**Python数据处理编程**

**实验报告**

****

|  |  |
| --- | --- |
| **姓名** |  |
| **学号** |  |
| **班级** |  |
| **专业** |  |
| **年级** |  |

目录

[一、 问题描述](#_Toc6127_WPSOffice_Level1) [1](#_Toc6127_WPSOffice_Level1)

[1.1 实验目的](#_Toc6127_WPSOffice_Level2) [1](#_Toc6127_WPSOffice_Level2)

[1.2 实验开发环境和工具](#_Toc5360_WPSOffice_Level2) [1](#_Toc5360_WPSOffice_Level2)

[1.3 实验内容](#_Toc27459_WPSOffice_Level2) [1](#_Toc27459_WPSOffice_Level2)

[二、 需求和规格说明](#_Toc5360_WPSOffice_Level1) [1](#_Toc5360_WPSOffice_Level1)

[三、概要设计](#_Toc27459_WPSOffice_Level1) [10](#_Toc27459_WPSOffice_Level1)

[四、详细设计](#_Toc25971_WPSOffice_Level1) [14](#_Toc25971_WPSOffice_Level1)

[4.1设计思想](#_Toc19683_WPSOffice_Level2) [14](#_Toc19683_WPSOffice_Level2)

[五、结果分析：](#_Toc4055_WPSOffice_Level1) [26](#_Toc4055_WPSOffice_Level1)

[5.1实现注释程序清单](#_Toc10628_WPSOffice_Level2) [26](#_Toc10628_WPSOffice_Level2)

[5.2结果分析](#_Toc20659_WPSOffice_Level2) [28](#_Toc20659_WPSOffice_Level2)

[六、调试及总结](#_Toc20218_WPSOffice_Level1) [29](#_Toc20218_WPSOffice_Level1)

## 实验三 员工离职调查报告的数据清洗与分析

1. **问题描述** 
   1. **实验目的**

通过本项目将Pandas库中的一些函数知识应用在实践案例中，需要熟悉和掌握的函数包括：可用来清理字符串类型的矢量化的字符串方法；数据格式转换函数apply、map()、applymap()等；删除缺失值或不需要值的函数fillna()、dropna()、drop()；重塑数据函数melt()；合并数据函数concat() 和merge()。

* 1. **实验开发环境和工具**

可以在Windows或Linux操作系统上搭建开发环境，使用Anaconda科学计算包安装Python 3，里面会包含大多数科研常用的库。.ipynb文件推荐使用Anaconda内置的web版Jupyter Notebook。

* 1. **实验内容**

在本实验中，我们将对澳大利亚昆士兰教育培训就业部（DETE）和职业技术教育学院（TAFE）的员工进行离职调查。你可以在这里找到TAFE离职调查和这里的DETE调查。

TAFE的离职报告数据：

<https://data.gov.au/dataset/ds-qld-89970a3b-182b-41ea-aea2-6f9f17b5907e/details?q=exit%20survey>

DETE的离职报告数据：

h[ttps://data.gov.au/dataset/ds-qld-fe96ff30-d157-4a81-851d-215f2a0fe26d/details?q=exit%20survey](https://data.gov.au/dataset/ds-qld-fe96ff30-d157-4a81-851d-215f2a0fe26d/details?q=exit%20survey)

我们将这些数据集进行了一些细微的修改，保证它们更易于使用。

修改内容包括：将编码格式更改为UTF-8（原来的编码格式为cp1252）

1. **需求和规格说明**

**1.了解项目和数据集：**

本任务将主要按照如下内容展开分析：

* 那些只在研究院工作了很短一段时间的员工是否是由于存在某些不满而辞职?工作的更久一点的员工是否因存在不满而辞职？
* 年轻员工辞职是因为存在某种不满吗?老员工辞职是因为存在某种不满吗?

我们应当能够对这两个调查数据的结果进行综合考评，然后再回答上面的问题。我们的目标：完成大部分数据清理工作，并保证你可以开始分析第一个问题。

数据集中没有提供数据字典。因此对于这个项目，将使用我们的个人常识来定义列。下面是我们将在dete\_survey.csv中使用的一组列的解释:

* ID：用于识别调查参与者的id
* SeparationType：雇员雇佣关系终止的原因
* Cease Date：雇员结束工作的年或月
* DETE Start Date：雇员开始受雇于DETE的哪一年

下面是我们将在tafe\_survey.csv中使用的一组列的基本解释：

* Record ID: 用于标识调查参与者的id
* Reason for ceasing employment: 雇员雇佣关系终止的原因
* LengthofServiceOverall：学院总工作年限（单位：年）：员工的工作年限（单位：年）

具体编程任务如下：

1. 首先在markdown单元格中编写一段介绍项目和数据集的注释。
2. 导入pandas和NumPy库。
3. 通过pandas读取dete\_survey.csv文件，并赋值给变量dete\_survey。
4. 通过pandas读取tafe\_survey.csv文件，并赋值给变量tafe\_survey。
5. 使用DataFrame.info() 和DataFrame.head()方法查看这两个DataFrame的基本信息（打印前6行）。使用其他数据探索方法，比如：Series.value\_counts()和DataFrame.isnull()方法来查看数据并确定一些后续步骤。
   1. 另起markdown单元格，简要描述你的观察结果。

**2.识别缺失的数据和丢弃不必要的列：**

通常，确定下一步如何对数据进行清洗和重塑都是比较关键且最困难的部分。但也不用太过担担心，后续我们将会提供一些步骤，并留有一些您自己的空间进行拓展。

首先可以进行如下观察：

* dete\_survey中包含了值为"Not Stated"，这些值表示缺失值，但它们并不是NaN。
* dete\_survey和tafe\_survey中都包含许多我们不需要分析的列。
* 这两个dataframe都包含许多类似的列，但列名不同。
* 有多少个列/答案表示员工因不满而辞职。

首先，我们将处理两个问题。使用pd.read\_csv()确定应该表示为NaN的值。我们将首先对缺失值进行处理。然后，我们再删除那些在分析中不需要的列。

具体编程任务如下：

1. 通过pandas读取dete\_survey.csv文件，但是本次读取时，将值为"Not Stated"的地方都修改为"NaN"

* 可通过设置pd.read\_csv()方法中的参数na\_values为Not Stated实现
* 将数据读取结果赋值给变量dete\_survey

1. 通过pandas读取tafe\_survey.csv文件，但是本次读取时，将值为"Not Stated"的地方都修改为"NaN"

* 通过设置pd.read\_csv()方法中的参数na\_values为Not Stated实现
* 将数据读取结果赋值给变量tafe\_survey

1. 然后，删除dataframe中一些不需要的列，保证dataframe更易于使用

* 使用dataframe .drop()方法从dete\_survey中删除以下列:

dete\_survey.columns[28:49]，请将删除维度参数axis设置为1

* + 将删除后的结果赋值给变量dete\_survey\_updated
* 使用dataframe .drop()方法从tete\_survey中删除以下列:

tete\_survey.columns[17:66]，请将删除维度参数axis设置为1

* + 将删除后的结果赋值给变量tafe\_survey\_updated

1. 新建markdown单元格，阐述你对数据集所做的改变，以及原因

**3.清洗列名：**

接下来，让我们将注意力转向列名。这两个dataframe包含许多类似的列，但是列名不同。下面是一些想用于最终分析的列：

|  |  |  |
| --- | --- | --- |
| dete\_survey | tafe\_survey | Definition |
| ID | Record ID | 用于识别调查参与者的id |
| SeparationType | Reason for ceasing employment | 参与者终止雇佣关系的原因 |
| Cease Date | CESSATION YEAR | 参与者就业结束的年份或月份 |
| DETE Start Date |  | 参与者开始受雇于DETE的年份 |
|  | LengthofServiceOverall. Overall Length of Service at Institute (in years) | 参与者的工作年限（年） |
| Age | CurrentAge. Current Age | 参与者的年龄 |
| Gender | Gender. What is your Gender? | 参与者的性别 |

因为最终想要合并这两个DataFrame，所以必须对列名进行标准化。回想一下，可以使用DataFrame.columns以及向量化的字符串方法，以一次性更新所有列。下面是之前的一个实例：

|  |
| --- |
| happiness2017.columns =  happiness2017.columns.str.replace('.', ' ').str.replace('\s+', ' ').str.strip().str.upper() |

