Python与金融数据挖掘(13)

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…

…

Tushare简介

Tushare： 是一个**免费**、**开源**的python财经数据接口包。主要实现对股票等金融数据从**数据采集、清洗加工到数据存储**的过程，能够为金融分析人员提供快速、整洁和多样的便于分析的数据，使他们更加专注于**策略和模型**的研究与实现上。

官网： https://www.tushare.pro/

获取数据方法

使用方法：获取Tushare Pro的数据API，首先需注册一个Pro账号(100积分)，然后登录Pro网站在个人主页里**拿到token码**，修改个人信息(20积分)。对于股票行情数据，只要有120积分就可以相对高频的获取数据。

注册：https://tushare.pro/document/1?doc_id=38

数据接口

```
import tushare as ts
ts.set_token('XXX')   #换成自己的token
pro = ts. pro_api()      #初始化
#获取股票代码为'600000'（浦发银行）的历史行情
df=pro. daily(ts_code='600000.SH', start_date='20220101',
                                                       end_date='20220228')
print(df)
```

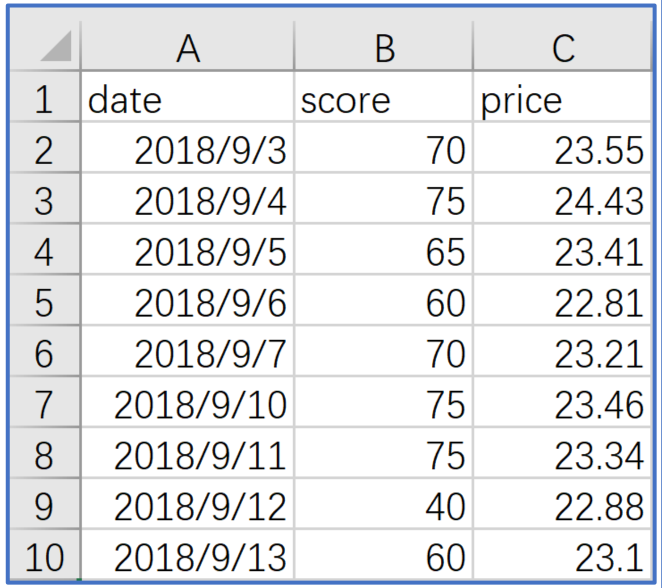
数据接口：https://tushare.pro/document/2

DataFrame数据选取方法

选取类型	选取方法	说明
基于位置序号选取	Obj.iloc[iloc, cloc]	选取某行某列
	Obj.iloc[ilocList,clocList]	选取多行多列
	Obj.iloc[a:b,c:d]	选取a~b-1行，
		c~d-1列

存取部分数据

```
import pandas as pd
data=pd.read_csv('data.csv')
print(data. iloc[7,1])
print(data. iloc[[0,2],[1,2]])
result=data.iloc[0:3,1:3]
result.to_csv("result.csv")
```



	A	B	C
1	date	score	price
2	2018/9/3	70	23.55
3	2018/9/4	75	24.43
4	2018/9/5	65	23.41
5	2018/9/6	60	22.81
6	2018/9/7	70	23.21
7	2018/9/10	75	23.46
8	2018/9/11	75	23.34
9	2018/9/12	40	22.88
10	2018/9/13	60	23.1

```
40
   score  price
0     70  23.55
2     65  23.41
```

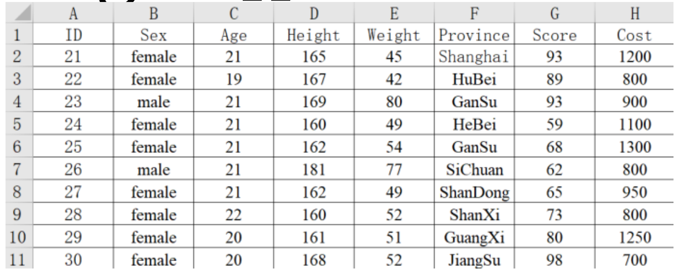
DataFrame数据选取方法

选取类型	选取方法	说明
基于索引名选取	Obj[col]	选取某列
	Obj[colList]	选取某几列
	Obj.loc[index,col]	选取某行某列
	Obj.loc[indexList,colList]	选取多行多列

存取部分数据

```
>>> 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.loc[[21,23],["Height","Weight"]]
print(result)
result.to_excel("result.xlsx")
```



	A	B	C	D	E	F	G	H
1	ID	Sex	Age	Height	Weight	Province	Score	Cost
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

DataFrame数据选取方法

选取类型	选取方法	说明
条件筛选	**Obj.loc[condition,colList]**	使用索引构造条件表达式
	Obj.iloc[condition,clocList]	使用位置序号构造条件表达式

根据年龄统计信息

```
import matplotlib.pyplot as plt
import pandas as pd
data=pd.read_excel("info.xlsx","Group1",index_col=0)
#筛选出年龄大于20岁的学生Sex, Age
result= data. loc[data['Age'] >20,["Sex","Age"]]
print(result)
```

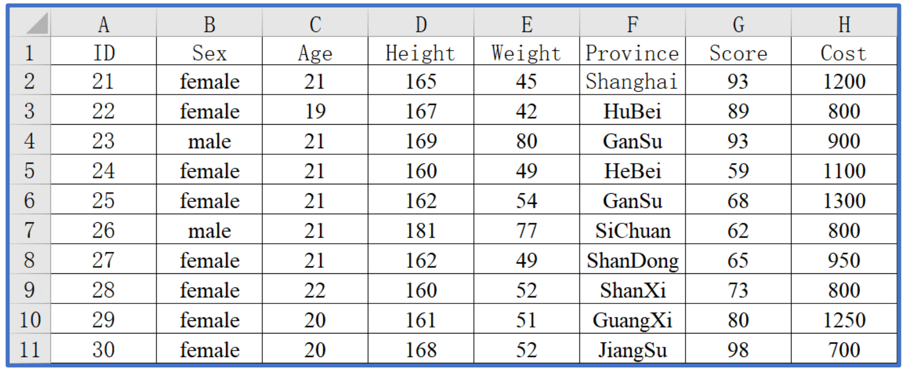
```
       Sex  Age
ID
21  female   21
23    male   21
24  female   21
25  female   21
26    male   21
27  female   21
28  female   22
```

Pandas常用统计函数

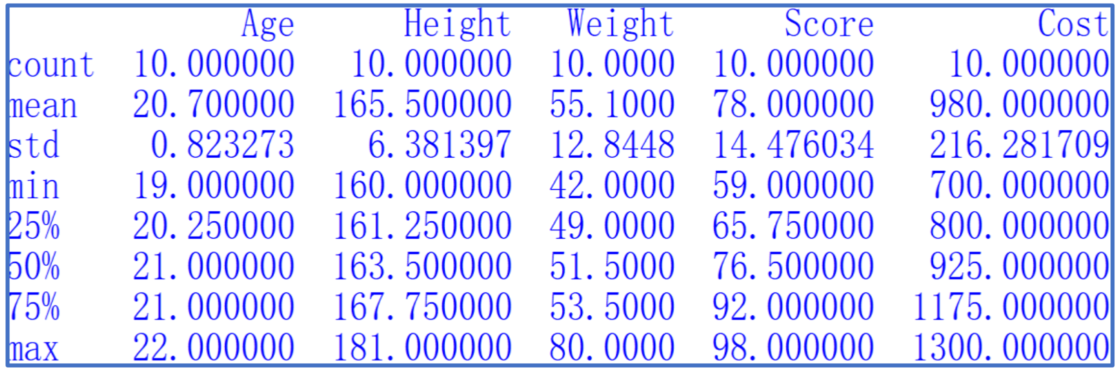
函数	描述
df.mean()	计算样本数据的算术平均值
df.value_counts()	统计频数
df.describe()	返回基本统计量和分位数
df.corr(sr)	df与sr的相关系数
df.count()、df.sum()	统计每列(或行)数据的个数或总和
df.max()、df.min()	最大值和最小值
df.idxmax()、df.idxmin()	最大值、最小值对应的索引
df.qantile()	计算给定的四分位数
df.var()、df.std()	计算方差、标准差
df.mode()	计算众数
df.cov()	计算协方差矩阵

Pandas常用统计案例

```
import pandas as pd
data=pd.read_excel("info.xlsx","Group1",index_col=0)
result=data.describe()  #对数据进行统计描述
print(result)
```



	A	B	C	D	E	F	G	H
1	ID	Sex	Age	Height	Weight	Province	Score	Cost
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



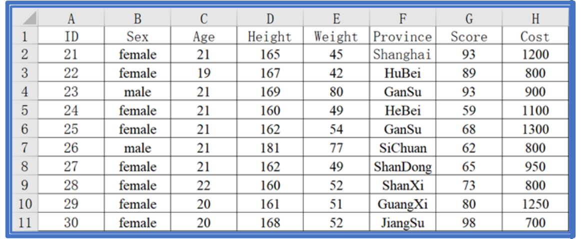
```
             Age       Height   Weight      Score         Cost
count  10.000000    10.000000  10.0000  10.000000    10.000000
mean   20.700000   165.500000  55.1000  78.000000   980.000000
std     0.823273     6.381397  12.8448  14.476034   216.281709
min    19.000000   160.000000  42.0000  59.000000   700.000000
25%    20.250000   161.250000  49.0000  65.750000   800.000000
50%    21.000000   163.500000  51.5000  76.500000   925.000000
75%    21.000000   167.750000  53.5000  92.000000  1175.000000
max    22.000000   181.000000  80.0000  98.000000  1300.000000
```

