



UNIVERSITÀ DEGLI STUDI DI CAMERINO
School of Advanced Studies

DOCTORAL COURSE IN
COMPUTER SCIENCE AND MATHEMATICS

XXXIV cycle

MACHINE LEARNING APPLICATIONS TO
E-LEARNING COURSES

Ph.D. Student:
Giacomo Nalli

Supervisors:
Prof. Andrea Perali
Prof. Leonardo Mostarda

Contents

Introduction	3
Chapter 1: Basic Concepts and Technologies	12
1.1 Artificial Intelligence and Machine Learning	12
1.2 Moodle E-learning platform	15
1.3 Machine Learning and Education	18
Chapter 2: State of Art and Objectives.....	21
2.1 Research Activity 1: Moodle plugin for creation of student heterogeneous groups in online university courses.....	21
2.1.1 State of the Art	21
2.1.2 Objective	23
2.2 Research Activity 2: Tool for classifying students' sentiments in an e-learning course.....	24
2.2.1 State of the Art	24
2.2.2 Objective	25
2.3 Research Activity 3: Intelligent chatbot as a virtual assistant for university courses with a large number of students.....	26
2.3.1 State of the Art	26
2.3.2 Objective	27
2.4 Research Activity 4: Automatic tutoring system for programming course	28
2.4.1 State of the Art	28
2.4.2 Objective	32
Chapter 3. Methodological Aspects and Models	32
3.1 Research Activity 1: Moodle plugin for creation of student heterogeneous groups in online university courses.....	33
3.1.1 Description of the activity.....	33
3.1.2 Machine Learning.....	35
3.1.3 Formation of the Heterogeneous Groups	35
3.1.4 Cluster Performance Evaluation Method	48
3.1.5 Selection of the Best Clustering Algorithm	49
3.1.6 Creation of Heterogeneous Groups	50
3.2 Research Activity 2: Tool for classifying students' sentiments in an e-learning course.....	51
3.2.1 Description of the activity.....	51

3.2.2 Sentiment Classification.....	53
3.2.3 Classification performance metrics	56
3.3 Research Activity 3: Intelligent chatbot as a virtual assistant for university courses with a large number of students.....	58
3.3.1 Description of the activity.....	58
3.3.2 Software Development	59
3.4 Research Activity 4: Automatic tutoring system for programming course	62
3.4.1 Description of the activity.....	62
3.4.2 Course and software development	62
3.4.3 Clustering.....	67
Chapter 4. Results and Discussions	68
4.1 Research Activity 1: Moodle plugin for creation of student heterogeneous groups in online university courses.....	68
4.1.1 Results of Clustering Algorithms	68
4.1.2 Comparison of Clustering Algorithms	74
4.1.3 Selection of the Best Clustering Algorithm and Development of the Software	75
4.1.4 Moodle Plugin.....	78
4.2 Research Activity 2: Tool for classifying students' sentiments in an e-learning course.....	80
4.2.1 Dataset	80
4.2.2 Data cleaning.....	82
4.2.3 Predictive model	83
4.3 Research Activity 3: Intelligent chatbot as a virtual assistant for university courses with a large number of students.....	86
4.3.1 Dataset	86
4.3.2 Predictive Model.....	87
4.4 Research Activity 4: Automatic tutoring system for programming course	89
4.4.1 Use of the online resources.....	89
4.4.2 Effectiveness of the online tutoring course	91
4.4.3 Students' perception.....	94
Chapter 5. Conclusions and Future Perspectives	97
Bibliography.....	105

Introduction

The Ph.D. thesis project is aimed at improving the quality and the effectiveness of on-line teaching in scientific degree courses at the University Level that required the use of E-learning platform, based on the Moodle Content Management System. The aim of this research project is to assist the teacher, through the development of new tools based on Artificial Intelligence, to design innovative successful e-learning courses to give to the students the opportunity to improve their learning outcomes. These originals tools overcome the limitations of the standard Moodle activities applying machine learning techniques by analysing large amount of students' data extracted by Moodle log data.

Recently many e-learning resources have been developed for university students, are available on the Web. The increase of LMS (Learning Management System) as Moodle and their ease of use led many teachers to realize e-learning paths for their students, often supporting them with some frontal activities, giving to them the advantages of on-line learning. The aim was to deepen the topics discussed in class through the consultation of additional materials, video recordings of lessons, and other activities to exploiting the potentials of on-line courses.

However, students can find difficulties to retrieve relevant materials to achieve their learning goals if they lack concepts and skills that don't allow to find what they want to learn (Mbipom et al., 2018).

Sometimes, even if the teacher did a good designing and building of the e-learning course, it can happen that the students didn't get benefits in terms of learning and performance, not achieving the learning outcomes expected (Nalli et al., 2020).

These items can negatively affect the student's motivation in the study; the lack of motivation can be one factor of school and University dropout (Fan et al., 2014).

E-learning platforms store several users' information which are useful also to analyse students' behaviour (Mostow et al., 2006).

They can record any type of activities performed by the students such as reading, writing, tests, tasks, and peer activities (Mostow et al., 2005).

The users' info stored by Moodle, are called Moodle log data. One way to check student progress in an online course is to monitor the Moodle log data (Młynarska et al., 2016).

Teachers can examine activities, logs, time spent and assessments to check the student progress in the on-line course. In this way they can suggest specific online activities for students that find difficulty, through feedbacks and tips to improve the students' engagement, skills, and knowledge. Different research studies analyse the relationships between Moodle log data and student success identifying which activities have significant impact on student success (Kadoić et al., 2018).

Although e-learning platforms provide reporting tools, it can be hard to extract useful information for a teacher that manages a course with a large number of students (Dringus et al., 2005). For example, the teacher that has only online interactions in e-learning platform with a high number of students, can find difficulty profiling student behaviour to create successful workgroups for collaborative activities, based only on monitoring students' Moodle Log data (Nalli et al., 2019). Another problem can also arise when teacher must interpret student's answers for a qualitative analysis. It could be tedious to manually go through all the qualitative comments and extract information (Gottipati et al., 2018).

These surveys provide valuable feedback that helps course designers towards improving teaching style, course content and assessment design and overall student learning (Lewis, 2001; Moore, 2005).

At the same time, the feedback needs to be analysed and interpreted with great care to ensure that action and improvement can come from the feedback process (Lizzio et al., 2002; Beran et al., 2005).

E-learning systems do not provide specific tools allowing educators to thoroughly track and assess all learners' activities to evaluate the structure and contents of the course and its effectiveness for the learning process (Zorrilla et al., 2005).

A very important area for overcoming this weakness could be the use of Artificial Intelligence (AI) techniques (Zaïane et al., 2001). In this field, the use of Machine Learning (ML) models is increasing in education (Bognár et al., 2020).

However, even if several AI tools are developed to facilitate the task of teachers to improve teaching, there are still many other gaps that need to overcome to allow teachers to be able to apply successful methodologies in online teaching.

The main goal of my PhD thesis is to provide solutions that can easily help teachers to increase the quality of their teaching, giving the possibility to design successful online courses, with the aim of increase student's engagement, satisfaction, and performance, using Machine Learning applied to Moodle log data.

I developed software applications to solve some limitations that the teachers at the University of Camerino encountered in their e-learning courses. These limitations did not allow the teachers to undertake successful educational action.

The focus of my research is based on four main aspects:

1. Design and implement a software that helps the teacher to create groups of heterogeneous students to allow successful collaborative activities.
2. Design and implement a new software that:
 - (a) allows the sentiments analysis of the students who follow an e-learning course;
 - (b) helps the teacher to edit and customize the online activities to improve student motivation while attending the course.
3. Design and implement a new software that can analyze and understand students' questions by providing the right answer through an automatic Chat. The software:
 - (a) allows the student having course information they need immediately, through the answers generated by the chat;
 - (b) helps the teacher in managing individual student support.
4. Designing of an online tutoring system for programming course to improve exam pass rate. In this course a Machine Learning technique (Clustering) is applied to Moodle log data to create similar groups of students, to define the most influential behaviors and activities that allow a fast pass exam rate.

Four scientific online courses, available in the Unicam E-learning Platform, are designed to check the effectiveness of the works previously described: “on-line Physics laboratory”, “Chemistry”, “Physics” and “Programming in Java”. These courses have been tested with the students at the University of Camerino.

The on-line Physics laboratory has the main goal to allow the students to participate in a practical example on how a real physics experiment is

conducted, following the experimental method of the physical sciences. The idea was to make the on-line laboratory experience as much real as possible, including the collaborative aspects which are typical of a laboratory. This was possible thanks to high resolution video recordings of real experiments. The videos of the experiments were realized in the Physics Division of the University of Camerino. With the collaborations of Physics professors, an important topic of the General Physics course was selected: the elastic force and the harmonic oscillations.

During the visualization of the videos experiment, the data can be extracted from the videos by the students in an interactive way and can be analyzed by them using the free software Gnuplot. The students had the possibility to repeat the visualizations of the videos many times, to extract the data and understand the experiment in a careful and complete way. The data were then critically analyzed by the students to formulate the physical law describing the phenomenon. The results of their analysis were reported in a document marked by the teachers. Finally, to reproduce the collaborative environment of a laboratory, the on-line collaborative activity consisting in peer assessment on the documents elaborated by the students was set.

The designing of the on-line Chemistry course was preceded by an accurate identification of topics related to stoichiometry, explained in the first part of the course, where students found difficulty in studying. This is confirmed by the bad students' performances in the stoichiometry task. The on-line course was composed by seven modules, one for each detected issue. The modules were designed to allow students to rapidly interconnect the three levels of representation in Chemistry (macroscopic, sub microscopic and symbolic). The macroscopic level was introduced by a short video clip of

the experiment related to the assignment. The sub microscopic level of the phenomena composed by computer animations, found on Chemistry teaching websites, used as open educational resource, that consist in free availability over the Internet and as few restrictions as possible on the use of the resource (Hylén, 2006).

The symbolic level consisted in a video tutorial that drove the students, step-by-step, to solve the stoichiometry exercises, related to the investigated chemical phenomena. Video tutorials were recorded with voice and handwriting, simulating teacher's exposition, and addressing students with different backgrounds of knowledge and problem-solving skills. Handwriting is accomplished by using a Wacom tablet. The videos last on average 15 minutes and the file size is around 150 MB. Detailed step-by-step explanations show the solution of the assigned problems and exercises, according to the principles and formulas of the symbolic level need for the specific task. Key information about this method of analysis and solution, as well as theoretical references, are included in the videos, with the aim to make the student able to apply the method to similar cases, once mastered the required skills.

Finally, at the end of the modules a questionnaire was inserted, consisting of questions aiming to test students' perception and satisfaction.

The on-line Physics course was a e-learning support for the face-to-face Physics course. It was composed by video lectures recorded during the frontal lessons of the Physics course. Video lectures were integrated with supplementary teaching material to provide student an interactive learning environment. 30 videos lectures (covering all the syllabus of the course) were recorded, for a total of 60 hours. These video lectures were used to support face to face lessons and not to replace them. As recording method,

the video lectures were recorded during the front lessons, and not in a neutral and empty environment in front of the webcam, for three main reasons: (i) reducing recording costs; (ii) saving working hours of the teacher; (iii) and face-to-face lessons motivate the students (Fritze 2003; Ronchetti, 2011). Students had the possibility to see video lectures, teaching materials, starting from the day of the face-to-face lessons at any time of the day, until the end of the final exam. The aim was to provide students an opportunity to see the materials several times, to find their individual learning needs, enhancing learning outcomes and performances.

The Programming in Java on-line course was an online tutorial system, and it was preceded by an accurate identification of the topics that the last years students found difficulties.

The main goal of this course was to acquire skills to create computer software that solve real problems, developing computational skills. The course was divided into 6 modules, each of which consisted in topics that allow the students to achieve their learning outcomes and to implement and create simple software to solve computational problems. In particular, the following topics were explained in each module:

1. primitive data types, cast, bitwise operators;
2. vectors and cycles;
3. control flows and logical operators;
4. classes and exceptions;
5. math and Util libraries;
6. inheritance and interfaces.

For each module, activities were developed to allow effective learning of students' knowledge and skills. Each module initially consisted of a video

tutorial, often used in e-learning courses to stimulate student engagement (Amendola D. et al., 2016).

The following activities were also included in each module:

1. multiple choice quiz;
2. text completion;
3. code evaluation;
4. exercise with solution;
5. programming exercise.

Students had to perform autonomously the “programming exercise”. Students have been requested to produce the source code on the java editor and check if it wasn’t correct.

The Java editor used in this course was an online compiler made by "Trinket" that was embedded into the Moodle course. It run the java code online and checked its correct execution.

When the source code was correct, students checked the quality of the syntax code. The students executed their code using the “interactive tutoring software”, especially developed for this online tutoring course. This software provided automatically immediate feedback on the quality of the code entered, analyzing criteria like “keywords”, “use of variables”, “use of methods”, “correct use of classes” to satisfy the object-oriented paradigms. Thanks to these tips, students were able to understand their mistakes and edited their code until they got positive feedback, improving their code without the human intervention.

In addition to the 6 modules, the course had an introductory video that explained: i) aims and structure of the course; ii) how to use the editor; iii) general intro to java topics. Finally, a questionnaire has been inserted

consisting of questions aiming to acquire interest in the study and to check students' perception and satisfaction.

The contents of the Ph.D. thesis are organized as follows.

In the *first* chapter I give an overview on the basic concepts and technologies.

In the *second* chapter I explain the state of the art and objectives. Starting from this overview, the operative objectives will be detailed, together with the original contribution I intend to give compared to the state of the art.

In the *third* chapter I describe the methods and the instruments used.

In the *fourth* chapter I report the results and discussions about the four aspects of the PhD research.

Finally, in the *conclusion* the key results of the thesis are discussed along with future perspectives.

Chapter 1: Basic Concepts and Technologies

This chapter is based on the basic concepts that inspired this Ph.D. project. Concepts of Machine Learning and the importance of its application in the education are described, in e-learning field.

1.1 Artificial Intelligence and Machine Learning

Artificial Intelligence (AI) is a discipline that tries to generate systems able to learn and reason as a human being (Colin et al., 2007). These algorithms learn from experience, having the ability to solve problems if some conditions happen, contrast information, and perform logical tasks. A software based on Artificial Intelligence can analyse big data, identify patterns and trends, and provide accurately predictions automatically. It makes experiences smarter (Zhou et al., 2019).

Machine learning (ML) is a part of AI that allow computers the ability to learn. It is based on data analysis, through which new patterns are identified that permit changes of the behaviour (Kim et al., 2019).

ML is the process to allow a model the access to data and letting it learn for itself. In 1959, Arthur Samuel uses the term “machine learning” to explain this theory, that is now the standard definition for the ability of computers to learn autonomously (Samuel, 1959).

Machine learning consists of algorithms that allow an optimization of performance using example data and experience (Alpaydin, 2009). Implementing a machine learning algorithm means implementing a model that outputs correct information given that we have provided input data. You can think of a model as a black box: data go in at the beginning, and some other data go out at the end — but the processes in between are complex (Kučak et al., 2018).

Machine learning is a branch of artificial intelligence that uses methods or algorithms for the automatic creation of models from data. Unlike a system that performs an activity following explicit and predefined rules, a machine learning model constantly learns from experience.

There are three main types of Machine Learning: supervised learning; unsupervised learning and reinforcement learning.

Types of Machine Learning

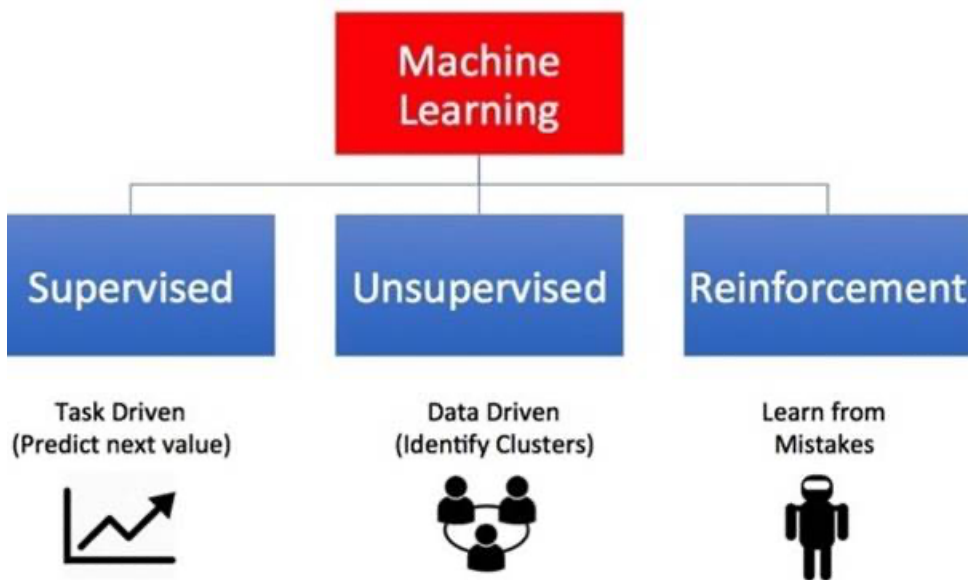


Figure 1 Types of Machine Learning. Image from <https://medium.com>.

The main goal in *supervised learning* is to learn a model from labelled training data (dataset) that allows us to make predictions about unseen or future data. As input data is fed into the model, it adjusts its weights until the model has been fitted appropriately.

Classification is a subcategory of supervised learning where the goal is to predict the categorical class labels of new instances based on past

observation. Another type of supervised learning consists in prediction of continuous outcomes, called regression analysis (Raschka, 2015).

The supervised process can be seen as a three-step process: training set, validation set and test set.

The training set is a set of data that is used to train and allow the model to learn the hidden features and patterns in the data. It should have different inputs so that the model is trained in various scenarios and can predict any unseen data sample that may appear in the future.

The validation set is a set of data, separate from the training set, that is used to validate our model performance during training.

The test set is a separate set of data used to test the model after completing the training and check how well the model perform.

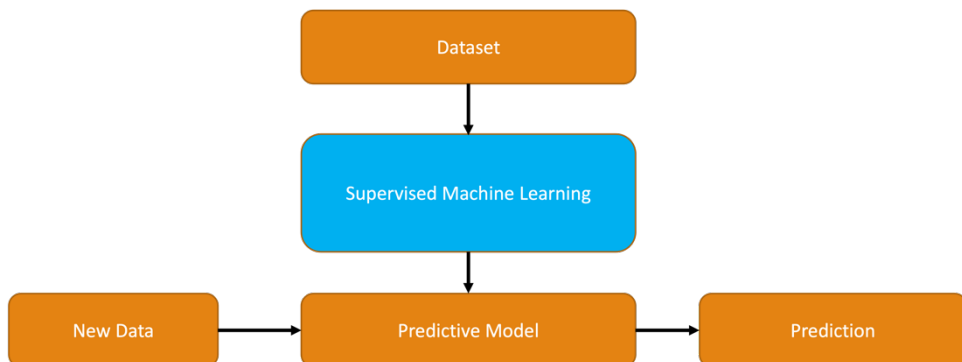


Figure 2 Workflow in supervised learning.

The goal of *reinforcement learning* is to develop a system that improves its performance based on interactions with the environment. Because of information on the current state of the environment includes a reward signal, this type of ML can be correlated with supervised learning.

In unsupervised learning we are dealing with unlabelled data or data with an unknown structure. By *employing unsupervised learning techniques*, it is possible to explore the data structure to extract useful information without the guidance of a known resultant variable or reward function (Raschka et al., 2019).

Clustering is an unsupervised technique that permits to organize a pile of information into meaningful subgroups, called clusters, without having no knowledge about their group memberships. Each cluster defines a group of objects that share a certain degree of similarity, but more dissimilar to object in other clusters (Raschka, 2015).

