AI Bootcamp

Nonlinear Classification

Module 13 Day 2

Class Objectives

By the end of class, you will be able to:

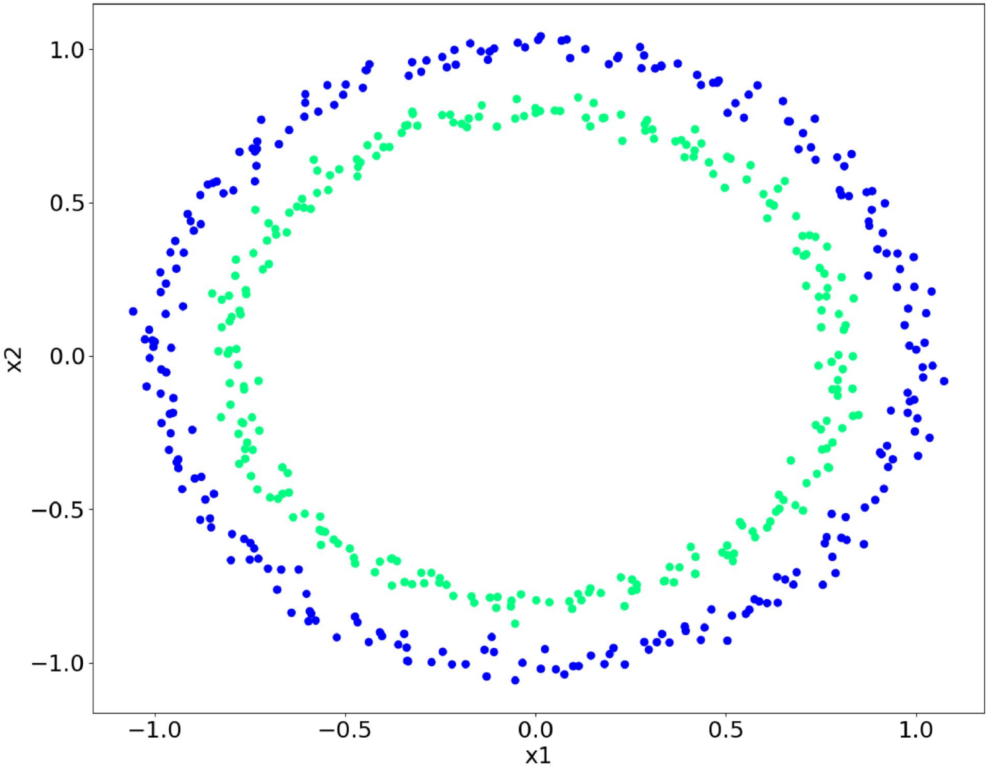
1. Explain how the KNN algorithm works as a classifier and how it differs from other classifiers.
2. Explain how decision trees and random forest work as classifiers and how they differ from each other.
3. Apply fundamental classification algorithms, namely random forest, decision trees, and KNN in machine learning models.

Instructor **Demonstration**

Nonlinear SVM

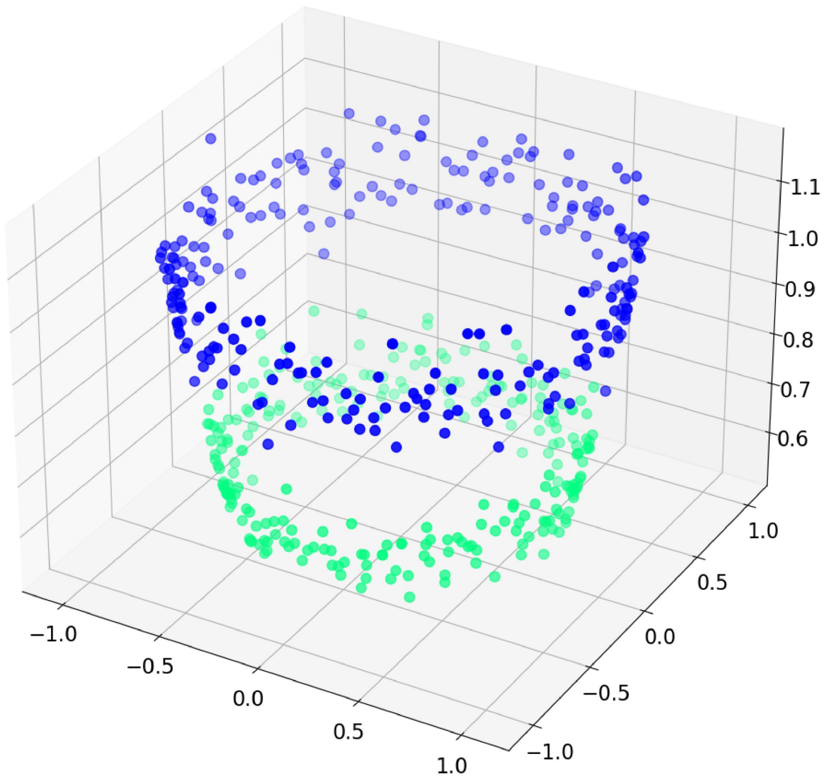
Nonlinear SVMs

Two classes of data arranged in concentric circles and visualized in two dimensions:



Nonlinear SVMs

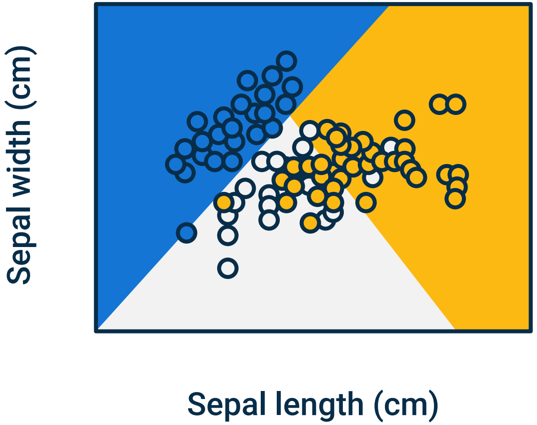
Two classes of data arranged in circles and visualized in three dimensions:



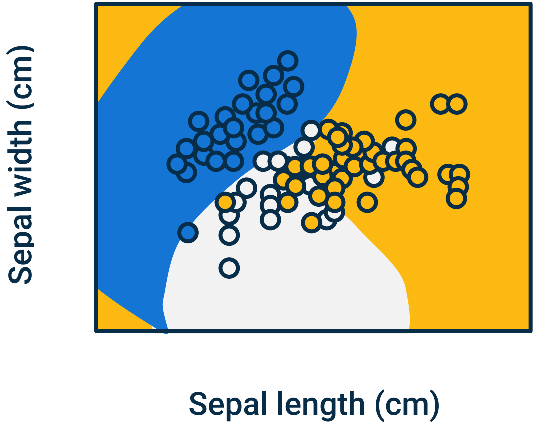
SVC kernel boundaries

Examples of how the SVC model draws boundaries with the same data, depending on kernel choice:

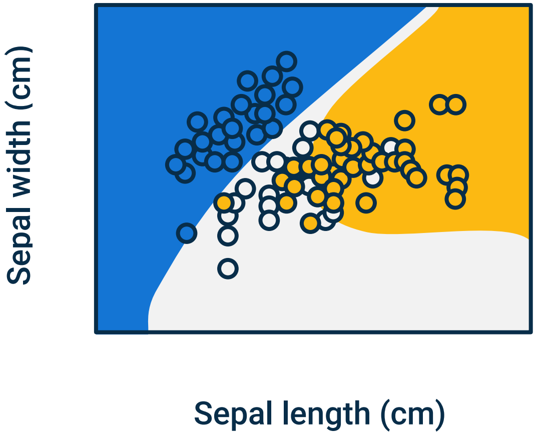
SVC with linear kernel



SVC with RBF kernel



SVC with polynomial (degree 3) kernel



Instructor **Demonstration**

k-Nearest Neighbors

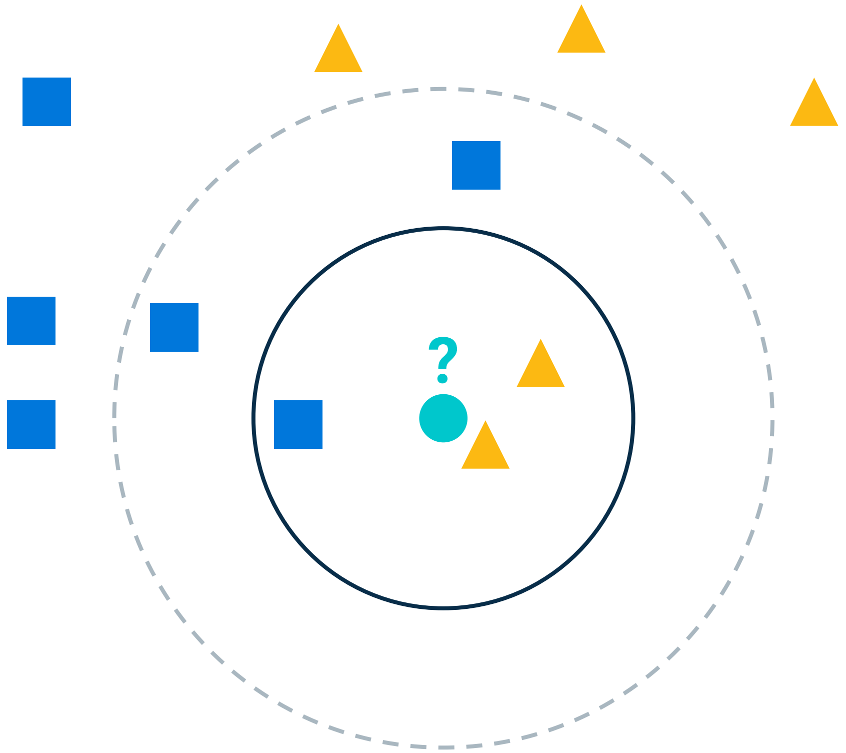
One of the most popular and flexible supervised machine learning models is the **k-nearest neighbor (KNN) model**.

