# 深度学习库

Python被大量应用在数据挖掘和深度学习领域，其中使用极其广泛的是：Numpy、pandas、PTL、Matplotid等库。

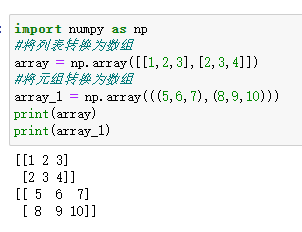
* **numpy**是Python科学计算库的基础。包含了强大的N维数组对象和向 量运算。
* **pandas**是建立在numpy基础上的高效数据分析处理库，是Python的重要数据分析库。
* **Matplotlib**是一个主要用于绘制二维图形的Python库。用途：绘图、可视化。
* **PIL**库是一个具有强大图像处理能力的第三方库。用途：图像处理。

## Numpy基础

Numpy数组中的所有元素类型必须是一致的（要么全字符串，要么全数字）。

### 1.1 array()函数

创建数组最简单的方法就是使用array函数。它接受一切序列类型的对象（包括其他数组），然后含有传入数据的numpy数组。其中，嵌套序列（比如有一组等长列表组成的列表）将会被转换为一个多位数组。



### 1.2 创建具有初始占位符内容的数组

* zeros():可以创建指定长度或者形状的全0数组



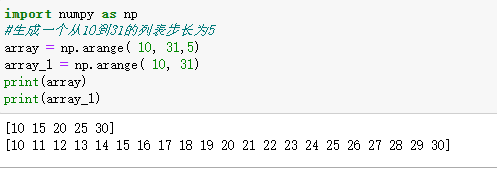
* ones():可以创建指定长度或者形状的全1数组



### 1.3 arange ()函数

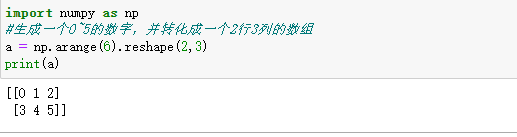
为了创建数字组成的数组，NumPy提供了一个类似于range的函数，该函数返回数组而不是列表

