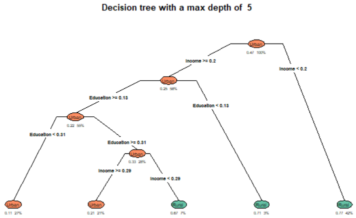
Machine Learning for Official Statistics & SDGs

Decision Trees

UNITED NATIONS SIAP
Statistical Institute for Asia and the Pacific

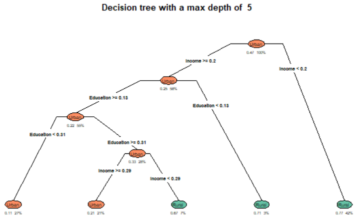
[Decision Trees]

Trees are method for classification or regression analysis.
↪ Focus on classification



[Decision Trees]

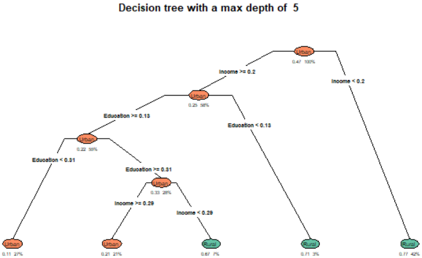
Trees are method for classification or regression analysis.
↪ Focus on classification



- Trees split the space into non-overlapping spaces
- Used to assign/predict a *class* following conditions
- Optimally, final classification is homogenous

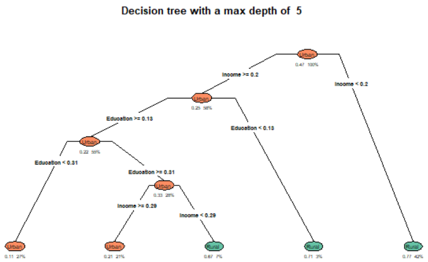
[What's in a tree?]

Trees have a very simple structure and are easy to understand:



[What's in a tree?]

Trees have a very simple structure and are easy to understand:

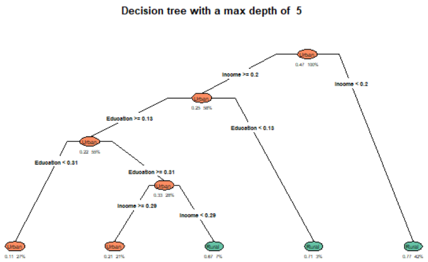


Trees have:

- **Nodes** where splitting decisions are done

[What's in a tree?]

Trees have a very simple structure and are easy to understand:

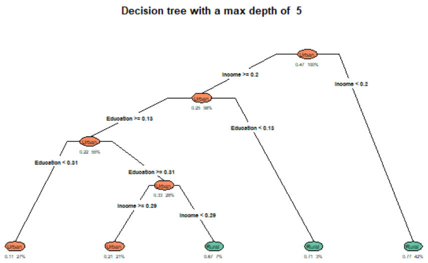


Trees have:

- **Nodes** where splitting decisions are done
- **Branches** following conditions

[WHAT'S IN A TREE?]

Trees have a very simple structure and are easy to understand:

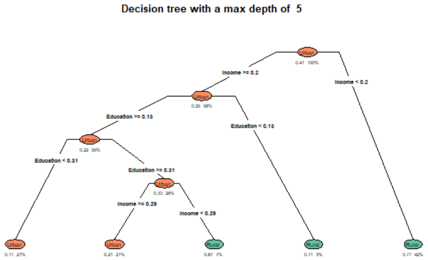


Trees have:

- **Nodes** where splitting decisions are done
- **Branches** following conditions
- **Leaves** are terminal nodes of the classification

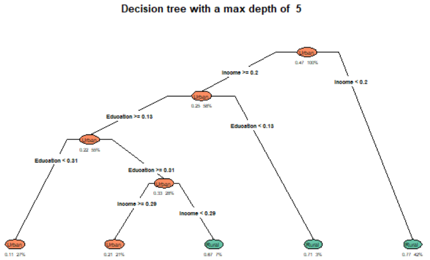
[What's in a tree?]

How to build a tree?



[What's in a tree?]

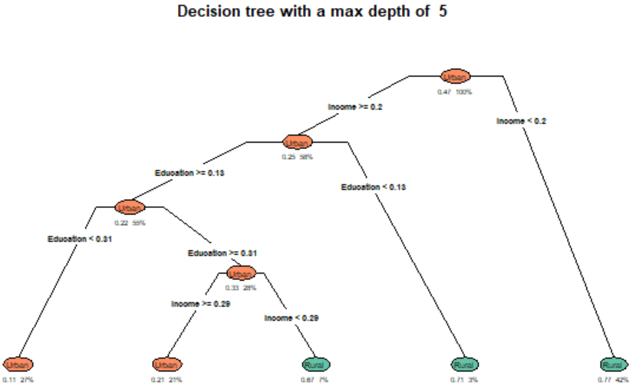
How to build a tree?



- Trees are based on recursive binary splits
- Each node uses a threshold on a variable
- Each node separates the observations in two sets

[Example on a simple tree]

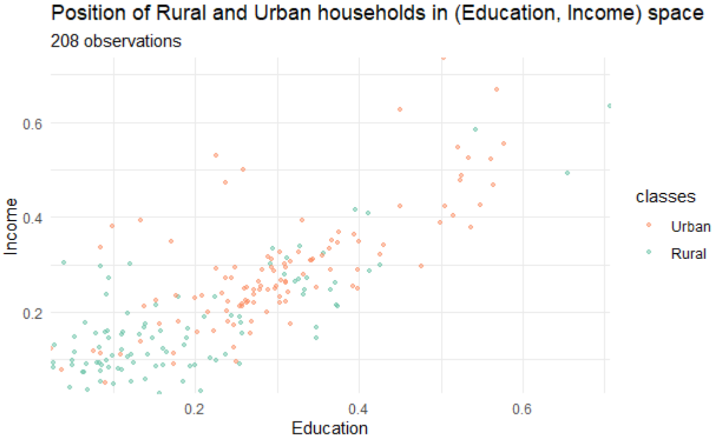
Let us see how this tree is constructed:



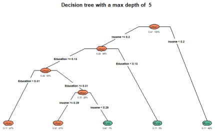
4 nodes leading to final leaves $\hookrightarrow$ Depth = 5

[Example on a simple tree]