具体编程任务如下：

1. 对dete\_survey\_updated中的剩余列重命名

* 按照以下要求更新列名：
  + 所有大写字母均改为小写字母
  + 删除字符串结尾的所有空格。
  + 用下划线('\_')替换单词之间的空格。
* 例如：Cease Date应当更新为cease\_date
* 可以用DataFrame.columns属性，以一个数组的形式输出现有的列名。

1. 使用datafame.rename()方法更新tafe\_survey\_updated中的以下列。

|  |  |
| --- | --- |
| Record ID | id |
| CESSATION YEAR | cease\_date |
| Reason for ceasing employment | separationtype |
| Gender. What is your Gender? | gender |
| CurrentAge. Current Age | age |
| Employment Type. Employment Type | employment\_status |
| Classification. Classification | position |
| LengthofServiceOverall. Overall Length of Service at Institute (in years) | institute\_service |
| LengthofServiceCurrent. Length of Service at current workplace (in years) | role\_service |

1. 使用datafame.head()方法查看已更新的dete\_survey\_updated和tafe\_survey\_updated的更新情况，确保更新列名准确。
2. 新建markdown单元格，阐述你对数据集所做的改变，以及原因。

**5.滤除数据：**

在上一关中，对需要使用的列进行了重命名。后面则是删除不需要的数据。回想一下，我们的最终目标是回答以下问题：

* 那些只在研究院工作了很短一段时间的员工是否是由于存在某些不满而辞职?工作的更久一点的员工是否因存在不满而辞职？

如果我们查看这两个dataframe的separationtype列中的惟一值情况，可以看到这两个DataFrame都各自包含一组不同的离职类型。对于本项目，我们将只分析辞职的员工，所以他们的离职类型的值中会包含字符串'Resignation'。

如果想挑战难度，请尝试使用所有的离职类型来完成项目——这将发现在数据清洗过程中会有更多的问题需要解决。

注意，dete\_survey\_updated中包含多种包含字符串'Resignation'的离职原因类型，这些值中都包含字符串'Resignment'：

* Resignation-Other reasons
* Resignation-Other employer
* Resignation-Move overseas/interstate

我们必须考虑到每个变量，这样我们才不会无意中丢失数据！

注：无论进行什么操作，都请记住先对DataFrame进行副本保存，然后进行修改。

具体编程任务如下：

1. 使用Series.value\_counts()方法分别查看dete\_survey\_updated和tafe\_survey\_updated的separationtype列中每一种值的数量情况。
2. 在这两个dataframes中，筛选出辞职原因类型值中包含Resignation字符串的调查对象对应的数据。

* dete\_survey\_updated中有三种离职原因都包含Resignation字符串，这三种都需要考虑进来，作为离职原因为Resignation的数据。
  + 将dete\_survey\_updated中separationtype进行切片，拿到第一个单词并赋值回separationtype，从而使得三种都包含Resignation字符串统一为一种。
* 筛选dete\_survey\_updated 中separationtype列的值为Resignation的项，所查找出的结果通过DataFrame.copy()方法赋值给dete\_resignations
* 筛选tafe\_survey\_updated中 separationtype列的值为Resignation的项，所查找出的结果通过DataFrame.copy()方法赋值给tafe\_resignations

1. 新建markdown单元格，阐述你对数据集所做的改变，以及原因。

**6.验证数据：**

在开始清洗和操作其余的数据之前，我们还需要验证数据是否存在一些问题。当你处理真正的数据时，请小心，不要猜测你正在分析的数据肯定是未被污染的！

或许无法发现所有错误，但是通过确保数据在我们所知的范围内看起来是合理的，以免在进行一个数据分析项目时，这个项目最终因为糟糕的数据而变得毫无用处。

在本关，我们将重点检查“cease\_date”和“dete\_start\_date”列中的年份是否有意义。也鼓励同学们自行检查其他数据。

* 因为“cease\_date”是员工工作的最后一年，而“dete\_start\_date”是员工工作的第一年，因此在当前日期之后日期是没有意义的。
* 考虑到这个领域的大多数人都是在20多岁开始工作的，所以dete\_start\_date的值不太可能早于1940年。

如果数据中存在比当前年份更大或者比1940年的年份更小的数据，就应当人停止对数据的分析，因为这可能意味着数据有一些不合乎常理的地方。如果有少量的值过高或过低，那就直接删除这些数据。

具体编程任务如下：

1. 检查dataframe中的年份是否存在逻辑问题

* 首先，清洗dete\_resignations中的cease\_date列的数据
  + 使用Series.value\_counts()方法查看cease\_date列中的每一种值的数量情况
  + 使用向量化的字符串方法提取年份。
  + 使用Series.astype()方法将原类型转换为浮点型。
* 使用Series.value\_counts()查看dete\_resignations 中cease\_date 列和dete\_start\_date列的值；以及tafe\_resignations中cease\_date列的值。
  + 因为Series.value\_counts()返回一个series对象，所以可以使用Series.sort\_index()方法查看数量最多或是数量最少的值。通过修改参数ascending为True或是False即可设置升序或者为降序输出。
* 可以使用箱形图绘制任何数字列的值，以查看任何看起来存在问题的值。

1. 编写一个新的markdown单元格，阐述你在结果中的发现。

**7.创建新列:**

上一关，我们可以看出：

1. 年份数值不存在问题。
2. 两个DataFrame中的年份跨度不完全相同。因此需要放弃一些不需要分析的年份。

现在已经证实了dete\_resignations中年份是没有问题的，将使用它们创建一个新的列。回想一下，我们的最终目标是回答以下问题：

* 那些只在研究院工作了很短一段时间的员工是否是由于存在某些不满而辞职?工作的更久一点的员工是否因存在不满而辞职？

在人力资源部门有一个专门的称呼，员工在工作地点停留的时间被称为他们的工作年限。可以发现，tafe\_resignations中已经包含一个service列。将其重命名为institute\_ service。为了同时分析这两份调查报告数据，必须在dete\_resignations中创建一个相应的institute service列。

我们是否有数据可以用来计算员工在工作地点停留的时间?花一分钟再次回顾dete\_resignations的列，是存在两个关键的列可以计算的。

具体编程任务如下：

1. 在dete\_resignations中创建一个institute\_service列

* 用cease\_date列的值减去dete\_start\_date列的值，并将结果赋值给列institute\_service列。

1. 另起一个markdown单元格，阐述你对数据的改变，以及原因 。

**8.识别不满的雇员:**

上一关中，我们新建了一个institute\_service列，用于保存被调查者的工作入职时长（单位：年份）。接下来我们将会找出那些因为不满而辞职的员工。

我们将对dataframe中员工进行分类，

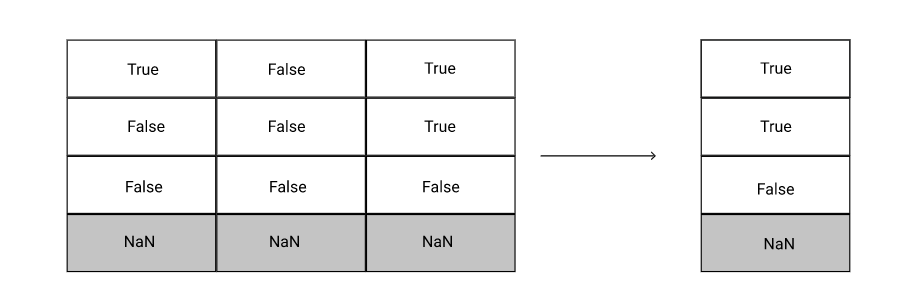
下面是我们将用于从dataframe中将员工分类为“不满意”的列。如果你不认可这种分类策略，可以自行修改！但是需要解释原因。

1. tafe\_survey\_updated：
   * + Contributing Factors. Dissatisfaction
     + Contributing Factors. Job Dissatisfaction
2. Dete\_survey\_updated：
   * + job\_dissatisfaction
     + dissatisfaction\_with\_the\_department
     + physical\_work\_environment
     + lack\_of\_recognition
     + lack\_of\_job\_security
     + work\_location
     + employment\_conditions
     + work\_life\_balance
     + workload

