Interpretable Machine Learning

Introduction to Loss-based Feature Importance

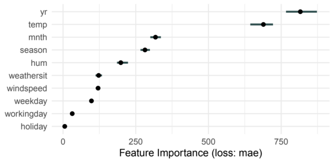


Figure: Bike Sharing Dataset

Learning goals

- Understand motivation for feature importance
- Develop an intuition for possible use-cases
- Know characteristics of feature importance methods

Interpretable Machine Learning

Feature Importances 1
Intro to Loss-based Feature Importance

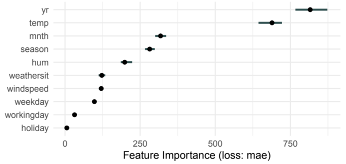
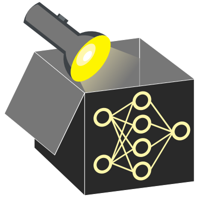


Figure: Bike Sharing Dataset

Learning goals

- Understand motivation for feature importance
- Develop an intuition for possible use-cases
- Know characteristics of feature importance methods

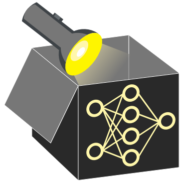
MOTIVATION



- **Feature effects** describe how a feature x_j influences the prediction $\hat{y}$
  - requires one plot per feature (e.g., PDPs, ALEs)
  - purely model-based; ignores true target y

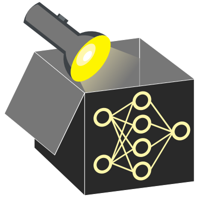
MOTIVATION

- **Feature effects** describe how a feature x_j influences the prediction $\hat{y}$
  - requires one plot per feature (e.g., PDPs, ALEs)
  - purely model-based; ignores true target y



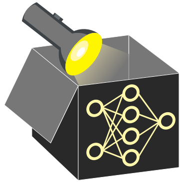
Interpretable Machine Learning – 1 / 8

MOTIVATION



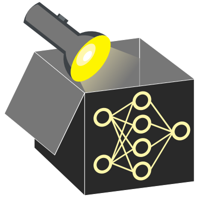
- **Feature effects** describe how a feature x_j influences the prediction $\hat{y}$
  - requires one plot per feature (e.g., PDPs, ALEs)
  - purely model-based; ignores true target y
- **Feature importance** quantifies how much each x_j contributes to prediction error
  - condenses information into one number per feature
  - typically compares prediction errors (involves y) with and without feature

MOTIVATION



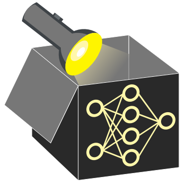
- **Feature effects** describe how a feature x_j influences the prediction $\hat{y}$
  - requires one plot per feature (e.g., PDPs, ALEs)
  - purely model-based; ignores true target y
- **Feature importance** quantifies how much each x_j contributes to prediction error
  - condenses information into one number per feature
  - typically compares prediction errors (involves y) with/without feature

MOTIVATION



- **Feature effects** describe how a feature x_j influences the prediction $\hat{y}$
  - requires one plot per feature (e.g., PDPs, ALEs)
  - purely model-based; ignores true target y
- **Feature importance** quantifies how much each x_j contributes to prediction error
  - condenses information into one number per feature
  - typically compares prediction errors (involves y) with and without feature
- **Clarification:** By *feature importance*, we mean *loss-based* methods that assess a feature's impact via changes in *prediction error*.
  $\rightsquigarrow$ Other notions exist (e.g., variance-based methods; see ▸ Greenwell et al. (2020)).