The **KNN** compares known data points to determine the classification of a novel data point.

KNN: Logistic Regression Friendly Neighbor

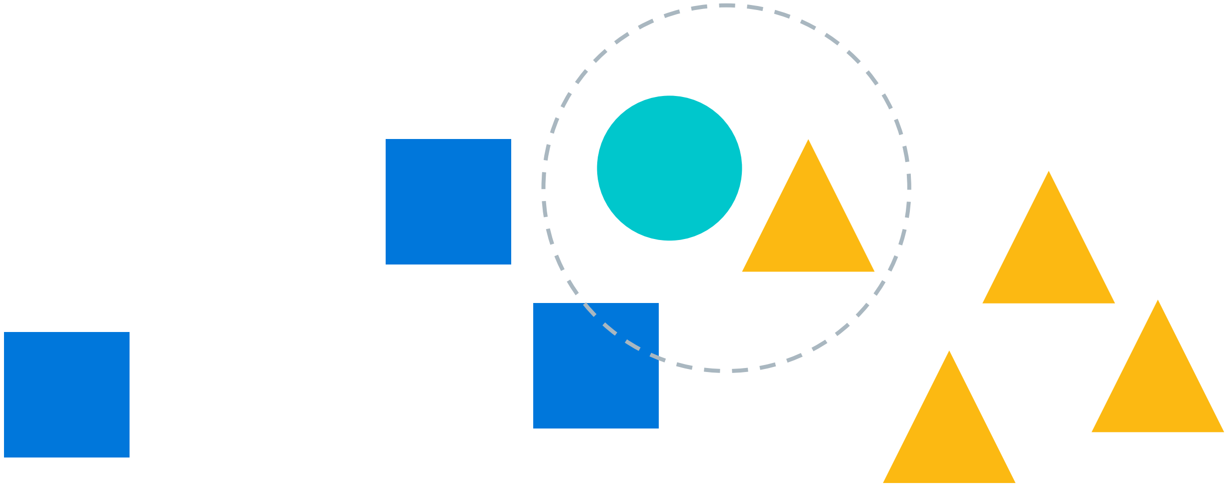
We set the k-value parameter to tell the algorithm to find the k-number of closest known data points.

Then, the algorithm determines what the majority of surrounding data points are classified as to determine the class of the new data point.



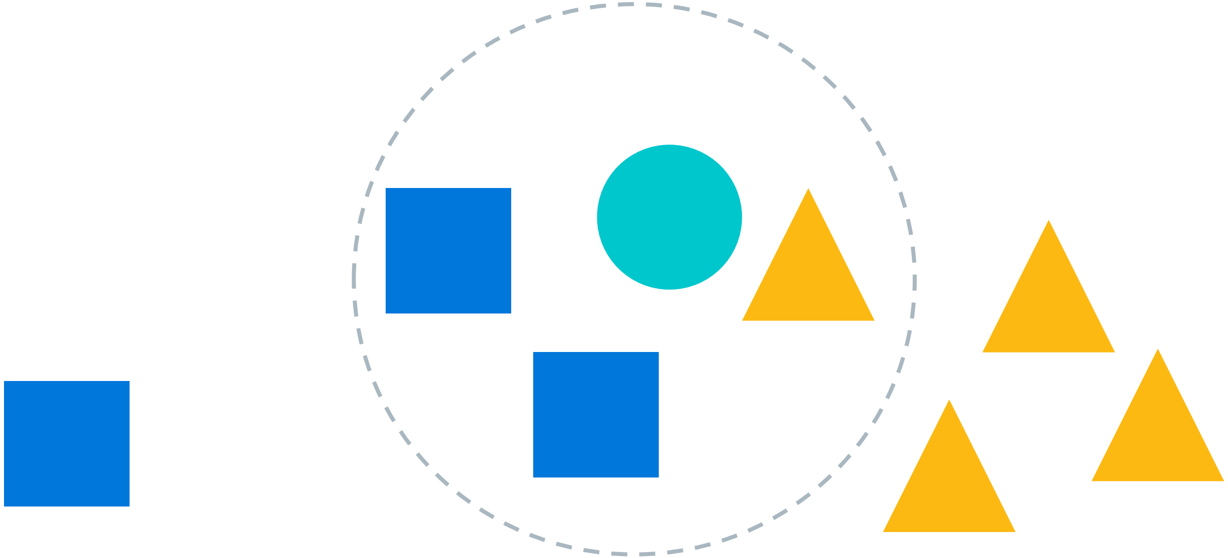
KNN: Logistic Regression Friendly Neighbor

Consider an example in which we wanted to classify a new data point (aqua circle) from known square and triangle data points. If **k = 1**, then the algorithm classifies our new data point to whatever is the closest known single neighbor. In this case, our aqua circle would be classified as a yellow triangle.



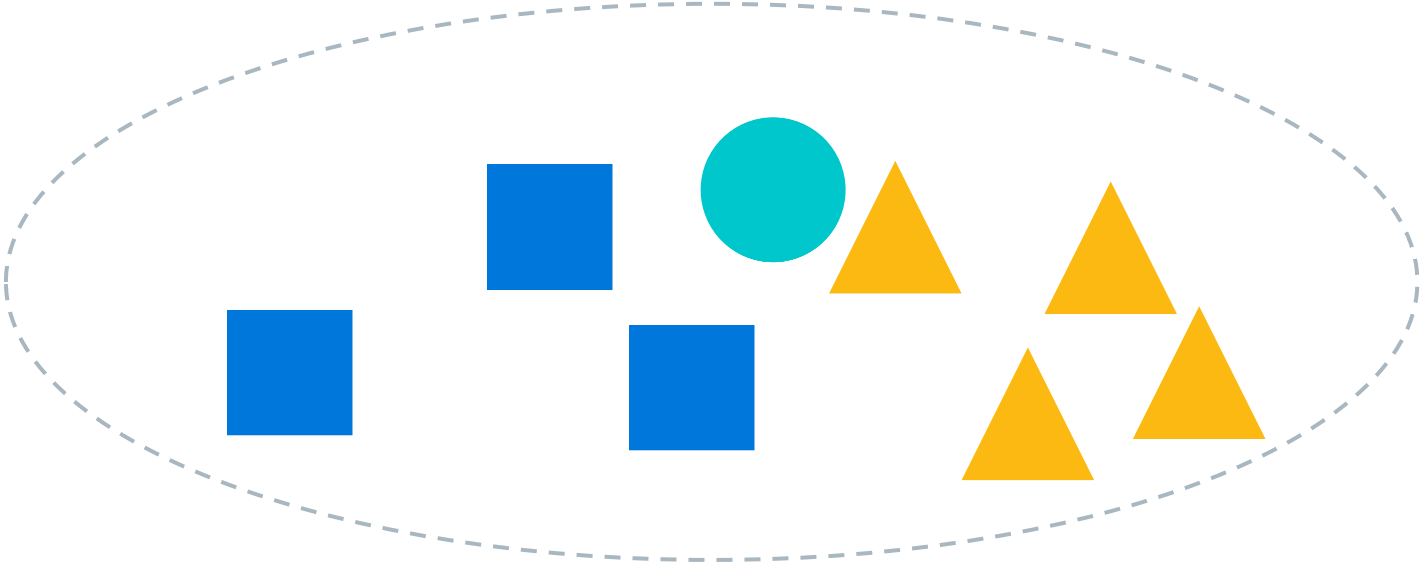
KNN: Logistic Regression Friendly Neighbor

As the number of k-nearest neighbors increase, the distribution of nearest classifications can change. If in our example we used **k = 3**, there are two squares over the one triangle, so our model would classify the new data point as a blue square.



