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" \l "!/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”】

项目背景：安然曾是 2000 年美国最大的公司之一。2002 年，由于其存在大量的企业欺诈行为，这个昔日的大集团土崩瓦解。 在随后联邦进行的调查过程中，大量有代表性的保密信息进入了公众的视线，包括成千上万涉及高管的邮件和详细的财务数据。

目标：根据公开的安然财务和电子邮件数据集，构建算法，找出有欺诈嫌疑的安然员工

数据集背景：

数据集中有146个对象（18个是POI,128个不是POI）,大部分是安然公司高层交流的真实邮件。数据集有21个特征，分为经济特征、邮件特征、POI

**经济特征**：

['salary','deferral\_payments','total\_payments','loan\_advances','bonus','restricted\_stock\_deferred','deferred\_income','total\_stock\_value','expenses','exercised\_stock\_options','other','long\_term\_incentive', 'restricted\_stock', 'director\_fees'] (单位都是美元)

**邮件特征**：

['to\_messages','email\_address','from\_poi\_to\_this\_person','from\_messages', 'from\_this\_person\_to\_poi', 'shared\_receipt\_with\_poi']

(单位是个数)

**POI特征**：

['poi']

一些特征有很多丢失值，下面每个特征的具体丢失值数量

{'salary': 51,

'to\_messages': 60,

'deferral\_payments':107,

'total\_payments': 21,

'exercised\_stock\_options': 44,

'bonus': 64,

'director\_fees': 129,

'restricted\_stock\_deferred': 128,

'from\_messages': 60,

'total\_stock\_value': 20,

'expenses': 51,

'from\_poi\_to\_this\_person': 60,

'loan\_advances': 142,

'email\_address': 35,

'other': 53,

'from\_this\_person\_to\_poi': 60,

'poi': 0, 'deferred\_income': 97,

'shared\_receipt\_with\_poi': 60,

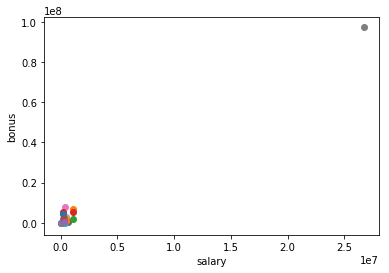
'restricted\_stock': 36,

'long\_term\_incentive': 80}

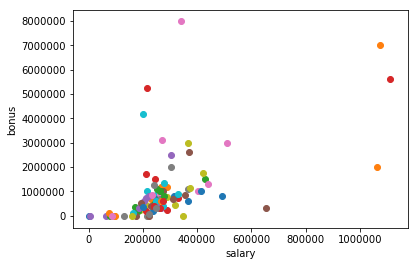
丢失值可以全部用 0 代替

**异常值**：

绘制 bonus vs salary 散点图时，发现异常值，异常列是“TOTAL”



移除后，散点图显示如下



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”】

我创建了2个新特征：

fraction\_from\_poi:代表从POI发到这个人的邮件数占他总收到的邮件数的比率

fraction\_to\_poi: 代表这个人发给POI的邮件数占他所有发送的邮件数的比率

下面的代码就是为了计算上述比率的：

def computeFraction(poi\_messages, all\_messages):

fraction = 0.

if poi\_messages != ‘NaN’ and all\_messages != ‘NaN’:

Fraction = poi\_messages/float(all\_messages)

return fraction

计算每一个雇员的收发比率，同时把新特征加入到原始数据集中：

for emp in my\_dataset:

from\_poi\_to\_this\_person = my\_dataset[emp][‘from\_poi\_to\_this\_person’]

to\_messages = my\_dataset[emp][‘to\_messages’]

fraction\_from\_poi = computeFraction(from\_poi\_to\_this\_person, to\_messages)

my\_dataset[emp][‘fraction\_from\_poi’] = fraction\_from\_poi

from\_this\_person\_to\_poi = my\_dataset[emp][‘from\_this\_person\_to\_poi’]

from\_messages = my\_dataset[emp][‘from\_messages’]

fraction\_to\_poi = computeFraction(from\_this\_person\_to\_poi, from\_messages)