如果员工指出以上任何导致他们辞职的因素，则在新列中标记为dissatisfied。

创建新列需要执行以下操作：

1. 将tafe\_resignations中'Contributing Factors. Dissatisfaction'和'Contributing Factors. Job Dissatisfaction'这两列的值进行转换，转换为True、False或者NaN。
2. 如果以上列的名单中，任意一个列的值为True，则个新增的dissatisfied列填上True。将通过DataFrame.any()方法完成此工作，具体如下：
   * + 如果所选中的列的元素中有任意一个为True，返回True。
     + 如果所选中的列的元素中一个True都不存在，返回False。
     + 如果为NaN，则直接返回NaN。



DataFrmae.any()方法的用法为：

|  |
| --- |
| df.any(axis=1, skipna=False) |

详细链接可参考：

<https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.any.html>

经过以上操作之后，dissatisfied列必须只包含以下几个值：

* + - True：表示员工辞职是因为对工作不满
    - False：表示员工辞职不是因为对工作不满，而是其他原因
    - NaN：表示该值缺失

具体编程任务如下：：

1. 使用Series.value\_counts()方法查看tafe\_resignations中以下两个列的内容。
   1. Contributing Factors. Dissatisfaction
   2. Contributing Factors. Job Dissatisfaction
2. 将这两列的值更新为仅含True、False、NaN三种情况
   1. 声明一个叫做update\_vals的方法用于进行一下变换：
      1. 如果值为NaN，则返回np.nan（可以通过方法pd.isnull(val)判断一个值是否为NaN）
      2. 如果值为`-`, 返回False
      3. 如果是其他的值，则返回True
   2. 使用DataFrame.applymap()方法，将所声明的函数应用在上述两列中
      1. 传入update\_vals到方法applymap()中不需要括号
3. 使用df.any()方法在上述tafe\_resignations和dete\_resignations中分别创建一个名为dissatisfied的列
4. 使用df.copy()方法对结果进行备份，并将结果分别赋值为dete\_resignations\_up和tafe\_resignations\_up
5. 另起一个markdown单元格，阐述你对数据的改变，以及原因 。

**9.合并数据：**

对已经完成的任务进行如下总结：

1. 给列进行重命名
2. 删除我们分析中不需要的列
3. 检查了数据的正确性
4. 创建了一个新列institute\_service
5. 对Contributing Factors列的值进行了数据清洗
6. 创建一个新列，表明员工是否因某些方面的不满而辞职

现在，可以将数据集进行合并了。最终，需要根据institute service列将数据合并，因此在组合数据时，请考虑如何将数据转换为易于聚合的形式。

具体编程要求如下：

1. 首先，在这两个dataframe中添加一列，更加易于区分这两列的数据。
   1. 添加一个institute列到dete\_resignations\_up中。该列的值均设为DETE
   2. 添加一个institute列到tafe\_resignations\_up中。该列的值均设为DETE
2. 合并数据集，将结果赋值给combined
3. dataframe中还有一些列不需要完成分析。使用datafame.dropna()方法删除任何非空值小于500的列
   1. 可以通过设置thresh参数，删除非空值小于一定数量的列。
   2. 将结果赋值给combined\_updated
4. 另起一个markdown单元格，阐述你对数据的改变，以及原因。：

**10.清洗Service列：**

完成了之前的任务之后，现在已经很接近可以开始分析了。不过，首先，我们必须清理institute\_service列的值。因为它当前包含两个不同形式的值，所以清理过程相对复杂。



为了分析数据，将把这些数字转换为类别。我们将在这篇文章的基础上进行分析，这篇文章的论点是：根据职业阶段而不是年龄来了解员工的需求更有效。

我们将使用下面稍微修改过的定义

* + - New：在一家公司少于3年的员工
    - Experienced：在一家公司工作3-6年的员工
    - Established：在一家公司工作7-10年的员工
    - Veteran：在一家公司工作大于等于11年的员工

让我们使用上面的定义对institute\_service列中的值进行分类。

具体编程要求如下：

1. 首先，我们将从institute\_service列中的每个值中提取工作年限
   1. 使用Series.astype()方法将类型变为str类型
   2. 使用向量化的字符串方法从每个pattern中提取工作年限。可以在这里找到向量化字符串方法的完整列表。

<https://pandas.pydata.org/pandas-docs/stable/user_guide/text.html>

* 1. 仔细检查是否存在漏掉数字的情况。
  2. 使用Series.astype()方法将类型转化为float类型。

1. 接下来，将把每个值都分别映射到上面的某个职业阶段中。
   1. 创建一个函数，将每一个年份值都分别映射到上面的一个职业阶段。
      1. 您必须单独处理缺失值。可以通过pd.isnull(val)方法检测变量val的值是否为NaN
   2. 使用Series.apply()方法将声明的函数应用于institute\_service列上。并将结果赋值给一个新列，名为service\_cat
2. 另起一个markdown单元格，阐述你对数据的改变，以及原因。
3. 进行初步分析：

在之前的任务中，创建了一个service\_cat列，这是根据以下工作年限对员工进行分类的：

* + - New：在一家公司少于3年的员工
    - Experienced：在一家公司工作3-6年的员工
    - Established：在一家公司工作7-10年的员工
    - Veteran：在一家公司工作大于等于11年的员工

不仅在dissatisfied列汇总仍然存在缺失值，需要进行填充；还存在其他的缺失值需要处理。这意味着还需要进行一定的数据处理，保证后续工作正常开始。

Dissatisfied列中全为True或者False这种的布尔值。df.pivot\_table()会将布尔值当做整型值，将True当做1，False当做0处理。这意味着我们可以把dissatisfied那一列进行累加，从而计算出每一组的人数，每一组的人数百分比等等。

具体编程要求如下：

1. 使用Series.value\_counts()方法查看dissatisfied列中True和False各自的数量，设置参数dropna为False，保证也能看到缺失值的数量。
2. 使用DataFrame.fillna()方法替换dissatisfied列汇总的缺失值，替换策略为：是用出现次数最多的值进行填充（True或False）。
3. 使用DataFrame.pivot\_table()方法计算dissatisfied员工在每一个service\_cat组中的百分比。
   1. 因为True被看作是1，计算平均值也会计算不满意员工的百分比。默认的聚合函数是平均值，因此可以不使用aggfunc参数。
4. 使用DataFrame.plot()方法对结果进行制图。将kind参数绘制为bar，从而绘制一个柱状图
   1. 请在你的jupyter中顶部添加%matplotlib inline，保证你的图能够正常显示
5. 另起一个markdown单元格，阐述你对数据的改变，以及原因。

**三、概要设计**

**1.了解项目和数据集：**

本任务将主要按照如下内容展开分析：

* 那些只在研究院工作了很短一段时间的员工是否是由于存在某些不满而辞职?工作的更久一点的员工是否因存在不满而辞职？
* 年轻员工辞职是因为存在某种不满吗?老员工辞职是因为存在某种不满吗?

我们应当能够对这两个调查数据的结果进行综合考评，然后再回答上面的问题。我们的目标：完成大部分数据清理工作，并保证你可以开始分析第一个问题。

数据集中没有提供数据字典。因此对于这个项目，将使用我们的个人常识来定义列。下面是我们将在dete\_survey.csv中使用的一组列的解释:

* ID：用于识别调查参与者的id
* SeparationType：雇员雇佣关系终止的原因
* Cease Date：雇员结束工作的年或月
* DETE Start Date：雇员开始受雇于DETE的哪一年

下面是我们将在tafe\_survey.csv中使用的一组列的基本解释：

* Record ID: 用于标识调查参与者的id
* Reason for ceasing employment: 雇员雇佣关系终止的原因
* LengthofServiceOverall：学院总工作年限（单位：年）：员工的工作年限（单位：年）

**2.识别缺失的数据和丢弃不必要的列：**

通常，确定下一步如何对数据进行清洗和重塑都是比较关键且最困难的部分。但也不用太过担担心，后续我们将会提供一些步骤，并留有一些您自己的空间进行拓展。

首先可以进行如下观察：

* dete\_survey中包含了值为"Not Stated"，这些值表示缺失值，但它们并不是NaN。
* dete\_survey和tafe\_survey中都包含许多我们不需要分析的列。
* 这两个dataframe都包含许多类似的列，但列名不同。
* 有多少个列/答案表示员工因不满而辞职。

