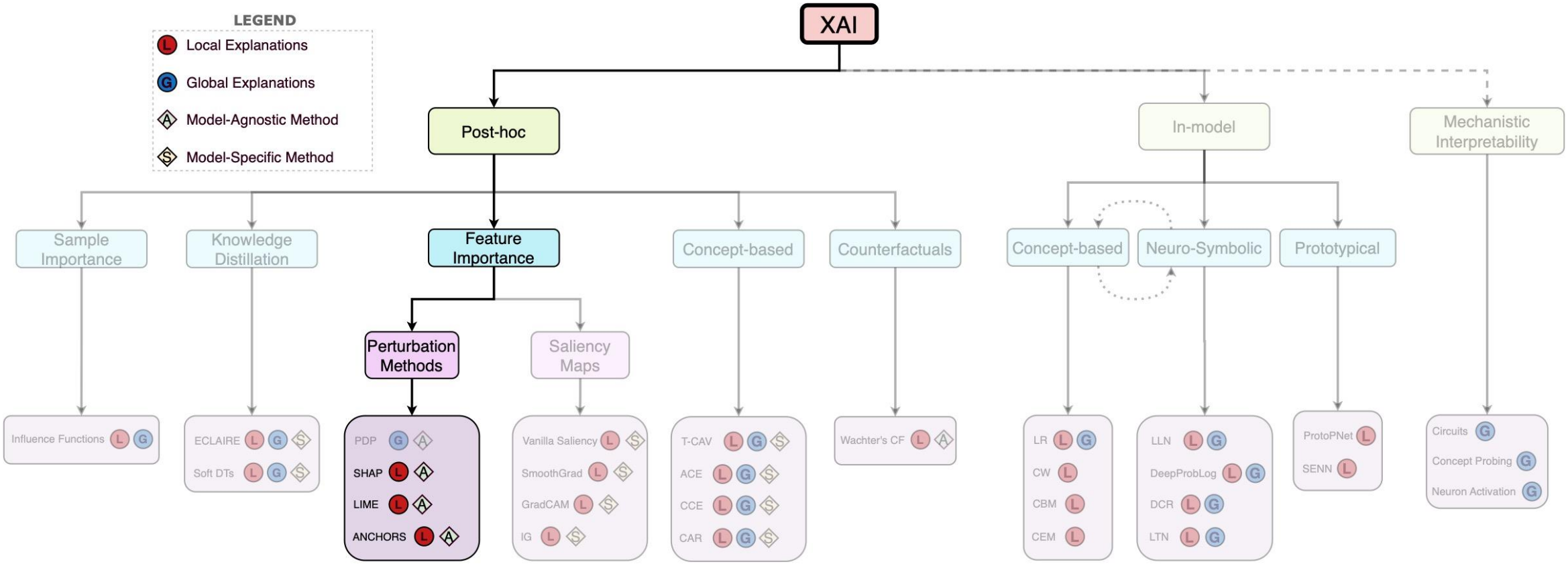
# EXPLAINABLE ARTIFICIAL INTELLIGENCE

L193 – Lecture 2 – Lent 2025

UNIVERSITY OF CAMBRIDGE

# LOCAL PERTURBATION METHODS

# LOCAL PERTURBATION METHODS



# WHAT DOES IT MEAN TO BE LOCAL?

- **Central assumption in local explainability**: While the model may be complex globally, it is probably way less complex locally
- Locality refers to the **vicinity of a particular sample** for which we seek an explanation
- Local explanations vary for each sample despite being based on the same complex model

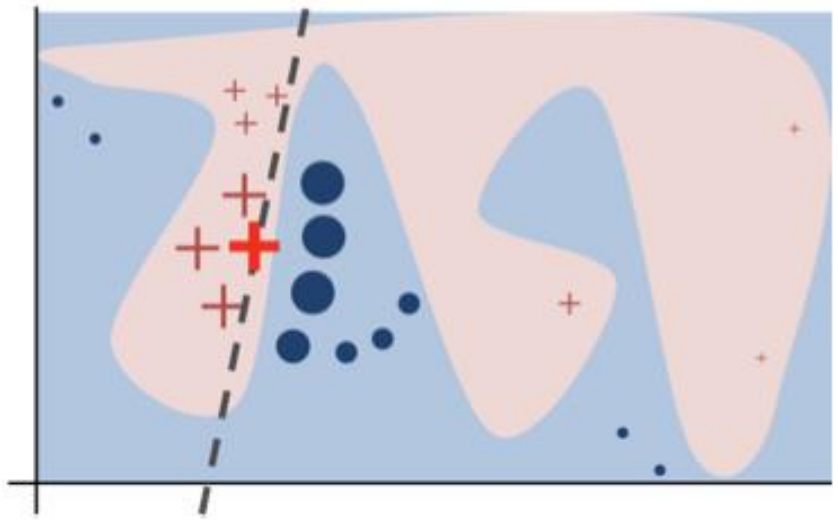


Image taken from: Ribeiro, Marco Tulio, Sameer Singh, and Carlos Guestrin. " Why should I trust you?" Explaining the predictions of any classifier." Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. 2016.

# EXPLAINING A SAMPLE ONE FEATURE AT A TIME

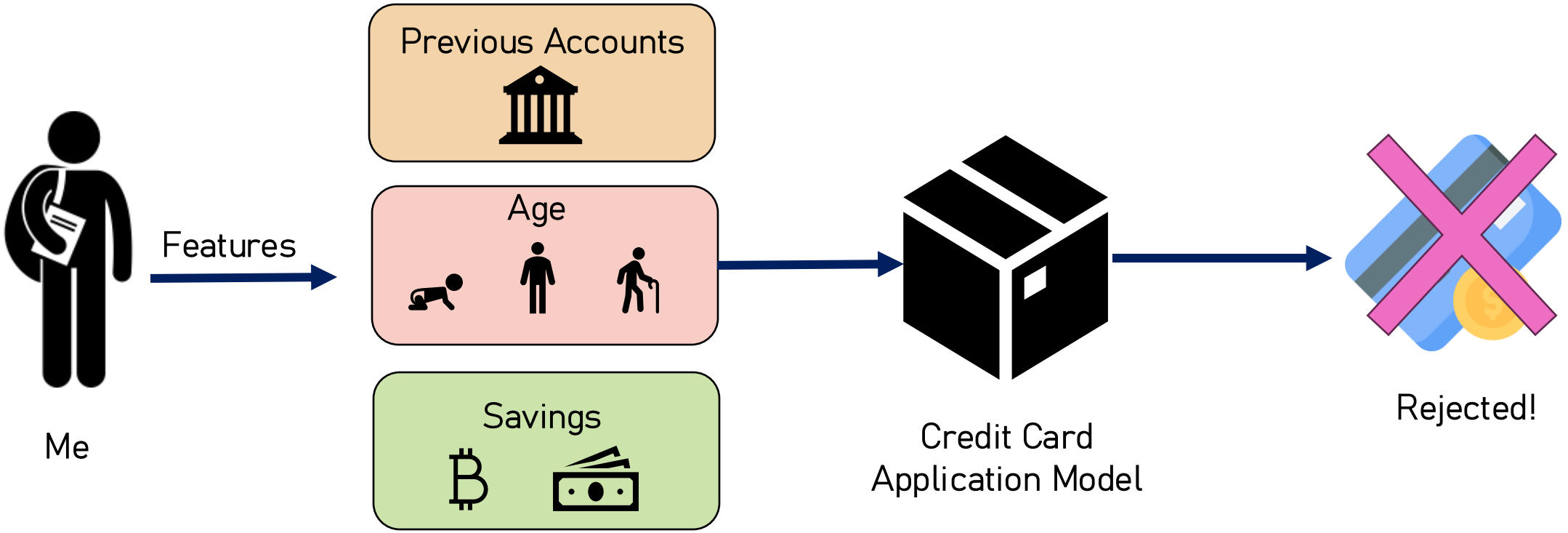
- Ask a child why  is an apple?
  - It's round
  - It's red
  - ...
- Common for humans to explain things based on features

| shape | colour | size | fruit type |
|---|---|---|---|
| round | purple | small | blueberry |
| round | red | medium | apple |
| oval | yellow | medium | banana |
| heart | red | small | strawberry |

