

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

立即体

验

CSDN

博客 (//blog.csdn.net?ref=toolbar) 学院 (//edu.csdn.net?ref=toolbar)

下载 (//download.csdn.net?ref=toolbar) GitChat (//gitbook.cn/?ref=csdn)

更多 ▾



0



weixin_3506... (//my.csdn.net?ref=toolbar)

(//write.blog.csdn.net/page/edit?ref=toolbar)

ref=toolbar_source=csdnblog

王小草【机器学习】笔记--提升之XGBoost工具的应用

原创

2016年12月29日 16:44:10

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

1398

笔记整理时间：2016年12月29日

整理者：王小草

欢迎关注：

王小草的FM喜马拉雅主播频道：搜索账号名“好吧我真的叫王草”

王小草的个人微信公众号：bigdataML

王小草的CSDN博客地址：http://my.csdn.net/sinat_33761963 (http://my.csdn.net/sinat_33761963)

2016年的最后第三天，终于有阳光，连续一个月的咳嗽终于见好。

这是及其忙碌的一年，忙着适应从学校到社会的血腥，忙着在车水马龙的竞争里立足安身，忙着屈服于又抗争于的生活。

在整理“提升”算法的笔记时，我想每年的自己多像一个基函数，在梯度里不断塑造下一个更好的自己，直到实现目标函数最优才停止迭代。而生活不像函数，至少在百年之前，都不愿停止寻找更优。



+ 关注

(http://blog.csdn.net/sinat_33761963)

码云

原创

粉丝

喜欢

未开通
(https://gite
utm_sourc

78

225

1

他的最新文章

更多文章 (http://blog.csdn.net/sinat_33761963)

王小草【机器学习】笔记--提升 (http://blog.csdn.net/sinat_33761963/article/details/71272342)

06：Tensorflow的可视化工具Tensorboard的初步使用 (http://blog.csdn.net/sinat_33761963/article/details/62433234)

05：Tensorflow高级API的进阶--利用tf.contrib.learn建立输入函数 (http://blog.csdn.net/sinat_33761963/article/details/62433150)

1. XGBoost介绍

根据网络的好坏，下载需要一点点时间，我大概是花了5分钟。

第二步：进入工程，然后编译。

```
1 cd xgboost
2 make -j4
```

编译的过程大概不到1分钟吧。

编译成功的画面：

```
a - build/tree/updater_refresh.o
a - build/tree/updater_sync.o
a - build/tree/updater_skmaker.o
a - build/tree/updater_colmaker.o
a - build/tree/tree_model.o
a - build/gbm/gbtree.o
a - build/gbm/gblinear.o
a - build/gbm/gbm.o
a - build/c_api/c_api.o
a - build/c_api/c_api_error.o
cc@cc-910S3K-9310SK-910S3P-911S3K:~/PycharmProjects/xgboost$
```

注意：

我机子上有2个python版本，一个是安装anaconda中带着的python，一个是我自己下载python来单独安装的。我的系统默认的是后者，所以上安装的XGBoost是安装在默认的python中的。

现在我打开pycharm,输入import xgboost as xgb,并没有红色波浪线的报错，表示可以成功使用了！

3.XGBoost实践

安装好了之后，我们来尝试着学习与使用这个工具。以下介绍一些从简到繁的案例。

3.1 官网get start小案例

7356

01：一文入门谷歌深度学习框架Tensorflow (http://blog.csdn.net/sinat_33761963/article/details/56286408)

5886

分类算法之逻辑回归--理论+案例+代码 (http://blog.csdn.net/sinat_33761963/article/details/51693836)

4595

预处理数据的方法总结（使用sklearn-preprocessing）(http://blog.csdn.net/sinat_33761963/article/details/53433799)

4020

相关推荐

xgboost的使用简析 (http://blog.csdn.net/John159151/article/details/45549143)

xgboost原理 (http://blog.csdn.net/a819825294/article/details/51206410)

XGBoost：在Python中使用XGBoost (http://blog.csdn.net/zc02051126/article/details/46771793)

xgboost原理及应用 (http://blog.csdn.net/wuzhongdehua1/article/details/52488424)