首先，我们将处理两个问题。使用pd.read\_csv()确定应该表示为NaN的值。我们将首先对缺失值进行处理。然后，我们再删除那些在分析中不需要的列。

**3.清洗列名：**

接下来，让我们将注意力转向列名。这两个dataframe包含许多类似的列，但是列名不同。下面是一些想用于最终分析的列：

|  |  |  |
| --- | --- | --- |
| dete\_survey | tafe\_survey | Definition |
| ID | Record ID | 用于识别调查参与者的id |
| SeparationType | Reason for ceasing employment | 参与者终止雇佣关系的原因 |
| Cease Date | CESSATION YEAR | 参与者就业结束的年份或月份 |
| DETE Start Date |  | 参与者开始受雇于DETE的年份 |
|  | LengthofServiceOverall. Overall Length of Service at Institute (in years) | 参与者的工作年限（年） |
| Age | CurrentAge. Current Age | 参与者的年龄 |
| Gender | Gender. What is your Gender? | 参与者的性别 |

因为最终想要合并这两个DataFrame，所以必须对列名进行标准化。回想一下，可以使用DataFrame.columns以及向量化的字符串方法，以一次性更新所有列。下面是之前的一个实例：

|  |
| --- |
| happiness2017.columns =  happiness2017.columns.str.replace('.', ' ').str.replace('\s+', ' ').str.strip().str.upper() |

**5.滤除数据：**

在上一关中，对需要使用的列进行了重命名。后面则是删除不需要的数据。回想一下，我们的最终目标是回答以下问题：

* 那些只在研究院工作了很短一段时间的员工是否是由于存在某些不满而辞职?工作的更久一点的员工是否因存在不满而辞职？

如果我们查看这两个dataframe的separationtype列中的惟一值情况，可以看到这两个DataFrame都各自包含一组不同的离职类型。对于本项目，我们将只分析辞职的员工，所以他们的离职类型的值中会包含字符串'Resignation'。

如果想挑战难度，请尝试使用所有的离职类型来完成项目——这将发现在数据清洗过程中会有更多的问题需要解决。

注意，dete\_survey\_updated中包含多种包含字符串'Resignation'的离职原因类型，这些值中都包含字符串'Resignment'：

* Resignation-Other reasons
* Resignation-Other employer
* Resignation-Move overseas/interstate

我们必须考虑到每个变量，这样我们才不会无意中丢失数据！

注：无论进行什么操作，都请记住先对DataFrame进行副本保存，然后进行修改。

**6.验证数据：**

在开始清洗和操作其余的数据之前，我们还需要验证数据是否存在一些问题。当你处理真正的数据时，请小心，不要猜测你正在分析的数据肯定是未被污染的！

或许无法发现所有错误，但是通过确保数据在我们所知的范围内看起来是合理的，以免在进行一个数据分析项目时，这个项目最终因为糟糕的数据而变得毫无用处。

在本关，我们将重点检查“cease\_date”和“dete\_start\_date”列中的年份是否有意义。也鼓励同学们自行检查其他数据。

* 因为“cease\_date”是员工工作的最后一年，而“dete\_start\_date”是员工工作的第一年，因此在当前日期之后日期是没有意义的。
* 考虑到这个领域的大多数人都是在20多岁开始工作的，所以dete\_start\_date的值不太可能早于1940年。

如果数据中存在比当前年份更大或者比1940年的年份更小的数据，就应当人停止对数据的分析，因为这可能意味着数据有一些不合乎常理的地方。如果有少量的值过高或过低，那就直接删除这些数据。

**7.创建新列:**

上一关，我们可以看出：

1.年份数值不存在问题。

2.两个DataFrame中的年份跨度不完全相同。因此需要放弃一些不需要分析的年份。

现在已经证实了dete\_resignations中年份是没有问题的，将使用它们创建一个新的列。回想一下，我们的最终目标是回答以下问题：

* 那些只在研究院工作了很短一段时间的员工是否是由于存在某些不满而辞职?工作的更久一点的员工是否因存在不满而辞职？

在人力资源部门有一个专门的称呼，员工在工作地点停留的时间被称为他们的工作年限。可以发现，tafe\_resignations中已经包含一个service列。将其重命名为institute\_ service。为了同时分析这两份调查报告数据，必须在dete\_resignations中创建一个相应的institute service列。

我们是否有数据可以用来计算员工在工作地点停留的时间?花一分钟再次回顾dete\_resignations的列，是存在两个关键的列可以计算的。

具体编程任务如下：

1. 在dete\_resignations中创建一个institute\_service列

* 用cease\_date列的值减去dete\_start\_date列的值，并将结果赋值给列institute\_service列。

1. 另起一个markdown单元格，阐述你对数据的改变，以及原因 。

**8.识别不满的雇员:**

上一关中，我们新建了一个institute\_service列，用于保存被调查者的工作入职时长（单位：年份）。接下来我们将会找出那些因为不满而辞职的员工。

我们将对dataframe中员工进行分类，

下面是我们将用于从dataframe中将员工分类为“不满意”的列。如果你不认可这种分类策略，可以自行修改！但是需要解释原因。

**9.合并数据：**

对已经完成的任务进行如下总结：

1. 给列进行重命名
2. 删除我们分析中不需要的列
3. 检查了数据的正确性
4. 创建了一个新列institute\_service
5. 对Contributing Factors列的值进行了数据清洗
6. 创建一个新列，表明员工是否因某些方面的不满而辞职

现在，可以将数据集进行合并了。最终，需要根据institute service列将数据合并，因此在组合数据时，请考虑如何将数据转换为易于聚合的形式。

**10.清洗Service列：**

完成了之前的任务之后，现在已经很接近可以开始分析了。不过，首先，我们必须清理institute\_service列的值。因为它当前包含两个不同形式的值，所以清理过程相对复杂。

为了分析数据，将把这些数字转换为类别。我们将在这篇文章的基础上进行分析，这篇文章的论点是：根据职业阶段而不是年龄来了解员工的需求更有效。

1. 进行初步分析：

在之前的任务中，创建了一个service\_cat列，这是根据以下工作年限对员工进行分类的：

* + - New：在一家公司少于3年的员工
    - Experienced：在一家公司工作3-6年的员工
    - Established：在一家公司工作7-10年的员工
    - Veteran：在一家公司工作大于等于11年的员工

不仅在dissatisfied列汇总仍然存在缺失值，需要进行填充；还存在其他的缺失值需要处理。这意味着还需要进行一定的数据处理，保证后续工作正常开始。

Dissatisfied列中全为True或者False这种的布尔值。df.pivot\_table()会将布尔值当做整型值，将True当做1，False当做0处理。这意味着我们可以把dissatisfied那一列进行累加，从而计算出每一组的人数，每一组的人数百分比等等。

**四、详细设计**

**4.1设计思想**

**（一）了解项目和数据集：**

具体编程任务如下：

1. 首先在markdown单元格中编写一段介绍项目和数据集的注释。

2. 导入pandas和NumPy库。

3. 通过pandas读取dete\_survey.csv文件，并赋值给变量dete\_survey。

4. 通过pandas读取tafe\_survey.csv文件，并赋值给变量tafe\_survey。

5. 使用DataFrame.info() 和DataFrame.head()方法查看这两个DataFrame的基本信息（打印前6行）。使用其他数据探索方法，比如：Series.value\_counts()和DataFrame.isnull()方法来查看数据并确定一些后续步骤。

%matplotlib inline

import pandas as pd

import numpy as np

import matplotlib

import matplotlib.pyplot as plt

dete\_survey = pd.read\_csv('dete\_survey.csv')

df1 = pd.DataFrame(dete\_survey)

tafe\_survey = pd.read\_csv('tafe\_survey.csv')

df2 = pd.DataFrame(tafe\_survey)

print("数据的第一行为:\n",df1.iloc[0])

#输出CSV文件的前5行和最后5行，pandas默认的输出5行

print("\n数据的前5:\n",df1.head())

print("\n数据:\n",df1.info())

print("\n数据丢失数据:\n",df1.isnull())

print("\nID\n",dete\_survey['ID'].value\_counts())

**（二）识别缺失的数据和丢弃不必要的列：**

具体编程任务如下：

1. 通过pandas读取dete\_survey.csv文件，但是本次读取时，将值为"Not Stated"的地方都修改为"NaN"

○ 可通过设置pd.read\_csv()方法中的参数na\_values为Not Stated实现

○ 将数据读取结果赋值给变量dete\_survey

2. 通过pandas读取tafe\_survey.csv文件，但是本次读取时，将值为"Not Stated"的地方都修改为"NaN"

