15. 决策树常见问题

1．谈谈你对决策树的理解?

决策树是基于树的结构进行决策的，学习过程包括特征选择，决策树的生成和剪枝过程。决策树的学习过程通常是递归地选择最优特征，并用最优特征对数据集进行分割。开始时，构建根节点，选择最优特征，该特征有几种值就划分为多少子集，每个子集递归调用此方法，返回结点。返回的结点就是上一层的子节点，直到所有特征都已经用完，或者数据集只有一维特征为止。

2．ID3

ID3算法是决策树的一种，它是基于奥卡姆剃刀原理的，即用尽量用较少的东西做更多的事。ID3算法，即Iterative Dichotomiser 3，迭代二叉树3代，是Ross Quinlan发明的一种决策树算法，这个算法的基础就是上面提到的奥卡姆剃刀原理，越是小型的决策树越优于大的决策树，尽管如此，也不总是生成最小的树型结构，而是一个启发式算法。在信息论中，期望信息越小，那么信息增益就越大，从而纯度就越高。ID3算法的核心思想就是以信息增益来度量属性的选择，选择分裂后信息增益最大的属性进行分裂。该算法采用自顶向下的贪婪搜索遍历可能的决策空间。

设S是训练样本集，它包括n个类别的样本，这些方法用C_i表示，那么熵和信息增益用下面公式表示：
信息熵：

$$E(S) = -\sum_{i=0}^{n} p_i \log_2 p_i$$

其中p_i表示C_i的概率
样本熵：

$$E_A(S) = -\sum_{j=1}^{m} \frac{|S_j|}{|S|} E(S_j)$$

其中S_i表示根据属性A划分的S的第i个子集，S和S_i表示样本数目
信息增益：

$$Gain(S, A) = E(S) - E_A(S)$$

3．ID3的不足

- ID3没有考虑连续特征，比如长度，密度都是连续值，无法在ID3运用。这大大限制了ID3的用途。
- ID3采用信息增益大的特征优先建立决策树的节点。很快就被人发现，在相同条件下，取值比较多的特征比取值少的特征信息增益大。比如一个变量有2个值，各为1/2，另一个变量为3个值，各为1/3，其实他们都是完全不确定的变量，但是取3个值的比取2个值的信息增益大。

计算总熵 $$info(D) = -\sum_{i=1}^{m} p_i log_2(p_i)$$

例如：一个身份证号就确定一个人log2pi等于0，infoA(D)=0信息增益很大，所以偏向属性值多的属性

属性条件熵 $$info_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} info(D_j)$$ 当特征属性值很多时

属性信息增益 $gain(A) = info(D) - info_A(D)$

- ID3算法对于缺失值的情况没有做考虑

- 没有剪枝策略，容易过拟合

4．C4.5

C4.5是机器学习算法中的另一个分类决策树算法，它是基于ID3算法进行改进后的一种重要算法，相比于ID3算法，改进有如下几个要点：

- 用信息增益率来选择属性。
- 引入悲观剪枝策略进行剪枝
- 对连续数据的处理。
- 对缺失数据的处理。

5．C4.5的信息增益率

前三个步骤其实就是ID3算法的计算过程，如果不会的话请在我的有关ID3决策树中学习(下面是三个步骤的公式):

$$info(D) = -\sum_{i=1}^{m} p_i log_2(p_i)$$

$$info_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} info(D_j)$$

$$gain(A) = info(D) - info_A(D)$$

之后的两个步骤其实在前三个步骤基础上的一个进一步的计算

$$H(A) = -\sum_{i=1}^{m} p_i log_2(p_i)$$

$$IGR(A) = gain(A)/H(A)$$

6．C4.5的剪枝策略

在决策树的创建时，由于数据中的噪声和离群点，许多分枝反映的是训练数据中的异常。剪枝方法是用来处理这种过分拟合数据的问题。通常剪枝方法都是使用统计度量，剪去最不可靠的分枝。剪枝一般分两种方法：预剪枝和后剪枝。

