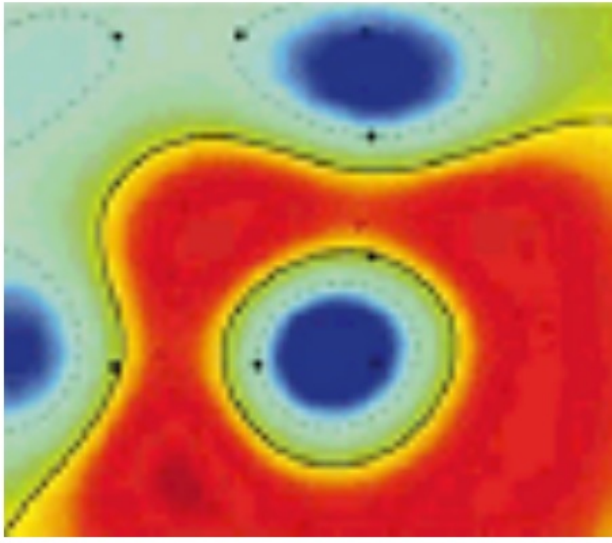
WiSe 2024/25

Machine Learning 1/1-X

Technische Universität Berlin



Lecture 14 | Explainable AI

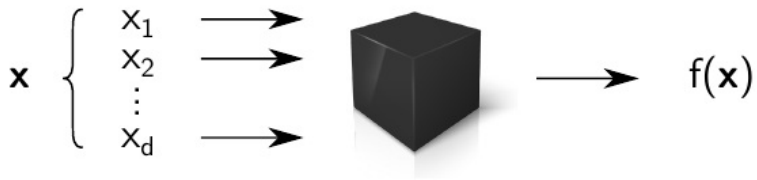
Outline

- What is Explainable AI?
- Desiderata of an Explainable AI technique
- Uses of Explainable AI
- Methods for Explainable AI
  - Activation Maximization
  - Shapley Values
  - Taylor Expansions
  - Layer-wise Relevance Propagation

What is Explainable AI?

Standard machine learning:

- The function f is typically considered to be a "black-box" whose parameters are learned from the data using e.g. gradient descent. The objective to minimize encourages the predictions $f(\boldsymbol{x})$ to coincide with the ground truth on the training and test data.



Machine learning + Explainable AI:

- We do not only look at the outcome $f(\boldsymbol{x})$ of the prediction but also at the way the prediction is produced by the ML model, e.g. which features are used, how these features are combined, or to what input pattern the model responds the most.

What is Explainable AI?

Example 1: Synthesize an input pattern that most strongly activates the output of the ML model associated to a particular class.

Image source: Nguyen et al. (2016) Multifaceted Feature Visualization: Uncovering the Different Types of Features Learned By Each Neuron in Deep Neural Networks

What is Explainable AI?

Example 2: Highlight features that have contributed for a given data point to the ML prediction.

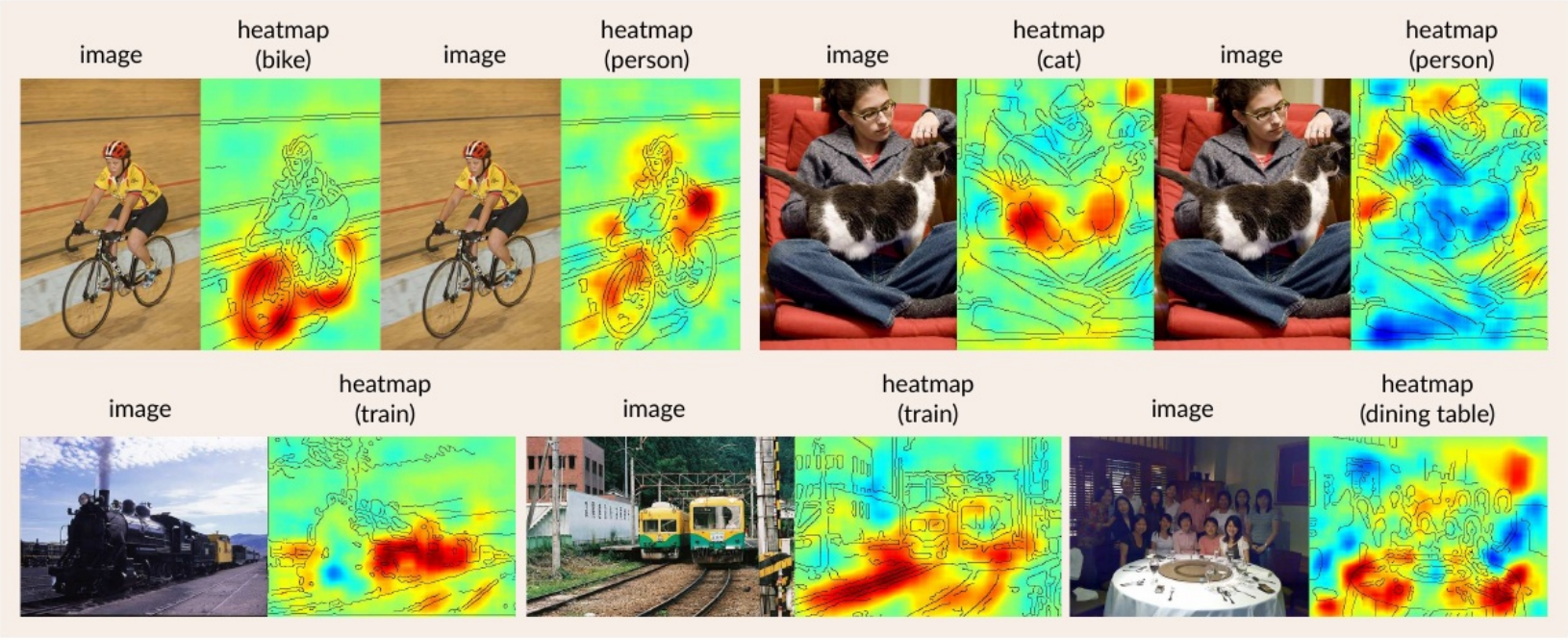
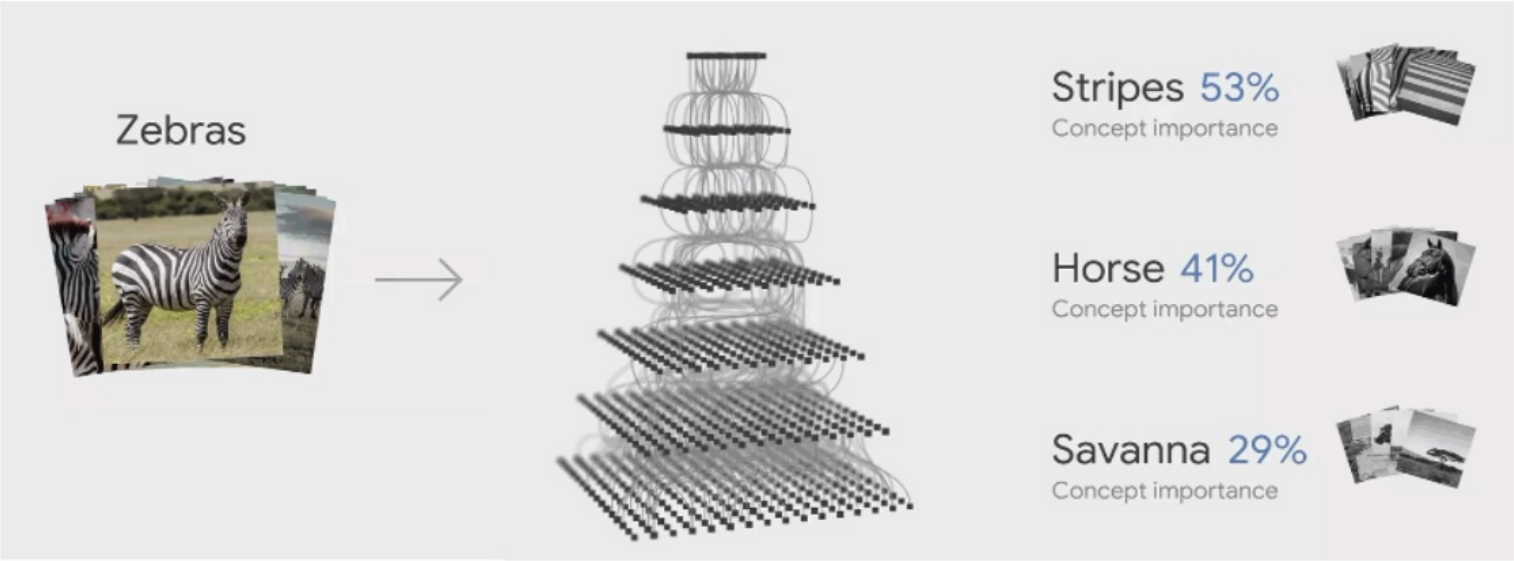