MOTIVATION

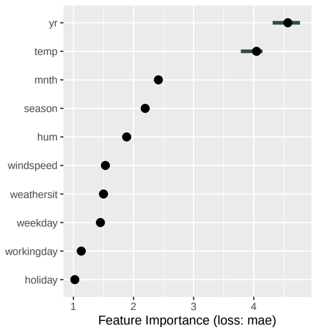


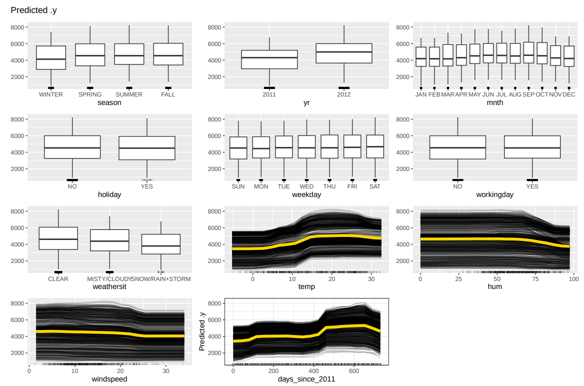
- **Feature effects** describe how a feature x_j influences the prediction $\hat{y}$
  - requires one plot per feature (e.g., PDPs, ALEs)
  - purely model-based; ignores true target y
- **Feature importance** quantifies how much each x_j contributes to prediction error
  - condenses information into one number per feature
  - typically compares prediction errors (involves y) with/without feature
- **Clarification:** By *feature importance*, we mean *loss-based* methods that assess a feature's impact via changes in *prediction error*.
  $\rightsquigarrow$ Other notions exist (e.g., variance-based methods; see ▸ Greenwell 2020).

EXAMPLE

Feature importance offers condensed summary of feat. relevance w.r.t. performance

- Fit random forest on bike sharing data
- Left: Feature importance ranking by permutation feature importance (PFI)
- Right: Feature effects for all features

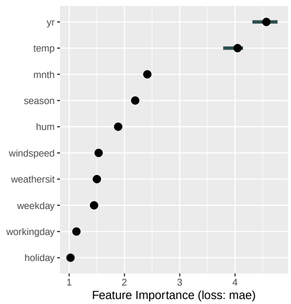


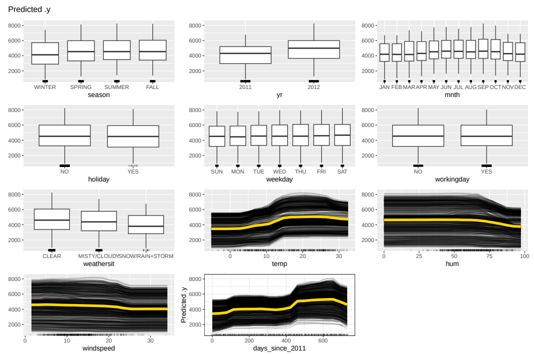


EXAMPLE

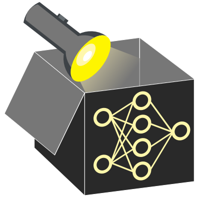
Feature importance provides a condensed summary of feature relevance w.r.t. performance

- Fit random forest on bike sharing data
- Left: Feature importance ranking by permutation feature importance (PFI)
- Right: Feature effects for all features





FEATURE IMPORTANCE - DIFFERENCES ▸ Ewald et al. 2024

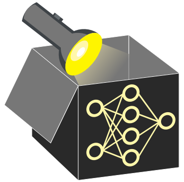


Many loss-based feature importance methods exist, which mainly differ in

(1) How they "remove" or "perturb" the feature of interest (FOI) X_j

- **Remove X_j and refit:** Drop the X_j and retrain model without it
- **Perturb X_j:** Replace X_j by $\tilde{X}_j$ sampled from *marginal* or *conditional* distribution
- **Marginalize X_j:** integrate out X_j via *marginal* or *conditional* distribution

FEATURE IMPORTANCE - DIFFERENCES ▸ EWALD_2024

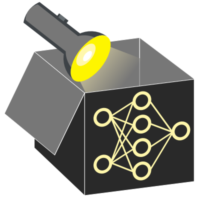


Many loss-based feature importance methods exist, which mainly differ in

(1) How they "remove" or "perturb" the feature of interest (FOI) X_j

- **Remove X_j and refit:** Drop the X_j and retrain model without it
- **Perturb X_j:** Replace X_j by $\tilde{X}_j$ sampled from *marginal*/*conditional* distrib.
- **Marginalize X_j:** integrate out X_j via *marginal* or *conditional* distribution

FEATURE IMPORTANCE - DIFFERENCES ▸ Ewald et al. 2024



Many loss-based feature importance methods exist, which mainly differ in

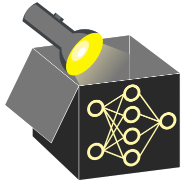
(1) How they "remove" or "perturb" the feature of interest (FOI) X_j

