Python与金融数据挖掘(12)

文欣秀

wenxinxiu@ecust.edu.cn

Python应用领域

科学计算： Numpy、SciPy…

数据分析： Pandas、Matplotlib…

机器学习： Scikit-Learn、Keras…

深度学习： Pytorch、Mindspore…

…

Pandas读CSV文档并存储部分数据

```
import pandas as pd

data=pd. read_csv('client.csv', index_col=0)

result=data. head()

result. to_csv("part.csv")
```

	A	B	C	D	E
1	No.	Gender	Age	Height	Weight
2	202201	male	20	170	70
3	202202	male	22	180	71
4	202203	male	21	180	62
5	202204	male	20	177	72
6	202205	male	20	172	64
7	202206	male	20	179	75
8	202207	female	21	166	53
9	202208	female	20	162	47
10	202209	female	20	162	47
11	202210	male	19	169	76
12	202211	female	21	162	49

	A	B	C	D	E
1	No.	Gender	Age	Height	Weight
2	202201	male	20	170	70
3	202202	male	22	180	71
4	202203	male	21	180	62
5	202204	male	20	177	72
6	202205	male	20	172	64

Pandas读EXCEL文档并存储部分数据

>>> **pip install openpyxl #安装第三方库**

```
import pandas as pd

data=pd.read_excel("info.xlsx","Group1",index_col=0)

result=data.tail(3)

result.to_excel("analysis.xlsx")
```

	A	B	C	D	E	F	G	H
1	序号	性别	年龄	身高	体重	省份	成绩	月生活费
2	21	female	21	165	45	Shanghai	93	1200
3	22	female	19	167	42	HuBei	89	800
4	23	male	21	169	80	GanSu	93	900
5	24	female	21	160	49	HeBei	59	1100
6	25	female	21	162	54	GanSu	68	1300
7	26	male	21	181	77	SiChuan	62	800
8	27	female	21	162	49	ShanDong	65	950
9	28	female	22	160	52	ShanXi	73	800
10	29	female	20	161	51	GuangXi	80	1250
11	30	female	20	168	52	JiangSu	98	700

	A	B	C	D	E	F	G	H
1	序号	性别	年龄	身高	体重	省份	成绩	月生活费
2	28	female	22	160	52	ShanXi	73	800
3	29	female	20	161	51	GuangXi	80	1250
4	30	female	20	168	52	JiangSu	98	700

Pandas读EXCEL文档并存储部分数据

```
import pandas as pd
data=pd.read_excel("info.xlsx","Group1",index_col=0)
result=data. sample()     #任意取1条
print(result)
result=data. sample(5)    #任意取5条
print(result)
```

	A	B	C	D	E	F	G	H
1	序号	性别	年龄	身高	体重	省份	成绩	月生活费
2	21	female	21	165	45	Shanghai	93	1200
3	22	female	19	167	42	HuBei	89	800
4	23	male	21	169	80	GanSu	93	900
5	24	female	21	160	49	HeBei	59	1100
6	25	female	21	162	54	GanSu	68	1300
7	26	male	21	181	77	SiChuan	62	800
8	27	female	21	162	49	ShanDong	65	950
9	28	female	22	160	52	ShanXi	73	800
10	29	female	20	161	51	GuangXi	80	1250
11	30	female	20	168	52	JiangSu	98	700

```
      性别  年龄  身高  体重    省份  成绩  月生活费
序号
22  female  19  167  42  HuBei  89  800
```

```
      性别  年龄  身高  体重      省份  成绩  月生活费
序号
23    male  21  169  80    GanSu  93  900
29  female  20  161  51  GuangXi  80  1250
22  female  19  167  42    HuBei  89  800
25  female  21  162  54    GanSu  68  1300
28  female  22  160  52   ShanXi  73  800
```

Pandas常用统计函数

函数	描述
df.mean()	计算样本数据的算术平均值
df.value_counts()	统计频数
df.describe()	返回基本统计量和分位数
df.corr(sr)	df与sr的相关系数
df.count()、df.sum()	统计每列(或行)数据的个数或总和
df.max()、df.min()	最大值和最小值
df.idxmax()、df.idxmin()	最大值、最小值对应的索引
df.qantile()	计算给定的四分位数
df.var()、df.std()	计算方差、标准差
df.mode()	计算众数
df.cov()	计算协方差矩阵

股价相关性分析

```
import pandas as pd
import tushare as ts
ts.set_token('XXX')
pro = ts. pro_api()
df=pro.daily(ts_code='601398.SH', start_date='20250301',end_date='20250331')
correlation = df['close'].corr(df['open'])
print(f'Correlation between Close and Open: {correlation}')
```

```
Correlation between Close and Open: 0.6185131598503488
```

|r|<0.4为低相关；0.4=<|r|<0.7为中等相关，|r|>=0.7为高相关

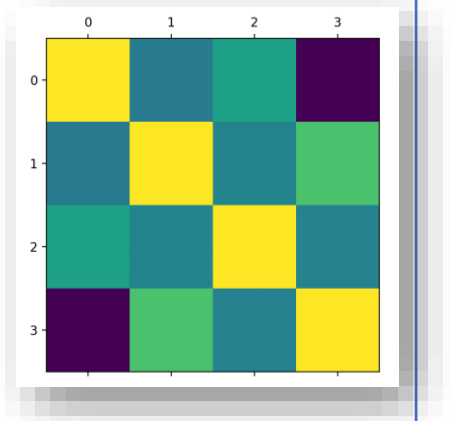
股价相关性分析

```
import pandas as pd
import tushare as ts
ts.set_token('XXX')
pro = ts. pro_api()
df=pro.daily(ts_code='601398.SH', start_date='20250301',end_date='20250331')
correlation = df[['open','high','low','close']].corr()
print(f'{correlation}')
```

	open	high	low	close
open	1.000000	0.777229	0.835987	0.618513
high	0.777229	1.000000	0.790285	0.890275
low	0.835987	0.790285	1.000000	0.786927
close	0.618513	0.890275	0.786927	1.000000

