数据挖掘中的pulearning问题

汇报人：崔博涵

目录

1. 研究背景

實事求是

1.1 pulearning背景简介

- P表示positive，U表示Unlabel，属于（semi-supervised learning）中的一种，用于二分类问题，给定的样本只有正样本和未标注样本；

- 业务场景中应用pulearning 的主要目的是识别无标记样本（除严格正样本之外的）中我们想要的正样本

- 风控业务背景：金融风控场景，只有部分用户被标记为欺诈用户，剩下的大量用户未被标记。虽然这其中大多数信用良好，但仍有少量可能为欺诈用户。（由于数据的缺乏，对pulearning的研究相对较少为方便操作，我们可以将未标记的样本都作为负样本进行训练，但存在几个缺陷：1. 正负样本极度不平衡，负样本数量远远超过正样本，效果很差。2.某些关键样本会干扰分类器的最优分隔面的选择，尤其是SVM。

- 如何选择合理的正负样本？--打标
- 如何训练一个较为准确的分类器？

2.技术方法

实事求是

2.1 常用解决思路

1、启发式地从未标注样本里找到可靠的负样本，以此训练二分类器。
2、将未标注样本作为负样本训练分类器，由于负样本中含有正样本，分类效果严重依赖先验知识

标准方法、pubagging、two_steps

2.2 标准方法

将正样本和未标记样本分别看作是positive samples和negative samples，然后利用这些数据训练一个标准分类器。这种朴素的方法在文献Learning classifiers from only positive and unlabeled data中有介绍。论文核心结果是，**在有标签样本满足SCAR假设下，可以使用通过positive和unlabel的数据直接训练的分类器结果代替正确分类器（结果之间只相差一个比例系数）**

我们发现：**这个公式通过 c 搭建起了传统分类器与非传统分类器之间的桥梁。**

因此，该非传统分类器便具有了以下特点：

1. 与传统分类器具有相同的排序性（Ranking Order）：

$$Pr(y=1|x_1) > Pr(y=1|x_2) \Leftrightarrow Pr(s=1|x_1) > Pr(s=1|x_2) \tag{13}$$

2. 让传统分类器达到某个目标的召回率，等价于非传统分类器达到这个目标。

3. 给定类别先验 α （或标签频率 c ）的条件下，非传统分类器能转化为传统分类器，也就是：

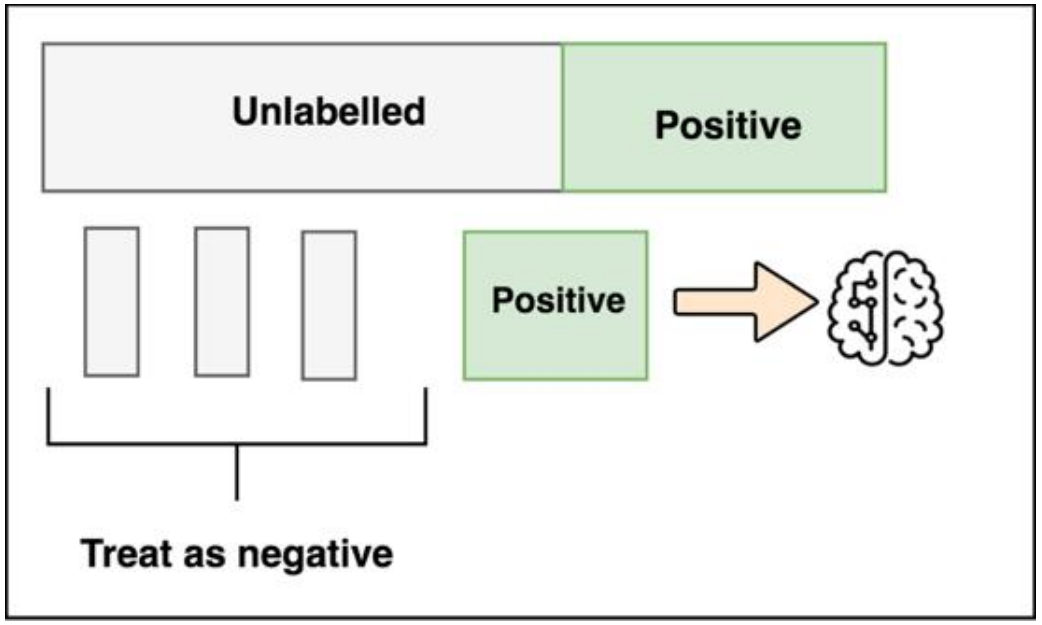
$$Pr(y=1|x) = \frac{Pr(s=1|x)}{c} \tag{14}$$

