zhuanlan.zhihu.com

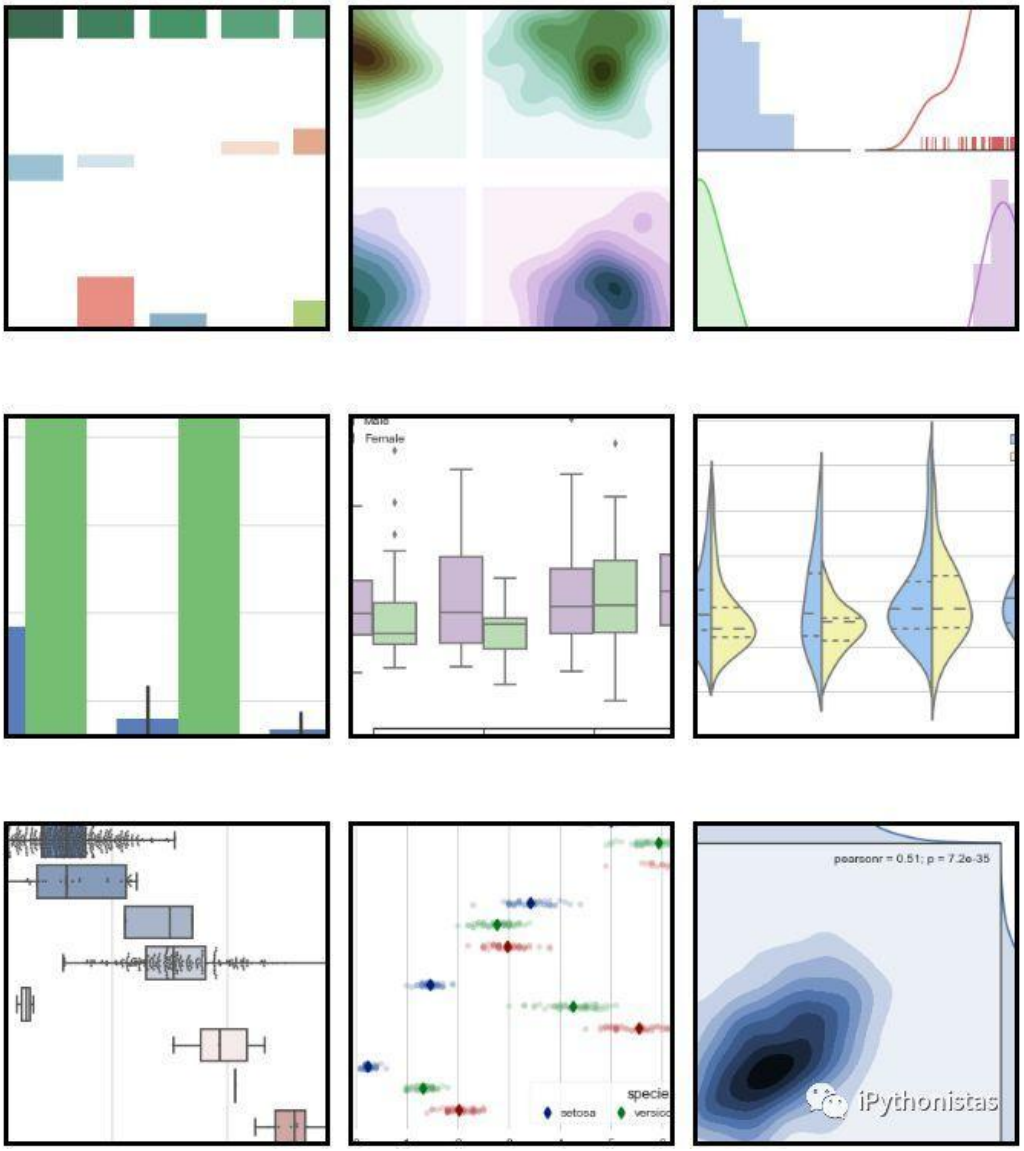
Python可视化 | Seaborn5分钟入门(二)——barplot和countplot

16-20 minutes

微信公众号：「Python读财」

如有问题或建议，请公众号留言

Seaborn是基于matplotlib的Python可视化库。 它提供了一个高级界面来绘制有吸引力的统计图形。Seaborn其实是在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，不需要经过大量的调整就能使你的图变得精致。**但应强调的是，应该把Seaborn视为matplotlib的补充，而不是替代物。**



注：所有代码均在IPython notebook中实现

---

barplot(条形图)

条形图表示数值变量与每个矩形高度的中心趋势的估计值，并使用误差线提供关于该估计值附近的不确定性的一些指示。具体用法如下：

> seaborn.barplot(x=None, y=None, hue=None, data=None, order=None, hue_order=None, estimator=(function mean), ci=95, n_boot=1000, units=None, orient=None, color=None, palette=None, saturation=0.75, errcolor='.26', errwidth=None, capsize=None, dodge=True, ax=None, **kwargs)

接下来还是通过具体例子学习里面的一些参数的用法：

```
%matplotlib inline
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
plt.rc("font",family="SimHei",size="12")  #用于解决中文显示不了的问题
sns.set_style("whitegrid")
```

本篇文章所采用的数据集内容如下

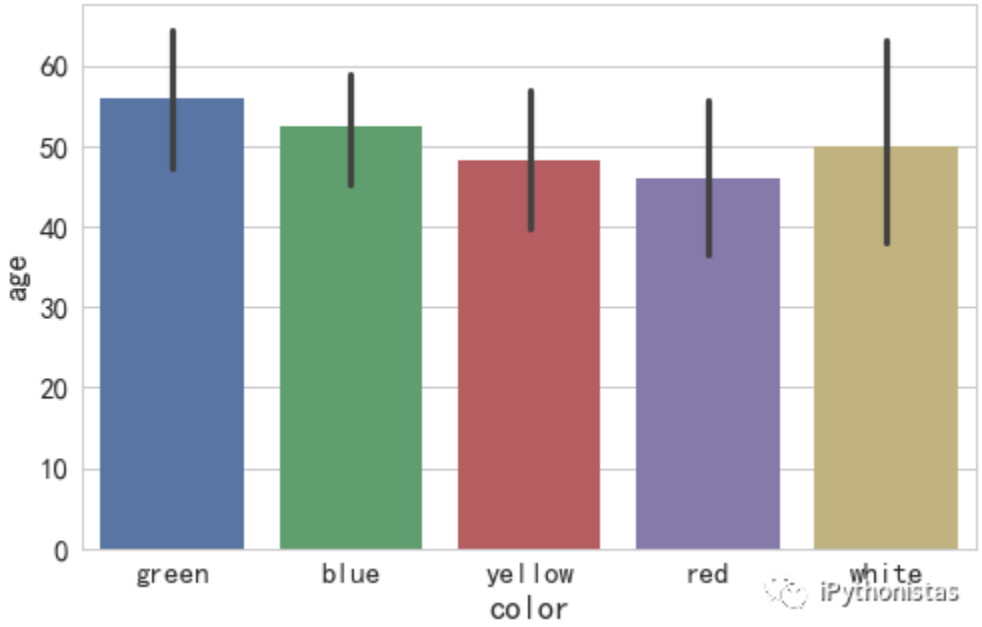
```
data.head(5)   #data是一个dataframe
```

	age	color	gender	height	smoker	weight
0	55	green	男	154	False	87
1	35	green	女	151	False	66
2	35	green	女	168	False	40
3	81	blue	男	180	False	41
4	45	blue	男	173	False	94