Image source: Lapuschkin et al. (2016) Analyzing Classifiers: Fisher Vectors and Deep Neural Networks

What is Explainable AI?

Example 3: Concept activation vectors (TCAV). Highlight the mid-level concepts that explain, for a given data point, the ML prediction.



Source: Google Keynote'19 (URL: `https://www.youtube.com/watch?v=lyRPyRKHO8M&t=2279s`)

Desiderata of an Explanation

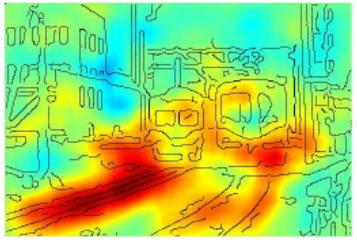
In practice, we would like the explanation technique to satisfy a number of properties:

1. **Fidelity:** The explanation should reflect the quantity being explained and not something else.
2. **Understandability:** The explanation must be easily understandable by its receiver.
3. **Sufficiency:** The explanation should provide sufficient information on how the model came up with its prediction.
4. **Low Overhead:** The explanation should not cause the prediction model to become less accurate or less efficient.
5. **Runtime Efficiency:** Explanations should be computable in reasonable time.

see also *Swartout & Moore (1993), Explanation in Second Generation Expert Systems.*

image

heatmap (train)

