Sílvia Sanromà Martorell

Assessment of diabetic retinopathy risk using random forests

FINAL DEGREE PROJECT

Advisors: Dr. Antonio Moreno Ribas and Dra. Aïda Valls Mateu

Degree in Computer Engineering

UNIVERSITAT ROVIRA I VIRGILI

Tarragona

April, 29th 2016

Agraïments

Aquest treball ha rebut el suport d'una beca de col·laboració amb departaments del MECD 2015 dins del grup de recerca *Intelligent Technologies for Advanced Knowledge Acquisition* (ITAKA). També ha comptat amb el finançament dels projectes de recerca "Desarrollo de un modelo de cribado de la retinopatía diabética a partir de una cohorte histórica, generando una aplicación de apoyo clínico" (Instituto de Salud Carlos III, PI12/01535, 2013-2015) i "Ampliación e implantación de una aplicación de apoyo clínico para el cribado de retinopatía diabètica, incluyendo la lectura de imágenes" (Instituto de Salud Carlos III, PI15/01150, 2016-2018).

Vull aprofitar aquestes línies per agrair als meus directors, el Dr. Antonio Moreno i la Dra. Aïda Valls, tota l'ajuda, els consells i les directrius que m'han proporcionat durant la realització d'aquest treball de fi de grau. També agrair-los la oportunitat que m'han donat de descobrir el camp de la Intel·ligència Artificial i tots els coneixements i motivació que m'han transmès des del primer dia.

Aquest treball no hagués estat possible sense el Grup de recerca en Oftalmologia (HUSJR – IISPV), i per tan agrair-los el seu esforç i treball en facilitar-nos les dades i la part mèdica del projecte. Vull agrair especialment al Dr. Pere Romero, la Sofia de la Riva i el Ramon Sagarra.

Agrair també al Dr. Sergio Martínez el treball previ que va realitzar i sobre el qual es basa aquest projecte.

També vull donar les gràcies als meus companys del grup de recerca i companys de classe que en tot moment m'han animat i m'han ajudat sempre que ho he necessitat.

Finalment, agrair sobretot a la meva família, que m'ha recolzat en tot moment i m'ha donat forces per tirar endavant tan aquest projecte com la carrera.

Table of contents

1 Introduction 7
1.1 Objectives 8
1.2 Overview of the methodology 9
1.3 Document structure 10
2 Data from diabetic patients 11
2.1 Extraction of the data 11
2.1.1 Group of patients 1 12
2.1.2 Group of patients 2 12
2.2 Data processing 12
2.2.1 Class diagram to store the data 14
2.2.2 How the program reads and stores the data 15
2.3 Preparation of the training and testing set 16
2.4 Balanced data for non-balancing methods 17
2.5 Summary 18
3 Methods 19
3.1 Logistic Regression 20
3.2 Classification model based on Decision trees 21
3.2.1 The ID3 algorithm 22
3.2.2 Modification of classic ID3 24
3.3 Classification model based on Random Forest 25
3.3.1 Classification using a Random Forest: Voting function 27
3.4 Comparison between Decision Trees and Random Forest 28
3.5 Summary 29
4 Evaluation 31
4.1 Classification results of Logistic Regression 32
4.2 Classification results of Decision Trees 32
4.3 Classification results of Random Forests 34
4.4 Comparison of the methods 40
4.5 Summary 42
5 Clinical application 43
5.1 How will the program be integrated in the medical protocol? 43
5.2 Design 44
5.3 Implementation 47
5.4 Examples 48
5.5 Summary 52

6 Conclusions and future work 53
6.1 Conclusions 53
6.2 Future work 54
6.3 Publications and presentations 54
References 57
Appendix A. Publication in ESANN 59
Appendix B. Poster in ESANN 67
Appendix C. Poster in Fira Recerca en directe 2016 71

List of tables

Table 1: Illustrative example of the data structure 11
Table 2: Data format with 9 attributes and RD 13
Table 3: Training and testing set partition 17
Table 4: Balanced training set partition 18
Table 5: Pros and cons of a single Decision Tree 28
Table 6: Pros and cons of Random Forest 29
Table 7: Sensitivity, Specificity and Accuracy. 31
Table 8: Result table for Logistic Regression 32
Table 9: Tested thresholds for Decision Trees 33
Table 10: Table of results for the Decision Trees thresholds 33
Table 11: Results of the best Decision Tree 34
Table 12: Threshold analysis parameters 35
Table 13: Numerical results for the threshold analysis 35
Table 14: Number of attributes analysis parameters 36
Table 15: Numerical results for the number of attributes analysis 37
Table 16: Number of trees analysis parameters 38
Table 17: Numerical results for the number of trees analysis 38
Table 18: Random Forest final configuration 40
Table 19: Classification results for Random Forest 40
Table 20: Comparison of the measures of the classification methods 41
Table 21: Data patient example 1 48
Table 22: Data patient example 2 49
Table 23: Data patient example 3 51

List of figures

Figure 1: Comparison between normal and diabetic retinopathy vision 7
Figure 2: Process of learning a model and classifying a patient 9
Figure 3: Categorisation process 13
Figure 4: Class diagram for data storage 14
Figure 5: Example of joining patients with the same attribute values 16
Figure 6: Supervised learning data flow 16
Figure 7: Decision Tree example 21
Figure 8: General architecture of a Random Forest 25
Figure 9: Voting method of the Random Forest 28
Figure 10: Results for the Decision Trees thresholds 33
Figure 11: Graphical results for the threshold analysis 35
Figure 12: Unknown percentage for the threshold analysis 36
Figure 13: Graphical results for the number of attributes analysis 37
Figure 14: Unknown percentage for the number of attributes analysis 38
Figure 15: Graphical results for the number of trees analysis 39
Figure 16: Unknown percentage for the number of attributes analysis 39
Figure 17: Results comparison of the classification methods 41
Figure 18: RETIPROGRAM logo 43
Figure 19: RETIPROGRAM in the medical protocol for DR detection 43
Figure 20: RETIPROGRAM design 45
Figure 21: RETIPROGRAM GUI 46
Figure 22: Probability bar in the GUI 47
Figure 23: Prediction for example 1 49
Figure 24: Prediction for example 2 50
Figure 25: Prediction for example 3 51

List of Algorithms

Algorithm 1: loadData..... 15
Algorithm 2: ID3 algorithm..... 23
Algorithm 3: Random Forest algorithm..... 27

1 Introduction

Diabetes Mellitus (DM) is a group of metabolic diseases in which there are high blood sugar levels over a prolonged period. DM is one of the more prevalent chronic diseases in the world. According to the World Health Organization, 347 million people worldwide (around 4.6% of the population) suffer from DM, and it has been predicted that it will be the 7th cause of death by 2030. Only in 2012 it was the direct cause of 1.5 million deaths. It is also a leading cause of complications such as blindness, amputation and kidney failure [1].

Diabetic retinopathy (DR), also known as diabetic eye disease, happens when damage occurs to the retina. It is one of the more widespread morbidities of Diabetes Mellitus, and it has been increasing steadily in the last years. Its main effect, secondary blindness, has a large social and economic impact in healthcare. In the following figure it can be seen the same scene viewed by a person with normal vision on the right and with diabetic retinopathy on the left, so that it can be seen the impact it has in their life.

Figure 1: Comparison between normal and diabetic retinopathy vision

The longer a person has diabetes, the higher his or her chances of developing diabetic retinopathy. The onset of this disease is affected by different risk factors such as its duration, poor metabolic control or association of hypertension. The early detection of DR, by means of periodic controls, reduces significantly the financial cost of the treatments and decreases the number of patients who develop blindness. In Spain it is estimated that screening for DR in diabetic patients is less than 30% per year, well below other European countries, where the ratio is about 50% as is the case in Germany. European countries pledged to generate systematic screening programs to reach at least an 80% screening level in diabetics [2]. There is therefore a clear need to increase the ratio of screening programs by establishing efficient, effective and fast protocols.

Some scientific societies recommend that diabetic patients should be screened for DR every year[1]; however, in practice this periodicity is very hard to achieve, due to the large number of diabetic people, the lack of enough human and material resources in medical centres and the economic cost of the screening procedure. Currently, there is not any informatic system to help the professionals to decide whether a patient is prone

[1] For example, the American Diabetes Association [3], the American Academy of Ophthalmology and the Royal College of Ophthalmologists [4].

to develop diabetic retinopathy or not. Thus, there is a strong interest in developing a tool that can analyse the personal and clinical data of a diabetic person and help the medical practitioner to determine his or her risk of developing DR, so that the temporal distance between successive controls may be adjusted depending on it. With such a tool, human and material resources would be used more efficiently.

This Bachelor's Degree Final Project aims to use Artificial Intelligence methods to help doctors to decide if a patient is likely to have diabetic retinopathy so that they are able to both detect these patients early and to save money by running the expensive tests only on the most prone patients. For the success of this project, it is important to work closely with doctors specialized in diabetic retinopathy. Therefore, this project has been developed in close collaboration with the team of Dr. Pere Romero, head researcher of the Ophthalmology Group from Sant Joan University Hospital from Reus.

The work developed in this final project is framed within the following funded Spanish research projects:

- Desarrollo de un modelo de cribado de la retinopatía diabética a partir de una cohorte histórica, generando una aplicación de apoyo clínico (Instituto de Salud Carlos III, PI12/01535, 2013-2015).
- Ampliación e implantación de una aplicación de apoyo clínico para el cribado de retinopatía diabética, incluyendo la lectura de imágenes (Instituto de Salud Carlos III, PI15/01150, 2016-2018).

1.1 Objectives

Herein lie the objectives of this project:

1) Study the basic methods of supervised classification, mainly decision trees and random forests.
2) Prepare a training and a testing set based on real medical data corresponding to patient records from Sant Joan Hospital of Reus. It is necessary to pre-process the data and generate a file with a specific format.
3) Find a model based on historical data that may accurately predict the risk of diabetic retinopathy of a new patient given his or her clinical data. Artificial Intelligence and mathematical methods such as logical regression, decision trees and random forests should be studied in order to find the best model. According to the doctors, this model should give at least an 80% of specificity and sensitivity[2] when validating it.
4) Design and implement a tool that uses that model so that the doctors can easily and efficiently interact with it. The tool should receive the clinical data of the patient and give a prediction of the diabetic retinopathy risk according to the model. It would be useful to have some graphic representation in the application to present the result of the prediction.

[2] See section 5 for the definition of these terms

1.2 Overview of the methodology

The basic task of the work is to design a Clinical Decision Support System that, taking historical data of patients and the data of an unlabelled patient as an input will be able to:

a) Analyse the historical data in order to find a model that may predict the class of any other patient. The model should be general enough to make a prediction on any kind of patient, even if there is no other similar patient in the input data.
b) Give a prediction for the unlabelled patient in an easy understandable way for the doctors.

Step (a) is focused on objectives (1), (2), (3) while step (b) is focused on objective (4).

The whole process is supervised, which means that it is a machine learning task of inferring a classification function from labelled training data. In supervised learning, each example is a pair consisting of an input object (typically a vector of attribute values) and a desired output value (also called the supervisory signal). A supervised learning algorithm analyses the training data and produces an inferred function, which can be used for mapping new examples. An optimal scenario will allow for the algorithm to correctly determine the class labels for unseen instances. This requires the learning algorithm to generalize from the training data to unseen situations in a "reasonable" way. The following figure describes this process.

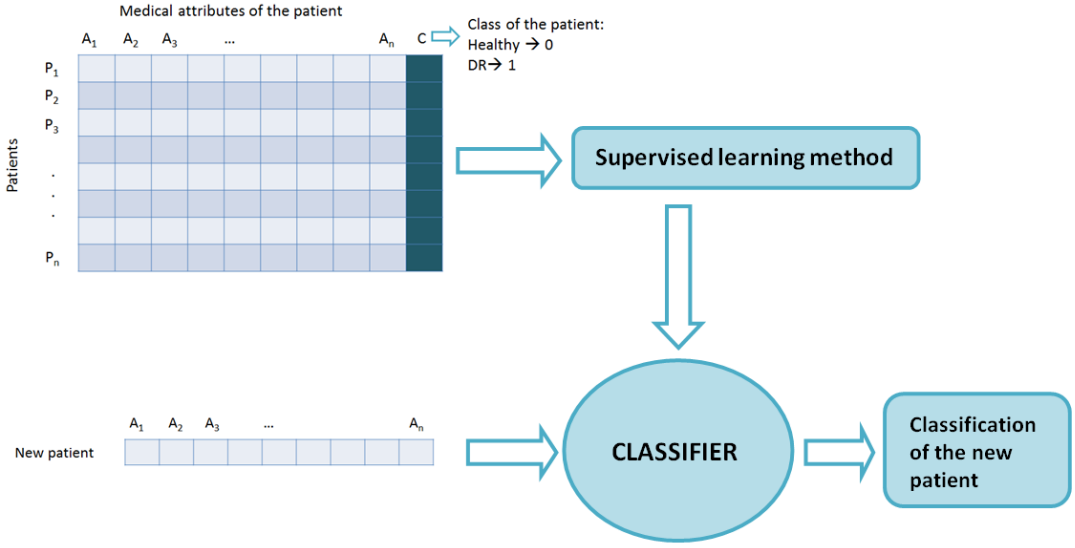


Figure 2: Process of learning a model and classifying a patient

In order to solve a given problem of supervised learning, one has to perform the following steps:

1. Determine the type of training examples. Before doing anything else, the user should decide what kind of data is used as a training set. In the case of handwriting analysis, for example, this might be a single handwritten character, an entire handwritten word, or an entire line of handwriting.

2. Gather a training set. The training set needs to be representative of the real-world use of the function. Thus, a set of input objects is gathered and corresponding outputs are also gathered, either from human experts or from measurements.
3. Determine the input feature representation of the learned function. The accuracy of the learned function depends strongly on how the input object is represented. Typically, the input object is transformed into a feature vector, which contains a number of attributes that are descriptive of the object. The number of attributes should not be too large, because of the curse of dimensionality, but it should contain enough information to accurately predict the output.
4. Determine the structure of the learned function and the corresponding learning algorithm. For example, the engineer may choose to use support vector machines or decision trees.
5. Complete the design. Run the learning algorithm on the gathered training set. Some supervised learning algorithms require the user to determine certain control parameters. These parameters may be adjusted by optimizing performance on a subset (called a validation set) of the training set, or via cross-validation.
6. Evaluate the accuracy of the learned function. After parameter adjustment and learning, the performance of the resulting function should be measured on a test set that is separate from the training set.

1.3 Document structure

The rest of the document is organised as follows:

- Chapter 2 explains how the data from a set of patients were processed and divided into a training and testing dataset.
- In chapter 3, three methodologies are studied to solve the problem to classify diabetic retinopathy patients.
- Chapter 4 presents the results obtained for the three methodologies of chapter 3. In addition, it will be seen the experimental setting of the parameters of the algorithms used for these methodologies.
- Chapter 5 explains how the methodology with the best results has been incorporated in a clinical application that could be integrated in the healthcare system.
- Chapter 6 summarises a list of conclusions of this work and devises some lines of future work. Moreover, the contributions of this work and publications are presented.

2 Data from diabetic patients

In this chapter it is exposed all the information related to the data used to subsequently train and test the methods. Firstly, it will be seen from where the data was extracted and what it contains. Secondly, it will be explained which data were used and how the data were processed so that they fit our program input. Finally, it will be shown how the training and testing set were generated and how the data was balanced for the non-balancing learning methods.

2.1 Extraction of the data

Sant Joan Hospital of Reus has been recording patient data since 2006 so they have an extensive database. This database contains the name of the patient and the values of all the attributes they measure to diagnose diabetes, for instance the BMI[3], creatinine or arterial hypertension. Moreover, they also record if the patient has developed diabetic retinopathy. They keep updating the data so that if a patient, for instance, increases the level of creatinine or is diagnosed diabetic retinopathy, it is recorded in the database.

According to the data obtained by the doctors, our field of study are the patients with Diabetes Mellitus from a Health Care Area (CAP Sant Pere) that depends on Sant Joan University Hospital from Reus, with a 7118 population censured as diabetic of a total of 106,772 inhabitants (urban in 68% and rural in 32%) and 73 Family Physicians. A total of 6,433 patients (90.37%) have been screened for DR annually since 2006. Data are available for such patients since 2006 from the periodic revisions of the primary care centre and the annual retina status.

The ophthalmology group of Sant Joan University Hospital from Reus could retrieve two groups of patients from that database in a matrix format. Each row corresponds to a patient and each column to an attribute. In the cells it can be found the value for the attribute of a specific patient. In order to preserve the anonymity of the patients they are identified by an id tag instead of their name. The following table shows an example of how the data is structured taking into account 3 patients and 6 attributes for illustrative purposes (A1, A2, .. , An refer to the attributes and P1, P2, .. , Pn to the different patients).

	A1	A2	A3	A4	A5	A6
P1	2	1,39	32,42	71	0	19
P2	6	1,12	29,94	76	1	15
P3	15	1,14	20,85	66	0	14

Table 1: Illustrative example of the data structure

[3] Body Mass Index

The first group received (group 1) was formed by both ill and healthy patients and it is the largest group since it contains more than 2000 registers. The second group of patients (group 2) was retrieved in order to get more ill patients, since in the group 1 they were a small part of the total set. Therefore, the group 2 contains more than 200 patients, all of which are ill.

2.1.1 Group of patients 1

The group 1 contains 2084 registers of patients. Although in the beginning there were more patients, some of the registers were dismissed because they had missing data, and the methods used in this work require all the data.

Analysing the first group, there are 1743 patients that have not been diagnosed diabetic retinopathy (healthy patients) and 341 patients that have been diagnosed with this condition (ill patients).

Taking these numbers into account, it can be easily seen that the amount of data of ill and non-ill patients is not equivalent and that could be problematic in the future. For this reason, the second group of patients is introduced.

2.1.2 Group of patients 2

In the group 2 there were 238 registers of patients. As in the first group, in the beginning there were more patients, but some of the registers were dismissed because they lacked some data. In this group, all the patients have been diagnosed diabetic retinopathy so all of them are classified as ill.

This set of registers helped us to shorten the gap between ill and non-ill patients in the group 1, and therefore have more information about the ill patients.

2.2 Data processing

The data of the patients of the hospital database is formed by many attributes of each patient. In this work, only 9 attributes are used plus the classification (if the patient is or healthy). These 9 attributes have been chosen by the doctors using previous studies[5]. In these studies they concluded that these 9 attributes are the ones with more relevance when classifying a patient, and for this reason they have been used in this work. In these attributes it is important to make a distinction between categorical and continuous data.

