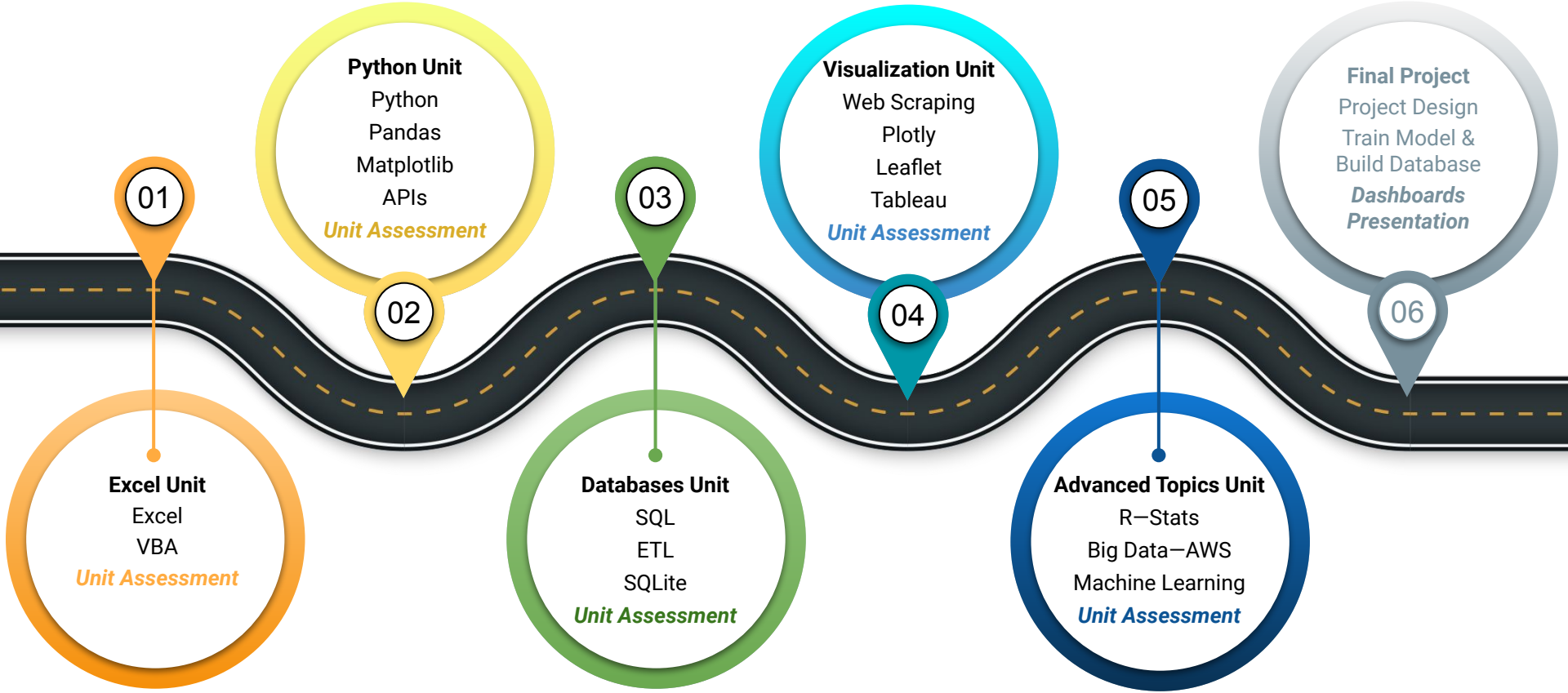
Confusion Matrices, Decision Trees, and Ensemble Learning

Data Boot Camp

Lesson 17.2

The Big Picture



This Week: Supervised Machine Learning

By the end of this week, you'll know how to:

- Explain how a machine learning algorithm is used in data analytics
- Create training and test groups from a given data set
- Implement the logistic regression, decision tree, and random forest algorithms and interpret their results.
- Compare the advantages and disadvantages of each supervised learning algorithm
- Determine which supervised learning algorithm is best used for a given dataset or scenario
- Use ensemble and resampling techniques to improve model performance

This Week's Challenge

Using the skills learned throughout the week, you will evaluate three machine learning models using different algorithms to determine which is better at predicting credit risk.

Today's Agenda

By completing today's activities, you'll learn the following skills:

01 Interpreting confusion matrices and accuracy measures

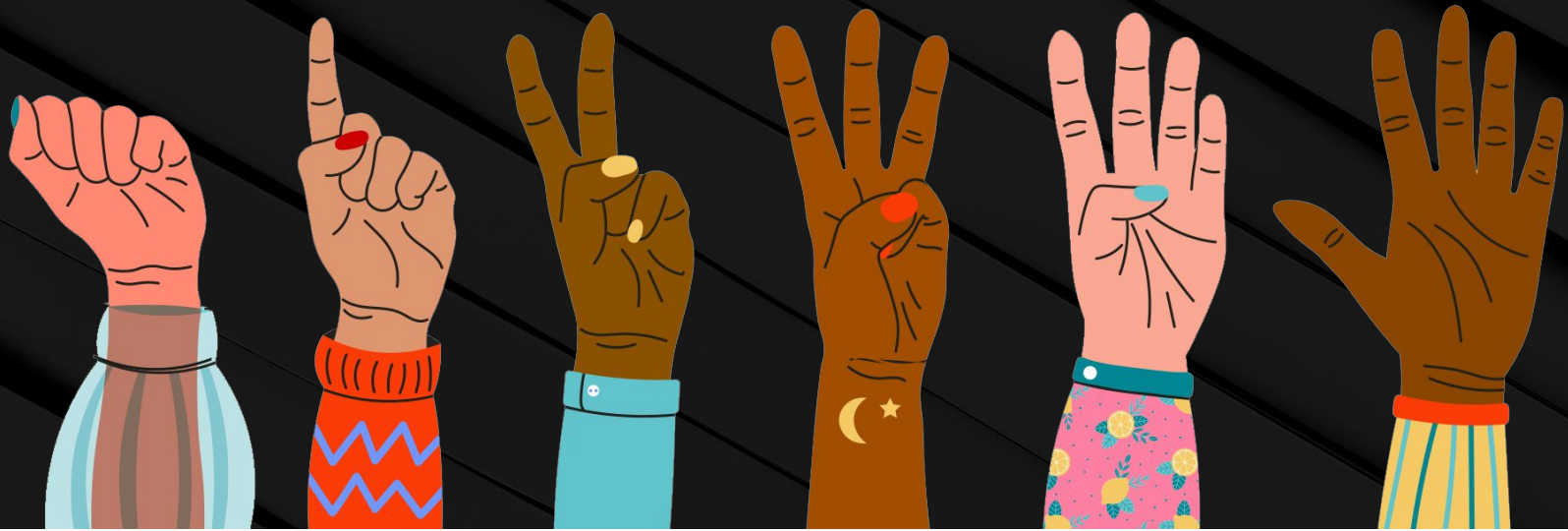
02 Implementing ensemble learning: bagging and boosting

