Decision Trees

Some images and notes were retrieved from Aurélien Géron - Hands-on Machine Learning with Scikit-Learn and TensorFlow (2019, O'Reilly)

Decision Trees

- Decision Trees are both used for regression and classification tasks.

- They also compose the basis of Random Forests which is a powerful ensemble learning method.

- In this slide set, we will cover
  - how decision trees work
  - how to visualize them
  - how to make predictions and see class probabilities
  - how to regularize them
  - how to use them for regression
  - limitations of decision trees

Training Decision Trees on Sklearn

```
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


tree_clf = DecisionTreeClassifier(max_depth=2)

tree_clf.fit(X, y)
```

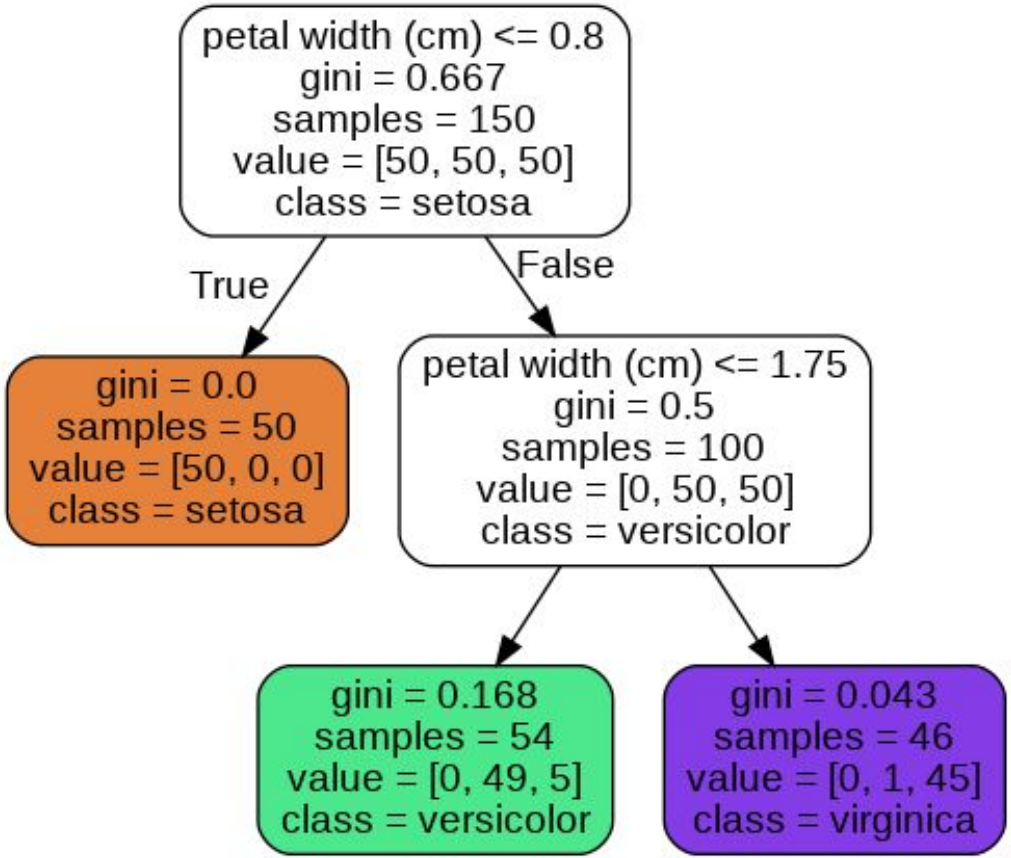
Training Decision Trees on Sklearn

```
from sklearn.tree import export_graphviz

export_graphviz(
        tree_clf,
        out_file="iris_tree.dot",
        feature_names=iris.feature_names[2:],
        class_names=iris.target_names,
        rounded=True,
        filled=True
    )
```

```
$ dot -Tpng iris_tree.dot -o iris_tree.png
```

Decision Tree Model



More on Decision Trees

- In order to make a prediction, we just need to start from the root and answer all the questions until we reach a leaf node.

- Before the training, they do not require scaling.

- **gini** attribute given in each node represents the impurity of the node.
  - If all instances in the node belong to the same class → then, gini = 0
  - Here is the formula for gini impurity

$$G_i = 1 - \sum_{k=1}^{n} {p_{i,k}}^2$$

- For instance, the depth2 left node had the gini value 0.168, since

```
= 1 - (0/54)**2 - (49/54)**2 - (5/54)**2
```

More on Decision Trees

- Sklearn uses CART algorithm which always generates a binary tree. It means that every non leaf node has exactly 2 children.

- There are some other algorithms (like ID3) that composes a node with more than 2 children.

- Decision Trees are interpretable. You may explain the reason of a prediction (this is not the case for Random Forests and ANNs).

- In sklearn, use predict_proba to calculate class probabilities.

```
1 sample1 = [1.6, 0.9]
2 tree_clf.predict_proba([sample1])
```

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

How does CART work?

