CONNECARE

WP3 – SMART ADAPTIVE CASE MANAGEMENT SYSTEM

D3.4: STRATIFICATION AND MAPPING DSS

H2020-EU.3.1: Personalised Connected Care for Complex Chronic Patients

Project No. 689802

Start date of project: 01-04-2016

Duration: 45 months

Project funded by the European Commission, call H2020 – PHC - 2015	
☑ **PU**	Public
PP	Restricted to other programme participants (including the Commission Services)
RE	Restricted to a group specified by the consortium (including the Commission Services)
CO	Confidential, only for members of the consortium (including the Commission Services)

Revision: 02

Date: 30-11-2019

Document Information

Project Number	689802	**Acronym**	CONNECARE
Full title	Personalised Connected Care for Complex Chronic Patients		
Project URL	**http://www.CONNECARE.eu**		
Project officer	Birgit Morlion		

Deliverable	**Number**	3.4	**Title**	Stratification and Mapping DSS
Work Package	**Number**	3	**Title**	Smart Adaptive Case Management System

Date of delivery	**Contractual**	Month 39 (June 2019)	**Actual**	Month 44 (Nov. 2019)
Nature	Prototype ☑ Report ☐ Dissemination ☐ Other ☐			
Dissemination Level	Public ☑ Consortium ☐			

Responsible Author	Stefano Mariani	**Email**	stefano.mariani@unimore.it
Partner	UNIMORE	**Phone**	–

Abstract	This deliverable has the threefold goal of *(i)* reporting on the activities carried out to improve the 1st prototype of the risk assessment DSS presented in D3.2 “First Screening and Risk Stratification DSS”, *(ii)* reporting the same for the mapping DSS currently released in CONNECARE production environment, and *(iii)* describing the resulting software artefacts. Accordingly, section 1 motivates and gives context to the work done, section 2 recaps the main characteristics of the DSS for risk assessment and summarises the improvements done, section 3 introduces the DSS for mapping by describing its functionalities and architecture, section 4 looks forward to future iterative improvement steps, and section 5 concludes the document.

Table of Contents

Executive Summary

This document describes the research and development activity made to deploy two software artefacts: *(i)* the Decision Support System (DSS) produced to accomplish **Task 3.4 "Screening and risk stratification DSS"**, as an improvement over the first prototype presented in D3.2 "First Screening and Risk Stratification DSS", and *(ii)* the DSS for mapping produced to accomplish **Task 3.5 "Mapping DSS"**. This document thus represents a continuation of the collaboration with the team responsible for the design of the CONNECARE strategy for improving health risk assessment and patient stratification (WP2, task 2.3), whose deliverable D2.3 "Patient-based Health Risk Assessment and Stratification" is both *preparatory* and *complementary* to this one (as deliverable D3.2 itself). This work has been done in particular with the support of researchers from IDIBAPS and UMCG. The system called "Mapping DSS" (Section 3) has been developed in close collaboration with professionals from the Hospital Santa Maria in the Lleida clinical site, where it is now actively used in production, by gathering specific requirements. Finally, both systems have been developed in collaboration with teams responsible for the design of the CONNECARE SACM (deliverable D3.6, backend and frontend), so as to be easily integrated. In particular, the integration has been made together with researchers and developers from TUM, ADI, and EURECAT.

This deliverable directly contributes to the specific objectives of WP3 regarding the provisioning of "*ICT tools for the adaptive case management of personalised clinical pathways (OBJ2), which takes into account the patient's medical history*" (objective OBJ2 in DoA). Also, it contributes to make "Healthcare professionals [to be] continuously and proactively supported in their decisions through Decision Support Systems (DSSs) for: screening and risk stratification; mapping; and intervention and surveillance (suggesting personalized clinical pathways)" (OUT3-4 in DOA).

It is also worth mentioning another deliverable complementary and to be released simultaneously to this one: *D3.5 "Self-Adaptive Clinical Pathways CDSS"*, which describes the functionalities, architecture, datasets involved in the development of the "Pathways CDSS".

The table below summarizes the suggested readings and their role w.r.t. the present document.

Table 1 List of related deliverables, either preparatory or complementary.

Number	Title	Description
D2.3	Patient-based Health Risk Assessment and Stratification	Describes the consensus achieved by the clinical partners of the consortium regarding conceptual and pragmatic aspects of health risk assessment, and proposes an operational formulation of enhanced clinical risk predictive modelling to be adopted in CONNECARE.

D2.5	PDSA process and final design of the CONNEARE system	The current document provides a complete view of the Plan Do Study Act (PDSA) methodology used through-out the project, including the main objectives, methods and outcomes for each cycle and how this iterative strategy allowed to shape the CONNECARE system. Moreover, it provides a summary of the final design of the system, with a focus on the functional and non-functional requirements that fostered the development and improvement of the system and how these requirements were tackled.
D3.2	First Screening and Risk Stratification DSS	This deliverable has the twofold goal of *(i)* reporting on the development activities carried out to deliver the 1st prototype of the risk assessment DSS for screening and risk stratification, as well as of *(ii)* describing the resulting software artefact.
D3.5	Self-Adaptive Clinical Pathways CDSS	This deliverable has the goal of reporting on the activities carried out to develop a prototype of the CDSS for clinical pathways. Accordingly, Section 1 introduces the document by motivating the need for the Pathways CDSS and its goals, Section 2 reports on the requirements collection stage informing the Pathways CDSS design, Section 3 presents the requirements, Section 4 describes the design of the Pathways CDSS, Section 5 describes the implemented prototype, Section 6 discusses next steps, and Section 5 concludes the document.
D3.6	Final Smart Adaptive Case Management System	This deliverable goes with the final release of the Smart Adaptive Case Management system (SACM) by TUM and ADI, integrated to the SMS by EURECAT and the contribution of UNIMORE for the clinical decision support systems.

As a last remark, this document is an updated version of the same document submitted at the end of June 2019, extended with an evaluation of the Mapping DSS provided by the professionals using it, and details of the new prediction models developed and evaluated for the Risk DSS.

CONNECARE project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement Nº 689802 (Art. 29.4)

1. Introduction

The landscape of both clinical research and clinical practice is rapidly and substantially changing mostly due to the fundamental role played by Information and Communication Technologies (ICT) tools in *(i)* facilitating access to and exploitation of an enormous amount of *patient-related information*, and in *(ii)* providing increasingly reliable and precise computational and machine learning algorithms offering great potential for *predictive modelling* [1–11]. Indeed, in this novel scenario, the paradigm shift towards predictive and personalised medicine is triggering a whole new set of computational requirements in terms of predictive modelling for decision making.

The CONNECARE consortium is well-aware of this changing landscape, thus aims at developing novel strategies and methodologies, as well as the software tools supporting them, for enhancing *health risk assessment and stratification* beyond the state of art. Accordingly, whereas deliverable D2.3 "Patient-based Health Risk Assessment and Stratification" aimed at developing strategies and methodologies in this field, this document focuses on the technical side of the challenge, by proposing **two fully functional software artefacts for risk assessment and stratification** ("Risk DSS" henceforth, Section 2)**,** and **mapping** ("Mapping DSS" henceforth, Section 3), developed in compliance with the requirements and needs expressed by clinical partners. In particular, a team from IDIBAPS and Hospital Clinic Barcelona (HCB), and UMCG (Groningen, The Netherlands) composed by both clinicians and data scientists led the process of requirements collection for the Risk DSS, whereas the clinical professionals in IRBLL (Lleida site) led the process of requirements collection for the Mapping DSS.

1.1 Motivation

The need for advancing computational tools supporting risk assessment for screening, stratification, as well as mapping arises when recognising that, while *rule-based systems* for clinical management are accepted in the current clinical practice, exploitation of *predictive modelling tools for clinical decision support* is still far from reaching maturity. In fact, a European Union study on Big Data in Public Health, Telemedicine, and Healthcare [12] (also summarised in D2.3 "Patient-based Health Risk Assessment and Stratification", Table 3), highlighted many opportunities as well as barriers for improving current clinical practice. Among the many, the following areas of improvement are especially focussed by the two software artefacts described in this document:

- **standards and protocols**, by adopting standard formats specifically conceived for predictive models exchange (such as PMML / PFA [13]), on the hand, as well as technical software standards used to promote interoperability and portability across software platforms (as RESTful web services [14]), on the other hand
- **technological development**, by leveraging state-of-art "machine learning as a service" approaches as well as the most modern software technologies and best practices

- **data analytics**, by being open for integration of different data sources for feeding the DSS algorithms

In particular, the specific motivations leading the design choices behind the Risk DSS have been already discussed in D3.2.

The main argument motivating the need for a Mapping DSS has been given by the clinical partners in Lleida, asserting that *"having all the patients that are recorded in the system located and represented in a map…* [would mean] *…being able to carry out a global management that facilitates the surveillance"* (excerpt of a requirements collection document circulated amongst partners during co-design of the Mapping DSS). More specifically, professionals in Lleida argued that a map-based representation of the global status of clinical cases would be a valuable complement to the list-based view already present in the SACM, providing additional insights.

1.2 Goals

The specific goals pursued while designing the Risk DSS have been already described in D3.2, and briefly concern the need for having a *flexible* predictive tool enabling *separation of concerns* between the clinical staff and the team of data scientists / developers creating the desired risk assessment and prediction models.

The goal of the Mapping DSS is *"to fit the Mapping* [DSS] *in the CONNECARE* SACM *as an* ***accessory tool****. In this area, we would like to have access to a map were the dwelling situation of the patients included in the system were geographically located. Beyond this, we would like to see the* ***patients represented in relation to their clinical status*** *reported by the level of warning. The level of alert is obtained with the automated analysis of the information recorded by the devices and auto-tests the last 24 h."*

The envisioned value-added of the Mapping DSS is well conveyed by another excerpt of the requirements collection document: *"Through the mapping, doctors, nurses and social workers* ***might use its information to plan the daily route of home visits****. They could design it based on which patients are in alert situation and which are not. Beyond this, only by consulting the mapping they might have a* ***global overview of the clinical status of the patients****."*

2. The DSS for Risk Assessment

The Risk DSS described in this section is an *improved version* of the one already presented in D3.2 “First Screening and Risk Stratification DSS” as a 1st prototype. As such, the motivations behind its development within CONNECARE, the goal it pursues, the requirements it aims at satisfying, as well as its designed architecture are substantially the same as those thoroughly described in D3.2 itself, hence will not be repeated in this document. Instead, a brief recap of its functionalities is deemed necessary to make this document more self-contained and understandable by the reader.

In brief, the Risk DSS is a Decision Support System aiding clinicians in assessing the health risk of patients along specific metrics (e.g. re-admission to hospital, mortality, likelihood of acute episodes), so as to support stratification according to severity. In particular, the Risk DSS delivers health risk prediction, whose main goal is to provide *ubiquitous access* (from any web-enabled workstation) to and *seamless deployment* of (regardless of the hosting platform) prediction models, as well as to promote collaboration and interoperability between data science teams and, possibly, clinicians.

Due to limitations in data availability during the early days of CONNECARE, when clinical studies were not yet started, hence data for training prediction models were not available, the Risk DSS has been conceived to provide two **operation modes**:

- The **plugin mode** (recalled in Section 2.1), which leverages on the growing momentum around the “X-as-a-Service” paradigm, especially in the form of “Machine Learning as a Service” (MLaaS) [15], while also considering the difficulty of obtaining adequate amount of quality data, and concerns around data sharing and disclosure in a privacy sensitive domain such as healthcare. Accordingly, the plugin mode focus is on serving already trained prediction models in a platform agnostic way.
- The **learning mode** (recalled in Section 2.2), whose focus is instead on enabling the Risk DSS to train its own prediction models from scratch, based on data available within the CONNECARE system or 3rd party datasets.

The Risk DSS thus fosters and promotes *collaboration* amongst clinicians and data science teams, where the former validates and eventually adopts prediction models developed by the latter, as pictorially represented in Figure 1.

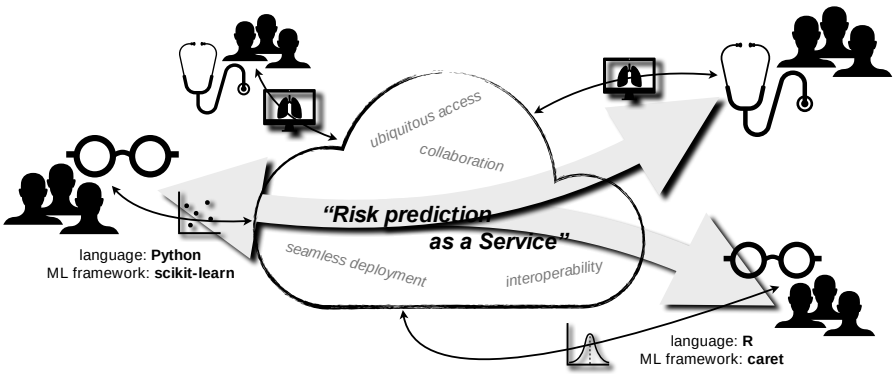


Figure 1 The goals of the Risk DSS: interoperability between ML platforms and seamless deployment.

To deliver its functionalities, the Risk DSS is designed according to the architecture depicted in Figure 2 –and already described in D3.2 with more detail, as well as in a recent publication [16] -- whose most important elements for this document are:

- The **Learning Service**, actually implementing the learning mode of operation, which is new with respect to D3.2
- The **Translation service**, which has been extended with support to the Python scikit-learn framework (see Section 2.5)

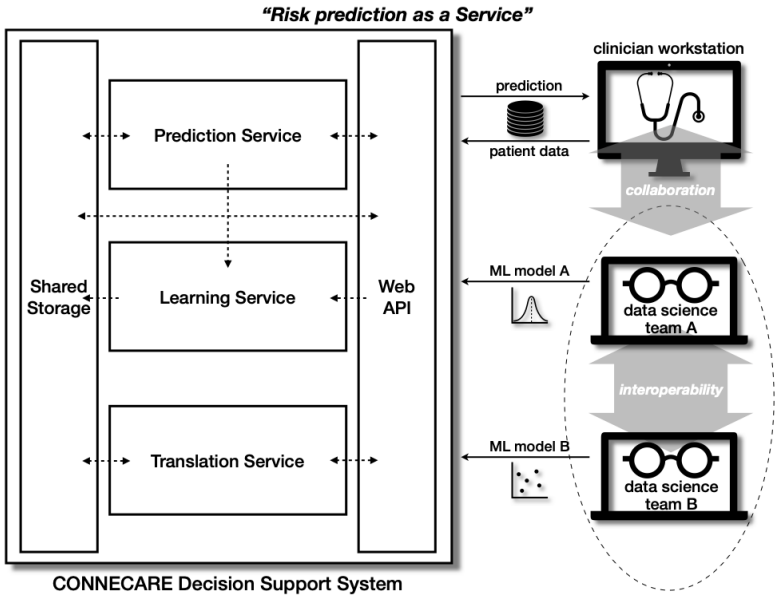


Figure 2 The Risk DSS architecture.

2.1 Plugin mode

The DSS is able to absorb *already trained* risk prediction models and apply them on new data samples (whose data schema is compatible with the one used for their training, obviously). The operations provided to either data scientist or clinicians are those already mentioned in D3.2:

- *Upload model* to "plug into" the DSS service a new risk prediction model anytime, from anywhere —through a single file upload operation, so as to be quick and easy for anyone
- *List available models* to get a listing of the currently available models with some selected information on their purpose (e.g. 30 days mortality risk), main health indicators exploited for the prediction (e.g. age, BMI), and metadata regarding their nature (e.g. random forest), so as to let clinicians decide which best suit their goals, case by case
- *Inspect model* to observe inner parameters of a model (e.g. equation of a regression line), metadata (e.g. date of last update, clinical site providing the model, reference publication if available), and requirements for application (e.g. input data schema)
- *Download model* to get any of the available models anytime, from anywhere, in either the native format they were created with (e.g. R or Python serialisation formats) or in the standard PMML / PFA formats [17] (see D3.2, Section 4.2 therein)– again, through a single file download operation
- *Delete model* to remove a model from the DSS. It is worth noting that safety measures against accidental or malicious attempts to delete models are in place (e.g. check if model has been used recently, ask consensus to model provider)
- *Apply model* to apply whichever model available to whatever data sample compatible, getting as result its predictions, anytime, from anywhere. It is worth noting that, for the sake of flexibility and data confidentiality, the data used to feed the model could be either the data actually stored in the SACM and representing a patient already enrolled in the CONNECARE program, or external data uploaded contextually with the prediction request (e.g. for testing reasons): as the Risk DSS is a separate component from the backend of the SACM, it is not inherently bound to get data from the SACM solely.

The models available and tested in a lab setting for this operation mode are two –the former provided by IDIBAPS as a "black box", the latter created by UNIMORE:

A. One is an improvement over the model already mentioned in D3.2 (Section 4.5 therein) and here reported for convenience in Figure 3. Briefly, the model is built to predict three target variables:
    a. **Hospitalisation**
    b. **Readmission** as emergency case
    c. **Mortality**

    both *during home hospitalisation* program and *at 30 days after discharge*.

B. Another is a brand new model developed on the basis of the predictive model described in [18] by clinical partners and data scientist at UMCG: there, the model crafted is meant to assist the clinician in diagnosing COPD (Chronic Obstructive Pulmonary Disease), ACOS (Asthma–COPD Overlap Syndrome) or asthma. In the model developed for the DSS, instead, the goal is to predict
    a. The amount of **exacerbations** at 1 year (either 0, 1, or 2 or more) after ACQ or CCQ assessment
    b. The **ACQ category** (Asthma Control Questionnaire) at 3 and 12 months after baseline assessment
    c. The **CCQ category** (Clinical COPD Questionnaire) at 3 and 12 months after baseline assessment

    by using the same data described in [18] and whole analysis is described in Appendix 8.1.

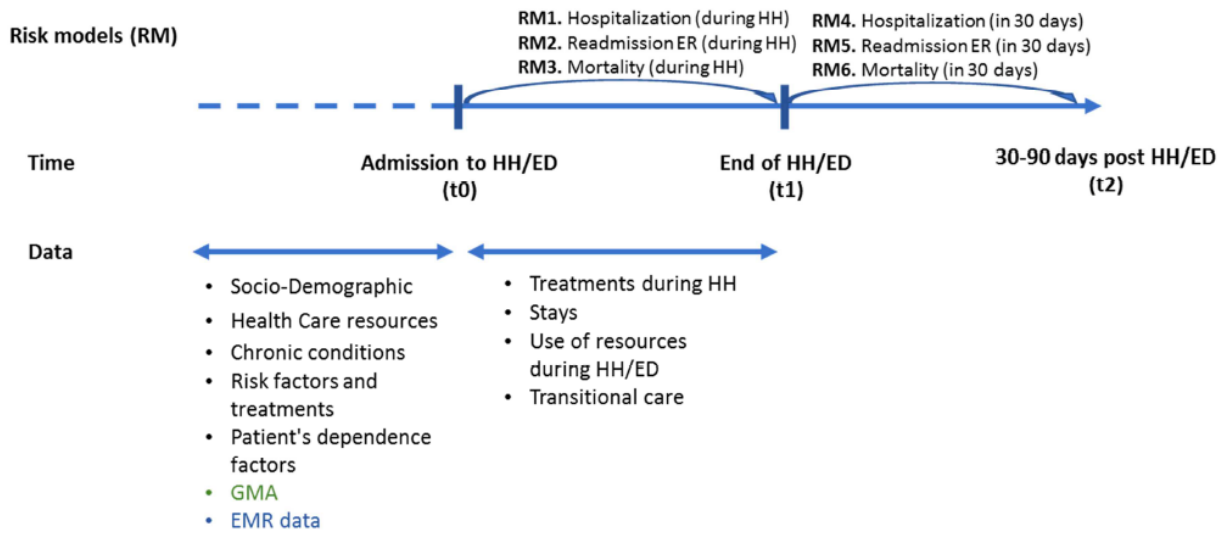


Figure 3 Risk prediction models with corresponding input data and time window for application (image taken from D2.3, Annex I).

It is worth highlighting that this operation mode improves the chance of knowledge sharing amongst clinical and data science teams, as by directly exchanging the trained predictive models ready for deployment, they do not have to disclose data –which always proven to be a big disincentive to clinical knowledge sharing in the highly privacy sensitive healthcare domain.

2.2 Learning mode

The learning mode (screenshot of web UI in Figure 4) represents the main novelty of the Risk DSS with respect to its 1st version described in D3.2, which only featured the plugin operation mode.

