A survey about Explainable AI

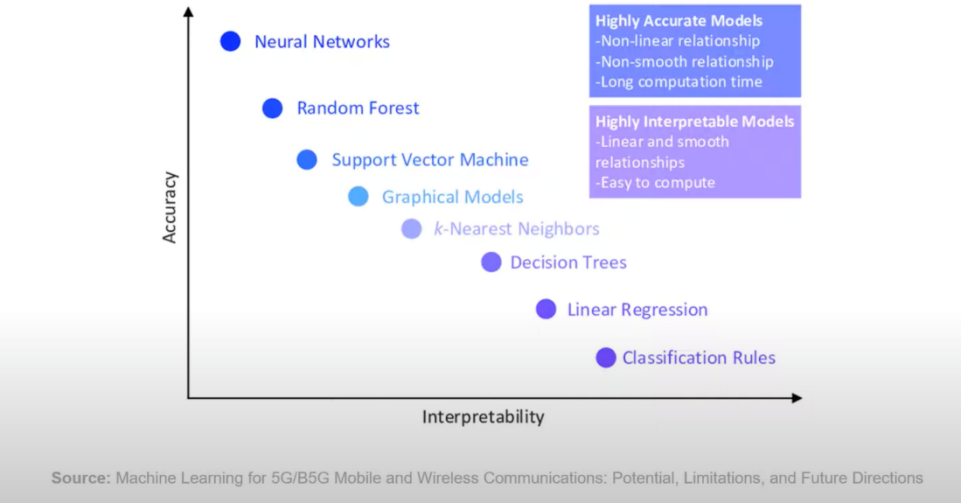
1 Definition

Interpretability is the degree to which a human can **consistently predict** the model's result.

2 Importance

2.1 Premise

Precise and interpretability are two **mutually exclusive directions**. For a problem, white-box models like machine learning methods (linear regression or decision tree) are easy and direct to understand, however, may not perform well under some tricky situations. On the other hand, black-box models like neural network or random forest are better at dealing with hard questions, but their interpretability definitely drop a lot.



As a result, accuracy and interpretability cannot be achieved to a high degree both. It is actually a tradeoff between these, and **explainable AI is to explore this balance boundary**.

2.2 Why needs interpretability

- Human curiosity and learning: find meaning about the model
- From qualitative to quantitative: goal of science is to gain knowledge
- Real world requires safety measures and testing: e.g., self-driving car
- Eliminate biases from the training data: DL models are likely to learn the biases from the dataset
-

3 Taxonomy of interpretability methods

Methods for XAI (Explainable AI) can be classified according to various criteria.

3.1 Intrinsic or post-hos

- Build interpretable ML models
- Derived explanations for complex ML models

3.2 Agnosticity

- Model-agnostic: applicable to all model types
- Model-specific: only applicable to a specific model type

3.3 Scope

- Global explanation
- Local explanation

3.4 Data types

- Tabular
- Image
- Text / speech

3.5 Explanation types

- Visualization
- Feature importance
- Data points
- Surrogate models

4 Interpretable ML models

Linear regression, logistic regression, decision tree, RuleFit, Naïve Bayes classifier, KNN, etc.

5 Model-agnostic methods

The desirable result of the model-agnostic interpretability. Detailed steps are shown in the image below.

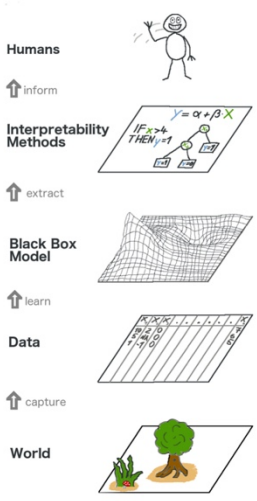


FIGURE 5.1 The big picture of explainable machine learning. The real world goes through many layers before it reaches the human in the form of explanations.

Model-agnostic interpretation methods can be further distinguished into **local and global** methods. Global methods describe how features affect the prediction **on average**. In contrast, local methods aim to explain **individual predictions**.

