Creating Regression Tree

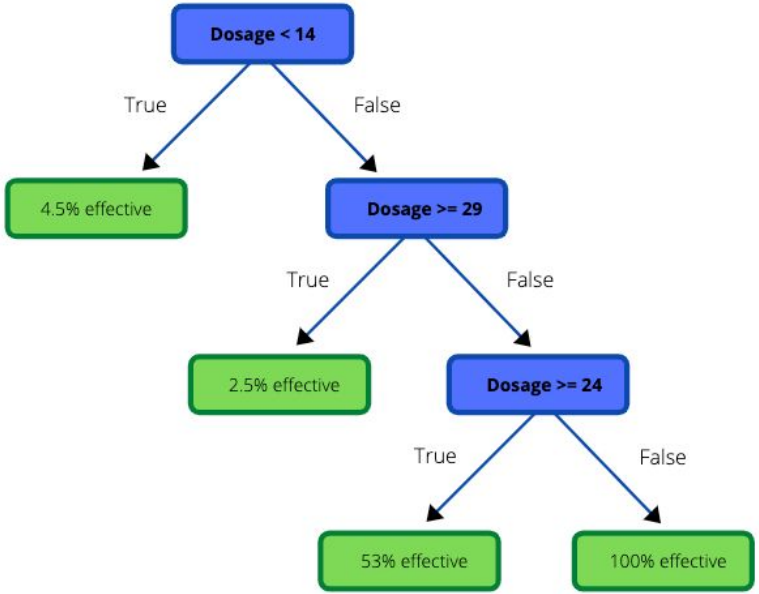
By Carmen Azorín and Sara Cerro

Definition of Regression Tree

Regression trees are a type of decision tree where the target variables have continuous values rather than categorical labels at the leaves.

They employ adjusted split selection and stopping criteria tailored for continuous data.

Utilizing a regression tree allows for an explanation of decisions, identification of potential events, and visualization of possible outcomes, aiding in determining the optimal decision. These models segment the data into subsets through the use of nodes, branches, and leaves.



How to build Regression Trees – Pruning

- **Objective**: Identify a subtree that minimizes the test error rate.
- **Estimating Test Error**:
  - Use cross-validation (CV) to estimate the test error rate of a given subtree.
- **Challenge**:
  - Evaluating CV for every possible subtree is computationally expensive due to the large number of subtrees.
- **Solution**:
  - Instead of evaluating all possible subtrees, consider a sequence of trees defined by a nonnegative tuning parameter, α.
  - This sequence balances model complexity and error, enabling more efficient subtree selection.

By following this approach, we effectively prune the tree and find the optimal subtree with minimal computational effort.

Algorithm for Building a Regression Tree I

1. **Grow a Large Tree**:
    - Use recursive binary splitting on the training data to grow a large tree.
    - Stop only when each terminal node has fewer than a specified minimum number of observations.
2. **Cost Complexity Pruning**:
    - Apply cost complexity pruning to the large tree.
    - This yields a sequence of subtrees indexed by a tuning parameter α.

Algorithm for Building a Regression Tree II

3. **Choose Optimal α Using K-Fold Cross-Validation**:
    - Divide the training data into KKK folds.
    - For each fold:
        1. Perform Steps 1 and 2 on the training data excluding the kkk-th fold.
        2. Calculate the mean squared prediction error on the kkk-th fold for different values of α.
    - Average the errors for each α across all folds.
    - Select the α that minimizes the average error.
4. **Return the Optimal Subtree**:
    - The subtree corresponding to the selected α from Step 3 is returned as the final model.

By following these steps, you ensure that the regression tree is both accurate and generalizes well to new data.

Creating a Regression Tree with SAS

The HPSPLIT procedure is a high-performance procedure that builds tree-based statistical models for classification and regression. The procedure produces classification trees, which model a categorical response, and regression trees, which model a continuous response.

PROC HPSPLIT *<options>*;
CLASS *variable... </options>*;
CODE FILE=*filename*;
GROW *criterion </ options>*;
ID *variables*;
MODEL *response <(response-options)> = variable <variable...>*;
OUTPUT *output-options*;
PARTITION *<partition-options>*;
PERFORMANCE *performance-options*;
PRUNE *prune-method <(prune-options)>*;
RULES FILE=*filename*;

