商场中精确定位用户所在店铺

一、 研究动机

如今，商场规模越来越大。商场规模的扩大在某种程度上带来了两个方面的问题。一方面，对于用户来说，在一个店铺密集型的商场里如何找到自己的目标变得越来越困难。用户进入店铺以后，对店家的促销信息、新品推介等都不了解，在商场里购物的效率不高。另一方面，商场不熟悉每天每间店铺的客流量，不能做出最贴切的营销策略。

因此，能够精确判断用户所在店铺变得尤为重要。精确定位用户所在的店铺也是当前研究的一个难题。在真实生活中，当用户在商场环境中打开手机的时候，存在定位信号不准、环境信息不全、店铺信息缺失、不同店铺空间距离太近等挑战。GPS 在室外定位中有着广泛的应用，但是在室内环境里，GPS 信号会受到墙体等障碍物的阻隔而失效。在这种情况下，需要使用 WIFI 环境信息、店铺信息等其他数据对用户所在店铺进行判断。

本项目目标为：在已知商场店铺相关信息、用户所处的 GPS 位置和周围的 WIFI 等环境信息的情况下，精确判断用户当前所在的店铺。

本项目的应用场景如下：当用户走入商场的某家餐厅时，手机自动弹出该餐厅的优惠券；当用户走入商场服装店时，手机可以自动推荐这家店铺里用户喜欢的衣服；当用户路过商场的一家珠宝店时，手机可以自动提示用户心仪了很久的一款钻戒已经有货。

本项目的研究意义在于：精确定位用户所在店铺，以在正确的时间、正确的地点给用户最有效的服务。

二、 问题描述

训练集大小：596967（2017-08-20 14:00 之前的行为数据）

测试集大小：451608（2017-08-20 14:00 及之后的行为数据）

输入：row_id, user_id, mall_id, time_stamp, longitude, latitude, wifi_infos

输出：shop_id

三、 特征提取方法

因为每一个测试数据的 mall_id 是知道的，所以特征提取针对于某个 mall 来做。

1. 提取所有 WIFI 信息

训练数据中，每个行为数据都有一个 WIFI 信息表，记录了这个行为记录发生时，有什么 WIFI，这些 WIFI 的信号强度，以及用户是否有连接这个 WIFI。

我们需要统计某个 mall 存在多少不同的 WIFI，然后用这些 WIFI 的强度作为每个行为数据的特征，这个 mall 的每个 WIFI 作为一个特征列。

假设某个 mall 共有 r 行行为数据，这个 mall 的所有店铺共有 n 个不同的 WIFI：

则提取所有 WIFI 信息结果如下表所示：

Row_id	WIFI_1	WIFI_2	WIFI_3	WIFI_4	…	WIFI_n
1				NaN	…	
2		NaN	NaN		…	
3			NaN		…	
…	NaN		NaN	NaN	…	
…				NaN	…	
…					…	NaN
r			NaN	NaN	…	

其中，NaN 表示当前行为数据没有这个 WIFI。

2. 提取稳定 WIFI 信息（降维）

这个时候，会发现，对于某个 mall，提取的 WIFI 列数可能有好几千列。WIFI 中有位置比较固定的稳定 WIFI，也有用户自带的移动 WIFI，必须将不稳定的噪音 WIFI 过滤掉。

我们提出了一种提取稳定 WIFI 的方法。这个方法基于这样的思想：如果一个 WIFI 属于用户自带的移动 WIFI，那么它出现在行为数据的次数肯定不多。

因此，我们删除出现次数小于 m（其中 m＜r）的 WIFI 列，例如 WIFI_2、WIFI_4 出现的次数较少，会被过滤掉。经过实验，最后的 m 设为 20。可以把原来好几千列的 WIFI 数据降为几百列。

提取稳定 WIFI 信息结果如下表所示：

Row_id	WIFI_1	WIFI_3	…	WIFI_n
1			…	
2		NaN	…	
3		NaN	…	
…	NaN	NaN	…	
…			…	
…			…	NaN
r		NaN	…	

