**华东师范大学数据科学与工程学院实验报告**

|  |  |  |
| --- | --- | --- |
| **课程名称**：数据挖掘 | **年级**：2021级 | **上机实践成绩**： |
| **指导教师**：周昉 | **姓名**：李睿恩 | **学号**：10215501434 |
| **上机实践名称**：Data Preprocessing | | **上机实践日期**：2024.3.20 |
| **上机实践编号**：1 | **组号**：无 | **上机实践时间**：22:14 |

1. 实验目的

* 完成数据预处理任务。

1. 实验任务

* 编写Python代码，实现数据预处理。
* 了解数据质量问题、数据聚合、采样、离散化与主成分分析。

1. 实验环境

本次实验基于JetBrains的PyCharm集成开发环境，使用Python编程语言实现。

1. 实验过程

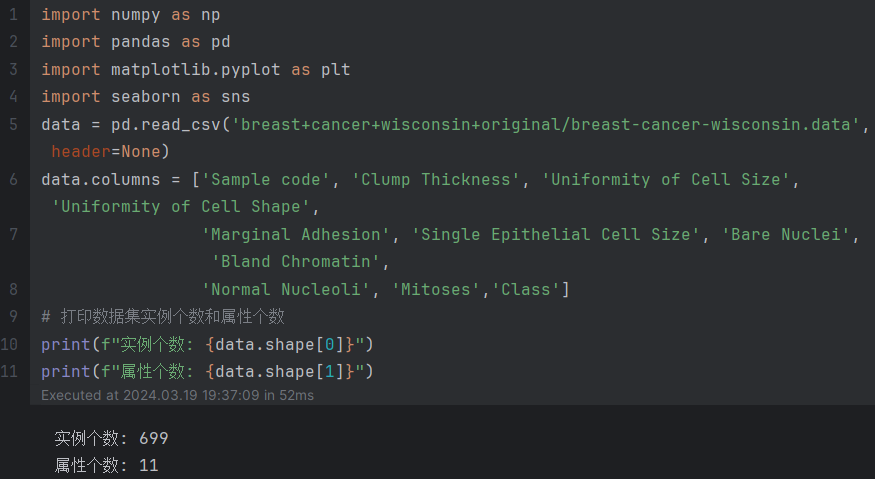
**4.1 数据质量问题**

在实际应用中，我们得到的数据往往并不能直接使用。这主要是因为，我们收集的原始数据非常有可能存在数据质量问题。较差的数据质量可能会对数据挖掘产生不利影响。常见的数据质量问题包括噪声、异常值、缺失值和重复数据。在这里，我们会尝试解决一些常见的数据质量问题。

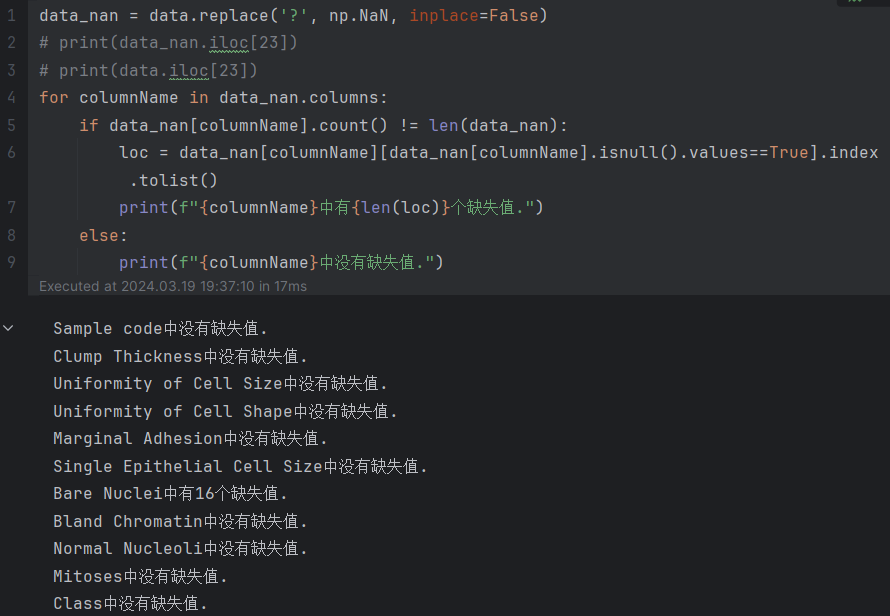
我们使用的数据是breast-cancer-wisconsin.data，在这里我们对数据进行导入，同时导入一些之后可能会使用的模块。利用Python，我们可以发现该数据集中存在699个实例，每个实例都有11个属性，如图1所示。

