ENGR 212 – PROGRAMMING PRACTICE

SPRING 2015

MINI PROJECT 1

March 6, 2015

Student life is full of challenges and difficulties like deciding what movie to watch each night or deciding at what campus you should have your lunch so that you would minimize your walking time to the afternoon classes. Perhaps, the hardest to deal with challenge is to decide on what classes you would take during the registration time at the beginning of each semester. Most students would like to choose classes that they will like; hence, be successful (Assumption: A student is more likely to be successful in a class that s/he likes than the one s/he does not like). In this mini project, you are going to help students with this challenge.

What? : You are going to write a Python program that will act as a virtual advisor for students, and help them choose classes that they will most likely pass with higher grades.

How? : Using the collaborative filtering techniques (?) that you have learnt so far!

Input? : Past grades of a set of students (in a separate excel file per student), and the current transcript (in another excel file) of the student whom you will helping with the course selection.

Output? : The virtual assistant will suggest six courses that the student would most likely be successful for the next semester.

Can you provide a little bit more details? :

1. Your program will have a graphical user interface (GUI) which will look like as shown in Figure 1. You may use any coloring-scheme you like. The below one is for illustration purposes only. Details about how it should work are provided below.

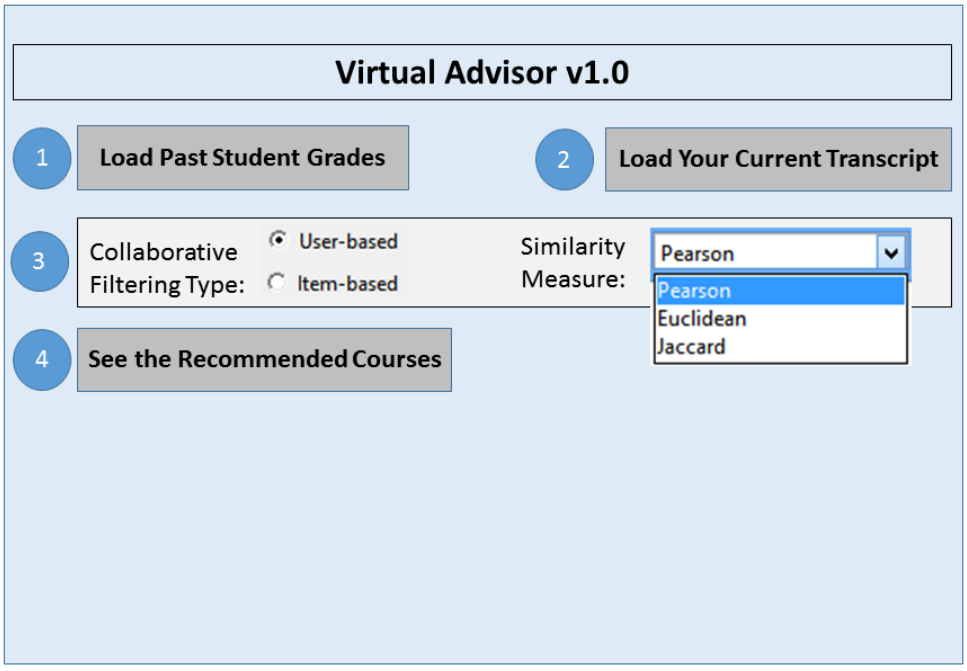

Figure 1

- First, the user will click on the "Load Past Student Grades", and a file dialog window will open (see Figure 2) to allow the user choose a set of excel files each of which contains past grades of a student. The Excel file content should be assumed to be formatted as in the

example Excel documents uploaded on LMS. We will test your programs with similar or the same files.

