XAI in the limelight: an application of LIME to classification on tabular data

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

The lack of interpretability of most machine learning algorithms is an obstacle to their increasing presence in society and to troubleshooting. One of the responses to this issue is the framework of Local Interpretable Model-Agnostic Explanations (LIME), which attempts to tackle it by locally approximating the behaviour of any model applied on image, text, or tabular data in terms of human-readable features, also providing a means of aggregation of local explanations to derive information on its global behaviour. Despite its widespread use, the application of LIME to tabular data is not as extensively documented as on images or text. In this work, we test the use of LIME on a classification problem of tabular data from CERN collisions, exploring the advantages of the framework for feature engineering (by comparing it with Principal Component Analysis) and model selection (among Random Forest, a Neural Network and XGBoost). We find both applications of LIME inconvenient, due to the impossibility of aggregating local explanations in a globally meaningful way: our experience suggests the need of developing LIME on the global explanatory level, as well as the exploration of more general and systematic explanatory frameworks and the importance of domain knowledge in feature engineering.

1 Introduction

Machine learning (ML) methods are widely considered as a *black box* approach, in the sense that the decision criteria applied by a trained algorithm are generally opaque to the user or developer. This is a problem when assessing the robustness of a model's predictions, as there are cases where test accuracy overestimates the generalisation accuracy due to phenomena like data leakage (an over-abundance of signal on test) or spurious correlations (for instance, a husky-wolf image classifier identifying wolfs through a snowy background, or a medical diagnosis classifier basing predictions on patient id). While the former is a matter of finding a proper data sample, the latter can only be addressed by looking at the decision process of the algorithm.

Moreover, as ML technology evolves and its role in society expands, the number of cases where transparency is required increases: medical diagnosis, hiring, loans, bail/parole determination or automated vehicles are some examples where one would naturally ask for insight on the decision process not only to ensure its robustness (especially when the stakes are high, *e.g.* if human lives are compromised) but also its fairness (when there is a risk of bias, such as racial or gender discrimination). This has been enforced in the European Union by the General Data Protection Regulation[1] (GDPR) since 2018, namely by the acknowledgement of a *right to explanation* (*e.g.* when one is refused a loan based on an ML outcome).

[1]https://gdpr-info.eu/.

Explainable Artificial Intelligence (XAI) aims to open the ML black box, or to render it transparent, in order to guarantee the key requirements of robustness and fairness, thereby increasing trust in the technology.

Such a problem may be addressed from two different and complementary perspectives: that of *intrinsic interpretability* (or *transparent box-design*) and that of *post-hoc interpretability*. In the former case, the approach is to simply build directly interpretable and transparent models, also known as "glass boxes", like linear models or decision trees. This allows to closely monitor the decision-making process and correct undesirable biases, although the process is model-specific and usually requires solving an optimisation problem with complicated interpretability constraints.

Concerning *post-hoc* interpretability, it involves explaining existing models by building an abstraction thereof, drawing for instance on argumentation theory or simpler, intrinsically-interpretable ML models, as LIME [1](Adrián, Marcos and Rachel) does. This facilitates a model-agnostic approach, although introducing a second model potentially increases error.

Furthermore, in both approaches, the optimisation of the trade-off between simplicity (as these models have to be accessible not only to developers, but in many cases also to non ML experts) and accuracy of representation is still an open issue, as is the adaptation of the explanation to the variability of user profiles (physicians, drivers, candidates to different positions, inmates, etc.).

Both intrinsic and *post-hoc* interpretability may be framed in global or local terms: the first addresses the general behaviour of the model, useful for correcting biases and predicting generalisation behaviour, while the second is concerned with tracing specific outputs, also useful for troubleshooting and a tool for model selection, as well as the focus of the GDPR's right to explanation.

Finally, another dimension of classification of XAI methods regards their methodology can be considered: What is the algorithmic way of approaching the problem? Is it focused on the input data or the model parameters? A division of this problem into BackProb and Perturbation fueled methods helps explain better the model regarding whether the core algorithmic logic depends on back-propagated gradients or randomly/carefully picked instances of the data.

In this exploratory work, we test the established XAI framework of Local Interpretable Model-Agnostic Explanations [1](Adrián, Marcos and Rachel) (also known as LIME), a local *post-hoc* interpretability proposal based on perturbation, with the tabular data provided by the CERN Kaggle competition[2].We assess the capabilities of LIME for feature engineering and discuss briefly how it can supplement conventional performance assessment methods for model selection.