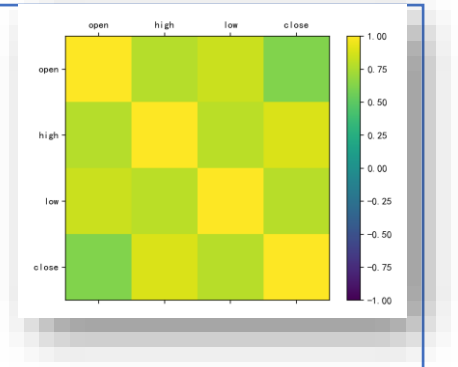
股价相关性分析图示

```
import pandas as pd
import tushare as ts
import matplotlib.pyplot as plt
ts.set_token('XXX')
pro = ts. pro_api()
df=pro.daily(ts_code='601398.SH', start_date='20250301',end_date='20250331')
correlation = df[['open','high','low','close']].corr()
plt.matshow(correlation) #相关矩阵图展示两个不同属性相互影响的程度
plt.show()
```



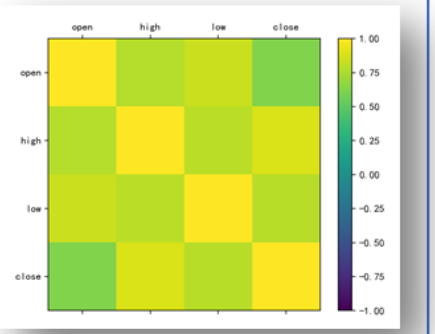
股价相关性分析（拓展）

```
import matplotlib.pyplot as plt
import pandas as pd
import tushare as ts
import numpy as np
plt.rcParams['axes.unicode_minus'] = False    #显示负号
ts.set_token('XXX')
pro = ts. pro_api()
df=pro.daily(ts_code='601398.SH',start_date='20250301',end_date='20250331')
correlation = df[['open','high','low','close']].corr()
```



股价相关性分析（拓展）

```
fig=plt.figure()
ax=fig.add_subplot(111)
cax=ax.matshow(correlation, vmin=-1, vmax=1)    #相关矩阵图
fig.colorbar(cax)
ticks=np.arange(0,4,1)
names=['open','high','low','close']
ax.set_xticks(ticks); ax.set_yticks(ticks)
ax.set_xticklabels(names); ax.set_yticklabels(names)
plt.show()
```



数据排序

- 按索引排序

```
import pandas as pd
data=pd.read_excel("info.xlsx","Group1",index_col=0)
#按行索引降序排序
result1=data.sort_index(ascending=False)
print(result1)
```

序号	性别	年龄	身高	体重	省份	成绩	月生活费
30	female	20	168	52	JiangSu	98	700
29	female	20	161	51	GuangXi	80	1250
28	female	22	160	52	ShanXi	73	800
27	female	21	162	49	ShanDong	65	950
26	male	21	181	77	SiChuan	62	800
25	female	21	162	54	GanSu	68	1300
24	female	21	160	49	HeBei	59	1100
23	male	21	169	80	GanSu	93	900
22	female	19	167	42	HuBei	89	800
21	female	21	165	45	Shanghai	93	1200

股价按索引排序

```
import pandas as pd
import tushare as ts
ts.set_token('XXX')
pro = ts. pro_api()
df=pro.daily(ts_code='601398.SH',start_date='20250301',end_date='20250331')
result1=df.sort_index(ascending=False)
print(result1)
```

	ts_code	trade_date	open	high	...	change	pct_chg	vol	amount
20	601398.SH	20250303	6.87	6.89	...	-0.02	-0.2911	3964019.25	2705652.927
19	601398.SH	20250304	6.82	6.84	...	-0.07	-1.0219	3436561.71	2336781.175
18	601398.SH	20250305	6.77	6.90	...	0.09	1.3274	4599123.60	3147082.337
17	601398.SH	20250306	6.85	6.86	...	-0.04	-0.5822	3656712.05	2495349.517
16	601398.SH	20250307	6.83	6.85	...	-0.03	-0.4392	3223755.78	2195284.348
15	601398.SH	20250310	6.79	6.80	...	-0.06	-0.8824	3242116.47	2186689.995
14	601398.SH	20250311	6.72	6.78	...	0.04	0.5935	2680002.41	1809114.543
13	601398.SH	20250312	6.76	6.77	...	-0.06	-0.8850	3177702.18	2135384.159
12	601398.SH	20250313	6.72	6.77	...	0.01	0.1488	2572950.17	1734795.847
11	601398.SH	20250314	6.72	6.81	...	0.02	0.2972	4890721.29	3311312.295
10	601398.SH	20250317	6.75	6.83	...	0.06	0.8889	3586198.28	2436756.611
9	601398.SH	20250318	6.83	6.84	...	-0.01	-0.1468	2369528.10	1611444.863
8	601398.SH	20250319	6.81	6.88	...	0.07	1.0294	3032694.20	2077764.026
7	601398.SH	20250320	6.87	6.92	...	-0.03	-0.4367	3008452.41	2058985.286
6	601398.SH	20250321	6.82	6.85	...	-0.08	-1.1696	3087991.23	2096634.446
5	601398.SH	20250324	6.76	6.85	...	0.06	0.8876	3060401.69	2084392.843
4	601398.SH	20250325	6.83	6.87	...	0.03	0.4399	2318056.72	1587419.428
3	601398.SH	20250326	6.86	6.88	...	-0.02	-0.2920	2337800.88	1598912.190
2	601398.SH	20250327	6.82	6.89	...	0.04	0.5857	2580278.10	1771784.348
1	601398.SH	20250328	6.86	6.88	...	0.01	0.1456	2128381.48	1458086.737
0	601398.SH	20250331	6.88	6.94	...	0.01	0.1453	4071337.79	2804461.039

[21 rows x 11 columns]

数据排序

- 按值排序

```
import pandas as pd
data=pd.read_excel("info.xlsx","Group1",index_col=0)
result2=data.sort_values(by='成绩', ascending=False)
print(result2)
result3=data.sort_values(by=['身高','体重'], ascending=True)
print(result3)
```

序号	性别	年龄	身高	体重	省份	成绩	月生活费
30	female	20	168	52	JiangSu	98	700
21	female	21	165	45	Shanghai	93	1200
23	male	21	169	80	GanSu	93	900
22	female	19	167	42	HuBei	89	800
29	female	20	161	51	GuangXi	80	1250
28	female	22	160	52	ShanXi	73	800
25	female	21	162	54	GanSu	68	1300
27	female	21	162	49	ShanDong	65	950
26	male	21	181	77	SiChuan	62	800
24	female	21	160	49	HeBei	59	1100

