Microsoft

人工智能系统 System for AI

自动机器学习系统 AutoML Systems

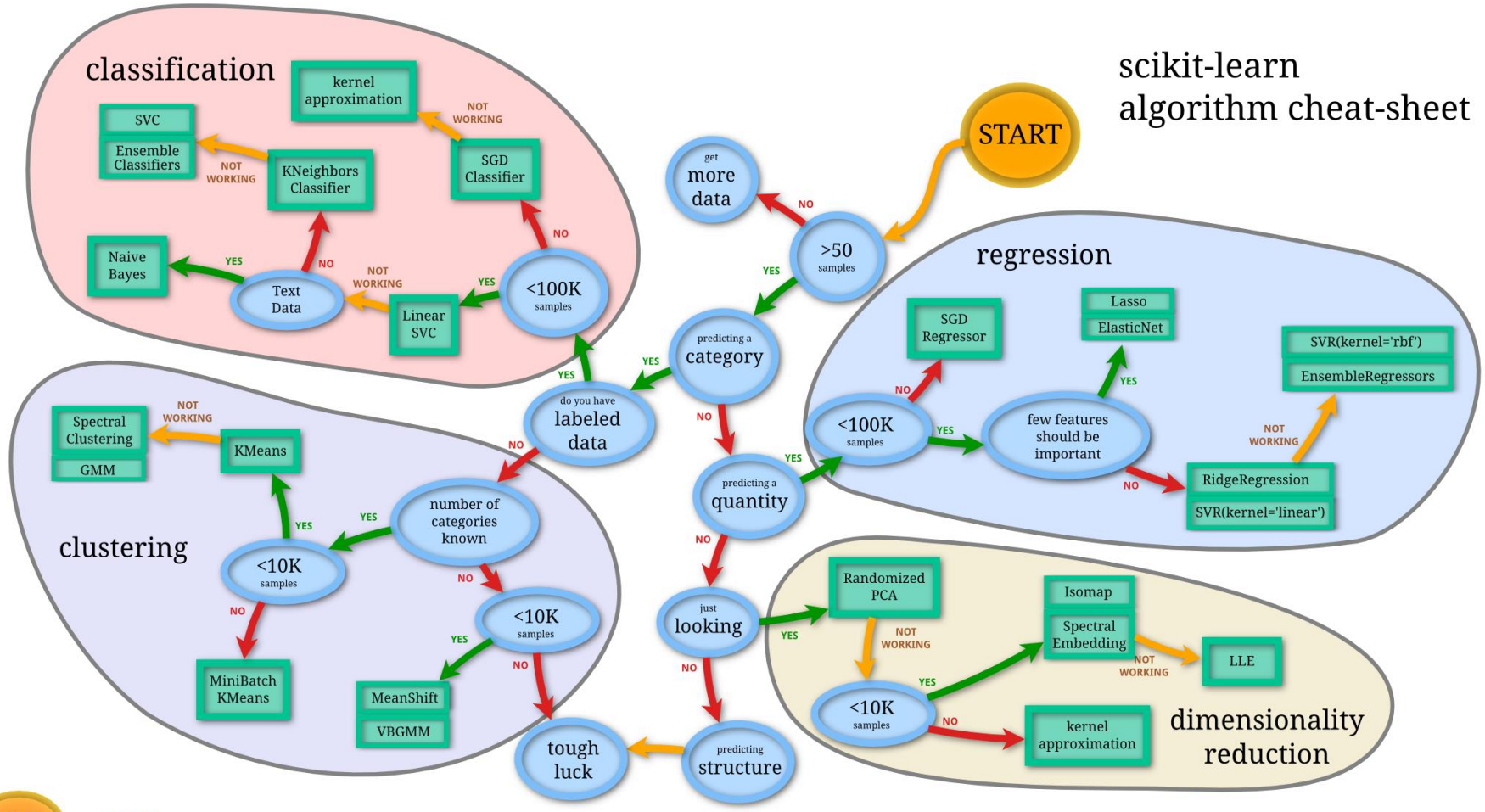
主要内容

- 机器学习建模的挑战
- 自动机器学习概述与原理
- 主要算法介绍
- 应用
- 自动机器学习系统
- 现状

机器学习建模挑战

- 算法繁多、新算法层出不穷
- 模型调参工作复杂、难寻规律
- 计算资源与人工分析交替成为瓶颈

挑战 - 算法繁多

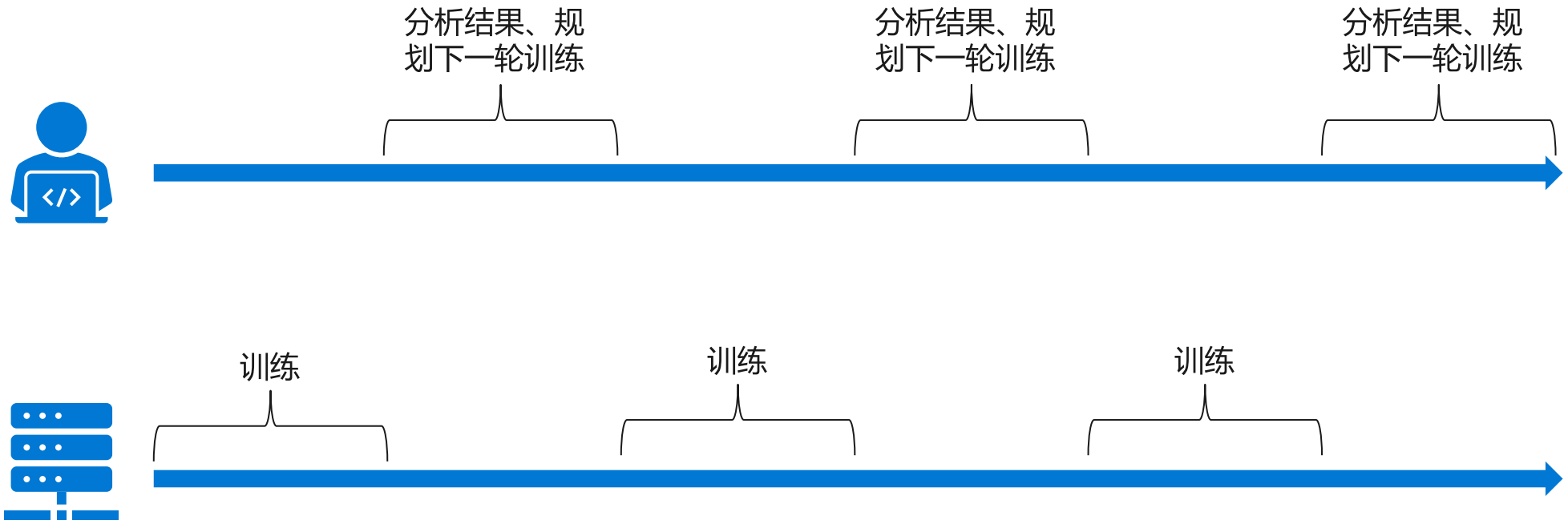


Reference: https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html

挑战 - 超参复杂

参数	类型	RandomForestClassifier	RandomForestRegressor	GradientBoostingClassifier	GradientBoostingRegressor
loss	目标			损失函数 ● exponential：模型等同AdaBoost ★ deviance：和Logistic Regression的损失函数一致	损失函数 ● exponential：模型等同AdaBoost ★ deviance：和Logistic Regression的损失函数一致
alpha	目标			损失函数为huber或quantile的时，alpha为损失函数中的参数	损失函数为huber或quantile的时，alpha为损失函数中的参数
class_weight	目标	类别的权值			
n_estimators	性能	子模型的数量 ● int：个数 ★ 10：默认值	子模型的数量● int：个数 ★ 10：默认值	子模型的数量● int：个数 ★ 100：默认值	子模型的数量● int：个数 ★ 100：默认值
learning_rate	性能			学习率（缩减）	学习率（缩减）
criterion	性能	判断节点是否继续分裂采用的计算方法 ● entropy ★ gini	判断节点是否继续分裂采用的计算方法 ★ mse		
max_features	性能	节点分裂时参与判断的最大特征数 ● int：个数 ● float：占所有特征的百分比 ★ auto：所有特征数的开方 ● sqrt：所有特征数的开方 ● log2：所有特征数的log2值 ● None：等于所有特征数	节点分裂时参与判断的最大特征数 ● int：个数 ● float：占所有特征的百分比 ★ auto：所有特征数的开方 ● sqrt：所有特征数的开方 ● log2：所有特征数的log2值 ● None：等于所有特征数	节点分裂时参与判断的最大特征数 ● int：个数 ● float：占所有特征的百分比 ● auto：所有特征数的开方 ● sqrt：所有特征数的开方 ● log2：所有特征数的log2值 ★ None：等于所有特征数	节点分裂时参与判断的最大特征数 ● int：个数 ● float：占所有特征的百分比 ● auto：所有特征数的开方 ● sqrt：所有特征数的开方 ● log2：所有特征数的log2值 ★ None：等于所有特征数
subsample	性能			子采样率 ● float：采样率 ★ 1.0：默认值	子采样率 ● float：采样率 ★ 1.0：默认值
init	性能			初始子模型	初始子模型
n_jobs	效率	并行数 ● int：个数 ● -1：跟CPU核数一致 ★ 1:默认值	并行数 ● int：个数 ● -1：跟CPU核数一致 ★ 1:默认值		
warm_start	效率	是否热启动，如果是，则下一次训练是以追加树的形式进行 ● bool：热启动 ★ False：默认值	是否热启动，如果是，则下一次训练是以追加树的形式进行 ● bool：热启动 ★ False：默认值	是否热启动，如果是，则下一次训练是以追加树的形式进行 ● bool：热启动 ★ False：默认值	是否热启动，如果是，则下一次训练是以追加树的形式进行 ● bool：热启动 ★ False：默认值
presort	效率			是否预排序,预排序可以加速查找最佳分裂点，对于稀疏数据不管用● Bool ★ auto：非稀疏数据则预排序，若稀疏数据则不预排序	是否预排序,预排序可以加速查找最佳分裂点，对于稀疏数据不管用● Bool ★ auto：非稀疏数据则预排序，若稀疏数据则不预排序
oob_score	附加	是否计算袋外得分 ★ False：默认值	是否计算袋外得分 ★ False：默认值		
random_state	附加	随机器对象	随机器对象	随机器对象	随机器对象