The list of the model-agnostic methods is shown below:

Explanation Method	Scope	Result
Partial Dependence Plot (PDP) [2]	Global	Feature summary
Individual Condition Expectation [3]	Global/Local	Feature summary
Accumulated Local Effects Plot [4]	Global	Feature summary
Feature Interaction [5]	Global	Feature summary
Feature Importance [6]	Global/Local	Feature summary
Local Surrogate Model [7]	Local	Surrogate interpretable model
Shapley Values (SHAP) [8]	Local	Feature summary
Local Interpretable Model-agnostic Explanations （LIME) [9]	Local	Feature summary
Individual Conditional Expectation (ICE)	Local	Feature summary
BreakDown [10]	Local	Feature summary
Anchors [11]	Local	Feature summary
Counterfactual Explanations [12]	Local	(new) Data point
Prototypes and Criticisms [13]	Global	(existent) Data point
Influence Functions [14]	Global/Local	(existent) Data point

5.1 Global model-agnostic methods

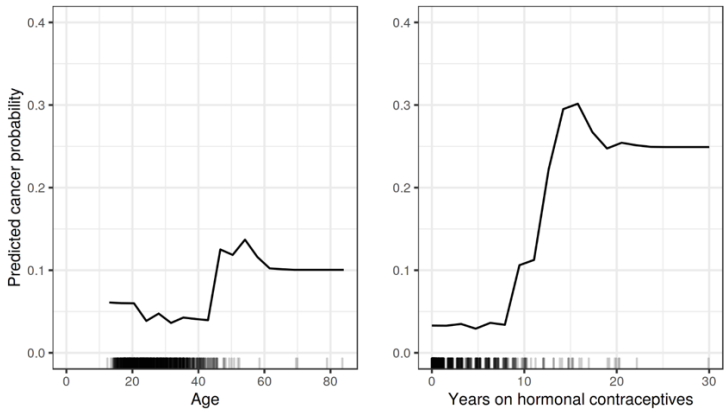
5.1.1 Partial Dependence Plot (PDP)

Actually, it is a very naïve and intuitive method. A partial dependence plot can show **one or two** features of marginal effect on the predicted outcome. PDP demonstrates whether the relationship between the target and a feature is linear, monotonic or more complex.

$$\overline{f_{xs}(x_s)} = \frac{1}{n}\sum_{i=1}^{n} \overline{f_{xs}(x_s, x_C^{(i)})}$$

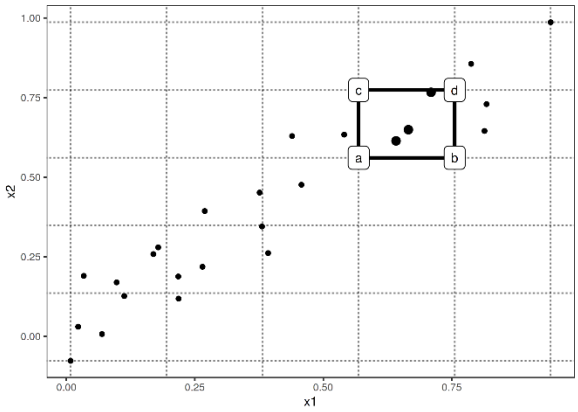
where $\overline{f_{xs}(x_s)}$ means averages of the prediction under x_s, x_s refers to the interested feature and x_C are the rest features that we disinterested.

The visualization result is like the followed.



5.1.2 Accumulated Local Effects Plot (ALE)

Similar to the PDP, they reduce the complex prediction function f to a function that depends on only one or two features. However, ALE plots average the changes in the predictions and accumulate them **over the grid**.



5.2 Local model-agnostic methods

5.2.1 LIME

5.2.1.1 Overview

In the paper[1], author proposes a concrete implementation of **local surrogate models** (like a Linear Regression model) to approximate the predictions of the underlying black box model.

The learned model should be a good approximation of the machine learning model predictions locally, but it does not have to be a good global approximation. This kind of accuracy is also called **local fidelity**.

