Classification Models

Data Boot Camp

Lesson 20.2


WELCOME

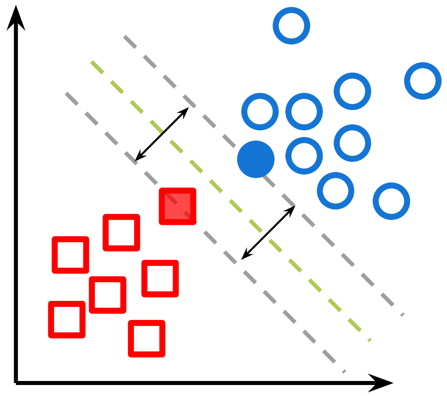
Class Objectives

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

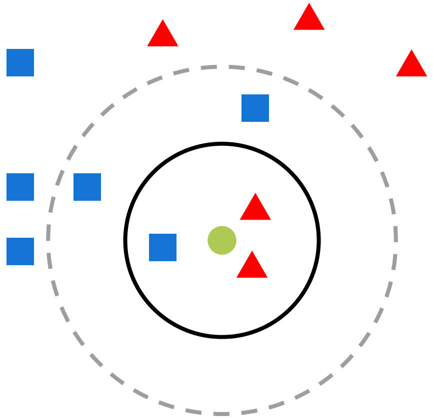
- Deploy various machine learning models for classification, such as support vector machines (SVM), k-nearest neighbors, and random forests.
- Identify when categorical variables are useful for a machine learning algorithm.
- Perform feature engineering on categorical features and convert labels to numerical class representations.
- Recognize the type of business cases where decision trees and random forests are a suitable solution for classification problems.
- Demonstrate how random forests avoid overfitting and perform better than decision trees.
- Identify the pros and cons of tree-based algorithms.
- Understand the implications of overfitting and how boosting and bagging can help to deal with it.
- Apply Gradient Tree Boosting models in classification problems.

Trees and Ensemble Learning

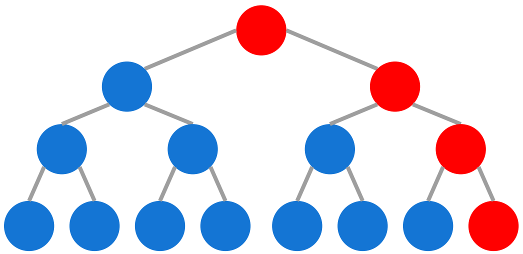
Today you will learn about several new machine learning models:



Support vector machines (SVM)



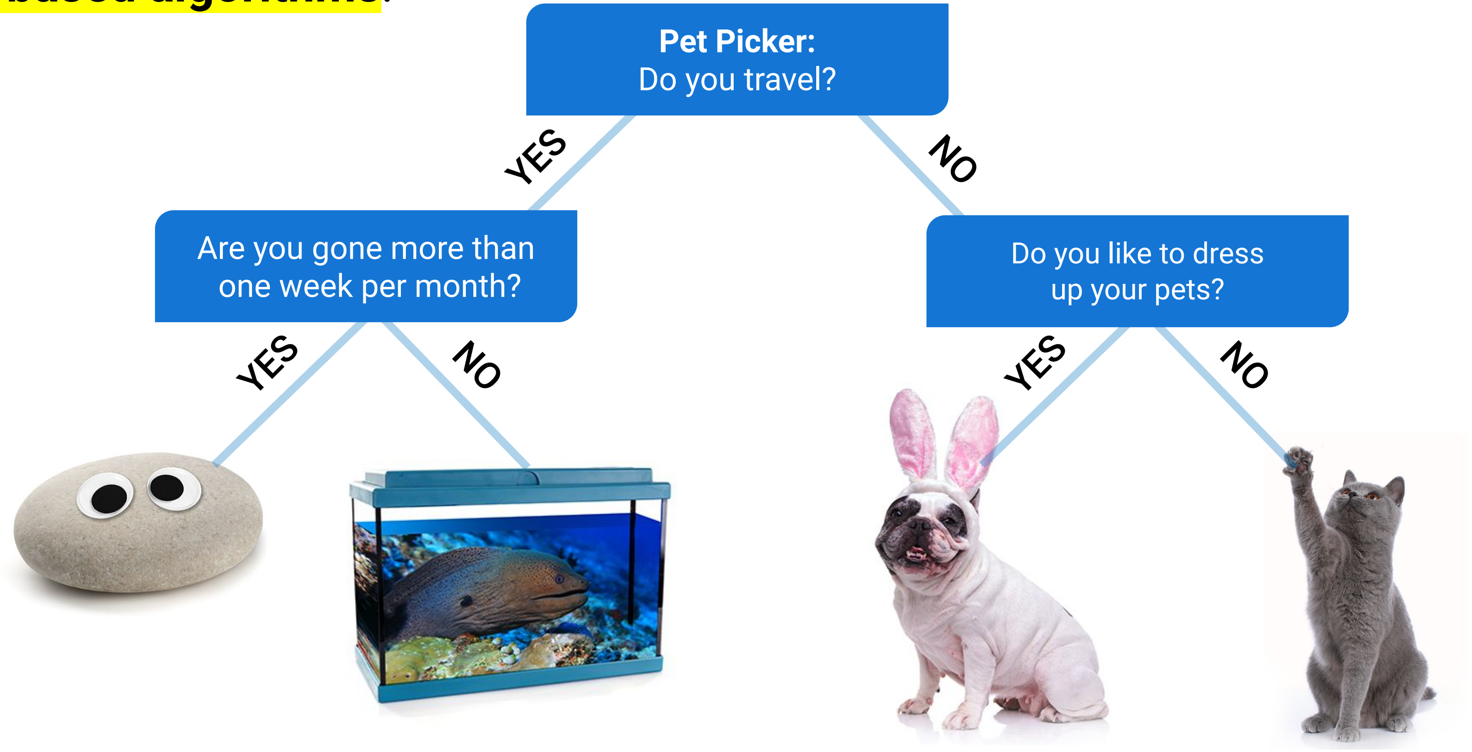
k-nearest neighbors



Random forests

Trees and Ensemble Learning

You will also be introduced to a new family of machine learning algorithms: **tree-based algorithms**.



Tree-based algorithms are supervised learning methods that programmers and analysts use primarily for classifications and regression problems.

Trees and Ensemble Learning

This class will cover the following algorithms and methods:

01 Decision trees

02 Random forests

03 Weak learners

04 Ensemble methods

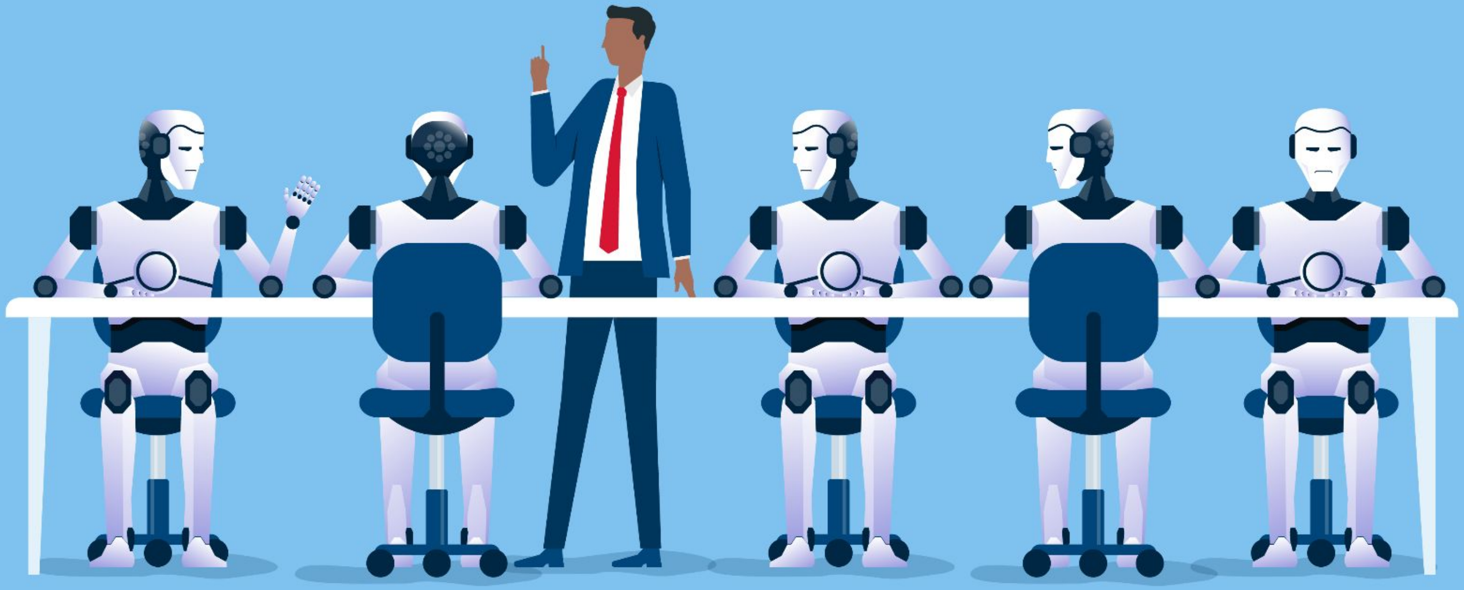

Questions?

