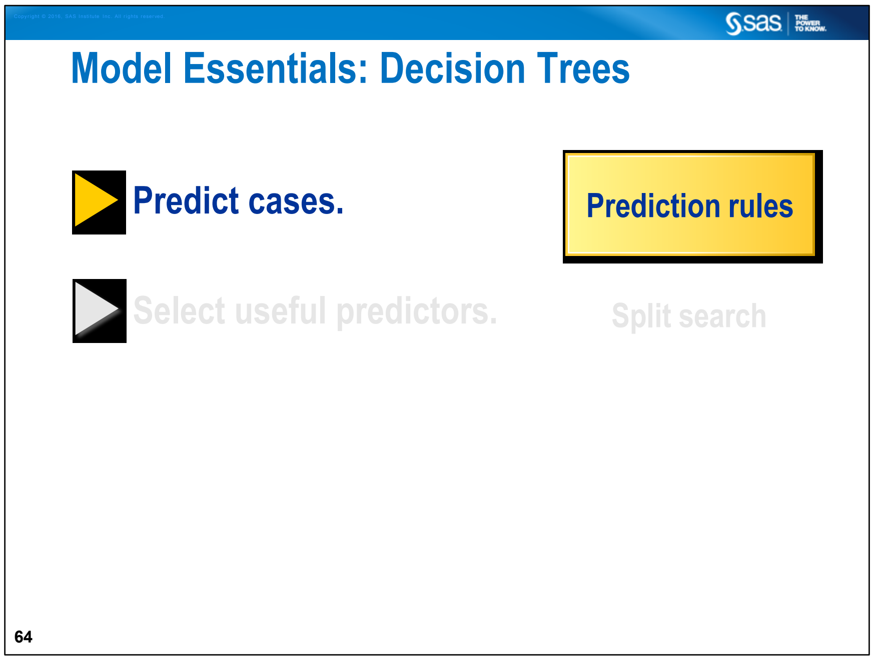
Tree Split Search in SAS Visual Statistics (Self-Study)

As seen in the previous section, regressions, as parametric models, assume a specific association structure between inputs and target. By contrast, decision trees, as predictive algorithms, do not assume any association structure. They simply seek to isolate concentrations of cases with like-valued target measurements.

Decision trees are similar to other modeling methods described in this course. Cases are scored using *prediction rules*. A *split-search* algorithm facilitates predictor variable selection.

Useful predictions depend, in part, on a well-formulated model. Good formulation primarily consists of preventing the inclusion of redundant and irrelevant predictors (input variables) in the model. The predictor variable selection function is complicated with large data. There are usually many predictors to consider and several pieces of information (rows) about these columns to process. This complication adds to the requirements of the input search method for any given model. The method must eradicate redundancies and irrelevancies, and also be extremely efficient. The input search methods that are available in the decision tree algorithm in Visual Statistics are described in this section.

The simple prediction problem shown below illustrates each of these model essentials.