x，y（str）：dataframe中的列名

data：dataframe或者数组

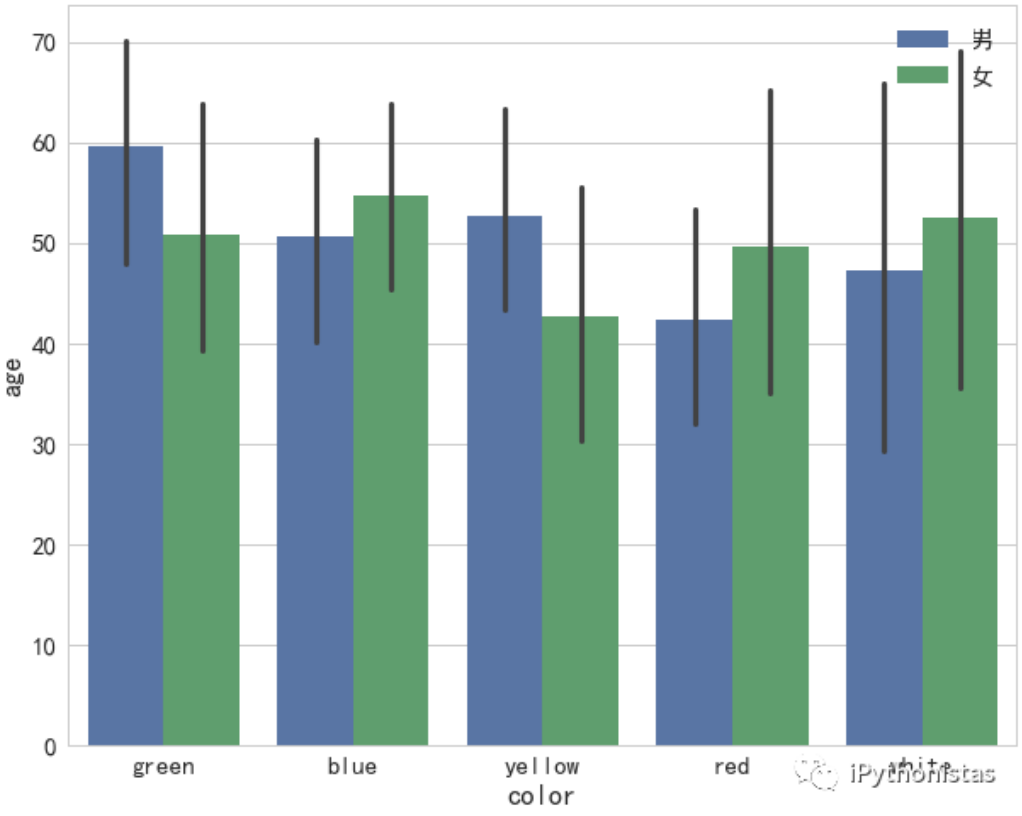
```
sns.barplot(x="color",y="age",data=data)
```



关于图像的解释：Seaborn会对”color“列中的数值**进行归类**后按照***estimator***参数的方法（默认为平均值）计算相应的值，**计算出来的值就作为条形图所显示的值**（条形图上的误差棒则表示**各类的数值相对于条形图所显示的值的误差**）

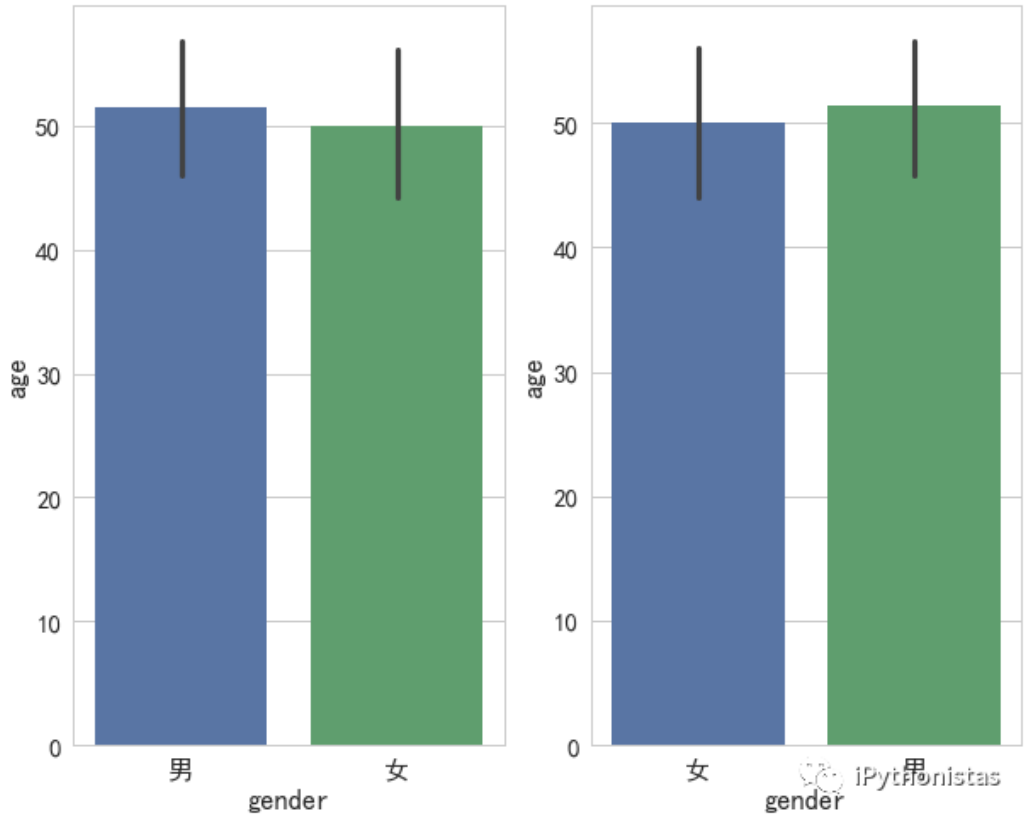
hue（str）：dataframe的列名，按照列名中的值分类形成分类的条形图

```
sns.barplot(x="color",y="age",data=data,hue="gender")
```



