Datawhale

动手学数据分析

Datawhale 出品

目录

项目初衷

这件事始于datawhale以前的数据分析课程，那时我作为一名学员的以《python for data analysis》这本书为教材教材，通过刷这本教材的代码来学习数据分析，书里对于pandas和numpy操作讲的很细，但是对于数据分析的逻辑的内容，就少了很多。所以很多学习者和我学完之后发现，敲了一堆代码并不知道它们有什么用。然后我也上过datawhale的另一门课程—数据挖掘实战。这门课程又比较偏模型和实战，直接给你一个任务，让你去完成，上手难度比较大，但是它的实战性可以让你对于什么是数据挖掘，以及数据挖掘的逻辑有很好的把握。所以有没有这样一门课，以项目为主线，将知识点孕育其中，通过边学，边做以及边被引导的方式来使学习效果达到更好，学完之后既能掌握pandas等的知识点又能掌握数据分析的大致思路和流程。通过调查发现，市面上这样的项目好像没有可以完全符合这样的标准（失望.jpg）。所以datawhale的小伙伴一起来做一门这样的开源课程，完成上面所说的那些小目标，让所有使用了我们课程的小伙伴可以更好的开启他的数据分析之路。

这门课程现在是1.0版本，从基础的数据分析操作和数据分析流程讲起。之后会不断加入新的内容（比如数据挖掘的算法之类的）。这是开源课程，会不断迭代，大家共同参与，一起努力。

既然这是一门诞生于datawhale的课程，学习它的时候搭配datawhale所配备其他资源会更好。我们提供的代码是jupyter形式的，里面有你所要完成的任务，也有我们给你的提示和引导，所以这样的形式再结合datawhale的组队学习，可以和大家一起讨论，一起补充资料，那么学习效果一定会加倍。还有，datawhale之前开源了一门pandas的教程—Joyful-Pandas。里面梳理了Pandas的逻辑以及代码展示，所以在我们数据分析的课程中，关于Pandas的操作，你可以参考*Joyful-Pandas*，可以让你的数据分析学习事半功倍。

关于我们项目的名字——[动手学数据分析]（Hands-on data analysis）。数据分析是一个要从一堆数字中看到真相的过程。学会操作数据只是数据分析的一半功力，剩下的另一半要用我们的大脑，多多思考，多多总结，更要多动手，实打实的的敲代码。所以也希望在学习这门课时，多去推理，多去问问为什么；多多练习，确保理论实践结合起来，在课程结束的时候一定会有大收获。

项目地址：https://github.com/datawhalechina/hands-on-data-analysis

课程编排与服用方法

课程编排

课程现分为三个单元，大致可以分为：数据基础操作，数据清洗与重构，建模和评估。

1. 第一部分：我们获得一个要分析的数据，我要学会如何加载数据，查看数据，然后学习Pandas的一些基础操作，最后开始尝试探索性的数据分析。
2. 第二部分：当我们可以比较熟练的操作数据并认识这个数据之后，我们需要开始数据清洗以及重构，将原始数据变为一个可用好用的数据，为之后放入模型做准备
3. 第三部分：我们根据任务需求不同，要考虑建立什么模型，我们接触流行的sklearn库，建立模型。然后一个模型的好坏，我们是需要评估的，之后我们会引入模型评估的一些改变和实现。

服用方法

我们的代码都是jupyter形式，每个部分的课程都分为课程和答案两个部分。学习期间，在课程代码中，完成所有的学习，自己查找资料，自己完成里面的代码操作，思考部分以及心得。之后可以和小伙伴讨论，分享资料和心得。关于答案部分，大家可以参考，但是由于数据分析本身是开放的，所以答案也是开放式的，更多希望大家可以有自己理解和答案。

反馈

1. 如果有任何想法可以联系邮箱（chenands@qq.com）
2. 欢迎大家提issues

成员名单

陈安东

中央民族大学研究生

GitHub： https://github.com/andongBlue

知乎：https://www.zhihu.com/people/wang-ya-fei-48

金娟娟

浙江大学硕士

业务与数据分析师

知乎：https://www.zhihu.com/people/wu-shi-lan-xiao-wang-zi

老表

数据分析爱好者，公众号简说Python作者

个人公众号：简说Python

杨佳达

数据挖掘师

GitHub:https://github.com/yangjiada

李玲

算法工程师

知乎：https://www.zhihu.com/people/liu-yu-18-38

张文涛

中山大学博士研究生

GitHub： https://github.com/Fatflower

高立业

太原理工大学研究生

GitHub：https://github.com/0-yy-0

写在最前面

这门课程得主要目的是通过真实的数据，以实战的方式了解数据分析的流程和熟悉数据分析python的基本操作。知道了课程的目的之后，我们接下来我们要正式的开始数据分析的实战教学，完成kaggle上泰坦尼克的任务，实战数据分析全流程。

这里有两份资料需要大家准备：

图书《Python for Data Analysis》第六章和 baidu.com and google.com（善用搜索引擎）

1 第一章(PART 1)：数据加载

1.1 载入数据

数据集下载 https://www.kaggle.com/c/titanic/overview

1.1.1 任务一：导入numpy和pandas

```
import numpy as np
import pandas as pd
```

【提示】如果加载失败，学会如何在你的python环境下安装numpy和pandas这两个库

1.1.2 任务二：载入数据

(1) 使用相对路径载入数据

(2) 使用绝对路径载入数据

```
df = pd.read_csv('train.csv')
df.head(3)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S

```
df = pd.read_csv('/Users/chenandong/Documents/datawhale数据分析每个人题目设计/招募阶段/第一单元项目集合/train.csv')
df.head(3)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S

提示：相对路径载入报错时，尝试使用os.getcwd()查看当前工作目录。

思考：知道数据加载的方法后，试试pd.read_csv()和pd.read_table()的不同，如果想让他们效果一样，需要怎么做？了解一下'.tsv'和'.csv'的不同，如何加载这两个数据集？

总结：加载的数据是所有工作的第一步，我们的工作会接触到不同的数据格式（eg:.csv;.tsv;.xlsx）,但是加载的方法和思路都是一样的，在以后工作和做项目的过程中，遇到之前没有碰到的问题，要多多查资料吗，使用googel，了解业务逻辑，明白输入和输出是什么。

1.1.3 任务三：每1000行为一个数据模块，逐块读取

```
1 chunker = pd.read_csv('train.csv', chunksize=1000)
```

思考：什么是逐块读取？为什么要逐块读取呢？

1.1.4 任务四：将表头改成中文，索引改为乘客ID

对于某些英文资料，我们可以通过翻译来更直观的熟悉我们的数据

PassengerId => 乘客ID

Survived => 是否幸存

Pclass => 乘客等级(1/2/3等舱位)

Name => 乘客姓名

Sex => 性别

Age => 年龄

SibSp => 堂兄弟/妹个数

Parch => 父母与小孩个数

Ticket => 船票信息

Fare => 票价

Cabin => 客舱

Embarked => 登船港口

```
df = pd.read_csv('train.csv', names=['乘客ID','是否幸存','仓位等级','姓名','性别','年龄','兄弟姐妹个数','父母子女个数','船票信息','票价','客舱','登船港口'],index_col='乘客ID',header=0)
df.head()
```

	是否幸存	仓位等级	姓名	性别	年龄	兄弟姐妹个数	父母子女个数	船票信息	票价	客舱	登船港口
乘客ID											
1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

思考：所谓将表头改为中文其中一个思路是：将英文额度表头替换成中文。还有其他的方法吗？

1.2 初步观察

导入数据后，你可能要对数据的整体结构和样例进行概览，比如说，数据大小、有多少列，各列都是什么格式的，是否包含null等

1.2.1 任务一：查看数据的基本信息

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
Data columns (total 11 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   是否幸存    891 non-null    int64
 1   仓位等级    891 non-null    int64
 2   姓名      891 non-null    object
 3   性别      891 non-null    object
 4   年龄      714 non-null    float64
 5   兄弟姐妹个数  891 non-null    int64
 6   父母子女个数  891 non-null    int64
 7   船票信息    891 non-null    object
 8   票价      891 non-null    float64
 9   客舱      204 non-null    object
 10  登船港口    889 non-null    object
dtypes: float64(2), int64(4), object(5)
memory usage: 83.5+ KB
```

1.2.2 任务二：观察表格前10行的数据和后15行的数据

```
1 df.head(10)
```

	是否幸存	仓位等级	姓名	性别	年龄	兄弟姐妹个数	父母子女个数	船票信息	票价	客舱	登船港口
乘客ID											
1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q
7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	S
8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	S
9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	NaN	S
10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	NaN	C

```
1 df.tail(15)
```

	是否幸存	仓位等级	姓名	性别	年龄	兄弟姐妹个数	父母子女个数	船票信息	票价	客舱	登船港口
乘客ID											
877	0	3	Gustafsson, Mr. Alfred Ossian	male	20.0	0	0	7534	9.8458	NaN	S
878	0	3	Petroff, Mr. Nedelio	male	19.0	0	0	349212	7.8958	NaN	S
879	0	3	Laleff, Mr. Kristo	male	NaN	0	0	349217	7.8958	NaN	S
880	1	1	Potter, Mrs. Thomas Jr (Lily Alexenia Wilson)	female	56.0	0	1	11767	83.1583	C50	C
881	1	2	Shelley, Mrs. William (Imanita Parrish Hall)	female	25.0	0	1	230433	26.0000	NaN	S
882	0	3	Markun, Mr. Johann	male	33.0	0	0	349257	7.8958	NaN	S
883	0	3	Dahlberg, Miss. Gerda Ulrika	female	22.0	0	0	7552	10.5167	NaN	S
884	0	2	Banfield, Mr. Frederick James	male	28.0	0	0	C.A./SOTON 34068	10.5000	NaN	S
885	0	3	Sutehall, Mr. Henry Jr	male	25.0	0	0	SOTON/OQ 392076	7.0500	NaN	S
886	0	3	Rice, Mrs. William (Margaret Norton)	female	39.0	0	5	382652	29.1250	NaN	Q
887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S
888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	S
889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	S
890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	C
891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	Q

1.2.3 任务三：判断数据是否为空，为空的地方返回True，其余地方返回False

```
1 df.isnull().head()
```

	是否幸存	仓位等级	姓名	性别	年龄	兄弟姐妹个数	父母子女个数	船票信息	票价	客舱	登船港口
乘客ID											
1	False	False	False	False	False	False	False	False	False	True	False
2	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	True	False
4	False	False	False	False	False	False	False	False	False	False	False
5	False	False	False	False	False	False	False	False	False	True	False

总结：上面的操作都是数据分析中对于数据本身的观察

思考：对于一个数据，还可以从哪些方面来观察？找找答案，这个将对下面的数据分析有很大的帮助

1.3 保存数据

1.3.1 任务一：将你加载并做出改变的数据，在工作目录下保存为一个新文件train_chinese.csv