官方文档分别给出了4中语言的小例子，这里只讲解python的。

样本数据说明：

读取的数据是工程里自带的数据，在工程目录下的/demo/data/文件夹下。打开数据文件，格式是这样的：

1	1 3:1 10:1 11:1 21:1 30:1 34:1 36:1 40:1 41:1 53:1 58:1 65:1 69:1 77:1 86:1 88:1 92:1 95:1 102:1 105:1 117:1 124:1
2	0 3:1 10:1 20:1 21:1 23:1 34:1 36:1 39:1 41:1 53:1 56:1 65:1 69:1 77:1 86:1 88:1 92:1 95:1 102:1 106:1 116:1 120:1
3	0 1:1 10:1 19:1 21:1 24:1 34:1 36:1 39:1 42:1 53:1 56:1 65:1 69:1 77:1 86:1 88:1 92:1 95:1 102:1 106:1 116:1 122:1
4	1 3:1 9:1 19:1 21:1 30:1 34:1 36:1 40:1 42:1 53:1 58:1 65:1 69:1 77:1 86:1 88:1 92:1 95:1 102:1 105:1 117:1 124:1
5	0 3:1 10:1 14:1 22:1 29:1 34:1 37:1 39:1 41:1 54:1 58:1 65:1 69:1 77:1 86:1 88:1 92:1 95:1 98:1 106:1 114:1 120:1

每一行是一个观测样本，第一列是label,后面的列都是特征，一个特征由特征的索引，冒号，特征值组成，列与列之间是用空格隔开的。

在某样本点中没有出现的特征索引，说明该处特征值维0，也就是说，整个数据是一个稀疏的矩阵，所以只存储不为0的数据。

数据读取说明：

直接将path传入xgb.DMtrix（）中，会将数据变成DMtrix的格式，这是一个XGBoost自己定义的数据格式（就像numpy中有ndarray, pandas中有dataframe数据格式一样）。这个格式会将第一列作为label,其余的作为features。

参数说明：

模型中可以传入一些列参数，在训练之前，先把这些参数以map的形式定义好。参数有很多，下面的例子中是最基本的。

每个参数的意义在代码中都有注释，再此不赘述。



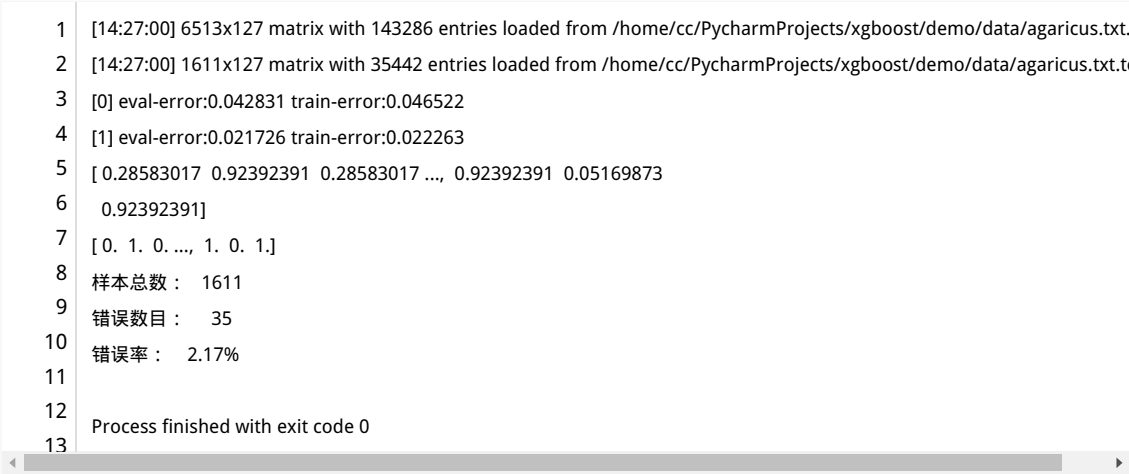
0



```
1 # /usr/bin/python
2 # -*- encoding:utf-8 -*-
3
4 import xgboost as xgb
5
6 # 读取数据
7 dtrain = xgb.DMatrix('/home/cc/PycharmProjects/xgboost/demo/data/agaricus.txt.train')
8 dtest = xgb.DMatrix('/home/cc/PycharmProjects/xgboost/demo/data/agaricus.txt.test')
9
10 # 设置参数，参数的格式用map的形式存储
11 param = {'max_depth': 2,          # 树的最大深度
12          'eta': 1,                # 一个防止过拟合的参数，默认0.3
13          'silent': 1,             # 打印信息的繁简指标，1表示简，0表示繁
14          'objective': 'binary:logistic'} # 使用的模型，分类的数目
15
16 num_round = 2 # 迭代的次数
17
18 # 看板，每次迭代都可以在控制台打印出训练集与测试集的损失
19 watchlist = [(dtest, 'eval'), (dtrain, 'train')]
20
21 # 训练模型
22 bst = xgb.train(param, dtrain, num_round, evals=watchlist)
23
24 # 做预测
25 preds = bst.predict(dtest)
26
27 # 打印结果
28 y_hat = preds
29 y = dtest.get_label()
30 print y_hat
31 print y
32
33 error_count = sum(y != (y_hat > 0.5))
34 error_rate = float(error_count) / len(y_hat)
35 print "样本总数：\t", len(y_hat)
36
```

```
37 print "错误数目：\t%d" % error_count
    print "错误率：\t%.2f%%" % (100 * error_rate)
```

