**图片包含 游戏机, 房间, 食物

描述已自动生成 南昌大学实验报告**

学生姓名： 周华俊 学 号： 8008122013 专业班级： 软件工程2201班

实验类型：□ 验证■ 综合 ☑ 设计 □ 创新 实验日期： 2024.12.1 实验成绩：

**一、实验项目名称**

数据预处理

**二、实验目的**

1. 掌握pandas、sklearn、matplotlib库
2. 掌握数据质量、缺失、离群点、的概念
3. 掌握重复数据、融合、采样
4. 信息熵、信息增益的概念
5. 掌握PCA主成分分析法

**三、实验任务**

1.参照课堂讲解titanic案例，完成给定的年龄数据集（AgeDataset-V1.csv）的数据清洗与转换、相关性分析。

**四、主要仪器设备及耗材**

软件：pycharm。

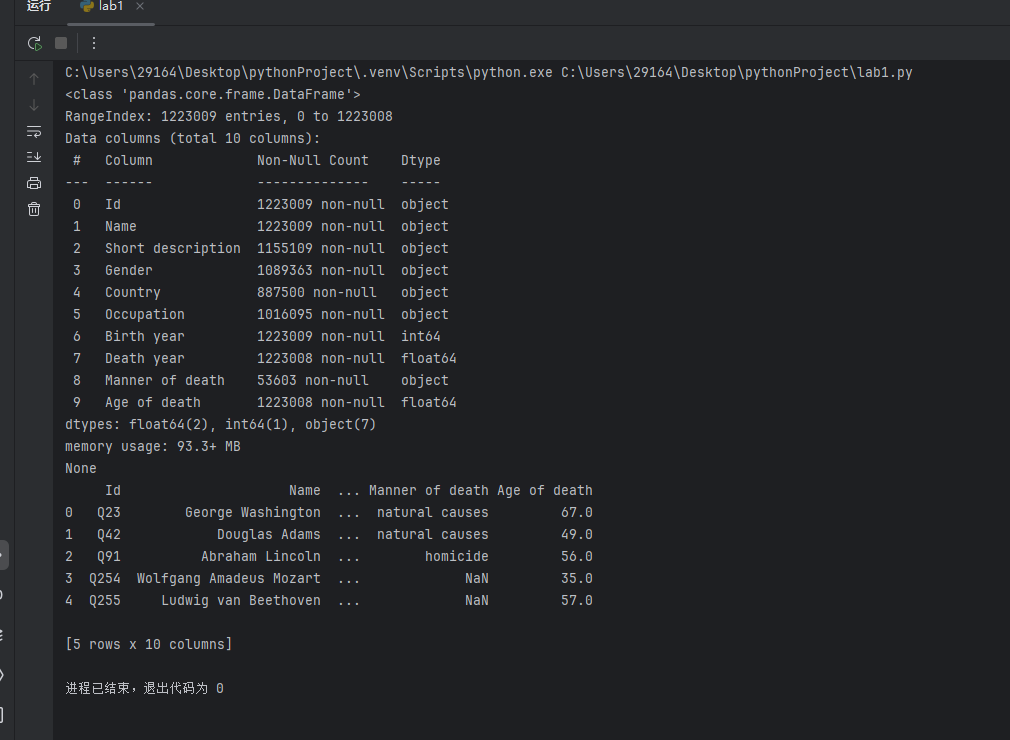
**五、实验步骤**

1. 导入库和加载数据

安装了 pandas、numpy、scikit-learn 和 matplotlib，并加载数据：

import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt  
import seaborn as sns  
from sklearn.decomposition import PCA  
from sklearn.preprocessing import StandardScaler  
  
# 加载数据  
data=pd.read\_csv(r'C:\ProgramFiles\JetBrains\fordaterevise\AgeDataset-V1.csv')  
# 查看数据基本信息  
print(data.info()) # 查看每列的类型和是否有缺失值  
print(data.head()) # 显示前几行数据，了解数据内容