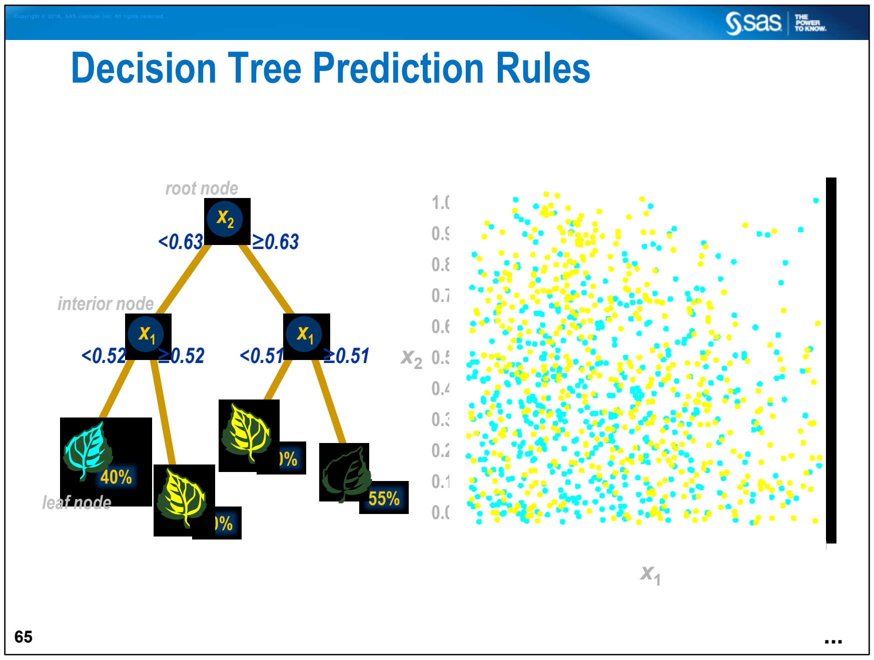


Model Essentials: Decision Trees

- Predict cases. — Prediction rules
- Select useful predictors. — Split search

64

Consider again a data set with two predictors and a binary target. The predictors, x_1 and x_2, locate each case in the unit square. The target outcome is represented by a color. Yellow is primary and blue is secondary. The analysis goal is to predict the outcome based on the location in the unit square.

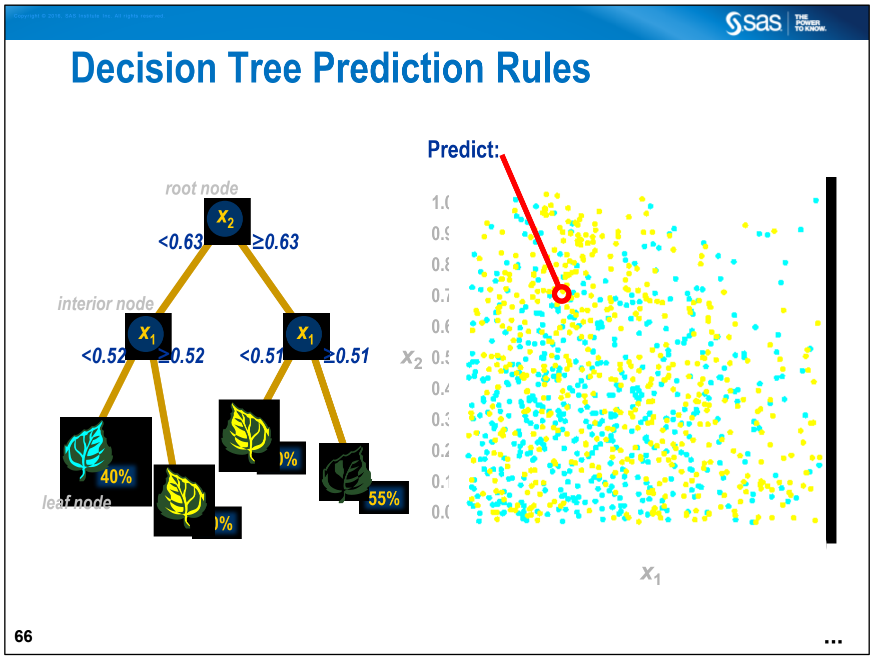


Decision Tree Prediction Rules

To predict cases, decision trees use rules that involve the values of the predictor variables.

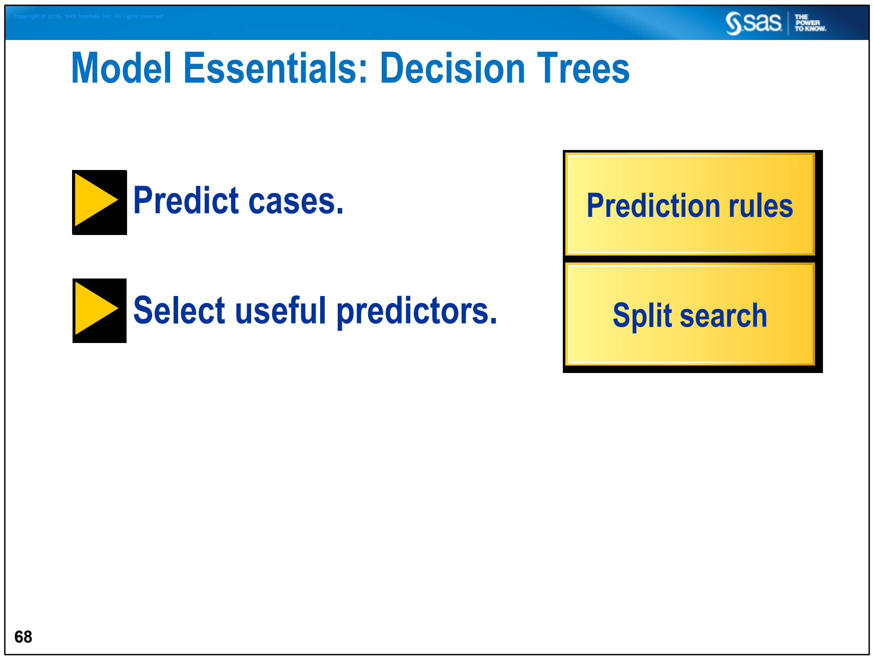
The rules are arranged hierarchically in a tree-like structure with nodes connected by lines. The nodes represent decision rules, and the lines order the rules. The first rule, at the base (top) of the tree, is named the *root node*. Subsequent rules are named *interior nodes*. Nodes with only one connection are *leaf nodes*.

