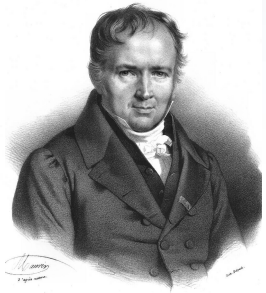
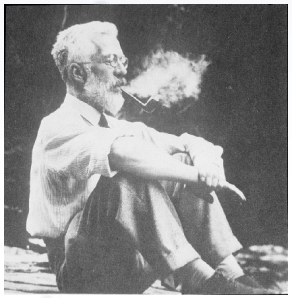
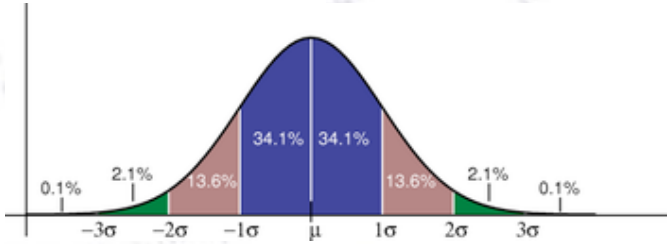
Big Data Analysis

Input Feature Ranking - SHAP values




Troels C. Petersen (NBI)



"Statistics is merely a quantisation of common sense - Big Data is a sharpening of it!"

Input Feature Ranking

It is of course useful to know, which of your input features/variables are useful, and which are not. Thus a **ranking of the features** is desired.

And this is actually a generally nice feature of ML and feature ranking:
It works as an automation of the detective work behind finding relations.

In principle, one could obtain a variables ranking by testing **all combinations** of variables. But that is not feasible on most situation (N features > 7)…

Most algorithms have a build-in input feature ranking, which is based on the very simple idea of **"permutation importance"**.

Permutation Importance

One of the most used methods is “permutation importance” (below quoting Christoph M.: "Interpretable ML" chapter 5.5). The idea is really simple:

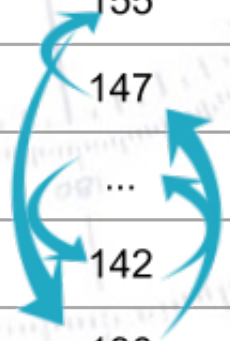
We measure the importance of a feature **by calculating the increase in the model’s loss function after permuting the feature**.
A feature is “important” if shuffling its values increases the model error, because in this case the model relied on the feature for the prediction.
A feature is “unimportant” if shuffling its values leaves the model error unchanged, because the model thus ignored the feature for the prediction.

Permutation Importance

One of the most used methods is “permutation importance” (below quoting Christoph M.: "Interpretable ML" chapter 5.5). The idea is really simple:

We measure the importance of a feature **by calculating the increase in the model’s loss function after permuting the feature**.
A feature is “important” if shuffling its values increases the model error, because in this case the model relied on the feature for the prediction.
A feature is “unimportant” if shuffling its values leaves the model error unchanged, because the model thus ignored the feature for the prediction.

Height at age 20 (cm)	Height at age 10 (cm)	...	Socks owned at age 10
182	155	...	20
175	147	...	10
...	...	...	...
156	142	...	8
153	130	...	24



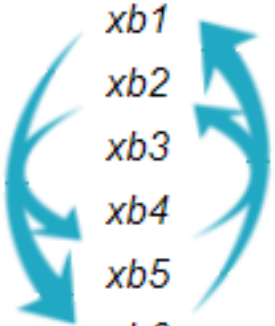
Permutation Importance

Input: Trained model f, feature matrix X, target vector y, loss function L(y,f).

[Fisher, Rudin, and Dominici (2018)]

- Estimate the original model error $e_{orig} = L(y, f(X))$
- For each feature $j = 1,\ldots,p$ do:
  - Generate feature matrix X_{perm} by permuting feature j in the data X. This breaks the association between feature j and true outcome y.
  - Estimate error $e_{perm} = L(Y,f(X_{perm}))$ based on the predictions of X_{perm}.
  - Calculate permutation feature importance $FI_j = e_{perm} / e_{orig}$ (or $e_{perm} - e_{orig}$).
- Sort features by descending FI.

X_A	X_B	X_C	Y
xa1	*xb1*	*xc1*	*y1*
xa2	*xb2*	*xc2*	*y2*
xa3	*xb3*	*xc3*	*y3*
xa4	*xb4*	*xc4*	*y4*
xa5	*xb5*	*xc5*	*y5*
xa6	*xb6*	*xc6*	*y6*



Feature importance with Neural Networks (Towards Data Science)

Shapley Values

A better approximation was developed by Scott Lundberg with **SHAP values**:
SHAP (SHapley Additive exPlanations):
https://github.com/slundberg/shap

This algorithm provides - for each entry - a ranking of the input variables, i.e. a sort of explanation for the result.

