Decision Trees: CART

Machine Learning

Classification and Regression

- Decision Trees can be used for both

X1	X2	Y
0.268	0.266	Bad
0.219	0.372	Bad
0.517	0.573	Bad
0.269	0.908	Good
0.181	0.202	Bad
0.519	0.898	Good
0.563	0.945	Bad
0.129	0.661	Bad

- <u>Classification</u>
  - Spam / not Spam
  - Admit to ICU /not
  - Lend money / deny
  - Intrusion detections

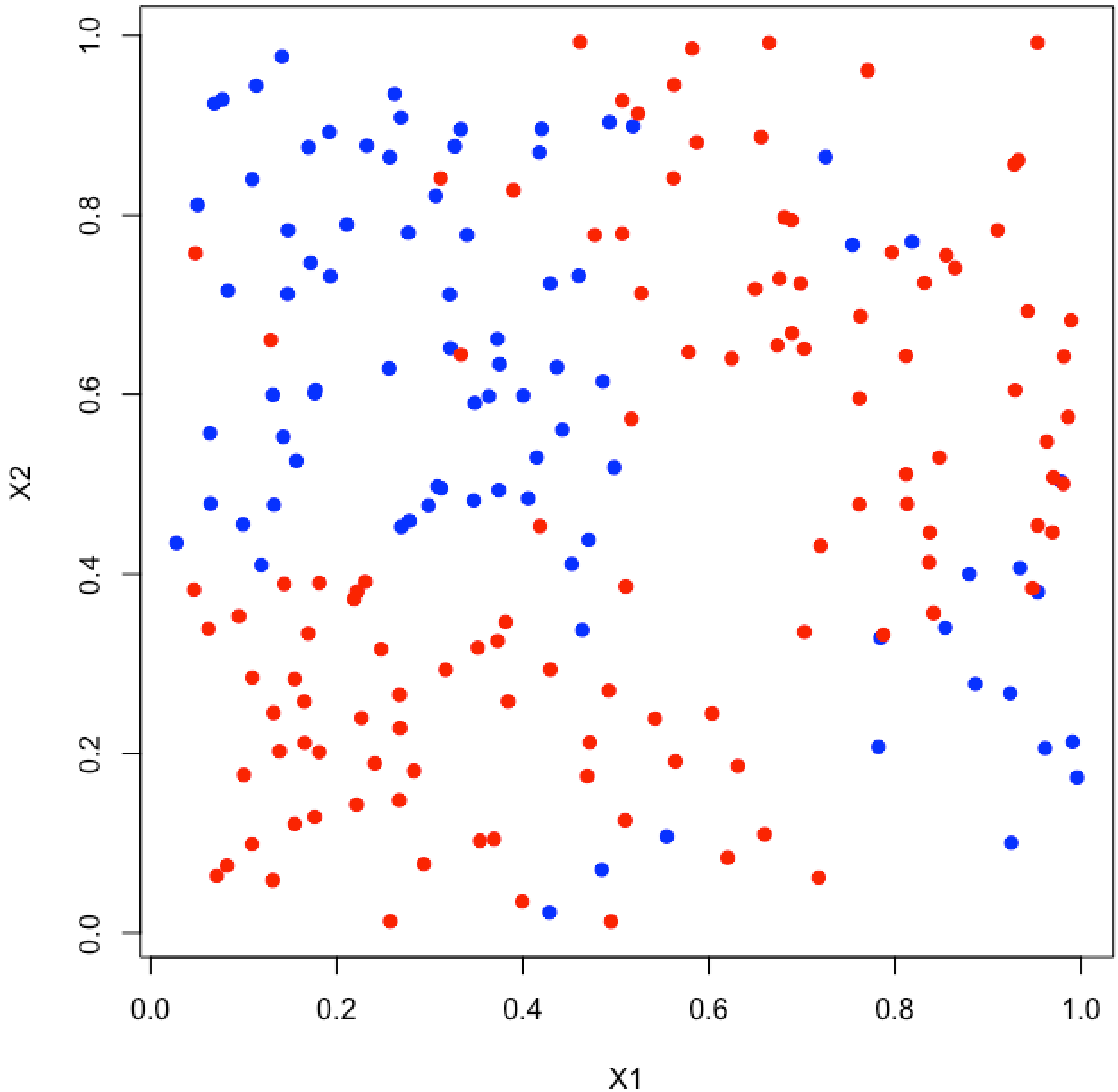
X1	X2	Y
0.268	0.266	64.41
0.219	0.372	28.08
0.517	0.573	95.76
0.269	0.908	15.84
0.181	0.202	41.83
0.519	0.898	25.20
0.563	0.945	9.44
[illegible]	[illegible]	[illegible]

