Proseminar
Advanced topics in machine learning

Bagging, Boosting, and Ensemble Learning

Author

Florian Zager

Matrikelnummer: *2489018*

Date:

Karlsruhe, December 28, 2023

Abstract — Zusammenfassung

Mandatory. Short summary of the report.

Contents

1 Introduction

Mandatory. Questions like: What is the topic of this work, what's the broader context (topic of the proseminar), why is it relevant?

2 Ensemble Learning

Ensemble learning is an advanced machine learning approach that combines the strengths of multiple smaller learning algorithms to improve predictive performance. The concept behind ensemble learning is analogous to the "wisdom of crowds". Which describes, that a crowd, on average, makes collectively better decisions, than any single member of it.

Just as a diverse group of people can provide a more accurate collective decision than an individual, in ensemble learning, a combination of learning algorithms often predicts more accurately than an individual learning algorithm. This approach is based on the principle that a diverse set of learning algorithms can capture different patterns or trends in the data, leading to more robust and accurate predictions.

To be more precise, ensemble methods use multiple smaller learning algorithms, which specialize in small aspects of the problem. However, by combining these algorithms, the ensemble often achieves better predictive performance than the used algorithm could achieve alone because they complement each others strengths and weaknesses.

So the goal of ensemble learning is to achieve a better predictive performance. Nevertheless, it comes at the cost of increased computational resources for training as well as prediction and storage.

Overall, there are many different ensemble methods, such as bagging and boosting, which we will go into more detail in this report. However, there are many more like stacking and blending.

2.1 Bagging

Bootstrap Aggregating, commonly known as bagging, is an ensemble learning method developed by Breiman (1996). The models for the ensemble get trained individually by using the bootstrapping technique. Bootstrapping involves creating random subsets of the original training dataset. The subsets are created by drawing random data points with replacement and have the same size as the original training dataset. This means that data points can be chosen more than once and that some data points might not be in the subset. The models can be trained in parallel, because of the individual training.

Figure 1 shows how bootstrapping might work on an imaginary dataset. In subset 1 each class is equally distributed. Subset 2 has a focus on the red and orange bubbles. The subset N has a strong focus on blue, however it doesn't even have a single orange bubble. This will probably mean that the model trained on subset N will be very good at predicting blue, but not very accurate with orange.

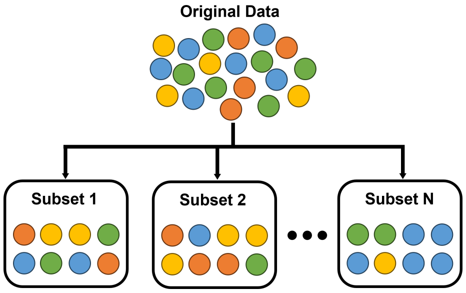

Figure 1: Creating subsets with bootstrapping.

In the bagging ensemble, each model makes its prediction independently. Because of that the predictions of the individual models can be run in parallel, similar to the training. Once every model in the ensemble has made their prediction, they get aggregated to form a final ensemble prediction. The method of aggregation depends on the problem that is being solved by the bagging ensemble. For classification problems, a common method is majority voting. Each model votes for a particular class. The class that receives the most votes is chosen for the final ensemble prediction. For regression problems, the predictions of the individual models are typically averaged.

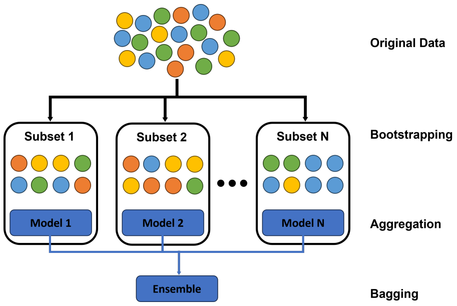

Figure 2: Bagging prediction example

Bagging can be particularly effective when using base learning algorithms that have high variance or tend to overfit quickly. By bootstrapping and averaging the predictions, the variance gets reduced and potential overfitting is avoided. The same goes for unstable learning algorithms, that produce significantly different results on small changes in the data. That's why bagging is often used with decision trees as they tend to be unstable and to have high variance. Additionally, bagging can be very helpful when dealing with noisy, imbalanced datasets or datasets with missing data. The bootstrapping helps to create diverse datasets with each class being adequately represented and also averaging out the noise in combination with aggregation.

All in all, bagging can help to reduce variance, prevent overfitting, and to build a more resilient, robust, and generalized model.

2.2 Random Forest

2.3 Boosting

Boosting (Schapire, 1990) is another ensemble learning method like bagging. However, unlike bagging, boosting trains the models sequentially, where each model learns from the mistakes of its predecessors. In the first step, the data subset for the first model gets created and a model is trained on it. Initially, the dataset is equally weighted, similar to the initial setup in bagging: data points are drawn with replacement until the subset has the same size as the original dataset. In the second step, the performance of the model is evaluated. The weight gets increased for incorrectly predicted samples and decreased for correctly predicted samples. In the third step, the next model is trained on the dataset with the updated weights. This weight adjustment makes the next model focus more on the data points that previous models predicted incorrectly. Steps two and three, the process of updating weights and training new models now gets repeated until all models are trained. Different from bagging, boosting can't be trained in parallel, because each model is trained based on the previous model's performance.