挑战 - 调优过程低效



应对挑战

- 总结经验
  - 确定可推广的规律
  - 发现不确定的范畴
- 提升效率
  - 更快、更好的规划计算资源

自动机器学习

- 构建于机器学习算法上的学习超参的算法 - 机器学习的机器学习。用机器学习的方法来学习超参。

自动机器学习 - 方向

- 超参数优化
  - 应用广泛
  - 其它方向也大量应用超参优化相关算法
  - 可推广到系统优化等非自动机器学习领域
- 神经网络架构搜索
  - 应用于深度学习
  - 前景广阔
- 特征选择
  - 应用于经典机器学习算法
- 模型压缩
  - 应用于深度学习
  - 新兴方向

与机器学习算法的比较

- 训练的特征（即超参数）较少，算法较简单。
- 训练结果可相互独立，找出指标最好的结果即可。
- 交互式学习，需要生成数据，与强化学习相似。
- 作为自动机器学习算法，其本身的超参数应越少越好，需要良好的自适应性。

机器学习概念 - 参数与超参数

- 参数（Parameters）：几十到上百亿个数值，大部分是矩阵和向量。通过数据让计算机不断调优这些参数，从而能学习到数据中的规律，即知识。
- 超参数（Hyper-parameters）：需要手工设定，模型不易自动调优的参数。如：算法，循环次数，学习率，神经元数量，神经网络结构等。

机器学习概念 - 超参数：经典机器学习算法

- **DecisionTreeClassifier**(*, *criterion='gini'*, *splitter='best'*, *max_depth=None*, *min_samples_split=2*, *min_samples_leaf=1*, *min_weight_fraction_leaf=0.0*, *max_features=None*, *random_state=None*, *max_leaf_nodes=None*, *min_impurity_decrease=0.0*, *min_impurity_split=None*, *class_weight=None*, *presort='deprecated'*, *ccp_alpha=0.0*)
- **LinearSVC**(*penalty='l2'*, *loss='squared_hinge'*, *, *dual=True*, *tol=0.0001*, *C=1.0*, *multi_class='ovr'*, *fit_intercept=True*, *intercept_scaling=1*, *class_weight=None*, *verbose=0*, *random_state=None*, *max_iter=1000*)
- **GradientBoostingClassifier**(*, *loss='deviance'*, *learning_rate=0.1*, *n_estimators=100*, *subsample=1.0*, *criterion='friedman_mse'*, *min_samples_split=2*, *min_samples_leaf=1*, *min_weight_fraction_leaf=0.0*, *max_depth=3*, *min_impurity_decrease=0.0*, *min_impurity_split=None*, *init=None*, *random_state=None*, *max_features=None*, *verbose=0*, *max_leaf_nodes=None*, *warm_start=False*, *presort='deprecated'*, *validation_fraction=0.1*, *n_iter_no_change=None*, *tol=0.0001*, *ccp_alpha=0.0*)
- **LogisticRegression**(*penalty='l2'*, *, *dual=False*, *tol=0.0001*, *C=1.0*, *fit_intercept=True*, *intercept_scaling=1*, *class_weight=None*, *random_state=None*, *solver='lbfgs'*, *max_iter=100*, *multi_class='auto'*, *verbose=0*, *warm_start=False*, *n_jobs=None*, *l1_ratio=None*)
- **SGDRegressor**(*loss='squared_loss'*, *, *penalty='l2'*, *alpha=0.0001*, *l1_ratio=0.15*, *fit_intercept=True*, *max_iter=1000*, *tol=0.001*, *shuffle=True*, *verbose=0*, *epsilon=0.1*, *random_state=None*, *learning_rate='invscaling'*, *eta0=0.01*, *power_t=0.25*, *early_stopping=False*, *validation_fraction=0.1*, *n_iter_no_change=5*, *warm_start=False*, *average=False*)

机器学习概念 - 超参数：深度学习算法

parser.add_argument('--batch-size', type=int, default=64, metavar='N', help='input batch size for training (default: 64)')

parser.add_argument('--test-batch-size', type=int, default=1000, metavar='N', help='input batch size for testing (default: 1000)')

parser.add_argument('--epochs', type=int, default=14, metavar='N', help='number of epochs to train (default: 14)')

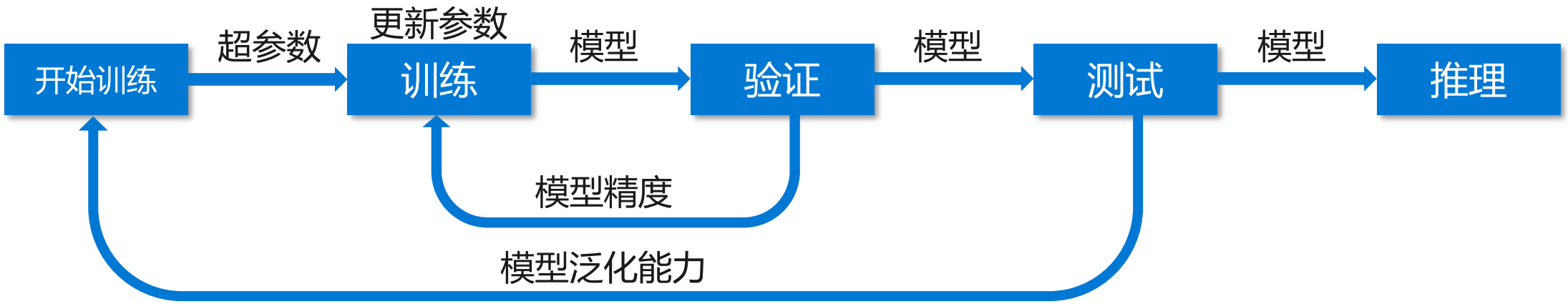
parser.add_argument('--lr', type=float, default=1.0, metavar='LR', help='learning rate (default: 1.0)')

parser.add_argument('--gamma', type=float, default=0.7, metavar='M', help='Learning rate step gamma (default: 0.7)')

机器学习概念 - 超参数：学习率

- 不同的算法、数据集的最优学习率不同。
- 学习率太大，结果震荡，不易收敛。
- 学习率太小，训练时间长，容易落入局部最优。

机器学习概念 - 参数与超参数



模型：参数和部分超参数

自动机器学习概念 - 搜索空间

- 选择超参，并限定其取值范围。
  - 搜索空间大，不易漏掉最佳超参组合，但需要的计算资源也多。
  - 搜索空间小，需要的计算资源少，但有可能漏掉了较好的超参组合。
- 常用类型
  - 枚举
  - 整数
  - 浮点
    - 均匀分布、正态分布、指数分布、正态指数分布：提高搜索效率
    - 分段：减少搜索空间

自动机器学习概念 - 算法

- 优化算法
  - 尽量利用已有信息来获得更优结果。
  - 算法：随机、遍历、进化、模拟退火、贝叶斯优化等。
- 评估算法
  - 训练过程中，评估最终结果好坏的可能性。可提前终止较差结果的执行，从而节省计算资源。
  - 算法：中位数、曲线拟合等。

自动机器学习中的概念 – 挖掘与探索

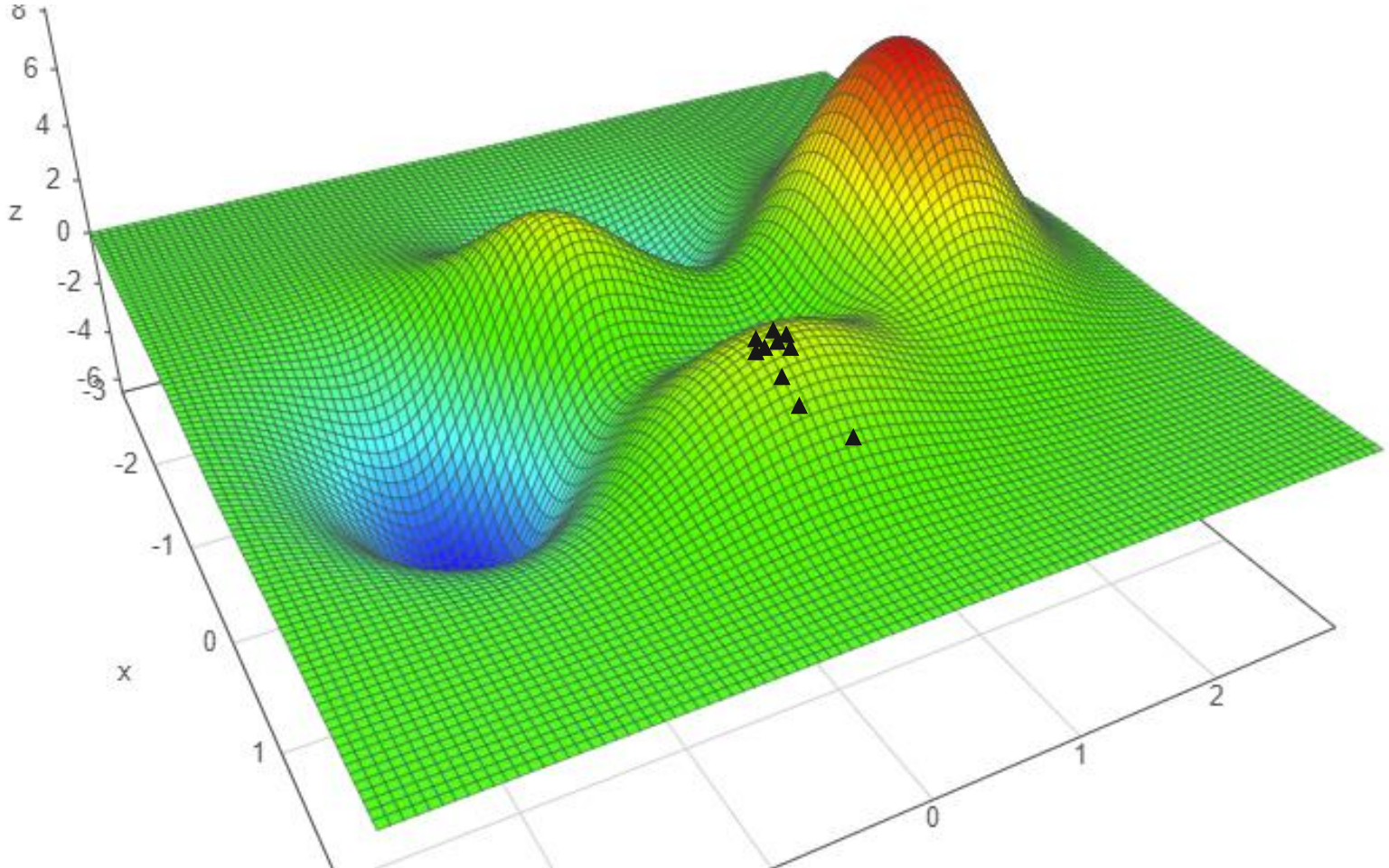
- 生成超参数的策略
- 挖掘（Exploitation，或作“开发，利用”）
  - 在较好结果的搜索空间附近，继续尝试是否能找到局部最优结果。
- 探索（Exploration）
  - 在覆盖较少的搜索空间进行探索，减少遗漏全局最优结果。
- 自动机器学习算法要在两者间取得平衡，有的算法会提供参数来控制两者的比例。

