Shapley values

CSEP 590B: Explainable AI

Ian Covert & Su-In Lee
University of Washington

Course announcements

- HW0 grades posted today
  - Solutions are on Canvas
- HW1 covers content from last week and this week
  - **From last week:** permutation tests, removal-based explanations
  - **From this week:** Shapley values (properties, estimation)

Shapley values

- An old idea from game theory (1953), unrelated to AI/ML
- Now the basis of a popular XAI tool, SHAP
- Will also come up later in the course

Today

- Section 1
  - Cooperative game theory background
  - The Shapley value
  - Shapley values in XAI
- Section 2
  - Challenge #1: feature removal
  - Challenge #2: estimation
  - SHAP examples

Cooperative game theory

- Probably not the part of game theory you've heard of
  - For example, Nash equilibrium is from non-cooperative game theory
- Here, we focus on games where coalitions of players form to achieve different profits

Cooperative game notation

- Set of *players* $D = \{1, \ldots, d\}$
- A *game* is given by specifying a value for every coalition $S \subseteq D$
- Mathematically represented by a *characteristic function*:

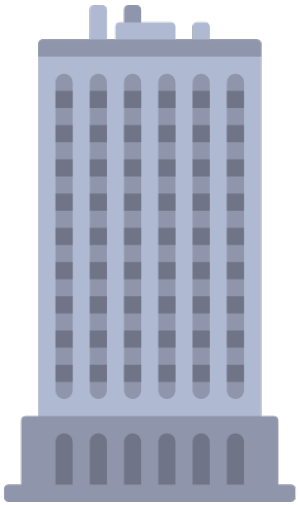
$$v: 2^D \mapsto \mathbb{R}$$

- Grand coalition value $v(D)$, null coalition $v(\emptyset)$, arbitrary coalition $v(S)$