- <u>Regression</u>
  - Predict stock returns
  - Pricing a house or a car
  - Weather predictions (temp, rain fall etc)
  - Economic growth predictions
  - Predicting sports scores

Decision Trees

- The general idea is that we will segment the space into a number of simple regions.
- The segmentation can be illustrated as a tree
- The end nodes can have a category (classification) or a continuous number (regression)
- These methods, while quite simple are very powerful.

Visualizing Classification as a Tree



Metrics

- Algorithms for constructing decision trees usually work top-down, by choosing a variable at each step that best splits the set of items.
- Different algorithms use different metrics for measuring “best"
- These metrics measure how similar a region or a node is. They are said to measure the impurity of a region.
- Larger these impurity metrics the larger the “dissimilarity” of a nodes/regions data.
- Examples: Gini impurity, Entropy, Variance

Algorithms for building Decision Trees

- Popular ones include
  - CART (Classification And Regression Tree)
  - CHAID (CHi-squared Automatic Interaction Detector)
  - C4.5

- We will focus on CART, that uses the Gini impurity as its impurity measure.

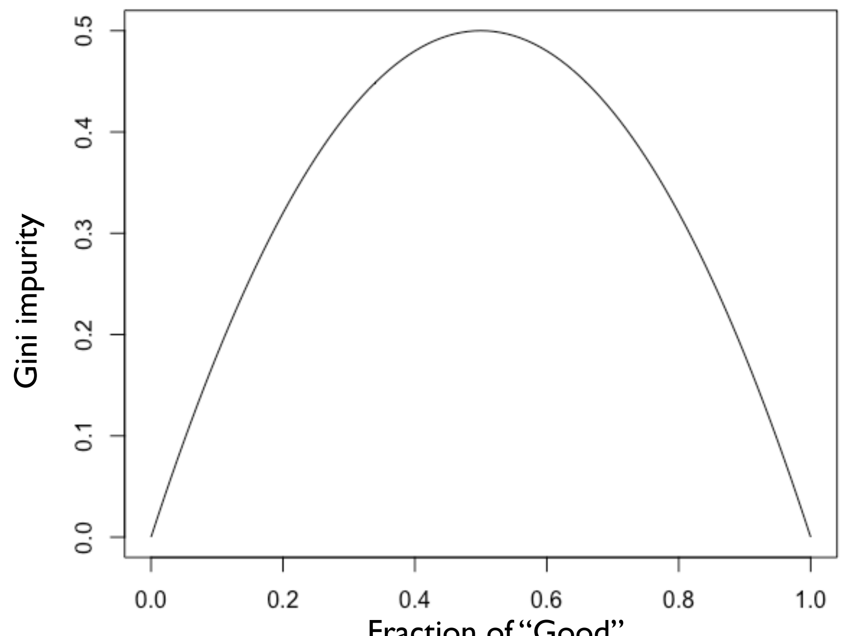
CART: An Example

Cust_ID	Gender	Occupation	Age	Target
1	M	Sal	22	1
2	M	Sal	22	0
3	M	Self-Emp	23	1
4	M	Self-Emp	23	0
5	M	Self-Emp	24	1
6	M	Self-Emp	24	0
7	F	Sal	25	1
8	F	Sal	25	0
9	F	Sal	26	0
10	F	Self-Emp	26	0