KNN: Logistic Regression Friendly Neighbor

If we make our k-value too large, our KNN classification will classify all new data types as the most dominant classification. In this final part to our example, **k = 7** means that all known data points are considered, and our aqua circle would be classified as a yellow triangle.



KNN: Logistic Regression Friendly Neighbor

Because k can vary your results, the easiest technique for choosing a k value is to loop through a range of k and calculate the score. Choose the lowest value of k where the score starts to stabilize.

Note: We only use odd numbers so there are no ties between classes.

```
for k in range(1, 20, 2):
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(data, labels)
    score = knn.score(data, labels)
    print(f"K: {k}, Score: {score}")
```

Activity:

KNN

In this activity, you will revisit the malware detection problem and dataset. You will determine the best k-value in KNN to predict malware.

Suggested Time:

15 Minutes

Time’s up!

Let’s review

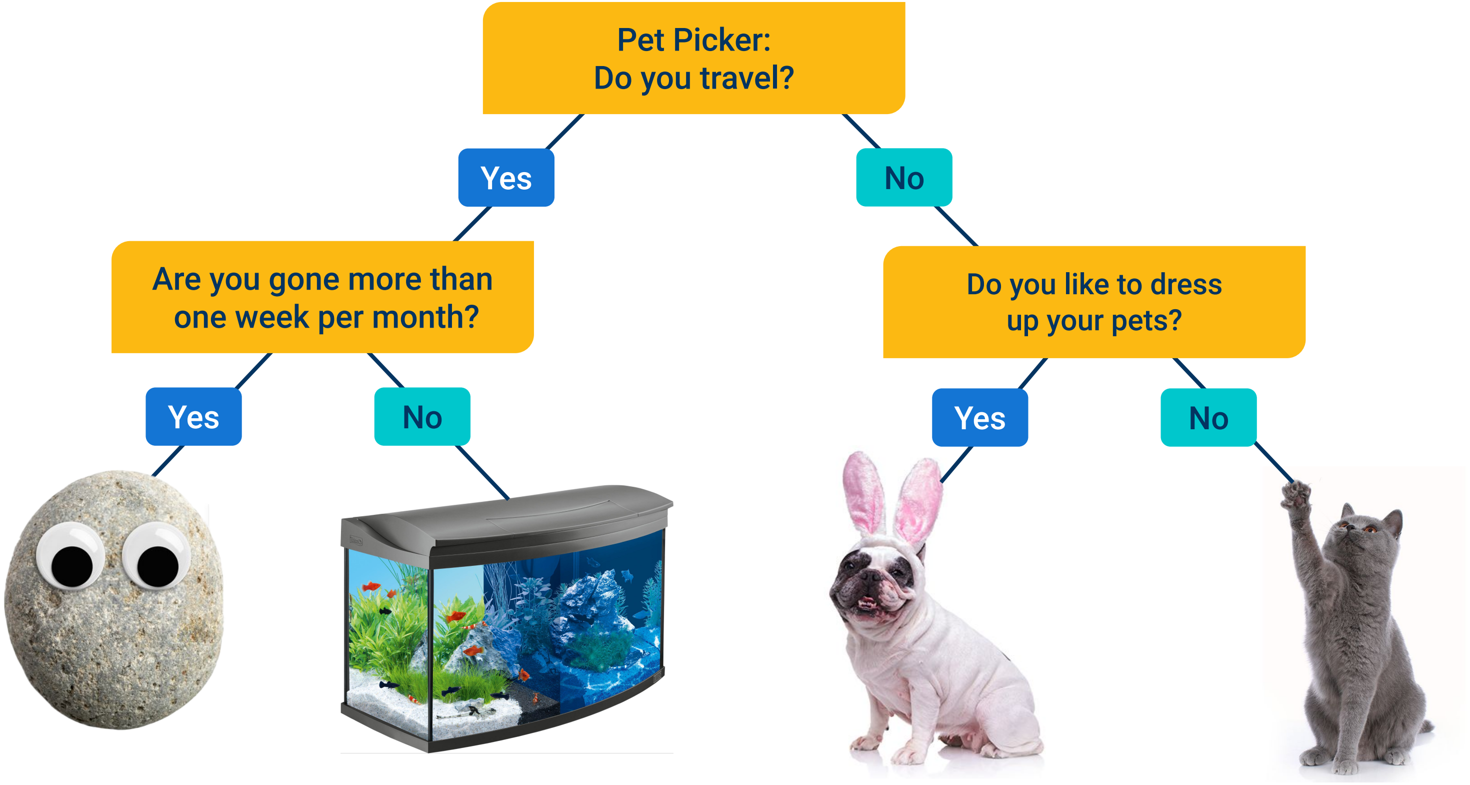
Questions?

Instructor **Demonstration**

Decision Trees

Decision Trees

Decision trees encode a series of true/false questions.



Pet Picker:
Do you travel?

Yes

No

Are you gone more than one week per month?

Do you like to dress up your pets?

Yes

No

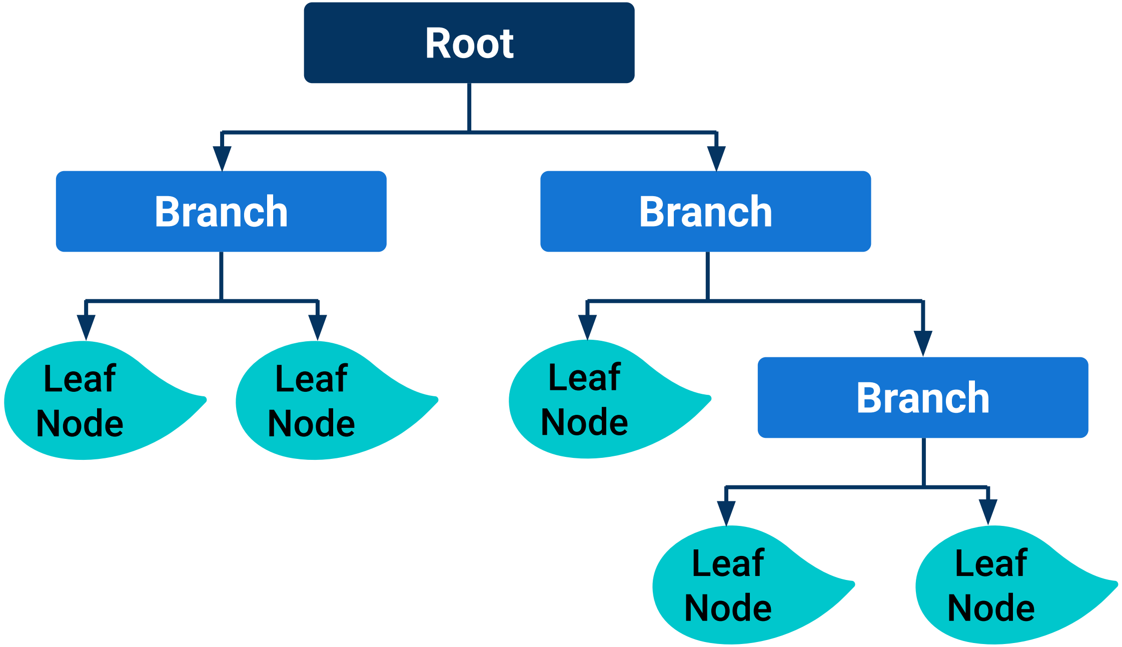
Yes

No

Decision Trees

Trees are a family of supervised learning models that provide an alternative to logistic regression and SVMs.

- **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 subnodes.
- **Splitting:** The process of dividing a node into two or more subnodes.
- **Child node:** Subnode of a parent node. Parent nodes split into child nodes.
- **Leaf or terminal node:** A node that does not split.



Decision Trees

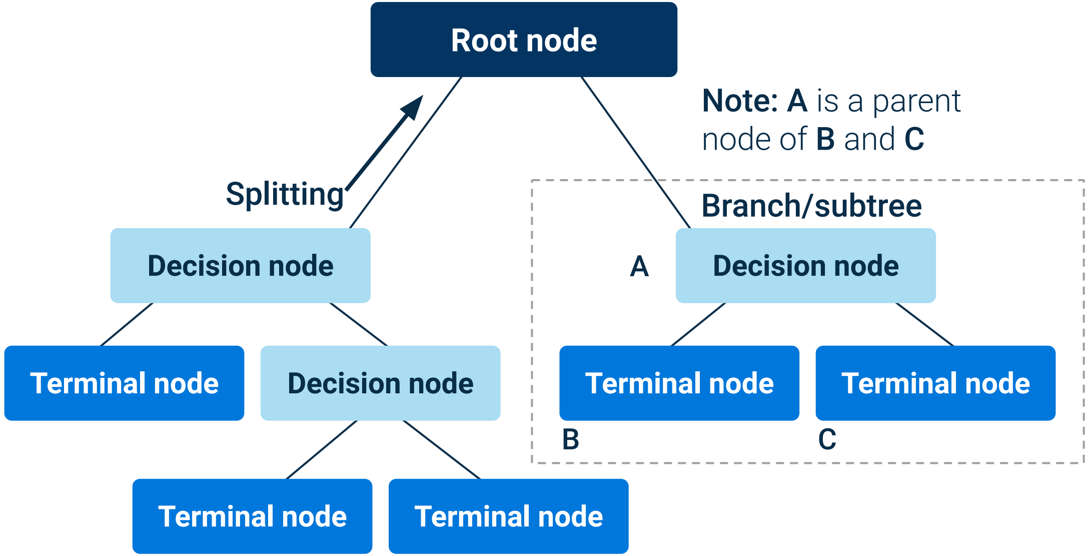
Trees are a family of supervised learning models that provide an alternative to logistic regression and SVMs.

