****

****

**QG中期考核详细报告书**

**题    目  QG中期考核详细报告书**

**学   院     计算机学院**

**专 业 计算机类**

**年级班别 20级计算机类12班**

**学 号**  3220005188

**学生姓名 聂芳琪**

**2020年 4 月 15 日**

目录

[一、前言 3](#_Toc23336)

[二、赛题分析 4](#_Toc13010)

[数据介绍简要概括 4](#_Toc11538)

[数据集 4](#_Toc22341)

[任务 4](#_Toc30664)

[目的 4](#_Toc15134)

[描述性特征 4](#_Toc15340)

[三、流程（两个阶段） 5](#_Toc27838)

[第一阶段 5](#_Toc17796)

[数据探索 5](#_Toc10931)

[导入数据 5](#_Toc22805)

[四、特征工程 24](#_Toc28370)

[数据预处理 24](#_Toc23265)

[处理过程： 24](#_Toc7953)

[a. 首先把数据型的列放在一个列表中，把数据型的列放在另一个列表中 24](#_Toc18648)

[b. 先处理数据型: 归一化 24](#_Toc31273)

[c. 处理非数据型：LabelEncoder OneHotEncoder 25](#_Toc25909)

[五、模型融合以及测试 25](#_Toc17991)

[（1）模型选择 25](#_Toc7904)

[（2）模型优化 26](#_Toc14938)

[（3）集成学习（Model Ensemble） 28](#_Toc28172)

[六、 应用 28](#_Toc7873)

[七、 总结 28](#_Toc20146)

# 一、前言

在接到这个任务后的第二晚，我开始了对数据挖掘竞赛的了解，Datawhale上的大佬有一大堆的总结，说了好多，但是我在脑海里只记住了两句话，（这里不是说大佬讲的太多，纯粹是我没记住）：

1. 代码可以复制的就不自己敲

2. 解决问题远比证明自己更加优秀

听起来真的真的真的好高级，尤其是第一点！（不过由于自己编程太菜，我还是手动输入吧，自觉）于是乎，我就上kaggle，找了一篇预测挺高分的泰坦尼克号的新手文章，打算用一个上午的时间研读一番，但是，由于自己的英文水平，以及python的水平远远不及大佬，一个超级菜的聂某人，非常自觉地离开了这篇文章。转战知乎，找了另一篇数据挖掘实战分享笔记，是中文版。

这里附上一份链接：《Kaggle Titanic 生存预测 -- 详细流程吐血梳理》<https://zhuanlan.zhihu.com/p/31743196>

我真的非常非常非常感谢这位大佬的“吐血”梳理以及感谢Datawhale上大佬的无私奉献，让我对数据挖掘竞赛的流程有了基本的认识，以及大概的操作。而我也正式从一名萌新从入门到入坑。当我看到通过自己的处理数据，调整参数，设置特征……使得预测值不断上升的时候，开心！这两个字足以形容，这一次QG训练营之后，我想继续在kaggle玩，然后呢，还要提高自己的英文水平（残忍现实）