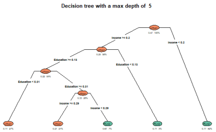
The problem is a 2D space

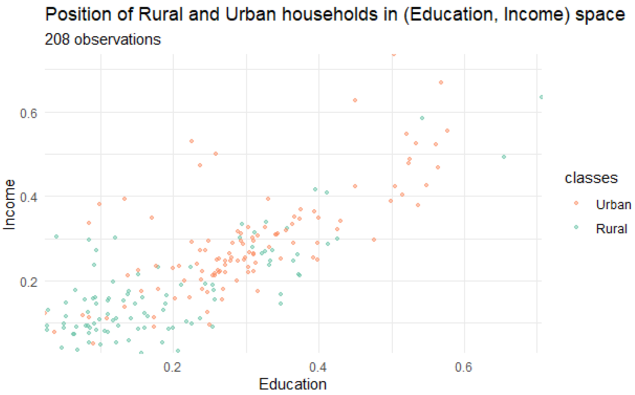


[Example on a simple tree]



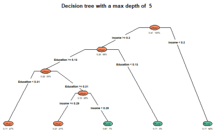
[EXAMPLE ON A SIMPLE TREE]

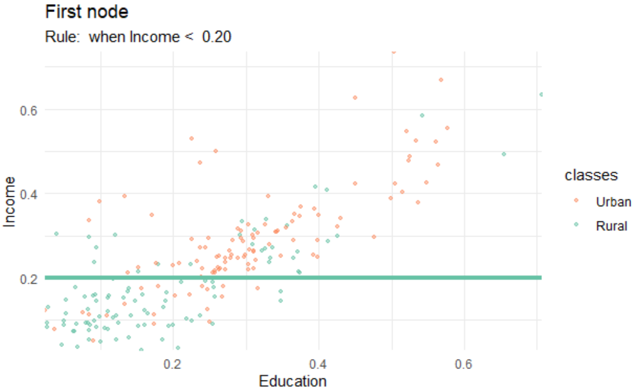




How to split the (Education,Income) space?

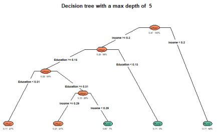
[Example on a simple tree]

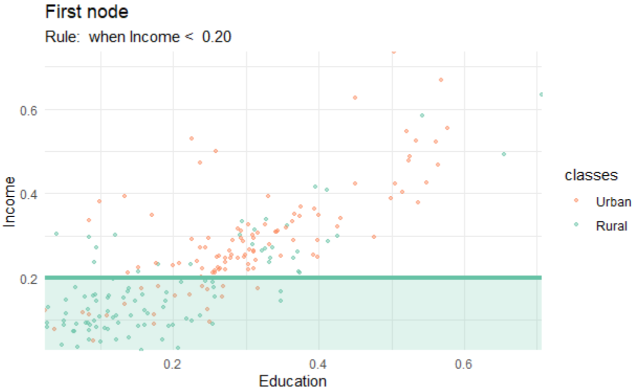




The first boundary decision line

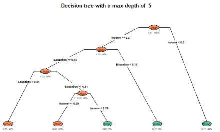
[Example on a simple tree]

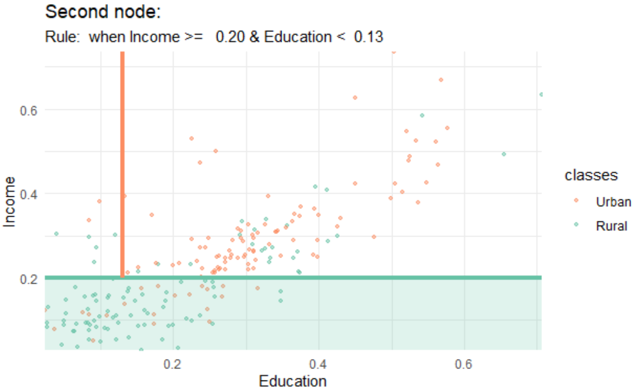




The space below the line is classified as rural

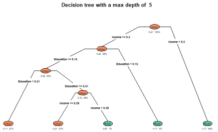
[EXAMPLE ON A SIMPLE TREE]

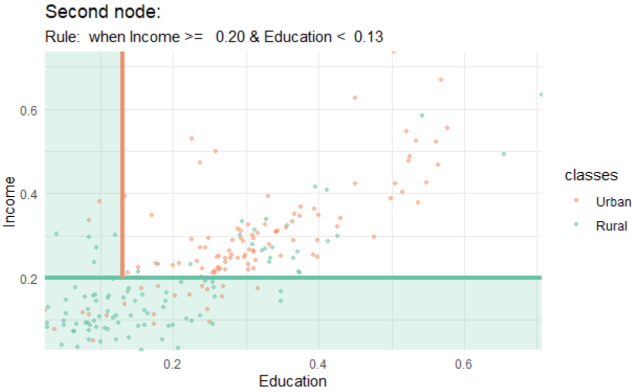




Second boundary decision line

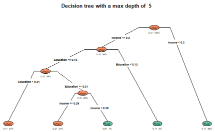
[EXAMPLE ON A SIMPLE TREE]

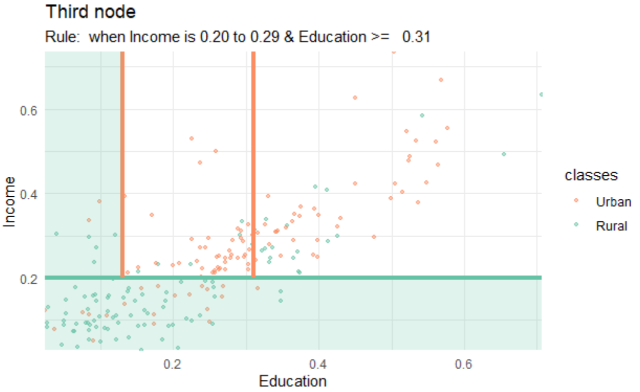




The space on the left of the line is classified as rural

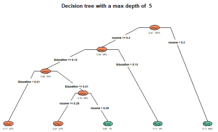
[Example on a simple tree]

