Data Mining and Machine Learning

LECTURE 12 Decision Trees

Dr. Edgar Acuna
Department of Mathematics

Universidad de Puerto Rico- Mayaguez

academic.uprm.edu/eacuna

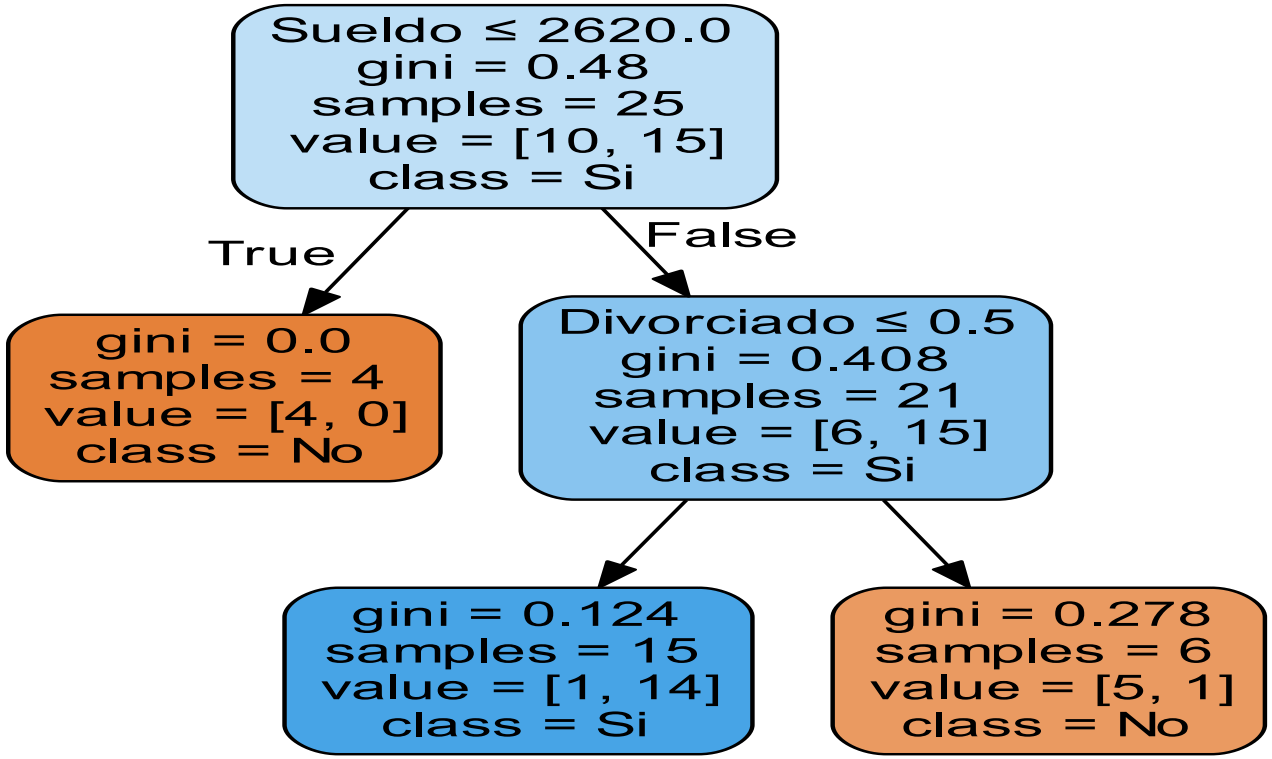
The use of decisión trees originated in the social sciences with the research work of Sonquist and Morgan (1964) and, Morgan and Messenger (1979) at the Survey Research Center of the Institute for Social Research from the University of Michigan. But it was only applied to regression. The program THAID (Theta Automatic Interaction Detection) due to Sonquist, Baker y Morgan (1971) extended the use of decisión trees to classification problems.

In Statistics, Kass (1980) introduced a recursive non-binary tree algorithm called CHAID (Chi-square automatic interaction detection). Later, Breiman, Friedman, Olshen and Stone (1984) introduced a new algorithm for the construction of trees and it was applied to regression and classification. The method was named CART (Classification and regression trees). Almost at the same time the inductive process using trees started to be used by the Machine Learning community [Michalski, (1973), Quinlan (1983)]. In the field of Pattern Recognition , Henrichon and Fu (1969) wrote an article in nonparametric partitioning that is quiete similar to decisión trees, but it uses hyperplanes that are not parallel to the coordinates axes .

Example: Predicting a bank decision to offer a loan to a customer for buying a car

Sexo	Familia	CasPropia	AnosEmpleo	Sueldo	StatustMarital	Prestamo
Hombre	3	No	17	2500	Soltero	No
Mujer	5	Si	10	3000	Casado	Si
Mujer	4	No	15	2000	Viudo	No
Hombre	3	Si	16	2800	Soltero	Si
Hombre	6	Si	11	4000	Viudo	Si
Mujer	4	Si	26	3200	Soltero	Si
Mujer	2	Si	14	1800	Soltero	No
Hombre	5	Si	10	3750	Casado	Si
Hombre	6	No	18	2970	Divorciado	No
Hombre	4	Si	12	3350	Divorciado	No
Hombre	1	No	23	1950	Soltero	No
Mujer	2	Si	25	2740	Soltero	Si
Mujer	3	No	7	3100	Soltero	Si
Hombre	5	Si	5	3845	Divorciado	No
Hombre	3	No	13	3200	Casado	Si
Mujer	3	Si	9	2800	Soltero	No
Hombre	2	No	6	3200	Soltero	Si
Hombre	3	Si	7	3815	Viudo	Si
Mujer	2	Si	11	2980	Divorciado	No
Hombre	4	Si	15	2850	Viudo	Si
Mujer	1	No	6	3125	Divorciado	No
Hombre	1	No	8	3500	Soltero	Si
Hombre	4	No	22	4500	Divorciado	Si
Hombre	2	Si	10	3200	Casado	Si
Hombre	3	Si	9	3000	Casado	Si

Example: Predicting a bank decision to offer a loan to a customer for buying a car



Notice that the split on the categorical variable is not uderstandable. This is due to the fact that scikit-learn works only with numerical attributes

Algorithms for decision trees

C4.5. Introduced por Quinlan (1993) in the Machine Learning community . It is a descendant of ID3 (Quinlan, 1986).