One can also sum of the SHAP values over all entries, and then get the overall ranking of feature variables. **They are based on Shapley values**.

Shapley values

Shapley values is a concept from **corporative game theory**, where they are used to provide a possible answer to the question:

"How important is each player to the overall cooperation,
and what payoff can each player reasonably expect?"

The Shapley values are considered "fair", as they are the only distribution with the following properties:

- **Efficiency**: Sum of Shapley values of all agents equals value of grand coalition.
- **Linearity**: If two coalition games described by v and w are combined, then the distributed gains should correspond to the gains derived from the sum of v and w.
- **Null player**: The Shapley value of a null player is zero.
- **Stand alone test**: If v is sub/super additive, then $\varphi_i(v) \leq / \geq v(\{i\})$
- **Anonymity**: Labelling of agents doesn't play a role in assignment of their gains.
- **Maginalism**: Function uses only marginal contributions of player i as arguments.

From such values, one can determine which variables contribute to a final result. And summing the values, one can get an overall idea of which variables are important.

SHAP value calculation

Consider a set **N** (of n players) and a (characteristic or worth) function **v** that maps any subset of players to real numbers:

$$v \ : \ 2^N \rightarrow \mathbb{R}, \quad v(\emptyset) = 0$$

If S is a coalition of players, then v(S) yields the total expected sum of payoffs the members of S can obtain by cooperation.

The Shapley values are calculated as:

$$\varphi_i(v) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(n - |S| - 1)!}{n!} \left[v(S \cup \{i\}) - v(S) \right]$$

To formula can be understood, of we imagine a coalition being formed one actor at a time, with each actor demanding their contribution **v(S ∪ {i}) − v(S)** as a fair compensation, and then for each actor take the average of this contribution over the possible different permutations in which the coalition can be formed.

SHAP value calculation

Consider a set **N** (of n players) and a (characteristic or worth) function **v** that maps any subset of players to real numbers:

$$v \ : \ 2^N \to \mathbb{R}, \quad v(\emptyset) = 0$$

If S is a coalition of players, then v(S) yields the total expected sum of payoffs the members of S can obtain by cooperation.

The Shapley values can also be calculated as:

$$\varphi_i(v) = \frac{1}{n!} \sum_R \left[v(P_i^R \cup \{i\}) - v(P_i^R) \right]$$

where the sum ranges over all n! orders R of the players and P_i^R is the set of players in N which precede i in the order R. This has the interpretation:

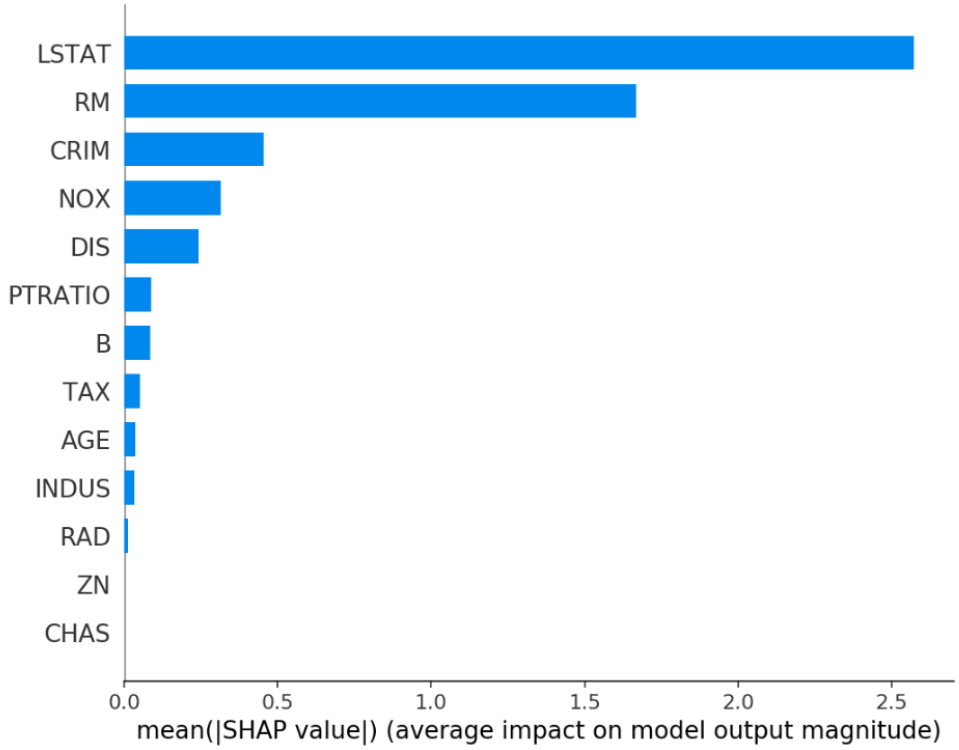
$$\varphi_i(v) = \frac{1}{N_{\text{players}}} \sum_{C \setminus i} \frac{\text{marginal contribution of } i \text{ to coalition C}}{\text{number of coalitions excluding } i \text{ of this size}}$$

Input Feature Ranking

Here is an example from SHAP's github site.