my\_dataset[emp][‘fraction\_to\_poi’] = fraction\_to\_poi

单变量特征选择：

为了找出最好的特征，使用了自动选择特征函数：SelectKBest, 可以选择K个最重要的特征，结果如下：

[('exercised\_stock\_options', 25.097541528735491),

('total\_stock\_value', 24.467654047526398),

('bonus', 21.060001707536571),

('salary', 18.575703268041785),

('fraction\_to\_poi', 16.641707070468989),

('deferred\_income', 11.595547659730601),

('long\_term\_incentive', 10.072454529369441),

('restricted\_stock', 9.3467007910514877),

('total\_payments', 8.8667215371077717),

('shared\_receipt\_with\_poi', 8.7464855321290802),

('loan\_advances', 7.2427303965360181),

('expenses', 6.2342011405067401),

('from\_poi\_to\_this\_person', 5.3449415231473374),

('other', 4.204970858301416),

('fraction\_from\_poi', 3.2107619169667441),

('from\_this\_person\_to\_poi', 2.4265081272428781),

('director\_fees', 2.1076559432760908),

('to\_messages', 1.6988243485808501),

('deferral\_payments', 0.2170589303395084),

('from\_messages', 0.16416449823428736),

('restricted\_stock\_deferred', 0.06498431172371151)]

为了观察新增特征的影响，对每个算法都做了对比

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Algorithm | precision | recall | Precision | recall |
|  | Original feature | Original feature | New feature | New feature |
| Naive Bayes | 0.401150793651 | 0.337827380952 | 0.396896194084 | 0.332306547619 |
| SVM | 0.225744047619 | 0.0862450396825 | 0.163958333333 | 0.0596775793651 |
| Decision Tree | 0.283698697136 | 0.286919642857 | 0.296833062771 | 0.286919642857 |
| Random Forest | 0.282708333333 | 0.150768849206 | 0.347232142857 | 0.15029265873 |
| Logistic Regression | 0.0 | 0.0 | 0.0 | 0.0 |

可以发现，每个算法的表现都不一样，加新特征后，决策树、随机森林算法表现更好，贝叶斯、向量机反而表现更差。

我打算选择10个通过SelectKBest选出来的得分最高的特征来研究

['poi',

'exercised\_stock\_options',

'total\_stock\_value', 'bonus',

'salary', 'fraction\_to\_poi',

'deferred\_income',

'long\_term\_incentive',

'restricted\_stock', 'total\_payments',

'shared\_receipt\_with\_poi']

特征缩放

因为选择的特征单位不一致，且有些特征数值特别大，因些要做特征缩放，使用MinMaxScaler 函数

from sklearn import preprocessing

data = featureFormat(my\_dataset, kBest\_features, sort\_keys = True)

labels, features = targetFeatureSplit(data)

scaler = preprocessing.MinMaxScaler()

features = scaler.fit\_transform(features)

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”】

最终选择 贝叶斯（Naive Bays）算法,因为评估指标得分最高，还尝试了：

SVM、Decision Tree、Random Forest、Logistic Regression

用SVM、Logistic Regression 算法，花的时间比较长，另外，各个算法的准确率（accuracy）都比较高，如果用这个指标做评估，就不太好

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”】

大多数算法都有一些参数，传入不同的参数值，会影响模型性能。

算法调整：

我使用 GridSearchCV 来调整算法，它会自动选择最优参数来构建模型算法

如下，在执行80次迭代调整后，会找出最佳的参数搭配，输出平均评估指标结果

def tune\_params(grid\_search, features, labels, params, iters = 80)

acc = []

pre = []

recall = []

for i in range(iters):

features\_train, features\_test, labels\_train, labels\_test = \

train\_test\_split(features, labels, test\_size = 0.3, random\_state = i)

grid\_search.fit(features\_train, labels\_train)

predicts = grid\_search.predict(features\_test)

acc = acc + [accuracy\_score(labels\_test, predicts)]

pre = pre + [precision\_score(labels\_test, predicts)]

recall = recall + [recall\_score(labels\_test, predicts)]

print “accuracy:{}”.format(np.mean(acc))

print “precision:{}”.format(np.mean(pre))

print “recall:{}”.format(np.mean(recall))