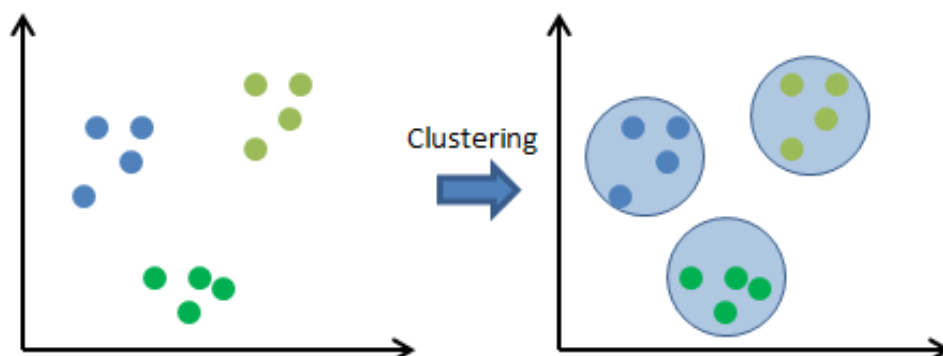


Figure 3 Clustering in Machine Learning. Image from <https://learnai1.home.blog/>.

1.2 Moodle E-learning platform

Moodle is a free and constantly evolving Learning Management System (LMS) and Course Management System (CMS), which allows the creation of virtual classes aimed at e-learning or on-line learning.

The fact that this software is free and easy to use has contributed to its rapid diffusion: currently, from the official Moodle statistics (stats.moodle.org) the total installations are 180000, and the number of its users is more than

350,000,000. It is mostly used by training centres and, in Italy, also by Public Administrations for training projects (especially schools and universities). Moodle can be installed on any platform that supports Php, Apache and MySQL and that is translated into more than 80 languages. Another aspect that further qualify Moodle is the administrative ease of tailoring the user interface with the possibility to move, delete or add modules. The system uses style sheets or CSS and PHP web pages to which it is relatively easy to make changes.

Moodle is the best possible choice for those who are looking for many functionalities and those who are planning to invest considerable resources in the training program at an organizational level. This project was created in the 90s, and it is still developing with the aim of creating a support for teaching, which is based on social constructivism. This is realised by integrating various tools for collaborative work (for instance, forum, messaging snapshot, integrated e-mail system, workgroup management, shared agenda, blog, wiki). A useful way for you to track the users' progress through Moodle course. Thanks to the Moodle logs data, reports track several Moodle activities and resources users viewed or list what activities need to be completed. There are multiple types of Reports available: logs, live logs, activity report, course participation.

The Unicam E-learning platform, that provide online course used in this PhD research, it is installed on a Virtual Machine based on Ubuntu, Apache Server, Mysql and Php. It is managed internally by the E-learning Office of the University, which deals with the technical management of the Virtual Machine and makes changes and customizations of the technical management of the application. Those changes are realised with graphic customizations of the design of online courses with innovative techniques

in the field of teaching ensuring support for educational activities for both students, tutors, and teachers of Unicam. The University of Camerino has implemented an internal system for managing digital identities. This is realised with an institution-oriented management by using, mainly, Single Sign On authentication service connected to the database Microsoft Active Directory, which consists of a hierarchical directory system, in which the profiles of all the professors, researchers, PhDs, students and administrative staff of Unicam are managed. This service is widely used, as it allows all users to access to the E-learning Platform with they own credentials that they access to all on-line platforms of the University of Camerino.

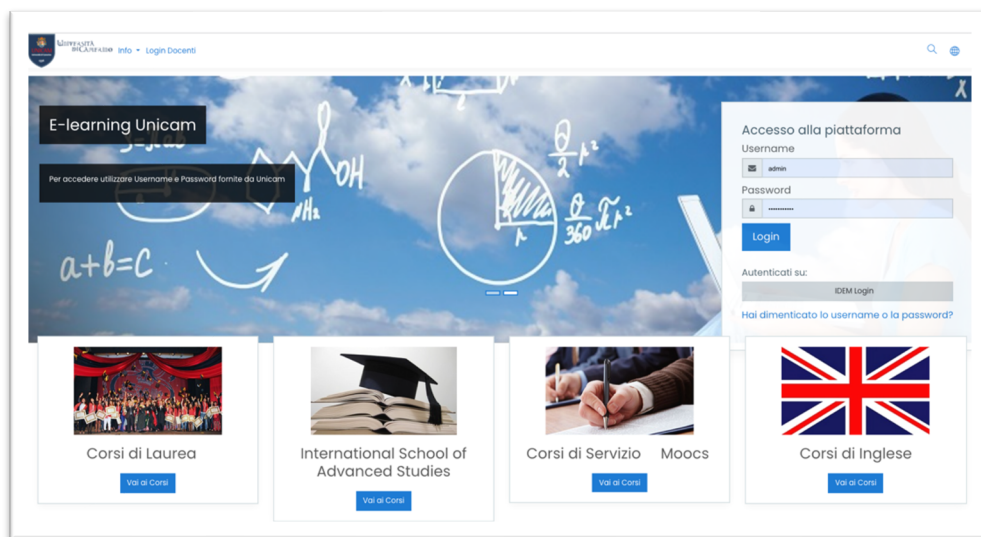


Figure 4 Unicam E-learning Platform.

1.3 Machine Learning and Education

Machine Learning is a computer science area, which uses, in much simpler words, statistical techniques to permit the computer systems to “learn”. Recently a lot of implementations of machine learning are developed in education, especially in the e-learning field.

ML can help the students improve their learning by analysing their academic performance and suggesting the best way to learn. Machine Learning applications in education are mostly used in:

1. adaptive learning;
2. learning analytics;
3. predictive analytics;
4. personalized learning experience;
5. assessment evaluation.

Adaptive learning is an educational method which uses computer algorithms as well as artificial intelligence to orchestrate the interaction with the learner and deliver customized resources and learning activities to address the unique needs of each learner (Kaplan, 2021). Machine learning is allowing us to analyse the performance of each student in real time and then modify the teaching methods and the curriculum based on the analysed data. The algorithm can assist in suggesting the learning paths that the learner should study like teaching materials and other better learning methodologies (Hamid et al., 2018).

The Machine Learning applied to *learning analytics* can help the teachers gain insight into the data and allow them to understand better the data. As the teachers shift through countless pieces of content, analyse it, make connections and conclusions, it improves the overall teaching and learning

process. Learning analytics give a lot of benefits to the students also by providing suggestions regarding materials and other learning methodologies (Kurilovas, 2019).

Predictive analytics in education focuses on understanding the mindset and needs of individual learners. ML in this area can help to anticipate the things that may happen in the field of education in the foreseeable future. The student grades can help the educators understand which students are going to perform better in the finals and who find difficulties (Brohi et al., 2019). ML has also made way for *personalised learning*, a customisable education model, where students have the freedom to choose the subjects they are interested in, and patterns they want to follow. Students can have to experience a learning environment that suits their needs and preferences based on (Rad et al., 2018).

Machine Learning can offer an *assessment evaluation* for assignments and exams more accurately than a human. Even though in some cases, human assistance is required for the evaluation to take place, ML ensures that the results are as accurate as they can be, offering an evaluation of students' performance with higher validity and reliability with low risk of error (Mahboob et al., 2016).

Based on this topic, research evidence different Machine learning applications in teaching field. These systems can be used occasionally to help teacher to calculate automatic on-line recommendations for active students based on their recent browsing history and based on their background, as well as exploiting similarities and dissimilarity between user preferences and the contents of learning resources. Some techniques are developed such as prediction, clustering, relationship, learning behaviour, course recommendation and adaptation to student behaviour (Gavriushenko

et al., 2017). In other cases, tools based on Machine Learning are used to predict student performance (Anozie et al., 2006), improve student retention (Đambić et al., 2016), support teachers and institution staff (Celar et al., 2015). Sometimes collecting and processing data, through clustering and classification techniques (e.g., Fuzzy C Means algorithm (FCM) and neural network propagation algorithm (GSPNN)) are used to divide the students into groups and associate them with specific profiles based on their learning style and interactions with e-learning activities (Kolekar et al., 2017).

Chapter 2: State of Art and Objectives

In this section a general state of the art and objectives regarding the main aspects developed in my PhD research are described:

- 1. Moodle plugin for creation of student heterogeneous groups in online university courses;*
- 2. tool for classifying students' sentiments in an e-learning course;*
- 3. intelligent chatbot as a virtual assistant for university courses with a large number of students;*
- 4. automatic tutoring system for programming course.*

It involves an overview on the state of the art of the application of the different on-line teaching methods used in this thesis, focusing on the limits that inspired the solutions proposed.

2.1 Research Activity 1: Moodle plugin for creation of student heterogeneous groups in online university courses

2.1.1 State of the art

Collaborative activities are a significant teaching methodology for students, as they improve learning outcomes by promoting active learning (Foote, 1997). Collaboration also allows students to develop their social skills, such as decision making, communication, collaborative skills, and critical thinking (Marjian Laal et al., 2012; Stevens et al., 2005; Amendola et al., 2018). Due to the large number of students enrolled in university courses, it is easy to run into organizational difficulties while planning collaborative activities. Today, the development of digital technologies allows the organization of collaborative learning experiences online in a more flexible way both for students and teachers (Abedin et al., 2012). Online learning

environments such as e-learning platforms are often used to encourage collaborative activities between students. The e-learning platforms are web 2.0 environments that emphasize participation, connection, and sharing of knowledge and ideas among students by providing tools designed for this purpose (McLoughlin et al., 2007). Both in the online environment and in frontal teaching, one of the fundamental factors that can influence the success of collaborative learning is student group formation.

This needs to consider the number and the heterogeneity of the components. Concerning the number of members, groups need to be formed by at least three or more people, although groups of four or five members are more effective (Alison et al., 2011).

Groups can also consist of two people, but in this case, the lack of creativity and variety of ideas does not encourage teamwork (Csernica et al., 2002). Another characteristic of success in group works is certainly the heterogeneity of students in terms of cognitive resources, characteristics, and behaviours (Nijstad et al., 2002).

In our university, the first experiences of online collaborative activities were based on shaping the groups by means of a random selection of students, without an underlying criterion. This led to several negative outcomes.

More in detail, according to our previous studies for an online laboratory of genomics, which included a peer review collaborative activity, a final questionnaire on the students' perception of the effectiveness of the collaborative activities demonstrated the intrinsic problems arising from the way the groups of students were formed. A random choice of the students forming a given group, as usually done by the standard Moodle plugin "Workshop", determined a sizable percentage of students, around 35%, which were not satisfied by the quality of the feedback received from their

peers on their report (Amendola et al., 2018; Amendola et al., 2016). This is the background motivation for our work: the formation of heterogeneous groups of students with a precise criterion, should allow the enhancement of the quality of the collaborative activities, thanks to the driving capacities of the high-performance students, distributed among different groups, avoiding the risk of clustering them, or excluding completely, in given groups and not in others.

In university courses, where e-learning platforms are also used for educational activities, the processing of Learning Analytics (LA) can be useful for creating heterogeneous groups of students based on their characteristics and behaviours, extracted from the platform itself. Unfortunately, in online learning environments, there is still no intelligent mechanism that allows the creation of heterogeneous groups automatically, facilitating the teacher's work. In recent years, some research groups have started working on Machine Learning projects applied to the LAs extracted from e-learning platforms to create heterogeneous groups automatically: some authors used data taken from an online questionnaire (Razmerita et al., 2011), while others relied on data extracted from the interaction of students within the Moodle forum tool (Maina et al., 2017). In the literature, online heterogeneous group formation is mainly based on the elaboration of LA extracted from a single activity carried out by students on the e-learning platform.

2.1.2 Objective

Our work consists of the creation of a software to implement a new Moodle plugin, which allows to elaborate all LAs extracted from various activities that students carry out during online courses, using the correlations between data. The aim is to obtain an overall view of the students' behaviours to

better characterize them, so that we are allowed to create heterogeneous groups. Our software creates heterogeneous groups of students automatically, by using unsupervised Machine Learning techniques. This is applied to the LAs produced by students during their use of an online course in our Moodle platform.

Although approaches for the creation of heterogeneous groups already exist, they are usually based on qualitative data (Maina et al., 2017). To the best of our knowledge, this is the first quantitative approach that compares clustering algorithms for creation of heterogeneous groups of students working online. Our approach has been implemented by realizing and using a new Moodle plugin and has been validated on a real course.

The contribution of our work lies in enhancing the teaching of scientific subjects through the development of an Intelligent Moodle Plugin, to improve the student's performance and acquire knowledge of the key concepts of the course. This Moodle plugin allows to create heterogeneous groups of students that are fundamental for the success of online collaborative activities such as peer review (Razmerita et al., 2011; Graf et al., 2006; Paredes et al., 2010).

2.2 Research Activity 2: Tool for classifying students' sentiments in an e-learning course

2.2.1 State of the art

Sometimes, despite the teacher having done a good job in the design and implementation of an e-learning course, it may happen that the students, even if they have participated in the online activities, do not find an improvement in learning or performance, not obtaining the results expected by the teacher. This can happen for example if the student experiences

negative sentiments during the use of a course that led him to not be motivated. Indeed, affective, and emotional factors seem to influence students' motivation and, in general, the outcome of the learning process (Shen et al., 2009). Students' feelings towards a course can serve as feedback for teachers, especially in the case of online learning, where contact is less frequent (Gkontiz et al. 2017). With this information on the emotions felt by the students, the teacher can verify if the course is useful or if it is difficult to understand (Ortigiosa et al., 2014). Thanks to the study of the feelings experienced by the students, we can perceive which activities tend to do not give motivation in the study, for example because the activity is too difficult or not clear. Understanding student emotions could be very important for the teacher that can eventually edit the course based on bad feedback (in a survey) left by students to enhance the motivation and engagement. This could be an indicator of the quality of the course for the teacher.

In the literature, Machine Learning techniques are used to extract the emotions of students, which are based on the processing and analysis of texts, for example posts or tweets from social networks (Hemalatha et al., 2013). One of the most used AI techniques for extracting emotions is Sentiment Analysis (SA). SA is a research area of Natural Language Processing (NLP) and Deep Learning in which written language is classified using a computational linguistics, according to the polarity of opinion and identification of emotions (Gkontzis et al., 2017).

2.2.2 Objective

Our project consists in creating intelligent software able to extract the students' emotions from the analysis of the texts of the open answers of a questionnaire delivered after the use of an online course.

The software will allow the teacher to have an overview of the feelings of the students. The teacher can analyze the student emotion and eventually edit the aspects of the course that produced negative feeling with the aim to change the students' emotions to improve their motivation and performance. Building a classification model, the teacher can make predictions after the changes of the course structure. In fact, he can verify whether there is an improvement on an emotional level or not, and if not, repeat this procedure. Through a continuous check of these data during the course, a virtuous circle will be created between teacher and student aimed at improving the e-learning course.

2.3 Research Activity 3: Intelligent chatbot as a virtual assistant for university courses with a large number of students

2.3.1 State of the art

Recently Universities found the spread of e-learning platforms to support frontal teaching. Students can benefit from these online tools as, for example, they can review the topics covered in class without space-time limitations, at any time of the day. However, it may happen that many students, despite having the need to ask the teacher for information, often fail to contact him physically and send him emails or using communication tools on the Moodle platform, such as Forum, chat, or private messages. In university courses with many students, answering all the students' questions about a particular course could be difficult for the teacher. This led to an

inefficient service especially for the student, who could wait several days before receiving an answer (Nalli et al., 2020). Indeed, teachers should manage all requests received by students through online tools like chat and forum. The use of a Chatbot can be an excellent choice because it automatically allows to provide answers to frequently asked questions at any time of the day, providing effective assistance to users. The Chatbot is an application that attempts to simulate the conversation of a human being through text or voice interactions (Shawar et al., 2007). In recent years, research led to the creation of Chatbots based on artificial intelligence techniques that interpret natural language to better understand and learn over time (Goyal et al., 2018). In e-learning systems, some prototypes are developed, aimed for example at keeping track of students' results and training activities (Zappi et al., 2018), or as a source of social learning, where students from different backgrounds can share their point of view and perspectives on a specific issue while the bot can adapt to each one individually (Hussain et al., 2018). One of the most used methods for creating Chatbot is Natural Language Processing (NLP) (Yan et al., 2016) which, using algorithms and technical specifications, can understand what the user writes, and what his requests are.

2.3.2 Objective

Moodle platform lacks a system that allows dynamic and intelligent technological support to the individual needs of users.

This main goal of this project consists in creating intelligent software, which, thanks to the use of Artificial Intelligence algorithms, can analyze and understand students' questions by providing the right answer in real time through an automatic Chat.

Once executed, the software presents a chat to the student, that can type the question in the appropriate field and immediately receive the right answer in a completely automatic way.

The software will allow: i) the student to have the course information he needs available in real time; ii) the teacher will have some help in managing individual student support. Indeed, the Chatbot will immediately provide answers to the most frequently asked questions, while the teacher will only have to answer those requests that the software cannot process.

2.4 Research Activity 4: Automatic tutoring system for programming course

2.4.1 State of the art

In the first year at university, starting a new programming course can be hard, especially for first year students. Sometimes they can also have difficulty to pass the exam. Programming courses are perceived as difficult. This can increase the student dropout rate (Robins et al., 2010) and can negatively affect student study (Ambrosio et al., 2011). A lot of students that don't have any programming experience before, think that programming is difficult to learn and use. They refuse to learn programming skills unconsciously (Tan et al., 2009). Students' views regard the difficulties that they faced in programming themes that cover semantics, programming skills and programming knowledge (Özmen et al., 2014). Many students had difficulties developing computational thinking. Computational thinking is an attitude that permits problems solving using computational techniques (Wing, 2006). Recent studies identify teaching methodology as the main reason for problems in teaching and learning

programming. They rarely focused on developing the skills and knowledge acquired in the student's learning experience (Figueiredo et al., 2020). Students must study not only theoretical concepts, but also interpret their own mistakes, find examples, and search solutions to known problems. The tutorial aspect is important because it can allow the improvement of the quality of teaching. In fact, it can drive the student to acquire generic and disciplinary skills. Tutoring represents an action to support the different aspects of a student's learning development. It helps students during their study and supports them in any difficult moments, such as the new approach to university study. The purpose of tutoring is to provide students to approach the critical steps of their studies in a good way. It is aimed at promoting academic success and avoiding university drop-out (Da Re, 2018). Sometimes it can happen that the degree courses in Computer Science are composed of many students enrolled. IT resources are often used to have a good management of classrooms with a lot of students. These resources are very effective because they allow to improve learning processes, thanks to new pedagogical approaches like e-learning courses applied in the field of Computer Science. Indeed, the use of Mooc for teaching IT in a university course with a high number of students, showed how these teaching methods have been effective, and expanded the students' computer knowledge and skills. Furthermore, the use of the online course did not reduce the teaching effectiveness and allowed the improvement of the student's learning results (Guelfi et al., 2020). Blended programming e-learning courses provided some help for students to gain practical experience through the involvement of Java programming activities. They allowed students to learn more easily and are adapted to acquire programming skills (Hadjerrout, 2008).

Virtual Laboratories are also provided in e-learning for learning techniques and knowledge relating to computer programming. These online labs provide hands-on simulations that permit the students to learn comprehensively, through face-to-face lessons (Richter et al., 2012). In online laboratories for learning graphic programming software such as Gnuplot, Video Tutorials are often used for data collection applied to the code (Amendola et al., 2016). Given the success of e-learning courses in education, its effectiveness for student learning, and the difficulty of managing a lot of students in the classroom, a programming tutoring e-learning course is created.

