

EBM Models

An Explainable Boosting Machine (EBM) is a type of machine learning model that combines the principles of gradient boosting with explainability. Gradient boosting is a popular technique in machine learning that involves combining multiple weak learners to create a strong model. However, these models can be difficult to interpret and explain to stakeholders.

EBM addresses this challenge by generating a model that is both highly accurate and interpretable. It works by creating a set of simple and easily understandable rules that can be used to make predictions. These rules are learned through an iterative process that involves building decision trees, evaluating their performance, and then adding new trees to improve the model.

The key difference between EBM and traditional gradient boosting models is that EBM generates additive models that are linear in the inputs. This means that the model's predictions can be easily explained by identifying which inputs were important in making the prediction and how they influenced the final outcome.

The interpretability features of an Explainable Boosting Machine (EBM) include:

1. Rule-based representation: EBM generates a set of simple and easily understandable rules that can be used to make predictions. Each rule represents a decision that the model uses to determine the outcome. This rule-based representation allows stakeholders to understand how the model arrived at its predictions.
2. Importance of features: EBM can identify which input features were important in making a prediction. This allows stakeholders to understand which features have the most influence on the outcome and how they are related to each other.
3. Monotonicity constraints: EBM allows users to specify monotonicity constraints on the input features. This means that the model will always increase or decrease its prediction as the input feature increases or decreases, making it easier to understand how changes in the input features affect the outcome.
4. Global and local explanations: EBM provides both global and local explanations of the model's predictions. The global explanation identifies the overall trends and relationships between input features and the outcome, while the local explanation explains how the model arrived at a specific prediction for a particular instance.
5. Transparency and simplicity: EBM is designed to be transparent and simple, making it easier for stakeholders to understand how the model works and trust its predictions.

The Workings of EBM

EBM has a few major improvements over traditional GAMs. First, EBM learns each feature function f_j using modern machine learning techniques such as bagging and gradient boosting. The boosting procedure is carefully restricted to train on one feature at a time in round-robin fashion using a very low learning rate so that feature order does not matter. It round-robin cycles through features to mitigate the effects of co-linearity and to learn the best feature function f_j for each feature to show how each feature contributes to the model's prediction for the problem. Second, EBM can automatically detect and include pairwise interaction terms. This further increases accuracy while maintaining intelligibility. EBM is a fast implementation of the GA2M algorithm, written in C++ and Python. The implementation is parallelizable, and takes advantage of joblib to provide multi-core and multi-machine parallelization. The algorithmic details for the training procedure, selection of pairwise interaction terms, and case studies can be found in.

EBMs are highly intelligible because the contribution of each feature to a final prediction can be visualized and understood by plotting f_j . Because EBM is an additive model, each feature contributes to predictions in a modular way that makes it easy to reason about the contribution of each feature to the prediction.

To make individual predictions, each function f_j acts as a lookup table per feature, and returns a term contribution. These term contributions are simply added up, and passed through the link function g to compute the final prediction. Because of the modularity (additivity), term contributions can be sorted and visualized to show which features had the most impact on any individual prediction.

To keep the individual terms additive, EBM pays an additional training cost, making it somewhat slower than similar methods. However, because making predictions involves simple additions and lookups inside of the feature functions f_j , EBMs are one of the fastest models to execute at prediction time. EBM's light memory usage and fast predict times makes it particularly attractive for model deployment in production.

Extra Information to use EBM via InterpretML to interpret results (EBM in Python):

InterpretML's built-in interactive and exploratory visualization dashboard gives data scientists a wide range of insights about their dataset, model performance, and model explanations. InterpretML includes a new interpretability algorithm—the Explainable Boosting Machine, which is a highly intelligible and explainable—“glassbox”—model, with accuracy that's comparable to machine learning methods like random forests and boosted trees. InterpretML is a community-driven initiative and invites further expansion by researchers and data scientists.

InterpretML introduces a new glassbox model, the Explainable Boosting Machine (EBM). EBM, developed by Microsoft Research, is an interpretable model that uses machine learning techniques like bagging, gradient boosting, and automatic interaction detection to breathe new life into traditional generalized additive models (GAMs). This makes EBMs as accurate as state-of-the-art techniques like random forests and gradient boosted trees. However, unlike these, which are black-box models, EBMs produce lossless explanations and are editable by domain experts.

Glassbox models means learning algorithms that are designed to be interpretable i.e Linear Models , Decision trees .Glassbox Models typically provides exact explainability , that is you can trace and reason about how any Glassbox makes its decisions . Besides the regular scikit learn models , Glassbox contains a new model called Explainable Boosting Machine (EBM)

Glassbox Models vs. Blackbox explanations

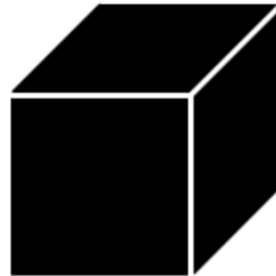
Algorithms that are designed to be interpretable are called Glassbox models. These include algorithms like simple decision trees, rule lists, linear models, etc. Glassbox approaches typically provide exact or lossless explainability. This means it is possible to trace and reason about how any prediction is made. The interpretation of GlassBox models is Model-specific because each method is based on some specific model's internals. For instance, the interpretation of weights in linear models count towards model-specific explanations.


Blackbox explainers, on the contrary, are model agnostic. They can be applied to any model, and such are post-hoc in nature since they are applied after the model has been trained. Blackbox explainers work by treating the model as a BlackBox and assume that they only have access to the model's inputs and outputs. They are particularly useful for complex algorithms like boosted trees and deep neural nets. Blackbox explainers work by repeatedly perturbing the input and analyzing the resultant changes in the model output. The examples include SHAP, LIME, Partial Dependence Plots, etc., to name a few.

Glassbox Approaches



Blackbox Approaches



Also called Intrinsic or model based	Also called Post-hoc techniques i.e analyzing the model after training
Learning algorithms that are designed to be interpretable,	The algorithms are not inherently interpretable, hence called 
Examples include simple decision trees, Rule lists, linear models etc	Examples include SHAP, LIME, Partial Dependency plots etc
provide exact or lossless explainability	Provide approximate explainability.
Trained directly on raw data	Trained on Existing models or pipelines

Prediction with EBM: Using ROC curve to explain the prediction quality of EBM .

Pros

- EBM creates new knowledge
- EBM helps debug data
- EBM helps understand data

Explanation of these pros in the examples provided at this link:

<https://pub.towardsai.net/explainable-boosting-machines-c71b207231b5>