小组作业：大规模机器学习算法的并行实现

选题的动机

- 目前工业界的应用对机器学习算法有两个基本要求：
  - 效果好
  - 能处理大规模的数据
- 为满足第一个要求，我们基本只有三类方法可选：决策树+Boosting, 随机森林，深度学习算法。
- 为满足第二个要求，我们需要把机器学习算法实现成可以同时在很多个CPU/GPU上并行运算的形式。
- 因此，这次小组作业的主题是“大规模机器学习算法的并行实现”。大家只要认真实现并弄清楚其中的细节和原理，基本上有能力到百度或阿里巴巴的数据分析部门去工作。

题目1：Random Forest的并行实现

- 随机森林是Kaggle比赛中常用算法，特点是效果好且很容易并行化。阿里的ODPS分布式平台上就有随机森林的并行化实现。
- RF的相关代码和论文见下一页。
- 主要任务：
  - 用C++实现一个随机森林算法
  - 把随机森林用多线程并行化
  - （可选）有能力和兴趣的同学，可以把随机森林改成可以在MPI上运行。MPI是一种分布式协议，按照它规定的形式来编程，可以把随机森林算法中的计算任务分配给多个（在不同机器上）CPU上分别进行计算，从而提高能处理的数据量和计算效率。
- 测试数据：第二次个人作业的数据集

Random Forest

1. Regression Trees and Classification Trees: Section 9.2 in [2]
2. Random Forest: [1] or Chapter 15 in [2]
3. **Source Codes:**

Python, Sklearn(http://scikit-learn.org/stable/)

C++, RT-Rank(https://sites.google.com/site/rtranking/)

Or search on GitHub

Reference:

[1] Breiman, “Random Forests”, Machine Learning, 45(1), 5-32, 2001.

[2] T. Hastie, R. Tibshirani and J. Friedman. Elements of Statistical Learning Ed. 2, Springer, 2009.

题目2：Gradient Boosted Decision Trees（GBDT）的并行实现

- 跟随机森林一样，GBDT也是一种常用算法，相关代码和论文见下一页。
- 主要任务：
  - 用C++实现一个GBDT算法
  - 把GBDT改成多线程实现
  - （可选）有能力和兴趣的同学，可以把GBDT改成可以在MPI上运行。MPI是一种分布式协议，按照它规定的形式来编程，可以把随机森林算法中的计算任务分配给多个（在不同机器上）CPU上分别进行计算，从而提高能处理的数据量和计算效率。
- 测试数据：第二次个人作业的数据集

Gradient Boosting

Gradient Boosting: [1], [2] and Chapter 10 in [3]

Source Codes:

Python, Sklearn(http://scikit-learn.org/stable/)

C++, RT-Rank(https://sites.google.com/site/rtranking/)

Or search on GitHub

Reference:

[1] J. Friedman, Greedy Function Approximation: A Gradient Boosting Machine, The Annals of Statistics, Vol. 29, No. 5, 2001.

[2] Friedman, Stochastic Gradient Boosting, 1999

[3] T. Hastie, R. Tibshirani and J. Friedman. Elements of Statistical Learning Ed. 2, Springer, 2009.

题目3：Regularized Greedy Forest（RGF）的并行实现

- RGF是Kaggle上那个300万美金奖金比赛的冠军算法。它属于decision tree+boosting这一类，网上有论文和C++的实现代码：
  - http://stat.rutgers.edu/home/tzhang/software/rgf/
- 主要任务：
  - 把RGF实现成多线程版本，可以利用多线程来提高计算速度。
  - （可选）有能力和兴趣的同学，可以把RGF改成可以在MPI上运行。MPI是一种分布式协议，按照它规定的形式来编程，可以把RGF算法中的计算任务分配给多个（在不同机器上）CPU上分别进行计算，从而提高能处理的数据量和计算效率。
- 测试数据：第二次个人作业的数据集

关于MPI

- 以下是一个用MPI来实现决策树+boosting的代码，大家可从中学习如何利用MPI来实现并行化。
  - http://machinelearning.wustl.edu/pmwiki.php/Main/Pgbrt

小组作业的评价要点

- 代码只能用C/C++实现，不能采用其他语言
- 由于多线程实现跟平台相关，推荐使用开源的Boost里提供的多线程
- 需要提交详细的实验报告，程序源代码，详细的配置文档
- 实验报告应该包括实现方法描述，测试结果（分类准确率，单线程和多线程的效率、效果比较），心得体会。
- 如有其他补充要求，将另行通知