挖掘（Exploitation）与探索（Exploration）

- 生成超参数的平衡策略
- 社会性动物采用类似的策略
  - 蚂蚁在寻找食物时，会四处探索。找到食物后，一部分会聚集到食物附近，开发食物资源。
  - 有的人倾向于探索新事物，扩展人类各方面（体育、科研、创新）的边界。有的人倾向于守成，让人类社会稳定存在。

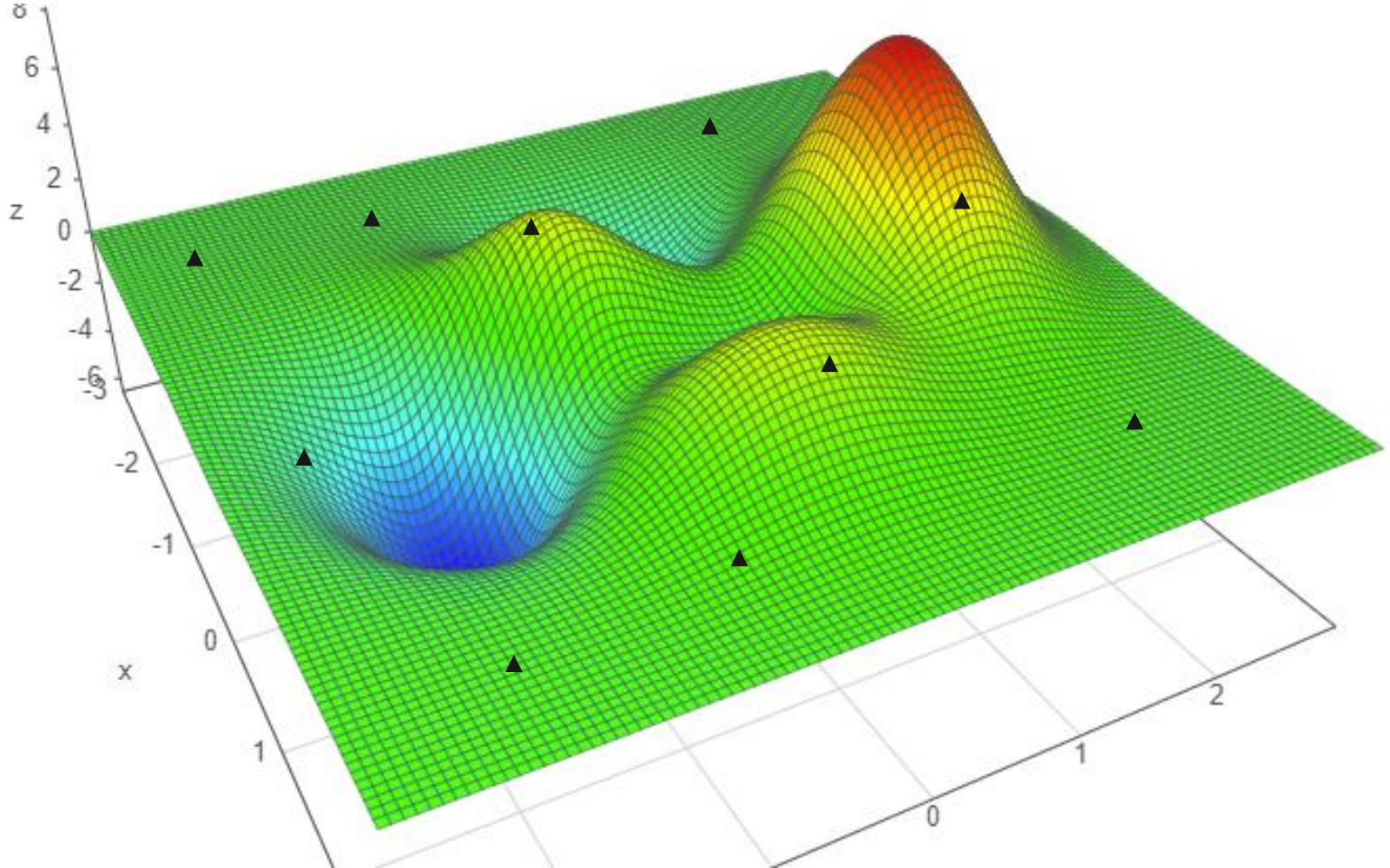
挖掘（Exploitation）与探索（Exploration）

- 挖掘（Exploitation ）
  - 在较好结果的搜索空间附近，继续尝试是否能找到局部最优结果。
  - 典型算法：贪心算法。



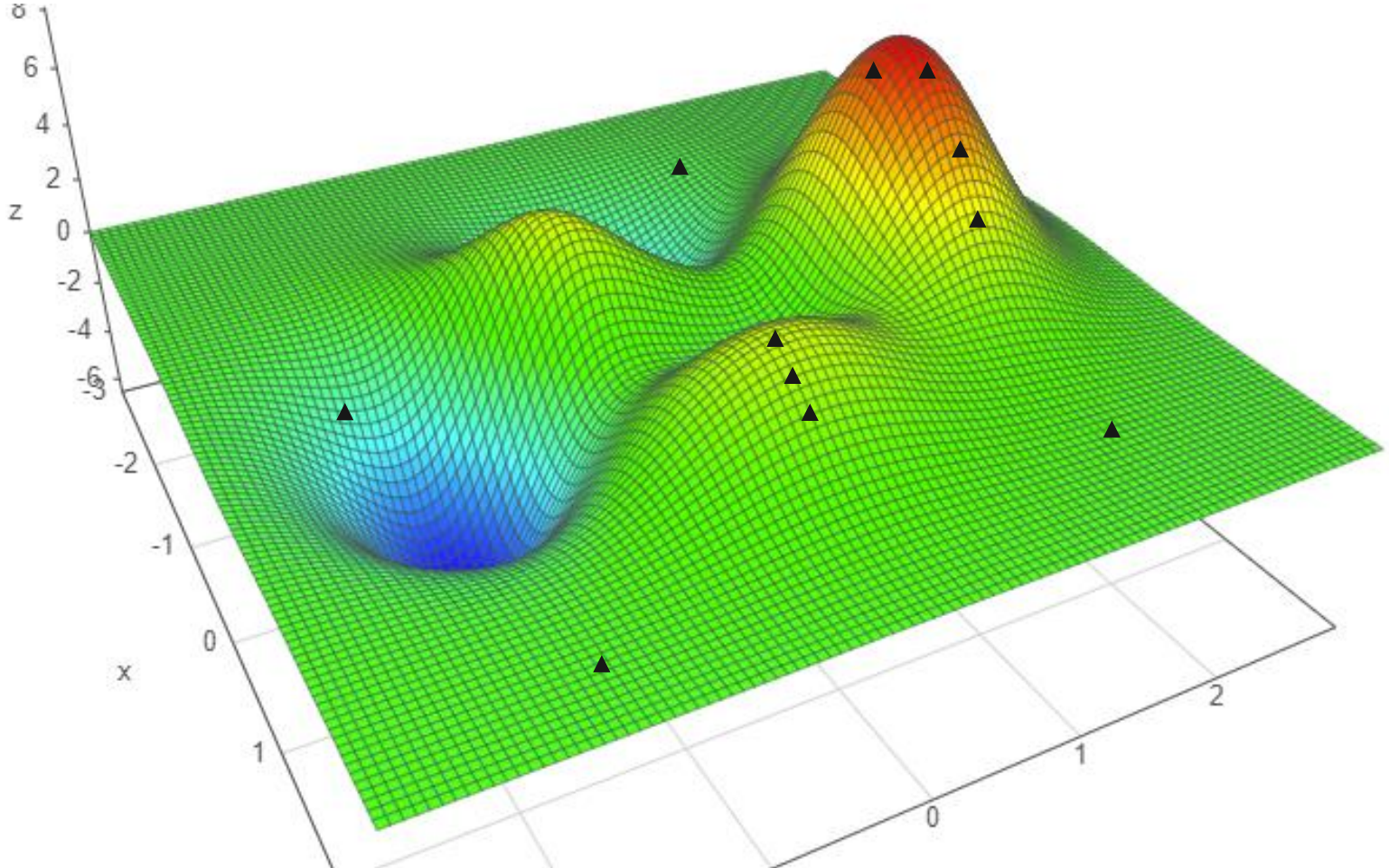
挖掘（Exploitation）与探索（Exploration）

- 探索（Exploration）
  - 在覆盖较少的搜索空间进行探索，减少遗漏全局最优结果。
  - 典型算法：随机



挖掘（Exploitation）与探索（Exploration）

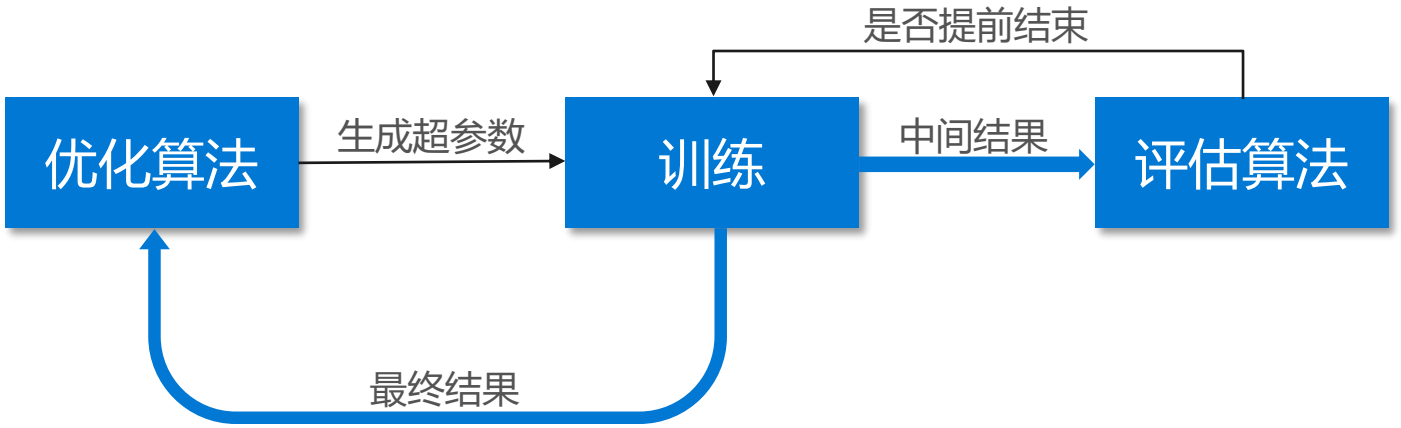
好的自动机器学习算法要在两者间取得平衡，有的算法会提供参数来控制两者的比例。



本章小结

- 回顾
  - 机器学习中的挑战
  - 自动机器学习中的重要概念
    - 搜索空间
    - 自动机器学习主要算法类别：优化算法、评估算法
    - 挖掘（Exploitation）与探索（Exploration）
- 思考
  - 根据自己的机器学习经验，对某个算法的超参进行分析。看看哪些经验是可推广的，哪些是不确定的，需要试验的？
  - 自动机器学习算法与一般机器学习算法相比，有哪些共同点和特点？

自动机器学习概念 - 算法



中间结果、最终结果：即模型的优化目标，如精度（Accuracy）、损失值（Loss）

优化算法类型

- 穷举搜索 Exhaustive Search
- 启发式搜索 Heuristic search
- 贝叶斯优化 Bayesian optimization
- 其他优化算法

优化算法 – 随机

· 随机选择每个超参数。

	随机
数值类型	支持所有数值类型
历史数据	不利用
开发与探索的平衡	相当于只探索，但无探索策略
搜索空间大小的适应	大搜索空间不易找到较优结果
其它特点	通常也能得出可接受的结果，可作为其它优化算法的基准

优化算法 – 遍历（Grid Search）

· 遍历整个搜索空间，找到确定的最优解。

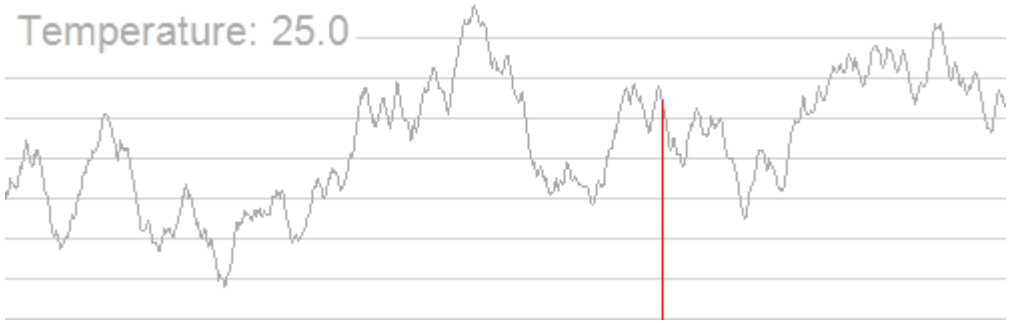
	遍历
数值类型	支持有限的数值类型，如枚举、整型、以及分段的浮点数
历史数据	不利用
开发与探索的平衡	尝试所有可能性
搜索空间大小的适应	搜索空间大小直接影响搜索时间
其它特点	仅适用于非常小的搜索空间

优化算法 – 模拟退火

- 固体在高温时，内部粒子呈无序状。让其徐徐冷却，粒子渐趋有序。通过此过程，让粒子间内能最小，更稳定。
- 模拟退火算法，从较高的初始温度开始，出发, 这个温度称为初始温度, 伴随着温度参数的不断下降, 算法中的解趋于稳定, 但是, 可能这样的稳定解是一个局部最优解, 此时, 模拟退火算法中会以一定的概率跳出这样的局部最优解, 以寻找目标函数的全局最优解. 这个概率就与温度参数的大小有关, 温度参数越大, 概率越大, 反之则越小.

优化算法 – 模拟退火

- 综合随机搜索与贪心算法。
  - 初期高温，有较高概率选择较差结果。
  - 后期温度下降，更倾向于选择较好结果。
- 若新结果较好则直接采用，否则以一定概率采用。



优化算法 – 模拟退火

```
// From: https://www.cnblogs.com/heaad/archive/2010/12/20/1911614.html
/*
* J(y)：在状态y时的评价函数值     Y(i)：表示当前状态      Y(i+1)：表示新的状态
* r:  用于控制降温的快慢          T:  系统的温度，系统初始应该要处于一个高温的状态
* T_min ：温度的下限，若温度T达到T_min，则停止搜索
*/
while( T > T_min )
{