Regression Tree Example – Birthweight data

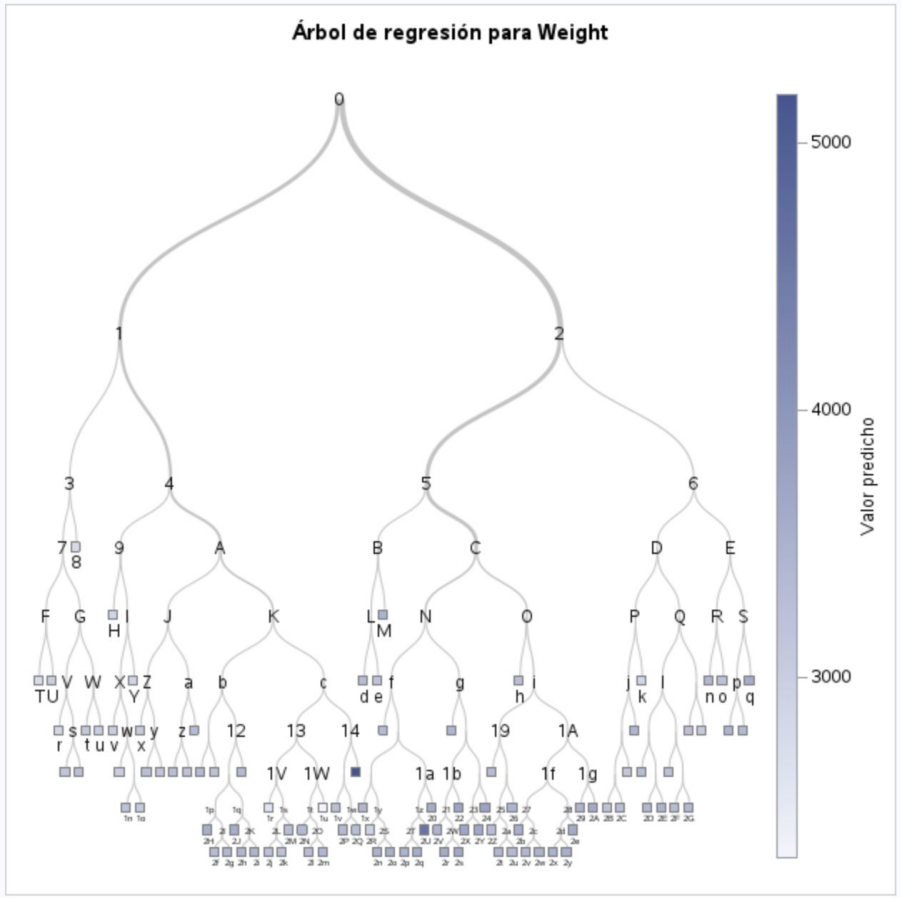
Predicting low birth weight is essential because of the related health issues. Traditional methods like OLS and logistic regression fall short: they either struggle to model the lower quantiles accurately or lose important details. Simplifying birth weight into categories can result in misleading comparisons between significantly different weights. SAS provides birthweight data that is useful for illustrating PROC HPSPLIT

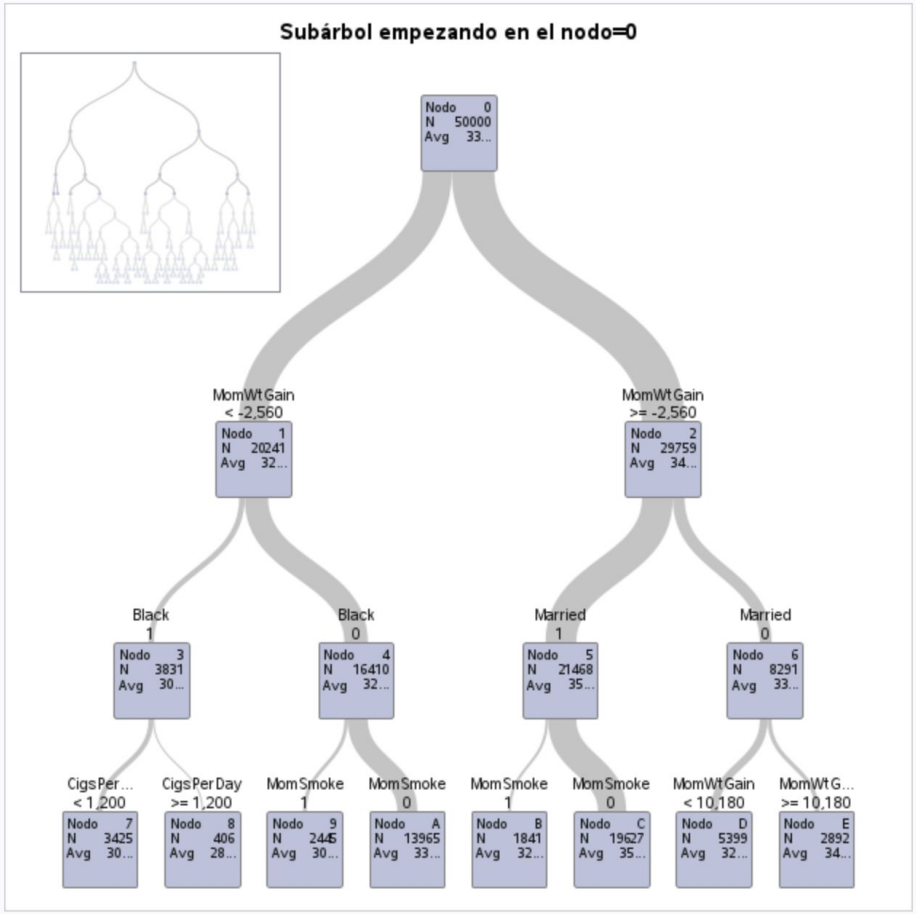
We add an ID variable to the data set provided by SAS (this will be useful later):

```
data trees.birthweight;
    set sashelp.bweight;
    count + 1;
run;
```

