1. 数据集层面：重采样resampling

1.1 随机欠采样——随机剔除 majority class的样本

* 优点：当数据量很庞大时，可以通过减少数据量来改善执行速度和内存问题。
* 缺点：
    * 丢弃了潜在的有用信息，它们可能对创建rule classifiers 很重要；
    * 随机下采样选择的样本可能是a biased sample，它不是总体的准确代表，从而导致在实际测试集上不准确。

1.2 随机过采样——随机复制增加minority class 样本

* 优点：信息无损；表现比下采样好
* 缺点：增加了过拟合的可能，因为它增加了少数类的样本，它们都是近似的，换句话说，试卷上增加的题目都是相同的题型。

1.3 基于聚类的过采样，步骤如下：分别对少数类和多数类应用K-means算法，为了找出数据集中（相近的）簇（clusters）。之后，对每个簇进行过采样使同一类中的所有clusters有等量的样例，且所有类有相同的size。具体地：

```
![](cluster-based_oversampling.png)
```

* 优点：聚类克服了【类间】不平衡；克服了【类内】不平衡——一个类由多个sub clusters组成，每个sub cluster所含的样本量不一致
* 缺点：如同多数过采样技巧一样，过拟合。

1.4 Informed Over Sampling 合成少数类过采样技术（Synthetic Minority Over-sampling Technique--SMOTE）

避免了【单纯精确复制少数样例添加到主数据集中发生】的过拟合。方法：从少数类中抽取一个子集，然后合成新的相似样例，添加到原始数据集中。局限性：适用于低维空间，且低维空间中少数类与多数类是能区分的，少数类中抽取的子集label如-1，是无疑为-1的。

具体的：

若现有低维空间没有区分度，合成的新样例可能是噪音，合成无效。

* 优点：减少了随机过采样造成的过拟合问题，因为新样例是合成得到而不是单纯复制；信息无损。
* 缺点：生成新合成样本的同时，SMOTE没有考虑邻近的其它类的样本。这可能增加类别的重叠，并可能引入额外的噪音；SMOTE对高维数据不是很有效。

1.5 Modified synthetic minority oversampling technique(MSMOTE)

SMOTE的修正版。SMOTE 不考虑少数类隐含的分布和数据集中的潜在噪音，特此修正。

该算法通过计算【少数类样本】与【训练集样本】之间的距离，将少数类分为3个不同的组：

1. Security/Safe samples —— 能现在改善分类器性能的数据点
2. Border samples —— 难以归为另2组的数据点
3. latent noise samples —— 降低分类器性能的数据点

具体流程与SMOTE相同，选择最近邻的策略有所不同。该算法从安全样本的k个最近邻点中随机选择一个，从边界样本中选择最近邻点，对潜在噪点不做处理。

2. 算法层面：集成算法

集成方法的目标是提升单个classifier 的性能。

2.1 Bagging --Bootstrap Aggregating

与Boosting不同， Bagging 的随机抽样是【有放回】的：

具体的，从总体中，获得10个样本集boostrapped samples，每个样本集含有200个样例（随机有放回抽样得到）。每个样本集都与原始数据集不同，但分布与方差相似。像逻辑回归、神经网络、决策树算法，适用于200样例的样本集。

