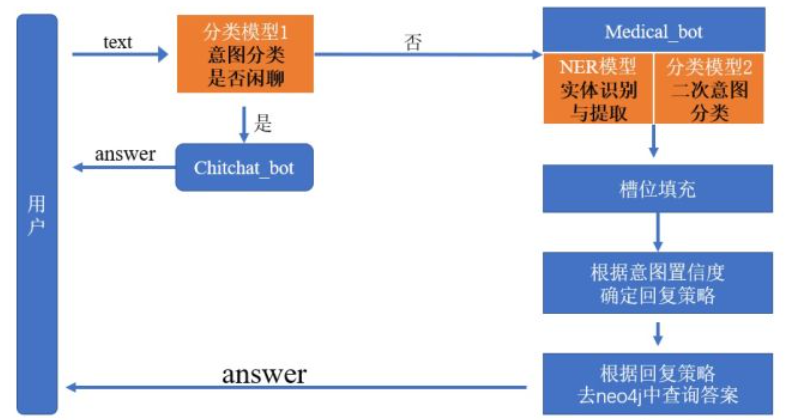
**关于医疗问答机器人回答策略及设计方案草案**

**姓名：文楚童**

**日期：2022年2月16日**

# 决策框架

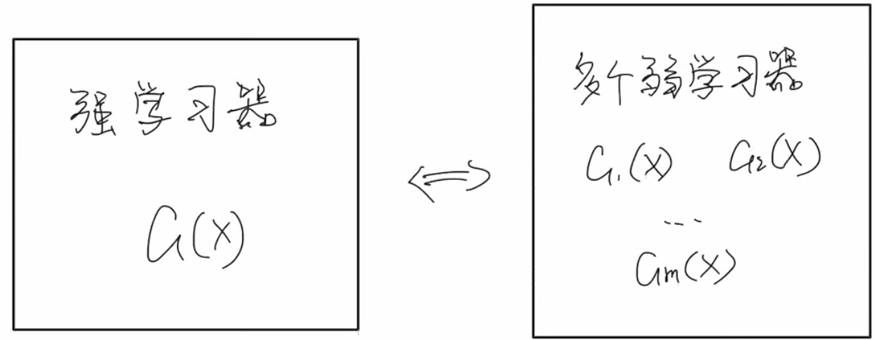


网上较为靠谱的一个框架

# 分类模型1

LR+GBDT：建议先看后面附加的链接里面的文章，再看下面的推导内容

## GBDT(Gradient Boosting Decision Tree)



GBDT是集成学习的一种，是由多个弱学习器组成，每个个体学习器是一个回归树。个体学习器通过改变训练数据的权值(概率分布)来训练个体学习器。

且GBDT是一种boosting的方法，其他boosting方法还有Adaboost,XGBoost,boosting的特点是个体学习器间存在强依赖关系，必须串行生成的序列化方法。

其**优化机制**为：

**提高**那些在前一轮被弱分类器**分错**的样本的权值

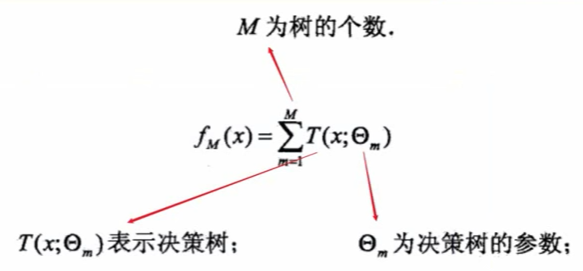
**减少**那些在前一轮被弱分类器**分对**的样本的权值

即让分类错误的样本在后续收到更多的关注

最后对于训练完的训练器做线性组合得到最后结果

注：和bagging的随机森林方法有所不同，不同点其中之一在于随机森林在为每个弱分类器抽取样本时对样本会随机抽取，同时每个样本的特征通道也会做随机抽取。

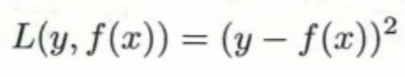
### GBDT梯度提升树原理



提升树本质上就是弱分类器为决策树(回归/分类树)的加法模型，且加法模型的参数均为1。

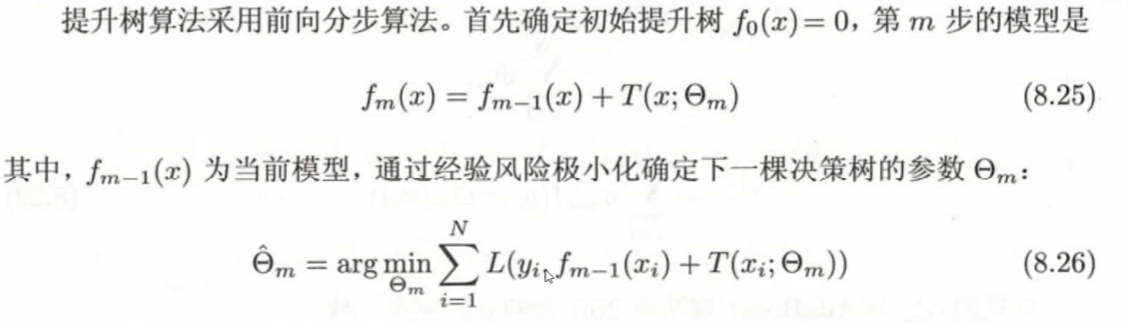
其损失函数按具体问题选择不同的函数。

1. 回归问题：MSE(均方误差损失函数)



1. 分类问题：
   1. 二分类问题：指数损失函数
   2. 多分类问题：softmax
2. 一般决策问题：自定义损失函数

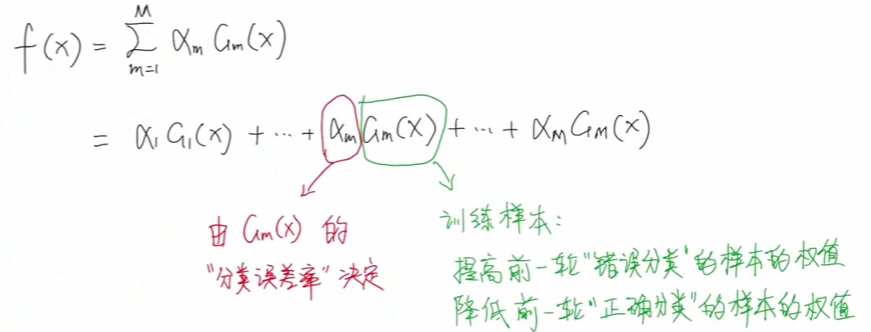
优化方法即更新方法用前向分步方法



**一．二分类问题**

弱学习器用分类树

损失函数用指数损失函数



上图为Adaboost的加法模型

提升树解决二分类模型时相当于Adaboost的特殊情况：

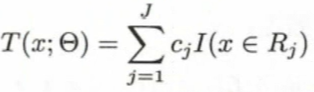
1. 基分类器G(x)限制为二类分类树(输出只有正负1，设置阈值)
2. 基分类器权重αm全部时1