前言大概差不多是这些了，感谢QG工作室的师兄非常有耐心的看完，万分感谢。

# 二、赛题分析

## 数据介绍简要概括

### 数据集

针对培训课程测试明智的学员绩效数据集

### 任务

是根据人口统计信息和培训计划/测试详细信息来预测此类测试的性能。

### 目的

通过找出最重要的因素来提高受训者的参与度和表现，这将使该公司加强其培训问题

## 描述性特征

id\_num 唯一ID

program\_id 程序的ID

program\_type 程序类型

program\_duration 计划持续时间（天）

test\_id 测试ID

test\_type 测试类型（离线/在线）

difficulty\_level 测试难度级别

trainee\_id 学员的ID

gender 受训者性别

education 学员的教育水平

city\_tier 实习生居住城市的等级

age 受训者年龄

total\_programs\_enrolled 总课程的学生通过实习

is\_handicapped 受训者是否患有残疾？

trainee\_engagement\_rating 讲师/教学助理为课程提供学员参与度

is\_pass 0 测试失败，1 测试通过

# 三、流程（两个阶段）

 第一阶段：专注于数据预处理和探索

 第二阶段：模型的建立，验证和预测

## 第一阶段

### 数据探索

## 导入数据

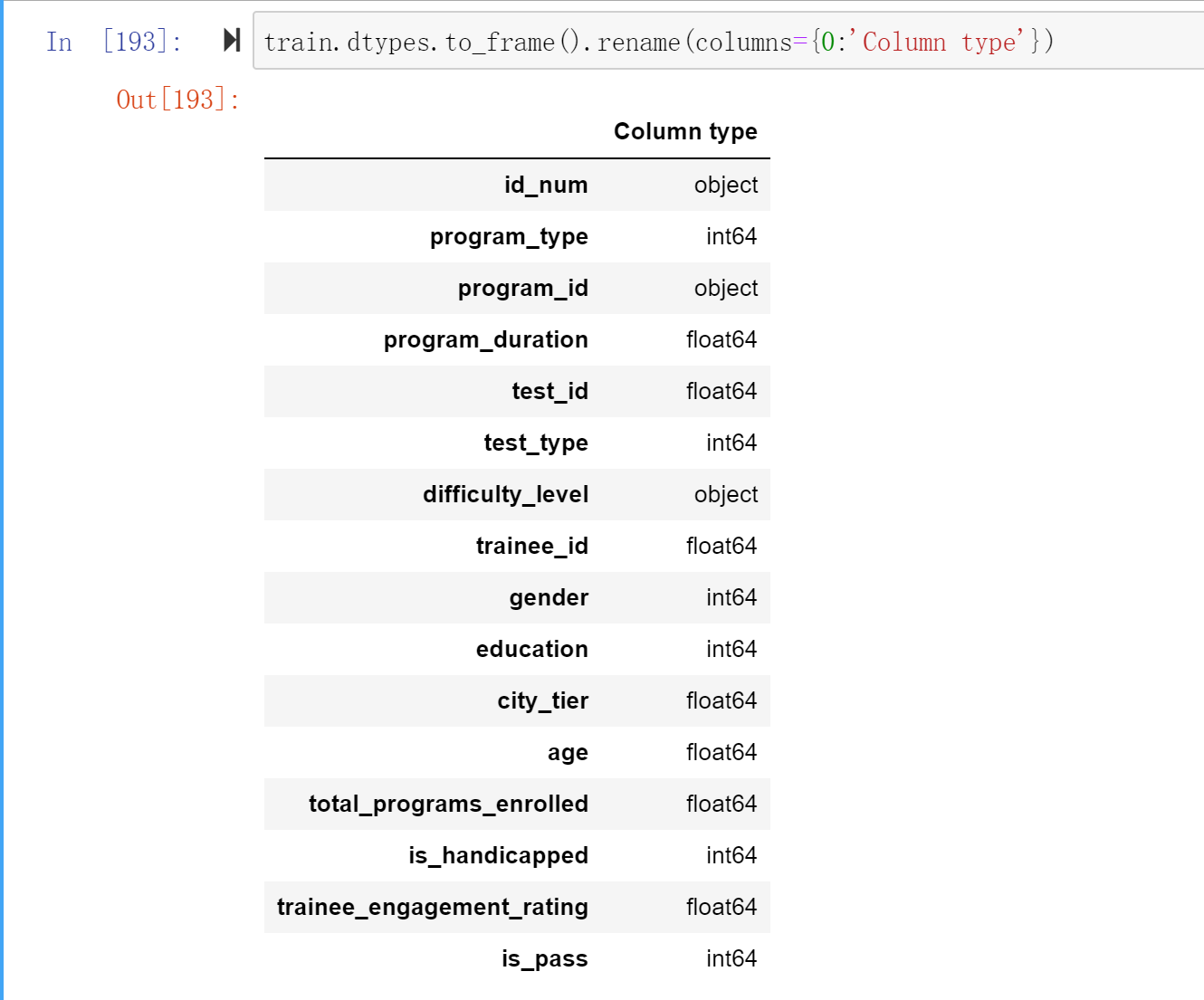
读取并复制数据

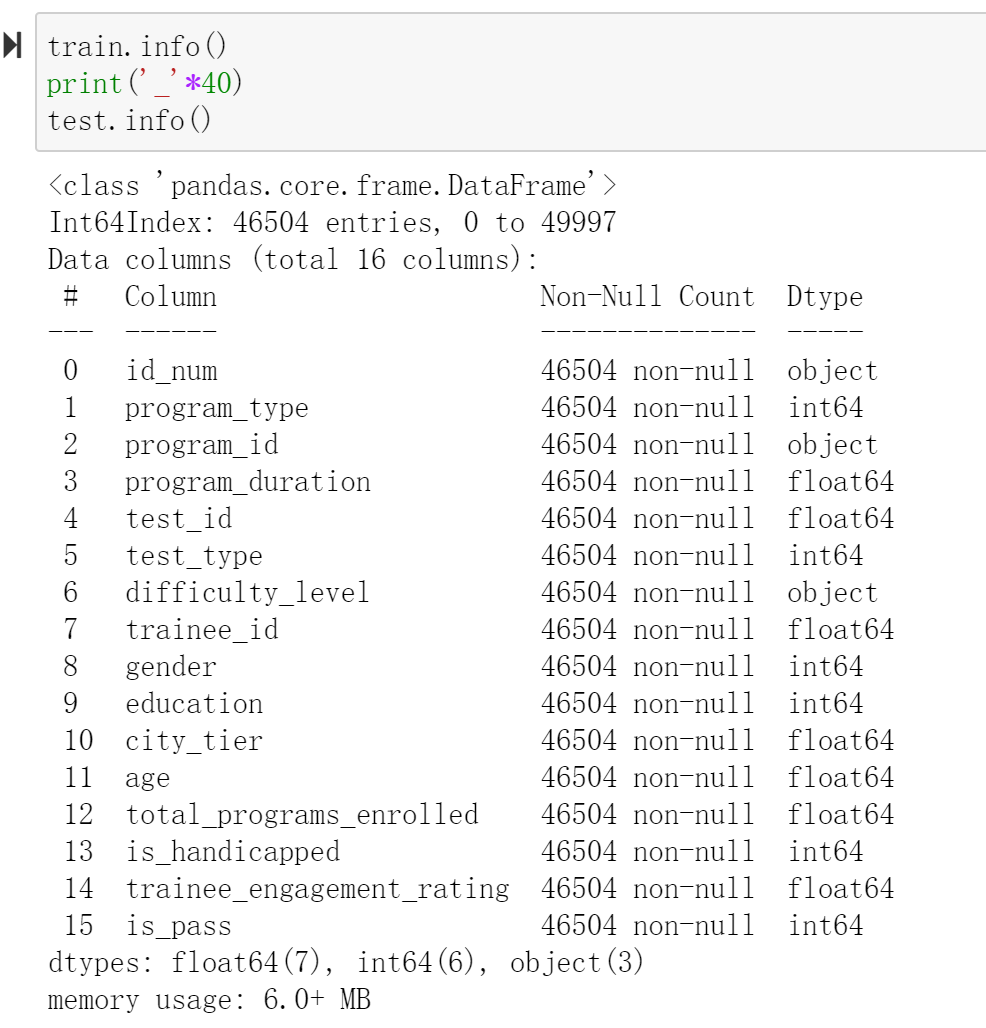
可能后来还会再用来分析，不过我可能不会。

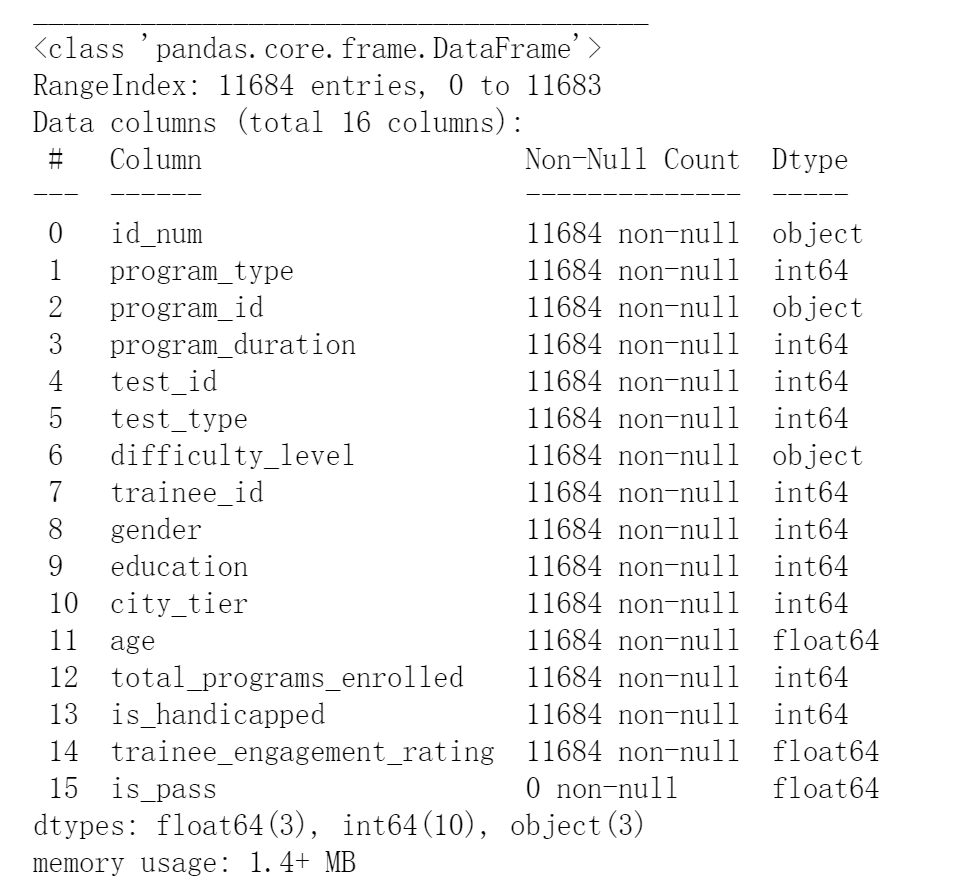
认识数据

数据整体认知

研究**训练集、测试集、记录数、变量数、变量类型、变量属性值、标签**等内容。



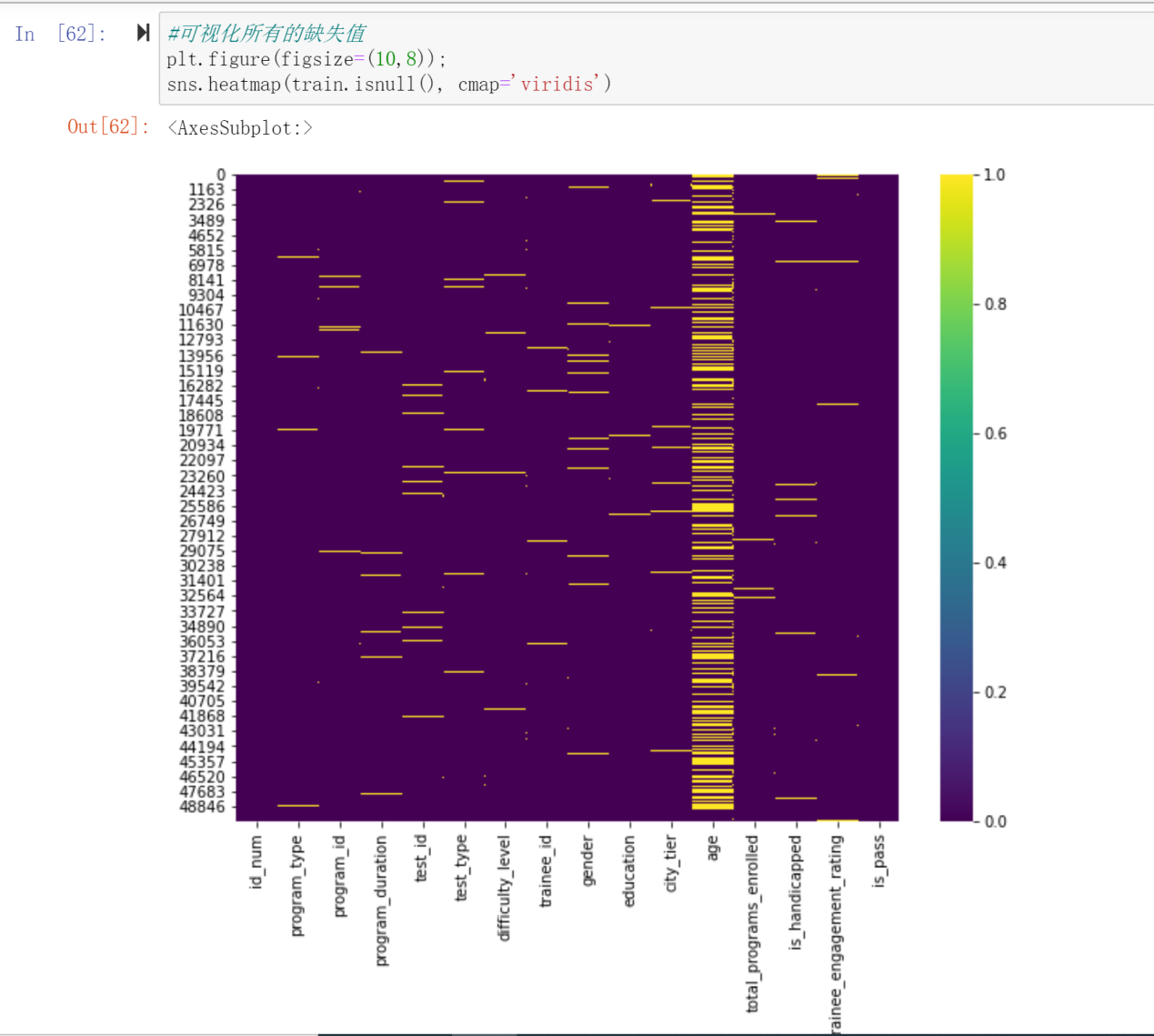




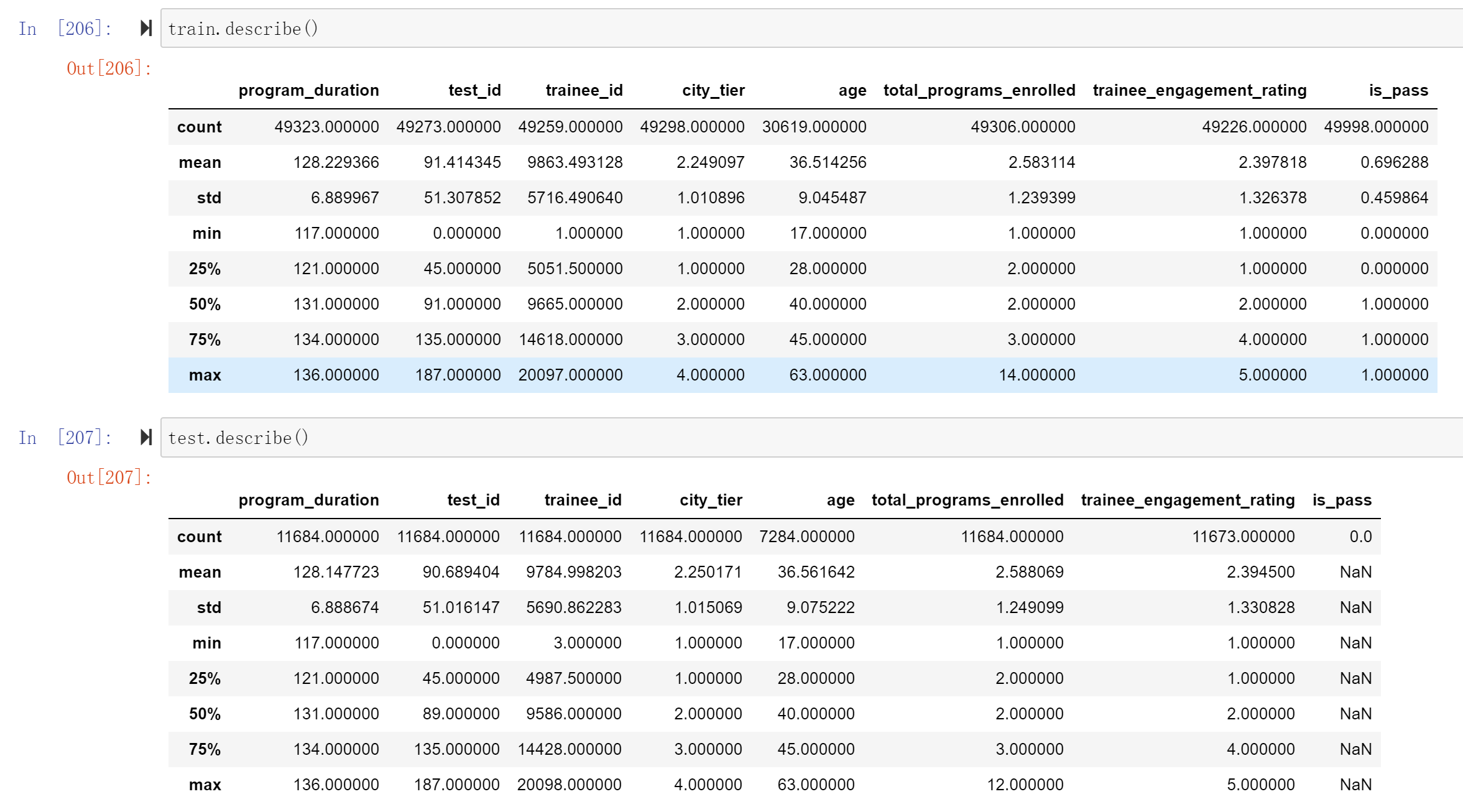
数据质量分析

分析数据的缺失值,异常值等特性。





检查有无异常值



可见没有异常值

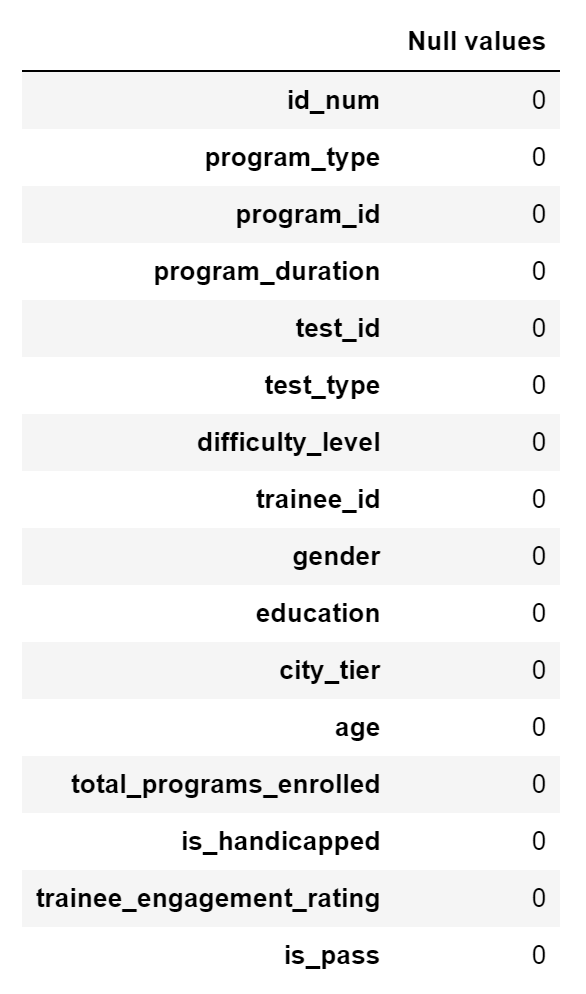
经过大约计算在50000份数据里800份数据大概占比0.016，所以difficulty\_level、education、city\_tier、test\_type、program\_duration、total\_programs\_enrolled、is\_handicapped、trainee\_engagement\_rating这些列的缺失值用众数填充；

age缺失值较多，用平均值填充

其他不太好暴力填充的缺失值，则直接删除。

测试集采用相同的办法。

查看一下是否还有缺失值



反思：这里的数据处理过于简单，比如age是缺失值最多的，所以保证一定的缺失值填充准确率是非常重要的，对结果也会产生较大的影响。如果我能使用回归，随机森林模型预测缺失值则会很大地提高填充数地准确度，下一次做数据处理的时候，希望我可以运用到。

