MSBD 5001 individual project

Description

This competition is about modeling the performance of computer programs. The dataset provided describes a few examples of running SGDClassifier in Python. The features of the dataset describes the SGDClassifier as well as the features used to generate the synthetic training data. The data to be analyzed is the training time of the SGDClassifier.

My project is written by python language, and it can be compiled by jupyter notebook directly. The necessary packages are shown below:

```
import numpy as np
import pandas as pd
import sklearn.feature_selection
from sklearn.feature_selection import f_regression
from sklearn import metrics
import xgboost as xgb
from xgboost import XGBRegressor
from sklearn.feature_selection import VarianceThreshold
from sklearn.model_selection import train_test_split, GridSearchCV, KFold
import matplotlib.pyplot as plt
import seaborn as sns
#%matplotlib inline
#from __future__ import print_function
#计算分类的正确率
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import LabelEncoder
from sklearn import preprocessing
from sklearn.preprocessing import scale
from xgboost import plot_importance
```

Key points and difficulties of the project

In this individual project, the main point is to find the basic linear relationship of running time and (max_iter* n_samples* n_features* n_classes/ n_jobs). In addition, too little data in the training data set makes the results of the model easy to overfit.

Data preprocess and Feature engineering

Observing the training data, we can drop the id and random_state directly, which have little impact on running time. Besides, after checking some material on internet, I found the penalty, l1_ratio, alpha, n_clusters_per_class, flip_y, scale, n_informative, not the major factors of the time. At last, I chose penalty, l1_ratio, alpha, n_clusters_per_class,

flip_y, scale, n_informative, max_iter, n_samples* n_features, (n_samples* n_features/ n_jobs), (n_classes* n_clusters_per_class) as feature. When the n_jobs is -1, I use the max values of n_jobs in dataset to replace them.

Model

I use SVM regression and XGBoost regression to predict the result, in the public test dataset, the xgboost model has better performance. The final parameter is shown below:

```
XGB = XGBRegressor(silent=1 ,#设置成1则没有运行信息输出，最好是设置为0.是否在运行升级时打印消息。
                   #booster='gblinear',#采用线性模型进行提升计算，默认是gbtree
                   learning_rate=0.2, # 如同学习率
                    min_child_weight=1,
                    # 这个参数默认是 1，是每个叶子里面 h 的和至少是多少，对正负样本不均衡时的 0-1 分类而言
                    #，假设 h 在 0.01 附近，min_child_weight 为 1 意味着叶子节点中最少需要包含 100 个样本。
                    #这个参数非常影响结果，控制叶子节点中二阶导的和的最小值，该参数值越小，越容易 overfitting。
                    max_depth=4, # 构建树的深度，越大越容易过拟合
                    gamma=0.2,  # 树的叶子节点上作进一步分区所需的最小损失减少,越大越保守，一般0.1、0.2这样子。
                    subsample=1, # 随机采样训练样本 训练实例的子采样比
                    max_delta_step=1,#最大增量步长，我们允许每个树的权重估计。
                    colsample_bytree=1, # 生成树时进行的列采样
                    reg_lambda=6,  # 控制模型复杂度的权重值的L2正则化项参数，参数越大，模型越不容易过拟合。
                    reg_alpha=0, # L1 正则项参数
                    scale_pos_weight=1, #如果取值大于0的话，在类别样本不平衡的情况下有助于快速收敛。平衡正负权重
                    n_estimators=360, #树的个数
                    seed=60 #随机种子
                   )
```

Actually, I also use cross validation to avoid overfitting, however, it did not perform well.