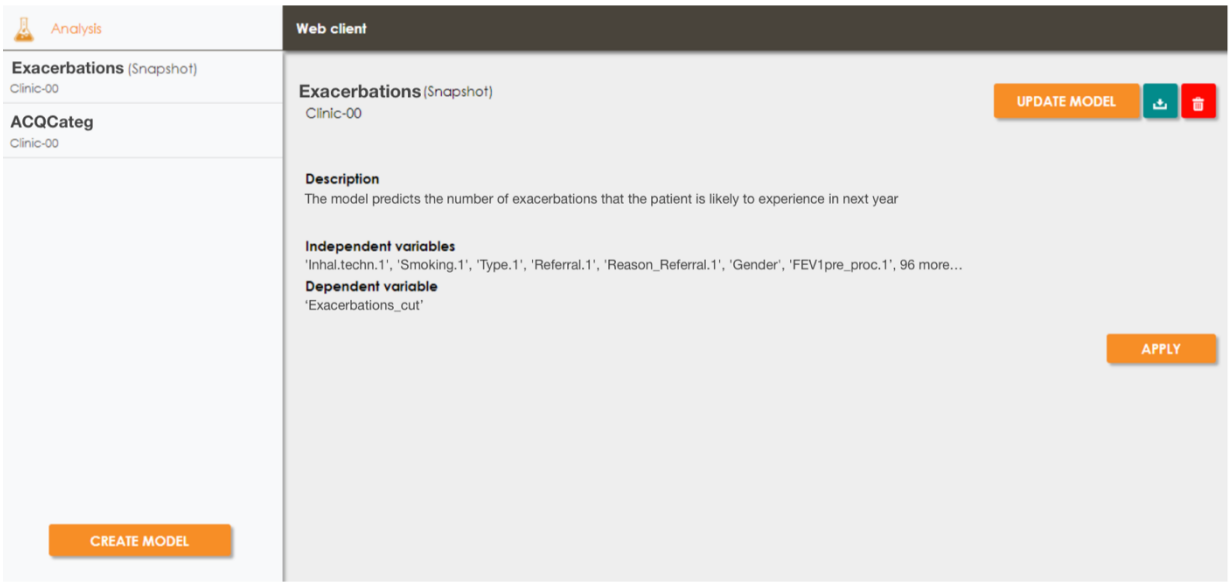


Figure 4 The demo web UI for the Risk DSS in "learning mode": on the left, the learnt model are displayed; in the central panel, some quick information about the model itself is shown; buttons trigger the functionalities described in main text.

The Risk DSS is able to create its own risk prediction models as well as to adjust / re-train them, exploiting for training and testing external datasets. The operations provided to either data scientist or clinicians are mostly in line with those envisioned in D3.2 design stage:

- *Train / test model* to (re)train or test any of the available models with new data.
- *Save model* to “freeze” a model in time and save it for later usage. This means that the model is actually replicated in two instances: one copy would not be further trained with new data streams, whereas the other one will continue online training (see “set online” configuration described below)
- *Discard model* to stop training a model and delete it anytime. This means that the training / testing data will be deleted too if coming from an external source

Two additional aspects originally (in D3.2) marked as functionalities of the learning mode have been more correctly re-interpreted as configuration options available to both operation modes (that is, plugin mode too):

- *Set local / global* to restrict or not scope and applicability of any available model (either uploaded to plugin mode, or built by learning mode). In fact:
    - in the former case, the Risk DSS allows the model to be trained only with data coming from the same clinical site provider of the model (or training it), and its application will be restricted to new data coming from the same clinical site

- in the latter case, the Risk DSS allows the model to be trained with any compatible data regardless of the provider site, and thus its application may target any compatible data as well

- *Set online* to restrict or not learning capabilities of any available model (again, either uploaded to plugin mode, or built by learning mode).
  - in the former case, the model continues to be trained with data streams subject of predictions, which become "training material" once the actual, measured values become available (e.g. whenever a COPD exacerbation is actually registered)
  - in the latter case the model is "frozen" as soon as the training process terminates – newly incoming data will be subject of predictions solely, not training material

It is worth emphasising that the version currently available of the Risk DSS does not fetch data from the SACM, and only works with external datasets supplied when the functionalities described above are requested. The reason is that the amount of data actually present in the SACM is inadequate for properly training a predictive model, and the data made available by partners (e.g. the UMCG dataset) is not completely aligned to that in the SACM, hence training on such data to later apply the model built on SACM data is unfeasible –as the dataset schemas do not match. This is also the reason why the Risk DSS has not been integrated in the SACM and put in CONNECARE production environment: usage by professionals would be meaningless without any chance to get useful predictions. Nevertheless, the Risk DSS has shown promising results as regards its predictive power (see Section 2.3), and technical integration has been already carefully designed (see Section 2.6), hence could be put to work as soon as the SACM gathers more data or new datasets aligned with the one in SACM comes in.

To deliver the learning mode functionalities, besides exploiting the same web technologies described in D3.2 for the plugin mode (such as a Java backend powered by the Jersey library for leveraging a RESTful architecture), the Risk DSS relies on Python scripts. Regardless, any model produced is seamlessly interoperable with any platform and programming language being them passed to the "plugin mode", which converts them to the PMML / PFA format. Python has been used to quickly experiment with and compare a variety of "off-the-shelf" learning algorithms provided by the well-known scikit-learn library.

2.3 Machine learning pipeline and models evaluation

The dataset as well as the exploratory analysis conducted to build the prediction models is described in Appendix 8.1. Besides exploratory data analysis and correlation testing, pre-processing included inputting of missing values (median value for numbers, and a random category for categorical variables, drawn probabilistically according to value counts of categories –so as to preserve relative percentages), One Hot encoding for categorical variables, and scaling of numerical ones to achieve normal distribution with mean 1 and standard deviation 0 (required by most learning algorithms described below). For all learning

algorithms described below, train/test splitting with a 0.33 (test) ratio has been done, k-Fold cross validation with k=10, and Grid Search for hyper-parameters tuning (e.g. C regularization factor for SVC models, n-neighbours for kNN, max depth of trees for random forest, etc.).

The Python models exploited are Linear SVC, Radial Basis Function SVC, k-Nearest Neighbours (kNN), Random Forest. For each model, prediction of number of exacerbations as a categorical variable encompassing “none”, “one”, or “two or more” categories, has been tested, as well as prediction of ACQ and CCQ category (categorical variable as per ACQ and CCQ score).

For all the models tested, the confusion matrix, where predictions are compared against true labels, is reported in the following. For the models featuring probability distributions amongst classes (e.g. KNN and Random Forest), also the precision-recall curves and the ROC curves (with AUC) of each prediction class is shown.

Exacerbations, Linear SVC. Support vector machines (SVMs) [1] are a set of *supervised learning* methods used for classification, regression, and outliers detection. The advantages of support vector machines are effectiveness in high dimensional spaces and versatility, as different *Kernel functions* can be specified for the decision function (e.g. Linear SVC is a SVM with a linear kernel). The main disadvantage is that SVMs do not directly provide *probability estimates*, but these are calculated using an expensive five-fold cross-validation (for Linear SVC they are not available at all).

Figure 5 shows the confusion matrix showing true classes (y-axis) versus predicted classes (x-axis). Briefly, the main diagonal should be the darkest (hence, with the highest values) as it represents the samples (patients) which have been correctly classified (the correct number of exacerbations has been predicted). As apparent from the matrix, this is not the case for patients which 2 or more exacerbations, which are most often predicted as having only 1. Also, it is worth noting that the model does not behave particularly well with patients with no exacerbations, as it is only slightly better than a random classifier.

While the result may seem extremely disappointing, it is worth emphasising here that classification of the number of exacerbations is extremely difficult due to the very nature of the UMCG dataset available: classes distribution across the samples is extremely unbalanced, with patients having 2 or more exacerbations being more than an order of magnitude less than patients with no exacerbations (Figure 6). Even if all the available means to account for such imbalance have been taken (by appropriately setting parameters of the Linear SVC implementation provided by scikit-learn), the only solution would be to downsample the dataset, but that would cause it to become too small to proficiently train a model able to generalise to unforeseen examples.

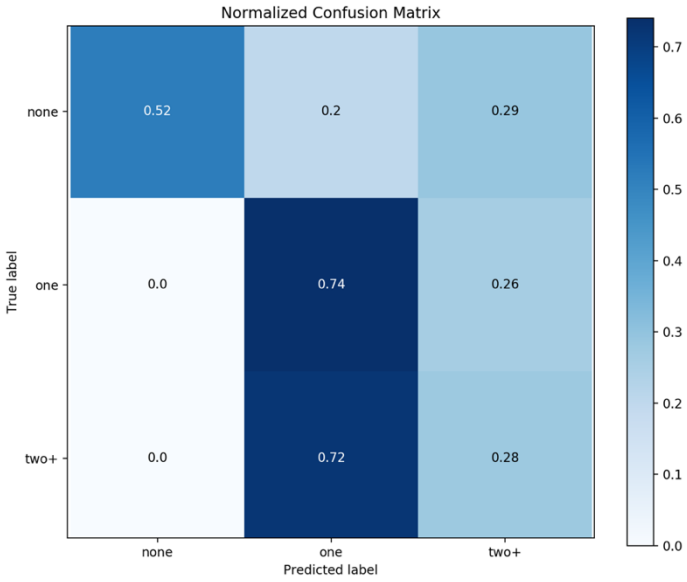


Figure 5 Confusion matrix for best Linear SVC model (C hyperparameter auto-tuned with Grid Search in a logarithmic lattice of 10 points equally spaced between e^{-6} and e^{-1}).

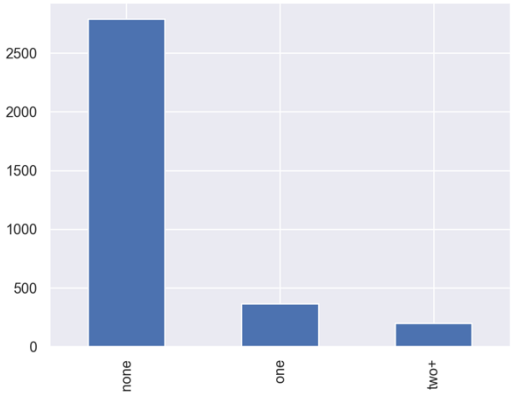


Figure 6 Class distribution for Exacerbations prediction.

Exacerbations, Radial Basis Function SVC. As an alternative to the linear kernel, a Radial Basis Function kernel has been adopted, without success: as shown by Figure 7, the trained model overfits the "no exacerbations" category ("none" in figure), actually worsening performance.

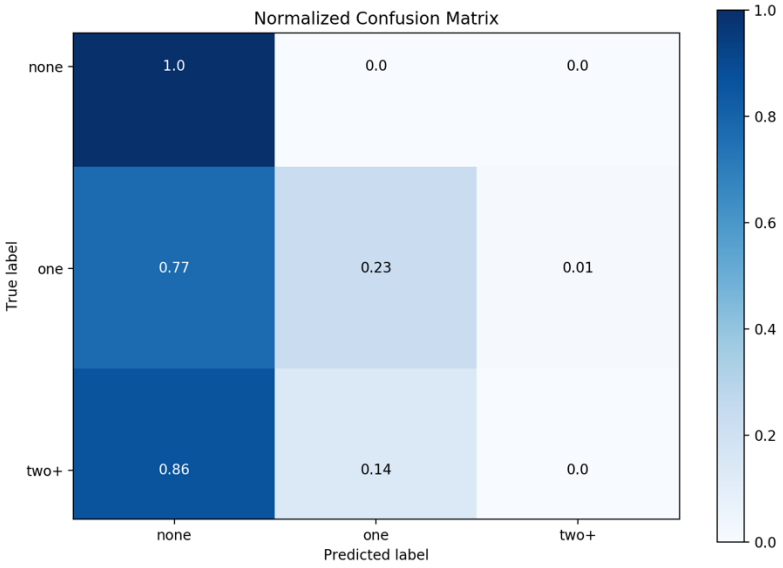


Figure 7 Confusion matrix for best SVC model using Radial Basis Function as kernel (C hyperparameter auto-tuned with Grid search as for Linear SVC, gamma hyperparameter scaled according to features).

Exacerbations, kNN. Neighbours-based classification is a type of instance-based learning or non-generalizing learning: it does not attempt to construct a general internal model, but simply stores instances of the training data. Classification is computed from a simple majority vote of the nearest neighbours of each point: a query point is assigned the data class which has the most representatives within the nearest neighbours of the point. The specific nearest neighbours classifier used is kNeighborsClassifier (kNN) [2], which implements learning based on the k nearest neighbours of each query point, where k is an integer value specified by the user (hence, which has to be tuned, e.g. though Grid Search automatic procedure). Instead of k, a radius r can be used to consider all the points within a certain radius; although such technique is documented as being more suitable for unbalanced classes distribution, we found no improvement hence this paragraph only reports about using k.

As clarified by Figure 8, the kNN model does better than the SVC with Radial Basis Function, but still misclassifies a lot of patients with 1 or 2 or more exacerbations. The Linear SVC model still seems to the better up to now. It is worth mentioning that two more graphs may help clarify "goodness" of a model: the precision-recall curve and the Receiving Operating Characteristic (ROC) curve, depicted in Figure 9 and Figure 10, respectively. The latter shows, for each class, how good the model is in classifying positive

samples, that is, samples that belong to the considered class. It may seem that kNN is quite good then, but the precision-recall graph unveils another truth: the model may be quite good in finding “true positives”, but it is quite bad in finding "true negatives” (the optimal curve is depicted in black). The reason being that ROC tell us nothing about true/false negative predictions, hence it is a partial measure of quality which can be misleading, as in this case.

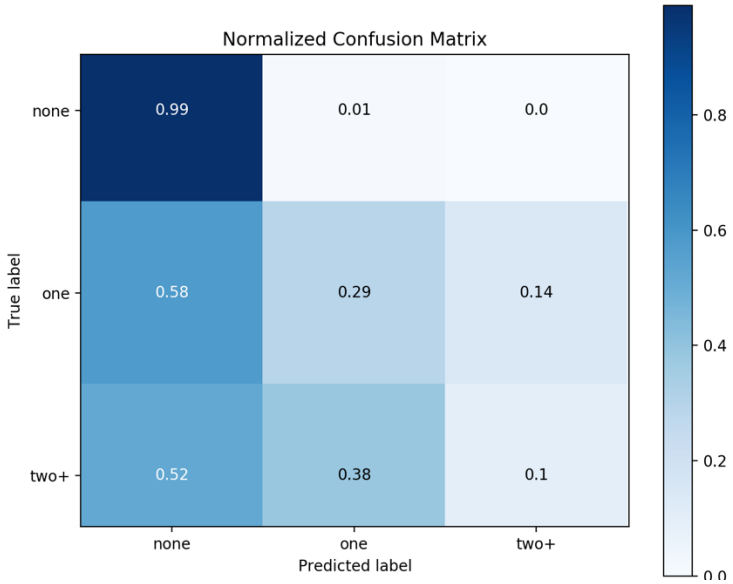


Figure 8 Confusion matrix for best kNN model (leaf size and neighbours hyperparameters auto-tuned with Grid search, to 20 and 4 respectively).

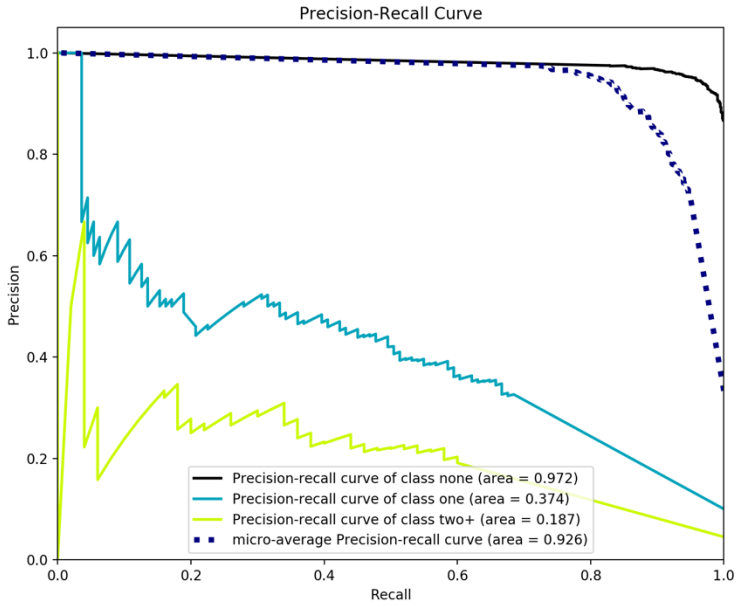


Figure 9 Precision and recall curves for best kNN model, with AUC (see legend), for each class.

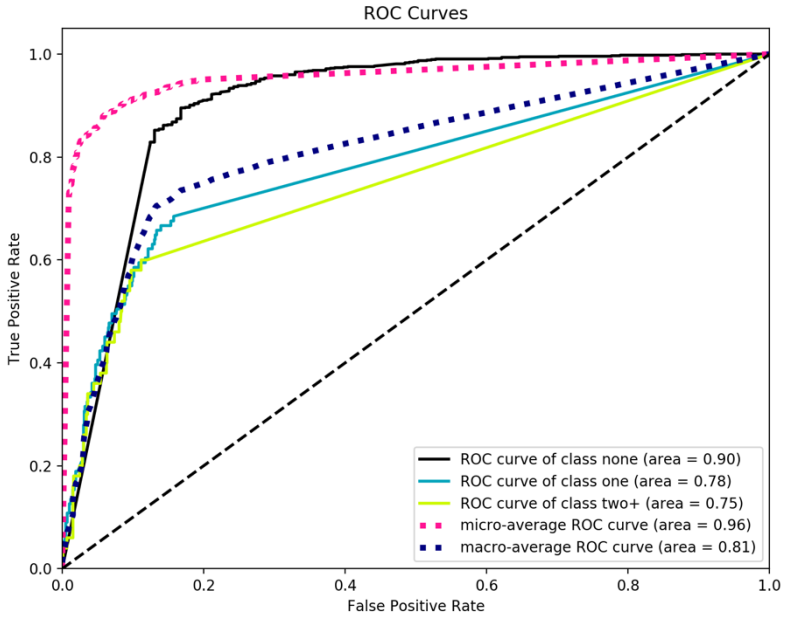


Figure 10 ROC curves for best kNN model, with AUC (see legend), for each class.

Exacerbations, Random Forest. The Random Forest [3] method belongs to the family of "ensemble methods", whose main trait is to combine the predictions of several base estimators built with a given learning algorithm in order to improve generalizability / robustness over a single estimator. Two families of ensemble methods are usually distinguished: in averaging methods, the driving principle is to build several estimators independently and then to average their predictions; by contrast, in boosting methods, base estimators are built sequentially and one tries to reduce the bias of the combined estimator. Random Forest is an averaging method, while, for instance, Gradient Boosting is a boosting method (not reported here as it always got worst results). In random forests each tree in the ensemble is built from a sample drawn with replacement (i.e., a bootstrap sample) from the training set. Furthermore, when splitting each node during the construction of a tree, the best split is found either from all input features or a random subset of size max_features. The purpose of these two sources of randomness is to decrease the variance of the forest estimator. Indeed, individual decision trees typically exhibit high variance and tend to overfit.

Figure 11 shows that applying a Random Forest with different hyperparameters set through automated Grid search does not yield improvements over the Linear SVC model: on the contrary, overfitting gets worse. Once again, we reported both precision-recall curves as well as ROC curves to reveal how misleading can the latter be in evaluating an estimator –Figure 12 and Figure 13, respectively.

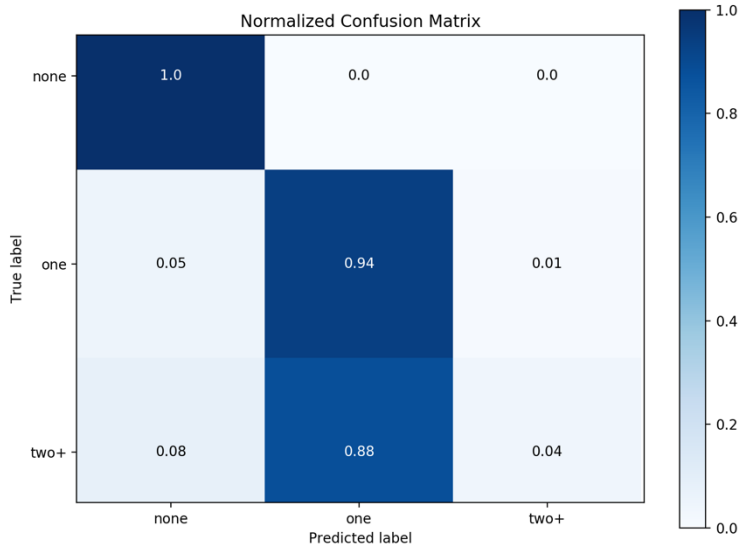


Figure 11 Confusion matrix for best Random Forest model (max depth of each tree, minimum number of samples to split, and number of trees hyperparameters auto-tuned with Grid search, to 40, 30, and 40 respectively).