An online programming tutoring course can be very useful because E-learning platforms, like Moodle, allow the integration of different tools for programming purposes. Furthermore, they give the possibility to set for each tool automatic feedback. Feedback helps students to understand both the learning outcomes of the course and their relative progress toward meeting those goals (Hattie et al., 2007). Research evidence shows how effective feedback is an important element for student learning (Shute, 2008). Feedback finds many applications in the programming field, and it is used to contribute to improving students' cognitive outcomes (Gusukuma et al., 2018). One of the fundamentals criteria that affect efficacy in feedback is the timing. Immediate feedback gives a lot of advantages, like the promoting of actively learning. It allows students' partial knowledge to be rewarded with partial credit and it is strongly preferred by students compared to other techniques (Di Battista, 2005). An example of immediate feedback is the IF-AT that provide to students' immediate feedbacks about the accuracy of their answers, as the students are completing each exercise. The IF-AT system provides immediate affirmative feedback or corrective

feedback, depending on the answer given by the students (Epstein et al., 2002). Research evidence shows how immediate feedback is more effective compare the delays feedback, on complex tasks, when students have less prior knowledge (Shute, 2008) like the new university students that approach programming. Students involved in immediate feedback got a high level of scores and correct answers than the other students. This method engages students in the process of discovering and correcting initially imprecise response strategies (Epstein et al., 2002).

Learning Management System (LMS) like Moodle permits to create an effective programming tutoring e-learning course, because it covers most of the features that allow the successful learning of java programming, like videos, online editors, and different types of multiple-choice quizzes. These tools allow the integration of video tutorials for writing code, give the possibility to perform the code and, thanks to the feedback functions, check errors in the execution of the created software, check theoretical knowledge, evaluate the code, test the functionality of the code. However, testing the quality of the syntax produced by the student is important to complete an online Programming Tutoring course.

An online editor can recognize errors at runtime; anyway, if the program runs correctly, the code may not be written optimally. It could not respect the logic of the object-oriented paradigm, or there could be an improper use of java commands that could create optimization problems. Unfortunately, in Moodle there is no automatic tool that allows the students, based on the code described, to have feedback and advice to improve the quality of the code syntax of the Java programming language. A new tool was created to analyse the syntax of the written code and, like a tutor, automatically provides feedback and tips to improve quality.

2.4.2 Objective

The aim of the work is to realize an effective online Tutoring course that contributes to an improvement in the quality of teaching in the Java Programming course. The final goal is to demonstrate how it improves students' computer skills and knowledge and check if it can give benefits to students, in terms of performance (if they quickly pass the final exam) and in terms of satisfactions (analyzing also the questionnaire based on student perceptions).

Indeed, I want to evaluate the effectiveness of the online tutoring course, to ensure an effective tool for improving the study approach to students.

The evaluation of the course was analyzed by the following research questions:

1. *How did the students use the online resources?*
2. *Has the massive use of the E-learning platform helped students to pass the exam within the academic year?*
3. *How did students perceive the effect of online resources in acquiring skills and passing the exam?*

A teaching model was designed and adopted to obtain the answers to these research questions. Logs for the various activities were extracted from the e-learning platform and processed using Machine learning techniques. In this way it was possible to determine and compare the behaviors held by the students on the platform and how they affected the results of the final exam. Finally, the questionnaire filled by students was analyzed.

Chapter 3. Methodological Aspects and Models

In this chapter I focused the attention of the methodological aspects relating to the different tools and applications developed in this PhD research. In particular, they cover these main aspects: i) description of the activity used like a pilot course, ii) research methods adopted to prepare the on-line activities and to analyze all the possible data and learning outcomes, iii) the design of the software, describing the environment, the activity flow, the methods and machine learning techniques, iv) description of the main steps for the software development that consist in data preparation, data analysis and data visualization.

3.1 Research Activity 1: Moodle plugin for creation of student heterogeneous groups in online university courses

3.1.1 Description of the activity

An online physics laboratory was made available through the University's Moodle e-learning platform and involved 54 international students. The educational path was structured in two parts: an individual learning part and a group project part.

In the individual learning part, students learn by using 5 video experiments, carried out in the laboratory. These are related to the elastic force and the harmonic oscillator. Each video experiment lasts an average of 10 min for a total of 50 min. Online video experiments allow students to examine elastic phenomena, collecting and processing experimental data by using the free Gnuplot software to obtain a data fit and find the mathematical formulation of the corresponding physical law. Two video tutorials are available on the platform for using the software: (i) how to install Gnuplot; (ii) how to use Gnuplot to create a graph, prepare figures and process data. The video tutorials last 15 min. There are also other files (pdf and web

pages) with text and images, which contain further explanations relating to Gnuplot and the theoretical part of the video experiments. At the end of the learning part, students must perform on an online test. This must be done on the platform via the Moodle submission form, where each student must answer a series of questions and solve exercises explained by video tutorials and video experiments. The essay is evaluated by the teacher.

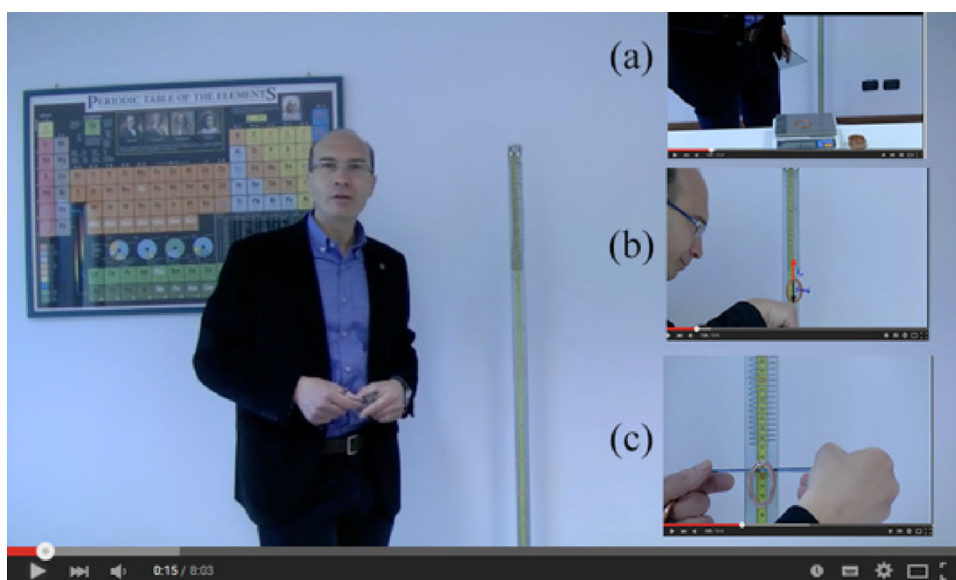


Figure 5: Screenshots of a video experiment to collect experimental data.

The second part (i.e., group project part) is characterized by a collaborative activity. This consists of a group project that must be performed on a topic that is proposed by the teacher. To make the collaborative activity more effective, in this step heterogeneous groups of 4 or 5 students are created. To reproduce the collaborative experience of a laboratory in an online environment, we organized the peer assessment on the documents elaborated by the students. From the feedback obtained by the other students belonging to the same heterogeneous group, the student had the possibility

to modify his essay. The final version of the essay made by the student will also be evaluated by the teacher.

3.1.2 Machine Learning

For group formation Unsupervised Machine Learning techniques have been applied to the Learning Analytics that were produced by the students during the individual learning part. There are several clustering algorithms that can be used to form groups of students based on their behavior while taking an online course. These algorithms are used for several purposes: for example, the “Self-Organizing Maps” algorithm can be used to group students by analyzing their background while the “Fuzzy c-means” algorithm allows to group students according to their personality and their learning strategy (Vellido et al., 2010). After testing and comparing different clustering algorithms, we chose the K-Means clustering algorithm because it was the best in terms of group formation (Vellido et al., 2010). In this work, to create heterogeneous groups, I first applied clustering techniques using the K-means algorithm to organize groups of students with similar characteristics (behavior while using the online platform). These data were obtained from the processing of the learning analytics provided by the Moodle Reports extracted from the platform. Then an algorithm distributes the students of the same groupings into different groups, which are therefore heterogeneous within.

3.1.3 Formation of the Heterogeneous Groups

This section describes how student behavior data were collected from the platform and how data were processed for heterogeneous student group creation. This collaborative activity belongs to the second part of the Physics laboratory course. Many studies provide a theoretical framework for our work. The learning analytics that are related to the platform student behavior are very useful in teaching; this is also confirmed by several research studies. For instance, the frequency of access to e-learning platforms can be used to predict a student's final grade (Froissard et al. 2015). In (Jo et al., 2014), an effective way was shown to use user log data, such as total online time and login frequency, as predictors of learning performance. In (MacFadyen et al., 2010), the use of student tracking data such as time spent online, numbers of logins, numbers of files viewed, web links, and exercises uploaded to the platform, correlate with the student's final grade. The dataset contains the list of the students enrolled in the online courses. We chose to import all logs from the Moodle database, because they indicate the material that each student has seen or not and how often. The list of the features extracted from the Moodle platform allows the calculation of different aspects of the student learning process, such as "presence coefficient", "study coefficient", and "activity coefficient" (Bovo et al., 2013). These coefficients permit the identification of student behavior using clustering techniques. In particular, the following features were selected: login frequency, last login, total time spent online, number of video tutorials viewed, frequency of video tutorials viewed, number of video experiments viewed, frequency of video experiments viewed, number of web pages viewed, number of pdf files downloaded, number of exercises performed.

All the forementioned data will be used in our approach to analyze student behavior.

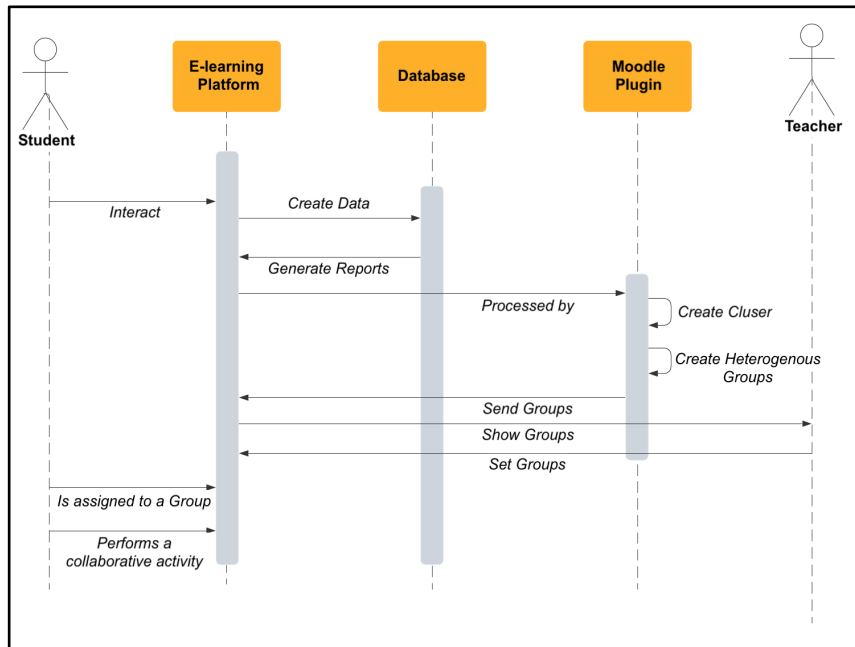


Figure 6: Sequence diagram of the flow of activities, research, and software development.

The Figure 6 shows the flow of online activities, research, software development and the creation of heterogeneous groups. The process for forming heterogeneous groups required distinct work phases:

1. implementation of clustering techniques for the formation of homogeneous groupings for similar characteristics and behaviors of the students during the use of the individual online course;
2. performance evaluation of cluster algorithms;
3. selection of the best clustering algorithm;
4. development of an automatic software able to select the students from the different homogeneous groupings for the formation of heterogeneous groups.

During the first step of the research project, after a selection, I extracted all the Learning Analytics related to student behavior produced on the platform after completing the individual online course part: login frequency, last login, total time spent online, number of video tutorials viewed, frequency of video tutorials viewed, number of video experiments viewed, frequency video experiments viewed, number of web pages viewed, number of pdf files downloaded, number of exercises performed. To simplify the datafile, I grouped them into features, i.e., data belonging to the same type of activities (e.g., visualization of the 5 video experiments) have been aggregated into the same feature. The feature is one individual and measurable property of an observed phenomenon (Bishop et al., 2006). For the characterization of the students and the creation of homogeneous groupings, we used only quantitative data. We didn't use the evaluation of the tasks, to avoid that the weight of the grade could influence the process subdivision of students. The evaluations of the individual essays were instead used later to verify the effectiveness of clustering techniques. More precisely, I compared the essay evaluations with the clusters (groupings) to check if there was a correlation between them, to verify if the behavior of students influenced their final performance. Before applying the clustering algorithm, the collected data are pre-processed. Each student was represented by an "input vector" with features related to the values of the attributes associated with the student.

All data, organized in feature vectors, one for each student, were inserted into a single Excel file, called a dataset, which represents the software input file to create clusters. To have the same scale of values, the data have been normalized in a range from 0 to 1. I then moved on to the realization of the software for the creation of homogeneous groupings using Python as a

programming language, because it has many libraries for data processing using Machine Learning algorithms. Initially the software takes the dataset as input and processes it to determine the number of clusters to create. This number must be provided to the clustering algorithm.

For the selection of the algorithms, an analysis of the best clustering algorithms was carried out to better determine the behavior of the students in the e-learning platforms. Based on the literature (Alias et al., 2017; Kausar et al., 2018), the best clustering algorithms were selected to identify student behaviors in the e-learning environment. I decided to test six different clustering algorithms, to determine which algorithm was best for our purpose:

1. K-means;
2. Mean-Shift Clustering;
3. Agglomerative Clustering;
4. Density-based spatial clustering of applications with noise (DBSCAN);
5. Gaussian Mixture Models Clustering;
6. Self-Organizing Map (SOM).

K-Means

The algorithm uses a K number of clusters (to be set before execution), for which you want to divide the dataset and assign a centroid to each cluster, which must represent the central point of each cluster. To determine the appropriate number of clusters, a function was developed in Python, using the “Elbow Method” (Hackeling, 2014), an interpretation method within the cluster analysis to find the appropriate number of clusters in a dataset. The method allows to generate a graph, having in the axis of the abscissas a

range of values of K (i.e., the number of clusters in which the dataset is divided) and on the ordinate axis the sum of the distances of observed data from cluster centroids, called WithinCluster-Sum-of-Squares (WCSS). The number of K (value in the abscissa axis) where, in the graph, the decrease in the value of WCSS causes a significant drop in speed of increment (when K increases), it is called “elbow”. This value represents the optimal number of clusters to build based on the dataset provided. The next step is the execution of the clustering algorithm. This takes the number K of clusters obtained from the elbow method for subdividing the dataset and assigning a centroid to each cluster. The choice of the centroid, in the first iteration, occurs randomly. The proximity of a vector (set of data relating to a certain student) to the centroid of a cluster establishes the belonging of that vector to that cluster. The algorithm calculates the Euclidean distance between each vector x and each centroid, assigning the vector to the centroid c for which distance is minimum:

$$c = \operatorname{argmin}_{c_i \in C} \operatorname{dist}(c_i, x)^2$$

where “ c_i ” represents a centroid of the set C (set of centroids), x represents the input vectors, while “ dist ” is the standard Euclidean distance. Then, the values of the centroids are recalculated.

The new value of a centroid will be the average of all vectors that are been assigned to the centroid cluster. The execution continues by iteration (a maximum number of iterations are defined a priori for avoid an infinite loop) and ends when:

1. no vector of the dataset changes clusters;
2. the sum of the distances is reduced to a minimum;

-
3. a maximum number of iterations is reached.
 4. The steps to achieve clustering using K-means algorithm are described here:
 5. select the number of clusters(K) with Elbow Method to obtain the data points;
 6. insert the centroids c_1, c_2, c_k randomly;
 7. repeat 4 and 5 points, until convergence or the end of a fixed number of iterations;
 8. for each data point x_i :
 - a. find the nearest centroid ($c_1, c_2 \dots c_k$);
 - b. assign the point to that cluster ;
 9. for each cluster $j = 1..k$:
 - a. new centroid = mean of all points assigned to that cluster;
 10. end.

Mean-Shift Clustering

Mean-Shift clustering is a powerful clustering algorithm used in unsupervised learning. Unlike the popular K-Means cluster algorithm (Duda et al., 2001), there are no built-in assumptions about the shape of the distribution or the number of clusters. But the number of clusters is determined by the algorithm relative to the data, but this has a cost in terms of time.

This algorithm was improved by Comaniciu, Meer and Ramesh to low-level vision problems, including, segmentation, adaptive smoothing (Comaniniu et al., 2002) and tracking (Comaniniu et al 2003).

Since it is a sliding-window-based algorithm, it attempts to find dense areas of data points. It is also a centroid-based algorithm, which has the goal to

locate the central points of each group/class, and this works by updating candidates for central points to be the mean of the points within the sliding-window. And then these candidate windows are filtered in a post-processing step to eliminate neighboring duplicates, forming the final set of central points and their corresponding groups.

Mean-Shift has two important parameters. The first parameter is the bandwidth that sets the radius of the area (i.e., kernel), used to determine the direction to shift. In our analogy, bandwidth was how far a person could see through the fog. We can set this parameter in one of two ways: manually or automatically. By default, a reasonable bandwidth is estimated automatically (with a significant increase in computational cost). Second, sometimes in mean-shift there are no other observations within an observation's kernel. That is, a person in this case cannot see a single other person. By default, Mean-Shift assigns all these "orphan" observations to the kernel of the nearest observation. However, if we want to leave out these orphans, we can set "cluster_all = False" wherein orphan observations the label of -1.

The steps needed to achieve clustering using the Mean-Shift algorithm are:

1. We consider a set of points in two-dimensional space or more. Then we begin with a circular sliding window centered at a given point from the dataset (randomly selected) and having radius r as the kernel. The mean shift is a hill-climbing algorithm that shifts the kernel iteratively to a higher density region on each step until convergence;
2. during each iteration, the algorithm shifts the sliding window towards regions of higher density by shifting the central point to the

-
- mean of the points within the window. The density within the sliding window is proportional to the number of points inside it. Obviously, by shifting to the mean of the points in the window it will gradually move towards areas of higher point density;
3. it continues the shifting process of the sliding window according to the mean until there is no direction at which a shift can accommodate more points inside the kernel;
 4. the process of steps 1 to 3 is repeated with many sliding windows until all points are inside one window. If multiple sliding windows overlap, the window containing most points is preserved. Finally, the data points are grouped according to the sliding window in which they reside;
 5. end.

Agglomerative Clustering

The agglomerative clustering is a method of hierarchical clustering, and it is used to group objects in clusters based on their similarity.

Algorithms in this category generate a cluster tree (or dendrogram) by using heuristic splitting or merging techniques (Omran et al., 2007). A cluster tree is defined as “a tree showing a sequence of clustering with each clustering being a partition of the data set” (Leung et al., 2000).

Agglomerative clustering is also known as AGNES (Agglomerative Nesting) is a bottom-up approach. This algorithm starts by treating each object as a singleton cluster. Then, pairs of clusters are successively merged until all clusters are merged into one big cluster containing all objects. The result is a tree-based representation of the objects.

The steps we need to perform to achieve clustering using the agglomerative clustering algorithm are:

- firstly, we need to do is to consider each data point as a cluster;
- during this phase we take the two closest clusters and try to merge them into a cluster;
- we repeat step 2 until all clusters are merged;
- end.

Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

Density-based spatial clustering of applications with noise (DBSCAN) is a well-known unsupervised learning method that is commonly used in machine learning and data mining. The DBSCAN algorithm (Ester et al., 1996) is a famous algorithm used in detecting arbitrary-shaped clusters and removing noises. Based on the dataset provided, DBSCAN will group together points that are close to each other using the distance measurement (usually Euclidean distance), and a minimum number of points. It will also mark, as outliers, the points of the dataset that are in low-density regions. The DBSCAN algorithm determines that clusters are dense regions in data space, separated by regions of the lower density of points, so in simple words, it is based on “clusters” and “noise”. The DBSCAN algorithm basically requires 2 parameters to work:

- eps: this parameter specifies how close points should be to each other to be considered a part of a cluster; if the distance between two points of the dataset is lower or equal to this value (eps), these points are considered neighbors by the algorithm;

-
- **minPoints:** It specifies the minimum number of points within eps radius. For instance, if we set this parameter to 10, then we will need at least 10 points of the dataset to form a dense region.

