AI Literature: Prologue

Sanjana Chowdhury

From July 2020 to November 2020, I was a Technical Program Manager Intern at a startup called WhyLabs. There, I helped run their AI community called Rsqrd AI. Part of my responsibilities was to turn several tech talk recordings into blog posts.

I've written 8 blog posts for the community on topics ranging from using synthetic data to the importance of building reliable pipelines. My posts have received a lot of recognition and praise from AI leaders in the industry.

There were several challenges in making this happen.

1. The content of these talks were vastly different from anything I've learned about AI/ML before. Every talk was a different subject matter I had to learn quickly, and I had to get comfortable enough to write an entire article on the content I just learned.
2. I had no model or precedent on how to write these posts. I had to figure out on my own what the best way was on how to present the information in a sustainable, efficient manner.
3. I needed to determine how to best communicate the information to both technical and non-technical audiences - how can I make this easy to understand but still engaging for everyone?

The following two entries are blog posts I've written for the community under the field of explainable machine learning. The first one, *Challenges in Explainable Machine Learning*, was written in September and the second one, *Beyond Feature Importance: Explaining Uncertainty Estimates*, was written in October.

To write these posts, I had to quickly get familiar with the field of explainable ML. I spent quite some time understanding the research presented in both pieces to determine how to best present them.

Note the use of timestamps and the liberal amount of links to external resources throughout the posts. This was my solution to guide all kinds of readers through the article: if they wanted to go into further detail or didn't understand a topic, they could find a hyperlink for their needs. This also helped me write content quickly - if there was something I felt like I couldn't explain well or I thought would ruin the flow of the article, I would just send the reader to a timestamp in the recording or an external resource.

Challenges in Deploying Explainable Machine Learning (w/ Umang Bhatt)

In this talk, Rsqrd AI welcomes Umang Bhatt, Ph.D. student in the Machine Learning Group at the University of Cambridge and Research Fellow at Partnership on AI! Umang speaks about the challenges in ML explainability and his work in collaboration with Partnership on AI on ML explainability in the industry.

How are existing approaches to explainability used in practice? [Answer: only used by developers]

The following is the work of Umang and multiple collaborators and was also presented at the ACM FAccT 2019 Conference as the paper Explainable Machine Learning in Deployment.

Umang mentions how there's been massive growth in the literature surrounding XAI (explainable AI). In particular, there has been an increase in proposed algorithms that

aim to “explain” machine learning output. With this observation, one of the things Umang’s team set out to do was study how organizations use these algorithms.

For their study, they had semi-structured interviews with 50 people from 30 different organizations, with interviews lasting between 30 minutes - 2 hours.

Shared Language

The team realized there needs to be shared and established language for thinking about explainable AI to be able to have these conversations around the topic. Umang provides the following two definitions:

- Transparency: Providing stakeholders with relevant information about how the model works: this includes documentation of the training procedure, analysis of training data distribution, code releases, feature level explanations, etc.
- (Local) Explainability: Providing insights into a model’s behavior for specific datapoint(s)

The key point is that explainability is an algorithmic approach to transparency.

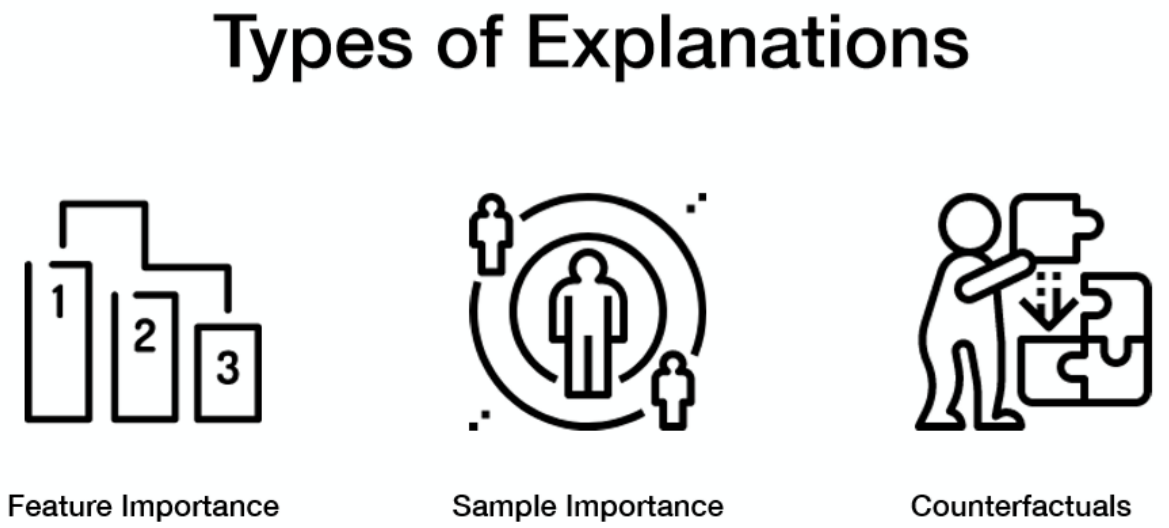
Questions That Were Asked

Once the shared language had been established in the interview, the interviewees were asked the following questions:

- What type of explanations have you used (e.g., feature-based, sample-based, counterfactual, or natural language)?
- Who is the audience for the model explanation (e.g., research scientists, product managers, domain experts, or users)?
- In what context have you deployed the explanations (e.g., informing the development process, informing human decision makers about the model, or informing the end user on how actions were taken based on the model’s output)?

Types of Explanations

Umang and his team found certain types of explanations that were very popular:



- Feature importance: using features of relevance (important) over less relevant features (unimportant) to make the right decisions and avoid misleading conclusions like spurious correlations
- Sample importance: attempts to answer the question “which training data points are the most influential when predicting on a new test point?”
- Counterfactuals: attempts to answer the question “what do you need to change about yourself in order to change your outcome with respect to this predictor?”

Who Cares about Explainability?

