

大数据课程设计报告

队 名： 方形混凝土移动工程队

队 长： 崔西鹏

队员姓名： 戴宁 公绪蒙 陈加旭








专 业： 计算机科学与技术

完成日期： 2019.6.2

组内成员任务分配及比重

成员	工作内容	内容占比
崔西鹏	神经网络训练及结果上交	30%
戴宁	对比 lightgbm、XGBoost 模型结果 PPT、实践报告制作	35%
公绪蒙	特征工程，PPT 制作	30%
陈加旭	数据分析	5%

最终结果排名情况

Name	Submitted	Wait time	Execution time	Score		
simple_submission.csv	11 minutes ago	1 seconds	1 seconds	0.90048		
Complete						
Jump to your position on the leaderboard						
321	▼ 2	J Ma		0.90052	97	2mo
322	—	Linar Khusnutdinov		0.90051	60	2mo
323	▲ 2	Jack (Japan)		0.90050	4	2mo
324	▲ 3	Hugues		0.90038	22	2mo
325	▲ 2346	poutyface		0.90031	24	2mo
326	▲ 381	make_the_future		0.90028	41	2mo
327	▲ 3	Marek Wyborski		0.90025	124	2mo

排名：324/8802 前 3.7%

注：因提交结果时比赛已经结束，所以我们的结果并没有计入到排行榜中

摘要

关键词： PCA 降维 特征相关性分析 欠采样、过采样

全连接神经网络 XGBoost LightGBM

在本课程设计中，我们采用了特征关联分析进行特征提取，通过数据过采样、欠采样方法解决了样本分布比例不平衡的问题，通过对比全连接神经网络、XGBoost、LightGBM 等机器学习模型挑选出了效果最好的算法。

目录

1. 问题描述	1
1.1 原题描述	
1.2 要求简介	
2. 数据说明	2
2.1 数据概况	
2.2 数据预处理思路	
3. 解决思路解决方案	4
3.1 解决思路	
3.2 数据预处理方法	
3.3 模型选择	
4. 总结与展望	9
4.1 经验收获	
4.2 不足教训	

1. 问题描述

1.1 原题描述

At Santander our mission is to help people and businesses prosper. We are always looking for ways to help our customers understand their financial health and identify which products and services might help them achieve their monetary goals.

Our data science team is continually challenging our machine learning algorithms, working with the global data science community to make sure we can more accurately identify new ways to solve our most common challenge, binary classification problems such as: is a customer satisfied? Will a customer buy this product? Can a customer pay this loan?

In this challenge, we invite Kagglers to help us **identify which customers will make a specific transaction in the future, irrespective of the amount of money transacted**. The data provided for this competition has the same structure as the real data we have available to solve this problem.

1.2 要求简介

题目要求根据主办方银行提供的 20K 条用户数据，建立模型预测 20K 条测试集用户是否会进行交易。

2. 数据说明

2.1 数据概况

文件名	文件规格	内容描述
simple_submission.csv	20K * 2	用于提交的 test.csv 的数据结果
Train.csv	20K * 201	带标记的 20K 条用户数据，训练集
Test.csv	20K * 201	不带标记的 20K 条用户数据，测试集

主办方提供了 20K 条已标记的用户数据用作训练集，其中包括 200 个特征项，这些特征项的名称都经过了加密处理，其具体含义不得而知。测试集也包括 20K 条用户数据，内容与测试集类似但是不带标记，部分具体的数据格式如下：

训练集

ID_code	target	var_0	var_1	var_2	var_3	var_4	var_5
train_0	0	8.9255	-6.7863	11.9081	5.093	11.4607	-9.2834
train_1	0	11.5006	-4.1473	13.8588	5.389	12.3622	7.0433
train_2	1	8.6093	-2.7457	12.0805	7.8928	10.5825	-9.0837
train_3	0	11.0604	-2.1518	8.9522	7.1957	12.5846	-1.8361
train_4	0	9.8369	-1.4834	12.8746	6.6375	12.2772	2.4486
train_5	0	11.4763	-2.3182	12.608	8.6264	10.9621	3.5609