3. WIFI 信号强度的变换

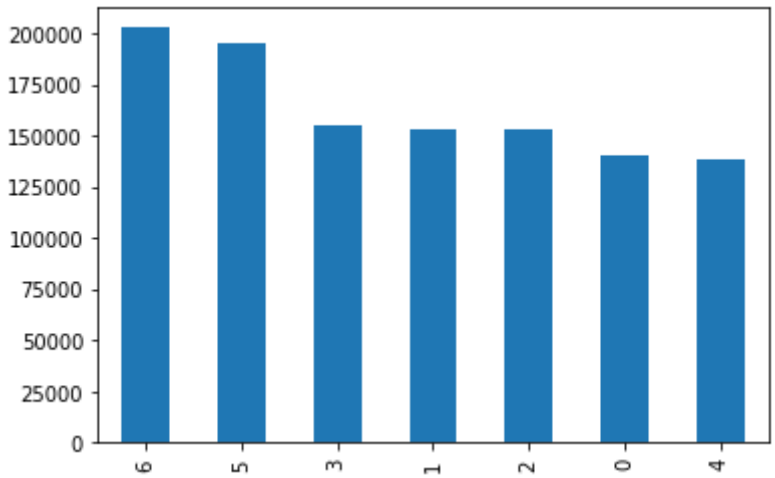
为了便于分析，我们将 WIFI 信号强度变换为 0-100，这样，WIFI 强度为 NaN 的记录就可以填充为 0，表示没有这个 WIFI。

变换方式为：对于某一列 WIFI，记录所有行为数据中该 WIFI 强度的最大值为 max，强度最小值为 min，则对于某一行行为数据，假设其原来的 WIFI 强度为$\text{signal}_{\text{old}}$，变换之后的 WIFI 强度记为$\text{signal}_{\text{new}}$。则变换公式可以表示为：

$$\text{Signal}_{\text{new}} = \frac{100 \times (signal_{old} - min)}{\max - min}$$

4. 提取周末/非周末属性

下图表示所有行为数据中，周一到周日出现的频次直方图。

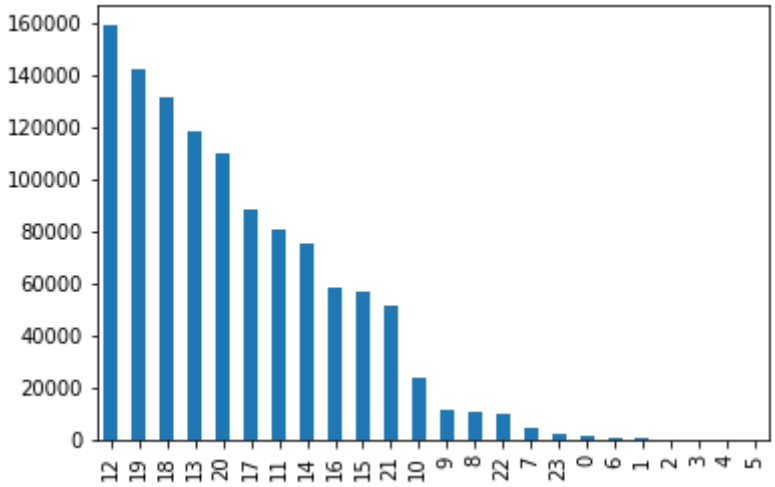


可以看到，周六周日的行为数据比较多，说明人流量较大，这时候的 WIFI 环境更加复

杂。因此，增加 weekend 属性，以区分周末/非周末数据。周六周日的 weekend 属性值为 1，周一到周五的 weekend 属性值为 0。

5. 提取繁忙/非繁忙时间段属性

下图表示所有行为数据中，一天二十四小时，每个小时出现的频次直方图。



可以看到，12 点、19 点、18 点、13 点的行为数据较多，说明人流量大，这时候的 WIFI 环境更加复杂。因此，增加 busy 属性，以区分繁忙/非繁忙时间段的数据。时间段属于 12、19、18、13 的行为数据，busy 属性记录为 1，其他时间的行为数据，busy 属性记录为 0。