- **Continuous data** (also known as quantitative) is data where the values can change continuously, and the number of different values cannot be counted. Continuous data can take upon an infinite number of real values. Examples include weight, price, profits, counts, etc.
- **Categorical data** (also known as qualitative), in contrast, appear in those attributes that can take on one of a limited, and usually fixed, number of possible values, thus assigning each individual to a particular group or category. This includes product type, gender, age group, etc.

The data used in this work are both continuous and categorical attributes. Out of the 9 attributes, 3 of them are categorical and the remaining 6 are continuous. The different categories of the 3 categorical attributes have been labelled as 0, 1, 2, etc. For instance, if one attribute is gender, it is labelled 0 if the patient is a man and 1 if it is a woman. Moreover, the classifying attribute that indicates if the patient is ill or not (it

will be named it as RD) is also categorical, as it can take only two values: 0 if the patient does not have diabetic retinopathy and 1 if the patient is ill.

The following figure shows an example of the structure of the data with the 9 input attributes, the classifying attribute and the values for each patient-attribute pair. Note that the attributes that are categorical have been painted in red, while the continuous ones are shown in blue.

	A1	A2	A3	A4	A5	A6	A7	A8	A9	RD
P1	2	1,39	32,42	71	0	19	1	1	9,4	1
P2	6	1,12	29,94	76	1	15	1	1	10,7	1
P3	15	1,14	20,85	66	0	14	1	0	7,2	0

Table 2: Data format with 9 attributes and RD

The doctors studied the influence of the values in the classification of a patient [5]. This work concluded that it is not necessary to work with precise numbers (e.g. 32,42), because the range of the value (e.g. 30-40 interval) is informative enough. Therefore, it is better to work with categories, and not with accurate numbers, as then a qualitative diagnosis can be made. This fact resulted in the partition of the continuous attributes in some ranges fixed by the doctors [5], where each range corresponds to a category. In order to describe a detailed example, the following figure shows how the attribute A9 was categorised:

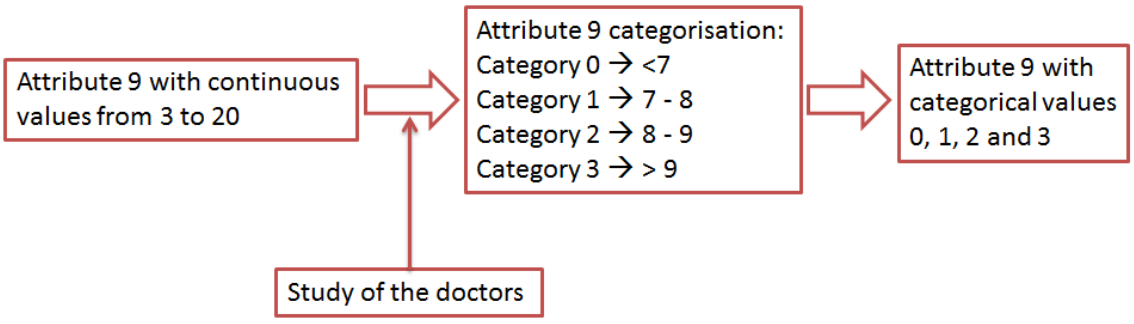


Figure 3: Categorisation process

As showed in the figure above, first there were the attribute 9 with a continuous value. Then, taking into account the categories defined by the doctors it is decided in which category attribute 9 is labelled according to its continuous value. Finally, attribute 9 contains only categorical values so that we can work with it. For instance, patient 1 from Table 2 has 9,4 as the value of A9. Therefore, it will be labelled in category 3. In the same way, patient 2 will be labelled to category 3 and patient 3 to category 1.

The mechanism of processing the data (calculating the category of the continuous attributes) has been automatized with a method in Java.

2.2.1 Class diagram to store the data

The following class diagram describes the Java structure used to store the data of the patients, showing the classes, the attributes and their relation:

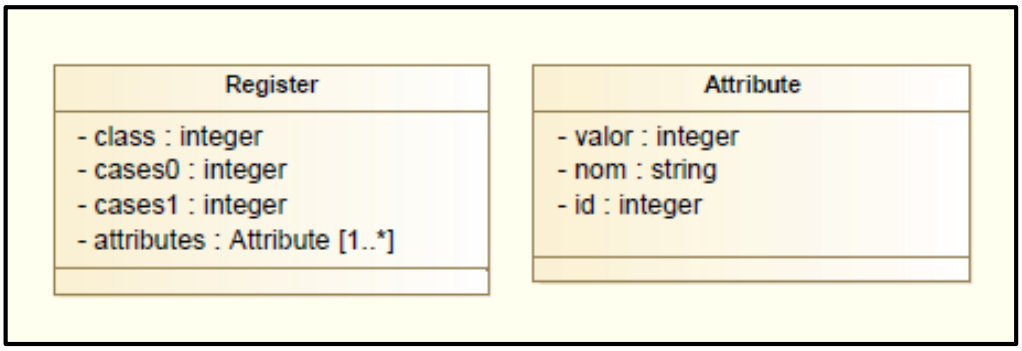


Figure 4: Class diagram for data storage

The class *Register* stores all the data of a patient. The attribute *class* is used to tag the patient as 0 if he or she is healthy or 1 if they are ill. The attributes *cases0* and *cases1* are used by the algorithm explained in section 3.2.1[4], since the patients that are equal (meaning that they have the same values for all the attributes) are joined into one register but keeping the number of original patients. As it may be the case that two equal patients belong to different classes, it is needed to differentiate it with these two counters. Finally, the class register contains an array of attributes that stores the specific value for each attribute of a patient.

The class *Attribute* contains the value of the specific attribute, its name, and the id of the attribute. The 9 input attributes have been tagged with an integer id to identify them more easily.

[4] See point 4) of this section.

2.2.2 How the program reads and stores the data

The method that reads and stores the data given in a text file is detailed in the following algorithm:

Algorithm. loadData

```
Inputs: file (text file where each line is a register, and contains
the value of the attributes ordered and separated by ";")
Outputs: registerList (list of Register objects that contains all the
patients of the file)
1   while (there are still lines to read) do
2           divide the line according to ";" and store it in a
            String array registerString
3           create object of the class Register register
4           calculate categorical values for each partition of
            registerString and add the attribute to register
5           add register to registerList
6   done
7   return registerList
```

Algorithm 1: loadData

The data of the patients are stored in a text file. Each row of the file corresponds to a patient and the different attributes are the columns, which are ordered and divided with ";" so that the program can identify the different parts. In this file, the continuous values have not yet been categorized.

The program does the following steps for each row of the file:

1. Reads the row, splits the line into the different attributes and creates an array of the class Attribute.
2. Transforms the continuous values into categorical values using the ranges calculated by the doctors and creates an Attribute. Note that each attribute has its own range. The program takes it into account automatically. Then, it puts the new attribute into the array of attributes.
3. Creates the class Register with the array of attributes and the class of the patient, and initializes to 0 the two counters.

It has to be taken into account that there may be patients with exactly all the same attribute values, and in order not to repeat information another step is performed. Once the entire file is read, all the registers are joined with the same data values and the counter of the class is used to know how many patients there were with this combination as exemplified in the following figure.

	A1	A2	A3	A4	A5	A6	A7	A8	A9	RD
P1	0	1	1	3	1	2	0	0	1	1
P2	0	1	1	3	1	2	0	0	1	1
P3	0	1	1	3	1	2	0	0	1	0
P4	0	1	1	3	1	2	0	0	1	1

	A1	A2	A3	A4	A5	A6	A7	A8	A9	Cases0	Cases1
P1	0	1	1	3	1	2	0	0	1	1	3

Figure 5: Example of joining patients with the same attribute values

2.3 Preparation of the training and testing set

- A **training set** is a set of data used to discover potentially predictive relationships.
- A **testing set** is a set of data used to assess the strength and utility of a predictive relationship.

In the following figure it can be seen how the training and testing sets are used:

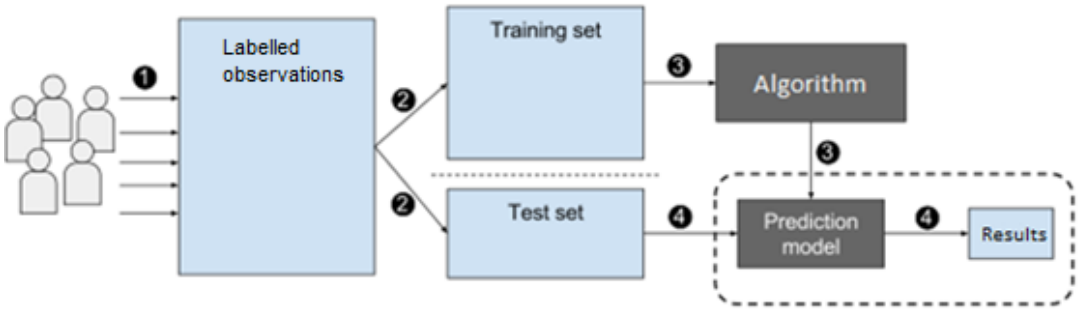


Figure 6: Supervised learning data flow

The figure shows the flow of data in a step-by-step process to train and test a prediction model. The steps shown in the diagram are the following:

1. Doctors collect data from the patients and create a set of labelled observations.
2. Labelled observations are split into a training set and a testing set. The training set is provided to the algorithm to learn from. The test set is held aside for future use.
3. The algorithm uses the training set to learn how to predict the provided labels and a prediction model is produced. This prediction model can be used to automatically label new observations.

4. The previously withheld test set is given to the prediction model and the predicted labels are compared to the patient labels to generate statistics about the model's performance.

As explained in section *2.1 Extraction of the data*, two groups of patients were given to work with: group 1 and group 2. These data were used to create the training and testing sets, trying to have two groups with a similar number of ill and healthy patients. As shown in the previous figure, the training set is usually a bit larger than the testing set, to take advantage of most of the patients to create the model.

Therefore, this partition was made:

- **Training set**: it is formed by half the negative (class 0 – non-ill) patients of the first group and all the positive (class 1 – ill) patients of the first group.
- **Testing set**: it is formed by half the negative patients of the first group and all the patients of the second group (which are positive).

This partition enables us to have similar groups. Even though there are still more negative than positive patients in both groups, the difference between them in the group 1 was significantly reduced. Moreover, with this group partition the training group contains more than half of the total patients (52.19%).

To sum up, two sets were defined with the following number of patients:

	Training set	Testing set
Number of 0 (non-ill patients)	871	872
Number of 1 (ill patients)	341	238
Total	1212	1110

Table 3: Training and testing set partition

2.4 Balanced data for non-balancing methods

In some cases it is better to work with unbalanced or balanced data, depending on the employed algorithm.

- **Unbalanced data** means that the number of observations (in our case patients) in each class (in our case class 0 and class 1) are significantly different.
- **Balanced data** means that there is approximately the same number of observations of each class.

In the training dataset described in Table 3, the data are unbalanced. For some of the methods used in this work, this fact is problematic because there are more registers from class 0 than from the class 1. For this reason, it is convenient to have a training set with balanced data.

There are two main different balancing techniques in a data level:

- **Undersampling**. Random undersampling is a non-heuristic method that aims to balance class distribution through the random elimination of majority class examples. The major drawback of random undersampling is that this method can discard potentially useful data that could be important for the induction process.
- **Oversampling**. Random over-sampling is a non-heuristic method that aims to balance class distribution through the random replication of minority class examples. Several authors [6], [7] agree that random over-sampling can increase the likelihood of overfitting, since it makes exact copies of the minority class examples. In this way, a classifier might construct rules that are apparently accurate, but actually cover one replicated example [8].

The oversampling technique has been used in this work because we do not want to lose information and also because the highest possible number of examples is needed. Therefore the balanced training set is formed by the following number of patients:

	Training set
Number of 0 (non-ill patients)	871
Number of 1 (ill patients)	871
Total	1742

Table 4: Balanced training set partition

In this set the number of ill patients is equal to the number of non-ill patients. Therefore, 530 random replications of patients from class 1 have been added.

2.5 Summary

In this chapter the doctors collected the data in a matrix form that stores each patient with its medical attributes and the two groups of patients received that contain both healthy and ill patients were specified. Moreover, it was necessary to categorize all the continuous attributes. Furthermore, the data is read and stored in our program and prepared the training and testing set by mixing and organizing the patients data. Finally, a balanced training set was created to use it in the methods that need it.

Once all the data is processed, stored and ready to use, different methodologies are introduced to create a classification model based on that supervised data.

3 Methods

In this chapter the different learning methods implemented to achieve the goals of the work are presented. All these methods use supervised learning. Therefore, there is a set of labelled training examples and the objective is to obtain a classifier model that can process future cases. There are several approaches to address this problem. Some of the most popular ones are the following:

- **Nearest neighbour algorithm**. This is the simplest method of classification. In this method, the class of an object is determined by the class of the nearest object. Therefore, if we have a set of examples $X = \{x_1, x_2, \ldots, x_N\}$ and we want to classify an object x, we need to find the nearest object x_i in X. Then, it is concluded that the class of x is the one of x_i. In order to implement this method and determine the nearest neighbour it is necessary to define a distance between the elements of the domain. A generalization of this method is the *k*-nearest neighbour (K-NN), where the *k* nearest neighbours are taken into account. Thus, the class of x is the most frequent class among the *k* selected objects [9].
- **Support vector machines (SVM)**. A SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces [10].
- **Decision trees**. A decision tree builds classification models in the form of a tree structure. The attributes are used to create partitions of the dataset of examples, as the nodes correspond to the attributes and the branches of a node represent the possible values of the associated attribute. The leaf nodes are a set of classified examples and represent a classification or decision [11]. Some techniques, such as Random Forest, construct more than one decision tree in order to improve the classification rate.

Out of these types of supervised learning, the decision trees are the most interesting ones. Using them, a qualitative model can be made based on categorical values and they also enable us to extract rules and therefore understand why the final decisions are taken.

In this work the Logistic Regression was studied, which is a statistical method used initially by the doctors, the decision trees and random forest. Not only were these methods studied and implemented, but they were also modified so that they adjust better to our needs.

3.1 Logistic Regression

The logistic regression is a statistical classification model that predicts a binary response from a binary predictor to anticipate the outcome of a categorical dependent variable. The logistic regression measures the relationship between a categorical dependent variable and one or more independent variables. Logistic regression provides a flexible and easy-to-interpret method for building models from binary data [12].

This is the classification method used by the ophthalmologists of the hospital before the start of this work, so it was taken as the reference baseline. As logistic regression does not balance the data for itself, the balanced training set was used for this method.

The statistical package *XLSTAT* was used to calculate the regression function with the following parameters:

- Model: Logit → to indicate that we want to use a logistic regression function. The analytical expression of the model is as follows:

  $$p = 1 / (1 + exp(-\beta X))$$

  In this expression, βX represents the linear combination of variables (including constants).
- Type of answer: binary → in our case we want to decide if the patient belongs to class 0 or to class 1.
- Confidence interval: 95 % → a 95% confidence interval is a range of values that, with a 95% of certainty, contains the true mean of the population.
- Stop conditions: 100 iterations and 0.000001 of convergence → The analysis stops after 100 iterations or when the error is smaller than 0.000001.
- Algorithm for the maximization of the likelihood function: Newton-Raphson → Contrary to linear regression, an exact analytical solution does not exist, so an iterative algorithm has to be used. The Newton–Raphson method finds successively better approximations to the roots (or zeroes) of a real-valued function.

With the logistic regression function a model equation is obtained for the variable DR that, given the values of the 9 studied attributes, is able to determine if the patient belongs to class 0 or to class 1. For confidentiality reasons we cannot publish the constants that multiply the attributes in the function, but here it is the main structure of the model:

$$Pred(DR) = \frac{1}{1 + \exp(-(k1 * A1 + k2 * A2 + k3 * A3 + k4 * A4 + k5 * A5 + k6 * A6 + k7 * A7 + k8 * A8 + k9 * A9))}$$

If the outcome of the function is lower or equal than 0.50 then the patient is predicted as class 0 (healthy). On the other hand, if the result is higher than 0.50 the patient is predicted as class 1 (ill).

3.2 Classification model based on Decision trees

A decision tree is a schematic tree-shaped diagram used to determine a course of action or to show a statistical probability. The attributes are used to create partitions of a set of examples. The tree nodes are the names or identifiers of the attributes, while the children of a node represent the possible values of the attribute associated to the node. The leaf nodes are sets of classified examples [11]. The tree structure shows how one choice leads to the next, and the use of branches indicates that each option is mutually exclusive, which means that two options cannot occur at the same time [13].

The strategy to build the tree consist of selecting, in each moment, the attribute that is potentially more useful for the classification, that means that it is the one that can generate a better tree from this moment on.

Then, when an object is received to be classified, we go through the tree from the root node to a leaf node that classifies the object. The process to reach the leaf is recursive and works in the following way. Given a node, a child node is selected, dismissing the others [9]. In the following figure you can find an example of a decision tree[5].