Support Vector Machines

Support Vector Machines

sklearn follows a common pattern of model-fit-predict, which allows machine learning engineers to train, test, and evaluate a variety of machine learning models.

To illustrate this, we will look at a new model called a **support vector machine**, or SVM.



Support Vector Machines (SVM) is a supervised learning model that we can use for classification and regression analysis. SVM separates classes of data points into multidimensional space.

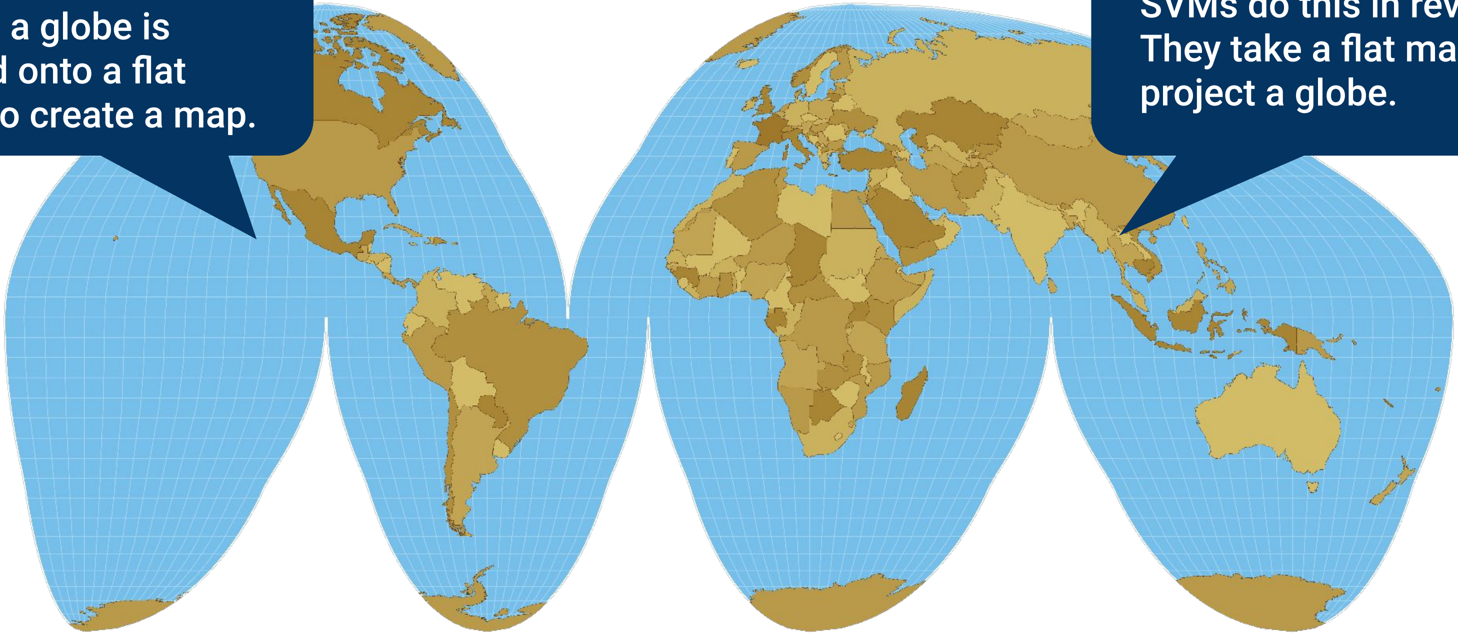
Support vector machines are a widely applied model in Data Science, especially for assessing credit risk and fraud detection.

Support Vector Machines

The idea behind SVMs is that a dataset and its labels are projected into a higher dimensional space.

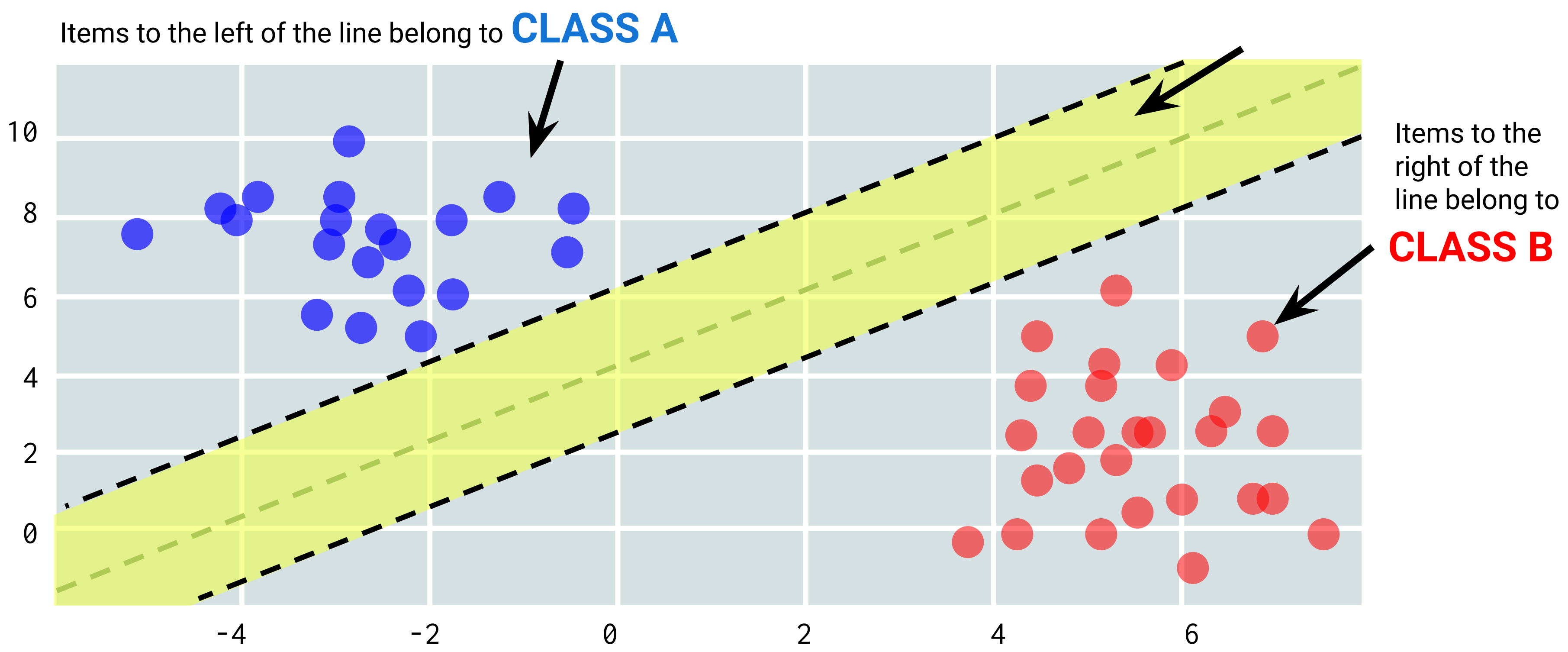
You might be familiar with how a globe is projected onto a flat surface to create a map.