Decision node: A parent node with at least one child node that is a terminal node.

Branch or subtree: A subsection of the entire tree.

Pruning: The process of removing subnodes of a decision node.

Tree depth: The number of decision nodes encountered before making a decision.



Note: **A** is a parent node of **B** and **C**

In contrast to linear models, tree-based algorithms can map nonlinear relationships in data.

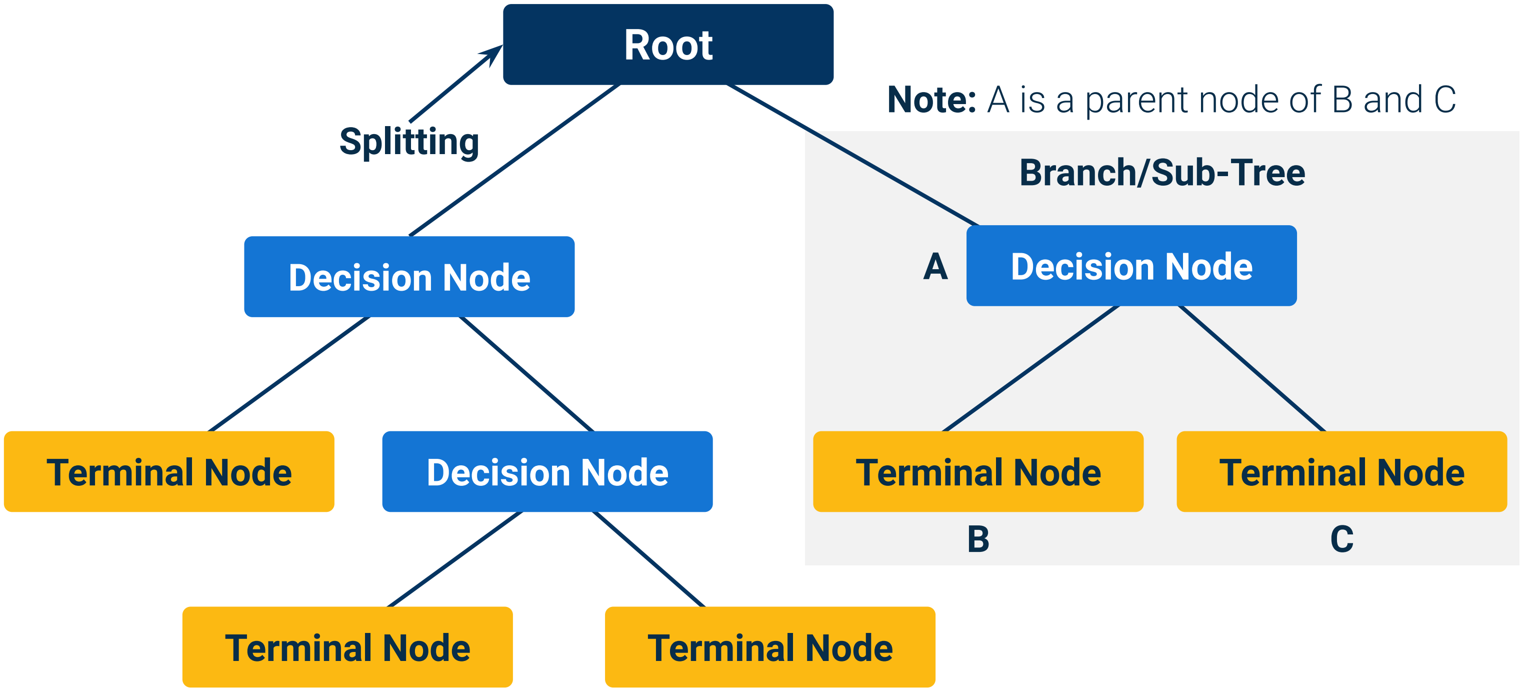
This is an important advantage of using trees.

Decision Trees: Key Terms

Root Node	A node that is divided into two or more homogeneous sets and represents the entire population or sample data.
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.
Leaf or Terminal Node	Nodes that do not split.
Branch or Sub-Tree	A subsection of the entire tree.
Splitting	The process of dividing a node into two or more sub-nodes.
Pruning	The process of removing sub-nodes of a decision node.
Trees Depth	The number of decision nodes that the algorithm encounters before it makes a decision.