运行结果如下：



2. 数据清洗与转换

接下来，我们将根据数据中的缺失值和数据类型问题进行清洗和转换。

2.1 处理缺失值

我们将处理以下几种类型的缺失值：

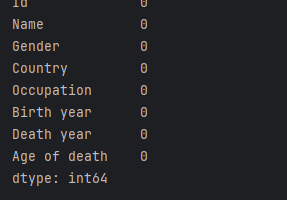
Age of death 列：这列是数值型，使用中位数填充缺失值。

Gender、Country、Occupation ，death yeat列：这些列是类别型数据，使用最常见值来填充缺失值。

Manner of death. Short description 列：缺失值非常多，删除这两列

输出填充后的结果确保填充完成

**# 填充数值型列（如 Age of death）缺失值  
data['Age of death'] = data['Age of death'].fillna(data['Age of death'].median())  
# 填充类别型列（如 Gender, Country, Occupation）缺失值，使用众数填充  
data['Gender'] = data['Gender'].fillna(data['Gender'].mode()[0])  
data['Country'] = data['Country'].fillna(data['Country'].mode()[0])  
data['Occupation'] = data['Occupation'].fillna(data['Occupation'].mode()[0])  
data['Death year'] = data['Death year'].fillna(data['Death year'].median())  
# 删除缺失值过多的列  
data.drop(columns=['Manner of death'], inplace=True)  
data.drop(columns=['Short description'], inplace=True)  
# 检查填充后的结果  
print(data.isnull().sum())**



2.2 删除重复数据

# 检查重复行

duplicate\_rows = data[data.duplicated()]

print(f"Number of duplicate rows: {len(duplicate\_rows)}")

# 删除重复行

data.drop\_duplicates(inplace=True)

2.3 数据转换

·计算年龄：

Age of death 列缺失部分通过计算 Death year - Birth year 填充，补全关键特征。

性别数值化：

将 Gender 转换为 0 和 1，便于后续的机器学习分析。

对类别型特征进行编码：

国家，职业

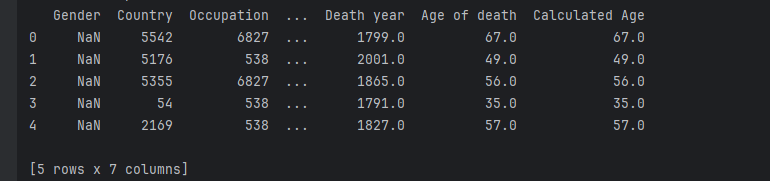
删除无关列：

删除 Id 和 Name，简化数据集以专注于关键变量。

# 计算年龄  
data['Calculated Age'] = data['Death year'] - data['Birth year']  
# 替换可能为负值或异常值的年龄为 NaN  
data.loc[data['Calculated Age'] < 0, 'Calculated Age'] = None  
# 将 'Gender' 转换为数值  
data['Gender'] = data['Gender'].map({'male': 0, 'female': 1})  
# 对类别型特征进行编码  
for col in ['Country', 'Occupation']:  
 if col in data.columns:  
 le = LabelEncoder()  
 data[col] = le.fit\_transform(data[col].astype(str))

# 删除无关列，例如 Id, Name  
data.drop(columns=['Id', 'Name'], inplace=True)

print(data.head()) # 检查当前数据集

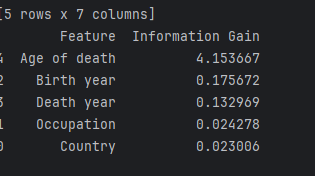


1. 计算信息增益

这里以先前计算的年龄为目标变量计算信息增益

如果 Calculated Age 是数值型，直接作为目标变量。如果有缺失值，先进行填充，提取特征和目标变量，计算信息增益

data['Calculated Age'] = data['Calculated Age'].fillna(data['Calculated Age'].median())  
X = data[['Country', 'Occupation', 'Birth year', 'Death year', 'Age of death']]  
y = data['Calculated Age'] # 目标变量  
# 计算信息增益  
info\_gain = mutual\_info\_regression(X, y)  
info\_gain\_df = pd.DataFrame({  
 'Feature': X.columns,  
 'Information Gain': info\_gain  
}).sort\_values(by='Information Gain', ascending=False)  
print(info\_gain\_df)