CHAID (Chi-square automatic interaction detection), was introduced by Kass (1980) and it is derivated from THAID (see “A sequential search program for the analysis of nominal scale dependent variables” by Morgan and Messenger, 1973). The critrerion to split nodes is based on the $\chi 2$ statistical test.

NewId. (Boswell, 1990). It is a descendant of ID3 (Quinlan, 1986)

CART. Introduced by Breiman et al. (1984), properly it is binary decisión tree algorithm. There exists a similar versión called IndCART, which is available in the package IND distributed by NASA. The Decision tree algorithm used in scikit-learn is based on CART. RPART (Recursive Partitioning) is a versión of CART which is available at R.

Bayesian Trees: It is based on the Bayesian techniques for splitting nodes. (Buntine 992). Available in the package IND distributed by NASA.

CN2. Introduced by Clark and Niblett (1989).

Construction of a decisión tree

A decisión tree partitions the sample space of predictors features in a subset of hyper-rectangles and in each of them it fits a simple model, say y=c, where y represents the response.

The construction of a decisión tree is based on four elements:

a) A set Q of binary questions having the form { $x \in A$?} where A is a subset of the sample space of the variable x.

b) The method used for partitioning the node.

c) The strategy used to grow the tree.

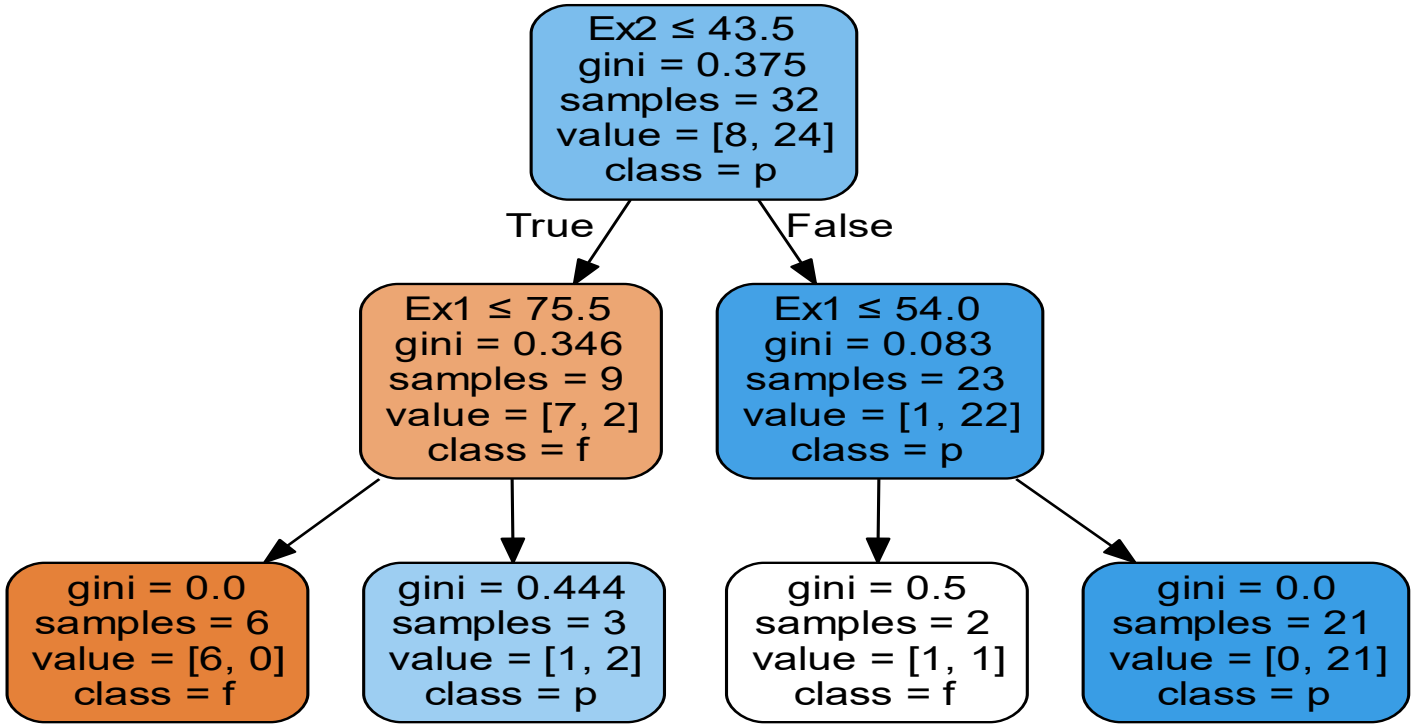
d) The assignment of each terminal node to a value of the response variable (regression) or to a class (classificación).

The main differences between the algorithms to construct trees are in the rule for partitioning the nodes, the strategy to prune the trees and the treatment of missing values.

