【Python数据挖掘课程】六.Numpy、Pandas和Matplotlib包基础知识

前面几篇文章采用的案例的方法进行介绍的，这篇文章主要介绍Python常用的扩展包，同时结合数据挖掘相关知识介绍该包具体的用法，主要介绍Numpy、Pandas和Matplotlib三个包。目录：

一.Python常用扩展包
二.Numpy科学计算包
三.Pandas数据分析包
四.Matplotlib绘图包

前文推荐：

【Python数据挖掘课程】一.安装Python及爬虫入门介绍
【Python数据挖掘课程】二.Kmeans聚类数据分析及Anaconda介绍
【Python数据挖掘课程】三.Kmeans聚类代码实现、作业及优化
【Python数据挖掘课程】四.决策树DTC数据分析及鸢尾数据集分析
【Python数据挖掘课程】五.线性回归知识及预测糖尿病实例

绘图强推：http://python.jobbole.com/85106/

希望这篇文章对你有所帮助，尤其是刚刚接触数据挖掘以及**大数据**的同学，这些基础知识真的非常重要。如果文章中存在不足或错误的地方，还请海涵~

部分截图参考张良均的《Python数据分析与挖掘实战》，推荐大家购买阅读。

一. Python常用扩展包

参考张良均的《Python数据分析与挖掘实战》，下图展示了常见的Python扩展包。

扩 展 库	简 介
Numpy	提供数组支持，以及相应的高效的处理函数
Scipy	提供矩阵支持，以及矩阵相关的数值计算模块
Matplotlib	强大的数据可视化工具、作图库
Pandas	强大、灵活的数据分析和探索工具
StatsModels	统计建模和计量经济学，包括描述统计、统计模型估计和推断
Scikit-Learn	支持回归、分类、聚类等的强大的机器学习库
Keras	深度学习库，用于建立神经网络以及深度学习模型
Gensim	用来做文本主题模型的库，文本挖掘可能用到

常用的包主要包括：

1.Numpy

Python没有提供数组，列表（List）可以完成数组，但不是真正的数据，当数据量增大时，，它的速度很慢。所以Numpy扩展包提供了数组支持，同时很多高级扩展包依赖它。例如：Scipy、Matplotlib、Pandas。

2.Scipy

该包提供矩阵支持，以及矩阵相关的数值计算模块。如果说Numpy让Python有了Matlab的味道，那么Scipy就让Python真正地成为二半个Matlib。因为涉及到矩阵内容，而课程中主要使用数组，所以不再介绍。

3.Pandas

Pandas是面板数据（Panel Data）的简写。它是Python最强大的数据分析和探索工具，因金融数据分析工具而开发，支持类似SQL的数据增删改查，支持时间序列分析，灵活处理缺失数据，后面详细介绍。

4.Scikit-Learn

Scikit-Learn是一个基于python的用于数据挖掘和数据分析的简单且有效的工具，它的基本功能主要被分为六个部分：分类(Classification)、回归(Regression)、聚类(Clustering)、数据降维(Dimensionality Reduction)、模型选择(Model Selection)、数据预处理(Preprocessing)，前面写的很多文章算法都是出自该扩展包。

详见官网：http://scikit-learn.org/stable/

5.Matplotlib

该包主要用于绘图和绘表，强大的数据可视化工具，做图库，语法类似MATLAB。同时，Seaborn也是数据可视化的工具包。

注意：这些包在Anaconda集成环境中已经存在，可以直接使用，最早我是通过Python2.7来编写代码的，安装过程通过pip install numpy，而且安装顺序非常讲究，容

易出错，所以推荐大家使用该集成包。

二. Numpy科学计算包

NumPy（Numeric Python）系统是Python的一种开源的数值计算扩展，一个用python实现的科学计算包。它提供了许多高级的数值编程工具，如：矩阵数据类型、矢量处理，以及精密的运算库。专为进行严格的数字处理而产生。