The next thing they realize is who are the stakeholders that have a vested interest on ML explainability:

- Executives: the team found that corporations, engineering managers, chief data scientists, etc. had experiences where their executives would ask them to start using explainable ML so they can say that their company uses explainable ML
- Engineers: they are the most popular stakeholder. ML engineers and data scientists use these explainability techniques to sanity check their models.
- End users: developing consumable explanations creates trust between end users and AI
- Regulators: regulators, such as laws, diplomats, and the research community, have a vested interest in understanding how XAI works to see how it affects the surrounding community.

Final Findings

The team finalizes 3 key findings of the study.

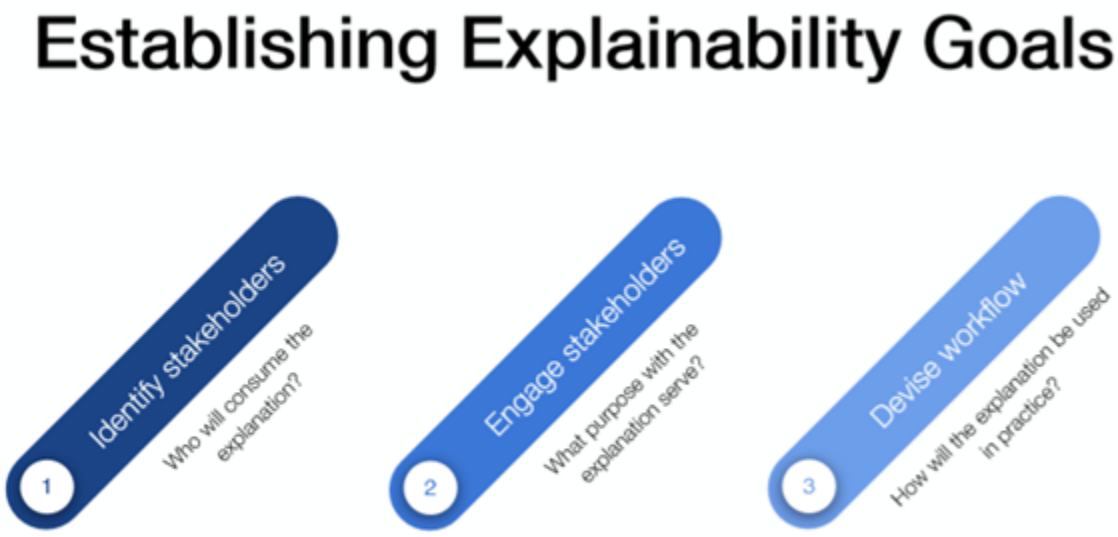
Explainability is used for debugging internally

As seen earlier, explainable AI is used primarily by ML engineers for ML engineers. They kind of add this layer of XAI on top of their existing pipelines to check their models. These systems don't find their way to the end user as exclaimed by some XAI literature. The primary uses for these systems that they found were in content moderation and in finance.

Goals of explainability aren't defined internally

They found that important stakeholders are not involved from the beginning which ultimately makes it hard for explanations to be consumed by the right people. It's

important to engage with these stakeholders and define the purpose and role of these explanations and to establish explainability goals.



Technical limitations make explainability hard to deploy in real-time

There were some frequent limitations that came up in the study.

- Spurious correlations exposed by feature level explanations: if I find a problem with the feature attribution, how do I fix it?
- No causal underpinnings to the models themselves: execs and regulators want to throw things into a model and want to see its impact on the outcome: they expect to see some sort of correlation. The models might not be causal at all and support this system that creates correlations. From this stems causal explanations for non-causal models, and disinterest from execs and regulators because there isn't causal understanding.
- Sample importance is computationally infeasible to deploy at scale: which training points are similar to the model and create similar outcomes? That's hard to answer, especially at scale.
- Privacy concerns of model inversion exist: if I can give you explanations, I can invert your model and invert your training data and reconstruct it, and it becomes a huge privacy concern.

Can existing explainability tools be used to ensure model unfairness? [Answer: not in their current form]

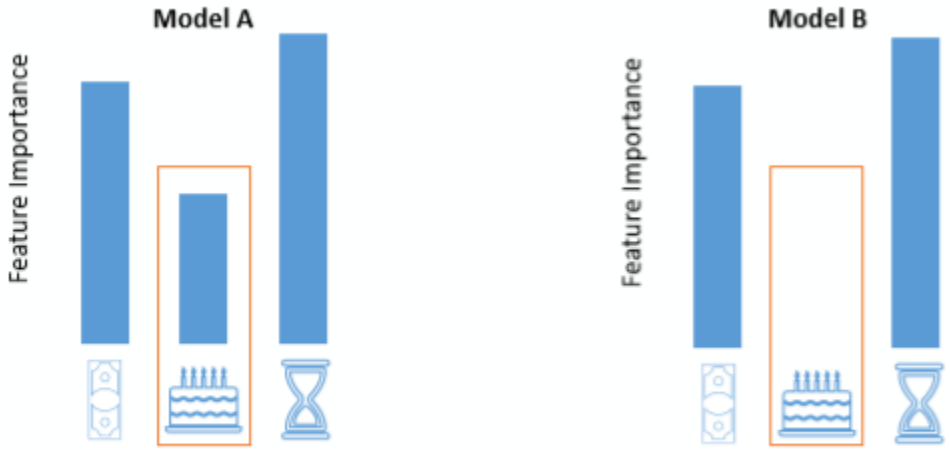
The work that Umang talks about that answers this question comes from another paper Umang worked on called [You Shouldn't Trust Me: Learning Models Which Conceal Unfairness From Multiple Explanation Methods].

The key (and slightly depressing) takeaway from this section that Umang emphasizes is:

Feature importance reveals nothing reliable about model fairness.

Why should people care about good explanations?

Heed the following example from Umang.