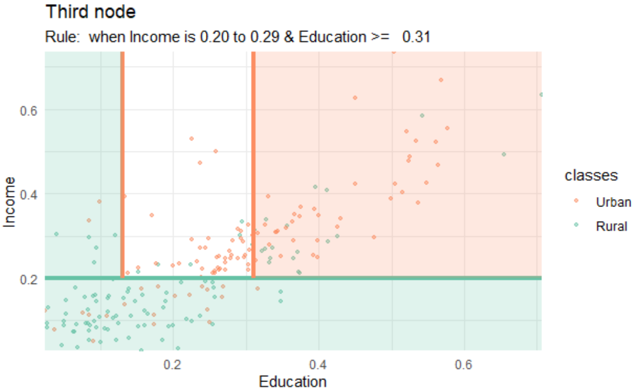




Third boundary decision line

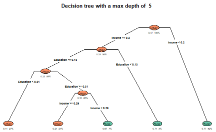
[Example on a simple tree]

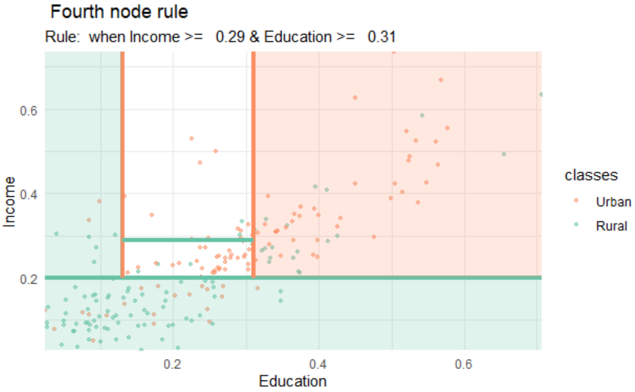




The space on the right of the line is classified as Urban

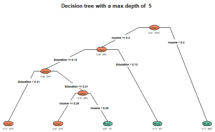
[Example on a simple tree]

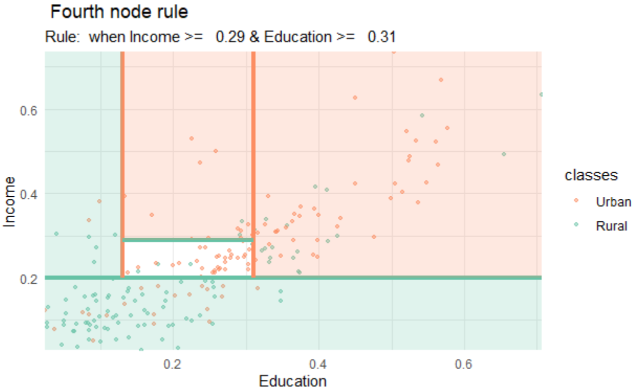




Fourth boundary decision line

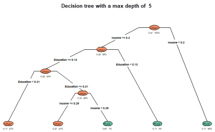
[Example on a simple tree]

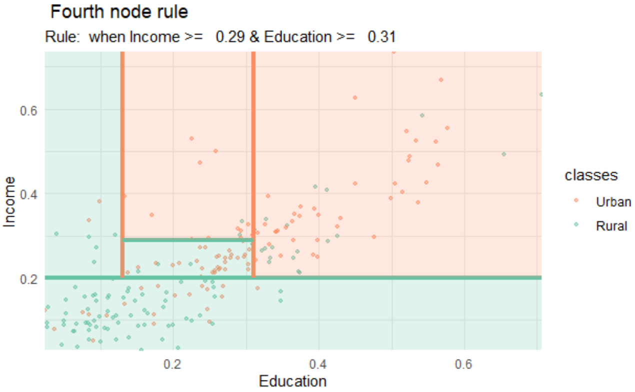




The space above the line is classified as Urban

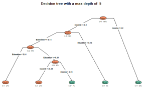
[Example on a simple tree]





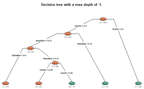
Finally, the remaining space is classified as rural

[How to build a tree?]



We need several tools to build a tree:

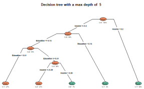
[How to build a tree?]



We need several tools to build a tree:

- A method to choose the decision variable (one per node)

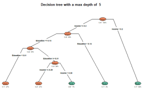
[How to build a tree?]



We need several tools to build a tree:

- A method to choose the decision variable (one per node)
- A criterion to define best threshold

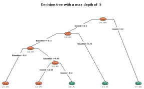
[How to build a tree?]



We need several tools to build a tree:

- A method to choose the decision variable (one per node)
- A criterion to define best threshold
- A criterion to measure the quality of each split

[How to build a tree?]



We need several tools to build a tree:

- A method to choose the decision variable (one per node)
- A criterion to define best threshold
- A criterion to measure the quality of each split
- A way to decide when to stop (the terminal node becomes a leaf)

[Tools to build a tree]

At each node, one can measure the *purity* of each split

[Tools to build a tree]

At each node, one can measure the *purity* of each split

- Misclassification error rate

[Tools to build a tree]

At each node, one can measure the *purity* of each split

- Misclassification error rate
- The Gini coefficient measures the purity in each node κ

$$D_\kappa = \sum_{m=1}^{M} \widehat{p}_{m\kappa}(1 - \widehat{p}_{m\kappa})$$

where $\widehat{p}_{m\kappa}$ is the proportion of class m in node κ.

[Tools to build a tree]

At each node, one can measure the *purity* of each split

- Misclassification error rate
- The Gini coefficient measures the purity in each node κ

$$D_\kappa = \sum_{m=1}^{M} \widehat{p}_{m\kappa}(1 - \widehat{p}_{m\kappa})$$

where $\widehat{p}_{m\kappa}$ is the proportion of class m in node κ.

- The entropy or information:

$$D_\kappa = -\sum_{m=1}^{M} \widehat{p}_{m\kappa} \log \widehat{p}_{m\kappa}$$

[Tools to build a tree]

At each node, one can measure the *purity* of each split

- Misclassification error rate
- The Gini coefficient measures the purity in each node κ

$$D_\kappa = \sum_{m=1}^{M} \widehat{p}_{m\kappa}(1 - \widehat{p}_{m\kappa})$$

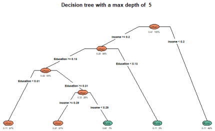
  where $\widehat{p}_{m\kappa}$ is the proportion of class m in node κ.
- The entropy or information:

$$D_\kappa = -\sum_{m=1}^{M} \widehat{p}_{m\kappa} \log \widehat{p}_{m\kappa}$$

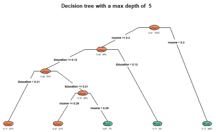
- We expect that there is an Information gain from the splitting