### 

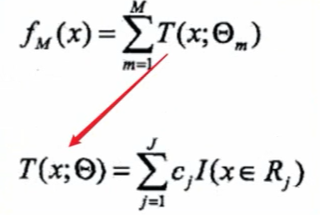
通过指数损失函数来调整样本数据的权值，从而让每个弱分类器学到不一样的内容

**二．回归问题**

弱学习器用回归树

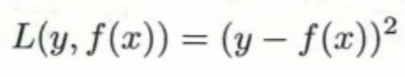


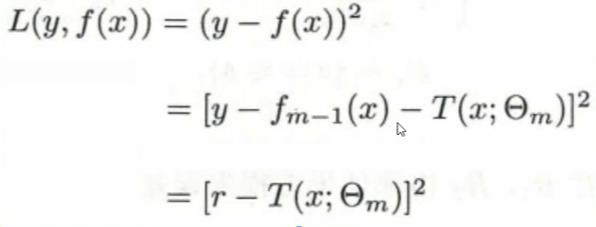
单个回归树



带入到加法模型

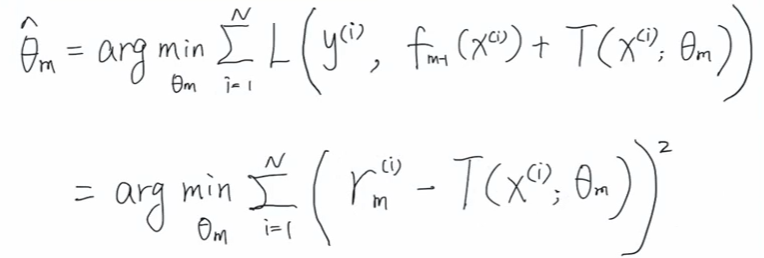
损失函数用平方误差损失





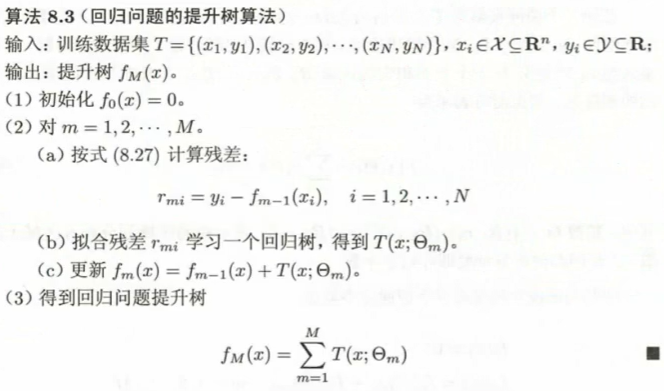
将加法模型带入损失函数的表达式

其中r是残差即上一步的预测值与真实值的差距(这边不应该说成损失值，损失值是指每一轮的最新的预测值和恒定不点的真实值之间的差距)  
 所以加法模型的任务就是拟合这个残差，训练第m个若训练器时，利用残差数据构建训练样本，让新的弱学习器拟合该训练样本



其中θm是决策树的参数

优化算法用前向分步算法



1. **一般决策问题(梯度提升树GBDT)**

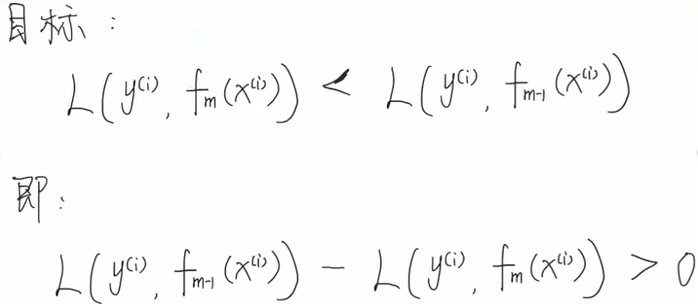
不同的损失函数，对应的凸优化问题不一样，希望找到一个通用的方式，求解一般性凸优化问题

弱学习器用回归树

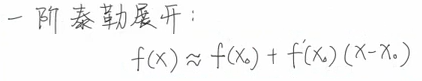
损失函数为自定义函数

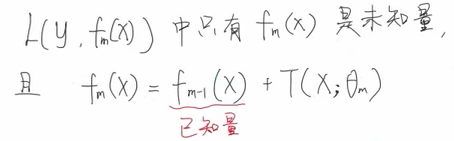
核心思想：

要确保每增加一个弱学习器，都要使得总体损失越来越小，即第m步要比第m-1步的损失要小



具体处理方式为对损失函数进行一阶泰勒展开(比较可行的一种方法)



