Software Model of Student Performance Evaluation

Asst.D Eman Hato
Computer Science Department, College of Science, Al-Mustansiriyah University
Email: emanhato@uomustansiriyah.edu.iq

Abstract

Teachers and students should take advantage of computers technology and educational programs which provide efficient analysis of student information. In this paper, a model for student evaluation is presented to assist the teachers in their decision making process of student evaluation. The model consists of three stages, the first stage is reading the excel file due to the excel file is the most widely utilized by scientific departments that contains the student's grades. Also, the exams weights are reading in this stage to determine the weight of each exam. In the second stage, the student's averages are computed depending on the exams number and their weights. The last stage of model provides options which help teachers to adjust the average grades. The proposed model accurately calculates the student's performance with less time and effort than manual assessment. It's giving the teachers the ability to choose adjustment methods according to his requirements. As well as, the proposed model interfaces are simple, flexible and easy to use.

Keywords: Educational programs, exams weights, adjustment methods, evaluation model.

الملخـص

يجب أن يستفيد التدريسيون والطلاب من تكنولوجيا الكمبيوتر والبرامج التعليمية التي توفرتحليلاً فعالاً لمعلومات الطلاب. في هذا البحث ، يتم تقديم نظام لتقييم الطلاب وذلك لمساعدة التدريسيين في عملية أحتساب معدل الطالب (درجة السعي) واتخاذ القرار بشأن تقييم الطلاب. يتكون النموذج المقترح من ثلاث مراحل ، المرحلة الأولى هي قراءة ملف الاكسل لأن ملف اكسل هو الأكثر استخداما من قبل الأقسام العلمية التي تحتوي على درجات الطالب. كما يتم قراءة أوزان الامتحانات في هذه المرحلة وذلك لتحديد وزن كل اختبار. في المرحلة الثانية ، يتم احتساب معدل الطالب على أساس عدد الامتحانات وأوزانها. توفر المرحلة الأخيرة من النموذج خيارات تساعد التدريسيين على تعديل معدل الطالب ان تطلب الامر. يحسب النظام المقترح أداء الطالب والمتمثل بحساب درجة السعي بدقة وبأقل وقت وجهد من التقييم اليدوي. إنه يمنح التدريسيين القدرة على إختيار طرق التعديل وفقًا لمتطلباتهم. بالإضافة إلى ذلك ، فإن واجهات النظام المقترح بسيطة ومرنة وسهلة الإستخدام.

الكلمات المفتاحية: البرامج التعليمية، أوزان الإمتحانات، طرق التعديل، نظام التقييم.

1. Introduction

The knowledge nowadays consider a critical factor in global economy, education plays an important role as a knowledge center and developer of human resources. The main objective of educational colleges and one of the biggest challenges they face is to provide quality education to their students. One way to achieve a higher level of quality is to predict the performance of students and then taking early actions by providing appropriate support to students with learning difficulties to improve their performance [1].

Use of computers technology in education has completely changed the way things were made before. For example, lectures to the students or the presentation making a talk very attractive. In addition, various projects and educational programs are prepared with the assistance of expert educators to assess the achievement of student learning outcomes [2].

Teachers and students should take advantage of these educational programs which provide efficient processing and analysis of student information. The need for changing traditional methods of student evaluation to proactive programs is essential and important to educators and decision makers [3, 4].

The aim of this paper is to design student evaluation-assisted model for accurately predicting the performance of student at the end of the course. The model provides options and facilities those help in decision-making to support the evaluation process and obtain satisfactory results.

2. Computer-Assisted Models Motivation

Modern computer technologies have been widely applied in all areas of life, allowing to perform many tasks that were previously very time consuming while providing tools that has expanded intellectual powers possessed by human [5].

Computer assisted programs and models are general terms used to describe collection of computer programs, documentation and procedures that perform tasks on a computer system. The computer assisted programs perform tasks faster and more accurately [6].

During the last decade, the computer programs application on educational data to develop effective and accurate decision support systems for monitoring students' performance has become very popular. The use of computers tools can bring about better results than the traditional methods due to the speed of achievement, performance and accuracy in work [7].

Therefore, developing an academic computer model is important for teachers, students, and educational organizations and it will be more valuable for educational administrators in their decision making process of student evaluation.

3. Educational System

Education is essential not only for the progress of individuals, but also for the development of society and nation. Assessment is important parts of the educational process. The assessments focus on students acquiring knowledge, as well as decide whether the learning of a student is proper or not [8].

Assessment of student performance in colleges is a difficult task for teachers. Evaluation consists of a judgment based on the comparison of student results against established performance criteria. The performance evaluation is usually expressed manually, based on many factors such as student attendance, homework assignments, class assignments, examinations, tests and exercises [9].

New technologies offer great possibilities for teachers to use computer hardware and software in the learning process. For successful use of computer technologies in education, certain key conditions must be fulfilled [10, 11]:

- The software must be able to carry out the tasks required in an educational.
- The right tool must be available when needed and must be reliable.
- It is essential that it be user friendly.