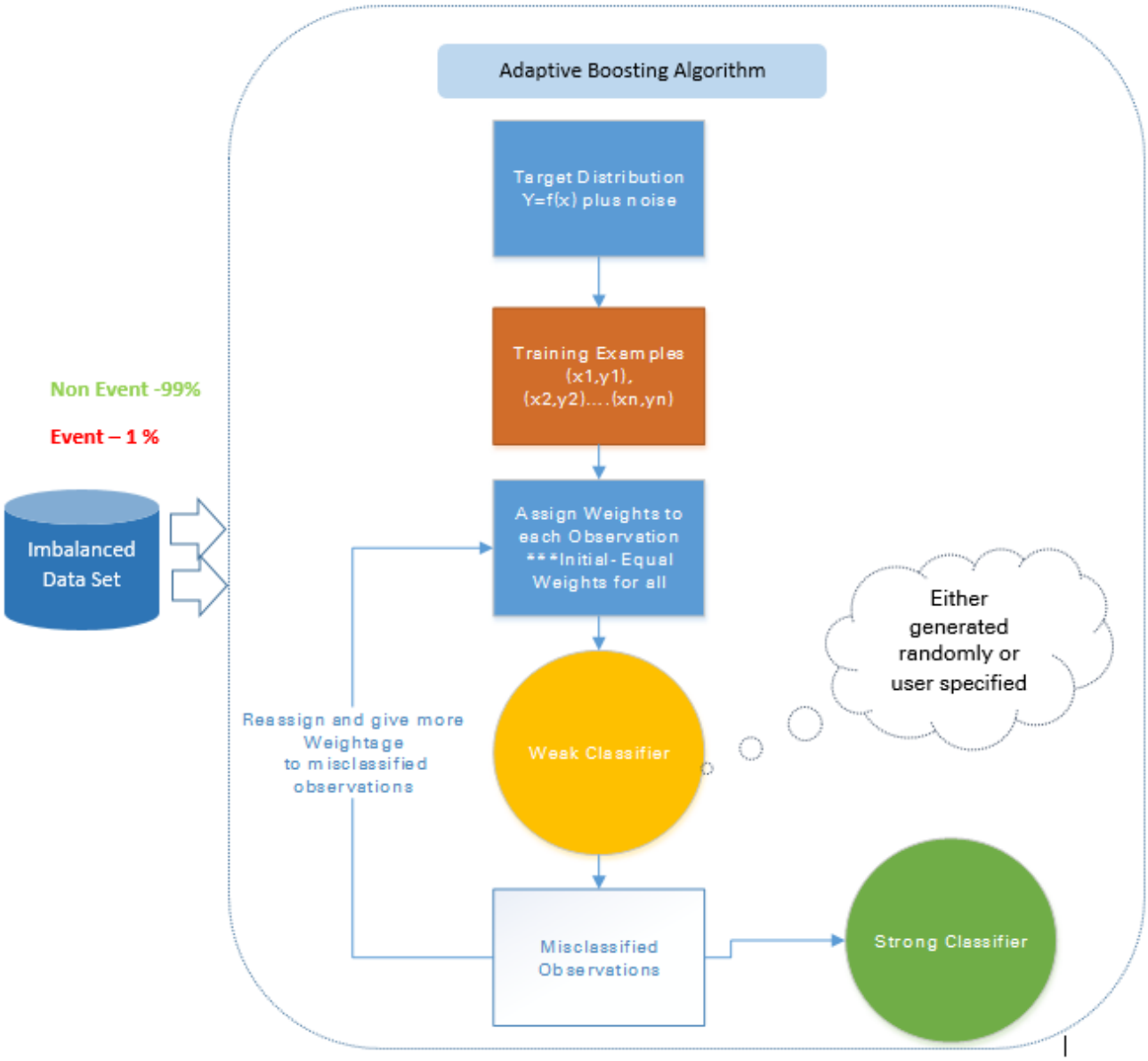
优点：
1. 提升算法稳定性和准确度
2. 降低方差
3. 克服过拟合
4. 改善bagged classifier 的误分率
5. **在噪点多的数据环境下，Bagging优于Boosting**

缺点：只有基分类不差的情况下，Bagging才work。否则会进一步降低性能。

2.2 Boosting

Boosting 的基分类器/弱分类器建立在整个训练集上。什么是weak classifiers? 预测准确度仅仅比平均好一点点。数据上很小的变动却使分类模型产生很大改变，则称这个分类学习算法 weak。在下轮迭代中，新的分类器投放更大的权重在那些上轮分错的样例上。