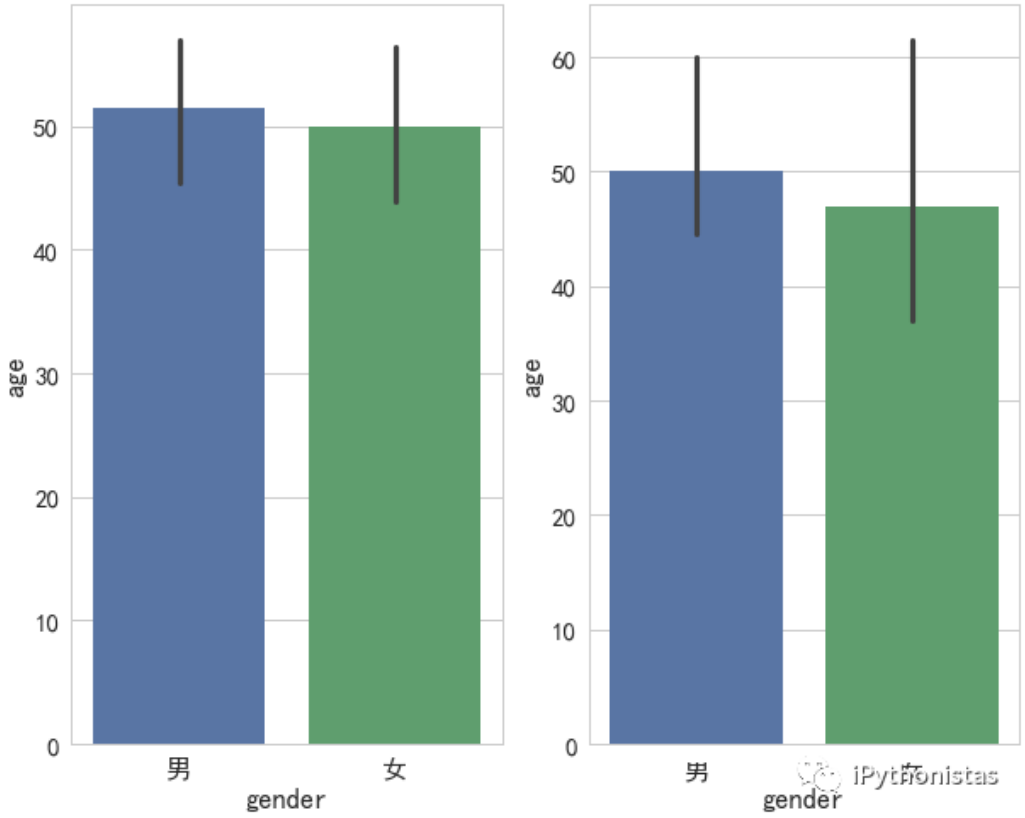
order, hue_order (lists of strings)：用于控制条形图的顺序

```
fig,axes=plt.subplots(1,2)
sns.barplot(x="gender",y="age",data=data,ax=axes[0])
sns.barplot(x="gender",y="age",data=data,ax=axes[1],order=["女","男"])
```



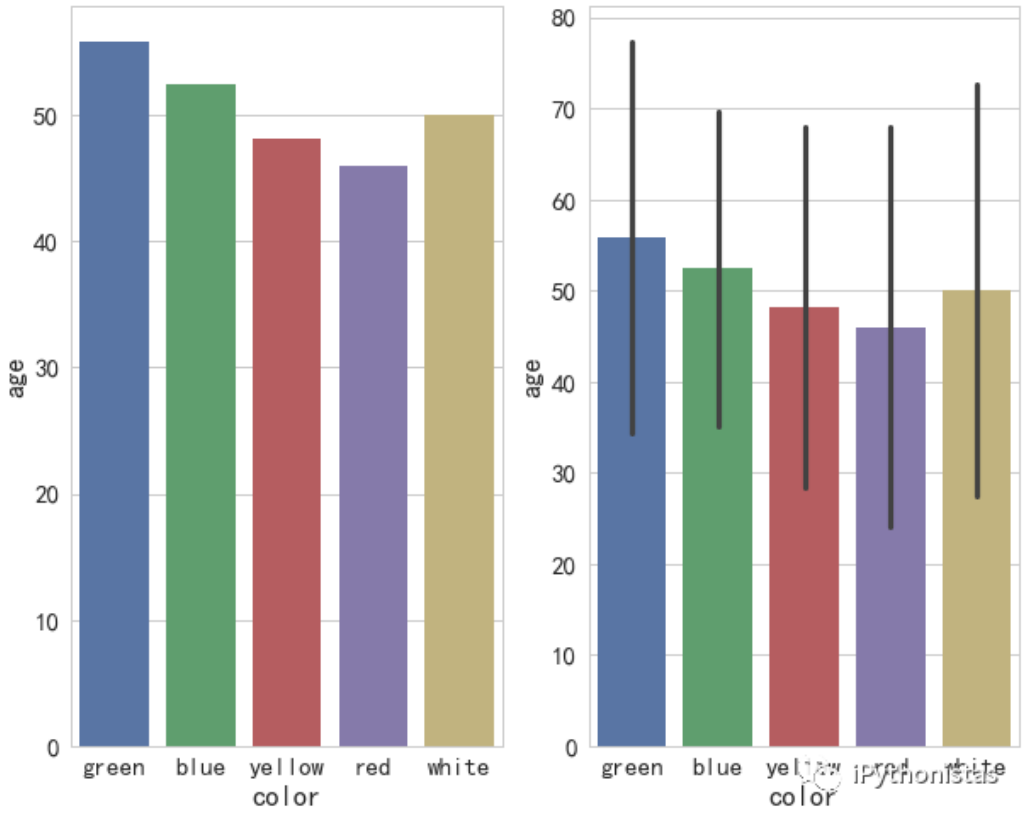
estimator:<function name>控制条形图的取整列数据的什么值

```
fig,axes=plt.subplots(1,2)
sns.barplot(x="gender",y="age",data=data,ax=axes[0])  #左图，默认为平均值
sns.barplot(x="gender",y="age",estimator=np.median,data=data,ax=axes[1])  #右图，
中位数
```



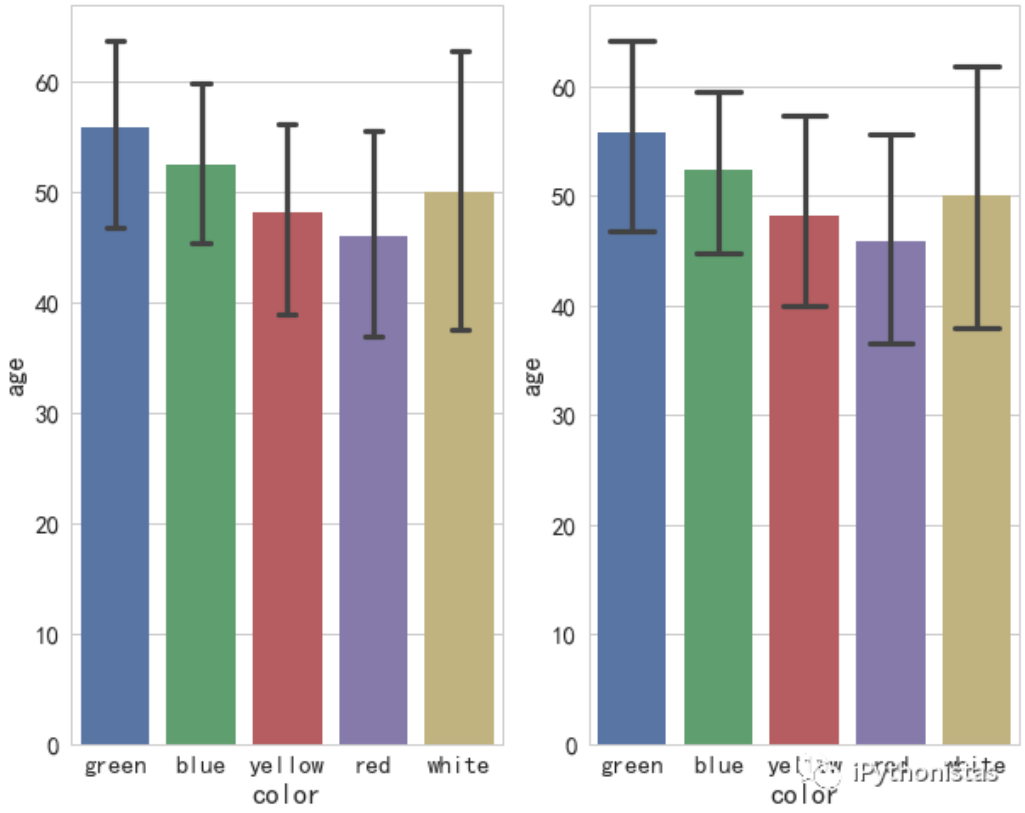
ci（float）：允许的**误差的范围**（控制误差棒的百分比，在0-100之间）,若填写"sd",则误差棒用标准误差。(默认为95)

```
fig,axes=plt.subplots(1,2)
sns.barplot(x="color",y="age",data=data,ci=0,ax=axes[0])  #左图
sns.barplot(x="color",y="age",data=data,ci="sd",ax=axes[1])  #右图
```



