MSF-eCare
Modular Clinical Decision Support Networks for Data-Driven Diagnostic Predictions

Louis Bettens, Louis Gounot, Xavier Nal
iGH: intelligent Global Health, EPFL Lausanne, Switzerland

Abstract—**The development of artificial intelligence has opened up new possibilities in the field of medicine. It can be valuable in clinical decision support (CDSS), especially if resources are limited. CDSS can reduce diagnostic errors and improve clinical management by giving probabilistic guidance. This work applies machine learning techniques to a dataset of 300'000 consultations collected by Médecins Sans Frontières with the objective to derive data-driven diagnostic predictions.**

I. Introduction

MSF is an international humanitarian non-governmental organization providing access to medical care to vulnerable populations. In an effort to standardise medical consultations for pediatric outpatients, they developed a mobile application, eCARE that hosts a CDSS guiding clinicians through a consultation. Since 2016, over 300'000 consultations have been logged on the device across eight countries in African (Kenya, Tanzania, RCA, Niger, Nigeria, ...), capturing answers to the clinical questionnaire such as demographic information, clinical signs and symptoms as well as the diagnosis.

The goal of this project is to deploy the Modular Clinical Decision Support Network (mCDSN) developed previously by the iGH group at EPFL. the mCDSN encodes the answers at each stage of the questionnaire to create a continuously updating mathematical representation of the patient which can then be used to decode a diagnostic prediction at any point of the consultation. Additionally, we assess the dataset for its anonymity characteristics.

II. Objectives

- **OBJ1: Clean and explore the data**
- **OBJ2: Find the questionnaire structure:** The dataset suffers from heterogeneity and systematic missingness, where the questionnaire structure filters questions, according to initial triage responses, resulting in many unanswered questions. It is difficult to implement a predictive algorithm with systematic missing data, especially if the presence an value of one answer depends on a previous one. Thus, the first task is to find the structure of question tree to have an idea of of which question were asked to the patient.
- **OBJ3: Implement the modular decision support algorithm:** with the objective of providing a probability of the different disease at each step of the consultation, so after every feature.
- **OBJ4: Assess the anonymity of the cleaned dataset:** Provide a privacy report using.

III. Methods

All the code we used is available at https://github.com/epfl-iglobalhealth/CS433-2021-eCARE1.

A. *OBJ1: Data cleaning and exploration*

1) Data Exploration: Our data set is made up of 271 columns and 310 000 rows. We classified them by searching if a feature corresponds to a symptom or a disease. All features starting with *classify* are the disease we want to predict. We have 133 features which are symptoms and 91 which are diseases. An important part of the feature are binary values is the symptoms or the laboratory analyse was true or false. We find quantitative data too.

2) Data Cleaning: **Missing data.** While missingness in the dataset is an important feature with which to construct the tree logic, we discarded 45 columns with over 87% of missing data (the others columns are under 73%). In the code, the missing values in the features are very important as they allow us to know if a question was asked or not in our questionnaire. So we don't replace them. We just ignore them later in the code. In the `data_cleaning.py` file you can see that we had a lot of different values that are completely non valuable for us. Extremely sparse data columns (with just one unique value like 0) are also dropped due to being uninformative in predictive analysis.

Incorrect data points We look at all the strange value in the feature and replace them with the program's *not available* marker. We also put the columns in the float type for ease our manipulation. Some of the columns in the feature do not interesting use because there are not in relationship with the prediction we want to make. So we decided to not use the columns with id consultation, user code, old diagnostic. But we observed that these columns were not normalized. In most of the columns some there are some absurd values, such as words in a numerical column. We also observed hexadecimal numbers that we could not make sense of. The data set is still huge and very difficult to have good results with all those symptoms

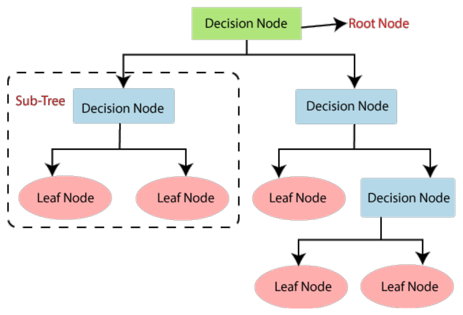

Fig. 1. Example of a binary tree

and diseases. Indeed, with the advice of a medical doctor, we decided to only focus on thete diseases : Pneumonia, anemia, bronchiolitis, severe disease, malnutrition, Upper respiratory tract infection, malaria.

