【python数据挖掘课程】十六.逻辑回归LogisticRegression分析鸢尾花数据

今天是教师节，容我先感叹下。

祝天下所有老师教师节快乐，这是自己的第二个教师节，这一年来，无限感慨，有给一个人的指导，有给十几个人讲毕设，有几十人的实验，有上百人的课堂，也有给上千人的Python网络直播。但每当站到讲台前，还是那么兴奋，那么紧张，那么享受，仿佛整个世界都是我的，很满足。

站在讲台前的老师永远是最美的，很荣幸能教导自己的每一个学生，很开心给每一个陌生网友和朋友解惑，这是缘分，应该珍惜；再苦再累，都觉得值当。最后还是祝所有同行教师节快乐，娜老师和璋老师，节日快乐！用离开北理选择回来教书的那首诗结束。

《再见，北理工》

但行好事，莫问前程。

待随满天李桃，在追学友趣事。

别时到一句珍重，

不去思量，非常难忘。

无人可以诉衷肠，那又何妨。

留一段剪影，于心中回放。

几十年生死，不也两茫茫。

言归正传，前面第九篇文章作者介绍了线性回归相关知识，但是很多时候数据是非线性的，所以这篇文章主要讲述逻辑回归及Sklearn机器学习包中的LogisticRegression算法。希望文章对你有所帮助，如果文章中存在错误或不足之处，还请海涵~

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
【Python数据挖掘课程】六.Numpy、Pandas和Matplotlib包基础知识
【Python数据挖掘课程】七.PCA降维操作及subplot子图绘制
【Python数据挖掘课程】八.关联规则挖掘及Apriori实现购物推荐
【Python数据挖掘课程】九.回归模型LinearRegression简单分析氧化物数据
【python数据挖掘课程】十.Pandas、Matplotlib、PCA绘图实用代码补充
【python数据挖掘课程】十一.Pandas、Matplotlib结合SQL语句可视化分析
【python数据挖掘课程】十二.Pandas、Matplotlib结合SQL语句对比图分析
【python数据挖掘课程】十三.WordCloud词云配置过程及词频分析
【python数据挖掘课程】十四.Scipy调用curve_fit实现曲线拟合
【python数据挖掘课程】十五.Matplotlib调用imshow()函数绘制热图

