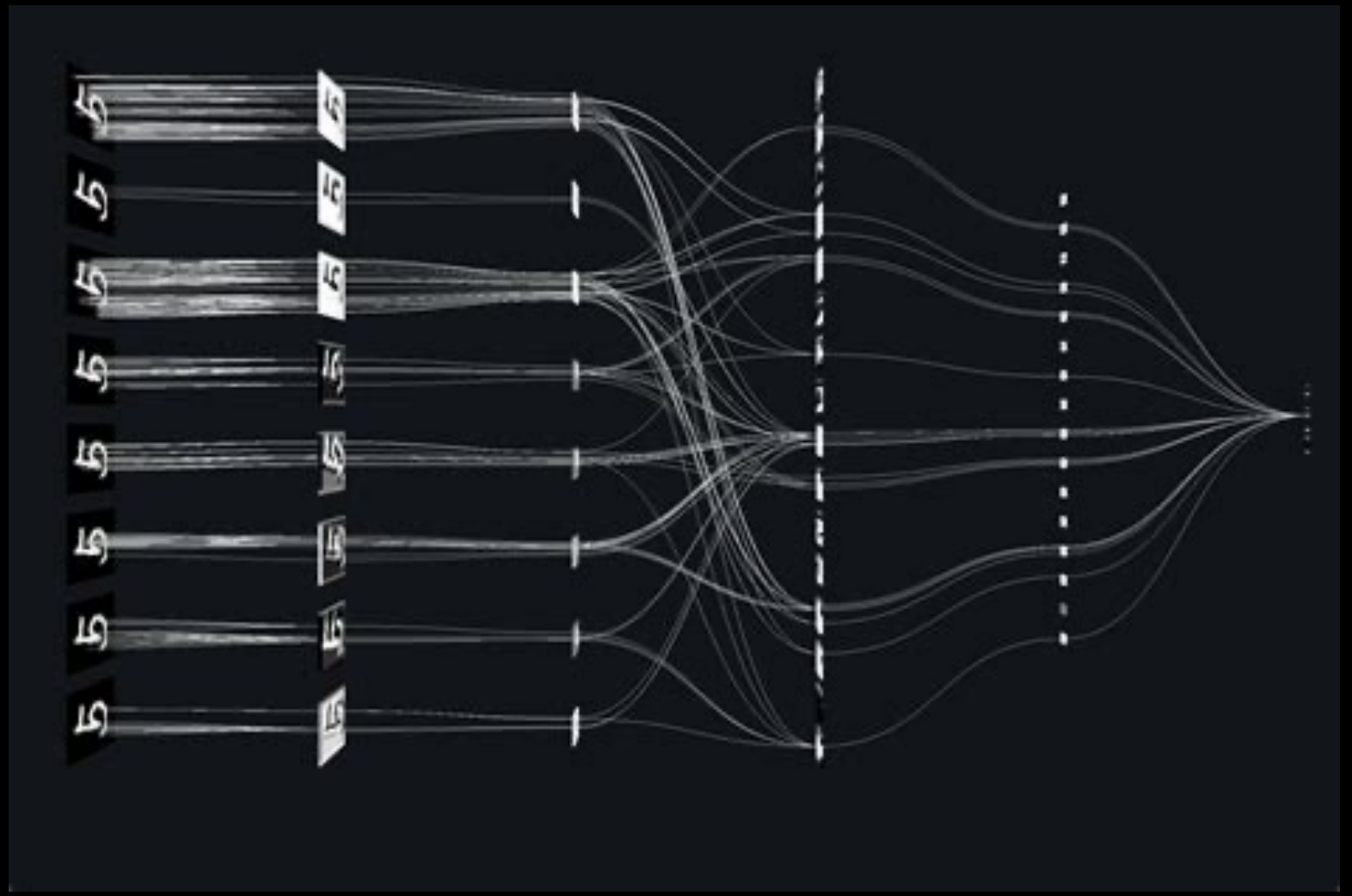
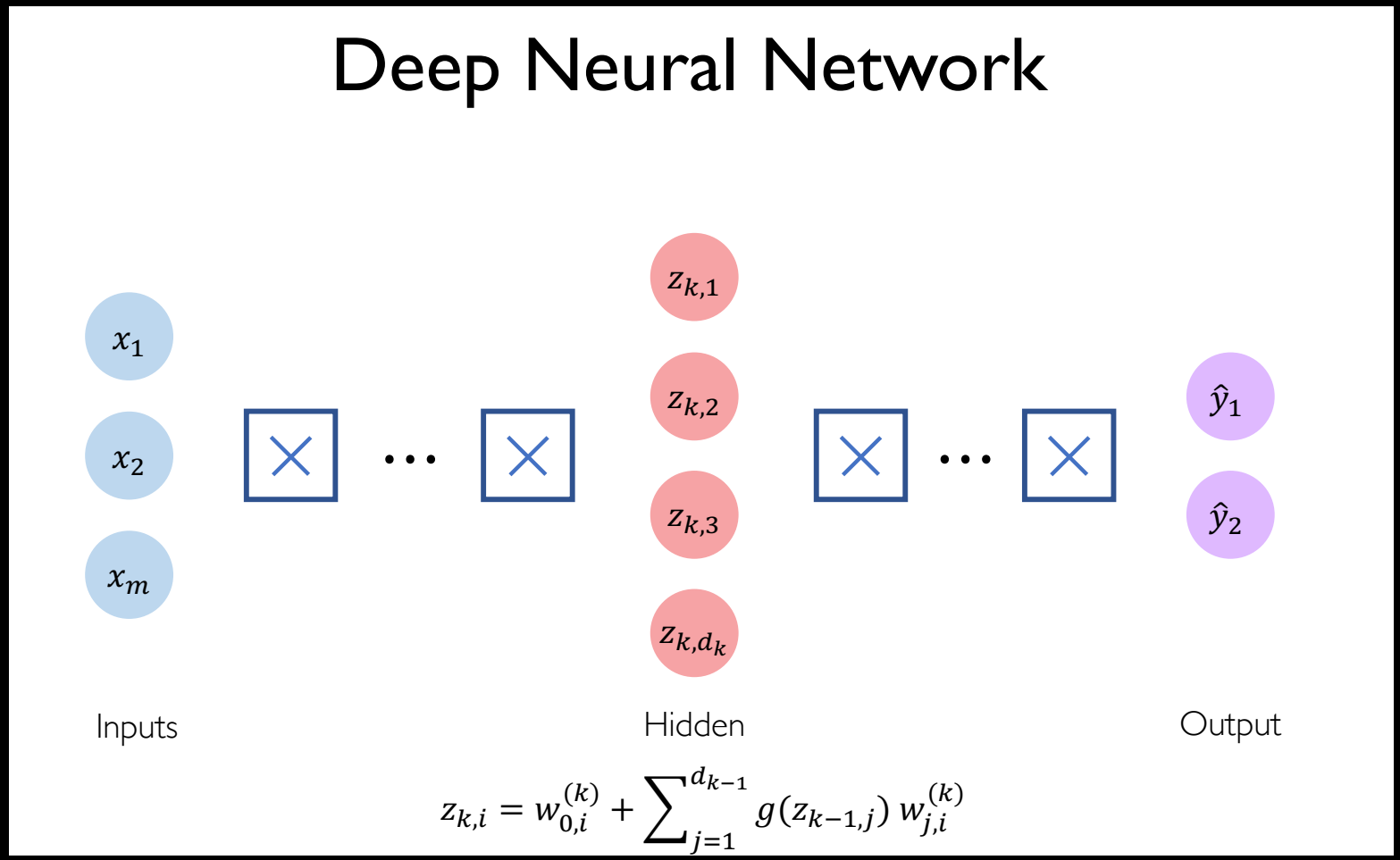
Explainable ML - XML

Pierre Gentine – Columbia University



The problem

Major advances with the deep learning



Many (many) weights → black box

How to open up the black box?

