Report IN104:Explainability in Artificial Intelligence

Reynaud Nils
Empereur Orane*

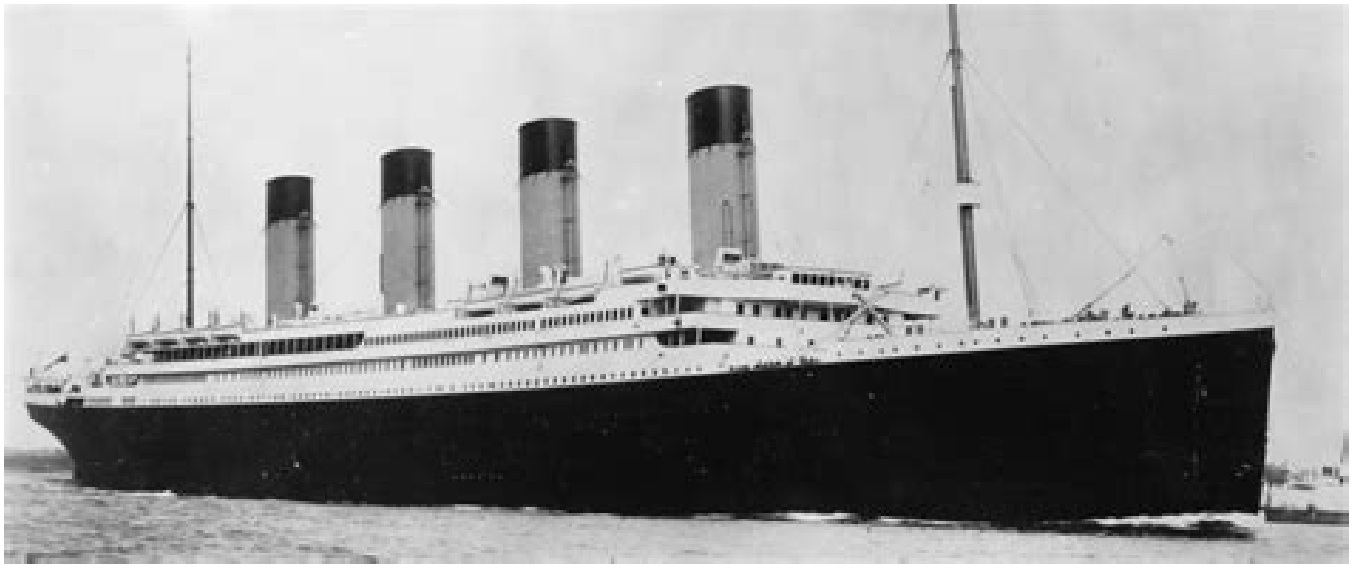


Figure 1. Titanic.

Abstract

The purpose of this project is to introduce machine learning and get acquainted with some methods in python langage (SHAP, LIME, PDP, CAM). Furthermore the goal is also to try to use already done works on this method and know how to readjust there to use its with other datasets. All the datatsets used in this project was found on Kaggle. The code of method presented in this project are available on kaggle with the adress : https://github.com/ReynaudNils/IN104$_{O}rane_{E}mpereur_{N}ils_{R}eynaud$

ACM Reference Format:
Reynaud Nils and Empereur Orane. . Report IN104:Explainability in Artificial Intelligence. In ,. ACM, New York, NY, USA, 6 pages.

1 Tabular Dataset:Titanic Survivors

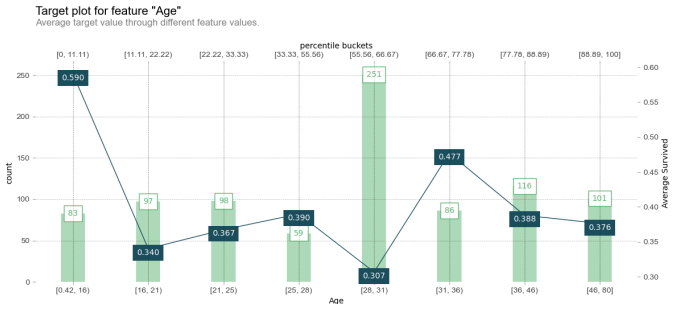
The data-set used for this method is the data-set of the titanic survivor. The goal is to predict the most important factors in the chance of survival.