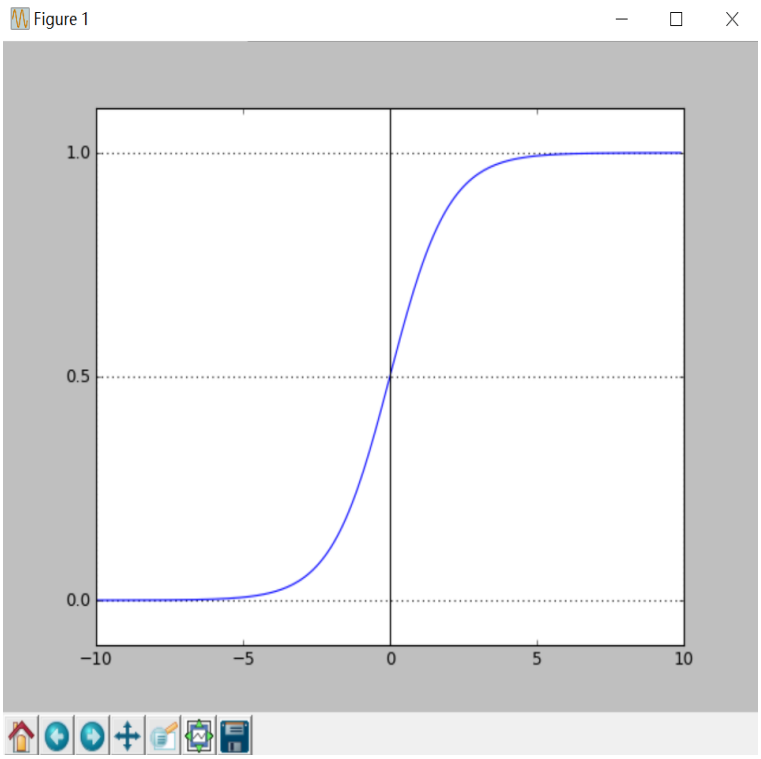
一. 逻辑回归

在前面讲述的回归模型中，处理的因变量都是数值型区间变量，建立的模型描述是因变量的期望与自变量之间的线性关系。比如常见的线性回归模型：

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_n x_n$$

而在采用回归模型分析实际问题中，所研究的变量往往不全是区间变量而是顺序变量或属性变量，比如二项分布问题。通过分析年龄、性别、体质指数、平均血压、疾病指数等指标，判断一个人是否换糖尿病，Y=0表示未患病，Y=1表示患病，这里的响应变量是一个两点（0-1）分布变量，它就不能用h函数连续的值来预测因变量Y（只能取0或1）。
总之，线性回归模型通常是处理因变量是连续变量的问题，如果因变量是定性变量，线性回归模型就不再适用了，需采用逻辑回归模型解决。

逻辑回归（Logistic Regression）是用于处理因变量为分类变量的回归问题，常见的是二分类或二项分布问题，也可以处理多分类问题，它实际上是属于一种分类方法。
二分类问题的概率与自变量之间的关系图形往往是一个S型曲线，如图所示，采用的Sigmoid函数实现。



这里我们将该函数定义如下：

$$\mathrm{f}(x)=\frac{1}{1+e^{-x}}$$

函数的定义域为全体实数，值域在[0,1]之间，x轴在0点对应的结果为0.5。当x取值足够大的时候，可以看成0或1两类问题，大于0.5可以认为是1类问题，反之是0类问题，而刚好是0.5，则可以划分至0类或1类。对于0-1型变量，y=1的概率分布公式定义如下：

$$\mathrm{P}(\mathrm{y}=1)=p$$

y=0的概率分布公式定义如下：

$$\mathrm{P}(\mathrm{y}=0)=1-p$$

其离散型随机变量期望值公式如下：

$$E(\mathrm{y})=1*p+0*(1-\mathrm{p})=p$$

采用线性模型进行分析，其公式变换如下：

$$\mathrm{p}(\mathrm{y}=1\mid\mathrm{x})=\theta_0+\theta_1\mathrm{x}_1+\theta_2\mathrm{x}_2+\ldots+\theta_n\mathrm{x}_n$$

而实际应用中，概率p与因变量往往是非线性的，为了解决该类问题，我们引入了logit变换，使得logit(p)与自变量之
间存在线性相关的关系，逻辑回归模型定义如下：

$$\mathrm{logit}(\mathrm{p})=In\left(\frac{p}{1-p}\right)=\theta_0+\theta_1\mathrm{x}_1+\theta_2\mathrm{x}_2+\ldots+\theta_n\mathrm{x}_n$$

通过推导，概率p变换如下，这与Sigmoid函数相符，也体现了概率p与因变量之间的非线

性关系。以0.5为界限，预测p大于0.5时，我们判断此时y更可能为1，否则y为0。

$$p = \frac{1}{1 + e^{-(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_n x_n)}}$$

得到所需的Sigmoid函数后，接下来只需要和前面的线性回归一样，拟合出该式中n个参数θ即可。test17_05.py为绘制Sigmoid曲线，输出上图所示。

```
import matplotlib.pyplot as plt
import numpy as np

def Sigmoid(x):
    return 1.0 / (1.0 + np.exp(-x))

x= np.arange(-10, 10, 0.1)
h = Sigmoid(x)              #Sigmoid函数
plt.plot(x, h)
plt.axvline(0.0, color='k')   #坐标轴上加一条竖直的线（0位置）
plt.axhspan(0.0, 1.0, facecolor='1.0', alpha=1.0, ls='dotted')
plt.axhline(y=0.5, ls='dotted', color='k')
plt.yticks([0.0, 0.5, 1.0])  #y轴标度
plt.ylim(-0.1, 1.1)         #y轴范围
plt.show()
```