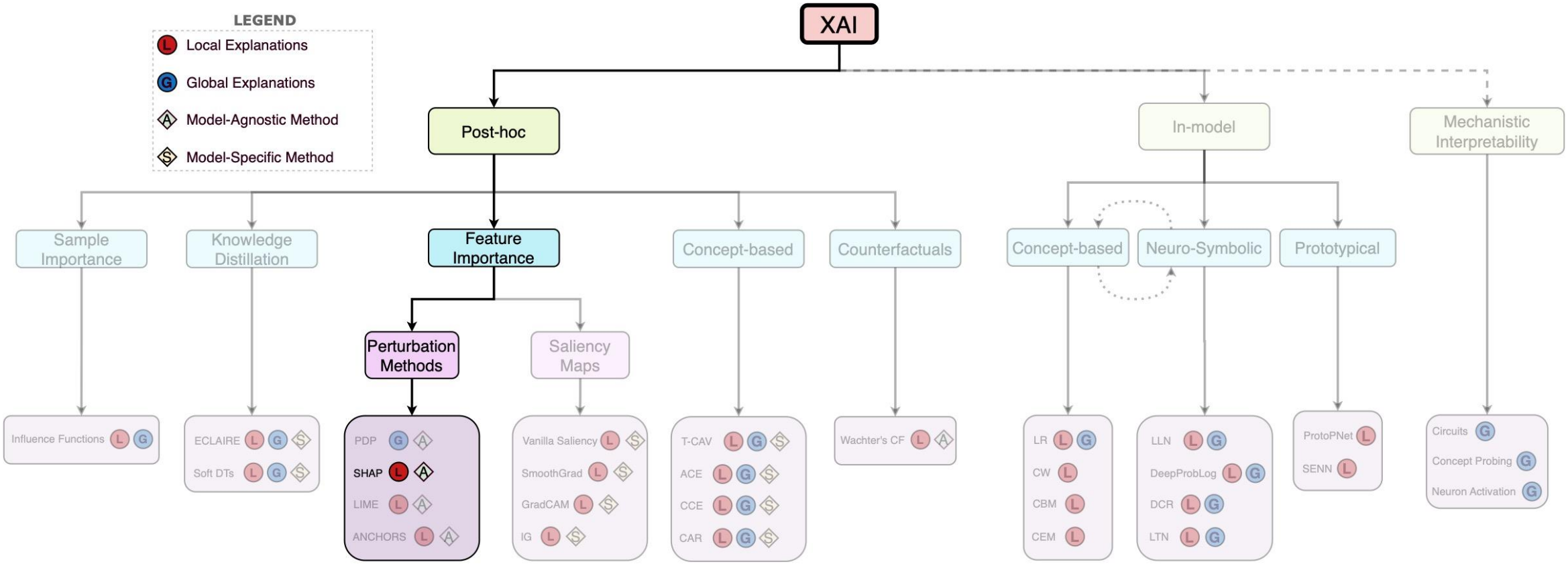
# FEATURE ATTRIBUTION

**We want**: a sense of which features were most relevant for the prediction of the model!

This is called feature attribution or feature importance

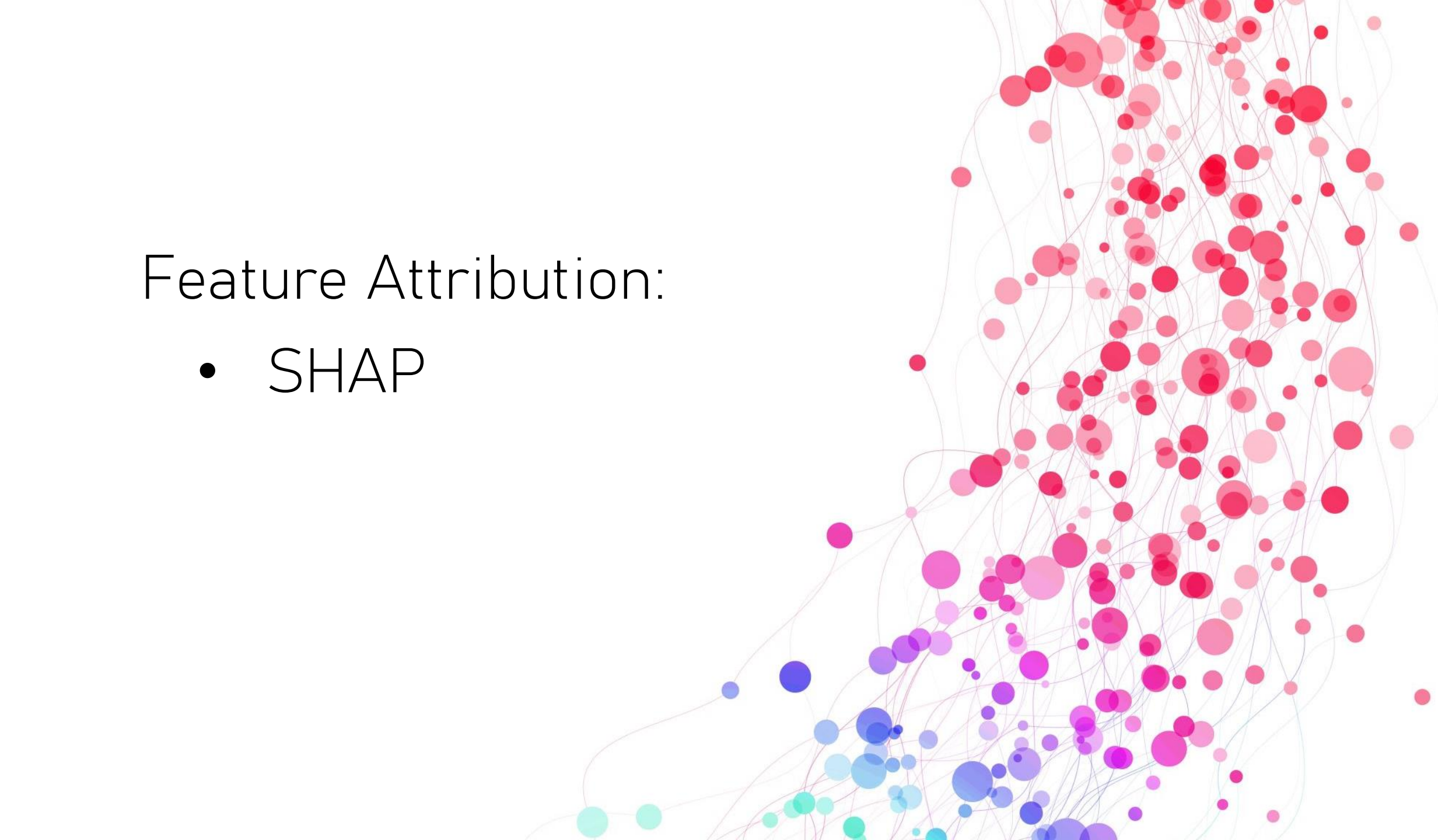


# DIFFERENT TYPES OF EXPLANATIONS



# Feature Attribution:

- SHAP



# HOW TO COMPUTE FEATURE IMPORTANCE?

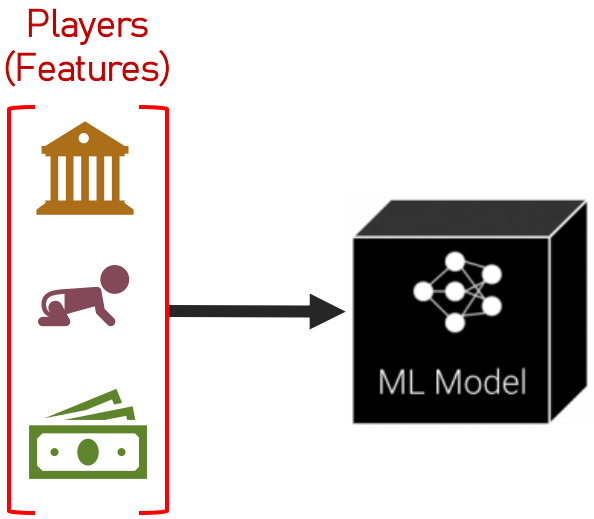
We can think of the model's prediction as a "collaborative game" where:

1. Each feature is a player

# HOW TO COMPUTE FEATURE IMPORTANCE?

We can think of the model's prediction as a "collaborative game" where:

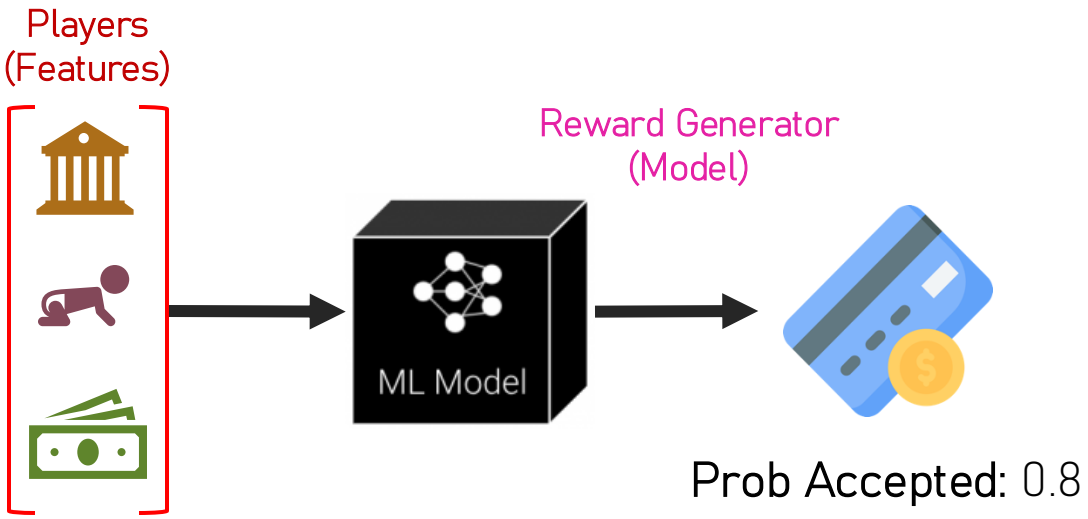
1. Each feature is a player
2. All players cooperate to produce the observed output of the model



# HOW TO COMPUTE FEATURE IMPORTANCE?

We can think of the model's prediction as a "collaborative game" where:

1. Each feature is a player
2. All players cooperate to produce the observed output of the model
3. The model's output is a reward to be distributed across all players

