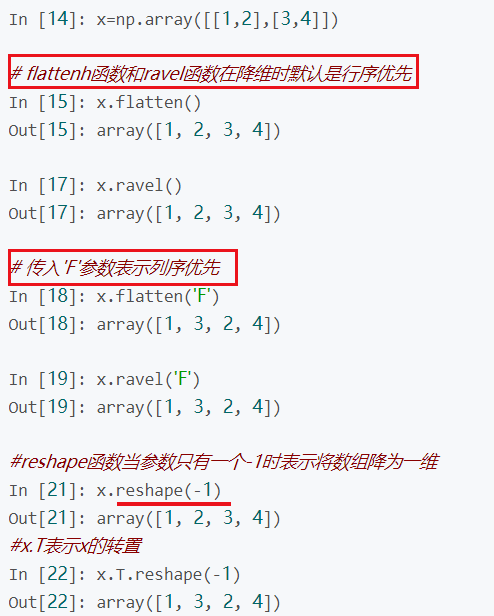
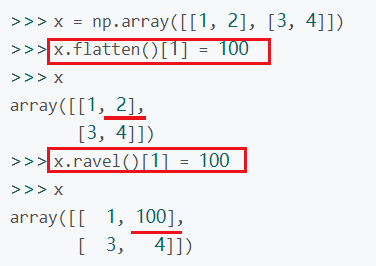
1. **numpy.ravel()、numpy.flatten()**

**功能：**都是实现多维数组—>一维数组

**相同点：**



**区别：**



1. **numpy.reshape()**

**功能：**实现数组维度的调整

**Eg：**import numpy as np

a = np.array([1,2,3],[4,5,6])

print(a.reshape(1,-1)) # 1\*n，n:由元素个数来决定，自动计算

print(a.reshape(-1,1)) # n\*1，n:由元素个数来决定，自动计算

print(a.reshape(2,-1)) # 2\*n，n:由元素个数来决定，自动计算

print(a.reshape(-1,2)) # n\*2，n:由元素个数来决定，自动计算

1. **numpy.zeros\_like()**

**功能：**生成一个和输入数组一样维度的全零数组

**Eg**：theta = theta.reshape((n, 1))  
grad = np.zeros\_like(theta)

1. **.dot()**

**功能**：实现矩阵点成功能

**Eg:**import numpy as np

a = np.array([1,2,3])

x = np.array([1,2],[3,4],[4,6])

print(a.T.dot(a)) # 14

print(a.dot(a)) # 14

print(x.T.dot(a)) # 2\*1，用x的转置\*a的对应元素，再相加。[19,25]

1. **numpy.linspace(start, stop, num=50, endpoint=True, retstep=False, dtype=None)**

**功能：**生成一个指定间隔数的数组。

**参数：**

（1）start：序列的起始值

（2）stop：序列的终止值

（3）生成的样本数（默认值=50）

（4）endpoint：该值取True时，序列的终止值包括stop；反之不包括。

（5）retstep：该值取True时，生成的数组中显示间距；反正不显示。

（6）dtype：数据类型；输出数组的类型。如果没有给出dtype，则从其他输入参数中推断数据类型

1. **np.argmin(a)，****np.argmax(a),**

**np.maximum(a,b,c,…..), np.minimum(…….)**

**功能：**

1）：np.argmin(a)、np.argmax(a)：查找数组中最小\最大元素的**位置（序号）**

**eg：求解a数组每行最大元素的下标**

np.argmax(a, axis=1 )

2）：np.maximum(a,b,c,…..)：提取数组a，b，c…中**对应位置**最大的元素

1. **random.uniform(x,y)**

**功能**：生成一个随机数，范围[x,y]

**np.random.randint(1,5,5)**

**功能：**[1-5]区间内的5个数

**a = np.random.random(5)**

功能：生成5个正态分布的随机数，区间[0-1]

1. **np.multiply(np.mat(A),np.mat(B))**

**功能**：矩阵对应元素位置相乘，利用np.mat()将数组转换为矩阵

Eg:A = np.array([1,2],[3,4])

B = np.array([0,1],[2,3])

Print(np.mulyiply(A,B))

结果：array([0,2]

[6,12])