2.2.1 Adaptive Boosting- Ada Boost



Ada Boost通过结合许多弱的和不准确的规则来创建高度准确的预测规则。每个分类器都经过连续训练，目的是在每轮中对前一轮中错误分类的例子进行正确分类。一个经过训练的分类器要做出有力的预测，应当满足以下3个条件：

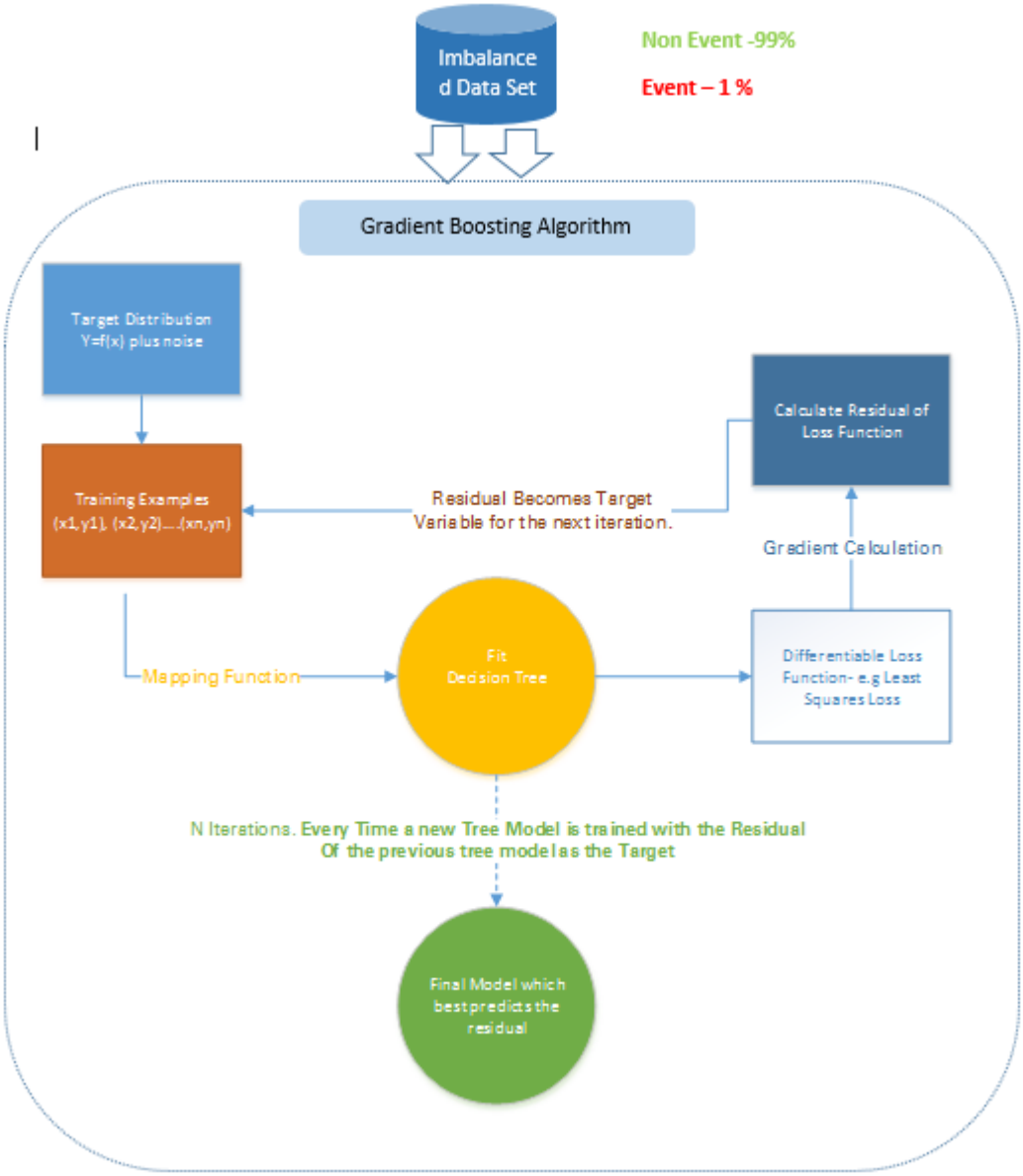
1. 规则应该是简单的
2. 训练集样本量足够充分
3. 足够低的训练误差

每个weak hypothesis的准确率都要比随机瞎猜要好一点点，这是Boosting算法最终得到低误差的根本前提假设

每轮之后，它会更加关注难以分类的例子。关注的程度通过一个权重来衡量，所有样例的初始权重都是相等的。每轮迭代后，分错样例的权重增加，正例的权重降低，然后继续投喂给弱学习器进行下一轮迭代。直到misclassification rate 降到满意为止，得到最终的强分类器。

* 优点：易于实现；适用广，适用于任何分类问题，不容易过拟合
* 缺点：**对噪点和异常点敏感**

2.2.2 Gradient Tree Boosting



Gradient Boosting 的许多模型是依序（sequentially）训练的。即采用梯度下降算法的Boosting，其每个模型都采用Gradient Descent方法来最小化损失函数 y = ax+b+e. （损失函数是y = ax+b+e 的形式，才可用GD？）

Gradient Boosting 采用决策树作为基学习器。

AdaBoost 和 Gradient Boosting 根本不同：

* AdaBoost 要求用户在实际学习过程之前指定【一组】弱学习器或随机生成弱学习器。根据是否正确预测样本，每个学习器的权重在每一步都会进行调整。

* Gradient Boosting 首先在训练集上构建第【一个】学习器，对样本做预测，计算出loss，然后基于这个loss来构建更优的学习器（即所谓的sequentially）。在每一步中，使用梯度下降法计算the residual of the loss function，新的residual成为后续迭代的目标变量。

缺点：