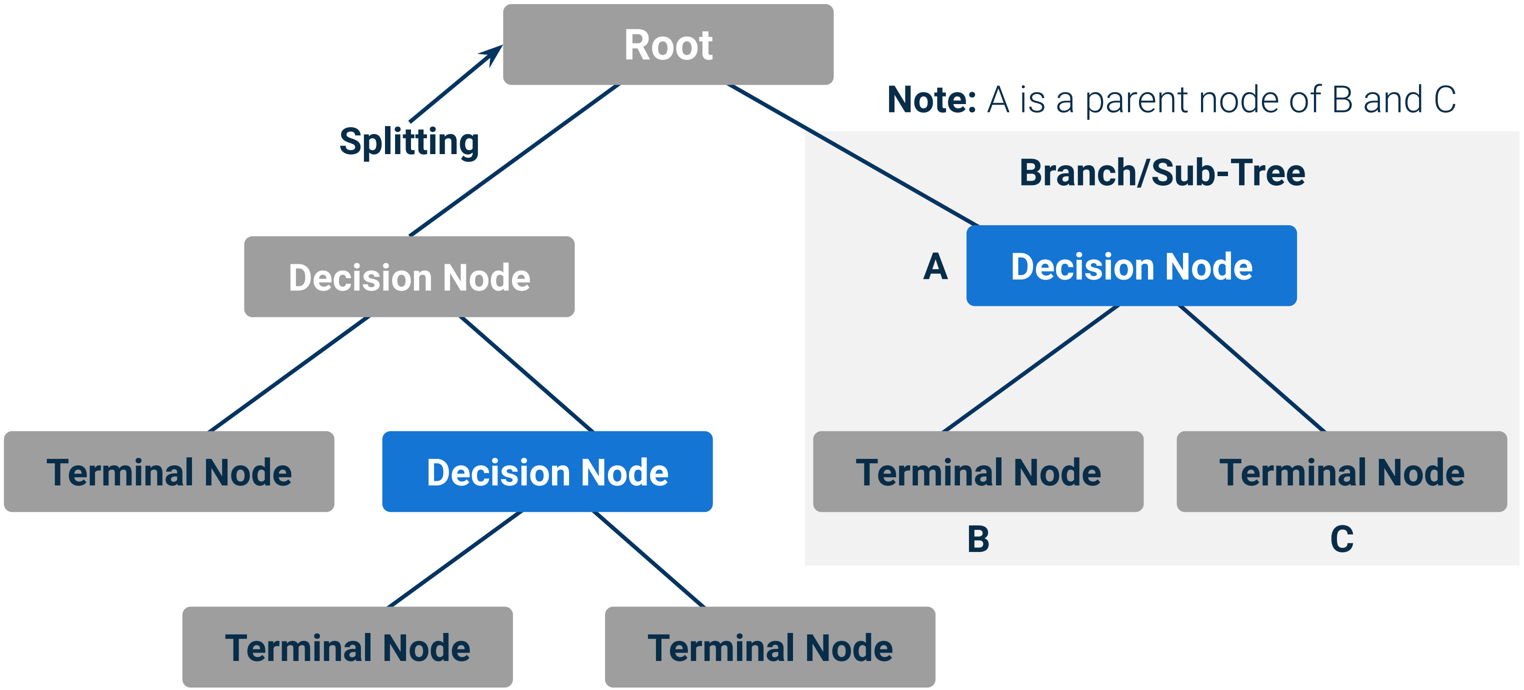
Root Node

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



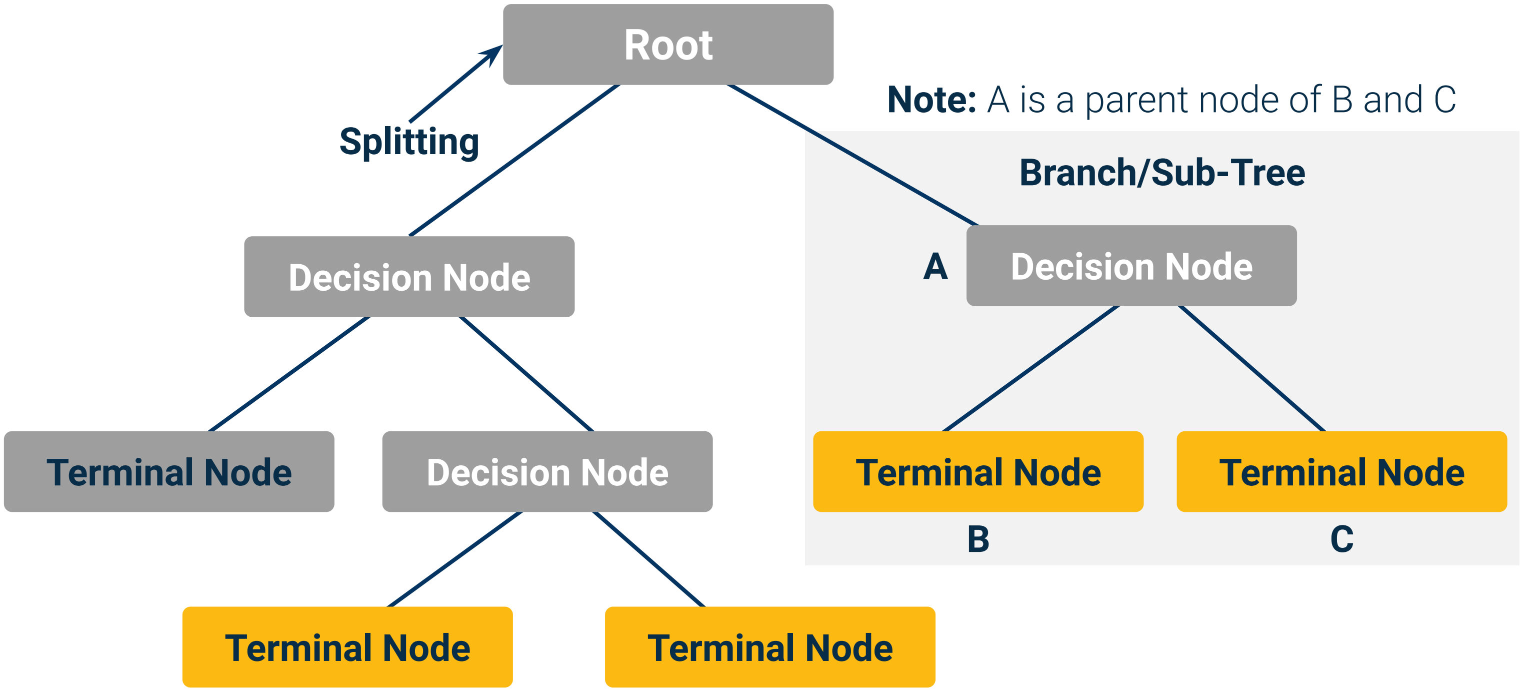
Parent Node

A node that is divided into sub-nodes.



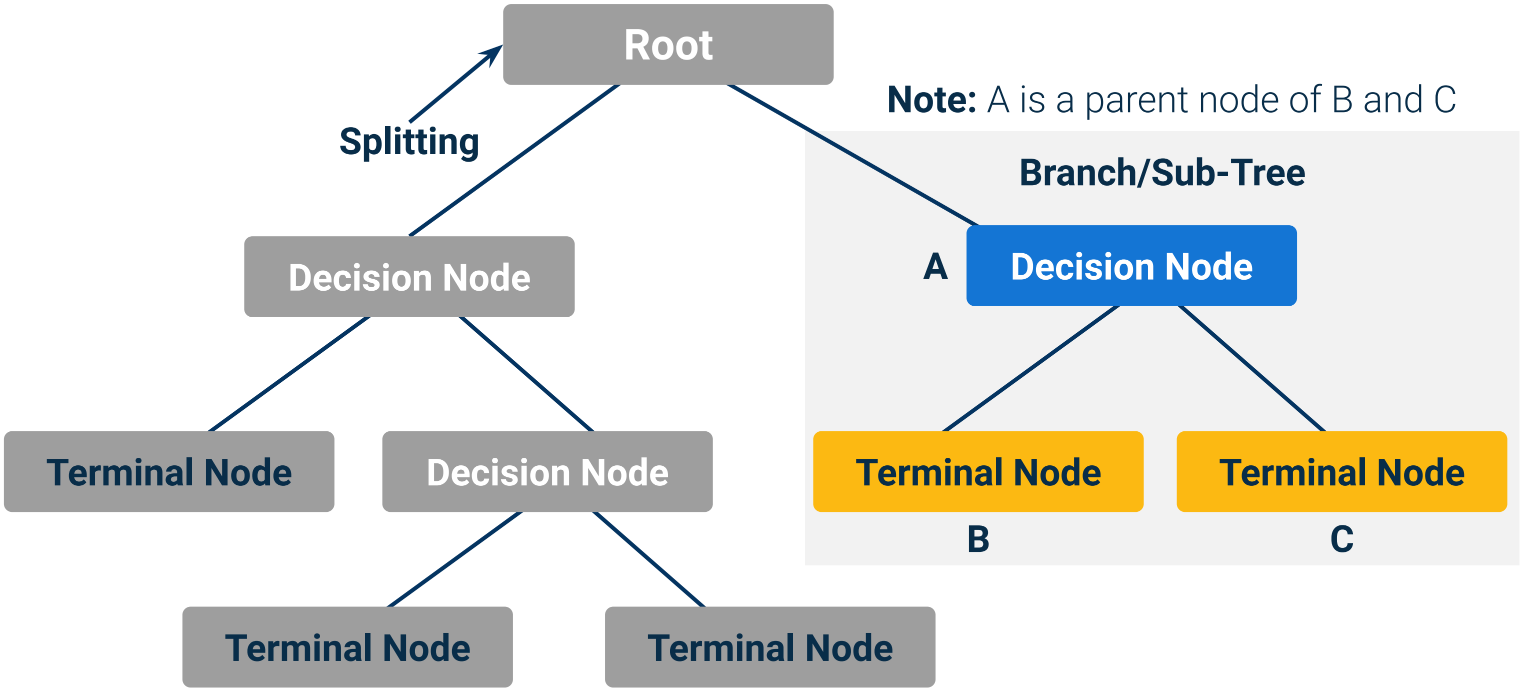
Child Node

Sub-nodes of a parent node.



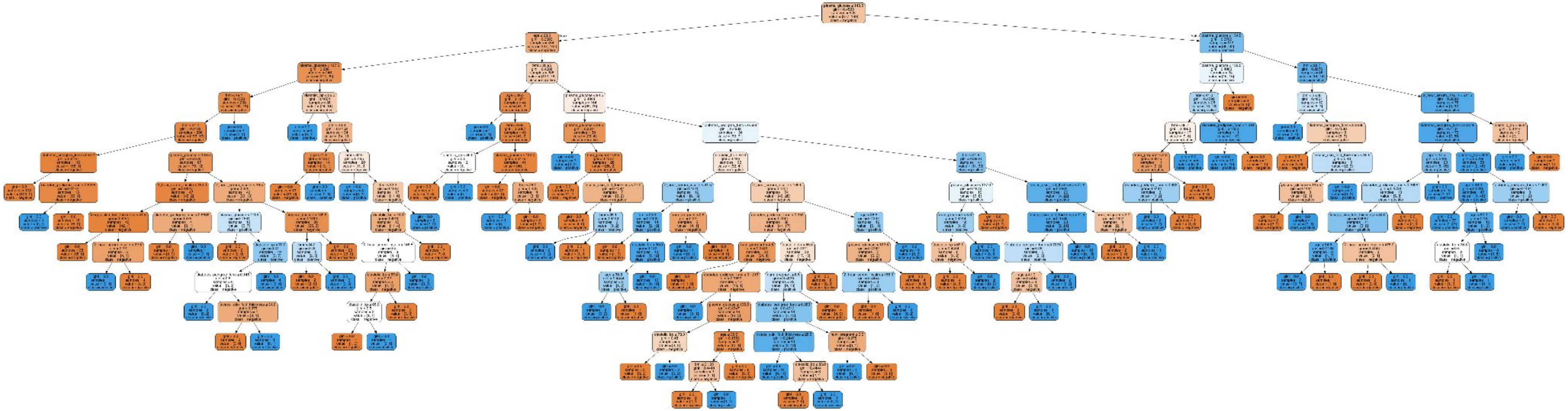
Decision Node

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



Decision Trees

Decision trees can become very complex and deep, depending on how many questions have to be answered. Deep and complex trees tend to overfit to the training data and do not generalize well to new data.