- **Remove X_j and refit:** Drop the X_j and retrain model without it
- **Perturb X_j:** Replace X_j by $\tilde{X}_j$ sampled from *marginal* or *conditional* distribution
- **Marginalize X_j:** integrate out X_j via *marginal* or *conditional* distribution

(2) How they compare model performance before and after feature removal

- **Compare "reduced model" without FOI vs. full model**:
  Measure drop in performance when FOI is "removed"
  ⇝ Similar idea as backward feature elimination
- **Compare "empty model" (no features) vs. model with only FOI**:
  Measure gain in performance when only FOI is used
  ⇝ Similar idea as forward feature selection
- **Compare models with/without FOI across different feature sets**:
  Measure average contribution when FOI joins any feature set (Shapley-based)

FEATURE IMPORTANCE - DIFFERENCES ▸ EWALD_2024



Many loss-based feature importance methods exist, which mainly differ in

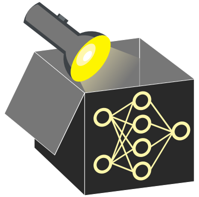
(1) How they "remove" or "perturb" the feature of interest (FOI) X_j

- **Remove X_j and refit:** Drop the X_j and retrain model without it
- **Perturb X_j:** Replace X_j by $\tilde{X}_j$ sampled from *marginal*/*conditional* distrib.
- **Marginalize X_j:** integrate out X_j via *marginal* or *conditional* distribution

(2) How they compare model performance before and after feat. removal

- **Compare "reduced model" without FOI vs. full model**:
  Measure drop in performance when FOI is "removed"
  ⇝ Similar idea as backward feature elimination
- **Compare "empty model" (no features) vs. model with only FOI**:
  Measure gain in performance when only FOI is used
  ⇝ Similar idea as forward feature selection
- **Compare models with/without FOI across different feature sets**:
  Measure average contrib. when FOI joins any feat. set (Shapley-based)

FEATURE IMPORTANCE - DIFFERENCES ▸ Ewald et al. 2024



Many loss-based feature importance methods exist, which mainly differ in

(1) How they "remove" or "perturb" the feature of interest (FOI) X_j

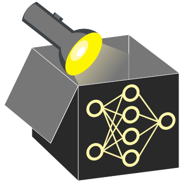
- **Remove X_j and refit:** Drop the X_j and retrain model without it
- **Perturb X_j:** Replace X_j by $\tilde{X}_j$ sampled from *marginal* or *conditional* distribution
- **Marginalize X_j:** integrate out X_j via *marginal* or *conditional* distribution

(2) How they compare model performance before and after feature removal

- **Compare "reduced model" without FOI vs. full model**:
  Measure drop in performance when FOI is "removed"
  $\rightsquigarrow$ Similar idea as backward feature elimination
- **Compare "empty model" (no features) vs. model with only FOI**:
  Measure gain in performance when only FOI is used
  $\rightsquigarrow$ Similar idea as forward feature selection
- **Compare models with/without FOI across different feature sets**:
  Measure average contribution when FOI joins any feature set (Shapley-based)

Depending on the different removal/perturbation and comparison strategies, feature importance methods provide insight into different aspects of model and data.

FEATURE IMPORTANCE - DIFFERENCES ▸ EWALD_2024



Many loss-based feature importance methods exist, which mainly differ in

(1) How they "remove" or "perturb" the feature of interest (FOI) X_j

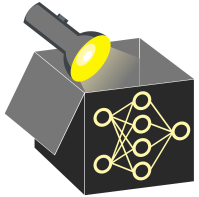
- **Remove X_j and refit:** Drop the X_j and retrain model without it
- **Perturb X_j:** Replace X_j by $\tilde{X}_j$ sampled from *marginal*/*conditional* distrib.
- **Marginalize X_j:** integrate out X_j via *marginal* or *conditional* distribution

(2) How they compare model performance before and after feat. removal

- **Compare "reduced model" without FOI vs. full model**:
  Measure drop in performance when FOI is "removed"
  $\rightsquigarrow$ Similar idea as backward feature elimination
- **Compare "empty model" (no features) vs. model with only FOI**:
  Measure gain in performance when only FOI is used
  $\rightsquigarrow$ Similar idea as forward feature selection
- **Compare models with/without FOI across different feature sets**:
  Measure average contrib. when FOI joins any feat. set (Shapley-based)

Depending on the different removal/perturbation and comparison strategies, feat. imp. methods provide insight into different aspects of model and data.

