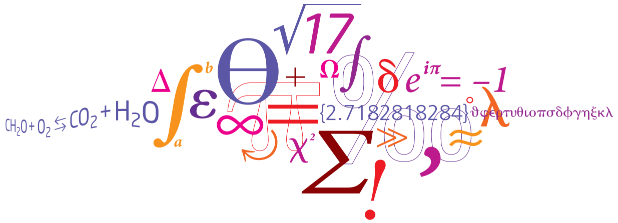
Nakul R Padalkar
GEORGETOWN UNIVERSITY

Tree Ensemble Methods

Bagging and Random Forests



Overview

Classification Problem

In general, we want our classifier to have lowest generalized error.

- **Data $(X, Y) \in \mathbb{R}^p \times \{0, 1\}$.**
- ***X* is predictor, feature; *Y* is class label, response.**
- **(X, Y) have joint probability distribution $\mathcal{D}$.**
- **Goal: Based on *N* training pairs (X_i, Y_i) drawn.**
- **from $\mathcal{D}$ produce a classifier $\hat{C}(X) \in \{0, 1\}$.**
- **Goal: choose $\hat{C}$ to have low generalization error.**

Generative Methods

$$\underbrace{\mathbb{E}_D[(h(x|D) - y)^2]}_{ExpectedError} = \underbrace{\mathbb{E}_D[(\theta - \hat{\theta})^2]}_{Variance} + \underbrace{\hat{\theta}^2}_{Bias}$$

- Boosting - converts weak learners to strong learners. Intuitively, a weak learner is just slightly better than a random guess, while a strong learner is very close to perfect performance.
- **Goal:** Reduce Bias.
- Bagging or bootstrap aggregation averages a given procedure over many samples to reduce its variance — a poor man's Bayes.
- **Goal:** Reduce Variance.

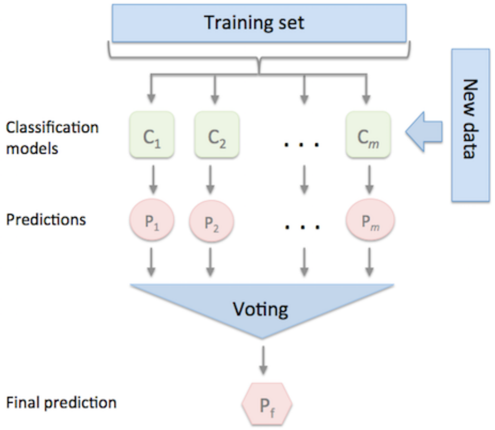
Weak and Strong Learners

Learners

- A weak learner produces a classifier that is only slightly more accurate than random classification.
- A class of concepts is learnable (or strongly learnable) if a polynomial-time algorithm achieves low error with high confidence for all concepts in the class.

Principle of Averaging

Voting Classifier: Average the predictions of a collection of classifiers.



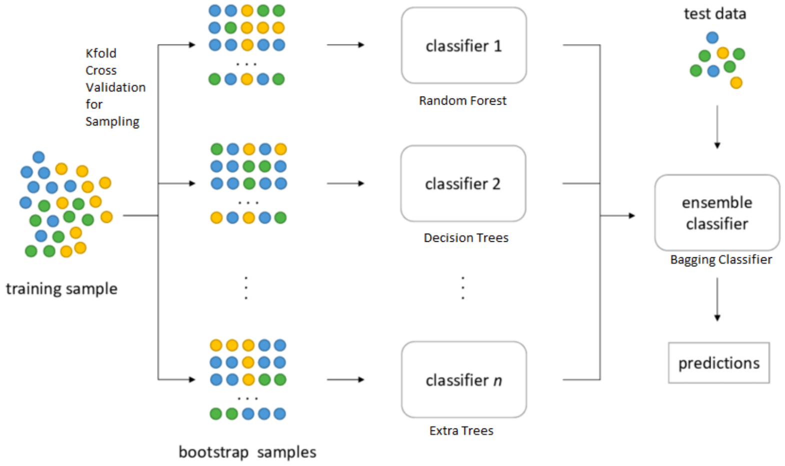
Principle of Averaging

- Instead of training different models on the same data, train the same model multiple times on different data sets and "combine" these "different" models
- We can use some simple/weak models as the base model
- **How do we get multiple training data sets?** (in practice, we only have one data set at training time)

