[关于2015阿里移动推荐算法大赛的总结（一）](http://blog.csdn.net/sin_geek/article/details/45715981)

**写在最前面，第一场比赛的第一轮早已结束，lz组的团队已被淘汰~跟lz组队的人跟lz一样也是新手菜鸟，参加比赛只是兴趣与好奇。现在第二场比赛开始了，lz还会继续玩下去（虽然lz校内科研、项目压力都很大）~**

官方给了两个集合表——用户在商品全集上的移动端行为数据表和商品子集表。行为数据里面分别有用户名、商品名、行为类型（浏览、收藏、加购物车、购买）、地理位置、商品类别、行为时间。商品子集里面有商品名、商品类别、地理位置。

题目是根据官方给的一个月的用户行为数据，预测这个月过完的第一天在商品子集中用户可能购买的商品。评分使用的是正确率与召回率的加权。

首先，我们想到的是根据行为的统计特征进行购买的预测，简单的说就是假如用户加入购物车，那么购买的可能性就很大，收藏，浏览产生购买行为的可能性递减。

同时还有地理位置这组信息，从经验判断，地理位置相近购买可能性也越大，在武汉的童鞋一般不会买杭州的电影票或者餐券吧。（这里要说明下，因为是移动端行为数据，很大一部分是类似电影票、门票、代金券、外卖等等商品，当然也肯定会有我们熟知的淘宝物品。）

还有一点，很显然商品子集里的商品是行为数据表中商品的子集（也就是说假如购买了pad，会继续购买个pad套，而这个pad套在子集里，pad只在行为数据表里，需要寻找出这样一种规律）。

初期，我们把侧重点放在题目的“推荐”两个字上了，搜集并研究了大量的“推荐算法”，然后也分析整理出了一套模型，但是在最后要得出结果那一步发现，根本不是题目要的结果。题目要的就是在那一天用户会购买那些商品，而不是用户可能对那些商品感兴趣。这样我们就推翻了我们的这条路，当然我们自己在这条“黑路”上也学到了不少，后面我会单独总结整理一下，我们关于“推荐算法”的种种。

后来，看了一些官方给的建议。我们才明确，简单的说我们要做就是二分法的预测，买还是不买。然后我们就想到对用户行为抽象为特征向量，利用时间划分数据，使用机器学习的方法训练模型，对某一天的购买行为进行预测。我们在这方面也是从开始学习，到做了一些工作，分析总结。虽然最后的结果还是差强人意，最终是948名，但是中间有最好排名303过（最终取前500进入第二轮）。关于这部分的工作，后面我也会单独总结整理的。这里就先开个总结的头，逼自己把做过的东西整理出来~~~

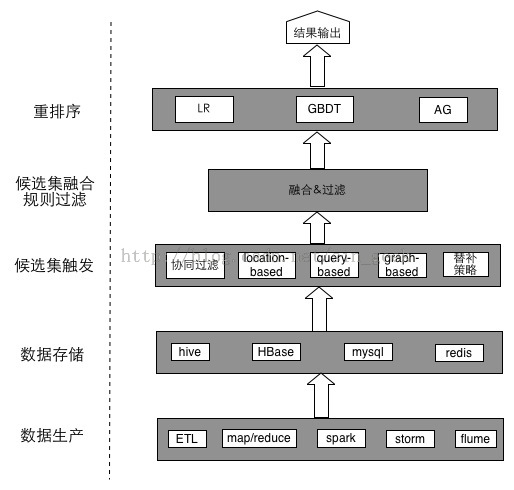
# [关于2015阿里移动推荐算法大赛的总结（二）——推荐算法](http://blog.csdn.net/sin_geek/article/details/45765393)

虽然开始走错了路，但是也学到了东西，美团技术团队的文档还是不错的，喜欢的童鞋可以经常去瞅瞅，后面我会给链接的~~~~

——————————————————————————————————————————————————————————————

# 具体流程

基本流程如下，借用美团的图。



从框架的角度看，推荐系统基本可以分为数据层、触发层、融合过滤层和排序层。数据层包括数据生成和数据存储，主要是利用各种数据处理工具对原始日志进行清洗，处理成格式化的数据，落地到不同类型的存储系统中，供下游的算法和模型使用。候选集触发层主要是从用户的历史行为、实时行为、地理位置等角度利用各种触发策略产生推荐的候选集。候选集融合和过滤层有两个功能，一是对出发层产生的不同候选集进行融合，提高推荐策略的覆盖度和精度；另外还要承担一定的过滤职责，从产品、运营的角度确定一些人工规则，过滤掉不符合条件的item。排序层主要是利用机器学习的模型对触发层筛选出来的候选集进行重排序。

在这次比赛中相当于给了数据，不需要考虑数据产生，有可能要考虑存储，暂时先不考虑。所以大体流程是先对数据进行分析，然后对数据进行预处理，进入候选集触发环节，考虑采用协同过滤与位置聚类的方法推荐出集合，然后通过机器学习的方法训练得出最终结果。

# 理论分析

## 数据应用

| 行为类别 | 行为详情 |
| --- | --- |
| 主动行为数据 | 搜索、筛选、点击、收藏、下单、支付、评分 |
| UGC | 文本评价、上传图片 |
| 负反馈数据 | 左滑删除、取消收藏、取消订单、退款、负评、低评 |
| 用户画像 | 用户人口属性、美团DNA、品类偏好、消费水平、工作地与居住地 |

用户主动行为数据记录了用户在美团平台上不同的环节的各种行为，这些行为一方面用于候选集触发算法（在下一部分介绍）中的离线计算（主要是浏览、下单），另外一方面，这些行为代表的意图的强弱不同，因此在训练重排序模型时可以针对不同的行为设定不同的回归目标值，以更细地刻画用户的行为强弱程度。此外，用户对deal的这些行为还可以作为重排序模型的交叉特征，用于模型的离线训练和在线预测。

负反馈数据反映了当前的结果可能在某些方面不能满足用户的需求，因此在后续的候选集触发过程中需要考虑对特定的因素进行过滤或者降权，降低负面因素再次出现的几率，提高用户体验；同时在重排序的模型训练中，负反馈数据可以作为不可多得的负例参与模型训练，这些负例要比那些展示后未点击、未下单的样本显著的多。

用户画像是刻画用户属性的基础数据，其中有些是直接获取的原始数据，有些是经过挖掘的二次加工数据，这些属性一方面可以用于候选集触发过程中对deal进行加权或降权，另外一方面可以作为重排序模型中的用户维度特征。