被标注为正样本的可能性

标签频率：即已标注正样本占原始正样本比例

2.3 pubagging

a)通过将所有正样本和未标记样本进行随机组合来创建训练集；
b)利用这个“bootstrap”样本来构建分类器，分别将正样本和未标记样本视为positive和negative；
c)将分类器应用于不在训练集中的未标记样本 - OOB（“out of bag”）- 并记录其分数；
d)重复上述三个步骤，最后为每个样本的分数为OOB分数的平均值。
通过bagging的方法可以将所有未标记样本进行分类（粗），增大了分类精度。描述这种方法的一篇论文是A bagging SVM to learn from positive and unlabeled examples。



Algorithm 1 Inductive bagging PU learning

INPUT : $\mathcal{P}, \mathcal{U}$, K = size of bootstrap samples, T = number of bootstraps
OUTPUT : a function $f : \mathcal{X} \to \mathbb{R}$
for $t = 1$ to T **do**
  Draw a subsample $\mathcal{U}_t$ of size K from $\mathcal{U}$.
  Train a classifier f_t to discriminate $\mathcal{P}$ against $\mathcal{U}_t$.
end for
Return

$$f = \frac{1}{T}\sum_{t=1}^{T} f_t$$

Algorithm 2 Transductive bagging PU learning

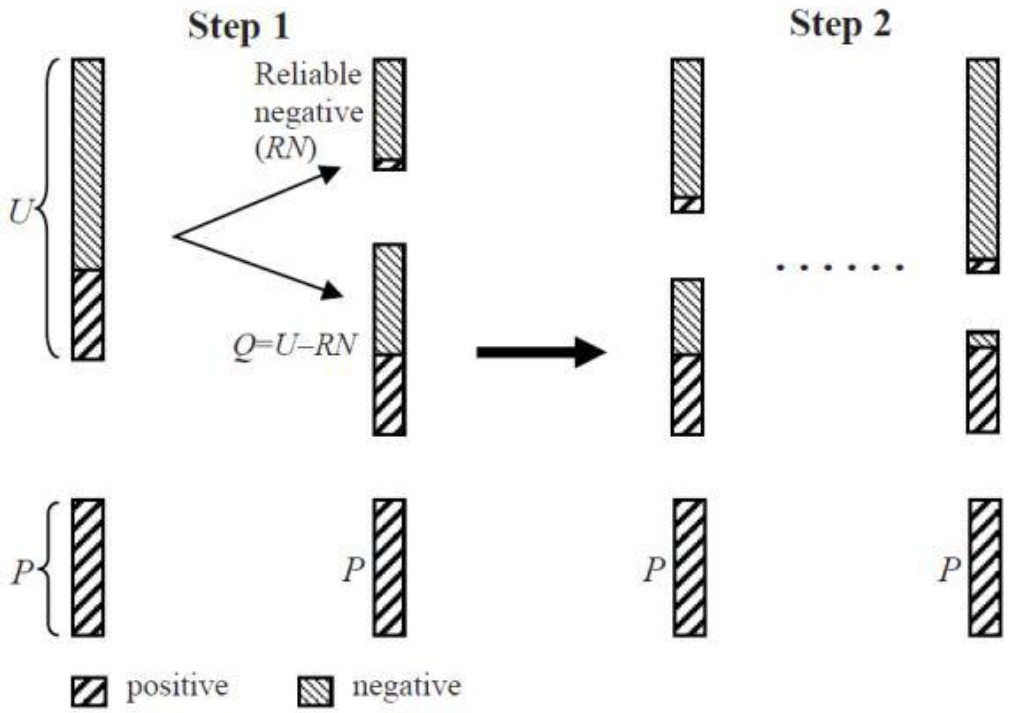
INPUT : $\mathcal{P}, \mathcal{U}$, K = size of bootstrap samples, T = number of bootstraps
OUTPUT : a score $s : \mathcal{U} \to \mathbb{R}$
Initialize $\forall x \in \mathcal{U}, n(x) \leftarrow 0, f(x) \leftarrow 0$
for $t = 1$ to T **do**
  Draw a bootstrap sample $\mathcal{U}_t$ of size K in $\mathcal{U}$.
  Train a classifier f_t to discriminate $\mathcal{P}$ against $\mathcal{U}_t$.
  For any $x \in \mathcal{U} \setminus \mathcal{U}_t$, update:

$$f(x) \leftarrow f(x) + f_t(x),$$
$$n(x) \leftarrow n(x) + 1.$$

end for
Return $s(x) = f(x)/n(x)$ for $x \in \mathcal{U}$

2.3 two_steps

识别可以百分之百标记为negative的未标记样本子集（这些样本称为“reliable negatives”。
Step1：用正样本和未标记样本训练一个模型
Step2：然后对未标记样本进行预测，按照概率排序，选取前面的样本作为reliable negatives。使用正负样本来训练标准分类器并将其应用于剩余的未标记样本。