格式：np.arange(开始,结束,步长) 包括开始数，不包括结束数（左闭右开）

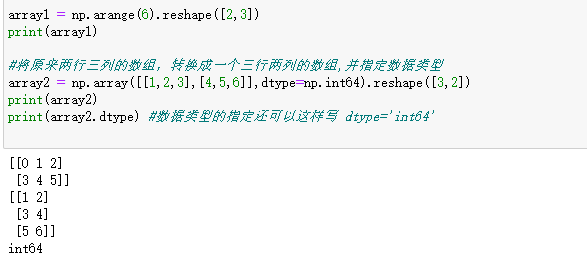


### 1.4 reshape()函数

* numpy.arange(n).reshape(a, b)依次生成n个自然数，并且以a行b列的数组形式显示。

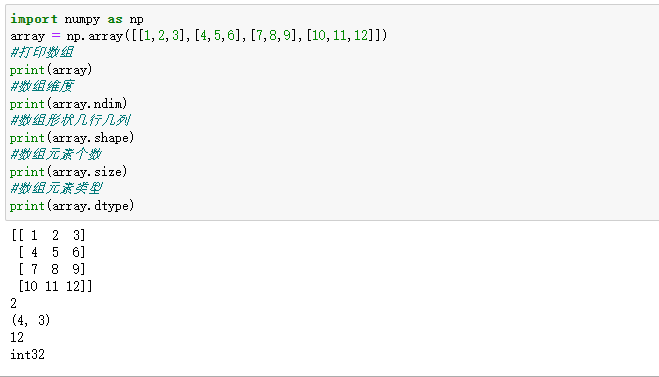


* 重新定义数组的形状



### 1.5 输出数组的一些信息

* ndim 输出数组的维度
* shape 输出数组的形状，几行几列
* size 输出数组中的元素个数
* dtype 输出数组的类型

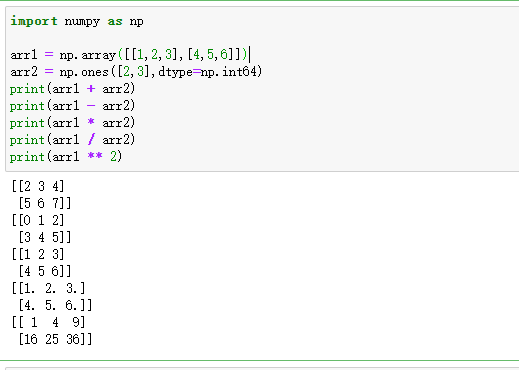


### 1.6 数组的计算

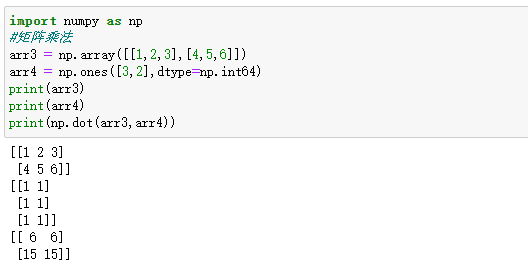
数组很重要，因为它可以使我们不用编写循环即可对数据执行批量运算。这通常叫做矢量化（vectorization）。

**大小相等的数组之间的任何算术运算都会将运算应用到元素级**。同样，数组与标量的算术运算也会将那个标量值传播到各个元素.

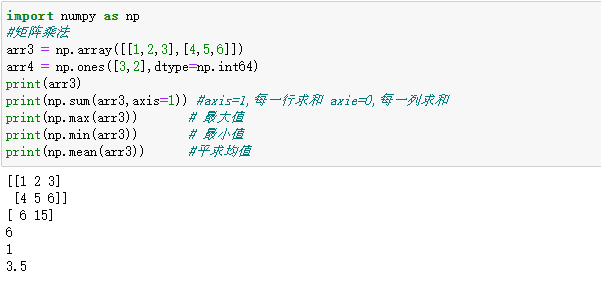
* 矩阵的基础运算：前提同类型矩阵



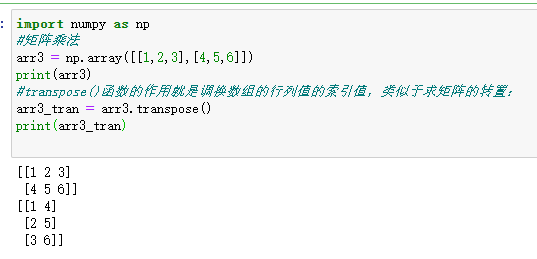
* 矩阵的乘法：dot()函数



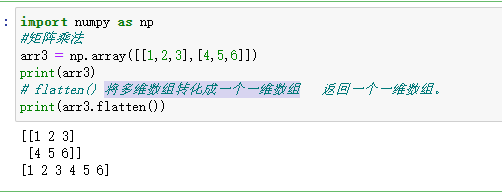
* 矩阵的其他运算：sum()、max()、min()、 mean()