    dE = J( Y(i+1) ) - J( Y(i) ) ;

    if ( dE >=0 ) //表达移动后得到更优解，则总是接受移动
         Y(i+1) = Y(i) ; //接受从Y(i)到Y(i+1)的移动
    else
    {
         // dE < 0, 函数exp( dE/T )的取值范围是(0,1) , dE/T越大，则exp( dE/T )也越大
         if ( exp( dE/T ) > random( 0 , 1 ) )
              Y(i+1) = Y(i) ; //接受从Y(i)到Y(i+1)的移动
    }
    T = r * T ; //降温退火 , 0<r<1 。r越大，降温越慢；r越小，降温越快
    /*
    * 若r过大，则搜索到全局最优解的可能会较高，但搜索的过程也就较长。若r过小，则搜索的过程会很快，但最终可能会达到一个局部
最优值
    */
    i ++ ;
}
```

优化算法 – 模拟退火

	模拟退火
数值类型	支持各种数据类型
历史数据	使用历史数据来确定最优结果及温度
开发与探索的平衡	初期趋于探索，后期趋于挖掘
搜索空间大小的适应	大搜索空间不易找到较优结果
其它特点	

优化算法 – 进化

- 举例：遗传算法（Genetic Algorithm，GA）
  - 原理：

    繁殖过程中会发生基因交叉(Crossover)，基因突变 (Mutation)，适应度(Fitness)低的个体会被逐步淘汰，而适应度高的个体会越来越多。

    经过N代的自然选择后，保存下来的个体都是适应度很高的，其中很可能包含史上产生的适应度最高的那个个体。
  - 术语：
    - 种群(Population)：生物的进化以群体的形式进行，这样的一个群体称为种群。
    - 基因 (Gene)： 一个遗传因子。
    - 染色体 (Chromosome)： 包含一组的基因。
  - 操作：
    - 基因交叉(Crossover)
    - 基因突变 (Mutation)
    - 适应度(Fitness)低的个体会被逐步淘汰

Reference: https://www.cnblogs.com/heaad/archive/2010/12/23/1914725.html

优化算法 – 进化

- 举例：遗传算法（Genetic Algorithm，GA）
  - 过程：
    - Step 1 **种群初始化**：根据问题特性设计合适的初始化操作（初始化操作应尽量简单，时间复杂度不易过高）对种群中的N个个体进行初始化操作；
    - Step 2 **个体评价**：根据适应度函数（Fitness Function）计算种群中个体的适应值（Fitness Value）；
    - Step 3 **迭代设置**：设置种群最大迭代次数gmax，并令当前迭代次数g=1；
    - Step 4 **个体选择**：设计合适的选择算子来对种群P(g)个体进行选择，被选择的个体将进入交配池中组成父代种群FP(g)，用于交叉变换以产生新的个体。
      常用的选择策略有轮盘赌（Roulette Wheel Selection），锦标赛等。

Reference: https://www.cnblogs.com/maybe2030/p/4665837.html#_label0

优化算法 – 进化

- 举例：遗传算法（Genetic Algorithm，GA）
  - 过程：
    - Step 5 **交叉算子**：根据交叉概率p_m（预先指定，一般为0.9）来判断父代个体是否需要进行交叉操作。交叉算子要根据被优化问题的特性来设计，它是整个遗传算法的核心，其设计的好坏将直接决定整个算法性能的优劣。
    - Step 6 **变异算子**：根据变异概率p_c（预先指定，一般为0.1）来判断父代个体是否需要进行变异操作。变异算子的主要作用是保持种群的多样性，防止种群陷入局部最优，般被设计为一种随机变换。
    - 通过交叉变异操作以后父代种群FP(g)生成了新的子代种群P(g+1)，令种群迭代次数g=g+1，进行下一轮的迭代操作（跳转到Step 4），直至迭代次数达到最大的迭代次数。操作：

