Class Objectives

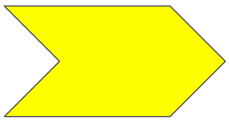
In today's class we'll learn about a new ML algorithms family: Tree based algorithms

- Decision trees
- Random forest
- Weak learners
- Ensemble methods

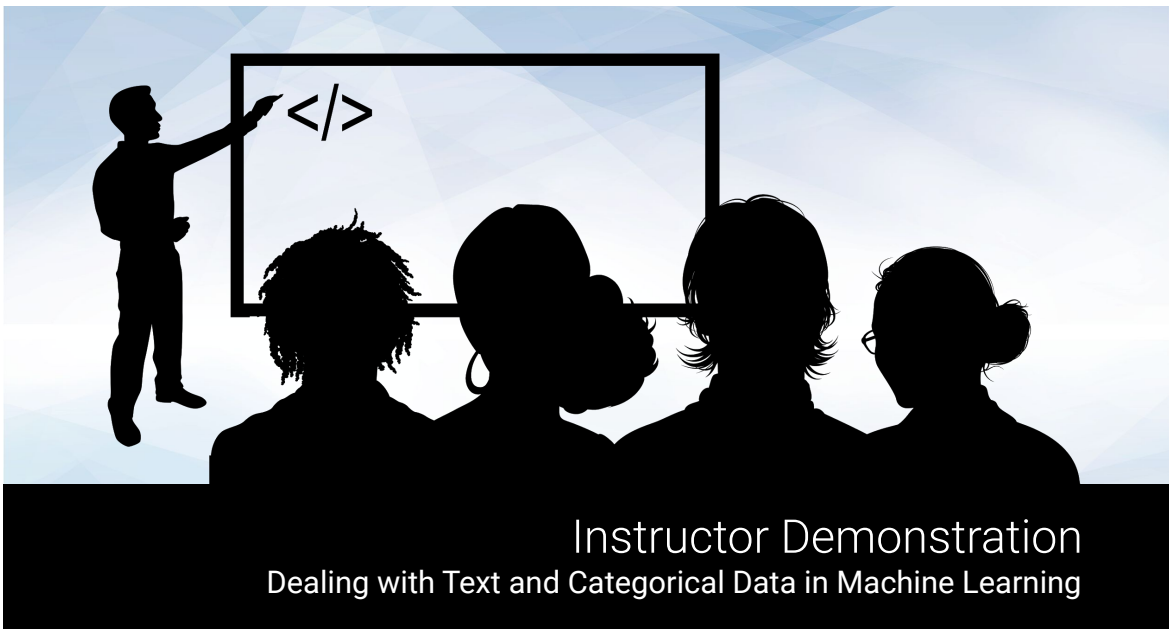
Categorical Data

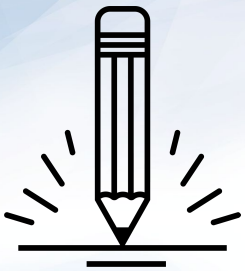
Also we'll learn how to deal with categorical data

Color
Red
Red
Yellow
Green
Yellow



Red	Yellow	Green
1	0	0
1	0	0
0	1	0
0	0	1





Activity: Encoding Categorical Data for Machine Learning

In this activity, you will be tasked with encoding categorical and text features of a dataset that contains 2,097 loan applications.

Suggested Time:
10 minutes

5

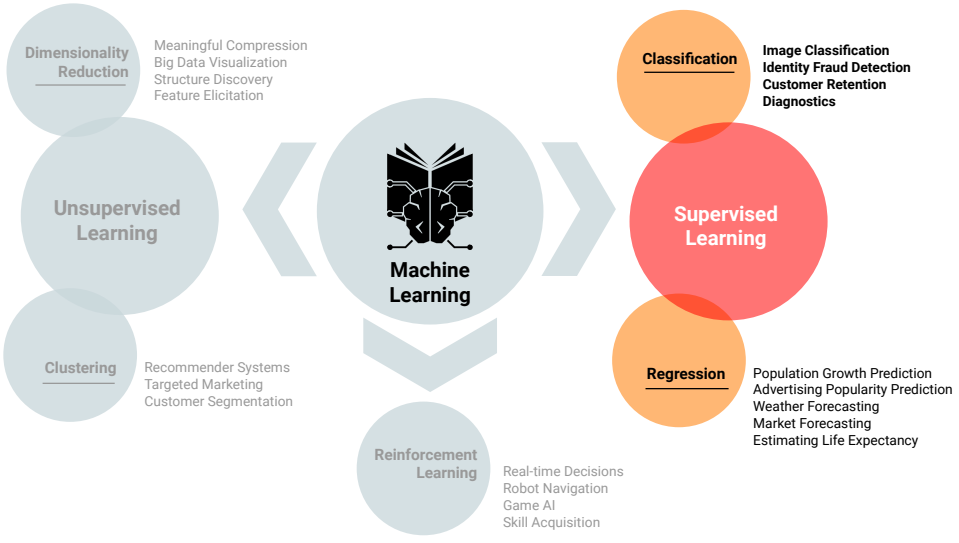
Time's Up! Let's Review.

6

Walking into the Algorithms Forest

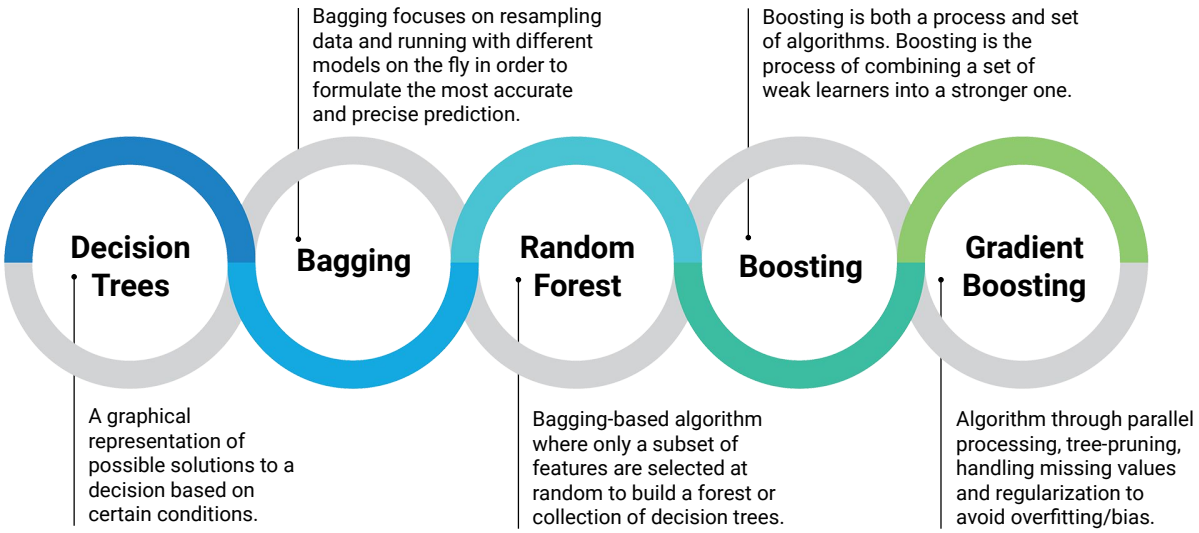
Tree based algorithms

Tree based algorithms, are part of the supervised machine learning methods.



Tree-based algorithms are supervised learning methods that are mostly used for classifications and regression problems.