1. **z = ((x).sum(axis=0))**

功能：实现x矩阵的**行**相加

**Eg：**sum(axis=**1**):实现X矩阵的**列**相加

注：默认是axis=**0**，即：**行**相加

1. **3D绘图**

ax=plt.subplot(projection='3d')

ax.scatter3D(data[:,0],data[:,1],r,c=target,s=50,cmap='autumn')

ax.view\_init(elev=elev,azim=azim)

**说明：**view\_init（elev，azim）

elev、azim：这两个参数规定了3D图形的视角

1. **同一窗口，绘制多图**

fig,ax = plt.subplots()

**解释：（1）fig,ax = plt.subplots()**等价于：

1、fig = plt.figure()

2、ax = fig.add\_subplot(1,1,1)

**（2）fig, ax = plt.subplots(1,3)**

其中参数1和3分别代表子图的行数和列数，一共有 1x3个子图像。函数返回一个figure图像和子图ax的array列表。

**（3）fig, ax = plt.subplots(1,3,figsize=(15,7))**

1、1行3个15x7大小的子图.

2、ax.scatter(X[:, 0], X[:, 1], c=y, s=50, cmap=**'autumn'**)

用这个命令画图

1. **numpy.power(x, a)**

功能：求解x数组各个元素的a次方

Eg：x = [1,2,3]

numpy.power(x,2)

结果：[1,4,9]

1. **mat()函数与array()函数生成矩阵所需的数据格式有区别**

（1）mat（）：数据可以使用；分隔，也可使用，分隔

array（）：只能使用，分隔

https://img-blog.csdn.net/20180408105657634?watermark/2/text/aHR0cHM6Ly9ibG9nLmNzZG4ubmV0L0J1aWxkX1RpZ2Vy/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70https://img-blog.csdn.net/20180408105528631?watermark/2/text/aHR0cHM6Ly9ibG9nLmNzZG4ubmV0L0J1aWxkX1RpZ2Vy/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70

https://img-blog.csdn.net/20180408110248365?watermark/2/text/aHR0cHM6Ly9ibG9nLmNzZG4ubmV0L0J1aWxkX1RpZ2Vy/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70

（2）mat()：矩阵的乘积可以使用（星号） \*  或 .dot()函数，其结果相同。而矩阵对应位置元素相乘需调用numpy.multiply()函数。

（3）array()：矩阵的乘积只能使用 .dot()函数。而星号乘 （\*）则表示矩阵对应位置元素相乘，与numpy.multiply()函数结果相同。

1. **numpy.arange**

**功能：**返回一组有间隔的数据

**Eg: for** e **in** np.arange(np.min(p), np.max(p), stepSize):

说明： e：有间距的数组

起始：p数组的最小值

终点：**（小于）**p数组的最大值

**步长**：stepSize，**默认为1**

1. **numpy.linspace(a,b,n)**

功能：实现区间a-b，等间隔n个数

eg：a = np.linspace(0,2,100)

维度：100\*1

起点：0

终点：2

1. **矩阵乘法：**

eg：1）点乘：A.dot(B) == np.dot(A,B) ：内积

2）星乘：A\*B ：对应元素相乘

1. **矩阵拼接、切割，扩展**

np.hstack((a,b)):按照列拼接，[a,b]

np.vstack((a,b)):按照行拼接，[a

b]

np.hsplit(a,3):按照列切割，分成三份

np.vsplit(a,3):按照行切割，分成三份

np.hsplit(a,(3,4)):按照列切割，分成3行4列的

z = np.tile(a,(1,3)):扩展，复制a矩阵，z=[a,a,a]

1. **矩阵求解最大、最小索引值：np. argsort（a）**

**eg：**

aa = np.array([1,2,3,5,4,5])

f = np.argsort(aa)

**结果：**f= array([0, 1, 2, 4, 3, 5]

f：aa数组所有元素**从小到大**排序的**标签**

即：提取其对应的index(索引)

**应用**：可以用过f[**-**（n：m）]，来得到数组aa的**前n~m个最大的元素**

1. **np.linalg.eig(a)**

**功能：**计算a矩阵特征值、特征向量

**eg**：

import numpy as np

#均值归一化，dataMat为数据集

meanVals = np.mean(dataMat, axis=0)

meanRemoved = dataMat - meanVals

covMat = np.cov(meanRemoved, rowvar = 0)

#计算特征值、特征向量

eigVals， eigVects = np.linalg.eig(mat(covMat))