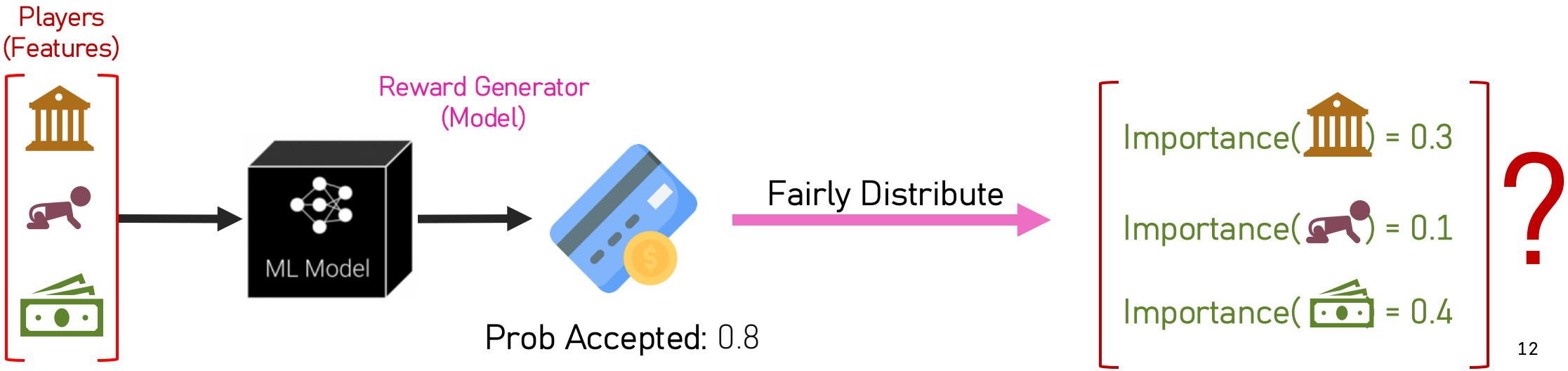


# HOW TO COMPUTE FEATURE IMPORTANCE?

We can think of the model's prediction as a "collaborative game" where:

1. Each feature is a player
2. All players cooperate to produce the observed output of the model
3. The model's output is a reward to be distributed across all players

We want to know how to "fairly" distribute this reward to players based on their contribution



# PREDICTION AS A GAME

Take a second to think about how you would go about doing this…

# PREDICTION AS A GAME

Take a second to think about how you would go about doing this...

1. What does it even mean to be fair?

All rewards should add up to exactly the model's output (but rewards can be negative!)

Interchangeable players should get equal rewards

"Null" player should get no reward

# PREDICTION AS A GAME

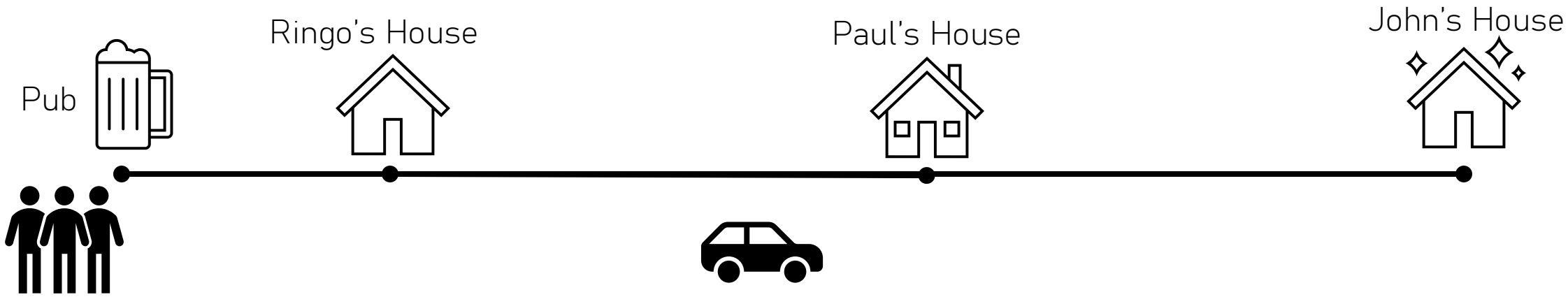
Take a second to think about how you would go about doing this…

2. How do we account for interactions?

# PREDICTION AS A GAME

Take a second to think about how you would go about doing this…

2. How do we account for interactions?



# SHAPLEY SCORES

A score that fairly assigns credit across all players/features by averaging marginal contribution of a feature across all possible coalitions

$$\phi_i = \sum_{S \subseteq F \setminus \{i\}} \frac{|S|!\,(|F| - |S| - 1)!}{|F|!} \left[ f_{S \cup \{i\}}\left(x_{S \cup \{i\}}\right) - f_S(x_S) \right]$$

$F$: the set of all features

$f_{S \cup \{i\}}$: the model trained with $i$ present

$f_S$: the model trained with $i$ withheld

$x_S$: the values of input features in subset $S$

# SHAPLEY SCORES

A score that fairly assigns credit across all players/features by averaging marginal contribution of a feature across all possible coalitions

$$\phi_i = \sum_{S \subseteq F \setminus \{i\}} \frac{|S|!\,(|F| - |S| - 1)!}{|F|!} \; [f_{S \cup \{i\}}(x_{S \cup \{i\}}) - f_S(x_S)]$$

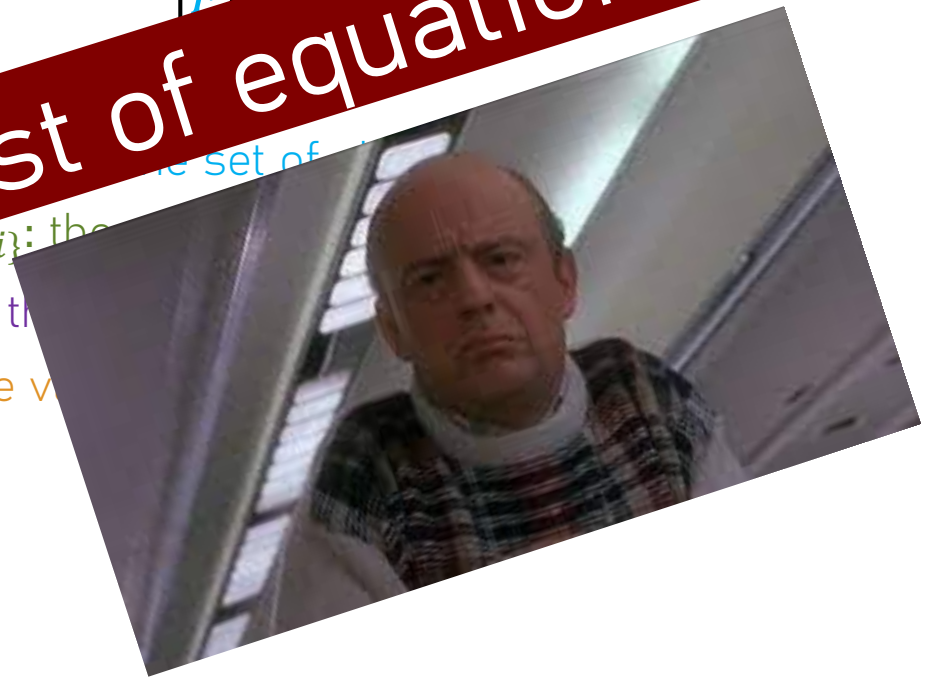
Not the nicest of equations to look at!

$S$: the set of [illegible]

$S \cup \{i\}$: the [illegible]

$f_S$: t[illegible]

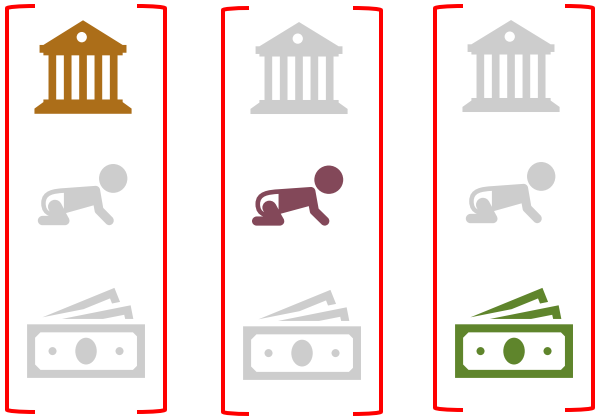
$x_S$: the v[illegible]



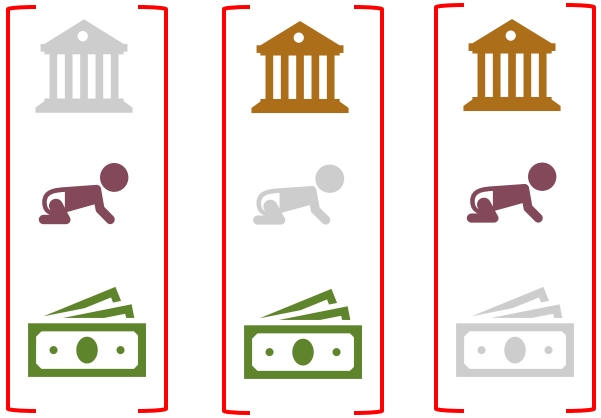
# SHAPLEY SCORES

All this equation says is that we compute a feature's importance by marginalising over its contributions across all possible subsets of features

Groups of one feature



Groups of two features

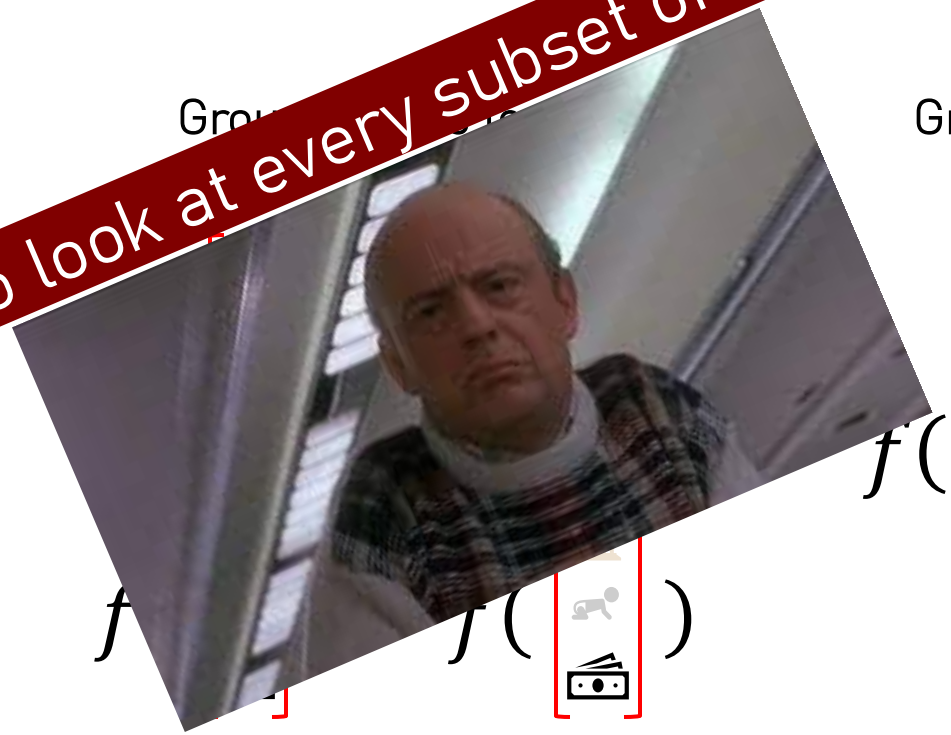


Groups of three features

# SHAPLEY SCORES

All this equation says is that we compute a feature's importance by marginalizing over its contributions across all possible subsets of features