Then running the basic HPSPLIT is fairly straightforward:

```
proc hpsplit data=trees.birthweight seed=123;
    class black boy married momedlevel momsmoke;
    model weight=black boy married momedlevel
    momsmoke momage momwtgain visit cigsperday;
    output out=trees.bwregout;
run;
```


Árbol de regresión para Weight
Valor predicho
5000
4000
3000


Subárbol empezando en el nodo=0
Nodo 0
N 50000
Avg 33...
MomWtGain
< -2,560
MomWtGain
>= -2,560
Nodo 1
N 20241
Avg 32...
Nodo 2
N 29759
Avg 34...
Black
1
Black
0
Married
1
Married
0
Nodo 3
N 3831
Avg 30...
Nodo 4
N 16410
Avg 32...
Nodo 5
N 21468
Avg 35...
Nodo 6
N 8291
Avg 33...
CigsPer...
< 1,200
CigsPerDay
>= 1,200
MomSmoke
1
MomSmoke
0
MomSmoke
1
MomSmoke
0
MomWtGain
< 10,180
MomWtG...
>= 10,180
Nodo 7
N 3425
Avg 30...
Nodo 8
N 406
Avg 28...
Nodo 9
N 2445
Avg 30...
Nodo A
N 13965
Avg 33...
Nodo B
N 1841
Avg 32...
Nodo C
N 19627
Avg 35...
Nodo D
N 5399
Avg 32...
Nodo E
N 2892
Avg 34...

Regression Tree Example

The first plot provides only limited information; its primary value lies in presenting an overall view of the tree and serving as a reference for further analysis.

The second plot is easy to interpret. It starts with 5,000 observations in one node.

The first split is based on whether the mother's weight gain was more or less than 2.560 kg.

- For women with less weight gain (left node), the next split is by race (Black or not).
- For those with more weight gain (right node), the next split is by marital status.

This demonstrates the power of decision trees to model interactions that regular regression cannot.

