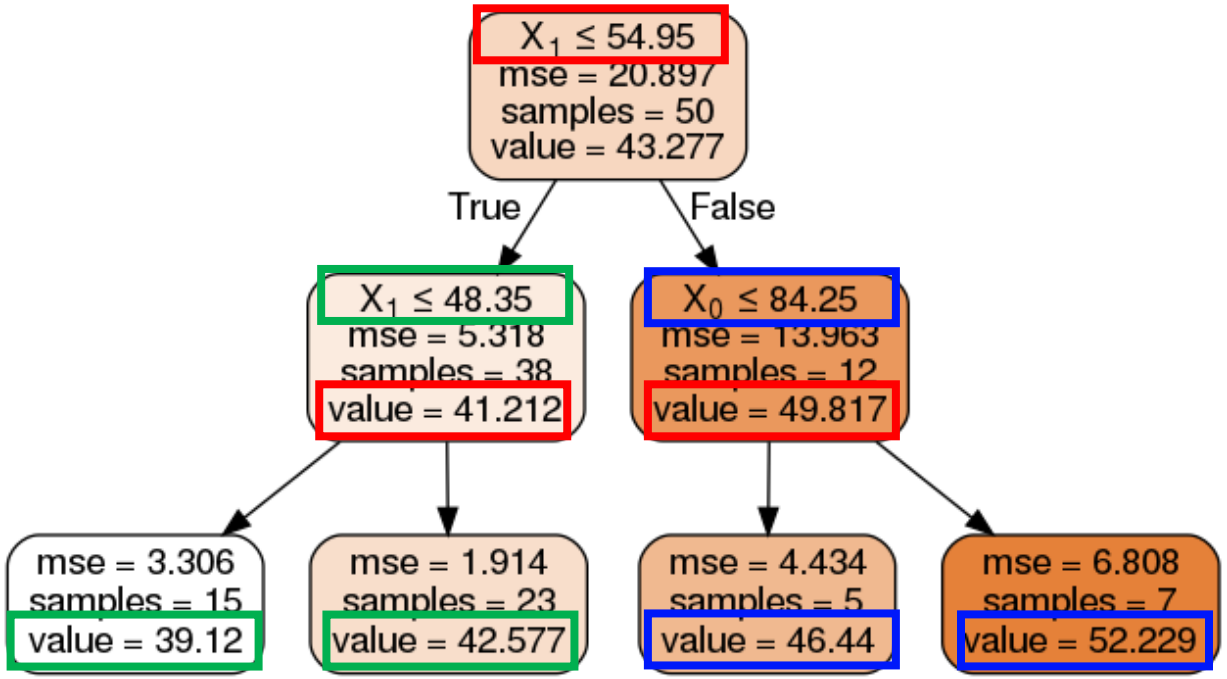
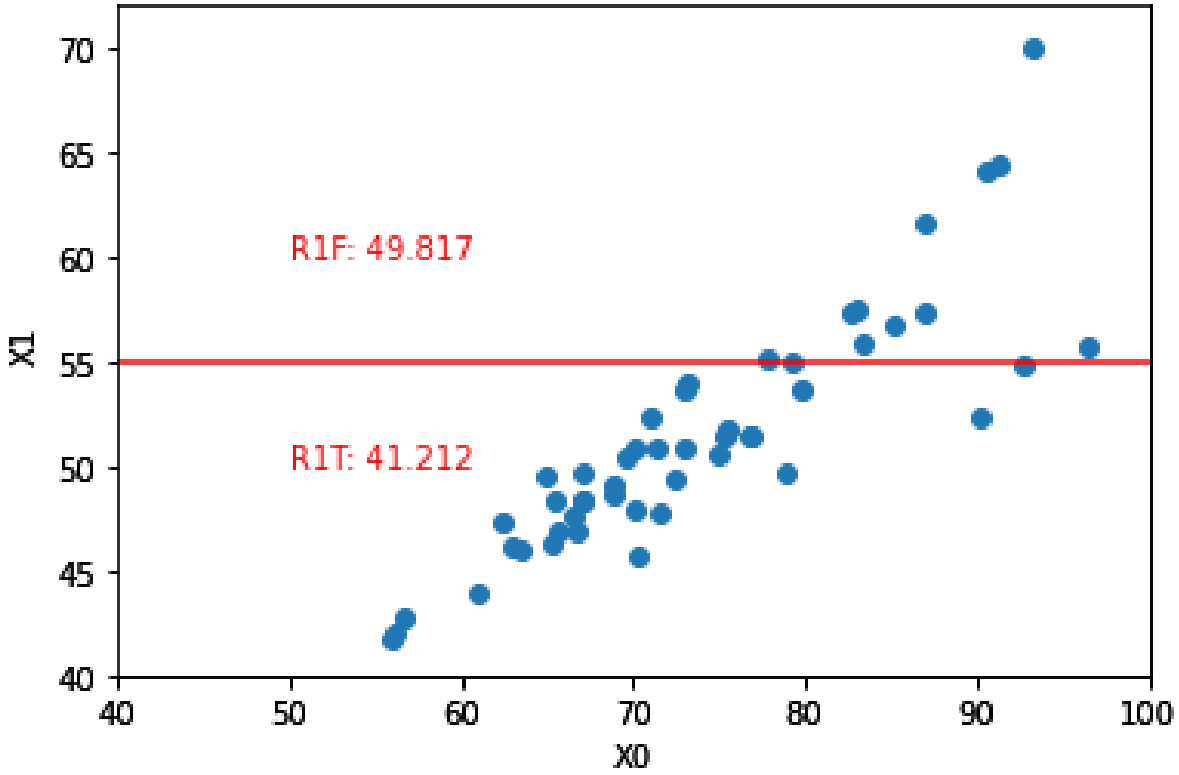
Decision Trees Hyperparameters

