Intermediate Report : Risk prevention on a web service using machine learning, with a trace validation

Guerre Berthelot Quentin

UGA University, 621 Avenue Centrale, 38400 Saint-Martin-d'Hères, France &
Kobe University, CS27, Kobe university, Japan

Abstract. The main purpose of this internship is to research about the risks (of all kind) that exist in a predictive system using Machine Learning prediction, and how it could be dealt with thanks to a validation using traces.

Keywords: Machine Learning · Risks· Validation · traces.

1 Introduction

During our internship in Kobe university we were entrusted to develop an application using a prediction based on a Machine learning model. The validation of a system that includes some learning feature is difficult, the final usage of the application is not easily predictable since it varies with respect to the environment [4],and it lead to appearance of risk. Since then we are not capable of defining precise characteristic, but we can imagine some general rules that could be define and check. Firstly, I am going to introduce you the application I have created thanks to a data set sample used in Microsoft Azure Machine Learning Studio[1]. Secondly, I will talk about the risks, that exist with this kind of structure. Then, I'll explain how I envisage to use the validation with traces using the language PartTrap developed in the laboratory LIG in Grenoble, to monitor and detect it. I'll conclude with what I have learned and what could be done in the next weeks.

2 Creating a predictive Model base on Machine Learning prediction.

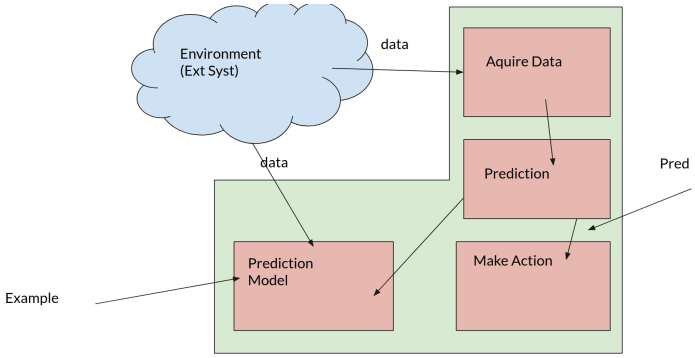


Fig. 1. System model

2.1 The Data set

To create a ML (Machine Learning) predictive model, it is necessary to have a database with enough data for the results to be the most accurate possible. In the following, I will illustrate the approach on :

Breast Cancer Data case study : It was possible to collect a sample about breast cyst diagnosis on UCI website [6]. A breast cyst could turn into a cancer, we call it "Malignant", if not then it is "Benign".

The data set is composed of 32 columns, which contains : The Id of the patient, the diagnosis, and ten real-valued features computed on three cell nucleus(radius, texture, perimeter, area, smoothness, compactness, concavity, concave point, symmetry, fractal dimension).

Using this data, we create a model taking in input the three cell's nucleus data and determine a diagnosis (B or M).

2.2 Machine Learning model with Microsoft Azure

Using Microsoft Azure Machine Learning Studio [1], it was possible to create a model using a two class boosted decision tree [5]. This algorithm was used because there was only two answers expected (malignant or benign). First I took the raw data and exclude the useless information such as the patient ID. Next, the data were splitted to create a testing sample to verified the accuracy of my model (70% for the model). Finally I applied the Model on the sample and test it with the 30% remaining.

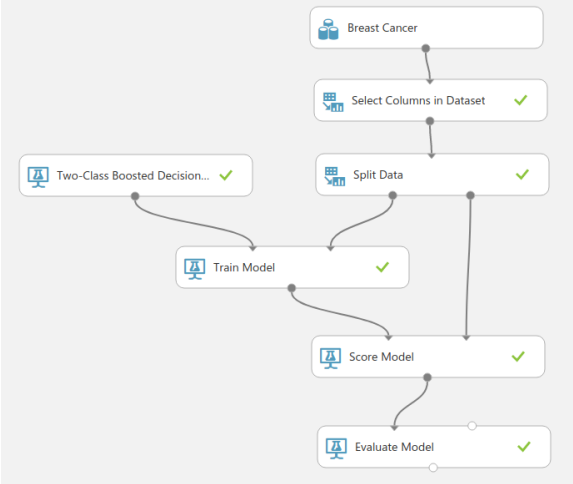


Fig. 2. Machine Learning Model

The result: The quality of the model is evaluated through the accuracy, which consist to measures the goodness of a classification model as the proportion of true results to total cases. It was quite satisfying with an accuracy of 0.982 but it might be because the sample wasn't big enough. It could be good in the future to create a new collect of data with a bigger sample.