序号	性别	年龄	身高	体重	省份	成绩	月生活费
24	female	21	160	49	HeBei	59	1100
28	female	22	160	52	ShanXi	73	800
29	female	20	161	51	GuangXi	80	1250
27	female	21	162	49	ShanDong	65	950
25	female	21	162	54	GanSu	68	1300
21	female	21	165	45	Shanghai	93	1200
22	female	19	167	42	HuBei	89	800
30	female	20	168	52	JiangSu	98	700
23	male	21	169	80	GanSu	93	900
26	male	21	181	77	SiChuan	62	800

股价按值排序

```
import pandas as pd
import tushare as ts
ts.set_token('XXX')
pro = ts. pro_api()
df=pro.daily(ts_code='601398.SH',start_date='20250301',end_date='20250331')
result2=df.sort_values(by='amount', ascending=False)
print(result2)
```

	ts_code	trade_date	open	high	...	change	pct_chg	vol	amount
11	601398.SH	20250314	6.72	6.81	...	0.02	0.2972	4890721.29	3311312.295
18	601398.SH	20250305	6.77	6.90	...	0.09	1.3274	4599123.60	3147082.337
0	601398.SH	20250331	6.88	6.94	...	0.01	0.1453	4071337.79	2804461.039
20	601398.SH	20250303	6.87	6.89	...	-0.02	-0.2911	3964019.25	2705652.927
17	601398.SH	20250306	6.85	6.86	...	-0.04	-0.5822	3656712.05	2495349.517
10	601398.SH	20250317	6.75	6.83	...	0.06	0.8889	3586198.28	2436756.611
19	601398.SH	20250304	6.82	6.84	...	-0.07	-1.0219	3436561.71	2336781.175
16	601398.SH	20250307	6.83	6.85	...	-0.03	-0.4392	3223755.78	2195284.348
15	601398.SH	20250310	6.79	6.80	...	-0.06	-0.8824	3242116.47	2186689.995
13	601398.SH	20250312	6.76	6.77	...	-0.06	-0.8850	3177702.18	2135384.159
6	601398.SH	20250321	6.82	6.85	...	-0.08	-1.1696	3087991.23	2096634.446
5	601398.SH	20250324	6.76	6.85	...	0.06	0.8876	3060401.69	2084392.843
8	601398.SH	20250319	6.81	6.88	...	0.07	1.0294	3032694.20	2077764.026
7	601398.SH	20250320	6.87	6.92	...	-0.03	-0.4367	3008452.41	2058985.286
14	601398.SH	20250311	6.72	6.78	...	0.04	0.5935	2680002.41	1809114.543
2	601398.SH	20250327	6.82	6.89	...	0.04	0.5857	2580278.10	1771784.348
12	601398.SH	20250313	6.72	6.77	...	0.01	0.1488	2572950.17	1734795.847
9	601398.SH	20250318	6.83	6.84	...	-0.01	-0.1468	2369528.10	1611444.863
3	601398.SH	20250326	6.86	6.88	...	-0.02	-0.2920	2337800.88	1598912.190
4	601398.SH	20250325	6.83	6.87	...	0.03	0.4399	2318056.72	1587419.428
1	601398.SH	20250328	6.86	6.88	...	0.01	0.1456	2128381.48	1458086.737

数据排名

- 排名

```
import pandas as pd
data=pd.read_excel("info.xlsx","Group1",index_col=0)
#对成绩数据降序排名，增加“排名”列，method为并列名次取值
#比如（2，3名成绩相同，min取2,max取3）
data['排名'] = data['成绩'].rank(method='min', ascending=False).astype(int)
print( data )
```

序号	性别	年龄	身高	体重	省份	成绩	月生活费	排名
21	female	21	165	45	Shanghai	93	1200	2
22	female	19	167	42	HuBei	89	800	4
23	male	21	169	80	GanSu	93	900	2
24	female	21	160	49	HeBei	59	1100	10
25	female	21	162	54	GanSu	68	1300	7
26	male	21	181	77	SiChuan	62	800	9
27	female	21	162	49	ShanDong	65	950	8
28	female	22	160	52	ShanXi	73	800	6
29	female	20	161	51	GuangXi	80	1250	5
30	female	20	168	52	JiangSu	98	700	1

数据分组

- 按列分组

```
import pandas as pd
data=pd.read_excel("info.xlsx","Group1",index_col=0)
result=data.groupby('性别')['年龄'].count()
print(result)
```

```
性别
female    8
male      2
Name: 年龄, dtype: int64
```

数据清洗

	A	B	C	D	E	F	G	H
1	ID	Sex	Age	Height	Weight	Province	Score	Cost
2	1	male	20	170	70	LiaoNing		800
3	2	male	22	180	71	GuangXi	77	1300
4	3	male		180	62	FuJian	57	1000
5	4	male	20	177	72	LiaoNing	79	900
6	5	male	20	172		ShanDong	91	
7	6	male	20	179	75	YunNan	92	950
8								
9	7	female	21	166	53	LiaoNing	80	1200
10	8	female	20	162	47	AnHui	78	1000
11	9	female	20	162	47	AnHui	78	1000
12	10	male	120	169	76	HeiLongJiang	88	1100

数据清洗

数据清洗：对采集的数据进行重新审查和校验的过程，其目的在于删除重复信息、纠正存在的错误，保证数据的一致性。

常见问题：

- 数据缺失
- 数据重复
- 数据不一致

	A	B	C	D	E	F	G	H
1	序号	性别	年龄	身高	体重	省份	成绩	月生活费
2	1	male	20	170	70	LiaoNing		800
3	2	male	22	180	71	GuangXi	77	1300
4	3	male		180	62	FuJian	57	1000
5	4	male	20	177	72	LiaoNing	79	900
6	5	male	20	172		ShanDong	91	
7	6	male	20	179	75	YunNan	92	950
8								
9	7	female	21	166	53	LiaoNing	80	1200
10	8	female	20	162	47	AnHui	78	1000
11	9	female	20	162	47	AnHui	78	1000
12	10	male	120	169	76	HeiLongJiang	88	1100

数据清洗

丢弃缺失值dropna(axis,how,thresh,…)

axis： 0表示按行滤除，1表示按列滤除， 默认为axis=0

data. dropna()　　#每行只要有空值，就将该行删除

data. dropna(axis=1)　#每列只要有空值，就将该列删除

数据清洗案例

```
import pandas as pd
data=pd.read_excel("info.xlsx","Group2",index_col=0)
data1=data. dropna()   #默认按行删除
print(data1)
```