推荐学习：http://old.sebug.net/paper/books/scipydoc/numpy_intro.html

下面通过这段代码详细讲解这个包在数据分析中的常见用法：

1.一维数组处理

```
#导入包并重命名
import numpy as np

#定义一维数组
a = np.array([2, 0, 1, 5, 8, 3])
print u'原始数据:', a

#输出最大、最小值及形状
print u'最小值:', a.min()
print u'最大值:', a.max()
print u'形状', a.shape

#数据切片
print u'切片操作:'
print a[:-2]
print a[-2:]
print a[:1]

#排序
print type(a)
a.sort()
print u'排序后:', a
```

输出结果如下所示：

```
原始数据: [2 0 1 5 8 3]
最小值: 0
最大值: 8
形状 (6L,)
```

```
切片操作:
        [2 0 1 5]
[8 3]
[2]
<type 'numpy.ndarray'>
排序后: [0 1 2 3 5 8]
```

核心代码:

代码通过np.array定义了一个数组[2, 0, 1, 5, 8, 3]，其中min计算最小值，max计算最大值，shape表示数组的形状，因为是一维数组，故6L（6个数字）。

最重要的一个知识点是数组的切片操作，因为在数据分析过程中，通常会对数据集进行"80%-20%"或"70%-30%"的训练集和测试集划分，通常采用的方法就是切片。

a[:-2]表示从头开始获取，"-2"表示后面两个值不取，结果：[2 0 1 5]

a[-2:]表示后往前数两个数字，获取数字至结尾，即获取最后两个值[8 3]

a[:1]表示从头开始获取，获取1个数字，即[2]

2.二维数组处理

注意的是定义二维数组括号不要弄错，正确的应该是：[[1,2,3],[4,5,6]]

同时计算机的存储下标都是从0开始计算的。

```
>>> a[0,3:5]
array([3,4])
>>> a[4:,4:]
array([[44,45],[54,55]])
>>> a[:,2]
array([2,12,22,32,42,52])
>>> a[2::2,::2]
array([[20,22,24],
       [40,42,44]])
```

代码如下:

```
#定义二维数组
import numpy as np
c = np.array([[1, 2, 3, 4],[4, 5, 6, 7], [7, 8, 9, 10]])
```

```
#获取值
       print u'形状:', c.shape
print u'获取值:', c[1][0]
print u'获取某行:'
print c[1][:]
print u'获取某行并切片:'
print c[0][:-1]
print c[0][-1:]

#获取具体某列值
print u'获取第3列:'
print c[:,np.newaxis, 2]

#调用sin函数
print np.sin(np.pi/6)
print type(np.sin(0.5))

#范围定义
print np.arange(0,4)
print type(np.arange(0,4))
```

代码输出结果如下所示:

```
形状: (3L, 4L)
获取值: 4
获取某行:
[4 5 6 7]
获取某行并切片:
[1 2 3]
[4]
获取第3列:
[[3]
 [6]
 [9]]
0.5
<type 'numpy.float64'>
[0 1 2 3]
<type 'numpy.ndarray'>
```

需要注意:

（1）获取二维数组中的某行，如第2行数据[4,5,6,7]，采用方法是：c[1][:];

（2）获取二维数组中的某列，如第2列数据[[3] [6] [9]]，**c[:,np.newaxis, 2]**。因为通常在数据可视化中采用获取某列数据作为x或y坐标，同时多维数据也可以采用PCA降低

成两维数据，再进行显示。

最后希望读者自己去阅读该段代码。

三. Pandas数据分析包

Pandas是面板数据（Panel Data）的简写。它是Python最强大的数据分析和探索工具，因金融数据分析工具而开发，支持类似SQL的数据增删改查，支持时间序列分析，灵活处理缺失数据。

注意：首先声明改包功能非常强大，我只是学习了它的非常小的一部分，后面随着学习深入会写更多它的用法，同时建议读者自行学习，不喜勿喷。

3.3.1 基本统计特征函数

统计特征函数用于计算数据的均值、方差、标准差、分位数、相关系数和协方差等，这些统计特征能反映出数据的整体分布。本小节所介绍的统计特征函数如表3-8所示，它们主要作为Pandas的对象DataFrame或Series的方法出现。