如何选取可靠的负样本？
可能是一门玄学

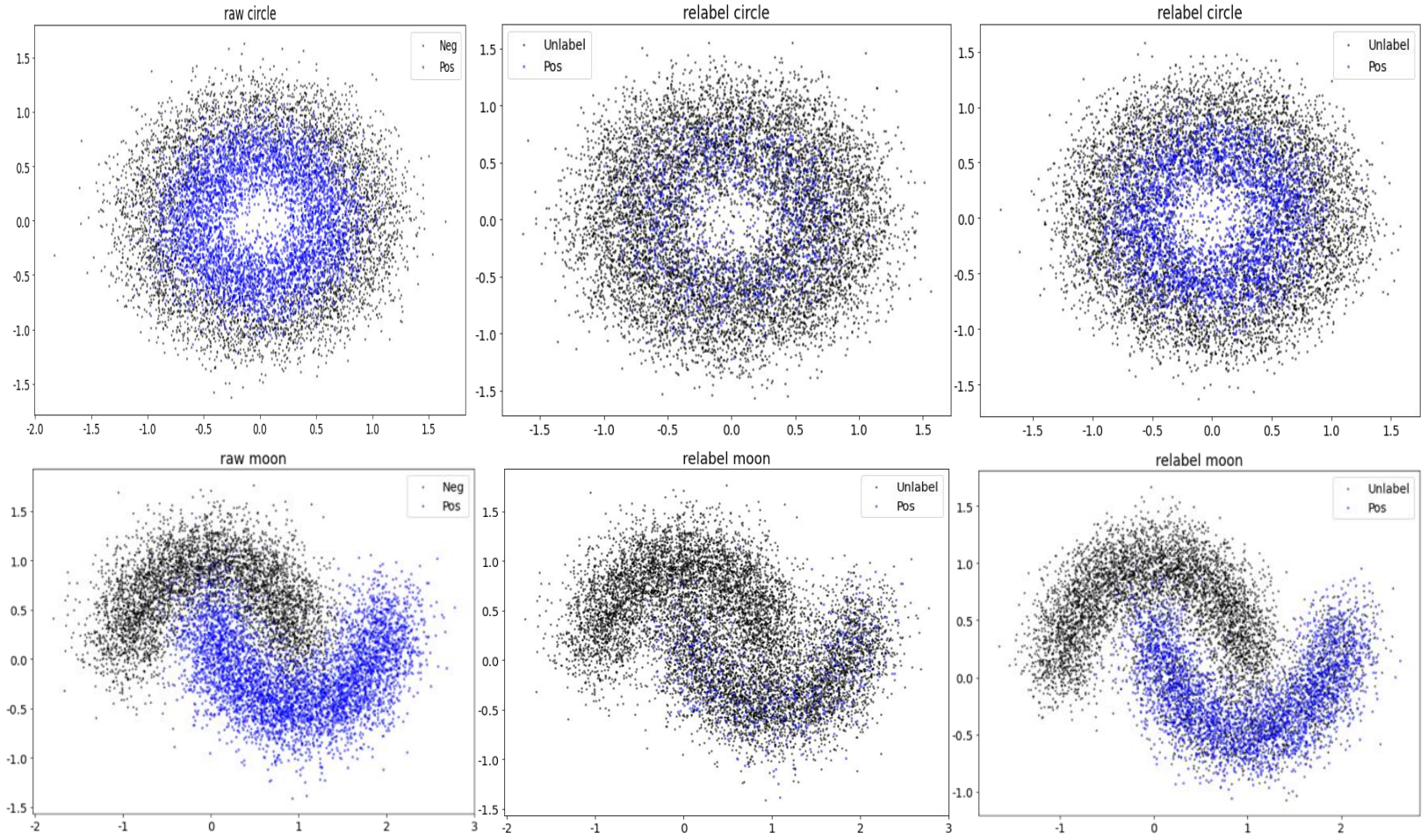
3.实验设计

实事求是

3.1 实验设计

- **实验目的：**探讨影响分类器识别出未标记样本中的正类样本的因素
- **实验数据集（from sklearn.Datasets）：**
  - Moon：噪声0.25，数据点10000，pos5000，neg5000，分类界限较为明显
  - Circle:　　噪声0.2 ，　pos5000，neg5000，分类界限比较不明显
- **实验变量：**
  - 伪负样本数：（随机从正样本中抽样标注为0（作为混入负样本中的正样本），将整体负样本看做无标签样本，本次的伪负样本数设置为4500,4000,3500,3000,2000
  - 模型：标准分类器（随机森林、Xgboost）,Bagging分类器（基分类器为决策树和xgboost），改良的两步法
- **评估**
  - 对重打标数据进行按照3:1的比例进行训练集和验证集的划分，根据训练好的模型在验证集（标签为原始正确的标签）上的auc进行评估，AUC可以反映分类器区分正负样本的能力
  - 已训练好的模型在“unlabel”数据集上（标签同上为原始标签）原正样本的召回率（即预测正确的正样本占原正样本的比例），阈值取所有unlabel数据集预测值的平均
  - 模型在unlabel数据集上预测分值的colorbar
  - 模型在整体数据集上不同标签的预测概率分布

