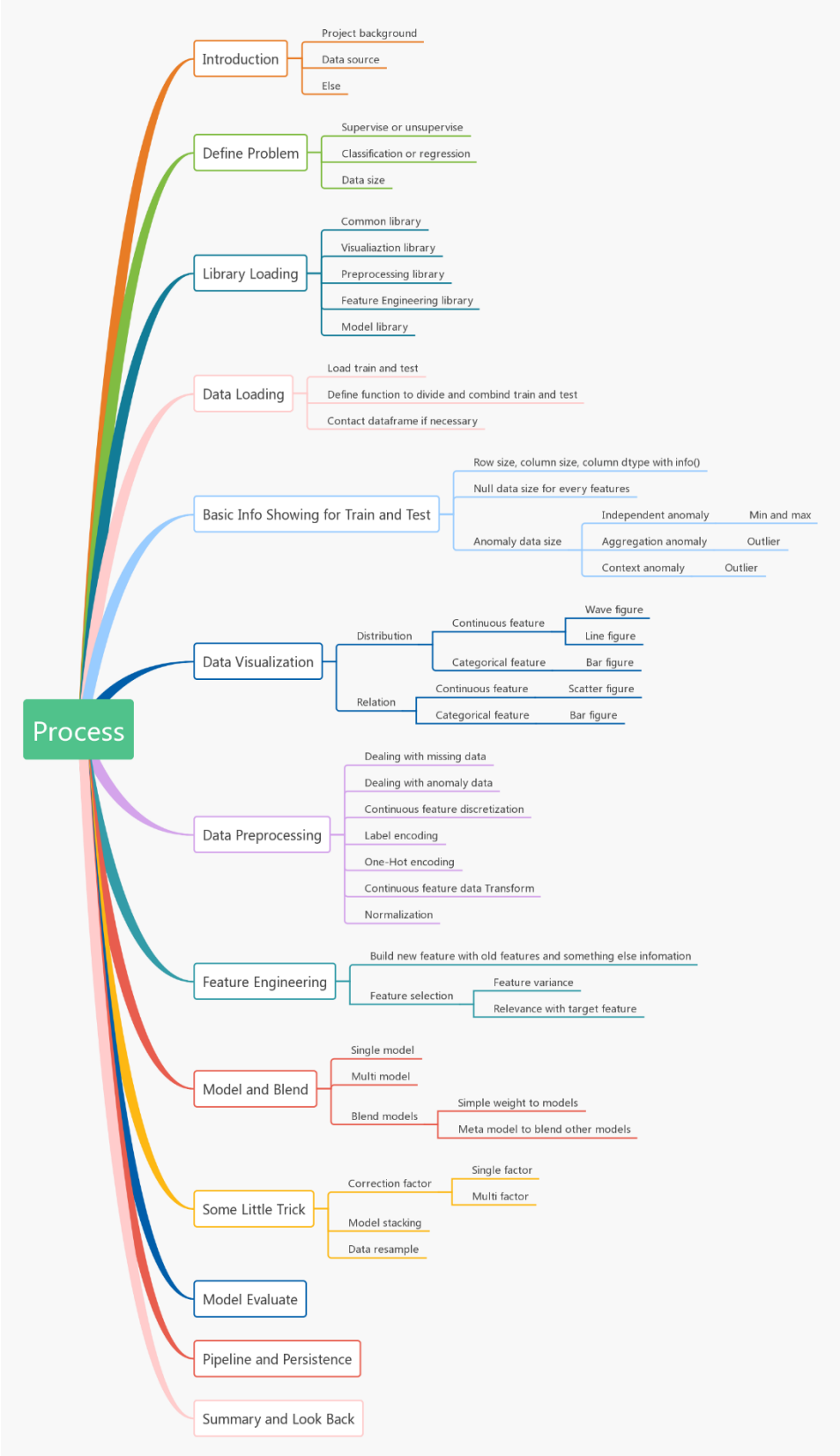
机器学习的工作流程

ML 是一个流程性很强的工作（所以很多人后面会用 PipeLine），数据采集、数据清洗、数据预处理、特征工程、模型调优、模型融合、模型验证、模型持久化；



1 数据采集

所有的机器学习算法在应用场景、优势劣势、对数据要求、运行速度上都各有优劣，但有一点不变的是都是数据贪婪的，也就是说任何一个算法，都可以通过增加数据来达到更好的结果，因此第一步数据采集也是最基础，最终的一步；
几种方式介绍：