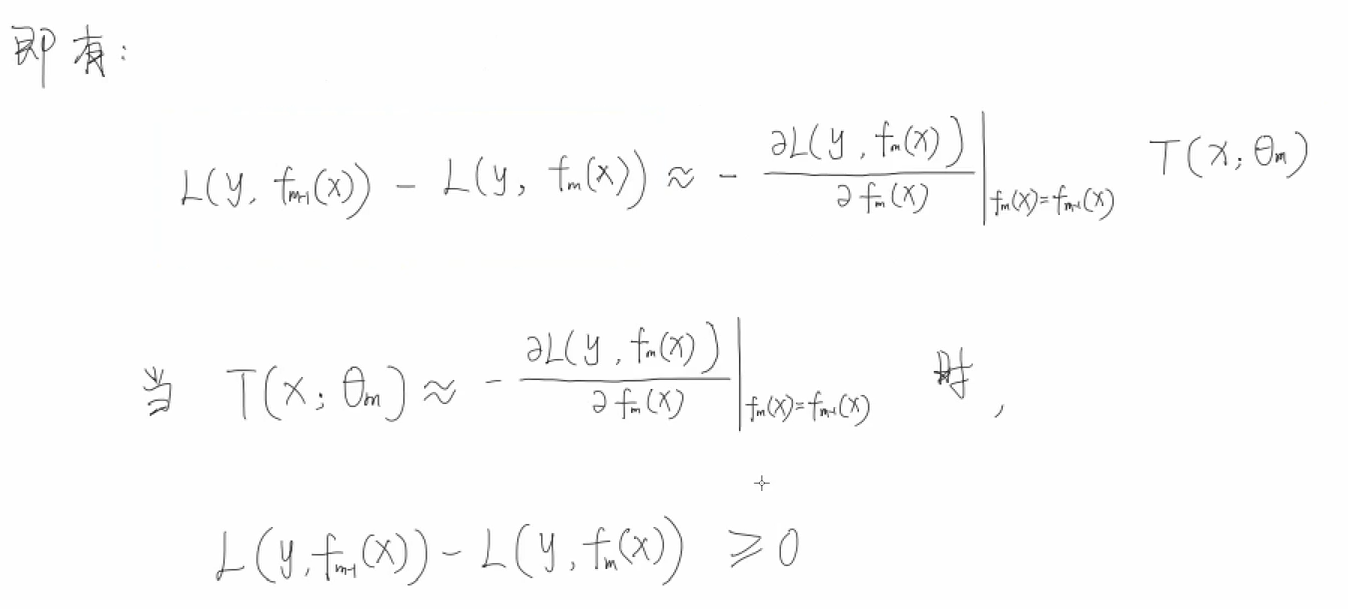




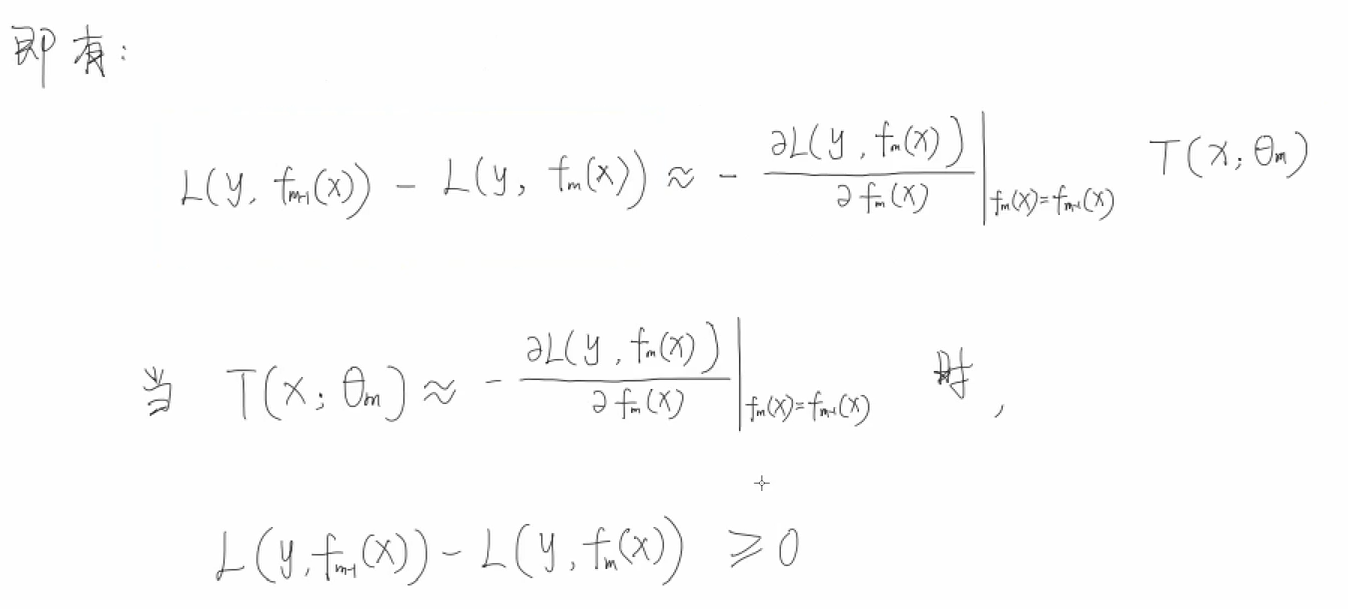
注：这一步和梯度下降算法的公式和思路是一模一样的

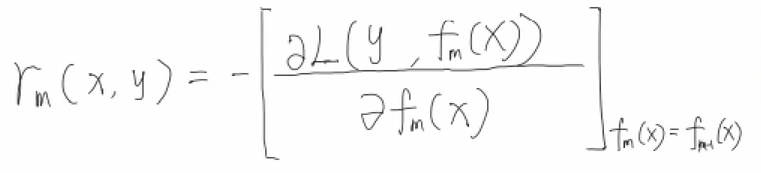
即中间求导部分为损失函数的负梯度

移项后



应该让左侧恒大于零，则右侧两个应该是互为相反数，即



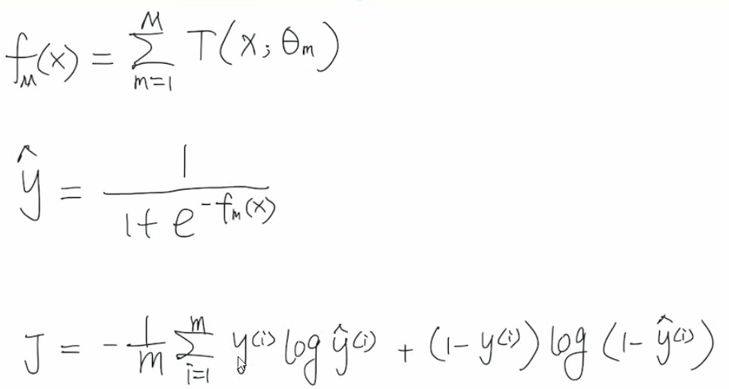


其中rm表示第m步的训练样本，是用损失函数的负梯度构造出来的

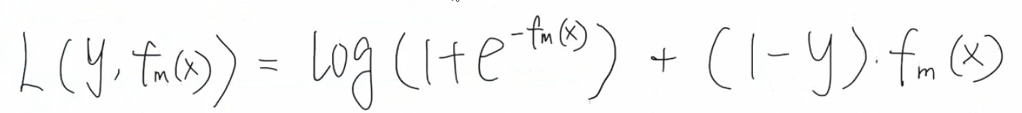
具体步骤为：

1. 对上一步的损失函数对于fm(x)求负导
2. 将fm-1(x)带入
3. 将原来的样本(xi,yi)带入函数rm(x,y)得到rmi
4. 最后得到第m轮的训练数据集Tm
5. **二分类问题(梯度提升树GBDT)**

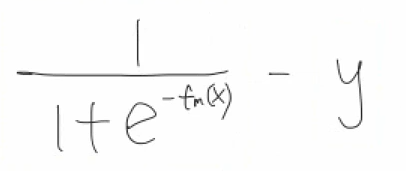
在上面解决二分类问题时我们用的是分类树，而梯度提升树GBDT的基分类器是回归树，因此不能直接作用，其中的一种解决办法是将得到的回归树的数值转化到0~1的范围，如下图