03 Fine-tuning data using feature selection with RandomForestClassifier

Make sure you've downloaded any relevant class files!

FIST TO FIVE:

How comfortable do you feel with this topic?



Confusion Matrices

A **confusion matrix** compares the predicted values from a model against the actual values.

Although accuracy is important, it's not the only measure that we want to consider when evaluating the performance of a classification model.

Confusion Matrices

A confusion matrix compares the predicted values from a model against the actual values.

The entries in the confusion matrix are the number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).

	Predicted True	Predicted False
Actually True	128 (True Positives)	5 (False Negatives)
Actually False	6 (False Positives)	111 (True Negatives)

Confusion Matrices

We can calculate measures like the accuracy of a model from the values in the confusion matrix.

$$\text{Accuracy} = TP + TN \,/\, (TP + TN + FP + FN)$$

What steps do we need to take to validate the model?

Confusion Matrices

Steps we need to take to validate the model:

01 Split the data into training and testing data.

02 Create the logistic regression model.

03 Fit or train the model with our training data.

04 Validate the model using the testing data.

Instructor Demonstration

Confusion Matrices

Accuracy

Accuracy is the percentage of correct predictions.

$$\text{Accuracy} = TP + TN \,/\, (TP + TN + FP + FN)$$

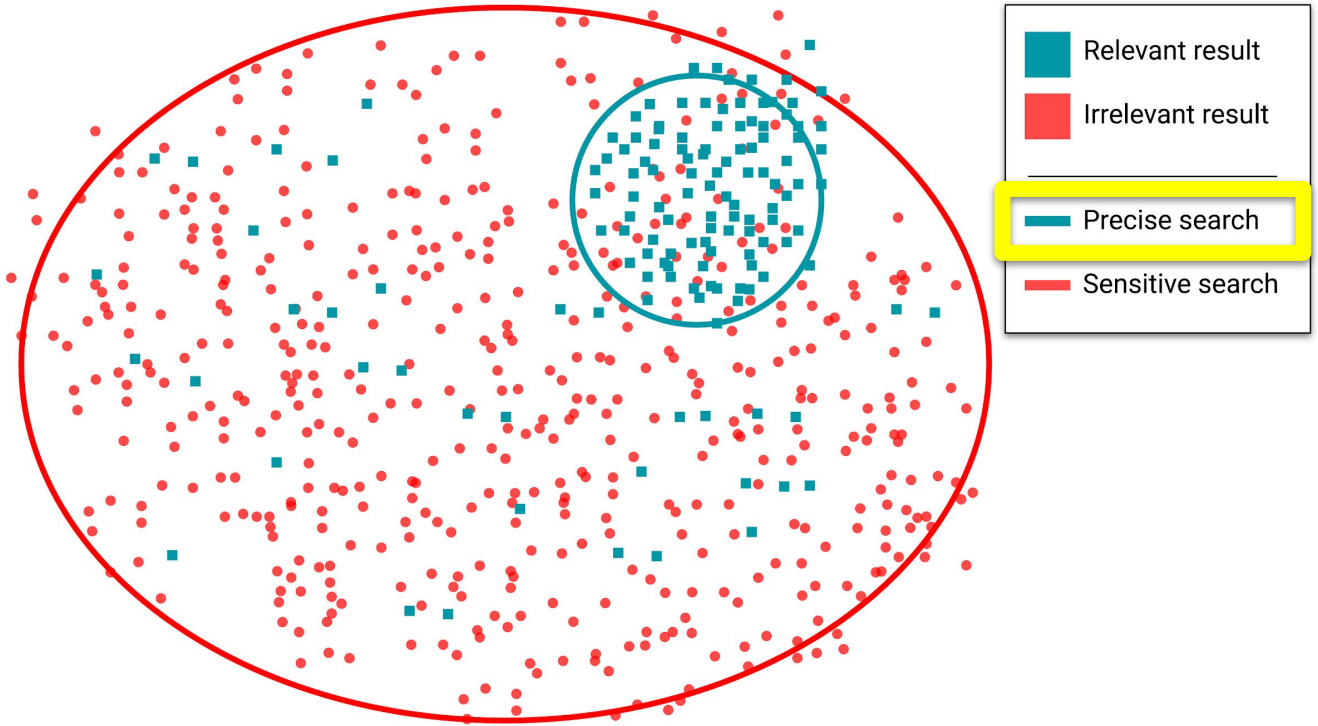
Precision

Precision is the percentage of positive predictions that are correct.

$$\text{Precision} = TP / (TP + FP)$$

Precision

Precision is a measure of how reliable a positive classification is.



