用户预订售卖房型概率预测

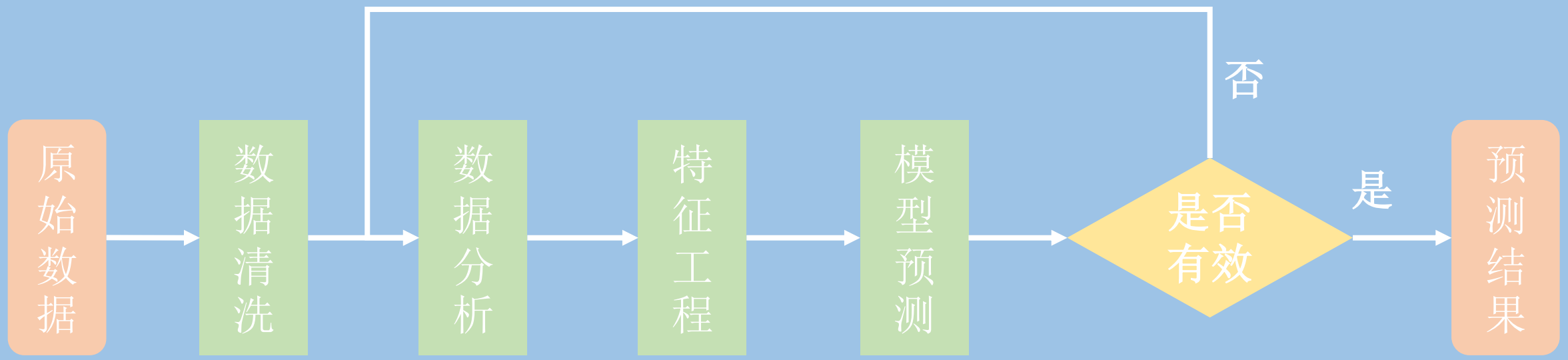
团队介绍

队名：看命
成员：**hwade**
来自：华南理工大学

问题描述

赛题为携程用户预订售卖房型的概率预测，官方提供的数据是初步统计处理后的数据，包括一些统计值（下单率等），使得选手减少了一定的原始数据粗加工的工作。无形中数据量也变大了不少，总数据量达到14G，同时里面还有大量无法直接使用的特征，原因在于本次赛题预测对象的特殊性：训练和测试样本中所有订单/用户都一定会有最终确认下单的行为。这就导致了基于用户和订单的一些统计或原始特征在这些用户和订单上没有区分性，进而对预测结果没有任何影响。所以必须做特征转换，或者简单去除这些特征以提高算法效率。

处理流程



数据清洗

由于数据量比较大，数据无法一次加载如内存，所以在第一步读入数据时采用分块读入的方法，每次读入chunksize大小的数据，同时还要及时回收内存。

```
data = pd.read_csv(file_name, iterator=True)
loop = True
while loop:
    try:
        data_c = data.get_chunk(chunksize)
        yield data_c
        del data_c
        gc.collect()
    except StopIteration:
        loop = False
```

