

数据可视化

讲师名称：李川

Can//ay 嘉为数字咨询

数 字 化 人 才 培 养 先 行 者



目录

1

**Seaborn
简介**

2

关系图

3

分布图

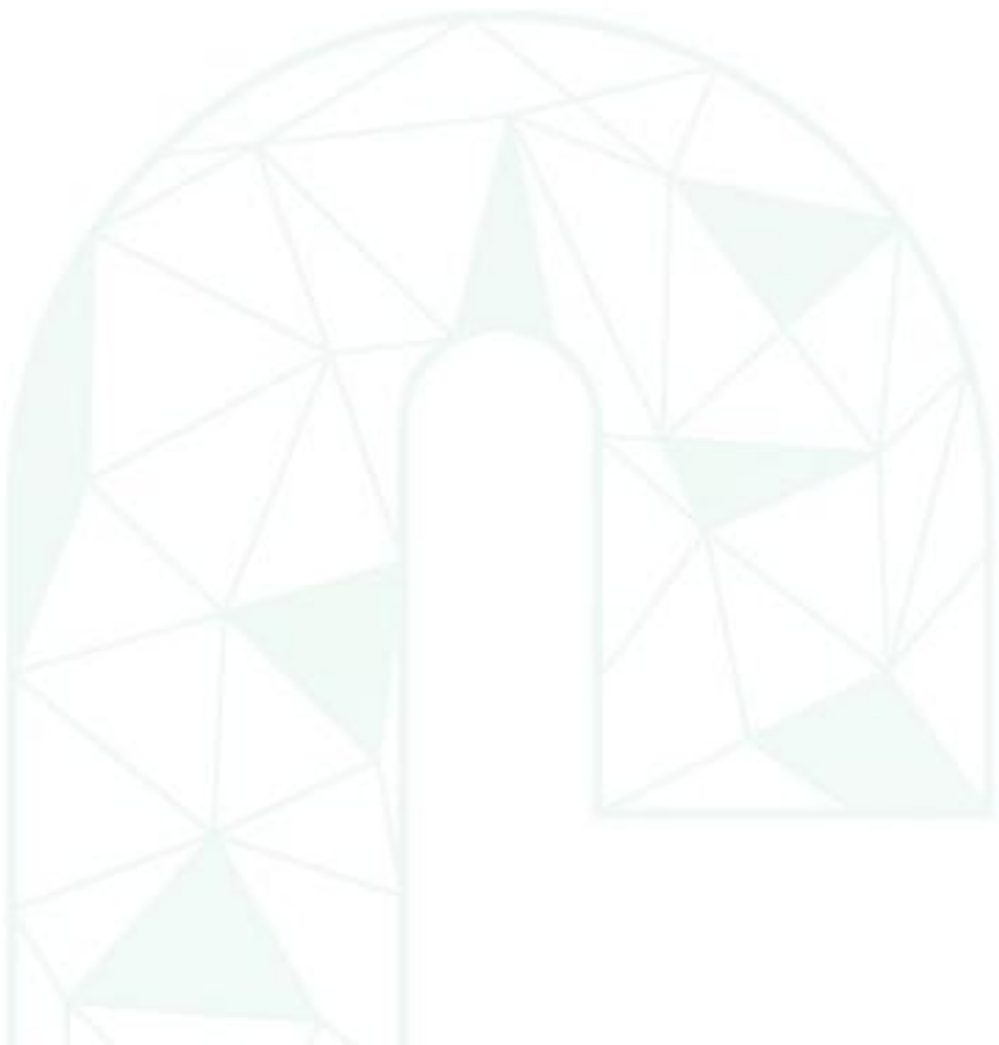
4

分类图

5

通用设置

CONTENTS



/01

Seaborn简介

1 Seaborn简介

Seaborn是基于matplotlib的图形可视化python包。它提供了一种高度交互式界面，便于用户能够做出各种有吸引力的统计图表。

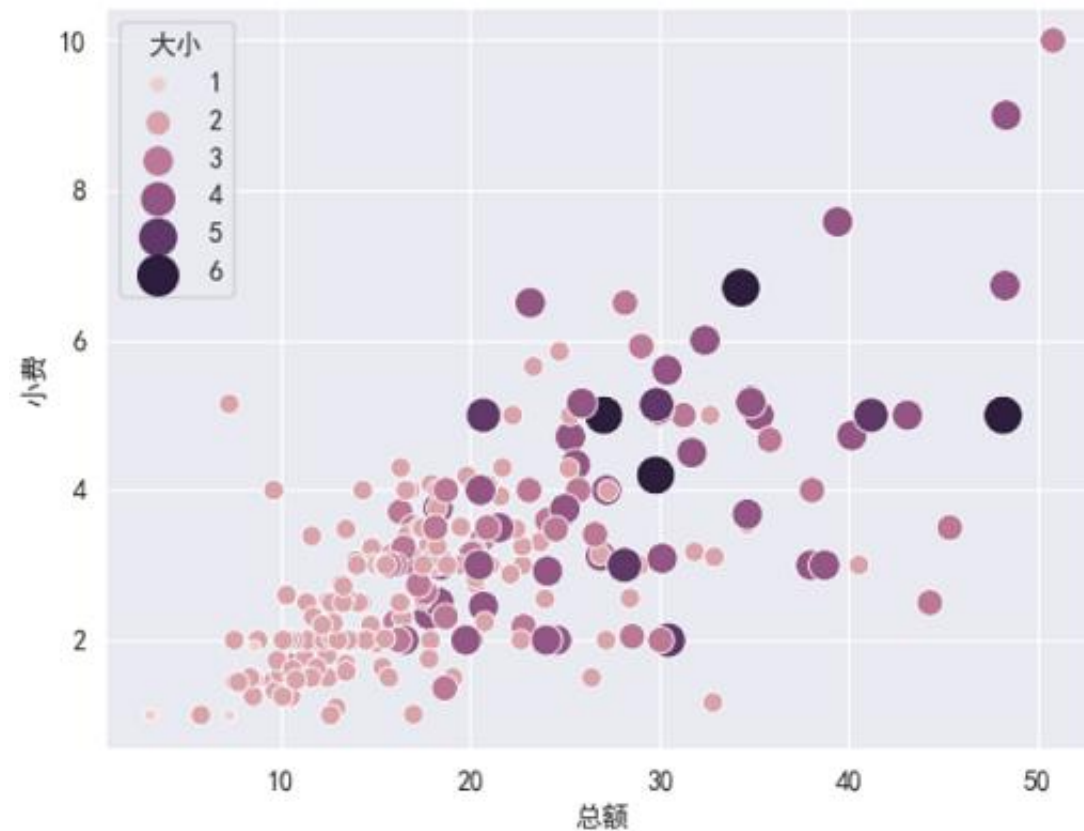
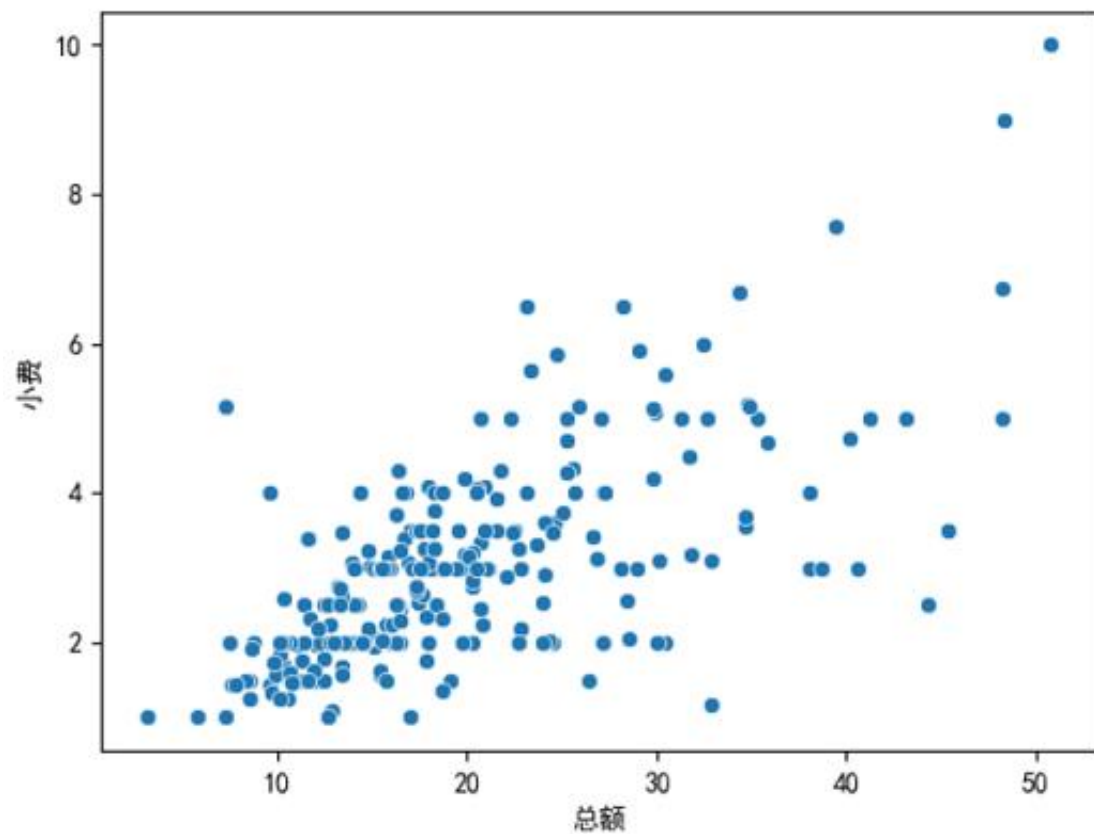
Seaborn是在matplotlib的基础上进行了更高级的API封装，从而使得作图更加容易，在大多数情况下使用seaborn能做出很具有吸引力的图，而使用matplotlib就能制作具有更多特色的图。应该把Seaborn视为matplotlib的补充，而不是替代物。

seaborn关注的是统计量之间的关系。Seaborn有2个最重要的优势：①语法更加简单好用；②图表更加高级美观。

安装Seaborn安装

```
pip install seaborn
```

1.1 Seaborn与matplotlib对比



1.2 Seaborn使用注意事项

- 对于Seaborn来说，有以下4点是需要特别清楚的。了解这几点，对于我们后续的学习也非常重要。
 - 在实际工作中，我们应该首选Seaborn来实现数据可视化。如果Seaborn实现不了，再去考虑使用Matplotlib。
 - Seaborn是基于Matplotlib实现的，所以很多方法有相似之处。
 - Seaborn只提供最常用图表的绘制函数，对于其他不常用的图表并没有提供绘制函数。
 - Seaborn是结合Pandas一起使用的，它的数据都是Series或DataFrame，而不能是其他数据类型。

1.2 Seaborn的使用注意事项（续）

- 由于Seaborn是基于Matplotlib实现的，所以在使用Seaborn之前必须引入Matplotlib这个库才行。此外，Seaborn都是结合Pandas一起使用的，所以我们也要引入Pandas库。

- **语法：**

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

1.2 Seaborn的使用注意事项（续）

【常见问题】：使用import导入Seaborn时，为什么简称为sns？这个sns是怎么来的呢？

之所以将Seaborn简称为sns，其实这里是有一个“梗”的。在美剧The West Wing中，有一人物的名字叫做Samuel Norman Seaborn，他名字的首字母就是sns。然后Seaborn官方为了避免与其他库冲突，所以就干脆使用这个比较特殊的“sns”作为Seaborn的简称了。



/02

关系图

2.1 折线图

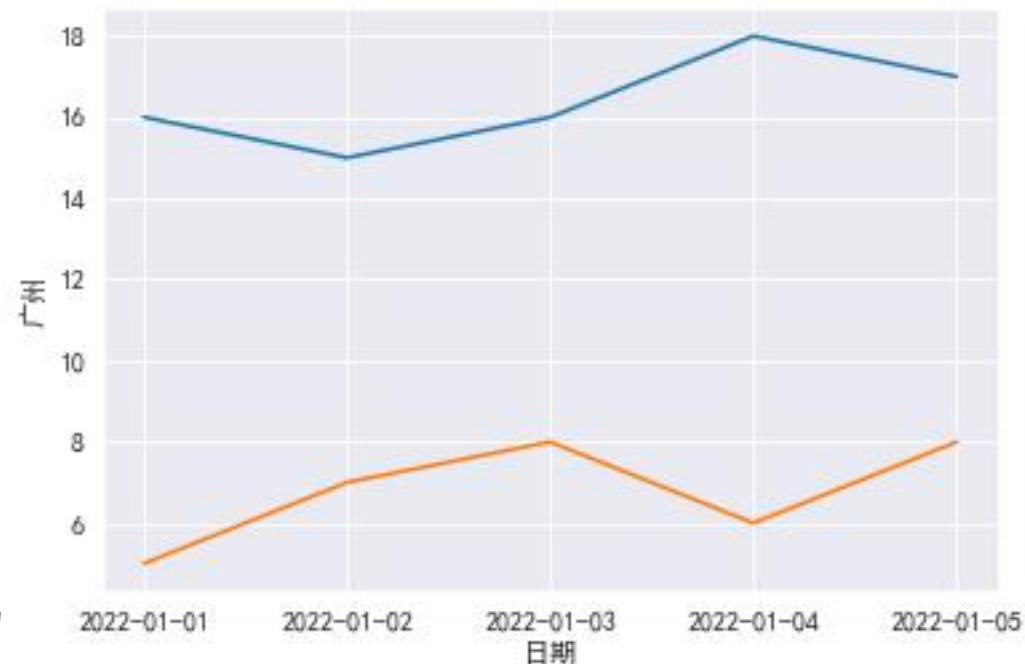
- 在Seaborn中，最基本的也是绘制一个折线图。这一节先来简单介绍如何绘制一个折线图，在后面再去学习如何绘制其他图表。
- 在Seaborn中，我们可以使用lineplot()函数来绘制一个折线图。折线图的主要作用是：观察“因变量y”随着“自变量x”的变化而变化的趋势。
- **语法：**

```
sns.lineplot(data, x, y)
```