a．先剪枝方法中通过提前停止树的构造（比如决定在某个节点不再分裂或划分训练元组的子集）而对树剪枝。一旦停止，这个节点就变成树叶，该树叶可能取它持有的子集最频繁的类作为自己的类。先剪枝有很多方法，比如（1）当决策树达到一定的高度就停止决策树的生长；（2）到达此节点的实例具有相同的特征向量，而不必一定属于同一类，也可以停止生长（3）到达此节点的实例个数小于某个阈值的时候也可以停止树的生长，不足之处是不能处理那些数据量比较小的特殊情况（4）计算每次扩展对系统性能的增益，如果小于某个阈值就可以让它停止生长。先剪枝有个缺点就是视野效果问题，也就是说在相同的标准下，也许当前扩展不能满足要求，但更进一步扩展又能满足要求。这样会过早停止决策树的生长（欠拟合）。

b．另一种更常用的方法是后剪枝，它由完全成长的树剪去子树而形成。通过删除节点的分枝并用树叶来替换它。树叶一般用子树中最频繁的类别来标记。后剪枝一般有

两种方法：第一种方法，也是最简单的方法，称之为基于误判的剪枝。这个思路很直接，完全的决策树不是过度拟合么，我再搞一个测试数据集来纠正它。对于完全决策树中的每一个非叶子节点的子树，我们尝试着把它替换成一个叶子节点，该叶子节点的类别我们用子树所覆盖训练样本中存在最多的那个类来代替，这样就产生了一个简化决策树，然后比较这两个决策树在测试数据集中的表现，如果简化决策树在测试数据集中的错误比较少，并且该子树里面没有包含另外一个具有类似特性的子树（所谓类似的特性，指的就是把子树替换成叶子节点后，其测试数据集误判率降低的特性），那么该子树就可以替换成叶子节点。该算法以自底向上的方式遍历所有的子树，直至没有任何子树可以替换使得测试数据集的表现得以改进时，算法就可以终止。

c. 第一种方法很直接，但是需要一个额外的测试数据集，能不能不要这个额外的数据集呢？为了解决这个问题，于是就提出了悲观剪枝。悲观剪枝就是递归得估算每个内部节点所覆盖样本节点的误判率。剪枝后该内部节点会变成一个叶子节点，该叶子节点的类别为原内部节点的最优叶子节点所决定。然后比较剪枝前后该节点的错误率来决定是否进行剪枝。该方法和前面提到的第一种方法思路是一致的，不同之处在于如何估计剪枝前分类树内部节点的错误率。把一颗子树（具有多个叶子节点）的分类用一个叶子节点来替代的话，在训练集上的误判率肯定是上升的，但是在新数据上不一定。于是我们需要把子树的误判计算加上一个经验性的惩罚因子。对于一颗叶子节点，它覆盖了N_i个样本，其中有E个错误，那么该叶子节点的错误率为（E+0.5）/N_i。这个0.5（详细请参考连续性校正）就是惩罚因子，那么一颗子树，它有L个叶子节点，那么该子树的误判率估计为$(\sum E_i + 0.5 * L)/\sum N_i$。这样的话，我们可以看到一颗子树虽然具有多个子节点，但由于加上了惩罚因子，所以子树的误判率计算未必占到便宜。剪枝后内部节点变成了叶子节点，其误判个数J也需要加上一个惩罚因子，变成J+0.5。那么子树是否可以被剪枝就取决于剪枝后的错误J+0.5在$\sum E_i + 0.5 * L$的标准误差内。对于样本的误差率e，我们可以根据经验把它估计成各种各样的分布模型，比如是二项式分布，或者正态分布。

那么一棵树对于一个数据来说，错误分类一个样本值为1，正确分类一个样本值为0，该树错误分类的概率（误判率）为e_1（可以通过$(\sum E_i + 0.5 * L)/\sum N_i$统计出来），那么树的误判次数就是二项分布，我们可以估计出该树的误判次数均值和标准差：

$$E(subtree\_err\_count) = N * e_1$$

$$sd(subtree\_err\_count) = \sqrt{N * e_1 * (1 - e_1)}$$

其中，

$$e_1 = (\sum E + 0.5 * L)/N$$

$$N = \sum N_i$$

把子树替换成叶子节点后，该叶子的误判次数也是一个伯努利分布，其中N是到达该叶节点的数据个数，其概率误判率e_2为(J+0.5)/N，因此叶子节点的误判次数均值为

$$e_2 = (J + 0.5)/N$$

$$E(leaf\_err\_count) = N * e_2$$

使用训练数据，子树总是比替换为一个叶节点后产生的误差小，但是使用校正后有误差计算方法却并非如此，当子树的误判个数大过对应叶节点的误判个数一个标准差之后，就决定剪枝：

$$E(leaf\_err\_count) < E(subtree\_err\_count) + sd(subtree\_err\_count)$$

这个条件就是剪枝的标准。

通俗点讲，就是看剪枝后的错误率会不会变得很大（比剪枝前的错误率加上其标准差还大），如果剪枝后的错误率变得很高，则不剪枝，否则就剪枝。下面通过一个具体的实例来看一下到底是如何剪枝的。