Company example

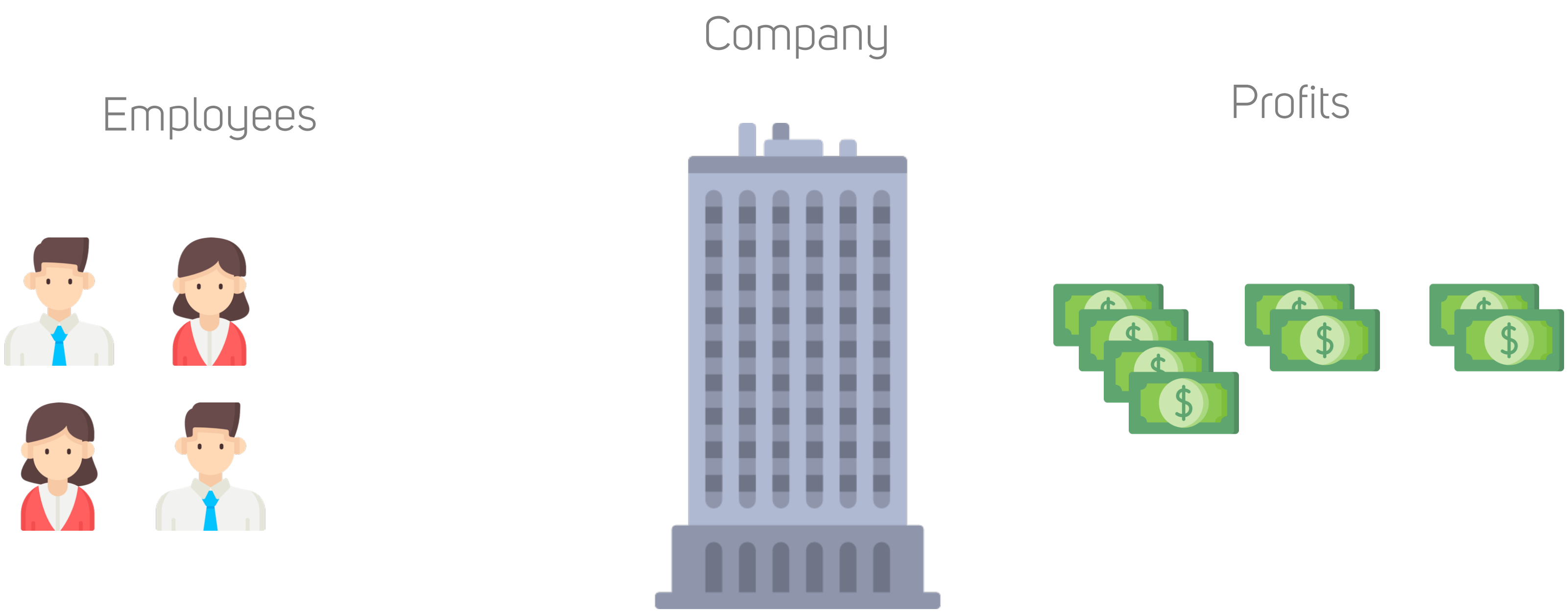
Employees

Company

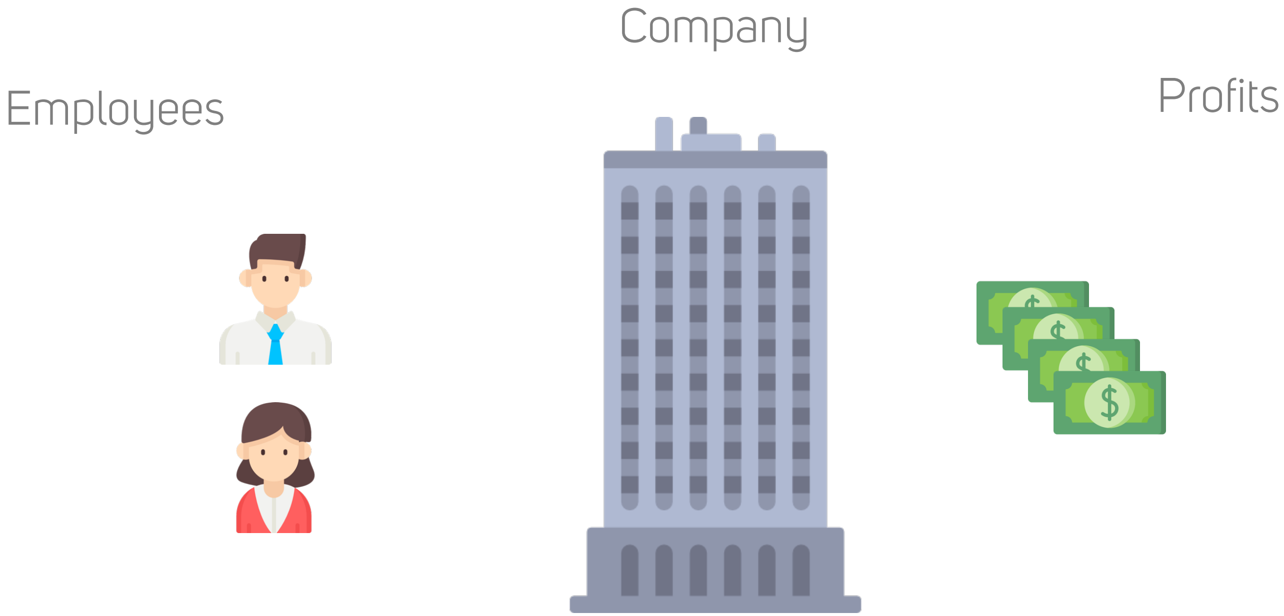


Profits

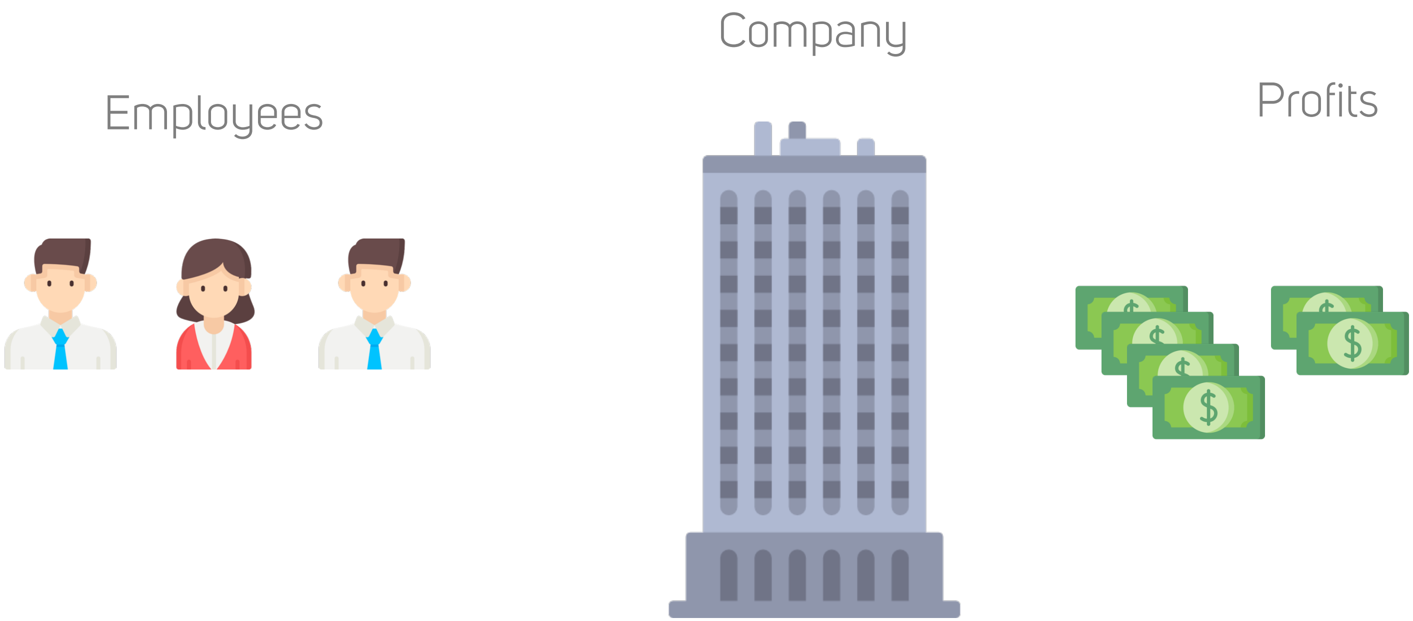
Company example



Company example



Company example

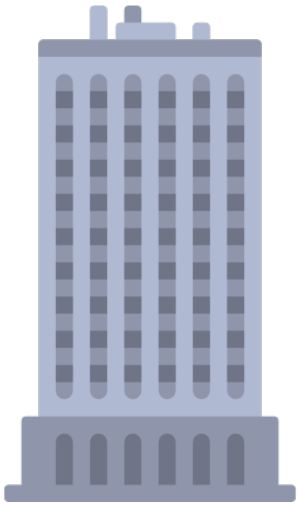


Company example

Company

Employees

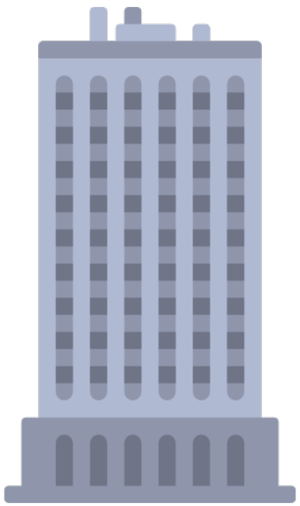
Profits



Company example

Employees

Company



Profits

Players $S \subseteq D$

Value $v(S)$

Key game theory questions

- Which players will participate vs. break off on their own?
- How to allocate credit among players?