1.1 PDP-Partial Dependence Plots

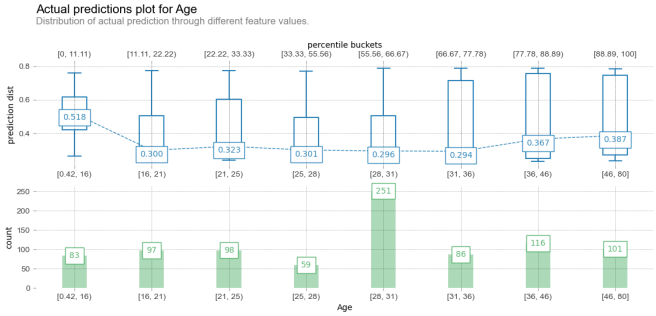
The first method we used to understand the various machine learning models is the PDP-one, Partial Dependence Plots, which shows the marginal effect one or two features have on the predicted outcome of the model used. After exploring several PDP codes on Kaggle, there weren't many difficulties in coding. We used it to explain two different machine learning models, xgboost and RandomForest. We'll mainly focus on the first one here. Firstly, this method allowed us to study the importance of each feature for the model: it gives us the percentage of survivors for every value of the feature according to the real data, the percentage of survivors for every value of the feature according to the prediction of the model, and finally a partial dependance plot that shows whether the relationship between the feature we're studying and the variable target, here the survival or not of each passenger, is linear, monotonic or more complex in the prediction. Let's first take a look at what we got for the feature Age, firstly the percentage of survivors for

*Both authors contributed equally to this project.

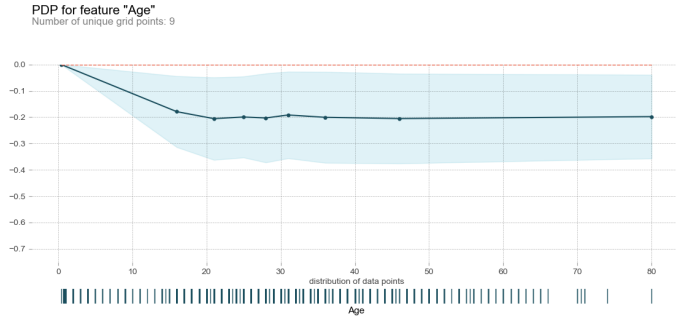
every age category according to the real data:



Then this same percentage but according to the prediction:

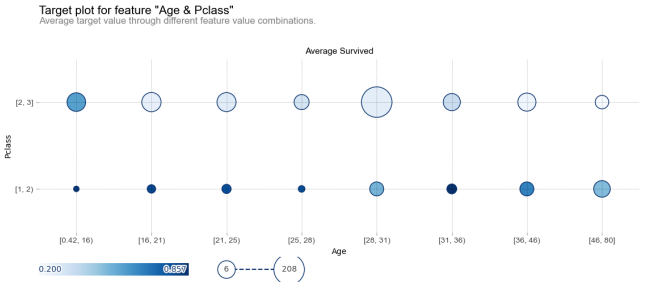


Finally, we get the interaction between the Age of the passenger and his survival, which is linear between 0 and 20 and then between 20 and 80.

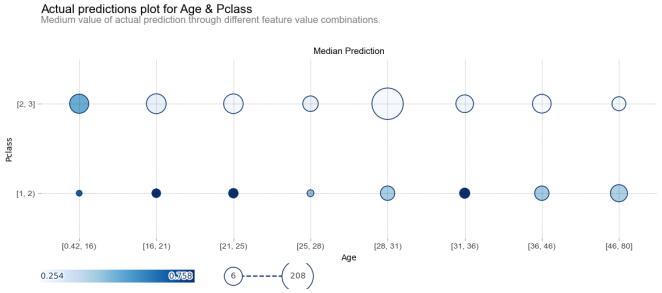


We did the same for other features such as Passenger class, Gender. As we can see, these interpretations are marginal, considering only one feature at a time. PDPs for two features are more relevant, as they show the interactions among the two features. We studied multiple combinations of two features, starting with Age and Passenger class: we were able to obtain the pourcentage of survivors based on these two features