The steps required to achieve clustering using the DBSCAN algorithm are:

- the algorithm finds all neighboring points within the eps to identify all central or visited points that have multiple neighboring MinPts;
- during this phase, for each central point if it is not already assigned to a cluster, a new cluster will be created;
- then recursively all its connected density points will be found and assigned to the same cluster as the central point. At this point, a point “x” and “y” are connected in density if there is another point “z” that has enough number of neighboring points and both points (“x” and “y”) are within the distance “eps”. This is called the “chaining process”. So, in other words, if “a” is the neighbor of “b” and “b” is the neighbor of “c”, and “c” is the neighbor of “d”, which is also the neighbor of “e”, it implies that “a” is the neighbor of “e”.
- during this last step, all remaining unvisited points in the dataset are iterated. Those points that do not belong to any cluster are called “noise”;
- end.

Gaussian Mixture Models Clustering

Using GMM (Marin et al., 2005), each cluster is described by its centroid (mean), covariance, and cluster size (weight). Instead of identifying clusters

with the “closest” centroids, we will fit a set of k Gaussians to the data set. Then we estimate gaussian distribution parameters such as mean and Variance for each cluster and the weight of a cluster. After we have learned the parameters for each data point, we are able to calculate the probabilities of it belonging to each of the clusters.

So mathematically we can define a gaussian mixture model as a mixture of K gaussian distribution that means it is a weighted average of “ k ” gaussian distribution. So, we will write data distribution as below:

$$p(x) = \sum_{k=1}^K \pi_k N(x|\mu_k, \Sigma_k)$$

where “ $N(x|\mu_k, \Sigma_k)$ ” represents the cluster in the dataset mean “ μ_k ”, covariance “ Σ_k ” and “weight π_k ”. So, after we learn the parameters “ μ_k, Σ_k ” and “ π_k ” we learnt “ k ” gaussian distribution from the data distribution, that “ k ” gaussian distributions are clusters.

Self-OrganizingMap (SOM)

Self-organizing map (SOM) (Kohonen et al., 2006) is a neural network method (the most popular among other neural network methods) used for cluster analysis. It is a clustering technique aimed at discovering categories in large datasets, for example to find customer profiles based on a list of past purchases. In SOM the neurons (called nodes or reference vectors) are arranged in a single two-dimensional grid, which can take the form of hexagons or rectangles as in Figure 6:

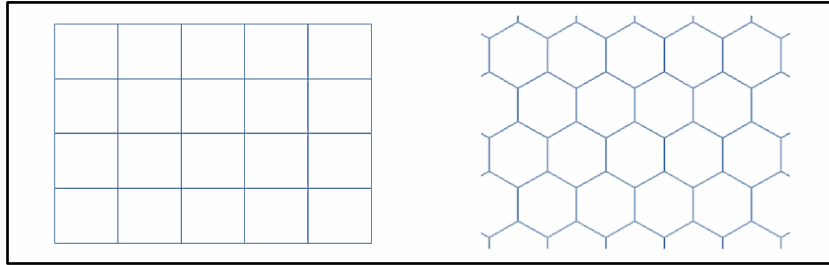


Figure 7: Self-Organizing Map (SOM) grids. The neurons are arranged in a single two-dimensional grid, which can take the form of hexagons or rectangles. Through many iterations, they will merge around areas with high density of data points.

Through many iterations, neurons on the grid will merge around areas with high density of data points. So, areas where there are too many neurons may reflect underlying clusters in the data. As the neurons move, they inadvertently bend and twist the grid to reflect the overall topological shape of our data more closely. The steps required to perform to achieve clustering using SOM are:

- each node is initialized with a weight;
- a vector from the training data is chosen randomly;
- each node is analyzed to calculate which weights are most as the input vector that was chosen. The winning node obtained is commonly known as the Best Matching Unit (BMU);
- then the neighborhood of the BMU is calculated. Now the number of neighbors decreases over time;
- the winning weight is rewarded by becoming more like the sample vector. But neighboring vectors also become more such as the sample vector. When a node is close to the BMU, its weights are also altered;
- the step “b” is repeated for “N” iterations;
- end.

3.1.4 Cluster Performance Evaluation Method

After applying different machine learning algorithms on the data, it was necessary to measure cluster performance. There are many metrics to examine the quality of clustering results when the method is unsupervised. These metrics can give a view on how the clusters change depending on the algorithm selected and the natural trend of the data to group. It was selected Silhouette Analysis (De Amorim et al., 2015), an internal evaluation method. Internal evaluation methods are more precise in the real group number determination compared to external indices. Furthermore, Silhouette Analysis was selected because it provided most accurate results compared to other internal evaluation methods (Rendon et al., 2011).

Silhouette Analysis

Silhouette analysis is a method used to interpret and validate the consistency within clusters of data. The silhouette value is a measure that returns how similar an object is to its own cluster (called cohesion) compared to other clusters (called separation). It is used to study the separation distance between the clusters found. With the silhouette plot, we can see how close each point in one cluster is to points in the neighboring clusters and this provides a way to evaluate cluster parameters, such as numbers.

This validation technique calculates the average silhouette index for each cluster, the silhouette index for each sample, and overall average silhouette index for a dataset. In this way, with this approach each cluster could be represented by the Silhouette index, which is based on the comparison of its tightness and separation. The Silhouette Coefficient is measured using the following formula:

$$S(i) = \frac{b(i) - a(i)}{\max \{a, b(i)\}}$$

where:

1. “ $a(i)$ ”—is the average distance of the point i from all other points in the same cluster;
2. “ $b(i)$ ”—is the smallest average distance of the point i to all points in the closest cluster. Now, obviously, the range of the silhouette value $S(i)$ will be between $(-1, 1)$.

If the silhouette value is close to 1, the sample is far away from the neighboring clusters. So, the sample is well-clustered and assigned to an appropriate cluster.

If the silhouette value is about 0, the sample is very close to the neighboring clusters and so it could be assigned to another cluster closest to it. And this means it indicates an overlapping cluster.

If silhouette value is close to -1 , the sample is assigned to the wrong cluster and is simply placed somewhere between the clusters.

So according to this method, it's needed have coefficients to be as large as possible and close to 1 to have good clusters.

3.1.5 Selection of the Best Clustering Algorithm

At the end of the executions of the different algorithms, groupings of students are generated, who have similar behaviors and characteristics in the e-learning platform.

To see which algorithm was the best, the cluster performance results were compared, using the “Silhouette Analysis” method.

In this way, the most performing algorithm was selected and used for the realization of the software. The software will use the clusters generated by the best algorithm for the creation of heterogeneous groups.

Finally, the marks of the individual task were used to be able to see if there are similarities between students, belonging to the same grouping. These similarities are not only checked at the level of behavior but also at the level of performance, to ensure greater heterogeneity in the formation of heterogeneous groups.

The execution of the software continues with the formation of heterogeneous groups, through an algorithm that selects students from different clusters and places them into heterogeneous groups.

3.1.6 Creation of Heterogeneous Groups

After the creation of the clusters, the process continues with the execution of a Python function based on a new algorithm developed to divide the students belonging automatically and uniformly to different clusters in different heterogeneous groups.

Each grouping of students obtained from the cluster represents the different levels of participation in platform activities and identifies different types of behavior for each cluster.

The algorithm selects and inserts at least one student belonging to a different type of cluster into each group. This ensures that in each group there are students with different characteristics and behaviors, to improve student performance in collaborative activities.

The function first determines the number of groups to create, calculated on the number of students to be included in the group, which must be set at the beginning. It then sorts the cluster lists in ascending order, from shortest to

longest and calculates, based on lengths, how many students for each cluster to include in a group.

$$nr_groups = round\left(\frac{nr_students}{stud\ for\ group}\right)$$

Subsequently, the algorithm selects the students based on their own ID, taking one or more values at a time from each cluster (based on the calculations considering the lengths of the clusters) in sequence and saves them in a single new list in the order in which they are taken from the different clusters.

$$group[i]_{nr_students} = round\left(\frac{len(cluster[i])}{nr_groups}\right)$$

The resulting list is then divided into sub lists by defining an interval. This range represents the number of participants that you want to get within each group. In this way, we will be sure that at least one component of each cluster (excluding the case where the number of members of the groups is not less than the number of clusters) will be part of each sub list (representing each individual group). In this way we obtain internal heterogeneity among the students. At the end of this phase, the teacher can see the list of heterogeneous groups containing the Student IDs.

3.2 Research Activity 2: Tool for classifying students' sentiments in an e-learning course

3.2.1 Description of the activity

To develop the tool, an online tutorial on Chemistry was used to support frontal teaching for the first year of the degree course in Biosciences and Biotechnology at the University of Camerino delivered through the Moodle platform. 122 international students attended the course.



Figure 8: Structure of Module 1 in Chemistry course.

The course consisted of 7 modules each of which covers a specific topic of chemistry. Each module features:

1. "video experiment", video experiment relating to the exercise with a maximum duration of 5 minutes;
2. "submicroscopic view", (image or video maximum of 2 minutes);
3. "overview", overview of the resolution phases of the exercise;
4. "video tutorial", video of the exercise, with a maximum duration of 20 minutes;
5. "multiple choice", multiple choice quiz;
6. "other materials", any supporting materials (videos, images, texts, and files).

At the end of the course, students were asked to answer a satisfaction questionnaire with closed / open answers. The open answers "What you liked most about the course and the platform" and "what you liked least about the course and the platform" were those used in this project for the analysis of students' feelings.

3.2.2 Sentiment Classification

In this section we describe the methodology used for collecting and processing data to extract the feelings of the students and create a classification model. Figure 9 shows the flow of on-line activities, research, and software development.

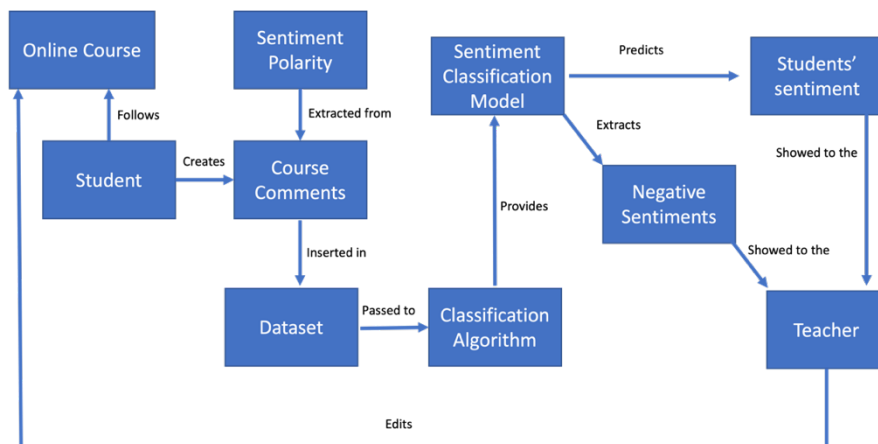


Figure 9: Block diagram of the flow of online activities, research, and software.

Three phases were required for the development of the software:

1. selection of the data to be analyzed;
2. choice of Machine Learning techniques;

3. data visualization.

In the first phase, all the results of the satisfaction questionnaire carried out by the students were extracted, downloading the related Excel file directly from Moodle's Feedback activity.

Among all the answers of the questionnaire, only the open answers were selected which represented the students' opinions about the course, in relation to the aspects they liked, and which did not. These data were organized, in a single Excel file, in feature vectors (Gkontzis et al., 2017), associating their answers with the students. The Excel file, called dataset, represents the input file of our sentiment classification software.

However, to develop a predictive model in a supervised algorithm such as Classification, it is not enough to have only the text to be analyzed. In fact, our dataset must be complete with labels indicating the polarity of the sentiment associated with that text, providing samples that the machine learning algorithm can study, extracting a function that will allow it to make predictions future on new unclassified texts.

To complete the dataset, it was therefore necessary to associate manually the polarities of the sentiments relating to the texts, indicating whether these are negative, positive, or neutral.

Once the complete dataset was obtained, it was possible to implement the software, choosing to apply a Classification technique that allows to make predictions on students' sentiments based on the text entered. This technique requires an input in numerical values; so, it was necessary to apply a method that would allow you to convert the text into a number to extract functionality. To do this, the Term Frequency – Inverse Document Frequency (TF-IDF) algorithm was used, a statistical model that aims to

determine how relevant a word is in each document (Rajaraman et al., 2010).

This algorithm allows to know **how relevant a word** is for the purpose of extracting a feeling and it is based on these formulas:

$$\textit{Term Frequency: } \frac{\textit{number of repetition of a word in a sentence}}{\textit{total number of words in sentence}}$$

$$\textit{Inverse Document Frequency: } \frac{\textit{number of sentences}}{\textit{number of sentences containing words}}$$

$$\textit{Tf - Idf: Term Frequency * Inverse Document Frequency}$$

Subsequently, we moved on to the phase relating to the creation of the Classification predictive model, using the K-nearest neighbors (KNN) technique.

K-nearest neighbors (KNN)

KNN is a supervised learning algorithm, the purpose of which is to predict a new instance by knowing the data points which are separated into different classes (Altman, 1992). The k-nearest neighbors (k-NN) is an algorithm used in pattern recognition for classifying objects based on similarities in characteristics. When an instance is close to a data point, k-NN will consider them similar. Usually, the similarity is calculated thanks to the Euclidean distance between an instance that we want to predict and a datapoint. If the distance between the two points is small, the similarity between the data point and the instance to be predicted will be greater. In addition to the distance, the algorithm envisages setting a parameter k, chosen arbitrarily, which identifies the number of closest data points. The algorithm evaluates

the k minimum distances obtained. The class that obtains the greatest number of these distances is chosen as the prediction.

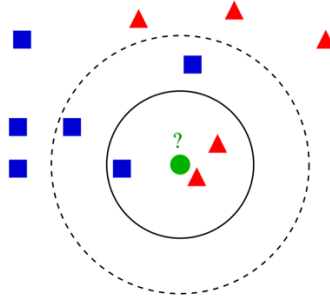


Figure 10: Example of K-nearest neighbors' classification. Image from <https://en.wikipedia.org/>.

The figure shows an example of classification using k -NN. The point under observation is the green dot. There are two classes: i) that of the red triangles; ii) that of the blue squares. If $k = 3$ (i.e., the 3 closest objects are considered), then the green dot is placed in the same class as the red triangles because there are 2 triangles and 1 square. If $k = 5$ then it is placed in the same class as the blue squares because there are 3 squares and 2 triangles.

To apply k -NN we divided the dataset into two parts: training set, and test set.

The training set was used for training the algorithm while the test set was used to evaluate the accuracy of the machine learning model.

3.2.3 Classification performance metrics

A variety of metrics exist to evaluate the performance of classifiers against trusted labels. The most common metric is the accuracy.

A confusion matrix, or contingency table, can be used to visualize true and false positive and negatives.

Confusion Matrix

In the context of artificial intelligence, the confusion matrix returns a representation of the statistical classification accuracy. The confusion matrix is simply a square matrix that reports the counts of the true positive, true negative, false positive and false negative predictions of a classifier, as shown in the following figure:

		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

Figure 11: Confusion Matrix. Image from <https://medium.com>.

Each column of the matrix represents the predicted values, while each row represents the real values. The element on row “i” and column “j” is the number of cases in which the classifier has classified the "true" class “i” as class “j”. Through this matrix it is observable if there is "confusion" in the classification of different classes.

Accuracy measures the percentage of exact predictions out of the total number of instances. Ranges from 0 (worst) to 1 (best).

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

The accuracy is calculated as the sum of correct predictions divided by the total number of predictions (Raschka, 2015).

Subsequently, the algorithm was implemented, which allows to make predictions on the sentiments experienced by students.

The software, once the execution is complete, provides for the printing of the precision indicator of the model to evaluate how valid the algorithm training is and how reliable the model is for predictions regarding future unclassified data.

The execution of the software ends with the automatic sending to the teacher of an e-mail indicating the number of opinions that have produced negative, positive, and neutral sentiments, together with the sending of an excel table that contains the comments of the students who they encountered negative sentiments.

3.3 Research Activity 3: Intelligent chatbot as a virtual assistant for university courses with a large number of students

3.3.1 Description of the activity

To implement the Chatbot, an on-line Physics course was used as support for the face-to-face Physics course for all of course degree belonged to School of Pharmacy. It was composed by video lectures recorded during the frontal lessons of the Physics course. Video lectures were integrated with supplementary teaching material to provide student an interactive learning environment. 30 videos lectures (covering all the syllabus of the course)

were recorded, for a total of 60 hours. Students had the possibility to see video lectures, teaching materials, starting from the day of the face-to-face lessons at any time of the day, until the end of the final exam. 155 students attended the course. The enrollment to on-line support was voluntary.

3.3.2 Software Development

This section describes the methodology used for data collection and processing to train the Machine Learning model implemented in the software. Figure 12 shows the flow relating to the operation and development of the software.

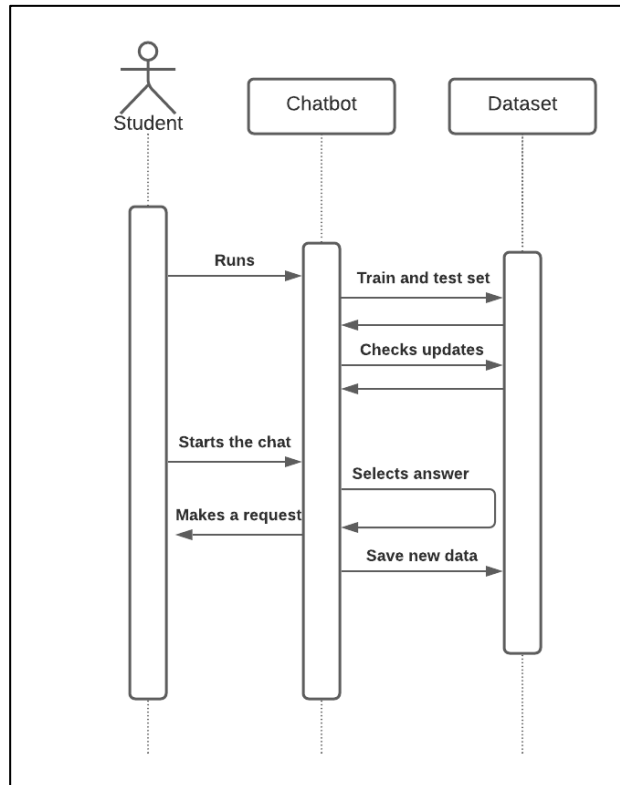


Figure 12: Sequence Diagram of user interaction with chatbot.

The software application was developed with the Python programming language using supervised Machine Learning algorithm to determine if an

input instruction satisfies a particular set of criteria that guarantees the generation of a response. To do this, a feedforward neural network has been implemented.

Feedforward neural network

Neural networks are non-linear structures of statistical data organized as modeling tools. They can be used to simulate complex relationships between inputs and outputs that other analytic functions cannot represent.

The multilayer perceptron (MLP) is the one of the most used artificial neural networks. The name is a slight misnomer; a multilayer perceptron is not a single perceptron with multiple layers, but rather multiple layers of artificial neurons that can be perceptron. The layers of the MLP form a directed, acyclic graph. Generally, each layer is fully connected to the subsequent layer; the output of each artificial neuron in a layer is an input to every artificial neuron in the next layer towards the output.

MLPs have three or more layers of artificial neurons. The input layer consists of simple input neurons. The input neurons are connected to at least one hidden layer of artificial neurons. The hidden layer represents latent variables; the input and output of this layer cannot be observed in the training data. Finally, the last hidden layer is connected to an output layer. The following diagram depicts the architecture of a multilayer perceptron with three layers. The neurons labeled +1 are bias neurons and are not depicted in most architecture diagrams.

Neural networks and machine learning algorithms require numerical values as input; so, we need a way to represent sentences with numbers.

