Title: Bootstrap aggregating

URL: https://en.wikipedia.org/wiki/Bootstrap_aggregating

PageID: 1307911

Categories: Category:Computational statistics, Category:Ensemble learning, Category:Machine learning algorithms

-----

Supervised learning

Unsupervised learning

Semi-supervised learning

Self-supervised learning

Reinforcement learning

Meta-learning

Online learning

Batch learning

Curriculum learning

Rule-based learning

Neuro-symbolic AI

Neuromorphic engineering

Quantum machine learning

Classification

Generative modeling

Regression

Clustering

Dimensionality reduction

Density estimation

Anomaly detection

Data cleaning

AutoML

Association rules

Semantic analysis

Structured prediction

Feature engineering

Feature learning

Learning to rank

Grammar induction

Ontology learning

Multimodal learning

Apprenticeship learning

Decision trees

Ensembles Bagging Boosting Random forest

Bagging

Boosting

Random forest

k -NN

Linear regression

Naive Bayes

Artificial neural networks

Logistic regression

Perceptron

Relevance vector machine (RVM)

Support vector machine (SVM)

BIRCH

CURE

Hierarchical

k -means

Fuzzy

Expectation–maximization (EM)

DBSCAN

OPTICS

Mean shift

Factor analysis

CCA

ICA

LDA

NMF

PCA

PGD

t-SNE

SDL

Graphical models Bayes net Conditional random field Hidden Markov

Bayes net

Conditional random field

Hidden Markov

RANSAC

k -NN

Local outlier factor

Isolation forest

Autoencoder

Deep learning

Feedforward neural network

Recurrent neural network LSTM GRU ESN reservoir computing

LSTM

GRU

ESN

reservoir computing

Boltzmann machine Restricted

Restricted

GAN

Diffusion model

SOM

Convolutional neural network U-Net LeNet AlexNet DeepDream

U-Net

LeNet

AlexNet

DeepDream

Neural field Neural radiance field Physics-informed neural networks

Neural radiance field

Physics-informed neural networks

Transformer Vision

Vision

Mamba

Spiking neural network

Memtransistor

Electrochemical RAM (ECRAM)

Q-learning

Policy gradient

SARSA

Temporal difference (TD)

Multi-agent Self-play

Self-play

Active learning

Crowdsourcing

Human-in-the-loop

Mechanistic interpretability

RLHF

Coefficient of determination

Confusion matrix

Learning curve

ROC curve

Kernel machines

Bias–variance tradeoff

Computational learning theory

Empirical risk minimization

Occam learning

PAC learning

Statistical learning

VC theory

Topological deep learning

AAAI

ECML PKDD

NeurIPS

ICML

ICLR

IJCAI

ML

JMLR

Glossary of artificial intelligence

List of datasets for machine-learning research List of datasets in computer vision and image processing

List of datasets in computer vision and image processing

Outline of machine learning

v

t

e

Bootstrap aggregating , also called bagging (from b ootstrap agg regat ing) or bootstrapping , is a machine learning (ML) ensemble meta-algorithm designed to improve the stability and accuracy of ML classification and regression algorithms. It also reduces variance and overfitting . Although it is usually applied to decision tree methods, it can be used with any type of method. Bagging is a special case of the ensemble averaging approach.

Description of the technique

Given a standard training set D {\displaystyle D} of size n {\displaystyle n} , bagging generates m {\displaystyle m} new training sets D i {\displaystyle D_{i}} , each of size n ′ {\displaystyle n'} , by sampling from D {\displaystyle D} uniformly and with replacement . By sampling with replacement, some observations may be repeated in each D i {\displaystyle D_{i}} . If n ′ = n {\displaystyle n'=n} , then for large n {\displaystyle n} the set D i {\displaystyle D_{i}} is expected to have the fraction (1 - 1/ e) (~63.2%) of the unique samples of D {\displaystyle D} , the rest being duplicates. [1] This kind of sample is known as a bootstrap sample. Sampling with replacement ensures each bootstrap is independent from its peers, as it does not depend on previous chosen samples when sampling. Then, m {\displaystyle m} models are fitted using the above bootstrap samples and combined by averaging the output (for regression) or voting (for classification).

Bagging leads to "improvements for unstable procedures", [2] which include, for example, artificial neural networks , classification and regression trees , and subset selection in linear regression . [3] Bagging was shown to improve preimage learning. [4] [5] On the other hand, it can mildly degrade the performance of stable methods such as k -nearest neighbors . [2]

Process of the algorithm

Key Terms

There are three types of datasets in bootstrap aggregating. These are the original, bootstrap, and out-of-bag datasets. Each section below will explain how each dataset is made except for the original dataset. The original dataset is whatever information is given.

