Matt Ross,
Manager, Artificial Intelligence
Information Technology Services,
City Manager's Office
The Corporation of the City of London

Suite 300 – 201 Queens Avenue
London, ON, N6A 1J1

Addressing Explainable AI and Bias in the City of London's Chronic Homeless AI Model – December 2, 2020

Overview:
In August 2020 the City of London deployed a machine learning system which predicts the risk probability that consenting shelter users would become chronically homeless (>= 180 days in shelter per year) within the next 6 months. High risk predictions are then prioritized for review to receive additional resources at the discretion of the shelter and its case workers. The current phase of the implementation involves one early adopter shelter who is using the model's predictions and hosting community led conversations involving shelter leaders, case workers and individuals with lived experience to determine how the model's predictions should be used, what resources should be prioritized, when, how and by what criteria.

The machine learning model used was a deep learning neural network (architecture diagram). We employed an explainable AI technique called Local Interpretable Model-Agnostic Explanations (LIME) which allowed us to unpack and understand the inner correlations and innerworkings of what would normally be considered a "black box" model.

Explainable AI:
Explainable AI and Interpretable AI are often used interchangeably. To clarify the distinction, interpretability often refers to whether the machine learning model is a black box model or not, that is, can explanations be simply pulled out of the model. Explainability refers to whether the internal mechanics can be explained to humans. Our Chronic Homelessness AI (CHAI) model was **not** inherently interpretable, but **was** explainable AI with the use of LIME.

The reason we built explainable AI into the system requirements was driven by both bottom-up and top-down considerations. In our bottom-up stakeholder engagement conversations, trust was a high priority issue. It was made clear this prototype would not be deployed if case workers couldn't understand "why" a particular prediction was made. In our top-down AI governance environmental scan, explanations were a common recommended risk mitigation measure for models such as ours that could have a significant impact on resident wellbeing.

Once we knew explainable AI would be a requirement, there were two broad approaches we could take:

1) Use inherently interpretable models (e.g. decision trees, logistic regression, etc). This completely side steps the issue of uninterpretable "black box" models and is the most common approach in government when faced with this choice. The downside of this approach is sometimes a "black box" model is the best model with the best performance metrics for the given use case.

2) Use model-agnostic interpretability methods (article) to enable explanations of any model whether or not it's inherently interpretable. The downside here is these models can be highly configurable and can introduce grey area into what is a "good enough" explanation. They also tend to be computationally expensive.

It's important to note, that during a data science implementation, you commonly first begin by prototyping with a variety of different models that could work with your given use case and dataset. For example, when we first prototyped our system, we explored both interpretable and uninterpretable "black box" models. The "black box" model gave significantly better performance metrics than the interpretable ones. So, given the requirement of needing to trust and understand "why" our model made its predictions, we had to either go with inherently interpretable models and accept decreased performance (and thereby increased risk of negative outcomes) or utilize a model-agnostic method.

After some research, we decided upon LIME as it had significant support and wider spread adoption in the data science community. Though there are other methods (e.g. SHAP, Anchors, etc). LIME works by probing the trained black box model by feeding it perturbed (slightly changed) data and seeing how its predictions changes. In our implementation, LIME probed the model approximately 390,000,000 times, then fit an interpretable linear model based on how the model reacted to those probes. You get the graphs below from those linear models that are fit on the perturbed data.

Our implementation of LIME in the CHAI model provides both a) an explanation of what features correlate with a high risk of chronic homelessness across the whole City/all-clients in the system (graph on left, called the **global** surrogate model); as well as b) an explanation of what features correlate with high risk of chronic homelessness for a particular individual client in the shelter system (graph on the right, called the **local** surrogate model). The global surrogate (left graph) is useful to explain the entire model's functioning, to generate trust and identify if there could be significant unintended bias. The local surrogate (right graph) is useful to explain to a client why a prediction was made on them specifically.

