山东大学 计算机科学与技术 学院

大数据分析实践 课程实验报告

队名：Four in Big Data	队长：何青青	学号：202300130262
队员 1：	刘逸宁	学号：202300130160
队员 2：	邱珺	学号：202300130063
队员 3：	徐瑞良	学号：202200120100
实验题目：数据采样方法实践		
实验学时：2	实验日期：2025/9/19	

实验目标：

利用 Pandas 库实现多种数据采样和过滤的方法

实验环境：

python3.9，Jupyter Notebook

实验步骤与内容：

1、进行库的导入与数据的读入，在这一步出现了问题，如图：

```
import pandas as pd
from pandas import DataFrame
import numpy as np

primitive_data=pd.read_csv("data.csv")
primitive_data
```

```
---------------------------------------------------------------------------
UnicodeDecodeError                        Traceback (most recent call last)
Cell In[4], line 5
      2 from pandas import DataFrame
      3 import numpy as np
----> 5 primitive_data=pd.read_csv("data.csv")
      6 primitive_data

File E:\Anaconda\Lib\site-packages\pandas\io\parsers\readers.py:948, in read_csv(filepath_or_buffer, sep, delimiter, header, names,
index_col, usecols, dtype, engine, converters, true_values, false_values, skipinitialspace, skiprows, skipfooter, nrows, na_values,
```

发现这是由于文件编码导致的问题，pandas 默认使用 utf-8 编码读取 CSV 文件，与 data.csv 文件使用的编码格式不符，因而解码失败。

为解决这一问题，引入 chardet 库进行自动检测编码

```
import chardet
import pandas as pd
from pandas import DataFrame
import numpy as np

# 检测文件编码
with open('data.csv', 'rb') as f:
    result = chardet.detect(f.read())

# 使用检测到的编码读取数据
primitive_data = pd.read_csv('data.csv', encoding=result['encoding'])
primitive_data
```

实验结果如下所示：

[5]:

	from_dev	from_port	from_city	from_level	to_dev	to_port	to_city	to_level	traffic	bandwidth
0	47	71	通辽	一般节点	1756	585	北京	网络核心	49636052613	1.000000e+11
1	47	74	通辽	一般节点	1756	776	北京	网络核心	50056871412	1.000000e+11
2	47	240	通辽	一般节点	1756	802	北京	网络核心	49453581081	1.000000e+11
3	47	241	通辽	一般节点	1997	464	天津	网络核心	49733361585	1.000000e+11
4	47	242	通辽	一般节点	474	672	哈尔滨	一般节点	50492573662	1.000000e+11
...	...	...	...	...	...	...	...	...	...	...
1113	1129	546	上海	网络核心	2050	502	石家庄	网络核心	48731433404	1.000000e+11
1114	1129	514	上海	网络核心	2473	946	吉林	一般节点	50060666120	1.000000e+11
1115	36036	499	长春	一般节点	1257	178	上海	网络核心	50545082113	1.000000e+11
1116	36422	346	天津	网络核心	1997	41	天津	网络核心	50628787089	1.000000e+11
1117	2701	619	大连	网络核心	2549	1070	沈阳	网络核心	48753971761	1.000000e+11

1118 rows × 10 columns

观察到数据底部有较多的空行

2、删除多余的空行并进行过滤。采用 dropna 方法并指定参数为 any 删除多余的空行。代码及实验结果如下。

[6]:
```
primitive_data_1=primitive_data.dropna(how='any')
primitive_data_1
```

[6]:

	from_dev	from_port	from_city	from_level	to_dev	to_port	to_city	to_level	traffic	bandwidth
0	47	71	通辽	一般节点	1756	585	北京	网络核心	49636052613	1.000000e+11
1	47	74	通辽	一般节点	1756	776	北京	网络核心	50056871412	1.000000e+11
2	47	240	通辽	一般节点	1756	802	北京	网络核心	49453581081	1.000000e+11
3	47	241	通辽	一般节点	1997	464	天津	网络核心	49733361585	1.000000e+11
4	47	242	通辽	一般节点	474	672	哈尔滨	一般节点	50492573662	1.000000e+11
...	...	...	...	...	...	...	...	...	...	...
1113	1129	546	上海	网络核心	2050	502	石家庄	网络核心	48731433404	1.000000e+11
1114	1129	514	上海	网络核心	2473	946	吉林	一般节点	50060666120	1.000000e+11
1115	36036	499	长春	一般节点	1257	178	上海	网络核心	50545082113	1.000000e+11
1116	36422	346	天津	网络核心	1997	41	天津	网络核心	50628787089	1.000000e+11
1117	2701	619	大连	网络核心	2549	1070	沈阳	网络核心	48753971761	1.000000e+11

1118 rows × 10 columns

3、接下来过滤得到 traffic 不等于 0 且 from_level 为一般节点的数据

```
data_before_filter=primitive_data_1
data_after_filter_1=data_before_filter.loc[data_before_filter["traffic"]!=0]
data_after_filter_2=data_after_filter_1.loc[data_after_filter_1["from_level"]=='一般节点']
data_after_filter_2
```