○ 通过设置pd.read\_csv()方法中的参数na\_values为Not Stated实现

○ 将数据读取结果赋值给变量tafe\_survey

3. 然后，删除dataframe中一些不需要的列，保证dataframe更易于使用

○ 使用dataframe .drop()方法从dete\_survey中删除以下列:

dete\_survey.columns[28:49]，请将删除维度参数axis设置为1

 将删除后的结果赋值给变量dete\_survey\_updated

○ 使用dataframe .drop()方法从tete\_survey中删除以下列:

tete\_survey.columns[17:66]，请将删除维度参数axis设置为1

 将删除后的结果赋值给变量tafe\_survey\_updated

dete\_survey = pd.read\_csv('dete\_survey.csv',na\_values=["Not Stated"])

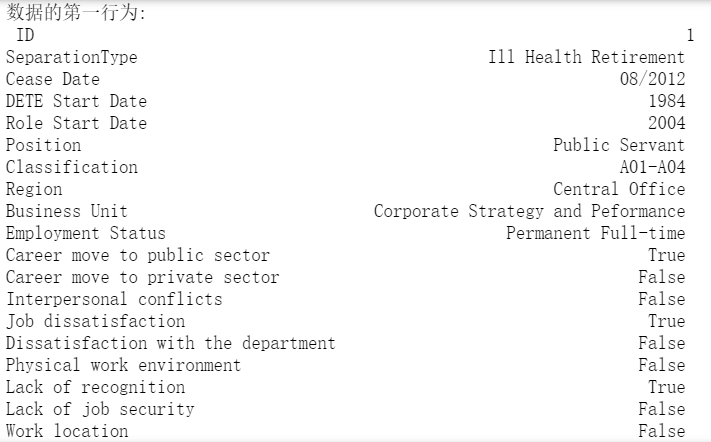
tafe\_survey = pd.read\_csv('tafe\_survey.csv',na\_values=["Not Stated"])

dt = pd.DataFrame(dete\_survey)

tf = pd.DataFrame(tafe\_survey)

dete\_survey\_updated = dt.drop(dt.columns[28:49], axis=1)

tafe\_survey\_updated = tf.drop(tf.columns[17:66], axis=1)



1. **清洗列名：**

1.对dete\_survey\_updated中的剩余列重命名

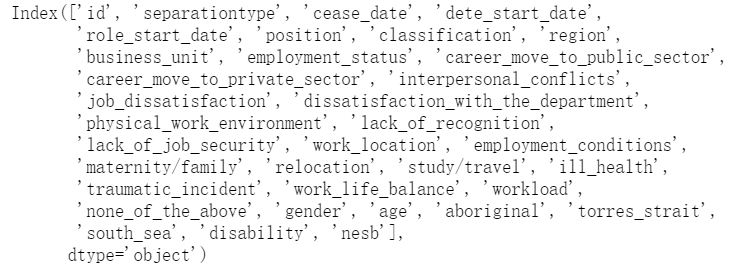
* 按照以下要求更新列名：
  + 所有大写字母均改为小写字母
  + 删除字符串结尾的所有空格。
  + 用下划线('\_')替换单词之间的空格。
* 例如：Cease Date应当更新为cease\_date
* 可以用DataFrame.columns属性，以一个数组的形式输出现有的列名。

2.使用datafame.rename()方法更新tafe\_survey\_updated中的以下列。

3.使用datafame.head()方法查看已更新的dete\_survey\_updated和tafe\_survey\_updated的更新情况，确保更新列名准确。

dete\_survey\_updated.columns = dete\_survey\_updated.columns.str.replace(' ','\_').str.rstrip().str.lower()

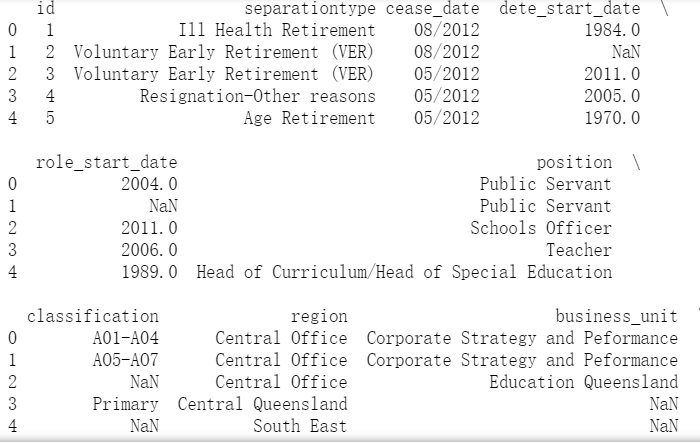
print(dete\_survey\_updated.columns)



tafe\_survey\_updated.columns = tafe\_survey\_updated.columns.str.replace('Record ID','id').str.replace('CESSATION YEAR','cease\_date').str.replace('Reason for ceasing employment','separationtype').str.replace('Gender. What is your Gender?','gender').str.replace('CurrentAge. Current Age','age').str.replace('Employment Type. Employment Type','employment\_status').str.replace('Classification. Classification','position').str.replace('LengthofServiceOverall. Overall Length of Service at Institute (in years)','institute\_service').str.replace('LengthofServiceCurrent. Length of Service at current workplace (in years)','role\_service')

print(dete\_survey\_updated.head())

print(tafe\_survey\_updated.head())



1. **滤除数据：**

具体编程任务如下：

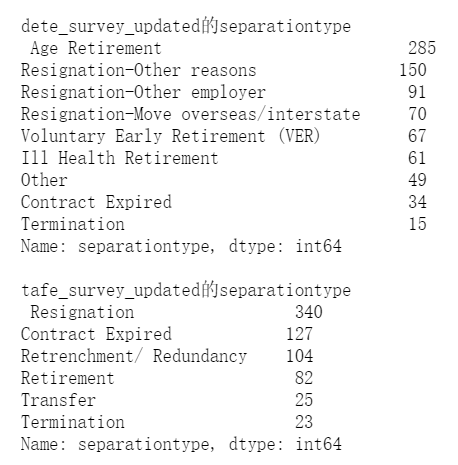
1.使用Series.value\_counts()方法分别查看dete\_survey\_updated和tafe\_survey\_updated的separationtype列中每一种值的数量情况。

2.在这两个dataframes中，筛选出辞职原因类型值中包含Resignation字符串的调查对象对应的数据。

* dete\_survey\_updated中有三种离职原因都包含Resignation字符串，这三种都需要考虑进来，作为离职原因为Resignation的数据。
  + 将dete\_survey\_updated中separationtype进行切片，拿到第一个单词并赋值回separationtype，从而使得三种都包含Resignation字符串统一为一种。
* 筛选dete\_survey\_updated 中separationtype列的值为Resignation的项，所查找出的结果通过DataFrame.copy()方法赋值给dete\_resignations
* 筛选tafe\_survey\_updated中 separationtype列的值为Resignation的项，所查找出的结果通过DataFrame.copy()方法赋值给tafe\_resignations

print("\ndete\_survey\_updated的separationtype\n",dete\_survey\_updated['separationtype'].value\_counts())

print("\ntafe\_survey\_updated的separationtype\n",tafe\_survey\_updated['separationtype'].value\_counts())



def extract\_first\_word(element):

return str(element).split()[0]

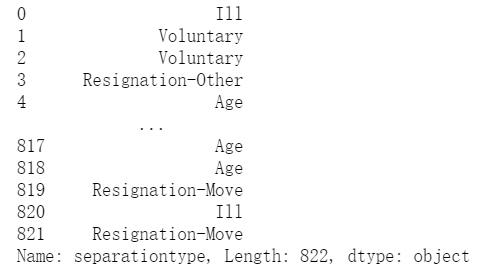
dete\_survey\_updated = pd.DataFrame(dete\_survey\_updated)

#tf = pd.DataFrame(tafe\_survey)

dete\_survey\_updated['separationtype'] = dete\_survey\_updated['separationtype'].apply(extract\_first\_word)

#dete\_survey\_updated['separationtype'] = dete\_survey\_updated['separationtype'].str.upper().str.split()

print(dete\_survey\_updated['separationtype'])



dete\_resignations = dete\_survey\_updated.copy()

tafe\_resignations = tafe\_survey\_updated.copy()