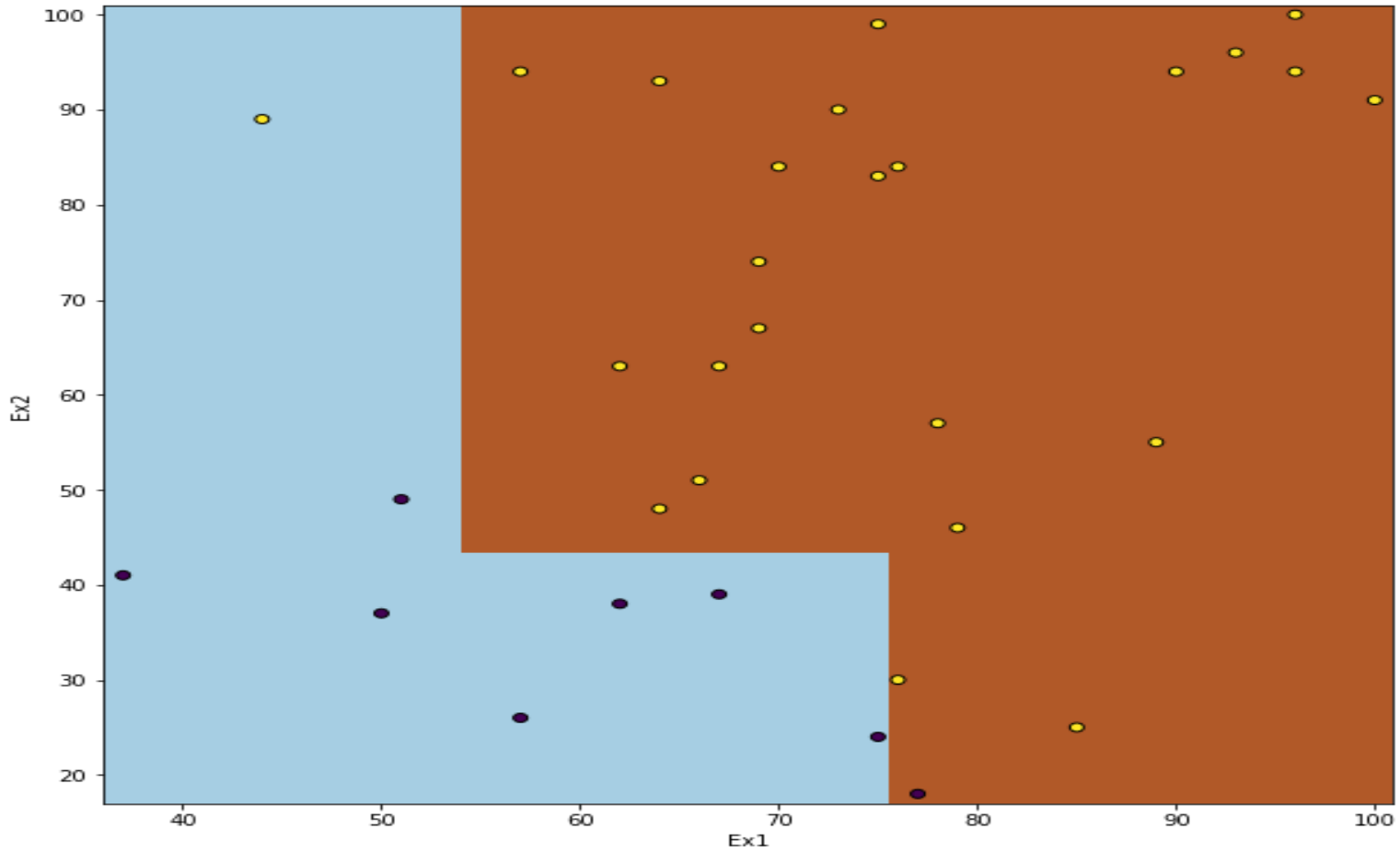
Example: Predicting the final grade in a class based on the scores in E1 and E2

```
df=pd.read_csv("http://academic.uprm.edu/eacuna/eje1dis.csv")
#extracting the matrix of predictors and the column corresponding to classes
y=df['Nota']
X=df.iloc[:,0:2]
#Applying the decisión tree algorithm
modeltree = tree.DecisionTreeClassifier(max_depth=2)
modeltree = modeltree.fit(X,y)
modeltree = tree.DecisionTreeClassifier(max_depth=2)
modeltree = modeltree.fit(X,y)
#Finding the predictions
pred=modeltree.predict(X)
print pred
['p' 'p' 'p' 'p' 'p' 'p' 'p' 'p' 'p' 'p' 'p' 'f' 'p' 'p' 'p' 'p' 'p' 'p' 'p' 'p' 'p' 'p' 'p' 'f' 'f' 'f' 'f' 'f' 'f' 'p' ‘f’]
# There are two instances with a wrong prediction
```

