  **Seaborn可视化库入门** 

—————

06/20 - Mindlake - 创建

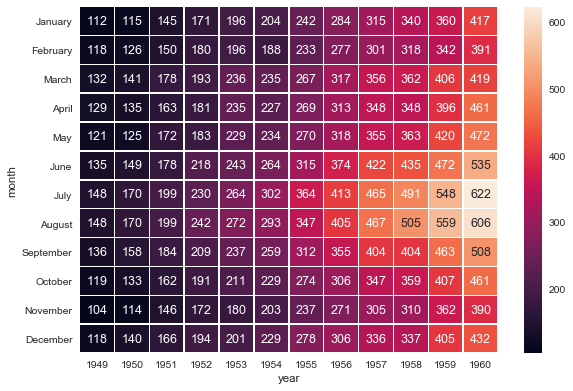
06/21 - Mindlake - 编辑完成

**简介**

很难想象，Seaborn的官网是那么朴实而低调，因为它把所有的美都奉献给了每个函数所绘制的可视化图形。于是我一改自己的惯例，把我最喜欢的一张Seaborn可视化图形，放在这里，表达对Seaborn这种行事风格的敬意。

Seaborn也是一个Python的可视化库，同时它又是建立在Matplotlib库基础上的。目前的版本是0.8.1。按照官网上的定义来说，“Seaborn能绘制出有吸引力的可视化图形”（Seaborn provides a high-level interface for drawing attractive statistical graphics.）

那么在这里我们就一起看一看Seaborn到底是怎么做到的。



**第0件事：准备**

[ 在我们让Seaborn真正提供可视化服务前，我们需要了解和准备的事情 ]

**- 需要了解的事情：**这篇入门教程，主要是为一些刚刚了解python，但又觉得Matplotlib库绘制的图形不够炫酷，在艺术上和美学上对可视化图形有更高要求的同学准备的。我会Step by Step的教大家怎么去写代码和调整代码。

**- 注意：**本文的所有代码需要在UDA课程中的项目工作区，或者Jupyter Notebook中运行。

**- 代码1：**import seaborn as sns

**- 代码说明：**这是首先且必备的代码，用来告诉python引入Seaborn包。

**- 代码2：**%matplotlib inline

**- 代码说明：**这是Jupyter Notebook的专用代码，用来告诉Jupyter Notebook把可视化图形绘制到工作区内。（没有这行代码，所有的输出都只是一串字符，不会有图像。）

**- 代码3：**sns.set(color\_codes=True , context='talk')

**- 代码说明：**set函数，告诉系统要以seaborn的方式绘图。后面这个context='talk'设定了‘talk’的背景模板，它会直接影响显示的字体大小。你也可以尝试其他几种模板：‘paper’, ‘notebook’和‘poster’。**修改时一定不要忘了引号。**

**- 数据来源：**所有数据均来源于我们的项目《5个城市PM2.5数据分析》，在使用下面的代码前，我假定了你已经知道如何获得**df\_all\_cities**这个变量的值。同时你需要使用下面的代码来删除**df\_all\_cities**这个变量的所有空值并赋值给一个新的变量df\_test。

**- 代码4：**df\_test=df\_all\_cities.dropna()

如果你想直接对**df\_all\_cities**动手，直接删除**df\_all\_cities**里面的空值，可以使用下面的代码：

**- 代码5：**df\_all\_cities.dropna(inplace=True)

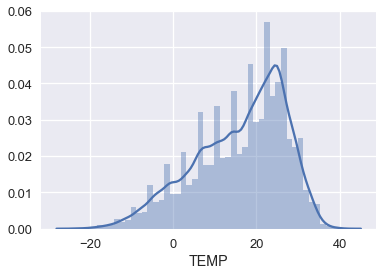
**第一件事：直方图**

[ 直方图用来表示不同分组的频率分布情况 ]

**- 函数：**sns.distplot(x,bins=None)

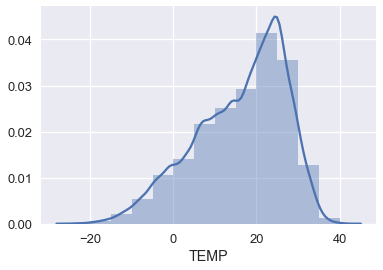
**- 函数说明：**x表示一个数组，一般是整形数组或者浮点数数组。bins代表分组标准，默认自动分组，但也可以指定分组。

**- 代码1：**sns.distplot(df\_test['TEMP'])



绘图如上，可以看出和Matplotlib中的hist()函数一样，自动对数据进行了分组，但是这个分组并不是按照整数（按照我们习惯的分类方式）进行的。我们想让它按照（-25，-20）、（-20，-15）这样，每5个单位分一组。如果你想这样，那么就必须指定bins的值。