一种常见的数据质量问题是缺失值。在我们使用的数据中，缺失值的编码为“?”，为了更通用地处理整个数据表格，我们可以利用data.replace(‘?’, np.NaN)将缺失值转换为NaN。这之后，我们还编写了一段代码以计算每列数据中缺失值的数量。代码如图2所示。

另外一种处理缺失值的方法是将缺失值替换为该列的中值。在这里，我们使用了data.fillna(data.median())的代码将缺失值替换为了中值。第三种处理缺失值的方式是直接将这些缺失值的数据点删除，我们采用data.dropna()即可实现。这部分代码如图3所示。



**图1 导入模块与数据集**



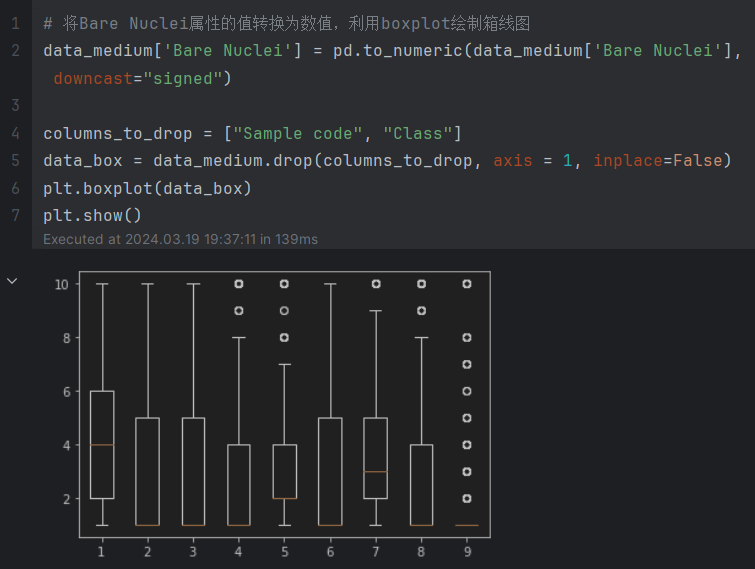
**图2 缺失值转为NaN并计算每列数据中缺失值的数量**



**图3 将缺失值替换为一列的中值或丢弃包含缺失值的数据点**

数据质量不佳也可能是因为存在过多异常值，而发现异常值的方法是绘制箱线图。在这里，由于第1个属性与第11个属性均是离散型变量，没有绘制箱线图的意义，因此我们只绘制第2个到第9个属性的箱线图。为了图像的绘制，我们选取了之前将缺失值换为中值的数据集用于绘制箱线图。

