

Treball de Fi de Grau
GRAU EN ENGINYERIA EN TECNOLOGIES INDUSTRIALS

**Improvement of the performance of
decision trees in the prediction of
academic results**

MEMÒRIA

Autor: Marc Paraira Serra

Director: Luís José Talavera Méndez

Convocatòria: Juliol 2019



Escola Tècnica Superior d'Enginyeria Industrial de Barcelona



Commonly used terms

There are several terms that will be used repetitively throughout the project. For this reason, we will refer to them using acronyms or simplifications of them. They are listed below.

- First, second and third semester of university will be referred to as Q1, Q2 and Q3 respectively.
- The previous project has the name *“Estudi del rendiment de tècniques de mineria de dades en la predicció de resultats acadèmics”*. But from this point on, we shall sometimes refer to it as PDMP, which stands for “Previous Data Mining Project”. Expressions like “previous project” or “previous work” are also used to refer to it.
- Later on, a verification method called stratified k-fold cross validation will be used. As long as we are in context, we might sometimes refer to it as simply “stratified k-CV” or k-cross validation.
- IDE (Integrated Development Environment): informatic application that contains all the tools necessary to make programming easier. They are designed to maximize the productivity of the programmer by uniting anything that might be necessary for development in a single program.
- CRISP-DM: Cross-industry standard process of data mining.
- DM: data mining.
- ML: Machine learning.

Summary

Data mining is the process of finding relevant feedback from a set of data. This is done by looking for patterns of behavior that can give the information necessary to make predictions of how the sample, and in most cases new data, will behave.

This work will treat this topic and use some of its techniques to make predictions on the performance of the students in ETSEIB.

Despite the concept of data mining embraces a whole range of tools and methods, in this project, the only technique that is going to be used is the decision tree. This will be tested in different ways but the technique itself will not vary.

For this project, a very commonly used method will be our guidance for the sequence of steps that should be followed. This is named CRISP-DM. It is no less than one of the reference models in data mining, which covers the process from the setting of the objectives to the validation of the model.

For the development of this project several tools have been used. As it was decided that the programming language to be used would be Python, all the tools are prepared for this language or are somehow related to it. These tools and the reason to use Python will be explained in its own section of the project.

1. Index

1. Index.....	6
2. Introduction.....	8
2.1. Objectives	9
2.2. CRISP-DM Methodology in the development of data mining projects	10
2.3. Tools used	12
3. Business and data comprehension stage.....	16
3.1. Data comprehension	17
4. Data preparation stage	20
4.1. Data cleaning	21
4.2. Data transformation.....	22
4.3. Data enhancement	24
5. Modeling stage.....	27
5.1. On decision trees.....	27
5.2. On validation methods	30
5.3. Finding baseline results.....	33
5.4. Holdout vs. Stratified K-fold cross validation	35
5.5. Deciding whether to keep or drop the students of 2016.....	44
5.6. Comparing several data frames	49
5.7. Is the amount of data large enough?.....	58
6. Budget.....	63
7. Environmental Impact	65
9. Conclusions	67
10. Future work.....	69
11. Sources	70
12. Annex	71
13. Index of Figures	72
14. Index of Tables	73

2. Introduction

Whether we are aware of it or not, data mining is very present in our lives in one way or another. In fact, one could very easily be surprised at the number of fields in which it is present. Behind any process, any object that we buy, or any service that we use there is some sort of information that is being extracted. If this information is correctly analyzed, it can give relevant knowledge to companies, governments, etc.

For this reason, data mining has become so popular in the past few years. As society becomes more and more digitalized, it is becoming easier to get large amounts of data from the people. As a matter of fact, some argue that it is unavoidable for us as individuals to be part of this new industry.

Take for example online advertisement companies, which are arguably the paradigm of this new tendency. These companies gather data from the sites that a certain user has visited, and through data mining algorithms they are able to predict which products that user is more likely to buy (among other things). With that information the company can then show the user only those commercials of the products that he/she might want the most.

We even see data mining in medicine. One of the examples of its use in medicine is the prediction of cardiovascular diseases. By recording the heart-beat rate of healthy people and of people with certain heart diseases, it is possible to establish patterns that can tell if a person has a heart disease before that person suffers the effects of it, making it possible for the doctors to react in time.

All in all, data mining is a very powerful tool that has, with almost complete certainty, come to our society to stay.

2.1. Objectives

For this project, the setting of the objectives was done before the registration of the project.

The goal of this work is to follow a methodology that uses data mining to predict whether a student from the school of industrial engineering (ETSEIB) would pass or fail the subjects of the third semester of university (one by one).

This task would be based on a previous work from another student of our school of engineering whose work will be evaluated and analyzed in order to find better solutions to the problems that the original project might have struggled with. The name of the previous project is "*Estudi del rendiment de tècniques de mineria de dades en la predicció de resultats acadèmics*" by Montse Heng.

For this purpose, we will attempt to recreate the work of the previous project and use it as our starting point. After that, new possible solutions will be tested, while using the scientific method to isolate the possible factors of impact on the results.

The outcome of the project should be the finding of new feedback that might give new information about the data we have.

2.2. CRISP-DM Methodology in the development of data mining projects

As *Figure 1* shows, CRISP-DM is divided into 6 phases. These phases are:

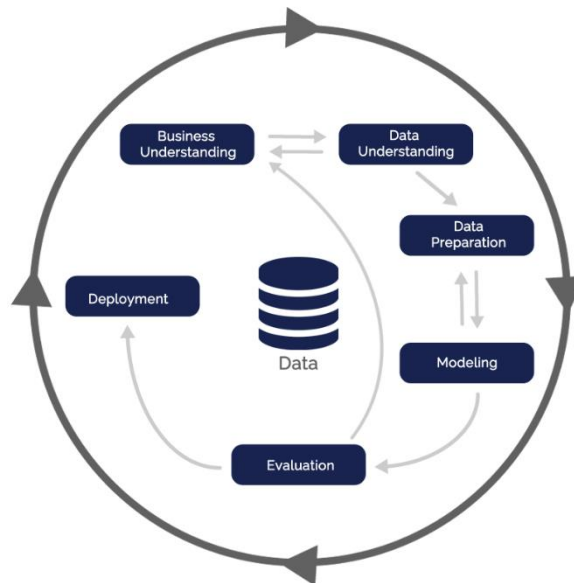


Figure 1. Stages of the CRISP-DM.

Business comprehension stage

The first stage of the methodology contains the setting of the objectives of the project and its explanation in technical terms. The importance of this stage lies on the fact that no success can be achieved in a Data Mining project if the objectives have not been well set, regardless of how good the model or how accurate the data are.

Good comprehension of the project will allow a correct selection of data and a good interpretation of it.

The business comprehension stage is divided into several tasks^[1].

- **Setting the objectives of the project:** this should be the first task to do, in some cases even before it has been decided whether or not the project should be launched. This is because in some cases it might be found that the objectives cannot be precisely set or that they are not likely to be achieved with machine learning tools.
- **State of art:** this part should analyze the current situation before initiating the DM process. The following questions should be made.
What is the previous knowledge about the project?
Is the amount of data enough to get good predictions?
Could the results be highly dependent on data that cannot be obtained?
- **Setting the objectives of the DM:** this part should set what the goals of the Data Mining are.

Data comprehension stage

This step can include initial data collection, data description, data exploration, and the verification of data quality; but these parts are quite flexible. Data exploration such as viewing summary statistics (which includes the visual display of categorical variables) can occur at the end of this phase. Models such as cluster analysis can also be applied during this phase, with the intent of identifying patterns in the data.

Data preparation stage

Data needs to be selected, cleaned, built into the desired format and, if possible, enriched.

The aim of this stage is to make the data readable for the model that we are going to use. This means that a certain amount of communication between this stage and the modeling stage is required because certain models might require that the data is structured in different ways.

The data preparation stage tends to be the most time-consuming stage of the machine learning projects. This is due to the fact that in many cases the data comes in formats that are very far from what the model is able to understand, and thus the data needs a lot of transformation.

Modeling stage

In this stage the most appropriate modeling technique should be selected and applied. In some cases, several modeling techniques could be chosen and compared (as it was done in the previous project). This stage is highly connected to the evaluation stage, to the point where in many cases they are applied simultaneously.

Validation stage

The aim of the validation stage is to determine if the results obtained by the model are good or not.

If they are not good, one must go back to the data understanding stage and modify the data of the preparation stage or the modeling stage depending on what the error has been. If there has been no error and the results are still unsatisfactory it probably means that the available data is not good enough.

Deployment

Once the results have been obtained and validated, they must be implemented and define the bases for a new strategy. In some cases, such as in this one, this stage may include comparing the obtained results with older ones.

2.3. Tools used

Python

Python is a programming language which has recently gained a lot of popularity because it offers many benefits in comparison to other ones.

Among these, there is the fact that Python is an interpreted language, which makes it an easy programming language to work with, because most of its implementations do not require to compile a program into machine language instructions and can thus be easily modified.

It is also a high-level language, which implies that it is easier to understand than many other lower-level languages.

Python can work in many fields as it is a general-purpose language, and this makes it very attractive for non-specialized programmers.

One of the things that makes Python the best choice for this project is that Python includes a set of what we call libraries. Libraries are tools that provide functionalities that would otherwise be inaccessible for Python programmers. Libraries simplify the programming a lot because they include functions that have been programmed which would be really hard to program with a higher-level programming language such as Python. For this project we are going to use libraries such as Pandas, Numpy or Sk-learn.

Python is arguably the best programming language for machine learning, or at least the most commonly used one.

SciKit and TensorFlow are two popular machine learning libraries available to Python developers.



Figure 2. Python logo.

Spyder (Anaconda)

Anaconda is a python and R *distribution*. It aims to provide everything a programmer might need (python wise) for data science. Despite its many functionalities, the main tool that we are going to use from it is its IDE, which is called Spyder.

Spyder is a powerful development environment which is very interactive and that is prepared to work with Python language.



Figure 3. Spyder logo.

We will use Spyder in order to develop all the Python code. Furthermore, it will also be used to interactively and constantly test it in the search for mistakes and solutions.

One of the main benefits of Anaconda provides pre-installed libraries that save the programmer all the effort of downloading and correctly installing them. This is the case of the libraries that we are going to use for this project.

Pandas

It has been mentioned that one of the strongest points that Python has is that it has a whole set of libraries that make it much easier to work with. One of these libraries is Pandas.

Pandas is a library that specializes in data preparation. The reason why Pandas is so good for data treatment is that it includes a set of functions that make tasks that would otherwise be really hard to execute very easy to apply.

For this reason, this will be our main tool for data preparation and manipulation.

Amongst its most important functionalities we have:

- DataFrames are an object that this library includes that resembles a table of data (its appearance is similar to that of an Excel grid) which can be treated and modified through several commands.
- Tools to read and save data from .csv, .xlsx, txt files and SQL databases among others.
- Powerful data modification functionalities such as pivoting and concatenating.
- Capability to understand other libraries' objects such as Nan values from NumPy.

The reasons for us to choose this tool above another one is that it is one of the easiest ones to use when large amounts of data are involved (such as in this case), that it is very easily downloadable and that a lot of documentation from it can be found on the internet.

The main drawback that we must struggle against when using this library is that I personally have never used it before. This means that some time must be invested in learning how the main functions work. But this is not a big concern as Pandas provides a lot of information in its documentation and there are other sources on the internet to be relied on.

SciKit-learn

SciKit-learn is yet another Python library which is specialized in machine learning. It includes functions and algorithms that simplify the process of data mining. It embraces four different branches:



Figure 4. SciKit-learn logo.

- Classification: the algorithms included in this section fracture the data in several categories based on their nature. Decision trees are an example of classification method, which will later be further explained.
- Regression: it is a statistical method that aims to determine tendencies and patterns that the data follows. These patterns can later be used to get predictions of future data.

- Clustering: it is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense) to each other than to those in other groups (clusters). It is a main task of exploratory data mining, and a common technique for statistical data analysis, used in many fields, including machine learning.
This can be useful for example to create patterns of behavior of clients that buy certain products and could be used to predict what other products will these clients be interested in.
- Dimensionality reduction: In machine learning classification problems, there are often too many factors based on which the final classification is done. The higher the number of factors, the harder it gets to visualize the training set and then work on it. Sometimes, most of these factors are correlated, and hence redundant. Dimensionality reduction is the process of reducing the number of random variables under consideration, by obtaining a set of principal variables.

It must be kept in mind that SciKit has a drawback, which is that its machine learning tools are not very well prepared to work with non-numerical data (it is the case of our decision tree). This means that we might have some variables that give us information and that might have to be transformed or simply dropped because of their unreadable nature.

To our luck, most of our relevant data seems, at first sight, to be numerical and sortable data, which pretty much solves the issue.

NumPy

As it is stated in its official webpage NumPy is the fundamental package for scientific computing with Python. It contains among other things:

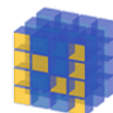



Figure 5. NumPy logo.

- A powerful N-dimensional array object.
- Sophisticated (broadcasting) functions.
- Tools for integrating C/C++ and Fortran code.
- Useful linear algebra, Fourier transform, and random number capabilities.

We are only going to use a small part of what NumPy is because it has some functions and objects that will make our job easier, but we are not going to use it to its full potential. The main object that we are going to use from it are NumPy arrays, which are an object that makes it easier to operate and to visualize certain types of data.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms.  *Figure 6. Matplotlib logo.*

It offers ways to create and modify many different types of plots of data with just a few lines of Python code.

Matplotlib will be our tool when it comes to plotting any sort of data.

3. Business and data comprehension stage

The next step should be to ask ourselves the introductory questions that should be asked when facing a project that follows the CRISP-DM structure.

At the end of our work we should have trustworthy information about:

What the previous knowledge about the project is.

At this point, the knowledge we have about the state of art is the following. There is a previous project on the field which we will take as a reference where a preprocessing of the data obtained a table of data that will be taken as reference in this work. This table contained a sample of students and predicting columns. This will be our starting point, from which we will start gathering our results.

If the previous results can be improved.

Our work will focus into analyzing the steps that were taken in the previous project and contribute with additional feedback those parts that could have been developed differently. For example, we will test different validation methods and check for alternative data frames to the one previously used.

If the amount of data is enough to get good predictions.

At this point, we do not know if the amount of data is large enough to get good predictions. This remains as one of the possible reasons for the last results not being good enough, but we do not know at this point.

In order to answer this question, we will plot the learning curve of the model and analyze it.

If the results could be highly dependent on data that cannot be obtained.

Yes, the results will in fact be highly dependent on data which is not possible to obtain mainly due to its complexity but also to the need to preserve the anonymity of the students that we are studying.

In fact, this is the main reason why, even in the best scenario, the results we will obtain will have inaccuracies.

Just to give some examples of data that cannot be obtain we have things such as the following ones: *personal situation*, *motivation*, *attendance to academies*, *attendance to "Aula Lliure"*, *available time to study*, *working while studying*, etc. One can see without much difficulty that this, amongst others, are factors that can influence the performance of a student and which are really hard to obtain data from.

3.1. Data comprehension

We have data from both a quantitative and a qualitative nature.

One of the most important information that we have are the grades of the students. This will be the most important data we have, as it determines our response variables and many of our input variables.

The data was divided into three Microsoft Excel documents, each one of them containing the following information:

<u>File 1</u> Name: "dadespersnombrespreins" Summary: Data from the students prior to their entrance to ETSEIB Information (columns): CODI_EXPEDIENT SEXE CP_FAMILIAR ANY_ACCES TIPUS_ACCES NOTA_ACCES CP_CENTRE_SEC	<u>File 2</u> Name: "qfaseini" Summary: Data from all the subjects of the initial phase taken at ETSEIB since 2010 Information (columns): CODI_PROGRAMA CODI_EXPEDIENT CODI_UPC_UD CREDITS CURS QUAD SUPERA NOTA_PROF NOTA_NUM_AVAL NOTA_NUM_DEF GRUP_CLASSE	<u>File 3</u> Name: "qfnoini" Summary: Data from all the subjects not belonging to the initial phase taken at ETSEIB since 2011 Information (columns): CODI_PROGRAMA CODI_EXPEDIENT CODI_UPC_UD CREDITS CURS QUAD SUPERA NOTA_PROF NOTA_NUM_AVAL NOTA_NUM_DEF GRUP_CLASSE
--	--	---

Table 1. Data files and their information.

- CODI_EXPEDIENT: contains the code which is assigned to a student at the moment of registration to the university. This code is unique and unchangeable for each student. Its importance is key because it will be used to track which data corresponds to which student.
- SEXE: gender of the student.
- CP_FAMILIAR: postal code of the place of residence of the student.
- ANY_ACCES: year of entry to ETSEIB.

- TIPUS_ACCES: in theory there are several ways to enter the school, each one of them has an assigned code. The standard way to enter the school has the value of "1" assigned.
- NOTA_ACCES: grade of entry, prior to starting in the school.
- CP_CENTRE_SEC: postal code of the school where the student went to prior to its entrance to ETSEIB.
- CODI_PROGRAMA: Since the school has given several degrees (Industrial engineering, chemical engineering...), a code has been assigned to each one of them. Most of the subjects belong to the degree of industrial engineering or "Grau en Enginyeria en Technologies Industrials", which has the code "752".
- CODI_UPC: code of a certain subject.
- CREDITS: ECTS that each subject has.
- CURS: year in which a subject was taken.
- QUAD: semester in which a subject was taken.
- SUPERA: passed or failed.
- NOTA_PROF, NOTA_NUM_AVAL: grade before compensation point.
- NOTA_NUM_DEF: grade after compensation point

This data contains information from all the subjects held at the faculty of industrial engineering, but we are only interested in 16 of the subjects; the ones belonging to Q1, Q2 and Q3.