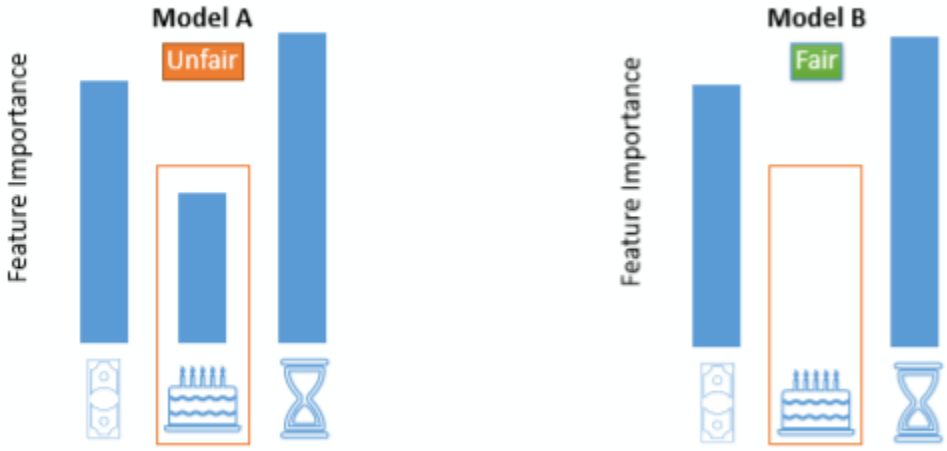
mock models with the same accuracy, but different importance assigned to different features

Here, there are two mock models with 3 features: 1. wealth 2. age, and 3. time spent on doing some downstream tasks. Say someone uses the same explanation techniques on them.

The explanation on the first model ranks the importance of the features as time> wealth > age.

The explanation on the second model ranks the importance of the features as time > wealth, and age is unimportant.

These models can perform the same with the same accuracy, but their explanations are vastly different. In theory, a regulator can come in and say that model A is unfair because it uses a discriminatory attribute, age.



model B is deemed more fair than model A because of the importance of 'age'

This problem begs the question: how can someone turn model A into model B without losing accuracy?

Attempting to manipulate explanations

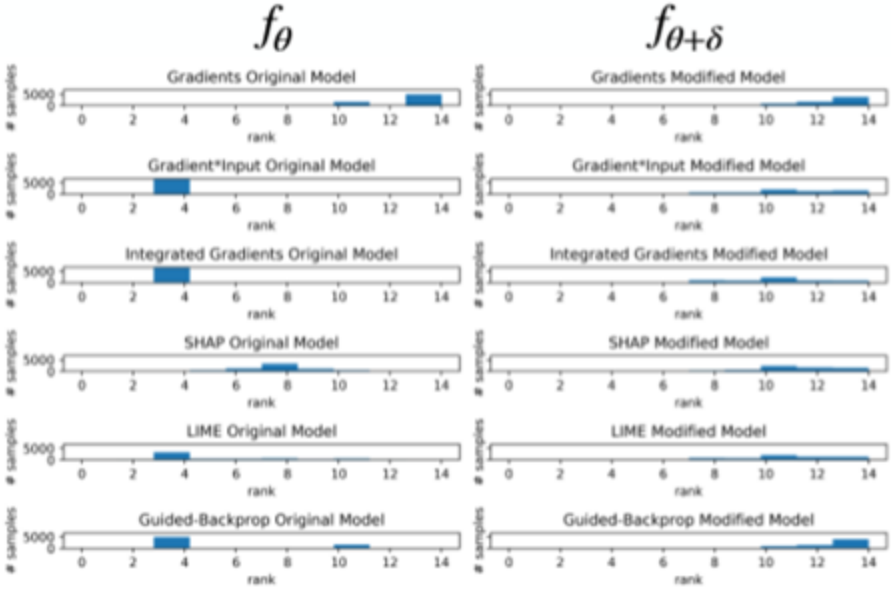
There is some work already that's gone into manipulating explanations. The solution explored in this paper is downgrading explanations with adversarial perturbations on the parameters such that the authors can hide the unfair or discriminatory feature. Or simply explained, attacking the parameters of a model such that the attribution given to a certain feature (like age) decreases.

Results

The setup and method used in the paper can be found at 25:50 in more detail. The goal of the experiment parameterized by theta is to find some delta that gives the aforementioned results. Here, f_θ is model A, and $f_{\theta+\delta}$ is model B.

What Umang and the authors are looking at is importance rankings. Here, they are looking at 14 features. They can take feature importance rankings and just rank features based on the magnitude of attribution given to them. The paper reports the magnitude and ranking of the discriminatory features with respect to different explanation techniques.

Results (Importance Ranking)



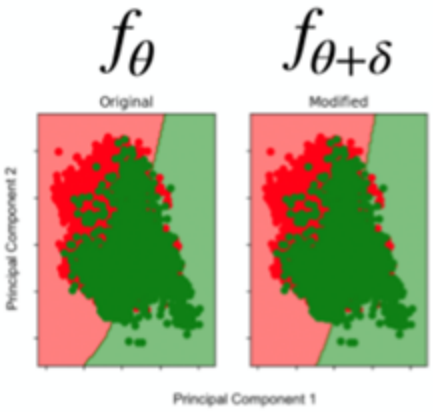
importance rankings of a discriminatory feature in model A (f_θ) and model B ($f_{\theta+\delta}$)

The thought behind this is that a regulator can look at the rankings for a discriminatory feature. If the ranking is very high in an explanation such as in model A, the regulator can say that the discriminatory feature is being used too much and that another model would be better. In model B, the authors show that they can flatten out the gradients with respect to multiple explanation techniques and scale back the magnitude of attribution given to a feature.

TLDR; the paper's adversarial explanation attack does 3 key things: significantly decreases relative importance, generalizes to test points, and transfers across explanation methods, along with the following key findings:

Findings

- Little change in accuracy, but difference in outputs is detectable
- Low attribution achieved with respect to **multiple** explanation methods
- High unfairness across multiple fairness metrics (compared to holding feature constant)



How can we create explainability tools for external stakeholders? [Answer: community engagement and thoughtful deployment]

This work is called [Machine Learning Explainability for External Stakeholders], and it was based on a closed convening hosted by the Centre for the Future Intelligence, Partnership on AI, and IBM.

The Discussion

In this activity, 33 participants from 5 countries came together in New York in early February.

They were 15 ML experts, 3 designers, 6 legal experts, and 9 policymakers. They came from a wide variety of domain expertise with backgrounds in finance, media, healthcare, and social services. The goal was to facilitate an inter-stakeholder conversation around explainable machine learning.

There are two main takeaways:

1. There is a need for community engagement in the development of explainable machine learning.
2. There are nuances in the deployment of explainable machine learning.