Pandas常用统计函数

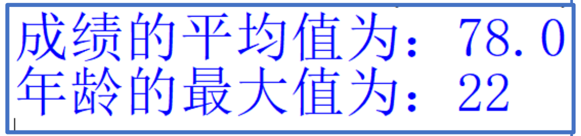
函数	描述
df.mean()	计算样本数据的算术平均值
df.value_counts()	统计频数
df.describe()	返回基本统计量和分位数
df.corr(sr)	df与sr的相关系数
df.count()、df.sum()	统计每列(或行)数据的个数或总和
df.max()、df.min()	最大值和最小值
df.idxmax()、df.idxmin()	最大值、最小值对应的索引
df.qantile()	计算给定的四分位数
df.var()、df.std()	计算方差、标准差
df.mode()	计算众数
df.cov()	计算协方差矩阵

Pandas常用统计案例

```
import pandas as pd
data=pd.read_excel("info.xlsx","Group1",index_col=0)
avg=data['Score'].mean()
print("成绩的平均值为：{}".format(avg))
max_age=data['Age'].max()
print("年龄的最大值为：{}".format(max_age))
```



	A	B	C	D	E	F	G	H
1	ID	Sex	Age	Height	Weight	Province	Score	Cost
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



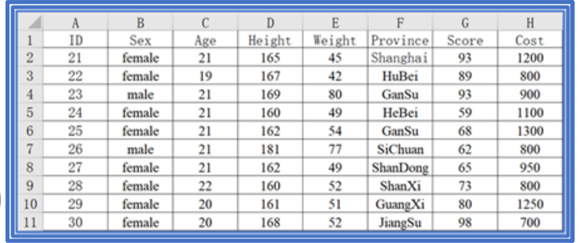
```
成绩的平均值为：78.0
年龄的最大值为：22
```

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

Pandas常用统计案例

```
import pandas as pd
data=pd.read_excel("info.xlsx","Group1",index_col=0)
score=data["Score"].sum()
print("学生的总成绩为：{}".format(score))
age=data["Age"].mode()
print("学生多数年龄为：{}".format(age))
```



	A	B	C	D	E	F	G	H
1	ID	Sex	Age	Height	Weight	Province	Score	Cost
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

```
学生的总成绩为：780
学生多数年龄为：0    21
Name: Age, dtype: int64
```

相关性分析

相关性分析：研究现象之间是否存在依赖关系，定量分析可以通过计算样本之间的相关系数r来实现，r具有以下特征：

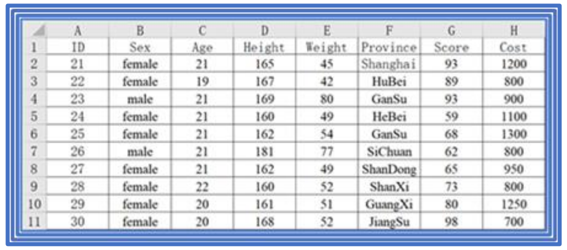
1. r的值介于-1和+1之间，r=1表示正相关，r=0表示不相关，r=-1表示负相关
2. 当0<|r|<1，表示两个对象存在一定程度的相关性，|r|越接近1，关系越密切，越接近0，相关性越弱
3. |r|<0.4为低相关；0.4=<|r|<0.7为中等相关，|r|>=0.7为高相关

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

Pandas常用统计案例

```
import pandas as pd
data=pd.read_excel("info.xlsx","Group1",index_col=0)
result=data['Height']. corr( data['Weight'] )
print("身高和体重的相关性为：{}".format(result))
```



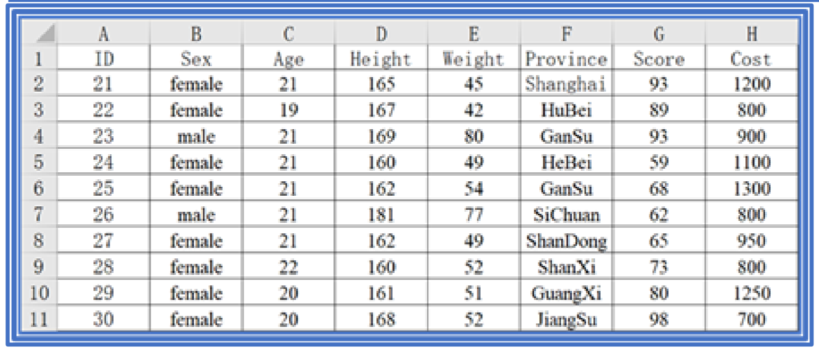
	A	B	C	D	E	F	G	H
1	ID	Sex	Age	Height	Weight	Province	Score	Cost
2	21	female	21	165	45	Shanghai	93	1200
3	22	female	19	167	42	HuBei	89	800
4	23	male	21	169	80	GanSu	93	900
5	24	female	21	160	49	HeBei	59	1100
6	25	female	21	162	54	GanSu	68	1300
7	26	male	21	181	77	SiChuan	62	800
8	27	female	21	162	49	ShanDong	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

```
身高和体重的相关性为：0.6757399098527682
```

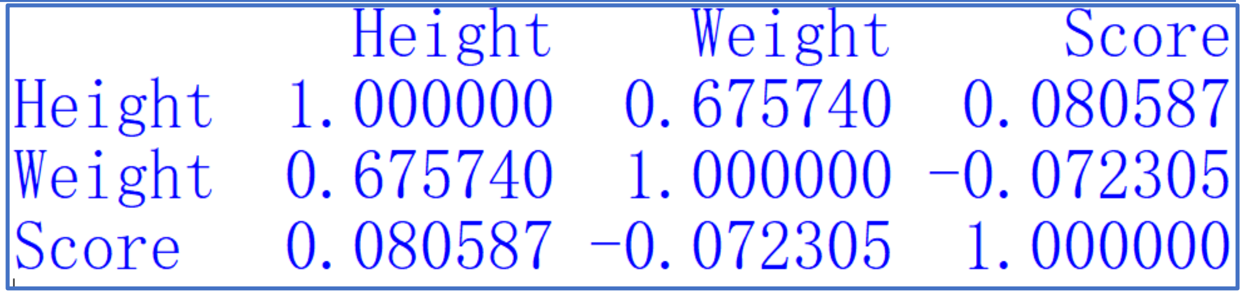
Pandas常用统计案例

```
import pandas as pd
data=pd.read_excel("info.xlsx","Group1",index_col=0)
result=data[['Height','Weight','Score'] ].corr()
print(result)
```

如果想画图显示三者关系如何处理？

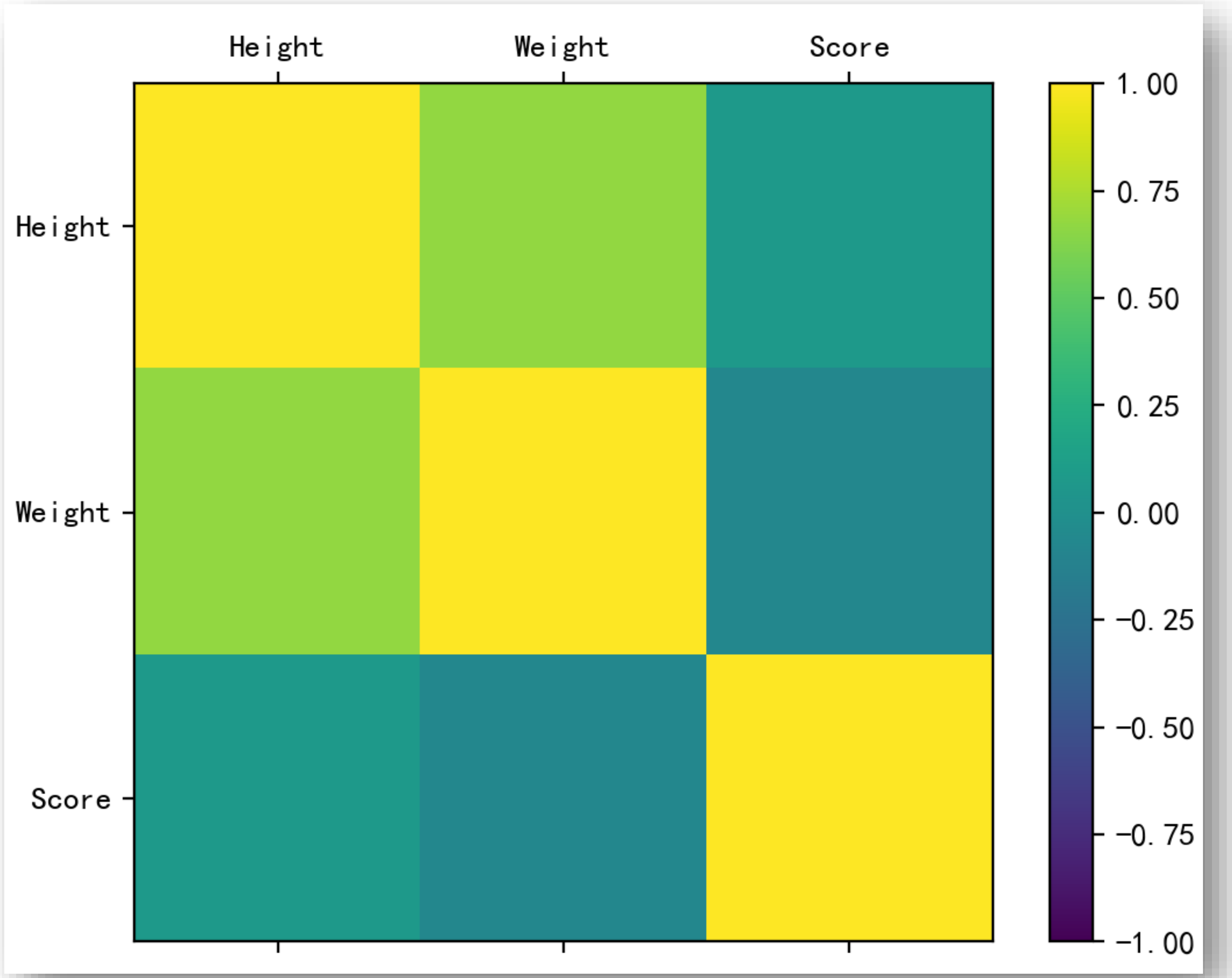


	A	B	C	D	E	F	G	H
1	ID	Sex	Age	Height	Weight	Province	Score	Cost
2	21	female	21	165	45	Shanghai	93	1200
3	22	female	19	167	42	HuBei	89	800
4	23	male	21	169	80	GanSu	93	900
5	24	female	21	160	49	HeBei	59	1100
6	25	female	21	162	54	GanSu	68	1300
7	26	male	21	181	77	SiChuan	62	800
8	27	female	21	162	49	ShanDong	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



```
          Height    Weight     Score
Height  1.000000  0.675740  0.080587
Weight  0.675740  1.000000 -0.072305
Score   0.080587 -0.072305  1.000000
```