2.1.1 绘制折线实例-城市天气情况

#例3-1

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_style("darkgrid")
sns.set_style({"font.sans-serif": "SimHei"})
data = [ ["2022-01-01", 16, 5],  ["2022-01-02", 15, 7],
        ["2022-01-03", 16, 8],  ["2022-01-04", 18, 6],
        ["2022-01-05", 17, 8]] # 气温数据 (单位: 度)
df = pd.DataFrame(data, columns=["日期", "广州", "北京"])
sns.lineplot(data=df, x="日期", y="广州")
sns.lineplot(data=df, x="日期", y="北京")
plt.show()
```



2.1.2 缺省data, x,y

- Seaborn的所有Data必须是Series或Dataframe, 可以不显式指定x和y, 然后自动地绘图。

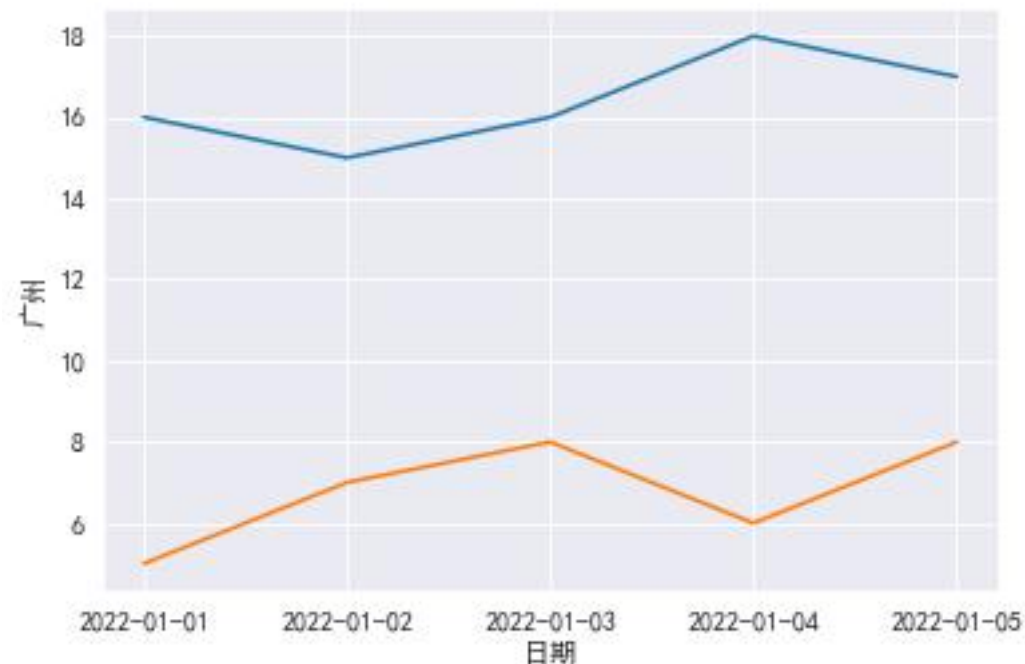
The diagram illustrates a DataFrame structure. A dashed box encloses the entire table. A red dashed box highlights the first column, with a red arrow pointing to the label 'index' below it. Another red dashed box highlights the header row, with a red arrow pointing to the label 'columns' to its right. A third red dashed box highlights the data rows, with a red arrow pointing to the label 'values' below it.

	学号	姓名	性别	年龄	分数
s1	202201	小杰	男	20	650
s2	202202	小红	女	19	645
s3	202203	小明	男	21	590
s4	202204	小华	男	20	640
s5	202205	小莉	女	19	635

2.1.2 缺省data

#例3-2

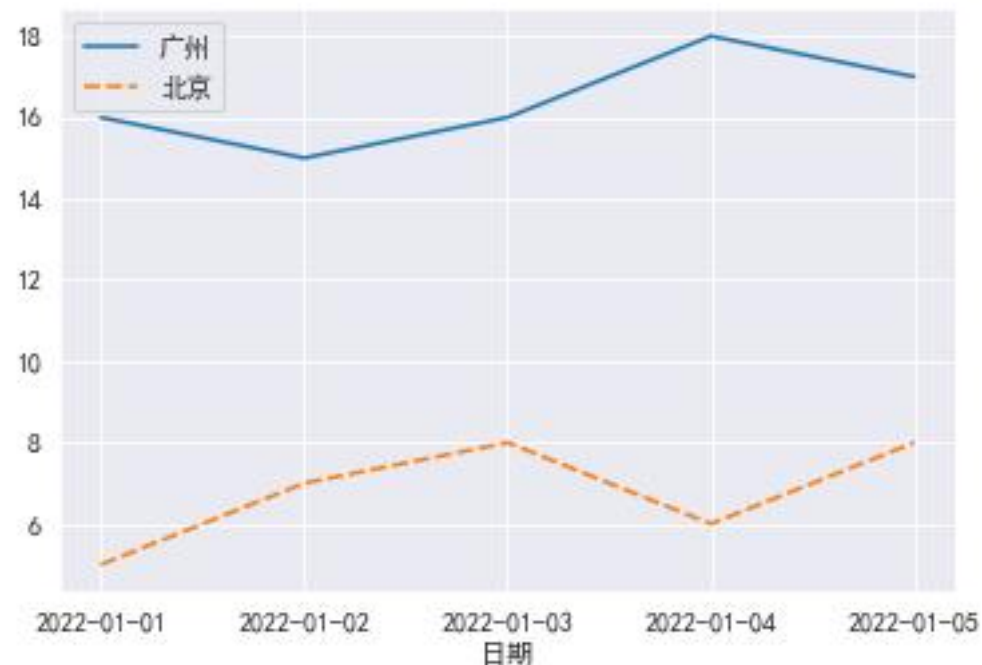
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_style("darkgrid")
sns.set_style({"font.sans-serif": "SimHei"})
data = [ ["2022-01-01", 16, 5],  ["2022-01-02", 15, 7],
        ["2022-01-03", 16, 8],  ["2022-01-04", 18, 6],
        ["2022-01-05", 17, 8]] # 气温数据 (单位: 度)
df = pd.DataFrame(data, columns=["日期", "广州", "北京"])
sns.lineplot(x=df["日期"], y=df["广州"])
sns.lineplot(x=df["日期"], y=df["北京"])
plt.show()
```



2.1.2 缺省x, y

#例3-3

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns# 设置
sns.set_style("darkgrid")
sns.set_style({"font.sans-serif": "SimHei"})# 气温数据
(单位: 度)
data = [ ["2022-01-01", 16, 5],  ["2022-01-02", 15, 7],
        ["2022-01-03", 16, 8],  ["2022-01-04", 18, 6],
        ["2022-01-05", 17, 8]]
df = pd.DataFrame(data, columns=["日期", "广州", "北京"])
df.set_index("日期", inplace=True)
sns.lineplot(data=df)
plt.show()
```



2.1.3 实际案例

- flight.csv文件保存的是某航空公司1949~1960这12年内每个月的乘客人数，部分内容如下图所示。我们尝试绘制其对应的折线图。

```
年份,月份,人数
1949,1月,112
1949,2月,118
1949,3月,132
1949,4月,129
1949,5月,121
1949,6月,135
1949,7月,148
1949,8月,148
1949,9月,136
1949,10月,119
1949,11月,104
1949,12月,118
```

2.1.3 实际案例-1月份乘客数据

#例3-4

```
import pandas as pd
```

```
import matplotlib.pyplot as plt
```

```
import seaborn as sns# 设置
```

```
sns.set_style("darkgrid")
```

```
sns.set_style({"font.sans-serif": "SimHei"})# 读取数据
```

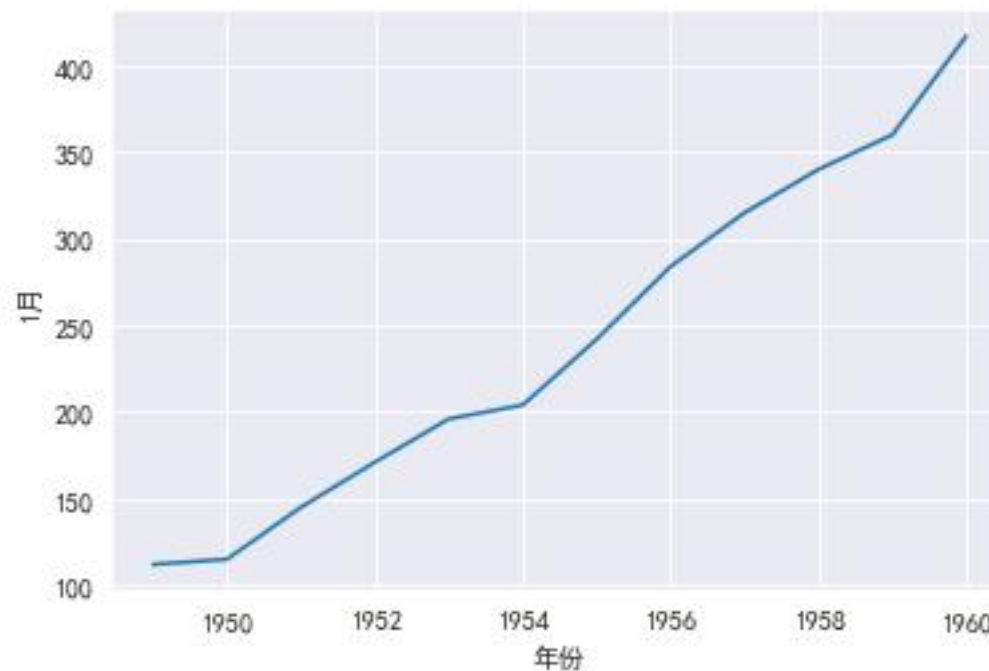
```
df = pd.read_csv(r"data/flight.csv")# 使用透视表, 重构  
DataFrame
```

```
df = df.pivot_table(index="年份", columns="月份",  
values="人数")
```

```
# 绘制图表
```

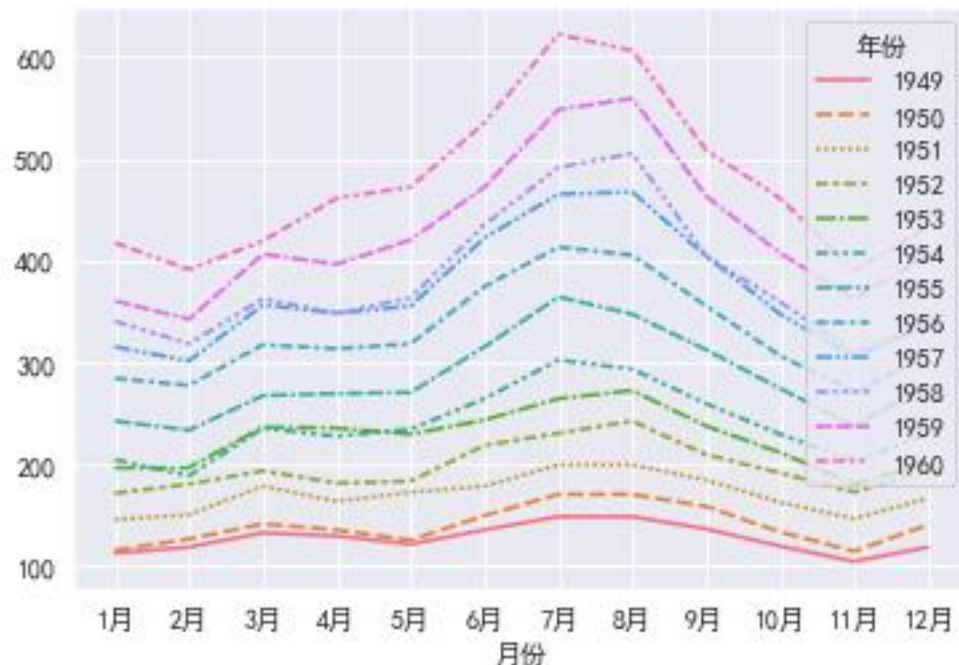
```
sns.lineplot(data=df["1月"])# 显示
```

```
plt.show()
```



2.1.3 实际案例-按年、月统计乘客数量

```
import pandas as pd#例3-5
import matplotlib.pyplot as plt
import seaborn as sns# 设置
sns.set_style("darkgrid")
sns.set_style({"font.sans-serif": "SimHei"})# 读取数据
df = pd.read_csv(r"data/flight.csv")# 使用透视表, 重构
DataFrame
df = df.pivot_table(index="年份", columns="月份",
values="人数")# 调整顺序
orders = [str(i)+"月" for i in range(1, 13)]
df = df[orders]
df.to_excel(r"data/2.xlsx")# 行列转置
df = df.T
# 绘制图表
sns.lineplot(data=df)
# 显示
plt.show()
```



2.2 关系图

relplot可绘制散点图（默认）或线图，支持多子图（col/row 参数），适合多维度分组分析，探索变量关系，尤其是需要按多个分类变量拆分视图时。语法结构

```
seaborn.relplot(*,x=None,y=None,hue=None,size=None,style=None,data=None,kind=None,dashed=None,markers=None)
```

x, y: x轴和y轴指定变量

hue: 分组变量产生不同的颜色

size: 分组变量产生不同大小元素

style: 分组变量将产生具有不同样式的元素

Data: 输入数据结构

dashes: 确定如何为style变量的不同级别绘制线条的对象

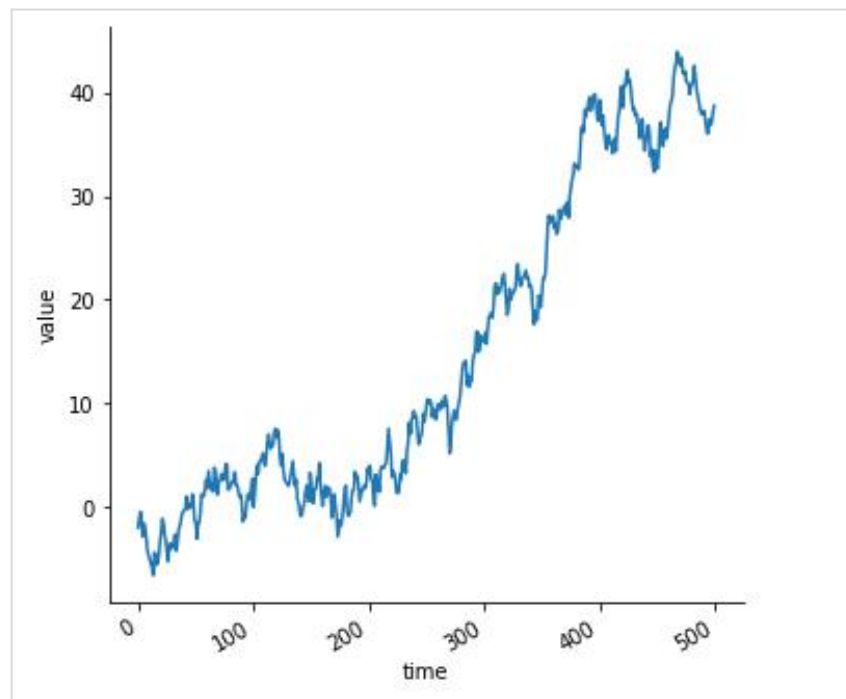
markers : 绘制图例

kind: 默认是 散点图，line为折线图

2.2.1 普通折线图

折线图是排列在工作表的列或行中的数据可以绘制到折线图中。折线图可以显示随时间（根据常用比例设置）而变化的连续数据，因此非常适用于显示在相等时间间隔下数据的趋势。