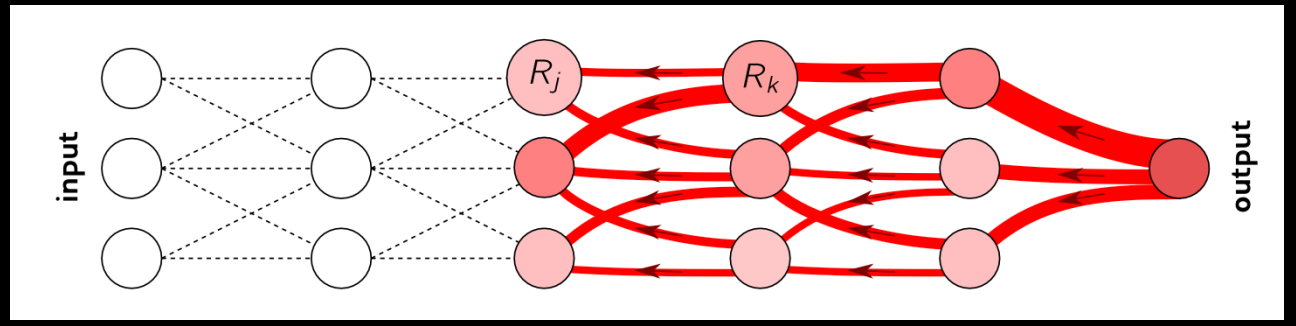
Interpreting what is in the gut of deep NN/CNN

Explainable ML
Also helps build trustworthiness in models
→ Right answer for the right reason

A few examples:
1. Layerwise Relevance Propagation (LRP)
2. Adjoint/gradient – Saliency maps
3. SHAPley values

1. Layerwise Relevance Propagation (LRP)

Trying to assess where the information is coming from.
Backward – from output to input that most explain the output

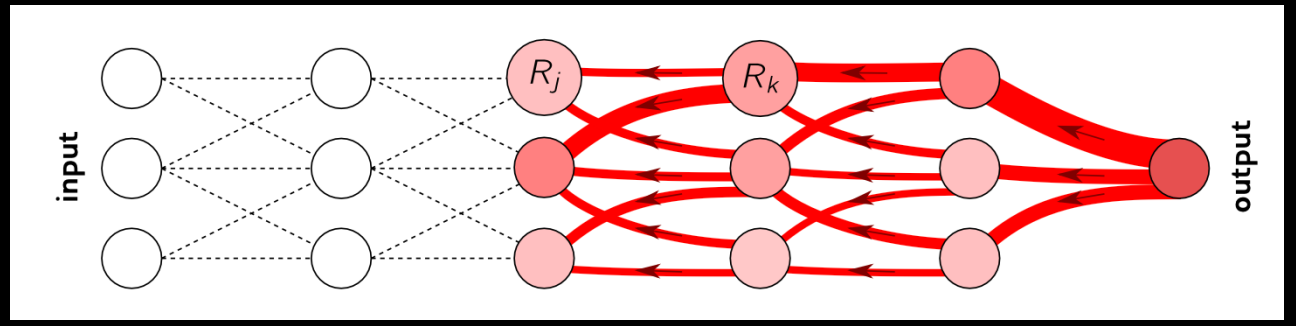


Pass information/relevance backward – a la Kirchoff law (conservation of current):

Toms et al. 2020 JAMES

1. Layerwise Relevance Propagation (LRP)

Trying to assess where the information is coming from
Backward – from output to input that most explain the output



Pass information/relevance backward – a la Kirchoff law (conservation of current):

$$R_j = \sum_k \frac{a_j w_{jk}}{\sum_{0,j} a_j w_{jk}} R_k$$

Layer j before layer k
a_j – output from layer j (after activation function)
W_{jk} - weights

Toms et al. 2020 JAMES

1. Layerwise Relevance Propagation (LRP)

Different LRP rules

1. LRP-0

$$R_j = \sum_k \frac{a_j w_{jk}}{\sum_{0,j} a_j w_{jk}} R_k$$

Equivalent to gradient x input

2. LRP-epsilon

Limit noisy information – filter by small noise epsilon

$$R_j = \sum_k \frac{a_j w_{jk}}{\epsilon + \sum_{0,j} a_j w_{jk}} R_k$$

3. LRP-gamma

Favor positive contributions compared to negative contributions

$$R_j = \sum_k \frac{a_j \cdot (w_{jk} + \gamma w_{jk}^+)}{\sum_{0,j} a_j \cdot (w_{jk} + \gamma w_{jk}^+)} R_k$$

Montavon et al.

1. Layerwise Relevance Propagation (LRP)

Different LRP rules → generic rule

$$R_j = \sum_k \frac{a_j \cdot \rho(w_{jk})}{\epsilon + \sum_{0,j} a_j \cdot \rho(w_{jk})} R_k$$

Can split computation into 4 substeps

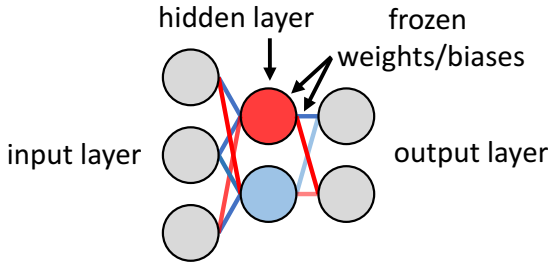
$$\begin{aligned}
\forall_k : \; & z_k = \epsilon + \textstyle\sum_{0,j} a_j \cdot \rho(w_{jk}) && \text{(forward pass)} \\
\forall_k : \; & s_k = R_k / z_k && \text{(element-wise division)} \\
\forall_j : \; & c_j = \textstyle\sum_k \rho(w_{jk}) \cdot s_k && \text{(backward pass)} \\
\forall_j : \; & R_j = a_j \, c_j && \text{(element-wise product)}
\end{aligned}$$

What we will do in the notebook – stay tuned

Montavon et al.

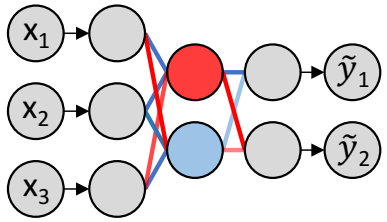
1. Layerwise Relevance Propagation (LRP)

1) Train network & freeze weights/biases



Information learned during training:
positive weights/biases
negative weights/biases

2) Input sample into frozen network and retain output

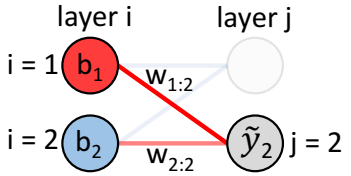


full output $\tilde{y}_1$ $\tilde{y}_2$

selected output node to propagate relevance

$\tilde{y}_2$ ← starting relevance (R_j)

3) Propagate relevance from output node to previous layer



*ignored nodes/weights are transparent

propagation rule

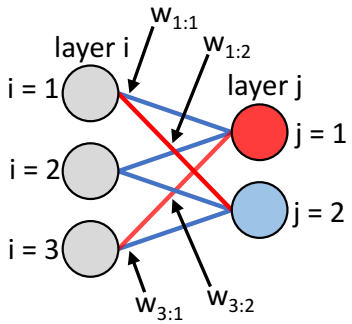
$$R_i = \sum_j \frac{a_i w_{ij}^+ + \max(0, b_j)}{\sum_i a_i w_{ij}^+ + max(0, b_j)} R_j$$

example relevance calculations

$$R_{i=1} = \left(\frac{a_1 w_{1:2}}{a_1 w_{1:2} + a_2 w_{2:2}}\right) \tilde{y}_2$$

$$R_{i=2} = \left(\frac{a_2 w_{2:1}}{a_1 w_{1:2} + a_2 w_{2:2}}\right) \tilde{y}_2$$

4) Propagate relevance from hidden layer to input layer



propagation rule

$$R_i = \sum_j \left(\frac{w_{ij}^2}{\sum_i w_{ij}^2}\right) R_j$$

*all biases are ignored for this rule

relevance at input layer
R_1 R_2 R_3

example relevance calculation

$$R_{i=1} = \left(\frac{{w_{1:1}}^2}{{w_{1:1}}^2 + {w_{2:1}}^2 + {w_{3:1}}^2}\right) R_{j=1} + \left(\frac{{w_{1:2}}^2}{{w_{1:2}}^2 + {w_{2:2}}^2 + {w_{3:2}}^2}\right) R_{j=2}$$