SVMs do this in reverse. They take a flat map and project a globe.



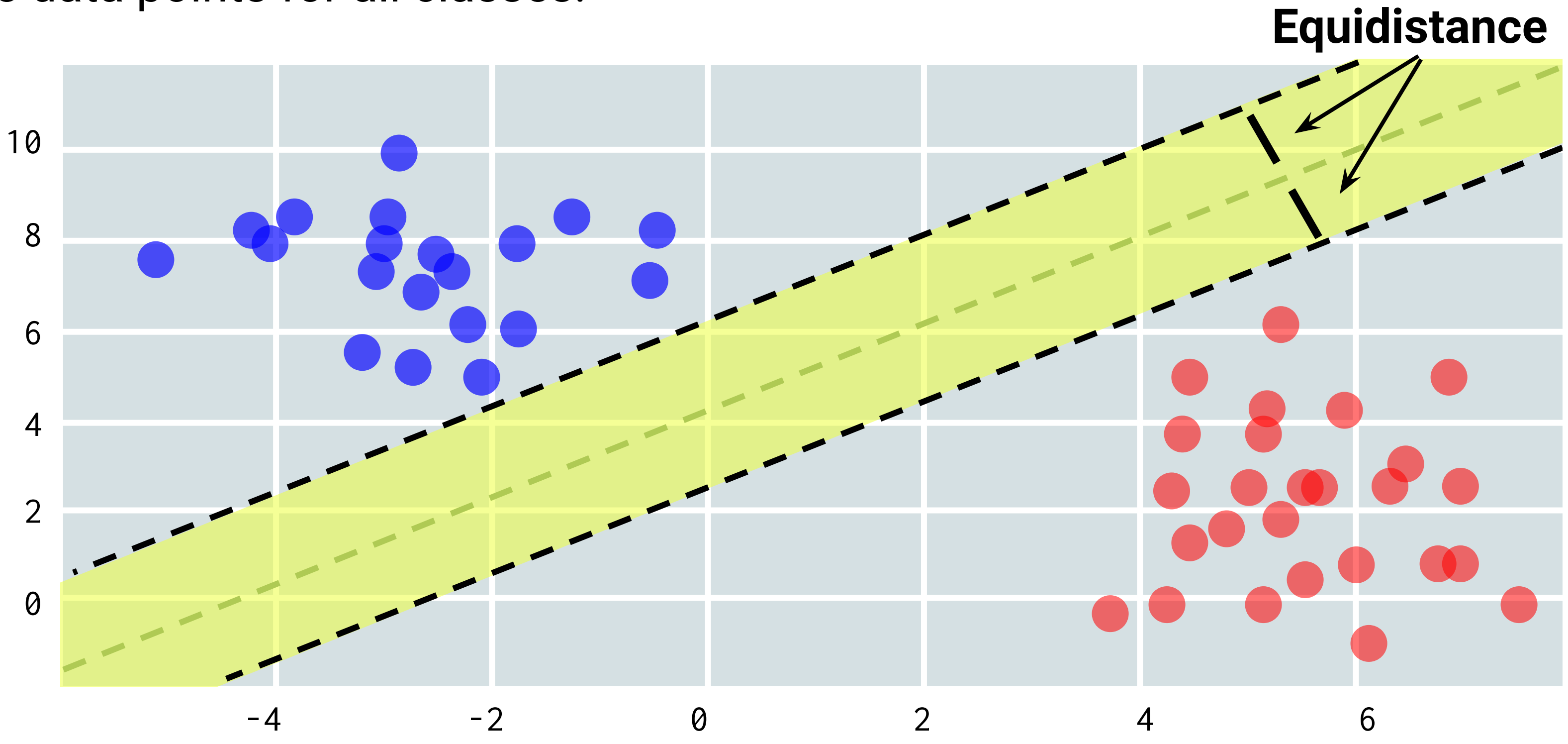
Support Vector Machines

This boundary's projection is called a **hyperplane**, and we can use it to classify the points.



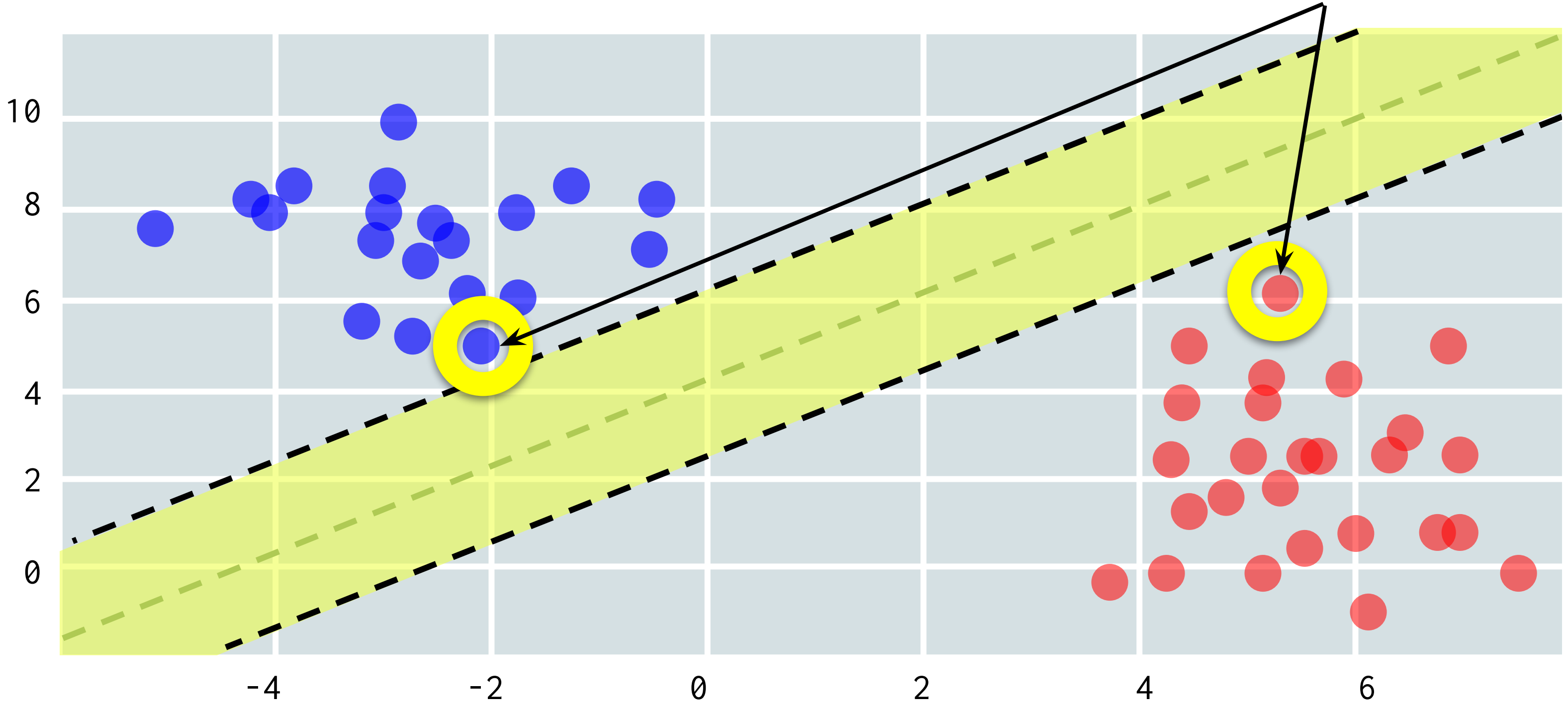
Support Vector Machines

The goal with hyperplanes is to get the margin of the hyperplane equidistant to the data points for all classes.



Support Vector Machines

The data closest to/within the margin of the hyperplane are called support vectors. They define the boundaries of the hyperplane.



These points are sometimes the most difficult to classify because they live closest to the margin and could belong to either class.

Instructor Demonstration

Support Vector Machines


Questions?

Activity: Predicting Occupancy

In this activity, you will build an SVM classifier to predict if an office space is occupied for a set of input features.

Suggested Time:

15 Minutes

Time's Up! Let's Review.