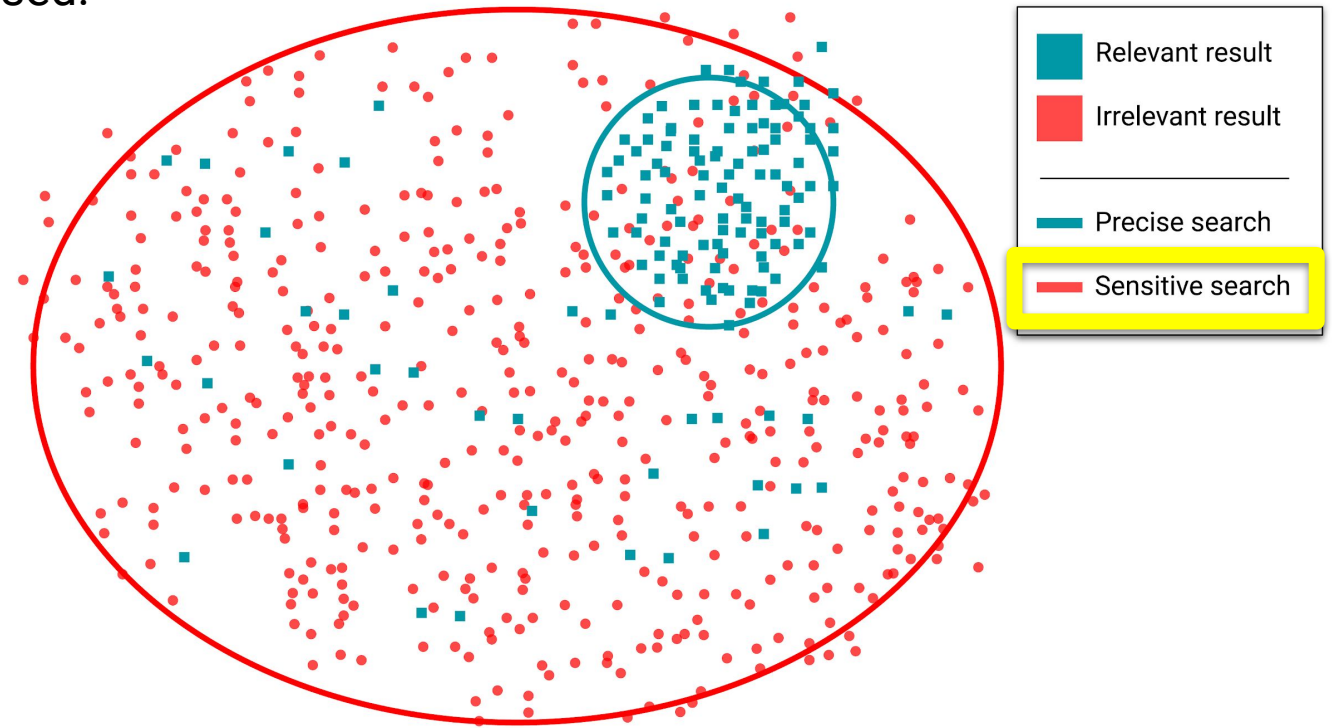
Sensitivity/Recall

Sensitivity (or Recall) is the percentage of actual positive results that are predicted correctly. Sometimes it is called the True Positive Rate.

$$\text{Sensitivity} = TP / (TP + FN)$$

Sensitivity

Sensitivity is a measure of how many observations with a positive condition will be correctly diagnosed.



Can you think of situations where precision would be more important? What about when sensitivity is more important?

Sometimes precision is better, and sometimes sensitivity is better.