*relevance calculations are similar for other input nodes

5) Repeat for each sample of interest…

Toms et al. 2020 JAMES

1. Layerwise Relevance Propagation (LRP)

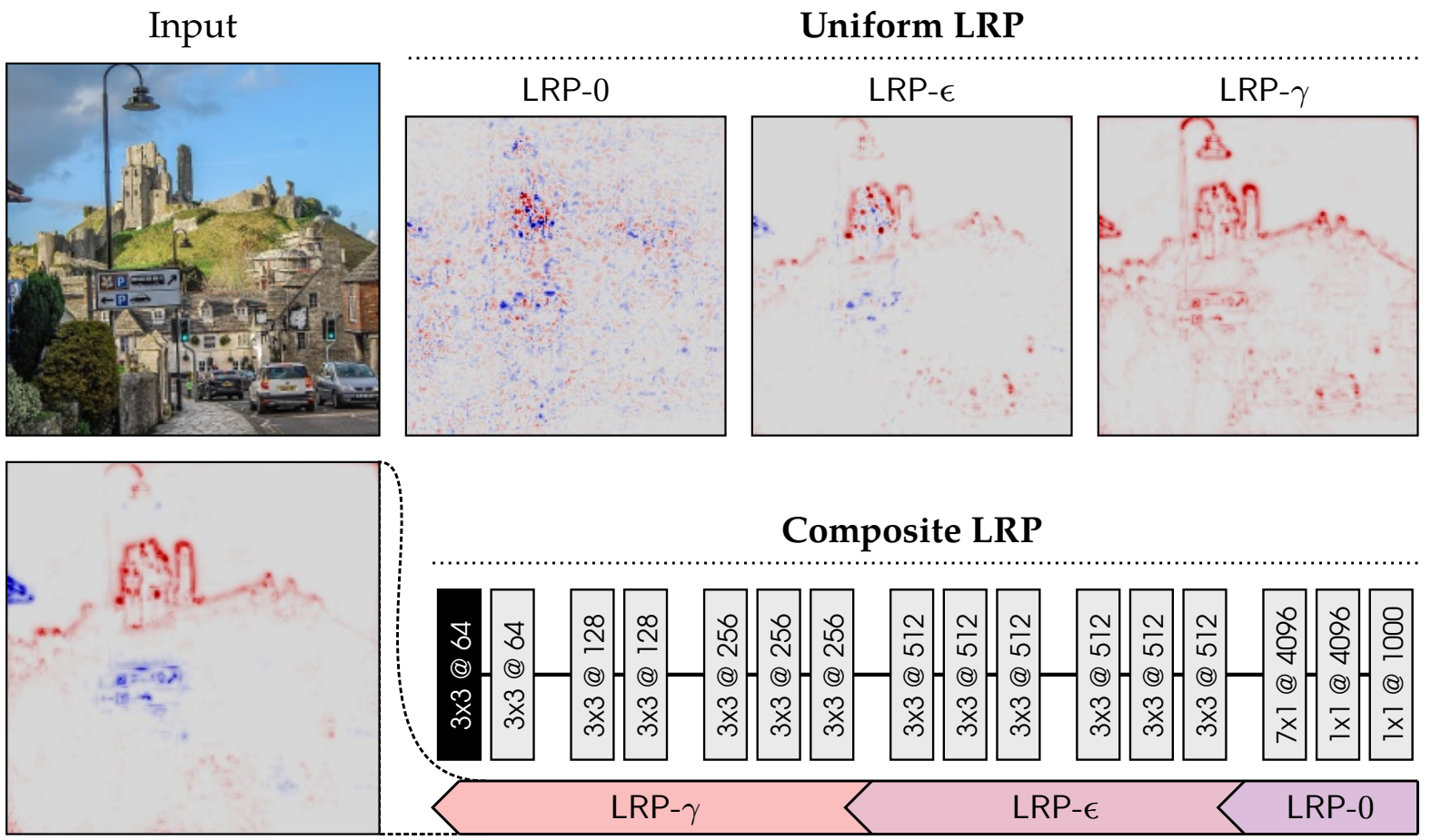


Fig. 10.4. Input image and pixel-wise explanations of the output neuron 'castle' obtained with various LRP procedures. Parameters are $\epsilon = 0.25\,\text{std}$ and $\gamma = 0.25$.

2. Gradient/adjoint

Gradient/adjoint of output **y** to inputs **x**

$$\nabla_{\mathbf{x}}\mathbf{y}$$

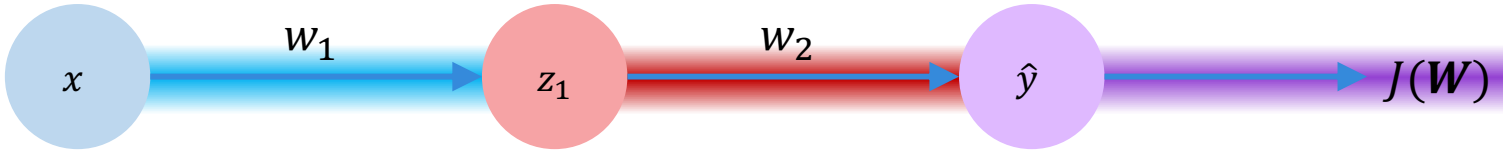
$$J = \begin{pmatrix} \frac{\partial \mathbf{y}}{\partial x_1} & \cdots & \frac{\partial \mathbf{y}}{\partial x_n} \end{pmatrix} = \begin{pmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_n} \end{pmatrix}$$

2. Gradient/adjoint

Gradient/adjoint of output **y** to inputs **x**

$$J = \begin{pmatrix} \frac{\partial \mathbf{y}}{\partial x_1} & \cdots & \frac{\partial \mathbf{y}}{\partial x_n} \end{pmatrix} = \begin{pmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_n} \end{pmatrix}$$

Once upon a time: **backpropagation**
Chain's rule



$$\frac{\partial J(\boldsymbol{W})}{\partial w_1} = \frac{\partial J(\boldsymbol{W})}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z_1} * \frac{\partial z_1}{\partial w_1}$$

Same thing!

2. Gradient/adjoint

Personal experience:

Non-linear so average over different samples

Use non-dimensional/normalized inputs (and ideally outputs)

Tends to work well for first cut – but can be noisy

2. Gradient/adjoint

Personal experience:

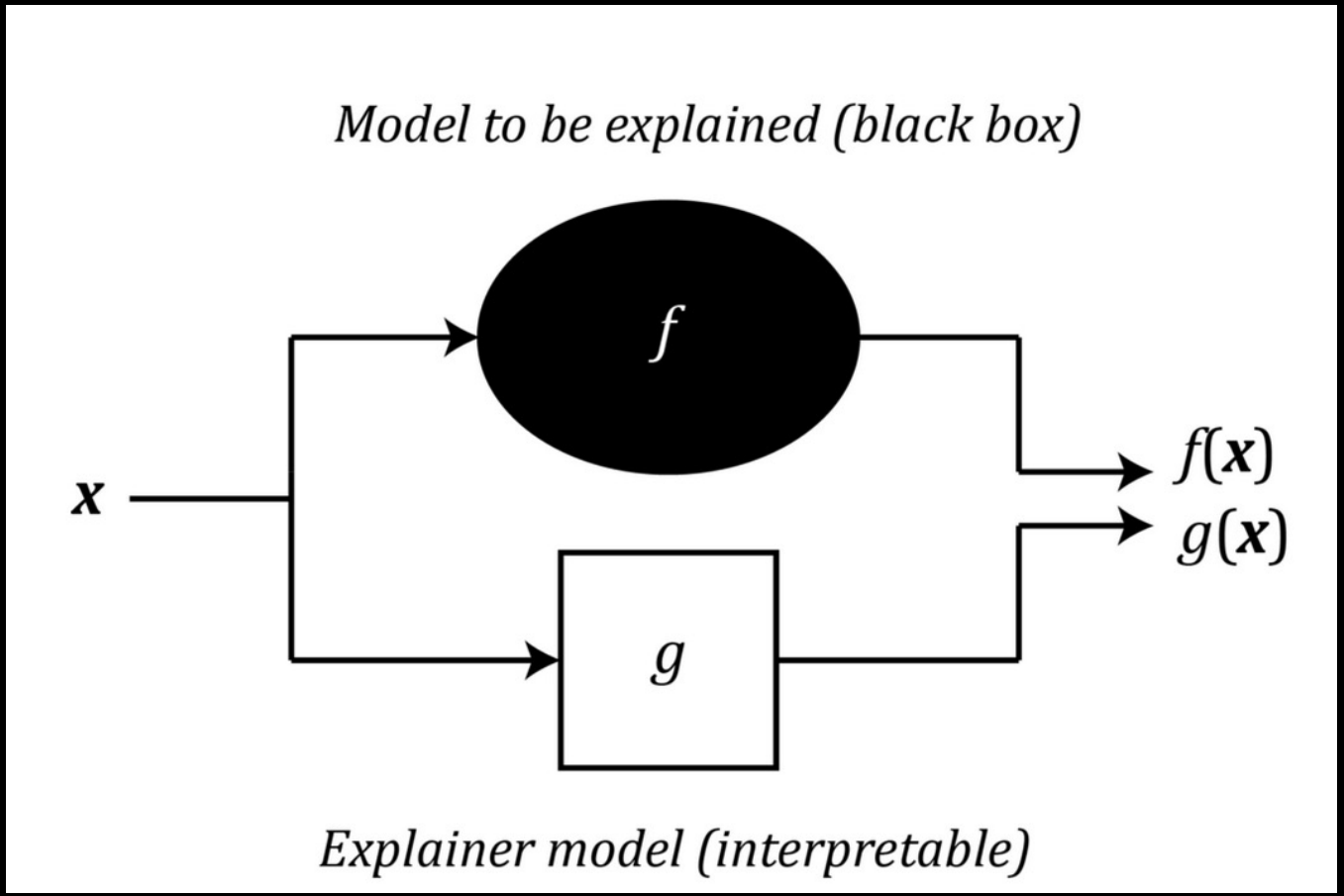
Non-linear so average over different samples