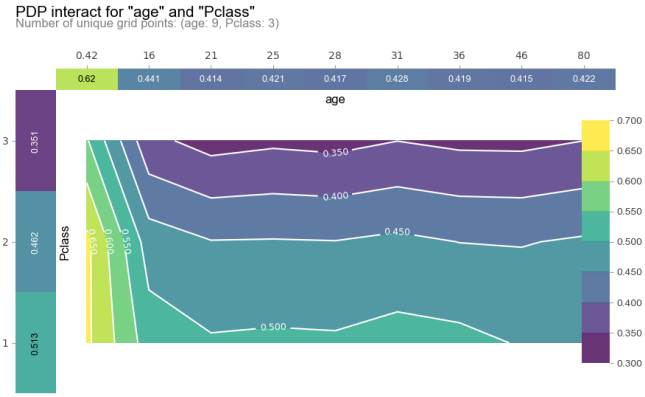
according to the real data.



then according to the prediction:



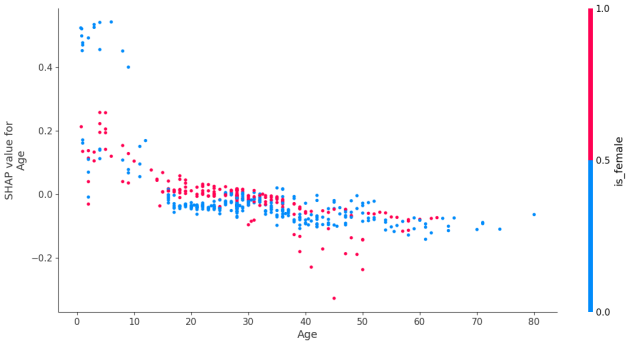
And finally the dependence of the target variable survival on joint values of Age and Passenger class



1.2 ICE

Individual conditional expectation, or ICE, is a method similar to PDP but a bit improved. It also shows the dependance between the target function, here if you survived or not, and an input feature of interest. But while PDP shows the average effect of the input feature, an ICE plot visualizes the dependence of the prediction on a feature for each sample separately with one line for each sample. We weren't able to make the code work because when entering the line display = $\text{plot}_{p}artial_{d}ependence(titanic_{m}odel, titanic_{d}ata; titanic_{f}eatures, k$ $\text{"both"}), thecomputerrespondedthatplot_{p}artial_{d}ependence()gotanu$

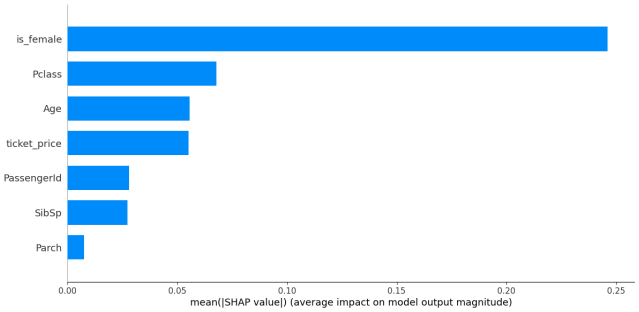
The problem is that we need to update the version of scikit-learn to 0.24.1, however this wasn't working with the usual lines conda update conda followed by conda install scikit-learn=0.24.1. Therefore the following graph that illustrates this method was taken online.