Creating the bootstrap dataset

The bootstrap dataset is made by randomly picking objects from the original dataset. Also, it must be the same size as the original dataset. However, the difference is that the bootstrap dataset can have duplicate objects. Here is a simple example to demonstrate how it works along with the illustration below:

Suppose the original dataset is a group of 12 people. Their names are Emily, Jessie, George, Constantine, Lexi, Theodore, John, James, Rachel, Anthony, Ellie, and Jamal.

By randomly picking a group of names, let us say our bootstrap dataset had James, Ellie, Constantine, Lexi, John, Constantine, Theodore, Constantine, Anthony, Lexi, Constantine, and Theodore. In this case, the bootstrap sample contained four duplicates for Constantine, and two duplicates for Lexi, and Theodore.

Creating the out-of-bag dataset

The out-of-bag dataset represents the remaining people who were not in the bootstrap dataset. It can be calculated by taking the difference between the original and the bootstrap datasets. In this case, the remaining samples who were not selected are Emily, Jessie, George, Rachel, and Jamal. Keep in mind that since both datasets are sets, when taking the difference the duplicate names are ignored in the bootstrap dataset. The illustration below shows how the math is done:

Application

Creating the bootstrap and out-of-bag datasets is crucial since it is used to test the accuracy of ensemble learning algorithms like random forest . For example, a model that produces 50 trees using the bootstrap/out-of-bag datasets will have a better accuracy than if it produced 10 trees. Since the algorithm generates multiple trees and therefore multiple datasets the chance that an object is left out of the bootstrap dataset is low. The next few sections talk about how the random forest algorithm works in more detail.

Creation of Decision Trees

The next step of the algorithm involves the generation of decision trees from the bootstrapped dataset. To achieve this, the process examines each gene/feature and determines for how many samples the feature's presence or absence yields a positive or negative result. This information is then used to compute a confusion matrix , which lists the true positives, false positives, true negatives, and false negatives of the feature when used as a classifier. These features are then ranked according to various classification metrics based on their confusion matrices. Some common metrics include estimate of positive correctness (calculated by subtracting false positives from true positives), measure of "goodness", and information gain . These features are then used to partition the samples into two sets: those that possess the top feature, and those that do not.

The diagram below shows a decision tree of depth two being used to classify data. For example, a data point that exhibits Feature 1, but not Feature 2, will be given a "No". Another point that does not exhibit Feature 1, but does exhibit Feature 3, will be given a "Yes".

This process is repeated recursively for successive levels of the tree until the desired depth is reached. At the very bottom of the tree, samples that test positive for the final feature are generally classified as positive, while those that lack the feature are classified as negative. These trees are then used as predictors to classify new data.

Random Forests

The next part of the algorithm involves introducing yet another element of variability amongst the bootstrapped trees. In addition to each tree only examining a bootstrapped set of samples, only a small but consistent number of unique features are considered when ranking them as classifiers. This means that each tree only knows about the data pertaining to a small constant number of features, and a variable number of samples that is less than or equal to that of the original dataset. Consequently, the trees are more likely to return a wider array of answers, derived from more diverse knowledge. This results in a random forest , which possesses numerous benefits over a single decision tree generated without randomness. In a random forest, each tree "votes" on whether or not to classify a sample as positive based on its features. The sample is then classified based on majority vote. An example of this is given in the diagram below, where the four trees in a random forest vote on whether or not a patient with mutations A, B, F, and G has cancer. Since three out of four trees vote yes, the patient is then classified as cancer positive.

Because of their properties, random forests are considered one of the most accurate data mining algorithms, are less likely to overfit their data, and run quickly and efficiently even for large datasets. [6] They are primarily useful for classification as opposed to regression , which attempts to draw observed connections between statistical variables in a dataset. This makes random forests particularly useful in such fields as banking, healthcare, the stock market, and e-commerce where it is important to be able to predict future results based on past data. [7] One of their applications would be as a useful tool for predicting cancer based on genetic factors, as seen in the above example.

There are several important factors to consider when designing a random forest. If the trees in the random forests are too deep, overfitting can still occur due to over-specificity. If the forest is too large, the algorithm may become less efficient due to an increased runtime. Random forests also do not generally perform well when given sparse data with little variability. [7] However, they still have numerous advantages over similar data classification algorithms such as neural networks , as they are much easier to interpret and generally require less data for training. [citation needed] As an integral component of random forests, bootstrap aggregating is very important to classification algorithms, and provides a critical element of variability that allows for increased accuracy when analyzing new data, as discussed below.