```
import numpy as np#例3-6
import pandas as pd
import seaborn as sns
# 二维数组 cumsum累计和
df = pd.DataFrame(dict(time=np.arange(500),
value=np.random.randn(500).cumsum()))
g = sns.relplot(x="time", y="value", kind="line",
data=df)
# 更改x显示方式，斜着显示
g.fig.autofmt_xdate()
```



2.2.2 关系图实例-数据集为fmri

事件相关功能核磁共振成像数据。由测试者、时间点、事件、刺激类型、大脑区域，信号等组成。

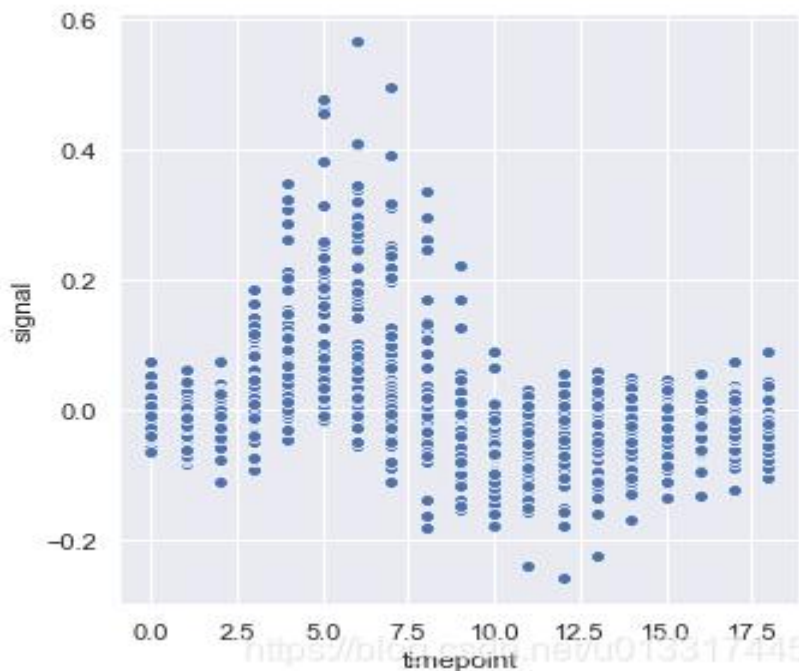
```
fmri= sns.load_dataset("fmri")  
fmri.head()
```

	subject	timepoint	event	region	signal
0	s13	18	stim	parietal	-0.017552
1	s5	14	stim	parietal	-0.080883
2	s12	18	stim	parietal	-0.081033
3	s11	18	stim	parietal	-0.046134
4	s10	18	stim	parietal	-0.037970

2.2.2 关系图实例——数据集为fmri-散点图测试

事件相关功能核磁共振成像数据。由测试者、时间点、事件、刺激类型、大脑区域，信号等组成。

```
sns.relplot(x="timepoint", y="signal", data=fmri)
```



一个x有多个y，怎么聚合呢？默认的是 aggregate the multiple measurements at each x value by plotting the mean and the 95% confidence interval around the mean:

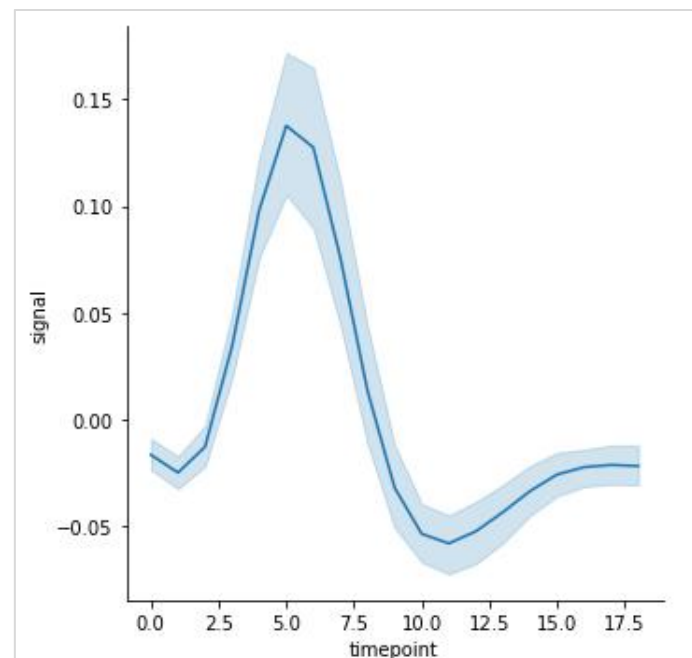
2.2.2 关系图实例——折线图-带置信区

聚合表示不确定性

使用环境

对于x变量的相同值，更复杂的数据集将具有多个度量。seaborn的默认行为是x通过绘制均值和均值周围的95%置信区间来汇总每个值的多次测量：

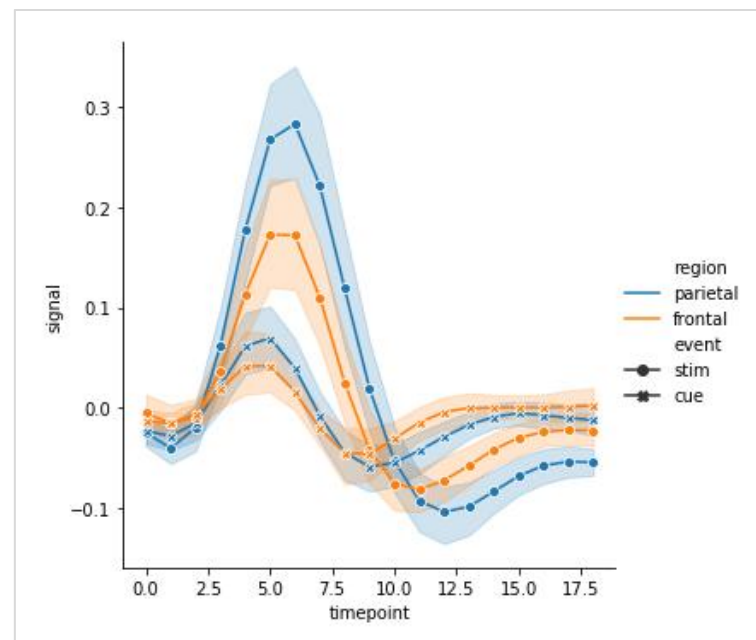
```
import seaborn as sns#例3-7
# 读入seaborn自身带的数据
fmri = sns.load_dataset("fmri")
sns.relplot(x="timepoint", y="signal", kind="line",
            data=fmri)
```



2.2.2 关系图实例-颜色与样式分类

添加具有两个级别的色相语义会将绘图分成两条线和错误带，并分别给它们上色以指示它们对应于数据的哪个子集。

```
import seaborn as sns#例3-8
# 读入seaborn自身带的数据
fmri = sns.load_dataset("fmri")
print(fmri)
sns.relplot(x="timepoint", y="signal", hue="region",
            style="event",
            dashes=False, markers=True, kind="line",
            data=fmri)
```

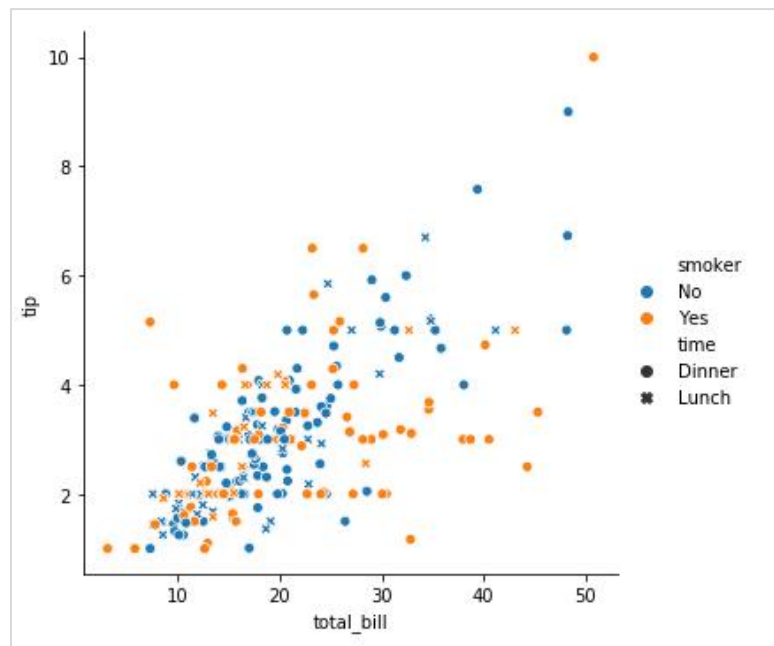


2.3 关系图- 散点图

使用环境：

散点图是统计可视化的主要内容。它使用点云描绘了两个变量的联合分布，其中每个点代表数据集中的观测值。这种描述使眼睛可以推断出有关它们之间是否存在任何有意义关系的大量信息。

```
import seaborn as sns#例3-9
tips = sns.load_dataset("tips")
print(tips)
sns.relplot(x="total_bill", y="tip", hue="smoker",
style="time", data=tips)
```



2.3.1 散点图-实例

本例中，我们的数据集采用seaborn库自带的小费数据集tips。

tips小费数据集：

是一个餐厅服务员收集的小费数据集，包含了7个变量：

总账单、小费、顾客性别、是否吸烟、日期、吃饭时间、顾客人数

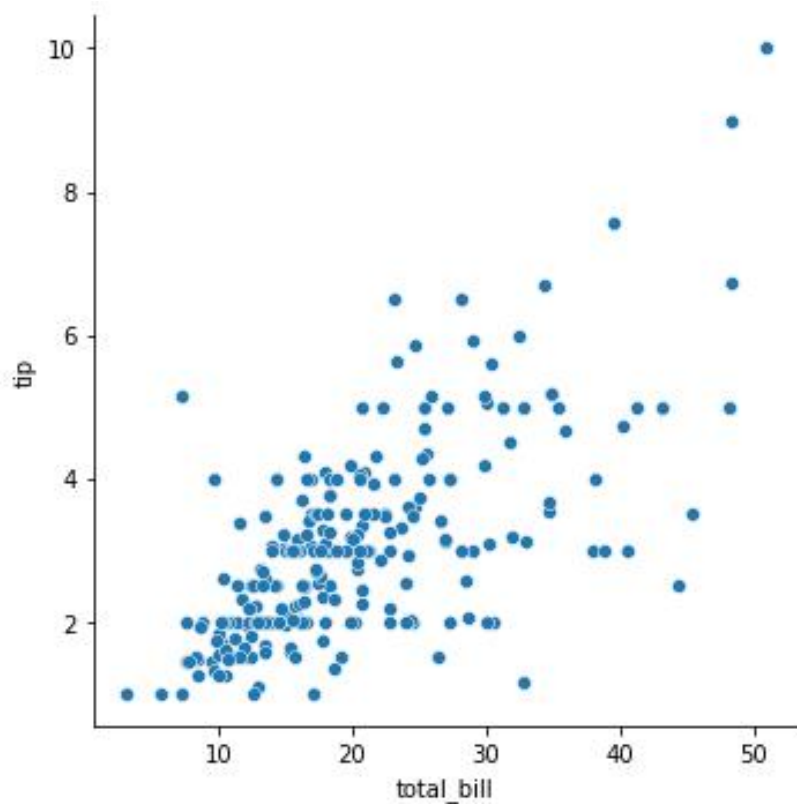
```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import numpy as np
tips= sns.load_dataset("tips")
tips.head()
```

2.3.1 散点图-实例

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

2.3.1 散点图-实例（总价与消费）

```
sns.relplot(x='total_bill', y='tip', data=tips)#例3-10
```



可视化后，可以看出：
给的小费大小集中在[0,6]
账单集中在[7,36]

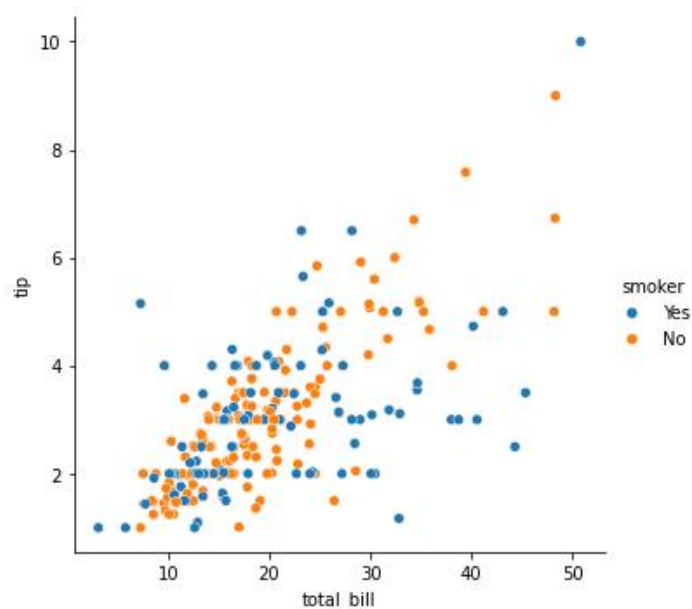
可以看出，消费高的小费多。

2.3.1 散点图-实例（参数hue区分性别）

用不同的颜色区分出来

在smoker维度，smoker:取值有：No、Yes 蓝yes橙no

```
sns.relplot(x='total_bill', y='tip', data=tips, hue='smoker')#例3-11
```

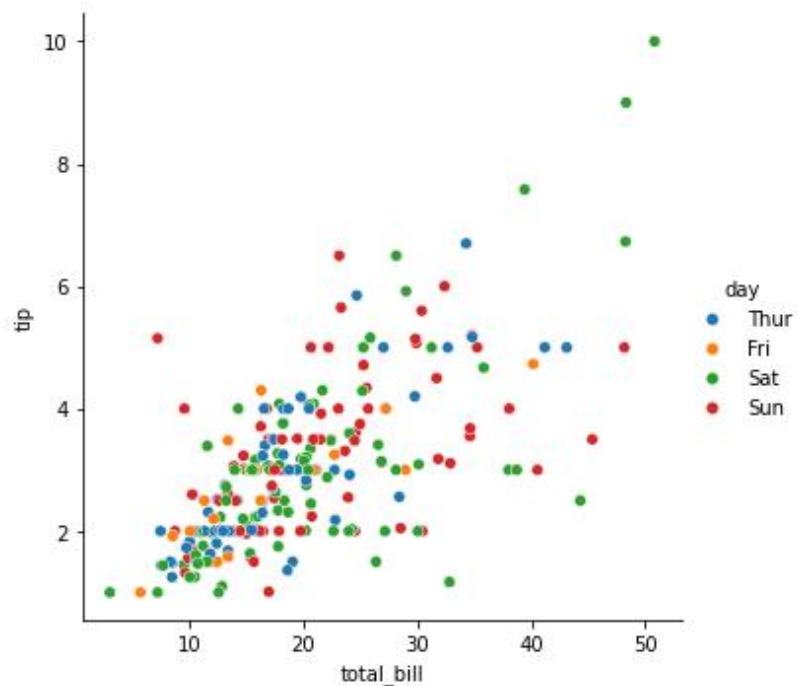


可视化后，可以看出：
不抽烟的给小费高于抽烟的。

2.3.1 散点图-实例 (hue+palette)

palette自定义颜色范围

