WinterfalI文档说明：

谢谢滴滴提供这次的数据和平台，下面是我队伍的模型说明

1．每支队伍需提交说明文档，并详细描述自己的算法思路和在比赛过程中模型成绩不断提升的改进方法，其中算法思路包含五个方面：

a. 数据预处理过程

a) 文件读取

i. 正常读取数据，树模型对数据分布要求较低，故省去

ii. poi: 用str.split()做字符处理，转成相应csv文件

iii. traffic：用str.split()做字符处理，转成相应csv文件

iv. order处理：

1. 取时间粒度为5min，i.e., 00:00~00:10，以及00:05~00:15。每天共有144*2 = 288个timeslice，每个timeslice长度为10min。

2. 统计各个timeslice的未应答订单数（driver_id == NULL）,记为’miss’。该label并没有标记成gap，因为是根据司机没有应答，错过了的订单，而并不代表真实的gap

3. 统计各个timeslice的去重未应答订单数（一直未被应答的订单），记为’gap’。由于有部分用户在失败后会多次重复呼叫，而最后得到应答，该行为会产生大量的miss，但并不一定会形成gap

4. 对各个timeslice，0~30min之内的未应答订单按照指数衰减加权，得到decaying_weighted_gaps,分去重gap和未去重gap，半衰期分为10min和5min。此特征会进行后续说明

b. 样本选取构造

a) 训练集选取：我们的样本选取经历了三个阶段的变化，分别是10分钟采样，5分钟采样以及2分钟采样。刚开始很长一段时间的10分钟为一时间片并且不重叠，然后发现数据训练量不够，因此想办法增加数据量，在与别的队伍进行交流沟通的过程中发现可以将时间片划分成5分钟一次，但线下及线上的效果表现并不理想，然后发现可以进行重叠，即时间片仍然是10分钟，但时间片与时间片间进行重叠，运用这个方法后，训练量从原本的20万变成近40万，极大的增加了训练样本，线下线上也有提升。在尝到该方法的甜头后，我们将采样变为2分钟一次，训练量增加到100万，但增加后预测结果非常奇特，线上线下表现也不好，因此我们最终还是回归了5分钟采样。

b) 另外，对于训练集的处理，由于刚开始处理数据的原因，请求为0的时间点被删除掉，之后发现该问题后将该时间点加回去，效果有所提升。

c) 交叉检验集的选取：试过很多种组合，最终还是返璞归真，将1月2日到1月15日进行训练，16日到21日对应时间片进行交叉检验。刚开始变化较为同步，后期纯粹看天吃饭，只能根据线上集以及以往的经验进行参数的调整，交叉检验和线上变化并不太同步。

c. 特征工程（需要详细说明，列举出所有的特征及其维度）

miss （target）： （1维）

未去重的gaps，直接统计在该时间段driver_id为null的数目，作为target进行训练。

gaps：（1维）

去重的gaps，该特征首先统计重复出现的passenger_id, start_id, destination_id, price这4个重复出现的变量，因为当一个乘客在重复呼叫时，这4个变量会保持一致。然后统计重复出现的passenger_id, start_id, destination_id, price，driver_id，因为若一名乘客长期打不了车，driver_id应一直为null，因此这两个数量一致的订单就为gap，即一致打不了车的乘客订单，供需缺口。

made：（1维）

对应miss的未去重的接单单数

made_new：（1维）

对应gap的去重接单数

decaying_weighted_gaps1_10min：（1维）

未去重gaps，10min半衰期加权，为了更好地体现过去30分钟的miss变化，因此我们考虑对过去30分钟的miss进行加权处理，一般会使用前10分钟一个权重，前10-20分钟一个权重，前20-30分钟一个权重，但这样赋予权重会使得10分钟之内的信息被抹去，同样采样粒度为5分钟，5分钟内的信息也会被抹去，因为为了更加细致地反应缺口的变化趋势，我们采用物理上的一个概念，半衰期，来进行趋势的分析，没5分钟，或者每10分钟，权重衰减1半，其中每个时刻的权重都不一样，公式如下：

$$\frac{1}{2}N_{(0)} = N_0 e^{-\lambda t_{\frac{1}{2}}}$$

其中$t_{1/2}$为半衰期的周期，这里我们分别用了5分钟和10分钟来进行计权，同时也对gap和miss分别进行了统计。下三个变量同。

decaying_weighted_gaps1_5min：（1维）

未去重gaps，5min半衰期加权

decaying_weighted_gaps2_10min：（1维）

去重gaps，10min半衰期加权

decaying_weighted_gaps2_5min：（1维）

去重gaps，5min半衰期加权

made_new：（1维）

去重后的接单数

coming：（1维）

希望使用目的地为该地区以及价钱的信息，因此通过coming这个变量将两者融合起来。虽然滴滴在对顾客进行了较大力度的补贴，即使同一点到另一点价钱的变化会较大，但总体而言，价格还实惠随着距离以及交通拥堵程度有一个正太分布的量。当距离较远，或行驶时间较长，价格就越高，同时到达该地区支援的时间就越长，因此通过统计目的地为该地区的总数，同时通过价格的倒数进行加权，以此来反映到达该地区的时间。曾考虑把地区间的距离作为权重，但由于地区也挺大的，点和点都不一样，同时交通拥堵情况也不能反映，因此还是使用价格倒数进行加权。