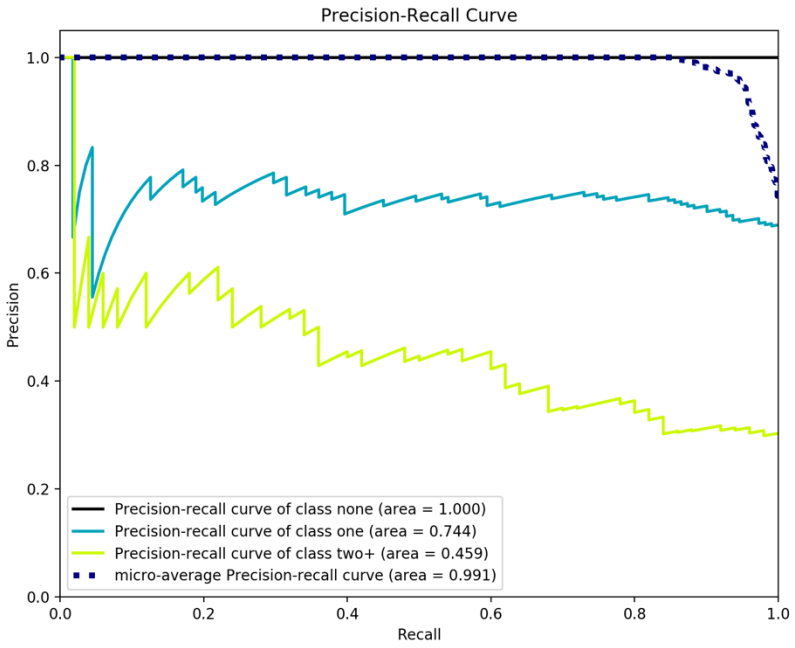


Figure 12 Precision and recall curves for best Random Forest model, with AUC (see legend), for each class.

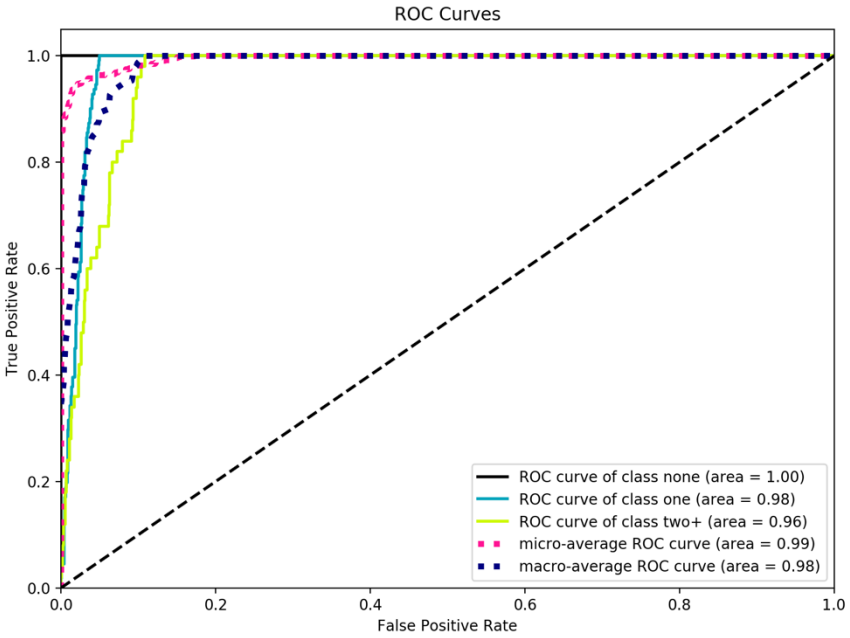


Figure 13 ROC curves for best Random Forest model, with AUC (see legend), for each class.

ACQ category, Linear SVC. Figure 14 shows the confusion matrix of a Linear SVC model trained to predict the ACQ category of patients. It is worth mentioning that amongst patient data there is the ACQ score, which has obviously been removed from the training data as it is a clear proxy for the predicted variable. The confusion matrix shows how the model behaves quite well in predicting controlled and uncontrolled patients, whereas performance in predicting partially controlled patients is poor, being half the time patients wrongly attributed to the controlled class, and half the time to the uncontrolled class. We speculate that a reason for this could be that such a category is actually the most difficult to assign also in practice, for human professionals.

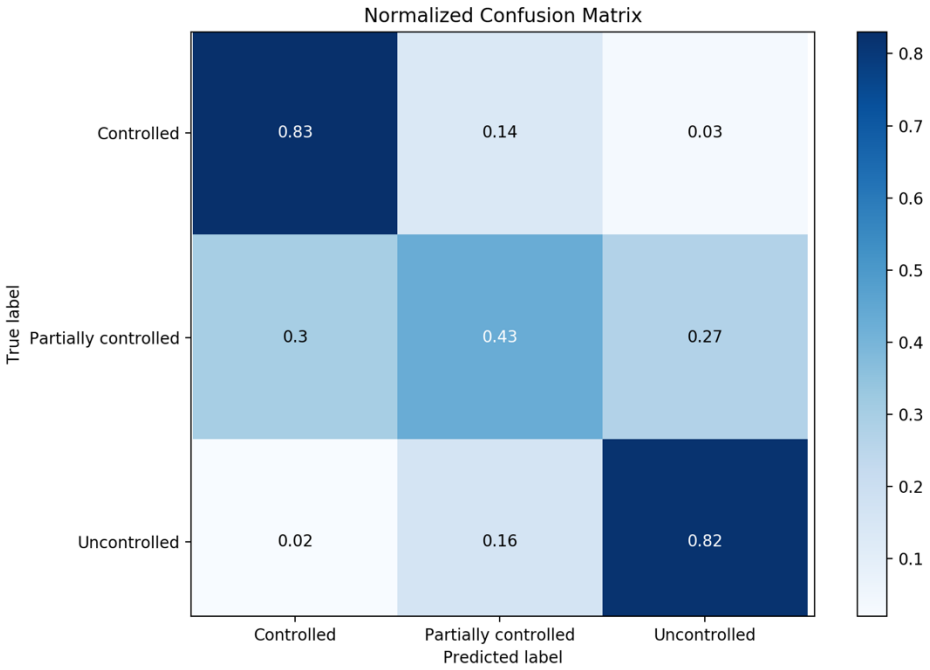


Figure 14 Confusion matrix for best Linear SVC model (C hyperparameter auto-tuned with Grid Search in a logarithmic lattice of 10 points equally spaced between e^{-6} and e^{-1}).

ACQ category, Random Forest. Figure 15 shows how a Random Forest model behaves in predicting ACQ category. Compared to the Linear SVC model, it does even better as telling apart controlled patients from uncontrolled one, but does not sensibly improve performance on partially controlled category, as confirmed by the precision-recall curves plotted in Figure 16.

Other models (kNN and SVC with radial basis function kernel) did not show different results, hence are omitted.

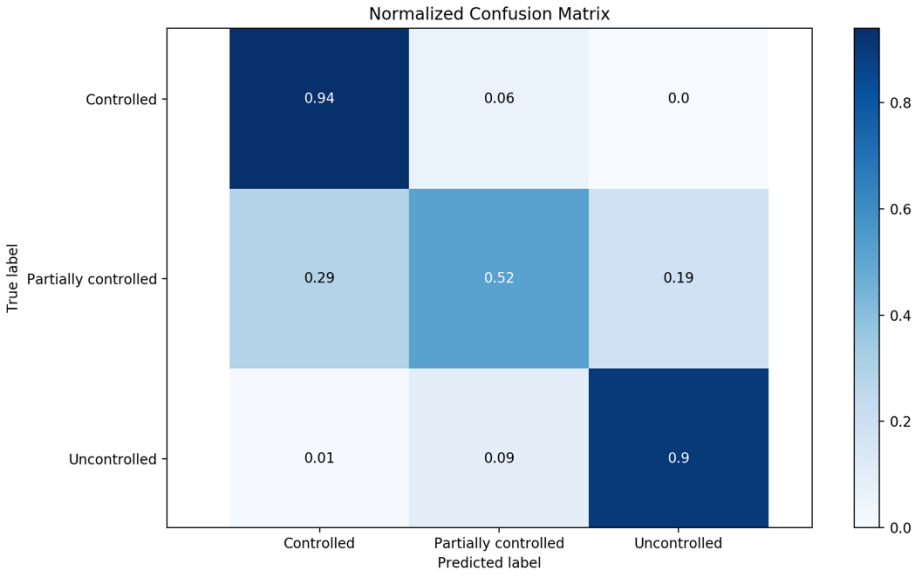


Figure 15 Confusion matrix for best Random Forest model (max depth of each tree, minimum number of samples to split, and number of trees hyperparameters auto-tuned with Grid search, to 30, 10, and 40 respectively).

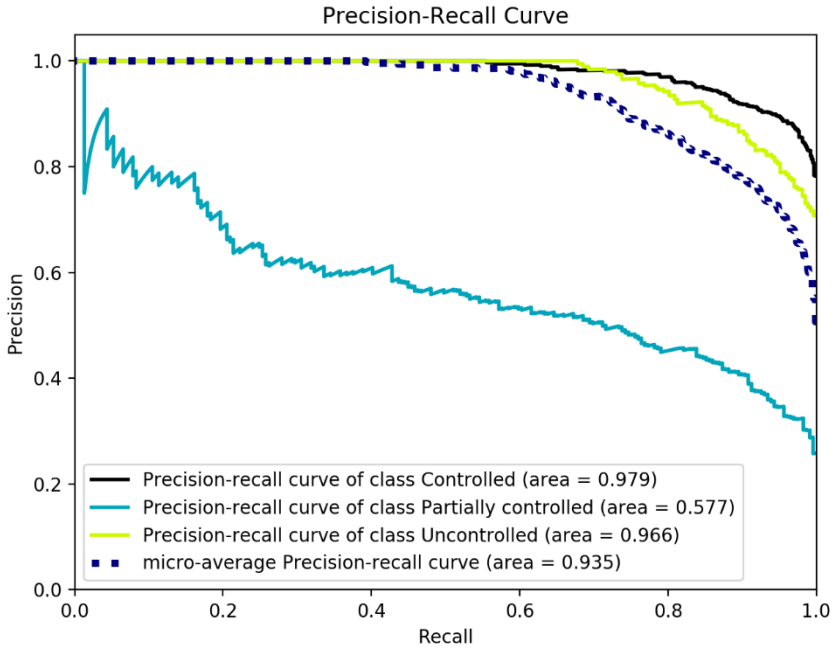


Figure 16 Precision-recall curves for the best Random Forest model.

CCQ category, Linear SVC. For predicting CCQ category we had to remove proxy independent variables as we did for the ACQ category, that is, we removed every variable related to the CCQ questionnaire (such as total score, functional score, etc.). Despite several Grid Search sessions, the model does not

behave well in any category, as the peak of performance is 75% for the “Stable” category. However, one interesting thing to note is that most of misclassification happens by attributing classes to the “Not entirely stable” class: by removing that one, it is reasonable to assume that such misclassification would be mitigated.

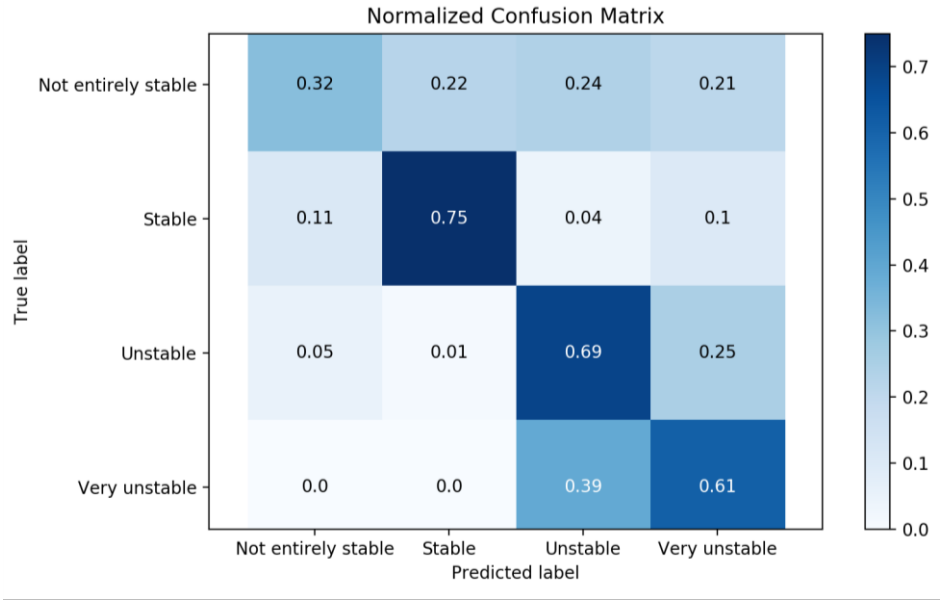


Figure 17 Confusion matrix for best Linear SVC model (C hyperparameter auto-tuned with Grid Search in a logarithmic lattice of 20 points equally spaced between e^(-6) and e^(-1)).

Furthermore, we must bear in mind that, as for any previous predicted variable, CCQ category is extremely imbalanced, as depicted in Figure 18: in particular, the “Not entirely stable category is much more represented than the “Unstable” and “Very unstable” categories, hence few examples are available for the latter two.

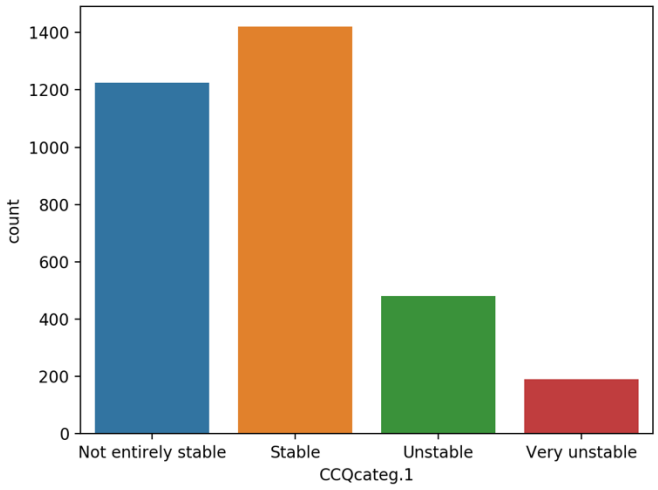


Figure 18 Class imbalance for CCQ category predicted variable.

By comparing this behaviour with the one we commented for ACQ category in previous paragraphs, we should also say that “Not entirely stable” category is difficult to predict likewise the “Partially controlled” one for ACQ, as it represents a mild condition with no particular distinguish features (independent variable).

CCQ category, Random Forest. The only other predictive model which has shown acceptable results is a Random Forest finely auto-tuned using Grid Search, as reported in the confusion matrix of Figure 19: with respect to Figure 17 performance notably increased in predicting “Not entirely stable” category at the price of “Unstable” category, which lost a bit of accuracy.

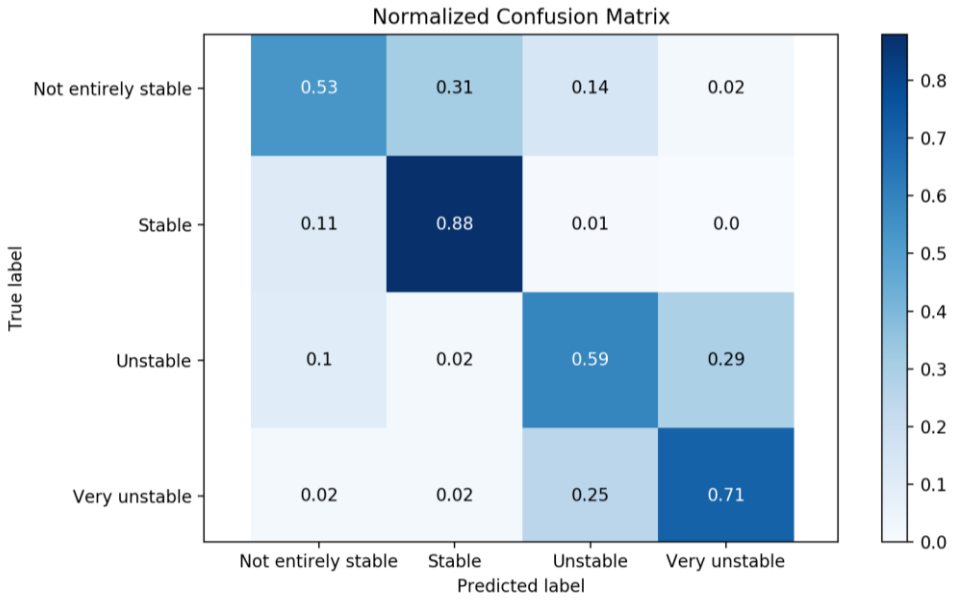


Figure 19 Confusion matrix for best Random Forest model (fully developed trees, minimum number of samples to split, and number of trees hyperparameters auto-tuned with Grid search, 10%, and 90 respectively).

Figure 20 and Figure 21 better clarify performance of the model by showing the precision-recall curves and the ROC curves (both with AUC), respectively. The apparently very good performance visible from ROC curves is partially mitigated by looking at the precision-recall curves, which emphasise false positives and negatives.

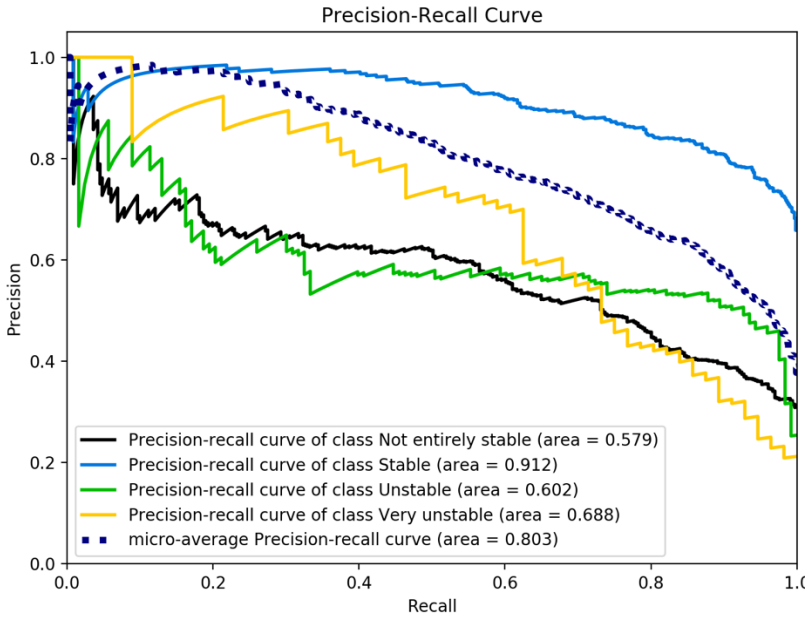


Figure 20 Precision-recall curves for best Random Forest model.