表3-8 Pandas主要统计特征函数

方 法 名	函 数 功 能	所 属 库
sum()	计算数据样本的总和（按列计算）	Pandas
mean()	计算数据样本的算术平均数	Pandas
var()	计算数据样本的方差	Pandas
std()	计算数据样本的标准差	Pandas
corr()	计算数据样本的 Spearman（Pearson）相关系数矩阵	Pandas
cov()	计算数据样本的协方差矩阵	Pandas
skew()	样本值的偏度（三阶矩）	Pandas
kurt()	样本值的峰度（四阶矩）	Pandas
describe()	给出样本的基本描述（基本统计量如均值、标准差等）	Pandas

约定俗成的导入惯例：

```
from pandas import Series, DataFrame
import pandas as pd
```

1.常见用法：读写文件

这里读文件最常用的是两种方法：

```
#写入excel文件：
```

```
df.to_excel('foo.xlsx', sheet_name='Sheet1') #从excel文件中读取：
pd.read_excel('foo.xlsx', 'Sheet1', index_col=None, na_values=['NA'])
#写入csv文件：
df.to_csv('foo.csv')
#从csv文件中读取：
pd.read_csv('foo.csv')
#写入HDF5存储：
df.to_hdf('foo.h5','df')
#从HDF5存储中读取：
pd.read_hdf('foo.h5','df')
```

下面通过一个具体的案例来讲解该包，这里读取的数据是张良均的《Python数据分析与挖掘实战》的第六章的电力用户数据集，missing_data.xls文件。内容如下，共3列数据，分别是用户A、用户B、用户C，共21行，对应21天的用电量，其中包含缺失值。

```
235.8333        324.0343        478.3231
236.2708        325.6379        515.4564
238.0521        328.0897        517.0909
235.9063                514.89
236.7604        268.8324
        404.048 486.0912
237.4167        391.2652        516.233
238.6563        380.8241
237.6042        388.023 435.3508
238.0313        206.4349        487.675
235.0729
235.5313        400.0787        660.2347
        411.2069        621.2346
234.4688        395.2343        611.3408
235.5   344.8221        643.0863
235.6354        385.6432        642.3482
234.5521        401.6234
236     409.6489        602.9347
235.2396        416.8795        589.3457
235.4896                556.3452
236.9688                538.347
```

部分Excel文件数据截图如下所示：

	A	B	C
1	235.8333	324.0343	478.3231
2	236.2708	325.6379	515.4564
3	238.0521	328.0897	517.0909
4	235.9063		514.89
5	236.7604	268.8324	
6		404.048	486.0912
7	237.4167	391.2652	516.233
8	238.6563	380.8241	
9	237.6042	388.023	435.3508
10	238.0313	206.4349	487.675

具体代码如下所示：

```
#读取数据 header设置Excel无标题头
import pandas as pd
data = pd.read_excel("missing_data.xls", header=None)
print data

#计算数据长度
print u'行数', len(data)

#计算用户A\B\C用电总和
print data.sum()

#计算用户A\B\C用点量算术平均数
mm = data.sum()
print mm

#输出预览前5行数据
print u'预览前5行数据'
print data.head()

#输出数据基本统计量
print u'输出数据基本统计量'
print data.describe()
```

输出结果如下所示：

```
           0         1         2
0   235.8333  324.0343  478.3231
1   236.2708  325.6379  515.4564
2   238.0521  328.0897  517.0909
3   235.9063       NaN  514.8900
4   236.7604  268.8324       NaN
5        NaN  404.0480  486.0912
6   237.4167  391.2652  516.2330
7   238.6563  380.8241       NaN
8   237.6042  388.0230  435.3508
...
行数 21
0    4488.9899
1    6182.3265
2    9416.3276
dtype: float64
0    4488.9899
1    6182.3265
2    9416.3276
dtype: float64
预览前5行数据
          0         1         2
0  235.8333  324.0343  478.3231
1  236.2708  325.6379  515.4564
2  238.0521  328.0897  517.0909
3  235.9063       NaN  514.8900
4  236.7604  268.8324       NaN
输出数据基本统计量
                0           1           2
count   19.000000   17.000000   17.000000
mean   236.262626  363.666265  553.901624
std      1.225465   57.600529   67.707729
min    234.468800  206.434900  435.350800
25%           NaN         NaN         NaN
50%           NaN         NaN         NaN
75%           NaN         NaN         NaN
max    238.656300  416.879500  660.234700
```