通过对UGC数据的挖掘可以提取出一些关键词，然后使用这些关键词给deal打标签，用于deal的个性化展示。

# 推荐引擎

## 1、推荐引擎是不是为不同的用户推荐不同的数据

**根据大众行为的推荐引擎**，对每个用户都给出同样的推荐，这些推荐可以是静态的由系统管理员人工设定的，或者基于系统所有用户的反馈统计计算出的当下比较流行的物品。

**个性化推荐引擎**，对不同的用户，根据他们的口味和喜好给出更加精确的推荐，这时，系统需要了解需推荐内容和用户的特质，或者基于社会化网络，通过找到与当前用户相同喜好的用户，实现推荐。

这是一个最基本的推荐引擎分类，其实大部分人们讨论的推荐引擎都是将个性化的推荐引擎，因为从根本上说，只有个性化的推荐引擎才是更加智能的信息发现过程。

## 2、根据推荐引擎的数据源

其实这里讲的是如何发现数据的相关性，因为大部分推荐引擎的工作原理还是基于物品或者用户的相似集进行推荐。根据不同的数据源发现数据相关性的方法可以分为以下几种：

根据系统用户的基本信息发现用户的相关程度，这种被称为**基于人口统计学的推荐**（Demographic-based Recommendation）

根据推荐物品或内容的元数据，发现物品或者内容的相关性，这种被称为**基于内容的推荐**（Content-based Recommendation）

根据用户对物品或者信息的偏好，发现物品或者内容本身的相关性，或者是发现用户的相关性，这种被称为**基于协同过滤的推荐**（CollaborativeFiltering-based Recommendation）。

## 3、根据推荐模型的建立方式

可以想象在海量物品和用户的系统中，推荐引擎的计算量是相当大的，要实现实时的推荐务必需要建立一个推荐模型，关于推荐模型的建立方式可以分为以下几种：

**基于物品和用户本身的**，这种推荐引擎将每个用户和每个物品都当作独立的实体，预测每个用户对于每个物品的喜好程度，这些信息往往是用一个二维矩阵描述的。由于用户感兴趣的物品远远小于总物品的数目，这样的模型导致大量的数据空置，即我们得到的二维矩阵往往是一个很大的稀疏矩阵。同时为了减小计算量，我们可以对物品和用户进行聚类， 然后记录和计算一类用户对一类物品的喜好程度，但这样的模型又会在推荐的准确性上有损失。

**基于关联规则的推荐**（Rule-basedRecommendation）：关联规则的挖掘已经是数据挖掘中的一个经典的问题，主要是挖掘一些数据的依赖关系，典型的场景就是“购物篮问题”，通过关联规则的挖掘，我们可以找到哪些物品经常被同时购买，或者用户购买了一些物品后通常会购买哪些其他的物品，当我们挖掘出这些关联规则之后，我们可以基于这些规则给用户进行推荐。

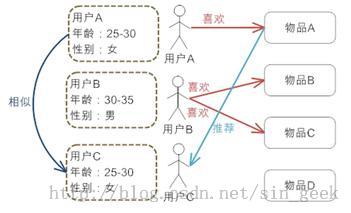
**基于模型的推荐**（Model-basedRecommendation）：这是一个典型的机器学习的问题，可以将已有的用户喜好信息作为训练样本，训练出一个预测用户喜好的模型，这样以后用户在进入系统，可以基于此模型计算推荐。这种方法的问题在于如何将用户实时或者近期的喜好信息反馈给训练好的模型，从而提高推荐的准确度。

其实在现在的推荐系统中，很少有只使用了一个推荐策略的推荐引擎，一般都是在不同的场景下使用不同的推荐策略从而达到最好的推荐效果，例如 Amazon 的推荐，它将基于用户本身历史购买数据的推荐，和基于用户当前浏览的物品的推荐，以及基于大众喜好的当下比较流行的物品都在不同的区域推荐给用户，让用户可以从全方位的推荐中找到自己真正感兴趣的物品。

# 算法原理

## 1、基于人口统计学的推荐

基于人口统计学的推荐机制（Demographic-based Recommendation）是一种最易于实现的推荐方法，它只是简单的根据系统用户的基本信息发现用户的相关程度，然后将相似用户喜爱的其他物品推荐给当前用户，下图给出了这种推荐的工作原理。



基于人口统计学的推荐机制的工作原理图

从图中可以很清楚的看到，首先，系统对每个用户都有一个用户 Profile 的建模，其中包括用户的基本信息，例如用户的年龄，性别等等；然后，系统会根据用户的 Profile计算用户的相似度，可以看到用户 A 的 Profile 和用户 C 一样，那么系统会认为用户 A 和 C 是相似用户，在推荐引擎中，可以称他们是“邻居”；最后，基于“邻居”用户群的喜好推荐给当前用户一些物品，图中将用户 A 喜欢的物品 A 推荐给用户 C。

这种基于人口统计学的推荐机制的好处在于：

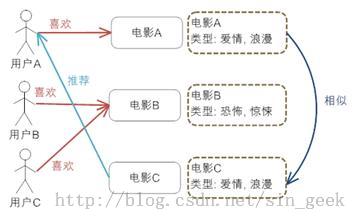
1.      因为不使用当前用户对物品的喜好历史数据，所以对于新用户来讲没有“冷启动（Cold Start）”的问题。

2.      这个方法不依赖于物品本身的数据，所以这个方法在不同物品的领域都可以使用，它是领域独立的（domain-independent）。

那么这个方法的缺点和问题是什么呢？这种基于用户的基本信息对用户进行分类的方法过于粗糙，尤其是对品味要求较高的领域，比如图书，电影和音乐等领域，无法得到很好的推荐效果。可能在一些电子商务的网站中，这个方法可以给出一些简单的推荐。另外一个局限是，这个方法可能涉及到一些与信息发现问题本身无关却比较敏感的信息，比如用户的年龄等，这些用户信息不是很好获取。

## 2、基于内容的推荐

基于内容的推荐是在推荐引擎出现之初应用最为广泛的推荐机制，它的核心思想是根据推荐物品或内容的元数据，发现物品或者内容的相关性，然后基于用户以往的喜好记录，推荐给用户相似的物品。下图给出了基于内容推荐的基本原理。



基于内容推荐机制的基本原理

上图给出了基于内容推荐的一个典型的例子，电影推荐系统，首先我们需要对电影的元数据有一个建模，这里只简单的描述了一下电影的类型；然后通过电影的元数据发现电影间的相似度，因为类型都是“爱情，浪漫”电影 A 和 C 被认为是相似的电影（当然，只根据类型是不够的，要得到更好的推荐，我们还可以考虑电影的导演，演员等等）；最后实现推荐，对于用户 A，他喜欢看电影 A，那么系统就可以给他推荐类似的电影 C。

这种基于内容的推荐机制的好处在于它能很好的建模用户的口味，能提供更加精确的推荐。但它也存在以下几个问题：

1.      需要对物品进行分析和建模，推荐的质量依赖于对物品模型的完整和全面程度。在现在的应用中我们可以观察到关键词和标签（Tag）被认为是描述物品元数据的一种简单有效的方法。

2.      物品相似度的分析仅仅依赖于物品本身的特征，这里没有考虑人对物品的态度。

3.      因为需要基于用户以往的喜好历史做出推荐，所以对于新用户有“冷启动”的问题。

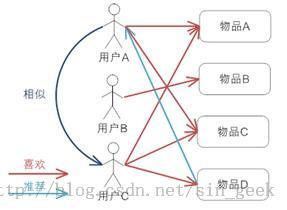
虽然这个方法有很多不足和问题，但他还是成功的应用在一些电影，音乐，图书的社交站点，有些站点还请专业的人员对物品进行基因编码，比如潘多拉，在一份报告中说道，在潘多拉的推荐引擎中，每首歌有超过 100 个元数据特征，包括歌曲的风格，年份，演唱者等等。

