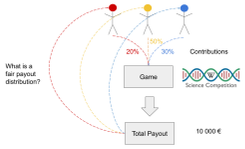
Interpretable Machine Learning

Shapley Values

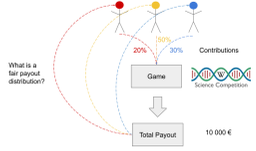


Learning goals

- Learn cooperative games and value functions
- Define the marginal contribution of a player
- Study Shapley value as a fair payout solution
- Compare order and set definitions

Interpretable Machine Learning

Shapley Values

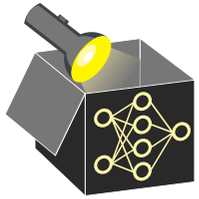


Learning goals

- Learn cooperative games and value functions
- Define the marginal contribution of a player
- Study Shapley value as a fair payout solution
- Compare order and set definitions

COOPERATIVE GAMES IN GAME THEORY

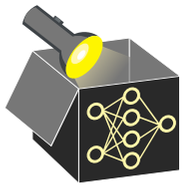
▸ Shapley (1951)



- **Game theory:** Studies strategic interactions among "players" (who act to maximize their utility), where outcomes depend on collective behavior
- **Cooperative games:** Any subset $S \subseteq P = \{1, \ldots, p\}$ can form a coalition to cooperate in a game, each achieving a payout $v(S)$

COOPERATIVE GAMES IN GAME THEORY

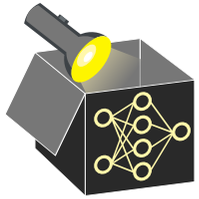
▸ SHAPLEY_1951



- **Game theory:** Studies strategic interactions among "players" (who act to maximize their utility), where outcomes depend on collective behavior
- **Cooperative games:** Any subset $S \subseteq P = \{1, \ldots, p\}$ can form a coalition to cooperate in a game, each achieving a payout $v(S)$

COOPERATIVE GAMES IN GAME THEORY

Shapley (1951)

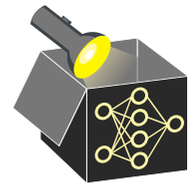


- **Game theory:** Studies strategic interactions among "players" (who act to maximize their utility), where outcomes depend on collective behavior
- **Cooperative games:** Any subset $S \subseteq P = \{1, \ldots, p\}$ can form a coalition to cooperate in a game, each achieving a payout $v(S)$
- **Value function:** $v : 2^P \to \mathbb{R}$ assigns each coalition S a payout $v(S)$
  - Convention: $v(\emptyset) = 0 \rightsquigarrow$ Empty coalitions generate no gain
  - $v(P)$: Total achievable payout when all players cooperate
    $\rightsquigarrow$ Forms the game's budget to be fairly distributed
- **Marginal contribution:** Measure how much value player j adds to coalition S by

$$\Delta(j, S) := v(S \cup \{j\}) - v(S) \quad \text{(for all } j \in P \; S \subseteq P \setminus \{j\})$$

COOPERATIVE GAMES IN GAME THEORY

SHAPLEY_1951



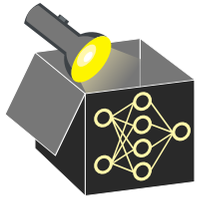
- **Game theory:** Studies strategic interactions among "players" (who act to maximize their utility), where outcomes depend on collective behavior
- **Cooperative games:** Any subset $S \subseteq P = \{1, \ldots, p\}$ can form a coalition to cooperate in a game, each achieving a payout $v(S)$
- **Value function:** $v : 2^P \to \mathbb{R}$ assigns each coalition S a payout $v(S)$
  - Convention: $v(\emptyset) = 0 \rightsquigarrow$ Empty coalitions generate no gain
  - $v(P)$: Total achievable payout when all players cooperate
    $\rightsquigarrow$ Forms the game's budget to be fairly distributed
- **Marginal contribution:** Measure how much value player j adds to coalition S by

$$\Delta(j, S) := v(S \cup \{j\}) - v(S) \quad \text{(for all } j \in P \; S \subseteq P \setminus \{j\})$$

Interpretable Machine Learning – 1 / 11

COOPERATIVE GAMES IN GAME THEORY

▸ Shapley (1951)



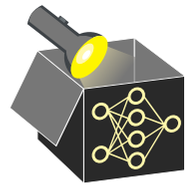
- **Game theory:** Studies strategic interactions among "players" (who act to maximize their utility), where outcomes depend on collective behavior
- **Cooperative games:** Any subset $S \subseteq P = \{1, \ldots, p\}$ can form a coalition to cooperate in a game, each achieving a payout $v(S)$
- **Value function:** $v : 2^P \to \mathbb{R}$ assigns each coalition S a payout $v(S)$
  - Convention: $v(\emptyset) = 0 \rightsquigarrow$ Empty coalitions generate no gain
  - $v(P)$: Total achievable payout when all players cooperate
    $\rightsquigarrow$ Forms the game's budget to be fairly distributed
- **Marginal contribution:** Measure how much value player j adds to coalition S by

$$\Delta(j, S) := v(S \cup \{j\}) - v(S) \quad \text{(for all } j \in P \; S \subseteq P \setminus \{j\})$$

- **Challenge:** Players vary in their contributions & how they influence each other
- **Goal:** Fairly distribute $v(P)$ among players by accounting for player interactions
  $\rightsquigarrow$ Assign each player $j \in P$ a fair share ϕ_j (**Shapley value**)

COOPERATIVE GAMES IN GAME THEORY

▸ SHAPLEY_1951

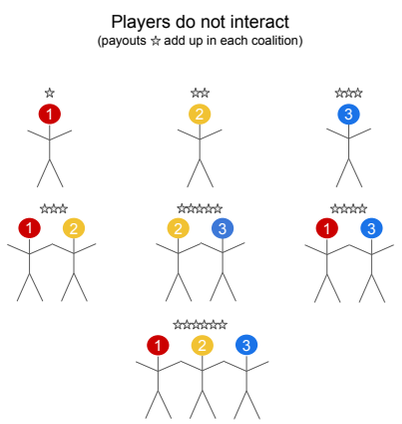


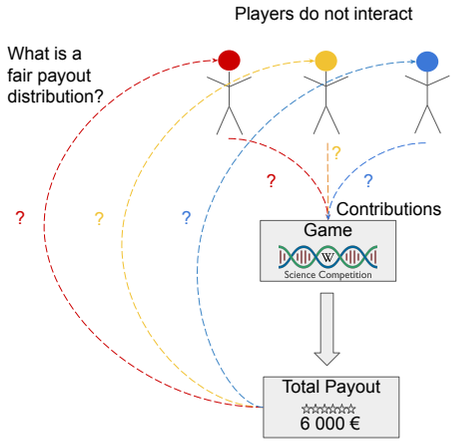
- **Game theory:** Studies strategic interactions among "players" (who act to maximize their utility), where outcomes depend on collective behavior
- **Cooperative games:** Any subset $S \subseteq P = \{1, \ldots, p\}$ can form a coalition to cooperate in a game, each achieving a payout $v(S)$
- **Value function:** $v : 2^P \to \mathbb{R}$ assigns each coalition S a payout $v(S)$
  - Convention: $v(\emptyset) = 0 \rightsquigarrow$ Empty coalitions generate no gain
  - $v(P)$: Total achievable payout when all players cooperate
    $\rightsquigarrow$ Forms the game's budget to be fairly distributed
- **Marginal contribution:** Measure how much value player j adds to coalition S by

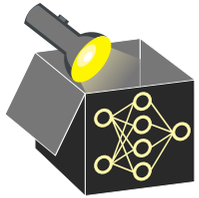
$$\Delta(j, S) := v(S \cup \{j\}) - v(S) \quad \text{(for all } j \in P \; S \subseteq P \setminus \{j\})$$

- **Challenge:** Players vary in their contrib. & how they influence each other
- **Goal:** Distribute $v(P)$ among players by considering player interactions
  $\rightsquigarrow$ Assign each player $j \in P$ a fair share ϕ_j (**Shapley value**)

COOPERATIVE GAMES - NO INTERACTIONS

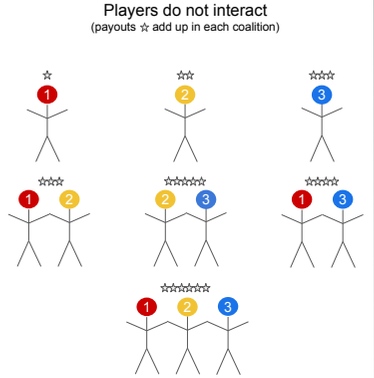


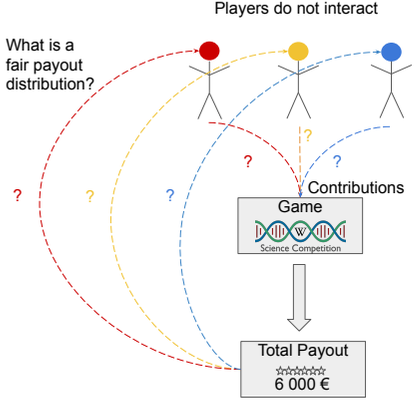


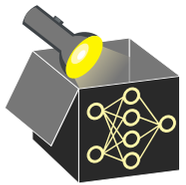


Question: What are the individual marginal contributions and what is a fair payout?

COOPERATIVE GAMES - NO INTERACTIONS

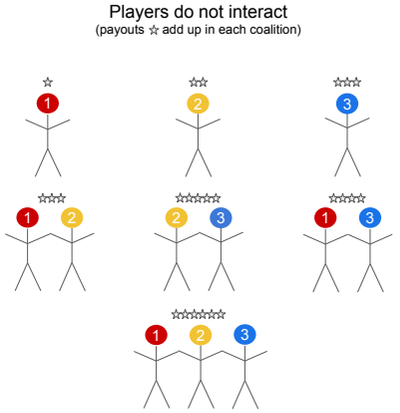




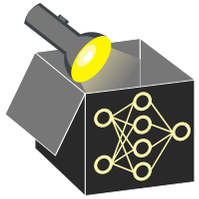


Question: What are individual marginal contributions and what's a fair payout?

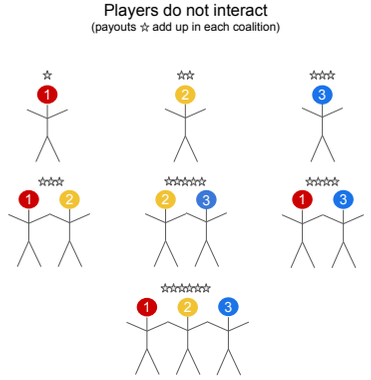
COOPERATIVE GAMES - NO INTERACTIONS



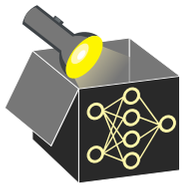
Player	Coalition S	$v(S \cup \{j\})$	$v(S)$	$\Delta(j, S)$
1	$\emptyset$	1000	0	1000
1	{2}	3000	2000	1000
1	{3}	4000	3000	1000
1	{2, 3}	6000	5000	1000
2	$\emptyset$	2000	0	2000
2	{1}	3000	1000	2000
2	{3}	5000	3000	2000
2	{1, 3}	6000	4000	2000
3	$\emptyset$	3000	0	3000
3	{1}	4000	1000	3000
3	{2}	5000	2000	3000
3	{1, 2}	6000	3000	3000



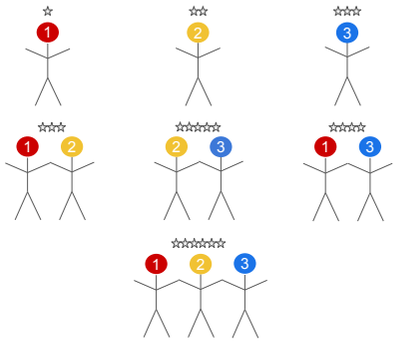
COOPERATIVE GAMES - NO INTERACTIONS



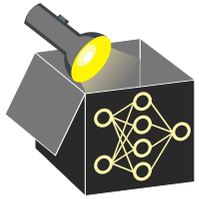
Player	Coalition S	$v(S \cup \{j\})$	$v(S)$	$\Delta(j, S)$
1	$\emptyset$	1000	0	1000
1	{2}	3000	2000	1000
1	{3}	4000	3000	1000
1	{2, 3}	6000	5000	1000
2	$\emptyset$	2000	0	2000
2	{1}	3000	1000	2000
2	{3}	5000	3000	2000
2	{1, 3}	6000	4000	2000
3	$\emptyset$	3000	0	3000
3	{1}	4000	1000	3000
3	{2}	5000	2000	3000
3	{1, 2}	6000	3000	3000



COOPERATIVE GAMES - NO INTERACTIONS



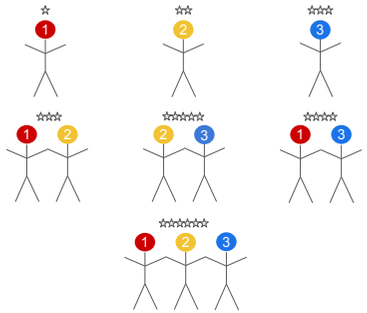
Player	Coalition S	$v(S \cup \{j\})$	$v(S)$	$\Delta(j, S)$
1	$\emptyset$	1000	0	1000
1	{2}	3000	2000	1000
1	{3}	4000	3000	1000
1	{2, 3}	6000	5000	1000
2	$\emptyset$	2000	0	2000
2	{1}	3000	1000	2000
2	{3}	5000	3000	2000
2	{1, 3}	6000	4000	2000
3	$\emptyset$	3000	0	3000
3	{1}	4000	1000	3000
3	{2}	5000	2000	3000
3	{1, 2}	6000	3000	3000



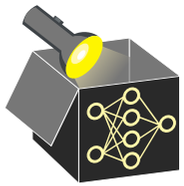
- **No interactions:** Each player contributes the same fixed value to each coalition
  - ⇝ Player 1 always adds 1000, 2 adds 2000, and 3 adds 3000
  - ⇝ Marginal contributions are constant across all coalitions S
- **Conclusion:** Fair payout = average marginal contribution across all S
  - ⇝ Total value $v(P) = 6000$ splits proportionally by individual contributions:

$$1 = \frac{1}{6}, \quad 2 = \frac{1}{3}, \quad 3 = \frac{1}{2}$$

COOPERATIVE GAMES - NO INTERACTIONS



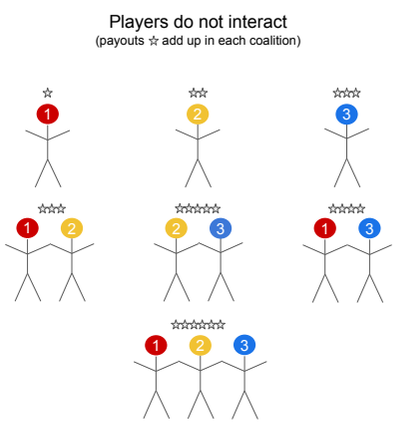
Player	Coalition S	$v(S \cup \{j\})$	$v(S)$	$\Delta(j, S)$
1	$\emptyset$	1000	0	1000
1	{2}	3000	2000	1000
1	{3}	4000	3000	1000
1	{2, 3}	6000	5000	1000
2	$\emptyset$	2000	0	2000
2	{1}	3000	1000	2000
2	{3}	5000	3000	2000
2	{1, 3}	6000	4000	2000
3	$\emptyset$	3000	0	3000
3	{1}	4000	1000	3000
3	{2}	5000	2000	3000
3	{1, 2}	6000	3000	3000

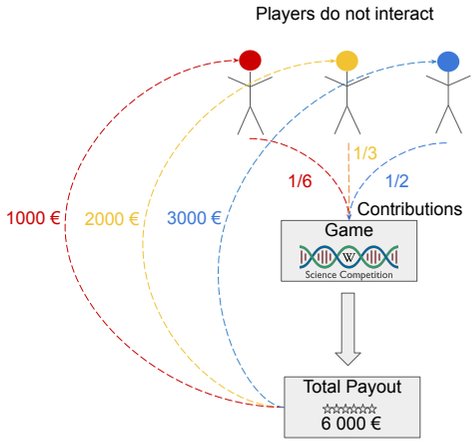


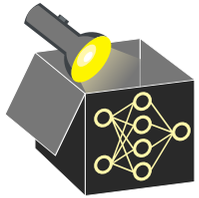
- **No interactions:** Each player contrib.s same fixed value to each coalition
  - ⇝ Player 1 always adds 1000, 2 adds 2000, and 3 adds 3000
  - ⇝ Marginal contributions are constant across all coalitions S
- **Conclusion:** Fair payout = average marginal contribution across all S
  - ⇝ Total value $v(P) = 6000$ splits proportionally by individual contribs:

$$1 = \frac{1}{6}, \quad 2 = \frac{1}{3}, \quad 3 = \frac{1}{2}$$

COOPERATIVE GAMES - NO INTERACTIONS

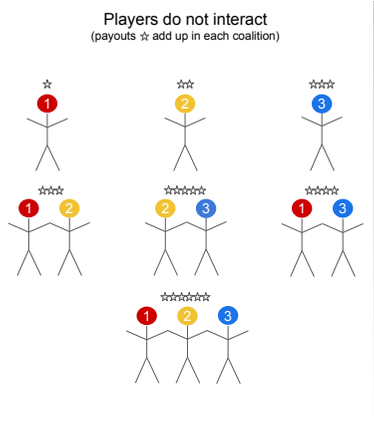


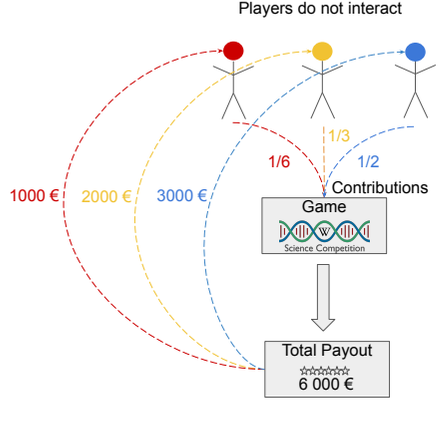


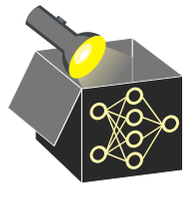


⇒ Fair payouts are trivial without interactions

COOPERATIVE GAMES - NO INTERACTIONS

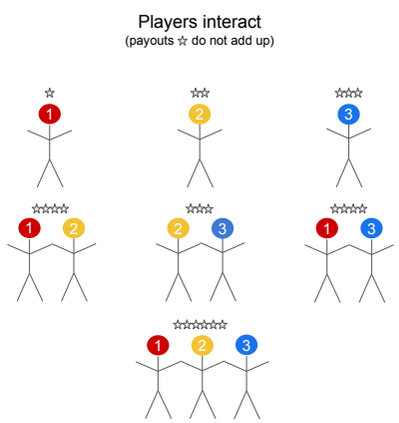


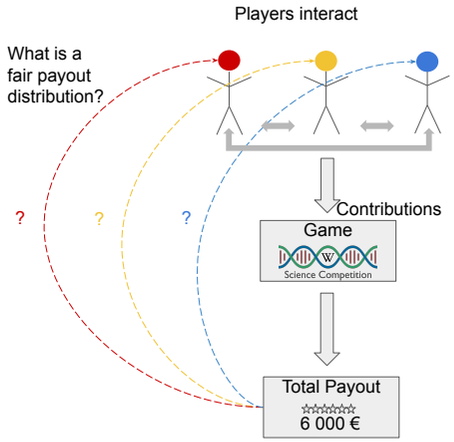


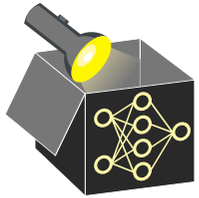


⇒ Fair payouts are trivial without interactions

COOPERATIVE GAMES - INTERACTIONS

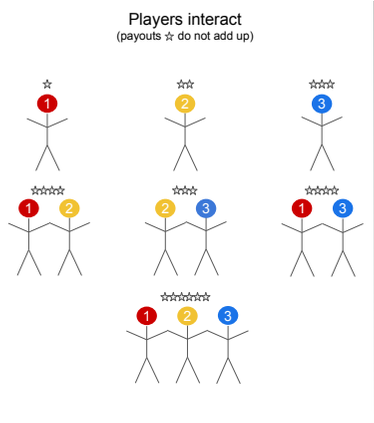


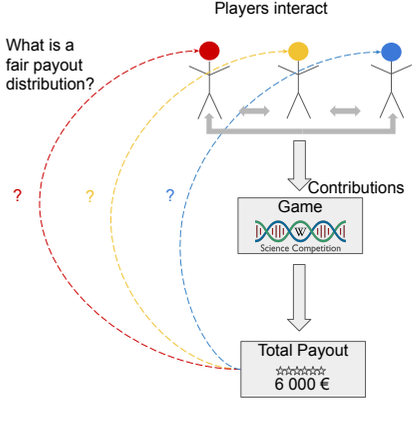


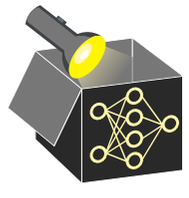


⇒ Unclear how to fairly distribute payouts when players interact

COOPERATIVE GAMES - INTERACTIONS

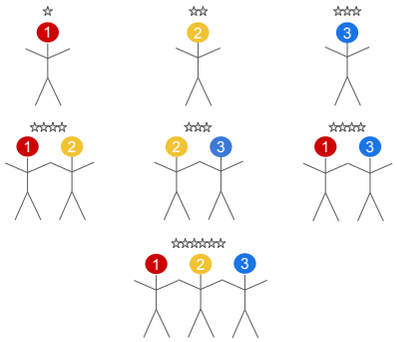




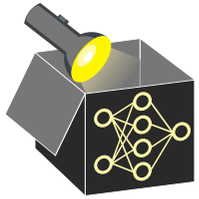


⇒ Unclear how to fairly distribute payouts when players interact

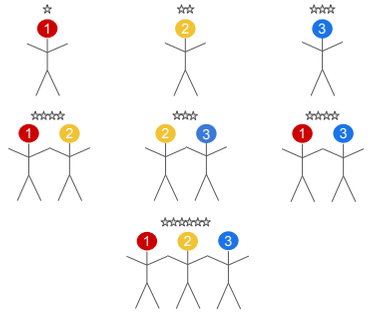
COOPERATIVE GAMES - INTERACTIONS



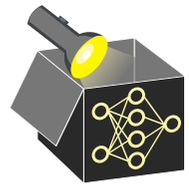
Player	Coalition S	$v(S \cup \{j\})$	$v(S)$	$\Delta(j, S)$
1	$\emptyset$	1000	0	1000
1	$\{2\}$	4000	2000	2000
1	$\{3\}$	4000	3000	1000
1	$\{2, 3\}$	6000	3000	3000
2	$\emptyset$	2000	0	2000
2	$\{1\}$	4000	1000	3000
2	$\{3\}$	3000	3000	0
2	$\{1, 3\}$	6000	4000	2000
3	$\emptyset$	3000	0	3000
3	$\{1\}$	4000	1000	3000
3	$\{2\}$	3000	2000	1000
3	$\{1, 2\}$	6000	4000	2000



COOPERATIVE GAMES - INTERACTIONS

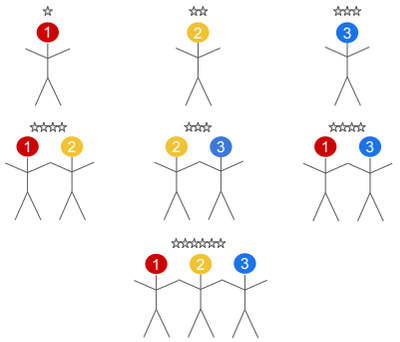


Player	Coalition S	$v(S \cup \{j\})$	$v(S)$	$\Delta(j, S)$
1	$\emptyset$	1000	0	1000
1	$\{2\}$	4000	2000	2000
1	$\{3\}$	4000	3000	1000
1	$\{2, 3\}$	6000	3000	3000
2	$\emptyset$	2000	0	2000
2	$\{1\}$	4000	1000	3000
2	$\{3\}$	3000	3000	0
2	$\{1, 3\}$	6000	4000	2000
3	$\emptyset$	3000	0	3000
3	$\{1\}$	4000	1000	3000
3	$\{2\}$	3000	2000	1000
3	$\{1, 2\}$	6000	4000	2000

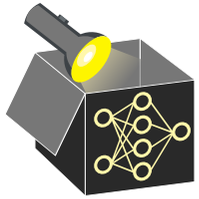


Interpretable Machine Learning – 4 / 11

COOPERATIVE GAMES - INTERACTIONS

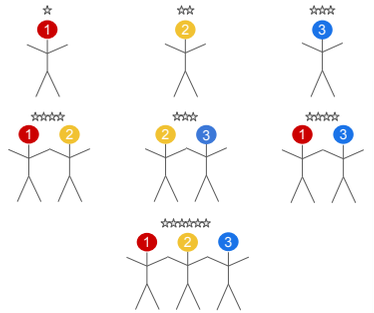


Player	Coalition S	$v(S \cup \{j\})$	$v(S)$	$\Delta(j, S)$
1	$\emptyset$	1000	0	1000
1	{2}	4000	2000	2000
1	{3}	4000	3000	1000
1	{2, 3}	6000	3000	3000
2	$\emptyset$	2000	0	2000
2	{1}	4000	1000	3000
2	{3}	3000	3000	0
2	{1, 3}	6000	4000	2000
3	$\emptyset$	3000	0	3000
3	{1}	4000	1000	3000
3	{2}	3000	2000	1000
3	{1, 2}	6000	4000	2000

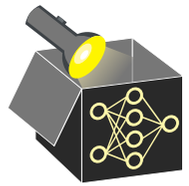


- **With interactions:** Players contribute different amounts depending on coalition
  ⇝ Marginal contributions vary across coalitions S (e.g., due to overlap, synergy)
- Averaging over subsets does not recover total payout $v(P)$ ⇝ unfair payout distr.
  ⇝ average contrib. 1 = 1750, 2 = 1750, 3 = 2250 do not sum to $v(P) = 6000$
- Value a player adds depends on joining order, not just who else is in the coalition
  ⇝ Shapley values fairly average over all possible joining orders

COOPERATIVE GAMES - INTERACTIONS

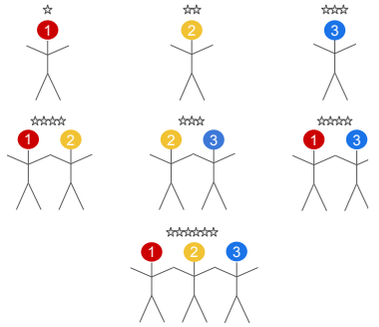


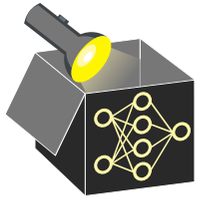
Player	Coalition S	$v(S \cup \{j\})$	$v(S)$	$\Delta(j, S)$
1	$\emptyset$	1000	0	1000
1	{2}	4000	2000	2000
1	{3}	4000	3000	1000
1	{2, 3}	6000	3000	3000
2	$\emptyset$	2000	0	2000
2	{1}	4000	1000	3000
2	{3}	3000	3000	0
2	{1, 3}	6000	4000	2000
3	$\emptyset$	3000	0	3000
3	{1}	4000	1000	3000
3	{2}	3000	2000	1000
3	{1, 2}	6000	4000	2000



- **With interactions:** Players contribute differently depending on coalition
  ⇝ Marginal contribs vary across coalitions S (e.g. overlap, synergy)
- Averaging over subsets does not recover total payout $v(P)$
  ⇝ unfair payout distribution
  ⇝ avg. contrib. 1 = 1750 2 = 1750 3 = 2250 don't sum to $v(P) = 6000$
- Value a player adds depends on joining order, not just who's in coalition
  ⇝ Shapley values fairly average over all possible joining orders

COOPERATIVE GAMES - INTERACTIONS





Ordering 1: 3 → 2 → 1

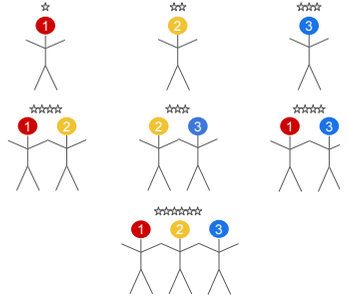
- 3 joins alone: 3 ☆
- 2 joins: total = 3 ☆, marginal = 0
- 1 joins: total = 6 ☆, marginal = +3

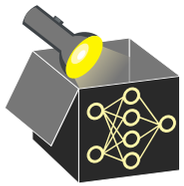
But what if 1 joins before 2?