7．C4.5对连续数据的处理

将连续型的属性变量进行离散化处理，形成决策树的训练集，分三步：

- 把需要处理的样本（对应根节点）或样本子集（对应子树）按照连续变量的大小**从小到大**进行排序
- 假设该属性对应的不同的属性值一共有N个，那么总共有**N-1**个可能的候选分割阈值点，每个候选的分割阈值点的值为上述排序后的属性值中两两前后连续元素的中点
- 用**信息增益率**选择最佳划分

8．C4.5对缺失值的处理

缺失值：在某些情况下，可供使用的数据可能缺少某些属性的值 。例如 (X, y) 是 样本集 S 中的一个训练实例 ， X=(F1_v,F2_v, … Fn_v) 。但 是其属 性 Fi 的值 Fi_v 未 知 。

处理策略：

- 处理缺少属性值的一种策略是赋给它结点t所对应的训练实例中该属性的最常见值
- 另外一种更复杂的策略是为 Fi 的 每个可能值赋予一个概率。例如，给定一个布尔属 性 Fi ，如果结 点 t 包 含 6 个已 知 Fi_v =1 和 4 个 Fi_v =0 的实例，那 么 Fi_v =1 的概率是 0.6 ， 而 Fi_v =0 的概率是 0.4 。于 是，实例 x 的 60% 被分配 到 Fi_v =1 的分支，40% 被分配到另一个分支。这些片断样例 （ fractional examples ）的目的是计算信息增益，另外，如果有第二个缺少值的属性必须被测试，这些样例可以在后继的树分支中被进一步细分 。（C4.5 中使 用 ）
- 简单处理策略就是丢弃这些样本

9．C4.5的缺点

- C4.5生成的是多叉树，一个父节点可以有多个子节点。计算的时候，运算效率没有二叉树高；
- C4.5使用了熵模型，里面有大量的对数运算。
- 如果有连续值的属性，还涉及到排序运算，运算量很大。
- C4.5只能用于分类任务，无法解决回归问题

10．CART

分类回归树(CART,Classification And Regression Tree)算法是一种决策树分类方法。它采用一种二分递归分割的技术，分割方法采用基于最小距离的基尼指数估计函数，将当前的样本集分为两个子样本集，使得生成的的每个非叶子节点都有两个分支。因此，CART算法生成的决策树是结构简洁的二叉树。其核心思想与ID3和C4.5相同，主要的不同处在于CART在每一个节点上都采用二分法，即每个节点都只能有两个子节点，最后构成的是二叉树。CART的生成就是递归的构建二叉树的过程，**对回归树用平方误差最小化准则进行特征选择**，生成二叉树；**对于分类树用基尼指数最小化准则**，进行特征选择，生成二叉树。

在分类树中，用基尼指数来选择最优特征。

输入：训练数据集D、特征集A、停止计算的条件
输出：$CART$决策树
1、计算现有特征对该数据集D的基尼指数。对每一个特征A，对其可能取值的每个值a，根据样本点对$A=a$的测试为"是"或"否"将D分割成D_1和D_2两部分，计算$A=a$时的基尼指数 2、在所有可能的特征A以及它们所有可能的切分点a中，选择基尼指数最小的特征及其对应的切分点作为最优特征与最优切分点。依最优特征和最优切分点，从现节点生成两个子节点，将训练数据集依特征分配到两个子节点中去 3、对两个子节点递归调用步骤1、2，直到满足停止条件 4、生成$CART$决策树