3.2 实验数据集



上三幅图分别表示circle数据集的最原始分布、抽取4500正样本作为间谍与原始负样本一起作为无标签样本（pos:unlabel=500:9500）、抽取3000正样本作为间谍与原始负样本一起作为无标签样本（pos:unlabel=3000:7000）

下三幅图是moon数据集同上规则的可视化

3.3 实验结果

		rf	xgb	Bag_tree	Bag_xgb	Two_steps	avg
Circle(500)	valid_auc	79.05%	87.49%	83.69%	87.71%	**88.65%**	88.12%
	recall	41.38%	77.36%	75.16%	84.82%	89.49%	89.16%
Circle(1000)	valid_auc	79.68%	88.21%	81.84%	88.46%	**90.16%**	88.81%
	recall	51.38%	85.55%	78.60%	89.22%	93.43%	92.85%
Circle(1500)	valid_auc	82.34%	89.68%	83.15%	**90.33%**	76.34%	90.23%
	recall	57.34%	85.97%	78.77%	91.00%	98.54%	93.66%
Circle(2000)	valid_auc	83.04%	89.70%	82.84%	89.87%	85.20%	**89.98%**
	recall	62.97%	84.07%	79.77%	88.77%	95.70%	95.30%
Circle(3000)	valid_auc	85.25%	90.88%	83.72%	91.06%	87.08%	**91.14%**
	recall	74.20%	89.85%	82.10%	91.30%	96.70%	96.00%

		rf	xgb	Bag_tree	Bag_xgb	Two_steps	avg
Moon(500)	valid_auc	89.91%	96.71%	94.63%	97.25%	96.77%	**97.29%**
	recall	47.71%	87.87%	88.96%	95.64%	93.36%	93.38%
Moon(1000)	valid_auc	94.64%	97.15%	95.52%	**97.70%**	96.45%	97.64%
	recall	61.55%	91.85%	91.53%	96.83%	94.43%	94.80%
Moon(1500)	valid_auc	96.87%	98.49%	96.82%	**98.79%**	95.62%	98.67%
	recall	74.23%	96.70%	95.00%	97.11%	98.74%	98.69%
Moon(2000)	valid_auc	97.92%	96.47%	95.20%	**99.37%**	98.29%	99.33%
	recall	79.20%	99.13%	97.41%	98.07%	98.60%	98.50%
Moon(3000)	valid_auc	98.14%	98.82%	97.54%	**99.15%**	96.91%	99.01%
	recall	92.10%	97.70%	97.15%	98.30%	98.70%	99.10%

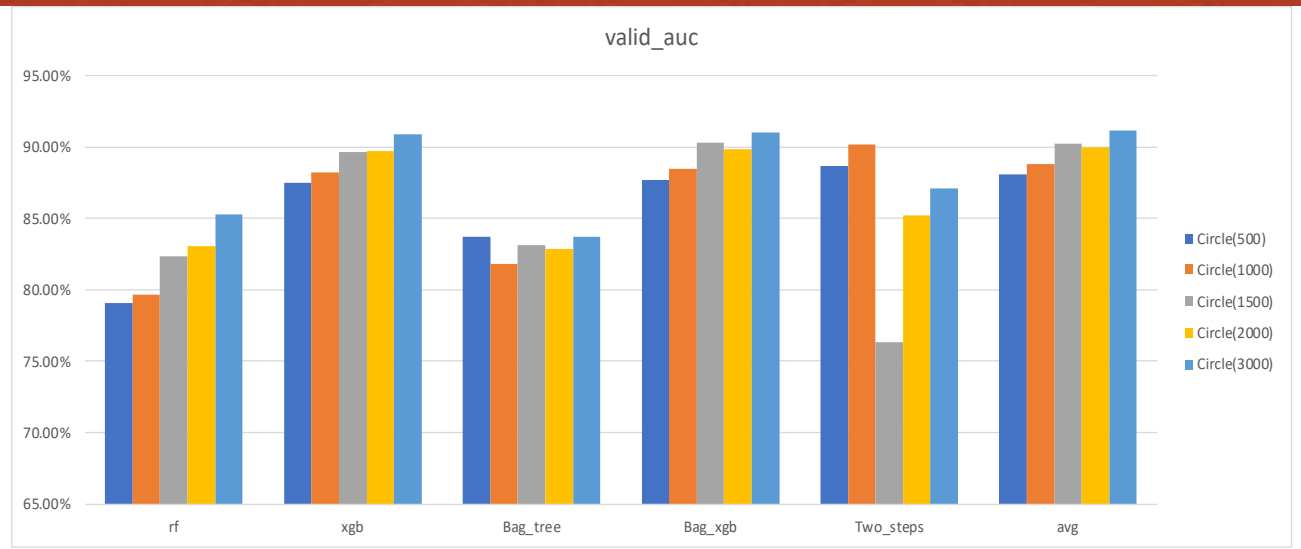
Circle(500)----仅500正样本，其他10000-500为未标注
Rf----随机森林
Xgb----标准分类器，单模型
Bag----集成方法，后跟基分类器名称
Avg----xgb，bag_xgb，two_steps的平均值
Valid_auc----同一批验证集上的auc
Recall----未标注数据集上“伪负样本”的召回率