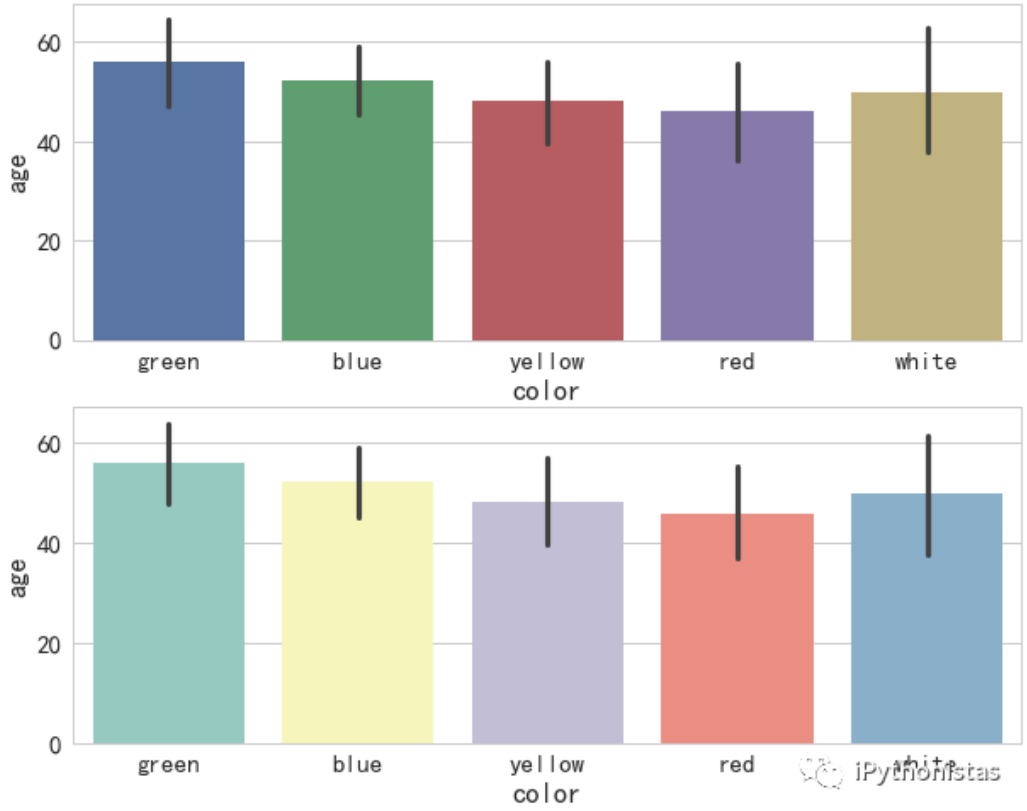
capsize（float）:设置误差棒帽条（上下两根横线）的宽度

```
fig,axes=plt.subplots(1,2)
sns.barplot(x="color",y="age",data=data,ax=axes[0],capsize=.2)  #左图
sns.barplot(x="color",y="age",data=data,ax=axes[1],capsize=.5)  #右图
```



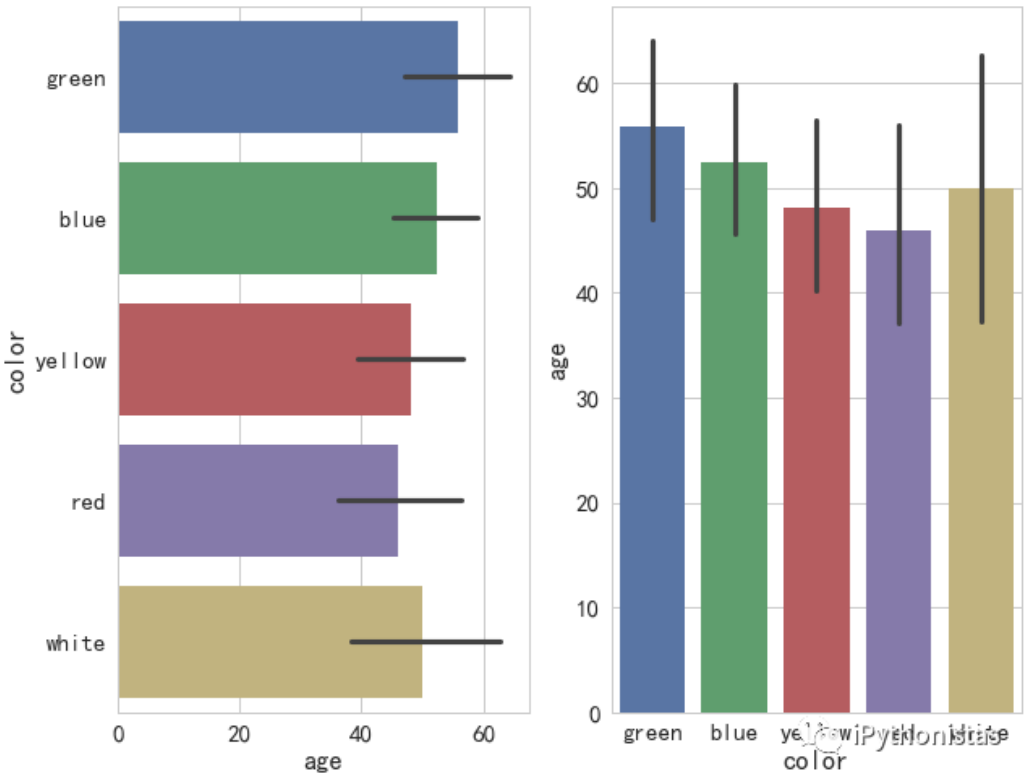
palette:调色板，控制不同的颜色style

```
fig,axes=plt.subplots(2,1)
sns.barplot(x="color",y="age",data=data,ax=axes[0])  #上图
sns.barplot(x="color",y="age",data=data,palette="Set3",ax=axes[1])  #下图
```



X,Y轴互换

```
fig,axes=plt.subplots(1,2)
sns.barplot(x="age",y="color",data=data,ax=axes[0])  #左图
sns.barplot(x="color",y="age",data=data,ax=axes[1])  #右图
```



