Interpretable Machine Learning

Introduction to Loss-based Feature Importance

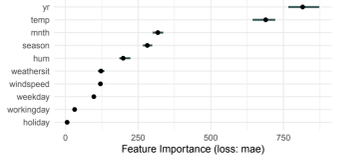


Figure: Bike Sharing Dataset

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

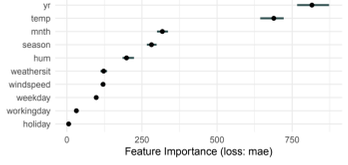


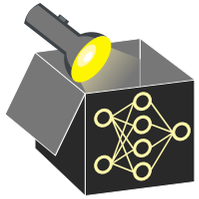
Figure: Bike Sharing Dataset

Learning goals

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

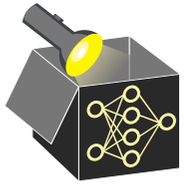
MOTIVATION

- **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

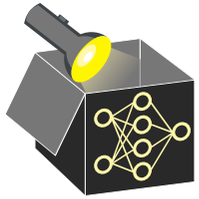


MOTIVATION

- **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

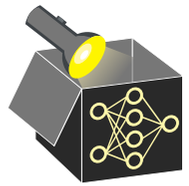


MOTIVATION



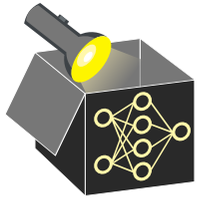
- **Feature effects** describe how a feature x_j influences the prediction $\hat{y}$
  - requires one plot per feature (e.g., PDPs, ALEs)
  - purely model-based; ignores true target y
- **Feature importance** quantifies how much each x_j contributes to prediction error
  - condenses information into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

MOTIVATION



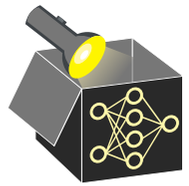
- **Feature effects** describe how a feature x_j influences the prediction $\hat{y}$
  - requires one plot per feature (e.g., PDPs, ALEs)
  - purely model-based; ignores true target y
- **Feature importance** quantifies how much each x_j contributes to prediction error
  - condenses information into 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

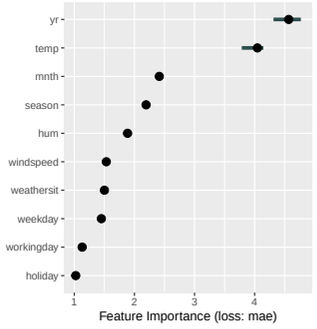


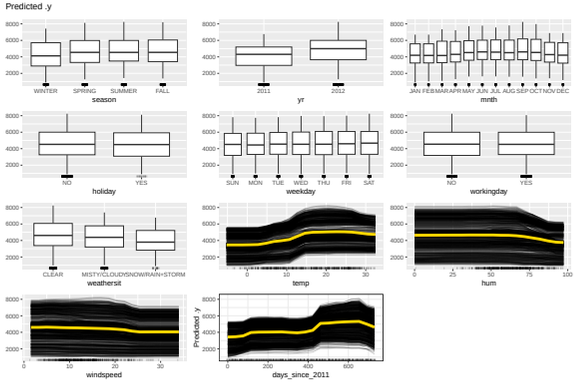
- **Feature effects** describe how a feature x_j influences the prediction $\hat{y}$
  - requires one plot per feature (e.g., PDPs, ALEs)
  - purely model-based; ignores true target y
- **Feature importance** quantifies how much each x_j contributes to prediction error
  - condenses information into 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

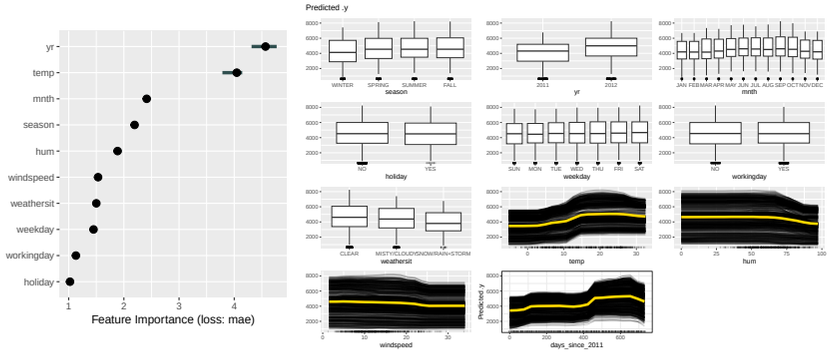




EXAMPLE

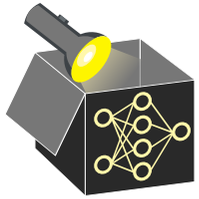
Feature importance provides a condensed summary of feature relevance w.r.t. performance

- 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



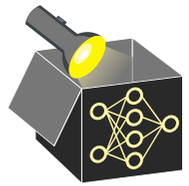
Many loss-based feature importance methods exist, which mainly differ in

(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



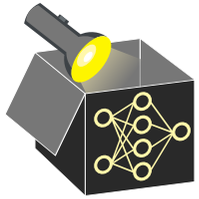
Many loss-based feature importance methods exist, which mainly differ in

(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

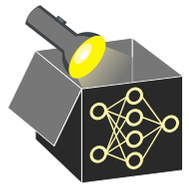
- **Remove X_j and refit:** Drop the X_j and retrain model without it
- **Perturb X_j:** Replace X_j by $\tilde{X}_j$ sampled from *marginal* or *conditional* distribution
- **Marginalize X_j:** integrate out X_j via *marginal* or *conditional* distribution

(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"
  $\rightsquigarrow$ 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)

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

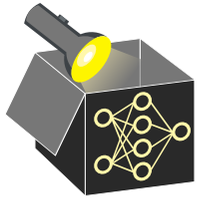
- **Remove X_j and refit:** Drop the X_j and retrain model without it
- **Perturb X_j:** Replace X_j by $\tilde{X}_j$ sampled from *marginal*/*conditional* distrib.
- **Marginalize X_j:** integrate out X_j via *marginal* or *conditional* distribution

(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)

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

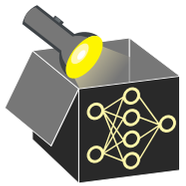
(2) How they compare model performance before and after feature removal

- **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

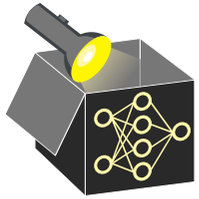
- **Remove X_j and refit:** Drop the X_j and retrain model without it
- **Perturb X_j:** Replace X_j by $\tilde{X}_j$ sampled from *marginal*/*conditional* distrib.
- **Marginalize X_j:** integrate out X_j via *marginal* or *conditional* distribution

(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).

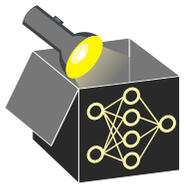
For example, one may be interested in getting insight into whether the ...

(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 ...

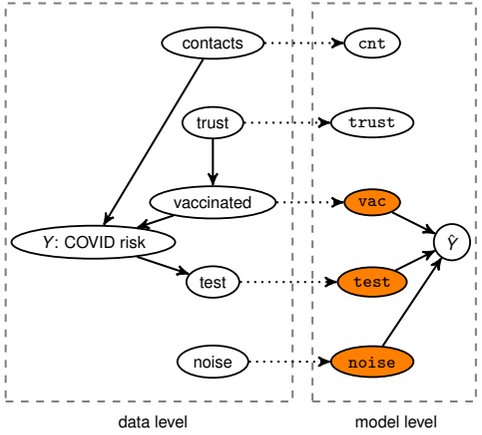
(1) feature x_j is causal for the prediction?

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

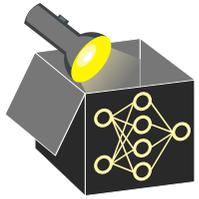
(3) model requires access to x_j to achieve it's prediction performance?

EXAMPLE: CAUSAL FOR THE PREDICTION (1)

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

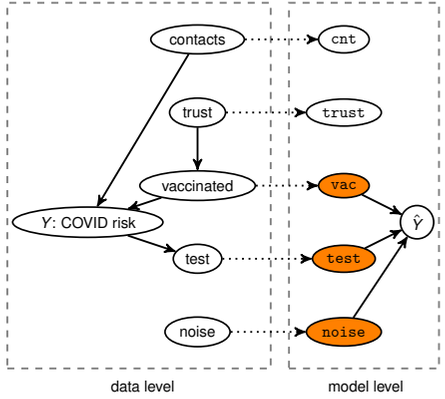


Example: overfitting due to noisy features

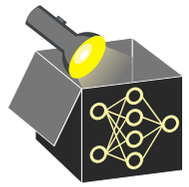


EXAMPLE: CAUSAL FOR THE PREDICTION (1)

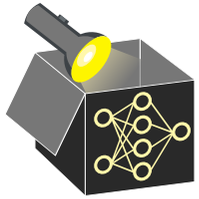
A feature may be causal for $\hat{y}$ (1) without containing prediction-relevant information about y (2)



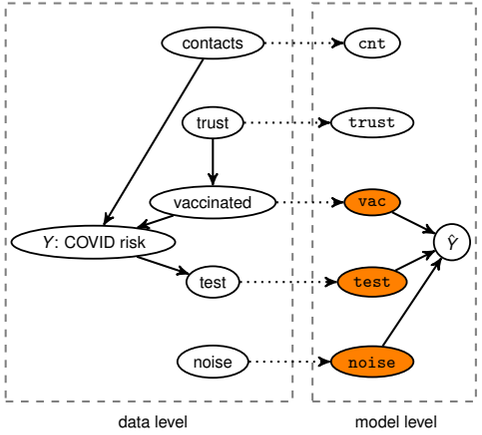
Example: overfitting due to noisy features



EXAMPLE: CAUSAL FOR THE PREDICTION (1)



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

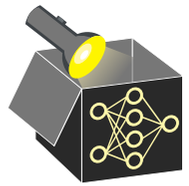


Example: overfitting due to noisy features

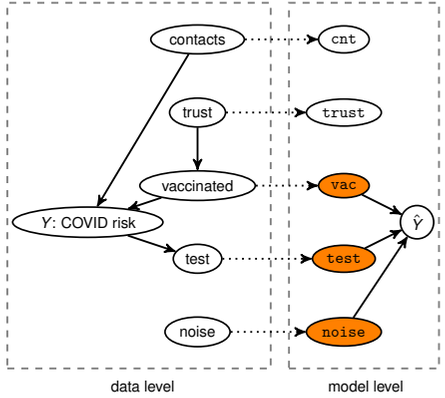
- All features used by the model are of interest
- Here: Model uses feature `noise`, although it does not contain prediction-relevant information about y (data level)

$\Rightarrow$ 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)



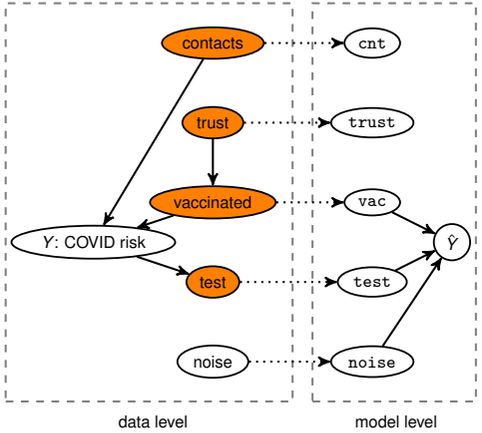
Example: overfitting due to noisy features

- 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)

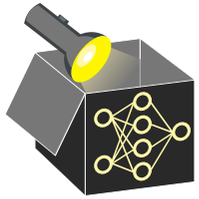
$\Rightarrow$ Overfitted models may use many noise features which are deemed relevant on model level (but not on data level)

EXAMPLE: PREDICTION-RELEVANT INFORMATION (2)

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

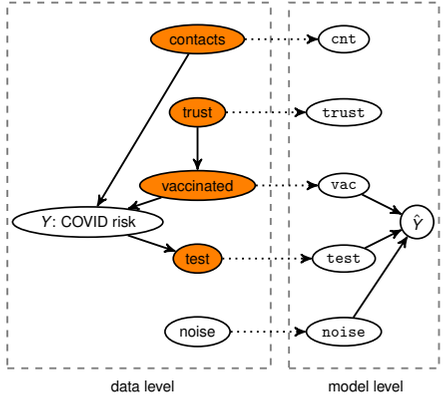


Example: underfitting, model multiplicity

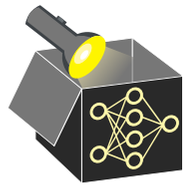


EXAMPLE: PREDICTION-RELEVANT INFORMATION (2)

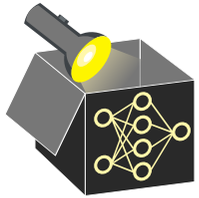
A feature may contain prediction-relevant information (2) without causing the prediction (1)



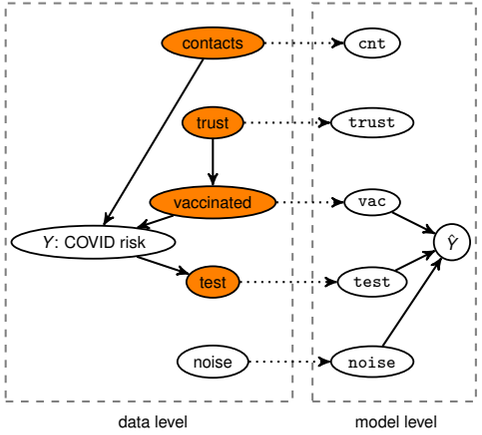
Example: underfitting, model multiplicity



EXAMPLE: PREDICTION-RELEVANT INFORMATION (2)



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

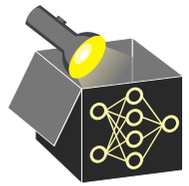


Example: underfitting, model multiplicity

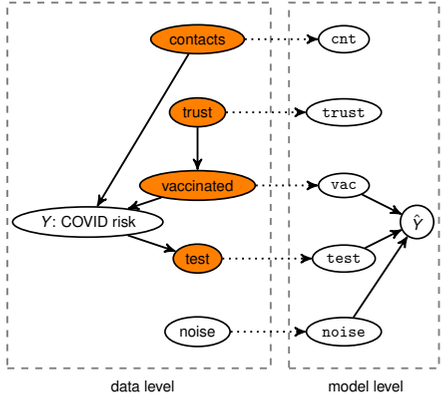
- All prediction-relevant features for y are of interest
- Example: All features that are directly or indirectly (i.e., via another feature) connected to y

$\Rightarrow$ 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)

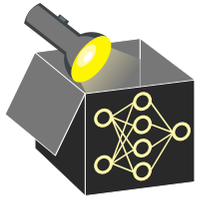


Example: underfitting, model multiplicity

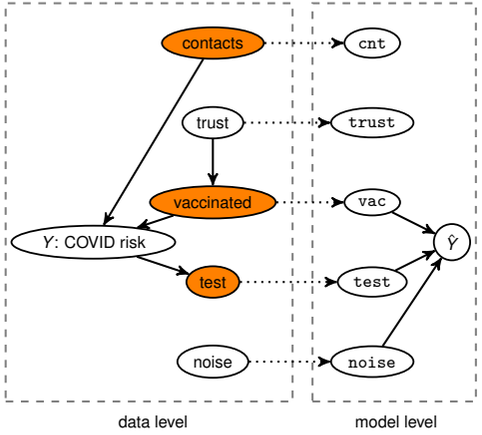
- All prediction-relevant features for y are of interest
- Example: All features that are directly or indirectly (i.e., via another feature) connected to y

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

EXAMPLE: REQUIRES ACCESS TO FEATURE (3)

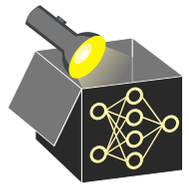


A feature may contain prediction-relevant information (2), without the model requiring access to the feature for (optimal) prediction performance (3)

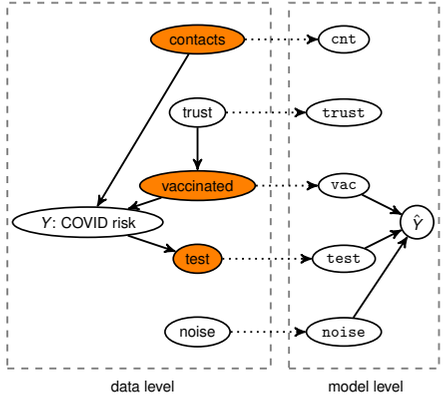


Example: correlations, confounding

EXAMPLE: REQUIRES ACCESS TO FEATURE (3)

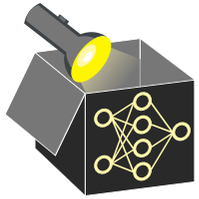


A feature may contain prediction-relevant information (2), without the model requiring access to the feature for (optimal) prediction performance (3)

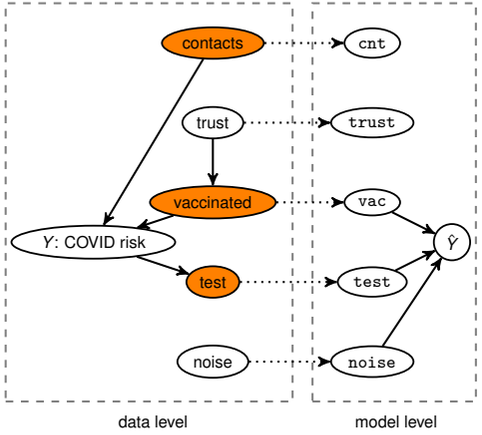


Example: correlations, confounding

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)

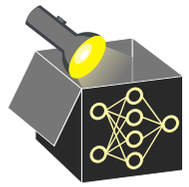


Example: correlations, confounding

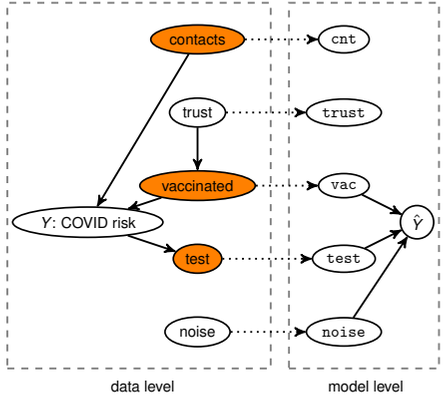
- All unique prediction-relevant features for y are of interest
- Example: All features that are directly connected to y

$\Rightarrow$ `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)

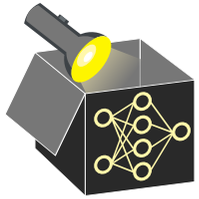


Example: correlations, confounding

- 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)

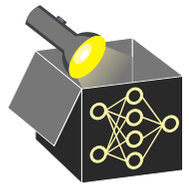
(2) feature x_j contains prediction-relevant information about y?

- 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).

Interpretable Machine Learning

Permutation Feature Importance (PFI)

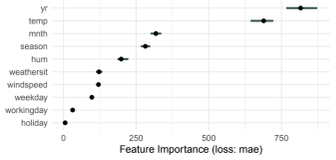


Figure: Bike Sharing Dataset

Learning goals

- Understand how PFI is computed
- Understanding strengths and weaknesses

Interpretable Machine Learning

Feature Importances 1
Permutation Feature Importance (PFI)

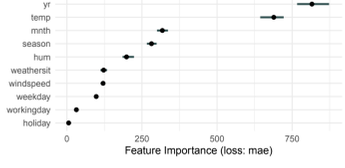


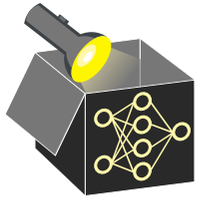
Figure: Bike Sharing Dataset

Learning goals

- Understand how PFI is computed
- Understanding strengths and weaknesses

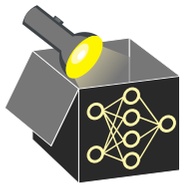
MOTIVATION FOR PFI

- **Goal:** Assess how important feature(s) X_S are for predictive performance of a **fixed trained model** $\hat{f}$ on a given dataset $\mathcal{D}$
- **Idea:** Estimate change in model performance when X_S is "made uninformative"



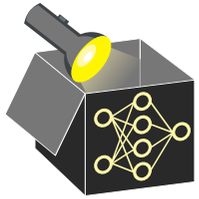
MOTIVATION FOR PFI

- **Goal:** Assess how important feature(s) X_S are for predictive performance of a **fixed trained model** $\hat{f}$ on a given dataset $\mathcal{D}$
- **Idea:** Estimate performance change when X_S is "made uninformative"



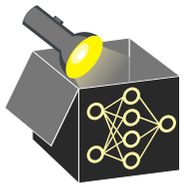
MOTIVATION FOR PFI

- **Goal:** Assess how important feature(s) X_S are for predictive performance of a **fixed trained model** $\hat{f}$ on a given dataset $\mathcal{D}$
- **Idea:** Estimate change in model performance when X_S is "made uninformative"
- **Question:** Can we make X_S uninformative by removing it from the model?
  $\rightsquigarrow$ No, $\hat{f}$ was trained with X_S and retraining without X_S gives a different model

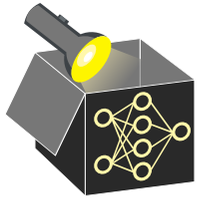


MOTIVATION FOR PFI

- **Goal:** Assess how important feature(s) X_S are for predictive performance of a **fixed trained model** $\hat{f}$ on a given dataset $\mathcal{D}$
- **Idea:** Estimate performance change when X_S is "made uninformative"
- **Question:** Can we make X_S uninformative by removing it from model?
  $\rightsquigarrow$ No, $\hat{f}$ was trained with X_S; retraining without X_S gives a different model



MOTIVATION FOR PFI

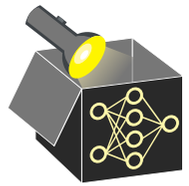


- **Goal:** Assess how important feature(s) X_S are for predictive performance of a **fixed trained model** $\hat{f}$ on a given dataset $\mathcal{D}$
- **Idea:** Estimate change in model performance when X_S is "made uninformative"
- **Question:** Can we make X_S uninformative by removing it from the model?
  $\rightsquigarrow$ No, $\hat{f}$ was trained with X_S and retraining without X_S gives a different model
- **Solution:** Simulate feature removal by replacing X_S with a perturbed version $\tilde{X}_S$ that is independent of (X_{-S}, Y) but preserves distribution $\mathbb{P}(X_S)$
  $\rightsquigarrow$ Compare baseline predictions $\hat{f}(X)$ with perturbed predictions $\hat{f}(\tilde{X}_S, X_{-S})$

$$\text{PFI}_S := \underbrace{\mathbb{E}\Big[L\big(\hat{f}(\tilde{X}_S, X_{-S}), Y\big)\Big]}_{\text{risk after "destroying" } X_S} - \underbrace{\mathbb{E}\Big[L\big(\hat{f}(X), Y\big)\Big]}_{\text{baseline risk}},$$

- **How to perturb X_S?**
  - Add random noise: distorts $\mathbb{P}(X_S)$ (not used)
  - Permutation: preserves marginal $\mathbb{P}(X_S)$, breaks dependence with Y (used)