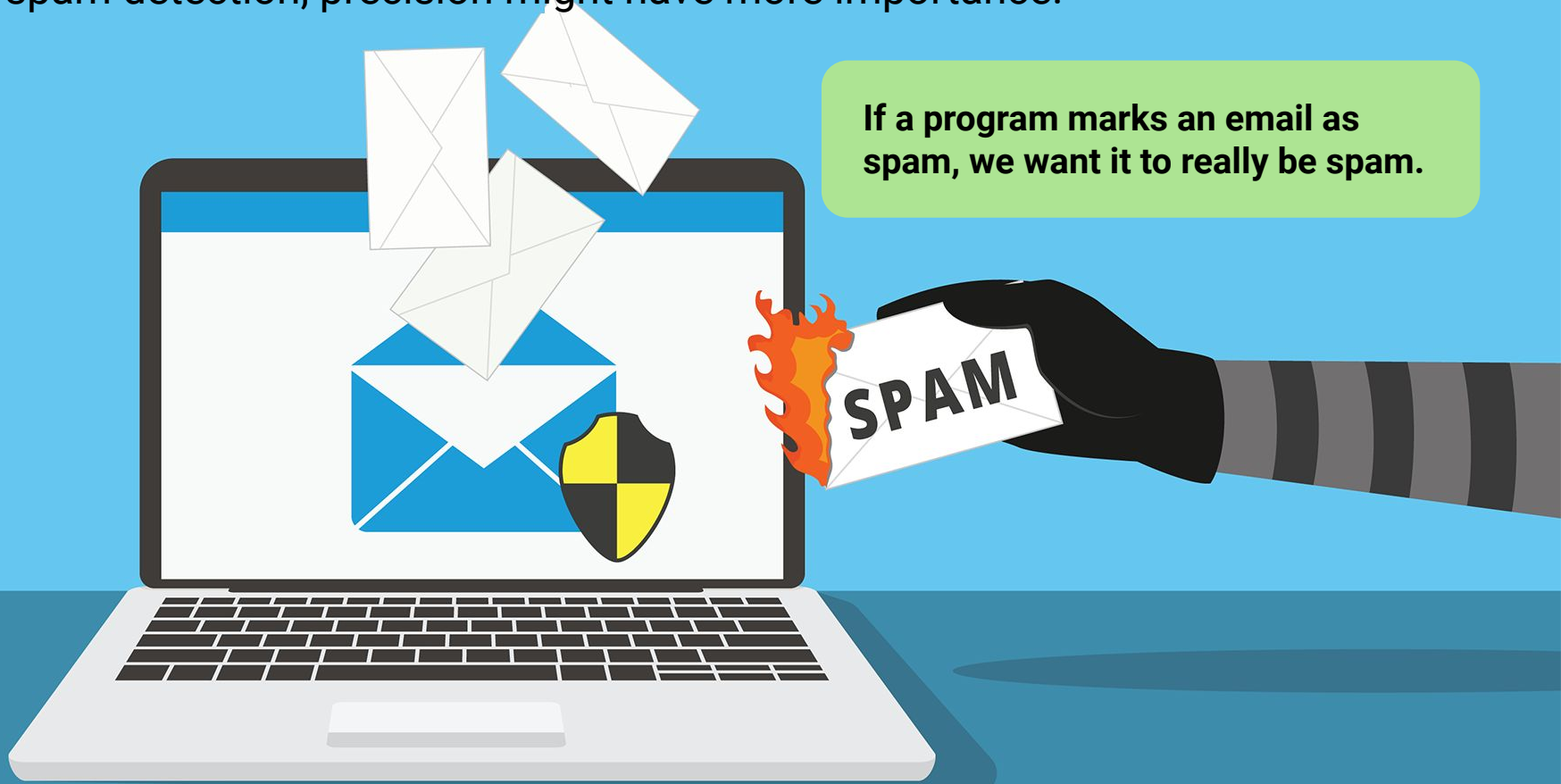
Precision or Sensitivity

For a cancer screening test, more sensitivity is probably more important than more precision.

The idea is that more sensitivity increases the chances of finding true positives, and people can then follow up with a more precise test.

Precision or Sensitivity

For spam detection, precision might have more importance.



F1 Score

The F1 score (also called the harmonic mean) balances precision and sensitivity.

$$F1 = 2(Precision * Sensitivity) / (Precision + Sensitivity)$$


RECAP

Confusion Matrix Measure Formulas

$$\text{Accuracy} = TP + TN \,/\, (TP + TN + FP + FN)$$

$$\text{Precision} = TP \,/\, (TP + FP)$$

$$\text{Sensitivity} = TP \,/\, (TP + FN)$$

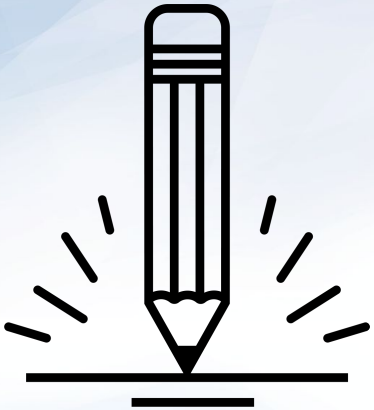
$$F1 = 2(\text{Precision} * \text{Sensitivity}) \,/\, (\text{Precision} + \text{Sensitivity})$$

Classification Report

Because the classification report doesn't know which class is positive and which is negative, it gives the scores for both classes—"0" and "1".

```
print(classification_report(y_true, y_pred))
```

	precision	recall	f1-score	support
0	0.96	0.96	0.96	133
1	0.96	0.95	0.95	117
accuracy			0.96	250
macro avg	0.96	0.96	0.96	250
weighted avg	0.96	0.96	0.96	250



Activity: Create a Confusion Matrix

In this activity, you'll create a logistic regression model to predict the onset of diabetes within five years for at-risk patients in the 1988 Pima Diabetes dataset. Then, you'll interpret the confusion matrix that the model produces.

Suggested Time:
15 minutes

Activity: Create a Confusion Matrix

Instructions:

01 Load the Pima Diabetes dataset, and then split the data into training and testing sets.

02 Fit a logistic regression model to the training set.

03 Create a confusion matrix of the predicted vs. the actual outcomes for the testing set.

04 Manually calculate the precision, sensitivity, and F1 score of the model by using the values from the confusion matrix.

05 Create the classification report, then compare it to your manually calculated scores.

06 Answer the following questions:
Is your model more precise or more sensitive? Which measure is more important for this model?

Let's Review

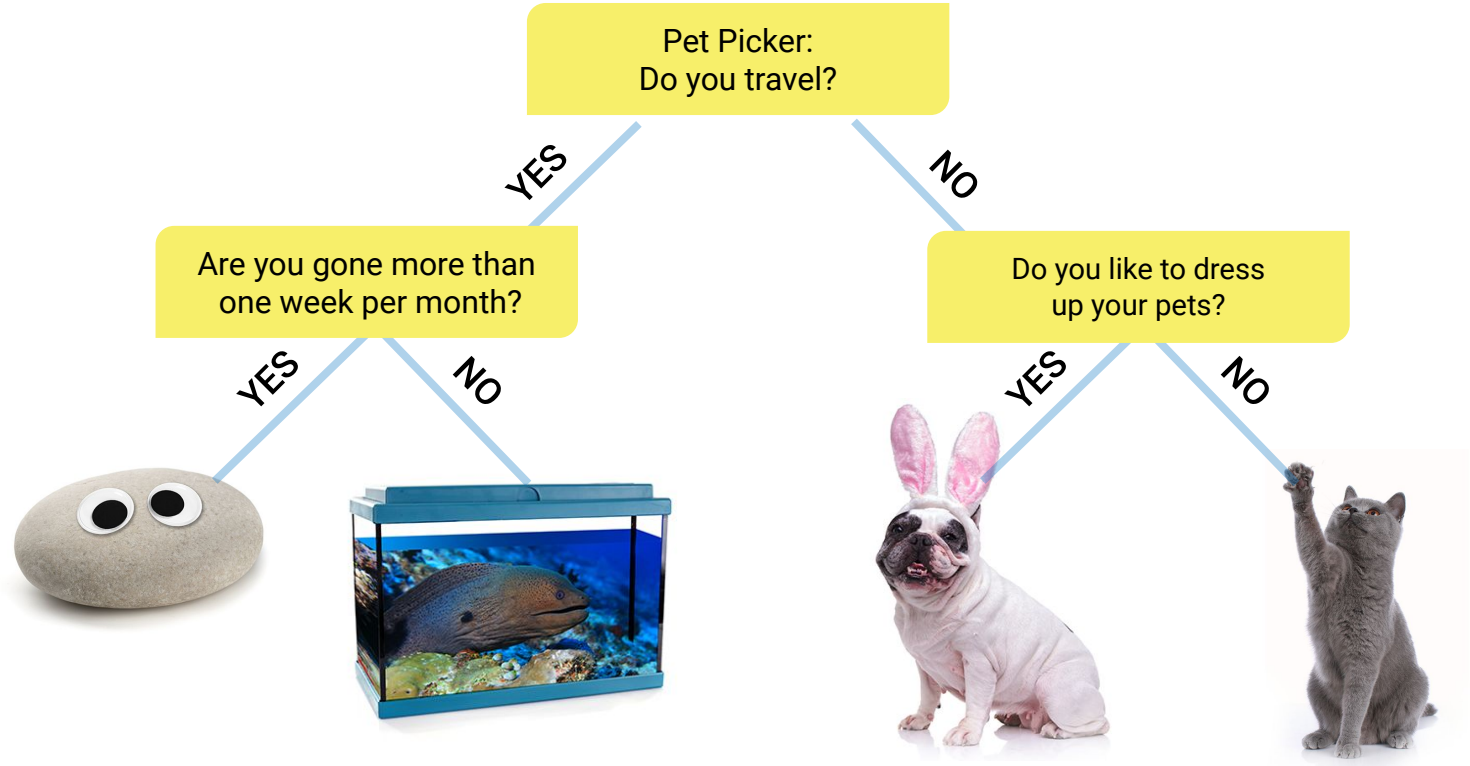
Questions?