算法停止的条件可以是：结点中样本个数小于预定阈值、样本集的基尼指数小于预定阈值(样本基本属于同一类)、没有更多的特征。

在回归树中，通常用平方误差来表示回归树对于训练数据的预测误差。回归树生成算法如下：

输入：训练数据集D
输出：回归树$f(x)$
1、选择最优切分变量j和切分点s，求解 $$\min_{j,s}[\min_{c_1}\sum_{x_i\in R_1(j,s)}(y_i-c_1)^2+\min_{c_2}\sum_{x_i\in R_2(j,s)}(y_i-c_2)^2]$$ 遍历变量j，对固定的切分变量j扫描切分点s，选择使上式达到最小值的对(j,s) 2、用选定的对(j,s)划分区域并决定相应的输出值： $$R_1(j,s)=\{x\|x^{(j)}\le s\},R_2(j,s)=\{x\|x^{(j)}>s\}$$ $$\hat{c_m}=\frac{1}{N_m}\sum_{x_i\in R_m(j,s)}y_i,x\in R_m,m=1,2$$ 3、继续对两个子区域调用步骤1、2，直到满足停止条件 4、将输入空间划分为M个区域$R_1,R_2,\ldots,R_M$，生成决策树： $$f(x)=\sum_{m=1}^{M}\hat{c_m}I(x\in R_m)$$

11．谈谈你对信息增益和信息增益比的理解?

熵**H(X)**度量了X的不确定性，**条件熵H(X|Y)**度量了我们在知道Y以后X剩下的不确定性，那么**H(X)-H(X|Y)**度量了X在知道Y以后不确定性减少程度，在信息论中称为互信息，记为**I(X,Y)**，在决策树ID3算法中叫做信息增益。ID3算法就是用信息增益来判断当前节点应该用什么特征来构建决策树。信息增益大，则越适合用来分类。

什么是信息增益比？信息增益比计算公式： $I_R(D,A)=\frac{I(A,D)}{H_A(D)}$，这里 $H_A(D)$ 是特征熵，特征越多对应的特征熵越大，它作为分母，可以校正信息增益容易偏向于取值较多的特征的问题。

$$H_A(D)=-\sum_{i=1}^{n}\frac{|D_i|}{|D|}log_2\frac{|D_i|}{|D|}$$

12．CART对C4.5的改进

- 将多叉树改为二叉树
- 既可用于分类任务，有可用于回归任务
- 采用GINI系数代替熵，减少log计算
- 采用其他方式对缺失值进行处理
- 采用其他方式处理连续值
- 采用代价复杂度剪枝策略进行剪枝
- 处理类别不平衡的数据

13．GINI系数

基尼指数是CART决策树的特征选择准则，有关CART的内容在下文进行展开说明，本部分只对基尼指数进行说明。

基尼指数：在分类问题中，假设有K个类，样本点属于第k类的概率为p_k，则概率分布的基尼指数定义为：

$$Gini(p)=\sum_{k=1}^{K}p_k(1-p_k)=1-\sum_{k=1}^{K}p_k^2$$

对于给定的样本集D，基尼指数为：

$$Gini(D)=1-\sum_{k=1}^{K}(\frac{|C_k|}{|D|})^2$$

其中C_k是D中属于第k类的样本子集，K是类的个数。

基尼指数$Gini(D)$反映了从数据集D中随机抽取两个样本，其类别标记不一致的概率。基尼指数值越大，样本集合的不确定性就越大。

特征A的基尼指数定义为：

$$Gini(D,A)=\sum_{i=1}^{n}\frac{|D_i|}{|D|}Gini(D_i)$$

$Gini(D,A)$表示经$A=a$分割后集合D的不确定程度。在选择特征的时候，选择使得划分后基尼指数最小的特征作为最优划分特征。

14．CART的剪枝方法

当决策树的结构过于复杂时，容易出现过拟合现象。在决策树中，通常通过剪枝(pruning)对树进行简化，以此来防止过拟合。决策树的剪枝往往通过极小化决策树整体的损失函数或代价函数来实现。设树T的叶子节点个数为|T|，t是树T的叶子节点，该叶子节点有N_t个样本点，其中k类的样本点有N_{tk}个，k = 1,2,3,...K，$H_t(T)$为叶子节点t上的经验熵，$\alpha \geq 0$为参数，则决策树的损失函数定义为：

$$C_\alpha(T) = \sum_{t=1}^{|T|} N_t H_t(T) + \alpha|T| \tag{4.1}$$

其中叶子节点t的经验熵$H_t(T)$为：

$$H_t(T) = -\sum_k \frac{N_{tk}}{N_t} log \frac{N_{tk}}{N_t} \tag{4.2}$$

将(4.2)式带入(4.1)式得：

$$C_\alpha(T) = -\sum_{t=1}^{|T|} \sum_{k=1}^{K} N_{tk} log \frac{N_{tk}}{N_t} + \alpha|T| \tag{4.3}$$