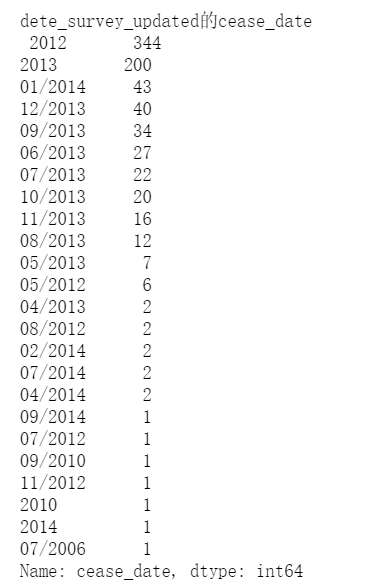
1. **验证数据：**

具体编程任务如下：

1.检查dataframe中的年份是否存在逻辑问题

* 首先，清洗dete\_resignations中的cease\_date列的数据
  + 使用Series.value\_counts()方法查看cease\_date列中的每一种值的数量情况
  + 使用向量化的字符串方法提取年份。
  + 使用Series.astype()方法将原类型转换为浮点型。
* 使用Series.value\_counts()查看dete\_resignations 中cease\_date 列和dete\_start\_date列的值；以及tafe\_resignations中cease\_date列的值。
  + 因为Series.value\_counts()返回一个series对象，所以可以使用Series.sort\_index()方法查看数量最多或是数量最少的值。通过修改参数ascending为True或是False即可设置升序或者为降序输出。
* 可以使用箱形图绘制任何数字列的值，以查看任何看起来存在问题的值。

print("\ndete\_survey\_updated的cease\_date\n",dete\_survey\_updated['cease\_date'].value\_counts())



pattern = r"([1-2][0-9]{3})"

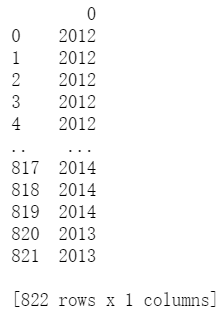
# 根据指定pattern截取年份

years = dete\_survey\_updated['cease\_date'].str.extract(pattern)

# 打印结果

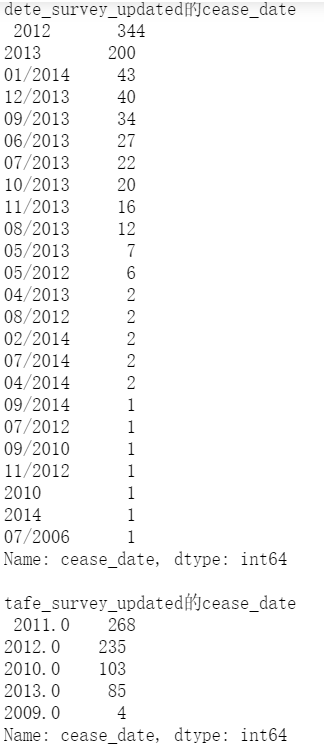
print(years)

dete\_survey\_updated['cease\_date'] = years.astype(float)



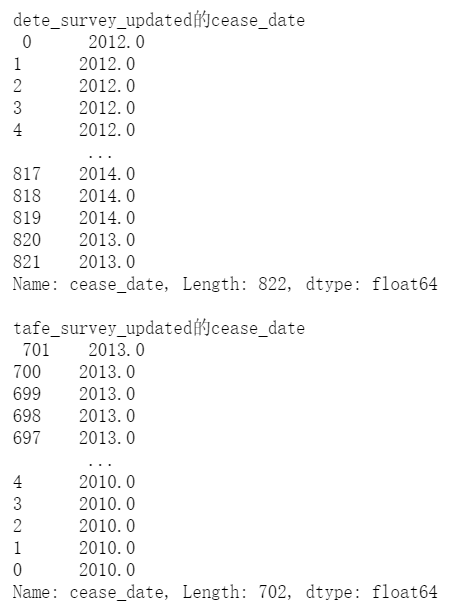
print("\ndete\_survey\_updated的cease\_date\n",dete\_survey\_updated['cease\_date'].value\_counts())

print("\ntafe\_survey\_updated的cease\_date\n",tafe\_survey\_updated['cease\_date'].value\_counts())



print("\ndete\_survey\_updated的cease\_date\n",dete\_survey\_updated['cease\_date'].sort\_index(ascending=True))

print("\ntafe\_survey\_updated的cease\_date\n",tafe\_survey\_updated['cease\_date'].sort\_index(ascending=False))



1. **创建新列:**

具体编程任务如下：

1.在dete\_resignations中创建一个institute\_service列

* 用cease\_date列的值减去dete\_start\_date列的值，并将结果赋值给列institute\_service列。

dete\_survey\_updated['institute\_service'] = dete\_survey\_updated['cease\_date'] - dete\_survey\_updated['dete\_start\_date']

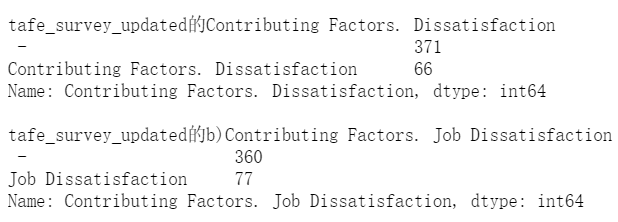
**(七)识别不满的雇员:**

具体编程任务如下：：

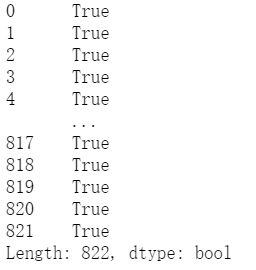
1. 使用Series.value\_counts()方法查看tafe\_resignations中以下两个列的内容。
   1. Contributing Factors. Dissatisfaction
   2. Contributing Factors. Job Dissatisfaction
2. 将这两列的值更新为仅含True、False、NaN三种情况
   1. 声明一个叫做update\_vals的方法用于进行一下变换：
      1. 如果值为NaN，则返回np.nan（可以通过方法pd.isnull(val)判断一个值是否为NaN）
      2. 如果值为`-`, 返回False
      3. 如果是其他的值，则返回True
   2. 使用DataFrame.applymap()方法，将所声明的函数应用在上述两列中
      1. 传入update\_vals到方法applymap()中不需要括号
3. 使用df.any()方法在上述tafe\_resignations和dete\_resignations中分别创建一个名为dissatisfied的列
4. 使用df.copy()方法对结果进行备份，并将结果分别赋值为dete\_resignations\_up和tafe\_resignations\_up

print("\ntafe\_survey\_updated的Contributing Factors. Dissatisfaction\n",tafe\_survey\_updated['Contributing Factors. Dissatisfaction'].value\_counts())

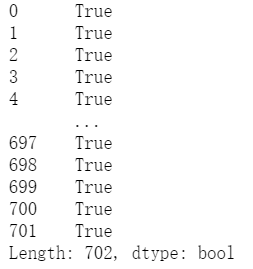
print("\ntafe\_survey\_updated的Contributing Factors. Job Dissatisfaction\n",tafe\_survey\_updated['Contributing Factors. Job Dissatisfaction'].value\_counts())



dete\_survey\_updated.any(axis=1, skipna=False)



tafe\_survey\_updated.any(axis=1, skipna=False)



7.

dete\_survey['dissatisfied'] = dete\_survey.any(axis=1, skipna=False)

tafe\_survey['dissatisfied'] = tafe\_survey.any(axis=1, skipna=False)

dete\_resignations\_up = dete\_survey['dissatisfied'].copy()

tafe\_resignations\_up = tafe\_survey['dissatisfied'].copy()

**（八）合并数据：**

具体编程要求如下：

1.首先，在这两个dataframe中添加一列，更加易于区分这两列的数据。

* 1. 添加一个institute列到dete\_resignations\_up中。该列的值均设为DETE
  2. 添加一个institute列到tafe\_resignations\_up中。该列的值均设为DETE