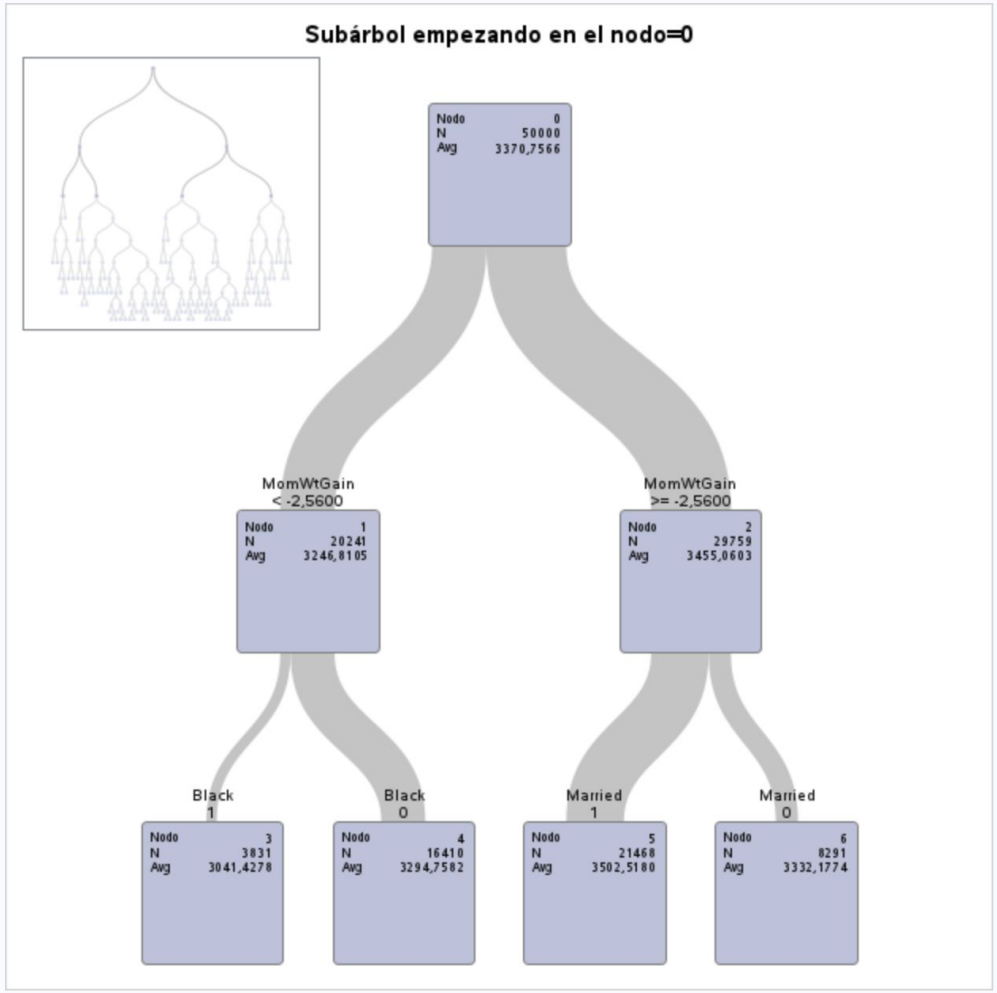
Regression Tree Example

The default output of HPSPLIT doesn't provide detailed means for each node.

We can introduce three new options:

- FRACPREC and PREDICTORPREC: to obtain frequencies and means in each node.
- NODES: to obtain a table of the final nodes with relevant information.

```
proc hpsplit data=trees.birthweight seed=123 plots=zoomedtree(nodes=("0") depth=2 fracprec=4
        predictorprec=4) nodes;
    class black boy married momedlevel momsmoke;
    model weight=black boy married momedlevel momsmoke momage momwtgain visit
        cigsperday;
    output out=bwregout;
run;
```


Subárbol empezando en el nodo=0
Nodo 0
N 50000
Avg 3370,7566
MomWtGain
< -2,5600
MomWtGain
>= -2,5600
Nodo 1
N 20241
Avg 3246,8105
Nodo 2
N 29759
Avg 3455,0603
Black
1
Black
0
Married
1
Married
0
Nodo 3
N 3831
Avg 3041,4278
Nodo 4
N 16410
Avg 3294,7582
Nodo 5
N 21468
Avg 3502,5180
Nodo 6
N 8291
Avg 3332,1774

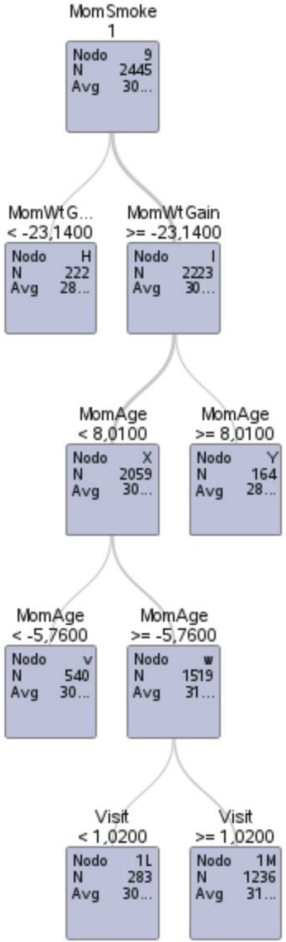
Regression Tree Example

Last options seen are inside the option PLOTS = ZOOMEDTREE, useful to examine specific areas of the tree more closely.