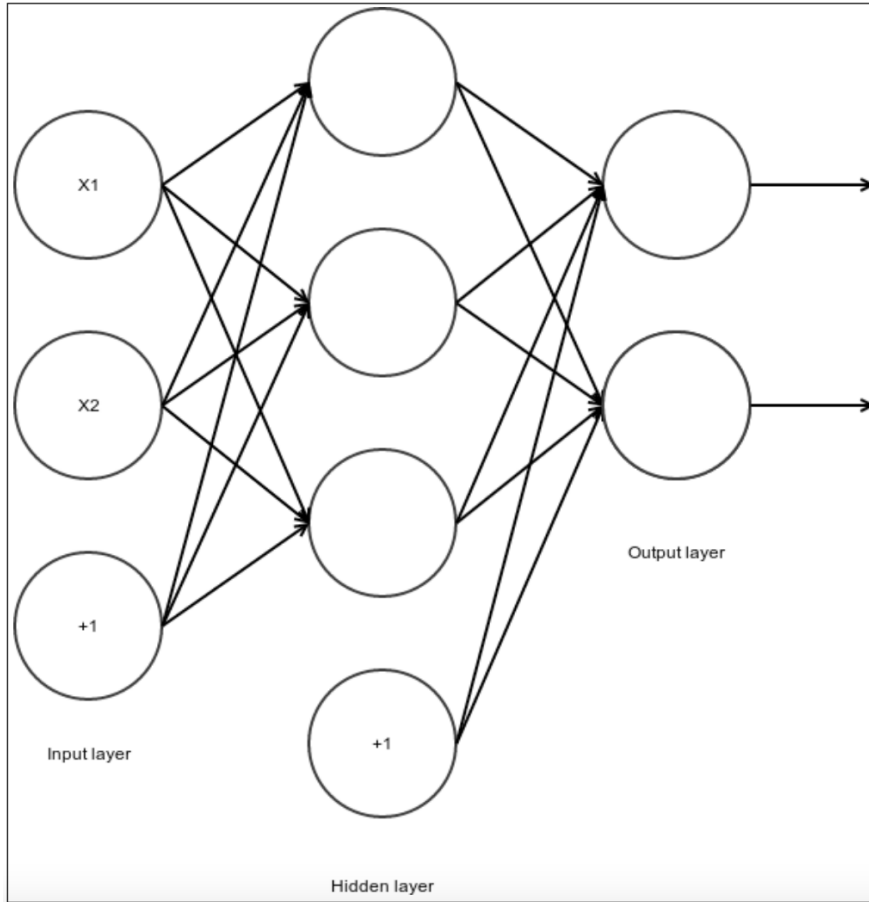


Figure 13: Example of Feedforward Neural Network. Image from <https://www.itread01.com>.

To do this you can use the “Bag of Words” (BoW) method.

BoW represents each sentence with a list of the length of the number of words in the model vocabulary. This representation uses a multiset, or bag, that encodes the words that appear in a text; the bag-of-words does not encode any of the text's syntax, ignores the order of words, and disregards all grammar. Bag-of-words can be thought of as an extension to one-hot encoding. It creates one feature for each word of interest in the text (Rashka et al., 2015).

To have a good model, every time a student enters an input question and receives an answer, the software saves the combinations of questions and answers in the dataset. The Chatbot will check if the matchings are in the dataset or not and if not, it will save them in it. In subsequent executions of the Chatbot, the algorithm will carry out the training phase again, also analyzing the new texts that have been saved. By having more examples of data to compare, the model will be better, and the software will be able to provide more accurate answers to future questions, thus increasing the efficiency and reliability of the Chatbot.

3.4 Research Activity 4: Automatic tutoring system for programming course

3.4.1 Description of the activity

This case study was carried out with a group of 151 first-year students of the degree course in Computer Science at the University of Camerino. The course was delivered in Italian. Participation in the online tutoring course was voluntary. The online tutoring course was available to students after the theoretical and laboratory face to face lectures, to have a basic preparation before taking the tutoring course. A professor, a tutor, and a Ph.D. student in Computer Science course at the University of Camerino prepared the teaching materials, the video tutorial and designed the online activities.

3.4.2 Course and software development

The project of the online tutoring course was preceded by an accurate identification of the topics that the last years students found difficulties.

The main goal of teaching programming course is to acquire skills to create computer software that solve real problems, developing computational skills. The course was divided into 6 modules, each of which consists of topics that allow to achieve the learning outcomes of the programming course. These permit to implement and create simple software to solve computational problems. In particular, the following topics were explained in each module:

1. primitive data types, cast, bitwise operators;
2. vectors and cycles;
3. control flows and logical operators;
4. classes and exceptions;
5. math and Util libraries;
6. inheritance and interfaces.

For each module, activities were developed to allow effective learning of students' knowledge and skills. Each module initially consists of a video tutorial, often used in e-learning courses to stimulate student engagement (Amendola D. et al., 2016). The video tutorial drives students step by step in writing the programming code aimed at solving programming exercises. Video tutorials last a maximum of 20 minutes, about 40 MB in size and consist of voice and screen recording. A Java online editor is displayed, and the step-by-step explanations show the code to solve exercises. Key information on the methods and solutions are used; theoretical references are included in the video, to provide the tools to solve similar tasks. To give the opportunity to develop computational knowledge, various activities and exercises are included where students with difficulties can test their skills. The activities to be delivered in the course are based on teaching models that found a fast and effective learning of computer skills and knowledge in

programming (Tan J. et al., 2014). These models consist of different exercises like multiple choice and open cloze exercise. The interactive tutor is another important tool for learning success which supports step by step the development of simple computer software (Figueiredo J. et al., 2020).

The following activities are included in each module:

1. multiple choice quiz;
2. text completion;
3. code evaluation;
4. exercise with solution;
5. programming exercise.

The “multiple choice quiz” consists of written java code and student must give the right answer selecting what output the code performs.

The “text completion” is composed by open cloze exercises, used for the acquisition of knowledge related to the Java syntax (e.g., logical operators, conditional operators, data types etc ...).

The “code evaluation” is a multiple-choice quiz that, analysing a written code, consists of more than one alternative that students must detect. In this exercise there is the code that has anomalies. Students must check what anomalies it presents by selecting the correct ones from a series of items. The system automatically gives feedback showing the theoretical explanation.

In the "exercise with solution" activity there is a task that describes the problem to perform and the solution, giving the complete source code. This exercise is useful for students because they can try to solve the task by themselves and later compare with the solution provided. In this way they can understand the correct way on how to solve the problem properly.

Like this activity, there is a “programming exercise” that students must perform autonomously. Students must produce the source code on the java editor and check if it’s correct.

The Java editor used in this course is an online compiler made by "Trinket" that is embedded into the Moodle course. It runs the java code online and checks its correct execution.

When the source code is correct, students check the quality of the syntax code. The students execute the implement code using the “interactive tutoring software”, that we especially developed for this online tutoring course to overcome Moodle limits (Moodle lacks a tool that supports these features). This software provides automatically immediate feedback on the quality of the code entered, analysing criteria like “keywords”, “use of variables”, “use of methods”, “correct use of classes” to satisfy the object-oriented paradigms.

There are several studies that highlight the effectiveness of an interactive tutor in teaching computer science: i)working with an interactive tutor who supports the making of programs is more effective than learning to program by doing the same exercise using only a compiler; ii)the use of automatic tutors requires less help from the teacher; iii) use of online tutors increases self-confidence in students; iv)immediate feedback from the tutor appears to be preferable to feedback given later in class (Gerdes A. et al., 2012). The interactive tutor application is developed in PHP and is installed on a LAMP server. It consists of an html input file, which consists of a textbox where students insert their own java code and a button for code verification and an output php file, which shows to student’s feedback related to the inserted code. Students receive, on output, feedback on mistakes and tips to improve the quality of the java syntax. The system consists of an accurate analysis

of the text that it takes as input. It compares the written text with different criteria that the code should satisfy, set by the teacher for a given exercise. These criteria can be: i) use of specific keywords (e.g., “final”, “static”, “public”), ii) length of the code, iii) good use of java classes, methods, libraries and functions. Thanks to these tips, students can understand their mistakes and edit their code until they get positive feedback, improving their code without the human intervention. In addition to the 6 modules, the course has an introductory video that explains: i) aims and structure of the course; ii) how to use the editor; iii) general intro to java topics. Finally, there is a questionnaire, consisting of 33 questions aiming to acquire interest in the study and to check students' perception and satisfaction.

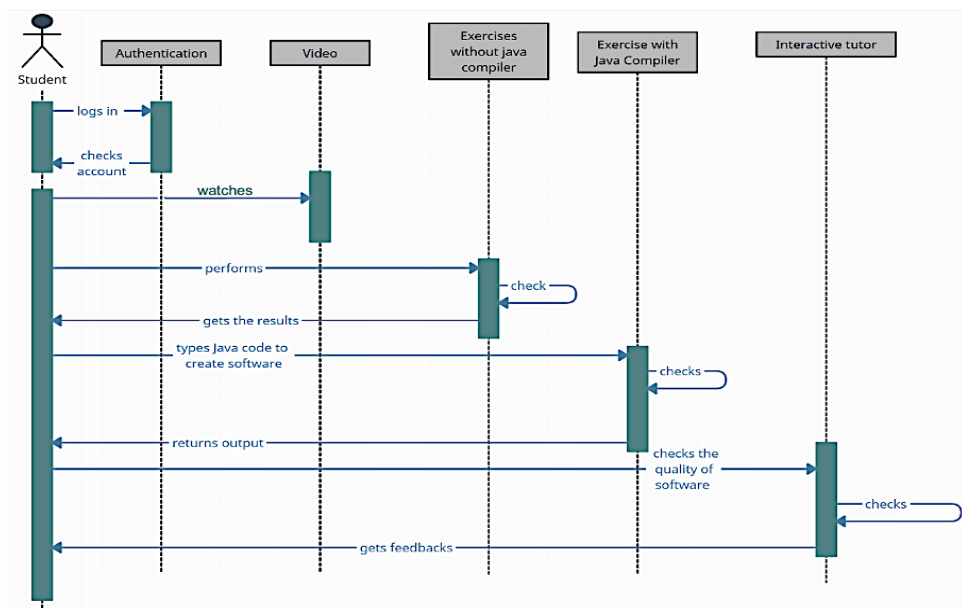


Figure 14: The flow of online activities and software execution.

3.4.3 Clustering

In this course a Machine Learning technique (Clustering) is applied to Moodle log data to create similar groups of students, to define the most influential behaviors and activities that allow a fast pass exam rate.

In way clustering returns important feedback for the teacher that can test the effectiveness of the tutoring e-learning course, checking the final performance for students in each cluster. The clustering software with K-Means algorithm was implemented through Python scikit learn library (Hackeling, 2014). The software creates groups of students that have similar behaviours and characteristics on the platform. Once obtained the clusters, we compared them with the percentages of students who passed the final exam before the end of the academic year 2019/2020; thus, determining the most influential behaviours and activities that allowed a fast pass exam rate.

Chapter 4. Results and Discussions

In this chapter the main results and corresponding discussions about the different activities are explained. The chapter is organized in different sections, one for each research activity, underlining strengths and weaknesses of each work.

4.1 Research Activity 1: Moodle plugin for creation of student heterogeneous groups in online university courses

4.1.1 Results of Clustering Algorithms

K-means

The K-means clustering algorithm was used to generate the groupings, created through a function of Python's scikit learn library (Hackeling, 2014). Based on the dataset provided, the algorithm applies the “elbow method” and returns, in output, the graph from which it is possible to extract the ideal number of Clusters to generate, equal to 3 (Figure 15).

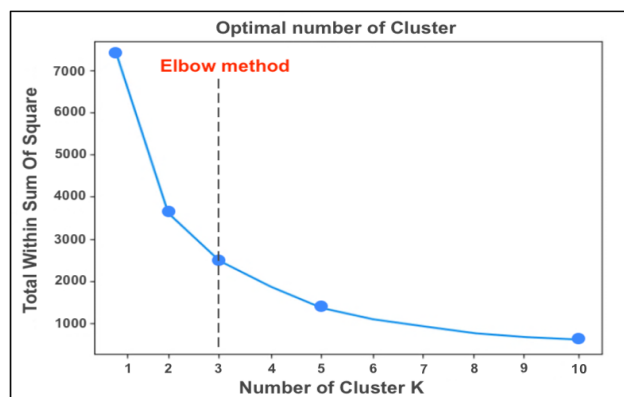


Figure 15: Elbow method graph.

This algorithm created 3 homogeneous clusters, characterized respectively by 10 (“cluster 0” in red), 29 (“cluster 1” in blue) and 16 (“cluster 2” in green) students (Figure 16).

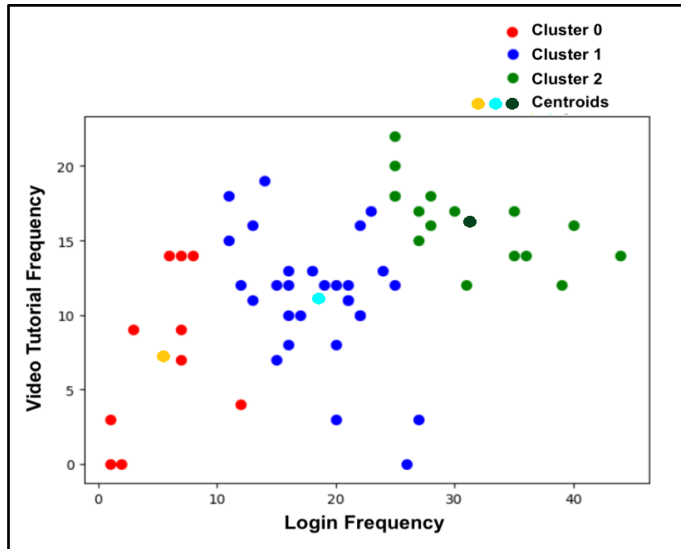


Figure 16: Representation of a two-dimensional projection (video tutorials viewing frequencies, login frequency) of the groupings that emerged from the clustering phase based on the different features selected.

In output, the graph represents the vectors associated with the different clusters on a two-dimensional plane of two of the 10 features used and the text containing the name of the grouping, followed by a list of numbers, in which each number identifies each student.

The results of K-Means are 3 clusters:

- Cluster 0: (4, 8, 12, 22, 25, 29, 30, 34, 43, 46),
- Cluster 1: (0, 1, 2, 3, 5, 6, 7, 10, 11, 14, 15, 21, 23, 24, 26, 27, 28, 31, 33, 35, 36, 38, 39, 40, 41, 45, 47, 51, 52),
- Cluster 2: (9, 13, 16, 17, 18, 19, 20, 32, 37, 42, 44, 48, 49, 50, 53, 54).

Mean-Shift

Mean-Shift was the second algorithm to be tested. The first step was to import the necessary libraries that Mean-Shift needs to perform the clustering process, then using Scikit-Learn's Mean-Shift class, a Mean-Shift model was created easily, and ran the algorithm using a bandwidth of 120. After running the information for each feature were analyzed, for example, the mean, standard deviation, minimum. Based on these values, the algorithm defined whether a student belonged to a specific cluster. In cluster 0 there were 20 students, in cluster 1, 11 students, in cluster 2, there were 7 students, in cluster 3, 9 students and in cluster 4 there were 8 students.

The results of the Mean-Shift were 5 clusters:

- Cluster 0: (1, 6, 7, 8, 9, 22, 23, 25, 29, 30, 33, 35, 36, 37, 41, 42, 43, 48, 49, 50),
- Cluster 1: (5, 10, 11, 12, 13, 14, 16, 26, 28, 31, 32),
- Cluster 2: (17, 24, 34, 38, 44, 52, 53),
- Cluster 3: (2, 3, 4, 20, 21, 40, 45, 51, 54),
- Cluster 4: (0, 15, 18, 19, 27, 39, 46, 47).

Agglomerative Clustering

After the Mean shift, Agglomerative Clustering was tested, using the scikit learn Agglomerative Clustering library. The algorithm generated a dendrogram to visualize the clusters in a better way and to find the optimal number of clusters. The dendrogram (Figure 17) showed the highest vertical distance that didn't intersect with any clusters was the middle blue one. Given that 2 vertical lines cross the threshold, the optimal number of clusters was 2. In this way two groups of students were created.

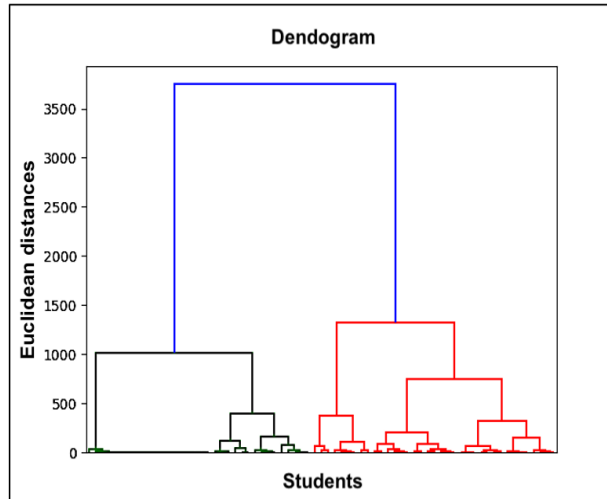


Figure 17: Dendrogram created from Agglomerative Clustering.

The algorithm continued the execution with the creation of an instance of Agglomerative Clustering using the Euclidean distance as the measure of distance between points and ward linkage to calculate the proximity of clusters.

The results of Agglomerative Clustering are 2 clusters:

- Cluster 0: (1, 6, 7, 8, 9, 17, 22, 23, 24, 25, 29, 30, 33, 34, 35, 36, 37, 38, 41, 42, 43, 44, 48, 49, 50, 52, 53),
- Cluster 1: (0, 2, 3, 4, 5, 10, 11, 12, 13, 14, 15, 16, 18, 19, 20, 21, 26, 27, 28, 31, 32, 39, 40, 45, 46, 47, 51, 54).

DBSCAN Clustering

DBSCAN clustering was another algorithm tested to create clusters of students. To use the algorithm, the scikit learn DBSCAN library was imported, setting by default the variable “epsilon” equal to 67 and minimal number of samples are equal to 3, which was the minimum number of points needed to create a cluster. The labels formed by the DBSCAN algorithm

and the number of clusters, that the algorithm created by ignoring the noise (using the variable “n_noise”), were stored.

The results of the DBSCAN were 5 clusters:

- Cluster 0: (1, 17, 24, 34, 38, 42, 44, 52, 53)
- Cluster 1: (0, 19, 39)
- Cluster 2: (6, 7, 8, 9, 22, 23, 25, 29, 30, 33, 35, 36, 37, 41, 43, 48, 49, 50)
- Cluster 3: (15, 18, 27, 46, 47)
- Cluster 4: (2, 3, 4, 5, 10, 11, 12, 13, 14, 16, 20, 21, 26, 28, 31, 32, 40, 45, 51, 54)

After the execution of the algorithm, no noise occurred because the number of students in 5 clusters was the same as the initial number of students.

Gaussian Mixture Model

Then, using the Gaussian Mixture class of scikit-learn, a Gaussian Mixture Model was created, and the algorithm was performed. After the convergence of the model, the weights, means, covariances and the clusters were printed. The results of the “Means” were ((2.49090909e+01 1.81818182e-01 9.09090909e-02 0.00000000e+00...)), while the “Covariances” were (((3.96399174e+03 2.01074380e+01 1.03677686e+01...))). Based on these results, 3 clusters were created.

The results of Gaussian Mixture are:

- Cluster 1: (0, 2, 3, 4, 5, 10, 12, 14, 15, 16, 17, 18, 19, 20, 21, 24, 26, 27, 28, 31, 32, 38, 39, 40, 42, 46, 47, 54),
- Cluster 2: (1, 6, 7, 8, 9, 22, 23, 25, 29, 30, 33, 35, 36, 37, 41, 43, 48, 49, 50),

- Cluster 3: (11, 13, 34, 44, 45, 51, 52, 53).