```
df.to_csv('train_chinese.csv')
```

总结：数据的加载以及入门，接下来就要接触数据本身的运算，我们将主要掌握numpy和pandas在工作和项目场景的运用。

复习：数据分析的第一步，加载数据我们已经学习完毕了。当数据展现在我们面前的时候，我们所要做的第一步就是认识他，今天我们要学习的就是**了解字段含义以及初步观察数据**。

2 第一章(PART 2)：pandas基础

2.1 知道你的数据叫什么

我们学习pandas的基础操作，那么上一节通过pandas加载之后的数据，其数据类型是什么呢？

2.1.1 任务一：pandas中有两个数据类型DateFrame和Series，通过查找简单了解他们。然后自己写一个关于这两个数据类型的小例子[开放题]

```
import numpy as np
import pandas as pd
```

```
sdata = {'Ohio': 35000, 'Texas': 71000, 'Oregon': 16000, 'Utah': 5000}
example_1 = pd.Series(sdata)
example_1
```

```
Ohio      35000
Texas     71000
Oregon    16000
Utah       5000
dtype: int64
```

```
data = {'state': ['Ohio', 'Ohio', 'Ohio', 'Nevada', 'Nevada', 'Nevada'],
        'year': [2000, 2001, 2002, 2001, 2002, 2003],'pop': [1.5, 1.7, 3.6, 2.4, 2.9, 3.2]}
example_2 = pd.DataFrame(data)
example_2
```

	state	year	pop
0	Ohio	2000	1.5
1	Ohio	2001	1.7
2	Ohio	2002	3.6
3	Nevada	2001	2.4
4	Nevada	2002	2.9
5	Nevada	2003	3.2

2.1.2 任务二：根据上节课的方法载入"train.csv"文件

```
df = pd.read_csv('/Users/chenandong/Documents/datawhale数据分析每个人题目设计/titanic/train.csv')
df.head(3)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S

也可以加载上一节课保存的"train_chinese.csv"文件。

2.1.3 任务三：查看DataFrame数据的每列的项

```
df.columns
```

```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

2.1.4 任务四：查看"cabin"这列的所有项 [有多种方法]

```
df['Cabin'].head(3)
```

```
0     NaN
1     C85
2     NaN
3    C123
4     NaN
Name: Cabin, dtype: object
```

```
df.Cabin.head(3)
```

```
0     NaN
1     C85
2     NaN
3    C123
4     NaN
Name: Cabin, dtype: object
```

2.1.5 任务五：加载文件"test_1.csv"，然后对比"train.csv"，看看有哪些多出的列，然后将多出的列删除

经过我们的观察发现一个测试集test_1.csv有一列是多余的，我们需要将这个多余的列删去

```
test_1 = pd.read_csv('test_1.csv')
test_1.head(3)
```

	Unnamed: 0	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	a
0	0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	100
1	1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	100
2	2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	100

```
# 删除多余的列
del test_1['a']
test_1.head(3)
```

	Unnamed: 0	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S

思考：还有其他的删除多余的列的方式吗？

2.1.6 任务六： 将['PassengerId','Name','Age','Ticket']这几个列元素隐藏，只观察其他几个列元素

```
df.drop(['PassengerId','Name','Age','Ticket'],axis=1).head(3)
```

	Survived	Pclass	Sex	SibSp	Parch	Fare	Cabin	Embarked
0	0	3	male	1	0	7.2500	NaN	S
1	1	1	female	1	0	71.2833	C85	C
2	1	3	female	0	0	7.9250	NaN	S

思考：对比任务五和任务六，是不是使用了不一样的方法(函数)，如果使用一样的函数如何完成上面的不同的要求呢？

思考回答：

如果想要完全的删除你的数据结构，使用inplace=True，因为使用inplace就将原数据覆盖了，所以这里没有用

```
# 思考回答
df.head(3)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S

2.2 筛选的逻辑

表格数据中，最重要的一个功能就是要具有可筛选的能力，选出我所需要的信息，丢弃无用的信息。

下面我们还是用实战来学习pandas这个功能。

2.2.1 任务一： 我们以"Age"为筛选条件，显示年龄在10岁以下的乘客信息。

```
df[df["Age"]<10].head(3)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.075	NaN	S
10	11	1	3	Sandstrom, Miss. Marguerite Rut	female	4.0	1	1	PP 9549	16.700	G6	S
16	17	0	3	Rice, Master. Eugene	male	2.0	4	1	382652	29.125	NaN	Q

2.2.2 任务二： 以"Age"为条件，将年龄在10岁以上和50岁以下的乘客信息显示出来，并将这个数据命名为midage

```
midage = df[(df["Age"]>10)& (df["Age"]<50)]
midage.head(3)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S

提示：了解pandas的条件筛选方式以及如何使用交集和并集操作

2.2.3 任务三：将midage的数据中第100行的"Pclass"和"Sex"的数据显示出来

```
midage = midage.reset_index(drop=True)
midage.head(3)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S

思考：这个reset_index()函数的作用是什么？如果不用这个函数，下面的任务会出现什么情况？

```
midage.loc[[100],['Pclass','Sex']]
```

	Pclass	Sex
100	2	male

2.2.4 任务四：使用loc方法将midage的数据中第100，105，108行的"Pclass"，"Name"和"Sex"的数据显示出来

```
midage.loc[[100,105,108],['Pclass','Name','Sex']] #因为你主动的延长了行的距离，所以会产生表格形式
```

	Pclass	Name	Sex
100	2	Byles, Rev. Thomas Roussel Davids	male
105	3	Cribb, Mr. John Hatfield	male
108	3	Calic, Mr. Jovo	male

提示：使用pandas提出的简单方式，你可以看看loc方法

对比整体的数据位置，你有发现什么问题吗？那么如何解决？

2.2.5 任务五：使用iloc方法将midage的数据中第100，105，108行的"Pclass"，"Name"和"Sex"的数据显示出来

```
midage.iloc[[100,105,108],[2,3,4]]
```

	Pclass	Name	Sex
100	2	Byles, Rev. Thomas Roussel Davids	male
105	3	Cribb, Mr. John Hatfield	male
108	3	Calic, Mr. Jovo	male

复习：在前面我们已经学习了Pandas基础，知道利用Pandas读取csv数据的增删查改，今天我们要学习的就是**探索性数据分析**，主要介绍如何利用Pandas进行排序、算术计算以及计算描述函数describe()的使用。

3 第一章(PART 3)：探索性数据分析

3.1 开始之前，导入numpy、pandas包和数据

```
#加载所需的库
import numpy as np
import pandas as pd

#载入之前保存的train_chinese.csv数据，关于泰坦尼克号的任务，我们就使用这个数据
text = pd.read_csv('train_chinese.csv')
text.head()
```

	乘客ID	是否幸存	仓位等级	姓名	性别	年龄	兄弟姐妹个数	父母子女个数	船票信息	票价	客舱	登船港口
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

3.2 了解你的数据吗？

教材《Python for Data Analysis》第五章

3.2.1 任务一：利用Pandas对示例数据进行排序，要求升序

```
# 具体请看《利用Python进行数据分析》第五章 排序和排名 部分

#自己构建一个都为数字的DataFrame数据
frame = pd.DataFrame(np.arange(8).reshape((2, 4)),
                     index=['2', '1'],
                     columns=['d', 'a', 'b', 'c'])
frame

```

	d	a	b	c
2	0	1	2	3
1	4	5	6	7

代码解析

pd.DataFrame() ： 创建一个DataFrame对象

np.arange(8).reshape((2, 4)) : 生成一个二维数组（2*4）,第一列：0，1，2，3 第二列：4，5，6，7

index=['2, 1] ： DataFrame 对象的索引列

columns=['d', 'a', 'b', 'c'] ： DataFrame 对象的索引行

```
# 大多数时候我们都是想根据列的值来排序,所以，将你构建的DataFrame中的数据根据某一列，升序排列
frame.sort_values(by='c', ascending=False)
```

	d	a	b	c
1	4	5	6	7
2	0	1	2	3

可以看到sort_values这个函数中by参数指向要排列的列，ascending参数指向排序的方式（升序还是降序）

总结：下面将不同的排序方式做一个小总结

```
# 让行索引升序排序
frame.sort_index()
```

	d	a	b	c
1	4	5	6	7
2	0	1	2	3

```
# 让列索引升序排序
frame.sort_index(axis=1)
```

	a	b	c	d
2	1	2	3	0
1	5	6	7	4

```
# 让列索引降序排序
frame.sort_index(axis=1, ascending=False)
```

	d	c	b	a
2	0	3	2	1
1	4	7	6	5

```
# 让任选两列数据同时降序排序
frame.sort_values(by=['a', 'c'])
```

	d	a	b	c
2	0	1	2	3
1	4	5	6	7

3.2.2 任务二：对泰坦尼克号数据（trian.csv）按票价和年龄两列进行综合排序（降序排列），从数据中你能发现什么

```
'''
在开始我们已经导入了train_chinese.csv数据，而且前面我们也学习了导入数据过程，根据上面学习，我们直接对目标列进行排序即可
head(20) ： 读取前20条数据
'''

text.sort_values(by=['票价', '年龄'], ascending=False).head(3)
```

	乘客ID	是否幸存	仓位等级	姓名	性别	年龄	兄弟姐妹个数	父母子女个数	船票信息	票价	客舱	登船港口
679	680	1	1	Cardeza, Mr. Thomas Drake Martinez	male	36.0	0	1	PC 17755	512.3292	B51 B53 B55	C
258	259	1	1	Ward, Miss. Anna	female	35.0	0	0	PC 17755	512.3292	NaN	C
737	738	1	1	Lesurer, Mr. Gustave J	male	35.0	0	0	PC 17755	512.3292	B101	C

思考：排序后，如果我们仅仅关注年龄和票价两列。根据常识我知道发现票价越高的应该客舱越好，所以我们会明显看出，票价前20的乘客中存活的有14人，这是相当高的一个比例，那么我们后面是不是可以进一步分析一下票价和存活之间的关系，年龄和存活之间的关系呢？当你开始发现数据之间的关系了，数据分析就开始了。

当然，这只是我的想法，你还可以有更多想法，欢迎写在你的学习笔记中。

3.2.3 任务三：利用Pandas进行算术计算，计算两个DataFrame数据相加结果