测试集

ID_code	var_0	var_1	var_2	var_3	var_4	var_5
test_0	11.0656	7.7798	12.9536	9.4292	11.4327	-2.3805
test_1	8.5304	1.2543	11.3047	5.1858	9.1974	-4.0117
test_2	5.4827	-10.3581	10.1407	7.0479	10.2628	9.8052
test_3	8.5374	-1.3222	12.022	6.5749	8.8458	3.1744
test_4	11.7058	-0.1327	14.1295	7.7506	9.1035	-8.5848
test_5	5.9862	-2.2913	8.6058	7.0685	14.2465	-8.6761
test_6	8.4624	-6.1065	7.3603	8.2627	12.0104	-7.2073

3.2 数据预处理思路

我们通过对数据进行了一系列可视化操作，得出的结果如下：

1. 训练集的样本分类比例不平衡，约为 1：9，后续需考虑提出解决不平衡样本的方法。

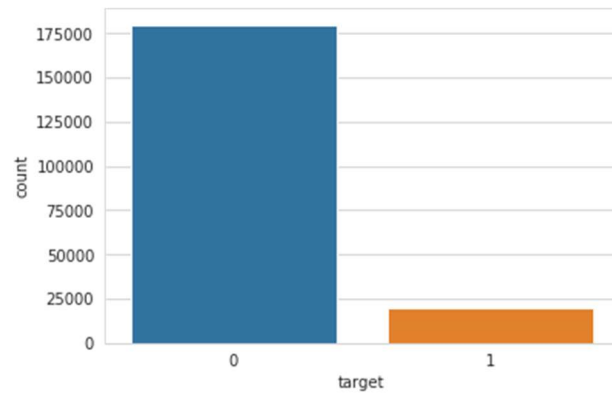


图 1 两类样本数量对比

2. 原始数据的特征数量为 200 个，数量过多且存在许多噪声特征项，如下图所示。这类特征项会影响数据训练的效率和速度，后期考虑去除部分噪声特征项。

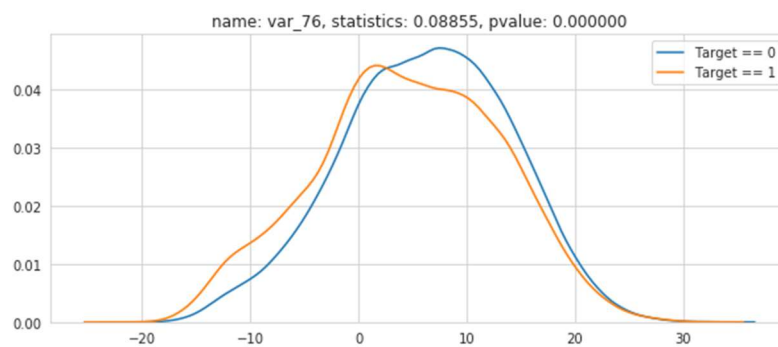


图 2 不同类样本的特征值存在显著差异

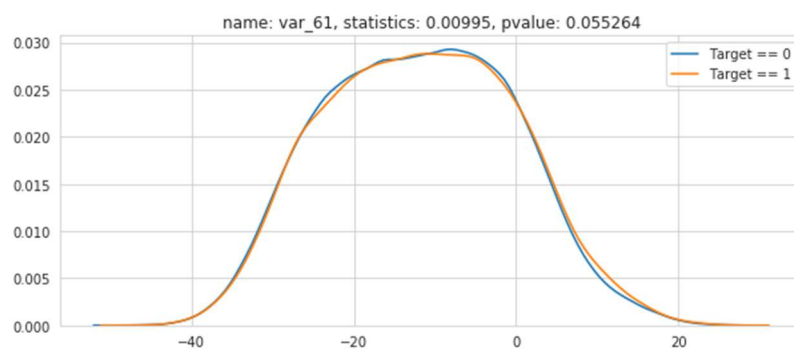


图 3 不同类样本的特征值没有明显差异

3. 解决思路解决方案

3.1 解决思路

基本的解决思路是先进行数据清洗，然后进行特征项提取，得到处理后的数据分别使用不同的机器学习模型训练数据，预测测试集，通过提交结果和训练的速度、内存的消耗对比模型之间的差异。