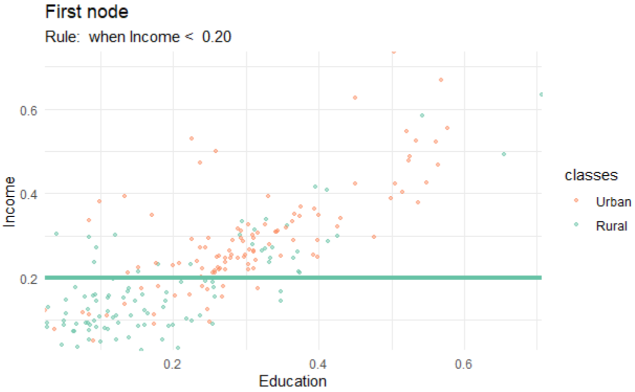
$$\mathit{Information\ Gain} = \mathit{Entropy}_{\,\mathit{Before}} - \mathit{Entropy}_{\,\mathit{After}}$$

[Example on a simple tree]



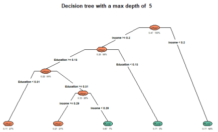
[Example on a simple tree]

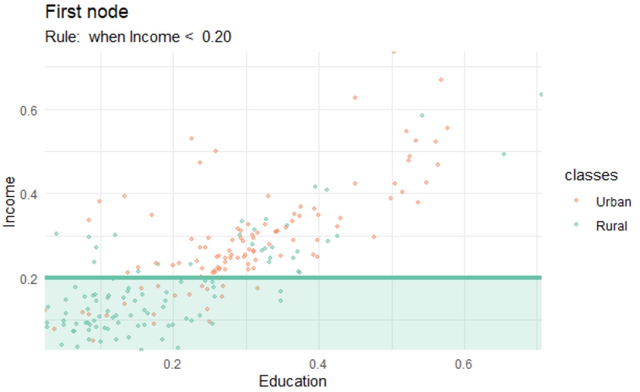




First the boundary decision line

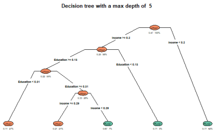
[Example on a simple tree]

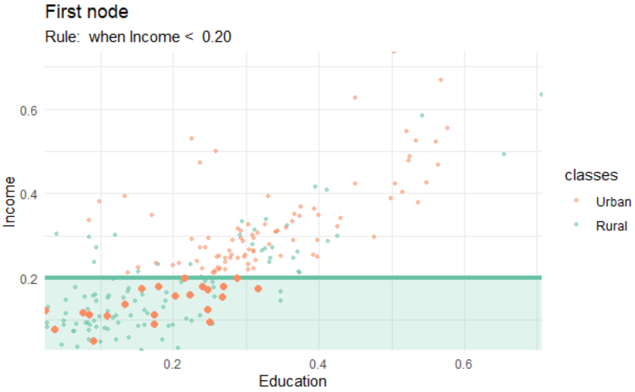




The space below the line is classified as rural

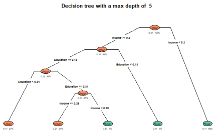
[EXAMPLE ON A SIMPLE TREE]

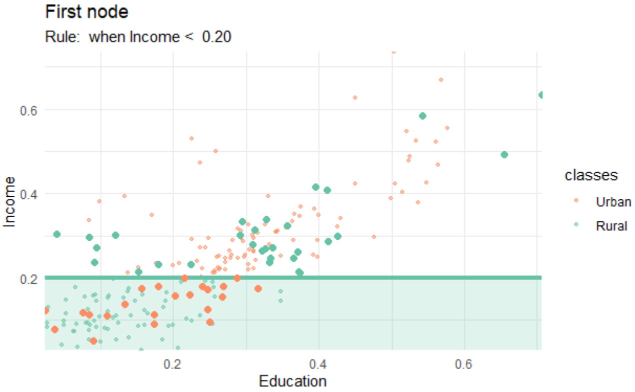




Some "Urban" are classified as rural ↪ Impurity

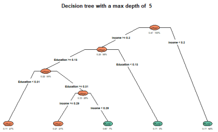
[Example on a simple tree]

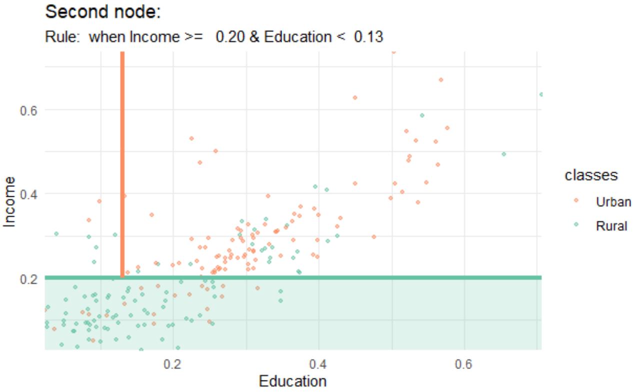




Some "Rural" are classified as Urban $\hookrightarrow$ Impurity

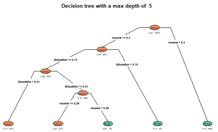
[Example on a simple tree]

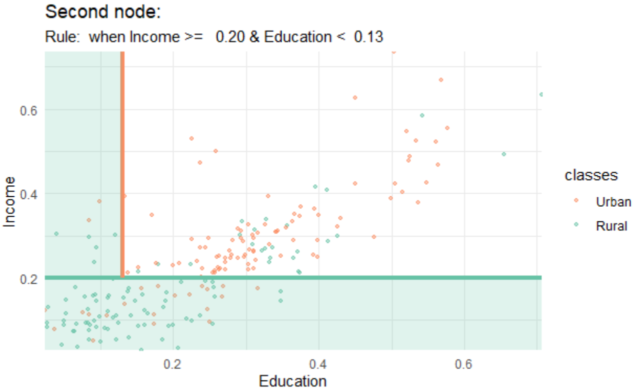




Second boundary decision line

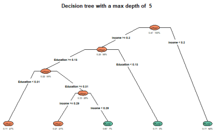
[Example on a simple tree]