In *Table 2*, we can see the subjects that we are interested in. The subjects of Q1 and Q2, belong to our predicting variables. The rest belong to Q3 and are our response variables.

The subjects are seen in the following table:

Code	Official name	Name in English	Semester
240011	Àlgebra Lineal	Linear Algebra	Q1
240012	Càlcul I	Calculus I	Q1
240013	Mecànica Fonamental	Fundamental Mechanics	Q1
240014	Química I	Chemistry I	Q1
240015	Fonaments d'Informàtica	Fundamental Informatics	Q1
240021	Geometria	Geometry	Q2
240022	Càlcul II	Calculus II	Q2
240023	Termodinàmica Fonamental	Fundamental Thermodynamics	Q2
240024	Química II	Chemistry II	Q2
240025	Expressió Gràfica	Graphical Expression	Q2
240031	Electromagnetisme	Electromagnetics	Q3
240032	Mètodes Numèrics	Numerical Methods	Q3
240033	Materials	Materials	Q3
240131	Equacions Diferencials	Differential Equations	Q3
240132	Informàtica	Informatics	Q3
240133	Mecànica	Mechanics	Q3

Table 2. Identification of the subjects in Q1, Q2 and Q3.

At this point we have acquired the basic knowledge about the data we needed so that to begin its preparation.

4. Data preparation stage

The data used for the project was supplied by the university itself, which has information from all the grades of all the students (as it was mentioned in the previous section). Despite the data was already in quite a good format, it still needed quite a few changes.

Given that we are working with Pandas and with Sk-learn, the data we have must be converted to a data frame. The data frame we are going to use for the model will initially be the one used in the previous project. This is because, as we have said, we take as our starting point the point where the last project left the work.

Ideally, we could just grab the code in the last project, run it, and we should obtain the data frame. But the project makes reference to several files that were used to run the code which we do not have. This means that we are unable to directly run the code from the previous project to obtain the data frame. Instead of investigating what these files might contain, we decide to write the code to build the data frame from scratch, following the descriptions of the previous project on how the data frame was built.

The path we will follow is to create a basic data set and add all the extra columns that it needs.

As the first step, the objective is a rather simple data set. In order to make it easy for us to modify the initial data set we must be careful when programming and always keep in mind to write the code so that certain parts can be changed without having to change the main structure of the code or forcing us to start it back from scratch.

The information that the basic *Data Frame* should contain is shown in *Table 3*.

	LG_S1	NT1	LG_Sj	NTj	LG_Sn	NTn	P/F1	...	P/Fn
S1	GS1.1	TS1.1	GSj.1	TSj.1	GSn.1	TSn.1	P/F1.1	...	P/Fn.1
...
Si	GS1.i	TS1.i	GSj.i	TSj.i	GSn.i	TSn.i	P/F1.i	...	P/Fn.i
...
Sm	GS1.m	TS1.m	GSj.m	TSj.m	GSn.m	TSn.m	P/F1.m	...	P/Fn.m

Table 3. Information in the basic data frame.

Si: Student number i

LG_Sj: Grade (of a certain student) on the subject j at its last attempt

NTi: Number of tries (of a certain student) to pass subject i

GSj.i: last grade on subject j of the student i

TSj.i: Tries on subject j of student i

P/Fi.j: whether a student i passed or failed a certain subject j

A sample of the basic data frame can be seen in *Table 4*. This sample only contains 14 out of the 2530 that the basic data frame has.

240133		1																									
240132		1																									
240131		1																									
240033		1																									
240032		1																									
240031		1																									
240025_c		1																									
240025_n		5.1																									
240024_c		1																									
240024_n		7.0																									
240023_c		1																									
240023_n		6.8																									
240022_c		1																									
240022_n		7.0																									
240021_c		1																									
240021_n		7.6																									
240015_c		1																									
240015_n		8.0																									
240014_c		2																									
240014_n		8.3																									
240013_c		1																									
240013_n		6.3																									
240012_c		1																									
240012_n		7.6																									
240011_c		1																									
240011_n		6.6																									

Table 4. Sample of the basic data frame.

Notes:

- The columns with “_n” at the end are the grades whereas the columns with “_c” at the end are the number of attempts. The columns for the response variables have not got such terminations.
- “un” stands for “unattempted”. These are subjects which have not been attempted by a certain student. We will later explain how this values should be treated.

After the basic data set has been created, the necessary extra columns of data will be added to obtain the data set from the previous project. These columns will be explained in the data enhancement section.

4.1. Data cleaning

Firstly, we must select the data that we need for transformation and enhancement.

The steps of the data cleaning are:

1. The first step is to create a function that opens the Excel files as Pandas data frames (the data has been given to us in the 3 Excel files we explained) and saves the data in the form of one or more data frames. In our case we have created 3 data frames because it makes it easier to work with. For this part only 2 out of the 3 Excel files need to be opened. The ones called “qfaseini” and “qfasenoini”.
2. We are only interested in students who have taken the degree in industrial engineering (“Grau en Enginyeria en Teconologies Industrials”). The information about the degree is kept in the column “CODI_PROGRAMA” with the value 752. Therefore we drop all the students with “CODI_PROGRAMA” != 752.
3. Some students coming from other schools of engineering have some validated subjects, which means that they are subjects that they took in another school and that due to its similarity to a certain subject in ETSEIB, they do not need to be retaken. We cannot include these students and therefore we drop students with “GRUP_CLASSE” == “CONV”, which is the condition to the case explained.
4. There are some columns of data in the files we opened that are redundant or that provide information that will not be necessary from this point on. Therefore, we drop “CODI_PROGRGAMA”, “CREDITS”, “SUPERA”, “NOTA_PROF” and “NOTA_NUM_AVAL”.
5. We are only interested in the subjects belonging to Q1, Q2 and Q3, therefore we drop the rest.
6. We drop all the Nan values because our model is unable to handle them.
7. We drop all the students who have not passed all the subjects in Q1 and Q2.

After these seven steps the data has been cleaned and is ready for transformation.

4.2. Data transformation

The main step of the data transformation is the pivoting. This could be in fact a very complicated task was it not because Pandas includes a function which simplifies the task to just a few lines of code.

Pivoting means transforming the values of a given column into new columns. A visual example of this can be seen in *Figure 7*, where the values of the original column “bar” are established as the new columns.

Pivot

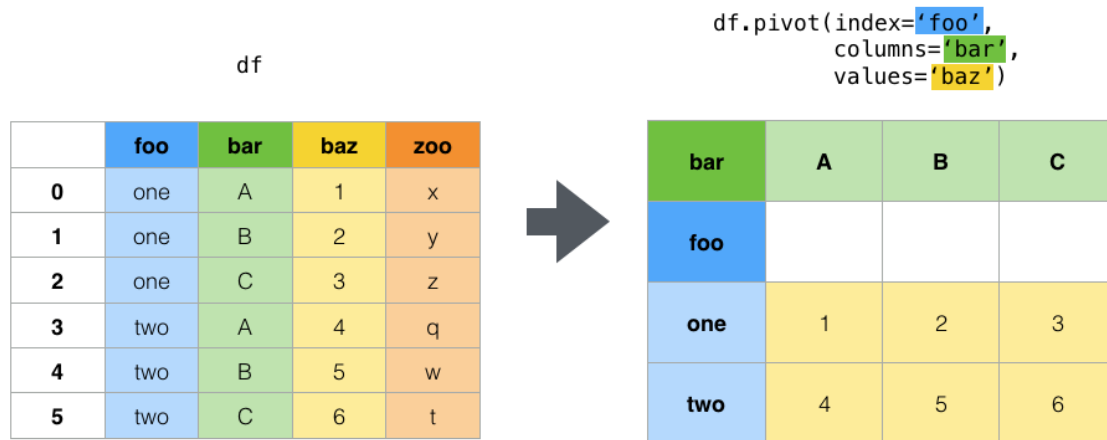


Figure 7. Pivoting.

As has been explained before we are interested that each row contains a student.

The problem with this procedure in our case is that there are students who have taken more subjects than others and therefore they should have a larger number of columns because they have more grades. This would result in the data frame having Nan values, which cannot be read by the model.

To overcome this problematic the following choice has been made. Only the best (and last) grade of each subject will be kept from each student and the rest will not be in the data frame (the best grade of a student will be their grade on their last attempt). Instead, we will count the number of attempts of each student at each subject. This has already been shown in *Table 3*.

Since we are trying to predict whether a student will pass or fail, which is a binary response, another transformation is required. The First attempt on the courses belonging to Q3 of each student must be transformed so that grades lower than 5.0 are turned into 0s and grades equal or higher than 5.0 are turned into 1. If the subject has not been taken the value is changed to the string “unattempted”.

By doing this we will obtain the response in a binary way, with “unattempted” being a reference for the rows that we will drop for each subject. The last project’s data frame did not keep the students who have not taken all the subjects of Q3. When we recreate that data frame, we will drop these students as well because otherwise the results of our test would be influenced by this change of rows, and that is unwanted. The reason why we decide to keep these values for now is because later on we might want to test if keeping them could improve the results. They could be interesting to use if we see that the quality of the results increases when the number of rows increases and that the

model has still not reached a point where the learning curve is stable, meaning that more rows will still improve the results.

At this point our data frame has 20 columns as part of the prediction variable and 6 columns which correspond to the variables to predict. It has a total of 2530 rows, each one corresponding to one student.

4.3. Data enhancement

This part could easily be considered as a section inside of the data transformation, as what we do is take the data we have and give it another nature. Nonetheless, a separate section is given to it as it is not part of the process to obtain the basic data set, meaning that we could apply the model to the data frame we have before the steps that are about to be applied and obtain some results.

We consider that the following columns of data could give us information that could influence the response variable:

- “0”: this column corresponds the total number of subjects failed at Q1 and Q2.
- “NOTA_ACCES”: this column contains the grades of entrance to the university of the students which should go up to 14.
- “NOTA_NUM_DEF”: average grade of all the subjects that the student has taken at Q1 and Q2. This includes all the attempts.
- “N_QUAD_FASEIN”: number of semesters that the student took to pass all of the subjects of Q1+Q2.
- “ANY_ACCES”: year of entrance to the university.
- “GRUP_INI”: first group in which the student signed in.

The data frame from the previous project contained all the mentioned columns (with different names but same data) except for “NOTA_ACCES”, and “ANY_ACCES”. The last one of these two was used to split the data into two groups as will be explained in the modeling stage, but it was not used as an input variable for the model.

There is a problem with some of the columns. We do not have all the data that should fill them for all 2530 students. Instead, we only have these three columns of data for 2462 out of the 2530 students we initially had.

This means that in order to add these three columns to the data frame we would have to lose 68 rows of information. In any case, 68 rows do not seem like too many in front of 2530 that we initially had, and it is the only way to recreate the data frame from the last project.

Because we would also like to test how good the data frame in the PDMP is, we will also create a couple more data frames which will be tested in the means to compare the results to the ones obtained by the last project's data frame.

We decided that 3 data frames be created and tested (one of them being the one from the last project):

- **“df0”**: this will be the basic data frame that we have already spoken about. It will include 2 columns for each predicting subject, one for the best grade in that subject and one for the number of attempts, as it has already been said. This data frame is not enriched, but it can still be tested to determine if data enhancement has any positive effect on the results or if, on the contrary, it is unnecessary. Despite its size is greater than the size of the other data frames in terms of rows, we will be forced to drop the extra students because otherwise the results would not be comparable. This idea will be further developed in the modeling stage. Size: [2530x26] or [2320x26] if students having an “unattempted” are dropped.
- **“dfanterior”**: this data frame is a recreation of the data frame of the last project. This includes all the columns of data explained before except for “NOTA_ACCES” and “ANY_ACCES” (as said, “ANY_ACCES” is used but only for the split). The columns are exactly the same as in the last project, but the number of rows varies a bit. But as the project does not explain what it does in certain situations, we have to face during the data cleaning, and it does not provide some of the files required to recreate the exact data frame we can only get as close as we can. In fact, though, it is not critical, as the number of rows is not that far apart. The last data frame had a total of 2387 rows. Size: [2462x31*] or [2275x31*] if students having an “unattempted” are dropped.
- **“dfall”**: finally, we decide to make a data frame with all of the columns of data that we have. Which means it will have 25 predicting variables and 6 response variables. At the end, we have gained 5 columns of data in front of losing just 68 rows. [2462x32*] or [2275x32*] if students having an “unattempted” are dropped.

**The number of columns decreases by 1 if we do not count “ANY_ACCES”.*

Note that no columns are added to the response variable as what we are trying to predict is always the same.

Also note that “dfanterior” has “unattempted” values in the response variables. In the last project these values were dropped, so in order to obtain the data frame that matches the one from the last project (as close as the information we have allows as to) they will have to be dropped as well. For now, they are kept as we do not know if they could be useful at some point.

5. Modeling stage

5.1. On decision trees

A decision tree is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences. It is one way to display an algorithm that only contains conditional control statements. In machine learning, decision trees are often used as a data classification tool. The tree divides the data on several groups based on its attributes.

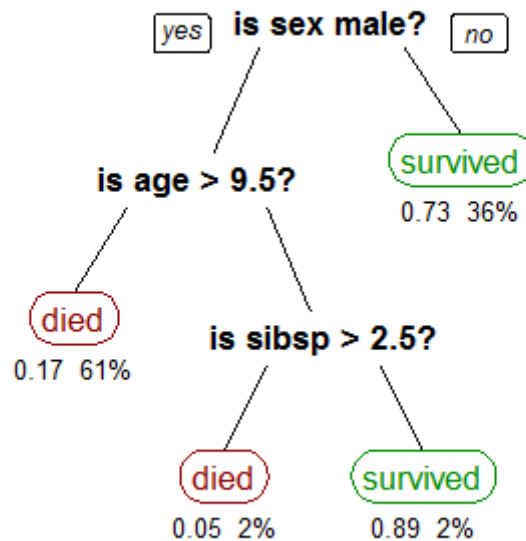


Figure 8. A decision tree showing the survival of passengers of the Titanic.

Some common terms used with decision trees:

- **Root Node:** It represents entire population or sample, and it gets further divided into two or more homogeneous sets.
- **Splitting:** It is a process of dividing a node into two or more sub-nodes. *Note: do not confuse the splitting of a decision tree with the splitting of a data frame.*
- **Decision Node:** When a sub-node splits into further sub-nodes, then it is called decision node.
- **Leaf/ Terminal Node:** Nodes do not split is called Leaf or Terminal node.
- **Pruning:** When we remove sub-nodes of a decision node, this process is called pruning. You can say opposite process of splitting.
- **Branch / Sub-Tree:** A sub section of entire tree is called branch or sub-tree.

- **Parent and Child Node:** A node, which is divided into sub-nodes is called parent node of sub-nodes whereas sub-nodes are the child of parent node.

We use the decision tree implementation included in the sklearn library, specifically the class `sklearn.tree.DecisionTreeClassifier()`.

It has one of the limitations that has been mentioned about sklearn. This tool is only capable of working with numerical data, which means that Nan values and other non-numerical values such as strings that could reference attributes must have been dropped or modified onto numerical values before being used in the tree. The only exception are bools, as Python interprets them as 1 for True and 0 for False, but in some cases, they can still bring some problems to the modeling. This is not a big issue for our study as our data is mostly numerical.

One concept that we must keep in mind when we build the model is *overfitting*. Overfitting is "the production of an analysis that corresponds too closely or exactly to a particular set of data and may therefore fail to fit additional data or predict future observations reliably" ^[10].

In a data mining model, overfitting translates in a very good fitting of the training data but a low "fraction of correct guesses" when it comes to predicting new data.

Since our model is supposed to be used for prediction purposes and not for simple description of the data we have, it is important that we always avoid overfitting. Otherwise the model would not be able to predict the new data correctly.

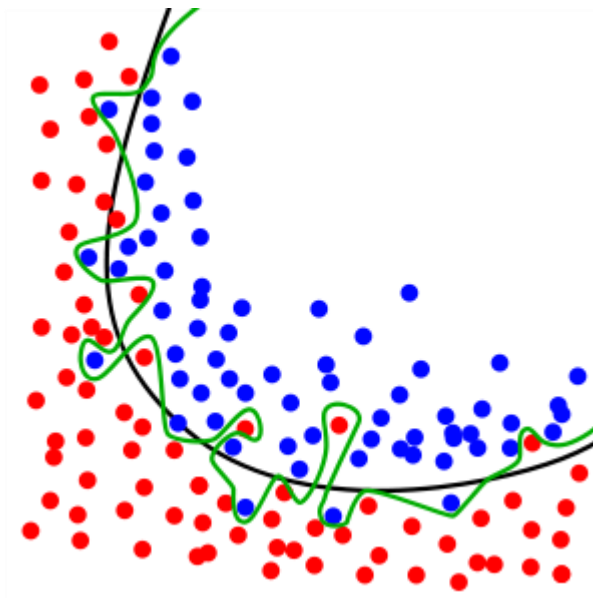


Figure 9. Graphical representation of the phenomena of overfitting.