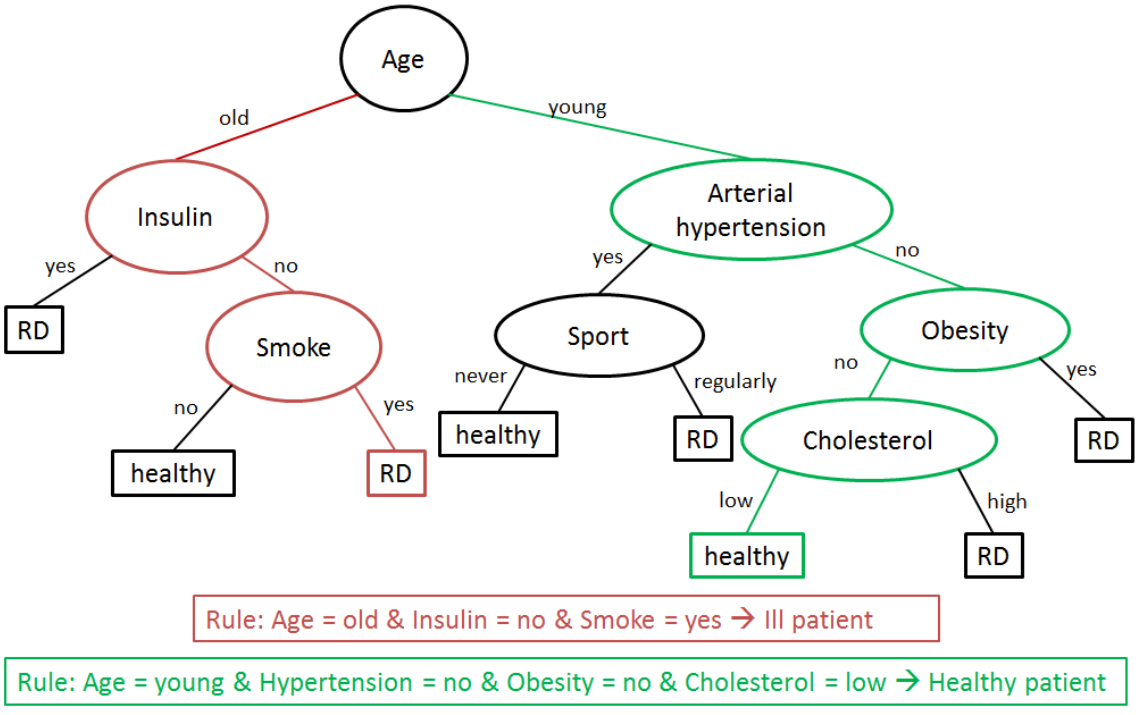


Figure 7: Decision Tree example

[5] Note that this tree is an example and they are not the attributes used to create our final model.

In the figure the branches represent the decisions on the attribute of the father node. Thus, in a decision tree the depth indicates the conjunction and the width the disjunction. Moreover, rules can be extracted from the tree. In particular, it can be seen that there is one rule for each leaf node. In order to generate the rules, we have to trace each path in the decision tree from the root node to a leaf node.

There are different kinds of decision tree induction algorithms. One of the most popular is ID3, which is the one used in this work. In the following section the algorithm is explained in a detailed way.

3.2.1 The ID3 algorithm

This algorithm was first developed by Quinlan [14] and it belongs to the family of TDIDT[6]. Each element or instance of the input sequence given to the algorithm takes the form of a list of attribute-value pairs. Each instance also has the class to which it belongs. The objective is to build a decision tree following the criteria explained in the previous section.

The algorithm builds the tree by choosing the best attribute in each node[7]. Let $\boldsymbol{\chi}$ be the initial set of instances, $\boldsymbol{A}$ the set of attributes that describe these instances and $\boldsymbol{C}$ the possible classes. From now on, when in any node, $\boldsymbol{X}$ will refer to a set of instances (subset of $\boldsymbol{\chi}$) such that their values coincide with the ones on the path from the root to that node for the attributes involved. For the root node we have $\boldsymbol{X} = \boldsymbol{\chi}$. Given an attribute $\boldsymbol{A}$, an element $\mathbf{x} \in \boldsymbol{\chi}$ and a value $\boldsymbol{v}$, it can be defined:

$$V(A) = \{possible\ values\ of\ A\}$$

$$A(x) = x\ value\ for\ A$$

$$A^{-1}(X, v) = \{x\ \in \mathrm{X} \mid A(x) = v\}$$

$$P_c(X) = partition\ of\ X\ in\ the\ classes\ of\ C$$

[6] Top-down induction of decision trees

[7] Later we will see the function used to determine the best attribute.

The following algorithm describes the basic outline of the construction of the tree.

Algorithm. ID3

```
Inputs: X (set of instances), A (set of the remaining attributes),
threshold (it indicates when a leaf node can be created)
Outputs: dt (decision-tree)
1   var tree1, tree2: decision-tree
2   if (∃C∀x∈X: x∈C or (#x∈X: x∈C)/(#x∈X) > threshold)[8] then
3     tree1 := create_tree(C);
4   else
5     if A≠∅ then
6           a_M := max{G(X, a)}_a∈A;
7           tree1 := create_tree(a_M);
8           for all v ∈ V(a_M) do
9                 tree2 := ID3(A^-1(X,v), A-{a_M});
10                tree1 := add_branch(tree1, tree2, v);
11          ffor all
12    else tree1 := create_tree(majority_class(X));
13    fif
14  fif
15  return tree1;
```

Algorithm 2: ID3 algorithm

The initial call of the algorithm is ID3(χ, ***A, threshold***). As it can be seen, the algorithm uses the following auxiliary functions:

1) **create_tree(Y)**: returns a decision tree that is formed by only one node tagged with class Y.
2) **G(X, a)**: it is the selection function, which has the maximum value for the attribute that is considered to be the best to continue the classification. Depending on the chosen selection function, different trees may be obtained. The function used in this work is based on the Shannon entropy. The information gain is given by the following function:

$$G(X, A) = I\left(P_C(X)\right) - E(X, A)$$

[8] If all the instances of *X* belong to the same class *C* or the percentage of instances of *X* that belong to *C* exceeds the threshold.

$I(P_C(X))$ estimates the randomness of the distribution of the instances X in the classes C:

$$I\left(P_C(X)\right) = -\sum_{C \in P_C(X)} p(X,C)\, log_2 p(X,C)$$

$$p(X,C) = \frac{number\ of\ X\ of\ class\ C}{total\ number\ of\ X}$$

p(X,C) is the probability for a specific instance of X of belonging to C. This probability is the proportion of the elements of X that also belong to C.
If a set of elements of X is partitioned according to the values of a specific attribute A, the entropy for that attribute can be obtained:

$$E(X,A) = \sum_{x \in Part(X,A)} \frac{number\ of\ x}{number\ of\ X}\, I(P_C(x))$$

Part(X,A) represents the partition of X in classes using the values of A. The E function estimates the randomness of the distribution of the instances in the classes, given the value of attribute A. This estimation is done by means of a weighted average.
Finally, the G function selects the attribute A that minimises E(X,A), since $I(P_C(X))$ is the same for all the attributes.

3) **add_branch(X, Y, Z)**: returns the resulting tree when it is added to the tree X a new branch Y tagged with the value Z, which is one of the possible values of the last attribute of the tree X.
4) **majority_class(X)**: returns the majority class of the elements of X [11]. This function is used when there are no attributes left to analyse. In our case, it may be that there are several instances with the same values in the attributes but they belong to different classes.

3.2.2 Modification of classic ID3

As it was explained before, the algorithms were modified so that they can be properly applied to our data and domain. Thus, the following modification has been introduced. The basic algorithm of ID3 does not include the *threshold*[9]. In our case, given that it is difficult that all the patients belong to the same class, the algorithm can also create a leaf node when the percentage of instances of a class exceeds a certain threshold. Therefore, it is not necessary that all the instances belong to the same class, it is just needed that the percentage of instances belonging to a class exceeds the threshold. In the algorithm above, this idea is implemented in the condition of line 2. In Chapter 4 different thresholds were tested in order to see which of them provided a better performance.

For this method the balanced training set was used, as the algorithm is not able to balance it itself.

[9] See line 2 Algorithm 2.

3.3 Classification model based on Random Forest

While a decision tree has many advantages, such as comprehensibility and scalability, it still suffers from several drawbacks—instability, for instance. One way to realize the full potential of decision trees is to build a decision forest [15]. In the Random Forest method several decision trees are constructed and the final decision takes into account the predictions of all the trees.

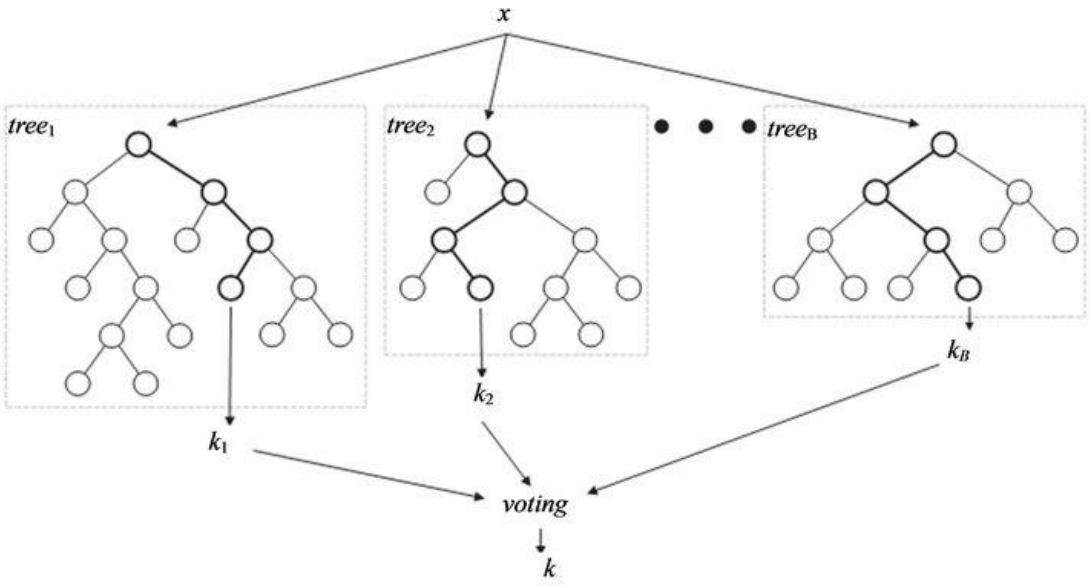


Figure 8: General architecture of a Random Forest

As shown in the above figure [16], B trees are constructed and the new instance x is evaluated by all of them. Eventually, the final classification is done by a voting function that receives the results of all the trees.

Here is how each tree of the random forest is obtained from a training data set T:

1. Pick up randomly N items of the training data. Some studies suggest that N should be around two thirds of the training set [17]. As we want to have a balanced set of items to build each tree, 340 patients are taken from each of the two classes, for a total number of 680 items (56% of the training set).
2. At each node:
    a. m attributes are randomly selected from all the ones that have not been used yet in that branch. Previous works suggest that this number should be around log(number of attributes) [15].
    b. The entropy of each of these attributes is computed to determine the one that classifies better the training examples remaining in that branch and successors nodes for each of its values are created. The process stops (and a leaf of the tree is created) when, considering the combinations covered by the branch, the percentage of individuals of the training set from one class exceeds a given threshold. An “unknown” label is given to a leaf if there are not any more attributes to consider and none of the two classes exceeds the threshold. This is the difference between the Random Forest trees and the ID3 trees used in the previous section. In

the ID3, when there were no attributes left, the leaf node was tagged with the majority class. In the Random forest trees, it can be tagged as one of the classes if they exceed the threshold, but if none of the classes does, the leaf node is tagged as unknown.

In the Random Forest the unbalanced training set is used since in step 1 the data are balanced by the algorithm.

The standard RF technique has two basic parameters: the number of trees of the forest and the number of attributes considered in each node. However, in our case we have the following parameters that can be changed:

- **Number of trees (nTrees)**: number of trees that are built. The theoretical studies show that when more trees are added the algorithm improves until you reach a point where it gets worse. Our objective is getting at that point to determine the best configuration.
- **Number of attributes (m)**: number of attributes taken into account when choosing the attribute to make a node. Note that these attributes will be chosen randomly from the set of attributes still not used on that branch.
- **Number of negative registers (nRegisters0)**: number of negative registers that have to be selected randomly.
- **Number of positive registers (nRegisters1)**: number of positive registers that have to be selected randomly.
  In our case, the parameters *nRegisters0* and *nRegisters1* were fixed to 340 because it is how the data can be best balanced and also respecting the recommendation of using around 66% of the total training set.
- **Threshold (threshold)**: specifies the minimum percentage of registers from one class to tag the node with that class.

In chapter different values are considered for these parameters and it is determined which are the best ones in an experimental way.

The Random Forest algorithm to build the trees is described below and uses the ID3 algorithm explained in the previous section.

Algorithm. RandomForest

```
Inputs: X (set of instances), A (set of the remaining attributes),
threshold (it indicates a leaf node can be created), nTrees, m,
nRegisters0, nRegisters1, threshold[10].
Outputs: rnodes (array of the root nodes of the trees)
1   var i:int, x:array_instances, root:node;
2   i := 0;
3   while ( i < nTrees ) do
4      x := pick_random_registers(X, nRegister0, nRegister1);
5      root := ID3(x, A, threshold);
6      rnodes := add(rnodes, root);
7      i := i + 1;
8   done
9   return rnodes;
```

Algorithm 3: Random Forest algorithm

3.3.1 Classification using a Random Forest: Voting function

A Random Forest is a set of *n* different trees. When a new instance (patient in our case) has to be classified, it is necessary to use a voting methodology that takes into account all the tree predictions and makes a final decision.

The idea of the voting function used in this work is to decide what the majority of the trees suggest. Therefore, if most of the trees classify the new patient as class 0, 1 or unknown, the final decision will be that majority. More technically, we have the root nodes of each tree, a counter for the negative decisions, a counter for the positive decisions and another one for the unknown decisions. For each new patient (instance):

1. The counters are initialized at 0.
2. The decision of each tree is calculated from its root node. The positive/negative/unknown counter are incremented depending on the decision obtained.
3. The majority is calculated according to the values of the counters, and a final decision is made. In case that the majority of the trees return unknown, the patient class will be unknown because there is not enough information to decide. If there is a tie in the number of predictions, the preference is in the following order: unknown, 0, 1.

This method also allows having a probability associated to the decision. This probability is calculated as the number of decision trees that took the majority decision divided by the total number of trees. For instance, if 30 trees predicted class 0, 15 trees predicted class 1 and 5 trees predicted unknown, the classification is class 0 with a probability of 60%.

[10] Parameters described above.

In the following figure it is shown how a patient would be classified as healthy (0), ill (1) or unknown according to the random forest, and how the probability is extracted.

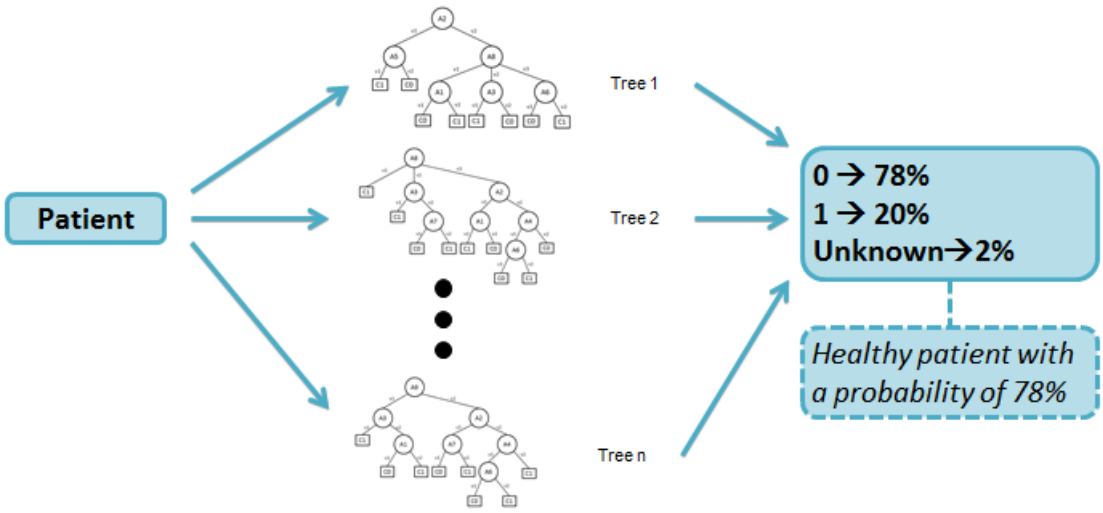


Figure 9: Voting method of the Random Forest

3.4 Comparison between Decision Trees and Random Forest

In this section, the two techniques we have focused on are compared: decision trees and random forest. Before evaluating each method in section 4, the advantages and disadvantages of using one or another are analysed.

The advantages and disadvantages of using a decision tree are the following[18].

Advantages of Decision Trees	Disadvantages of Decision Trees
They are simple to understand and interpret. It is easy to extract rules from the tree and understand why a specific prediction was made.	For data including categorical variables with different number of values, information gain in decision trees is biased in favour of those attributes with more values.
Important insights can be generated based on experts describing a situation and their preferences for outcomes.	Trees do not usually have optimal performance when compared to other methods.
Trees can be computed very quickly.	Small changes in the data can drastically affect the structure of a tree.
If a predictor was not used in any split, the model is completely independent of that data.	Overfitting the training data.

Table 5: Pros and cons of a single Decision Tree

The last disadvantage mentioned on the table has been exploited to improve the performance of the trees via ensemble methods where many trees are fit and predictors are aggregated across the trees. These methods include Random Forest, which is the method analysed in the following table.

Advantages of Random Forests	Disadvantages of Random Forests
It is one of the most accurate learning algorithms available. For many data sets, it produces a highly accurate classifier.	A large number of trees may make the algorithm slow for real-time prediction.
It generates an internal unbiased estimate of the generalization error as the forest building progresses.	They are difficult to understand and interpret.
Good algorithm to use for complex classification tasks.	
Less overfitting.	

Table 6: Pros and cons of Random Forest

As we have a very complex and incomplete dataset[11], it makes us think that Random Forest will adapt better to our data. The fact that different trees are generated will make the model consider different possibilities before taking a final decision. The random forest uses random subsets of the training set and that could be useful to find the best solution in general. However, several tests have been made with both models to see which has a better performance, as will be shown in the next chapter.

3.5 Summary

In this chapter, three different methods have been proposed to solve the problem explained in section 1.

First, the Logistic Regression was explained, which was the method used initially by the doctors. It has been explained how the formula is obtained and the parameters used to do it. This method has been the baseline of our work.

Then, the decision trees have been studied. The method to build a general tree and specifically the algorithm that we used, ID3, were described with all the parameters and auxiliary functions needed. Also it has been explained how a new patient is evaluated by the model.

Furthermore, the Random Forest method, that uses decision trees, was studied. It has been explained how the set of trees is constructed and the parameters that have to be determined. Moreover, with the Random Forest, a new patient is evaluated by means of the voting function.

Finally, decision trees and random forests were compared before testing them in the following chapter. Chapter 4, shows the performance of each technique and the best parameter values for both of them.

[11] Incomplete meaning that we don't have patient information for all the possible data combinations.

4 Evaluation

In this chapter the optimal values for the parameters of the Decision Trees and Random Forest methods were determined. Moreover, the results obtained with the three studied methods are compared (Logistic Regression, Decision Trees, Random Forest).