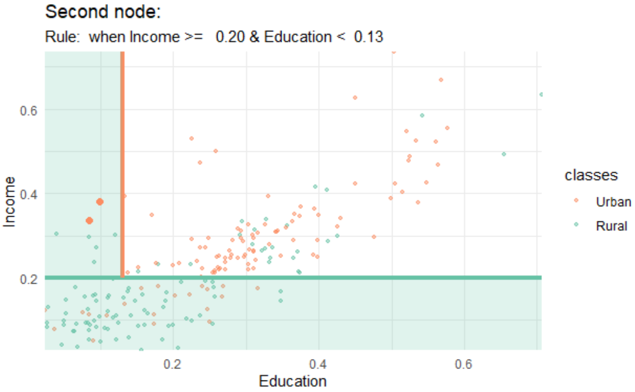




The space on the left of the line is classified as rural

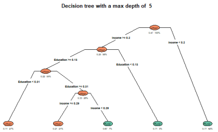
[Example on a simple tree]

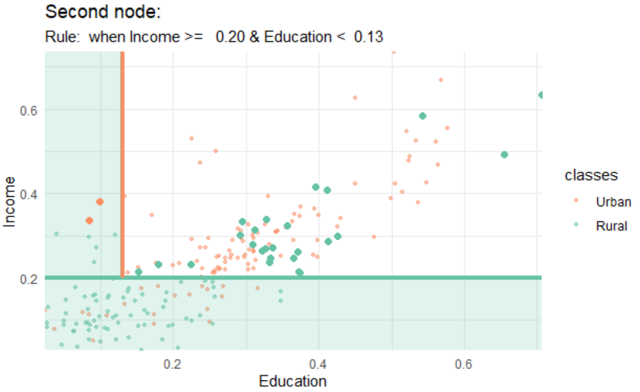




Some "Urban" are classified as Rural

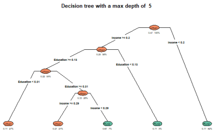
[Example on a simple tree]

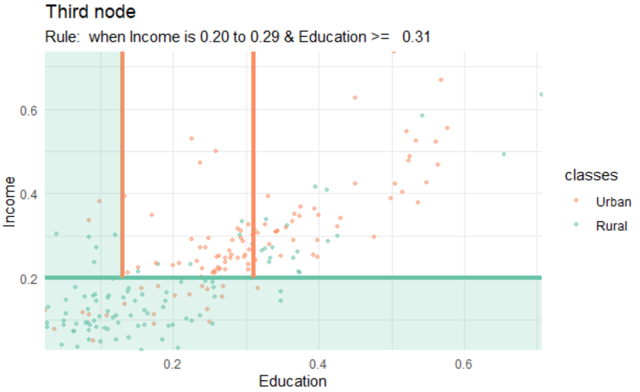




Some "Rural" are classified as Urban

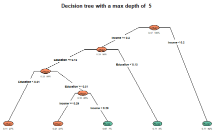
[Example on a simple tree]

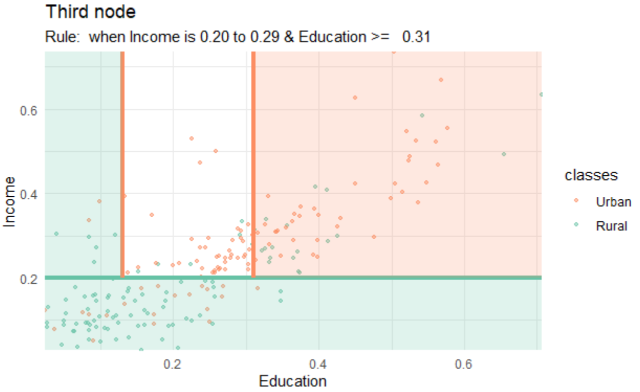




Third boundary decision line

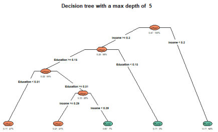
[Example on a simple tree]

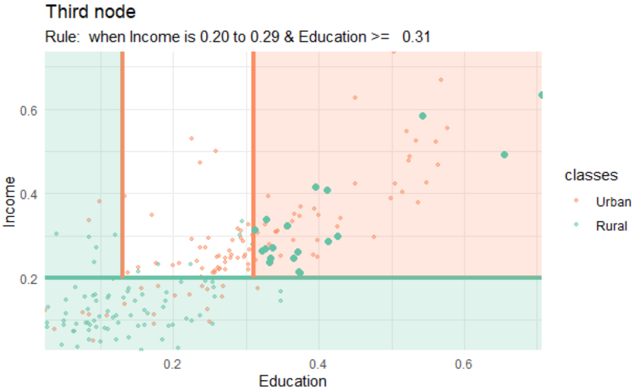




The space on the right of the line is classified as Urban

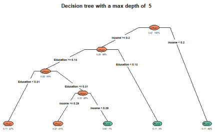
[Example on a simple tree]

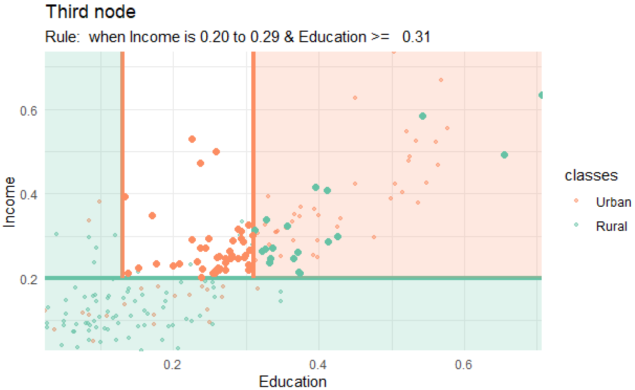




Some "Rural" are classified as Urban

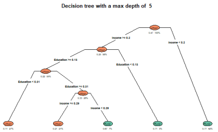
[EXAMPLE ON A SIMPLE TREE]

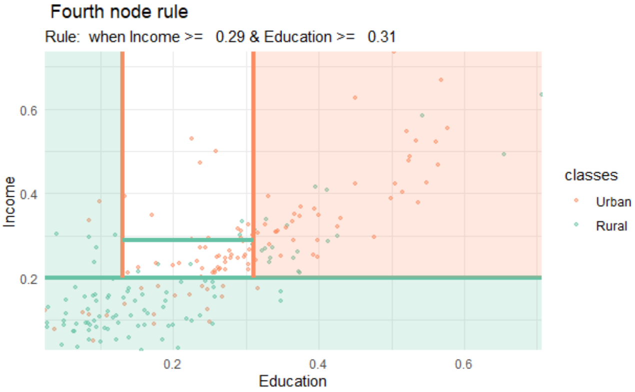




"Urban" seem well classified in the remaining space

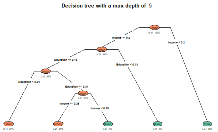
[Example on a simple tree]