countplot入门

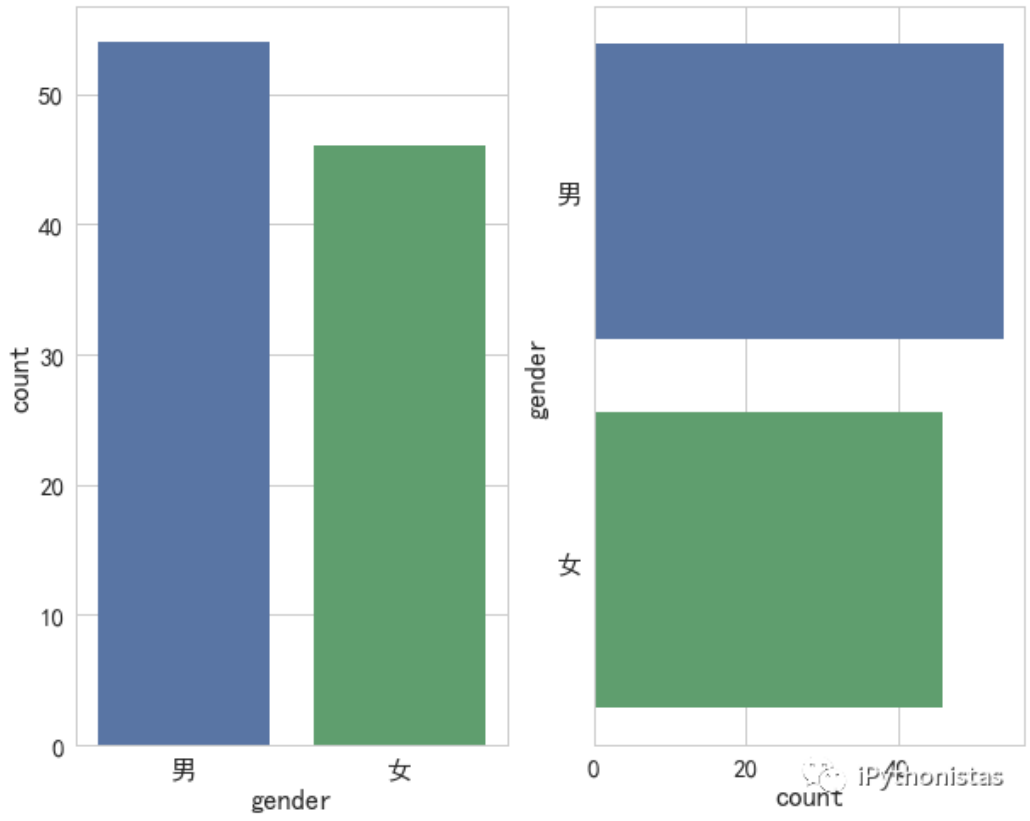
一个计数图可以被认为是一个**分类直方图**，而不是定量的变量。基本的api和选项与barplot（）相同，因此您可以比较嵌套变量中的计数。（工作原理就是**对输入的数据分类**，条形图显示**各个分类的数量**）具体用法如下：

> *seaborn.countplot(x=None, y=None, hue=None, data=None, order=None, hue_order=None, orient=None, color=None, palette=None, saturation=0.75, dodge=True, ax=None, **kwargs)*

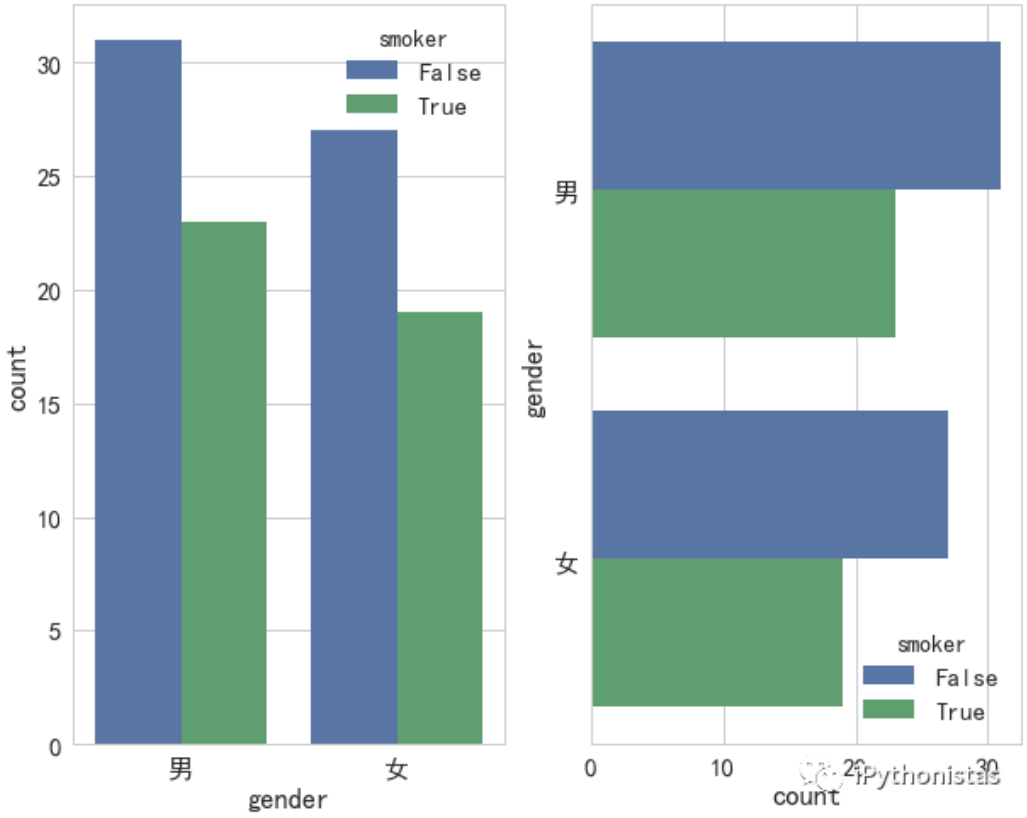
注：countplot参数和barplot基本差不多，可以对比着记忆，有一点不同的是countplot中不能同时输入x和y，且countplot没有误差棒。

根据例子体验一下：

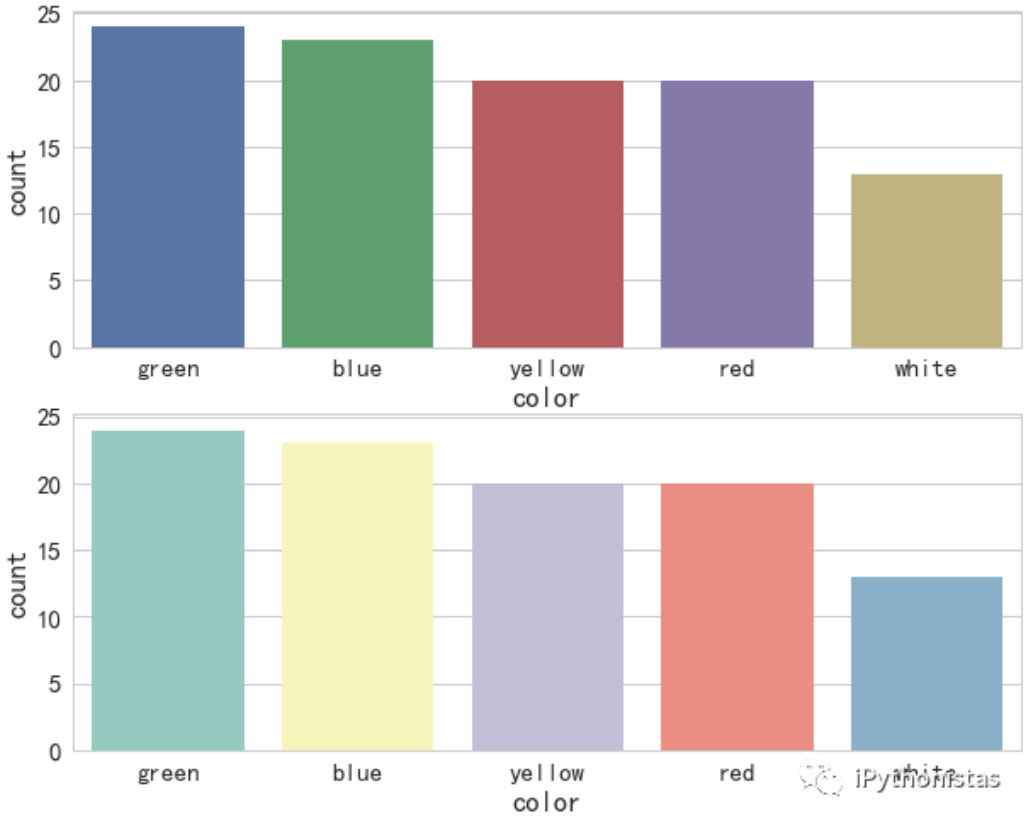
```
fig,axes=plt.subplots(1,2)
sns.countplot(x="gender",data=data,ax=axes[0]) #左图
sns.countplot(y="gender",data=data,ax=axes[1])  #右图
```




```
fig,axes=plt.subplots(1,2)
sns.countplot(x="gender",hue="smoker",data=data,ax=axes[0]) #左图
sns.countplot(y="gender",hue="smoker",data=data,ax=axes[1])  #右图
```




```
fig,axes=plt.subplots(2,1)
sns.countplot(x="color",data=data,ax=axes[0])  #上图
sns.countplot(x="color",data=data,palette="Set3",ax=axes[1])  #下图
```



