scikit-learn 梯度提升树（GBDT）算法实战

原创 石头 机器学习算法那些事 2018-12-14

前言

上一篇文章对GBDT算法原理进行了总结，本文使用GBDT回归类探讨了损失函数对训练数据的灵敏度，并介绍了参数调参实例。文末给出代码链接，欢迎下载。

目录

scikit-learn GBDT类库概述

GBDT类库包含了GradientBoostingClassifier分类类和GradientBoostingRegressor回归类，回归类和分类类除了损失函数参数不相同，其他参数大致相同。之前有boosting族Adaboosting分类类的实践，因此，本文重点介绍GradientBoostingRegressor回归类的使用。

GBDT类库boosting框架参数

GBDT类包含了boosting框架和弱学习器，GBDT类使用CART决策树作为弱学习器，前面文章有介绍决策树的相关参数，这里主要介绍boosting框架参数。

1）n_estimators：最大弱学习器个数。n_estimators过小容易欠拟合，模型处于高偏差状态；n_estimators过大容易过拟合，模型处于高方差状态。n_estimators常和learning_rate结合使用，默认值是100。

2）learning_rate：每个弱学习器的权重缩减系数v，相当于正则化参数。考虑权重系数v，那么GBDT模型的迭代公式为：

$$f_{k+1}(x) = f_k(x) + vG_{k+1}(x)$$

其中，f(x)代表学习器模型，G(x)代表每次迭代的弱学习器，v为权重缩减系数($0<v\leq 1$)。系数 v的作用：达到相同的训练效果，含有权重缩减系数的模型需要更多的迭代次数，因此，通过设置 v 降低了模型的复杂度，防止过拟合。

3）subsample：从训练样本随机无放回的抽取一定比例的子样本集进行训练，达到降低模型方差的效果，但同时会增大模型偏差。

4）loss：GBDT算法的损失函数，分类模型和回归模型的损失函数是不同的。

分类模型：有对数损失函数"deviance"和指数损失函数"exponential"，默认是对数损失函数。一般用对数损失函数进行二元分类和多元分类。

回归模型：有均方差"ls"，绝对损失"lad"，Huber损失"huber"和分位数损失"quantile"。默认是均方差。如果训练数据较好，推荐使用 ls，训练数据含有噪声或异常点时，推荐使用抗噪音的损失函数"huber"。如果需要对训练集进行分段预测的时候，则采用损失函数"quantile"，表达式可参考上文。

5）alpha：这个参数只有GradientBoostingRegressor有，当我们使用Huber损失"huber"和分位数损失"quantile"时，需要制定分位数的值。默认是0.9，如果噪音较多，可以适当降低这个分位数的值。

GBDT回归类损失函数的灵敏度探讨

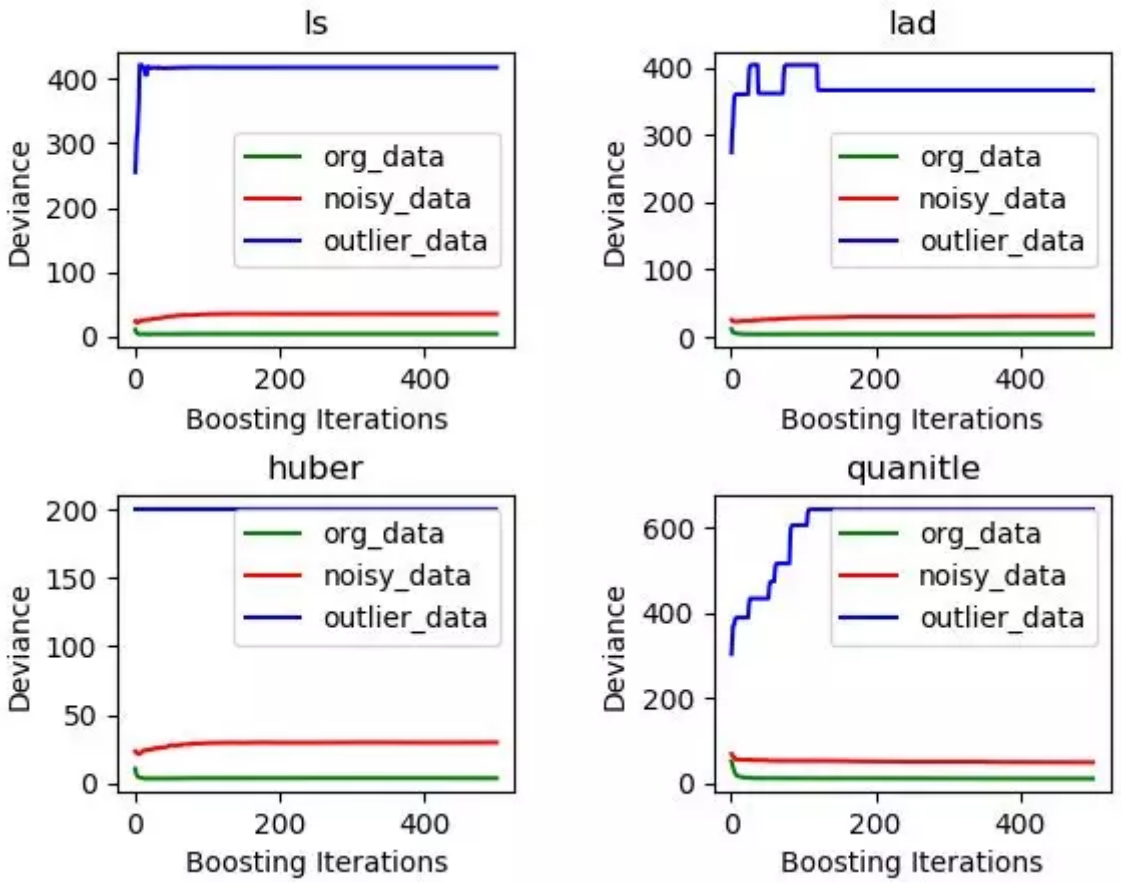
问题：当训练数据含有噪声或异常点时，对不同损失函数的GBDT回归模型影响有多大。