MOTIVATION FOR PFI



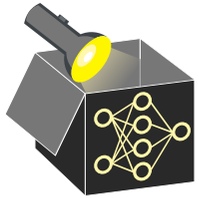
- **Goal:** Assess how important feature(s) X_S are for predictive performance of a **fixed trained model** $\hat{f}$ on a given dataset $\mathcal{D}$
- **Idea:** Estimate performance change when X_S is "made uninformative"
- **Question:** Can we make X_S uninformative by removing it from model?
  $\rightsquigarrow$ No, $\hat{f}$ was trained with X_S; retraining without X_S gives a different model
- **Solution:** Simulate feature removal by replacing X_S with a perturbed version $\tilde{X}_S$ that is independent of (X_{-S}, Y) but preserves distrib. $\mathbb{P}(X_S)$
  $\rightsquigarrow$ Compare baseline predictions $\hat{f}(X)$ with perturbed predictions $\hat{f}(\tilde{X}_S, X_{-S})$

$$\text{PFI}_S := \underbrace{\mathbb{E}\Big[L\big(\hat{f}(\tilde{X}_S, X_{-S}), Y\big)\Big]}_{\text{risk after "destroying" } X_S} - \underbrace{\mathbb{E}\Big[L\big(\hat{f}(X), Y\big)\Big]}_{\text{baseline risk}},$$

- **How to perturb X_S?**
  - Add random noise: distorts $\mathbb{P}(X_S)$ (not used)
  - Permutation: preserves marginal $\mathbb{P}(X_S)$, breaks dependence with Y (used)

PERMUTATION FEATURE IMPORTANCE (PFI)

Breiman (2001)



Sample estimator (using independent test set $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{n}$)

- Measure the error **with feat. values** x_S and **with permuted feat. values** $\tilde{x}_S$
- Repeat permutation (e.g., m times) and average difference of both errors:

$$\widehat{PFI}_S = \frac{1}{m} \sum_{k=1}^{m} \left[\mathcal{R}_{\text{emp}}(\hat{f}, \tilde{\mathcal{D}}^S_{(k)}) - \mathcal{R}_{\text{emp}}(\hat{f}, \mathcal{D}) \right]$$

- $\mathcal{D}^{(k)}_S$: dataset where column(s) x_S are **permuted** once (in repetition k)
- $\mathcal{R}_{\text{emp}}(\hat{f}, \mathcal{D}) = \frac{1}{n} \sum_{(x,y) \in \mathcal{D}} L(\hat{f}(x), y)$: Measures performance of $\hat{f}$ using $\mathcal{D}$
- Average over m permutations to reduce Monte-Carlo variance

Example of permuting feature x_S with $S = \{1\}$ and $m = 6$ permutations:

$\mathcal{D}$

$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$
1	4	7
2	5	8
3	6	9

$\Rightarrow$

$\tilde{\mathcal{D}}^S_{(1)}$

$\mathbf{x}_S$	$\mathbf{x}_2$	$\mathbf{x}_3$
1	4	7
2	5	8
3	6	9

$\tilde{\mathcal{D}}^S_{(2)}$

$\mathbf{x}_S$	$\mathbf{x}_2$	$\mathbf{x}_3$
2	4	7
1	5	8
3	6	9

$\tilde{\mathcal{D}}^S_{(3)}$

$\mathbf{x}_S$	$\mathbf{x}_2$	$\mathbf{x}_3$
2	4	7
3	5	8
1	6	9

$\tilde{\mathcal{D}}^S_{(4)}$

$\mathbf{x}_S$	$\mathbf{x}_2$	$\mathbf{x}_3$
1	4	7
3	5	8
2	6	9

$\tilde{\mathcal{D}}^S_{(5)}$

$\mathbf{x}_S$	$\mathbf{x}_2$	$\mathbf{x}_3$
3	4	7
1	5	8
2	6	9

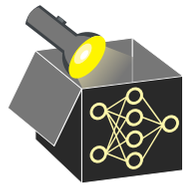
$\tilde{\mathcal{D}}^S_{(6)}$

$\mathbf{x}_S$	$\mathbf{x}_2$	$\mathbf{x}_3$
3	4	7
2	5	8
1	6	9

Note: S refers to a subset of features, here $|S| = 1$ to measure impact of permuting x_1 on performance

PERMUTATION FEATURE IMPORTANCE (PFI)

BREIMAN_2001



Sample estimator (using independent test set $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{n}$)

- Measure error **with feat. values** x_S and **with permuted feat. values** x_S
- Repeat permutation (e.g., m times) and average difference of both errors:

$$\widehat{PFI}_S = \frac{1}{m} \sum_{k=1}^{m} \left[\mathcal{R}_{\text{emp}}(\hat{f}, \mathcal{D}S_{(k)}) - \mathcal{R}_{\text{emp}}(\hat{f}, \mathcal{D}) \right]$$

- $\mathcal{D}^{(k)}_S$: dataset with column(s) x_S are **permuted** once (in repetition k)
- $\mathcal{R}_{\text{emp}}(\hat{f}, \mathcal{D}) = \frac{1}{n} \sum_{(x,y) \in \mathcal{D}} L(\hat{f}(x), y)$: Measures performance of $\hat{f}$ using $\mathcal{D}$
- Average over m permutations to reduce Monte-Carlo variance

Example of permuting feature x_S with $S = \{1\}$ and $m = 6$ permutations:

$\mathcal{D}$

$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$
1	4	7
2	5	8
3	6	9

$\Rightarrow$

$\tilde{\mathcal{D}}^S_{(1)}$

$\mathbf{x}_S$	$\mathbf{x}_2$	$\mathbf{x}_3$
1	4	7
2	5	8
3	6	9

$\tilde{\mathcal{D}}^S_{(2)}$

$\mathbf{x}_S$	$\mathbf{x}_2$	$\mathbf{x}_3$
2	4	7
1	5	8
3	6	9

$\tilde{\mathcal{D}}^S_{(3)}$

$\mathbf{x}_S$	$\mathbf{x}_2$	$\mathbf{x}_3$
2	4	7
3	5	8
1	6	9

$\tilde{\mathcal{D}}^S_{(4)}$

$\mathbf{x}_S$	$\mathbf{x}_2$	$\mathbf{x}_3$
1	4	7
3	5	8
2	6	9

$\tilde{\mathcal{D}}^S_{(5)}$

$\mathbf{x}_S$	$\mathbf{x}_2$	$\mathbf{x}_3$
3	4	7
1	5	8
2	6	9

$\tilde{\mathcal{D}}^S_{(6)}$

$\mathbf{x}_S$	$\mathbf{x}_2$	$\mathbf{x}_3$
3	4	7
2	5	8
1	6	9

Note: S refers to a subset of features, here $|S| = 1$ to measure impact of permuting x_1 on performance

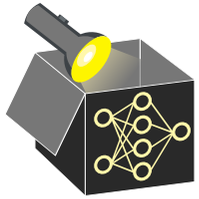
PERMUTATION FEATURE IMPORTANCE

$\tilde{\mathcal{D}}^S_{(k)}$

i	$\mathbf{x}_S$	$\mathbf{x}_2$	$\mathbf{x}_3$
1	2	4	7
⋮	1	5	8
n	3	6	9

$\mathcal{D}$

$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$
1	4	7
2	5	8
3	6	9



1. **Perturbation:** Sample feature values from the distribution of x_S ($P(X_S)$).
   $\Rightarrow$ Randomly permute feature x_S
   $\Rightarrow$ Replace x_S with permuted feature $\tilde{x}_S$ and create data $\tilde{\mathcal{D}}^S$ containing $\tilde{x}_S$

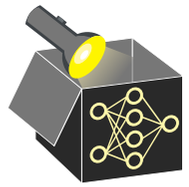
PERMUTATION FEATURE IMPORTANCE

$\mathcal{D}S_{(k)}$

i	$\mathbf{x}_S$	$\mathbf{x}_2$	$\mathbf{x}_3$
1	2	4	7
⋮	1	5	8
n	3	6	9

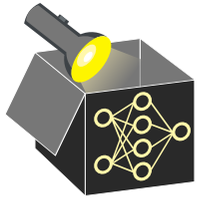
$\mathcal{D}$

$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$
1	4	7
2	5	8
3	6	9



1. **Perturbation:** Sample feature values from the distribution of x_S ($P(X_S)$).
   $\Rightarrow$ Randomly permute feature x_S
   $\Rightarrow$ Replace x_S with permuted feat. x_S and create data $\mathcal{D}S$ containing x_S

PERMUTATION FEATURE IMPORTANCE



$\tilde{\mathcal{D}}^S_{(k)}$

i	$\mathbf{x}_S$	$\mathbf{x}_2$	$\mathbf{x}_3$
1	2	4	7
⋮	1	5	8
n	3	6	9

$\mathcal{D}$

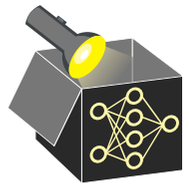
$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$
1	4	7
2	5	8
3	6	9

$\hat{f}$
0.6
0.6
0.6

$\hat{f}$
0.4
0.8
0.6

1. **Perturbation:** Sample feature values from the distribution of x_S ($P(X_S)$).
   $\Rightarrow$ Randomly permute feature x_S
   $\Rightarrow$ Replace x_S with permuted feature $\tilde{x}_S$ and create data $\tilde{\mathcal{D}}^S$ containing $\tilde{x}_S$
2. **Prediction:** Make predictions for both data, i.e., $\mathcal{D}$ and $\tilde{\mathcal{D}}^S$

PERMUTATION FEATURE IMPORTANCE



$\mathcal{DS}_{(k)}$

i	$\mathbf{x}_S$	$\mathbf{x}_2$	$\mathbf{x}_3$
1	2	4	7
⋮	1	5	8
n	3	6	9

$\mathcal{D}$

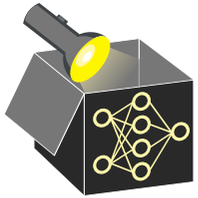
$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$
1	4	7
2	5	8
3	6	9

$\hat{f}$
0.6
0.6
0.6

$\hat{f}$
0.4
0.8
0.6

1. **Perturbation:** Sample feature values from the distribution of x_S ($P(X_S)$).
   $\Rightarrow$ Randomly permute feature x_S
   $\Rightarrow$ Replace x_S with permuted feat. x_S and create data $\mathcal{DS}$ containing x_S
2. **Prediction:** Make predictions for both data, i.e., $\mathcal{D}$ and $\mathcal{DS}$

PERMUTATION FEATURE IMPORTANCE



$\tilde{\mathcal{D}}^S_{(k)}$

i	x_S	x_2	x_3
1	2	4	7
⋮	1	5	8
n	3	6	9

$L(\hat{f}, y)$
0.9
0.5
0.1

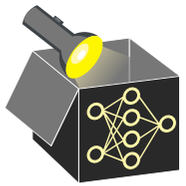
$\mathcal{D}$

x_1	x_2	x_3
1	4	7
2	5	8
3	6	9

$L(\hat{f}, y)$
0.25
0.35
0.1

3. **Aggregation:**
   - Compute the loss for each observation in both data sets

PERMUTATION FEATURE IMPORTANCE



$\mathcal{DS}_{(k)}$

i	x_S	x_2	x_3
1	2	4	7
⋮	1	5	8
n	3	6	9

$L(\hat{f}, y)$
0.9
0.5
0.1

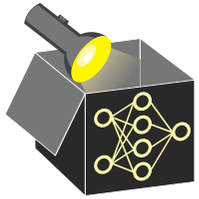
$\mathcal{D}$

x_1	x_2	x_3
1	4	7
2	5	8
3	6	9

$L(\hat{f}, y)$
0.25
0.35
0.1

3. **Aggregation:**
   - Compute the loss for each observation in both data sets

PERMUTATION FEATURE IMPORTANCE



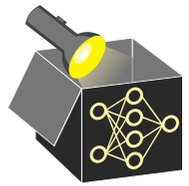
$\tilde{\mathcal{D}}^S_{(k)}$ $\mathcal{D}$

i	x_S	x_2	x_3	x_1	x_2	x_3	ΔL
1	2	4	7	1	4	7	0.65
⋮	1	5	8	2	5	8	0.15
n	3	6	9	3	6	9	0

$L(\hat{f}, y)$		$L(\hat{f}, y)$
0.9	-	0.25
0.5		0.35
0.1		0.1

3. **Aggregation:**
   - Compute the loss for each observation in both data sets
   - Take the difference of both losses ΔL for each observation

PERMUTATION FEATURE IMPORTANCE



$\mathcal{DS}_{(k)}$ $\mathcal{D}$

i	x_S	x_2	x_3	x_1	x_2	x_3	ΔL
1	2	4	7	1	4	7	0.65
⋮	1	5	8	2	5	8	0.15
n	3	6	9	3	6	9	0

$L(\hat{f}, y)$		$L(\hat{f}, y)$
0.9	-	0.25
0.5		0.35
0.1		0.1

3. **Aggregation:**
   - Compute the loss for each observation in both data sets
   - Take the difference of both losses ΔL for each observation

PERMUTATION FEATURE IMPORTANCE

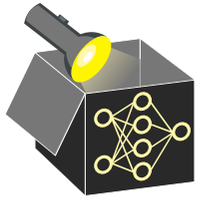
$$\mathcal{R}_{\text{emp}}(\hat{f}, \tilde{\mathcal{D}}^{S}_{(k)}) - \mathcal{R}_{\text{emp}}(\hat{f}, \mathcal{D})$$

i	$\mathbf{x}_S$	$\mathbf{x}_2$	$\mathbf{x}_3$
1	2	4	7
⋮	1	5	8
n	3	6	9

$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$
1	4	7
2	5	8
3	6	9

$\Delta\mathbf{L}$
0.65
0.15
0

= 0.267



3. **Aggregation:**
   - Compute the loss for each observation in both data sets
   - Take the difference of both losses ΔL for each observation
   - Average this change in loss across all observations
     Note: Same as computing $\mathcal{R}_{\text{emp}}$ on both data sets and taking difference

PERMUTATION FEATURE IMPORTANCE

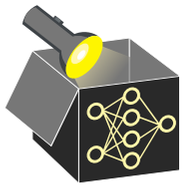
$$\mathcal{R}_{\text{emp}}(\hat{f}, \mathcal{DS}_{(k)}) - \mathcal{R}_{\text{emp}}(\hat{f}, \mathcal{D})$$

i	$\mathbf{x}_S$	$\mathbf{x}_2$	$\mathbf{x}_3$
1	2	4	7
⋮	1	5	8
n	3	6	9

$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$
1	4	7
2	5	8
3	6	9

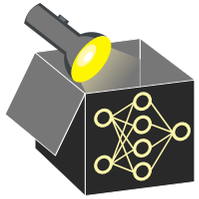
$\Delta\mathbf{L}$
0.65
0.15
0

= 0.267



3. **Aggregation:**
   - Compute the loss for each observation in both data sets
   - Take the difference of both losses ΔL for each observation
   - Average this change in loss across all observations
     Note: Same as computing $\mathcal{R}_{\text{emp}}$ on both data sets and taking difference

PERMUTATION FEATURE IMPORTANCE



$$\mathcal{R}_{\text{emp}}(\hat{f}, \tilde{\mathcal{D}}^{S}_{(k)}) - \mathcal{R}_{\text{emp}}(\hat{f}, \mathcal{D})$$

	i	$\mathbf{x}_S$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\Delta\mathbf{L}$
1	1	2	4	7	1	4	7	0.65
	⋮	1	5	8	2	5	8	0.15
	n	3	6	9	3	6	9	0
⋮								
m	1	3	4	7	1	4	7	0.85
	⋮	2	5	8	2	5	8	0
	n	1	6	9	3	6	9	0.35

= 0.267

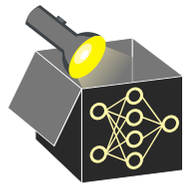
$\widehat{PFI}_S$ = ½ (0.267 + 0.4)

= 0.4

3. Aggregation:

- Compute the loss for each observation in both data sets
- Take the difference of both losses ΔL for each observation
- Average this change in loss across all observations
- Repeat perturbation and average over multiple repetitions

PERMUTATION FEATURE IMPORTANCE



$$\mathcal{R}_{\text{emp}}(\hat{f}, \mathcal{DS}_{(k)}) - \mathcal{R}_{\text{emp}}(\hat{f}, \mathcal{D})$$

	i	$\mathbf{x}_S$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\Delta\mathbf{L}$
1	1	2	4	7	1	4	7	0.65
	⋮	1	5	8	2	5	8	0.15
	n	3	6	9	3	6	9	0
⋮								
m	1	3	4	7	1	4	7	0.85
	⋮	2	5	8	2	5	8	0
	n	1	6	9	3	6	9	0.35

= 0.267

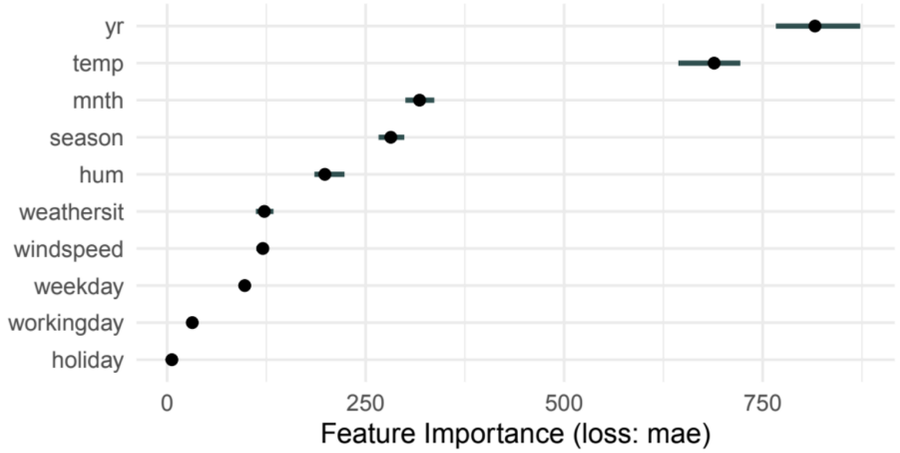
$\widehat{PFI}_S$ = ½ (0.267 + 0.4)

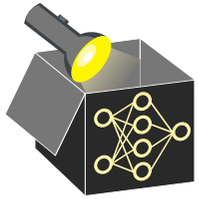
= 0.4

3. Aggregation:

- Compute the loss for each observation in both data sets
- Take the difference of both losses ΔL for each observation
- Average this change in loss across all observations
- Repeat perturbation and average over multiple repetitions

EXAMPLE: BIKE SHARING DATASET

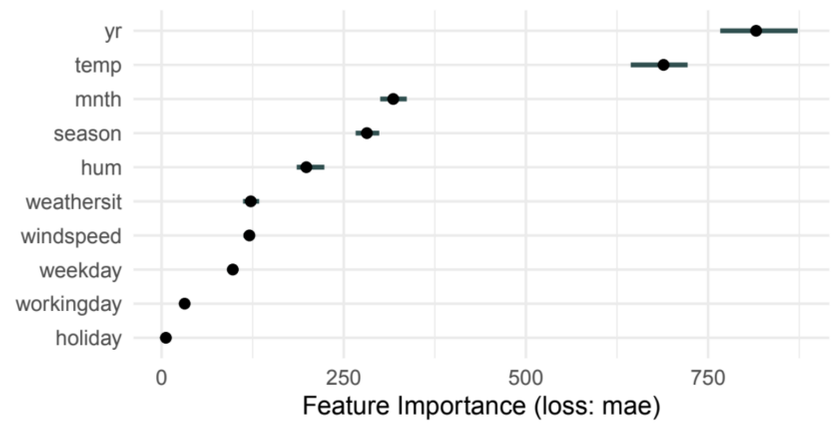


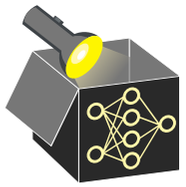


Interpretation:

- `yr` and `temp` are most important features using mean absolute error (MAE)
- Destroying information about `yr` by permuting it increases MAE of model by 816
- Error bars show 5% and 95% quantiles over multiple permutations

EXAMPLE: BIKE SHARING DATASET



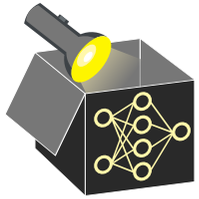


Interpretation:

- `yr` and `temp` are most important feats using mean absolute error (MAE)
- Destroying info. about `yr` by permuting it increases MAE of model by 816
- Error bars show 5% and 95% quantiles over multiple permutations

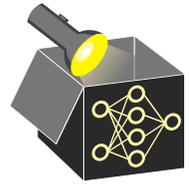
COMMENTS ON PFI

- Interpretation: Increase in error when feature's information is destroyed



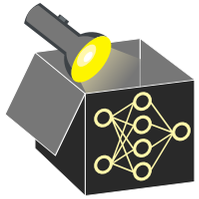
COMMENTS ON PFI

- Interpretation: Increase in error when feature's information is destroyed



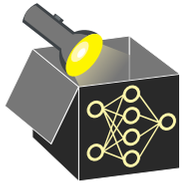
COMMENTS ON PFI

- Interpretation: Increase in error when feature's information is destroyed
- Results can be unreliable due to random permutations
  $\Rightarrow$ Solution: Average results over multiple repetitions

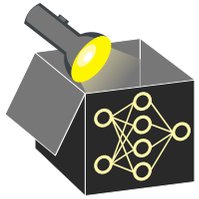


COMMENTS ON PFI

- Interpretation: Increase in error when feature's information is destroyed
- Results can be unreliable due to random permutations
  $\Rightarrow$ Solution: Average results over multiple repetitions

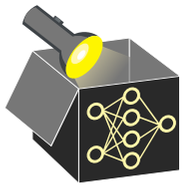


COMMENTS ON PFI



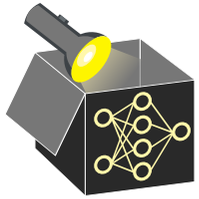
- Interpretation: Increase in error when feature's information is destroyed
- Results can be unreliable due to random permutations
  $\Rightarrow$ Solution: Average results over multiple repetitions
- Permuting features despite correlation/dependence with other features can lead to unrealistic combinations of feature values $\rightsquigarrow$ Extrapolation issue

COMMENTS ON PFI



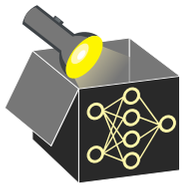
- Interpretation: Increase in error when feature's information is destroyed
- Results can be unreliable due to random permutations
  $\Rightarrow$ Solution: Average results over multiple repetitions
- Permuting features despite correlation/dependence with other features can lead to unrealistic combinations of feature values
  $\rightsquigarrow$ Extrapolation issue

COMMENTS ON PFI



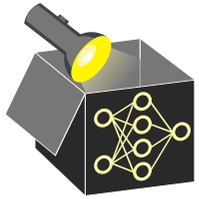
- Interpretation: Increase in error when feature's information is destroyed
- Results can be unreliable due to random permutations
  $\Rightarrow$ Solution: Average results over multiple repetitions
- Permuting features despite correlation/dependence with other features can lead to unrealistic combinations of feature values $\rightsquigarrow$ Extrapolation issue
- PFI automatically includes importance of interaction effects with other features
  $\Rightarrow$ Permuting x_j also destroys interactions with permuted feature
  $\Rightarrow$ PFI score contains importance of all interactions with permuted feature

COMMENTS ON PFI



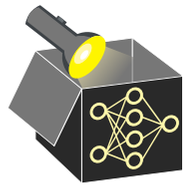
- Interpretation: Increase in error when feature's information is destroyed
- Results can be unreliable due to random permutations
  $\Rightarrow$ Solution: Average results over multiple repetitions
- Permuting features despite correlation/dependence with other features can lead to unrealistic combinations of feature values
  $\rightsquigarrow$ Extrapolation issue
- PFI automatically includes importance of interaction effects with other features
  $\Rightarrow$ Permuting x_j also destroys interactions with permuted feature
  $\Rightarrow$ PFI score contains importance of all interactions with permuted feature

COMMENTS ON PFI



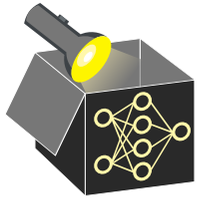
- Interpretation: Increase in error when feature's information is destroyed
- Results can be unreliable due to random permutations
  $\Rightarrow$ Solution: Average results over multiple repetitions
- Permuting features despite correlation/dependence with other features can lead to unrealistic combinations of feature values $\rightsquigarrow$ Extrapolation issue
- PFI automatically includes importance of interaction effects with other features
  $\Rightarrow$ Permuting x_j also destroys interactions with permuted feature
  $\Rightarrow$ PFI score contains importance of all interactions with permuted feature
- Interpretation of PFI depends on whether training or test data is used

COMMENTS ON PFI



- Interpretation: Increase in error when feature's information is destroyed
- Results can be unreliable due to random permutations
  $\Rightarrow$ Solution: Average results over multiple repetitions
- Permuting features despite correlation/dependence with other features can lead to unrealistic combinations of feature values
  $\rightsquigarrow$ Extrapolation issue
- PFI automatically includes importance of interaction effects with other features
  $\Rightarrow$ Permuting x_j also destroys interactions with permuted feature
  $\Rightarrow$ PFI score contains importance of all interactions with permuted feature
- Interpretation of PFI depends on whether training or test data is used

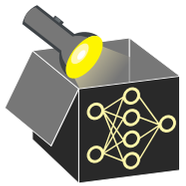
COMMENTS ON PFI - EXTRAPOLATION



Example: Let $y = x_3 + \epsilon_y$, with $\epsilon_y \sim \mathcal{N}(0, 0.1)$.