- 爬虫：这种通常在个人项目、公司资源不足以提供数据、原始数据不足需要扩展数据情况下使用较多，比如根据时间获取天气数据，一般都是通过爬虫爬来的；
- API:现在有很多公开的数据集,一些组织也提供开放的 API 接口来获取相关数据，比如 OpenDota 提供 Dota2 相关数据，好处是通常数据更加规范；
- 数据库：这种算是最常见，也最理想的状态，通过公司自身的数据库保存数据，更加可控，也更加自由灵活；

2 数据清洗

更多是针对类似爬虫这种方式获取的数据,这种数据通常没有一个非常固定规范的格式,数据非常不稳定,因此需要进行前期的清洗工作,工作量巨大,约占整个项目工作量的 70%。
几种清洗方向：

检查数据合理性：比如爬到的数据是否满足需求；
检查数据有效性：爬到的数据量是否足够大，以及是否都是相关数据；
检查工具：爬虫工具是否有 bug；
现实世界的数据是“脏的 (Dirty Data)”

数据缺失：记录为空&属性为空

bug 导致缺失：因为程序 bug 导致缺失，这种缺失通常是少数的，一般都需要进行某种方式的填充；

正常业务情况导致缺失：比如性别字段本身就是可以不填的，那么性别就存在缺失，且这种缺失可能是大量的，这里就要首先评估该字段的重要性以及缺失率，再考虑是填充，还是丢弃；

数据重复：完全重复&不完全重复

数据错误：异常值&不一致

绝对异常：比如人的年龄 200 岁，这个数据放到什么场景下都是异常；
统计异常：比如某用户 1 分钟内登录 100 次，虽然每次登录都是正常的，但统计起来发现是异常的（可能是脚本在自动操作）；
上下文异常：比如冬天的北京，晚上温度为 30℃，虽然看数据是正常，但是跟当前的日期、时间一关联，发现是异常；

数据不可用：数据正确，但不可用

3 数据处理

（1）无量纲化

在机器学习算法实践中，我们往往有着将不同规格的数据转换到同一规格，或不同分布的数据转换到某个特定分布的需求，这种需求统称为将数据“无量纲化”。譬如梯度和矩阵为核心的算法中，譬如逻辑回归，支持向量机，神经网络，无量纲化可以加快求解速度；而在距离类模型，譬如K近邻，K-Means聚类中，无量纲化可以帮我们提升模型精度，避免某一个取值范围特别大的特征对距离计算造成影响。（一个特例是决策树和树的集成算法，对决策树我们不需要无量纲化，决策树可以把任意数据都处理得很好。）

（2）缺失值

（3）定量特征离散化

将连续型变量划分为分类变量类别，能够将连续型变量排序后按顺序分箱后编码。例如：把年龄按照 10,20,40,60 划分为幼年、青年、中年和老年，并进行 onehot 或者 ordinal 编码。

（4）定性特征**哑编码**

在机器学习中，大多数算法，譬如逻辑回归，支持向量机SVM，k近邻算法等都只能够处理**数值型数据**，不能处理文字，在sklearn当中，除了专用来处理文字的算法，其他算法在 fit 的时候全部要求输入数组或矩阵，也不能够导入文字型数据（其实手写决策树和普斯贝叶斯可以处理文字，但是 sklearn 中规定必须导入数值型）。

然而在现实中，许多标签和特征在数据收集完毕的时候，都不是以数字来表现的。比如说，学历的取值可以是["小学"，“初中”，“高中”，"大学"]，付费方式可能包含["支付宝"，“现金”，“微信”]等等。在这种情况下，为了让数据适应算法和库，我们必须将数据进行编码，即是说，将文字型数据转换为数值型。

（5）文本特征向量化

机器学习的模型算法均要求输入的数据必须是数值型的，所以对于文本类型的特征属性，需要进行文本数据转换，也就是需要将文本数据转换为数值型数据。常用方式如下：词袋模型(BOW/TF)、TF-IDF(Term frequency-inverse document frequency)模型、HashTF模型、Word2Vec模型(主要用于单词的相似性考量)。

4 特征工程

特征工程决定了机器学习的上限，模型只是逼近这个上限；

这绝对不是一句空话，以目前在Kaggle上看到的各个比赛的情况，基本胜负都是出在特征工程上，这一点也是我认为机器学习中最重要，也最难的部分，它难并不是技术上的，而是经验上的，一个经验丰富的Kaggler在看到项目、数据的同时，脑子里已经有了特征工程的雏形，这可以帮助他很快的得到一个不错的分数，而后续的优化上，经验也是最重要的参考；