In the main code, to decrease the number of symptoms, we analyse each columns : if a column has less than 0.1% of 0 or 1 then we don't take this symptom in account. This symptom would not be valuable as nobody has this symptom or everybody has it and in this case we can wonder if it's really a symptom. And finally, we arrived with a number of feature under at 70 columns and some time under 50 (depending of the country we choose).

B. *OBJ2: Finding the questionnaire structure*

The developed model done by Cecile is not specific to a data set and works on any type of questionnaires-like data. The key for a medical questionnaire is the pattern in the missingness. A decision tree is used to explain the type of pattern. Every node in a tree corresponds to a question asked by one of the clinician. The consultation path of a patient shows which feature was recorded and which was missing. In our project, we use a binary decision tree as symptoms are binary values (yes or no questions). By looking to 1, for a patient, the decision tree

can be understanding in the following way : starting at the root node, we move to the left or the right depending of the answer of the decision node(patient's answers to the questions). By doing this procedure recursively, we attain a leaf node corresponding to a final diagnosis.

C. *OBJ3: Implementing the mCDSN*

1) Neural Nets: To accomplish this task, we modified Cécile's code to fit our dataset ([1]). However, the majority of the time, the problems we met, were solve with a better understanding of the different functions she created. We will summarize here the model, but for more details we refer the reader to [1].

The main element of the model is a state vector which we update after every new question (feature in input). This state vector represent the patient. It takes in input the feature and gives in output the disease. More features our model get in input and more precise it becomes. The input are a neural network we call an encoder. For every feature we attached a neural network which we trained. The structure of the encoder is a neural network with one hidden layer fully connected. The encoders take in input the value of the associated columns and the previous state vector. The input get through a neural network. Then the output is send to the new state vector. For the disease we have the same neural structure without hidden layer. The decoder can not modify the state vector. This structure take in input the state vector and give a scalar in output. To have the estimated value of the disease we have to pass this value through an exponential.

D. *OBJ4: Privacy assessment*

1) Freeform fields examination: Since the data contains free form fields: columns that can contain arbitrary text, we decided to examine the contents of these columns to see if any identifying information might have been left there and not removed. We performed a simple word counting analysis of all free-form fields and manually assessed the results. This also serves the additional purpose of potentially receiving hidden features that could be useful for machine learning.

2) k-anonymity: k-anonymity is a property of a dataset that indicates how hard it would be for a malicious party to, given the dataset, match patients with their records in the dataset.[2] Methods exist to alter a dataset to increase its k-anonymity while reducing its usefulness for machine learning as little as possible. We use the R package sdcMicro[3] to measure and increase k-anonymity. Our goal is to ensure that the dataset satisfies k-anonymity with $k = 5$ before we perform machine learning.

IV. Results

A. *Implementing the mCDSN*

The training loss and the prediction accuracy are for different countries are display in the I, II and III. We separated the data frame into countries in order to have better analysis and reduce the calculation time. After we split the dataframe in two groups with respect to 80% and 20% of the data. The bigger group is used for the training and the other for testing our model.