4. Equation of Straight Line

A straight line is an endless one dimensional shape that has no width and no curves on it. The straight line is a shape formed when two points A(x1,y1) and B(x2,y2) are connected by the shortest distance between them and the ends of line extend to infinity. In general, straight lines can be classified based on their alignment which is defined as the angle they form with the X-axis or the Y-axis. According to the alignment of straight lines, they are can be horizontal lines, vertical lines or slanted lines. A linear equation of a straight line given as [12]:

$$y = m * x + b \quad (1)$$

Where:

x: how far along.

y: how far up.

b: value of y when x=0.

m: gradient or slope (how steep the line is).

The straight line slope determines the straight line direction and tells how steep the line is. The slope is calculated as the difference in y coordinates divided by difference in x coordinates. It is given by [12]:

$$m = (x_1 - x_2)/(y_1 - y_2) \quad (2)$$

5. The Proposed Model of Student Evaluation

The proposed model composed of three main stages that are performed different processes. The first stage is reading the excel file that contains the student's grades. Also, the exams weights are reading in this stage to determine the weight of each exam. The student's averages are computed in the second stage depending on the exams number and their weights. The last stage of model provides options which help teachers to adjust the average grades such as weights adjustment and adding manual or auto curve. In following subsections, these stages are explained in more details.

5.1 Excel File Reading

The excel file is employed by widely scientific departments and college as a list of students' names and attendance. Therefore, the proposed model use the excel file as an input file in order to reading students names and their grades.

The reading process starts from the rows with real data (student's names and exams grades) to read the student's name and grades and discard the rows that continents the useless information such as the college or department logo. Each column contains numerical value represented exam or activity grade of student, i.e. the number of columns represents the number of exams. Then, the weights for each exam are read and are stored in array that is used later to calculate student's average.

5.2 Student Average Computing

The average grades are calculated in this stage. Each exam grade is multiplied by the corresponding weight to obtain the average grade of the student. It should be noted, a question was asked to the teacher if he wants to save the average grades without applying any modification or not. In the case of acceptance, the average grades are saved to the excel file, otherwise, various adjustment methods are presented for selection according to the teachers requirements as explained in the following subsection.

5.3 Average Grades Adjustment

Before declaring the final average grades of students, adjustment process is presented as an option for teacher to enhance the average grades, if teacher is not satisfied with it.

Three methods are presented; the first one is enabling the teacher to redefine the weight of the exams. For example, reduce the weight of an exam with low scores, or increase the weight of an exam with higher scores.

The other two methods include adding a curve manually or automatically. The manual method is based on a classic style, where equal scores are added to all students determined by the teacher.

The automatica method is utilized the equation of a straight line. If the avarage is considered as the equation of a straight line detrmined by the maximum (oldmax) and minimum (oldmin) grades. Then the method simply resets the slope of the straight line equation by determining the new minimum (newmin) grade and the new maximum (newmax) grade as shown in Figure (1).

Note, the newmin grade represents the lowest possible grade in the average, therefore, determined by the teacher. While the newmax no change so that the maximum value of the average is not exceeded.

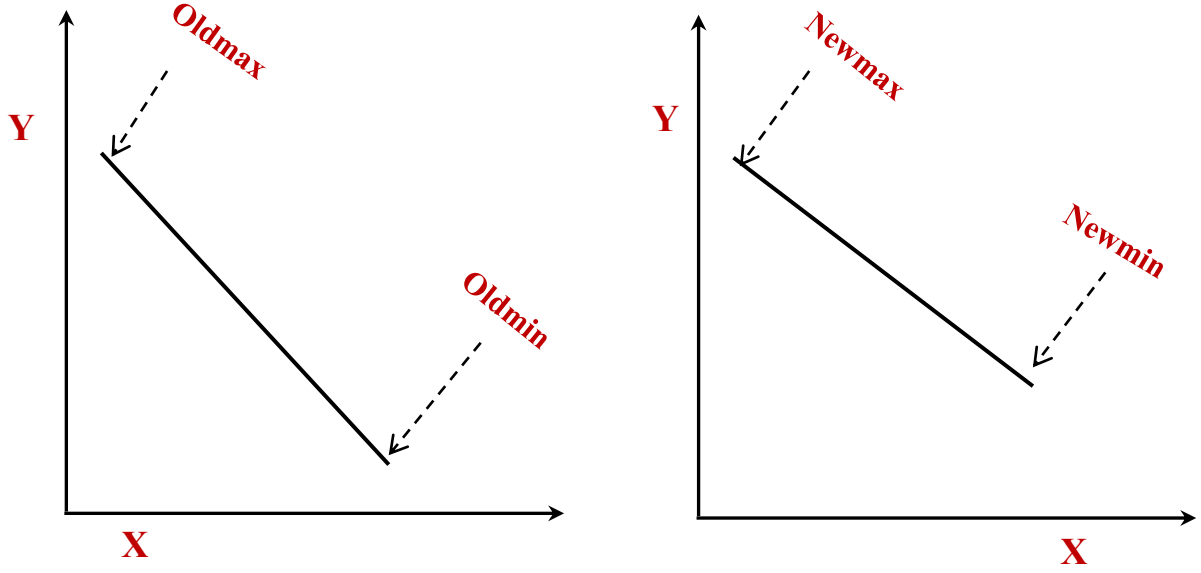


Figure 1: Changing of line slope for determining new minimum grade.