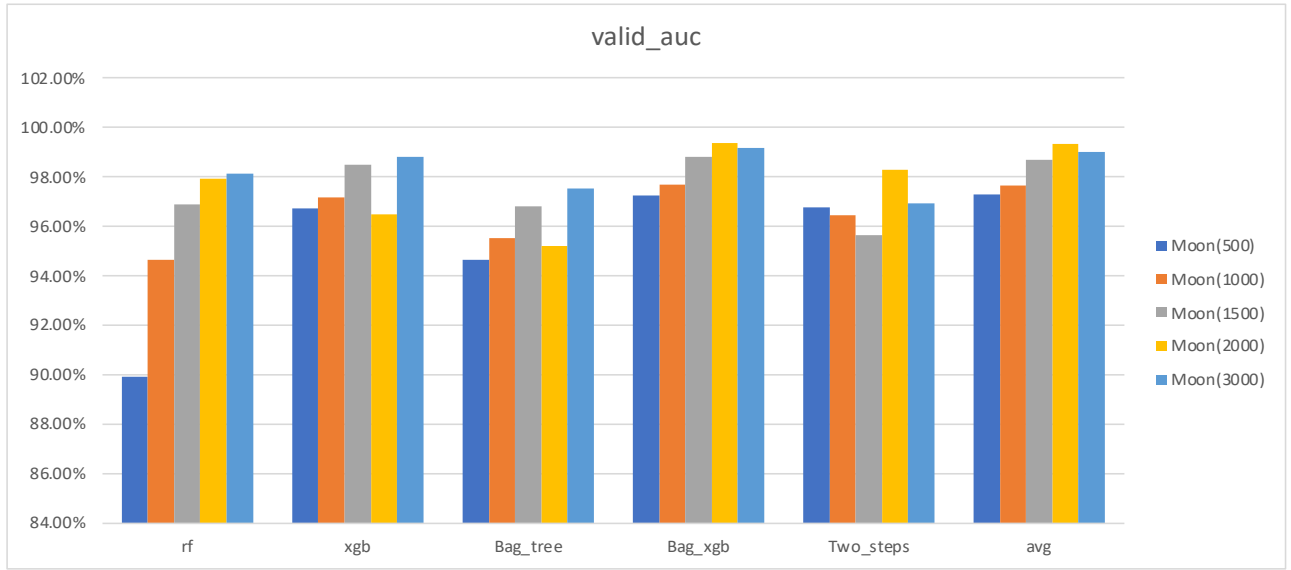
baseline_circle：
基于未处理前数据训练的随机森林在相同验证集上的auc为90.29%，xgboost为91.48%，bagging方法为91.78%