We can see that our model is accurate most of the time. We observe a loss function under 5 for the three countries. However sometime the percentage of good prediction are quick lower compare to other illness. For example, `cough_urti` in the Nigeria has just a prediction accuracy at 62.4% compared to other above 94%. It could be explained by the fact that the model has not enough data for this particular target to provide better prediction. Indeed, we just have information for 3935 patients from Nigeria. It is much less than the Tanzania where we have 163 835 patients. Another observation we made is the absence of the illness `anemia_severe` for the Nigeria and Tanzania. The reason is that for this countries, this targets is

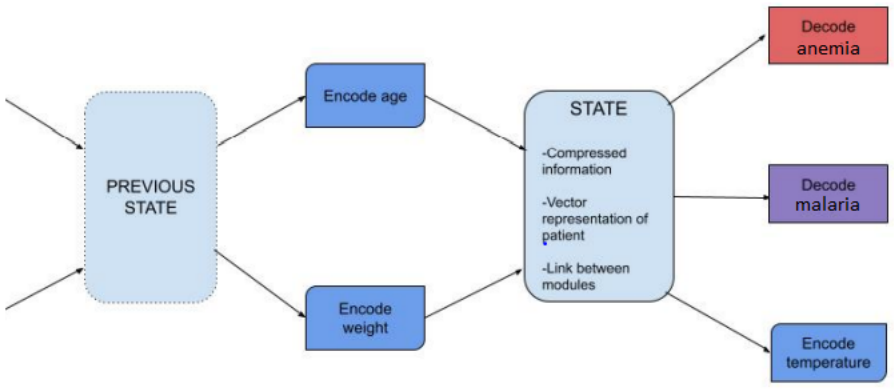

Fig. 2. Model, from [1]

TABLE I
DISEASE PREDICTION ACCURACIES FOR NIGER

DISEASE	PREDICTION ACCURACY
ANEMIA_NON_SEVERE	99.8%
ANEMIA_SEVERE	99.9%
COUGH_BRONCHIOLITIS	96.9%
COUGH_PNEUMONIA	95%
COUGH_URTI	88%
MALARIA_SEVERE	99%
MALARIA_SIMPLE_NEW	98%
VERY_SEVERE_DISEASE_MAL	99%

epoch : 150 : disease_loss 4.71087
epoch : 150 : disease_loss_valid 2.4935

TABLE II
DISEASE PREDICTION ACCURACIES FOR NIGERIA

DISEASE	PREDICTION ACCURACY
ANEMIA_NON_SEVERE	99.49%
COUGH_BRONCHIOLITIS	98.5%
COUGH_PNEUMONIA	94.7%
COUGH_URTI	63.4%
MALARIA_SEVERE	99.49%
MALARIA_SIMPLE_NEW	97.58%
VERY_SEVERE_DISEASE_MAL	98.34%

epoch : 150 : disease_loss 3.226
epoch : 150 : disease_loss_valid 3.869

unbalanced, and this columns is remove during the cleaning phase. You can find in 3 an example how the model evolves with the information of the encoder on the state vector. We can see that our system needs sufficient features to find the true disease.

B. *Privacy assessment*

1) Freeform fields examination: We found and removed a minor incident of a health worker intentionally inserting their own contact data. To MSF's credit, we found no breach of patient privacy.

We found that for the vast majority of consultations, there was no free form data at all. No word had more that 1'000 occurrences. Thus, we decided not to extract any features from the free-form columns. We also dropped them entirely as a precaution.

We humbly suggest that, if the same or similar datasets need to be disseminated in the future, the same procedure should be applied at some point since free form fields are by nature prone to containing sensitive data.

2) k-anonymity: The data, once cleaned, already has the desired $k = 5$ k-anonymity for more than 99% of records. A simple local suppression procedure contained in the sdcMicro toolbox brings it all the way to 100% by removing less than 3'000 values, the vast majority of which are related to the location of the health facility where the patient was seen. Overall, we believe that this process made the data safer and hope it can be part of the data protection strategy for MSF and iGH later on.

