CSDN新首页上线啦,邀请你来立即体验!(http://blog.csdn.net/)

更多 ▼

立即体

(https://g

utm_sour

CSDN

博客 (http://blog.csdn.net/?ref=toolbar)

学院 (http://edu.csdn.net?ref=toolbar)

下载 (http://db

Logistic回归及梯度上升算法

2016年12月06日 20:36:26



 \odot

Q

标签:机器学习(http://so.csdn.net/so/search/s.do?q=机器学习&t=blog)







大灰狼爱技术 ▼ (//my_csdn.net?ref=toolbar)

(http://w/nittentallogigbcooknometr/prostytetchtat/activity?

ref=toolltan) source=csdnblog

1975

Summit_Yue (http://blo...

+ 关注

粉丝

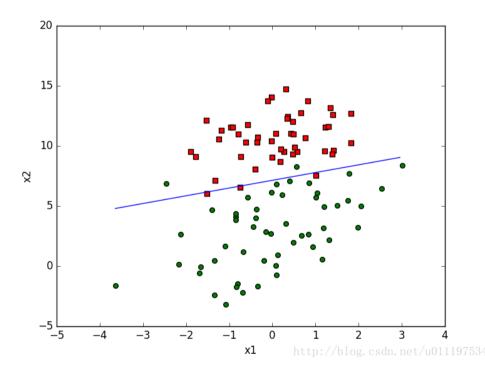
(http://blog.csdn.net/u011197534)

码云

开始学习机器学习算法的时候就接触了Logistic回归及梯度下降法,但是当时并没有深入去自己 推导一下公式,写一写代码,现在学习xgboost的时候又碰到Logistic回归相关的知识,干脆自 己推一遍,写一下代码吧。。

逻辑回归

逻辑回归可以用来进行回归与分类,两者仅有略微不同,主体算法是一样的,本文以分类进行讲解。如下 图二分类问题,我们希望找到一个直线(高维空间为超平面)来将数据划分开。



这样的线性边界可以表示为: $heta_0 x_1 + heta_1 x_2 + \ldots + heta_m x_m = heta^T x$ 上式右边x为向量。

我们取预测函数为Sigmoid函数,Sigmoid函数有一个很棒的特点是它的导数 则预测函数可表示为:

$$P(y=1|x; heta)=h_{ heta}(x)$$

$$P(y=0|x;\theta)=1-h_{\theta}(x)$$

将这两个式子合并一下:

他的最新文章

百分

44

更多文章 (http://blog.csdn.net/u011197534)

JavaScript作用域与作用域链 (http://b log.csdn.net/u011197534/article/de tails/78399838)

JavaScript事件对象 (http://blog.csd n.net/u011197534/article/details/78 399766)

JavaScript的绑定方法bind (http://blo g.csdn.net/u011197534/article/deta ils/78399758)











在线课程



腾讯云容器服务架构实现 介绍 ()

讲师: 董晓杰





高器技术在58同域的家践ourse/series_detail/ (http://edu.csdn.net/hu utm_source=blog9) 妍师owse/series_detail/ 73?utm_source=blog9)

TOP 返回顶部

他的热门文章

Windows10下VS2013+PCL1.8环境配置 (http://blog.csdn.net/u011197534/arti cle/details/52960394)

$$P(y|x; heta) = \left(h_{ heta}(x)
ight)^y \left(1-h_{ heta}(x)
ight)^{1-y}$$

显然:

当y=0的时候上式等价于 $P(y=0|x;\theta)=1-h_{\theta}(x)$ 当y=1的时候上式等价于 $P(y=1|x;\theta)=h_{\theta}(x)$

取似然函数

我们的目的就是求解似然函数的最大值,为了方便求解,我们取对数似然函数如下:

 \square

$$logL(heta) = \sum_{i=1}^m (y_i logh_{ heta}(x^{(i)}) + (1-y_i) log(1-h_{ heta}(x^{(i)})))$$