By using equation (1) the values of newmin and newmax will be as follows:

$$Newmax = m * Oldmax + b \quad (3)$$

$$Newmin = m * Oldmin + b \quad (4)$$

The equations are solved simultaneously by subtracting equation (4) from (3) to find the value of *m* and *b*.

The result is:

$$Newmax - Newmin = m\ (Oldmax - Oldmin) \quad (5)$$

The m value is:

$$m = (Newmax - Newmin) / (Oldmax - Oldmin) \quad (6)$$

To compute *b* value, the *m* value substitute into equation (3) or (4):

$$b = Newmax - m * Oldmax \quad (7)$$

By using the new value of *m* and *b*, the new avarege values can be computed as:

$$NewAvr = m * OldAvr + b \quad (8)$$

Whare the *NewAvr* is the new avarage grade and *OldAvr* the old avarage grade.

This method helps to add the curve unevenly where the addition to lower grades is slightly more than higher grades.

The general structure of the proposed model of student evaluation is illustrated in algorithm 1.

Algorithm 1: The Proposed Model of Student Evaluation.

Input: Excel file.
Output: Excel file with students' average grades.

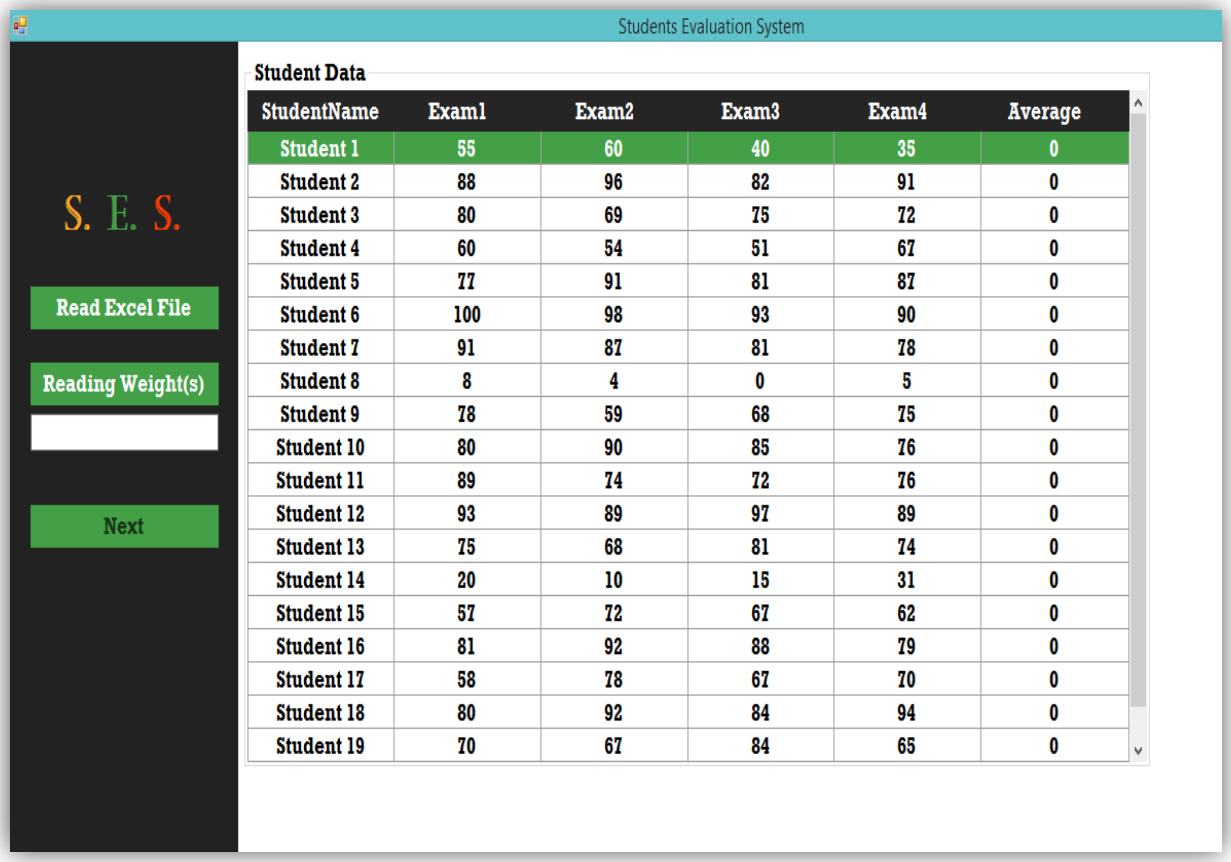
Begin
Step1: Determine actual rows and columns used in excel file.
RowCount ← Range-cell ≠ null in excel file.
ColCount ← Range-cell value ≠ null of excel file.
Step2: Reading the student grades.
For (i ← 1 to RowCount)
Student ←new Student ().
Student. Name ← value of cell [i, 1].
For (j ← 2 to ColCount)
Student.Exam [j-1] ← value of cell [i, j].
End for
SList ← Student record. // Add student record to student list.
End for
Step3: Read the weights values.
NoExam ← ColCount-1
Wexam ← Double [NoExam]. // Array to hold exams weights.
For (i ← 1 to NoExam)
Wexam [i] ← Read the value of weight.
End for
Step4: Make sure the total weights are not higher than 100.
If (Sum (Wexam) ≠ 100)
Print error message.
Return to **Step3.**
End if
Step5: Compute student average using the exams weights values.
Average ← double [RowCount]. // Array to average grades.
For (i ← 1 to RowCount)
For (j ← 1 to NoExam)
Average [i] ← Average [i]+Slist.Exam[i,j] * Wexam[j].
End for
Average [i] ← Average [i] / Sum (Wexam).
End for