Bootstrap Aggregating

- Bagging or bootstrap aggregation averages a given procedure over many samples, to reduce its variance — a poor man's Bayes.
- Bagging classifier helps in reducing the variance of individual estimators by introducing randomization into the training stage of each of the estimators and making an ensemble out of all the estimators. Note that the high variance means that changing the training data set results in the constructed or trained estimator by a great deal.

Bagging Sequence

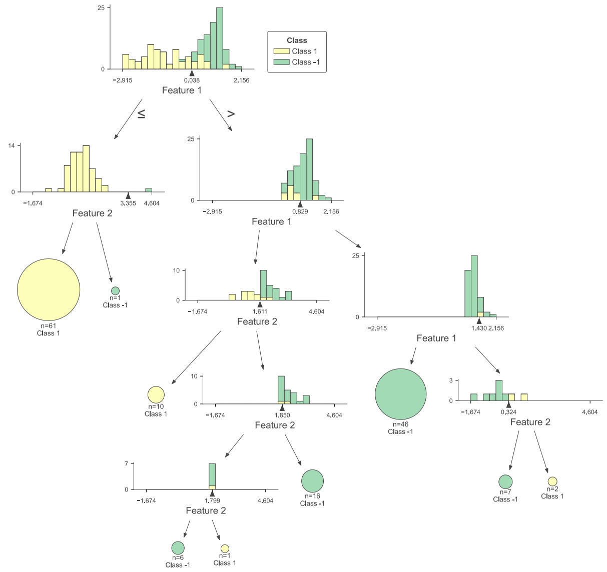


Bagging Algorithm

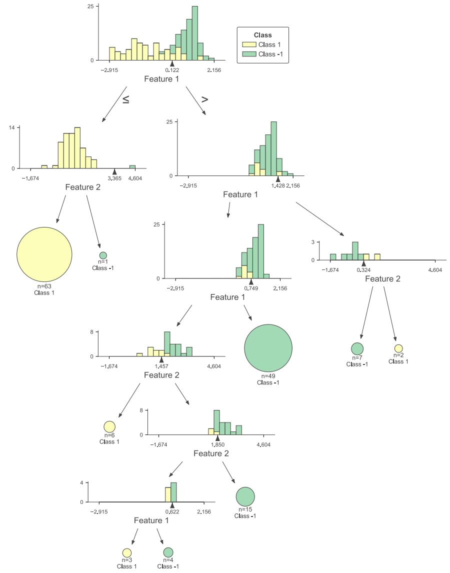
Algorithm 1: Bagged Averaging

```
Input  : Original dataset $\mathcal{D}$, Number of classifiers $T$
Output: Aggregated prediction

Initialize empty set of classifiers $\mathcal{C}$;
Initialize empty set of predictions $\mathcal{P}$;

for $t = 1$ to $T$ do
    Sample a bootstrap dataset $\mathcal{D}_t$ from the original dataset $\mathcal{D}$;
    Train a classifier $C_t$ using $\mathcal{D}_t$;
    Make predictions $\mathcal{P}_t$ using classifier $C_t$;
    $\mathcal{C} \leftarrow \mathcal{C} \cup C_t$;
    $\mathcal{P} \leftarrow \mathcal{P} \cup \mathcal{P}_t$;
end

Aggregate predictions in $\mathcal{P}$ using averaging or voting;
return Aggregated prediction;
```