如此,我们就可以使用如下的式子进行梯度上升算法迭代更新 θ 的取值:

$$heta_j = heta_j + lpha \, rac{\partial log L(heta))}{\partial heta_j}$$

下面求解 $\frac{\partial log L(\theta)}{\partial \theta}$

$$\begin{split} \frac{\partial log L(\theta))}{\partial \theta_j} &= \sum_{i=1}^m (y_i \frac{1}{h_{\theta}(x^{(i)})} + (1-y_i) \frac{1}{1-h_{\theta}(x^{(i)})}) \frac{\partial h_{\theta}(x^{(i)})}{\partial \theta_j} \\ &= \sum_{i=1}^m (y_i \frac{1}{g(\theta^T x^{(i)})} + (1-y_i) \frac{1}{1-g(\theta^T x^{(i)})}) g(\theta^T x^{(i)}) (1-g(\theta^T x^{(i)})) x^{(i)} \\ &= \sum_{i=1}^m (y_i (1-g(\theta^T x^{(i)})) - (1-y_i) g(\theta^T x^{(i)})) x^{(i)} \\ &= \sum_{i=1}^m (y_i - g(\theta^T x^{(i)})) x^{(i)} \\ &= \sum_{i=1}^m (y_i - h_{\theta}(x^{(i)})) x^{(i)} \end{split}$$

所以权重的迭代更新式为:

$$heta_j = heta_j + lpha \sum_{i=1}^m (y_i - h_ heta(x^{(i)})) x^{(i)}$$

其中 α 为更新率

梯度上升

有了以上的逻辑回归的理论基础,下面我们编程实现这一步骤。就以第一张图的样本为例进行,样本维数为2维,采用梯度上升算法进行迭代。

迭代步数自己选择

批量梯度上升

批量梯度上升每进行一次迭代更新就会计算所有样本,因此得到的模型正确率比较高,但同时计算复杂度高,算法耗时。计算过程如下:

1.首先根据权重和训练样本计算估计值

3 5478

Logistic回归及梯度上升算法 (http://blo g.csdn.net/u011197534/article/detail s/53492915)

III 1969

决策树的Python实现 (http://blog.csdn. net/u011197534/article/details/52749 325)

470

Mac下Laravel的Homestead环境配置 (http://blog.csdn.net/u011197534/article/details/69487978)

1302

Deep Learning(深度学习)代码/课程/学 习资料整理【持续更新】(http://blog.cs dn.net/u011197534/article/details/53 053844)

3 790

⚠
内容举报

企 返回顶部

$$h = egin{pmatrix} x_1^{(1)} & x_2^{(1)} & x_3^{(1)} \ x_1^{(2)} & x_2^{(2)} & x_3^{(2)} \ \dots & \dots & \dots \ x_1^{(m)} & x_2^{(m)} & x_3^{(m)} \end{pmatrix} egin{pmatrix} heta_1 \ heta_2 \ heta_3 \end{pmatrix} = egin{pmatrix} \hat{y}_1 \ \hat{y}_2 \ \dots \ \hat{y}_m \end{pmatrix}$$

2.分算误差

3.迭代更新

$$w = w + egin{pmatrix} x_1^{(1)} & x_1^{(2)} & \dots & x_1^{(m)} \ x_2^{(1)} & x_2^{(2)} & \dots & x_2^{(m)} \ x_3^{(1)} & x_3^{(2)} & \dots & x_3^{(m)} \end{pmatrix} egin{pmatrix} e_1 \ e_2 \ \dots \ e_m \end{pmatrix}
ightarrow egin{pmatrix} \Delta heta_1 \ \Delta heta_2 \ \Delta heta_3 \end{pmatrix}$$

随机梯度上升

根据样本数量进行迭代,每计算一个样本就进行一次更新,过程如下: 1.计算 $x^{(i)}$ 样本对应的估计值

$$h = egin{pmatrix} x_1^{(i)} & x_2^{(i)} & x_3^{(i)} \end{pmatrix} egin{pmatrix} heta_1 \ heta_2 \ heta_3 \end{pmatrix}$$

2.计算误差

$$error = y_i - h(number)$$

注意,此处的误差是个数,不再是个向量3.迭代更新

$$w = w + lpha egin{pmatrix} x_1^{(i)} \ x_2^{(i)} \ x_3^{(i)} \end{pmatrix} error$$