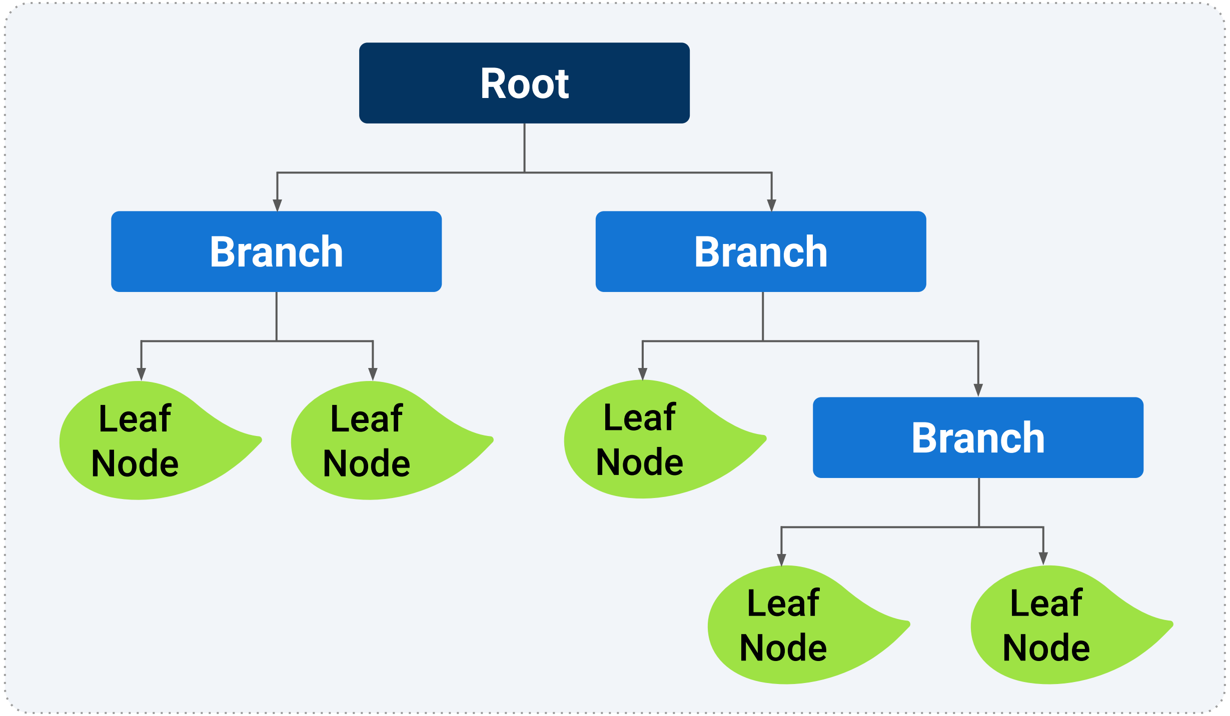
Questions?

Decision Trees

Decision Trees

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

First, we'll look at the basic tree model called a **decision tree**.



Decision Trees

Unlike logistic regression and support vector machines, decision trees are easy to audit, which is important for business applications.

In other words, you can trace the decision logic throughout each step of the model to see how the model reached the final prediction.

This may be critical if you need to justify a loan decision or other financial decision.

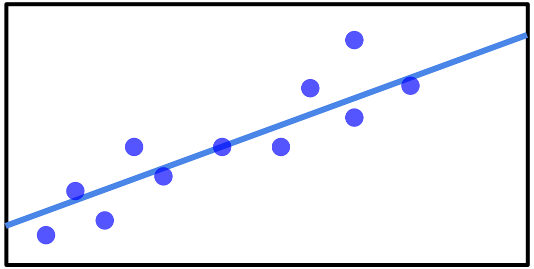
In contrast to linear models, tree-based algorithms can map non-linear relationships in data.

This is an important advantage of using trees.

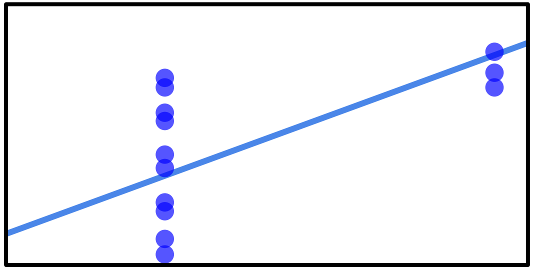
Linear vs. Non-Linear Models

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

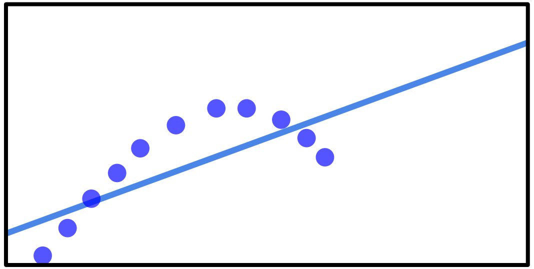
Linear



Non-linear



Non-linear



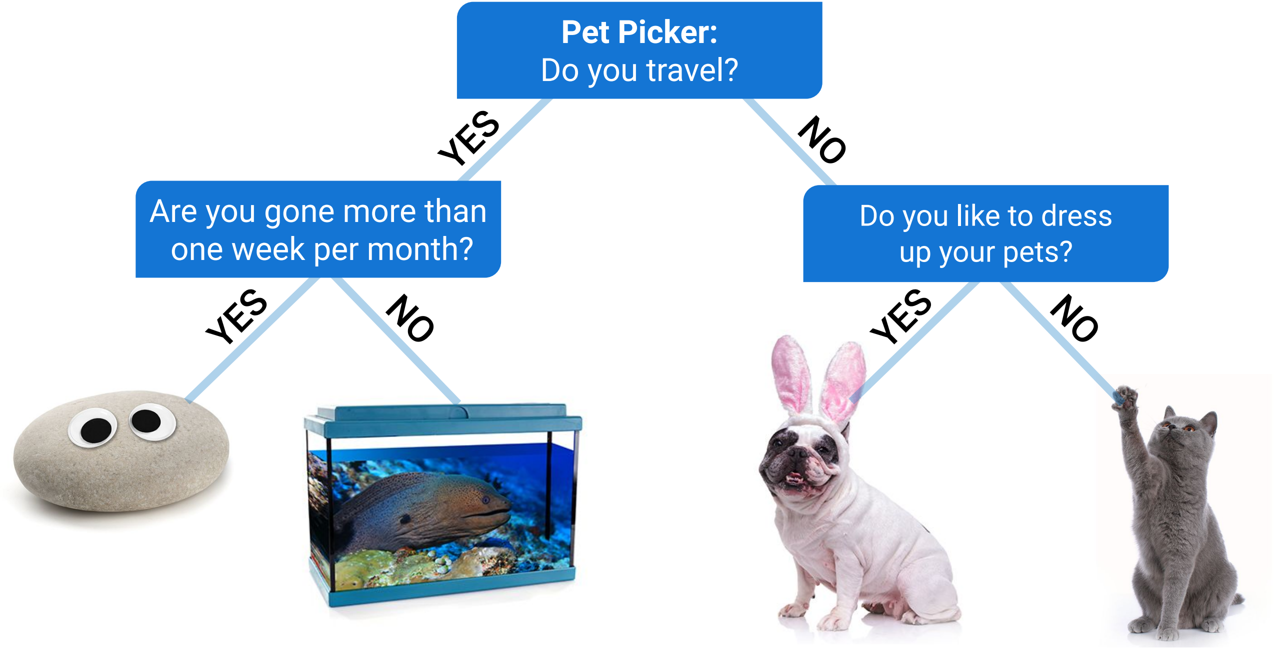
Predicting the price of a house based on its size is an example of a **linear problem**.

Predicting if a credit application is going to be fraudulent is an example of a **non-linear problem**.

Decision Trees

Decision trees encode a series of `True/False` questions.

`True/False` questions can be represented with a series of if/else statements.