基本步骤：

（1）特征构建：

特征组合：例如组合日期、时间两个特征，构建是否为上班时间(工作日的工作时间为1，其他为0)特征，特征组合的目的通常是为了获得更具有表达力、信息量的新特征；

特征拆分：将业务上复杂的特征拆分开，比如将登录特征，拆分为多个维度的登录次数统计特征，拆分为多个的好处一个是从多个维度表达信息，另一个多个特征可以进行更多的

组合；

外部关联特征：例如通过时间信息关联到天气信息，这种做法是很有意义的，首先天气数据不是原始数据集的，因此这样想当于丰富了原始数据，通常来讲会得到一个比仅仅使用原始数据更好的结果，不仅仅是天气，很多信息都可以这样关联（比如在一个 Kaggle 上的房屋预测问题上,可以通过年份关联到当时的一些地方政策、国际大事等等,都是有影响的，比如金融危机）；

（2）特征选择：

特征自身的取值分布：主要通过方差过滤法，比如性别特征，1000 个数据，999 个是男的，1 个是女的，这种特征由于自身过于偏斜，因此是无法对结果起到足够的帮助；

特征与目标的相关性：可以通过皮尔逊系数、信息熵增益等来判断，思路是如果一个特征与目标的变化是高度一致的，那么它对于预测目标就是具有很大指导意义的；

5 模型调优

同一个模型不同参数下的表现依然是天差地别,通常在特征工程部分结束后就进入到模型参数调优的步骤，这一步也是最无聊最耗时间的（反正我家电脑经常跑一晚上），由于 Kaggle 上个人项目一般都是在家做，因此个人电脑的性能大家都懂的，因此一个好的技巧还是比较实用的；

流程：模型选择——>交叉验证——>结果评估——>超参选择。

调参方式与工具：

首先工具选择：GridSearch，也确实比较方便；

调参顺序上，建议是先重要的影响大的参数，后没那么重要的，影响小的参数；

举例随机森林：作为集成方法中最常用的模型之一，通常第一个调的是 n_estimator 即树的个数,其次是学习率,再其次是 max_feature,会发现这三个都是随机森林自身的参数，后面可以细调每棵树的参数，比如最小分裂样本数等等；

6 模型融合

一般来讲，任何一个模型在预测上都无法达到一个很好的结果，这是因为通常来说单个模型无法拟合所有数据,不具备对所有未知数据的泛化能力,因此需要对多个模型进行融合，这一点在 Kaggle 上体现的也很明显，好的排名中基本都用了模型融合；

融合方式：

简单融合：

分类问题：投票法融合，不考虑单个模型自身的得分；

回归问题：假设每个模型权重一致，计算融合结果；

加权融合：基本同上，区别是考虑每个模型自身得分，得分高的权重大；

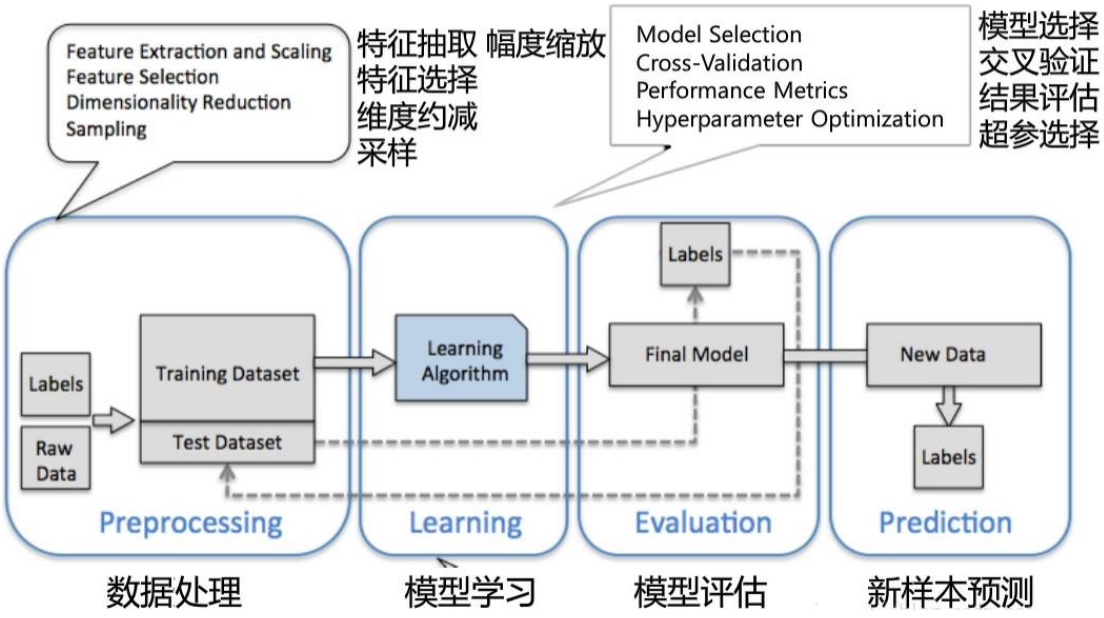
使用模型进行融合：即将多个单模型的输出作为输入送入到某个模型中，让模型去做融合，通常可以达到最好的效果，但是由于用到了模型，因此要注意过拟合问题；