[1] Ribeiro, Marco Tulio, Sameer Singh, and Carlos Guestrin. "" Why should I trust you?" Explaining the predictions of any classifier." *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 2016.(https://arxiv.org/abs/1602.04938v3)

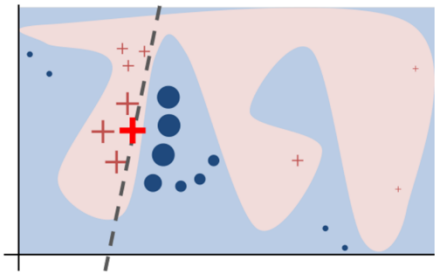


Figure 3: Toy example to present intuition for LIME. The black-box model's complex decision function f (unknown to LIME) is represented by the blue/pink background, which cannot be approximated well by a linear model. The bold red cross is the instance being explained. LIME samples instances, gets predictions using f, and weighs them by the proximity to the instance being explained (represented here by size). The dashed line is the learned explanation that is locally (but not globally) faithful.

Steps can be described as follow:

1. Consider the individual input
2. Give small perturbation to the input, so can generate a huge partially similar but different data
3. Use these new datasets to train a interpretability model (listed in the chapter 4)

After these several steps, the interpretability model can explain why some outputs are different from others given to partially similar but different inputs.

Mathematically, local surrogate models with interpretability constraint can be expressed as follows:

$$explanation(x) = \arg min\, L(f, g, \pi_x) + \Omega(g)$$

f refers to black-box model, g refers to the white-box model. π_x measures how large the neighborhood around instance x. $\Omega(g)$ is the hyper-parameters to be defined by practice in order to make the explainable model simple enough.

5.2.1.2 Implementation scope

For text and images, the solution is to **turn single words or super-pixels on or off**. In the case of tabular data, LIME creates new samples by **perturbing each feature individually**, drawing from a normal distribution with mean and standard deviation taken from the feature.

- Turn single words: randomly removing words from the original text
- Super-pixels: segmenting the image into super-pixels and turn it on-or-off

5.2.1.3 Disadvantages

- Definition of **neighborhood scale**
- **Instability** of the explanations: tow very close points varied greatly in a simulated setting[2]

5.2.1.4 Conclusion

Local surrogate models, with LIME as a concrete implementation, are very promising. But the method is still in development phase and many problems need to be solved before it can be safely applied.

---

[2] Alvarez-Melis, David, and Tommi S. Jaakkola. "On the robustness of interpretability methods." *arXiv preprint arXiv:1806.08049* (2018).(https://arxiv.org/pdf/1806.08049.pdf)

5.2.2 Shapley Additive exPlanations (SHAP)

5.2.2.1 Overview

In the paper[3], authors propose a method to explain individual predictions, which is based on the game theoretically optimal **Shapley Values**. SHAP has a **solid theoretical foundation** in game theory. The prediction is **fairly distributed** among the feature values. We get **contrastive explanations** that compare the prediction with the average prediction.

The Shapley value is the **average marginal contribution** of a feature value across all possible coalitions. For example, the interpretation of the Shapley value for feature value j is: The value of the j-th feature contributed Φ_j to the prediction of this particular instance compared to the average prediction for the dataset.

In specific, the Φ_j can be calculated as followed:

$$\Phi_j(f,x) = \sum_{z' \subseteq x'} \frac{|z'|!\,(M-|z'|-1)!}{M!} * (f_x(z') - f_x(z' \backslash j))$$

where z' is a subset of the features used in the model, j is the vector of feature values of the instance to be explained and M the number of features. $f_x(z')$ and $f_x(z' \backslash j)$ is the prediction for feature values in set z' that are marginalized over features that are not included in set z'.

Additionally, when calculating different permutation, we can use some methods to **decrease the complex computation** like kernel SHAP, Tree SHAP, etc.

Last but not least, with the help of tree-based implementation, SHAP can also achieve the **global interpretability**, which include feature importance, feature dependence, interactions, clustering and summary plots.

5.2.2.2 Implementation scope

Like the LIME, for image using the super-pixels, for text using randomly choose word permutation.

5.2.2.3 Conclusion

There are various improvements based on the fundamental Shapely values, which to some extent solve problems like slow compotation or feature dependence partially. Despite these, it is still a relatively fair method to assign the prediction to individual features.

5.2.3 Counterfactual explanations

Unlike the above two that are attribution methods, counterfactual explanations[4] are **example-based** one. Counterfactual explanations explain a prediction by examining **which features would need to be changed** to achieve a desired prediction.