Step6: Save average grades or select the adjustment methods.
Exit ← false.
Repeat
Switch (Button)
Case Button1.Click **=** true**:** // Save average grades**.**
Store average grades in a worksheet of excel file.
Exit ← true.
Break.
Case Button2.Click = true**:** // Redefine exams weights.
Go to **Step3**.
Break.
Case Button3.Click = true**:** // Add a curve manually.
Mcurve ← Read manual curve value of average.
For (i ← 1 to RowCount)
NewAverage[i] = Average[i] + Mcurve.
If (NewAverage[i]>MaxAverage)
NewAverage[i]=MaxAverage.
End if
End for
Break.
Case Button4.Click = true**:** // Add a curve automatically.
Newmin ← Read new minimum value of average that required by teacher.
Compute **m, b** as explained in section **4.3**
For (i←1 to RowCount)
NewAverage[i] = m * Average[i] + b.
End for
Break.
End switch
Until (Exit ← true).
End.

6. Implementation and Discussion

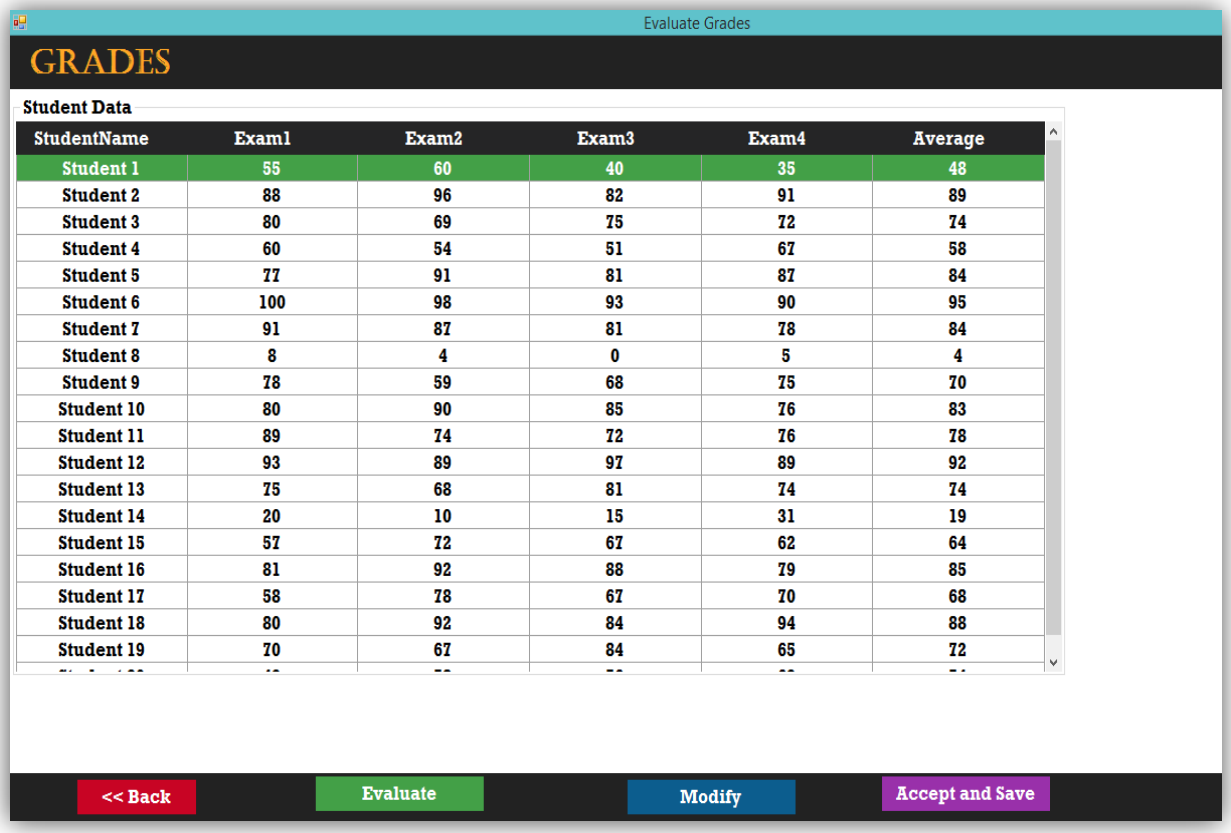
The model was designed using C# Language that is a powerful language primarily aimed at developers creating applications using Microsoft .NET Framework.