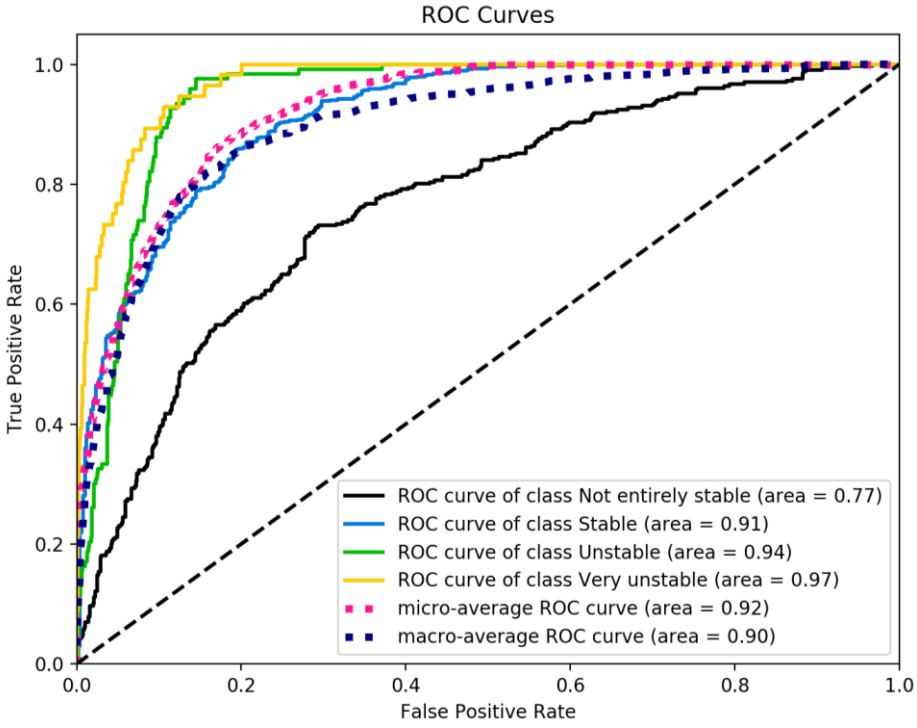


Figure 21 ROC curves for best Random Forest model.

2.4 Summary of models evaluation

Considering the results commented above, we are satisfied with the models produced and clinical partners have shown appreciation and interest in the perspectives enabled by injection of such a decision support tooling in their usual workflows. However, performance is still to be improved before such a system could be used in day-to-day practice. More data is needed to improve the quality of the models. Moreover, before putting it into the real clinical practice, as the implementation studies of CONNECARE, the models and the proposed solutions have to be tested in a controlled environment by the clinicians (e.g. a laboratory or in a clinical trial). In so doing, the corresponding feedback should be used to improve the effectiveness of the approach. Only afterwards, the Risk DSS can be made available to real implementation studies. Thus, in case of CONNECARE, integrated in the SACM.

2.5 Other improvements

Besides purely technical improvements to the Risk DSS backend or cosmetic changes to the RESTful APIs, another important improvement has been extending the reach of the plugin operation mode to support **automatic translation** of models built by using the Python language and, specifically, the **scikit-learn** framework –the 1st version of the Risk DSS only supported automatic translation back and forth R models built with the `caret` package (see D3.2, Section 2.4 therein).

Python is natively able to persist data (beyond prediction models solely) on disk by using the `pickle` library[1], hence Python built-in persistence model. In the specific case of scikit-learn objects (thus, there including prediction models, or "estimators"), furthermore, a dedicated replacement of `pickle` is available, through the `dump` and `load` modules of the `joblib` Python library[2]. Nevertheless, in scikit-learn online documentation page dedicated to model persistence (https://scikit-learn.org/stable/modules/model_persistence.html) it is advised not to use such libraries for production models, as there is no guarantee that models semantics will be preserved between scikit-learn versions. Fortunately, an alternative solution exists in the form of *open standards*, such as the already mentioned PMML / PFA model interchange formats.

Accordingly, as already envisioned in D3.2 (Section 4.2 therein), automatic conversion of Python scikit-learn models to PMML is now supported by Translation service of the Risk DSS by exploiting the JPMML-SkLearn[3] Java library. However, the situation is more complex as far as the automatic conversion to PFA is concerned: for this task, in fact, the only available software tool is a Python library, `sklearn-to-pfa`[4],

[1] https://docs.python.org/2/library/pickle.html

[2] https://joblib.readthedocs.io/en/latest/, https://joblib.readthedocs.io/en/latest/generated/joblib.dump.html, https://joblib.readthedocs.io/en/latest/generated/joblib.load.html

[3] https://github.com/jpmml/jpmml-sklearn

[4] https://pypi.org/project/sklearn-to-pfa/

whereas no Java solutions exist, hence conversion should be charged upon the producers of the model —e.g. the data science team working with Python. Nevertheless, a few possibilities to achieve automatic conversion anyways are currently under investigation:

- By wrapping the `sklearn-to-pfa` library as a RESTful service with the Django framework[5], so as to interact with it via HTTP (network latency will be almost zero as both this service and the Risk DSS will be running in the same server, resembling a microservice architecture)
- By executing the `sklearn-to-pfa` library as a Java process and interacting with it through the file system or standard input / standard output, by using tools such as Java native Process API[6]

2.6 Integration

As already pointed out, the Risk DSS system have not been integrated in the SACM: for the plugin mode, the model there available are not suitable for the SACM data schemas, whereas for the learning mode, the SACM has too few data samples to proficiently train good predictive models ready to be put in CONNECARE production environment and, thus, used in the implementation studies.

Even if the system has not been considered mature enough to be available during the implementation studies (as summarized in Section 2.4), its integration has been already planned for the first version of the Risk DSS (deliverable D3.2). With the current version of the DSS, it could be applied with minor modifications and ready for further investigation or adoption.

The Risk DSS is a web service that exposes suitable HTTP endpoints for any 3rd party system willing to exploit its functionalities, in the form of RESTful resources. This holds true for functionalities regarding both the plugin and the learning mode. Hence, it is possible to ask the Risk DSS to apply a model on a given data sample (patient) programmatically, as well as ask the Risk DSS to start training a novel model on a given dataset. This means that, as we described in deliverable D3.2, the SACM backend could ask the Risk DSS to make predictions about exacerbations for a patient who just completed the evaluation stage, in case the prediction model is readily available (that is, for the plugin mode).

Integrating the learning mode functionalities is technically the same, but is more difficult from the organisational perspective: for the Risk DSS fetching data and start training prediction models can happen at the click of a button on the SACM, but careful decisions should be made about who is entitled to ask for training new models and, most importantly, who is responsible for evaluating the models to decide to make them available in the SACM as predictions. The latter step necessarily involves the technical aspects of evaluation we commented for our models described in Section 2.3, which can be difficult to grasp for a non-programmer / data scientist.

[5] https://www.djangoproject.com

[6] https://docs.oracle.com/en/java/javase/12/core/process-api1.html

Hence, a suitable workflow should be set up in the hospital organisation before this kind of Decision Support System could be adopted in day-to-day practice.

Figure 22 shows how integration could be seen by users: besides usual SACM fields, the summary page could be added a few fields whose values are taken from the DSS predictions.

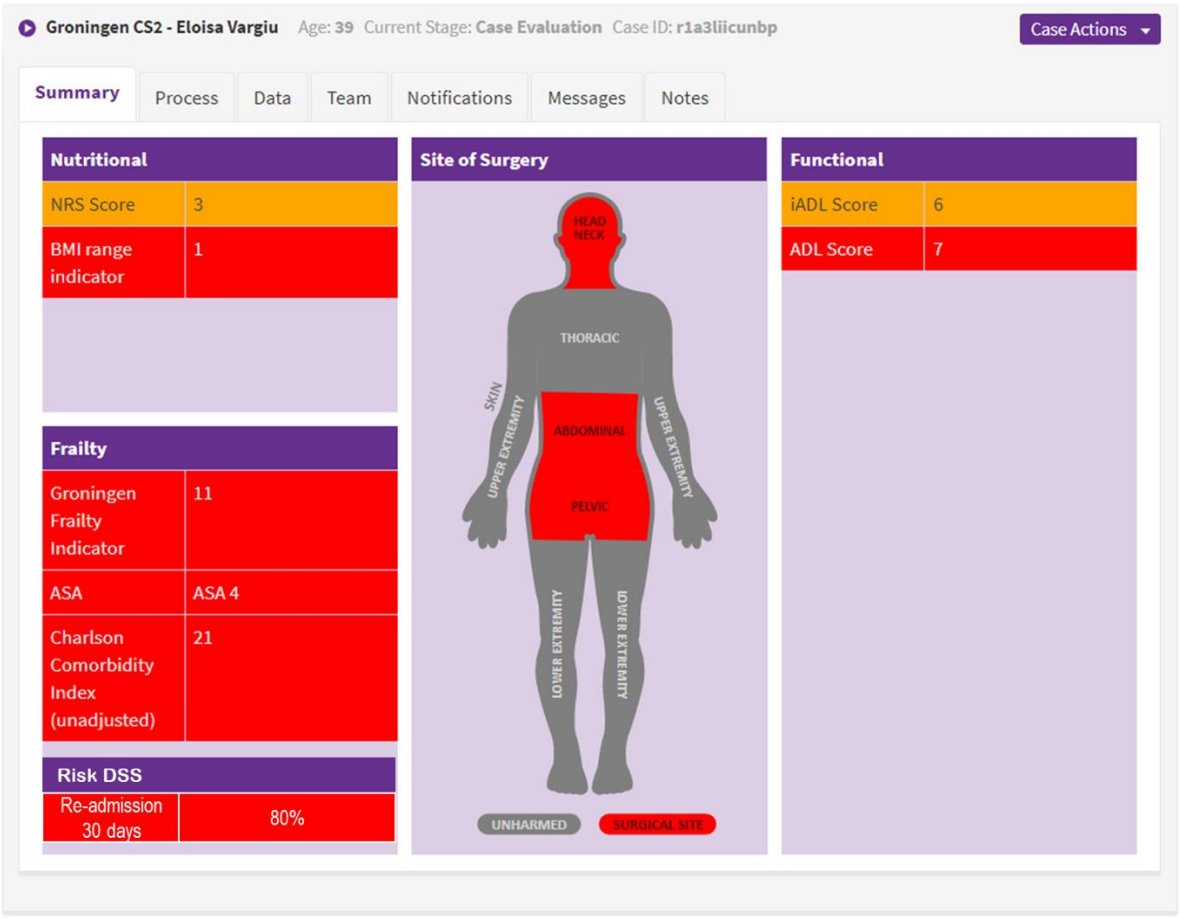


Figure 22 How the SACM summary page could look like when integrated with DSS.

3. The DSS for Mapping

The Mapping DSS is a software tool providing a **global view** of the cases (hence, patients) pertaining to a given clinician with the goal of facilitating *(i)* **monitoring** of patients' conditions and *(ii)* **focussing** on patients with specific conditions. The global view is based on a map, rendering patients as markers **geolocalised** according to their dwelling address as registered in the SACM. *Rendering* of patient markers is based on selected conditions of the case, such as scores of risk assessment questionnaires manually input by the professionals through the SACM, or barriers to treatment. *Filters* are available to enable the clinician to focus on selected conditions and render patient markers accordingly.

In the following sections, we describe the functionalities provided by the Mapping DSS as stemming from a requirement collections and co-design stage, mostly conducted as a joint effort with clinical partners in IRBLL, the software architecture designed accordingly, and the integration architecture enabling seamless integration with the SACM platform already available in clinicians' workstations.

3.1 Requirements

A requirements collection document has been circulated amongst clinical partners to gather high-level requirements on the kind of functions the Mapping DSS would make available, according to the following process:

1. in a first iteration, all the desiderata of all the interested clinical partners have been collected
2. in a second iteration these desiderata have been divided into *mandatory* requirements, and *potential* functionalities
3. afterwards, the technical requirements stemming from the high-level ones have been extracted by UNIMORE, with the goal to assess the technical feasibility of each functionality during a third iteration –this time with the technical partners leading development of the SACM frontend and backend
4. finally, the mandatory high-level requirements have been further detailed so as to provide precise implementation criteria regarding markers rendering

As a result of steps 1-2 along this process, the following mandatory functionalities have been implemented and are actually available in the first version of the Mapping DSS recently released in CONNECARE production environment:

A. *Locate* all patients belonging to cases of the professional currently logged in the SACM on the map
B. *Render* patients differently depending on a selected clinical metric amongst a pool of available ones

C. On click on patient *quick access* to the summary of the corresponding case in the SACM

D. *Automatic planning* of home visits to selected patients' dwelling

The following have been instead flagged as potential functionalities to be considered for future releases of iteratively improved versions of the Mapping DSS:

a. Render patients differently depending on whether they have active alerts (pending messages or warnings about monitored metrics thresholds violations)

b. Locate all available medical facilities on the map

c. Locate also patients' relatives/caretakers

d. Locate only the medical facilities relevant to the selected patients on the map

Nevertheless, soon after development started and the first prototype was shared with clinical partners to gather feedback as part of the co-design process, two brand new requirements arose –as well as a few enhancements to existing functionalities (not reported here but deferred to description of the Mapping DSS screenshots in Section 3.2):

e. Complement the map-based view with a scrollable, searchable **table-based view** of the same data

f. Enable printing a *report* of the data shown in the map

These high-level requirements have been translated into the following technical requirements in step 3 of the requirements collection process described at the beginning of this section:

- On SACM backend
    - Provide access to data of all the cases belonging to currently logged clinician
    - Provide data on active alerts, associated to either a case or a patient
    - Provide direct access to summary page of a given patient
- On Mapping DSS itself
    - Have well-defined criteria for implementing the rendering of markers based on clinical data
    - Have well-defined criteria for computing the relevance of medical facilities with respect to patients' clinical conditions
    - Have well-defined criteria for planning home visits routes
- On either SACM frontend or Mapping DSS own separate frontend, depending on design choice for integration (see Section 3.4)

- Rendering of a map
- Rendering of markers on the map
- Dynamic rendering process (can re-draw rendered elements on-the-fly, depending on changing criteria)
- Clickable markers
- Render routing information (e.g. road links between markers)

It is worth anticipating that all the technical requirements regarding the frontend have been tackled by the Mapping DSS, as for technical reasons it was not feasible to re-use SACM frontend –as described in Section 3.4.

The last step in the requirement collection process has been to thoroughly specify the *criteria* mentioned in the technical requirements, limited to the mandatory functionalities. Hence, *(i)* the pool of *clinical metrics* to consider as well as *(ii)* the *criteria for rendering* patient markers depending on patients' clinical conditions with respect to such metrics have been established, once again in co-design with the clinical partners in IRBLL. This task led to the following specification:

- A first set of clinical metrics influencing rendering of patients' markers according to the "traffic light model" (colouring markers in red, orange, or green according to decreasing severity) is called **risk scores** and includes (more details on each questionnaire in D3.6):
  - **LACE**. Clinical partners proposed to work with the same cut-off points implemented in the SACM, so as to be consistent:
    - Green: LACE score < 5
    - Orange: 4 < LACE score < 10
    - Red: LACE score > 9
  - **Charlson**. Given that a patient is considered complex if the Charlson index is ≥ 3, clinical partners proposed to work with only two cut-off points:
    - Green: 2 < Charlson score < 6
    - Red: Charlson > 5
  - **GMA**. Green if GMA is 1 or 2, orange is GMA is 3, red if GMA is 4
  - **Barthel**. Green if between 90 (excluded) and 99 (included), orange if between 60 (excluded) and 90 (included), red if below 60 (included)
  - **ASA**. Green if ASA category is II, red if it is III (any other category is base marker colour, hence purple)

- A second set of clinical metrics influencing rendering of patients' markers by increasing / decreasing transparency (higher transparency corresponds to a less severe condition) is called **barriers** and includes[7]:
    - **Skills to carry out treatment**. Based on a custom questionnaires used in Lleida and ASSUTA (namely, Treatment adherence and capability to follow up), the score ranges from 0 to 4, and transparency is the lowest (hence, most severe condition) above 2 (included) and the highest at 0
    - **Retrieval of tablets from pharmacy**. Another custom questionnaire (namely, Medication adherence) with one question only, hence the score is either "Yes" or "No". Transparency is the lowest if answer is "No"
    - **Selfcare skills**. Another custom questionnaire (namely, Self-management capability) whose score ranges from 0 to 4, hence treated likewise the "Skills to carry out treatment" (lowest transparency above 2, included)
    - **Dwelling**. Another custom questionnaire (namely, Dwelling status) treated likewise the "Skills to carry out treatment"
    - **Carer skills**. Another custom questionnaire (namely, Carers' capabilities) treated likewise the "Skills to carry out treatment"

Based on these criteria, the Mapping DSS calculates the transparency to give to the patient marker, from a minimum of 0 (fully opaque marker) to a maximum of 80% (barely visible marker), also considering that multiple barriers may be selected simultaneously (contrarily to risk scores).

This specification guided the implementation process of the Mapping DSS functionalities, thoroughly described in next section.

3.2 Functionalities

For the sake of clarity, we describe each functionality provided by the Mapping DSS using as a reference the screenshot of its window as it appears upon launch depicted in Figure 23. How to get to the Mapping DSS page is discussed in Section 3.4, as it happens solely by logging into the SACM.

[7] Here the names are listed has appear in the Mapping DSS interface, the name used at Case Evaluation time is put into parenthesis in the description. The full list of questionnaires used in each sites and, thus, in Lleida is given in D3.6

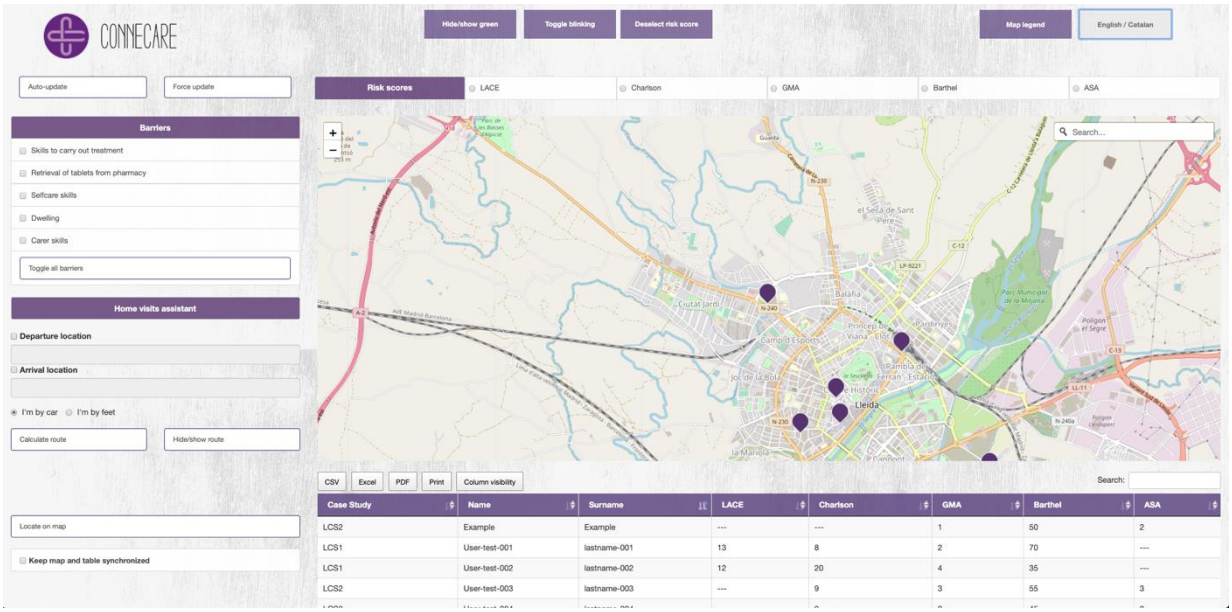


Figure 23 The Mapping DSS frontend upon launch.

The DSS mapping window is organised in *5 areas*, as emphasised in Figure 25:

1. The topmost area (A) provides two sets of buttons: the three on the left provide for configurations of the rendering and filtering of the markers, whereas the two on the left provide utility functions. In particular:
    a. Button “Hide/show green” enables fine-grained filtering of the markers (hence, patients) to show on the map, allowing the clinician to focus on a certain degree of severity by hiding patient markers coloured in a certain way (green, orange, or red, following the traffic light model), according to the currently selected risk score. Figure 24 shows an example where the clinician wants to focus on the most severe patients (red markers).

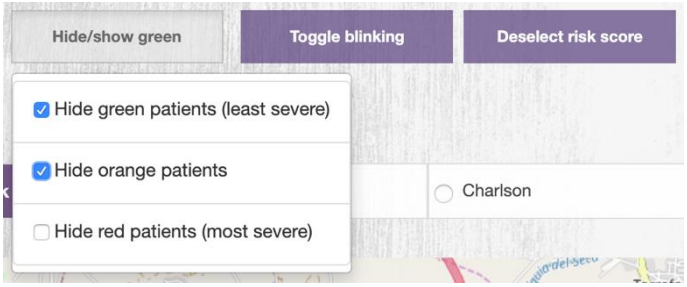


Figure 24 Filter on patients severity.

    b. Button “Toggle blinking” enables/disables the blinking decoration for markers with pending messages/alerts (see point 2.c below).
    c. Button “Deselect risk score” removes selection of the risk score, which causes markers to reset to default colour (purple).