True Positive	False Negative	Accuracy	Precision
44	1	0.982	0.978
False Positive	**True Negative**	**Recall**	**F1 Score**
1	68	0.978	0.978
Positive Label	**Negative Label**		
1	0		

Fig. 3. Results of the accuracy of the model

2.3 The App

Azure Machine Learning Studio [1] offer the possibility to deploy a web service that allow a connection to it, threw a programming language as Python or Java. I used python and have developed a small application that allows anybody to enter there data (cell's nucleus ones) and receive a prediction of the diagnosis (malignant or benign).

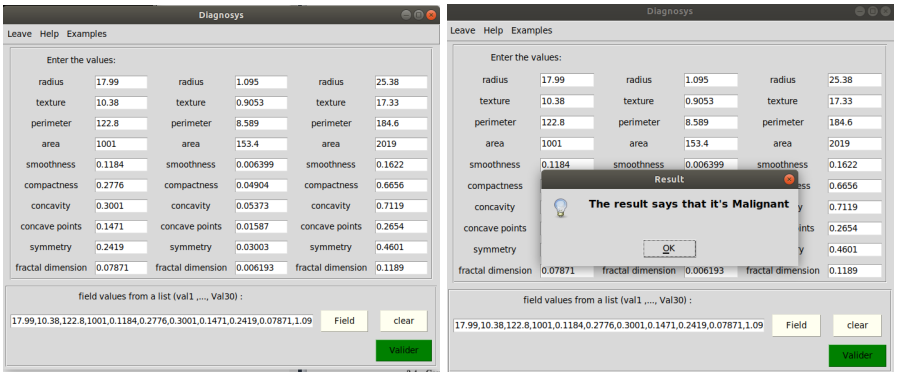


Fig. 4. App

3 The Risks

The prediction based on machine learning structure are still not totally operational, and there is a lot of different risk, using it, the main purpose of this internship is to work on it and try to find a way to deal with it. In addition to risks, there is also some threats brought by adversaries. In the following, we will study the risks and threats :

Data compatibility : The input data should be compatible with the one used in model.

For example : if the model only reads positive float then the client should not send a negative float or a string. Similarly, if it is waiting for values, it can't receive only 9 on then.

Action performed: The diagnosis performed should be tolerance to the environment. This means that with time the values could change, with the evolution of the mankind, the global warming or anything else.

Confidence of prediction: In addition to the result, the model should give the probability of confidence. If the confidence is low, the system can delegate the decision to the user or more precisely to a medic in this case.

security of the data: The security and privacy issues must be addressed.

Substitution & Tampering : The user should be sure that the information he receives are not altered, read, or stored by an other person. This is something we are going to prevent or at least detect thanks to traces.

For example : in the case of the "man in the middle" attack.

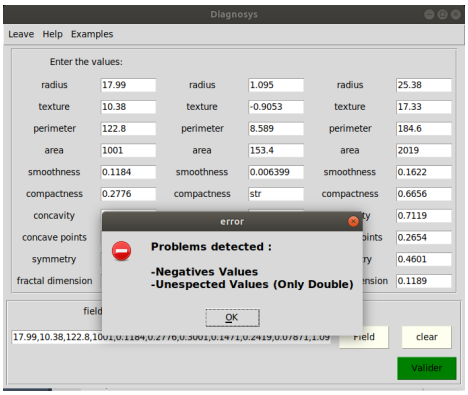
The data, model and algorithms should be protected too.

Information leakage Like the possibility of tampering data, it is important to protect the information from leaking.