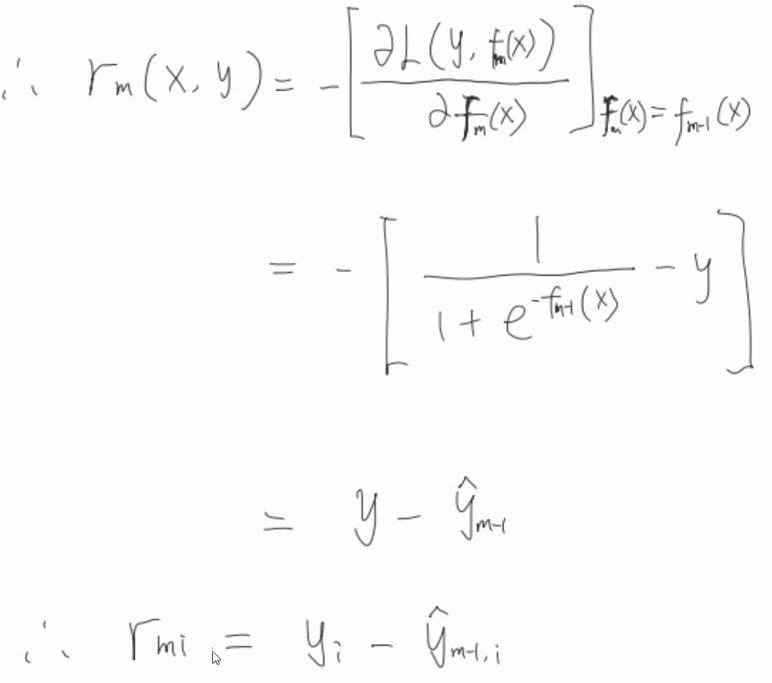
损失函数经过推导成标准形式后如下(具体推导就是把y^的表达式带入J)

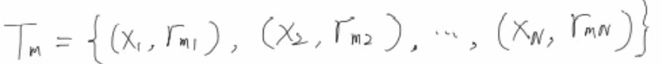


之后的步骤和上面解决一般决策问题时一样，求损失函数的负梯度，结果为



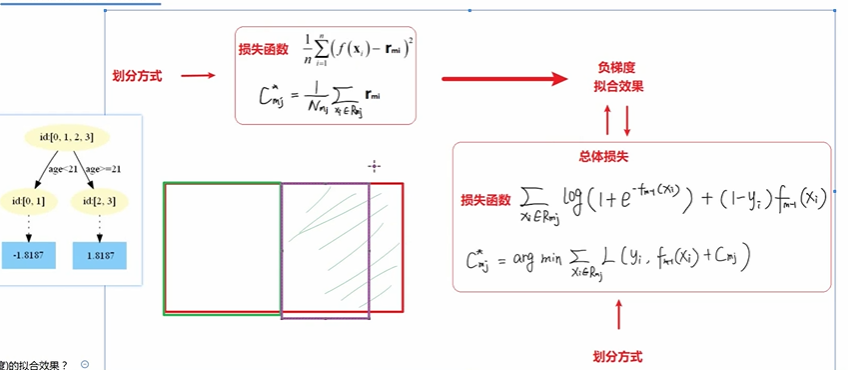
因此构造新的训练样本的公式为

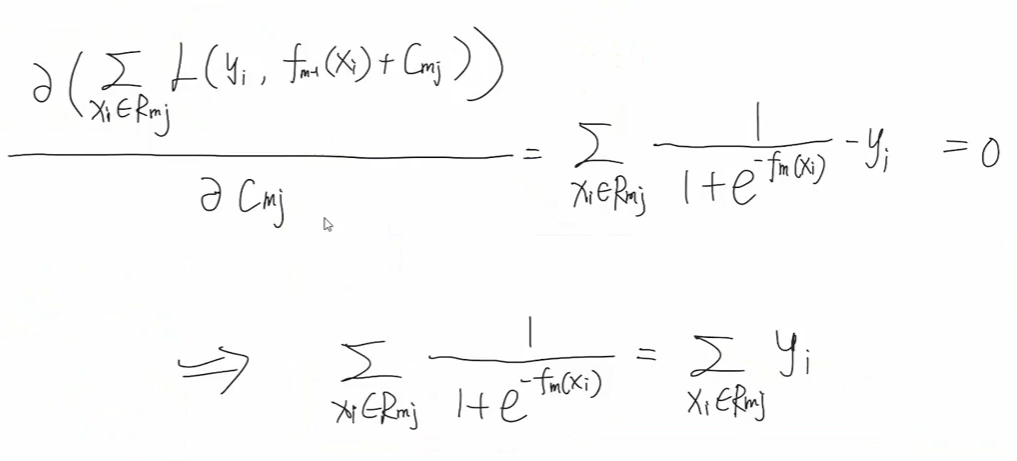




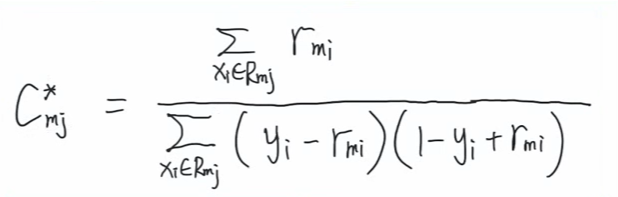
接下来就是用第m步的回归树去拟合Tm训练样本

但有一个问题是，如何去衡量对于残差的拟合效果，因为我们选择的损失函数很难是完美的，即损失函数能够帮助回归树去拟合负梯度的效果都是取决于损失函数本身的。因此可以换一个角度，不去着眼于损失函数的作用，而是通过衡量剩下尚未拟合的部分来衡量负梯度的拟合效果。但问题在于GBDT的损失函数求导后要求出Cmj(这里的Cmj相当于回归树的输出，即ymi)并没有闭式解，如下图所示，因此只能借助二阶泰勒展开来近似表示

