量更大的样本. 最简单的方法是将少数类样本整体性地复制多次, 使得两类样本数量基本相当. 过采样方法的缺陷是对少数类样本可能存在过拟合风险. 为了克服这一缺陷, 通过人工生成少数类样本的方法被提出. 这类方法有多种产生少数类样本的策略, 例如, 在少数类样本的基础上加上随机扰动形成新的“少数类”样本或者将两个少数类样本观测的线性组合作为新的“少数类”样本.

本节介绍一种过采样方法——SMOTE算法. 它由Chawla et al. (2002)提出, 通过人工合成增加少数类样本.

对于定量变量, SMOTE算法产生1个新的少数类样本观测的步骤是:

第1步: 随机选择1个少数类样本观测$\boldsymbol{x}$，获得其k个近邻.

第2步: 从k个近邻中随机选取1个，记为$\boldsymbol{z}$.

第3步: 生成$(0,1)$中的随机数r，在$\boldsymbol{x}$与$\boldsymbol{z}$之间通过线性插值得到一个新的少数类样本观测$\boldsymbol{x}_{\text{new}}$:

$$\boldsymbol{x}_{\text{new}} = \boldsymbol{x} + r(\boldsymbol{z} - \boldsymbol{x}).$$

重复以上步骤N次，就可以获得N个合成的少数类样本观测. 需要注意, 各次重复合成新样本是独立的, 也就是说允许多次抽到相同的少数类样本观测, 以及相同的近邻.

若数据中同时含有定量和定性变量, 对于定量维度可按上述SMOTE算法合成新样本的相应维度; 对于定性维度, 可以取k个近邻在该维度上的众数作为合成样本的相应维度值, 而不需要线性插值, 故无需随机地抽取近邻. 这就是SMOTE-NC算法.

需要注意, SMOTE算法和SMOTE-NC算法中都需要找近邻, 因而涉及样本观测之间的距离运算. 为了保持各定量变量在距离运算中的影响程度一致, 应对定量变量作标准化处理. 当含有定性变量时, 若采用欧氏距离, SMOTE-NC算法的处理方法是: 首先计算所有定量变量标准差的中位数(记为Med). 对于某一个定性变量, 若2个样本观测在该定性变量上取值相同, 则二者差异为0; 若取值不同, 则二者差异为Med. 若已对所有定量变量作标准化处理, 则它们标准差的中位数$Med = 1$. 此时, 若2个样本观测在定性变量

上取值不同, 则二者差异为1. 这样的处理可以使得定性变量与定量变量在欧氏距离运算中的影响度大致相当.

例4.1. 基于golf数据集中5个Play="no"的少数类样本观测(如表4.1第2至5列所示), 按照SMOTE-NC算法产生1个新的少数类样本观测. 取近邻数量$k = 3$, 基于欧氏距离确定近邻. 可以看到, 数据集中含有2个定量维度, 即Temperature和Humidity, 二者的单位并不相同. 为了在距离运算中尺度一致, 需要对定量变量标准化, 结果如表4.1第6和7列所示.

对于Temperature和Humidity两个定量维度, 执行SMOTE算法, 步骤为:

第1步: 随机选择1个少数类样本观测$\boldsymbol{x}$，假定抽到第1个样本观测.

由于定量变量都已标准化, 标准差为1, 因此标准差的中位数$Med = 1$. 由此, 可计算样本观测之间的欧氏距离. 例如, 第1与第2个样本观测在变量Windy上取值不同, 而在变量Outlook上取值相同, 因此它们在这2个维度上的差异分别为1和0. 从而, 二者的欧氏距离为:

$$\sqrt{(1.47-0.76)^2+(-0.14-0.44)^2+1^2+0^2} \approx 1.36.$$

类似地，可得到第1个样本观测与第3至第5个样本观测之间的欧氏距离, 分别为3.60, 2.17和2.53. 可以看到, 第2、4、5个样本观测成为第1个样本观测的3个近邻.

第2步: 从3个近邻中随机选取1个, 例如, 选中第2个样本观测.