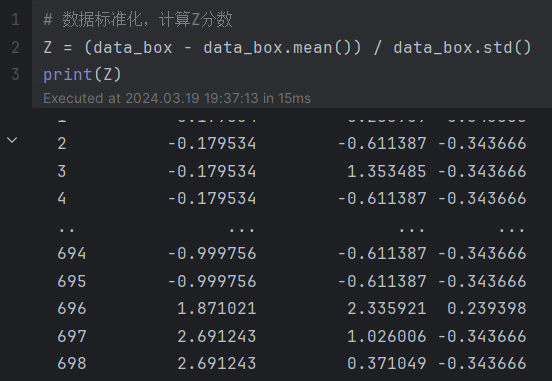
注意到在刚才的数据预处理中，我们发现只有Bare Nuclei属性中存在‘?’，因此这一列的值均是字符串对象，这是无法绘制箱线图的。因此，我们采用了pandas.to\_numeric将这一列转为了整数。最终，我们利用boxplot函数绘制的箱线图如图4所示。



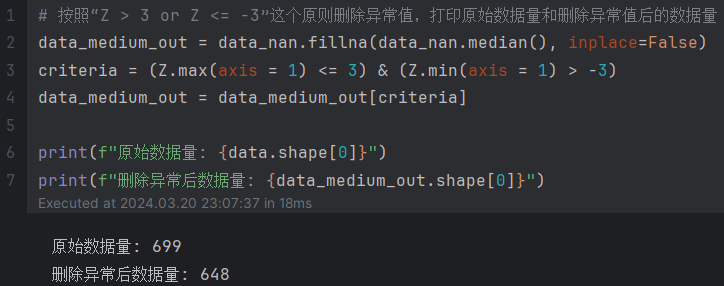
**图4 绘制箱线图**

这样的箱线图充分体现出了我们的数据中存在异常值。为了丢弃异常值，我们可以计算每个属性的Z分数（即每个数据在该属性中标准化的结果，如图5），并删除那些包含Z分数异常高或异常低的属性的实例。在这里，我们要求9个属性都必须小于等于3且大于-3，如果有一行数据中存在某个属性大于3或者小于等于-3，那么就需要将这行数据删除。

相比于删除异常数据，我们的方法是选择其中正常的数据。如果我们规定一个正常数据的9个属性都必须同时小于等于3并且大于-3，那么就意味着该行数据的9个属性中的最大值必然小于等于3，最小值必然大于-3，而用这种方法我们可以取出符条件的数据的索引。我们可以获得9个属性中的最大值小于等于3的数据的索引，再取出9个属性中的最小值大于-3的数据的索引，对两份索引进行按位与，即可得到正常数据的索引。该方法的实现如图6.



**图5 计算属性的Z分数**



**图6 删除异常值**

数据质量不佳也有可能是因为其中存在重复数据。在这里，我们采用data.duplicated()以获得重复的数据行，最后，我们利用data.drop\_duplicates()删除重复的行。该流程如图7所示。