In the following section we review the key aspects of LIME. Subsequently, we present our proposal, whose results are discussed in the following section. Finally, we provide the conclusions of this work and suggest future lines of enquiry in the light of our findings. The full code to reproduce the results found here is in our GitHub repository[3].

The above introduction to XAI is based on the reviews of [2](Adrián), [3](Adrián), [4](Adrián), [5](Marcos), [6](Marcos), [7](Rachel) and [8](Rachel). As these contain significant overlap, rather than inconveniently citing almost every one of them in each sentence we refer to them collectively here.

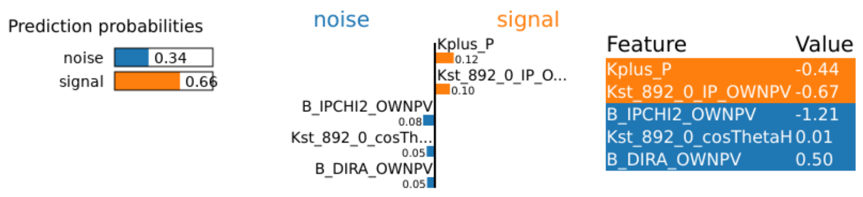
2 The juice of LIME

Local Interpretable Model-Agnostic Explanations (LIME) is an open source framework proposed by Ribeiro et al. in 2016 [1](Adrián, Marcos and Rachel). As the name implies, it focuses on specific instances, and it can be applied to any existing or future model (black box or not).

As illustrated in Figure 1 for the CERN tabular data, LIME provides local feature importance characterisations of interpretable representations of image, textual and tabular data: in the first case it highlights salient pixel sets (or superpixels), in the second it uses words, and in the third it addresses raw (as opposed to a more interpretable representation of) tabular attributes. It models the local input-output relation by linear approximation, which is generally sufficiently accurate on a local scale

[2]https://www.kaggle.com/c/cernsignal.

[3]https://github.com/adrifcid/XAI.



Feature	Value
Kplus_P	-0.44
Kst_892_0_IP_OWNPV	-0.67
B_IPCHI2_OWNPV	-1.21
Kst_892_0_cosThetaH	0.01
B_DIRA_OWNPV	0.50

Figure 1: Example of a LIME explanation on a prediction for the CERN data: as can be seen, it includes the predicted probabilities, the importance ranking up to the desired number of features (5 here, for visualisation purposes) and the corresponing feature values. As explained in the text, our procedure consists on registering the top 3 most important (*i.e.* Kplus_P, Kst_892_0_IP_OWNPV and B_IPCHI2_OWNPV here) and the single bottom (in the 14th position) features for the selected explanations.

and provides an understandable account of the local behaviour of the ML model. More specifically, the algorithm may be qualitatively summarized as follows:

1. Given an observation (input-output instance), generate a new dataset by permutation of the input features.
2. Calculate distances of each permutation to the original input and assign the corresponding similarity scores.
3. Use ML model on the new data.
4. Find a minimal set of features that reproduces the results on the new data.
5. Fit a simple (linear) model using the new data with the minimal set of features and weighted by the similarity scores.
6. The aggregated feature weights of the simple model provide an explanation of the local behaviour of the ML model.

The authors also propose a method called Submodular-Pick (SP) to gather representative local explanations in order to provide insight on the global behaviour of the model. Given a set of LIME explanations, the module SP-LIME computes the global importance of each feature and recurrently picks explanations maximising the increase of a measure of importance coverage (not counting any feature more than once) at each step (it is a greedy algorithm) until the desired number of explanations is obtained.

Finally, in their original paper, Ribeiro et al. report results from a user study of their approach: the use of LIME by non ML experts increased their trust in the model considered, enhanced their ability to predict generalisation behaviour and allowed them to improve the model through feature selection.

In short, LIME seems to address fairly well the XAI problem: its locality has the potential to comply with GDPR requirements, while allowing for a global perspective through aggregation; model-agnosticism yields a universal range of applications; and both experts and non-experts seem to be able to use it for assessing trust in models, predicting their behaviour and improving them.

3 Proposal

This work was inspired by [9](Marcos), where the authors point out the lack of studies on the application of LIME to tabular data, demonstrate the use of the framework on a local and global level on 4 different classifiers (applied to weather tabular data) and evaluate the interpretability of the LIME output via a user study involving non-experts and an original usability-assessment framework.