The objective function can be described as followed:

$$\arg\min distance(x, x'), \qquad while\ f(x) = C, \qquad s.t.\ f(x') = C'$$

---

[3] Lundberg, Scott M., and Su-In Lee. "A unified approach to interpreting model predictions." *Advances in neural information processing systems* 30 (2017). (https://proceedings.neurips.cc/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf)

[4] For the detail, can check here: https://christophm.github.io/interpretable-ml-book/counterfactual.html

where x' refers to the minimized modification of the original input x that can make the prediction to the other side.

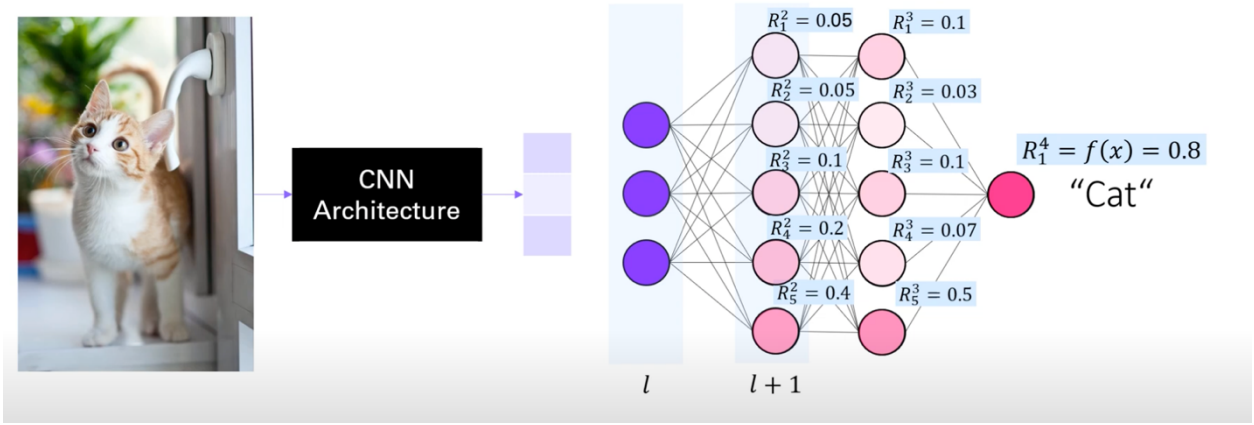
6 Neural network interpretation

For DNN, most methods are **model-specific**. In CV field, normal methods are like guided backpropagation, integrated gradients, SmoothGrad saliency maps, Grad-CAM, Concept activation vectors. Others are like knowledge distillation, dimensionality reduction and tree regularization.

6.1 Layer-wise relevance propagation (LRP)

It evaluates **each layer neurons** contribute to the maximum predicted outcome. The formula for calculating the importance of each layer's neurons is as followed:

$$R_i^{(l)} = \sum_j \frac{z_{ij}}{\sum_{i'} z_{i'j}} R_i^{(l+1)}, \quad with\ z_{ij} = x_i^{(l)} w_{ij}^{(l,l+1)}$$



6.2 Pixel attribution (saliency maps)

Pixel attribution is a special case of **feature attribution**, but most for images. Feature attribution explains individual predictions by attributing each input feature according to how much it changed the prediction (either positively or negatively).

There is a confusing amount of pixel attribution approaches and mostly there are tow different types of attribution methods:

- Occlusion- or perturbation-based: methods like SHAP and LIME manipulate **parts** of the image to generate explanations
- Gradient-based: the explanation has the **same size as the input image** (or at least can be meaningfully projected onto the original image) and they assign **each pixel** a value that can be interpreted as the relevance of the pixel to the prediction or classification of that image. Examples of gradient methods are like Vanilla Gradient and Grad-CAM.

6.2.1 Vanilla gradient

Algorithm steps of Vanilla Gradient[5]:

1. Forward pass with data
2. Backward pass to input layer to get the gradient
3. Render the gradient as a normalized heatmap

Backpropagation normally stops at the second layer during training for efficiency as input cannot be changed. Crucially, however, Vanilla Gradient continues to **backprop to the input layer** to see which pixels would affect our output the most. The backpropagation step here gives us good saliency clues because it calculates the gradient of the given output class with respect to the input image. The gradient is just a list of derivatives, one for each pixel.