For example : a woman don't want anyone else to know her name and the fact the the model predict that she could have a cancer.

3.1 Data compatibility

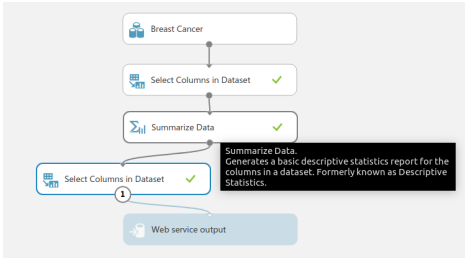
The data entered as input should be checked, and the application should warn the user about the abnormality and ask him to adjust it. In this application, what is expected are : Float, positive value, 30 inputs value (3*10 values).



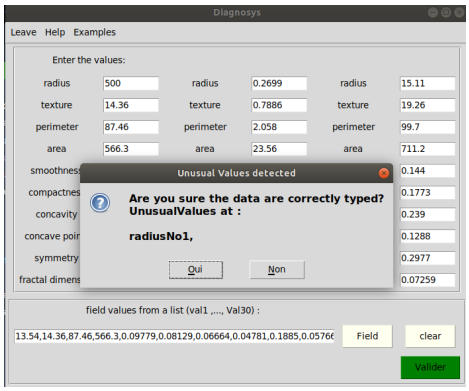
The solution envisaged: What was done in this part is quite simple : before sending the values on the web service, all of those are checked. If it passes all the test the values are sent. If not, it is asked to the user to re-enter the value.

3.2 Action performed

As told before, the basic values can change with time, this should be really monitored, because the accuracy of the data could drop and endangered the women concerned.



The solution envisaged (Need more verification) : For example : if the app said that the cyst is malignant but is not that is not really dangerous, it would only scared the woman for nothing. However in the other case it could really lead to the death of the women if she thinks she is not seek, and actually has a Breast Cancer and nothing is done to treat it on time. Thanks to Microsoft Azure[1] using the same sample data I used to create the model, I created a Summarize of the data and for each kind of data I take the mean and the standard deviation and I checked that each data aren't unusual. If the data are unusual (out of bound [mean-standard deviation, mean+standard deviation]) there is two possibilities, one of then is just the user who tip it wrong, the other is a value never seen before in the data during the creation of the model.



That is why the app ask the user once if it was wrongly typed. If it was then the user can change it? If not the user is warned that the diagnosis could not be really accurate and it would be good to check it with a medic.

Secondly (Method unchecked yet) : The data should be record regularly with their date and the model should be updated on a regular period excluding the oldest data. Like that if (for example) the humankind evolve slowly the model could be updated and adapt to it.

3.3 Confidence of prediction

The web service of Azure Machine Learning Studio[1] give an accuracy of the result obtained. It is easy to warn the user that the accuracy is under 0.95(supposed value should be checked with an actual expert), and advise him that he should probably check it more precisely with a doctor. But if he doesn't, it is really important to warn him that if the diagnosis is not right it is under his responsibility. Furthermore, even if the App give a diagnosis, it is important to go check it with the doctor periodically, but on a period larger than the actual one.

3.4 Security of the data

It is really important to respect the security and the privacy of the user. Any information about the user should remain secret. For the moment the App sends the information to Azure, and azure answers with a diagnosis. Normally, Azure studio deals with the encryption. A key is given when the web service is created and you can't decrypt the information without it.

But it would be interesting to create a subscription system with a server which protect more efficiently the user.

4 Validation and monitoring

Now that we have targeted the risks we want to prevent or at least detect it when it occur, we are going to use a tool developed in the Laboratory LIG [3] in Grenoble, called PartTrap [2] to monitor traces of software.

PartTrap: It is an expressive language that allows you to express properties on parametric event traces. It was designed to ease the understanding and writing of properties by software engineers without background in formal methods [2]. The language takes a json file which is a traces of a software execution. Each traces has an id, a time ,and other attributes that you can decide to use. Once the json file is generated, we need to expressed the properties the software need to respect and a java program will be generated. Finally, when the program generated is lunch, it will return an evaluation saying if the rules defined in PartTrap are respected or not.