## 3、基于协同过滤的推荐

随着 Web2.0 的发展，Web 站点更加提倡用户参与和用户贡献，因此基于协同过滤的推荐机制因运而生。它的原理很简单，就是根据用户对物品或者信息的偏好，发现物品或者内容本身的相关性，或者是发现用户的相关性，然后再基于这些关联性进行推荐。基于协同过滤的推荐可以分为三个子类：基于用户的推荐（User-basedRecommendation），基于项目的推荐（Item-based Recommendation）和基于模型的推荐（Model-based Recommendation）。下面我们一个一个详细的介绍着三种协同过滤的推荐机制。

## 4、基于用户的协同过滤推荐

基于用户的协同过滤推荐的基本原理是，根据所有用户对物品或者信息的偏好，发现与当前用户口味和偏好相似的“邻居”用户群，在一般的应用中是采用计算“K- 邻居”的算法；然后，基于这 K 个邻居的历史偏好信息，为当前用户进行推荐。下图给出了原理图。



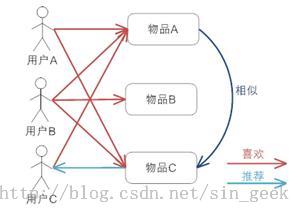
基于用户的协同过滤推荐机制的基本原理

上图示意出基于用户的协同过滤推荐机制的基本原理，假设用户 A 喜欢物品 A，物品 C，用户 B 喜欢物品 B，用户 C 喜欢物品 A ，物品 C 和物品 D；从这些用户的历史喜好信息中，我们可以发现用户 A 和用户 C 的口味和偏好是比较类似的，同时用户 C 还喜欢物品 D，那么我们可以推断用户 A 可能也喜欢物品 D，因此可以将物品 D 推荐给用户 A。基于用户的协同过滤推荐机制和基于人口统计学的推荐机制都是计算用户的相似度，并基于“邻居”用户群计算推荐，但它们所不同的是如何计算用户的相似度，基于人口统计学的机制只考虑用户本身的特征，而基于用户的协同过滤机制可是在用户的历史偏好的数据上计算用户的相似度，它的基本假设是，喜欢类似物品的用户可能有相同或者相似的口味和偏好。

## 5、基于项目的协同过滤推荐

基于项目的协同过滤推荐的基本原理也是类似的，只是说它使用所有用户对物品或者信息的偏好，发现物品和物品之间的相似度，然后根据用户的历史偏好信息，将类似的物品推荐给用户，下图很好的诠释了它的基本原理。

假设用户 A 喜欢物品 A 和物品 C，用户 B 喜欢物品 A，物品 B 和物品 C，用户 C 喜欢物品 A，从这些用户的历史喜好可以分析出物品 A 和物品 C 时比较类似的，喜欢物品 A 的人都喜欢物品 C，基于这个数据可以推断用户 C 很有可能也喜欢物品 C，所以系统会将物品 C 推荐给用户 C。与上面讲的类似，基于项目的协同过滤推荐和基于内容的推荐其实都是基于物品相似度预测推荐，只是相似度计算的方法不一样，前者是从用户历史的偏好推断，而后者是基于物品本身的属性特征信息。



基于项目的协同过滤推荐机制的基本原理

同时协同过滤，在基于用户和基于项目两个策略中应该如何选择呢？其实基于项目的协同过滤推荐机制是 Amazon 在基于用户的机制上改良的一种策略，因为在大部分的 Web 站点中，物品的个数是远远小于用户的数量的，而且物品的个数和相似度相对比较稳定，同时基于项目的机制比基于用户的实时性更好一些。但也不是所有的场景都是这样的情况，可以设想一下在一些新闻推荐系统中，也许物品，也就是新闻的个数可能大于用户的个数，而且新闻的更新程度也有很快，所以它的形似度依然不稳定。所以，其实可以看出，推荐策略的选择其实和具体的应用场景有很大的关系。

## 6、基于模型的协同过滤推荐

基于模型的协同过滤推荐就是基于样本的用户喜好信息，训练一个推荐模型，然后根据实时的用户喜好的信息进行预测，计算推荐。

基于协同过滤的推荐机制是现今应用最为广泛的推荐机制，它有以下几个显著的优点：

1.      它不需要对物品或者用户进行严格的建模，而且不要求物品的描述是机器可理解的，所以这种方法也是领域无关的。

2.      这种方法计算出来的推荐是开放的，可以共用他人的经验，很好的支持用户发现潜在的兴趣偏好

而它也存在以下几个问题：

1.      方法的核心是基于历史数据，所以对新物品和新用户都有“冷启动”的问题。

2.      推荐的效果依赖于用户历史偏好数据的多少和准确性。

3.      在大部分的实现中，用户历史偏好是用稀疏矩阵进行存储的，而稀疏矩阵上的计算有些明显的问题，包括可能少部分人的错误偏好会对推荐的准确度有很大的影响等等。

4.      对于一些特殊品味的用户不能给予很好的推荐。

5.      由于以历史数据为基础，抓取和建模用户的偏好后，很难修改或者根据用户的使用演变，从而导致这个方法不够灵活。

## 7、混合的推荐机制

在现行的 Web 站点上的推荐往往都不是单纯只采用了某一种推荐的机制和策略，他们往往是将多个方法混合在一起，从而达到更好的推荐效果。关于如何组合各个推荐机制，这里讲几种比较流行的组合方法。

1.      **加权的混合**（WeightedHybridization）：用线性公式（linear formula）将几种不同的推荐按照一定权重组合起来，具体权重的值需要在测试数据集上反复实验，从而达到最好的推荐效果。

2.      **切换的混合**（SwitchingHybridization）：前面也讲到，其实对于不同的情况（数据量，系统运行状况，用户和物品的数目等），推荐策略可能有很大的不同，那么切换的混合方式，就是允许在不同的情况下，选择最为合适的推荐机制计算推荐。

3.      **分区的混合**（MixedHybridization）：采用多种推荐机制，并将不同的推荐结果分不同的区显示给用户。其实，Amazon，当当网等很多电子商务网站都是采用这样的方式，用户可以得到很全面的推荐，也更容易找到他们想要的东西。

4.      **分层的混合**（Meta-LevelHybridization）：采用多种推荐机制，并将一个推荐机制的结果作为另一个的输入，从而综合各个推荐机制的优缺点，得到更加准确的推荐。

# 具体模型

## 1 建立矩阵

用户的评分分为显式评分和隐式评分，这次的数据只有隐式评分即，浏览、收藏、加购物车及购买。

表1. 用户行为定义表

|  |  |
| --- | --- |
| 行为名称 | 行为描述 |
| 浏览次数 | 取值为浏览点击数字 |
| 收藏 | 取值为（0,1），1为收藏 |
| 加入购物车 | 取值为（0,1），1为加入 |
| 购买 | 取值为（0,1），1为购买 |

进行数据的处理得到结构化数据

表2. 访问行为结构化数据表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 用户标识 | 商品标识 | 浏览次数 | 收藏 | 加购物车 | 购买 |
| 1 | User1 | Item1 | 1 | 1 | 1 | 0 |
| 2 | User1 | Item2 | 5 | 0 | 1 | 1 |
| 3 | User2 | Item1 | 1 | 1 | 0 | 1 |
| N | UserN | Item1 | 0 | 0 | 0 | 1 |