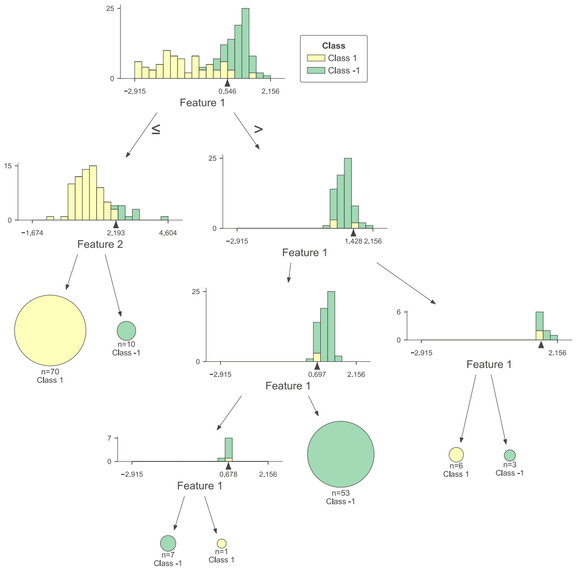
Bagged Classifier Trees



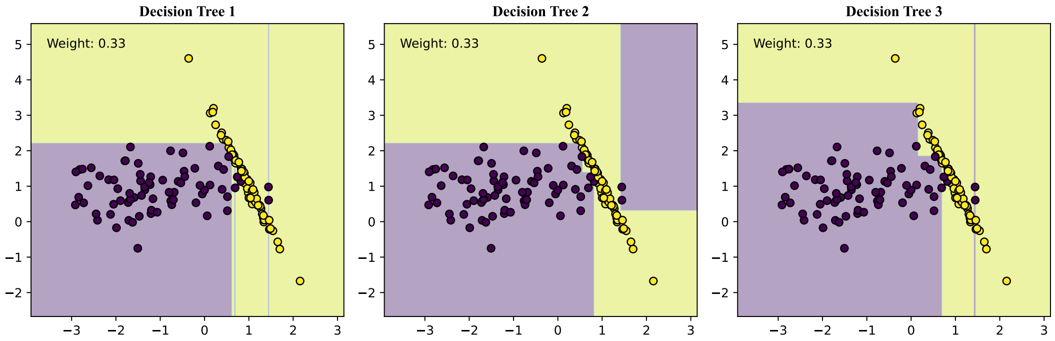
Bagged Classifier Trees



Bagged Classifier Trees



Average Bagged Classifier



Classifier Aggregation I

- The bagged classifier is an ensemble learning method that combines multiple individual classifiers, known as decision trees, to make predictions.
- Each decision tree in the ensemble is trained on a different subset of the original training dataset, created through bootstrap sampling (sampling with replacement).
- During the training process, the bagged classifier fits each decision tree to a random subset of the training data, allowing each tree to learn different patterns and reduce overfitting.
- When making predictions, the bagged classifier combines the predictions from all decision trees by averaging the predicted probabilities (for classification problems) or taking the majority vote (for binary classification).

Classifier Aggregation II

- The weights of the individual decision trees in the bagged classifier are calculated as $1/N$, where N is the ensemble's total number of decision trees. This ensures that each decision tree contributes equally to the final prediction.
- By aggregating the predictions of multiple decision trees, the bagged classifier aims to improve the overall accuracy, robustness, and generalization ability compared to using a single decision tree.
- The bagged classifier can be used for both classification and regression tasks, depending on the type of base classifier (e.g., decision trees, random forests) used in the ensemble.

Random Forest Algorithm

Algorithm 2: Random Forest Algorithm

```
Input   : $X$: Input features (training data)
Input   : $y$: Output labels (training data)
Input   : $n\_estimators$: Number of decision trees to include in the forest
Input   : $max\_features$: Maximum number of features to consider for each split
Output: $RandomForest$: Trained random forest model
$RandomForest \leftarrow \emptyset$;
for $i \leftarrow 1$ to $n\_estimators$ do
    # Bootstrap the training data:;
    $X\_bootstrap, y\_bootstrap \leftarrow$ RandomlySelectSamplesWithReplacement($X, y$);
    # Construct a decision tree:;
    $tree \leftarrow$ DecisionTreeAlgorithm($X\_bootstrap, y\_bootstrap, max\_features$);
    # Append the tree to the RandomForest:;
    $RandomForest \leftarrow RandomForest \cup \{tree\}$;
end
return $RandomForest$;
```