1. **SVD**

**def SVD**(data\_normal):  
 m, n = data\_normal.shape  
 sigma = data\_normal.T.dot(data\_normal) / m *# np.cov()，协方差矩阵* U, S, V = np.linalg.svd(sigma) *# np.linalg.eig() 奇异值分解* **return** U, S, V

1. **np.linalg.norm()**

功能：计算范数

1. **np.random.rand(d0,d1,d2……dn)**

**功能：**返回一个或一组服从“0~1”均匀分布的随机样本值。随机样本取值范围是[0,1)，不包括1。

可以用于生成矩阵

（注：使用方法与np.random.randn()函数相同）

eg：

np.ramdom.rand(1) #生成一个[0,1)范围内的数

np.random.rand(3,3)#生成一个3x3的矩阵，元素取值范围：[0,1)

1. np.max(a)

功能：求a矩阵内所有元素的最大值

eg：

zz= np.max([[1,2,41],[1,5,4]],axis=0)

print(zz)

zz=[1,5,41]

1. lambda:

功能：用于定义匿名函数

eg：

add = lambda x,y : x + y

add(1,2)

result=3

1. **extend、append区别**

list.append(object) 向列表中添加一个对象object  
list.extend(sequence) 把一个序列seq的内容添加到列表中

eg：

music\_media = ['compact disc']

new\_media = ['DVD Audio disc']

#使用append

music\_media.**append**(new\_media)

print(music\_media)

结果：

['compact disc'，['DVD Audio disc']]

#使用extend

music\_media.**extend**(new\_media)

print(music\_media)

结果：

['compact disc'，'DVD Audio disc']

1. **.clf(); .gcf(); .gca()**

当前的图表和子图可以使用gcf()和gca()获得，它们分别是“Get Current Figure”和“Get Current Axis”的开头字母缩写。

gcf()：获得图表的Figure对象。

gca()：获得子图的Axes对象。

clf()：清除图形窗口上的旧图形。

1. **Kfold(n, n\_folds, shuffle=False, random\_state=None)**

**功能：**用于构造交叉验证数据集

**参数**：(n, n\_folds=3, shuffle=False, random\_state=None)

n为总数

n\_folds为分为多少个交叉验证集

shuffle为是否随机

random\_state设置随机因子

**eg：**

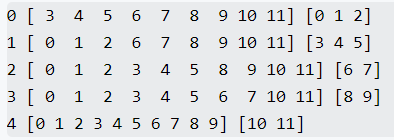
from sklearn.cross\_validation import KFold

kf = KFold(12,n\_folds=5,shuffle=False)

for i,(train\_index,test\_index) in enumerate(kf):

print(i,train\_index,test\_index)

**结果：**



（注：i=0-4，其数值来源于枚举enumerate）

1. **//:向下取整**

eg：

>>> 9//2

4

>>> -9//2

-5

1. **list转化为矩阵：**

使用np.array();

**eg:**

a = []

a = np.array(a)

（注：列表没有shape属性，只用矩阵有shape属性）

**使用技巧：**

想要使用append（）挂载数据，

1、定义a=[]，即：用列表的形式存放数据

2、等所有数据挂载完毕，在使用np.array(a),变为矩阵形式（方便处理数据）

1. **图像、矩阵维度压缩：**

使用scipy.misc库中的imresize（）。

**eg：**

**import** scipy.misc **as** smi

im\_conds = smi.imresize(img, (condense\_rows, condense\_cols),mode=**"L"**)

（**注：**

mode = 1： 二值图像；

mode = L : 灰度图像；

mode = P： 8位彩图（FCN网络）；

mode = None：default）

1. 查找列表中所有不同的元素：

使用set()

eg：

#d：列表

a = set(d)

print(a)

结果：{0.0, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0}

（**注：寻找标签总数方法：**  
 labelnum = len(set(train\_label)) *#标签总个数*）

1. **numpy数据类型强制转换**

使用astype（数据类型）

eg：a数组元素为float，转换为int

a = a.astype(int)

1. **assert**

**功能：**用来判断2个矩阵、列表元素是否相同，返回bool类型数据

eg：

**assert len(lists) >=5,'列表元素个数小于5'**

**assert 2==1,'2不等于1'**

1. 枚举：enumerate（）

>>>seq = ['one', 'two', 'three']

>>> for i, element in enumerate(seq):

