## matplotlib常用命令

### matplotlib.pyplot.bar():

参数：

x：x轴

height：y轴，柱状图的高度

width：柱体宽度

bottom：柱体底部，实现柱体叠加功能

align：”center”或”edge”柱体在x轴刻度的右边或中间，需要在左边则通过改变width实现

其他参数：

color：柱体颜色，’blue’或’#000000’

edgecolor：柱体边缘颜色

linewidth：柱体边缘线宽

tick\_labelL

ecolor：标准差线的颜色

返回参数：

container：一个容器

### matplotlib.pyplot.plot()

### matplotlib.pyplot.pie()

参数：

x：数据

explode：分离显示，一个数组表示分离程度

labels：数据标签

colors：数组，数据的颜色

autopct：数据占的百分比

pvtdistance：百分比数字离圆心的距离

shadow：是否有阴影

labeldistance：数据标签离圆心的距离

startangle：开始的角度

radius：饼状图的半径

counterclock：顺时针绘制或者逆时针绘制

textprops：传递给文本对象的参数字典

center：饼状图的中心坐标

frame：如果为真，用图表绘制坐标轴框架。

rotatelabels：按照切线方向打印数据标签

wedgeprops：楔形属性设置wedgeprops=dict(width=size , edgecolor='w')

返回参数：

patches：列表，

texts：列表，标签列表

autotexts：列表

matplotlib.pyplot.hist()

直方图，连续型数据

参数：

x：输入数据

bins：列表，统计的区间分布

range：设置输入数据的最大和最小

denstity：为false时显示频数统计结果，为true时显示频率统计结果

weight：权重，归一化后乘权重得到条形的高度

cumulative：

bottom：底部

histtype：直方图类型

align：同上

orientation：’horizeontal’或’vertical’ 水平或竖直

rwidth：条形的宽度

log：y轴设置为对数刻度

color：颜色

label：

stacked：堆叠

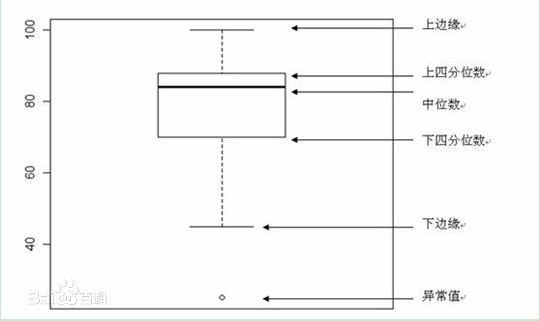
返回参数：

n：表示频率数组

bins：表示统计区间

patches：

### matplotlib.pyplot.boxplot()



参数：

x：输入数据

notch：改变箱型图的样式

sym：异常点的图标

vert：默认True表示竖直，false表示水平

whis：IQR，默认1.5，也可以设置区间比如[5 , 95]，代表强制上下边缘为数据95%和5%位置

patch\_artist：是否填充

bootstrap：

usermedians：设置中位数值

conf\_intercals：

positions：设置箱型图画的位置

width：设置箱型宽度，可以是序列或者浮点数

labels：标签

manage\_xticks：是否可以改变x轴

autorange：自动调整

meanline：设置为True且showmeans为True，显示平均值线

其他参数：

showcaps：显示上下边缘。默认为True显示

showbox：显示箱型，默认为True显示

showfliers：显示异常值，默认为True显示

showmeans：显示平均值，默认为False不显示

capprops：边缘线属性

boxprops：箱型线属性

whiskerprops：正中间那条线的属性

flierprops：异常点属性

medianprops：中位数线属性

meanprops：平均线属性

返回值：

result：字典类型，包含箱型，中位数，边缘线，中间那条线，异常点，平局数等信息。用matplotlib.line2D对象方式呈现，使用plt.getp()方法可以查看对象的具体参数

### 多子图的使用

plt.figure()

for i in range(1,10):

plt.subplot(330+i)

plt.scatter(team['winrate'],team[col[i]])

plt.title(col[i])