Quick Review





Quick Review- metrics for splitting

Regression Tree

MSE

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

MAE

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

Classification Tree

Gini

$$H(X_m) = \sum_k p_{mk}(1 - p_{mk})$$

Entropy

$$H(X_m) = -\sum_k p_{mk} \log(p_{mk})$$

Information Gain = E(parent)-E(children)

Outline

- **Usage in sklearn**
- **Hyperparameters**

Usage

```
from sklearn.tree import DecisionTreeRegressor, DecisionTreeClassifier
```

```
clf = DecisionTreeClassifier(random_state=0)
clf.fit(X,y)
```

class sklearn.tree.**DecisionTreeRegressor**(*, *criterion='mse', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, ccp_alpha=0.0*)

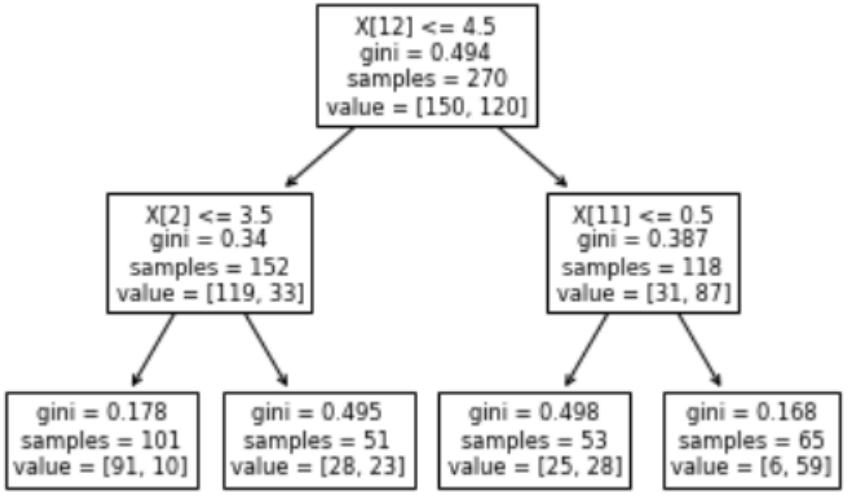
class sklearn.tree.**DecisionTreeClassifier**(*, *criterion='gini', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, class_weight=None, ccp_alpha=0.0*)

https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html
https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

Visualization

```
1 from sklearn import tree
2 tree.plot_tree(clf.fit(X, y))
```