3.2 数据预处理方法

上文提到需要解决数据的两个问题：一是样本比例分布不平衡，比例约为 1: 9；二是特征项过多且存在与结果相关性较小的特征项。解决方法如下。

1. 样本比例分布不平衡

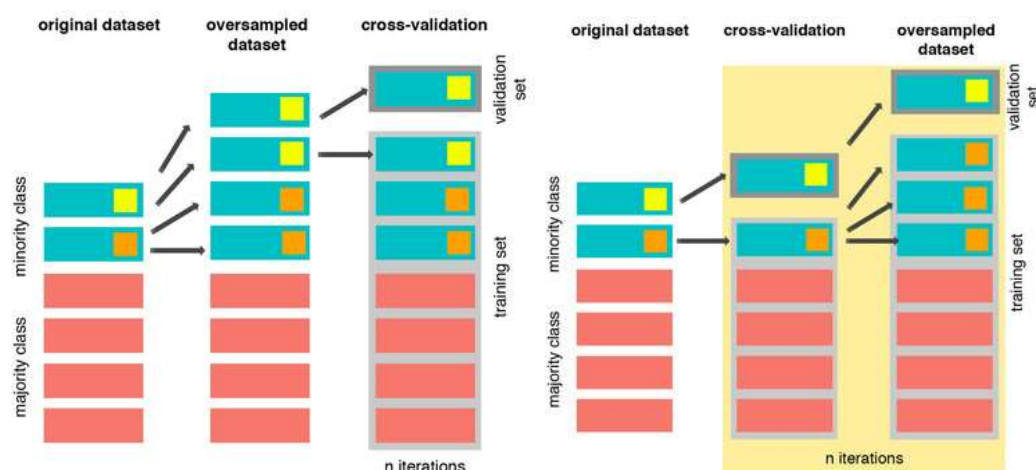
解决样本比例不平衡的问题，我们尝试了两种方法：①尝试 over-sampling 和 under-sampling。②将评判指标改为 ROC 和 AUC。

(1) under-sampling 欠采样（丢失信息）

从占比较大的类别下的样本中随机选择 n 个样本，其中 n 的值等于占比较小的类别下的样本的总数，并在训练阶段使用它们，然后在验证中排除掉这些样本。这种方法虽然容易丢失有用信息但是对算法影响较小

(2) over-sampling 过采样（过拟合）

在交叉验证之前做了过采样，复制原来的样本，然后使用留一法做交叉验证，也就是说我们在每次迭代中使用 $N-1$ 份样本做训练，而只使用 1 份样本验证。但是我们注意到在其实在 $N-1$ 份的样本中是包含了那一份用来做验证的样本的。



(3) 改变评判指标

为了查看样本比例不平衡所带来的结果，我们在程序中先使用了逻辑回归方法，在分类结果中我们发现，其错误率很低，然而这样的模型真的好么，当我们尝试查看类别 1 的精度时我们发现其 FNR ($FN/(FN+TP)$) 高达 0.76，发现逻辑回归虽然错误率很低，但是其真正效果并不好，所以在本实验中不采用准确率评判，而采用 ROCAUC (横轴 FPR，纵轴 TPR)

```
Recall : 0.23559322033898306
Accuracy : 0.9118333333333334
TP 139 : no. of 1 which are predicted 1
TN 5332 : no. of 0 which are predicted 0
FP 78 : no. of 0 which are predicted 1
FN 451 : no. of 1 which are predicted 0
FN rate : 0.764406779661017
```