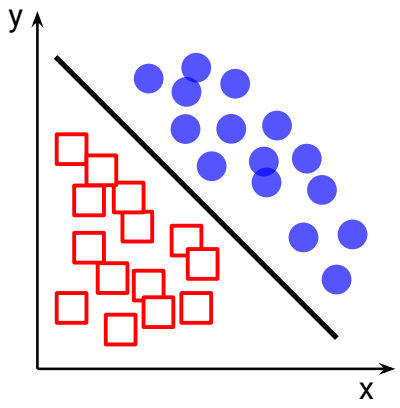
Tree Based Algorithms at a Glance



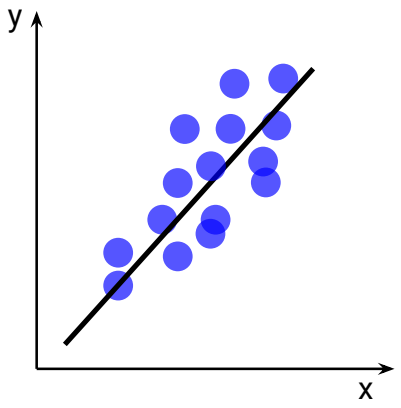
Algorithms

These algorithms can be used to solve classification or regression problems.

Classification

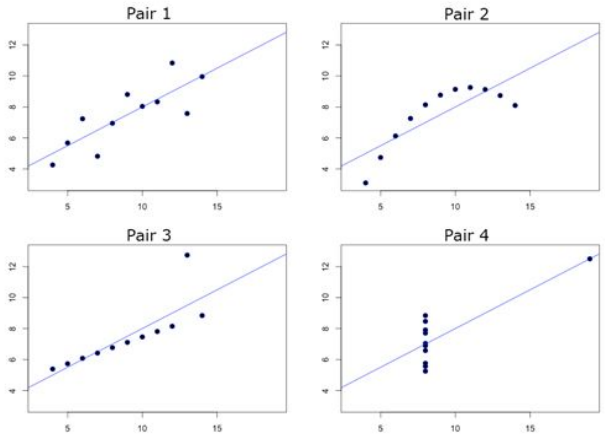


Regression



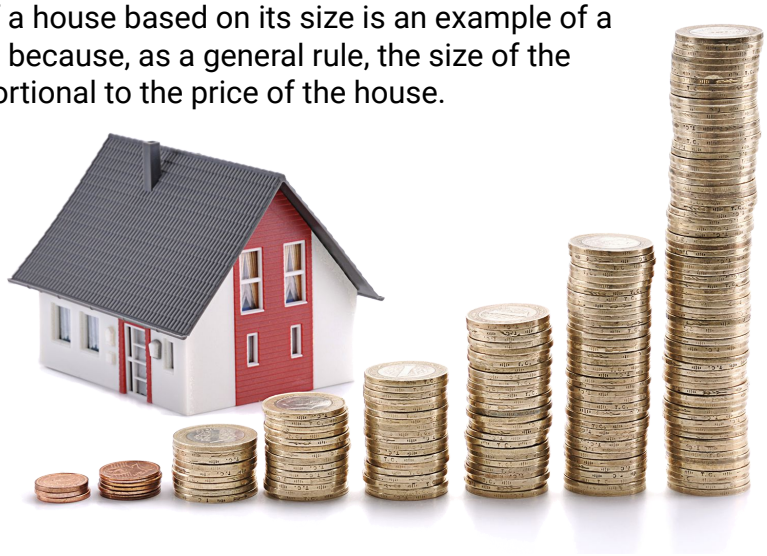
Linear vs. Non-Linear Models

In linear models, the relationship among input variables can be represented as a straight line, while non-linear models have a different shape.



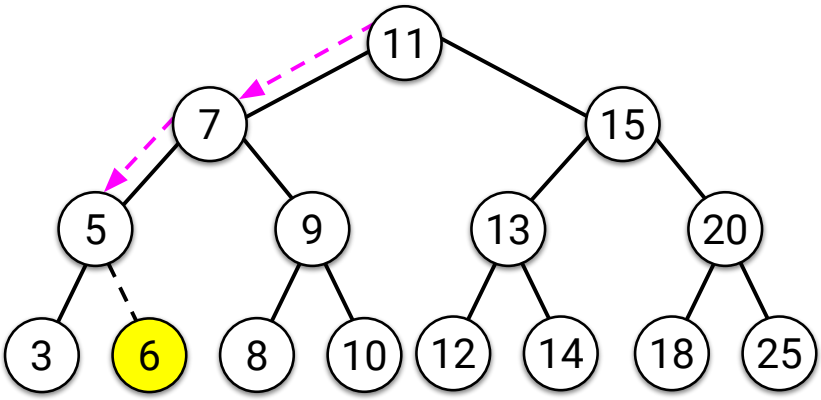
Linear Models

Predicting the price of a house based on its size is an example of a linear problem. This is because, as a general rule, the size of the house is directly proportional to the price of the house.



Non-Linear Models

Tree-based algorithms can map non-linear relationships in data.



Non-Linear Models

Predicting if a credit application is going to be fraudulent or not may be an example of a non-linear problem due to the complex relationship between the input features and the output prediction.

Tree-Based Algorithms

These algorithms are quite often used in finance for assessing risk, preventing fraud, or fighting money laundering.

sklearn

`sklearn` has two modules that implement tree-based algorithms that we will be covering Today.

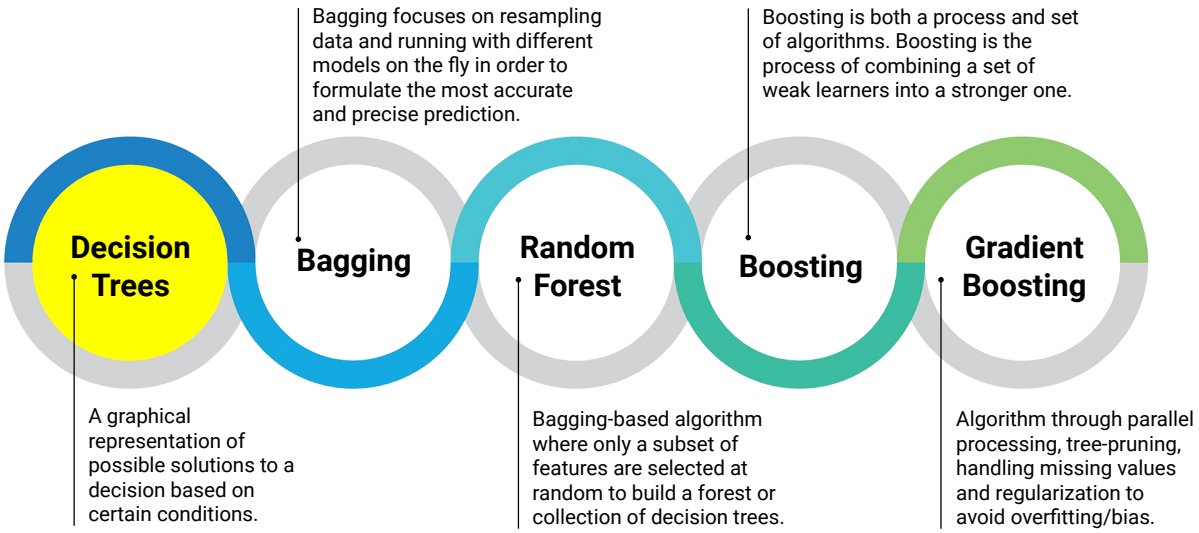
01 `sklearn.tree`

`sklearn.tree` implements decision trees.