To score a new case, examine the associated input variable values and then apply the rules defined by the decision tree.



Decision Tree Prediction Rules

The input variables' values of a new case eventually lead to a single leaf in the tree. A tree leaf provides a decision (for example, classify as yellow) and an estimate (for example, the primary-target proportion).



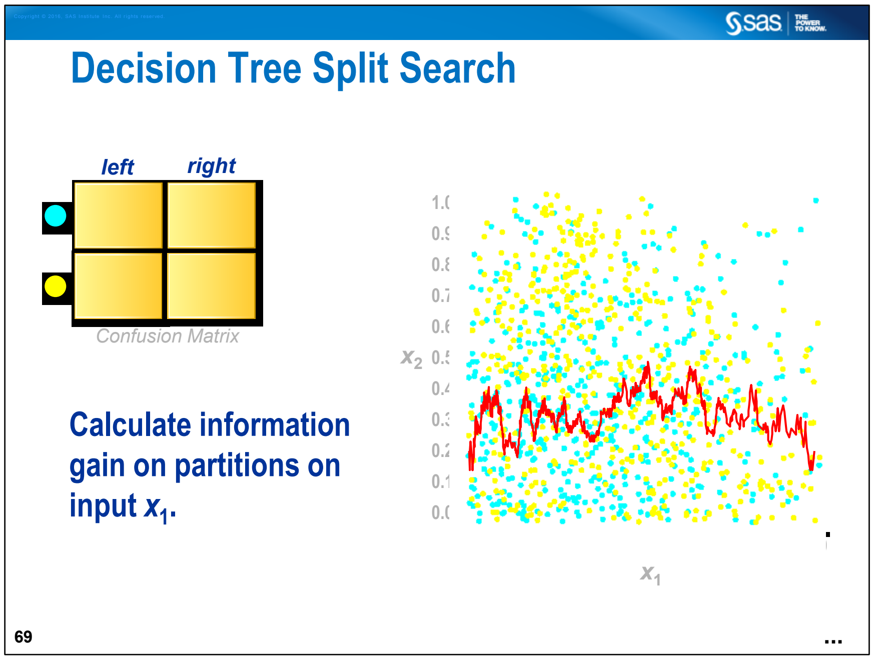
Model Essentials: Decision Trees

- Predict cases.
- Select useful predictors.

Prediction rules
Split search

68

To select useful predictors, trees use a *split-search* algorithm. Decision trees confront the curse of dimensionality by ignoring irrelevant predictors.



Decision Tree Split Search

Calculate information gain on partitions on input x_1.

69 ...

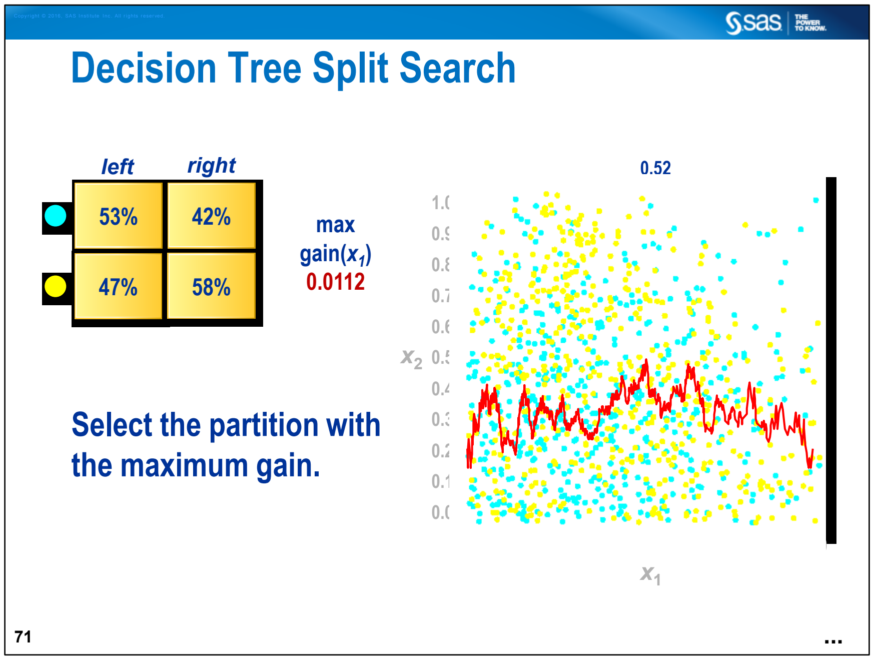
Understanding a split search algorithm for building trees enables you to better use the Tree tool and interpret your results. The description presented here assumes a binary target, but the algorithm for interval targets is similar.