Decision Trees

A **decision tree** encodes a series of true-or-false questions that we can interpret as `if-else` statements.

Decision Trees

Decision trees encode a series of true/false questions.



Decision Trees

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?

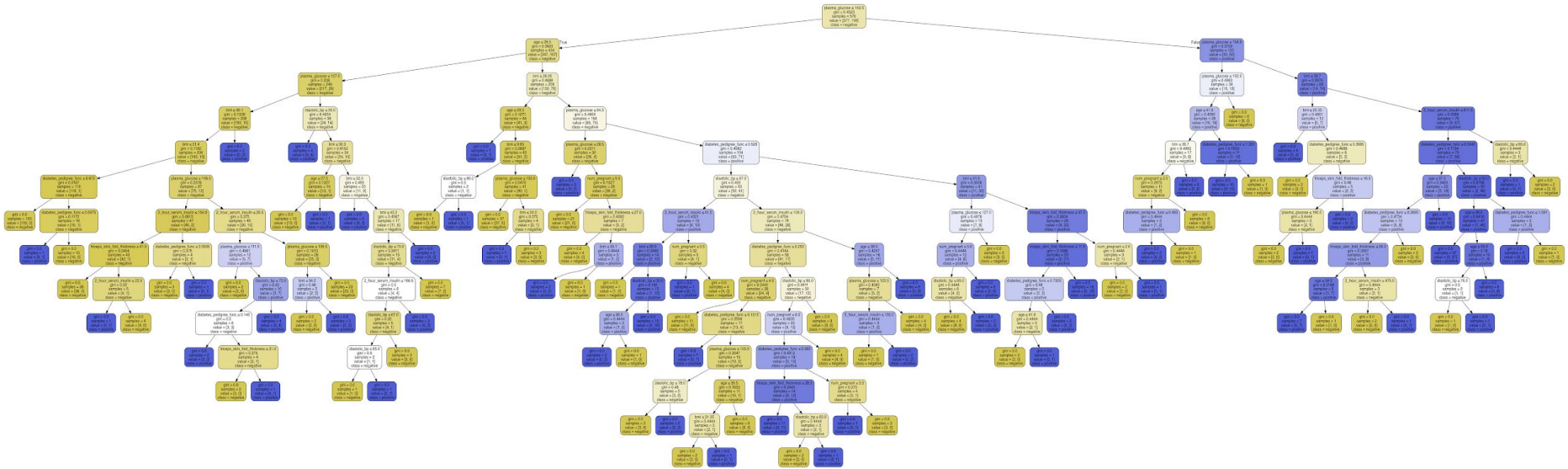
Yes Dress Up: Pet Dog

No Dress Up: Pet Cat

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

Decision Tree Complexity

Decision trees can become deep and complex depending on the number of questions that have to be answered. Deep and complex trees tend to overfit to the data and don't generalize well.