数据统计量分析

【**数据探索】**

**对比**，对比训练集不同样本之间的特征分布

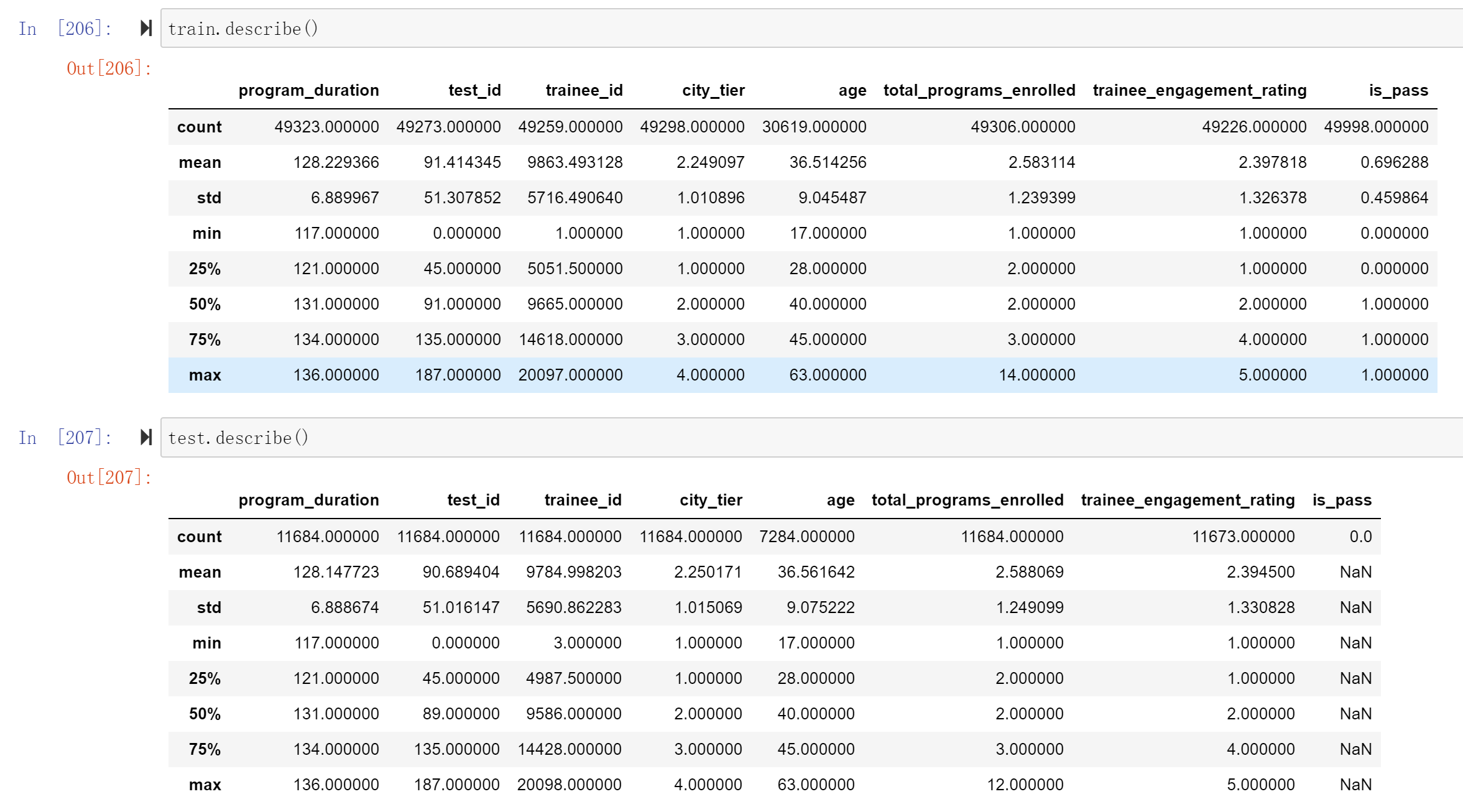
**分组**，到按类别标签、某个离散变量的不同取值groupby后的sum、unique。