**- 代码3：**sns.distplot(df\_test['TEMP'] , [-25,-20,-15,-10,-5,0,5,10,15,20,25,30,35,40,45])

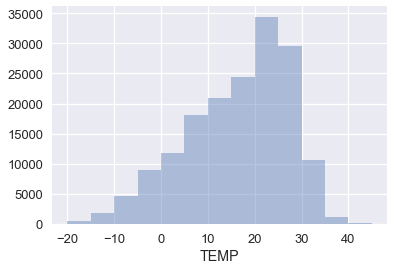


这下整齐好看了，但是这一长串的bins值，太长，而且非常容易计算错误。怎么办，我们使用range(-25,50,5)来生成这样一组数据。-25表示计算起点，50表示计算终点（但不包括50），5表示从计算起点到计算终点每次增加的值。于是我们刚才复杂的代码就简化成下面这个：

**- 代码4：**sns.distplot(df\_test['TEMP'] , range(-25,50,5))

此时的绘图效果和上面代码3的一样。可能大家一直有一个疑问：那个曲线是什么？围绕在直方图外围的曲线！这里需要简单给大家说下，这个叫KED核密度估计曲线。有兴趣的同学可以继续研究，没有兴趣的同学，也可以通过设置参数，不让这条曲线产生。

**- 代码5：**sns.distplot(df\_test['TEMP'] , range(-25,50,5) , kde=False)



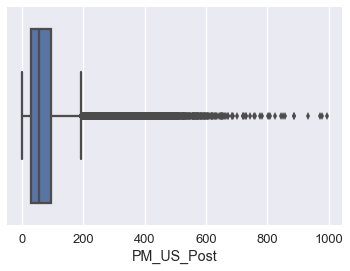
**- 上述直方图的解释：**从上面的图可以看出，20摄氏度到25摄氏度相对于其他的分组来说，占的比例最高。超过30摄氏度的分组，占的比例比较小，说明从整体上来说。高温天气比较少。

**第二件事：箱线图**

[ 箱线图用来，展示数据中值的分布情况，可以用箱线图来展示数据的最大值、最小值、四分位值（Q1，Q3）和中位数 ]

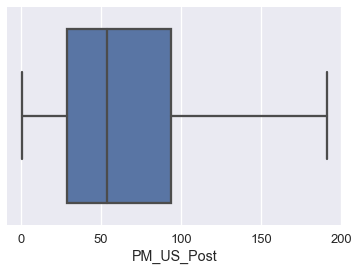
我们继续使用上面生成的df\_test变量进行演示。在Seaborn里，一次可以绘制多个箱线图（当然在Matplotlib中也可以，但Seaborn更优雅）。

**-代码1：**sns.boxplot('PM\_US\_Post',data=df\_test)



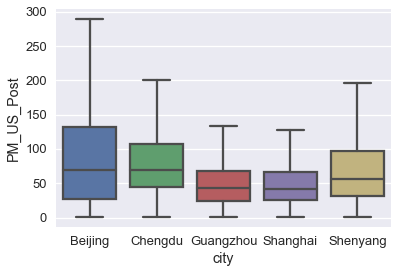
右侧的黑色线是什么？因为函数会默认将大于某个值的数据定为异常值，并在图中呈现出点的状态，所以你就看了一条黑线。如果你不想这样，想把所有的点都纳入箱线图内，不自动设定异常值，那么可以使用下面的函数参数：

**-代码2：**sns.boxplot('PM\_US\_Post',data=df\_test , showfliers=False)



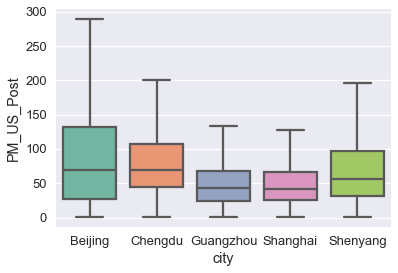
可以看到下方的黑线消失了，Seaborn隐藏了它。刚刚谁说，Seaborn能够绘制多条箱线图？让我们一起来试试看吧。

**-代码3：**sns.boxplot(x='city',y='PM\_US\_Post',data=df\_test , showfliers=False)



喏~一目了然，颜色样式什么的都已经设定好，如果你认为这种搭配你还不是很喜欢，可以在试下别的。不多说，上代码！

**-代码4：**sns.boxplot(x='city',y='PM\_US\_Post',data=df\_test , showfliers=False , palette='Set2')



怎么样？满意吗？还不满意！！！请修改'Set2'中的数字。

**第三件事：散点图**

[ 散点图用来，展示2组数据之间的关系，一般用于大量数据的情形 ]

