|  |  |
| --- | --- |
| **统计学** | 偏差和方差对模型的好坏进行衡量 |
| 偏差（值越大，约偏离真实数据） | 描述的是预测值的期望与真实值之间的差距。 |
| 方差（值越大，表示数据的分布越分散） | 描述的是预测值的变化范围，离散程度，是预测值的方差，也就是离其期望值的距离。 |
| 模型的真实误差 | 偏差+方差 |
| **不同类型训练集适用于的分类器** |  |
| 小训练集 | 高偏差、低方差的分类器（例如：NB）  此时分类模型过于简单 |
| 大训练集 | 低偏差、高方差的分类器（例如：KNN）  此时分类模型过于复杂 |
| **生成模型和判别式模型** |  |
| 生成模型：    以统计学和贝叶斯作为理论依据，其是通过学习得到联合概率分布P(x,y)，然后求解条件概率分布。（可以学习到数据生成的机制） | 优点：   1. 对异常值可以较好的探测； 2. 收敛速度较快，样本数量较多时更好； 3. 可以应付存在的隐变量的情况；   缺点：   1. 需要大量的样本和计算； 2. 实践中判别式模型更好； |
| 判别式模型：    直接用过学习得到条件概率分布P(x|y)。 | 优点：   1. 需要的样本和计算资源都少于生成模型； 2. 准确率高于生成模型； 3. 可以简化学习问题；   缺点：  1、没有生成模型的优点； |

