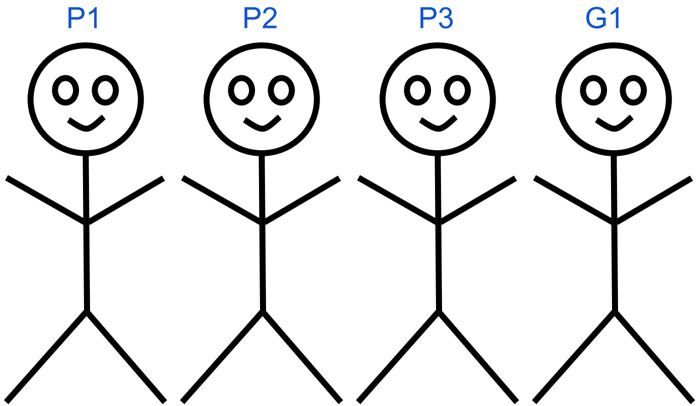
Explaining Model Predictions using Shapley Values

Ankur Ankan
(Github: @ankurankan)

Problem Statement

In a cooperative game, how do we divide the generated reward among the players?

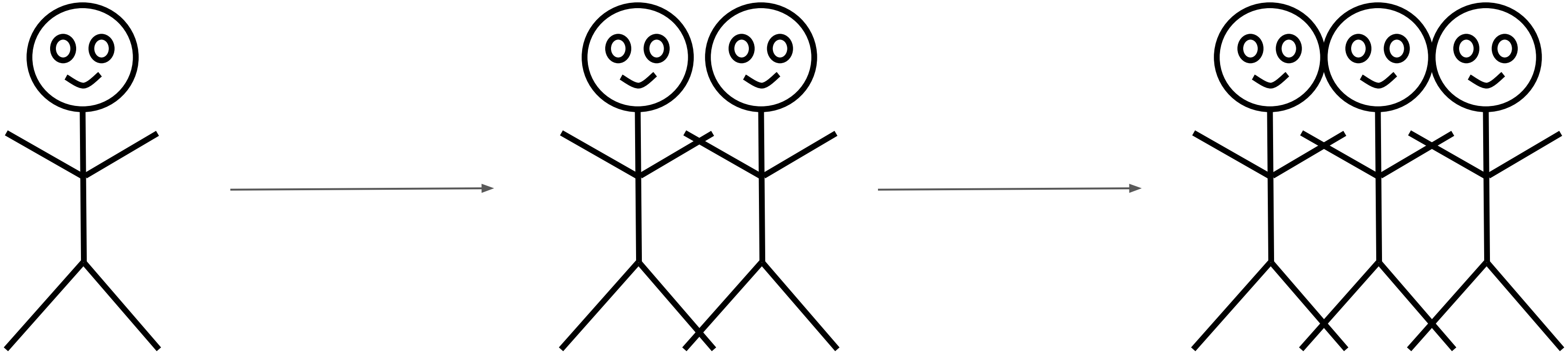
Total value: 100



P = Player, G = Goalkeeper

Solution: Shapley Value

Assume players to be joining in sequence. And the increase in the reward after each player joins is the value that player brings to the team.