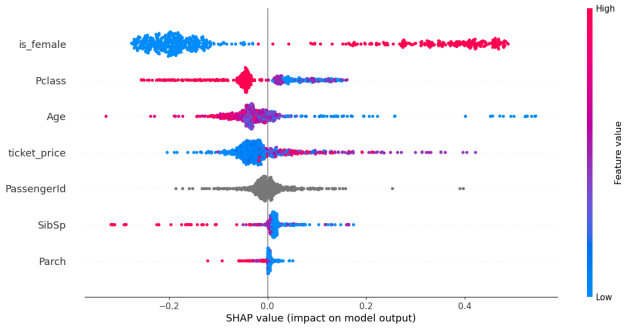
).png

1.3 SHAP

Let's now focus on the second explainability method we explored in order to understand the predictions made by the model, SHAP, or SHapley Additive ExPlanations.It is an approach based on game theory. This time the model whose predictions we will try to understand is RandomForest. The first plot we obtain shows the SHAP feature importance measured as the mean absolute Shapley values:



However, this plot gives no information beyond the importance of each feature. We need a more informative plot. The next one combines feature importance with feature effect. It ranks again the features by their importance, colors show if the feature is associated with a high or low prediction, and the x-axis if the impact is positive or negative.



Then this method gave us another plot, the SHAP dependence plot that shows the marginal effect one or two features have on the predicted outcome of our machine learning model.

The SHAP method even enables us to compare various machine learning models, RandomForest, xgb, but also MLP or LogisticRegression. It tells us the most important features for each model, which allows us to calculate a weighted SHAP Value, that takes into consideration everyone of these model. Moreover, it creates a cross-model feature importance graph. We can now objectively see the most important features. We faced huge difficulties coding this part and weren't able to make it work, the main problem being that it took several hours for the computer to run it.

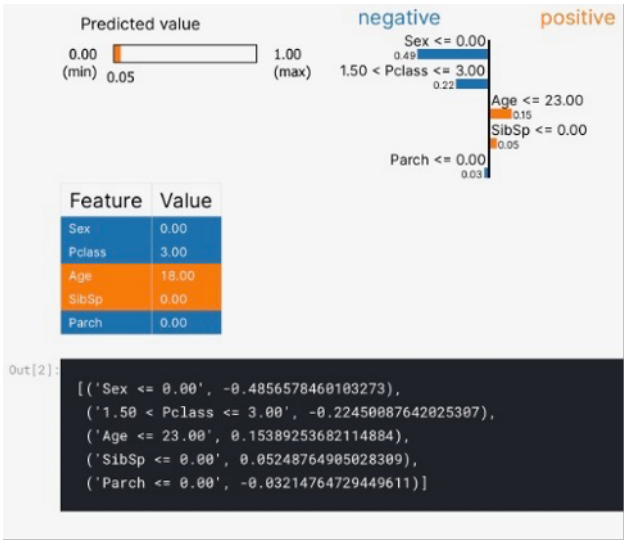
1.4 LIME

LIME (Local Interpretable Model-agnostic) is a method explaining the prediction of any classifier. The model predicts that only 0.312 percent of passenger survived, that in fact a result close of reality, so the method seem to be correct.

```
Intercept 0.8464555752865954
Prediction_local [0.31272063]
Right: 0.04913294499707087
```

