Ensemble Techniques

Ensemble Techniques - Topics

- Ensemble Methods
- Bias-Variance error
- Bagging concept
- Boosting concept
- Random Forest
- Case Study

Ensemble methods

greatlearning

- Ensemble is a group of models that are used together for prediction both in classification and regression class.

- We employ various methods to achieve this for eg: tuning hyper parameters, up-sampling/ down-sampling etc.

- The motivation behind ensemble is the belief that a committee of experts working together are more likely to be accurate than individual experts.

Ensemble methods- Characteristics

greatlearning

1. For effective ensemble we have to ensure -
    a. The models are as different from each other as possible.
    b. The errors of each model should be independent.

2. The models in the ensemble
    a. Are not regularized and hence each model tend to overfit prone to variance errors.

    b. The variance errors across all models put together cancel out at the time of aggregation

Bias-Variance error

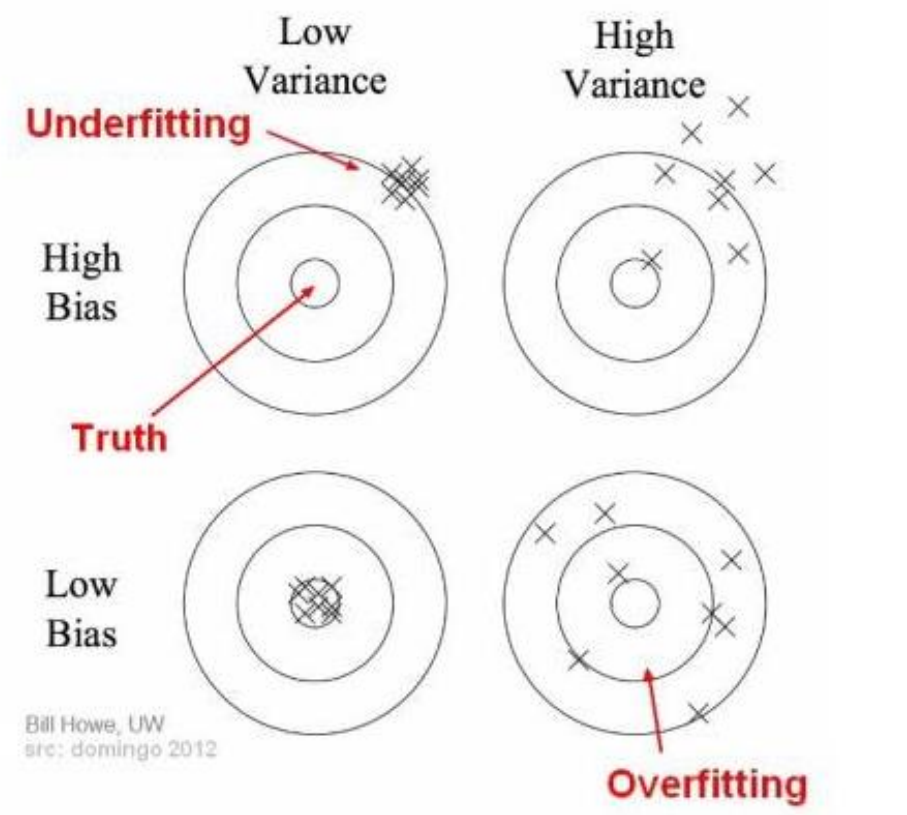
- Bias are the simplifying assumptions made by a model to make the target function easier to learn.

- Variance is the amount that the estimate of the target function will change if different training data was used.

- The goal of any supervised machine learning algorithm is to achieve low bias and low variance. In turn the algorithm should achieve good prediction performance.

(P.S. The parameterization of machine learning algorithms is often a battle to balance out bias and variance)

Bias and variance using bulls-eye diagram



Two Families of Ensemble

Two families of ensemble methods are usually distinguished :

