Course Recommendation Using MPI for Python

Hazel Litwiller, ENGR-E517

Basic Information and How to Use This System

This system takes a student's major, GPA, and year in school as input and outputs a list of courses that are the best fit for the student's attributes.

Required Python modules: pandas, mpi4py

Scripts without MPI: reccomend_no_mpi.py, sklearnrecommend_no_mpi.py

These scripts require no special setup, aside from loading the required modules.

Scripts using MPI: reccomend.py, sklearnrecommend.py

I used Microsoft MPI v10.0 running on my local machine to execute these scripts with the command "mpiexec -n {numProcs} python {filename}" (16 processes worked best for the dataset this system was tested on). Additional configuration would likely be needed if running on BigRed3 or another supercomputer.

Parallelization: These scripts parallelize the task of recommending courses by spliting a dataset between all MPI processes. For example, the dataset used to test these programs has 17440 records, which means executing either MPI program on 16 processes would assign each process $\frac{17440}{16}$ records. Each process then computes its own list of best courses and sends it to process 0, which then selects the final top courses for the student.

Example Inputs

Get courses for an Freshman English major with a 3.62 GPA: main("ENG",3.62,1)

Generate a random major, gpa, and year: main(simulate=True)

Both MPI programs contain the options printResult, to toggle printing the resulting table of courses to the screen, and writeTimeToFile, which toggles writing the number of processes and execution time to a file.

IU Grade Distribution Dataset

This system was tested using the Fall 2020 and Spring 2021 Datasets from Indiana University's Grade Distribution Database. Each semester contains about 8000 course sections which, when all sections of the same course are combined, result in a dataset of about 4000 courses which are used to make the final predictions for the student.

A record from the Spring 2021 Dataset:

```
Institution Code                                        IUBLA
Inst Description                                   Bloomington
Term Code                                                4212
Term Description                                  Spring 2021
Session Code                                              13W
Session Description                             Thirteen Week
Academic Group Code                                       BUS
Academic Group Description          Kelley School of Business
Academic Organization Code                             BL-BUS
Academic Organization Description                    Business
Department Code                                           BUS
Course Subject                                          BUS-C
Catalog Number                                            204
Class #                                                  6233
Course Description                     BUSINESS COMMUNICATION
Course Topic                                              NaN
```

```
Instructor Name                          Cannon,Jeffrey Day
GPA Grades                                               21
Total Grades                                             24
Percent Majors                                         95.7
Avg Class Grade                                        3.69
Avg Student GPA                                       3.503
Percent A Grades                                       76.2
Percent B Grades                                       23.8
Percent C Grades                                        0.0
Percent D Grades                                        0.0
All Other Grades #                                        3
A+                                                        0
A                                                         9
A-                                                        7
B+                                                        3
B                                                         1
B-                                                        1
C+                                                        0
C                                                         0
C-                                                        0
D+                                                        0
D                                                         0
D-                                                        0
F                                                         0
P                                                         0
S                                                         0
I                                                         0
R                                                         0
NY                                                        0
NR                                                        0
NC                                                        0
W                                                         2
WX                                                        1
Other                                                     0
Data Freeze Date                                 06-11-2021
```

Functions and Code Contents

Both the Non-SKLearn and SKLearn versions of this course recommender follow a similar function structure. The following is a description of each function included in those scripts.

main(major,gpa,year,nCourses,simulate,printResult,writeTimeToFile):

major: The abbreviated version of a student's major (Ex. Computer Science → "CSCI")
gpa: The student's GPA
year: The student's year in school represented as an integer (Ex. Freshman → 1)
nCourses: The number of top courses to output (default=5)
simulate: If true, generates a random student major, gpa, and year to compute (default=False)
printResult: Whether the resulting course list should be printed (default=True)
writeTimeToFile: Whether the number of processes and execution time should be written to a CSV file (default=False)

Non-SKLearn: The main method subsets the course data for each process and generates a random student if neccesary. This method also handles the sending and receiving of data between processes and chooses the final *nCourses* courses to output based on the combined best list of courses received from each process.

SKLearn: The main method subsets the course data for each process, generates a random student if neccesary, and computes a dictionary of academic group codes pointing to arrays of the department codes they contain. This method also handles the sending and receiving of data between processes chooses the final *nCourses* courses to output based on the combined best list of courses received from each process.

preprocess():

Non-SKLearn: In this method records from the F20 and S21 datasets are aggregated by course, and all uneeded attributes are removed. This method outputs a dataframe containing the group, department, a string representation of the course subject and catalog number (Ex. "ENGR-E"+"317"→"ENGR-E317"), the course level as an integer, and the aggregated average student GPAs and total grades of each section of the course. A dictionary of academic group codes pointing to arrays of the department codes they contain is also computed and returned.