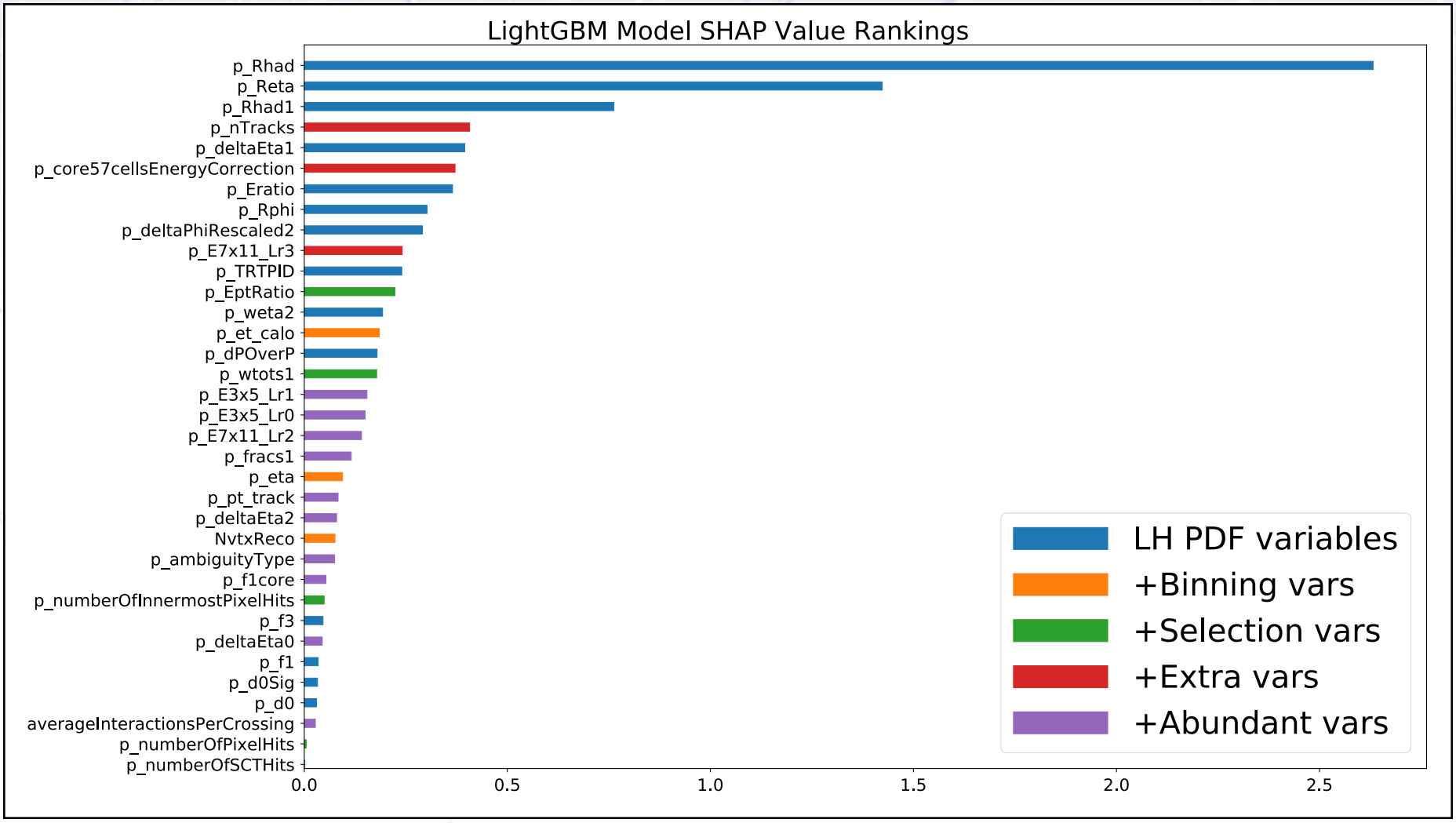
Clearly, LSTAT and RM are the best variables (whatever they are!).

```
shap.summary_plot(shap_values, X, plot_type="bar")
```



Input Feature Ranking

Here is an example from particle physics. The blue variables were "known", but with SHAP we discovered three new quite good variables in data.



Input Feature Ranking

We could of course just add all variables, but want to stay simple, and training the models, we see that the three extra variables gives most of gain.