第3步: 生成$(0,1)$中的随机数r, 例如, $r = 0.6$. 在第1与第2个样本观测之间通过线性插值得到一个新的少数类样本观测, 它在变量Temperature上的取值为$85 + 0.6 \times (80 - 85) = 82$, 在变量Humidity上的取值为$85 + 0.6 \times (90 - 85) = 88$. 对于2个定性变量Windy和Outlook, 应取3个近邻在相应维度上的众数作为合成值, 因而取Windy为"True", Outlook为"Sunny".

SMOTE算法直观易懂，运算简单，其思想富有启发性。SMOTE算法也存在一些缺点, 例如:

(1) 合成的"少数类"样本可能与真实的少数类样本有较大差异, 影响模型效果.

(2) 如果抽中的少数类样本观测是有噪声的, 那么由SMOTE所合成的样本观测可能存在较大的噪声, 导致样本中的噪声数据增多.

表 4.1: golf数据集中Play="no"的5个样本观测

序号	Temperature	Humidity	Windy	Outlook	ST	SH
1	85	85	False	Sunny	1.47	-0.14
2	80	90	True	Sunny	0.76	0.44
3	65	70	True	Rainy	-1.36	-1.86
4	72	95	False	Sunny	-0.37	1.01
5	71	91	True	Rainy	-0.51	0.55

ST和SH分别是Temperature和Humidity的标准化值.

(3) SMOTE算法未考虑数据的分布, 可能产生距离相近、甚至相互重叠的样本观测.

(4) 位于少数类边界区域的样本观测可能为分类提供更多的信息, 但SMOTE算法同等地对待所有少数类样本观测, 未能充分利用信息更多的样本观测.

4.4 案例分析

基于car数据集, 首先划分训练集和测试集, 样本量分别为40和20. 按照油耗Mileage是否小于30, 形成二分类的因变量(取值为0和1).

```
alldata = pd.read_csv('car.csv')

y = alldata[['Mileage']]
X = alldata[['Country', 'Reliability', 'Price', 'Type',
       'Weight', 'Disp.', 'HP']]
X_train, X_test, y_train, y_test = train_test_split(
    X,y, test_size=1/3, random_state=1)

y_train_group= copy.deepcopy(y_train)
y_train_group[y_train_group['Mileage']<30] = 0
y_train_group[y_train_group['Mileage']>=30] = 1
```

可以看到, 训练集中32个样本观测的因变量取值为0, 8个样本观测的因变量取值为1, 两个类别不平衡. 假定少数类发生误判具有比多数类发生误判更高的损失, 因此产生了类别不平衡问题.

采用RandomUnderSampler函数实现欠采样. 由于RandomUnderSampler函数不允许缺失值, 因此先进行缺失值处理. 以中位数插补定量变量的缺失值, 以众数插补定性变量的缺失值, 处理后的训练集为X_trainpreprocess.

设置随机数种子为1, 在多数类中实施无放回采样. 可以看到, 采样后的两个类别样本量都为8.

```
rus = RandomUnderSampler(random_state=1)
X_rus, y_rus = rus.fit_resample(X_trainpreprocess,
    y_train_group)
```

预测阶段无需实施类别不平衡采样, 故不需要将上述处理应用到测试集中.

下面, 采用EasyEnsembleClassifier函数实现EasyEnsemble算法. 由于EasyEnsembleClassifier函数使用了scikit-learn中的AdaboostClassifier函数, 需按照AdaboostClassifier函数的要求作数据预处理, 即处理缺失值, 并对定性变量作独热编码. 只需修改上面预处理中的categorical_transformer.

```
categorical_transformer = Pipeline(steps=[('imputer2',
    SimpleImputer(strategy='most_frequent')),
    ('onehot', OneHotEncoder(handle_unknown='ignore'))])
```

设置随机数种子为1, 基于训练集建立20个Adaboost模型并作集成.

```
eec = EasyEnsembleClassifier(n_estimators=20, random_state=1)
eec.fit(X_trainpreprocess, y_train_group['Mileage'])
```

将EasyEnsemble算法应用到测试集时, 应先按照训练好的数据预处理对测试集作预处理, 随后进行预测. 预测结果显示, 20个样本观测中有4个发生了误判, 它们的真实类别都是多数类.

```
X_testpreprocess = pd.DataFrame(preprocessor.transform(X_test)
    )
y_test_group = copy.deepcopy(y_test)
y_test_group[y_test_group['Mileage']<30] = 0
y_test_group[y_test_group['Mileage']>=30] = 1

y_predeec = eec.predict(X_testpreprocess)
```

```
print(confusion_matrix(y_test_group['Mileage'], y_predeec))
```