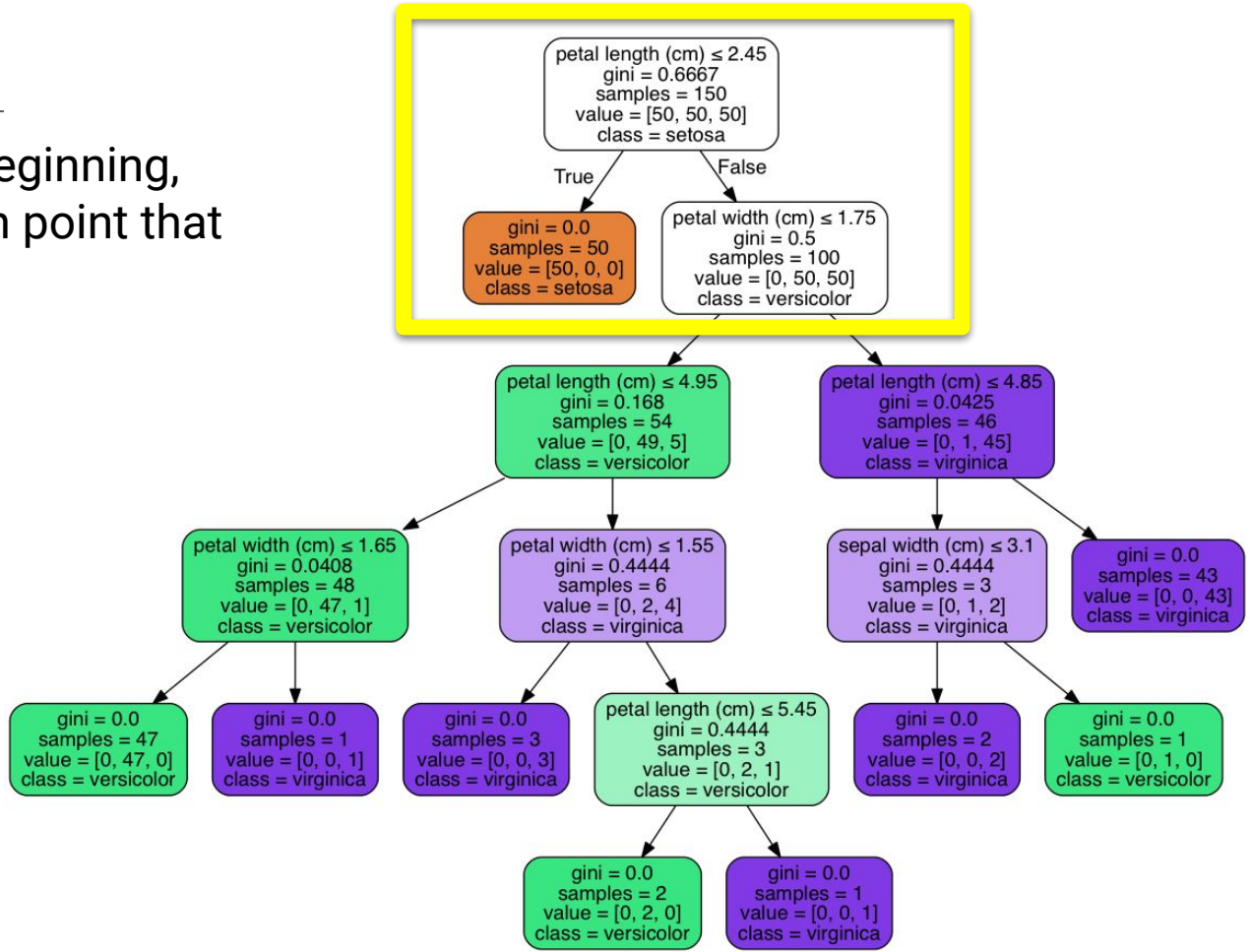


Instructor Demonstration

Decision Trees

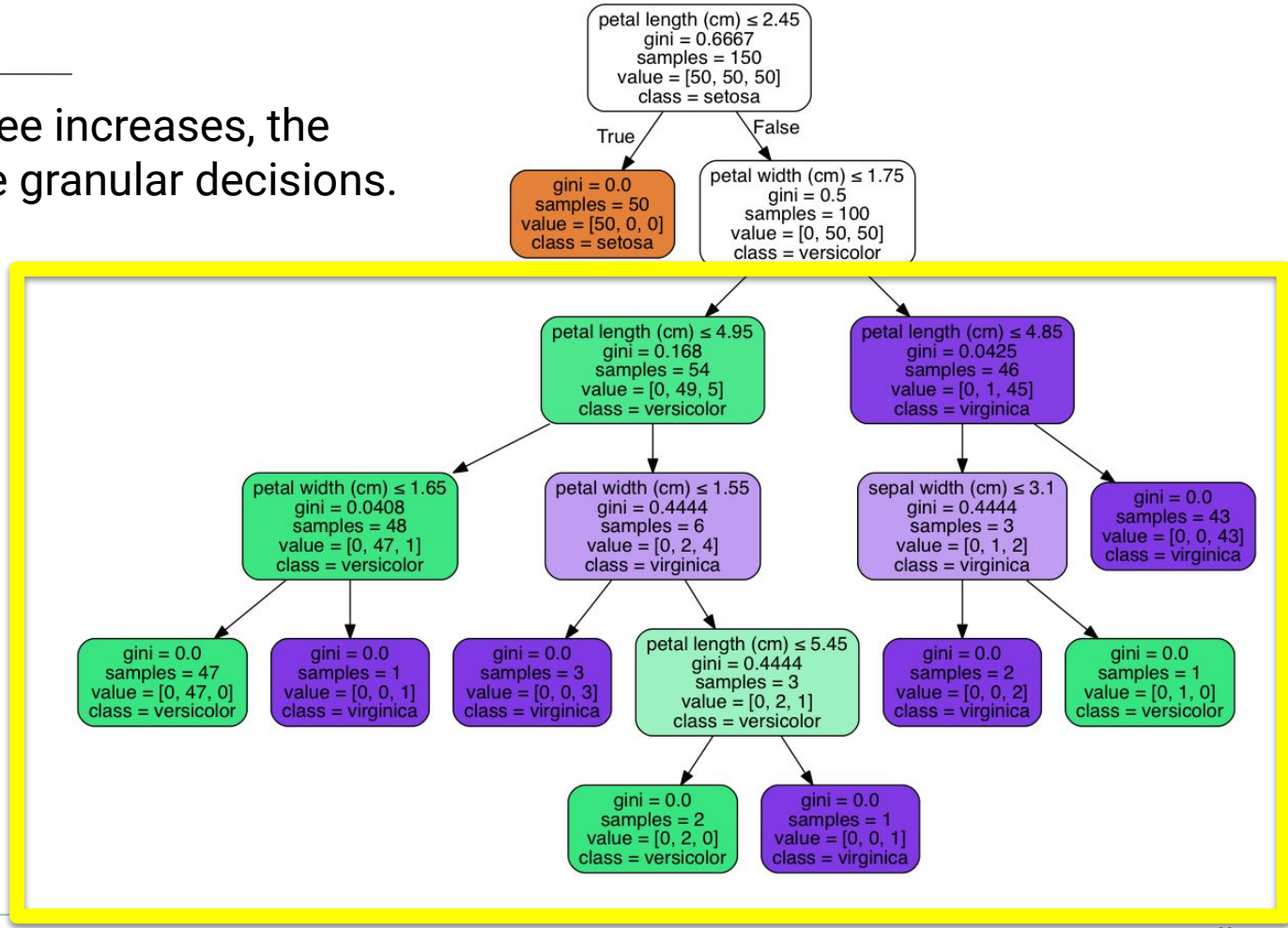
Decision Trees

The node at the top, or beginning, of the tree is the decision point that makes the biggest split.