Too expensive to look at every subset of features



Groups of one feature

Groups of three features

$$f(\quad) - f(\quad)$$

# ENTER SHAP

Shapley Additive exPlanation (SHAP) [1] efficiently estimates Shapley scores by looking at how a model's output deviates from its mean one feature at a time

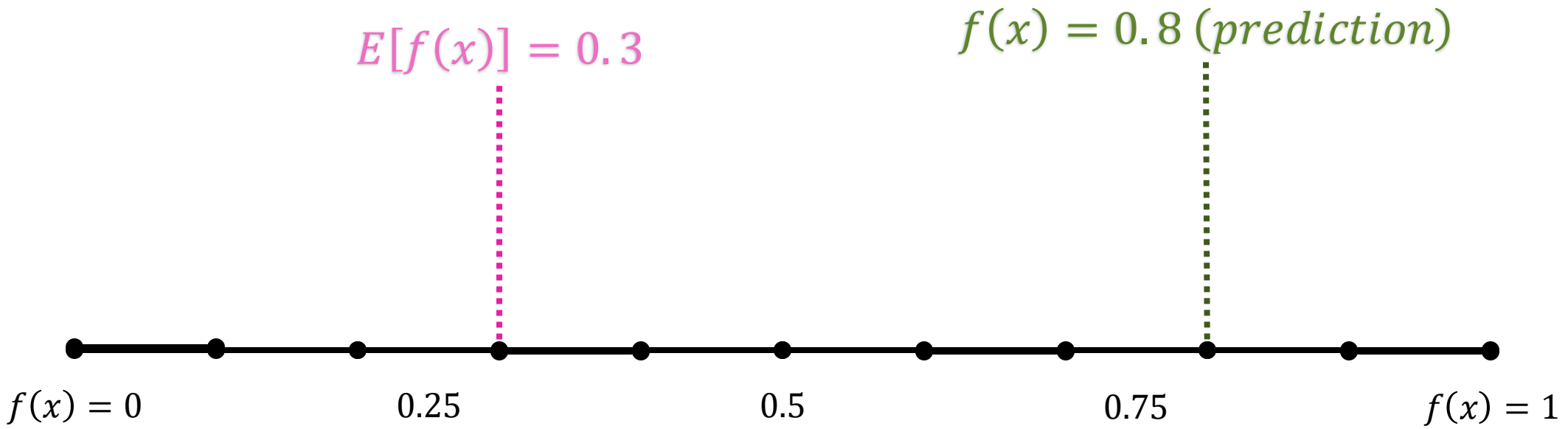
To understand this, let $f(x)$ be our credit card predictor. Consider its output on a new sample:

$$f\left(\text{  }\right) = 0.8$$

[1] Lundberg, Scott M., and Su-In Lee. "A unified approach to interpreting model predictions." *Advances in neural information processing systems* 30 (2017).

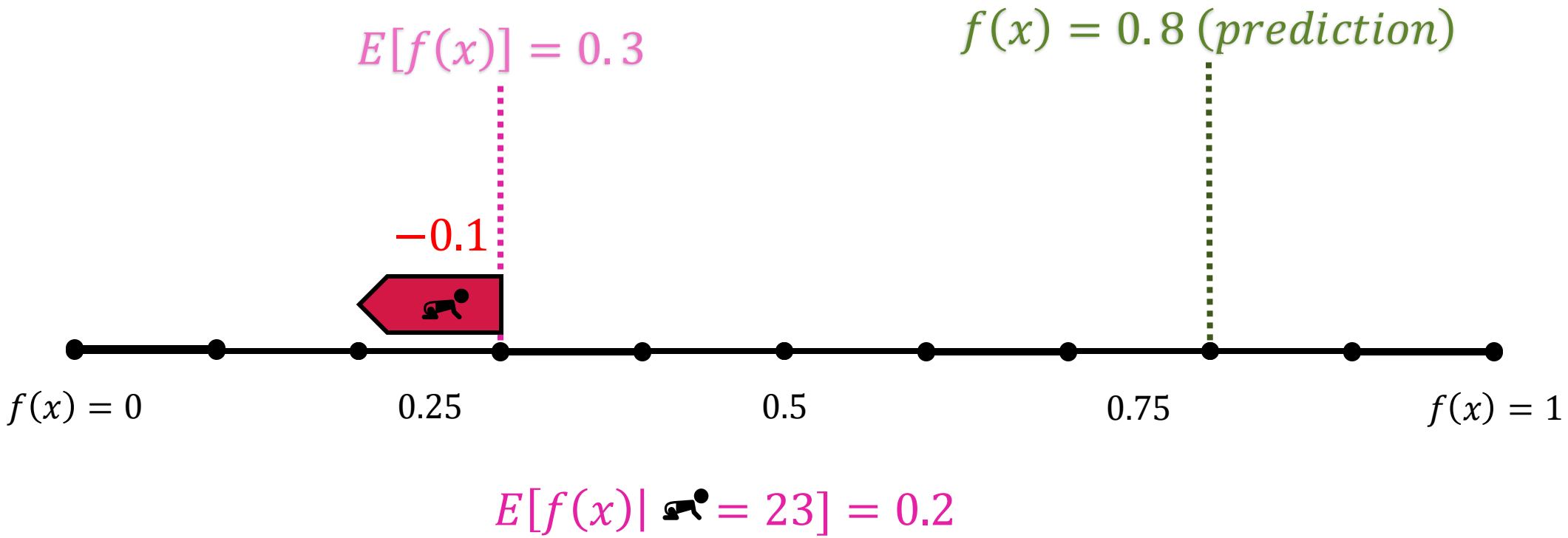
# ENTER SHAP

SHAP distributes credit for the output by assigning a score to each feature that indicates how much **on average** that feature's value caused the function to deviate from its mean:



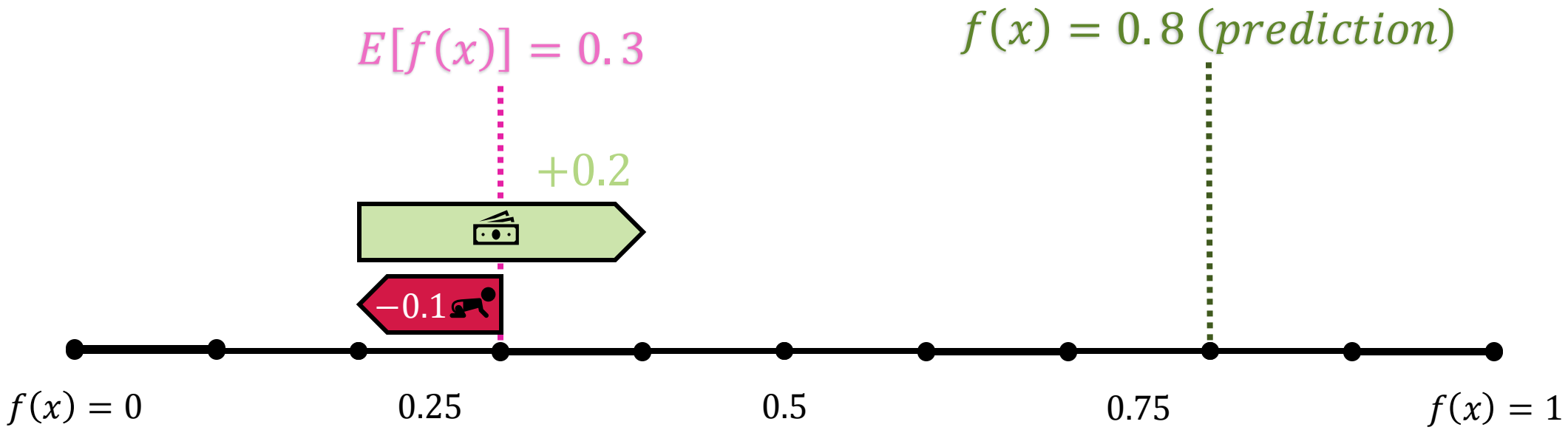
# ENTER SHAP

SHAP distributes credit for the output by assigning a score to each feature that indicates how much **on average** that feature's value caused the function to deviate from its mean:



# ENTER SHAP

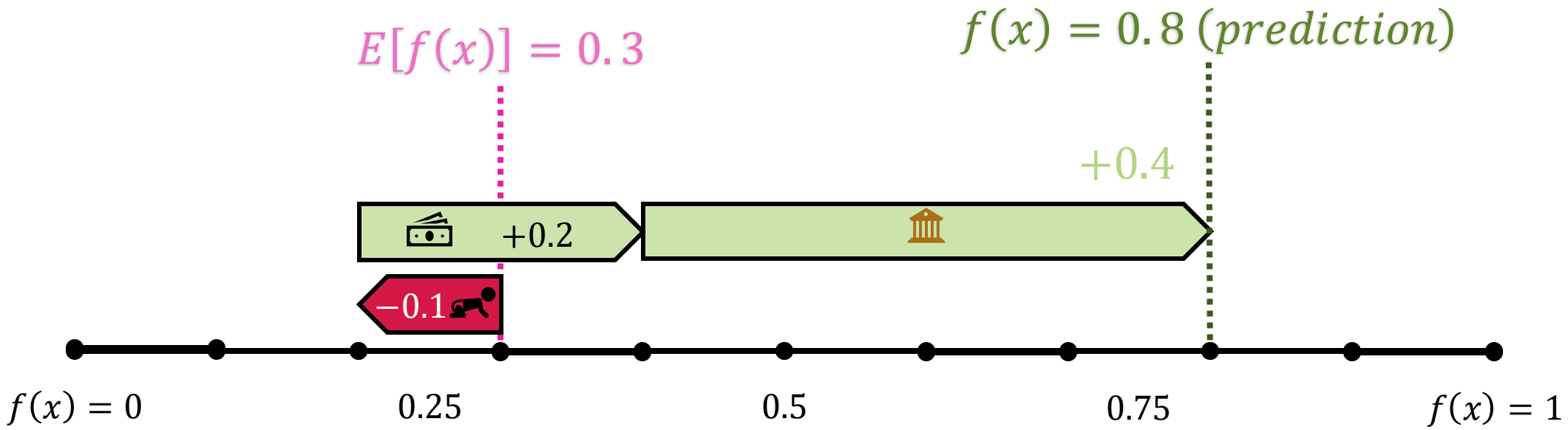
SHAP distributes credit for the output by assigning a score to each feature that indicates how much **on average** that feature's value caused the function to deviate from its mean:




$E[f(x)|$ 🦾 $= 23,$ 💵 $= 40\text{k}] = 0.4$

# ENTER SHAP

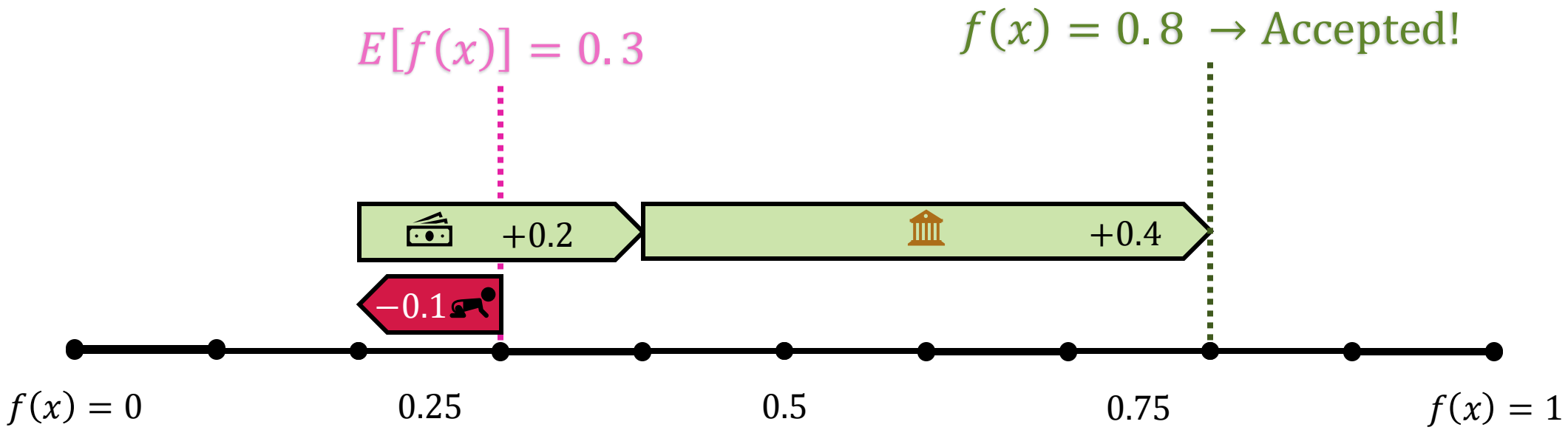
SHAP distributes credit for the output by assigning a score to each feature that indicates how much **on average** that feature's value caused the function to deviate from its mean:




$E[f(x)|$ 🔫$= 23,$ 💵 $= 40\text{k},$ 🏛 $= 4] = 0.8$

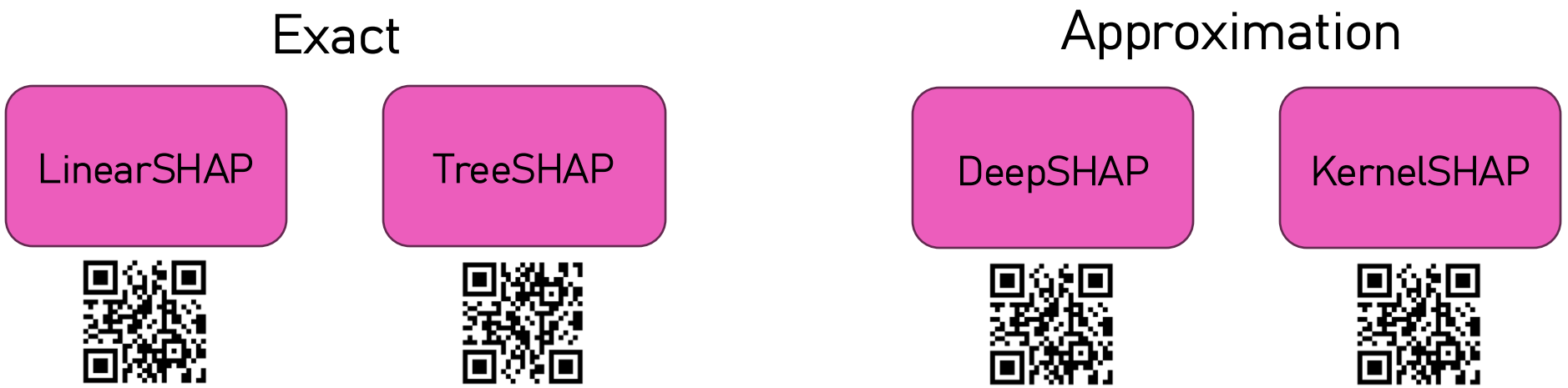
# ENTER SHAP

SHAP distributes credit for the output by assigning a score to each feature that indicates how much **on average** that feature's value caused the function to deviate from its mean:



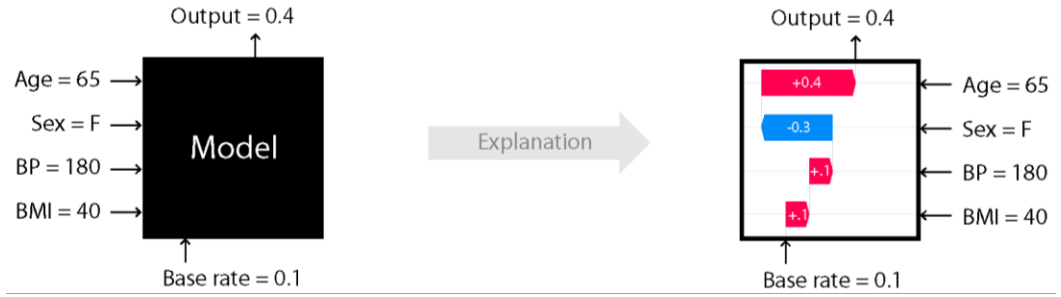
# SHAP VARIANTS

Details of how SHAP approximates Shapley values efficiently depends on the model:



# SHAP LIBRARY

SHAP has an amazing open source library which allows you to explain any black-box model and explore its behaviour using all sorts of visualisations!





Taken from O'Sullivan at towardsdatascience.com/introduction-to-shap-with-python-d27edc23c454

SHAP https://shap.readthedocs.io/en/latest/ SHAP

# ONE FRAMEWORK TO RULE THEM ALL

Not only is SHAP very useful but different instantiations of how it selects features and estimates its Shapley values allow it to generalize enough to cover 7 known methods!