假设m代表用户数，n代表商品数；代表用户口对商品j的实际评分，1≤i≤m，1≤j≤n；则将用户行为转化为隐式评分的规则如下：

1)如果用户i购买了商品j，则=5；

2)如果用户i将商品j加入购物车，则=4；

3)如果用户i将商品j加人收藏夹，则=3；

4)如果用户i对商品j的浏览次数为2次以上，则=2；若点击次数为1次，则=1；

评分规则可以通过推荐结果的准确程度进行调整；

通常用户对于一件商品会同时做多项操作，例如一个用户先点击商品，加人收藏夹，然后放入购物车，并最终购买，则取其中评分值最高者。

然后可以建立用户-商品评分矩阵



## 2 时间维度

#### 2.1 H.Ebbinghaus遗忘曲线

用户的兴趣是动态变化的，用户近期访问和评分的商品更能反应用户当前的兴趣爱好，更能影响用户当前的购买决策。而早期访问的商品对于用户当前可能产生兴趣的商品的影响作用较小，即用户的访问行为和评分的重要性会随着时间不断衰减。用户的消费行为可以认为是一种心理行为，遵循遗忘曲线的规律。

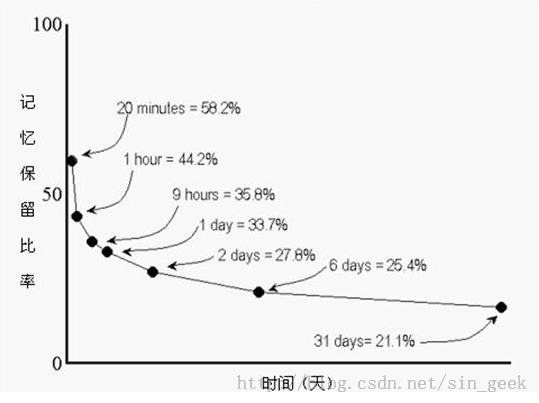


图1 遗忘曲线

表示开始时间，表示对项目的评分时间。该处的评分时间是指用户对商品的综合评分时间，即行为发生的时间。t表示用户对商品的评分时间与有效起始时间的间隔：t=-。

表示用户兴趣随时间变化的指数函数公式(1)如下所示：

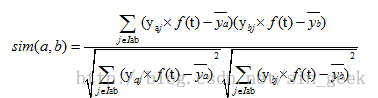
http://img.blog.csdn.net/20150516071514738?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvc2luX2dlZWs=/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/SouthEast (1)

式中，权重λ属于(0，1)，可以根据推荐结果的准确性动态调整。λ越大，表示兴趣随时间衰减越快，反之则越慢。

### 2.2 推荐过程

#### 2.2.1 基于用户的协同过滤

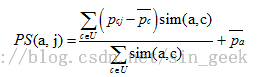
步骤1：利用改进的Pearson相关系数公式计算两个用户之间的相似性，公式如下：

(2)

式中：yaj,ybj分别表示用户a和用户b对商品j的评分，Iab表示用户a和用户b共同评分过的项目集合，f(t)为遗忘函数，http://img.blog.csdn.net/20150516072234506?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvc2luX2dlZWs=/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/SouthEast表示用户a评分过的商品集合的平均得分，http://img.blog.csdn.net/20150516072215689?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvc2luX2dlZWs=/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/SouthEast表示用户b评分过的商品集合的平均得分。

步骤2：将和用户a相似度最高的前k个用户作为它的最近邻居集合U。

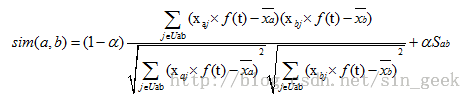
步骤3：综合邻居用户对商品j的评价并预测用户a对商品j的评分。假设c代表邻居用户，PS(a，j)代表目标用户的预测评分，则预测评分的公式如下：

 (3)

步骤4：将预测评分最高的前n个商品作为系统推荐的商品。

#### 2.2.2 基于商品的协同过滤

步骤1：利用改进的Pearson相关系数公式计算两个商品之间的相似性，可以加上项目类别属性，公式如下：

  (4)

式中：，分别表示商品a和商品b对商品j的评分，表示商品a和商品b有共同评分的用户集合，为遗忘函数，，分别表示商品a和商品b在平均得分。表示商品a和商品b的类别相似度，这里取值0或1。是平衡参数，可先取0.5。

步骤2：将和商品a相似度最高的前k个作为它的最近邻集合I。

步骤3：对目标用户未评分的商品根据公式（4）进行预测评分，从大到小排序，取N个值所对应的项目进行推荐。

#### 2.2.3 混合加权

把基于用户的推荐与基于商品的推荐，再考虑地理位置的因素，综合加权。

http://img.blog.csdn.net/20150516072440551?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvc2luX2dlZWs=/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/SouthEast

至于空间地理位置直接单独聚类，然后加权进去。

——————————————————————————————————————————————————————————————

参考资料：

[1]美团推荐算法实践 http://tech.meituan.com/mt-recommend-practice.html

[2]非常好的协同过滤的入门文章http://www.cnblogs.com/wentingtu/archive/2011/12/16/2289926.html

[3]张红霞, 杨渊, 郎维. 基于客户行为和兴趣变化的电子商务推荐系统[J]. 宝鸡文理学院学报：自然科学版, 2012, 32(2):52-56. DOI:10.3969/j.issn.1007-1261.2012.02.011.

[4]韦素云, 业宁, 杨旭兵. 结合项目类别和动态时间加权的协同过滤算法[J]. 计算机工程, 2014, 40(6):206-210. DOI:10.3969/j.issn.1000-3428.2014.06.044.

[5]朱彦松, 窦桂琴. 综合项目权值分配与时间相关的协同过滤模型[J]. 计算机工程与科学, 2014, 36(11). DOI:10.3969/j.issn.1007-130X.2014.11.030.

[关于2015阿里移动推荐算法大赛的总结（三）——机器学习](http://blog.csdn.net/sin_geek/article/details/45787309)

**后来我们回归到正途上，虽然我们也想用深度学习的方法，但是毕竟还是菜鸟的水平，所以把目标定在能用机器学习跑通一遍，顺带熟悉一下各种机器学习算法的实际应用。但是最后的最后我们只用了LR，然后就受打击了。哈哈~**

想用机器学习的方法，那么思路其实也很明确，问题是那一天用户是买还是不买，那么可以看成是二分法。通过用户行为方式来判断是否会购买。就是把用户行为等抽象成特征向量得到训练集，扔进机器学习算法训练出一个模型，然后用测试集预测出结果。虽然机器学习的算法有不少，而且需要不断的调整参数和优化，但是我认为特征向量的选取才是最关键的，很显然我们到最后也没搞好~~~

下面详细整理总结一下~~~   
首先，肯定要进行数据预处理，去除噪声数据（如非正常流量，爬虫或者机器行为产生的数据），数据时间刚好是双11刚过，双12在其中，所以也应该考虑到购物节对用户购买行为的影响。

关于特征向量的选取，我用了下面这种主观上可以接受的想法~   
数据集是一个月的用户行为，那么我直接把数据分成4周，用前3周作为训练集，最后1周作为测试集。用每周过后的1~2天的购买行为当作标签，给这一周的行为统计向量打标签。