**抓大放小**，就是**对于那些特征重要性较高的变量，要做重点分析**。因为这些变量对你模型预测能力的影响是较大的。

**可视化，**图形给人的冲击力往往是要大于数字本身的。

1. 对单个变量的统计分析

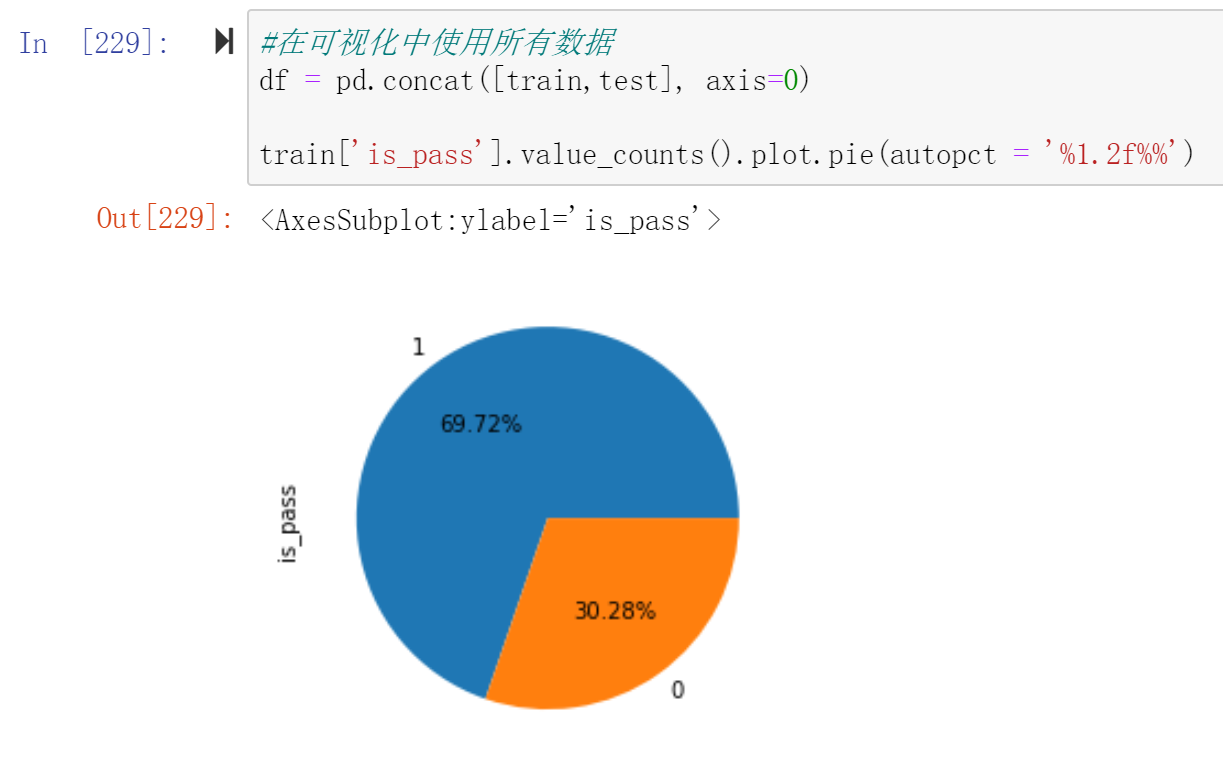
考察单个变量的均值、中位数、众数、分位数、方差、变异系数等。

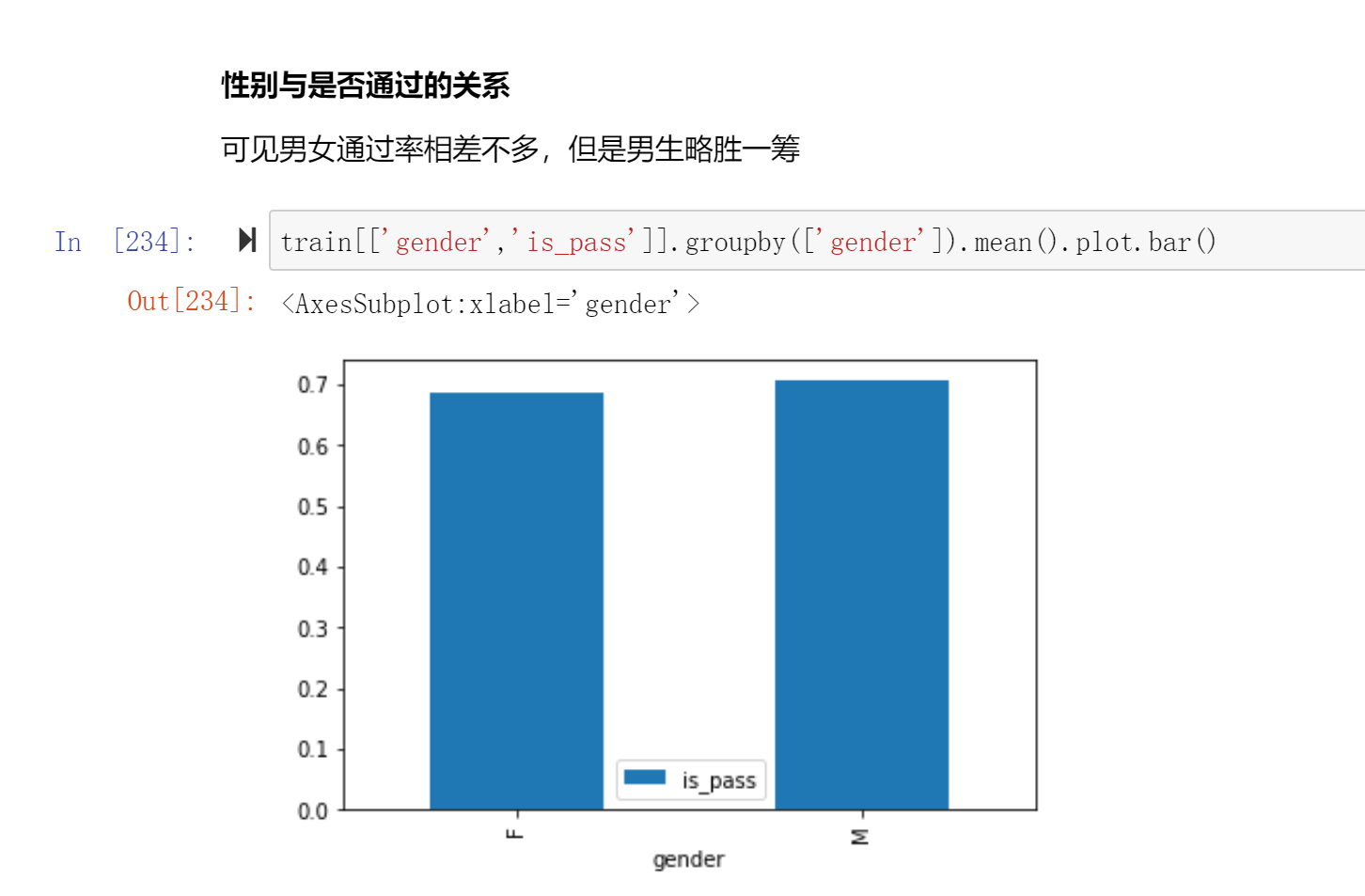


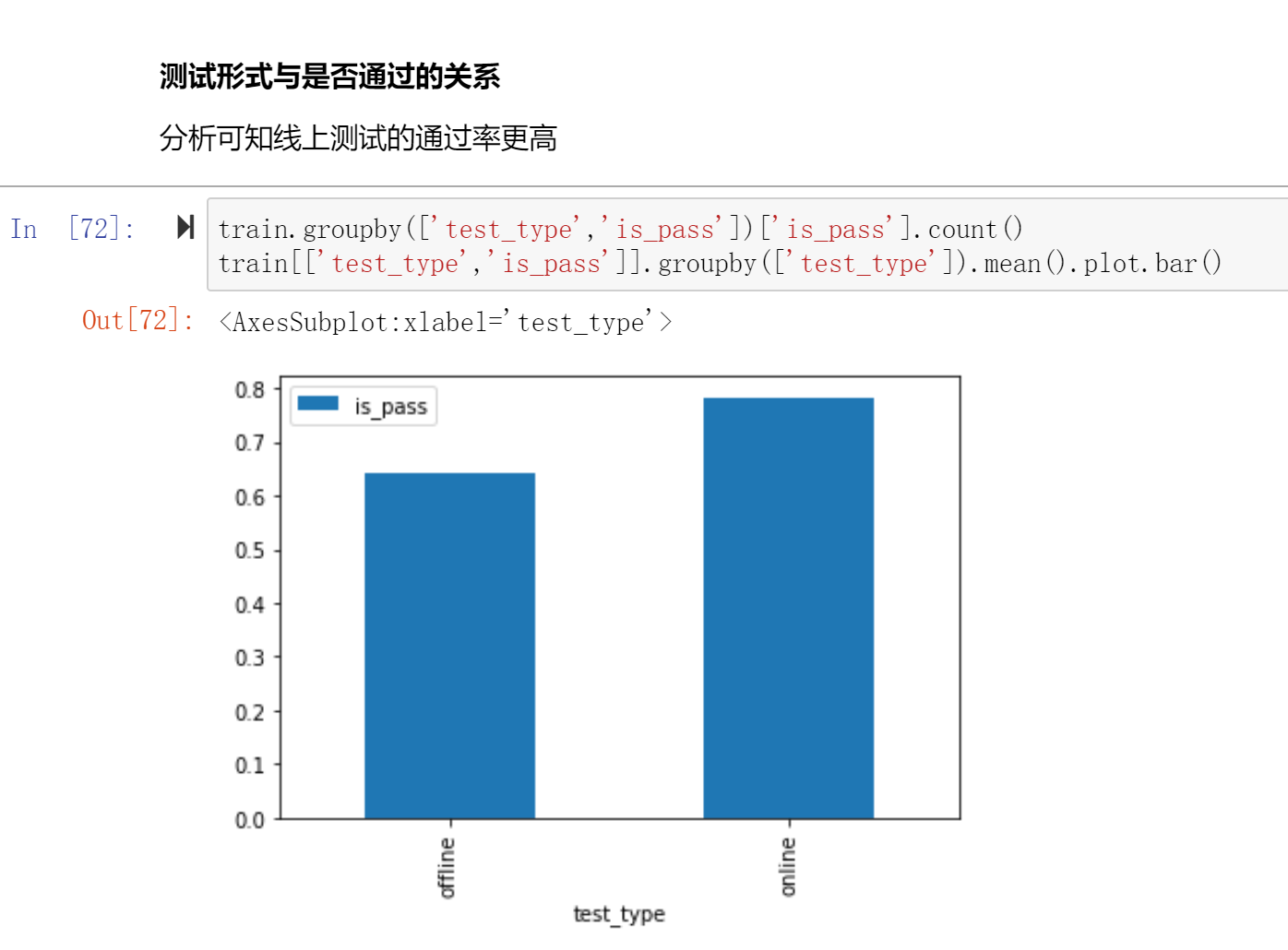
1. 两个变量的统计分析

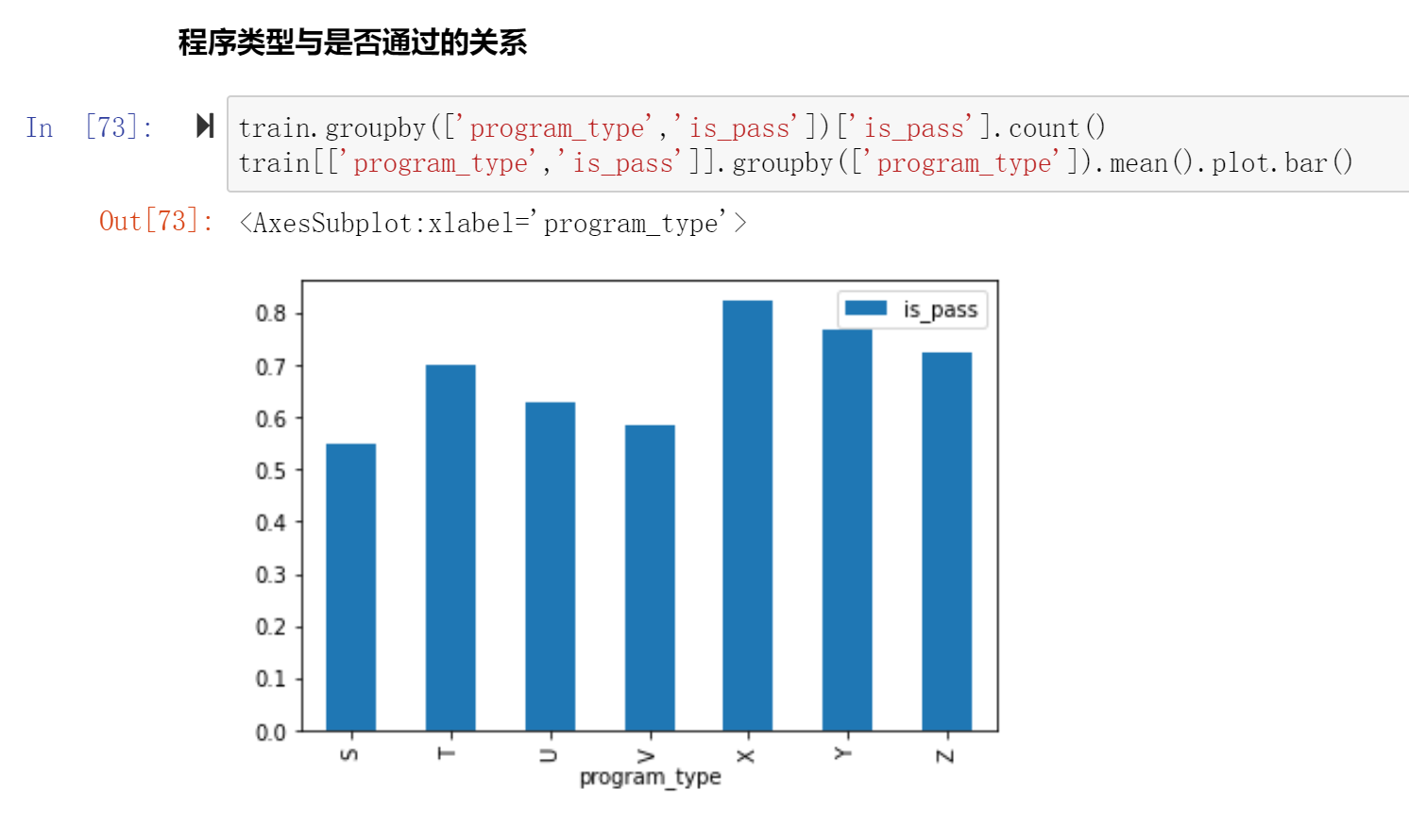
两个变量统计分布之间的关系。

先看看训练集中地通过率

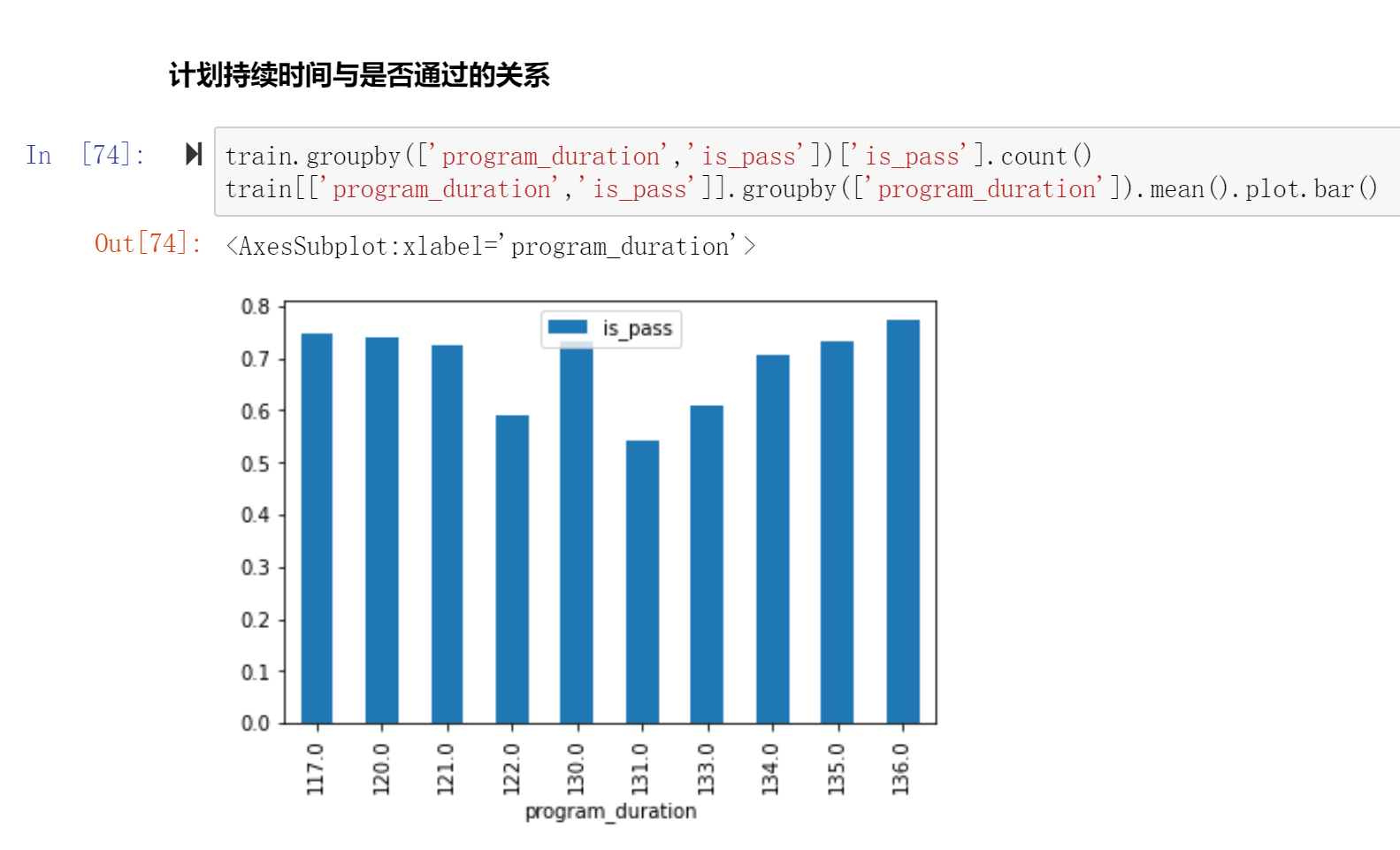




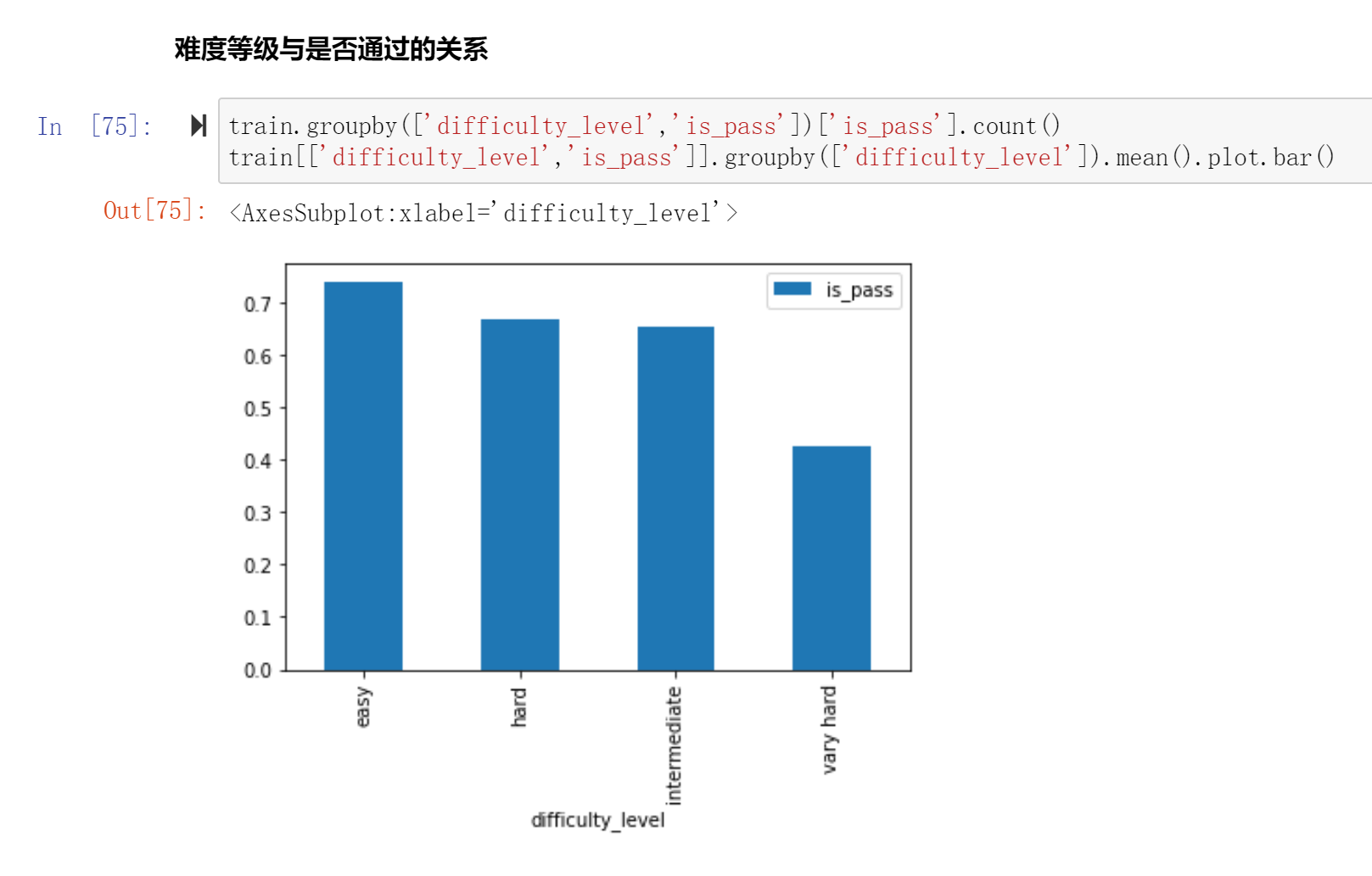




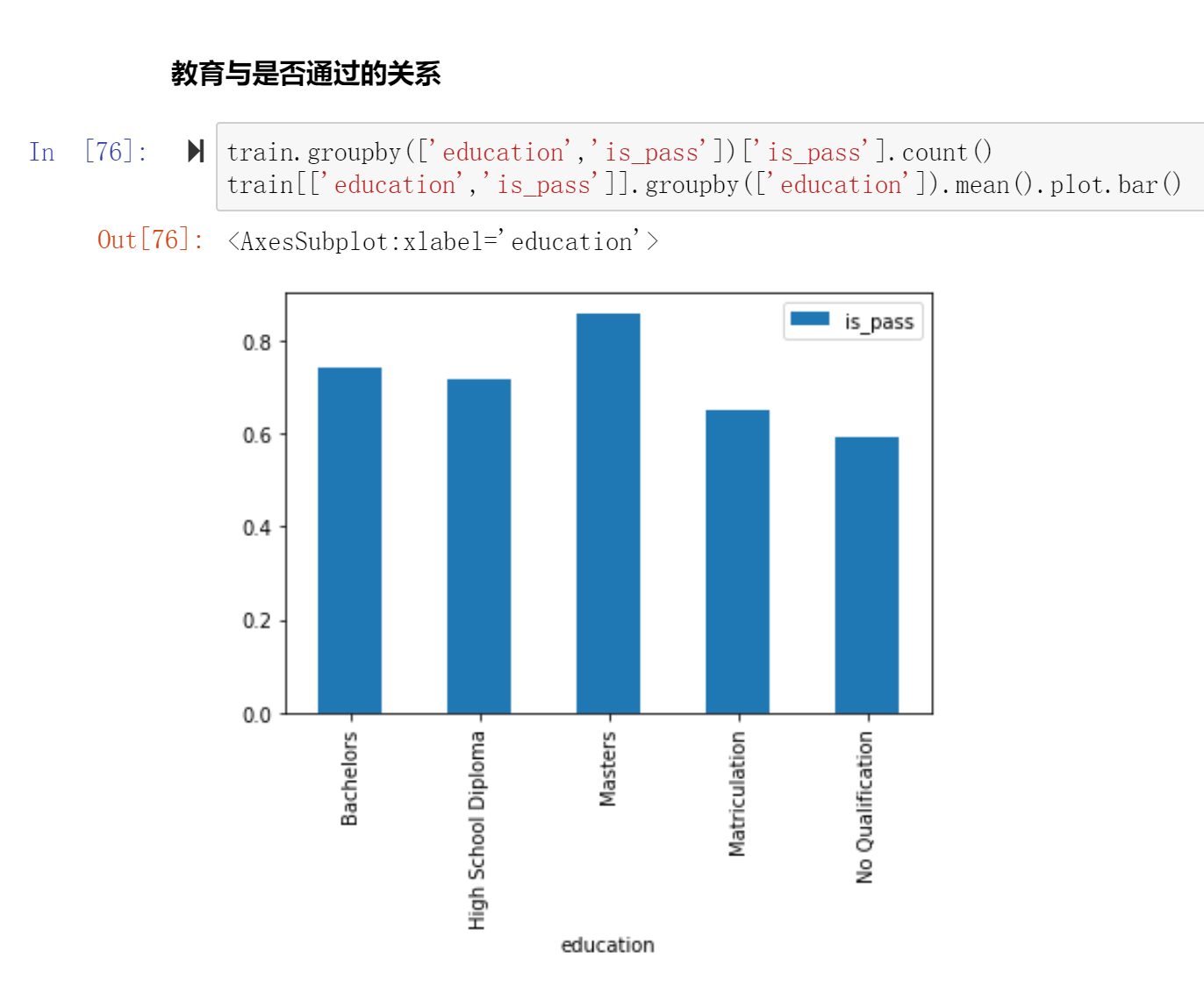
程序类型时X的通过率最高，程序类型是S的通过率最低



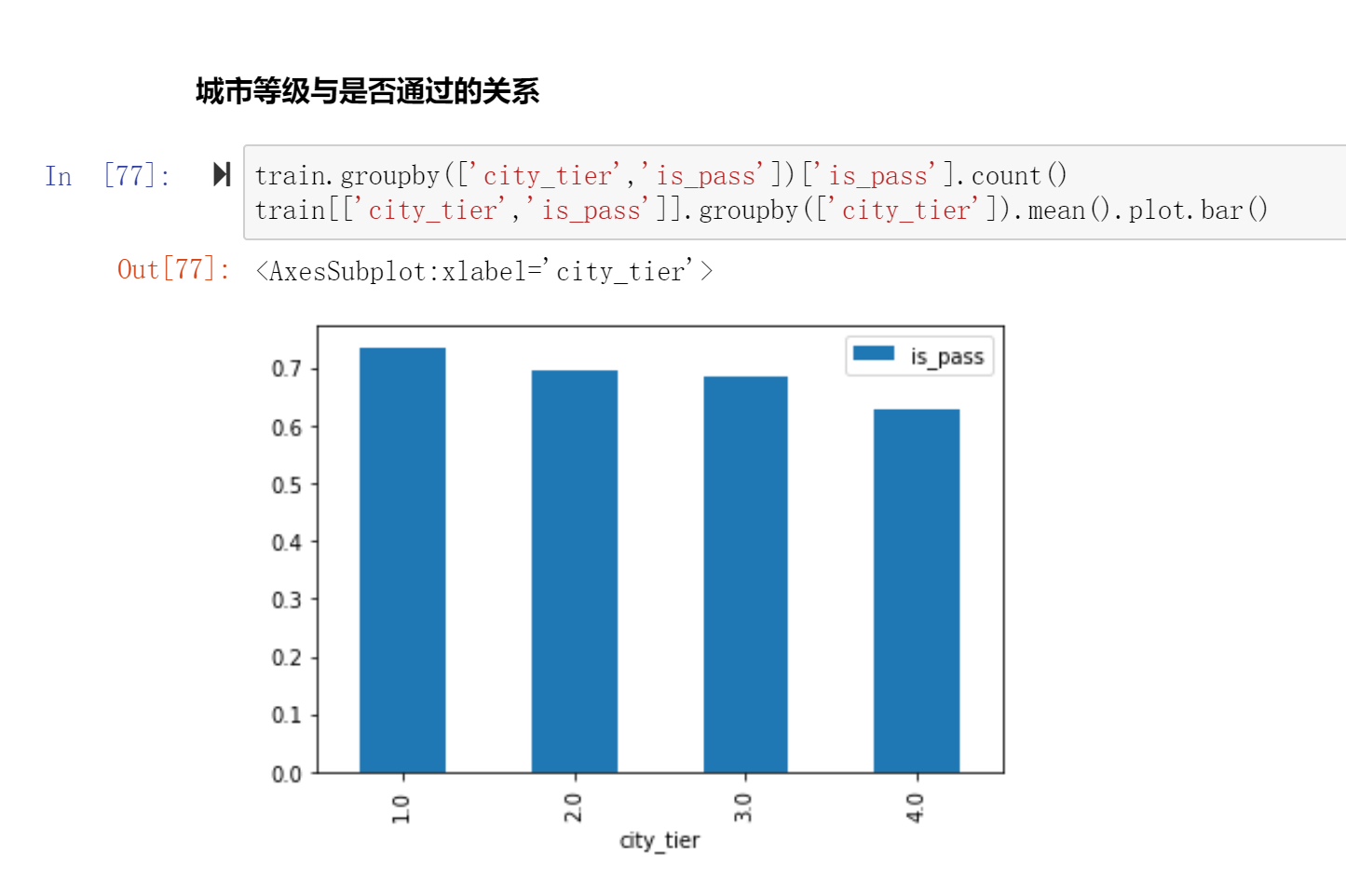
计划时间136的通过率最高，计划持续时间为131的通过率最低



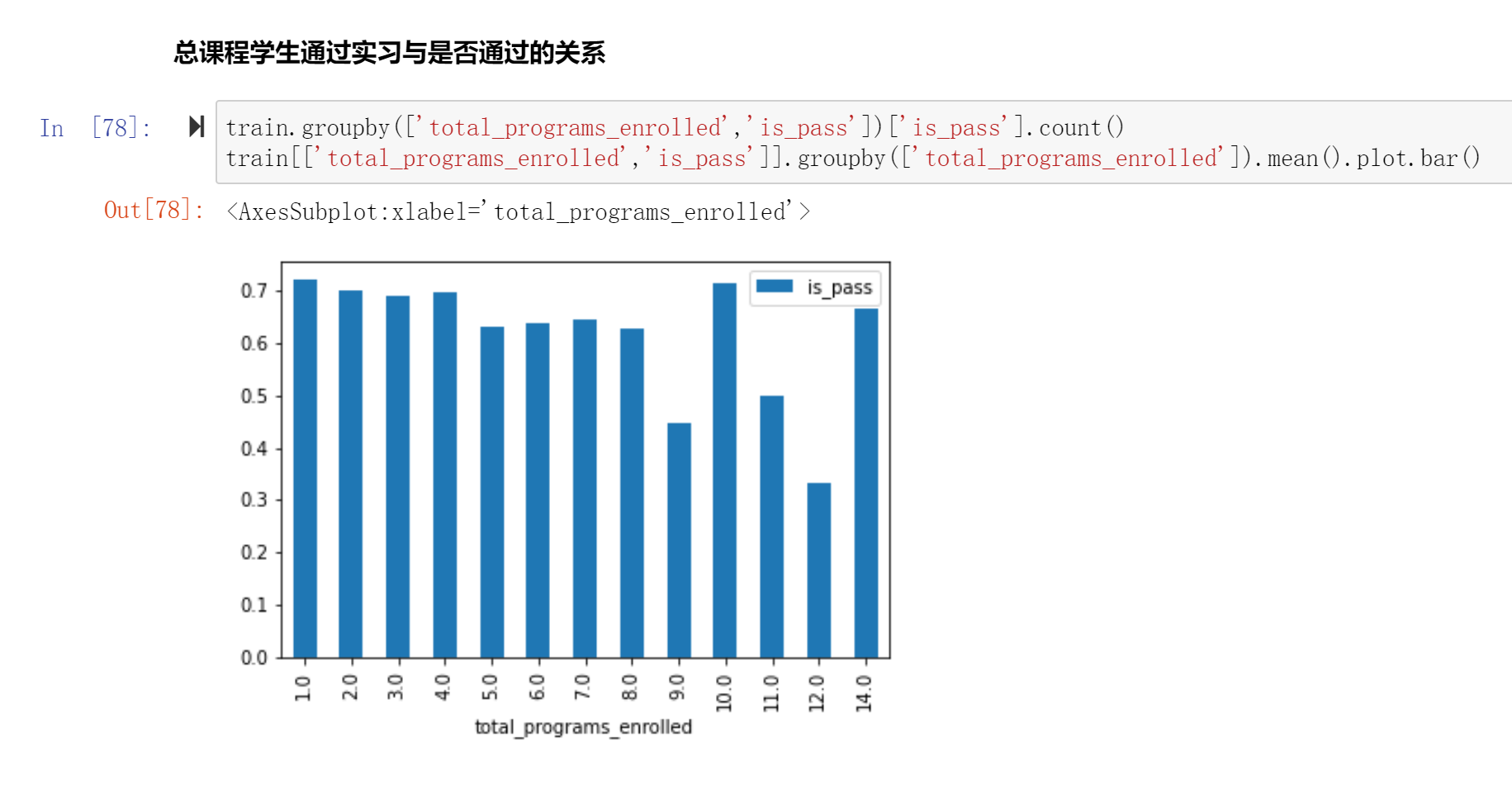
难度等级与通过率成反比例关系



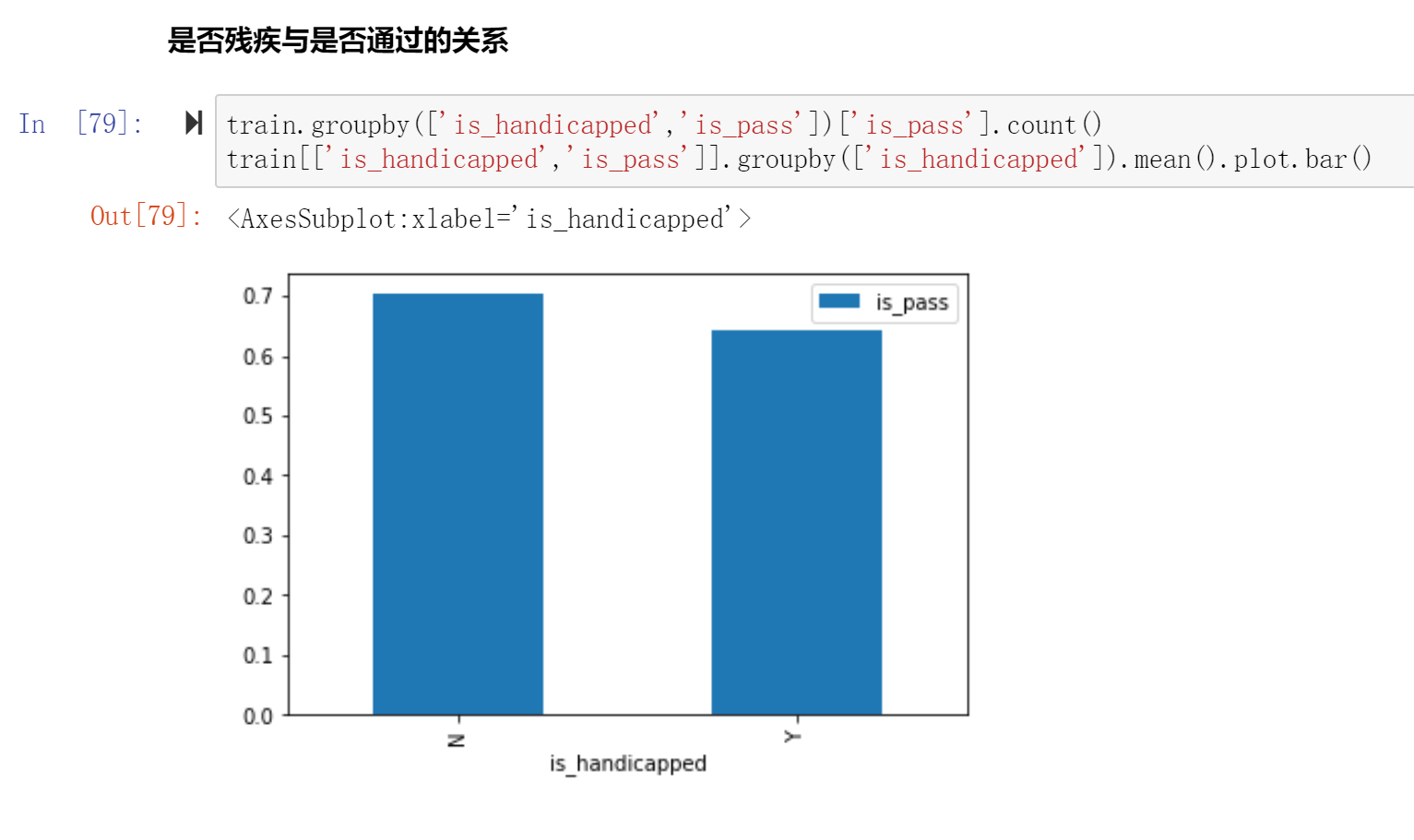
所受教育程度与通过率成正比



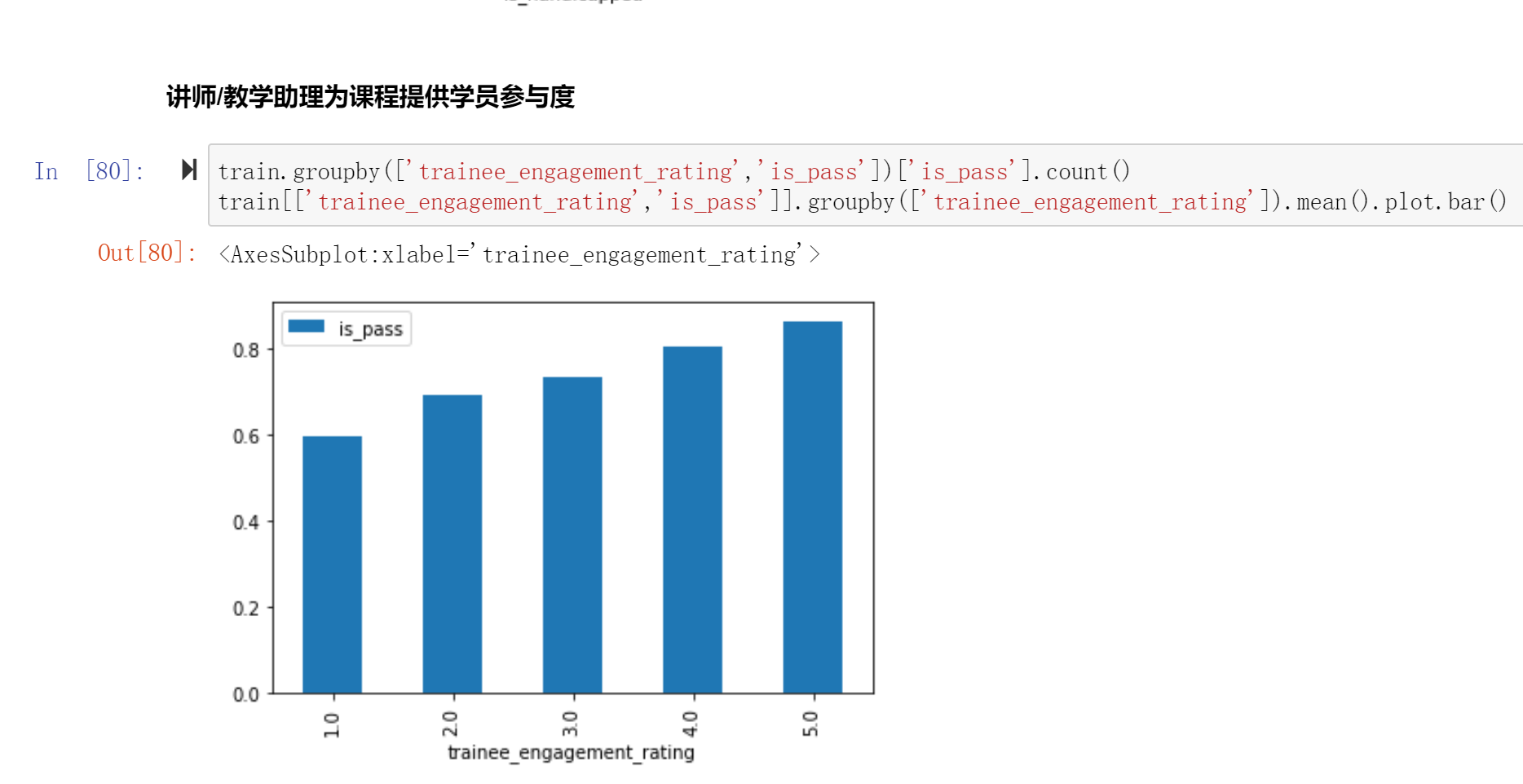
城市等级与通过率成正比



总课程是12的通过率最低

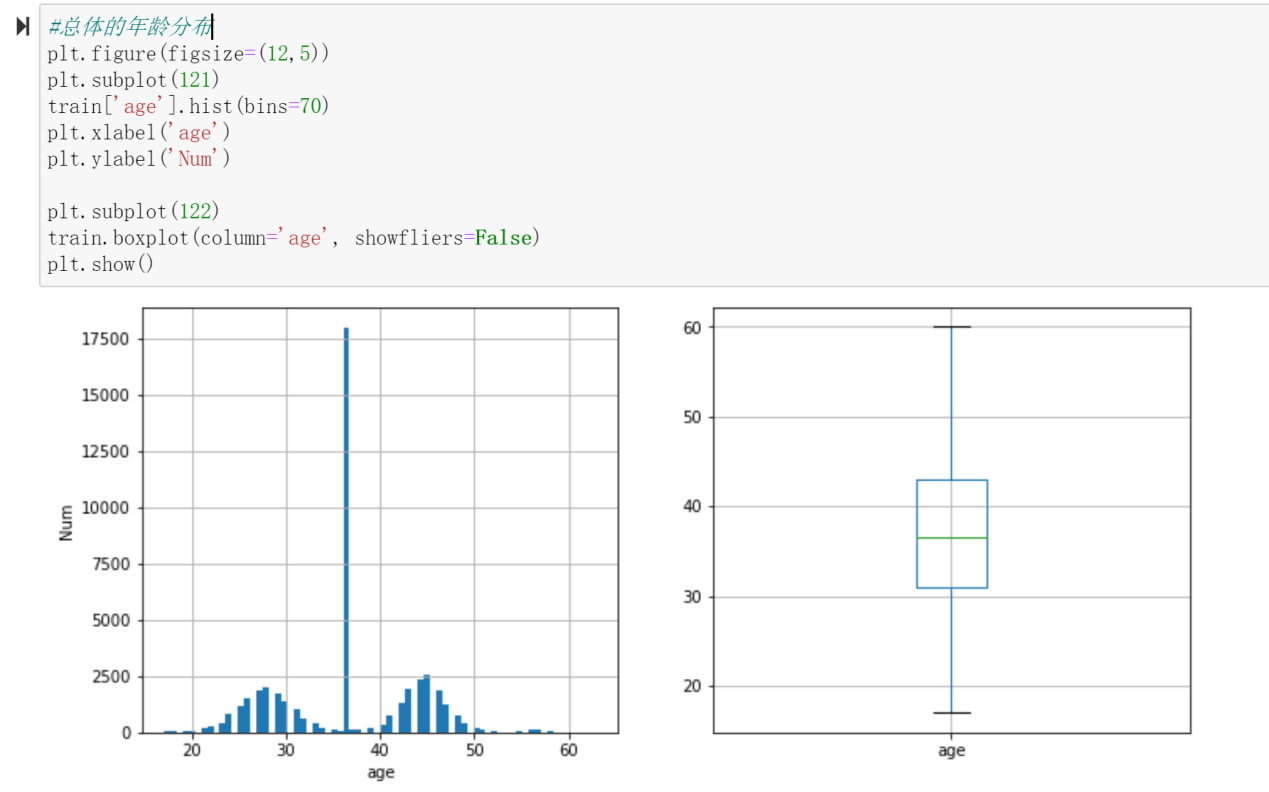