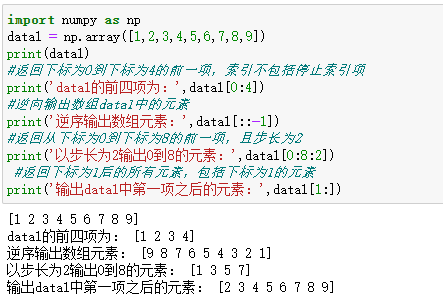
* 矩阵的转置：transport()函数

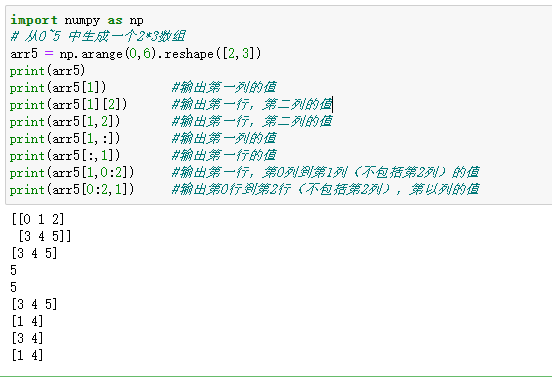


* 将多维数组转化成一个一维数组：flatten()函数



### 数组的索引与切片



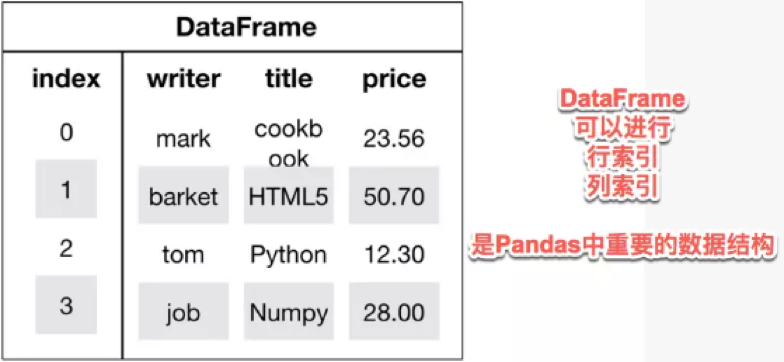


## pandas库

pandas是python第三方库，提供高性能易用数据类型和分析工具。

pandas基于numpy实现，常与numpy和matplotlib一同使用

### 1. DataFrame

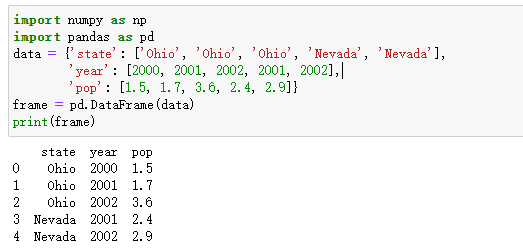


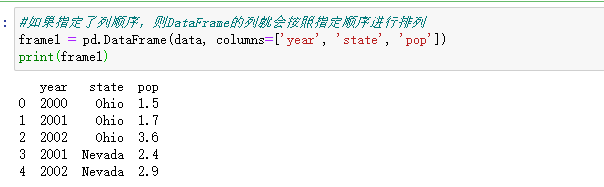
DataFrame是一个表格型的数据结构，类似于Excel或sql表

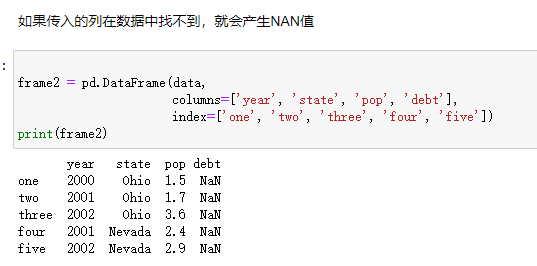
它含有一组有序的列，每列可以是不同的值类型（数值、字符串、布尔值等）

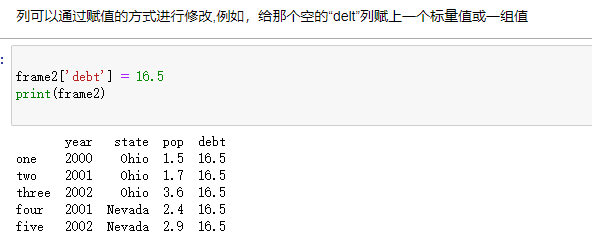
DataFrame既有行索引也有列索引，它可以被看做由Series组成的字典（共用同一个索引）。