In *Figure 8* the green line represents a model where there is overfitting. The black line does not explain the training data that well but will make better predictions of new data.

Our `sklearn.tree.DecisionTreeClassifier()` has several parameters that we are going to work with at some stage of our modeling. The sk-learn documentation gives the following information about them^[6]:

`max_depth` : *int or None, optional (default=None)*

The maximum depth of the tree. If `None`, then nodes are expanded until all leaves are pure or until all leaves contain less than `min_samples_split` samples.

`min_samples_leaf` : *int, float, optional (default=1)*

The minimum number of samples required to be at a leaf node. A split point at any depth will only be considered if it leaves at least `min_samples_leaf` training samples in each of the left and right branches. This may have the effect of smoothing the model, especially in regression.

- If `int`, then consider `min_samples_leaf` as the minimum number.
- If `float`, then `min_samples_leaf` is a fraction and `ceil(min_samples_leaf*n_samples)` are the minimum number of samples for each node.

`random_state` : *int, RandomState instance or None, optional (default=None)*

If `int`, `random_state` is the seed used by the random number generator; If `RandomState` instance, `random_state` is the random number generator; If `None`, the random number generator is the `RandomState` instance used by `np.random`.

The tree has more attributes, but they are not going to be changed from their default values in this project.

5.2. On validation methods

Validation techniques in machine learning are used to split the data into training and testing data and to validate the quality of this cut. If the data volume is large enough to be representative of the population, you may not need the validation techniques. However, in real-world scenarios, such as in this one, we work with samples of data that may not be a true representative of the population. This is where validation techniques gain importance, as a way must be found to pick a sample that is as representative of the overall population as possible.

Holdout Validation

In this technique, the data is split into two different datasets labeled as a training and a testing dataset. This can be a 60/40 or 70/30 or 80/20 split, for example. The training set is used to fit the model and the testing set used to make predictions on the already fixed model.

With this method the evaluation may depend heavily on which data ends up in the training set and which ends up in the testing set, and thus the evaluation may be significantly different depending on how the division is made.

K-Fold Cross-Validation

The data set is divided into k subsets, and the holdout method is repeated k times. Each time, one of the k subsets is used as the test set and the other $k-1$ subsets are put together to form a training set.

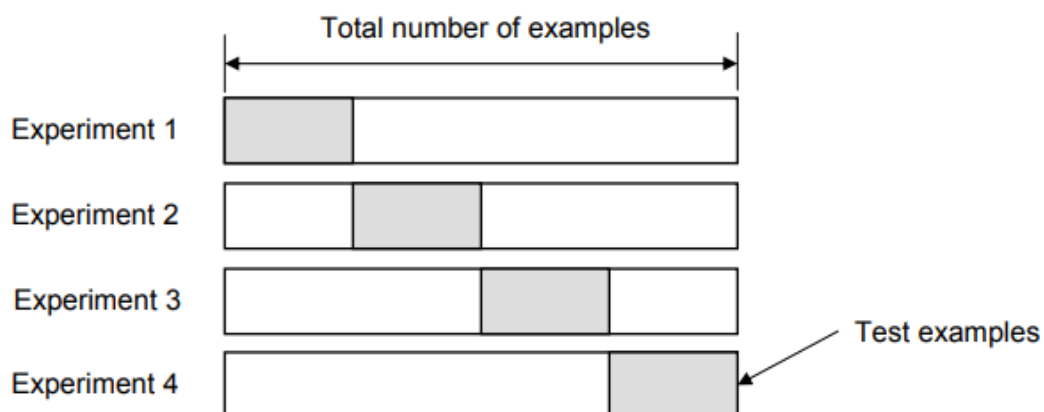


Figure 10. Basic principle the k -CV.

The advantage is that entire data is used for training and testing, which means that it matters less how the data is divided. The error rate of the model is average of the error rate of each iteration. This technique can also be called a form the repeated hold-out method. The error rate could be improved by using stratification technique.

The disadvantage of this method is that the training algorithm has to be rerun from scratch k times, which means it takes k times as much computation to make an evaluation.

Stratified k-fold cross validation

This is a special case within k-fold cross validation. The procedure is the same when it comes to creating k subsamples and using all of them for training in testing in the same way as in the last case, but in this case, the way in which the cuts are done is not random, instead stratification is followed.

Stratification consists on rearranging the data to ensure that each fold is a good representative of the whole. This ensures that the splits are as homogeneous as possible, which reduces the impact of randomness on the results. For this reason, this validation method has a big acceptance among programmers.

About this method, Ron Kohavi, professor at Standford university states in his report “*A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection*” that: “stratification is generally a better scheme, in terms of variance, when compared to regular cross-validation”.

Confusion matrix

As part of our validation, we will use a tool which is not a validation method itself, but that will provide valuable information: the confusion matrix.

TN	FP
FN	TP

Figure 11. Confusion matrix.

TN (true negative): correct prediction of a negative value.

FP (false positive): returned positive but value was negative.

FN (false negative): returned negative but value was positive.

TP (true positive): correct prediction of a positive value.

In a model with 100% of correct guesses we would only have values different than 0 in TN and TP. The higher that FP and FN are in comparison to TN and TP the higher the quantity of incorrect guesses is.

The matrix will be useful in order to know what type of error is more common, or in other words, if our model predicts “passes” or “fails” better.

Learning curve

A learning curve is a plot that helps determine if the amount of data of a data mining project is enough.

The idea is to plot the “fraction of correct guesses” for different amounts of data.

If the curve ceases to increase before all the data has been used it means that it has reached a point where adding more data does not improve the results. If that is the case, the amount of data is large enough. If on the contrary, it does not stabilize, it means that the amount of data is not enough.

This is a very visual method that does not follow a strict statistical analysis, but it tends to be sufficient, as the answer to whether the amount of data is large enough is quite open.

5.3. Finding baseline results

For later stages of the study, we will vary several parameters and inputs of the modeling stage and we will check how that affects the final results. In order to see if we are following the right path, we would like to find the results for a non-optimized combination of inputs. What we mean by that is that we will use the data frame from the last project “dfanterior” and set the parameters `max_depth` and `min_samples_leaf` to their default values (`max_depth=None` and `min_samples_leaf=1`).

Later on, we will generate results for several combinations of inputs to see if the results improve.

Test 1

We will generate several decision trees while leaving all of the mentioned inputs constant. The only parameter that we will vary is the “`random_state`” of the decision tree.

We generate a range of models while varying the “`random_state`”. This is done to reduce the effect of variability on the results we obtain.

The validation technique we use for this state is the stratified k-CV because it is generally considered to have a small amount of variability.

After obtaining the results, three parameters are found for each subject:

- `max_value`: highest value out of a range of 50 random states for each subject. In this case it is not that relevant as it is higher than the rest by chance. In later stages of this project this value will gain relevance.
- `mean_value`: average value of all the “fraction of correct guesses” for 50 random states.
- `std`: standard deviation of 50 results we found for each subject.

Subject	<i>max_value</i>	<i>mean_value</i>	<i>std</i>
Electromagnetics	0.6633	0.6409	0.0082
Numerical Methods	0.8501	0.8372	0.0064
Materials	0.7077	0.6894	0.0068
Differential Equations	0.7446	0.7312	0.0081
Informatics	0.7605	0.7429	0.0071
Mechanics	0.6259	0.609	0.0101

Table 5. Baseline results of the decision tree.

Conclusions

The parameters *max_value* and *mean_value* are the “fraction of correct guesses” for a non-optimized combination of inputs. Following the line of the last project at first and going beyond it later on, we will try to modify several inputs of the model to increase these two values. They are a reference to what our baseline results should look like.

5.4. Holdout vs. Stratified K-fold cross validation

In the last project, the method that was used to split the data between training and testing data was a holdout method.

The reference that was used in the last project to split the data was the year of entry to the university. The cut was done in the following way:

- If a student has entered the school before 2015 → training
- If a student has entered the school in 2015 or afterwards → testing

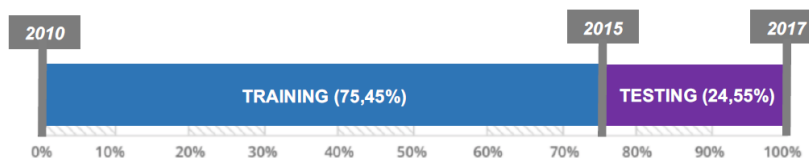


Figure 12. Train-test split used in the PDMP.

Figure 12, which was taken directly from the last project shows it.

This split follows the idea of a learning curve, where it is tried to simulate how a model built at a certain point in time would have determined the following years. This is just one of the possible ways to split the data, but there are more. We would like to see if any contributions can be done to this step of the process in the form of another way to split the data.

For this purpose, we propose to use stratified k-fold cross validation. As we have said, this can be in some cases a more reliable method than holdout and we would like to see if this is one of these cases.

Our first hypothesis is that stratified k-CV could be a more reliable method than holdout because the amount of data we have is quite small, and this has the tendency to make the results more susceptible to where the cut is done.

To see if there is evidence to claim that using stratified k-CV is more desirable with the data we have, several tests have been run.

Test 1

The first prove to our hypothesis comes when pick our response variables and find the fraction of "1" that it has for each subject. We find the fraction of passes for the subsample that was used in the PDMP as training set and for the subsample that was used as testing set.

Note that this test comes from direct search on the response variables, and that no predictions have been done yet nor has any sort of model been created; we are only looking at the data we have in the data frame.

When finding the percentage of “Passes” we obtain the following table:

Subject	Fraction of “Passes” for training set	Fraction of “Passes” for testing set	Overall fraction of “Passes”
Electrom.	0.691345	0.825328	0.7160
Num. Meth.	0.90147	0.965049	0.9130
Materials	0.730288	0.874759	0.7543
Diff. Equa.	0.789989	0.966535	0.8237
Informatics	0.812738	0.916667	0.8321
Mechanics	0.498911	0.792373	0.5534

Table 6. Fraction of passes of each set of data used in the PDMP.

As we see in Table 6, there is a clear unbalance for all subjects with this method. This means that the data used for the training does not match the one used for the testing so well. This helps sustain the hypothesis that there could be another validation method that split the data more homogeneously.

Apart from this, we have noticed that students of 2016 only have a value of 2 for the column “N_QUAD_FASEIN” which means that they have taken the minimum possible number of semesters to pass Q1 and Q2. For the rest of the student this value can only be equal or higher. This could mean that this sample of students, and to a smaller degree the testing set, have a better performance than the overall set, and this could have an impact on the results. Table 7 shows how the “N_QUAD_FASEIN” only remains constant at 2 for students of 2016. The tendency is the same for all the rows of the data frame.

N_QUAD_FASEIN	ANY_ACCES	N_QUAD_FASEIN	ANY_ACCES
2	2010	4	2014
5	2010	3	2014
2	2010	2	2015
4	2011	3	2015
2	2011	4	2015
2	2011	2	2016
3	2012	2	2016
3	2012	2	2016
2	2012	2	2016
3	2013	2	2016
4	2013	2	2016
2	2013	2	2016
3	2014	2	2016

Table 7. “N_QUAD_FASEIN” and “ANY_ACCES” for a sample of the data set.

This makes us think that the 2016 students we have could perform better than the average student.

As a comment, *Table 6* shows that the overall fraction of “Passes” is between that of the training and the data set, which should be this way. Also, it is closer to that of the training set, which makes sense because the size of the training set is larger than the size of the testing set.

Test 2

To see if this issue has a significative effect on the results, we test the data frame through holdout validation and through stratified k-fold cross validation. We know that stratified k-cross validation, attempts to do the split in a way that the testing data is represented in the training data. For this reason, it can be used as a sort of reference. If the values found applying the last project methodology are far from the ones found through stratified k-fold cross validation, then we can guess that the matter we are talking about does have numerical evidence to support it.

For this test we will proceed in the following way:

Step 1

While using holdout validation, we will create table for each subject containing a combination of the parameters “min_samples_leaf” as rows and “max_depth” as columns and its respective value, which will be the fraction of correct guesses. These will attempt to be a recreation of the tables that were used in the previous project to find the best combination of “min_samples_leaf” and “max_depth” for each subject. For this reason, the combination of both of these values will be extracted from the las project.

Step 2

We will find the same tables for the same combinations of “min_samples_leaf” and “max_depth” but this time using stratified k-cross validation.

Step 3

We will compare the tables and see if the results though holdout are close enough to those found through stratified k-cross validation so that we can consider that the train-test split has been done correctly. As the tables contain a big range of results for each subject, we compare them using the basic statistical parameters. To have a correct representation of all of them the parameters used for comparison will be the following ones:

1. **max_value:** highest value for any combination of “min_samples_leaf” and “max_depth” for each subject. It is the same concept as in the previous section but for a different combination of inputs.
2. **mean_value:** average value of all the possible combinations of “min_samples_leaf” and “max_depth”. This value will provide a double check in case the max_value shows something fishy that does not follow the general tendency of the rest of the results. It is the same concept as in the previous section but for a different combination of inputs.
3. **row_std:** sum of the standard deviation of all rows calculated separately divided by the number of rows. Shows if changing “min_samples_leaf” changes the results a lot or if it does not.
4. **column_std:** sum of the standard deviation of all columns calculated separately divided by the total number of columns. Shows if changing “max_depth” changes the results a lot or if it does not

Step 1

The results obtained through holdout can be seen in the 6 following tables:

Electromagnetics (holdout)										
max_depth										
min_samples_leaf		3	5	7	10	12	14	16	18	20
	2	0.6937	0.6734	0.6441	0.6126	0.5946	0.5788	0.5991	0.5968	0.5968
	3	0.6937	0.6734	0.6396	0.6284	0.6171	0.6216	0.6239	0.6171	0.6036
	5	0.6937	0.6802	0.6599	0.6284	0.6081	0.6306	0.6239	0.6149	0.6171
	10	0.6937	0.6892	0.6464	0.6441	0.6464	0.6396	0.6396	0.6396	0.6441
	20	0.6937	0.6802	0.6644	0.6599	0.6554	0.6599	0.6667	0.6622	0.6599
	25	0.6937	0.6802	0.6824	0.6667	0.6667	0.6667	0.6667	0.6667	0.6667
	30	0.6937	0.6892	0.6712	0.6712	0.6712	0.6712	0.6712	0.6712	0.6712
	35	0.6937	0.7027	0.6959	0.6959	0.6959	0.6959	0.6959	0.6959	0.6959
	40	0.6937	0.7117	0.6757	0.6757	0.6757	0.6757	0.6869	0.6869	0.6757
	50	0.6937	0.7117	0.6982	0.6982	0.6982	0.6982	0.6982	0.6982	0.6982

Table 8. Modeling: Holdout, Electromagnetics.

Numerical Methods (holdout)										
<i>max_depth</i>										
<i>min_samples_leaf</i>		3	5	7	10	12	14	16	18	20
	2	0.9437	0.9459	0.9302	0.8829	0.8604	0.8468	0.8468	0.8468	0.8423
	3	0.9437	0.9505	0.9302	0.9077	0.9009	0.8941	0.9032	0.8851	0.8874
	5	0.9437	0.9505	0.9212	0.9099	0.9009	0.8941	0.9032	0.9032	0.9054
	10	0.9437	0.9505	0.9392	0.9414	0.9392	0.9392	0.9392	0.9392	0.9414
	20	0.9414	0.9414	0.9279	0.9279	0.9347	0.9279	0.9279	0.9279	0.9279
	25	0.9324	0.9324	0.9324	0.9189	0.9324	0.9324	0.9189	0.9324	0.9324
	30	0.9212	0.9212	0.9144	0.9144	0.9144	0.9144	0.9144	0.9144	0.9144
	35	0.9212	0.9212	0.9212	0.9212	0.9212	0.9212	0.9212	0.9212	0.9212
	40	0.9279	0.9279	0.9279	0.9279	0.9279	0.9279	0.9279	0.9279	0.9279
	50	0.9279	0.9279	0.9279	0.9279	0.9279	0.9279	0.9279	0.9279	0.9279

Table 9. Modeling: Holdout, Numerical Methods.

Materials (holdout)										
<i>max_depth</i>										
<i>min_samples_leaf</i>		3	5	7	10	12	14	16	18	20
	2	0.8536	0.7883	0.7477	0.6914	0.6959	0.7005	0.705	0.7027	0.6779
	3	0.8536	0.7883	0.7635	0.705	0.7252	0.714	0.7072	0.7117	0.7185
	5	0.8536	0.7883	0.7477	0.741	0.7342	0.7297	0.7275	0.723	0.7185
	10	0.8536	0.786	0.7162	0.7297	0.7117	0.7185	0.723	0.714	0.7162
	20	0.8536	0.7995	0.7725	0.7725	0.777	0.7725	0.7725	0.7725	0.7725
	25	0.8536	0.7995	0.7838	0.7748	0.7748	0.7748	0.7748	0.7748	0.7748
	30	0.8536	0.8086	0.7838	0.7928	0.7928	0.7838	0.7838	0.7928	0.7928
	35	0.8536	0.8221	0.8131	0.8131	0.8041	0.8131	0.8131	0.8131	0.8041
	40	0.8536	0.7905	0.7838	0.777	0.786	0.777	0.7838	0.777	0.777
	50	0.6937	0.7117	0.6982	0.6982	0.6982	0.6982	0.6982	0.6982	0.6982

Table 10. Modeling: Holdout, Materials.