POTENTIAL INTERPRETATION GOALS



Feature importance methods provide condensed insights, but only into specific aspects of model and data. Interpretation goals often differ and typically address non-overlapping questions (except for special cases).

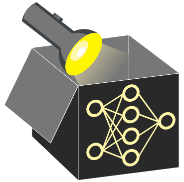
For example, one may be interested in getting insight into whether the ...

(1) feature x_j is causal for the prediction?

(2) feature x_j contains prediction-relevant information about y?

(3) model requires access to x_j to achieve it's prediction performance?

POTENTIAL INTERPRETATION GOALS



Feature importance methods provide condensed insights, but only into specific aspects of model and data. Interpretation goals often differ and typically address non-overlapping questions (except for special cases).

For example, one may be interested in getting insight into whether the ...

(1) feature x_j is causal for the prediction?

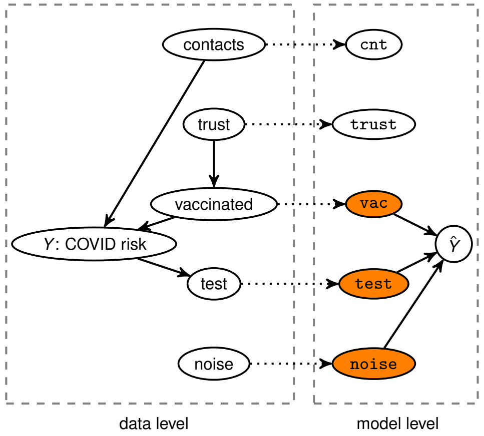
(2) feature x_j contains prediction-relevant information about y?

(3) model requires access to x_j to achieve it's prediction performance?

EXAMPLE: CAUSAL FOR THE PREDICTION (1)

A feature may be causal for $\hat{y}$ (1) without containing prediction-relevant information about y (2)

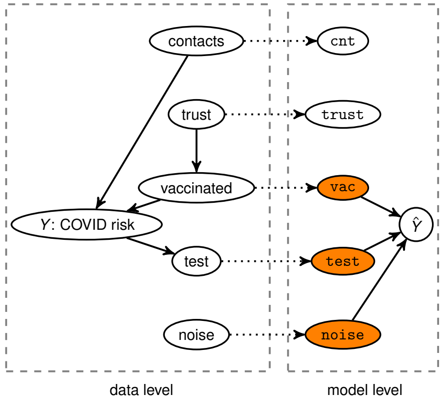
Example: overfitting due to noisy features



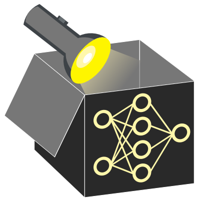
EXAMPLE: CAUSAL FOR THE PREDICTION (1)

A feature may be causal for $\hat{y}$ (1) without containing prediction-relevant information about y (2)

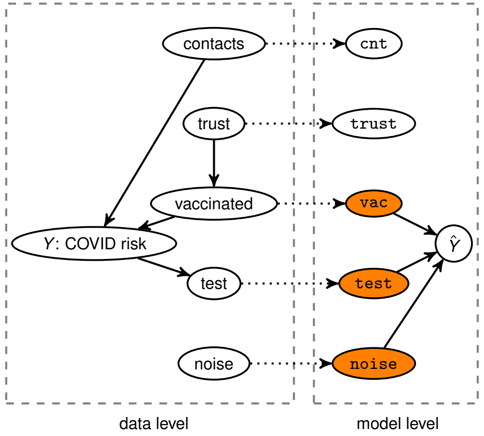
Example: overfitting due to noisy features



EXAMPLE: CAUSAL FOR THE PREDICTION (1)



A feature may be causal for $\hat{y}$ (1) without containing prediction-relevant information about y (2)