Use non-dimensional/normalized inputs (and ideally outputs)

Tends to work well for first cut – but can be noisy

3. Shapley Additive Explanations (SHAP) values

Inspired by Game theory.



g mimics *f* for a single prediction

3. Shapley Additive Explanations (SHAP) values

Inspired by **Game theory**
Lloyd Shapley (Nobel in Economics)

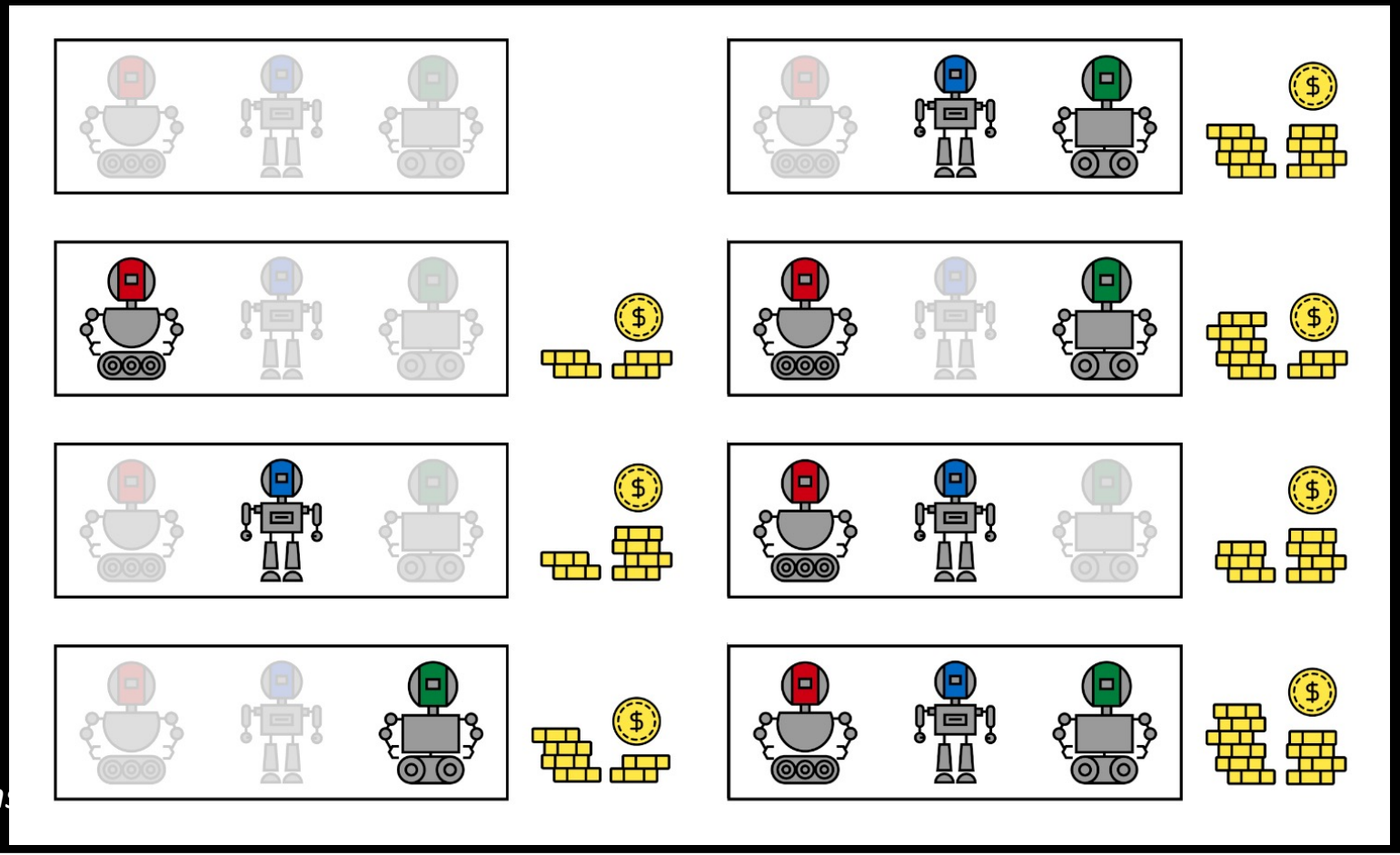
If we have a coalition C that collaborates to produce a value V

How much did each individual member contribute to the final value?

Cooperative Game Theory

Cooperative game { A player set (coalition: subset of players)

A characteristic function that defines the value of coalitions



Rozemberczki et al., 2022

3. SHAP values

Cooperative Game Theory

Coalition: $\mathcal{S} \subseteq \mathcal{M} = \{1, \ldots, M\}$

Shapley value:

$$\phi_j(v) = \phi_j = \sum_{\mathcal{S} \subseteq \mathcal{M} \setminus \{j\}} \frac{|\mathcal{S}|!(M - |\mathcal{S}| - 1)!}{M!} (v(\mathcal{S} \cup \{j\}) - v(\mathcal{S})), \quad j = 1, \ldots, M,$$

$\phi_i(N, v)$ is the Shapley value for player i.

N is the set of all players.

v is the characteristic function that maps coalitions to the value they generate.

S is a subset of players (a coalition).

$|S|$ represents the number of players in coalition S.

Aas et al., 2017

3. SHAP values

Shapley value: is the mean marginal contribution of each feature value across all possible values in the feature space.