SKLearn: This version of the method takes in the students's major code. The F20 and S21 datasets are aggregated by course, and all uneeded attributes are removed. This method outputs a dataframe containing the a string representation of the course subject and catalog number group, whether the student's major is in the same academic group and the course, whether the student's major is the same as the department of the course, the course level as an integer, and the aggregated average student GPAs and total grades of each section of the course.

compareCourses():

Non-SKLearn: Computes a score for each course similar to the distance metric used in a nearest neighbor classifier (Higher score is better). Factors included in the score are the difference in GPA between the student's GPA and the average GPA for the course, the difference between the student's year in school and the level of the course, and whether the course shares an academic group and department code with the student's major. The resulting score and course name pairs are returned in a data frame.

SKLearn: Uses the SKLearn NearestNeighbors object to compute a distance between the student's attributes and the attributes of each course. Returns the $nCourses$ nearest courses to the student's data point with their corresponding distances.

chooseCourses():

Non-SKLearn: Selects the $nCourses$ courses with the largest score from the data frame outputted by *compareCourses()*

SKLearn: Selects the $nCourses$ courses with the smallest distance from the data frame outputted by *compareCourses()*

Results

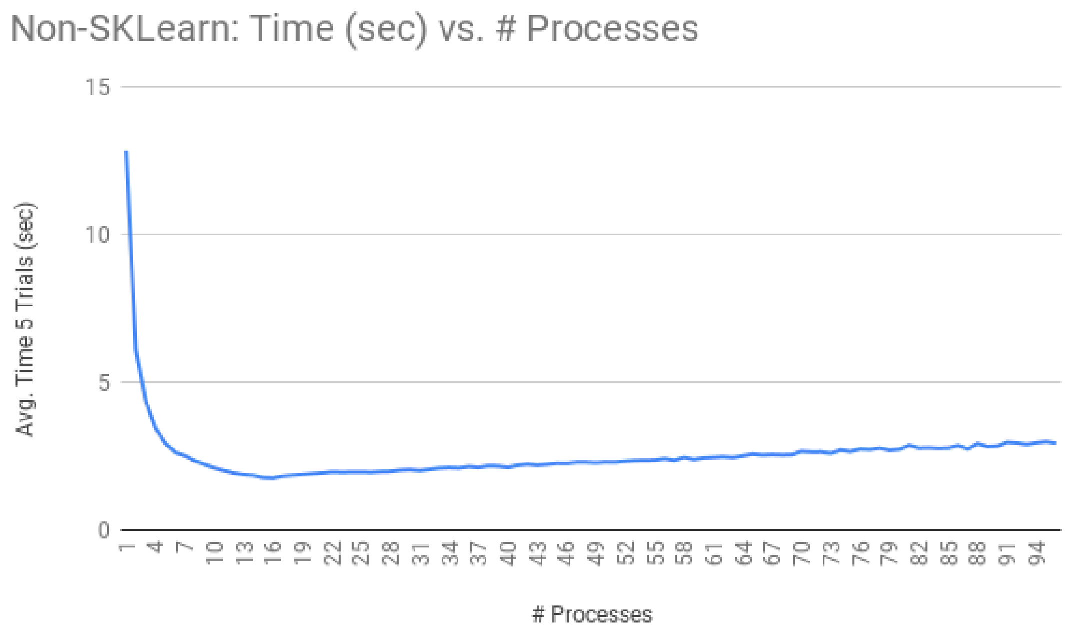
Both the Non-SKLearn and SKLearn MPI systems were tested on the combined Fall 2020 and Spring 2021 datasets. The following plots show the strong scaling and weak scaling test results.