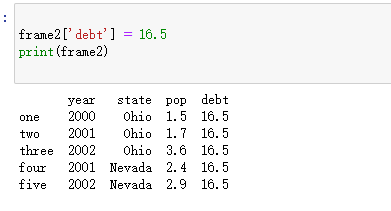
用多维数组字典、列表字典生成 DataFrame。

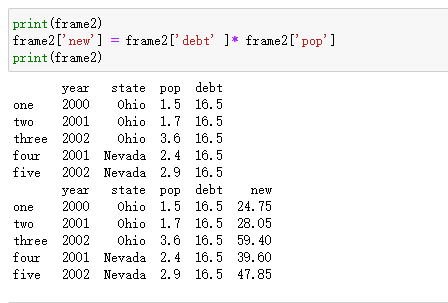


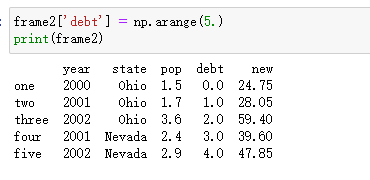












## Matplotlib库

Matplotlib库由各种可视化类构成，内部结构复杂。

matplotlib.pylot是绘制各类可视化图形的命令字库

### 3.1 plot()函数

格式：**plt.plot(x,y,\*\*kwargs)**

x:x轴数据

y:y轴数据

\*\*kwargs的参数大致有如下几种：

color： 颜色

linestyle： 线条样式

linewidth:线条大小

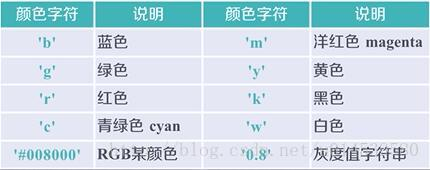
marker： 标记风格

markerfacecolor： 标记颜色

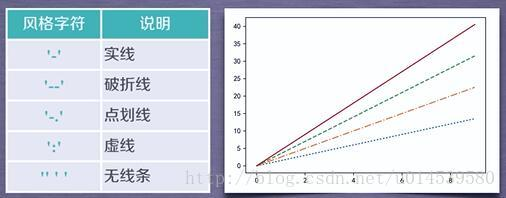
markersize： 标记大小

备注：

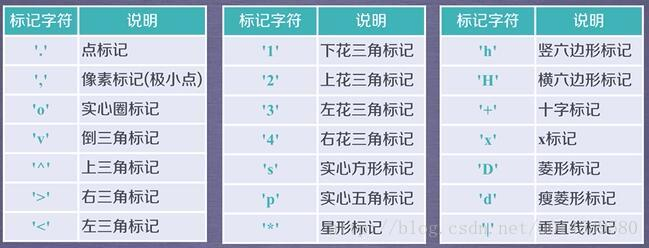
color =



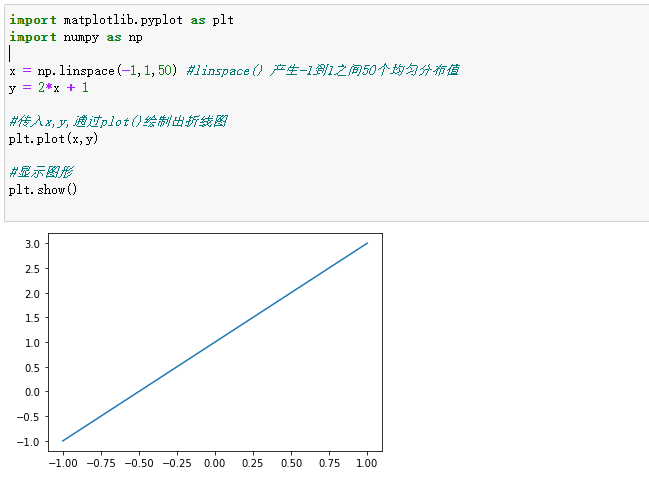
linestyle =（linestyle = 'none' 或 linestyle = ' ' 表示没有线条）