LIME

Shapley Regression Values


QII

Path Explanations

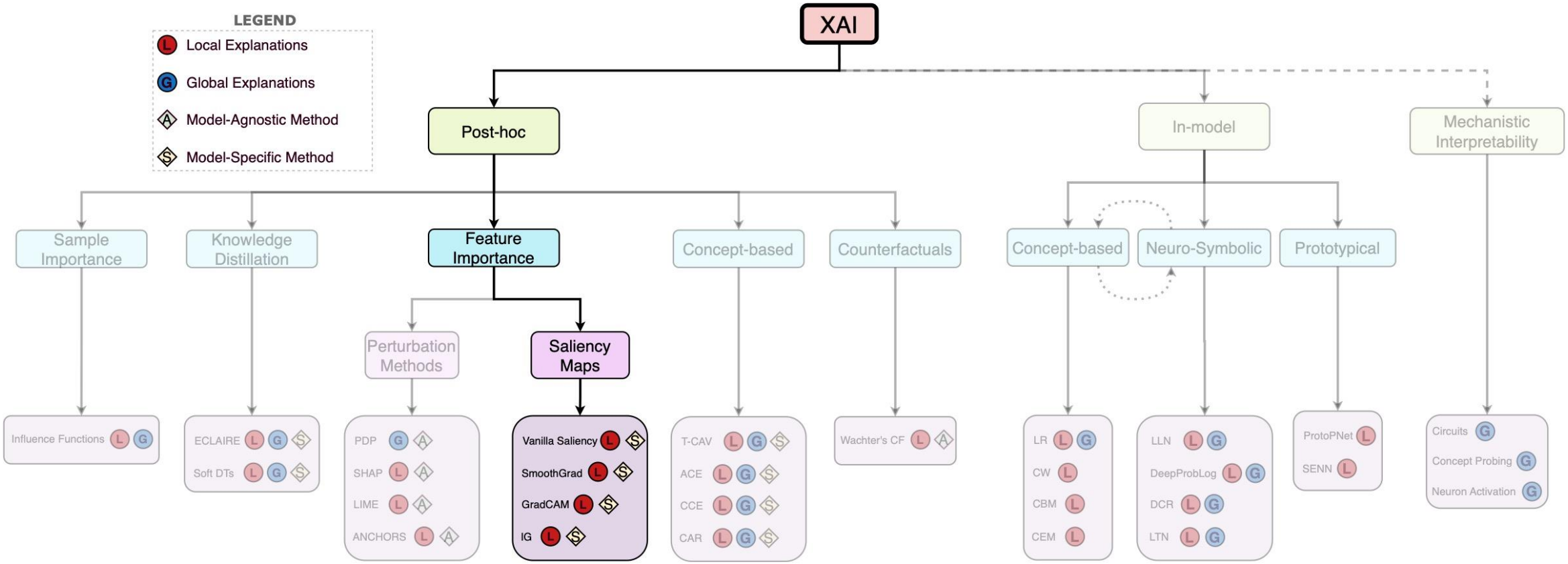
DeepLIFT

Relevance Propagation

Shapley Sampling

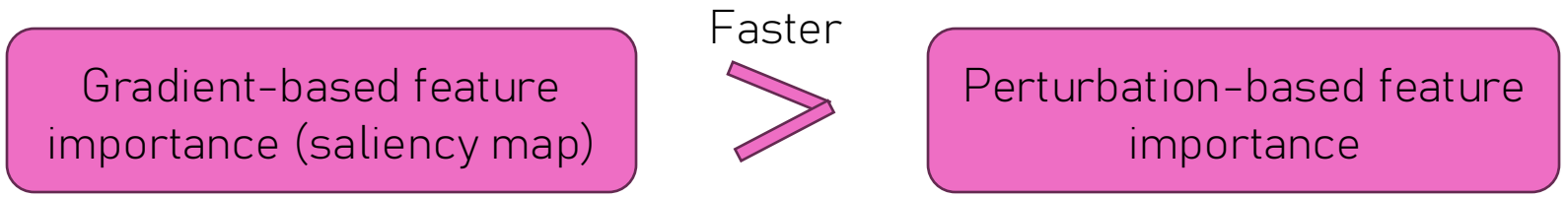
[Lundberg, Scott M., and Su-In Lee. "A unified approach to interpreting model predictions." *Advances in neural information processing systems* 30 (2017).

# DIFFERENT TYPES OF EXPLANATIONS



# EXPLAINING WITH VISUALISATIONS

- **Perturbation-based** methods such as SHAP manipulate parts of the input to generate explanation
- They rely on simplicity **assumption** (e.g., linearity) of **local** decision boundaries and can also be **computationally expensive** due to local model creation

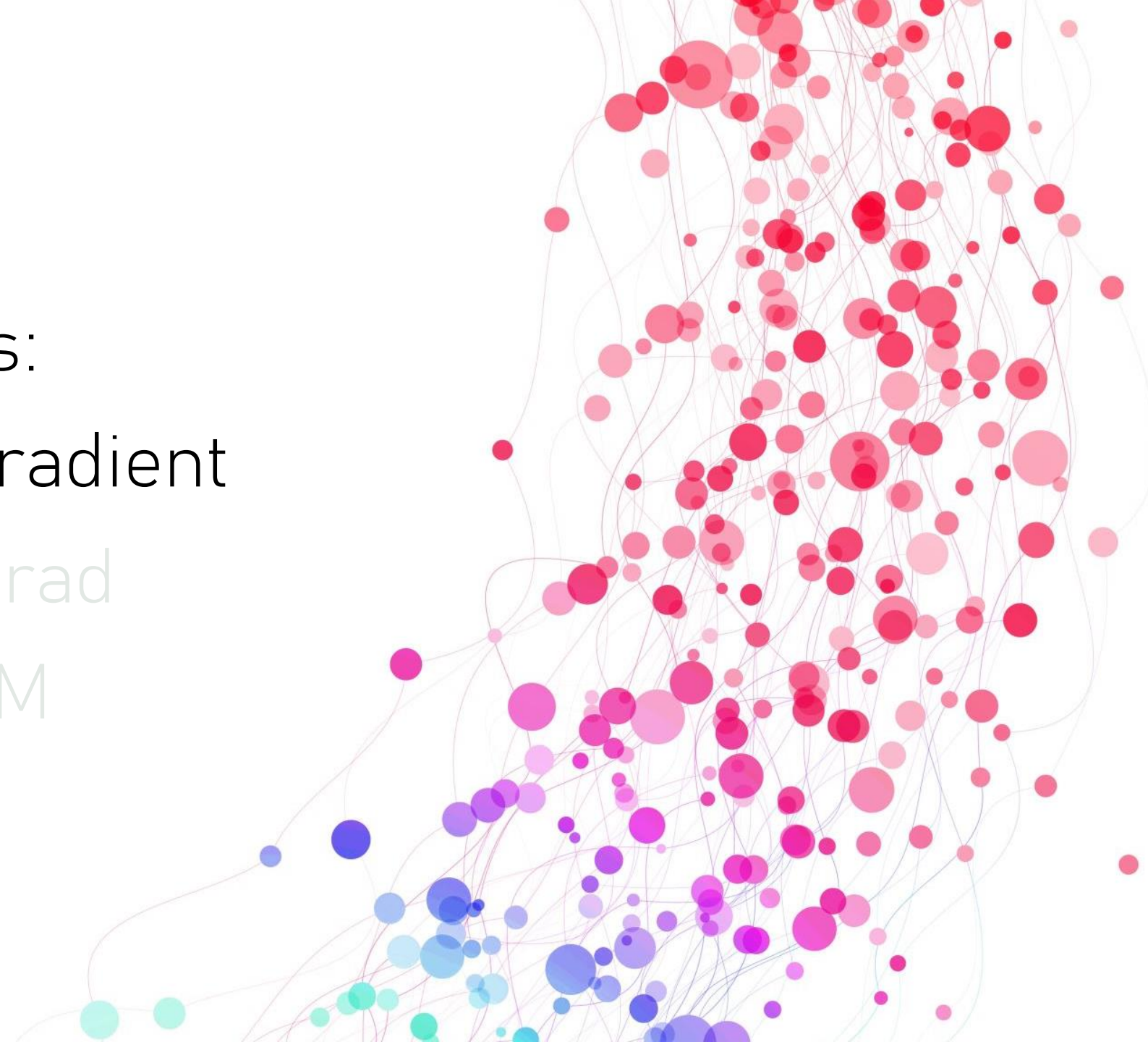


# EXPLAINING WITH VISUALISATIONS

- **Perturbation-based** methods such as SHAP manipulate parts of the input to generate explanation
- They rely on simplicity **assumption** (e.g., linearity) of **local** decision boundaries and can also be **computationally expensive** due to local model creation

Saliency Maps:

- Vanilla Gradient
- SmoothGrad
- Grad-CAM



# WHY TO CARE ABOUT THE GRADIENT AGAIN?

- Get some samples and do a forward pass through the network and calculate predictions and loss
- Calculate gradients of the loss  for all parameters of the network using back propagation
- The partial derivatives in gradient show how much each parameter needs to change to minimize loss, so the parameters are tweaked accordingly

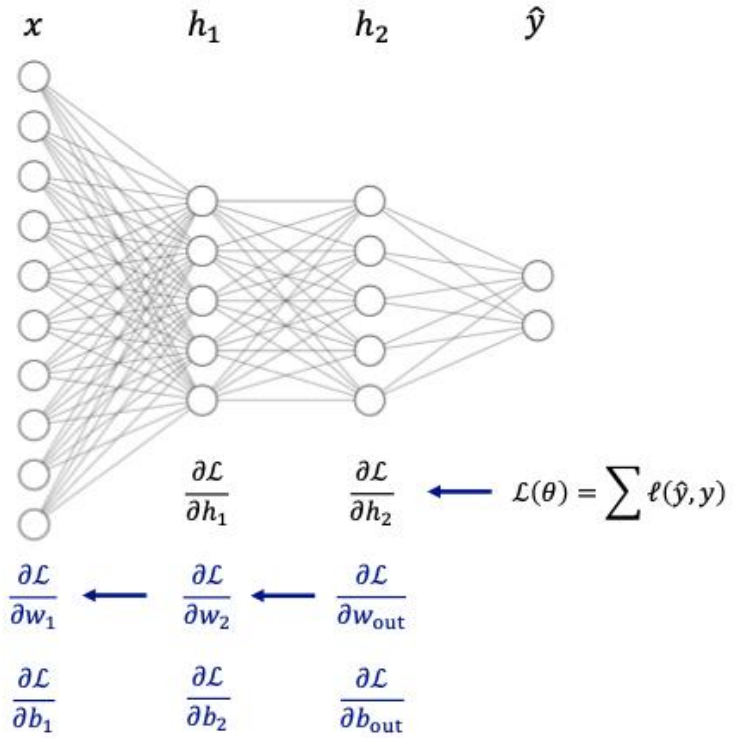
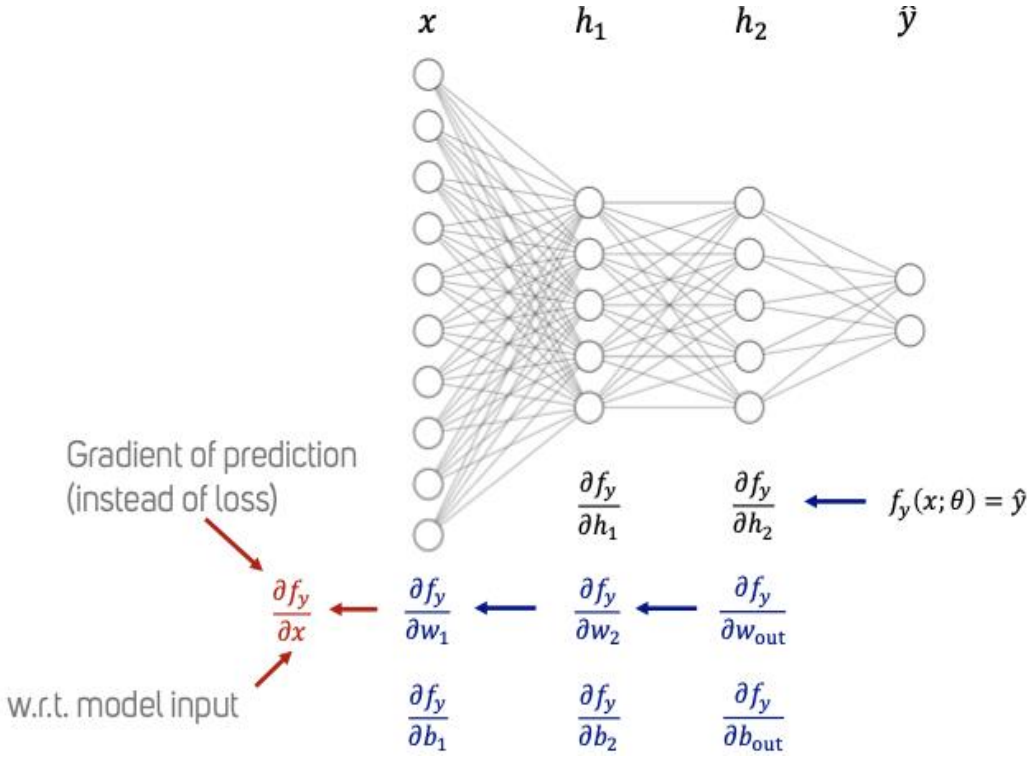