Umang proposes to tackle these takeaways as questions to be answered by the community and further research.

Community Engagement

The following questions should be asked when engaging with the community.

1. In which context will this explanation be used? Does the context change the properties of the explanations we expose?
2. How should the explanation be evaluated? Both quantitatively and qualitatively
3. Can we prevent data misuse and preferential treatment by involving affected groups in the development process?
4. Can we educate external stakeholders (and data scientists) regarding the functionalities and limitations of explainable machine learning?

Deploying Explainability

In the same way when considering community engagement, the following important questions should be asked when deploying explainability.

1. How does uncertainty in the model and introduced by the (potentially approximate) explanation technique affect the resulting explanations?
2. How can stakeholders interact with the resulting explanations? Can explanations be a conduit for interacting with the model?
3. How, if at all, will stakeholder behavior change as a result of the explanation shown?
4. Over time, how will the explanation technique adapt to changes in stakeholder behavior?

All of these things are completely context dependent, and that's why it's important to have community engagement from creating the model to iterating on it.

Conclusion

It's become critical to be able to explain machine learning and to be able to answer questions on where outputs come from. There's increasing interest from stakeholders in explainable AI, but it's a lot easier said than done. While current explainability tools have a long way to go and are not sufficient in their current form, AI practitioners can continue to create meaningful explainability tools for stakeholders through community engagement and thoughtful deployment.

Cool Stuff to Check Out

Explainable Machine Learning in Deployment.

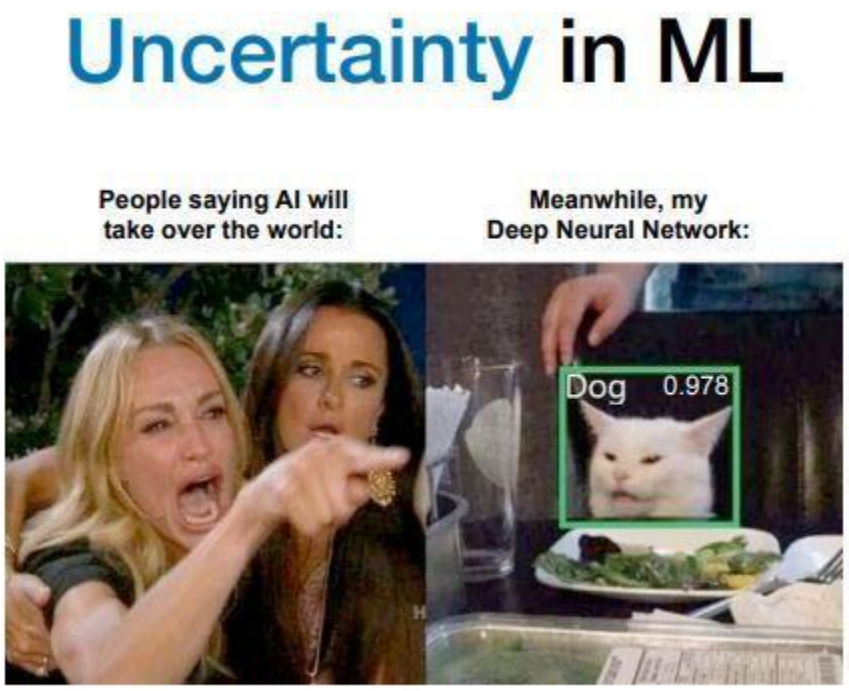
You Shouldn't Trust Me: Learning Models Which Conceal Unfairness From Multiple Explanation Methods.
Machine Learning Explainability for External Stakeholders

Interesting questions from the video:

- How strongly do regulators insist on explainable models? Are they open to non-explainable models? And if so, what are the desiderata? 20m 50s
- Is there an agreed method to explain a given model regardless of where it's deployed? 32m 30s
- What are your thoughts on liability, policy, and safety standards in spaces like self-driving cars in regards to explainable ML? 38m 34s
- What are the overall incentives of using explainable AI? 55m 43s

Video: Rsqrd AI - Umang Bhatt - Challenges in Deploying Explainable Machine Learning
Slides: Challenges in Deploying Explainable Machine Learning
All information and ideas presented in this post are that of the speaker and the talk.

Beyond Feature Importance: Explaining Uncertainty Estimates (w/ Javier Antorán)



In this talk, Rsqrd AI welcomes Javier Antorán, PhD student at the University of Cambridge! Javier talks about the existing approaches to interpretability and how a new approach using counterfactuals can be used to explain uncertainty.

What is Interpretability?

Machine Learning interpretability is a broad term which is used loosely, but generally refers to the ability to explain or present something in understandable terms to a human. Someone could say an ML model is interpretable if humans are able to understand the reasoning behind its decisions. Others might go a step further and argue that a model is interpretable when humans are able to reproduce its reasoning and thus predict the model's outputs given its inputs (this is known as forward stimulability). A really good overview of ML interpretability is given by Finale Doshi-Velez and Been Kim.

Existing Approaches to Interpretability

Interpretable Data Driven Decision Making

There are many existing interpretable “data-driven-decision-making” models, like linear regression or decision trees. In these models, it’s very simple to find the reasoning behind a decision. A linear regression’s output is the sum of the weights of its input features, meaning features that have more weight directly have a bigger impact on the output. A decision tree is dictated by yes-no questions and once all the questions have been asked, predictions are made and are constant within the leaf node. It’s fairly simple to trace back a decision up the tree.