进行训练的时候大体思想如下：

统计表：

| **用户-商品** | **浏览** | **收藏** | **加购物车** | **购买** | **最终是否购买（标注）** |
| --- | --- | --- | --- | --- | --- |
| user1-item1 | 4 | 1 | 1 | 1 | 1 |
| user2-item1 | 4 | 1 | 1 | 1 | 0 |
| user3-item4 | 4 | 1 | 1 | 1 | 1 |
| user2-item1 | 7 | 0 | 1 | 1 | 0 |
| user7-item8 | 1 | 0 | 1 | 0 | 1 |

输入表：

| **标签** | **特征1（浏览）** | **特征2（收藏）** | **特征3（加购）** | **特征4（购买）** |
| --- | --- | --- | --- | --- |
| 1 | 4 | 1 | 1 | 1 |
| 0 | 4 | 1 | 1 | 1 |
| 1 | 4 | 1 | 1 | 1 |
| 0 | 7 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 |

训练完模型内部可能：

| **特征1（浏览）** | **特征2（收藏）** | **特征3（加购）** | **特征4（购买）** | **会输出结果** |
| --- | --- | --- | --- | --- |
| 4 | 1 | 1 | 1 | 1 |
| 7 | 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |

进行预测的时候大体思想如下：

统计表：

| **用户-商品** | **浏览** | **收藏** | **加购物车** | **购买** |
| --- | --- | --- | --- | --- |
| user10-item41 | 4 | 1 | 1 | 1 |
| user12-item51 | 4 | 1 | 1 | 1 |
| user13-item24 | 4 | 1 | 1 | 1 |
| user12-item71 | 7 | 0 | 1 | 1 |
| user17-item98 | 1 | 0 | 1 | 0 |

输入表：

| **浏览** | **收藏** | **加购物车** | **购买** |
| --- | --- | --- | --- |
| 4 | 1 | 1 | 1 |
| 4 | 1 | 1 | 1 |
| 4 | 1 | 1 | 1 |
| 7 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 |

输出表：

| **浏览** | **收藏** | **加购物车** | **购买** | **输出结果** |
| --- | --- | --- | --- | --- |
| 4 | 1 | 1 | 1 | 1 |
| 4 | 1 | 1 | 1 | 1 |
| 4 | 1 | 1 | 1 | 1 |
| 7 | 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |

然后我们寻找预测为购买的特征对应的用户-商品对。

输出表：

| **浏览** | **收藏** | **加购物车** | **购买** | **输出结果** |
| --- | --- | --- | --- | --- |
| 4 | 1 | 1 | 1 | user10-item41 |
| 4 | 1 | 1 | 1 | user12-item51 |
| 4 | 1 | 1 | 1 | user13-item24 |
| 1 | 0 | 1 | 0 | user17-item98 |

注意点：   
（1）在选取样本集的时候可以是多段累加数据。   
（2）后期可以对不同的类型的物品建立不同的训练模型，比如电影票饭票一个模型，淘宝物品一个模型，因为不同类别购买行为规律不同。根据类别分类，分别输入进不同模型输出。   
（3）根据地理位置用一个简单的推荐算法进行加权融合。   
（4）考虑行为中的时间因素，如在一个周期对某商品内浏览4次，在第一天浏览4次和最后一天浏览4次购买的可能性也不同。就是引入时间对行为加权，可能本来浏览4次，最后加权出来浏览是2.33次。最后使用的是总结（二）里的遗忘曲线加权的。

结果很不尽人意，甚至不如直接取最后一天加购物车当作第二天会购买的用户商品对的评分高。但是起码这个从头到尾跑了一遍，在这个过程中，学到了好多，如思考模型算法，学习使用Python，对数据的划分处理以及对机器学习的使用调参。希望自己不断的在进步，不要求自己有多牛，但要比之前的自己牛~哈哈

[事非经过不知难-阿里大数据推荐大赛总结](http://blog.csdn.net/litoupu/article/details/38350157)

**1、成绩单**

大赛地址：<http://102.alibaba.com/competition/addDiscovery/gameTopic.htm>

最终成绩：http://102.alibaba.com/competition/addDiscovery/totalRank.htm

参赛队伍数：7276

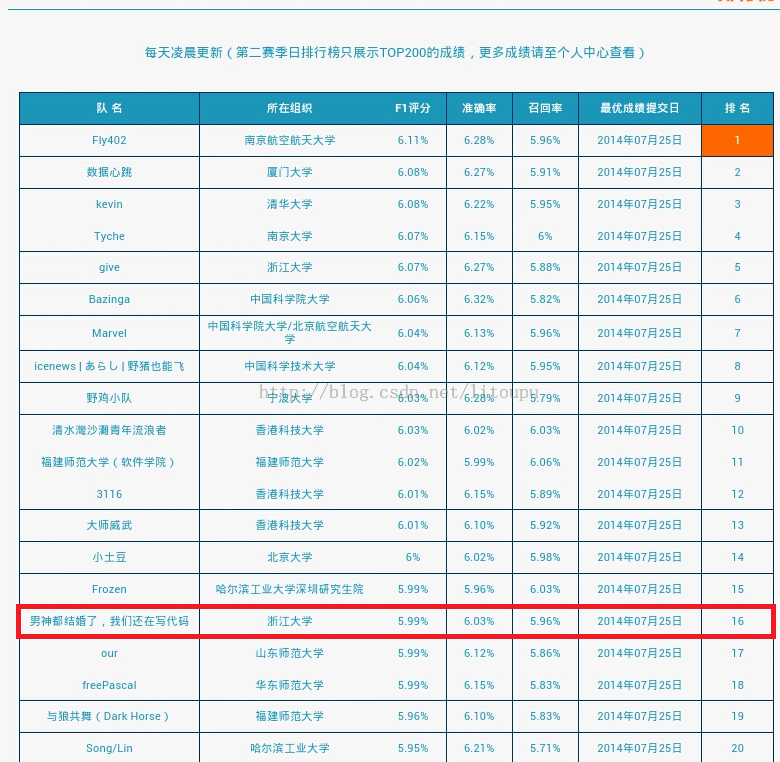
         S1：                  成绩F1|排名 **7.07%|205**

   S2换数据前：           成绩F1|排名  **5.98%|21**

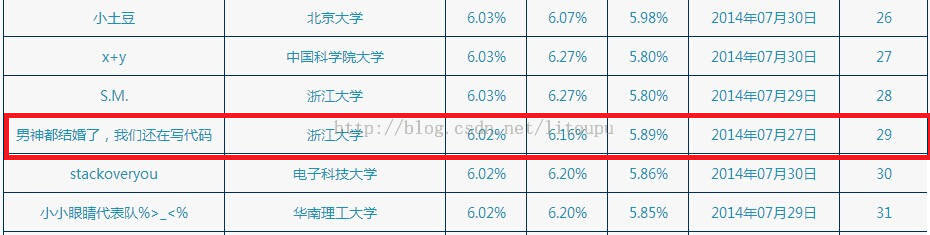
S2换数据后（最终）： 成绩F1|排名  **6.02%|29**

 **参赛前的目标就是Top50，也算是圆满完成目标了,现在只求这次比赛对于明年阿里校招的奖励机制给力一些！**

中间曾多次闯入首页Top20，截图留念：



 S2**最终排名**

****

**2、心路历程**

         首先谈谈报名参赛初衷：

     1).主要想锻炼自己，想要积累一些推荐这个领域的实战经验，再加上刚好去年刚看了一些传统的经典的推荐算法的书籍，去年参与实验室的一个项目就是和推荐相关；

     2).对阿里的云平台odps比较感兴趣，想体验一下，真实地触摸大数据体验大数据；

     3).当时报名的时候希望能进Top50，能拿到明年阿里的校招的绿色通道.