... print i, element

... 0 one 1 two 2 three

1. **print(‘{}’.format())的用法：**

eg：

print('{:>8}'.format('zhang'))

print('{:0>8}'.format('zhang'))

print('{:a<8}'.format('zhang'))

print('{0},{1}'.format('zhangk', 32)) #{1}对应的索引位置

print('{},{},{}'.format('zhangk','boy',32)) #直接打印对应元素

结果：

zhang

000zhang

zhangaaa

ppzhangppp

zhangk,32

zhangk,boy,32

1. **在.csv中保存数据**：用于比赛时填写提交的结果文件

#在.csv中保存数据：保存列，元素必须是以list形式存放

save\_result = predictions\_lgb\_round.tolist()

#打开文件

data = pd.read\_csv(r'./data/happiness\_submit.csv')

#修改某列的数据

data['happiness'] = save\_result

#print(data.columns)

#mode = 'a',以追加的形式打开文件

data.to\_csv(r'./data/happiness\_submit.csv',mode = 'a', index = False)

1. np.where使用

用于查找数组中元素的位置，或者筛选出符合条件的元素

**eg：**

条件为[[True,False], [True,False]]，分别对应最后输出结果的四个值。第一个值从[1,9]中选，因为条件为True，所以是选1。第二个值从[2,8]中选，因为条件为False，所以选8，后面以此类推

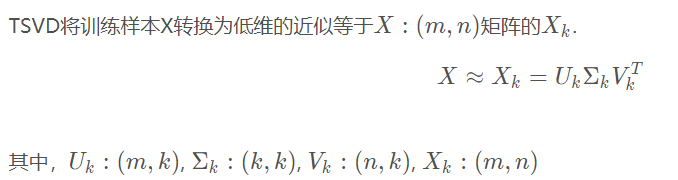
print(np.where([[True,False], [True,True]],   
 [[1,2], [3,4]],  
 [[9,8], [7,6]]))

**结果：**

[[1 8]

[3 4]]

4. **NLP常用函数：**
5. **TruncatdSVD**



**eg：**

**from** sklearn.decomposition **import** TruncatedSVD

svd = TruncatedSVD(n\_components = k, n\_iter = n\_iters)  
date\_reduced = svd.fit\_transform(data)



1. **Sklearn库函数：**
2. **X, y = make\_blobs(n\_samples, n\_features,centers,\**

**cluster\_std,random\_state)**

**功能**：产生随机数据

n\_samples: 产生的数据数量

n\_features：特征维度数量

centers: 数据种群数量

cluster\_std: 标准差

random\_state：相同的值具有可重现性，指定一个相同的数，用于下次重现

**Eg:**

**from** sklearn.datasets.samples\_generator **import** make\_blobs

X,y = make\_blobs(n\_samples=100,n\_features=2,\

centers=2,cluster\_std=0.6,random\_state=0)  
plt.scatter(X[:,0],X[:,1],c=y,s=50,cmap=**'autumn'**)

1. 使用**SVM**训练数据，训练结果存放于对应的对象中（eg：linearKernal）

**from** sklearn **import** svm

#使用线性核训练数据

linearKernal = svm.SVC(kernel=**'linear'**) #选择核函数  
linearKernal.fit(X\_linear, y\_linear) #传入训练数据集

#使用高斯核训练数据

gaussKernal = svm.SVC(kernel=**'rbf'**,C=1E3) #软间隔参数C的设置

gaussKernal.fit(X\_circle, y\_ circle)

1. pickle模块：

作用：序列化对象，将任意对象转化成文本和二进制表示的过程。

接口：dumps、loads

1. dumps（object）返回字符串对象，包含一个pickle格式的对象
2. loads（string）返回包含在pickle字符串中的对象

eg：、转换、保存对象

**import** pickle  
a = ("pickle",123,[4,5,6],{'A','1','B',2})  
p1 = pickle.dump(a)  
p2 = pickle.load(a)  
**with** open('text.txt','wb') **as** file:  
 pickle.dump(a,file)

1. joblib模块：由scikit-learn外带的库

joblib只能将对象保存在硬盘文件中，不能保存为字符串

3. **Pandas库**
4. **打开文件：pandas.read\_csv(“xxx.csv”)**

eg: data = pandas.read\_csv(“data.csv”)

(注：**Pandas 读取文件时候**，注意文件的存储路径不能带有中文，否则读取可能出错)