数据清洗

对一些没有区分度的特征则进行处理，能够进行转换的特征暂时留着，不能转换的特征直接删除。

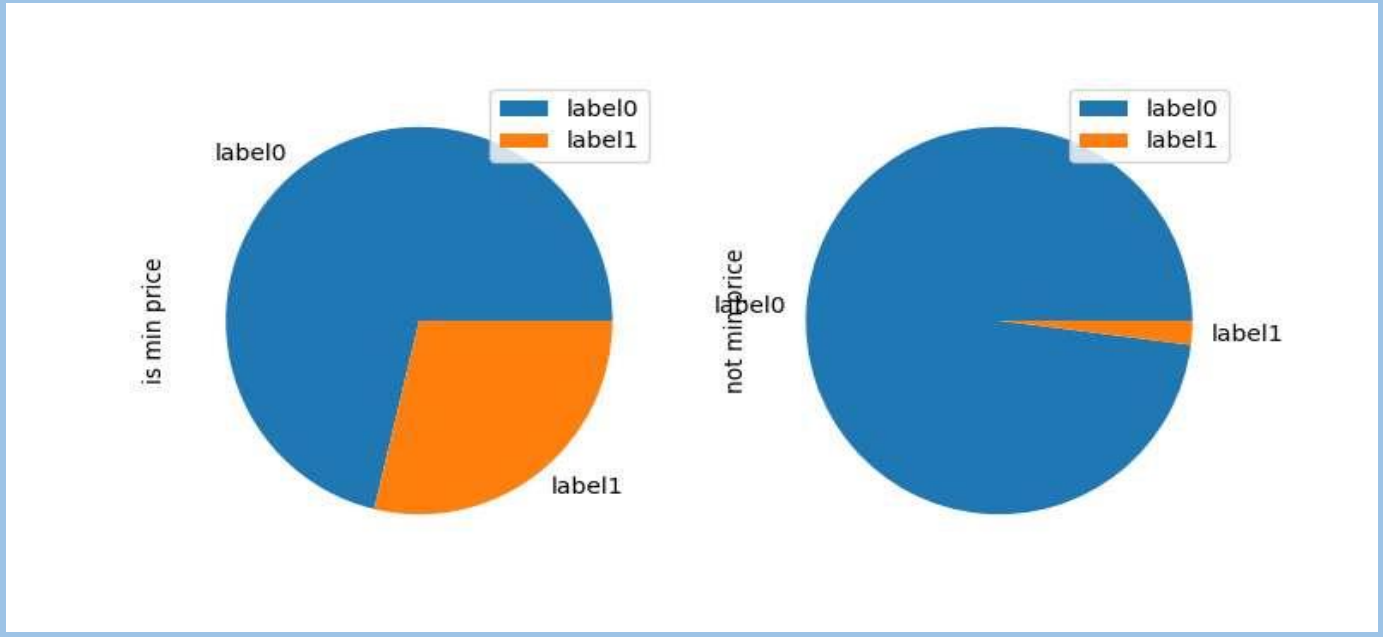
由于数据中存在大量的空值，如果简单的对含有空值的样本进行去除容易造成样本不足或者分布不平衡，所以采用不同方式对空值进行填充。

另外，需要采用低存储消耗的存储方式进行数据加载，如int64转成int32，float64转成float32.

```
for col in df_data.columns:
    if col.find('ratio') >= 0 or col.find('price') >= 0:
        df_data[col].fillna(df_data[col].mean(), inplace=True)
    elif col.find('roomservice') >= 0 or col.find('roomtag') >= 0:
        df_data[col].fillna(10, inplace=True)
    elif fill_method == 'mode':
        df_data[col].fillna(df_data[col].mode()[0], inplace=True)
    elif fill_method == 'mean':
        df_data[col].fillna(df_data[col].mean(), inplace=True)
    elif fill_method == 'quan':
        # quantile 0.5
        df_data[col].fillna(df_data[col].quantile(0.5), inplace=True)
    elif fill_method == 'pad':
        df_data[col].fillna(method='pad', inplace=True)
    elif fill_method == 'bfill':
        df_data[col].fillna(method='bfill', inplace=True)
```