6. 最终特征

最后，使用经度、纬度、稳定 WIFI 列、周末/非周末、繁忙/非繁忙时间段这些属性作为特征去训练我们的分类器

四、 算法

1. 随机森林

可以直接使用随机森林进行店铺多分类。

随机森林的基本想是：随机森林是一个包含多个决策树的分类器，随机森林的输出类别由个别树输出的类别的众数而定。之所以叫随机森林，是因为这些决策树的形成采用了随机的方法。

随机森林建立一个决策树的流程如下：

1) 采用有放回的方式，对原始训练进行行采样，得到一个新的训练集，用该训练集训

练一个决策树。在每次决定用哪个特征对数据集进行划分的时候，并不是选所有特征上的划分信息增益最大的那个特征，而是先进行一个列采样，从采样出来的特征里面选最好的那个特征。

2) 对采样之后的数据使用完全分裂的方式建立决策树。完全分裂的意思是：所有样本指向同一个分类或者列采样的特征不足以继续分裂。

2. 逻辑回归

普通的逻辑回归只能针对二分类问题，而我们这个店铺分类是一个多分类问题。因此，我们把我们的多分类问题，转化成多个二分类问题，然后用逻辑回归求解。

当我们把问题看成多分类的时候，对于某个 mall，我们训练得到一个多分类器。当我们把问题看成二分类的时候，对于某个 mall，假设这个 mall 有 k 个店铺（假设店铺 id 为 1~k），则我们可以训练得到 k 个不同的二分类器。

每个二分类器的建立方法为：对于店铺 id 为 c 的店铺建立一个二分类器，则把店铺 id 为 c 的所有行为数据作为正样本，二分类类别标记为 1，其余店铺标记不为 c 的所有行为数据作为负样本，二分类类别标记为 0。

得到 k 个不同的二分类器后，针对一个测试样本，我们需要找到这 k 个分类函数输出值最大的那一个，即为测试样本的标记：

$$\arg\max_{c} h_c(x)\ c = 1, 2, ..., k$$

3. xgboost

Boosting 分类器属于集成学习模型，它基本思想是把成百上千个分类准确率较低的树模型组合起来，成为一个准确率很高的模型。这个模型会不断地迭代，每次迭代就生成一颗新的树。

GBDT(Gradient Boosting Decision Tree) 又叫 MART（Multiple Additive Regression Tree)，是一种迭代的决策树算法，该算法由多棵决策树组成，所有树的结论累加起来做最终答案。GBDT 的核心在于，每一棵树学的是之前所有树结论和的残差，这个残差就是一个加预测值后能得真实值的累加量。与随机森林不同，随机森林采用多数投票输出结果；而 GBDT 则是将所有结果累加起来，或者加权累加起来。

XGBoost 的全称为 eXtreme Gradient Boosting，是 GBDT 的一种高效实现，XGBoost 中的基学习器除了可以是 CART（gbtree）也可以是线性分类器（gblinear）。

XGBoost 对 GBDT 的改进：

1. 避免过拟合

目标函数之外加上了正则化项整体求最优解，用以权衡目标函数的下降和模型的复杂程度，避免过拟合。基学习为 CART 时，正则化项与树的叶子节点的数量 T 和叶子节点的值有关。

2. 二阶的泰勒展开，精度更高

不同于传统的 GBDT 只利用了一阶的导数信息的方式，XGBoost 对损失函数做了二阶的泰勒展开，精度更高。

3. 列抽样（column subsampling）

xgboost 借鉴了随机森林的做法，支持列抽样，不仅能降低过拟合，还能减少计算，这也是 xgboost 异于传统 gbdt 的一个特性。

4. 对缺失值的处理

对于特征的值有缺失的样本，xgboost 可以自动学习出它的分裂方向。

5. xgboost 工具支持并行

xgboost 的并行不是 tree 粒度的并行，xgboost 也是一次迭代完才能进行下一次迭代的（第 t 次迭代的代价函数里包含了前面 t-1 次迭代的预测值）。xgboost 的并行是在特征粒度上的。我们知道，决策树的学习最耗时的一个步骤就是对特征的值进行排序（因为要确定最佳分割点），xgboost 在训练之前，预先对数据进行了排序，然后保存为 block 结构，后面的迭代中重复地使用这个结构，大大减小计算量。这个 block 结构也使得并行成为了可能，在进行节点的分裂时，需要计算每个特征的增益，最终选增益最大的那个特征去做分裂，那么各个特征的增益计算就可以开多线程进行。