- $x_1 := \epsilon_1$, $x_2 := x_1 + \epsilon_2$ are highly correlated ($\epsilon_1 \sim \mathcal{N}(0, 1)$, $\epsilon_2 \sim \mathcal{N}(0, 0.01)$)
- $x_3 := \epsilon_3$, $x_4 := \epsilon_4$, with $\epsilon_3, \epsilon_4 \sim \mathcal{N}(0, 1)$ and all noise terms ϵ_j are independent
- Fitting a linear model yields $\hat{f}(\mathbf{x}) \approx 0.3x_1 - 0.3x_2 + x_3$

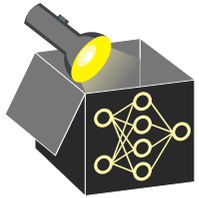
COMMENTS ON PFI - EXTRAPOLATION



Example: Let $y = x_3 + \epsilon_y$, with $\epsilon_y \sim \mathcal{N}(0, 0.1)$.

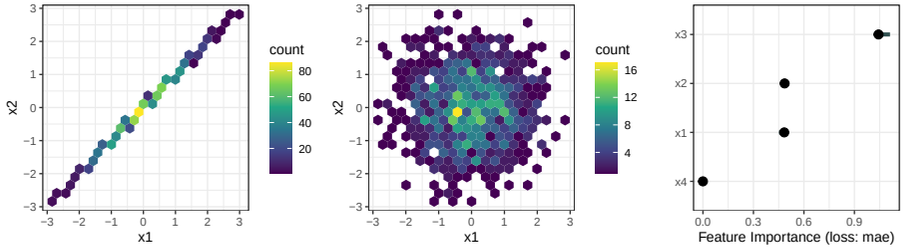
- $x_1 := \epsilon_1$, $x_2 := x_1 + \epsilon_2$; highly correlated ($\epsilon_1 \sim \mathcal{N}(0, 1)$, $\epsilon_2 \sim \mathcal{N}(0, 0.01)$)
- $x_3 := \epsilon_3$, $x_4 := \epsilon_4$, with $\epsilon_3, \epsilon_4 \sim \mathcal{N}(0, 1)$; all noise terms ϵ_j are indep.
- Fitting a linear model yields $\hat{f}(\mathbf{x}) \approx 0.3x_1 - 0.3x_2 + x_3$

COMMENTS ON PFI - EXTRAPOLATION



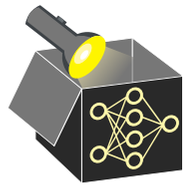
Example: Let $y = x_3 + \epsilon_y$, with $\epsilon_y \sim \mathcal{N}(0, 0.1)$.

- $x_1 := \epsilon_1$, $x_2 := x_1 + \epsilon_2$ are highly correlated ($\epsilon_1 \sim \mathcal{N}(0, 1)$, $\epsilon_2 \sim \mathcal{N}(0, 0.01)$)
- $x_3 := \epsilon_3$, $x_4 := \epsilon_4$, with $\epsilon_3, \epsilon_4 \sim \mathcal{N}(0, 1)$ and all noise terms ϵ_j are independent
- Fitting a linear model yields $\hat{f}(\mathbf{x}) \approx 0.3x_1 - 0.3x_2 + x_3$



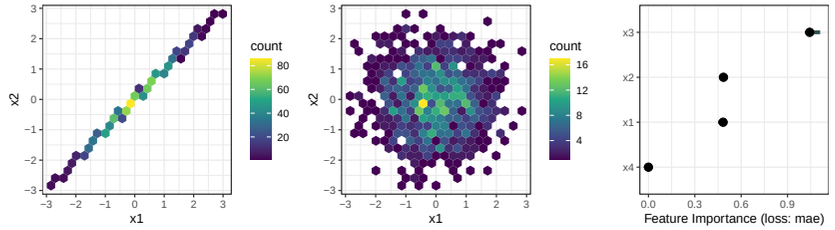
Hexbin plot of (x_1, x_2) before (left) and after (center) permuting x_1; PFI scores (right).

COMMENTS ON PFI - EXTRAPOLATION



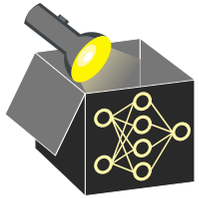
Example: Let $y = x_3 + \epsilon_y$, with $\epsilon_y \sim \mathcal{N}(0, 0.1)$.

- $x_1 := \epsilon_1$, $x_2 := x_1 + \epsilon_2$; highly correlated ($\epsilon_1 \sim \mathcal{N}(0, 1)$, $\epsilon_2 \sim \mathcal{N}(0, 0.01)$)
- $x_3 := \epsilon_3$, $x_4 := \epsilon_4$, with $\epsilon_3, \epsilon_4 \sim \mathcal{N}(0, 1)$; all noise terms ϵ_j are indep.
- Fitting a linear model yields $\hat{f}(\mathbf{x}) \approx 0.3x_1 - 0.3x_2 + x_3$



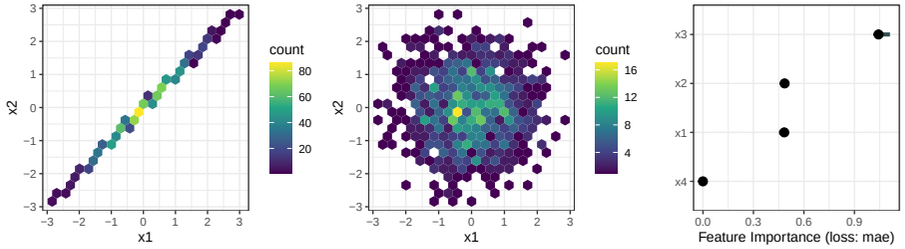
Hexbin plot of (x_1, x_2) before (left) and after (center) permuting x_1;
PFI scores (right).

COMMENTS ON PFI - EXTRAPOLATION



Example: Let $y = x_3 + \epsilon_y$, with $\epsilon_y \sim \mathcal{N}(0, 0.1)$.

- $x_1 := \epsilon_1$, $x_2 := x_1 + \epsilon_2$ are highly correlated ($\epsilon_1 \sim \mathcal{N}(0,1)$, $\epsilon_2 \sim \mathcal{N}(0, 0.01)$)
- $x_3 := \epsilon_3$, $x_4 := \epsilon_4$, with $\epsilon_3, \epsilon_4 \sim \mathcal{N}(0,1)$ and all noise terms ϵ_j are independent
- Fitting a linear model yields $\hat{f}(\mathbf{x}) \approx 0.3x_1 - 0.3x_2 + x_3$



Hexbin plot of (x_1, x_2) before (left) and after (center) permuting x_1; PFI scores (right).

$\Rightarrow$ x_1, x_2 cancel in $\hat{f}$ since $x_1 \approx x_2$, hence $0.3x_1 - 0.3x_2 \approx 0$ $\rightsquigarrow$ should be irrelevant

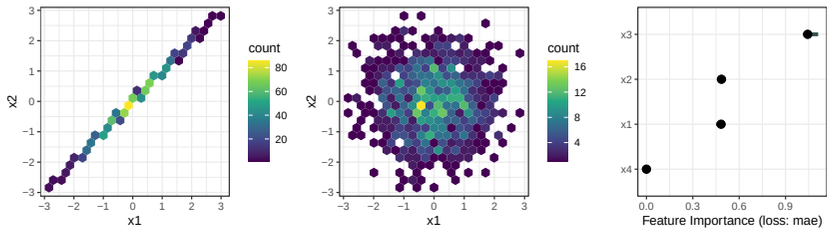
$\Rightarrow$ Permuting x_1 breaks joint structure $\rightsquigarrow$ unrealistic inputs

$\Rightarrow$ $PFI > 0$ due to extrapolation (PFI evaluates model on unrealistic inputs)
$\rightsquigarrow$ x_1, x_2 are misleadingly considered relevant

COMMENTS ON PFI - EXTRAPOLATION

Example: Let $y = x_3 + \epsilon_y$, with $\epsilon_y \sim \mathcal{N}(0, 0.1)$.

- $x_1 := \epsilon_1$, $x_2 := x_1 + \epsilon_2$; highly correlated ($\epsilon_1 \sim \mathcal{N}(0,1)$, $\epsilon_2 \sim \mathcal{N}(0, 0.01)$)
- $x_3 := \epsilon_3$, $x_4 := \epsilon_4$, with $\epsilon_3, \epsilon_4 \sim \mathcal{N}(0,1)$; all noise terms ϵ_j are indep.
- Fitting a linear model yields $\hat{f}(\mathbf{x}) \approx 0.3x_1 - 0.3x_2 + x_3$

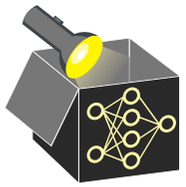


Hexbin plot of (x_1, x_2) before (left) and after (center) permuting x_1; PFI scores (right).

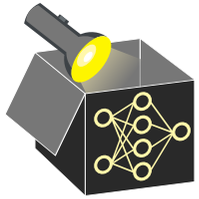
$\Rightarrow$ x_1, x_2 cancel in $\hat{f}$ since $x_1 \approx x_2$, hence $0.3x_1 - 0.3x_2 \approx 0$ $\rightsquigarrow$ should be irrelevant

$\Rightarrow$ Permuting x_1 breaks joint structure $\rightsquigarrow$ unrealistic inputs

$\Rightarrow$ $PFI > 0$ due to extrapolation (PFI evaluates model on unrealistic inputs)
$\rightsquigarrow$ x_1, x_2 are misleadingly considered relevant



COMMENTS ON PFI - INTERACTIONS

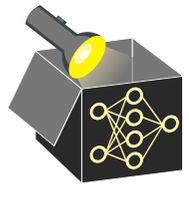


Example: Let $x_1, \ldots, x_4$ be independently and uniformly sampled from $\{-1, 1\}$ and

$$y := x_1 x_2 + x_3 + \epsilon_Y \text{ with } \epsilon_Y \sim N(0, 1)$$

Fitting a LM yields $\hat{f}(x) \approx x_1 x_2 + x_3$.

COMMENTS ON PFI - INTERACTIONS

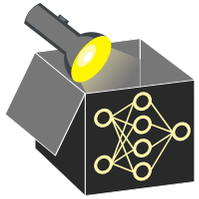


Example: Let $x_1, \ldots, x_4$ be independently and uniformly sampled from $\{-1, 1\}$ and

$$y := x_1 x_2 + x_3 + \epsilon_Y \text{ with } \epsilon_Y \sim N(0, 1)$$

Fitting a LM yields $\hat{f}(x) \approx x_1 x_2 + x_3$.

COMMENTS ON PFI - INTERACTIONS



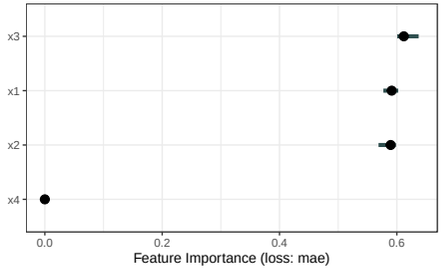
Example: Let $x_1, \ldots, x_4$ be independently and uniformly sampled from $\{-1, 1\}$ and

$$y := x_1 x_2 + x_3 + \epsilon_Y \text{ with } \epsilon_Y \sim N(0, 1)$$

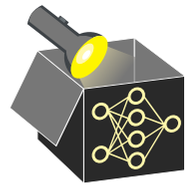
Fitting a LM yields $\hat{f}(x) \approx x_1 x_2 + x_3$.

Although x_3 alone contributes as much to the prediction as x_1 and x_2 jointly, all three are considered equally relevant.

$\Rightarrow$ PFI does not fairly attribute the performance to the individual features.



COMMENTS ON PFI - INTERACTIONS



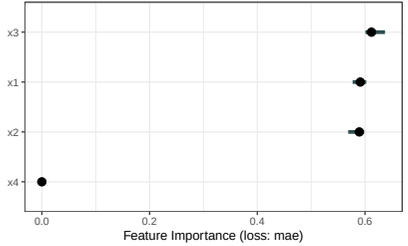
Example: Let $x_1, \ldots, x_4$ be independently and uniformly sampled from $\{-1, 1\}$ and

$$y := x_1 x_2 + x_3 + \epsilon_Y \text{ with } \epsilon_Y \sim N(0, 1)$$

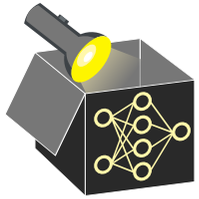
Fitting a LM yields $\hat{f}(x) \approx x_1 x_2 + x_3$.

Although x_3 alone contributes as much to the prediction as x_1 and x_2 jointly, all three are considered equally relevant.

$\Rightarrow$ PFI does not fairly attribute the performance to the individual features.

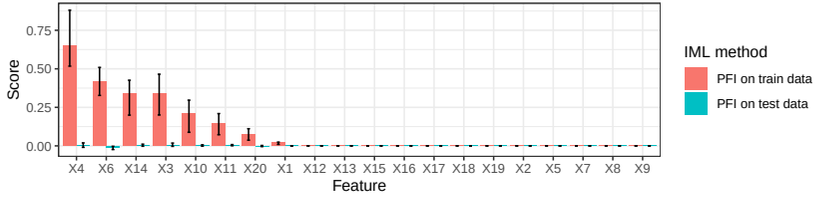


COMMENTS ON PFI - TRAIN VS. TEST DATA



Example:

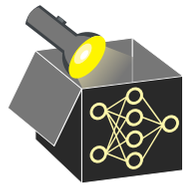
- $x_1, \ldots, x_{20}, y$ are independently sampled from $\mathcal{U}(-10, 10)$
- Train set: $n = 50$ (intentionally small) and large test set
- Model: `xgboost` with default settings (overfits strongly)



Observation:

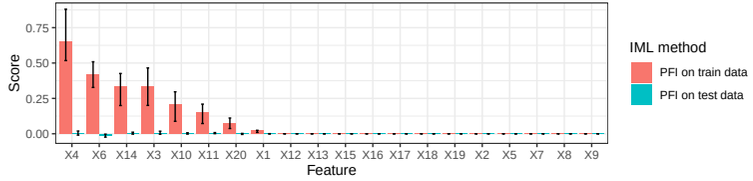
- PFI on train data highlights features that the model overfitted to.
- PFI on test data detects no relevant features.

COMMENTS ON PFI - TRAIN VS. TEST DATA



Example:

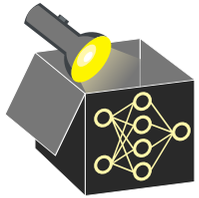
- $x_1, \ldots, x_{20}, y$ are independently sampled from $\mathcal{U}(-10, 10)$
- Train set: $n = 50$ (intentionally small) and large test set
- Model: `xgboost` with default settings (overfits strongly)



Observation:

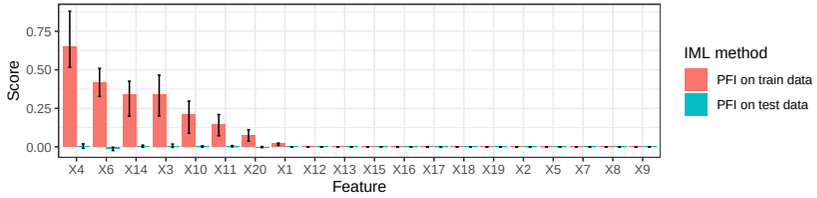
- PFI on train data highlights features that the model overfitted to.
- PFI on test data detects no relevant features.

COMMENTS ON PFI - TRAIN VS. TEST DATA



Example:

- $x_1, \ldots, x_{20}, y$ are independently sampled from $\mathcal{U}(-10, 10)$
- Train set: $n = 50$ (intentionally small) and large test set
- Model: `xgboost` with default settings (overfits strongly)



Observation:

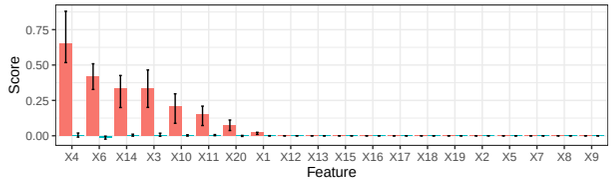
- PFI on train data highlights features that the model overfitted to.
- PFI on test data detects no relevant features.

Why? *PFI* $\neq 0$ if permuting a feature breaks a dependency the model relies on. Model overfits due to spurious feature-target dependencies in train that vanish on test.
$\Rightarrow$ To identify features that help the model to generalize, compute PFI on test data.

COMMENTS ON PFI - TRAIN VS. TEST DATA

Example:

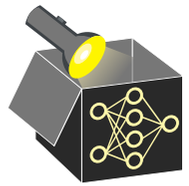
- $x_1, \ldots, x_{20}, y$ are independently sampled from $\mathcal{U}(-10, 10)$
- Train set: $n = 50$ (intentionally small) and large test set
- Model: `xgboost` with default settings (overfits strongly)



Observation:

- PFI on train data highlights features that the model overfitted to.
- PFI on test data detects no relevant features.

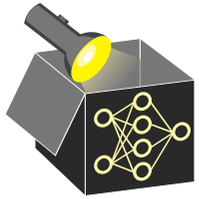
Why? *PFI* $\neq 0$ if permuting a feature breaks a dependency the model relies on. Model overfits due to spurious feature-target dependencies in train that vanish on test.
$\Rightarrow$ To find features that help the model to generalize, compute PFI on test data.



IMPLICATIONS OF PFI

Can we get insight into whether the ...

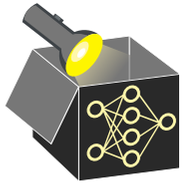
1. feature x_j is causal for the prediction?
   - $PFI_j \neq 0 \Rightarrow$ model relies on x_j
   - As the train vs. test data example shows, the converse does not hold



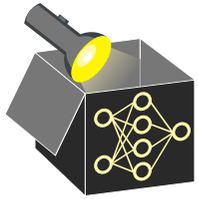
IMPLICATIONS OF PFI

Can we get insight into whether the ...

1. feature x_j is causal for the prediction?
   - $PFI_j \neq 0 \Rightarrow$ model relies on x_j
   - As the train vs. test data example shows, the converse does not hold



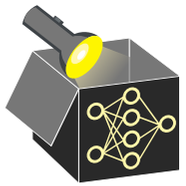
IMPLICATIONS OF PFI



Can we get insight into whether the ...

1. feature x_j is causal for the prediction?
   - $PFI_j \neq 0 \Rightarrow$ model relies on x_j
   - As the train vs. test data example shows, the converse does not hold
2. feature x_j contains prediction-relevant information?
   - $PFI_j \neq 0 \Rightarrow x_j$ is dependent on y, x_{-j}, or both (due to extrapolation)
   - x_j is not exploited by model (regardless of whether it is useful for y or not) $\Rightarrow PFI_j = 0$

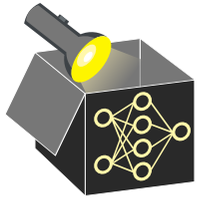
IMPLICATIONS OF PFI



Can we get insight into whether the ...

1. feature x_j is causal for the prediction?
   - $PFI_j \neq 0 \Rightarrow$ model relies on x_j
   - As the train vs. test data example shows, the converse does not hold
2. feature x_j contains prediction-relevant information?
   - $PFI_j \neq 0 \Rightarrow x_j$ is dependent on y, x_{-j}, or both (due to extrapolation)
   - x_j is not exploited by model (regardless of its usefulness for y) $\Rightarrow PFI_j = 0$

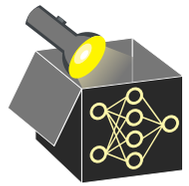
IMPLICATIONS OF PFI



Can we get insight into whether the ...

1. feature x_j is causal for the prediction?
   - $PFI_j \neq 0 \Rightarrow$ model relies on x_j
   - As the train vs. test data example shows, the converse does not hold
2. feature x_j contains prediction-relevant information?
   - $PFI_j \neq 0 \Rightarrow x_j$ is dependent on y, x_{-j}, or both (due to extrapolation)
   - x_j is not exploited by model (regardless of whether it is useful for y or not) $\Rightarrow PFI_j = 0$
3. model requires access to x_j to achieve it's prediction performance?
   - As the extrapolation example demonstrates, such insight is not possible

IMPLICATIONS OF PFI

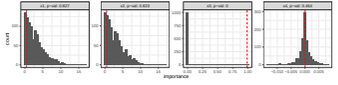


Can we get insight into whether the ...