	A	B	C	D	E	F	G	H
1	序号	性别	年龄	身高	体重	省份	成绩	月生活费
2	1	male	20	170	70	LiaoNing		800
3	2	male	22	180	71	GuangXi	77	1300
4	3	male		180	62	FuJian	57	1000
5	4	male	20	177	72	LiaoNing	79	900
6	5	male	20	172		ShanDong	91	
7	6	male	20	179	75	YunNan	92	950
8								
9	7	female	21	166	53	LiaoNing	80	1200
10	8	female	20	162	47	AnHui	78	1000
11	9	female	20	162	47	AnHui	78	1000
12	10	male	120	169	76	HeiLongJiang	88	1100

序号	性别	年龄	身高	体重	省份	成绩	月生活费
2.0	male	22.0	180.0	71.0	GuangXi	77.0	1300.0
4.0	male	20.0	177.0	72.0	LiaoNing	79.0	900.0
6.0	male	20.0	179.0	75.0	YunNan	92.0	950.0
7.0	female	21.0	166.0	53.0	LiaoNing	80.0	1200.0
8.0	female	20.0	162.0	47.0	AnHui	78.0	1000.0
9.0	female	20.0	162.0	47.0	AnHui	78.0	1000.0
10.0	male	120.0	169.0	76.0	HeiLongJiang	88.0	1100.0

数据清洗案例

```
import pandas as pd
data=pd. read_excel("info.xlsx","Group2",index_col=0)
data1=data. dropna(axis=1) #按列删除
print(data1)
```

	A	B	C	D	E	F	G	H
1	序号	性别	年龄	身高	体重	省份	成绩	月生活费
2	1	male	20	170	70	LiaoNing		800
3	2	male	22	180	71	GuangXi	77	1300
4	3	male		180	62	FuJian	57	1000
5	4	male	20	177	72	LiaoNing	79	900
6	5	male	20	172		ShanDong	91	
7	6	male	20	179	75	YunNan	92	950
8								
9	7	female	21	166	53	LiaoNing	80	1200
10	8	female	20	162	47	AnHui	78	1000
11	9	female	20	162	47	AnHui	78	1000
12	10	male	120	169	76	HeiLongJiang	88	1100

```
Empty DataFrame
Columns: []
Index: [1.0, 2.0, 3.0, 4.0, 5.0, 6.0, nan, 7.0, 8.0, 9.0, 10.0]
```

数据清洗

丢弃缺失值dropna(axis,how,thresh,…)

how: "all"表示滤除全部值都为NaN的行或列

data. dropna(how='all')　#一行中全部为NaN才丢弃该行

数据清洗案例

```
import pandas as pd
data=pd.read_excel("info.xlsx","Group2",index_col=0)
data1=data. dropna(how="all")  #一行全部为NaN才删
print(data1)
```

	A	B	C	D	E	F	G	H
1	序号	性别	年龄	身高	体重	省份	成绩	月生活费
2	1	male	20	170	70	LiaoNing		800
3	2	male	22	180	71	GuangXi	77	1300
4	3	male		180	62	FuJian	57	1000
5	4	male	20	177	72	LiaoNing	79	900
6	5	male	20	172		ShanDong	91	
7	6	male	20	179	75	YunNan	92	950
8								
9	7	female	21	166	53	LiaoNing	80	1200
10	8	female	20	162	47	AnHui	78	1000
11	9	female	20	162	47	AnHui	78	1000
12	10	male	120	169	76	HeiLongJiang	88	1100

序号	性别	年龄	身高	体重	省份	成绩	月生活费
1.0	male	20.0	170.0	70.0	LiaoNing	NaN	800.0
2.0	male	22.0	180.0	71.0	GuangXi	77.0	1300.0
3.0	male	NaN	180.0	62.0	FuJian	57.0	1000.0
4.0	male	20.0	177.0	72.0	LiaoNing	79.0	900.0
5.0	male	20.0	172.0	NaN	ShanDong	91.0	NaN
6.0	male	20.0	179.0	75.0	YunNan	92.0	950.0
7.0	female	21.0	166.0	53.0	LiaoNing	80.0	1200.0
8.0	female	20.0	162.0	47.0	AnHui	78.0	1000.0
9.0	female	20.0	162.0	47.0	AnHui	78.0	1000.0
10.0	male	120.0	169.0	76.0	HeiLongJiang	88.0	1100.0

数据清洗

丢弃缺失值dropna(axis,how,thresh,…)

thresh： 只留下有效数据数大于或等于thresh的行或列

```
data. dropna(thresh=6)      # 每行至少6个非空值才保留
```

数据清洗案例

```
import pandas as pd
data=pd.read_excel("info.xlsx","Group2",index_col=0)
data1=data. dropna(thresh=6)  # 每行至少6个非空值才保留
print(data1)
```

	A	B	C	D	E	F	G	H
1	序号	性别	年龄	身高	体重	省份	成绩	月生活费
2	1	male	20	170	70	LiaoNing		800
3	2	male	22	180	71	GuangXi	77	1300
4	3	male		180	62	FuJian	57	1000
5	4	male	20	177	72	LiaoNing	79	900
6	5	male	20	172		ShanDong	91	
7	6	male	20	179	75	YunNan	92	950
8								
9	7	female	21	166	53	LiaoNing	80	1200
10	8	female	20	162	47	AnHui	78	1000
11	9	female	20	162	47	AnHui	78	1000
12	10	male	120	169	76	HeiLongJiang	88	1100

序号	性别	年龄	身高	体重	省份	成绩	月生活费
1.0	male	20.0	170.0	70.0	LiaoNing	NaN	800.0
2.0	male	22.0	180.0	71.0	GuangXi	77.0	1300.0
3.0	male	NaN	180.0	62.0	FuJian	57.0	1000.0
4.0	male	20.0	177.0	72.0	LiaoNing	79.0	900.0
6.0	male	20.0	179.0	75.0	YunNan	92.0	950.0
7.0	female	21.0	166.0	53.0	LiaoNing	80.0	1200.0
8.0	female	20.0	162.0	47.0	AnHui	78.0	1000.0
9.0	female	20.0	162.0	47.0	AnHui	78.0	1000.0
10.0	male	120.0	169.0	76.0	HeiLongJiang	88.0	1100.0