No MPI Non-SKLearn Average Execution Time (Avg. 20 Trials): 12.73s

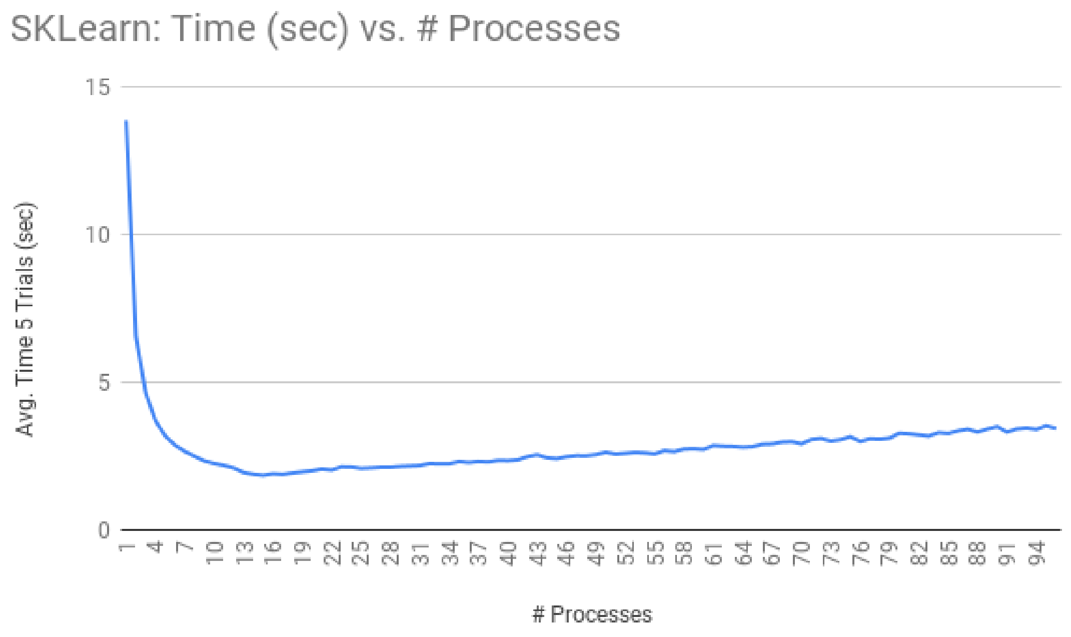
No MPI SKLearn Average Execution Time (Avg. 20 Trials): 13.52s

Strong Scaling:

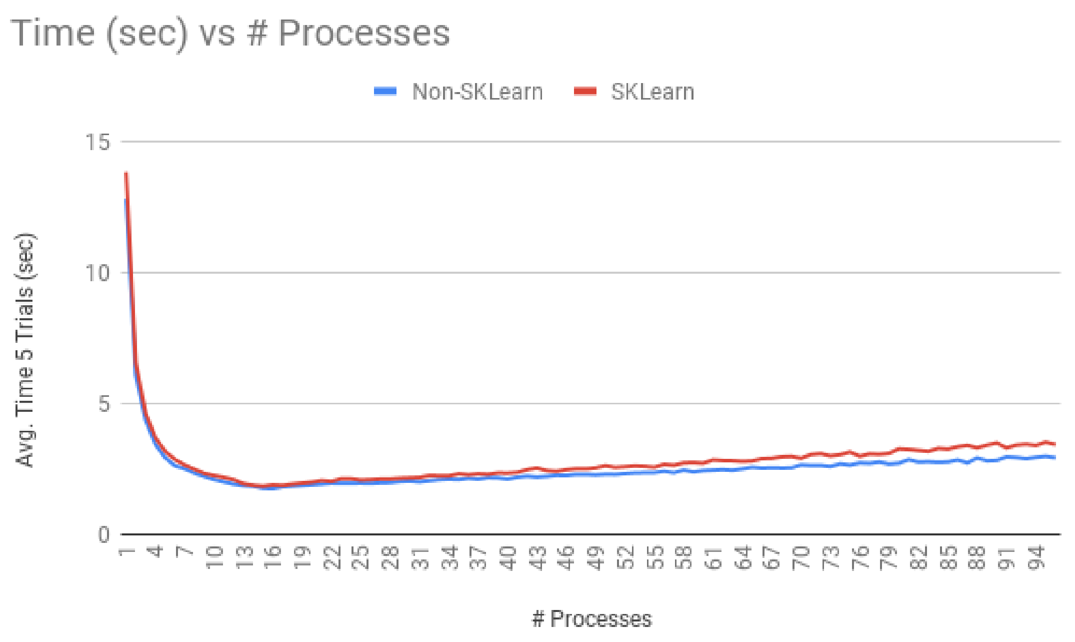
Tested on MPI process counts 1 to 96 (incremented by 1), using the entire dataset. Execution times are the average of 5 trials on the same process count.