1. **.head(n)、.tail(n):默认值都是5**

功能：显示前(后)n行数据

eg：data.head(3)

data.tail(3)

1. **.columns**

功能：得到列名

eg:data.columns

1. **新建列：**

eg:data[“kg”] = 数据

1. **.max()**

功能：寻找列最大值

eg：maxColumn = data[“kg”].max()

1. **.pivot\_table(index=”x1”,values=”x2”,aggfun=np.mean)**

功能：求解数据表对应行的平均值

1. **.apply(函数名)**

功能：在对应的数据集上，用于使用自定义函数

1. **.to\_datetime(a[“DATE”])**

功能：将a对应的列[“DATE”]的数据，转化为时间格式

1. **.iloc()、loc()**

.iloc：根据标签的所在位置，从0开始计数，选取**列**

.loc：根据DataFrame的具体**标签**选取**列**

eg:

data\_test1=data.iloc[:,:8] #选取位置为[0,8)列的整列数据

data\_test2=data.iloc[0:2,8] #选取位置为8的列的[0,2)行的数据

data\_test3=data.loc[0:2,'工龄'] #选取列名为‘工龄’的[0,2]行的数据

1. . fillna()

用于填充缺失的数据

1. **pandas的数据创建、均值求解：**

b = pd.Series([8,9,5,4],index=['one',"two","three","four"])  
print(b["one"])  
  
# 创建数据框  
c = pd.DataFrame([[1,5,7],[4,5,6]],columns=['one','two','three'],index=['1','2'])  
print(c)  
# 用字的字典的方式创建数据框  
d = pd.DataFrame({  
 'one':4,  
 'two':[4,8,7]  
 # 'three':list(str(852))  
})  
# head默认选择数据的前几行：默认是5行  
# tail默认取后5行  
print(d.head)  
# 求解数据列的均值、最大值、最小值、标准差：**describe()仅仅计算数字量**  
print(d.describe())  
# 求解转置矩阵  
print(d.T)

1. **导入csv、excel数据:进行数据处理**

# 读取数据  
data = pd.read\_csv("./data/test.csv")  
# 显示前10行数据  
print(data.head(10))  
print(data.describe())  
# 按照列名‘Pclass’排序：升序  
print(data.sort\_values(by="Pclass"))

1. **Mysql数据库：读取、保存数据**

**从MySQL中读取数据**

import pymysql

import pandas as pd  
conn = pymysql.connect(host = '192.168.154.1', user = "root", passwd = "1230re0321re", db = "dangdang")

# 查询语句  
sql = "select \* from bookinfo"

# 读取mysql中的数据  
dataBookInfo = pd.read\_sql(sql, conn)  
print(dataBookInfo.shape)

**保存数据到MySQL中**

# 打开数据库连接  
db = pymysql.connect(server, user, password,database)  
# 使用cursor()方法获取操作游标  
cursor = db.cursor()  
for i in range(0, len(item["title"])):

# 将title、price、comment、url，4种数据写入Mysql的bookinfo表格  
 title = item["title"][i]  
 price = item["price"][i]  
 comment = item["comment"][i]  
 url = item["url"][i]  
 sql = "insert into bookinfo(title, price, comment, url) \  
 values ('%s', '%s', '%s', '%s')" % \  
 (title, price, comment, url)  
 print(title + " --- " + price + " --- " + comment + url + "\n")  
 # 执行sql语句  
 cursor.execute(sql)  
 # 提交到数据库执行  
 db.commit()  
# 关闭数据库  
db.close()

1. **pandas导入文本数据**

# 读取文本数据  
print("read data from 'testData.txt'")  
txtData = pd.read\_table("./data/testData.txt")  
print(type(txtData))

2. **Seaborn库:实质在matplotlib上进行封装的**

**import seanborn as sns**

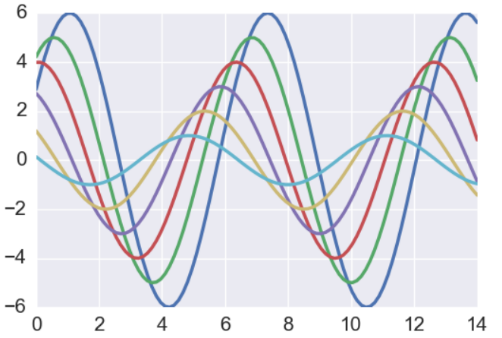
def sinplot(flip=1):

x = np.linspace(0, 14, 100)

for i in range(1, 7):

plt.plot(x, np.sin(x + i \* .5) \* (7 - i) \* flip)