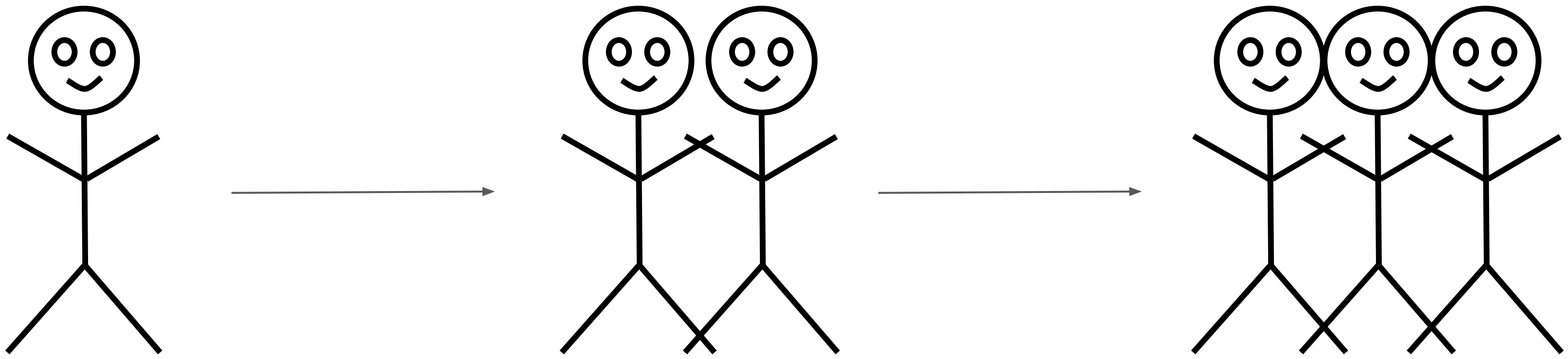
P1
Total value: 20

P1 + P2
Total value: 20 + 15 = 35

P1 + P2 + G1
Total value: 20 + 15 + 30 = 65

Solution: Shapley Value

But value added by a player is dependent on the order in which they are added



P1
Total value: 20

P1 + G1
Total value: 20 + 30 = 50

P1 + G1 + G2
Total value: 20 + 30 + 5 = 55

Solution: Shapley Value

Shapley value suggests to average the value added by a player over all possible combination of orders.

Possible orders

(P1, G1, G2), (P1, G2, G1), (G1, P1, G2), (G1, G2, P1), (G2, P1, G1), (G2, G1, P1)

$$\phi_i(v) = \sum_{S \subseteq N / \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!}(v(S \cup \{i\}) - v(S))$$

Solution: Shapley Value

Shapley value of player i
given a value function v

Total value generated
without including i-th player

$$\boxed{\phi_i(v)} = \sum_{S \subseteq N / \{i\}} \boxed{\frac{|S|!(|N| - |S| - 1)!}{|N|!}} \left(\boxed{v(S \cup \{i\})} - \boxed{v(S)}\right)$$

Total number of possible
orderings for i without
change in value generated
for a given subset of players

Total value generated
with i-th player included

Shapley Value and Machine Learning

Player ⟵ Feature

Team Value ⟵ Model’s output

Shapley Value ⟵ How much does each feature contribute to the model’s output.

SHAP (SHapley Additive exPlanations)

SHAP[1] unifies the concept of Shapley Values along with a few other model explanation methods.

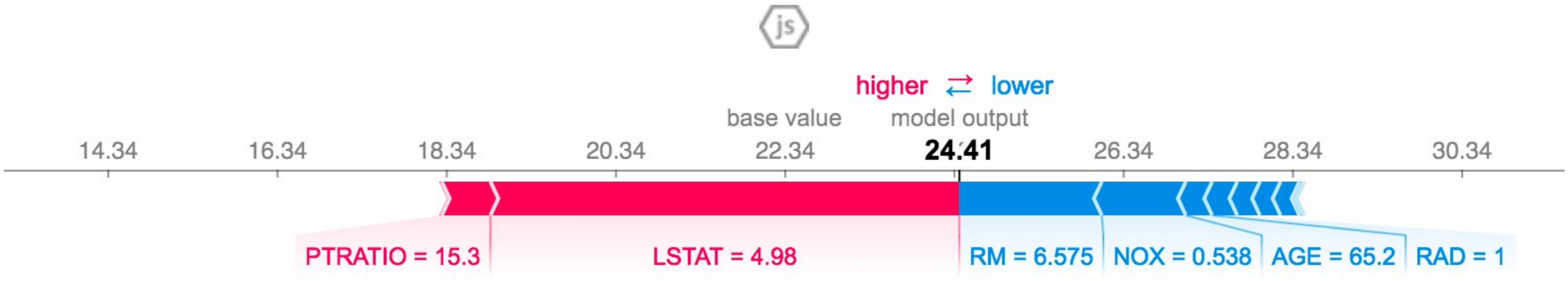
Python has an amazing package for computing SHAP called shap[2].

[1] Lundberg, Scott M., and Su-In Lee. "A unified approach to interpreting model predictions." *Advances in Neural Information Processing Systems*. 2017.
[2] https://github.com/slundberg/shap