2.合并数据集，将结果赋值给combined

3.dataframe中还有一些列不需要完成分析。使用datafame.dropna()方法删除任何非空值小于500的列

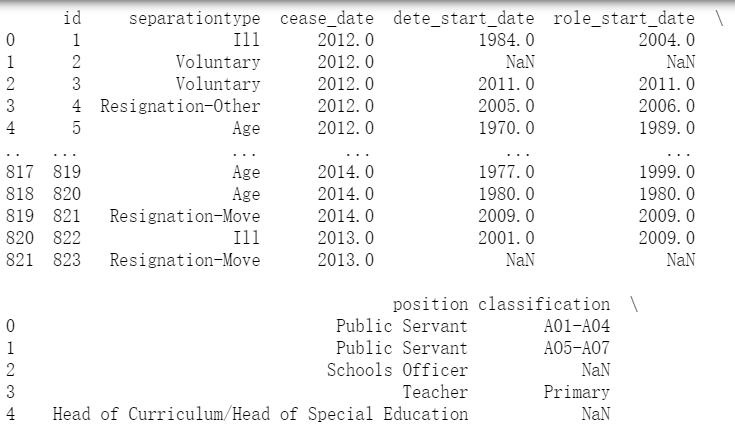
* 1. 可以通过设置thresh参数，删除非空值小于一定数量的列。
  2. 将结果赋值给combined\_updated

dete\_resignations\_up['institute'] = 'DETE'

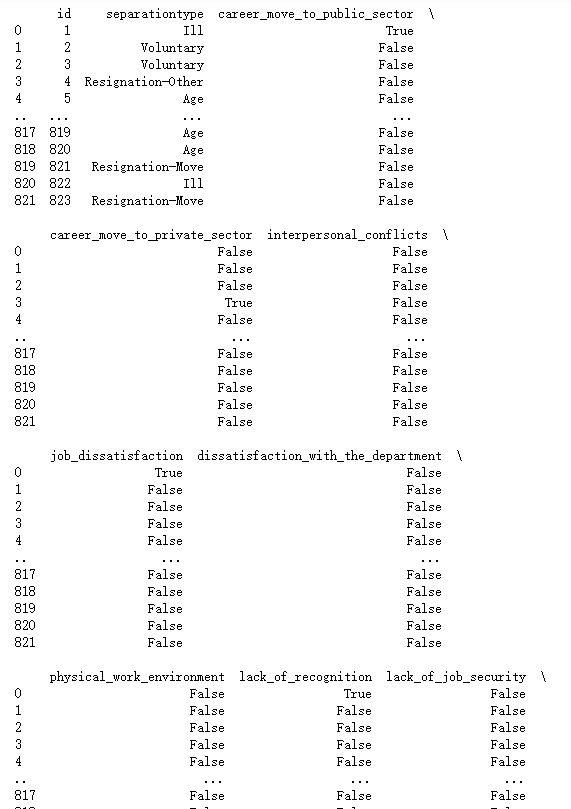
tafe\_resignations\_up['institute'] = 'TAFE'

combined = pd.concat([dete\_survey\_updated,tafe\_survey\_updated],axis =1)

print(combined)

combined\_updated = combined.dropna(axis=1, thresh=None)

print(combined\_updated)



**（九）清洗Service列：**

具体编程要求如下：

1.首先，我们将从institute\_service列中的每个值中提取工作年限

* 1. 使用Series.astype()方法将类型变为str类型
  2. 使用向量化的字符串方法从每个pattern中提取工作年限。可以在这里找到向量化字符串方法的完整列表。

<https://pandas.pydata.org/pandas-docs/stable/user_guide/text.html>

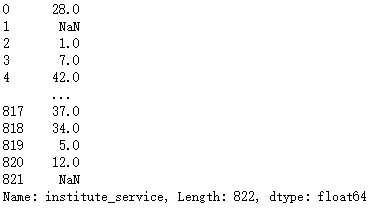
* 1. 仔细检查是否存在漏掉数字的情况。
  2. 使用Series.astype()方法将类型转化为float类型。

2.接下来，将把每个值都分别映射到上面的某个职业阶段中。

* 1. 创建一个函数，将每一个年份值都分别映射到上面的一个职业阶段。
     1. 您必须单独处理缺失值。可以通过pd.isnull(val)方法检测变量val的值是否为NaN
  2. 使用Series.apply()方法将声明的函数应用于institute\_service列上。并将结果赋值给一个新列，名为service\_cat

combined['institute\_service'].astype(str)

combined['institute\_service'].astype(float)



service\_cat = combined['institute\_service'].apply(extract\_first\_word)

(**十) 进行初步分析：**

具体编程要求如下：

1.使用Series.value\_counts()方法查看dissatisfied列中True和False各自的数量，设置参数dropna为False，保证也能看到缺失值的数量。

2.使用DataFrame.fillna()方法替换dissatisfied列汇总的缺失值，替换策略为：是用出现次数最多的值进行填充（True或False）。

3.使用DataFrame.pivot\_table()方法计算dissatisfied员工在每一个service\_cat组中的百分比。

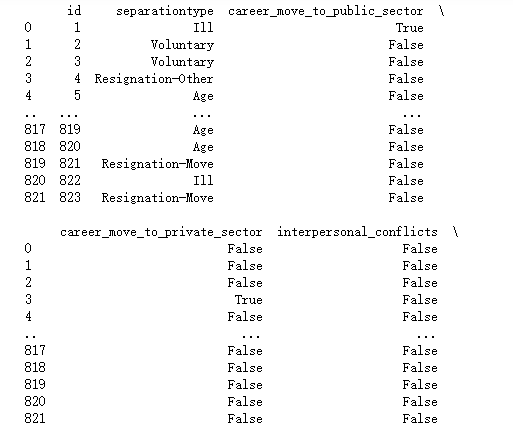
* 1. 因为True被看作是1，计算平均值也会计算不满意员工的百分比。默认的聚合函数是平均值，因此可以不使用aggfunc参数。

4.使用DataFrame.plot()方法对结果进行制图。将kind参数绘制为bar，从而绘制一个柱状图

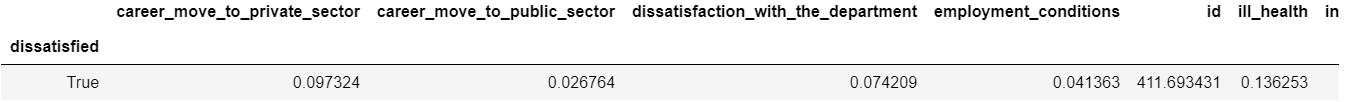
* 1. 请在你的jupyter中顶部添加%matplotlib inline，保证你的图能够正常显示

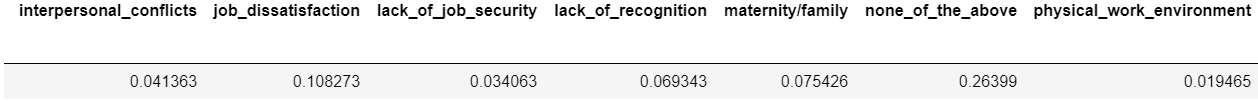
print(combined\_updated['dissatisfied'].value\_counts().dropna(inplace=False))

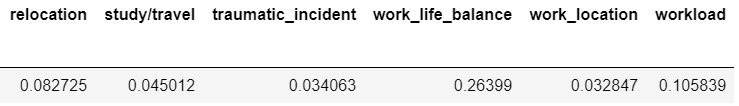


print(combined\_updated.fillna(False)) 

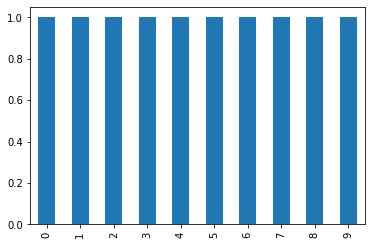
pd.pivot\_table(combined\_updated,index=["dissatisfied"])







combined\_updated[:10]['dissatisfied'].astype(float).plot(kind='bar')



**五、结果分析：**

**5.1实现注释程序清单**

%matplotlib inline

import pandas as pd

import numpy as np

import matplotlib

import matplotlib.pyplot as plt

dete\_survey = pd.read\_csv('dete\_survey.csv')

df1 = pd.DataFrame(dete\_survey)

tafe\_survey = pd.read\_csv('tafe\_survey.csv')

df2 = pd.DataFrame(tafe\_survey)

print("数据的第一行为:\n",df1.iloc[0])