执行：sinplot（）



1. **sns.set()**

**功能：**默认的seanborn设置

eg：

sns.set()

sinplot()

1. **sns.set\_style()**

**功能**：设置风格

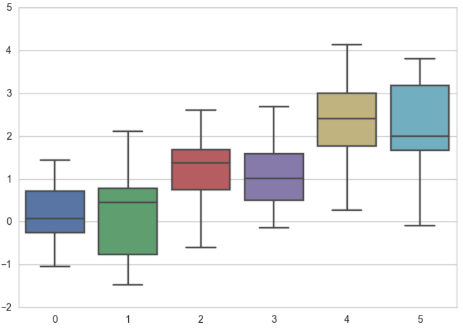
**5种风格：**darkgrid、whitegrid、dark、white、ticks

**eg：**（注：np.random.normal——生成正态分布数据）

sns.set\_style("**whitegrid**")

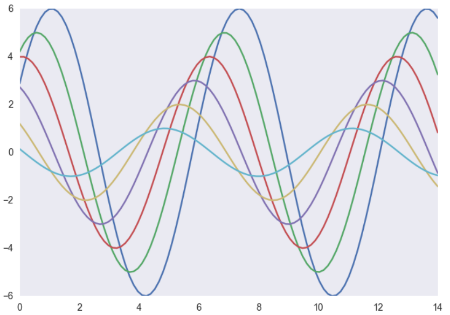
data = np.random.normal(size=(20, 6)) + np.arange(6) / 2

sns.boxplot(data=data)



sns.set\_style("**dark**")

sinplot()

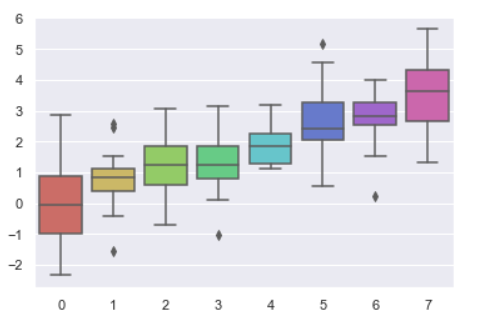


1. **sns.palplot(sns.color\_palette("hls", 8))**

**功能：**画板颜色

data = np.random.normal(size=(20, 8)) + np.arange(8) / 2

sns.boxplot(data=data,**palette=**sns.color\_palette("hls", 8))



（注释：sns.palplot(sns.color\_palette("hls", 8))：hls，8种颜色）



1. **sns.palplot(sns.hls\_palette(8, l=.7, s=.9))**

**功能：**调节亮度、饱和度

l-亮度 lightness

s-饱和 saturation



1. **连续变色**

**1）**sns.palplot(sns.color\_palette("Blues"))



（注：如果想要翻转渐变，可以在面板名称中添加一个\_r后缀）

**2）**sns.palplot(sns.color\_palette("**cubehelix**", 8))



（注：线性色调变换）

**3）**绘制连续型等高线

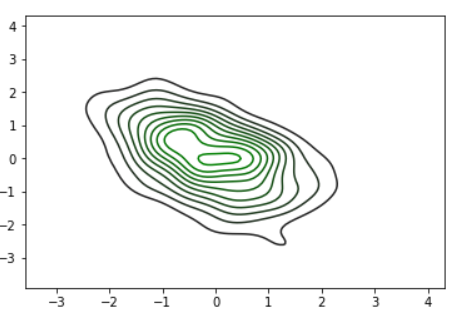
import seaborn as sns

import numpy as np

x, y = np.random.multivariate\_normal([0, 0], [[1, -.5], [-.5, 1]], size=300).T

**pal** = sns.**dark\_palette**("**green**", as\_cmap=True)

sns.kdeplot(x, y, **cmap=pal**);