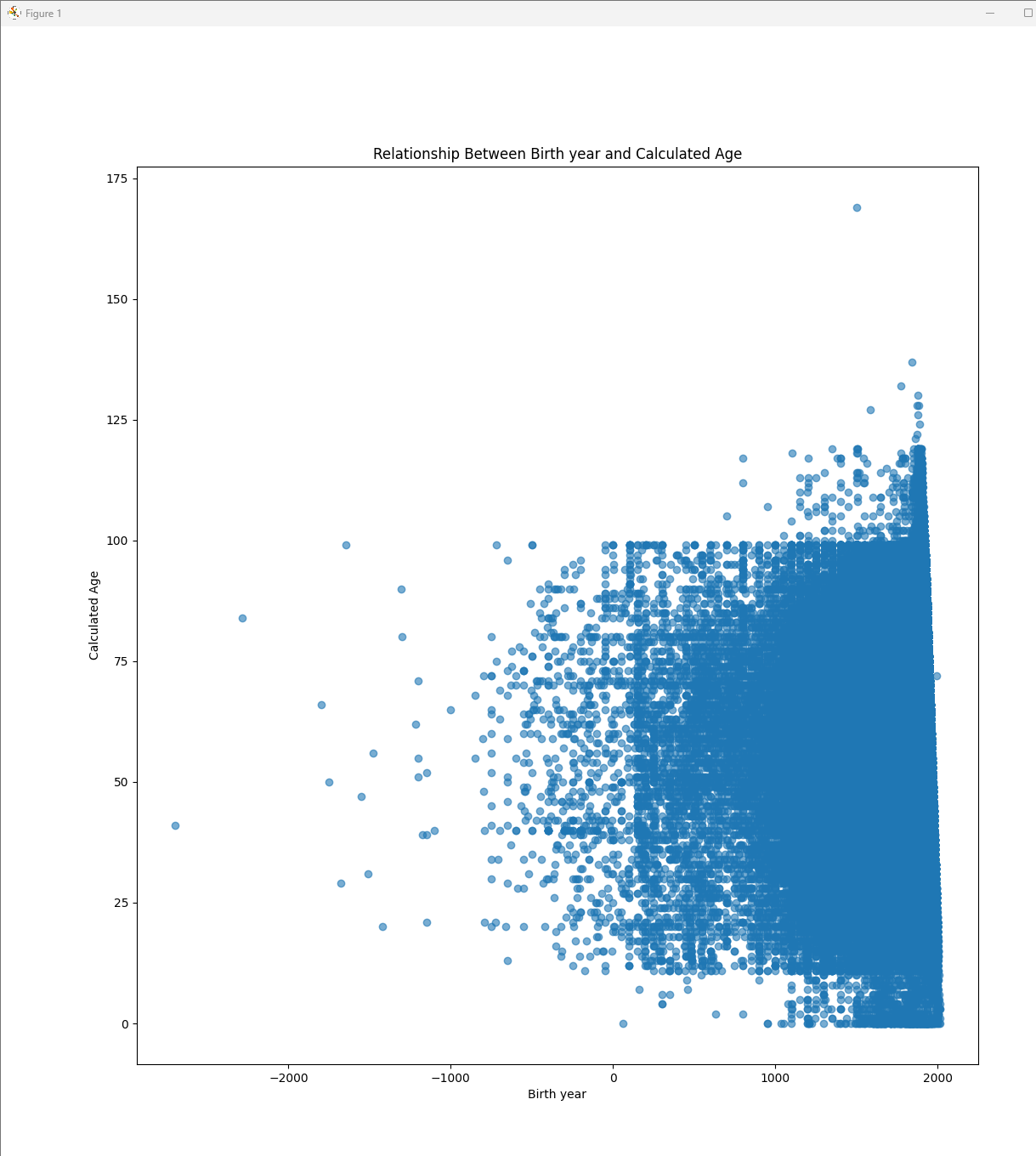
看到结果突然反应过来先前计算的值就是原数据集中的age of death 哈哈哈哈