1. Gradient Boosting Tree 比随机森林更难拟合

2. Gradient Boosting Algorithms 通常有3个参数可以调优：Shrinkage parameter, depth of the tree, the number of trees。为得到好的fit，必须进行适当训练调参，如果参数没调好，可能会导致过拟合。

2.2.2 XGBoost

Gradient Boosting Algorithm 的更高级且高效实现。
比其它Boosting 技术的优势：
1. 并行处理，比正常Gradient Boosting 快10倍；高度灵活可自定义；可自行处理缺失值；
2. **Gradient Boosting 一旦遇到负loss就停止分割节点。XGBoost将分裂到指定的最大深度，然后向后剪树，移除的splits为【超过该splits仅有1个负loss】

3. 总结

面对非均衡数据集，没有一步到位的解决方案，往往需要尝试多种方法。比如SMOTE/MSMOTE + Boosting。SMOTE Bagging，是一个常用的应对非均衡数据的高级Bagging technique。它创建Bag/Boostrap 的方式完全不同于常规Bagging。它通过在每次迭代中设置SMOTE重采样率，由SMOTE算法生成正实例。负实例集合在每次迭代中boostrap得到。

根据不平衡数据集的特点，最有效的技术会有所不同。在模型比较中应考虑相关的评估参数。穷尽上述技巧的组合方式所建立的所有模型孰优孰劣，可通过AUC ROC 来对比评估。

4.更多细节问题

- 当适用XGBoost的时候，sampling techniques 是否必要？处理非均衡问题时，集成方法和采样方法是不是二选一的关系？
  XGBoost会自动处理非均衡数据，无需使用采样方法。集成和采样不是二选一的替代关系，二者可以分开用，也可以组合适用，但一般建议直接用XGBoost。
- 基于平衡数据集建的模型，能否直接用在原始的非均衡数据集上？还是需要做某些分数上的调整？ 如果不平衡数据集上也用相同cut off，调整分数可以实现。但如果selection是基于前10分之n，则不需要调整。cut off 的设置取决于实际业务情况。