02 `sklearn.ensemble`

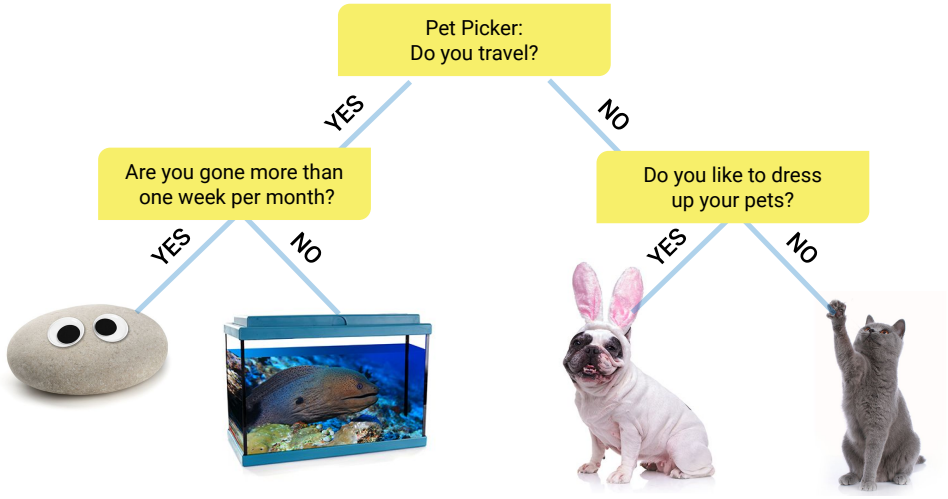
`sklearn.ensemble` offers implementations for random forest, gradient boosting, boosting and bagging algorithms.

Decision Trees



Decision Trees

Decision trees encode a series of true/false questions.



Decision Trees

These true/false questions can be represented with a series of if/else statements

Do you travel?

Yes Travel:

Are you gone for more than one week per month?

Yes: Pet Rock

No: Pet Fish

No Travel:

Do you like to dress up your pet?

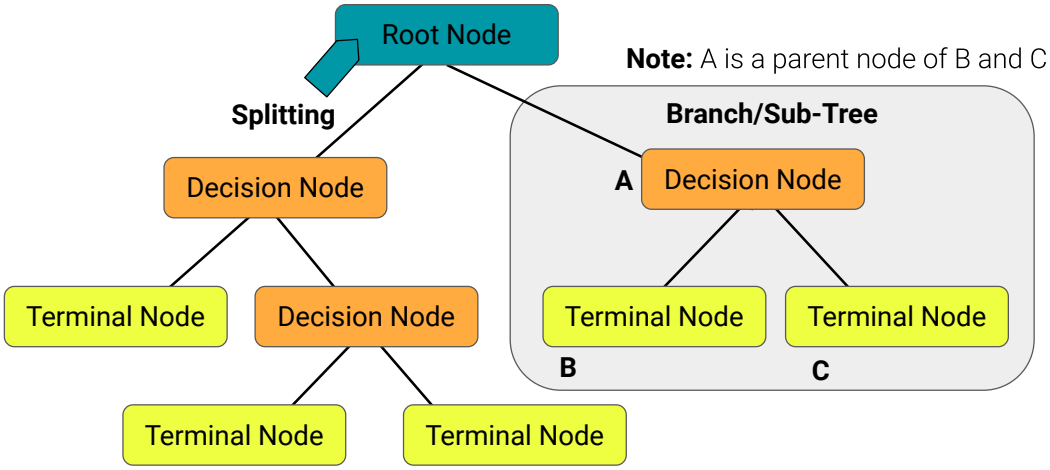
Yes Dress Up: Pet Dog

No Dress Up: Pet Cat

```
if (travel):
  if (time > week):
    print("Rock")
  else:
    print("Fish")
else:
  if (dress_up):
    print("Dog")
  else:
    print("Cat")
```

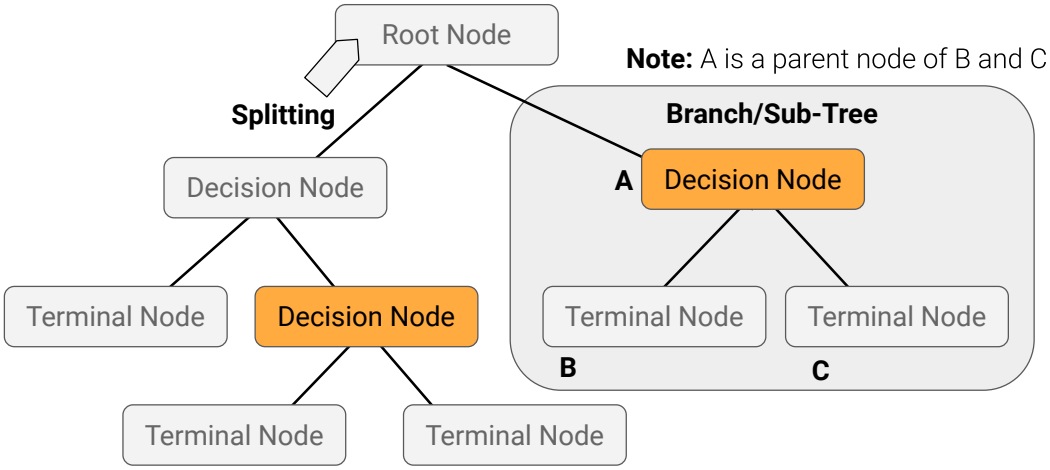
Root Node

Represents the entire population or sample data, this node gets divided into two or more homogeneous sets.



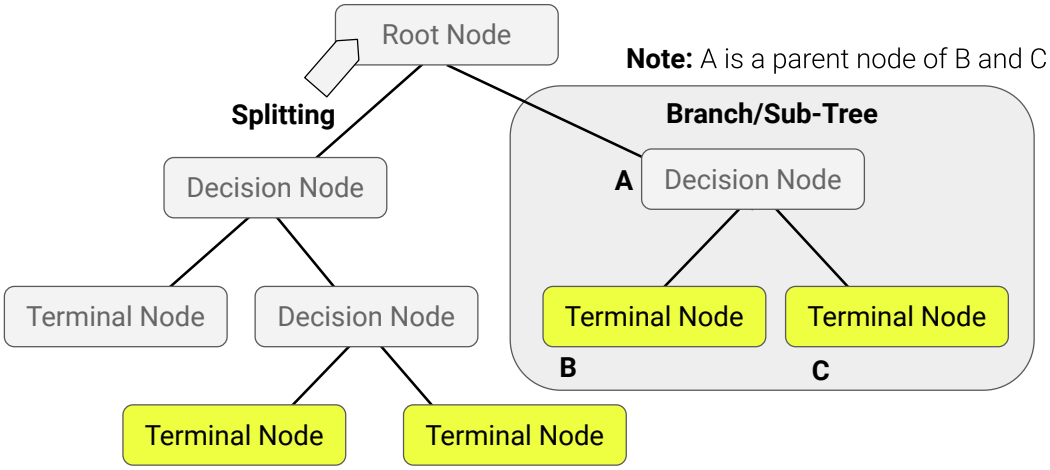
Parent Node

A node that is divided into sub-nodes.