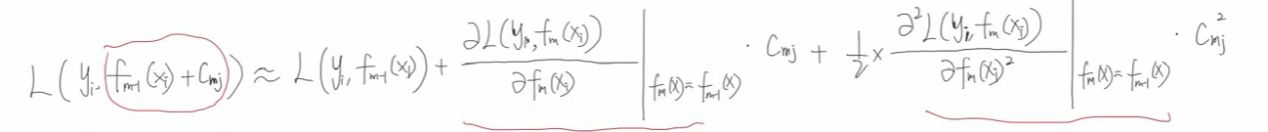
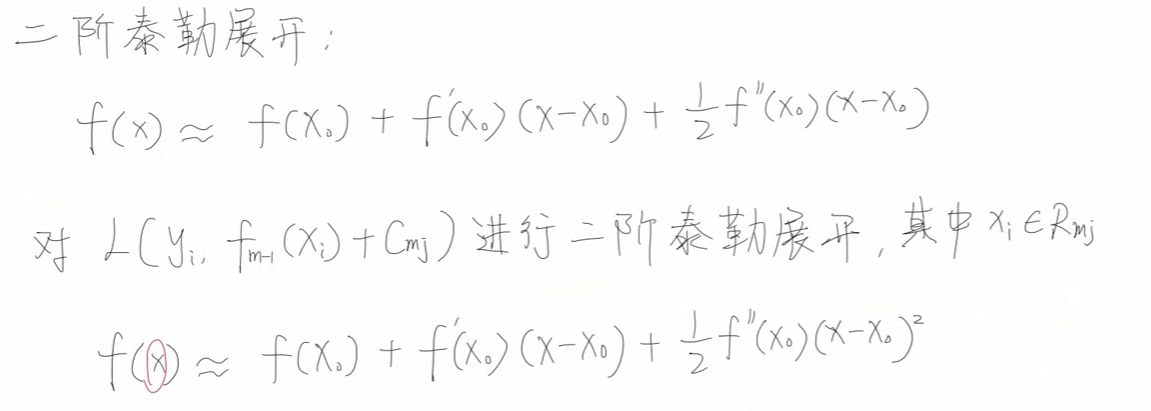


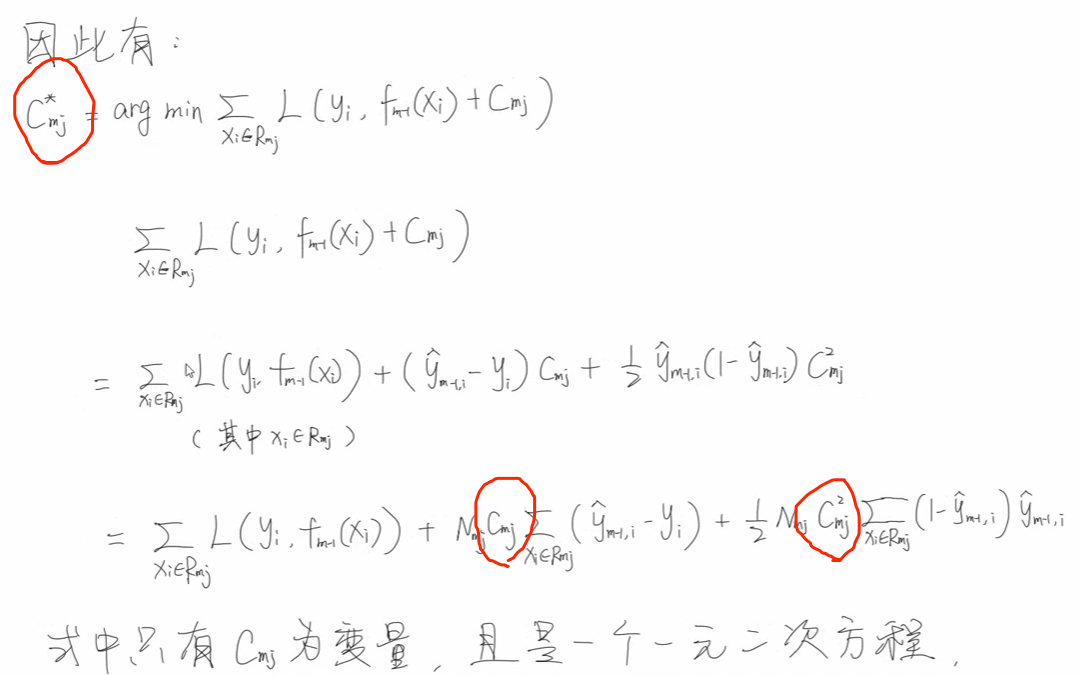
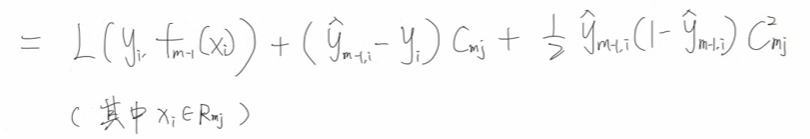