由于篇幅有限，逻辑回归构造损失函数J函数，求解最小J函数及回归参数θ的方法就不在叙述，原理和前面小节一样，请读者下去深入研究。

二. LogisticRegression回归算法

LogisticRegression回归模型在Sklearn.linear_model子类下，调用sklearn逻辑回归算法步骤比较简单，即：

(1) 导入模型。调用逻辑回归LogisticRegression()函数。
(2) fit()训练。调用fit(x,y)的方法来训练模型，其中x为数据的属性，y为所属类型。
(3) predict()预测。利用训练得到的模型对数据集进行预测，返回预测结果。

代码如下：

```
from sklearn.linear_model import LogisticRegression  #导入逻辑回归模型
clf = LogisticRegression()
                            print clf
clf.fit(train_feature,label)
predict['label'] = clf.predict(predict_feature)
```

输出结果如下：

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

其中，参数penalty表示惩罚项（L1、L2值可选。L1向量中各元素绝对值的和，作用是产生少量的特征，而其他特征都是0，常用于特征选择；L2向量中各个元素平方之和再开根号，作用是选择较多的特征，使他们都趋近于0。）； C值的目标函数约束条件：s.t.||w||1<C，默认值是0，C值越小，则正则化强度越大。

三. 分析鸢尾花数据集

下面将结合Scikit-learn官网的逻辑回归模型分析鸢尾花示例，给大家进行详细讲解及拓展。由于该数据集分类标签划分为3类（0类、1类、2类），很好的适用于逻辑回归模型。

1. 鸢尾花数据集

在Sklearn机器学习包中，集成了各种各样的数据集，包括前面的糖尿病数据集，这里引入的是鸢尾花卉（Iris）数据集，它是很常用的一个数据集。鸢尾花有三个亚属，分别是山鸢尾（Iris-setosa）、变色鸢尾（Iris-versicolor）和维吉尼亚鸢尾（Iris-virginica）。

该数据集一共包含4个特征变量，1个类别变量。共有150个样本，iris是鸢尾植物，这里存储了其萼片和花瓣的长宽，共4个属性，鸢尾植物分三类。如表17.2所示：

表 17.2 鸢尾花数据集

列名	说明	类型
SepalLength	花萼长度	float
SepalWidth	花萼宽度	float
PetalLength	花瓣长度	float
PetalWidth	花瓣宽度	float
Class	类别变量。0 表示山鸢尾，1 表示变色鸢尾，2 表示维吉尼亚鸢尾。	int

iris里有两个属性iris.data，iris.target。data是一个矩阵，每一列代表了萼片或花瓣的长宽，一共4列，每一列代表某个被测量的鸢尾植物，一共采样了150条记录。

```
from sklearn.datasets import load_iris   #导入数据集iris
iris = load_iris()  #载入数据集
print iris.data
```

输出如下所示：

```
[[ 5.1  3.5  1.4  0.2]
 [ 4.9  3.   1.4  0.2]
 [ 4.7  3.2  1.3  0.2]
 [ 4.6  3.1  1.5  0.2]
 ....
 [ 6.7  3.   5.2  2.3]
 [ 6.3  2.5  5.   1.9]
 [ 6.5  3.   5.2  2. ]
 [ 6.2  3.4  5.4  2.3]
 [ 5.9  3.   5.1  1.8]]
```

target是一个数组，存储了data中每条记录属于哪一类鸢尾植物，所以数组的长度是150，数组元素的值因为共有3类鸢尾植物，所以不同值只有3个。种类为山鸢尾、杂色鸢尾、维吉尼亚鸢尾。

```
print iris.target          #输出真实标签
print len(iris.target)      #150个样本  每个样本4个特征
print iris.data.shape
```

```
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2]
150
(150L, 4L)
```

