南開大學
Nankai University

南　开　大　学

计 算 机 学 院

Python 实验报告

看山杯——机器学习实现

学号：1712991

姓名：周辰霏

年级：2017 级

专业：计算机科学与技术

2019 年 11 月 28 日

摘要

通过竞赛问题完整体会机器学习训练模型的全过程, 并着重考量数据预处理、特征选取以及模型调参部分, 最终得到一个不错的训练结果, 并在过程中得到多方面的提升。

关键字： 机器学习；看山杯 2019；lightGBM；调参；交叉验证

目录

一、 实验目的

基于问题背景下的简化数据集机器学习训练问题, 按照 8:1:1 的比例把数据分成训练集: 验证集: 测试集, 使用机器学习模型预测测试集的问题是否被回答。如果类别值为 1 表示邀请被回答, 值为 0 表示邀请没有被回答

二、 实验环境

- 系统环境:MacOS 10.15.1
- Jupyter Notebook 6.0.1
- Python 3.7.4
- numpy、matplotlib.pyplot、pandas、sklearn、seaborn、lightgbm

三、 问题分析

概括下来就是根据给出的问题信息、用户信息以及邀请回答状况等数据对之后邀请用户回答问题, 用户是否会回答问题进行预测。

针对原有赛题是一个简化, 给定数据集在特征量上进行了缩减, 且最终需要获取的结果变成了二分类问题, 虽然实质上没有太大的区别, 但是在给出结果的准确性上会有不少的提升。不过由于没有给出回答问题的相关特征, 少了一个个人认为也许可以很好发挥作用的特征。

不过这次大作业应该并不是需要我们做的有多么好, 能够对机器学习的全流程有一定程度了解并可以自己调出理想参数, 得到一个相对不错的结果应该就达到考核要求了。

四、 数据预处理

(一) 前期准备

首先是根据官网数据说明理解数据内容以及包含的内容信息, 有一些特征直观的就会觉得他们并没有什么用处, 但是是否舍弃还留待进一步验证, 比如「用户注册平台类型」等从特征名就能初步推断与问题相关度不大的特征。

之后参照网上一些做的不错的 baseline 模型以及一般机器学习进行预处理的步骤, 首先使用 pandas 以及 numpy 等的工具包, 对数据集整体情况有一个大致的了解, 随后可以结合 seaborn 等作图工具, 对一些数值或是分类信息进行可视化更清晰的表达, 以达到更好的数据选取目的。

通过对数据集初步简单的 pandas 和 seaborn 描绘, 对数据集有如下一系列认识:

```
question_id 9669
user_id 7334
invite_time 691
label 2
gender 3
keyword 1
grade 1
hotness 1
reg_type 1
reg_plat 1
freq 5
A1 2
B1 2
C1 2
D1 2
E1 2
A2 385
B2 33
C2 97
D2 336
E2 2
score 484
topic_attent 6208
topic_interest 6279
question_time 1660
title_sw_series 9669
title_w_series 9650
desc_sw_series 3842
desc_w_series 3796
topic 8901
```

图 1: 全部数据 unique 个数分析

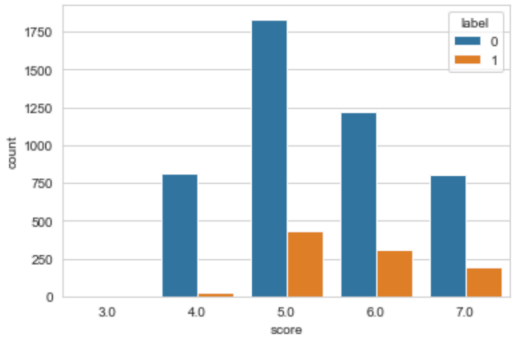

图 2: 可视化举例——盐值分数概况

通过以上分析, 可以发现创作关键词的编码序列、创作数量等级、创作热度等级、注册类型、注册平台这五个特征只有单值, 显然它对我们的模型预测没有任何帮助, 所以可以舍弃, 而盐值分数可以看到有较强的趋向性, 所以可以更多的关注这一特征。