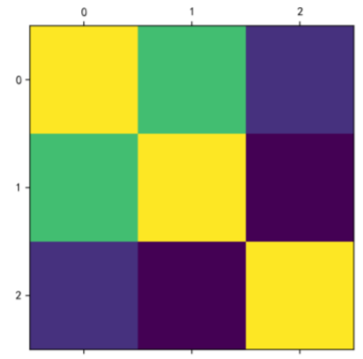
Matshow图



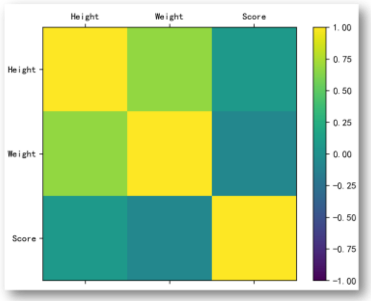
Height
Weight
Score
Height
Weight
Score
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

Pandas统计分析案例

```
import matplotlib.pyplot as plt  #导入matplotlib.pyplot
import pandas as pd
plt.rcParams['font.family']=['SimHei']
data=pd.read_excel("info.xlsx","Group1",index_col=0)
result=data[['Height','Weight','Score'] ].corr()
plt.matshow(result)  #相关矩阵图展示两个不同属性相互影响的程度
plt.show()
```



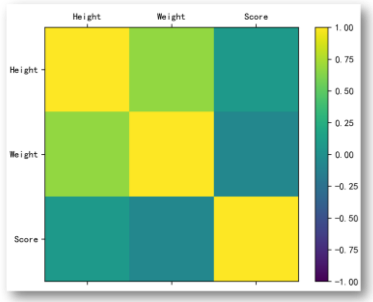
Pandas统计分析案例（拓展）



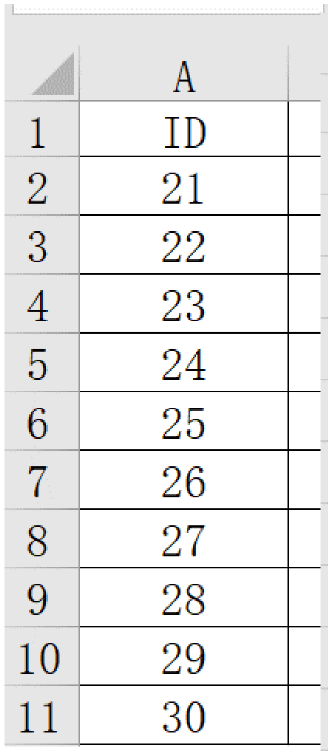
```
import matplotlib.pyplot as plt   #导入matplotlib.pyplot
import pandas as pd
import numpy as np
#plt.rcParams['font.family']=['SimHei']        #显示中文
plt.rcParams['axes.unicode_minus'] = False    #显示负号
data=pd.read_excel("info.xlsx","Group1",index_col=0)
result=data[['Height','Weight','Score'] ].corr()
```

Pandas统计分析案例（拓展）

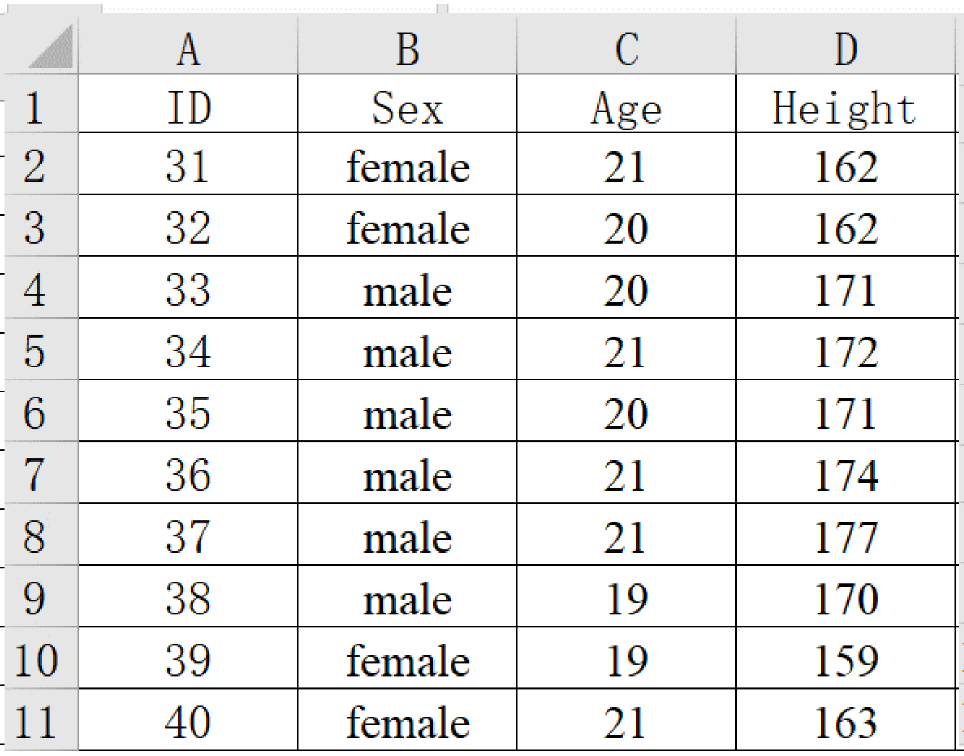
```
fig=plt.figure()
ax=fig.add_subplot(111)
cax=ax.matshow(result, vmin=-1, vmax=1)    #相关矩阵图
fig.colorbar(cax)
ticks=np.arange(0,3,1)
names=['Height','Weight','Score']
ax.set_xticks(ticks); ax.set_yticks(ticks)
ax.set_xticklabels(names); ax.set_yticklabels(names)
plt.show()
```



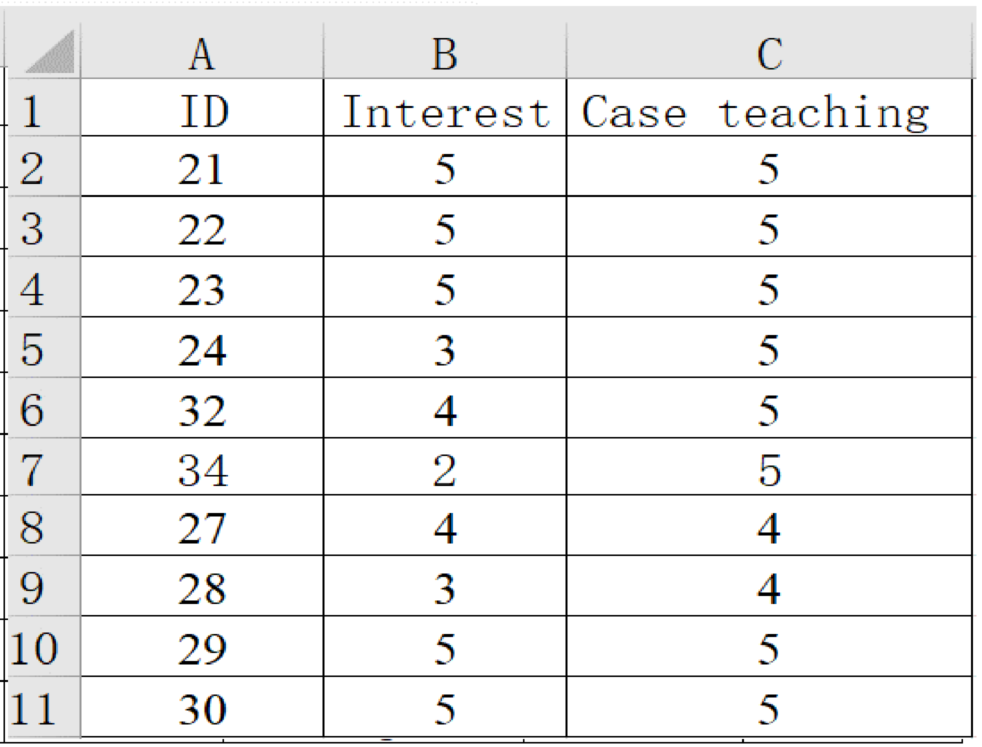
数据规整化



	A
1	ID
2	21
3	22
4	23
5	24
6	25
7	26
8	27
9	28
10	29
11	30



	A	B	C	D
1	ID	Sex	Age	Height
2	31	female	21	162
3	32	female	20	162
4	33	male	20	171
5	34	male	21	172
6	35	male	20	171
7	36	male	21	174
8	37	male	21	177
9	38	male	19	170
10	39	female	19	159
11	40	female	21	163