从输出结果可以看到，类标共分为三类，前面50个类标位0，中间50个类标位1，后面为2。下面给详细介绍使用决策树进行对这个数据集进行测试的代码。

2. 散点图绘制

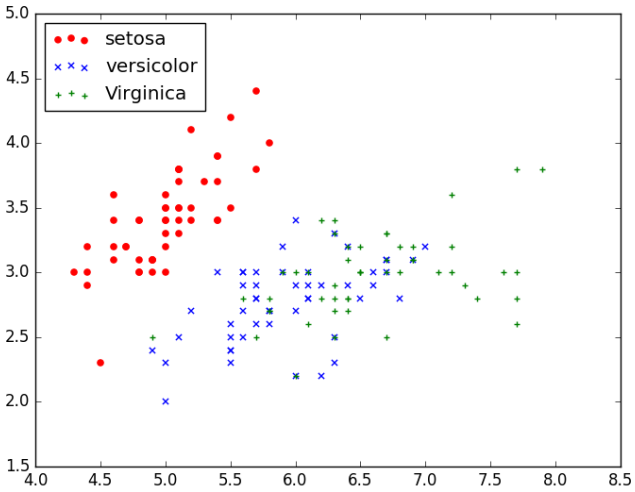
下列代码主要是载入鸢尾花数据集，包括数据data和标签target，然后获取其中两列数据或两个特征，核心代码为：X = [x[0] for x in DD]，获取的值赋值给X变量，最后调用scatter()函数绘制散点图。

```
import matplotlib.pyplot as plt
import numpy as np
from sklearn.datasets import load_iris    #导入数据集iris

#载入数据集
iris = load_iris()
print iris.data          #输出数据集
print iris.target         #输出真实标签
#获取花卉两列数据集
DD = iris.data
X = [x[0] for x in DD]
print X
Y = [x[1] for x in DD]
print Y

#plt.scatter(X, Y, c=iris.target, marker='x')
plt.scatter(X[:50], Y[:50], color='red', marker='o', label='setosa') #前50个样本
plt.scatter(X[50:100], Y[50:100], color='blue', marker='x', label='versicolor')
#中间50个
plt.scatter(X[100:], Y[100:],color='green', marker='+', label='Virginica') #后50
个样本
plt.legend(loc=2) #左上角 plt.show()
```

绘制散点图如图所示：



3. 逻辑回归分析

从图中可以看出，数据集线性可分的，可以划分为3类，分别对应三种类型的鸢尾花，下面采用逻辑回归对其进行分类预测。前面使用X=[x[0] for x in DD]获取第一列数据，Y=[x[1] for x in DD]获取第二列数据，这里采用另一种方法，iris.data[:, :2]获取其中两列数据（两个特征），完整代码如下：

```
import matplotlib.pyplot as plt
import numpy as np
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression

#载入数据集
iris = load_iris()
X = X = iris.data[:, :2]   #获取花卉两列数据集
Y = iris.target

#逻辑回归模型
lr = LogisticRegression(C=1e5)
lr.fit(X,Y)

#meshgrid函数生成两个网格矩阵
h = .02
x_min, x_max = X[:, 0].min() - .5, X[:, 0].max() + .5
y_min, y_max = X[:, 1].min() - .5, X[:, 1].max() + .5
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))

#pcolormesh函数将xx,yy两个网格矩阵和对应的预测结果Z绘制在图片上
Z = lr.predict(np.c_[xx.ravel(), yy.ravel()])
```

```
Z = Z.reshape(xx.shape) plt.figure(1, figsize=(8,6))
plt.pcolormesh(xx, yy, Z, cmap=plt.cm.Paired)

#绘制散点图
plt.scatter(X[:50,0], X[:50,1], color='red',marker='o', label='setosa')
plt.scatter(X[50:100,0], X[50:100,1], color='blue', marker='x',
label='versicolor')
plt.scatter(X[100:,0], X[100:,1], color='green', marker='s', label='Virginica')
  plt.xlabel('Sepal length')
plt.ylabel('Sepal width')
plt.xlim(xx.min(), xx.max())
plt.ylim(yy.min(), yy.max())
plt.xticks(())
plt.yticks(())
plt.legend(loc=2)
plt.show()
```