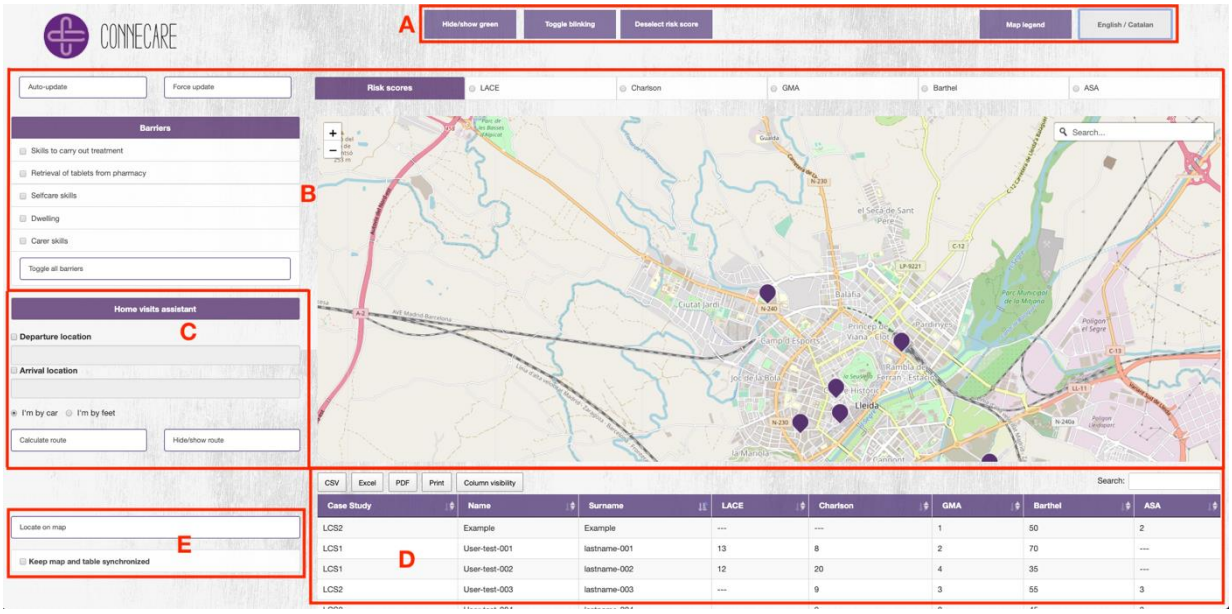


Figure 25 The five areas organising the functionalities.

d. Button “Map legend” displays a simple explanation of the colour codes upon hovering with the mouse cursor.
e. Button “English / Catalan” switches the language of the main window elements.

2. The central area (B) is the largest one as it provides the core functionality of the Mapping DSS: localisation of patient on an interactive map as clickable markers rendered according to the selected clinical criteria. In particular:
   a. The topmost set of radio buttons (labelled “Risk scores”) enables selection of one and only one at a time of the 5 risk scores required by clinical partners, which will set the colour of the markers according to the criteria described in the 4th step of the process described in Section 3.1. Figure 26 and Figure 27 show, respectively, an example where some patients do not have a LACE score yet, hence they remain with the default colour, and one where the whole spectrum of severity is represented.

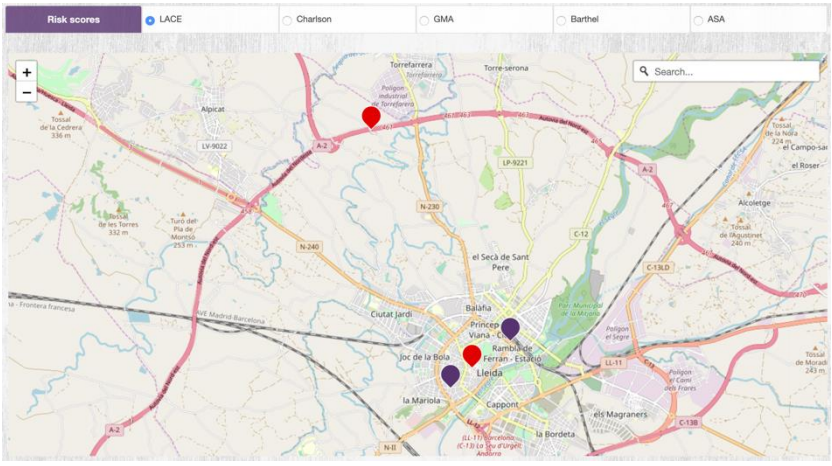


Figure 26 An example of risk score selection.

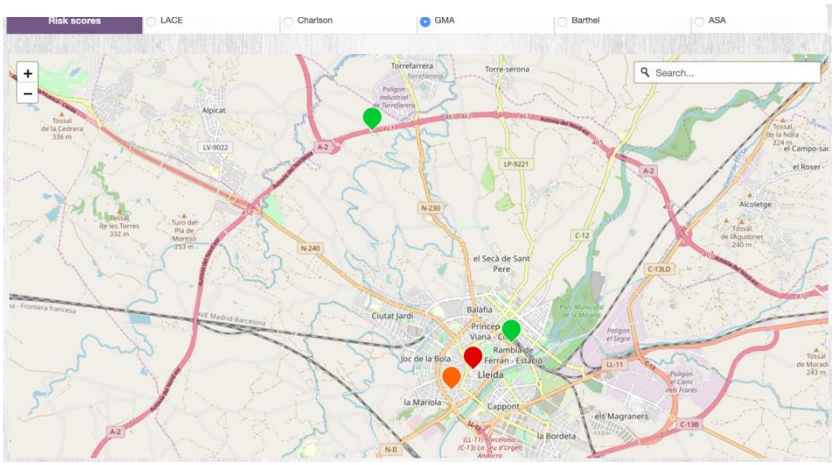


Figure 27 Another example of risk score selection.

b. The leftmost set of checkboxes (labelled “Barriers”), instead, enable selection of the possibly multiple barriers required by clinical partners, which will set the transparency of the markers according to the criteria described in the 4th step of the process described in Section 3.1. Figure 28 shows an example where two barriers have been selected, hence some markers became partially transparent; in particular, all the three markers at the bottom of the map gained some transparency, indicating that the selected barriers are NOT concurring in worsening the patients’ condition. A button is also provided below the barriers to either select or deselect all of them at once.
c. The map itself is interactive in a number of ways: the visible area may be zoomed in and out either using the mouse controls (typically, the wheel) or the “+ / -“ buttons on the top left corner; the visible area may be also moved by clicking and dragging around with the

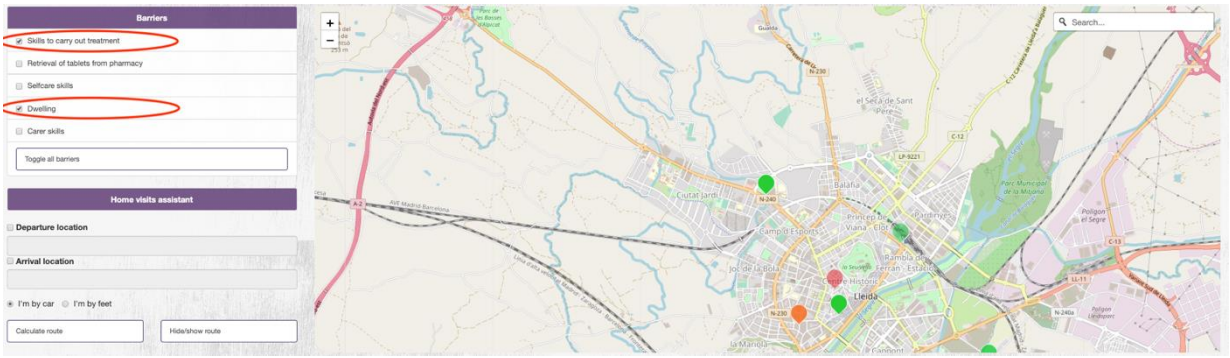


Figure 28 An example of barriers selection.

mouse cursor; the search box in the top right corner enables searching for locations (e.g. addresses, Points of Interest, etc., basically what you would search for in systems such as Google Maps); finally, and perhaps most importantly, clicking on markers shows a popover decorated with the patient picture (if available), its dwelling address (as registered in the SACM), a “Go to summary” button enabling quick jump to the summary screen of the selected patient in the SACM, and an indication of whether the patient has pending messages or alerts. The popover with the button and the summary screen are shown in Figure 29 and Figure 31, respectively. Figure 29 also shows the clickable pending messages alert, which brings to the messages screen in the SACM (shown in Figure 32). Likewise, in Figure 30 the clickable pending alerts notice is shown, which brings to the alerts screen in the SACM (shown in Figure 33). Also, the “blinking effect” is visible (in picture below the purple marker is temporarily invisible).

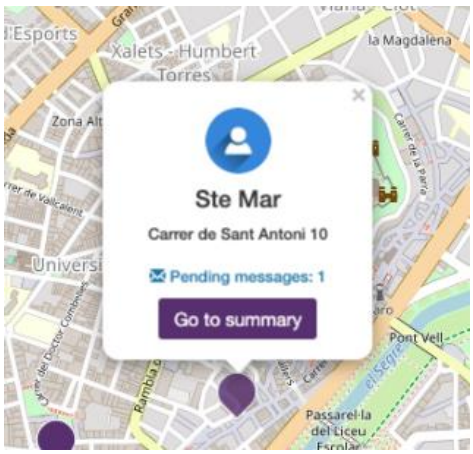


Figure 29 An example of the popover.

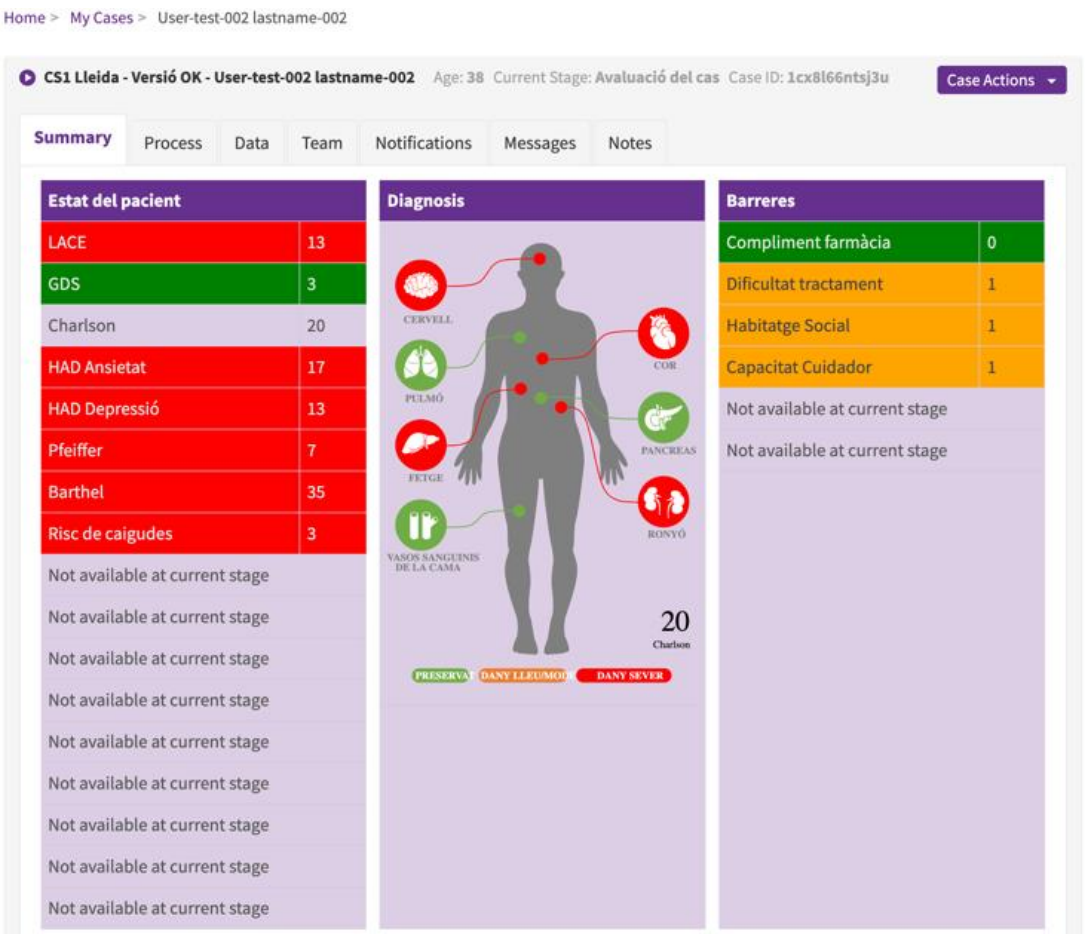


Figure 31 An example of the SACM summary screen.

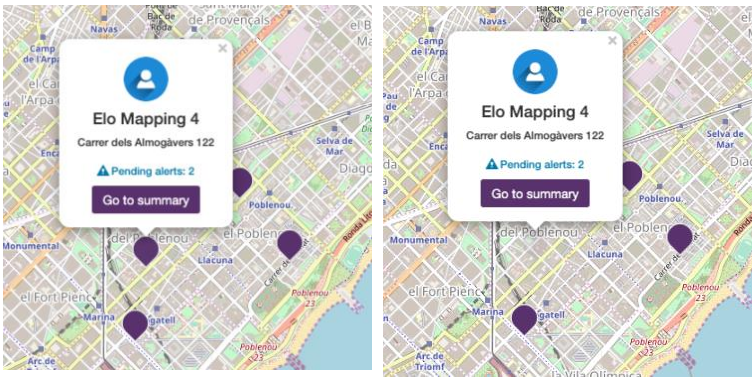


Figure 30 Example of blinking effect and pending alerts clickable notice.

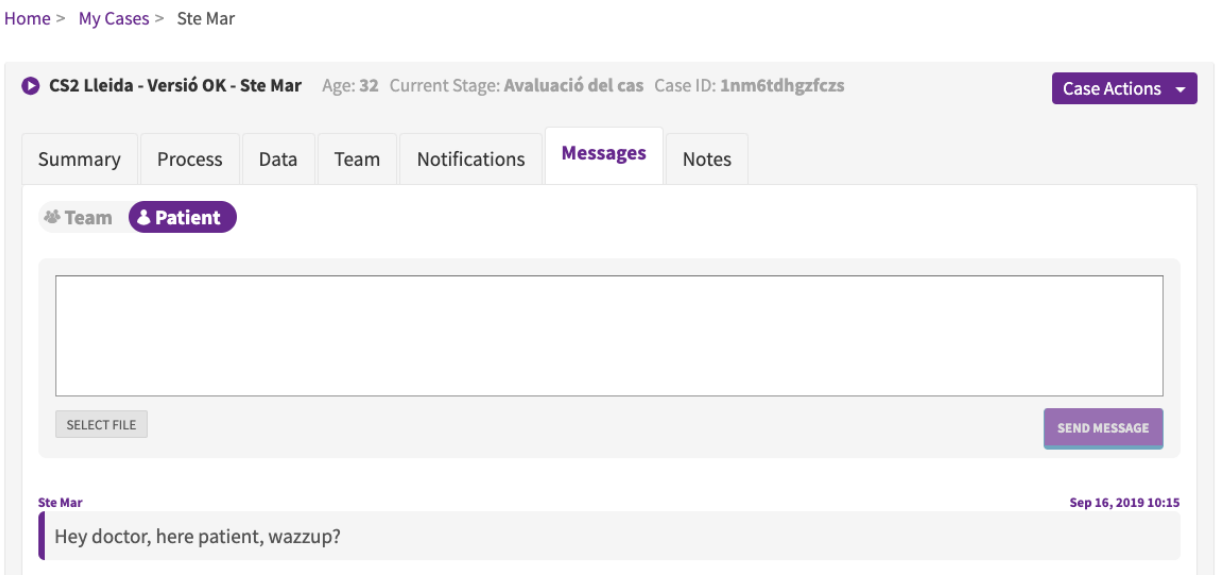


Figure 32 The SACM pending messages screen, target of the link in the popover shown in Figure 29.

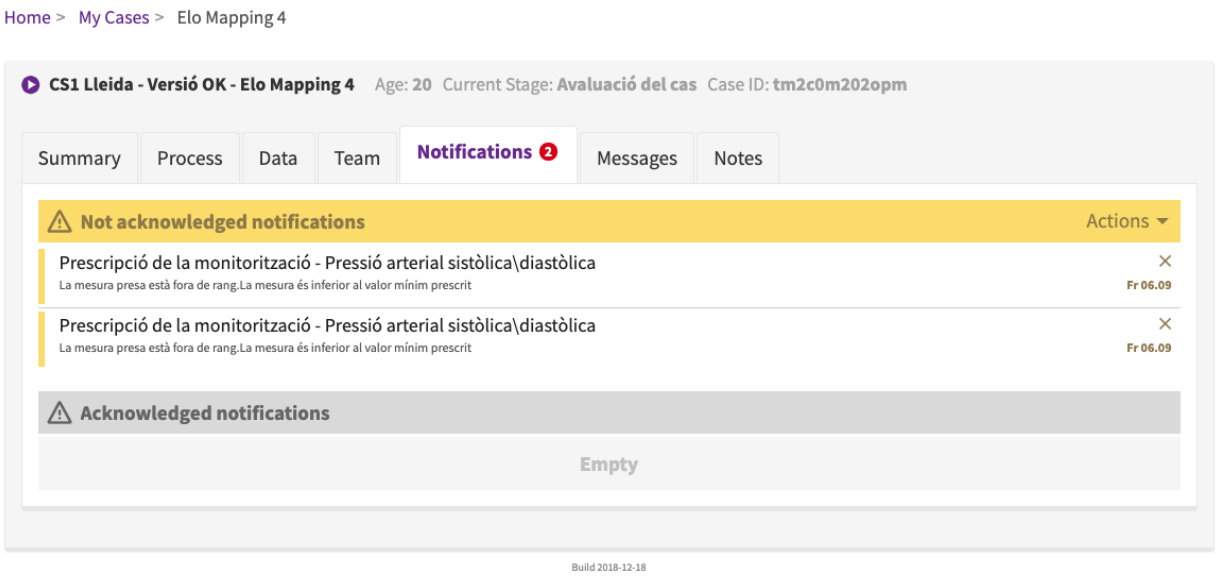


Figure 33 The SACM pending alerts screen, target of the link in the popover shown in Figure 30.

3. The area complementing the central part (C) is meant to assist clinicians in planning home visits: by selecting a set of patients from the table view and then clicking on button “Calculate route”, a driving route visiting all the patients is displayed, and navigation instructions appear on a dedicated popover window right below the search box, as depicted in Figure 34. Besides choosing which patients to visit, the professional may choose a starting and/or ending point for the route (e.g. the hospital, or its home) by putting the address in the dedicated text fields. If left blank, the route will be a closed loop visiting all the patients. Finally, given the fact the, for instance

in Lleida site, the city centre is closed to cars and only walkable by feet, a choice whether the suggested route should consider driving by car of going by feet is available from a dedicated radio button (circled in Figure 34).

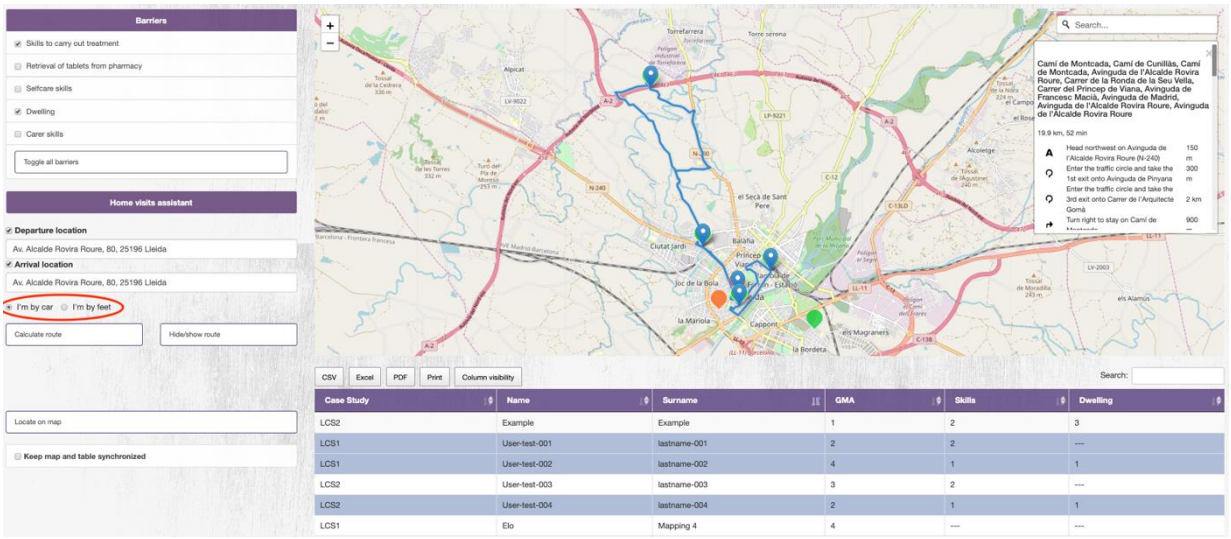


Figure 34 An example of a routing request.