信息增益值反映了每个特征对目标变量 Calculated Age 的影响：

较高的信息增益（如 Age of death 和 Birth year）表明这些特征与 Calculated Age 关联性较强。

较低的信息增益（如 Country 和 Occupation）说明这些特征对目标变量的影响较小。

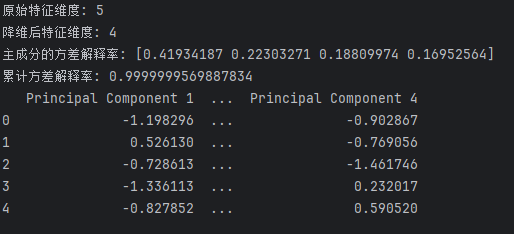
这里可视化信息增熵最高的birth year和目标变量的关系

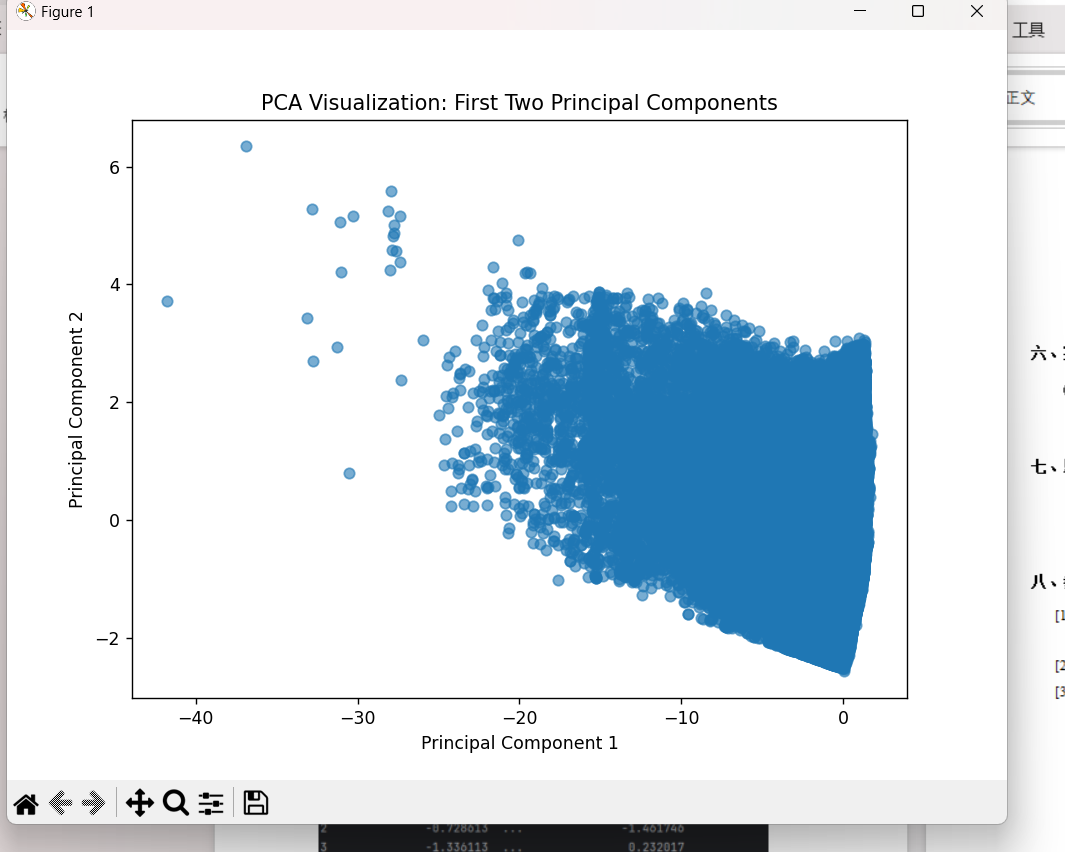


1. 主成分分析
2. 确保数据只包含数值型特征
3. 检查数据是否存在空值
4. 去掉空值
5. PCA 对数据的尺度非常敏感，因此需要对特征进行标准化处理
6. 初始化 PCA 并选择主成分数目，这里保留 95% 的累计方差
7. 主成分的方差解释率可以帮助了解每个主成分的重要性
8. 可以将降维后的主成分与原始特征进行比较或进一步分析
9. 如果降维后的维度大于 2，可以绘制前两个主成分的散点图