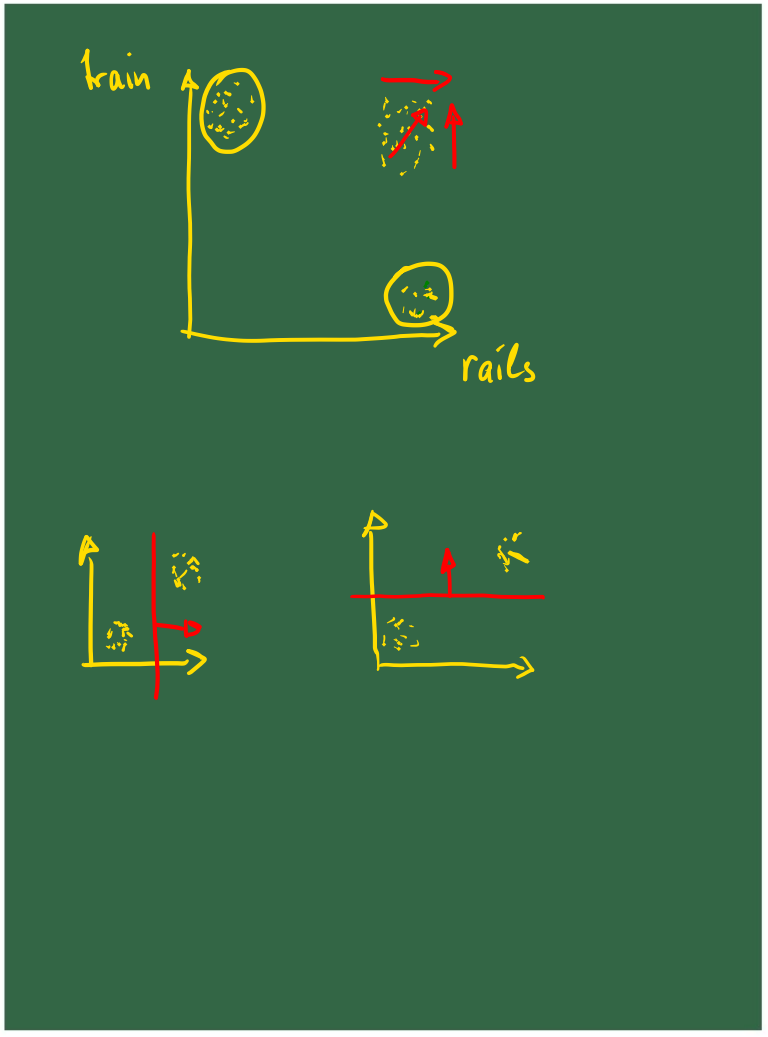


0.0	0.0
0.8	0.2

Uses of an Explanation

Verify (and improve?) a ML model

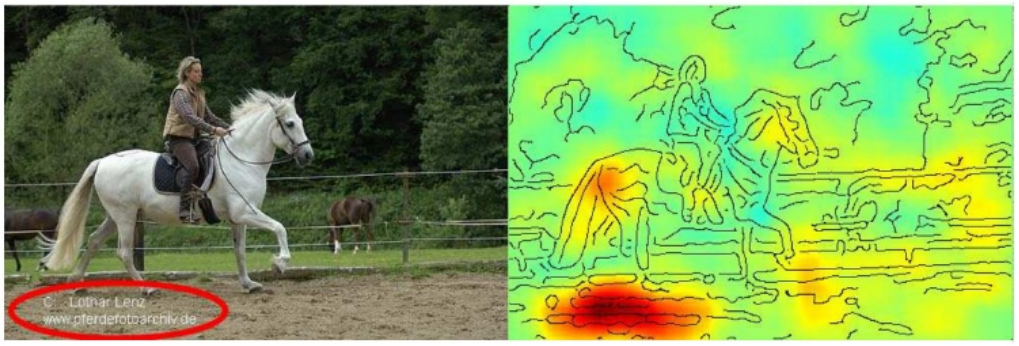
- Verify that the model is based on features which *generalize well* to examples outside the current data distribution (this cannot be done with standard validation techniques!).
- Reliance of the ML models on wrong features is often encountered when there are *spurious correlation* in the data.
- From the explanation, the model's trustworthiness can be reevaluated, and the flawed ML model can be potentially *retrained* based on the user feedback.



Uses of an Explanation

Example: The classifier is right for the wrong reasons

image heatmap (horse)



average precision of the Fisher Vector model on the PascalVOC dataset

aer 79.08	bic 66.44	bir 45.90	boa 70.88	bot 27.64
bus 69.67	car 80.96	cat 59.92	cha 51.92	cow 47.60
din 58.06	dog 42.28	hor 80.45	mot 69.34	per 85.10
pot 28.62	she 49.58	sof 49.31	tra 82.71	tvm 54.33

- In this example, the classifier accurately predicts the horse class, but based on the wrong features (some copyright tag in the corner).
- This incorrect decision strategy cannot be detected by just looking at the test error.

cf. Lapuschkin et al. (2019) Unmasking Clever Hans Predictors and Assessing What Machines Really Learn. Nature Communications

Uses of an Explanation

Learn something about the data
(or about the system that produced the data)

- **Step 1:** Train a ML model that predicts well the data.
- **Step 2:** Apply XAI to the trained ML model to produce explanations of the ML decision strategy.
- **Step 3:** Based on the XAI explanations, the user can compare his reasoning with that of the ML model, and can potentially refine his own domain knowledge.

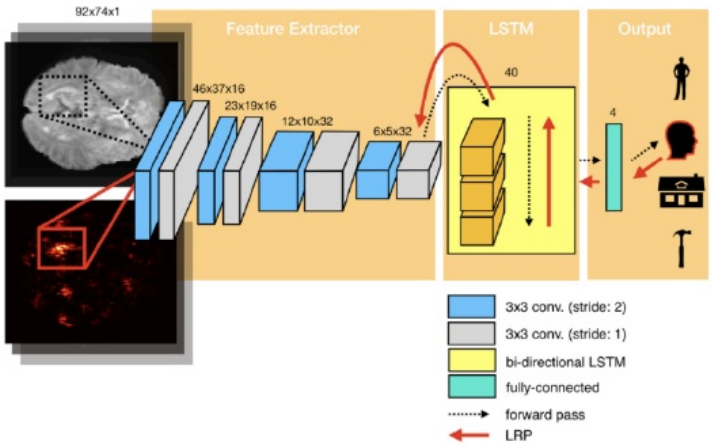


Image source: Thomas et al. (2019) Analyzing Neuroimaging Data Through Recurrent Deep Learning Models

Part II: Methods of XAI

Presented methods

- Activation maximization
- Shapley values
- Taylor expansion
- Layer-wise relevance propagation

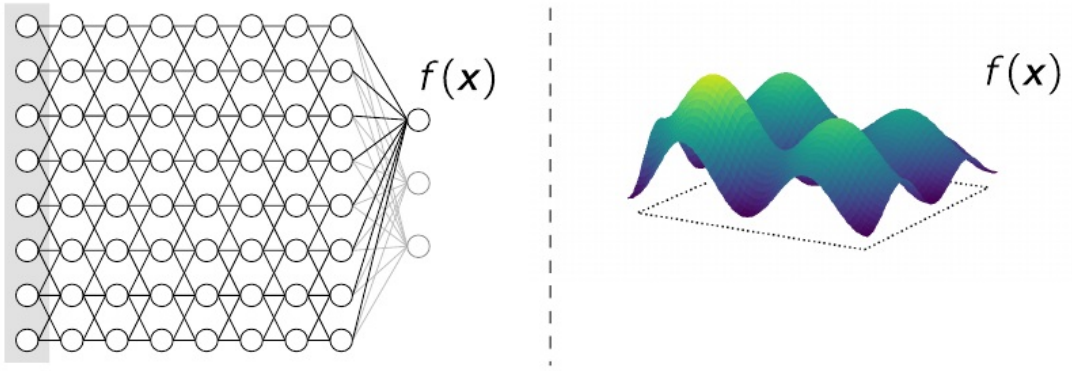
Other methods

- Surrogate models (LIME)
- Integrated gradients / expected gradients / SmoothGrad
- Influence functions
- ...