The first part of the algorithm is called the *split search*. The split search starts by selecting an input for partitioning the available training data. If the measurement scale of the selected input is *interval*, unique values serve as a potential split point for the data. If the input is *categorical*, the average value of the target is taken within each categorical input level. The averages serve the same role as the unique interval input values.

For a selected input and fixed split point, two groups are generated. Cases with input values less than the split point are said to *branch left*. Cases with input values greater than the split point are said to *branch right*. The groups, combined with the target outcomes, form a 2x2 contingency table with columns specifying branch direction (left or right) and rows specifying target value (0 or 1). An information gain statistic that is based on the entropy of the root node and the entropy of the data in each partition of the split can be used to quantify the separation of counts in the table's columns. Large values for the gain statistic suggest that the proportion of zeros and ones in the left branch is different from the proportion in the right branch. A large difference in outcome proportions indicates a good split. An example of this calculation is given below.

The split search diagnostic used in Visual Statistics depends on the approach used to grow or train the tree. Under Interactive training, a chi-squared log-worth-based statistic is used to evaluate splits. The default split search method under autonomous tree growth is based on an information gain statistic. An example of calculating the gain under the default method is given below. The Rapid Growth functionality combines *k*-means clustering with the gain statistic to grow the tree.

A gain ratio statistic is used for split evaluation when the Split Best option is used in combination with the Rapid Growth property.



Decision Tree Split Search

	left	*right*
🔵	53%	42%
🟡	47%	58%

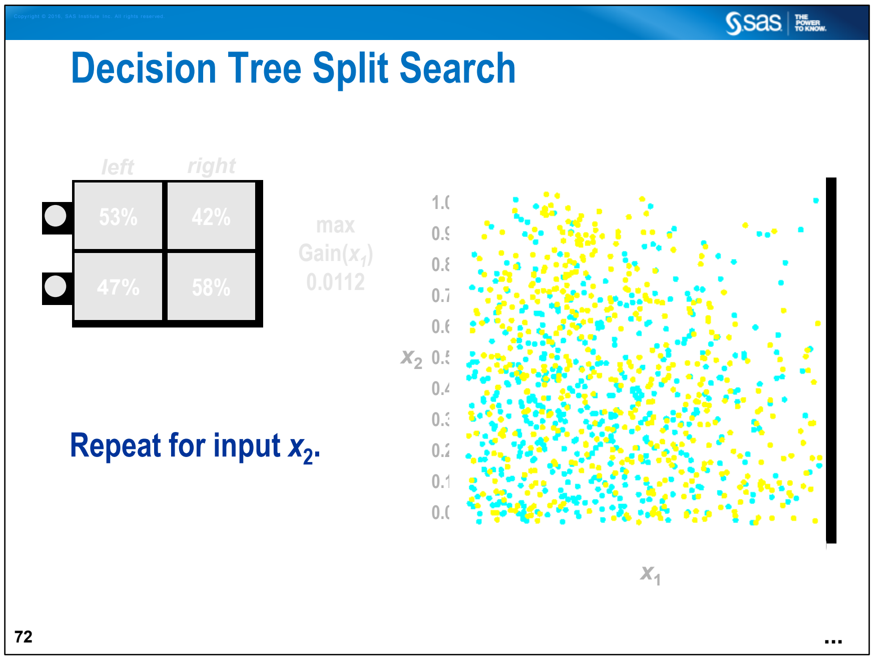
max gain(x_1) 0.0112

Select the partition with the maximum gain.

71 ...