- It tries find the pair <k, t_k> where k is the feature and t_k is the threshold, to minimize the following cost function

$$J(k, t_k) = \frac{m_{\text{left}}}{m} G_{\text{left}} + \frac{m_{\text{right}}}{m} G_{\text{right}}$$

where $\begin{cases} G_{\text{left/right}} \text{ measures the impurity of the left/right subset,} \\ m_{\text{left/right}} \text{ is the number of instances in the left/right subset.} \end{cases}$

- After it splits the dataset into two different nodes, it applies the same logic on both subsets in a recursive way.

- This iteration continues until
  - a node with gini as 0 (all the sample in the node belongs to same class)
  - a control parameter (like `max_depth`) breaks the process

- It's a greedy algorithm which may not find the optimal tree (but the resulting tree will be still satisfactory).

Gini or Entropy?

- Entropy concept derived from thermodynamics to measure of molecular disorder. Entropy is also used in information theory as well.

- In our case, it states the impurity of a node (ie. entropy is 0 when a node contains single class instances).

$$H_i = -\sum_{\substack{k=1 \\ p_{i,k} \neq 0}}^{n} p_{i,k} \log_2 \left(p_{i,k}\right)$$

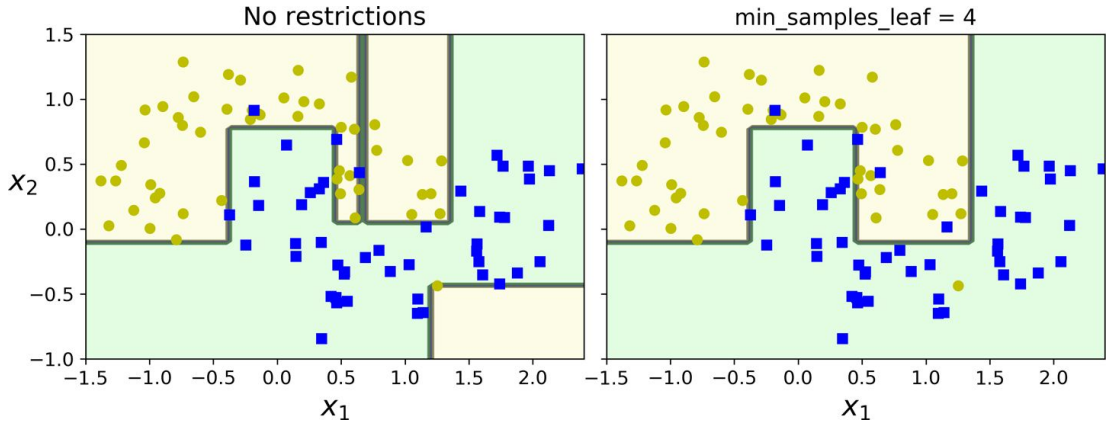
- For depth 2 left node, the entropy is
`-1*(49/54)*np.log2(49/54) + -1*(5/54)*np.log2(5/54) = 0.445`

- Using entropy or gini will not impact that much.

Regularization

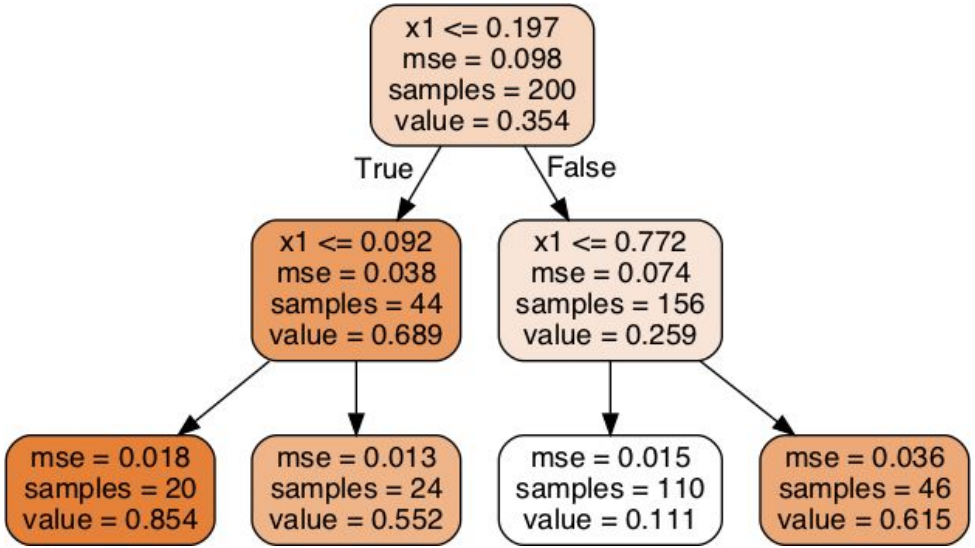
- Decision Trees easily tend to overfit the training data.