```
# 具体请看《利用Python进行数据分析》第五章 算术运算与数据对齐 部分

#建立一个例子
frame1_a = pd.DataFrame(np.arange(9.).reshape(3, 3),
                        columns=['a', 'b', 'c'],
                        index=['one', 'two', 'three'])
frame1_b = pd.DataFrame(np.arange(12.).reshape(4, 3),
                        columns=['a', 'e', 'c'],
                        index=['first', 'one', 'two', 'second'])
frame1_a
```

	a	b	c
one	0.0	1.0	2.0
two	3.0	4.0	5.0
three	6.0	7.0	8.0

```
frame1_b
```

	a	e	c
first	0.0	1.0	2.0
one	3.0	4.0	5.0
two	6.0	7.0	8.0
second	9.0	10.0	11.0

```
#将frame_a和frame_b进行相加
frame1_a + frame1_b
```

	a	b	c	e
first	NaN	NaN	NaN	NaN
one	3.0	NaN	7.0	NaN
second	NaN	NaN	NaN	NaN
three	NaN	NaN	NaN	NaN
two	9.0	NaN	13.0	NaN

提醒：两个DataFrame相加后，会返回一个新的DataFrame，对应的行和列的值会相加，没有对应的会变成空值NaN。

当然，DataFrame还有很多算术运算，如减法，除法等，有兴趣的同学可以看《利用Python进行数据分析》第五章 算术运算与数据对齐 部分，多在网络上查找相关学习资料。

3.2.4 任务四：通过泰坦尼克号数据如何计算出在船上最大的家族有多少人？

```
'''
还是用之前导入的chinese_train.csv如果我们想看看在船上，最大的家族有多少人（‘兄弟姐妹个数’+‘父母子女个数’），我们该怎么做呢？
'''
max(text['兄弟姐妹个数'] + text['父母子女个数'])
```

```
10
```

是的，如上，很简单，我们只需找出兄弟姐妹个数和父母子女个数之和最大的数就行，先让这两列相加返回一个DataFrame，然后用max函数求出最大值，当然你还可以想出很多方法和思考角度，欢迎你来说出你的看法。

3.2.5 任务五：学会使用Pandas describe()函数查看数据基本统计信息

```
#(1) 关键知识点示例做一遍（简单数据）
# 具体请看《利用Python进行数据分析》第五章 汇总和计算描述统计 部分

#建立一个例子
frame2 = pd.DataFrame([[1.4, np.nan],
                       [7.1, -4.5],
                       [np.nan, np.nan],
                       [0.75, -1.3]
                      ], index=['a', 'b', 'c', 'd'], columns=['one', 'two'])
frame2
```

	one	two
a	1.40	NaN
b	7.10	-4.5
c	NaN	NaN
d	0.75	-1.3

```
# 调用 describe 函数，观察frame2的数据基本信息

frame2.describe()

'''
count ： 样本数据大小
mean ： 样本数据的平均值
std ： 样本数据的标准差
min ： 样本数据的最小值
25% ： 样本数据25%的时候的值
50% ： 样本数据50%的时候的值
75% ： 样本数据75%的时候的值
max ： 样本数据的最大值
'''
```

	one	two
count	3.000000	2.000000
mean	3.083333	-2.900000
std	3.493685	2.262742
min	0.750000	-4.500000
25%	1.075000	-3.700000
50%	1.400000	-2.900000
75%	4.250000	-2.100000
max	7.100000	-1.300000

3.2.6 任务六：分别看看泰坦尼克号数据集中 票价、父母子女 这列数据的基本统计数据，你能发现什么？

```
'''
看看泰坦尼克号数据集中 票价 这列数据的基本统计数据
'''
text['票价'].describe()
```

```
count    891.000000
mean      32.204208
std       49.693429
min        0.000000
25%        7.910400
50%       14.454200
75%       31.000000
max      512.329200
Name: 票价, dtype: float64
```

思考：从上面数据我们可以看出，

一共有891个票价数据，

平均值约为：32.20，

标准差约为49.69，说明票价波动特别大，

25%的人的票价是低于7.91的，50%的人的票价低于14.45，75%的人的票价低于31.00，

票价最大值约为512.33，最小值为0。

当然，这只是我的想法，你还可以有更多想法，欢迎写在你的学习笔记中。

```
'''
通过上面的例子，我们再看看泰坦尼克号数据集中 父母子女个数 这列数据的基本统计数据，然后可以说出你的想法
'''
text['父母子女个数'].describe()
```

```
count    891.000000
mean       0.381594
std        0.806057
min        0.000000
25%        0.000000
50%        0.000000
75%        0.000000
max        6.000000
Name: 父母子女个数, dtype: float64
```

思考：有更多想法，欢迎写在你的学习笔记中。

总结：本节中我们通过Pandas的一些内置函数对数据进行了初步统计查看，这个过程最重要的不是大家得掌握这些函数，而是看懂从这些函数出来的数据，构建自己的数据分析思维，这也是第一章最重要的点，希望大家学完第一章能对数据有个基本认识，了解自己在做什么，为什么这么做，后面的章节我们将开始对数据进行清洗，进一步分析。

回顾&引言：前面一章的内容大家可以感觉到我们主要是对基础知识做一个梳理，让大家了解数据分析的一些操作，主要做了数据的各个角度的观察。那么在这里，我们主要是做数据分析的流程性学习，主要是包括了数据清洗以及数据的特征处理，数据重构以及数据可视化。这些内容是为数据分析最后的建模和模型评价做一个铺垫。

4 第二章(PART 1)：数据清洗及特征处理

4.1 开始之前，导入numpy、pandas包和数据

```
#加载所需的库
import numpy as np
import pandas as pd

#加载数据train.csv
df = pd.read_csv('train.csv')
df.head(3)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

4.2 数据清洗简述

我们拿到的数据通常是不干净的，所谓的不干净，就是数据中有缺失值，有一些异常点等，需要经过一定的处理才能继续做后面的分析或建模，所以拿到数据的第一步是进行数据清洗，本章我们将学习缺失值、重复值、字符串和数据转换等操作，将数据清洗成可以分析或建模的样子。

4.3 缺失值观察与处理

我们拿到的数据经常会有很多缺失值，比如我们可以看到Cabin列存在NaN，那其他列还有没有缺失值，这些缺失值要怎么处理呢

4.3.1 任务一：缺失值观察

(1) 请查看每个特征缺失值个数

(2) 请查看Age， Cabin， Embarked列的数据

以上方式都有多种方式，所以建议大家学习的时候多多益善

```
#方法一
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

```
#方法二
df.isnull().sum()
```

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

```
df[['Age','Cabin','Embarked']].head(3)
```

	Age	Cabin	Embarked
0	22.0	NaN	S
1	38.0	C85	C
2	26.0	NaN	S
3	35.0	C123	S
4	35.0	NaN	S

4.3.2 任务二：对缺失值进行处理

(1)处理缺失值一般有几种思路

(2) 请尝试对Age列的数据的缺失值进行处理

(3) 请尝试使用不同的方法直接对整张表的缺失值进行处理

以下是举例：

```
df.dropna().head(3)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	S
10	11	1	3	Sandstrom, Miss. Marguerite Rut	female	4.0	1	1	PP 9549	16.7000	G6	S
11	12	1	1	Bonnell, Miss. Elizabeth	female	58.0	0	0	113783	26.5500	C103	S

```
df.fillna(0).head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	0	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	0	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	0	S

```
df[df['Age']==None]=0
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

思考：dropna和fillna有哪些参数，分别如何使用呢？

参考： https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.dropna.html

参考：https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.fillna.html

4.4 重复值观察与处理

由于这样那样的原因，数据中会不会存在重复值呢，如果存在要怎样处理呢

4.4.1 任务一：请查看数据中的重复值

```
df[df.duplicated()]
```

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
-	-	-	-	-	-	-	-	-	-	-	-

4.4.2 任务二：对重复值进行处理

(1)重复值有哪些处理方式呢？

(2)处理我们数据的重复值

方法多多益善

以下是对整个行有缺失值的清理的方法举例：

```
df.drop_duplicates().head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

4.4.3 任务三：将前面清洗的数据保存为csv格式

```
df.to_csv('test_clear.csv')
```

4.5 特征观察与处理

我们对特征进行一下观察，可以把特征大概分为两大类：

数值型特征：Survived，Pclass，Age，SibSp，Parch，Fare，其中Survived，Pclass为离散型数值特征，Age，SibSp，Parch，Fare为连续型数值特征

文本型特征：Name，Sex，Cabin，Embarked，Ticket，其中Sex，Cabin，Embarked，Ticket为类别型文本特征。

数值型特征一般可以直接用于模型的训练，但有时候为了模型的稳定性及鲁棒性会对连续变量进行离散化。文本型特征往往需要转换成数值型特征才能用于建模分析。

4.5.1 任务一：对年龄进行分箱（离散化）处理

(1) 分箱操作是什么？

(2) 将连续变量Age平均分箱成5个年龄段，并分别用类别变量12345表示

(3) 将连续变量Age划分为[0,5) [5,15) [15,30) [30,50) [50,80)五个年龄段，并分别用类别变量12345表示

(4) 将连续变量Age按10% 30% 50 70% 90%五个年龄段，并用分类变量12345表示

(5) 将上面的获得的数据分别进行保存，保存为csv格式

```
#将连续变量Age平均分箱成5个年龄段，并分别用类别变量12345表示
df['AgeBand'] = pd.cut(df['Age'], 5,labels = ['1','2','3','4','5'])
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	AgeBand
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	2
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	3
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	2
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	3
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	3

```
df.to_csv('test_ave.csv')
```

```
#将连续变量Age划分为[0,5) [5,15) [15,30) [30,50) [50,80)五个年龄段，并分别用类别变量12345表示
df['AgeBand'] = pd.cut(df['Age'],[0,5,15,30,50,80],labels = ['1','2','3','4','5'])
df.head(3)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	AgeBand
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	3
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	4
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	3
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	4
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	4

```
df.to_csv('test_cut.csv')
```

```
#将连续变量Age按10% 30% 50 70% 90%五个年龄段，并用分类变量12345表示
df['AgeBand'] = pd.qcut(df['Age'],[0,0.1,0.3,0.5,0.7,0.9],labels = ['1','2','3','4','5'])
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	AgeBand
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	2
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	5
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	3
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	4
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	4

```
df.to_csv('test_pr.csv')
```