以上步骤更新m次。

代码如下

⚠
内容举报

忘 返回顶部

```
1 import numpy as np
   2 import re
   3 from pandas import DataFrame
   4 import time as time
      import matplotlib.pyplot as plt
   5
   6
       import math
   7
6 def get_data(filename):
                                           #读取数据
0 9
         f = open(filename)
   10
           data = DataFrame(columns=['x0','x1','x2','label']) #构造DataFrame存放数据,列名为x与y
\square_{11}
          line = f.readline()
   12
          line = line.strip()
\[ \[ \[ \] \]_{13}
          p = re.compile(r'\s+')
                                          #由于数据由若干个空格分隔,构造正则表达式分隔
          while line.
   14
            line = line.strip()
   15
   16
             linedata = p. split(line)
             data.set_value(len(data), ['x0', 'x1', 'x2', 'label'], [1, float(linedata[0]), float(linedata[1]), int(lined
   17
   18 ata[2])]) #数据存入DataFrame
   19
             line = f. readline()
          return np. array(data. loc[:,['x0','x1','x2']]), np. array(data['label'])
   20
   21 def sigmoid(x):
          return 1.0/(1+np. \exp(-x))
   22
   23 def stocGradAscent(dataMat, labelMat, alpha = 0.01): #随机梯度上升
   24
          start time = time.time()
                                                        #记录程序开始时间
          m,n = dataMat.shape
   25
   26
          weights = np.ones((n, 1))
                                                        #分配权值为1
   27
          for i in range(m):
            h = sigmoid(np.dot(dataMat[i],weights).astype('int64')) #注意:这里两个二维数组做内积后得到的dtype是
   28
   29
      object, 需要转换成int64
   30
              error = labelMat[i]-h
               weights = weights + alpha*dataMat[i].reshape((3,1))*error #更新权重
   31
          duration = time.time()-start time
   32
   33
          print('time:',duration)
   34
          return weights
   35 def gradAscent(dataMat, labelMat, alpha = 0.01, maxstep = 1000): #批量梯度上升
   36
         start_time = time.time()
   37
           m, n = dataMat.shape
   38
           weights = np. ones((n, 1))
   39
          for i in range(maxstep):
             h = sigmoid(np.dot(dataMat,weights).astype('int64')) #这里直接进行矩阵运算
   40
             labelMat = labelMat.reshape((100,1))
                                                               #label本为一维,转成2维
   41
                                                                #批量计算误差
             error = labelMat-h
   42
              weights = weights + alpha*np.dot(dataMat.T,error) #更新权重
   43
           duration = time.time()-start_time
   44
           print('time:',duration)
   45
   46
           return weights
      def betterStoGradAscent(dataMat, labelMat, alpha = 0.01, maxstep = 150):
   47
   48
          start_time = time.time()
   49
           m, n = dataMat.shape
   50
           weights = np.ones((n.1))
   51
          for j in range(maxstep):
            for i in range(m):
   52
                alpha = 4/(1+i+j) + 0.01
                                                                #设置更新率随迭代而减小
   53
   54
                h = sigmoid(np.dot(dataMat[i],weights).astype('int64'))
   55
                 error = labelMat[i]-h
   56
                 weights = weights + alpha*dataMat[i].reshape((3,1))*error
   57
          duration = time.time()-start_time
   58
           print('time:',duration)
           return weights
   59
   60 def show(dataMat. labelMat. weights):
           #dataMat = np mat(dataMat)
   61
           #labelMat = np.mat(labelMat)
   62
   63
           m, n = dataMat.shape
```

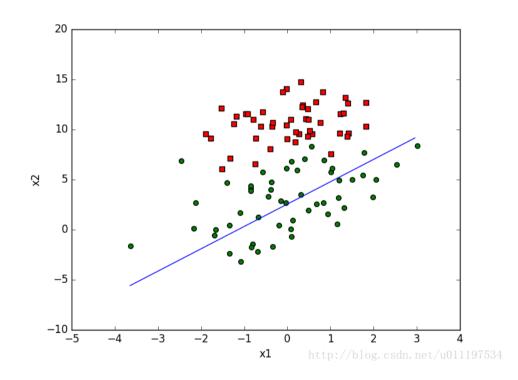
⚠
内容举报



```
64
            min_x = min(dataMat[:, 1])
   65
            max_x = max(dataMat[:, 1])
   66
            xcoord1 = []; ycoord1 = []
   67
            xcoord2 = []; ycoord2 = []
   68
            for i in range(m):
   69
                if int(labelMat[i]) == 0:
   70
                    xcoord1.append(dataMat[i, 1]); ycoord1.append(dataMat[i, 2])
凸71
                elif int(labelMat[i]) == 1:
o 72
                    {\tt xcoord2.\,append\,(dataMat[i,\ 1]);\,\,ycoord2.\,append\,(dataMat[i,\ 2])}
   73
            fig = plt.figure()
\square 74
            ax = fig.add_subplot(111)
   75
            ax.scatter(xcoord1, ycoord1, s=30, c="red", marker="s")
            ax.scatter(xcoord2, ycoord2, s=30, c="green")
₩ 76
   77
            x = np.arange(min_x, max_x, 0.1)
   78
            y = (-float(weights[0]) - float(weights[1])*x) / float(weights[2])
   79
            ax.plot(x, y)
   80
            plt.xlabel("x1"); plt.ylabel("x2")
   81
            plt.show()
   82
   83
       if __name__='__main__':
            dataMat, labelMat = get_data('data1.txt')
   84
            weights = gradAscent(dataMat, labelMat)
            show(dataMat, labelMat, weights)
```