In all the tests described in this chapter, the same evaluation measures were used. Given that the test outcome can be positive (classifying the person as having the disease) or negative (classifying the person as not having the disease), the test results for each subject may or may not match the subject's actual status. Therefore, the following four possibilities must be considered:

- **True positive**: Sick people correctly identified as sick (correctly identified).
- **False positive**: Healthy people incorrectly identified as sick (incorrectly identified).
- **True negative**: Healthy people correctly identified as healthy (correctly rejected).
- **False negative**: Sick people incorrectly identified as healthy (incorrectly rejected).

Therefore, the evaluation measures used in this work are the following:

- **Sensitivity**: measures the proportion of positives that are correctly identified as such. In our case it is the percentage of sick people who are correctly identified as having the condition.
- **Specificity**: measures the proportion of negatives that are correctly identified as such. In our case it is the percentage of healthy people who are correctly identified as not having the condition.
- **Accuracy**: is the proportion of true results (both true positives and true negatives) among the total number of cases [19].

The following table shows a graphical representation of these measures:

		Condition	
		Condition positive	Condition negative
Test outcome	Test outcome positive	**True positive**	**False positive** (Type I error)
	Test outcome negative	**False negative** (Type II error)	**True negative**
		Sensitivity = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	**Specificity** = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Accuracy = $\frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$

Table 7: Sensitivity, Specificity and Accuracy.

The objective of this work is to achieve around 80% of both sensitivity and specificity, since it is the minimum percentage acceptable for a medical application according to the doctors. Therefore, when setting the parameters and contrasting the results, the best configuration will be considered as the one whose values for specificity and sensitivity are closer to 80%.

4.1 Classification results of Logistic Regression

In this section the performance of the Logistic Regression model that was calculated in section 3.1 is shown. As explained in chapter 3, for this method the balanced training set was used. Here it is the structure of the function used in the tests.

$$Pred(DR) = \frac{1}{1 + \exp(-(k1 * A1 + k2 * A2 + k3 * A3 + k4 * A4 + k5 * A5 + k6 * A6 + k7 * A7 + k8 * A8 + k9 * A9))}$$ [12]

As mentioned in previous chapters, the Logistic Regression model was the baseline in our work, as it was the function initially used by the doctors. For this reason, for this method we did not experiment with the parameters.

The result table is the following.

	Real 1	Real 0	
Predicted 1	**198**	**40**	
Predicted 0	**187**	**686**	
	Sensitivity = 51,42%	**Specificity = 94,49%**	**Accuracy = 79,56%**

Table 8: Result table for Logistic Regression

In the result table, it can be seen that the regression function provides a high specificity, as there is a low number of False Positives. However, the sensitivity hardly exceeds 50%. For this reason, this formula is not useful for the doctors, since it fails to provide a good model to predict the positive patients.

4.2 Classification results of Decision Trees

In this section the results obtained with Decision Trees with the balanced training set are shown. As it was explained in algorithm 2, the decision tree induction algorithm ID3 allows one parameter, the threshold, which specifies the minimum percentage of registers from one class to tag the node with that class. An empiric experiment was done to test the results with different thresholds.

[12] Where A_n are the attributes and k_n the constants of the function. Note that we have not published the constant values and attributes for data privacy reasons.

The tested thresholds were in the range from 60% to 100%:

Tested Thresholds								
60	65	70	75	80	85	90	95	100

Table 9: Tested thresholds for Decision Trees

The following table shows the values of sensitivity, specificity and accuracy for each threshold.

Threshold	Sensitivity (%)	Specificity (%)	Accuracy (%)
60	0	100	78,57
65	55,88	63,30	61,71
70	62,18	62,73	62,61
75	62,18	62,73	62,61
80	67,65	66,05	66,40
85	66,81	66,17	66,30
90	68,49	64,22	65,13
95	64,29	62,66	63,01
100	64,29	62,66	63,01

Table 10: Table of results for the Decision Trees thresholds

The following figure shows the results in a graphical way. In the x-axis we can find the different values of the threshold, and in the y-axis the percentage regarding to each measure (sensitivity, specificity and accuracy).

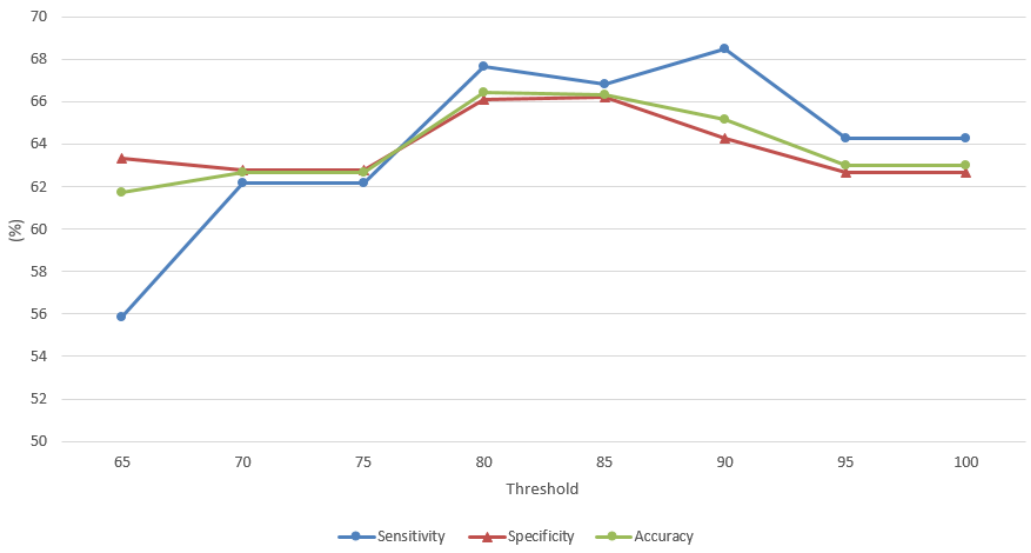


Figure 10: Results for the Decision Trees thresholds

As we can see in the results, with the 60% threshold the system is unable to classify any patient in class 1. This could be because the percentage is too low to make an appropriate choice with enough information. The algorithm improves from the 70% threshold, since in the following thresholds, the percentage of sensitivity and specificity is always around 60%. Here are the specific results for the best threshold, which was **90%**:

	Real 1	Real 0	
Predicted 1	**163**	**312**	
Predicted 0	**75**	**560**	
	Sensitivity = 68,49%	**Specificity = 64,22%**	**Accuracy = 65,13%**

Table 11: Results of the best Decision Tree

This method is an improvement in relation to Logistic Regression, as it is able to equilibrate more the percentage of sensitivity and specificity. However, if we focus in our objective, the results are far from the 80% requested by the doctors. For this reason, the next method is introduced, which uses decision trees in a different way.

4.3 Classification results of Random Forests

In this section it is presented the performance of Random Forests, that uses the non-balanced training and testing set as it is a balancing method. To do so, first it is shown how the optimal values for the parameters of the Random Forest method were experimentally determined. Note that Random Forests is a random technique and every time the results may vary. For this reason each test has been done 3 times and the result shown for all the measures is the average of the 3 tests.

The standard RF technique has two basic parameters: the number of trees of the forest and the number of attributes considered in each node. Moreover, in this case it is also necessary to determine the value of the threshold that controls the creation of the leaves of the tree[13]. As explained in section 3, two more parameters were introduced: nRegisters0 and nRegisters1 (number of registers used of class 0 and class 1 respectively) that were fixed to 340 in all the tests. For testing this method, the non-balanced data were used, since these two parameters (nRegisters0 and nRegisters1) allow balancing the data.

Let us start the analysis with the threshold. To analyse this parameter the other two were fixed at a standard value. Tests were made with values between 60% and 95% for the threshold, taking 200 trees in the forest and 2 attributes in each node. The following table shows the values for each parameter and highlights the analysed parameter, with all the values that were tested.

[13] See section 3.3 for the detailed description of those parameters.

Parameter	Value/s
Threshold	60 – 65 – 70 – 75 – 80 – 85 – 90 – 95
Number of trees	200
Number of attributes	2

Table 12: Threshold analysis parameters

The method was executed for all the values of the threshold shown in the table and with the other two parameters fixed. As we saw in the introduction of this chapter, the sensitivity, specificity and accuracy of this method were analysed. Moreover, in this method the number of unknown predictions was also analysed, since it is interesting to make it as low as possible. The following table shows these four measures for all the tests done with the possible values of the threshold.

Threshold	Sensitivity (%)	Specificity (%)	Accuracy (%)	Unknown (%)
60	83,19	72,82	75,04	0
65	80,67	78,32	78,82	0
70	77,87	82,4	80,63	0,99
75	72,41	86,3	80,99	2,79
80	61,63	89,93	78,91	5,67
85	45,98	92,76	76,3	7,56
90	28,36	94,09	72,79	9,54
95	24,27	95,32	72,43	10,18

Table 13: Numerical results for the threshold analysis

In the following figure these results can be seen more graphically.

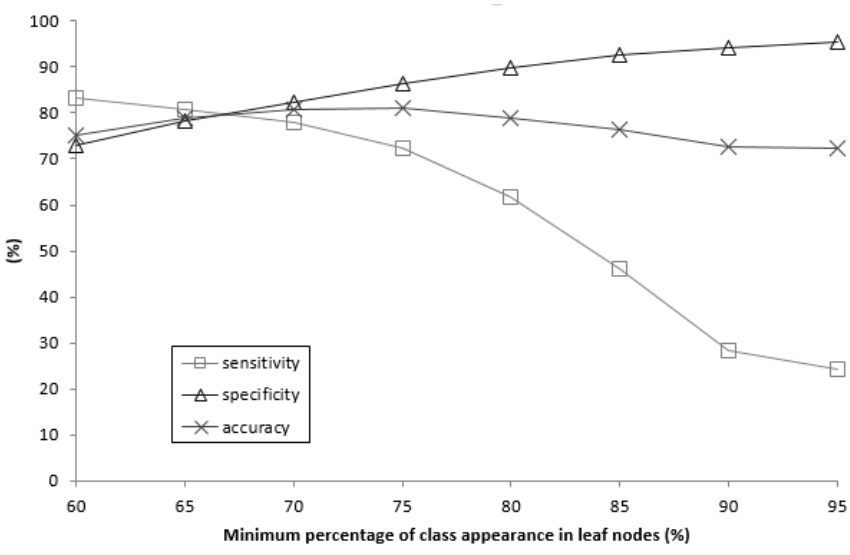


Figure 11: Graphical results for the threshold analysis

Analysing the results, with a value of 68, the three evaluation measures are closer to 80%. With a higher value it is possible to increase specificity keeping a good accuracy, but there is a very strong decrease in sensitivity. With a lower value we exceed 80% for the sensitivity but both accuracy and specificity drop. Analysing the percentage of unknown predictions in the following graphic, we can see that it increases when the threshold is increased. However, the percentage is always below 10% and for the selected threshold (68%) it is near 0.

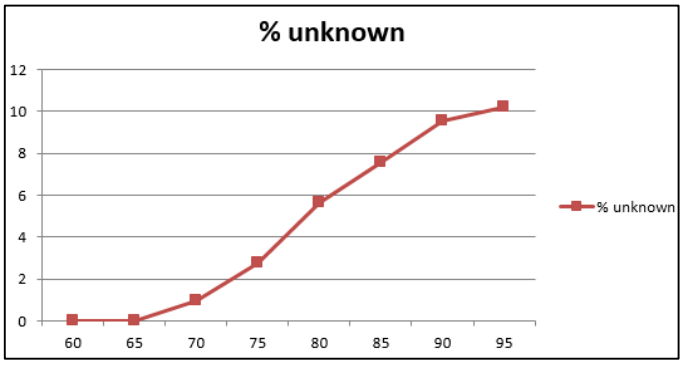


Figure 12: Unknown percentage for the threshold analysis

From now on, the threshold parameter is fixed to 68%.

On the second place, the influence of the number of attributes considered in each node of the tree were studied. In the tests we tried the values from 1 to 4, with the threshold 68% and 200 trees in the RF. The following table shows the values for each parameter and it is highlighted the analysed parameter and all the values that were tested.

Parameter	Value/s
Threshold	68
Number of trees	200
Number of attributes	1 – 2 – 3 – 4

Table 14: Number of attributes analysis parameters

The following table shows the four evaluation measures for all the tests done according to the different number of attributes.

nAttributes	Sensitivity (%)	Specificity (%)	Accuracy (%)	Unknown (%)
1	78,72	80,85	79,81	0,72
2	78,72	79,79	78,91	0,81
3	80,42	80,96	80,27	0,72
4	79,14	80,46	79,45	0,90

Table 15: Numerical results for the number of attributes analysis

In the following figure these results can be seen more graphically.

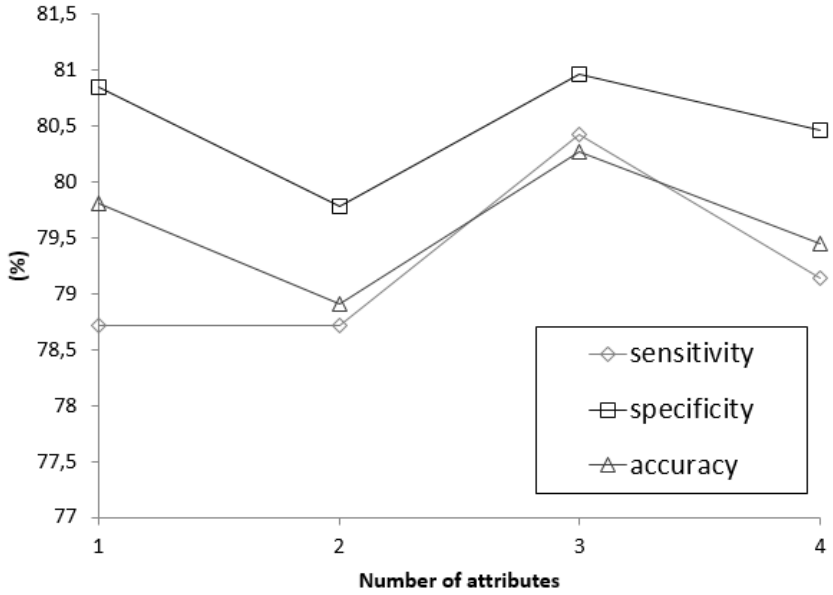


Figure 13: Graphical results for the number of attributes analysis

The results show that 3 is the only value for which the three evaluation measures exceed 80%. The specificity is pretty high for all the values. However, the sensitivity and accuracy are always below 80% with the exception of the value 3. For this reason, the value of 3 was chosen as the optimal number of attributes.

Analysing the percentage of unknown in the following graphic, it can be seen that it is lower for the values 1 and 3 and it increases for 2 and 4. However, the percentage is always below 1% and for the selected number of attributes (3) is near 0.

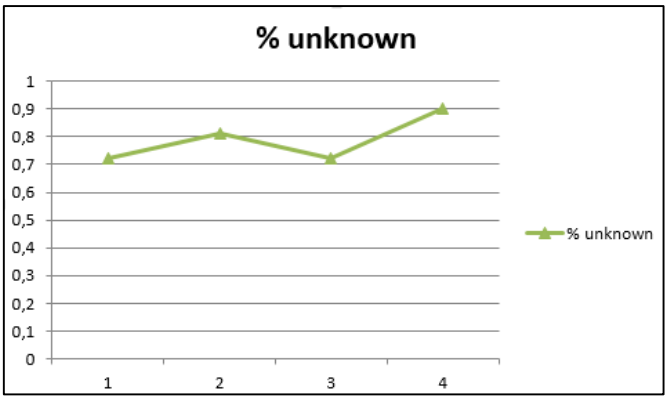


Figure 14: Unknown percentage for the number of attributes analysis

From now on, the number of attributes parameter is fixed to 3.

Finally, the influence of the number of trees in the RF was analysed, taking the 68% threshold and 3 attributes in each node. The values considered in the study were 50, 100, 200 and 300. The following table shows the values for each parameter and it is highlighted the analysed parameter and the values that were tested.

Parameter	Value/s
Threshold	68
Number of trees	50 – 100 – 200 – 300
Number of attributes	3

Table 16: Number of trees analysis parameters

The following table shows the results of the four evaluation measures for all the tests made with different number of trees.

nTrees	Sensitivity (%)	Specificity (%)	Accuracy (%)	Unknown (%)
50	76,59	78,93	77,65	0,99
100	78,72	80,6	79,54	0,81
200	80	80,96	80,18	0,72
300	79,14	80,25	79,36	0,81

Table 17: Numerical results for the number of trees analysis

In the following figure these results can be seen more graphically.

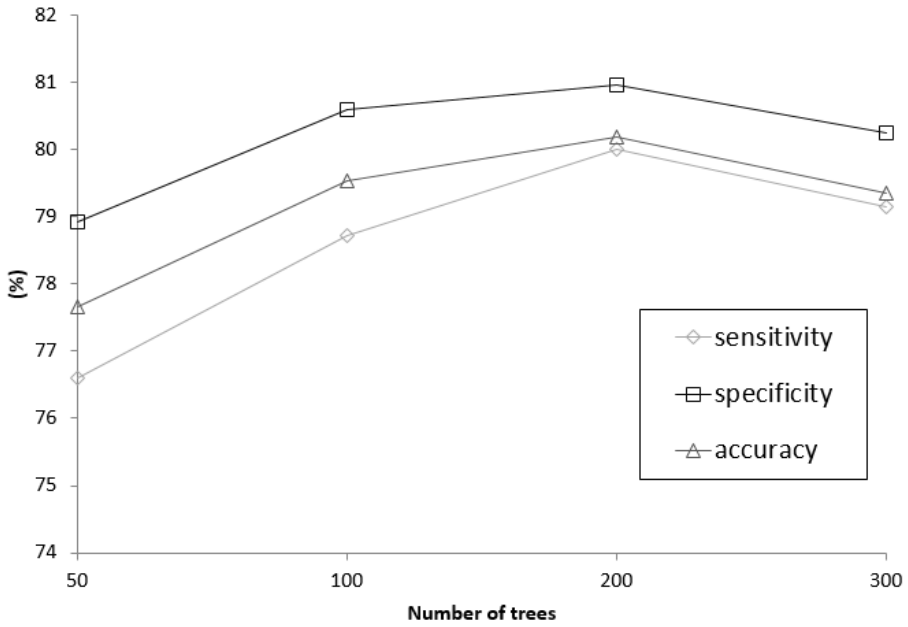


Figure 15: Graphical results for the number of trees analysis

As seen in Figure 15, the best performance of the three evaluation measures was reached when **200** trees were considered. The three measures increase when the number of trees is increased until 200 is reached, the higher point. From that point on the three measures decrease. In the 200 point the 80% for the three measures is reached, which was the objective of this work.