调整向量机（SVM）分类器

from sklearn import svm

svm\_clf = svm.SVC()

svm\_param = {‘kernel’:(‘linear’, ‘rbf’, ‘sigmoid’),

‘gamma’:[1, 0.1, 0.01, 0.001, 0.0001],

‘C’: [0.1, 1, 10, 100, 1000]}

svm\_grid\_search = GridSearchCV(estimator = svm\_clf, param\_grid = svm\_param)

tune\_params(svm\_grid\_search, features, labels, svm\_params)

结果：

kernel = ‘rbf’, C = 1000, gamma = 0.01

调整决策树（Desicion Tree）分类器

from sklearn import tree

dt\_clf = tree.DecisionTreeClassifier()

dt\_param = {‘criterion’: (‘gini’, ‘entropy’),

‘splitter’:(‘best’, ‘random’)}

dt\_grid\_search = GridSearchCV(estimator = dt\_clf, param\_grid = dt\_param)

tune\_params(dt\_grid\_search, features, labels, dt\_param)

结果

没有新特征：

criterion = ‘gini’,splitter = ‘random’

有新特征：

criterion = ‘entropy’, splitter = ‘best’

调整逻辑回归（Logistic Regression）分类器

From sklearn.linear\_model import LogisticRegression

Lr\_clf = LogisticRegression()

Lr\_param = {‘tol’: [1, 0.1, 0.01, 0.001, 0.0001],

‘C’:[0.1, 0.01, 0.001, 0.0001]}

Lr\_grid\_search = GridSearchCv(estimator = lr\_clf, param\_grid = lr\_param)

Tune\_params(lr\_grid\_search, features, labels, lr\_param)

结果：

C= 0.1， tol = 1

1. 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”】

验证是使用测试数据集评估训练模型的过程，未正确执行的典型错误是未把数据集分为训练数据集和测试数据集，会导致过拟合，我使用交叉验证函数 train\_test\_split 将数据集切割30%用作测试集，再使用sklearn.metrics 中的accuracy、precision、recall 验证算法

from sklearn.cross\_validation import train\_test\_split

features\_train, features\_test, labels\_train, labels\_test = \

Train\_test\_split(features, labels, test\_size = 0.3, random\_state = 42)

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score

accuracy\_score(labels\_test, predicts)

precision\_score(labels\_test, predicts)

recall\_score(labels\_test, predicts)

下面表格列出了结果

|  |  |  |  |
| --- | --- | --- | --- |
| Algorithm | Accuracy | Precision | recall |
| Naive Bayes | 0.849431818182 | 0.401150793651 | 0.337827380952 |
| SVM | 0.868465909091 | 0.225744047619 | 0.0862450396825 |
| Decision Tree | 0.804545454545 | 0.234825591076 | 0.291840277778 |
| Random Forest | 0.869318181818 | 0.431041666667 | 0.17962797619 |
| Logistic Regression | 0.875284090909 | 0.0 | 0.0 |

1. 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”】

**精确度(precision)**是针对我们**预测结果**而言的，它表示的是预测为正的样本中有多少是真正的正样本。那么预测为正就有两种可能了，一种就是把正类预测为正类(TP)，另一种就是把负类预测为正类(FP)

equation

本次项目中，代表被识别为POI的人中，有0.4011是真正的POI

**召回率（recall）**是针对我们原来的**样本**而言的，它表示的是样本中的正例有多少被预测正确了。那也有两种可能，一种是把原来的正类预测成正类(TP)，另一种就是把原来的正类预测为负类(FN)

2

本次项目中，代表数据集所有是POI的人中，被识别为POI的比例是0.33

使用test.py 测试，如下



发现precision,recall 更低，经过观察发现，其使用交叉验证函数StratifiedShuffleSplit，此函数合并了K折验证功能，用了K次数据集分成训练集和测试集，而train\_test\_split只会分一次，因此精确度更高

参考引用：

1.https://www.zhihu.com/question/19645541

2.http://scikit-learn.org/stable/modules/generated/sklearn.linear\_model.LogisticRegression.html

3.http://machinelearningmastery.com/how-to-tune-algorithm-parameters-with-scikit-learn/

4.http://scikit-learn.org/stable/modules/cross\_validation.html

5.https://github.com/yielder/identifying-fraud-from-enron-email

6.https://en.wikipedia.org/wiki/Cross-validation\_(statistics)

7.http://scikit-learn.org/stable/modules/generated/sklearn.cross\_validation.StratifiedShuffleSplit.html

优达学城

2016年9月