baseline_moon：
基于未处理前数据训练的随机森林在相同验证集上的auc为98.01%，xgboost为98.43%

由此可以看出moon数据集比circle更易分，表中moon上的结果显然好于circle；

3.4 结果可视化

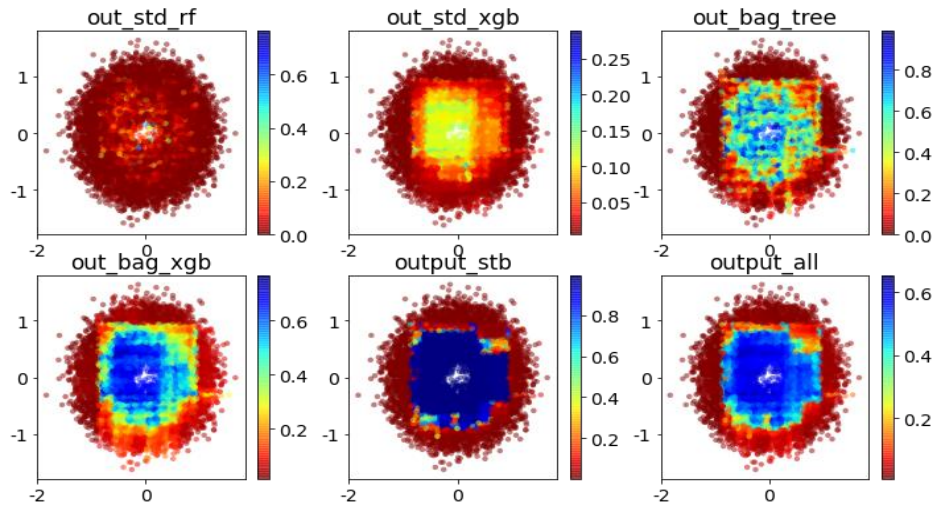


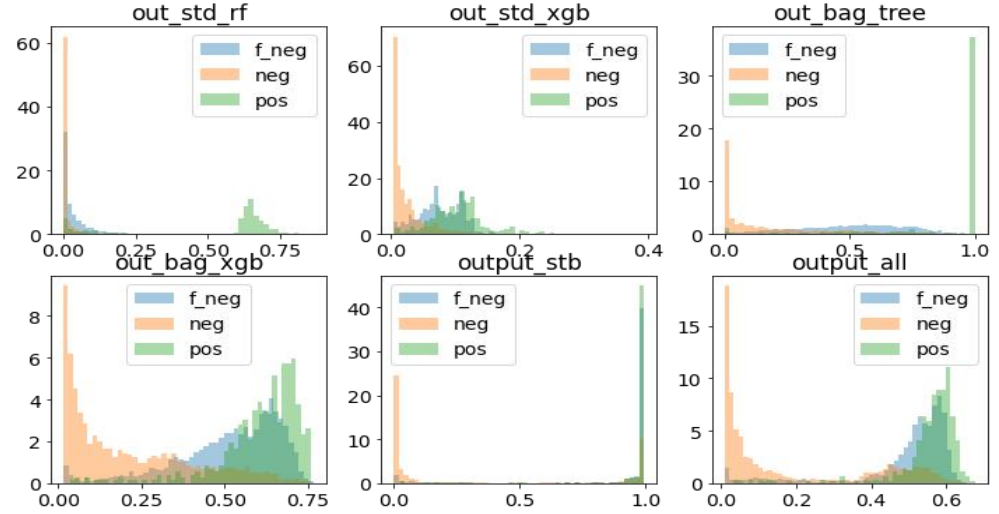


- 随着正样本比例增加，除两步法之外，其他模型识别能力逐渐增强（这里考虑两步法是否收到“可靠负样本”选取规则、正样本比例、原始数据集噪声大小等因素的影响）

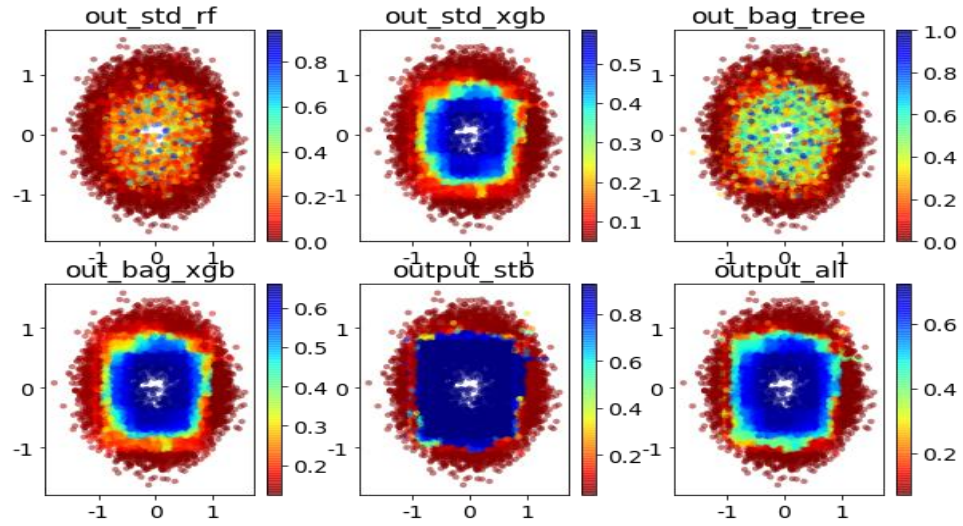
- 最好实验结果出现在bagging上，moons数据集频率显著高于circle，猜测two_steps方法更适用于样本噪声较大（原始正负样本存在一定的重叠），正样本比例较小的数据集上；bagging方法泛化性能较强