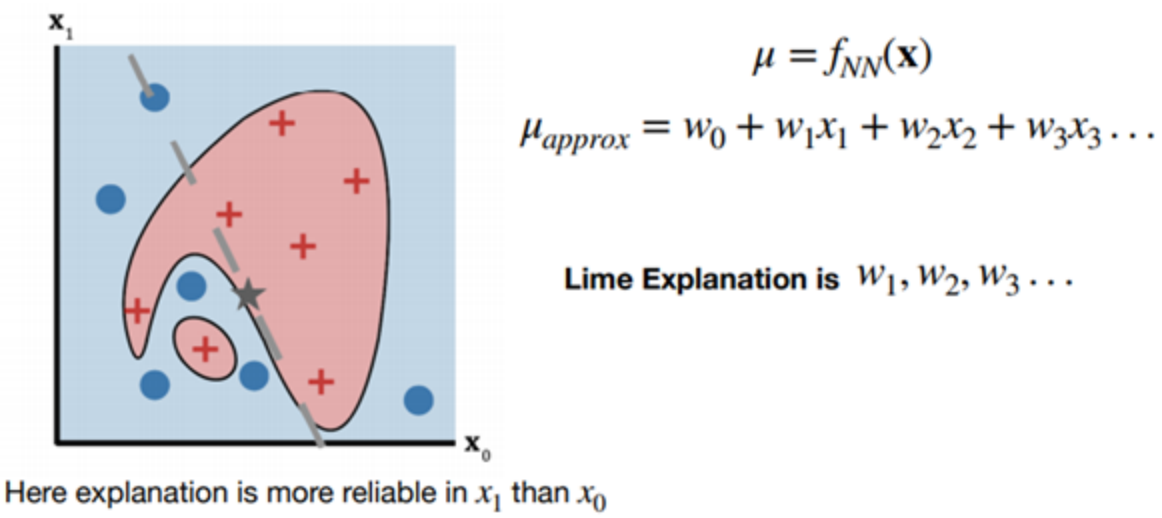
Not Very Interpretable Data Driven Decision Making

If one sacrifices interpretability and the ability to easily determine the pretense of an output (ability to correlate where a decision comes from), a wider range of predictive models become available, like neural networks. They can do some cool things like take in the image of a road and output driving commands. Given this, it’s not as easy to predict the decision making for the output of a neural network like linear regression and decision trees.

Feature Importance

LIME

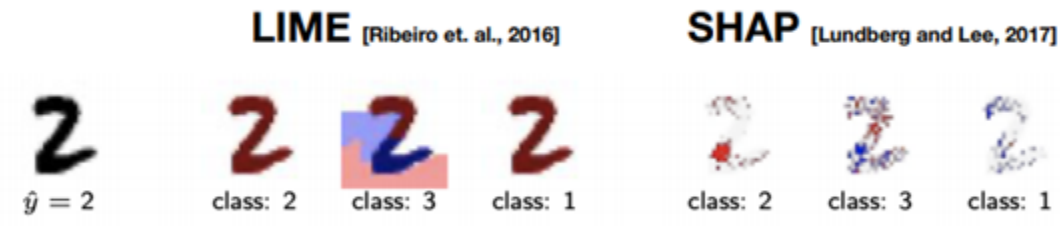
Once interpretability is sacrificed in applications such as neural networks, how does someone get it back? The most commonly used approach is feature importance, and within the family of feature importance techniques is LIME (Local Interpretable Model-agnostic Explanations). The idea behind LIME is that someone can approximate their nonlinear model with an interpretable linear function. For a more in-depth look at how LIME works, you can go to 5m 05s.



Applying LIME on a non-linear function

Feature Importance on Images

When approximating with linear models on non-linear functions, it can become meaningless on strongly non-linear functions. Javier goes into detail on an example where LIME falsely classifies the pixels of an image of the number '2' as positive evidence for the number '1'. The example can be found at 6m 52s.



Approximating becomes meaningless on extreme non-linear functions

Counterfactual Explanations

The alternative to feature importance is counterfactual explanations. The idea behind counterfactuals is that they capture the notion of how things would have worked out if some factor would've been different – the "what-ifs" if a factor were to change. An example that Javier gives is in the medical field when giving a medical diagnosis. Once a final outcome and diagnosis has been resolved from running several tests and procedures, it's commonplace to ask how an outcome would have been different had the medical staff done something differently. Another example of a counterfactual would be to think about "what if fish could fly? How would our

ecosystem be different?". Here's an article that goes further in depth on counterfactuals.

The interpretability community has started using counterfactuals to ask questions like what sort of interventions does someone need in order to get a desirable result – what would need to change? In the context of classifiers, it's questions like:

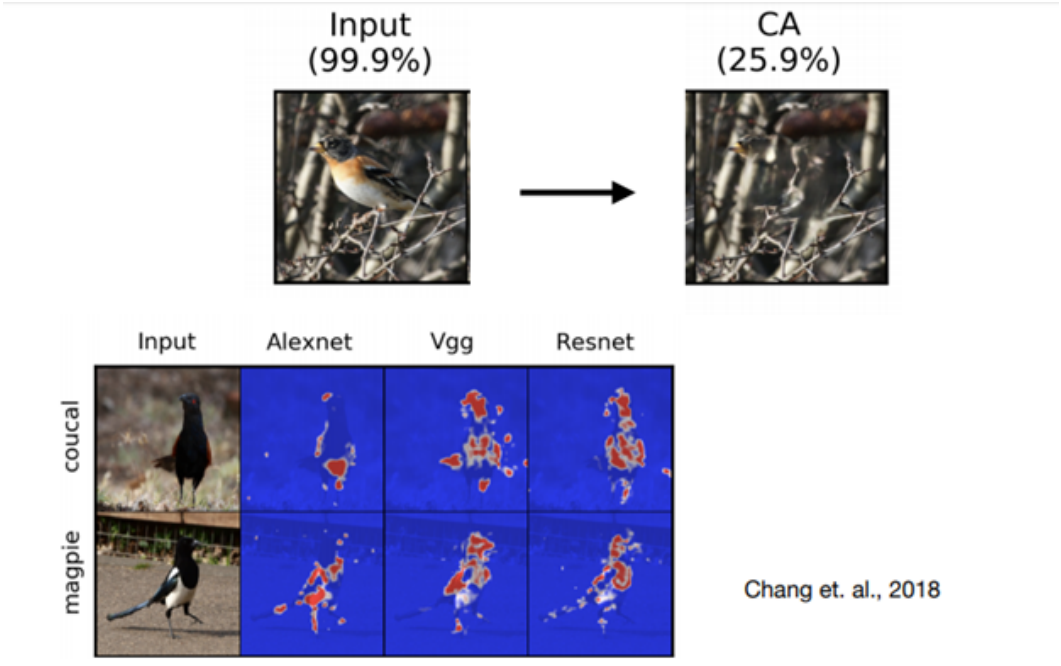
- What features would I need to remove such that my model's confidence decreases?
- What features would I need to remove such that my model's prediction changes?