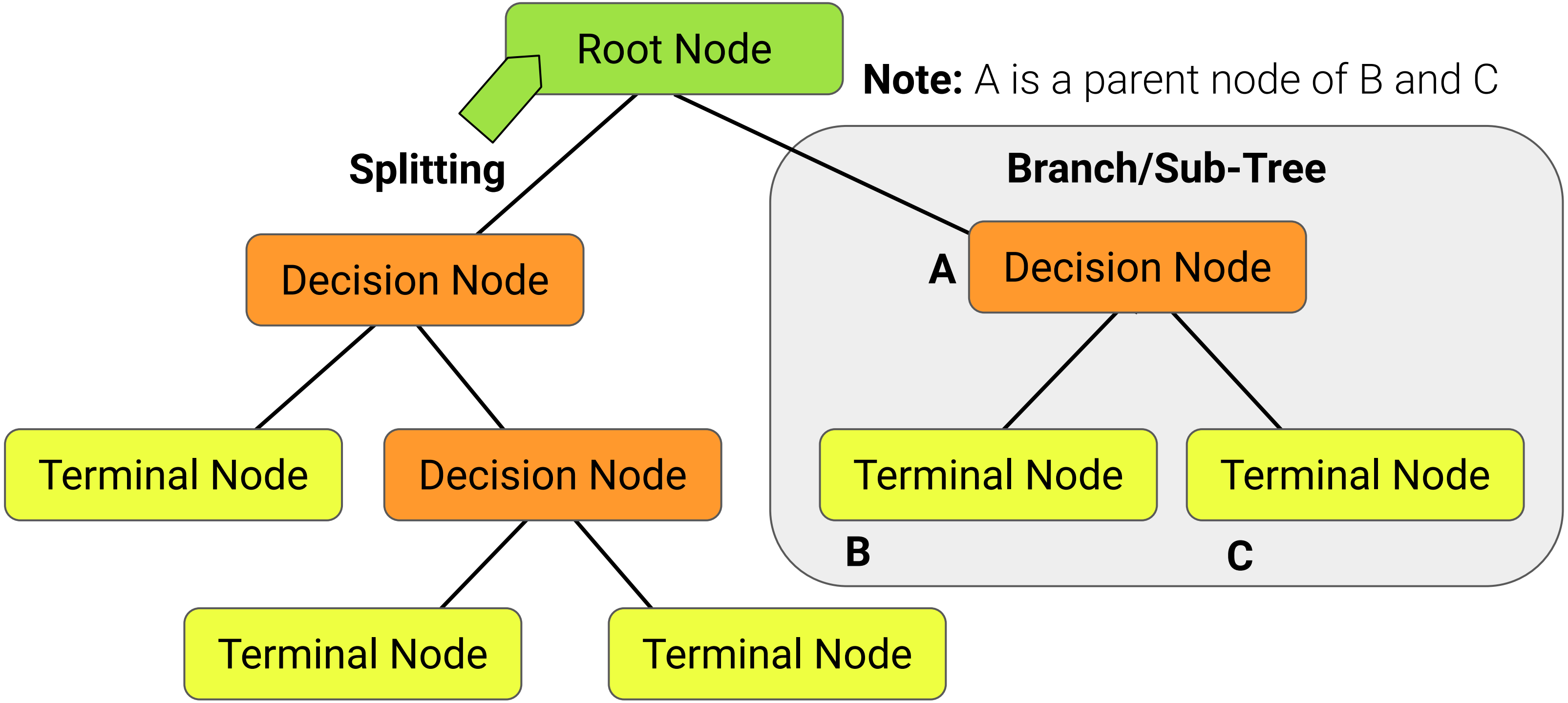

```
if (travel):
  if (time > week):
    print("Rock")
  else:
    print("Fish")
else:
  if (dress_up):
    print("Dog")
  else:
    print("Cat")
```

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.