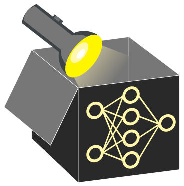


Example: overfitting due to noisy features

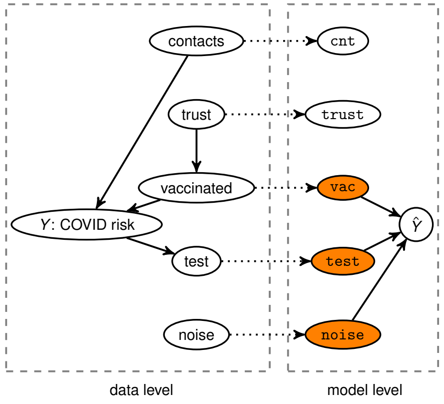
- All features used by the model are of interest
- Here: Model uses feature `noise`, although it does not contain prediction-relevant information about y (data level)

$\Rightarrow$ Overfitted models may use many noise features which are deemed relevant on model level (but not on data level)

EXAMPLE: CAUSAL FOR THE PREDICTION (1)



A feature may be causal for $\hat{y}$ (1) without containing prediction-relevant information about y (2)



Example: overfitting due to noisy features

- All features used by the model are of interest
- Here: Model uses feature `noise`, although it does not contain prediction-relevant information about y (data level)

$\Rightarrow$ Overfitted models may use many noise features which are deemed relevant on model level (but not on data level)

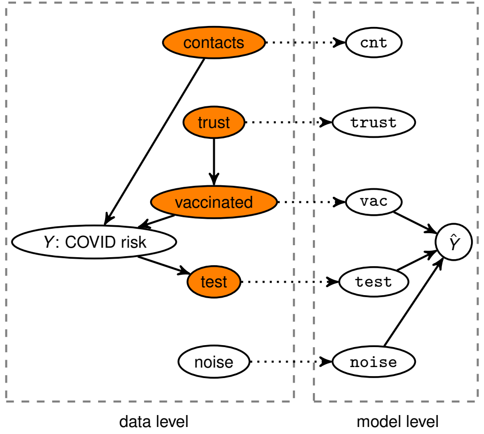
©

Interpretable Machine Learning – 5 / 8

EXAMPLE: PREDICTION-RELEVANT INFORMATION (2)

A feature may contain prediction-relevant information (2) without causing the prediction (1)

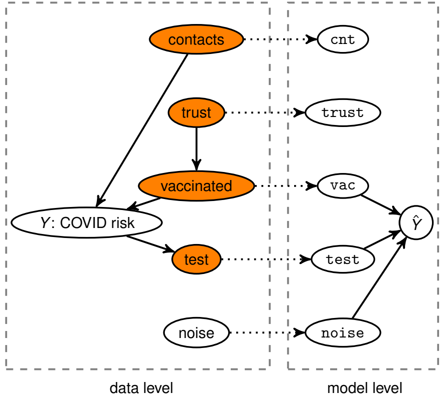
Example: underfitting, model multiplicity



EXAMPLE: PREDICTION-RELEVANT INFORMATION (2)

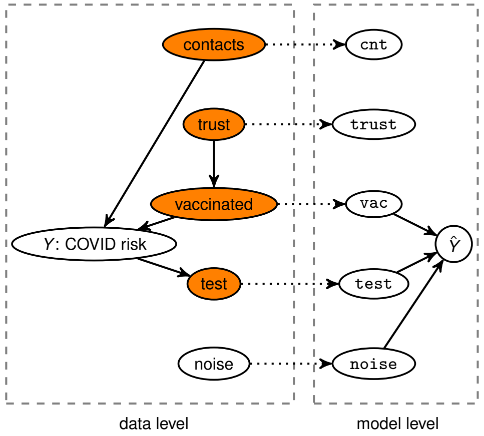
A feature may contain prediction-relevant information (2) without causing the prediction (1)

Example: underfitting, model multiplicity



EXAMPLE: PREDICTION-RELEVANT INFORMATION (2)

A feature may contain prediction-relevant information (2) without causing the prediction (1)



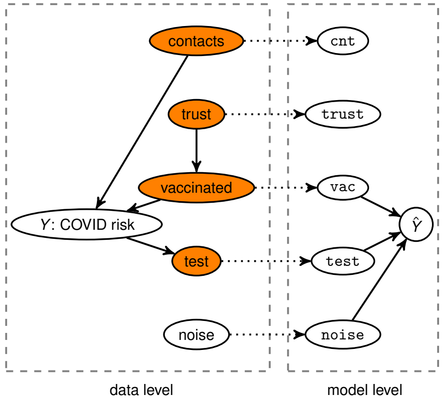
Example: underfitting, model multiplicity