(二) 特征提取

1. 首先看到问题创建时间和邀请创建时间都是形如 $D3833 - H8$ 的字符串, 为了方便模型处理, 我们将天数和时间拆开并转换为 int64 型

2. 问题标题和描述的单字、切词编码序列也都是一长串的字符串列表, 它们具体表示什么含义, 我们不需要深究, 但是我们可以根据它们的计数得到题目字数等的信息

3. 而通过上一节的准备工作, 我们发现给定数据集中的问题 id 和用户 id 并不都是独一无二的, 有不少的重复性, 所以我们也可以对这两个特征做聚合并统计其数字特征 (均值、方差、计数等)

4. 用户关注和感兴趣的话题表示用户关注和感兴趣的话题的序列编号, 所以对其计数, 猜测可以和活跃程度相对照作为参考; 相应的用户感兴趣的话题可以找出用户最感兴趣的话题, 并对每个用户对各话题感兴趣程度的数值进行数理统计。

5. 由于用户最感兴趣的话题以及用户 ID、问题 ID 等一系列特征都不是连续数字, 所以再使用 LabelEncoder 对其进行编号将之标准化

6. 将前面我们找出的一系列具有很好区分度的特征进行单特征计数

7. 最后删去那些我们处理过的特征, 就完成了最终的数据预处理

	A1	A2	B1	B2	C1	C2	D1	D2	E1	E2	...	A1_count	B1_count	C1_count	D1_count	E1_count	A2_count	B2_count	C2_count	D2_count	E2_count
0	2	248	1	25	1	77	1	162	1	1	...	1.0	1.0	1.0	1.0	1.0	0.026306	0.113601	0.074886	1.000000	0.0
1	2	34	1	26	1	43	1	123	1	2	...	1.0	1.0	1.0	1.0	1.0	0.007794	0.151403	0.290411	0.038154	1.0
2	1	208	2	12	1	94	2	309	2	2	...	0.0	0.0	1.0	0.0	0.0	1.000000	1.000000	0.448402	0.126579	1.0
3	2	208	1	12	1	42	1	162	1	2	...	1.0	1.0	1.0	1.0	1.0	1.000000	1.000000	0.260274	1.000000	1.0
4	1	208	2	12	1	92	1	162	1	2	...	0.0	0.0	1.0	1.0	1.0	1.000000	1.000000	0.613699	1.000000	1.0

5 rows × 54 columns

图 3: 数据预处理结果

五、 模型训练

完成了上述繁冗复杂但又最基本不可缺失的一步之后，就进入到了机器学习最重要又最有意思的步骤了，当务之急是选取一个合适的模型

(一) 模型选取

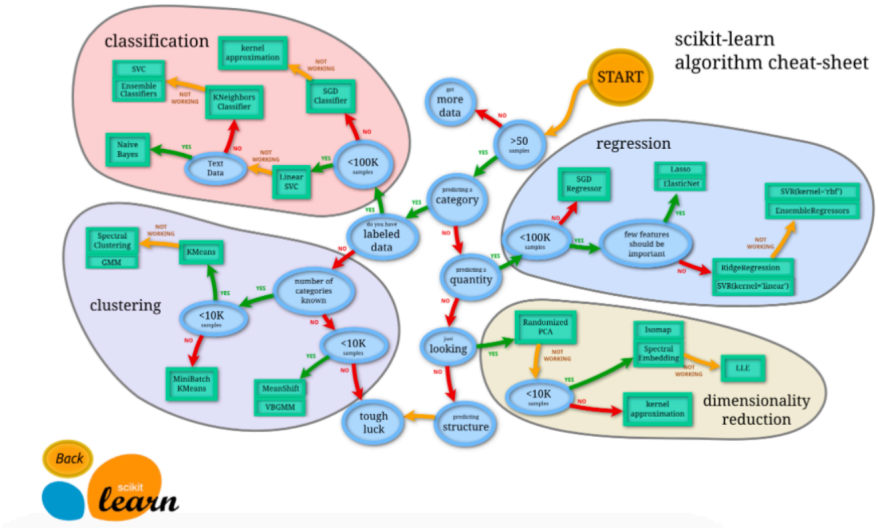

图 4: sklearn 模型选取

虽然有 sklearn 官方图支持，但实际上我们可选的模型远不止这些，但这张图可以帮我们排除一些不合适的模型。

查阅相关资料可得，常用的分类模型有:

- **KNN** K 近邻，简单但实践中并不常用
- 线性回归 简单，不适用于大数据集
- **SVM** 支持向量机，适合小样本且具有较强鲁棒性，但是可能我的特征过多，致使支持向量较多，所以速度极慢
- 朴素贝叶斯 简单，但是对先验概率的处理有一定难度
- 决策树 在处理特征关联性比较强的数据时表现得不是太好，容易过拟合，剪枝不易处理
- **LGBM**、**GBDT**、**XGB**、**CAT** 这几个都是在决策树、随机森林上优化的梯度提升算法

```
Gbdt
准确率: 0.8320000000000001
精确率: 0.7109139194139195
召回率: 0.07576653719211278
F1_score: 0.12628565536950048
AUC: 0.6915880679320303
Used time: 75
gbm
准确率: 0.8287000000000001
精确率: 0.5276738288891866
召回率: 0.0950990424173849
F1_score: 0.17192534997172146
AUC: 0.6977695361563947
Used time: 16
Xgbc
准确率: 0.8232999999999999
精确率: 0.6366163003663003
召回率: 0.05045990284966895
F1_score: 0.10938744833489648
AUC: 0.7002481283787463
Used time: 80
```

图 5: 模型选取

经过一个全部默认参数的试跑, 可以初步选 LGBM 模型, 原因很简单, 我们也不用冲比赛名次做得很好, 加上电脑配置一般, 就选个模型拟合较好且速度够快 (毕竟 GridSeachCV 调参够慢) 的就好

(二) 模型调参

使用 $train\_test\_split$ 将训练集和测试集分开以后, 通过 $StratifiedKFold$ 按要求将测试集划分为测试集和验证集以进行 9 折交叉验证, 提高训练准确率。

GridSeachCV 调参比较省事, 给定参数列表它就自己组合着去跑结果, 最后会给一个最优分数下的参数选择, 通过这种方式两个参数两个参数的调整, 可以逐步调节到合适的参数, 必要的话, 可以进行个位数量级的调参, 如 num_leaves、max_bin 等

最后得到一个合适的模型参数选择:

```
最优分类器: {'boosting_type': 'gbdt', 'colsample_bytree': 1, 'learning_rate': 0.01, 'max_bin': 425, 'min_child_sample
s': 10, 'min_child_weight': 5, 'min_split_gain': 0, 'n_estimators': 2000, 'n_jobs': -1, 'num_boost_round': 100, 'nu
m_leaves': 64, 'objective': 'binary', 'reg_alpha': 3, 'reg_lambda': 5, 'seed': 2019, 'silent': True, 'subsample': 0
.8, 'subsample_for_bin': 50000, 'subsample_freq': 1} 最优分数: 0.9998748604676765
```

图 6: 参数选取

(三) 结果验证

有了合适的模型以及参数, 就需要根据测试集预测出结果并进一步修正加强模型。最终的结果准确度可以有 accuracy、precision、recall、f1、auc 五个常用分数的衡量。

名称	公式	含义
accuracy 准确率	$\frac{TP+TN}{TP+FP+TN+FN}$	所有结果中检索正确的
precision 精确率	$\frac{TP}{TP+FP}$	正确被检索结果占实际被检索结果的比例
recall 召回率	$\frac{TP}{TP+FN}$	正确被检索结果占应该检索到结果的比例
f1 准确率与召回率的调和平均数	$\frac{2TP}{2TP+FP+FN}$	精确率与召回率的调和平均数
auc		ROC 曲线下与坐标轴围成的面积 衡量检测方法真实性