1. feature x_j is causal for the prediction?
   - $PFI_j \neq 0 \Rightarrow$ model relies on x_j
   - As the train vs. test data example shows, the converse does not hold
2. feature x_j contains prediction-relevant information?
   - $PFI_j \neq 0 \Rightarrow x_j$ is dependent on y, x_{-j}, or both (due to extrapolation)
   - x_j is not exploited by model (regardless of its usefulness for y) $\Rightarrow PFI_j = 0$
3. model requires access to x_j to achieve it's prediction performance?
   - As shown by the extrapolation example, such insight is not possible

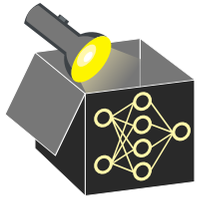
Interpretable Machine Learning

Permutation IMPortance (PIMP)



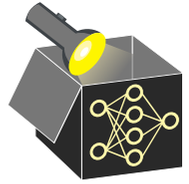
Learning goals

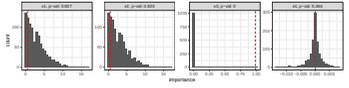
- Understand PIMP and its motivation
- Address multiple testing in feature importance



Interpretable Machine Learning

Feature Importances 1
Permutation IMPortance (PIMP)



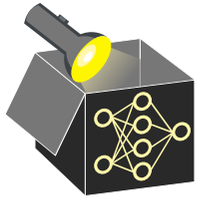


Learning goals

- Understand PIMP and its motivation
- Address multiple testing in feature importance

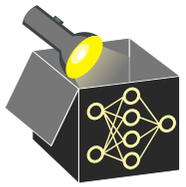
TESTING IMPORTANCE (PIMP) ▸ Altmann et al. (2010)

- PIMP was originally introduced for random forest's built-in PFI scores

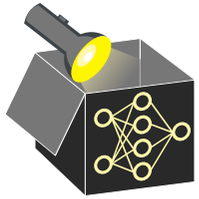


TESTING IMPORTANCE (PIMP) ▸ ALTMANN_2010

- PIMP was originally introduced for random forest's built-in PFI scores

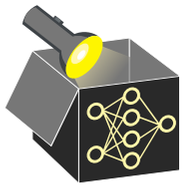


TESTING IMPORTANCE (PIMP) ▸ Altmann et al. (2010)



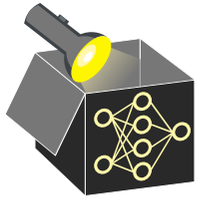
- PIMP was originally introduced for random forest's built-in PFI scores
- PIMP idea: Test if an observed $\widehat{\text{PFI}}_j^{\text{obs}}$ score is *significantly* greater than expected under the null hypothesis of X_j being not important
  $\rightsquigarrow$ Accounts for spurious importance due to randomness

TESTING IMPORTANCE (PIMP) ▸ ALTMANN_2010



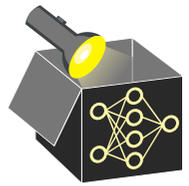
- PIMP was originally introduced for random forest's built-in PFI scores
- PIMP idea: Test if an observed $\widehat{\text{PFI}}_j^{\text{obs}}$ score is *significantly* greater than expected under the null hypothesis of X_j being not important
  $\rightsquigarrow$ Accounts for spurious importance due to randomness

TESTING IMPORTANCE (PIMP) ▸ Altmann et al. (2010)



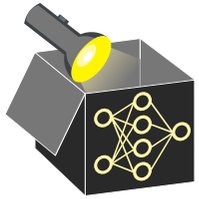
- PIMP was originally introduced for random forest's built-in PFI scores
- PIMP idea: Test if an observed $\widehat{\text{PFI}}_j^{\text{obs}}$ score is *significantly* greater than expected under the null hypothesis of X_j being not important
  $\rightsquigarrow$ Accounts for spurious importance due to randomness
- Null hypothesis H_0: Feature X_j is conditionally independent of y (unimportant)

TESTING IMPORTANCE (PIMP) ▸ ALTMANN_2010



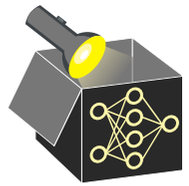
- PIMP was originally introduced for random forest's built-in PFI scores
- PIMP idea: Test if an observed $\widehat{\text{PFI}}_j^{\text{obs}}$ score is *significantly* greater than expected under the null hypothesis of X_j being not important
  $\rightsquigarrow$ Accounts for spurious importance due to randomness
- Null hypothesis H_0: Feature X_j is conditionally indep. of y (unimportant)

TESTING IMPORTANCE (PIMP) ▸ Altmann et al. (2010)



- PIMP was originally introduced for random forest's built-in PFI scores
- PIMP idea: Test if an observed $\widehat{\text{PFI}}_j^{\text{obs}}$ score is *significantly* greater than expected under the null hypothesis of X_j being not important
  $\rightsquigarrow$ Accounts for spurious importance due to randomness
- Null hypothesis H_0: Feature X_j is conditionally independent of y (unimportant)
- Approximate null distribution of PFI scores under H_0 by repeated permutations: Permute $y \to$ retrain model $\to$ recompute $\widehat{\text{PFI}}_j$ scores for all $j \to$ repeat B times
  $\Rightarrow$ Permuting y breaks relationship to all features (PFI scores reflect noise only)

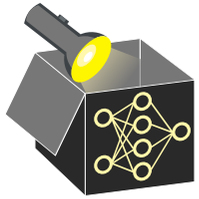
TESTING IMPORTANCE (PIMP) ▸ ALTMANN_2010



- PIMP was originally introduced for random forest's built-in PFI scores
- PIMP idea: Test if an observed $\widehat{\text{PFI}}_j^{\text{obs}}$ score is *significantly* greater than expected under the null hypothesis of X_j being not important
  $\rightsquigarrow$ Accounts for spurious importance due to randomness
- Null hypothesis H_0: Feature X_j is conditionally indep. of y (unimportant)
- Approximate null distrib. of PFI scores under H_0 by repeated permuts: Permute $y \to$ retrain $\to$ recompute $\widehat{\text{PFI}}_j$ scores for all $j \to$ repeat B times
  $\Rightarrow$ Permuting y breaks relationship to all features (PFI scores reflect noise only)

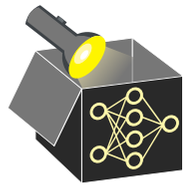
Interpretable Machine Learning – 1 / 4

TESTING IMPORTANCE (PIMP) ▸ Altmann et al. (2010)



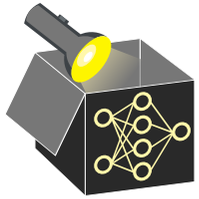
- PIMP was originally introduced for random forest's built-in PFI scores
- PIMP idea: Test if an observed $\widehat{\text{PFI}}_j^{\text{obs}}$ score is *significantly* greater than expected under the null hypothesis of X_j being not important
  $\rightsquigarrow$ Accounts for spurious importance due to randomness
- Null hypothesis H_0: Feature X_j is conditionally independent of y (unimportant)
- Approximate null distribution of PFI scores under H_0 by repeated permutations:
  Permute $y \rightarrow$ retrain model $\rightarrow$ recompute $\widehat{\text{PFI}}_j$ scores for all $j \rightarrow$ repeat B times
  $\Rightarrow$ Permuting y breaks relationship to all features (PFI scores reflect noise only)
- Assess the significance of PFI scores via tail probability under H_0
  $\Rightarrow$ Use this as a new feature importance score, adjusting for random chance

TESTING IMPORTANCE (PIMP) ▸ ALTMANN_2010



- PIMP was originally introduced for random forest's built-in PFI scores
- PIMP idea: Test if an observed $\widehat{\text{PFI}}_j^{\text{obs}}$ score is *significantly* greater than expected under the null hypothesis of X_j being not important
  $\rightsquigarrow$ Accounts for spurious importance due to randomness
- Null hypothesis H_0: Feature X_j is conditionally indep. of y (unimportant)
- Approximate null distrib. of PFI scores under H_0 by repeated permuts:
  Permute $y \rightarrow$ retrain $\rightarrow$ recompute $\widehat{\text{PFI}}_j$ scores for all $j \rightarrow$ repeat B times
  $\Rightarrow$ Permuting y breaks relationship to all features (PFI scores reflect noise only)
- Assess the significance of PFI scores via tail probability under H_0
  $\Rightarrow$ Use this as a new feat. importance score, adjusting for random chance

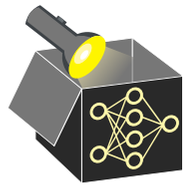
PIMP ALGORITHM



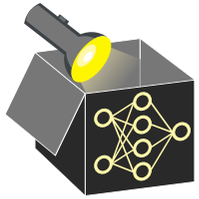
1. For $b \in \{1, \ldots, B\}$:
   - Permute response vector $\mathbf{y}$, denote permuted target as $\mathbf{y}^{(b)}$
   - Retrain model on data $(\mathbf{X}, \mathbf{y}^{(b)})$ with permuted target
   - Compute feature importance $\widehat{\text{PFI}}_j^{(b)}$ for each feature j (under H_0)

PIMP ALGORITHM

1. For $b \in \{1, \ldots, B\}$:
   - Permute response vector $\mathbf{y}$, denote permuted target as $\mathbf{y}^{(b)}$
   - Retrain model on data $(\mathbf{X}, \mathbf{y}^{(b)})$ with permuted target
   - Compute feature importance $\widehat{\text{PFI}}_j^{(b)}$ for each feature j (under H_0)

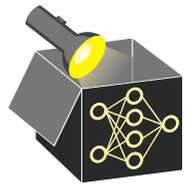


PIMP ALGORITHM



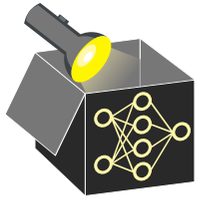
1. For $b \in \{1, \ldots, B\}$:
   - Permute response vector $\mathbf{y}$, denote permuted target as $\mathbf{y}^{(b)}$
   - Retrain model on data $(\mathbf{X}, \mathbf{y}^{(b)})$ with permuted target
   - Compute feature importance $\widehat{\text{PFI}}_j^{(b)}$ for each feature j (under H_0)
2. Train model on original data $(\mathbf{X}, \mathbf{y})$ with unpermuted target

PIMP ALGORITHM



1. For $b \in \{1, \ldots, B\}$:
   - Permute response vector $\mathbf{y}$, denote permuted target as $\mathbf{y}^{(b)}$
   - Retrain model on data $(\mathbf{X}, \mathbf{y}^{(b)})$ with permuted target
   - Compute feature importance $\widehat{\text{PFI}}_j^{(b)}$ for each feature j (under H_0)
2. Train model on original data $(\mathbf{X}, \mathbf{y})$ with unpermuted target

PIMP ALGORITHM



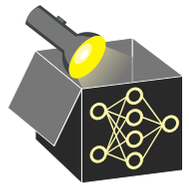
1. For $b \in \{1, \ldots, B\}$:
   - Permute response vector $\mathbf{y}$, denote permuted target as $\mathbf{y}^{(b)}$
   - Retrain model on data $(\mathbf{X}, \mathbf{y}^{(b)})$ with permuted target
   - Compute feature importance $\widehat{\text{PFI}}_j^{(b)}$ for each feature j (under H_0)
2. Train model on original data $(\mathbf{X}, \mathbf{y})$ with unpermuted target
3. For each feature $j \in \{1, \ldots, p\}$:
   - Compute $\widehat{\text{PFI}}_j^{\text{obs}}$ for the model without permutation of y (under H_1)
   - Fit probability distribution to all PFI scores $\{\widehat{\text{PFI}}_j^{(b)}\}_{b=1}^B$ (under H_0) e.g., by assuming Gaussian/lognormal/gamma distribution (parametric)
   - Compute p-value: Probability that null importance exceeds observed:
     - parametric by taking tail probability of assumed distribution

$$\mathbb{P}(\widehat{\text{PFI}}_j^{(m)} \geq \widehat{\text{PFI}}_j^{\text{obs}})$$

     - non-parametric by computing empirical tail probability:

$$p_j := \frac{1}{B} \sum_{b=1}^{B} \mathbb{I}[\widehat{\text{PFI}}_j^{(b)} \geq \widehat{\text{PFI}}_j^{\text{obs}}]$$

PIMP ALGORITHM



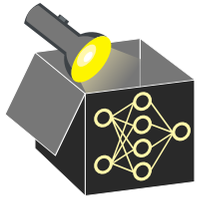
1. For $b \in \{1, \ldots, B\}$:
   - Permute response vector $\mathbf{y}$, denote permuted target as $\mathbf{y}^{(b)}$
   - Retrain model on data $(\mathbf{X}, \mathbf{y}^{(b)})$ with permuted target
   - Compute feature importance $\widehat{\text{PFI}}_j^{(b)}$ for each feature j (under H_0)
2. Train model on original data $(\mathbf{X}, \mathbf{y})$ with unpermuted target
3. For each feature $j \in \{1, \ldots, p\}$:
   - Compute $\widehat{\text{PFI}}_j^{\text{obs}}$ for the model without permutation of y (under H_1)
   - Fit probability distribution to all PFI scores $\{\widehat{\text{PFI}}_j^{(b)}\}_{b=1}^B$ (under H_0) e.g., by assuming Gaussian/lognormal/gamma distrib (parametric)
   - Compute p-value: Prob. that null importance exceeds observed:
     - parametric by taking tail probability of assumed distribution

$$\mathbb{P}(\widehat{\text{PFI}}_j^{(m)} \geq \widehat{\text{PFI}}_j^{\text{obs}})$$

     - non-parametric by computing empirical tail probability:

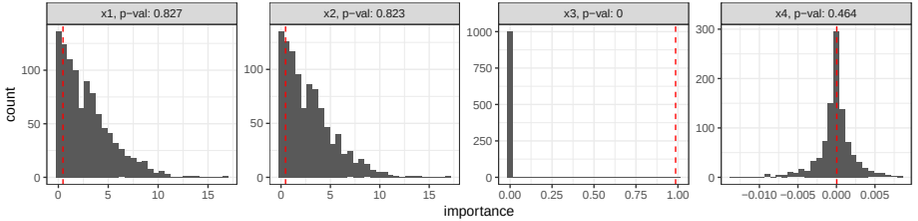
$$p_j := \frac{1}{B} \sum_{b=1}^{B} \mathbb{I}[\widehat{\text{PFI}}_j^{(b)} \geq \widehat{\text{PFI}}_j^{\text{obs}}]$$

PIMP FOR EXTRAPOLATION EXAMPLE



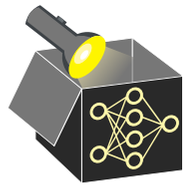
Recall: Let $y = x_3 + \epsilon_y$, with $\epsilon_y \sim \mathcal{N}(0, 0.1)$.

- $x_1 := \epsilon_1$, $x_2 := x_1 + \epsilon_2$ are highly correlated ($\epsilon_1 \sim \mathcal{N}(0, 1)$, $\epsilon_2 \sim \mathcal{N}(0, 0.01)$)
- $x_3 := \epsilon_3$, $x_4 := \epsilon_4$, with $\epsilon_3, \epsilon_4 \sim \mathcal{N}(0, 1)$ and all noise terms ϵ_j are independent
- Fitting a linear model yields $\hat{f}(\mathbf{x}) \approx 0.3x_1 - 0.3x_2 + x_3$



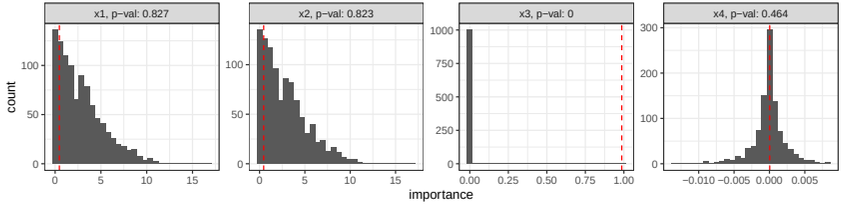
- Histograms: H_0 distribution of PFI scores after permuting y (1000 repetitions)
- Red: Observed PFI score (under H_1) $\rightsquigarrow$ compare against H_0 distribution
- Recall: PFI for x_1, x_2, x_3 is nonzero suggesting they are important (red lines)
- PIMP considers x_1, x_2 not significantly relevant (p-value > 0.05)

PIMP FOR EXTRAPOLATION EXAMPLE



Recall: Let $y = x_3 + \epsilon_y$, with $\epsilon_y \sim \mathcal{N}(0, 0.1)$.

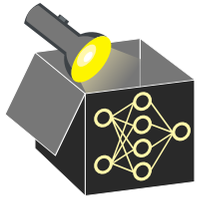
- $x_1 := \epsilon_1$, $x_2 := x_1 + \epsilon_2$; highly correlated ($\epsilon_1 \sim \mathcal{N}(0, 1)$, $\epsilon_2 \sim \mathcal{N}(0, 0.01)$)
- $x_3 := \epsilon_3$, $x_4 := \epsilon_4$, with $\epsilon_3, \epsilon_4 \sim \mathcal{N}(0, 1)$; all noise terms ϵ_j are indep.
- Fitting a linear model yields $\hat{f}(\mathbf{x}) \approx 0.3x_1 - 0.3x_2 + x_3$



- Histograms: H_0 distrib. of PFI scores after permuting y (1000 repetitions)
- Red: Observed PFI score (under H_1) $\rightsquigarrow$ compare against H_0 distribution
- Recall: PFI for x_1, x_2, x_3 is non-0 suggesting they are important (red lines)
- PIMP considers x_1, x_2 not significantly relevant (p-value > 0.05)

DIGRESSION: MULTIPLE TESTING

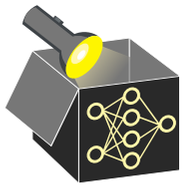
Romano et al. (2010)



- When should we reject H_0 for a given feature?
- PIMP conducts one hypothesis test per feature $\Rightarrow$ **multiple testing problem**
- With many tests, rejections of true H_0 just by chance (type-I errors) accumulate
- To account for this, control a suitable error rate, e.g., the **family-wise error rate**
  FWE: probability of making at least one type-I error across all tests
- A classical method is the **Bonferroni correction**:
  reject H_0 if p-value $< \alpha/m$ where m is the number of tests

DIGRESSION: MULTIPLE TESTING

ROMANO_2010



- When should we reject H_0 for a given feature?
- PIMP conducts one hypothesis test per feature
  $\Rightarrow$ **multiple testing problem**
- With many tests, rejections of true H_0 just by chance (type-I errors) accumulate
- To account for this, control a suitable error rate, e.g., the **family-wise error rate**
  FWE: probability of making at least one type-I error across all tests
- A classical method is the **Bonferroni correction**:
  reject H_0 if p-value $< \alpha/m$ where m is the number of tests

Interpretable Machine Learning

Conditional Feature Importance (CFI)

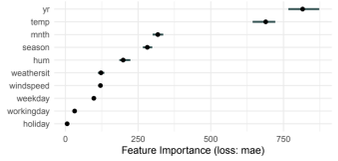


Figure: Bike Sharing Dataset

Learning goals

- Extrapolation and Conditional Sampling
- Conditional Feature Importance (CFI)
- Interpretation of CFI and difference to PFI

Interpretable Machine Learning

Feature Importances 1
Conditional Feature Importance (CFI)

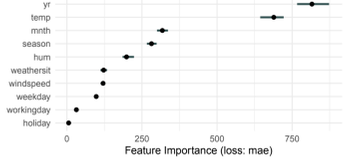


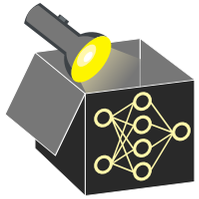
Figure: Bike Sharing Dataset

Learning goals

- Extrapolation and Conditional Sampling
- Conditional Feature Importance (CFI)
- Interpretation of CFI and difference to PFI

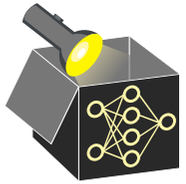
CFI MOTIVATION

- **PFI Idea:** Replace feature(s) X_S with perturbed $\tilde{X}_S$ to preserve marginal distibution $\mathbb{P}(X_S)$ so that $\tilde{X}_S \perp\!\!\!\perp Y$ (independent), e.g., by random permutations



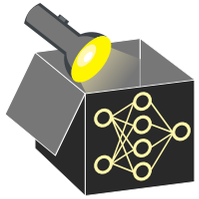
CFI MOTIVATION

- **PFI Idea:** Replace feature(s) X_S with perturbed $\tilde{X}_S$ to preserve marginal distib. $\mathbb{P}(X_S)$ so that $\tilde{X}_S \perp\!\!\!\perp Y$ (indep.), e.g., by random permutations



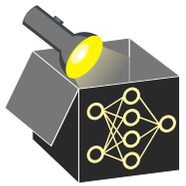
CFI MOTIVATION

- **PFI Idea:** Replace feature(s) X_S with perturbed $\tilde{X}_S$ to preserve marginal distibution $\mathbb{P}(X_S)$ so that $\tilde{X}_S \perp\!\!\!\perp Y$ (independent), e.g., by random permutations
- **Problem:** Breaks not only association between X_S and Y (what we want) but also between X_S and $X_{-S} \Rightarrow \mathbb{P}(X_S, X_{-S}) \neq \mathbb{P}(\tilde{X}_S, X_{-S})$ (extrapolation)

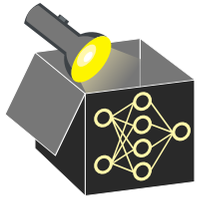


CFI MOTIVATION

- **PFI Idea:** Replace feature(s) X_S with perturbed $\tilde{X}_S$ to preserve marginal distib. $\mathbb{P}(X_S)$ so that $\tilde{X}_S \perp\!\!\!\perp Y$ (indep.), e.g., by random permutations
- **Problem:** Breaks not only association between X_S and Y (what we want) but also between X_S, $X_{-S} \Rightarrow \mathbb{P}(X_S, X_{-S}) \neq \mathbb{P}(\tilde{X}_S, X_{-S})$ (extrapolation)

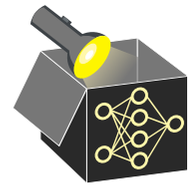


CFI MOTIVATION



- **PFI Idea:** Replace feature(s) X_S with perturbed $\tilde{X}_S$ to preserve marginal distibution $\mathbb{P}(X_S)$ so that $\tilde{X}_S \perp\!\!\!\perp Y$ (independent), e.g., by random permutations
- **Problem:** Breaks not only association between X_S and Y (what we want) but also between X_S and $X_{-S} \Rightarrow \mathbb{P}(X_S, X_{-S}) \neq \mathbb{P}(\tilde{X}_S, X_{-S})$ (extrapolation)
- **CFI Idea:** Replace feature(s) X_S with perturbed $\tilde{X}_S$ to preserve joint distibution so that $\mathbb{P}(X_S, X_{-S}) = \mathbb{P}(\tilde{X}_S, X_{-S})$ (no extrapolation) while still $\tilde{X}_S \perp\!\!\!\perp Y$

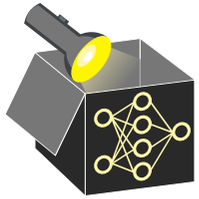
CFI MOTIVATION



- **PFI Idea:** Replace feature(s) X_S with perturbed $\tilde{X}_S$ to preserve marginal distib. $\mathbb{P}(X_S)$ so that $\tilde{X}_S \perp\!\!\!\perp Y$ (indep.), e.g., by random permutations
- **Problem:** Breaks not only association between X_S and Y (what we want) but also between X_S, $X_{-S} \Rightarrow \mathbb{P}(X_S, X_{-S}) \neq \mathbb{P}(\tilde{X}_S, X_{-S})$ (extrapolation)
- **CFI Idea:** Replace feature(s) X_S with perturbed $\tilde{X}_S$ to preserve joint distib. so that $\mathbb{P}(X_S, X_{-S}) = \mathbb{P}(\tilde{X}_S, X_{-S})$ (no extrapolation) while still $\tilde{X}_S \perp\!\!\!\perp Y$