     第一赛季，和实验室同项目组的谷神和ycc一组，当时组队的时候对我们的期许都很高，但是因为各种各样的原因，三个人的参与程度都不是很高，交流也不多，S1的成绩并不是好，记得当时用了我用了师兄论文中的一个Baseline 的推荐算法，Top Sensitive Pagerank来实现，发现效果很差，后来和其他同学交流得知用规则就可以做得很高，后来基本就光想规则，没有思考模型思考算法层面的东西，所以S1感觉收获不大。最后一天，我们的队伍500多名，即将面临淘汰，因为这个比赛最吸引我的还是ODPS（阿里的云平台）的使用，就想继续玩玩，就抱大腿临时加入到小鱼他们队，才以至于有这之后的三个月S2的故事。

    S2 队友小鱼很给力，又是个可爱漂亮的妹纸，她的总结在这里，诸多细节我就不再赘述。

<http://blog.csdn.net/muyuxiaozi_2013/article/details/38302489>

**3、4个月比赛中我做了些什么**

          1、平均每天的参与程度时间为4-5h左右，S2三个月共花了400+ h

          2、了解和熟悉LR、RF、GBRT等算法的一些原理，以及这些算法的Python接口在ODPS中的调用  
          3、知道了机器学习在实战中的完整过程，从数据集Train和Test集的划分，过拟合，模型的选择，特征的抽取，正负样本的处理，采样方式（向上采样、向下采样），各种调参，特征的处理，y值处理，融合的方式都有了一个不一样的理解。  
          4、作为数据库方向的学生，说实话有些许惭愧，之前对普通关系型数据库最基本的SQL操作都不是很熟练，通过这个比赛对SQL已经基本掌握（包括Join，Union，Group By，Order By等）  
          5、重点了解了分布式odps的基本原理（其实就是Hadoop），对MapRedece有了更加深入的了解（包括任务分解合并的过程等），并且自己能够够独立地编写MapReduce程序来提取特征，通过这个比赛完成不重复代码的编写5000行+.

**4、分享一些干货**

**A.整个模型建模过程**，这里用了很早之前的LR模型（特征也是最普通的SQL暴力提取）,整个流程其实是一样的

     阿里大数据的任务就是根据用户4个月在天猫的行为日志，建立用户的品牌偏好，并预测他们在将来一个月内对品牌下商品的购买行为。考虑到S1过程中我们初步判断点击的数据只有最近的半个月影响较大，故我们最初的LR模型就采用半个月的数据作为一个时间周期，这里所说的对点击率做LR模型是指逻辑回归，也就是推荐系统中常见的分类问题，即9月份买或者不买的问题，买用1表示，不买用0表示。

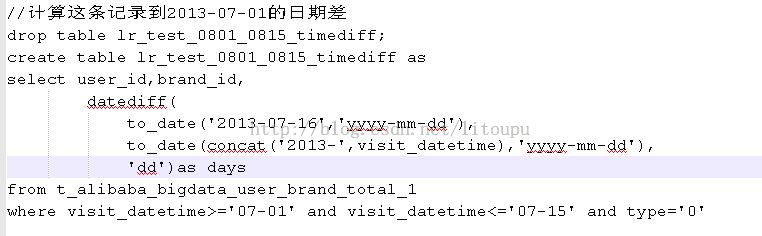
**TrainData准备**：7.01-7.15(A)的点击数据作为feature，7.16-7.31(B)购买与否数据作为y，同时控制正负样本的比例，抽取A中的点击同时B中购买，同时随机抽样A中的点击但是B中未购买的样本(这种样本数量很多，需要对之随机抽样，可以使用Xlab中的随机抽样功能)，控制正负样本的比例。将A的点击+B中的购买与否y构成完整的TrainData。

     在构造TrainData的过程中，需要考虑时间衰减的因素，我们引入衰减因子，sum(1/(1+aipha\*Days)).

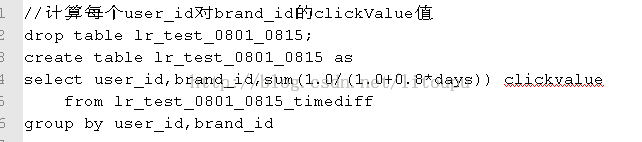
**TestData准备**：8.01-8.15(C)之间所有购买的数据。

在虚拟机中，具体过程如下：

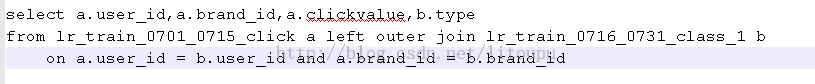
     (1)计算A中所有点击visit\_datetime距7.16时间间隔，用如下SQL代码执行：



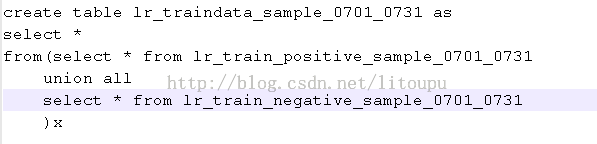
     (2)考虑时间衰减因子，group by同一个user\_id对brand\_id的总的clickValue，这里的alpha初步设置为0.8



     (3)通过步骤1和步骤2已经生成了triandata中的点击数据，现在生成相对应的y为1数据样本：



抽取所有y=1的样本，利用Xlab填充y为NULL的样本的y值为0，随机采样y=0的样本，控制和y=1的比例接近（正负样本比例接近），将正负样本进行合并，最终生成完整的TrainData(user\_id,brand\_id,clickvalue,y)



     (4)利用Xlab工具对完整的TrainData进行逻辑训练，生成训练的模型表。

     (5)针对模型，在TestData(C)上上进行预测，产生训练结果表（包括调参）。

     (6)对结果表按照提交格式进行提交数据。

**B.通用多模型Ensemble过程**

     在推荐系统中，比较通用的多模型Ensemble过程应该是这样的，我们以阿里大数据推荐竞赛来具体理解一下。

将数据的前3个月的交互数据（其Label为第4个月的购买情况）划分为80%的数据作为Train，20%的数据作为Test。

1、80%的Train数据训练出Model\_0，Model\_1，Model\_2...Model\_i.

2、20%的Test中的前三个月交互数据（不包括Label）作为测试数据输入Train出来的各个Model\_i,分别得到每一个Model\_i的(u,b,probability\_i).