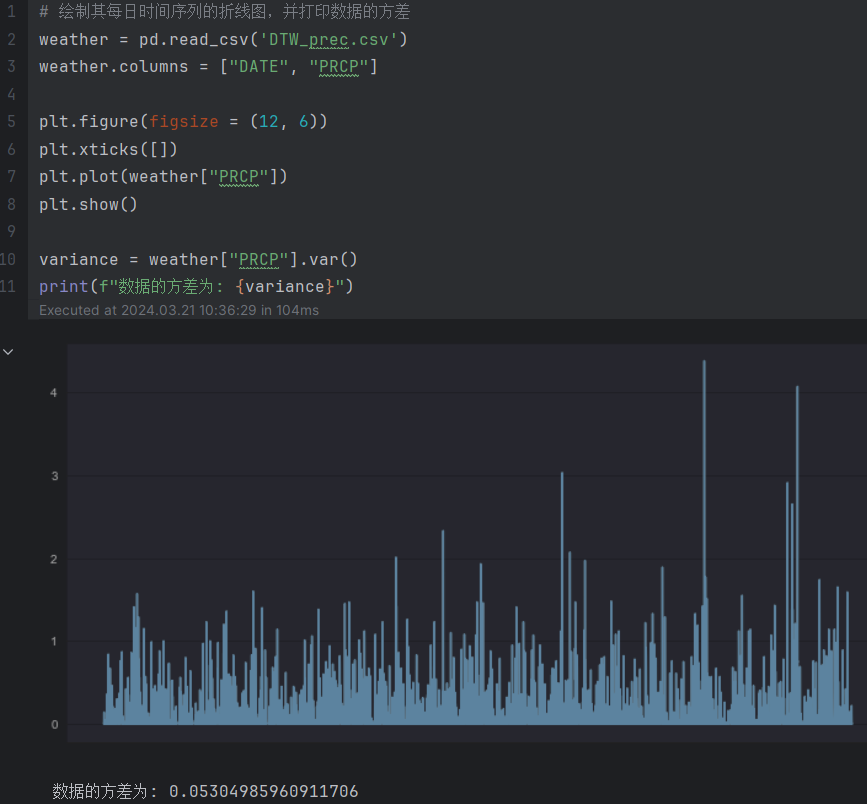


**图7 删除重复数据**

**4.2 数据聚合**

如果数据的量过大，直接地分析数据可能是一件费力且效果不佳的事。为此，我们需要进行数据聚合，其主要目的是减少要处理的数据的大小，改变分析的粒度，并提高数据的稳定性。本次实验中我们使用DTW\_prec.csv中的降水量数据。

我们首先使用matplotlib模块绘制其每日时间序列的折线图，并打印数据的方差。效果如图8.

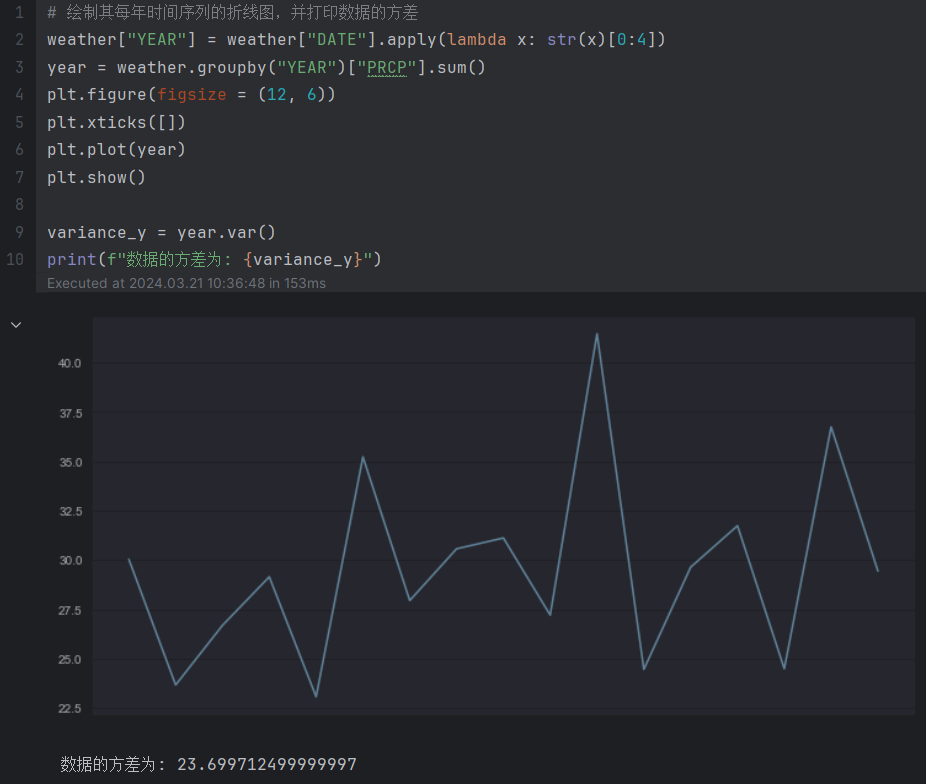


**图8 绘制原始数据图像**

但这里的数据包括了每一天的数据，数据量非常大，有时候在分析气候时我们不需要细化到每一天的程度，因此我们可以进行数据聚合。我们通过将日期转换为标准格式，对日期进行切片后使用groupby方法，可以得到聚合后的数据。图9展示了每月时间序列的折线图，图10展示了每年时间序列的折线图。