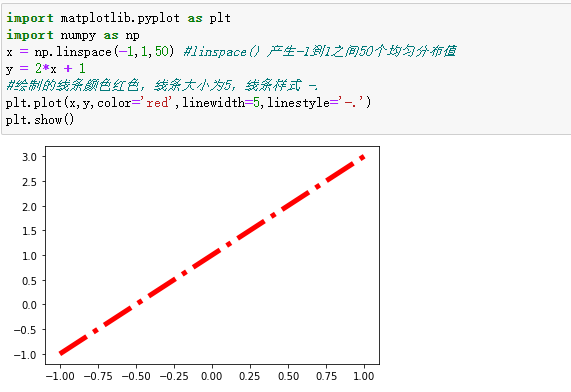
 marker =



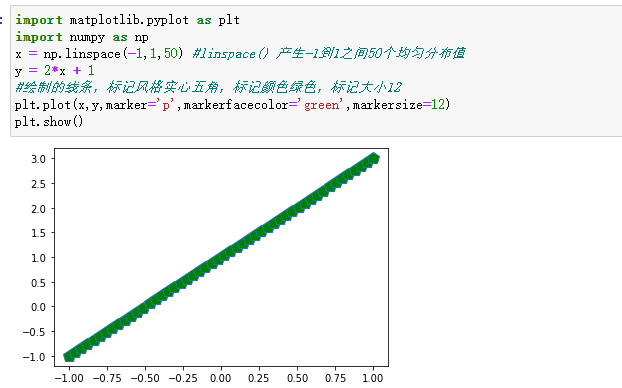
实例一



实例二



实例三

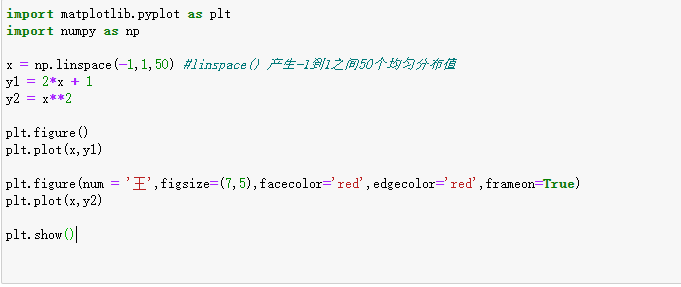


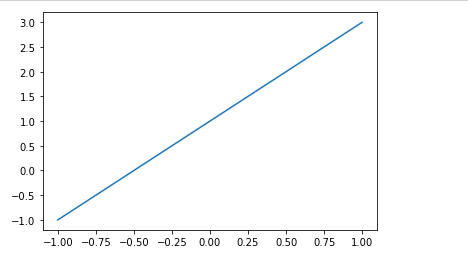
### 3.2 figure() 函数

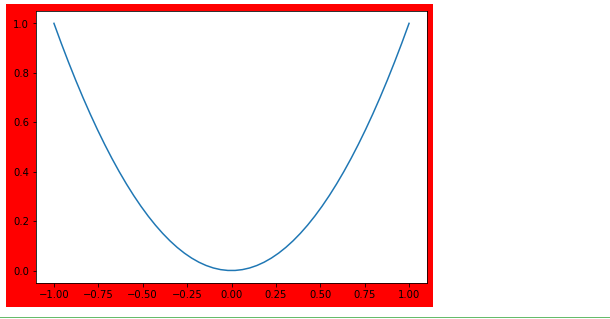
作用：设置绘制的图像属性值

figure(num=None, figsize=None, dpi=None, facecolor=None, edgecolor=None, frameon=True)

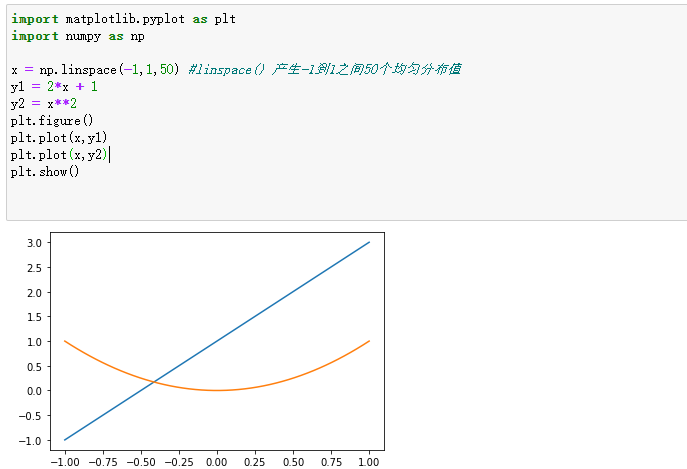
* num:图像编号或名称，数字为编号 ，字符串为名称
* figsize:指定figure的宽和高，单位为英寸；
* dpi参数指定绘图对象的分辨率，即每英寸多少个像素，缺省值为80 ，1英寸等于2.5cm,A4纸是 21\*30cm的纸张
* facecolor:背景颜色
* edgecolor:边框颜色
* frameon:是否显示边框