Image taken from: https://courses.cs.washington.edu/courses/csep590b/22sp/files/lectures/lecture4.pdf

# WHY TO CARE ABOUT THE GRADIENT AGAIN?

- In gradient-based explanation we use backpropogation to quantify feature importance: gradients of the predictions (instead of loss) are calculated w.r.t. inputs (instead of model parameters)

- Now the partial derivatives represent **sensitivity of output to input change**

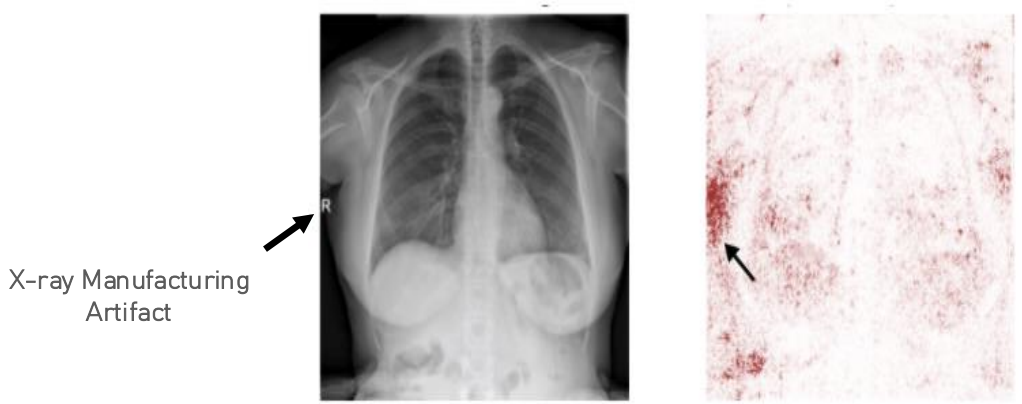



Images taken from: https://courses.cs.washington.edu/courses/csep590b/22sp/files/lectures/lecture4.pdf

# SALIENCY: FORMULATION AND APPLICATION

Vanilla gradient [1]: for an input $x$ and label $y$, we calculate the gradient of the prediction $f_y(x)$ w.r.t. input features (pixels): $\frac{\partial f_y}{\partial x_i}(x)$

Debugging in image classification: right for the wrong reason in safety critical domains




[1] Simonyan, Karen, Andrea Vedaldi, and Andrew Zisserman. "Deep inside convolutional networks: Visualising image classification models and saliency maps." *arXiv preprint arXiv:1312.6034* (2013).
Image adapted from: DeGrave, Alex J., Joseph D. Janizek, and Su-In Lee. "AI for radiographic COVID-19 detection selects shortcuts over signal." Nature Machine Intelligence 3.7 (2021): 610-619.

# PROBLEM WITH VANILLA GRADIENT:

- ReLU saturation problem: inputs that contributed to the output negatively may be disregarded and their attribution may be concealed
- There is also the problem of noisy gradient and saliency maps not being sharp enough
- These problem has given rise to a large family of alternative gradient-based methods

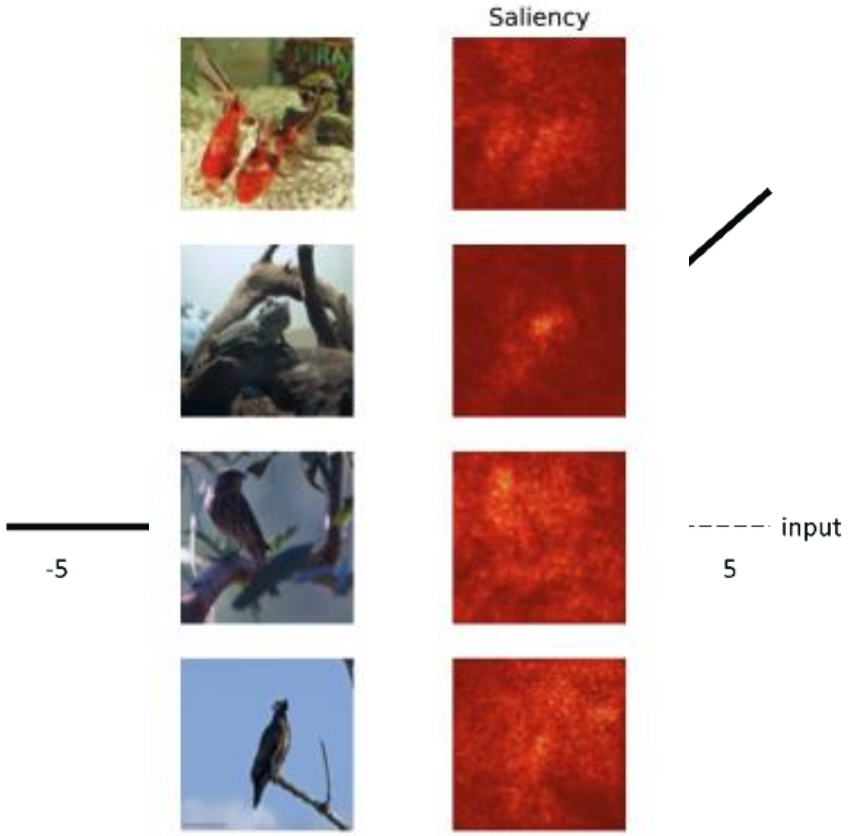



Image adapted from: Nielsen, Ian E., et al. "Robust explainability: A tutorial on gradient-based attribution methods for deep neural networks." *IEEE Signal Processing Magazine* 39.4 (2022): 73-84.

Saliency Maps:

- Vanilla Gradient
- SmoothGrad
- Grad-CAM

# SMOOTHGRAD I

Intuition: the main idea behind SmoothGrad [1] is to remove noise by adding noise!

Formulation:

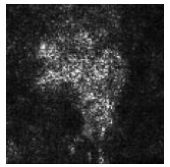
- For an image of interest we create multiple versions by adding noise
- For each version we get the saliency map
- We average over all of them

$$M_c(x) = \frac{\partial f_c(x)}{\partial x}$$

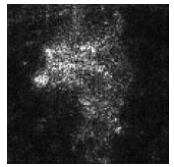
$$\widehat{M}_c(x) = \frac{1}{n}\sum_{1}^{n} M_c(x + \mathcal{N}(0, \sigma^2))$$

[1] Smilkov, Daniel, et al. "SmoothGrad: removing noise by adding noise." arXiv preprint arXiv:1706.03825 (2017).