The best split for a predictor is the split that yields the highest information gain. For the gain calculation example, assume that there are 100 total observations and a 50/50 split of yellow and blue in the training data. Also, there are 52 observations to the left of the 0.52 split point and 48 observations to the right of the split in the diagram shown above. Based on this and the numbers given in the table, gain can be formulated as shown below.

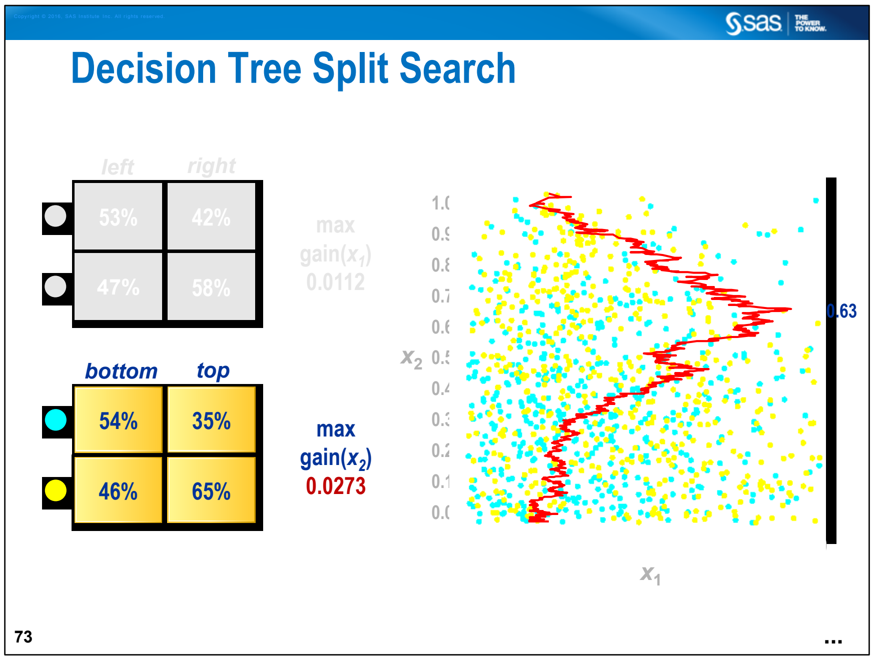
$$
\begin{aligned}
&\mathit{Entropy\,Total} \quad -.5\cdot\log_2(.5)-.5\cdot\log_2(.5)=1\\
&\mathit{Entropy\,Left} \quad -.53\cdot\log_2(.53)-.47\cdot\log_2(.47)=0.997\\
&\mathit{Entropy\,Right} \quad -.42\cdot\log_2(.42)-.58\cdot\log_2(.58)=0.98\\
&\mathit{Gain} \quad 1-(52/100)\cdot 0.997-(48/100)\cdot 0.98=0.0112
\end{aligned}
$$



Decision Tree Split Search

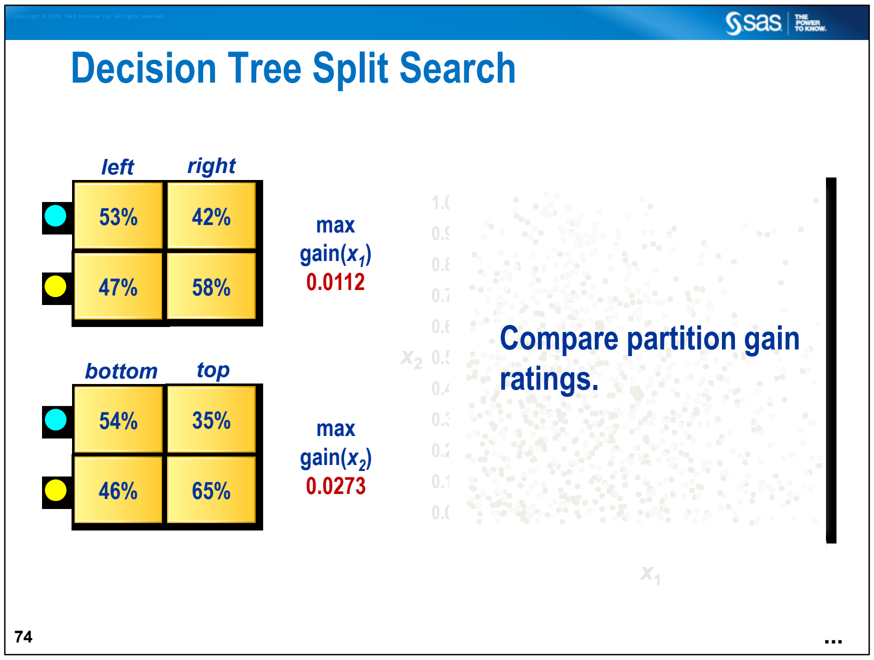
Repeat for input x_2.

