Decision Tree Classification

Decision Trees

- Intuitive
- Easy to interpret
- Can perform both classification and regression

Limitations

- Decision trees tend to have decision boundaries perpendicular to an axis
- This is also known as being orthogonal
- This makes decision trees sensitive to training set rotation
- Decision trees are also sensitive to small variations in the training data

Decision Tree Classifier

- Part of scikit-learn’s tree library
- Visualize the tree by using export-graphviz to save the tree out to a graph definition file (it has a dot extension)
- The graph-viz command line package will then allow you to convert data files to other graphical formats

Example

Let’s load some data

```
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier

iris = load_iris()
X = iris.data[:, 2:] # petal length and width
y = iris.target
```

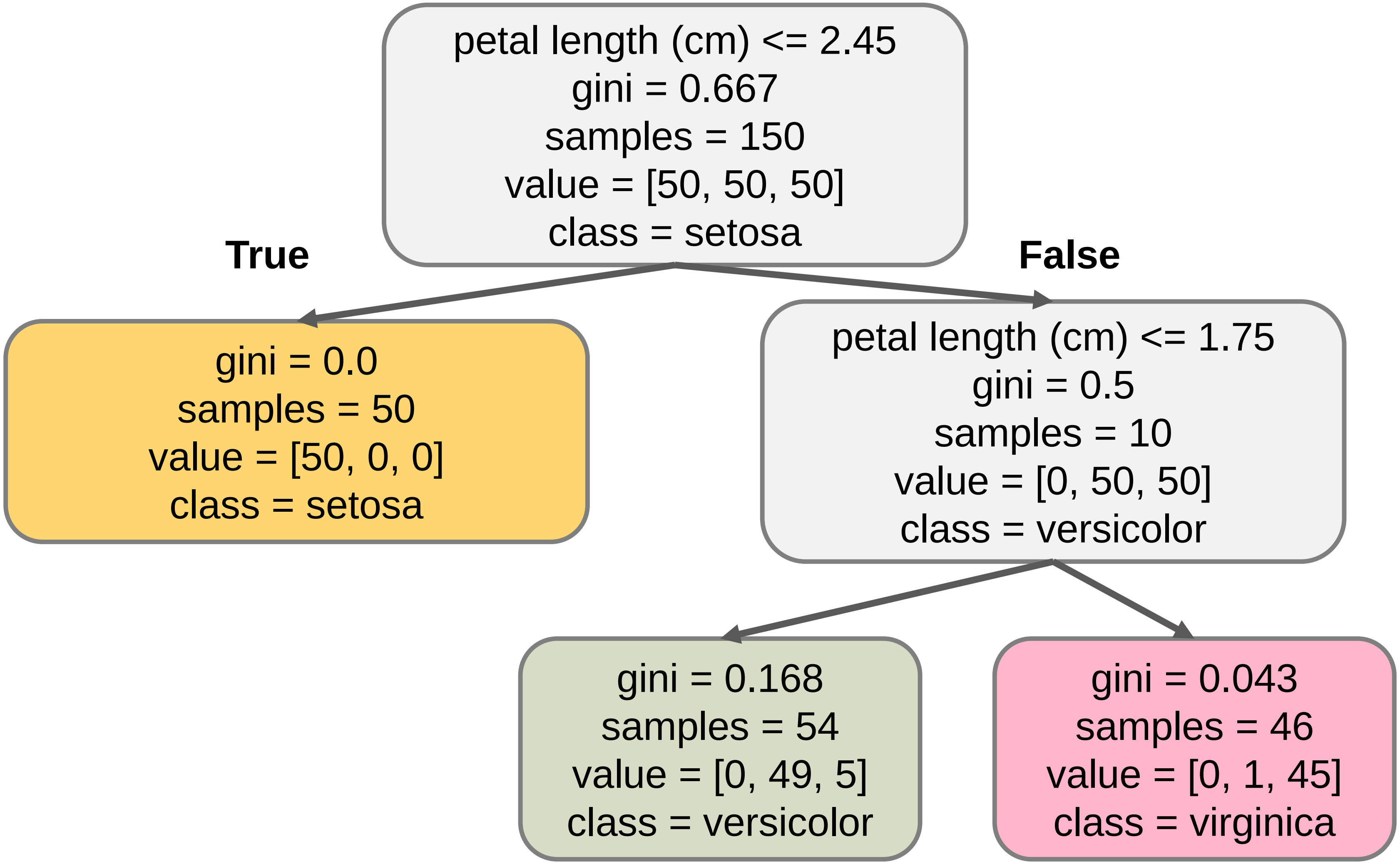
Let’s train a classifier

```
tree_clf = DecisionTreeClassifier(max_depth=2, random_state=42)
tree_clf.fit(X, y)
```