**图9 按月份聚合的图像**

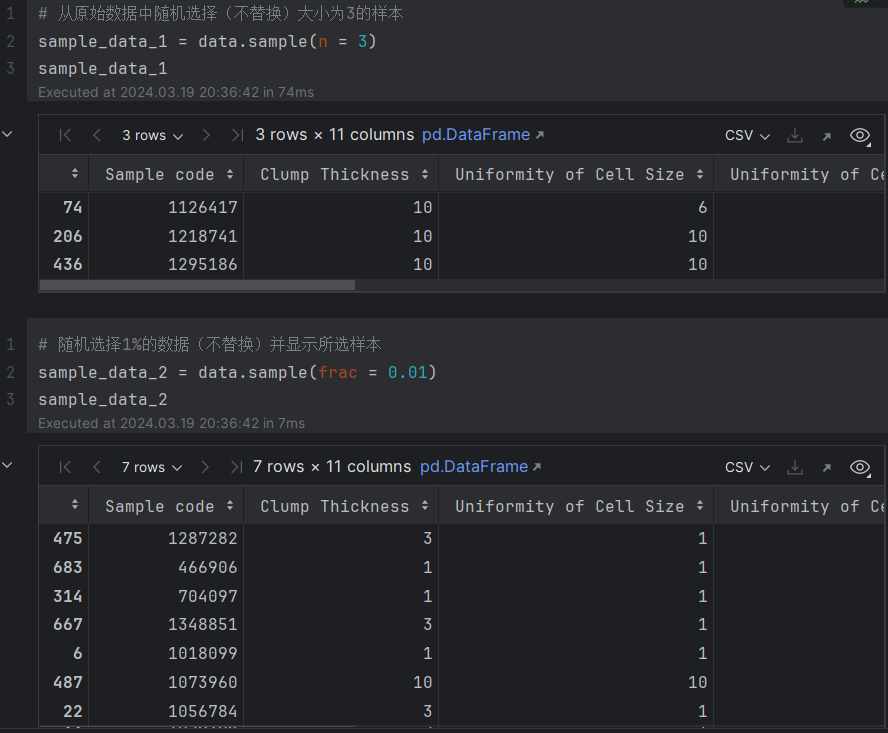


**图10 按年份聚合的图像**

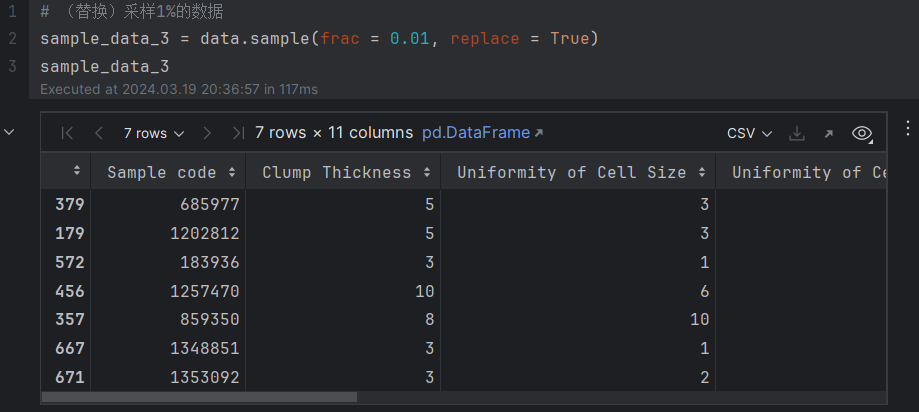
**4.3 采样**

数据过多会影响我们分析数据的效率，有时候可能只需要提取部分就足以掌握数据特征，这样的提取操作即为采样。在这里，我们将使用breast-cancer-wisconsin.data来实现采样。