如果值在一个plot前添加一个figure()或者不写。如果有多个函数，会在一个画布上绘制。（如果每个函数想单独生成一个图像，则应该每个函数绘制之前使用figure()）

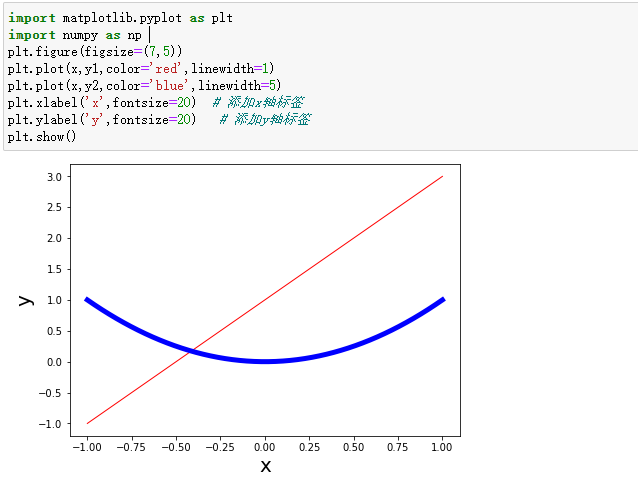


### 3.3 xlabel()与ylabel()

xlabel()与ylabel()主要是为x轴与y轴添加标签。

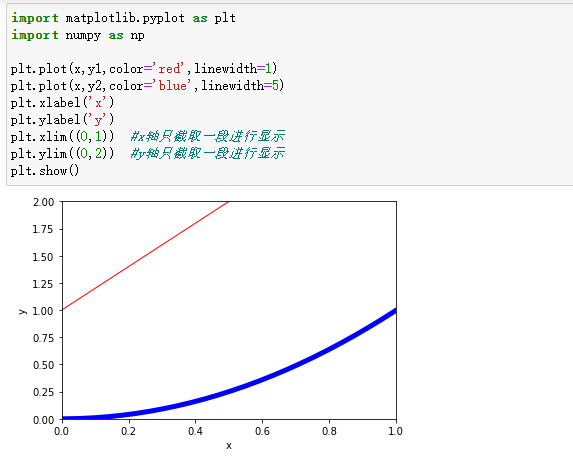
用法xlabel('标签名称',fontsize=标签大小)

例：



### 3.4 xlim()与ylim()函数

作用在x或y轴上截取一段进行显示。



### 3.5 legend()函数

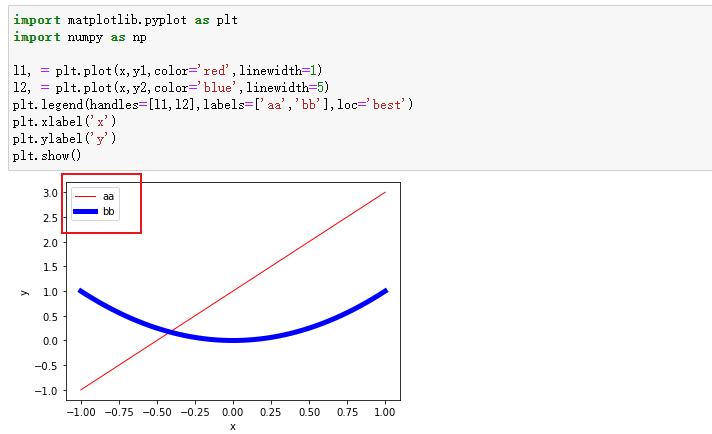
对于数据可视化而言，为了能够让可视化结果更加清晰直观，常常会对视图中的不同数据进行标注，也就是图例展示。告知用户，可视化结果中呈现的线条、图样等，用以指代何种数据。

plt.legend(handles=[l1,l2],labels=['aa','bb'],loc='best')