4. The bottom area (D), is dedicated to the table view, also delivering export and printing functionalities. In this table, a few selected information is displayed by default for each patient: the case study number, name and surname, the risk scores values. Then, barriers too are displayed as soon as they are selected from the respective checkboxes –as depicted in Figure 35. This choice has been done in compliance with Lleida partners requirement of avoiding cluttering the UI with too much information as much as possible, promoting strong focus on a few things at once. The table view based functionalities are the following:
    a. The table is interactive: clicking on rows selects the patient and opens the associated popover on the map; clicking on columns sorts the table rows according to that column.
    b. Button “Column visibility” enables customisation of the columns displayed.
    c. Navigation buttons enable browsing the table pages sequentially or by directly jumping to the desired page.
    d. A dedicated search box enables clinicians to quickly look for patients whose data matches the inserted text (data in any column is searchable, hence both name and surname as well as risk scores and barriers values).
    e. Buttons “CSV”, “Excel”, and “PDF” enable to export data to the correspondent format. If filters on data displayed are active, such as searching for specific patients or configuring the columns visibility, such filtering will be preserved during export –as shown in Figure 36.

f. Button “Print” enables quick printing of the filtered data –thus, again any filtering applied is preserved during printing –as shown in Figure 37.

5. The bottom left area (E) provides functionalities requiring collaboration between the map view and the table view, that have been added to the original requirements in an effort to improve clarity of the User Interface and configuration options. In particular:
   a. Button “Locate on map” enables to position the map in the smallest visible area including all the patients selected in the table view (clicking a row selects a patient, clicking more rows while pressing keyboard button “ctrl” on Windows/Linux, or “cmd” on macOS), while also opening their popovers. This is particularly useful in combination with the search functionality provided by the table view (through the search box in the top right corner of the table), as it enables quick view of searched patients (and, eventually, quick access to their summary page).
   b. Checkbox “Keep map and table synchronised” enables to keep the data shown in the map view and in the table view coherent. This means that the table will show only those patients which are currently within the visible area of the map, and, otherwise, the map will show the smallest visible area including all the patients selected in the table view. Similarly to the previous functionality, this is particularly useful in combination with the search functionality provided by the table view.

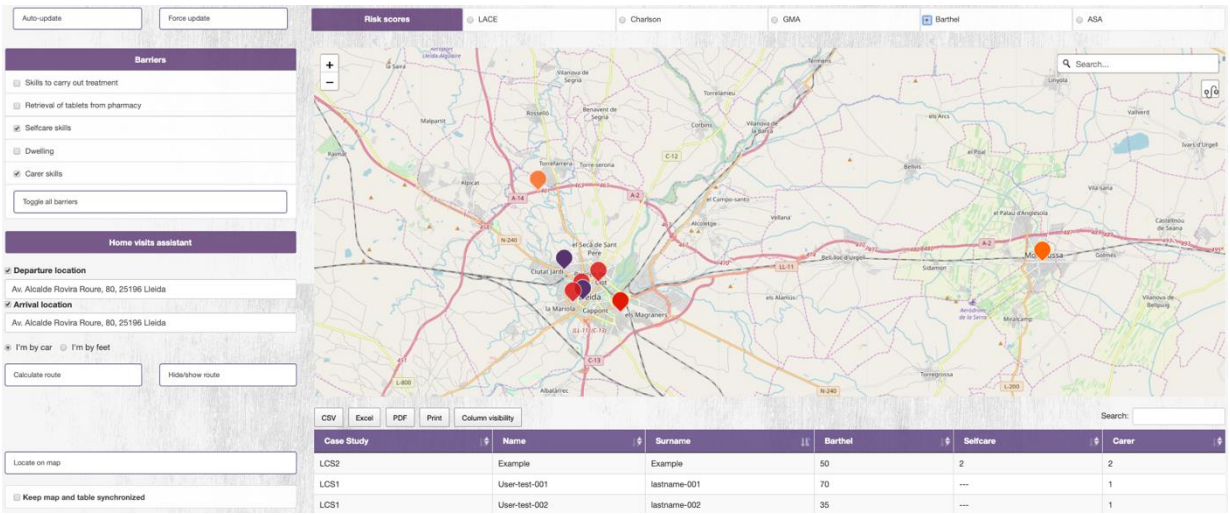


Figure 35 Dynamic columns display depending on barriers selection.

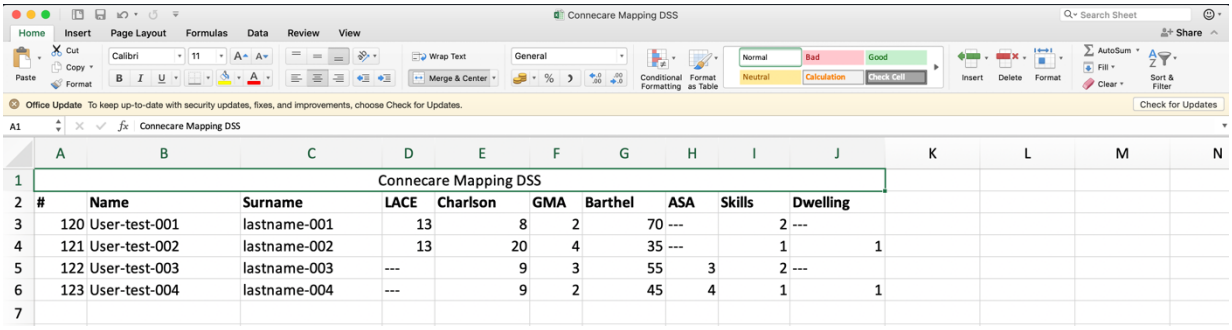


Figure 36 Export of filtered data to Excel format.

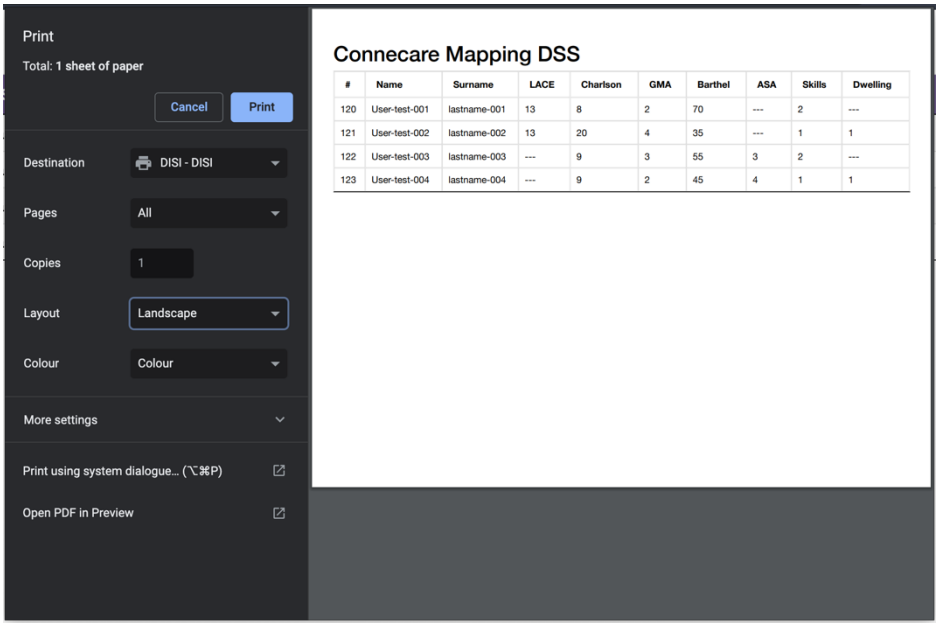


Figure 37 Printing report on filtered data.

As commented on in Section 3.5 the Mapping DSS has been featured in a demo shown to professionals in Lleida, which showcases some of its features mentioned above. The video of the demo is not publicly available due to sensitive data belonging to the patients of the implementation studies in Lleida. The video will be suitably edited or informed consent will be asked to involved patients so as to show it during the final review.

3.3 Architecture

The Mapping DSS is a system actually composed by two separate executables –depicted in Figure 38: the frontend, that is, the web page displayed to the logged clinician, and the backend, that is, the server providing to the frontend the data it needs to function properly. The former integrates with the SACM frontend, while the latter with the SACM backend, as described in Section 3.4.

The frontend is the one actually shown in the screenshots of Section 3.2: it is a dynamic web page exploiting state of art web development technologies such as HTML5, Bootstrap4, JQuery, Javascript, and served by the NGINX open source web server technology. For the map functionalities the Leaflet open source JavaScript library has been used, extended with a few of its plugins (e.g. Leaflet Routing Machine and Leaflet Control Geocoder), while for the table view the JQuery plugin DataTables has been exploited, extended with a few of its extensions providing additional capabilities (e.g. the Buttons extension for export and printing, and the Columns extension for configuring columns visibility).

The backend is a RESTful web service implemented on top of the Node.js JavaScript runtime. The base modules http and https have been used to handle RESTful requests, the Express framework to expose RESTful endpoints, and the body-parser JavaScript middleware, integrated with Express, to handle JSON data.

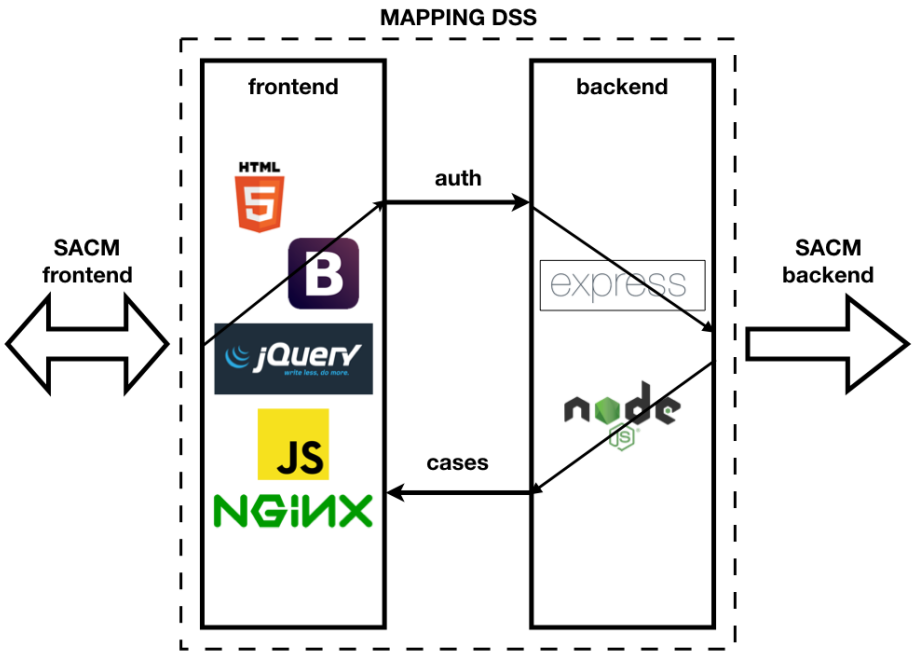


Figure 38 The Mapping DSS inner architecture.

Responsibility of computations to be carried out for providing the Mapping DSS functionalities described in Section 3.2 is attributed as follows:

- The backend takes care of data fetching and parsing so as to only retain the data relevant to the frontend amongst all the data delivered by the SACM backend RESTful endpoints
- The frontend takes care of implementing the criteria described in Section 3.1 so as to compute the correct rendering of the patient markers

- A few functionalities are delegated to 3rd party services, such as geocoding and routing (obtained through appropriate Leaflet plugins, as already described)

The flow of interactions supporting operation of the Mapping DSS is as follows:

1. The Mapping DSS backend is launched at release time as a RESTful web server running on CONNECARE production environment servers. Launch of the executable happens by using the Docker platform for distribution of the application image and containerisation of the application process, supporting high flexibility in application management (e.g. restarting the server, migrating the application, etc.)
2. The Mapping DSS frontend is launched by the SACM frontend upon user interaction (as clarified in Section 3.4)
3. At launch, the frontend authenticates against the backend, then asks for data to populate the map view and the table view
4. The Mapping DSS backend periodically checks whether updated data is available from the SACM backend

It is worth clarifying that during user interaction with the Mapping DSS the data is temporarily stored in the Mapping DSS backend, so as to avoid large bandwidth consumption and too much network interactions with the SACM backend, that would lead to high latency from the user perspective.

3.4 Integration

The Mapping DSS has been deployed already integrated with the SACM in the production environment, which means that it is not perceived by users as a separate application, but as an integral part of the SACM platform.

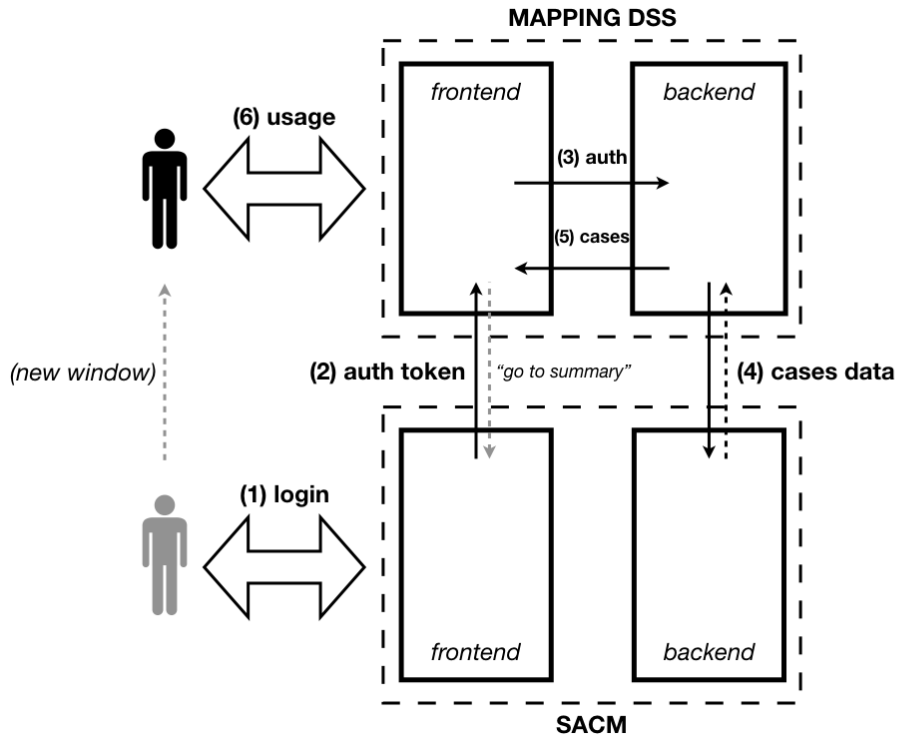


Figure 39 Architecture of the integration of the Mapping DSS with the SACM.

The flow of interactions is depicted in Figure 39, and is as follows:

1. A clinician logs into the SACM through its frontend, which causes an authentication token to be generated based on the user login credentials (if they are successfully validated)
2. The clinician then clicks on the "Go To Mapping" button to open the Mapping DSS frontend –as depicted in Figure 40. While redirecting the user to the new web page, the SACM frontend passes the authentication token to the Mapping DSS frontend by appending it to its URL
3. The Mapping DSS frontend then, in turn, forwards the token to its backend, which stores it for later usage (namely, inclusion as "Authorisation" header in all HTTPS requests to the SACM backend). In the meanwhile, the Mapping DSS frontend also asks to the Mapping DSS backend the data it needs to operate (namely, the list of cases for the logged user, there including data about patient of the case and its risk scores and barriers)
4. At this point, the Mapping DSS backend starts a round of interaction with the SACM backend – periodically repeated to stay up-to-date with respect to, for instance, insertion of new patients or completion of pending risk scores / barriers questionnaires –that encompasses the following steps (which are constrained by the data provided by the available SACM endpoints):
    a. Fetch the list of all the cases ID associated to the authentication token of the currently logged clinician (this ensure access control, thus privacy)
    b. For each case ID, fetch the list of pending messages and alerts

c. For each case ID, fetch the list of scores and barriers, provided that they have an associated value (namely, that the corresponding questionnaires / measurements have been completed)
d. For each score / barrier, fetch its value

5. The Mapping DSS backend replies to its frontend as soon as the required data has been acquired from the SACM backend, and then either waits for new connections (new users logging in) or for the update interval timer to trigger a check for updates
6. Finally, the Mapping DSS frontend displays the data acquired by its backend to the logged clinician, which may then start interacting with the UI. This interaction may include the user click on the “Go to summary” button of one or multiple markers, in which case the Mapping DSS frontend triggers the SACM frontend by opening the summary screen (see Figure 31) of the corresponding patient

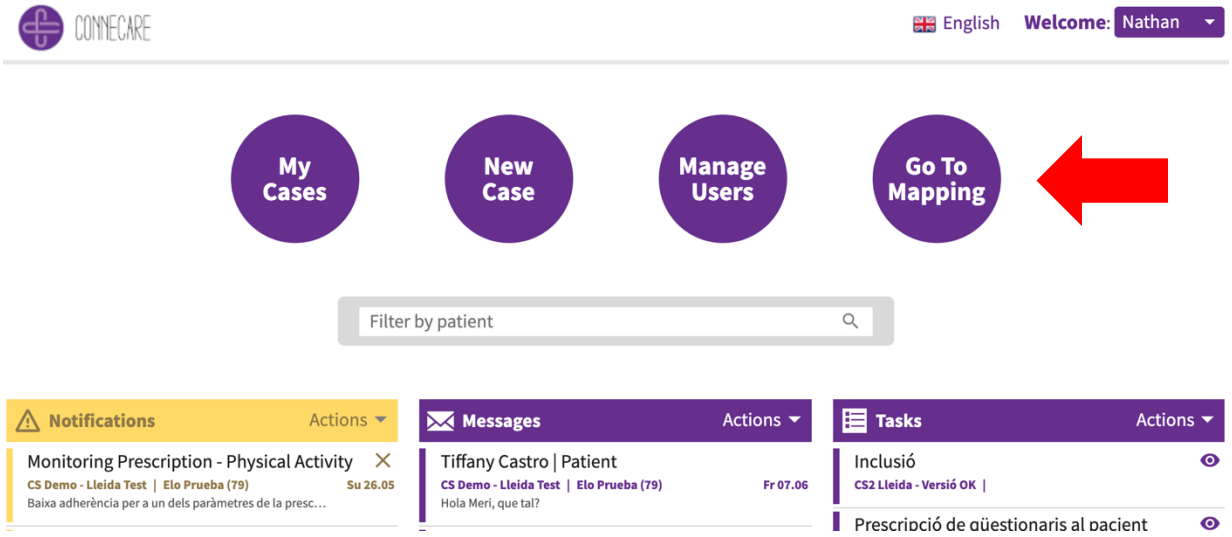


Figure 40 The point of access to the Mapping DSS from the SACM frontend (red arrow added).

3.5 Evaluation

The Mapping DSS has been evaluated by using the System Usability Scale (SUS) questionnaire on a pool of testers and viewers [1]. Testers were 4 professionals working in Lleida that used the Mapping DSS deployed in CONNECARE production environment during the first release of the Mapping DSS between August and September 2019. Viewers were 19 more professionals from Hospital Santa Maria and Hospital Arnau de Vilanova in Lleida and also primary care doctors from different CAPs of the Lleida Region, who attended a meeting in which a live demonstration of the Mapping DSS in CONNECARE production environment has been shown.

When considering all the responders (n=23) median (p25-p75) SUS was 68 (59-73), which rates the product as “ok" (50-70) and almost "good" (70-80) –Figure 41.