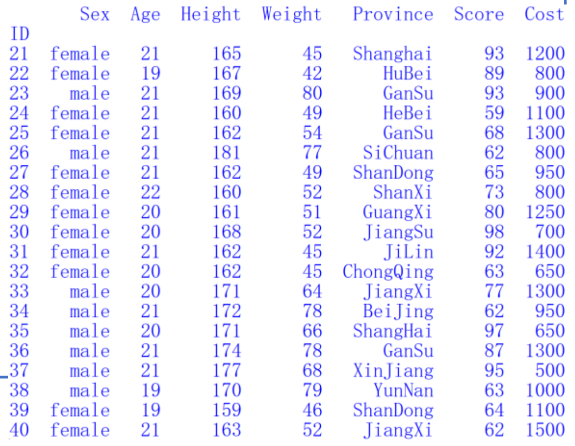


	A	B	C
1	ID	Interest	Case teaching
2	21	5	5
3	22	5	5
4	23	5	5
5	24	3	5
6	32	4	5
7	34	2	5
8	27	4	4
9	28	3	4
10	29	5	5
11	30	5	5

数据规整化

➢**concat()：** 行数据连接函数

```
import pandas as pd
data1=pd.read_excel("info.xlsx","Group1",index_col=0)
data2=pd.read_excel("info.xlsx","Group3",index_col=0)
#axis=0表示按行追加
data = pd.concat([data1,data2], axis=0)
print(data)
```



```
       Sex  Age  Height  Weight   Province  Score  Cost
ID
21  female   21     165      45   Shanghai     93  1200
22  female   19     167      42      HuBei     89   800
23    male   21     169      80      GanSu     93   900
24  female   21     160      49      HeBei     59  1100
25  female   21     162      54      GanSu     68  1300
26    male   21     181      77    SiChuan     62   800
27  female   21     162      49   ShanDong     65   950
28  female   22     160      52     ShanXi     73   800
29  female   20     161      51    GuangXi     80  1250
30  female   20     168      52    JiangSu     98   700
31  female   21     162      45      JiLin     92  1400
32  female   20     162      45  ChongQing     63   650
33    male   20     171      64    JiangXi     77  1300
34    male   21     172      78    BeiJing     62   950
35    male   20     171      66   ShangHai     97   650
36    male   21     174      78      GanSu     87  1300
37    male   21     177      68   XinJiang     95   500
38    male   19     170      79     YunNan     63  1000
39  female   19     159      46   ShanDong     64  1100
40  female   21     163      52    JiangXi     62  1500
```

数据规整化

- **merge(x,y,how,left_on,right_on…)**

  x： 左数据对象

  y： 右数据对象

  how： 数据对象连接方式：inner, outer, left, right

  - **inner：** 内连接，连接两个数据对象中键值交集的行
  - **left:** 左连接，取出x的全部行，连接y中匹配的键值行

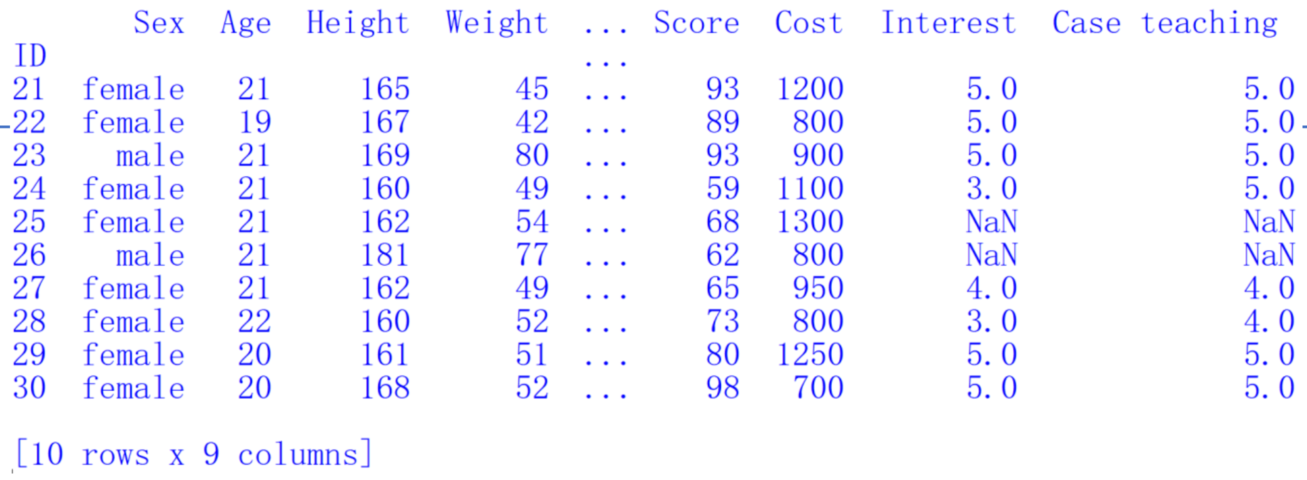
  left_on： 左数据用于连接的键

  right_on： 右数据用于连接的键

数据规整化

➢**merge()：** 列数据连接函数

```
import pandas as pd
data1=pd.read_excel("info.xlsx","Group1",index_col=0)
data2=pd.read_excel("info.xlsx","Group4",index_col=0)
result= pd.merge(data1,data2, how='left',left_on="ID",right_on="ID")
print(result) #左连接
```



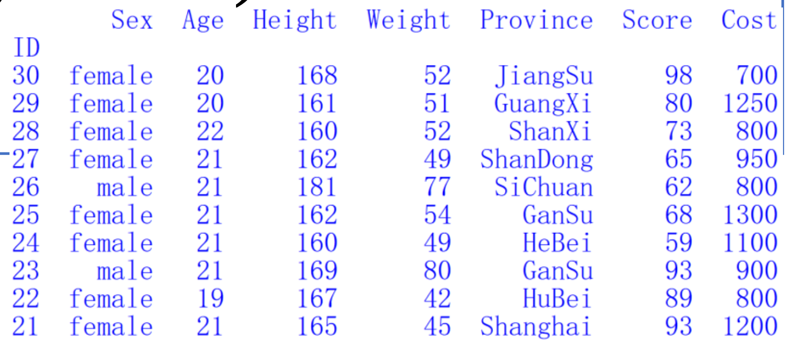
```
       Sex  Age  Height  Weight  ...  Score  Cost  Interest  Case teaching
ID                               ...
21  female   21     165      45  ...     93  1200       5.0            5.0
22  female   19     167      42  ...     89   800       5.0            5.0
23    male   21     169      80  ...     93   900       5.0            5.0
24  female   21     160      49  ...     59  1100       3.0            5.0
25  female   21     162      54  ...     68  1300       NaN            NaN
26    male   21     181      77  ...     62   800       NaN            NaN
27  female   21     162      49  ...     65   950       4.0            4.0
28  female   22     160      52  ...     73   800       3.0            4.0
29  female   20     161      51  ...     80  1250       5.0            5.0
30  female   20     168      52  ...     98   700       5.0            5.0

[10 rows x 9 columns]
```

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)
```

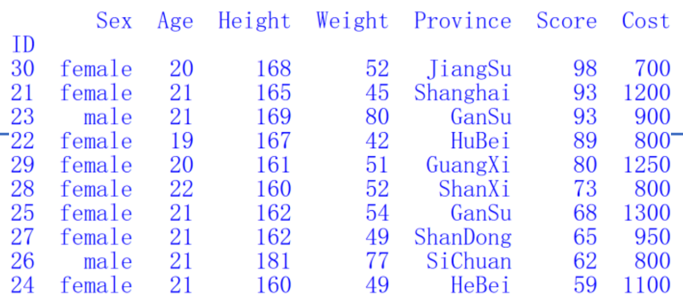


```
       Sex  Age  Height  Weight  Province  Score  Cost
ID
30  female   20     168      52   JiangSu     98   700
29  female   20     161      51   GuangXi     80  1250
28  female   22     160      52    ShanXi     73   800
27  female   21     162      49  ShanDong     65   950
26    male   21     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
```

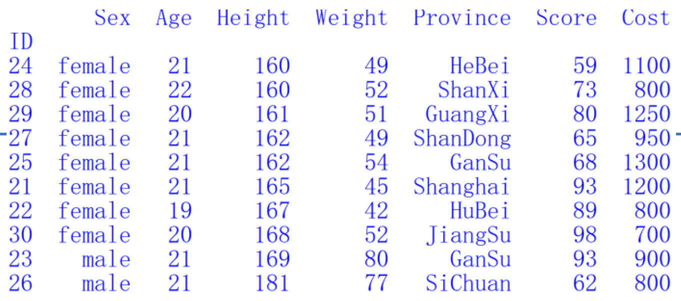
数据排序

➢按值排序

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



```
       Sex  Age  Height  Weight  Province  Score  Cost
ID
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
```

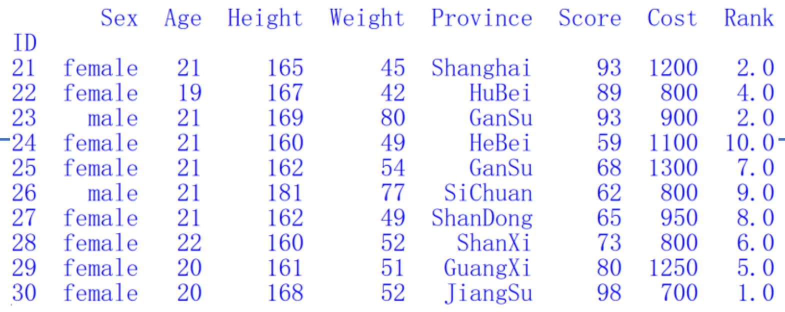


