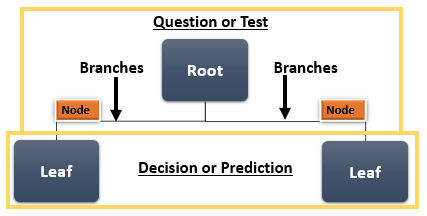
**Decision Tree Basics & Regression Trees**

**Basic Tree Structure and Nomenclature**

* A decision tree starts with a single **root node** that splits into multiple branches, leading to further **nodes**, each of which may further split or else terminate as a ***leaf node***.
* Associated with each non-leaf node will be a test or question that determines which branch to follow.
* The leaf nodes contain the "decisions" or "predictions."

****

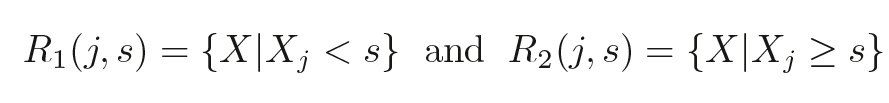
**Recursive Binary Splitting (A Greedy Methodology)**

Because we are looking through every possible partition of the feature space it would be computationally infeasible to consider every possible partition of the feature space, so we use a top-down , Greedy Methodology, that is also known as recursive binary splitting .

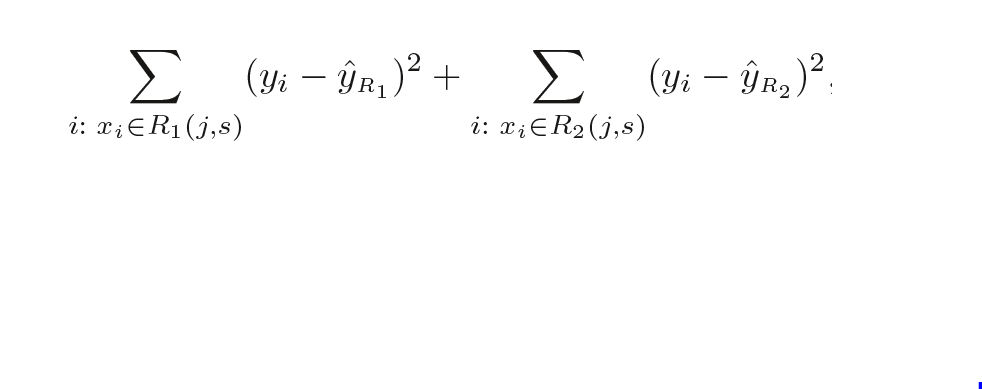
* It is top-down - because it begins at the top of the tree and then successively splits the predictor space
* It is greedy - because at each step of the tree building process, the best split is made at that particular step, rather than looking ahead and picking a split that will lead to a better tree in some future step
* It is recursive - because we repeat the process, looking for the best predictor and best cutpoint in order to split the data further as to minimize the RSS within the resulting regions.

**Splitting Predictor Space Notation:**

The notion means the region of predictor space in which Xj takes on a value less than the split (s).

****

**Residual Sum of Squares:**



**Application in R**

**regression.tree = rpart( ~ . ,data= " ", method="anova",**

**parms=list(split="information"), control=rpart.control (minsplit=10,**

**minbucket=5, maxdepth=20))**

* rpart() function builds the tree using two arguments for tuning the algorithm
  + parms =
  + control =
* method = "anova" - indicates the type of model to be built - use "anova" for numeric variables when we build regression models
* control = rpart.control ()
  + minsplit = specifices the minimum number of observation in a node - the default is minsplit = 20
  + minbucket = sets the minimum number of observations in any leaf node - default is set for 7 - the smaller the minsplit values, the larger the decision tree
  + maxdepth = sets the maximum depth of any node of the final tree, with the root node counted as depth 0

**Build a Maximal Spanning Tree using the following parameters:**

* **cp = 0, minbucket = 1, and minsplit = 2**
  + You will use the parameters listed above to build the maximal spanning tree because you want the minimums to be low so the tree will continue to be split
  + Building a maximal spanning tree is useful if you want to look at the values for CP (complexity Parameter) for various tree sizes
* **printcp( )**
  + This will show you the cp table with the number of splits, error, cross-validation error, and standard deviation
  + Use this table to find the lowest cross-validation error by calling:
    - xerr = mymodel\_max$cptable[,"xerror"]
    - minxerr = which.min(xerr)
    - mincp = mymodel\_max$cptable[minxerr,"CP"]
  + Or you can apply the one standard deviation rule
    - simplest model within one standard deviation of the min
  + plotcp(model)
    - You can plot the cross validation errors to get a visual representation of the table that you printed (printcp())
* **Prune the tree using the values found in the cp table:**
  + The following lines of code will be used in R to prune your maximal spanning tree:
    - mymodel\_max.prune = prune(mymodel\_max,cp=mincp)
    - mymodel\_max.prune$cptable