Shapley value  **Machine learning**

Scott Lundberg

Su-In Lee

SHAP:
Shapley **A**ddictive Ex**p**lanations

A Unified Approach of Interpreting Model Predictions. Lundberg & Lee, 2017

3. SHAP values

Shapley value – machine learning(model-agnostic $f(x)$)

Local explanation :

$$f(\boldsymbol{x}^*) = \phi_0 + \sum_{j=1}^{M} \phi_j^*$$

$$\boldsymbol{x} = \boldsymbol{x}^*$$

$$\phi_0 = \mathrm{E}[f(\boldsymbol{x})]$$

$$\phi_j(v) = \phi_j = \sum_{\mathcal{S} \subseteq \mathcal{M} \setminus \{j\}} \frac{|\mathcal{S}|!(M - |\mathcal{S}| - 1)!}{M!} (v(\mathcal{S} \cup \{j\}) - v(\mathcal{S})), \quad j = 1, \ldots, M,$$

$$v(\mathcal{S}) = \mathrm{E}[f(\boldsymbol{x}) | \boldsymbol{x}_{\mathcal{S}} = \boldsymbol{x}_{\mathcal{S}}^*]$$

(Lundberg and Lee, 2017)

3. SHAP values

$$v(\mathcal{S}) = \mathrm{E}[f(\boldsymbol{x})|\boldsymbol{x}_{\mathcal{S}} = \boldsymbol{x}^*_{\mathcal{S}}]$$

$\bar{S}$ is the complement of S

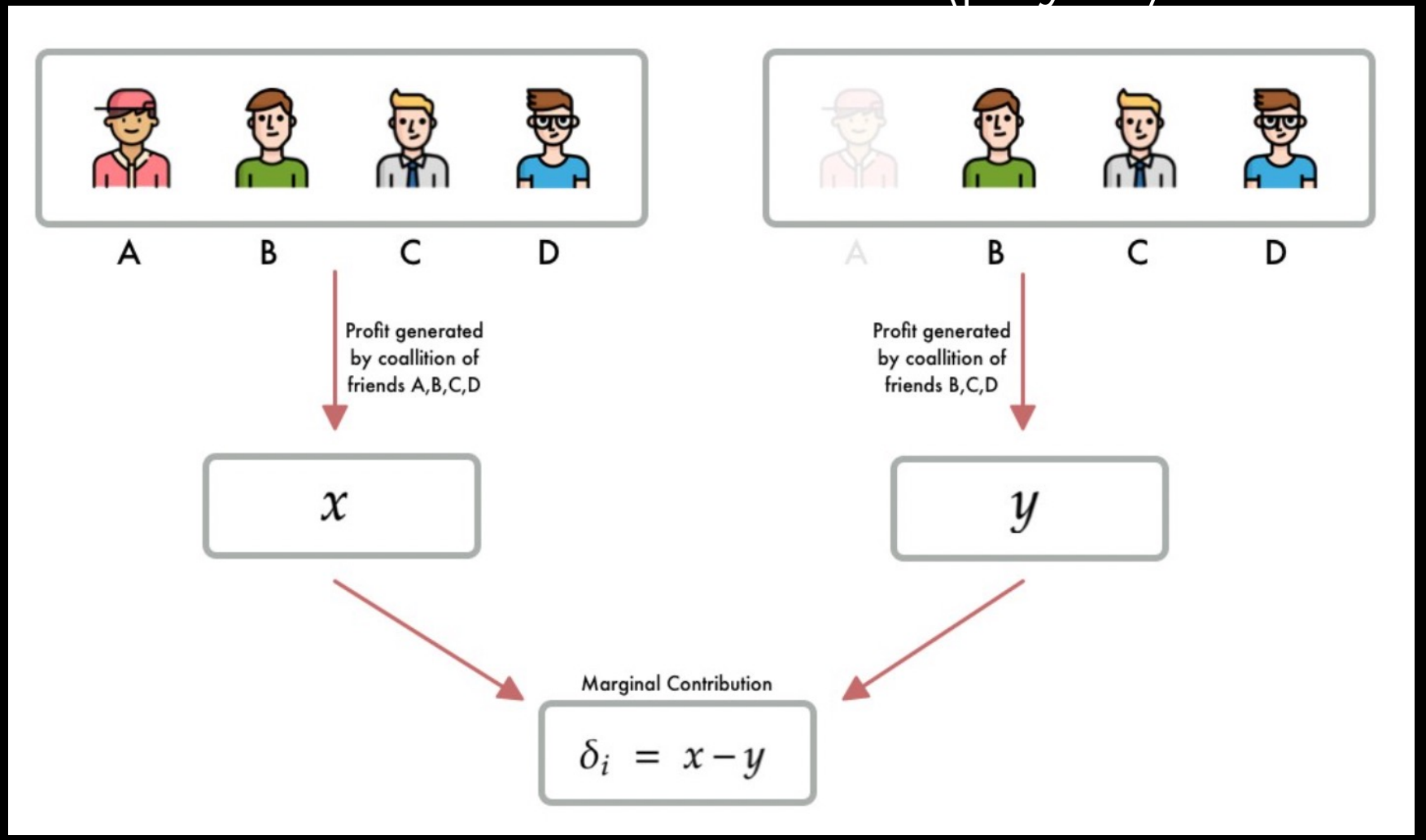
$$E[f(\boldsymbol{x})|\boldsymbol{x}_{\mathcal{S}} = \boldsymbol{x}^*_{\mathcal{S}}] = E[f(\boldsymbol{x}_{\bar{\mathcal{S}}}, \boldsymbol{x}_{\mathcal{S}})|\boldsymbol{x}_{\mathcal{S}} = \boldsymbol{x}^*_{\mathcal{S}}]$$
$$= \int f(\boldsymbol{x}_{\bar{\mathcal{S}}}, \boldsymbol{x}^*_{\mathcal{S}})\, p(\boldsymbol{x}_{\bar{\mathcal{S}}}|\boldsymbol{x}_{\mathcal{S}} = \boldsymbol{x}^*_{\mathcal{S}}) d\boldsymbol{x}_{\bar{\mathcal{S}}},$$

$$p(\boldsymbol{x}_{\bar{\mathcal{S}}}|\boldsymbol{x}_{\mathcal{S}} = \boldsymbol{x}^*_{\mathcal{S}}) = p(\boldsymbol{x}_{\bar{\mathcal{S}}})$$ (SHAP, Lundberg & Lee, 2017)

$$P(\mathbf{X}_{\bar{S}}|do(\mathbf{X}_S = \mathbf{x}_S)) = P(\mathbf{X}_{\bar{S}})$$ (Causal Shapley values, Heskes et al., 2020)

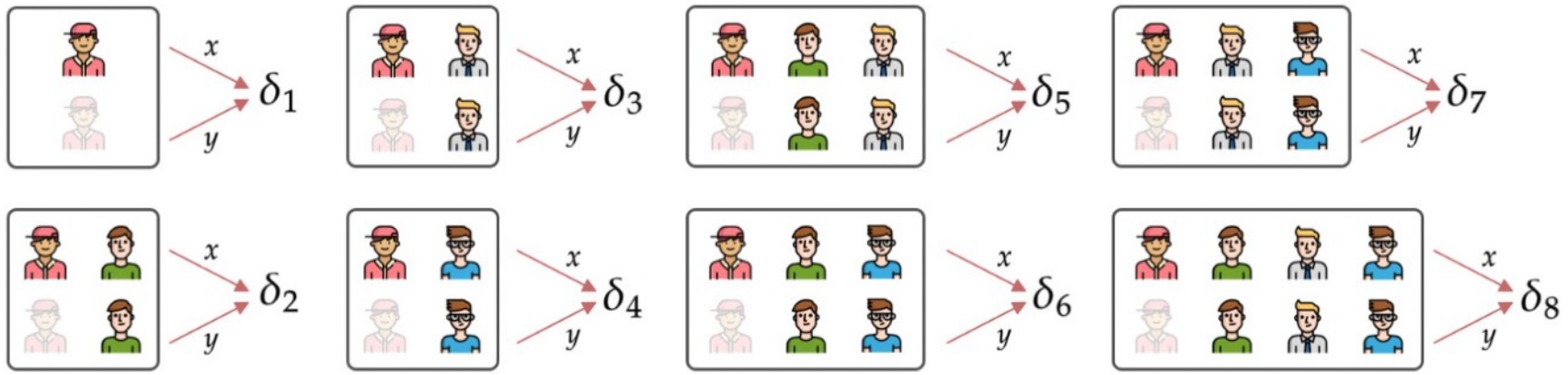
3. SHAP values

Marginal contribution of member A
Coalition: subset of features (players)