Differential Equations (holdout)										
<i>max_depth</i>										
<i>min_samples_leaf</i>		3	5	7	10	12	14	16	18	20
	2	0.964	0.9144	0.8896	0.8041	0.7635	0.7568	0.7455	0.7432	0.75
	3	0.964	0.9144	0.8874	0.795	0.7995	0.7928	0.7793	0.786	0.7815
	5	0.964	0.9144	0.8986	0.8198	0.8153	0.8131	0.8108	0.8086	0.8131
	10	0.964	0.9257	0.8874	0.8581	0.8649	0.8581	0.8559	0.8581	0.8581
	20	0.964	0.9257	0.9009	0.9009	0.9009	0.9009	0.9009	0.9009	0.9009
	25	0.964	0.9257	0.9032	0.9032	0.9032	0.9032	0.9032	0.9032	0.9032
	30	0.964	0.9257	0.9144	0.9144	0.9144	0.9144	0.9144	0.9144	0.9144
	35	0.964	0.9257	0.9144	0.9144	0.9144	0.9144	0.9144	0.9144	0.9144
	40	0.964	0.9257	0.9144	0.9144	0.9144	0.9144	0.9144	0.9144	0.9144
	50	0.964	0.8941	0.8919	0.8919	0.8919	0.8919	0.8919	0.8919	0.8919

Table 11. Modeling: Holdout, Differential Equations.

Informatics (holdout)										
<i>max_depth</i>										
<i>min_samples_leaf</i>		3	5	7	10	12	14	16	18	20
	2	0.8491	0.8986	0.8288	0.8086	0.777	0.7725	0.786	0.7838	0.7905
	3	0.8491	0.8986	0.8356	0.8018	0.7973	0.795	0.7995	0.8018	0.7883
	5	0.8491	0.9009	0.8536	0.8041	0.8063	0.7815	0.7905	0.8108	0.7883
	10	0.9144	0.8964	0.8491	0.8221	0.8198	0.8198	0.8198	0.8198	0.8198
	20	0.9144	0.9032	0.8896	0.8896	0.8896	0.8896	0.8896	0.8896	0.8896
	25	0.9144	0.8806	0.8761	0.8761	0.8761	0.8761	0.8761	0.8761	0.8761
	30	0.9144	0.8806	0.8716	0.8716	0.8716	0.8716	0.8716	0.8716	0.8716
	35	0.9144	0.8829	0.8739	0.8739	0.8739	0.8739	0.8739	0.8739	0.8739
	40	0.9144	0.9077	0.8986	0.8986	0.8986	0.8986	0.8986	0.8986	0.8986
	50	0.9144	0.8829	0.8829	0.8829	0.8829	0.8829	0.8829	0.8829	0.8829

Table 12. Modeling: Holdout, Informatics.

Mechanics (holdout)										
<i>max_depth</i>										
<i>min_samples_leaf</i>		3	5	7	10	12	14	16	18	20
	2	0.6622	0.5766	0.518	0.5293	0.518	0.5203	0.5225	0.509	0.518
	3	0.6622	0.5698	0.5158	0.5586	0.5405	0.5563	0.5518	0.5338	0.5473
	5	0.6622	0.5743	0.5113	0.536	0.518	0.5203	0.536	0.5225	0.5315
	10	0.6622	0.5968	0.5608	0.5541	0.5631	0.5608	0.5631	0.5608	0.5608
	20	0.6622	0.6104	0.5518	0.5315	0.5315	0.5315	0.5315	0.5315	0.5315
	25	0.6622	0.6126	0.5338	0.5428	0.5428	0.5428	0.5428	0.5428	0.5428
	30	0.6622	0.6126	0.527	0.527	0.527	0.527	0.527	0.527	0.5338
	35	0.6622	0.6216	0.5428	0.5518	0.5518	0.5518	0.5518	0.5518	0.5518
	40	0.6622	0.6216	0.5541	0.5541	0.5541	0.5541	0.5541	0.5541	0.5541
	50	0.6644	0.6239	0.6036	0.6036	0.6036	0.6036	0.6036	0.6036	0.6036

Table 13. Modeling: holdout, Mechanics.

Step 2:

Finding the same tables through stratified k-fold cross validation:

Electromagnetics (Stratified k-CV)										
<i>max_depth</i>										
<i>min_samples_leaf</i>		3	5	7	10	12	14	16	18	20
	2	0.7125	0.7037	0.6914	0.6554	0.6387	0.6352	0.6497	0.6448	0.6378
	3	0.713	0.7073	0.6967	0.665	0.6488	0.6545	0.6479	0.6505	0.6452
	5	0.713	0.7081	0.6927	0.6681	0.6576	0.6576	0.6607	0.6598	0.6668
	10	0.7139	0.716	0.7015	0.676	0.6712	0.6694	0.6672	0.6703	0.6721
	20	0.7134	0.7112	0.6936	0.6831	0.6853	0.6844	0.6848	0.6835	0.6866
	25	0.7134	0.716	0.6962	0.691	0.6897	0.6897	0.6897	0.6914	0.6901
	30	0.7139	0.7143	0.6967	0.691	0.687	0.6883	0.6901	0.6892	0.687
	35	0.7143	0.7213	0.7033	0.6976	0.6976	0.6976	0.6971	0.6976	0.6976
	40	0.7143	0.7204	0.7073	0.7002	0.7015	0.7011	0.7015	0.6998	0.7015
	50	0.7147	0.7112	0.7016	0.6985	0.6985	0.6993	0.6993	0.6993	0.6985

Table 14. Modeling: Stratified k-CV, Electromagnetics.

Numerical Methods (Stratified k-CV)										
<i>max_depth</i>										
<i>min_samples_leaf</i>		3	5	7	10	12	14	16	18	20
	2	0.9108	0.8932	0.8782	0.8496	0.8457	0.8378	0.8369	0.8316	0.8294
	3	0.9108	0.8958	0.8831	0.8571	0.8505	0.8475	0.8514	0.8501	0.8461
	5	0.9103	0.8998	0.8866	0.8725	0.8725	0.8699	0.873	0.8664	0.8708
	10	0.9099	0.9015	0.8888	0.8817	0.8892	0.8866	0.8875	0.8831	0.8839
	20	0.9077	0.9037	0.8976	0.8967	0.8967	0.8967	0.8967	0.8967	0.8967
	25	0.9086	0.9068	0.9046	0.9046	0.9046	0.9046	0.9046	0.9046	0.9046
	30	0.9086	0.9068	0.9068	0.9068	0.9068	0.9068	0.9068	0.9068	0.9068
	35	0.9086	0.9077	0.9077	0.9077	0.9077	0.9077	0.9077	0.9077	0.9077
	40	0.9125	0.9125	0.9125	0.9125	0.9125	0.9125	0.9125	0.9125	0.9125
	50	0.9121	0.9121	0.9121	0.9121	0.9121	0.9121	0.9121	0.9121	0.9121

Table 15. Modeling: Stratified k-CV, Numerical Methods.

Materials (Stratified k-CV)										
<i>max_depth</i>										
<i>min_samples_leaf</i>		3	5	7	10	12	14	16	18	20
	2	0.7662	0.7547	0.7486	0.7248	0.7112	0.6954	0.6953	0.687	0.6839
	3	0.7662	0.7573	0.7499	0.7336	0.7173	0.7059	0.7116	0.7063	0.7046
	5	0.7662	0.7591	0.753	0.7433	0.7306	0.7209	0.7182	0.7191	0.7186
	10	0.7662	0.76	0.7552	0.7389	0.7314	0.7231	0.7239	0.7209	0.7204
	20	0.7653	0.7565	0.7499	0.7367	0.738	0.7371	0.7402	0.7363	0.7424
	25	0.7653	0.76	0.7569	0.7508	0.7508	0.7543	0.7538	0.7508	0.7512
	30	0.7653	0.7622	0.7538	0.7477	0.7486	0.7477	0.7477	0.7477	0.7481
	35	0.7653	0.7591	0.7477	0.7473	0.7481	0.7477	0.7481	0.7477	0.7481
	40	0.7653	0.7613	0.7503	0.7486	0.7486	0.7486	0.7486	0.7486	0.7486
	50	0.7653	0.7618	0.7556	0.7534	0.7534	0.7534	0.7534	0.7534	0.7534

Table 16. Modeling: Stratified k-CV, Materials.

Differential Equations (Stratified k-CV)										
<i>max_depth</i>										
<i>min_samples_leaf</i>		3	5	7	10	12	14	16	18	20
	2	0.8255	0.8119	0.7982	0.7657	0.745	0.738	0.723	0.7248	0.7331
	3	0.8255	0.8101	0.7965	0.7609	0.7472	0.7424	0.738	0.7428	0.7389
	5	0.8255	0.811	0.797	0.7706	0.7618	0.7578	0.7569	0.7539	0.7574
	10	0.8255	0.8163	0.8066	0.7741	0.7696	0.7639	0.7652	0.7683	0.7683
	20	0.8255	0.8229	0.811	0.793	0.7934	0.7934	0.7925	0.7947	0.7947
	25	0.8264	0.8277	0.8246	0.8185	0.8185	0.8185	0.8185	0.8185	0.8185
	30	0.8268	0.8264	0.8215	0.8215	0.8215	0.8215	0.8215	0.8215	0.8215
	35	0.8295	0.8251	0.8238	0.8215	0.8215	0.8215	0.8215	0.8215	0.8215
	40	0.8303	0.8273	0.8238	0.8238	0.8238	0.8238	0.8238	0.8238	0.8238
	50	0.8317	0.8282	0.8242	0.8242	0.8242	0.8242	0.8242	0.8242	0.8242

Table 17. Modeling: Stratified k-CV, Differential Equations.

Informatics (Stratified k-CV)										
<i>max_depth</i>										
<i>min_samples_leaf</i>		3	5	7	10	12	14	16	18	20
	2	0.8251	0.8158	0.7824	0.7587	0.7451	0.7398	0.7363	0.7398	0.739
	3	0.8255	0.8158	0.782	0.7631	0.7499	0.7535	0.7543	0.7477	0.7574
	5	0.8255	0.8158	0.7873	0.7666	0.7618	0.76	0.7582	0.7649	0.7587
	10	0.8255	0.8237	0.7965	0.7838	0.7829	0.7838	0.7824	0.7869	0.7838
	20	0.8255	0.8286	0.8101	0.8079	0.8075	0.8079	0.8079	0.8075	0.8079
	25	0.8255	0.8251	0.8172	0.8189	0.8189	0.8181	0.8189	0.8203	0.8203
	30	0.829	0.8264	0.822	0.8229	0.8238	0.8233	0.8238	0.8233	0.8238
	35	0.8317	0.8299	0.8317	0.8303	0.8303	0.8308	0.8303	0.8308	0.8303
	40	0.8317	0.8294	0.8294	0.8294	0.8294	0.8294	0.8294	0.8294	0.8294
	50	0.8312	0.8303	0.8303	0.8303	0.8303	0.8303	0.8303	0.8303	0.8303

Table 18. Modeling: Stratified k-CV, Informatics.

Mechanics (Stratified k-CV)										
<i>max_depth</i>										
<i>min_samples_leaf</i>		3	5	7	10	12	14	16	18	20
	2	0.6774	0.6686	0.6638	0.622	0.6198	0.6137	0.6093	0.6088	0.6049
	3	0.6774	0.6699	0.6602	0.6304	0.6198	0.6202	0.6194	0.6158	0.6141
	5	0.6774	0.6673	0.6611	0.6269	0.6097	0.6124	0.6203	0.6181	0.6146
	10	0.6774	0.6734	0.6646	0.644	0.6418	0.6405	0.6444	0.6449	0.6405
	20	0.6774	0.6782	0.6716	0.651	0.6523	0.6536	0.6554	0.6514	0.6514
	25	0.6774	0.6769	0.6699	0.6615	0.662	0.662	0.662	0.6611	0.662
	30	0.6774	0.6725	0.6686	0.6642	0.6629	0.6624	0.6642	0.6642	0.6624
	35	0.6774	0.6752	0.6743	0.6655	0.6655	0.6655	0.6655	0.6646	0.6655
	40	0.6774	0.673	0.6673	0.666	0.6655	0.666	0.666	0.6655	0.6655
	50	0.6774	0.6717	0.6717	0.669	0.669	0.669	0.669	0.669	0.6703

Table 19. Modeling: Stratified k-CV, Mechanics.

Step 3:

We have created a function that takes an array of values containing all the values of the tables shown in steps 1 and 2 and returns the statistical parameters explained. It returns the following table of data:

Subject	<i>max_value</i>		<i>mean_value</i>		<i>row_std</i>		<i>column_std</i>	
	Holdout	kcross	Holdout	kcross	Holdout	kcross	Holdout	kcross
Electrom.	0.7117	0.7213	0.6671	0.6887	0.0227	0.0143	0.015	0.0148
Num. Meth.	0.9505	0.9125	0.9219	0.8937	0.0166	0.0191	0.0091	0.008
Materials	0.8536	0.7662	0.7742	0.7432	0.0278	0.0124	0.032	0.0125
Diff. Equa.	0.9640	0.8317	0.8880	0.8010	0.0371	0.0247	0.0337	0.0144
Informatics	0.9144	0.8316	0.863	0.8056	0.0323	0.0249	0.0197	0.0116
Mechanics	0.6644	0.6782	0.5652	0.6555	0.0200	0.0151	0.0391	0.0125

Table 20. Results for holdout vs. stratified k-CV.

As *Table 20* shows, there is a considerable difference between both methods. As we see the fractions of correct guesses tend to find considerable differences for both “max_value” and “mean_value”, which gives further proof of what we had anticipated.

But *Table 20* gives us much more information than that. We see, for example, that the standard deviations we have found are considerably different for many subjects.

This tells us that the results found using the holdout method have a tendency to being more dependent on us varying these parameters and finding the right combination of them. While in the case of stratified k-cross validation the effect of setting a certain “max_depth” or “min_samples_leaf” is not so great. This is another pro to use stratified k-fold cross validation because if the results are not so dependent on these values, the consequences of not finding the right combination of these two parameters (imagine for example that the combinations we generate do not contain the optimal one) would not be so great.

This claim is only further sustained if we compare the “max_value” and the “mean_value” of each method separately. See that the difference between them is greater for holdout than it is for stratified k-cross validation. Meaning that changing “max_depth” and “min_samples_leaf” can have a greater impact on the first method.

Note that despite all of this, if we sort the subjects by their difficulty to be predicted with both methods (how many correct predictions they get), the order is quite similar with both methods. This suggests that despite hold out obtains significantly different results from the stratified k-CV, we can still suggest the difficulty to predict each subject with holdout.

Conclusion of the tests

As enough proof has been found to sustain that the holdout method can be problematic with the data we have, it has decided that from this point on stratified k-fold cross validation will be the only validation method we will use. By creating n stratified splits and using them all for training and testing we make sure that the data we use for the testing is representative of the overall data.

5.5. Deciding whether to keep or drop the students of 2016

We have already decided that, from this point on, only stratified k-fold cross validation will be used.

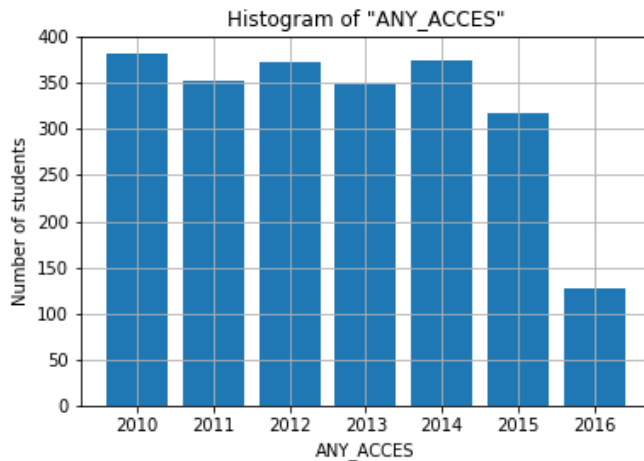


Figure 13. Histogram of the year of entry of students at ETSEIB.

As we see in *Figure 13*, the number of students of 2016 is critically smaller than the number of students for the rest of the years. We only have 127 students for 2016 whereas for the rest of the years we always have over 300.

We think that this could be because the students of this year had not had as much time as the rest to pass Q1 and Q2 when the data was gathered. This goes along with the fact that we only have students with “N_QUAD_FASEIN” equal to 2 for the year 2016 (what we saw in *Table 7*). In other words, students of 2016 would have only had the chance to take all the Q3 subjects if they had passed Q1 and Q2 in 2 semesters, which means they are good students with a performance over the average.

We would like to know if with our new verification method keeping or dropping these students can have a big impact on the results. If that is the case, we should analyze the reasons and decide if it is necessary to drop them.

Test 1

This test will conclude if it is better to keep the 2016 students or if they cause problems to the model and thus it is preferable to drop them.

Using the same combination of “max_depth” and “mean_samples_leaf” we generate a new table of results but having erased the students who entered the school in 2016. That means dropping a total of 127 rows of data.

Despite we already have data for the case with 2016 students we generate new results to check that they are in the same range of values as the ones in the last test. We use

the same *random_state* value for both data frames to reduce the impact of randomness on the results.

Due to the similarity of the results found while keeping those students with the ones of the previous section the tables of results are not shown, considering them redundant.