Gini impurity

- Used by the CART
- Is a measure of how often a randomly chosen element from the set would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset.
- Can be computed by summing the probability of an item with label i being chosen (p_i), times the probability of a mistake ($1 - p_i$) in categorizing that item.
- Simplifying gives, the Gini impurity of a set:

$$1 - \sum_{i=1}^{J} p_i^2$$

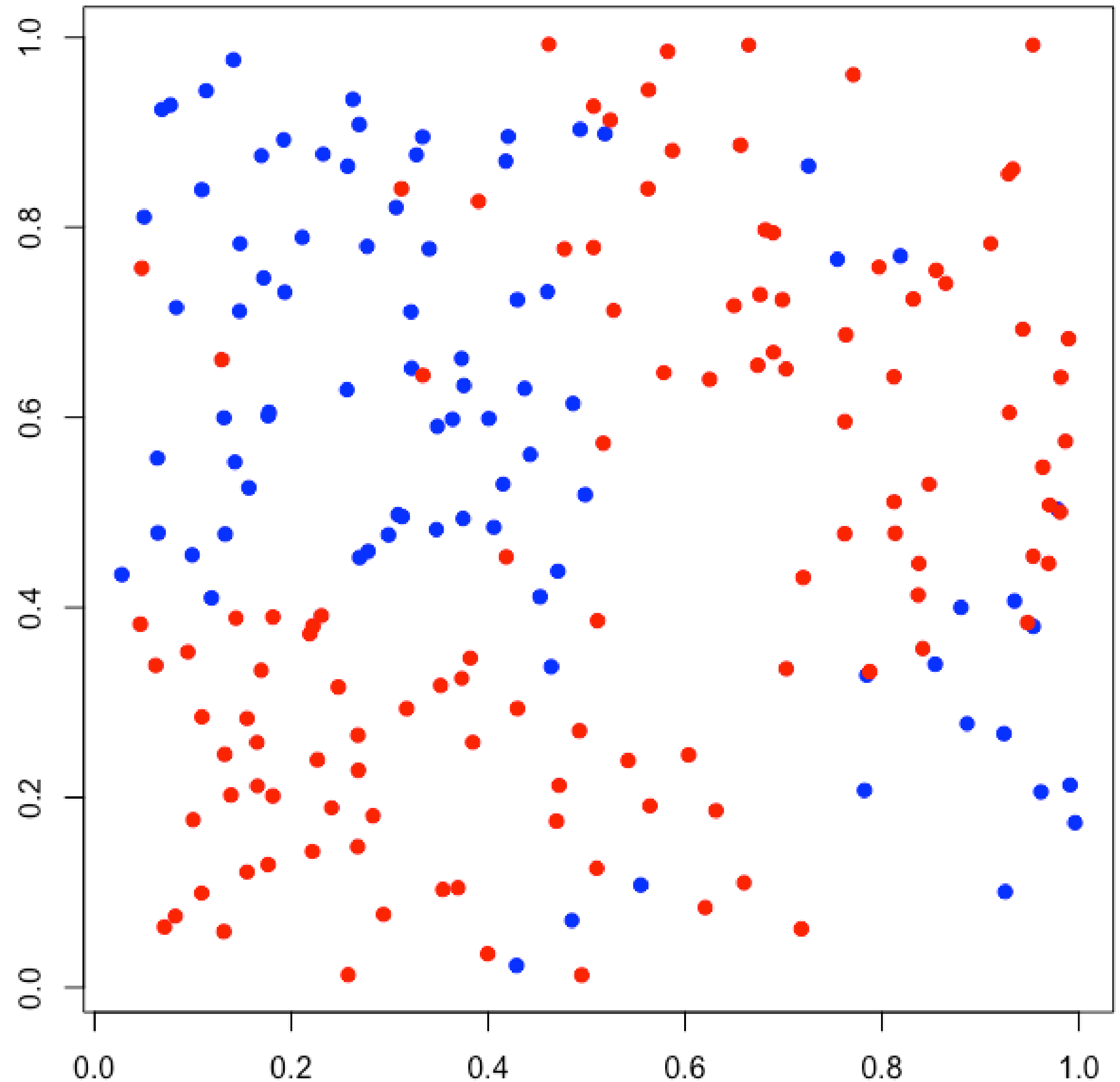