我们使用了data.sample(n=3)来获取大小为3的样本，使用data.sample(frac = 0.01)来获取大小为整体1%的样本。以上的采样均为不替换采样，即选定的实例会从数据集中删除，以确保每次得到的都是不一样的数据，如图11所示。而如果我们希望使用替换采样，即选定的实例不会从数据集中删除，我们只需要添加参数replace=True即可，如图12所示。



**图11 不替换采样**

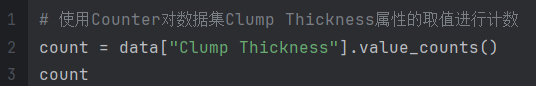


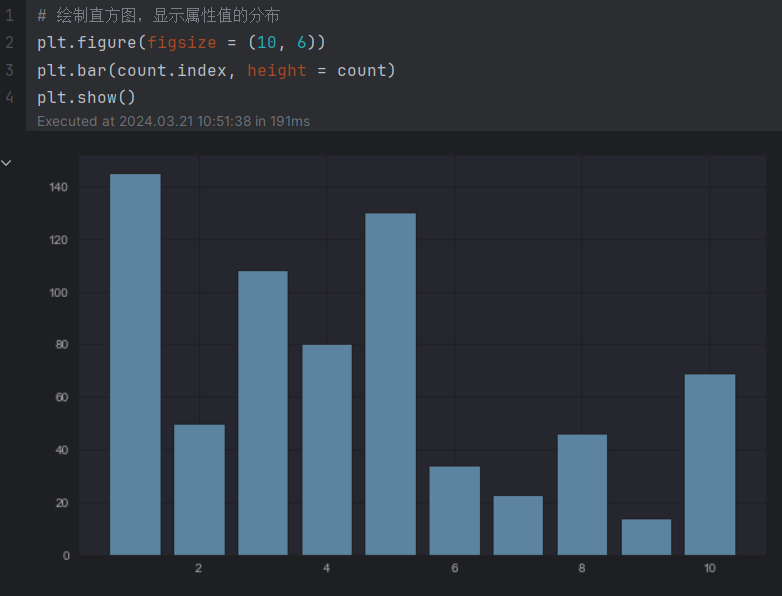
**图12 替换采样**

**4.4 离散化**

有时候，我们希望可以将连续属性转换为分类属性。在这里，我们使用breast-cancer-wisconsin.data来完成离散化的实验。

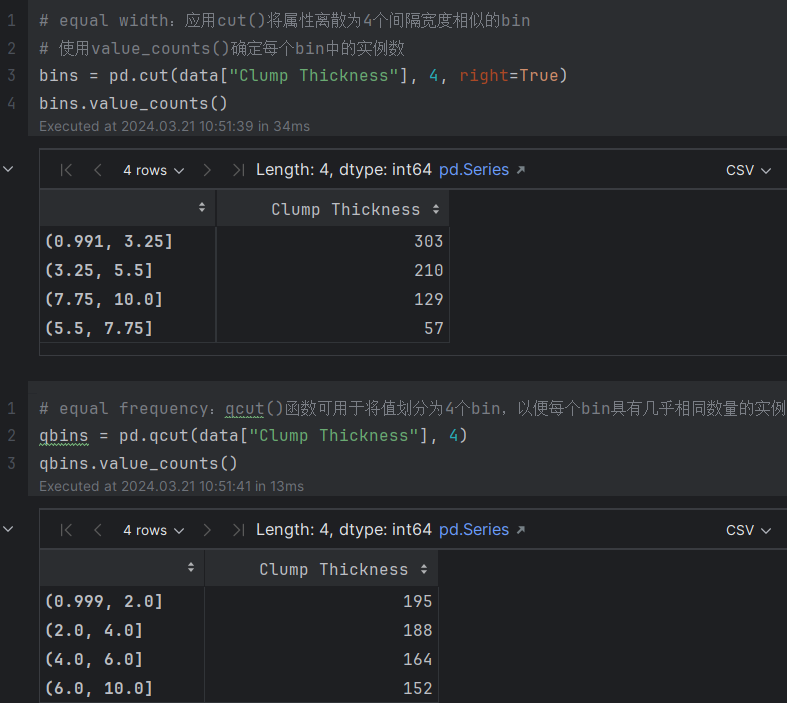
我们提取数据集中的Clump Thickness列，并且对其中的每个值使用value\_counts()进行计数，再利用plt.bar来绘制直方图观看数据分布。具体操作如图13所示。