Ordering 2: 3 → 1 → 2

- 3 joins alone: 3 ☆
- 1 joins: total = 4 ☆, marginal = +1
- 2 joins: total = 6 ☆, marginal = +2

COOPERATIVE GAMES - INTERACTIONS





Ordering 1: 3 → 2 → 1

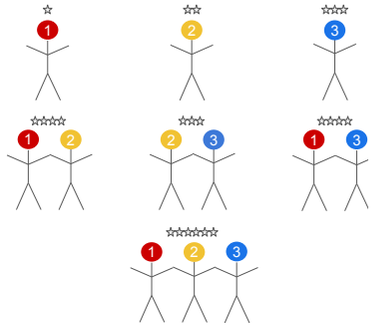
- 3 joins alone: 3 ☆
- 2 joins: total = 3 ☆, marginal = 0
- 1 joins: total = 6 ☆, marginal = +3

But what if 1 joins before 2?

Ordering 2: 3 → 1 → 2

- 3 joins alone: 3 ☆
- 1 joins: total = 4 ☆, marginal = +1
- 2 joins: total = 6 ☆, marginal = +2

COOPERATIVE GAMES - INTERACTIONS



Ordering 1: 3 → 2 → 1

- 3 joins alone: 3 ☆
- 2 joins: total = 3 ☆, marginal = 0
- 1 joins: total = 6 ☆, marginal = +3

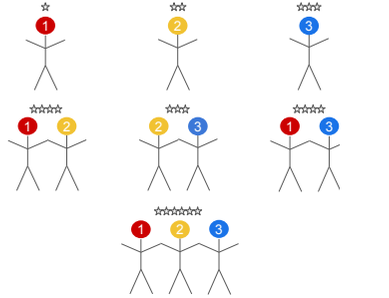
But what if 1 joins before 2?

Ordering 2: 3 → 1 → 2

- 3 joins alone: 3 ☆
- 1 joins: total = 4 ☆, marginal = +1
- 2 joins: total = 6 ☆, marginal = +2

- **Order sensitivity:** A player's marginal contribution depends on when they join S
- **Shapley value:** Averages each player's contribution over all possible join orders
  - ⇝ Resolves redundancy (e.g., 3's contribution/skill overlaps with 2's)
  - ⇝ Accounts for order sensitivity (e.g., 1 brings more value if added last)
  - ⇝ Ensures fairness (no player is advantaged or penalized by order of joining)

COOPERATIVE GAMES - INTERACTIONS



Ordering 1: 3 → 2 → 1

- 3 joins alone: 3 ☆
- 2 joins: total = 3 ☆, marginal = 0
- 1 joins: total = 6 ☆, marginal = +3

But what if 1 joins before 2?

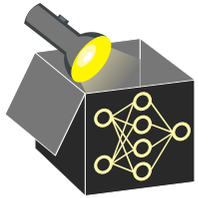
Ordering 2: 3 → 1 → 2

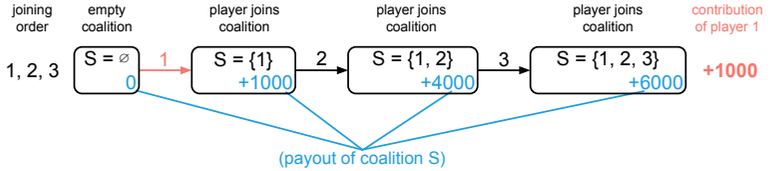
- 3 joins alone: 3 ☆
- 1 joins: total = 4 ☆, marginal = +1
- 2 joins: total = 6 ☆, marginal = +2

- **Order sensitivity:** A player's marginal contribution depends on when they join S
- **Shapley value:** Averages each player's contribution over all possible join orders
  - ⇝ Resolves redundancy (e.g., 3's contribution/skill overlaps with 2's)
  - ⇝ Accounts for order sensitivity (e.g., 1 brings more value if added last)
  - ⇝ Ensures fairness (order of joining gives no advantage/disadvantage)

SHAPLEY VALUES - ILLUSTRATION

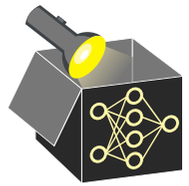
- Generate all possible joining orders of players (all permutations of full set P)
- For each order: track player j-th marginal contribution when j joins a coalition

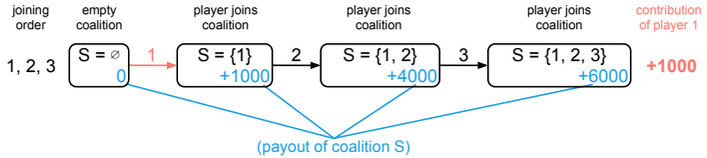




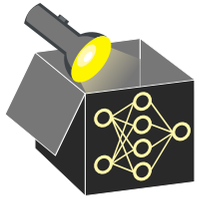
SHAPLEY VALUES - ILLUSTRATION

- Generate all possible joining orders (all permutations of full set P)
- For each order: track player j-th marginal contrib when j joins a coalition





SHAPLEY VALUES - ILLUSTRATION

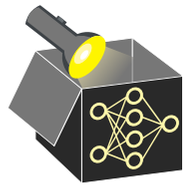


- Generate all possible joining orders of players (all permutations of full set P)
- For each order: track player j-th marginal contribution when j joins a coalition
- Shapley value of j: Average this marginal contribution over all joining orders
- **Example:** Compute payout difference after player 1 enters coalition $\rightsquigarrow$ average

joining order	empty coalition		player joins coalition		player joins coalition		player joins coalition	contribution of player 1
1, 2, 3	S = ∅ 0	1	S = {1} +1000	2	S = {1, 2} +4000	3	S = {1, 2, 3} +6000	+1000
1, 3, 2	S = ∅ 0	1	S = {1} +1000	3	S = {1, 3} +4000	2	S = {1, 2, 3} +6000	+1000
2, 1, 3	S = ∅ 0	2	S = {2} +2000	1	S = {1, 2} +4000	3	S = {1, 2, 3} +6000	+2000
2, 3, 1	S = ∅ 0	2	S = {2} +2000	3	S = {2, 3} +3000	1	S = {1, 2, 3} +6000	+3000
3, 1, 2	S = ∅ 0	3	S = {3} +3000	1	S = {1, 3} +4000	2	S = {1, 2, 3} +6000	+1000
3, 2, 1	S = ∅ 0	3	S = {3} +3000	2	S = {2, 3} +3000	1	S = {1, 2, 3} +6000	+3000

Shapley value of player 1 : +1833.33

SHAPLEY VALUES - ILLUSTRATION

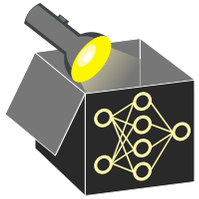


- Generate all possible joining orders (all permutations of full set P)
- For each order: track player j-th marginal contrib when j joins a coalition
- Shapley value of j: Average this marginal contrib over all joining orders
- **Example:** Compute payout diff. after player 1 enters coalition $\rightsquigarrow$ average

joining order	empty coalition		player joins coalition		player joins coalition		player joins coalition	contribution of player 1
1, 2, 3	S = ∅ 0	1	S = {1} +1000	2	S = {1, 2} +4000	3	S = {1, 2, 3} +6000	+1000
1, 3, 2	S = ∅ 0	1	S = {1} +1000	3	S = {1, 3} +4000	2	S = {1, 2, 3} +6000	+1000
2, 1, 3	S = ∅ 0	2	S = {2} +2000	1	S = {1, 2} +4000	3	S = {1, 2, 3} +6000	+2000
2, 3, 1	S = ∅ 0	2	S = {2} +2000	3	S = {2, 3} +3000	1	S = {1, 2, 3} +6000	+3000
3, 1, 2	S = ∅ 0	3	S = {3} +3000	1	S = {1, 3} +4000	2	S = {1, 2, 3} +6000	+1000
3, 2, 1	S = ∅ 0	3	S = {3} +3000	2	S = {2, 3} +3000	1	S = {1, 2, 3} +6000	+3000

Shapley value of player 1 : +1833.33

SHAPLEY VALUES - ILLUSTRATION

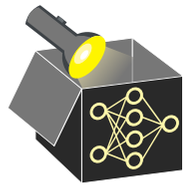


- Generate all possible joining orders of players (all permutations of full set P)
- For each order: track player j-th marginal contribution when j joins a coalition
- Shapley value of j: Average this marginal contribution over all joining orders
- **Example:** Compute payout difference after player 2 enters coalition $\rightsquigarrow$ average

joining order	empty coalition		player joins coalition		player joins coalition		player joins coalition	contribution of player 2
1, 2, 3	S = ∅ 0	1	S = {1} +1000	2	S = {1, 2} +4000	3	S = {1, 2, 3} +6000	+3000
1, 3, 2	S = ∅ 0	1	S = {1} +1000	3	S = {1, 3} +4000	2	S = {1, 2, 3} +6000	+2000
2, 1, 3	S = ∅ 0	2	S = {2} +2000	1	S = {1, 2} +4000	3	S = {1, 2, 3} +6000	+2000
2, 3, 1	S = ∅ 0	2	S = {2} +2000	3	S = {2, 3} +3000	1	S = {1, 2, 3} +6000	+2000
3, 1, 2	S = ∅ 0	3	S = {3} +3000	1	S = {1, 3} +4000	2	S = {1, 2, 3} +6000	+2000
3, 2, 1	S = ∅ 0	3	S = {3} +3000	2	S = {2, 3} +3000	1	S = {1, 2, 3} +6000	+0

Shapley value of player 2: +1833.33

SHAPLEY VALUES - ILLUSTRATION

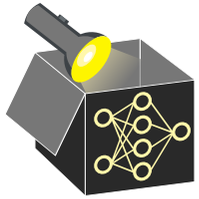


- Generate all possible joining orders (all permutations of full set P)
- For each order: track player j-th marginal contrib when j joins a coalition
- Shapley value of j: Average this marginal contrib over all joining orders
- **Example:** Compute payout diff. after player 2 enters coalition $\rightsquigarrow$ average

joining order	empty coalition		player joins coalition		player joins coalition		player joins coalition	contribution of player 2
1, 2, 3	S = ∅ 0	1	S = {1} +1000	2	S = {1, 2} +4000	3	S = {1, 2, 3} +6000	+3000
1, 3, 2	S = ∅ 0	1	S = {1} +1000	3	S = {1, 3} +4000	2	S = {1, 2, 3} +6000	+2000
2, 1, 3	S = ∅ 0	2	S = {2} +2000	1	S = {1, 2} +4000	3	S = {1, 2, 3} +6000	+2000
2, 3, 1	S = ∅ 0	2	S = {2} +2000	3	S = {2, 3} +3000	1	S = {1, 2, 3} +6000	+2000
3, 1, 2	S = ∅ 0	3	S = {3} +3000	1	S = {1, 3} +4000	2	S = {1, 2, 3} +6000	+2000
3, 2, 1	S = ∅ 0	3	S = {3} +3000	2	S = {2, 3} +3000	1	S = {1, 2, 3} +6000	+0

Shapley value of player 2: +1833.33

SHAPLEY VALUES - ILLUSTRATION

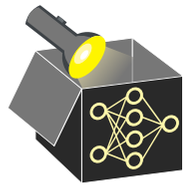


- Generate all possible joining orders of players (all permutations of full set P)
- For each order: track player j-th marginal contribution when j joins a coalition
- Shapley value of j: Average this marginal contribution over all joining orders
- **Example:** Compute payout difference after player 3 enters coalition $\rightsquigarrow$ average

joining order	empty coalition		player joins coalition		player joins coalition		player joins coalition	contribution of player 3
1, 2, 3	S = ∅ 0	1	S = {1} +1000	2	S = {1, 2} +4000	3	S = {1, 2, 3} +6000	+2000
1, 3, 2	S = ∅ 0	1	S = {1} +1000	3	S = {1, 3} +4000	2	S = {1, 2, 3} +6000	+3000
2, 1, 3	S = ∅ 0	2	S = {2} +2000	1	S = {1, 2} +4000	3	S = {1, 2, 3} +6000	+2000
2, 3, 1	S = ∅ 0	2	S = {2} +2000	3	S = {2, 3} +3000	1	S = {1, 2, 3} +6000	+1000
3, 1, 2	S = ∅ 0	3	S = {3} +3000	1	S = {1, 3} +4000	2	S = {1, 2, 3} +6000	+3000
3, 2, 1	S = ∅ 0	3	S = {3} +3000	2	S = {2, 3} +3000	1	S = {1, 2, 3} +6000	+3000

Shapley value of player 3 : +2333.33

SHAPLEY VALUES - ILLUSTRATION



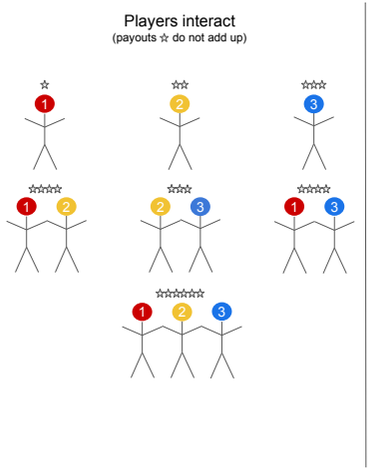
- Generate all possible joining orders (all permutations of full set P)
- For each order: track player j-th marginal contrib when j joins a coalition
- Shapley value of j: Average this marginal contrib over all joining orders
- **Example:** Compute payout diff. after player 3 enters coalition $\rightsquigarrow$ average

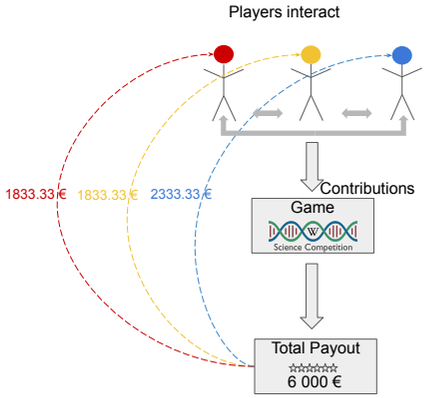
joining order	empty coalition		player joins coalition		player joins coalition		player joins coalition	contribution of player 3
1, 2, 3	S = ∅ 0	1	S = {1} +1000	2	S = {1, 2} +4000	3	S = {1, 2, 3} +6000	+2000
1, 3, 2	S = ∅ 0	1	S = {1} +1000	3	S = {1, 3} +4000	2	S = {1, 2, 3} +6000	+3000
2, 1, 3	S = ∅ 0	2	S = {2} +2000	1	S = {1, 2} +4000	3	S = {1, 2, 3} +6000	+2000
2, 3, 1	S = ∅ 0	2	S = {2} +2000	3	S = {2, 3} +3000	1	S = {1, 2, 3} +6000	+1000
3, 1, 2	S = ∅ 0	3	S = {3} +3000	1	S = {1, 3} +4000	2	S = {1, 2, 3} +6000	+3000
3, 2, 1	S = ∅ 0	3	S = {3} +3000	2	S = {2, 3} +3000	1	S = {1, 2, 3} +6000	+3000

Shapley value of player 3 : +2333.33

SHAPLEY VALUES - ILLUSTRATION

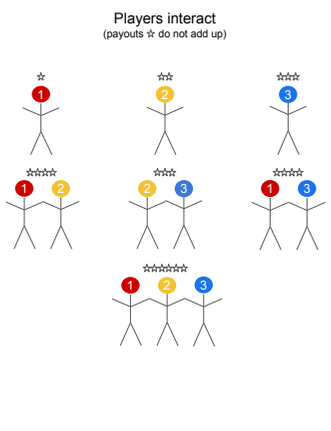
- Generate all possible joining orders of players (all permutations of full set P)
- For each order: track player j-th marginal contribution when j joins a coalition
- Shapley value of j: Average this marginal contribution over all joining orders

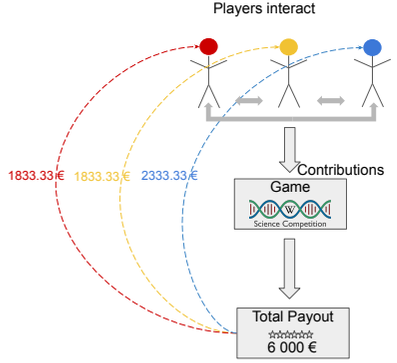




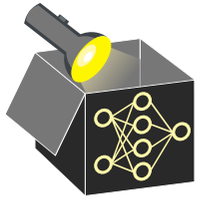
SHAPLEY VALUES - ILLUSTRATION

- Generate all possible joining orders (all permutations of full set P)
- For each order: track player j-th marginal contrib when j joins a coalition
- Shapley value of j: Average this marginal contrib over all joining orders





SHAPLEY VALUE - ORDER DEFINITION

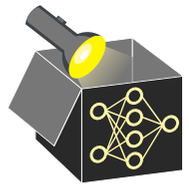


The Shapley value order definition averages the marginal contribution of a player across all possible player orderings:

$$\phi_j = \frac{1}{|P|!} \sum_{\tau \in \Pi} (v(S_j^\tau \cup \{j\}) - v(S_j^\tau))$$

- Π: Set of all permutations (joining orders) of the players – there are $|P|!$ in total

SHAPLEY VALUE - ORDER DEFINITION

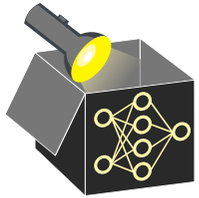


The Shapley value order definition averages the marginal contribution of a player across all possible player orderings:

$$\phi_j = \frac{1}{|P|!} \sum_{\tau \in \Pi} (v(S_j^\tau \cup \{j\}) - v(S_j^\tau))$$

- Π: Set of all permutations (joining orders) of the players – $|P|!$ in total

SHAPLEY VALUE - ORDER DEFINITION

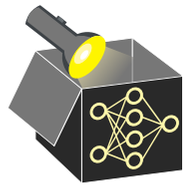


The Shapley value order definition averages the marginal contribution of a player across all possible player orderings:

$$\phi_j = \frac{1}{|P|!} \sum_{\tau \in \Pi} (v(S_j^\tau \cup \{j\}) - v(S_j^\tau))$$

- Π: Set of all permutations (joining orders) of the players – there are $|P|!$ in total
- S_j^τ: Set of players before j joins, for each ordering $\tau = (\tau^{(1)}, \ldots, \tau^{(p)})$
  E.g.: $\Pi = \{(1,2,3), (1,3,2), (2,1,3), (2,3,1), (3,1,2), (3,2,1)\}$
  $\rightsquigarrow$ For joining order $\tau = (2,1,3)$ and player $j = 3 \Rightarrow S_j^\tau = \{2,1\}$
  $\rightsquigarrow$ For joining order $\tau = (3,1,2)$ and player $j = 1 \Rightarrow S_j^\tau = \{3\}$

SHAPLEY VALUE - ORDER DEFINITION



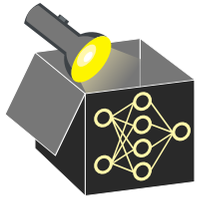
The Shapley value order definition averages the marginal contribution of a player across all possible player orderings:

$$\phi_j = \frac{1}{|P|!} \sum_{\tau \in \Pi} (v(S_j^\tau \cup \{j\}) - v(S_j^\tau))$$

- Π: Set of all permutations (joining orders) of the players – $|P|!$ in total
- S_j^τ: Set of players before j joins, for each ordering $\tau = (\tau^{(1)}, \ldots, \tau^{(p)})$
  E.g.: $\Pi = \{(1,2,3), (1,3,2), (2,1,3), (2,3,1), (3,1,2), (3,2,1)\}$
  $\rightsquigarrow$ For joining order $\tau = (2,1,3)$ and player $j = 3 \Rightarrow S_j^\tau = \{2,1\}$
  $\rightsquigarrow$ For joining order $\tau = (3,1,2)$ and player $j = 1 \Rightarrow S_j^\tau = \{3\}$

Interpretable Machine Learning – 7 / 11

SHAPLEY VALUE - ORDER DEFINITION



The Shapley value order definition averages the marginal contribution of a player across all possible player orderings:

$$\phi_j = \frac{1}{|P|!} \sum_{\tau \in \Pi} (v(S_j^\tau \cup \{j\}) - v(S_j^\tau))$$

- Π: Set of all permutations (joining orders) of the players – there are $|P|!$ in total
- S_j^τ: Set of players before j joins, for each ordering $\tau = (\tau^{(1)}, \ldots, \tau^{(p)})$
  E.g.: $\Pi = \{(1,2,3), (1,3,2), (2,1,3), (2,3,1), (3,1,2), (3,2,1)\}$
  $\rightsquigarrow$ For joining order $\tau = (2,1,3)$ and player $j = 3 \Rightarrow S_j^\tau = \{2,1\}$
  $\rightsquigarrow$ For joining order $\tau = (3,1,2)$ and player $j = 1 \Rightarrow S_j^\tau = \{3\}$
- Order definition allows to approximate Shapley values by sampling permutations
  $\rightsquigarrow$ Sample a fixed number $M \ll |P|!$ of random permutations and average:

$$\phi_j \approx \frac{1}{M} \sum_{\tau \in \Pi_M} \left(v(S_j^\tau \cup \{j\}) - v(S_j^\tau)\right)$$

where $\Pi_M \subset \Pi$ is the random sample of M player orderings

SHAPLEY VALUE - ORDER DEFINITION

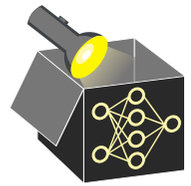
The Shapley value order definition averages the marginal contribution of a player across all possible player orderings:

$$\phi_j = \frac{1}{|P|!} \sum_{\tau \in \Pi} (v(S_j^\tau \cup \{j\}) - v(S_j^\tau))$$

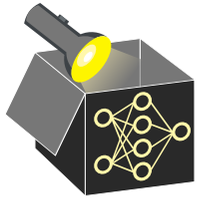
- Π: Set of all permutations (joining orders) of the players – $|P|!$ in total
- S_j^τ: Set of players before j joins, for each ordering $\tau = (\tau^{(1)}, \ldots, \tau^{(p)})$
  E.g.: $\Pi = \{(1,2,3), (1,3,2), (2,1,3), (2,3,1), (3,1,2), (3,2,1)\}$
  $\rightsquigarrow$ For joining order $\tau = (2,1,3)$ and player $j = 3 \Rightarrow S_j^\tau = \{2,1\}$
  $\rightsquigarrow$ For joining order $\tau = (3,1,2)$ and player $j = 1 \Rightarrow S_j^\tau = \{3\}$
- Order definition allows to approximate Shapley values by sampling permutations
  $\rightsquigarrow$ Sample a fixed $M \ll |P|!$ random permutations and average:

$$\phi_j \approx \frac{1}{M} \sum_{\tau \in \Pi_M} \left(v(S_j^\tau \cup \{j\}) - v(S_j^\tau)\right)$$

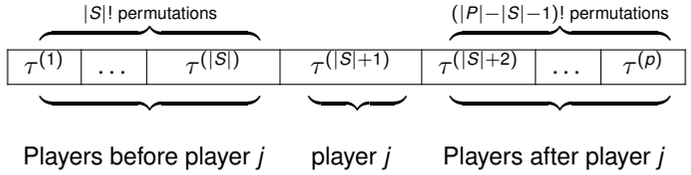
where $\Pi_M \subset \Pi$ is the random sample of M player orderings



FROM ORDER DEFINITION TO SET DEFINITION

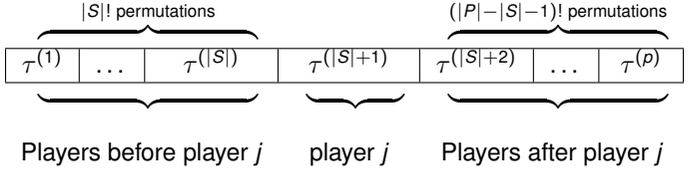


- **Note:** The same subset S_j^τ can occur in multiple permutations (joining orders)
  $\rightsquigarrow$ Its marginal contribution is included multiple times in the sum in ϕ_j
- **Example (for set of players $P = \{1, 2, 3\}$, player of interest $j = 3$):**
  $\Pi = \{(1, 2, 3), (1, 3, 2), (2, 1, 3), (2, 3, 1), (3, 1, 2), (3, 2, 1)\}$
  $\rightsquigarrow$ In both $(1, 2, 3)$ and $(2, 1, 3)$, player 3 joins after coalition $S_j^\tau = \{1, 2\}$
  $\Rightarrow$ Marginal contribution $v(\{1, 2, 3\}) - v(\{1, 2\})$ occurs twice in ϕ_j
- **Reason:** Each subset S appears in $|S|!(|P| - |S| - 1)!$ orderings before j joins
  $\Rightarrow$ There are $|S|!$ possible orders of players within coalition S
  $\Rightarrow$ There are $(|P| - |S| - 1)!$ possible orders of players without S and j