Splitting using Gini impurity

- When splitting, the Gini impurity of the two resulting nodes are combined using a weighted average.
- With weights being the fraction of data on each node.
- The CART algorithm simply chooses the right “split” by finding the split that maximizes the “decrease in Gini impurity” - also called the Gini Gain.

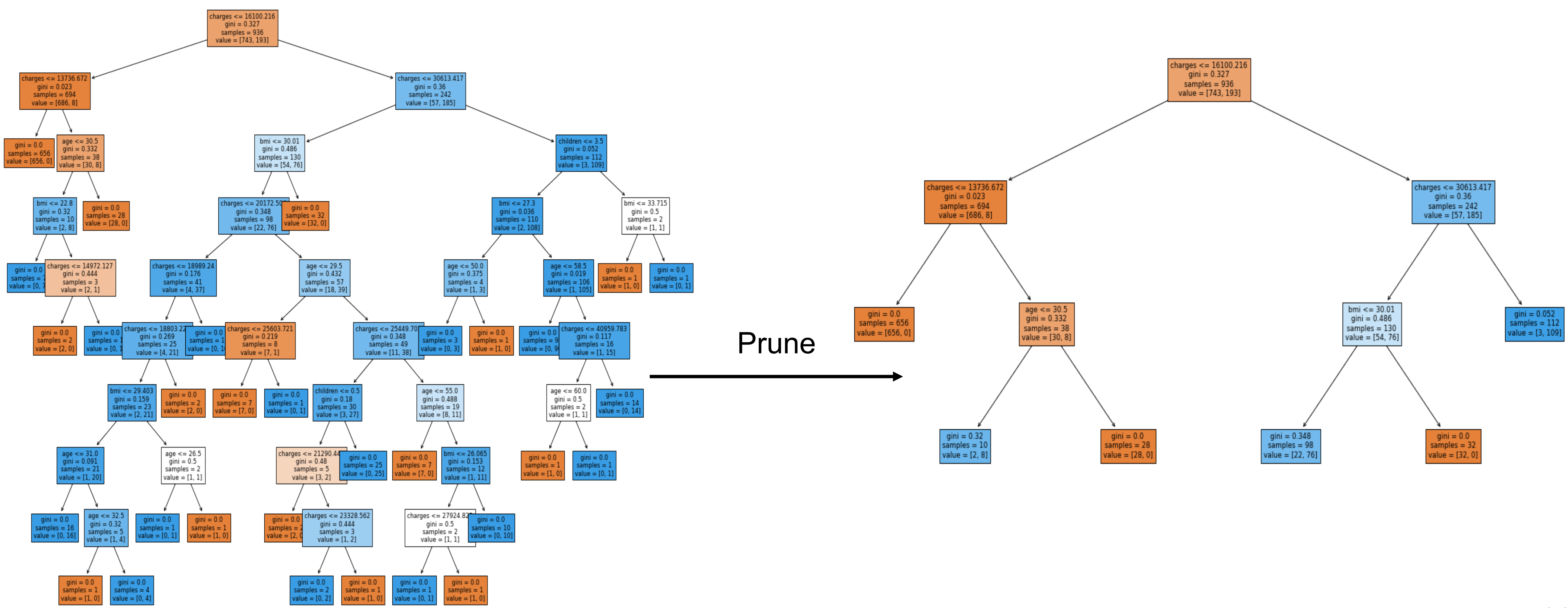
Decision trees are prone to ‘overfitting’