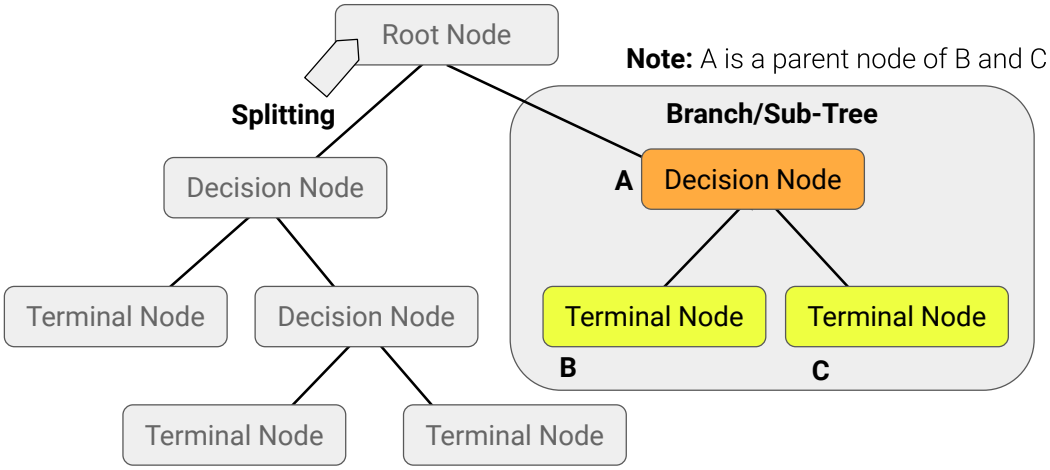
Child Node

Sub-nodes of a parent node.



Decision Node

A sub-node that is split into further sub-nodes.



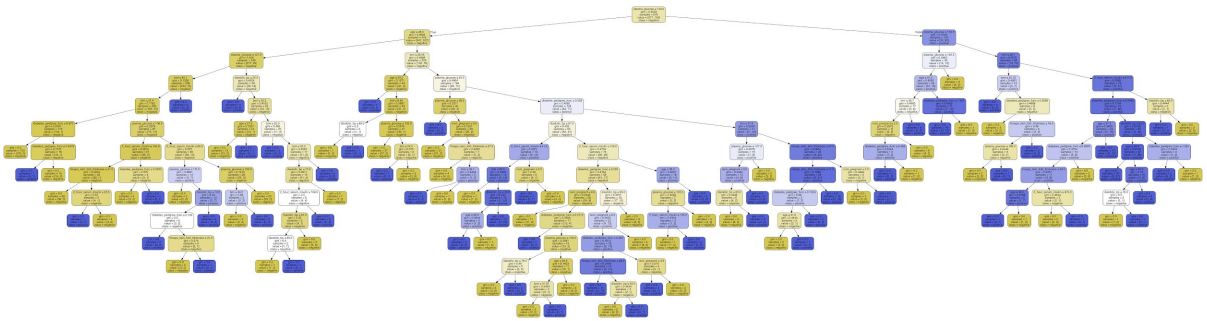
Decision Trees

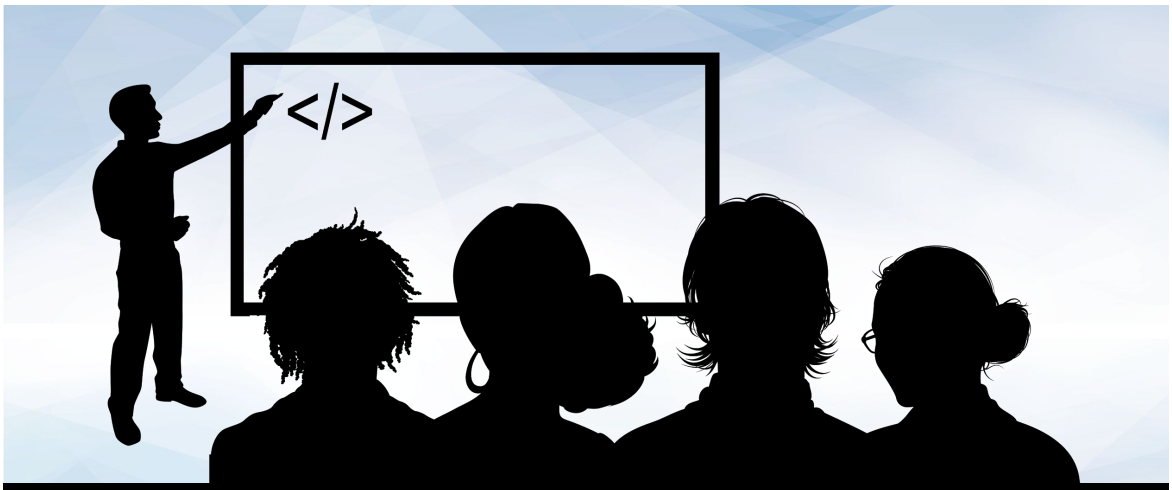
Key concepts to know while working with decision trees:

Leaf or Terminal Node	Nodes that do not split.
Branch or Sub-Tree	A subsection of entire tree.
Splitting	Process of dividing a node into two or more sub-nodes.
Pruning	Process of removing sub-nodes of a decision node.
Tree's Depth	The number of decision nodes encountered before making a decision.

Decision Trees

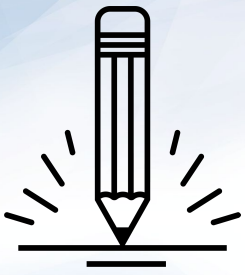
Decision trees can become very complex and may not generalize well.





Instructor Demonstration

Decision Trees



Activity: Predicting Fraudulent Loans Applications

In this activity, you will create a decision tree model to predict fraudulent loan applications.

Suggested Time:
10 minutes

Time's Up! Let's Review.

Introduction to Ensemble Learning

The Classification Algorithm Race

If we compare the performance of classification algorithms, we'll find that some algorithms performed better than others

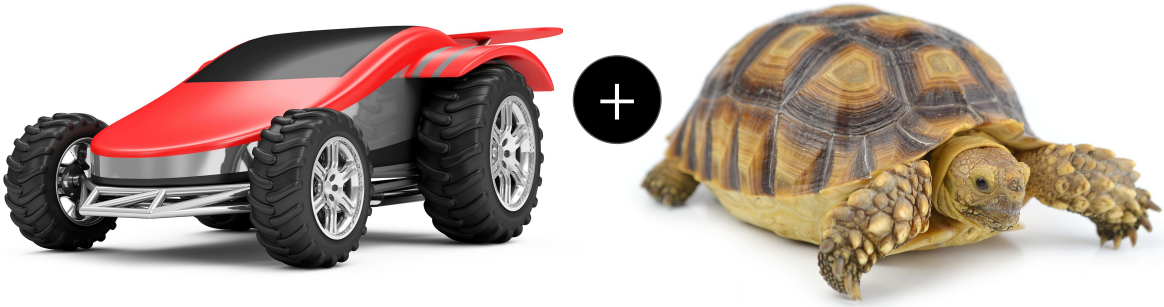
Weak Learners

- Algorithms that actually fail at learning in an adequate fashion.
- They are a consequence of limited data to learn from.
- Their predictions are only a little better than random chance.

Weak Learners are still valuable in Machine Learning

They can be combined with other classifiers in order to make a more accurate and robust prediction engine.