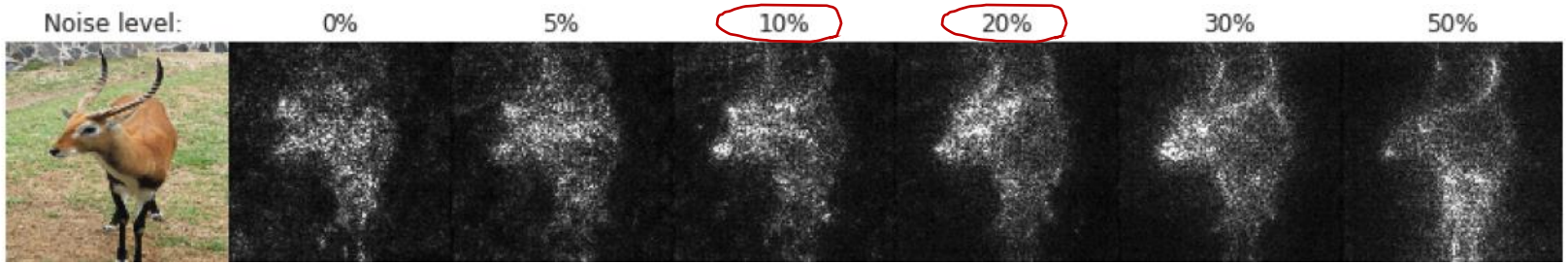
# SMOOTHGRAD EXAMPLE






$n = 50$

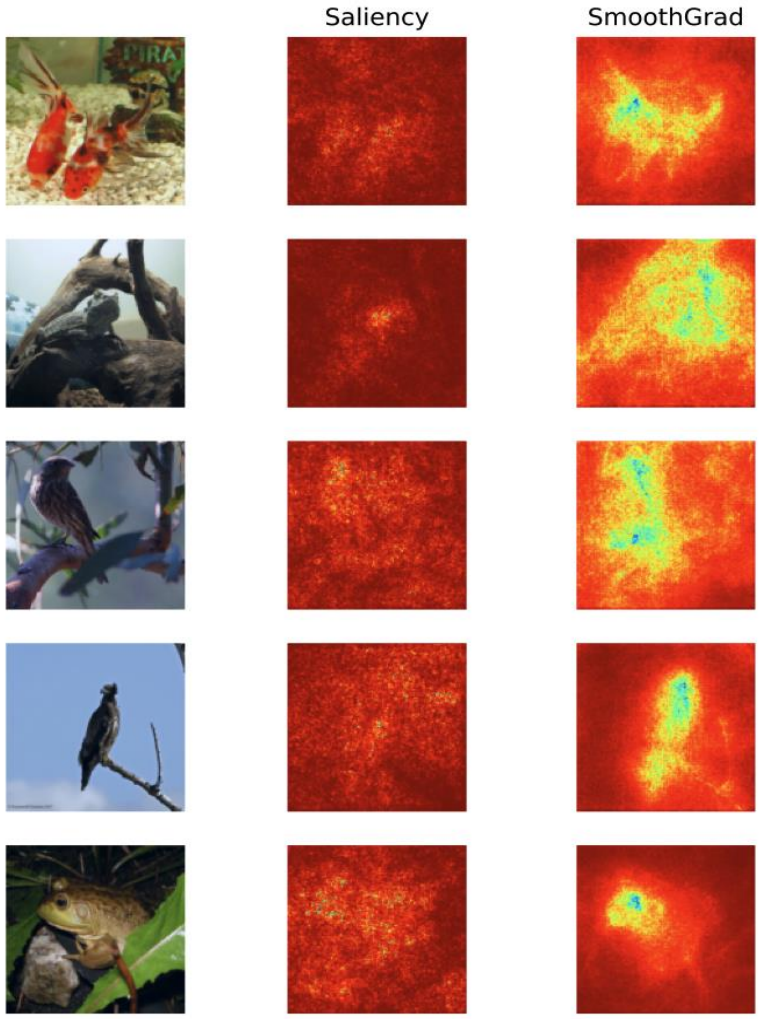



$$\mathcal{N}(0, \sigma^2)$$

$$0.1 < \frac{\sigma}{x_{max} - x_{min}} < 0.2$$

[1] Smilkov, Daniel, et al. "SmoothGrad: removing noise by adding noise." arXiv preprint arXiv:1706.03825 (2017).
Image taken from [1]

# SMOOTHGRAD II




This is improvement!

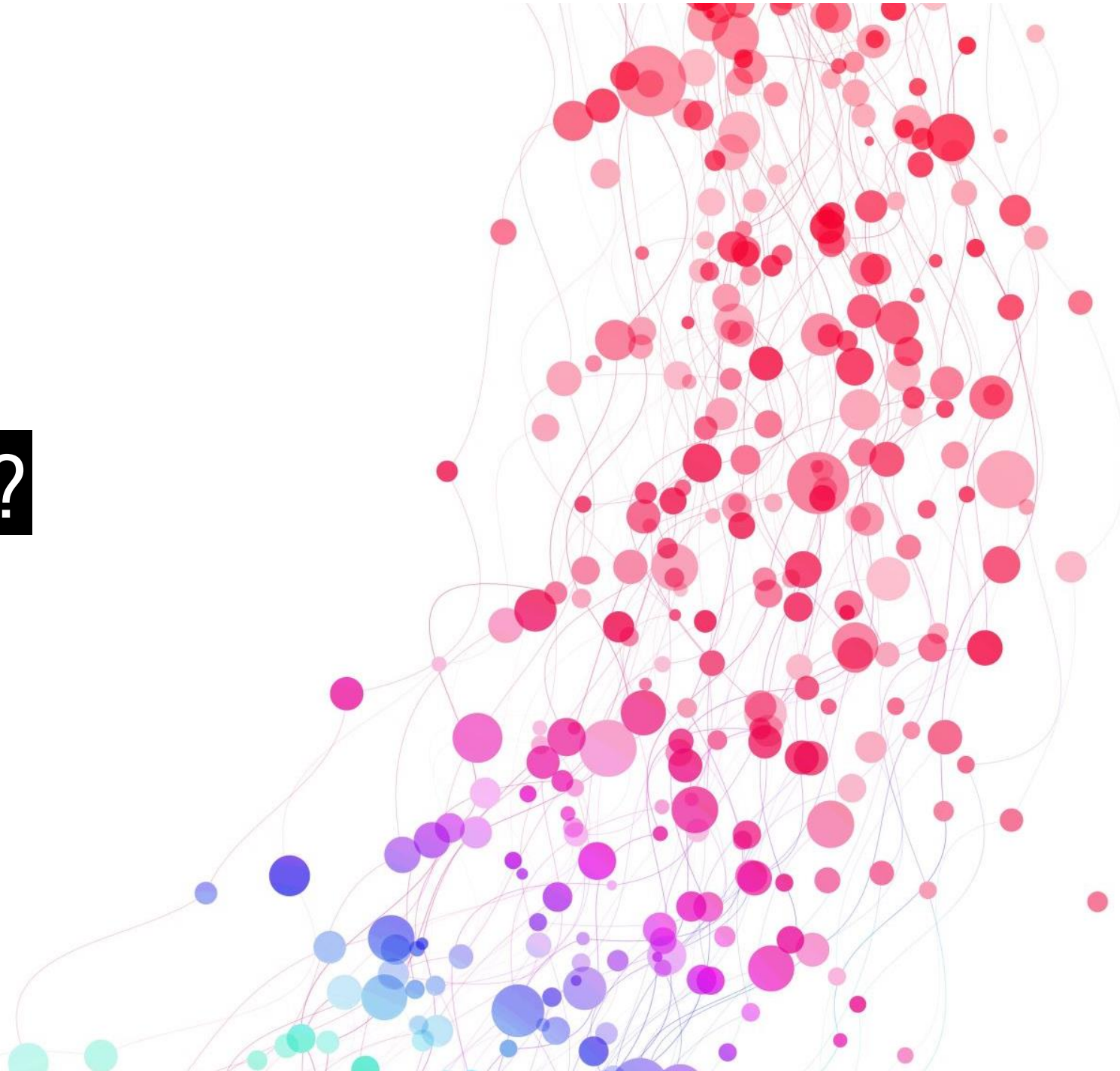
But we are still highlighting importance at pixel level

Can we improve further?

From pixel-based feature importance to important semantic features

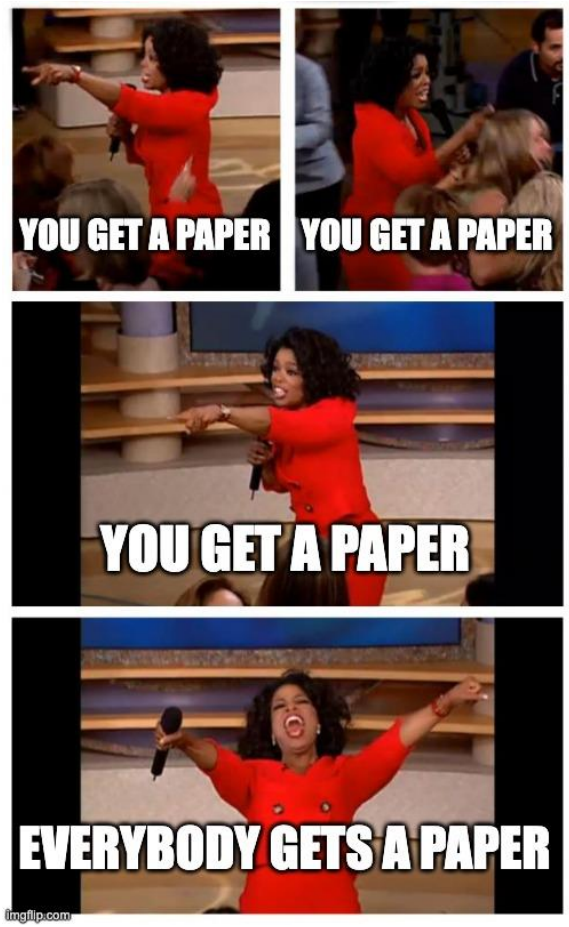
Image adapted from Nielsen, Ian E., et al. "Robust explainability: A tutorial on gradient-based attribution methods for deep neural networks." *IEEE Signal Processing Magazine* 39.4 (2022): 73-84.

# QUESTIONS?



# PAPER DISTRIBUTION TIME!

We will have two student paper presentations per lecture



# PAPER DISTRIBUTION TIME!

How many accepted NeurIPS papers had “Language Model” in their titles in 2024?

(Hint: there were a total of 4540 papers accepted)

# PAPER DISTRIBUTION TIME!

How many accepted NeurIPS papers had “Language Model” in their titles in 2024?

(Hint: there were a total of 4540 papers accepted)

Answer: 391

# PAPER DISTRIBUTION TIME!

Paper 1: Ribeiro et al. "**Anchors: High-precision model-agnostic explanations.**" AAAI 2018.

Paper 2: Sundararajan et al. "**Axiomatic attribution for deep networks.**" ICML 2017.