数据清洗

缺失值填充fillna(value, method,…)

value： 填充值，可以是标量、字典等

data. fillna(0)　**#用0填充**

数据清洗案例

```
import pandas as pd
data=pd. read_excel("info.xlsx","Group2",index_col=0)
data1=data. fillna(0)   #用0填充
print(data1)
```

	A	B	C	D	E	F	G	H
1	序号	性别	年龄	身高	体重	省份	成绩	月生活费
2	1	male	20	170	70	LiaoNing		800
3	2	male	22	180	71	GuangXi	77	1300
4	3	male		180	62	FuJian	57	1000
5	4	male	20	177	72	LiaoNing	79	900
6	5	male	20	172		ShanDong	91	
7	6	male	20	179	75	YunNan	92	950
8								
9	7	female	21	166	53	LiaoNing	80	1200
10	8	female	20	162	47	AnHui	78	1000
11	9	female	20	162	47	AnHui	78	1000
12	10	male	120	169	76	HeiLongJiang	88	1100

序号	性别	年龄	身高	体重	省份	成绩	月生活费
1.0	male	20.0	170.0	70.0	LiaoNing	0.0	800.0
2.0	male	22.0	180.0	71.0	GuangXi	77.0	1300.0
3.0	male	0.0	180.0	62.0	FuJian	57.0	1000.0
4.0	male	20.0	177.0	72.0	LiaoNing	79.0	900.0
5.0	male	20.0	172.0	0.0	ShanDong	91.0	0.0
6.0	male	20.0	179.0	75.0	YunNan	92.0	950.0
NaN	0	0.0	0.0	0.0	0	0.0	0.0
7.0	female	21.0	166.0	53.0	LiaoNing	80.0	1200.0
8.0	female	20.0	162.0	47.0	AnHui	78.0	1000.0
9.0	female	20.0	162.0	47.0	AnHui	78.0	1000.0
10.0	male	120.0	169.0	76.0	HeiLongJiang	88.0	1100.0

数据清洗

缺失值填充fillna(value, method,…)

value： 填充值，可以是标量、字典等

data. fillna(**{'年龄': data['年龄'].mean(), '性别': 'male'}**)

数据清洗案例

```
import pandas as pd
data=pd.read_excel("info.xlsx","Group2",index_col=0)
data1=data. fillna({'年龄': data['年龄'].mean(), '性别': 'male'})
print(data1)
```

	A	B	C	D	E	F	G	H
1	序号	性别	年龄	身高	体重	省份	成绩	月生活费
2	1	male	20	170	70	LiaoNing		800
3	2	male	22	180	71	GuangXi	77	1300
4	3	male		180	62	FuJian	57	1000
5	4	male	20	177	72	LiaoNing	79	900
6	5	male	20	172		ShanDong	91	
7	6	male	20	179	75	YunNan	92	950
8								
9	7	female	21	166	53	LiaoNing	80	1200
10	8	female	20	162	47	AnHui	78	1000
11	9	female	20	162	47	AnHui	78	1000
12	10	male	120	169	76	HeiLongJiang	88	1100

序号	性别	年龄	身高	体重	省份	成绩	月生活费
1.0	male	20.000000	170.0	70.0	LiaoNing	NaN	800.0
2.0	male	22.000000	180.0	71.0	GuangXi	77.0	1300.0
3.0	male	31.444444	180.0	62.0	FuJian	57.0	1000.0
4.0	male	20.000000	177.0	72.0	LiaoNing	79.0	900.0
5.0	male	20.000000	172.0	NaN	ShanDong	91.0	NaN
6.0	male	20.000000	179.0	75.0	YunNan	92.0	950.0
NaN	male	31.444444	NaN	NaN	NaN	NaN	NaN
7.0	female	21.000000	166.0	53.0	LiaoNing	80.0	1200.0
8.0	female	20.000000	162.0	47.0	AnHui	78.0	1000.0
9.0	female	20.000000	162.0	47.0	AnHui	78.0	1000.0
10.0	male	120.000000	169.0	76.0	HeiLongJiang	88.0	1100.0

数据清洗

缺失值填充ffill()、bfill()

data. ffill() #在列方向上以上一个值替换

data. bfill() #在列方向上以下一个值替换

数据清洗案例

```
import pandas as pd
data=pd. read_excel("info.xlsx","Group2",index_col=0)
data1=data. ffill() #在列方向上以上一个值替换
print(data1)
```

	A	B	C	D	E	F	G	H
1	序号	性别	年龄	身高	体重	省份	成绩	月生活费
2	1	male	20	170	70	LiaoNing		800
3	2	male	22	180	71	GuangXi	77	1300
4	3	male		180	62	FuJian	57	1000
5	4	male	20	177	72	LiaoNing	79	900
6	5	male	20	172		ShanDong	91	
7	6	male	20	179	75	YunNan	92	950
8								
9	7	female	21	166	53	LiaoNing	80	1200
10	8	female	20	162	47	AnHui	78	1000
11	9	female	20	162	47	AnHui	78	1000
12	10	male	120	169	76	HeiLongJiang	88	1100

序号	性别	年龄	身高	体重	省份	成绩	月生活费
1.0	male	20.0	170.0	70.0	LiaoNing	NaN	800.0
2.0	male	22.0	180.0	71.0	GuangXi	77.0	1300.0
3.0	male	22.0	180.0	62.0	FuJian	57.0	1000.0
4.0	male	20.0	177.0	72.0	LiaoNing	79.0	900.0
5.0	male	20.0	172.0	72.0	ShanDong	91.0	900.0
6.0	male	20.0	179.0	75.0	YunNan	92.0	950.0
NaN	male	20.0	179.0	75.0	YunNan	92.0	950.0
7.0	female	21.0	166.0	53.0	LiaoNing	80.0	1200.0
8.0	female	20.0	162.0	47.0	AnHui	78.0	1000.0
9.0	female	20.0	162.0	47.0	AnHui	78.0	1000.0
10.0	male	120.0	169.0	76.0	HeiLongJiang	88.0	1100.0

数据清洗

值替换replace(to_replace, value, …)

to_replace： 将被替代的值

value: 替换为的值

```
data['年龄']=data['年龄'].replace(120,20) #将年龄120替换为20
```

数据清洗案例

```
import pandas as pd
data=pd. read_excel("info.xlsx","Group2",index_col=0)
data['年龄']=data['年龄'].replace(120,20)#将年龄120替换为20
print(data)
```