An excel file was used as a test material that simulates what is used in the Department of Computer Science - College of Science - Al-Mustansiriyah University. The Figure (2) shows the first interface that is used to read input excel file and the weights of exams. While the Figure (3) shows the second interface after students average grades are computed, where the weight **25** was adopted for each exam.



StudentName	Exam1	Exam2	Exam3	Exam4	Average
Student 1	55	60	40	35	0
Student 2	88	96	82	91	0
Student 3	80	69	75	72	0
Student 4	60	54	51	67	0
Student 5	77	91	81	87	0
Student 6	100	98	93	90	0
Student 7	91	87	81	78	0
Student 8	8	4	0	5	0
Student 9	78	59	68	75	0
Student 10	80	90	85	76	0
Student 11	89	74	72	76	0
Student 12	93	89	97	89	0
Student 13	75	68	81	74	0
Student 14	20	10	15	31	0
Student 15	57	72	67	62	0
Student 16	81	92	88	79	0
Student 17	58	78	67	70	0
Student 18	80	92	84	94	0
Student 19	70	67	84	65	0

Figure 2: The first interface of the proposed software model.



StudentName	Exam1	Exam2	Exam3	Exam4	Average
Student 1	55	60	40	35	48
Student 2	88	96	82	91	89
Student 3	80	69	75	72	74
Student 4	60	54	51	67	58
Student 5	77	91	81	87	84
Student 6	100	98	93	90	95
Student 7	91	87	81	78	84
Student 8	8	4	0	5	4
Student 9	78	59	68	75	70
Student 10	80	90	85	76	83
Student 11	89	74	72	76	78
Student 12	93	89	97	89	92
Student 13	75	68	81	74	74
Student 14	20	10	15	31	19
Student 15	57	72	67	62	64
Student 16	81	92	88	79	85
Student 17	58	78	67	70	68
Student 18	80	92	84	94	88
Student 19	70	67	84	65	72

Figure 3: The interface of student's average grades.

As shown in Figure (3), the teacher can save the average grades without applying any modification or he can select one of the various adjustment methods after click on the **Modify** button.

The third interface was presented with three modification method. In the Figure (4) the reading weight was selected to redefine the weights of exams. The input weights were **35, 15, 30** and **20** for exam1, exam2, exam3 and exam4 respectively. On the basis of these weighs, the average was calculated again.

The other method of modification is the manual method. The implementation step for adding manual curve is illustrated in Figure(5), where the curve value **4** is added to the average grades that are shown in Figure (3).