Decision Trees

As the depth of the tree increases, the subnodes make more granular decisions.



The training phase of the decision tree algorithm learns which features best split the data.

Ensemble Methods

Anomalies in the training dataset can trick these trees. Aggregating is a technique that we can use to make decision trees more useful.

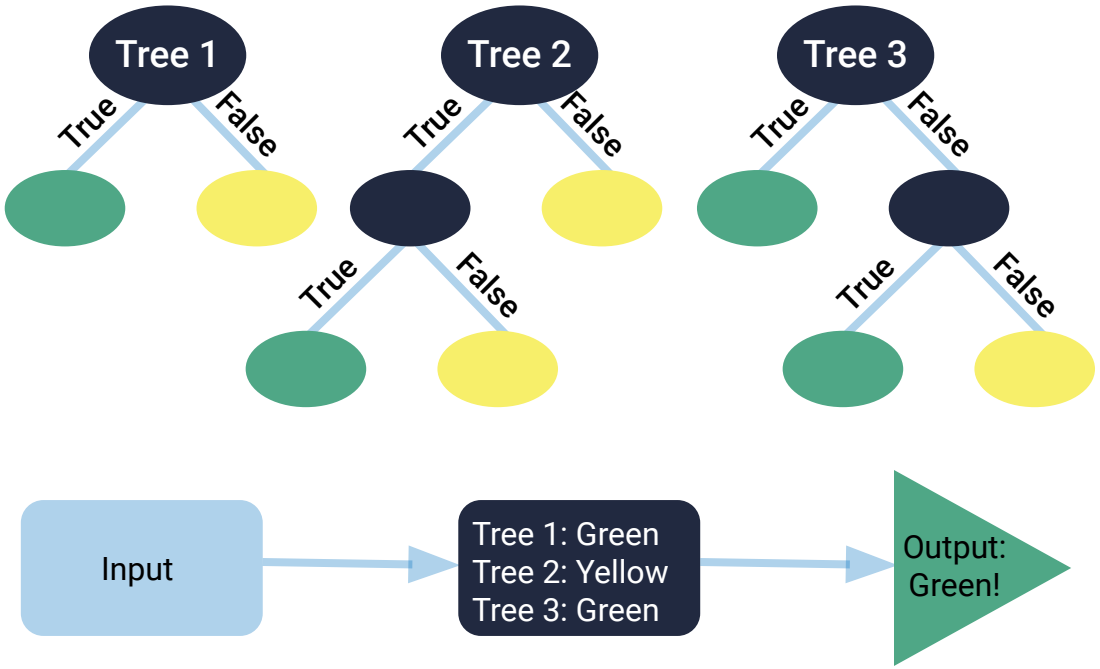
Aggregating makes lots of simple algorithms and then takes the consensus.

Random Forests

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

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

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



Instructor Demonstration

Ensemble Methods

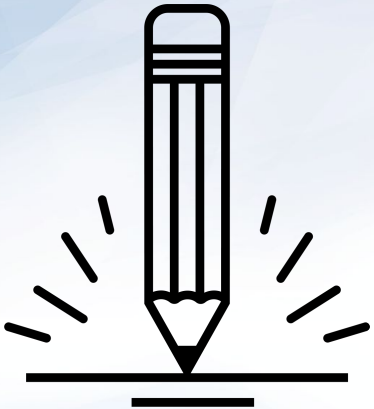
Questions?

Time to Code

Bag and Boost

Suggested Time:

15 minutes



Activity: Bag and Boost

In this activity, you'll apply three aggregate classifiers to predict diabetes onset from the Pima Diabetes dataset.

Suggested Time:
15 minutes

Activity: Bag and Boost

Instructions:

01 Import a random forest classifier, and then fit the model to the data.

02 Import an extremely randomized trees classifier, and then fit the model to the data.

03 Import an Adaptive Boosting classifier, and then fit the model to the data.

04 Calculate the classification report for each model. Also, calculate the score for both the training and the testing sets. Compare the performance of the three models.

05 Refactor to reduce repetitive code. Create a function that takes in a model and a dataset and prints a classification report to compare different models.

06 Choose one of the models, and then read the scikit-learn documentation. Use your newly created function to try different parameters. Can you improve the model?

Feature Selection with Random Forest

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

Feature Selection

There are many reasons to perform feature selection:

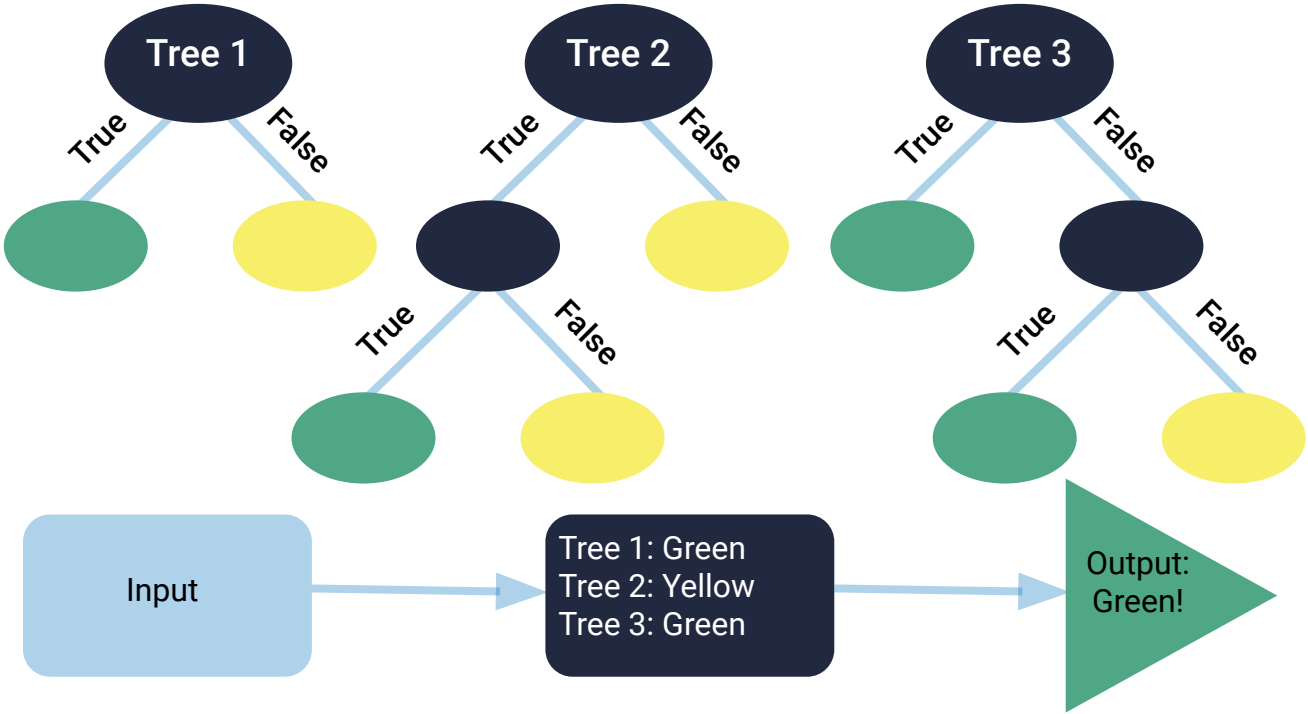
Simplified models are less likely to overfit.