泰勒展开后的结果为

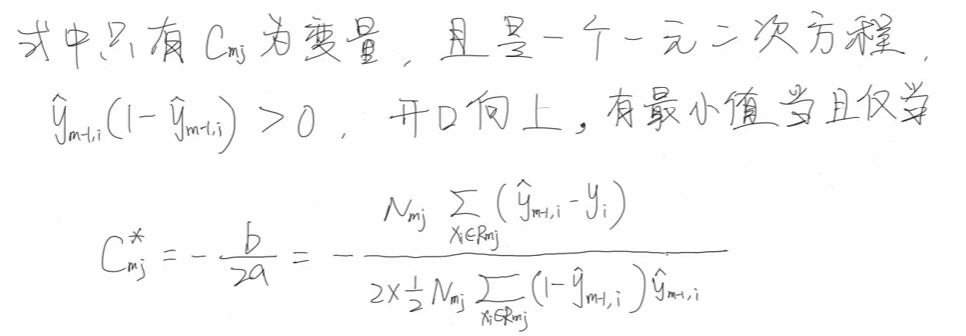


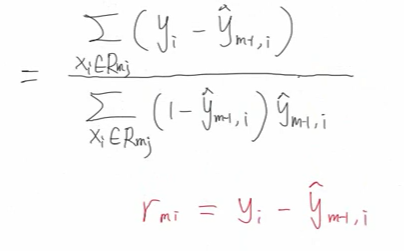
下面是推导过程

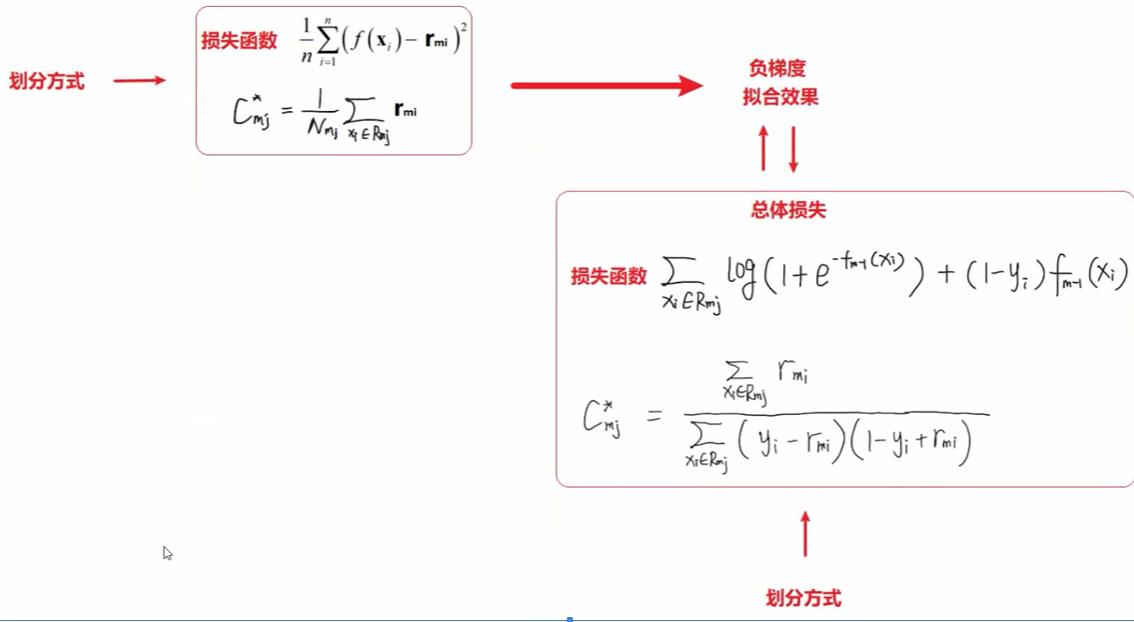




式中只有Cmj为变量，且是一元二次方程，开口向上，可求最小值当且仅当





因此上图课修改为

注：

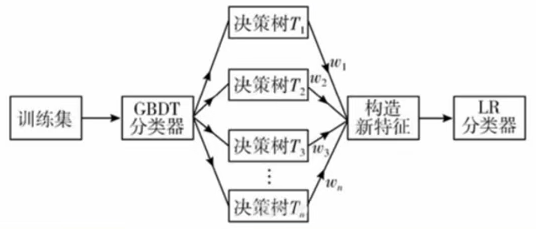
使用“总体损失”对回归树进行优化，计算过于复杂，而两种方法得出的划分方式都是一样的。

于是可以使用传统回归树构建好树的结构

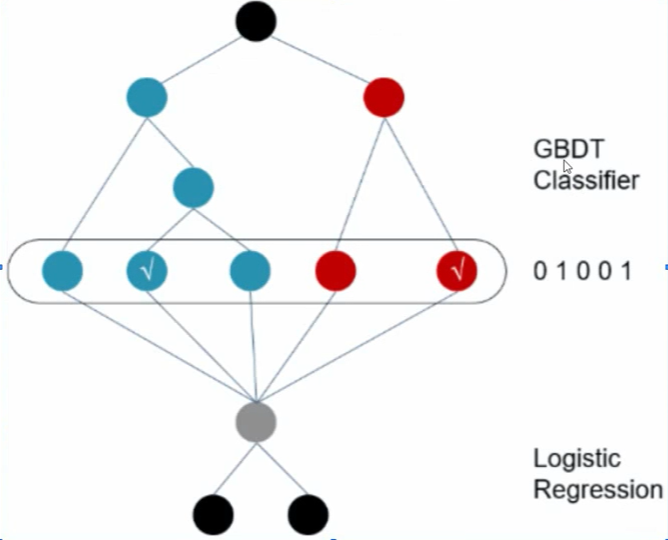
再用“总体损失”中的方式计算树的叶子节点中的cmj

## GBDT+LR 组合模型

LR模型无法进行特征交叉，特征筛选等，只能依赖于人工的特征工程，而GBDT+LR是将征工程模型化的开端。



可以把GBDT中树的生成过程理解成自动进行多维度的特征组合与特征筛选的过程



单独训练好GBDT和LR模型，再进行组合

打算用网上可查阅资料较多的[sklearn](https://so.csdn.net/so/search?q=sklearn&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/yyy430/article/details/_blank).ensemble.GradientBoostingClassifier实现

附一些觉得不错的GBDT文章：

1. <https://www.jianshu.com/p/005a4e6ac775>

# 分类模型2

Bert+TextCNN

前提知识：transformer机器模型

附带论文：《attention is all you need》

Bert是用transformer的encoder部分堆叠而成的语言预训练模型，而transformer的encoder部分又是由self-attention和

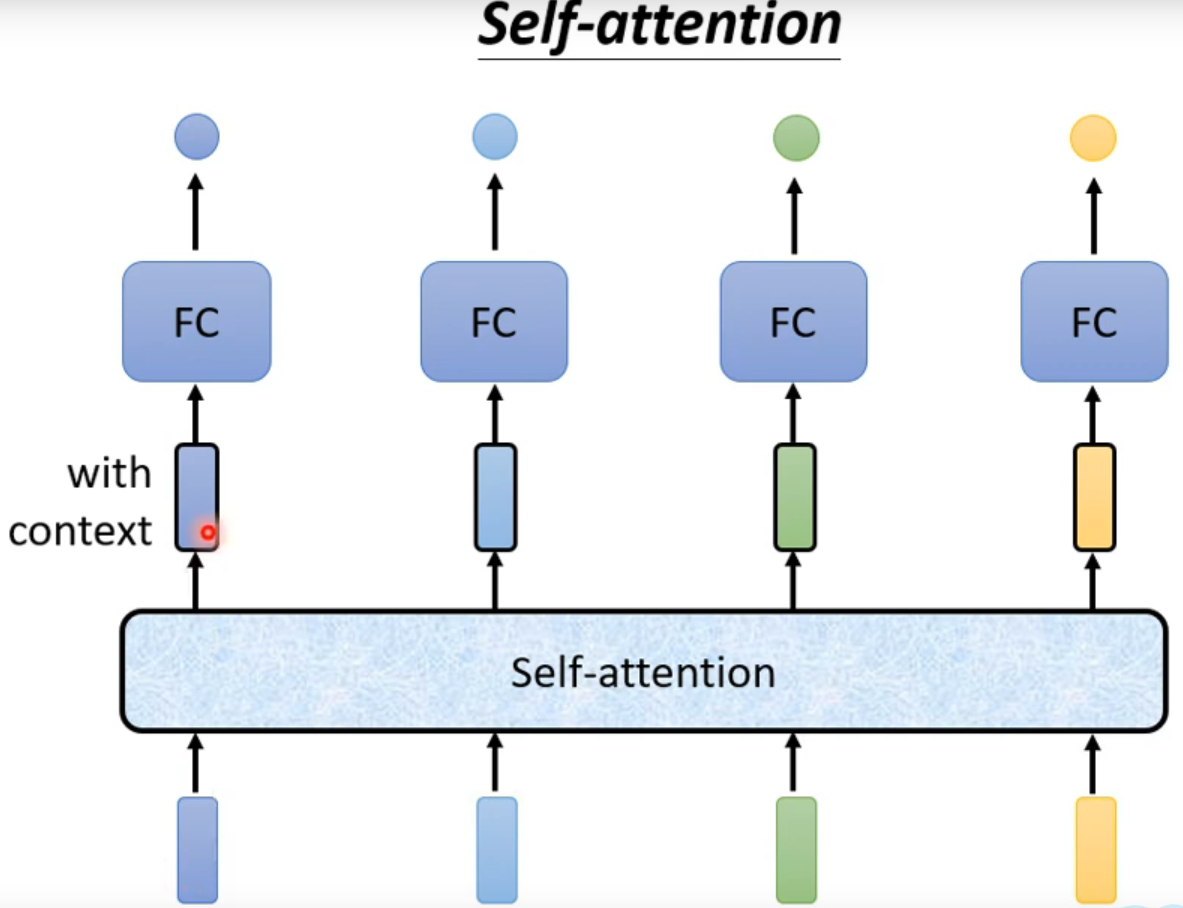
Bert可以解决知用一份非常大的预训练模型，让它能理解语言后，再用相对少量的专业化训练资料，就能得到很好的训练效果