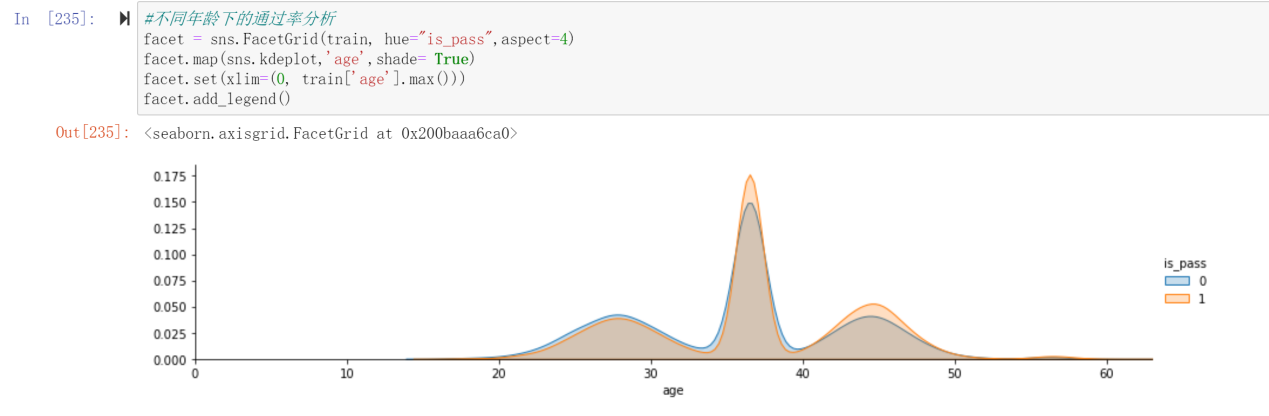
不残疾的通过率较高于残疾人士，不过差别也不明显



讲师/教学助理为课程提供学员参与度与学生通过率成正比

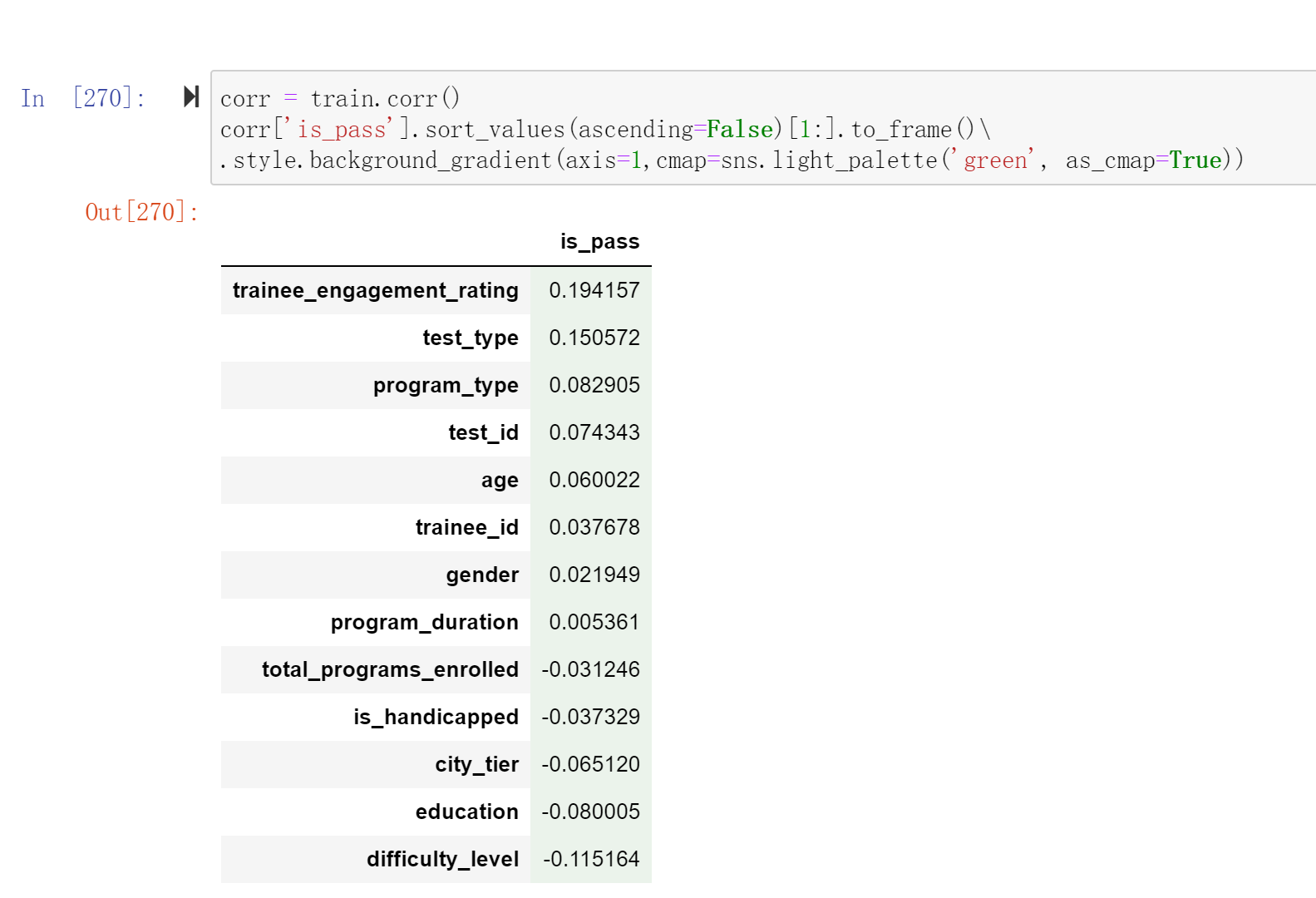
对年龄的分析



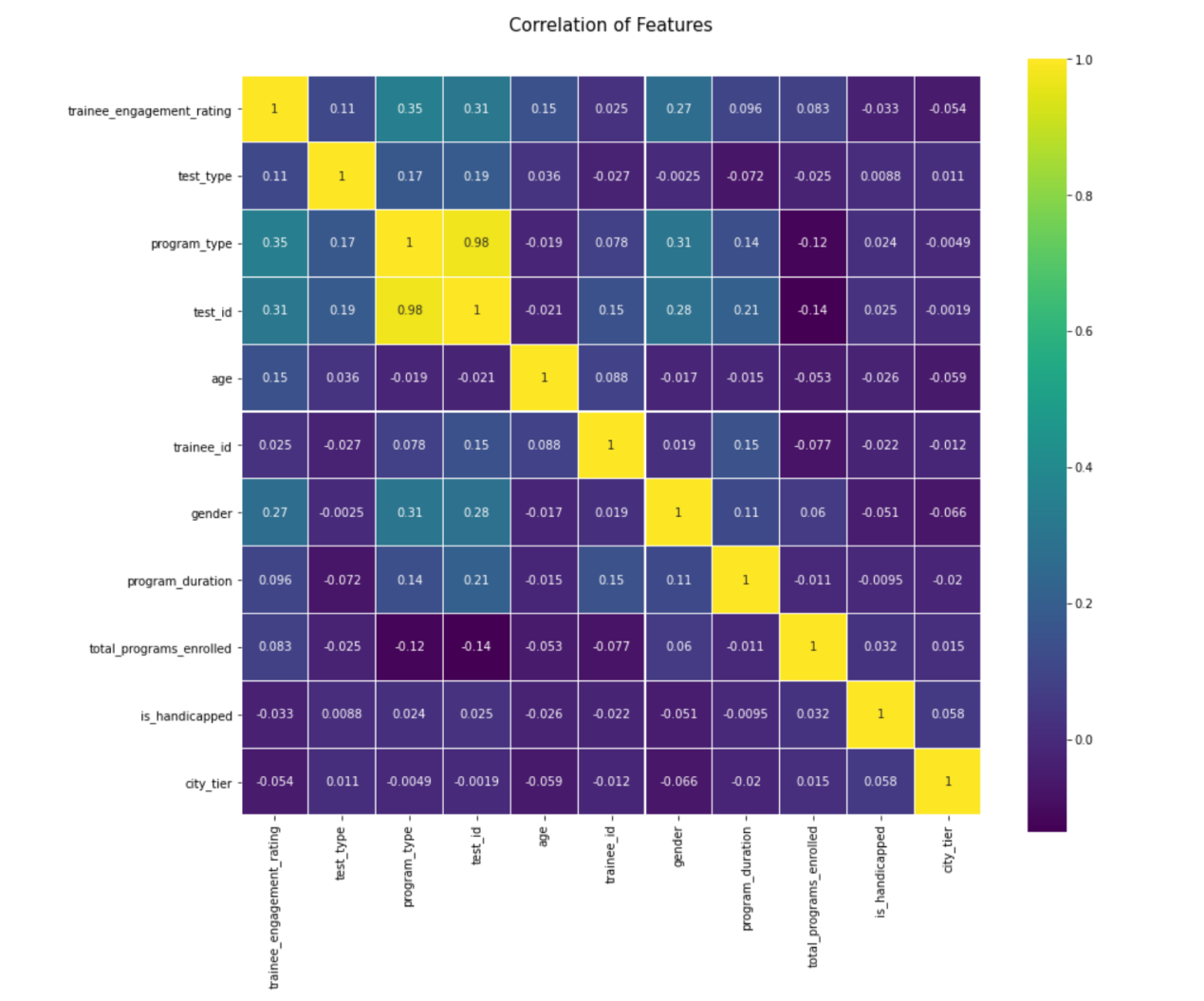


1. 对多个变量的统计分析

先看看相关性分析



可视化

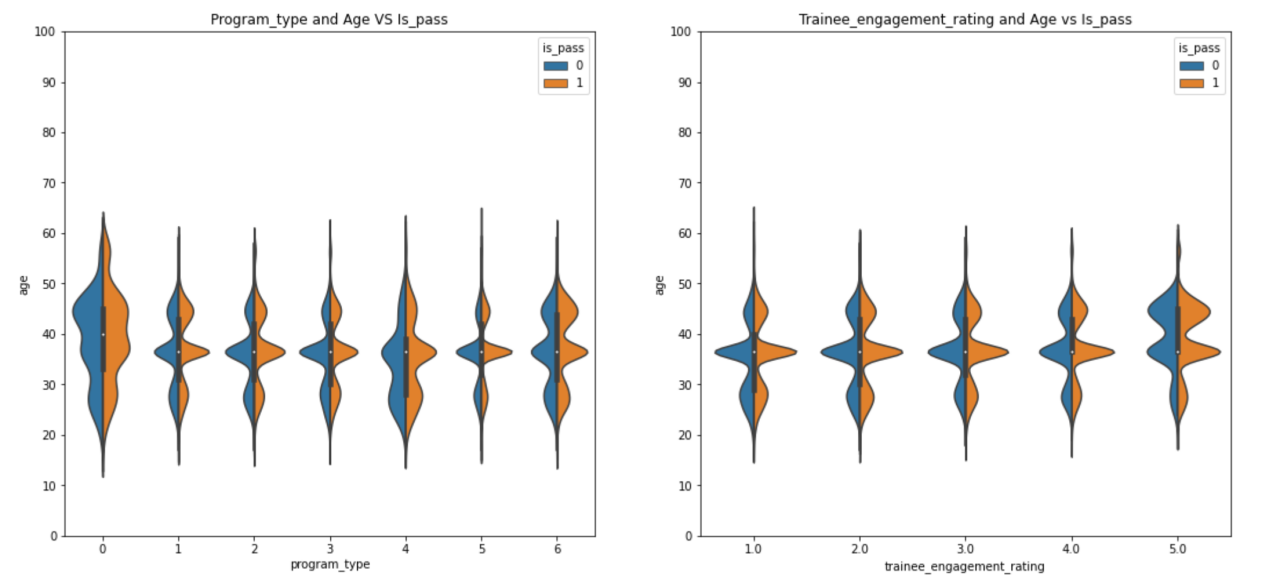


从图中可以看出

皮尔逊相关图可以告诉我的一件事是，没有太多的特征彼此之间具有很强的相关性。这意味着训练集中没有太多冗余或多余的数据，并且每个功能都带有一些独特的信息。

以下是两个最相关的列：讲师/教学助理为课程提供学员参与度和测试形式

根据相关性做出的多变量之间的关系



反思：多变量之间的相关性可视化还不够。

### 特征工程

做数据竞赛时最重要的一项内容。

1. 输入机器学习模型的数据必须是标准的向量形式。但处理的数据并没有以格式规范的特征向量的形式呈现。呈现的数据是数据库记录、字母、文字等形式，同时还存着在大量的噪声数据。所以需要一定的方法把非结构化的数据转化为结构化的数据。

#### 数据预处理

a. 海量原始数据中存在大量信息缺失、不一致、冗余值、异常值等，会影响模型的学习效果。

b. 在用各种模型算法时的监督学习的假设，训练集和测试集样本是独立同分布的。

c. 在模型训练时，数据规范化的操作可以让梯度下降算法收敛得更快，更快地找到最优超参数。

数据转换

1. 首先把数据型的列放在一个列表中，把数据型的列放在另一个列表中
2. 先处理数据型: 归一化