Drawing on part of the cited article, we considered it of interest to follow a similar procedure with the Kaggle data, chosen due to the familiarity we all acquired with it during the course (which facilitates data manipulation and also discussion and communication of the results). Our goal was twofold: first, to attempt a LIME-based feature extraction on different models to see how far in accuracy we could get without any technical domain knowledge (*i.e.* without accounting for the physics behind the features); and second, to make use of the insights shared by the winners of the competition to discuss model selection based on LIME explanations.

The former application we deemed particularly interesting, because if we were able to obtain an accuracy comparable or superior to that of the winning team by just introducing interpretability (in the form of local feature importance, provided by LIME), it would substantiate some debate on the accuracy-interpretability trade-off seen in class. As we explain below, there are a number of reasons why we did not.

The LIME-based feature extraction consists on seeing what features are important for the model and removing unimportant ones while boosting those that are useful (through addition of powers or products of them, for instance). Therefore, for this strategy to work, one needs to have an accurate-enough model to begin with: if the model is giving importance to the wrong variables, the approach will worsen the results. This is why we start without the "Id" and "Butter" columns and with models with an AUC score at least higher than 75%. We have chosen the AUC score as our measure of accuracy, as was done for the Kaggle competition.

Secondly, both for feature engineering and model selection, we are talking about model-behaviour, so we need to aggregate local explanations to infer global feature importance. Our initial intention was to use SP-LIME to identify 20 representative explanations (not many, but the number was chosen so that this project would not turn into a transcription exercise) but, rather disappointingly, the method does not seem to be available for classification purposes (at least with tabular data): as soon as we switched the mode of the example code[4] from "regression" to "classification", it gave an unexplained error. As far as we have been able to verify, this implementation bug is not disclosed by the authors, and we suspect it to be the reason why [9](Marcos) did not use the module (as they also work on a classification problem with tabular data), turning to random pick (RP) of local explanations instead. We follow their example, and settle for the (less powerful) RP and posterior aggregation (see tables in the Results section). We are aware that this reduces the significance of our analysis (the number of explanations used being already low compared to the size of the dataset), and we considered for instance coding the SP algorithm ourselves (since it is not very complicated), but we did not have the time.

The procedure followed for this work was:

1. Load the data, remove "Id" and "Butter", rescale values and split it with a random seed of 42 and a 0.3 test fraction.
2. Train and test model. Get an AUC score of at least 75% (baseline).
3. Randomly pick 20 correct ($\hat{y} = y_{test}$) predictions. In order to compensate for unbalance in data, 10 are predictions of a positive signal ($\hat{y} = 1$) and 10 of a negative one ($\hat{y} = 0$).
4. Explain RP predictions with LIME. For each explanation, report the top 3 and least important features, and aggregate results (by count) on a table.
5. Use the previous information to improve the AUC score of the model by removing unimportant features and boosting important ones.

The above routine is followed with three different models: Random Forest, a Neural Network (the Kaggle winning kind of model) and XGBoost. To better assess the improvement brought by feature extraction, we compare it with that of including Principal Component Analysis in the pipeline.

Regarding model selection, once we had the 20 explanations for every model, we made use of the insights shared by the wining team on which features were meaningful and why to discuss the performance of each model in terms of the affinity of their respective aggregation tables with this knowledge.

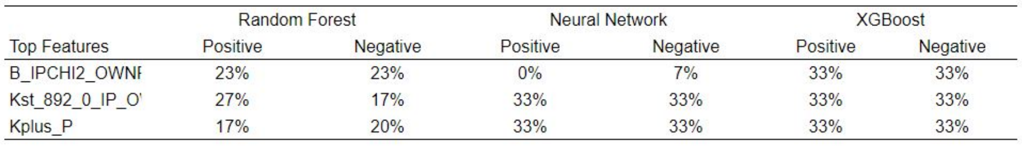
[4]See https://github.com/marcotcr/lime/blob/master/doc/notebooks/Submodular%20Pick%20examples.ipynb, where the authors implement SP-LIME on a regression problem with tabular data.

4 Results

4.1 Feature exploration with LIME

To understand the weight of the features on a global level with LIME, we assessed the frequency that they appear in the top three positions and the bottom position in terms of predictive power per model for both the positive signal and the negative signal.