---

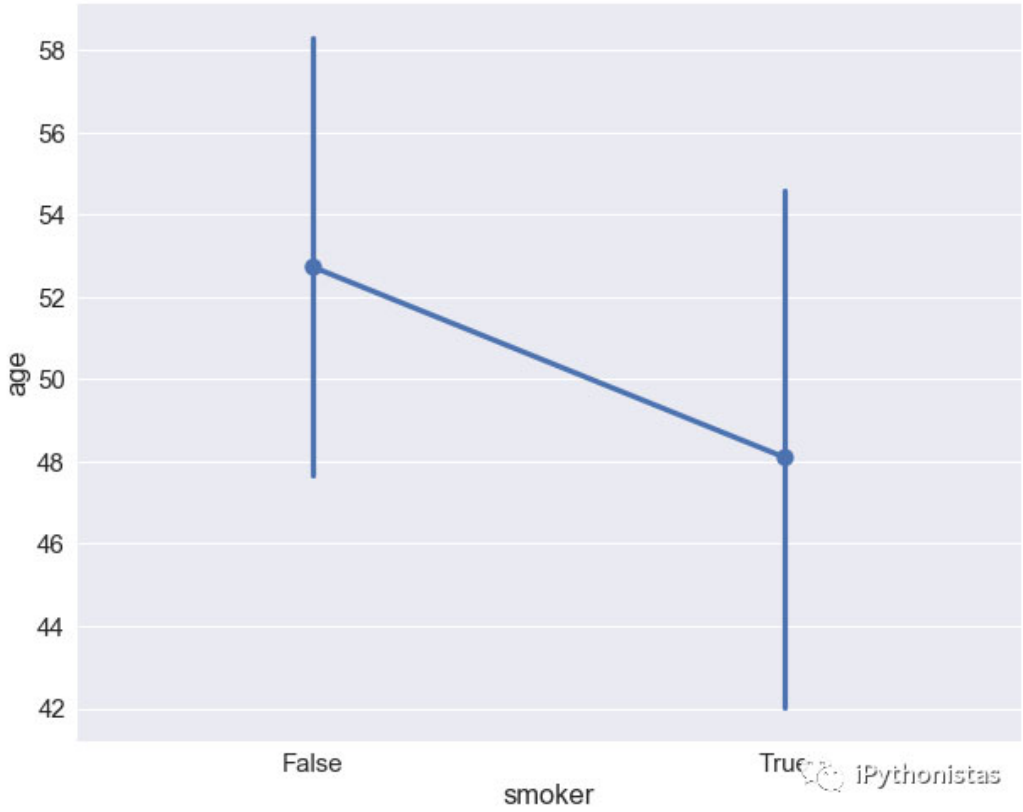
pointplot入门

点图代表散点图位置的数值变量的中心趋势估计，并使用**误差线**提供关于该估计的不确定性的一些指示。点图可能**比条形图更有用于聚焦一个或多个分类变量的不同级别之间的比较。**他们尤其善于表现交互作用：**一个分类变量的层次之间的关系如何在第二个分类变量的层次之间变化。**连接来自**相同色调等级**的每个点的线允许交互作用通过**斜率的差异**进行判断，这比对几组点或条的高度比较容易。具体用法如下：

> *seaborn.pointplot(x=None, y=None, hue=None, data=None, order=None, hue_order=None, estimator=(function mean), ci=95, n_boot=1000, units=None, markers='o', linestyles='-', dodge=False, join=True, scale=1, orient=None, color=None, palette=None, errwidth=None, capsize=None, ax=None, **kwargs)*

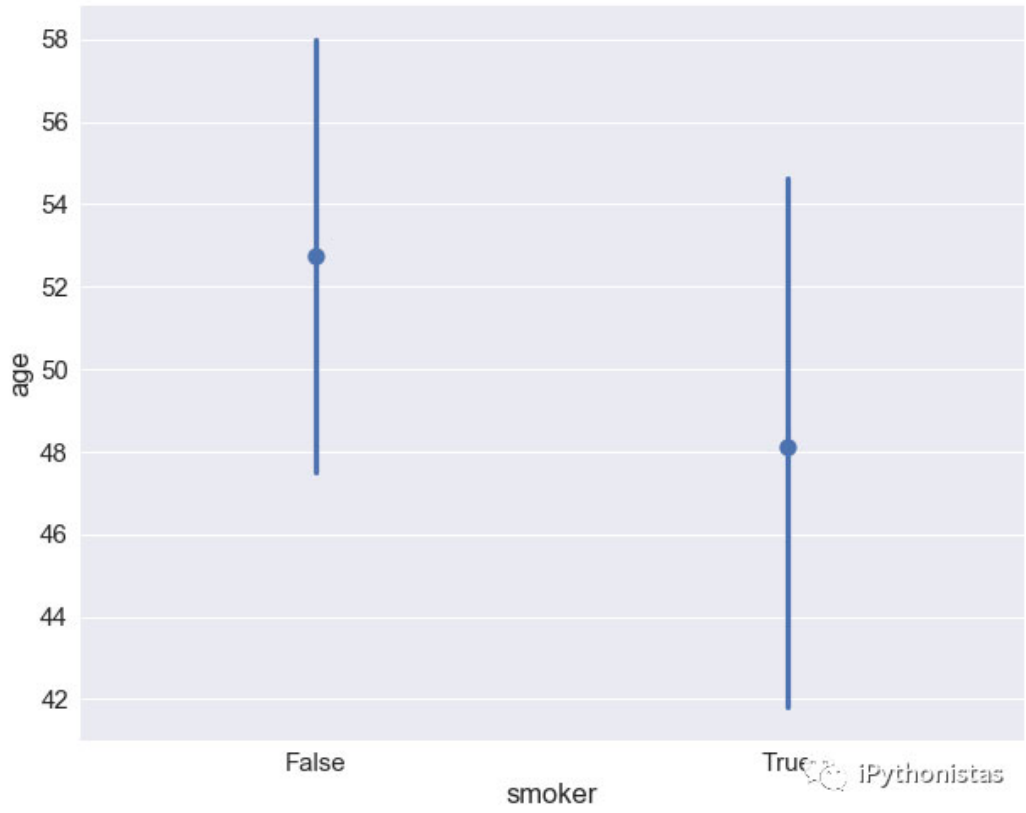
下面继续使用之前的数据集进行绘图，和barplot相同的参数就不再具体演示，重点演示pointplot独有的。

```
sns.set(font_scale) #初始化seaborn配置,并设置字体大小
sns.set_style("darkgrid") #灰色网格背景
sns.pointplot(x="smoker",y="age",data=data)
```