3、将2中得到的test结果组成一个大的特征，(u,b,probability\_0,probability\_2...probability\_i.label)，其中Label来自Test中原本存在的Label，将这个数据重新训练(LR,RF,GBDT)，产生**EnsembleModel.**

4、将后三个月的数据用Model\_0,Model\_1..训练得到相应的probability，将这些结果拼装成(u,b,probability\_0,probability\_2...probability\_i)的形式，再用EnsembleModel再训练一次，得到最后的Probability，最后取TOPN进行提交

**注意**：在线上提交阶段，真正的是需要用100%的训练数据进行Model训练，但是考虑到比赛时间的限制，80%的数据进行训练的Model\_i 可以作为线上直接使用。

**C.在ODPS中运用SVD分解思路**(虽然对于这次大赛效果很不好，最后也没有用上任何FM，但是对于我理解SVD还是有一些帮助，就随便写一点了)

     由于odps中表格的列最大为1024，所以我们选取打分最高的top1000 brand作为svd分解估计的brand。

线下测试：

1、把gbrt结果y\_val作为开始u和b之间的pro

2、svd分解，选出未交互的ub对，然后加上u特征和b特征，进行gbrt或者rf训练

3、得到数量较小的但是准确率较高未交互的ub属性。

**SVD基本思路：**

构建userNum\*brandNum矩阵matrix，矩阵的初始值matrix(i,j)表示用户i对品牌j的相关度，值越大越表示该用户与品牌的相关度越大

[u,s,v]=matrix.svd(k)

取奇异值的前k个

讲分解的结果重新乘回去，得到新的矩阵，即可以得到某用户与brand未交互的相关度

result=u\*s\*v.transpose();

**5、分享一些体会与收获**

     1、这比赛对我最大的收获应该是从比赛中重新找回自信吧，从宁大保研来浙大之后，可能没有完全适应浙大竞争激烈的环境，不管是在实验室科研上，还是生活中的种种都不是特别顺利，开始慢慢地怀疑自己，变得不像本科的时候那样的自信，通过比赛，重拾自信，重新上路。

     2、参加这种比赛，选择靠谱队友很重要，队友之间的配合非常重要，对于我们最后能够取得Top30的成绩，我觉得我和队友小鱼之间的配合做的还不错，两个人的参与热情高，愿意花时间，小鱼主要负责前期的建模过程，利用SQL提取特征，以及后期的模型融合以及上交结果，而我主要负责在模型初期和小鱼讨论特征以及建模整个过程，最重要的工作主要是利用ODPS上的mapreduce代码的编写用来提取我们想要的特征。

     3、信息的获取很重要，我们承认，至少是我承认，我们有没有非常厉害的创造创新的能力，我们的思路主要来自组内的交流以及和其他队伍的交流，我觉得我们这一点一开始就非常重视也做的还不错，但也有像实验室另一个同学所说的，这可能是不能超越其他强队的原因，我觉得他说的也非常有道理。而我更加认为偏向于当我们没有达到某一种高度成绩没有到达一定的程度就去做创新的东西很难，就比如这个比赛中应该是夺冠的Marvel，他做了未交互的ub部分预测，并且取得了意想不到的效果，但是你要想想，有多少支队伍尝试做过这部分工作，至少最终的top50都有做过吧，并且花的时间肯定都不少，都没有取得想要的好的结果。

     4、比赛过程中代码的自动化非常重要，前期花了不少时间花在一些无谓的地方，现在回过头来想想可能觉得有些可惜，不过也有可能就是摸着石头过河吧。

     5、比赛的笔记记录非常重要，因为这次比赛时间非常长，数据有非常多的版本，所以需要记录每一个version的算法、特征提交后的成绩等，要不然到后期根本搞不清楚。

     6、关于比赛的一点小遗憾就是，最后的模型融合可能没做好，原因之一是前期花的时间也不是很多，没有划分Train和Test再次训练模型之间参数来Ensemble，现在想想有些许小遗憾。

     7、这三个月，和队友，瞒着导师偷偷做阿里的大数据比赛，异常地艰辛，不敢在实验室激烈的讨论，这三个月生物钟完全乱掉，每天凌晨4点多钟起来看成绩，和小鱼互发短信简单分析成绩没提高的原因，抑或是分享成绩提高名次进步带来的快乐，特别是最后换数据之后，因为集群很卡，为了避开使用的高峰，凌晨起来跑程序，记得7.24-25那几天，晚上就只睡3-4个小时，比赛的激烈程度真是只有经历过的人才懂，这个比赛能够坚持下来并且取得不错的成绩真的很辛苦，从S2开始到结束起码创建了1000+张的表格了吧，我们每一份特征都用一个新的Label标识，起码20+吧，我想如果这三个多月没有小鱼的鼓励和帮助，我一个人是肯定坚持不下来的，这三个多月的付出，其中的酸甜苦辣真的只有自己经历过了才知道，现在想想一切都是那么值得。

**特别感谢**：本科同学宁大的清风，北理的Jark，浙大的算法大神黎老师，还有大葱、宣、主公、小猫等。

当然最最最最要感谢的是我的亲爱的给力靠谱队友**小鱼**~

**6、推荐一些和比赛相关不错的博文**

<http://wuchong.me/categories/%E6%8E%A8%E8%8D%90%E7%B3%BB%E7%BB%9F/>

http://armysheng.farbox.com/post/2014-04-21#toc\_1  
http://oilbeater.com/  
http://bbs.aliyun.com/read/155044.html?spm=5176.7189909.0.0.c7Tm4P

# [2015阿里移动推荐算法比赛第一赛季总结](http://www.letiantian.me/2015-05-17-alibaba-mobile-recommend-competition-summary/)

Tags:[机器学习](http://www.letiantian.me/tag/ji-qi-xue-xi/)

2015-05-17

第一赛季是在四月底结束，分数还凑合，过了9.00%。本文只是一篇总结性的文章。

发现的问题：   
1、还是学的不怎么样   
2、了解的不够深入，实战经验少   
3、PRML和NG的视频都没看，不算好好学过机器学习   
4、论文看得太少   
5、没人指点   
6、机器设备太差   
7、比赛参加的有点晚，中间还出去溜达了几天   
8、单人作战很酸爽～

## 比赛题目

官方提供了两个文本文件，第一个是tianchi\_mobile\_recommend\_train\_user.csv：

user\_id,item\_id,behavior\_type,user\_geohash,item\_category,time

99512554,37320317,3,94gn6nd,9232,2014-11-26 20

9909811,266982489,1,,3475,2014-12-02 23

......

每一行代表了用户user\_id对属于分类cat\_id的物品item\_id在time这个时间于地点user\_geohash发生了交互，交互类型是behavior\_type。behavior\_type包括浏览、收藏、加购物车、购买，对应取值分别是1、2、3、4。tianchi\_mobile\_recommend\_train\_user.csv中的数据约有1200万行。

一共有31天的交互数据，最后要预测地32天有哪些user会购买哪些item。 举办方对你提交的数据计算F1，并进行排名。

然而，还有一个商品子集的文件tianchi\_mobile\_recommend\_train\_item.csv：

item\_id,item\_geohash,item\_category

327414838,,11991

169831798,,7876

320523991,,3370

......