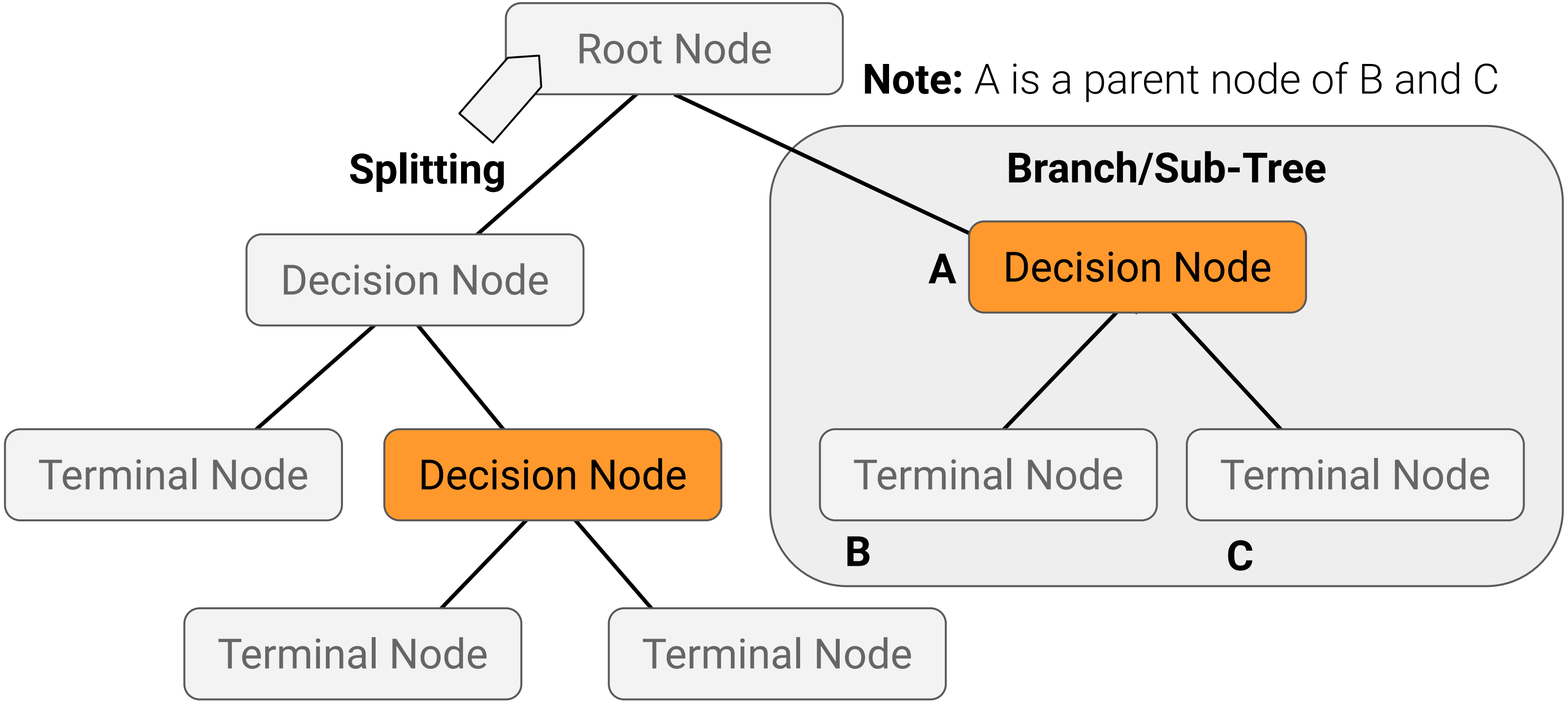
Root Node

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



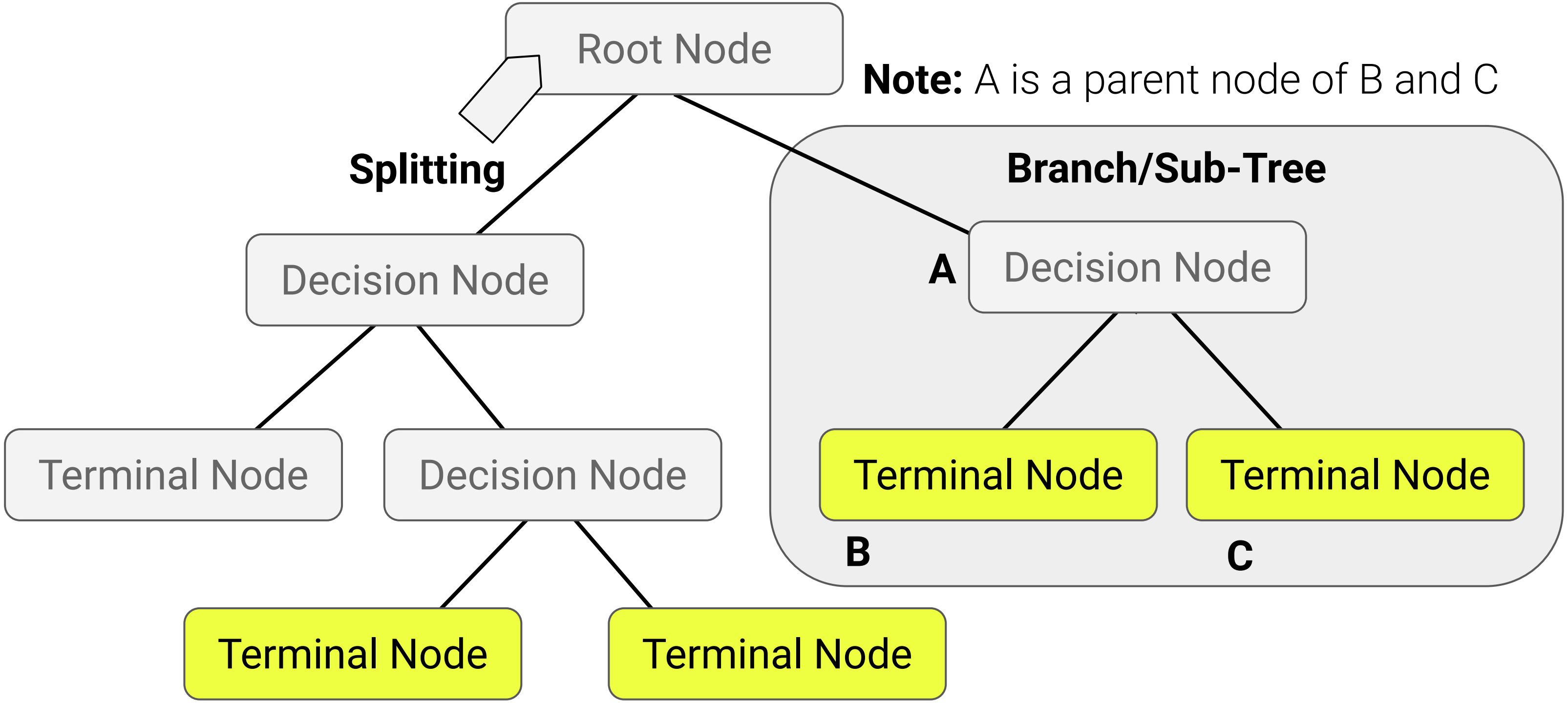
Parent Node

A node that is divided into sub-nodes.



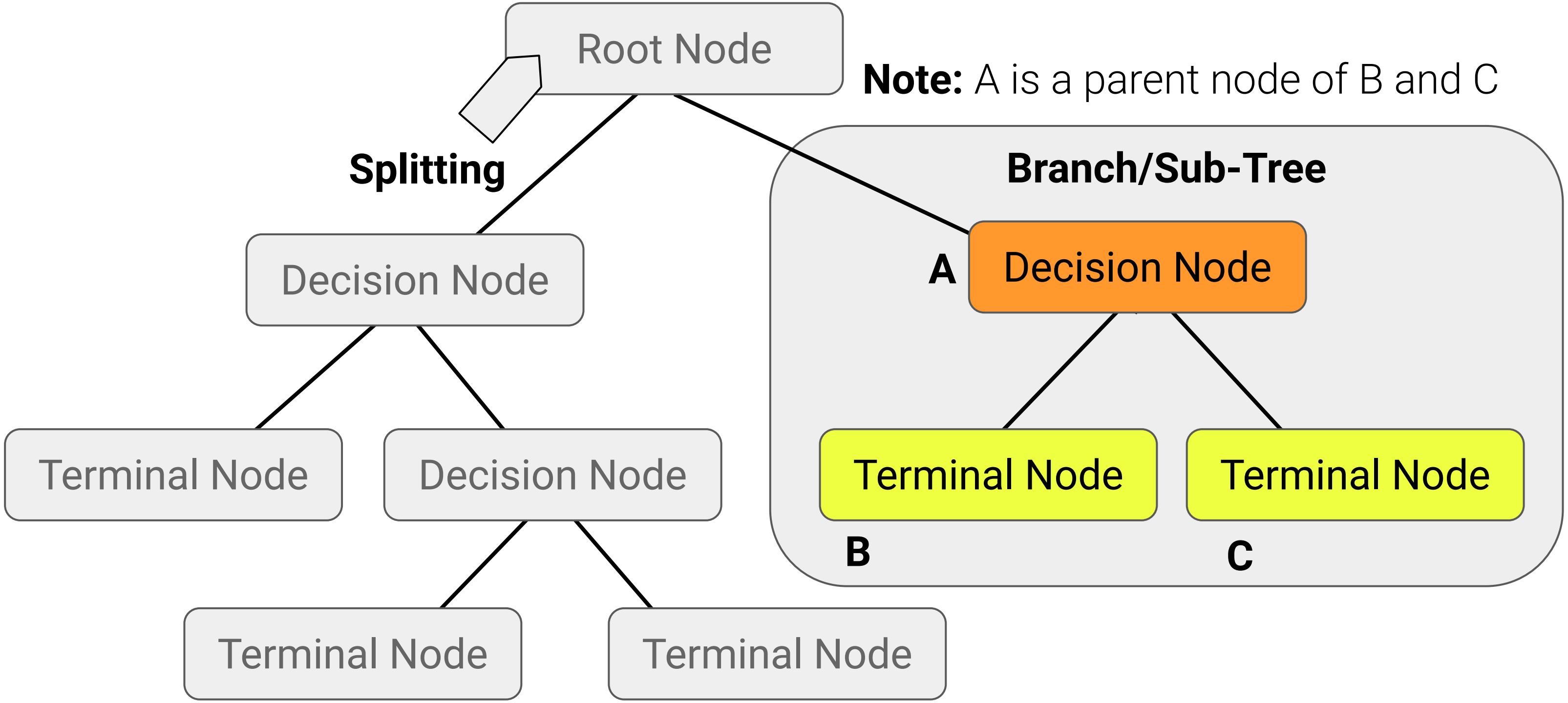
Child Node

Sub-nodes of a parent node.



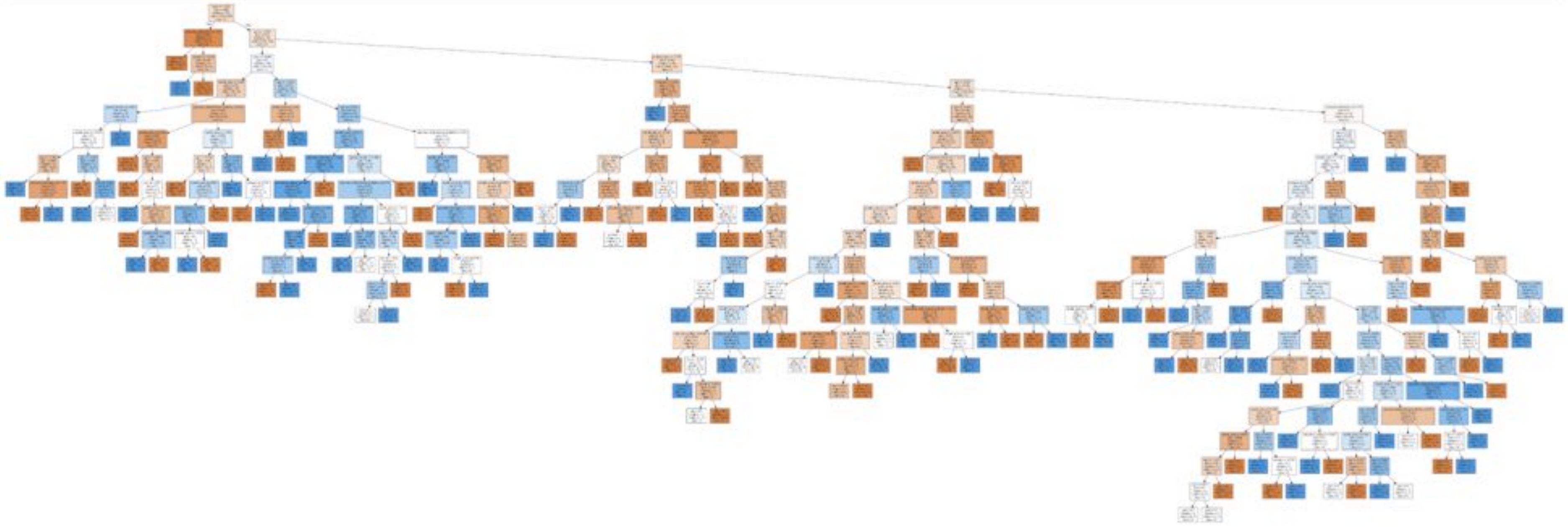
Decision Node

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.