下面作者对导入数据集后的代码进行详细讲解。

lr = LogisticRegression(C=1e5)
lr.fit(X,Y)
初始化逻辑回归模型并进行训练，C=1e5表示目标函数。

x_min, x_max = X[:, 0].min() - .5, X[:, 0].max() + .5
y_min, y_max = X[:, 1].min() - .5, X[:, 1].max() + .5
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
获取的鸢尾花两列数据，对应为花萼长度和花萼宽度，每个点的坐标就是(x,y)。 先取X二维数组的第一列（长度）的最小值、最大值和步长h（设置为0.02）生成数组，再取X二维数组的第二列（宽度）的最小值、最大值和步长h生成数组， 最后用meshgrid函数生成两个网格矩阵xx和yy，如下所示：

```
[[ 3.8   3.82  3.84 ...,  8.36  8.38  8.4 ]
 [ 3.8   3.82  3.84 ...,  8.36  8.38  8.4 ]
 ...,
 [ 3.8   3.82  3.84 ...,  8.36  8.38  8.4 ]
 [ 3.8   3.82  3.84 ...,  8.36  8.38  8.4 ]]
[[ 1.5   1.5   1.5  ...,  1.5   1.5   1.5 ]
 [ 1.52  1.52  1.52 ...,  1.52  1.52  1.52]
 ...,
```

```
 [ 4.88  4.88  4.88 ...,  4.88  4.88  4.88]
 [ 4.9   4.9   4.9  ...,  4.9   4.9   4.9 ]]
```

Z = lr.predict(np.c_[xx.ravel(), yy.ravel()])

调用ravel()函数将xx和yy的两个矩阵转变成一维数组，由于两个矩阵大小相等，因此两个一维数组大小也相等。np.c_[xx.ravel(), yy.ravel()]是获取矩阵，即：

```
xx.ravel()
[ 3.8   3.82  3.84 ...,  8.36  8.38  8.4 ]
yy.ravel()
[ 1.5  1.5  1.5 ...,  4.9  4.9  4.9]
np.c_[xx.ravel(), yy.ravel()]
[[ 3.8   1.5 ]
 [ 3.82  1.5 ]
 [ 3.84  1.5 ]
 ...,
 [ 8.36  4.9 ]
 [ 8.38  4.9 ]
 [ 8.4   4.9 ]]
```

总结下：上述操作是把第一列花萼长度数据按h取等分作为行，并复制多行得到xx网格矩阵；再把第二列花萼宽度数据按h取等分，作为列，并复制多列得到yy网格矩阵；最后将xx和yy矩阵都变成两个一维数组，调用np.c_[]函数组合成一个二维数组进行预测。
调用predict()函数进行预测，预测结果赋值给Z。即：

```
Z = logreg.predict(np.c_[xx.ravel(), yy.ravel()])
[1 1 1 ..., 2 2 2]
size: 39501
```

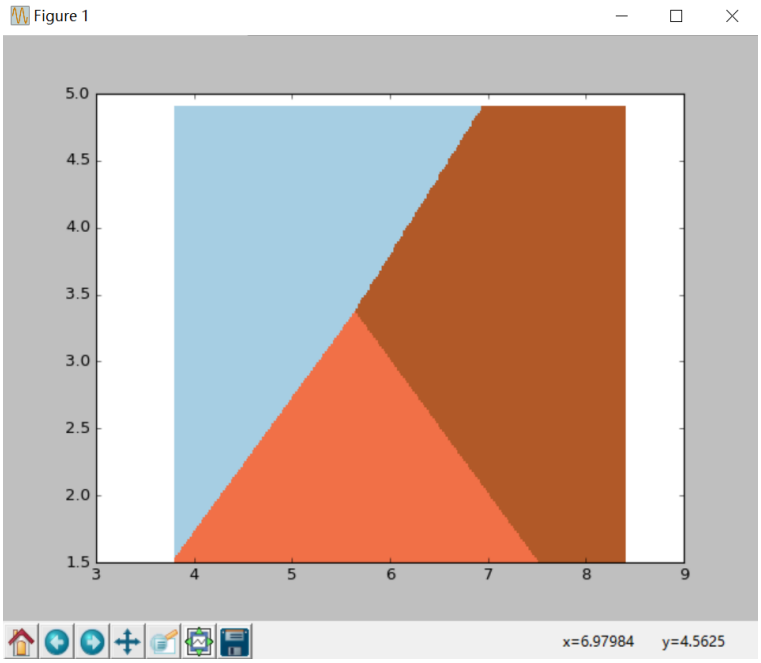
Z = Z.reshape(xx.shape)