Activity:

Decision Tree

In this activity, you will return to the malware prediction problem. You will evaluate a decision tree model approach to the problem, and visualize a subset of the decision tree model.

Suggested Time:

15 Minutes

Time’s up!

Let’s review

Questions?

Break

15 mins

Instructor **Demonstration**

Overfitting

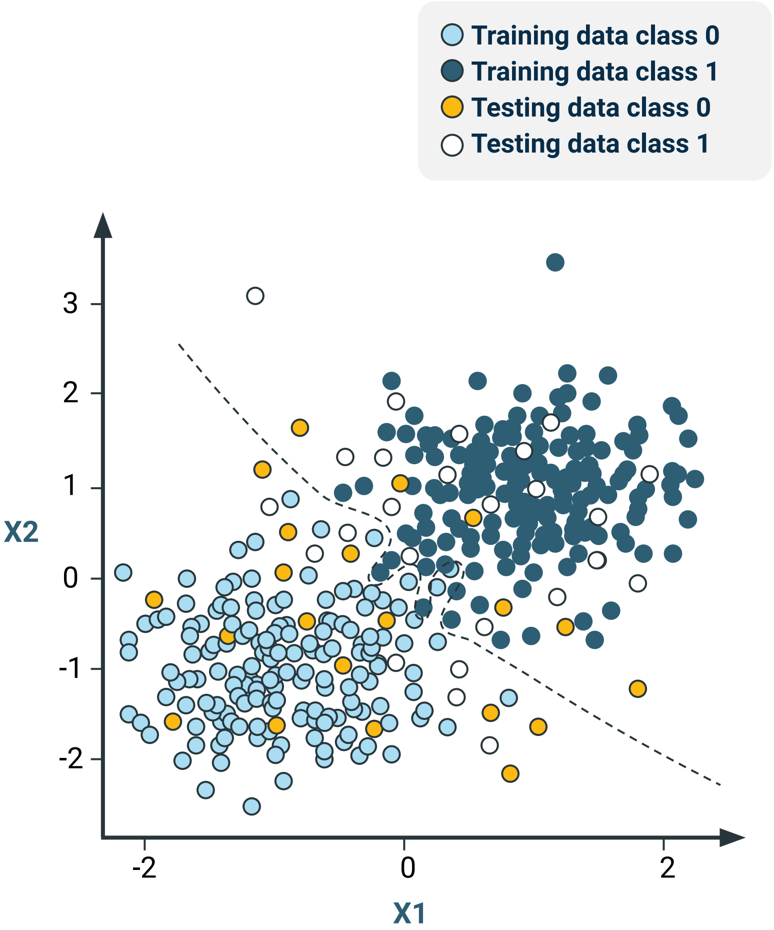
Overfitting

A quick recap of overfitting and its consequences:

In this image, the classification line correctly identifies all the training data points but deviates to incorrectly identify some of the testing data points.

A model that accurately predicts each training data point but misses a significant number of testing data points overfits the data.

Overfitting is like memorizing the answers to a practice exam; it will help you achieve a high score on a particular test, but the results will not be generalizable to a similar but different test.

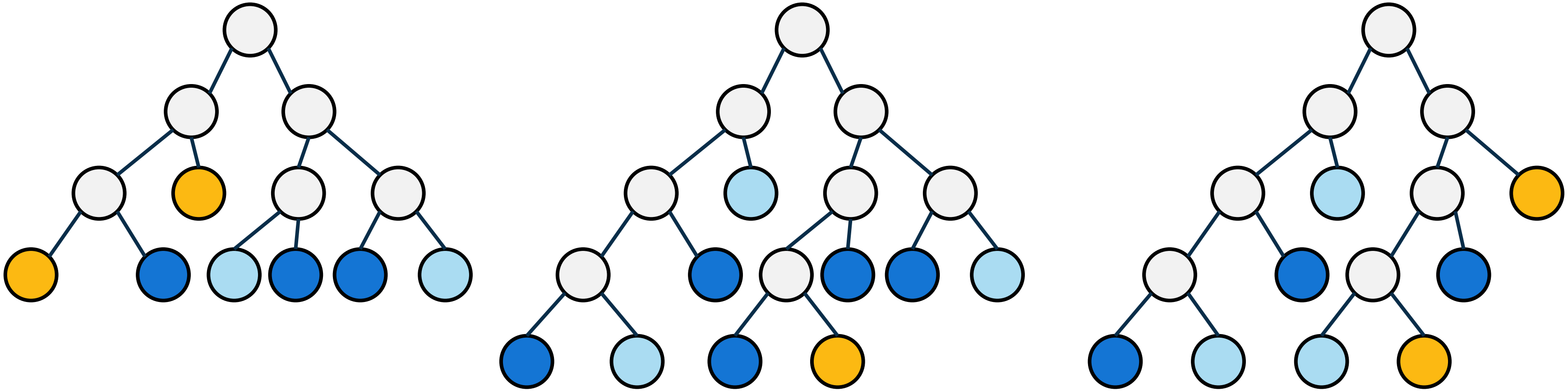


Instructor **Demonstration**

Ensemble Learning

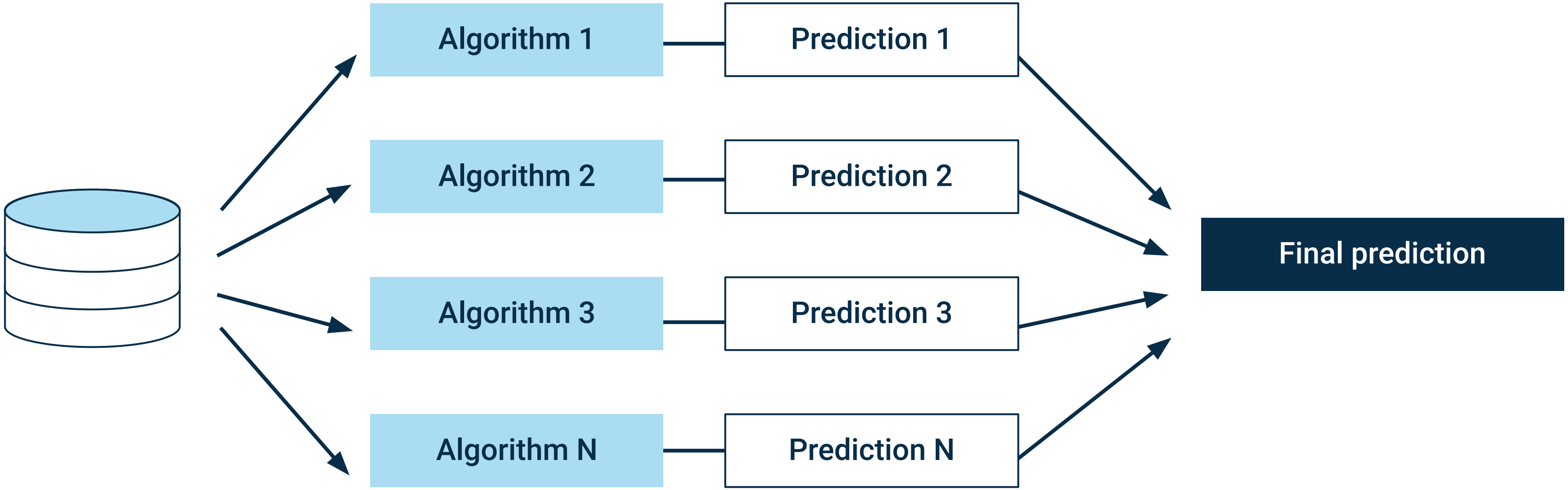
Ensemble Learning

Ensemble learning starts with the idea that two is better than one. A single decision tree might be prone to errors, but many of them can be combined to form a stronger model. A random forest model, for example, combines many decision trees into a forest of trees.



Ensemble Learning

Accumulating predictions to make a final prediction:



You will encounter algorithms that actually fail at learning adequately.

These algorithms/classifiers are considered **weak learners**.

Weak Learners

- Weak learners are a consequence of limited data to learn from.
- This may mean there are too few features, or that the data provided does not allow for data points to be classified.
- Weak learners make predictions that are only a little better than random chance.
- Individually, their predictions of the relationship between inputs and targets are not very accurate.

Despite their flaws, weak learners are still valuable in machine learning.