```
set(tips.day) # {'Fri', 'Sat', 'Sun', 'Thur'} sns.relplot(x='total_bill', y='tip', data=tips,  
hue='day')#例3-12
```



可视化后，可以看出：
Saturday和Sunday小费略
高于其他两天。

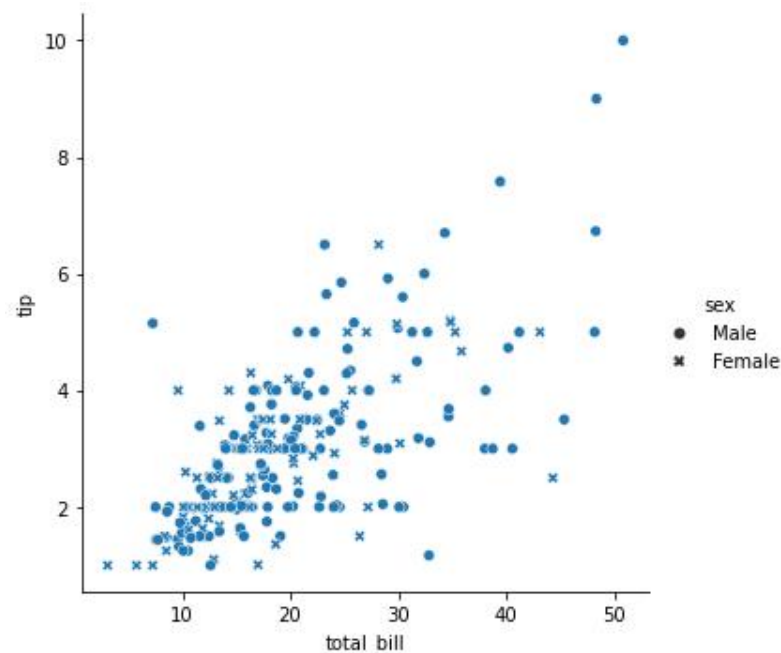
2.3.1 散点图-实例（参数style）

不同的表示形状上区分

性别维度上，不同的性别， 原点： Male 叉叉： Female

#例3-13

```
sns.relplot(x='total_bill', y='tip', data=tips, style='sex')
```



可视化后，可以看出：
给小费的男性多，男性给的小费高一点。

2.3.1 散点图-实例 (hue+style)

抽烟的女性:蓝色+叉叉

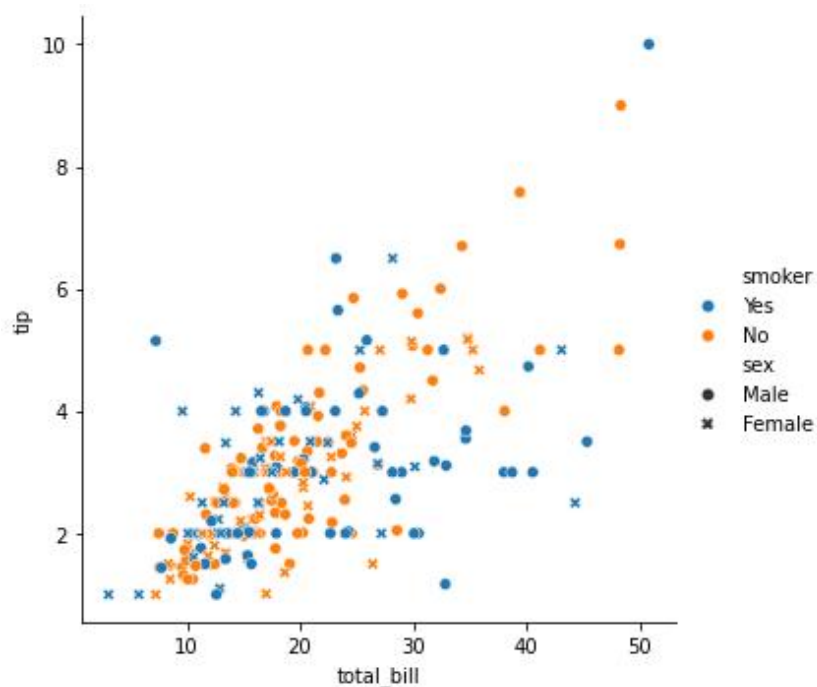
抽烟的男性: 蓝色+原点

不抽烟的女性: 橘色+叉叉

不抽烟的男性: 橘色+原点

#例3-14

```
sns.relplot(x='total_bill', y='tip', data=tips, hue='smoker', style='sex')
```



可视化后，可以看出：
给高小费的（超过5的），
竟然大都是不抽烟的男性
（橘色+原点）。

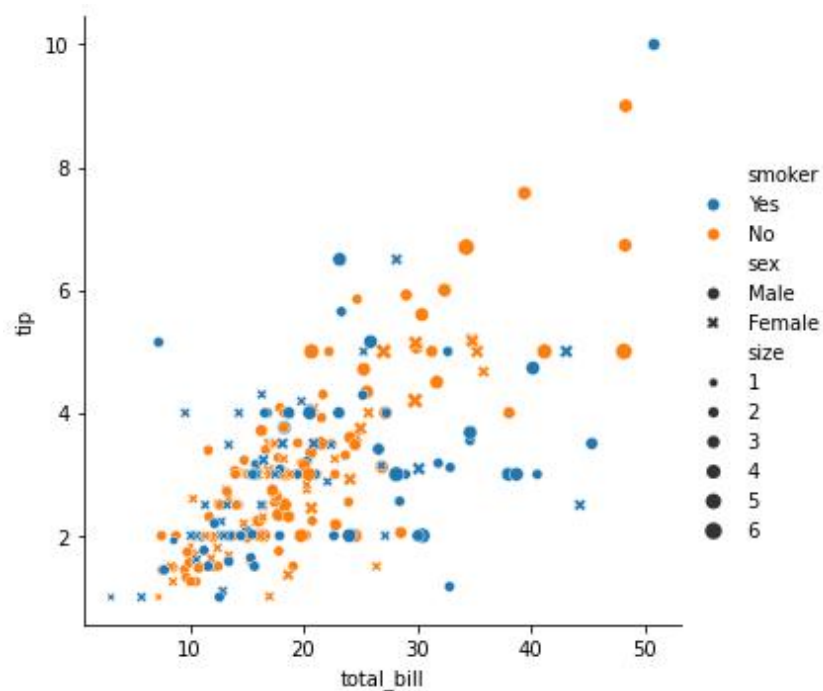
2.3.1 散点图-实例（参数size）

控制点的大小或者线条粗细

巧妙引入size维度（顾客人数）信息

#例3-15

```
sns.relplot(x='total_bill', y='tip', data=tips, hue='smoker', style='sex', size='size')
```



可视化后，可以看出：
图上可以看出，人多消费大，
消费大小费高。

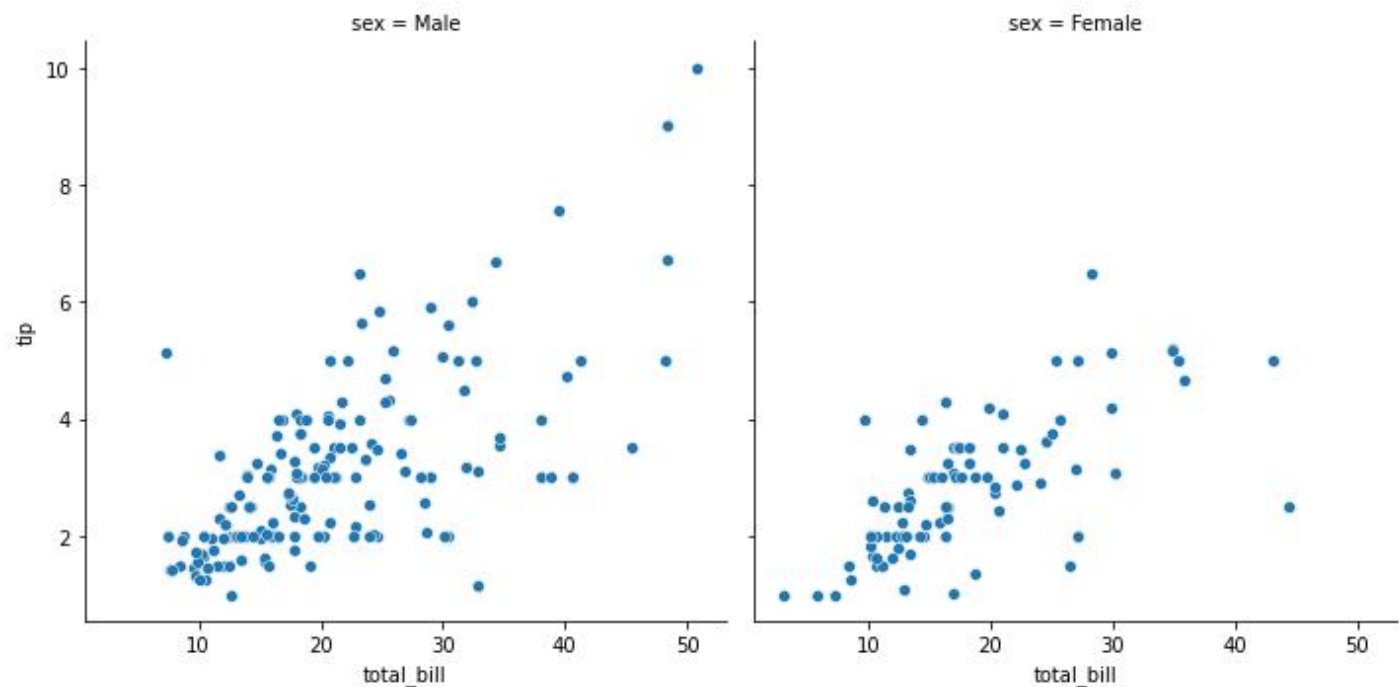
2.3.2 散点图——使用子图展示多重关系

参数col、row可以帮助我们实现分子图展示。

col="sex" 有几种sex，就有几列图（2种male和female，2列图）

row="smoker" 有几种smoker，就有几行图

```
#例3-16sns.relplot(x="total_bill", y="tip", col="sex", data=tips)
```



可视化后，可以看出：
男性的小费高于女性的小费。

2.3.2 散点图——使用子图展示多重关系（续）

参数col、row可以帮助我们实现。

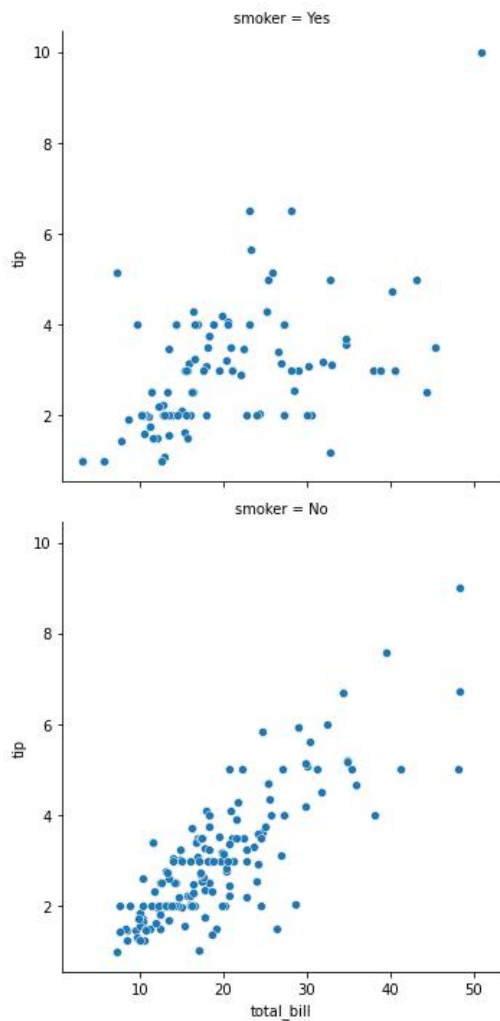
col="sex" 有几种sex，就有几列图（2种male和female，2列图）

row="smoker" 有几种smoker，就有几行图

```
#3-17
```

```
sns.relplot(x="total_bill", y="tip", row="smoker", data=tips)
```

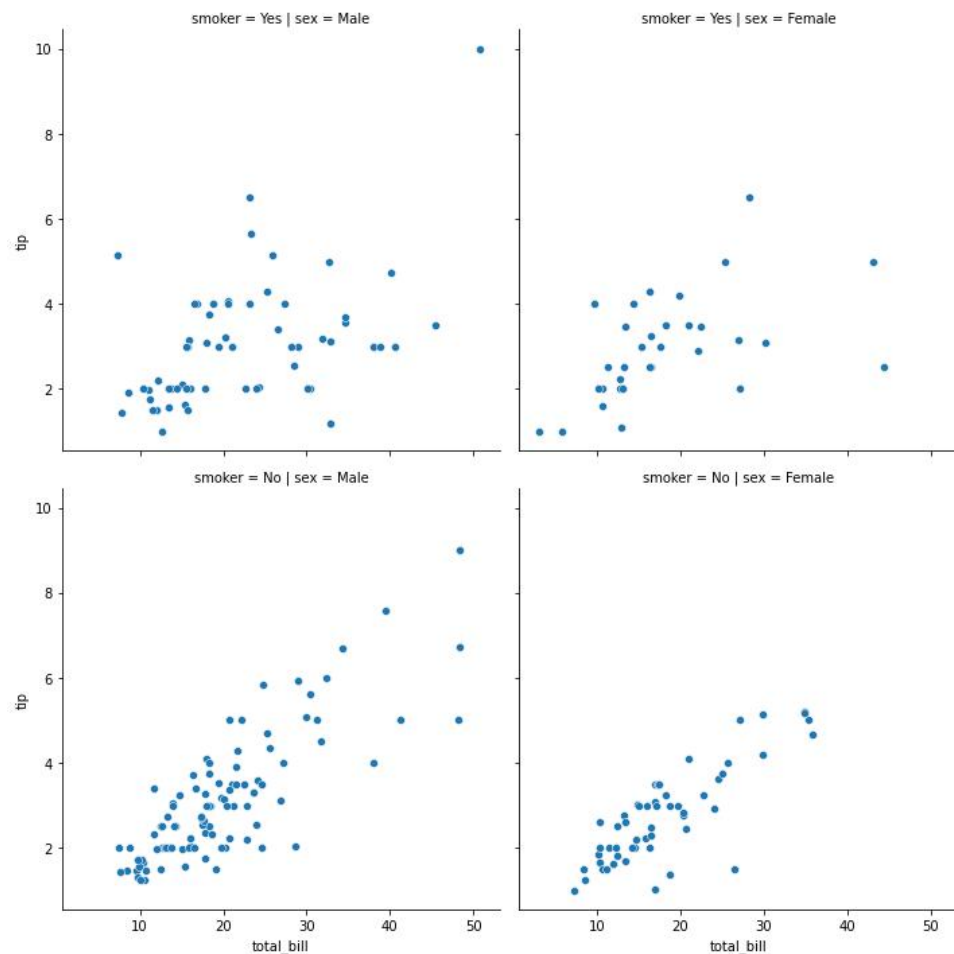
2.3.2 散点图——使用子图展示多重关系（续）



可视化后，可以看出：
不抽烟的小费
高于抽烟的。

2.3.2 散点图——使用子图展示多重关系（续）