	A	B	C	D	E	F	G	H
1	序号	性别	年龄	身高	体重	省份	成绩	月生活费
2	1	male	20	170	70	LiaoNing		800
3	2	male	22	180	71	GuangXi	77	1300
4	3	male		180	62	FuJian	57	1000
5	4	male	20	177	72	LiaoNing	79	900
6	5	male	20	172		ShanDong	91	
7	6	male	20	179	75	YunNan	92	950
8								
9	7	female	21	166	53	LiaoNing	80	1200
10	8	female	20	162	47	AnHui	78	1000
11	9	female	20	162	47	AnHui	78	1000
12	10	male	120	169	76	HeiLongJiang	88	1100

序号	性别	年龄	身高	体重	省份	成绩	月生活费
1.0	male	20.0	170.0	70.0	LiaoNing	NaN	800.0
2.0	male	22.0	180.0	71.0	GuangXi	77.0	1300.0
3.0	male	NaN	180.0	62.0	FuJian	57.0	1000.0
4.0	male	20.0	177.0	72.0	LiaoNing	79.0	900.0
5.0	male	20.0	172.0	NaN	ShanDong	91.0	NaN
6.0	male	20.0	179.0	75.0	YunNan	92.0	950.0
NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
7.0	female	21.0	166.0	53.0	LiaoNing	80.0	1200.0
8.0	female	20.0	162.0	47.0	AnHui	78.0	1000.0
9.0	female	20.0	162.0	47.0	AnHui	78.0	1000.0
10.0	male	20.0	169.0	76.0	HeiLongJiang	88.0	1100.0

数据清洗

去掉重复值drop_duplicates()

data. drop_duplicates() #去掉重复的数据

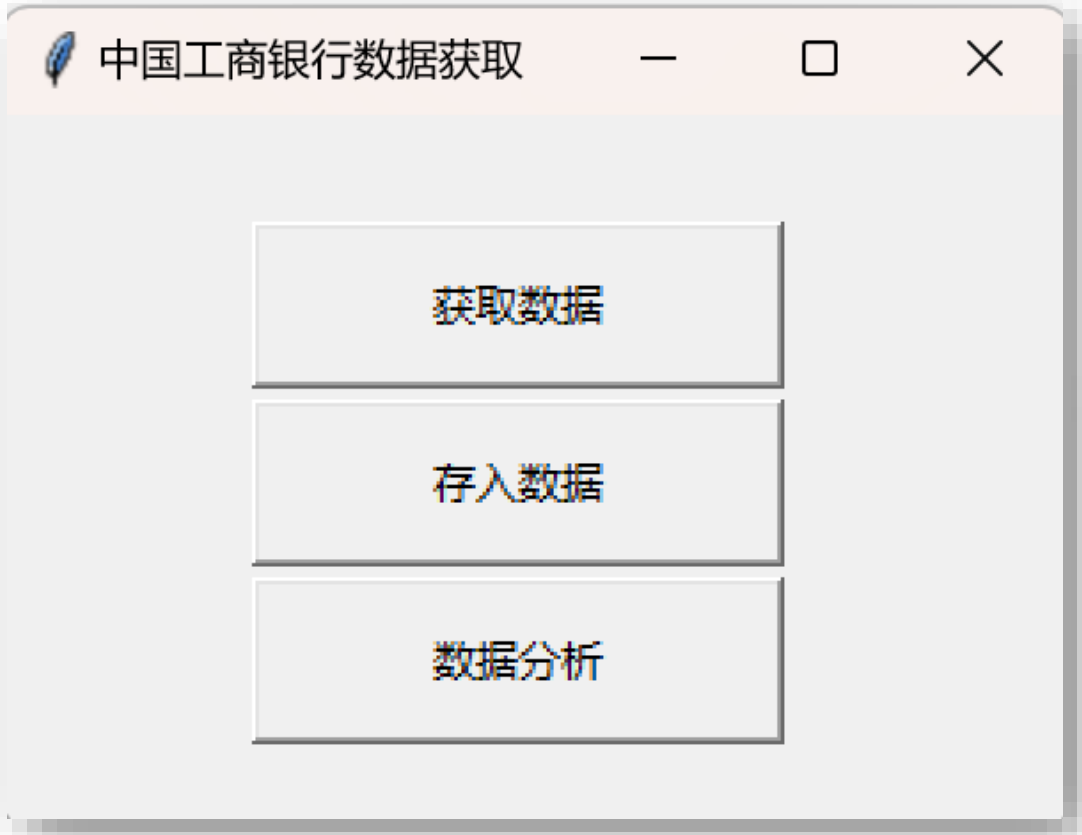
数据清洗案例

```
import pandas as pd
data=pd. read_excel("info.xlsx","Group2",index_col=0)
data1=data. drop_duplicates()  #去掉重复的数据
print(data1)
```

	A	B	C	D	E	F	G	H
1	序号	性别	年龄	身高	体重	省份	成绩	月生活费
2	1	male	20	170	70	LiaoNing		800
3	2	male	22	180	71	GuangXi	77	1300
4	3	male		180	62	FuJian	57	1000
5	4	male	20	177	72	LiaoNing	79	900
6	5	male	20	172		ShanDong	91	
7	6	male	20	179	75	YunNan	92	950
8								
9	7	female	21	166	53	LiaoNing	80	1200
10	8	female	20	162	47	AnHui	78	1000
11	9	female	20	162	47	AnHui	78	1000
12	10	male	120	169	76	HeiLongJiang	88	1100

