Package ‘interpret’

October 12, 2020

Title  Fit Interpretable Machine Learning Models
Version  0.1.26
Date  2020-10-09

Description  Package for training interpretable machine learning models. Historically, the most interpretable machine learning models were not very accurate, and the most accurate models were not very interpretable. Microsoft Research has developed an algorithm called the Explainable Boosting Machine (EBM) which has both high accuracy and interpretable characteristics. EBM uses machine learning techniques like bagging and boosting to breathe new life into traditional GAMs (Generalized Additive Models). This makes them as accurate as random forests and gradient boosted trees, and also enhances their intelligibility and editability. Details on the EBM algorithm can be found in the paper by Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad (2015, <doi:10.1145/2783258.2788613>).

URL  https://github.com/interpretml/interpret
BugReports  https://github.com/interpretml/interpret/issues
License  MIT + file LICENSE
Depends  R (>= 3.0.0)
NeedsCompilation  yes
SystemRequirements  C++11
Author  Samuel Jenkins [aut],
        Harsha Nori [aut],
        Paul Koch [aut],
        Rich Caruana [aut, cre],
        Microsoft Corporation [cph]
Maintainer  Rich Caruana <interpretml@outlook.com>
Repository  CRAN
Date/Publication  2020-10-12 21:20:02 UTC

R topics documented:

   ebm_classify ................................................................. 2
ebm_classify

Build an EBM classification model

Description
Builds a classification model

Usage

```r
ebm_classify(
  X,
  y,
  max_bins = 255,
  outer_bags = 16,
  inner_bags = 0,
  learning_rate = 0.01,
  validation_size = 0.15,
  early_stopping_rounds = 50,
  early_stopping_tolerance = 1e-4,
  max_rounds = 5000,
  max_leaves = 3,
  min_samples_leaf = 2,
  random_state = 42
)
```

Arguments

- `X` features
- `y` targets
- `max_bins` number of bins to create
- `outer_bags` number of outer bags
- `inner_bags` number of inner bags
- `learning_rate` learning rate
- `validation_size` amount of data to use for validation
- `early_stopping_rounds` how many rounds without improvement before we quit
- `early_stopping_tolerance` how much does the round need to improve by to be considered as an advancement
- `max_rounds` number of boosting rounds
\textit{ebm\_predict\_proba}

\begin{itemize}
\item \texttt{max\_leaves} \hspace{1cm} how many leaves allowed
\item \texttt{min\_samples\_leaf} \hspace{1cm} number of samples required for a split
\item \texttt{random\_state} \hspace{1cm} random seed
\end{itemize}

\textbf{Value}

Returns an EBM model

\textbf{Examples}

\begin{verbatim}
data(mtcars)
  X <- subset(mtcars, select = -c(vs))
  y <- mtcars$vs

  set.seed(42)
  data_sample <- sample(length(y), length(y) * 0.8)

  X_train <- X[data_sample, ]
  y_train <- y[data_sample]
  X_test <- X[-data_sample, ]
  y_test <- y[-data_sample]

  ebm <- ebm\_classify(X_train, y_train)
\end{verbatim}

\begin{verbatim}
  ebm\_predict\_proba  ebm\_predict\_proba
\end{verbatim}

\textbf{Description}

Predicts probabilities using an EBM model

\textbf{Usage}

\begin{verbatim}
  ebm\_predict\_proba(
    model,
    X
  )
\end{verbatim}

\textbf{Arguments}

\begin{itemize}
\item \texttt{model} \hspace{1cm} the model
\item \texttt{X} \hspace{1cm} features
\end{itemize}

\textbf{Value}

returns the probabilities predicted
Examples

data(mtcars)
X <- subset(mtcars, select = -c(vs))
y <- mtcars$vs

set.seed(42)
data_sample <- sample(length(y), length(y) * 0.8)

X_train <- X[data_sample, ]
y_train <- y[data_sample]
X_test <- X[-data_sample, ]
y_test <- y[-data_sample]

ebm <- ebm_classify(X_train, y_train)
proba_test <- ebm_predict_proba(ebm, X_test)

Description

Shows the GAM plot for a single feature

Usage

ebm_show(  
  model,  
  name  
)

Arguments

model the model
name the name of the feature to plot

Value

None

Examples

data(mtcars)
X <- subset(mtcars, select = -c(vs))
y <- mtcars$vs

set.seed(42)
data_sample <- sample(length(y), length(y) * 0.8)

X_train <- X[data_sample, ]
y_train <- y[data_sample]
X_test <- X[-data_sample, ]
y_test <- y[-data_sample]

ebm <- ebm_classify(X_train, y_train)
ebm_show(ebm, "mpg")
Index

ebm_classify, 2
ebm_predict_proba, 3
ebm_show, 4