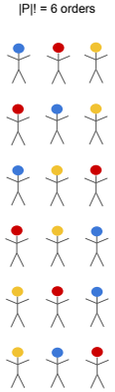


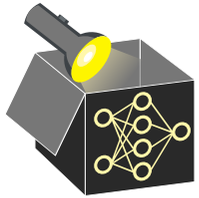
FROM ORDER DEFINITION TO SET DEFINITION

- **Note:** The same subset S_j^τ can occur in multiple permutations
  $\rightsquigarrow$ Its marginal contribution is included multiple times in the sum in ϕ_j
- **Example Π (for players $P = \{1, 2, 3\}$, player of interest $j = 3$):**
  $\{(1, 2, 3), (1, 3, 2), (2, 1, 3), (2, 3, 1), (3, 1, 2), (3, 2, 1)\}$
  $\rightsquigarrow$ In $(1, 2, 3)$ and $(2, 1, 3)$, player 3 joins after coal. $S_j^\tau = \{1, 2\}$
  $\Rightarrow$ Marginal contribution $v(\{1, 2, 3\}) - v(\{1, 2\})$ occurs twice in ϕ_j
- **Reason:** Each subset S appears in $|S|!(|P| - |S| - 1)!$ orderings before j joins
  $\Rightarrow$ There are $|S|!$ possible orders of players within coalition S
  $\Rightarrow$ There are $(|P| - |S| - 1)!$ possible orders of players without S and j



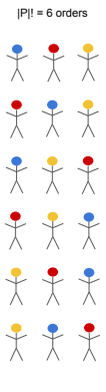
FROM ORDER DEFINITION TO SET DEFINITION

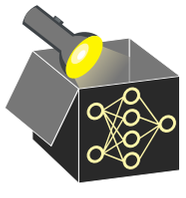




- **Order view:** Each of the $|P|!$ permutations contributes one term with weight $\frac{1}{|P|!}$
- Same subset $S \subseteq P \setminus \{j\}$ can appear before j in multiple orders
  $\rightsquigarrow$ e.g., S = {🔵, 🔴} = {🔴, 🔵}
- **Set view:** Group by unique subsets S, not permutations
- Each S occurs in $|S|!(|P| - |S| - 1)!$ orderings $\rightsquigarrow$ Weight: $\frac{|S|!(|P|-|S|-1)!}{|P|!}$

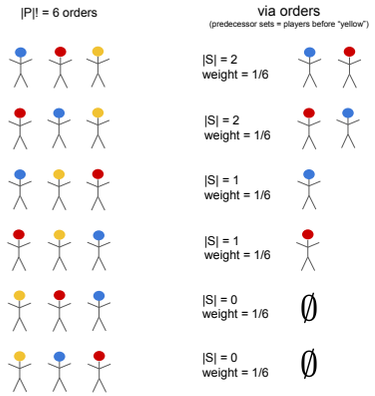
FROM ORDER DEFINITION TO SET DEFINITION

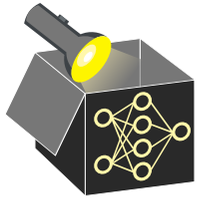




- **Order view:** Each of the $|P|!$ permuts contributes 1 term with weight $\frac{1}{|P|!}$
- Same subset $S \subseteq P \setminus \{j\}$ can appear before j in multiple orders
  $\rightsquigarrow$ e.g., S = {🔵, 🔴} = {🔴, 🔵}
- **Set view:** Group by unique subsets S, not permutations
- Each S occurs in $|S|!(|P| - |S| - 1)!$ orderings $\rightsquigarrow$ Weight: $\frac{|S|!(|P|-|S|-1)!}{|P|!}$

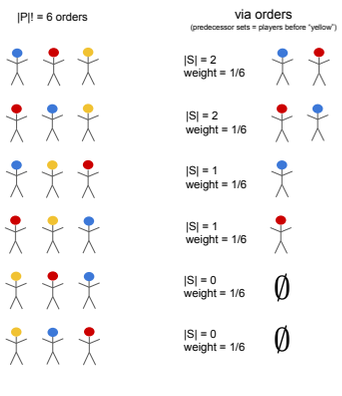
FROM ORDER DEFINITION TO SET DEFINITION

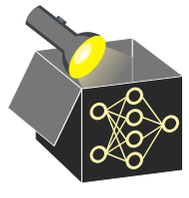




- **Order view:** Each of the $|P|!$ permutations contributes one term with weight $\frac{1}{|P|!}$
- Same subset $S \subseteq P \setminus \{j\}$ can appear before j in multiple orders
  $\rightsquigarrow$ e.g., S = {●, ●} = {●, ●}
- **Set view:** Group by unique subsets S, not permutations
- Each S occurs in $|S|!(|P| - |S| - 1)!$ orderings $\rightsquigarrow$ Weight: $\frac{|S|!(|P|-|S|-1)!}{|P|!}$

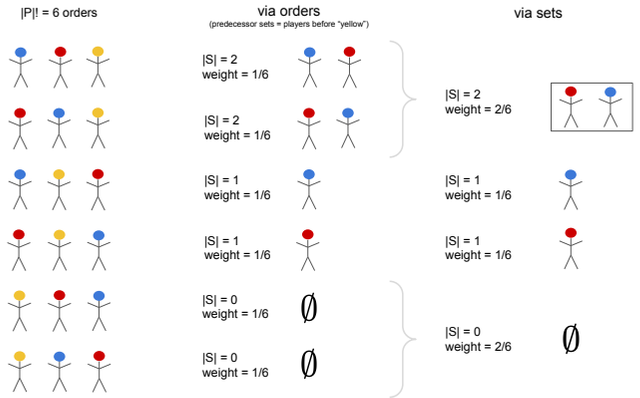
FROM ORDER DEFINITION TO SET DEFINITION

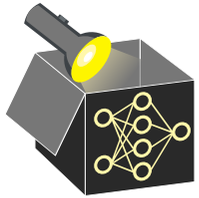




- **Order view:** Each of the $|P|!$ permuts contributes 1 term with weight $\frac{1}{|P|!}$
- Same subset $S \subseteq P \setminus \{j\}$ can appear before j in multiple orders
  $\rightsquigarrow$ e.g., S = {●, ●} = {●, ●}
- **Set view:** Group by unique subsets S, not permutations
- Each S occurs in $|S|!(|P| - |S| - 1)!$ orderings $\rightsquigarrow$ Weight: $\frac{|S|!(|P|-|S|-1)!}{|P|!}$

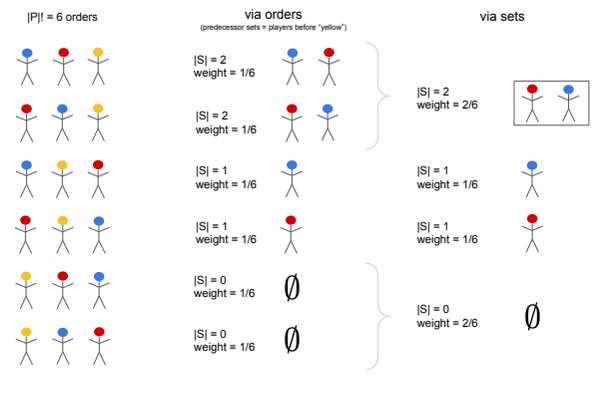
FROM ORDER DEFINITION TO SET DEFINITION

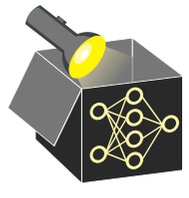




- **Order view:** Each of the $|P|!$ permutations contributes one term with weight $\frac{1}{|P|!}$
- Same subset $S \subseteq P \setminus \{j\}$ can appear before j in multiple orders
  $\rightsquigarrow$ e.g., S = {●, ●} = {●, ●}
- **Set view:** Group by unique subsets S, not permutations
- Each S occurs in $|S|!(|P| - |S| - 1)!$ orderings $\rightsquigarrow$ Weight: $\frac{|S|!(|P|-|S|-1)!}{|P|!}$

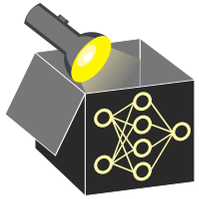
FROM ORDER DEFINITION TO SET DEFINITION





- **Order view:** Each of the $|P|!$ permuts contributes 1 term with weight $\frac{1}{|P|!}$
- Same subset $S \subseteq P \setminus \{j\}$ can appear before j in multiple orders
  $\rightsquigarrow$ e.g., S = {●, ●} = {●, ●}
- **Set view:** Group by unique subsets S, not permutations
- Each S occurs in $|S|!(|P| - |S| - 1)!$ orderings $\rightsquigarrow$ Weight: $\frac{|S|!(|P|-|S|-1)!}{|P|!}$

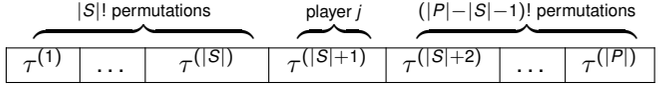
SHAPLEY VALUE - SET DEFINITION



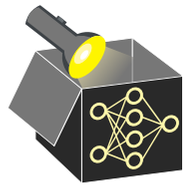
Shapley value via **set definition** (weighting via multinomial coefficient):

$$\phi_j = \sum_{S \subseteq P \setminus \{j\}} \frac{|S|!(|P| - |S| - 1)!}{|P|!} (v(S \cup \{j\}) - v(S))$$

The coefficient gives the probability that, when randomly arranging all $|P|$ players, the exact set S appears before player j, and the remaining players appear afterward.



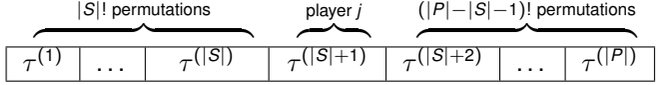
SHAPLEY VALUE - SET DEFINITION



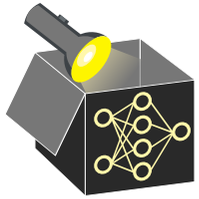
Shapley value via **set definition** (weighting via multinomial coefficient):

$$\phi_j = \sum \frac{|S|!(|P| - |S| - 1)!}{|P|!} (v() - v(S))$$

The coefficient gives the probability that, when randomly arranging all $|P|$ players, the exact set S appears before player j, and the remaining players appear afterward.



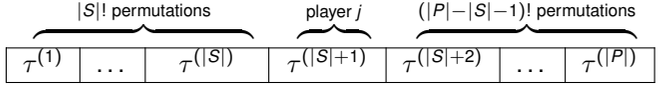
SHAPLEY VALUE - SET DEFINITION

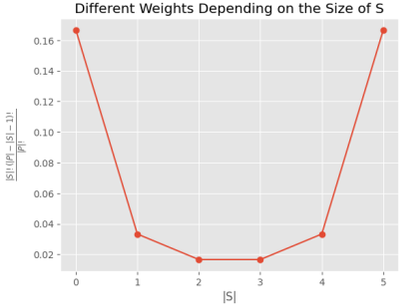


Shapley value via **set definition** (weighting via multinomial coefficient):

$$\phi_j = \sum_{S \subseteq P \setminus \{j\}} \frac{|S|!(|P| - |S| - 1)!}{|P|!} (v(S \cup \{j\}) - v(S))$$

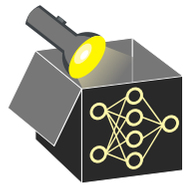
The coefficient gives the probability that, when randomly arranging all $|P|$ players, the exact set S appears before player j, and the remaining players appear afterward.





- $|S| = 0$: player j joins first
  $\Rightarrow$ many permutations $\Rightarrow$ high weight
- $|S| = |P| - 1$: player j joins last
  $\Rightarrow$ many permutations $\Rightarrow$ high weight
- Middle-sized $|S|$: fewer exact matches
  $\Rightarrow$ lower weight
- Result: U-shaped weight distribution

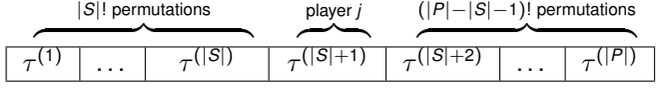
SHAPLEY VALUE - SET DEFINITION

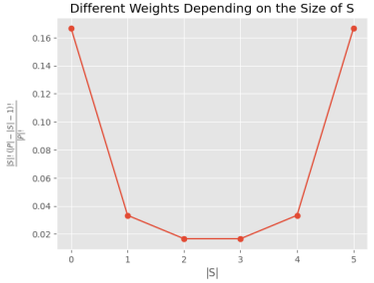


Shapley value via **set definition** (weighting via multinomial coefficient):

$$\phi_j = \sum \frac{|S|!(|P| - |S| - 1)!}{|P|!} (v() - v(S))$$

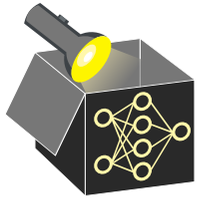
The coefficient gives the probability that, when randomly arranging all $|P|$ players, the exact set S appears before player j, and the remaining players appear afterward.





- $|S| = 0$: player j joins first
  $\Rightarrow$ many permutations $\Rightarrow$ high weight
- $|S| = |P| - 1$: player j joins last
  $\Rightarrow$ many permutations $\Rightarrow$ high weight
- Middle-sized $|S|$: fewer exact matches
  $\Rightarrow$ lower weight
- Result: U-shaped weight distribution

AXIOMS OF FAIR PAYOUTS

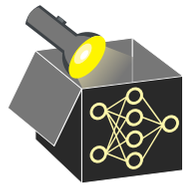


What makes a payout fair? The Shapley value provides a fair payout ϕ_j for each player $j \in P$ and uniquely satisfies the following axioms for any value function v:

- **Efficiency**: Total payout $v(P)$ is fully allocated to players:

$$\sum_{j \in P} \phi_j = v(P)$$

AXIOMS OF FAIR PAYOUTS

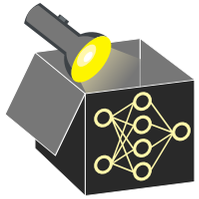


What makes a payout fair? The Shapley value provides a fair payout ϕ_j for each player $j \in P$ and uniquely satisfies the following axioms for any value function v:

- **Efficiency**: Total payout $v(P)$ is fully allocated to players:

$$\sum_{j \in P} \phi_j = v(P)$$

AXIOMS OF FAIR PAYOUTS



What makes a payout fair? The Shapley value provides a fair payout ϕ_j for each player $j \in P$ and uniquely satisfies the following axioms for any value function v:

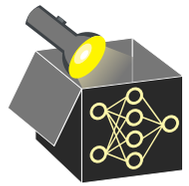
- **Efficiency**: Total payout $v(P)$ is fully allocated to players:

$$\sum_{j \in P} \phi_j = v(P)$$

- **Symmetry**: Indistinguishable players $j, k \in P$ receive equal shares:

$$\text{If } v(S \cup \{j\}) = v(S \cup \{k\}) \text{ for all } S \subseteq P \setminus \{j, k\}, \text{ then } \phi_j = \phi_k$$

AXIOMS OF FAIR PAYOUTS



What makes a payout fair? The Shapley value provides a fair payout ϕ_j for each player $j \in P$ and uniquely satisfies the following axioms for any value function v:

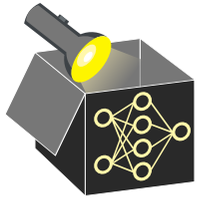
- **Efficiency**: Total payout $v(P)$ is fully allocated to players:

$$\sum_{j \in P} \phi_j = v(P)$$

- **Symmetry**: Indistinguishable players $j, k \in P$ receive equal shares:

$$\text{If } v(S \cup \{j\}) = v(S \cup \{k\}) \text{ for all } S \subseteq P \setminus \{j, k\}, \text{ then } \phi_j = \phi_k$$

AXIOMS OF FAIR PAYOUTS



What makes a payout fair? The Shapley value provides a fair payout ϕ_j for each player $j \in P$ and uniquely satisfies the following axioms for any value function v:

- **Efficiency**: Total payout $v(P)$ is fully allocated to players:

$$\sum_{j \in P} \phi_j = v(P)$$

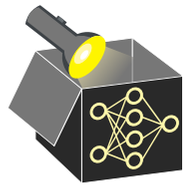
- **Symmetry**: Indistinguishable players $j, k \in P$ receive equal shares:

$$\text{If } v(S \cup \{j\}) = v(S \cup \{k\}) \text{ for all } S \subseteq P \setminus \{j, k\}, \text{ then } \phi_j = \phi_k$$

- **Null Player (Dummy)**: Players who contribute nothing receive nothing:

$$\text{If } v(S \cup \{j\}) = v(S) \text{ for all } S \subseteq P \setminus \{j\}, \text{ then } \phi_j = 0$$

AXIOMS OF FAIR PAYOUTS



What makes a payout fair? The Shapley value provides a fair payout ϕ_j for each player $j \in P$ and uniquely satisfies the following axioms for any value function v:

- **Efficiency**: Total payout $v(P)$ is fully allocated to players:

$$\sum_{j \in P} \phi_j = v(P)$$

- **Symmetry**: Indistinguishable players $j, k \in P$ receive equal shares:

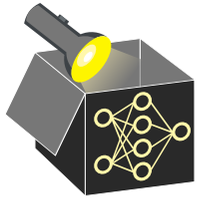
$$\text{If } v(S \cup \{j\}) = v(S \cup \{k\}) \text{ for all } S \subseteq P \setminus \{j, k\}, \text{ then } \phi_j = \phi_k$$

- **Null Player (Dummy)**: Players who contribute nothing receive nothing:

$$\text{If } v(S \cup \{j\}) = v(S) \text{ for all } S \subseteq P \setminus \{j\}, \text{ then } \phi_j = 0$$

Interpretable Machine Learning – 11 / 11

AXIOMS OF FAIR PAYOUTS



What makes a payout fair? The Shapley value provides a fair payout ϕ_j for each player $j \in P$ and uniquely satisfies the following axioms for any value function v:

- **Efficiency**: Total payout $v(P)$ is fully allocated to players:

$$\sum_{j \in P} \phi_j = v(P)$$

- **Symmetry**: Indistinguishable players $j, k \in P$ receive equal shares:

$$\text{If } v(S \cup \{j\}) = v(S \cup \{k\}) \text{ for all } S \subseteq P \setminus \{j, k\}, \text{ then } \phi_j = \phi_k$$

- **Null Player (Dummy)**: Players who contribute nothing receive nothing:

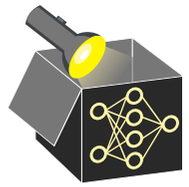
$$\text{If } v(S \cup \{j\}) = v(S) \text{ for all } S \subseteq P \setminus \{j\}, \text{ then } \phi_j = 0$$

- **Additivity**: For two separate games with value functions v_1, v_2, define a combined game with $v(S) = v_1(S) + v_2(S)$ for all $S \subseteq P$. Then:

$$\phi_{j, v_1 + v_2} = \phi_{j, v_1} + \phi_{j, v_2}$$

$\rightsquigarrow$ Payout of combined game = payout of the two separate games

AXIOMS OF FAIR PAYOUTS



What makes a payout fair? The Shapley value provides a fair payout ϕ_j for each player $j \in P$ and uniquely satisfies the following axioms for any value function v:

- **Efficiency**: Total payout $v(P)$ is fully allocated to players:

$$\sum_{j \in P} \phi_j = v(P)$$

- **Symmetry**: Indistinguishable players $j, k \in P$ receive equal shares:

$$\text{If } v(S \cup \{j\}) = v(S \cup \{k\}) \text{ for all } S \subseteq P \setminus \{j, k\}, \text{ then } \phi_j = \phi_k$$

- **Null Player (Dummy)**: Players who contribute nothing receive nothing:

$$\text{If } v(S \cup \{j\}) = v(S) \text{ for all } S \subseteq P \setminus \{j\}, \text{ then } \phi_j = 0$$

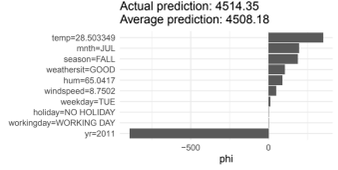
- **Additivity**: For two separate games with value functions v_1, v_2, define a combined game with $v(S) = v_1(S) + v_2(S)$ for all $S \subseteq P$. Then:

$$\phi_{j, v_1 + v_2} = \phi_{j, v_1} + \phi_{j, v_2}$$

$\rightsquigarrow$ Payout of combined game = payout of the two separate games

Interpretable Machine Learning

Shapley Values for Local Explanations

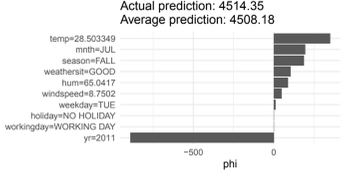


Learning goals

- See model predictions as a cooperative game
- Transfer the Shapley value concept from game theory to machine learning

Interpretable Machine Learning

Shapley
Shapley Values for Local Explanations

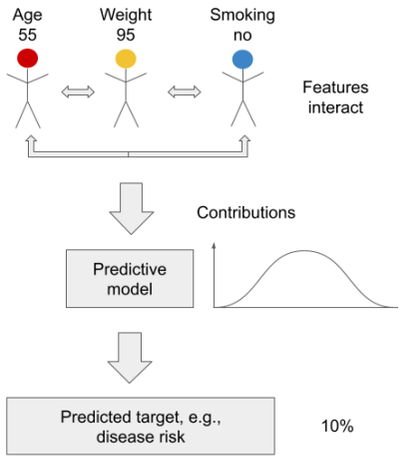


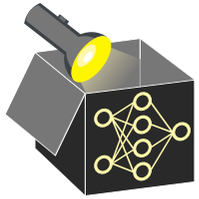
Learning goals

- See model predictions as a cooperative game
- Transfer the Shapley value concept from game theory to machine learning

FROM GAME THEORY TO MACHINE LEARNING

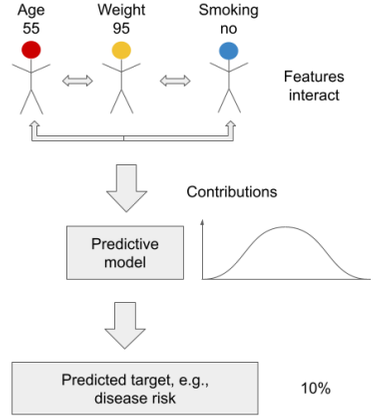
- Model prediction depends on feature interactions for a specific observation
- **Goal:** Decompose prediction into **individual feature contributions**
- **Idea:** Treat features as players jointly producing a prediction
- How to fairly assign credit to features? ⇒ Shapley values

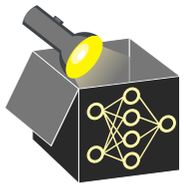




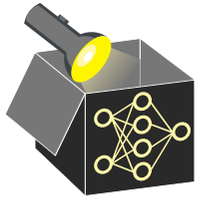
FROM GAME THEORY TO MACHINE LEARNING

- Model prediction depends on feature interactions for a specific observation
- **Goal:** Decompose prediction into **individual feature contributions**
- **Idea:** Treat features as players jointly producing a prediction
- How to fairly assign credit to features? ⇒ Shapley values





FROM GAME THEORY TO MACHINE LEARNING

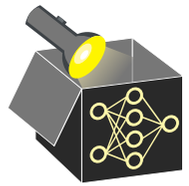


- **Game:** Predict $\hat{f}(x_1, x_2, \ldots, x_p)$ for a single observation $\mathbf{x}$
- **Players:** Features $x_j, j \in \{1, \ldots, p\}$, cooperate to produce a prediction
- **Value function:** Defines payout of coalition $S \subseteq P$ for observation $\mathbf{x}$ by

$$v(S) = \hat{f}_S(\mathbf{x}_S) - \hat{f}_\emptyset, \text{ where}$$

  - $\hat{f}_S : \mathcal{X}_S \mapsto \mathcal{Y}$ is the PD function $\hat{f}_S(\mathbf{x}_S) := \int \hat{f}(\mathbf{x}_S, X_{-S})\, d\mathbb{P}_{X_{-s}}$
    $\rightsquigarrow$ "Removes" features in $-S$ by marginalizing, keeping $\hat{f}$ fixed
  - Mean prediction $\hat{f}_\emptyset := \mathbb{E}_{\mathbf{x}}(\hat{f}(\mathbf{x}))$ is subtracted to ensure $v(\emptyset) = 0$
- **Goal:** Distribute total payout $v(P) = \hat{f}(\mathbf{x}) - \hat{f}_\emptyset$ fairly among features

FROM GAME THEORY TO MACHINE LEARNING

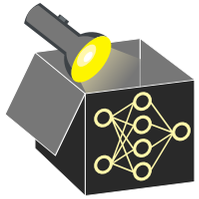


- **Game:** Predict $\hat{f}(x_1, x_2, \ldots, x_p)$ for a single observation $\mathbf{x}$
- **Players:** Features $x_j, j \in \{1, \ldots, p\}$, cooperate to produce a prediction
- **Value function:** Defines payout of coalition $S \subseteq P$ for observation $\mathbf{x}$ by

$$v(S) = \hat{f}_S(\mathbf{x}_S) - \hat{f}_\emptyset, \text{ where}$$

  - $\hat{f}_S : \mathcal{X}_S \mapsto \mathcal{Y}$ is the PD function $\hat{f}_S(\mathbf{x}_S) := \int \hat{f}(\mathbf{x}_S, X_{-S})\, d\mathbb{P}_{X_{-s}}$
    $\rightsquigarrow$ "Removes" features in $-S$ by marginalizing, keeping $\hat{f}$ fixed
  - Mean prediction $\hat{f}_\emptyset := \mathbb{E}_{\mathbf{x}}(\hat{f}(\mathbf{x}))$ is subtracted to ensure $v(\emptyset) = 0$
- **Goal:** Distribute total payout $v(P) = \hat{f}(\mathbf{x}) - \hat{f}_\emptyset$ fairly among features