Bert的缺点是需要非常大的训练资料，有点是Google，亚马逊等大公司都开源了它们与训练过的模型，使用者只需要在这之上训练一些专业化的训练数据就能得到较好的结果

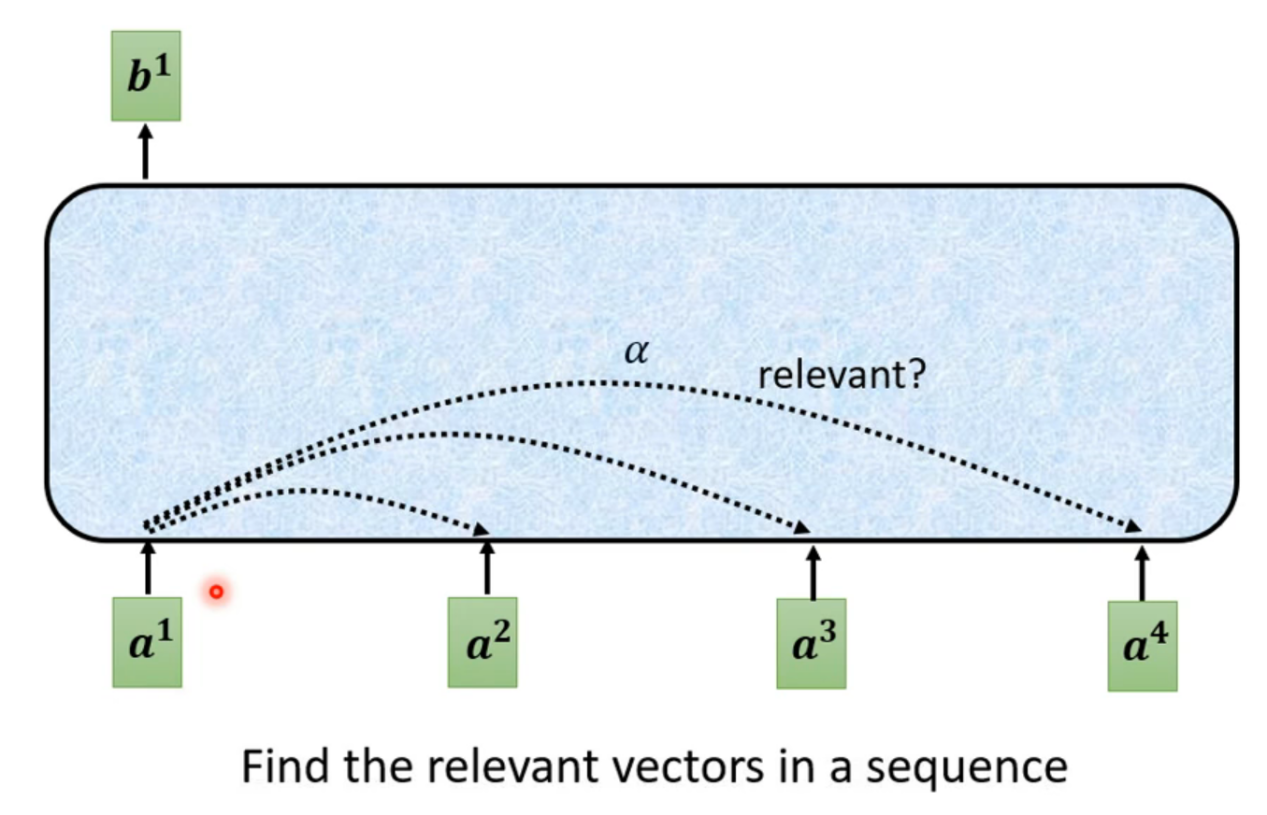
## Bert（了解，因为google都训练好了）

输入：两段句子

输出：两段句子的向量表示，特点是模型充分理解了每个单词的意思



Self-attention的结构如下

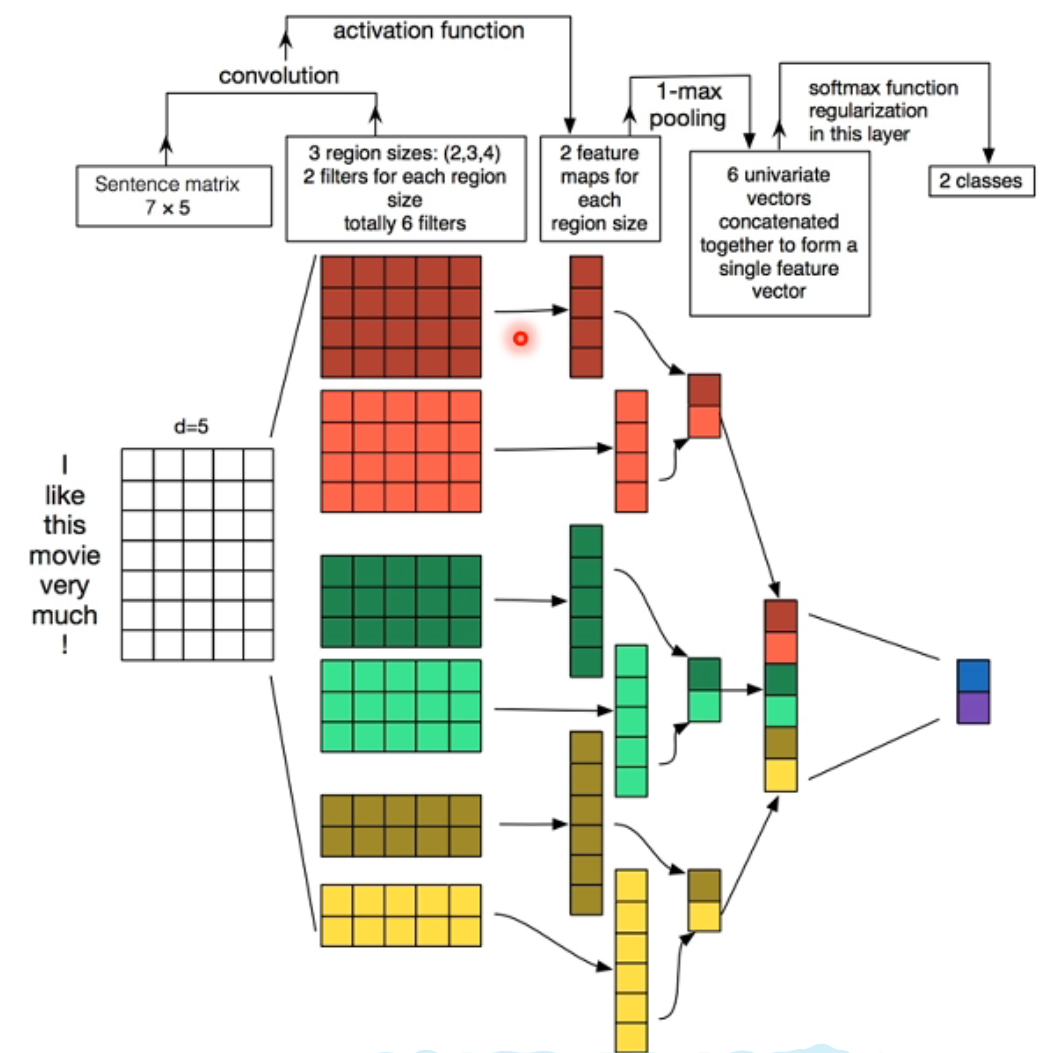


懒得写了，附李宏毅老师的讲解视频：

1. <https://www.bilibili.com/video/BV1Wv411h7kN?p=25> P25P26的自注意力机制讲解和P38P39的transoformer讲解
2. <https://www.bilibili.com/video/BV1hM4y157xX?from=search&seid=8610550601082068644&spm_id_from=333.337.0.0> P18P19的bert讲解
3. <https://www.bilibili.com/video/BV1PL411M7eQ?from=search&seid=9997900332185408282&spm_id_from=333.337.0.0> 李沐的bert论文逐段精读，读完最好自己再啃一遍论文