参考： https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.cut.html

参考： https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.qcut.html

4.5.2 任务二：对文本变量进行转换

(1) 查看文本变量名及种类

(2) 将文本变量Sex， Cabin ， Embarked用数值变量12345表示

(3) 将文本变量Sex， Cabin， Embarked用one-hot编码表示

方法多多益善

```
#查看类别文本变量名及种类

#方法一: value_counts
df['Sex'].value_counts()
```

```
male      577
female    314
Name: Sex, dtype: int64
```

```
df['Cabin'].value_counts()
```

```
G6             4
B96 B98        4
C23 C25 C27    4
F33            3
D              3
               ..
A36            1
C62 C64        1
A20            1
C148           1
C111           1
Name: Cabin, Length: 147, dtype: int64
```

```
df['Embarked'].value_counts()
```

```
S    644
C    168
Q     77
Name: Embarked, dtype: int64
```

```
#方法二: unique
df['Sex'].unique()
```

```
array(['male', 'female'], dtype=object)
```

```
df['Sex'].nunique()
```

```
2
```

```
#将类别文本转换为12345

#方法一: replace
df['Sex_num'] = df['Sex'].replace(['male','female'],[1,2])
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	AgeBand	Sex_num
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	2	1
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	5	2
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	3	2
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	4	2
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	4	1

```
#方法二: map
df['Sex_num'] = df['Sex'].map({'male': 1, 'female': 2})
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	AgeBand	Sex_num
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	2	1
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	5	2
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	3	2
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	4	2
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	4	1

```
#方法三: 使用sklearn.preprocessing的LabelEncoder
from sklearn.preprocessing import LabelEncoder
for feat in ['Cabin', 'Ticket']:
    lbl = LabelEncoder()
    label_dict = dict(zip(df[feat].unique(), range(df[feat].nunique())))
    df[feat + "_labelEncode"] = df[feat].map(label_dict)
    df[feat + "_labelEncode"] = lbl.fit_transform(df[feat].astype(str))

df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	AgeBand	Sex_num	Cabin_labelEncode	Ticket_labelEncode
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	2	1	147	523
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th ...	female	38.0	1	0	PC 17599	71.2833	C85	C	5	2	81	596
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	3	2	147	669
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	4	2	55	49
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	4	1	147	472

```
#将类别文本转换为one-hot编码

#方法一: OneHotEncoder
for feat in ["Age", "Embarked"]:
#     x = pd.get_dummies(df["Age"] // 6)
#     x = pd.get_dummies(pd.cut(df['Age'],5))
    x = pd.get_dummies(df[feat], prefix=feat)
    df = pd.concat([df, x], axis=1)
    #df[feat] = pd.get_dummies(df[feat], prefix=feat)

df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	...	Age_65.0	Age_66.0	Age_70.0	Age_70.5	Age_71.0	Age_74.0	Age_80.0	Embarked_C	Embarked_Q	Embarked_S
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	...	0	0	0	0	0	0	0	0	0	1
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	...	0	0	0	0	0	0	0	1	0	0
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	...	0	0	0	0	0	0	0	0	0	1
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	...	0	0	0	0	0	0	0	0	0	1
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	...	0	0	0	0	0	0	0	0	0	1

4.5.3 任务三（附加）：从纯文本Name特征里提取出Titles的特征(所谓的Titles就是Mr,Miss,Mrs等)

```
df['Title'] = df.Name.str.extract('([A-Za-z]+)\.', expand=False)
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	...	Age_66.0	Age_70.0	Age_70.5	Age_71.0	Age_74.0	Age_80.0	Embarked_C	Embarked_Q	Embarked_S	Title
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	...	0	0	0	0	0	0	0	0	1	Mr
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	...	0	0	0	0	0	0	1	0	0	Mrs
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	...	0	0	0	0	0	0	0	0	1	Miss
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	...	0	0	0	0	0	0	0	0	1	Mrs
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	...	0	0	0	0	0	0	0	0	1	Mr

```
# 保存上面的为最终结论
df.to_csv('test_fin.csv')
```

复习：在前面我们已经学习了Pandas基础，第二章我们开始进入数据分析的业务部分，在第二章第一节的内容中，我们学习了**数据的清洗**，这一部分十分重要，只有数据变得相对干净，我们之后对数据的分析才可以更有力。而这一节，我们要做的是数据重构，数据重构依旧属于数据理解（准备）的范围。

```
# 导入基本库
import numpy as np
import pandas as pd

# 载入data文件中的:train-left-up.csv
text = pd.read_csv('/Users/chenandong/Documents/datawhale数据分析每个人题目设计/招募阶段/第二章项目集合/data/train-left-up.csv')
text.head()
```

	PassengerId	Survived	Pclass	Name
0	1	0	3	Braund, Mr. Owen Harris
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...
2	3	1	3	Heikkinen, Miss. Laina
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)
4	5	0	3	Allen, Mr. William Henry

5 第二章(PART 2)：数据重构(上)

5.1 数据的合并

5.1.1 任务一：将data文件夹里面的所有数据都载入，与之前的原始数据相比，观察他们的之间的关系

```
text_left_up = pd.read_csv("data/train-left-up.csv")
text_left_down = pd.read_csv("data/train-left-down.csv")
text_right_up = pd.read_csv("data/train-right-up.csv")
text_right_down = pd.read_csv("data/train-right-down.csv")

text_left_up.head()
```

	PassengerId	Survived	Pclass	Name
0	1	0	3	Braund, Mr. Owen Harris
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...
2	3	1	3	Heikkinen, Miss. Laina
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)
4	5	0	3	Allen, Mr. William Henry

```
text_left_down.head()
```

	PassengerId	Survived	Pclass	Name
0	440	0	2	Kvillner, Mr. Johan Henrik Johannesson
1	441	1	2	Hart, Mrs. Benjamin (Esther Ada Bloomfield)
2	442	0	3	Hampe, Mr. Leon
3	443	0	3	Petterson, Mr. Johan Emil
4	444	1	2	Reynaldo, Ms. Encarnacion

```
text_right_down.head()
```

	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	male	31.0	0	0	C.A. 18723	10.500	NaN	S
1	female	45.0	1	1	F.C.C. 13529	26.250	NaN	S
2	male	20.0	0	0	345769	9.500	NaN	S
3	male	25.0	1	0	347076	7.775	NaN	S
4	female	28.0	0	0	230434	13.000	NaN	S

```
text_right_up.head()
```

	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	male	22.0	1.0	0.0	A/5 21171	7.2500	NaN	S
1	female	38.0	1.0	0.0	PC 17599	71.2833	C85	C
2	female	26.0	0.0	0.0	STON/O2. 3101282	7.9250	NaN	S
3	female	35.0	1.0	0.0	113803	53.1000	C123	S
4	male	35.0	0.0	0.0	373450	8.0500	NaN	S

提示：结合之前我们加载的train.csv数据，大致预测一下上面的数据是什么

5.1.2 任务二：使用concat方法：将数据train-left-up.csv和train-right-up.csv横向合并为一张表，并保存这张表为result_up

```
list_up = [text_left_up,text_right_up]
result_up = pd.concat(list_up,axis=1)
result_up.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1.0	0.0	3.0	Braund, Mr. Owen Harris	male	22.0	1.0	0.0	A/5 21171	7.2500	NaN	S
1	2.0	1.0	1.0	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1.0	0.0	PC 17599	71.2833	C85	C
2	3.0	1.0	3.0	Heikkinen, Miss. Laina	female	26.0	0.0	0.0	STON/O2. 3101282	7.9250	NaN	S
3	4.0	1.0	1.0	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1.0	0.0	113803	53.1000	C123	S
4	5.0	0.0	3.0	Allen, Mr. William Henry	male	35.0	0.0	0.0	373450	8.0500	NaN	S

5.1.3 任务三：使用concat方法：将train-left-down和train-right-down横向合并为一张表，并保存这张表为result_down。然后将上边的result_up和result_down纵向合并为result。

```
list_down=[text_left_down,text_right_down]
result_down = pd.concat(list_down,axis=1)
result = pd.concat([result_up,result_down])
result.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1.0	0.0	3.0	Braund, Mr. Owen Harris	male	22.0	1.0	0.0	A/5 21171	7.2500	NaN	S
1	2.0	1.0	1.0	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1.0	0.0	PC 17599	71.2833	C85	C
2	3.0	1.0	3.0	Heikkinen, Miss. Laina	female	26.0	0.0	0.0	STON/O2. 3101282	7.9250	NaN	S
3	4.0	1.0	1.0	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1.0	0.0	113803	53.1000	C123	S
4	5.0	0.0	3.0	Allen, Mr. William Henry	male	35.0	0.0	0.0	373450	8.0500	NaN	S

5.1.4 任务四：使用DataFrame自带的方法join方法和append：完成任务二和任务三的任务

```
resul_up = text_left_up.join(text_right_up)
result_down = text_left_down.join(text_right_down)
result = result_up.append(result_down)
result.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1.0	0.0	3.0	Braund, Mr. Owen Harris	male	22.0	1.0	0.0	A/5 21171	7.2500	NaN	S
1	2.0	1.0	1.0	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1.0	0.0	PC 17599	71.2833	C85	C
2	3.0	1.0	3.0	Heikkinen, Miss. Laina	female	26.0	0.0	0.0	STON/O2. 3101282	7.9250	NaN	S
3	4.0	1.0	1.0	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1.0	0.0	113803	53.1000	C123	S
4	5.0	0.0	3.0	Allen, Mr. William Henry	male	35.0	0.0	0.0	373450	8.0500	NaN	S

5.1.5 任务五：使用Panads的merge方法和DataFrame的append方法：完成任务二和任务三的任务

```
result_up = pd.merge(text_left_up,text_right_up,left_index=True,right_index=True)
result_down = pd.merge(text_left_down,text_right_down,left_index=True,right_index=True)
result = resul_up.append(result_down)
result.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1.0	0.0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1.0	0.0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0.0	0.0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1.0	0.0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0.0	0.0	373450	8.0500	NaN	S

思考：对比merge、join以及concat的方法的不同以及相同。思考一下在任务四和任务五的情况下，为什么都要求使用DataFrame的append方法，如何只要求使用merge或者join可不可以完成任务四和任务五呢？

5.1.6 任务六：完成的数据保存为result.csv

```
result.to_csv('result.csv')
```

5.2 换一种角度看数据

5.2.1 任务一：将我们的数据变为Series类型的数据

这个stack函数是干什么的？

```
# 将完整的数据加载出来
text = pd.read_csv('result.csv')
text.head()
# 代码写在这里
unit_result=text.stack().head(20)
unit_result.head()

```

```
0  Unnamed: 0                          0
   PassengerId                         1
   Survived                            0
   Pclass                              3
   Name           Braund, Mr. Owen Harris
dtype: object
```

```
#将代码保存为unit_result,csv
unit_result.to_csv('unit_result.csv')
```

```
test = pd.read_csv('unit_result.csv')

test.head()
```

	0	Unnamed: 0	0.1
0	0	PassengerId	1
1	0	Survived	0
2	0	Pclass	3
3	0	Name	Braund, Mr. Owen Harris
4	0	Sex	male

6 第二章(PART 3)：数据重构(下)