Self-Organizing Map (SOM)

To implement the SOM algorithm, SOMPY Python library was used. It was initialized as a 20-by-20 Self-Organizing Map and it used the default parameters of SOMPY, like the initialization and neighborhood method. In Figure 18 there is a view of the clusters using the hitmap .

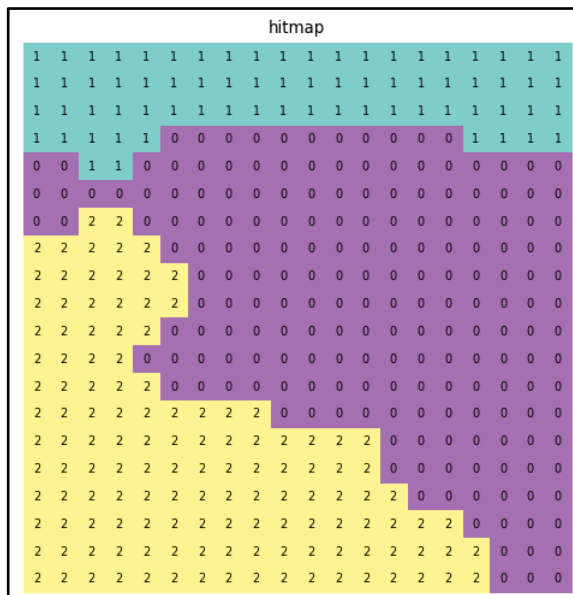


Figure 18: Presentation of the clusters generated by SOM algorithm.

The results of SOM were 3 cluster:

- Cluster 0: (1, 6, 7, 8, 9, 10, 14, 22, 23, 25, 29, 30, 33, 35, 36, 37, 41, 42, 43, 48, 49, 50),
- Cluster 1: (0, 2, 3, 4, 12, 13, 15, 24, 27, 28, 31, 32, 38, 47, 52, 54),
- Cluster 2: (5, 11, 16, 17, 18, 19, 20, 21, 26, 34, 39, 40, 44, 45, 46, 51, 53).

4.1.2 Comparison of Clustering Algorithms

Silhouette Analysis

After running the different clustering algorithms, the Silhouette analysis method was used to interpret and validate the consistency within the data clusters. Using the result of each algorithm (generated clusters) and the data used to train the model, the silhouette score was calculated. The results of this analysis were displayed in the bar chart below (see Figure 19 for details).

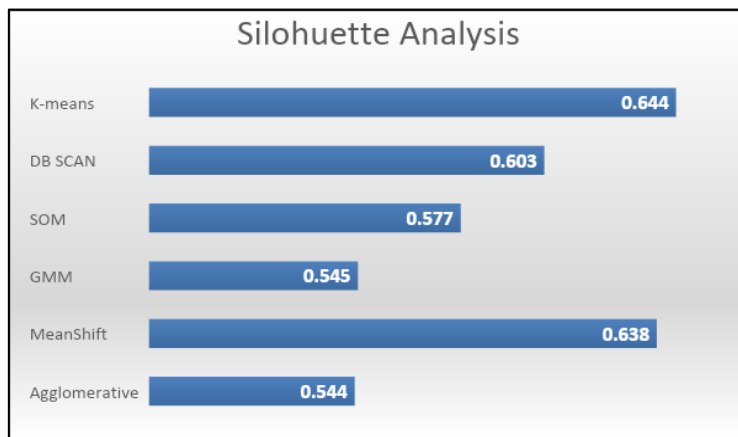


Figure 19: Silhouette score of the algorithms. Since the value is closer to 1, according to Silhouette's analysis, K-means was the best algorithm in comparison to the other algorithms. It was selected to implement the plugin to get optimal clusters.

The highest values achieved from two algorithms were: K-Means (a value of 0.644) and Mean Shift (with a value of 0.638); instead, the algorithms that have the lowest values are Agglomerative Clustering and Gaussian Mixture Models Clustering, with a value of 0.54.

So, based on the theory of silhouette analysis and the results, the best algorithm was K- Means because it had the highest values among other algorithms with the value of 0.644.

4.1.3 Selection of the Best Clustering Algorithm and Development of the Software

After choosing the best clustering algorithm, the software was developed, and the K- means algorithm was inserted into it.

The clusters obtained by the software were then analyzed to determine the behaviors on the platform, highlighting which are the most influential features within the groupings. In this way, it is possible to profile the students who belong to a certain grouping.

As can be seen from Table 1, “cluster 0” represents the least active students on the platform, with a low number of accesses (average = 5.4), low frequency of views related to video tutorials (average = 7.4), and a low number and frequency of views related to the experiments (average = 2.6). The low participation is also reflected in the delivery of the individual paper, where only 40% of the students delivered the assignment on the platform.

“Cluster 1” represents students who are inactive or not too active in the platform. These users had an average of the number of accesses equal to 18.4, average of the number of video tutorials viewed equal to 2.5, average of the frequency of views of video tutorials equal to 11.2, average of the number of experiments viewed equal to 4.1, average of the viewing frequency of video experiments equal to 6.7. In this case, participation allowed 90% of students to carry out and deliver the final exercise. “Cluster 2” represents highly active students on the platform. These users had an average of the number of visits of 31.2, average of the number of video tutorials viewed equal to 3, and an average frequency of views of 16.2. The average number of experiments seen is 4.3 (5 total video experiments) while the average frequency of views is 7. This level of participation allowed all students to deliver the final paper (100% of the students).

Features	Cluster 0	Cluster 1	Cluster 2
Behaviour	Not very active students	Moderately active students	Very active students
N° students	10	29	16
N° access to the platform (average)	5,4	18,4	31,2
N° video tutorials viewed(average)	1,8	2,5	3
Total viewing frequency of video tutorials (average)	7,4	11,2	16,2
N° video viewed(average)	2	4,1	4,3
Total display frequency, of video experiments (average)	2,6	6,7	7
% Students who completed the assignment	40%	90%	100%

Table 1: Differences between clusters and details of the analyzed features.

To verify if the behavior carried out by students on the e-learning platform had feed- back in terms of performance, we compared each cluster with the average of the marks for the individual work done by the students belonging to the same cluster. The results obtained are very interesting. As reported in Figure 20, the level of participation in the platform of the students reflects the final grade. The most active students (“cluster 2”), in fact, received a higher score than the others with an average grade of 82.5. Students of “cluster 1”, with an average activity, had a lower rating with an average of 72.04, while students of “cluster 0” had an average grade, equal to 53. This strong correlation highlighted by the Pearson index equal to 0.986 allowed us to confirm the differences between the students belonging to the different clusters, not only in terms of behavior but also in terms of performance.

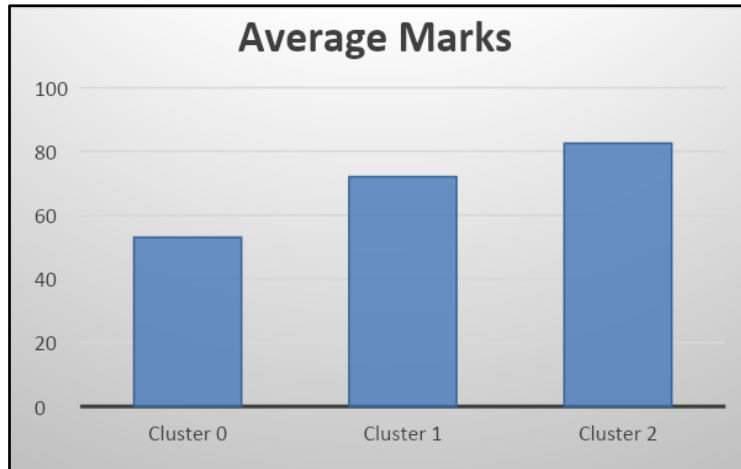


Figure 20: Student marks averages in the different clusters.

Once obtained the clusters, the software returned the creation of heterogeneous groups. The number of students set for each group was 5. The software applies the algorithm for creating heterogeneous groups, initially dividing the total number of students by 5, calculating the number of heterogeneous groups to create. Then the number of students belonging to each group was calculated, by dividing the number of students in each cluster by the number of groups. In figure 21 after the setting of the algorithm, the software created 11 groups and automatically distributed the students in groups. In this way, in each group there was at least one student belonging to different clusters. It creates heterogeneous groups based on the students' behavior. Finally, the software automatically sent an e-mail to the teacher, with the list of heterogeneous groups, indicating the group name (e.g., group 0, group 1...) and for each group the list of 5 numbers, the IDs that identify the students. In this way, the teacher was able to select students based on their ID, insert them into relevant groups within Moodle and start the collaborative part in the course.

```
cluster 0:
[4, 8, 12, 22, 25, 29, 30, 34, 43, 46]

cluster 1:
[0, 1, 2, 3, 5, 6, 7, 10, 11, 14, 15, 21, 23, 24, 26, 27, 28, 31, 33, 35, 36, 38, 39,
40, 41, 45, 47, 51, 52]

cluster 2:
[9, 13, 16, 17, 18, 19, 20, 32, 37, 42, 44, 48, 49, 50, 53, 54]

Group 0[4, 9, 0, 1, 2]
Group 1[8, 13, 3, 5, 6]
Group 2[12, 16, 7, 10, 11]
Group 3[22, 17, 14, 15, 21]
Group 4[25, 18, 23, 24, 26]
Group 5[29, 19, 27, 28, 31]
Group 6[30, 20, 33, 35, 36]
Group 7[34, 32, 38, 39, 40]
Group 8[43, 37, 41, 45, 47]
Group 9[46, 42, 44, 51, 52]
Group 10[48, 49, 50, 53, 54]
```

Figure 21: Clusters and heterogeneous groups obtained from the execution of the software.

4.1.4 Moodle Plugin

After the testing, the software was transformed into a new Moodle plugin to help teachers creating heterogeneous groups. It allows the loading of the dataset, which is the combination of two report files generated by the e-learning platform, and the creation of heterogeneous groups.

Initially, the teacher can enter the e-mail address, which is used by the plugin to send the list of heterogeneous groups by email. The plugin then presents a button that allows the teacher to upload the excel files (Moodle Reports) downloaded from Moodle.

The files contain the features concerning the time, the behavior and the activities carried out by the students on the e-learning platform that represent the dataset that the machine learning algorithm used for the creation of heterogeneous groups. This algorithm is performed thanks to the “Generate” button, which permits uploading files and performing clustering (Figure 22).

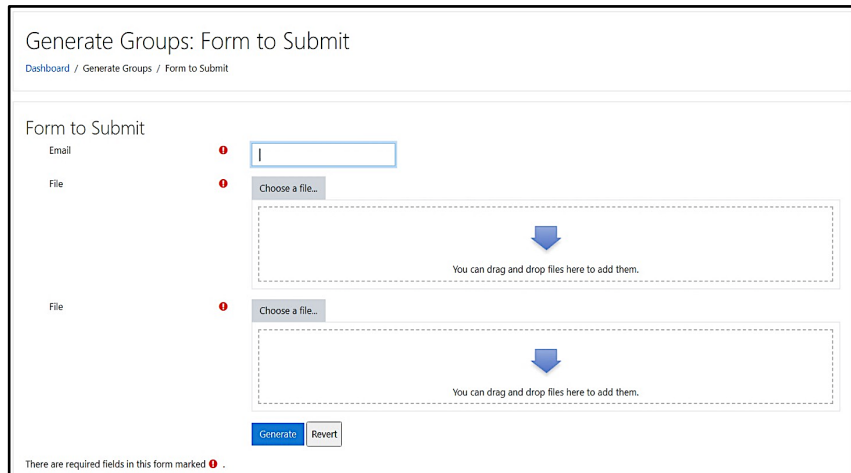


Figure 22: Moodle plugin used to generate heterogeneous groups.

The Moodle plugin was developed using PHP as a programming language and Python for the part of machine learning. After uploading the files and clicking the submit button by the user, the plug-in uploads these files to a specific folder which is called "upload" (Figure 23).

Name	Type	Size
db	File folder	
lang	File folder	
upload	File folder	
styles	Cascading Style S...	1 KB
block_machinelearning	PHP File	4 KB
edit_form	PHP File	1 KB
settings	PHP File	1 KB
version	PHP File	2 KB

Figure 23: A view of the plugin directory in the Moodle server. In this directory the plugin saves the files that the user uploads in the plugin, to perform the creation of the heterogeneous groups.

Then, the plugin gets these excel files, merges them and generates an excel file called “merge.excel”, which is the input dataset file. This file is placed in the same directory where the excel files are uploaded from the user. Then the machine learning algorithm gets the dataset file and performs the

clustering process. The result of the algorithm, which is a string with the groups created, are displayed to the teacher, and sent to the teacher email (Figure 24).

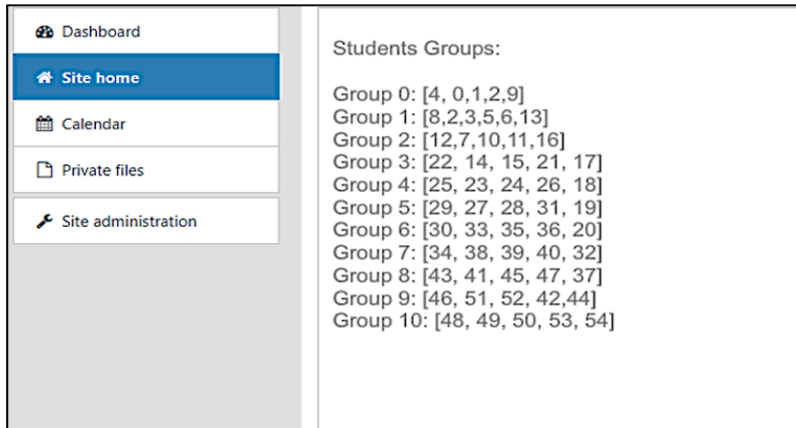


Figure 24: List of heterogeneous groups in the Moodle Plugin. These groups are the result after the execution of the plug-in, provided by selecting students from different clusters thanks to the developed algorithm. Each number within the groups represents an "id" linked to a student.

4.2 Research Activity 2: Tool for classifying students' sentiments in an e-learning course

4.2.1 Dataset

For the classification of students' sentiments, a new software was created using classification technique. It allowed to predict the sentiments of the students by analyzing the text they wrote on the Moodle feedback activity (questionnaire).

To test the software, we had to create the dataset, the file to be provided as input to the software, by inserting the comments that the students entered in the questionnaire. The answers available were 132, out of a total of 66 students who completed the questionnaire. In this work, two types of open

answers of the satisfaction questionnaire were analyzed, relating to the aspects of the course and the platform that were liked or not by the students.

Questions	Examples of the students' comments
What did you like most about the course?	<ul style="list-style-type: none"> - The multiple-choice tests for self-evaluation were very useful for testing myself in a way like the exam. - Easy to use very helpful for the exam. - It is well organized, and materials are clear and not difficult to understand. - It is easy to find learning materials. - The chemistry course materials available here are comprehensive. - I like exercises for practicing and having a general overview of the test.
What did you like least about the course?	<ul style="list-style-type: none"> - There are few exercises. - the absence of material for the second partial. - For me it's a little bit chaotic. - boring, it doesn't cover all the program s arguments. - I couldn't find all the materials. - lack of some topic's exercises.

Table 2: Examples of students' comments.

The questions were:

- What did you like most about the course?
- What did you like least about the course?

The dataset was composed by different field such as *numeric id* that identifies the student, the *text* that the students wrote in the questionnaire and the *label* that indicates whether the text is positive, negative, or neutral. This data was organized into a single excel file (dataset) and it matched the students with their opinions with the polarity of sentiment label.

To match the polarity to the comments, the python Textblob library was used. This library used Natural Language Processing (NLP) for processing

textual data and returned a numeric value in the range between -1 and 1, indicating polarity. The value -1 represents that the sentiment is negative, the value 0 that the sentiment is neutral, while if it is 1, the sentiment is positive.

id	text	Sentiment	Polarity
1	The chemistry course materials available here are comprehensive	Positive	0.4
2	The multiple choice tests for self-evaluation were very useful for testing myself.	Positive	0.13
3	It's convinced	Neutral	0.0
4	I dont think there is something about not to like	Neutral	0.0
5	I don't like the Submicroscopic views	Negative	-0.3
6	I like least about the platform the user interface	Negative	-0.3
7	It is well organized and materials are clear and not difficult to understand	Positive	0.17500000000000002
8	You can check at home if you're studying in the right way by test yourself with exercises and self evaluation	Positive	0.2857142857142857
9	I like exercises for practicing and having a general overview of the test	Positive	0.05000000000000002
10	the absence of material for the second partial	Negative	-0.103125
11	Nothing.	Neutral	0.0
12	I like the exercises we had available to get ready for the 1st partial	Positive	0.16666666666666667
13	Easy to use very helpful for the exam	Positive	0.31666666666666665
14	it helped me understand important things	Positive	0.4
15	Possibility to study from home.	Neutral	0.0
16	it helps to keep notions organized ordered it gives good help to be better ready for the exercises of the exam	Positive	0.46666666666666666
17	I like least about the Video tutorial on how to solve exercises	Negative	-0.3
18	I don't like the organization of the website	Negative	-0.3
19	It is easy to find learning materials.	Positive	0.43333333333333335
20	There are few exercises	Negative	-0.2
21	Hard to use from other devices than computer	Negative	-0.20833333333333334
22	For me its a little bit caotic	Negative	-0.1875

Figure 25: Screenshot of the dataset with sentiment label and polarity.

If the numerical value is close to the limit of the range, it means that it is strongly positive (if it is close to +1) or strongly negative (if it is close to -1). The polarity value was used only to detect the sentiment label, that represent the value to predict with the classification model. After that the polarity column was deleted from the dataset.

4.2.2 Data cleaning

Once obtained the Dataset, the classification model was implemented. Comments contained many punctuation marks. It was important to clean comments before used them for training the machine learning model.

I removed all the special characters from the comments, thanks to the python expression: `re.sub (r'\\W', '', str(features[sentence]))`.

I removed all the single characters left as a result of removing the special character using the regular expression `re.sub(r'\s+[a-zA-Z]\s+', ' ', processed_feature)`.

Replacing all single characters with space, multiple spaces were created. Therefore, I replaced multiple spaces with single spaces using `re.sub(r'\s+', ' ', processed_feature, flags=re.I)`.

Finally, the text was converted into lowercase using the `lower()` function.

4.2.3 Predictive model

Statistical algorithms use mathematics to train machine learning models. However, mathematics only works with numbers. To use classification, it was necessary to use the TF-IDF algorithm to convert the text into numbers to extract features. The TF-IDF algorithm was used, a function used in information retrieval to measure the importance of a term compared to a document or a collection of documents.

A python scikit-learn library was used to implement the algorithm. This library contains the `TfidfVectorizer` class that allows the conversion of the text features into TF-IDF feature vectors.

Standard setting was used to implement it:

- the value of `max_features` was 2500, which means that it only used the 2500 most frequently occurring words to create feature vector. Words that occur less frequently were not very useful for classification,
- `max_df` specifies that it only used those words that occur in a maximum of 80% of the comments. Words that occurred in all documents were too common and were not very useful for classification,
- `min-df` was set to 7 which showed that include words that occur in at least 7 comments.

As the last step before we trained our algorithms, we needed to divide our dataset into training and testing sets. The training set was used to train the algorithm while the test set was used to evaluate the performance of the machine learning model. I used the `train_test_split` class from the `sklearn.model_selection` to split data into training and testing set.

The dataset was split in 80% for training and 20% for testing.

Then the classification algorithm was implemented using the KNN technique using the Scikit-Learn's KNN implementation to train and test the k-nearest neighbor classifier on the dataset.

Standard setting was used to implement it:

- `n_neighbors=5`, represent number of neighbors to use by default for k-neighbors queries;
- `weights='uniform'`, All points in each neighborhood are weighted equally;
- `algorithm='auto'`, attempt to decide the most appropriate algorithm based on the values passed to fit method;
- `metric='minkowski'`, the distance metric to use for the tree. The default metric is minkowski, and with $p=2$ is equivalent to the standard Euclidean metric.