做聚类分析的时候，聚类的效果往往特别受其中一列数据的影响，使得原本应该散布在二维平面图上的点，变成聚集在一条线上的点。所以为了避免这个情况，采用数据归一化处理，使数值落在【0，1】之间。

c. 处理非数据型：LabelEncoder OneHotEncoder

LabelEncoder：是用来对分类型特征值进行编码，即对不连续的数值或文本进行编码。

OneHotEncoder：对于这个样本，如[" M "，"vary hard "，" N"]，我们需要将这个分类值的特征数字化，最直接的方法，我们可以采用序列化的方式：[0,1,3]。但是这样的特征处理并不能直接放入机器学习算法中。【这里说个题外话，哪个师兄写的数据集？？？我从小到大都没看到过有人把very 能写成vary，直到这次，直接去世（）】

    对于上述的问题，性别的属性是二维的，困难等级是四维的，是否残疾也是二维的，这样，采用One-Hot编码的方式对上述的样本“[" M "，"vary hard "，" N"]”编码，“M”则对应着[1，0]，同理“vary hard”对应着[0，0，0，1]，“N”对应着[0, 1]。则完整的特征数字化的结果为：[1,0,0,0,0,1,0,1]。

反思和改进：其实数据预处理还包含了很多的部分，数据清洗部分：缺失值的插补有些可以利用最近邻插补，离群值可以采取替换，或者是直接删除，异常，重复的数据直接删除。我首先只考虑到了处理缺失值，这次是比较幸运没有遇到有个性的数据。下一次做一个比赛的时候，希望自己考虑的更加周全。这一次做的比较好的：是数据变换和数据规范化。

# **第二阶段**

### 准备用于学习的数据，使用模型预测

这一次的数据预测的特征工程部分我只做了数据的预处理，然后直接通过相关性，探究变量与变量之间的相关性，手动筛选特征，尝试了五种不同的特征组合，尝试了

（1）模型选择