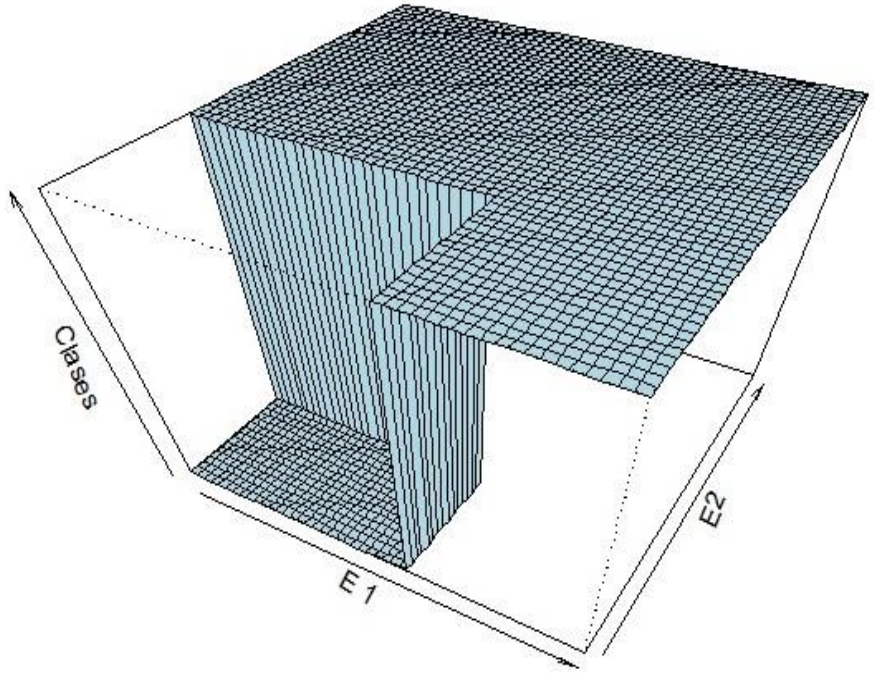
Example: Predicting the final grade in a class based on the scores in E1 and E2



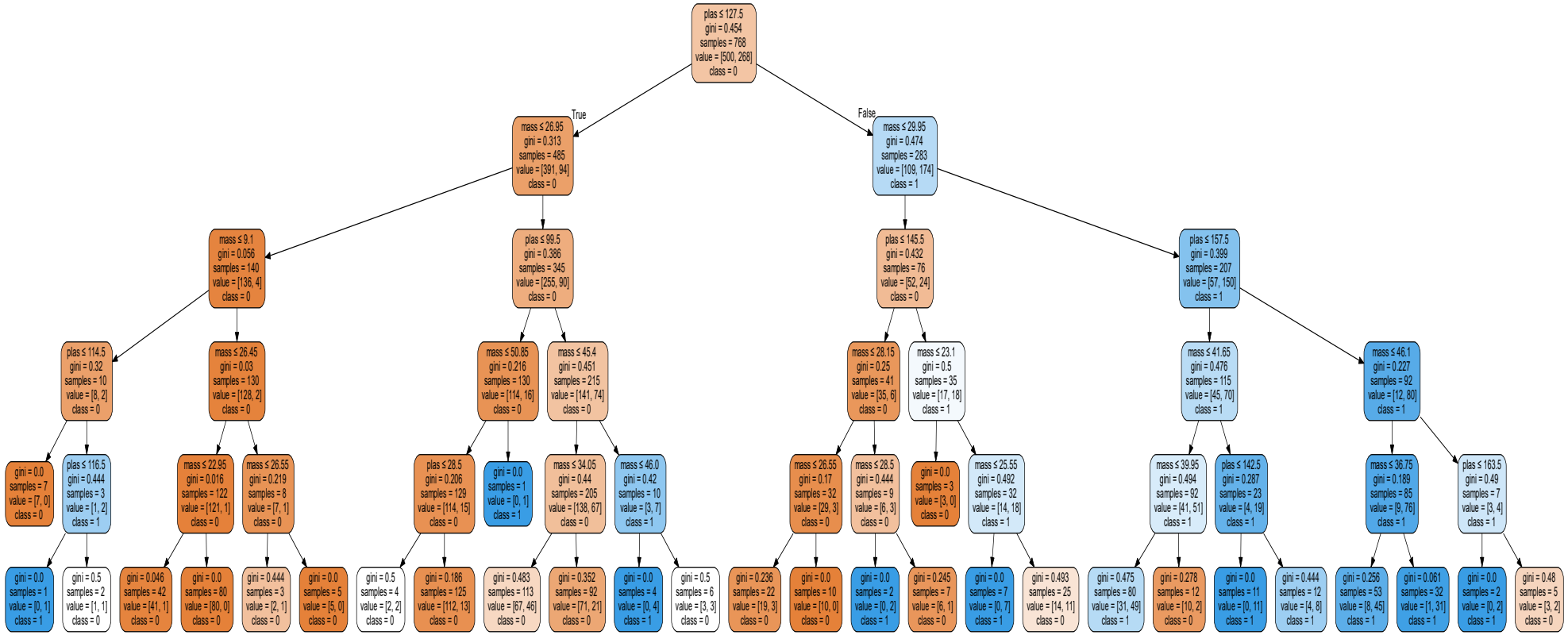
Partitioning of the sample space of predictors