Once the algorithm has been run, the software processed the data and returned the results.

Out of 132 answers given by students, 86 were positive, 28 were neutral and 18 were negative.

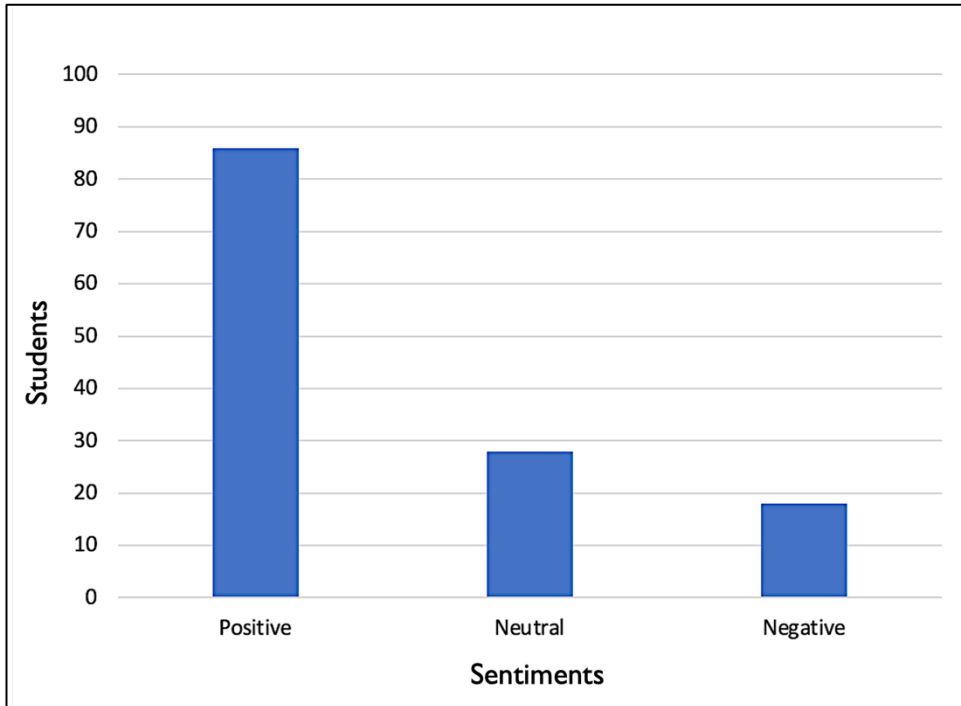


Figure 26: Bar plot of the students' sentiments.

The results obtained and the list of opinions that produced negative sentiments are sent automatically to the teacher via email.

This step ensured the teacher to have an overview of the results obtained and all the negative aspects highlighted by the students to be able to improve the aspects of the course that received negative feedback.

To evaluate the performance of the predictive model, the classification metrics confusion matrix was used, analyzing the `accuracy_score` using `sklearn.metrics`. The model achieved an accuracy of 80.30%.

This allows the teacher to be able to use the model to make new predictions on unclassified answers, extracting the feelings of the students with a reliable accuracy.

4.3 Research Activity 3: Intelligent chatbot as a virtual assistant for university courses with a large number of students

4.3.1 Dataset

To create the initial dataset, most frequently asked questions and students' answers collected over the years were selected. In particular, the questions relating different aspects of Physics course were used, as can be seen in Table 3.

Category	Description
Exams	Questions related to the structure and modalities of the exams.
Reserved Area	Questions for information on how to access the reserved area of the University for booking exams.
Teaching Material	Questions related to textbooks, online teaching materials, recorded video lessons, sample exams and questionnaires.
Time and Date	Questions related to dates and times of lessons and exams.
Contacts	Questions about ways to contact teachers and information about how to receive students.
E-learning	Questions related to the university e-learning platform

Table 3: Categories for the answers.

The Dataset consists of sets of examples of questions that can be requested by users in the chat.

These questions were associated with a group of related answers. A tag declared in the file organizes messages into different groups based on the topic.

Tag (Class)	Questions	Answer
Greeting	“Hi”, “Good Morning” ...	“Hi, can I help you?”
Schedule	“Where can I find the lectures timetable?”, “When will the next lesson be?” ...	“You can find the schedule at this link: https://www.unicam.it/studente/OrariLezioni ”
E-learning	“Where can I find the online courses?”, “What is the e-learning web address?” ...	“You can find the online courses at this link: https://elearning.unicam.it/ ”
Teacher info	“Where can I find the information of the teachers?” “Where can I find the teacher email?” ...	“You can find all info of the teachers at this link: https://docenti.unicam.it/ ”

Table 4: Dataset composed by question and answer grouped by Class (target).

Then the Chatbot tool was implemented, choosing the classification technique because it allows to predict the correct answer based on the text entered by the student. Indeed, the software provides the answer based on the input received. To do this, it matches the input text to one of the questions included in the dataset and replies to the student’s request selecting the answer related to the question from the dataset.

4.3.2 Predictive Model

The software implementation includes the training and the test set for the evaluation of the predicting model, allowing the classification model to predict the correct answer from the students' questions.

I used a model from python Keras library that is one of the simplest neural networks, a multi-layer perceptron.

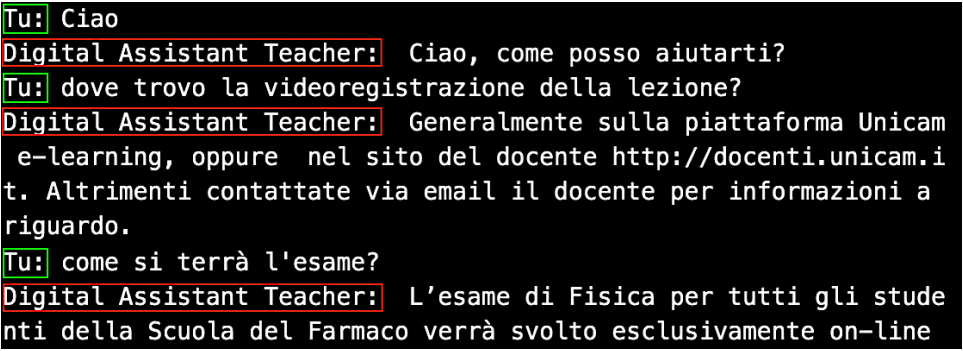
This is a standard network that consists in 3 layers, the first one with 128 neurons, the second one 64 neurons, and the third one with the number of intents as the number of neurons.

Neural networks and machine learning algorithms require numerical values as input; so we needed a way to represent sentences with numbers. To do this the “Bag of Words” (BoW) method was used.

BoW represents each sentence with an array of the length of the number of words in the pattern vocabulary.

Each position in the list represents a vocabulary word. If the position in the list is a 1, it means that the word of the sentence exists in the vocabulary, if it is a 0 the word is not present.

In addition, to increase the accuracy of the model, each time that a student enters a new input question, and the model returns an answer from the dataset, the software saves the combinations of questions and answers in the dataset.



```
Tu: Ciao
Digital Assistant Teacher: Ciao, come posso aiutarti?
Tu: dove trovo la videoregistrazione della lezione?
Digital Assistant Teacher: Generalmente sulla piattaforma Unicam
e-learning, oppure nel sito del docente http://docenti.unicam.i
t. Altrimenti contattate via email il docente per informazioni a
riguardo.
Tu: come si terrà l'esame?
Digital Assistant Teacher: L'esame di Fisica per tutti gli stude
nti della Scuola del Farmaco verrà svolto esclusivamente on-line
```

Figure 27: Example of Chatbot execution.

In future Chatbot executions, the algorithm will perform the training phase again, also analyzing the new saved texts. With more examples to compare, the accuracy of the model will be better, and the software will be able to provide more accurate answers to future questions, increasing the efficiency and reliability of the Chatbot.

The software is still being tested and no experimentation with students has been performed. Currently the software runs in the Python editor, but once the testing phase will be complete, and verified that the Chatbot answers the questions accurately, it will be included in a Moodle plugin and let the students try it out. After that the effectiveness and accuracy of the predictive model will then be analyzed and evaluated.

4.4 Research Activity 4: Automatic tutoring system for programming course

4.4.1 Use of the online resources

How did the students use the online resources?

The data relating to students enrolled in the first year of the Computer Science course of the academic year 2019/2020 were analysed.

Table 5 summarizes the level of general interaction that the students had with the materials within the different modules. For each user, multiple logs occurred for the same activity.

Access to the activities of each module was unrestricted and therefore students didn't need to follow an order for viewing the materials. Therefore, students viewed only the activities in which they had lot of difficulties or the activities useful for passing the exam.

Materials	Logs
Module 1 – Primitive data types, cast, bitwise operators	4672
Module 2 – Vectors and cycles	3708
Module 3 – Flow control and logical operators	1891
Module 4 - Classes and exceptions	2042
Module 5 – Math and Util libraries	1708
Module 6 – Inheritance and interfaces	1433

Table 5: Number of student logs for materials in all Modules.

Table 6 shows the types of activities most used in the course. From the number of accesses, the practical activities, which return an evaluation or feedback, had more success than the theoretical activities (such as videos). This happened because the students enrolled in the tutoring course needed to test their knowledge through exercises, compared to the theory carried out on face-to-face lessons. All exercises were delivered on the online course at the same time and students were able to perform the exercises with an unlimited number of attempts. The most used activity was the "Multiple choice quiz". It was an important tool for students to fix their knowledge. Other activities widely used were "Text Completion" and "Code Evaluation", which return an immediate evaluation based on the option selected or word entered. The possibility of having immediate feedback allowed students to check if they understood the concepts, or to identify

mistakes with the aim of improving their programming knowledge. The "programming exercise" activity was also widely used, with 1331 accesses, which was preferred to "Exercise with solution". This reinforced the importance of the interactive tutor to learn programming and develop computational knowledge.

Activities	Logs
Video	987
Multiple choice quiz	8391
Text completion	2057
Code evaluation	2061
Exercise with solution	639
Programming exercises	1331

Table 6: Most used activities in the online course.

Thanks to the online tutor with immediate feedback, the student is driven in writing the code with a good quality. This allowed students to be more effective in learning programming than carrying out this activity by themselves or using only the compiler; it also improved their self-confidence. (Figueiredo J. et al., 2018).

4.4.2 Effectiveness of the online tutoring course

Has the massive use of the E-learning platform helped students to pass the exam within the academic year?

To determine the effectiveness of the online tutoring course, a quantitative analysis on the dataset for the first-year students of the Computer Science Degree Course of the academic year 2019/2020 was performed. Initially Clustering technique was used to determine groups of students who had similar characteristics relating to the use of the course activities. The second step was the correlation of each cluster with the performance obtained by students belonging to the same cluster, to determine which behaviours influenced passing the exam. Only exams passed within the 2019/2020 academic year were analysed. The clusters were examined to determine the different behaviours on the platform, highlighting which features are important. In this way it was possible to profile the students who belong to the different clusters. Cluster 0 (26 users) represented students with high activity in the course, Cluster 1 (40 users) represented students with average activity and Cluster 2 (85 users) reflected inactive students. This was also confirmed by the analysis of very influential features in the creation of clusters, using the K-means clustering algorithm.

The most important features were number of quiz attempts, number of views of the example performed and number of exercises to be performed with the compiler and interactive tutor. The feature values within the clusters were represented with the average between the feature values of all students belonging to the respective clusters.

Cluster 2, that represented the inactive students, showed few attempts among all the quizzes of the course (7.37), few views of the types of "exercises with solution" (0.42 out of 6) and a low number of different exercises carried out by java compiler and interactive tutor (0.81 out of 12). This little interaction during the course was also confirmed by the number of accesses of the students belonging to the cluster (3.42) and by the number

of video views (1.8). Students belonging to cluster 1 reflected average values in all activities. In fact, Cluster 1, compared to Cluster 2, found a significant difference such as the number of quiz attempts (80.52), number of views of exercises with solutions (2.52 out of 6) and number of different exercises carried out with java compiler and interactive tutor (4.62 out of 12). The number of accesses (16.12) confirmed the greater activity of these students and the video views (12.02). Cluster 0, that represented active students, showed a high number of quiz attempts (130.26), lot of views of exercises performed (4.84 out of 6) and an high number of exercises performed with java compiler and tutor (10, 23 of 12).

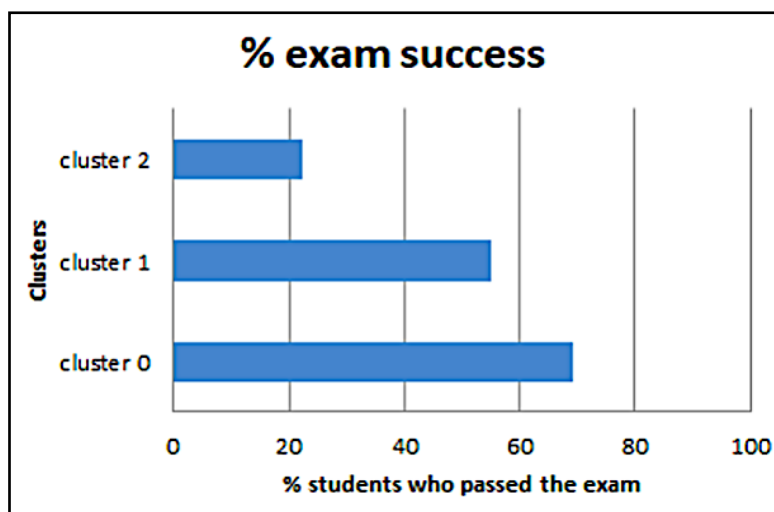


Figure 28 – Number of students per cluster who passed the final exam within the academic year 2019/2020.

To check if the behaviour carried out by students on the e-learning platform had benefits in terms of performance the percentage of students who passed the final exam before the end of the academic year 2019/2020 from each cluster was extracted.

As showed from Figure 28, the students of cluster 0, with high activity on the platform, achieved a higher success rate than the other clusters, equal to 69%. A good success rate returned also by the students of cluster 1 who, with average activity, passed the exam with a percentage of 55%. Cluster 2 had a very low percentage of students who passed the final exam, equal to 22%. It confirmed the tutoring course gives benefits for the students.

4.4.3 Students' perception

How did students perceive the effect of online resources in acquiring skills and passing the exam?

In this part we reported the results of the data returned by the questionnaire proposed to first year students at the end of the last exam session of the academic year 2019/2020. The goal was to analyse the perception of students on the use of the online tutoring course, and especially if this course influenced the improvement of computational knowledge and skills in addition to passing the final exam.

Most of the students believe that the use of the online tutoring course is effective for passing the exam. Figure 29 shows that 76% of students believe in the usefulness of the tools and exercises, to increase the chances of success, while only 23% think that they had little or no impact. This last part of students was represented by those who had less activity online. They, as shown by the Clusters analysis, didn't easily pass the exam.

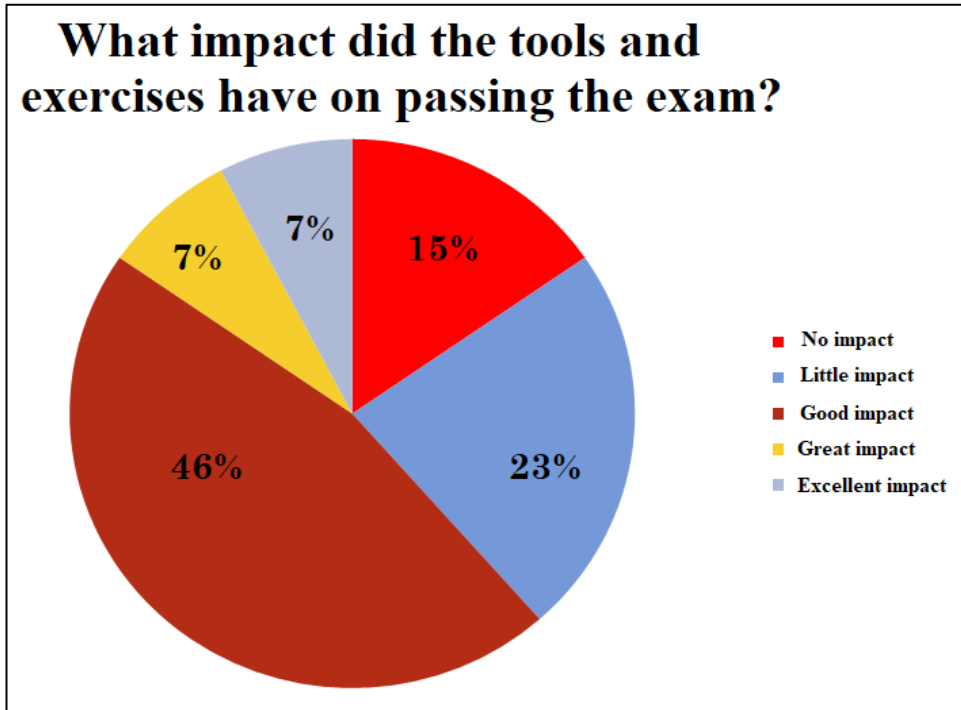


Figure 29 –Impact that the tools and exercises had on passing the exam within the academic year 2019/2020.

The usefulness of the tutoring course is also confirmed in the improvement of computational skills and knowledge. 80% of students claim that the tools and exercises used in the course improved their computational skills, while 20% believe that they had little influence on learning skills. Regarding the improvement of the knowledge, 85% of students believe that the tools improved them, compared to 15% of students who think that they had little impact for the knowledge enhancement. 95% of students consider all online activities (Text completion, Code Evaluation, Exercise with solution, Programming exercise) useful as a teaching support tool.

The video tutorials were appreciated as support tools, with only positive judgments (50% = good, 28% = very good, 22% = excellent). This result is confirmed by the students that consider video tutorials positively for the

improvement of knowledge (89%) and skills (83%). However, only 57% consider the video tutorials useful for passing the exam, while 43% consider them not very useful. This highlights the need for students to have tools that allow them to practice computation, compared to theoretical notions.

Chapter 5. Conclusions and Future Perspectives

The main achievement of this Ph.D. thesis was to improve the quality and the effectiveness of on-line teaching in scientific degree courses delivered by an e-learning platform based on Moodle. New tools based on Artificial Intelligence were developed to help the professors to create successful e-learning courses to support the students at improving their learning outcomes. In this way, the potentialities of Machine Learning for education are implemented and tested. Indeed, Machine Learning can help the students to improve learning by analysing their behaviour and performance and suggesting the best way to achieve learning outcomes.

The developed tools overcome the limitations of Moodle activities, applying machine learning techniques by analysing students' data extracted by Moodle analytics. Thanks to these applications, it was possible to increase the success of the teaching experience in the on-line activities where it was difficult to detect students' behaviour and check their emotions and engagement.

These tools provided an easy way for the teacher to gain insight and to understand properly the information contained in the data. In this way, teachers without specific capacities in computer science and statistics can easily extract students' information or using machine learning algorithm to optimize their teaching activities.

The research method adopted in our analysis has been based on a quantitative approach derived by the state of the art of the artificial intelligence applied to the teaching of scientific topics.

In this thesis, I fulfilled the objectives by designing, implementing and experimenting with the students at the University of Camerino different machine learning applications to e-learning courses:

1. Moodle plugin for creation of heterogeneous groups of students in online university courses;
2. Software tool for classifying students' sentiments in an e-learning course;
3. Intelligent chatbot as a virtual assistant for university courses with a large number of students;
4. Automatic tutoring system for programming course.

The summary of the main outcomes of the thesis' work organized in four research activities is the following.