```
       Sex  Age  Height  Weight  Province  Score  Cost
ID
24  female   21     160      49     HeBei     59  1100
28  female   22     160      52    ShanXi     73   800
29  female   20     161      51   GuangXi     80  1250
27  female   21     162      49  ShanDong     65   950
25  female   21     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
```

DataFrame数据排序

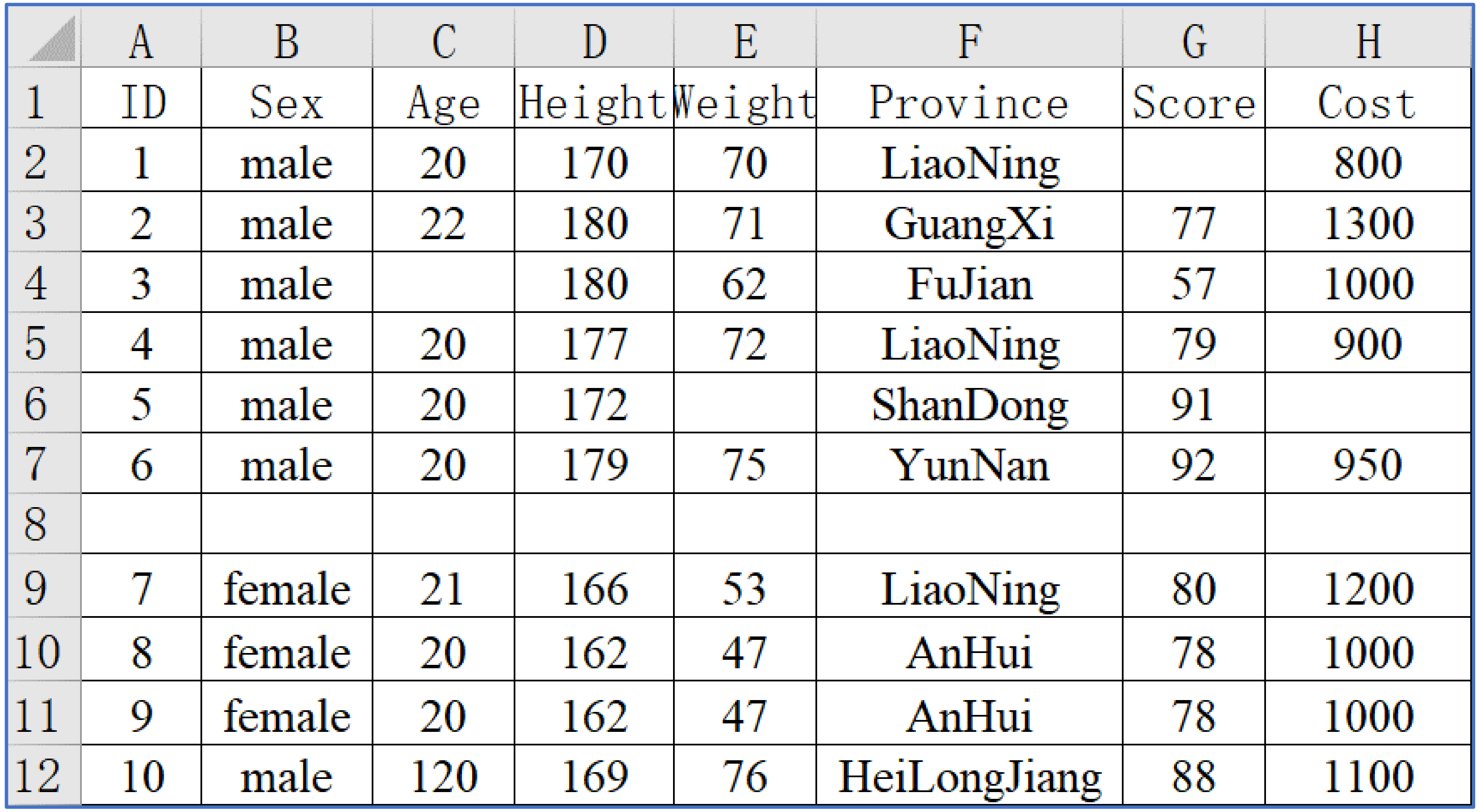
➢排名

```
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['Rank'] = data['Score'].rank(method='min', ascending=False)
print( data )
```



```
       Sex  Age  Height  Weight  Province  Score  Cost  Rank
ID
21  female   21     165      45  Shanghai     93  1200   2.0
22  female   19     167      42     HuBei     89   800   4.0
23    male   21     169      80     GanSu     93   900   2.0
24  female   21     160      49    HeBei      59  1100  10.0
25  female   21     162      54     GanSu     68  1300   7.0
26    male   21     181      77   SiChuan     62   800   9.0
27  female   21     162      49  ShanDong     65   950   8.0
28  female   22     160      52    ShanXi     73   800   6.0
29  female   20     161      51   GuangXi     80  1250   5.0
30  female   20     168      52   JiangSu     98   700   1.0
```

数据清洗



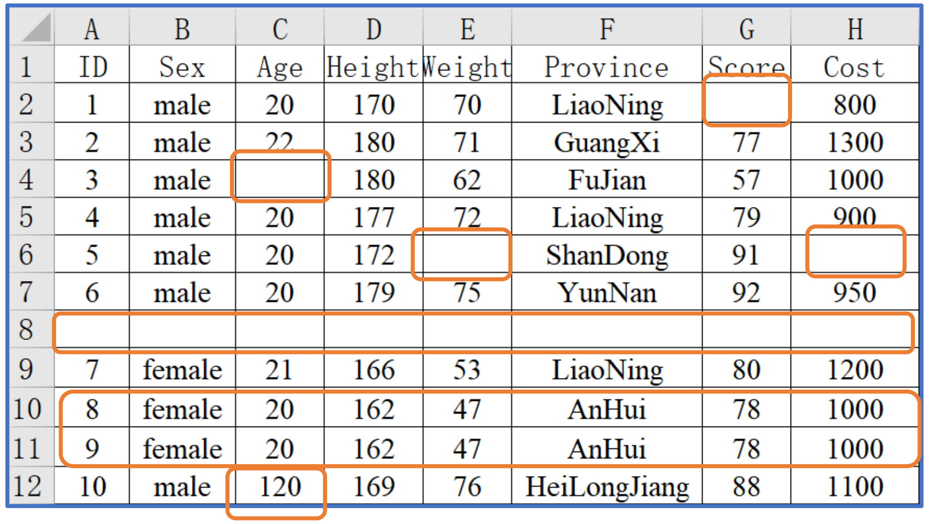
	A	B	C	D	E	F	G	H
1	ID	Sex	Age	Height	Weight	Province	Score	Cost
2	1	male	20	170	70	LiaoNing		800
3	2	male	22	180	71	GuangXi	77	1300
4	3	male		180	62	FuJian	57	1000
5	4	male	20	177	72	LiaoNing	79	900
6	5	male	20	172		ShanDong	91	
7	6	male	20	179	75	YunNan	92	950
8								
9	7	female	21	166	53	LiaoNing	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。

常问题：

- 数据缺失
- 数据重复
- 数据不一致



	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

丢弃缺失值dropna(axis,how,thresh,…)

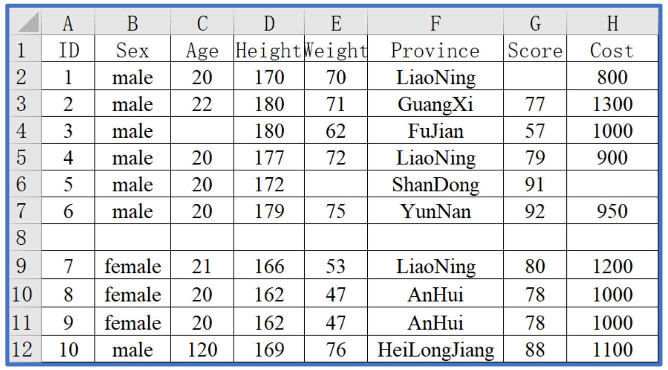
axis： 0表示按行滤除，1表示按列滤除，默认为axis=0

data. dropna()　　　　　#每行只要有空值，就将该行删除

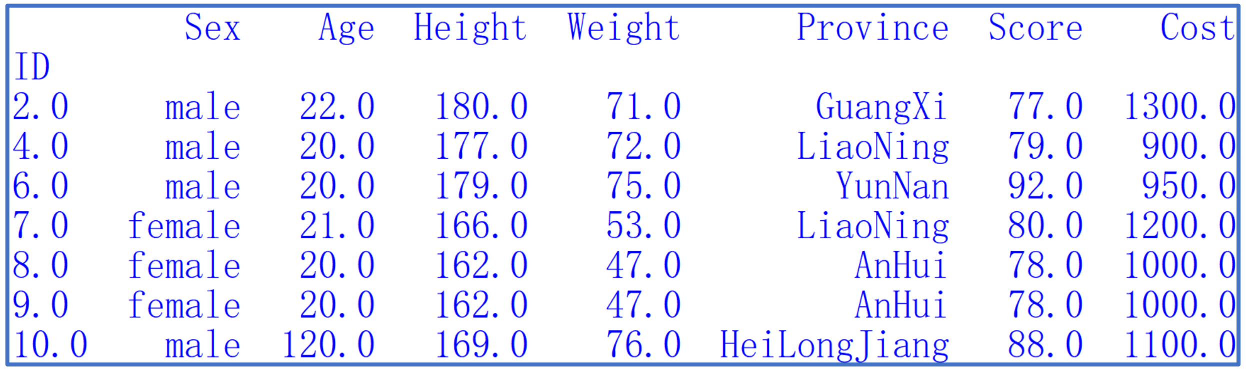
data. dropna(axis=1)　#每列只要有空值，就将该列删除

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)
```