```
# 导入基本库
import numpy as np
import pandas as pd

# 载入data文件中的:result.csv
text = pd.read_csv('result.csv')
text.head()
```

	Unnamed: 0	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1.0	0.0	A/5 21171	7.2500	NaN	S
1	1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1.0	0.0	PC 17599	71.2833	C85	C
2	2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0.0	0.0	STON/O2. 3101282	7.9250	NaN	S
3	3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1.0	0.0	113803	53.1000	C123	S
4	4	5	0	3	Allen, Mr. William Henry	male	35.0	0.0	0.0	373450	8.0500	NaN	S

6.1 数据聚合与运算

6.1.1 任务一：通过《Python for Data Analysis》P303、Google or Baidu来学习了解GroupBy机制

```
#写入心得

```

在了解GroupBy机制之后，运用这个机制完成一系列的操作，来达到我们的目的。

下面通过几个任务来熟悉GroupBy机制。

6.1.2 任务二：计算泰坦尼克号男性与女性的平均票价

```
df  = text['Fare'].groupby(text['Sex'])
means = df.mean()
means
```

```
Sex
female    44.479818
male      25.523893
Name: Fare, dtype: float64
```

6.1.3 任务三：统计泰坦尼克号中男女的存活人数

```
survived_sex = text['Survived'].groupby(text['Sex']).sum()
survived_sex.head()
```

```
Sex
female    233
male      109
Name: Survived, dtype: int64
```

6.1.4 任务四：计算客舱不同等级的存活人数

```
survived_pclass = text['Survived'].groupby(text['Pclass'])
survived_pclass.sum()
```

```
Pclass
1    136
2     87
3    119
Name: Survived, dtype: int64
```

提示：表中的存活那一栏，可以发现如果还活着记为1，死亡记为0

思考：从数据分析的角度，上面的统计结果可以得出那些结论

```
#思考心得


```

6.1.5 任务五：统计在不同等级的票中的不同年龄的船票花费的平均值

```
text.groupby(['Pclass','Age'])['Fare'].mean().head()
```

```
Pclass  Age
1       0.92     151.5500
        2.00     151.5500
        4.00      81.8583
        11.00    120.0000
        14.00    120.0000
Name: Fare, dtype: float64
```

6.1.6 任务六：将任务二和任务三的数据合并，并保存到sex_fare_survived.csv

```
result = pd.merge(means,survived_sex,on='Sex')
result
```

	Fare	Survived
Sex		
female	44.479818	233
male	25.523893	109

```
result.to_csv('sex_fare_survived.csv')
```

6.1.7 任务七：得出不同年龄的总的存活人数，然后找出存活人数的最高的年龄，最后计算存活人数最高的存活率（存活人数/总人数）

```
#不同年龄的存活人数
survived_age = text['Survived'].groupby(text['Age']).sum()
survived_age.head()
```

```
Age
0.42    1
0.67    1
0.75    2
0.83    2
0.92    1
Name: Survived, dtype: int64
```

```
#找出最大值的年龄段
survived_age[survived_age.values==survived_age.max()]
```

```
Age
24.0    15
Name: Survived, dtype: int64
```

```
_sum = text['Survived'].sum()
print(_sum)
```

```
342
```

```
#首先计算总人数
_sum = text['Survived'].sum()

print("sum of person:"+str(_sum))

precetn =survived_age.max()/_sum

print("最大存活率："+str(precetn))
```

```
sum of person:342
最大存活率：0.043859649122807015
```

复习：回顾学习完第一章，我们对泰坦尼克号数据有了基本的了解，也学到了一些基本的统计方法，第二章中我们学习了数据的清理和重构，使得数据更加的易于理解；今天我们要学习的是第二章第三节：**数据可视化**，主要给大家介绍一下Python数据可视化库Matplotlib，在本章学习中，你也许会觉得数据很有趣。在打比赛的过程中，数据可视化可以让我们更好的看到每一个关键步骤的结果如何，可以用来优化方案，是一个很有用的技巧。

7 第二章(PART 4)：数据可视化

7.1 开始之前，导入numpy、pandas包和数据

```
# 加载所需的库
# 如果出现 ModuleNotFoundError: No module named 'xxxx'
# 你只需要在终端/cmd下 pip install xxxx 即可
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# 导入result.csv这个文件
text = pd.read_csv(r'result.csv')
text.head()
```

	Unnamed: 0	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1.0	0.0	A/5 21171	7.2500	NaN	S
1	1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1.0	0.0	PC 17599	71.2833	C85	C
2	2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0.0	0.0	STON/O2. 3101282	7.9250	NaN	S
3	3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1.0	0.0	113803	53.1000	C123	S
4	4	5	0	3	Allen, Mr. William Henry	male	35.0	0.0	0.0	373450	8.0500	NaN	S

7.2 如何让人一眼看懂你的数据？

《Python for Data Analysis》第九章

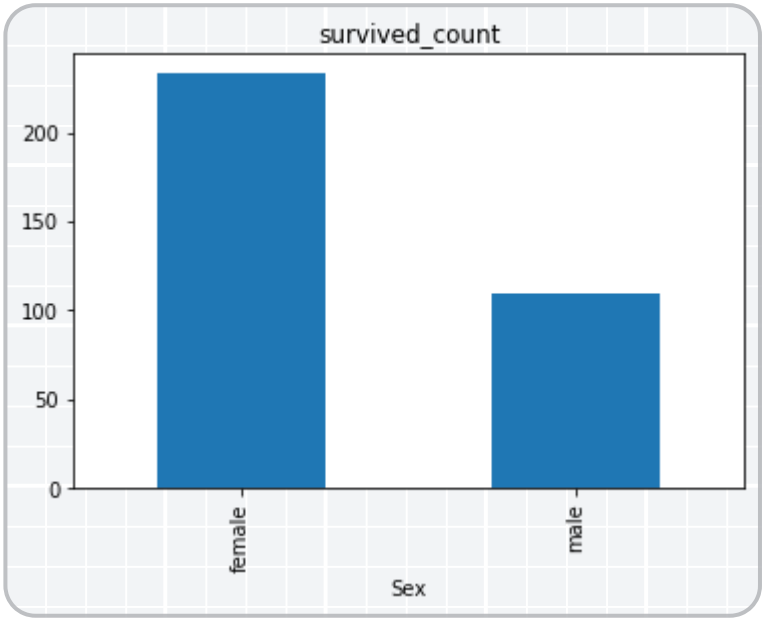
7.2.1 任务一：跟着书本第九章，了解matplotlib，自己创建一个数据项，对其进行基本可视化

思考：最基本的可视化图案有哪些？分别适用于那些场景？（比如折线图适合可视化某个属性值随时间变化的走势）

```
#思考回答
#这一部分需要了解可视化图案的的逻辑，知道什么样的图案可以表达什么样的信号
```

7.2.2 任务二：可视化展示泰坦尼克号数据集中男女中生存人数分布情况（用柱状图试试）。

```
sex = text.groupby('Sex')['Survived'].sum()
sex.plot.bar()
plt.title('survived_count')
plt.show()
```

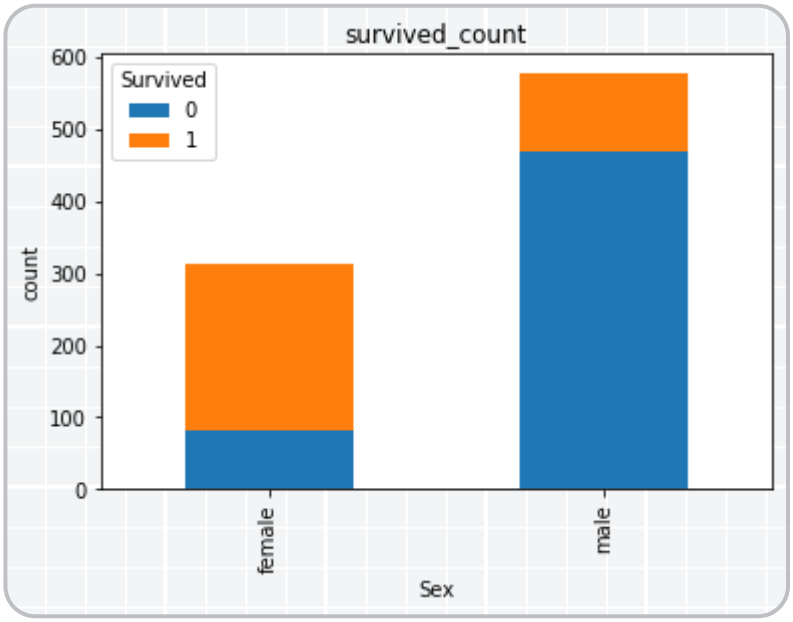


思考：计算出泰坦尼克号数据集中男女中死亡人数，并可视化展示？如何和男女生存人数可视化柱状图结合到一起？看到你的数据可视化，说说你的第一感受（比如：你一眼看出男生存活人数更多，那么性别可能会影响存活率）。

7.2.3 任务三：可视化展示泰坦尼克号数据集中男女中生存人与死亡人数的比例图（用柱状图试试）。

```
# 提示：计算男女中死亡人数 1表示生存，0表示死亡
text.groupby(['Sex','Survived'])
['Survived'].count().unstack().plot(kind='bar',stacked='True')
plt.title('survived_count')
plt.ylabel('count')
```