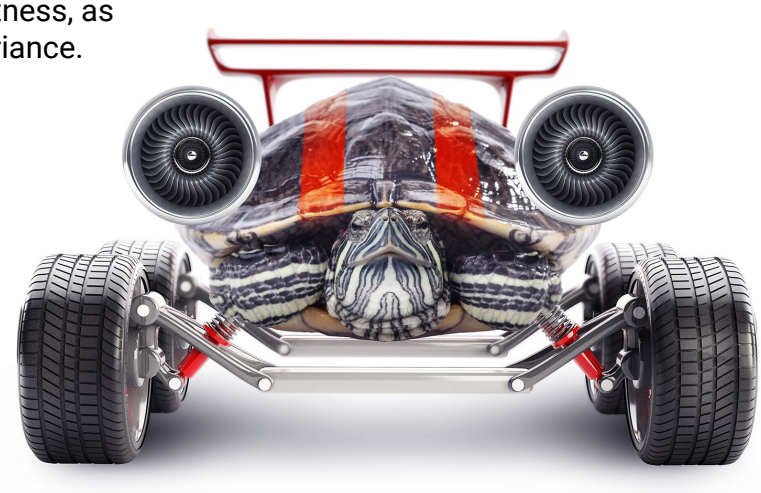
Combined weak learners are an example of ensemble learning:



Ensemble Learners

Ensemble learners improves accuracy and robustness, as well as decrease variance.

Combined, weak learners can perform as well as strong learners.

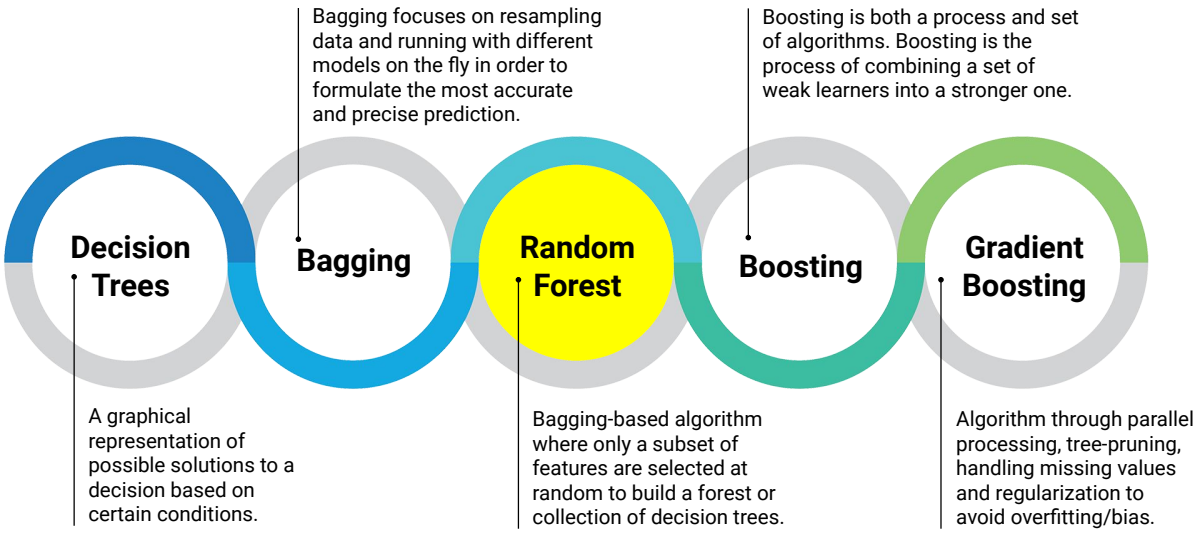


Combining Weak Learners

Weak learners have to be combined using specific algorithms like:

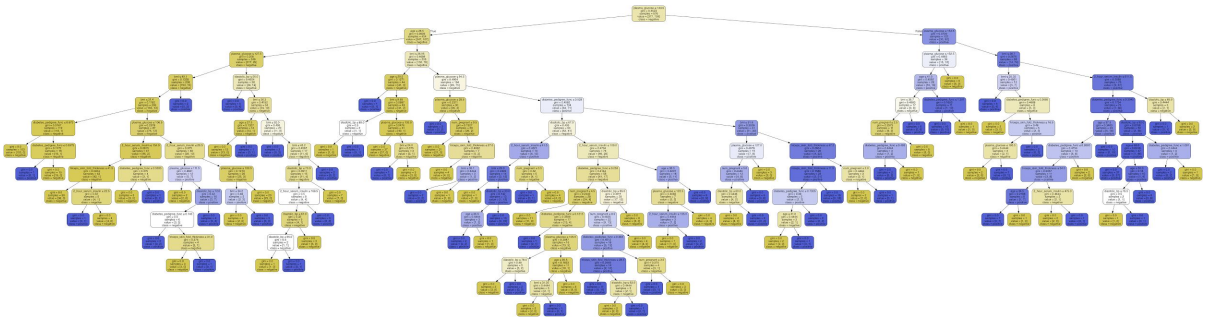
- GradientBoostingTree
- XGBoost
- Random Forest

Random Forest



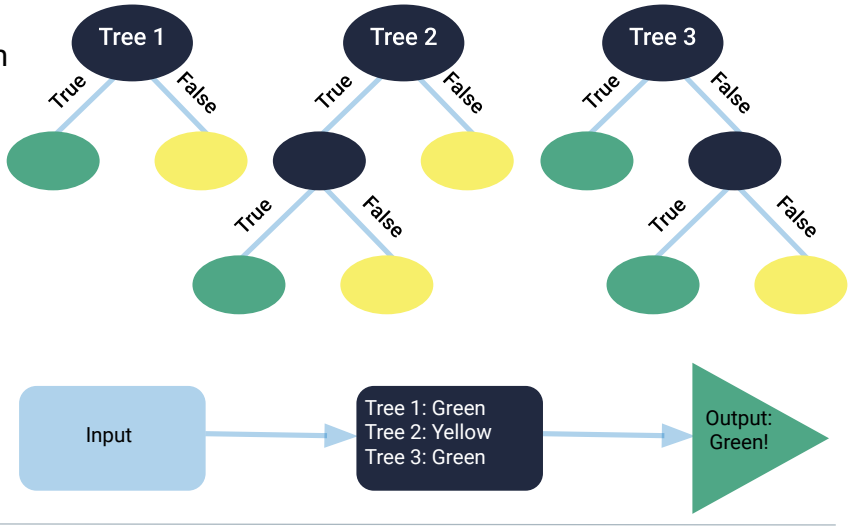
Random Forest

Instead of having single, complex tree like the ones created by decision trees, a random forest algorithm will sample the data and build several smaller, simpler decisions trees.



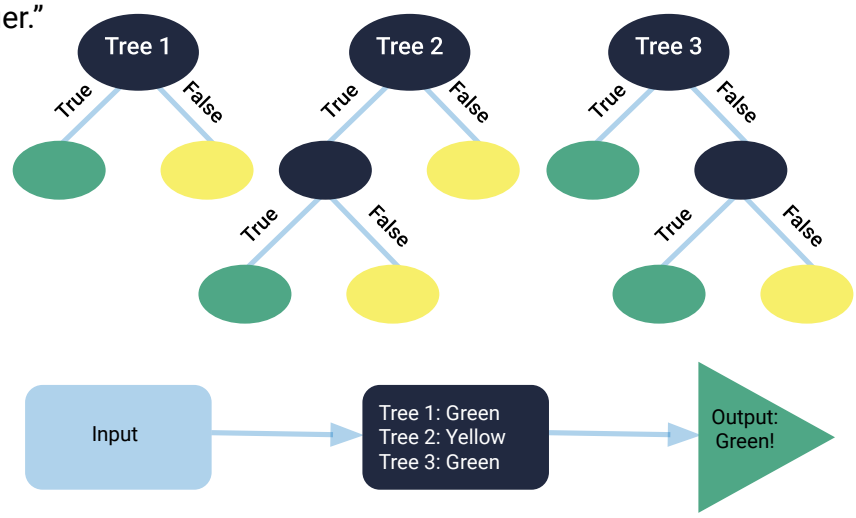
A Forest of Trees

In a random forest, each tree is much simpler because it is built from a subset of the data.