6. 可并行的近似直方图算法

树节点在进行分裂时，我们需要计算每个特征的每个分割点对应的增益，即用贪心法枚举所有可能的分割点。当数据无法一次载入内存或者在分布式情况下，贪心算法效率就会变得很低，所以 xgboost 还提出了一种可并行的近似直方图算法，用于高效地生成候选的分割点。

4. Bagging 集成学习

Bagging 算法是机器学习领域的一种集成学习算法，其主要思想是集成多个模型的预测结果，从而提高算法的稳定性，避免过拟合现象的产生。

Bagging 的主要流程：

1) 假设训练集大小为 N

2) Bagging 算法从中均匀、有放回地选出 m 个大小为 n 的子集，作为新的训练集。

3) 在每个训练集上，都可以训练得到一个模型。

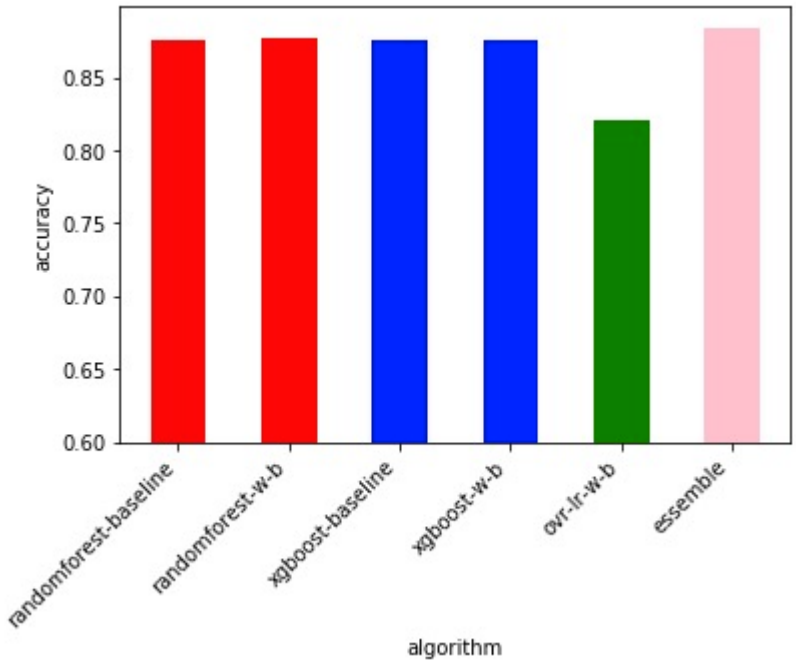
4) 最后，一共得到 m 个模型。

5) 在分类问题中，通过 m 个模型取多数票的方法，即可得到分类结果。

我们使用 Bagging 集成了随机森林、逻辑回归、xgboost 三个模型，以提高我们分类器的分类效果。

五、 实验结果

算法	正确率
randomforest-baseline 随机森林（WIFI + 经纬度）	0.875463
randomforest-w-b 随机森林（WIFI + weekend + busy + 经纬度）	0.876621
ovr-lr-w-b 逻辑回归（WIFI + weekend + busy + 经纬度）	0.821002
xgboost-baseline xgboost（WIFI + 经纬度）	0.875463
xgboost-w-b xgboost（WIFI + weekend + busy + 经纬度）	0.875748
Bagging 集成 （randomforest-w-b、ovr-lr-w-b、xgboost-w-b）	0.884415


0.85
0.80
0.75
0.70
0.65
0.60
accuracy
randomforest-baseline
randomforest-w-b
xgboost-baseline
xgboost-w-b
ovr-lr-w-b
essemble
algorithm