FROM GAME THEORY TO MACHINE LEARNING



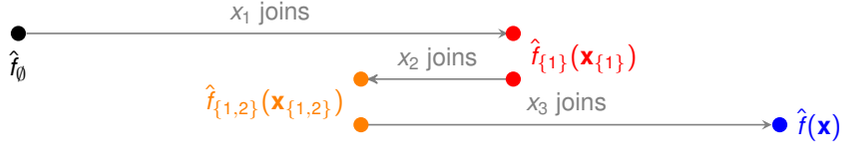
- **Game:** Predict $\hat{f}(x_1, x_2, \ldots, x_p)$ for a single observation $\mathbf{x}$
- **Players:** Features $x_j, j \in \{1, \ldots, p\}$, cooperate to produce a prediction
- **Value function:** Defines payout of coalition $S \subseteq P$ for observation $\mathbf{x}$ by

$$v(S) = \hat{f}_S(\mathbf{x}_S) - \hat{f}_\emptyset, \text{ where}$$

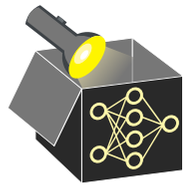
  - $\hat{f}_S : \mathcal{X}_S \mapsto \mathcal{Y}$ is the PD function $\hat{f}_S(\mathbf{x}_S) := \int \hat{f}(\mathbf{x}_S, X_{-S})\, d\mathbb{P}_{X_{-S}}$
    $\rightsquigarrow$ "Removes" features in $-S$ by marginalizing, keeping $\hat{f}$ fixed
  - Mean prediction $\hat{f}_\emptyset := \mathbb{E}_{\mathbf{x}}(\hat{f}(\mathbf{x}))$ is subtracted to ensure $v(\emptyset) = 0$
- **Goal:** Distribute total payout $v(P) = \hat{f}(\mathbf{x}) - \hat{f}_\emptyset$ fairly among features
- **Marginal contribution of feature j joining coalition** S ($\hat{f}_\emptyset$ cancels):

$$\Delta(j, S) = v(S \cup \{j\}) - v(S) = \hat{f}_{S\cup\{j\}}(\mathbf{x}_{S\cup\{j\}}) - \hat{f}_S(\mathbf{x}_S)$$

- **Example (3 features):** Feature contributions for joining order $x_1 \to x_2 \to x_3$ toward total payout $v(P) = \hat{f}(\mathbf{x}) - \hat{f}_\emptyset$, each step reflects a marginal contribution



FROM GAME THEORY TO MACHINE LEARNING



- **Game:** Predict $\hat{f}(x_1, x_2, \ldots, x_p)$ for a single observation $\mathbf{x}$
- **Players:** Features $x_j, j \in \{1, \ldots, p\}$, cooperate to produce a prediction
- **Value function:** Defines payout of coalition $S \subseteq P$ for observation $\mathbf{x}$ by

$$v(S) = \hat{f}_S(\mathbf{x}_S) - \hat{f}_\emptyset, \text{ where}$$

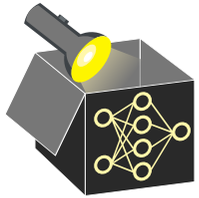
  - $\hat{f}_S : \mathcal{X}_S \mapsto \mathcal{Y}$ is the PD function $\hat{f}_S(\mathbf{x}_S) := \int \hat{f}(\mathbf{x}_S, X_{-S})\, d\mathbb{P}_{X_{-S}}$
    $\rightsquigarrow$ "Removes" features in $-S$ by marginalizing, keeping $\hat{f}$ fixed
  - Mean prediction $\hat{f}_\emptyset := \mathbb{E}_{\mathbf{x}}(\hat{f}(\mathbf{x}))$ is subtracted to ensure $v(\emptyset) = 0$
- **Goal:** Distribute total payout $v(P) = \hat{f}(\mathbf{x}) - \hat{f}_\emptyset$ fairly among features
- **Marginal contribution of feature j joining coalition** S ($\hat{f}_\emptyset$ cancels):

$$\Delta(j, S) = v() - v(S) = \hat{f}(\mathbf{x}) - \hat{f}_S(\mathbf{x}_S)$$

- **Example (3 features):** Feature contributions for joining order $x_1 \to x_2 \to x_3$ toward total payout $v(P) = \hat{f}(\mathbf{x}) - \hat{f}_\emptyset$, each step reflects a marginal contribution

SHAPLEY VALUE - DEFINITION

Shapley (1953) | Strumbelj et al. (2014)



Order definition: Shapley value $\phi_j(\mathbf{x})$ quantifies contribution of x_j via

$$\phi_j(\mathbf{x}) = \frac{1}{|P|!} \sum_{\tau \in \Pi} \underbrace{\hat{f}_{S_j^\tau \cup \{j\}}(\mathbf{x}_{S_j^\tau \cup \{j\}}) - \hat{f}_{S_j^\tau}(\mathbf{x}_{S_j^\tau})}_{\Delta(j, S_j^\tau) \text{ marginal contribution of feature } j}$$

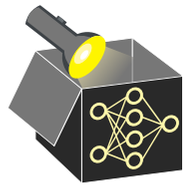
- **Interpretation:** $\phi_j(\mathbf{x})$ quantifies how much feature x_j contributes to the difference between $\hat{f}(\mathbf{x})$ and the mean prediction $\hat{f}_\emptyset$
  $\rightsquigarrow$ Marginal contributions and Shapley values can be negative
- **Exact computation of ϕ_j:** Using PD function $\hat{f}_S(\mathbf{x}_S) = \frac{1}{n} \sum_{i=1}^n \hat{f}(\mathbf{x}_S, \mathbf{x}_{-S}^{(i)})$ yields

$$\phi_j(\mathbf{x}) = \frac{1}{|P|!} \sum_{\tau \in \Pi} \frac{1}{n} \sum_{i=1}^n \hat{f}(\mathbf{x}_{S_j^\tau \cup \{j\}}, \mathbf{x}^{(i)}_{-\{S_j^\tau \cup \{j\}\}}) - \hat{f}(\mathbf{x}_{S_j^\tau}, \mathbf{x}^{(i)}_{-S_j^\tau})$$

  $\rightsquigarrow$ $\hat{f}_S$ marginalizes over all features not in S using all observations $i = 1, \ldots, n$
  $\rightsquigarrow$ Exact computation requires $|P|! \cdot n$ marginal contribution terms

SHAPLEY VALUE - DEFINITION

SHAPLEY_1953 | STRUMBELJ_2014



Order definition: Shapley value $\phi_j(\mathbf{x})$ quantifies contribution of x_j via

$$\phi_j(\mathbf{x}) = \frac{1}{|P|!} \sum_{\tau \in \Pi} \underbrace{\hat{f}_{S_j^\tau \cup \{j\}}(\mathbf{x}_{S_j^\tau \cup \{j\}}) - \hat{f}_{S_j^\tau}(\mathbf{x}_{S_j^\tau})}_{\Delta(j, S_j^\tau) \text{ marginal contribution of feature } j}$$

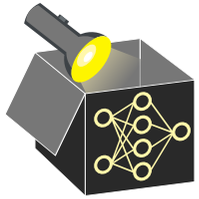
- **Interpretation:** $\phi_j(\mathbf{x})$ quantifies how much feature x_j contributes to the difference between $\hat{f}(\mathbf{x})$ and the mean prediction $\hat{f}_\emptyset$
  $\rightsquigarrow$ Marginal contributions and Shapley values can be negative
- **Exact computation of ϕ_j:** Using PD function $\hat{f}_S(\mathbf{x}_S) = \frac{1}{n} \sum_{i=1}^n \hat{f}(\mathbf{x}_S, \mathbf{x}_{-S}^{(i)})$ yields

$$\phi_j(\mathbf{x}) = \frac{1}{|P|!} \sum_{\tau \in \Pi} \frac{1}{n} \sum_{i=1}^n \hat{f}(\mathbf{x}_{S_j^\tau \cup \{j\}}, \mathbf{x}^{(i)}_{-\{S_j^\tau \cup \{j\}\}}) - \hat{f}(\mathbf{x}_{S_j^\tau}, \mathbf{x}^{(i)}_{-S_j^\tau})$$

  $\rightsquigarrow$ $\hat{f}_S$ marginalizes over all features not in S using all obs. $i = 1, \ldots, n$
  $\rightsquigarrow$ Exact computation requires $|P|! \cdot n$ marginal contribution terms

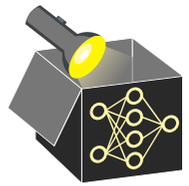
ESTIMATION: A PRACTICAL PROBLEM

- **Exact computation is infeasible for many features:**
  For $|P| = 10$, the number of permutations is $10! \approx 3.6$ million
  $\rightsquigarrow$ Complexity grows factorially with feature count

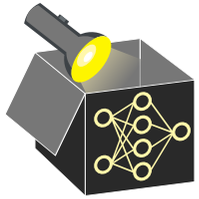


ESTIMATION: A PRACTICAL PROBLEM

- **Exact computation is infeasible for many features:**
  For $|P| = 10$, the number of permutations is $10! \approx 3.6$ million
  $\rightsquigarrow$ Complexity grows factorially with feature count

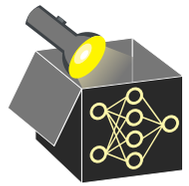


ESTIMATION: A PRACTICAL PROBLEM



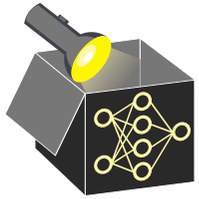
- **Exact computation is infeasible for many features:**
  For $|P| = 10$, the number of permutations is $10! \approx 3.6$ million
  $\rightsquigarrow$ Complexity grows factorially with feature count
- **Additional challenge: Estimating marginal predictions (PD functions)**
  Each permutation τ defines a coalition S_j^τ needing its own estimate of $\hat{f}_{S_j^\tau}(\mathbf{x}_{S_j^\tau})$
  $\rightsquigarrow$ With $|P|!$ permutations and n data points, the number of such estimates grows rapidly, making marginalization costly

ESTIMATION: A PRACTICAL PROBLEM



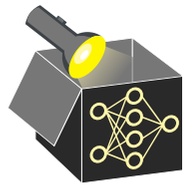
- **Exact computation is infeasible for many features:**
  For $|P| = 10$, the number of permutations is $10! \approx 3.6$ million
  $\rightsquigarrow$ Complexity grows factorially with feature count
- **Additional challenge: Estimating marginal predictions (PD funcs)**
  Each permut. τ defines a coal. S_j^τ needing its own estimate of $\hat{f}_{S_j^\tau}(\mathbf{x}_{S_j^\tau})$
  $\rightsquigarrow$ With $|P|!$ permutations and n data points, the number of such estimates grows rapidly, making marginalization costly

ESTIMATION: A PRACTICAL PROBLEM



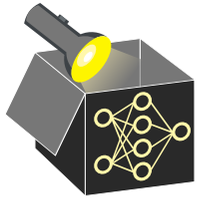
- **Exact computation is infeasible for many features:**
  For $|P| = 10$, the number of permutations is $10! \approx 3.6$ million
  $\rightsquigarrow$ Complexity grows factorially with feature count
- **Additional challenge: Estimating marginal predictions (PD functions)**
  Each permutation τ defines a coalition S_j^τ needing its own estimate of $\hat{f}_{S_j^\tau}(\mathbf{x}_{S_j^\tau})$
  $\rightsquigarrow$ With $|P|!$ permutations and n data points, the number of such estimates grows rapidly, making marginalization costly
- **Solution: Sampling-based approximation**
  Instead of computing $|P|! \cdot n$ terms, we approximate using M random samples of permutations τ and data points

ESTIMATION: A PRACTICAL PROBLEM



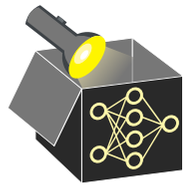
- **Exact computation is infeasible for many features:**
  For $|P| = 10$, the number of permutations is $10! \approx 3.6$ million
  $\rightsquigarrow$ Complexity grows factorially with feature count
- **Additional challenge: Estimating marginal predictions (PD funcs)**
  Each permut. τ defines a coal. S_j^τ needing its own estimate of $\hat{f}_{S_j^\tau}(\mathbf{x}_{S_j^\tau})$
  $\rightsquigarrow$ With $|P|!$ permutations and n data points, the number of such estimates grows rapidly, making marginalization costly
- **Solution: Sampling-based approximation**
  Instead of computing $|P|! \cdot n$ terms, we approximate using M random samples of permutations τ and data points

ESTIMATION: A PRACTICAL PROBLEM



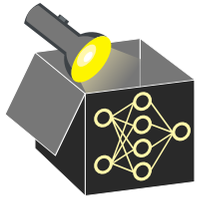
- **Exact computation is infeasible for many features:**
  For $|P| = 10$, the number of permutations is $10! \approx 3.6$ million
  $\rightsquigarrow$ Complexity grows factorially with feature count
- **Additional challenge: Estimating marginal predictions (PD functions)**
  Each permutation τ defines a coalition S_j^τ needing its own estimate of $\hat{f}_{S_j^\tau}(\mathbf{x}_{S_j^\tau})$
  $\rightsquigarrow$ With $|P|!$ permutations and n data points, the number of such estimates grows rapidly, making marginalization costly
- **Solution: Sampling-based approximation**
  Instead of computing $|P|! \cdot n$ terms, we approximate using M random samples of permutations τ and data points
- **Tradeoff: Accuracy vs. Efficiency**
  Larger M improves Shapley approximation
  $\rightsquigarrow$ Higher cost, but better fidelity to the exact value

ESTIMATION: A PRACTICAL PROBLEM



- **Exact computation is infeasible for many features:**
  For $|P| = 10$, the number of permutations is $10! \approx 3.6$ million
  $\rightsquigarrow$ Complexity grows factorially with feature count
- **Additional challenge: Estimating marginal predictions (PD funcs)**
  Each permut. τ defines a coal. S_j^τ needing its own estimate of $\hat{f}_{S_j^\tau}(\mathbf{x}_{S_j^\tau})$
  $\rightsquigarrow$ With $|P|!$ permutations and n data points, the number of such estimates grows rapidly, making marginalization costly
- **Solution: Sampling-based approximation**
  Instead of computing $|P|! \cdot n$ terms, we approximate using M random samples of permutations τ and data points
- **Tradeoff: Accuracy vs. Efficiency**
  Larger M improves Shapley approximation
  $\rightsquigarrow$ Higher cost, but better fidelity to the exact value

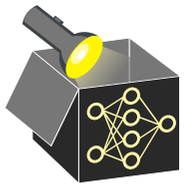
APPROXIMATION ALGORITHM ▸ Strumbelj et al. (2014)



Estimate Shapley value ϕ_j of observation **x** for feature *j*:

- **Input:** **x** obs. of interest, *j* feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, *M* iterations

APPROXIMATION ALGORITHM ▸ STRUMBELJ_2014

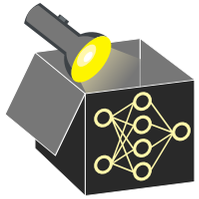


Estimate Shapley value ϕ_j of observation **x** for feature *j*:

- **Input:** **x** obs. of interest, *j* feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, *M* iterations

APPROXIMATION ALGORITHM

Strumbelj et al. (2014)



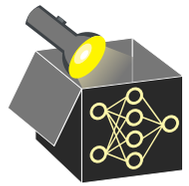
Estimate Shapley value ϕ_j of observation **x** for feature *j*:

- **Input:** **x** obs. of interest, *j* feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, *M* iterations

1. For $m = 1, \ldots, M$ **do**:

APPROXIMATION ALGORITHM

STRUMBELJ_2014



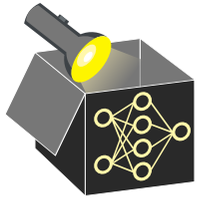
Estimate Shapley value ϕ_j of observation **x** for feature *j*:

- **Input:** **x** obs. of interest, *j* feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, *M* iterations

1. For $m = 1, \ldots, M$ **do**:

APPROXIMATION ALGORITHM

Strumbelj et al. (2014)



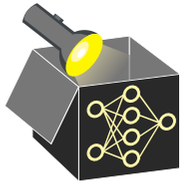
Estimate Shapley value ϕ_j of observation **x** for feature *j*:

- **Input:** **x** obs. of interest, *j* feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, *M* iterations

1. For $m = 1, \ldots, M$ **do**:
   1. Sample random permutation $\tau = (\tau^{(1)}, \ldots, \tau^{(p)}) \in \Pi$ of feature indices

APPROXIMATION ALGORITHM

STRUMBELJ_2014



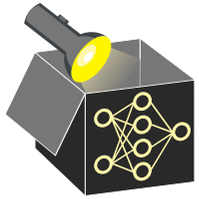
Estimate Shapley value ϕ_j of observation **x** for feature *j*:

- **Input:** **x** obs. of interest, *j* feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, *M* iterations

1. For $m = 1, \ldots, M$ **do**:
   1. Sample random permut. $\tau = (\tau^{(1)}, \ldots, \tau^{(p)}) \in \Pi$ of feature indices

APPROXIMATION ALGORITHM

Strumbelj et al. (2014)



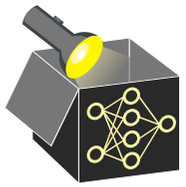
Estimate Shapley value ϕ_j of observation **x** for feature j:

- **Input:** **x** obs. of interest, j feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, M iterations

1. For $m = 1, \ldots, M$ **do**:
   1. Sample random permutation $\tau = (\tau^{(1)}, \ldots, \tau^{(p)}) \in \Pi$ of feature indices
   2. Let coalition $S_m := S_j^\tau$ be the set of features preceding j in τ

APPROXIMATION ALGORITHM

STRUMBELJ_2014

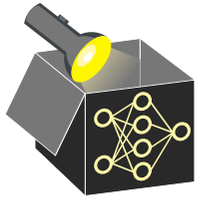


Estimate Shapley value ϕ_j of observation **x** for feature j:

- **Input:** **x** obs. of interest, j feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, M iterations

1. For $m = 1, \ldots, M$ **do**:
   1. Sample random permut. $\tau = (\tau^{(1)}, \ldots, \tau^{(p)}) \in \Pi$ of feature indices
   2. Let coalition $S_m := S_j^\tau$ be the set of features preceding j in τ

APPROXIMATION ALGORITHM ▸ Strumbelj et al. (2014)

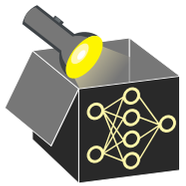


Estimate Shapley value ϕ_j of observation **x** for feature j:

- **Input:** **x** obs. of interest, j feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, M iterations

1. For $m = 1, \ldots, M$ **do**:
   1. Sample random permutation $\tau = (\tau^{(1)}, \ldots, \tau^{(p)}) \in \Pi$ of feature indices
   2. Let coalition $S_m := S_j^\tau$ be the set of features preceding j in τ
   3. Sample random data point $\mathbf{z}^{(m)} \in \mathcal{D}$ (so-called background data)

APPROXIMATION ALGORITHM ▸ STRUMBELJ_2014

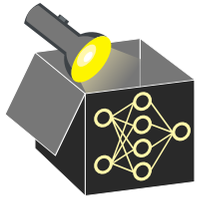


Estimate Shapley value ϕ_j of observation **x** for feature j:

- **Input:** **x** obs. of interest, j feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, M iterations

1. For $m = 1, \ldots, M$ **do**:
   1. Sample random permut. $\tau = (\tau^{(1)}, \ldots, \tau^{(p)}) \in \Pi$ of feature indices
   2. Let coalition $S_m := S_j^\tau$ be the set of features preceding j in τ
   3. Sample random data point $\mathbf{z}^{(m)} \in \mathcal{D}$ (so-called background data)

APPROXIMATION ALGORITHM ▸ Strumbelj et al. (2014)

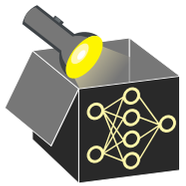


Estimate Shapley value ϕ_j of observation $\mathbf{x}$ for feature j:

- **Input:** $\mathbf{x}$ obs. of interest, j feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, M iterations

1. For $m = 1, \ldots, M$ **do**:
   1. Sample random permutation $\tau = (\tau^{(1)}, \ldots, \tau^{(p)}) \in \Pi$ of feature indices
   2. Let coalition $S_m := S_j^\tau$ be the set of features preceding j in τ
   3. Sample random data point $\mathbf{z}^{(m)} \in \mathcal{D}$ (so-called background data)
   4. Construct two hybrid observations by combining values from $\mathbf{x}$ and $\mathbf{z}^{(m)}$:

APPROXIMATION ALGORITHM ▸ STRUMBELJ_2014



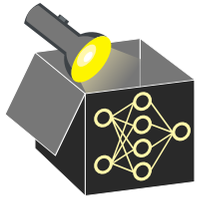
Estimate Shapley value ϕ_j of observation $\mathbf{x}$ for feature j:

- **Input:** $\mathbf{x}$ obs. of interest, j feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, M iterations

1. For $m = 1, \ldots, M$ **do**:
   1. Sample random permut. $\tau = (\tau^{(1)}, \ldots, \tau^{(p)}) \in \Pi$ of feature indices
   2. Let coalition $S_m := S_j^\tau$ be the set of features preceding j in τ
   3. Sample random data point $\mathbf{z}^{(m)} \in \mathcal{D}$ (so-called background data)
   4. Construct two hybrid obs. by combining values from $\mathbf{x}$ and $\mathbf{z}^{(m)}$:

APPROXIMATION ALGORITHM

Strumbelj et al. (2014)



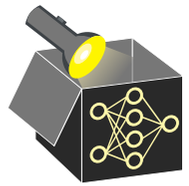
Estimate Shapley value ϕ_j of observation $\mathbf{x}$ for feature j:

- **Input:** $\mathbf{x}$ obs. of interest, j feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, M iterations

1. For $m = 1, \ldots, M$ **do**:
   1. Sample random permutation $\tau = (\tau^{(1)}, \ldots, \tau^{(p)}) \in \Pi$ of feature indices
   2. Let coalition $S_m := S_j^\tau$ be the set of features preceding j in τ
   3. Sample random data point $\mathbf{z}^{(m)} \in \mathcal{D}$ (so-called background data)
   4. Construct two hybrid observations by combining values from $\mathbf{x}$ and $\mathbf{z}^{(m)}$:
      - $\mathbf{x}_{+j}^{(m)} = (x_{\tau(1)}, \ldots, x_{\tau(|S_m|)}, x_j, z^{(m)}_{\tau(|S_m|+2)}, \ldots, z^{(m)}_{\tau(p)})$
        $\rightsquigarrow$ includes $\mathbf{x}_{S_m \cup \{j\}}$ (features in $S_m \cup \{j\}$ from $\mathbf{x}$), rest from $\mathbf{z}^{(m)}$

APPROXIMATION ALGORITHM

STRUMBELJ_2014



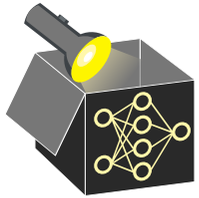
Estimate Shapley value ϕ_j of observation $\mathbf{x}$ for feature j:

- **Input:** $\mathbf{x}$ obs. of interest, j feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, M iterations

1. For $m = 1, \ldots, M$ **do**:
   1. Sample random permut. $\tau = (\tau^{(1)}, \ldots, \tau^{(p)}) \in \Pi$ of feature indices
   2. Let coalition $S_m := S_j^\tau$ be the set of features preceding j in τ
   3. Sample random data point $\mathbf{z}^{(m)} \in \mathcal{D}$ (so-called background data)
   4. Construct two hybrid obs. by combining values from $\mathbf{x}$ and $\mathbf{z}^{(m)}$:
      - $\mathbf{x}_{+j}^{(m)} = (x_{\tau(1)}, \ldots, x_{\tau(|S_m|)}, x_j, z^{(m)}_{\tau(|S_m|+2)}, \ldots, z^{(m)}_{\tau(p)})$
        $\rightsquigarrow$ includes $\mathbf{x}_{S_m \cup \{j\}}$ (features in $S_m \cup \{j\}$ from $\mathbf{x}$), rest from $\mathbf{z}^{(m)}$

APPROXIMATION ALGORITHM

Strumbelj et al. (2014)



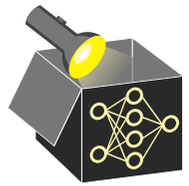
Estimate Shapley value ϕ_j of observation $\mathbf{x}$ for feature j:

- **Input:** $\mathbf{x}$ obs. of interest, j feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, M iterations

1. For $m = 1, \ldots, M$ **do**:
   1. Sample random permutation $\tau = (\tau^{(1)}, \ldots, \tau^{(p)}) \in \Pi$ of feature indices
   2. Let coalition $S_m := S_j^\tau$ be the set of features preceding j in τ
   3. Sample random data point $\mathbf{z}^{(m)} \in \mathcal{D}$ (so-called background data)
   4. Construct two hybrid observations by combining values from $\mathbf{x}$ and $\mathbf{z}^{(m)}$:
      - $\mathbf{x}_{+j}^{(m)} = (x_{\tau(1)}, \ldots, x_{\tau(|S_m|)}, x_j, z^{(m)}_{\tau(|S_m|+2)}, \ldots, z^{(m)}_{\tau(p)})$
        $\rightsquigarrow$ includes $\mathbf{x}_{S_m \cup \{j\}}$ (features in $S_m \cup \{j\}$ from $\mathbf{x}$), rest from $\mathbf{z}^{(m)}$
      - $\mathbf{x}_{-j}^{(m)} = (x_{\tau(1)}, \ldots, x_{\tau(|S_m|)}, z_j^{(m)}, z^{(m)}_{\tau(|S_m|+2)}, \ldots, z^{(m)}_{\tau(p)})$
        $\rightsquigarrow$ includes $\mathbf{x}_{S_m}$ (features in S_m excl. x_j from $\mathbf{x}$), rest from $\mathbf{z}^{(m)}$

APPROXIMATION ALGORITHM

STRUMBELJ_2014



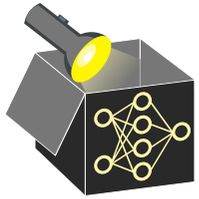
Estimate Shapley value ϕ_j of observation $\mathbf{x}$ for feature j:

- **Input:** $\mathbf{x}$ obs. of interest, j feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, M iterations

1. For $m = 1, \ldots, M$ **do**:
   1. Sample random permut. $\tau = (\tau^{(1)}, \ldots, \tau^{(p)}) \in \Pi$ of feature indices
   2. Let coalition $S_m := S_j^\tau$ be the set of features preceding j in τ
   3. Sample random data point $\mathbf{z}^{(m)} \in \mathcal{D}$ (so-called background data)
   4. Construct two hybrid obs. by combining values from $\mathbf{x}$ and $\mathbf{z}^{(m)}$:
      - $\mathbf{x}_{+j}^{(m)} = (x_{\tau(1)}, \ldots, x_{\tau(|S_m|)}, x_j, z^{(m)}_{\tau(|S_m|+2)}, \ldots, z^{(m)}_{\tau(p)})$
        $\rightsquigarrow$ includes $\mathbf{x}_{S_m \cup \{j\}}$ (features in $S_m \cup \{j\}$ from $\mathbf{x}$), rest from $\mathbf{z}^{(m)}$
      - $\mathbf{x}_{-j}^{(m)} = (x_{\tau(1)}, \ldots, x_{\tau(|S_m|)}, z_j^{(m)}, z^{(m)}_{\tau(|S_m|+2)}, \ldots, z^{(m)}_{\tau(p)})$
        $\rightsquigarrow$ includes $\mathbf{x}_{S_m}$ (features in S_m excl. x_j from $\mathbf{x}$), rest from $\mathbf{z}^{(m)}$

APPROXIMATION ALGORITHM

Strumbelj et al. (2014)



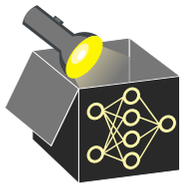
Estimate Shapley value ϕ_j of observation $\mathbf{x}$ for feature j:

- **Input:** $\mathbf{x}$ obs. of interest, j feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, M iterations

1. For $m = 1, \ldots, M$ **do**:
   1. Sample random permutation $\tau = (\tau^{(1)}, \ldots, \tau^{(p)}) \in \Pi$ of feature indices
   2. Let coalition $S_m := S_j^\tau$ be the set of features preceding j in τ
   3. Sample random data point $\mathbf{z}^{(m)} \in \mathcal{D}$ (so-called background data)
   4. Construct two hybrid observations by combining values from $\mathbf{x}$ and $\mathbf{z}^{(m)}$:
      - $\mathbf{x}_{+j}^{(m)} = (x_{\tau(1)}, \ldots, x_{\tau(|S_m|)}, x_j, z^{(m)}_{\tau(|S_m|+2)}, \ldots, z^{(m)}_{\tau(p)})$
        $\rightsquigarrow$ includes $\mathbf{x}_{S_m \cup \{j\}}$ (features in $S_m \cup \{j\}$ from $\mathbf{x}$), rest from $\mathbf{z}^{(m)}$
      - $\mathbf{x}_{-j}^{(m)} = (x_{\tau(1)}, \ldots, x_{\tau(|S_m|)}, z^{(m)}_j, z^{(m)}_{\tau(|S_m|+2)}, \ldots, z^{(m)}_{\tau(p)})$
        $\rightsquigarrow$ includes $\mathbf{x}_{S_m}$ (features in S_m excl. x_j from $\mathbf{x}$), rest from $\mathbf{z}^{(m)}$
   5. Compute marginal contribution $\Delta(j, S_m) = \hat{f}(\mathbf{x}_{+j}^{(m)}) - \hat{f}(\mathbf{x}_{-j}^{(m)})$

APPROXIMATION ALGORITHM

STRUMBELJ_2014



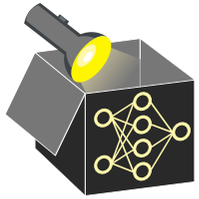
Estimate Shapley value ϕ_j of observation $\mathbf{x}$ for feature j:

- **Input:** $\mathbf{x}$ obs. of interest, j feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, M iterations

1. For $m = 1, \ldots, M$ **do**:
   1. Sample random permut. $\tau = (\tau^{(1)}, \ldots, \tau^{(p)}) \in \Pi$ of feature indices
   2. Let coalition $S_m := S_j^\tau$ be the set of features preceding j in τ
   3. Sample random data point $\mathbf{z}^{(m)} \in \mathcal{D}$ (so-called background data)
   4. Construct two hybrid obs. by combining values from $\mathbf{x}$ and $\mathbf{z}^{(m)}$:
      - $\mathbf{x}_{+j}^{(m)} = (x_{\tau(1)}, \ldots, x_{\tau(|S_m|)}, x_j, z^{(m)}_{\tau(|S_m|+2)}, \ldots, z^{(m)}_{\tau(p)})$
        $\rightsquigarrow$ includes $\mathbf{x}_{S_m \cup \{j\}}$ (features in $S_m \cup \{j\}$ from $\mathbf{x}$), rest from $\mathbf{z}^{(m)}$
      - $\mathbf{x}_{-j}^{(m)} = (x_{\tau(1)}, \ldots, x_{\tau(|S_m|)}, z^{(m)}_j, z^{(m)}_{\tau(|S_m|+2)}, \ldots, z^{(m)}_{\tau(p)})$
        $\rightsquigarrow$ includes $\mathbf{x}_{S_m}$ (features in S_m excl. x_j from $\mathbf{x}$), rest from $\mathbf{z}^{(m)}$
   5. Compute marginal contribution $\Delta(j, S_m) = \hat{f}(\mathbf{x}_{+j}^{(m)}) - \hat{f}(\mathbf{x}_{-j}^{(m)})$

APPROXIMATION ALGORITHM

Strumbelj et al. (2014)



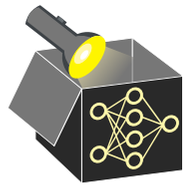
Estimate Shapley value ϕ_j of observation $\mathbf{x}$ for feature j:

- **Input:** $\mathbf{x}$ obs. of interest, j feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, M iterations

1. For $m = 1, \ldots, M$ **do**:
   1. Sample random permutation $\tau = (\tau^{(1)}, \ldots, \tau^{(p)}) \in \Pi$ of feature indices
   2. Let coalition $S_m := S_j^\tau$ be the set of features preceding j in τ
   3. Sample random data point $\mathbf{z}^{(m)} \in \mathcal{D}$ (so-called background data)
   4. Construct two hybrid observations by combining values from $\mathbf{x}$ and $\mathbf{z}^{(m)}$:
      - $\mathbf{x}_{+j}^{(m)} = (x_{\tau(1)}, \ldots, x_{\tau(|S_m|)}, x_j, z^{(m)}_{\tau(|S_m|+2)}, \ldots, z^{(m)}_{\tau(p)})$
        $\rightsquigarrow$ includes $\mathbf{x}_{S_m \cup \{j\}}$ (features in $S_m \cup \{j\}$ from $\mathbf{x}$), rest from $\mathbf{z}^{(m)}$
      - $\mathbf{x}_{-j}^{(m)} = (x_{\tau(1)}, \ldots, x_{\tau(|S_m|)}, z_j^{(m)}, z^{(m)}_{\tau(|S_m|+2)}, \ldots, z^{(m)}_{\tau(p)})$
        $\rightsquigarrow$ includes $\mathbf{x}_{S_m}$ (features in S_m excl. x_j from $\mathbf{x}$), rest from $\mathbf{z}^{(m)}$
   5. Compute marginal contribution $\Delta(j, S_m) = \hat{f}(\mathbf{x}_{+j}^{(m)}) - \hat{f}(\mathbf{x}_{-j}^{(m)})$
2. Compute Shapley value $\phi_j = \frac{1}{M} \sum_{m=1}^{M} \Delta(j, S_m)$

APPROXIMATION ALGORITHM

STRUMBELJ_2014



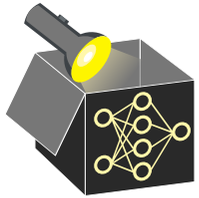
Estimate Shapley value ϕ_j of observation $\mathbf{x}$ for feature j:

- **Input:** $\mathbf{x}$ obs. of interest, j feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, M iterations

1. For $m = 1, \ldots, M$ **do**:
   1. Sample random permut. $\tau = (\tau^{(1)}, \ldots, \tau^{(p)}) \in \Pi$ of feature indices
   2. Let coalition $S_m := S_j^\tau$ be the set of features preceding j in τ
   3. Sample random data point $\mathbf{z}^{(m)} \in \mathcal{D}$ (so-called background data)
   4. Construct two hybrid obs. by combining values from $\mathbf{x}$ and $\mathbf{z}^{(m)}$:
      - $\mathbf{x}_{+j}^{(m)} = (x_{\tau(1)}, \ldots, x_{\tau(|S_m|)}, x_j, z^{(m)}_{\tau(|S_m|+2)}, \ldots, z^{(m)}_{\tau(p)})$
        $\rightsquigarrow$ includes $\mathbf{x}_{S_m \cup \{j\}}$ (features in $S_m \cup \{j\}$ from $\mathbf{x}$), rest from $\mathbf{z}^{(m)}$
      - $\mathbf{x}_{-j}^{(m)} = (x_{\tau(1)}, \ldots, x_{\tau(|S_m|)}, z_j^{(m)}, z^{(m)}_{\tau(|S_m|+2)}, \ldots, z^{(m)}_{\tau(p)})$
        $\rightsquigarrow$ includes $\mathbf{x}_{S_m}$ (features in S_m excl. x_j from $\mathbf{x}$), rest from $\mathbf{z}^{(m)}$
   5. Compute marginal contribution $\Delta(j, S_m) = \hat{f}(\mathbf{x}_{+j}^{(m)}) - \hat{f}(\mathbf{x}_{-j}^{(m)})$
2. Compute Shapley value $\phi_j = \frac{1}{M} \sum_{m=1}^{M} \Delta(j, S_m)$

APPROXIMATION ALGORITHM

Strumbelj et al. (2014)



Estimate Shapley value ϕ_j of observation $\mathbf{x}$ for feature j:

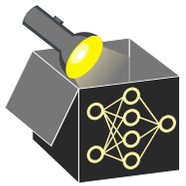
- **Input:** $\mathbf{x}$ obs. of interest, j feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, M iterations

1. For $m = 1, \ldots, M$ **do**:
   1. Sample random permutation $\tau = (\tau^{(1)}, \ldots, \tau^{(p)}) \in \Pi$ of feature indices
   2. Let coalition $S_m := S_j^\tau$ be the set of features preceding j in τ
   3. Sample random data point $\mathbf{z}^{(m)} \in \mathcal{D}$ (so-called background data)
   4. Construct two hybrid observations by combining values from $\mathbf{x}$ and $\mathbf{z}^{(m)}$:
      - $\mathbf{x}_{+j}^{(m)} = (x_{\tau(1)}, \ldots, x_{\tau(|S_m|)}, x_j, z^{(m)}_{\tau(|S_m|+2)}, \ldots, z^{(m)}_{\tau(p)})$
        $\rightsquigarrow$ includes $\mathbf{x}_{S_m \cup \{j\}}$ (features in $S_m \cup \{j\}$ from $\mathbf{x}$), rest from $\mathbf{z}^{(m)}$
      - $\mathbf{x}_{-j}^{(m)} = (x_{\tau(1)}, \ldots, x_{\tau(|S_m|)}, z_j^{(m)}, z^{(m)}_{\tau(|S_m|+2)}, \ldots, z^{(m)}_{\tau(p)})$
        $\rightsquigarrow$ includes $\mathbf{x}_{S_m}$ (features in S_m excl. x_j from $\mathbf{x}$), rest from $\mathbf{z}^{(m)}$
   5. Compute marginal contribution $\Delta(j, S_m) = \hat{f}(\mathbf{x}_{+j}^{(m)}) - \hat{f}(\mathbf{x}_{-j}^{(m)})$
2. Compute Shapley value $\phi_j = \frac{1}{M} \sum_{m=1}^{M} \Delta(j, S_m)$

$\rightsquigarrow$ Over M iterations, the PD functions $\hat{f}_{S_m}(\mathbf{x}_{S_m})$ and $\hat{f}_{S_m \cup \{j\}}(\mathbf{x}_{S_m \cup \{j\}})$ are approximated by $\hat{f}(\mathbf{x}_{-j}^{(m)})$ and $\hat{f}(\mathbf{x}_{+j}^{(m)})$, where features not in the coalition (to be marginalized) are imputed with values from the random data points $\mathbf{z}^{(m)}$

APPROXIMATION ALGORITHM

STRUMBELJ_2014



Estimate Shapley value ϕ_j of observation $\mathbf{x}$ for feature j:

- **Input:** $\mathbf{x}$ obs. of interest, j feat. of interest, $\hat{f}$ model, $\mathcal{D}$ data, M iterations

1. For $m = 1, \ldots, M$ **do**:
   1. Sample random permut. $\tau = (\tau^{(1)}, \ldots, \tau^{(p)}) \in \Pi$ of feature indices
   2. Let coalition $S_m := S_j^\tau$ be the set of features preceding j in τ
   3. Sample random data point $\mathbf{z}^{(m)} \in \mathcal{D}$ (so-called background data)
   4. Construct two hybrid obs. by combining values from $\mathbf{x}$ and $\mathbf{z}^{(m)}$:
      - $\mathbf{x}_{+j}^{(m)} = (x_{\tau(1)}, \ldots, x_{\tau(|S_m|)}, x_j, z^{(m)}_{\tau(|S_m|+2)}, \ldots, z^{(m)}_{\tau(p)})$
        $\rightsquigarrow$ includes $\mathbf{x}_{S_m \cup \{j\}}$ (features in $S_m \cup \{j\}$ from $\mathbf{x}$), rest from $\mathbf{z}^{(m)}$
      - $\mathbf{x}_{-j}^{(m)} = (x_{\tau(1)}, \ldots, x_{\tau(|S_m|)}, z_j^{(m)}, z^{(m)}_{\tau(|S_m|+2)}, \ldots, z^{(m)}_{\tau(p)})$
        $\rightsquigarrow$ includes $\mathbf{x}_{S_m}$ (features in S_m excl. x_j from $\mathbf{x}$), rest from $\mathbf{z}^{(m)}$
   5. Compute marginal contribution $\Delta(j, S_m) = \hat{f}(\mathbf{x}_{+j}^{(m)}) - \hat{f}(\mathbf{x}_{-j}^{(m)})$
2. Compute Shapley value $\phi_j = \frac{1}{M} \sum_{m=1}^{M} \Delta(j, S_m)$

$\rightsquigarrow$ Over M iterations, the PD functions $\hat{f}_{S_m}(\mathbf{x}_{S_m})$ and $\hat{f}_{S_m \cup \{j\}}(\mathbf{x}_{S_m \cup \{j\}})$ are approximated by $\hat{f}(\mathbf{x}_{-j}^{(m)})$ and $\hat{f}(\mathbf{x}_{+j}^{(m)})$, where features not in the coalition (to be marginalized) are imputed with vals from random data points $\mathbf{z}^{(m)}$

SHAPLEY VALUE APPROXIMATION - ILLUSTRATION

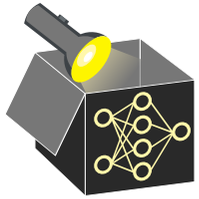
Definition

x: obs. of interest

x with feature values in $\mathbf{x}_{S_m}$ (other are replaced)

$$\phi_j(\mathbf{x}) = \frac{1}{M}\sum_{m=1}^{M}\left[\hat{f}(\mathbf{x}_{+j}{}^{(m)}) - \hat{f}(\mathbf{x}_{-j}{}^{(m)})\right]$$

x with feature values in $\mathbf{x}_{S_m \cup \{j\}}$



	Temperature	Humidity	Windspeed	Year
x	10.66	56	11	2012
x_{+j}	10.66	56	$random: z^{[m]}_{windspeed}$	2012
x_{-j}	10.66	56	$random: z^{[m]}_{windspeed}$	$random: z^{[m]}_{year}$

j

SHAPLEY VALUE APPROX. - ILLUSTRATION

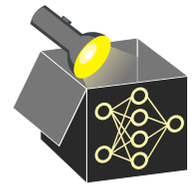
Definition

x: obs. of interest

x with feature values in $\mathbf{x}_{S_m}$ (other are replaced)

$$\phi_j(\mathbf{x}) = \frac{1}{M}\sum_{m=1}^{M}\left[\hat{f}(\mathbf{x}_{+j}{}^{(m)}) - \hat{f}(\mathbf{x}_{-j}{}^{(m)})\right]$$

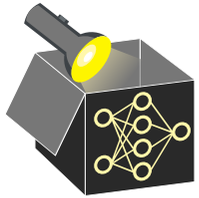
x with feature values in $\mathbf{x}_{S_m \cup \{j\}}$



	Temperature	Humidity	Windspeed	Year
x	10.66	56	11	2012
x_{+j}	10.66	56	$random: z^{[m]}_{windspeed}$	2012
x_{-j}	10.66	56	$random: z^{[m]}_{windspeed}$	$random: z^{[m]}_{year}$

j

SHAPLEY VALUE APPROXIMATION - ILLUSTRATION



Definition

Contribution of feature j to coalition S_m

$$\phi_j(\mathbf{x}) = \frac{1}{M}\sum_{m=1}^{M}\underbrace{\left[\hat{f}(\mathbf{x}_{+j}^{(m)}) - \hat{f}(\mathbf{x}_{-j}^{(m)})\right]}_{:=\Delta(j,S_m)}$$

- $\Delta(j, S_m) = \hat{f}(\mathbf{x}_{+j}^{(m)}) - \hat{f}(\mathbf{x}_{-j}^{(m)})$ is marginal contribution of feature j to coalition S_m
- Here: Feature *year* contributes +700 bike rentals if it joins coalition $S_m = \{temp, hum\}$

	Temperature	Humidity	Windspeed	Year	Count
x	10.66	56	11	2012	5600
x_{+j}	10.66	56	$random: z_{windspeed}^{[m]}$	2012	4900
x_{-j}	10.66	56	$random: z_{windspeed}^{[m]}$	$random: z_{year}^{[m]}$	700

j | $\hat{f}$ | $\Delta(j, S_m)$ marginal contribution

SHAPLEY VALUE APPROX. - ILLUSTRATION

Definition

Contribution of feature j to coalition S_m

$$\phi_j(\mathbf{x}) = \frac{1}{M}\sum_{m=1}^{M}\underbrace{\left[\hat{f}(\mathbf{x}_{+j}^{(m)}) - \hat{f}(\mathbf{x}_{-j}^{(m)})\right]}_{:=\Delta(j,S_m)}$$

- $\Delta(j, S_m) = \hat{f}(\mathbf{x}_{+j}^{(m)}) - \hat{f}(\mathbf{x}_{-j}^{(m)})$ is marginal contribution of feature j to coalition S_m
- Here: Feature *year* contributes +700 bike rentals if it joins coalition $S_m = \{temp, hum\}$

	Temperature	Humidity	Windspeed	Year	Count
x	10.66	56	11	2012	5600
x_{+j}	10.66	56	$random: z_{windspeed}^{[m]}$	2012	4900
x_{-j}	10.66	56	$random: z_{windspeed}^{[m]}$	$random: z_{year}^{[m]}$	700

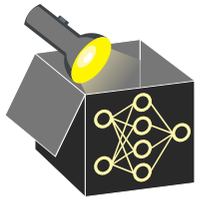
j | $\hat{f}$ | $\Delta(j, S_m)$ marginal contribution

SHAPLEY VALUE APPROXIMATION - ILLUSTRATION

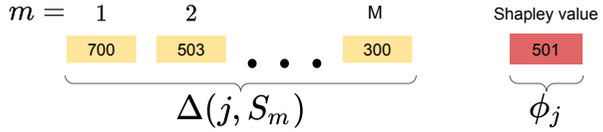
Definition

average the contributions of feature j

$$\phi_j(\mathbf{x}) = \frac{1}{M}\sum_{m=1}^{M}\left[\hat{f}(\mathbf{x}_{+j}^{(m)}) - \hat{f}(\mathbf{x}_{-j}^{(m)})\right]$$



- Compute marginal contribution of feature j towards the prediction across all randomly drawn feature coalitions $S_1, \ldots, S_m$
- Average all M marginal contributions of feature j
- Shapley value ϕ_j is the payout of feature j, i.e., how much feature *year* contributed to the overall prediction in bicycle counts of a specific observation **x**

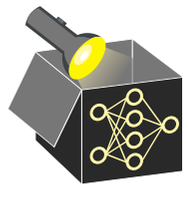


SHAPLEY VALUE APPROX. - ILLUSTRATION

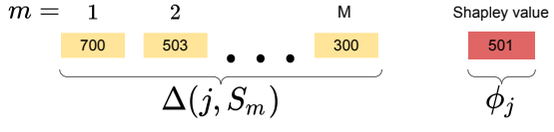
Definition

average the contributions of feature j

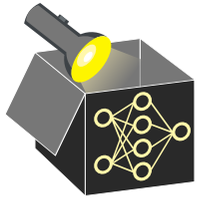
$$\phi_j(\mathbf{x}) = \frac{1}{M}\sum_{m=1}^{M}\left[\hat{f}(\mathbf{x}_{+j}^{(m)}) - \hat{f}(\mathbf{x}_{-j}^{(m)})\right]$$



- Compute marginal contribution of feature j towards the prediction across all randomly drawn feature coalitions $S_1, \ldots, S_m$
- Average all M marginal contributions of feature j
- Shapley value ϕ_j is the payout of feature j, i.e., how much feature *year* contributed to the overall prediction in bicycle counts of a specific obs. **x**



REVISITED: AXIOMS FOR FAIR ATTRIBUTIONS



We adapt the classic Shapley axioms to the setting of model predictions:

- **Efficiency**: Sum of Shapley values adds up to the centered prediction:

$$\sum_{j=1}^{p} \phi_j(\mathbf{x}) = \hat{f}(\mathbf{x}) - \mathbb{E}_{\mathbf{x}}[\hat{f}(\mathbf{x})]$$

  $\rightsquigarrow$ All predictive contribution is fully distributed among features
- **Symmetry**: Identical contributors receive equal value:

$$\hat{f}_{S\cup\{j\}}(\mathbf{x}_{S\cup\{j\}}) = \hat{f}_{S\cup\{k\}}(\mathbf{x}_{S\cup\{k\}}) \; \forall S \subseteq P \setminus \{j,k\} \Rightarrow \phi_j = \phi_k$$

  $\rightsquigarrow$ Interaction effects are shared equitably
- **Dummy (Null Player)**: Irrelevant features receive zero attribution:

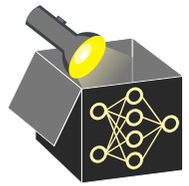
$$\hat{f}_{S\cup\{j\}}(\mathbf{x}_{S\cup\{j\}}) = \hat{f}_S(\mathbf{x}_S) \; \forall S \subseteq P \Rightarrow \phi_j = 0$$

  $\rightsquigarrow$ Shapley value is zero for unused features (e.g., trees or LASSO)
- **Additivity**: Attributions are additive across models:

$$\phi_j(v_1 + v_2) = \phi_j(v_1) + \phi_j(v_2)$$

  $\rightsquigarrow$ Enables combining Shapley values for model ensembles

REVISITED: AXIOMS FOR FAIR ATTRIBUTIONS



We adapt the classic Shapley axioms to the setting of model predictions:

- **Efficiency**: Sum of Shapley values adds up to the centered prediction:

$$\sum_{j=1}^{p} \phi_j(\mathbf{x}) = \hat{f}(\mathbf{x}) - \mathbb{E}_{\mathbf{x}}[\hat{f}(\mathbf{x})]$$

  $\rightsquigarrow$ All predictive contribution is fully distributed among features
- **Symmetry**: Identical contributors receive equal value:

$$\hat{f}(\mathbf{x}) = \hat{f}(\mathbf{x}) \; \forall S \subseteq P \setminus \{j,k\} \Rightarrow \phi_j = \phi_k$$

  $\rightsquigarrow$ Interaction effects are shared equitably
- **Dummy (Null Player)**: Irrelevant features receive zero attribution:

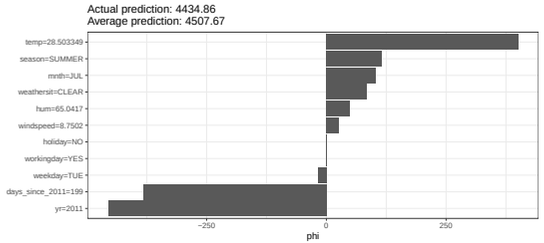
$$\hat{f}(\mathbf{x}) = \hat{f}_S(\mathbf{x}_S) \; \forall S \subseteq P \Rightarrow \phi_j = 0$$

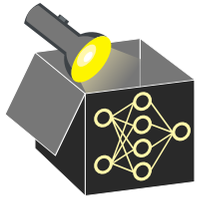
  $\rightsquigarrow$ Shapley value is zero for unused features (e.g., trees or LASSO)
- **Additivity**: Attributions are additive across models:

$$\phi_j(v_1 + v_2) = \phi_j(v_1) + \phi_j(v_2)$$

  $\rightsquigarrow$ Enables combining Shapley values for model ensembles

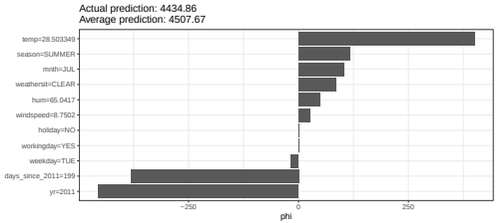
BIKE SHARING DATASET

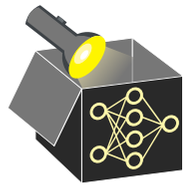




- Shapley decomposition for a single prediction in bike sharing dataset
- Model prediction: $\hat{f}(\mathbf{x}^{(200)}) = \mathbf{4434.86}$ vs. dataset average: $\mathbb{E}_{\mathbf{x}}[\hat{f}(\mathbf{x})] = \mathbf{4507.67}$
- Total feature attribution: $\sum_j \phi_j = \mathbf{-72.81}$
  $\rightsquigarrow$ Explain downward shift from mean prediction
- Temperature (with value 28.5°C) is the strongest positive contributor: $+400$
- Features `yr = 2011` and `days_since_2011 = 199` strongly reduce prediction
  $\rightsquigarrow$ Model captures lower bike demand in 2011 compared to 2012