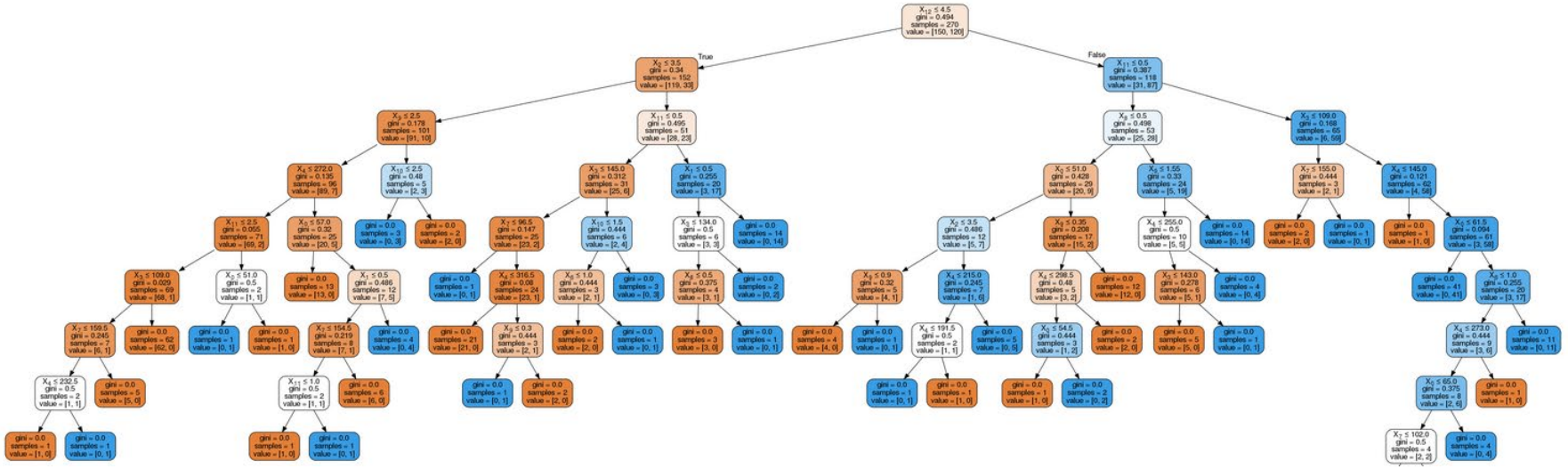
```
[Text(167.4, 181.2, 'X[12] <= 4.5\ngini = 0.494\nsamples = 270\nvalue = [150, 120]'),
 Text(83.7, 108.72, 'X[2] <= 3.5\ngini = 0.34\nsamples = 152\nvalue = [119, 33]'),
 Text(41.85, 36.23999999999998, 'gini = 0.178\nsamples = 101\nvalue = [91, 10]'),
 Text(125.55000000000001, 36.23999999999998, 'gini = 0.495\nsamples = 51\nvalue = [28, 23]'),
 Text(251.10000000000002, 108.72, 'X[11] <= 0.5\ngini = 0.387\nsamples = 118\nvalue = [31, 87]'),
 Text(209.25, 36.23999999999998, 'gini = 0.498\nsamples = 53\nvalue = [25, 28]'),
 Text(292.95, 36.23999999999998, 'gini = 0.168\nsamples = 65\nvalue = [6, 59]')]
```



Visualization

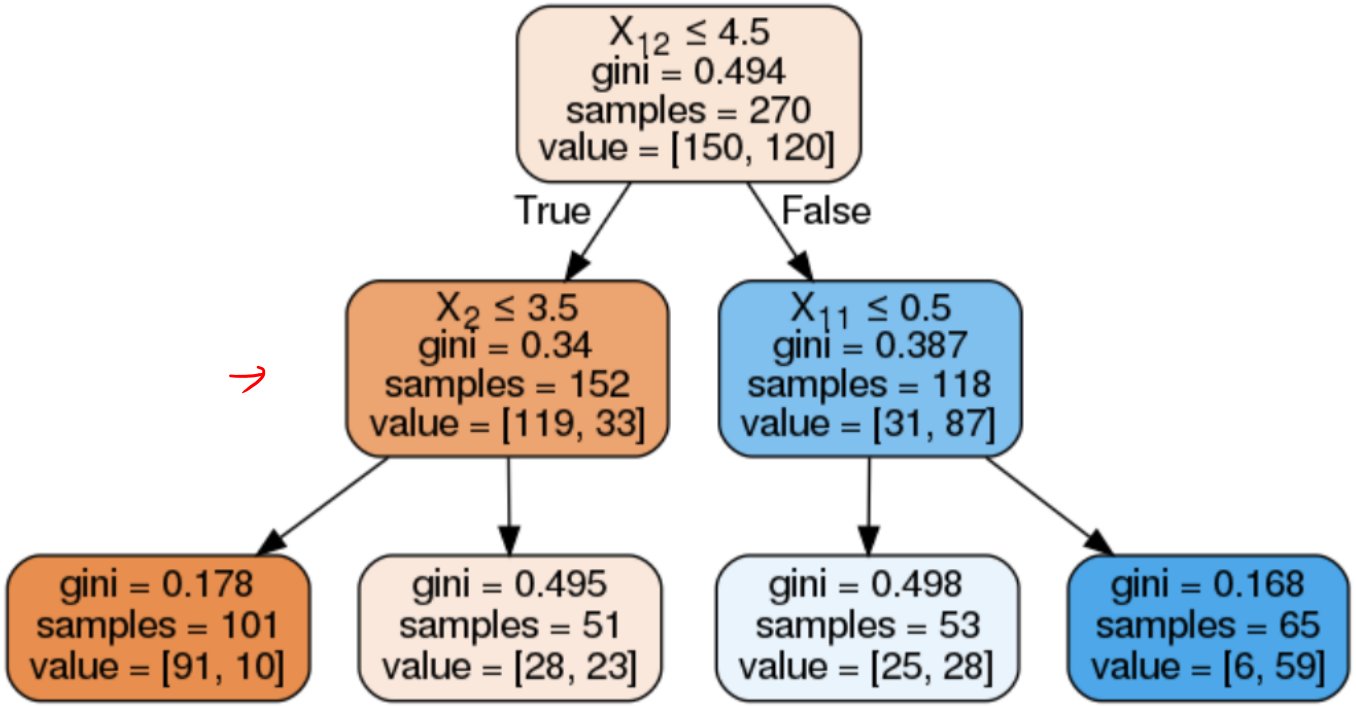
```
from sklearn.tree import export_graphviz
import graphviz
from six import StringIO
from IPython.display import Image
import pydotplus
```

```
dot_data = StringIO()
export_graphviz(clf, out_file=dot_data, filled=True, rounded=True, special_characters=True)
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
Image(graph.create_png())
```