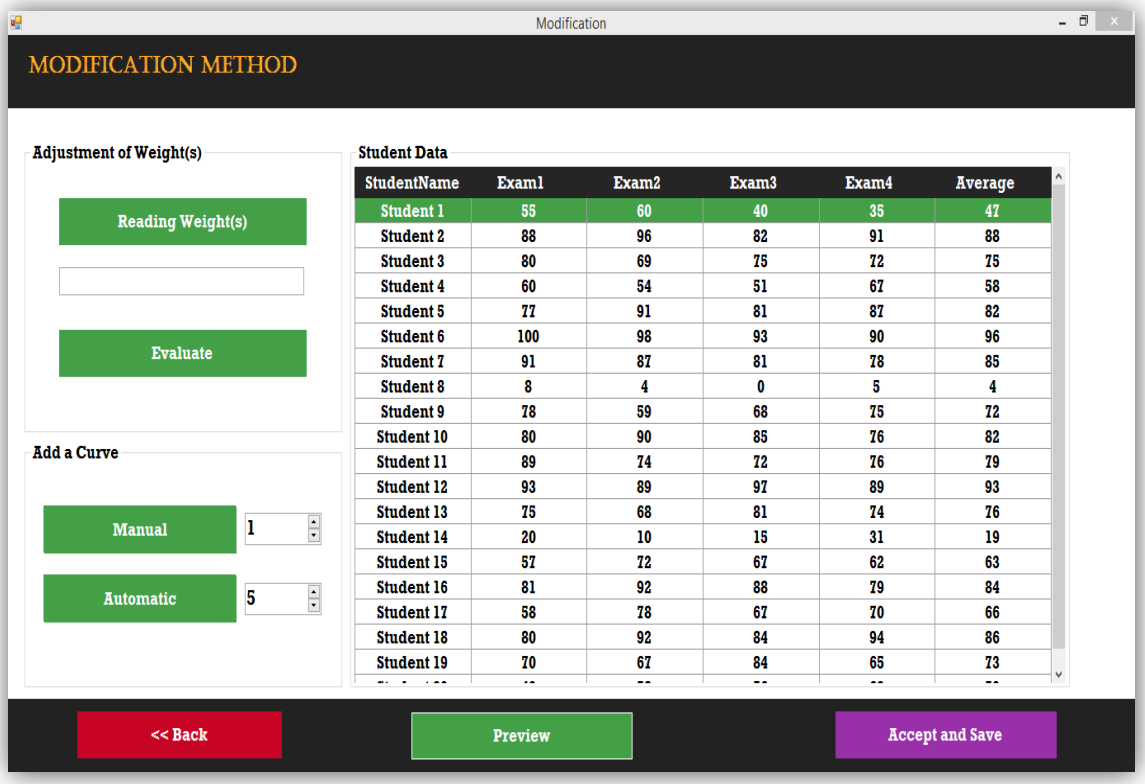


Figure 4: Re-define exams weights to modify average grades.

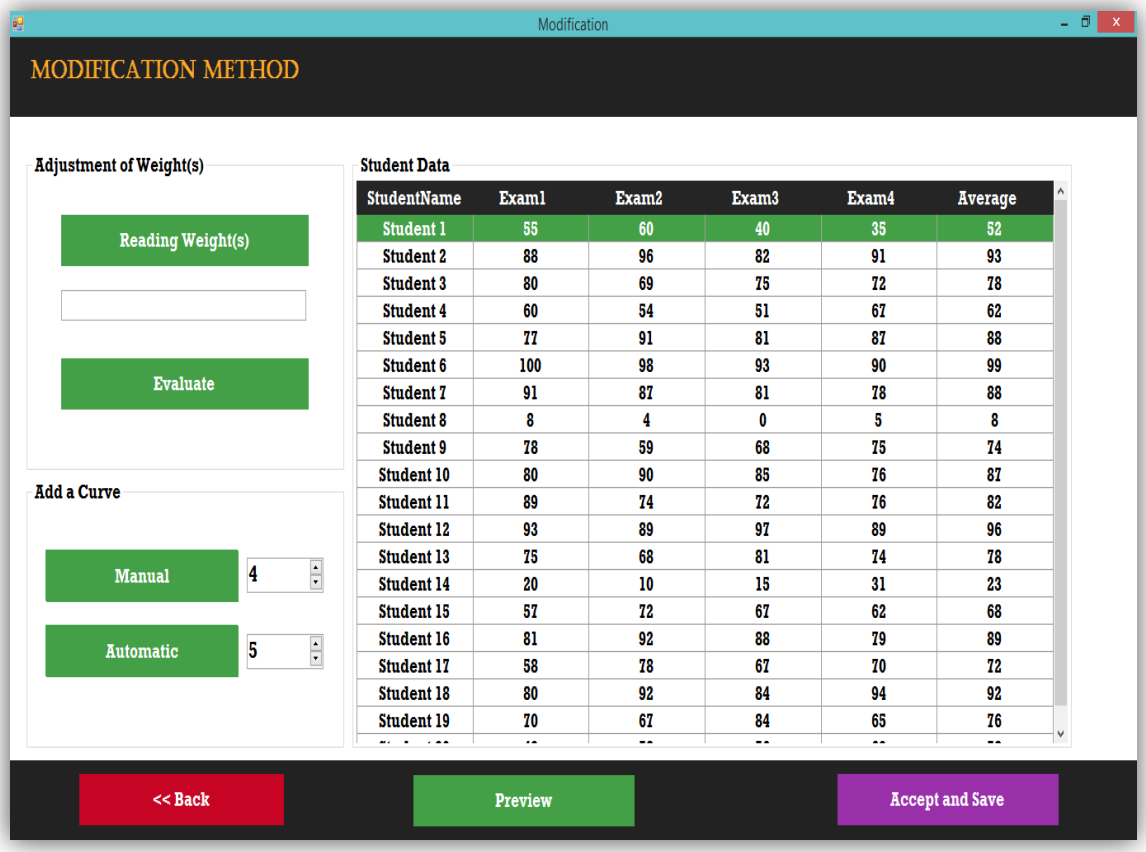


Figure 5: Adding manual curve to modify average grades.

It is worth noting that each addition is based on the value of the original average grades, i.e. before any addition is made. This enables the teacher to compare the average grades after and before the modification, making it easier for him to choose the appropriate average grades for students.

The average grades can be modified by adding automatic curve as illustrated in Figure (6), where the minimum value **20** is used. The curve was computed and added to the average grades that are shown in Figure (3).