评价准则：均方差结果 。

数据类型：原始数据（org_data），数据含有随机噪声（noisy_data），数据含有异常值（outlier data）。

GBDT回归模型：损失函数分别为 ls，lad，huber，quantile，其他参数都是默认值。

下图为不同损失函数的回归模型对不同数据类型的预测误差情况：



上面四个图横坐标表示迭代次数，纵坐标表示预测值与实际值的均方差。

分析思路：比较同一模型对不同数据的均方差情况，就可以分析得到模型对噪声或异常值的敏感情况，异常值可以看成是加入了比较大的噪声。

结果分析：回归模型对加入噪声的数据不是很敏感，当数据加了一些异常值时，损失函数为huber的回归模型有最好的预测结果，因为均方差是最小的。因此，当你的数据有较大的噪声或异常值时，我建议你使用损失函数是huber的GBDT回归模型。

GBDT回归类的调参实例

前面有一篇文章介绍了boosting族的分类调参实例，因此本节介绍GBDT回归类的调参数实例，其他集成方法类的参数择优算法基本相同。

本节对损失函数的quantile的回归模型进行优化，如果参数都是默认值，训练数据的均方差情况：

```
# 利用默认参数，对原始数据进行回归
clf = GradientBoostingRegressor(loss='quantile')
clf.fit(diabetes_X_train, diabetes_y_train)
y_pred = clf.predict(diabetes_X_test)
```

```
#评价模型的均方差情况
mse_Default = mean_squared_error(y_pred,diabetes_y_test)
print 'mse_Default = %f' %mse_Default
```

输出结果：

```
mse_Default = 8.810127
```

使用sklearn.model_selection.GridSearchCV和sklearn.model_selection.cross_validate进行调参，本文采用GridSearchCV函数进行调参：

```
# 对boosting框架参数：对损失函数huber的分位数α和权重缩放率learning_rate进行调参
param_test1 = {'alpha':np.linspace(0.3,0.9,7),
               'learning_rate':np.linspace(0.2,0.9,8)}
gsearch1 = GridSearchCV(estimator = GradientBoostingRegressor(n_estimators=100,
                                        loss='quantile',random_state=10),
                        param_grid = param_test1,iid=False,cv=5)
gsearch1.fit(diabetes_X_train,diabetes_y_train)
print gsearch1.best_params_,gsearch1.best_score_
```

得到最佳参数：

```
{'alpha': 0.5, 'learning_rate': 0.2} 0.89743825326
```

用最佳参数拟合训练数据得到新模型，用该模型预测测试数据，得到均方差：

```
#用该模型预测测试数据，得到均方差
y_pred1 = clf.predict(diabetes_X_test)
mse_ParamOpt = mean_squared_error(y_pred1,diabetes_y_test)
print 'mse_ParamOpt = %f' %mse_ParamOpt
```

均方差结果：

```
mse_ParamOpt = 2.246403
```

参数优化后的均方差结果大大的降低了。仍可以使用GridSearchCV再次优化框架参数和决策树参数，因为均方差结果比较小了，本文就不继续优化参数了，感兴趣的童鞋可以继续优化。

总结

当训练数据含有较大的噪声或异常值时，建议选择损失函数为"huber"的GBDT回归模型，参数择优一般采用交叉验证的方法，重点是理解交叉验证的算法思想，若还有疑问，欢迎微信交流。

后台回复"GBDT"获取本文源码链接

参考：

https://www.cnblogs.com/pinard/p/6143927.html

推荐阅读

提升树算法原理小结

集成学习原理总结

比较全面的随机森林算法总结

比较全面的Adaboost算法总结

-END-