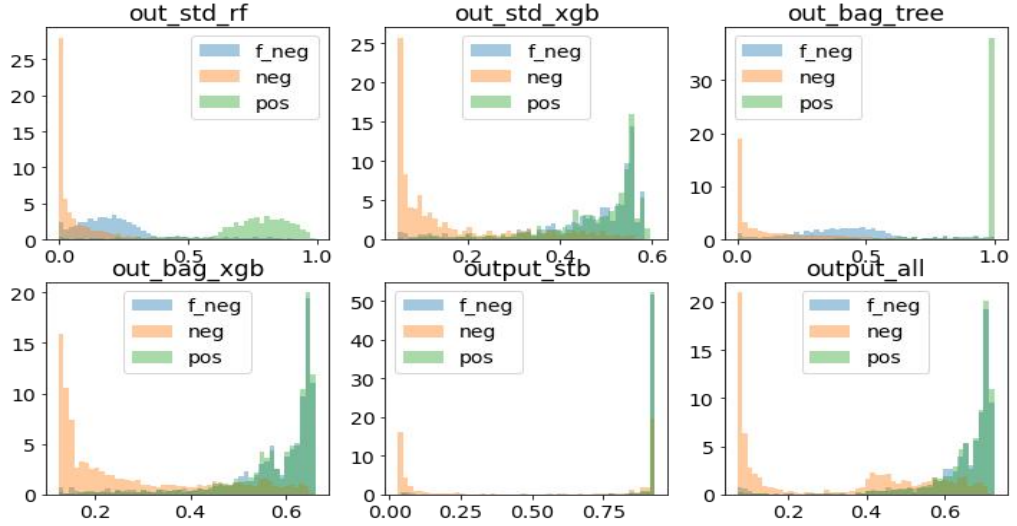
3.5 circle可视化-500&3000





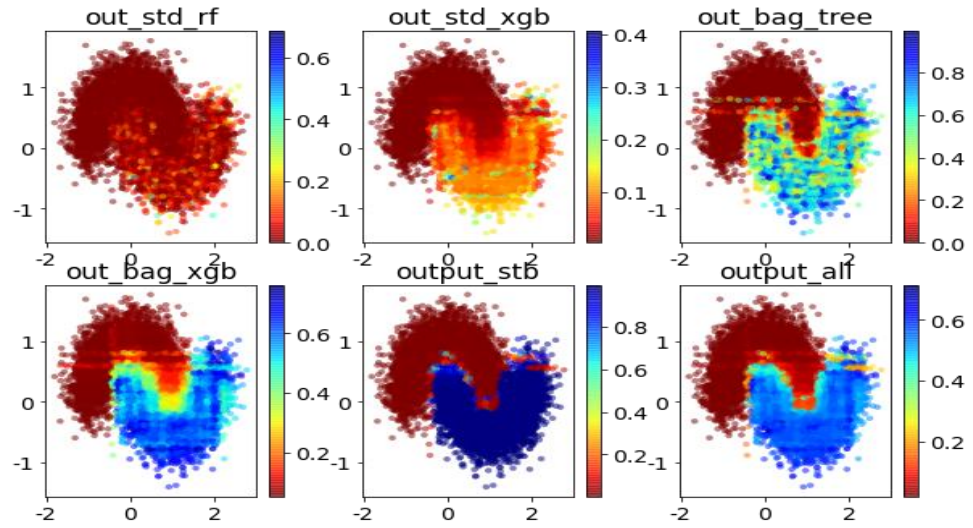
上图分别表示circle数据集在pos:unlabel=500:9500上预测结果的可视化，概率分布图（右）f_neg越贴合pos，原理neg，模型识别效果越好；