来看看上面代码的输出：



```
1 [14:27:00] 6513x127 matrix with 143286 entries loaded from /home/cc/PycharmProjects/xgboost/demo/data/agaricus.txt.
2 [14:27:00] 1611x127 matrix with 35442 entries loaded from /home/cc/PycharmProjects/xgboost/demo/data/agaricus.txt.
3 [0] eval-error:0.042831 train-error:0.046522
4 [1] eval-error:0.021726 train-error:0.022263
5 [ 0.28583017  0.92392391  0.28583017 ...,  0.92392391  0.05169873
6   0.92392391]
7 [0.  1.  0. ...,  1.  0.  1.]
8 样本总数： 1611
9 错误数目： 35
10 错误率： 2.17%
11
12 Process finished with exit code 0
13
```

3.2 自定义基函数与损失

在做提升的时候我们一般选取决策树这样的弱预测器来作为基函数，也可以使用逻辑回归。逻辑回归其实本身是一个强分类器，提升强分类器不一定会有更好的表现，但也具体问题具体分析。这个基函数f其实是可以根据实际需求来调整的，当然也可以自己构造。

下面的案例中我们不适用XGBboost自带的基函数，而是自己定义基函数，然后再传入模型中训练。



0



```
1  # /usr/bin/python
2  # -*- encoding:utf-8 -*-
3
4  import xgboost as xgb
5  import numpy as np
6
7
8  # 定义f: theta * x
9  def log_reg(y_hat, y):
10     p = 1.0 / (1.0 + np.exp(-y_hat))
11     g = p - y.get_label()
12     h = p * (1.0-p)
13     return g, h
14
15
16 # 定义误差率
17 def error(y_hat, y):
18     return 'error', float(sum(y.get_label() != (y_hat > 0.0))) / len(y_hat)
19
20
21 if __name__ == "__main__":
22     # 读取数据
23     dtrain = xgb.DMatrix('/home/cc/PycharmProjects/xgboost/demo/data/agaricus.txt.train')
24     dtest = xgb.DMatrix('/home/cc/PycharmProjects/xgboost/demo/data/agaricus.txt.test')
25
26     # 设置参数，参数的格式用map的形式存储
27     param = {'max_depth': 2, # 树的最大深度
28             'eta': 1, # 一个防止过拟合的参数，默认0.3
29             'silent': 1} # 打印信息的繁简指标，1表示简，0表示繁
30
31     num_round = 2 # 迭代的次数
32
33     # 看板，每次迭代都可以在控制台打印出训练集与测试集的损失
34     watchlist = [(dtest, 'eval'), (dtrain, 'train')]
35
36     # 训练模型
```

```
37 bst = xgb.train(param, dtrain, num_round, evals=watchlist, obj=log_reg, feval=error)
38
39 # 计算错误率
40 y_hat = bst.predict(dtest)
41 y = dtest.get_label()
42 print y_hat
43 print y
44 error = sum(y != (y_hat > 0))
45 error_rate = float(error) / len(y_hat)
46 print '样本总数 : \t', len(y_hat)
47 print '错误数目 : \t%4d' % error
48 print '错误率 : \t%.2f%%' % (100 * error_rate)
```