```
#例3-18sns.relplot(x="total_bill", y="tip", row="smoker", col="sex", data=tips)
```



可视化后，可以看出：
不抽烟的男性
小费给的最多。

2.4 热力图

- 在Seaborn中，我们可以使用heatmap()函数来绘制一个热力图。热力图的主要作用是：以高亮的方式来显示区域的密度情况，以展示数据的差异性。
- **语法：**
- `sns.heatmap(data)`

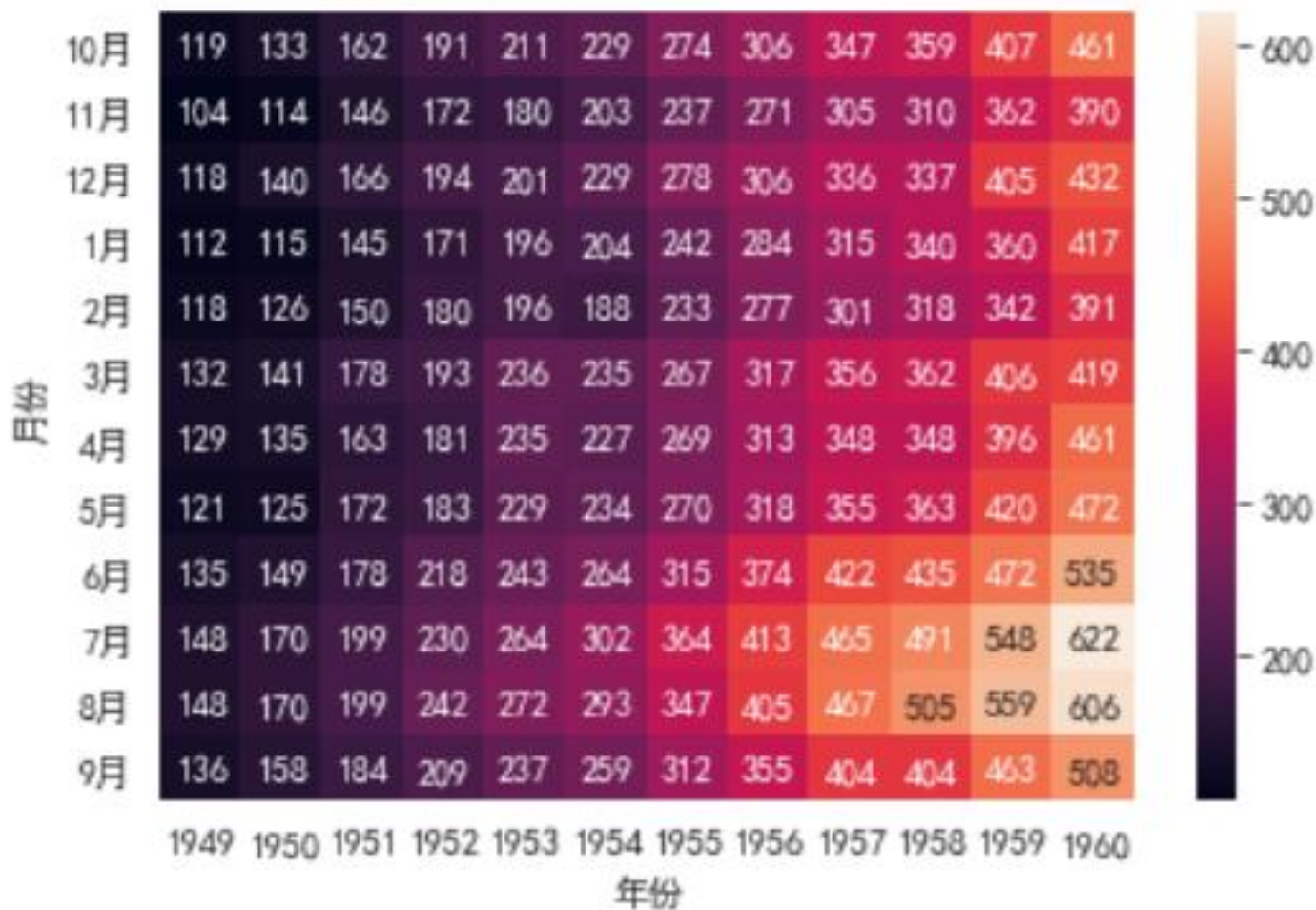
2.4.1 heatmap()的参数

参数	说明
data	数据部分（二维）
annot	添加注释文本，True或False
fmt	定义数据格式
cmap	定义颜色风格
linewidth	添加间距
cbar	显示颜色条，True或False

2.4.2 热力图实例

```
import pandas as pd#例3-19-1
import matplotlib.pyplot as plt
import seaborn as sns# 设置
sns.set_style("darkgrid")
sns.set_style({"font.sans-serif": "SimHei"})# 读取数据
df = pd.read_csv(r"data/flight.csv")# 重构DataFrame（透视表）
df = df.pivot_table(index="年份", columns="月份", values="人数")
# 行列转置
df = df.T
# 绘制图表
sns.heatmap(data=df, annot=True, fmt="d")
# 显示
plt.show()
```

2.4.2 热力图实例（续）



2.5 线性回归图

线性回归图，也叫做“回归图”，它主要用于表现两个变量之间的线性关系。线性回归图建立在散点图的基础上，它会在散点图上面增加一条直线（也可能是曲线）。对于这条直线来说，它是使用最小二乘法预测的两个变量的关系： $y=ax+b$ 。

2.5.1 基本语法

- 在Seaborn中，我们可以使用regplot()函数来绘制一个线性回归图。
- **语法：**

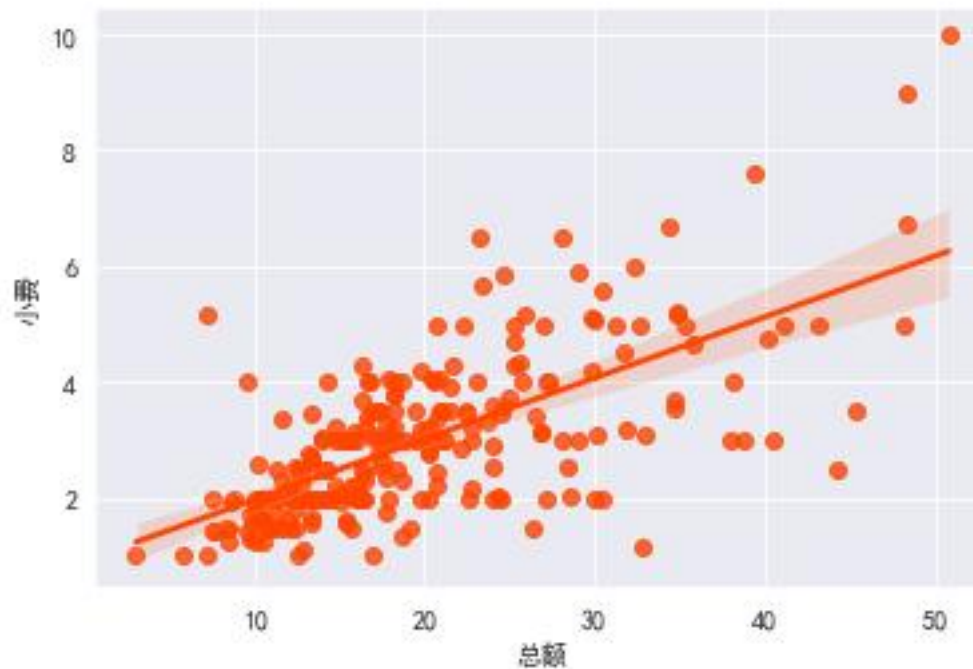
```
sns.regplot(data, x, y)
```

2.5.2 regplot()的参数

参数	说明
data	数据部分
x	x轴坐标
y	y轴坐标
color	整体颜色
marker	散点外观
ci	置信区间

2.5.3 线性回归图实例

```
import pandas as pd#例3-19-2
import matplotlib.pyplot as plt
import seaborn as sns
# 设置
sns.set_style("darkgrid")
sns.set_style({"font.sans-serif": "SimHei"})
# 读取数据
df = pd.read_csv(r"data/tip.csv")
# 绘制图表
sns.regplot(data=df, x="总额", y="小费",
color="orangered")
# 显示
plt.show()
```



$$a = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - (\sum x_i)^2}$$

$$b = \frac{\sum y_i - a\sum x_i}{n}$$

2.6 多变量图

- 多变量图，也叫做“矩阵图”或“多变量关系图”。它其实是将数据集中不同的字段进行“两两比较”（包括自己与自己对比）。对于多变量图来说，默认情况下它对角线上的图表是直方图，其他的都是散点图。
- 在Seaborn中，我们可以使用pairplot()函数来绘制一个多变量图。
- 语法: `sns.pairplot(data)`

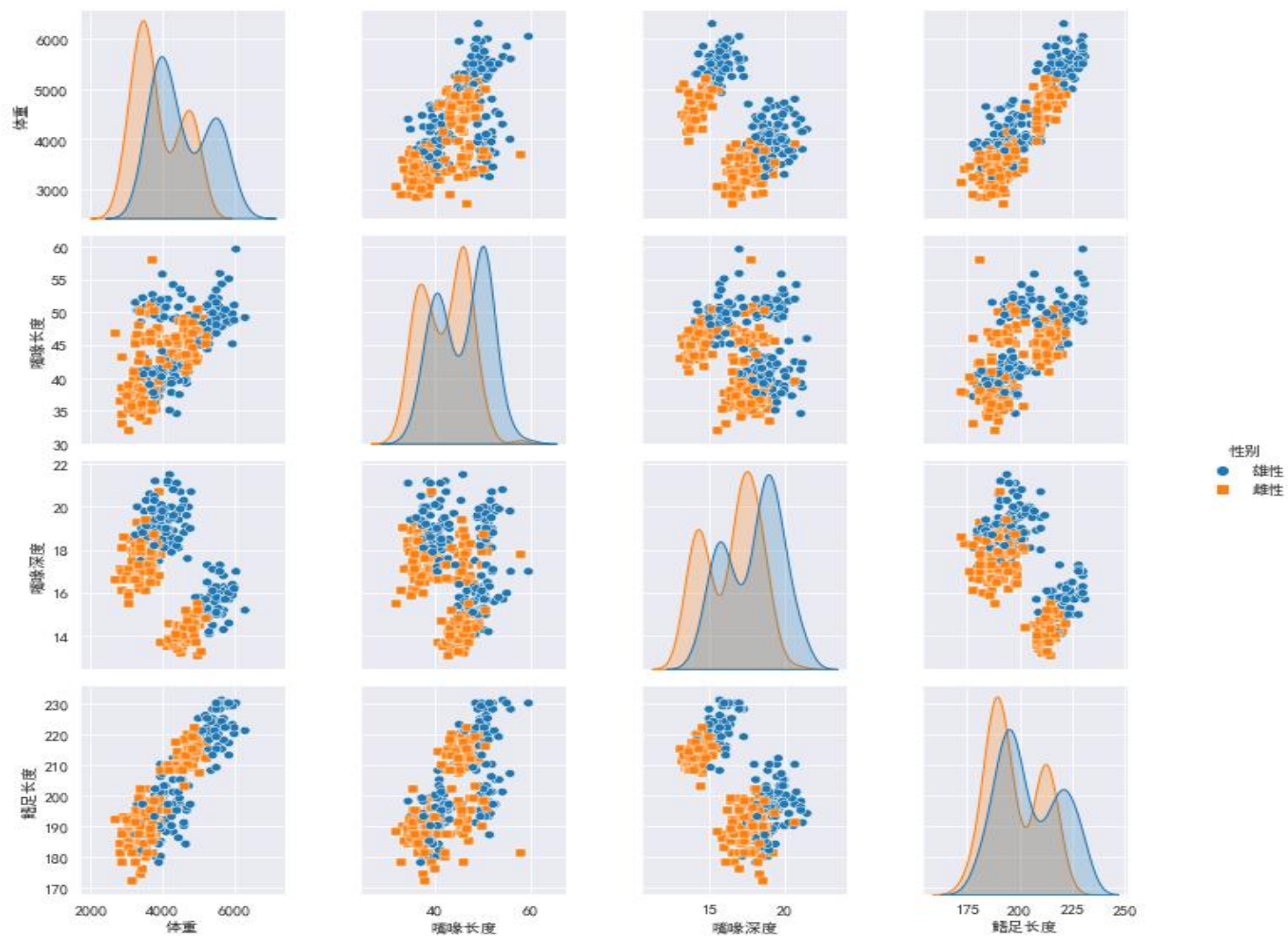
2.6.1 pairplot()的参数

参数	说明
data	数据部分
hue	添加区分（颜色）
diag_kind="hist"	对角使用直方图
marker	散点外观
vars	x轴和y轴都使用的字段
x_vars	x轴使用的字段
y_vars	y轴使用的字段

2.6.2 多变量图实例

```
import pandas as pd#例2-19-3
import matplotlib.pyplot as plt
import seaborn as sns
# 设置
sns.set_style("darkgrid")
sns.set_style({"font.sans-serif": "SimHei"})
# 读取数据
df = pd.read_csv(r"data/penguin.csv")
# 绘制图表
sns.pairplot(data=df, hue="性别",
markers=["o", "s"])
# 显示
plt.show()
```

2.6.2 多变量图实例 (续)





/03

分布图

3.1 直方图

1. 比较漂亮的直方图

直方图是条形图，其中代表数据变量的轴分为一组离散的条带，并使用相应条形的高度显示了每个条带内的观测值计数

