\section{Interpretation Tools}

Interpretability of Machine Learning models is a key component of the trust-building process. For clinicians and ML applied to diagnosis, this is especially true as both for the patients and the clinicians, there needs to be trust in the results. Clinicians need to be able to understand the results from a ML model and why they were outputted if they are to trust it.

There exists a trade off between interpretability and model complexity. With a more simple model, its behavior can be better understood – but it lacks the predicting power to deal with complex data. A more complex model can achieve great prediction score at the cost of a more complex architecture and poorly understood behavior. This is where interpretability tools come in: while simpler models are self-explanatory in their behavior (e.g. linear regression), more complex models can be explained through these tools.

Overall, interpretation tools can be split into two categories; those that focus on the entirety of the model’s outputs (general), and those that focus on single instances or regions around an instance (local). The former can be computed on smaller parts of the dataset in order to give a more granular view. The latter can be computed over the entire dataset (or parts of it) in order to yield a general view.

In the context of the Medalai platform the interpretation tools used should be model-agnostic, meaning that they can be used with any type of model, since the platform offers a range of model choices which differ in their structures.

\paragraph{Data Collection} Understanding what causes outputs to be predicted can help with data collection: if a user knows that having data points with a certain feature helps the prediction, data collection of this particular feature can be insisted upon. In general, having an understanding of how the model behaves yields powerful insights on the value of the features currently available. Moreover since ML algorithms do not behave as clinicians do, the importance or lack thereof of some features might not be intuitive – most people might require to analyze the ML model’s behavior in detail before understanding from which features it benefits more.

\paragraph{Error Tracking}

Interpretability is also important for error tracking. When the algorithm misclassifies or outputs a wrong regression output, the user should be able to understand why. Understanding the model’s behavior generally, locally, or on a single data instance can give insight on its behavior, and some possible confounding factors in the data input used.

\paragraph{Machine learning and the medical field}

In the medical field especially, end-users want to be able why a model acts the way it does. Understanding the edge cases, and in which cases the model might be biased is paramount to a fair use of this kind of technique.

It is vital to be able to explain all or most results. For example, a result that seems to be very different from the rest of the predictions, and that is unexpected, can be analyzed using interpretability tools to gain insight on what in the data or in the model caused this prediction. Understanding edge cases and acknowledging them leads to a better understanding of the model. It is important not to blindly trust the model predictions, but to understand in what it succeeds and in what it fails, even if it does admirably or poorly.

\paragraph{Diagnosis Particularities}

If in a specific context, there is no way to maximize both specificity and sensitivity, the clinician should be allowed to choose which one would be more suited to the context. For example in a case of epidemics, the clinician would not want to miss out on any case – at the cost of treating patients that are falsely positive (high sensitivity but low specificity – meaning that there should be an aim at keeping false negatives low perhaps at the expense of false positives. In a low-resource context however, if there is only enough medication to treat a certain number of patients, a clinician would want to be sure to treat patients that are certain to be positive – maximize specificity (keep the number of false positives low at the expense of potential false negatives). The algorithm should be able to leave this amount of choice to the end-user as it can have a strong impact on results. Specificity and sensitivity are threshold-based measures. In the latter context, it can be important for a clinician to maximize specificity, however within the predictions made by the model, there still exists differences within a “category” (e.g. True Positives). Since the predictions are probability-based, the clinician could be advantaged by selecting only the highest-probability outcomes. This is akin to selecting a very specific sensitivity or specificity threshold.

\subsection{General Model Interpretation}

\paragraph{Importance}

One of the interpretation methods that is focused on the dataset as a whole is feature permutation importance.

This method yields a very easy to interpret result: features can be compared to one another with regards to how much they helped the model achieve a certain metric score.

It is a measure of the impact that different features have on the predictions. The process happens after the model has been trained. The chosen feature is randomly shuffled in the test set, while all other features remain the same, and the ML is tested on this newly partly shuffled dataset. Multiple shuffles occur, and the results of the ML model using shuffled version and normal non-shuffled versions are compared. This technique offers a way of comparing the feature’s importance to a prediction versus randomized versions of itself. If the model relied on this feature for predictions, then there should be a drop in accuracy / chosen metric with the randomized version.