- All prediction-relevant features for y are of interest
- Example: All features that are directly or indirectly (i.e., via another feature) connected to y

$\Rightarrow$ Underfitted models may ignore prediction-relevant features such as `contacts` here

EXAMPLE: PREDICTION-RELEVANT INFORMATION (2)

A feature may contain prediction-relevant information (2) without causing the prediction (1)



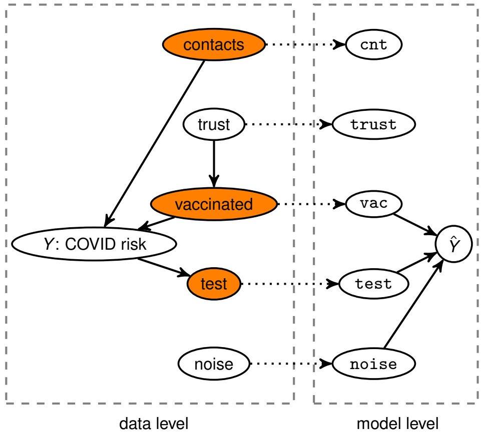
Example: underfitting, model multiplicity

- All prediction-relevant features for y are of interest
- Example: All features that are directly or indirectly (i.e., via another feature) connected to y

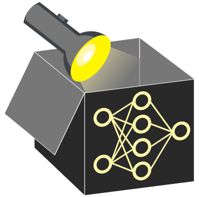
$\Rightarrow$ Underfitted models may ignore prediction-relevant features such as `contacts` here

EXAMPLE: REQUIRES ACCESS TO FEATURE (3)

A feature may contain prediction-relevant information (2), without the model requiring access to the feature for (optimal) prediction performance (3)

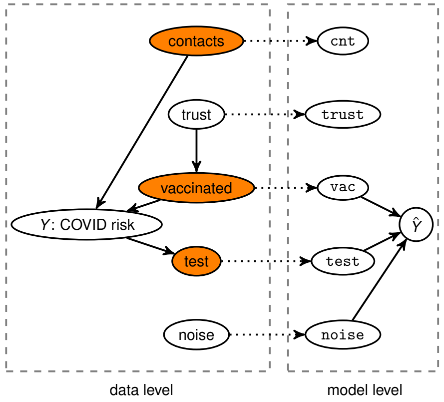


Example: correlations, confounding

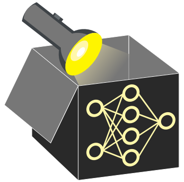


EXAMPLE: REQUIRES ACCESS TO FEATURE (3)

A feature may contain prediction-relevant information (2), without the model requiring access to the feature for (optimal) prediction performance (3)

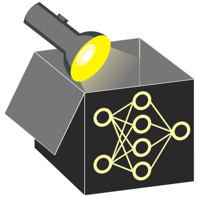


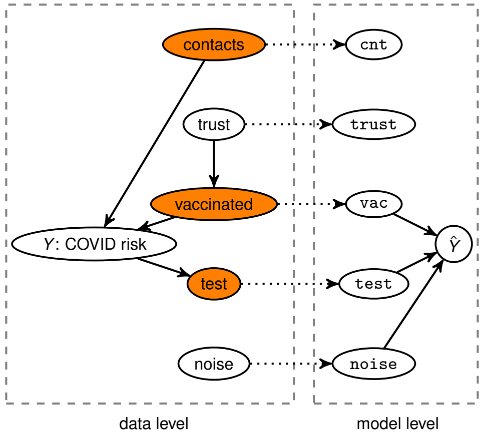
Example: correlations, confounding



EXAMPLE: REQUIRES ACCESS TO FEATURE (3)

A feature may contain prediction-relevant information (2), without the model requiring access to the feature for (optimal) prediction performance (3)