```
seaborn.displot(data=None, *, x=None, y=None, hue=None, row=None, col=None, kind=None)
```

displot绘制函数。

data: pandas.DataFrame, numpy.ndarray数据结构。

x,y: 向量。

hue: 定义子集绘制在不同构面上的变量。

kind: 选择基础绘图功能并确定其他有效参数集。

row,col定义子集以绘制在不同构面上的变量。

3.1.1 Penguins数据

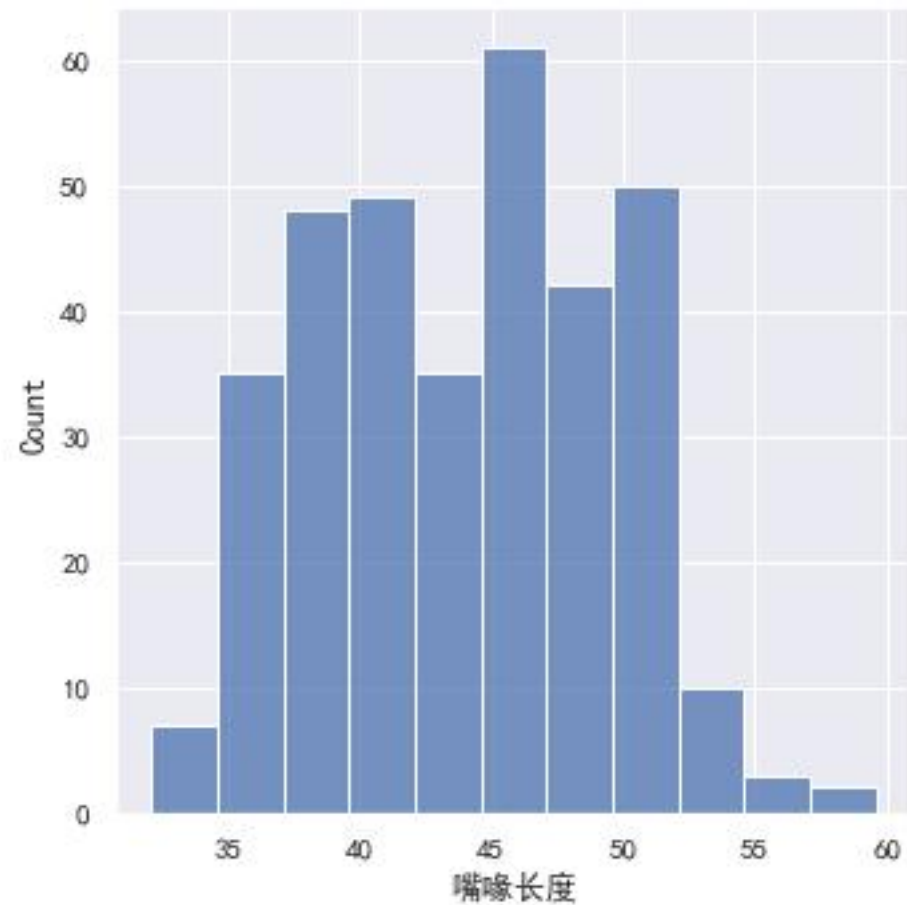
自带数据集，包含的三种企鹅的七个特征，分别是类型、所在岛屿，嘴巴长度，嘴巴深度，脚蹼长度，身体体积，性别。

	种类	岛屿	性别	体重	嘴喙长度	嘴喙深度	鳍足长度
0	阿德利企鹅	托格森岛	雄性	3750.0	39.1	18.7	181.0
1	阿德利企鹅	托格森岛	雌性	3800.0	39.5	17.4	186.0
2	阿德利企鹅	托格森岛	雌性	3250.0	40.3	18.0	195.0
3	阿德利企鹅	托格森岛	NaN	NaN	NaN	NaN	NaN
4	阿德利企鹅	托格森岛	雌性	3450.0	36.7	19.3	193.0

3.1.2 统计不同嘴喙长度的数量

#例3-19

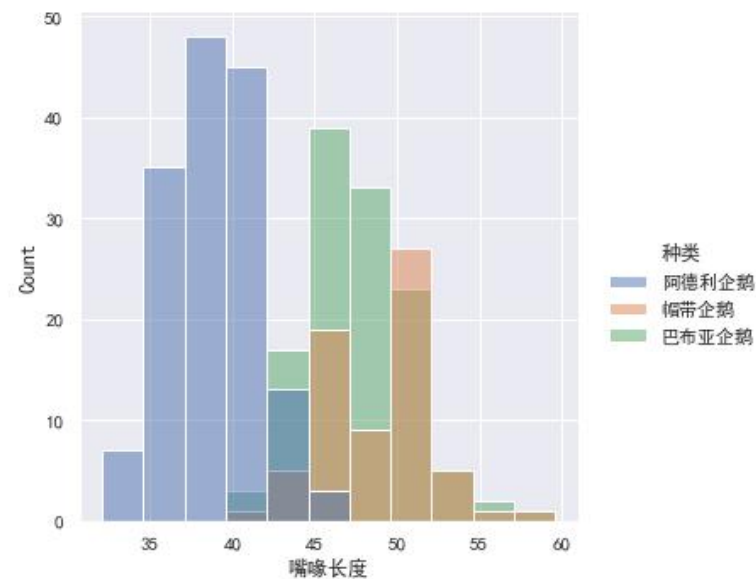
```
import seaborn as sns
import pandas as pd
#设置背景风格
sns.set_theme(style="darkgrid")
penguins=pd.read_csv("data/penguin.csv")
#penguins = sns.load_dataset("penguins")
print(penguins)
sns.displot(penguins,x="嘴喙长度")
```



3.1.2 统计不同嘴喙长度的数量——特征分布直方图

#例3-20

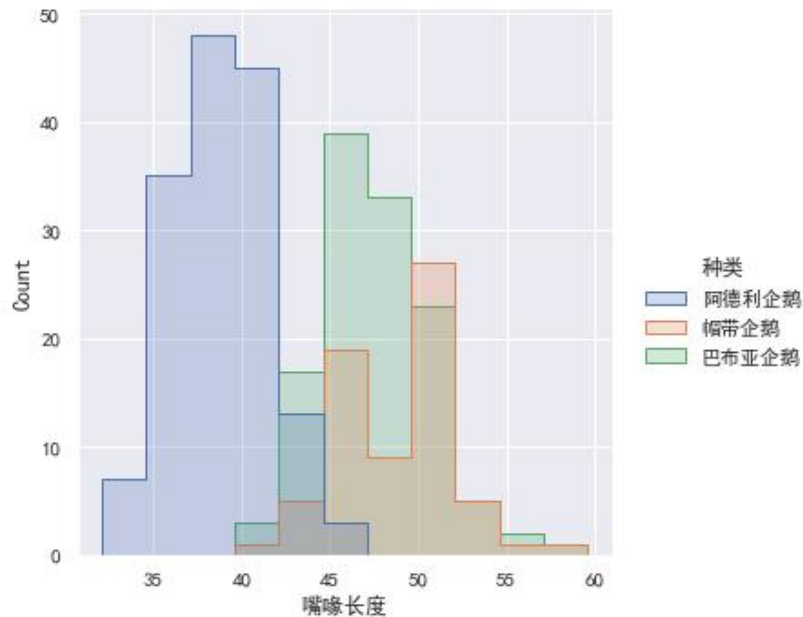
```
import seaborn as sns
import pandas as pd
#设置背景风格
sns.set_theme(style="darkgrid")
sns.set_style({"font.sans-serif": "SimHei"})
penguins=pd.read_csv("data/penguin.csv")
#penguins = sns.load_dataset("penguins")
# print(penguins)
sns.displot(penguins, x="嘴喙长度", hue="种类")
```



3.2 阶梯图

#例3-21

```
import seaborn as sns
import pandas as pd
#设置背景风格
sns.set_theme(style="darkgrid")
sns.set_style({"font.sans-serif": "SimHei"})
penguins=pd.read_csv("data/penguin.csv")
#penguins = sns.load_dataset("penguins")
# element类型=步梯
sns.displot(penguins, x="嘴喙长度", hue="种类",
            element="step")
```



3.3 堆叠图和避开图

#例3-22

```
import seaborn as sns
```

```
import pandas as pd
```

#设置背景风格

```
sns.set_theme(style="darkgrid")
```

```
sns.set_style({"font.sans-serif": "SimHei"})
```

```
penguins=pd.read_csv("data/penguin.csv")
```

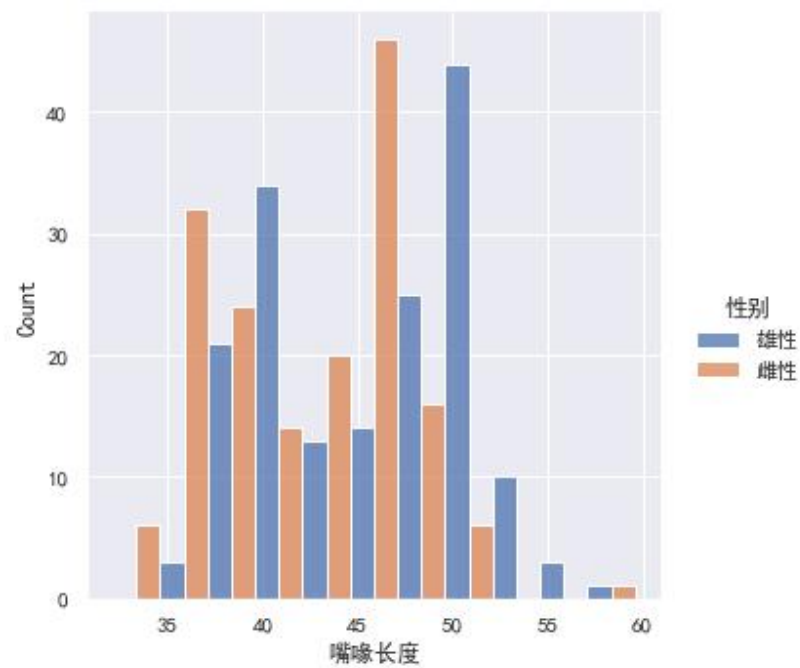
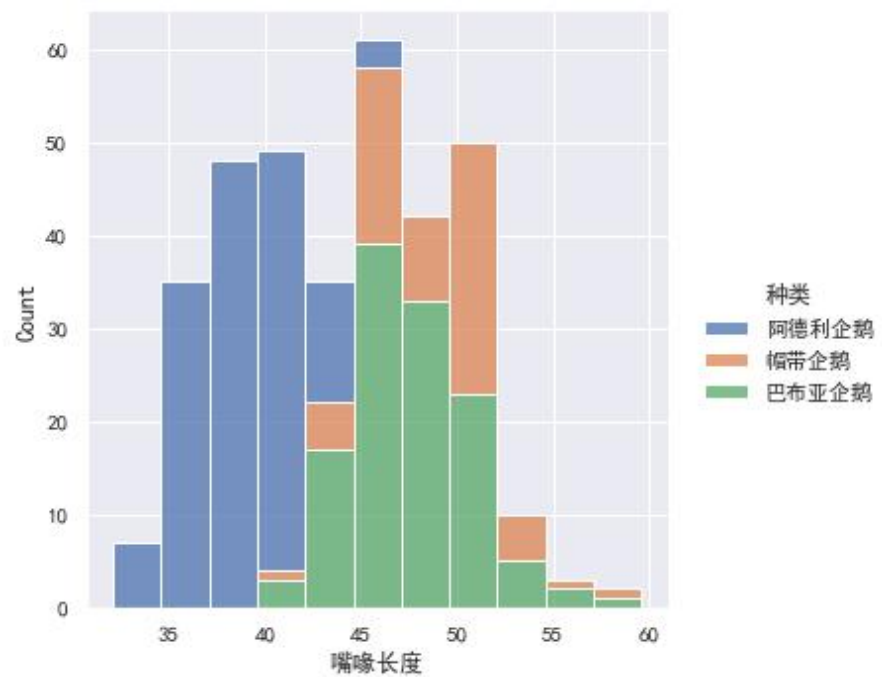
```
#penguins = sns.load_dataset("penguins")
```

multiple: 数量列表, stack堆叠, dodge避开

```
sns.displot(penguins, x="嘴喙长度", hue="种类", multiple="stack")
```

```
sns.displot(penguins, x="嘴喙长度", hue="性别", multiple="dodge")
```

3.3 堆叠图和避开图 (续)

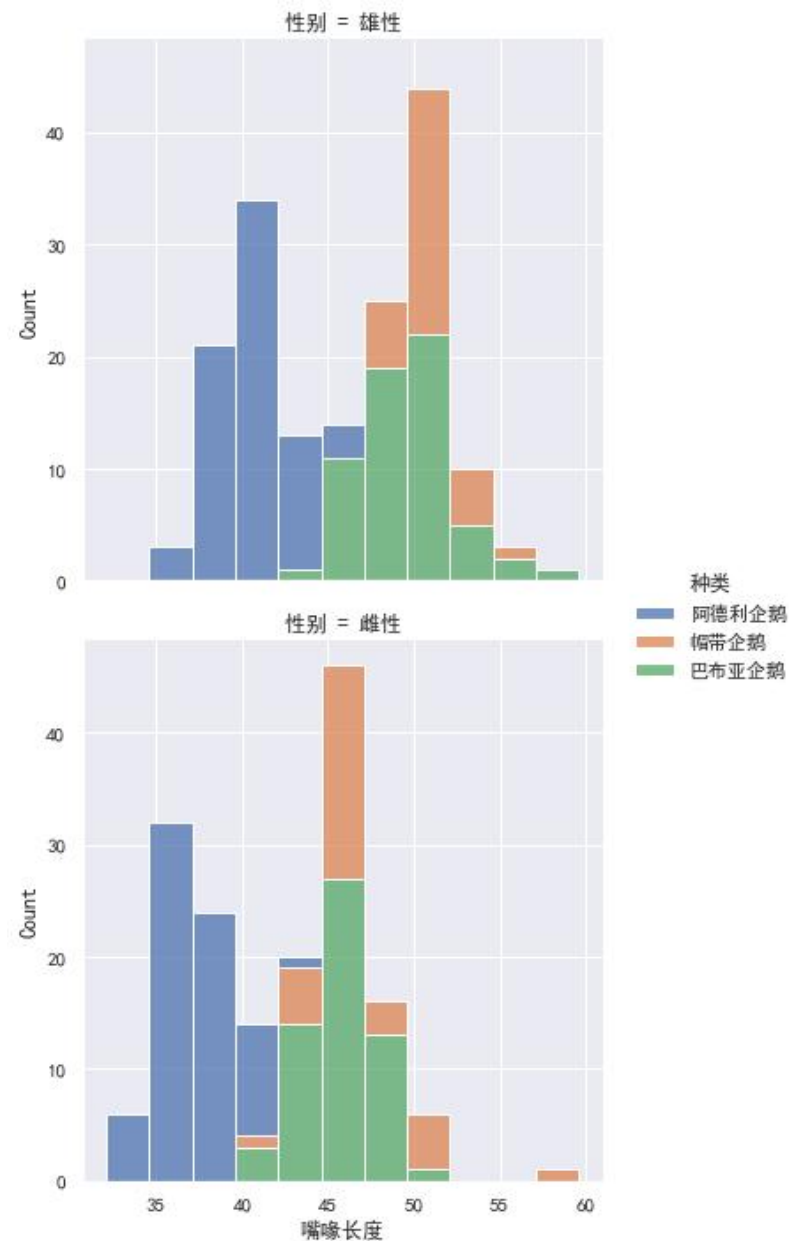


3.4 按行或列划分不同构面子图

#例3-23