The cool thing about counterfactuals is that they're model-agnostic – their explanations have completely clear meanings with no need to approximate like in linear explanations in LIME.

Counterfactuals in Image Classification

Javier gives an example of counterfactual explanations in image classification.

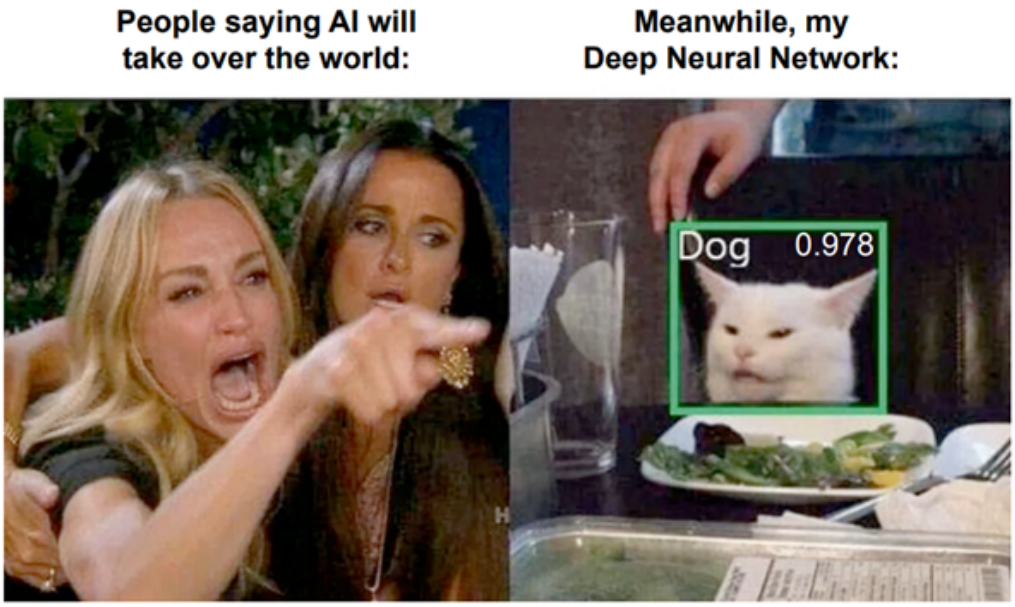
Given a picture of a bird, each pixel can be removed and replaced. With each step, there's a question that is asked: is it still a picture of a bird? This process can keep going until the model doesn't think there's a bird in the picture anymore, creating a collection of pixels that identified the animal. Full explanation can be found at 10m 09s.



Identifying the "bird" pixel by pixel

Uncertainty in ML

Despite neural networks having gained a lot of popularity recently, a well-known issue is that neural networks are very overconfident and tend to behave erratically in scenarios that are different from the ones that they've been trained for



As mentioned, neural networks can seem very confused and place high confidence in decisions a human wouldn't make. The solution to this is to build systems that are uncertainty aware - systems that can recognize where they could be confused.

Sources of Uncertainty

There are 2 reasons why a model can be uncertain:

1. There's noise in the data (Aleatoric Uncertainty)
2. Asking about points very different from the ones in the training set (Epistemic Uncertainty)

Javier goes further in explaining what uncertainty looks like in a detailed example at 12m 53s.

Most existing ML models have ways of expressing noise uncertainty: the probabilities outputted by a regular NN's softmax activation layer represent noise uncertainty. However, it is much more difficult to capture epistemic or "model uncertainty". This latter form of uncertainty captures the notion that the data that the model has seen during training was insufficient for it to make a confident prediction about the test point with which it is being queried.

In practice, the more complex the model, the harder it is to capture epistemic uncertainty. Neural networks are very complicated which makes it nearly impossible to quantify what these models "don't know". Instead ML practitioners resort to approximations, like MC dropout. There is often a tradeoff between using simpler approximations which are more unreliable but scale to big networks and using more exact approximations that provide better uncertainty estimates.

Uncertainty in Practice

Here are some applications of how uncertainty is used in practice today:

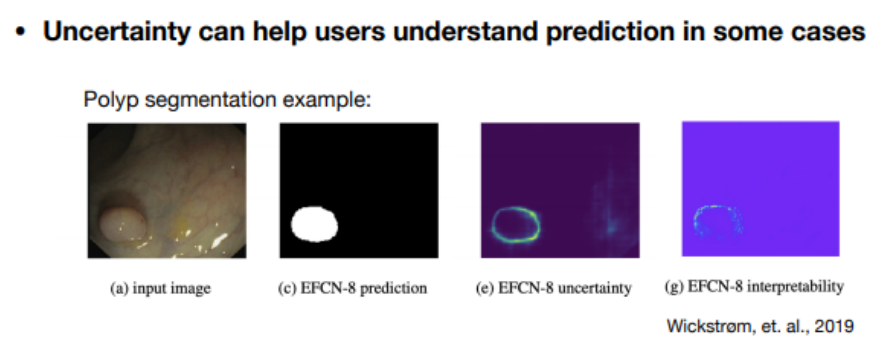
- Robustness: no one would want self-driving cars or medical diagnoses to behave erratically when faced with something unfamiliar (One way to handle a situation like this would be to reject anything unfamiliar and ask for the intervention of a human driver or doctor, respectively)
- Dataset building, safety, and awareness: if there's a model with uncertainty awareness, it can identify uncertainty in certain groups of data and reevaluate potential biases.
- Active learning: this is where there aren't many labels but getting them is really expensive. In these situations, we want to identify the points which will be most useful to train our model. It turns out that the most telling points are the points with high estimates of epistemic uncertainty.

To play around with these ideas and implement your own Bayesian neural network, you can go to the GitHub repo here.

Are Models that are Aware of Uncertainty Interpretable?

Uncertainty-aware models are interpretable as regular ML models. If using linear regression as mentioned before, the model will be very interpretable. On the other hand, if using something like a Bayesian neural network, it probably won't be interpretable.