效果

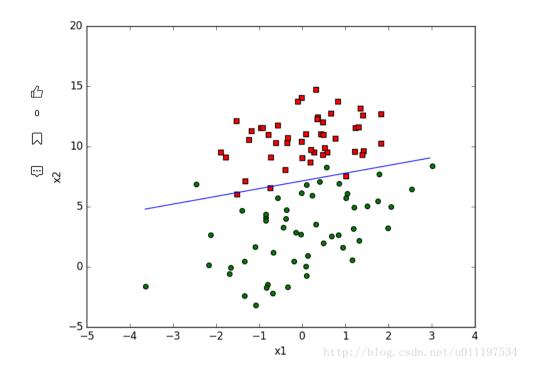
随机梯度:



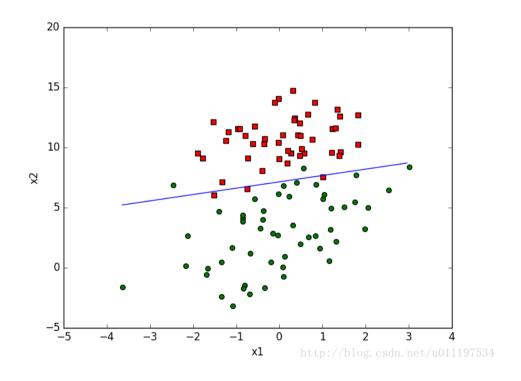
批量梯度:

⚠
内容举报

命 返回顶部



改进的随机梯度: 实际就是进行多次随机梯度,学习率随着迭代步数而减小。



⚠
内容举报

命 返回顶部

版权声明:本文为博主原创文章,未经博主允许不得转载。

发表你的评论 (http://my.csdn.net/forever2008250)		
ssfaker (/ssfaker) 2017-09-28 13:52 (/ssfa x 的)求导的前两行计算错误,修改一下 回复	1楼	
ο	相关文章推荐	

ubuntu下MySql部分配置 (http://bloq.csdn.net/u011197534/article/details/53408739)

安装与卸载ubuntu 14.04下安装mysqlsudo apt-get install mysql-serverubuntu 14.04下完全卸载mysqlsudo rm /var/li b/mys...

📵 u011197534 (http://blog.csdn.net/u011197534) 2016年11月30日 15:27 🔘 250

在phpmyadmin中创建存储过程并在php中调用 (http://blog.csdn.net/coder9999/articl...

语法: CREATE PROCEDURE p() BEGIN END CREATE PROCEDURE productpricing() BEGIN ...

🦱 coder9999 (http://blog.csdn.net/coder9999) 2017年04月16日 11:12 🛛 🖺 1174



不止20K, Python薪酬又飙升了??

2017年 Pytyhon薪酬曝光啦!看完后薪资报告后,同事说了一句:人生苦短,不学Python算白活.....

 $(http://www.baidu.com/cb.php?c=IgF_pyfqnHmknjnvPjc0IZ0qnfK9ujYzP1f4PjDs0Aw-line) = (http://www.baidu.com/cb.php?c=IgF_pyfqnHmknjnvPjc0IZ0qnfK9ujYzP1f4PjDs0Aw-line) = (http://www.baidu.com/cb.php.) = (http://www.baidu.com/cb.php.com/cb.php.)$ 5Hc3rHnYnHb0TAq15HfLPWRznjb0T1YYujRvrH0YPW0vmH6zPAuh0AwY5HDdnHcsnWn1rjD0IgF_5y9YIZ0IQzquZR8mLPbUB48ugfEIAqspynETZ-YpAq8nWqdIAdxTvqdThP-

 $5yF_UvTkn0KzujYz0AFV5H00TZcqn0KdpyfqnHRLPjnvnfKEpyfqnHc4rj6kP0KWpyfqP1cvrHnz0AqLUWYs0ZK45HcsP6KWThnqnHDdn6)$

Python3《机器学习实战》学习笔记(六):Logistic回归基础篇之梯度上升算法(http://blo...