7 模型验证

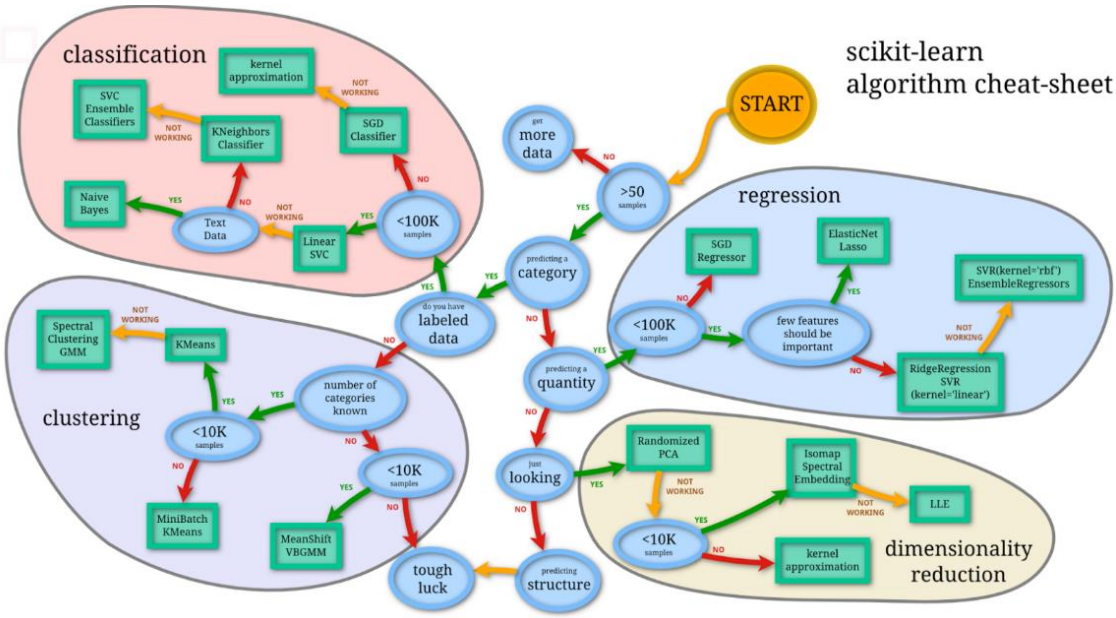
通过交叉验证对模型性能进行检验，这里通常都是一致的做法，需要注意的是在**时间序列数据预测**上，不能直接随机的划分数据，而是要考虑时间属性，因为很多特征都依赖于时间的前后关系，利用了趋势；

8 模型持久化

最后，最好将得到的模型持久化到磁盘，方便后续使用、优化时，不需要从头开始；



sklearn 流程图


scikit-learn
algorithm cheat-sheet
START
get more data
>50 samples
predicting a category
do you have labeled data
predicting a quantity
just looking
predicting structure
tough luck
classification
kernel approximation
SVC Ensemble Classifiers
KNeighbors Classifier
SGD Classifier
Naive Bayes
Text Data
Linear SVC
<100K samples
regression
SGD Regressor
ElasticNet Lasso
SVR(kernel='rbf') EnsembleRegressors
<100K samples
few features should be important
RidgeRegression SVR (kernel='linear')
clustering
Spectral Clustering GMM
KMeans
number of categories known
<10K samples
MiniBatch KMeans
MeanShift VBGMM
dimensionality reduction
Randomized PCA
Isomap Spectral Embedding
LLE
<10K samples
kernel approximation
YES
NO
NOT WORKING