[7]:

	from_dev	from_port	from_city	from_level	to_dev	to_port	to_city	to_level	traffic	bandwidth
0	47	71	通辽	一般节点	1756	585	北京	网络核心	49636052613	1.000000e+11
1	47	74	通辽	一般节点	1756	776	北京	网络核心	50056871412	1.000000e+11
2	47	240	通辽	一般节点	1756	802	北京	网络核心	49453581081	1.000000e+11
3	47	241	通辽	一般节点	1997	464	天津	网络核心	49733361585	1.000000e+11
4	47	242	通辽	一般节点	474	672	哈尔滨	一般节点	50492573662	1.000000e+11
...	...	...	...	...	...	...	...	...	...	...
1097	2473	1460	吉林	一般节点	591	586	绥化	一般节点	48409925693	1.000000e+11
1103	36036	18	长春	一般节点	3443	650	青岛	网络核心	48663350759	1.000000e+11
1104	63	6	通辽	一般节点	36036	20	长春	一般节点	50355678076	1.000000e+11
1107	36036	52	长春	一般节点	1129	171	上海	网络核心	49345226162	1.000000e+11
1115	36036	499	长春	一般节点	1257	178	上海	网络核心	50545082113	1.000000e+11

550 rows × 10 columns

4、对数据进行抽样。采取不同的采样方式采取 50 个样本并比较采样结果

①加权采样：to_level 的值为一般节点与网络核心的权重之比为 1 ： 5

```
data_before_sample=data_after_filter_2
columns=data_before_sample.columns
weight_sample=data_before_sample.copy()
weight_sample['weight']=0
for i in weight_sample.index:
    if weight_sample.at[i, 'to_level'] == '一般节点':
        weight = 1
    else:
        weight = 5
    weight_sample.at[i, 'weight'] = weight

weight_sample_finish=weight_sample.sample(n=50,weights='weight')
#data_before_sample=data_before_sample[columns]
weight_sample_finish=weight_sample[columns]
weight_sample_finish
```

:

	from_dev	from_port	from_city	from_level	to_dev	to_port	to_city	to_level	traffic	bandwidth
0	47	71	通辽	一般节点	1756	585	北京	网络核心	49636052613	1.000000e+11
1	47	74	通辽	一般节点	1756	776	北京	网络核心	50056871412	1.000000e+11
2	47	240	通辽	一般节点	1756	802	北京	网络核心	49453581081	1.000000e+11
3	47	241	通辽	一般节点	1997	464	天津	网络核心	49733361585	1.000000e+11
4	47	242	通辽	一般节点	474	672	哈尔滨	一般节点	50492573662	1.000000e+11
...	...	...	...	...	...	...	...	...	...	...
1097	2473	1460	吉林	一般节点	591	586	绥化	一般节点	48409925693	1.000000e+11
1103	36036	18	长春	一般节点	3443	650	青岛	网络核心	48663350759	1.000000e+11
1104	63	6	通辽	一般节点	36036	20	长春	一般节点	50355678076	1.000000e+11
1107	36036	52	长春	一般节点	1129	171	上海	网络核心	49345226162	1.000000e+11
1115	36036	499	长春	一般节点	1257	178	上海	网络核心	50545082113	1.000000e+11

550 rows × 10 columns

②随机抽样，从原数据中随机抽取 50 行样本

```
random_sample=data_before_sample
random_sample_finish=random_sample.sample(n=50)
random_sample_finish=random_sample_finish[columns]
random_sample_finish
```

	from_dev	from_port	from_city	from_level	to_dev	to_port	to_city	to_level	traffic	bandwidth
29	63	230	通辽	一般节点	2701	71	大连	网络核心	50037668767	1.000000e+11
724	4069	1205	宁波	一般节点	36036	52	长春	一般节点	50994646887	1.000000e+11
97	474	416	哈尔滨	一般节点	1257	178	上海	网络核心	50599061005	1.000000e+11
1115	36036	499	长春	一般节点	1257	178	上海	网络核心	50545082113	1.000000e+11
1103	36036	18	长春	一般节点	3443	650	青岛	网络核心	48663350759	1.000000e+11
614	180	252	呼和浩特	一般节点	36036	52	长春	一般节点	49966571450	1.000000e+11
539	47	425	通辽	一般节点	3227	787	济南	网络核心	49862690338	1.000000e+11
1063	47	314	通辽	一般节点	47	252	通辽	一般节点	49900452417	1.000000e+11
644	47	314	通辽	一般节点	96	152	呼和浩特	一般节点	51384841448	1.000000e+11