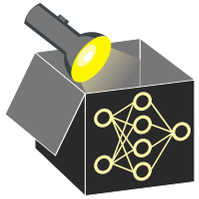
BIKE SHARING DATASET





- Shapley decomposition for a single prediction in bike sharing dataset
- Model pred.: $\hat{f}(\mathbf{x}^{(200)}) = \mathbf{4434.86}$ vs. dataset avg.: $\mathbb{E}_{\mathbf{x}}[\hat{f}(\mathbf{x})] = \mathbf{4507.67}$
- Total feature attribution: $\sum_j \phi_j = \mathbf{-72.81}$
  $\rightsquigarrow$ Explain downward shift from mean prediction
- Temperature (with value 28.5°C) – strongest positive contributor: $+400$
- `yr = 2011` and `days_since_2011 = 199` strongly reduce prediction
  $\rightsquigarrow$ Model captures lower bike demand in 2011 compared to 2012

ADVANTAGES AND DISADVANTAGES



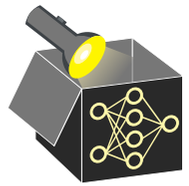
Advantages:

- **Strong theoretical foundation** from cooperative game theory
- **Fair attribution:** Prediction is additively distributed across features
  $\rightsquigarrow$ Easy to interpret for users
- **Contrastive explanations:** Quantify each feature's role in deviating from the average prediction

Disadvantages:

- **Computational cost:** Exact computation scales factorially with feature count
  $\rightsquigarrow$ Without sampling, all 2^p coalitions (or $p!$ permutations) must be evaluated
- **Issue with correlated features:** Shapley values may evaluate the model on feature combinations that do not occur in the real data

ADVANTAGES AND DISADVANTAGES



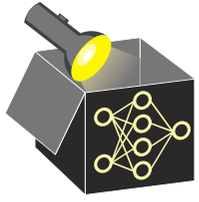
Advantages:

- **Strong theoretical foundation** from cooperative game theory
- **Fair attribution:** Prediction is additively distributed across features
  $\rightsquigarrow$ Easy to interpret for users
- **Contrastive explanations:** Quantify each feature's role in deviating from the average prediction

Disadvantages:

- **Comput. cost:** Exact computation scales factorially with feature count
  $\rightsquigarrow$ Without sampling, all 2^p coalitions (or $p!$ permuts) must be evaluated
- **Issue with correlated features:** Shapley values may evaluate the model on feature combinations that do not occur in the real data

Interpretable Machine Learning

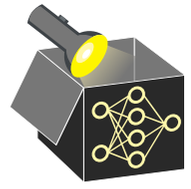


SHAP (SHapley Additive exPlanation) Values

Learning goals

- Recall order- and set-based definitions of Shapley values in ML
- Interpret predictions via additive Shapley decomposition
- Understand SHAP as surrogate-based model
- Understand SHAP properties

Interpretable Machine Learning

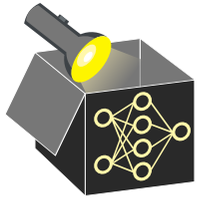


Shapley
SHAP (SHapley Additive exPlanation)

Learning goals

- Recall order- and set-based definitions of Shapley values in ML
- Interpret predictions via additive Shapley decomposition
- Understand SHAP as surrogate-based model
- Understand SHAP properties

SHAPLEY VALUES IN ML - A SHORT RECAP

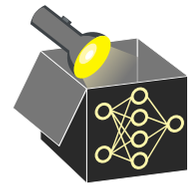


Shapley values (order definition): Average over marginal contributions across all permutations of feature indices $\tau \in \Pi$:

$$\phi_j(\mathbf{x}) = \frac{1}{p!} \sum_{\tau \in \Pi} \underbrace{\hat{f}_{S_j^\tau \cup \{j\}}(\mathbf{x}_{S_j^\tau \cup \{j\}}) - \hat{f}_{S_j^\tau}(\mathbf{x}_{S_j^\tau})}_{\text{marginal contribution of feature } j}$$

- For each permutation τ, determine coalition S_j^τ: features before j in τ
- In $\hat{f}_S$, features not in S are marginalized (e.g., replaced by random imputations)
- Compute marginal contribution of adding j to S_j^τ via the difference above
- Average over all $p!$ permutations (in practice, over $M << p!$)

SHAPLEY VALUES IN ML - A SHORT RECAP

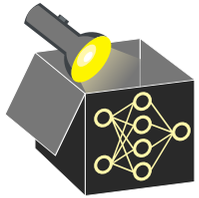


Shapley values (order definition): Average over marginal contributions across all permutations of feature indices $\tau \in \Pi$:

$$\phi_j(\mathbf{x}) = \frac{1}{p!} \sum_{\tau \in \Pi} \underbrace{\hat{f}(\mathbf{x}) - \hat{f}(\mathbf{x})}_{\text{marginal contribution of feature } j}$$

- For each permutation τ, determine coalition : features before j in τ
- In $\hat{f}_S$, features not in S are marginalized (e.g., randomly imputed)
- Compute marginal contribution of adding j to via the difference above
- Average over all $p!$ permutations (in practice, over $M << p!$)

SHAPLEY VALUES IN ML - A SHORT RECAP



Shapley values (order definition): Average over marginal contributions across all permutations of feature indices $\tau \in \Pi$:

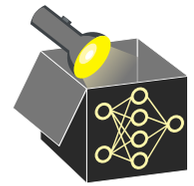
$$\phi_j(\mathbf{x}) = \frac{1}{p!} \sum_{\tau \in \Pi} \underbrace{\hat{f}_{S_j^\tau \cup \{j\}}(\mathbf{x}_{S_j^\tau \cup \{j\}}) - \hat{f}_{S_j^\tau}(\mathbf{x}_{S_j^\tau})}_{\text{marginal contribution of feature } j}$$

- For each permutation τ, determine coalition S_j^τ: features before j in τ
- In $\hat{f}_S$, features not in S are marginalized (e.g., replaced by random imputations)
- Compute marginal contribution of adding j to S_j^τ via the difference above
- Average over all $p!$ permutations (in practice, over $M << p!$)

Alternative (set definition): Average marginal contribution over all subsets, weighted by their relative number of appearances in permutations:

$$\phi_j(\mathbf{x}) = \sum_{S \subseteq \{1,\ldots,p\} \setminus \{j\}} \frac{|S|!(p-|S|-1)!}{p!} \left[\hat{f}_{S \cup \{j\}}(\mathbf{x}_{S \cup \{j\}}) - \hat{f}_S(\mathbf{x}_S) \right].$$

SHAPLEY VALUES IN ML - A SHORT RECAP



Shapley values (order definition): Average over marginal contributions across all permutations of feature indices $\tau \in \Pi$:

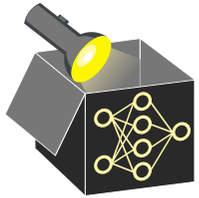
$$\phi_j(\mathbf{x}) = \frac{1}{p!} \sum_{\tau \in \Pi} \underbrace{\hat{f}(\mathbf{x}) - \hat{f}(\mathbf{x})}_{\text{marginal contribution of feature } j}$$

- For each permutation τ, determine coalition : features before j in τ
- In $\hat{f}_S$, features not in S are marginalized (e.g., randomly imputed)
- Compute marginal contribution of adding j to via the difference above
- Average over all $p!$ permutations (in practice, over $M << p!$)

Alternative (set definition): Average marginal contribution over all subsets, weighted by their relative number of appearances in permutations:

$$\phi_j(\mathbf{x}) = \sum_{S \subseteq \{1,\ldots,p\} \setminus \{j\}} \frac{|S|!(p-|S|-1)!}{p!} \left[\hat{f}_{S \cup \{j\}}(\mathbf{x}_{S \cup \{j\}}) - \hat{f}_S(\mathbf{x}_S) \right].$$

SHAPLEY VALUES IN ML - EXAMPLE



Example (Bike sharing data):

- Train random forest using humidity (hum), temperature (temp), windspeed (ws)
- Consider observation of interest $\mathbf{x}$ with prediction $\hat{f}(\mathbf{x}) = 2573$
- Mean prediction $\mathbb{E}_{\mathbf{x}}[\hat{f}(\mathbf{x})] = 4515$
- Compute exact Shapley value for $\mathbf{x}$ for feature hum:

S	$S \cup \{j\}$	$\hat{f}_S$	$\hat{f}_{S\cup\{j\}}$	weight
$\varnothing$	hum	4515	4635	$2/6$
temp	temp, hum	3087	3060	$1/6$
ws	ws, hum	4359	4450	$1/6$
temp, ws	temp, ws, hum	2623	2573	$2/6$

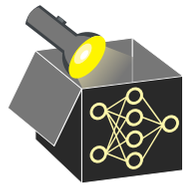
$\Rightarrow \phi_{\text{hum}}(\mathbf{x}) = \frac{2}{6}(4635-4515)+\frac{1}{6}(3060-3087)+\frac{1}{6}(4450-4359)+\frac{2}{6}(2573-2623) = 34$

$\Rightarrow$ Analogously $\phi_{\text{temp}}(\mathbf{x}) = -1654$, $\phi_{\text{ws}}(\mathbf{x}) = -322$

SHAPLEY VALUES IN ML - EXAMPLE

Example (Bike sharing data):

- Train random forest using humidity (hum), temperature (temp), windspeed (ws)
- Consider observation of interest $\mathbf{x}$ with prediction $\hat{f}(\mathbf{x}) = 2573$
- Mean prediction $\mathbb{E}_{\mathbf{x}}[\hat{f}(\mathbf{x})] = 4515$
- Compute exact Shapley value for $\mathbf{x}$ for feature hum:



S	$S \cup \{j\}$	$\hat{f}_S$	$\hat{f}_{S\cup\{j\}}$	weight
$\varnothing$	hum	4515	4635	$2/6$
temp	temp, hum	3087	3060	$1/6$
ws	ws, hum	4359	4450	$1/6$
temp, ws	temp, ws, hum	2623	2573	$2/6$

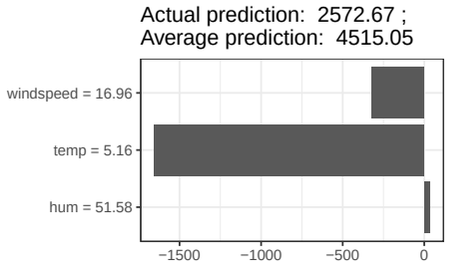
$\Rightarrow \phi_{\text{hum}}(\mathbf{x}) = \frac{2}{6}(4635-4515)+\frac{1}{6}(3060-3087)+\frac{1}{6}(4450-4359)+\frac{2}{6}(2573-2623) = 34$

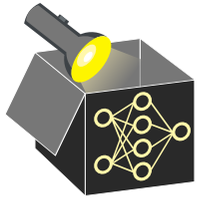
$\Rightarrow$ Analogously $\phi_{\text{temp}}(\mathbf{x}) = -1654$, $\phi_{\text{ws}}(\mathbf{x}) = -322$

FROM SHAPLEY VALUES TO SHAP

Shapley value interpretation (for x):

- `hum` $(+34)$ pushes prediction *above* baseline (= average prediction).
- `temp` (-1654) and `ws` (-322) pull prediction *below* baseline.
- Together, they explain full deviation from average prediction.

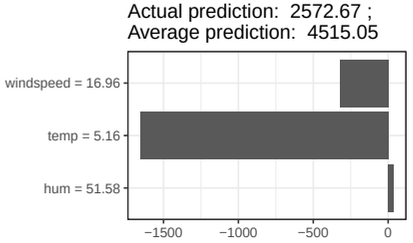


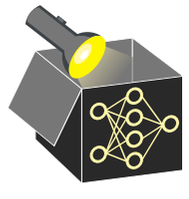


FROM SHAPLEY VALUES TO SHAP

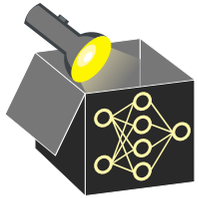
Shapley value interpretation (for x):

- `hum` $(+34)$ pushes pred. *above* baseline (= average prediction).
- `temp` (-1654) and `ws` (-322) pull prediction *below* baseline.
- Together, they explain full deviation from average prediction.



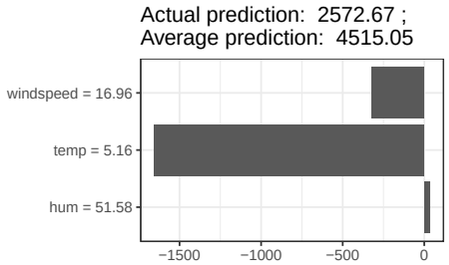


FROM SHAPLEY VALUES TO SHAP



Shapley value interpretation (for x):

- `hum` $(+34)$ pushes prediction *above* baseline (= average prediction).
- `temp` (-1654) and `ws` (-322) pull prediction *below* baseline.
- Together, they explain full deviation from average prediction.



Shapley-based additive decomposition of prediction for **x** gives insights on how features shift prediction from baseline $\mathbb{E}(\hat{f})$:

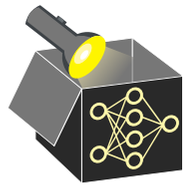
$$\underbrace{\hat{f}(\mathbf{x})}_{\text{actual prediction}} = \underbrace{\phi_0}_{\mathbb{E}_{\mathbf{X}}[\hat{f}(\mathbf{X})]} + \sum_{j \in \{\text{hum,temp,ws}\}} \phi_j(\mathbf{x})$$

$$2573 = 4515 + \left(34 - 1654 - 322\right) = 4515 - 1942$$

$\rightsquigarrow$ Like a LM evaluated at **x**: global intercept ϕ_0 plus per-feature contributions $\phi_j(\mathbf{x})$.

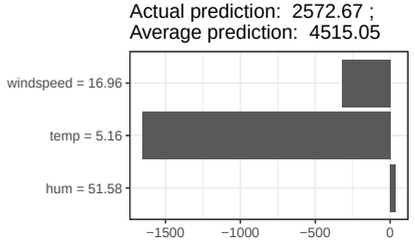
SHAP Motivation: Can we efficiently estimate this Shapley-based additive decomposition of $\hat{f}(\mathbf{x})$ using a surrogate model (while preserving Shapley axioms)?

FROM SHAPLEY VALUES TO SHAP



Shapley value interpretation (for x):

- `hum` $(+34)$ pushes pred. *above* baseline (= average prediction).
- `temp` (-1654) and `ws` (-322) pull prediction *below* baseline.
- Together, they explain full deviation from average prediction.



Shapley-based additive decomposition of prediction for **x** gives insights on how features shift prediction from baseline $\mathbb{E}(\hat{f})$:

$$\underbrace{\hat{f}(\mathbf{x})}_{\text{actual prediction}} = \underbrace{\phi_0}_{\mathbb{E}_{\mathbf{X}}[\hat{f}(\mathbf{X})]} + \sum_{j \in \{\text{hum,temp,ws}\}} \phi_j(\mathbf{x})$$

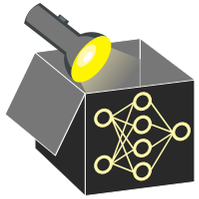
$$2573 = 4515 + \left(34 - 1654 - 322\right) = 4515 - 1942$$

$\rightsquigarrow$ Like a LM evaluated at **x**: global intercept ϕ_0 plus per-feature contribs $\phi_j(\mathbf{x})$.

SHAP Motivation: Can we efficiently estimate this Shapley-based additive decomp. of $\hat{f}(\mathbf{x})$ via a surrogate model (while preserving Shapley axioms)?

SHAP FRAMEWORK

▸ Lundberg et al. 2017



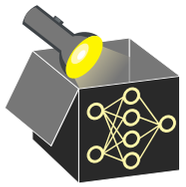
SHAP expresses the prediction of **x** as a sum of contributions from each feature:

$$g(\mathbf{z}') = \phi_0 + \sum_{j=1}^{p} \phi_j z_j'$$

- $\mathbf{z}' \in \{0,1\}^p$: simplified binary input referring to a coalition (coalition vector)
- $z_j' = 1$: feature j is "present" $\Rightarrow$ use x_j in model evaluation
- $z_j' = 0$: feature j is "absent"
  $\Rightarrow$ influence of x_j is removed via marginalization over a reference distribution

SHAP FRAMEWORK

▸ LUNDBERG_2017



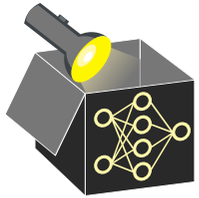
SHAP expresses the prediction of **x** as a sum of contribs from each feature:

$$g(\mathbf{z}') = \phi_0 + \sum_{j=1}^{p} \phi_j z_j'$$

- $\mathbf{z}' \in \{0,1\}^p$: simplified binary input referring to a coalition (coal. vector)
- $z_j' = 1$: feature j is "present" $\Rightarrow$ use x_j in model evaluation
- $z_j' = 0$: feature j is "absent"
  $\Rightarrow$ influence of x_j is removed via marginalization over a reference distrib.

SHAP FRAMEWORK

Lundberg et al. 2017



SHAP expresses the prediction of $\mathbf{x}$ as a sum of contributions from each feature:

$$g(\mathbf{z}') = \phi_0 + \sum_{j=1}^{p} \phi_j z_j'$$

- $\mathbf{z}' \in \{0,1\}^p$: simplified binary input referring to a coalition (coalition vector)
- $z_j' = 1$: feature j is "present" $\Rightarrow$ use x_j in model evaluation
- $z_j' = 0$: feature j is "absent"
  $\Rightarrow$ influence of x_j is removed via marginalization over a reference distribution

SHAP as a theoretical framework: Fit a surrogate model $g(\mathbf{z}')$ satisfying Shapley axioms and recovering $\hat{f}(\mathbf{x})$ when all features are "present":

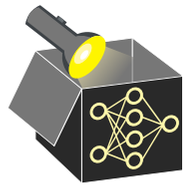
$$\hat{f}(\mathbf{x}) = g(\mathbf{1}) = \phi_0 + \sum_{j=1}^{p} \phi_j$$

Evaluation of $g(\mathbf{z}')$: Let $S = \{j : z_j' = 1\}$ be the active coalition. Then:

- $g(\mathbf{z}') \approx \mathbb{E}[\hat{f}(\mathbf{X}) \mid \mathbf{X}_S = \mathbf{x}_S]$ (conditional expectation)
- $g(\mathbf{z}') \approx \mathbb{E}_{\mathbf{X}_{-S}}[\hat{f}(\mathbf{x}_S, \mathbf{X}_{-S})]$ (marginal expectation, i.e., PD function)
- *Note: Practical implementations (e.g., KernelSHAP) use the marginal expectation, approximated via random imputations from background data.*

SHAP FRAMEWORK

LUNDBERG_2017



SHAP expresses the prediction of $\mathbf{x}$ as a sum of contribs from each feature:

$$g(\mathbf{z}') = \phi_0 + \sum_{j=1}^{p} \phi_j z_j'$$

- $\mathbf{z}' \in \{0,1\}^p$: simplified binary input referring to a coalition (coal. vector)
- $z_j' = 1$: feature j is "present" $\Rightarrow$ use x_j in model evaluation
- $z_j' = 0$: feature j is "absent"
  $\Rightarrow$ influence of x_j is removed via marginalization over a reference distrib.

SHAP as a theoretical framework: Fit a surrogate model $g(\mathbf{z}')$ satisfying Shapley axioms and recovering $\hat{f}(\mathbf{x})$ when all features are "present":

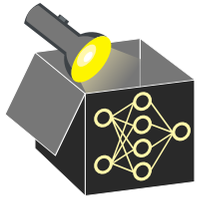
$$\hat{f}(\mathbf{x}) = g(\mathbf{1}) = \phi_0 + \sum_{j=1}^{p} \phi_j$$

Evaluation of $g(\mathbf{z}')$: Let $S = \{j : z_j' = 1\}$ be the active coalition. Then:

- $g(\mathbf{z}') \approx \mathbb{E}[\hat{f}(\mathbf{X}) \mid \mathbf{X}_S = \mathbf{x}_S]$ (conditional expectation)
- $g(\mathbf{z}') \approx \mathbb{E}_{\mathbf{X}_{-S}}[\hat{f}(\mathbf{x}_S, \mathbf{X}_{-S})]$ (marginal expectation, i.e., PD function)
- *Note: Practical implementations (e.g., KernelSHAP) use the marginal expectation, approximated via random imputations from background data.*

SHAP FRAMEWORK

Lundberg et al. 2017



SHAP defines an additive surrogate $g(\mathbf{z}')$ over a binary simplified input $\mathbf{z}' \in \{0,1\}^p$:

$$g\left(\mathbf{z}'^{(k)}\right) = \phi_0 + \sum_{j=1}^{p} \phi_j z_j'^{(k)}$$

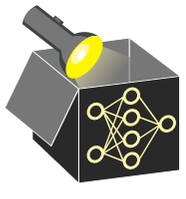
$\mathbf{z}'^{(k)}$: **coalition vector** subset of features

ϕ_0: **baseline** $\mathbb{E}[\hat{f}(\mathbf{X})]$

ϕ_j: **feature attribution** marginal effect of j in coalition

SHAP FRAMEWORK

LUNDBERG_2017



SHAP defines an additive surrogate $g(\mathbf{z}')$ over a binary input $\mathbf{z}' \in \{0,1\}^p$:

$$g\left(\mathbf{z}'^{(k)}\right) = \phi_0 + \sum_{j=1}^{p} \phi_j z_j'^{(k)}$$

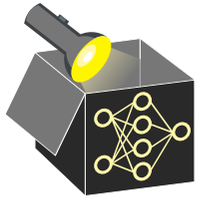
$\mathbf{z}'^{(k)}$: **coalition vector** subset of features

ϕ_0: **baseline** $\mathbb{E}[\hat{f}()]$

ϕ_j: **feature attribution** marginal effect of j in coalition

SHAP FRAMEWORK

Lundberg et al. 2017



SHAP defines an additive surrogate $g(\mathbf{z}')$ over a binary simplified input $\mathbf{z}' \in \{0,1\}^p$:

$g(\mathbf{z}'^{(k)})$: **approx. prediction** for coalition

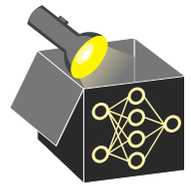
$$g\left(\mathbf{z}'^{(k)}\right) = \underbrace{\phi_0 + \sum_{j=1}^{p} \phi_j z_j'^{(k)}}_{\textbf{Additive Feature Attribution}}$$

ϕ_j: **Shapley value**

Next: How do we estimate the Shapley values ϕ_j efficiently?

SHAP FRAMEWORK

LUNDBERG_2017



SHAP defines an additive surrogate $g(\mathbf{z}')$ over a binary input $\mathbf{z}' \in \{0,1\}^p$:

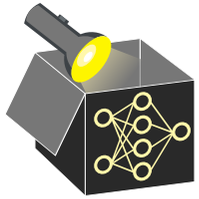
$g(\mathbf{z}'^{(k)})$: **approx. prediction** for coalition

$$g\left(\mathbf{z}'^{(k)}\right) = \underbrace{\phi_0 + \sum_{j=1}^{p} \phi_j z_j'^{(k)}}_{\textbf{Additive Feature Attribution}}$$

ϕ_j: **Shapley value**

Next: How do we estimate the Shapley values ϕ_j efficiently?

PROPERTIES



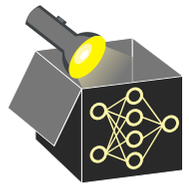
Local Accuracy

$$\hat{f}(\mathbf{x}) = g(\mathbf{z}') = \phi_0 + \sum_{j=1}^{p} \phi_j z_j'$$

Intuition: If the coalition includes all features ($\mathbf{z}' = (z_1', \ldots, z_p')^\top = (1, \ldots, 1)^\top$), the attributions ϕ_j and the baseline ϕ_0 sum up to the original model output $\hat{f}(\mathbf{x})$

Local accuracy corresponds to the **axiom of efficiency** in Shapley game theory

PROPERTIES



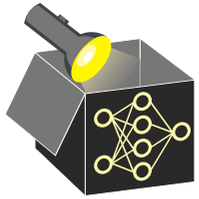
Local Accuracy

$$\hat{f}(\mathbf{x}) = g(\mathbf{z}') = \phi_0 + \sum_{j=1}^{p} \phi_j z_j'$$

Intuition: If coalition includes all features ($\mathbf{z}' = (z_1', \ldots, z_p')^\top = (1, \ldots, 1)^\top$), the attributions ϕ_j and the baseline ϕ_0 sum up to the original model output $\hat{f}(\mathbf{x})$

Local accuracy corresponds to **axiom of efficiency** in Shapley game theory

PROPERTIES



Local Accuracy

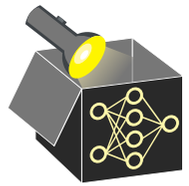
$$\hat{f}(\mathbf{x}) = g(\mathbf{z}') = \phi_0 + \sum_{j=1}^{p} \phi_j z_j'$$

Missingness

$$z_j' = 0 \implies \phi_j = 0$$

Intuition: A "missing" feature (whose value is imputed) gets an attribution of zero

PROPERTIES



Local Accuracy

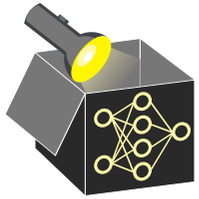
$$\hat{f}(\mathbf{x}) = g(\mathbf{z}') = \phi_0 + \sum_{j=1}^{p} \phi_j z_j'$$

Missingness

$$z_j' = 0 \implies \phi_j = 0$$

Intuition: A "missing" feature (whose value is imputed) gets zero attribution

PROPERTIES



Local Accuracy

$$\hat{f}(\mathbf{x}) = g(\mathbf{z}') = \phi_0 + \sum_{j=1}^{p} \phi_j z_j'$$

Missingness

$$z_j' = 0 \implies \phi_j = 0$$

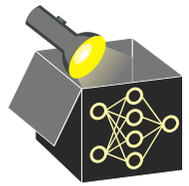
Consistency (Let $\mathbf{z}_{-j}^{\prime(k)}$ refer to $z_j^{\prime(k)} = 0$)
For any two models $\hat{f}$ and $\hat{f}'$, if for all inputs $\mathbf{z}^{\prime(k)} \in \{0,1\}^p$

$$\hat{f}_x'\left(\mathbf{z}^{\prime(k)}\right) - \hat{f}_x'\left(\mathbf{z}_{-j}^{\prime(k)}\right) \geq \hat{f}_x\left(\mathbf{z}^{\prime(k)}\right) - \hat{f}_x\left(\mathbf{z}_{-j}^{\prime(k)}\right) \implies \phi_j\left(\hat{f}', \mathbf{x}\right) \geq \phi_j(\hat{f}, \mathbf{x})$$

Intution: If a model changes so that the marginal contribution of a feature value increases or stays the same, the Shapley value also increases or stays the same

From **consistency** the Shapley **axioms of additivity, dummy and symmetry** follow

PROPERTIES



Local Accuracy

$$\hat{f}(\mathbf{x}) = g(\mathbf{z}') = \phi_0 + \sum_{j=1}^{p} \phi_j z_j'$$

Missingness

$$z_j' = 0 \implies \phi_j = 0$$

Consistency (Let $\mathbf{z}_{-j}^{\prime(k)}$ refer to $z_j^{\prime(k)} = 0$)
For any two models $\hat{f}$ and $\hat{f}'$, if for all inputs $\mathbf{z}^{\prime(k)} \in \{0,1\}^p$

$$\hat{f}_x'\left(\mathbf{z}^{\prime(k)}\right) - \hat{f}_x'\left(\mathbf{z}_{-j}^{\prime(k)}\right) \geq \hat{f}_x\left(\mathbf{z}^{\prime(k)}\right) - \hat{f}_x\left(\mathbf{z}_{-j}^{\prime(k)}\right) \implies \phi_j\left(\hat{f}', \mathbf{x}\right) \geq \phi_j(\hat{f}, \mathbf{x})$$

Intution: If a model changes so that the marginal contribution of a feature value increases or stays the same, the Shapley value also increases or stays the same

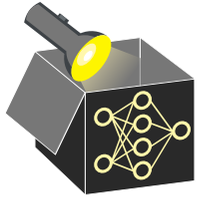
Consistency implies Shapley's axioms of **additivity, dummy, symmetry**.