handles：指向的图图形

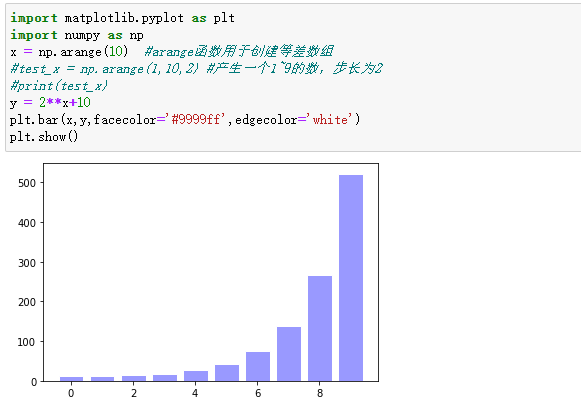
labels:图形说明的信息

loc：说明信息出现的位置



### 3.6 bar()函数

bar()函数用于生成柱状图，参数同plot()函数。



### 3.7 scatter()函数

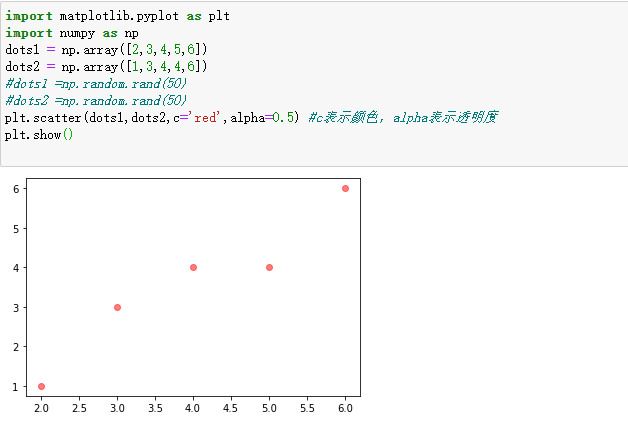
格式：plt. Scatter(x,y,c=’’, alpha = 数字)