Instructor Demonstration

Decision Trees

Questions?

Activity: Predicting Fraudulent Loan Applications

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

Suggested Time:

10 minutes

Time’s Up! Let’s Review.


Questions?


Break
Countdown timer
15:00
(with alarm)

Introduction to Ensemble Learning

The Classification Algorithm Race

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

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 learning approach.

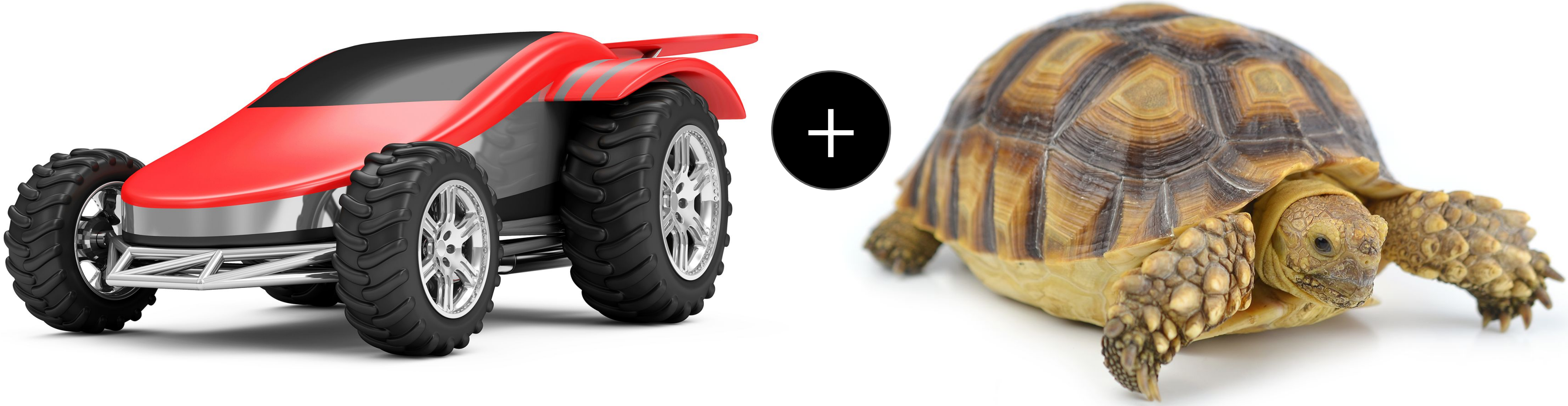
A decision tree can sometimes be classified as a weak learner. Why?

This can happen when a decision tree has only one branch (i.e., a stump).

Weak Learners Are Still Valuable in Machine Learning

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

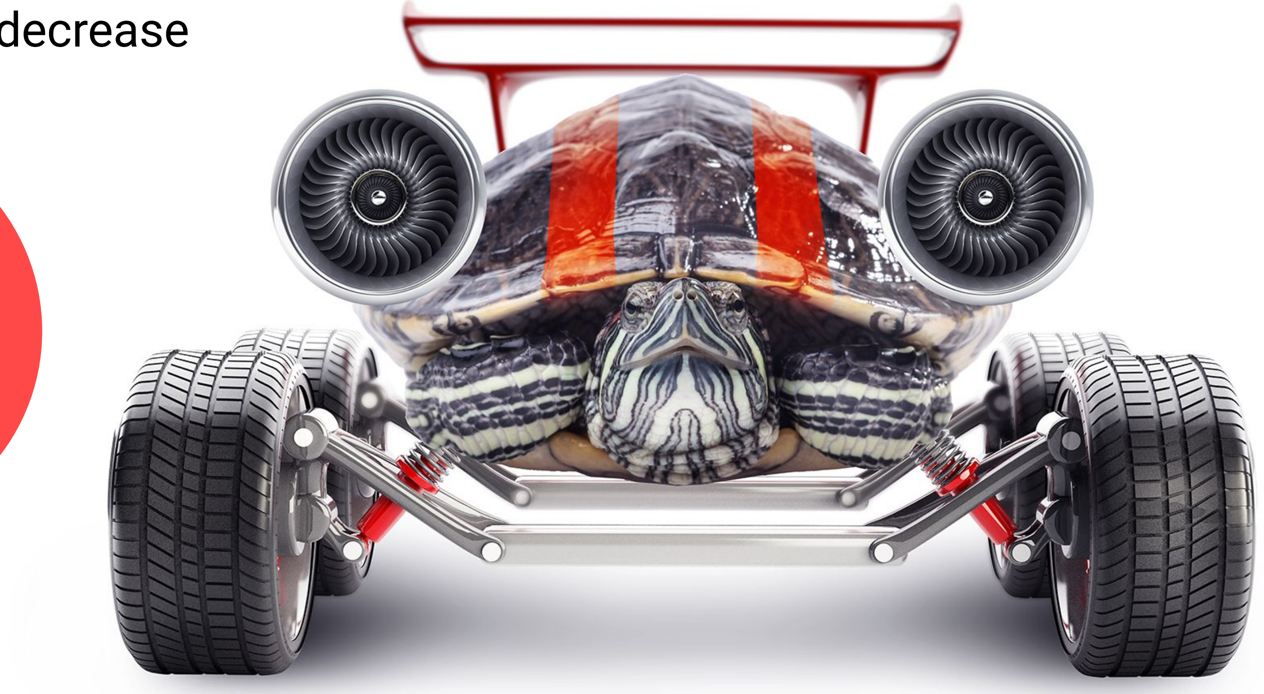
Combined weak learners are an example of ensemble learning:



Ensemble Learners

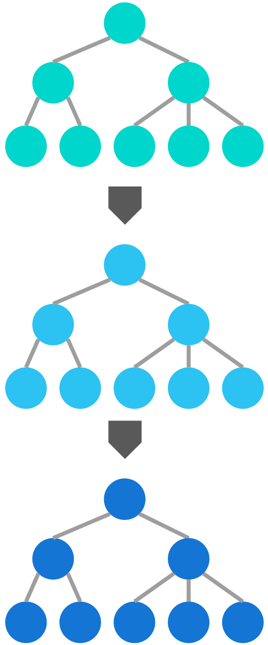
Working together, ensemble learners improve accuracy and robustness and decrease variance.