Nonetheless, the tables found while dropping those students are new, so they are shown.

They are the following ones:

Electromagnetics (without 2016)										
<i>max_depth</i>										
<i>min_samples_leaf</i>		3	5	7	10	12	14	16	18	20
	2	0.7044	0.6913	0.6797	0.6322	0.6368	0.6215	0.6215	0.6303	0.6224
	3	0.7044	0.6936	0.6848	0.6475	0.642	0.635	0.6401	0.6382	0.6401
	5	0.7048	0.6899	0.6759	0.6438	0.6289	0.6369	0.6359	0.6392	0.6364
	10	0.7048	0.6964	0.6825	0.6518	0.6509	0.6495	0.6527	0.6541	0.6513
	20	0.7048	0.6895	0.6722	0.6499	0.6476	0.6452	0.6457	0.6457	0.6448
	25	0.7048	0.682	0.675	0.6648	0.661	0.661	0.6615	0.6629	0.6615
	30	0.7072	0.6862	0.6774	0.6671	0.6648	0.6648	0.6657	0.6662	0.6653
	35	0.7076	0.6946	0.6815	0.6745	0.6722	0.6722	0.6722	0.6722	0.6722
	40	0.7067	0.6909	0.6787	0.6773	0.6769	0.6769	0.6773	0.6773	0.6773
	50	0.7081	0.6964	0.6862	0.6815	0.6815	0.6815	0.6815	0.6815	0.6815

Table 21. Modeling: Stratified k-CV without 2016 students, Electromagnetics.

Numerical methods (without 2016)										
<i>max_depth</i>										
<i>min_samples_leaf</i>		3	5	7	10	12	14	16	18	20
	2	0.9008	0.8887	0.8785	0.844	0.8291	0.824	0.8208	0.8156	0.8189
	3	0.9013	0.8906	0.8752	0.8561	0.8459	0.8408	0.8412	0.8459	0.8389
	5	0.9008	0.8906	0.8799	0.8571	0.8538	0.8501	0.8529	0.8538	0.8557
	10	0.9027	0.8939	0.8915	0.8855	0.8817	0.8836	0.8831	0.885	0.8836
	20	0.9027	0.9013	0.8976	0.8976	0.8976	0.8976	0.8976	0.8976	0.8976
	25	0.9018	0.9022	0.8999	0.9008	0.9008	0.8999	0.8999	0.8999	0.9008
	30	0.9004	0.9008	0.9008	0.9008	0.9008	0.9008	0.9008	0.9008	0.9008
	35	0.9008	0.9013	0.9013	0.9013	0.9013	0.9013	0.9013	0.9013	0.9013
	40	0.9004	0.9004	0.8999	0.9004	0.8999	0.9004	0.9004	0.8999	0.9004
	50	0.9032	0.9032	0.9032	0.9032	0.9032	0.9032	0.9032	0.9032	0.9032

Table 22. Modeling: Stratified k-CV without 2016 students, Numerical Methods.

Materials (without 2016)										
<i>max_depth</i>										
<i>min_samples_leaf</i>		3	5	7	10	12	14	16	18	20
	2	0.7463	0.7375	0.7346	0.6997	0.6811	0.6751	0.6732	0.677	0.6746
	3	0.7463	0.7356	0.7342	0.71	0.6979	0.6998	0.6848	0.6848	0.6825
	5	0.7468	0.7346	0.7351	0.7118	0.6988	0.6932	0.6876	0.6909	0.6858
	10	0.7537	0.7444	0.7328	0.7193	0.7132	0.7123	0.7118	0.7105	0.7118
	20	0.7537	0.7477	0.7407	0.7314	0.7296	0.7296	0.7319	0.7305	0.731
	25	0.7547	0.7505	0.7547	0.7468	0.7477	0.7482	0.7472	0.7468	0.7477
	30	0.7547	0.7523	0.7533	0.7491	0.7486	0.7481	0.75	0.7505	0.7481
	35	0.7547	0.7482	0.7542	0.7565	0.7547	0.7542	0.7542	0.7565	0.7565
	40	0.7547	0.7449	0.7463	0.7453	0.7453	0.7439	0.7439	0.7439	0.743
	50	0.7547	0.75	0.743	0.737	0.737	0.737	0.737	0.737	0.737

Table 23. Modeling: Stratified k-CV without 2016 students, Materials.

Differential equations (without 2016)										
<i>max_depth</i>										
<i>min_samples_leaf</i>		3	5	7	10	12	14	16	18	20
	2	0.8166	0.804	0.7933	0.7602	0.7444	0.7272	0.7253	0.7286	0.7281
	3	0.8166	0.8045	0.7933	0.7691	0.7477	0.7458	0.7426	0.7435	0.7379
	5	0.8166	0.8026	0.7984	0.7672	0.7589	0.7524	0.7547	0.7524	0.7538
	10	0.8166	0.8105	0.7891	0.7621	0.7533	0.7523	0.7514	0.7556	0.7547
	20	0.8166	0.8087	0.7979	0.7812	0.7807	0.7807	0.7803	0.7803	0.7789
	25	0.818	0.8124	0.8035	0.7989	0.7984	0.7984	0.7989	0.7984	0.7984
	30	0.8184	0.818	0.8157	0.8129	0.8129	0.8129	0.8129	0.8129	0.8129
	35	0.8166	0.8157	0.8115	0.8115	0.8115	0.8115	0.8115	0.8115	0.8115
	40	0.8161	0.8119	0.8063	0.8063	0.8063	0.8063	0.8063	0.8063	0.8063
	50	0.8166	0.8091	0.8068	0.8068	0.8068	0.8068	0.8068	0.8068	0.8068

Table 24. Modeling: Stratified k-CV without 2016 students, Differential Equations.

Informatics (without 2016)										
<i>max_depth</i>										
<i>min_samples_leaf</i>		3	5	7	10	12	14	16	18	20
	2	0.8208	0.8143	0.7891	0.7584	0.7449	0.7491	0.7458	0.7398	0.743
	3	0.8208	0.8142	0.7956	0.7612	0.7542	0.7584	0.7579	0.7565	0.7523
	5	0.8208	0.8142	0.7938	0.7705	0.7626	0.7663	0.7667	0.7691	0.7621
	10	0.8208	0.8161	0.8017	0.784	0.7845	0.7877	0.7835	0.7844	0.783
	20	0.8212	0.817	0.803	0.8012	0.8016	0.8007	0.8012	0.8007	0.8012
	25	0.8212	0.8175	0.8063	0.8077	0.8058	0.8058	0.8063	0.8063	0.8058
	30	0.8212	0.818	0.8114	0.8114	0.8114	0.8114	0.8114	0.8114	0.8114
	35	0.8231	0.8203	0.8175	0.8175	0.8175	0.8175	0.8175	0.8175	0.8175
	40	0.8231	0.8207	0.8189	0.8189	0.8193	0.8189	0.8193	0.8193	0.8189
	50	0.8245	0.8222	0.8222	0.8222	0.8222	0.8222	0.8222	0.8222	0.8222

Table 25. Modeling: Stratified k-CV without 2016 students, Informatics.

Mechanics (without 2016)										
max_depth										
min_samples_leaf		3	5	7	10	12	14	16	18	20
	2	0.6387	0.6439	0.6304	0.6085	0.5875	0.5848	0.5884	0.588	0.5843
	3	0.6387	0.643	0.6285	0.6127	0.6052	0.5964	0.5945	0.5861	0.5969
	5	0.6387	0.6439	0.6364	0.6196	0.6224	0.6145	0.615	0.6192	0.6206
	10	0.6387	0.6495	0.6448	0.6322	0.628	0.6243	0.6261	0.6238	0.6275
	20	0.6387	0.6527	0.6485	0.6304	0.6304	0.6313	0.6299	0.6313	0.6294
	25	0.6383	0.6495	0.6537	0.6411	0.642	0.6425	0.6411	0.642	0.642
	30	0.6383	0.6439	0.6513	0.6392	0.6401	0.6406	0.6392	0.6397	0.6392
	35	0.6383	0.6453	0.6425	0.6364	0.636	0.6364	0.636	0.6355	0.635
	40	0.6383	0.6457	0.649	0.6406	0.6406	0.6406	0.6411	0.6411	0.6406
	50	0.6392	0.6383	0.6392	0.6373	0.6373	0.6373	0.6373	0.6373	0.6373

Table 26. Modeling: Stratified k-CV without 2016 students, Mechanics.

We already have a function that calculates the basic statistical parameters for these tables in the previous section which we are going to use with our new tables.

Another reason to use the same function is that we want to use the same evaluation method as in the previous section so that the orders of magnitude can also be put to perspective with those of the previous section.

The statistical parameters we have found for both cases are:

Subject	max_value		mean_value		row_std		column_std	
	Keep	Drop	Keep	Drop	Keep	Drop	Keep	Drop
Electrom.	0.7222	0.7081	0.6929	0.6677	0.0122	0.0135	0.0157	0.0187
Num. Meth.	0.9094	0.9032	0.8926	0.8863	0.0166	0.0204	0.0072	0.0081
Materials	0.7701	0.7565	0.7453	0.7319	0.0154	0.019	0.0108	0.0116
Diff. Equa.	0.8299	0.8184	0.8029	0.7907	0.0213	0.0202	0.0136	0.0142
Informatics	0.8356	0.8245	0.8103	0.8002	0.0191	0.0196	0.0117	0.0114
Mechanics	0.6765	0.6537	0.6497	0.6318	0.0161	0.0124	0.0118	0.0084

Table 27. Results on deciding whether to keep 2016 students.

The results show that keeping or dropping the 2016 students only changes the results to a small degree. The variation for the max_value of each subject oscillates between around 2.3% to under 1%, in all cases getting a lower “fraction of correct guesses” when dropping the students of 2016.

The mean values follow the same tendency.

This is a good sign that the data is shuffled correctly, because dropping a sample of the best performing students does not change the final results so much.

When it comes to the values of standard deviation for both row_std and columns_std we find that they are effectively equal in both cases. This is a good sign because in theory

the variability should not be very affected by dropping such a number of rows as long as we keep using the same method, as we are doing here.

Conclusion of the tests

As we have said, there is no considerable difference between the results with or without the students with “ANY_ACCES” of 2016. Therefore, we conclude that no numerical proof has been found to defend that students of 2016 should be dropped.

We take the decision to keep all the rows of data we have; the same ones the data frame of the PDMP had.

5.6. Comparing several data frames

In the last section we found numerical evidence that the data frame used in the last project was a good choice in terms of rows. Now, we would like to know if the results can be improved by modifying the number of columns.

Test 1

This section will test the three data frames that we have created using stratified k-fold cross validation for all of them.

Note that because we want to compare the effect of modifying the number of columns for the of “dfanterior”, we can only use data frames that have the same number of rows. This has two consequences. On the first hand, our students with “unattempted” cannot be used, because they would modify the number of rows. On the other hand, we must drop the 45 extra rows that “df0” has that “dfanterior” and “dfall” lost because of the data enhancement.

This leaves us with three data frames of 2275 rows and varying number of columns.

The same procedure as in the last two sections is followed in the sense that the tables of results are found for the same combination of “max_depth” and “min_samples_leaf” and then the results are compared using the same statistical parameters as in the other two sections.

As the results for “dfanterior” have already been found (they are the same ones as in section one with stratified k-fold cross validation), their tables are not shown in this section, to avoid redundancy.

For “df0” we have the following tables:

Electromagnetics (df0)										
max_depth										
min_samples_leaf		3	5	7	10	12	14	16	18	20
	2	0.7011	0.6945	0.6717	0.6568	0.6343	0.6348	0.6339	0.6299	0.6251
	3	0.7011	0.6967	0.6769	0.6629	0.6488	0.6435	0.6378	0.6444	0.6409
	5	0.7011	0.6954	0.6822	0.6647	0.651	0.6479	0.6558	0.6541	0.6541
	10	0.7011	0.6989	0.6786	0.6685	0.6584	0.6615	0.6593	0.6589	0.662
	20	0.7006	0.7041	0.7006	0.6958	0.6949	0.6936	0.6936	0.6927	0.6936
	25	0.7006	0.705	0.7002	0.6976	0.6971	0.6967	0.6967	0.6962	0.6971
	30	0.6997	0.705	0.7028	0.6998	0.7006	0.6984	0.6984	0.7006	0.7006
	35	0.7002	0.698	0.7033	0.6998	0.6993	0.6993	0.6993	0.6993	0.6993
	40	0.6984	0.6954	0.698	0.698	0.6976	0.6976	0.6976	0.6976	0.6976
	50	0.694	0.6958	0.6901	0.6892	0.6901	0.6892	0.6901	0.6901	0.6892

Table 28. Modeling: Stratified k-CV with df0, Electromagnetics.

Numerical Methods (df0)										
max_depth										
min_samples_leaf		3	5	7	10	12	14	16	18	20
	2	0.9081	0.9024	0.895	0.8541	0.8391	0.829	0.8338	0.8343	0.8281
	3	0.9081	0.9042	0.8954	0.8637	0.8567	0.851	0.8466	0.8483	0.8488
	5	0.9086	0.9042	0.8993	0.8747	0.8743	0.8725	0.8712	0.8677	0.8703
	10	0.9103	0.9029	0.8976	0.8901	0.8897	0.8901	0.8897	0.8879	0.8897
	20	0.9095	0.9077	0.9077	0.9059	0.9059	0.9059	0.9059	0.9059	0.9059
	25	0.9081	0.9081	0.9081	0.9081	0.9081	0.9081	0.9081	0.9081	0.9081
	30	0.9099	0.9099	0.9099	0.9099	0.9099	0.9099	0.9099	0.9099	0.9099
	35	0.9117	0.9117	0.9117	0.9117	0.9117	0.9117	0.9117	0.9117	0.9117
	40	0.9117	0.9117	0.9117	0.9117	0.9117	0.9117	0.9117	0.9117	0.9117
	50	0.913	0.913	0.913	0.913	0.913	0.913	0.913	0.913	0.913

Table 29. Modeling: Stratified k-CV with df0, Numerical Methods.

Materials (df0)										
max_depth										
min_samples_leaf		3	5	7	10	12	14	16	18	20
	2	0.7627	0.7587	0.738	0.6998	0.6901	0.6901	0.6897	0.687	0.691
	3	0.7627	0.7591	0.7376	0.7064	0.6945	0.6892	0.6919	0.6888	0.6883
	5	0.7627	0.7617	0.7424	0.7178	0.7099	0.709	0.7169	0.716	0.7152
	10	0.7631	0.7697	0.7424	0.7275	0.72	0.7231	0.7182	0.7187	0.7191
	20	0.7631	0.7661	0.7565	0.7565	0.7547	0.7538	0.7547	0.7534	0.7547
	25	0.7613	0.7644	0.7574	0.7569	0.7574	0.7569	0.7565	0.7574	0.7569
	30	0.7596	0.7635	0.7582	0.7582	0.7582	0.7582	0.7582	0.7582	0.7582
	35	0.7569	0.76	0.7543	0.753	0.753	0.753	0.753	0.753	0.753
	40	0.7631	0.7635	0.7556	0.7539	0.7539	0.7539	0.7539	0.7539	0.7539
	50	0.7649	0.767	0.7666	0.7666	0.7666	0.7666	0.7666	0.7666	0.7666

Table 30. Modeling: Stratified k-CV with df0, Materials.

Differential Equations (df0)										
max_depth										
min_samples_leaf		3	5	7	10	12	14	16	18	20
	2	0.8237	0.8141	0.7996	0.7534	0.7415	0.7332	0.7239	0.727	0.727
	3	0.8237	0.8158	0.8022	0.7622	0.749	0.749	0.7389	0.7442	0.7468
	5	0.8237	0.8163	0.8075	0.7754	0.7617	0.7503	0.745	0.7503	0.7472
	10	0.8246	0.8185	0.8088	0.7811	0.7784	0.7754	0.7736	0.7732	0.7754
	20	0.8246	0.8185	0.8022	0.7916	0.7934	0.7943	0.793	0.7952	0.7934
	25	0.8286	0.8211	0.8132	0.8132	0.8127	0.8127	0.8141	0.8127	0.8141
	30	0.8286	0.8229	0.8167	0.8163	0.8167	0.8167	0.8167	0.8167	0.8163
	35	0.8259	0.8194	0.8158	0.8154	0.8158	0.8154	0.8158	0.8158	0.8158
	40	0.8273	0.8193	0.818	0.818	0.818	0.818	0.818	0.818	0.818
	50	0.8259	0.8171	0.8171	0.8171	0.8171	0.8171	0.8171	0.8171	0.8171

Table 31. Modeling: Stratified k-CV with df0, Differential Equations.

Informatics (df0)										
max_depth										
min_samples_leaf		3	5	7	10	12	14	16	18	20
	2	0.8312	0.8215	0.8071	0.7745	0.7618	0.738	0.7472	0.7402	0.7402
	3	0.8312	0.8229	0.8049	0.7653	0.7504	0.7332	0.7433	0.7429	0.7437
	5	0.8317	0.8224	0.8079	0.7767	0.76	0.7627	0.7679	0.7609	0.7587
	10	0.8334	0.8264	0.8105	0.7912	0.7877	0.7894	0.7886	0.7877	0.7881
	20	0.8352	0.829	0.8149	0.8149	0.8149	0.8149	0.8145	0.8145	0.8145
	25	0.8361	0.829	0.8202	0.8189	0.8189	0.8189	0.8189	0.8189	0.8189
	30	0.8347	0.8273	0.822	0.8202	0.8207	0.8207	0.8202	0.8202	0.8202
	35	0.8347	0.8272	0.8242	0.8224	0.8224	0.8224	0.8224	0.8224	0.8224
	40	0.833	0.8241	0.8237	0.8237	0.8237	0.8237	0.8237	0.8237	0.8237
	50	0.8325	0.8228	0.8193	0.8193	0.8193	0.8193	0.8193	0.8193	0.8193

Table 32. Modeling: Stratified k-CV with df0, Informatics.