下面, 采用RandomOverSampler函数实现过采样. 取shrinkage的默认值0[1], 以1为随机数种子, 基于训练集拟合, 获得了两个类别样本量均为32的平衡类别数据集.

```
ros = RandomOverSampler(random_state=1)
X_ros, y_ros = ros.fit_resample(X_train, y_train_group)
```

采用SMOTENC函数实现SMOTE和SMOTENC算法. 需要进行以下数据预处理:

(1) 处理缺失值.

(2) 将字符串转换为数值编码. 这里选择将字符串转换为数值编码, 是因为SMOTE函数允许定性变量.

(3) 对定量变量标准化.

```
numeric_features = np.where(X_train.dtypes != object)[0]
numeric_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='median')),
    ('standardscaler', StandardScaler())])

categorical_features = np.where(X_train.dtypes == object)[0]
categorical_transformer = Pipeline(steps=[('imputer2',
    SimpleImputer(strategy='most_frequent')),
    ('oe', OrdinalEncoder())])

preprocessor = ColumnTransformer(
    transformers=[
        ('num', numeric_transformer, numeric_features),
        ('cat', categorical_transformer, categorical_features)
           ])
X_trainpreprocess = pd.DataFrame(preprocessor.fit_transform(
    X_train))
```

[1]注意, 若shrinkage¿0, 函数只能接受定量变量, 且不允许缺失值, 需要先作缺失值处理.

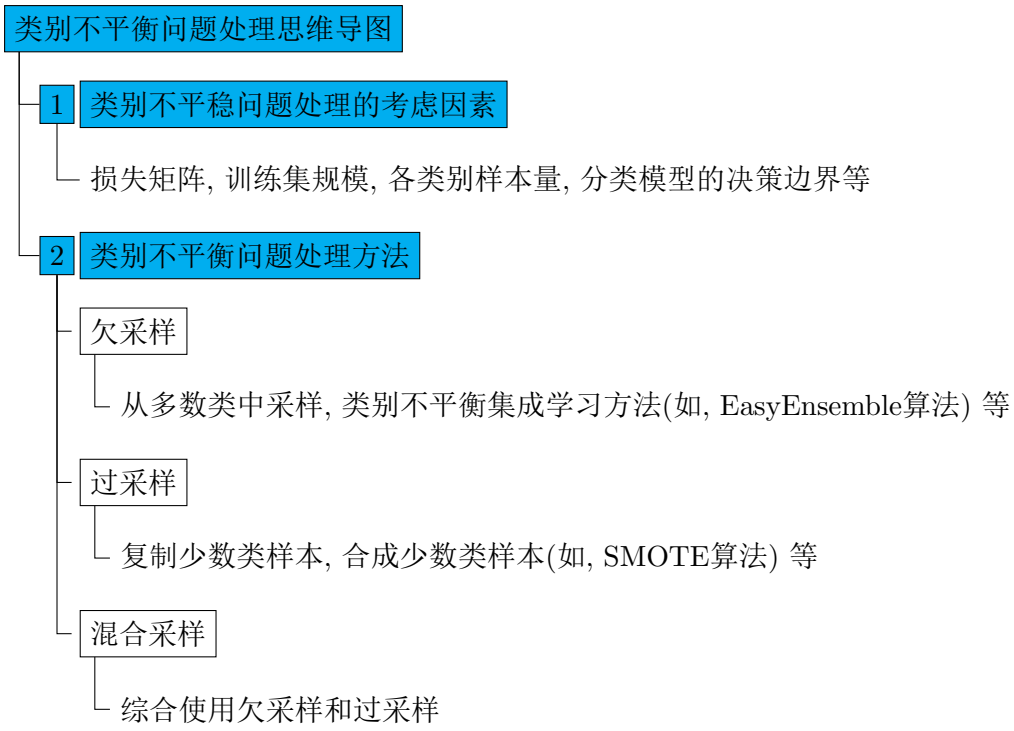
在SMOTENC函数中, 设置categorical_features=[5,6], 从而声明第6和第7个变量为定性变量[2]; 设置随机数种子为1, 近邻数量为5. 基于训练集拟合, 获得了两个类别样本量均为32的平衡类别数据集.

```
sm = SMOTENC(categorical_features=[5,6], random_state=1,
    k_neighbors=5)
X_sm, y_sm = sm.fit_resample(X_trainpreprocess, y_train_group)
```

预测阶段无需实施类别不平衡采样，故不需要将上述处理应用到测试集中.