最开始的尝试：使用简单线性回归，knn，逻辑回归和XGBoost单个模型来预测，但是效果都不太好。然后学会了一个小技巧，交叉验证：

就简单说一下，如果将数据集分为10折，做一次交叉验证，实际上它是计算了十次，将每一折都当做一次测试集，其余九折当做训练集，这样循环十次。通过传入的模型，训练十次，最后将十次结果求平均值。将每个数据集都算一次，然后再用打印出返回值。可以比较方便的知道那哪一个模型预测得更好。然后直接上传最好的那一份submission。

交叉验证优点：

1：交叉验证用于评估模型的预测性能，尤其是训练好的模型在新数据上的表现，可以在一定程度上减小过拟合。

2：还可以从有限的数据中获取尽可能多的有效信息。

（2）模型优化

优化参数

在机器学习模型中，需要人工选择的参数称为超参数。比如随机森林中决策树的个数，人工神经网络模型中隐藏层层数和每层的节点个数等等，需要事先指定。超参数选择不恰当，就会出现欠拟合或者过拟合的问题。而在选择超参数的时候，有两个途径，一个是凭经验微调（很显然我是没的经验的），另一个就是选择不同大小的参数，带入模型中，挑选表现最好的参数。

　　微调的一种方法是手工调制超参数，直到找到一个好的超参数组合，这么做的话会非常冗长，所以我使用了Scikit-Learn的GridSearchCV来做这项搜索工作。

GridSearchCV 网格搜索和交叉验证。网格搜索，搜索的是参数，即在指定的参数范围内，按步长依次调整参数，利用调整的参数训练学习器，从所有的参数中找到在验证集上精度最高的参数，是一个训练和比较的过程。

　　GridSearch 穷举搜索：在所有候选的参数选择中，通过循环遍历，尝试每一种可能性，表现最好的参数就是最终的结果。其原理就像是在数组里找到最大值。这种方法的主要缺点是比较耗时！

