优达学城数据分析师纳米学位项目 P5

安然提交开放式问题

说明：[你可以在这里下载此文档的英文版本](https://s3.cn-north-1.amazonaws.com.cn/static-documents/nd002/Enron+Submission+Free-Response+Questions.pdf)。

机器学习的一个重要部分就是明确你的分析过程，并有效地传达给他人。下面的问题将帮助我们理解你的决策过程及为你的项目提供反馈。请回答每个问题；每个问题的答案长度应为大概 1 到 2 段文字。如果你发现自己的答案过长，请看看是否可加以精简！

当评估员审查你的回答时，他或她将使用特定标准项清单来评估你的答案。下面是该标准的链接：[评估准则](https://review.udacity.com/?&_ga=1.206413269.797580181.1464528540#!/rubrics/310/view)。每个问题有一或多个关联的特定标准项，因此在提交答案前，请先查阅标准的相应部分。如果你的回答未满足所有标准点的期望，你将需要修改和重新提交项目。确保你的回答有足够的详细信息，使评估员能够理解你在进行数据分析时采取的每个步骤和思考过程。

提交回答后，你的导师将查看并对你的一个或多个答案提出几个更有针对性的后续问题。

我们期待看到你的项目成果！

1. 向我们总结此项目的目标以及机器学习对于实现此目标有何帮助。作为答案的部分，提供一些数据集背景信息以及这些信息如何用于回答项目问题。你在获得数据时它们是否包含任何异常值，你是如何进行处理的？【相关标准项：“数据探索”，“异常值调查”】

一个146个数据点，加上name有22个特征。

（1）首先删除异常点TOTAL，这是汇总表，需要删除。

（2）然后我把数据字典按照姓名转换为pandas的dataframe。然后使用df.info， 可以发现loan\_advances（4 non-null）、director\_fees（17）、deferral\_payments（39）、deferred\_income（49）、restricted\_stock\_deferred （18）缺失值过多，故直接舍弃。对于其他有缺失值的直接将其缺失值化为0。email\_address是各自不相同的字符串，也直接舍弃。

（3）然后通过观察df.describe，发现仅有BHATNAGAR SANJAY这个人的restricted\_stock为负，仅有BELFER ROBERT的total\_stock\_value为负，故查询pdf文档，发现实际上前者的值是正的，后者的值不存在，应该是处理数据时出现了认为错误，故将这个前者转为正，后者处理为0。

1. 你最终在你的 POI 标识符中使用了什么特征，你使用了什么筛选过程来挑选它们？你是否需要进行任何缩放？为什么？作为任务的一部分，你应该尝试设计自己的特征，而非使用数据集中现成的——解释你尝试创建的特征及其基本原理。（你不一定要在最后的分析中使用它，而只设计并测试它）。在你的特征选择步骤，如果你使用了算法（如决策树），请也给出所使用特征的特征重要性；如果你使用了自动特征选择函数（如 SelectBest），请报告特征得分及你所选的参数值的原因。【相关标准项：“创建新特征”、“适当缩放特征”、“智能选择功能”】
2. 最终选择了salary，total\_payments，bonus，total\_stock\_value，exercised\_stock\_options，long\_term\_incentive，restricted\_stock，ratio\_from\_poi， shared\_receipt\_with\_poi特征。首先排除掉name、emailadress等重复的字符串，缺失值太多的特征值也不用，然后我在jupternotebook中通过两两变量间与poi的关系的关系来初步确定要选的特征。最后使用SelectKBest来选择最后的特征，选择卡方值大于2的值，删除了expenses、other特征，最后选择了上面的9个特征。
3. 我最终选择了minmaxscaler来使各个特征值落在0-1内，因为各个特征之间值的差距很大会影响到部分算法（如SVM）的结果
4. 我创造了ratio\_to\_poi，ratio\_from\_poi，及收到或者发给poi邮箱的比例，因为我认为单独看邮件收发件数没什么意义，poi占邮件总数比例才更有意义。

（4） 使用的是特征变量与poi的卡方值，之所以选这个是因为用决策树做出的结果不是很满意，而且这个调出来的特征拟合得还行。

|  |  |  |
| --- | --- | --- |
| Salary  3.11664385 | total\_payments  2.81150913 | bonus  5.19334926 |
| total\_stock\_value  5.63277212 | Expenses  1.52565657 | exercised\_stock\_options  6.92747711 |
| Other  1.7085568 | long\_term\_incentive  2.57968986 | restricted\_stock  2.54385275 |
| ratio\_from\_poi  4.71795058 | ratio\_to\_poi  0.85098126 | shared\_receipt\_with\_poi  2.48230712 |