This technique yields an estimate of the importance of the features, relative to the outcome but also to each other. For example if two features are highly correlated, then their feature importance score might be false. If the model can get “as much” information from the correlated feature, the overall accuracy of the model will not suffer from the permutation.

\subsection{Local Model Interpretation}

\paragraph{SHAP}

SHAP values are a computation of a feature’s individual participation to the outcome. For a given feature, this method computes its average marginal contribution over the set of features. As previously discussed, if two features are correlated even slightly, the order in which they are used to make a prediction matters. In a scenario where two features have a correlation coefficient of 1, if a first feature is used, the second feature brings no additional information. The average contribution of the feature over all datapoints is then subtracted from the value being looked into, yielding a positive or negative value. This positive or negative sign indicates whether the feature value was pushing towards one or the other outcome value.

These values explain individual predictions but can be then aggregated over the whole dataset to gain general insights.

SHAP values allow to measure the feature’s sole contribution – disregarding dependence effects caused by correlation to other features or other interactions.

g(z')=\phi\_0+\sum\_{j=1}^M\phi\_jz\_j'

with g the explanation model, z the feature group vector and phi the feature value for a given feature j. The feature group is the number of features being considered.

This method is robust and can be used on anu model. However it can be more cumbersome to interpret than some other methods.

\paragraph{LIME}

**Cut this down to a few paragraphs #thisAintNoTextbook**

The overall idea of the LIME method is to be able to mimic the behavior of a model (usually a black-box model) with a simple linear model (surrogate model) in the neighborhood of a chosen point.

There is already a choice to be made, as the neighborhood can be large in which case the approximation will reflect a more general model behavior - or it can be small and the approximation will reflect the behavior of a model on a certain point.

Both come with a lot of drawbacks. The first one being: there is no way to define a value of the neighborhood size that fits all cases. If it is too small, the surrogate model will fail to capture a behavior that is representative of the model in general. However if it is too large, the fact that the surrogate model is so simple makes it unable to deal with complex boundaries.

**[Source \cite{altm1ann\_limitations\_2019}]**

Intuitively, if the surrogate model is trying to approximate a black box model's behavior around a certain point, there is a need to determine how large the neighborhood should be. If the width is too large, the surrogate model might approximate too much, while if it is too small, it might not generalize at all.

This is shown in the Fig.\ref{lime\_kernel\_widths}, where one can see that while one kernel width allows for a high-fidelity approximation of the true model, the other one yields an approximation that is overly general and can not be considered useful.

\begin{figure}

\centering

\includegraphics[width=0.9\linewidth]{images/lime\_kernel\_widths.png}

\label{lime\_kernel\_widths}

\caption{LIME method over a complex boundary with different kernel widths}

\end{figure}

One of the main issues with this method is linked to the nature of most medical data sets. If the method is used on a high-dimensional dataset, the distance metrics lose part of their separability power**. \textbf{should ref this probably}.**

The complex nature of the dataset might also make the decision boundaries more complex. Since the overall aim of the LIME method is to yield an interpretable model, the number of coefficients in the surrogate model should be kept low.

Procedure:

\begin{itemize}

\item select a specific data point to be investigated within a dataset

\item create a new dataset by adding noise to the original one (perturbations) so as to get more items especially around the point of interest

\item get the black box model predictions on this new data set

\item give each data point in the dataset a weight according to its proximity to the data point of interest (higher weight if closer)

\item train the surrogate model on the perturbed dataset with the labels being the black box model's predictions

\end{itemize}

The surrogate model is trained according to a loss function which takes into account both the targets and the number of coefficients used (regularization of complexity).

The training is only valid around the neighborhood of the selected data point.

The way the weights are distributed to the data points is done using a kernel function. The distances considered are between the data point of interest and each other data point from the dataset. Based on the different types of data, different proximity measures can be used. For categorical variables, Hamming distance can be used. For continuous variables, Euclidean distance can be used. If both types of data are present at the same time, Gower distance can be used.

There is however another parameter that is worth investigating: kernel width.

This is basically the number of points considered - those in the neighborhood of the data point being investigated.