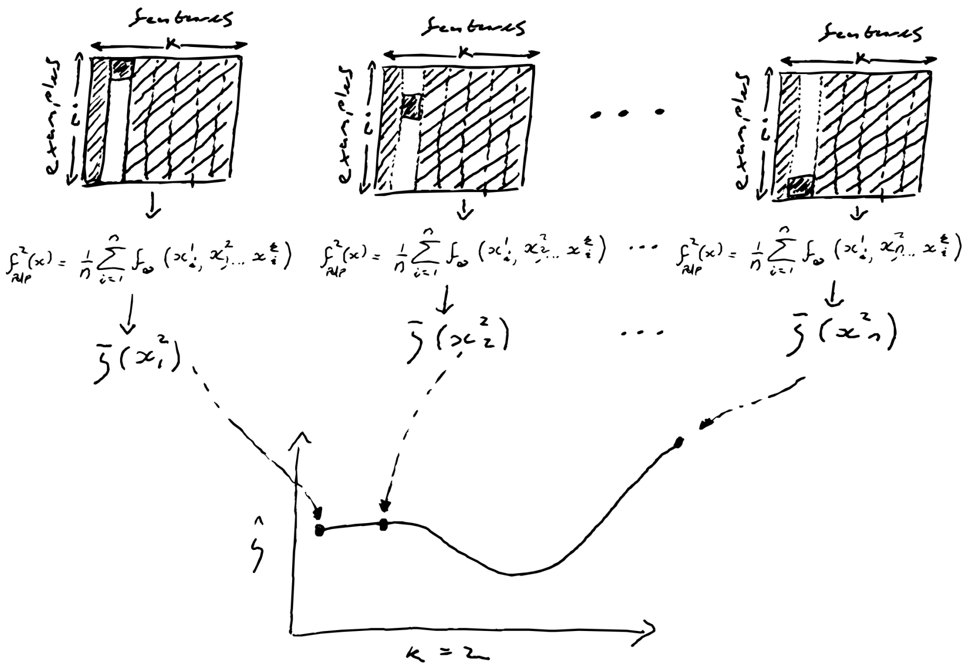
Partial Dependence Plots (PDPs)

PDPs allow us to judge the causal relationship between a feature and the models output. Consider the following situation: We have n data instances $x_i \in X$, these could be rows of examples in tabular data. Each row has k features (columns) such that $x_i \in \mathbb{R}^k$. We have a machine learning model such as a decision tree parameterised by θ that maps the examples to an output prediction, i.e., $f_\theta | x_i \rightarrow \hat{y}$, this could be a classification or regression task, and the data could be numeric or categorical. For clarity a single example vector x_i is given by the tuple$(x_i^1, x_i^2, \ldots, x_i^k)$. If we want to know the effect that a single feature (say feature x^p) has on the prediction as we change its value, then we have to integrate out the other variables such that our remaining function only depends on the value of feature x^p. PDP integrates or averages over all features and examples as seen in the figure. The intuition is that since the only feature that changes is x^p, all the changes to $\hat{y}$ should be attributed to this feature's interaction with the model. This averaging over features and examples allows us to measure the marginal effect of feature x^p on the model's output.


features
k
examples
i
$f^2_{PDP}(x) = \frac{1}{n}\sum_{i=1}^{n} f_\theta(x_i^1, x_1^2, \ldots x_i^k)$
$f^2_{PDP}(x) = \frac{1}{n}\sum_{i=1}^{n} f_\theta(x_i^1, x_2^2, \ldots x_i^k)$
$f^2_{PDP}(x) = \frac{1}{n}\sum_{i=1}^{n} f_\theta(x_i^1, x_n^2, \ldots x_i^k)$
$\bar{y}(x_1^2)$
$\bar{y}(x_2^2)$
$\bar{y}(x_n^2)$
$\hat{y}$
$k = 2$

Procedure: If we want to investigate the model's dependence for example on feature x^p, PDP requires that we calculate

$$f^p_{PDP}(x^p) = \frac{1}{n}\sum_{i=1}^{n} f\left(x_i^1, x_i^2, \ldots, x^p, \ldots, x_i^k\right)$$

for each x^p as seen in the figure, where

- f is the prediction function (e.g., a trained model).
- n is the number of data points.
- x_i^j are the values of the features for the i-th instance in the dataset.
- x^p is the feature for which the partial dependence is being computed.

Each point on the curve in the above figure corresponds to the average model output over the shaded regions. The only value that changes with each iteration is the value of x^p (in this case $p = 2$), giving us n marginal prediction outputs $\overline{y}(x_i^2)$, one for each example. We then order the output monotonically according to the value of x_i^2 and plot the predictions, allowing us to visually deduce any connections between the second feature and the models output. If the dataset is to large then one can approximate the calculation as the following expectation value

$$f^p_{PDP}\left(x^p\right) = \mathbb{E}_{X_{-p}}\left[f\left(x^1, x^2, \ldots, x^p, \ldots, x^k\right)\right],$$

emphasizing that $f^p_{PDP}(x^p)$ is the expected value of the prediction function f given x^p, averaging over the joint distribution of all other features X_{-p}.

PDPs are a global method since they provide a statement about the global relationship of a feature with the models output by considering all data instances/examples. Although this method is easy to implement and intuitive, PDP makes a strong assumption about feature independence, however, features are often highly correlated in practice, meaning that PDPs can generate unrealistic datapoints that do not respect the actual distribution of the data. Additionally, because of the massive aggregations, PDPs may cover up interesting dynamics associated with single examples. Finally PDPs suffer from sparsity issues, where some features' distributions lack a density of representatives. To fortify interpretations it is good practice to incorporate a rung on the $x - axis$ that captures the features distribution. One can also attempt to smooth the curves by introducing an artificial dense grid of feature, rather than calculating only over actual features that appear in the dataset.