Activation Maximization

Assume a trained a ML model (e.g. a neural network), and we would like to understand what concept is associated to some particular output neuron of the ML model, e.g. the output neuron that codes for the class 'cat'. Activation maximization proceeds in two steps:

- **Step 1:** Think of the ML model as a function of the input



- **Step 2:** Explain the function f by generating a maximally activating input pattern:

$$\boldsymbol{x}^\star = \arg\max_{\boldsymbol{x}} f(\boldsymbol{x}, \theta)$$

Activation Maximization

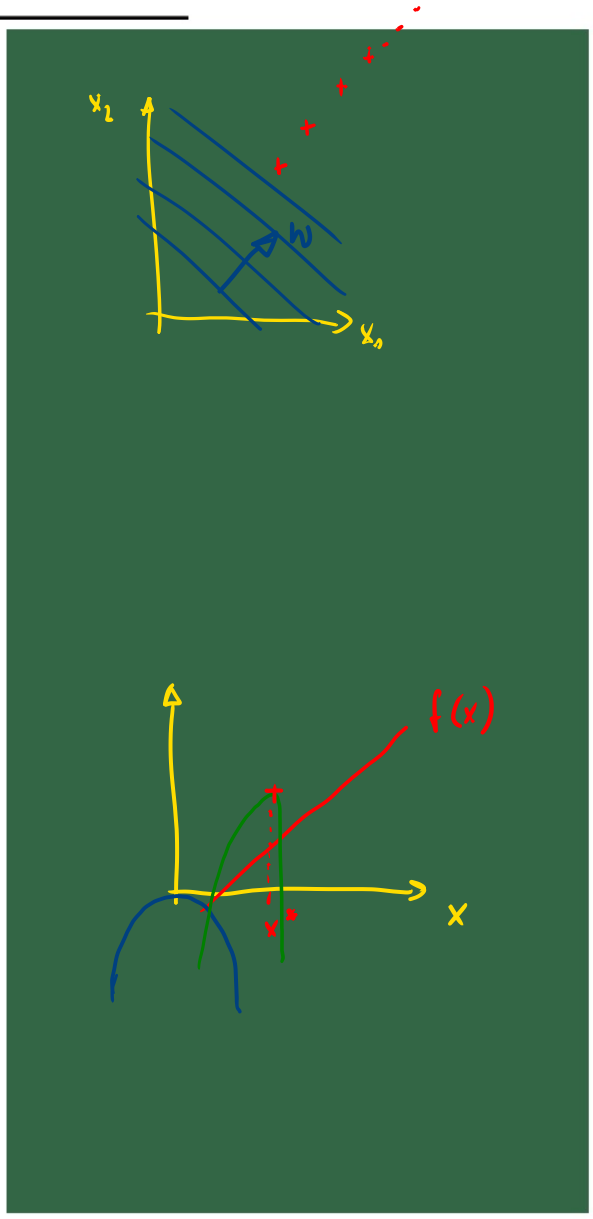
Problem: In most cases $f(\boldsymbol{x})$ does not have single point corresponding to the maximum.

> *E.g. in linear models, $f(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b$, we can keep moving the point $\boldsymbol{x}$ further along the direction $\boldsymbol{w}$, and the output continues to grow).*

Therefore, we would like to apply a preference for 'regular' regions of the input domain, i.e.

$$\boldsymbol{x}^\star = \arg\max_{\boldsymbol{x}} f(\boldsymbol{x}) + \Omega(\boldsymbol{x})$$

In practice, the preference can be for data points with small norm (i.e. we set $\Omega(\boldsymbol{x}) = -\lambda\|\boldsymbol{x}\|^2$ so that points with large norm are penalized.)



Activation Maximization: Examples

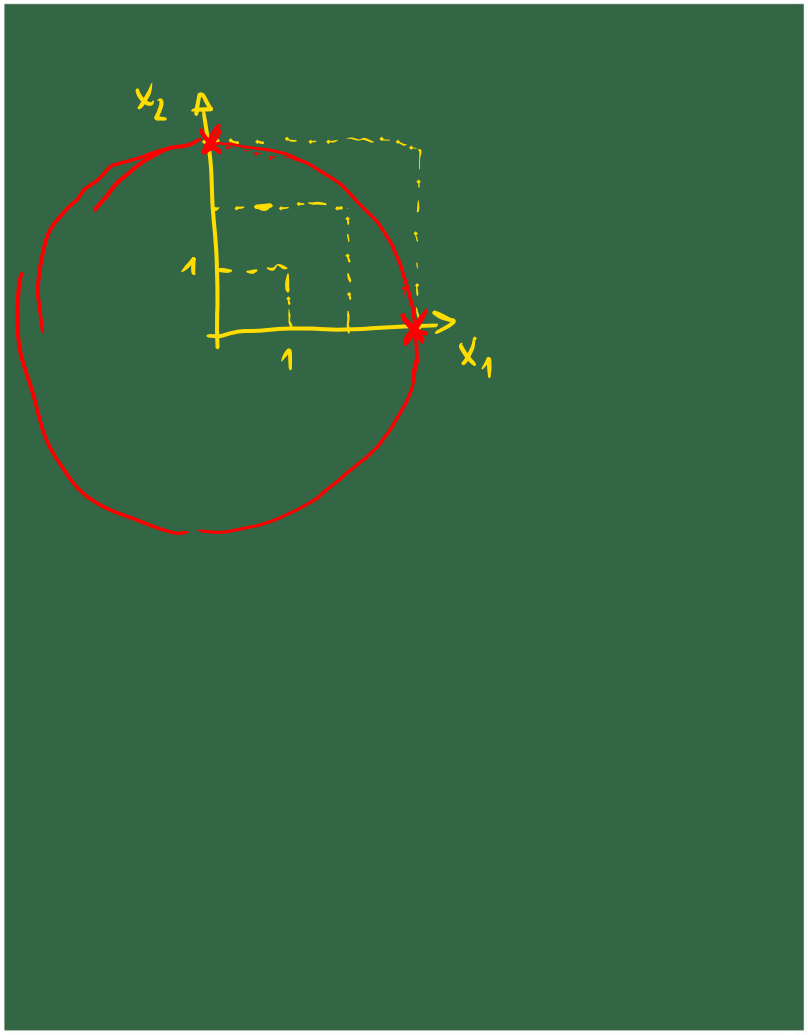
$f(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b$ and
$\Omega(\boldsymbol{x}) = -\lambda \|\boldsymbol{x}\|^2$

$$\underset{x}{\operatorname{argmax}} \underbrace{w^T x + b - \lambda \|x\|^2}_{J(x)}$$

$$\frac{\partial J}{\partial x} = w - 2\lambda x \overset{!}{=} 0$$

$$x^* = \frac{1}{2\lambda} \cdot w$$

$f(\boldsymbol{x}) = \max(x_1, x_2)$ and
$\Omega(\boldsymbol{x}) = -\lambda \|\boldsymbol{x}\|^2$