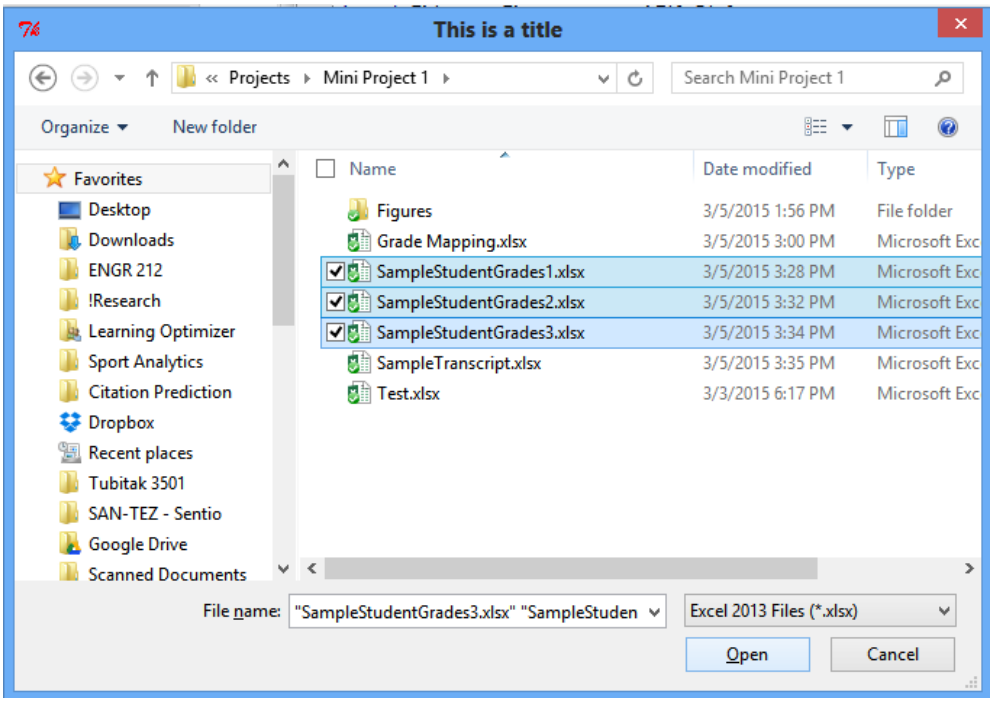

Figure 2

- Then, the user should click on the second button, "Load Your Current Transcript", which will open a file dialog to allow the user to choose an excel file that contains his/her grades from courses s/he has already taken. The format of this Excel file will be the same as the above ones in the previous step.
- Next, the user will choose the collaborative filtering approach and the similarity measure to be employed. Default values are "User-based" and "Pearson", respectively.

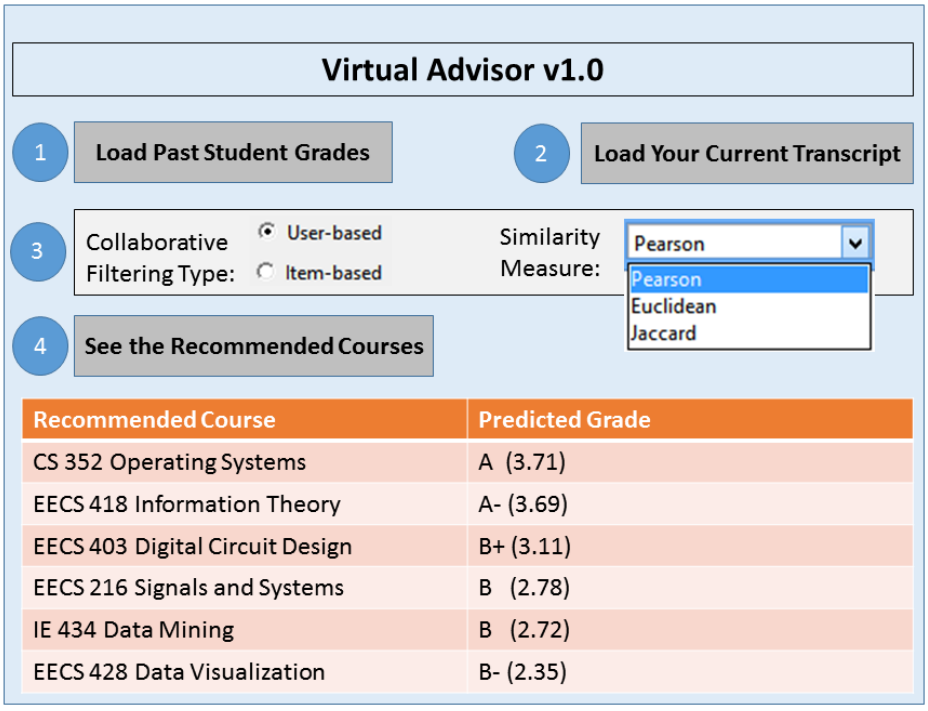

Figure 3

- Finally, the user will click on the last button, 'See the recommended courses', which will display the recommended top 6 courses with their predicted grades (sorted by the predicted grades) as shown in Figure 3. Each letter grade should be accompanied by the predicted actual grade (number) in parenthesis (e.g., for CS 352, predicted grade is A (3.71)). The sample output in Figure 3 was ***not*** produced from an actual program run; hence, may not be accurate. The program should not recommend any course that the student has already taken.