- Decision Tree is a powerful algorithm that can adapt well and capture various patterns in the data
- If allowed to grow fully, they become over-complex & tend to fit even the noise
- Thus, a fully grown tree may not ‘generalize’ well on test or new unseen data



Pruning

- Ideally we would like a tree that does not over-fit the given data
- One popular and simple way to prune a decision tree is by limiting the depth of the tree to avoid over fitting.
- For example the tree on the right below is generated with a max depth of 2 while the tree on the left has no depth restriction (and hence overfits the data)



Pre-Pruning

- Stop growing the tree before it grows too big

- This can be achieved by bounding hyperparameters

Hyperparameters in Decision Trees

- **max_depth** - The maximum depth of the tree. If set to 'None', then nodes are expanded until all leaves are pure. Higher the value, more complex the tree
- **min_samples_split** - The minimum number of samples required to split a node. Doesn't split any node that is smaller than this number. Higher the values, less complex the tree
- **min_samples_leaf** - The minimum number of samples required at a leaf node. All leaf nodes have at least these many data points. Higher the value, less complex the tree

Hyperparameter Tuning using Grid Search

- Grid Search is a process of searching the best combination of hyperparameters from a predefined set of values

- A parameter grid (Hyperparameters and corresponding values) is provided as an input to the Grid-search function

- It tries all the combinations of the values passed and evaluates the model for each combination

- It returns the combination of hyperparameter values that works best as per the metric provided for model evaluation