In the Research Activity 1, I designed and created a new algorithm and corresponding software aimed at helping teachers in defining groups of students for collaborative activities. Machine Learning techniques demonstrated how this help improved the creation of heterogeneous groups of students. The ability to test the efficiency and performance of the clustering algorithms was important in choosing the best one, to better analyze the characteristics for each cluster by identified different student profiles accurately. The comparison between the clusters obtained and the average of the final students' grade in each cluster confirmed the differences between the clusters also in terms of performance. The most active students ("cluster 2") received a higher score than the others with an average grade of 82.5. Students of "cluster 1", with an average activity, had a lower rating with average of 72.04, while the students of "cluster 0" with low activity in the platform had an average grade equal to 53. By applying these techniques

to the collected data and verifying the efficiency of the techniques, it was possible to guarantee and maximize the heterogeneity of the students within each group. This increases the chances of success for all students in group works within the online course. The Moodle log data has been taken as a feature so that the software realized in this work can be reused by other teachers and tutors who carry out online teaching activities on Moodle. This is a standard software, not limited only to the Moodle environment, but it is easily adaptable to any other LMS (Learning Management System), by editing the dataset. At the end of this experiment with the Physics Laboratory online course, it could be interesting to test if the use of the heterogeneous groups improved the students' perceptions regarding the peers' feedbacks. In addition, another goal could be to analyze the final grades of the students after peer review to check if there will be a uniform improvement compared to the past.

In the Research Activity 2, I developed a software aimed at classifying students' sentiments after the use of a course in blended learning mode.

Classification techniques allowed to create a predictive model, with good accuracy, to classify new unprocessed texts, without the need to train again the dataset with Machine Learning techniques.

For the teacher, the possibility of identifying comments labeled as negative can be useful in analyzing any problems on the course and trying to improve some aspects that can help students in the use of the course and in the motivational aspect.

The predictive model can be used after the edit of the course. In this way, the teacher can check whether this change allowed an improvement in sentiment in the students.

The research project will continue with the creation of an advanced Plugin for Moodle that will allow the teacher to create the predictive model in a simple way, automatically, selecting the text to be analyzed from the Moodle activities and with the possibility of viewing the results directly on the platform.

Thanks to the detection and management of information on students' sentiments, teachers can also contribute to know the needs of students and act by modifying the materials properly or driving them in personalized paths.

In the Research Activity 3, a software was developed aimed at creating an intelligent Chatbot that allows individual support to the student, automatically answering in real time the questions and doubts of students enrolled in a university course. The implementation of Machine Learning algorithms, of classification techniques, allowed to generate the answers in an intelligent way, thanks to the training and test set phase that permitted the software to select the most suitable answer, based on the accumulated experience following the training phase.

The possibility to save the combinations of questions and answers generated by the interaction of the user and the software, allowed the classification model to increase its accuracy and thus reduce the error. To optimize operation, an excellent strategy could be to test the software often, checking that the Chatbot responds adequately to new requests, verifying its effectiveness and, eventually, modifying properly the dataset. For the teacher, making this tool available to students could be very useful as it would help in managing contacts. Indeed, the Chatbot may already provide an answer to most of the users' needs, and only responds to the most

particular requests that the software cannot answer. For the student it could also be a very efficient service because it would allow him to obtain information in real time.

In the Research Activity 4, an online Automatic Java Programming tutoring course was created, which allowed students struggling with the material to have an effective online tool to improve their computational skills and knowledge.

Tutoring can be used to provide students the right approach to overcome the weakness and critical phases of their studies.

This is also possible thanks to the possibility of using video tutorials and carrying out specific programming exercises.

Due to the lack of a tool to check the syntax quality of the code written by the students, I decided to implement a new software to achieve this goal. I therefore created a new tool (Interactive Tutor) which performed a syntax analysis of the written code and, like a tutor, automatically provided feedback and tips to improve quality. This tool is very useful because it doesn't require teacher intervention and speeds up the learning process.

The results highlighted the effectiveness of the online tutoring course and a high level of student engagement. The comparison between the clusters obtained and the success rate, highlighted the impact that the course had in terms of passing the exam. Cluster 0, which consists of active students in the course, had a high success rate compared to the others equal to 69%. Cluster 1, that involved students with an average activity, had a lower exam pass rate of 55%, while cluster 2, which includes students with low activity, had a percentage of 22% exam pass rate.

The quantitative analysis, carried out by processing the data extracted from the questionnaire, reported excellent feedback in the students' perception and satisfaction.

In the future it can be interesting to test this course for other degree courses, to check if it's effective also for students of courses different from Computer Science.

Furthermore, it could be important to improve the interactive tutor by developing a new standard plug-in that easily allows the teacher to set the code criteria to check the quality of the code and allows interoperability between different programming languages.

Acknowledgement

I would like to express my deep and sincere gratitude to my supervisors Prof. Andrea Perali and Prof. Leonardo Mostarda, for giving me the opportunity to do research and making this work possible. It was a great privilege and honour to work and study under their guidance.

Special thanks to Researcher in innovative teaching and collaborative e-learning activities Daniela Amendola, for her help and for providing useful information throughout this research. She taught me the methodology to carry out the outcomes and to present the results effectively.

I would also like to thank Prof. Cristina Miceli, Full Professors in Bioscience and Biotechnology, Prof. Andrea Perali, Associate Professor in Physics, Prof. Rosario Culmone, Researcher in Computer Science, Prof. Rossana Galassi, Researcher in Chemistry for their collaboration and during my Ph.D. course.

During my PhD period, I was honoured to participate as Visiting Researcher at Middlesex University London and to be involved in research activities with Computer Science Department. This experience has been an added value to my career, and I would like to thank the Computer science department that hosted me.

Thanks to my parents, for their support, and for helping me not only giving me the opportunity to study, but also encouraging me over the years.

Thanks to Riccardo, Emanuela, Claudia, Lola, Alberto, Leonardo, Marcello, always available to give me precious tips, needful in my Ph.D.

Finally, a special thanks goes to my girlfriend Paola, who always believed in me and supported me when I had difficulties during the last months. Her love, affection and strength stimulated my research, reduced my worries, and encouraged me to achieve my goals with the enthusiasm that only she can give me.

Bibliography

Altman, N. S. (1992), *An introduction to kernel and nearest-neighbor nonparametric regression*, *The American Statistician*, 46 (3), 175–185.

Ambrósio A.P., Moreira Costa F., Almeida L, Franco A., Macedo J. (2011), *Identifying Cognitive Abilities to Improve CSI Outcome*, 41st ASEE/IEEE Frontiers in Education Conference, Rapid City, SD, USA.

Amendola D., Miceli C. (2018), *Online peer assessment to improve students' learning outcomes and soft skills*, *Ital. J. Educ. Technol*, 26, 71–84.

Amendola D., Miceli C. (2016) *Online Physics laboratory for University courses*. *J. e-Learn. Knowl.*, 12, 75–85

Anozie N., Junker B. W. (2006), *Predicting end-of-year accountability assessment scores from monthly student records in an online tutoring system*, *Educational Data Mining: Papers from the AAAI Workshop*, AAAI Press.

Alpaydin E. (2009). *Introduction to machine learning*. MIT press.

Beran T., Violato C., Kline D., Frideres J. (2005), *The utility of student ratings of instruction for students, faculty, and administrators: a “consequential validity” study*. *Canadian Journal of Higher Education*, 35(2), 49–70.

Bognár L., Fauszt T. (2020), *Different learning predictors and their effects for Moodle Machine Learning models*, 11th IEEE International Conference on Cognitive Infocommunications, 405-410.

Brohi S.N., Pillai T.R., Kaur S., Kaur H., Sukumaran S., Asirvatham D. (2019) *Accuracy Comparison of Machine Learning Algorithms for Predictive Analytics in Higher Education*. Springer, 285.

Colin H., Donnelly I.A. (2017), *Ambient intelligence: Technologies, applications, and opportunities*. Proc. Annu. Meet. ISSS, 91, 399–404.

Csernica J., Hanyka M., Hyde D., Shooter S., Toole M., Vigeant, M (2002) *Practical Guide to Teamwork*, College of Engineering, 1–70.

Da Re L. (2018), *Promoting the academic success: the Formative Tutoring between research and intervention in the experience of the University of Padua*, Italian Journal of Sociology of Education, 16(3), 185-199.

Đambić, G., Krajcar, M. & Bele, D. (2016). *Machine learning model for early detection of higher education students that need additional attention in introductory programming courses*. International Journal of Digital Technology & Economy, 1 (1), 1-11

Di Battista D. (2005), *The Immediate Feedback Assessment Technique: A Learner-centered Multiple-choice Response Form*, The Canadian Journal of Higher Education, 25(4), 111-131

Dringus L., Ellis T. (2005), *Using data mining as a strategy for assessing asynchronous discussion forums*. Computer & Education Journal, 45, 141–160.

Epstein M.L., Lazarus A.D., Calvano T.B., Matthews K.A., Hendel R.A., Epstein B.B., Brosvic G.M.(2002), *Immediate Feedback Assessment Technique Promotes Learning and Corrects Inaccurate first Responses*, Psychol Record 52, 187–201.

Fan W., Wolters C.A. (2014), *School motivation and high school dropout: the mediating role of educational expectation*, The British journal of educational psychology, 84 (1), 22-39.

Figueiredo J., García-Peñalvo F.J. (2020), *Intelligent Tutoring Systems approach to Introductory Programming Courses*, Eighth International Conference on Technological Ecosystems for Enhancing Multiculturality (TEEM'20), Association for Computing Machinery, New York, USA, 34–39.

Foote E. (1997) *Collaborative Learning in Community College*. ERIC, 1–5.

Fritze Y. & Nordkvelle Y (2003), *Comparing Lectures – Effects of the Technological Context of the Studio*, Education and Information, 8(4), 327-343.

Gavriushenko M. (2017), *On Personalized Adaptation of Learning Environments* Jyväskylä studies in computing, 2017, 1-55.

Gerdes A., Juering J., Heeren B. (2012), *An Interactive Functional Programming Tutor*, ITiCSE'12, 250-255.

Gkontzis A.F., Karachristos C.V., Panagiotakopoulos C.T., Stavropoulos E.C., Verykios V.S. (2017) *Sentiment Analysis to Track Emotion and Polarity in Student Fora*, Proceedings of the 21st Pan- Hellenic Conference on Informatics.

Gottipati S., Shankararaman V. Lin J.R (2018), *Text analytics approach to extract course improvement suggestions from students' feedback*, RPTTEL 13(6), 1-19.

Goyal P., Pandey S., Jain K. (2018) *Deep learning for natural language processing*, Springer.

Graf S., Bekele R. (2006) *Forming Heterogeneous Groups for Intelligent Collaborative Learning Systems with Ant Colony Optimization*. Proceedings of the Intelligent Tutoring Systems, Springer, 4053.

Guelfi M.R., Masoni M., Shtylla J., Formiconi A.R., (2020) *Utilizzo di un MOOC in un corso universitario: studio dell'impatto in termini di apprendimento e gradimento*, Reports on E-Learning, Media and Education Meetings, 8(1), 166-171.

Gusukuma L., Bart A.C., Kafura D., Ernst J. (2018), *Misconception-Driven Feedback: Results from an Experimental Study*, Proceedings of the 2018

ACM Conference on International Computing Education Research, 160-168.

Hackeling G. (2014), *Mastering Machine Learning with Scikit-Learn*, Packt Publishing.

Hadjerrouit S. (2008), *Towards a Blended Learning Model for Teaching and Learning Computer Programming: A Case Study*, Informatics in Education, 7(2), 181-210.

Hamid A., Suhaila S. (2018), *Dyslexia Adaptive Learning Model: Student Engagement Prediction Using Machine Learning Approach*, Springer International Publishing, 372-384.

Hattie J., Timperley H. (2007), *The Power of Feedback*, Review of educational research, 77(1). 81-112.

Hemalatha I., Saradhi Varma G.P., Govardhan A. (2013) *Sentiment Analysis Tool using Machine Learning Algorithms*. International Journal of Emerging Trends & Technology in Computer Science.

Hussain S., Athula G. (2018) *Extending a conventional chatbot knowledge base to external knowledge source and introducing user based sessions for diabetes education*, International Conference on Advanced Information Networking and Applications Workshops (WAINA).

Hylén J. (2021), *Open educational resources: Opportunities and challenges*, <https://docs.intersearch.com.au>.

Kadoić N. , Oreški D. (2018), *Analysis of Student Behavior and Success Based on Logs in Moodle*, Proceedings of the 41st International Convention on ICT, Electronics and Microelectronics MIPRO 2018, 730-735.

Kaplan A. (2021), *Higher Education at the Crossroads of Disruption*, The University of the 21st century. Emerald Publishers.

Kim T., Lim J. (2019) *Designing an Efficient Cloud Management Architecture for Sustainable Online Lifelong Education*. Sustainability, 11.

Kolekar S. V., Pai R. M., Pai M. M, M. (2017), *Prediction of Learner's Profile based on Learning Styles in Adaptive E-learning System*, International Journal of Emerging Technologies in Learning (iJET), 12(06), 31–51.

Kučak D., Juričić V., Đambić G. (2018) *Machine learning in education - a survey of current research trends*, Proceedings of the 29th DAAAM International Symposium.

Kurilovas E. (2019) *Advanced machine learning approaches to personalise learning: learning analytics and decision making*, Behaviour & Information Technology, 38(4), 410-421

Lewis, K.G. (2001), *Using midsemester student feedback and responding to it*, Techniques and strategies for interpreting student evaluations, New Directions for Teaching and Learning, 87, 33–44.

Lizzio A., Wilson K., Simons R. (2002), *University students' perceptions of the learning environment and academic outcomes: implications for theory and practice*, Studies in Higher Education, 27, 27–52.

Mahboob T., Irfan S., Karamat A. (2016), *A machine learning approach for student assessment in E-learning using Quinlan's C4.5, Naive Bayes and Random Forest algorithms*, 19th International Multi-Topic Conference (INMIC), 1-8.

Maina E.M., Oboko R.O., Waiganjo P.W. (2017) *Using Machine Learning Techniques to Support Group Formation in an Online Collaborative Learning Environment*, Int. J. Intell. Syst. Appl. 9, 26–33.

Mbipom B., Craw S., Massie S. (2018), *Improving e-learning recommendation by using background knowledge*, Expert Systems.

McLoughlin C., Lee J.W. (2007) *Social Software and Participatory Learning: Pedagogical Choices with Technology Affordances in the Web 2.0 Era*, Proceedings of the Ascilite—Australian Society for Computers in Learning in Tertiary Education Annual Conference, 664–675.

Mlynarska E., Greene D., Cunningham P. (2016), *Indicators of Good Student Performance in Moodle Activity Data*. CoRR.

Moore S., Kuol N. (2005), *A punitive tool or a valuable resource? Using student evaluations to enhance your teaching*, *Emerging issues in the practice of university learning and teaching*, 2005, 141–148.

Mostow J., Beck J. (2006), *Some useful tactics to modify, map and mine data from intelligent tutors*. *Natural Language Engineering*, 12(2), 195–208.

Mostow J., Beck J., Cen H., Cuneo A., Gouvea E., Heiner C. (2005), *An educational data mining tool to browse tutor–student interactions: Time will tell!*, *Proceedings of the workshop on educational data mining*, Pittsburgh, USA, 15–22.

Nalli G., Mostarda L., Perali A., Pilati S., Amendola D. (2019), *Application of machine learning to the learning analytics of the Moodle platform to create heterogeneous groups in on-line courses*. *Italian Journal of Educational Research*.

Nalli G., Amendola D., Schettini C., Galassi G. (2020), *Tool per la classificazione dei sentimenti implicati in moduli didattici universitari in Modalità E-learning*, *Atti del MoodleMoot Italia 2019*, 29-32.

Nalli G., Amendola D., Perali A., Mostarda L. (2021). *Comparative Analysis of Clustering Algorithms and Moodle Plugin for Creation of*

Student Heterogeneous Groups in Online University Courses. Applied Sciences, 11(13).

Nijstad B.A., De Dreu C.K. (2002) *Creativity and Group Innovation*, Appl. Psychol, 51, 400–406 .

Ortigosa A., Martín J.M., Carro R. M. Sentiment analysis in Facebook and its application to e-learning. *Computers in Human Behaviour*, 2014, pp. 527–541.

Özmen B., Altun A. (2014), *Undergraduate Students' Experiences in Programming: Difficulties and Obstacles*, Turkish Online Journal of Qualitative Inquiry 5(3), 9-27.

Paredes P., Ortigosa A., Rodríguez P. (2010) *A Method for Supporting Heterogeneous-Group Formation through Heuristics and Visualization*, J. UCS 2010, 16, 2882–2901.

Rad P., Roopaei M., Beebe N., Shadaram M., Au Y. (2018), *AI Thinking for Cloud Education Platform with Personalized Learning*, Proceedings of the 51st Hawaii International Conference on System Sciences, 3-12.

Rajaraman A., Ullman J.D. (2010) *Data Mining, Mining of Massive Datasets*, 1-17.

Raschka S. (2015), *Python Machine Learning: Unlock Deeper Insights Into Machine Learning with this Vital Guide to Cutting-edge Predictive Analytics*, Packt Publishing.

Raschka S., Mirjalili V. (2019), *Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2*, Packt Publishing.

Razmerita L., Brun A. (2011) *Collaborative learning in heterogeneous classes*, Towards a Group Formation Methodology, Proceedings of the 3rd International Conference on Computer Supported Education, 189–194.

Richter T., Rudlof S., Adjibadji B., Bernlohr H., Gruninger C., Munz C.D., Stock A., Rohde C., Helmig R. (2012), *ViPLab: A Virtual Programming Laboratory for Mathematics and Engineering*, Interactive Technology and Smart Education, 9(4), 246-262.

Robins A., Rountree J., Rountree N. (2010), *Learning and Teaching Programming: A Review and Discussion*, Computer Science Education, 13(2), 137-172.

Ronchetti M. (2011), *The impact of Internet-carried video-lectures on education*, Streaming Media Architectures, Techniques, and Applications: Recent Advances.

Samuel A.L. (1959) *Some Studies in Machine Learning Using the Game of Checkers*, IBM Journal of Research and Development, 3 (3), 210-229

Shawar A.S., Atwell E. (2007) *Different measurements metrics to evaluate a chatbot system*, Bridging the Gap: Academic and Industrial Research in Dialog Technologies Workshop Proceedings, 89-96.

Shen L., Wang M., Shen R. (2009) *Affective e-learning: Using emotional data to improve learning in pervasive learning environment*. Journal of Educational Technology & Society, 176–189.

Shute V.J. (2008), *Focus on Formative Feedback*, Review of Educational Research, 78(1), 153-189.

Stevens D.D., Levi A.J. (2005) *Introduction to Rubrics: An Assessment Tool to Save Grading Time, Convey Effective Feedback and Promote Student Learning*, Stylus Publishing Sterling.

Tan J., Guo X., Zheng W., Zhong M. (2014), *Case-based teaching using the Laboratory Animal System for learning C/C++ programming*, Computers & Education, 77, 39-49.

Tan P., Ting C., Ling S. (2009), *Learning Difficulties in Programming Courses: Undergraduates' Perspective and Perception*, International Conference on Computer Technology and Development, 42-46.

Vellido A., Castro F., Nebot A. (2010), *Clustering educational data*. In Handbook of Educational Data Mining; CRC Press: Boca Raton, FL, USA, 2010,75–92.

Wing J.M. (2006), *Computational thinking*, Communications of the ACM, 49(3), 33-35.

Yan M., Castro P., Cheng P., Ishakian V. (2016) Building a Chatbot with serverless computing. Proceedings of the 1st International Workshop on Mashups of Things and APIs.

Zaïane O., Luo J. (2001). *Web usage mining for a better web-based learning environment*, Proceedings of conference on advanced technology for education, Banff, Alberta , 60–64.

Zappi A., Beccari R. (2018) Zenbot - agente per il supporto delle attività formative in ambiente Moodle. Atti del MoodleMoot Italia.

Zhou J., Zhou Y., Wang B., Zang J. (2019). *Human–Cyber–Physical Systems (HCPSs) in the Context of New-Generation Intelligent Manufacturing*, Engineering, 5, 624–636.

Zorrilla M. E., Menasalvas E., Marin D., Mora E., & Segovia J. (2005), *Web usage mining project for improving web-based learning sites*. Web mining workshop, Cataluna, 1–22.