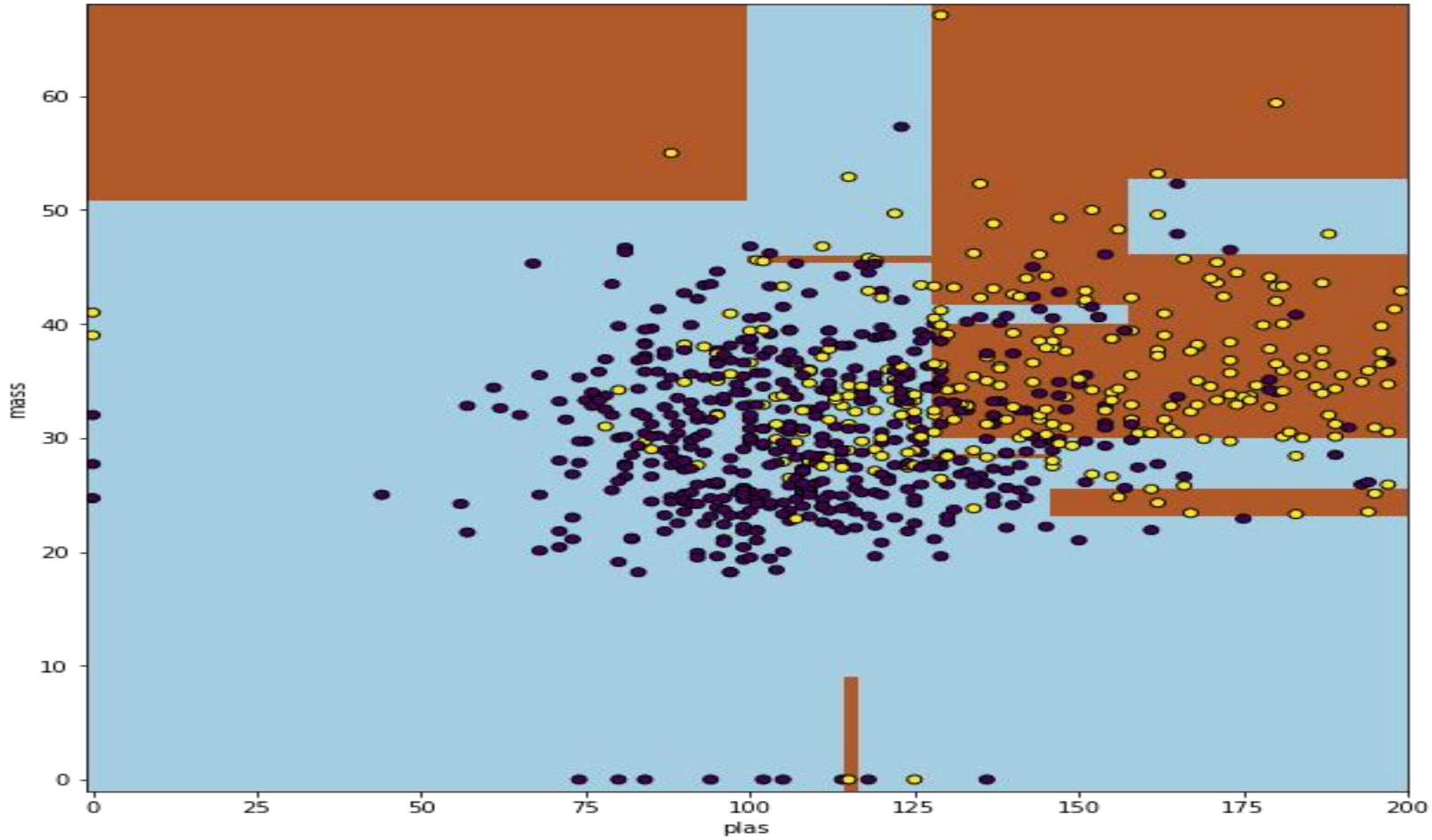
Surface response for the Decision Tree from the example



Example: Decision Tree for Diabetes using only two predictors: plas and mass



Partitioning of the sample space corresponding to the previous tree



I-The set Q of binary questions

- Suppose that the vector of predictors features is of the form $\mathbf{x}=(x_1,\ldots..x_p)$, where some of the features x_i are discrete and others are continuous. Then, the set Q of binary questions in each node must have the following characteristics:

  a) Each division of a node depends on the value of a single predictor variable.

  b) If the feature x_k is continuous, then Q includes questions of the form $\{Es\ x_k \leq c\}$, where c is any real number. Usually c is the midpoint between two consecutive values of the feature.

I-The set Q of binary questions

c) If the feature x_k is categorical taking values $\{b_1,b_2,\ldots\ldots b_m\}$ then Q includes questions of the form $\{ x_k \in A?\}$ where A is any subset of $\{b_1,b_2,\ldots\ldots b_m\}$. In total 2^{m-1}-1 questions can be considered.

For instance, if x2, x3 and x4 are continuous predictors and x1 is categorical assuming values 0, 1 y 2, then Q includes questions of the form:

Is x3≤ 4.5?

Is x4≤ -1.4?

Is x1= 0 ó 1?

- Also a node can be partitioned in more than two divisions, but this is not recommended because the dataset will be divided much faster leaving few data for the partition of the subsequent nodes.

II-Procedure for nodes partitioning

The main idea is try to get children nodes purer than their parent node. In the regression case, the partition of a node t of the tree T is in such way that the children node give a sums of squares error less than the is parent node. In classification, the best partition yields children nodes that distinguish better the classes than their parent node.

In a classification tree, let $p(s)=\{\#\ i\leq N: Xi\in s\}/N$ be te proportion of instances in the node s, and

$p(j/s)=\{\#\ i\leq N: Xi \in s\ y\ Yi=j\}/\{\#\ i\leq N: Xi \in s\}$

the proportion of instances in the node s that belong to the j-th class , (j=1,….,J), where J is the number of classes.

II-Procedure for nodes partitioning

The impurity index of the node is defined as $i(s)=\varphi(p(1/s),p(2/s),\ldots p(J/s))$ where φ is a impurity function, which must satisfy some properties.

Then, the rule for partitioning the node t is a follows:

Determine the right children node t_R and the left children t_L such that decrease of the impurity index , given by

$\Delta i(t)= i(t)-\{p(t_L)i(t_L)+p(t_R)i(t_R)\}$

Is maximized.

Impurity measures

In trees for classification, the following impurity measures can be used

a) The ***Gini coefficient***. *Given the node t is defined as*

$$i_G(t) = \sum_j \sum_k p(j/t)\,p(k/t) = \sum_{j=1}^{J} p(j/t)(1-p(j/t))$$

For two classes (J=2) , this becomes

$$i_G(t) = 2p(1-p),$$

where p is the proportion of instances belonging to the first class at the node.