其中data.describe()输出数据的基本信息统计，其方法参考前面的图，包括count计数、std、max等函数。同时因为Excel表格中存在空值，故Python显示为NaN（Not a Number）表示空。

2.Series

Series是一维标记数组，可以存储任意数据类型，如整型、字符串、浮点型和Python对象等，轴标一般指索引。

Series、Numpy中的一维array 、Python基本数据结构List区别：List中的元素可以是不同的数据类型，而Array和Series中则只允许存储相同的数据类型，这样可以更有效的使用内存，提高运算效率。

```
from pandas import Series, DataFrame

#通过传递一个list对象来创建Series，默认创建整型索引；
a = Series([4, 7, -5, 3])
print u'创建Series:'
print a

#创建一个带有索引来确定每一个数据点的Series ；
b = Series([4, 7, -5, 3], index=['d', 'b', 'a', 'c'])
print u'创建带有索引的Series:'
print b

#如果你有一些数据在一个Python字典中，你可以通过传递字典来创建一个Series；
sdata = {'Ohio': 35000, 'Texas': 71000, 'Oregon': 16000, 'Utah': 5000}
c = Series(sdata)
print u'通过传递字典创建Series:'
print c
states = ['California', 'Ohio', 'Oregon', 'Texas']
d = Series(sdata, index=states)
print u'California没有字典为空:'
print d
```

输出如下所示：

```
创建Series:
0    4
1    7
2   -5
3    3
dtype: int64
创建带有索引的Series:
d    4
b    7
a   -5
c    3
dtype: int64
通过传递字典创建Series:
Ohio      35000
Oregon    16000
Texas     71000
```

```
Utah       5000
dtype: int64
California没有字典为空:
California        NaN
Ohio          35000.0
Oregon        16000.0
Texas         71000.0
dtype: float64
```

Series的一个重要功能是在算术运算中它会自动对齐不同索引的数据。

3.DataFrame

DataFrame是二维标记数据结构，列可以是不同的数据类型。它是最常用的pandas对象，像Series一样可以接收多种输入：lists、dicts、series和DataFrame等。初始化对象时，除了数据还可以传index和columns这两个参数。

注意：

(1) 在pandas中用函数 isnull 和 notnull 来检测数据丢失：pd.isnull(a)、pd.notnull(b)。

Series也提供了这些函数的实例方法：a.isnull()。

(2) Pandas提供了大量的方法能够轻松的对Series，DataFrame和Panel对象进行各种符合各种逻辑关系的合并操作。如：Concat、Merge （类似于SQL类型的合并）、Append （将一行连接到一个DataFrame上）。

(3) DataFrame中常常会出现重复行，DataFrame的duplicated方法返回一个布尔型Series，表示各行是否是重复行；还有一个drop_duplicated方法，它返回一个移除了重复行的DataFrame。

总之，Pandas是非常强大的一个数据分析包，很多功能都需要我自己去慢慢摸索。

四. Matplotlib画图包

Matplotlib是一个Python的图形框架，类似于MATLAB和R语言。它是python最著名的绘图库，它提供了一整套和matlab相似的命令API，十分适合交互式地进行制图。而且也可以方便地将它作为绘图控件，嵌入GUI应用程序中。

补充两张图，原自《Python数据分析与挖掘实战》，对大家绘图很有帮助。

Python的主要作图库是Matplotlib，在第2章中已经进行了初步的介绍，而Pandas基于Matplotlib并对某些命令进行了简化，因此作图通常是Matplotlib和Pandas相互结合着使用。本小节仅对一些基本的作图函数做一下简介，而真正灵活地使用应当参考书中所给出的各个作图代码清单。我们要介绍的统计作图函数如表3-8所示。