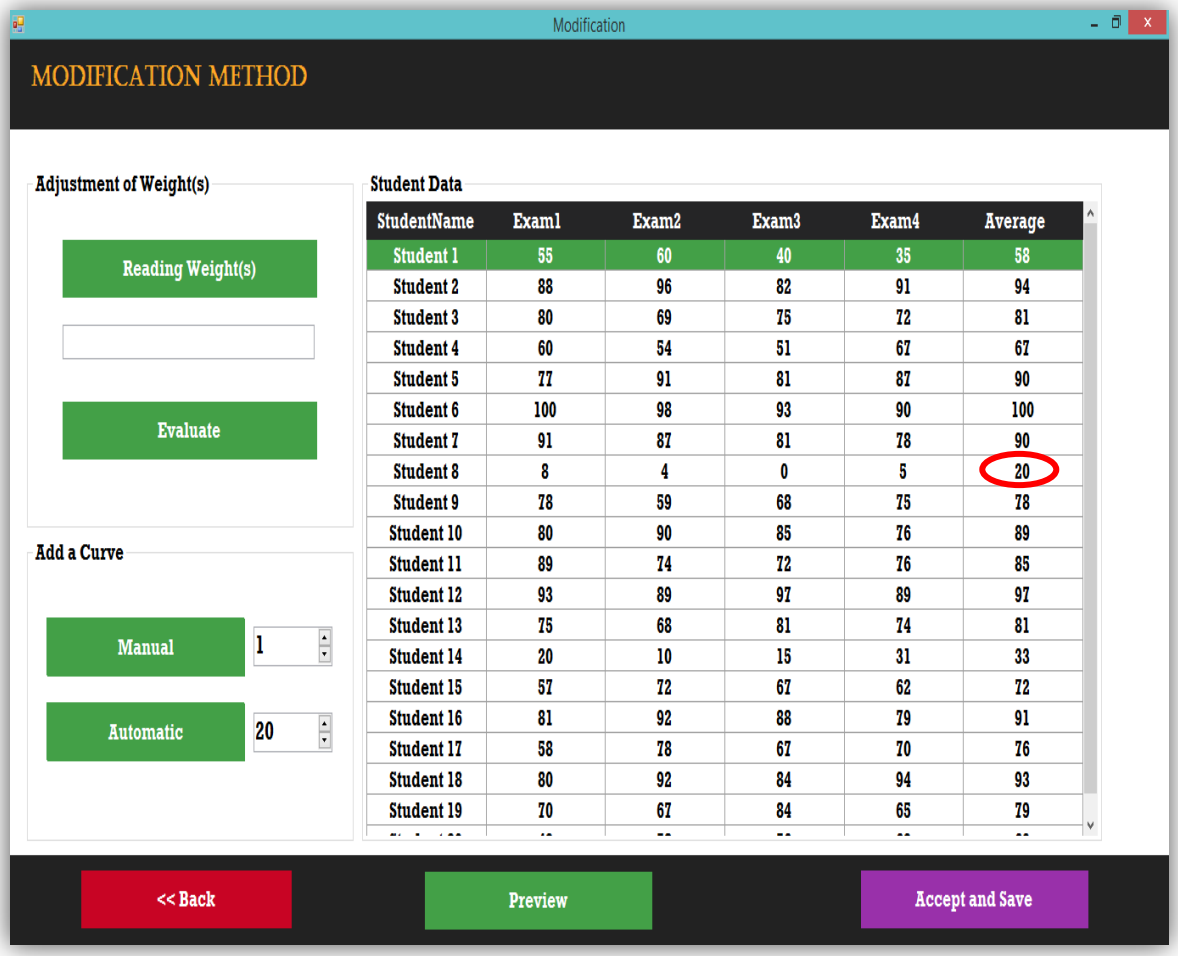


Figure 6: Adding automatic curve to modify average grades.

It is clear that the addition of this curve is not equal. Where a little more values were added to low average grads compared to the high average grads. For example, the average grad of Student 6 is 96 and after adding automatic curve it became 100, that is, the value of curve is 4 only. On the other hand, the average grad of Student 14 is 19 and after adding automatic curve it became 33, i.e. the value of curve is 14. Also, it can be seen that the lowest average grad is 20, depending on the minimum value which was can be determined by the teacher (see a red circle in Figure (6)). This method helps increase low grades while maintaining high grades within an acceptable range.

After the teacher conviction of the average grades, the students grades can be saved in an excel file when click the **Accept and Save** button.

Rresponse time is an important aspect for efficient systems. The execution time of the proposed model is compared to the time taken to manually assess students in the traditional way.

Figure (7) shows the details of a comparison of the execution time. It is clear that the execution time of the proposed model is very low compared to the traditional evaluation. Results of the execution time comparison showed the superior performance of the proposed model.