Reference: https://www.cnblogs.com/maybe2030/p/4665837.html#_label0

优化算法 – 进化

- 应用
  - 随机初始化多组超参进行训练（同随机算法）
  - 对结果较好的超参组合，随机变化少量参数后训练。

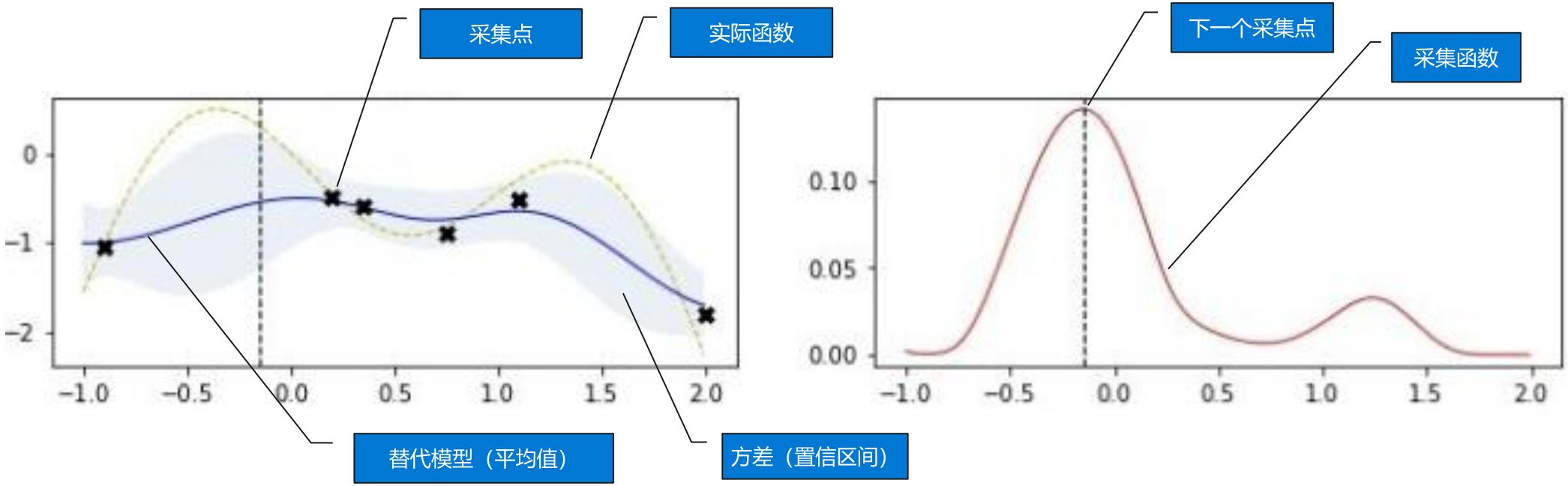
	进化
数值类型	支持各种数据类型
历史数据	使用历史数据来比较出较好的超参组合
开发与探索的平衡	初始种群越大，越倾向于探索。 变异的超参数量越多、变化范围越大，越趋向于探索。
搜索空间大小的适应	需要较多迭代次数
其它特点	

优化算法 – 贝叶斯优化

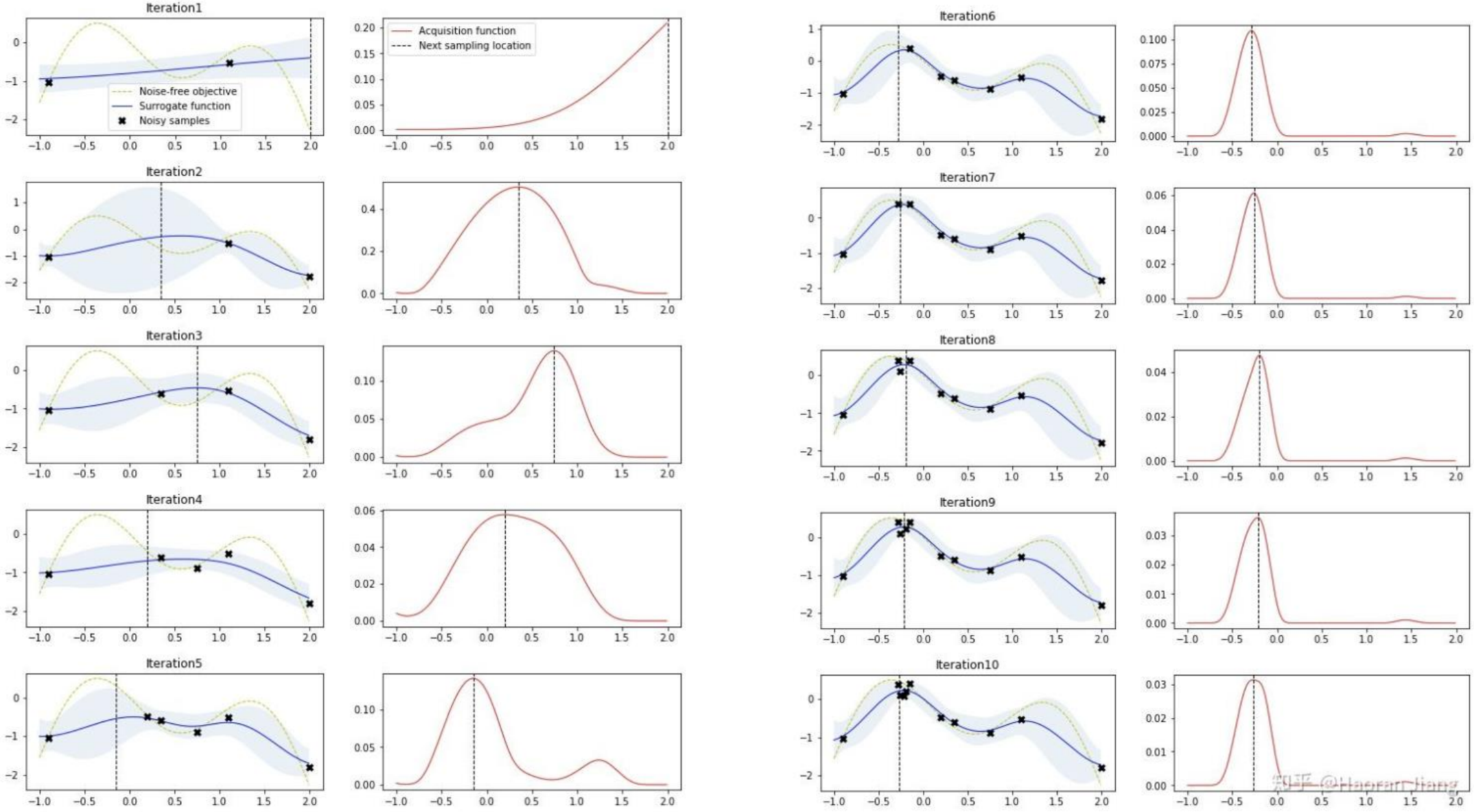
- 贝叶斯优化（Bayesian optimization）
  - 基于模型的序列化全局优化算法（Sequential Model-based Global Optimization）
- 替代模型（Surrogate model），用成本更低的函数（如，高斯过程）来拟合数据。
- 采集函数（Acquisition function），评估价值，确定下一个采集点。

优化算法 – 贝叶斯优化

- 替代模型（Surrogate model）
  - 高斯过程（Gaussian process）
- 采集函数（Acquisition function）
  - 平均值和方差加权求和



优化算法 – 贝叶斯优化


Iteration1
Noise-free objective
Surrogate function
Noisy samples
Acquisition function
Next sampling location
Iteration2
Iteration3
Iteration4
Iteration5
Iteration6
Iteration7
Iteration8
Iteration9
Iteration10

优化算法 – 贝叶斯优化

- Tree-structured Parzen Estimator (TPE)
  - Tree Parzen Estimators
- Sequential Model-based Algorithm Configuration (SMAC)
  - 随机森林

	贝叶斯优化
数值类型	支持各种数据类型，以及数据间的级联关系
历史数据	替代模型使用历史数据来评估搜索空间
开发与探索的平衡	采集函数用于平衡挖掘与探索
搜索空间大小的适应	对较大搜索空间也较有效
其它特点	特别适合解决自动机器学习这类主动学习的问题。 如果并发数量较多，优化效率会下降。

优化算法 – 其它优化算法

- 强化学习类算法
- Hyperband、BOHB (Bayesian Optimization on Hyperband)
  - 先产生大量超参组合，运行一小段时间后，停掉大部分结果不佳的。

评估算法

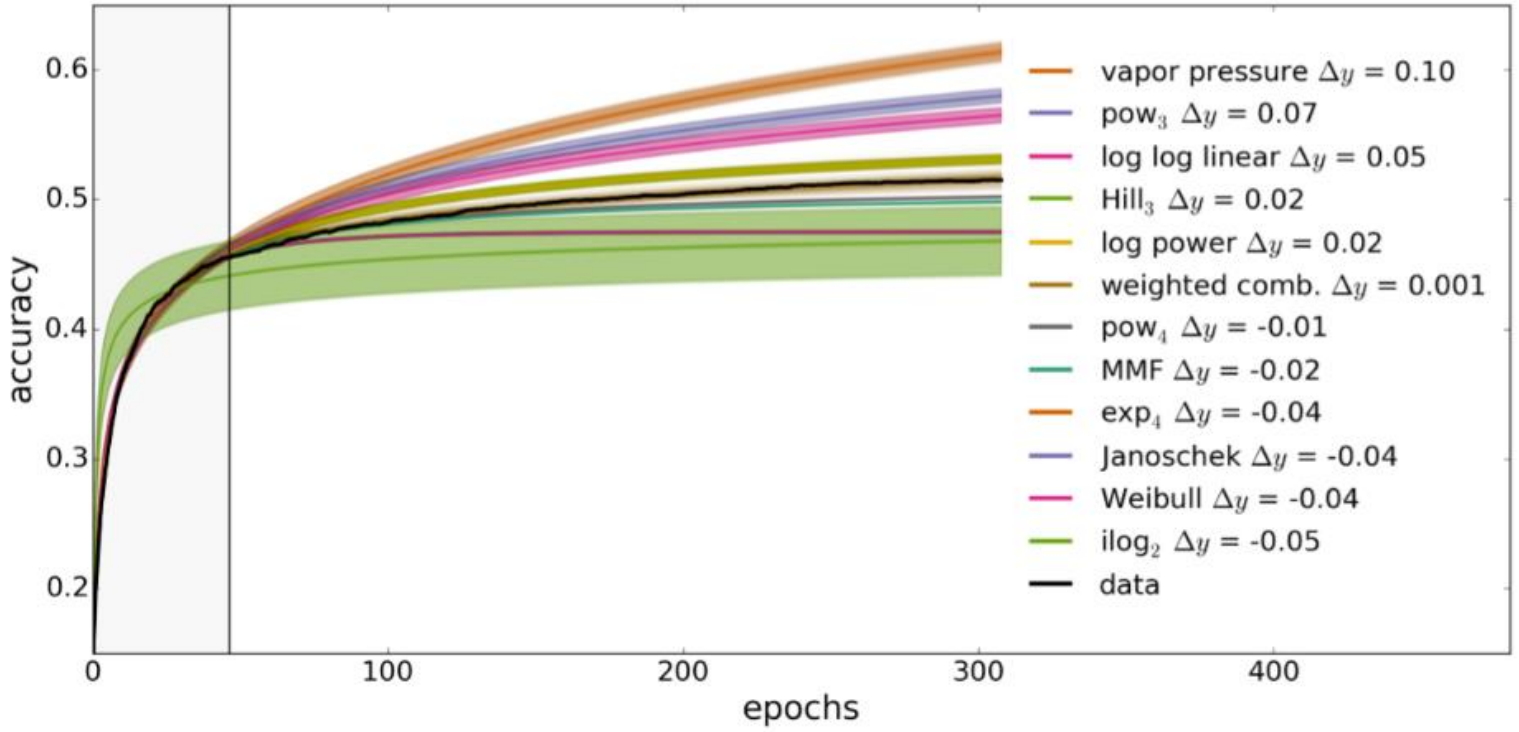
- 节约计算资源，通过训练的中间结果来提前预测最终结果。
- 常用算法
  - 中位数
  - 曲线拟合