Shapley value

- A technique for allocating credit to players in a cooperative game
- Famously derived from a set of *fairness axioms*

Today

- Section 1
  - Cooperative game theory background
  - The Shapley value
  - Shapley values in XAI
- Section 2
  - Challenge #1: feature removal
  - Challenge #2: estimation
  - SHAP examples

Lloyd Shapley

- Won 2012 Nobel Memorial Prize in economics

Shapley value setup

- Let G denote the set of games on d players
- The Shapley value assigns a vector of credits to each game (in $\mathbb{R}^d$, one credit per player)
- Mathematically, a function of the form

$$\phi: G \mapsto \mathbb{R}^d$$

- For a game v, Shapley values are $\phi_1(v), \ldots, \phi_d(v)$

Fairness axioms

Consider a game v and credit allocations $\phi(v) = [\phi_1(v), \ldots, \phi_d(v)]$. We want to satisfy the following properties:

- *(Efficiency)* The credits sum to the grand coalition's value, or $\sum_{i \in D} \phi_i(v) = v(D) - v(\emptyset)$
- *(Symmetry)* If two players (i, j) are interchangeable, or $v(S \cup \{i\}) = v(S \cup \{j\})$ for all $S \subseteq D$, then $\phi_i(v) = \phi_j(v)$
- *(Null player)* If a player contributes no value, or $v(S \cup \{i\}) = v(S)$ for all $S \subseteq D$, then $\phi_i(v) = 0$
- *(Linearity)* The credits for linear combinations of games behave linearly, or $\phi(c_1 v_1 + c_2 v_2) = c_1 \phi(v_1) + c_2 \phi(v_2)$, where $c_1, c_2 \in \mathbb{R}$

Lloyd Shapley, "A value for n-person games" (1953)

Axiomatic uniqueness

- The Shapley value (SV) is the only function $\phi: G \mapsto \mathbb{R}^d$ to satisfy these properties
- Given by the following equation:

$$\phi_i(v) = \sum_{S \subseteq D \setminus \mathrm{i}} \frac{|S|!\,(d - 1 - |S|)!}{d!} [v(S \cup \{i\}) - v(S)]$$

Weighted average across $S \subseteq D \setminus i$

Contribution from adding player i

Interpretation

- Intuitive meaning in terms of player orderings
  - Given an ordering π, each player contributes when added to the preceding ones
  - SV is the average contribution across all orderings

$$\phi_i(v) = \frac{1}{d!} \sum_{\pi \in \Pi} [v(\{ j \mid \pi^{-1}(j) \leq \pi^{-1}(i) \}) - v(\{ j \mid \pi^{-1}(j) < \pi^{-1}(i) \})]$$

Average across all orderings

Players up to and including i

Players preceding i

Today

- Section 1
  - Cooperative game theory background
  - The Shapley value
  - Shapley values in XAI
- Section 2
  - Challenge #1: feature removal
  - Challenge #2: estimation
  - SHAP examples

Application to ML

- Consider **features** as **players**
- Consider **model behavior** as **profit**
  - E.g., the prediction, the loss, etc.
- Then, use Shapley values to quantify each feature's impact