打印的结果：

```
1 [14:53:48] 6513x127 matrix with 143286 entries loaded from /home/cc/PycharmProjects/xgboost/demo/data/agaricus.txt.
2 [14:53:48] 1611x127 matrix with 35442 entries loaded from /home/cc/PycharmProjects/xgboost/demo/data/agaricus.txt.t
3 [0] eval-error:0.042831 train-error:0.046522
4 [1] eval-error:0.021726 train-error:0.022263
5 [-1.04997814 2.57504988 -1.04997814 ..., 2.57504988 -3.01916885
6 2.57504988]
7 [0. 1. 0. ..., 1. 0. 1.]
8 样本总数 : 1611
9 错误数目 : 35
10 错误率 : 2.17%
11
12 Process finished with exit code 0
13
```

显然，模型并不可观，仍然有35个错误的样本点。于是我们需要去调整参数优化模型，比如，将迭代的次数改成3，其他的不变，输出的结果如下，错误率降低到了0.62%



```

1 [14:54:55] 6513x127 matrix with 143286 entries loaded from /home/cc/PycharmProjects/xgboost/demo/data/agaricus.txt.
2 [14:54:55] 1611x127 matrix with 35442 entries loaded from /home/cc/PycharmProjects/xgboost/demo/data/agaricus.txt.t
3 [0] eval-error:0.042831 train-error:0.046522
4 [1] eval-error:0.021726 train-error:0.022263
5 [2] eval-error:0.006207 train-error:0.007063
6 [-1.83141637 1.79361176 -1.83141637 ..., 3.24044585 -3.8006072
7 3.24044585]
8 [0. 1. 0. ..., 1. 0. 1.]
9 样本总数 : 1611
10 错误数目 : 10
11 错误率 : 0.62%
12
13 Process finished with exit code 0
14

```

再比如，其他不变，只将最大深度max_depth改称3，结果如下，亮瞎眼睛，测试集的误差居然为0了！

```

1 [14:57:11] 6513x127 matrix with 143286 entries loaded from /home/cc/PycharmProjects/xgboost/demo/data/agaricus.txt.
2 [14:57:11] 1611x127 matrix with 35442 entries loaded from /home/cc/PycharmProjects/xgboost/demo/data/agaricus.txt.t
3 [0] eval-error:0.016139 train-error:0.014433
4 [1] eval-error:0 train-error:0.001228
5 [-3.03464127 3.02548742 -3.03464127 ..., 3.02548742 -3.26926422
6 3.02548742]
7 [0. 1. 0. ..., 1. 0. 1.]
8 样本总数 : 1611
9 错误数目 : 0
10 错误率 : 0.00%
11
12 Process finished with exit code 0
13

```

3.3 softmax多分类问题

下面案例介绍一个用softmax做多分类的问题。还是使用那个家喻户晓的iris数据集，通过一系列特征去预测花的种类，总共有3类。

看一眼数据集的格式是酱紫的，总共有5列，前4列是4个特征，最后一列是label。

可见这个Label是string类型，在输入模型前，我们需要把它转换成数字作为标识。

1	5.1,3.5,1.4,0.2,Iris-setosa
2	4.9,3.0,1.4,0.2,Iris-setosa
3	7.0,3.2,4.7,1.4,Iris-versicolor
4	6.4,3.2,4.5,1.5,Iris-versicolor
5	6.3,3.3,6.0,2.5,Iris-virginica
6	5.8,2.7,5.1,1.9,Iris-virginica