3. SHAP values

Marginal contribution of member A



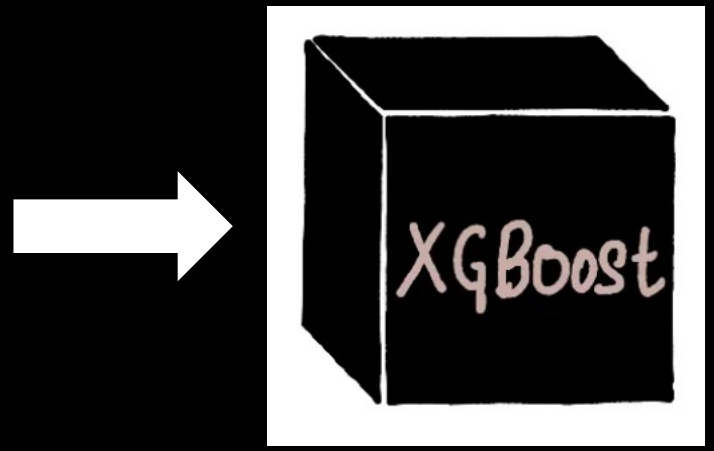
The Shapley value for member 

is given by:

$$\phi_i = \frac{\delta_1 + \delta_3 + \delta_4 + \delta_5 + \delta_6 + \delta_7 + \delta_8}{8}$$

Machine learning (model-agnostic): $f(\boldsymbol{x})$

RAD
TA
VPD
SWC
CO2



NEP

x^*

RAD = 408 TA = 19 VPD = 9 SWC = 46 CO2 = 372

NEP*= 16

Decomposing prediction:

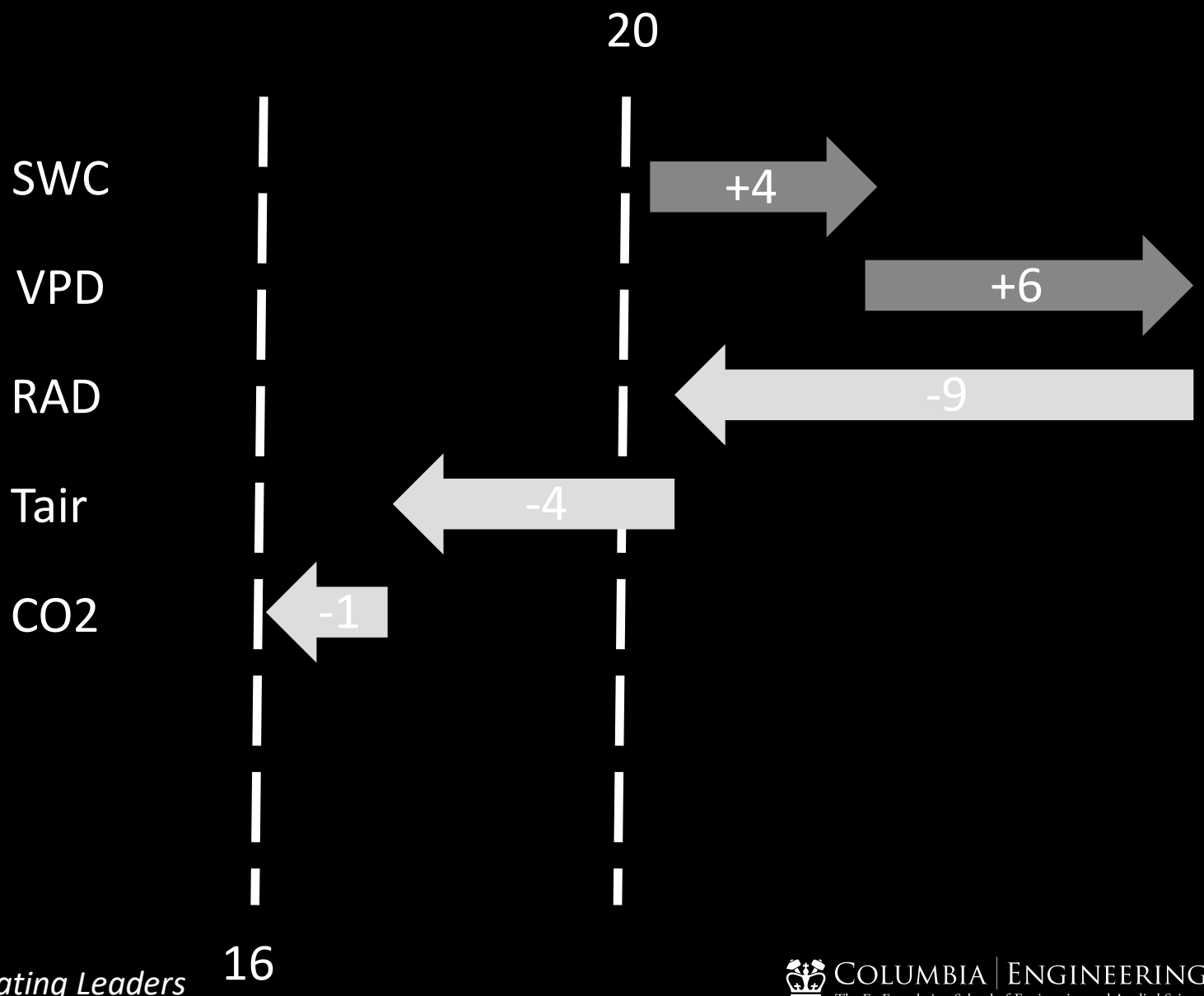
$$f(\boldsymbol{x}^*) = \phi_0 + \sum_{j=1}^{M} \phi_j^*$$

$$\phi_0 = \mathrm{E}[f(\boldsymbol{x})]$$

20

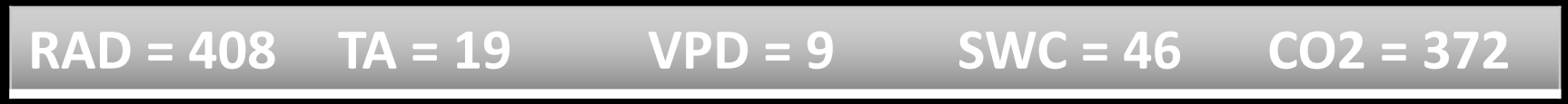
RAD = 408 TA = 19 VPD = 9 SWC = 46 CO2 = 372