1. **绘制散点图**

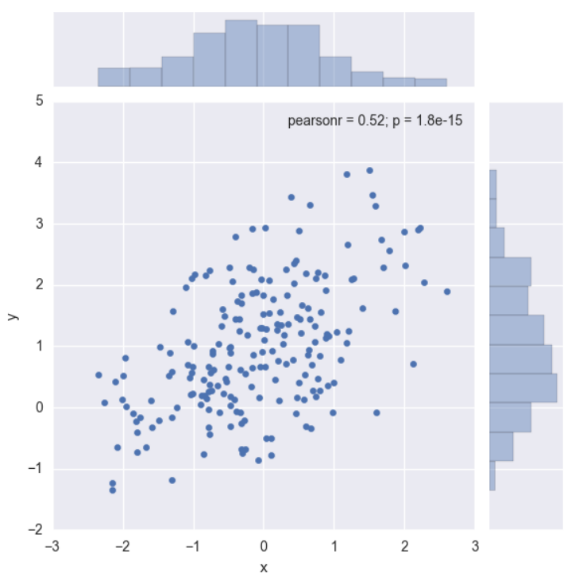
import pandas as pd

mean, cov = [0, 1], [(1, .5), (.5, 1)]

data = np.random.multivariate\_normal(mean, cov, 200)

df = pd.DataFrame(data, columns=["x", "y"])

sns.jointplot(x="x", y="y", data=df);



1. facetgrid()

功能：绘制多变量数据的图像

5. **matplotlib库**
6. **plt.xticks(rotation=n)**

功能：x坐标的刻度，旋转n°

1. **绘制子图.add\_subplot(x,y,z)**

功能：在同一个图片中，绘制多个子图

eg：import matplotlib as plt

fig = plt.figure(figsize=(长,宽))#图像的长宽

ax1 = fig.add\_subplot(2,2,**1**) #子图**1**

ax3 = fig.add\_subplot(2,2,**3**) #子图**3**

ax1.plot(np.arange(10)\*2,np.arange(10))

plt.show() #未指定的子图，不会被显示

1. **fig, ax = plt.subplot(1,2,figsize(15,10))**

**说明：该命令可替代（2）的命令**

fig, ax = plt.subplots(1, 2, figsize=(16, 6))

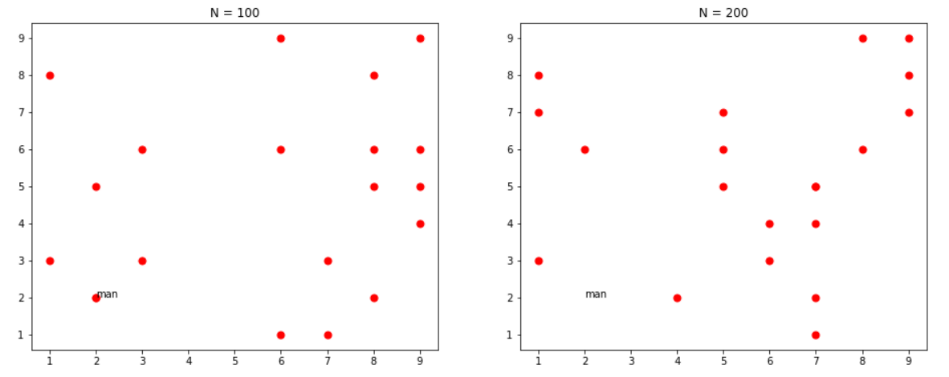
for axi, N in zip(ax, [100, 200]): # zip将ax、[100，200]打包成元组

axi.scatter(np.random.randint(1,10,20),np.random.randint(1,10,20), c='r', s=50, cmap='autumn')

axi.set\_title('N = {0}'.format(N))

axi.text(2,2,'man') #指定位置标注

plt.show()



1. **plt.plot(x,y,c=’r’,label=’xx’)**

功能：画图(**曲线图**)

**c**=’r’:颜色

**label**=’xx**’:**绘制图线的名字xx

（需要加上：plt.legend(loc=’best’),才能显示label）

1. **.scatter()**

功能：绘制**散点图**

1. **.barh()**

功能：绘制柱形图：

1. **.plot()**

绘制曲线函数

import matplotlib.pyplot as plt

import numpy as np

x = np.linspace(0.05, 10, 1000) #起点、终点、数据个数

y = np.cos(x)

plt.plot(x, y, ls="-", lw=2, c = 'r'，label="plot figure")

plt.legend()

plt.show()

1. **显示矩阵数据的图片：**

使用imshow（）

eg：

*#灰度显示*

plt.imshow(train\_images[i], cmap=**'gray'**)

*#暂停图片的时间*  
plt.pause(0.01)  
plt.show()