调用reshape()函数修改形状，将其Z转换为两个特征（长度和宽度），则39501个数据转换为171*231的矩阵。Z = Z.reshape(xx.shape)输出如下：

```
[[1 1 1 ..., 2 2 2]
 [1 1 1 ..., 2 2 2]
 [0 1 1 ..., 2 2 2]
 ...,
 [0 0 0 ..., 2 2 2]
 [0 0 0 ..., 2 2 2]
 [0 0 0 ..., 2 2 2]]
```

plt.pcolormesh(xx, yy, Z, cmap=plt.cm.Paired)

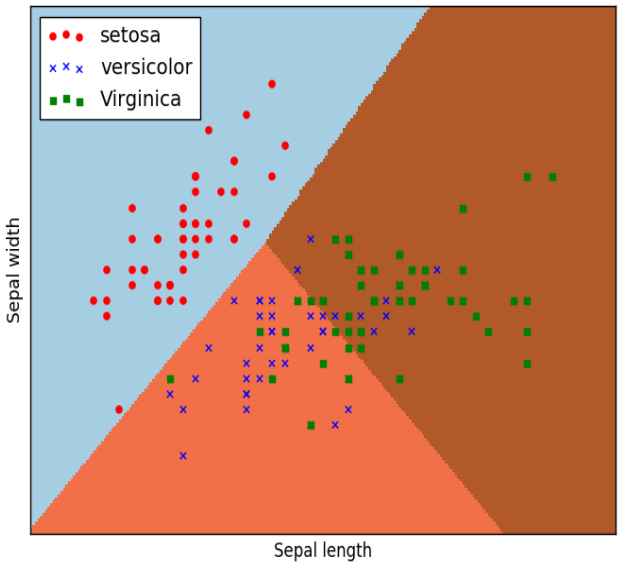
调用pcolormesh()函数将xx、yy两个网格矩阵和对应的预测结果Z绘制在图片上，可以发现输出为三个颜色区块，分布表示分类的三类区域。cmap=plt.cm.Paired表示绘图样式选择Paired主题。输出的区域如下图所示：



plt.scatter(X[:50,0], X[:50,1], color='red',marker='o', label='setosa')

调用scatter()绘制散点图，第一个参数为第一列数据（长度），第二个参数为第二列数据（宽度），第三、四个参数为设置点的颜色为红色，款式为圆圈，最后标记为setosa。

输出如下图所示，经过逻辑回归后划分为三个区域，左上角部分为红色的圆点，对应setosa鸢尾花；右上角部分为绿色方块，对应virginica鸢尾花；中间下部分为蓝色星形，对应versicolor鸢尾花。散点图为各数据点真实的花类型，划分的三个区域为数据点预测的花类型，预测的分类结果与训练数据的真实结果结果基本一致，部分鸢尾花出现交叉。



回归算法作为统计学中最重要的工具之一，它通过建立一个回归方程用来预测目标值，并求解这个回归方程的回归系数。本篇文章详细讲解了逻辑回归模型的原理知识，结合Sklearn机器学习库的LogisticRegression算法分析了鸢尾花分类情况。更多知识点希望读者下来后进行拓展，也推荐大学从Sklearn开源知识官网学习最新的实例。
希望文章对你有所帮助，祝自己和娜老师教师节快乐~接着工作去了

(By:Eastmount 2017-09-10 中午12点 http://blog.csdn.net/eastmount/)