**图13 观察数据分布**

接下来，我们使用两种离散化方法。一种是pd.cut()，它可以将属性离散为多个间隔宽度相似的bin，另一种是pd.qcut()，可以将值划分为多个bin，每个bin具有几乎相同数量的实例。效果如图14所示。



**图14 数据离散化**

注意到，我们在使用pd.cut()的时候，最小的区间左侧数值为0.991，这是一个比较异常的现象。通过阅读源码def cut，我们发现这主要是因为，最后显示给我们的区间必然是左开右闭或左闭右开区间，这意味着我们需要对其中的最小值或最大值的边界进行调整，从而确保所有区间可以容纳所有数据。在源码中，我们发现当列表中不存在无穷大的数值，列表的最大值与最小值不相等的时候，我们会执行正常的离散化操作，而为了可以让端点处被取到，我们会计算最大值与最小值的差的0.001倍作为调整量，如果我们计算的是左开右闭区间，那么我们就调整最左侧的端点，如果我们计算的是左闭右开区间，那么就调整最右侧的端点。在我们使用的数据中，最小值为1，最大值为10，因此调整量即为0.009，故最左侧的端点为1-0.009=0.991，这就解释了为什么最左侧的数值为0.991.

如果希望最左侧的数值好看一些，比如变为0.99，可以再添加一个参数include\_lowest=True，这可以让区间包括列表的最左边界。效果如图15.