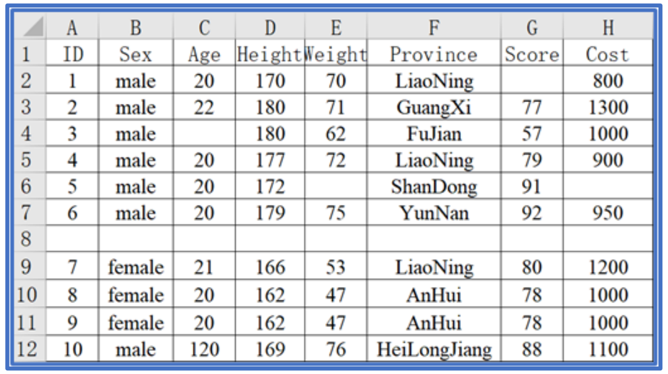
	A	B	C	D	E	F	G	H
1	ID	Sex	Age	Height	Weight	Province	Score	Cost
2	1	male	20	170	70	LiaoNing		800
3	2	male	22	180	71	GuangXi	77	1300
4	3	male		180	62	FuJian	57	1000
5	4	male	20	177	72	LiaoNing	79	900
6	5	male	20	172		ShanDong	91	
7	6	male	20	179	75	YunNan	92	950
8								
9	7	female	21	166	53	LiaoNing	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



```
         Sex    Age  Height  Weight      Province  Score    Cost
ID
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	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

```
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

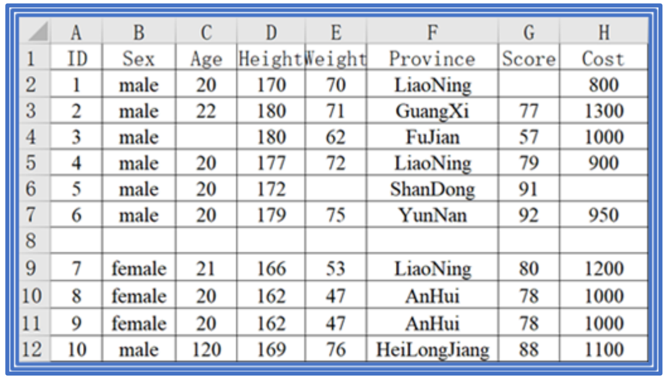
丢弃缺失值**dropna(axis,how,thresh,…)**

how: "all"表示滤除全部值都为NaN的行或列

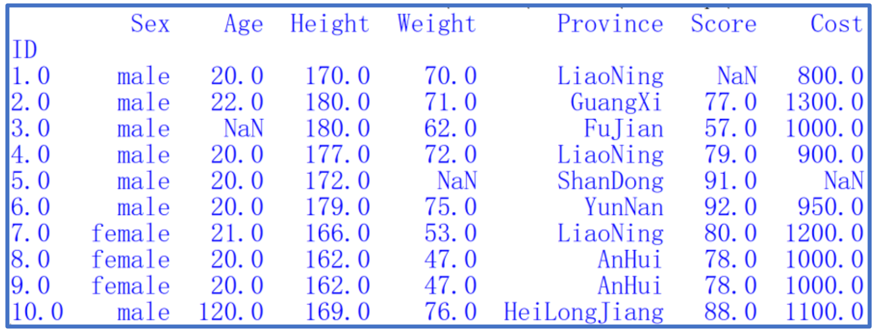
data. dropna(how='all')   #一行中全部为NaN才丢弃该行

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	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



```
         Sex    Age  Height  Weight      Province  Score    Cost
ID
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

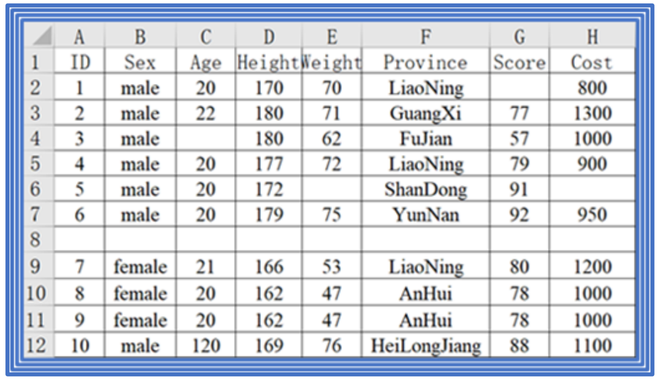
丢弃缺失值dropna(axis,how,thresh,…)

thresh： 只留下有效数据数大于或等于thresh的行或列

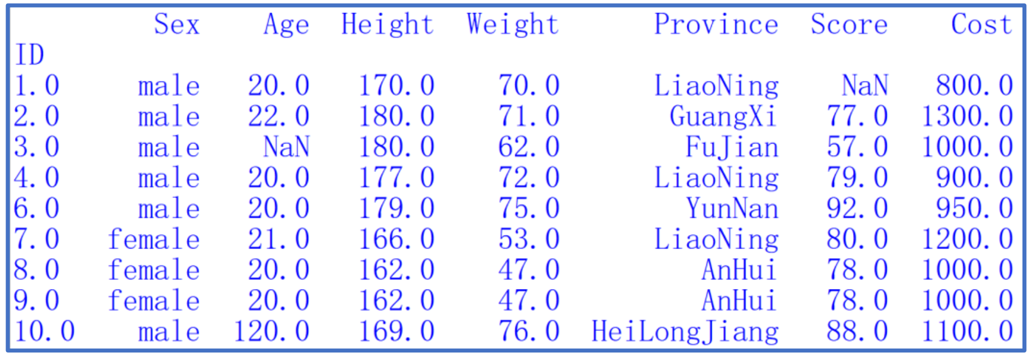
data. dropna(thresh=6)   # 每行至少6个非空值才保留

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	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



```
          Sex    Age  Height  Weight      Province  Score    Cost
ID
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

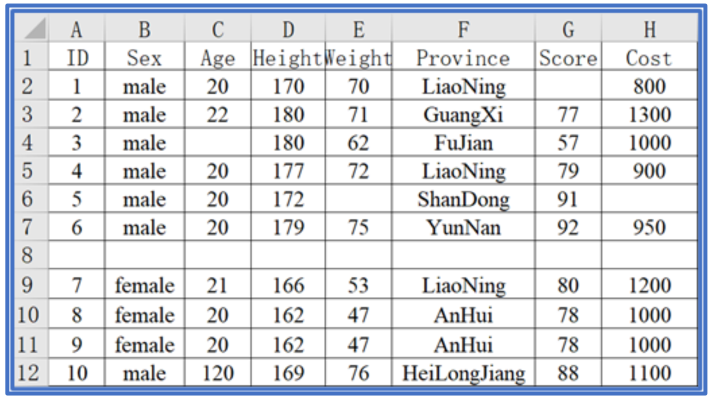
缺失值填充fillna(value, method,…)

value： 填充值，可以是标量、字典等

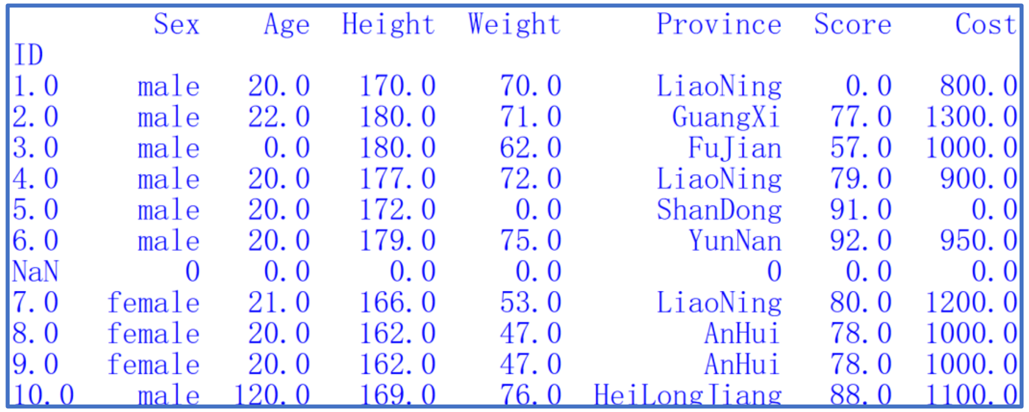
data. fillna(0) **#用0填充**

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	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



```
         Sex    Age  Height  Weight      Province  Score    Cost
ID
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

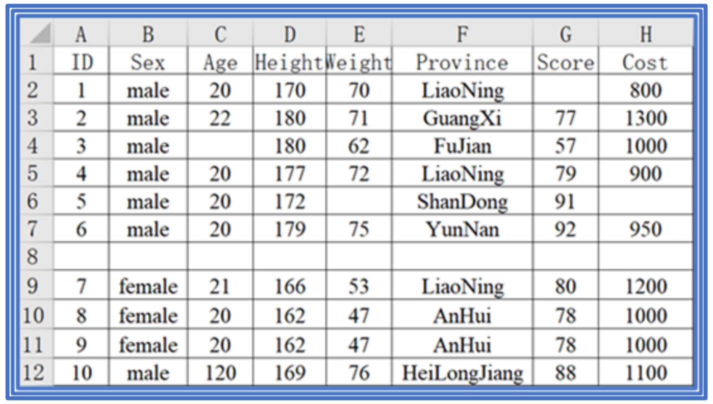
缺失值填充fillna(value, method,…)

value： 填充值，可以是标量、字典等

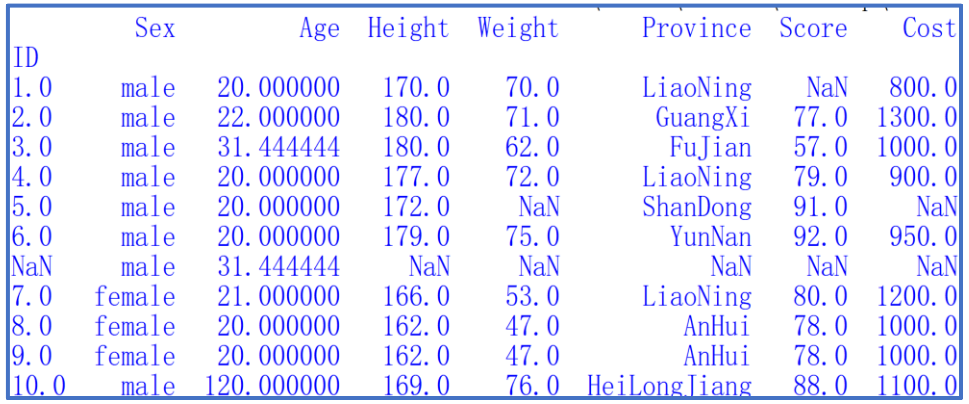
data. fillna({'Age': data['Age'].mean(), 'Sex': 'male'})

数据清洗案例

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



	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