③分层抽样：根据 to_level 的值进行分层采样。根据比例一般节点抽 17 个，网络核心抽 33 个：

```
ybjd=data_before_sample.loc[data_before_sample['to_level']=='一般节点']
wlhx=data_before_sample.loc[data_before_sample['to_level']=='网络核心']
after_sample=pd.concat([ybjd.sample(17),wlhx.sample(33)])
after_sample
```

	from_dev	from_port	from_city	from_level	to_dev	to_port	to_city	to_level	traffic	bandwidth
383	474	670	哈尔滨	一般节点	5058	144	南宁	一般节点	50998204735	1.000000e+11
151	591	586	绥化	一般节点	180	192	呼和浩特	一般节点	49061517661	1.000000e+11
493	47	251	通辽	一般节点	63	286	通辽	一般节点	49254299513	1.000000e+11
298	63	62	通辽	一般节点	474	683	哈尔滨	一般节点	50533229665	1.000000e+11
392	474	1228	哈尔滨	一般节点	96	134	呼和浩特	一般节点	51278220999	1.000000e+11
760	5058	70	南宁	一般节点	96	460	呼和浩特	一般节点	49703011825	1.000000e+11
984	4360	468	南京	一般节点	96	108	呼和浩特	一般节点	51700762978	1.000000e+11

④系统抽样：确定数据总量 N 和抽样间隔后，随机选择一个起始点，然后按间隔 k 从起始点开始选取样本量大小的数据，最终返回抽取 50 个的样本。

```
def systematic_sampling(data, sample_size):
    N = len(data)
    k = N // sample_size
    start = np.random.randint(0, k)
    indices = np.arange(start, N, k)[:sample_size]
    sample = data.iloc[indices]
    return sample

systematic_sample = systematic_sampling(primitive_data,50)
systematic_sample
```

	from_dev	from_port	from_city	from_level	to_dev	to_port	to_city	to_level	traffic	bandwidth
6	47	249	通辽	一般节点	1997	85	天津	网络核心	50499586948	1.000000e+11
28	63	228	通辽	一般节点	235	1506	北京	网络核心	49436165249	1.000000e+11
50	96	155	呼和浩特	一般节点	1536	681	广州	网络核心	51538493830	1.000000e+11
72	180	42	呼和浩特	一般节点	36539	1140	杭州	一般节点	49293665157	1.000000e+11
94	180	485	呼和浩特	一般节点	36422	102	天津	网络核心	52460156321	1.000000e+11
116	474	1227	哈尔滨	一般节点	2841	341	郑州	网络核心	48505909225	1.000000e+11
138	591	27	绥化	一般节点	3443	117	青岛	网络核心	49213859972	1.000000e+11
160	591	1258	绥化	一般节点	4448	127	无锡	一般节点	50322958171	1.000000e+11
182	1997	38	天津	网络核心	3227	766	济南	网络核心	49896147267	1.000000e+11

⑤整群抽样：获取数据中指定聚类列的所有唯一聚类值，随机选取指定数量的聚类，然后提取这些选中聚类中的所有数据作为样本。最后调用该函数从 primitive_data 中按‘from_city’列随机选取 3 个城市的所有数据作为样本。

```
def cluster_sampling(data, num_clusters, cluster_column):
    unique_clusters = data[cluster_column].unique()
    selected_clusters = np.random.choice(unique_clusters, num_clusters, replace=False)
    sample = data[data[cluster_column].isin(selected_clusters)]
    return sample

cluster_sample = cluster_sampling(primitive_data, 3, 'from_city')
cluster_sample
```

	from_dev	from_port	from_city	from_level	to_dev	to_port	to_city	to_level	traffic	bandwidth
567	2701	300	大连	网络核心	36036	499	长春	一般节点	50207589348	1.000000e+11
571	2701	135	大连	网络核心	2194	264	唐山	网络核心	49363764010	1.000000e+11
583	2473	946	吉林	一般节点	36539	1146	杭州	一般节点	50631070410	1.000000e+11
588	3227	344	济南	网络核心	4953	725	贵阳	一般节点	50925677715	1.000000e+11
593	2473	803	吉林	一般节点	3227	705	济南	网络核心	49383348895	1.000000e+11
...	...	...	...	...	...	...	...	...	...	...
1094	2701	300	大连	网络核心	235	1958	北京	网络核心	50541875029	1.000000e+11
1097	2473	1460	吉林	一般节点	591	586	绥化	一般节点	48409925693	1.000000e+11
1106	2701	227	大连	网络核心	235	1663	北京	网络核心	49364931565	1.000000e+11
1111	2701	195	大连	网络核心	36272	235	太原	网络核心	49757975617	1.000000e+11
1117	2701	619	大连	网络核心	2549	1070	沈阳	网络核心	48753971761	1.000000e+11

82 rows × 10 columns

结论分析与体会：

本次数据采样实践实验，提升了我们使用 Pandas 处理数据的实操能力，让我们面对不同的分析需求，思考如何选择更合适的抽样方法、如何处理数据中的异常情况。同时，实验中遇到的问题也让我们学会了查阅文档、调试代码的方法，为后续更复杂的大数据分析实验打下了坚实基础。