```
Text(0, 0.5, 'count')
```



提示：男女这两个数据轴，存活和死亡人数按比例用柱状图表示

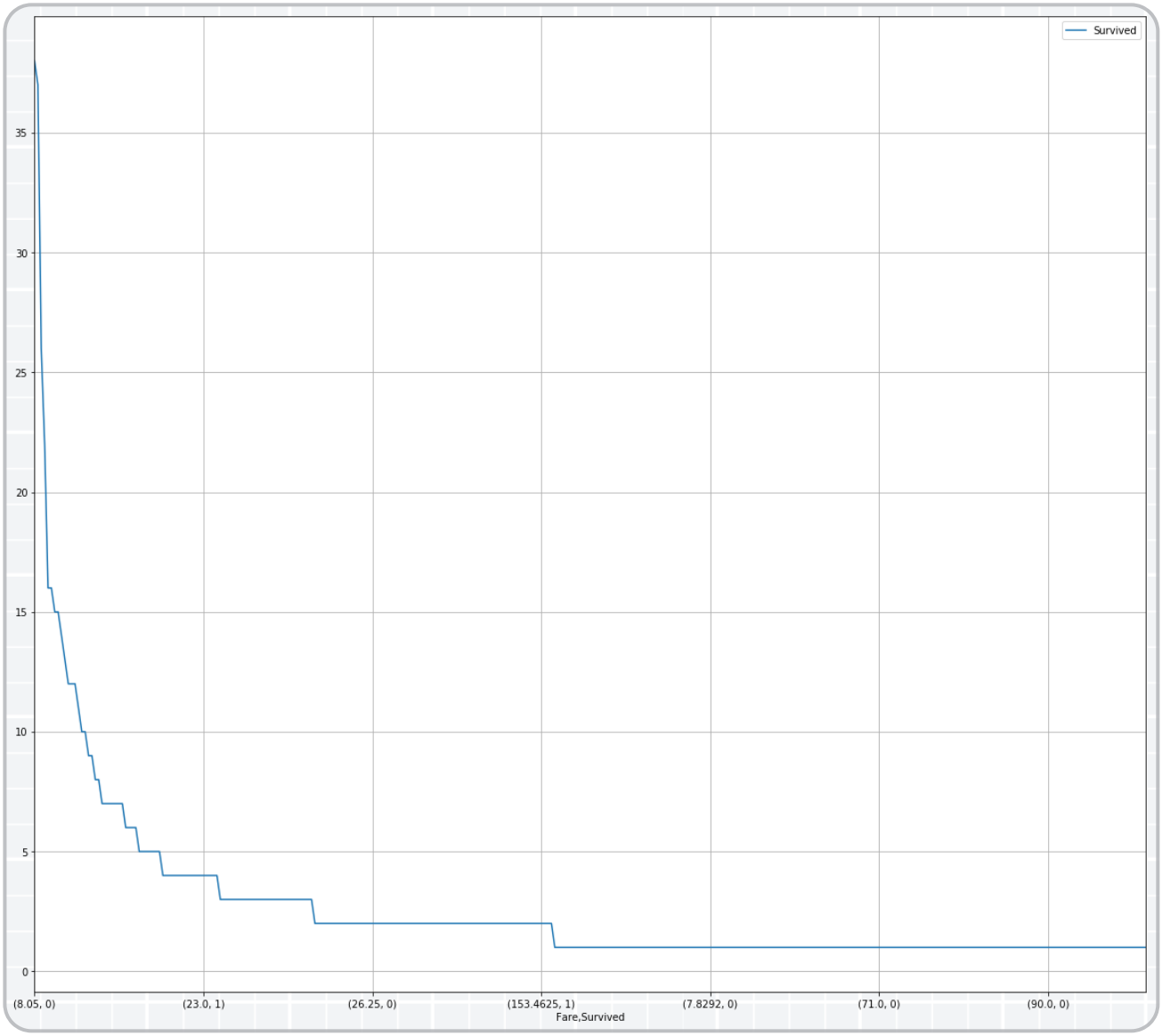
7.2.4 任务四：可视化展示泰坦尼克号数据集中不同票价的人生存和死亡人数分布情况。（用折线图试试）（横轴是不同票价，纵轴是存活人数）

提示：对于这种统计性质的且用折线表示的数据，你可以考虑将数据排序或者不排序来分别表示。看看你能发现什么？

```
# 计算不同票价中生存与死亡人数 1表示生存，0表示死亡
fare_sur = text.groupby(['Fare'])['Survived'].value_counts().sort_values(ascending=False)
fare_sur
```

```
Fare     Survived
8.0500   0           38
7.8958   0           37
13.0000  0           26
7.7500   0           22
26.0000  0           16
                     ..
20.2500  1            1
         0            1
18.7875  1            1
         0            1
15.0500  0            1
Name: Survived, Length: 330, dtype: int64
```

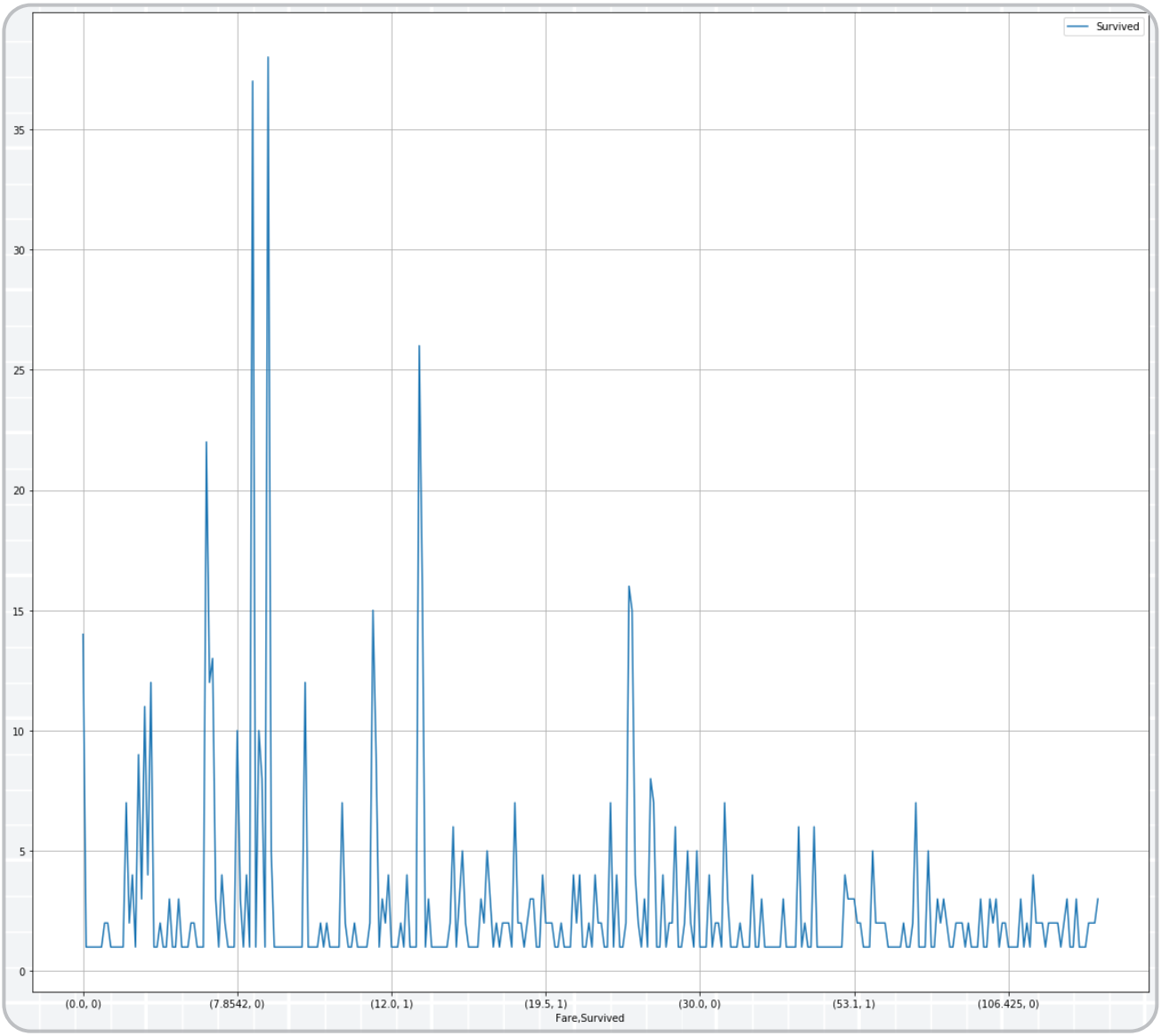
```
# 排序后绘折线图
fig = plt.figure(figsize=(20, 18))
fare_sur.plot(grid=True)
plt.legend()
plt.show()
```




```
# 排序前绘折线图
fare_sur1 = text.groupby(['Fare'])['Survived'].value_counts()
fare_sur1
```

```
Fare      Survived
0.0000    0           14
          1            1
4.0125    0            1
5.0000    0            1
6.2375    0            1
                      ..
247.5208  1            1
262.3750  1            2
263.0000  0            2
          1            2
512.3292  1            3
Name: Survived, Length: 330, dtype: int64
```

```
fig = plt.figure(figsize=(20, 18))
fare_sur1.plot(grid=True)
plt.legend()
plt.show()
```



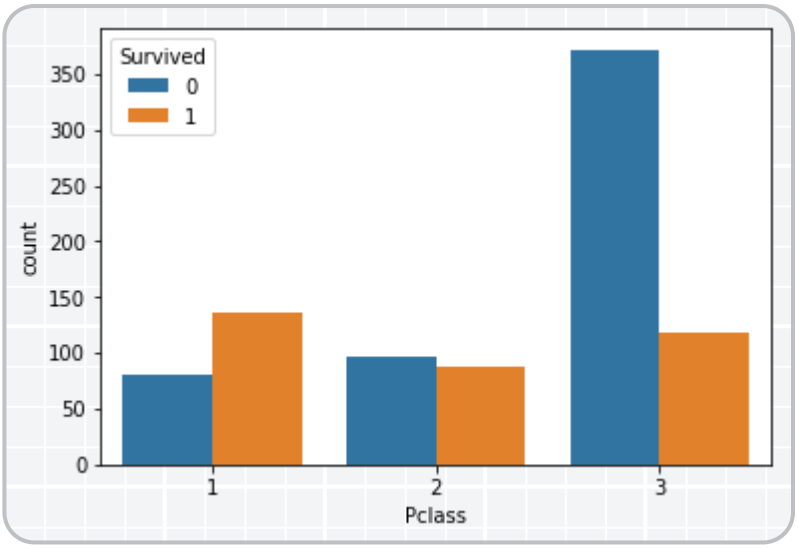
7.2.5 任务五：可视化展示泰坦尼克号数据集中不同仓位等级的人生存和死亡人员的分布情况。（用柱状图试试）

```
# 1表示生存，0表示死亡
pclass_sur = text.groupby(['Pclass'])['Survived'].value_counts()
pclass_sur
```

```
Pclass  Survived
1       1           136
        0            80
2       0            97
        1            87
3       0           372
        1           119
Name: Survived, dtype: int64
```