```
         Sex         Age  Height  Weight      Province  Score    Cost
ID
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

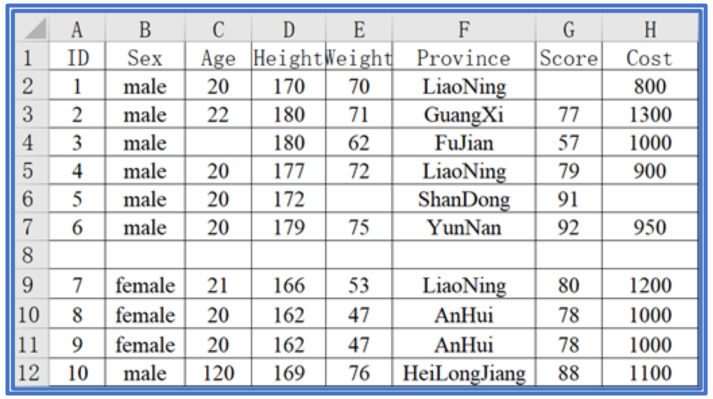
缺失值填充ffill()、bfill()

data. ffill() #在列方向上以上一个值替换

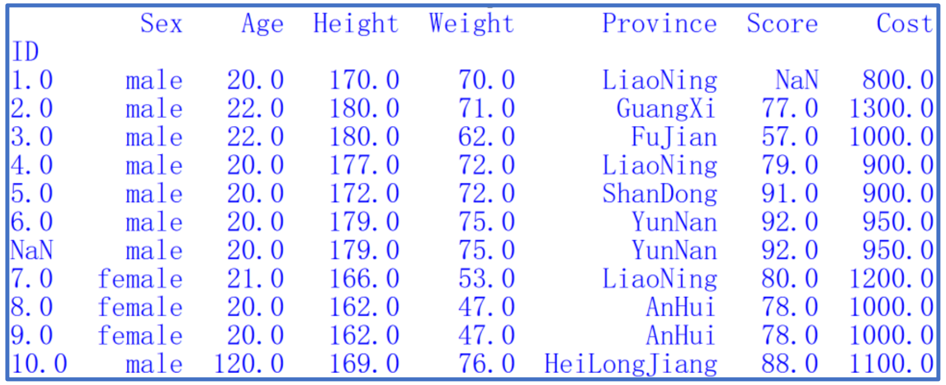
data. bfill() #在列方向上以下一个值替换

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	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



```
         Sex    Age  Height  Weight      Province  Score    Cost
ID
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, …)

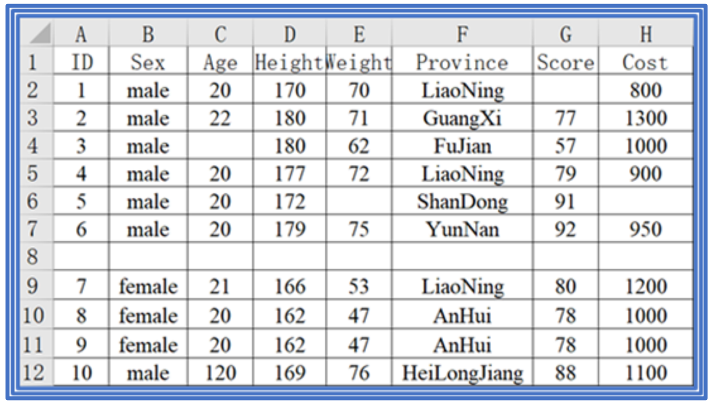
to_replace： 将被替代的值

value: 替换为的值

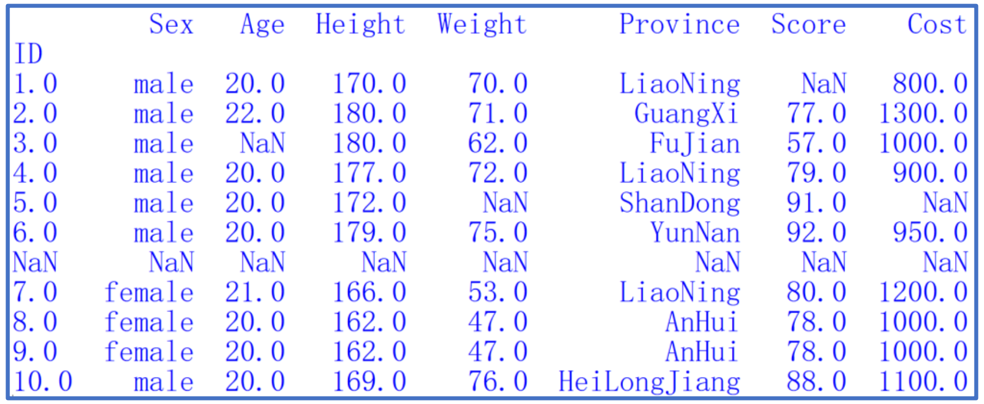
data['Age'].replace(120, 20)#将年龄120替换为20

数据清洗案例

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



	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



```
         Sex   Age  Height  Weight      Province  Score    Cost
ID
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
```

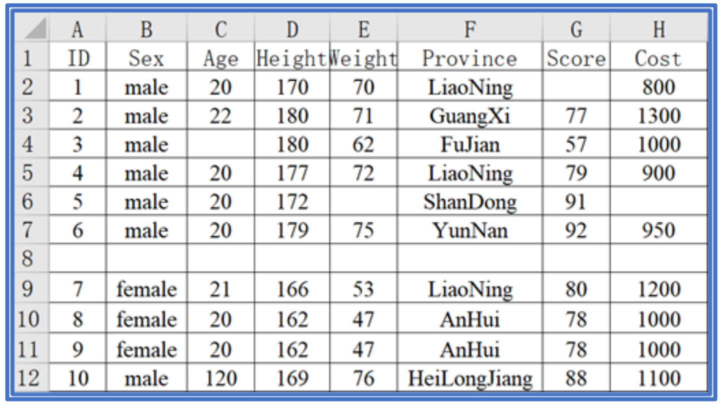
数据清洗

去掉重复值drop_duplicates()

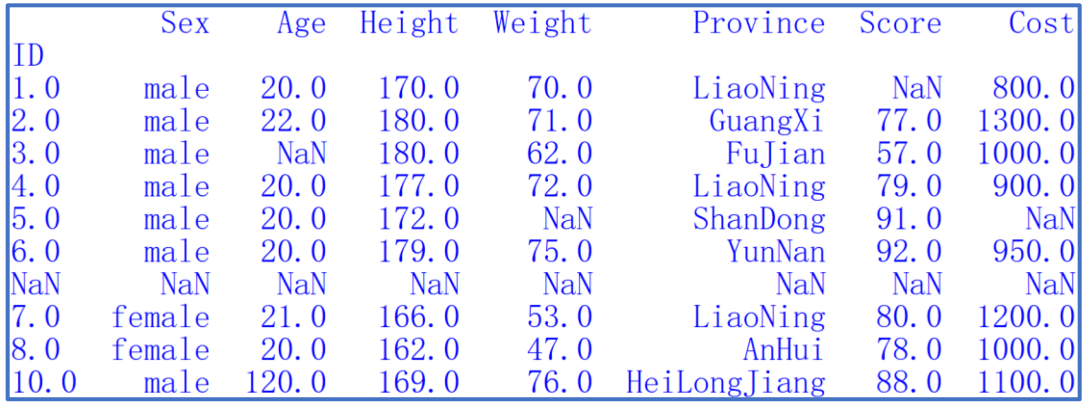
```
data. drop_duplicates()  #去掉重复的数据
```

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	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