Left: original image. Center: saliency map blended with original image. Right: saliency map.

The red parts mean the positive influence and blue one refers to the negative.

Also, there is something needed to be considered about color render. The original Vanilla Gradient paper used a white-spectrum colormap, while the above picture use a red-white-blue colormap. The advantage of a diverging colormap such as red-white-blue is that we can better capture the difference between positive and negative values. This is useful in white-digit-on-black-background MNIST, as positive derivatives indicate positive probability impact (and vice versa). However, in ImageNet it turns out that the implication of signage is context-dependent, so researchers have found the absolute value of the gradient and sequential color maps like white-spectrum to be most clear.

6.2.2 Grad-CAM

Grad-CAM[6] provides visual explanations for CNN decisions. Unlike other methods, the gradient is not backpropagated all the way back to the image, but (usually) **to the last convolutional layer** to produce a coarse localization map that highlights important regions of the image.

Grad-CAM analyzes which regions are activated in the feature maps of the last convolutional layers. And then, the heatmap is send through the ReLU function so it removes all negative values because we are only interested in the parts contributing to the selected class.

---

[5] Simonyan, Karen, Andrea Vedaldi, and Andrew Zisserman. "Deep inside convolutional networks: Visualising image classification models and saliency maps." arXiv preprint arXiv:1312.6034 (2013). (https://arxiv.org/pdf/1312.6034.pdf)

[6] Selvaraju, Ramprasaath R., et al. "Grad-cam: Visual explanations from deep networks via gradient-based localization." Proceedings of the IEEE international conference on computer vision. 2017. (https://arxiv.org/pdf/1610.02391.pdf)

$$L^c_{Grad-CAM} \in \mathbb{R}^{u \times v} = \underbrace{ReLU}_{\text{Pick positive values}} \left(\sum_k \alpha^c_k A^k \right)$$

where u is the width, v is the height of the explanation and c is the class of interest.

6.2.3 Guided Grad-CAM

From the description of Grad-CAM we can find that the localization is very coarse, since the last convolutional feature maps have a much coarser resolution compared to the input image. In contrast, other attribution techniques backpropagate all the way to the input pixels. They are therefore much more detailed and can show you individual edges or spots that contributed most to a prediction. **A fusion of both methods** is called Guided Grad-CAM. And it is super simple. We compute for an image both the Grad-CAM explanation and the explanation from another attribution method, such as Vanilla Gradient. The Grad-CAM output is then upsampled with bilinear interpolation, then both maps are multiplied element-wise. Grad-CAM works like a lense that focuses on specific parts of the pixel-wise attribution map.

6.2.4 SmoothGrad

The idea of SmoothGrad by Smilkov et al. (2017)[7] is to make gradient-based explanations less noisy by **adding noise and averaging over these artificially noisy gradients**.

$$R_{sg}(x) = \frac{1}{N} \sum_{i=1}^{n} R(x + g_i), \qquad g_i \; is \; noise \; and \; gi \sim N(0, \sigma^2)$$

SmoothGrad is not a standalone explanation method, but an extension to any gradient-based explanation method.

SmoothGrad works in the following way:

1. Generate multiple versions of the image of interest by adding noise to it.
2. Create pixel attribution maps for all images.
3. Average the pixel attribution maps.

6.2.5 Conclusion

Gradient-based methods are much faster to compute than model-agnostic methods and the explanations are visual so that sometimes we can easily recognize images.

However, it may also have some disadvantages. As there are **no ground truth for the explanations**, we even cannot tell whether one visualized image is correct or not. Instead, we can only, in a first step, reject explanations that obviously make no sense.

Also, the saliency methods are highly **unreliable and fragile**, which means that introduce small perturbations or constant shift may lead to different highlighted area as explanation.

Additionally, some methods may be **insensitive to model and data**. These means that the method will always highlight the edge of the object and are unrelated to a prediction model or abstract features of the image, which means does not requires training any more. What's more. The most importance is that we need **a fair evaluation metrics** to scrutinize various methods.

---

[7] Smilkov, Daniel, et al. "Smoothgrad: removing noise by adding noise." arXiv preprint arXiv:1706.03825 (2017). (https://arxiv.org/pdf/1706.03825.pdf)