**-函数：**sns.jointplot(x,y,data)

这里需要指定3个参数，x坐标代表的值类别，y坐标代表的值类别，和数据变量。

在开始之前，我们需要对df\_test这个变量再动下手脚。因为它的数据量太大了，先执行下面两行代码。

**-代码1：**df\_bj=df\_test[df\_test.cities=='Beijing']

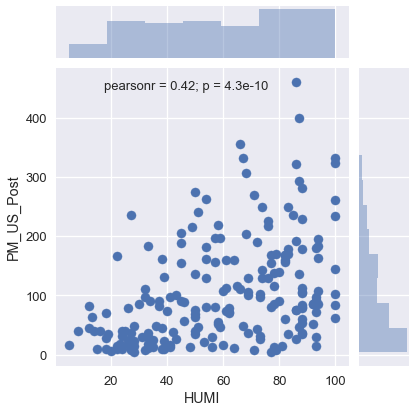
我们先将df\_test中属于北京的数据全部提取出来，然后保存到变量**df\_bj**中。

**-代码2：**df\_bj200=df\_bj.sample(200)

然后从df\_bj的数据集中使用sample()函数随机抽取200组数据。并保存到df\_bj200这个变量中，然后我们开始绘图啦~~~

**-代码3：**sns.jointplot(''HUMI', 'PM\_US\_Post',data=df\_bj200)

我们首先指定x和y参数分别代表的数据列，这里x代表湿度数据列（HUMI）,y代表PM2.5值的数据列（PM\_US\_Post），然后把数据变量df\_bj200赋值给data。DUANG！！

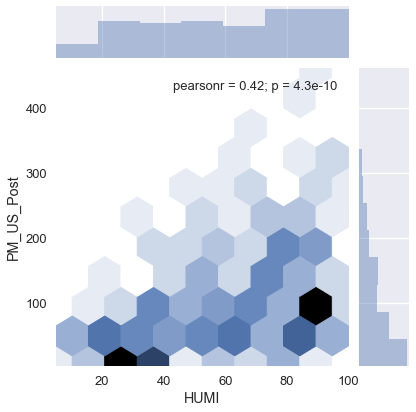


怎么样，是不是非常高大尚。不仅仅有散点图，而且还会有直方图在上方和右侧呈现。同时给你计算出了皮尔斯相关系数0.42。说明在这个样本中湿度和PM2.5的值有正相关（**一定要注意，仅仅是在这个样本中是这样的，而并不会因为它是随机取样，就想当然的认为，它代表了整体的情况。这里有疑惑的同学或者愿意探索为什么的同学，请在《数据分析》正式的课程中进行学习。**）

可能有同学觉得不就是增加了直方图而已，可视化图形本身并没有什么本质变化，那你就尝试一下下面的代码：

**-代码4：**sns.jointplot(''HUMI' , 'PM\_US\_Post' , data=df\_bj200 , kind='hex')

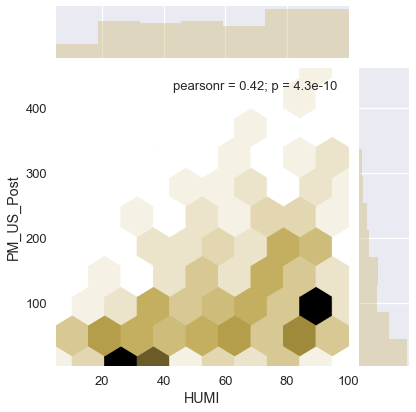
增加了一个kind='hex'，马上DUANG！！DUANG！！



是不是有点意思了，这里的每个六边形的蜂巢颜色代表数据的密度。是不是可以变化下颜色，蜂巢是黄色的好像？？

**-代码4：**sns.jointplot(''HUMI' , 'PM\_US\_Post' , data=df\_bj200 , kind='hex' ,color='y' )

再增加了一个color='y' ，马上DUANG！！DUANG！！DUANG！！



当然你也可以尝试一下其他的颜色。如红色：color='r'，绿色：color='g'

**好了，以上就是给大家简单介绍几个Seaborn函数的绘图方法，其实无论是matplotlib包还是Seaborn包，都只是在数据分析的最后，将结果进行的某种呈现。这种呈现的基础则是数据分析前面几个环节的结果----数据！只有好的数据才能带来有意义和有价值的呈现，所以数据分析需要扎实的统计学功底和熟练的python操作，对数据的处理技巧也不是朝夕之功。如果大家打算在数据分析这个热门领域中，有所收获，个人还是建议大家扎扎实实进行一番学习。**

- Seanborn(0.8.1) -