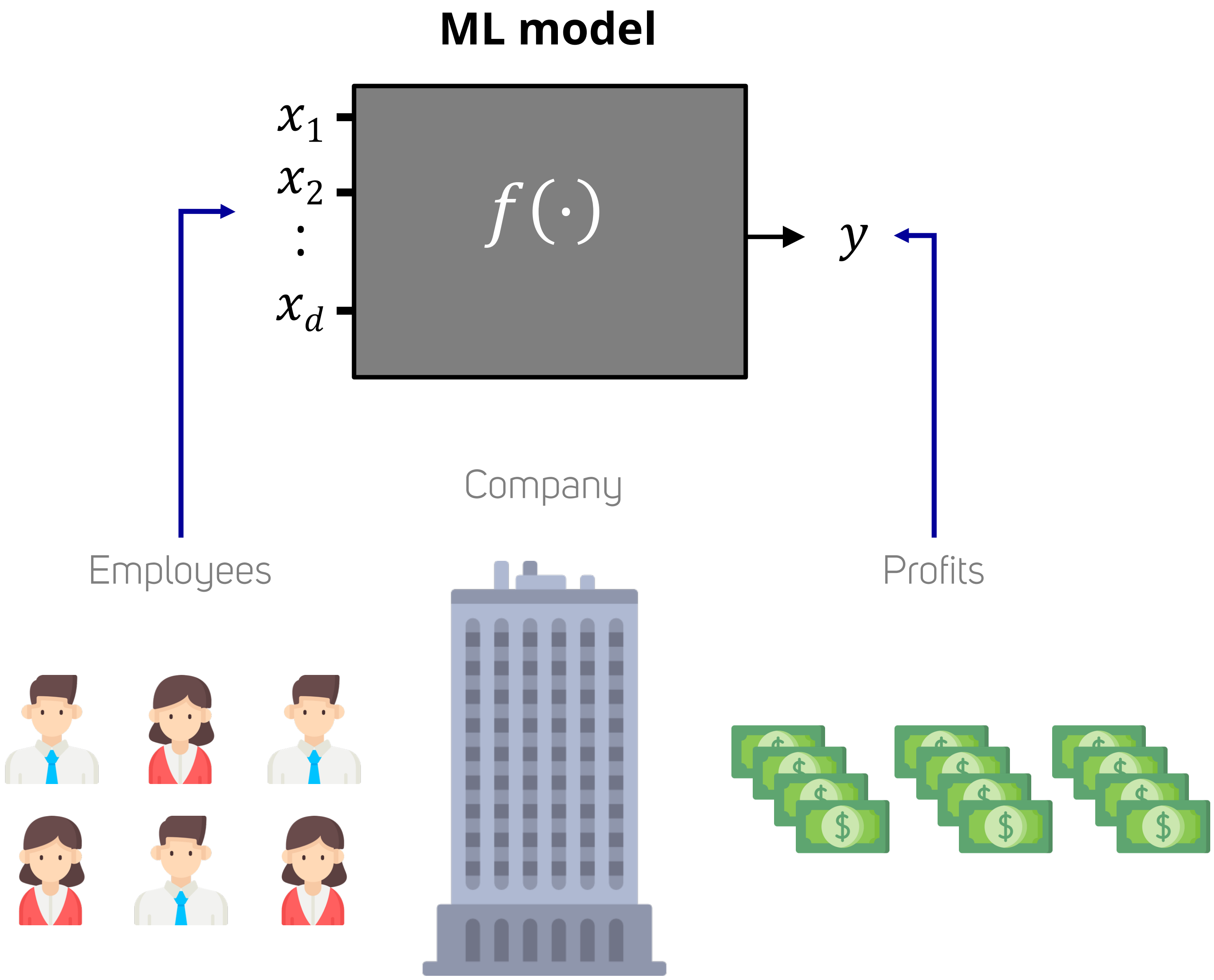
SHAP

- SHAP = *SHapley Additive exPlanations*

- Popularized use of Shapley values in ML
  - Also used in earlier work by Lipovetsky & Conklin (2001), Strumbelj et al. (2009), Datta et al. (2016)

- SHAP uses Shapley values to explain individual predictions

Lundberg & Lee, “A unified approach to interpreting model predictions” (2017)


ML model
x_1
x_2
x_d
$f(\cdot)$
y
Company
Employees
Profits

SHAP as a removal-based explanation

Recall the three choices for removal-based explanations:

1. **Feature removal:** $F(x_S) = \mathbb{E}_{x_{\bar{S}}|x_S}[f(x_S, x_{\bar{S}})]$
2. **Model behavior:** $v(S) = F_y(x_S)$
3. **Summary:** $a_i = \phi_i(v)$

Consider this more closely in the next slide

Shapley value

Notation clarification

- What is $\mathbb{E}_{x_{\bar{S}}|x_S}[f(x_S, x_{\bar{S}})]$?
- The expected value of the model output when conditioned on the feature values x_S

$$
\begin{aligned}
F(x_S) &= \mathbb{E}_{x_{\bar{S}}|x_S}[f(x_S, x_{\bar{S}})] \\
&= \mathbb{E}[f(x_S, x_{\bar{S}}) \mid x_s] \\
&= \sum_{x_{\bar{S}}} f(x_S, x_{\bar{S}}) \cdot \mathrm{p}(x_{\bar{S}} \mid x_S)
\end{aligned}
$$

Summation over all possible $x_{\bar{S}}$ values

Model output given $x_{\bar{S}}$

Probability of $x_{\bar{S}}$ conditioned on x_S

Notation clarification (cont.)

- Recall Bayes rule for conditional probability:

$$\mathrm{p}(\, x_{\bar{S}} \mid x_S \,) = \frac{\mathrm{p}(x_S\,, x_{\bar{S}})}{\mathrm{p}(x_S)}$$

Probability of $x_{\bar{S}}$ and x_S occurring together

Probability of x_S occurring on its own

Notation clarification (cont.)

- **Intuition:** in SHAP, we want to evaluate the model given a subset of features as follows
  - Fix the example to be explained x and the set of available features x_S
  - Withhold the remaining feature values $x_{\bar{S}}$
  - To do so, consider *all possible values* for $x_{\bar{S}}$, and make the corresponding predictions $f(x_S, x_{\bar{S}})$
  - Then average these predictions, weighting them according to the conditional probability $\mathrm{p}(x_{\bar{S}} \mid x_S)$

SHAP summary

- SHAP analyzes individual predictions by setting up the following cooperative game:

$$v(S) = F_y(x_S) = \mathbb{E}_{x_{\bar{S}}|x_S}[f(x_S, x_{\bar{S}})]$$

- Then determines feature attributions using the Shapley value:

$$a_i = \phi_i(v)$$

Other Shapley value-based methods

- **Shapley Net Effects:** Owen, "Sobol' indices and Shapley value" (2014)
- **QII:** Datta et al., "Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems" (2016)
- **IME:** Strumbelj & Kononenko, "Explaining instance classifications with interactions of subsets of feature values" (2009)
- **SAGE:** Covert et al., "Understanding global feature contributions with additive importance measures" (2020)
- **Causal Shapley values:** Heskes et al., "Causal Shapley values: Exploiting causal knowledge to explain individual predictions of complex models" (2020)
- **ASV:** Frye et al., "Asymmetric Shapley values: incorporating causal knowledge into model-agnostic explainability" (2020)
- **SP-VIM:** Williamson & Feng, "Efficient nonparametric statistical inference on population feature importance using Shapley values" (2020)

Today

- Section 1
  - Cooperative game theory background
  - The Shapley value
  - Shapley values in XAI
  - **10 min break**
- Section 2
  - Challenge #1: feature removal
  - Challenge #2: estimation
  - SHAP examples

Shapley values (continued)

CSEP 590B: Explainable AI

Ian Covert & Su-In Lee
University of Washington

SHAP challenges

I. Removing features properly

- Previewed last time (the first choice for removal-based explanations)

II. Calculating Shapley values

- A problem unique to Shapley values: exponential computational complexity

Today

- Section 1
  - Cooperative game theory background
  - The Shapley value
  - Shapley values in XAI
- Section 2
  - Challenge #1: feature removal
  - Challenge #2: estimation
  - SHAP examples