The partitioning process is repeated for every input in the training data.



Decision Tree Split Search

Again, the optimal split for the input is the one that maximizes the gain function.



Decision Tree Split Search

	left	right
●	53%	42%
●	47%	58%

max gain(x_1) 0.0112

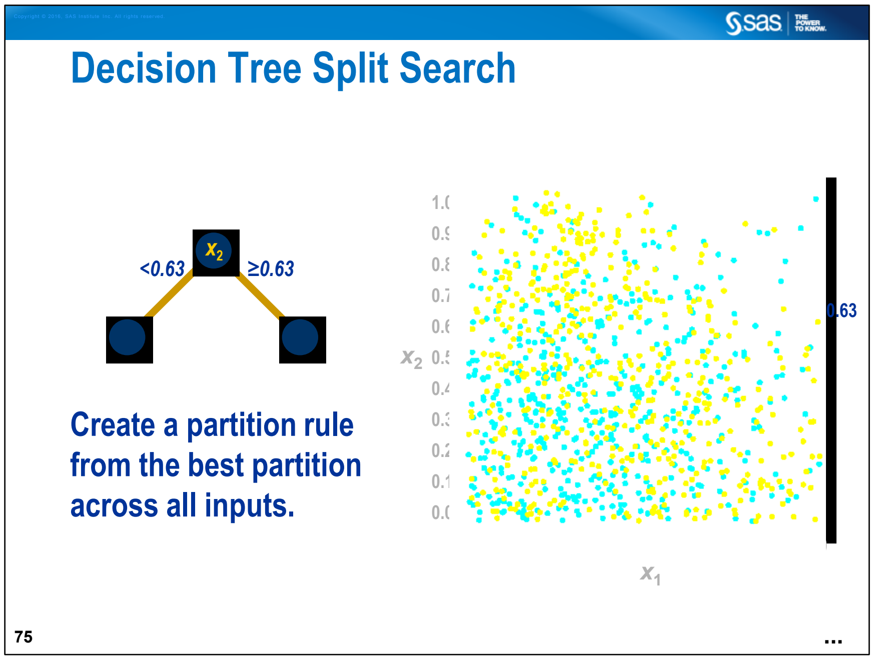
	bottom	top
●	54%	35%
●	46%	65%

max gain(x_2) 0.0273

Compare partition gain ratings.

74 ...

After you determine the best split for every input, the tree algorithm compares each best split's corresponding gain. The split with the highest gain is deemed best.

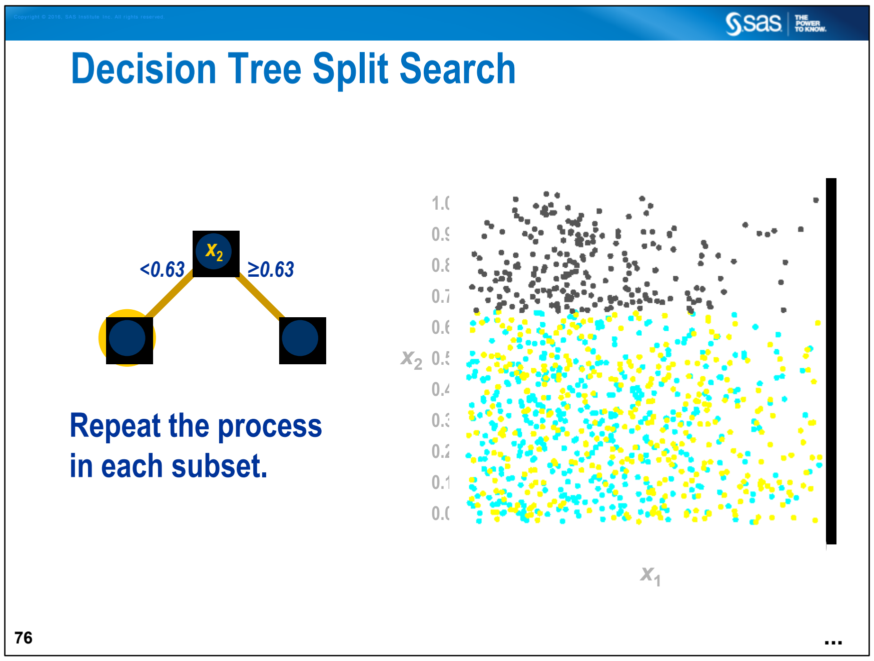


Decision Tree Split Search

Create a partition rule from the best partition across all inputs.