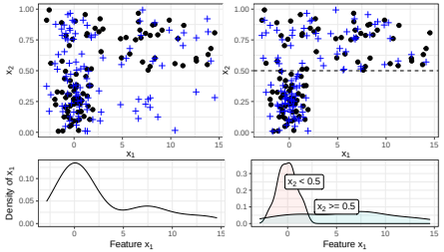
Interpretable Machine Learning – 1 / 6

CFI MOTIVATION



- **PFI Idea:** Replace feature(s) X_S with perturbed $\tilde{X}_S$ to preserve marginal distibution $\mathbb{P}(X_S)$ so that $\tilde{X}_S \perp\!\!\!\perp Y$ (independent), e.g., by random permutations
- **Problem:** Breaks not only association between X_S and Y (what we want) but also between X_S and $X_{-S} \Rightarrow \mathbb{P}(X_S, X_{-S}) \neq \mathbb{P}(\tilde{X}_S, X_{-S})$ (extrapolation)
- **CFI Idea:** Replace feature(s) X_S with perturbed $\tilde{X}_S$ to preserve joint distibution so that $\mathbb{P}(X_S, X_{-S}) = \mathbb{P}(\tilde{X}_S, X_{-S})$ (no extrapolation) while still $\tilde{X}_S \perp\!\!\!\perp Y$

Example: Conditional permutation scheme
Black dots: $X_2 \sim \mathcal{U}(0,1)$ and $X_1 \sim \mathcal{N}(0,1)$ (if $X_2 < 0.5$) or $\mathcal{N}(4,4)$ (if $X_2 \geq 0.5$)

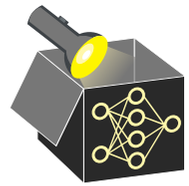


Left: For $X_2 < 0.5$, permuting X_1 (crosses) preserves marginal (but not joint) distribution
$\rightsquigarrow$ Bottom: Marginal density of X_1

Right: Permuting X_1 within subgroups $X_2 < 0.5$ & $X_2 \geq 0.5$ reduces extrapolation
$\rightsquigarrow$ Bottom: X_1-density conditional on groups

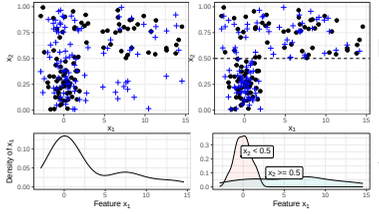
▸ Molnar et. al (2020)

CFI MOTIVATION



- **PFI Idea:** Replace feature(s) X_S with perturbed $\tilde{X}_S$ to preserve marginal distib. $\mathbb{P}(X_S)$ so that $\tilde{X}_S \perp\!\!\!\perp Y$ (indep.), e.g., by random permutations
- **Problem:** Breaks not only association between X_S and Y (what we want) but also between X_S, $X_{-S} \Rightarrow \mathbb{P}(X_S, X_{-S}) \neq \mathbb{P}(\tilde{X}_S, X_{-S})$ (extrapolation)
- **CFI Idea:** Replace feature(s) X_S with perturbed $\tilde{X}_S$ to preserve joint distib. so that $\mathbb{P}(X_S, X_{-S}) = \mathbb{P}(\tilde{X}_S, X_{-S})$ (no extrapolation) while still $\tilde{X}_S \perp\!\!\!\perp Y$

Example: Conditional permutation scheme
Black dots: $X_2 \sim \mathcal{U}(0,1)$ and $X_1 \sim \mathcal{N}(0,1)$ (if $X_2 < 0.5$) or $\mathcal{N}(4,4)$ (if $X_2 \geq 0.5$)

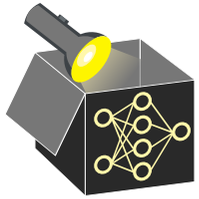


Left: For $X_2 < 0.5$, permuting X_1 (crosses) preserves marginal (but not joint) distrib.
$\rightsquigarrow$ Bottom: Marginal density of X_1

Right: Permuting X_1 within subgroups $X_2 < 0.5$ & $X_2 \geq 0.5$ reduces extrapolation
$\rightsquigarrow$ Bottom: X_1-density cond. on groups

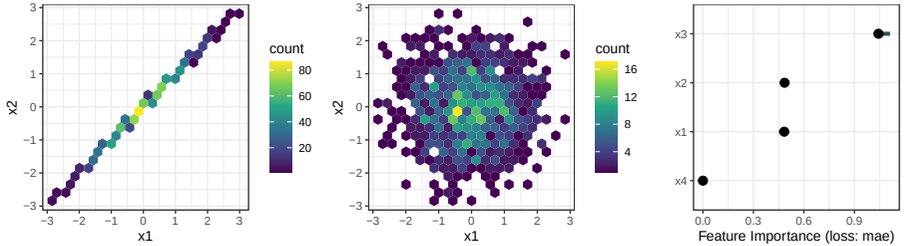
▸ Molnar 2020

RECALL: EXTRAPOLATION IN PFI



Recall: Let $y = x_3 + \epsilon_y$, with $\epsilon_y \sim \mathcal{N}(0, 0.1)$.

- $x_1 := \epsilon_1$, $x_2 := x_1 + \epsilon_2$ are highly correlated ($\epsilon_1 \sim \mathcal{N}(0,1)$, $\epsilon_2 \sim \mathcal{N}(0, 0.01)$)
- $x_3 := \epsilon_3$, $x_4 := \epsilon_4$, with $\epsilon_3, \epsilon_4 \sim \mathcal{N}(0,1)$ and all noise terms ϵ_j are independent
- Fitting a linear model yields $\hat{f}(\mathbf{x}) \approx 0.3x_1 - 0.3x_2 + x_3$



Hexbin plot of (x_1, x_2) before (left) and after (center) permuting x_1; PFI scores (right).

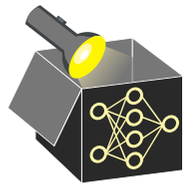
$\Rightarrow$ x_1, x_2 cancel in $\hat{f}$ and should be irrelevant

$\Rightarrow$ But PFI evaluates model on unrealistic inputs (caused by permutation)

$\rightsquigarrow$ *PFI* > 0 for x_1, x_2 due to extrapolation

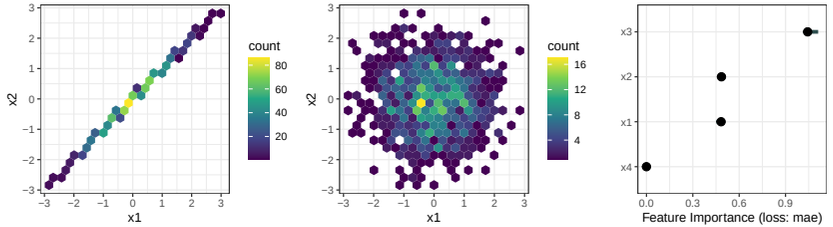
$\rightsquigarrow$ x_1, x_2 are misleadingly considered relevant

RECALL: EXTRAPOLATION IN PFI



Recall: Let $y = x_3 + \epsilon_y$, with $\epsilon_y \sim \mathcal{N}(0, 0.1)$.

- $x_1 := \epsilon_1$, $x_2 := x_1 + \epsilon_2$; highly correlated ($\epsilon_1 \sim \mathcal{N}(0,1)$, $\epsilon_2 \sim \mathcal{N}(0, 0.01)$)
- $x_3 := \epsilon_3$, $x_4 := \epsilon_4$, with $\epsilon_3, \epsilon_4 \sim \mathcal{N}(0,1)$; all noise terms ϵ_j are indep.
- Fitting a linear model yields $\hat{f}(\mathbf{x}) \approx 0.3x_1 - 0.3x_2 + x_3$



Hexbin plot of (x_1, x_2) before (left) and after (center) permuting x_1;
PFI scores (right).

$\Rightarrow$ x_1, x_2 cancel in $\hat{f}$ and should be irrelevant

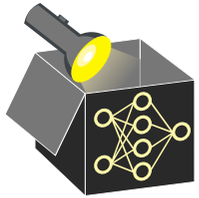
$\Rightarrow$ But PFI evaluates model on unrealistic inputs (caused by permutation)

$\rightsquigarrow$ *PFI* > 0 for x_1, x_2 due to extrapolation

$\rightsquigarrow$ x_1, x_2 are misleadingly considered relevant

CFI

Strobl et al. (2008) | Hooker et al. (2021)



CFI for X_S using test data $\mathcal{D}$:

- Measure the error **with unperturbed features** x_S.
- Measure the error **with perturbed feature values** $\tilde{x}_S \sim \mathbb{P}(X_S|X_{-S})$
- Repeat perturbing X_S (e.g., m times) and average difference of both errors:

$$\widehat{CFI}_S = \frac{1}{m} \sum_{k=1}^{m} \mathcal{R}_{\text{emp}}(\hat{f}, \tilde{\mathcal{D}}_{(k)}^{S|-S}) - \mathcal{R}_{\text{emp}}(\hat{f}, \mathcal{D})$$

Here, $\tilde{\mathcal{D}}^{S|-S}$ denotes data, where x_S values are conditionally resampled given x_{-S}.

Illustrative example: Conditional permutation when X_{-S} is categorical:

Original Data

ID	X_{-S}	X_S
1	A	3.1
2	A	2.7
3	A	3.4
4	B	6.0
5	B	5.4
6	B	6.2

Permuted Conditionally on X_{-S}

ID	X_{-S}	X_S
1	A	2.7
2	A	3.1
3	A	3.4
4	B	6.2
5	B	6.0
6	B	5.4

Here, X_S is permuted *within* each group of X_{-S} to preserve $\mathbb{P}(X_S, X_{-S})$.

CFI

STROBL_2008 | HOOKER_2021

CFI for X_S using test data $\mathcal{D}$:

- Measure the error **with unperturbed features** x_S.
- Measure the error **with perturbed feature values** $x_S \sim \mathbb{P}(X_S|X_{-S})$
- Repeat perturbing X_S (e.g., m times) and avg. difference of both errors:

$$\widehat{CFI}_S = \frac{1}{m} \sum_{k=1}^{m} \mathcal{R}_{\text{emp}}(\hat{f}, \mathcal{D}S{-}S_{(k)}) - \mathcal{R}_{\text{emp}}(\hat{f}, \mathcal{D})$$

Here, $\mathcal{D}S{-}S$ denotes data, where x_S values are conditionally resampled given x_{-S}.

Illustrative example: Conditional permutation when X_{-S} is categorical:

Original Data

ID	X_{-S}	X_S
1	A	3.1
2	A	2.7
3	A	3.4
4	B	6.0
5	B	5.4
6	B	6.2

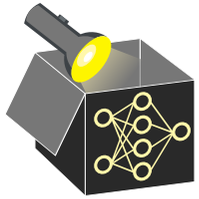
Permuted Conditionally on X_{-S}

ID	X_{-S}	X_S
1	A	2.7
2	A	3.1
3	A	3.4
4	B	6.2
5	B	6.0
6	B	5.4

Here, X_S is permuted *within* each group of X_{-S} to preserve $\mathbb{P}(X_S, X_{-S})$.

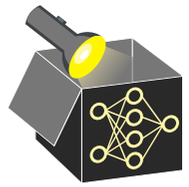
IMPLICATIONS OF CFI ▸ König et al. (2020)

Interpretation: Due to the conditional sampling w.r.t. all other features, CFI quantifies a feature's unique contribution to the model performance.

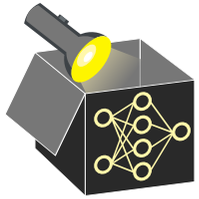


IMPLICATIONS OF CFI ▸ K_NIG_ET_2020

Interpretation: Due to the conditional sampling w.r.t. all other features, CFI quantifies a feature's unique contribution to the model performance.



IMPLICATIONS OF CFI ▸ König et al. (2020)

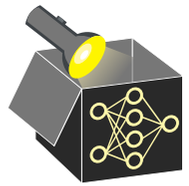


Interpretation: Due to the conditional sampling w.r.t. all other features, CFI quantifies a feature's unique contribution to the model performance.

Entanglement with data:

- If feature x_S does not contribute unique information about y, i.e., $x_S \perp\!\!\!\perp y | x_{-S}$ $\Rightarrow$ CFI $= 0$
- Why? Under the conditional independence $\mathbb{P}(\tilde{X}_S, X_{-S}, Y) = \mathbb{P}(X_S, X_{-S}, Y)$ $\rightsquigarrow$ no prediction-relevant information is destroyed by permutation of x_S conditional on x_{-S}

IMPLICATIONS OF CFI ▸ K_NIG_ET_2020

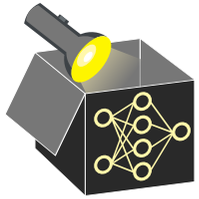


Interpretation: Due to the conditional sampling w.r.t. all other features, CFI quantifies a feature's unique contribution to the model performance.

Entanglement with data:

- If feat x_S does not contrib. unique information about y, i.e., $x_S \perp\!\!\!\perp y | x_{-S}$ $\Rightarrow$ CFI $= 0$
- Why? Under the conditional indep. $\mathbb{P}(X_S, X_{-S}, Y) = \mathbb{P}(X_S, X_{-S}, Y)$ $\rightsquigarrow$ no prediction-relevant information is destroyed by permutation of x_S conditional on x_{-S}

IMPLICATIONS OF CFI König et al. (2020)



Interpretation: Due to the conditional sampling w.r.t. all other features, CFI quantifies a feature's unique contribution to the model performance.

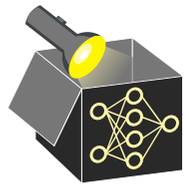
Entanglement with data:

- If feature x_S does not contribute unique information about y, i.e., $x_S \perp\!\!\!\perp y | x_{-S}$ $\Rightarrow$ CFI $= 0$
- Why? Under the conditional independence $\mathbb{P}(\tilde{X}_S, X_{-S}, Y) = \mathbb{P}(X_S, X_{-S}, Y)$ $\rightsquigarrow$ no prediction-relevant information is destroyed by permutation of x_S conditional on x_{-S}

Entanglement with model:

- If the model does not use a feature $\Rightarrow$ CFI $= 0$
- Why? Then the prediction is not affected by any perturbation of the feature $\rightsquigarrow$ model performance does not change after conditional permutation

IMPLICATIONS OF CFI K_NIG_ET_2020



Interpretation: Due to the conditional sampling w.r.t. all other features, CFI quantifies a feature's unique contribution to the model performance.

Entanglement with data:

- If feat x_S does not contrib. unique information about y, i.e., $x_S \perp\!\!\!\perp y | x_{-S}$ $\Rightarrow$ CFI $= 0$
- Why? Under the conditional indep. $\mathbb{P}(X_S, X_{-S}, Y) = \mathbb{P}(X_S, X_{-S}, Y)$ $\rightsquigarrow$ no prediction-relevant information is destroyed by permutation of x_S conditional on x_{-S}

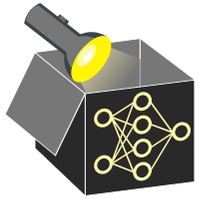
Entanglement with model:

- If the model does not use a feature $\Rightarrow$ CFI $= 0$
- Why? Then the prediction is not affected by any perturbation of the feat $\rightsquigarrow$ model performance does not change after conditional permutation

IMPLICATIONS OF CFI

Can we gain insight into whether ...

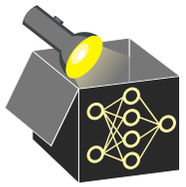
1. the feature x_j is causal for the prediction?
   - $CFI_j \neq 0 \Rightarrow$ model relies on x_j (converse does not hold, see next slide)



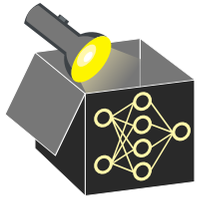
IMPLICATIONS OF CFI

Can we gain insight into whether ...

1. the feature x_j is causal for the prediction?
   - $CFI_j \neq 0 \Rightarrow$ model relies on x_j
   (converse does not hold, see next slide)



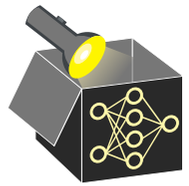
IMPLICATIONS OF CFI



Can we gain insight into whether ...

1. the feature x_j is causal for the prediction?
   - $CFI_j \neq 0 \Rightarrow$ model relies on x_j (converse does not hold, see next slide)
2. the variable x_j contains prediction-relevant information?
   - If $x_j \not\perp\!\!\!\perp y$ but $x_j \perp\!\!\!\perp y | x_{-j}$ (e.g., x_j and x_{-j} share information) $\Rightarrow CFI_j = 0$
   - x_j is not exploited by model (regardless of whether it is useful for y or not) $\Rightarrow CFI_j = 0$

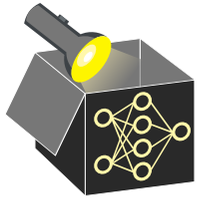
IMPLICATIONS OF CFI



Can we gain insight into whether ...

1. the feature x_j is causal for the prediction?
   - $CFI_j \neq 0 \Rightarrow$ model relies on x_j (converse does not hold, see next slide)
2. the variable x_j contains prediction-relevant information?
   - If $x_j \not\perp\!\!\!\perp y$ but $x_j \perp\!\!\!\perp y | x_{-j}$ (e.g., x_j and x_{-j} share information) $\Rightarrow CFI_j = 0$
   - x_j is not exploited by model (regardless of its usefulness for y) $\Rightarrow CFI_j = 0$

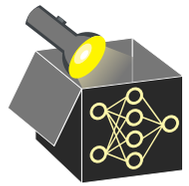
IMPLICATIONS OF CFI



Can we gain insight into whether ...

1. the feature x_j is causal for the prediction?
   - $CFI_j \neq 0 \Rightarrow$ model relies on x_j (converse does not hold, see next slide)
2. the variable x_j contains prediction-relevant information?
   - If $x_j \not\perp\!\!\!\perp y$ but $x_j \perp\!\!\!\perp y|x_{-j}$ (e.g., x_j and x_{-j} share information) $\Rightarrow CFI_j = 0$
   - x_j is not exploited by model (regardless of whether it is useful for y or not) $\Rightarrow CFI_j = 0$
3. Does the model require access to x_j to achieve its prediction performance?
   - $CFI_j \neq 0 \Rightarrow x_j$ contributes unique information (meaning $x_j \not\perp\!\!\!\perp y|x_{-j}$)
   - Only uncovers the relationships that were exploited by the model

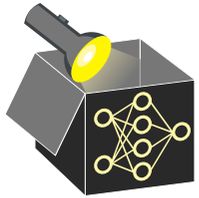
IMPLICATIONS OF CFI



Can we gain insight into whether ...

1. the feature x_j is causal for the prediction?
   - $CFI_j \neq 0 \Rightarrow$ model relies on x_j
     (converse does not hold, see next slide)
2. the variable x_j contains prediction-relevant information?
   - If $x_j \not\perp\!\!\!\perp y$ but $x_j \perp\!\!\!\perp y|x_{-j}$ (e.g., x_j and x_{-j} share information)
     $\Rightarrow CFI_j = 0$
   - x_j is not exploited by model (regardless of its usefulness for y)
     $\Rightarrow CFI_j = 0$
3. Does the model need access to x_j to achieve its prediction performance?
   - $CFI_j \neq 0 \Rightarrow x_j$ contributes unique information (meaning $x_j \not\perp\!\!\!\perp y|x_{-j}$)
   - Only uncovers the relationships that were exploited by the model

EXTRAPOLATION: COMPARE PFI AND CFI



Recall: Let $y = x_3 + \epsilon_y$, with $\epsilon_y \sim \mathcal{N}(0, 0.1)$.

- $x_1 := \epsilon_1$, $x_2 := x_1 + \epsilon_2$ are highly correlated ($\epsilon_1 \sim \mathcal{N}(0, 1)$, $\epsilon_2 \sim \mathcal{N}(0, 0.01)$)
- $x_3 := \epsilon_3$, $x_4 := \epsilon_4$, with $\epsilon_3, \epsilon_4 \sim \mathcal{N}(0, 1)$ and all noise terms ϵ_j are independent
- Fitting a linear model yields $\hat{f}(\mathbf{x}) \approx 0.3x_1 - 0.3x_2 + x_3$

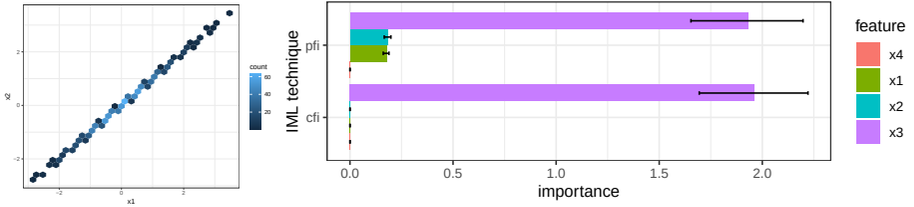
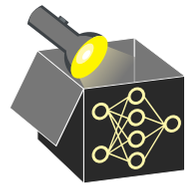


Figure: Density plot for x_1, x_2 before permuting x_1 (left). PFI and CFI (right).

- x_1 and x_2 cancel in $\hat{f}(\mathbf{x})$ and should be irrelevant for the prediction
- PFI evaluates model on unrealistic obs. $\rightsquigarrow$ x_1, x_2 appear relevant (PFI > 0)
- CFI evaluates model on realistic obs. (due to conditional sampling)
  $\rightsquigarrow$ x_1, x_2 appear irrelevant (CFI $= 0$)

EXTRAPOLATION: COMPARE PFI AND CFI



Recall: Let $y = x_3 + \epsilon_y$, with $\epsilon_y \sim \mathcal{N}(0, 0.1)$.

- $x_1 := \epsilon_1$, $x_2 := x_1 + \epsilon_2$; highly correlated ($\epsilon_1 \sim \mathcal{N}(0, 1)$, $\epsilon_2 \sim \mathcal{N}(0, 0.01)$)
- $x_3 := \epsilon_3$, $x_4 := \epsilon_4$, with $\epsilon_3, \epsilon_4 \sim \mathcal{N}(0, 1)$; all noise terms ϵ_j are indep.
- Fitting a linear model yields $\hat{f}(\mathbf{x}) \approx 0.3x_1 - 0.3x_2 + x_3$

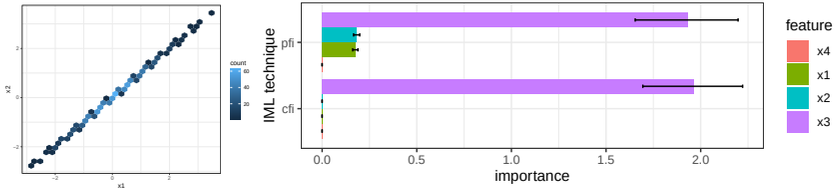


Figure: Density plot for x_1, x_2 before permuting x_1 (left). PFI and CFI (right).

