How to Grade a Class that Has Both Theoretical and Practical Components?

Vignesh Ponraj and Vladik Kreinovich
Department of Computer Science
University of Texas at El Paso
500 W. University
El Paso, TX 79968, USA
vigneshponraj97@gmail.com, vladik@utep.edu

1. How classes are usually graded: a reminder

- What do we want from students?
- For example, for CS1:
  - we want them to do the practical stuff, to be able to code, and
  - we want them to understand the theoretical part: how to trace the code, what for-loop does, and so on.
- The overall grade comes as a combination of these parts.
- How do you compute the overall grade?
- You have a grade for the tests on which you basically test the student's ability to understand the code and to write down simple code.
- There is a grade for programming assignments, which test the ability of students to code some reasonable problems.
- Usually, the overall grade for the class is a weighted average of the practical and theoretical grades.

2. Why is this a problem: a simple example

- Suppose that a student:
  - has a 100 on the theoretical part on all the tests, and
  - on all programming assignments, only gets 60, below satisfactory.
- If we consider these grades equally, this gives us 80.
- So, not only the student passes the class, he/she is considered a good student.
- Suppose now that upon graduation, this student is hired by a company.
- The company sees that on CS1, that student got a B.
- This means that the student is good, so he is given programming tasks.
- Unfortunately, on these tasks, he fails miserably.

3. Why is this a problem: a simple example (cont-d)

- So, the company says: the university is inflating the grades.
- And they never hire our graduates again.

4. A possible solution and its limitations

- A solution that some instructors use is to take the minimum of the two grades.
- So, if a student scored 100 for one part and 60 for another part, this student gets 60 and fails the class.
- A company looks at it and thinks that this student is useless in CS.
- However, in reality, for some tasks, the student can be useful.
- This student cannot write code, but he/she can understand others' codes and thus help.

5. A recent proposal

- To resolve this situation, Dr. Nigel Ward from UTEP's Computer Science Department proposed the following empirical formula:

$$g = \min(\alpha \cdot g_p + (1-\alpha) \cdot g_t, \alpha \cdot g_t + (1-\alpha) \cdot g_p).$$

- Here, $\alpha > 0.5$ is a constant. Dr. Ward uses this formula in his classes, and it seems to work.
- But why this particular formula?
- What are the general kind of formulas that we can use?

6. Let us use decision theory

- We are talking about making a decision – in this case, a decision on what grade to give to a student.
- So, a natural idea is to use decision theory to help.
- In decision theory, decision maker's preferences are described by a value called *utility*.
- This numerical value is defined in such a way that:
  - if an action leads to different outcomes with different probabilities,
  - then the utility of the action is equal to the expected value of the utility

$$u = p_1 \cdot u_1 + \ldots + p_n \cdot u_n.$$

7. Let us apply decision theory to our problem

- We want to estimate the student's utility to a company.
- It is reasonable to divide the possible tasks that the student can do for a company into three groups:
  - the company may need this hire to code something,
  - the company may need this hire to analyze someone else's code without writing a new code, and
  - the company may want the new hire to do both.
- For simplicity, let us assume that the utility is simply proportional to the grade.
- Then, in the first task, the student's utility is proportional to this student's *p*ractical grade.
- In the second task, the student's utility is proportional to this student's *t*heoretical grade.
- In the third task, *b*oth skills are required.

8. Let us apply decision theory to our problem (cont-d)

- So, the student's utility is proportional to the minimum of the two grades.
- Let us denote by p_p, p_t, and p_b the probability of each type of task.
- Then, we get the following formula for the student's utility

$$u = p_p \cdot g_p + p_t \cdot g_t + p_b \cdot \min(g_p, g_y).$$

- Interestingly, when we take both parts equally $p_p = p_t$, we get exactly Dr. Ward's formula with $p_p = p_t = 1 - \alpha$ and $p_b = 1 - p_p - p_t$.
- So:
  - not only we explained the empirically successful formula,
  - we also have a more general formula that we can use if we want to give more weight to one of the parts.

9. Interestingly, the same formula appears if we use linear interpolation

- Another idea is:
  - start with the cases when the answer is known, and
  - use linear interpolation – the simplest interpolation technique – to extend this answer to all other combinations of the grades.
- We need to transform the grades a and b for two parts into a single grade $f(a, b)$.
- What do we know about $f(a, b)$?
- If the students gets exactly the same grade a for both components, then this should be this student's overall grade: $f(a, a) = a$.

10. Linear interpolation (cont-d)

- We also need to know the student's grade in two extreme situations:
  - when the student in perfect in coding but fails theory, and
  - when the student is perfect in theory but fails programming.
- Let us denote the corresponding overall grades by $v_p = f(1, 0)$ and $v_t = f(0, 1)$.
- Let us show that based on this data, linear interpolation indeed leads to the same formula as the utility approach.

11. Cases when $a > b$ and $a < b$

- Let us consider two possible cases: when $a > b$ and when $a < b$.
- When $a > b$, we first apply linear interpolation by the first variable to the known values $f(0,0) = 0$ and $f(1,0) = v_p$.
- This way, we get $f(a,0) = a \cdot v_p$.
- Then, we apply linear interpolation by the second variable to the values $f(a,0) = a \cdot v_p$ and $f(a,a) = a$.
- This way, we get $f(a,b) = v_p \cdot a + (1 - v_p) \cdot b$.
- When $a < b$, we similarly get $f(a,b) = v_t \cdot b + (1 - v_t) \cdot a$.

12. The result is the same as for the utility formula

- Let us show that these two formulas can both be described in the above form $p_1 \cdot a + p_2 \cdot b + p_3 \cdot \min(a, b)$.
- Indeed, when $a > b$, this utility formula takes the form

$$p_p \cdot a + (p_t + p_b) \cdot b.$$

- So, to match with the interpolation formula $v_p \cdot a + (1 - v_p) \cdot b$, we need to take $p_p = v_p$.
- When $a < b$, the utility formula takes the form $(p_p + p_b) \cdot a + p_t \cdot b$.
- So, to match with the interpolation formula $(1 - v_t) \cdot a + v_t \cdot b$, we need to take $p_t = v_t$.
- So, indeed, we get the same formula as in the utility approach.

13. Important comment

- It should be mentioned that not all such cases can be interpreted in utility terms.
- Indeed, when $v_p + v_t > 1$, we get negative p_b, but the probability cannot be negative.

14. What is we need to combine three or more grades

- This was all about the case when we only have two components to combine.
- What if we have three or more components?
- In this case, you may have different utilities depending on which skills you have.
- For example, for the 3 components, you have the following formula:

$$u = p_1 \cdot g_1 + p_2 \cdot g_2 + p_3 \cdot g_3 +$$

$$p_{12} \cdot \min(g_1, g_2) + p_{13} \cdot \min(p_1, p_3) + p_{23} \cdot \min(p_2, p_3) +$$

$$p_{123} \cdot \min(p_1, p_2, p_3).$$

15. Acknowledgments

This work was supported by:

- the AT&T Fellowship in Information Technology,
- the Institute for Risk and Reliability, Leibniz Universitaet Hannover, Germany,
- the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) Focus Program SPP 100+ 2388, Grant Nr. 501624329,
- the European Union under the project ROBOPROX (No. CZ.02.01.01/00/22 008/0004590),
- the Center of Excellence in Econometrics, Faculty of Economics, Chiang Mai University, Thailand,
- the Ho Chi Minh City University of Banking, Vietnam, and
- Thang Long University, Hanoi, Vietnam.