Windows下安装python版的XGBoost（Anaconda）

XGBoost是近年来很受追捧的机器学习算法，由华盛顿大学的陈天奇提出，在国内外的很多大赛中取得很不错的名次，要具体了解该模型，可以移步GitHub，本文介绍其在Widows系统下基于Git的python版本的安装方法。

需要用到三个软件：

python软件（本文基于Anaconda，因为自带很多库，比较方便）

Git for Windows

MINGW

假设都已经安装好了Anaconda，建议安装python2版本的，虽然python3也可以，但python2较为主流。再安装Git，装完之后在开始菜单里打开Git Bash，首先新建一个文件夹xgboostCode用于存放代码和下载文件，然后在刚刚打开的Bash终端中输入下面命令，定位目录。

[plain] view plain copy

$ cd /c/Users/xgboostCode/

然后用下面的命令从GitHub下载XGBoost。

[plain] view plain copy

$ git clone --recursive https://github.com/dmlc/xgboost

$ cd xgboost

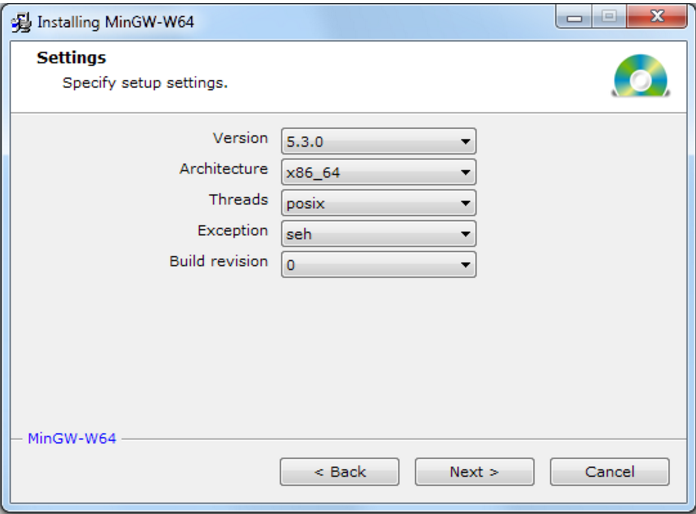
$ git submodule init

$ git submodule update

还需要下载64位的编译器MinGW-W64来编译下载的代码，上面已经提供下载地址。安装的界面如下：



然后选择x86\_64的Architecture，其他的选项默认。



按照指示完成安装，我电脑安装的路径为 C:\Program Files\mingw-w64\x86\_64-5.3.0-posix-seh-rt\_v4-rev0 。然后将C:\Program Files\mingw-w64\x86\_64-5.3.0-posix-seh-rt\_v4-rev0\mingw64\bin这个文件夹的路径添加到自己电脑的环境变量中去，具体步骤。

关闭Git Bash终端，再次打开，刚刚添加的路径变量就生效了，输入以下命令检查效果：

[plain] view plain copy

$ which mingw32-make

如果得到类似下面的结果，就说明配置成功了。

[plain] view plain copy

/c/Program Files/mingw-w64/x86\_64-5.3.0-posix-seh-rt\_v4-rev0/mingw64/bin/mingw32-make

再输入下面的命令：

[plain] view plain copy

$ alias make='mingw32-make'

下面就可以安装XGBoost了，首先定位到下载它的路径：

[plain] view plain copy

$ cd /c/Users/xgboostCode/xgboost

官方给的安装指导可能不成功，我们需要用下面的命令逐个地编译子模块：

[plain] view plain copy

$ cd dmlc-core

$ make -j4

$ cd ../rabit

$ make lib/librabit\_empty.a -j4

$ cd ..

$ cp make/mingw64.mk config.mk

$ make -j4

执行完成之后就可以在Anaconda中安装XGBoost的python模块了。在电脑的开始菜单中打开Anaconda Prompt，输入下面命令：

[plain] view plain copy

cd xgboostCode\xgboost\python-package

然后输入安装命令：

python setup.py install

安装已经完成，但在调用XGBoost之前，还应该将g++的运行库路径导入到os环境路径变量中，在Anaconda中打开Ipython，或者在python的命令行里，分别输入下面的命令：

[python] view plain copy

import os

mingw\_path = 'C:\\Program Files\\mingw-w64\\x86\_64-5.3.0-posix-seh-rt\_v4-rev0\\mingw64\\bin'

os.environ['PATH'] = mingw\_path + ';' + os.environ['PATH']

下面新建一个python文件，导入XGBoost，并测试下面的代码：

[python] view plain copy

import numpy as np

import xgboost as xgb

data = np.random.rand(5,10) # 5 entities, each contains 10 features

label = np.random.randint(2, size=5) # binary target

dtrain = xgb.DMatrix( data, label=label)

dtest = dtrain

param = {'bst:max\_depth':2, 'bst:eta':1, 'silent':1, 'objective':'binary:logistic' }

param['nthread'] = 4

param['eval\_metric'] = 'auc'

evallist = [(dtest,'eval'), (dtrain,'train')]

num\_round = 10

bst = xgb.train( param, dtrain, num\_round, evallist )

bst.dump\_model('dump.raw.txt')