How to use PartTrap to check the reliability of the diagnosis received: Has explained before, PartTrap is used to define some properties that need to be checked using traces of a software. Here we got two part of the software, the first is the Microsoft Azure[1] part, and the other is the software I developed in Kobe. One of the threats we introduce are the "Substitution & Tampering", we want to be sure the data were not altered. That means that me need to create a trace on the web service side and on the software side. A Second one is the accuracy of diagnosis, which is really important knowing that someone health is on the line, we want then to check that the accuracy is at least higher than 0.95.
Traces :

on server side

$$\{"id" : "DiagServer","idDiag" : y,"time" : x,"answer" : "Benign","acc" : z\}$$

on software side

$$\{"id" : "DiagReceived","idDiag" : y,"time" : x,"answer" : "Malignant","acc" : z\}$$

From those traces many rules can be created, and other parameters can be added depending the use we want from it. Once the traces acquire, it is needed to define the rules, we want to check.
Properties :

There is two rules that could be created with this kind of trace :

```
1 ValidMessageReceived: after each DiagServer t , occurrence_of  DiagReceived r where t.answer==r.answer && t.idDiag==r.idDiag;
2 ValidTimeReceived: after each DiagServer t, occurrence_of  DiagReceived r where t.time<=r.time && t.idDiag==r.idDiag;
3 ValidAccuracy: occurrence_of  DiagReceived r where r.accur>=0.95;
4
5 ValidAll : ValidMessageReceived and ValidTimeReceived and ValidAccuracy;
```

Fig. 5. PartTrap

The first one is about the answer received and if it is the same one sent. It's the first line of the figure 5.

The second is the difference between the time it was sent and the one it was received. If it was received before it was even sent, it could mean that someone else sent it. It's the second line of the figure 5.

The third is the accuracy checking, which verify that it is high enough. It's the third line of the figure 5.

The last line has for aim to verify all the rules at the same time.

5 Conclusion

Firstly we develop an Application using a Machine Learning model. Then we determined the risks, which exist with this kind of structure, and start to think how we would detect or prevent then. One of the main problem of software using machine prediction is to detect the unusual behavior. Thank to PartTrap [2] developed in LIG [3] in Grenoble, we start to think of a way to express properties on trace of our software to monitor it's behavior. It is just the beginning and for the moment nothing has been proved. But we can supposed that the usage of a validation using traces on a software based on IA (and more specifically here in Machine Learning) could be really useful. Especially for people who's not familiar with formal validation methods to checked the validity of the answer, action and behavior of the system.

What to do next : It is now needed to verify these hypothesis, implementing some simulations and creating the traces needed to experiment it.

6 Acknowledgment

I should thanks the Region of Auvergne-Rhône-Alpes for helping buying my flight ticket and support me.

Thank you to Masahide Nakamura our teacher in charge in Kobe University.

Thank you to Lydie du Bousquet and Philippe Lalanda my professor in UGA university, who gave me this wonderful opportunity.

References

[1] *Azure Studio*. URL: https://studio.azureml.net/.

[2] Ansem Ben Cheikh et al. "An Environment for the ParTraP Trace Property Language (Tool Demonstration)". In: *Runtime Verification - 18th International Conference, RV 2018, Limassol, Cyprus, November 10-13, 2018, Proceedings*. 2018, pp. 437–446. DOI: 10.1007/978-3-030-03769-7_26. URL: https://doi.org/10.1007/978-3-030-03769-7%5C_26.

[3] *Laboratory of computer sciences in Grenoble LIG*. URL: https://www.liglab.fr/.

[4] Masahide Nakamura and Lydie du Bousquet. "Improving Testability of Software Systems that Include a Learning Feature". In:

[5] *Two class boosted decision tree*. URL: https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/two-class-boosted-decision-tree.

[6] *UCI Homepage*. URL: https://archive.ics.uci.edu/ml/datasets.php.