Mechanics (df0)										
max_depth										
min_samples_leaf		3	5	7	10	12	14	16	18	20
	2	0.673	0.6756	0.6682	0.6488	0.6488	0.626	0.6352	0.6295	0.6281
	3	0.673	0.6769	0.6668	0.6532	0.6453	0.6395	0.6413	0.6369	0.6422
	5	0.673	0.6748	0.6624	0.6435	0.6374	0.6255	0.6325	0.6312	0.629
	10	0.6726	0.6726	0.6642	0.6524	0.6471	0.6493	0.6497	0.6493	0.6479
	20	0.6739	0.6739	0.6642	0.6541	0.6532	0.6484	0.6484	0.6497	0.6492
	25	0.6734	0.6743	0.6651	0.6624	0.6616	0.6607	0.6616	0.6607	0.6616
	30	0.6734	0.6664	0.6585	0.6541	0.655	0.6558	0.655	0.6558	0.655
	35	0.6734	0.6743	0.6699	0.6664	0.6668	0.6668	0.6668	0.6668	0.6668
	40	0.6734	0.6677	0.6642	0.6638	0.6638	0.6638	0.6642	0.6638	0.6638
	50	0.6712	0.673	0.673	0.6752	0.6752	0.6752	0.6752	0.6748	0.6752

Table 33. Modeling: Stratified k-CV with df0, Mechanics.

For “dfall” the tables are:

Electromagnetics (dfall)										
max_depth										
min_samples_leaf		3	5	7	10	12	14	16	18	20
	2	0.7069	0.6985	0.6861	0.6782	0.6633	0.6584	0.6571	0.6527	0.6518
	3	0.7073	0.6976	0.6857	0.6804	0.6624	0.6593	0.658	0.6601	0.6566
	5	0.7073	0.7024	0.6853	0.6791	0.6672	0.6712	0.6619	0.6589	0.6628
	10	0.7073	0.7103	0.6963	0.6712	0.6695	0.6686	0.6699	0.6708	0.6681
	20	0.7064	0.7037	0.6976	0.6919	0.6901	0.6905	0.6892	0.6888	0.6905
	25	0.7064	0.7099	0.7042	0.7011	0.7011	0.7011	0.6989	0.7011	0.7006
	30	0.7064	0.7059	0.6984	0.6905	0.6927	0.6923	0.6927	0.6923	0.6927
	35	0.7064	0.7116	0.709	0.7037	0.7041	0.7041	0.7037	0.7037	0.7037
	40	0.7064	0.7099	0.709	0.709	0.709	0.7077	0.709	0.7077	0.7077
	50	0.706	0.7121	0.7068	0.7055	0.7055	0.7055	0.7055	0.7055	0.7055

Table 34. Modeling: Stratified k-CV with dfall, Electromagnetics.

Numerical Methods (dfall)										
max_depth										
min_samples_leaf		3	5	7	10	12	14	16	18	20
	2	0.9108	0.8954	0.8795	0.8505	0.8373	0.8329	0.8294	0.8215	0.8237
	3	0.9108	0.8971	0.8817	0.8642	0.8589	0.8532	0.8474	0.8466	0.8488
	5	0.9112	0.8998	0.8769	0.8699	0.862	0.8628	0.8628	0.8602	0.8624
	10	0.9117	0.9011	0.8888	0.8848	0.8835	0.887	0.884	0.8826	0.8875
	20	0.9086	0.9042	0.8998	0.8985	0.8985	0.8985	0.8985	0.8985	0.8985
	25	0.9095	0.9033	0.9024	0.8998	0.9007	0.8998	0.8998	0.9007	0.9007
	30	0.909	0.9077	0.9042	0.902	0.902	0.902	0.902	0.902	0.902
	35	0.909	0.9081	0.9081	0.9081	0.9081	0.9081	0.9081	0.9081	0.9081
	40	0.9121	0.9121	0.9121	0.9121	0.9121	0.9121	0.9121	0.9121	0.9121
	50	0.9117	0.9117	0.9117	0.9117	0.9117	0.9117	0.9117	0.9117	0.9117

Table 35. Modeling: Stratified k-CV with dfall, Numerical methods.

Materials (dfall)										
max_depth										
min_samples_leaf		3	5	7	10	12	14	16	18	20
	2	0.7662	0.7508	0.738	0.7134	0.6984	0.6958	0.6901	0.6932	0.6962
	3	0.7662	0.7534	0.7433	0.716	0.7147	0.7068	0.7094	0.7094	0.7059
	5	0.7662	0.7552	0.7446	0.7222	0.7217	0.7112	0.7081	0.7059	0.7094
	10	0.7662	0.7565	0.749	0.7275	0.7169	0.7147	0.7156	0.7169	0.7125
	20	0.7648	0.756	0.753	0.7464	0.7398	0.7433	0.7415	0.7415	0.7398
	25	0.7653	0.7631	0.7565	0.7472	0.749	0.749	0.7494	0.7499	0.749
	30	0.7653	0.7639	0.7538	0.749	0.749	0.7481	0.7481	0.7481	0.7481
	35	0.7653	0.7596	0.7508	0.7464	0.7459	0.7459	0.7459	0.7459	0.7446
	40	0.7653	0.7591	0.7512	0.7494	0.7494	0.7499	0.7494	0.7499	0.7494
	50	0.7653	0.7626	0.7582	0.7556	0.7552	0.7556	0.7552	0.7556	0.7552

Table 36. Modeling: Stratified k-CV with dfall, Materials.

Differential Equations (dfall)										
max_depth										
min_samples_leaf		3	5	7	10	12	14	16	18	20
	2	0.8238	0.8185	0.8017	0.7679	0.7477	0.738	0.7336	0.7301	0.7297
	3	0.8238	0.8163	0.7978	0.7723	0.7591	0.7534	0.7442	0.7446	0.7521
	5	0.8255	0.8167	0.8022	0.7781	0.7653	0.7622	0.7614	0.7613	0.7636
	10	0.8238	0.8198	0.8057	0.7771	0.774	0.7705	0.7718	0.7718	0.7714
	20	0.8238	0.8207	0.8158	0.8066	0.8053	0.8061	0.8066	0.8053	0.8061
	25	0.8246	0.8259	0.8229	0.8114	0.8123	0.8123	0.8123	0.8114	0.8123
	30	0.8268	0.826	0.8246	0.8224	0.8229	0.8224	0.8229	0.8224	0.8224
	35	0.8295	0.8242	0.8211	0.8211	0.8211	0.8211	0.8211	0.8211	0.8211
	40	0.8303	0.826	0.8198	0.8198	0.8198	0.8198	0.8198	0.8198	0.8198
	50	0.8317	0.8299	0.8264	0.8264	0.8264	0.8264	0.8264	0.8264	0.8264

Table 37. Modeling: Stratified k-CV with dfall, Differential Equations.

Informatics (dfall)										
max_depth										
min_samples_leaf		3	5	7	10	12	14	16	18	20
	2	0.8251	0.8132	0.7868	0.7543	0.7447	0.7434	0.7407	0.7332	0.7341
	3	0.8255	0.8141	0.7807	0.757	0.7429	0.742	0.7411	0.7376	0.7389
	5	0.8255	0.8132	0.7846	0.7666	0.7596	0.7609	0.7623	0.7653	0.7614
	10	0.8255	0.8207	0.797	0.786	0.7851	0.7847	0.786	0.7864	0.7864
	20	0.8255	0.8233	0.8009	0.7982	0.7996	0.7996	0.8004	0.7982	0.7987
	25	0.8255	0.8238	0.8123	0.8106	0.8119	0.8097	0.8097	0.8097	0.8119
	30	0.829	0.8246	0.8176	0.8185	0.8176	0.8185	0.8176	0.8176	0.8176
	35	0.8317	0.8325	0.833	0.8339	0.833	0.833	0.8339	0.833	0.833
	40	0.8317	0.8312	0.8303	0.8303	0.8303	0.8303	0.8303	0.8303	0.8303
	50	0.8312	0.8312	0.8312	0.8312	0.8312	0.8312	0.8312	0.8312	0.8312

Table 38. Modeling: Stratified k-CV with dfall, Informatics.

Mechanics (dfall)										
max_depth										
min_samples_leaf		3	5	7	10	12	14	16	18	20
	2	0.6774	0.6681	0.6554	0.6347	0.6194	0.6185	0.615	0.6163	0.6198
	3	0.6774	0.6695	0.6563	0.6325	0.6228	0.6211	0.6264	0.6185	0.6176
	5	0.6774	0.6686	0.6532	0.6246	0.6189	0.6167	0.6198	0.6202	0.622
	10	0.6774	0.6747	0.6589	0.644	0.6427	0.6423	0.6352	0.6383	0.6387
	20	0.6774	0.6782	0.6615	0.6519	0.6483	0.6501	0.6527	0.6497	0.6514
	25	0.6774	0.6769	0.6615	0.6519	0.6549	0.6541	0.6536	0.6545	0.6541
	30	0.6774	0.673	0.6637	0.6562	0.6562	0.6562	0.6562	0.6567	0.6567
	35	0.6774	0.6752	0.6695	0.6602	0.6602	0.6602	0.6602	0.6602	0.6602
	40	0.6774	0.673	0.6655	0.6624	0.6624	0.6624	0.6624	0.6611	0.662
	50	0.6774	0.6708	0.6681	0.6655	0.6642	0.6655	0.6655	0.6642	0.6655

Table 39. Modeling: Stratified k-CV with dfall, Mechanics.

From these tables and the ones for “dfanterior” we find the statistical parameters we have found for each section.

They are seen in Table 40.

Subjects	df0		dfanterior		dfall	
	<i>max_value</i>	<i>mean_value</i>	<i>max_value</i>	<i>mean_value</i>	<i>max_value</i>	<i>mean_value</i>
Electrom.	0.705	0.6839	0.7213	0.6887	0.7121	0.6917
Num. Meth.	0.913	0.8965	0.9125	0.8938	0.9121	0.8925
Materials	0.7692	0.7436	0.7662	0.7432	0.7662	0.7411
Diff. Equa.	0.8286	0.7981	0.8317	0.8012	0.8317	0.803
Informatics	0.8361	0.8062	0.8317	0.8053	0.833	0.8033
Mechanics	0.6756	0.6595	0.6774	0.6558	0.6774	0.6538
Subjects	df0		dfanterior		dfall	
	<i>row_std</i>	<i>column_std</i>	<i>row_std</i>	<i>column_std</i>	<i>row_std</i>	<i>column_std</i>
Electrom.	0.0179	0.0099	0.0137	0.0146	0.0138	0.0092
Num. Meth.	0.0191	0.008	0.0191	0.008	0.0197	0.009
Materials	0.0198	0.0114	0.0125	0.0123	0.0144	0.0127
Diff. Equa.	0.022	0.015	0.0246	0.0143	0.0224	0.0134
Informatics	0.0214	0.0145	0.0247	0.0115	0.0258	0.0126
Mechanics	0.0096	0.0091	0.0148	0.0124	0.0123	0.0128

Table 40. Results test 1. Testing several data frames.

As we see, in this case the difference between the different fractions of correct guesses from one data frame to another one are very small. In most cases the difference is under 1 percent, and it is in no case greater than 2.5 percent. This makes us think that the results are non-dependent on adding the extra columns that we have added if stratified k-fold cross validation is used, but we want further proof of it.

As in this stage we are testing the model for several data frames, and not just deciding which model is going to be used as in the early stages, we can conduct a deeper research on the issue.

For now, looking at the “fraction of correct guesses” has been enough to sustain the different claims we have made, but since the results found here are inconclusive, we will find another approach to determine the quality of our results.

We will use a confusion matrix to see if, being the “fraction of correct guesses” as similar as it is, there is still a way to sustain that one data frame obtains better results than the rest. This brings us to creating Test 2.

In this case, the information that the standard deviations give us is not that relevant. It is very similar for all data frames which was to be expected, as the method is the same and the data frames contain similar information.

Test 2

For this test, we wish to separate the result “fraction of correct guesses” into two. The idea is to get the “fraction of correctly predicted fails” and the “fraction of correctly predicted passes”. Because of the nature of our study, it is more interesting to predict “fails” than “passes”. This will be done with the values of the confusion matrix.

As we explained the purpose of this whole project should be to develop a tool capable of telling students if they should put special effort in a certain subject that they are likely to fail. Therefore, it is more desirable to warn a student to study harder when it is not necessary than to tell a student that is likely to fail a subject that he/she is doing okay. Translated into data mining terms this means that for an equal “fraction of correct guesses” it is more desirable to have a higher “fraction of correctly predicted fails”, because wrongly predicted fails will mean not warning a student that is more likely to fail a subject.

These two values can be easily calculated from the values that the confusion matrix returns with the following procedure:

$$\text{Fraction of correctly predicted fails} = \frac{\text{True Negatives}}{\text{True Negatives} + \text{False Positives}}$$

$$\text{Fraction of correctly predicted passes} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

We calculate these two parameters for the whole range of “max_depth” and “min_samples_leaf” we have used until now (for all three data frames and for all subjects).

Note that there should be two types of table for each data frame, one for the “fraction of correctly predicted fails” and one for the “fraction of correctly predicted passes”. That means two types, for 6 subjects for 3 data frames, which would make a total of 36 tables and that is before calculating “max_value”, “mean_value”, “row_std” and “column_std”, which add 6 more tables in total. As the amount of data would be so large and it can be summarized with the last 6 tables, only these 6 are shown below, despite the rest have also been calculated.

Only an exemplary table is shown in the annex for both parameters.

The results for the “fraction of correctly predicted fails” are:

Fraction of correctly predicted fails						
Subjects	df0		dfanterior		dfall	
	max_value	mean_value	max_value	mean_value	max_value	mean_value
Electrom.	0.4288	0.2998	0.452	0.3113	0.452	0.3385
Num. Meth.	0.202	0.0594	0.202	0.0845	0.2121	0.0883
Materials	0.4436	0.3627	0.458	0.3731	0.4562	0.3715
Diff. Equa.	0.3117	0.1957	0.3267	0.2109	0.3491	0.2268
Informatics	0.3534	0.2082	0.4031	0.2371	0.3691	0.2385
Mechanics	0.6211	0.5808	0.6427	0.5971	0.6329	0.5932
Subjects	df0		dfanterior		dfall	
	row_std	column_std	row_std	column_std	row_std	column_std
Electrom.	0.0295	0.0791	0.0312	0.0707	0.0256	0.0633
Num. Meth.	0.0549	0.0193	0.0529	0.0209	0.057	0.0251
Materials	0.0326	0.0148	0.0327	0.021	0.0204	0.017
Diff. Equa.	0.0387	0.0409	0.0415	0.0386	0.0401	0.0471
Informatics	0.0389	0.0459	0.047	0.0491	0.0444	0.0482
Mechanics	0.01	0.041	0.0143	0.0348	0.0122	0.0323

Table 41. Results test 2, first part. Testing several data frames.

Whereas the “fraction of correctly predicted passes” obtain:

Fraction of correctly predicted passes						
Subjects	df0		dfanterior		dfall	
	max_value	mean_value	max_value	mean_value	max_value	mean_value
Electrom.	0.938	0.8364	0.9484	0.8381	0.9227	0.8322
Num. Meth.	1	0.976	0.9981	0.971	0.9986	0.9693
Materials	0.9248	0.8679	0.9009	0.8637	0.905	0.8616
Diff. Equa.	0.9851	0.9272	0.9829	0.9275	0.9829	0.9267
Informatics	0.9847	0.9266	0.9741	0.92	0.9741	0.9175
Mechanics	0.8276	0.7221	0.8118	0.7031	0.8118	0.7027
Subjects	df0		dfanterior		dfall	
	row_std	column_std	row_std	column_std	row_std	column_std
Electrom.	0.0336	0.0441	0.031	0.0481	0.0266	0.0364
Num. Meth.	0.0263	0.0108	0.0256	0.0106	0.0262	0.0121
Materials	0.0343	0.0187	0.0263	0.0197	0.0247	0.0199
Diff. Equa.	0.0346	0.0268	0.0368	0.0253	0.0337	0.0259
Informatics	0.0327	0.0264	0.0385	0.0236	0.0383	0.0243
Mechanics	0.0173	0.0481	0.0214	0.0461	0.0189	0.0455

Table 42. Results test 2, second part. Testing several data frames.

The first thing that catches our eye when looking at this tables is that the “fraction of correctly predicted passes” is much higher than the “fraction of correctly predicted fails”, for all subjects and for all data frames.

This goes accordingly to what we had already anticipated: that the “Passes” are much easier to predict than the “Fails”.

As we said, we would prefer to predict the “Fails” better than the “Passes” so this situation is not very desirable.