Original formulation

- Marginalize out features using their **conditional distribution**

$$F(x_S) = \mathbb{E}_{x_{\bar{S}}|x_S}[f(x_S, x_{\bar{S}})]$$

Condition on available features

Model output

Practical alternative

- The conditional distribution is hard to estimate
- Instead, we can marginalize out features using their **marginal distribution**

$$\mathbb{E}_{x_{\bar{S}} | x_S}[f(x_S, x_{\bar{S}})] \approx \mathbb{E}_{x_{\bar{S}}}[f(x_S, x_{\bar{S}})]$$

Drop conditioning

Remark

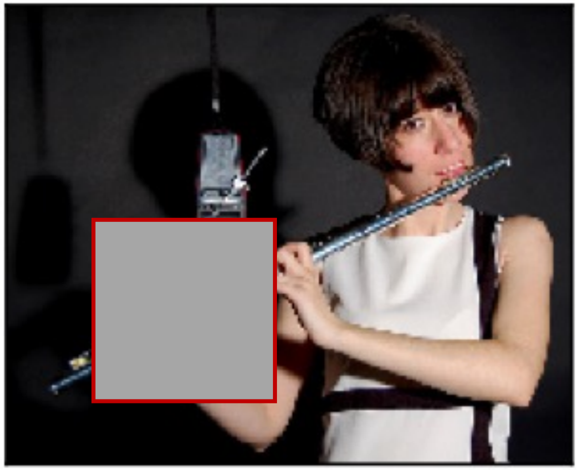
- In general, the conditional and marginal distributions are not equal

$$\mathrm{p}(x_{\bar{S}} \mid x_S) \neq \mathrm{p}(x_{\bar{S}})$$

- Assuming they're identical = assuming feature independence
- Can result in unlikely, *off-manifold* feature combinations

Off-manifold examples

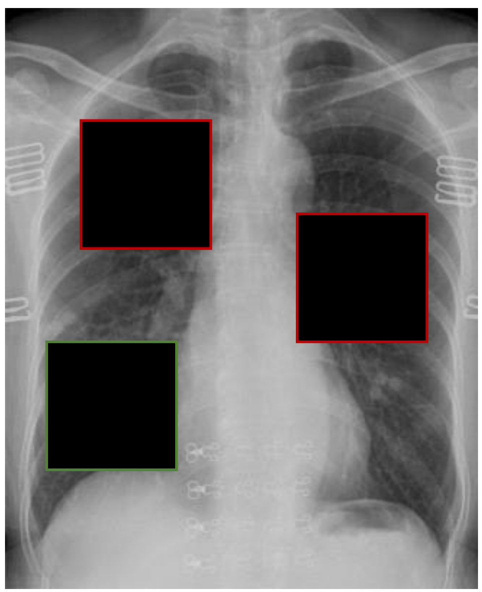
- **Tabular data:** male + housewife
- **Images:** implausible inpainting



- **Problem:** undefined model behavior

Remark

- Marginalizing out with conditional distribution may better represent human reasoning
- **Intuition:** given available information, what are plausible values for missing features?



Should recognize missing info and make best-effort prediction given available information

Subsequent debate

- Recent work has debated the “right” approach
- Some in favor of marginal distribution
  - Janzing et al., “Feature relevance quantification in explainable AI: A causality problem” (2019)
- Others in favor of conditional distribution
  - Aas et al., “Explaining individual predictions when features are dependent: More accurate approximations to Shapley values” (2019)
  - Frye et al., “Shapley-based explainability on the data manifold” (2020)
  - Covert et al., “Explaining by removing: a unified framework for model explanation” (2020)
- Subtle topic, depends on use-case and aims

Practical concern

- Can we implement these approaches for removing features?

Marginal distribution

- Easy to implement with Monte Carlo estimation
- Choose m datapoints $x^1, \ldots, x^m$ from dataset
- Approximate as follows:

$$\mathbb{E}_{x_{\bar{S}}}[f(x_S, x_{\bar{S}})] = \sum_{x_{\bar{S}}} \mathrm{p}(x_{\bar{S}}) f(x_S, x_{\bar{S}}) \approx \frac{1}{m} \sum_{i=1}^{m} f(x_S, x_{\bar{S}}^i)$$

Remark: permutation tests do this, but using a single sample

Conditional distribution

- Assume we can sample from $\mathrm{p}(x_{\bar{S}} \mid x_S)$
- Fix x_S, take m samples $x_{\bar{S}}^i \sim \mathrm{p}(x_{\bar{S}} \mid x_S)$, then approximate as follows:

$$\mathbb{E}_{x_{\bar{S}} | x_S}[f(x_S, x_{\bar{S}})] = \sum_{x_{\bar{S}}} \mathrm{p}(x_{\bar{S}} \mid x_S) f(x_S, x_{\bar{S}}) \approx \frac{1}{m} \sum_{i=1}^{m} f(x_S, x_{\bar{S}}^i)$$

- **Problem:** we rarely have access to the conditional distribution $\mathrm{p}(x_{\bar{S}} \mid x_S)$

Conditional distribution approximations

- Several options available
  - Make parametric assumptions about joint distribution $\mathrm{p}(x)$ (e.g., multivariate Gaussian)
  - Train a conditional generative model $\hat{\mathrm{p}}(x_{\bar{S}} \mid x_S)$
  - Train "supervised surrogate" model (Frye et al.)
  - Use a model that accommodates missing features

- Non-trivial to implement, can't guarantee perfect approximation

Frye et al., "Shapley explainability on the data manifold" (2020)

Implications for other methods

- This challenge is not unique to Shapley value-based methods
- Recall: all removal-based explanations require a feature removal approach
  - Because of its popularity, SHAP has received the most attention
  - Other methods face the same choice, and none have a perfect approach (see Covert et al.)

Covert et al., "Explaining by removing: a unified framework for model explanation" (2021)

Today

- Section 1
  - Cooperative game theory background
  - The Shapley value
  - Shapley values in XAI
- Section 2
  - Challenge #1: feature removal
  - Challenge #2: estimation
  - SHAP examples

Setup

- Assume we have a game $v: 2^D \mapsto \mathbb{R}$
- We want to calculate Shapley values
- How straightforward is this?