- x_1 and x_2 cancel in $\hat{f}(\mathbf{x})$ and should be irrelevant for the prediction
- PFI evaluates model on unrealistic obs.
  $\rightsquigarrow$ x_1, x_2 appear relevant (PFI > 0)
- CFI evaluates model on realistic obs. (due to conditional sampling)
  $\rightsquigarrow$ x_1, x_2 appear irrelevant (CFI $= 0$)

Interpretable Machine Learning

Shapley Additive Global Importance (SAGE)

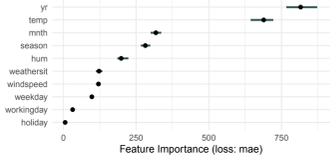


Figure: Bike Sharing Dataset

Learning goals

- How SAGE fairly distributes importance
- Definition of SAGE value function
- Difference SAGE value function and SAGE values
- Marginal and Conditional SAGE

Interpretable Machine Learning

Feature Importances 1
Shapley Additive Global Importance (SAGE)

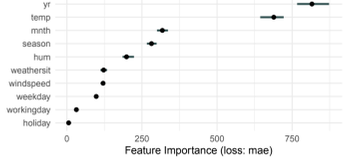
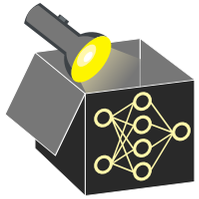


Figure: Bike Sharing Dataset

Learning goals

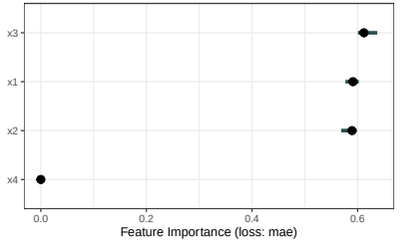
- How SAGE fairly distributes importance
- Definition of SAGE value function
- Difference SAGE value function and SAGE values
- Marginal and Conditional SAGE

CHALLENGE: FAIR ATTRIBUTION OF IMPORTANCE



Recap:

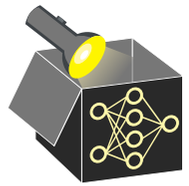
- Data: $x_1, \ldots, x_4$ uniformly sampled from $[-1, 1]$
- DGP: $y := x_1 x_2 + x_3 + \epsilon_Y$ with $\epsilon_Y \sim N(0, 1)$
- Model: $\hat{f}(x) \approx x_1 x_2 + x_3$



Although x_3 alone contributes as much to the prediction as x_1 and x_2 jointly, all three are considered equally relevant by PFI.

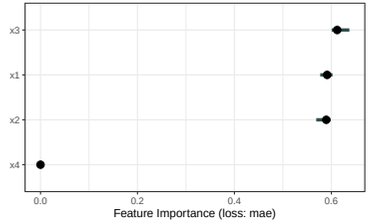
Reason: PFI assesses importance given that all remaining features are preserved. If we first permute x_1 and then x_2, permutation of x_2 would have no effect on the performance (and vice versa).

CHALLENGE: FAIR ATTRIBUTION OF IMPORTANCE



Recap:

- Data: $x_1, \ldots, x_4$ uniformly sampled from $[-1, 1]$
- DGP: $y := x_1 x_2 + x_3 + \epsilon_Y$ with $\epsilon_Y \sim N(0, 1)$
- Model: $\hat{f}(x) \approx x_1 x_2 + x_3$



Although x_3 alone contributes as much to the prediction as x_1 and x_2 jointly, all three are considered equally relevant by PFI.

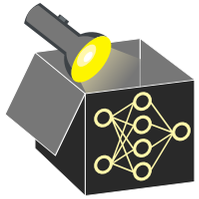
Reason: PFI assesses importance given that all remaining features are preserved. If we first permute x_1 and then x_2, permutation of x_2 would have no effect on the performance (and vice versa).

Interpretable Machine Learning – 1 / 12

SAGE IDEA

Covert et al. (2020)

SAGE: Use Shapley values to compute a fair attribution of importance (via model performance)



Idea:

- Feature importance attribution can be regarded as cooperative game
  $\rightsquigarrow$ features jointly contribute to achieve a certain model performance
- Players: features
- Payoff to be fairly distributed: model performance
- Surplus contribution of a feature depends on the coalition of features that are already accessible by the model

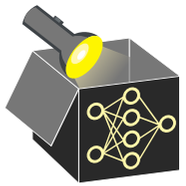
Note:

- Similar idea (called SFIMP) was proposed in Casalicchio et al. (2018)
- Definition based on model refits was proposed in context of feature selection in Cohen et al. (2007)

SAGE IDEA

COVERT_2020

SAGE: Use Shapley values to compute a fair attribution of importance (via model performance)



Idea:

- Feature importance attribution can be regarded as cooperative game
  $\rightsquigarrow$ features jointly contribute to achieve a certain model performance
- Players: features
- Payoff to be fairly distributed: model performance
- Surplus contribution of a feature depends on the coalition of features that are already accessible by the model

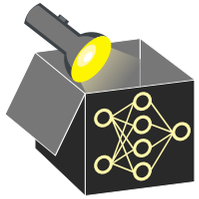
Note:

- Similar idea (called SFIMP) was proposed in Casalicchio 2018
- Definition based on model refits was proposed in context of feature selection in Cohen 2007

SAGE - VALUE FUNCTION

Removal Idea: To deprive information of the non-coalition features $-S$ from the model, marginalize the prediction function over the features $-S$ to be "dropped".

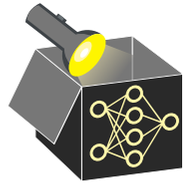
$$\hat{f}_S(x_S) = \mathbb{E}[\hat{f}(x)|X_S = x_S]$$



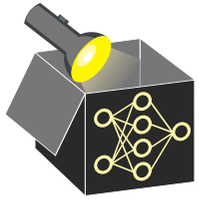
SAGE - VALUE FUNCTION

Removal Idea: To deprive information of the non-coalition features $-S$ from the model, marginalize the prediction function over feats $-S$ to be "dropped".

$$\hat{f}_S(x_S) = \mathbb{E}[\hat{f}(x)|X_S = x_S]$$



SAGE - VALUE FUNCTION



Removal Idea: To deprive information of the non-coalition features $-S$ from the model, marginalize the prediction function over the features $-S$ to be "dropped".

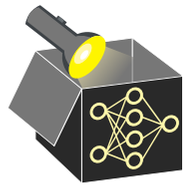
$$\hat{f}_S(x_S) = \mathbb{E}[\hat{f}(x)|X_S = x_S]$$

SAGE value function:

$$v_{\hat{f}}(S) = \mathcal{R}(\hat{f}_\emptyset) - \mathcal{R}(\hat{f}_S), \text{ where } \mathcal{R}(\hat{f}_S) = \mathbb{E}_{Y,X_S}[L(y, \hat{f}_S(x_S))]$$

$\rightsquigarrow$ Quantify the predictive power of a coalition S in terms of reduction in risk
$\rightsquigarrow$ Risk of predictor $\hat{f}_S(x_S)$ is compared to the risk of the mean prediction $\hat{f}_\emptyset$

SAGE - VALUE FUNCTION



Removal Idea: To deprive information of the non-coalition features $-S$ from the model, marginalize the prediction function over the features $-S$ to be "dropped".

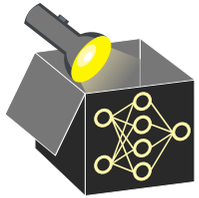
$$\hat{f}_S(x_S) = \mathbb{E}[\hat{f}(x)|X_S = x_S]$$

SAGE value function:

$$v_{\hat{f}}(S) = \mathcal{R}(\hat{f}_\emptyset) - \mathcal{R}(\hat{f}_S), \text{ where } \mathcal{R}(\hat{f}_S) = \mathbb{E}_{Y,X_S}[L(y, \hat{f}_S(x_S))]$$

$\rightsquigarrow$ Quantify the predictive power of a coalition S in terms of reduction in risk
$\rightsquigarrow$ Risk of predictor $\hat{f}_S(x_S)$ is compared to the risk of the mean prediction $\hat{f}_\emptyset$

SAGE - VALUE FUNCTION



Removal Idea: To deprive information of the non-coalition features $-S$ from the model, marginalize the prediction function over the features $-S$ to be "dropped".

$$\hat{f}_S(x_S) = \mathbb{E}[\hat{f}(x)|X_S = x_S]$$

SAGE value function:

$$v_{\hat{f}}(S) = \mathcal{R}(\hat{f}_\emptyset) - \mathcal{R}(\hat{f}_S), \text{ where } \mathcal{R}(\hat{f}_S) = \mathbb{E}_{Y,X_S}[L(y, \hat{f}_S(x_S))]$$

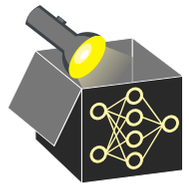
$\rightsquigarrow$ Quantify the predictive power of a coalition S in terms of reduction in risk
$\rightsquigarrow$ Risk of predictor $\hat{f}_S(x_S)$ is compared to the risk of the mean prediction $\hat{f}_\emptyset$

Surplus contribution of feature x_j over coalition x_S:

$$v_{\hat{f}}(S \cup \{j\}) - v_{\hat{f}}(S) = \mathcal{R}(\hat{f}_S) - \mathcal{R}(\hat{f}_{S\cup\{j\}})$$

$\rightsquigarrow$ Quantifies the added value of feature j when it is added to coalition S

SAGE - VALUE FUNCTION



Removal Idea: To deprive information of the non-coalition features $-S$ from the model, marginalize the prediction function over feats $-S$ to be "dropped".

$$\hat{f}_S(x_S) = \mathbb{E}[\hat{f}(x)|X_S = x_S]$$

SAGE value function:

$$v_{\hat{f}}(S) = \mathcal{R}(\hat{f}_\emptyset) - \mathcal{R}(\hat{f}_S), \text{ where } \mathcal{R}(\hat{f}_S) = \mathbb{E}_{Y,X_S}[L(y, \hat{f}_S(x_S))]$$

$\rightsquigarrow$ Quantify the predictive power of a coalition S in terms of reduction in risk
$\rightsquigarrow$ Risk of predictor $\hat{f}_S(x_S)$ is compared to the risk of the mean prediction $\hat{f}_\emptyset$

Surplus contribution of feature x_j over coalition x_S:

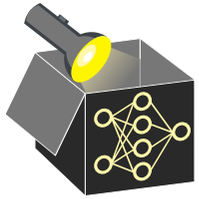
$$v_{\hat{f}}(S \cup \{j\}) - v_{\hat{f}}(S) = \mathcal{R}(\hat{f}_S) - \mathcal{R}(\hat{f}_{S\cup\{j\}})$$

$\rightsquigarrow$ Quantifies the added value of feature j when it is added to coalition S

SAGE - MARGINAL AND CONDITIONAL SAMPLING

When computing the marginalized prediction $\hat{f}_S(x_S)$, the "dropped" features can be sampled from

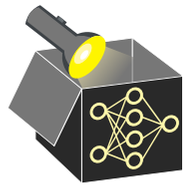
- the marginal distribution $\mathbb{P}(x_{-S}) \Rightarrow$ marginal SAGE
- the conditional distribution $\mathbb{P}(x_{-S}|x_S) \Rightarrow$ conditional SAGE



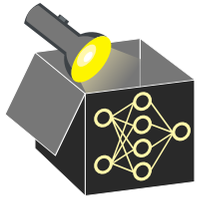
SAGE - MARGINAL AND COND. SAMPLING

When computing the marginalized prediction $\hat{f}_S(x_S)$, the "dropped" features can be sampled from

- the marginal distribution $\mathbb{P}(x_{-S}) \Rightarrow$ marginal SAGE
- the conditional distribution $\mathbb{P}(x_{-S}|x_S) \Rightarrow$ conditional SAGE



SAGE - MARGINAL AND CONDITIONAL SAMPLING



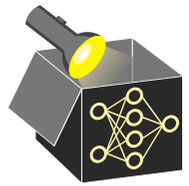
When computing the marginalized prediction $\hat{f}_S(x_S)$, the "dropped" features can be sampled from

- the marginal distribution $\mathbb{P}(x_{-S}) \Rightarrow$ marginal SAGE
- the conditional distribution $\mathbb{P}(x_{-S}|x_S) \Rightarrow$ conditional SAGE

Interpretation marginal sampling: $v(S)$ quantifies the reliance of the model on features x_S

- features x_S not being causal for the prediction $\Rightarrow v(S) = 0$

SAGE - MARGINAL AND COND. SAMPLING



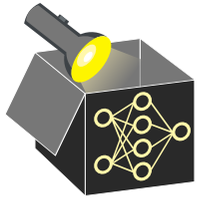
When computing the marginalized prediction $\hat{f}_S(x_S)$, the "dropped" features can be sampled from

- the marginal distribution $\mathbb{P}(x_{-S}) \Rightarrow$ marginal SAGE
- the conditional distribution $\mathbb{P}(x_{-S}|x_S) \Rightarrow$ conditional SAGE

Interpretation marginal sampling: $v(S)$ quantifies the reliance of the model on features x_S

- features x_S not being causal for the prediction $\Rightarrow v(S) = 0$

SAGE - MARGINAL AND CONDITIONAL SAMPLING



When computing the marginalized prediction $\hat{f}_S(x_S)$, the "dropped" features can be sampled from

- the marginal distribution $\mathbb{P}(x_{-S}) \Rightarrow$ marginal SAGE
- the conditional distribution $\mathbb{P}(x_{-S}|x_S) \Rightarrow$ conditional SAGE

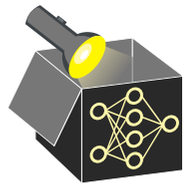
Interpretation marginal sampling: $v(S)$ quantifies the reliance of the model on features x_S

- features x_S not being causal for the prediction $\Rightarrow v(S) = 0$

Interpretation conditional sampling: $v(S)$ quantifies whether variables x_S contain prediction-relevant information (e.g. $y \not\perp\!\!\!\perp x_S$) that is (directly or indirectly) exploited by the model

- features x_S not being causal for the prediction $\not\Rightarrow v(S) = 0$
  - e.g., if x_1 and x_2 are perfectly correlated, even if only x_1 has a nonzero coefficient, both are considered equally important
- under model optimality, links to mutual information or the conditional variance exist

SAGE - MARGINAL AND COND. SAMPLING



When computing the marginalized prediction $\hat{f}_S(x_S)$, the "dropped" features can be sampled from

- the marginal distribution $\mathbb{P}(x_{-S}) \Rightarrow$ marginal SAGE
- the conditional distribution $\mathbb{P}(x_{-S}|x_S) \Rightarrow$ conditional SAGE

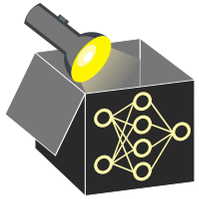
Interpretation marginal sampling: $v(S)$ quantifies the reliance of the model on features x_S

- features x_S not being causal for the prediction $\Rightarrow v(S) = 0$

Interpretation conditional sampling: $v(S)$ quantifies whether variables x_S contain prediction-relevant information (e.g. $y \not\perp\!\!\!\perp x_S$) that is (directly or indirectly) exploited by the model

- features x_S not being causal for the prediction $\not\Rightarrow v(S) = 0$
  - e.g., if x_1 and x_2 are perfectly correlated, even if only x_1 has a nonzero coefficient, both are considered equally important
- under model optimality, links to mutual information or the conditional variance exist

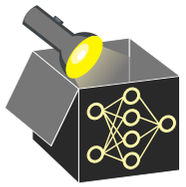
SAGE - MARGINAL AND CONDITIONAL SAMPLING



Example:

- $y = x_3 + \epsilon_y$
  $x_1 = \epsilon_1$
  $x_2 = x_1 + \epsilon_2$
  $x_3 = x_2 + \epsilon_3$ (all ϵ_j i.i.d.)
- Causal DAG:
  $x_1 \to x_2 \to x_3 \to y$
- Fitted LM:
  $\hat{f} \approx 0.95x_3 + 0.05x_2$

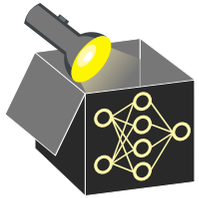
SAGE - MARGINAL AND COND. SAMPLING



Example:

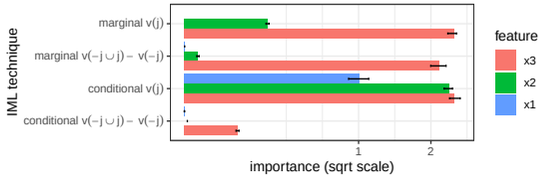
- $y = x_3 + \epsilon_y$
  $x_1 = \epsilon_1$
  $x_2 = x_1 + \epsilon_2$
  $x_3 = x_2 + \epsilon_3$ (all ϵ_j i.i.d.)
- Causal DAG:
  $x_1 \to x_2 \to x_3 \to y$
- Fitted LM:
  $\hat{f} \approx 0.95x_3 + 0.05x_2$

SAGE - MARGINAL AND CONDITIONAL SAMPLING



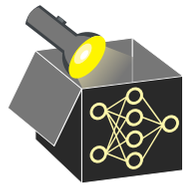
Example:

- $y = x_3 + \epsilon_y$
  $x_1 = \epsilon_1$
  $x_2 = x_1 + \epsilon_2$
  $x_3 = x_2 + \epsilon_3$ (all ϵ_j i.i.d.)
- Causal DAG:
  $x_1 \to x_2 \to x_3 \to y$
- Fitted LM:
  $\hat{f} \approx 0.95x_3 + 0.05x_2$



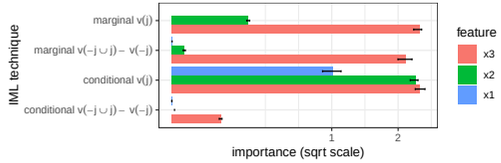
- Marginal $v(j)$ are only nonzero for features that are used by $\hat{f}$
- Conditional $v(j)$ are also nonzero for features that are not used by $\hat{f}$ (e.g., due to correlation)
- For conditional value function v, the difference $v(-j \cup j) - v(-j)$ quantifies the unique contribution of x_j over remaining features x_{-j}
  $\Rightarrow$ Since $y \perp\!\!\!\perp x_1, x_2 | x_3$, only $v(\{1,2,3\}) - v(\{1,2\})$ is nonzero (i.e., for feature $j = 3$)

SAGE - MARGINAL AND COND. SAMPLING



Example:

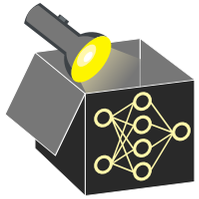
- $y = x_3 + \epsilon_y$
  $x_1 = \epsilon_1$
  $x_2 = x_1 + \epsilon_2$
  $x_3 = x_2 + \epsilon_3$ (all ϵ_j i.i.d.)
- Causal DAG:
  $x_1 \to x_2 \to x_3 \to y$
- Fitted LM:
  $\hat{f} \approx 0.95x_3 + 0.05x_2$



- Marginal $v(j)$ are only nonzero for features that are used by $\hat{f}$
- Conditional $v(j)$ are also nonzero for features that are not used by $\hat{f}$ (e.g., due to correlation)
- For conditional value function v, the difference $v(-j \cup j) - v(-j)$ quantifies the unique contribution of x_j over remaining features x_{-j}
  $\Rightarrow$ Since $y \perp\!\!\!\perp x_1, x_2 | x_3$, only $v(\{1,2,3\}) - v(\{1,2\})$ is nonzero (i.e., for feature $j = 3$)

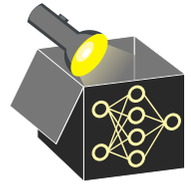
SAGE VALUE FUNCTIONS VERSUS SAGE VALUES

SAGE value function $v(S)$: measure contribution of a specific feature set over the empty coalition

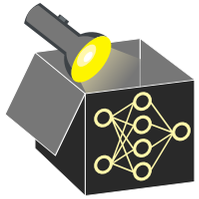


SAGE VALUE FUNCTIONS VS. SAGE VALUES

SAGE value function $v(S)$: measure contribution of a specific feature set over the empty coalition



SAGE VALUE FUNCTIONS VERSUS SAGE VALUES

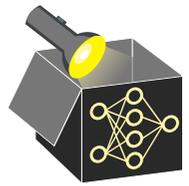


SAGE value function $v(S)$: measure contribution of a specific feature set over the empty coalition

SAGE values ϕ_j: fair attribution of importance

- can be computed by averaging the contribution of x_j over all feature orderings
- for feature permutation τ, the contribution of j in the set S_j^τ is given as $v(S_j^\tau \cup \{j\}) - v(S_j^\tau)$
  Note: S_j^τ is the set of features preceding j in permutation τ

SAGE VALUE FUNCTIONS VS. SAGE VALUES

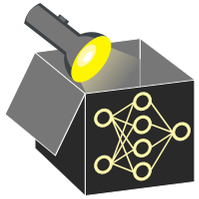


SAGE value function $v(S)$: measure contribution of a specific feature set over the empty coalition

SAGE values ϕ_j: fair attribution of importance

- can be computed by averaging the contribution of x_j over all feat orderings
- for feature permutation τ, the contribution of j in the set S_j^τ is given as $v(S_j^\tau \cup \{j\}) - v(S_j^\tau)$
  Note: S_j^τ is the set of features preceding j in permutation τ

SAGE VALUE FUNCTIONS VERSUS SAGE VALUES



SAGE value function $v(S)$: measure contribution of a specific feature set over the empty coalition

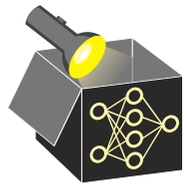
SAGE values ϕ_j: fair attribution of importance

- can be computed by averaging the contribution of x_j over all feature orderings
- for feature permutation τ, the contribution of j in the set S_j^τ is given as $v(S_j^\tau \cup \{j\}) - v(S_j^\tau)$
  Note: S_j^τ is the set of features preceding j in permutation τ

SAGE value approximation: Average over the contributions for M randomly sampled permutations

$$\phi_j = \frac{1}{M} \sum_{m=1}^{M} v(S_j^\tau \cup \{j\}) - v(S_j^\tau)$$

SAGE VALUE FUNCTIONS VS. SAGE VALUES



SAGE value function $v(S)$: measure contribution of a specific feature set over the empty coalition

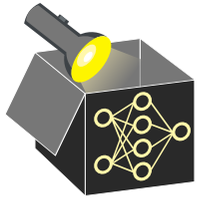
SAGE values ϕ_j: fair attribution of importance

- can be computed by averaging the contribution of x_j over all feat orderings
- for feature permutation τ, the contribution of j in the set S_j^τ is given as $v(S_j^\tau \cup \{j\}) - v(S_j^\tau)$
  Note: S_j^τ is the set of features preceding j in permutation τ

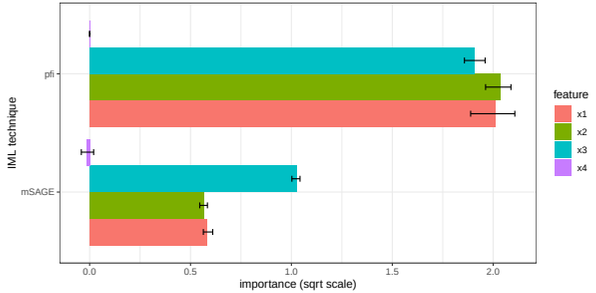
SAGE value approximation: Average over the contributions for M randomly sampled permutations

$$\phi_j = \frac{1}{M} \sum_{m=1}^{M} v(S_j^\tau \cup \{j\}) - v(S_j^\tau)$$

INTERACTION EXAMPLE REVISITED

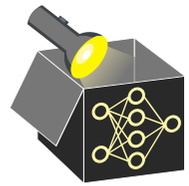


Recap: Data: $x_1, \ldots, x_4$ uniformly sampled from $\{-1, 1\}$ and $y := x_1 x_2 + x_3 + \epsilon_Y$ with $\epsilon_Y \sim N(0, 1)$. Model: $\hat{f}(x) \approx x_1 x_2 + x_3$.