```
          Sex    Age  Height  Weight      Province  Score    Cost
ID
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
```

常用数据挖掘算法

关联分析：旨在找出所有能把一组事件或数据项与另一组事件或数据项联系起来的强关联规则（拉关系）。**Apriori算法**是典型的关联分析算法。

例如，如果你今天在淘宝或京东等大型电商平台购买了手机，那么你明天登录网站的时候，很可能会购买手机套、数据线等和手机使用相关的产品。

常用数据挖掘算法

数据分类：分类是指通过对数据集的学习获得一个映射函数，从而将未知类别的样本映射到给定类别中（贴标签）。分类算法通常包括训练（即生成分类函数）和识别（即样品归类）两个阶段，典型的分类算法包括**决策树**、**贝叶斯**、**神经网络**、**支持向量机**等。

例如，在银行的历史信用卡数据中，有按时还款的正常客户，也有不能按时还款的逾期客户。银行希望通过用户的基础信息及历史交易数据等，判断其是正常客户还是逾期客户（这个就是分类标签），从而提前甄别出逾期客户，以尽可能地降低逾期发生率。

常用数据挖掘算法

数据聚类： 聚类旨在将数据集内具有相似特征的数据聚集成簇，从而使得同一个簇的数据特征尽可能相似，不同簇中的数据特征有明显的区别（找朋友）。典型的聚类算法包括**K均值聚类**、**K中心点**、**神经网络聚类**算法等。

聚类和分类算法的区别在于：分类任务中的训练数据集是有标签的，比方说正常/逾期，好/坏，Yes/No等，而聚类任务中的数据集则没有标签，只是根据特征的相似性将数据集聚集成不同的簇。比方说携程、去哪儿就会根据用户历史消费记录进行用户画像，总结出某一群体的共性，从而决定推荐的住宿酒店的档次、位置等属性。

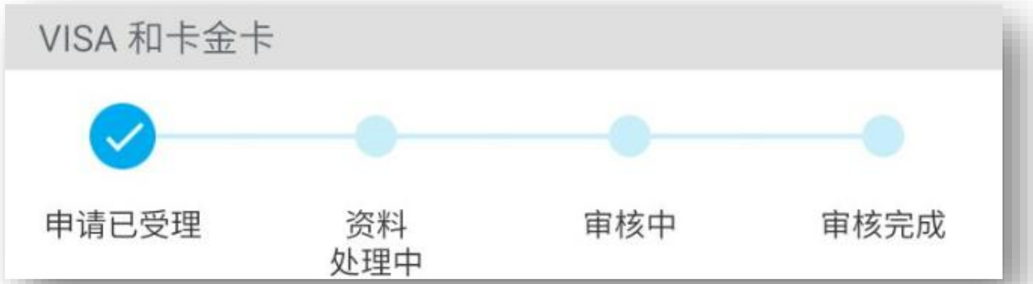
常用数据挖掘算法

时间序列分析：是根据数据过去和现在的变化规律去预测未来发展趋势的一种数据分析技术（测未来）。时间序列分析任务一般是针对那些与时间变化相关的指标，算法的目标是发现它随时间变化的趋势，从而能够进行预测。

在金融相关的宏观经济运行研究和管理工作中，经常要使用这时间序列分析来预测国民生产总值GDP，消费价格指数CPI等指标的变动情况。常见的时间序列分析算法主要包括简单移动平均、复杂差分整合移动平均、自回归等。

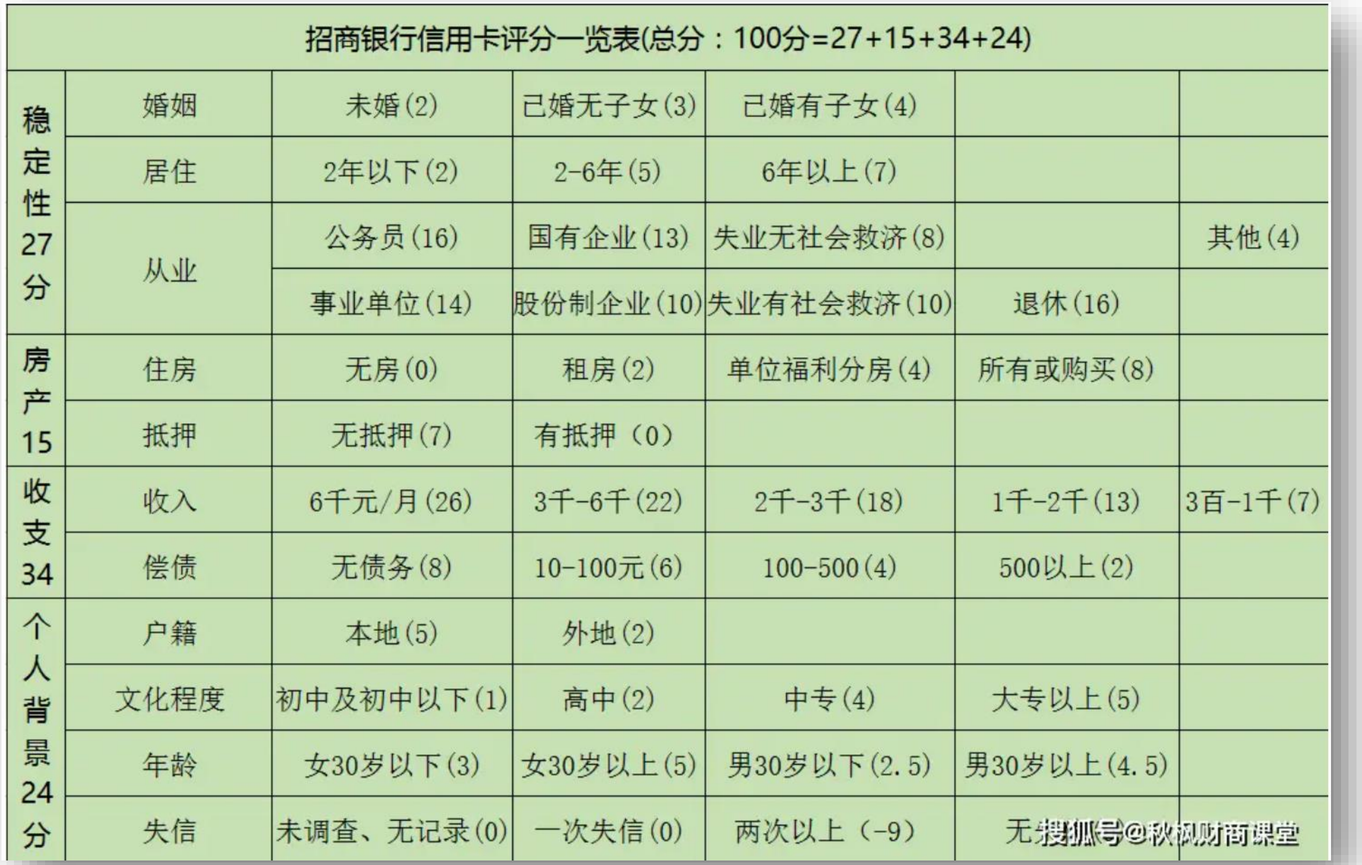
案例分析

以信用卡申请核发的业务为例，银行已经积累了大量客户的申请信息、消费记录和是否正常还款的历史数据，现在想以这些数据为基础，研发一套数据挖掘系统，用于在核准新用户申请办理信用卡期间的决策支持。



案例分析

1. 流程转化：将人工核准流程转换为计算机的处理流程。

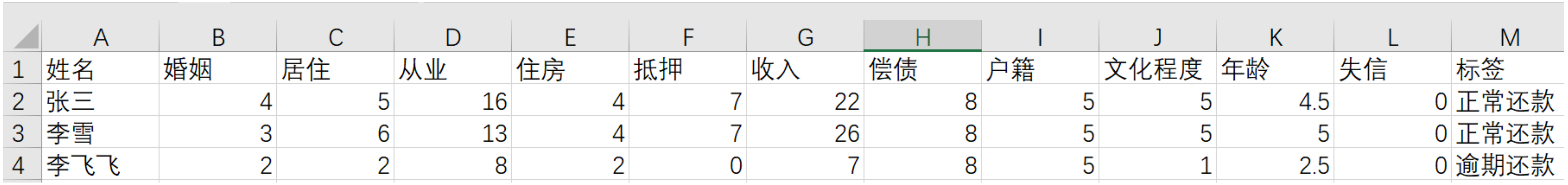


招商银行信用卡评分一览表(总分：100分=27+15+34+24)

稳定性27分	婚姻	未婚(2)	已婚无子女(3)	已婚有子女(4)		
	居住	2年以下(2)	2-6年(5)	6年以上(7)		
	从业	公务员(16)	国有企业(13)	失业无社会救济(8)		其他(4)
		事业单位(14)	股份制企业(10)	失业有社会救济(10)	退休(16)	
房产15	住房	无房(0)	租房(2)	单位福利分房(4)	所有或购买(8)	
	抵押	无抵押(7)	有抵押（0）			
收支34	收入	6千元/月(26)	3千-6千(22)	2千-3千(18)	1千-2千(13)	3百-1千(7)
	偿债	无债务(8)	10-100元(6)	100-500(4)	500以上(2)	
个人背景24分	户籍	本地(5)	外地(2)			
	文化程度	初中及初中以下(1)	高中(2)	中专(4)	大专以上(5)	
	年龄	女30岁以下(3)	女30岁以上(5)	男30岁以下(2.5)	男30岁以上(4.5)	
	失信	未调查、无记录(0)	一次失信(0)	两次以上（-9）	无	

案例分析

2. 选择算法：这是一个典型的分类问题，训练数据中的标签是正常还款和逾期还款，特征是用户还款的信息。希望能够使用老用户的申请信息来预测一个新客户是否会按时还款。



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	姓名	婚姻	居住	从业	住房	抵押	收入	偿债	户籍	文化程度	年龄	失信	标签
2	张三	4	5	16	4	7	22	8	5	5	4.5	0	正常还款
3	李雪	3	6	13	4	7	26	8	5	5	5	0	正常还款
4	李飞飞	2	2	8	2	0	7	8	5	1	2.5	0	逾期还款

案例分析

3. 模型的训练：将历史数据处理成算法能接受的数据格式后输入到算法中，生成预测函数。算法的输入数据是用户的申请信息或消费记录，输出结果是正常还款或逾期还款。

4. 调整参数：调整预测函数的参数，优化预测性能。

5. 模型预测：将预测函数用于后期预测，输入新用户信息，获得预测结果，即正常还款或逾期还款。

Python应用领域

文本分析：Jieba、Nltk…

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…

scikit-learn

scikit-learn： 基于NumPy, SciPy, matplotlib，可以实现数据预处理、分类、回归、降维、聚类、模型选择等常用的机器学习算法，是数据挖掘和数据分析的一个简单有效工具。

机器学习分类：有监督学习、无监督学习

谢　谢