#输出CSV文件的前5行和最后5行，pandas默认的输出5行

print("\n数据的前5:\n",df1.head())

print("\n数据:\n",df1.info())

print("\n数据丢失数据:\n",df1.isnull())

print("\nID\n",dete\_survey['ID'].value\_counts())

dete\_survey = pd.read\_csv('dete\_survey.csv',na\_values=["Not Stated"])

tafe\_survey = pd.read\_csv('tafe\_survey.csv',na\_values=["Not Stated"])

dt = pd.DataFrame(dete\_survey)

tf = pd.DataFrame(tafe\_survey)

dete\_survey\_updated = dt.drop(dt.columns[28:49], axis=1)

tafe\_survey\_updated = tf.drop(tf.columns[17:66], axis=1)

happiness2017.columns = happiness2017.columns.str.replace('.', ' ').str.replace('\s+', ' ').str.strip().str.upper()

dete\_survey\_updated.columns = dete\_survey\_updated.columns.str.replace(' ','\_').str.rstrip().str.lower()

print(dete\_survey\_updated.columns)

tafe\_survey\_updated.columns = tafe\_survey\_updated.columns.str.replace('Record ID','id').str.replace('CESSATION YEAR','cease\_date').str.replace('Reason for ceasing employment','separationtype').str.replace('Gender. What is your Gender?','gender').str.replace('CurrentAge. Current Age','age').str.replace('Employment Type. Employment Type','employment\_status').str.replace('Classification. Classification','position').str.replace('LengthofServiceOverall. Overall Length of Service at Institute (in years)','institute\_service').str.replace('LengthofServiceCurrent. Length of Service at current workplace (in years)','role\_service')

print(dete\_survey\_updated.head())

print(tafe\_survey\_updated.head())

print("\ndete\_survey\_updated的separationtype\n",dete\_survey\_updated['separationtype'].value\_counts())

print("\ntafe\_survey\_updated的separationtype\n",tafe\_survey\_updated['separationtype'].value\_counts())

def extract\_first\_word(element):

return str(element).split()[0]

dete\_survey\_updated = pd.DataFrame(dete\_survey\_updated)

#tf = pd.DataFrame(tafe\_survey)

dete\_survey\_updated['separationtype'] = dete\_survey\_updated['separationtype'].apply(extract\_first\_word)

#dete\_survey\_updated['separationtype'] = dete\_survey\_updated['separationtype'].str.upper().str.split()

print(dete\_survey\_updated['separationtype'])

dete\_resignations = dete\_survey\_updated.copy()

tafe\_resignations = tafe\_survey\_updated.copy()

print("\ndete\_survey\_updated的cease\_date\n",dete\_survey\_updated['cease\_date'].value\_counts())

pattern = r"([1-2][0-9]{3})"

# 根据指定pattern截取年份

years = dete\_survey\_updated['cease\_date'].str.extract(pattern)

# 打印结果

print(years)

dete\_survey\_updated['cease\_date'] = years.astype(float)

print("\ndete\_survey\_updated的cease\_date\n",dete\_survey\_updated['cease\_date'].value\_counts())

print("\ntafe\_survey\_updated的cease\_date\n",tafe\_survey\_updated['cease\_date'].value\_counts())

print("\ndete\_survey\_updated的cease\_date\n",dete\_survey\_updated['cease\_date'].sort\_index(ascending=True))

print("\ntafe\_survey\_updated的cease\_date\n",tafe\_survey\_updated['cease\_date'].sort\_index(ascending=False))

dete\_survey\_updated['institute\_service'] = dete\_survey\_updated['cease\_date'] - dete\_survey\_updated['dete\_start\_date']

print("\ntafe\_survey\_updated的Contributing Factors. Dissatisfaction\n",tafe\_survey\_updated['Contributing Factors. Dissatisfaction'].value\_counts())

print("\ntafe\_survey\_updated的b)Contributing Factors. Job Dissatisfaction\n",tafe\_survey\_updated['Contributing Factors. Job Dissatisfaction'].value\_counts())

dete\_survey\_updated.any(axis=1, skipna=False)

tafe\_survey\_updated.any(axis=1, skipna=False)

dete\_survey['dissatisfied'] = dete\_survey.any(axis=1, skipna=False)

tafe\_survey['dissatisfied'] = tafe\_survey.any(axis=1, skipna=False)

dete\_resignations\_up = dete\_survey['dissatisfied'].copy()

tafe\_resignations\_up = tafe\_survey['dissatisfied'].copy()

dete\_resignations\_up['institute'] = 'DETE'

tafe\_resignations\_up['institute'] = 'TAFE'

combined = pd.concat([dete\_survey\_updated,tafe\_survey\_updated],axis =1)

print(combined)

combined\_updated = combined.dropna(axis=1, thresh=None)

print(combined\_updated)

combined['institute\_service'].astype(str)

combined['institute\_service'].astype(float)

service\_cat = combined['institute\_service'].apply(extract\_first\_word)

print(combined\_updated['dissatisfied'].value\_counts().dropna(inplace=False))

print(combined\_updated.fillna(False))

pd.pivot\_table(combined\_updated,index=["dissatisfied"])

combined\_updated[:10]['dissatisfied'].astype(float).plot(kind='bar')

**5.2结果分析**

根据以上分析我们不难发现，不论是老员工还是新员工员工对于工作的满意程度实际上取决于多种因素，根据不同的百分比我们可以看出，work\_life\_balance导致对工作的不满意影响占比最大，可以得出工作量大使得员工不能很好的平衡生活和工作，会使员工降低对公司工作的满意程度，所以公司可以在一定基础上适当调整员工的休息时间，从而降低其辞职率提升对工作的满意度，这样也许会给公司带来更大的收益。

**六、调试及总结**

本次熟悉的几个方法：

1. replace() 方法

字符串中的 old（旧字符串） 替换成 new(新字符串)，如果指定第三个参数max，则替换不超过 max 次。

replace()方法语法：str.replace(old, new[, max])

old -- 将被替换的子字符串。

new -- 新字符串，用于替换old子字符串。

max -- 可选字符串, 替换不超过 max 次

2.str(element).split()[0]

str.split("[")[1]. split("]")[0]输出的是 [ 后的内容以及 ] 前的内容。

str.split("[")[1]. split("]")[0]. split(".") 是先输出 [ 后的内容以及 ] 前的内容，然后通过 . 作为分隔符对字符串进行切片。

3．.any()  
any(iterable) 参数 iterable 必须是元组或列表。any() 函数用于判断给定的可迭代参数 iterable 是否全部为 False，如果是，则返回 False，如果其中的参数有一个为 True，则返回 True。元素除了是 0、空、FALSE 外都算 TRUE。

4. pd.concat()

pd.concat(objs, axis=0, join='outer', join\_axes=None, ignore\_index=False, keys=None, levels=None, names=None, verify\_integrity=False, copy=True)

objs 需要连接的对象，eg [df1, df2]

axis axis = 0, 表示在水平方向(row)进行连接 axis = 1, 表示在垂直方向(column)进行连接

join outer, 表示index全部需要; inner,表示只取index重合的部分

join\_axes 传入需要保留的index

ignore\_index 忽略需要连接的frame本身的index。当原本的index没有特别意义的时候可以使用

keys 可以给每个需要连接的df一个label

5. .apply()

apply(func [, args [, kwargs ]]) 函数用于当函数参数已经存在于一个元组或字典中时，间接地调用函数。args是一个包含将要提供给函数的按位置传递的参数的元组。如果省略了args，任 何参数都不会被传递，kwargs是一个包含关键字参数的字典。apply()的返回值就是func()的返回值，apply()的元素参数是有序的，元素的顺序必须和func()形式参数的顺序一致.

1. pandas.DataFrame.fillna()函数

DataFrame.fillna(value=None, method=None, axis=None, inplace=False, limit=None, downcast=None, \*\*kwargs)

value: 变量、字典、Series，DataFrame；用于填充填充缺失值，或指定为每个索引（对于Series）或列（对于DataFrame）的缺失值使用字典/Series/DataFrame的值填充

method: {'backfill', 'bfill', 'pad', 'ffill', None}, 默认None， pad/ffill表示向后填充空值，backfill/bfill表示向前填充空值

axis: {0 or 'index', 1 or 'columns'}

inplace: boolean, 默认为False。若为True，在原地填满

limit: int, 默认为None，如果指定了方法，则这是连续的NaN值的前向/后向填充的最大数量

downcast: dict, 默认None,字典中的项为类型向下转换规则