### 其他操作

plt.subplot(2 , 5 , 1) #设置一个2行5列的子图，并且选择第1个图

plt.xticks(fontsize=15) #设置刻度的字体大小

plt.show() #显示图像

plt.rc("font" , family="SimHei" , size="12") #设置中文字体

sns.set\_style("whitegrid") #设置风格

## Seaborn

一些通用的参数

Alpha：透明度

Hue：分类的属性

x、y：数据集变量名

Row、col ：更多的分类变量，平铺显示，变量名

Col\_wrap：每行的最高平铺数 整数类型

Estimator：在每个分类中进行矢量到标量的映射 矢量

Ci：置信区间 浮点数或None

n\_boot：计算置信区间时使用的引导迭代次数 整数

Units：采样单元的标识符，用于执行多级引导和重复测量设计 数据变量或向量数据

Order、hue\_order：对应排序列表 字符串列表

Kind：默认为point、bar：柱形图、count：频次、box：箱型图、violin：提琴图、strip：散点图、swarm：分散点、size:每个面的高度（标量）、aspect：纵横比（标量）、orient：方向（”v”/”h”）、color：颜色 、palette：调色板、seaborn颜色色板或字典、legend hue：信息面板（true/false）、legend\_out是否拓展图形、并将信息框绘制在中心右边（True/False）、share（x,y）共享轴线（True/False）

### 主题风格

Whtiegrid

darkgrid

While

Dark

ticks

### Sns.despine

### 调色板

设置默认的颜色

Current\_color = sns.color\_palette() 默认调色板，共有10个颜色

Sns.palplot(current\_color) 显示调色板

sns.color\_palette(‘hls’,12) 所有颜色值12等分作为默认颜色，颜色饱和度的颜色并绘制出来

Sns.hls\_palette(8,l=1,s=9) HLS格式的颜色8等分，亮度为1，饱和度为9

sns.color\_palette('Paired',20) 设置调色板为为一对对的，每对的颜色相近

Sns.boxplot(data=data,palette=current\_color) palette参数传入调色板

使用xkcd命名颜色

Sns.xkcd\_rgb[‘pale red’]

Sns.xkcd\_rgb[‘medium green’]

连续型的画板

sns.color\_palette('Blues',2) 把蓝色2等分，呈现渐变效果，默认为6等分

sns.color\_palette('Blues\_r',2) 加\_r，渐变效果反过来，越来越浅

线性色调变换

Sns.color\_palette(‘cubehelix’,8)

Sns.light\_palette(‘green’) 由白色到指定颜色的调色板，reserve是否反向

Sns.dark\_palette(‘green’) 由黑色到指定颜色的调色板，reserve是否反向

### 不同子图不同风格

With sns.axes\_style(“darkgrid”):

Plt.subpolt(211)

Sinplot()

Sns.set\_context(“paper”,font\_scale=5,rc={“line.linewidth”:2.5})

## 单变量分析

### distplot() 直方图

seaborn.distplot( a, bins=None, hist=True, kde=True, rug=False,

fit=None, hist\_kws=None, kde\_kws=None, rug\_kws=None,

fit\_kws=None, color=None, vertical=False, norm\_hist=False,

axlabel=None, label=None, ax=None)

a：序列，数据

bins：数值或序列，数值表示最大值和最小值之间划分的区间数，序列表示手动划分的区间

hist：布尔，是否调用直方图，默认为True

kde：布尔，是否调用核函数估计，默认为True

rug：布尔，是否生成观测数值的小细条（数值出现的位置）

fit：控制拟合的参数分布图形，能够直观地评估它与观察数据的对应关系(黑色线条为确定的分布)，需要引入scipy的分布变量

hist\_kws

kde\_kws

rug\_kws

fit\_kws

color

vertical

norm\_hist

axlabel

label

ax

### kdeplot() 核密度估计图

核密度估计(kernel density estimation)是在****概率论****中用来估计未知的**概率**密度函数****

seaborn.kdeplot(data , data2=None , shade=False ,

vertical=False , kernel='gau' , bw='scott' ,

gridsize=100 , cut=3 , clip=None , legend=True ,

cumulative=False , shade\_lowest=True , cbar=False ,

cbar\_ax=None , cbar\_kws=None , ax=None , \*kwargs)

data：序列，一维数据

data2：序列，一维数据，有传入值时绘制二维图，一般加上shade参数

shade：布尔，是否绘制阴影，若是，color参数设置颜色

vertical：false表示以x轴绘制，true表示以y轴绘制

kernel：

bw：

Gridsize：

cut：数值，线延坐标轴延伸的长度

clip：

legend：

cumulative：

shade\_lowest：

cbar：参数若为True，则会添加一个颜色棒(颜色帮在二元kde图像中才有)