**常见分类算法的优缺点**

|  |  |
| --- | --- |
| 1、朴素贝叶斯算法（生成模型）    其模型建立的基础是基于“贝叶斯假设”之上 | 优点：  1、有着坚实的数学基础，以及稳定的分类效率。  2、对小规模的数据表现很好，能够处理多分类任务，适合增量训练；  3、对缺失的数据不太敏感，算法也比较简单，常用于文本分类。  缺点：   1. 需要计算先验概率； 2. 分类决策存在错误率； 3. 对输入数据的表达形式很敏感。 |
| 1. 逻辑回归（判别式模型）   需要对数据提前归一化 | 优点：   1. 分类时计算量非常小，速度很快，存储资源低； 2. 便利的观测样本概率分数； 3. 结合L2正则化，多重共线性问题可以得到解决；   缺点：   1. 当特征空间很大时，性能会下降； 2. 易欠拟合； 3. 对于非线性特征，需进行转换再处理； 4. 更易于处理两分类问题； |
| 3、KNN算法（可用于分类也可以回归） | 优点：   1. 理论成熟，思想简单，可用于分类，也可用于回归； 2. 可用于非线性分类； 3. 训练时间复杂度为O(n); 4. 对数据没有假设，准确度高，对异常值不敏感；   缺点：   1. 计算量大； 2. 易出现样本不平衡的问题； 3. 需要大量的内存； |
| 4、决策树（使用的是信息增益） | 优点：   1. 计算简单，易于解释； 2. 对数据是否线性可分以及异常值不敏感，比较适合处理有缺失属性的样本； 3. 在相对短的时间内能够对大型数据源做出可行且效果良好的结果。 4. 易于处理特征间的交互关系并且是非参数化的，处理不相关的特征；   缺点：   1. 不支持在线学习（即增量式学习）； 2. 易出现过拟合；（随机森林和提升树在很大程度上减少过拟合） 3. 忽略了数据之间的相关性； 4. 对于各类别样本数量不一致的数据，在决策树当中，信息增益的结果偏向于那些具有更多数值的特征；（只要使用了信息增益，都有这个缺点） |
| 4.1、Adaboosting(一种加和模型)  决策树的扩展 | 优点：   1. 是一种精度很高的分类器； 2. 可以使用各种方法构建子分类器，Adaboost算法提供的只是框架； 3. 结果可解释，弱分类器的构造及其简单； 4. 简单，不用做特征筛选； 5. 不易发生过拟合；   缺点：  1、对异常值较为敏感。 |
| 4.2 随机森林（决策树的扩展）  用随机的方式建立一个森林，森林有很多的决策树组成。  在建立决策树的过程中需注意：   1. 采样   两个随机采样的过程   1. 行采样：采用有放回的方式，对输入的数据进行采样，得到可能有重复的样本集。采样的样本数和输入样本数数量一样。 2. 列采样：从M个features中，选择m个（m<<M）用于分类器的训练。（即随机选择特征） 3. 完全分裂   对采用之后的数据使用完全分裂的方式建立出决策树，决策树的一个叶子节点要么是无法继续分裂，要么里面的所有样本都是指向的一个分类。  一般的决策算法都有一个重要的步骤——剪枝，但是由于随机采样过程保证了随机性，所以就算不剪枝，也不会出现过拟合。 | 优点：   1. 不易出现过拟合； 2. 只需要从全部特征中选取一部分进行训练得到分类器； 3. 一般其效果比bagging效果好；   缺点：  1、对于各类别样本数量不一致的数据，在决策树当中，信息增益的结果偏向于那些具有更多数值的特征；（只要使用了信息增益，都有这个缺点。 |
| 4.3 Gradient boosting /GBRT(梯度提升回归树)  是任意一个不同损失函数的泛化。  其中GBRT可以用于分类和回归中。 | 优点：   1. 可以用于混合数据类型的自然处理； 2. 预测能力强； 3. 健壮的输出空间；   缺点： |
| 4.4 xgboost（可分类可回归） | 优点：   1. 高准确率，高效率，高并发，支持自定义损失函数。 2. 可像随机森林一样输出特征重要性，因为速度快，适用于高维特征选择； 3. 在目标函数中加入正则项，控制模型的复杂度，避免过拟合； 4. 支持列抽样，增强模型的稳定性； 5. 对缺失值不敏感，可以学习到包含缺失值的特征的分裂方向；   缺点： |
| 1. SVM支持向量机   需要对数据提前归一化；  核函数的选择技巧（libsvm中自带了四种核函数：线性核、多项式核、RBF以及Sigmoid核）：   1. 样本数量小于特征数，选择简单的线性核即可；（也可以先对数据降维，再使用非线性核） 2. 样本数量大于特征数，可使用非线性核，将样本映射至更高维； 3. 样本数量等于特征数，采取2中方法； | 优点：   1. 高准确率； 2. 避免过拟合，提高了泛化能力； 3. 可以解决高维问题； 4. 利用核函数可以改变数据在原特征空间线性不可分的性质；（非线性特征的相互作用） 5. 无需依赖整个数据；   缺点：   1. 内存消耗大（观测样本较大的时候，效率低）； 2. 面对非线性问题，有时候难以找到一个合适的核函数； 3. 对缺失数据敏感； 4. 难以解释； 5. 调参繁琐； |
| 1. 人工神经网络 | 优点：   1. 分类的准确度高； 2. 并行分布处理能力能力强；（分布存储及学习能力）； 3. 在处理复杂的非线性关系上有优势，对噪声神经有较强的鲁棒性和容错能力； 4. 具备联想记忆的功能；   缺点：   1. 神经网络需要大量的参数（例如：网络拓扑结构、权值以及阈值的初始值）； 2. 难以解释； 3. 学习时间过长，有可能达不到学习的目的； |
| 1. K-means聚类 | 优点：   1. 算法简单，易于实现； 2. 在处理大数据集时，该算法是相对可伸缩和高效率的；（此算法通常局部收敛）   缺点：   1. 对数据类型要求较高，适合数值型数据； 2. 在大规模数据上收敛较慢； 3. K值难以选取； 4. 对初值的簇心值敏感，不同的初始值，可能会导致不同的聚类结果； 5. 不适合发现非凸面形状的簇，或者大小差别很大的簇； 6. 对于噪声和孤立点数据敏感，少量的该类数据对平均值产生极大的影响。 |

算法选择技巧：

首先选择逻辑回归，若其效果不佳，可以将其结果作为基准来参考，在此基础上和其他算法进行比较；

然后，试试决策树，看看是否可以大幅度提升你的模型性能。即便最后你没有将其作为最终模型，也可以使用随机森林来移除噪声变量，进行特征选择；

当特征数量和观测样本特别多，并且资源和时间充足时（前提条件），SVM此时也是一个较好的选择；

通常情况下：

Xgboost>=GBDT>=SVM>=RF>=Adaboost>=Other...

好的数据优于好的算法，特征的设计至关重要。当算法在分类性能上没有太大的区别时，我们可以考虑其速度和易用性进行抉择。