Analysing the percentage of unknown predictions in the following graphic, it can be seen that it decreases until 200 is reached and from that point on it increases. The percentage is always below 1% and for the selected number of trees (200) is near 0 and it is the lower percentage for unknown.

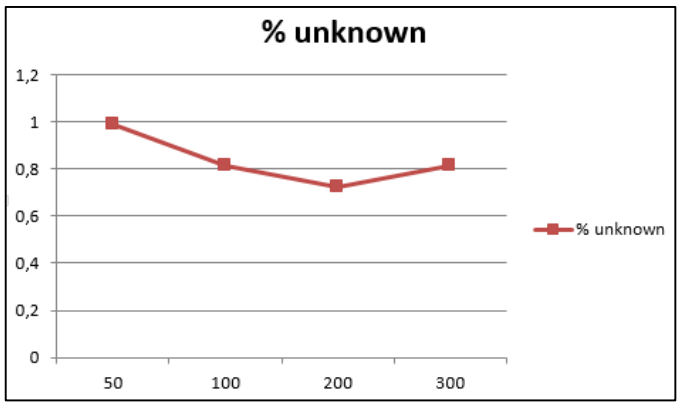


Figure 16: Unknown percentage for the number of attributes analysis

Given that the threshold was the first parameter analysed, the different thresholds for 3 attributes and 200 trees were tested again. This is to make sure that 68% is still the best result.

In summary, the final RF setting considered 200 trees, 3 randomly selected attributes in each node and a minimum leaf-creation percentage of 68% as it is shown in the following table.

FINAL CONFIGURATION	
Threshold	68%
Number of attributes	3
Number of trees	200

Table 18: Random Forest final configuration

The table of results for this configuration is the following. For this method we have added the unknown predictions (tagged as -2). When calculating the sensitivity, specificity and the accuracy the patients predicted as unknown have not been counted.

	Real 1	Real 0	
Predicted 1	**188**	**165**	
Predicted 0	**47**	**702**	
Predicted -2	**3**	**5**	**Unknown = 8 out of 1110**
	Sensitivity = 80.0%	**Specificity = 80.96%**	**Accuracy = 80.18%**

Table 19: Classification results for Random Forest

Table 19 shows the classification results. The system is able to make a prediction in almost all of the cases (it only fails to make a prediction in 8 out of 1110 patients, 0.72%). The values of specificity and sensitivity reach 80%, whereas the global accuracy of the predictions is 80.76%.

Analysing the results, the objective to have 80% in the three measures (sensitivity, specificity and accuracy) has been achieved and moreover the number of unknown predictions is really low. The unknown classification allows identifying those patients for which there is not enough information to classify them and prevent the system to make a "random" decision and fail in the prediction. The fact that the unknown patients are low means that the model is able to classify most of the patients.

4.4 Comparison of the methods

In this section the best results of each method (Logistic Regression, Decision Trees and Random Forest) are compared to see which one has the best performance. For the comparison the three measures are used again: sensitivity, specificity and accuracy. The objective is to get 80% for each of the three measures

	Logistic Regression	ID3	Random Forest
Sensitivity	51,42%	68,49%	80,0%
Specificity	94,49%	64,22%	80,96%
Accuracy	79,56%	65,13%	80,18%

Table 20: Comparison of the measures of the classification methods

The result is shown more graphically in the following figure:

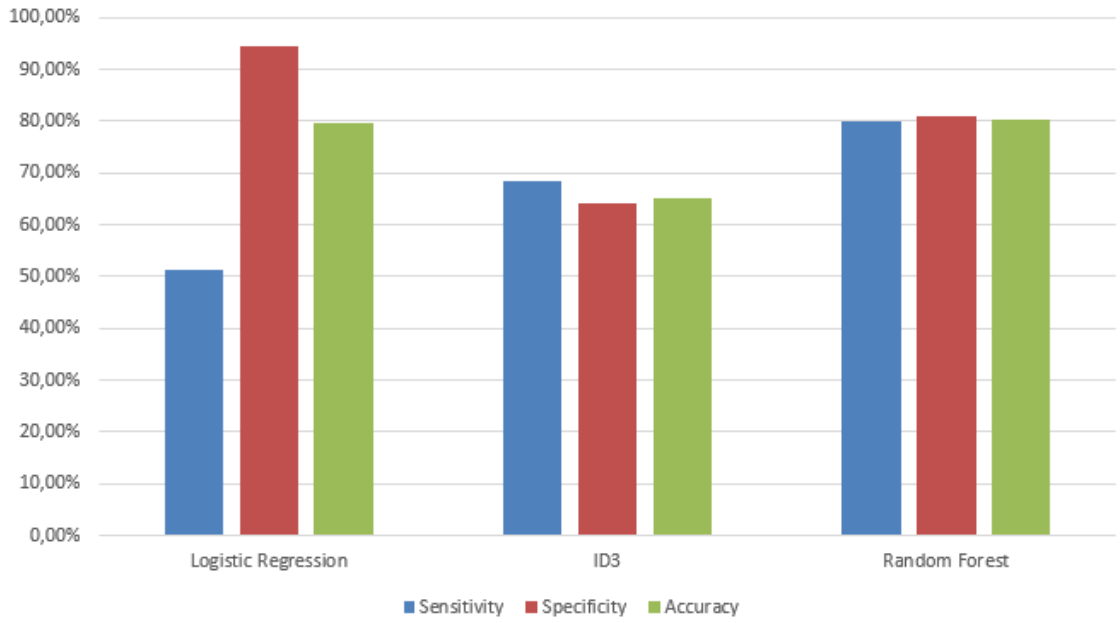


Figure 17: Results comparison of the classification methods

In the table above, the regression function provides a high specificity, which means that there are almost no False Positives. Moreover, the accuracy is around 80%, which is a good value. However, the sensitivity hardly exceeds 50% and it is a problem since the model is not useful for the doctors with this low sensitivity.

While Logistic Regression does not get a balanced percentage for specificity and sensitivity, ID3 achieves a similar value for them, which is an improvement. However, the sensitivity and specificity are too low (below 70%) and far from our objective of 80%.

In general, the main problem of the two methods is that they give a very high number of predictions for the class 0 even if the training set is balanced. Thus, the number of false negatives is too high to be acceptable because many patients with risk of developing DR are not detected.

In the Random Forest it can be seen that 80% is achieved in the three measures so we can conclude that it is the best method. Random Forest with the parameters found in section 4.3 has a good performance for the three measures and therefore it could be used by the doctors in real practice.

4.5 Summary

In this section the three studied methods have been evaluated. First, the concepts of True Positive, False Positive, True Negative and False Negative have been described, and they were used to calculate the three evaluation measures: sensitivity, specificity and accuracy.

The first method analysed has been Logistic Regression since it was the baseline. For this method the formula and its results were presented.

Then, the results of the Decision Trees modifying the leaf creation threshold have been analysed. This provided the best value for this parameter to get the best results. The final results for this method have been presented and commented.

Afterwards, the Random Forest has been analysed with an experimental setting in order to determine the best values for its three parameters. The final result is shown, and it can be seen that with it the initial objective to have 80% for the three measures studied was accomplished.

Finally, the results of the three methods have been compared in order to show that Random Forest is the one with the best performance and fulfils our objective.

5 Clinical application

Now that all the methods have been tested and compared, an application to be used for the doctors can be build. This application uses the Random Forest method with the parameters resulting from the experimental setting, since it is the one with the best results. An 80% in both sensitivity and specificity in a medical application is appropriate according to the doctors.

The medical application was designed to be integrated in the sanitary system. The application has been named as RETIPROGRAM and its logo is the following.

RETIPROGRAM

Figure 18: RETIPROGRAM logo

5.1 How will the program be integrated in the medical protocol?

The objective of the final application is to assist the doctors when deciding if a patient is prone to have diabetic retinopathy or not. Due to the fact that the tests are so expensive and there are many diabetic people it is impossible to test all the people. The program aims to identify those with a higher risk to suffer the illness and invest those expensive tests with them. According to the risk that the program detects, the next medical appointment will be arranged sooner (if the patient has a high risk) or later (if the risk is lower). Along with the doctors, a new protocol has been created to detect the illness where our program will be integrated. In the following figure it can be seen the steps that will be taken by the new system.

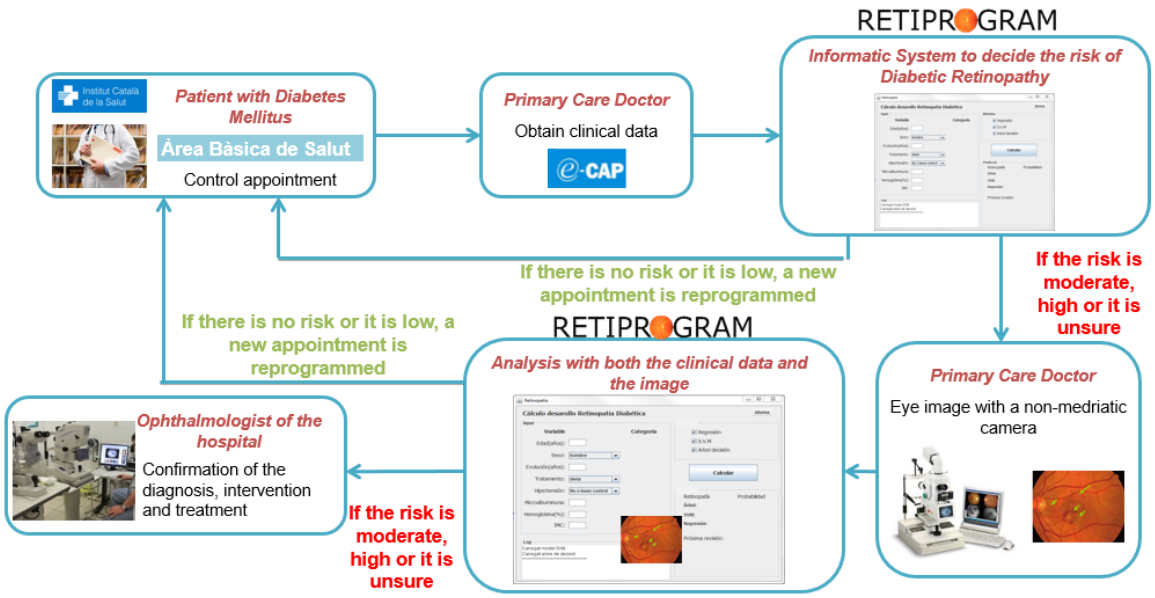


Figure 19: RETIPROGRAM in the medical protocol for DR detection

In the figure, the first step is to detect the Diabetes Mellitus in a patient. When this illness is detected, we go through the next step, where the doctor does the necessary analysis to get all the clinical data from the patient.

Once all the data is collected, the doctor can introduce the data in the RETIPROGRAM. The program can now predict the risk of developing Diabetic Retinopathy. If the program predicts that there is no risk or it is low, a new appointment is programmed. The date of the new appointment will vary according to the percentage of risk.

Otherwise, if when the doctor introduces the data in the RETIPROGRAM the risk is moderate, high or unknown then the patient will be sent to do the expensive tests (an image of the eye with a non-medriatic camera) mentioned earlier.

Once the test has been done and the eye image has been obtained, the doctor will compare the result of the RETIPROGRAM with the image obtained. If the doctor denies a risk or it is low, again a new appointment is programed. On the other hand, if the risk is still moderate, high or unknown, the patient is sent to the specialist, the ophthalmologist, who has to confirm the diagnosis and execute the required treatment. The ophthalmologist visit is also an expensive factor in the healthcare system, for this reason the program also aims to reduce the number of patients who make unnecessary visits and identify those patients who really need them.

5.2 Design

The program must meet the following requirements. The first one is about the main functioning of the application and the other three are about the structure of the graphical interface:

- The program must calculate the model when it is opened. The program has the data of the training patients, reads it and executes the Random Forest to create the model. As it is a trial application it is interesting to be able to change some parameters or information during the trial period and for this reason the program will calculate the model every time. However, the future work is to store the definitive model in the program so it does not have to be calculated each time.
- Regarding the graphical interface, the application must have a place for the doctor to introduce the data. The blanks to introduce the data have to be clearly organized so that the doctor can introduce the data quickly, easily and intuitively.
- The interface should be in three languages: Catalan, Spanish and English. Now the application will be locally tested but it was also translated into English for future use.
- The interface must include a result area, where the decision that the model takes is shown to the doctor. In this area it is shown the prediction, the probability (numerically and graphically) and the next appointment programmed. The probability is something important in the program, so it would be useful to show it in some graphical way for instance a bar that is more or less filled according to the probability.

The following diagram has been designed for the application so that it meets the requirements regarding to its functionality:

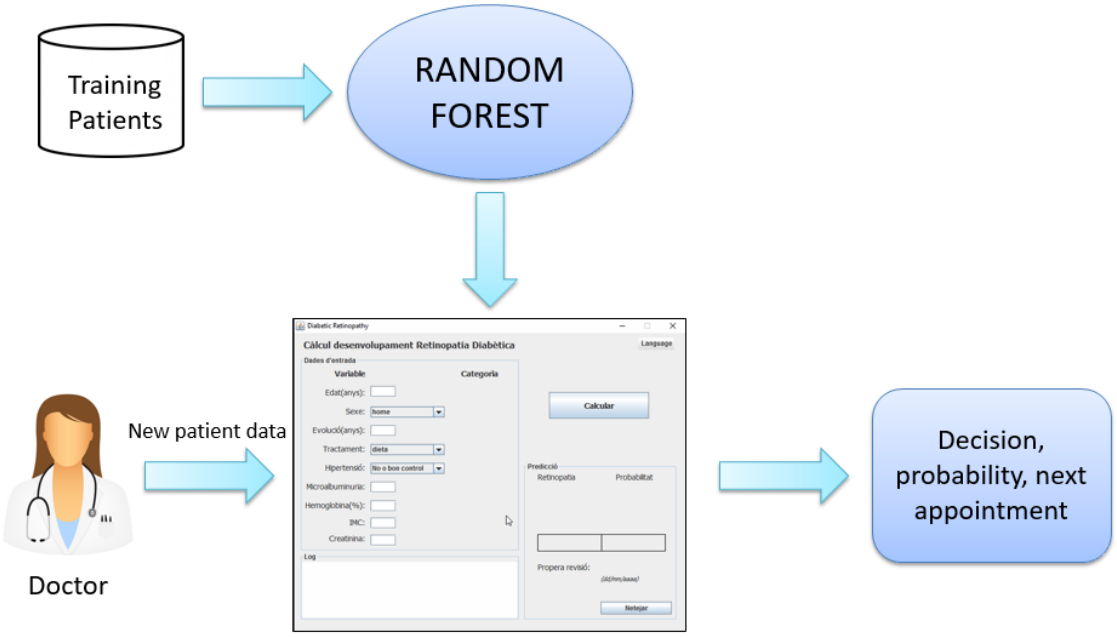


Figure 20: RETIPROGRAM design

As it can be seen, the program reads the data from the training patients and calculates the model based on the Random Forest. When the doctor introduces the data of the new patient, the program calculates the model for this specific patient and makes a prediction giving also the probability and the next appointment date.

To meet the requirements regarding the graphical interface, the following GUI was designed.

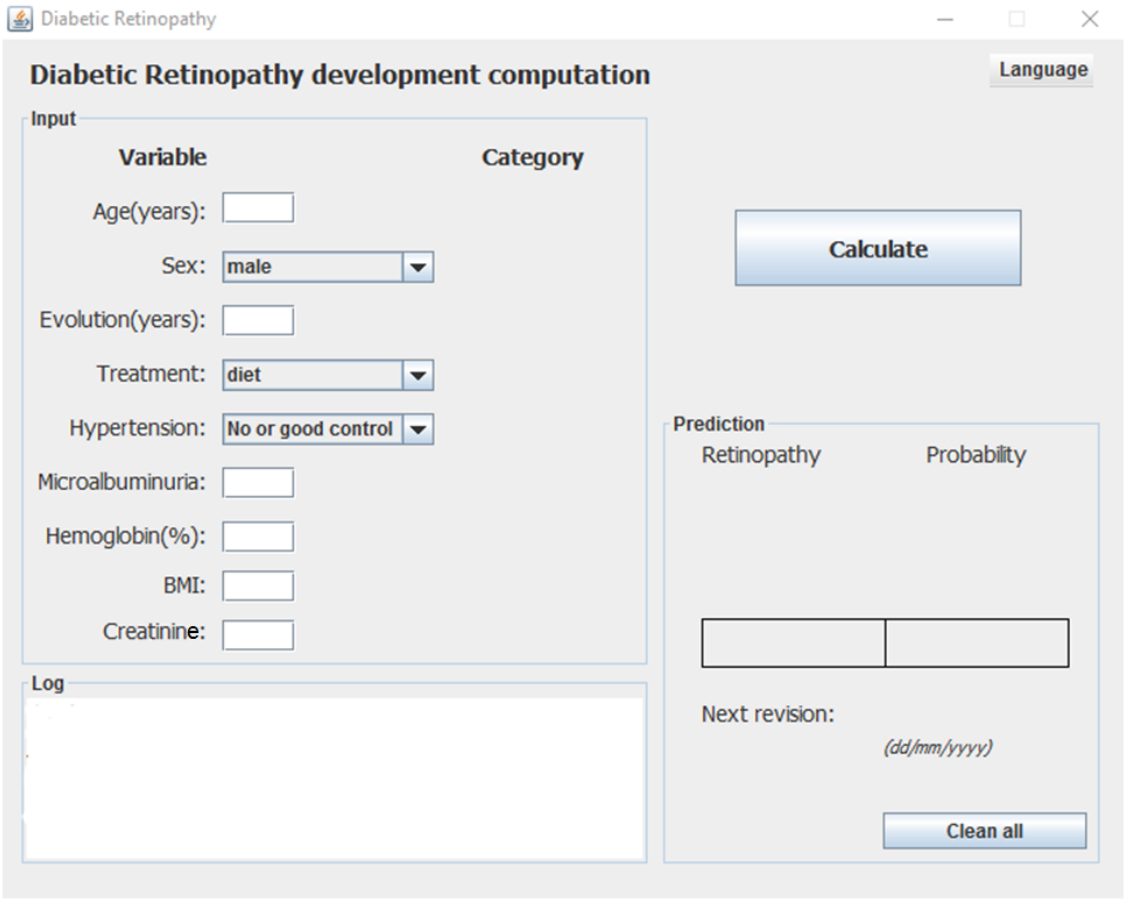


Figure 21: RETIPROGRAM GUI

The GUI has been divided into different sections in order to make it more organized and understandable. The section "input" contains all the blanks that the doctor has to fill to introduce the patient data. Whenever it is possible, a dropdown menu is used in order to prevent the human mistakes as much as possible. As it is a test application there is a "category" column where the program shows the categorization of the introduced data. In the final version of the application this column must be transparent for the user.