cbar\_ax：

cbar\_kws：

ax：

### Sns.jointplot() 散点图加直方图

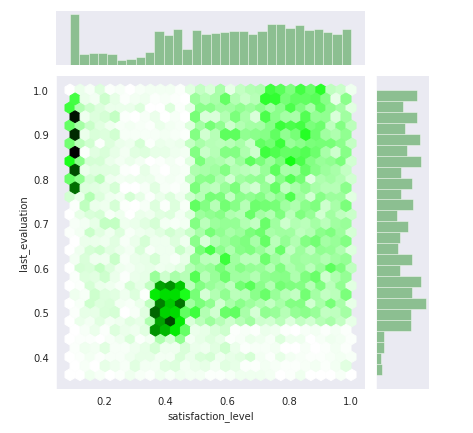
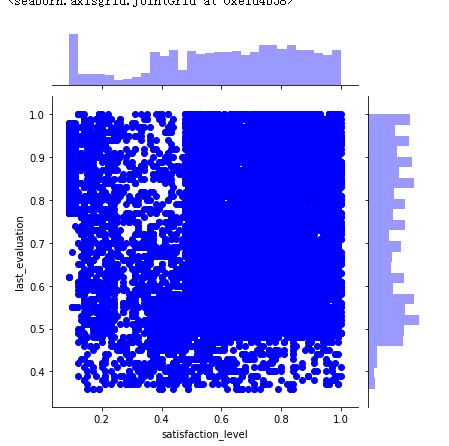
图一

sns.jointplot(x='satisfaction\_level',y='last\_evaluation',data=data,color='b')

图二

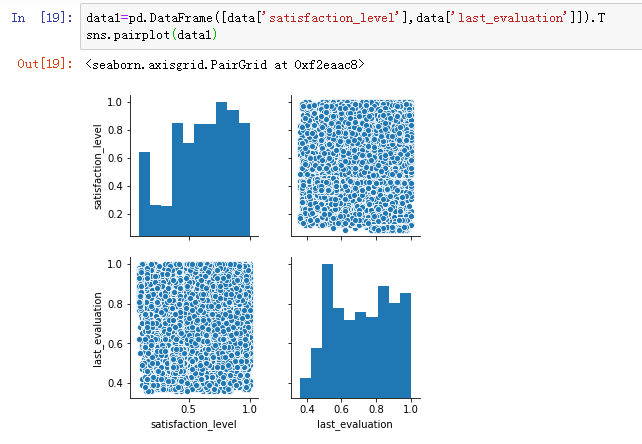
with sns.axes\_style('dark'):

sns.jointplot(x='satisfaction\_level',y='last\_evaluation',kind='hex',data=data,color='g')



### sns.pairplot(data)

对所有的特征两两配对做散点图，对角线做直方图



### Regplot()和implot()绘制回归图

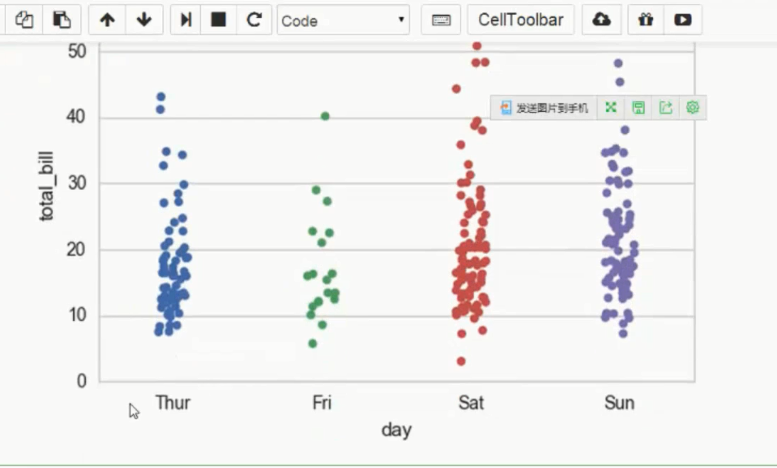
Regplot()比较简单，implot比较复杂

Sns.regplot(x=’hang’,y=’lie’,data=data,x\_jitter=0.5) hang和lie的关系

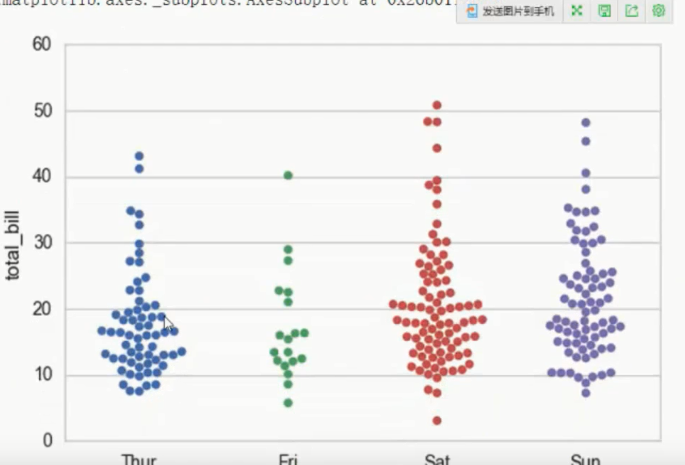
Sns.implot(x=’hang’,y=’lie’,data=data) 和regplot()一样，但是implot() 支持更高级的内容