Another thing that we see when we look at individual values for a certain combination of “max_depth” and “min_samples_leaf”, is that the “fraction of correctly predicted fails” tends to find its highest values where the “fraction of correctly predicted passes” finds its lowest values. This brings us to the conclusion that both variables cannot be optimized at the same time (we will not find a combination of inputs where both values have a peak).

An important finding that we make out of this test is that the values of the “fraction of correctly predicted fails” has the tendency to increase when adding new columns. For this reason, we decide to use an enhanced data frame.

Also, we see that “dfanterior” gets the best results for some subjects but in other cases it is “dfall” that gets them. This suggests us that the columns that we add are more or less beneficial depending on what we are trying to predict.

Finally, we see that the model struggles a lot when it tries to predict the “fraction of correctly predicted fails” for certain subjects. The clearest example of this is the subject of Numerical Methods.

Conclusion of the tests

The addition of new columns to the basic data frame does not increase the overall fraction of correct predictions in terms that we consider significative, but it is still a good choice to add them as it increases the “fraction of correctly predicted fails” in front of the “fraction of correctly predicted passes” which is preferable to us.

For this reason, we consider that “dfanterior” and “dfall” are the best choices to take when it comes to choosing the data frame if stratified k-CV is used. As we do not see any significative difference between them in the overall results, we consider that with the information we have they are both as valid. One option could be to use “dfanterior” to predict Materials, Informatics and Mechanics and use “dfall” to predict Electromagnetics, Numerical Methods and Differential Equations. This would ensure that we use the data frame that gets the highest “fraction of correctly predicted fails” for each subject.

5.7. Is the amount of data large enough?

In order to answer this question, we will generate a learning curve for each subject and analyze its behavior.

Test 1

The aim of this test is simply to create a learning curve for each subject and say if it is stable or if it indicates that the model still needs more data to reach a point of stability.

The way to go will be to generate a subsample of the data frame that will contain different percentages of the total amount of data. They are what we will refer to as `x_points`.

A model will be created and the “fraction of correct guesses” will be obtained each one of the `x_points`.

In order to reduce the variability in the results, each `x_point` will be modeled multiple times and a mean value will be found. These mean values are used for the plotting.

The data frame we have used is “`dfanterior`”, but for the reasons we explained in the previous section, it could have been “`dfall`” as well.

After following this procedure, we obtain the following learning curves:

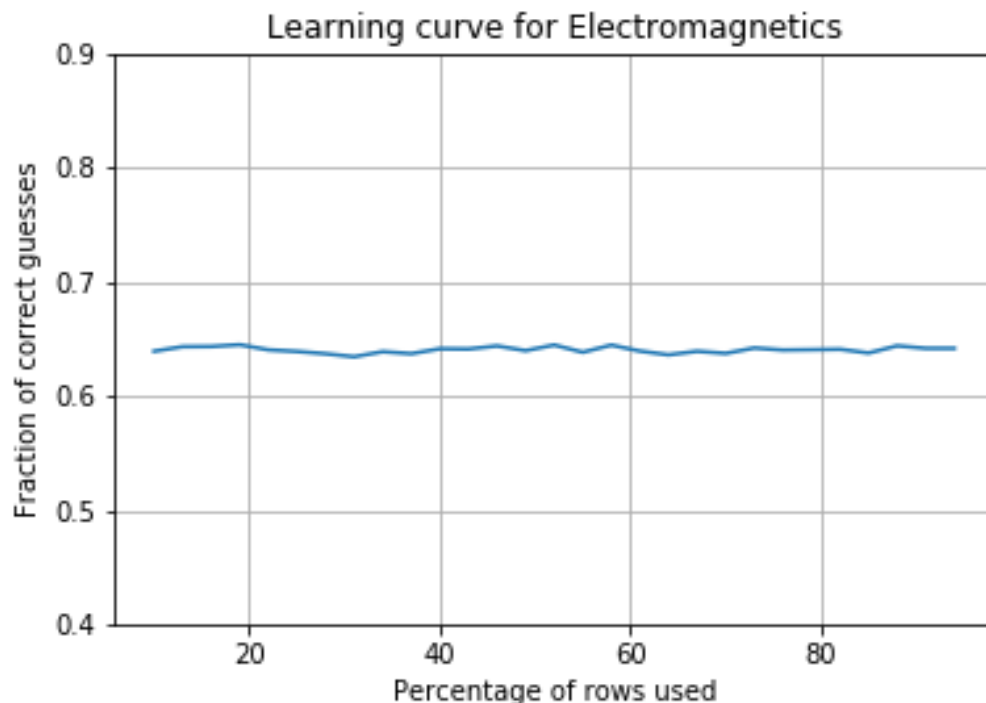


Figure 14. Learning curve for Electromagnetics.

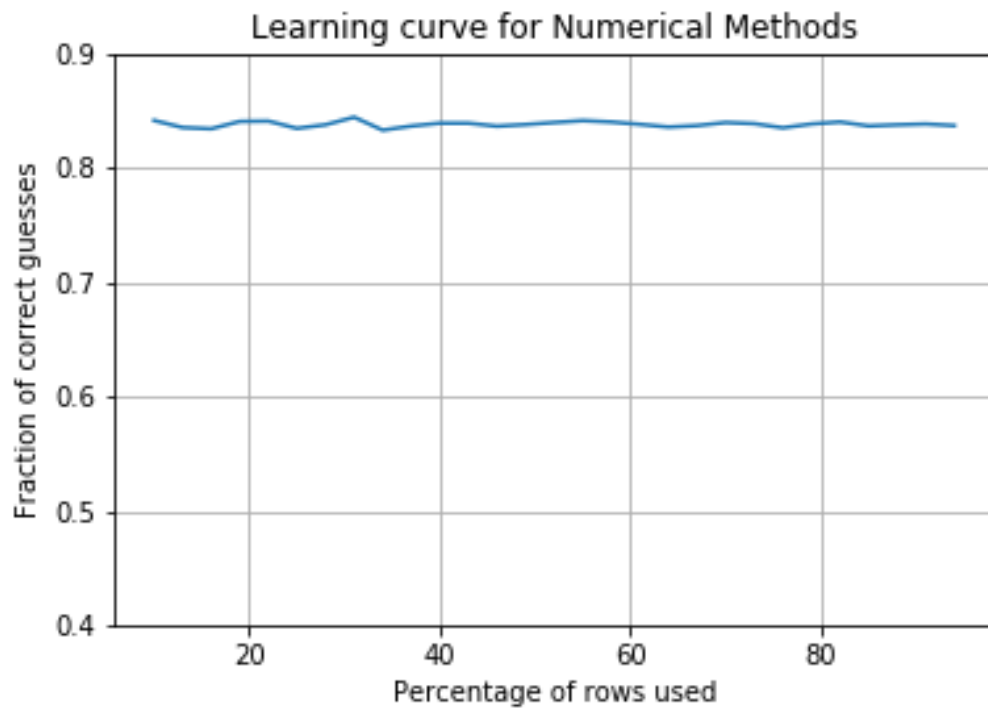


Figure 15. Learning curve for Numerical Methods.

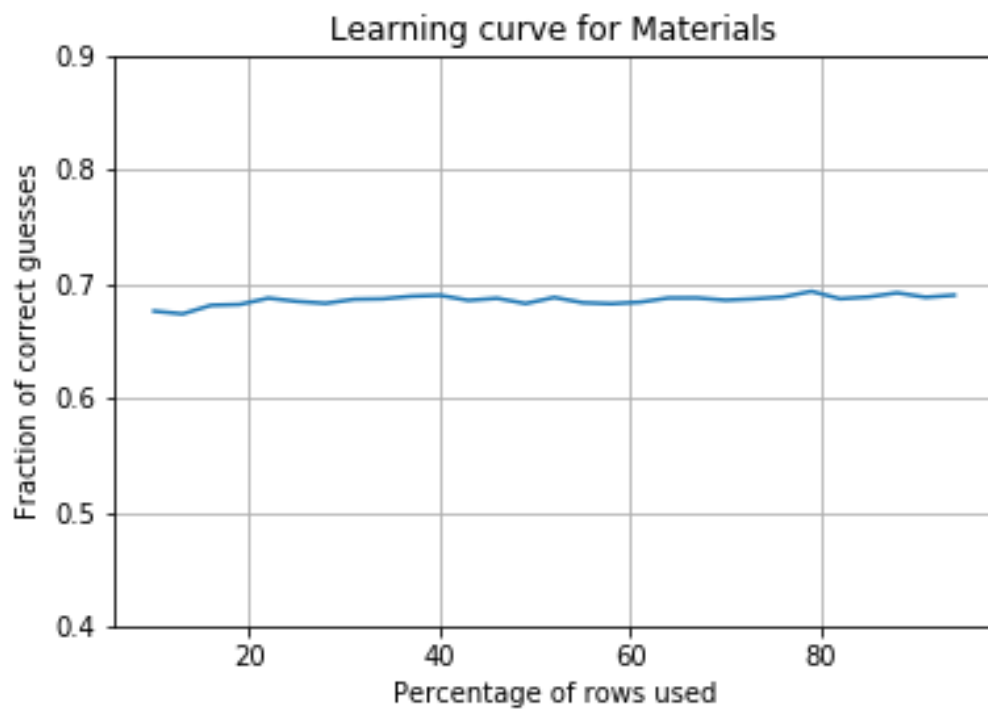


Figure 16. Learning curve for Materials.

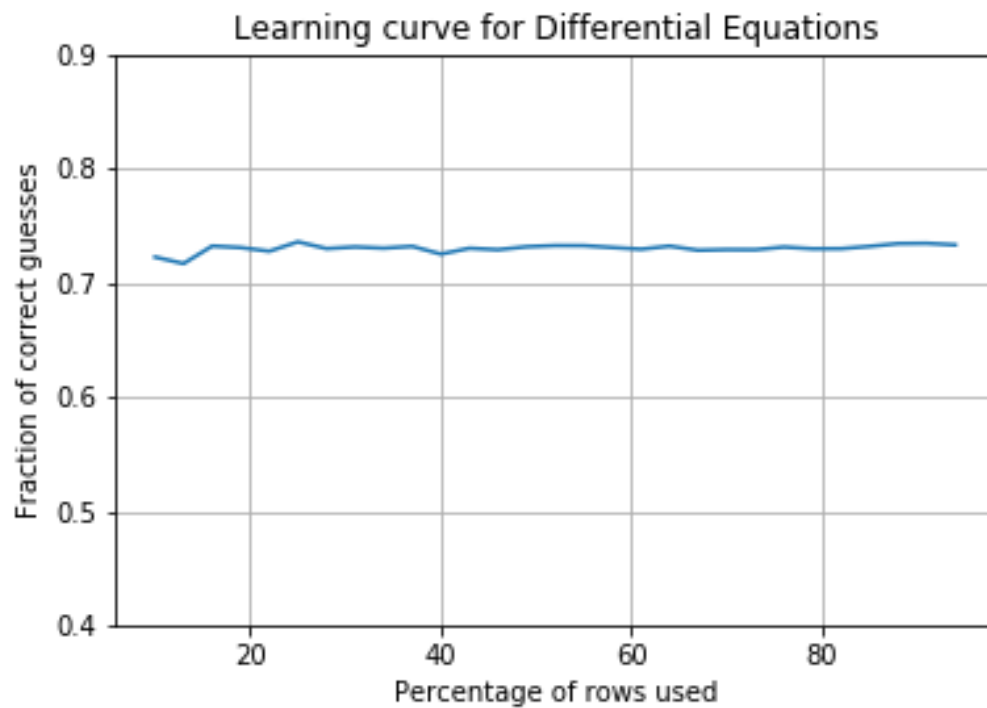


Figure 17. Learning curve for Differential Equations.

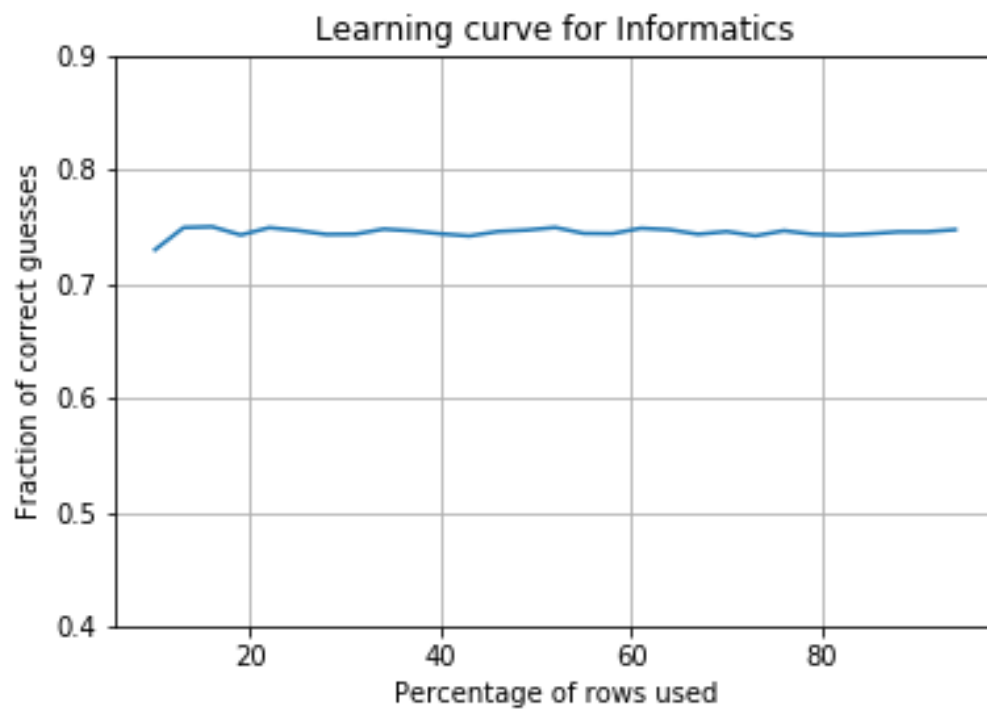


Figure 18. Learning curve for Informatics.

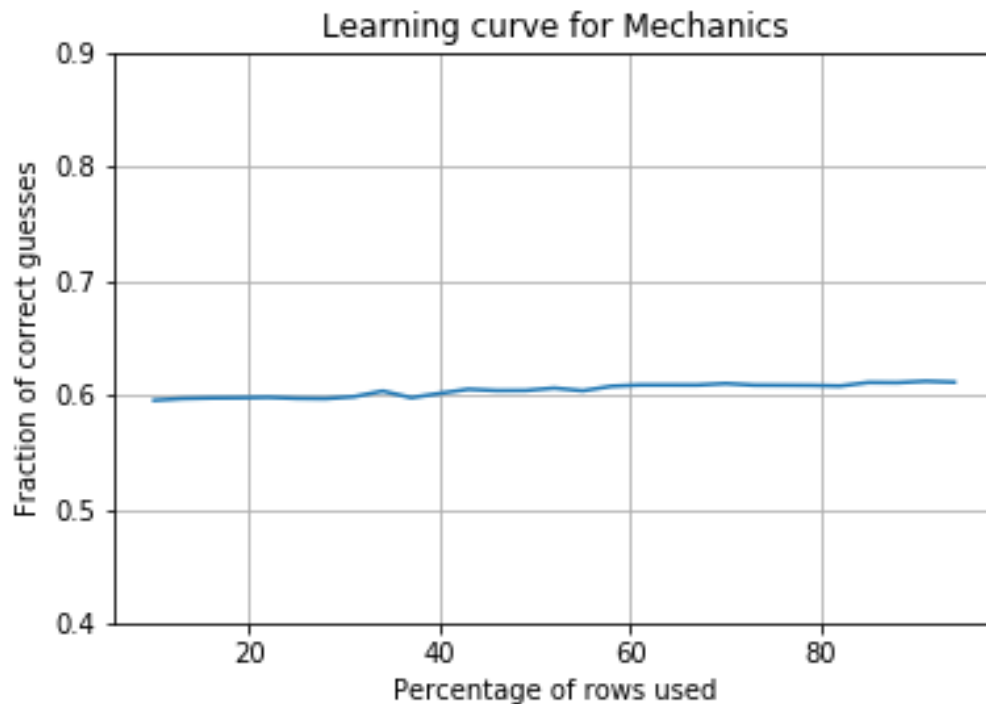


Figure 19. Learning curve for Mechanics.

Conclusions of the test

For the subjects Electromagnetics, Numerical Methods, Differential Equations and Informatics we consider that the curve is pretty much stable if we take over 20 percent of our data (this value is just an orientation). In other words, the “fraction of correct guesses” does not vary considerably if we use a small number of rows or if we use all the rows in the data frame. This is because, with stratified k-CV the data is selected in a way that is as representative as possible, which reduces the amount of data necessary to be in the stable region of the curve.

There are several repercussions to these results.

Firstly, this suggests us that the results of these subjects will not be improved when new data is added. Meaning that having data from 2017, 2018 and so on, will not mean that we get better results in terms of “fraction of correct guesses”. This answers the question we asked in the Business Comprehension stage about whether we have a large enough amount of data to create our model. This test concludes that as long as we are using stratified k-CV we do have enough data.

Secondly, we can conclude that it is not necessary to use the data we have in the form of students with “unattempted” subjects. They would provide some extra rows that are not needed according to these learning curves.