The "Calculate" button will allow the doctor to calculate the prediction once all the data are correctly introduced. The "clean all" button will delete all the input values, prediction and log.

In the log section, the program shows some messages, for instance when the model is charged, when some data are missing or when a prediction is done. Note that an error detection has been implemented to control if some data are incorrectly introduced. Thus, if the user introduces wrong the data (incorrect format, data missing, etc.) the application sends a message to the log indicating which data is incorrect.

In the prediction section, the decision is displayed below "Retinopathy" with a yes/no. Moreover it also displays the probability below the tag "Probability" and updates the date of the next visit. Moreover to meet the requirement of having a graphical way to display the probability it has been designed a bar that is filled according to the percentage in red (left side) if the patient has a positive prediction or in green (right side) if the patient has a negative prediction.

Here is an example of a positive patient (left figure) and negative patient (right figure) and how the bar is filled in each case.

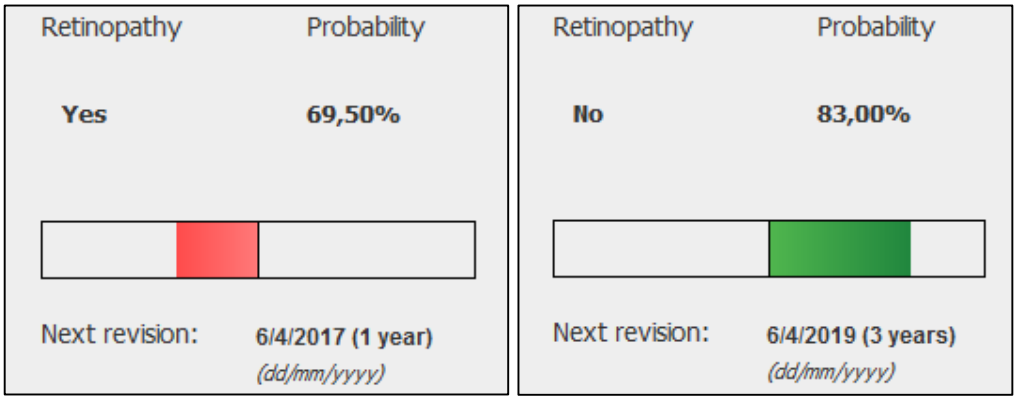


Figure 22: Probability bar in the GUI

For the unknown cases, the prediction section is left empty and a message to the log section is sent explaining that the application is not able to predict this patient.

5.3 Implementation

In order to create the application, *Java Swing* was used. Swing is a GUI widget toolkit for Java. It is part of Oracle's Java Foundation Classes (JFC) – an API for providing a graphical user interface (GUI) for Java programs. Java Swing was the best option since all the methods were programmed in Java. Therefore, an interface can be implemented using all the functions previously programmed.

Moreover, the *WindowBuilder* plugin for Eclipse was used. This tool allows designing the layout of the interface in a graphical way (position of the buttons, text areas, etc). The Swing code is automatically generated.

Then, the functionality of the buttons, text areas, menus etc, has been programmed so that they use the methods programmed in the methodology section. These are the functionalities applied to the GUI and how they were implemented:

- **Calculate Random Forest Model:** Once the application is launched, the method *RandomForest*[14] is automatically called with the parameters resulting of the experimental setting in section 4.3. Once the model is charged a message is sent to the log text area informing the user that it has been correctly loaded.
- **Language:** the radio button language has an action listener for each language. When one language is selected, the listener is actioned and its function is to set all the text fields in the language selected. A message will be sent to the log text area informing the user that the language has been successfully changed.
- **Calculate button:** The calculate button has an action listener that executes sequentially the following actions.
    - Verify if all the data introduced by the doctor in the input section is correct. The verification will include number format and if something is missing. In case that something is wrong, a message would be sent to the

[14] See algorithm 3.

log area specifying the mistake, and none of the following steps will be done.
    - In case the verification is correct the categorical values for the introduced data will be shown in the categories column.
    - The *RandomForestValidation* function will be called with the data of the new patient and the root nodes resulting of the model calculation. This function will give the result (if the patient is ill or not) and the probability.
    - The prediction area will be filled with the information of the previous point. Also, a message with the result will be sent to the log area.
- **Clean all button:** this button has an action listener and its action is to delete all the values in the input section and all the information in the prediction section.

5.4 Examples

In this section some examples are displayed in order to show how the application works. Some examples of new patients will be introduced to the system and it will be seen how the program gives the prediction for each specific patient.

Example 1

Let's imagine a patient with the following data:

Example 1	
Age	36 years
Gender	Male
Evolution of the diabetes	10 years
Treatment	Insulin and oral antidiabetics
Hypertension	Bad control
Microalbuminuria	10
Hemoglobin	9%
BMI	32
Creatinine	0.88

Table 21: Data patient example 1

Once the doctor introduces the data into the RETIPROGRAM, the model calculates the result for that patient and it is shown as follows:

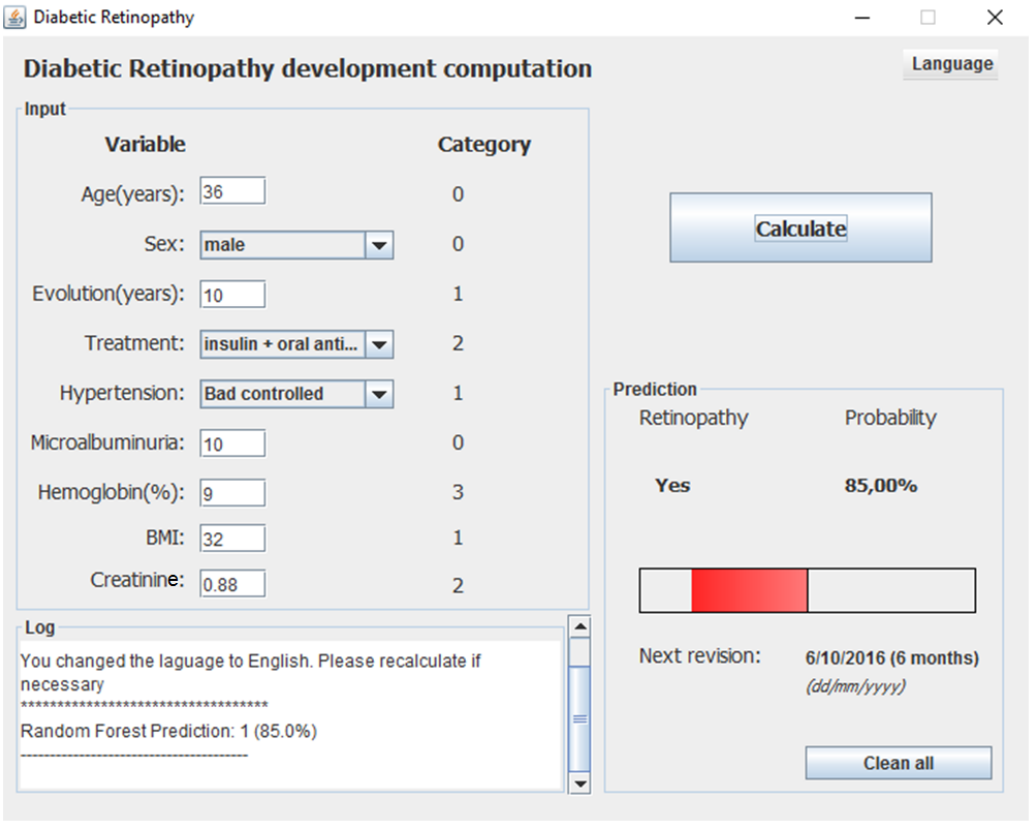


Figure 23: Prediction for example 1

As it can be seen, this patient has been predicted a risk of 85% to suffer diabetic retinopathy. This result is displayed in the "Prediction" section in the format commented before.

Example 2

Let's imagine a patient with the following data:

Example 2	
Age	83 years
Gender	Male
Evolution of the diabetes	15 years
Treatment	Oral antidiabetics
Hypertension	Bad control
Microalbuminuria	35
Hemoglobin	10%
BMI	23
Creatinine	1.4

Table 22: Data patient example 2

Once the doctor introduces the data into the RETIPROGRAM, the model calculates the result for that patient and it is shown as follows:

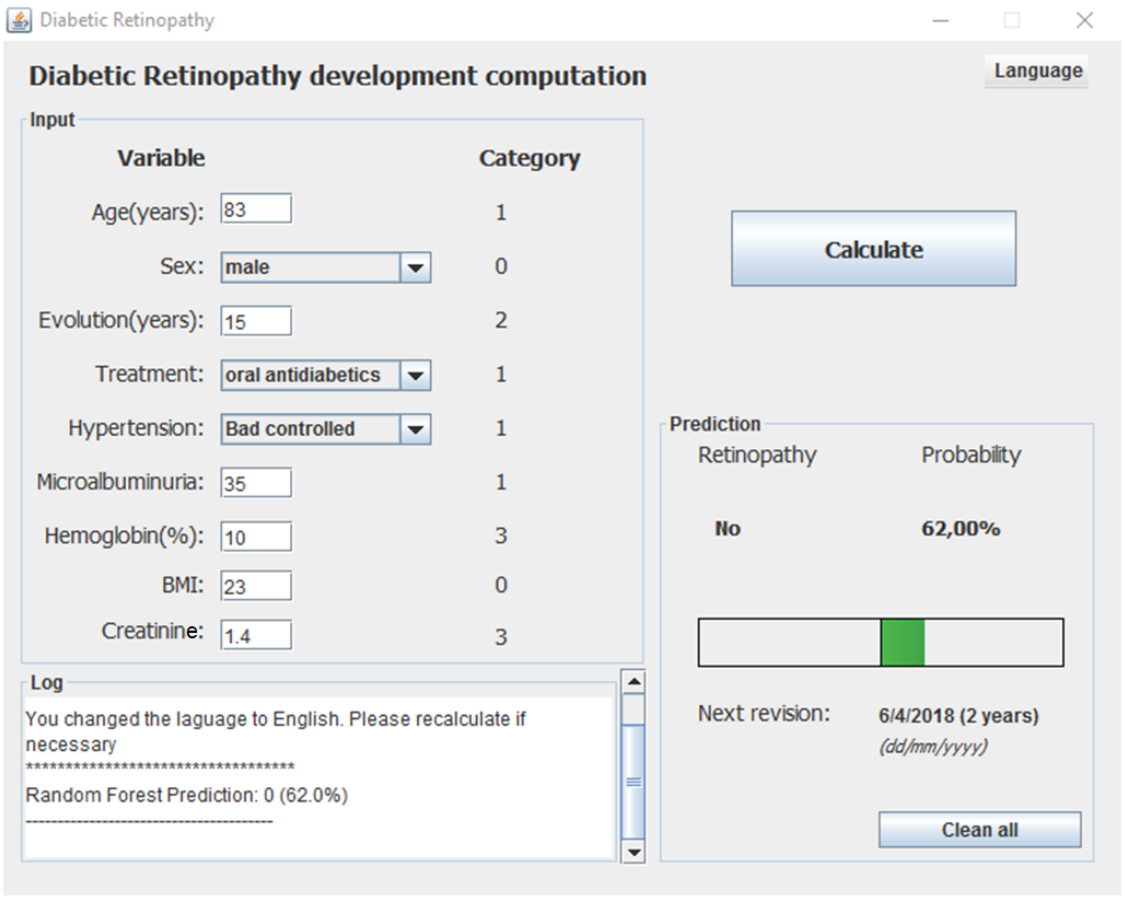


Figure 24: Prediction for example 2

As it can be seen, this patient has been predicted that he does not have a risk to suffer diabetic retinopathy with a probability of 62%. This result is displayed in the "Prediction" section in the format commented before.

Example 3

Let’s imagine a patient with the following data:

Example 3	
Age	8 years
Gender	Female
Evolution of the diabetes	1 year
Treatment	Diet
Hypertension	No
Microalbuminuria	35
Hemoglobin	6%
BMI	25
Creatinine	0.5

Table 23: Data patient example 3

Once the doctor introduces the data into the RETIPROGRAM, the model calculates the result for that patient and it is shown as follows:

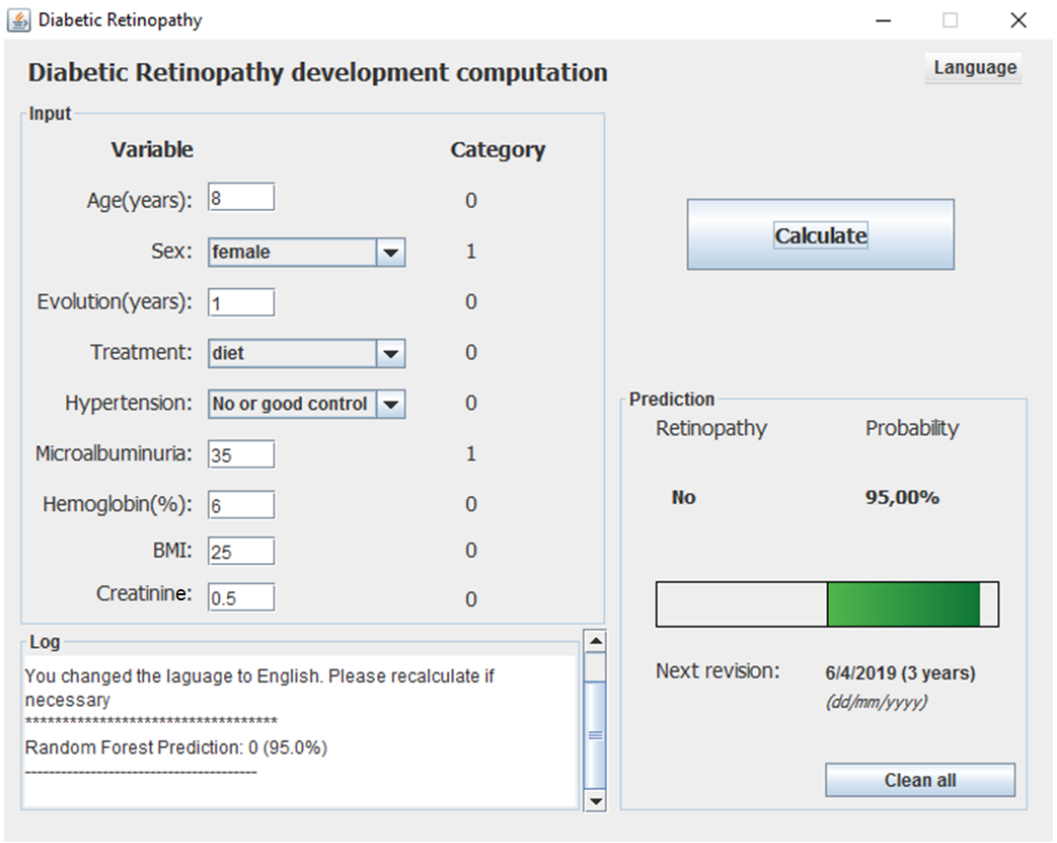


Figure 25: Prediction for example 3

As it can be seen, this patient has been predicted that she does not have a risk to suffer diabetic retinopathy with a probability of 95%. This result is displayed in the "Prediction" section in the format commented before.

5.5 Summary

In this chapter a clinical application has been build based on the Random Forest method with the best parameter values found in the previous chapter. The name of the application is RETIPROGRAM. The objective of RETIPROGRAM is to be integrated in the healthcare system and to become an assistance tool for the doctors. This tool may help to manage better the hospital resources.

The design process has been explained, including the previous requirements of the application regarding both the functionality and GUI. Moreover, the implementation has been explained, describing the most technical aspects of the application that were put into practice to accomplish the requirements. Finally, some examples of the final application were shown to illustrate how it works.

6 Conclusions and future work

Diabetes Mellitus is one of the more prevalent chronic diseases in the world. According to the World Health Organization, 347 million people worldwide (around 4.6% of the population) suffer from DM, and it has been predicted that it will be the 7th cause of death by 2030. Diabetic Retinopathy is one of the more widespread morbidities of Diabetes Mellitus. Its main effect is blindness, with large social and economic impact in healthcare. The early detection of DR, by means of periodic screening and control, reduces significantly the financial cost of the treatments and decreases the number of patients who develop blindness. Short periodicity is hard to achieve due to the large number of diabetic people, the lack of enough resources and the economic cost of the screening test.

This work aimed to find the best possible model based on historical data that is able to predict any new patient with 80% of sensitivity and specificity. To do so, different supervised classification methods were studied, such as decision trees and random forest. Training and testing datasets from real patient data were prepared. The final objective of the project is to integrate that model in a clinical application that may be used routinely by doctors as an assistance tool.

In order to reach the goals of the project, the Logistic Regression function, the Decision Trees and the Random Forest have been analysed. For the decision trees and Random Forest we have experimentally found the parameter values that give better results. Then, the results of the tests of the three methods have been compared. Finally, the best model, which was Random Forest with a threshold of 68%, 200 trees and 3 attributes, has been integrated in an application with a GUI that will be used in the healthcare system.

6.1 Conclusions

Considering the developed methodologies and the obtained results, it can be concluded that:

- The high number of diabetic people and the limited resources of hospitals raise the need from the health community to detect earlier Diabetic Retinopathy in a less expensive way.
- Computer Science and Artificial Intelligence can contribute to the medical field. In our case, the historical data of patients was used by computers in order to analyse how this illness can be detected.
- Out of the classification algorithms studied (Logistic Regression, Decision Trees and Random Forest), the Random Forest is the one with the best results. With that model, the objective to have an 80% in both sensitivity and specificity has been achieved. The parameter values of the Random Forest have been set experimentally, choosing the ones with the best results. The final configuration of the model was a Random Forest with a threshold of 68%, 200 trees and 3 attributes.
- A clinical application integrated in the healthcare system that is able to classify a patient will help doctors to make a better use of the hospital resources.