本文从Logistic回归的原理开始讲起,补充了书上省略的数学推导。本文可能会略显枯燥,理论居多,Sklearn实战内容会放 在下一篇文章。自己慢慢推导完公式,还是蛮开心的一件事。...

🌄 c406495762 (http://blog.csdn.net/c406495762) 2017年08月30日 20:18 🕮4637

Logistic回归和梯度上升算法 (http://blog.csdn.net/whai362/article/details/51860379)

—. Logistic回归原理Logistic回归是一种广义线性回归,常用的分类器函数是Sigmoid函数,其公式如下: σ(z)=11+e-z\si $gma(z) = \frac{1}{1+e^{-z}...}$

🥙 whai362 (http://blog.csdn.net/whai362) 2016年07月08日 16:01 □ 4324

机器学习-Logistic回归之使用随机梯度上升算法预测病马死亡率 (http://blog.csdn.net/she...

运行环境: ubuntu16.10+MATLAB2016a数据集:该数据集来自2010年1月11日的UCI机器学习数据库,该数据最早由加拿 大安大略省圭尔夫大学计算机系Mary McLeish和Matt ...



🐒 shenliwuji (http://blog.csdn.net/shenliwuji) 2017年01月24日 23:32 🖫126

内容举报 TOP

返回顶部

⚠



AI 工程师职业指南

我们请来商汤、杜邦、声智、希为、58同城、爱因互动、中科视拓、鲁朗软件等公司 AI 技术一线的专 家,请他们从实践的角度来解析 AI 领域各技术岗位的合格工程师都是怎样炼成的。

(http://www.baidu.com/cb.php?c=IgF_pyfgnHmknjfzrj00IZ0gnfK9ujYzP1f4Pjnd0Aw-

5Hc4nj6vPjm0TAq15Hf4rjn1n1b0T1Y3rHm1rHmvPvD3n1T1m1cY0AwY5HDdnHcsnWn1rjD0IqF 5y9YIZ0lQzqMpqwBUvqoQhP8QvIGIAPCmqfEmvq rA7Wuj0YmhP9PARvujmYmH0vm1qdIAdxTvqdThP-

5HDknWF9mhkEusKzujYz0AFV5H00TZcqn0KdpyfqnHRLPjnvnfKEpyfqnHnsnj0YnsKWpyfqP1cvrHnz0AqLUWYs0ZK45HcsP6KWThnqnH6YPHn)

[机器学习]Logistic回归梯度上升法与改进的随机梯度上升算法 (http://blog.csdn.net/sinat...

htt导/sbp810050504.blog.51cto.com/2799422/1608064/这个网址解释了多维空间的sigmoid函数与梯度上升算法的原



logistics回归--梯度上升算法以及改进--用于二分类 (http://blog.csdn.net/Gentle_Guan/a...

1.sigmoid函数应用 logistics回归是用来分类的,并且属于监督学习,分类也是仅限于二分类,就是结果非0即1 (这种函数通 常称作跃阶函数) 这个时候就出现问题了 01之间的分界点怎么处理?...



🦱 Gentle_Guan (http://blog.csdn.net/Gentle_Guan) 2017年07月30日 22:15 🛚 🕮443

Logistic回归梯度上升分类法 (http://blog.csdn.net/qq_34352010/article/details/5174...

#coding=utf-8 from numpy import * import re def load data(): """加载数据""" data=[]:label=[] ...



🤹 qq_34352010 (http://blog.csdn.net/qq_34352010) 2016年06月23日 13:52 🛚 🕮 190

【机器学习】Logistic回归的梯度上升法 (http://blog.csdn.net/HerosOfEarth/article/det...

现实生活中有一种问题,输出值只有两种情况:yes or no.这类问题常见有:电子邮箱中的垃圾邮件分类(spam or not spa m),肿瘤为良性或者恶性等。在这些问题中,我们想预测的变量v,可以...



🌇 HerosOfEarth (http://blog.csdn.net/HerosOfEarth) 2016年07月21日 22:43 皿3321

梯度上升算法迭代过程和数学原理 (http://blog.csdn.net/berg_weald/article/details/701...