评估算法 – 中位数

- 前 S 步的均值低于其它已完成的trial的前 S 步均值的中值，可终止。
- 计算量小，效果良好。
- Reference paper: https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/46180.pdf

评估算法 – 曲线拟合

- 对一系列常见曲线进行加权拟合。
- 计算较复杂，较精确，仅适合递增的目标值。
- Reference paper: http://aad.informatik.uni-freiburg.de/papers/15-IJCAI-Extrapolation_of_Learning_Curves.pdf



Reference name	Formula
vapor pressure	$\exp(a+\frac{b}{x}+c\log(x))$
pow_3	$c-ax^{-\alpha}$
log log linear	$\log(a\log(x)+b)$
Hill_3	$\frac{y_{\max}x^{\eta}}{\kappa^{\eta}+x^{\eta}}$
log power	$\frac{a}{1+\left(\frac{x}{e^{b}}\right)^{c}}$
pow_4	$c-(ax+b)^{-\alpha}$
MMF	$\alpha-\frac{\alpha-\beta}{1+(\kappa x)^{\delta}}$
exp_4	$c-e^{-ax^{\alpha}+b}$
Janoschek	$\alpha-(\alpha-\beta)e^{-\kappa x^{\delta}}$
Weibull	$\alpha-(\alpha-\beta)e^{-(\kappa x)^{\delta}}$
ilog_2	$c-\frac{a}{\log x}$

本章小结

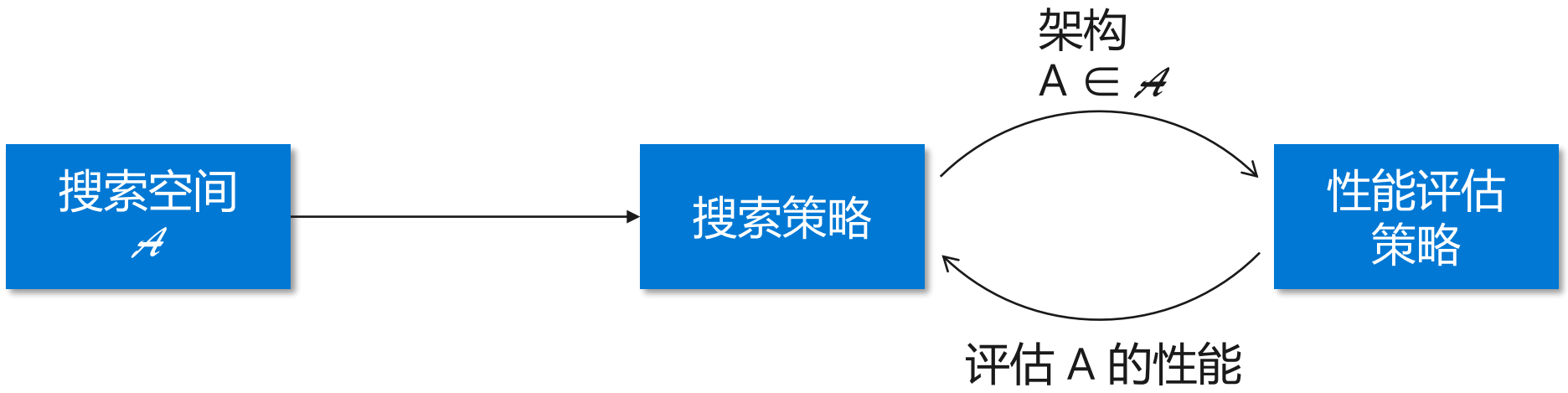
- 回顾
  - 优化、评估算法在自动机器学习过程中的作用
  - 优化算法
    - 随机、退火、进化算法
    - 贝叶斯优化的思想和概念：替代模型、采集函数
  - 评估算法
    - 中位数
    - 曲线拟合
- 思考
  - 自学 Hyperband，BOHB 算法，了解它们的核心，以及是如何平衡挖掘和探索的?
  - 并发运行对贝叶斯优化有什么影响，如何改进?
  - 评估算法对优化算法会产生什么影响，如何改进?
  - 如何实现递减的曲线拟合评估算法，需要注意什么?

自动机器学习其它方向与应用

- 大部分都使用了优化算法
- 应用方向
  - 神经网络架构搜索
  - 模型压缩
  - 特征选择
  - 需要初始化参数的系统优化评估

神经网络架构搜索

- Neural Architecture Search（NAS）- 生成并评估不同神经网络结构来找到更好的网络结构。
- 深度学习上，超参优化应用于现有的神经网络架构参数，一般不修改神经网络结构。

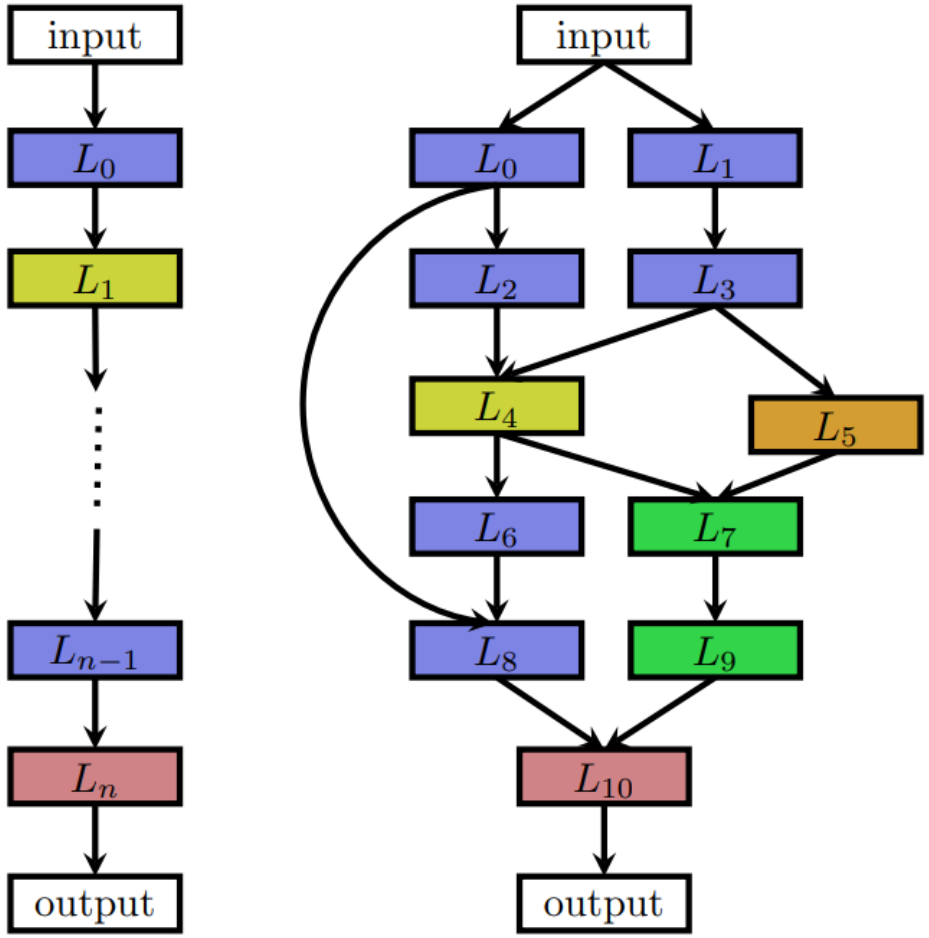


神经网络架构搜索

- 搜索空间
  - 通过先验知识构建
  - 易于搜索
- 搜索策略
  - 平衡挖掘与探索
- 性能评估策略
  - 尽量节约资源，不进行完整的训练和验证过程

神经网络架构搜索 – 搜索空间

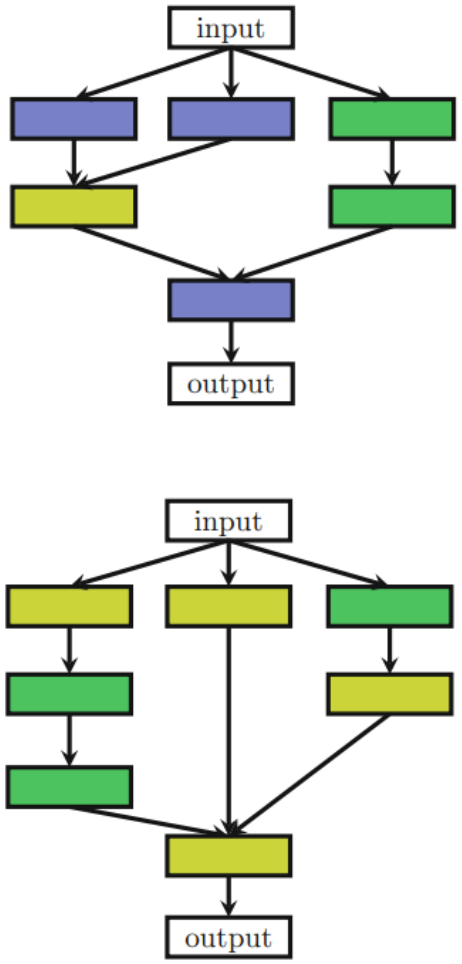
- 搜索空间的设计：在一定原则下可以表示的结构。
- 左侧为单一链式神经网络，层数、层算法及其超参数作为搜索空间。
- 右侧有分支、跳跃连接，可作为搜索空间。