Example: xgboost

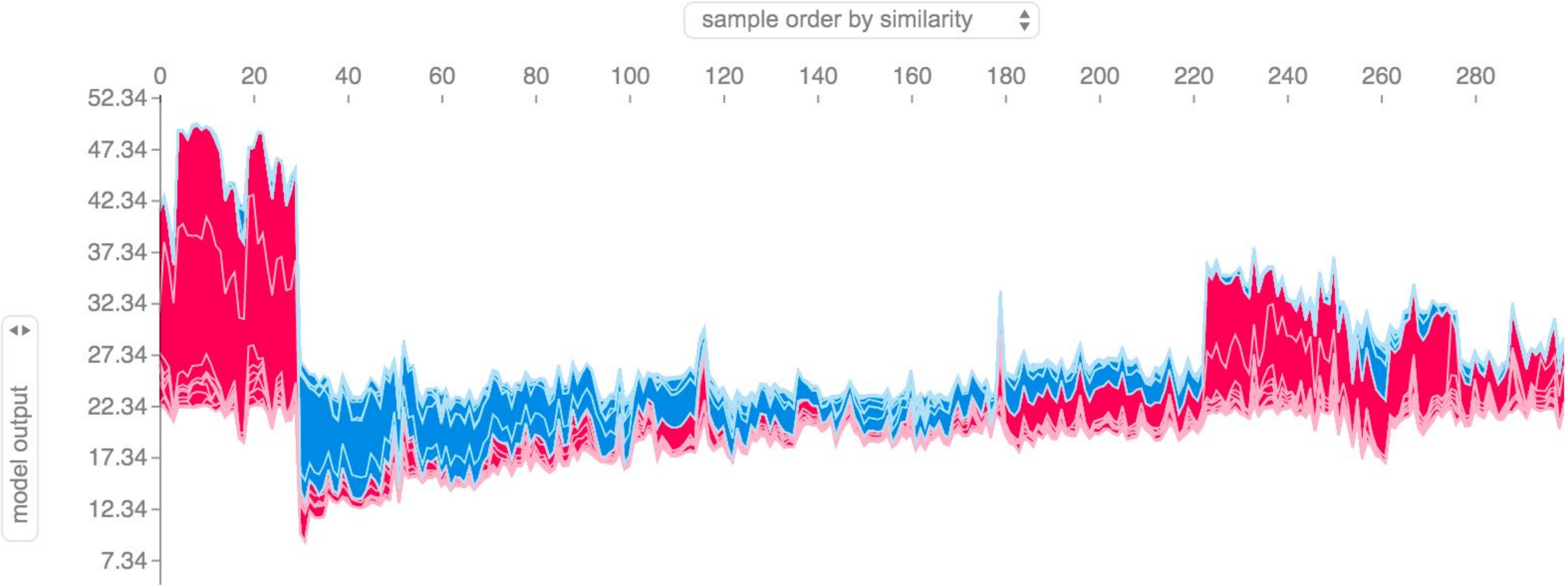
```python
import xgboost, shap

# load JS visualization code to notebook
shap.initjs()

X, y = shap.datasets.boston()
model = xgboost.train({"learning_rate": 0.01}, xgboost.DMatrix(X, label=y), 100)

# explain the model's predictions using SHAP values (TreeExplainer works for LightGBM, CatBoost and sklearn models)
explainer = shap.TreeExplainer(model)
shap_values = explainer.shap_values(X)

# visualize the first prediction's explanation
shap.force_plot(explainer.expected_value, shap_values[0,:], X.iloc[0,:])
```

Example: xgboost



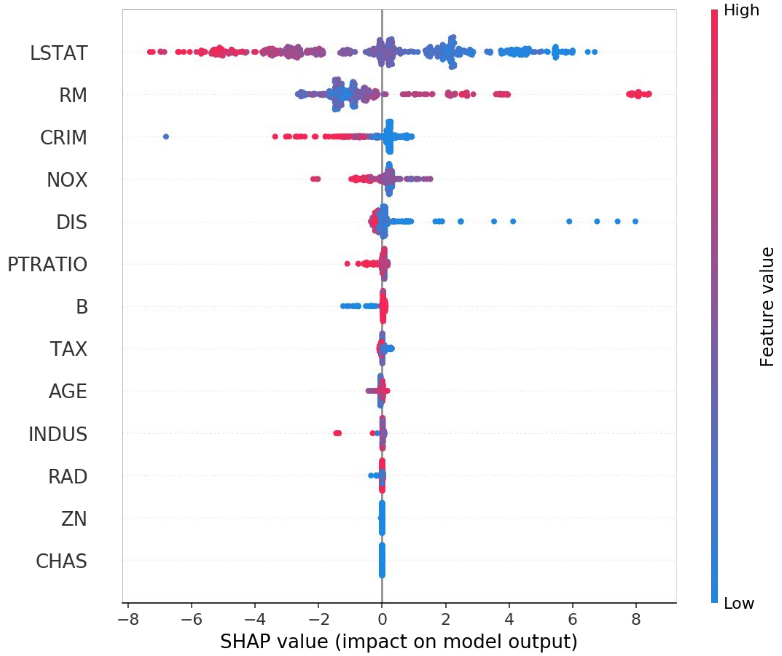
Example: xgboost

```
# visualize the training set predictions
shap.force_plot(explainer.expected_value, shap_values, X)
```