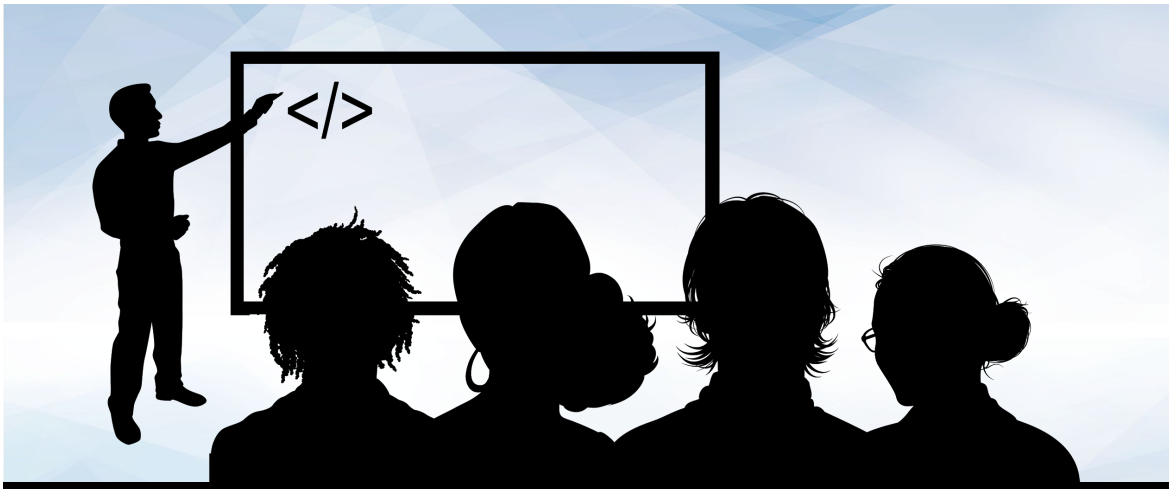
A Forest of Trees

Each tree is considered a “weak classifier” but when you combine them, they form a “strong classifier.”



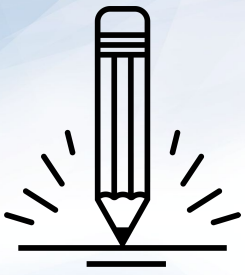
Benefits of Random Forest Algorithm

- It’s robust against overfitting.
- It can be used to rank the importance of input variables in a natural way.
- It can handle thousands of input variables without variable deletion.
- It’s robust to outliers and non-linear data.
- It runs efficiently on large databases.



Instructor Demonstration

Random Forest



Activity: Predicting Fraud with Random Forests

In this activity, you will explore how the random forest algorithm can be used to identify fraudulent loan applications. You will use the `sba_loans_encoded.csv` file that they created before to train the model.

Suggested Time:
10 minutes

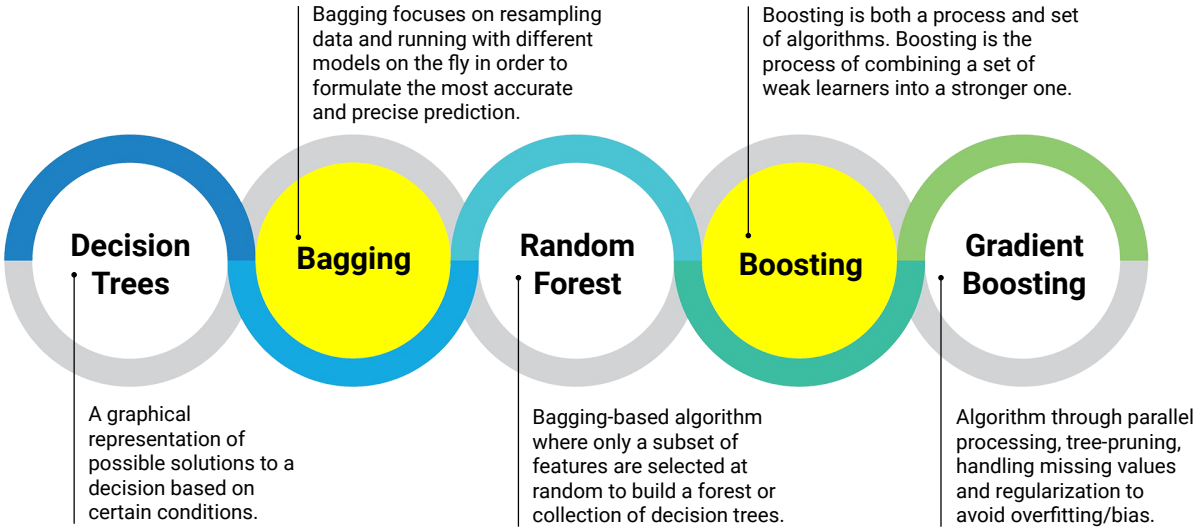
Time's Up! Let's Review.

Review: Predicting Fraud with Random Forests

- Would you trust in this model to deploy a fraud detection solution in a bank?
- What are your insights about the top 10 most importance features?

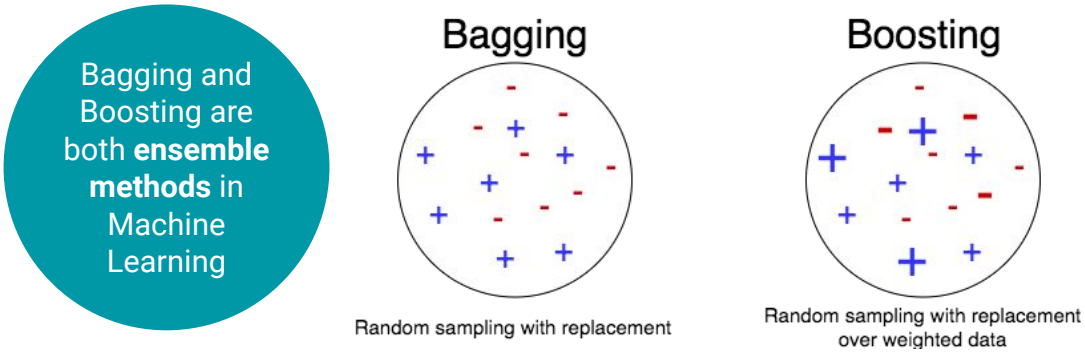
Boosting and Bagging

Boosting and Bagging



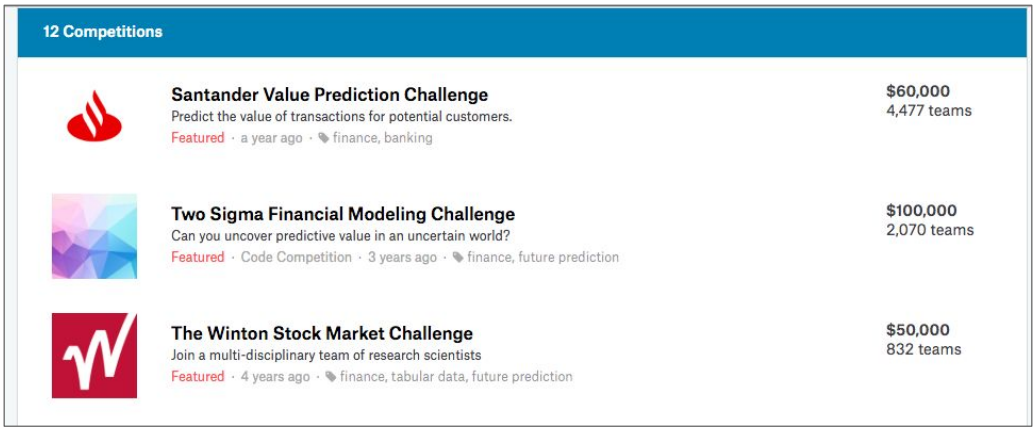
Boosting and Bagging

Boosting and bagging algorithms are used to improve the robustness and reliability of machine-learning models