```
import seaborn as sns
sns.countplot(x="Pclass", hue="Survived", data=text)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1be2b660ac8>
```



思考：看到这个前面几个数据可视化，说说你的第一感受和你的总结

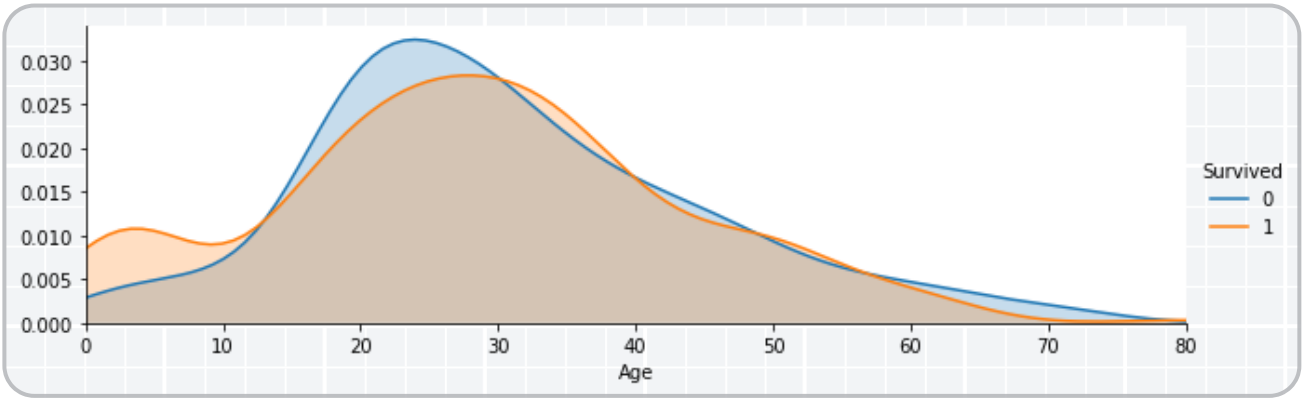
```
#思考题回答


```

7.2.6 任务六：可视化展示泰坦尼克号数据集中不同年龄的人生存与死亡人数分布情况。(不限表达方式)

```
facet = sns.FacetGrid(text, hue="Survived",aspect=3)
facet.map(sns.kdeplot,'Age',shade= True)
facet.set(xlim=(0, text['Age'].max()))
facet.add_legend()
```

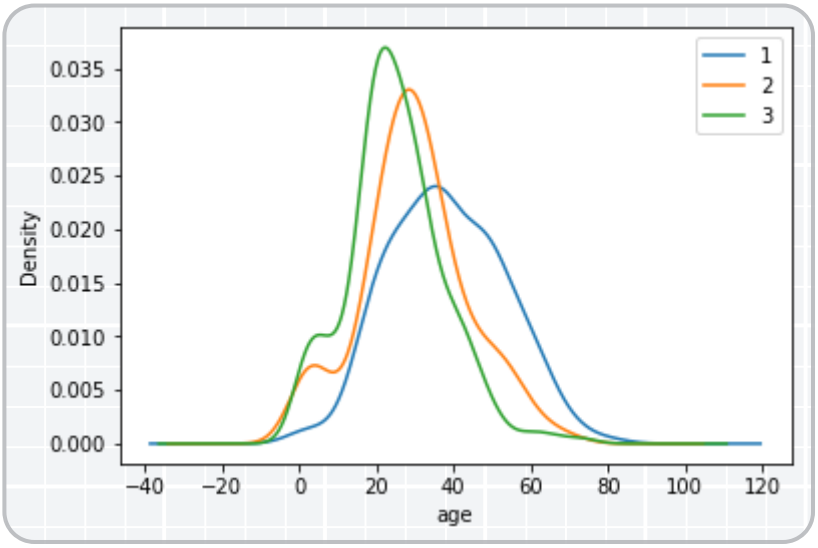
```
<seaborn.axisgrid.FacetGrid at 0x1be2bf11048>
```



7.2.7 任务七：可视化展示泰坦尼克号数据集中不同仓位等级的人年龄分布情况。（用折线图试试）

```
text.Age[text.Pclass == 1].plot(kind='kde')
text.Age[text.Pclass == 2].plot(kind='kde')
text.Age[text.Pclass == 3].plot(kind='kde')
plt.xlabel("age")
plt.legend((1,2,3),loc="best")
```

```
<matplotlib.legend.Legend at 0x1be2e921940>
```



思考：上面所有可视化的例子做一个总体的分析，你看看你能不能有自己发现

```
#思考题回答


```

总结：到这里，我们的可视化就告一段落啦，如果你对数据可视化极其感兴趣，你还可以了解一下其他可视化模块，如：pyecharts，bokeh等。

如果你在工作中使用数据可视化，你必须知道数据可视化最大的作用不是炫酷，而是最快最直观的理解数据要表达什么，你觉得呢？

8 第三章(PART 1)：模型搭建

经过前面的探索性数据分析我们可以很清楚的了解到数据集的情况，以及得出了一些结论。

下面我们将搭建一个预测模型，运用机器学习的方式来为泰坦尼克船只做一个预测，我们在测试集的数据中来预测哪些乘客将会存活，哪些乘客将遭遇不幸。然后我们会对我们的模型做一个评价。

这一章的内容可以学习到数据建模以及模型评价的知识，算是进阶的内容，为之后的数据分析课程打下基础。

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from IPython.display import Image
```

了解上面的库的作用

```
%matplotlib inline
```

```
plt.rcParams['font.sans-serif'] = ['SimHei']  # 用来正常显示中文标签
plt.rcParams['axes.unicode_minus'] = False  # 用来正常显示负号
plt.rcParams['figure.figsize'] = (10, 6)  # 设置输出图片大小
```

```
# 读取训练数集
train = pd.read_csv('train.csv')
train.shape
```

```
(891, 12)
```

```
train.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

8.1 特征工程

这一部分是对之前的学习内容的简单回顾

8.1.1 任务一：缺失值填充

1. 对分类变量缺失值：填充某个缺失值字符(NA)、用最多类别的进行填充
2. 对连续变量缺失值：填充均值、中位数、众数

```
# 对分类变量进行填充
train['Cabin'] = train['Cabin'].fillna('NA')
train['Embarked'] = train['Embarked'].fillna('S')

# 对连续变量进行填充
train['Age'] = train['Age'].fillna(train['Age'].mean())

# 检查缺失值比例
train.isnull().mean().sort_values(ascending=False)
```

```
Embarked       0.0
Cabin          0.0
Fare           0.0
Ticket         0.0
Parch          0.0
SibSp          0.0
Age            0.0
Sex            0.0
Name           0.0
Pclass         0.0
Survived       0.0
PassengerId    0.0
dtype: float64
```

8.1.2 任务二：编码分类变量

```
# 取出所有的输入特征
data = train[['Pclass','Sex','Age','SibSp','Parch','Fare', 'Embarked']]

# 进行虚拟变量转换
data = pd.get_dummies(data)

data.head()
```

	Pclass	Age	SibSp	Parch	Fare	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S
0	3	22.0	1	0	7.2500	0	1	0	0	1
1	1	38.0	1	0	71.2833	1	0	1	0	0
2	3	26.0	0	0	7.9250	1	0	0	0	1
3	1	35.0	1	0	53.1000	1	0	0	0	1
4	3	35.0	0	0	8.0500	0	1	0	0	1

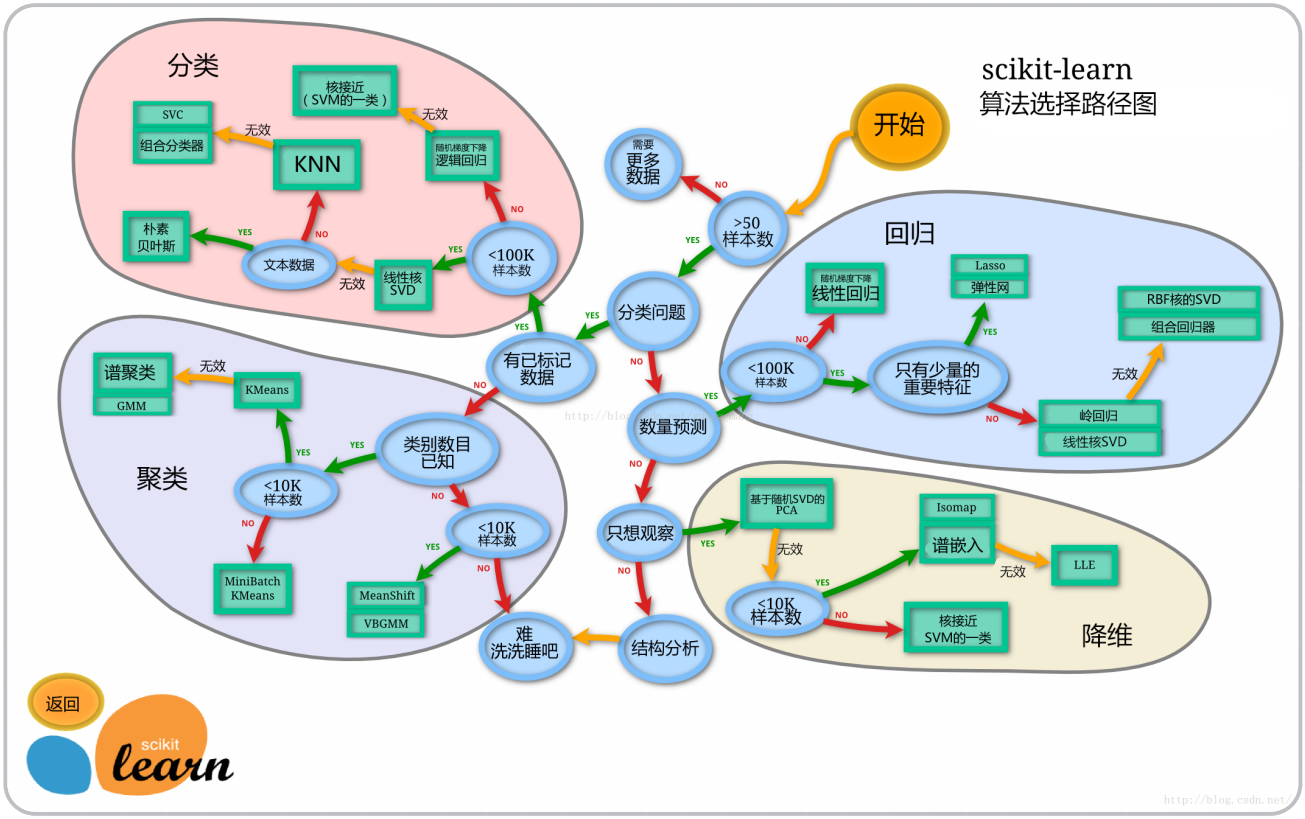
8.2 模型搭建

1. 处理完前面的数据我们就得到建模数据，下一步是选择合适模型
2. 在进行模型选择之前我们需要先知道数据集最终是进行**监督学习**还是**无监督学习**
3. 除了根据我们任务来选择模型外，还可以根据数据样本量以及特征的稀疏性来决定
4. 刚开始我们总是先尝试使用一个基本的模型来作为其baseline，进而再训练其他模型做对比，最终选择泛化能力或性能比较好的模型

思考

1. 数据集哪些差异会导致模型在拟合数据是发生变化