其中第一项表示模型对训练数据的训练误差，第二项可看做是正则项，|T|表示树的复杂度，当α较大时选择较简单的树模型；当α较小时，选择较复杂的树模型，当$\alpha = 0$意味着只考虑模型与训练数据的拟合程度，不考虑模型的复杂度。

剪枝，就是当α确定时，选择损失函数最小的模型，即损失函数最小的子树。

15．CART的缺失值处理方法

模型对于缺失值的处理会分为两个子问题：

1. 在特征值缺失的情况下进行划分特征的选择？

对于问题 1，CART 一开始严格要求分裂特征评估时只能使用在该特征上没有缺失值的那部分数据，在后续版本中，CART 算法使用了一种惩罚机制来抑制提升值，从而反映出缺失值的影响（例如，如果一个特征在节点的 20% 的记录是缺失的，那么这个特征就会减少 20% 或者其他数值）。

2. 选定该划分特征，对于缺失该特征值的样本如何处理？

对于问题 2，CART 算法的机制是为树的每个节点都找到代理分裂器，无论在训练数据上得到的树是否有缺失值都会这样做。在代理分裂器中，特征的分值必须超过默认规则的性能才有资格作为代理（即代理就是代替缺失值特征作为划分特征的特征），当 CART 树中遇到缺失值时，这个实例划分到左边还是右边是决定于其排名最高的代理，如果这个代理的值也缺失了，那么就使用排名第二的代理，以此类推，如果所有代理值都缺失，那么默认规则就是把样本划分到较大的那个子节点。代理分裂器可以确保无缺失训练数据上得到的树可以用来处理包含确实值的新数据。

16．CART的连续值处理方法（回归树）

1. 计算各个特征各个值划分的两部分 D_1 和 D_2 的误差平方和，选择误差平方和最小的（如下式）作为最优特征和最优切分点

$$\min_{A,a} \left[\min_{c_1} \sum_{x_i \epsilon D_1(A,a)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \epsilon D_2(A,a)} (y_i - c_2)^2 \right]$$

其中， c_1 为 D_1 的样本输出均值， c_2 为 D_2 的样本输出均值。

17．CART的类别不平衡数据的处理方法

CART 的一大优势在于：无论训练数据集有多失衡，它都可以将其子冻消除不需要建模人员采取其他操作。

CART 使用了一种先验机制，其作用相当于对类别进行加权。这种先验机制嵌入于 CART 算法判断分裂优劣的运算里，**在 CART 默认的分类模式中，总是要计算每个节点关于根节点的类别频率的比值，这就相当于对数据自动重加权，对类别进行均衡。**

对于一个二分类问题，节点 node 被分成类别 1 当且仅当：比如二分类，根节点属于 1 类和 0 类的分别有 20 和 80 个。在子节点上有 30 个样本，其中属于 1 类和 0 类的分别是 10 和 20 个。如果 10/20>20/80，该节点就属于 1 类。

通过这种计算方式就无需管理数据真实的类别分布。假设有 K 个目标类别，就可以确保根节点中每个类别的概率都是 1/K。这种默认的模式被称为“先验相等”。先验设置和加权不同之处在于先验不影响每个节点中的各类别样本的数量或者份额。先验影响的是每个节点的类别赋值和树生长过程中分裂的选择。

18．C4.5 算法构造的决策树只能进行分类任务，而CART树既可以做分类，也可以做回归。

CART树的本质上是对特征空间进行二元划分，所以CART算法生成的树是一棵二叉树，且可以对类别型变量和数值型变量进行分裂。对类别变量进行分类时，分为等于该属性和不等于该属性；在对连续型变量进行划分时，分为大于和小于。**所以 CART 树做分类的时候用 GINI 系数作为划分标准，在做回归的时候用的是均方误差。**

19．为什么CART树要用基尼系数?

基尼系数的优点：在保证准确率的情况下大大减小了计算量。

20．决策树出现过拟合的原因以及解决办法?