2. Note that in all grade files and in your program's output (Figure 3), letter grades are used. However, you will need to know the corresponding number values for each grade in order to be able to construct a collaborative filtering system. Please use the following mapping between letter grades and their corresponding number values. You will use the grades a person got on a course as a kind of "rating", and predict what grade (i.e., "rating") s/he will have on other classes, and display the top 6 of them (after mapping their predicted grades to letter grades by using the below table).

Letter Grade	*Number Value for Input Mapping*	*Range for Ouput Mapping*
A+	4.1	> 4
A	4	4 ≥ predicted grade > 3.7
A-	3.7	3.7 ≥ predicted grade > 3.3
B+	3.3	3.3 ≥ predicted grade > 3.0
B	3	3.0 ≥ predicted grade > 2.7
B-	2.7	2.7 ≥ predicted grade > 2.3
C+	2.3	2.3 ≥ predicted grade > 2.0
C	2	2.0 ≥ predicted grade > 1.7
C-	1.7	1.7 ≥ predicted grade > 1.3
D+	1.3	1.3 ≥ predicted grade > 1.0
D	1	1.0 ≥ predicted grade > 0.5
D-	0.5	0.5 ≥ predicted grade > 0.1
F	0	0.1 ≥ predicted grade

3. If the user clicks on the third button "See the recommended courses" before completing step 1 and/or 2, your program should display an error message that input files need to be provided first.
4. After getting recommended courses, the user should be able to change settings in step 3, and then click on the "See the recommended courses" button again to refresh the recommended courses based on new selections in Step 3 using the previously loaded grade files. This will allow comparing different collaborative filtering approaches and similarity measures.
5. Write your code in a single Python file in which you are going to import and use functions from the recommendations module (which is provided as part of the project in recommendations.py).

Can you provide any further pointers that may be helpful? :

- You may use **xlrd** module to read Excel files (install it on PyCharm in the same way you did the other modules). The following tutorial have example usages of this module. The code piece on page 9 of this tutorial should be sufficient for you.

  http://www.simplistix.co.uk/presentations/python-excel.pdf

- As for the GUI, you may use Tkinter that we have covered last semester (see ENGR 211 last week's slides). Our coverage did not include using a file dialog (tkFileDialog), but it is really simple, and the following link contains many useful examples of using tkFileDialog.

  http://tkinter.unpythonic.net/wiki/tkFileDialog

askopenfilenames() is the method that you will use to get multiple file names selected by the user. Its usage is very similar to askopenfilename() for which there is an example at the above link. The only difference is that it returns multiple file names.

- For the collaborative filtering type choice, you may use the RadioButton widget of Tkinter. The following example code pieces may help:

  http://effbot.org/tkinterbook/radiobutton.htm
  http://www.python-course.eu/tkinter_radiobuttons.php

- For similarity measure setting, you may use the ComboBox widget of Tkinter. The following example code piece may help:

  http://stackoverflow.com/questions/17757451/simple-ttk-combobox-demo

- To create a table for your output, you may use the treeview widget in Tkinter. The following demo and sample code may be helpful:

  https://www.daniweb.com/software-development/python/threads/350266/creating-table-in-python

How and when do I submit my project? :

- Projects may be done individually or as a small group of two students. If you are doing it as a group, only **one** of the members should submit the project. File name will tell us group members (Please see the next item for details).
- Submit your own code in a **single** Python file (Do **not** include recommendations.py that you import). Name it with your and your partner's first and last names (see below for naming).
  - If your team members are Deniz Barış and Ahmet Çalışkan, then name your code file as deniz_baris_ahmet_caliskan.py (Do **not** use any Turkish characters in file name).
  - If you are doing the project alone, then name it with your name and last name similar to the above naming scheme.
- Submit it online on LMS (Go to the Assignments Tab) by **17:00 on March 20**.

  Late Submission Policy:

  - -20%: Submissions between 17:01 – midnight (00:00) on the due date
  - -30%: Submissions which are 24 hour late.
  - -50%: Submissions which are 48 hours late.
  - Submission more than 48 hours late will not be accepted.

Grading Criteria? :

- Does it run? (Submissions that do not run will get some partial credit which will not exceed 30% of the overall project grade).
- Does it implement all the features according to the specifications and produce correct results?
- Code organization (Meaningful names, sufficient and appropriate comments, proper organization into functions and classes, clean and understandable, etc.)?
- Interview evaluation.

Have further questions? :

- Please contact your TAs (Gihad Sohsah or Mehmet Aytimur) if you have further questions. If you need help with anything, please use the office hours of your TAs and the instructor to get help.