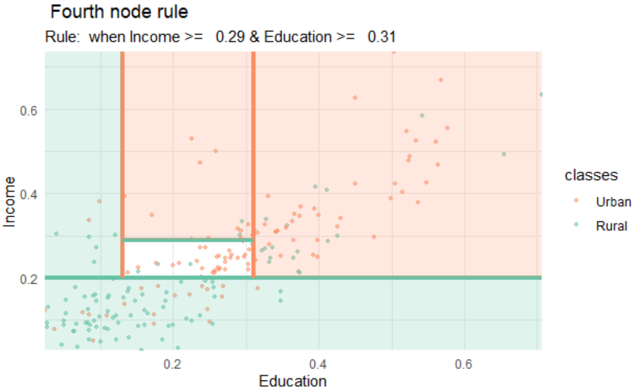




Fourth boundary decision line

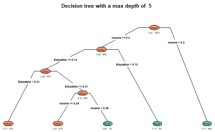
[Example on a simple tree]

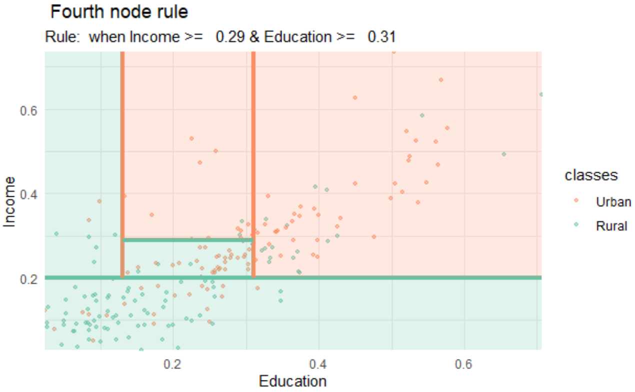




The space above the line is classified as Urban

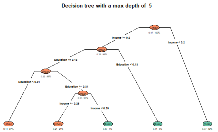
[Example on a simple tree]

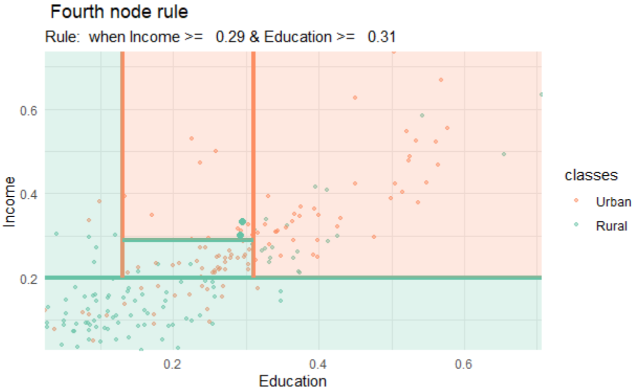




Finally, the remaining space is classified as rural

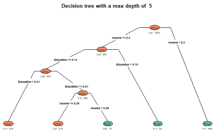
[Example on a simple tree]

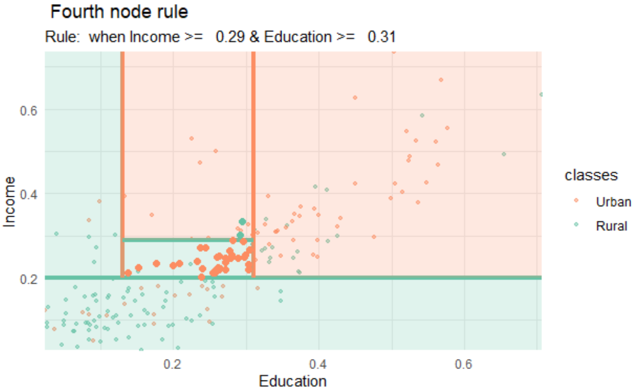




Few "Rural" are classified as Urban

[Example on a simple tree]





Many "Urban" are classified as Rural

[How to build a tree?]

The construction is based on recursive binary splits

[How to build a tree?]

The construction is based on recursive binary splits

- The goal is to increase the quality of the classification at the each stage

[How to build a tree?]

The construction is based on recursive binary splits

- ► The goal is to increase the quality of the classification at the each stage

$\hookrightarrow$ decrease the *impurity* at each node

[How to build a tree?]

The construction is based on recursive binary splits

- The goal is to increase the quality of the classification at the each stage

$\hookrightarrow$ decrease the *impurity* at each node

- The outcome is "one" tree, not the perfect one

[How to build a tree?]

The construction is based on recursive binary splits

- The goal is to increase the quality of the classification at the each stage

$\hookrightarrow$ decrease the *impurity* at each node

- The outcome is "one" tree, not the perfect one
- There are many parameters that can be adjusted

[How to build a tree?]

The construction is based on recursive binary splits

- The goal is to increase the quality of the classification at the each stage

$\hookrightarrow$ decrease the *impurity* at each node

- The outcome is "one" tree, not the perfect one
- There are many parameters that can be adjusted
  - The maximum *depth* of the tree

[How to build a tree?]

The construction is based on recursive binary splits

- The goal is to increase the quality of the classification at the each stage

$\hookrightarrow$ decrease the *impurity* at each node

- The outcome is "one" tree, not the perfect one
- There are many parameters that can be adjusted
  - The maximum *depth* of the tree
  - The number of final leaves

[How to build a tree?]

The construction is based on recursive binary splits

- The goal is to increase the quality of the classification at the each stage

$\hookrightarrow$ decrease the *impurity* at each node

- The outcome is "one" tree, not the perfect one
- There are many parameters that can be adjusted
  - The maximum *depth* of the tree
  - The number of final leaves
  - The impurity balance between classes

[How to build a tree?]

The construction is based on recursive binary splits

- The goal is to increase the quality of the classification at the each stage

$\hookrightarrow$ decrease the *impurity* at each node

- The outcome is "one" tree, not the perfect one
- There are many parameters that can be adjusted
  - The maximum *depth* of the tree
  - The number of final leaves
  - The impurity balance between classes
  - The *complexity parameter*

[How to build a tree?]