Simplified models reduce training time.

Simplified models are easier to interpret, etc.

Random Forests

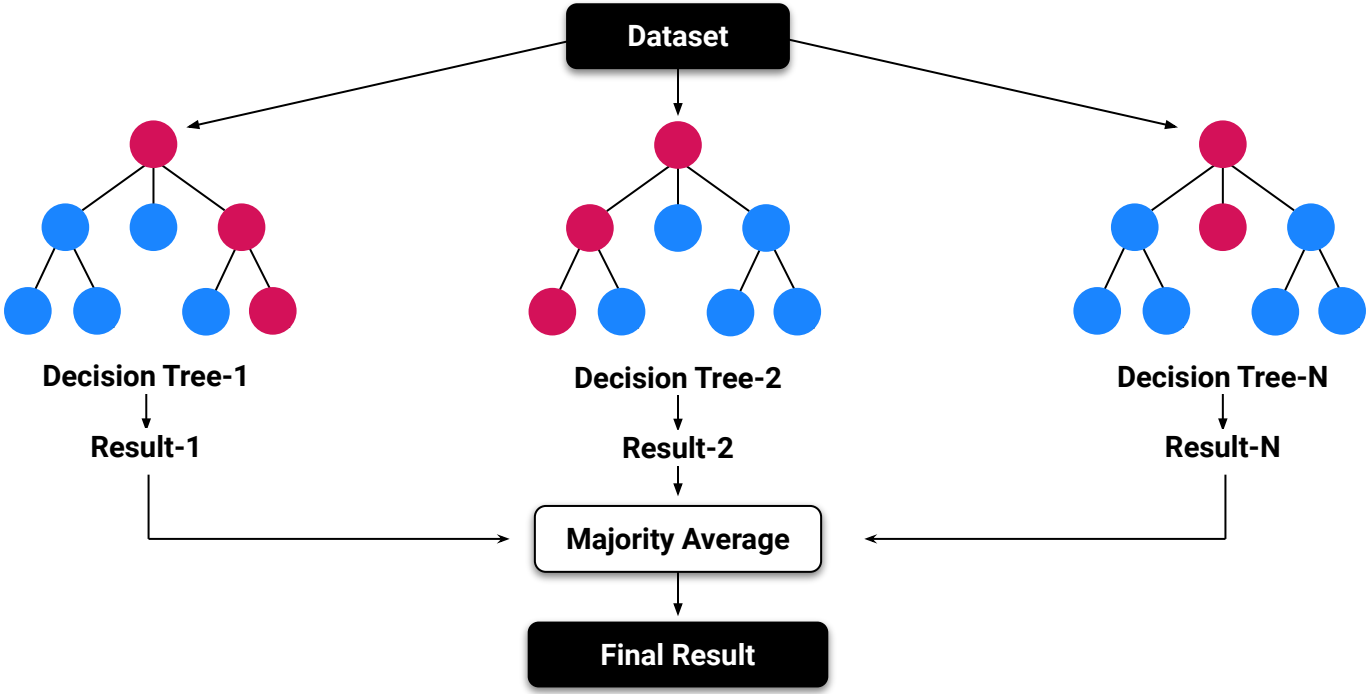
There are many ways to perform feature selection. One technique uses the information from a random forest model.



Random forests use decision trees that try to select the best feature at every split.

Random Forests

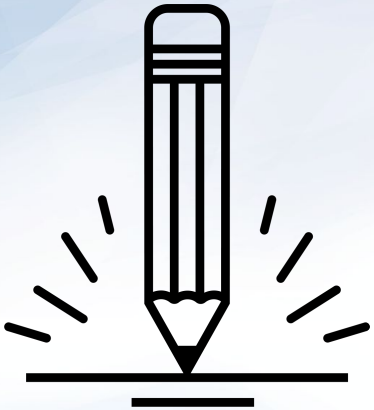
Therefore, how often a feature gets selected over the whole random forest model gives us an indication of how important that feature is.



Instructor Demonstration

RandomForest Feature Selection

Questions?



Activity: Finding the Features from the Trees

In this activity, you'll use a random forest model to find the most important features for predicting arrhythmia in heartbeats.

Suggested Time:
15 minutes

Activity: Finding the Features from the Trees

Instructions:

01 Import the arrhythmia data, and then fit a random forest model to the scaled and split data.

02 Import `SelectModel` to extract the best features from the random forest model.

03 Fit a logistic regression to the original dataset, and then print its score.

04 Fit a logistic regression to the selected dataset, and then print its score.

05 Compare the scores of the two logistic regression models.

Let's Review

Questions?