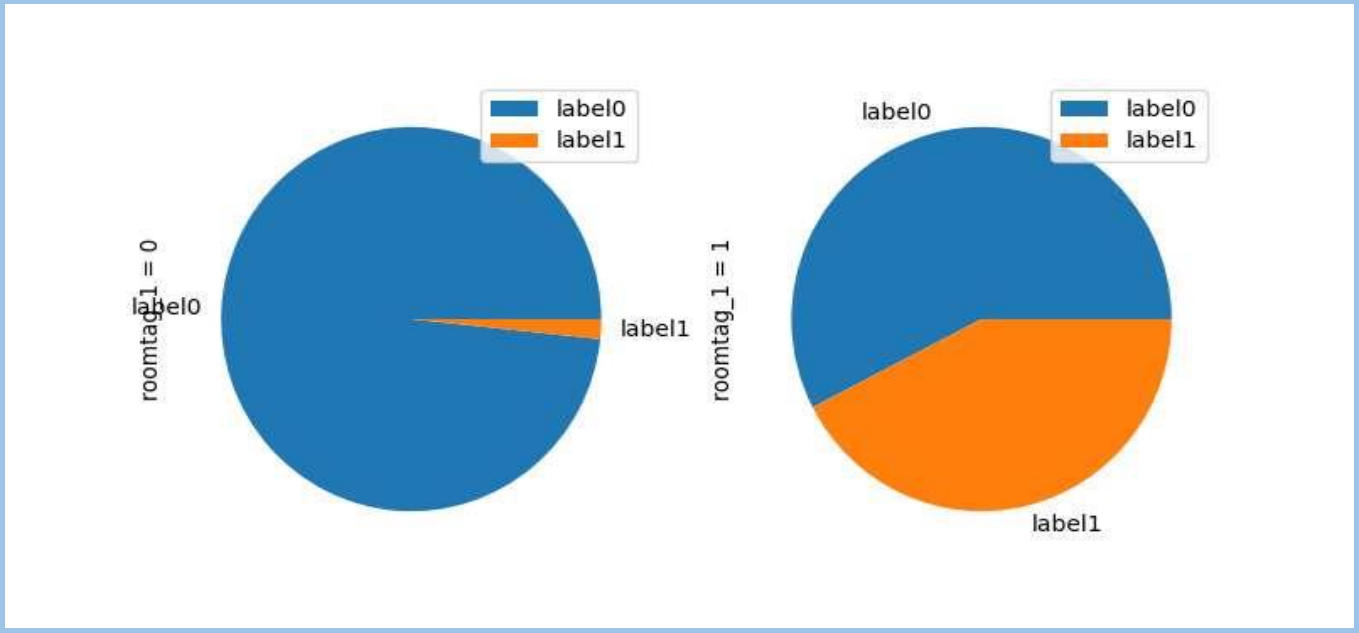
数据分析

数据中有很多维度的特征是针对价格的连续特征，从实际业务出发来思考，用户可能会因为价格的高低来决定是否下单，同一订单不同的房型中越低的价格越有可能下单。下图为标示是否为同一订单中价格最低的房型对最终是否下单的占比，可以看出最低价房型下单率明显比非最低价房型要高。



数据分析

下图表示物理房型roomtag_1为0和1的下单比率，可以看出roomtag_1为1时下单率接近50%，而为0时仅有不到5%的下单率。说明该特征是个强特征，在模型预测时具有很强的区分性。还有其他特征分析过程不一一列举。



特征工程

一般来说，首先需要对特征进行分类别，大致可分为用户特征、房型特征、交叉特征和其他特征，但是由于数据中所有用户/每一个订单都一定存在下单的记录，所以单独的用户特征并没有太多的意义。

根据上一步特征分析的结果，对特征进行转化。如对连续特征与最小值、中值、最大值、平均值等进行相减，用以表示该特征与具有标识性意义的数值之间的差值是多少。或者引入新特征对其进行判断，若该连续值等于该最小值（或其他分位值）时，标签置为1 。这种方法实际的意义在于从统计的角度来区分重要的数据和不重要的数据，例如某些用户更偏向于购买最便宜(min(price_deduct))的房型，某些用户更偏向购买最多优惠(max(returnvalue))的房型，有的则是偏向购买返现比最大的房型(max(price_deduct/returnvalue))

特征工程

根据上面的特征构造方法，可以构造出以下的特征：

price_diff_ordermin	price_deduct 和同一订单中最小房价的一阶差分
price_diff_ordermax	price_deduct 和同一订单中最大房价的一阶差分
price_diff_orderavg	price_deduct 和同一订单中平均房价的一阶差分
diff_deal_price	price_deduct和用户平均下单价格的一阶差分
diff_workday_price	price_deduct和用户工作日平均定单价格的一阶差分
diff_holiday_price	price_deduct和用户假日平均定单价格的一阶差分
price_diff_useravg	price_deduct和用户平均定单价格的一阶差分
diff_return_promotion	returnvalue和用户平均返现价格的一阶差分
diff_star_price	price_deduct和同一星级star的所有房型平均价格的一阶差分
diff_rank_price	price_deduct和同一排位rank的所有房型平均价格的一阶差分
…	其他特征

特征工程

统计上一次订单记录与本条记录的差别，可以构造出以下的特征：

is_last_roomid	该房型roomid是否是上一次下单的房型
is_last_basicroomid	该物理房型basicroomid是否是上一次下单的物理房型
is_last_rank	该位置rank是否是上一次下单的rank
is_last_star	该房型星级star是否是上一次下单的星级star
is_last_roomtag_2	该房型标签roomtag_2是否是上一次下单的房型标签
is_last_roomservice_2	该服务roomservice_2是否是上一次下单的服务类型
…	其他特征