完整代码：

自己写了一个方法iris_type，目的是将string类型的花名，转换成Float类型的数字标识。

与前面的案例不同，这次我们需要自己去分训练集与测试集，可直接调用方法train_test_split即可。

另外一点与以上案例不同的是系数中'silent'设置为0，表示打印出更多信息，在下面输出的结果中，后面的一大段信息就是这个导致的。

另外，案例中还调用了逻辑回归模型来做对比，发现正确率低于前者。



0



```
1  # /usr/bin/python
2  # -*- encoding:utf-8 -*-
3
4  import xgboost as xgb
5  import numpy as np
6  from sklearn.cross_validation import train_test_split
7
8
9  def iris_type(s):
10     it = {'Iris-setosa': 0, 'Iris-versicolor': 1, 'Iris-virginica': 2}
11     return it[s]
12
13
14 if __name__ == "__main__":
15     # 读数据
16     path = "/home/cc/下载/深度学习笔记/提升/8.数据/4.iris.data"
17     data = np.loadtxt(path, dtype=float, delimiter=',', converters={4: iris_type})
18
19     # 将数据集拆分成feature与label两部分
20     x, y = np.split(data, (4, ), axis=1)
21
22     # 将特征与标签数据分别拆分成训练集与测试集
23     x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=1, test_size=50)
24
25     # 将训练集与测试集变成DMtrix的数据格式
26     train = xgb.DMatrix(x_train, label=y_train)
27     test = xgb.DMatrix(x_test, label=y_test)
28
29     watch_list = [(test, 'eval'), (train, 'train')]
30
31     # 设置参数
32     param = {'max_depth': 3,
33             'eta': 1,
34             'silent': 0,
35             'objective': 'multi:softmax',
36             'num_class': 3}
```



0



```
37
38     num_round = 3
39
40     # 训练模型
41     bsg = xgb.train(param, train, num_round, evals=watch_list)
42
43     # 预测模型
44     y_hat = bsg.predict(test)
45
46     # 计算误差
47     result = y_test.reshape(1, -1) == y_hat
48     print 'XGBoost正确率:\t', float(np.sum(result)) / len(y_hat)
49     print 'END.....\n'
50
51     # =====#
52     # 逻辑回归
53     lr = LogisticRegression(penalty='l2')
54     lr.fit(x_train, y_train.ravel())
55     y_hat2 = lr.predict(x_test)
56
57     # 计算误差
58     result2 = y_test.reshape(1, -1) == y_hat2
59     print '逻辑回归正确率:\t', float(np.sum(result2)) / len(y_hat2)
60     print 'END.....\n'
```

输出的结果：



0



```
1 [0] eval-merror:0.02  train-merror:0.02
2 [1] eval-merror:0.02  train-merror:0.02
3 [2] eval-merror:0.04  train-merror:0.01
4 正确率: 0.96
5 END.....
6
7 [15:57:15] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 2 extra nodes, 0 pruned nodes, max_depth=1
8 [15:57:15] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 8 extra nodes, 0 pruned nodes, max_depth=3
9 [15:57:15] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 4 extra nodes, 0 pruned nodes, max_depth=2
10 [15:57:15] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 2 extra nodes, 0 pruned nodes, max_depth=1
11 [15:57:15] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 8 extra nodes, 0 pruned nodes, max_depth=3
12 [15:57:15] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 6 extra nodes, 0 pruned nodes, max_depth=2
13 [15:57:15] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 2 extra nodes, 0 pruned nodes, max_depth=1
14 [15:57:15] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 6 extra nodes, 0 pruned nodes, max_depth=3
15 [15:57:15] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 8 extra nodes, 0 pruned nodes, max_depth=3
16
17 逻辑回归正确率: 0.88
18 END.....
19
```

更多案例请参看：

<https://github.com/dmlc/xgboost/tree/master/demo> (<https://github.com/dmlc/xgboost/tree/master/demo>)

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/weixin_35068028)

相关文章推荐

xgboost的使用简析 (<http://blog.csdn.net/John159151/article/details/45549143>)

前言——记得在阿里mllib实习的时候，大家都是用mllib下的GBDT来train model的。但由于mllib不是开源的，所以在公司外是不能够使用。后来参加kaggle比赛的时候，认识到一个GD...

 John159151 (<http://blog.csdn.net/John159151>) 2015年05月07日 02:43  19487





0



xgboost原理 (<http://blog.csdn.net/a819825294/article/details/51206410>)

文章内容可能会相对比较杂，读者可以点击上方目录，直接阅读自己感兴趣的章节。1.序 距离上一次编辑将近10个月，幸得爱可可老师（微博）推荐，访问量陡增。最近毕业论文与xgboost相关，于是重新写一下...

 a819825294 (<http://blog.csdn.net/a819825294>) 2016年04月21日 10:15  87208





就刚刚，Python圈发生一件大事！

都说人生苦短，要学Python！但刚刚Python圈发生的这件事，你们怎么看？真相在这里...