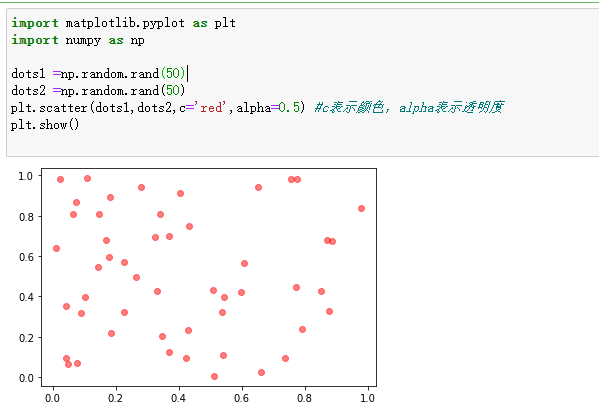
x:x轴

y:y轴

c:颜色

alpha：透明度





扩展：random()函数

random() 方法返回随机生成的一个实数，它在[0,1)范围内。

是不能直接访问的，需要导入 random 模块，然后通过 random或者rand 静态对象调用该方法。

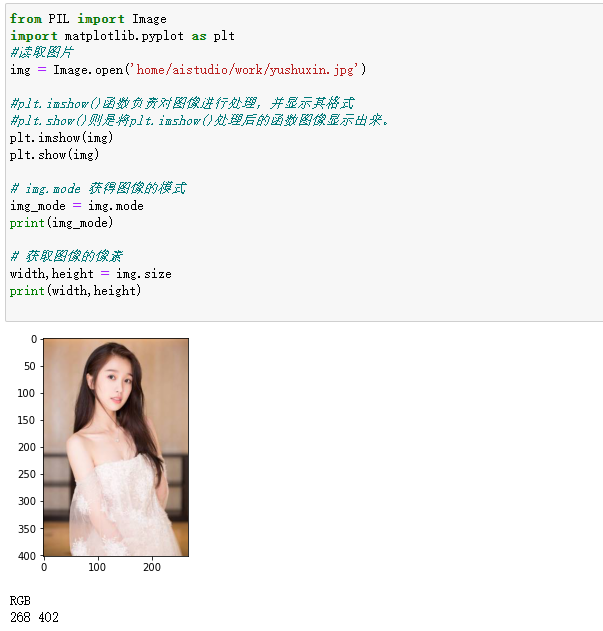
np.random.rand(50)，意思是随机生成50个[0,1)的数。

## PIL库

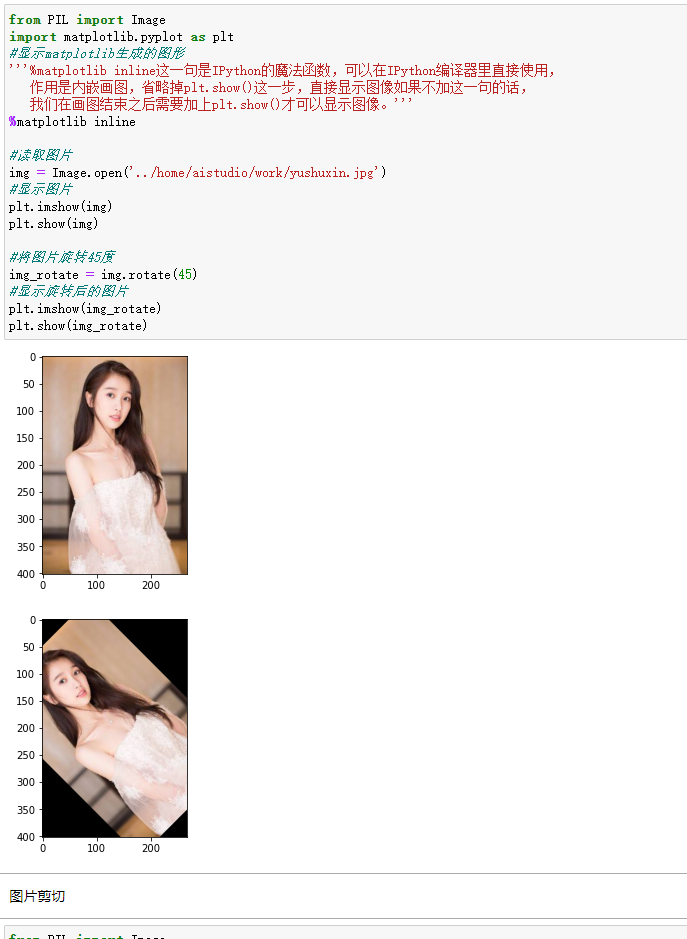
PIL库是一个具有强大图像处理能力的第三方库。

图像的组成：由RGB三原色组成,RGB图像中，一种彩色由R、G、B三原色按照比例混合而成。0-255区分不同亮度的颜色。图像的数组表示：图像是一个由像素组成的矩阵，每个元素是一个RGB值。

### 图片显示：



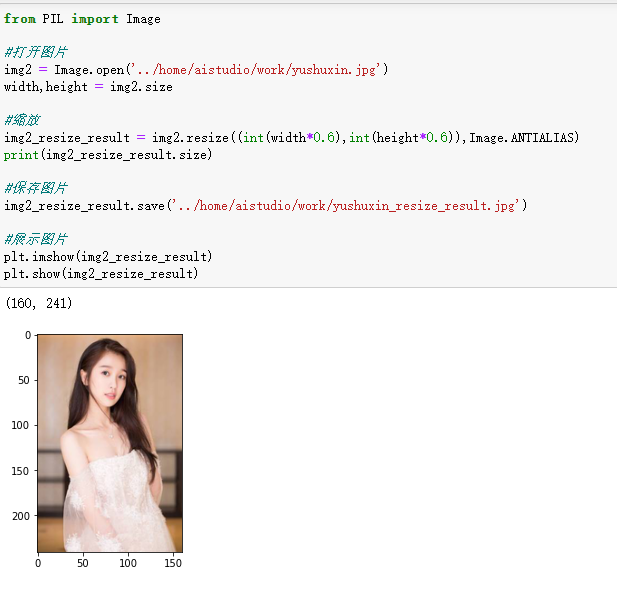
### 图片旋转：



### 图片剪切：



### 图片放缩：



### 镜像效果：左右旋转、上下旋转