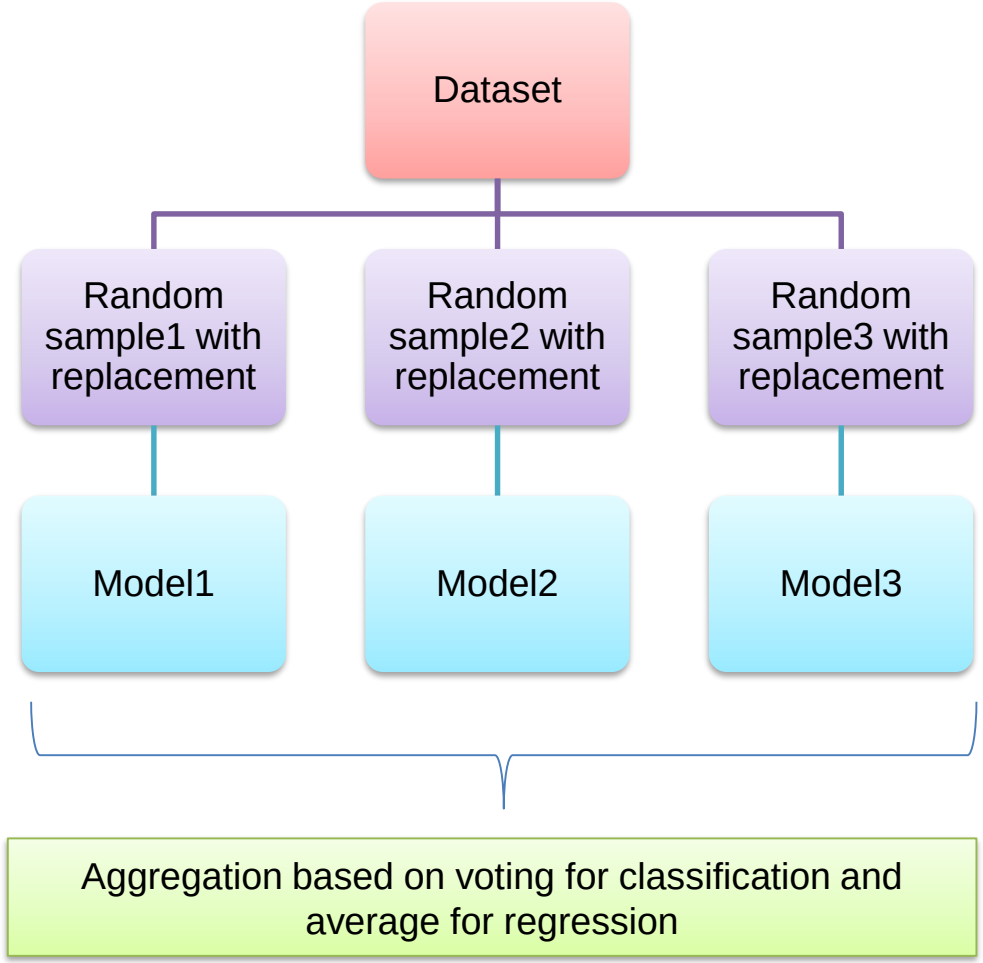
1. Averaging methods: The driving principle is to build several estimators independently and then to average/vote their predictors.

2. Boosting methods: Base estimators are built sequentially and one tries to reduce the bias of the combined estimator.

Bagging (Bootstrap Aggregation)

- Designed to improve the stability of classification and regression models.
- It reduces variance errors and helps to avoid overfitting.
- Can be used with any type of machine learning model, mostly used with decision tree.
- Use **sampling with replacement** to generate multiple samples of a given size.

Bagging example



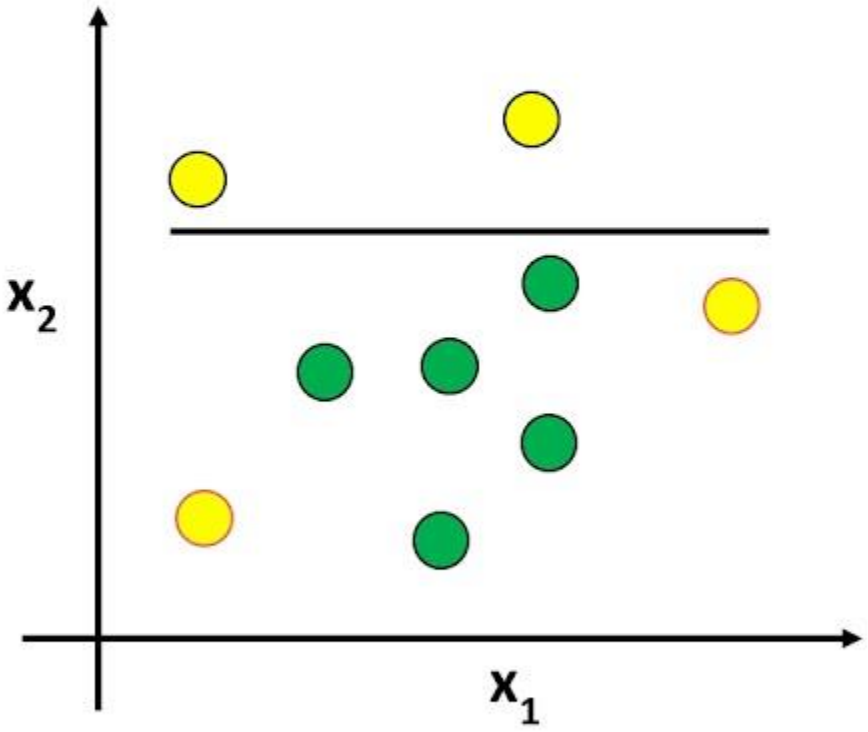
Boosting concept

- Boosting is a general ensemble method that creates a strong classifier from a number of weak classifiers.