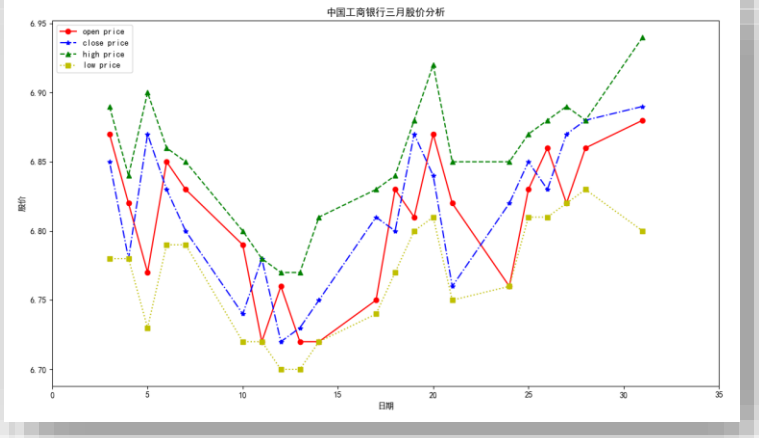
序号	性别	年龄	身高	体重	省份	成绩	月生活费
1.0	male	20.0	170.0	70.0	LiaoNing	NaN	800.0
2.0	male	22.0	180.0	71.0	GuangXi	77.0	1300.0
3.0	male	NaN	180.0	62.0	FuJian	57.0	1000.0
4.0	male	20.0	177.0	72.0	LiaoNing	79.0	900.0
5.0	male	20.0	172.0	NaN	ShanDong	91.0	NaN
6.0	male	20.0	179.0	75.0	YunNan	92.0	950.0
NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
7.0	female	21.0	166.0	53.0	LiaoNing	80.0	1200.0
8.0	female	20.0	162.0	47.0	AnHui	78.0	1000.0
10.0	male	120.0	169.0	76.0	HeiLongJiang	88.0	1100.0

案例分析



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ts_code	trade_date	open	high	low	close	pre_close	change	pct_chg	vol	amount
601398.SH	20250331	6.88	6.94	6.8	6.89	6.88	0.01	0.1453	4071337.79	2804461.039
601398.SH	20250328	6.86	6.88	6.83	6.88	6.87	0.01	0.1456	2128381.48	1458086.737
601398.SH	20250327	6.82	6.89	6.82	6.87	6.83	0.04	0.5857	2580278.1	1771784.348
601398.SH	20250326	6.86	6.88	6.81	6.83	6.85	-0.02	-0.292	2337800.88	1598912.19
601398.SH	20250325	6.83	6.87	6.81	6.85	6.82	0.03	0.4399	2318056.72	1587419.428
601398.SH	20250324	6.76	6.85	6.76	6.82	6.76	0.06	0.8876	3060401.69	2084392.843
601398.SH	20250321	6.82	6.85	6.75	6.76	6.84	-0.08	-1.1696	3087991.23	2096634.446
601398.SH	20250320	6.87	6.92	6.81	6.84	6.87	-0.03	-0.4367	3008452.41	2058985.286
601398.SH	20250319	6.81	6.88	6.8	6.87	6.8	0.07	1.0294	3032694.2	2077764.026
601398.SH	20250318	6.83	6.84	6.77	6.8	6.81	-0.01	-0.1468	2369528.1	1611444.863
601398.SH	20250317	6.75	6.83	6.74	6.81	6.75	0.06	0.8889	3586198.28	2436756.611
601398.SH	20250314	6.72	6.81	6.72	6.75	6.73	0.02	0.2972	4890721.29	3311312.295
601398.SH	20250313	6.72	6.77	6.7	6.73	6.72	0.01	0.1488	2572950.17	1734795.847
601398.SH	20250312	6.76	6.77	6.7	6.72	6.78	-0.06	-0.885	3177702.18	2135384.159
601398.SH	20250311	6.72	6.78	6.72	6.78	6.74	0.04	0.5935	2680002.41	1809114.543
601398.SH	20250310	6.79	6.8	6.72	6.74	6.8	-0.06	-0.8824	3242116.47	2186689.995
601398.SH	20250307	6.83	6.85	6.79	6.8	6.83	-0.03	-0.4392	3223755.78	2195284.348

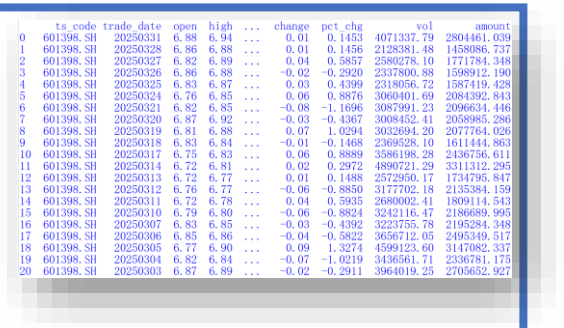


案例分析

```
import pandas as pd
import tushare as ts
from tkinter import *
from sqlalchemy import create_engine
import matplotlib.pyplot as plt
df = pd.DataFrame()
```

案例分析

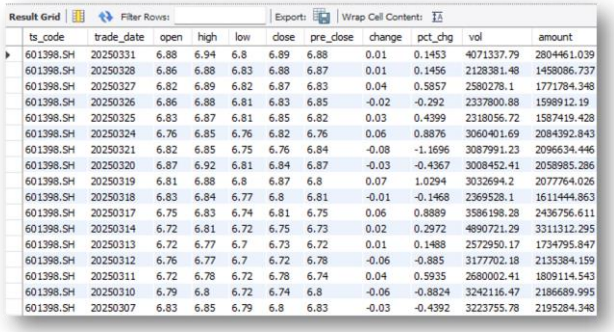
```
def gs_stock():
    global df
    ts.set_token('XXX')
    #换成自己的token
    pro = ts. pro_api()        #初始化
    #获取股票代码为'601398.SH'（中国工商银行）的历史行情
    df=pro.daily(ts_code='601398.SH', start_date='20250301',
                                        end_date='20250331')
    print(df)
```



	ts_code	trade_date	open	high	...	change	pct_chg	vol	amount
0	601398.SH	20250331	6.88	6.94	...	0.01	0.1453	4071337.79	2804461.039
1	601398.SH	20250328	6.86	6.88	...	0.01	0.1456	2128381.48	1458086.737
2	601398.SH	20250327	6.82	6.89	...	0.04	0.5857	2580278.10	1771784.348
3	601398.SH	20250326	6.86	6.88	...	-0.02	-0.2920	2337800.88	1598912.190
4	601398.SH	20250325	6.83	6.87	...	0.03	0.4399	2318056.72	1587419.428
5	601398.SH	20250324	6.76	6.85	...	0.06	0.8876	3060401.69	2084392.843
6	601398.SH	20250321	6.82	6.85	...	-0.08	-1.1696	3087991.23	2096634.446
7	601398.SH	20250320	6.87	6.92	...	-0.03	-0.4367	3008452.41	2058985.286
8	601398.SH	20250319	6.81	6.88	...	0.07	1.0294	3032694.20	2077764.026
9	601398.SH	20250318	6.83	6.84	...	-0.01	-0.1468	2369528.10	1611444.863
10	601398.SH	20250317	6.75	6.83	...	0.06	0.8889	3586198.28	2436756.611
11	601398.SH	20250314	6.72	6.81	...	0.02	0.2972	4890721.29	3311312.295
12	601398.SH	20250313	6.72	6.77	...	0.01	0.1488	2572950.17	1734795.847
13	601398.SH	20250312	6.76	6.77	...	-0.06	-0.8850	3177702.18	2135384.159
14	601398.SH	20250311	6.72	6.78	...	0.04	0.5935	2680002.41	1809114.543
15	601398.SH	20250310	6.79	6.80	...	-0.06	-0.8824	3242116.47	2186689.995
16	601398.SH	20250307	6.83	6.85	...	-0.03	-0.4392	3223755.78	2195284.348
17	601398.SH	20250306	6.85	6.86	...	-0.04	-0.5822	3656712.05	2495349.517
18	601398.SH	20250305	6.77	6.90	...	0.09	1.3274	4599123.60	3147082.337
19	601398.SH	20250304	6.82	6.84	...	-0.07	-1.0219	3436561.71	2336781.175
20	601398.SH	20250303	6.87	6.89	...	-0.02	-0.2911	3964019.25	2705652.927