2.特征项过多且存在噪声特征项

训练集中每条数据存在 200 个特征，且没有指明个特征的具体含义，所以我们不能从字面意思上判断某个特征项是否有训练价值，只能通过数据分析判断。

首先我们尝试的一种简单方法是主成分分析 (PCA)，运行结果如图所示。

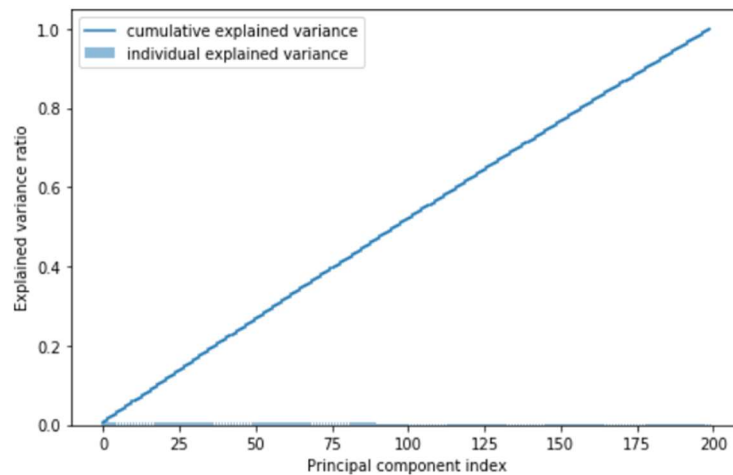


图 4 PCA 降维运行结果

PCA(Principal Component Analysis)，即主成分分析方法，是一种使用最广泛的数据降维算法。PCA 的主要思想是将 n 维特征映射到 k 维上，这 k 维是全新的正交特征也被称为主成分，是在原有 n 维特征的基础上重新构造出来的 k 维特征。PCA 的工作就是从原始的空间中顺序地找一组相互正交的坐标轴，新的坐标轴的选择与数据本身是密切相关的。其中，第一个新坐标轴选择是原始数据中方差最大的方向，第二个新坐标轴选取是与第一个坐标轴正交

的平面中使得方差最大的，第三个轴是与第 1,2 个轴正交的平面中方差最大的。依次类推，可以得到 n 个这样的坐标轴。通过这种方式获得的新的坐标轴，我们发现，大部分方差都包含在前面 k 个坐标轴中，后面的坐标轴所含的方差几乎为 0。于是，我们可以忽略余下的坐标轴，只保留前面 k 个含有绝大部分方差的坐标轴。事实上，这相当于只保留包含绝大部分方差的维度特征，而忽略包含方差几乎为 0 的特征维度，实现对数据特征的降维处理。

经过主成分分析我们发现，每个变量几乎都是互相独立的，其独立解释方差都相同，所以通过 PCA 降维的方法并没有预期效果。

PCA 降维失败后我们尝试对每个特征项与其 **target** 做相关性分析，并通过删除一些相关性较低的特征项来达到降维的效果。以下为通过相关性分析得到的相关数据。

target	1.000000		
var_81	0.080917	...	
var_139	0.074080	...	
var_12	0.069489	...	
var_6	0.066731	var_158	0.003817
var_110	0.064275	var_136	0.003554
var_146	0.063644	var_96	0.003037
var_53	0.063399	var_7	0.003025
var_26	0.062422	var_117	0.002591
var_76	0.061917	var_100	0.002215
var_174	0.061669	var_10	0.002213
var_22	0.060558	var_103	0.001395
var_21	0.058483	var_126	0.001393
var_99	0.058367	var_41	0.001298
var_166	0.057773	var_38	0.000970
var_80	0.057609	var_17	0.000864
var_190	0.055973	var_30	0.000638
var_2	0.055870	var_27	0.000582
var_165	0.055734	var_185	0.000053