Can you guess how to make a weak learner perform **more accurately?**

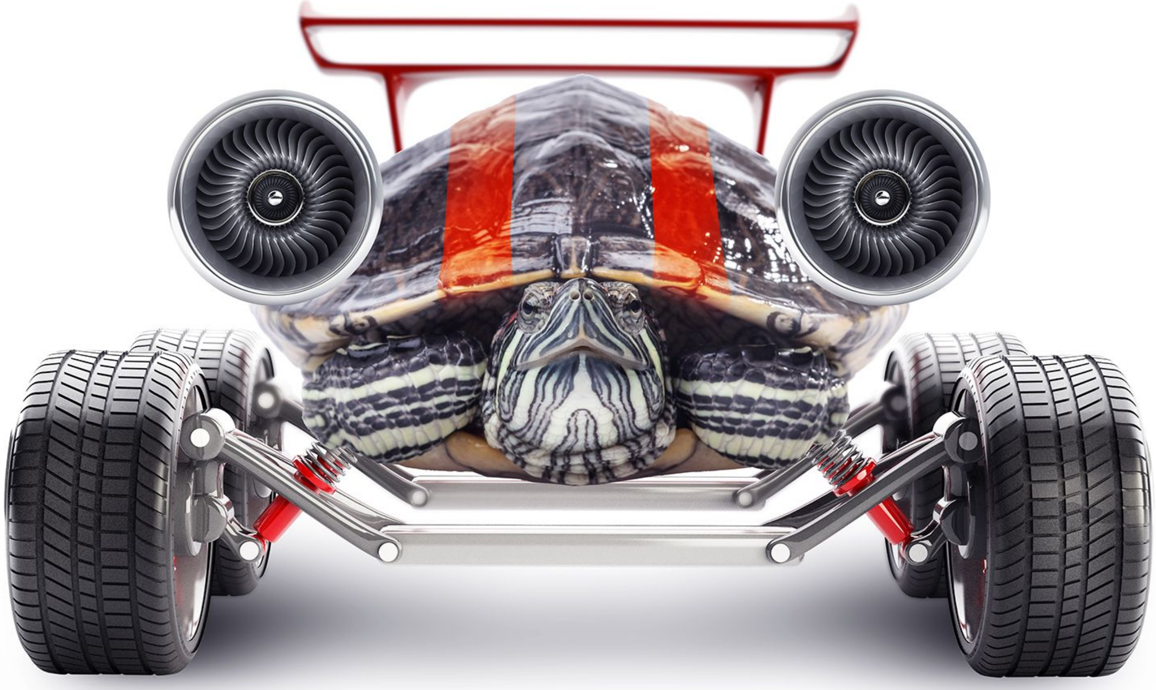
We can boost weak learners with other algorithms for an **ensemble approach**.

A decision tree can sometimes be classified as a weak learner. **Why?**

Ensemble Learners

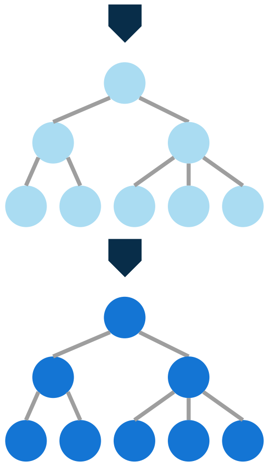
Working together, ensemble learners improve accuracy and robustness, and decrease variance.

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

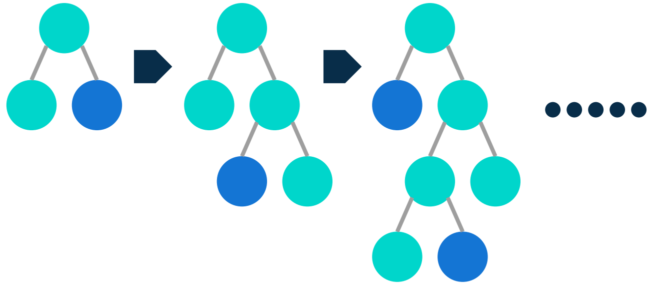


Introduction to Ensemble Learning

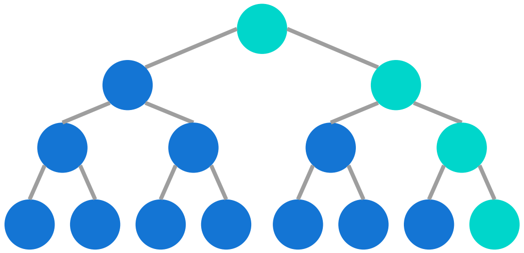
We have to combine weak learners by using a specific algorithm, such as:



Gradient boosted tree



XGBoost



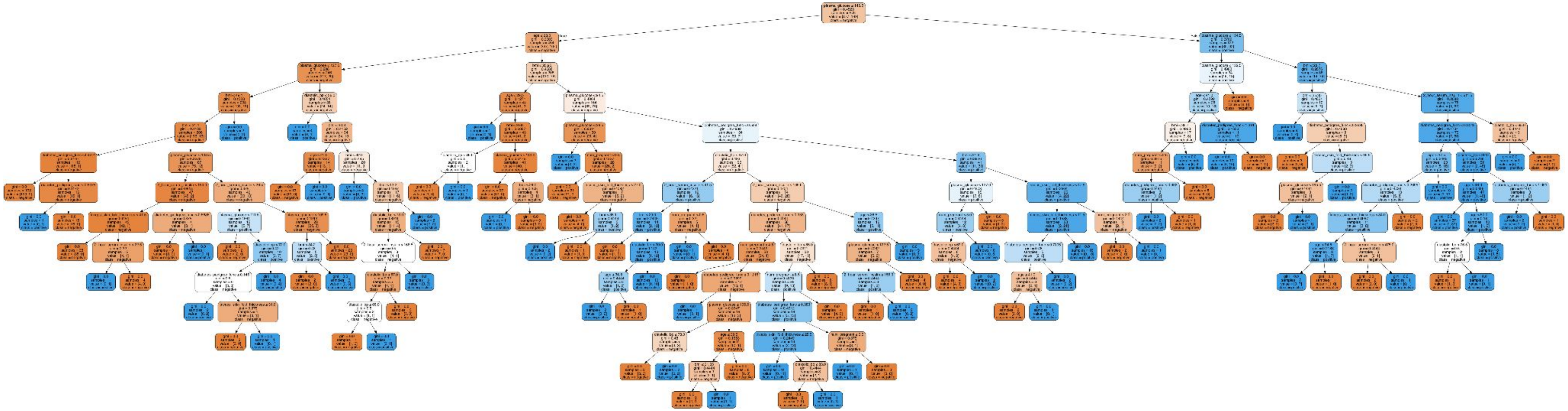
Random forests

Instructor **Demonstration**

Random Forest

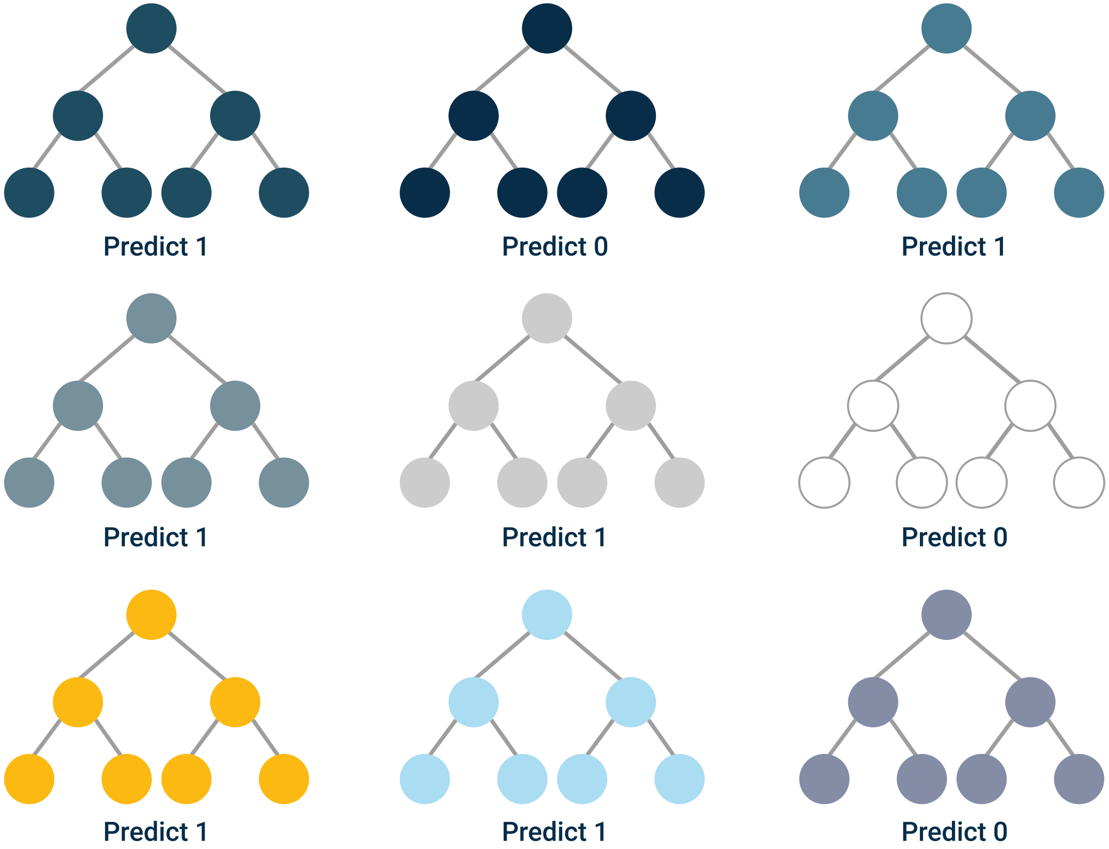
Random Forest Algorithm

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



Random Forest Algorithm

Each tree is much simpler because it is built from a subset of the data.