案例分析

```
def gs_save():
    global df
    # 创建 SQLAlchemy 引擎
    engine = create_engine('mysql+pymysql://root:123456@127.0.0.1:3306/test')
    # 写入 MySQL（如果表不存在，会自动创建）
    df.to_sql(name="stock",          # 表名
        con=engine,                  # 数据库连接
        index=False,                 # 不写入 DataFrame 的索引
        if_exists="replace" )        # 如果表存在，则替换
    print("DataFrame 已成功写入 MySQL！")
```



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

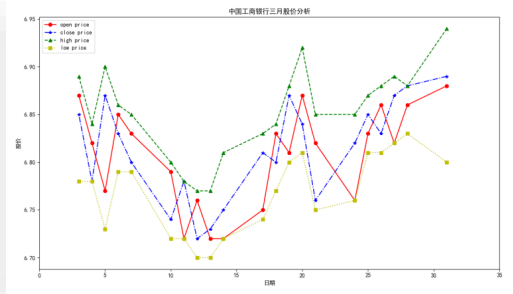
ts_code	trade_date	open	high	low	close	pre_close	change	pct_chg	vol	amount
601398.SH	20250331	6.88	6.94	6.8	6.89	6.88	0.01	0.1453	4071337.79	2804461.039
601398.SH	20250328	6.86	6.88	6.83	6.88	6.87	0.01	0.1456	2128381.48	1458086.737
601398.SH	20250327	6.82	6.89	6.82	6.87	6.83	0.04	0.5857	2580278.1	1771784.348
601398.SH	20250326	6.86	6.88	6.81	6.83	6.85	-0.02	-0.292	2337800.88	1598912.19
601398.SH	20250325	6.83	6.87	6.81	6.85	6.82	0.03	0.4399	2318056.72	1587419.428
601398.SH	20250324	6.76	6.85	6.76	6.82	6.76	0.06	0.8876	3060401.69	2084392.843
601398.SH	20250321	6.82	6.85	6.75	6.76	6.84	-0.08	-1.1696	3087991.23	2096634.446
601398.SH	20250320	6.87	6.92	6.81	6.84	6.87	-0.03	-0.4367	3008452.41	2058985.286
601398.SH	20250319	6.81	6.88	6.8	6.87	6.8	0.07	1.0294	3032694.2	2077764.026
601398.SH	20250318	6.83	6.84	6.77	6.8	6.81	-0.01	-0.1468	2369528.1	1611444.863
601398.SH	20250317	6.75	6.83	6.74	6.81	6.75	0.06	0.8889	3586198.28	2436756.611
601398.SH	20250314	6.72	6.81	6.72	6.75	6.73	0.02	0.2972	4890721.29	3311312.295
601398.SH	20250313	6.72	6.77	6.7	6.73	6.72	0.01	0.1488	2572950.17	1734795.847
601398.SH	20250312	6.76	6.77	6.7	6.72	6.78	-0.06	-0.885	3177702.18	2135384.159
601398.SH	20250311	6.72	6.78	6.72	6.78	6.74	0.04	0.5935	2680002.41	1809114.543
601398.SH	20250310	6.79	6.8	6.72	6.74	6.8	-0.06	-0.8824	3242116.47	2186689.995
601398.SH	20250307	6.83	6.85	6.79	6.8	6.83	-0.03	-0.4392	3223755.78	2195284.348

案例分析

```
def gs_analyse():
    # 创建 SQLAlchemy 引擎
    engine = create_engine('mysql+pymysql://root:123456@127.0.0.1:3306/test')
    # 使用pandas的read_sql_table函数读取整个表的数据
    data = pd.read_sql_table('stock', engine)
    date=pd.to_datetime(data["trade_date"]).dt.day
```

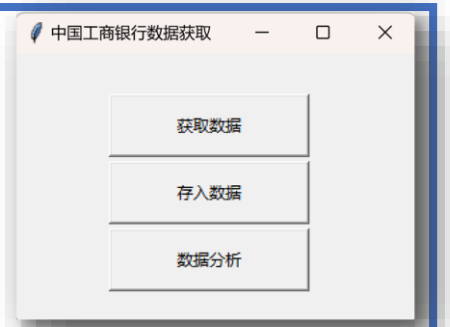
案例分析

```
plt.xlim(0,35)
plt.title("中国工商银行三月股价分析")
plt.xlabel("日期")
plt.ylabel("股价")
plt.rcParams['font.sans-serif']=['SimHei']
plt.plot(date,data["open"],"r-o",label="open price")
plt.plot(date,data["close"],"b-.*",label="close price")
plt.plot(date,data["high"],"g--^",label="high price")
plt.plot(date,data["low"],"y:s",label="low price")
plt.legend()
plt.show()
```



案例分析

```
root=Tk()
root.title("中国工商银行数据获取")
root.geometry("300x200")
one=Button(root,text='获取数据',width=20,
                        height=2,command=gs_stock)
one.place(x=70,y=30)
two=Button(root,text='存入数据',width=20,
                         height=2,command=gs_save)
two.place(x=70,y=100)
third=Button(root,text='数据分析',width=20,
                        height=2,command= gs_analyse)
third.place(x=70,y=130)
root.mainloop()
```



谢　谢