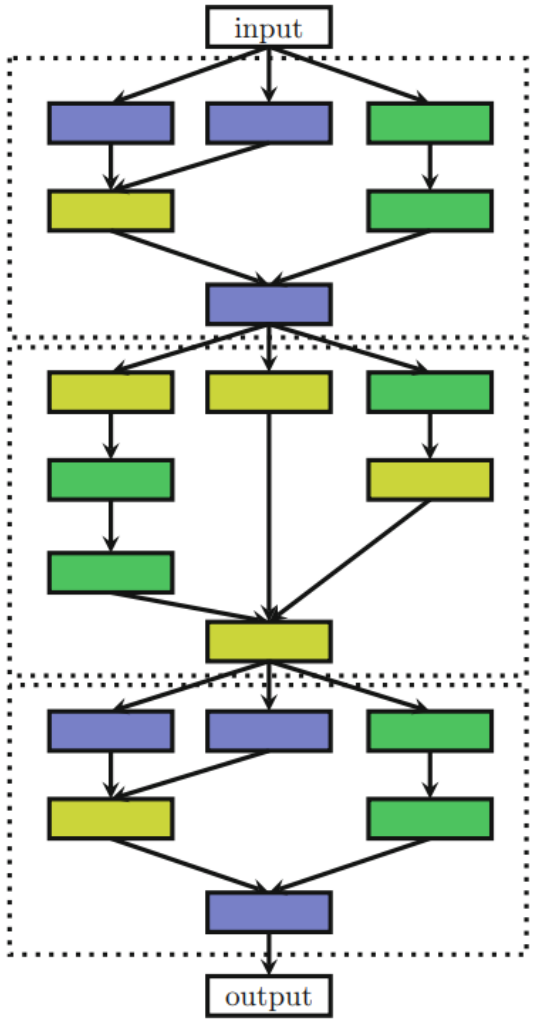


L_i 神经网络层

神经网络架构搜索 – 搜索空间

- 将细胞（cell、block）子网络作为模块进行训练。
  - 普通细胞网络：输入输出维度相同。可组合任意数量个。
  - 降维细胞网络：输出维度变少。
- 可增加普通细胞子网络，而不影响输出的结构。
- 优点
  - 搜索空间大大减小。
  - 细胞子网络可迁移到其它数据集。





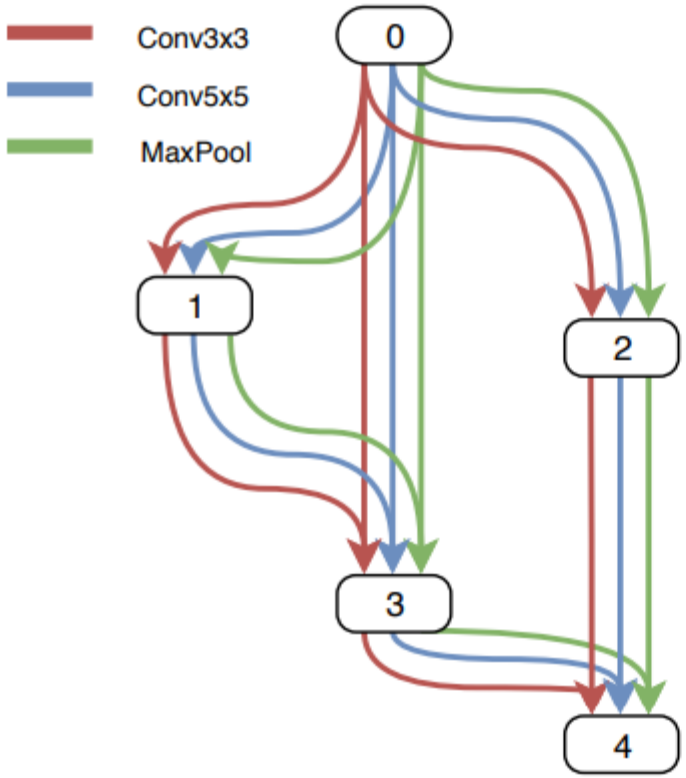
神经网络架构搜索 - 搜索策略

- 超参优化算法：随机、进化、贝叶斯优化等。
- 强化学习算法
  - 将参数生成看作代理的动作。
  - 将已训练架构的评估看作奖励。

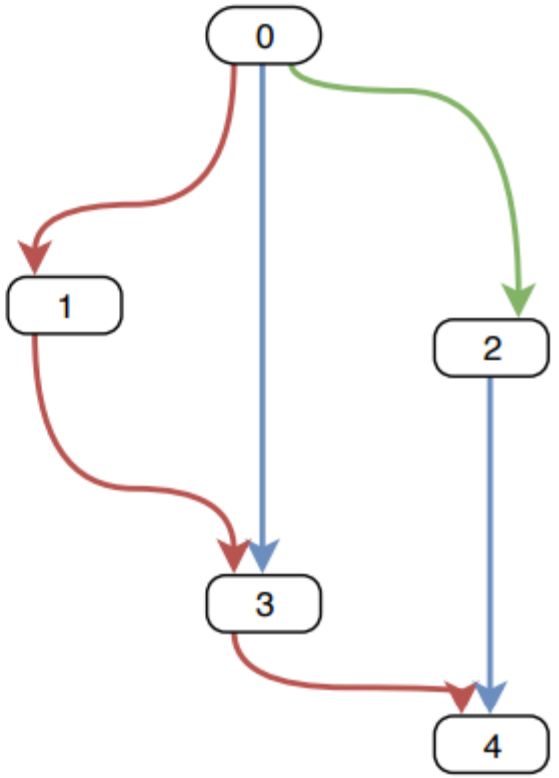
神经网络架构搜索 - 性能评估策略

- 完整训练：结果可靠、计算资源消耗大。
- 通过低精度训练评估，选好神经网络后，再进行完整训练。
  - 使用少量数据
  - 低精度数据（低分辨率图片）
  - 减少迭代次数，通过曲线拟合等方法估计。
- 每次训练后共享权重。 E.g.: ENAS 子网络权重共享
- 一次性（one-shot）架构搜索
  - 共享子网络间的权重，并训练子网络。
  - 仅通过推理来评估。

神经网络架构搜索 – one-shot



超网络



子网络

模型压缩

- 神经网络规模越来越大，最大的已有上百亿参数。
- 神经网络模型中均存在一定的冗余信息。
- 主要压缩方向
  - 剪枝（Pruning）：将冗余神经元甚至通道的权重设置为 0，使其不起作用。
  - 量化（Quantization）：降低浮点数精度。

    原理：减少表示每个量化对象（例如：权重，激活函数，梯度）所需的比特数来压缩原始网络。

    挑战：减少表示精度的同时，不降低模型的准确度

模型压缩 – 剪枝

- 剪枝阶段
  - 一次性剪枝，对训练完成的模型进行一次剪枝
  - 迭代剪枝，在训练过程中逐步剪枝
- 神经元重要性评估方法
  - 权重、输出激活值评估
- 评估值
  - 中位数、绝对值（L1）、平方和（L2）。
- Lottery Ticket Hypothesis：训练一定次数后剪枝，保存掩码。用同样的初始参数再次初始化并训练。从而找到最佳的剪枝方案。

特征选择

- Feature selection，又称降维。通常应用于结构化数据，而非图像、文本、语音等非结构化数据。
- 作用
  - 减小数据获取成本
  - 提高计算效率
  - 节约计算和存储开销
  - 降低过拟合风险

特征选择方法

- 过滤方法（Filter）：通过特征的统计值等，一次性选择出特征。小数据量时，速度快。大数据量时无法使用。
  - 主成分分析（PCA）：无监督，保留差异最大的特征，如方差。
  - 线性判别分析（LDA）：有监督，找出分类效果最好的特征。
- 递归方法：迭代选取子集，并通过训练评估。适合绝大多数场景，但训练过程时间相对较长。
- 嵌入式方法（Embedded）：将特征选择嵌入到模型的构建过程中，结合一次性选择和递归方法，通过算法来选择子集。

系统优化

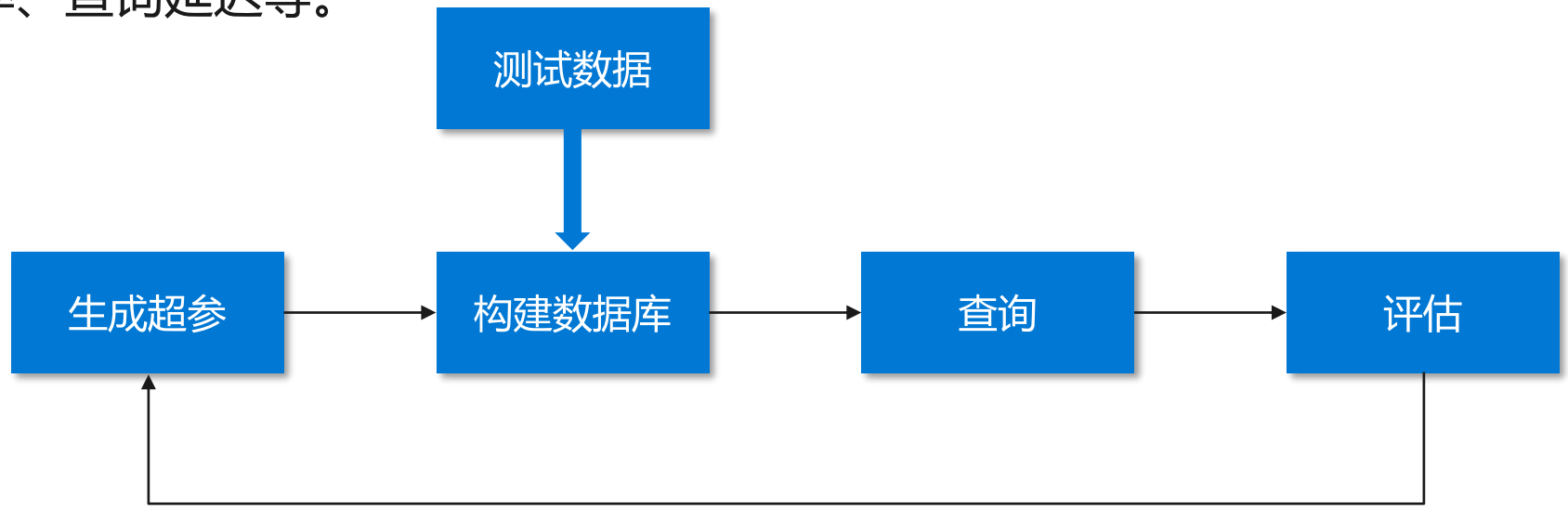
- 复杂软件系统通常有一些配置参数，以便适应不同的应用场景。
- 数据库
  - 不同场景下的配置
  - 数据量级。小规模数据甚至可全部放在内存中，PB级数据需要服务器集群来存取。
  - 数据高可用和灾备的要求。数据备份和同步需要额外时间。
  - 数据间关联。数据关联越多，单次数据操作需要的读写负载越大。
  - 数据读写频率。写入较多的数据需要更多优化写入效率，读取较多的则相反。
  - 数据特点。如：向量数据需要近似数据查询。文本数据可能需要全文索引。