Interpretable Machine Learning

SHAP (SHapley Additive exPlanation) Values

Learning goals

- Understand KernelSHAP as weighted least-squares regression over coalitions
- Grasp how background samples impute "absent" features
- Observational vs. interventional SHAP

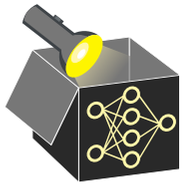


Interpretable Machine Learning

Shapley
Kernel SHAP

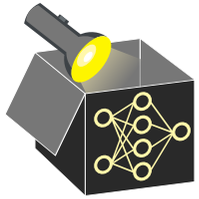
Learning goals

- Understand KernelSHAP as weighted least-squares regression over coalitions
- Grasp how background samples impute "absent" features
- Observational vs. interventional SHAP



KERNEL SHAP - IN 5 STEPS

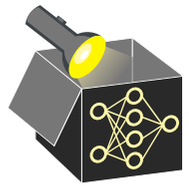
Definition: A kernel-based, model-agnostic method to compute Shapley values via local surrogate models (e.g. linear model)



1. Sample coalition vectors $\mathbf{z}' \in \{0,1\}^p$
2. Map coalition vectors to original feature space and predict
3. Compute kernel weights for surrogate model
4. Fit a weighted linear model
5. Return Shapley values

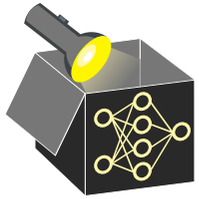
KERNEL SHAP - IN 5 STEPS

Definition: A kernel-based, model-agnostic method to compute Shapley values via local surrogate models (e.g. linear model)



1. Sample coalition vectors $' \in \{0,1\}^p$
2. Map coalition vectors to original feature space and predict
3. Compute kernel weights for surrogate model
4. Fit a weighted linear model
5. Return Shapley values

KERNEL SHAP - IN 5 STEPS



Step 1: Sample coalition vectors

- Sample K coalitions from the simplified (binary) feature space

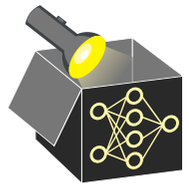
$$\mathbf{z}'^{(k)} \in \{0,1\}^p, \quad k \in \{1,\ldots,K\}$$

- $\mathbf{z}'^{(k)} \in \{0,1\}^p$ indicates which features are present in k-th coalition
- To evaluate the model on each coalition, we must map $\mathbf{z}'^{(k)}$ to original space
- Example ($\mathbf{x} = (51.6, 5.1, 17.0)$) $\Rightarrow 2^p = 2^3 = 8$ coalitions (without sampling)

Map to original feature space

Coalition	$\mathbf{z}'^{(k)}$	hum	temp	ws	$\mathbf{z}^{(k)}$	hum	temp	ws
$\varnothing$	$\mathbf{z}'^{(1)}$	0	0	0	$\mathbf{z}^{(1)}$	?	?	?
hum	$\mathbf{z}'^{(2)}$	1	0	0	$\mathbf{z}^{(2)}$	51.6	?	?
temp	$\mathbf{z}'^{(3)}$	0	1	0	$\mathbf{z}^{(3)}$	?	5.1	?
ws	$\mathbf{z}'^{(4)}$	0	0	1	$\mathbf{z}^{(4)}$	?	?	17.0
hum, temp	$\mathbf{z}'^{(5)}$	1	1	0	$\mathbf{z}^{(5)}$	51.6	5.1	?
temp, ws	$\mathbf{z}'^{(6)}$	0	1	1	$\mathbf{z}^{(6)}$	?	5.1	17.0
hum, ws	$\mathbf{z}'^{(7)}$	1	0	1	$\mathbf{z}^{(7)}$	51.6	?	17.0
hum, temp, ws	$\mathbf{z}'^{(8)}$	1	1	1	$\mathbf{z}^{(8)}$	51.6	5.1	17.0

KERNEL SHAP - IN 5 STEPS



Step 1: Sample coalition vectors

- Sample K coalitions from the simplified (binary) feature space

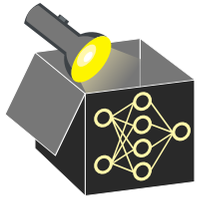
$$\mathbf{z}'^{(k)} \in \{0,1\}^p, \quad k \in \{1,\ldots,K\}$$

- $\mathbf{z}'^{(k)} \in \{0,1\}^p$ indicates which features are present in k-th coalition
- To evaluate the model on each coal., we must map $\mathbf{z}'^{(k)}$ to original space
- Example ($\mathbf{x} = (51.6, 5.1, 17.0)$) $\Rightarrow 2^p = 2^3 = 8$ coals (without sampling)

Map to original feature space

Coalition	$\mathbf{z}'^{(k)}$	hum	temp	ws	$\mathbf{z}^{(k)}$	hum	temp	ws
$\varnothing$	$\mathbf{z}'^{(1)}$	0	0	0	$\mathbf{z}^{(1)}$	?	?	?
hum	$\mathbf{z}'^{(2)}$	1	0	0	$\mathbf{z}^{(2)}$	51.6	?	?
temp	$\mathbf{z}'^{(3)}$	0	1	0	$\mathbf{z}^{(3)}$	?	5.1	?
ws	$\mathbf{z}'^{(4)}$	0	0	1	$\mathbf{z}^{(4)}$	?	?	17.0
hum, temp	$\mathbf{z}'^{(5)}$	1	1	0	$\mathbf{z}^{(5)}$	51.6	5.1	?
temp, ws	$\mathbf{z}'^{(6)}$	0	1	1	$\mathbf{z}^{(6)}$	?	5.1	17.0
hum, ws	$\mathbf{z}'^{(7)}$	1	0	1	$\mathbf{z}^{(7)}$	51.6	?	17.0
hum, temp, ws	$\mathbf{z}'^{(8)}$	1	1	1	$\mathbf{z}^{(8)}$	51.6	5.1	17.0

KERNEL SHAP – IN 5 STEPS



Step 2: Map coalition vectors to original feature space and predict

- Define mapping $h_{\mathbf{x},\mathbf{x}'} : \{0,1\}^p \to \mathbb{R}^p$, where: $\left(h_{\mathbf{x},\mathbf{x}'}(\mathbf{z}')\right)_j = \begin{cases} x_j & \text{if } z'_j = 1 \\ x'_j & \text{if } z'_j = 0 \end{cases}$
- Construct $\mathbf{z} = h_{\mathbf{x},\mathbf{x}'}(\mathbf{z}')$ where present features take their values from $\mathbf{x}$ and absent features are imputed with values from a random background sample $\mathbf{x}' = (64.3, 28.0, 14.5)$
- Evaluate the model on each constructed vector: $\hat{f} = \hat{f}(h_{\mathbf{x},\mathbf{x}'}(\mathbf{z}'^{(k)}))$

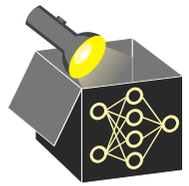
$h_{\mathbf{x},\mathbf{x}'}(\mathbf{z}'^{(k)})$

Coalition	$\mathbf{z}'^{(k)}$	hum	temp	ws	$\mathbf{z}^{(k)}$	hum	temp	ws	$\hat{f}(h_{\mathbf{x}}(\mathbf{z}'^{(k)}))$
∅	$\mathbf{z}'^{(1)}$	0	0	0	$\mathbf{z}^{(1)}$	64.3	28.0	14.5	6211
hum	$\mathbf{z}'^{(2)}$	1	0	0	$\mathbf{z}^{(2)}$	51.6	28.0	14.5	5586
temp	$\mathbf{z}'^{(3)}$	0	1	0	$\mathbf{z}^{(3)}$	64.3	5.1	14.5	3295
ws	$\mathbf{z}'^{(4)}$	0	0	1	$\mathbf{z}^{(4)}$	64.3	28.0	17.0	5762
hum, temp	$\mathbf{z}'^{(5)}$	1	1	0	$\mathbf{z}^{(5)}$	51.6	5.1	14.5	2616
temp, ws	$\mathbf{z}'^{(6)}$	0	1	1	$\mathbf{z}^{(6)}$	64.3	5.1	17.0	2900
hum, ws	$\mathbf{z}'^{(7)}$	1	0	1	$\mathbf{z}^{(7)}$	51.6	28.0	17.0	5411
hum, temp, ws	$\mathbf{z}'^{(8)}$	1	1	1	$\mathbf{z}^{(8)}$	51.6	5.1	17.0	2573

KERNEL SHAP IN 5 STEPS

Step 2: Map coalition vectors to original feature space and predict

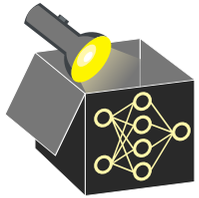
- Define mapping $h_{\mathbf{x},\mathbf{x}'} : \{0,1\}^p \to \mathbb{R}^p$: $\left(h_{\mathbf{x},\mathbf{x}'}(\mathbf{z}')\right)_j = \begin{cases} x_j & \text{if } z'_j = 1 \\ x'_j & \text{if } z'_j = 0 \end{cases}$
- Construct $= h_{\mathbf{x},\mathbf{x}'}(\mathbf{z}')$ where present features take their values from $\mathbf{x}$ and absent features are imputed with values from a random background sample $\mathbf{x}' = (64.3, 28.0, 14.5)$
- Evaluate the model on each constructed vector: $\hat{f} = \hat{f}(h_{\mathbf{x},\mathbf{x}'}(\mathbf{z}'^{(k)}))$



$h_{\mathbf{x},\mathbf{x}'}(\mathbf{z}'^{(k)})$

Coalition	$\mathbf{z}'^{(k)}$	hum	temp	ws	$\mathbf{z}^{(k)}$	hum	temp	ws	$\hat{f}(h_{\mathbf{x}}(\mathbf{z}'^{(k)}))$
∅	$\mathbf{z}'^{(1)}$	0	0	0	$\mathbf{z}^{(1)}$	64.3	28.0	14.5	6211
hum	$\mathbf{z}'^{(2)}$	1	0	0	$\mathbf{z}^{(2)}$	51.6	28.0	14.5	5586
temp	$\mathbf{z}'^{(3)}$	0	1	0	$\mathbf{z}^{(3)}$	64.3	5.1	14.5	3295
ws	$\mathbf{z}'^{(4)}$	0	0	1	$\mathbf{z}^{(4)}$	64.3	28.0	17.0	5762
hum, temp	$\mathbf{z}'^{(5)}$	1	1	0	$\mathbf{z}^{(5)}$	51.6	5.1	14.5	2616
temp, ws	$\mathbf{z}'^{(6)}$	0	1	1	$\mathbf{z}^{(6)}$	64.3	5.1	17.0	2900
hum, ws	$\mathbf{z}'^{(7)}$	1	0	1	$\mathbf{z}^{(7)}$	51.6	28.0	17.0	5411
hum, temp, ws	$\mathbf{z}'^{(8)}$	1	1	1	$\mathbf{z}^{(8)}$	51.6	5.1	17.0	2573

KERNEL SHAP – IN 5 STEPS



Step 2: Map coalition vectors to original feature space and predict

Fix coalition vector $\mathbf{z}' = (1, 0, 0)$; draw multiple background samples $\mathbf{x}'^{(1)}, \ldots, \mathbf{x}'^{(B)}$
$\Rightarrow$ keep **hum**, replace **temp** and **ws** by draws from the background data.

Sample b	hum (from $\mathbf{x}$)	temp (from $\mathbf{x}'^{(b)}$)	ws (from $\mathbf{x}'^{(b)}$)	$\hat{f}(h_{\mathbf{x},\mathbf{x}'^{(b)}}(\mathbf{z}'))$
1	51.6	28.0	14.5	4635
2	51.6	5.1	14.5	3295
3	51.6	28.0	17.0	5586
$\vdots$		$\ldots$		$\ldots$

- Typically, many background samples $\mathbf{x}'^{(1)}, \ldots, \mathbf{x}'^{(B)}$ are used to approximate the marginal expectation required for KernelSHAP via Monte-Carlo average:

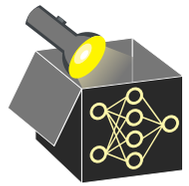
$$\mathbb{E}_{\mathbf{X}_{-S}}\left[f(\mathbf{x}_S, \mathbf{X}_{-S})\right] \approx \frac{1}{B}\sum_{b=1}^{B} \hat{f}(h_{\mathbf{x},\mathbf{x}'^{(b)}}(\mathbf{z}'))$$

- Background samples $\mathbf{x}'^{(b)}$ are drawn from:
  - Conditional distribution $\mathbf{x}'^{(b)} \sim P_{\mathbf{x}|\mathbf{x}_S=\mathbf{x}_S} \rightsquigarrow$ **Observational SHAP**
  - Marginal distribution $\mathbf{x}'^{(b)} \sim P_{\mathbf{x}} \rightsquigarrow$ **Interventional SHAP**
- The same procedure applies to every other coalition vector $\mathbf{z}'^{(k)}$.

KERNEL SHAP IN 5 STEPS

Step 2: Map coalition vectors to original feature space and predict

Fix $\mathbf{z}' = (1, 0, 0)$; draw multiple background samples $\mathbf{x}'^{(1)}, \ldots, \mathbf{x}'^{(B)}$
$\Rightarrow$ keep **hum**, replace **temp** and **ws** by draws from the background data.



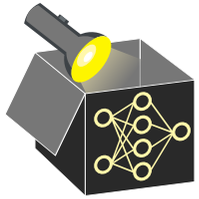
Sample b	hum (from $\mathbf{x}$)	temp (from $\mathbf{x}'^{(b)}$)	ws (from $\mathbf{x}'^{(b)}$)	$\hat{f}(h_{\mathbf{x},\mathbf{x}'^{(b)}}(\mathbf{z}'))$
1	51.6	28.0	14.5	4635
2	51.6	5.1	14.5	3295
3	51.6	28.0	17.0	5586
$\vdots$		$\ldots$		$\ldots$

- Typically, many background samples $\mathbf{x}'^{(1)}, \ldots, \mathbf{x}'^{(B)}$ are used to approximate the marginal expectation required for KernelSHAP via Monte-Carlo average:

$$\mathbb{E}_{\mathbf{X}_{-S}}\left[f(\mathbf{x}_S, \mathbf{X}_{-S})\right] \approx \frac{1}{B}\sum_{b=1}^{B} \hat{f}(h_{\mathbf{x},\mathbf{x}'^{(b)}}(\mathbf{z}'))$$

- Background samples $\mathbf{x}'^{(b)}$ are drawn from:
  - Conditional distribution $\mathbf{x}'^{(b)} \sim P_{\mathbf{x}|\mathbf{x}_S=\mathbf{x}_S} \rightsquigarrow$ **Observational SHAP**
  - Marginal distribution $\mathbf{x}'^{(b)} \sim P_{\mathbf{x}} \rightsquigarrow$ **Interventional SHAP**
- The same procedure applies to every other coalition vector $\mathbf{z}'^{(k)}$.

KERNEL SHAP - IN 5 STEPS



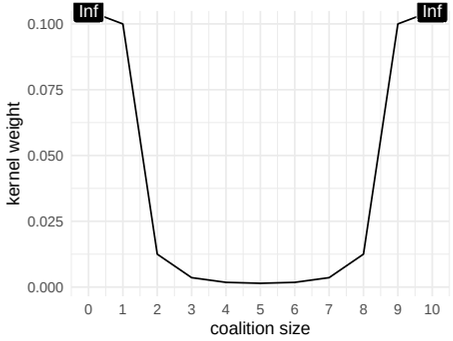
Step 3: Compute kernel weights for surrogate model

Intuition: We learn most about a feature's effect when (recall multinomial coefficient in Shapley value's set definition):

- it appears **in isolation** (small coalition), or
- in **near-complete context** (large coalition).

$\Rightarrow$ SHAP assigns highest weights to very small and very large coalitions.

Note: The figure below is illustrative and not tied to the running example.



KERNEL SHAP - IN 5 STEPS

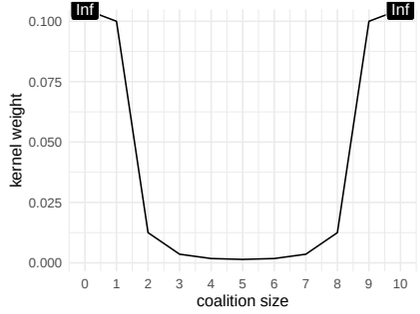
Step 3: Compute kernel weights for surrogate model

Intuition: We learn most about a feature's effect when (recall multinomial coefficient in Shapley value's set definition):

- it appears **in isolation** (small coalition), or
- in **near-complete context** (large coalition).

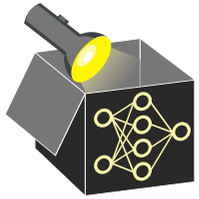
$\Rightarrow$ SHAP assigns highest weights to very small and very large coalitions.

Note: The figure below is illustrative and not tied to the running example.



KERNEL SHAP - IN 5 STEPS

Step 3: Compute kernel weights for surrogate model



$\pi_x(\mathbf{z}'^{(k)})$: kernel weight for coalition $\mathbf{z}'^{(k)}$

p: Number of features in $\mathbf{x}$

$$\pi_x\left(\mathbf{z}'^{(k)}\right) = \frac{(p-1)}{\binom{p}{\left|\mathbf{z}'^{(k)}\right|}\left|\mathbf{z}'^{(k)}\right|\left(p-\left|\mathbf{z}'^{(k)}\right|\right)}$$

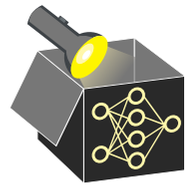
$|\ \mathbf{z}'^{(k)}\ |$: coalition size / sum of 1s in $\mathbf{z}'^{(k)}$

Note: Weights differ from multinomial coefficient in the Shapley value set-definiton but are constructed to yield the same Shapley values via weighted linear regression.

▸ see shapley_kernel_proof.pdf

KERNEL SHAP - IN 5 STEPS

Step 3: Compute kernel weights for surrogate model



$\pi_x(\mathbf{z}'^{(k)})$: kernel weight for coalition $\mathbf{z}'^{(k)}$

p: Number of features in $\mathbf{x}$

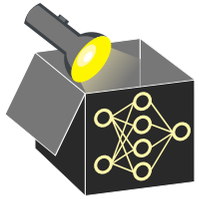
$$\pi_x\left(\mathbf{z}'^{(k)}\right) = \frac{(p-1)}{\binom{p}{\left|\mathbf{z}'^{(k)}\right|}\left|\mathbf{z}'^{(k)}\right|\left(p-\left|\mathbf{z}'^{(k)}\right|\right)}$$

$|\ \mathbf{z}'^{(k)}\ |$: coalition size / sum of 1s in $\mathbf{z}'^{(k)}$

Note: Weights differ from multinomial coefficient in the Shapley value set-definiton but are constructed to yield the same Shapley values via weighted linear regression.

▸ see *shap* 2017

KERNEL SHAP - IN 5 STEPS



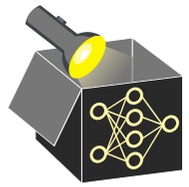
Step 3: Compute kernel weights for surrogate model

Purpose: Assign observation weights $\pi_x(\mathbf{z}')$ to each coalition vector $\mathbf{z}'$ when solving the local surrogate (weighted linear regression), e.g.:

$$\pi_x(\mathbf{z}') = \frac{(p-1)}{\binom{p}{|\mathbf{z}'|}\,|\mathbf{z}'|(p-|\mathbf{z}'|)} \rightsquigarrow \pi_x(\mathbf{z}'=(1,0,0)) = \frac{(3-1)}{\binom{3}{1}\,1\,(3-1)} = \frac{1}{3}$$

Coalition	$\mathbf{z}'^{(k)}$	hum	temp	ws	weight $\pi_x(\mathbf{z}')$
$\varnothing$	$\mathbf{z}'^{(1)}$	0	0	0	∞
hum	$\mathbf{z}'^{(2)}$	1	0	0	0.33
temp	$\mathbf{z}'^{(3)}$	0	1	0	0.33
ws	$\mathbf{z}'^{(4)}$	0	0	1	0.33
hum, temp	$\mathbf{z}'^{(5)}$	1	1	0	0.33
temp, ws	$\mathbf{z}'^{(6)}$	0	1	1	0.33
hum, ws	$\mathbf{z}'^{(7)}$	1	0	1	0.33
hum, temp, ws	$\mathbf{z}'^{(8)}$	1	1	1	∞

KERNEL SHAP - IN 5 STEPS



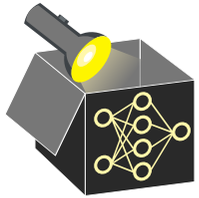
Step 3: Compute kernel weights for surrogate model

Purpose: Assign observation weights $\pi_x(\mathbf{z}')$ to each coalition vector $'$ when solving the local surrogate (weighted linear regression), e.g.:

$$\pi_x(\mathbf{z}') = \frac{(p-1)}{\binom{p}{|\mathbf{z}'|}\,|\mathbf{z}'|(p-|\mathbf{z}'|)} \rightsquigarrow \pi_x(\mathbf{z}'=(1,0,0)) = \frac{(3-1)}{\binom{3}{1}\,1\,(3-1)} = \frac{1}{3}$$

Coalition	$\mathbf{z}'^{(k)}$	hum	temp	ws	weight $\pi_x(\mathbf{z}')$
$\varnothing$	$\mathbf{z}'^{(1)}$	0	0	0	∞
hum	$\mathbf{z}'^{(2)}$	1	0	0	0.33
temp	$\mathbf{z}'^{(3)}$	0	1	0	0.33
ws	$\mathbf{z}'^{(4)}$	0	0	1	0.33
hum, temp	$\mathbf{z}'^{(5)}$	1	1	0	0.33
temp, ws	$\mathbf{z}'^{(6)}$	0	1	1	0.33
hum, ws	$\mathbf{z}'^{(7)}$	1	0	1	0.33
hum, temp, ws	$\mathbf{z}'^{(8)}$	1	1	1	∞

KERNEL SHAP - IN 5 STEPS

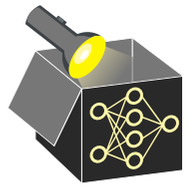


Step 3: Compute kernel weights for surrogate model

- For $p > 3$ features, the finite weights are all 0.33 as every shown coalition has the same size ($|S| = 1$ and $|-S| = 2$ and vice versa for $p = 3$).
- In general (when $p > 3$), weights vary with coalition size.
- Empty and full coalitions receive weight ∞ (division-by-zero term)
  $\rightsquigarrow$ These coalition vectors are not used as observations for the linear regression
  $\rightsquigarrow$ Instead constraints are used to ensure *local accuracy* and *missingness*

Coalition	$\mathbf{z}'^{(k)}$	hum	temp	ws	weight $\pi_x(\mathbf{z}')$
$\varnothing$	$\mathbf{z}'^{(1)}$	0	0	0	∞
hum	$\mathbf{z}'^{(2)}$	1	0	0	0.33
temp	$\mathbf{z}'^{(3)}$	0	1	0	0.33
ws	$\mathbf{z}'^{(4)}$	0	0	1	0.33
hum, temp	$\mathbf{z}'^{(5)}$	1	1	0	0.33
temp, ws	$\mathbf{z}'^{(6)}$	0	1	1	0.33
hum, ws	$\mathbf{z}'^{(7)}$	1	0	1	0.33
hum, temp, ws	$\mathbf{z}'^{(8)}$	1	1	1	∞

KERNEL SHAP - IN 5 STEPS

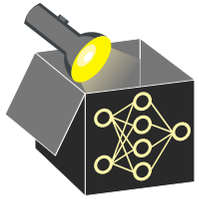


Step 3: Compute kernel weights for surrogate model

- For $p > 3$ features, the finite weights are all 0.33 as every shown coalition has the same size ($|S| = 1$ and $|-S| = 2$ and vice versa for $p = 3$).
- In general (when $p > 3$), weights vary with coalition size.
- Empty and full coalitions receive weight ∞ (division-by-zero term)
  $\rightsquigarrow$ These coalition vectors are not used as obs. for the linear regression
  $\rightsquigarrow$ Instead constraints are used to ensure *local accuracy* and *missingness*

