携程机票航班延误预测算法大赛

背景：随着国内民航的不断发展，航空出行已经成为人们比较普遍的出行方式，但是航班延误却成为旅客们比较头疼的问题。台风，雾霾或飞机故障等因素都有可能导致大面积航班延误的情况。大面积延误给旅客出行带来很多不便，如何在计划起飞前2小时预测航班延误情况，让出行旅客更好的规划出行方式，成为一个重大课题。

要求：提前2小时（航班计划起飞时间前2小时），预测航班是否会延误3小时以上（给出延误3小时以上的概率）

回归问题

Online dataset path:

../input/ctrip001/data/train\_data/airport\_city.csv

../input/ctrip001/data/train\_data/flight\_information.csv

../input/ctrip001/data/train\_data/weather.csv

../input/ctrip001/data/train\_data/special\_news.csv

数据集：

* 航班动态历史数据表

出发机场 到达机场 航班编号 计划起飞时间 计划到达时间 实际起飞时间实际到达时间 飞机编号 航班是否取消

* 城市天气表

城市名 天气 当天最低温度 当天最高温度 日期

* 机场城市对应表

机场编码 城市名称

* 机场特情表

特情机场 收集时间 开始时间 结束时间 特情内容

**Feature**

航班基本信息

1. 航班编号'
2. 计划起飞时间
3. 计划到达时间
4. 飞机编号
5. 飞行时间(flightTime)

航班历史延误信息

1. 平均延误时间
2. 延误时间标准差
3. 最大延误时间
4. 延误时间中位数

飞机编号、日期 ===> 前序航班

1. 前序航班（lastFlight）
2. 前序航班延误时间（timeLastFlightDelay）
3. 若本航班不延误，准备本次飞行的剩余时间（本次飞行预计起飞时间 - 上次飞行实际到达时间 timePrepareThisFlightRemain）
4. 计划准备本次飞行的时间（本次飞行预计起飞时间 - 上次飞行计划到达时间timePrepareThisFlightPlan）

机场 – 城市 – 天气

1. 出发机场最低气温
2. 出发机场最高气温
3. 出发机场天气（weatherVecFrom）
4. 到达机场最低气温
5. 到达机场最高气温
6. 到达机场天气（weatherVecTo）

机场特情

1. 出发机场或到达机场是否有特情

**Label**

1. 计划起飞时间、实际起飞时间 ===> 航班延误时间，是否延误三小时以上

**抽样** 随机抽取正样本，数量为负样本数量的2倍

1. Word2Vec使用

gensim.models.word2vec.Word2Vec

model = Word2Vec(sentences, size=100, window=5, min\_count=5, workers=4)

参数详解：

sentences：可以是一个·ist，对于大语料集，建议使用BrownCorpus,Text8Corpus或·ineSentence构建。  
sg： 用于设置训练算法，默认为0，对应CBOW算法；sg=1则采用skip-gram算法。  
size：是指特征向量的维度，默认为100。大的size需要更多的训练数据,但是效果会更好. 推荐值为几十到几百。  
window：表示当前词与预测词在一个句子中的最大距离是多少  
alpha: 是学习速率  
seed：用于随机数发生器。与初始化词向量有关。  
min\_count: 可以对字典做截断. 词频少于min\_count次数的单词会被丢弃掉, 默认值为5  
max\_vocab\_size: 设置词向量构建期间的RAM限制。如果所有独立单词个数超过这个，则就消除掉其中最不频繁的一个。每一千万个单词需要大约1GB的RAM。设置成None则没有限制。  
sample: 高频词汇的随机降采样的配置阈值，默认为1e-3，范围是(0,1e-5)  
workers参数控制训练的并行数。  
hs: 如果为1则会采用hierarchica·softmax技巧。如果设置为0（defau·t），则negative sampling会被使用。  
negative: 如果>0,则会采用negativesamp·ing，用于设置多少个noise words  
cbow\_mean: 如果为0，则采用上下文词向量的和，如果为1（defau·t）则采用均值。只有使用CBOW的时候才起作用。  
hashfxn： hash函数来初始化权重。默认使用python的hash函数  
iter： 迭代次数，默认为5  
trim\_rule： 用于设置词汇表的整理规则，指定那些单词要留下，哪些要被删除。可以设置为None（min\_count会被使用）或者一个接受()并返回RU·E\_DISCARD,uti·s.RU·E\_KEEP或者uti·s.RU·E\_DEFAU·T的函数。  
sorted\_vocab： 如果为1（defau·t），则在分配word index 的时候会先对单词基于频率降序排序。  
batch\_words：每一批的传递给线程的单词的数量，默认为10000

1. dataFrame – merge, append, concatenate, join

<http://pandas.pydata.org/pandas-docs/stable/merging.html>

# Merge: 通过键拼接列

用于通过一个或多个键将两个数据集的行连接起来，类似于 SQL 中的 JOIN。该函数的典型应用场景是，针对同一个主键存在两张包含不同字段的表，现在我们想把他们整合到一张表里。在此典型情况下，结果集的行数并没有增加，列数则为两个元数据的列数和减去连接键的数量。  
on=None 用于显示指定列名（键名），如果该列在两个对象上的列名不同，则可以通过 left\_on=None, right\_on=None 来分别指定。或者想直接使用行索引作为连接键的话，就将 left\_index=False, right\_index=False 设为 True。  
how='inner' 参数指的是当左右两个对象中存在不重合的键时，取结果的方式：inner 代表交集；outer 代表并集；left 和 right 分别为取一边。  
suffixes=('\_x','\_y') 指的是当左右对象中存在除连接键外的同名列时，结果集中的区分方式，可以各加一个小尾巴。  
对于多对多连接，结果采用的是行的笛卡尔积。

参数说明：  
left与right：两个不同的DataFrame  
how：指的是合并(连接)的方式有inner(内连接),left(左外连接),right(右外连接),outer(全外连接);默认为inner  
on : 指的是用于连接的列索引名称。必须存在右右两个DataFrame对象中，如果没有指定且其他参数也未指定则以两个DataFrame的列名交集做为连接键  
left\_on：左则DataFrame中用作连接键的列名;这个参数中左右列名不相同，但代表的含义相同时非常有用。  
right\_on：右则DataFrame中用作 连接键的列名  
left\_index：使用左则DataFrame中的行索引做为连接键  
right\_index：使用右则DataFrame中的行索引做为连接键  
sort：默认为True，将合并的数据进行排序。在大多数情况下设置为False可以提高性能  
suffixes：字符串值组成的元组，用于指定当左右DataFrame存在相同列名时在列名后面附加的后缀名称，默认为('\_x','\_y')  
copy：默认为True,总是将数据复制到数据结构中；大多数情况下设置为False可以提高性能  
indicator：在 0.17.0中还增加了一个显示合并数据中来源情况；如只来自己于左边(left\_only)、两者(both)

# join 拼接列，主要用于索引上的合并

# concat  可以沿着一条轴将多个对象连接到一起

1. **提取特征**

经验：

分块，多保存，打印进度，抽样均衡样本，添加特征