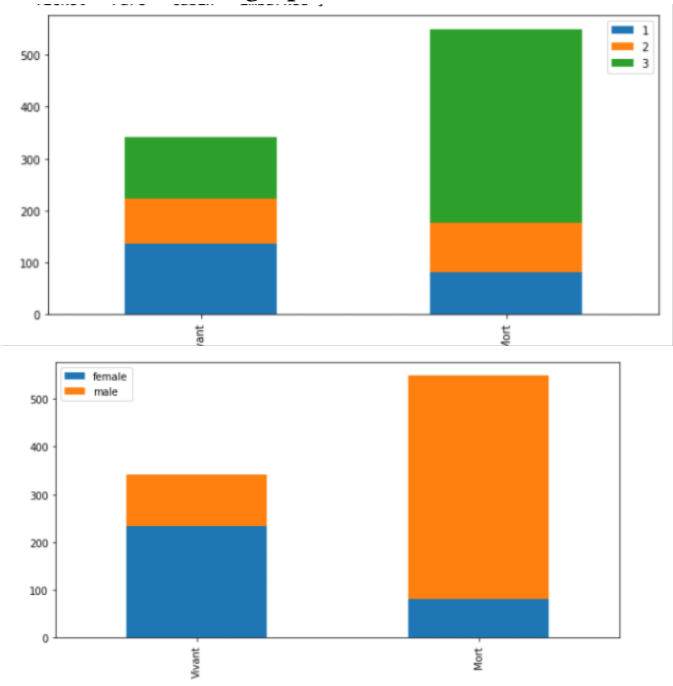
result give by LIME method

This paragraph will confirm that the model seem to be correct. LIME method show that the most important factor for survival is your sex, after it is the class of passenger and then the age. To obtain which features are the most important according to the model, LIME uses a slightly different input in which the data for one chosen feature are changed, and then observe the effect on the prediction outcome. In blue it is the factors which have a negative impact on the prediction and in orange a positive one. The table shows the most common values taken by the feature. For the sex 0 corresponds to a male person.



result give by LIME method

The image side of Out[2] was just a way to have more precise results if it is necessary. But even if our model is optimistic it corresponds to our expectation. We compare data (different features and if passenger survived or not) of passenger thanks to graph-sticks.



stick-graph obtained with data of titanic survivor

On this graph we can see that indeed sex and class of passenger are very important factors in survival.

1.5 Conclusion on the Tabular Dataset

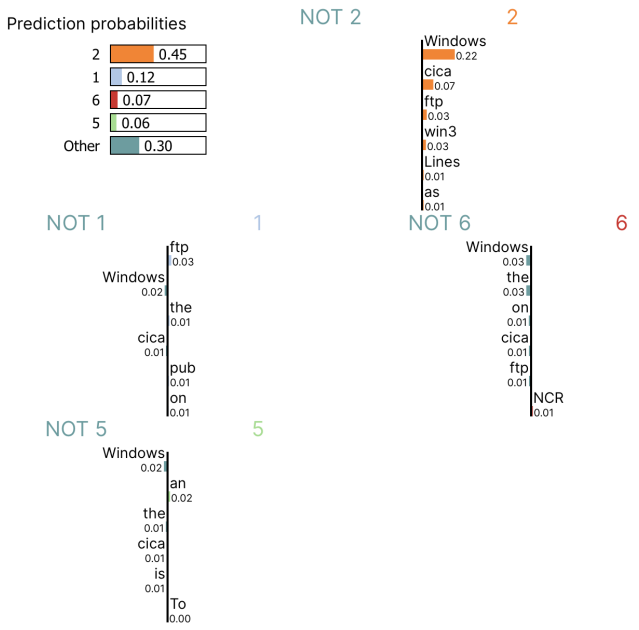
Time for a quick recap on this tabular dataset and the 4 different methods we saw to explain machine learning models. PDP is the simplest one, it shows the effect one chosen feature has on the prediction as well as the interaction between several features. ICE is the slightly more elaborate version. SHAP is much more useful, as it ranks for the given model every feature by importance, and can do so for every machine learning model. LIME offers a completely new point of view,as it shows how the prediction varies when you change the input of one the features; in other words, it describes the effect every feature has on the prediction.

2 Text Dataset:20 NewsGroups

We did not found text data-set about titanic survivor on Kaggle, that is why in order to test LIME method with text we chose to use another one. The data-set that we choose is 20 news groups, a famous one. The purpose is to use LIME method in order to know if the text is more a catholic opinion or a atheist one. This data-set is composed of more than 20 000 texts.
Example of result with the document number ten :

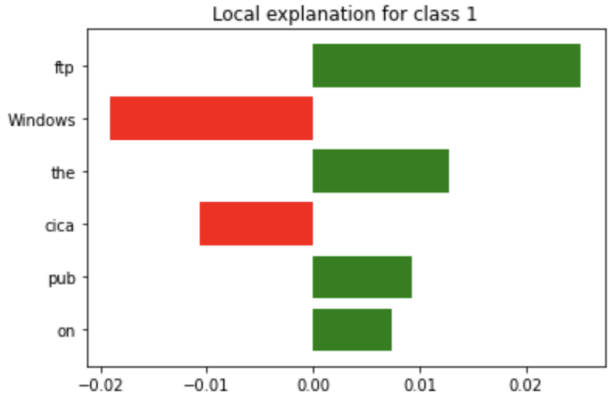
```
Document id: 10
Probability(atheism) = 0.002
Probability(christian) = 0.118
Probability(graphics) = 0.448
True class: graph
```

result gibe LIME method
There is also a graph that shows with more details the result obtained with this method (global result).