Raw Data in strong_scaling_data_recommend.csv



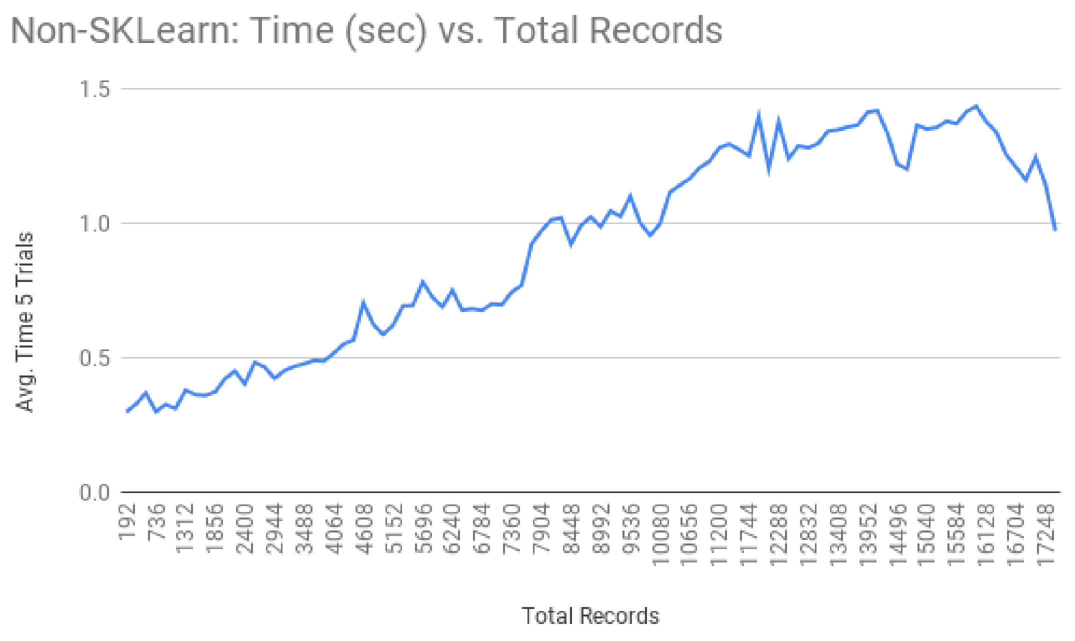
Raw Data in strong_scaling_data_sklearn_recommend.csv



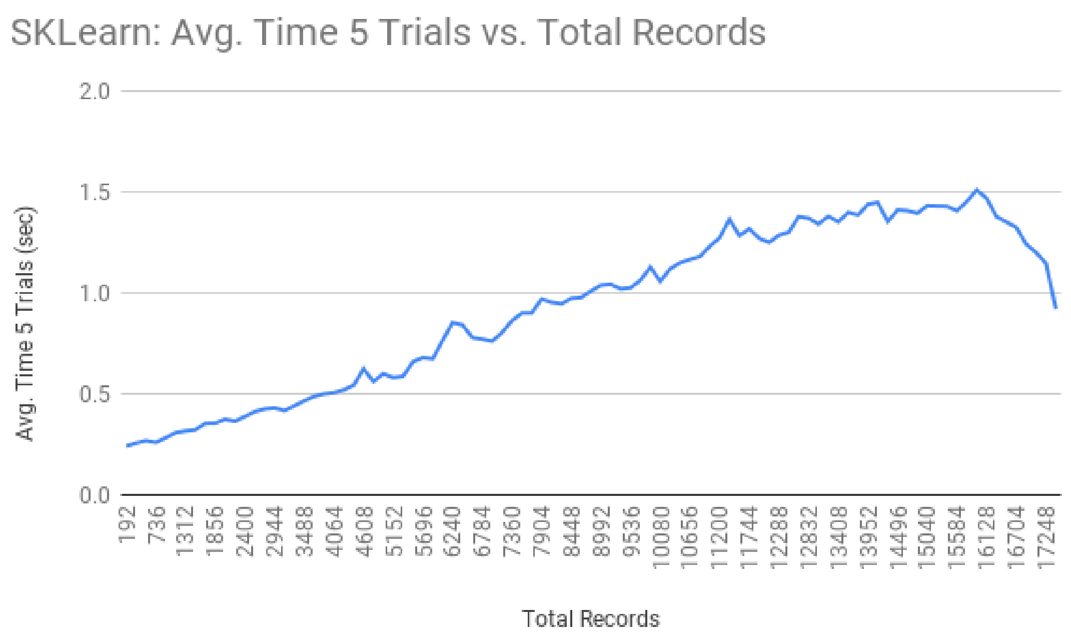
Both systems see a substantial execution time impovement from strong scaling up to approximately 16 processes. Adding more compute resources beyond this point harms the recommendation speed slightly for the dataset size used in testing (~17440 records). The SKLearn script has a slightly higher average execution time at most process counts, which is likely due to the overhead involved in the creation of an SKLearn NearestNeighbor object, but this difference is not particularly significant.