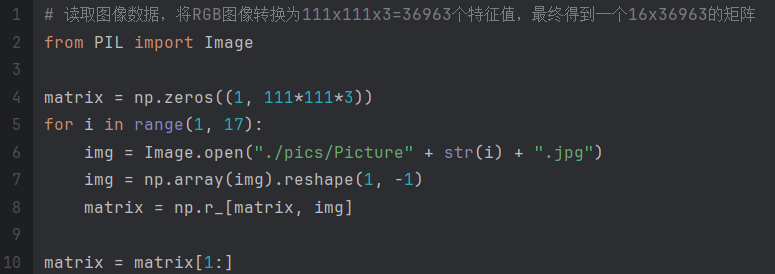


**图15 离散化的优化**

**4.5 主成分分析**

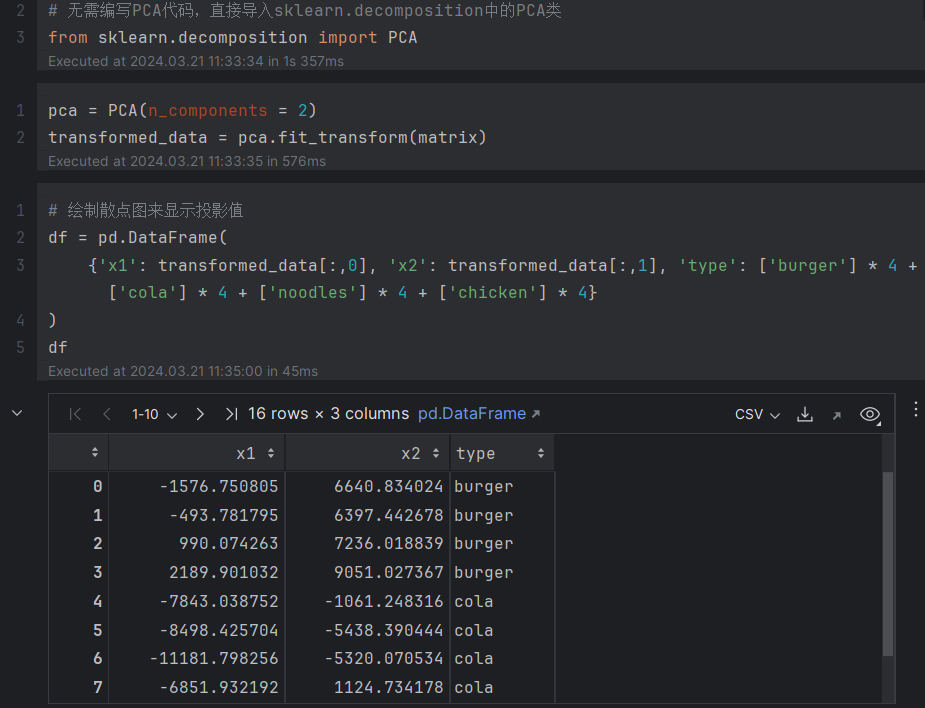
主成分分析是一种通过将数据从其原始高维空间投影到低维空间来减少数据中属性数量的经典方法。在这里，我们将使用16张图片进行主成分分析。

我们首先将16张图片拼接在一起，得到一个16\*36963规模的矩阵，代码如图16所示。



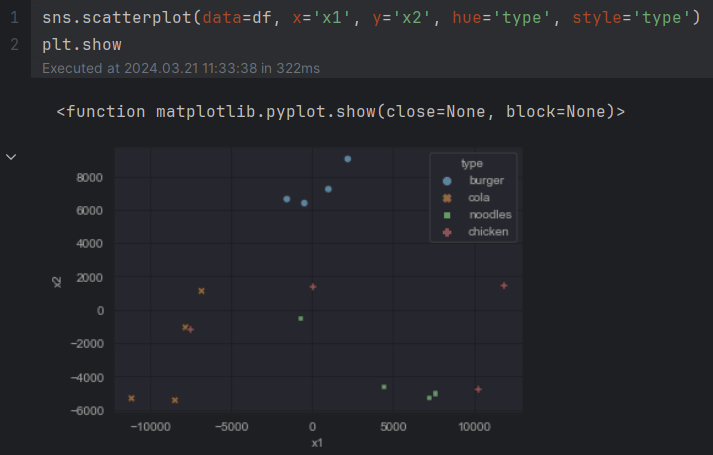
**图16 拼接图片**

接下来我们调用sklearn.decomposition中的PCA模块，对矩阵进行主成分的提取。在这里，我们只提取2个主成分，这让我们得到了一个16\*2的矩阵。在得到该矩阵后，我们希望为每一个数据设置上标签，因此我们为16张图片分别标上了“汉堡”、“可乐”、“面条”与“鸡肉”的标签。最终得到的数据如图17所示。



**图17 图片拼接处理后的数据**

最后，我们对得到的数据进行散点图的绘制。利用seaborn模块，我们可以绘制出如图18的散点图，可以观察到大致上4类图像的特征被区分开来。



**图18 绘制主成分分析后的散点图**

1. 总结与分析

本次实验我们体验了部分数据预处理相关的操作。数据预处理是在进行数据分析与数据挖掘时非常重要的一步，如果数据没有进行过任何处理，最终可能导致挖掘出的特征不准确。因此，数据预处理是每次在进行数据分析与数据挖掘时最重要的前置步骤，是不可省略的。然而，Python进行数据预处理需要借助非常多的模块的帮助，这样的操作对我来说较为陌生，我也需要再进一步地提升对Python数据预处理操作的熟练程度。