The construction is based on recursive binary splits

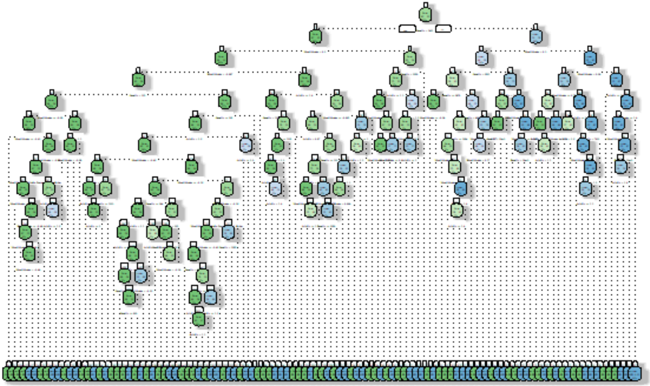
- The goal is to increase the quality of the classification at the each stage

$\hookrightarrow$ decrease the *impurity* at each node

- The outcome is "one" tree, not the perfect one
- There are many parameters that can be adjusted
  - The maximum *depth* of the tree
  - The number of final leaves
  - The impurity balance between classes
  - The *complexity parameter*
  - ...

[Trees can be complex]

Trees can decompose the space in very specific zones.
$\hookrightarrow$ Example with the full set of variables



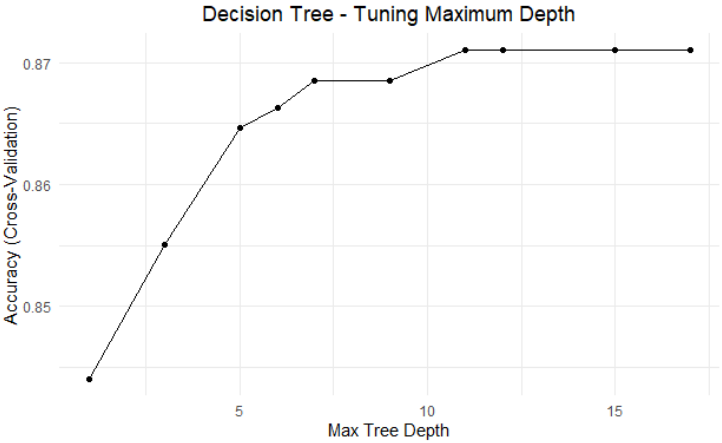
Decision tree with no constrains

[Selecting the **Depth** of a tree]

Using CV, we can select the *maximum depth* parameter

[Selecting the Depth of a tree]

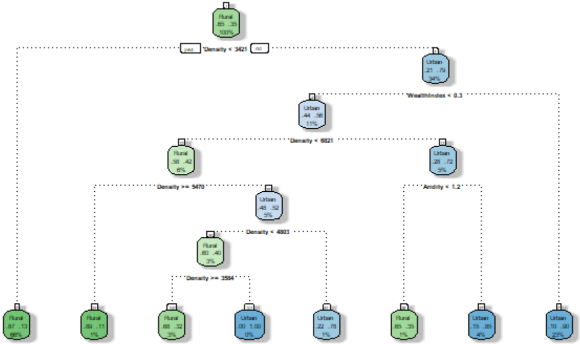
Using CV, we can select the *maximum depth* parameter



[Selecting the **Depth** of a tree]

[SELECTING THE DEPTH OF A TREE]

The resulting tree

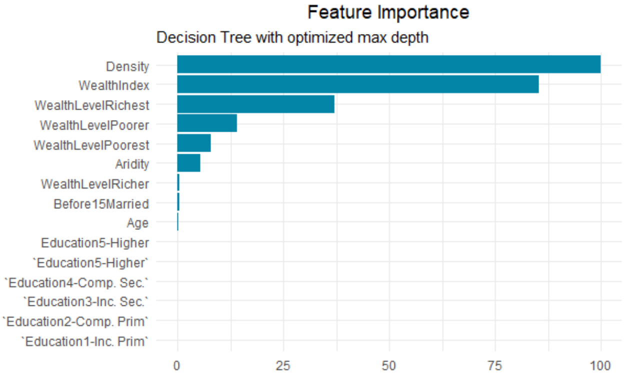


Tree Selected (max depth = 11)

Tree with optimal depth

[Selecting the Depth of a tree]

The resulting tree



Feature importance (also confusion matrix, kappa, etc..)

[Selecting the **Complexity** of a tree]

The complexity of a tree is a parameter C_p governing the trade-off between tree size $|T|$ and its overall accuracy $D(T)$:

$$D_{C_p}(T) = D(T) + C_p \cdot |T|$$

- $D(T) = \sum_{\kappa=1}^{K} D_\kappa$: the total *impurity* of the tree

[Selecting the **Complexity** of a tree]

The complexity of a tree is a parameter C_p governing the trade-off between tree size $|T|$ and its overall accuracy $D(T)$:

$$D_{C_p}(T) = D(T) + C_p \cdot |T|$$

- $D(T) = \sum_{\kappa=1}^{K} D_\kappa$: the total *impurity* of the tree
- $|T|$ is the number of terminal nodes of the tree

[Selecting the **Complexity** of a tree]

The complexity of a tree is a parameter C_p governing the trade-off between tree size $|T|$ and its overall accuracy $D(T)$:

$$D_{C_p}(T) = D(T) + C_p \cdot |T|$$

- $D(T) = \sum_{\kappa=1}^{K} D_\kappa$: the total *impurity* of the tree
- $|T|$ is the number of terminal nodes of the tree

$\hookrightarrow$ A model with $C_p = 0$ will impose no constrains

[Selecting the **Complexity** of a tree]

The complexity of a tree is a parameter C_p governing the trade-off between tree size $|T|$ and its overall accuracy $D(T)$:

$$D_{C_p}(T) = D(T) + C_p \cdot |T|$$

- $D(T) = \sum_{\kappa=1}^{K} D_\kappa$: the total *impurity* of the tree
- $|T|$ is the number of terminal nodes of the tree

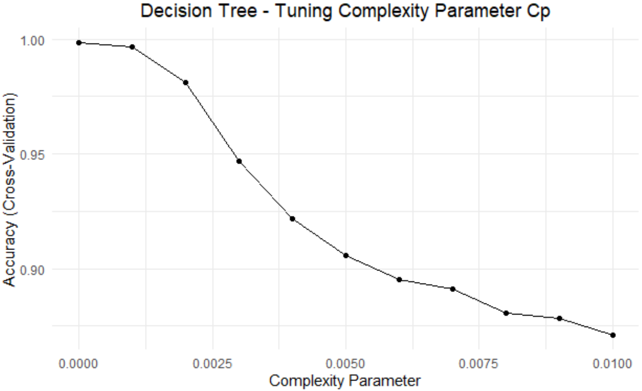
$\hookrightarrow$ A model with $C_p = 0$ will impose no constrains

$\hookrightarrow$ A value of $C_p = 1$ only **one** terminal (and initial) node.