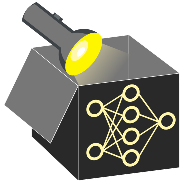
Example: correlations, confounding

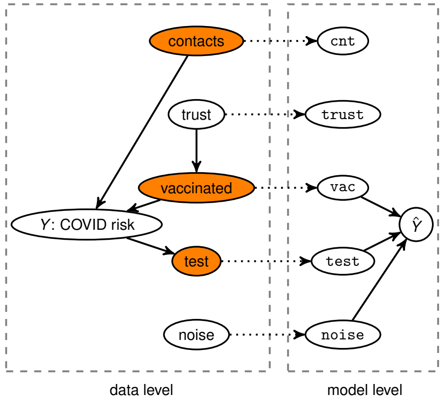
- All unique prediction-relevant features for y are of interest
- Example: All features that are directly connected to y

$\Rightarrow$ `trust` and `vaccinated` may be correlated but only `vaccinated` is directly connected to y

EXAMPLE: REQUIRES ACCESS TO FEATURE (3)

A feature may contain prediction-relevant information (2), without the model requiring access to the feature for (optimal) prediction performance (3)



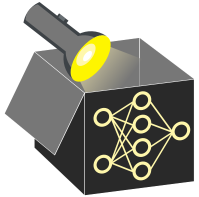


Example: correlations, confounding

- All unique prediction-relevant features for y are of interest
- Example: All features that are directly connected to y

$\Rightarrow$ `trust` and `vaccinated` may be correlated but only `vaccinated` is directly connected to y

POTENTIAL INTERPRETATION GOALS



For example, one may be interested in getting insight into whether the ...

(1) feature x_j is causal for the prediction?

- A symptom may help predict a disease ($\rightsquigarrow$ causal for $\hat{y}$)
- Intervening on symptom may not affect disease ($\rightsquigarrow$ not causal for y)

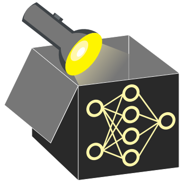
(2) feature x_j contains prediction-relevant information about y?

- x_j helps predict y (e.g., conditional expectation) w.r.t. performance
- If $x_j \perp\!\!\!\perp y$, then $\mathbb{E}[y|x_j] = \mathbb{E}[y]$ and x_j and y have zero mutual information
  $\rightsquigarrow$ x_j carries no prediction-relevant information

(3) model requires access to x_j to achieve it's prediction performance?

- x_j helps predict y w.r.t. performance, compared to using only x_{-j}
- If $x_j \perp\!\!\!\perp y|x_{-j}$, then $\mathbb{E}[y|x_{-j}] = \mathbb{E}[y|x_j, x_{-j}]$
  $\rightsquigarrow$ x_j does not contribute unique prediction-relevant information about y
- **Note:** A model may rely on features that can be replaced with others, e.g., if $\mathbb{E}[y \mid x_1] \neq \mathbb{E}[y]$ and $\mathbb{E}[y \mid x_1] = \mathbb{E}[y \mid x_1, x_2]$, a random forest may ignore x_1 in splitting and rely on x_2 instead (despite x_1 being informative).

POTENTIAL INTERPRETATION GOALS



For example, one may be interested in getting insight into whether the ...

(1) feature x_j is causal for the prediction?

- A symptom may help predict a disease ($\rightsquigarrow$ causal for $\hat{y}$)
- Intervening on symptom may not affect disease ($\rightsquigarrow$ not causal for y)

(2) feature x_j contains prediction-relevant information about y?

- x_j helps predict y (e.g., conditional expectation) w.r.t. performance
- If $x_j \perp\!\!\!\perp y$, then $\mathbb{E}[y|x_j] = \mathbb{E}[y]$ and x_j and y have 0 mutual info.
  $\rightsquigarrow$ x_j carries no prediction-relevant information

(3) model requires access to x_j to achieve it's prediction performance?

- x_j helps predict y w.r.t. performance, compared to using only x_{-j}
- If $x_j \perp\!\!\!\perp y|x_{-j}$, then $\mathbb{E}[y|x_{-j}] = \mathbb{E}[y|x_j, x_{-j}]$
  $\rightsquigarrow$ x_j does not contribute unique prediction-relevant information about y
- **Note:** A model may rely on features that can be replaced with others, e.g., if $\mathbb{E}[y \mid x_1] \neq \mathbb{E}[y]$ and $\mathbb{E}[y \mid x_1] = \mathbb{E}[y \mid x_1, x_2]$, a random forest may ignore x_1 in splitting and rely on x_2 instead (despite x_1 being informative).