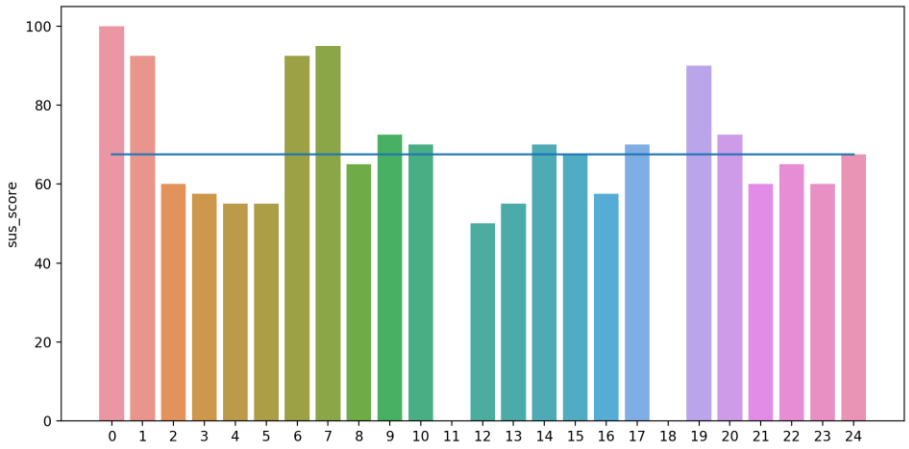


Figure 41 SUS score for each responder, with median.

When considering Case Study 1 (n=20) median SUS was 66 (58-79), whereas for Case Study 2 (n=15) median SUS was 68 (60-71). When restricting to testers (n=4) median SUS was 76 (59-94), whereas when considering potential users only (n=19) median SUS was 68 (59-71). Figure 42 plots the distribution of SUS score per case study (professional involved in CS1 only, CS2 only, or both) and per kind of user (testers or viewers).

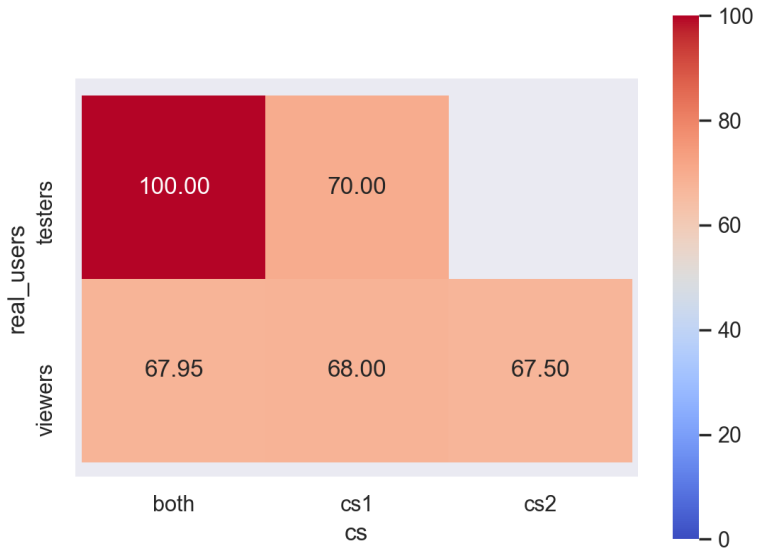


Figure 42 Distribution of SUS score for case studies (x-axis) and users (y-axis).

Besides overall SUS score, it is worth to analyse how the Mapping DSS scored has been perceived in general (Figure 43) and how friendly it appeared to users (Figure 44): for both charts, active users scored higher than potential users, which a nice indication that using the software improves the opinion of the users.

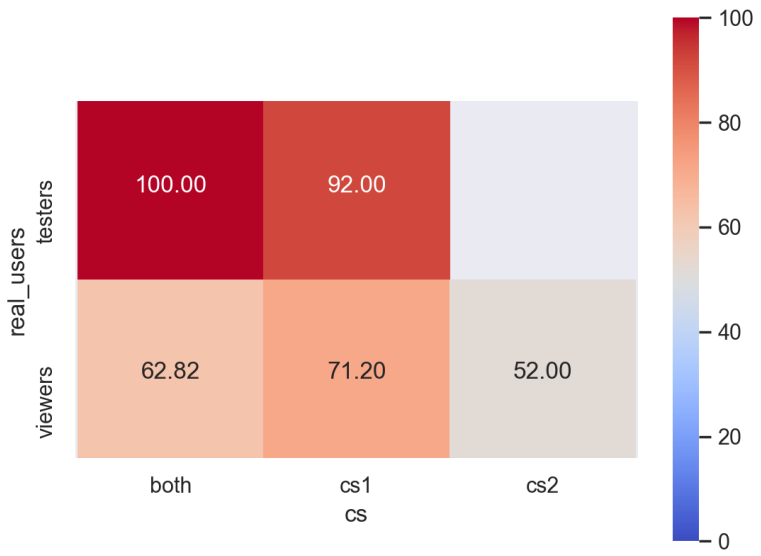


Figure 43 Distribution of "general impression" score, per case study and user.

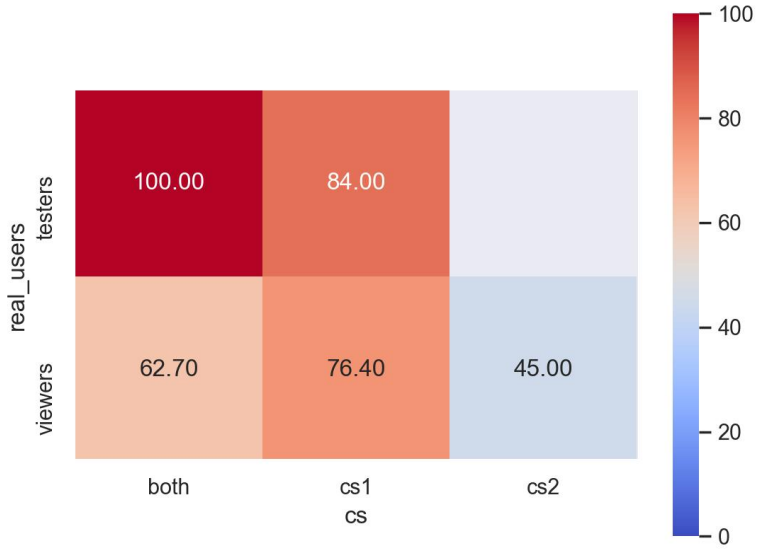


Figure 44 Distribution of "user friendliness" score, per case study and user.

Finally, it is worth zooming-in three specific questions from the SUS questionnaire, two of which linked to the previous charts: whether users would like to use the mapping DSS frequently (Figure 45), whether they would like to learn how to use it[8] (Figure 46), and whether they think technical assistance is needed to learn usage (Figure 47). A vast majority of responders would like to learn using the tool, and think they will use it frequently, whereas the need for technical assistance in the learning and usage phases is more or less evenly split amongst positive and negative opinions, which is a nice indication than technical assistance is not needed by everyone.

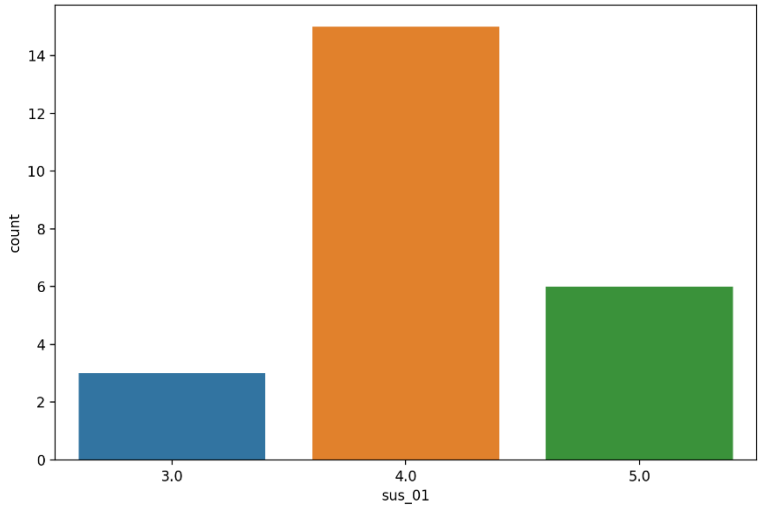


Figure 45 SUS question 1, about willingness of frequent usage.

[8] A comprehensive user manual, annex to D2.5, has been provided to IRBLL.

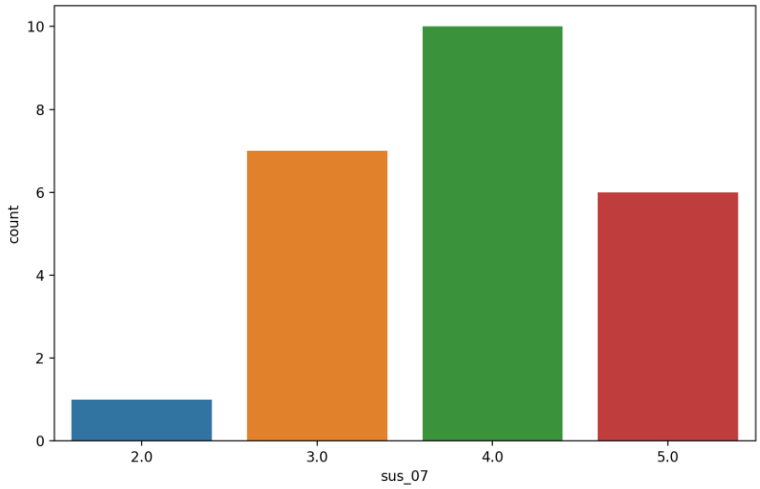


Figure 46 SUS question 7, about willingness of learning to use the tool.

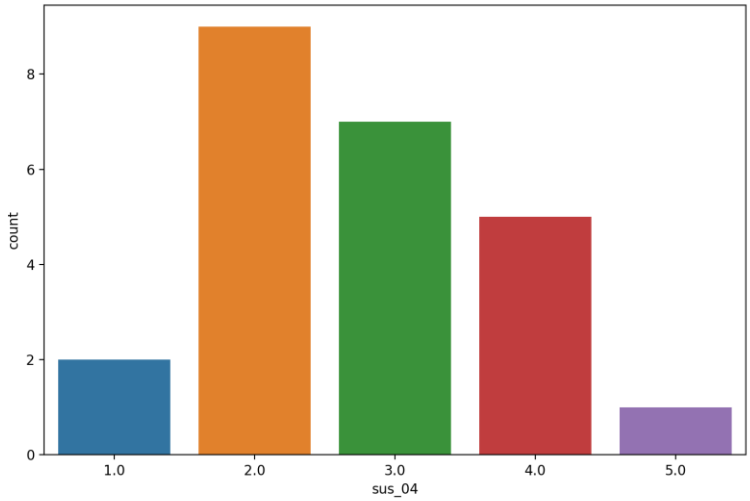


Figure 47 SUS question 4, about need of technical assistance for proficient usage.

In summary, data shows that the Mapping DSS has been received positively by the healthcare professionals in Lleida, where it has been deployed.

4. Outlook of Future Research and Implementation

The Risk DSS has different levels of maturity depending on the mode of operation of concern. For the plugin mode, we can safely state that it represents an advance of the state of art in building services and infrastructures for seamlessly deploying risk prediction models across computational platforms and languages, which could impact the current clinical practice by speeding up the process of model building and validation by fostering collaboration amongst heterogeneous teams.

For the learning mode, instead, it would be necessary both to gather more data into the CONNECARE platform so as to train models from scratch directly on the data they will operate on afterwards, and to run the software in CONNECARE production environment long enough not only to generate a reasonable number of predictions, but also to actually measure the actual value of the predicted data, to assess performance. Nevertheless, the evaluation conducted on UMCG dataset shows promising results and represents a preferential choice if such a decision support system is decided to be put at work in production environments.

UMCG already manifested interest is nurturing collaboration to improve prediction models and expand to other prediction targets, and IRBLL (Lleida) was also interested in the functionality.

The Mapping DSS instead has been actually released and used in the CONNECARE production environment as a tight integration with the SACM, hence collection of users' evaluation has been performed showing good results concerning the perceived value-added by the technology to the professionals' workflow. Indeed, IRBLL already manifested the intention to further collaborate, either maintaining integration with the SACM or making the Mapping DSS a standalone tool.

Further development may be related to flexibility of the tool to be configured with different risk scores and barriers, so as to be easily portable to other sites in CONNECARE. Also, additional functionalities such as location on the map of relevant medical facilities could be added. Finally, the Mapping DSS has been conceived as a Decision Support tool for clinicians and medical staff, but nothing prevents it to offer functionalities to patients, too, possibly through an integration with the SMS.

5. Conclusion

This document reports the work done to deliver an improved version of the Risk DSS, as well as the software artefact itself, so as to accomplish Task 3.4 “Screening and risk stratification DSS”, directly contributing to the specific objectives of WP3 regarding the provisioning of “*ICT tools for the adaptive case management of personalised clinical pathways (OBJ2), which takes into account the patient’s medical history*” (objective OBJ2 in DoA). Also, it contributes to make “Healthcare professionals [to be] continuously and proactively supported in their decisions through Decision Support Systems (DSSs) for: screening and risk stratification; mapping; and intervention and surveillance (suggesting personalized clinical pathways)” (OUT 3-4 in DOA).

Both the Risk DSS and the Mapping DSS may be adopted as stand-alone solutions (as actually the former is) or integrated to 3rd party services, such as the SACM (as the latter is). The Mapping DSS shown promising appreciation both by testers and viewers, hence further exploitation of the tool has been request by IRBLL in Lleida, and plans to nurture collaboration beyond CONNECARE lifespan are under discussion. The Risk DSS shown promising results but should be further refined and compared to state-of-art solutions for performance.

6. References

1. Cano I., Lluch-Ariet M., Gomez-Cabrero D., et al.: *"Biomedical research in a Digital Health Framework*" in Journal of Translational Medicine (2014). doi:10.1186/1479-5876-12-S2-S10
2. Viceconti M., Hunter P., Hose R. *"Big data, big knowledge: big data for personalized healthcare"* in IEEE Journal of Biomedical Health Informatics (2015). doi:10.1109/JBHI.2015.2406883
3. Andreu-Perez J., Poon C.C.Y., Merrifield R.D., et al. *"Big Data for Health"* in IEEE Journal of Biomedical Healthcare Informatics (2015). doi:10.1109/JBHI.2015.2450362
4. Hamad R., Modrek S., Kubo J., et al. *"Using 'big data' to capture overall health status: properties and predictive value of a claims-based health risk score"* in PLoS One (2015). doi:10.1371/journal.pone.0126054
5. Murdoch T.B., Detsky A.S. *"The inevitable application of big data to health care"* in JAMA (2013). doi:10.1001/jama.2013.393
6. Belle A., Thiagarajan R., Soroushmehr S.M.R., et al. *"Big Data Analytics in Healthcare"* in Biomedical Research International (2015). doi:10.1155/2015/370194
7. Luo J., Wu M., Gopukumar D., et al. *"Big Data Application in Biomedical Research and Health Care: A Literature Review"* in Biomedical Informatics Insights (2016). doi:10.4137/BII.S31559
8. Gerstein M. *"Genomics: ENCODE leads the way on big data"* in Nature (2012). doi:10.1038/489208b
9. Robertson A.R.R., Nurmatov U., Sood H.S., et al. *"A systematic scoping review of the domains and innovations in secondary uses of digitised health-related data"* in Journal of Innovative Healthcare Informatics (2016)
10. Herrett E., Gallagher A.M., Bhaskaran K., et al. *"Data Resource Profile: Clinical Practice Research Datalink (CPRD)"* in International Journal of Epidemiology (2015). doi:10.1093/ije/dyv098
11. McDonald L., Lambrelli D., Wasiak R., et al. *"Real-world data in the United Kingdom: opportunities and challenges"* in BMC Med (2016). doi:10.1186/s12916-016-0647-x
12. *"Big Data in Public Health, Telemedicine and Health"* available at https://ec.europa.eu/digital-singlemarket/en/news/study-big-data-public-health-telemedicine-and-healthcare
13. Pivarski J., Bennett C., Grossman R.L. *"Deploying Analytics with the Portable Format for Analytics (PFA)"* in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2016). doi:10.1145/2939672.2939731
14. Fielding R.T. *"Architectural Styles and the Design of Network-Based Software Architectures"* Ph.D. Dissertation (2000) at University of California, Irvine (AAI9980887)
15. Ribeiro M., Grolinger K., Capretz M.A.M. *"MLaaS: Machine Learning as a Service"* in IEEE 14th International Conference on Machine Learning and Applications (2015). doi:10.1109/ICMLA.2015.152

16. Mariani S., Zambonelli F., Tenyi A., et al. *"Risk Prediction as a Service: a DSS architecture promoting interoperability and collaboration"*, to appear in proceedings of the 32th IEEE International Symposium on Computer-Based Medical Systems (CBMS 2019)
17. J. Pivarski, C. Bennett, and R. L. Grossman, *"Deploying analytics with the portable format for analytics (pfa)",* in Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, (2016). doi:10.1145/2939672.2939731
18. Metting EI, In't Veen JC, Dekhuijzen PN, et al. *"Development of a diagnostic decision tree for obstructive pulmonary diseases based on real-life data"*, in ERJ Open Research (2016). doi: 10.1183/23120541.00077-2015
19. Hathaway R.J., Bezdek J.C. *"Switching regression models and fuzzy clustering"* in IEEE Transactions on Fuzzy Systems (1993).

7. List of Figures

8. Appendix

8.1 UMCG dataset analysis

UMCG provided UNIMORE with a dataset on asthma / COPD patients under a non-disclosure agreement as the dataset has not been generated within CONNECARE, but in another past project. The dataset has been exported from the SPSS proprietary software[9], which cannot be used as it cannot be embedded in web services as we want to do with the Machine Learning (ML) pipeline for the Risk DSS. Also, Python offers much more powerful capabilities, hence a first considerable effort has been devoted to transform the dataset from the SPSS format to a Python DataFrame. Automatic conversion has been possible, but manual checking of coherency w.r.t. original data and reconstruction of attributes meta-data (description, value ranges, frequencies, etc.) has been done by hand through Python programming.

The dataset contains baseline assessment of 19077 patients' conditions concerning Ashtma / COPD, and follow-ups information at different time points (mostly, 3 and 12 months, as per below analysis). Attributes describe basic data such as age, gender, BMI, and family history of the disease, habits associated to the syndrome such as smoking and inhalation technique, lab measurements such as Forced Expiratory Volume in 1 second (FEV), Forced Vital Capacity (FVC), and lung function assessed by pneumologist, symptoms such as coughing, short breath, and wheezing, questionnaires results such as ACQ and CCQ, and medications taken. All the measurements, symptoms, questionnaires, and in general any attribute whose value may change over time is repeated at each follow up. For this reason, in the original dataset there are 2454 attributes, which can be reduced to 164 after:

- filtering out columns with all missing values for follow up at 3 and 12 months (considering the attribute at baseline would be meaningless if no value is known for doing predictions at 3 or 12 months)
- filtering out repetitions (follow-ups), that is, focusing on each unique attribute sampled at baseline (then repeated at each follow-up, in particular, at 3 and 12 months as per below analysis)

Follow-ups are not homogeneous in the time interval between each other, hence data samples have been grouped according to the most common follow-up intervals, which are at 3 and 12 months, as depicted in Figure 48. Since a high variance is present amongst follow-ups dates, the aforementioned categories have been devised out by grouping the actual differences amongst baseline assessment date and second visit (or, first follow-up) date, as depicted in Figure 49. There, the central graph shows the distribution for follow-ups categorized as "3 months" (orange bar in Figure 48), the right-most graph the distribution for "12 months" (green bar in Figure 48), whereas the left-most graph the distribution for "other" follow-ups (blue bar in Figure 48). It can be easily seen how for the "other" category variance is extremely high,

[9] https://www.spss.it/statistics-for-data-analysis

whereas for the “3 months” and “12 months” peaks occur at the expected times (respectively, 3 and 11-12 months).

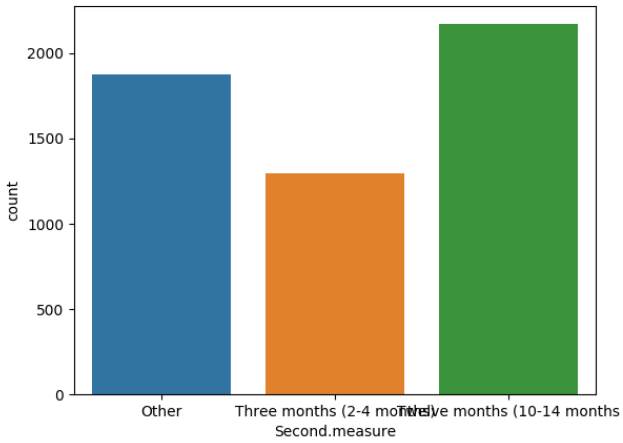


Figure 48 Most common follow-ups grouped as 3 months (orange), 12 months (green), other (blue).