[Selecting the **Complexity** of a tree]

[SELECTING THE **COMPLEXITY** OF A TREE]

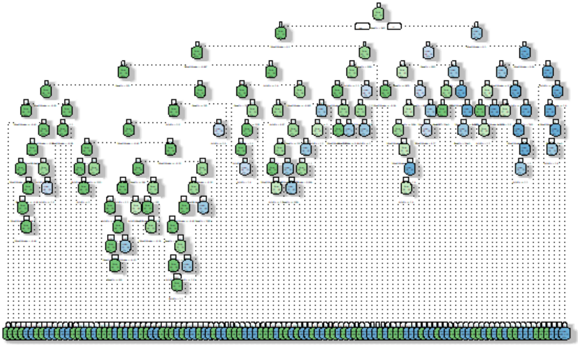
The result:



Grid search

[Selecting the Complexity of a tree]

The result:

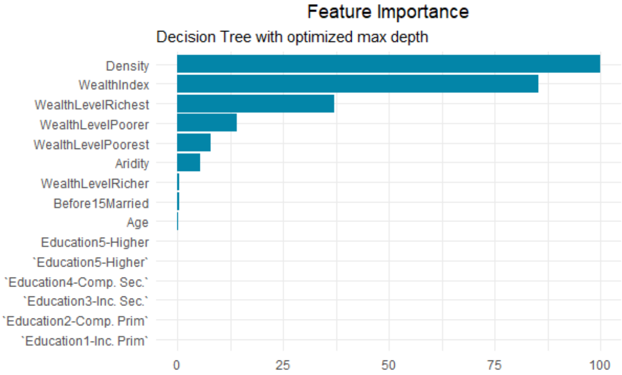


Tree with optimized Cp

Final tree with optimal complexity parameter

[Selecting the Complexity of a tree]

The result:



Feature importance (also confusion matrix, kappa, etc..)

[How to **prune** a tree?]

One can also *prune* a tree

[How to prune a tree?]

One can also *prune* a tree

1. Final trees may be too large and too complex

[How to prune a tree?]

One can also *prune* a tree

1. Final trees may be too large and too complex

$\hookrightarrow$ Risk of overfitting

[How to prune a tree?]

One can also *prune* a tree

1. Final trees may be too large and too complex

$\hookrightarrow$ Risk of overfitting

2. Pruning techniques use the same criteria on each leave

[How to prune a tree?]

One can also *prune* a tree

1. Final trees may be too large and too complex

$\hookrightarrow$ Risk of overfitting

2. Pruning techniques use the same criteria on each leave

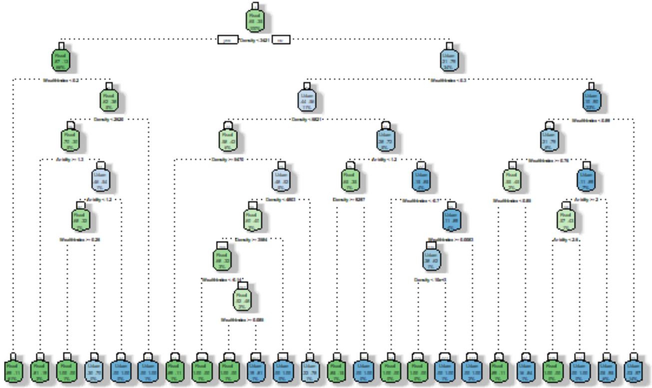
$\hookrightarrow$ Remove least important nodes

[Pruned tree]

After "*pruning*":

[PRUNED TREE]

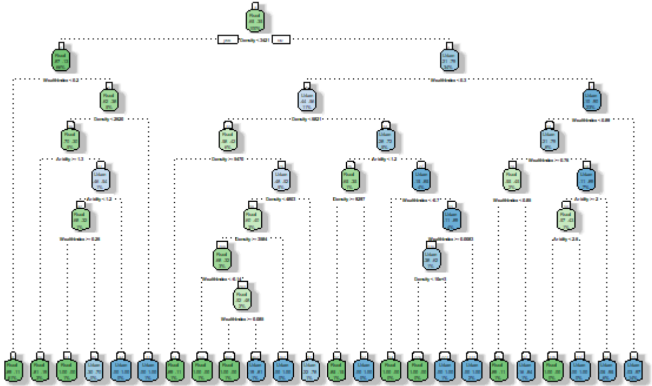
After "*pruning*":



Pruned tree

[PRUNED TREE]

After "*pruning*":



Pruned tree

↪ Easier to interpret & no loss of accuracy

[Quiz time]

[Quiz time]

[Takeaways]

- Trees are simple and easy to interpret

[Takeaways]

- Trees are simple and easy to interpret
- Each node is a split based on a threshold

[Takeaways]

- Trees are simple and easy to interpret
- Each node is a split based on a threshold
- Splits are determined to maximize some measure of accuracy in each strata of the predictors' space

[Takeaways]

- Trees are simple and easy to interpret
- Each node is a split based on a threshold
- Splits are determined to maximize some measure of accuracy in each strata of the predictors' space
- Regression trees apply the same logic, with different criteria and values

[TAKEAWAYS]

- Trees are simple and easy to interpret
- Each node is a split based on a threshold
- Splits are determined to maximize some measure of accuracy in each strata of the predictors' space
- Regression trees apply the same logic, with different criteria and values
- One can select the depth of a tree or its complexity and prune it

[TAKEAWAYS]

- Trees are simple and easy to interpret
- Each node is a split based on a threshold
- Splits are determined to maximize some measure of accuracy in each strata of the predictors' space
- Regression trees apply the same logic, with different criteria and values
- One can select the depth of a tree or its complexity and prune it
- Trees are very specific, not robust and prone to overfitting

[TAKEAWAYS]

- Trees are simple and easy to interpret
- Each node is a split based on a threshold
- Splits are determined to maximize some measure of accuracy in each strata of the predictors' space
- Regression trees apply the same logic, with different criteria and values
- One can select the depth of a tree or its complexity and prune it
- Trees are very specific, not robust and prone to overfitting

↪ There are powerful methods using many trees...