1. 你最终使用了什么算法？你还尝试了其他什么算法？不同算法之间的模型性能有何差异？【相关标准项：“选择算法”】

我最终采用了SVM：高斯核

尝试算法：（以下四个指标均是在test.py运行的结果）

（1）朴素贝叶斯，效果一般。Accuracy: 0.84040 Precision: 0.35098 Recall: 0.23200 F1: 0.27935 F2: 0.24887

（2）逻辑回归，LogisticRegression增加了对unbalance数据的权重，poi：非poi=7:1, penalty='l1'，效果还可以，后Accuracy:0.76980 Precision: 0.31941 Recall: 0.64250 F1: 0.42670 F2: 0.53439

加入交叉验证CV与加入正则项C之后效果反而变差了。

（3）SVM：高斯核，Accuracy: 0.74880 Precision: 0.30477 Recall: 0.69000 F1: 0.42279 F2: 0.55077，gama=0.19稍微调参后效果比逻辑回归好

Accuracy: 0.75493 Precision: 0.31599 Recall: 0.71950 F1: 0.43912 F2: 0.57312

1. 决策树，与随机森林，效果也一般。

Accuracy:0.84000 Precision: 0.37879 Recall: 0.31250 F1: 0.34247 F2: 0.32383

1. 调整算法的参数是什么意思，如果你不这样做会发生什么？你是如何调整特定算法的参数的？（一些算法没有需要调整的参数 – 如果你选择的算法是这种情况，指明并简要解释对于你最终未选择的模型或需要参数调整的不同模型，例如决策树分类器，你会怎么做）。【相关标准项：“调整算法”】
2. 调整算法的参数就是调整算法中的内部的配置变量，就是可以用数据估计模型参数的值。因为在现实中我们一般都不知道样本的真实分布情况，在学习过程中会有方差和偏差的存在，如果不调整参数，可能会导致过拟合或欠拟合，导致我们训练出来的模型预测能力不好。
3. 我刚开始选了logistic回归，调整了penalty和class\_weight参数，发现penalty=‘l1’训练出来的效果比默认的好。而LogisticRegressionCV尝试对cs参数进行调整，但是效果并不好。
4. 然后我选择了svm.SVC，选择了rbf核，因为线性映射的效果没这个好，同样也调整了class\_weight参数。然后用GridSearchCV调整了C参数与gama参数，发现C参数是默认值，但是gama是-0.0005,gama太小了，特别容易过拟合，结果也是如此。因而我最后手动调参，选择了kernel='rbf',class\_weight={0: 1, 1: 7},gamma=0.19，这个得出来的结果相对logistic回归好一点。
5. 什么是验证，未正确执行情况下的典型错误是什么？你是如何验证你的分析的？【相关标准项：“验证策略”】
6. 我理解的验证应该是为了评估模型学习的泛化误差，而将样本拆分，一部分训练模型另一部来测试模型，验证样本是否过拟合或者欠拟合。未正确执行情况下的典型错误是模型过拟合。
7. 我将样本按照7:3拆成了训练集与测试集，选择模型与调参时进行验证。同时还尝试GridSearchCV里的cv参数进行交叉验证。
8. 给出至少 2 个评估度量并说明每个的平均性能。解释对用简单的语言表明算法性能的度量的解读。【相关标准项：“评估度量的使用”】
9. 代码中我主要采用sklearn.metrics中的accuracy\_score, classification\_report。其中主要关注classification\_report中的precision与recall。
10. 因为我尝试的算法比较多，使用出现的评估结果也比较多，我仅说明SVM。

SVM中测试集precision 0.43，recall 0.33， f1-score 0.38，accuracy 0.772727272727。

而test.py结果 Precision: 0.30477 Recall: 0.69000 F1: 0.42279 Accuracy: 0.74880。

来看看我所关注的precision与recall，precision计算的是所有"正确被识别的poi"占所有"实际poi加上预测错的poi (TP+FP)"的比例，而recall计算的是所有"正确被识别的poi "占所有"实际poi (TP+FN)"的比例。

因为这个项目中poi相对于非poi的值是很少的，因而我们的目的就是尽量地提高识别poi的准确率，而非poi识别的准确率我们就不是很关注了。

相比于其他分类器，我们这个模型所得出的precision与recall我还是十分满意的，precision高保证了识别poi的准确性，recall高保证了尽量把poi都识别了。

优达学城

2016年9月