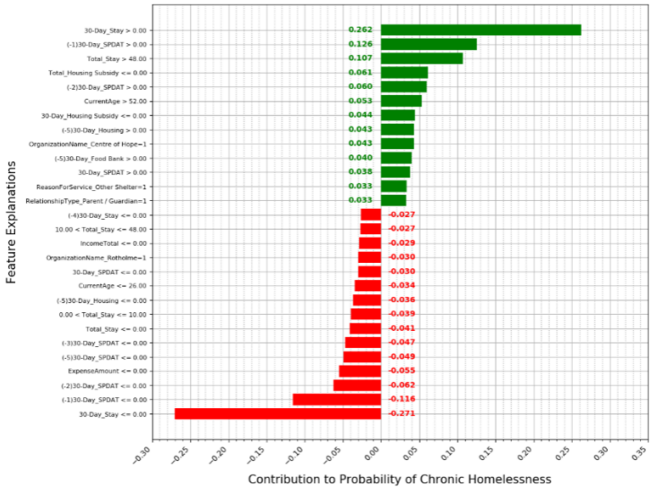
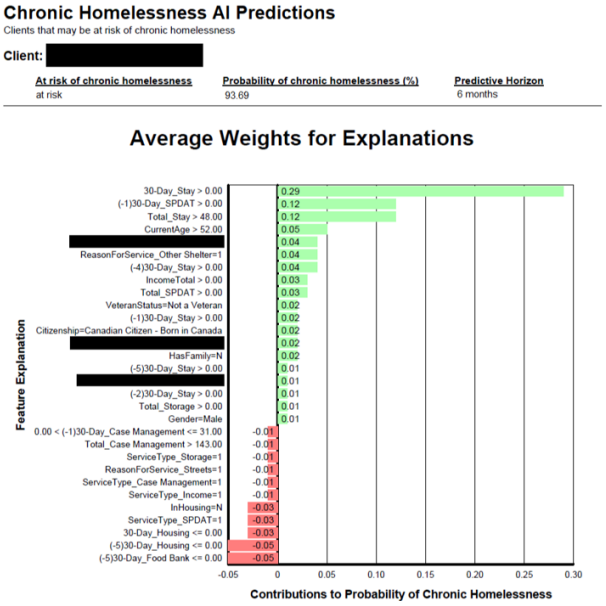


Figure 7: Results of the LIME submodular pick procedure. This graph communicates an approximation of model functionality. Each bar corresponds to the weight of a feature value or range. Green and red bars indicate contribution toward and against prediction of chronic homelessness. The magnitude of a bar indicates its relative influence in the model's decision.



The issue we found with explainable AI methods is what is a "good enough" explanation. From our case, implementing LIME is by no means a black and white endeavor. It required iterative cycles of implementing LIME, sharing results with our Homeless Prevention Division and tweaking hyperparameters of the LIME implementation, and repeating this cycle until the explanations were stable and reasonable to case workers and service delivery experts. The "what is good enough" problem can come up with inherently interpretable models as well, and you're often locked into a type of explanation driven by the particular model you are using.

One of the successes we had in implementing LIME was that our initial models showed a particular feature in the dataset as extremely contributory towards a clients prediction of chronic homelessness. In seeing this, we investigated with our Homeless Prevention division and our shelters. We discovered that the way this data field was being used by case workers was different than expected or documented. This use essentially leaked the ground truth to the model during training meaning it relied on that and essentially ignored all other features. This is a critical failure of a machine learning model due to a data leakage problem, and most performance metrics used in model analysis will not identify this issue, but explainable AI using LIME did identify this issue. Therefore, for us, explainable AI is an inherent part of good machine learning model development best practices at all impact levels, not just for the sake of transparency and end-user trust, but you end up building better models.

Bias:

An important point often brought up in discussions around AI and Bias is that even in a situation where a model is being "used for good" such as in our CHAI model, we still have an Human Rights obligation to ensure there is no discrimination. The first step in eliminating bias, is knowing whether you have any or not. That is, the first step in any data science endeavor should be exploring the data. Identifying where certain features and attributes might have no representation or over/underrepresentation in the dataset is a key early step. Knowing there is feature imbalance and underrepresentation doesn't necessarily mean your model has bias, but it can help identify where the model might struggle. Importantly, investigation of bias must happen not only at initial data exploration phases, but throughout model development and monitored throughout once a model is in production.