```
import seaborn as sns
import pandas as pd
#设置背景风格
sns.set_theme(style="darkgrid")
sns.set_style({"font.sans-serif": "SimHei"})
penguins=pd.read_csv("data/penguin.csv")
#penguins = sns.load_dataset("penguins")
# multiple: 数量列表, stack堆叠, dodge避开

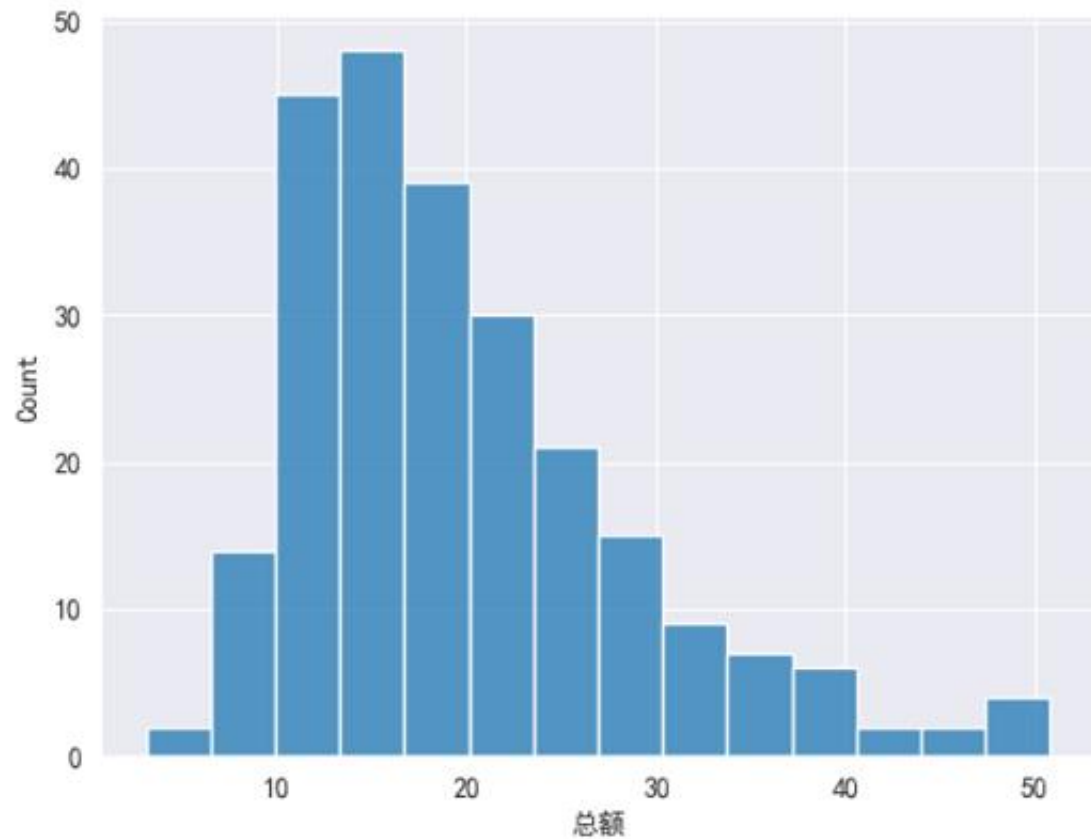
sns.displot(penguins, x="嘴喙长度", row='性别',
            hue="种类", multiple="stack")
```



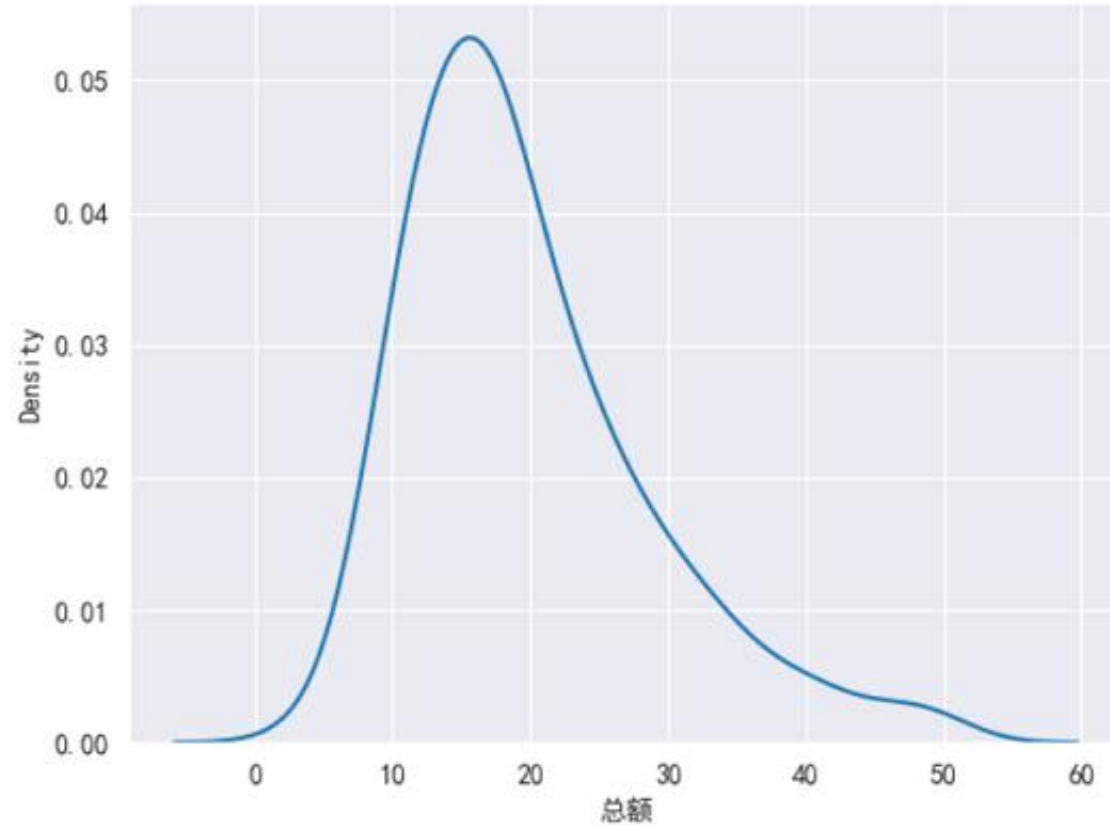
3.5 核密度图

- 核密度图和直方图的功能非常相似的，都是用于展示数据的分布情况，不过它们之间也有着以下2个方面的区别。
 - 直方图使用“条形”来展示，核密度图使用“曲线”来展示。
 - 直方图的纵坐标是数据的个数，核密度图的纵坐标是数据的密度

3.5 核密度图



直方图



核密度图

3.5.1 基本语法

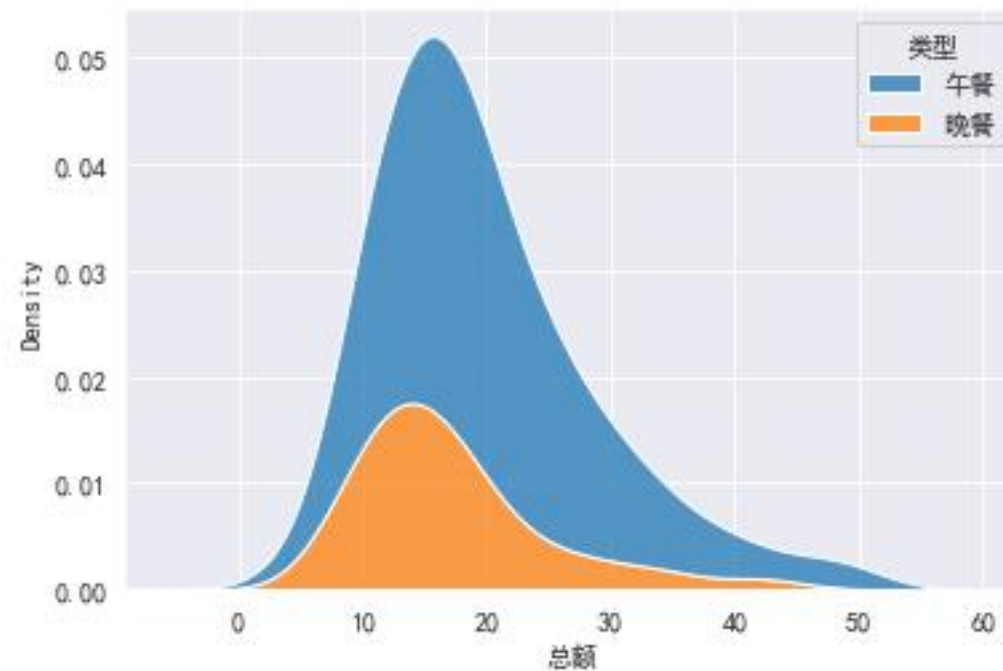
- 在Seaborn中，我们可以使用kdeplot()函数来绘制一个核密度图。其中，kdeplot是“Kernel Density Estimate（核密度估计）”的缩写。
- 语法: `sns.kdeplot(data, x, y)`

3.5.2 kdeplot()的参数

参数	说明
data	数据部分
x	x轴坐标
y	y轴坐标
hue	添加区分（颜色）
multiple="stack"	堆叠显示（面积图）
multiple="fill"	填充显示
bw_adjust	曲线平滑度
cut=0	切除小数部分
common_norm=False	独立计算面积

3.5.3 核密度图实例

```
import pandas as pd#例3-24
import matplotlib.pyplot as plt
import seaborn as sns
# 设置
sns.set_style("darkgrid")
sns.set_style({"font.sans-serif": "SimHei"})
# 读取数据
df = pd.read_csv(r"data/tip.csv")
# 绘制图表
sns.kdeplot(data=df, x="总额", hue="类型", multiple='stack')
# 显示
plt.show()
```



3.6 小提琴图

- 小提琴图，是一种组合型的图表，它同时结合了箱线图和核密度图的功能。
在Seaborn中，我们可以使用violinplot()函数来绘制一个小提琴图。
- **语法：**

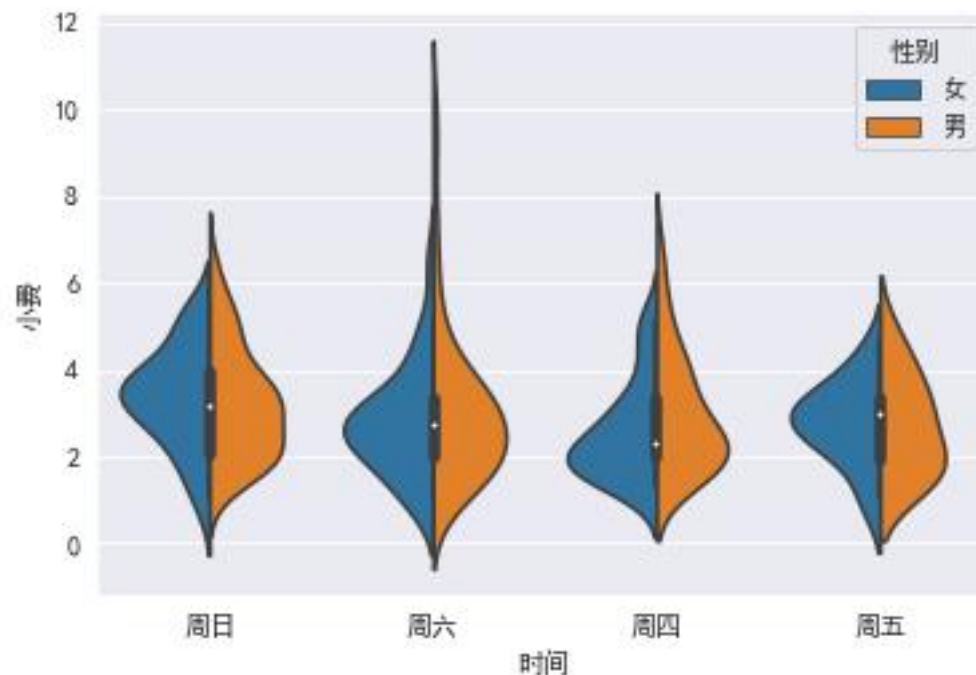
```
sns.violinplot(data, x, y)
```

3.6.1 violinplot()的参数

参数	说明
data	数据部分
x	x轴坐标
y	y轴坐标
hue	添加区分（颜色）
order	改变顺序
split=True	合并显示
inner="quartile"	添加百分位线
inner="stick"	每一个数据一条线
bw_adjust	曲线平滑度

3.6.1 小提琴图实例