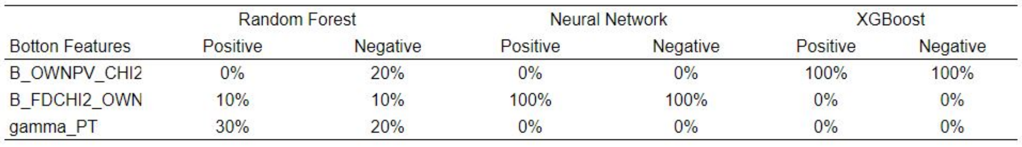
For the top three positions, we display the percentage of times the top three features were present in the randomly selected LIME observations per model. As seen in Figure 2, the same features contributed to the most predictive power in both the positive and the negative case, namely Kst_892_0_IP_OWNPV, Kplus_P, and B_IPCHI2_OWNPV. Another interesting point is that the top features are present both both kinds of signal in the three models, except for B_IPCHI2_OWNPV which is not an important feature in the positive case for the Neural Network.



	Random Forest		Neural Network		XGBoost	
Top Features	Positive	Negative	Positive	Negative	Positive	Negative
B_IPCHI2_OWNI	23%	23%	0%	7%	33%	33%
Kst_892_0_IP_O'	27%	17%	33%	33%	33%	33%
Kplus_P	17%	20%	33%	33%	33%	33%

Figure 2: Summary of the most frequent top 3 features.

Next we looked at the features that were located at the bottom of the LIME output tables, that is, the features that contributed least to the predictive power of the models. From Figure 3 we see that there was less homogeneity between the models, with all having different features at the bottom. Random Forest contained many different features, while Neutral Network and XGBoost had just one feature for both the positive and negative cases, namely B_FDCHI2_OWNPV and B_OWNPV_CHI2 respectively.



	Random Forest		Neural Network		XGBoost	
Botton Features	Positive	Negative	Positive	Negative	Positive	Negative
B_OWNPV_CHI2	0%	20%	0%	0%	100%	100%
B_FDCHI2_OWN	10%	10%	100%	100%	0%	0%
gamma_PT	30%	20%	0%	0%	0%	0%

Figure 3: Summary of the most frequent bottom features.

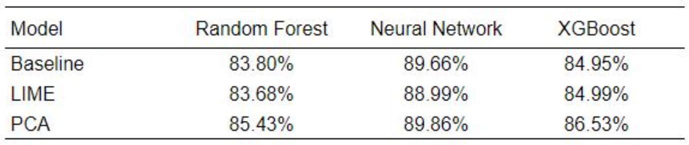
4.2 LIME feature engineering vs PCA

Utilising the results in the previous section, we applied feature engineering to the baseline models to test if we could improve the models thanks to LIME. Furthermore we compare the feature engineered models to PCA models by applying PCA to the baseline.

To perform feature engineering we first took the most important features for each model and boosted them by adding powers of -1, 2, 3, and 4 of each, as well as of some products. In addition, we took the bottom features for each model and removed them from the data again with the goal of improving performance.

We did not achieve any significant improvements from these feature engineering attempts. Random Forest and Neural Network both had worse results than the baseline and PCA models, while XGBoost had just a slight improvement over the baseline, but still worse than PCA. Figure 4 presents these results.

There are a number of reasons for these negative results. First, there is the mentioned fact that we use a small and randomly picked sample of local explanations, which means that it is likely to not be representative of the behaviour of the model (rendering our later feature engineering less significant). Second, when looking at these local observations, we noticed that usually the top 2 features contributed to the negative signal, while those contributing to the positive signal where more



Model	Random Forest	Neural Network	XGBoost
Baseline	83.80%	89.66%	84.95%
LIME	83.68%	88.99%	84.99%
PCA	85.43%	89.86%	86.53%

Figure 4: AUC values per model.

distributed in the ranking: this means that our particular exploration procedure might have overlooked smoke-gun features for the positive signal that could have improved performance upon boosting.

A third issue we have encountered when conducting feature engineering is the uncertainty about the relation of the potentially interesting features to the target value. Indeed, the possibilities for "blindly" boosting a feature or a set of them are endless, and even if our selected features had been truly important for the model as a whole and for the target value, it is very possible that none of the transformations considered for boosting was close enough to the right one to yield a sizeable improvement. We therefore see the importance of domain knowledge in feature engineering, with or without interpretability tools.

It should be noted that while investigating a method of aggregating the LIME observations, we came across the alternative library InterpretML [10](Rachel) produced by Microsoft, a general purpose, one-stop shop allowing among other things the exploration of the data, comparison of model performance and most importantly the examination of predictions globally through overall feature importance and partial dependence plots. This richer environment would potentially perform better at the tasks studied here due to its global functionalities, and particularly the partial dependence plots would be a way to infer to some extent (even if far less than with actual domain knowledge) the relation of each feature to the target value. This is why we include in our GitHub repository[5] an example of the use of this library with our CERN data, anticipating future work.