这个文件约有44万条数据，是tianchi\_mobile\_recommend\_train\_user.csv中出现的item的一个子集。官方的第32天的会发生购买的user, item中的item都是这个文件中的item，所以我们预测的user, item需要根据这个文件来过滤掉一些结果。

## 基本思路

这个预测，可以考虑预测**发生过交互行为**的user, item会不会在第32天购买，也可以预测**没有发生过交互行为**的user, item会不会在第32天购买。笔者这一次只考虑了**发生过交互行为**的user, item。我们要有一份自己的训练集和测试集来评估自己的模型（算法），当觉得自己的模型比较好了，就可以拿着模型去预测了。

## 清洗数据

比如去除总的交互次数很低（低于2或者低于3或者...）的user, item，去除最后10天没有发生交互的user, item。还可以继续删除，比如双12的所有数据。不过删多了可不一定好。

## 特征工程

一个简单算法在精心选择的特征上的效果比一个漂亮的算法在较差的特征上的效果还要好。

如何构造特征很重要，特征构造得好，同一算法下准确率也可以提升很多。个人认为构造特征是这次比赛中最重要的任务。

以[Iris数据集](http://zh.wikipedia.org/wiki/%E5%AE%89%E5%BE%B7%E6%A3%AE%E9%B8%A2%E5%B0%BE%E8%8A%B1%E5%8D%89%E6%95%B0%E6%8D%AE%E9%9B%86)为例，其默认提供了4个特征：花萼和花瓣的长度和宽度。对此我们可以构造更多的特征，例如花萼宽度/花萼长度、花萼宽度/花瓣长度、花萼宽度×花萼长度等等，只要觉得合理，就可以构造。

另外，特征的优劣会影响到一些机器学习算法的效果，或者特征太多机器会跑不动。筛选特征可能会成为一个很重要的步骤。假设现在有100个有标号的样本，每个样本10个特征，可以通过下面几种方法：

**方法1：** 拿出两个特征对应的两个100维的向量，使用pearson相关系数、归一化互信息量等方法计算两个特征的相关性，如果相关性很高，可以考虑删除其中一个特征。

**方法2：** 利用决策树的内置的特征筛选方法筛选好的特征

**方法3：**特征递归消除。可以参考<http://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFE.html>。

scikit-learn库提供了多个特征选择的方法，[点这里](http://scikit-learn.org/stable/modules/feature_selection.html)。

也可以考虑降维。[点这里](http://scikit-learn.org/stable/modules/decomposition.html)。

**那么，一个user, item可以构造什么特征呢？**

以前30天的所有数据为例（第31天有没有购买当作target）： 最后一天user有没有购买item、最后的4天user有没有购买item、最后的4天user有没有点击item、最后的4天user对item的点击量、最后的4到8天user对item的点击量，最后的4天user的点击量，最后的4天item的的点击量，最后的4天item的的购买量除最后的4天item的的点击量...... **去年比赛的PPT中都讲到了这些**。 另外,geohash也可以拿来生成特征，例如判断用户的最新位置和item最近的位置有多近；cat\_id也可以考虑一下。再开个脑洞，将user, item的构造的某些特征进行聚类，样本所属的簇、样本离某个簇心的距离等也可以当作该样本的新特征。

## 训练集和测试集的构造

基于上面的内容，现在可以构造训练集和测试集了。

* 新数据集1： 将1-27天的数据拿出来做特征，第28天作为target。 -> 用于训练
* 新数据集2： 将2-28天的数据拿出来做特征，第29天作为target。 -> 用于训练
* 新数据集3： 将3-29天的数据拿出来做特征，第30天作为target。 -> 用于训练
* 新数据集4： 将4-30天的数据拿出来做特征，第31天作为target。 -> 用于测试
* 新数据集5： 将5-31天的数据拿出来做特征。 -> 用于最终的预测。

**正负样本比例问题：** 对于上面生成的一个数据集，正例（target是购买）的数量只是几千，而负例上百万。只能说太不平衡了，不适合做训练集。可以这样做：将多个数据集的正例放在一起以增加正例数量；对负例进行随机采样。可以考虑生成多个训练集来观察模型的稳定性。

**如何评估模型：** 用新数据集4做测试，其实只需要把新数据集4其中的正例拿出来，计算F1即可。如果这个F1做的比较好了，可以认为提交结果的F1也会很好。

## 分类还是回归？

很明显，这是一个二分类问题。不过如果只看做二分类，会出一些问题。举办方那边有约500条数据（从第32天的实际购买user, item抽取出来的）来计算参赛者提交数据的F1，如果你只预测了100个user, item 或者预测了2000条user, item，这些数据并不适合提交上去，提交的比较好的数据量是400~1200之间。

利用回归来解决是个不错的思路。基于回归的预测结果可以简单看成user会购买item的概率，概率越高，购买的可能性越大。通过设置一个阈值就可以得到任意数量的提交结果了。

## 机器学习库、机器学习方法

网上能找到很多机器学习库，例如scikit-learn、Spark MLlib、Apache Mahout、R等。

上两个网址：   
[InfoQ：机器学习的11个开源项目](http://www.infoq.com/cn/news/2014/12/11-machine-learning-project)   
[知乎：请问学习机器学习有哪些好工具推荐呢？](http://www.zhihu.com/question/20472776)

scikit-learn支持很多机器学习方法，列举一二：

[Logistic Regression](http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html)：分类器，predict\_proba()函数可以当作回归来使用。

[决策树 - 分类](http://scikit-learn.org/stable/modules/tree.html#classification)

[决策树 - 回归](http://scikit-learn.org/stable/modules/tree.html#regression)

[随机森林 - 分类](http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html)

[随机森林 - 回归](http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html)

[GBDT - 分类](http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html)

[GBDT - 回归](http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingRegressor.html)

Spark 的机器学习库[MLlib](http://spark.apache.org/mllib/)提供的方法也不少，可以进入[Machine Learning Library (MLlib) Guide](http://spark.apache.org/docs/latest/mllib-guide.html)了解一下。

只用1个算法去预测的队伍应该挺多，也可以考虑多个经典机器学习算法融合到一起（[Ensemble learning](http://en.wikipedia.org/wiki/Ensemble_learning)）。

**参数调优** 每个模型都有很多参数，参数的选择也会影响到最后的效果。思路：

1、手动调参数；   
2、枚举一些参数值，一个个测试；   
3、遗传算法、粒子群算法等。

**方差和偏差问题**

训练后的模型测试的效果很差，叫做高偏差，这往往意味着模型太简单。解决方法是：增加更多特征、使用别的模型、使用更复杂的模型等。

同一模型，用随机采样的多个数据集训练并测试的多个结果相差很大，叫做高方差，这往往意味着模型太复杂，导致了过拟合。解决方法是：使用更多数据、降低模型复杂度等。

## 单机还是集群？

这次比赛数据量偏大，要不加内存，要不用集群。

## 提交多少数据？

同一模型、同一训练集得到的预测结果，通过阈值可以得到不同数量的结果，提交1000条结果和提交600条结果的F1分数可能差距很大，提交600条结果和提交560条结果的F1分数也可能差距很大。如果模型做得足够好的话，提交400～1000条数据都能得到很好的F1分数；如果模型方面到瓶颈了，就需要好好考虑提交多少条数据比较好了。

## 其他工具

使用MapReduce对数据做预处理。   
存储基本特征、中间结果等考虑使用数据库（MySQL或者MongoDB）。

## 如何查找作弊现象

想了想，可以是下面的一些方法：   
1、两个队伍提交的数据完全相同（先排序，再求编辑距离）   
2、两个队伍提交时使用的IP相同，但是不同的队伍可能用同一个代理   
3、一个浏览器提交了多个队伍的数据（在浏览器中加入cookie）   
4、实名认证   
5、提交代码，代码查重

## 总结

收获很多，自己在机器学习这方面还有很大进步空间。

学习并使用了Spark的MLlib。