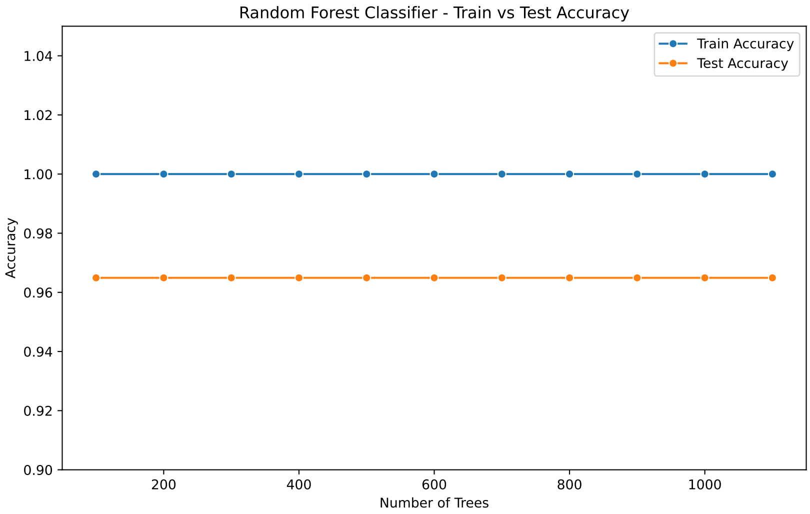
Scikit-Learn Python vs. R I

- `n_estimators`: The number of trees in the random forest.
- `max_depth`: The maximum depth of each decision tree in the ensemble.
- `min_samples_split`: The minimum number of samples required to split an internal node.
- `min_samples_leaf`: The minimum number of samples required to be at a leaf node.
- `max_features`: The number of features to consider when looking for the best split.
- `bootstrap`: Whether bootstrap samples should be used when building trees.
- `random_state`: The seed used by the random number generator.

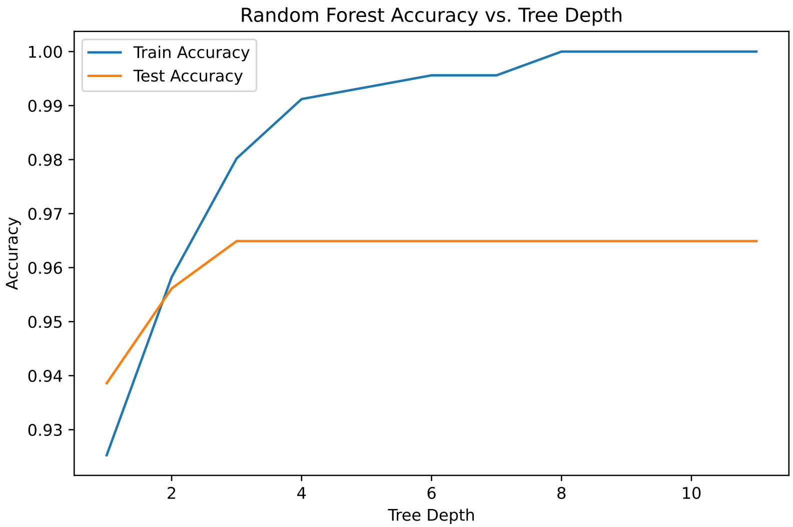
Scikit-Learn Python vs. R II

- `ntree`: The number of trees in the random forest.
- `mtry`: The number of variables randomly sampled as candidates at each split.
- `maxdepth`: The maximum depth of each tree in the ensemble.
- `nodesize`: The minimum number of samples required for a node to be split further.
- `sampsize`: The number of samples randomly drawn for each tree.
- `importance`: Whether to compute variable importance measures.

Effect of Number of Trees



Effect of Tree Depth


Random Forest Accuracy vs. Tree Depth
Train Accuracy
Test Accuracy
Accuracy
Tree Depth