The larger tree shows that node 8 is a final node and doesn't require further exploration, but you can focus on the details of node 9, for instance, using this option.

```
proc hpsplit data=trees.birthweight seed=123 plots=zoomedtree(nodes=("9") depth=4 fracprec=4
        predictorprec=4) nodes;
    class black boy married momedlevel momsmoke;
    model weight=black boy married momedlevel momsmoke momage momwtgain visit
        cigsperday;
run;
```

Subárbol empezando en el nodo=9



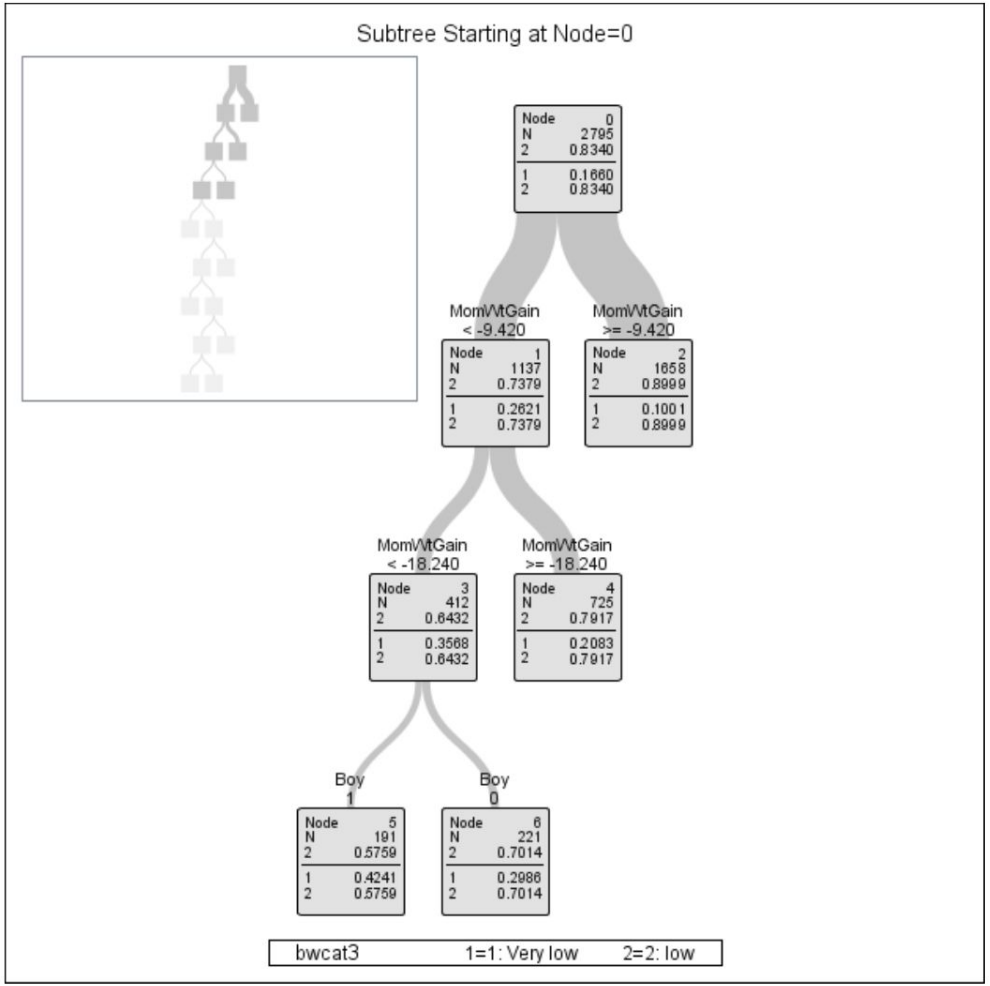
Regression Tree Example

One issue is that the tree's node labeling uses base 62 (0-9, A-Z, a-z). While this results in shorter labels on the tree, it requires recoding for the tables.

```
data trees.bwregout;
    set bwregout;
    retain possdigs
        '0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz';

    do power=0 to 100 while(62**power<=_NODE_);
    end;
    left=_NODE_;
    length node $10;
    node=' ';
    i=0;

    do power=power-1 to 0 by -1;
        i+1;
        r=int(left/(62**power));
        substr(node, i, 1)=substr(possdigs, r+1, 1);
        left=left-(62**power)*r;
    end;
run;
```


Subtree Starting at Node=0
Node 0
N 2795
2 0.8340
1 0.1660
2 0.8340
MomWtGain
< -9.420
MomWtGain
>= -9.420
Node 1
N 1137
2 0.7379
1 0.2621
2 0.7379
Node 2
N 1658
2 0.8999
1 0.1001
2 0.8999
MomWtGain
< -18.240
MomWtGain
>= -18.240
Node 3
N 412
2 0.6432
1 0.3568
2 0.6432
Node 4
N 725
2 0.7917
1 0.2083
2 0.7917
Boy
1
Boy
0
Node 5
N 191
2 0.5759
1 0.4241
2 0.5759
Node 6
N 221
2 0.7014
1 0.2986
2 0.7014
bwcat3
1=1: Very low
2=2: low