特征工程

统计触发下单率类型的特征，可以构造出以下的特征：

roomtag_1_tri	该roomtag_1触发下单的次数
roomservice_1tri	该roomservice_1触发下单的次数
roomtag_1_tri_ratio	该roomtag_1触发下单的比率
roomservice_1_tri_ratio	该roomservice_1触发下单的比率
…	其他特征

统计方法直接涉及到orderlabel的值，所以需要做时间的滑窗来提取特征，否则容易造成数据泄露，如某一天的订单需要用订单以前的统计结果进行预测。由于时间关系，并未做滑窗提取。

特征工程

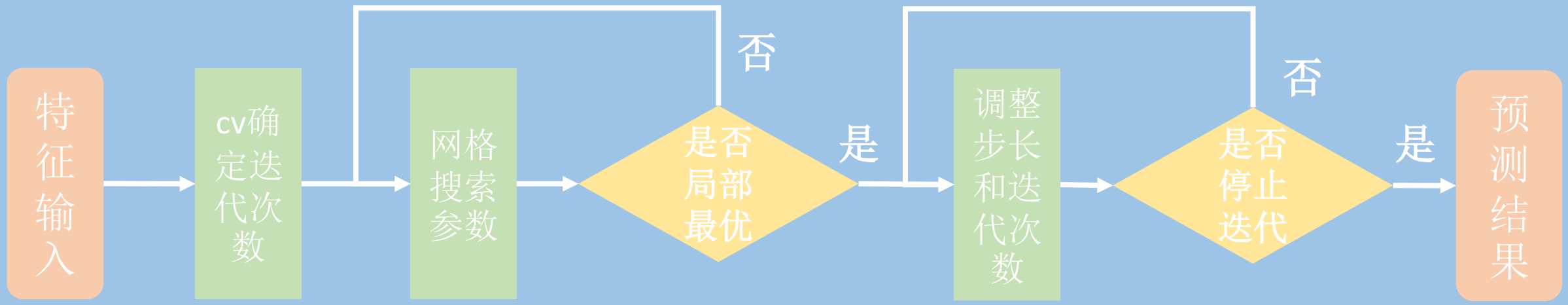
对强特征进行特征组合，构造的特征如下：

roomtag_service_1_1	roomtag_1*10 + roomservice_1 的简单合并
roomtag_1_rank	roomtag_1*10 + rank 的简单合并
…	其他特征

将强特征进行组合，可以更加充分地挖掘出特征的使用价值，例如**roomtag_1**这个特征，单独使用时具有很强的区分性，若同时考虑与其他特征组合时，可以更大限度地挖掘出深层规律。但是由于时间关系，并未做更多特征组合的尝试。