nearest_gaps：（1维）

模型还希望运用区域周边区域的特征，区域间的距离主要通过价钱来进行判断，价钱能判断区域间的距离已经在上一个特征中叙述过，当前district最近邻3个district的gaps之和，按照距离（price）倒数加权

weather及其变化量：（2维）

pm25及其变化量：（2维）

temperature及其变化量：（2维）

traffic_1:traffic_4：（4维）

week：（1维）

d. 模型选取

a) 本模型主要是通过两大不同的模型进行融合，一个是python中sklearn的开源库，选取合适的回归去来进行回归。另一个主要

模型是通过已有的软件lambdamart来进行预测，这是一套成熟算法开发的软件，预测结果较好。

b) 对于第一个模型，即sklearn的开源库模型，我们在初赛阶段就进行了部分模型的尝试，在线下是GBRT模型最佳，且自带的huber loss functionyo有较好的效果。通过前期的数据预处理和分析，很容易发现预测区域两级分化严重，gap较小的区域占了绝大部分，但gap大的部分是较难预测准确，即mape值较高的部分，因此两部分进行分开系数的选取以达到最优的系数组合，先通过线下验证集的来进行趋势的判断，发现当树的深度较浅时，预测的均值都较小，且小于等于1的值更多，而当树的深度变深，预测的值均逐渐增大，因此通过线下调试，确认对于gap较小的区域，树的深度在7-8较为合适，对于gap较大的值，树的深度在11-12较为合适。最终通过线上的确认，并对learning_rate和惩罚权重alpha的微调，得到该模型的系数。该模型主要用了3+5*3个特征，由于我们的时间粒度为5分钟采样，并且题目给出前30分钟的信息进行预测，故是通过前5个时间片的信息预测第7个时间片的信息。3位区域编号，时间片以及周一到周五，由于用的是非线性的树模型，因此并不需要向量化的操作，故直接赋1维变量。

   i. 对于gap大的区域，5*3为5个时间片的应答和打车失败的gap以及对应的需求和coming变量。但由于faq中写道不去重，因此该值应为真实的gap，但并不是本题目需要预测的gap。但gap大的区域万一真实值很低，但模型预测大了，风险很大，因此该部分采用了真实的gap作为变量，真实gap比预测的gap稍小，以此来达到对gap进行保守估计的效果。

ii. 对于gap小的区域，采取需要预测的gap和对应的总需求，在线下该组合表现最佳。

iii. 由于我们使用sklearn的gbrt模型，而并非xgboost的模型，因此加入其它特征将会极大延长代码的运行时间，而在我们的参数下加入其它特征的提升似乎并不大（线下测试），因此我们在该模型中并未加入其它特征。

c) 对于另一个主模型，我们采取了LambdaMART的方法，由于之前一直使用的是现有的软件，所以无法提交对应的代码。虽然LambdaMART的算法有现成的开源库，在最后才发现需要提交代码，而之前一直使用成熟的软件，因此没有足够的时间将此段模型进行重现。虽然如果有机会进入决赛，我将会尝试将软件的结果通过开源的库进行重现。不过在现阶段没办法做到，抱歉。在这里我只能描述我在该模型中所使用的特征以及mart的模型。我们在该模型中用的时间粒度同样为5分钟，即5分钟采样一次，所以所有的变量每天的数目均为288 * 66个。当然，前面不足的30分钟时间会被去掉。加入的特征有decaying_weighted_gaps1_10min, decaying_weighted_gaps2_5min, coming, 天气，pm25等量极其变化量，临近3个地区的30分钟miss值以及其它常规项。特征的作用与来源已经在特征工程那一部分叙述过，在此不多加叙述。mart的算法比较高校，收敛较快，因此不同于sklearn的gbrt函数，mart通过mini－batch的梯度下降算法，实现了较快的计算速度，因此在调试中便于可以多增加特征的量并获得较快的参数调试速度。模型的调试是按照上述所说的训练集合交叉检验集来划分，通过调参数来达到线下最佳的成绩，然后再在线上进行轻微的调试。

e. **模型融合**

a) 模型融合的系数，简单来说就是根据线上成绩来进行模型系数的筛选，由于线上测试样本数量较少，只有2838个，因此为了提高线上成绩，系数主要根据线上成绩进行调整和筛选，虽然这并不符合数据挖掘的本质，即根据已有数据，通过妥当的交叉检验设置，尽可能地调整模型的参数以达到最高的预测准确率，但由于线上样本太少以及尽可能高的线上成绩，因此我们采用了根据线上成绩这一下策。

这就是winterfall的模型说明报告，谢谢！