The main contributions of this work are:

- A comparison between Logistic Regression, Decision Trees and Random Forest methods.
- A model based on Random Forest that is able to classify as ill or healthy any new patient with an associated probability and with a sensitivity and specificity over 80%.
- A clinical application containing this model and a GUI that is easy for the doctors to use.

6.2 Future work

As further work, several research lines are proposed:

- Study the rules of the random forest trees in order to identify those which are more used to classify a patient and have a better percentage of hits. It would be interesting to see if there is a set of rules that are used frequently and show them to the doctors so they can check and contrast them.
- Create a data structure to store the calculated model so that it does not need to be computed each time.
- Modify the model so that it allows the prediction of a patient that does not have all the input values. The model should be able to give a result without the missing data, using only the data that are available for that specific patient.
- Validate the final model with a large database from another hospital. Other databases may be used to both create a stronger and more robust model and to validate it in a non-local area.
- Modify the algorithm so it can learn from the patients that are introduced to the system for classification. Therefore, the system could improve incrementally and automatically.

6.3 Publications and presentations

The results of this work have been presented in the following articles and conferences:

- **Title:** Assessment of diabetic retinopathy risk with random forests[15]
  Abstract: Diabetic retinopathy is one of the most usual morbidities associated to diabetes. Its appropriate control requires the implementation of expensive screening programs. This paper reports the use of Random Forests to build a classifier which may determine, with sensitivity and specificity levels over 80%, whether a diabetic person is likely to develop retinopathy. The use of this model in a decision support tool may help doctors to determine the best screening periodicity for each person, so that an appropriate care is provided and human, material and economic resources are more efficiently employed.
  Conference: 24th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN), Bruges (Belgium), 27 - 29 April 2016. Indexed as CORE B.
  State: PUBLISHED. I was the main author of the paper and the main designer of the poster. I attended the ESANN conference and presented the work.

[15] See Appendix A (paper) and B (poster presentation).

- **Title:** RETIPROGRAM[16]
  Conference: 14a Fira Recerca Directe 2016, Parc Científic of Barcelona.

[16] See Appendix C

References

[1] J.S. Edwards. Diabetic retinopathy screening: a systematic review of the economic evidence. Diabetic Medicine, 27 (3): 249-256, 2010.

[2] P. Romero-Aroca, R. Sagarra-Alamo, J. Basora-Gallisa, T. Basora-Gallisa, M. Baget-Bernaldiz, A. Bautista-Perez. Prospective comparison of two methods of screening for diabetic retinopathy by nonmydriatic fundus camera. Clin Ophthalmol 2010 8;4:1481-8.

[3] American Diabetes Association. Standards of medical care in diabetes. Microvascular complications and foot care. Diabetes Care 38:S58-S66, 2015.

[4] The Royal College of Ophthalmologists. Diabetic Retinopathy Guidelines, 2012. (rcophth.ac.uk)

[5] P. Romero-Aroca, S. de la Riva-Fernandez, A. Valls-Mateu, R. Segarra-Alamo, A. Moreno-Ribas, N. Soler, Changes observed in diabetic retinopathy: eight-year follow-up of a Spanish population, Br J Ophthalmol Published Online: 14th January 2016. doi:10.1136/bjophthalmol-2015-307689

[6] [17] N. V. Chawla, L. O. Hall, K. W. Bowyer, W. P. Kegelmeyer. SMOTE: Synthetic Minority Oversampling TEchnique. Journal of Artificial Intelligence Research, 16:321- 357, 2002.

[7] M. Kubat, S. Matwin. Addressing the curse of imbalanced training sets: One sided selection. In Proceedings of the Fourteenth International Conference on Machine Learning, pages 179-186, Nashville, Tennesse, 1997. Morgan Kaufmann.

[8] S. Kotsiantis, D. Kanellopoulos, P. Pintelas. Handling imbalanced datasets: A review. GESTS International Transactions on Computer Science and Engineering, Vol.30, 2006.

[9] V. Torra. Intel·ligència Artificial. Universitat Oberta de Catalunya ISBN: 9788469342350, 2010.

[10] B. Scholkopf, A. J. Smola. Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. ISBN: 0262194759, MIT Press Cambridge, MA, USA, 2001.

[11] A. Moreno, E. Armengol, J. Béjar, L. Belanche, U. Cortés, R. Gavaldà, J.M. Gimeno, B. López, M. Martín, M. Sànchez. Aprendizaje automático. 1994 Edicions UPC, Barcelona.

[12] G.J. Myatt, W.P. Johnson, Making Sense of Data I: A practical guide to exploratory Data Analysis and Data Mining. 2014 John Wiley & Sons, Inc, Hoboken, New Jersey.

[13] Investopedia. Decision Tree. http://www.investopedia.com/terms/d/decision-tree.asp

[14] J.R. Quinlan. Discovering rules from large collections of examples: a case study. 1979, Edinburgh University Press.

[15] L. Rokach. Decision forest: twenty years of research, Information Fusion, 27: 111-125, 2016.

[16] C. Nguyen, Y. Wang, H.N. Nguyen. Random forest classifier combined with feature selection for breast cancer diagnosis and prognostic, Journal of Biomedical Science and Engineering, Vol.6 No.5(2013).

[17] L. Breiman. Random forests, Machine Learning 45: 5-32, 2001.

[18] D. Kane. 2015, Data Science – Part V – Decision Trees & Random Forests. http://www.slideshare.net/DerekKane/data-science-v-decision-tree-random-forests

[19] D. G. Altman, J. M. Bland. Diagnostic tests: Sensitivity and specificity. BMJ, 308(6943): 1552, 1994.

Appendix A. Publication in ESANN

Title: Assessment of diabetic retinopathy risk with random forests

Conference: 24th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN), Bruges (Belgium), 27 - 29 April 2016. Since its first happening in 1993, the European Symposium on Artificial Neural Networks has become the reference for researchers on fundamentals and theoretical aspects of artificial neural networks, computational intelligence, machine learning and related topics. Each year, around 100 specialists attend ESANN, in order to present their latest results and comprehensive surveys, and to discuss the future developments in this field. The ESANN 2016 conference, held in April 2016, followed this tradition, while adapting its scope to the recent developments in the field. The ESANN conferences cover artificial neural networks, machine learning, statistical information processing and computational intelligence. Mathematical foundations, algorithms and tools, and applications are covered.

Assessment of diabetic retinopathy risk with random forests

Silvia Sanromà[1], Antonio Moreno[1], Aida Valls[1], Pedro Romero[2], Sofia de la Riva[2] and Ramon Sagarra[2*]

[1]Departament d'Enginyeria Informàtica i Matemàtiques – Universitat Rovira i Virgili
Av.Països Catalans, 26. 43007-Tarragona - Spain
[2]Hospital Universitari Sant Joan – Universitat Rovira i Virgili
Av. Dr. Josep Laporte, 2. 43204-Reus - Spain

Abstract. Diabetic retinopathy is one of the most usual morbidities associated to diabetes. Its appropriate control requires the implementation of expensive screening programs. This paper reports the use of Random Forests to build a classifier which may determine, with sensitivity and specificity levels over 80%, whether a diabetic person is likely to develop retinopathy. The use of this model in a decision support tool may help doctors to determine the best screening periodicity for each person, so that an appropriate care is provided and human, material and economic resources are more efficiently employed.

1 Introduction

Diabetes Mellitus (DM) is one of the more prevalent chronic diseases in the world. According to the World Health Organization, 347 million people worldwide (around 4.6% of the population) suffer from DM, and it has been predicted that it will be the 7th cause of death by 2030. Only in 2012 it was the direct cause of 1.5 million deaths†. It is also a leading cause of complications such as blindness, amputation and kidney failure. *Diabetic retinopathy* (DR) is one of its more widespread morbidities and it has been increasing steadily in the last years. Its main effect, secondary blindness, has a large social and economic impact in healthcare. The early detection of DR, by means of periodic controls, reduces significantly the financial cost of the treatments and decreases the number of patients who develop blindness [1].

Some scientific societies recommend that diabetic patients should be screened for DR every year‡; however, in practice this periodicity is very hard to achieve, due to the large number of diabetic people, the lack of enough human and material resources in medical centres and the economic cost of the screening procedure. Thus, there is a strong interest in developing a tool that can analyze the personal and clinical data of a diabetic person and help the medical practitioner to determine his/her risk of

* This study was funded by the research projects PI12/01535 and PI15/01150 (Instituto de Salud Carlos III) and the URV grant 2014PFR-URV-B2-60.

† http://www.who.int/features/factfiles/diabetes/facts/en/

‡ For example, the American Diabetes Association [2], the American Academy of Ophthalmology and the Royal College of Ophthalmologists [3].

developing DR, so that the temporal distance between successive controls may be adjusted depending on it and human and material resources may be used more efficiently.

In the last year researchers especialised in Ophthalmology and Artificial Intelligence at University Rovira i Virgili have been working on the application of *Intelligent Data Analysis* techniques to data from diabetic patients in order to develop a model that may predict whether a certain person is likely to suffer DR. Several classification techniques have been analyzed, including *k-Nearest Neighbours*, *Decision Trees* [4] and *regression functions*. This paper reports the results obtained with a classification model based on *Random Forests* (RF) [5].

The rest of the paper is organised as follows. The next section describes the data that have been analyzed and how the general RF method has been adjusted to the particularities of this problem. Section 3 presents the results of the classification model and compares it with the baseline classification mechanism used until now in the hospital, based on regression. The final section includes the main conclusions and the lines of future work.

2 Material and methods

2.1 Data from diabetic patients

A set of real patient data, including 1743 diabetic people that had not developed DR and 579 that suffered the disease, was provided by the ophthalmologists from Sant Joan Hospital (Reus). In order to test some classification methods this set was randomly divided into a training set T (871 healthy people, 341 people with DR) and a validation set S (872 healthy, 238 with DR). From now on, the class of healthy patients will be called 0 and the one of DR individuals will be called 1. Thus, the aim of the work was to develop a data analysis procedure that, after analyzing the data from set T, could build a classification model that could predict accurately whether the individuals in set S belong to class 0 or to class 1.

Each individual is described by 9 attributes including personal characteristics (e.g. age, gender) and clinical data (e.g. hypertension). The attributes were determined after the analysis of a period of 8 years on a population of 17000 diabetic patients. This study identified the attributes with stronger influence on the risk of having retinopathy [6]. The attributes were continuous or categorical. The continuous ones (e.g. age) were divided into relevant intervals according to [6], so that all attributes were finally treated as categorical. The values of these attributes were taken at the moment of the diagnosis of DR.

2.2 Classification model based on a Random Forest

Given a set of pre-classified objects defined on a set of categorical attributes, the algorithms for inducing decision trees (e.g. ID3 [4]) build a hierarchical structure that allows classifying any other object. In a decision tree each node represents an attribute, and the children of the node are labelled with the attribute values. The leaves of the tree indicate the class to which an object with the values shown in that branch belongs. These algorithms assume that all the objects in the training set that

share the same values in all the attributes belong to the same class, as they are indistinguishable. This fact presents a problem in our case, since it may be the case that, given a value associated to each of the 9 attributes, some patients in T belong to class 0 and others to class 1. More concretely, taking into account all the possible values of the 9 attributes there are 4608 combinations. The training set, containing 1212 individuals, only contains 451 of those combinations. Moreover, in 120 of them there are patients from both classes. In order to deal with this issue without losing information we keep the number of patients in each class for each combination. This number is used to assign a class (0, 1 or unknown) to each leaf of the decision tree.

While a decision tree has many advantages, such as comprehensibility and scalability, it still suffers from several drawbacks—instability, for instance. One way to realize the full potential of decision trees is to build a decision forest [7]. In the Random Forest method several decision trees are constructed and the final decision takes into account the predictions of all the trees. Here is how each tree of the random forest is obtained from a training data set T:

1. Pick up randomly N items of the training data. Some studies suggest that N should be around two thirds of the training set [5]. As we want to have a balanced set of items to build each tree, we take 340 patients from each of the two classes, for a total number of 680 items (56% of the training set).
2. At each node:
    a. *m* attributes are randomly selected from all the ones that have not been used yet in that branch. Previous works suggest that this number should be around *log*(number of attributes) [7].
    b. The entropy of each of these attributes is computed to determine the one that classifies better the training examples remaining in that branch and we create successors nodes for each of its values. The process stops (and a leaf of the tree is created) when, considering the combinations covered by the branch, the percentage of individuals of the training set from one class exceeds a given threshold. An "unknown" label is given to a leaf if there are not any more attributes to consider and none of the two classes exceeds the threshold.

3 Experimental setting

In this section it is explained how the optimal values for the parameters of the Random Forest method were determined. After that, the results of the RF classification are compared with those given by other well-known methods. In all the tests described in this section the following evaluation measures were considered:

- Sensitivity: TP/ (TP+FN)
- Specificity: TN/(TN+FP)
- Accuracy: (TP + TN)/ (TP+TN+FP+FN)

TP are True Positives (class 1, prediction 1), FP are False Positives (class 0, prediction 1), TN are True Negatives (class 0, prediction 0) and FN are False Negatives (class 1, prediction 0). Our aim was to obtain a classification method with sensitivity over 80%, as required for example by the British Diabetic Association.

3.1 **Random Forest parameters**

The standard RF technique has two basic parameters: the number of trees of the forest and the number of attributes considered in each node. Moreover, in our case it is also necessary to determine the value of the threshold that controls the creation of the leaves of the tree.

Let us start the analysis with this threshold. Tests were made with values between 60% and 95%, taking 200 trees in the forest and 2, 3 and 4 attributes in each node. All results show that 68% is the optimal value. Figure 1 shows the sensitivity, specificity and accuracy of the resulting RFs for 2 attributes. We can see that with a value of 68, the three evaluation measures are closer to 80%.With a higher value it is possible to increase specificity keeping a good accuracy, but there is a very strong decrease in sensitivity.

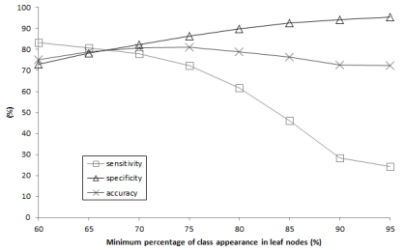


Fig. 1: Analysis of the leaf-creation threshold

On the second place we studied the influence of the number of attributes considered in each node of the tree. In the tests we tried the values from 1 to 4, with the threshold 68% and 200 trees in the RF. The results (Figure 2, left) show that 3 is the unique value for which the three evaluation measures exceed 80%.

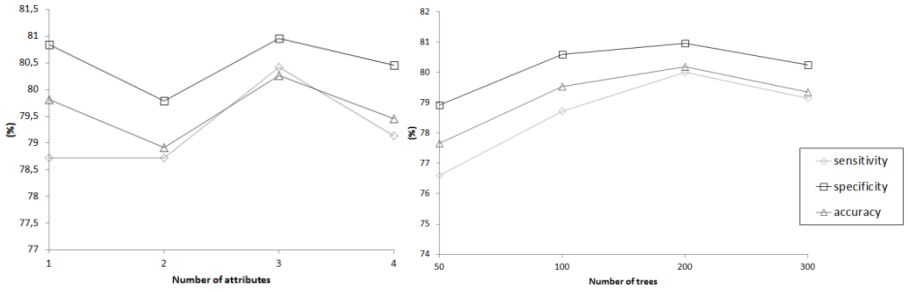


Fig. 2: Analysis of the number of attributes and the number of trees.

Finally, the influence of the number of trees in the RF was analyzed, taking the 68% threshold and 3 attributes in each node. The values considered in the study were 50, 100, 200 and 300. As seen in Figure 2 (right), the best performance of the three evaluation measures was reached when 200 trees were considered. In summary, the final RF setting considered 200 trees, 3 randomly selected attributes in each node and a minimum leaf-creation percentage of 68%.

3.2 **Classification results**

For each element of the validation set S the 200 trees are used to obtain 200 predictions of the class of the element, which may be 0, 1 or unknown (in some cases a tree may fail to classify an object because it lacks the branch with the attribute value in a given node or because there are no attributes left to explore and none of the classes has reached the required threshold). The element is assigned to the class with a higher number of predictions. If there is a tie in the number of predictions, the preference is in the following order: unknown, 0, 1.

	Predicted class				
		0	1	unk.	
Real class	0	702	165	5	Specificity: 80.96%
	1	47	188	3	Sensitivity: 80.00%

Table 1: Classification using Random Forest

Table 1 shows the classification results. It may be seen that the system is able to make a prediction in almost all of the cases (it only fails to make a prediction in 8 out of 1110 patients, 0.72%). The values of specificity and sensitivity reach 80%, whereas the global accuracy of the predictions is 80.76% (890/1102).

3.3 **Comparison with other methods**

We have compared the results of the system with three other well-known classification methodologies, given below. In the two first methods below, the dataset was previously balanced (replicating patients with RD) so that they can be fairly compared with Random Forest. However, in the last method, we used a non-majoritary prediction technique that internally manages the imbalance between the number of cases in each class.

- *Logistic regression*: this is the classification method used by the ophthalmologists of the hospital before the start of the research reported in this paper, so it can be taken as the reference baseline. A statistical package was used to calculate the regression function with a Logit model, 95% of confidence interval, 100 iterations, 0.000001 of convergence and using the Newton-Raphson algorithm for the maximization of the likelihood function.
- *Decision tree*: we built a decision tree from all the data of the training set T using the classical ID3 algorithm [4]. A leaf is introduced in the tree when the percentage of individuals belonging to a class (from all the individuals considered in that branch) exceeds 89%. This number was empirically found to be the one that leads to better classification results.
- *k-Nearest Neighbours*: for each patient of the validation set S, we look for the 5 patients in the training set T that are more similar. The similarity measure between two patients is the addition, for all the attributes, of the difference between the attribute values of the two patients divided by the number of possible values of that attribute. The best results of this method appear when the system predicts class 1 if at least one of the five neighbours belongs to class 1 (i.e. class 0 is predicted only if the five neighbours belong to class 0).

	Regression	ID3	k-NN	RF
Sensitivity	51.42%	60.08%	25.21%	80.0%
Specificity	94.49%	66.78%	77.52%	80.96%

Table 2: Comparison of the sensitivity and specificity of the classification methods

In Table 2 it may be seen that the regression function provides a high specificity (almost no False Positives), but the sensitivity hardly exceeds 50%. The k-NN method has specificity close to 80%, but sensitivity is too low (25%). ID3 achieves a similar sensitivity and specificity, but they are also too low (below 70%). In general, the main problem of the three methods is that they give a very high number of predictions for the class 0 even if the data is balanced. Thus, the number of false negatives is too high to be acceptable because many patients with risk of developing DR are not detected.

