Tree Methods

Max Kutschinski

2022-07-03

Baseline

A common-sense, non-machine learning baseline serves as a sanity check and is often used to establish a baseline of comparison for more advanced machine learning models.

In this case, bike demand can be assumed to be periodical with a daily period. Given hourly data, a common-sense baseline is to predict the bike demand at time t to be equal to the bike demand at time t-24.

$$\hat{x}_t = x_{t-24} \tag{1}$$

Random Forests

Random forests is an ensemble method for regression and classification tasks that is built on decision trees. It extends on the idea of bootstrap aggregation, or bagging, which is used to reduce the variance of a statistical learning method via averaging. In the context of regression trees, this is done by constructing B unpruned trees from B bootstrap samples and averaging the predictions as displayed in Eq. (1).

$$\hat{f}_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f^{*b}}(x) \tag{2}$$

One notable drawback of bagging is that the individual trees can be very correlated depending on how strong the predictors are. Random forests addresses this issue by randomly selecting a subset of the features at each split and thus de-correlating the trees.

XGBoost

XGBoost is an advanced implementation of Gradient Boosting, which is an ensemble method for regression and classification tasks that combines multiple weak learners into a stronger learner. In the context of decision trees, a weak learner is defined as a tree with a small number of terminal nodes. The trees are grown sequentially and they are fit on the residuals of the current fit as opposed to the outcome Y. This has the effect of capturing signal that is not yet accounted for by the current set of trees. In addition, each weak learner is shrunken down by some shrinkage factor before it is used, making boosting a “slow” learning approach.

The first step is to initialize the model with a constant value γ, which can be obtained by solving the following optimization problem:

$$F_0(x) = \underset{\gamma}{\operatorname{argmin}} \sum_{i=1}^{n} L(y_i, \gamma) \tag{3}$$

After specifying the number of base learners M, the following steps are repeated for each base learner from m=1 to m=M:

First, the pseudo-residuals are calculated for each ith training example.

$$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} \tag{4}$$

Then a base learner $h_m(x)$ is fit to the pseudo-residuals using the modified training set $\{(x_i, r_{im})_{i=1}^n\}$.

Lastly, the model is updated as follows:

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x) \tag{5}$$

$$\text{where } \gamma_m = \text{argmin}_\gamma \textstyle\sum_{i=1}^n L(y_i, F_{m-1}(x_i) + \gamma h_m(x_i))$$

XGBoost is a more regularized form of Gradient Boosting which uses L1 and L2 regularization to improve model generalization capabilities. It also allows for parallel processing, and has a built-in routine for handling missing values via its sparsity-aware split finding algorithm.