Computational complexity

- The equation for Shapley values is:

$$\phi_i(v) = \sum_{S \subseteq D \setminus \mathrm{i}} \frac{|S|!\,(d-1-|S|)!}{d!}[v(S \cup i) - v(S)]$$

Summation across 2^{d-1} subsets

- Exponential running time $\mathcal{O}(2^d)$
- Intractable for even moderate d (e.g., $d > 20$)

What can we do?

- We cannot calculate Shapley values exactly when d is large
- Instead, we can approximate them
- We'll discuss the following approaches:
  - Permutation-based estimation
  - Regression-based estimation
  - Others (briefly)

Permutation view

- Recall the Shapley value's ordering interpretation
- The value $\phi_i(v)$ is player i's average contribution across all player orderings

A B C

1. Enumerate all orderings
2. Find player contribution
3. Average

$\boxed{A}$ B C

$\boxed{A}$ C B

~~B~~ $\boxed{A}$ C

~~B~~ ~~C~~ $\boxed{A}$

~~C~~ $\boxed{A}$ B

~~C~~ ~~B~~ $\boxed{A}$

Mean = $\phi_A(v)$

Permutation-based estimation

- **Problem:** $d!$ orderings is too many for large values of d
- **Idea:** sample a moderate number of orderings
  - Calculate average contributions across those orderings

Permutation-based estimation (cont.)

Algorithm 1: Permutation estimation

Input: Game v, iterations $m > 0$

Output: Shapley value estimates $\hat{\phi}_1(v), \dots, \hat{\phi}_d(v)$

initialize $\hat{\phi}_i(v) = 0$ for $i = 1, \dots, d$

for $j = 1$ **to** m **do**

  sample permutation $\pi \in \Pi$ uniformly at random

  $S = \varnothing$

  $\texttt{prev} = v(\varnothing)$

  for $k = 1$ **to** d **do**

    $i = \pi(k)$ `// Get next player in ordering`

    $S = S \cup \{i\}$

    $\texttt{curr} = v(S)$

    $\hat{\phi}_i(v) = \hat{\phi}_i(v) + \Big(\texttt{curr} - \texttt{prev}\Big)$ `// Update estimate`

    $\texttt{prev} = \texttt{curr}$

  end

end

set $\hat{\phi}_i(v) = \frac{\hat{\phi}_i(v)}{m}$ for $i = 1, \dots, d$ `// Normalize`

return $\hat{\phi}_1(v), \dots, \hat{\phi}_d(v)$

Regression view

- An alternative Shapley value characterization
- Perhaps surprisingly, SVs are the solution to a weighted least squares problem

Regression view (cont.)

- Consider a game $v: 2^D \mapsto \mathbb{R}$
- Consider a weighting function $\mu(S)$:

$$\mu(S) = \frac{d-1}{\binom{d}{|S|} |S|(d-|S|)}$$

- Shapley values minimize the following objective:

$$\min_{\beta_0,\ldots\beta_d} \sum_{S \subseteq D} \mu(S) \left(\beta_0 + \sum_{i \in S} \beta_i - v(S) \right)^2$$

Squared error

Weighted summation

Additive approximation

Regression-based estimation

- **Problem:** WLS problems are easy to solve, but 2^d terms is too many

- **Idea:** approximate WLS problem by sampling subsets according to $\mu(S)$
  - Incorporate weights $\mu(\emptyset) = \mu(D) = \infty$ as constraints, $\beta_0 = v(\emptyset)$ and $\sum_{i \in D} \beta_i = v(D) - v(\emptyset)$
  - Solve the constrained least squares problem

Regression-based estimation (cont.)

- Omitting a detailed algorithm here
  - Constraints make things a bit complicated
  - Method known as **KernelSHAP**, introduced by Lundberg & Lee (2017)
  - See paper below for relatively simple exposition

Covert & Lee, "Improving KernelSHAP: Practical Shapley value estimation via linear regression" (2021)

Connection with LIME

- Surprising link between SHAP and LIME
  - Recall: LIME calculates attributions by fitting an additive proxy model
  - Requires weighting function $\pi(S)$ and regularizer Ω (see lecture 2 slides)

- Shapley values are equivalent to LIME with $\pi(S) = \mu(S)$ and $\Omega = 0$
  - SHAP is a special case of LIME, suggests a principled way to choose π and Ω

Lundberg & Lee, "A unified approach to interpreting model predictions" (2017)

Alternative approaches

- Permutation- and regression-based estimators are solid
  - Consistent, asymptotically unbiased, agnostic to game/model
  - Considerably faster than brute-force calculation

- However, still somewhat slow: they require many model evaluations

Deep learning estimation

- FastSHAP: estimate Shapley values with a learned explainer model
  - Train a separate deep learning model to generate explanations
  - Single forward pass = very fast
  - Must invest time in training for fast explanations

Jethani et al., “FastSHAP: Real-time Shapley value estimation” (2021)

Model-specific estimation

- Decision trees
  - ★ **TreeSHAP:** Lundberg et al., "Explainable AI for trees: from local explanations to global understanding" (2019)
  - **SHAFF:** Bénard et al., "SHAFF: Fast and consistent Shapley effects estimates via random forests" (2021)
- Neural networks
  - **DeepSHAP:** Lundberg & Lee, "A unfied approach to interpreting model predictions" (2017)
  - **DASP:** Ancona et al., "Explaining deep neural networks with a polynomial time algorithm for Shapley value estimation" (2019)
- Custom models
  - **SHAPNets:** Wang et al., "Shapley explanation networks" (2021)