# 确保数据只包含数值型特征

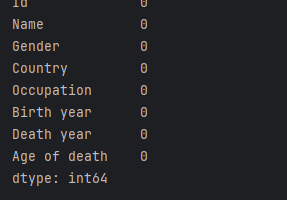
X = data[[ 'Country', 'Occupation', 'Birth year', 'Death year', 'Age of death']]  
# 检查数据是否存在空值  
print(X.isnull().sum())  
# 去掉空值  
X = X.dropna()  
#PCA 对数据的尺度非常敏感，因此需要对特征进行标准化处理  
scaler = StandardScaler()  
X\_scaled = scaler.fit\_transform(X)  
#初始化 PCA 并选择主成分数目，这里保留 95% 的累计方差  
pca = PCA(n\_components=0.95) # 保留累计方差达 95% 的主成分  
X\_pca = pca.fit\_transform(X\_scaled)  
print(f'原始特征维度: {X\_scaled.shape[1]}')  
print(f'降维后特征维度: {X\_pca.shape[1]}')  
#主成分的方差解释率可以帮助了解每个主成分的重要性  
explained\_variance = pca.explained\_variance\_ratio\_  
print(f'主成分的方差解释率: {explained\_variance}')  
print(f'累计方差解释率: {sum(explained\_variance)}')  
#可以将降维后的主成分与原始特征进行比较或进一步分析  
# 创建主成分的 DataFrame  
principal\_components\_df = pd.DataFrame(  
 X\_pca,  
 columns=[f'Principal Component {i+1}' for i in range(X\_pca.shape[1])]  
)  
print(principal\_components\_df.head())  
#如果降维后的维度大于 2，可以绘制前两个主成分的散点图  
plt.figure(figsize=(8, 6))  
plt.scatter(X\_pca[:, 0], X\_pca[:, 1], alpha=0.6)  
plt.xlabel('Principal Component 1')  
plt.ylabel('Principal Component 2')  
plt.title('PCA Visualization: First Two Principal Components')  
plt.show()



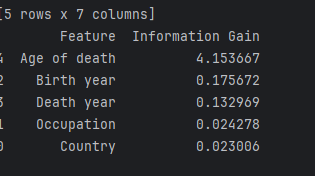


**六、实验数据及处理结果**

填充后结果：



信息增益计算结果;



**七、思考讨论题或体会或对改进实验的建议**

在数据预处理的过程中，涉及了数据清洗、缺失值处理、重复数据处理、数据转换、信息增益计算以及主成分分析等步骤。在这个实验中，可以思考以下问题：

数据清洗对最终分析结果有多大影响？清洗数据的过程中，采取了填充缺失值、处理重复数据等措施，这些步骤是否足够确保数据质量和可靠性？

在信息增益的计算中，选择了哪些特征作为目标变量，并计算了它们对目标变量的影响程度。可以讨论不同特征对目标变量的影响大小，以及如何根据信息增益值来优化特征选择的过程。

主成分分析是一种常用的降维技术，通过保留主要信息来减少数据的复杂度。我们可以探讨降维后数据的可解释性和对后续分析的影响，以及如何选择合适的主成分数量来保留足够的信息。

对改进实验的建议：

在数据清洗过程中，可以考虑更多的填充缺失值方法，比如基于其他特征的相关性填充，以提高数据的完整性。

在信息增益计算中，可以尝试不同的特征组合，或者使用其他特征选择方法，如基于模型的特征选择，来获取更准确的特征重要性排序。

在主成分分析中，可以尝试不同的主成分数量，比如根据累计方差解释率选择合适的主成分数量，以平衡降维效果和信息保留的程度。

**八、参考资料**

1. Pang-Ning Tan, Michael Steinbach etc. Introduction to Data Mining, Second Edition, Pearson, 2019.
2. 李锐，李鹏，曲亚东等译. 机器学习实战. 高等教育出版社.2013.
3. <http://www.cse.msu.edu/~ptan/dmbook/software/>.（课本实验代码）