通过特征相关性分析我们发现不同的特征项与 **target** 的相关性差异很大，最高与最低之间的甚至有几百倍的差距。我们去除掉最后 8 个相关性排名最低的特征项，达成了数据降维的预期目标。

```
test_x =
test.drop(['ID_code', 'var_185', 'var_27', 'var_30', 'var_17', 'var_38', 'var_41', 'var_126', 'var_103'],axis=1)
```

```
train_x =
train.drop(['ID_code', 'target', 'var_185', 'var_27', 'var_30', 'var_17', 'var_38', 'var_41', 'var_126', 'var_103'], axis=1)
train_y = train['target']
```

3.3 模型选择

我们尝试对比了神经网络、XgBoost、LightGBM 模型在训练数据的效率和结果。

1.首先我们将未处理的 submission.csv 提交系统，处理正确率为 0.50，即测试集中的样本分布比例为 1: 1，与训练集差别较大。

sample_submission.csv	0.50000	0.50000
9 days ago by RNG.mlxg		
add submission details		

2.神经网络是我们比较熟悉的一种模型，本题我们使用的是全连接网络模型。神经网络的搭建使用 Python 中的 keras 包，我们搭建了一个 5 层的全连接神经网络模型，通过对训练集进行训练并提交测试结果，得到了 0.75 的正确率。

Submission and Description	Private Score	Public Score
simple_submission.csv	0.75761	0.75875
a few seconds ago by RNG.mlxg		
add submission details		

3.XGBoost 是 boosting 算法的其中一种。Boosting 算法的思想是将许多弱分类器集成在一起形成一个强分类器。因为 XGBoost 是一种提升树模型，所以它是将许多树模型集成在一起，形成一个很强的分类器。XGBoost 算法思想就是不断地添加树，不断地进行特征分裂来生长一棵树，每次添加一个树，其实是学习一个新函数，去拟合上次预测的残差。当我们训练完成得到 k 棵树，我们要预测一个样本的分数，其实就是根据这个样本的特征，在每棵树中会落到对应的一个叶子节点，每个叶子节点就对应一个分数，最后只需要将每棵树对应的分数加起来就是该样本的预测值。

我们通过调整参数和模型训练后，得到了 0.80 的正确率。但该模型与 LightGBM 模型相比，内存开销较大，训练时间相对也长一些，且训练效果不如 LightGBM。

4. **LightGBM** 是个快速的，分布式的，高性能的基于决策树算法的梯度提升框架。可用于排序，分类，回归以及很多其他的机器学习任务中。

在竞赛题中，我们知道 **XGBoost** 算法非常热门，它是一种优秀的拉动框架，但是在使用过程中，其训练耗时很长，内存占用比较大。在 2017 年年 1 月微软在 **GitHub** 的上开源了一个新的升级工具--**LightGBM**。在不降低准确率的前提下，速度提升了 10 倍左右，占用内存下降了 3 倍左右。因为他是基于决策树算法的，它采用最优的叶明智策略分裂叶子节点，然而其它的提升算法分裂树一般采用的是深度方向或者水平明智而不是叶明智的。因此，在 **LightGBM** 算法中，当增长到相同的叶子节点，叶明智算法比水平-wise 算法减少更多的损失。因此导致更高的精度，而其他的任何已存在的提升算法都不能够达。与此同时，它的速度也让人感到震惊，这就是该算法名字的原因。

我们在多次调整 **LightGBM** 模型的参数后，得到了接近 0.9 的正确率，这是我们做出的最好的结果。

Submission and Description	Private Score	Public Score
simple_submission.csv 2 minutes ago by RNG.mlxg add submission details	0.89508	0.89913

4. 总结与展望

4.1. 经验收获

（1）学会了如何解决样本比例不平衡的问题，例如通过过采样与欠采样等方法平衡训练样本的比例

（2）学会利用特征相关性分析提取重要特征，这一方法在某些数据方面比通过字面意思提取特征项更有效。

（3）学习了 XGBoost、LightGBM 等在数据分析方面性能好、效率高的机器学习模型，了解了这一领域的前沿技术。

4.2. 不足与教训

（1）由于机器限制等原因，不能过多的迭代训练数据，对预测准确率可能有影响。

（2）没有尝试模型融合等方法，做出的实验结果只能在竞赛中排名中游。