Boosting and Bagging

Boosting and bagging algorithms like XGBoost are often the best performing in Kaggle machine-learning contests. Their ability to make accurate predictions with precision and substantial recall is almost unparalleled



Boosting vs. Bagging

Boosting

Boosting takes multiple algorithms and coordinates them as an ensemble and runs the algorithms in tandem to identify the best prediction

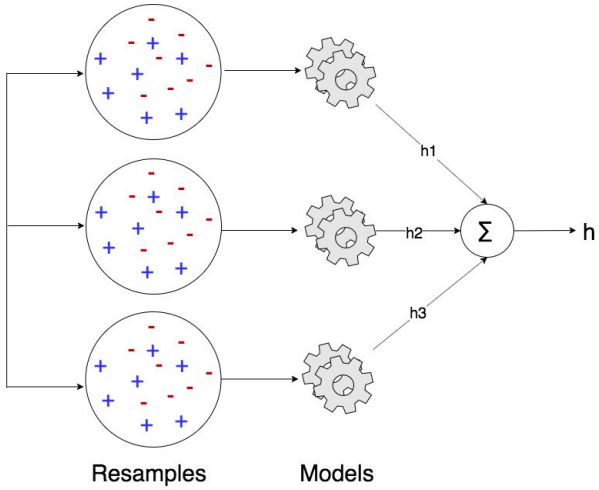
VS.

Bagging

Bagging focuses on resampling data and running with different models on the fly to formulate the most accurate and precise prediction.

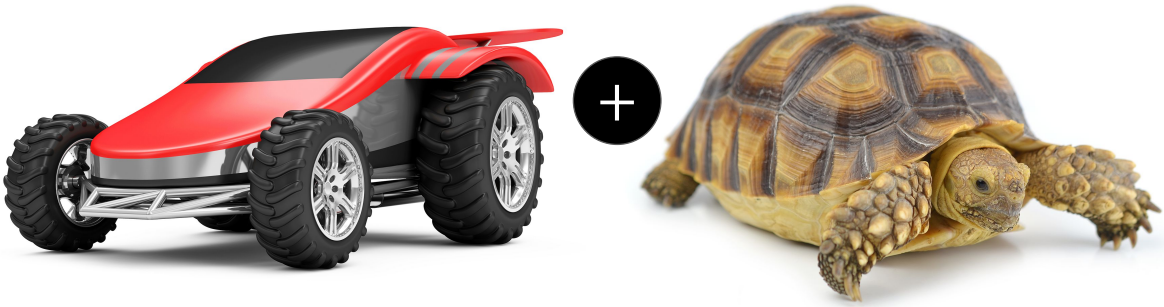
Bagging

Bagging averages predictions from multiple samples and or models.



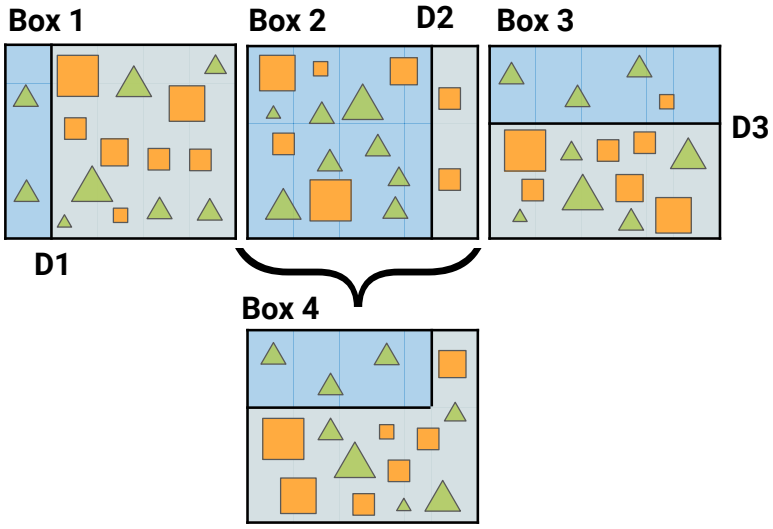
Boosting

Boosting algorithms work iteratively, by sampling more heavily the observations with worst predictions. Subsequent weak learners then learn these more. Weak learners are then aggregated to produce a more accurate and precise prediction. The goal goal of a boosting algorithm is to combine weak learners into ensemble learners.