表3-11 Python主要统计作图函数

作图函数名	作图函数功能	所属工具箱
plot()	绘制线性二维图，折线图	Matplotlib/Pandas
pie()	绘制饼型图	Matplotlib/Pandas
hist()	绘制二维条形直方图，可显示数据的分配情形	Matplotlib/Pandas
boxplot()	绘制样本数据的箱形图	Pandas
plot(logy = True)	绘制 y 轴的对数图形	Pandas
plot(yerr = error)	绘制误差条形图	Pandas

最常用的画图函数是plot，同时常用的设置样式方法见下图。

（1）plot

·功能：绘制线性二维图、折线图。

·使用格式：

plt.plot（x，y，S）

这是Matplotlib通用的绘图方式，绘制y对于x（即以x为横轴的二维图形），字符串参量S指定绘制时图形的类型、样式和颜色，常用的选项有：'b'为蓝色、'r'为红色、'g'为绿色、'o'为圆圈、'+'为加号标记、'-'为实线、'--'为虚线。当x、y均为实数同维向量时，则描出点（x(i)，y(i)），然后用直线依次相连。

D.plot（kind='box'）

这里使用的是DataFrame或Series对象内置的方法作图，默认以Index为横坐标，每列数据为纵坐标自动作图，通过kind参数指定作图类型，支持line（线）、bar（条形）、barh、hist（直方图）、box（箱线图）、kde（密度图）和area、pie（饼图）等，同时也能够接受plt.plot（）

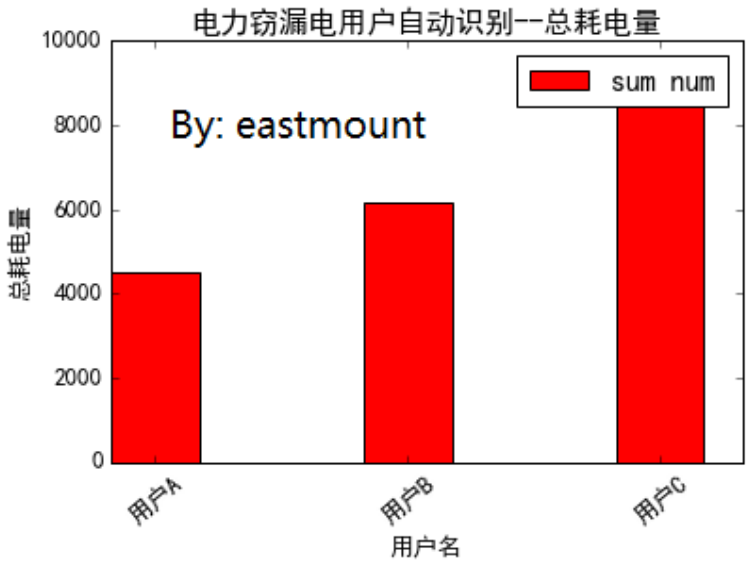
这里主要使用前面第三部分Pandas读取的电力数据绘制图形，主要是柱状图和饼图。

1.绘制柱状图

```
# -*- coding: utf-8 -*-
"""
```

```
Created on Mon Nov 14 04:06:01 2016

@author: yxz15
"""

#导入数据集
import pandas as pd
data = pd.read_excel("missing_data.xls", header=None)
mm = data.sum()
print u'计算用电量总数:'
print mm

#绘制图形
import numpy as np
import matplotlib.pyplot as plt
#中文字体显示
plt.rc('font', family='SimHei', size=13)
N = 3
#3个用户 0 1 2
ind = np.arange(N)  # the x locations for the groups
print ind
#设置宽度
width = 0.35
x = [u'用户A', u'用户B', u'用户C']
#绘图
plt.bar(ind, mm, width, color='r', label='sum num')
plt.xlabel(u"用户名")
plt.ylabel(u"总耗电量")
plt.title(u'电力窃漏电用户自动识别--总耗电量')
plt.legend()
#设置底部名称
plt.xticks(ind+width/2, x, rotation=40) #旋转40度
plt.show()
```