75 ...

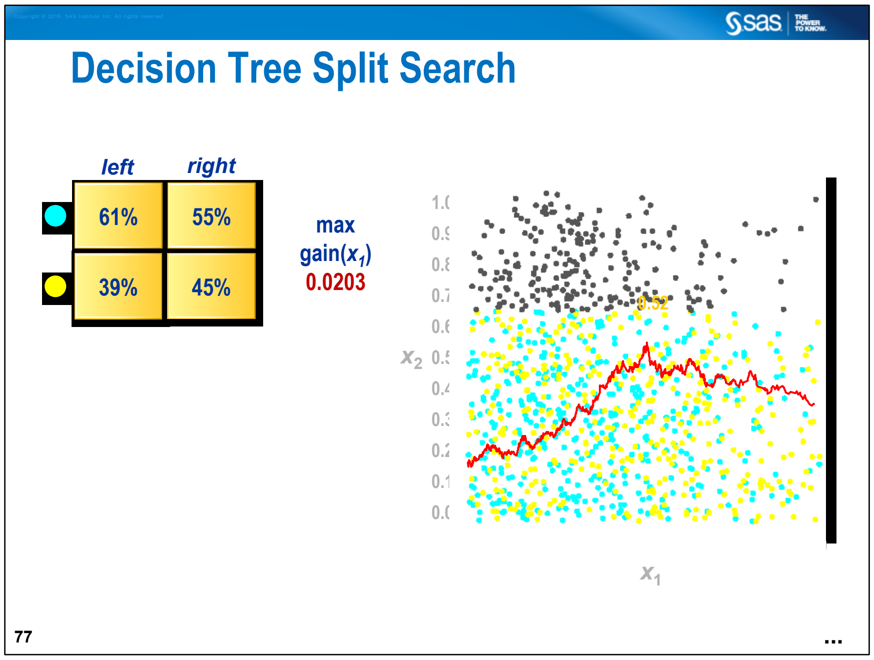
The training data are partitioned using the best split rule.



Decision Tree Split Search

Repeat the process in each subset.

76 ...

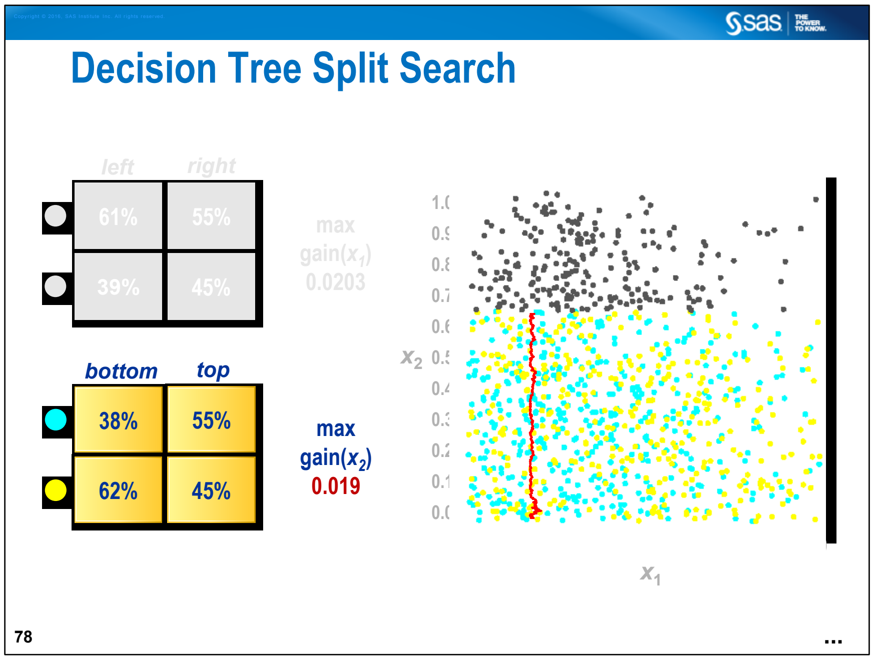


Decision Tree Split Search

	left	*right*
●	61%	55%
●	39%	45%

max gain(x_1) 0.0203

77 ...