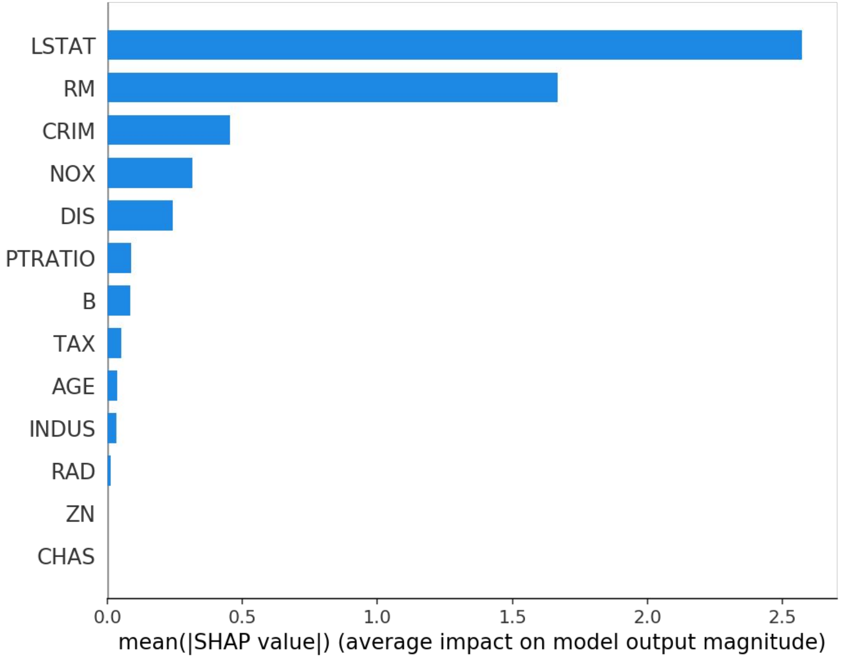
Example: xgboost

```
# create a SHAP dependence plot to show the effect of a single feature across the whole dataset
shap.summary_plot(shap_values, X)
```



Example: xgboost

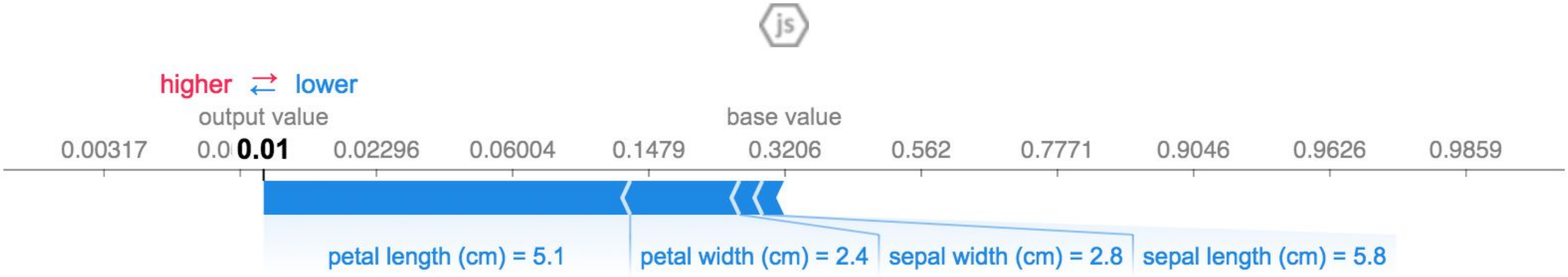
```
# summarize the effects of all the features
shap.summary_plot(shap_values, X, plot_type="bar")
```