4. <https://www.bilibili.com/video/BV15L4y1v7ts/?spm_id_from=333.788.recommend_more_video.0> bert的fine-tune
5. <https://blog.csdn.net/jiaowoshouzi/article/details/89073944> 从transformer到self-attention到bert都讲到了，但不是很深
6. <https://wmathor.com/index.php/archives/1456/>

## textCNN

textCNN就是在CNN的基础上，它的输入由图片像素的向量变成词向量组，在网络结构上没有任何变化，且只有一层卷积，一层max-pooling，最后输出外接softmax来最后完成文本分类等任务

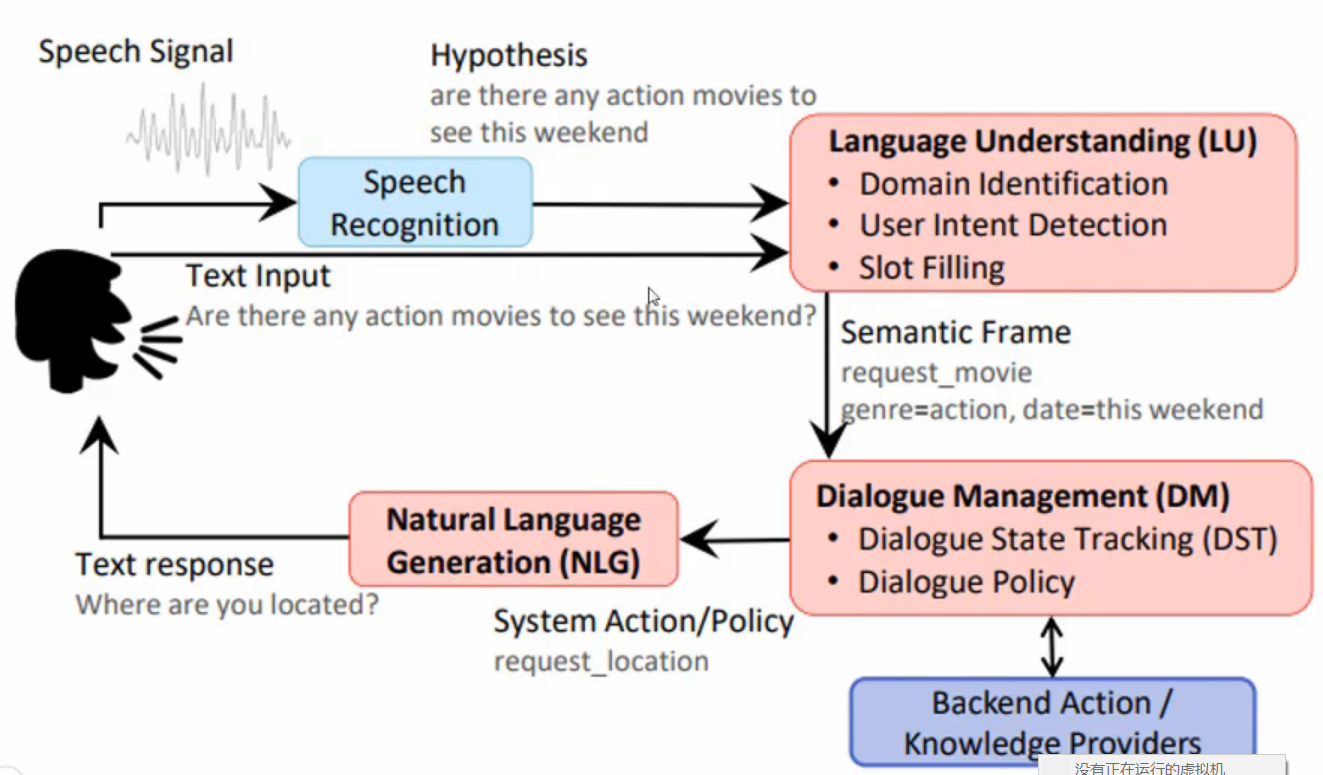


## Bert+textCNN

# NER模型

AC自动机提取所有子串+NER实体识别

**实体识别前需要实体标注**



**推荐顺序：闲聊意图识别->医疗意图识别->实体识别和槽位填充**

**DST模块**

**confirm操作**

**技术难点：**

1. **数据集(最大)**
2. **模型的训练**
   1. **分类模型(相对较简单)**
   2. **实体抽取模型(难点)**
3. **Rasa or pytorch(散装)？ 问题不大**
4. **后端即小程序(着急)**
5. **算力不够(GPU)**