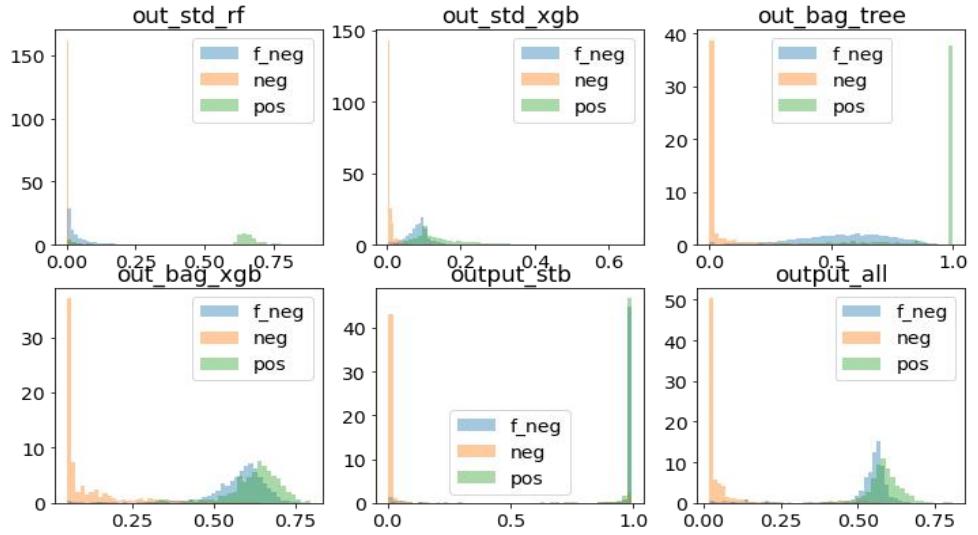




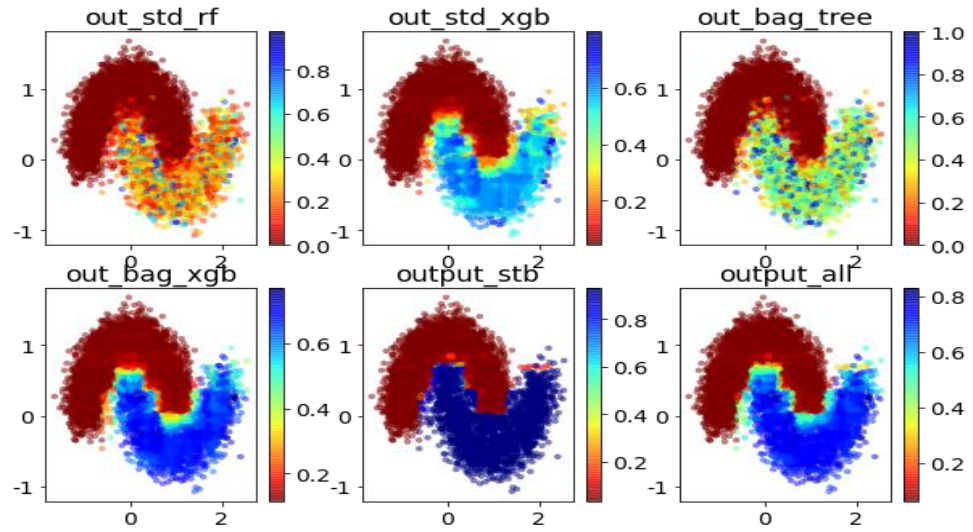
下图分别表示circle数据集在pos:unlabel=3000:7000上预测结果的可视化

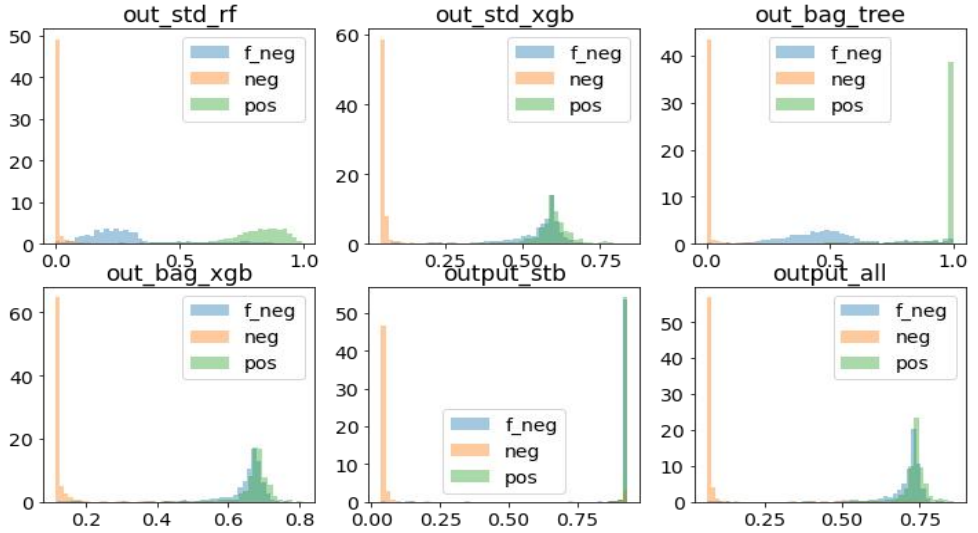
3.6 moon可视化-500&3000





上图分别表示moon数据集在pos:unlabel=500:9500上预测结果的可视化，概率分布图（右）f_neg越贴合pos，原理neg，模型识别效果越好；





下图分别表示moon数据集在pos:unlabel=3000:7000上预测结果的可视化

随着正样本占比提高，模型区分度也逐步提高

4. 总结

实事求是

4. 实验总结

实验结论：

- **正负样本比对**实验结果影响较大，随着正样本占比提高，一般模型区分度也逐步提高
- **数据本身噪声**对模型分类效果也有一定影响，一般而言，boost方法略由于bagging方法
- 两步法适用于**未标注数据中正样本含量较高**的数据集（不知是不是和负样本挑选规则有关系）
- 若数据集优良（噪声小，分布均衡），不再建议使用bagging，效果提升不明显反而浪费了资源
- 综上，pu-bagging 的方法略强于two-steps

实验问题：

- 两步法

  本次两步法挑选负样本的规则为：所有预测值小于在标注正样本上的最小预测值的样本；挑选正样本的规则为：选取unlabel中得分top5%的样本为正样本；迭代停止条件是基本上将unlabel中全部标注。这导致两个严重的问题：

  1、短短几步后，几乎挑选不到合适的负样本，使得随着正样本比提高，迭代后的正负样本比严重失衡，分类效果下降；

  2、模型最优效果的迭代步数大多出现在满足上述条件之前，因此迭代停止条件的设置有待优化

- 标准-bagging的提升

  不知道是不是xgboost本身分类效果就已经非常好还是在bagging的过程中模型出现了过拟合，bagging_xgboost的效果往往比xgboost好不了多少，有时甚至还差一点，这点很是怪异了。

4. 改进

1. 尝试更多类型数据集与业务场景应用（比如图像，文本天然的pulearning数据集-较少）
2. 进一步控制变量 研究pubagging与two-steps方法在不同数据集上的适用性
3. 进行更多深度学习+pulearning理论上的探索
4. Pulearning问题和样本不均衡问题的结合
5. 扩展pulearning问题的评估指标

6. 传送门：pubagging和传统分类器已经写成 python包（pip install pulearn）

感谢聆听