More papers on Shapley value estimation

- Castro et al., "Improving polynomial estimation of the Shapley value by stratified random sampling with optimum allocation" (2017)
- Chen et al., "L-Shapley and C-Shapley: Efficient model interpretation for structured data" (2018)
- Simon & Thouvenot, "A projected stochastic gradient algorithm for estimating Shapley value applied in attribute importance" (2020)
- Covert & Lee, "Improving KernelSHAP: Practical Shapley value estimation via linear regression" (2021)
- Van den Broeck et al., "On the tractability of SHAP explanations" (2021)
- Mitchell et al., "Sampling permutations for Shapley value estimation" (2021)
- Chen et al., "Algorithms to estimate Shapley value feature attributions" (2022)

Today

- Section 1
  - Cooperative game theory background
  - The Shapley value
  - Shapley values in XAI
- Section 2
  - Challenge #1: feature removal
  - Challenge #2: estimation
  - SHAP examples

Setup

- First, focus on Boston housing dataset
- Predict median house price in a neighborhood using 14 features
  - E.g., mean number of rooms, crime rate, distance to employment centers
  - Trained an XGBoost model (gradient boosted decision tree)

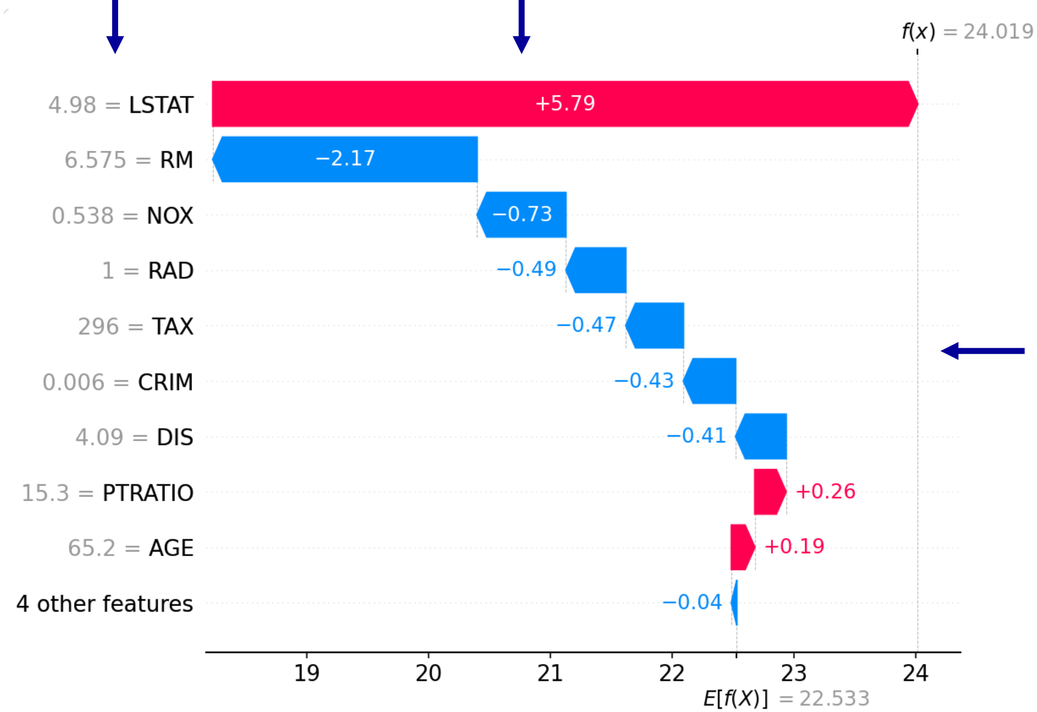
Local explanations

Actual prediction

Feature names/values

Shapley values

Directionality matters

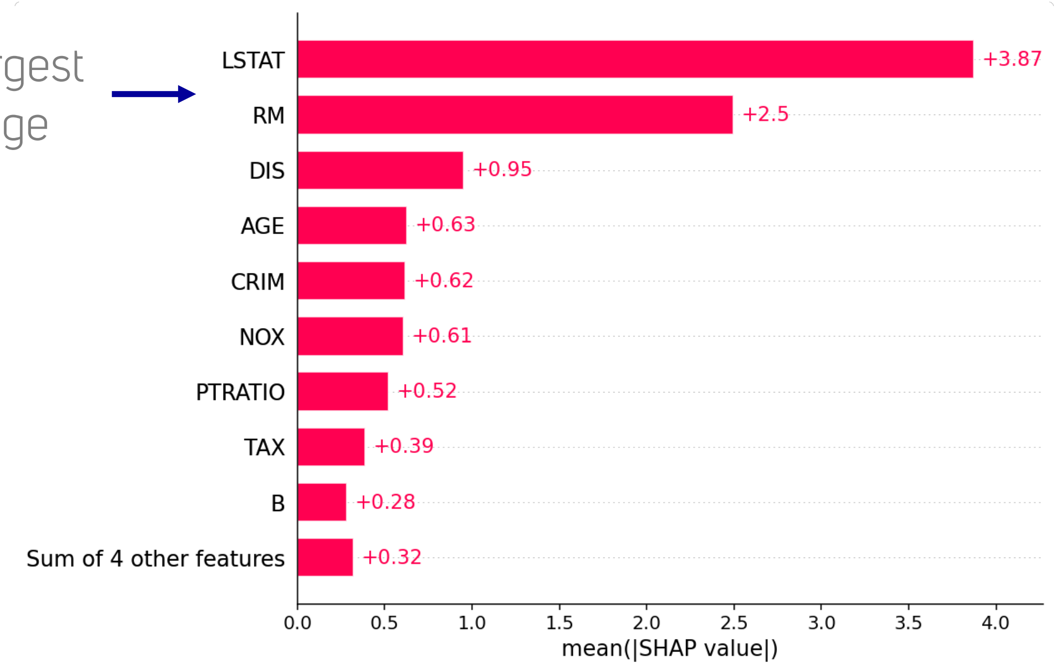


SHAP values add up to the difference (efficiency property)