For the subjects of Materials and Mechanics the issue is not that clear. In both cases we see that the curve is not completely stable, but that it slightly increases. The overall increase is of 1.275 percent for Materials and of 2.405 percent for Mechanics.

This suggests us that the results can be improved if new data is added. But if the curve keeps following the same tendency (which is not sure) a considerable amount of new rows will be necessary to improve the results significantly, because as we have found, adding around 1800 new students has only increased the “fraction of correct guesses” in the percentages we have just mentioned.

6. Budget

This section's purpose is to determine an overall estimated cost of the project.

As this project is a data mining project the costs are small in comparison to other engineering projects. This is because the costs of the materials are reduced to effectively nothing and the cost of infrastructures is small.

We will divide the costs into two:

Labor cost

For the development of this enterprise the work of a data analyst was required.

The line that was followed in the previous project will be followed for the prices of labor, as it seems reasonable in comparison to the real prices of these concepts. For this section, the labor will be divided into Research, Analysis and Presentation.

Infrastructure cost

The infrastructure cost only has two concepts in our case. The computer related costs (as all the software we used is free, we must only take into account the cost of the hardware used).

The hardware used was a laptop whose original price was 1300€.

Considering the life expectancy of a good quality laptop to be of 3 years (naturally this is just an estimation), we will consider the cost of the hardware used in this project to be the depreciation of value of this laptop during the time that the project lasted.

$$\text{Hardware cost} = 1300 \text{ euros} \times \frac{0.3 \text{ years}}{3 \text{ years}} = 130 \text{ euros}$$

Again, this is just an estimation. But given that there are other concepts in the budget that impact the cost of the project to a much larger degree it is not necessary to be that precise in this part.

Labor costs			
Concept	Price per hour	Hours	Total cost
Research	30€/h	70h	2.100€
Analysis	40€/h	210h	8.400€
Presentation	25€/h	30h	750€
Infrastructure costs			
Concept			Total cost
Computer resources			130 €
Office material			5€
Total cost			11.385€

Table 43. Budget.

The estimated cost of this project is of 11385€, which we could round to 11400€.

Therefore, the cost of the project is a bit larger than that of the last project (which was of 10.454,93€), but the difference is reasonable.

7. Environmental Impact

This project has almost entirely been developed virtually. For this reason, the environmental impact that it has caused is very small in comparison to other projects of other fields of engineering.

Most of the work has been developed in a computer, without generating any waste of material. The physical material that has been used apart from the laptop are the white sheets of paper that have been used to take notes. These will not be considered as waste as they have been taken into account in the Budget. The only environmental impact that this project has created comes from the spending of electrical energy of the laptop, the router and the lights used to enlighten the rooms. Nonetheless, this project has been developed for the most part in public libraries and in the library of ETSEIB, which means that the energy spent on the router and on the lights of the rooms would have been spent regardless of whether or not our project had been developed.

8. Project Planning

The following chart shows the planning of the project from its registration on the 27th of February of 2019.

The project was turned in on the 24th of June 2019.

The timeline is shown in a Gantt chart in *Figure 20*.

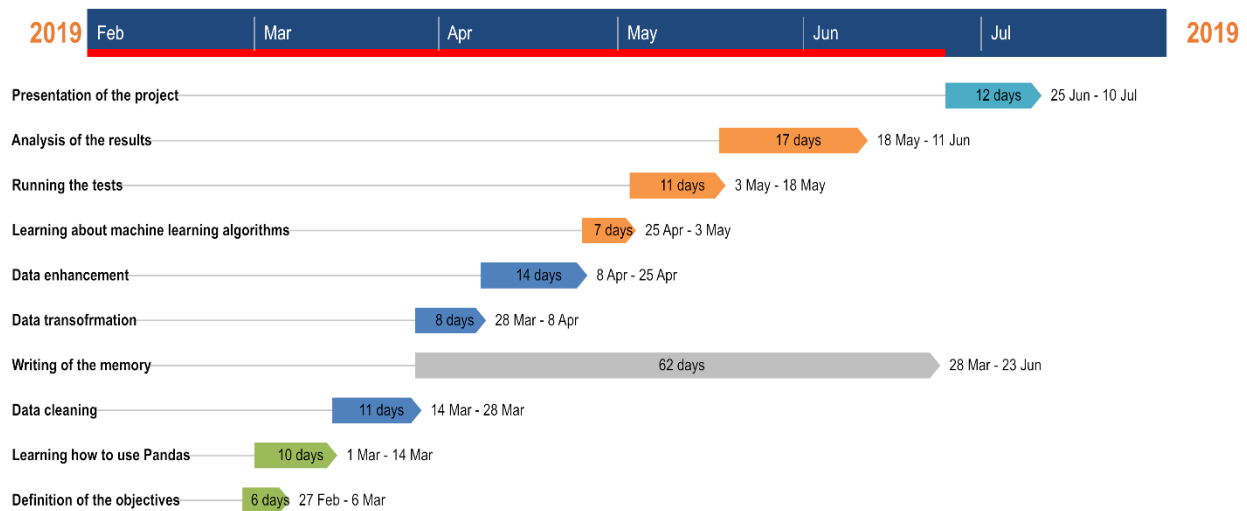


Figure 20. Timeline of the project.

9. Conclusions

The purpose of this project was to make predictions over whether ETSEIB students should pass or fail the subjects of Q3, using data mining techniques. At the same time, we wanted to suggest things that could have been done differently from the PDMP and see how changing them affected the results.

This has been accomplished using a structured step by step methodology, modifying one input at each time to see its impact on the results. The procedure has been to firstly state a hypothesis, and then run experiments to extract some conclusions on it. These conclusions have been then used as the starting point for the new hypothesis.

The first test we did could be considered a preliminary test. We generated the results for a non-optimized combination of inputs, in the sense that we had not looked for the combination of them that gave the best results. This led us to think that the “fraction of correct guesses” of this test could be taken as a reference to what the baseline results should look like.

After finding this initial reference we began to find the actual results.

Firstly, we made the hypothesis that the validation method used in the previous project, which was holdout, might lead to unrealistic values in the way that it was applied. The reason, was that some of the students used for the testing set, specifically those who entered ETSEIB in 2016, were only the best performing ones of their year, and that for this reason they were more likely to pass all the subjects, thus being easier to predict.

After running some tests on the PDMP data frame, we concluded that there was enough evidence to think that another method would suit better with the data we had. Therefore, we decided that from that point on, stratified k-CV would be used, as it seemed a more reliable verification method.

Once the verification method had been established, we still wondered if 2016 students could be problematic even with stratified k-CV. To prove or disclaim that hypothesis we set another test that consisted in applying stratified k-CV on two data frames; one containing the students of 2016 and one without them (and both keeping the same columns as in the original data frame).

The results gave similar results “fraction of correct guesses” for both data frames. This brought us to think that students from 2016 would be kept because no reason to drop them had been found.

Having decided that all rows of the PDMP were to be used, we asked ourselves if there was the possibility to improve the results by adding extra columns of information. After comparing the “fraction of correct guesses” of the original data frame to two other ones our results were inconclusive to what the best choice was. For this reason, we looked at the “fraction of correctly predicted fails”. The results showed that for certain subjects “dfanterior” was the best choice and that for others “dfall” was better. Therefore, our

conclusion was that adding new columns was justified, but that depending on the subject some columns were better than others.

Finally, we hypothesized over if the “fraction of correct guesses” could be improved in the following years, as data from new students was gathered. This question was answered by plotting the learning curve of the model.

For Electromagnetics, Numerical Methods, Differential Equations and Informatics the curve seemed stable, whereas for Materials and Mechanics it showed a slight increase. This leads us to believe that Materials and Mechanics could get slightly better results if the tendency of the curve continues in the following years, but it cannot be said with certainty.

All in all, we have given numerical evidence to suggest:

- An alternative validation method.
- What columns of information seem to improve the initial results.
- Why we should keep certain rows that seemed troubling at first.
- Which subjects could potentially get better results with the data from the following years.

For this reason, we consider that the initial objectives have been accomplished.

10. Future work

During the development of this project we have thought of several ideas of things that could have been done differently or that could be tested in future work.

The biggest change that we thought of was to apply another machine learning algorithm, different from the decision tree. One possible substitute could be a regression. Of course, this would mean that the project be an entirely different one, and it would not fit the objective of our project, which is why this idea was quickly discarded for this project.

If instead of using a whole different machine learning algorithm, the future work was to focus on using the decision tree but with some changes, there are several ideas that could be applied.

For example, instead of using the last grade of each subject and the number of times that it has been attempted by a student, we could just use the grade they got at their first attempt. This would mean testing a completely different data set.

Another idea could be to discretize the initial predicting variables of the data frame. What we mean by this is that a few integer values (such as for example 1, 2, 3, 4), could replace grades from 0.0 to 10.0. Different assignments could be tested to see if they can improve the results. For example, one idea could be to split each subject into groups of equal size and establishing one numerical code for each quartile. This way we would have four groups of performance for each subject.

As we have said, our results seem to show that with stratified k-CV we have enough rows for 4 of our subjects. This means for these subjects that there might be the possibility to drop some rows of data without worsening the results. Taking this idea, one could try to find rows of data which confuse the model and delete them to see if the model improves its performance. And for the other 2 subjects we could try to do the opposite. We could use the students that have “unattempted” subjects to increase the number of rows of these two subjects. This would require creating an independent data frame for each subject as the number of rows would not necessarily be the same.

11. Sources

- [1] Han, Jiawei et al. *Data Mining: Concepts and Techniques*. 3rd ed. Waltham: Elsevier, 2012. 673 p. ISBN 978-0-12-381479-1.
- [2] Olson, David L. et al. *Advanced Data Mining Techniques*. Berlin: Springer, 2008. 169 p. ISBN 978-3-54-076916-3.
- [3] Pyle, Dorian. *Data Preparation for Data Mining*. San Francisco: Morgan Kaufmann Publishers, 1999. 460 p. ISBN 978 -1-55-860529-9.
- [4] Ron Kohavi. *A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection*. Appears in the International Joint Conference on Artificial Intelligence (IJCAI), 1995.
- [5] Pandas: Documentation of the library. On how to manipulate data. Webpage: <https://pandas.pydata.org/>
- [6] SciKit-learn: Documentation of the library. On machine learning with Python. Webpage: <https://SciKit-learn.org/stable/>
- [7] For data mining related doubts. KDNuggets. Webpage: <https://www.kdnuggets.com>
- [8] For general programming doubts. Stackoverflow. Webpage: <https://stackoverflow.com>
- [9] For any doubts related to the plotting of data. Matplotlib. Webpage: <https://matplotlib.org/>
- [10] For definitions and doubts regarding translation. Oxford Dictionary.

12. Annex

Exemplary tables for the subject of Electromagnetics of the “fraction of correctly predicted fails” and the “fraction of correctly predicted passes”

Electromagnetics (dfall) (fraction of correctly predicted fails)										
<i>min_samples_leaf</i>	<i>max_depth</i>									
		3	5	7	10	12	14	16	18	20
	2	0.1796	0.274	0.3514	0.3808	0.4319	0.4427	0.4303	0.452	0.4412
	3	0.1796	0.2771	0.339	0.3901	0.4087	0.4164	0.4241	0.418	0.4319
	5	0.1796	0.2755	0.3313	0.37	0.4071	0.4009	0.3963	0.3885	0.3901
	10	0.1796	0.3111	0.3406	0.3653	0.3746	0.3793	0.3808	0.3808	0.3824
	20	0.1796	0.3127	0.305	0.3715	0.3746	0.3746	0.3777	0.3793	0.3793
	25	0.1796	0.3375	0.3313	0.3545	0.3529	0.3529	0.3529	0.3529	0.3498
	30	0.1796	0.3096	0.3297	0.3452	0.3452	0.3452	0.3483	0.3452	0.3452
	35	0.1796	0.3142	0.356	0.356	0.356	0.3576	0.356	0.356	0.3576
	40	0.1796	0.3019	0.356	0.3514	0.3483	0.3483	0.3483	0.3483	0.3514
	50	0.1594	0.2895	0.3344	0.3235	0.3235	0.3235	0.3235	0.3235	0.3235

Electromagnetics (dfall) (fraction of correctly predicted passes)										
<i>min_samples_leaf</i>	<i>max_depth</i>									
		3	5	7	10	12	14	16	18	20
	2	0.9159	0.8619	0.822	0.7956	0.7514	0.7489	0.7514	0.7428	0.7287
	3	0.9165	0.8686	0.8232	0.7962	0.7716	0.752	0.7532	0.7606	0.7594
	5	0.9165	0.8748	0.8275	0.7999	0.7759	0.7747	0.768	0.7649	0.7618
	10	0.9165	0.868	0.8392	0.7925	0.7882	0.7839	0.7833	0.7821	0.7815
	20	0.9153	0.8607	0.8514	0.8165	0.8122	0.8109	0.8109	0.8115	0.8146
	25	0.9153	0.8564	0.8508	0.841	0.8386	0.8355	0.8398	0.8373	0.841
	30	0.9153	0.8631	0.8459	0.8293	0.8293	0.8269	0.8275	0.83	0.8275
	35	0.9153	0.8692	0.8496	0.8422	0.8422	0.841	0.8422	0.841	0.8422
	40	0.9153	0.8717	0.8471	0.8508	0.8502	0.8502	0.8508	0.8508	0.8508
	50	0.9227	0.8797	0.8545	0.857	0.857	0.857	0.857	0.857	0.857

13. Index of Figures

Figure 1. Stages of the CRISP-DM.....	10
Figure 2. Python logo.	12
Figure 3. Spyder logo.....	12
Figure 4. SciKit-learn logo.....	13
Figure 5. NumPy logo.	14
Figure 6. Matplotlib logo.....	15
Figure 7. Pivoting.....	23
Figure 8. A decision tree showing the survival of passengers of the Titanic.	27
Figure 9. Graphical representation of the phenomena of overfitting.	28
Figure 10. Basic principle the k-CV.....	30
Figure 11. Confusion matrix.	31
Figure 12. Train-test split used in the PDMP.....	35
Figure 13. Histogram of the year of entry of students at ETSEIB.....	44
Figure 14. Learning curve for Electromagnetics.....	58
Figure 15. Learning curve for Numerical Methods.	59
Figure 16. Learning curve for Materials.	59
Figure 17. Learning curve for Differential Equations.	60
Figure 18. Learning curve for Informatics.	60
Figure 19. Learning curve for Mechanics.	61
Figure 20. Timeline of the project.....	66

14. Index of Tables

Table 1. Data files and their information.	17
Table 2. Identification of the subjects in Q1, Q2 and Q3.	19
Table 3. Information in the basic data frame.	20
Table 4. Sample of the basic data frame.	21
Table 5. Baseline results of the decision tree.	33
Table 6. Fraction of passes of each set of data used in the PDMP.	36
Table 7. "N_QUAD_FASEIN" and "ANY_ACCES" for a sample of the data set.	36
Table 8. Modeling: Holdout, Electromagnetics.	38
Table 9. Modeling: Holdout, Numerical Methods.	39
Table 10. Modeling: Holdout, Materials.	39
Table 11. Modeling: Holdout, Differential Equations.	39
Table 12. Modeling: Holdout, Informatics.	40
Table 13. Modeling: holdout, Mechanics.	40
Table 14. Modeling: Stratified k-CV, Electromagnetics.	40
Table 15. Modeling: Stratified k-CV, Numerical Methods.	41
Table 16. Modeling: Stratified k-CV, Materials.	41
Table 17. Modeling: Stratified k-CV, Differential Equations.	41
Table 18. Modeling: Stratified k-CV, Informatics.	42
Table 19. Modeling: Stratified k-CV, Mechanics.	42
Table 20. Results for holdout vs. stratified k-CV.	42
Table 21. Modeling: Stratified k-CV without 2016 students, Electromagnetics.	45
Table 22. Modeling: Stratified k-CV without 2016 students, Numerical Methods.	45
Table 23. Modeling: Stratified k-CV without 2016 students, Materials.	46
Table 24. Modeling: Stratified k-CV without 2016 students, Differential Equations.	46
Table 25. Modeling: Stratified k-CV without 2016 students, Informatics.	46
Table 26. Modeling: Stratified k-CV without 2016 students, Mechanics.	47
Table 27. Results on deciding whether to keep 2016 students.	47
Table 28. Modeling: Stratified k-CV with df0, Electromagnetics.	49
Table 29. Modeling: Stratified k-CV with df0, Numerical Methods.	50
Table 30. Modeling: Stratified k-CV with df0, Materials.	50
Table 31. Modeling: Stratified k-CV with df0, Differential Equations.	50
Table 32. Modeling: Stratified k-CV with df0, Informatics.	51
Table 33. Modeling: Stratified k-CV with df0, Mechanics.	51
Table 34. Modeling: Stratified k-CV with dfall, Electromagnetics.	51
Table 35. Modeling: Stratified k-CV with dfall, Numerical methods.	52
Table 36. Modeling: Stratified k-CV with dfall, Materials.	52
Table 37. Modeling: Stratified k-CV with dfall, Differential Equations.	52
Table 38. Modeling: Stratified k-CV with dfall, Informatics.	53
Table 39. Modeling: Stratified k-CV with dfall, Mechanics.	53
Table 40. Results test 1. Testing several data frames.	54
Table 41. Results test 2, first part. Testing several data frames.	56
Table 42. Results test 2, second part. Testing several data frames.	56
Table 43. Budget.	63