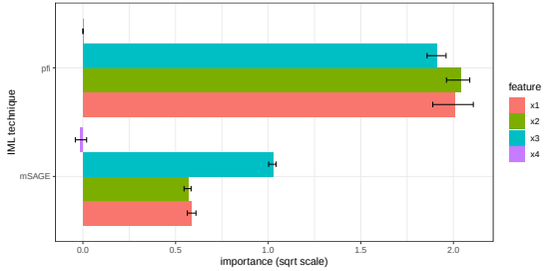


- PFI regards x_1, x_2 to be equally important as x_3
- Marginal SAGE fairly divides the contribution of the interaction x_1 and x_2

INTERACTION EXAMPLE REVISITED



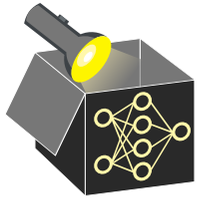
Recap: Data: $x_1, \ldots, x_4$ uniformly sampled from $\{-1, 1\}$ and $y := x_1 x_2 + x_3 + \epsilon_Y$ with $\epsilon_Y \sim N(0, 1)$. Model: $\hat{f}(x) \approx x_1 x_2 + x_3$.



- PFI regards x_1, x_2 to be equally important as x_3
- Marginal SAGE fairly divides the contribution of the interaction x_1 and x_2

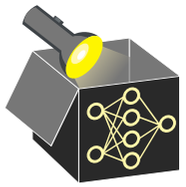
SAGE LOSS FUNCTIONS

When the loss-optimal model f^* is inspected using *conditional-sampling* based SAGE value functions, interesting links exist.



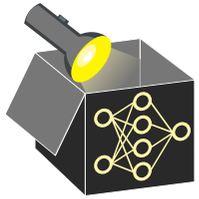
SAGE LOSS FUNCTIONS

When the loss-optimal model f^* is inspected using *conditional-sampling* based SAGE value functions, interesting links exist.



SAGE LOSS FUNCTIONS

When the loss-optimal model f^* is inspected using *conditional-sampling* based SAGE value functions, interesting links exist.

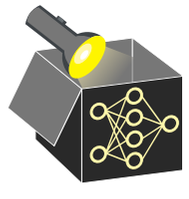


For cross-entropy loss:

- value function is the mutual information: $v_{f^*}(S) = I(y; x_S)$
- surplus contribution of a feature x_j is the conditional mutual information: $v_{f^*}(S \cup \{j\}) - v_{f^*}(S) = I(y, x_i | x_S)$

SAGE LOSS FUNCTIONS

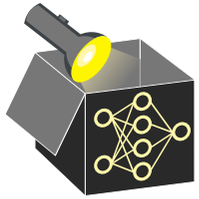
When the loss-optimal model f^* is inspected using *conditional-sampling* based SAGE value functions, interesting links exist.



For cross-entropy loss:

- value function is the mutual information: $v_{f^*}(S) = I(y; x_S)$
- surplus contribution of a feature x_j is the conditional mutual information: $v_{f^*}(S \cup \{j\}) - v_{f^*}(S) = I(y, x_i | x_S)$

SAGE LOSS FUNCTIONS



When the loss-optimal model f^* is inspected using *conditional-sampling* based SAGE value functions, interesting links exist.

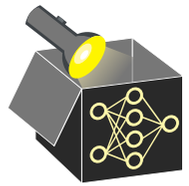
For cross-entropy loss:

- value function is the mutual information: $v_{f^*}(S) = I(y; x_S)$
- surplus contribution of a feature x_j is the conditional mutual information: $v_{f^*}(S \cup \{j\}) - v_{f^*}(S) = I(y, x_i | x_S)$

For MSE loss:

- value function is the expected reduction in variance given knowledge of the features x_S: $v_{f^*}(S) = Var(y) - \mathbb{E}[Var(y|x_S)]$
- surplus contribution is the respective reduction over x_S: $v_{f^*}(S \cup \{j\}) - v_{f^*}(S) = \mathbb{E}[Var(y|x_S)] - \mathbb{E}[Var(y|x_{S \cup j})]$

SAGE LOSS FUNCTIONS



When the loss-optimal model f^* is inspected using *conditional-sampling* based SAGE value functions, interesting links exist.

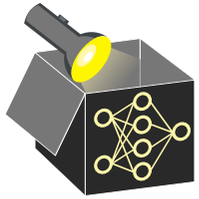
For cross-entropy loss:

- value function is the mutual information: $v_{f^*}(S) = I(y; x_S)$
- surplus contribution of a feature x_j is the conditional mutual information: $v_{f^*}(S \cup \{j\}) - v_{f^*}(S) = I(y, x_i | x_S)$

For MSE loss:

- value function is the expected reduction in variance given knowledge of the features x_S: $v_{f^*}(S) = Var(y) - \mathbb{E}[Var(y|x_S)]$
- surplus contribution is the respective reduction over x_S: $v_{f^*}(S \cup \{j\}) - v_{f^*}(S) = \mathbb{E}[Var(y|x_S)] - \mathbb{E}[Var(y|x_{S \cup j})]$

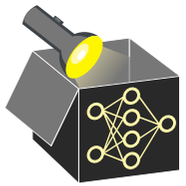
IMPLICATIONS MARGINAL SAGE VALUES



Can we gain insight into whether the ...

1. feature x_j is causal for the prediction?
   - for all coalitions S, $v(j \cup S) - v(S)$ can only be nonzero if $x_j \to \hat{f}(x)$ (as for PFI)
     $\rightsquigarrow \phi_j$ is only nonzero if x_j is causal for the prediction
   - $v(j \cup S) - v(S)$ may be zero due to independence $x_j \perp\!\!\!\perp y | x_S$ (as for PFI)
     $\rightsquigarrow \phi_j$ may be zero although the feature is causal for the prediction

IMPLICATIONS MARGINAL SAGE VALUES

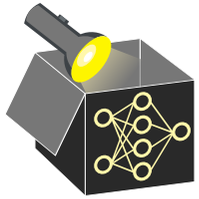


Can we gain insight into whether the ...

1. feature x_j is causal for the prediction?
   - for all coalitions S, $v(j \cup S) - v(S)$ can only be nonzero if $x_j \to \hat{f}(x)$ (as for PFI)
     $\rightsquigarrow \phi_j$ is only nonzero if x_j is causal for the prediction
   - $v(j \cup S) - v(S)$ may be zero due to indep. $x_j \perp\!\!\!\perp y | x_S$ (as for PFI)
     $\rightsquigarrow \phi_j$ may be zero although the feature is causal for the prediction

IMPLICATIONS MARGINAL SAGE VALUES

Can we gain insight into whether the ...

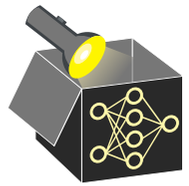


❷ feature x_j contains prediction-relevant information about y?

- value functions may be nonzero despite independence due to extrapolation (as for PFI)
  $\rightsquigarrow \phi_j$ may be nonzero without x_j being dependent with y
- value functions may be zero despite x_j containing prediction-relevant information due to underfitting (as for PFI)
  $\rightsquigarrow \phi_j$ may be zero although prediction-relevant information contained

IMPLICATIONS MARGINAL SAGE VALUES

Can we gain insight into whether the ...



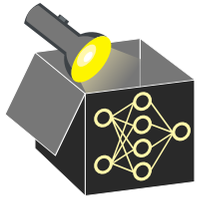
❷ feature x_j contains prediction-relevant information about y?

- value functions may be nonzero despite independence due to extrapolation (as for PFI)
  $\rightsquigarrow \phi_j$ may be nonzero without x_j being dependent with y
- value functions may be zero despite x_j containing prediction-relevant information due to underfitting (as for PFI)
  $\rightsquigarrow \phi_j$ may be zero although prediction-relevant information contained

Interpretable Machine Learning – 9 / 12

IMPLICATIONS MARGINAL SAGE VALUES

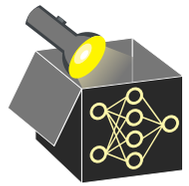
Can we gain insight into whether the ...



3. model requires access to x_j to achieve it's prediction performance?
   - like PFI, in general marginal value functions do not allow insight into unique contribution $\rightsquigarrow$ no insight from ϕ_j

IMPLICATIONS MARGINAL SAGE VALUES

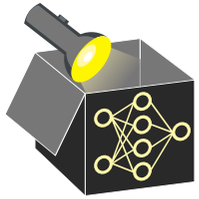
Can we gain insight into whether the ...



3. model requires access to x_j to achieve it's prediction performance?
   - like PFI, in general marginal value functions do not allow insight into unique contribution $\rightsquigarrow$ no insight from ϕ_j

Interpretable Machine Learning – 9 / 12

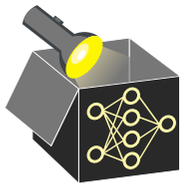
IMPLICATIONS CONDITIONAL SAGE VALUES



Can we gain insight into whether the ...

1. feature $\mathbf{x}_j$ is causal for the prediction?
   - value functions may be nonzero although feature is not directly used by $\hat{f}$
     $\rightsquigarrow$ nonzero ϕ_j does not imply $\mathbf{x}_j \to \hat{y}$
   - value functions may be zero although feature may be used by the model, e.g. if feature is independent with y and all other features
     $\rightsquigarrow$ zero ϕ_j does not imply $\mathbf{x}_j \not\to \hat{y}$

IMPLICATIONS CONDITIONAL SAGE VALUES

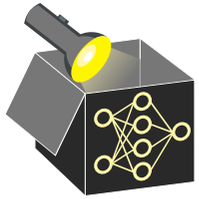


Can we gain insight into whether the ...

1. feature $\mathbf{x}_j$ is causal for the prediction?
   - value funcs may be nonzero although feature is not directly used by $\hat{f}$
     $\rightsquigarrow$ nonzero ϕ_j does not imply $\mathbf{x}_j \to \hat{y}$
   - value functions may be zero although feature may be used by the model, e.g. if feature is independent with y and all other features
     $\rightsquigarrow$ zero ϕ_j does not imply $\mathbf{x}_j \not\to \hat{y}$

IMPLICATIONS CONDITIONAL SAGE VALUES

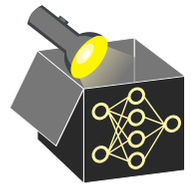
Can we gain insight into whether the ...



2. feature $\mathbf{x}_j$ contains prediction-relevant information about y?
   - e.g. for cross-entropy optimal $\hat{f}$, $v(j)$ measures mutual information $I(y; x_j)$
     $\rightsquigarrow$ prediction-relevance implies nonzero ϕ_j
   - $x_j \perp\!\!\!\perp y$ does not imply $x_j \perp\!\!\!\perp y|x_S$ and consequently does not imply $v(j \cup S) - v(S) = 0 \rightsquigarrow \phi_j$ may be nonzero although $\mathbf{x}_j \perp\!\!\!\perp y$

IMPLICATIONS CONDITIONAL SAGE VALUES

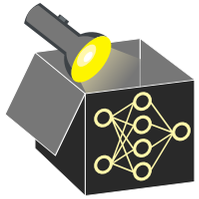
Can we gain insight into whether the ...



2. feature $\mathbf{x}_j$ contains prediction-relevant information about y?
   - e.g. for cross-entropy optimal $\hat{f}$, $v(j)$ measures mutual info. $I(y; x_j)$
     $\rightsquigarrow$ prediction-relevance implies nonzero ϕ_j
   - $x_j \perp\!\!\!\perp y$ does not imply $x_j \perp\!\!\!\perp y|x_S$ and consequently does not imply $v(j \cup S) - v(S) = 0 \rightsquigarrow \phi_j$ may be nonzero although $\mathbf{x}_j \perp\!\!\!\perp y$

IMPLICATIONS CONDITIONAL SAGE VALUES

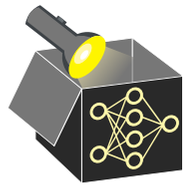
Can we gain insight into whether the ...



3. model requires access to x_j to achieve it's prediction performance?
   - e.g. for cross-entropy optimal $\hat{f}$, the surplus contribution $v(j \cup -j) - v(-j)$ captures the conditional mutual information $I(y; x_j | x_{-j})$
     $\rightsquigarrow \phi_j$ is nonzero for features with unique contribution
   - $x_j \perp\!\!\!\perp y | x_{-j}$ does not imply $x_j \perp\!\!\!\perp y | x_S$ (cond. w.r.t. to arbitrary coalitions S)
     $\rightsquigarrow \phi_j$ may be nonzero although the features has no unique contribution

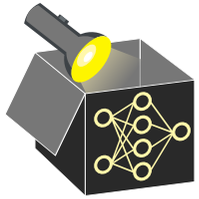
IMPLICATIONS CONDITIONAL SAGE VALUES

Can we gain insight into whether the ...



3. model requires access to x_j to achieve it's prediction performance?
   - e.g. for cross-entropy optimal $\hat{f}$, surplus contrib. $v(j \cup -j) - v(-j)$ captures the conditional mutual information $I(y; x_j | x_{-j})$
     $\rightsquigarrow \phi_j$ is nonzero for features with unique contribution
   - $x_j \perp\!\!\!\perp y | x_{-j}$ does not imply $x_j \perp\!\!\!\perp y | x_S$ (cond. w.r.t. to arbitrary coalitions S)
     $\rightsquigarrow \phi_j$ may be nonzero although feature has no unique contrib.

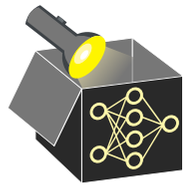
DEEP DIVE: SHAPLEY AXIOMS FOR SAGE



The Shapley axioms can be translated into properties of SAGE. The interpretation depends on whether conditional or marginal sampling is used.

Shapley property $\implies$	conditional SAGE property
efficiency	$\sum_{i=1}^{p} \phi_j(v) = \mathcal{R}(\hat{f}_\emptyset) - \mathcal{R}(\hat{f})$
symmetry	$x_j = x_i \implies \phi_i = \phi_j$
linearity	ϕ_j expecation of per-instance conditional SHAP applied to model loss
monotonicity	given models f, f', if $\forall S$: $v_f(S \cup j) - v_f(S) \geq v_{f'}(S \cup j) - v_{f'}(S)$ then $\phi_j(v_f) \geq \phi_j(v_{f'})$
dummy	if $\forall S : \hat{f}(x) \perp\!\!\!\perp x_j \vert x_S \Rightarrow \phi_j = 0$

DEEP DIVE: SHAPLEY AXIOMS FOR SAGE



The Shapley axioms can be translated into properties of SAGE. The interpretation depends on whether conditional or marginal sampling is used.

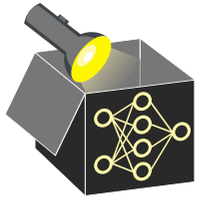
Shapley property $\implies$	conditional SAGE property
efficiency	$\sum_{i=1}^{p} \phi_j(v) = \mathcal{R}(\hat{f}_\emptyset) - \mathcal{R}(\hat{f})$
symmetry	$x_j = x_i \implies \phi_i = \phi_j$
linearity	ϕ_j expectation of per-instance conditional SHAP applied to model loss
monotonicity	given models f, f', if $\forall S$: $v_f(S \cup j) - v_f(S) \geq v_{f'}(S \cup j) - v_{f'}(S)$ then $\phi_j(v_f) \geq \phi_j(v_{f'})$
dummy	if $\forall S : \hat{f}(x) \perp\!\!\!\perp x_j \vert x_S \Rightarrow \phi_j = 0$

Interpretable Machine Learning – 11 / 12

DEEP DIVE: SHAPLEY AXIOMS FOR SAGE

The Shapley axioms can be translated into properties of SAGE. The interpretation depends on whether conditional or marginal sampling is used.

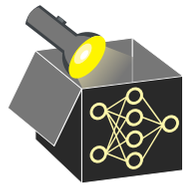
Shapley property $\Longrightarrow$	marginal SAGE property
efficiency	$\sum_{i=1}^{p} \phi_j(v) = \mathcal{R}(\hat{f}_\emptyset) - \mathcal{R}(\hat{f})$
symmetry	no intelligible implication
linearity	ϕ_j expecation of per-instance marginal SHAP applied to model loss
monotonicity	given models f, f', if $\forall S$: $v_f(S \cup j) - v_f(S) \geq v_{f'}(S \cup j) - v_{f'}(S)$ then $\phi_j(v_f) \geq \phi_j(v_{f'})$
dummy	model invariant to $x_j \Rightarrow \phi_j = 0$



DEEP DIVE: SHAPLEY AXIOMS FOR SAGE

The Shapley axioms can be translated into properties of SAGE. The interpretation depends on whether conditional or marginal sampling is used.

Shapley property $\Longrightarrow$	marginal SAGE property
efficiency	$\sum_{i=1}^{p} \phi_j(v) = \mathcal{R}(\hat{f}_\emptyset) - \mathcal{R}(\hat{f})$
symmetry	no intelligible implication
linearity	ϕ_j expecation of per-instance marginal SHAP applied to model loss
monotonicity	given models f, f', if $\forall S$: $v_f(S \cup j) - v_f(S) \geq v_{f'}(S \cup j) - v_{f'}(S)$ then $\phi_j(v_f) \geq \phi_j(v_{f'})$
dummy	model invariant to $x_j \Rightarrow \phi_j = 0$



Interpretable Machine Learning – 12 / 12

Interpretable Machine Learning

Leave One Covariate Out (LOCO)

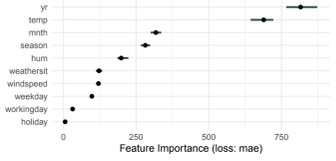
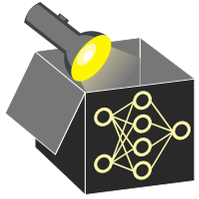


Figure: Bike Sharing Dataset

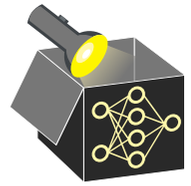
Learning goals

- Definition of LOCO
- Interpretation of LOCO



Interpretable Machine Learning

Feature Importances 1
Leave One Covariate Out (LOCO)



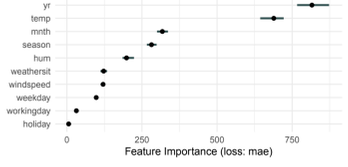


Figure: Bike Sharing Dataset

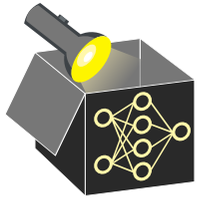
Learning goals

- Definition of LOCO
- Interpretation of LOCO

LOCO

Lei et al. (2018) Tibshirani (2018)

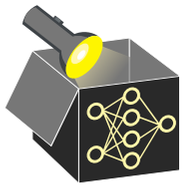
LOCO idea: Remove the feature from data, refit model on reduced data, and measure the loss in performance compared to model fitted on complete data.



LOCO

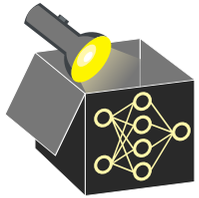
LEI_2018 TIBSHIRANI_2018

LOCO idea: Remove the feature from data, refit model on reduced data, and measure the loss in performance compared to model fitted on complete data.



LOCO

Lei et al. (2018) Tibshirani (2018)



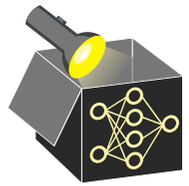
LOCO idea: Remove the feature from data, refit model on reduced data, and measure the loss in performance compared to model fitted on complete data.

Definition: Given train and test data $\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{test}} \subseteq \mathcal{D}$, a learner $\mathcal{I}$, and model $\hat{f} := \mathcal{I}(\mathcal{D}_{\text{train}})$, the LOCO importance for feature $j \in \{1, \ldots, p\}$ is computed by:

1. Learn model on $\mathcal{D}_{\text{train},-j}$ where feature x_j was removed, i.e. $\hat{f}_{-j} = \mathcal{I}(\mathcal{D}_{\text{train},-j})$

LOCO

LEI_2018 TIBSHIRANI_2018



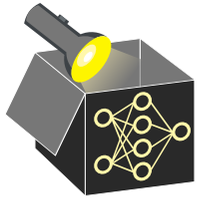
LOCO idea: Remove the feature from data, refit model on reduced data, and measure the loss in performance compared to model fitted on complete data.

Definition: Given train and test data $\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{test}} \subseteq \mathcal{D}$, a learner $\mathcal{I}$, and model $\hat{f} := \mathcal{I}(\mathcal{D}_{\text{train}})$, the LOCO importance for feat $j \in \{1, \ldots, p\}$ is computed by:

1. Learn model on $\mathcal{D}_{\text{train},-j}$ where feature x_j was removed, i.e. $\hat{f}_{-j} = \mathcal{I}(\mathcal{D}_{\text{train},-j})$

LOCO

Lei et al. (2018) Tibshirani (2018)



LOCO idea: Remove the feature from data, refit model on reduced data, and measure the loss in performance compared to model fitted on complete data.

Definition: Given train and test data $\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{test}} \subseteq \mathcal{D}$, a learner $\mathcal{I}$, and model $\hat{f} := \mathcal{I}(\mathcal{D}_{\text{train}})$, the LOCO importance for feature $j \in \{1, \ldots, p\}$ is computed by:

1. Learn model on $\mathcal{D}_{\text{train},-j}$ where feature x_j was removed, i.e. $\hat{f}_{-j} = \mathcal{I}(\mathcal{D}_{\text{train},-j})$
2. Compute the difference in local L_1 loss for each element in $\mathcal{D}_{\text{test}}$, i.e. $\Delta_j^{(i)} = \left| y^{(i)} - \hat{f}_{-j}(x_{-j}^{(i)}) \right| - \left| y^{(i)} - \hat{f}(x^{(i)}) \right|$ with $i \in \mathcal{D}_{\text{test}}$

LOCO

LEI_2018 TIBSHIRANI_2018

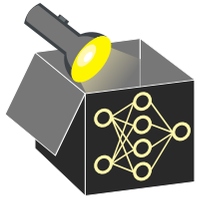
LOCO idea: Remove the feature from data, refit model on reduced data, and measure the loss in performance compared to model fitted on complete data.

Definition: Given train and test data $\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{test}} \subseteq \mathcal{D}$, a learner $\mathcal{I}$, and model $\hat{f} := \mathcal{I}(\mathcal{D}_{\text{train}})$, the LOCO importance for feat $j \in \{1, \ldots, p\}$ is computed by:

1. Learn model on $\mathcal{D}_{\text{train},-j}$ where feature x_j was removed, i.e. $\hat{f}_{-j} = \mathcal{I}(\mathcal{D}_{\text{train},-j})$
2. Compute the difference in local L_1 loss for each element in $\mathcal{D}_{\text{test}}$, i.e. $\Delta_j^{(i)} = \left| y^{(i)} - \hat{f}_{-j}(x_{-j}^{(i)}) \right| - \left| y^{(i)} - \hat{f}(x^{(i)}) \right|$ with $i \in \mathcal{D}_{\text{test}}$

LOCO

Lei et al. (2018) Tibshirani (2018)



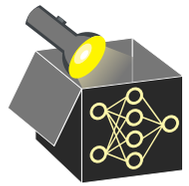
LOCO idea: Remove the feature from data, refit model on reduced data, and measure the loss in performance compared to model fitted on complete data.

Definition: Given train and test data $\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{test}} \subseteq \mathcal{D}$, a learner $\mathcal{I}$, and model $\hat{f} := \mathcal{I}(\mathcal{D}_{\text{train}})$, the LOCO importance for feature $j \in \{1, \ldots, p\}$ is computed by:

1. Learn model on $\mathcal{D}_{\text{train},-j}$ where feature x_j was removed, i.e. $\hat{f}_{-j} = \mathcal{I}(\mathcal{D}_{\text{train},-j})$
2. Compute the difference in local L_1 loss for each element in $\mathcal{D}_{\text{test}}$, i.e. $\Delta_j^{(i)} = \left|y^{(i)} - \hat{f}_{-j}(x_{-j}^{(i)})\right| - \left|y^{(i)} - \hat{f}(x^{(i)})\right|$ with $i \in \mathcal{D}_{\text{test}}$
3. Compute importance score by $\text{LOCO}_j = \text{med}\,(\Delta_j)$

LOCO

LEI_2018 TIBSHIRANI_2018



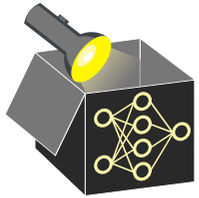
LOCO idea: Remove the feature from data, refit model on reduced data, and measure the loss in performance compared to model fitted on complete data.