(http://www.baidu.com/cb.php?c=lgF_pyfqHmknjnvPjn0lZ0qnfK9ujYzP1f4PjDs0Aw-5Hc3rHnYnHb0TAq15HfLPWRznjb0T1d9nhckrymdmw9mWc1nH-B0AwY5HDdnHckrjfLnHn0lgF_5y9YIZ0lQzq-uZR8mLPbUB48ugfEIAqspynEmybz5LNYUNq1ULNzmvRqmhkEu1Ds0ZFb5HD0mhYqn0KsTWYs0ZNGujYkPHTYn1mk0AqGujYknWb3rjDY0APGujYLnWm4n1c0ULI85H00TZbqnW)

XGBoost：在Python中使用XGBoost (<http://blog.csdn.net/zc02051126/article/details/4677...>)

在Python中使用XGBoost下面将介绍XGBoost的Python模块，内容如下：* 编译及导入Python模块 * 数据接口 * 参数设置 * 训练模型 * 提前终止程序 * ...

 zc02051126 (<http://blog.csdn.net/zc02051126>) 2015年07月06日 11:27  66417

xgboost原理及应用 (<http://blog.csdn.net/wuzhongdehua1/article/details/52488424>)

1.背景 关于xgboost的原理网络上的资源很少，大多数还停留在应用层面，本文通过学习陈天奇博士的PPT地址和xgboost导读和实战 地址，希望对xgboost原理进行深入理解。 2.xgbo...



wuzhongdehua1 (<http://blog.csdn.net/wuzhongdehua1>) 2016年09月09日 16:29 1373

xgboost使用调参 (<http://blog.csdn.net/q383700092/article/details/53763328>)

github : <https://github.com/dmlc/xgboost> 论文参考 : <http://www.kaggle.com/blobs/download/forum-message-atta...>



q383700092 (<http://blog.csdn.net/q383700092>) 2016年12月20日 15:14 7363



人人都能看懂的 AI 入门课

本课程将讲述人工智能的现状、应用场景和入门方法，并通过运用 TensorFlow，使得受众能清晰了解人工智能的运作方式。

(http://www.baidu.com/cb.php?c=lgF_pyfqHmknjzrjc0IZ0qnfK9ujYzP1f4Pjn10Aw-5Hc4nj6vPjm0TAq15Hf4rjn1n1b0T1YzP1TsnAfk1nvrAR1uHKb0AwY5HDdnHckrjflnHn0lgF_5y9YIZ0lQzqMpgwBUvqoQhP8QvIGIAPCmgfEmvq_lyd8Q1R4uWI-n16kPWKWrrHnvnHRvnnBuyD4PHqdlAdxTvqdThP-5HDknWFWmhhkEusKzujYk0AFV5H00TZcqn0KdpyfqHRLPjnvnfKEpyfqHnsnj0YnsKWpyfqP1cvrHnz0AqLUWYs0ZK45HcsP6KWThnqn1bvPHT)

xgboost代码示例 (<http://blog.csdn.net/zhuqiuhi/article/details/72584376>)

之前写过很久了，怕新更新的xgboost不再适用，重新调试了一下代码，可运行，但数据得换成自己的，xgboost，都应该知道它的威力了，这里不再多说，欢迎一起讨论！ # coding=utf-8 im...



zhuqiuhi (<http://blog.csdn.net/zhuqiuhi>) 2017年05月20日 19:28 831

XGBoost：二分类问题 (<http://blog.csdn.net/zc02051126/article/details/46709599>)

二分类问题本文介绍XGBoost的命令行使用方法。Python和R的使用方法见<https://github.com/dmlc/xgboost/blob/master/doc/README.md>。 ...



zc02051126 (<http://blog.csdn.net/zc02051126>) 2015年07月01日 15:38 16324

[Python] xgboost在windows上的安装 (http://blog.csdn.net/qq_34264472/article/details/53...)

1. xgboost的简介Xgboost是Tianqi Chen大神实现的一个boost算法框架。在Kaggle比赛的很多题目中都大放异彩，是个值得强力推荐的机器学习实战框架。网上有很多windows...



qq_34264472 (http://blog.csdn.net/qq_34264472) 2016年11月20日 10:41 1490

Python下如何安装导入xgboost (<http://blog.csdn.net/u012654847/article/details/71155322>)



0



u012654847 (<http://blog.csdn.net/u012654847>) 2017年05月03日 21:08 926



windows下，在python中安装xgboost的简易方法，解决 error C3861: “sleep”: 找不到标识符...