Base prediction

Global explanations

Features with largest impact, on average



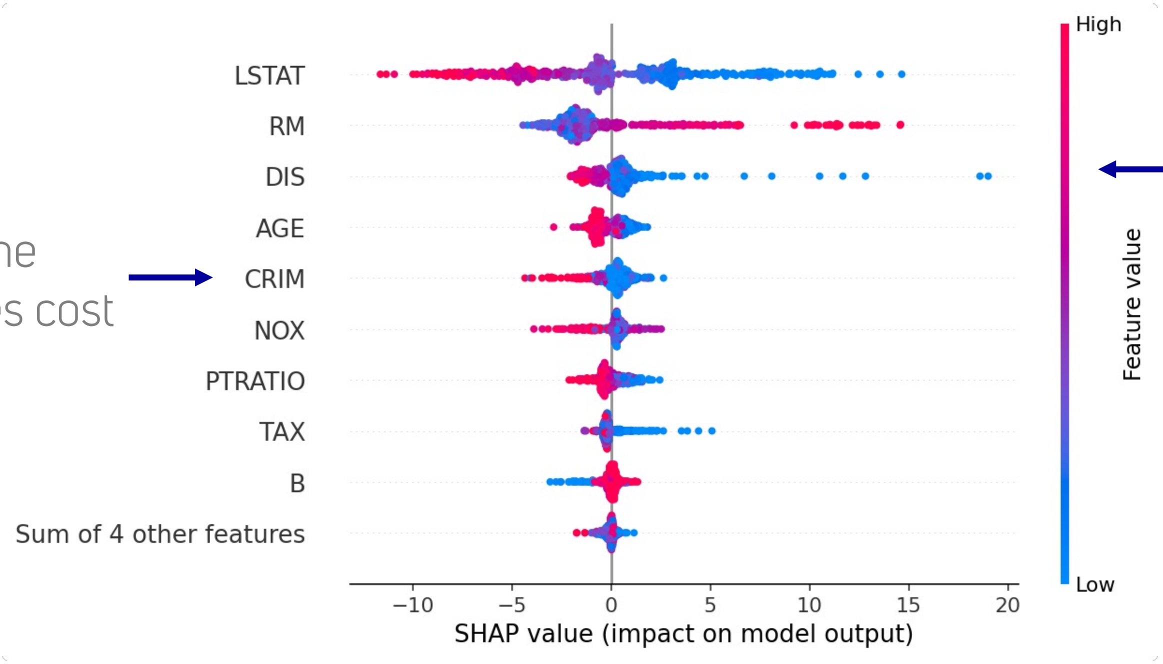
Aggregating local SHAP values

Summary plot

Each point colored by feature value

Low distance increases cost

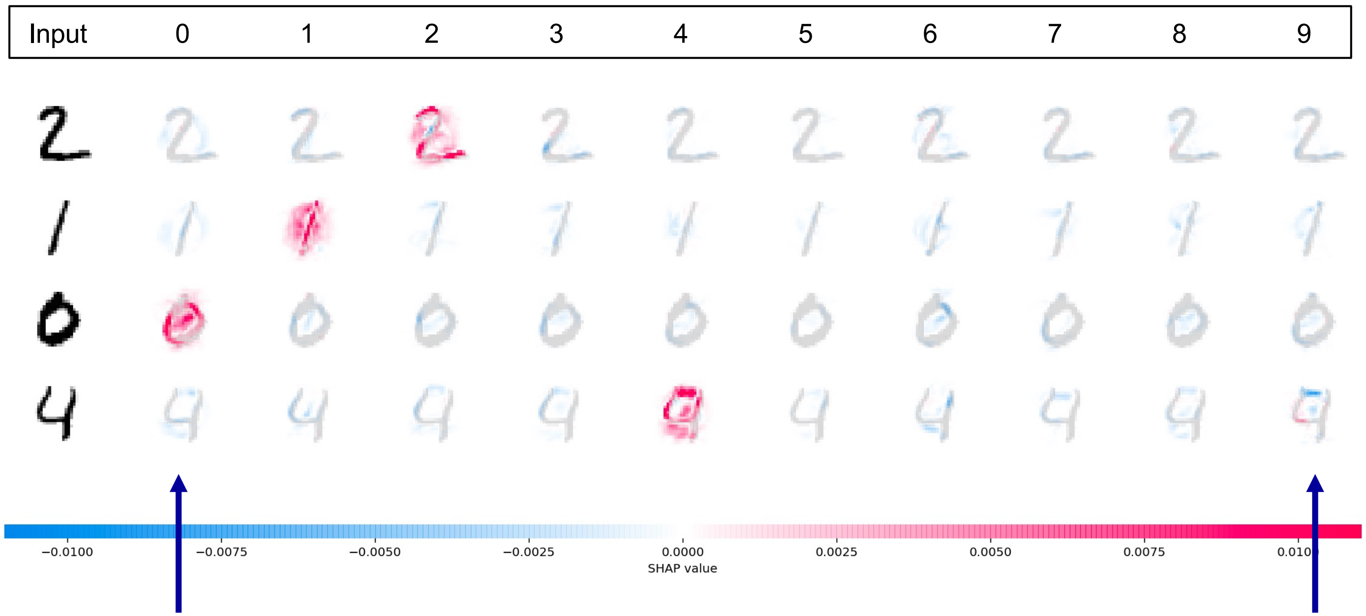
High crime decreases cost



Local Shapley values for individual data points

Image explanations

Pen strokes indicate true digit



Lack of arc means it's not a zero

Lack of top line means not a nine

Conclusion

- Shapley values are an elegant idea from game theory
- Now used by multiple XAI methods, most famously by SHAP for individual predictions
- Leads to computational challenges, so we use approximations in practice
  - Simulate feature removal
  - Approximate Shapley values