result LIME method, global

With the text number ten the prediction are indicated that the higher probability is that this text is atheist (orange). In blue it is the prediction about catholic.
There is also a local explanation about why the atheist class was chosen.



result lime method, local

This graph allows us to know the words of the text that impact the AI's decision to choose the class of the text, between catholic and atheist. In that way we can start to understand the prediction we got, why we got it.

2.1 Comparison with other method

We tried desperately to code several other methods for this text dataset that the teacher gave us, but we weren't able to make them work. For instance we tried Pyss3, but we couldn't download it on anaconda, when coding conda install Pyss3 it could not find the package. Therefore in this paragraph we will compare different methods completely already done and available on github at this different address :

https://github.com/robinvanschaik/interpret-flair

https://github.com/cdpierse/transformers-interpret

https://github.com/UKPLab/sentence-transformers

https://github.com/sergioburdisso/pyss3

The first one, interpret-flair, uses the LayerIntegratedGradients method in order to attribute score to the world, it can be an effective way to create a summary. The second one, transformers-interpret, uses the transformer package, it allows us to attribute a class to the word in a sentence : positive, negative or neutral. But with a larger vision it allows a support for NER models or Multiple Choice Model. Sentence-transformers like indicate this name use Sentence-tranformers method. This method also allows to realise a summary like the first one, but it can also used in semantic research. In fact this contain lot of different methods of sentence-transformers but interpret-flair method is a better one in term of accuracy. The next figure show the results obtain by this two methods.

- F-score (micro) 0.964
- F-score (macro) 0.9626
- Accuracy 0.964

results obtain by interpret-flair method

Model	STS benchmark
Avg. GloVe embeddings	58.02
BERT-as-a-service avg. embeddings	46.35
BERT-as-a-service CLS-vector	16.50
InferSent - GloVe	68.03
Universal Sentence Encoder	74.92
Sentence Transformer Models (NLI + MNLI)	
nli-distilroberta-base-v2	84.38
nli-roberta-base-v2	85.54
nli-mpnet-base-v2	86.53
Sentence Transformer Models (NLI + STS benchmark)	
stsb-distilroberta-base-v2	86.41
stsb-roberta-base-v2	87.21
stsb-mpnet-base-v2	88.57

results obtain with different sentence-transformers methods

The last method is a one which can be used in text classification, and have the ability to determined what it is rational. In addition, this method is very useful for model evaluation.
To conclude these four methods all treated about text data-sets (not necessary for the third and four), but there were not effective for the same application. In fact the better way to do is probably to use adapted method for each situation.

3 Management and difficulties encounter during the project

3.1 Management of the project

During the first week we tried to understand the purpose and the attempt of the project, and read documentation about the different methods in order to understand their operation. The second week we started coding with the tabular dataset. The last week we finished that and explored kaggle for different methods for the other datasets, and finally we took the time to write this report.

3.2 Difficulties encountered during the project

We encountered plenty of difficulties during the project. The first one is probably about the language. Another one is that it took a lot of time to understand the goal of this project. But the hardest things it is to try to understand the method, their code, and how to adjust it in order to use in each case and different data-set. Moreover some package did not work on python even if we installed it, that is why we used notebook for LIME method (text and table). We wish we would have been able to get going faster.However the project was complicated by the health problems of one of the author, so it took more time to understand the expectation of the course and get going. Indeed, it lead to absences which prevented us from acquainting ourselves with the tools used in IN104 (like github, Kaggle,...).

4 To Conclude

To conclude, this project was rather difficult but it was interesting to discover machine learning and to notice all the possibilities of application of it. It was also very interesting to learn how to use github, kaggle and to familiarize us a little bit more with Latex thanks to this report.