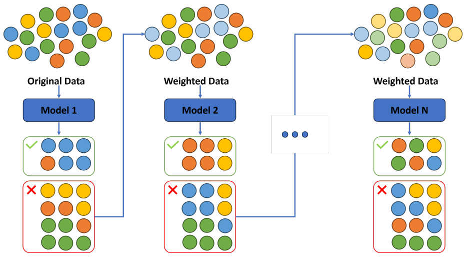

Figure 3: Boosting training example

In contrast to the training, each model makes its individual prediction in boosting. This means the predictions can be done in parallel. Unlike bagging where each model's prediction is given equal weight, boosting uses a weighted average approach to combine the individual predictions. This means each model has its own weight based on the performance in the training phase. For classification problems, this means that the vote of each model can have a different influence on the final ensemble prediction. In the case of regression problems, the final prediction is the weighted average of the predictions from all models.

Boosting is ideal to use when the base learning algorithm suffers from high bias. This is combated through the way the ensemble is trained, as each model focuses on the weaknesses of the previous one. Additionally, boosting is particularly effective with high-dimensional data, where the number of features is large. Other models might not be able to capture the complexity or more complex models might overfit. However, boosting can strike a balance by incrementally building complexity and focusing on features that improve the predictive performance. Furthermore, it's important to note that boosting works best on datasets with little to no outliers. The ensemble strongly focuses on correcting errors. Outliers can disproportionately influence the direction of the learning process, leading to overfitting.

To put it in a nutshell, boosting can help to reduce bias, avoid underfitting, and can help to build overall precise models. Nevertheless, you have to be aware of the challenges with boosting e.g. noisy data.

2.4 Gradient Boosting

2.5 Extreme Gradient Boosting

3 Examples

In the examples, we aim to compare various ensemble learning algorithms - Bagging, Boosting (AdaBoost - Freund, Schapire, et al. (1996)), Random Forest, and Gradient Boosting - against Decision Trees across different datasets. First, we're looking for the optimal depth of the Decision Tree which maximizes the precision. This Decision Tree will be used as the base learning algorithm for the ensemble learning algorithms. In order to determine the accuracy of the Decision Tree, the datasets are split into a validation set (20%) and the rest (80%). The rest is used for a 10-fold cross-validation, where the accuracy is calculated by averaging the accuracy of all folds. The same method is used to find the best hyperparameters for Bagging, Boosting, Random Forest and Gradient Boosting. The learning rate is always 0.1 if applicable. This approach allows us to comprehensively assess each algorithm's effectiveness and draw meaningful comparisons.

3.1 Example 1

In the first example we're using the Breast Cancer Wisconsin (Diagnostic) dataset by Wolberg, Mangasarian, Street, and Street (1995). The dataset originates from the University of Wisconsin Hospitals. It contains 30 features and has 569 data points. The goal of the dataset is to classify if breast cancer is benign or malignant.

Learning algorithm	Worst fold	Best fold	Average fold	Validation
Decision Tree	0.85	0.98	0.93	0.93
Bagging	0.84	**1.00**	0.93	0.96
Random Forest	0.85	**1.00**	**0.95**	0.97
Boosting	0.87	**1.00**	**0.95**	**0.99**
Gradient Boosting	**0.89**	**1.00**	0.94	**0.99**

Table 1: Breast Cancer Wisconsin (Diagnostic) model accuracy

3.2 Example 2

In the second example we're using the Heart Disease Cleveland dataset by Janosi, Steinbrunn, Pfisterer, and Detrano (1988). It contains 13 features and has 303 data points. The goal of the dataset is to determine if a patient has a heart disease.

Learning algorithm	Worst fold	Best fold	Average fold	Validation
Decision Tree	0.60	**0.92**	0.81	0.77
Bagging	0.72	0.88	0.81	0.82
Random Forest	**0.75**	0.88	**0.82**	**0.85**
Boosting	0.63	0.88	0.77	0.84
Gradient Boosting	0.60	0.88	0.79	0.80

Table 2: Heart Disease Cleveland model accuracy

4 Summary and conclusion

Mandatory. Short summary of the most important aspects of the report. If possible: What are open challenges?

- Bagging vs. Boosting - whats the difference?

References

Breiman, L. (1996, Aug 01). Bagging predictors. *Machine Learning*, *24*(2), 123-140. Retrieved from `https://doi.org/10.1007/BF00058655` doi: 10.1007/BF00058655

Freund, Y., Schapire, R. E., et al. (1996). Experiments with a new boosting algorithm. In *icml* (Vol. 96, pp. 148–156).

Janosi, A., Steinbrunn, W., Pfisterer, M., & Detrano, R. (1988). *Heart disease.* UCI Machine Learning Repository. (DOI: https://doi.org/10.24432/C52P4X)

Schapire, R. E. (1990, Jun 01). The strength of weak learnability. *Machine Learning*, *5*(2), 197-227. Retrieved from `https://doi.org/10.1007/BF00116037` doi: 10.1007/BF00116037

Wolberg, W., Mangasarian, O., Street, N., & Street, W. (1995). *Breast cancer wisconsin (diagnostic).* UCI Machine Learning Repository. (DOI: https://doi.org/10.24432/C5DW2B)