对于分类的离散值，x\_jitter可以加一下偏移

### Sns.stripplot(x=’’,y=’’,data=data,jitter=True)



### Sns.swarmplot(x=’’,y=’’,data=data,hue=’sex’) hue属性列，不同属性显示不同的颜色



### Sns.boxplot(x=’’,y=’’,hue=’’,data=data)

### Sns.violinplot(x=’’,y=’’,hue=’’,data=data,split=True)

小提琴图，小提琴越胖的地方表示这个地方出现的值越多

Split=True，把不同属性值分开画

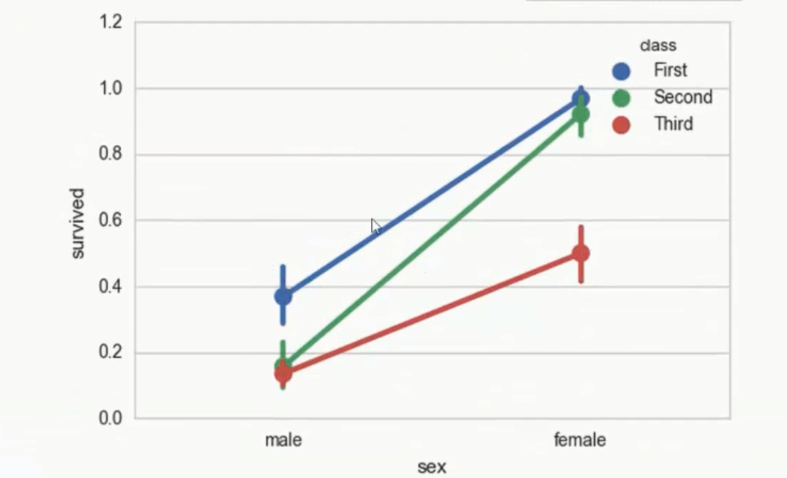
### sns.pointplot(x=’’,y=’’,hue=’’,data=data ，

Palette={‘male’:’g’ , ‘female’: ‘b’ }, #指定颜色

Markers=[‘ ^’ , ‘ o ‘], #指定标记符号

Linestyle=[‘-’ , ’--’]， #指定线型

Col=‘time’) #按time列平铺



### Sns.heatmap()热力图

Sns.heatmap(data=datra,

Vmin=0.5,vmax=1.0, #设置调色板的最大值和最小值

Center=0， #设置调色板的中心值

Annot=True, #显示数值

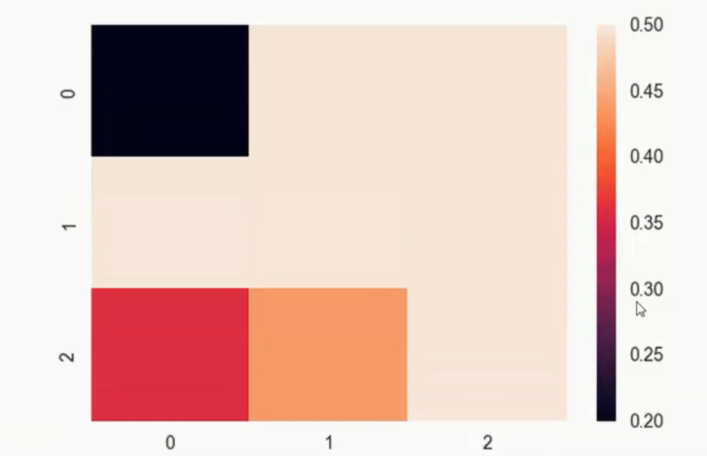
Fmt=’d’, #字体的默认格式

Linewidth=0.5，#格子之间的间距

Cmap=‘’， #设置调色板的颜色

Cbar=False， #隐藏调色板

)



### 条码图 sns.rugplot(x)

sns.kdeplot(x, shade=True, cut=-2) *# cut默认为3* sns.rugplot(x)

多子图的使用

fig, axes = plt.subplots(3, 3, figsize=(15,13), sharex=True)

axes = axes.flatten() #ax = ax.flatten()将ax由n\*m的Axes组展平成1\*nm的Axes组

for i,ax in enumerate(axes):

sns.regplot(x='winrate',y=col[i],data=team,x\_jitter=0.5,ax=ax)

plt.show()