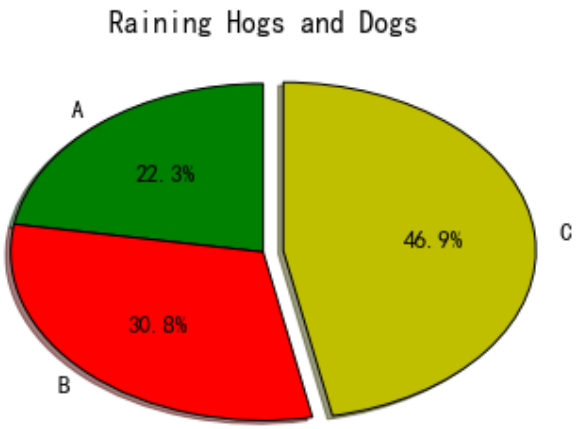
输出如下所示:



2.绘制饼图

```
import matplotlib.pyplot as plt

fracs = [45, 30, 25]                #每一块占得比例，总和为100
n = mm[0]+mm[1]+mm[2]
a = (mm[0]*1.0*100/n)
b = (mm[1]*1.0*100/n)
c = (mm[2]*1.0*100/n)
print a, b, c, n
fracs = [a, b, c]

explode=(0, 0, 0.08)                #离开整体的距离，看效果
labels = 'A', 'B', 'C'              #对应每一块的标志

plt.pie(fracs, explode=explode, labels=labels,
                autopct='%1.1f%%', shadow=True, startangle=90, colors = ("g",
"r", "y"))
                                    # startangle是开始的角度，默认为0，从这里开始按逆时针
方向依次展开
  plt.title('Raining Hogs and Dogs')   #标题
plt.show()
```

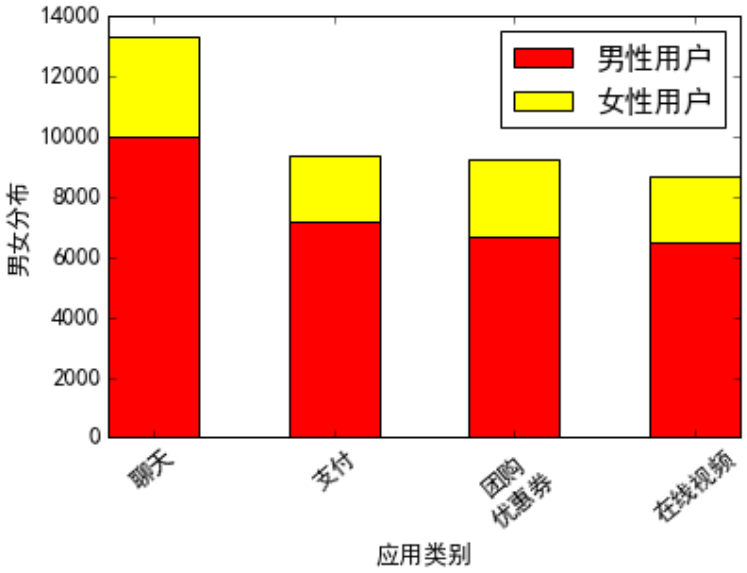
输出如下所示:



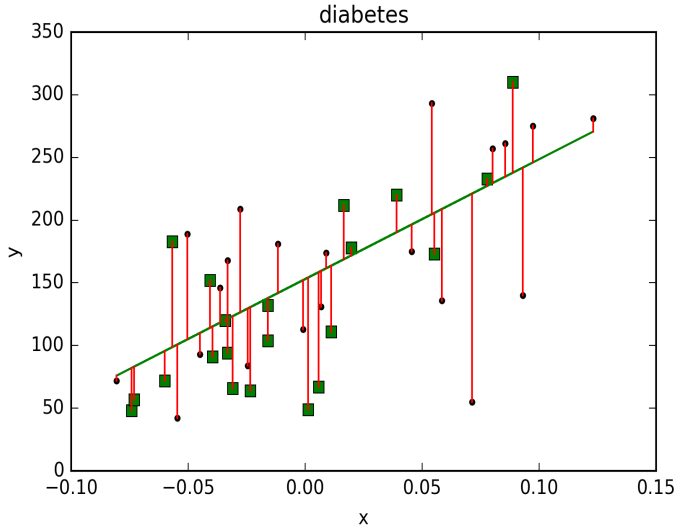
3.柱状图及比例显示

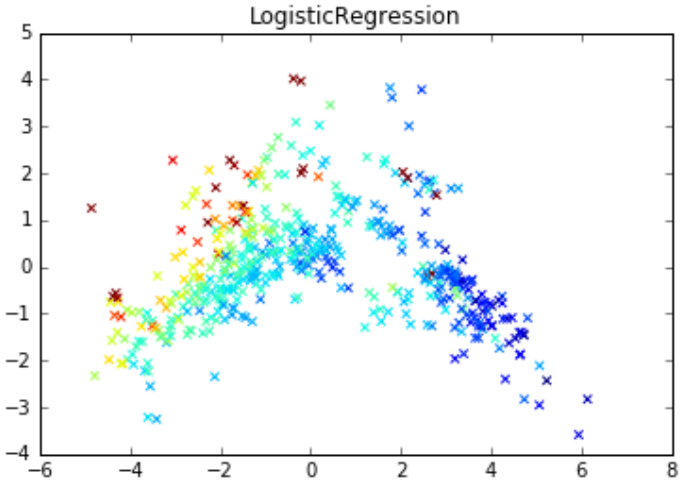
```
import matplotlib.pyplot as plt
import numpy as np
plt.rc('font', family='SimHei', size=13)

num = np.array([13325, 9403, 9227, 8651])
ratio = np.array([0.75, 0.76, 0.72, 0.75])
men = num * ratio
women = num * (1-ratio)
x = [u'聊天',u'支付',u'团购\n优惠券',u'在线视频']

width = 0.5
idx = np.arange(len(x))
plt.bar(idx, men, width, color='red', label=u'男性用户')
plt.bar(idx, women, width, bottom=men, color='yellow', label=u'女性用户')
plt.xlabel(u'应用类别')
plt.ylabel(u'男女分布')
plt.xticks(idx+width/2, x, rotation=40)
plt.legend()
plt.show()
```

输出如下所示（PS：该部分参考百度知道，网址忘记了，望提醒）。



当然该包可以绘制更多的图形，希望读者自己去学习。比如线性回归：





代码部分详解，引用前面自己写的第三篇文章：

matplotlib.pyplot是用来画图的方法，matplotlib是可视化包。

import matplotlib.pyplot as plt

绘制散点图（scatter），横轴为x，获取的第1列数据；纵轴为y，获取的第2列数据；c=y_pred对聚类的预测结果画出散点图，marker='o'说明用点表示图形。

plt.scatter(x, y, c=y_pred, marker='o')

表示图形的标题为Kmeans-heightweight Data。

plt.title("Kmeans-Basketball Data")

表示图形x轴的标题。

plt.xlabel("assists_per_minute")

表示图形y轴的标题。

plt.ylabel("points_per_minute")

设置右上角图例。

plt.legend(["Rank"])

表示显示图形。

plt.show()

最后希望文章对你有所帮助，上课内容还需要继续探索，但enjoy myself~

同时周末监考两天回来，确实挺累的，事情堆了很多，浪费15个小时，发现这份工作，赚点外块真不容易啊！甚至比程序猿累多了。

当老师难，当好老师更难，当贵州的好老师难上难。希望还能坚持自己的梦想，做个财大信院的扫地僧吧，但每每和学生一起还是非常享受的。同时，这次熬夜写文到深夜4点半，旁边也坐着一个自己的学生，在调试Struts、Json代码，所以说，还真不是这边的学生差，你懂得，但也并不是没有好老师，只是相对较少。fighting~

最后补充学生冯Y的一首朋友圈感言：

把握现在，活在当下。

不以物喜，不以己悲。

闲看花开花落，

静观云卷云舒。

顺其自然，随遇而安。
我也希望自己有朝一日能达到这种心境~
对这份工作、事业、校园、办公还是得看淡点。
(By:Eastmount 2016-11-14 中午4点半 http://blog.csdn.net/eastmount/)