Visualization

```
clf = DecisionTreeClassifier(max_depth=2, random_state=0)
clf.fit(X,y)
dot_data = StringIO()
export_graphviz(clf, out_file=dot_data, filled=True, rounded=True, special_characters=True)
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
Image(graph.create_png())
```



Ways to prevent overfitting

Decision trees are easy to overfit

- Early Stopping
- Pruning
- Ensembling

Ways to prevent overfitting: Early stopping

max_depth The maximum depth of the tree

min_samples_split The minimum number of samples required to split an internal node

min_samples_leaf The minimum number of samples required to be at a leaf node

min_weight_fraction_leaf The minimum weighted fraction of the sum total of weights required to be at a leaf node

min_impurity_decrease A node will be split if this split induces a decrease of the impurity greater than or equal to this value

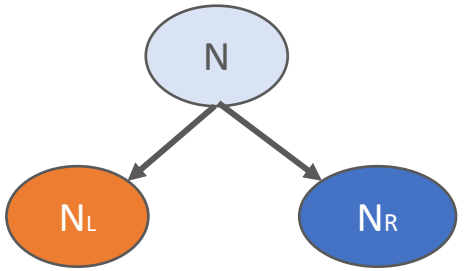
max_features The number of features to consider when looking for the best split

Hyperparameters

max_depth The maximum depth of the tree

min_samples_leaf The minimum number of samples required to be at a leaf node

min_impurity_decrease A node will be split if this split induces a decrease of the impurity greater than or equal to this value



$$H_0 - \left(\frac{N_L}{N} \cdot H_L + \frac{N_R}{N} \cdot H_R\right)$$

Other useful options

max_features The number of features to consider when looking for the best split

sqrt
log

class_weight None, balanced, custom

ccp_alpha Complexity parameter for minimal cost-complexity pruning

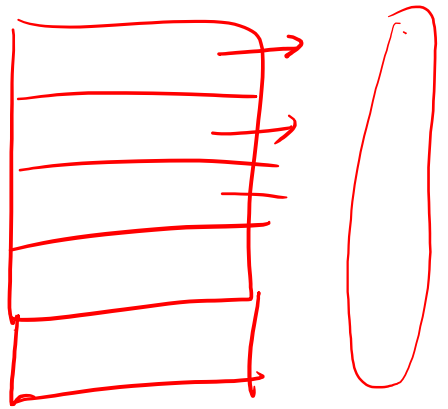
Grid Search

```
1 from sklearn.model_selection import GridSearchCV
```

```
1 rf = DecisionTreeClassifier()
2 parameters = {'max_depth':[3,5,7,10], 'min_samples_leaf':[1,2,5,10]}
3 clf = GridSearchCV(rf, parameters)
```

```
1 clf.fit(X,y)
```

```
GridSearchCV(estimator=DecisionTreeClassifier(),
             param_grid={'max_depth': [3, 5, 7, 10],
                         'min_samples_leaf': [1, 2, 5, 10]})
```



```
1 clf.best_estimator_
```

```
DecisionTreeClassifier(max_depth=3, min_samples_leaf=5)
```

```
1 clf.best_score_
```

```
0.8111111111111112
```