- Similar to bagging, but the learners are grown sequentially; except for the first, each subsequent learner is grown from previously grown learners.

- If the learner is a Decision tree, each of the trees can be small, with just a few terminal nodes (determined by the parameter d supplied)

Boosting



For example: we want to classify the dots
~ two **x** features + two output classes (yellow and green)

Boosting: combines very simple weak learners such as **decision trees** with depth **1** (capable of linear classification)

→ the classifier made **2** mistakes: two yellow dots are misclassified

→ boosting algorithm: in the next iteration it will focus on the misclassified items

Increase the **w** weights: for misclassified items
Decrease the **w** weights: for correctly classified items

Types of Boosting

AdaBoosting: It is so called as the weights are re-assigned to each instance, with higher weights to incorrectly classified instance.

Gradient Boosting: By fitting new models to the residuals, the overall learner gradually improves in areas where residuals are initially high.

greatlearning

Difference between Adaboost and GradientBoost

AdaBoost	GradientBoost
Both AdaBoost and Gradient Boost use a base weak learner and they try to boost the performance of a weak learner by iteratively shifting the focus towards problematic observations that were difficult to predict. At the end, a strong learner is formed by addition (or weighted addition) of the weak learners.	
In AdaBoost, shift is done by up-weighting observations that were misclassified before.	Gradient boost identifies difficult observations by large residuals computed in the previous iterations.
In AdaBoost "shortcomings" are identified by high-weight data points.	In Gradientboost "shortcomings" are identified by gradients.
Exponential loss of AdaBoost gives more weights for those samples fitted worse.	Gradient boost further dissect error components to bring in more explanation.
AdaBoost is considered as a special case of Gradient boost in terms of loss function, in which exponential losses.	Concepts of gradients are more general in nature.

Random Forest

- Each tree in the ensemble is built from a sample drawn with replacement (bootstrap) from the training set.
- In addition, when splitting a node during the construction of a tree, the split that is chosen is no longer the best split among all the features.
- Instead, the split that is picked is the best split among a random subset of the features.
- Due to averaging, its variance decreases, usually more than compensating the increase in bias

Case Study - Wine quality production

greatlearning

Context

We will continue with the case study used in the last week content. We used wines data and constructed a decision tree.

The dataset is related to red variants of the Portuguese “Vinho Verde” wine. Due to privacy and logistic issues, only physicochemical (inputs) and sensory (output) variables are available (eg: there is no data about grape types, wine brand, wine selling price, etc.)

These datasets can be viewed as classification or regression tasks. The classes are ordered and not balanced (eg, there are much more normal wines than excellent or poor ones).

Dataset

https://www.kaggle.com/uciml/red-wine-quality-cortez-et-al-2009

Problem Statement

Wine Quality Prediction - Here, we will apply a method of assessing wine quality using a Decision Tree, and test it against the wine-quality dataset from the UC Irvine machine learning repository. The wine dataset is a classic and very easy multi-class classification dataset. We will use the techniques learnt in this module to improve our model.

Data Attributes

1. Fixed Acidity
2. Volatile acidity
3. Citric acid
4. Residual sugar
5. Chlorides
6. Free sulphur dioxide
7. Total sulphur dioxide
8. Density
9. pH
10. Sulphates
11. Alcohol
12. Quality

Steps to follow

1. Import all necessary libraries and load the data
2. Print the descriptive statistics of each & every column using describe() function
3. Using univariate analysis check the individual attributes for their basic statistic such as central values, spread, tails etc.
4. Use pairplots and correlation method to observe the relationship between different variables and state your insights.
5. Split the wine_df into training and test set in the ratio of 70:30 (Training:Test) based on dependent and independent variables.
6. Create the decision tree model using “entropy” method of finding the split columns and fit it to training data.

7. Print the accuracy of the model & print the confusion matrix
8. Regularize the decision tree by limiting the max. depth of trees and print the accuracy.
9. Apply the Random forest model and print the accuracy of Random forest Model
10. Apply Adaboost Ensemble Algorithm for the same data and print the accuracy.
11. Apply Bagging Classifier Algorithm and print the accuracy.
12. Apply GradientBoost Classifier Algorithm for the same data and print the accuracy

greatlearning

HappyLearning

Questions?