```
# sklearn模型算法选择路径图
Image('20170624105439491.png')
```



8.2.1 任务一：切割训练集和测试集

1. 按比例切割训练集和测试集(一般测试集的比例有30%、25%、20%、15%和10%)
2. 按目标变量分层进行等比切割
3. 设置随机种子以便结果能复现

提示

1. 切割数据集是为了后续能评估模型泛化能力
2. sklearn中切割数据集的方法为`train_test_split`
3. 查看函数文档可以在jupyter noteboo里面使用`train_test_split?`后回车即可看到
4. 分层和随机种子在参数里寻找

思考

1. 什么情况下切割数据集的时候不用进行随机选取

```
from sklearn.model_selection import train_test_split

# 一般先取出X和y后再切割，有些情况会使用到未切割的，这时候X和y就可以用
X = data
y = train['Survived']

# 对数据集进行切割
X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y, random_state=0)

# 查看数据形状
X_train.shape, X_test.shape
```

```
((668, 10), (223, 10))
```

8.2.2 任务二：模型创建

1. 创建基于线性模型的分类模型（逻辑回归）
2. 创建基于树的分类模型（决策树、随机森林）
3. 查看模型的参数，并更改参数值，观察模型变化

提示

1. 逻辑回归不是回归模型而是分类模型，不要与`LinearRegression`混淆
2. 随机森林其实是决策树集成为了降低决策树过拟合的情况

3. 线性模型所在的模块为`sklearn.linear_model`
4. 树模型所在的模块为`sklearn.ensemble`

思考

1. 为什么线性模型可以进行分类任务，背后是怎么的数学关系
2. 对于多分类问题，线性模型是怎么进行分类的

```
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
```

```
# 默认参数逻辑回归模型
lr = LogisticRegression()
lr.fit(X_train, y_train)
```

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

```
# 查看训练集和测试集score值
print("Training set score: {:.2f}".format(lr.score(X_train, y_train)))
print("Testing set score: {:.2f}".format(lr.score(X_test, y_test)))
```

```
Training set score: 0.80
Testing set score: 0.78
```

```
# 调整参数后的逻辑回归模型
lr2 = LogisticRegression(C=100)
lr2.fit(X_train, y_train)
```

```
LogisticRegression(C=100, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

```
print("Training set score: {:.2f}".format(lr2.score(X_train, y_train)))
print("Testing set score: {:.2f}".format(lr2.score(X_test, y_test)))
```

```
Training set score: 0.80
Testing set score: 0.79
```

```
# 默认参数的随机森林分类模型
rfc = RandomForestClassifier()
rfc.fit(X_train, y_train)
```

```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=1,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
```

```
print("Training set score: {:.2f}".format(rfc.score(X_train, y_train)))
print("Testing set score: {:.2f}".format(rfc.score(X_test, y_test)))
```

```
Training set score: 0.97
Testing set score: 0.82
```

```
# 调整参数后的随机森林分类模型
rfc2 = RandomForestClassifier(n_estimators=100, max_depth=5)
rfc2.fit(X_train, y_train)
```

```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=5, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=100, n_jobs=1,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
```

```
print("Training set score: {:.2f}".format(rfc2.score(X_train, y_train)))
print("Testing set score: {:.2f}".format(rfc2.score(X_test, y_test)))
```

```
Training set score: 0.86
Testing set score: 0.83
```

8.2.3 任务三：输出模型预测结果

1. 输出模型预测分类标签
2. 输出不同分类标签的预测概率

提示

1. 一般监督模型在sklearn里面有个`predict`能输出预测标签，`predict_proba`则可以输出标签概率

思考

1. 预测标签的概率对我们有什么帮助

```
# 预测标签
pred = lr.predict(X_train)

# 此时我们可以看到0和1的数组
pred[:10]
```

```
array([0, 1, 1, 1, 0, 0, 1, 0, 1, 1], dtype=int64)
```

```
# 预测标签概率
pred_proba = lr.predict_proba(X_train)
```

```
pred_proba[:10]
```

```
array([[0.62887291, 0.37112709],
       [0.14897206, 0.85102794],
       [0.47162003, 0.52837997],
       [0.20365672, 0.79634328],
       [0.86428125, 0.13571875],
       [0.9033887 , 0.0966113 ],
       [0.13829338, 0.86170662],
       [0.89516141, 0.10483859],
       [0.05735141, 0.94264859],
       [0.13593291, 0.86406709]])
```

9 第三章(PART 2)：模型评估

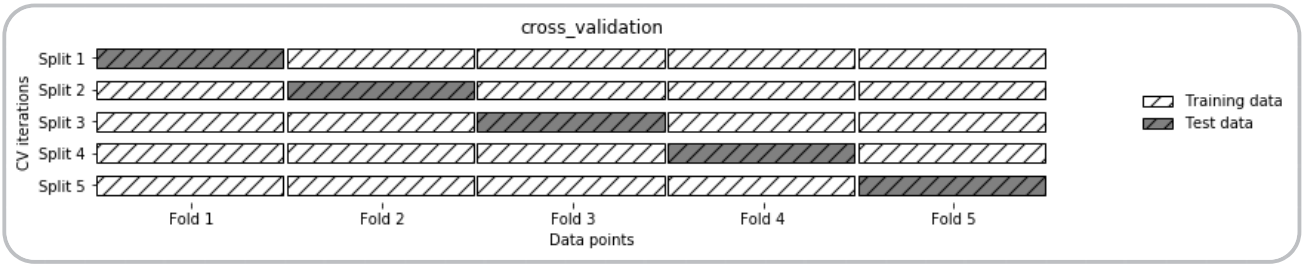
9.1 模型评估

1. 模型评估是为了知道模型的泛化能力。
2. 交叉验证（cross-validation）是一种评估泛化性能的统计学方法，它比单次划分训练集和测试集的方法更加稳定、全面。
3. 在交叉验证中，数据被多次划分，并且需要训练多个模型。
4. 最常用的交叉验证是 k 折交叉验证（k-fold cross-validation），其中 k 是由用户指定的数字，通常取 5 或 10。
5. 准确率（precision）度量的是被预测为正例的样本中有多少是真正的正例
6. 召回率（recall）度量的是正类样本中有多少被预测为正类
7. f-分数是准确率与召回率的调和平均

9.1.1 任务一：交叉验证

1. 用10折交叉验证来评估逻辑回归模型
2. 计算交叉验证精度的平均值

```
1  Image('Snipaste_2020-01-05_16-37-56.png')
```



提示

1. 交叉验证在sklearn中的模块为`sklearn.model_selection`

思考

1. k折越多的情况下会带来什么样的影响？

```
from sklearn.model_selection import cross_val_score
```

```
lr = LogisticRegression(C=100)
scores = cross_val_score(lr, X_train, y_train, cv=10)
```

```
# k折交叉验证分数
scores
```

```
array([0.82352941, 0.79411765, 0.80597015, 0.80597015, 0.8358209 ,
       0.88059701, 0.72727273, 0.86363636, 0.75757576, 0.71212121])
```

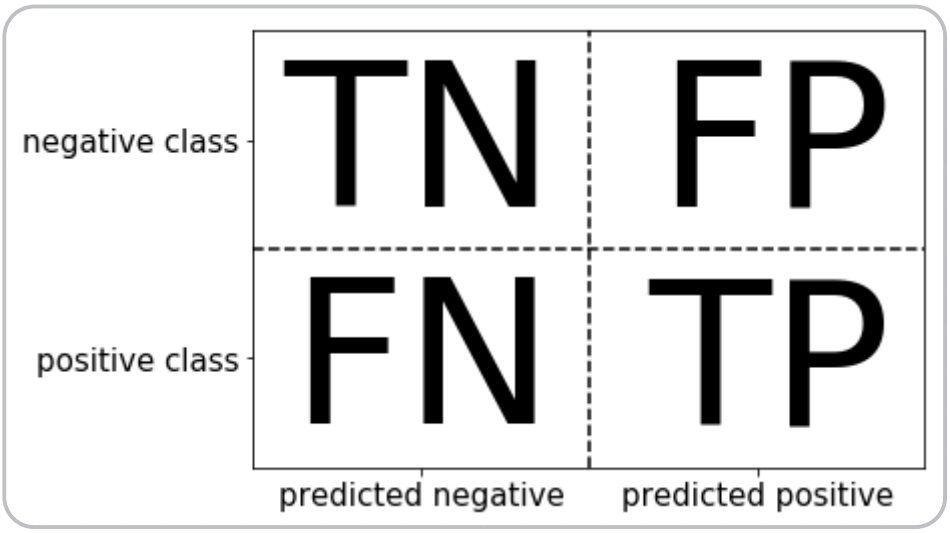
```
# 平均交叉验证分数
print("Average cross-validation score: {:.2f}".format(scores.mean()))
```

```
Average cross-validation score: 0.80
```

9.1.2 任务二：混淆矩阵

1. 计算二分类问题的混淆矩阵
2. 计算精确率、召回率以及f-分数

```
Image('Snipaste_2020-01-05_16-38-26.png')
```



```
Image('Snipaste_2020-01-05_16-39-27.png')
```

Relation to accuracy

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

Precision, recall and f-score

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{F} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

提示

1. 混淆矩阵的方法在sklearn中的 `sklearn.metrics` 模块
2. 混淆矩阵需要输入真实标签和预测标签

思考

1. 如果自己实现混淆矩阵的时候该注意什么问题

```
from sklearn.metrics import confusion_matrix
```

```
# 训练模型
lr = LogisticRegression(C=100)
lr.fit(X_train, y_train)
```

```
LogisticRegression(C=100, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

```
# 模型预测结果
pred = lr.predict(X_train)

# 混淆矩阵
confusion_matrix(y_train, pred)
```

```
array([[350,  62],
       [ 71, 185]], dtype=int64)
```

```
from sklearn.metrics import classification_report

# 精确率、召回率以及f1-score
print(classification_report(y_train, pred))
```

```
             precision    recall  f1-score   support

          0       0.83      0.85      0.84       412
          1       0.75      0.72      0.74       256

avg / total       0.80      0.80      0.80       668
```

9.1.3 任务三：ROC曲线

1. 绘制ROC曲线

提示

1. ROC曲线在sklearn中的模块为`sklearn.metrics`
2. ROC曲线下面所包围的面积越大越好

思考

1. 对于多分类问题如何绘制ROC曲线

```
from sklearn.metrics import roc_curve
```

```
fpr, tpr, thresholds = roc_curve(y_test, lr.decision_function(X_test))
plt.plot(fpr, tpr, label="ROC Curve")
plt.xlabel("FPR")
plt.ylabel("TPR (recall)")
# 找到最接近于0的阈值
close_zero = np.argmin(np.abs(thresholds))
plt.plot(fpr[close_zero], tpr[close_zero], 'o', markersize=10, label="threshold zero",
fillstyle="none", c='k', mew=2)
plt.legend(loc=4)
```

```
<matplotlib.legend.Legend at 0x2e4ea25db00>
```