Coalition	$\mathbf{z}'^{(k)}$	hum	temp	ws	weight $\pi_x(\mathbf{z}')$
$\varnothing$	$\mathbf{z}'^{(1)}$	0	0	0	∞
hum	$\mathbf{z}'^{(2)}$	1	0	0	0.33
temp	$\mathbf{z}'^{(3)}$	0	1	0	0.33
ws	$\mathbf{z}'^{(4)}$	0	0	1	0.33
hum, temp	$\mathbf{z}'^{(5)}$	1	1	0	0.33
temp, ws	$\mathbf{z}'^{(6)}$	0	1	1	0.33
hum, ws	$\mathbf{z}'^{(7)}$	1	0	1	0.33
hum, temp, ws	$\mathbf{z}'^{(8)}$	1	1	1	∞

KERNEL SHAP - IN 5 STEPS



Step 4: Fit a weighted linear model

Goal Estimate Shapley values ϕ_j as coefficients of a local, weighted linear surrogate.

$$g(\mathbf{z}') \;=\; \phi_0 \;+\; \sum_{j=1}^{p} \phi_j z_j'$$

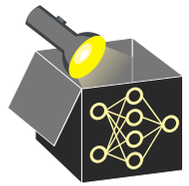
Weighted least-squares objective

$$\min_{\phi} \; \sum_{k=1}^{K} \pi_x(\mathbf{z}'^{(k)}) \Big[\hat{f}\big(h_{\mathbf{x}}(\mathbf{z}'^{(k)})\big) - g(\mathbf{z}'^{(k)})\Big]^2$$

Boundary coalitions ($\mathbf{z}' = \mathbf{1}$ and $\mathbf{z}' = \mathbf{0}$) enforce constraints on coefficients

$$\phi_0 = \mathbb{E}[\hat{f}(\mathbf{X})], \qquad \sum_{j=1}^{p} \phi_j = \hat{f}(\mathbf{x}) - \phi_0.$$

KERNEL SHAP - IN 5 STEPS



Step 4: Fit a weighted linear model

Goal
Estimate Shapley values ϕ_j as coefficents of a local, weighted linear surrogate.

$$g(\mathbf{z}') \;=\; \phi_0 \;+\; \sum_{j=1}^{p} \phi_j z_j'$$

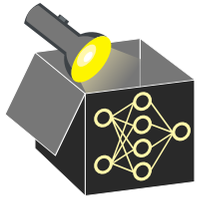
Weighted least-squares objective

$$\min_{\phi} \; \sum_{k=1}^{K} \pi_x(\mathbf{z}'^{(k)}) \Big[\hat{f}\big(h_{\mathbf{x}}(\mathbf{z}'^{(k)})\big) - g(\mathbf{z}'^{(k)})\Big]^2$$

Boundary coalitions ($' = \mathbf{1}$ and $' = \mathbf{0}$) enforce constraints on coefficients

$$\phi_0 = \mathbb{E}[\hat{f}(\mathbf{X})], \qquad \sum_{j=1}^{p} \phi_j = \hat{f}(\mathbf{x}) - \phi_0.$$

KERNEL SHAP - IN 5 STEPS



Step 4: Fit a weighted linear model

Goal Estimate Shapley values ϕ_j as coefficients of a local, weighted linear surrogate.

$$g(\mathbf{z}') \;=\; \phi_0 \;+\; \sum_{j=1}^{p} \phi_j z_j'$$

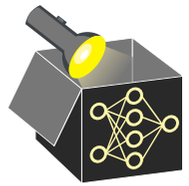
Numeric illustration ($p = 3$)

$$g(\mathbf{z}') = 4515 + 34\, z_1' - 1654\, z_2' - 323\, z_3'$$

$\mathbf{z}'$	hum	temp	ws	weight $\pi_x(\mathbf{z}')$	$\hat{f}(h_{\mathbf{x}}(\mathbf{z}'))$	$g(\mathbf{z}')$
$(1,0,0)$	1	0	0	0.33	4635	4549
$(0,1,0)$	0	1	0	0.33	3087	2861
$(0,0,1)$	0	0	1	0.33	4359	4192
$(1,1,0)$	1	1	0	0.33	3060	2895
$(0,1,1)$	0	1	1	0.33	2623	2538
$(1,0,1)$	1	0	1	0.33	4450	4226
	inputs				outputs	

The inputs and outputs are used to learn the weighted linear regression model.

KERNEL SHAP - IN 5 STEPS



Step 4: Fit a weighted linear model

Goal
Estimate Shapley values ϕ_j as coefficents of a local, weighted linear surrogate.

$$g(\mathbf{z}') \;=\; \phi_0 \;+\; \sum_{j=1}^{p} \phi_j z_j'$$

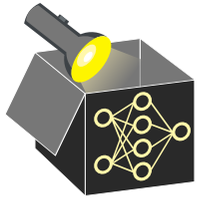
Numeric illustration ($p = 3$)

$$g(\mathbf{z}') = 4515 + 34\, z_1' - 1654\, z_2' - 323\, z_3'$$

$\mathbf{z}'$	hum	temp	ws	weight $\pi_x(\mathbf{z}')$	$\hat{f}(h_{\mathbf{x}}(\mathbf{z}'))$	$g(\mathbf{z}')$
$(1,0,0)$	1	0	0	0.33	4635	4549
$(0,1,0)$	0	1	0	0.33	3087	2861
$(0,0,1)$	0	0	1	0.33	4359	4192
$(1,1,0)$	1	1	0	0.33	3060	2895
$(0,1,1)$	0	1	1	0.33	2623	2538
$(1,0,1)$	1	0	1	0.33	4450	4226
	inputs				outputs	

The inputs and outputs are used to learn the weighted lin. regression model.

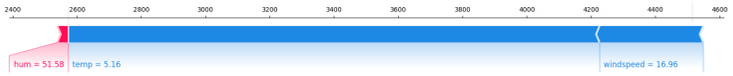
KERNEL SHAP - IN 5 STEPS

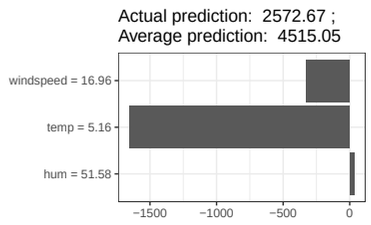


Step 5: Return SHAP values

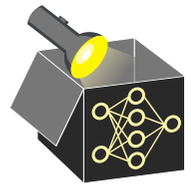
Intuition: Estimated Kernel SHAP values are equivalent to Shapley values

$$g(\mathbf{z}'^{(8)}) = \hat{f}(h_x(\mathbf{z}'^{(8)})) = 4515 + 34 \cdot 1 - 1654 \cdot 1 - 323 \cdot 1$$
$$= \underbrace{\mathbb{E}(\hat{f})}_{\phi_0} + \phi_{hum} + \phi_{temp} + \phi_{ws} = \hat{f}(\mathbf{x}) = 2573$$





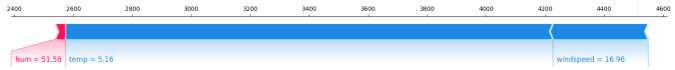
KERNEL SHAP - IN 5 STEPS

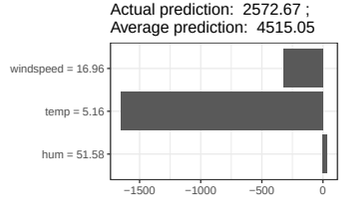


Step 5: Return SHAP values

Intuition: Estimated Kernel SHAP values are equivalent to Shapley values

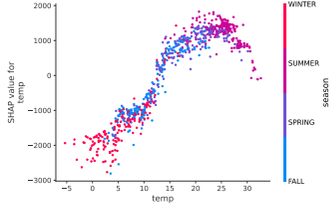
$$g(\mathbf{z}'^{(8)}) = \hat{f}(h_x(\mathbf{z}'^{(8)})) = 4515 + 34 \cdot 1 - 1654 \cdot 1 - 323 \cdot 1$$
$$= \underbrace{\mathbb{E}(\hat{f})}_{\phi_0} + \phi_{hum} + \phi_{temp} + \phi_{ws} = \hat{f}(\mathbf{x}) = 2573$$





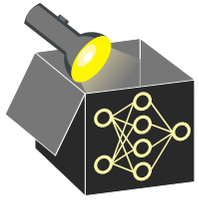
Interpretable Machine Learning

Global SHAP



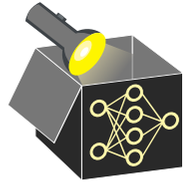
Learning goals

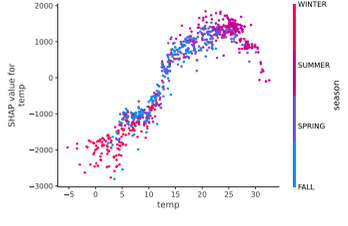
- Understand how SHAP values can be aggregated for global model interpretation
- Learn global SHAP visualizations: feature importance, summary, and dependence plots
- Recognize advantages and limitations of global SHAP explanations



Interpretable Machine Learning

Shapley
Global SHAP





Learning goals

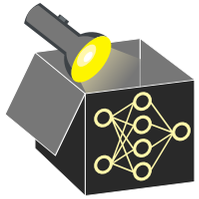
- Understand how SHAP values can be aggregated for global model interpretation
- Learn global SHAP visualizations: feature importance, summary, and dependence plots
- Recognize advantages and limitations of global SHAP explanations

GLOBAL SHAP

Lundberg et al. 2018

Idea:

- Run SHAP for every observation and thereby get a matrix of Shapley values
- The matrix has one row per data observation and one column per feature
- We can interpret the model globally by analyzing the Shapley value matrix



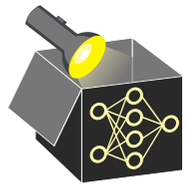
$$\Phi = \begin{bmatrix} \phi_{11} & \phi_{12} & \phi_{13} & \cdots & \phi_{1p} \\ \phi_{21} & \phi_{22} & \phi_{23} & \cdots & \phi_{2p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \phi_{n1} & \phi_{n2} & \phi_{n3} & \cdots & \phi_{np} \end{bmatrix}$$

GLOBAL SHAP

LUNDBERG_2018

Idea:

- Run SHAP for every obs. and thereby get a matrix of Shapley values
- The matrix has one row per data observation and one column per feature
- We can interpret the model globally by analyzing the Shapley value matrix



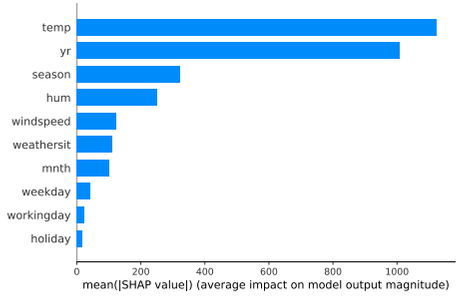
$$\Phi = \begin{bmatrix} \phi_{11} & \phi_{12} & \phi_{13} & \cdots & \phi_{1p} \\ \phi_{21} & \phi_{22} & \phi_{23} & \cdots & \phi_{2p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \phi_{n1} & \phi_{n2} & \phi_{n3} & \cdots & \phi_{np} \end{bmatrix}$$

Interpretable Machine Learning – 1 / 5

FEATURE IMPORTANCE

Idea: Average the absolute Shapley values of each feature over all observations.
This corresponds to calculating averages column by column in matrix Φ

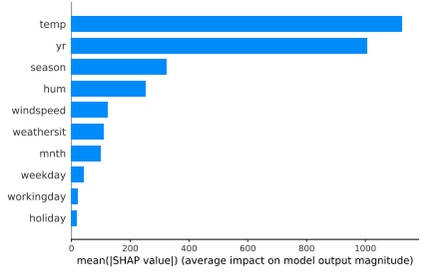
$$I_j = \frac{1}{n} \sum_{i=1}^{n} \left| \phi_j^{(i)} \right|$$



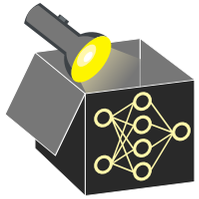
FEATURE IMPORTANCE

Idea: Average the absolute Shapley values of each feature over all obs.
This corresponds to calculating averages column by column in matrix Φ

$$I_j = \frac{1}{n} \sum_{i=1}^{n} \left| \phi_j^{(i)} \right|$$

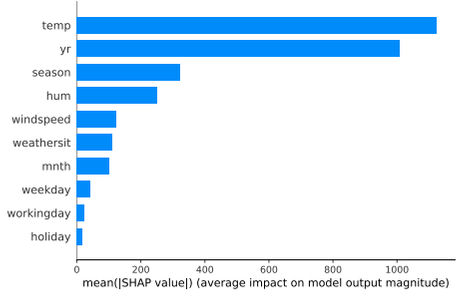


FEATURE IMPORTANCE

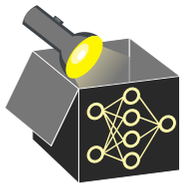


Interpretation:

- Features “temp” and “year” have highest influence on the model’s prediction
- Shapley FI does not provide information on direction of the effect
  $\rightsquigarrow$ Provides a feature ranking based on the magnitude of the Shapley values
- Shapley FI is based only on model predictions
  Note: Other FI measures are based on model’s performance (loss)

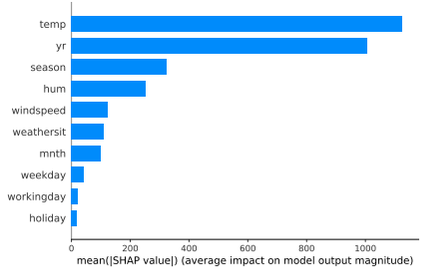


FEATURE IMPORTANCE



Interpretation:

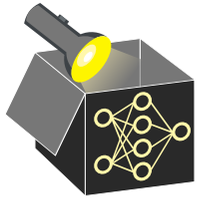
- Feats “temp” and “year” have highest influence on the model’s prediction
- Shapley FI does not provide information on direction of the effect
  $\rightsquigarrow$ Provides feature ranking based on magnitude of Shapley values
- Shapley FI is based only on model predictions
  Note: Other FI measures are based on model’s performance (loss)

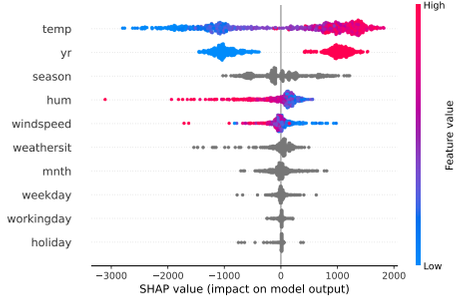


SUMMARY PLOT

Combines feature importance with feature effects

- Each point is a Shapley value for a feature and an observation
- The color represents the value of the feature from low to high
- Overlapping points are jittered in y-axis direction

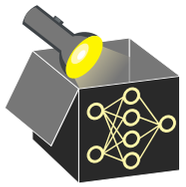


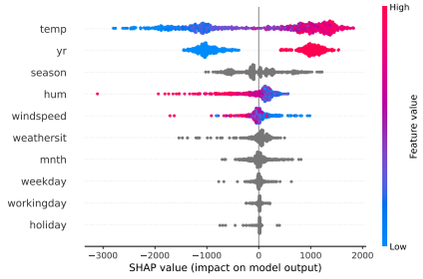


SUMMARY PLOT

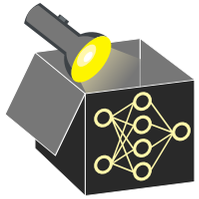
Combines feature importance with feature effects

- Each point is a Shapley value for a feature and an observation
- The color represents the value of the feature from low to high
- Overlapping points are jittered in y-axis direction



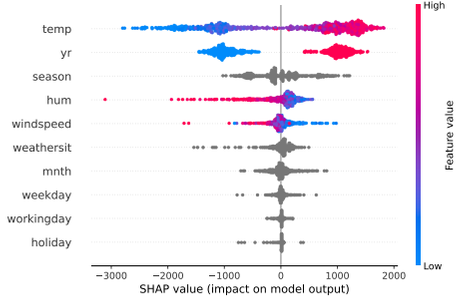


SUMMARY PLOT

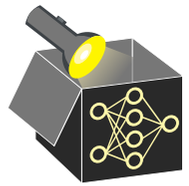


Interpretation:

- Low temp have a negative impact; high temp lead to more bike rentals
- Year: two point clouds for 2011 (low value) and 2012 (high value)
- Categorical features are gray (no low/high value)
- High humidity has a huge negative impact on bike rentals
- Low humidity has a rather minor positive impact on bike rentals

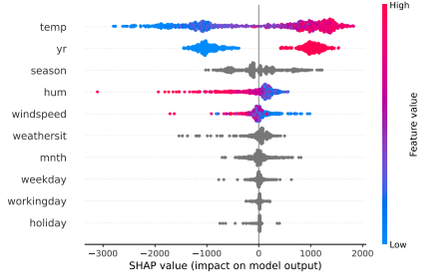


SUMMARY PLOT

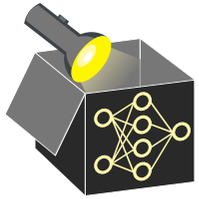


Interpretation:

- Low temp have a negative impact; high temp lead to more bike rentals
- Year: two point clouds for 2011 (low value) and 2012 (high value)
- Categorical features are gray (no low/high value)
- High humidity has a huge negative impact on bike rentals
- Low humidity has a rather minor positive impact on bike rentals

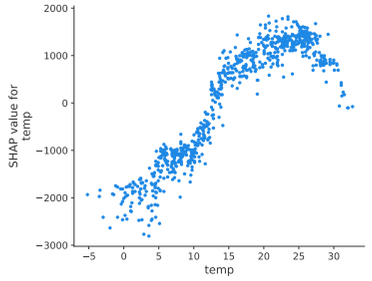


DEPENDENCE PLOT: EFFECT + INTERACTION

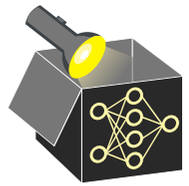


Interpretation of SHAP Dependence Plot (Feature = Temperature)

- Plot points with feature value on x-axis and corresponding SHAP value on y-axis

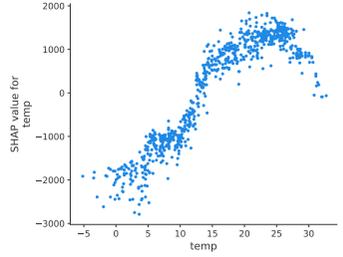


DEPENDENCE PLOT: EFFECT + INTERACTION

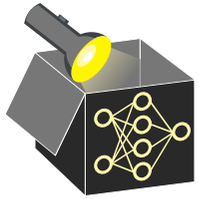


Interpretation of SHAP Dependence Plot (Feature = Temperature)

- Plot points with feature value on x-axis and corresponding SHAP value on y-axis

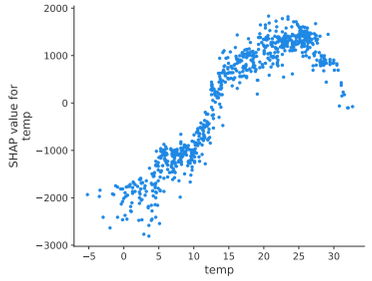


DEPENDENCE PLOT: EFFECT + INTERACTION

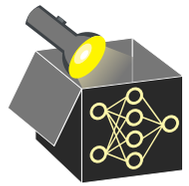


Interpretation of SHAP Dependence Plot (Feature = Temperature)

- Plot points with feature value on x-axis and corresponding SHAP value on y-axis
- Shows how temp influences bike rentals $\rightsquigarrow$ Marginal effect similar to PD plot
- SHAP values increase with temp until $\approx$25 °C: higher temp $\rightsquigarrow$ higher predictions
- After $\approx$25 °C, SHAP values decrease slightly

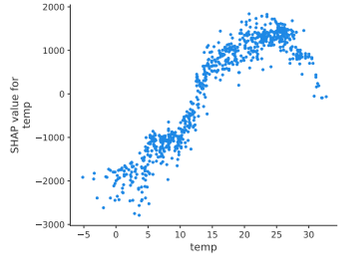


DEPENDENCE PLOT: EFFECT + INTERACTION

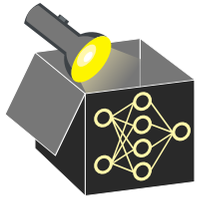


Interpretation of SHAP Dependence Plot (Feature = Temperature)

- Plot points with feature value on x-axis and corresponding SHAP value on y-axis
- Shows temp's influence on rentals $\rightsquigarrow$ Marginal effect similar to PD plot
- SHAP values increase with temp until $\approx$25 °C: higher temp $\rightsquigarrow$ higher predictions
- After $\approx$25 °C, SHAP values decrease slightly

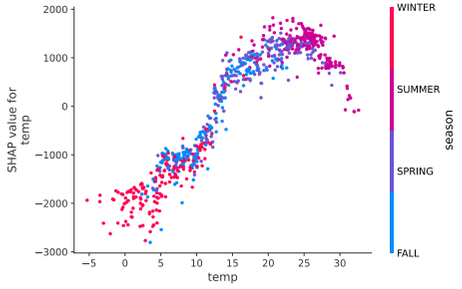


DEPENDENCE PLOT: EFFECT + INTERACTION

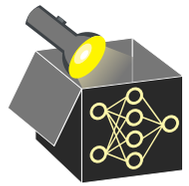


Interpretation of SHAP Dependence Plot (Feature = Temperature)

- Plot points with feature value on x-axis and corresponding SHAP value on y-axis
- Shows how temp influences bike rentals $\rightsquigarrow$ Marginal effect similar to PD plot
- SHAP values increase with temp until $\approx$25 °C: higher temp $\rightsquigarrow$ higher predictions
- After $\approx$25 °C, SHAP values decrease slightly
- Interaction with **season** is visible (via color-encoded observations):
  - In summer, higher temperatures decrease bike rentals
  - In winter, higher temperatures increase bike rentals

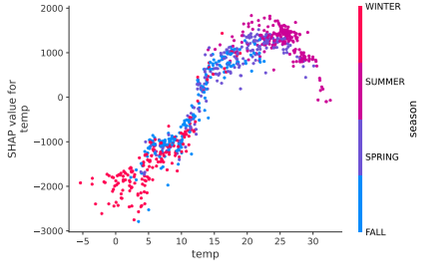


DEPENDENCE PLOT: EFFECT + INTERACTION

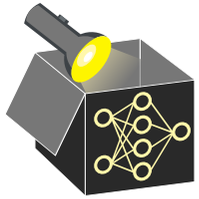


Interpretation of SHAP Dependence Plot (Feature = Temperature)

- Plot points with feature value on x-axis and corresponding SHAP value on y-axis
- Shows temp's influence on rentals $\rightsquigarrow$ Marginal effect similar to PD plot
- SHAP values increase with temp until $\approx$25 °C: higher temp $\rightsquigarrow$ higher predictions
- After $\approx$25 °C, SHAP values decrease slightly
- Interaction with **season** is visible (via color-encoded observations):
  - In summer, higher temperatures decrease bike rentals
  - In winter, higher temperatures increase bike rentals



DISCUSSION



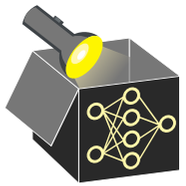
Advantages

- Retains local accuracy: SHAP values exactly decompose predictions
- Aggregating local SHAP values yields global model insights
  $\rightsquigarrow$ Visual diagnostics: feature importance, summary plot, dependence plots
- Efficient for tree-based models via TreeSHAP
  (See ▸ Lundberg et al 2018 and for intuitive explanation ▸ Sukumar: TreeSHAP)
- Unifies feature attribution under a consistent additive framework
- Can be used for images ▸ SHAP image examples and text ▸ SHAP text examples

Disadvantages

- KernelSHAP is inefficient for large datasets or complex models
- Ignores feature dependencies in marginal sampling (interventional SHAP)
- Conditional sampling (observational SHAP) is difficult in practice (would require estimating a conditional distribution)

DISCUSSION



Advantages

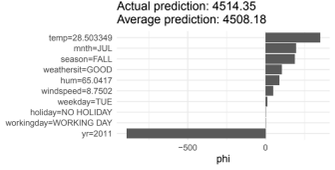
- Retains local accuracy: SHAP values exactly decompose predictions
- Aggregating local SHAP values yields global model insights
  $\rightsquigarrow$ Visual diagnostics: feat. importance; summary and dependence plots
- Efficient for tree-based models via TreeSHAP
  (See ▸ Lundberg 2018 and for intuitive explanation ▸ Sukumar n.d.)
- Unifies feature attribution under a consistent additive framework
- Can be used for images ▸ SHAP n.d. and text ▸ SHAP n.d.

Disadvantages

- KernelSHAP is inefficient for large datasets or complex models
- Ignores feature dependencies in marginal sampling (interventional SHAP)
- Conditional sampling (observational SHAP) is difficult in practice (would require estimating a conditional distribution)

Interpretable Machine Learning

Shapley: Further Resources and Software

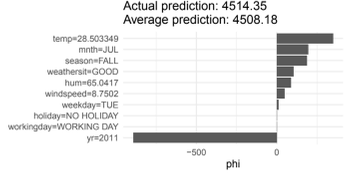


Learning goals

- Learn about further resources
- Get an overview of software packages in R and Python

Interpretable Machine Learning

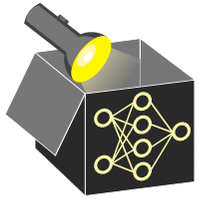
Shapley
Further Resources and Software



Learning goals

- Learn about further resources
- Get an overview of software packages in R and Python

RESOURCES AND SOFTWARE:



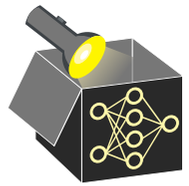
Shap

- The paper: ▸ Shapley 1953
- Chapter in Interpretable Machine Learning book: ▸ Molnar 2025
- R packages: ▸ iml; ▸ fastshap

Shapley

- The paper: ▸ Lundberg et al. 2018
- Chapter in Interpretable Machine Learning book: ▸ Molnar 2025
- R packages: ▸ shapper; ▸ fastshap
- Python packages: ▸ shap

RESOURCES AND SOFTWARE:



Shap

- The paper: ▸ Shapley 1953
- Chapter in Interpretable Machine Learning book: ▸ Molnar 2025
- R packages: ▸ iml n.d.; ▸ fastshap n.d.

Shapley

- The paper: ▸ Lundberg 2018
- Chapter in Interpretable Machine Learning book: ▸ Molnar 2025
- R packages: ▸ shapper n.d.; ▸ fastshap n.d.
- Python packages: ▸ shap n.d.