Activation Maximization: Probability View

Assume the model produces a log-probability for class ω_c:

$$f(\boldsymbol{x}) = \log p(\omega_c|\boldsymbol{x})$$

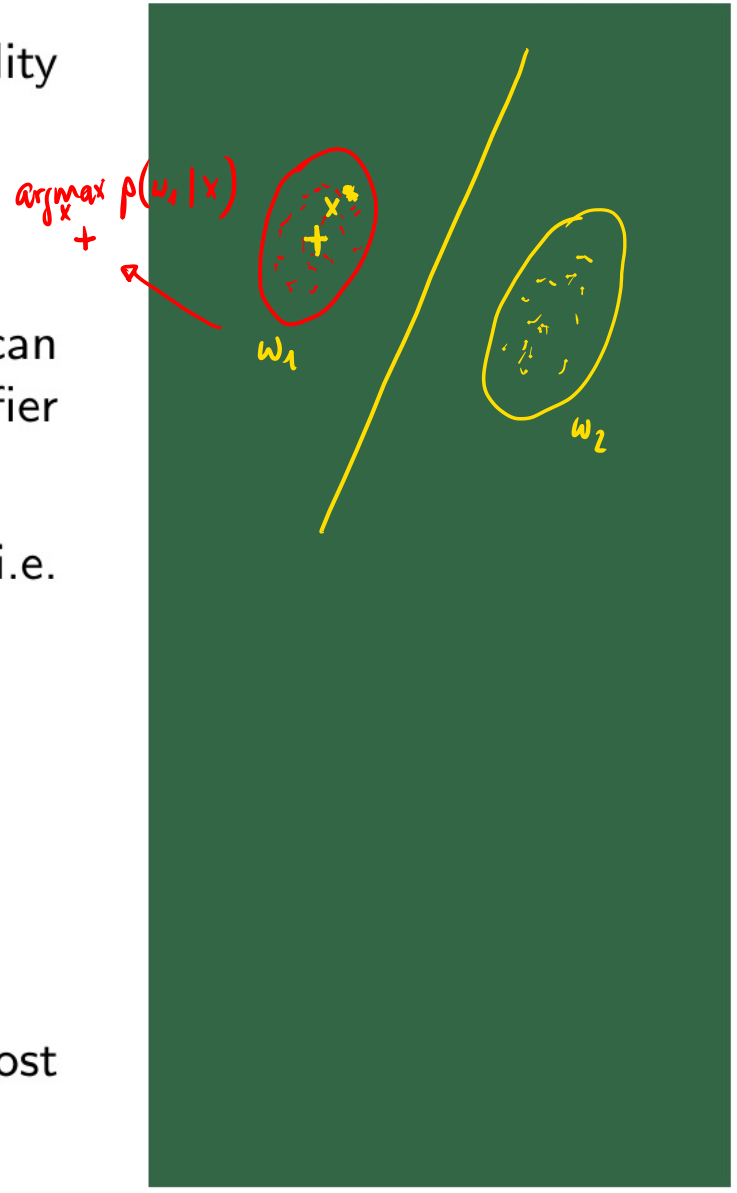
The input $\boldsymbol{x}^\star$ that maximizes this function can be interpreted as the point where the classifier is the most *sure* about class ω_c.

Choose the regularizer $\Omega(\boldsymbol{x}) = \log p(\boldsymbol{x})$, i.e. favor points that are *likely*.

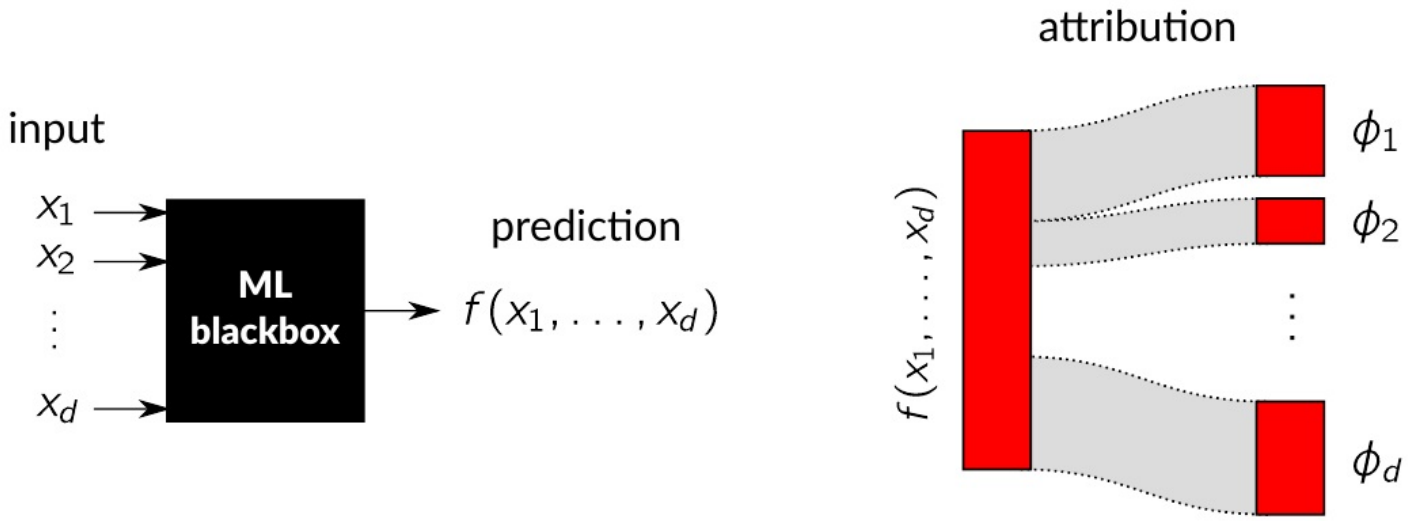
The optimization problem becomes:

$$\begin{aligned}\boldsymbol{x}^\star &= \arg\max_{\boldsymbol{x}} \ \log p(\omega_c|\boldsymbol{x}) + \log p(\boldsymbol{x}) \\ &= \arg\max_{\boldsymbol{x}} \ \log p(\boldsymbol{x}|\omega_c)\end{aligned}$$

where $\boldsymbol{x}^\star$ can now be interpreted as the most *typical* input for class ω_c.