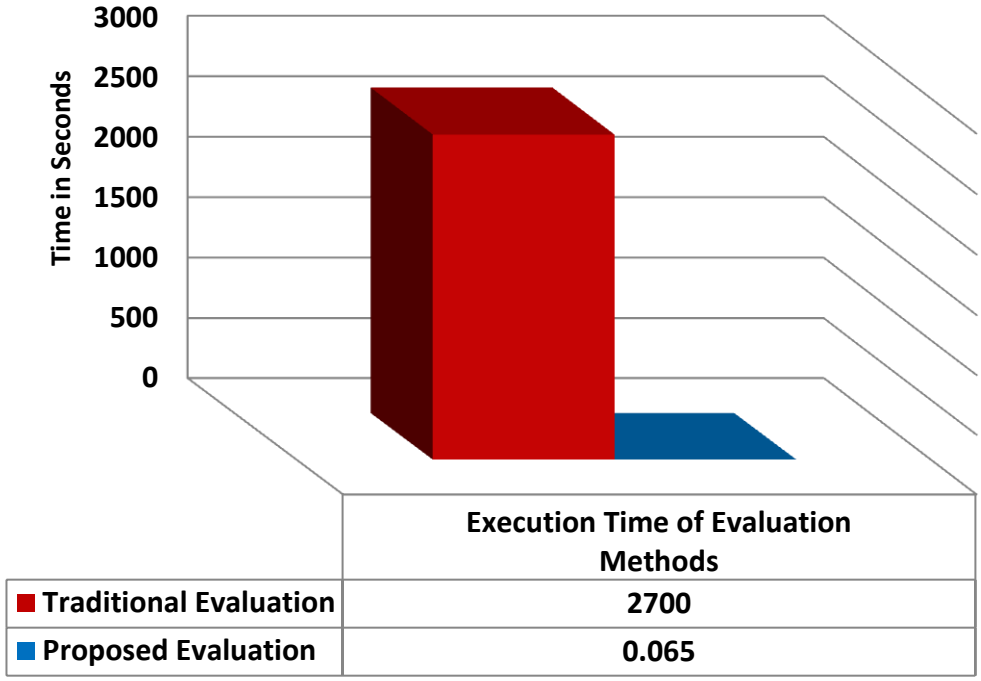


Figure 7: The execution time comparison.

7. Conclusions

A model for student evaluation was designed for predicting accurately the performance of students at the end of the course. The model consists of three stages, the first stage was to read the excel file that contains the student's grades, and exams weights. In the second stage, the student's averages were computed after determining the weight of each exam. The last stage of model provides multiple options and facilities to help in the evaluation process such as adding manual and auto curve or enabling teaches to redefine the weight of the exams.

The proposed model able to carry out the task of student evaluation required in an educational, which through its use will save teachers effort and time as well as accuracy in computing. Moreover, the

model interfaces are user-friendly, by providing easy and flexible use without any complexity, and this is a very important point of any educational application system. Finally, this model can help professors in our Iraqi universities in the process of evaluating students.

References

1. Livieris, I.E., Kotsilieris, T., Tampakas, V. and Pintelas, P., Improving the evaluation process of students' performance utilizing a decision support software. Neural Computing and Applications, 31(6), pp.1683-1694, **2019**.
2. Raja, R. and Nagasubramani, P.C., Impact of modern technology in education. Journal of Applied and Advanced Research, 3(1), pp.33-35, **2018**.
3. Lemoine, P.A., Waller, R.E., Garretson, C.J. and Richardson, M.D., Examining technology for teaching and learning. Journal of Education and Development, 4(2), pp.80-89, **2020**.
4. Salunkhe, S.S., Deshpande, A. and Joshi, Y., Degree of Certainty in Students' Academic Performance Evaluation Using a New Fuzzy Inference System. Journal of Intelligent Systems, 27(4), pp.537-554, **2018**.
5. Gilakjani, A.P., A detailed analysis over some important issues towards using computer technology into the EFL classrooms. Universal Journal of Educational Research, 2(2), pp.146-153, **2014**.

6. Galvin, P.B., Gagne, G. and Silberschatz, A., *Operating system concepts* (Vol. 10). John Wiley & Sons, **2011**.
7. Shahid, F., Aleem, M., Islam, M.A., Iqbal, M.A. and Yousaf, M.M., A review of technological tools in teaching and learning computer science. Eurasia Journal of Mathematics, Science and Technology Education, 15(11), pp.1-17, **2019**.
8. Kapur, R., Factors influencing the students' academic performance in secondary schools in India. University Of Delhi, 12(1), pp.1-25, **2018**.
9. Namli, N.A. and Şenkal, O., Using the fuzzy logic in assessing the programming performance of students. International Journal of Assessment Tools in Education, 5(4), pp.701-712, **2018**.
10. Akhter, N. and Fatima, Q., Computer Applications in Formative Assessment: A Gender-Based Comparison at Postgraduate Level. Bulletin of Education and Research, 39(2), pp.149-170, **2017**.
11. Thomas Jr, G**,** *Thomas Calculus, Early Transcendentals*. Cengage Learning, **2010**.
12. Samartkit, P. and Pullteap, S., A design of decision making-assisted software using fuzzy logic technique: a case study of solar cell investment project. Electrical Engineering, 101(1), pp.213-223, **2019**.