TABLE III
DISEASE PREDICTION ACCURACIES FOR TANZANIA

DISEASE	PREDICTION ACCURACY
ANEMIA_NON_SEVERE	99.8%
COUGH_BRONCHIOLITIS	99.3%
COUGH_PNEUMONIA	96.6%
COUGH_URTI	86.6%
MALARIA_SEVERE	99.8%
MALARIA_SIMPLE_NEW	96.8%
VERY_SEVERE_DISEASE_MAL	99.7%

epoch : 150 : disease_loss 1.035
epoch : 150 : disease_loss_valid 1.086

V. CONCLUSION

To conclude, we can say that we had a lot of difficulties to deal with the given data set which was very complicated. We

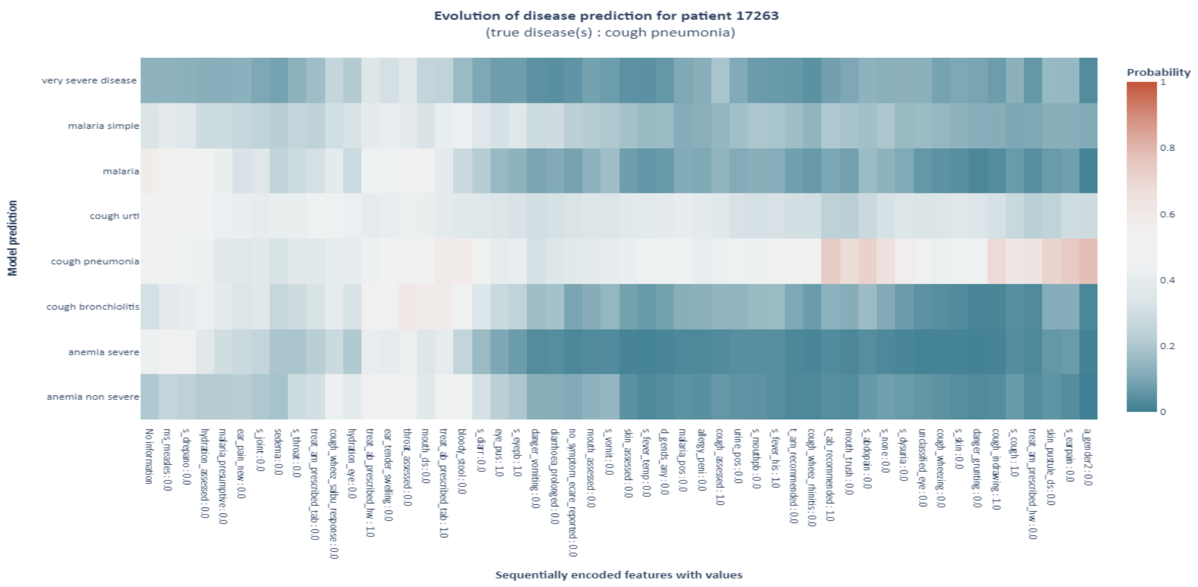

Fig. 3. Patient with cough pneumonia

had to exclude a lot of information in order to run the code properly and in a reasonable time. The final results are enough precise to help the clinicians with their diagnostic despite the fact that our algorithm may need a lot more information to predict a disease than the clinician. An improvement would be to choose the optimal features for a disease. By doing this, the amount of features needed to determine a disease would decrease and the algorithm more efficient. This work is subject to a more precise data exploration that would take a lot of time in order to determine the most relevant features for a target.

As far as data protection is concerned, we were able to gain insights as to what risks to patient privacy were posed by disclosure of the dataset, and to use tools to alleviate some of these risks. There is no one-step solution to these challenges, but anonymization can be a tool to build a robust data protection strategy.

We liked a lot to work on this project with MSF, it was a very motivating project for us, especially for our first semester of Machine Learning. We really enjoy working on a real dataset despite the complexity and it was very motivating to work on a project that could maybe help MSF to provide better clinical management.

Acknowledgements

The authors thank the iGH EPFL lab for giving us this opportunity to work on this project. Special thanks to Mary-Anne Hartley who guides us during the project. Big thanks to Cécile Trottet for her help during all the time of the project to understand her neural model. And thank as well as Praneeth Karimireddy for his advice regarding privacy.

References

[1] C. Trottet, "Modular clinical decision support networks," *Master thesis report from the iGH lab*, 2021.

[2] P. Samarati and L. Sweeney, "Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression," 1998.

[3] M. Templ, A. Kowarik, and B. Meindl, "Statistical disclosure control for micro-data using the R package sdcMicro," *Journal of Statistical Software*, vol. 67, no. 4, pp. 1–36, 2015.