4 Conclusion and future work

Doctors need to be able to predict accurately which patients have a high risk of developing diabetic retinopathy, so that the limited human, temporal and material resources available in the screening programs are efficiently used. Thus, classification methods with a high sensitivity are required. Some standard techniques like regression functions, single decision trees or k-nearest neighbors do not have a good performance in this problem, the main reason being the inherent uncertainty in clinical data (patients with the same characteristics may appear in both classes). As seen in this paper, Random Forests provide a good classification model, obtaining sensitivity and specificity values over 80%. In our current work we are studying the relationship between the certainty in the prediction (the percentage of the majoritary class) and the cases with a classification error (False Positive or False Negative). A study of the rules that give the best performance and the key attributes is also planned. On the medium term, our aim is to introduce this classification model in a decision support tool in Primary Care to help doctors decide whether to send a patient to an ophthalmologist for a more detailed examination. Moreover, we want to extend the model to predict also the DR severity (which is classified in 4 levels).

References

[1] J.S.Edwards. Diabetic retinopathy screening: a systematic review of the economic evidence. Diabetic Medicine, 27 (3): 249-256, 2010.

[2] American Diabetes Association. Standards of medical care in diabetes. Microvascular complications and foot care. Diabetes Care 38:S58-S66, 2015.

[3] The Royal College of Ophthalmologists. Diabetic Retinopathy Guidelines, 2012. (rcophth.ac.uk)

[4] J.R.Quinlan. Induction of decision trees. Machine Learning, 1:81-106, 1986.

[5] L.Breiman. Random forests, Machine Learning 45: 5-32, 2001.

[6] Romero-Aroca P, de la Riva-Fernandez S, Valls-Mateu A, Segarra-Alamo, R., Moreno-Ribas, A., Soler, N., Changes observed in diabetic retinopathy: eight-year follow-up of a Spanish population, Br J Ophthalmol Published Online: 14th January 2016. doi:10.1136/bjophthalmol-2015-307689

[7] L.Rokach. Decision forest: twenty years of research, *Information Fusion*, 27: 111-125, 2016.

Appendix B. Poster in ESANN

Title: Assessment of diabetic retinopathy risk with random forests

Conference: 24th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN), Bruges (Belgium), 27 - 29 April 2016. Poster presentation in the “Machine learning for medical applications” section.

Assessment of diabetic retinopathy risk with random forests

Silvia Sanromà[1], Antonio Moreno[1], Aida Valls[1], Pedro Romero[2], Sofia de la Riva[2] and Ramon Sagarra[2]

[1]Departament d'Enginyeria Informàtica i Matemàtiques – Universitat Rovira i Virgili

[2]Hospital Universitari Sant Joan – Universitat Rovira i Virgili

Diabetic Retinopathy

Diabetic Retinopathy (DR) is one of the more widespread morbidities of Diabetes Mellitus, with increasing incidence. Its **main effect is blindness**, with a large social and economic impact in healthcare.

The **early detection of DR**, by means of periodic screening and control, reduces significantly the financial cost of the treatments and decreases the number of patients who develop blindness.

Short periodicity is hard to achieve due to the large number of diabetic people, the lack of enough resources and the economic cost of the screening test.

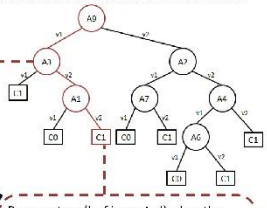
This work proposes the use of **Random Forests** to construct a decision support system to **classify the patients** according to the need of an screening, which is related to the risk of DR.

Decision Tree

We use 340 patients from each class (C0, C1) to build each tree.

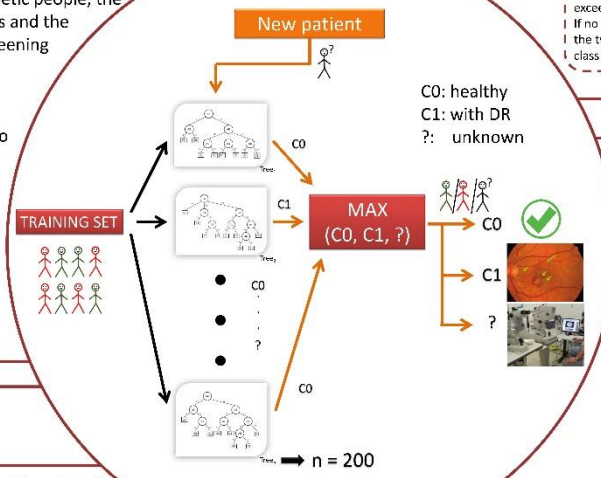
m attributes are randomly selected from the ones not used in that branch. Among the *m* attributes, we select 1 according to the best entropy

m = 3
threshold = 68%



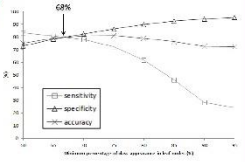
Process stops (leaf is created) when the percentage of individuals from one class exceeds a given ***threshold***.
If no more attributes are left and none of the two classes exceeds the *threshold*, the class of the leaf is labelled as unknown (?).

Random Forest

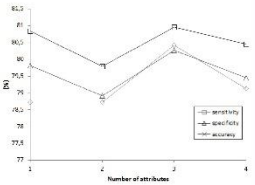


Parameters setting

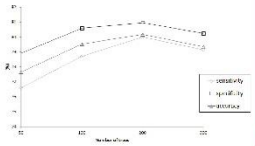
Influence of the stop **threshold**, taking 200 trees in the forest and 2 attributes.



Influence of the **number of attributes (m)** considered in each node of the tree, with the threshold 68% and 200 trees in the RF.



Influence of the **number of trees (n)** in the RF, taking the 68% threshold and 3 attributes



Comparison with other methods

Comparison table with logistic regression, single decision tree ID3 and k-nearest neighbors.

	Regression	ID3	K-NN	RF
Sensitivity	51,42%	60,08%	25,21%	80,0%
Specificity	94,49%	66,78%	77,52%	80,96%

Conclusions

- It is essential to predict accurately the risk to develop DR to make an efficient use of the resources available in screening programs.
- Regression functions, single decision trees or k-nearest neighbors do not have a good performance in this problem.
- Random Forests provide a good classification model, obtaining sensitivity and specificity values over 80%.

This study was funded by the research projects PI12/01535 and PI15/01150 (Instituto de Salud Carlos III) and the URV grant 2014PFR-URV-B2-60.

Appendix C. Poster in Fira Recerca en directe 2016

Title: RETIPROGRAM

Conference: 14a Fira Recerca Directe 2016, Barcelona Scientific Park. The "Fira Recerca Directe" is an exhibition of current research projects carried out in research institutes and Universities in Catalonia. Visitors may talk to young researchers from many different fields and solve enigmas using the scientific testing instruments with which the research projects are carried out. The attendees were explained the research developed in this work and they could interact directly with RETIPROGRAM.

RETIPROGRAM

Grup d'Oftalmologia - IISPV
Grups de recerca ITAKA-IRCV-URV

IISPV
INSTITUT D'INVESTIGACIÓ SANITÀRIA PERE VIRGILI

Retinopatia Diabètica

Què és i com afecta?

La **Retinopatia Diabètica (RD)** és una de les complicacions més habituals de la Diabetis.

Aquesta malaltia provoca danys als vasos sanguinis. El seu **principal efecte és la ceguesa**, que té un alt impacte social i econòmic.

La **Diabetis** afecta a Espanya a uns 2.393.000 pacients, segons dades del 2010.

Una **detecció primerenca de la RD**, mitjançant controls periòdics, redueix significativament el cost dels tractaments i el nombre de pacients que desenvolupen la RD.

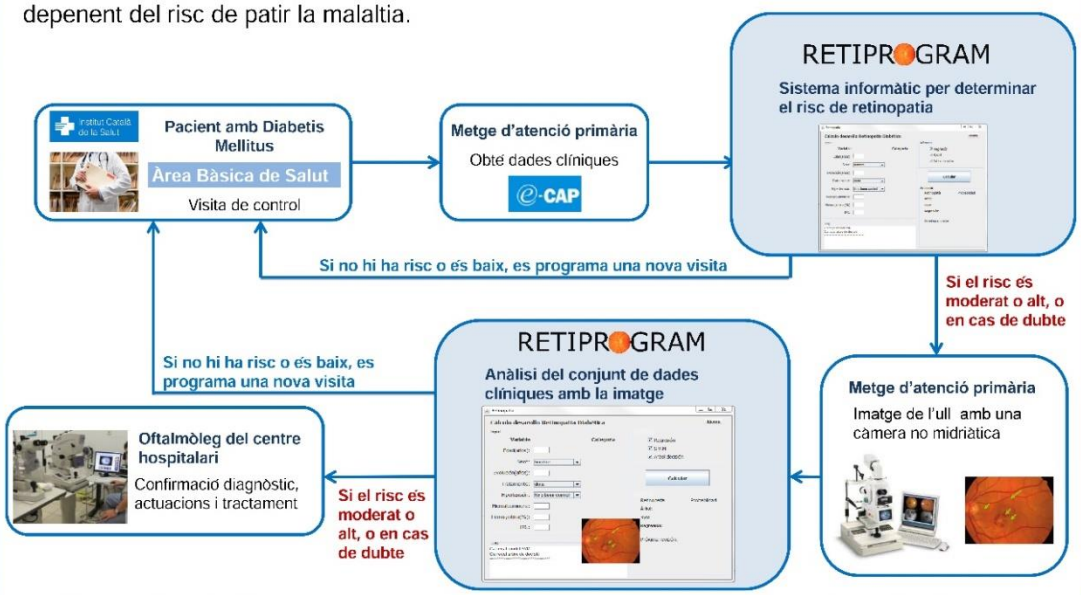
És difícil d'aconseguir fer visites anuals de seguiment a tothom, degut a l'alt nombre de diabètics, la falta de recursos i el cost econòmic dels tests de control.

Aquest treball proposa l'ús de tècniques d'**Intel·ligència Artificial** i **Visió per Computador** per construir sistemes que classifiquin els pacients segons el seu risc de desenvolupar la malaltia.

Imatges comparatives de la visió d'un pacient que té retinopatia diabètica i un pacient amb visió normal

Un sistema informàtic per ajudar a la detecció

El programa permet detectar els pacients amb més risc, que passaran a fer-se les proves, i els que no tenen tant de risc, a qui se'ls programarà una visita per d'aquí a un cert temps depenent del risc de patir la malaltia.



Permet estalviar recursos i usar-los per als pacients amb més risc

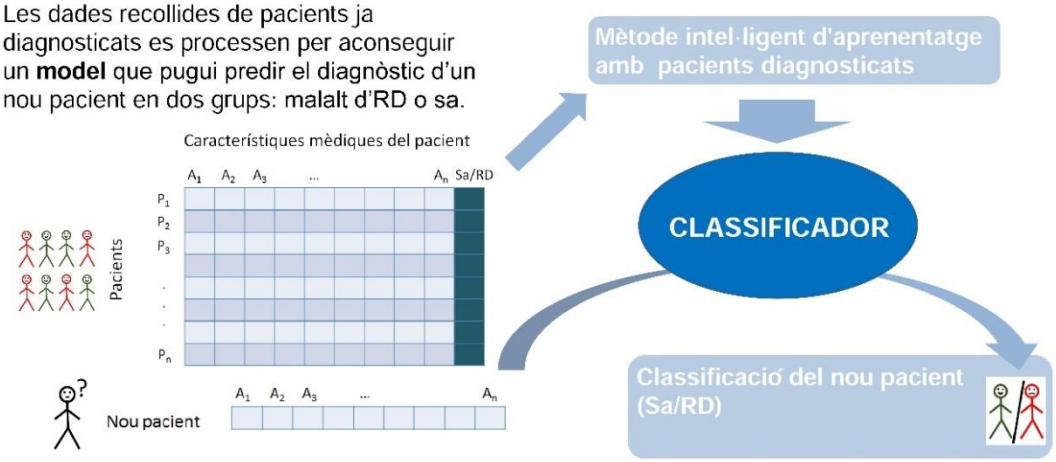
Organitza: Parc Científic de Barcelona
Sum Grup UB: Universitat de Barcelona
Amb el suport de: edu Caixa Obra Social "la Caixa"
Ajuntament de Barcelona

14a FIRA RECERCA16 EN DIRECTE
PARC CIENTÍFIC DE BARCELONA

INTEL·LIGÈNCIA ARTIFICIAL

Aconseguir un model de malalt

Les dades recollides de pacients ja diagnosticats es processen per aconseguir un **model** que pugui predir el diagnòstic d'un nou pacient en dos grups: malalt d'RD o sa.



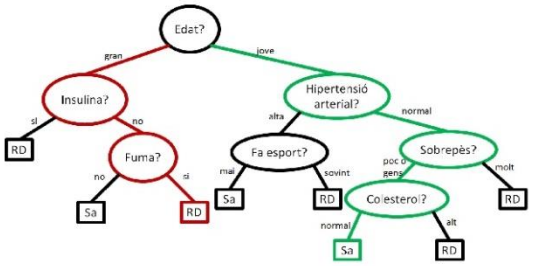
Com aconseguim el model? L'Arbre de decisió

Els arbres de decisió són un dels mètodes informàtics d'aprenentatge que es poden utilitzar per a construir el classificador.

Cada node de l'arbre és una pregunta sobre una característica del pacient i cada aresta és un possible valor.

Per a cada nou pacient s'avança en l'arbre segons les respostes a les preguntes, fins que s'arriba a un node final que el classifica. Aquests indiquen el diagnòstic.

Si seguim cada branca de l'arbre (des de dalt fins a baix) trobarem un conjunt de regles per al grup amb RD o sa.



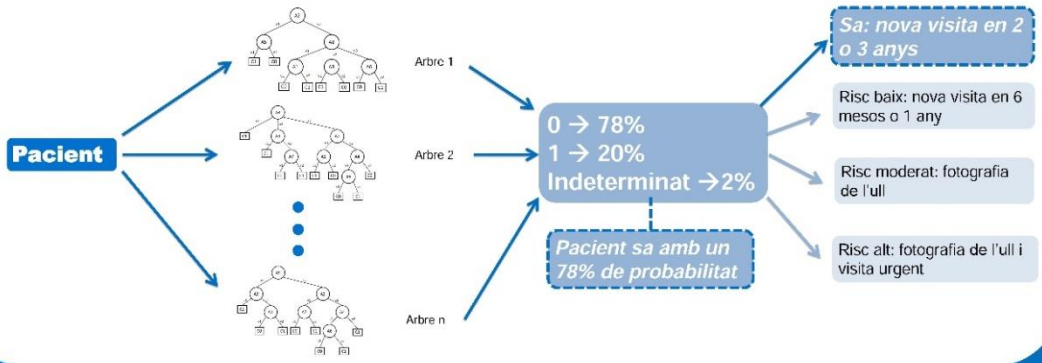
Regla: Edat = gran i Insulina = no i Fuma = si → **Pacient malalt**

Regla: Edat = jove i Hipertensió arterial = normal i Sobrepès = poc-gens i Colesterol = normal → **Pacient sa**

Classificació final dels pacients: Random Forest

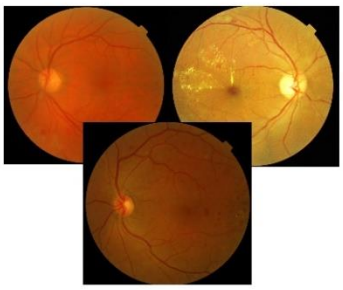
El *Random Forest* és una tècnica que genera múltiples arbres de decisió diferents amb una component aleatòria i pren la decisió basada en la predicció de la majoria d'arbres.

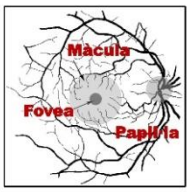
Ens permet contrastar diversos arbres en comptes de basar la decisió només en un.



14a FIRA RECERCA16 EN DIRECTE
PARC CIENTÍFIC DE BARCELONA

Processament digital d'imatges del fons d'ull





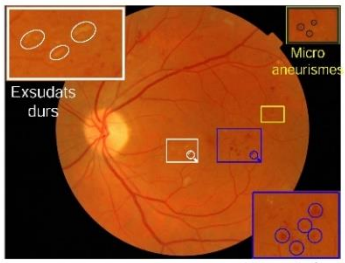

Vasos sanguinis

La detecció de lesions es fa per mitjà de programes de visió per computador

Estem estudiant tècniques d'anàlisi d'imatges per diferenciar les diverses lesions de les regions sanes.

La detecció precisa d'aquestes característiques s'enfronta als següents reptes:

Retinopatia diabètica lleu: amb microaneurismes, hemorràgies i algun exudat dur



1. La similitud de les lesions de la retina a las característiques estructurals normals de l'ull, tal com vasos sanguinis, disc òptic o màcula.
2. El color és molt homogeni i a més pot ser diferent segons la pigmentació de la persona.
3. Les imatges presenten diferents canvis d'il·luminació i contrast.

Cal desenvolupar nous algoritmes de detecció de les lesions per que les imatges del fons d'ull són molt complexes.

Retiprogram: Qui som?

Retiprogram és un projecte on estem implicats 3 grups de recerca experts en diferents àmbits:

1. Grup de recerca en Oftalmologia de l'Hospital Sant Joan de Reus (IISPV), format per metges especialistes en diagnòstic i tractament de la retinopatia diabètica.
2. Grup de recerca en tècniques avançades de gestió del coneixement (ITAKA-URV), experts en mètodes d'Intel·ligència Artificial per analitzar dades de l'historial clínic.
3. Grup de recerca en Robòtica i Visió (IRCV-URV), que desenvolupen mètodes informàtics per l'anàlisi i detecció de patrons en imatges, com les del fons de l'ull.

IISPV INSTITUT D'INVESTIGACIÓ SANITÀRIA PERE VIRGILI
Hospital Universitari Sant Joan REUS
ITAKA Intelligent Technologies for Advanced Knowledge Acquisition
IRCV INTELLIGENT ROBOTICS AND COMPUTER VISION
UNIVERSITAT ROVIRA i VIRGILI

14a FIRA RECERCA16 EN DIRECTE PARC CIENTÍFIC DE BARCELONA