With our CHAI model, explainable AI through LIME went hand in hand with ensuring there was not any unintended bias. What we found using LIME on our final model, is that the model didn't significantly rely on client demographic attributes in making its prediction. It relies heavily on service usage patterning (e.g. # of shelter stays this month, last month, two months ago, similar counts for other services, and trajectory of other service uses over time rather than simply a static

aggregate count). The second check we pursued, was to take our trained model, and run inference (i.e. make predictions on client data) using artificial client data where we would adjust protected demographic parameters such as age and gender, to see how the models predictions would change when we gave it feature values underrepresented in the training dataset. We found that there was never a significant enough change in the final prediction outcome from the model as service usage patterns were what the model relied on in making a risk prediction.

This is great for our model, but there are many examples where demographic attributes will heavily sway a model's predictions. In these cases, this unintended bias must be eliminated. In fact, our earlier version of the model had this problem, and it was only when we reframed the data pre-processing architecture to focus on service usage patterns rather than just demographic attributes that this bias was eliminated.

There are four broad approaches I've seen to eliminate unintended bias:

1) **Remove sensitive/protected fields.** If your model is misclassifying a particular protected data type because it is under (or over) represented in the training dataset, I've seen some implementations simply remove the field from model pre-processing and training all together. This is the simplest and cheapest approach and largely guarantees bias related to that field is eliminated. The downside from a data-science perspective is this may impact model performance. There are also some cases when these fields are essential (e.g. clinical machine learning models especially). Finally, this doesn't get to the core issue, and the deleted feature may have statistical traces/correlations in other non-deleted features which can still create bias if the model picks up on them.

2) **Get more data.** Another approach to eliminating bias is to increase the size and diversity of your training data to give your model enough diverse examples so that it doesn't exhibit the unintended bias. The downside of this is that it sometimes isn't possible, whether due to financial constraints as collecting new data is expensive, or perhaps you do have all possible training data and that bias is inextricably part of the dataset (e.g. systemic societal biases represented in our training data).

3) **Feature engineering.** If you cannot remove the sensitive field or generate more data, you can use feature engineering to create new data from the data you already have. Feature engineering is the process of performing manipulations and calculations using data you already have, to generate meaningful features for the model. This could have no impact on model bias or performance, but it's worth mentioning as this is what worked for us in the CHAI model. I mentioned above our original model in development suffered from bias. That model utilized client demographic attributes and service usage counts (e.g. how many total number of shelter stays in aggregate client history). We conducted feature engineering to calculate from the raw data, features which represented the time-series **pattern** of service usage of a client (e.g. shelter stays last month, two months ago, etc for every service in the dataset). This simple change had a significant impact on eliminating unintended bias.

4) **Algorithmic methods.** A final approach if the first three are infeasible or if they don't solve your problem, is to employ an algorithmic method(s) to eliminate unintended bias. These are methods which force the model during the model training process (whether an inherently interpretable or "black box" model) to learn and generalize in ways which don't exhibit the biased behavior. Many of these algorithmic methods force constraints upon the model during the training process to learn an unbiased representation which doesn't overly rely on those sensitive/protected features. The downside with some of these approaches from a pure data science perspective is that you often get a decrease in performance metrics (i.e. lower accuracy, precision, or recall). This decrease in pure performance metrics is in my opinion the reason why many data scientists never learn these methods. As there are awards, prizes and publications for who beats state-of-the-art performance metrics, not for who has the fairest model. This incentivizes ignoring anything that decreases performance.

In early phases of our CHAI model development, we were beginning to implement an algorithmic method called Adversarial Debiasing across data features such as Gender and Age. But then found the change to our model architecture and data-preprocessing (i.e. feature engineering mentioned above) to focus on service usage patterns rather than exclusively demographic features eliminated the biased behavior of the earlier model. Over the last 5-7 years, there has been significant work towards algorithmic methods of bias reduction/elimination, as well as a variety of tools to support data scientists in implementing a variety of bias reduction techniques (e.g. IMB's AI Fairness 360 library gives a good overview of some main ones).