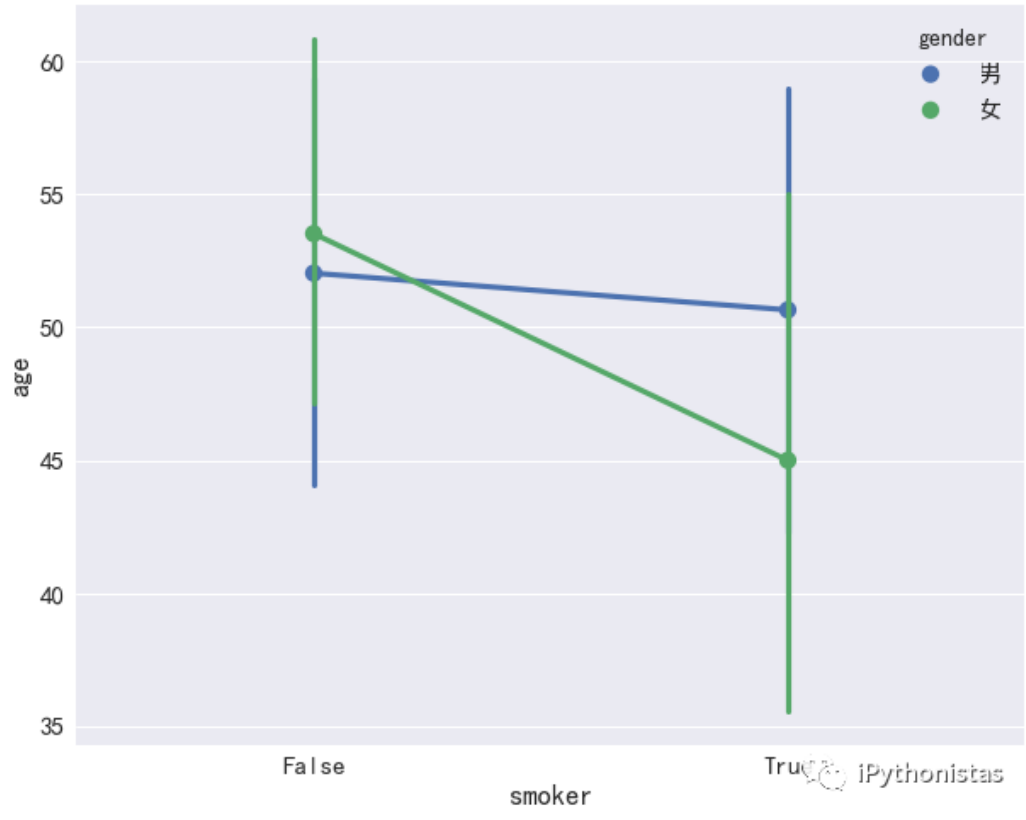
图中的点为这组数据的**平均值点**，竖线则为**误差棒**，默认两个均值点会相连接，若不想显示，可以通过***join***参数实现：

```
sns.pointplot(x="smoker",y="age",data=data,join=False)
```



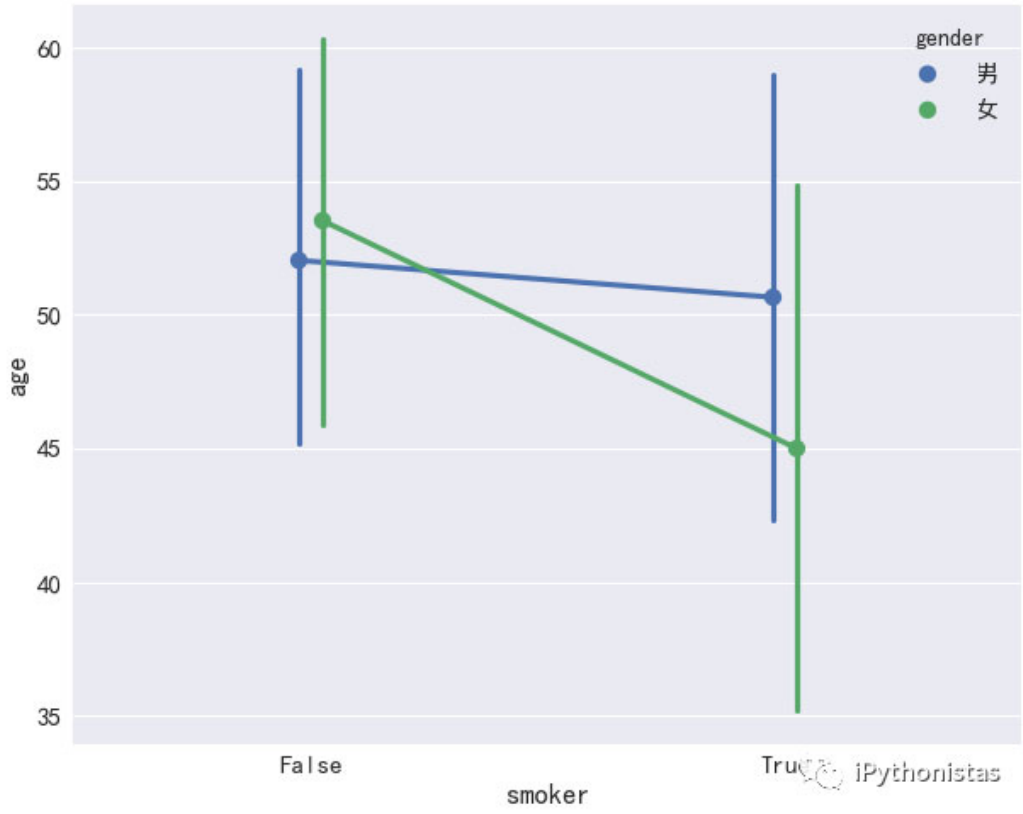
之前我们演示过barplot的hue参数，现在我们看一下pointplot的hue参数：

```
sns.pointplot(x="smoker",y="age",data=data,hue="gender")
```