- We can tweak with the following parameters:
  - max_depth: maximum depth of the tree
  - min_samples_split: the minimum number of samples a node must have before it can be split
  - min_samples_leaf: the minimum number of samples a leaf node must have



Regression with Decision Trees

- It’s also possible to use Decision Trees for regression. See the following tree



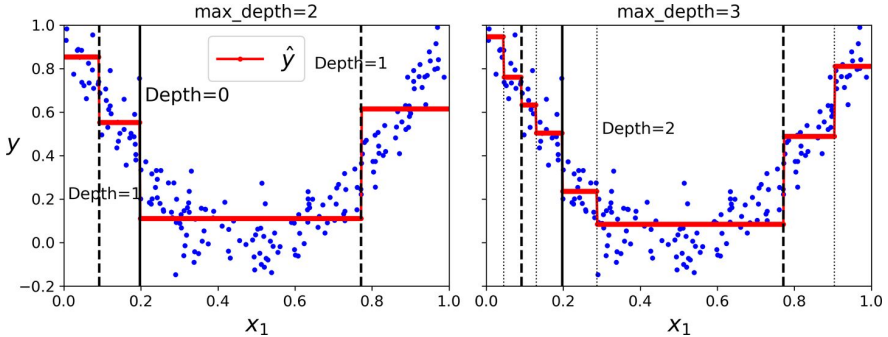
- It now contains **mse** score instead of gini score in each node. This score is calculated with the prediction value against the real values in this node. And the prediction value is the average of all real labels given in this node.

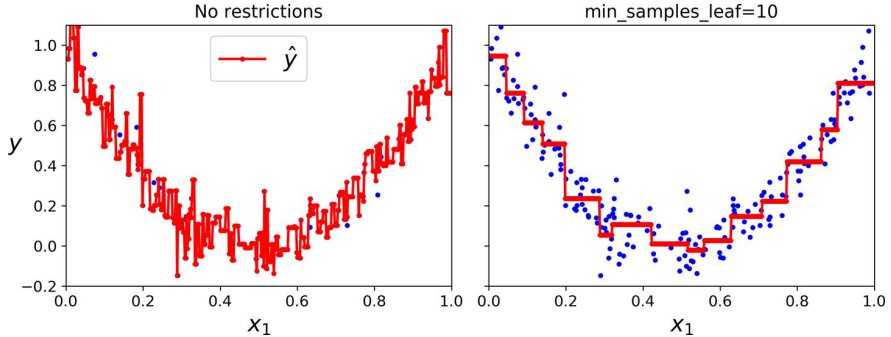
Regression with Decision Trees

- The CART algorithm now tries to minimize the following formula during training:

$$J(k, t_k) = \frac{m_{\text{left}}}{m}\text{MSE}_{\text{left}} + \frac{m_{\text{right}}}{m}\text{MSE}_{\text{right}} \quad \text{where} \quad \begin{cases} \text{MSE}_{\text{node}} = \sum_{i \in \text{node}} \left(\hat{y}_{\text{node}} - y^{(i)}\right)^2 \\ \hat{y}_{\text{node}} = \dfrac{1}{m_{\text{node}}} \sum_{i \in \text{node}} y^{(i)} \end{cases}$$

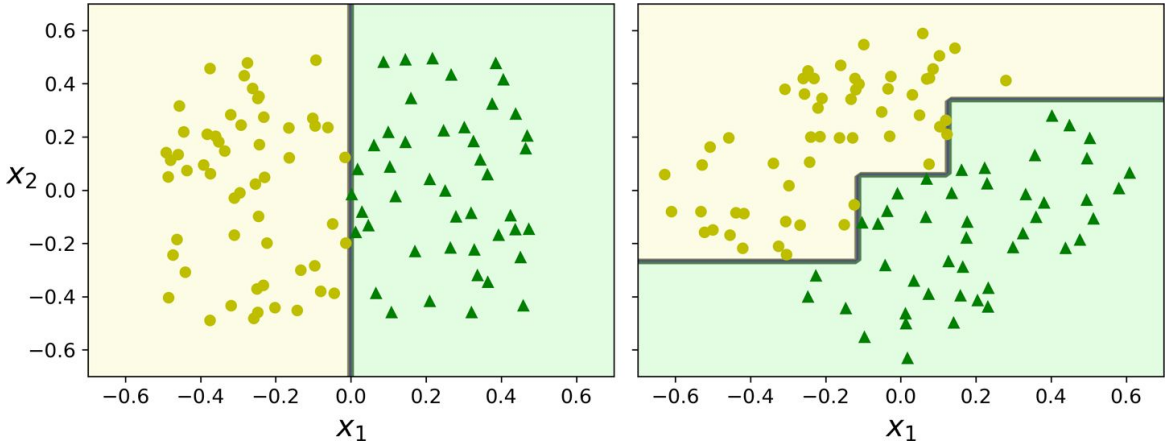
- We can still regularize the tree in regression





Limitations of Decision Trees

- Decision Trees love orthogonal decision boundaries. Therefore, they may not be able to generalize a model perfectly.



- The model on the right has some problems with generalization (PCA can be effective to solve this problem).