Definition: Given train and test data $\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{test}} \subseteq \mathcal{D}$, a learner $\mathcal{I}$, and model $\hat{f} := \mathcal{I}(\mathcal{D}_{\text{train}})$, the LOCO importance for feat $j \in \{1, \ldots, p\}$ is computed by:

1. Learn model on $\mathcal{D}_{\text{train},-j}$ where feature x_j was removed, i.e. $\hat{f}_{-j} = \mathcal{I}(\mathcal{D}_{\text{train},-j})$
2. Compute the difference in local L_1 loss for each element in $\mathcal{D}_{\text{test}}$, i.e. $\Delta_j^{(i)} = \left|y^{(i)} - \hat{f}_{-j}(x_{-j}^{(i)})\right| - \left|y^{(i)} - \hat{f}(x^{(i)})\right|$ with $i \in \mathcal{D}_{\text{test}}$
3. Compute importance score by $\text{LOCO}_j = \text{med}\,(\Delta_j)$

LOCO

Lei et al. (2018) Tibshirani (2018)



LOCO idea: Remove the feature from data, refit model on reduced data, and measure the loss in performance compared to model fitted on complete data.

Definition: Given train and test data $\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{test}} \subseteq \mathcal{D}$, a learner $\mathcal{I}$, and model $\hat{f} := \mathcal{I}(\mathcal{D}_{\text{train}})$, the LOCO importance for feature $j \in \{1, \ldots, p\}$ is computed by:

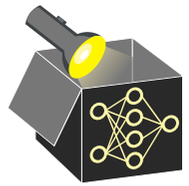
1. Learn model on $\mathcal{D}_{\text{train},-j}$ where feature x_j was removed, i.e. $\hat{f}_{-j} = \mathcal{I}(\mathcal{D}_{\text{train},-j})$
2. Compute the difference in local L_1 loss for each element in $\mathcal{D}_{\text{test}}$, i.e. $\Delta_j^{(i)} = \left|y^{(i)} - \hat{f}_{-j}(x_{-j}^{(i)})\right| - \left|y^{(i)} - \hat{f}(x^{(i)})\right|$ with $i \in \mathcal{D}_{\text{test}}$
3. Compute importance score by $\text{LOCO}_j = \text{med}\,(\Delta_j)$

The method can be generalized to other loss functions and aggregations. If we use mean instead of median we can rewrite LOCO as

$$\text{LOCO}_j = \mathcal{R}_{\text{emp}}(\hat{f}_{-j}) - \mathcal{R}_{\text{emp}}(\hat{f}).$$

LOCO

LEI_2018 TIBSHIRANI_2018



LOCO idea: Remove the feature from data, refit model on reduced data, and measure the loss in performance compared to model fitted on complete data.

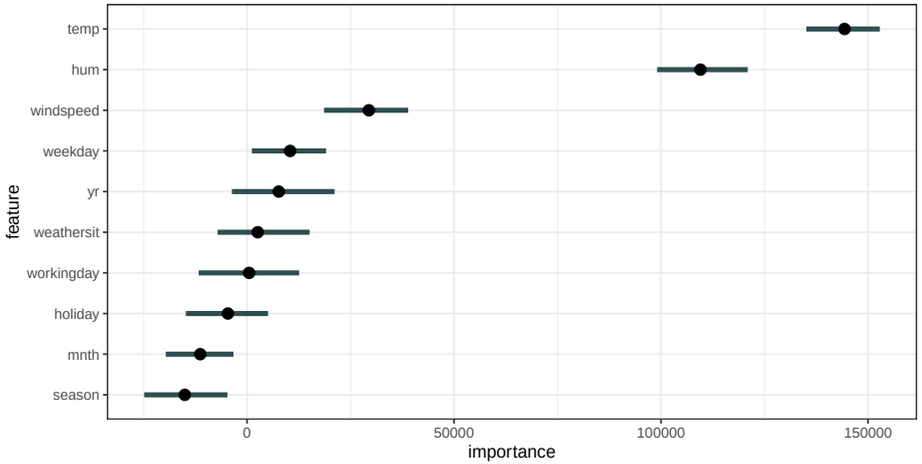
Definition: Given train and test data $\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{test}} \subseteq \mathcal{D}$, a learner $\mathcal{I}$, and model $\hat{f} := \mathcal{I}(\mathcal{D}_{\text{train}})$, the LOCO importance for feat $j \in \{1, \ldots, p\}$ is computed by:

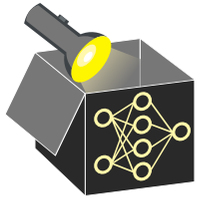
1. Learn model on $\mathcal{D}_{\text{train},-j}$ where feature x_j was removed, i.e. $\hat{f}_{-j} = \mathcal{I}(\mathcal{D}_{\text{train},-j})$
2. Compute the difference in local L_1 loss for each element in $\mathcal{D}_{\text{test}}$, i.e. $\Delta_j^{(i)} = \left|y^{(i)} - \hat{f}_{-j}(x_{-j}^{(i)})\right| - \left|y^{(i)} - \hat{f}(x^{(i)})\right|$ with $i \in \mathcal{D}_{\text{test}}$
3. Compute importance score by $\text{LOCO}_j = \text{med}\,(\Delta_j)$

The method can be generalized to other loss functions and aggregations. If we use mean instead of median we can rewrite LOCO as

$$\text{LOCO}_j = \mathcal{R}_{\text{emp}}(\hat{f}_{-j}) - \mathcal{R}_{\text{emp}}(\hat{f}).$$

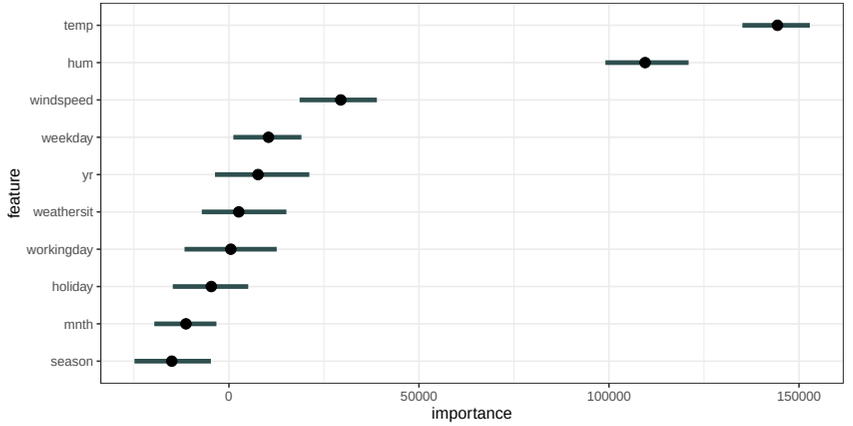
BIKE SHARING EXAMPLE

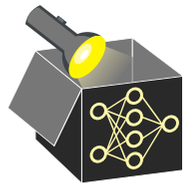




- Trained random forest (default hyperparameters) on 70% of bike sharing data
- Performance measure: mean squared error (MSE)
- Computed LOCO on test set for all features, measuring increase in MSE
- `temp` was most important: removing it increased MSE by approx. 140.000

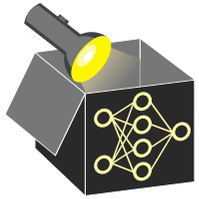
BIKE SHARING EXAMPLE





- Trained random forest (default hyperparams) on 70% of bike sharing data
- Performance measure: mean squared error (MSE)
- Computed LOCO on test set for all features, measuring increase in MSE
- `temp` was most important: removal increased MSE by approx. 140.000

INTERPRETATION OF LOCO

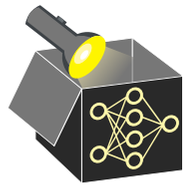


Interpretation: LOCO estimates the generalization error of the learner on a reduced dataset $\mathcal{D}_{-j}$.

Can we get insight into whether the ...

1. feature x_j is causal for the prediction $\hat{y}$?
   - In general, no also because we refit the model (counterexample next slide)
2. feature x_j contains prediction-relevant information?
   - In general, no (counterexample on the next slide)
3. model requires access to x_j to achieve its prediction performance?
   - Approximately, it provides insight into whether the *learner* requires access to x_j

INTERPRETATION OF LOCO



Interpretation: LOCO estimates the generalization error of the learner on a reduced dataset $\mathcal{D}_{-j}$.

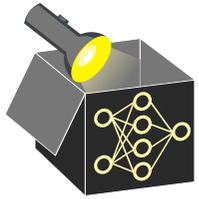
Can we get insight into whether the ...

1. feature x_j is causal for the prediction $\hat{y}$?
   - In general, no, also because we refit the model (counterexample on the next slide)
2. feature x_j contains prediction-relevant information?
   - In general, no (counterexample on the next slide)
3. model requires access to x_j to achieve its prediction performance?
   - Approximately, it provides insight into whether the *learner* requires access to x_j

INTERPRETATION OF LOCO

Example: Sample 1000 observations with

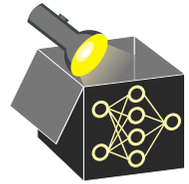
- $x_1, x_3 \sim N(0, 5)$, $x_2 = x_1 + \epsilon_2$ with $\epsilon_2 \sim N(0, 0.1)$
- $y = x_2 + x_3 + \epsilon$ with $\epsilon \sim N(0, 2)$
- Trained LM: $\hat{f}(x) = -0.02 - 1.02x_1 + 2.05x_2 + 0.98x_3$



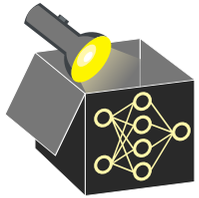
INTERPRETATION OF LOCO

Example: Sample 1000 observations with

- $x_1, x_3 \sim N(0, 5)$, $x_2 = x_1 + \epsilon_2$ with $\epsilon_2 \sim N(0, 0.1)$
- $y = x_2 + x_3 + \epsilon$ with $\epsilon \sim N(0, 2)$
- Trained LM: $\hat{f}(x) = -0.02 - 1.02x_1 + 2.05x_2 + 0.98x_3$

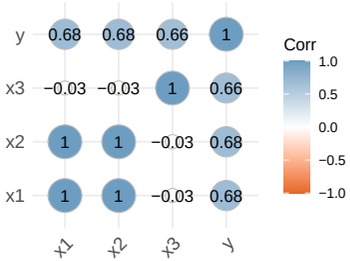


INTERPRETATION OF LOCO

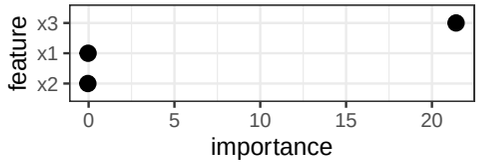


Example: Sample 1000 observations with

- $x_1, x_3 \sim N(0, 5)$, $x_2 = x_1 + \epsilon_2$ with $\epsilon_2 \sim N(0, 0.1)$
- $y = x_2 + x_3 + \epsilon$ with $\epsilon \sim N(0, 2)$
- Trained LM: $\hat{f}(x) = -0.02 - 1.02x_1 + 2.05x_2 + 0.98x_3$



Correlation matrix

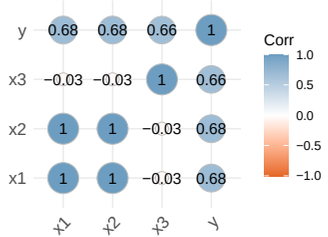


LOCO importance from LM trained on 70% of data, evaluated on remaining 30%

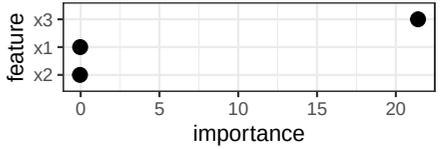
INTERPRETATION OF LOCO

Example: Sample 1000 observations with

- $x_1, x_3 \sim N(0, 5)$, $x_2 = x_1 + \epsilon_2$ with $\epsilon_2 \sim N(0, 0.1)$
- $y = x_2 + x_3 + \epsilon$ with $\epsilon \sim N(0, 2)$
- Trained LM: $\hat{f}(x) = -0.02 - 1.02x_1 + 2.05x_2 + 0.98x_3$

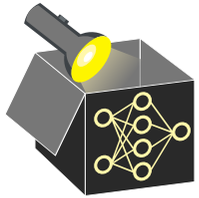


Correlation matrix



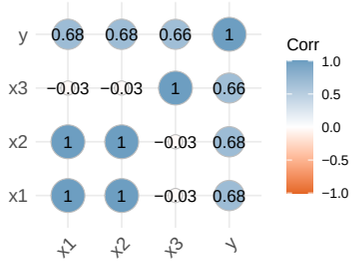
LOCO importance from LM trained on 70% of data, evaluated on remaining 30%

INTERPRETATION OF LOCO

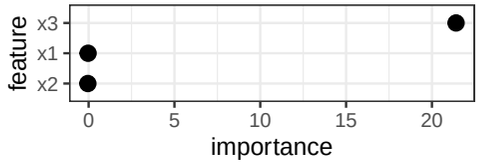


Example: Sample 1000 observations with

- $x_1, x_3 \sim N(0, 5)$, $x_2 = x_1 + \epsilon_2$ with $\epsilon_2 \sim N(0, 0.1)$
- $y = x_2 + x_3 + \epsilon$ with $\epsilon \sim N(0, 2)$
- Trained LM: $\hat{f}(x) = -0.02 - 1.02x_1 + 2.05x_2 + 0.98x_3$



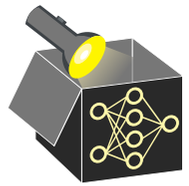
Correlation matrix



LOCO importance from LM trained on 70% of data, evaluated on remaining 30%

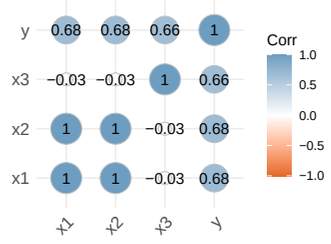
$\Rightarrow$ We cannot infer (1) from LOCO (e.g. $\text{LOCO}_2 \approx 0$ but coefficient of x_2 is 2.05)

INTERPRETATION OF LOCO

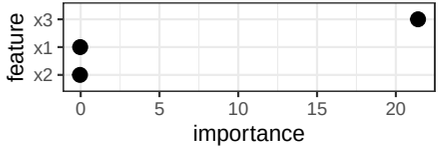


Example: Sample 1000 observations with

- $x_1, x_3 \sim N(0, 5)$, $x_2 = x_1 + \epsilon_2$ with $\epsilon_2 \sim N(0, 0.1)$
- $y = x_2 + x_3 + \epsilon$ with $\epsilon \sim N(0, 2)$
- Trained LM: $\hat{f}(x) = -0.02 - 1.02x_1 + 2.05x_2 + 0.98x_3$



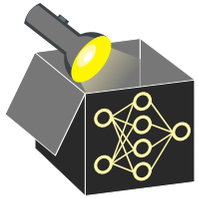
Correlation matrix



LOCO importance from LM trained on 70% of data, evaluated on remaining 30%

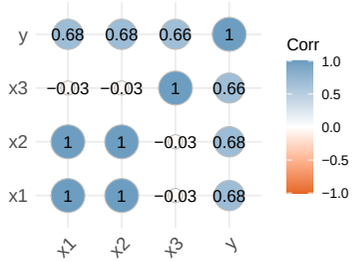
$\Rightarrow$ We cannot infer (1) from LOCO (e.g. $\text{LOCO}_2 \approx 0$ but coef. of x_2 is 2.05)

INTERPRETATION OF LOCO

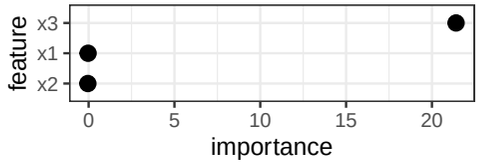


Example: Sample 1000 observations with

- $x_1, x_3 \sim N(0, 5)$, $x_2 = x_1 + \epsilon_2$ with $\epsilon_2 \sim N(0, 0.1)$
- $y = x_2 + x_3 + \epsilon$ with $\epsilon \sim N(0, 2)$
- Trained LM: $\hat{f}(x) = -0.02 - 1.02x_1 + 2.05x_2 + 0.98x_3$



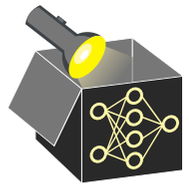
Correlation matrix



LOCO importance from LM trained on 70% of data, evaluated on remaining 30%

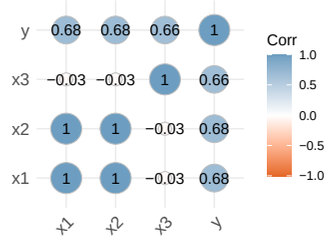
$\Rightarrow$ We cannot infer (1) from LOCO (e.g. $\text{LOCO}_2 \approx 0$ but coefficient of x_2 is 2.05)
$\Rightarrow$ We also can't infer (2), e.g., $Cor(x_2, y) = 0.68$ but $\text{LOCO}_2 \approx 0$

INTERPRETATION OF LOCO

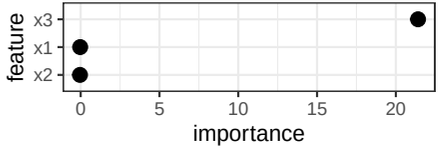


Example: Sample 1000 observations with

- $x_1, x_3 \sim N(0, 5)$, $x_2 = x_1 + \epsilon_2$ with $\epsilon_2 \sim N(0, 0.1)$
- $y = x_2 + x_3 + \epsilon$ with $\epsilon \sim N(0, 2)$
- Trained LM: $\hat{f}(x) = -0.02 - 1.02x_1 + 2.05x_2 + 0.98x_3$



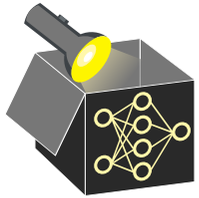
Correlation matrix



LOCO importance from LM trained on 70% of data, evaluated on remaining 30%

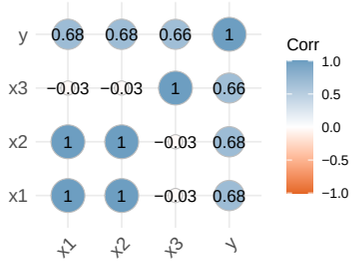
$\Rightarrow$ We cannot infer (1) from LOCO (e.g. $\text{LOCO}_2 \approx 0$ but coef. of x_2 is 2.05)
$\Rightarrow$ We also can't infer (2), e.g., $Cor(x_2, y) = 0.68$ but $\text{LOCO}_2 \approx 0$

INTERPRETATION OF LOCO

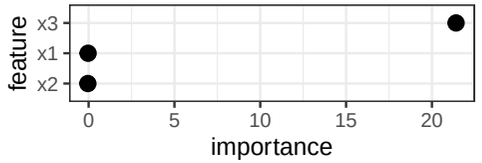


Example: Sample 1000 observations with

- $x_1, x_3 \sim N(0, 5)$, $x_2 = x_1 + \epsilon_2$ with $\epsilon_2 \sim N(0, 0.1)$
- $y = x_2 + x_3 + \epsilon$ with $\epsilon \sim N(0, 2)$
- Trained LM: $\hat{f}(x) = -0.02 - 1.02x_1 + 2.05x_2 + 0.98x_3$



Correlation matrix



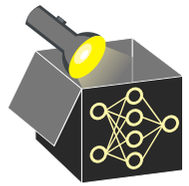
LOCO importance from LM trained on 70% of data, evaluated on remaining 30%

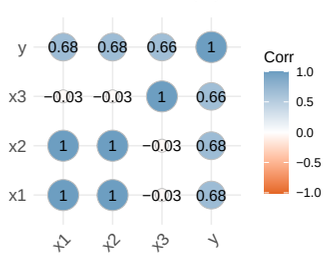
⇒ We cannot infer (1) from LOCO (e.g. $\text{LOCO}_2 \approx 0$ but coefficient of x_2 is 2.05)
⇒ We also can't infer (2), e.g., $Cor(x_2, y) = 0.68$ but $\text{LOCO}_2 \approx 0$
⇒ We can get insight into (3): x_2 and x_1 highly correlated with $\text{LOCO}_1 = \text{LOCO}_2 \approx 0$
$\rightsquigarrow$ x_2 and x_1 take each others place if one of them is left out (not the case for x_3)

INTERPRETATION OF LOCO

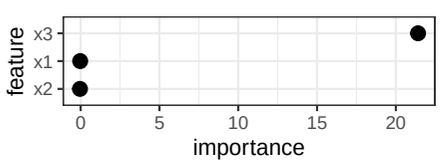
Example: Sample 1000 observations with

- $x_1, x_3 \sim N(0, 5)$, $x_2 = x_1 + \epsilon_2$ with $\epsilon_2 \sim N(0, 0.1)$
- $y = x_2 + x_3 + \epsilon$ with $\epsilon \sim N(0, 2)$
- Trained LM: $\hat{f}(x) = -0.02 - 1.02x_1 + 2.05x_2 + 0.98x_3$





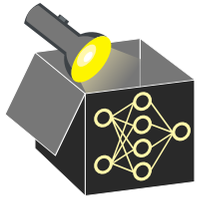
Correlation matrix



LOCO importance from LM trained on 70% of data, evaluated on remaining 30%

⇒ We cannot infer (1) from LOCO (e.g. $\text{LOCO}_2 \approx 0$ but coef. of x_2 is 2.05)
⇒ We also can't infer (2), e.g., $Cor(x_2, y) = 0.68$ but $\text{LOCO}_2 \approx 0$
⇒ We can get insight into (3): x_2, x_1 highly corr. with $\text{LOCO}_1 = \text{LOCO}_2 \approx 0$
$\rightsquigarrow$ x_2 and x_1 take each others place if one of them is left out (unlike x_3)

PROS AND CONS



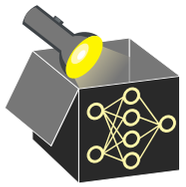
Pros:

- Requires (only?) one refitting step per feature for evaluation
- Easy to implement
- Testing framework available in ▸ Lei et al. (2018)

Cons:

- Provides insight into a learner on specific data, not a specific model
  + for algorithm-level insight
  − for model-specific insights
- Model training is a random process and LOCO estimates can be noisy
  ⇝ Limits inference about on model and data, or multiple refittings necessary?
- Requires re-fitting the learner for each feature
  ⇝ Computationally intensive compared to PFI

PROS AND CONS



Pros:

- Requires (only?) one refitting step per feature for evaluation
- Easy to implement
- Testing framework available in ▸ Lei 2018

Cons:

- Provides insight into a learner on specific data, not a specific model
  + for algorithm-level insight
  − for model-specific insights
- Model training is a random process and LOCO estimates can be noisy
  ⇝ Limits inference on model and data, or multiple refittings necessary?
- Requires re-fitting the learner for each feature
  ⇝ Computationally intensive compared to PFI