1、介绍xgboost是一个boosting+decision trees的工具包，看微博上各种大牛都说效果很好，于是下载一个，使用了一下，安装步骤如下。2、BOOST编译安装github下载地址：...



Eddy_zheng (http://blog.csdn.net/Eddy_zheng) 2015年12月05日 11:27 6472

xgboost在windows上的安装的使用 (<http://blog.csdn.net/fyjthcy/article/details/50562756>)

1、介绍 xgboost是一个boosting+decision trees的工具包，看微博上各种大牛都说效果很好，于是下载一个，使用了一下，安装步骤如下。2、BOOST编译安装 github下载地...



fyjthcy (<http://blog.csdn.net/fyjthcy>) 2016年01月22日 16:24 12009

本人64位windows下安装最新版XGBoost，附操作步骤图 (http://blog.csdn.net/Ai_Smith/art...)



最新更新的XGBoost与老版安装方式不再一样，具体安装步骤如下



Ai_Smith (http://blog.csdn.net/Ai_Smith) 2016年12月02日 17:42 4185



Windows下python的xgboost-0.47安装 (<http://blog.csdn.net/yueshibin/article/details/5275...>)

环境：Windows10，PyCharm，Anaconda，Microsoft Visual Studio 2010; 自己用PyCharm，并安装了Anaconda2（建议安装anac...

 yueshibin (<http://blog.csdn.net/yueshibin>) 2016年10月07日 17:23  549



基于scikit learn的logistic回归实现 (http://blog.csdn.net/liu_xiao_cheng/article/details/527...)

(logistic regression)属于概率型非线性回归，它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。在流行病学研究中，经常需要分析疾病与各危险因素之间的定量关系，...

 liu_xiao_cheng (http://blog.csdn.net/liu_xiao_cheng) 2016年10月10日 16:07  834



随机森林对鸢尾花数据的两特征组合的分类结果 (<http://blog.csdn.net/hb707934728/article/d...>)

特征：花萼长度 + 花萼宽度 预测正确数目：123 准确率: 82.00% 特征：花萼长度 + 花瓣长度 预测正确数目：143 准确率: 95.33% 特征：花萼...

 hb707934728 (<http://blog.csdn.net/hb707934728>) 2017年04月17日 15:48  412



xgboost入门与实战（原理篇）(<http://blog.csdn.net/sb19931201/article/details/52557382>)

xgboost入门与实战（原理篇）前言：xgboost是大规模并行boosted tree的工具，它是目前最快最好的开源boosted tree工具包，比常见的工具包快10倍以上。在数据科学方面...

 sb19931201 (<http://blog.csdn.net/sb19931201>) 2016年09月16日 20:26  45250

xgboost中的数学原理 (<http://blog.csdn.net/a358463121/article/details/68617389>)

xgboost中的数学原理boosting翻译过来就是提升的意思，通过研究如果将许多个弱分类器集成在一起提升为一个强分类器就是多数boosting算法所研究的内容。其中最为经典的算法就是Adaboos...

 a358463121 (<http://blog.csdn.net/a358463121>) 2017年03月30日 23:29  2068

XGBoost-Python完全调参指南-参数解释篇 (<http://blog.csdn.net/wzmsltw/article/details/50...>)

关于XGBoost的参数，发现已经有比较完善的翻译了。故本文转载其内容，并作了一些修改与拓展。 原文链接见：<http://blog.csdn.net/zc02051126/article/detail...>



wzmsltw (<http://blog.csdn.net/wzmsltw>) 2016年03月27日 22:28 38707

XGBoost：参数解释 (<http://blog.csdn.net/zc02051126/article/details/46711047>)



0

XGBoost参数在运行XGboost之前，必须设置三种类型成熟：general parameters，booster parameters和task parameters：
General para...



zc02051126 (<http://blog.csdn.net/zc02051126>) 2015年07月01日 17:06 49453



大杀器xgboost指南 (http://blog.csdn.net/Bryan___/article/details/52056112)



本文不做深入探讨，仅供自己备忘 原文：<http://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost/>



Bryan___ (http://blog.csdn.net/Bryan___) 2016年07月28日 16:08 22706