系统优化 - 向量数据库

- 应用
  - 对词语、句子嵌入（Embedding）的计算和查询，是自然语言应用中非常重要的中间数据结构。
- 特点
  - Embedding 值通常为一组浮点数组成的向量。
  - 向量数据库可综合多种算法来平衡读写需求。
    - kd-tree：树结构，写入效率高。
    - k-means：图结构，查询效率和精度高。
- 参数对系统的影响
  - 存储大小、索引构建时间、索引质量、搜索延迟。
- 在数据库构建前，大部分参数需要确定。

系统优化 - 向量数据库

- 超参数
  - 初始化参数，如：近邻图的邻居数量、树的子节点数量。
- 评估指标
  - 召回率、查询延迟等。



本章小结

- 回顾
  - 神经网络架构搜索
    - 搜索空间：层操作、连接操作、子网络
    - 搜索策略：超参搜索类算法、强化学习算法
    - 性能评估策略：完整训练、低精度训练、权重共享、一次性搜索
  - 模型压缩：剪枝、量化
  - 特征选择：过滤、递归、嵌入
  - 系统优化
- 思考
  - 神经网络架构搜索和超参搜索相比，在哪些方面有优势？
  - 在系统优化上，自动机器学习还可以有什么具体应用？

自动机器学习系统

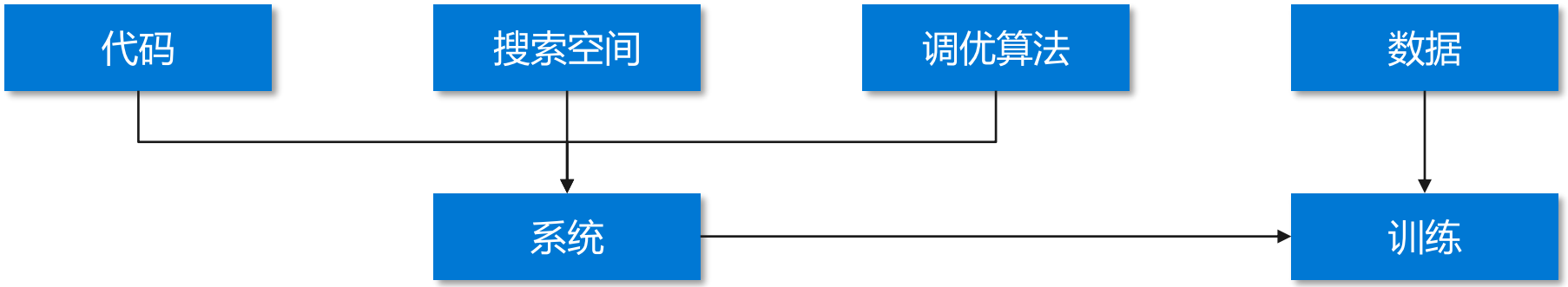
- 端到端
- 辅助优化

自动机器学习系统 – 端到端

- 输入数据，输出模型。
- 使用系统预先配置好搜索空间、网络模型。
- 优点
  - 简单易用，可直接提供推理服务。适合无机器学习经验的任何人使用。
- 缺点
  - 场景有限，如自然语言的应用场景非常广泛，如翻译、搜索、摘要生成等，需要分别开发。
  - 不能结合人工经验，模型性能可能会受到影响。

自动机器学习系统 – 辅助优化

- 输入代码、数据、搜索空间，输出搜索结果或模型。
- 优点
  - 结合人工经验，功能强大，适合机器学习领域人员使用。
  - 几乎适合任何场景。
- 缺点
  - 需要有编程经验和一定的机器学习领域知识。



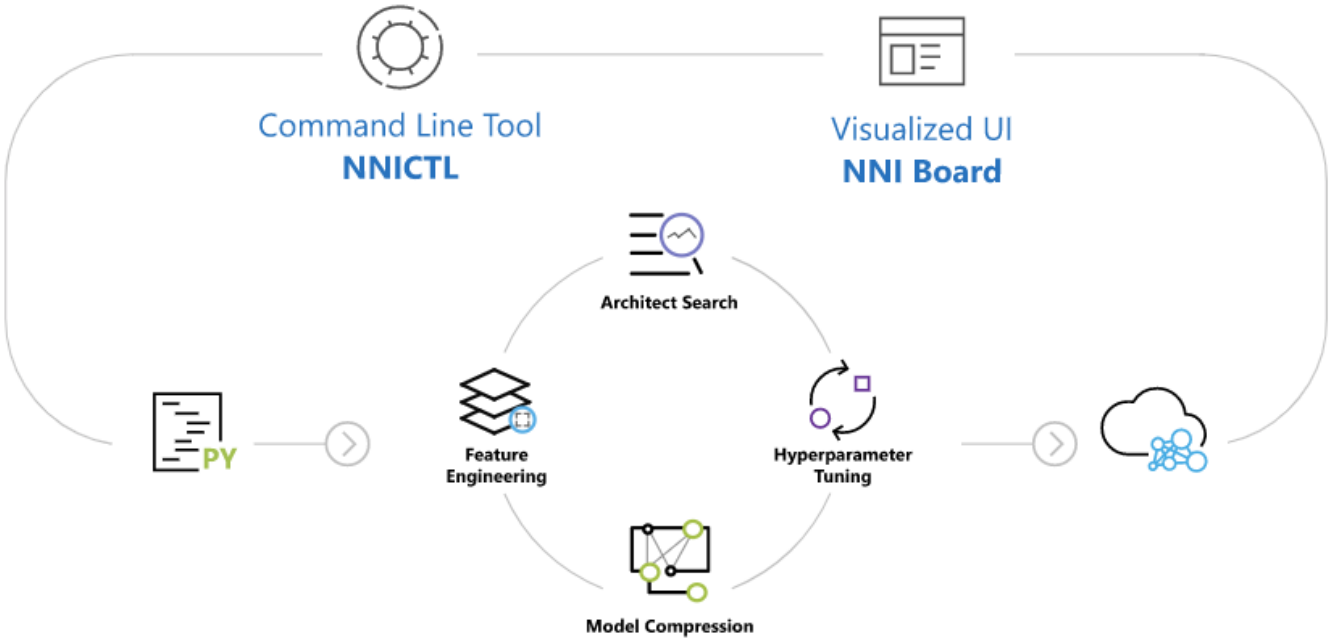
自动机器学习系统 – 功能

- 集成自动机器学习算法
  - 将机器学习算法与自动机器学习算法相分离。
- 管理计算资源
  - 通过多环境来并行评估超参组合。
- 提供分析工具
  - 分析超参和结果的相关性，可进一步优化搜索空间。
- 输出最佳参数配置或最终模型
  - 一些NAS算法只需要输出最佳参数配置，再进行最终训练

自动机器学习系统 – 实例

Neural Network Intelligence

- Neural Network Intelligence (NNI)
  - https://github.com/microsoft/nni
- 易于安装
  - pip3 install nni
- 集成算法、训练平台多
- 提供可视化监测和分析界面
- 开源项目，便于学习



本章小结

- 回顾
  - 功能：集成自动机器学习算法、管理计算资源、提供分析工具、输出结果。
  - 端到端：最大化易用性
  - 辅助优化：覆盖场景广泛
- 思考和实践
  - 使用 NNI 进行一次自动机器学习，体验并思考自动机器学习系统的价值。
  - 了解端到端的自动机器学习系统，它们都用来解决了哪些领域的问题？

自动机器学习的局限

- 资源消耗高，大部分机器学习算法需要大量资源。
- 与有经验的人工优化结果相当。

前沿

- 神经网络架构搜索
  - 更高效搜索方法，节约计算资源。
  - 更通用的算法评估方法，帮助筛选出更合适的算法。
- 自动机器学习系统
  - 更通用的算法集成方案，覆盖模型压缩等新兴的自动机器学习领域。
  - 更高效的性能评估方法，节约计算资源。

本章小结

- 思考
  - 结合自己的机器学习经验，还有哪些方面可能是自动机器学习的前沿？
  - 结合使用 NNI 等自动机器学习系统的经验，还有哪些方面需要完善？

参考资料

- NNI：https://nni.readthedocs.io
- 模拟退火：https://en.wikipedia.org/wiki/Simulated_annealing
- 高斯过程和贝叶斯优化：https://zhuanlan.zhihu.com/p/86386926
- Bayesian optimization: http://krasserm.github.io/2018/03/21/bayesian-optimization/
- Algorithms for Hyper-Parameter Optimization: https://papers.nips.cc/paper/4443-algorithms-for-hyper-parameter-optimization.pdf
- Neural Architecture Search: A Survey: https://arxiv.org/pdf/1808.05377.pdf
- Feature selection: https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/