决策树在最极端的情况下，会根据每一个样本生成单独的叶子节点，这就导致了很严重的过拟合，因此需要剪枝。决策树主要有两种剪枝方法：预剪枝和后剪枝。

21．预剪枝与后剪枝的优缺点?

- 后剪枝考虑了所有的可能性，因此有较高的计算开销，但模型效果最好；
- 预剪枝一定成都上降低了过拟合风险，但由于基于贪心策略，不能保证全局最优。

22．决策树与逻辑回归的区别?

- 逻辑回归通常用于分类问题，决策树可回归、可分类。
- 逻辑回归是线性函数，决策树是非线性函数（没有方程式可以表达自变量和因变量之间的关系）。
- 逻辑回归的核心是sigmoid函数，具有无限可导的优点，常作为神经网络的激活函数。
- 逻辑回归较为简单，不容易产生过拟合

23．决策树有哪些参数?

- splitter：特征选择标准：基尼系数or信息增益比
- max_depth：决策树最大深度
- min_impurity_decrease：节点划分最小不纯度（低于这个值，则决策树不再进行生长）
- min_samples_split：内部节点再划分所需最小样本数

- min_samples_leaf：叶子节点最少样本数
- max_leaf_nodes：最大叶子节点数
- class_weight：类别权重，主要是为了防止训练集某些类别的样本过多，导致训练的决策树过于偏向这些类别。

24．分类树与回归树?

分类树以 ID3 为例，在对一个特征进行划分时，首先是穷举这个特征的每一个阈值，找到使得 特征<=阈值 和 特征>阈值 的两个分支的熵的最大值，按照该标准分支得到两个新的节点。用同样的方法继续分支，直到得到种类唯一的叶子节点，或者达到预设的终止条件为止。

回归树的流程类似于分类树，不同的是回归树采用的是均方误差作为划分标准。回归树会尽可能的寻找最小化均方误差，即预测值和实际值的误差最小，且越小则越有可能作为新的节点。

25．决策树的优点及缺点

优点：

- 简单直观，生成的决策树很直观。
- 基本不需要预处理，不需要提前归一化，处理缺失值。
- 使用决策树预测的代价是O(logm)，m为样本数。
- 既可以处理离散值也可以处理连续值。很多算法只是专注于离散值或者连续值。
- 可以处理多维度输出的分类问题。
- 相比于神经网络之类的黑盒分类模型，决策树在逻辑上可以得到很好的解释
- 可以交叉验证的剪枝来选择模型，从而提高泛化能力。

缺点：

- 决策树算法非常容易过拟合，导致泛化能力不强。可以通过设置节点最少样本数量和限制决策树深度来改进。
- 决策树会因为样本发生一点点的改动，就会导致树结构的剧烈改变。这个可以通过集成学习之类的方法解决。
- 寻找最优的决策树是一个NP难的问题，我们一般是通过启发式方法，容易陷入局部最优。可以通过集成学习之类的方法来改善。
- 有些比较复杂的关系，决策树很难学习，比如异或。这个就没有办法了，一般这种关系可以换神经网络分类方法来解决。
- 如果某些特征的样本比例过大，生成决策树容易偏向于这些特征。这个可以通过调节样本权重来改善。
- 无论是ID3, C4.5还是CART,在做特征选择的时候都是选择最优的一个特征来做分类决策，但是大多数，分类决策不应该是由某一个特征决定的，而是应该由一组特征决定的。这样决策得到的决策树更加准确。

26．决策树模型为什么不需要进行数据归一化?

- 决策树是一种概率模型，它不关心变量的值，只关心变量的分布和变量之间的条件概

率。所以对数值进行归一化，并不会影响分裂节点的位置。

- 因为数值缩放不影响分裂点位置，对树模型的结构不造成影响，而且是不能进行梯度下降的，因为构建树模型（回归树）寻找最优点时是通过寻找最优分裂点完成的，因此树模型是阶跃的，阶跃点是不可导的，并且求导没意义，也就不需要归一化

- [ ] https://www.jianshu.com/p/71d8bf9a334c <https://www.jianshu.com/p/71d8bf9a334c>
- [ ] https://blog.csdn.net/weixin_38753213/article/details/114985901 <https://blog.csdn.net/weixin_38753213/article/details/114985901>