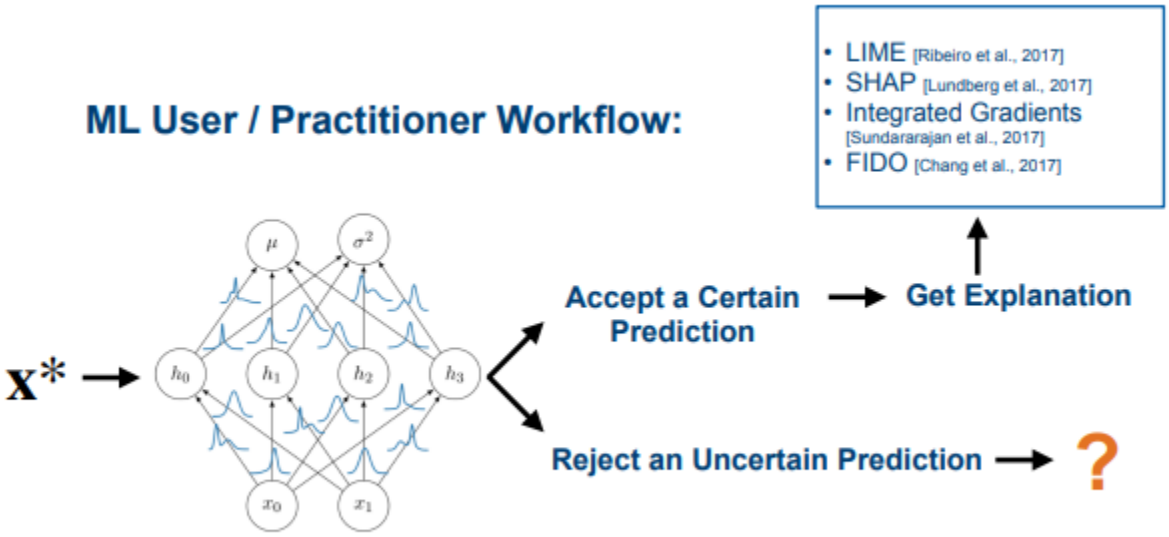
However, uncertainty can also be used as a tool to help users understand their models' predictions. Javier gives an example of a neural network identifying polyps in an image found at 19m 50s. Here, the Bayesian NN's uncertainty around the detected region is more telling than the output from an interpretability technique (LIME).



Identifying polyps

What Happens when a Model Doesn't Know the Answer?

Until now, we have just considered interpreting ML predictions where our model is confident. However, interpretability may be needed the most when this is not the case.

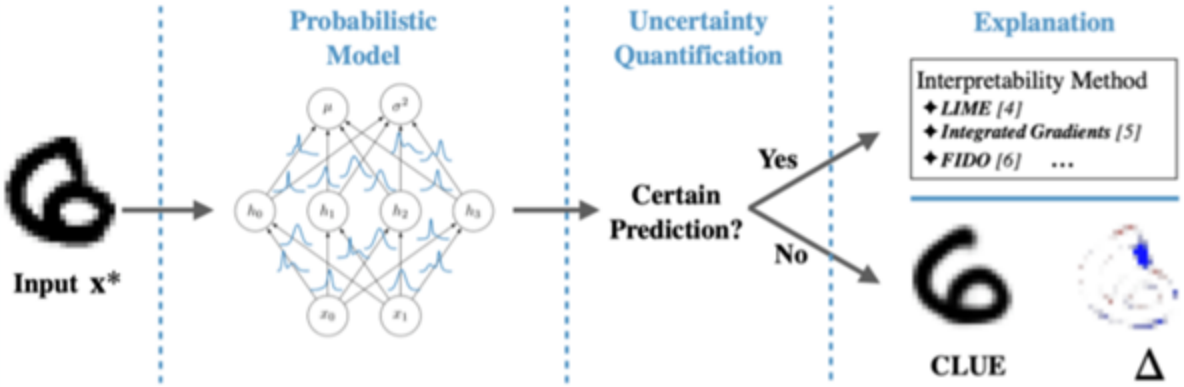


If a model decides to reject an option and doesn't know what to predict: what happens next?

CLUE: Counterfactual Latent Uncertainty Explanations

CLUE is a method that answers the question "what is the smallest change we need to make to an input, while staying in-distribution, such that our model produces more certain predictions?".

How CLUE would be used in practice is given by the following example:

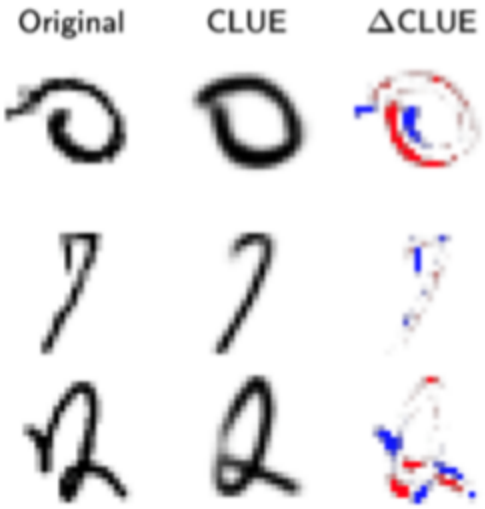


Given an uncertain image (here the ambiguous image could be a ‘6’ or ‘8’), the reject option can be triggered. Then, CLUE would run and give the nearest certain image. In this example, CLUE takes out a line in the original image and now identifies the image as a ‘6’ with confidence, highlighting what the issue was.

Javier goes further into explaining the algorithm at 27 m.

Presenting CLUE to Users

CLUE is able to show the difference in the original input and its own prediction. Basically, what changed in the original input to make it look like CLUE’s output. In the case of handwritten digits, CLUE shows which strokes can be erased to create a more certain prediction.