Decision Tree Split Search

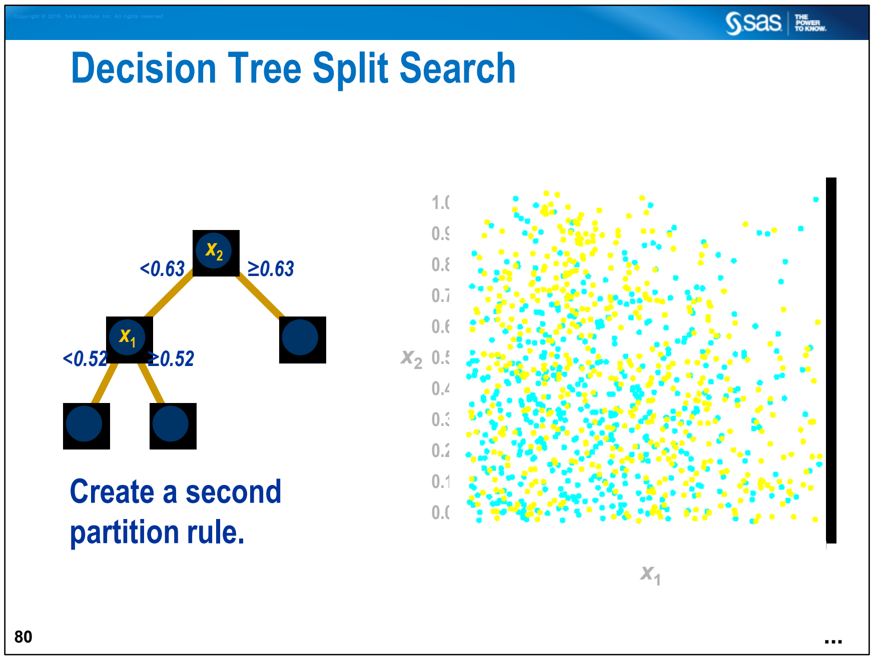
	left	right
●	61%	55%
●	39%	45%

max gain(x_1) 0.0203

	bottom	top
●	38%	55%
●	62%	45%

max gain(x_2) 0.019

78 ...

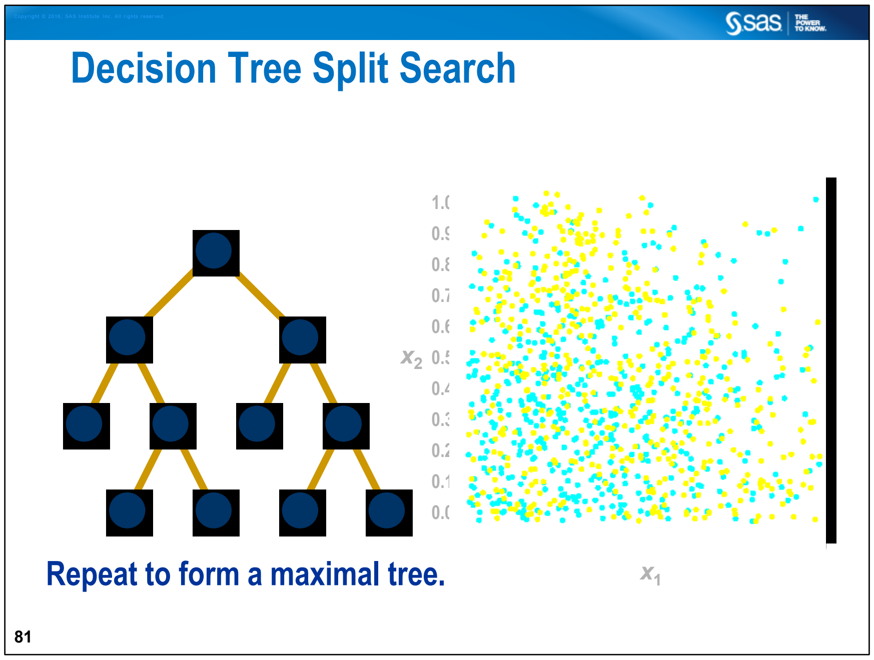


Decision Tree Split Search

Create a second partition rule.

80 ...

The split search continues within each leaf. Gain statistics are compared as before.



Decision Tree Split Search

Repeat to form a maximal tree.

81

The resulting partition of the predictor variable space is known as the *maximal tree*. Under the default settings, development of the maximal tree is based exclusively on statistical measures of gain on the training data.