Example: Generic Case using Kernel Explainer

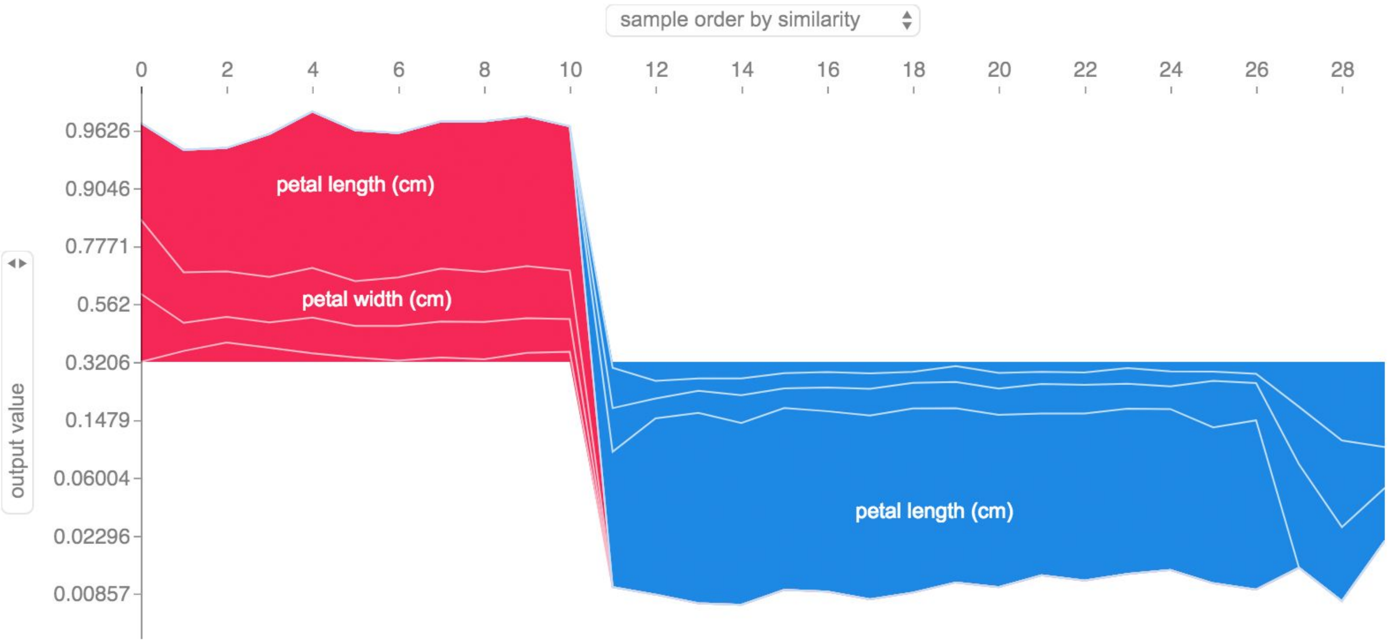
```
import sklearn, shap
from sklearn.model_selection import train_test_split

# print the JS visualization code to the notebook
shap.initjs()

# train a SVM classifier
X_train, X_test, Y_train, Y_test = train_test_split(*shap.datasets.iris(), test_size=0.2, random_state=0)
svm = sklearn.svm.SVC(kernel='rbf', probability=True)
svm.fit(X_train, Y_train)

# use Kernel SHAP to explain test set predictions
explainer = shap.KernelExplainer(svm.predict_proba, X_train, link="logit")
shap_values = explainer.shap_values(X_test, nsamples=100)

# plot the SHAP values for the Setosa output of the first instance
shap.force_plot(explainer.expected_value[0], shap_values[0][0,:], X_test.iloc[0,:], link="logit")
```

Example: Generic Case using Kernel Explainer



Example: Generic Case using Kernel Explainer

```
# plot the SHAP values for the Setosa output of all instances
shap.force_plot(explainer.expected_value[0], shap_values[0], X_test, link="logit")
```



Thank you.

Questions?

Extra Slide: Why to explain Model predictions

- To understand our models.
- To get a better idea of our features.
- To simplify models by removing non contributing/least contributing variables.

Extra Slide: Methods that SHAP unifies

SHAP combines the following 7 model explanation methods:

1. LIME
2. Shapley sampling values
3. DeepLIFT
4. QII
5. Layer-wise relevance propagation
6. Shapley regression values
7. Tree interpreter

SHAP compared to other methods

The only method that satisfies the three axioms of credit attribution:

1. **Dummy Player:** If a player never adds any marginal value, their payoff portion should be 0.
2. **Substitutability:** If two players always add the same marginal value to any subset to which they're added, their payoff portion should be the same.
3. **Additivity:** If a game is composed of two subgames, you should be able to add the payoffs calculated on the subgames, and that should match the payoffs calculated for the full game.

Extra Slide: Runtime of Shapley Value

- Computing all possible combinations of order is $O(2^n)$.
- Approximate Shapley value by only computing for sampled orders.
- But efficient exact computation possible for specific models.

Extra Slide: How to run an algorithm omitting some features

- Not possible to predict by passing None for some specific feature.
- Instead of passing None pass the expected value of the feature.