Attribution of a Prediction to Input Features

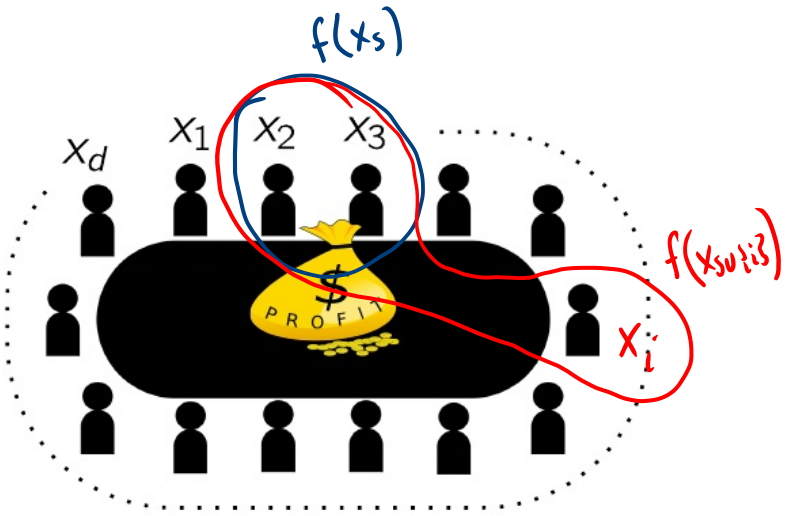


1. The data $\boldsymbol{x} \in \mathbb{R}^d$ is fed to the ML black-box and we get a prediction $f(\boldsymbol{x}) \in \mathbb{R}$.
2. We explain the prediction by determining the contribution of each input feature.

Key property of an explanation: **conservation** ($\sum_{i=1}^{d} \phi_i = f(\boldsymbol{x})$).

Attribution: Shapley Values

- Framework originally proposed in the context of game theory (Shapley 1951) for assigning payoffs in a cooperative game, and recently applied to ML models.
- Each input variable is viewed as a player, and the function output as the profit realized by the cooperating players.



The Shapley values $\phi_1, \ldots, \phi_d$ measuring the contribution of each feature are:

$$\phi_i = \sum_{\mathcal{S}\,:\, i \notin \mathcal{S}} \frac{|\mathcal{S}|!(d-|\mathcal{S}|-1)!}{d!} \left[f(\boldsymbol{x}_{\mathcal{S}\cup\{i\}}) - f(\boldsymbol{x}_{\mathcal{S}}) \right]$$

where $(\boldsymbol{x}_{\mathcal{S}})_{\mathcal{S}}$ are all possible subsets of features contained in the input $\boldsymbol{x}$.

Attribution: Shapley Values

Recall: $$\phi_i = \sum_{\mathcal{S}:\, i \notin \mathcal{S}} \underbrace{\frac{|\mathcal{S}|!(d-|\mathcal{S}|-1)!}{d!}}_{\alpha_{\mathcal{S}}} \underbrace{\left[f(\boldsymbol{x}_{\mathcal{S}\cup\{i\}}) - f(\boldsymbol{x}_{\mathcal{S}})\right]}_{\Delta_{\mathcal{S}}}$$

$1 \cdot (1+1) = 2$

$0 \cdot (1+1) = 0$

Worked-through example: Consider the function $f(\boldsymbol{x}) = x_1 \cdot (x_2 + x_3)$. Calculate the contribution of each feature to the prediction $f(\mathbf{1}) = 1 \cdot (1+1) = 2$.

ϕ_1:

S	α_S	Δ_S
$\{\}$	$1/3$	0
$\{2\}$	$1/6$	1
$\{3\}$	$1/6$	1
$\{2,3\}$	$1/3$	2

$\phi_1 = 1$

ϕ_2:

S	α_S	Δ_S
$\{\}$	$1/3$	0
$\{1\}$	$1/6$	1
$\{3\}$	$1/6$	0
$\{1,3\}$	$1/3$	1

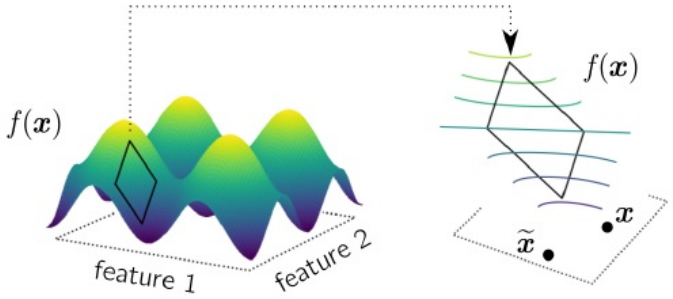
$\phi_2 = 1/2$

ϕ_3: symmetry / conservation

$\phi_3 = 1/2$

Attribution: Taylor Expansions

- Many ML models $f(\boldsymbol{x})$ are complex and nonlinear when taken globally but are simple and linear when taken locally.



- The function can be approximated locally by some Taylor expansion:

$$f(\boldsymbol{x}) = \underbrace{f(\widetilde{\boldsymbol{x}})}_{=0} + \sum_{i=1}^{d} \underbrace{[\nabla f(\widetilde{\boldsymbol{x}})]_i \cdot (x_i - \widetilde{x}_i)}_{\phi_i} + \dots$$

- First-order terms ϕ_i of the expansion can serve as an explanation.
- The explanation $(\phi_i)_i$ depends on the choice of root point $\widetilde{\boldsymbol{x}}$.

Attribution: Taylor Expansions

Example: Attribute the prediction $f(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x}$ with $\boldsymbol{x} \in \mathbb{R}^d$ on the d input features.

$\tilde{x} = 0$

$$f(x) = \underbrace{f(\tilde{x})}_{=0} + \sum_{i=1}^{d} \underbrace{\underbrace{[\nabla f(\tilde{x})]_i}_{w_i} \cdot \underbrace{(x_i - \tilde{x}_i)}_{x_i}}_{\phi_i = w_i x_i}$$

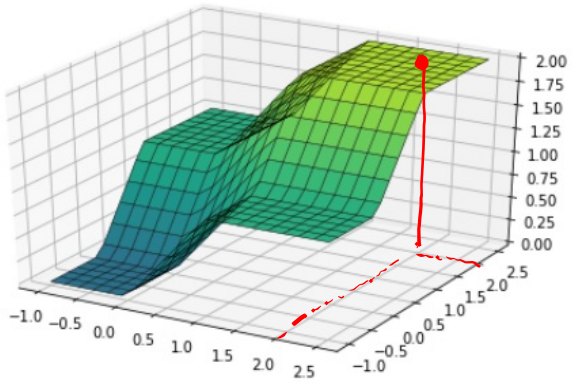
Attribution: Taylor Expansions

Limitations: Gradient information is too localized.