Improving Random Forests and Bagging

While the techniques described above utilize random forests and bagging (otherwise known as bootstrapping), there are certain techniques that can be used in order to improve their execution and voting time, their prediction accuracy, and their overall performance. The following are key steps in creating an efficient random forest:

Specify the maximum depth of trees: Instead of allowing the random forest to continue until all nodes are pure, it is better to cut it off at a certain point in order to further decrease chances of overfitting.

Prune the dataset: Using an extremely large dataset may create results that are less indicative of the data provided than a smaller set that more accurately represents what is being focused on. Continue pruning the data at each node split rather than just in the original bagging process.

Continue pruning the data at each node split rather than just in the original bagging process.

Decide on accuracy or speed: Depending on the desired results, increasing or decreasing the number of trees within the forest can help. Increasing the number of trees generally provides more accurate results while decreasing the number of trees will provide quicker results.

Algorithm (classification)

For classification, use a training set D {\displaystyle D} , Inducer I {\displaystyle I} and the number of bootstrap samples m {\displaystyle m} as input. Generate a classifier C ∗ {\displaystyle C^{*}} as output [12]

Create m {\displaystyle m} new training sets D i {\displaystyle D_{i}} , from D {\displaystyle D} with replacement

Classifier C i {\displaystyle C_{i}} is built from each set D i {\displaystyle D_{i}} using I {\displaystyle I} to determine the classification of set D i {\displaystyle D_{i}}

Finally classifier C ∗ {\displaystyle C^{*}} is generated by using the previously created set of classifiers C i {\displaystyle C_{i}} on the original dataset D {\displaystyle D} , the classification predicted most often by the sub-classifiers C i {\displaystyle C_{i}} is the final classification

Example: ozone data

To illustrate the basic principles of bagging, below is an analysis on the relationship between ozone and temperature (data from Rousseeuw and Leroy [clarification needed] (1986), analysis done in R).

The relationship between temperature and ozone appears to be nonlinear in this dataset, based on the scatter plot. To mathematically describe this relationship, LOESS smoothers (with bandwidth 0.5) are used. Rather than building a single smoother for the complete dataset, 100 bootstrap samples were drawn. Each sample is composed of a random subset of the original data and maintains a semblance of the master set's distribution and variability. For each bootstrap sample, a LOESS smoother was fit. Predictions from these 100 smoothers were then made across the range of the data. The black lines represent these initial predictions. The lines lack agreement in their predictions and tend to overfit their data points: evident by the wobbly flow of the lines.

By taking the average of 100 smoothers, each corresponding to a subset of the original dataset, we arrive at one bagged predictor (red line). The red line's flow is stable and does not overly conform to any data point(s).

Advantages and disadvantages

Advantages:

Many weak learners aggregated typically outperform a single learner over the entire set, and have less overfit

Reduces variance in high-variance low-bias weak learner, [13] which can improve efficiency (statistics)

Can be performed in parallel , as each separate bootstrap can be processed on its own before aggregation. [14]

Disadvantages:

For a weak learner with high bias, bagging will also carry high bias into its aggregate [13]

Loss of interpretability of a model.

Can be computationally expensive depending on the dataset.

History

The concept of bootstrap aggregating is derived from the concept of bootstrapping which was developed by Bradley Efron. [15] Bootstrap aggregating was proposed by Leo Breiman who also coined the abbreviated term "bagging" (b ootstrap agg regat ing). Breiman developed the concept of bagging in 1994 to improve classification by combining classifications of randomly generated training sets. He argued, "If perturbing the learning set can cause significant changes in the predictor constructed, then bagging can improve accuracy". [3]

See also

Boosting (machine learning)

Bootstrapping (statistics)

Cross-validation (statistics)

Out-of-bag error

Random forest

Random subspace method (attribute bagging)

Resampled efficient frontier

Predictive analysis: Classification and regression trees

References

Further reading

Breiman, Leo (1996). "Bagging predictors". Machine Learning . 24 (2): 123– 140. CiteSeerX 10.1.1.32.9399 . doi : 10.1007/BF00058655 . S2CID 47328136 .

Alfaro, E., Gámez, M. and García, N. (2012). "adabag: An R package for classification with AdaBoost.M1, AdaBoost-SAMME and Bagging" . {{ cite journal }} : Cite journal requires |journal= (help) CS1 maint: multiple names: authors list (link)

Kotsiantis, Sotiris (2014). "Bagging and boosting variants for handling classifications problems: a survey". Knowledge Eng. Review . 29 (1): 78– 100. doi : 10.1017/S0269888913000313 . S2CID 27301684 .

Boehmke, Bradley; Greenwell, Brandon (2019). "Bagging". Hands-On Machine Learning with R . Chapman & Hall. pp. 191– 202. ISBN 978-1-138-49568-5 .