Weak Scaling:

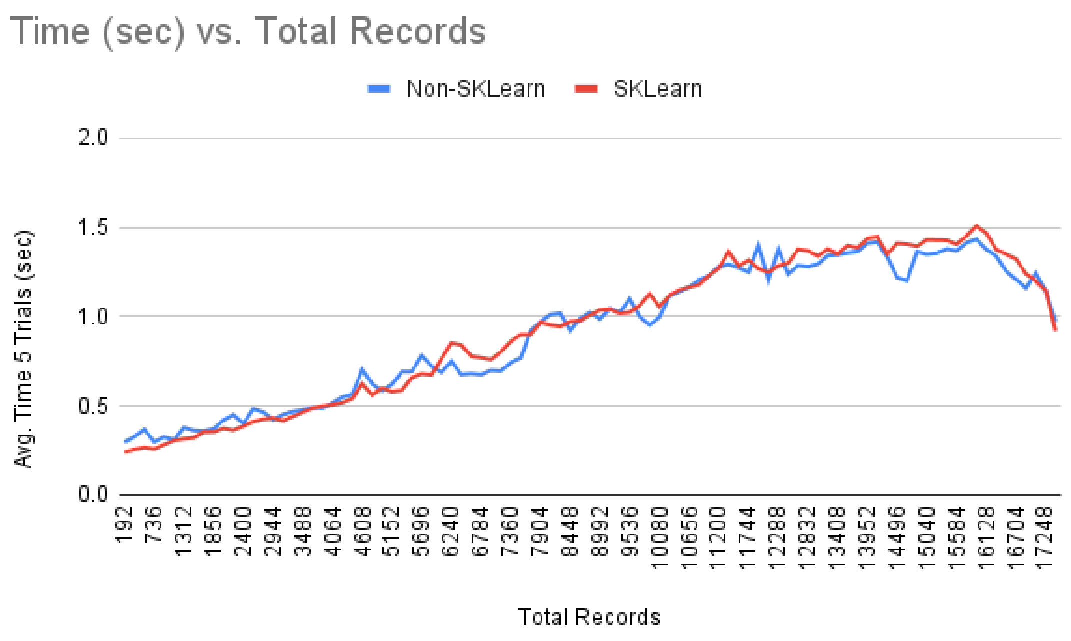
Tested on 16 MPI processes using between $\frac{1}{96} * length(dataset)$ and $length(dataset)$ records (incremented by$\frac{1}{96} * length(dataset)$ records). Execution times are the average of 5 trials on the same record count.



Raw Data in weak_scaling_data_recommend.csv



Raw Data in weak_scaling_data_sklearn_recommend.csv



Both systems scale steadily as the dataset size increases, excluding the last two record sizes which see an unexpectedly small execution time relative to the other smaller trials. This could be due to the size of the subset of the data being very close to or identical to the full dataset, which reduces overhead somewhere in the part of the code that subsets the data.

Limitations and Future Improvements

The most notable limitation of this system is inability to provide a diverse list of courses for the student. Outputs from both the Non-SKLearn and SKLearn versions of this system typically fall into to two categories. If the inputted student has a lower GPA, both systems output a list of courses that are among the easiest at Indiana University, which tend to only be within the same few departments. Alternatively, if the student has a higher GPA, both systems tend to output courses only within the student's major code, with a course level equal to that of the student's year in school. Both of these scenarios could help a student build part of their schedule, but will never allow them to find all of the courses they would need. To remedy this, additional parameters could be added to allow the user to select specific department codes and quantities of courses to return as an output. Both systems would still rank courses as they do now but would search down the rankings until they had found all of the requested department codes and quantities.

Another improvement that could be made to these programs would be to preprocess the data in advance and then store it for later use. This would be fairly straightforward in the Non-SKLearn system but the SKLearn system would need to be substantially reworked to enable this change as it needs the major of the student to do most of its preprocessing.