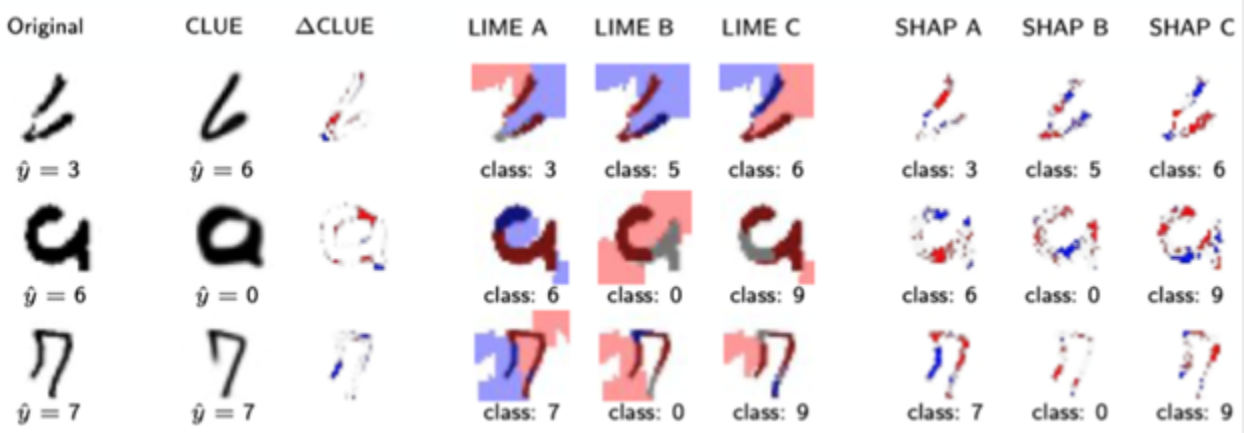
(a) MNIST

CLUE can also be applied to classification, regression, and image data.

CLUE vs Feature Importance (LIME/SHAP)

CLUE's counterfactual nature allows it to add more information than it starts with from the original input. CLUE can point out pieces (like different strokes in drawn numbers) that can be helpful or harmful to prediction. This is in contrast with feature importance techniques, like LIME or SHAP, which simply highlight different features if they provide evidence for or against a class, incapable of adding new information.

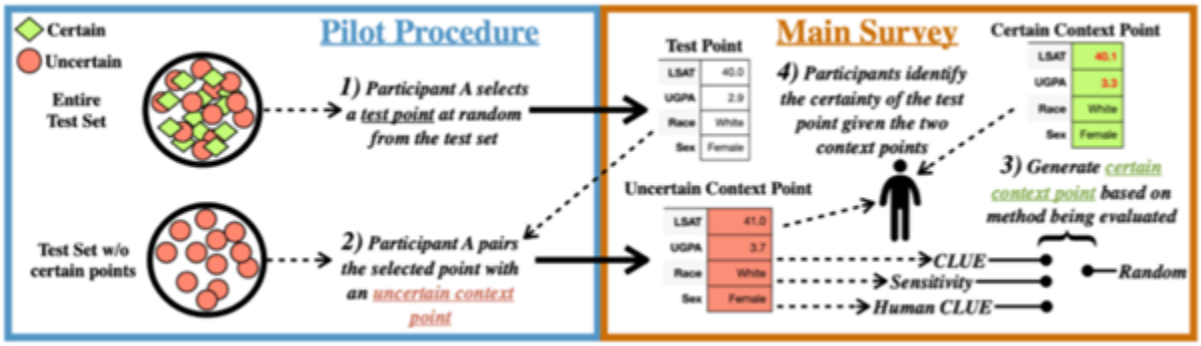
Javier goes over a great example where each of the 3 methods are used to interpret a model's predictions for some uncertain inputs. Here, traditional approaches to interpretability struggle due to a lack of strong evidence for any class or conflicting evidence for multiple classes. A counterfactual approach allows CLUE to give the most useful observations at 29m 41s.



User Study to Show the Value of Uncertainty Estimates

The authors of CLUE wanted to test and see if any of this information on uncertainty is useful and actually make a difference to practitioners. A human simulability task was set up with the idea that an explanation should give a user enough insight to be able to predict how a model is going to behave in an unforeseen circumstance.

The setup can be found at 31m 10s. There are 4 approaches (groups) tested: CLUE, Random (randomly select points), Sensitivity (similar to CLUE but doesn't take the step of restricting the hypothesis space of explanations to plausible inputs), and Human (actual people selecting relevant counterfactual explanations). Each approach has 10 "users" that will be shown its explanations. Each group is given some uncertain inputs and their explanations, and then asked to predict if their model will be certain or uncertain on some new inputs.



The results show that users provided with CLUEs did significantly better, with an accuracy of 82.22% over the average accuracy from the other approaches of 58.89%.

User Study: Results

Method	N. participants	Accuracy (%)
Random	10	61.67
Sensitivity	10	52.78
Human	10	62.22
CLUE	10	**82.22**

CLUE's improvement over all other approaches is statistically significant

(Using Nemenyi test for average ranks across test questions)

A similar experiment was held using images and can be found at 35m 36s.

Conclusion

Feature importance methods such as LIME have been valuable in interpretability, but as demonstrated, has its drawbacks such as an inability to represent extremely non-linear functions. An alternative to feature importance methods are counterfactual explanations that aim to tell how a decision was to change if some input factor were different.

Recently, the field of uncertainty aware NNs has gained a lot of traction. Uncertainty awareness opens up a whole new family of questions relating to ML interpretability which traditional methods are not well suited to answer. CLUE represents a first step towards reconciling ML interpretability and uncertainty. This method has shown success in helping users identify points for which their model is uncertainty and shows great potential in the growing field of interpretability.

Cool Stuff to Check Out

Paper: Getting a CLUE: A Method for Explaining Uncertainty Estimates
Javier's GitHub repo on Bayesian Neural Networks
Towards A Rigorous Science of Interpretable Machine Learning by Finale Doshi-Velez and Been Kim

What's a Counterfactual?

Interesting questions from the video:

- Can you talk a little bit about how CLUE compares to other explainability techniques and what gives it an advantage over others? 38m 38s
- Could counterfactuals be used to understand the effect of missing data and maybe optimal replacement missing data? 45m 16s
- Can you speak a little more on determining distribution data on CLUE? 50m 03s

Video: Beyond Feature Importance: Explaining Uncertainty Estimates
Slides: Rsqrd AI - ML Interpretability: Beyond Feature Importance
All information and ideas presented in this post are that of the speaker and the talk.