The Gini coefficient assigns each instance of a node to the j-th class with probability p(j/t) instead of classifying all the instances of a node to the class with the largest number of instances.

Impurity measures

b) The ***Cross Entropy*** **or Deviance or Impurity of Information** defined as

$$i_E(t) = -\Sigma_j\, p(j/t)\log[p(j/t)]$$

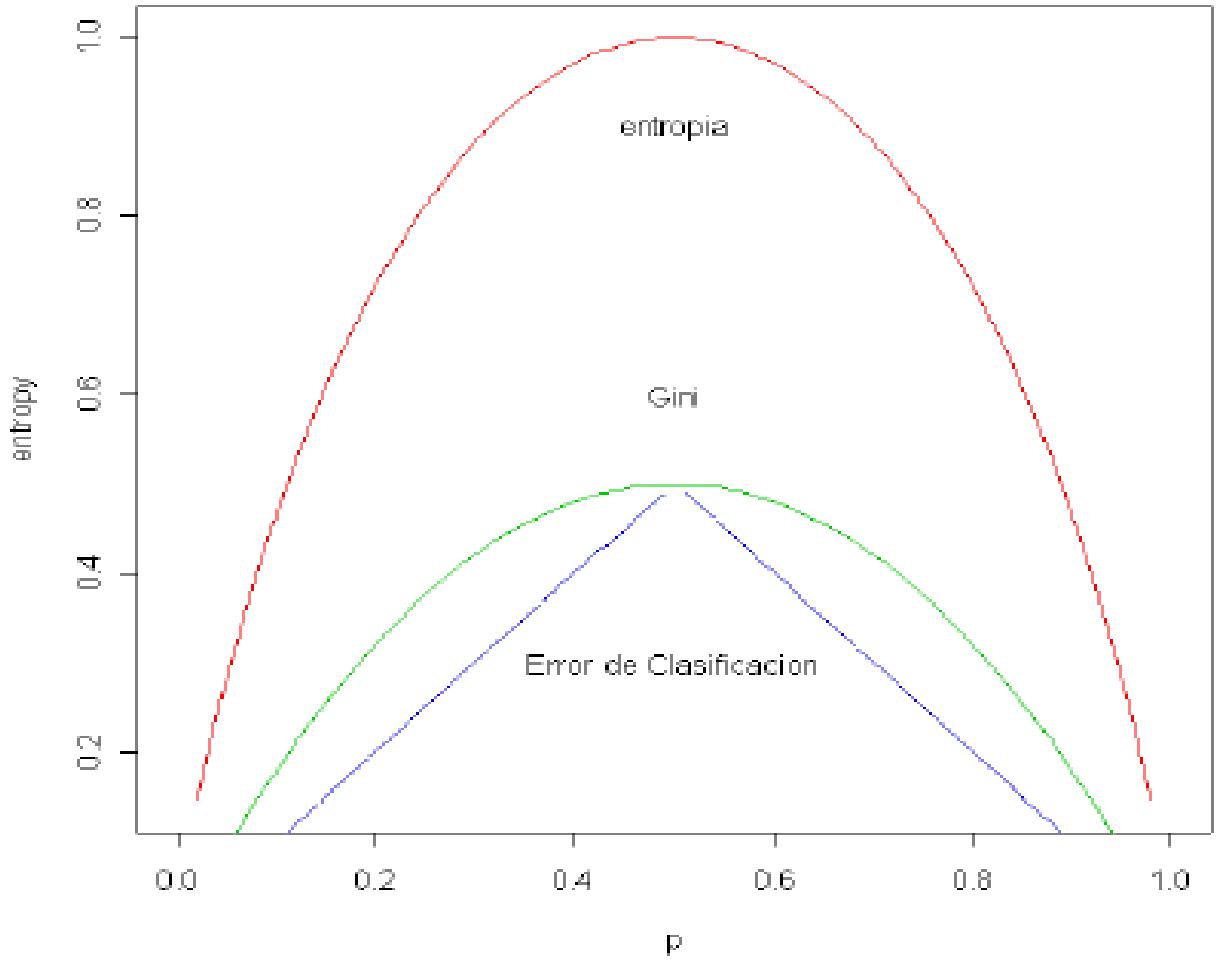
where the logarithm is taken is basis 2. When, entropy is applied to continuous features one can use natural logarithm. For two classes, one gets

$$i_E(t) = -p*\log(p) -(1-p)*\log(1-p)$$

In regression, the Deviance is equivalent to the sum of squares of errors and, it is given by the double of the logarithm of the likelihood function.

Scikit-learn uses by default the Gini coefficient as impurity measure, but choosing criterion="entropy" we can switch to the cross-entropy measure.

- c) The ***Missclasifficación error rate***, given by
- $i_{MC}(t)=1-max_j p(j/t)$. For two classes is given by: $i_{MC}(t)=1-max(p,1-p)$



Example 1: Computing the impurity. Loan dataset

In the loan example, if we consider the first partition in the two children's nodes:

t_R:Status Marital= Soltero , there are 4 subjects not receiving loan and 6 receiving loan.

t_L: Status Marital= Casado, Viudo o Divorciado, there are 6 subjects not receiving loan and 9 receiving loan.