4.3 Model selection

Regarding the use and capabilities of LIME in the context of model selection, we consider worth noting how easy LIME implementation is in any kind of model, from Random Forests to Neural Networks. This, has remarkably helped us compare the improvement the application of LIME has supposed to the three studied models.

Reflecting on the main difference when approaching the problem between the winners of the challenge and ourselves, we noticed two key points: while they chose to increment the number of features through the combination of all initial features, we only tried combining the resulting top features from the LIME analysis; and when going through their better performing new features, we noticed that those are not even primarily composed of our selected top features. This, leads us to think that our analysis may indeed not be representative enough and our results cannot be extended to the totality of the dataset, since the significant features we obtained are different from those derived from domain knowledge, and that combining the most important features from the model's decision making process, does not necessarily result in an improvement of its performance (at least when such combination is done without domain knowledge, as we have already mentioned).

Another possible reason why our important features did not coincide with those derived by the winning team is that we did not sufficiently take the unbalance of the data into account: since we did not weight the classes when training the models, they may have specialised on identifying noise, potentially giving importance to different features than those used when giving priority to a positive signal.

5 Conclusions

In the present work, we have examined the use of the LIME framework for feature engineering and model selection. While LIME is potentially useful (and advertised as such) for both, our results

[5]https://github.com/adrifcid/XAI.

for feature engineering are rather negative. This might be a sign that domain knowledge is key for proper feature engineering (as was indeed reported to be the case for the winning team of the Kaggle competition). Another possible reason is our procedure not properly accounting for both classes of the problem. Finally, it may also be due to the facts that we only pick 20 explanations per model (lest the use of LIME become highly unpractical) and that such subset is necessarily randomly chosen (since the SP-LIME method is not functional for classification), which lead us to believe that the information obtained through LIME is far from representative.

As a consequence, the use of LIME for any purpose that requires global information on the model (*i.e.* also model selection) seems inconvenient, at least until the SP module is extended to classification problems. This is unfortunate, since it means that our previously-thought complete candidate is in the end limited to local explanations.

On the other hand, we have gotten a glimpse of the potential of the InterpretML library, and consider it worth studying for a more general and systematic approach on interpretability. Therefore, future work could profitably focus on such a promising tool, yet keeping in mind the higher convenience of domain-knowledge-driven feature engineering.

References

(Adrián, Rachel and Marcos) [1] Tulio, M., Singh, S. & Guestrin., C. (2016) "Why should i trust you?": Explaining the predictions of any classifier. In:*Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* ACM. , pp. 1135–1144.

(Adrián) [2] Fouladgar, N., & Främling, K. (2020). XAI-P-T: A Brief Review of Explainable Artificial Intelligence from Practice to Theory. *arXiv preprint arXiv:2012.09636*

(Adrián) [3] Gerlings, J., Shollo, A., & Constantiou, I.D. (2020). Reviewing the Need for Explainable Artificial Intelligence (xAI). *arXiv preprint arXiv:2012.01007*

(Adrián) [4] Markus, A.F., Kors, J., & Rijnbeek, P.R. (2020). The role of explainability in creating trustworthy artificial intelligence for health care: a comprehensive survey of the terminology, design choices, and evaluation strategies. Journal of biomedical informatics, 103655

(Marcos) [5] Vilone, G., & Longo, L. (2020) Explainable artificial intelligence: a systematic review *arXiv preprint arXiv:2006.00093*

(Marcos) [6] Das, A. & Rad, P. (2020) Opportunities and Challenges in Explainable Artificial Intelligence (XAI): A Survey. *arXiv preprint arXiv:2006.11371*

(Rachel) [7] Koh P.W. & Liang P. (2017) Understanding black-box predictions via influence functions. *arXiv preprint arXiv:1703.04730*

(Rachel) [8] Gunning, D., Aha, D. (2019) Darpa's explainable artificial intelligence (xai) program.

(Marcos) [9] Dieber, J. & Kirrane, S. (2020) Why model why? assessing the strengths and limitations of lime *arXiv preprint arXiv:2012.00093*

(Rachel) [10] Nori, H., Jenkins, S., Koch, P. & Caruana, R (2019) InterpretML: A unified framework for machine learning interpretability *arXiv preprint arXiv:1909.09223*