模型预测

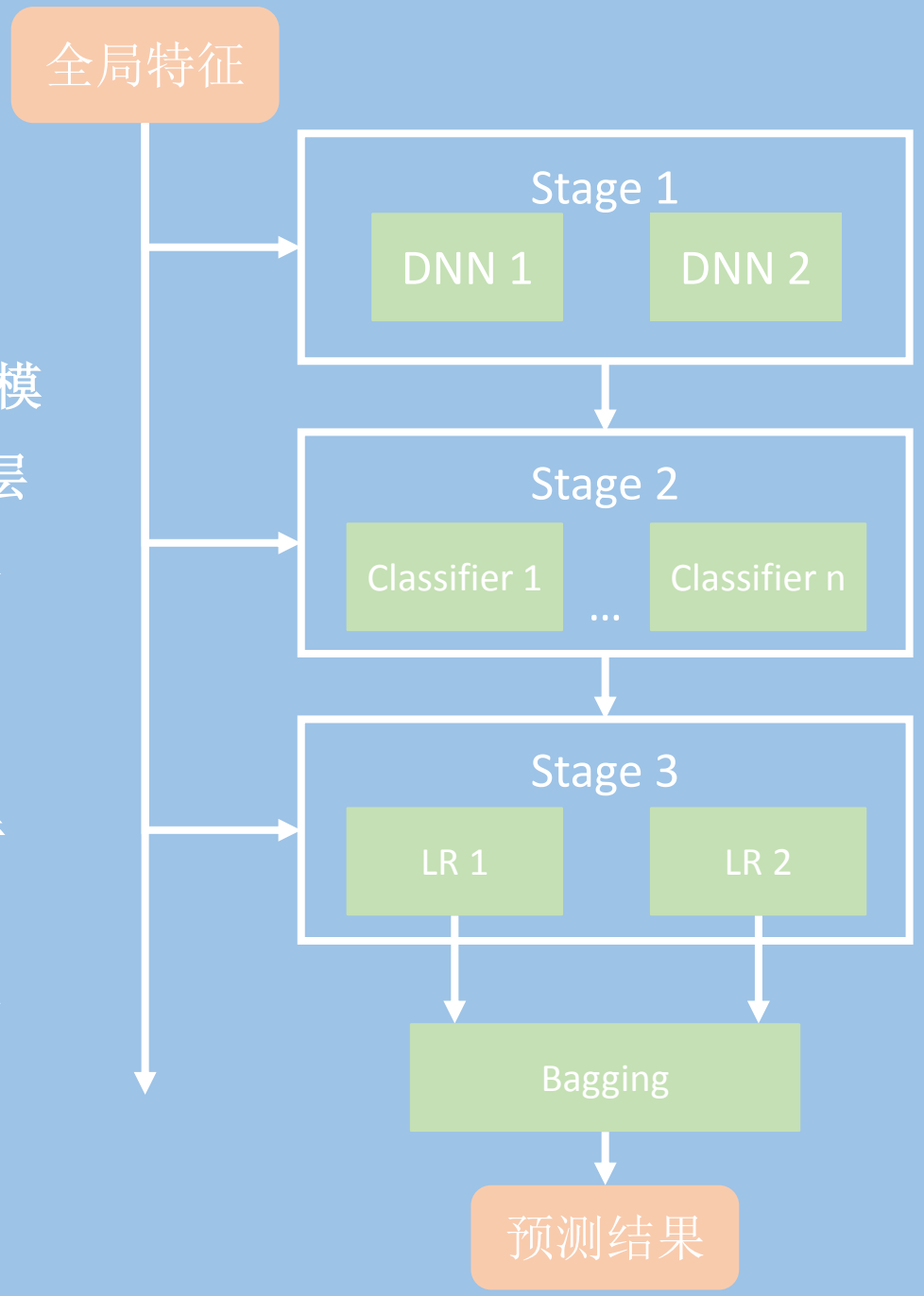
由于比赛做得比较晚，所以选用的是xgboost单模型，该模型是基于boosting的树模型，有比较稳定的预测结果，不易受异常值或者无用特征的影响，比较适合作为基础的单模型进行预测。单模型调参过程如下图所示，是一个贪婪算法思路，每次网格搜索最优参数为局部最优，为了加速迭代的过程，可以采用坐标下降等方法。



模型预测

以下仅为思路，比赛中并未实现。

采用ensemble的方法将模型进行融合。首先将复合模型分成3个层次，分别为stage1、stage2和stage3。第一层使用全局特征用深度网络分别进行预测，预测结果保存在本地；第二层使用不同参数的分类器用全局特征和第一层的预测值做为输入的特征；第三层将第二层的预测特征和全局特征合并，分别采用l1和l2线性的lr模型进行预测，分别预测出的结果再根据bagging方法进行加权。这样设计的目的在于充分利用各模型的特点，由于篇幅所限详细原因不做解释。模型设计如右图所示。



模型评估

由于比赛数据的特殊性，评估函数需要根据预测数据中的orderid分布进行划分，所以必须根据评测公式自己实现评估函数。采用auc作为评测函数。

在划分训练集和验证集时，由于orderid关系的捆绑，不好做随机划分，所以代码中只是简单地采用前80%作为训练集，后20%作为验证集。这样做的话可能会出现数据分布不一致的问题，训练出来的模型可能会过拟合。还有一个方案就是根据orderid来进行随机划分，不过并没有时间去实践。

比赛心得

本次比赛赛题和以往比赛类似，是二分类问题，但是提供的数据就有很大的不同，首先是进行了粗加工的数据。其次，数据中有许多维度的特征无法直接使用。另外，个人认为这次赛题的预测结果可能对实际业务提升并没有多大帮助，若确定一个用户一定会对某个酒店下单，只是预测这个用户下单的房型的话，个人觉得可能对整体的推荐帮助不大，如果是根据用户点击或浏览的数据来预测用户下单的行为，而且用户不是100%下单，这样可能实际意义更大一些。以上纯属个人愚见。