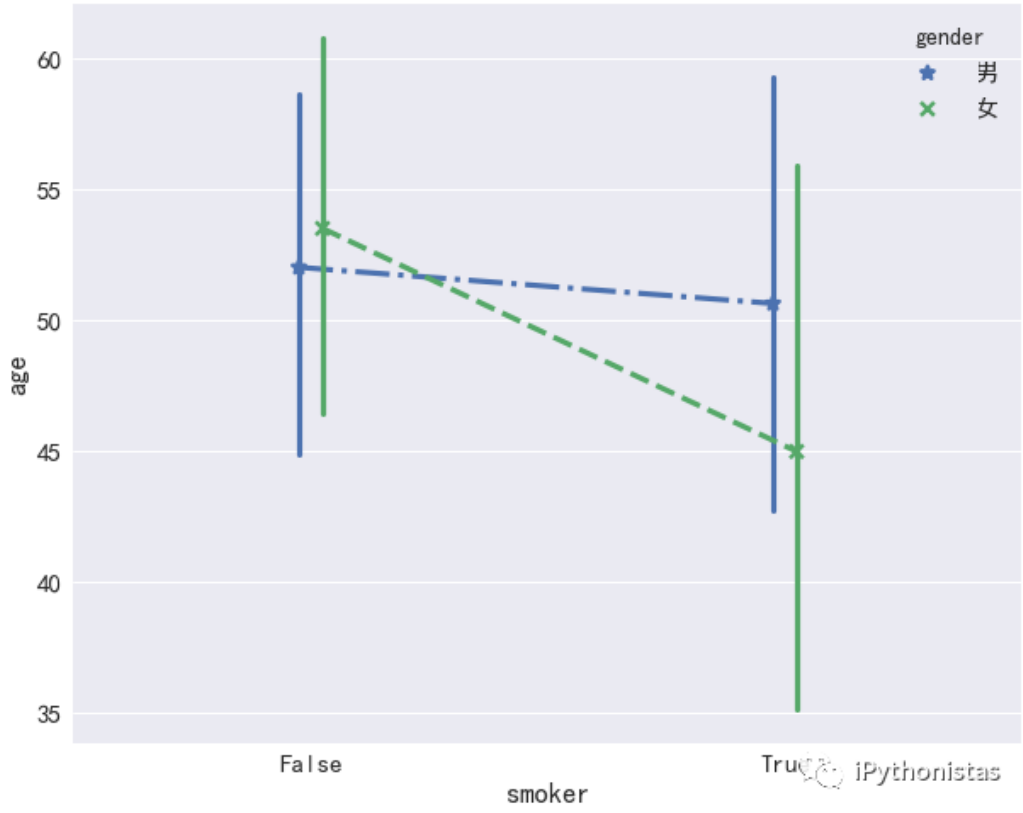
我们可以看到两个类别的误差棒重叠在了一起，使数据观测不清晰。怎么解决这个问题呢？pointplot的***dodge***参数可以使重叠的部分错开：

```
sns.pointplot(x="smoker",y="age",data=data,hue="gender",dodge=True)
```



接下来我们对均值点的样式（由参数***markers***控制）和相同色调的点之间的连线（由参数***linestyles***控制）做一下改动。

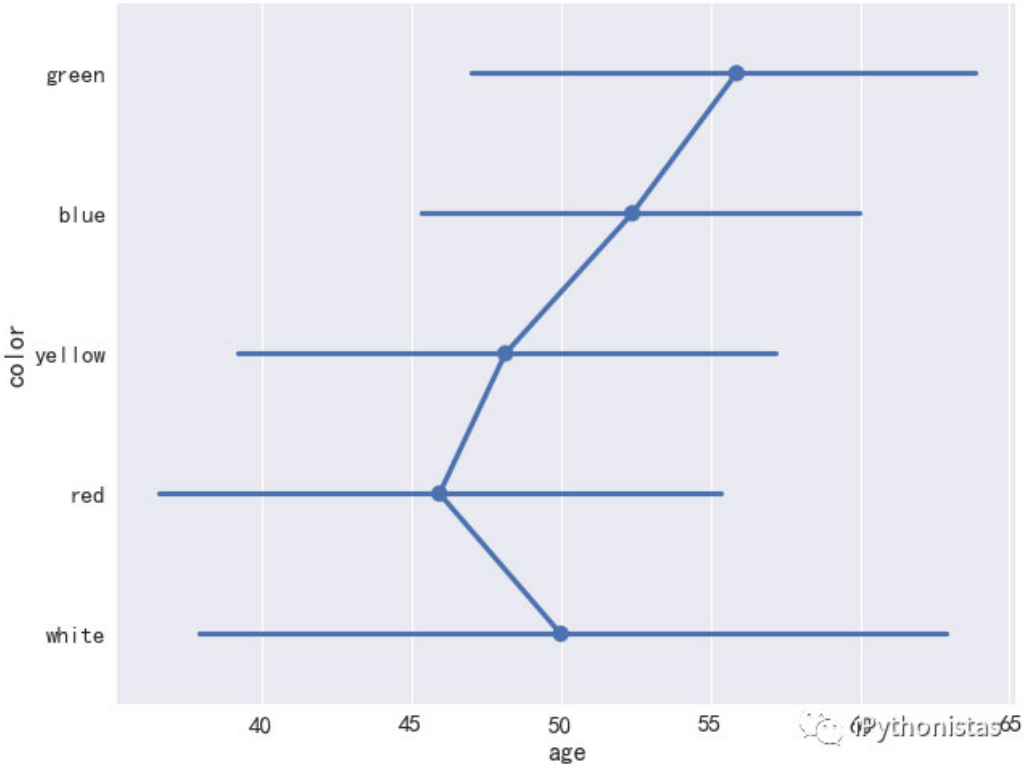
```
sns.pointplot(x="smoker",y="age",data=data,hue="gender",dodge=True,markers=
["*","x"],linestyles=["-.","--"])
```



其他样式请参考matplotlib线条样式

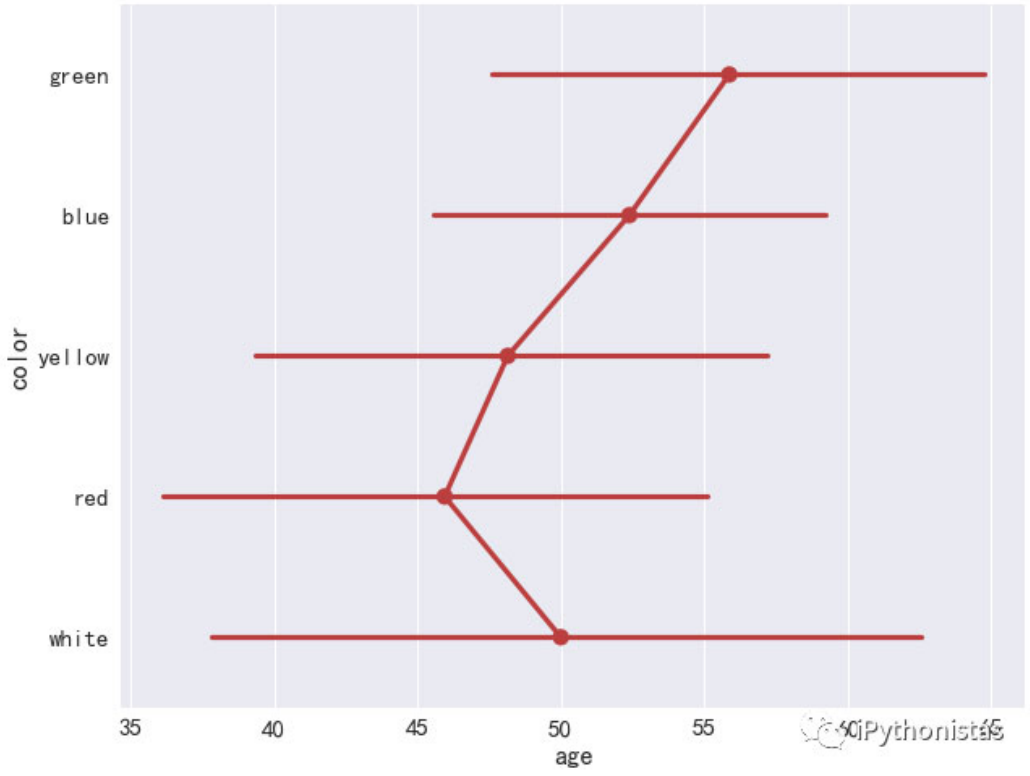
将X，Y轴互换

```
sns.pointplot(x="age",y="color",data=data)
```



通过***color***参数控制不同单层图的颜色

```
sns.pointplot(x="age",y="color",data=data,color="#bb3f3f")
```



还有其他效果和barplot一样的参数，大家可以动手自己试一下。以上内容是我结合官方文档和自己的一点理解写成的，有什么错误大家可以**指出来并提提意见**，**共同交流、进步**，也希望我写的这些能够给阅读完本文的你或或少的帮助！