The Output Is a New Model

```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=2,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=42,
            splitter='best')
```

Let's Look at the Tree



Classification Training

The End

Making Predictions

Making Predictions

- Start at the root node
- Follow the set of questions each node has
- Eventually you will reach a leaf node (a node that doesn’t have children)
- The leaf node you arrive at will show the predicted class as part of the output for that node
- You can also predict class probabilities

Code for Class Prediction

- Here is the code to get back a prediction and class probabilities

```
tree_clf.predict_proba([[5, 1.5]])
```

```
array([[0.        , 0.90740741, 0.09259259]])
```

```
tree_clf.predict([[5, 1.5]])
```

```
array([1])
```

Node Attributes

- Each node in a tree has attributes that describe the data pertinent to the state of the node
- A node’s sample attribute counts how many instances it applies to
- A node’s value attribute tells you how many instances of each class it applies to

Node Labeling

- The way a leaf node gets labeled is by taking a ratio of values per class and samples per node
- The value with the highest probability becomes the predicted class for that node

Gini

- The gini of a particular node measures its impurity
- A gini of 0 means that all the training instances in that node belong to the same class
- It is the sum of the ratios of each value divided by its sample squared subtracted from 1

Implementation

- Scikit-learn uses the class and regression Tree (CART) algorithm for training decision trees
- It produces binary trees, where each non-leaf node has two children
- There are other algorithms (ID3, for example) that can have more than two children per node

Making Predictions

The End

CART Training

Training

- CART splits the training set into two subsets using a single feature k and a threshold
- It does this by looking at the value pair that produces the purest subsets weighted by size
- Loss function:

$$G(Q, \theta) = \frac{n_{left}}{N_m} H(Q_{left}(\theta)) + \frac{n_{right}}{N_m} H(Q_{right}(\theta))$$

A Growing Tree

- CART is called a "growing tree"
- It starts at the base of the tree
- It iterates (grows the tree) at every branch until
    1. It reaches the max depth
    2. It can't get impurity to go down anymore

Entropy

- Entropy is another type of impurity measure
- Entropy is zero when it contains instances of only one class
- It is the negative sum of each of the class values/samples times the $\log_2$ of the values/samples

Regularization

- Decision trees tend to overfit the training data
- Since the structure of the tree is not known before the tree is built it is said to be non-parametric
- Compare this to a linear model, which has limited degrees of freedom
- The linear model is parametric
- Parametric models tend to cost less when fitting the data

Combatting Overfitting

- Specifying max-depth
- Specifying min-samples-split
  - Minimum number of samples a node must have before it is split
- Specifying Min_weight_fraction_leaf
  - A function of the total of weighted instances
- Specifying max_leaf_nodes
- Specifying max_features

CART Training

The End

Regression

Decision Tree Regression

- You can use the DecisionTreeRegressor class to perform regression
- Leaf nodes will specify a value instead of a class at each leaf node
- The CART algorithm splits each region in a way that makes most training instances as close as possible to the predicted values

Loss Function

- In decision tree classification we minimize impurity
- In regression problems we minimize MSE

$$H(X_m) = \frac{1}{N_m} \sum_{i \in N_m} (y_i - \bar{y}_m)^2$$

Regression Overfitting

- Decision trees can overfit in regression problems just like with classification
- Use max_depth to regularize decision tree regression problems

Regression

The End