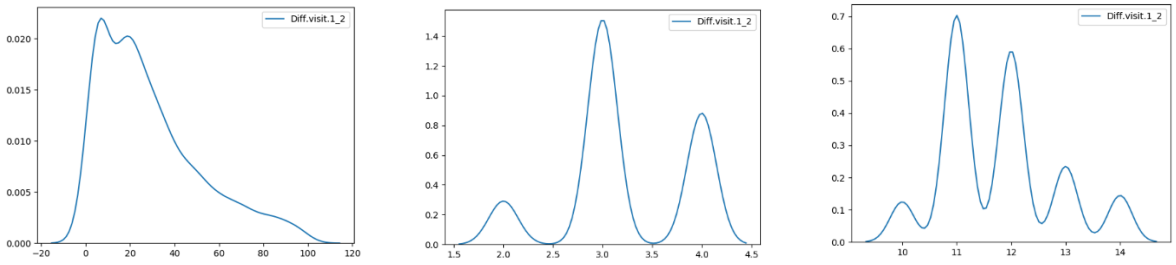


Figure 49 Distribution of difference, in months, between baseline assessment and second visit (first follow-up): categorised as 3 months follow-up (centre), 12 months (right-most), and other (left-most) in Figure Figure 48.

Then, the dataset has been restricted to samples (patients) having follow-ups at 3 and 12 months accordingly: 1299 patients for the former, 2360 for the latter. After such a filtering, missing data analysis was conducted, revealing missing values for above 83% of attributes for the 3rd follow-up, and worsening in case of further follow-ups (89% for 4th, 92% for 5th). Hence we decided to split the dataset in two, according to when the follow-up occurred: the former dataset storing all the patient with 2nd assessment at 3 months, comprising 1299 patients and roughly 160 attributes repeated a second time, the latter storing all the patient with 2nd assessment at 12 months, comprising 2360 patients and roughly 160 attributes repeated a second time.

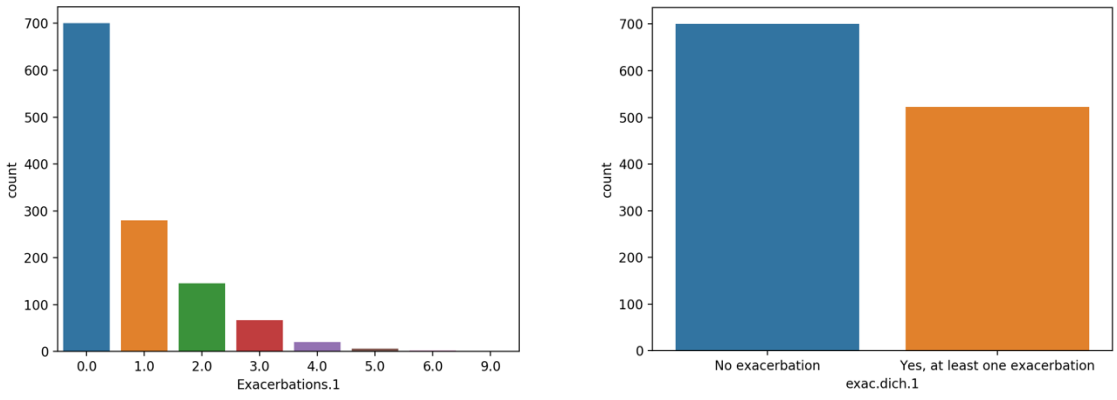
This restricted datasets are the ones representing the basis upon which different prediction models have been trained and compared, as described in Section 2.3. Before doing so, we performed several univariate, bi-variate, and multi-variate analysis on data attributes and their correlation, with the foremost goals of (1) devising out good candidate predictors amongst the several attributes at our disposal, and

(2) check data distribution for prediction targets, to decide, for instance, whether downsampling would be necessary to better represent imbalanced distributions of class labels.

A first analysis is then meant to find out how many missing values we have across columns, so that too sparse ones are dropped as they would not represent a good sample of the population. This analysis resulted in 138 “good” columns, with less than 25% of missing values, and 26 “bad” columns, with more than 75% of missing values, which has been discarded from further analysis steps.

Then we proceeded to check data distribution of both candidate predictors and prediction targets, as well as correlation between (1) prediction targets only, (2) predictors candidate only, and (3) both mixed up. Besides checking whether downsampling would be needed, this was done also to check whether obvious correlations existed between prediction targets (to avoid redundant predictions), or between prediction targets and predictors (to reduce dimensionality by selecting most discriminant attributes). Reporting the whole data analysis results is out of the scope of the deliverable, nevertheless some meaningful examples may be useful to better understand the complexities of the dataset.

For instance, 3 variables have been suggested by UMCG professionals as prediction targets they would like to have available from the Risk DSS (as described in Section 2.1): the number of exacerbations in the year next to assessment, the ACQ category, and the CCQ category. Their distribution of values is depicted in Figure 50: the number of exacerbations is highly imbalanced (top-left chart), but was already conveniently reshaped in the dataset as a dichotomous variable tracking whether the patient had no exacerbations or 1 or more, which is much more balanced instead (top-right chart). However, this way we may only make predictions about whether an exacerbation is likely or not, not about the actual number. Hence, we further reshaped the variable in two cases: none, one, or two or more exacerbations, and none, one, two, or three or more exacerbations, which are both unbalanced but in a way manageable by the learning algorithm. The ACQ category is instead naturally quite balanced, whereas the CCQ category has low sample size for “very unstable” patients: we may choose to drop them from the population, or to downsample a bit the “not entirely stable” category, which is the only one exhibiting a peak. We went for the latter option.



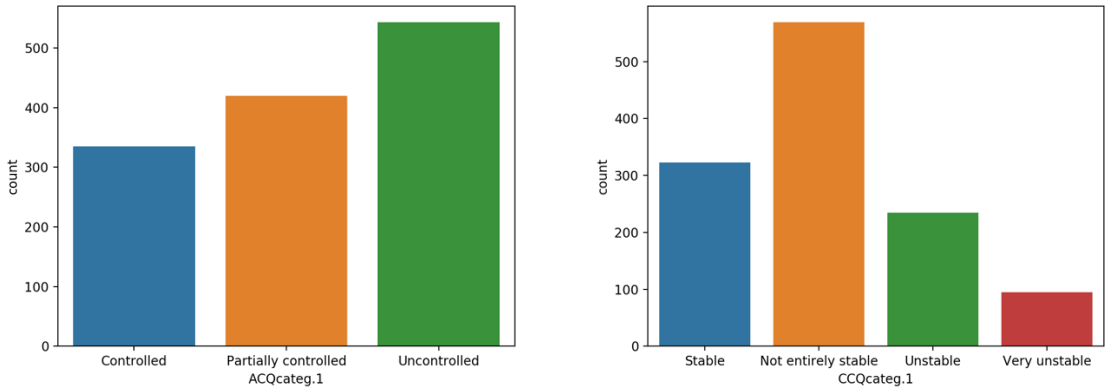


Figure 50 Values distribution for the number of exacerbations (top), ACQ questionnaire category (bottom-left), CCQ questionnaire category (bottom-right).

Amongst the candidate predictors, being them more than 100 attributes, the situation is extremely heterogeneous. For instance, whether treatment advice was given by the pulmonologist after the assessment is extremely imbalanced (Figure 51 left) whereas smoking history is quite balanced (Figure 51 right), hence the former attribute is better to be removed as further downsampling is not possible and it may skew predictions a lot due to low representation of “no” values, whereas the latter is ok to be exploited (but yet to see if meaningful and how much).

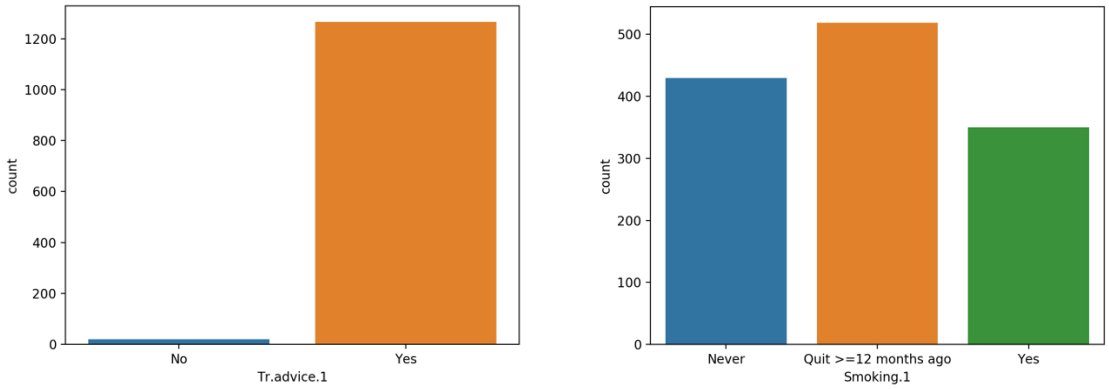


Figure 51 Example of imbalanced vs. balanced attribute.

Up to now we have seen mostly categorical data, but the same analysis can be done for numerical data, as in the case of age and Body Mass Index (BMI) shown in Figure 52: the former has a greater peak around 60 and a smaller one around 20, whereas the latter is more or less normally distributed. This kind of analysis is important as some machine learning models need normally distributed data to be effective. For instance, the LinearSVC and SVC with varying kernels approach we described In Section 2.3 both

need normally distributed data, hence required a pre-processing step (before actual model training) to scale data accordingly.

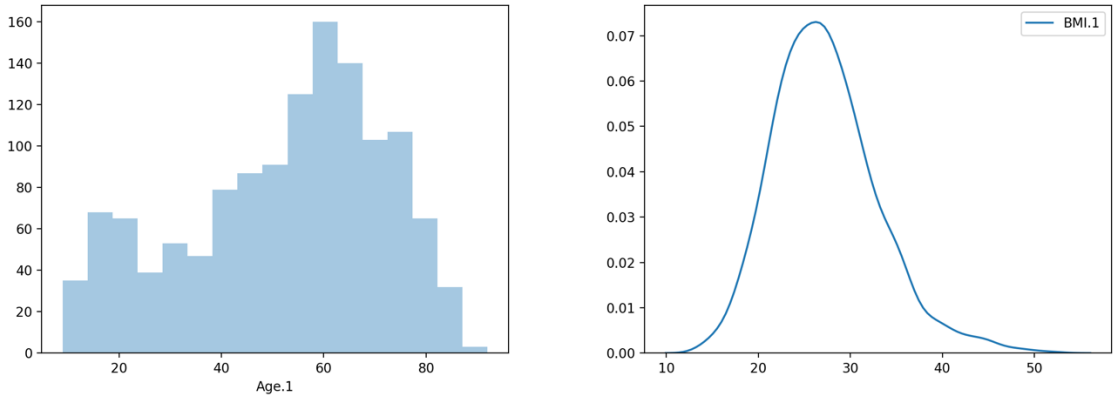


Figure 52 Age has peaks around 20 and 60 (left), whereas BMI is roughly normally distributed (right).

Without further lingering on univariate, showing examples of bi-variate and multi-variate analysis may help unveil further complexities of the dataset. We start with some meaningful overviews given by multi-variate analysis on prediction targets, then further detailed by specific bi-variate analysis when due.

Figure 54 for instance shows the average number of exacerbations for each pair CCQ category – ACQ category: highest values (high average number of exacerbations) correspond to uncontrolled ACQ, whereas CCQ seems to have a lesser influence as higher values span the whole “not entirely stable, “unstable”, and “very unstable” spectrum, which indicates that some (little) overlap between exacerbations and ACQ prediction targets exist, but none for exacerbations and CCQ. Since such an overlap is represented by a small sample size, as shown in Figure 53 in which no percentage is exceedingly high or low compared to others (the peak is 20), all the three prediction targets are kept as the goal of the prediction models developed.

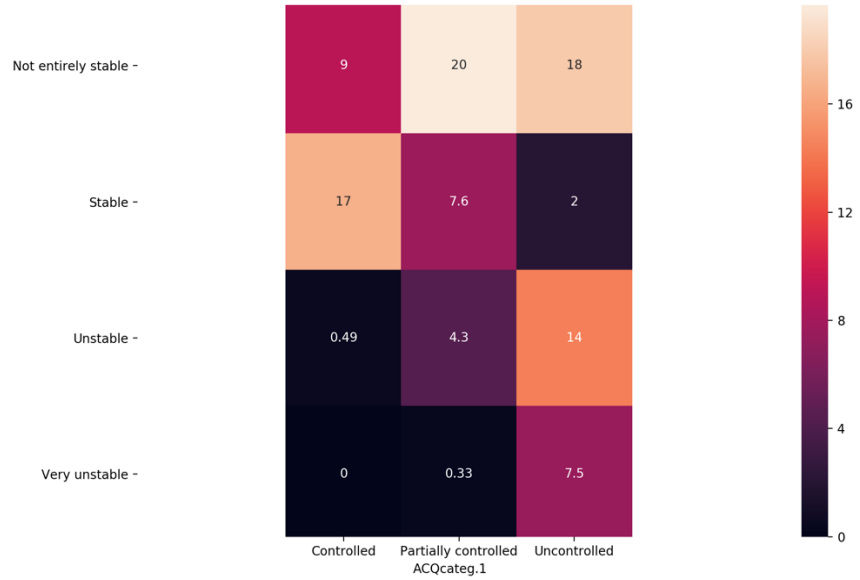


Figure 53 Percentage of patients with at least one exacerbation per ACQ-CCQ category combination.

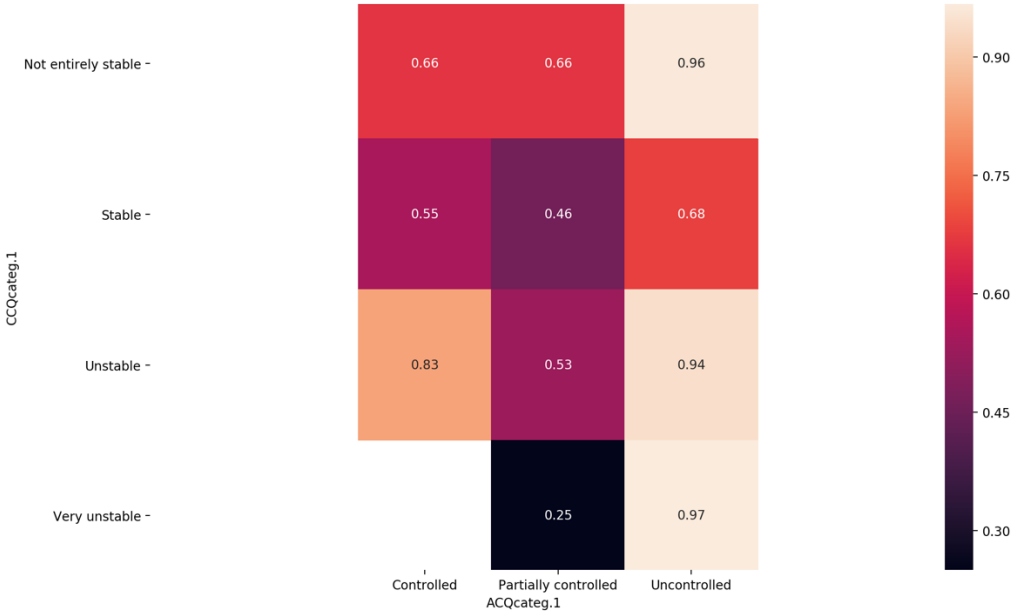


Figure 54 Average number of exacerbations per CCQ-ACQ category combination.

Nevertheless, the summary scores of such questionnaires, determining the category assigned to the patients, show high correlation as emphasised by the plot in Figure 55, which means that it is most likely to find in the dataset patients with either both ACQ and CCQ low, or both ACQ and CCQ high.

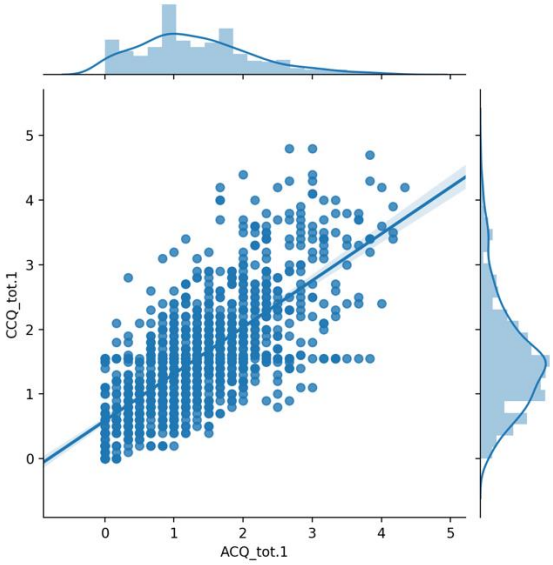


Figure 55 Correlation between the summary score of the ACQ questionnaire (x axis) and the summary score of the CCQ questionnaire (y axis).

Besides comparing prediction targets amongst themselves, also comparing candidate predictors to each other is meaningful to unveil obvious correlations which could be exploited to reduce dimensionality of the attributes used for training.

For instance, Figure 56 shows the heatmap built on the correlation between a few candidate predictors randomly sampled, which gives an overview of the heterogeneous situations commonly found in UMCG dataset: for instance, high correlation is found between FEV measured before and after usage of bronchodilator, as well as between FEV and FVC, whereas no meaningful correlation is found amongst smoke exposure and other variables. This indicates that the two variables measuring FEV before and after usage of bronchodilator may be reduced to either one of the two, to reduce the number of attributes used in training.

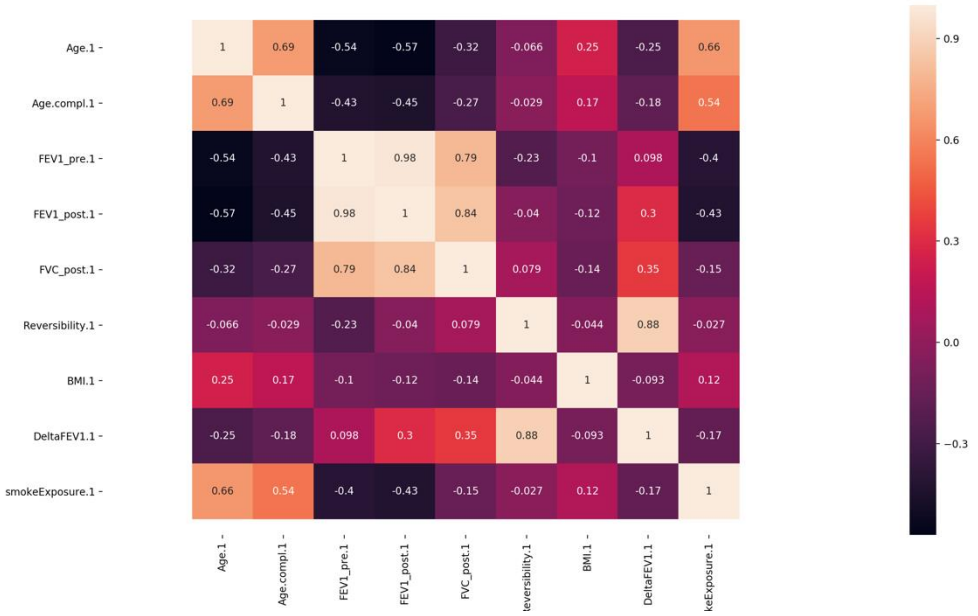


Figure 56 Correlation between a few random candidate predictors.

A similar analysis has been conducted systematically on the entire set of predictors, and all have been kept for training the prediction models described in Section 2.3 as no obvious “proxy” for the predicted variable has been found, and because computational power need by the learning task has been manageable across the different algorithm tested.

The exploratory analysis here briefly overviewed served the purpose of better informing the machine learning pipelines described in Section 2.3. For instance, as ACQ category and CCQ category are highly correlated in the dataset, as well as ACQ (CCQ) category and ACQ (CCQ) assessment score, when predicting one the other where removed from the train set: otherwise, the prediction models would seem almost perfect, but in fact would give no insight.