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

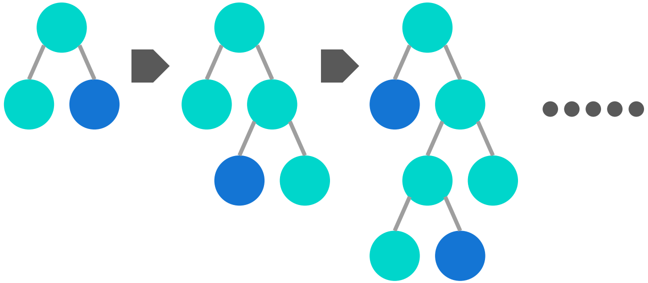


Introduction to Ensemble Learning

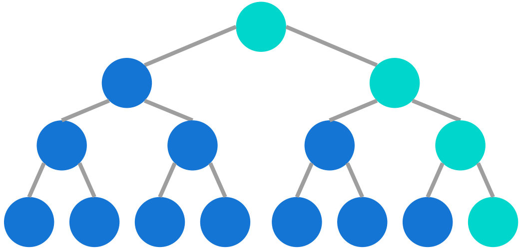
We have to combine weak learners by using a specific algorithm, such as:



Gradient Boosted Tree



XGBoost

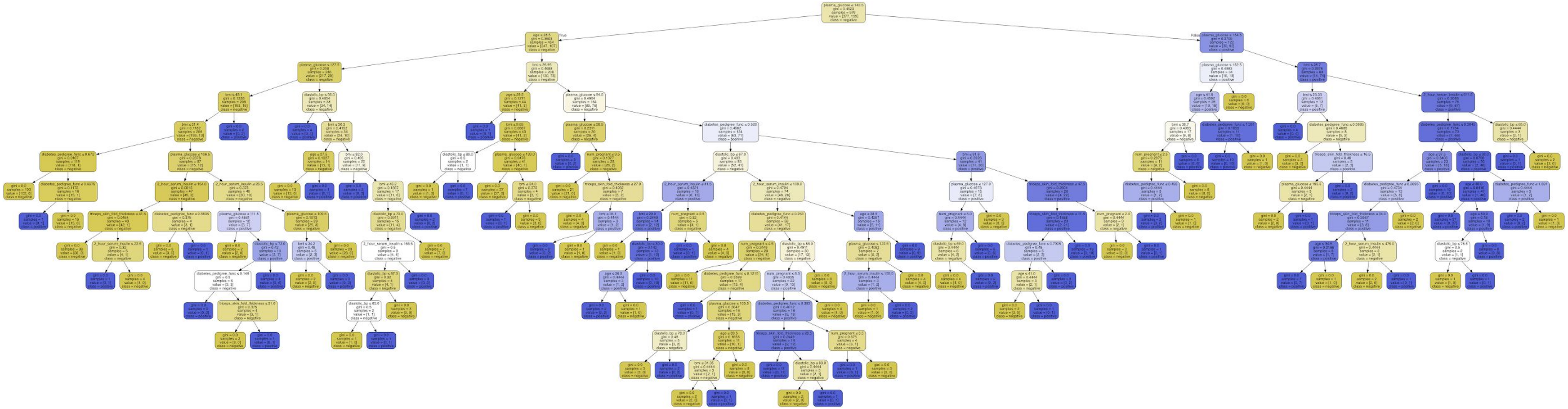


Random forests

Random Forest Algorithm

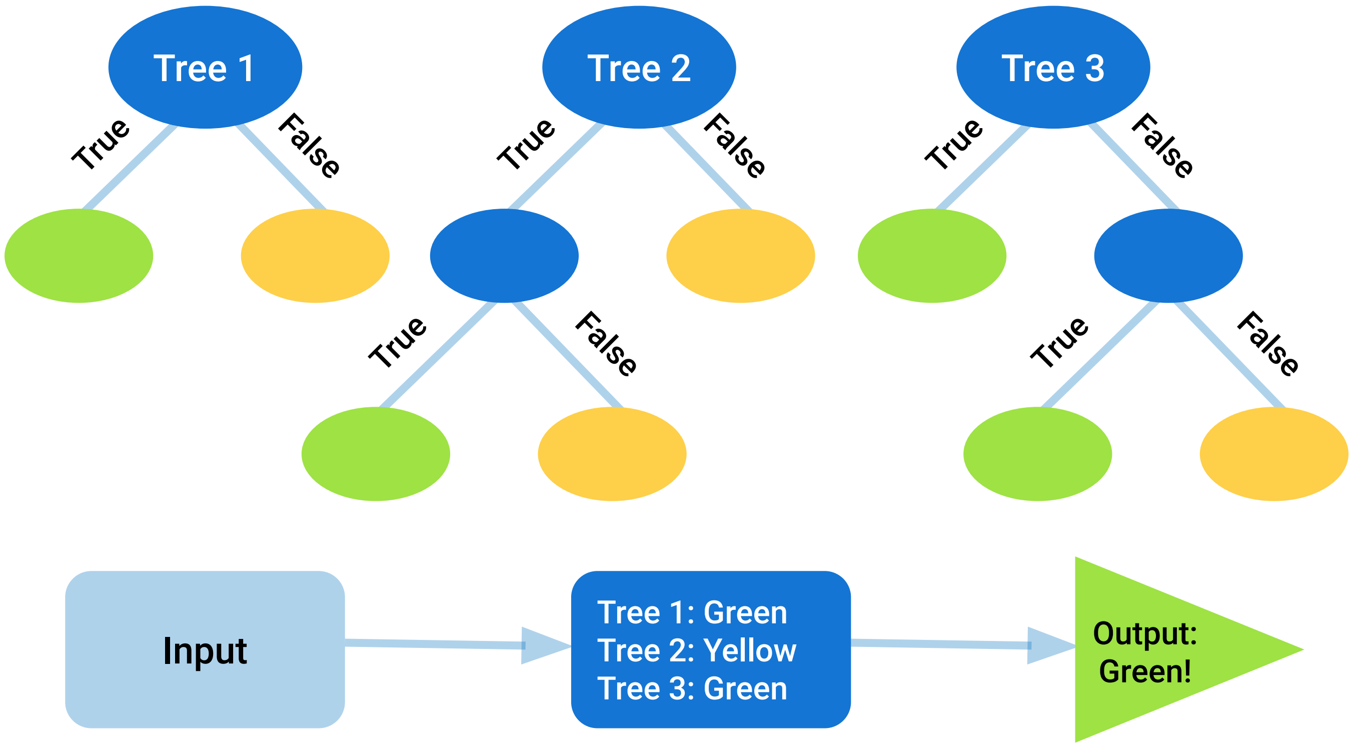
Random Forest Algorithm

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

- **Robust** against overfitting because each weak classifier is trained on different pieces of the data.

- **Robust** to outliers and non-linear data.

- **Runs** efficiently on large databases.


Questions?

Instructor Demonstration

Random Forest


Questions?

Activity: Predicting Fraud with Random Forests

In this activity, you will explore how to use the random forest algorithm for identifying fraudulent loan applications.

Suggested Time:

15 minutes

Time’s Up! Let’s Review.


Questions?

Instructor Demonstration

K-Nearest Neighbors

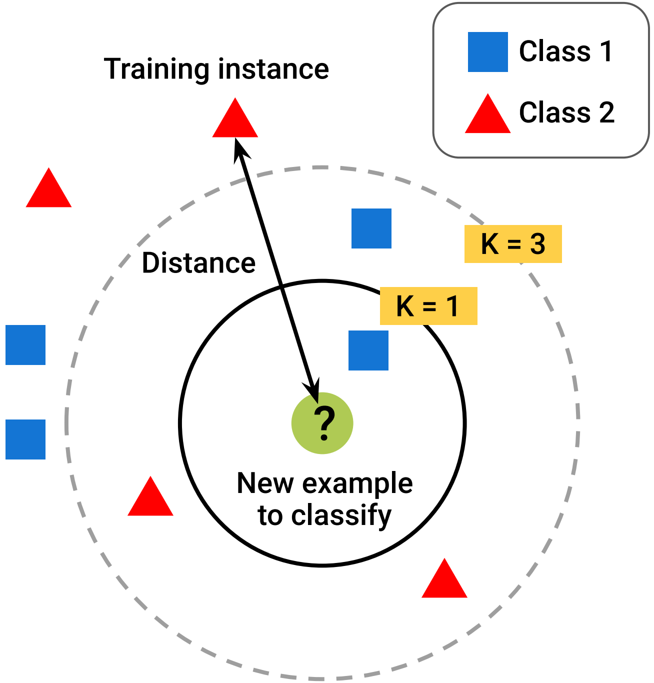
K-Nearest Neighbors

Earlier, we learned that the k-nearest neighbors algorithm is useful for unsupervised learning because it groups the dataset into different categories.

We also can use k-nearest neighbors for supervised learning (data with labeled outcomes).

k-nearest neighbors can categorize observations and choose class labels for data that are most similar to each other.

- We can choose to classify an unknown point by averaging the known label values around it.
- Closer points are weighted so that they contribute more to the average than distant points.




Questions?

Activity: K-Nearest Neighbors

In this activity, you will use data on a bank's telemarketing campaign to build a model to classify customers.

Suggested Time:

15 minutes

Time’s Up! Let’s Review.


Questions?

The End

69