就比如我下面需要找到最适宜的决策树的个数以及最适宜的决策树的最大深度。 我首先设置的n\_estimators参数择优的范围是:1~101，步长为10。太慢了。

实现过程：

sklearn 根据param\_grid的值，首先会评估n\_estimators和max\_features的组合方式，接下来在会在bootstrap=False的情况下（默认该值为True），评估n\_estimators和max\_features的组合方式，每一种组合方式要在训练集上训练10次，训练结束后，通过best\_params\_获得最好的组合方式。我得到的最好的参数是n\_estimators：80；max\_features：但是竟然发现还没有默认参数预测得高。实在是慢极了，只尝试了一次，

刚刚说了那么那么多，实际上预测的增幅仍然小极了。超级 big boss 即将登场。

这里先给大 boss 做个“美美”的铺垫。如下

（3）集成学习（Model Ensemble）

常见的模型融合方法有：Bagging、Boosting、Stacking、Blending。

这一次使用的是Bagging

Bagging全称bootstrap aggregating。这种方法，把个体预测器当做黑盒子处理，不进行进一步修改，所以，它的个体预测器可以是任何机器学习算法。它对训练数据集进行随机取样，并使用取样后的数据子集，训练每一个个体预测器。在预测时，每一个预测器都会做出预测，整体结果则是每个预测的平均。

Bagging能降低模型的variance（方差），所以它可以对抗过拟合。一般而言，bagging需要使用variance高、比较复杂的强个体模型。具体到常用的决策树上，就体现在单个决策树的层数更深。（boosting需要使用bias（偏差）高、比较简单的弱个体模型。）

1. **应用**
2. **总结**