梯度上升算法在统计学习中使用广泛,比如Logistic使用梯度上升优化每一个属性的对应的权值参数 即 Z = W0X0 + W1X 1 + W2X2+ +WnXn 中的w参数 ...

(berg_weald (http://blog.csdn.net/berg_weald) 2017年04月12日 17:41

机器学习算法(优化)之一:梯度下降算法、随机梯度下降(应用于线性回归、Logistic回归...

本文介绍了机器学习中基本的优化算法—梯度下降算法和随机梯度下降算法,以及实际应用到线性回归、Logistic回归、矩阵 分解推荐算法等ML中。...

qq_34531825 (http://blog.csdn.net/qq_34531825)2016年09月01日 09:142016年09月01日 09:14

Logistic回归与梯度下降法 (http://blog.csdn.net/ACdreamers/article/details/44657979)

Logistic回归为概率型非线性回归模型,是研究二分类观察结果与一些影响因素之间关系的一种多变量分析方法。通常的问题 是,研究某些因素条件下某个结果是否发生,比如医学中根据病人的一些症状来判断它是...



🧶 ACdreamers (http://blog.csdn.net/ACdreamers) 2015年03月26日 22:46 🕮 10014

⚠

梯度下降法和拟牛顿法(四):在Logistic回归中的应用(http://blog.csdn.net/z5718265/...

生活中我们有一个简单的经验:某一件事情发生的概率与另一些事情成正相关关系。比如说,一个开宝马车的人是百万富翁的 概率是很大的。使用数学记号可记为: P(Y=1) xwTXP(Y=1) propto w ...



🦠 z5718265 (http://blog.csdn.net/z5718265) 2016年07月16日 17:30 🕮262

内容举报



Logistic回归与梯度下降法 (http://blog.csdn.net/haluoluo211/article/details/52029388)

http://blog.csdn.net/acdreamers/article/details/44657979 Logistic回归为概率型非线性回归模型,是研究二分类观察结 果与一些影响因素之...



🥞 haluoluo211 (http://blog.csdn.net/haluoluo211) 2016年07月25日 23:34 🛚 🕮 268

Logistic回归原理【似然函数与梯度选择】 (http://blog.csdn.net/TaoTaoFu/article/detai...

Logistic回归为概率型非线性回归模型,是研究二分类观察结果与一些影响因素之间关系的一种 多变量分析方法。通常的问题 是,研究某些因素条件下某个结果是否发生,比如医学中根据病人的一些症状来判断...

🧽....jaoTaoFu (http://blog.csdn.net/TaoTaoFu) 2016年10月16日 18:24 □2226

logistic回归|梯度下降|牛顿法总结 (http://blog.csdn.net/u014664226/article/details/5...

1.logistic回归模型logistic回归是用线性模型解决分类问题的算法 考虑现在有一个样本集合,样本特征有两维,要用一条直线 作为这两类的分界线,如下图所示也就是说logistic算法就是...



👣 u014664226 (http://blog.csdn.net/u014664226) 2016年06月16日 11:17 🕮2044



坐标上升算法总结(文档c++代码) (http://download.csdn.net/detail/nu...

(http://downloa

2012年12月20日 13:57 1.08MB

坐标上升算法 (Coordinate Ascent)及C++编程实现 (http://blog.csdn.net/NUPTboyZ...

坐标上升算法 (Coordinate Ascent) 及C++编程实现 编程实现: #include using namespace std; #define f(x1,...



이 NUPTboyZHB (http://blog.csdn.net/NUPTboyZHB) 2012年12月20日 13:43 🕮 6640

理解Logistic回归算法原理与Python实现 (http://blog.csdn.net/chaipp0607/article/det...

一般的机器学习的实现大致都是这样的步骤: 1.准备数据,包括数据的收集,整理等等2.定义一个学习模型(learning funct ion model),也就是最后要用来去预测其他数据的那个模型 ...



🥌 chaipp0607 (http://blog.csdn.net/chaipp0607) 2017年05月19日 12:19 🖫 🕮 1178

机器学习--Logistic回归算法 (http://blog.csdn.net/zyl1042635242/article/details/454...

一、基本原理 假设有一些数据点,用一条直线对这些点进行拟合(该线称为最佳拟合直线),这个拟合过程就称作回 Sigmoid函数是一种阶跃函数,具体计算公式如下:



zyl1042635242 (http://blog.csdn.net/zyl1042635242) 2015年05月04日 21:04 🕮 477

⚠ 内容举报

TOP 返回顶部