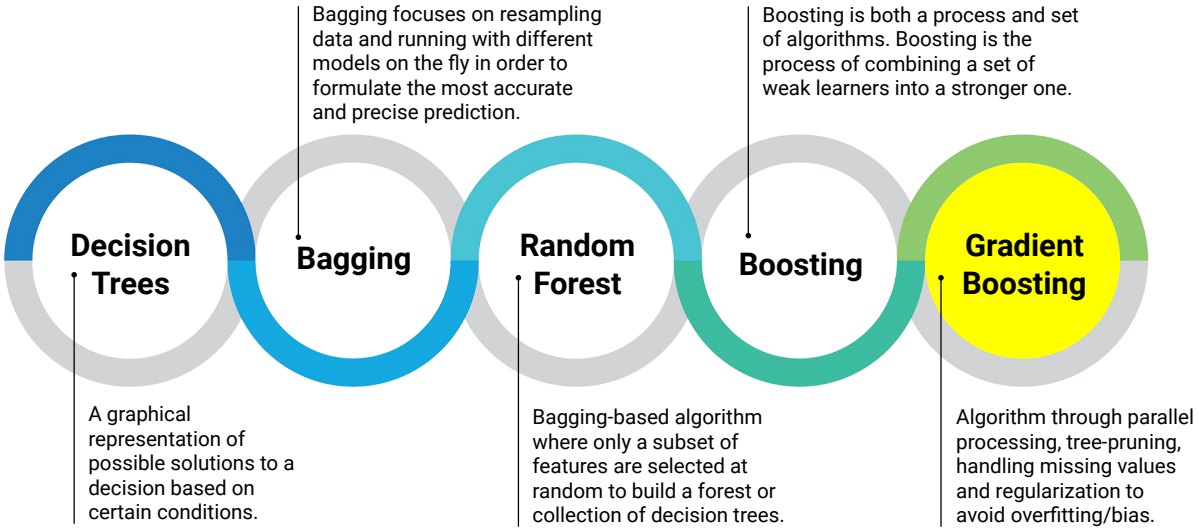
Boosting

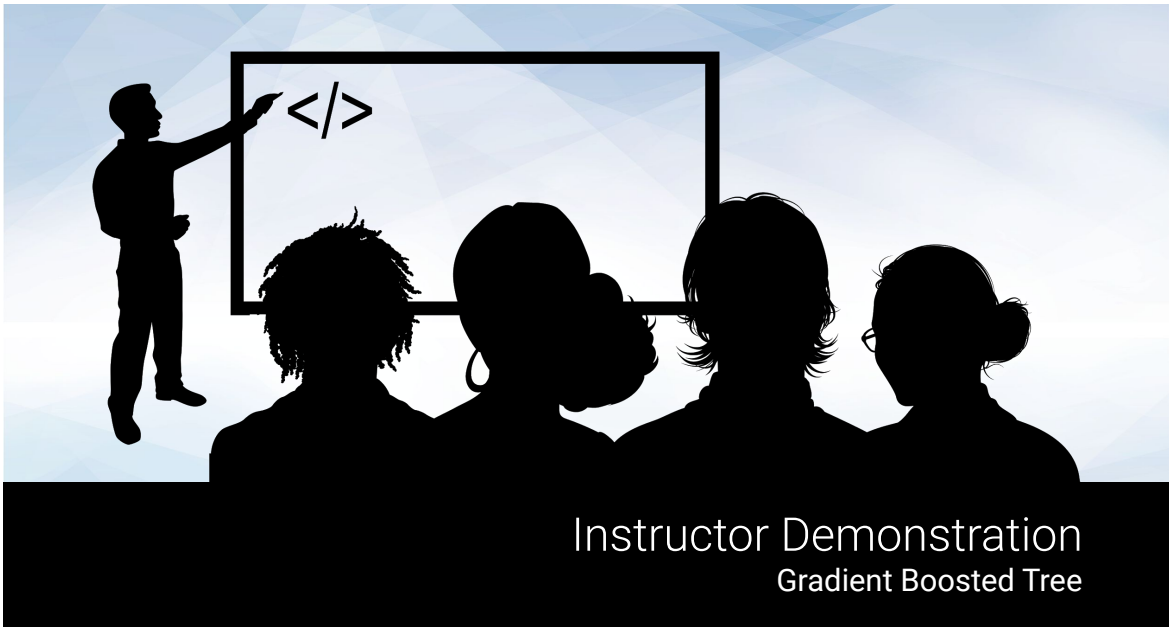
For this reason, boosting algorithms are considered meta-algorithms. Instead of working with and affecting data, boosting algorithms work with and affect other algorithms.



Gradient Boosted Tree

Gradient Boosting





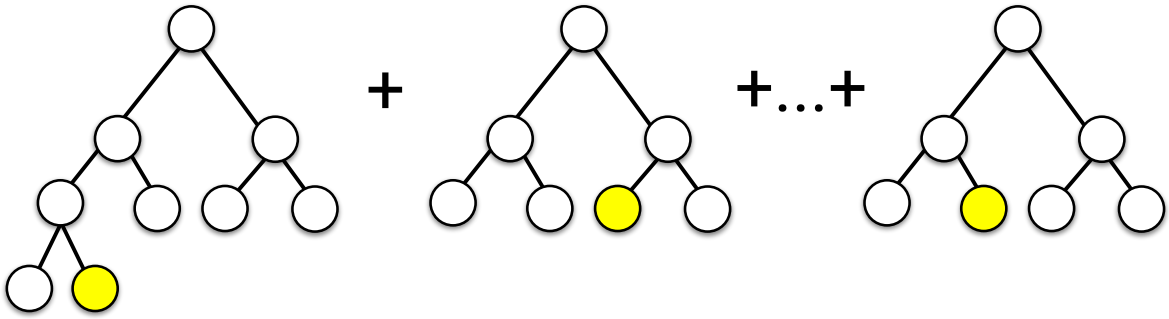
Instructor Demonstration

Gradient Boosted Tree

Gradient Boosted Tree

Gradient Boosted Trees can be created using the GradientBoostingClassifier module from the ensemble package

From `sklearn.ensemble` import `GradientBoostingClassifier`



Gradient Boosted Tree

Arguments

GradientBoostingClassifier has three main arguments:

`N_estimators`

`Learning_rate`

`Max_depth`

Definitions

The `n_estimators` parameter configures the number of weak learners being used with the boosting algorithm.

Gradient Boosted Tree

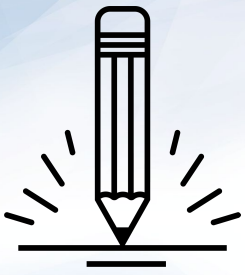
`Learning_rate`

`Learning_rate` controls overfitting.

Smaller values should be used when setting learning_rate.

`max_depth`

The `max_depth` argument identifies the size/depth of each decision tree being used.



Activity: Turbo Boost

In this activity you will use the sklearn `GradientBoostingClassifier` boosting algorithm to detect fraudulent loan applications using ensemble learning.

Suggested Time:
10 minutes

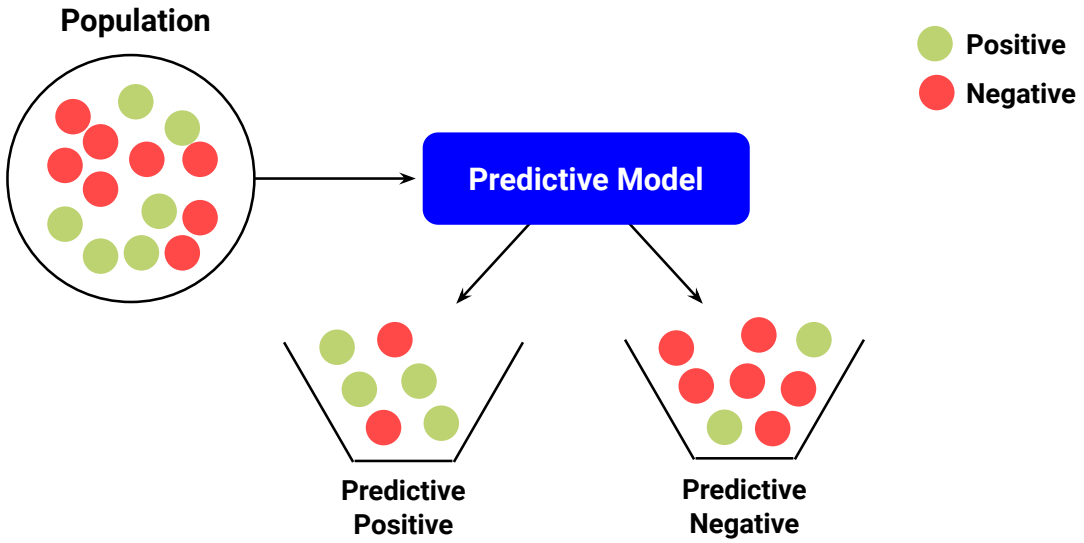
Time's Up! Let's Review.

The Trees vs. The World

Classification: A multidisciplinary challenge

Finance and Banking	Fraud detection, money laundering, credit risk assessment.
Retail and Marketing	Customized product offers, product recommendation, direct-marketing optimization.
Politics	Vote intention, party affinity.
Health	Trials tests, ills diagnosis.
Security	Intruders detection, predictive maintenance.
Education	Programs affinity, customized curricula, desertion prevention.

Classification: A multidisciplinary challenge



Are tree-based algorithms the strongest for classification?

Tree-based algorithms

- Are easy to represent, making a complex model much easier to interpret.
- Can be used for any type of data: Numerical (e.g., loan's amount) or categorical (e.g., name of bank that issues a loan).
- Require little data preparation.
- Can handle data that are not normally distributed.
- Can avoid overfitting.

Trees vs. Classical Classifiers

- Generally speaking, classical classifiers may be faster.
- Logistic regression may outperform decision trees or random forests having a large number of features with low noise.
- SVM also support linear and non-linear models.
- SVM handles outliers better.
- KNN naturally supports incremental learning (data streams).

Which algorithm should I use for classification?

Questions?