- GridSearchCV( ) is an implementation of Grid Search with Cross Validation

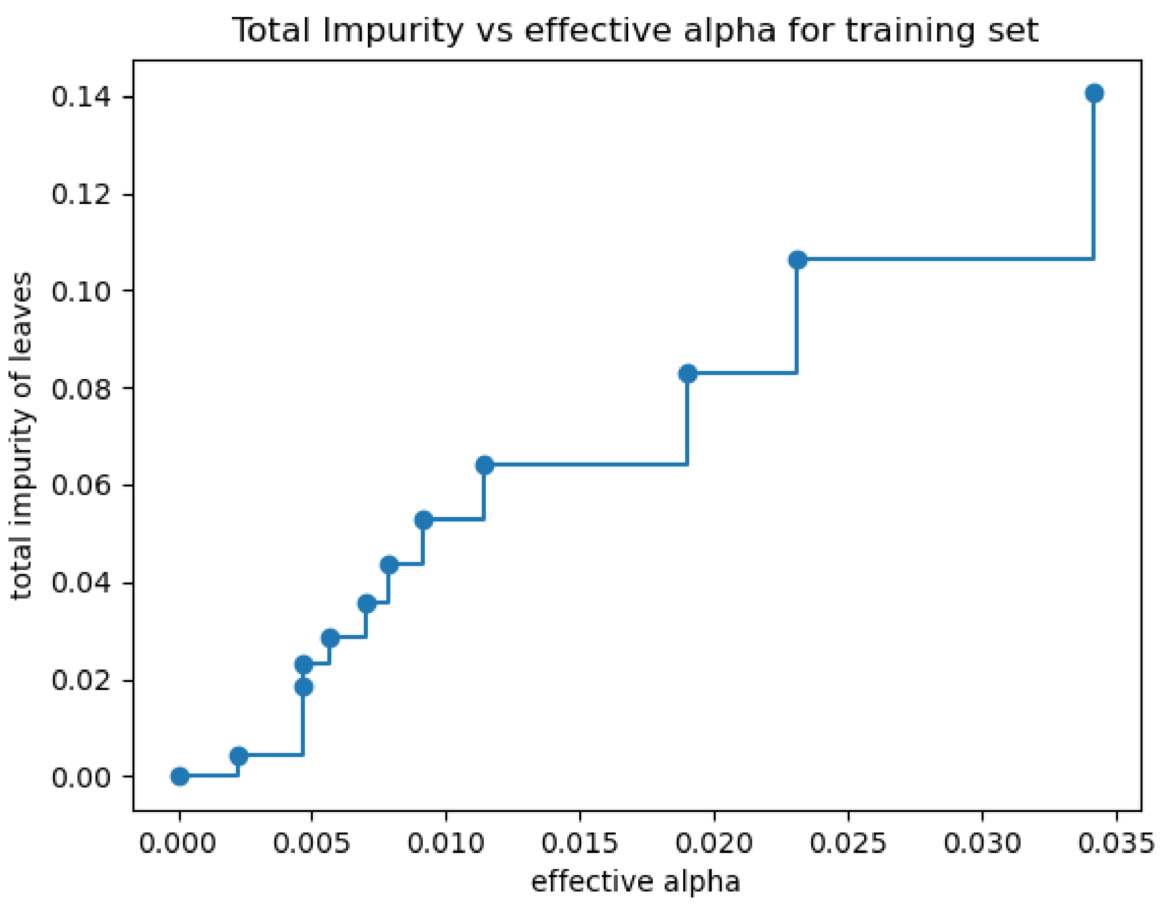
Cross Validation

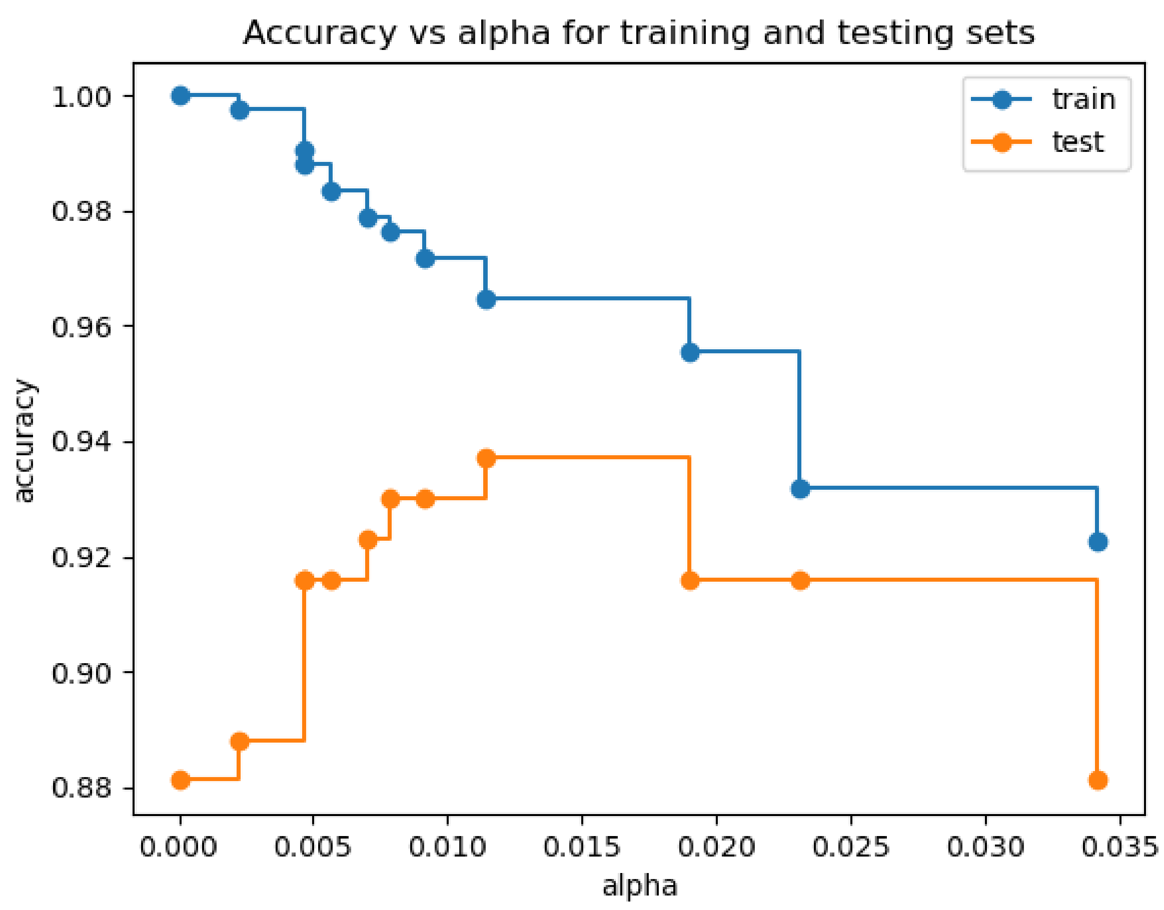
- Cross Validation is a common Machine Learning technique that splits the data into n non-overlapping groups, and runs n experiments:
    - In each experiment, n-1 groups are used to train a model and the model is tested on the left out group.
    - The results are summarized over the n experiments.
- It gives a mechanism that allows us to test a model repeatedly on data that was not used to build the model.
- For Decision Trees, a very common approach is simply to choose the tree with minimum cross validation error