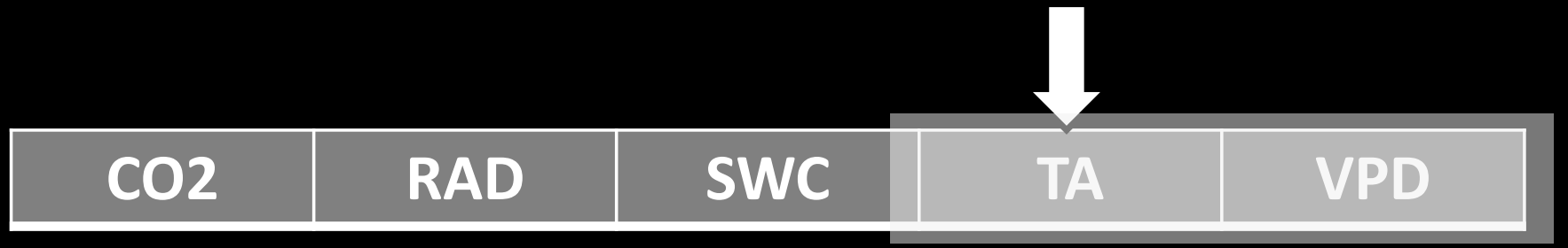
E(GPP) = 20, GPP* = 16



Step 1: Get a random permutation of the features

Original





Step 2: Pick a random sample from the dataset

CO2	RAD	SWC	TA	VPD
380	112	20	17	30

Step 3: Form two new samples

Random

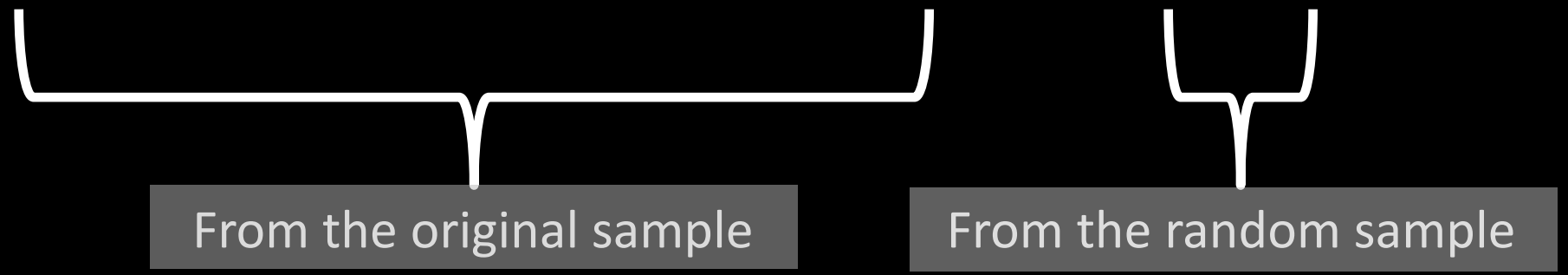
CO2	RAD	SWC	TA	VPD
380	112	20	17	30

Original

CO2	RAD	SWC	TA	VPD
372	408	46	19	9

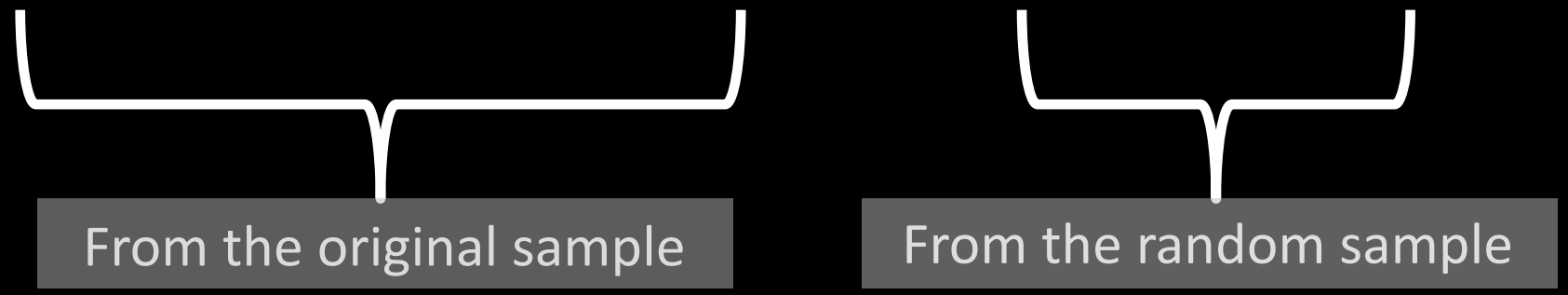
Step 3: Form two new samples (X1)

CO2	RAD	SWC	TA	VPD
372	408	46	19	30



Step 3: Form two new samples (X2)

CO2	RAD	SWC	TA	VPD
372	408	46	17	30



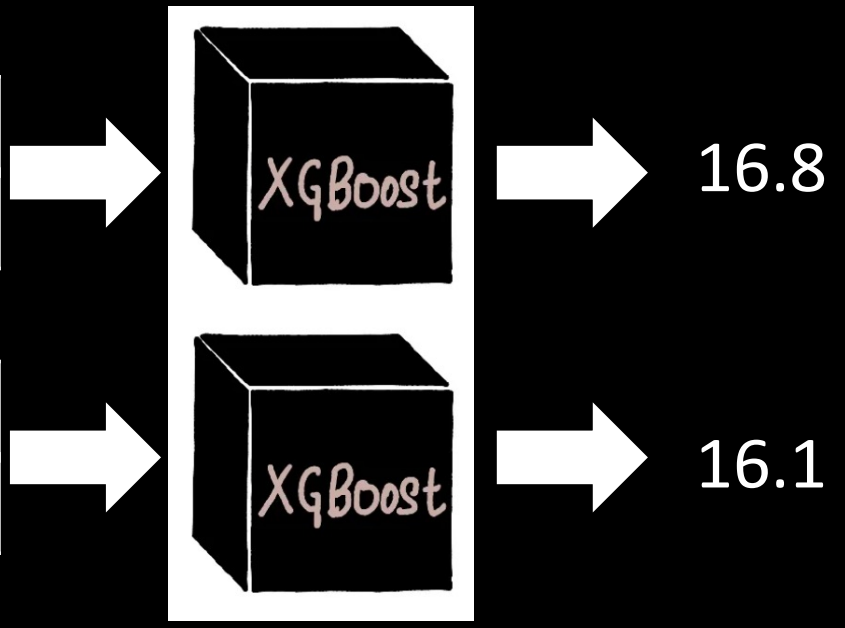
Step 4: Use the new samples to make predictions and find the difference in predictions

X1

CO2	RAD	SWC	TA	VPD
372	408	46	19	30

X2

CO2	RAD	SWC	TA	VPD
372	408	46	17	30



16.8

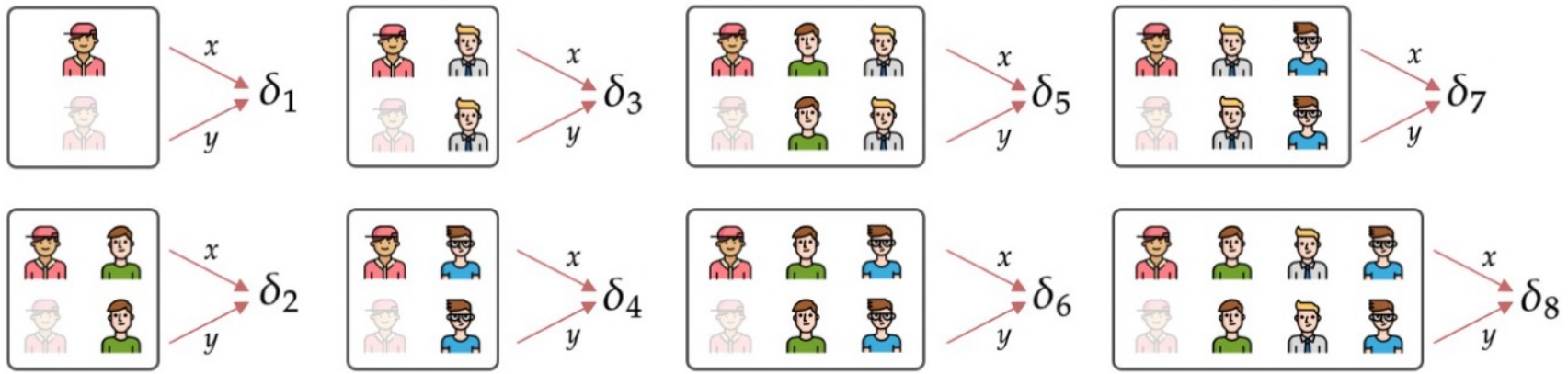
16.1

Contribution of TA: 16.8 – 16.1 = 0.7

Step 5: Repeat Steps 1–4 a couple of times and then calculate the average of the differences

3. SHAP values

Marginal contribution of member A



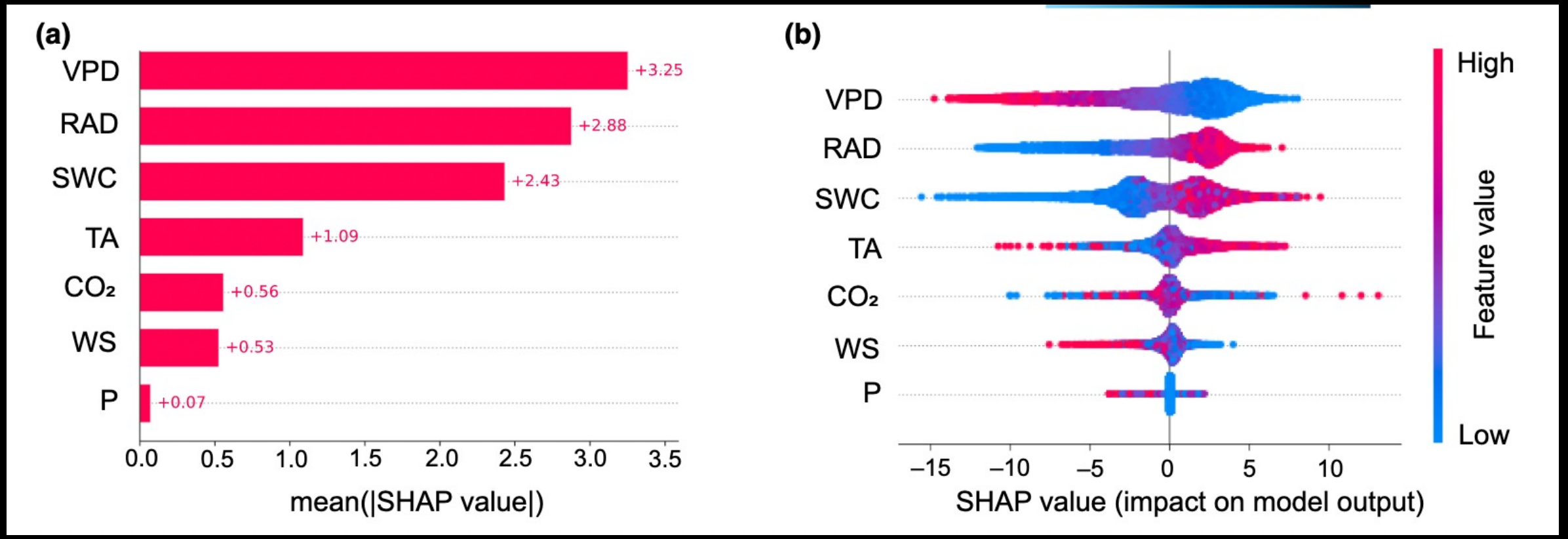
The Shapley value for member 

is given by:

$$\phi_i = \frac{\delta_1 + \delta_3 + \delta_4 + \delta_5 + \delta_6 + \delta_7 + \delta_8}{8}$$

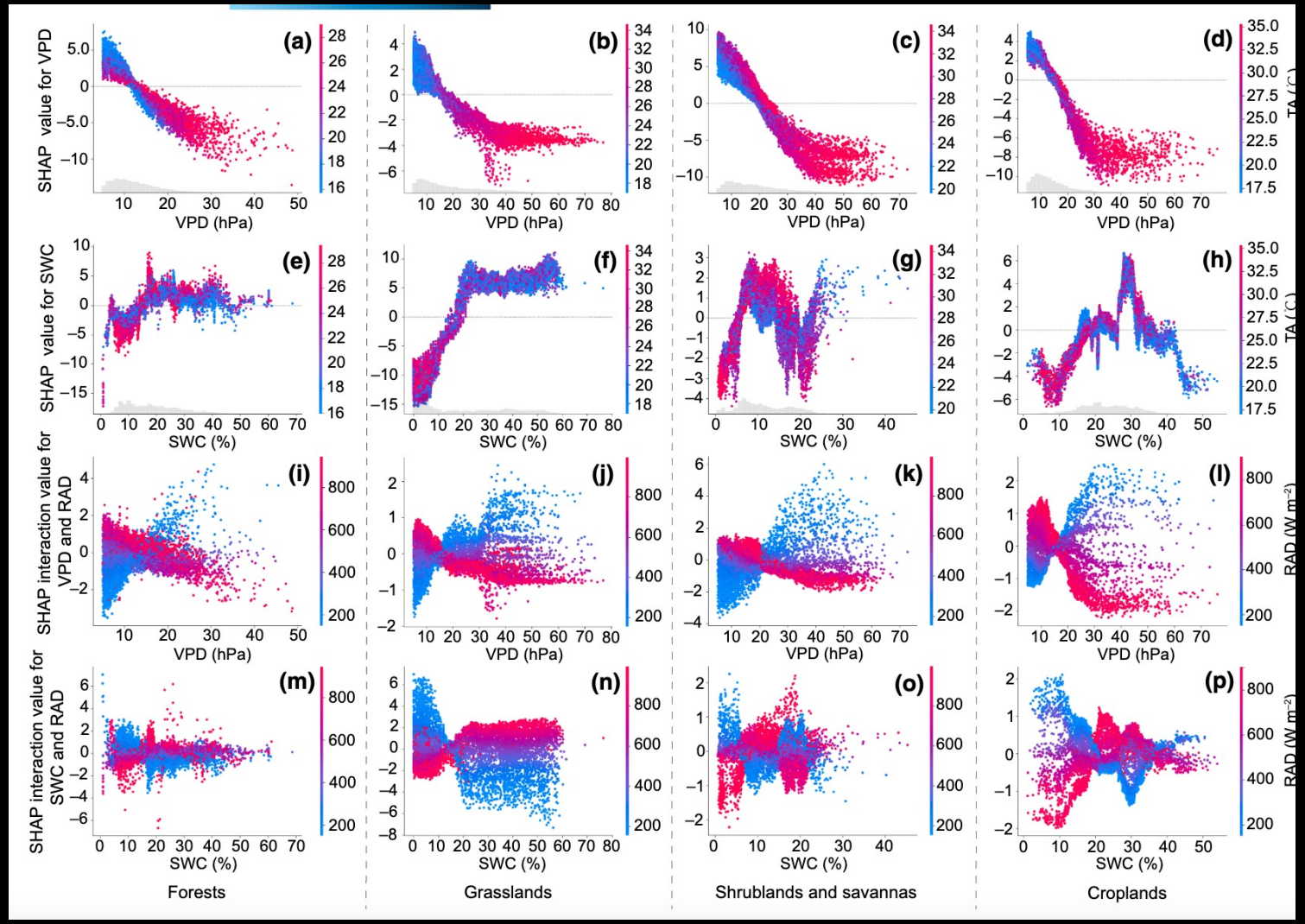
3. SHAP values

Example for continuous variables



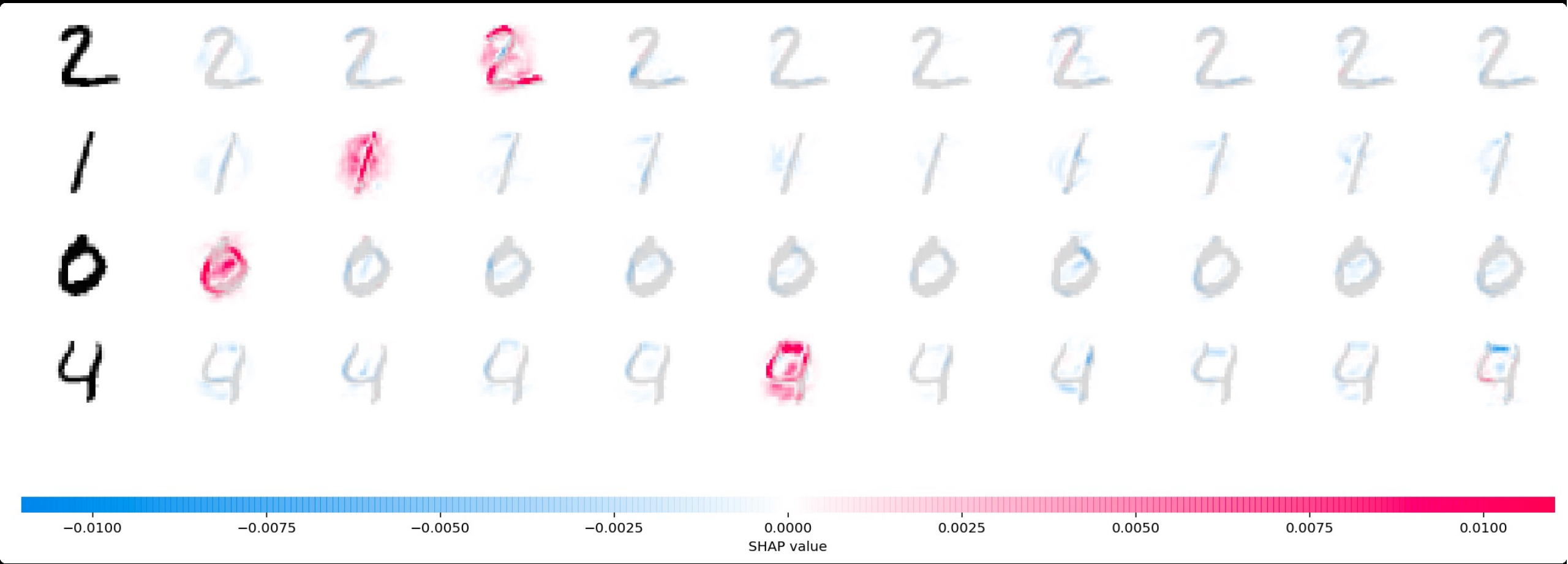
3. SHAP values

Example for continuous variables



3. SHAP values

Example for images



3. SHAP values

Example for images