- Cannot handle *saturation effects* and *discontinuities* e.g. cannot explain the function

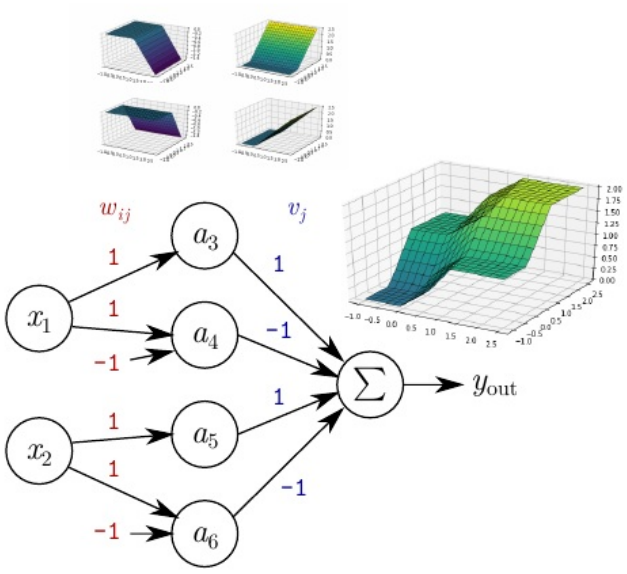
$$f(\boldsymbol{x}) = \sum_{i=1}^{d} x_i - \max(0, x_i - \theta)$$

at the point $\boldsymbol{x} = (2, 2)$.



This limitation can be overcome by looking at the structure of the model and decompose the problem of explanation in multiple parts (→ next slide).

Attribution: Look at the Structure of the Model



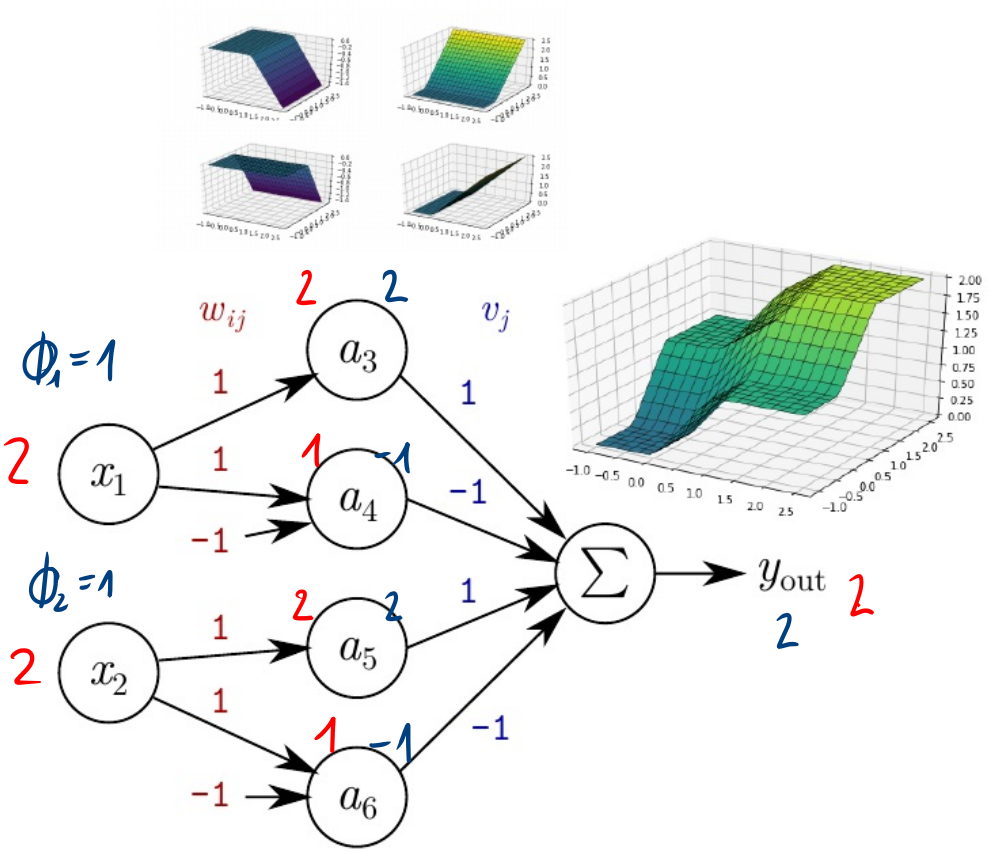
Observation:

- The function implemented by a ML model is typically a composition of simple elementary functions.
- These functions are simpler to analyze than the whole input-output function.

Idea:

- Treat the problem of explanation as propagating the prediction backward in the input-output graph.
- The layer-wise relevance propagation (LRP) method implements this approach and can be used to explain ML models ($\rightarrow$ next slide).

Attribution: The LRP Method



Example: Consider y_{out} to be the quantity to explain:

$$R_{\text{out}} \leftarrow y_{\text{out}}$$

- **Step 1:** Propagate on the hidden layer

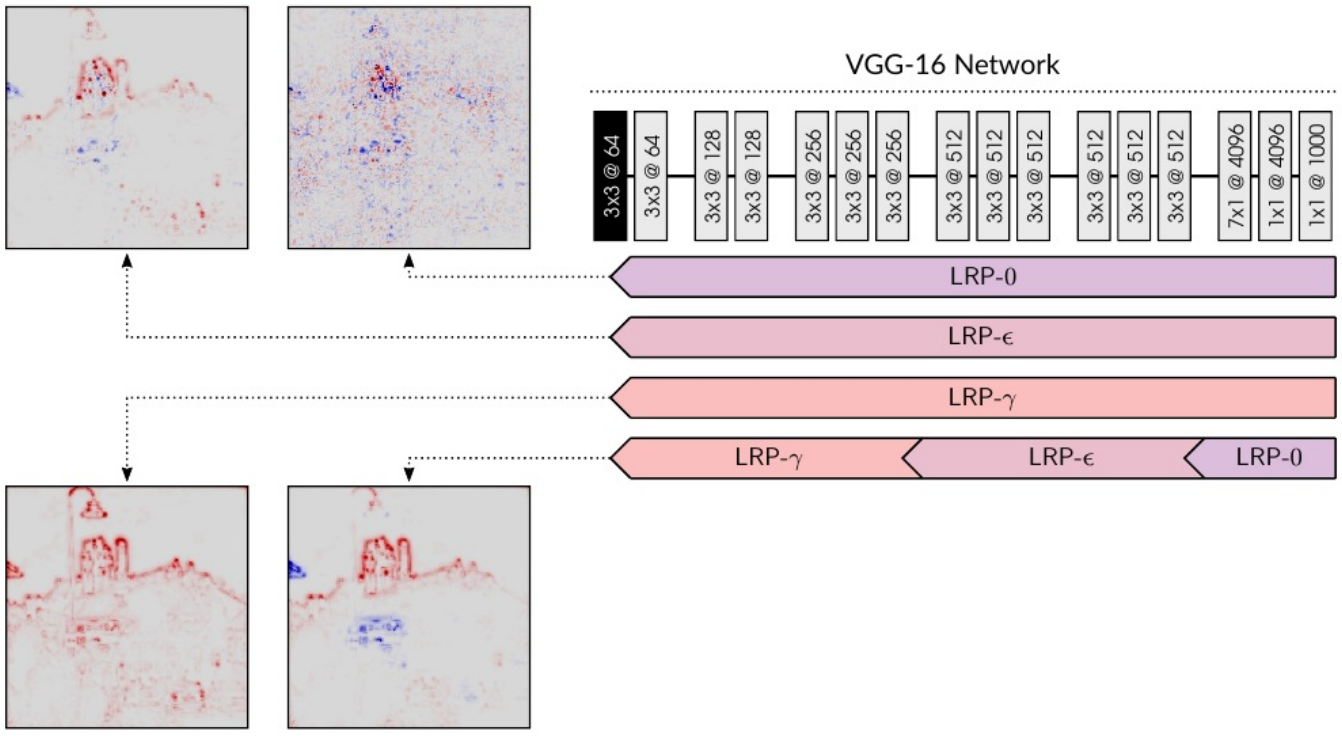
$$\forall_{j=3}^{6} : R_j \leftarrow \frac{a_j v_j}{\sum_{j=3}^{6} a_j v_j} R_{\text{out}}$$

- **Step 2:** Propagate on the first layer

$$\forall_{i=1}^{2} : R_i \leftarrow \sum_{j=3}^{6} \frac{x_i w_{ij}}{\sum_{i=1}^{2} x_i w_{ij}} R_j$$

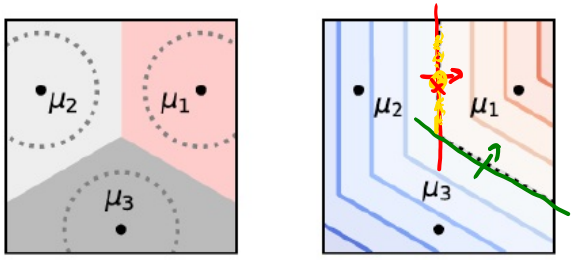
Attribution: The LRP Method

Effect of LRP rules on the explanation (e.g. class 'castle' predicted by a VGG-16 neural network.)

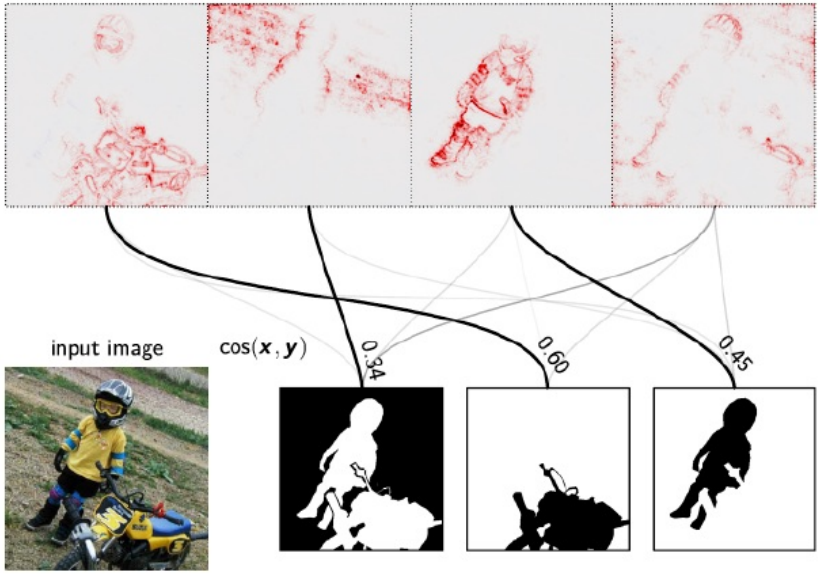


XAI for Unsupervised Learning

Example Clustering: Consider the K-means algorithm, and we would like to explain the cluster assignment of a data point $\boldsymbol{x}$.



Step 1: Transform the K-means solution into an equivalent neural network that assigns a cluster to $\boldsymbol{x}$.



Step 2: Apply LRP to the neural network to explain the cluster assignment.

XAI for Clustering

Step 1: K-means as a Neural Network:

$$\begin{aligned}
\text{decide cluster } c \quad &\Leftrightarrow \quad \min_{k \neq c} \{ \|\boldsymbol{x} - \boldsymbol{\mu}_k\|^2 \} > \|\boldsymbol{x} - \boldsymbol{\mu}_c\|^2 \\
&\Leftrightarrow \quad \min_{k \neq c} \{ \|\boldsymbol{x} - \boldsymbol{\mu}_k\|^2 - \|\boldsymbol{x} - \boldsymbol{\mu}_c\|^2 \} > 0 \\
&\Leftrightarrow \quad \min_{k \neq c} \{ \underbrace{\boldsymbol{w}_k{}^\top \boldsymbol{x} + b_k}_{h_k} \} > 0
\end{aligned}$$

Step 2: LRP applied to the neural network:

$$R_k = \frac{\exp(-\beta h_k)}{\sum_k \exp(-\beta h_k)} R_{\text{out}}$$

$$R_i = \sum_k \frac{w_{ik}(x_i - m_{ik})}{\sum_i w_{ik}(x_i - m_{ik})} R_k$$

Summary

- Explainable AI is an important addition to classical ML models (e.g. for validating a ML model or extracting knowledge from it).
- Many XAI methods have been developed, each of them, with their strengths and limitations:
  - *Activation maximization* can be used to understand what a ML model has learned, but is unsuitable for explaining an individual prediction $f(\boldsymbol{x})$.
  - *Shapley value* has strong theoretical foundations, but is computationally unfeasible for high-dimensional input data.
  - *Taylor expansions* are simple and theoretically founded for simple models, but the expansion does not extrapolate well in complex nonlinear models.
  - *LRP* leverages the structure of the ML model to handle nonlinear decision functions, but requires to carefully choose the propagation rules.