表 1: 衡量结果参数

根据上述几个参数的综合比较可以验证模型拟合程度, 之后借助这几个重要参数以及 lgb 提供的特征重要性图像进一步对特征进行优化, 详情见代码

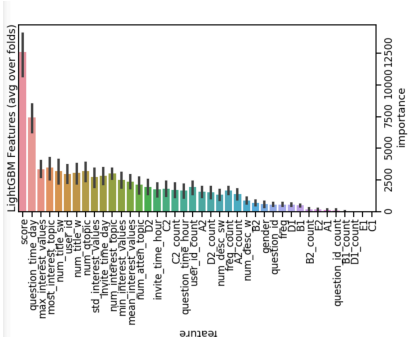

图 7: 特征重要程度

进过多次调整以后, 本地测试结果如下

```
Early stopping, best iteration is:
[1019]  valid_0's binary_logloss: 0.00175653
训练集分数:  0.9997499687460932 测试集分数 0.998
测试集结果比对 0.998
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       813
           1       0.99      0.99      0.99       187

    accuracy                           1.00      1000
   macro avg       1.00      1.00      1.00      1000
weighted avg       1.00      1.00      1.00      1000

   accuracy  precision    recall        f1       auc
0     0.998   0.994652  0.994652  0.994652  0.996711
Used time: 2
```

图 8: 本地测试结果

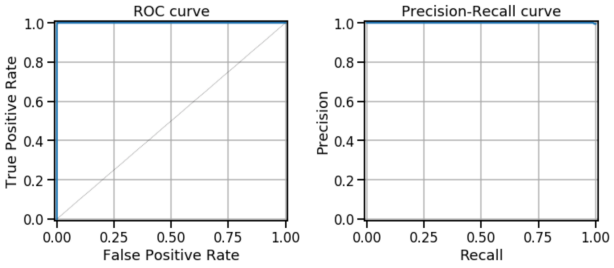

图 9: ROC 曲线

六、 实验遇到问题及解决

- 最开始选取模型的时候只对准确率和精确率进行了测试, 所以误以为 KNN 和朴素贝叶斯效果也挺好, 还一度尝试调参, 后续加入 auc 参数后发现这两个模型不太适合过多特征的情况, 一个是速度问题, 还有一个是这两个模型的 auc 只比 50% 高一点, 所以就说明这两个模型也就只比扔硬币好那么一点点, 完全没有深入研究的价值
- 尽管调参的分数数据很好看, 但实际跑测试集准确度也就是 85% 左右, 也就比我全猜 0(不回答) 好一点点, 后来对问题 id 和用户 id 进行聚合处理以后, 结果居然惊人的好, 至少本地数据集结果很好看
- CSDN 水平参差不齐, 有的确实写得深入浅出, 有的完全是瞎写, 所以还是直接看微软的官方文档比较不容易出错, 虽然看着困难但是少了纠错又查来查去的不必要时间
- 起初没有设置 seed, 给模型训练增添了不少不确定性
- 其实还想尝试一下 boosting 对 lgbm 的增强, 结果发现好像小数据集结果还挺可以的, 就没有继续弄了
- 不算这个 project 的, 其实第一次 pro 也有做甚至还找到一个不错的模型, 但是语义一直做得不好就一直没进行 (毕竟老师没说 DDL), 而且都过了提交时间了才告诉我们要提交, 很多人都没有准备啊 TAT, 所以希望后续的课程上能在作业考核的具体安排上可以更详细一点谢谢 TwT

七、 总结

虽然一起接触过不少神经网络 RNN 的调参, 但是这样从预处理到最终获得预测结果这样全程做完还是头一遭, 踩到了不少坑, 最后也都自己填上了, 而且养成了直接去 Google 搜官方文档的习惯 (虽然可能还是需要借助一些博客去理解), 虽然任务做了最大的简化, 但是还是收获良多, 也算是体验了一下 ML 竞赛的全流程吧, 还有不少没有学到的, 所以后续课业闲暇之余也可以关注这一些竞赛参考别人的好的方案, 积累积累经验。

参考文献

[1] lightGBM 官方文档 [EB/OL].https://lightgbm.readthedocs.io/en/latest/Python-API.html

[2] 2019 看山杯专家发现算法大赛 Baseline(0.7632)[EB/OL].https://biendata.com/models/category/2897/L_notebook/

[3] sklearn 官方文档 [EB/OL].https://scikit-learn.org/stable/index.html