本章小结

(1) 思维导图



(2) Python函数

表4.2列出了类别不平衡处理相关的Python函数.

[2]务必声明定性变量, 否则会被作为定量变量.

表 4.2: 类别不平衡处理Python函数

方法	函数名称与所属模块	重要参数	应用提示
欠采样	RandomUnderSampler (imbalanced-learn库 under_sampling)	(1)sampling_strategy: 采样策略, 默认值为'auto', 它等同于取值为'not minority', 即对除少数类以外的其他类都实施欠采样. 还可以设置为'majority', 'all'和'not majority'. 对于二分类问题, 还可以设置为一个数值, 表示采样后少数类样本量与多数类样本量之比 (2) replacement: 是否有放回地采样, 默认值为False, 即实施无放回采样 (3) random_state: 随机数种子, 默认值为None	-
EasyEnsemble 算法	EasyEnsembleClassifier (imbalanced-learn库ensemble)	n_estimators: 为Adaboost模型数量, 默认值为10	由于使用了scikit-learn中的AdaboostClassifier函数, 需按照AdaboostClassifier函数对数据的要求作预处理: (1) 函数不允许缺失值, 需处理缺失值 (2) 函数不能直接处理定性变量, 需作独热编码
过采样	RandomOverSampler (imbalanced-learn库 over_sampling)	(1) sampling_strategy: 采样策略, 默认值为'auto', 它等同于取值为'not majority', 即对除多数类以外的其他类都实施过采样. 其余取值与RandomUnderSampler函数类似 (2) shrinkage: smoothed Bootstrap样本计算中的参数. 默认值为None, 等同于shrinkage=0, 此时只有Bootstrap样本, 即从少数类样本中采样的值. 当shrinkage> 0时会获得smoothed Bootstrap样本 (3) random_state: 随机数种子, 默认值为None	当shrinkage=None或0时, 函数允许缺失值; 当shrinkage> 0时, 只能接受定量变量, 且不允许缺失值
SMOTE算法 SMOTE-NC算法	SMOTENC (imbalanced-learn库 over_sampling)	(1) categorical_features: 指明哪几列为定性变量. 对于定性变量, 将以邻居的众数作为合成值, 而非线性插值 (2) k_neighbors: 邻居数量, 默认值为5 (3) sampling_strategy: 采样策略. 取值及含义与RandomOverSampler函数中的sampling_strategy一致 (4) random_state: 随机数种子, 默认值为None	(1) 函数不允许缺失值, 需处理缺失值 (2) 函数不接受字符串, 需将字符串转化为数值编码. 这里选择将字符串转换为数值编码是因为函数允许定性变量, 切记在函数中声明定性变量, 否则会被作为定量变量 (3) 为了距离运算中变量尺度保持一致, 需将定量变量标准化

习题

1. 类别不平衡处理需要在什么情况下进行？确定所采用的类别不平衡处理方式时应考虑哪些因素？

2. 欠采样与过采样各有哪些优缺点？EasyEnsemble方法给予我们什么样的启发？

3. 设少数类样本数据集为{a,b}, 多数类样本数据集为{c,d,e,f,g,h}. 请列出EasyEnsemble算法中训练一个Adaboost模型所使用的所有可能数据集.

4. SMOTE算法产生合成一个样本的过程中, 哪些值需要随机产生, 哪些值是设置的？可以保证样本观测数量满足要求吗？

References

[1] Weiss G.M. (2004). Mining with rarity: a unifying framework. Acm Sigkdd Explorations Newsletter, 6(1): 7-19.

[2] Liu X.Y., Wu J., Zhou Z.H. (2009). Exploratory undersampling for class-imbalance learning. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 39(2): 539-550. Chawla N.V., Bowyer K.W., Hall L.O., Kegelmeyer W.P. (2002) SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research 16(1): 321 – 357.