```
import pandas as pd#例3-25
import matplotlib.pyplot as plt
import seaborn as sns
# 设置
sns.set_style("darkgrid")
sns.set_style({"font.sans-serif": "SimHei"})
# 读取数据
df = pd.read_csv(r"data/tip.csv")
# 绘制图表
sns.violinplot(data=df, x="时间", y="小费",
hue="性别", split=True)
# 显示
plt.show()
```





/04

分类图

4.1 分类散点图

- 在Seaborn中，我们可以使用stripplot()函数来绘制一个分布散点图。分布散点图和普通散点图不一样：普通散点图是用于判断两个变量是否存在关联趋势，而分类散点图是用于查看数据的分布情况。
- 语法：

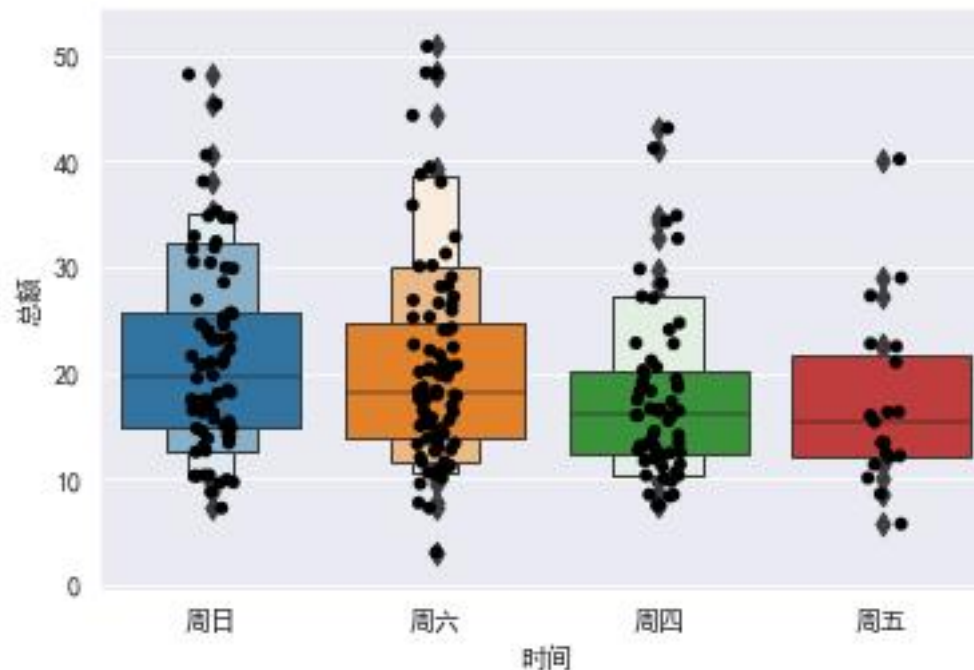
```
sns.stripplot(data, x, y)
```

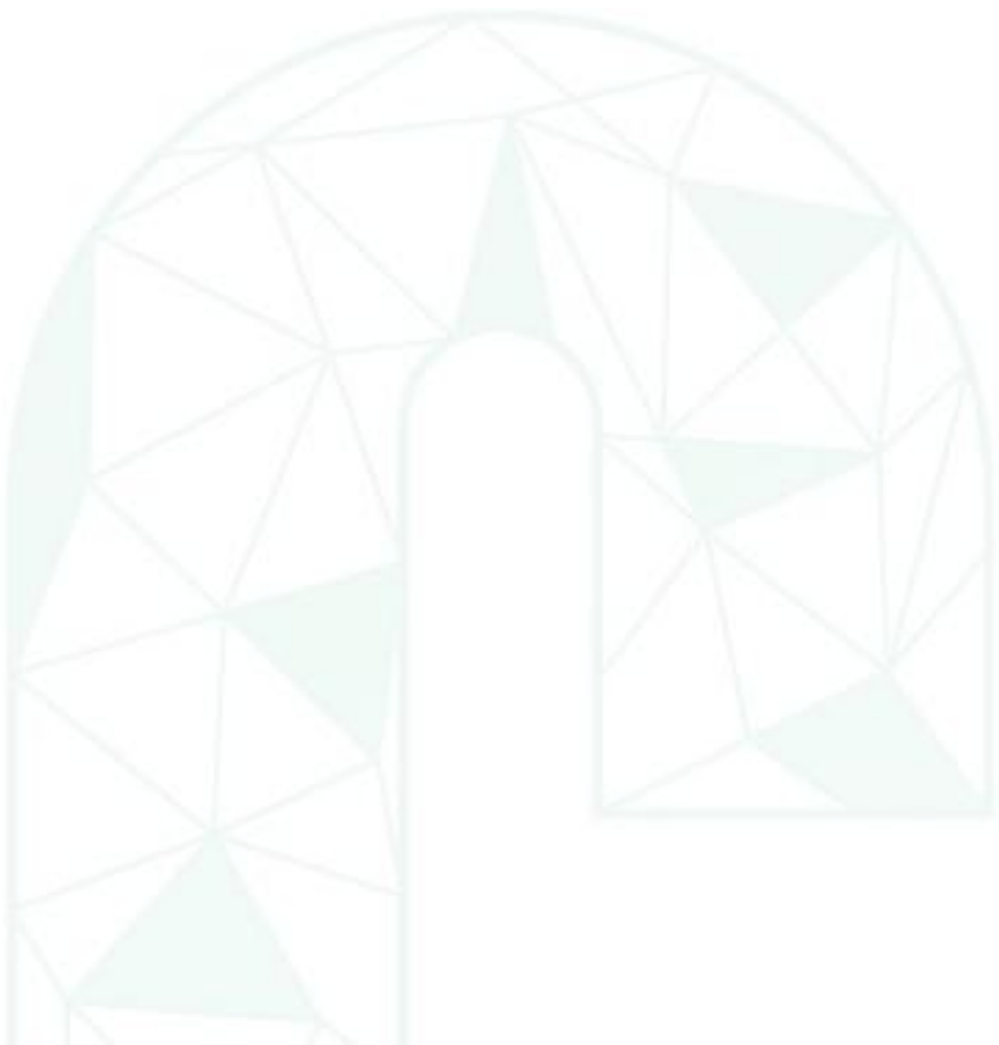
4.1.1 stripplot()的参数

参数	说明
data	数据部分
x	x轴坐标
y	y轴坐标
hue	添加区分
dodge=True	单独显示
order	改变顺序

4.1.2 分类散点图实例

```
import pandas as pd#例3-26
import matplotlib.pyplot as plt
import seaborn as sns
# 设置
sns.set_style("darkgrid")
sns.set_style({"font.sans-serif": "SimHei"})
# 读取数据
df = pd.read_csv(r"data/tip.csv")
# 绘制图表
sns.stripplot(data=df, x="时间", y="总额",
color="black")
sns.boxenplot(data=df, x="时间", y="总额")
# 显示
plt.show()
```





/05

通用设置

5 通用设置

- 通用设置不仅可以用于折线图，还可以用于其他大多数图表。对于Seaborn来说，通用设置主要包括以下5个方面。

- 主题风格
- 定义标题
- 定义图例
- 刻度标签
- 刻度范围

5.1 主题风格

- 在Seaborn中, 我们可以使用`set_style()`函数来设置一种主题风格。
- `sns.set_style(style)`
- 五种风格: **darkgrid** , **whitegrid** , **dark** , **white** ,和 **ticks** 默认

5.2 定义标题

- 在Seaborn中，所有的绘图函数都会返回一个AxesSubplot对象，该对象代表的就是当前的图表。然后我们可以使用AxesSubplot对象提供的函数来设置各种标题。

- **语法：**

定义主标题

```
ax.set_title()
```

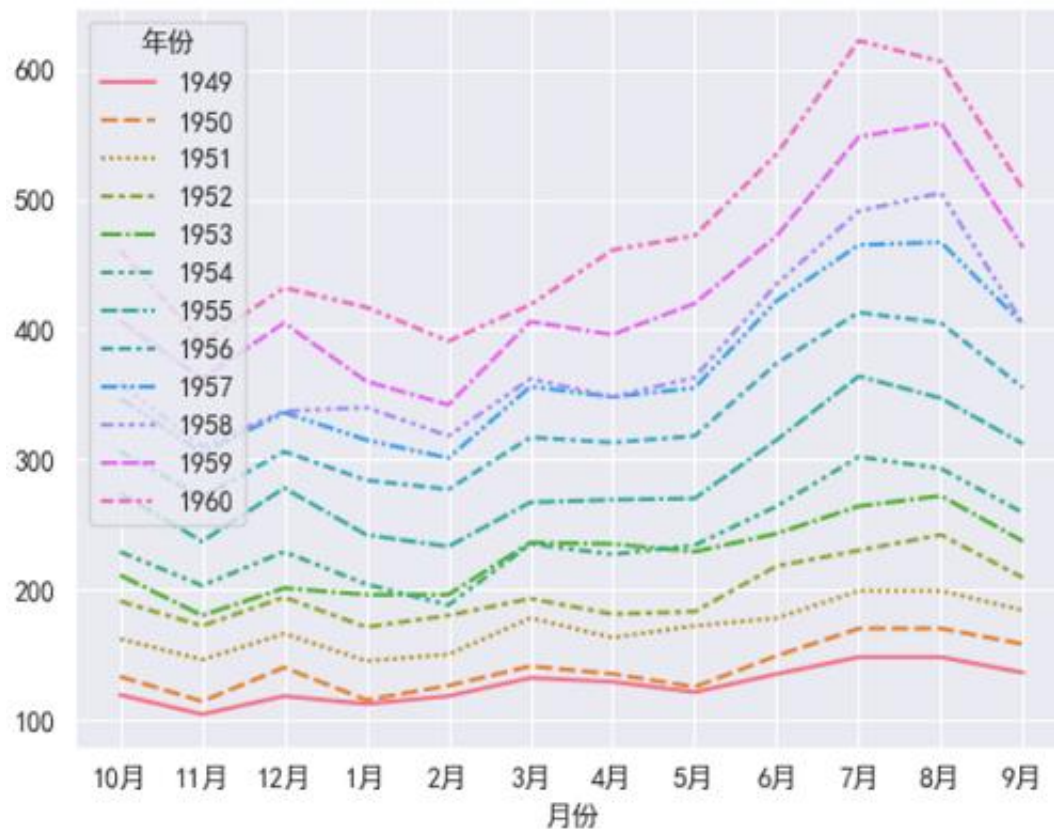
定义轴标题

```
ax.set_xlabel()
```

```
ax.set_ylabel()
```

5.3 定义图例

- 在实际开发中，图例都是在图表的内部进行绘制的。但是有些图例过多时，会覆盖原来的图表。
- 在Seaborn中，我们可以使用 `legend()` 这个函数来调整图例的位置。
- 语法：** `plt.legend(loc, bbox_to_anchor)`
- Loc:** 例upper, left
- bbox_to_anchor :** 例1.2,1



5.4 刻度标签

- 在Seaborn中，我们可以使用set_xticklabels()函数来定义x轴的刻度标签，也可以使用set_yticklabels()函数来定义y轴的刻度标签。
- **语法：**
- **ax.set_xticklabels(labels, rotation=n)**
- **ax.set_yticklabels(labels, rotation=n)**

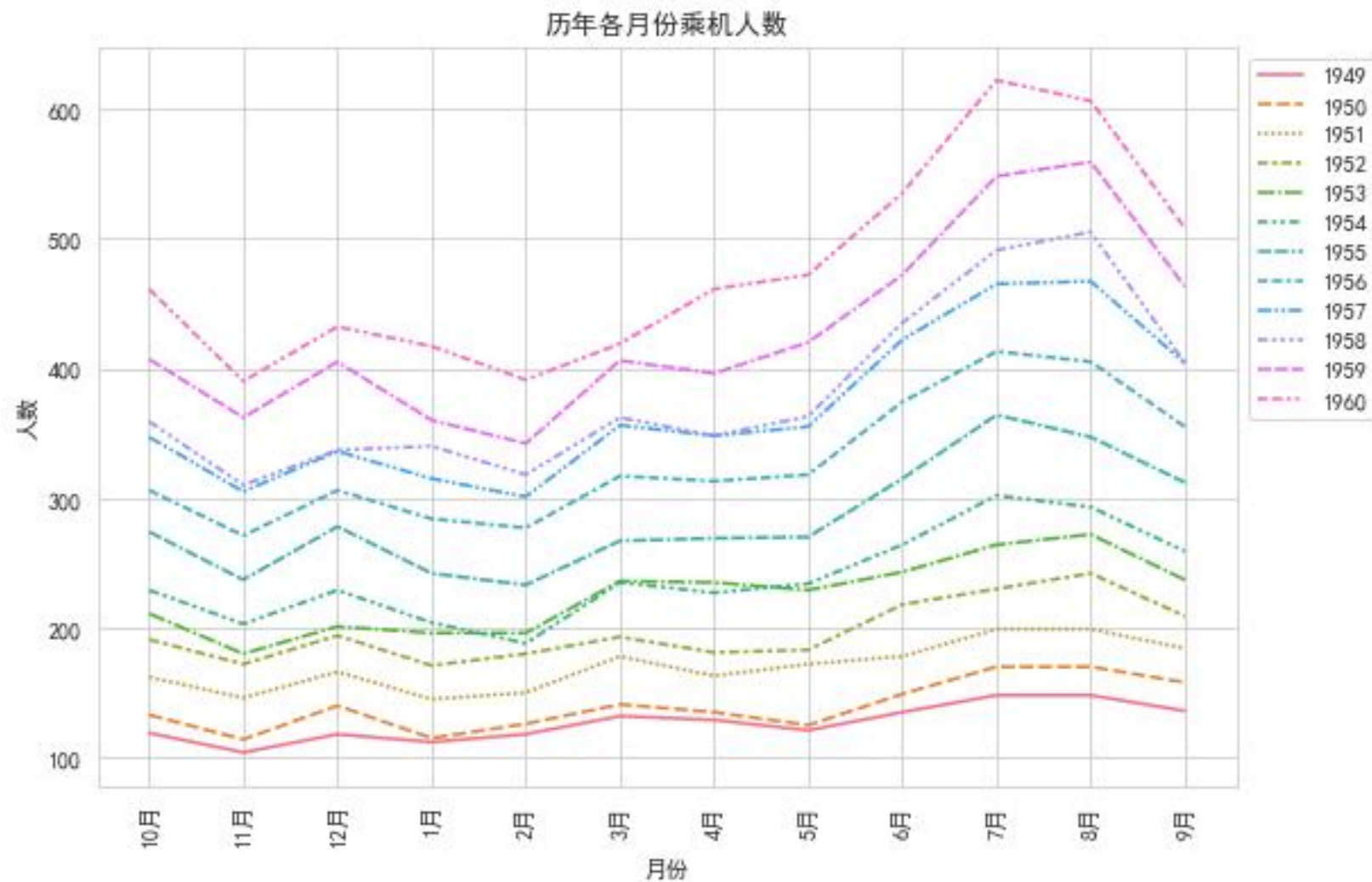
5.5 刻度范围

- 在Seaborn中，我们可以使用set_xlim()函数来定义x轴坐标的范围，也可以使用set_ylim()函数来定义y轴坐标的范围。
- 语法：
- **ax.set_xlim(left, right)**
- **ax.set_ylim(left, right)**

5.6 通用设置实例

```
import pandas as pd#例3-28
import matplotlib.pyplot as plt
import seaborn as sns# 设置
sns.set_style("whitegrid")
sns.set_style({"font.sans-serif": "SimHei"})# 画布大小
plt.figure(figsize=(9, 6))# 读取数据
df = pd.read_csv(r"data/flight.csv")# 使用透视表，重构DataFrame
df = df.pivot_table(index="年份", columns="月份", values="人数")# 行列转置
df = df.T# 绘制图表
ax=sns.lineplot(data=df)# 调整图例位置（注意这里是1，不再是1.2）
plt.legend(bbox_to_anchor=(1, 1))
ax.set_xticklabels(df.index,rotation=90)
ax.set_title('历年各月份乘机人数')
ax.set_ylabel('人数')# 显示plt.show()
```

5.6 通用设置实例（续）



The logo features a central green diamond containing the text. To the left of the diamond is a dark green chevron pointing right. To the right is a light green chevron pointing left. A horizontal line passes through the center of the composition. Several small diamonds in dark green, light green, and grey are positioned around the main elements.

Canllay 嘉为数字咨询

数字化人才培养
先行者