Post-Pruning: Cost-complexity pruning

α

- Starting from the Full tree, create a sequence of trees that are sequentially smaller (pruned)
- At each step the algorithm
  - try removing each possible subtree
  - find the ‘relative error decrease per node’ for that subtree - Complexity parameter,
  - And remove the subtree with the minimum
- With the list of subtrees, one usually reverts back to using cross-validation errors to find the best final pruned tree


Total Impurity vs effective alpha for training set
total impurity of leaves
effective alpha
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0.035


Accuracy vs alpha for training and testing sets
accuracy
alpha
train
test
0.88
0.90
0.92
0.94
0.96
0.98
1.00
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0.035

Few Additional Thoughts

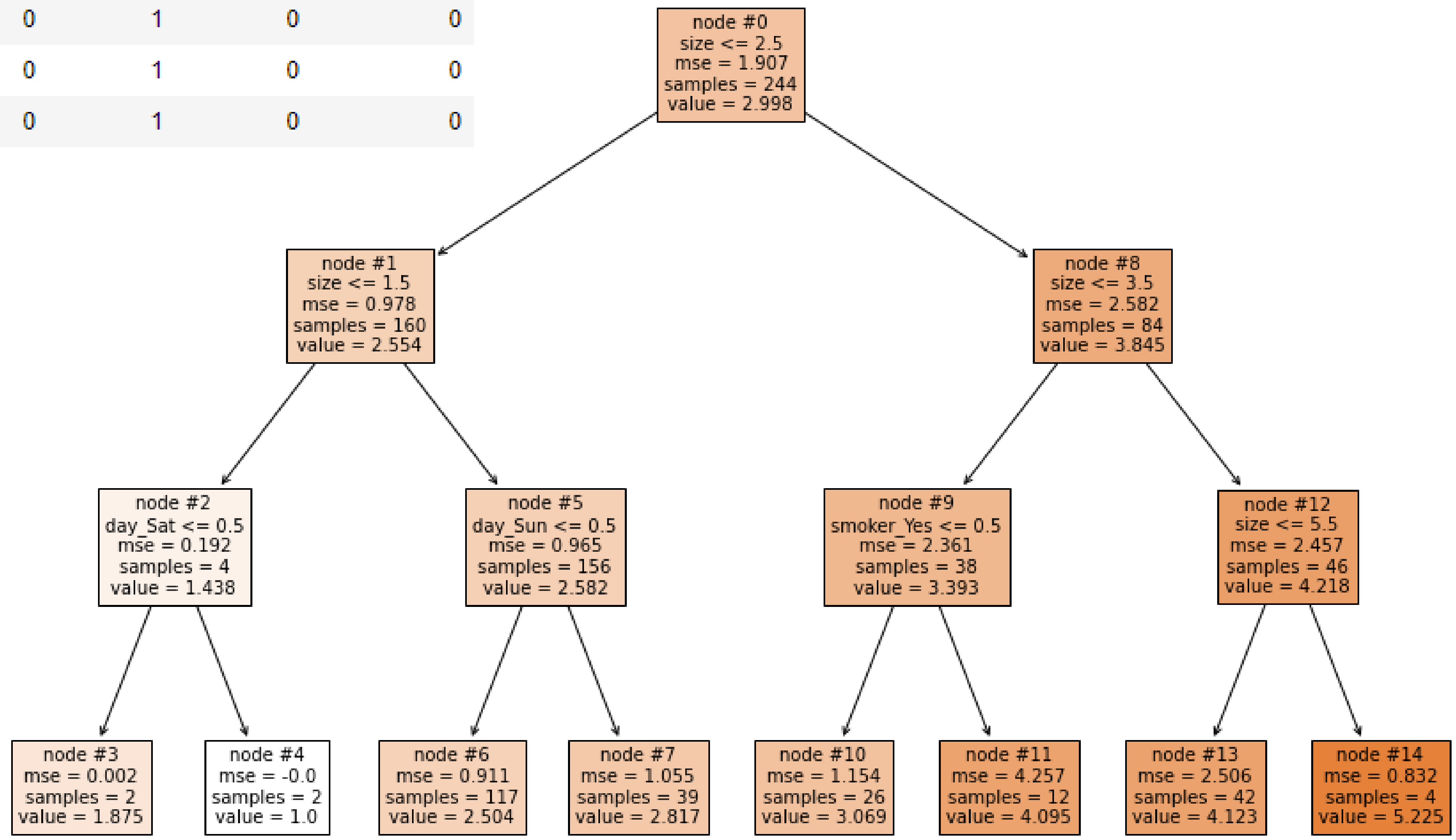
- Other impurity measures
- Regression Trees
- Pros and Cons

Impurity Measures in Decision Trees

	GINI INDEX	ENTROPY	INFORMATION GAIN	VARIANCE
When to use	Classification	Classification	Classification	Regression
Formula	$1 - \Sigma p_i^2$	$-\Sigma p_i \log(p_i)$	$E(Y) - E(Y\|X)$	$\Sigma(x - \bar{x})^2/N$
Range	0 to 0.5 0 = most pure 0.5 = most impure	0 to 1 0 = most pure 1 = most impure	0 to 1 0 = less gain 1 = more gain	>=0
Characteristics	Easy to compute Non-additive	Computationally intensive Additive	Computationally intensive	The most common measure of spread

Regression Trees

	tip	size	sex_Male	smoker_Yes	day_Sat	day_Sun	day_Thur	time_Lunch
0	1.01	2	0	0	0	1	0	0
1	1.66	3	1	0	0	1	0	0
2	3.50	3	1	0	0	1	0	0
3	3.31	2	1	0	0	1	0	0
4	3.61	4	0	0	0	1	0	0
5	4.71	4	1	0	0	1	0	0
6	2.00	2	1	0	0	1	0	0



Pros and Cons of Decision Trees

Pros -

- Easy to understand and interpret
- Useful in data exploration as it gives the splitting based on the significance of variables
- Not influenced by the outlier/Null values and hence requires less data cleaning. Requires less time and effort during data pre-processing than other algorithms.
- Can handle both continuous and categorical variables
- Does not require any underlying assumptions in data. Works with both linearly and nonlinearly related variables.

Pros and Cons of Decision Trees

Cons-

- A small change in the data-set can result in large change in the structure of the decision tree causing instability in the model.
- Large trees can be difficult to interpret.
- Tends to overfit.