Random Forest Algorithm

- We create these simple trees by randomly sampling the data and creating a decision tree for only that small portion of data.
- Each simple tree is a weak classifier because it is only trained on a small piece of the original data.
- By itself, any single tree is only slightly better than a random guess.
- However, we can combine many slightly better than average small decision trees to create a strong classifier that has much better decision-making power.

Let's examine some of the benefits of the random forest algorithm.

Random Forest Algorithm: Benefits

01 Robust against overfitting as all weak learners are trained on different pieces of the data

02 Can rank the importance of input variables in a natural way

03 Can handle thousands of input variables without variable deletion

04 Robust to outliers and nonlinear data

05 Run efficiently on large datasets

Machine learning models can be “confused” by an overabundance of features, creating an unwanted fit to the noise of irrelevant features.

Feature selection is a process of selecting a subset of relevant features, reducing the width of our dataset.

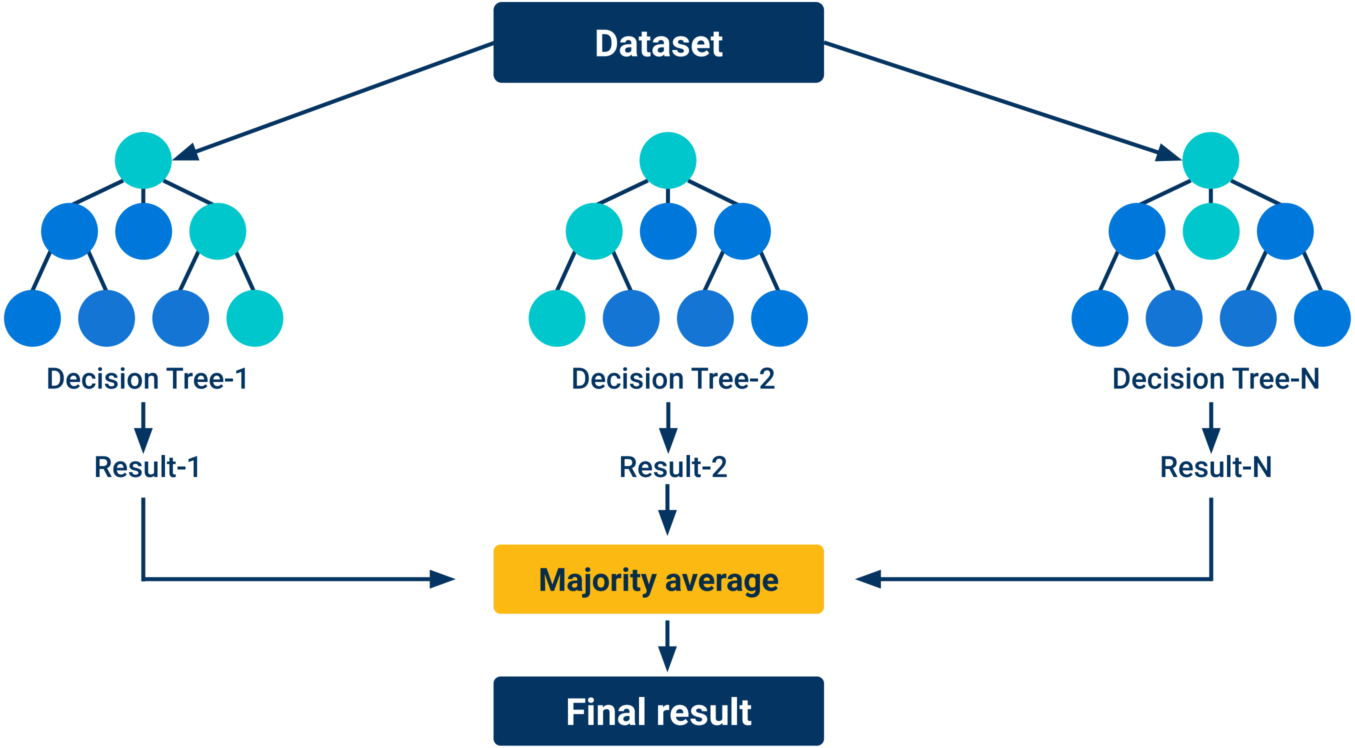
Feature Selection

There are many reasons to perform feature selection:

1. Simplified models are less likely to overfit.
2. Simplified models reduce training time.
3. Simplified models are easier to interpret, etc.

Feature Selection with Random Forests

There are many ways to perform feature selection. One technique uses the information from a Random Forest model.



Feature Selection with Random Forests

01

Remember, random forests use decision trees that try to select the best feature at every split.

02

How frequently a feature gets selected over the whole random forest model gives us an indication of how important that feature is.

03

Feature importances are accessible after fitting a `RandomForestClassifier` in scikit-learn with the `feature_importance`.

Activity:

Random Forest

In this activity, you will revisit the malware detection problem. This time, you will see how the model performs if it is designed as a random forest instead of logistic regression, SVM, KNN, and decision tree models.

Suggested Time:

15 Minutes

Time’s up!

Let’s review

Questions?

Instructor **Demonstration**

Bagging and Boosting

Extremely random trees (or ExtraTrees) is a similar method to random forests. Random forests resample the data for each new decision tree, but extremely random trees reuse the entire sample each time.

Extremely Random Trees vs. Random Forests

Extremely random trees

- Reuses the entire sample each time
- Picks at random the optimal split point, making it a much faster algorithm
- Increases bias (and decreases variance)

Random forests

- Resamples the data for each new decision tree
- Chooses the optimal split point for each decision

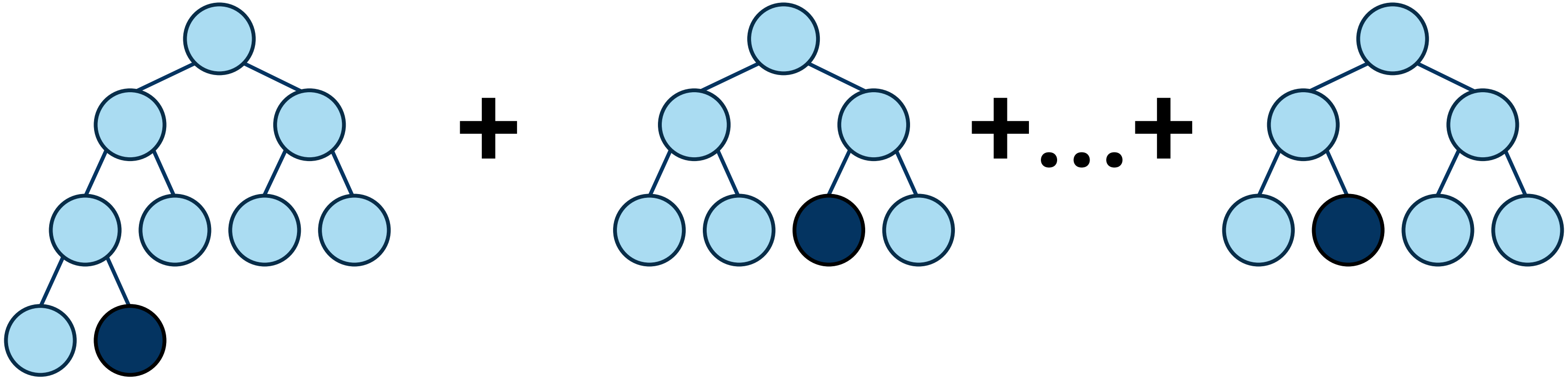
This is a very effective technique in terms of **accuracy**.

Even though it takes much longer to fit, the increase in accuracy is often worth the extra time.

Boosting is an aggregation method where individual weak learners get “boosted” if they are contributing to correct solutions, but get diminished if they are not.

Boosting

Boosting is a powerful extension of ensemble learning. Boosting is an aggregation method where individual weak learners get “boosted” if they are contributing correct solutions, but get diminished if they are not. While we can use boosting methods for regression problems, our focus here is on classification.



Instructor **Demonstration**

Bagging and Boosting in Practice

Let’s **recap**

Review the Class Objective

In this lesson you learned how to:

1. Explain how the KNN algorithm works as a classifier and how it differs from other classifiers.
2. Explain how decision trees and random forest work as classifiers and how they differ from each other.
3. Apply fundamental classification algorithms, namely random forest, decision trees, and KNN in machine learning models.

Next

In the next lesson, you will…

Questions?

The End