The impurity of t_R would be:

Entropy -4/10*log2(4/10)-6/10*log2(6/10)=.9709

Gini coefficient: 2(4/10)(6/10)=48/100=.48

Missclassification error rate:1-max(4/10,6/10)=4/10=.4

The impurity of t_L would be:

Entropy: -6/15*log2(6/15)-9/15*log2(9/15)=.9709

Gini coefficient: 2(6/15)(9/15)=108/225=.48

Missclassification error rate:1-max(6/15,9/15)=6/15=.4

Example 1: Computing the impurity (cont)

The impurity of the root node, where15 subjects received loan and 10 did not received it, is:

Entropy= -(10/25)*log2(10/25)-(15/25)*log2(15/25)=.9709

Gini=2*10/25*15/25=.48

Missclassification error rate:1-max(10/25,15/25)=.4

Therefore, the decrease of impurity with each function will be:

Δi(t)= .9709-(10/25*.9709+15/25*.9709)=0(Entropy)

Δi(t)= .48-(10/25*.48+15/25*.48)=0(Gini)

Δi(t)= .4-(10/25*.4+15/25*.4)=0 (ME rate)

However, the partition in the children nodes t_R: Sueldo < 2620 (4: No and 0: Yes) and t_L: Sueldo>= 2620 (6: No and 15: Yes), will produce the greatest decrease of impurity. Here, we will show the decrease using the entropy measure.

Δi(t)= .9709-(4/25*0+21/25*.8631)=.2459

In fact, this is the best partition one can get.

Example 2: Computing the impurity. Toy example

In the example to predict the final grade, suppose that our first node is obtained by splitting at E2=47. Then the children nodes would be:

t_R:E2<47, where 3 students pass and 7 fail and.
t_L:E2>=47, where 21 students pass and 1 fail.

Then,
The impurity of t_R with each measure will be :
Entropy -3/10*log2(3/10)-7/10*log2(7/10)=.8812
Gini coefficient: 2(3/10)(7/10)=.42
Missclassification error rate:1-max(3/10,7/10)=3/10=.3

The impurity of t_L with each measure will be : :
Entropy: -21/22*log2(21/22)-1/22*log2(1/22)=.2667
Gini coefficient i: 2(21/22)(1/22)=.086
Missclasification error rate:1-max(21/22,1/22)=1/22=.045

Example 2: Computing the impurity(cont)

The impurity of the root node, where 8 students failed the class and 24 students passed according to the three measures is given by

Entropy= -(8/32)*log2(8/32)-(24/32)*log2(24/32)=.8112

Gini=2*8/32*24/32=.375

Missclassification error rate:1-max(8/32,24/32)=8/32=.25

Therefore, the decrease of impurity with each function will be

Δi(t)= .8112-(10/32*.8812+22/32*.2667)=.3524(Entropia)

Δi(t)= .375-(10/32*.42+22/32*.086)=.184 (Gini)

Δi(t)= .25-(10/32*.3+22/32*.045)=.125 (ME)

However la partition at the split E2=43.5 will give the greatest decrease. Here we will show only the decrease using the entropy as impurity measure

Δi(t)= .8112-(9/32*.764+23/32*.2580)=.4108.

In fact, this is the best partition.

Impurity measures for trees in regression

For decision trees applied to regression problems, the following two measures can be applied

i) The variance

$$i(t) = \sum_{j \in t} \frac{(y_j - \bar{y}_t)^2}{n_t}$$

Where $\bar{y}_t$ is the mean of the response variable y at the node t.

ii) The median absolute deviation

$$i(t) = \sum_{j} \frac{|y_{j-\bar{Y}_t}|}{n_t}$$

III-Criteria to stop the growth of a tree

The growth of a tree ends when it is imposible to continue, That is,

1) There is only one instance in each of the children nodes
2) All the instances in each children node belong to the same class.
3) The maximum number of levels ("depth") of the tree set by the user has been reached.

The largest tree usually creates overfitting, that is it follows a lot of the trend of the data. The last partitions of the tree cause the overfitting. Therefore pruning the tree is required.

III-Criteria to stop the growth of a tree (cont)

The DecisionTreeClassifer function available at scikit-learn has several option to control the growth of the tree.

min_samples_split : ***(default=2).*** The minimum number of samples required to split an internal node:

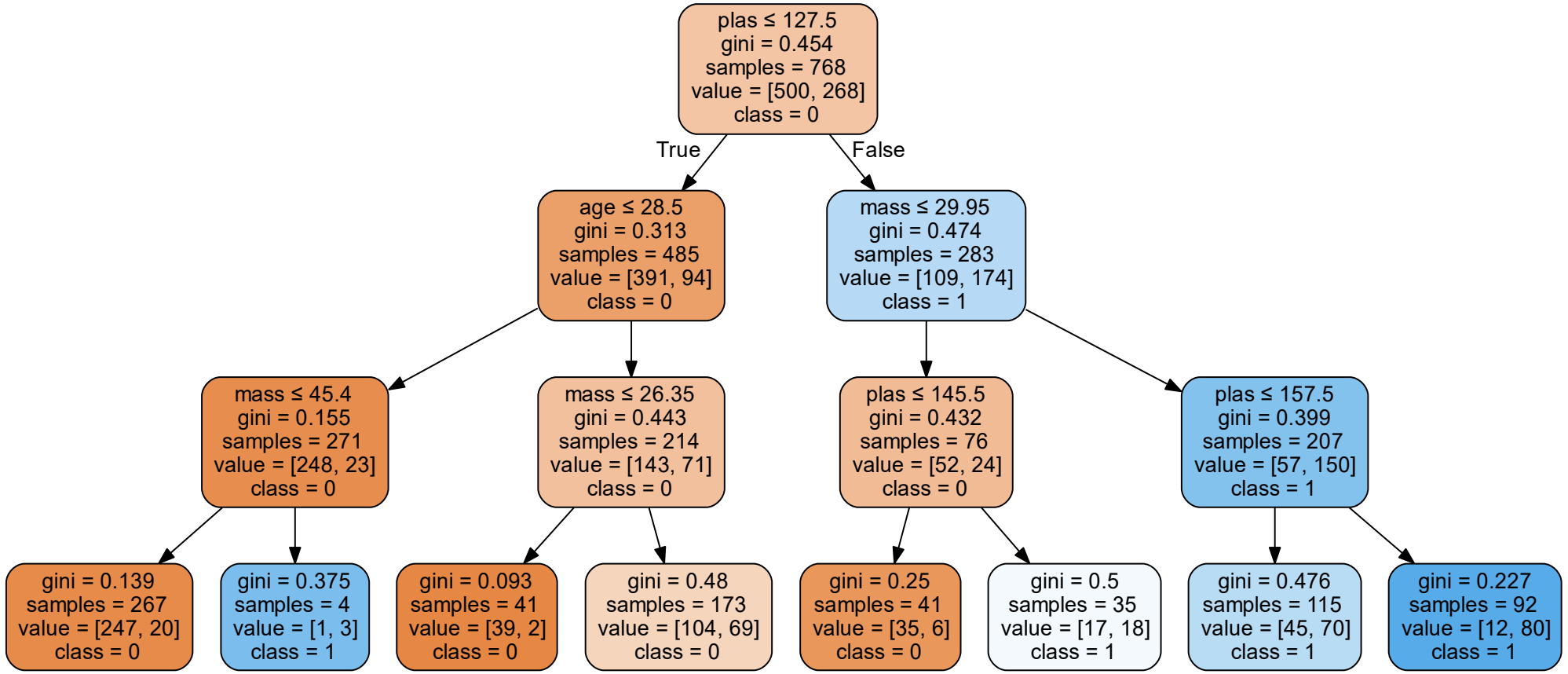
min_samples_leaf : ***int, float, optional (default=1).***The minimum number of samples required to be at a terminal node.

min_impurity_decrease : ***float, optional (default=0.)*** A node will be split if this split induces a decrease of the impurity greater than or equal to this value.

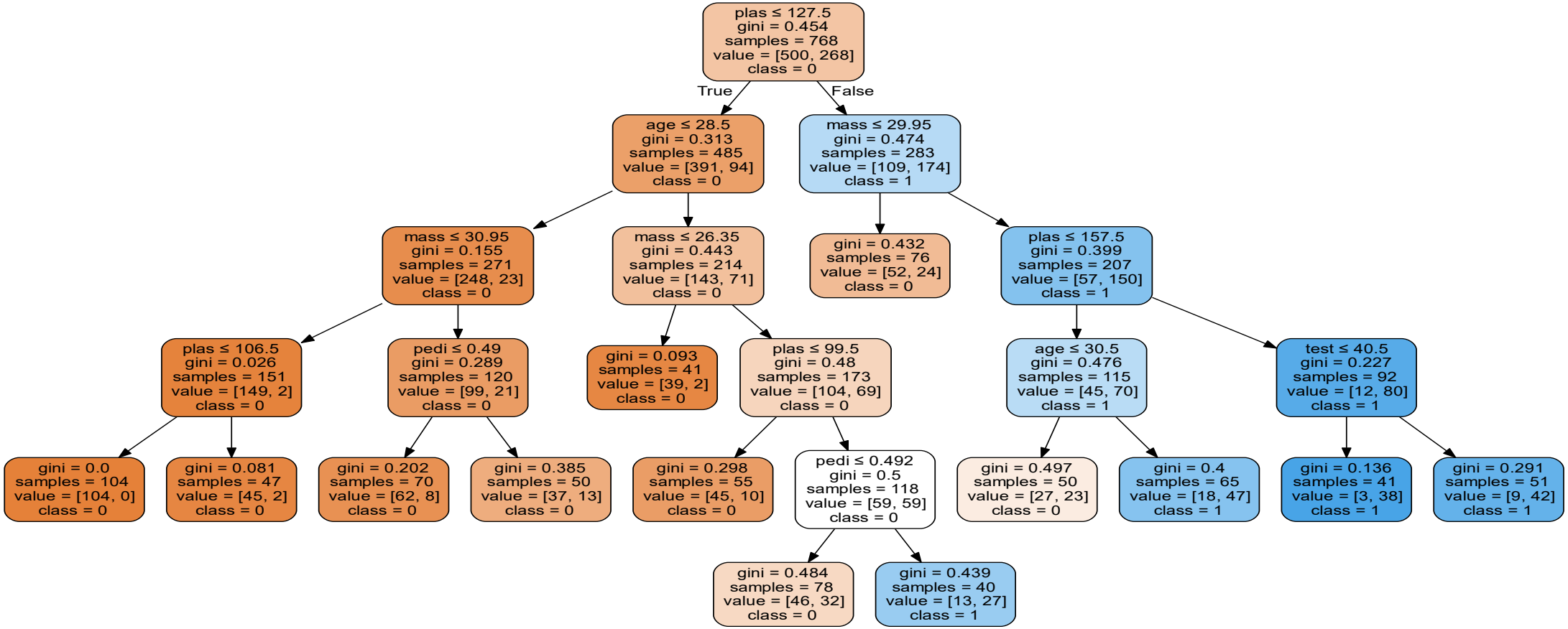
max_leaf_nodes : ***int or None, optional (default=None)*** Grow a tree with max_leaf_nodes in best-first fashion.

max_depth : ***int or None, optional (default=None).***The maximum depth of the tree.

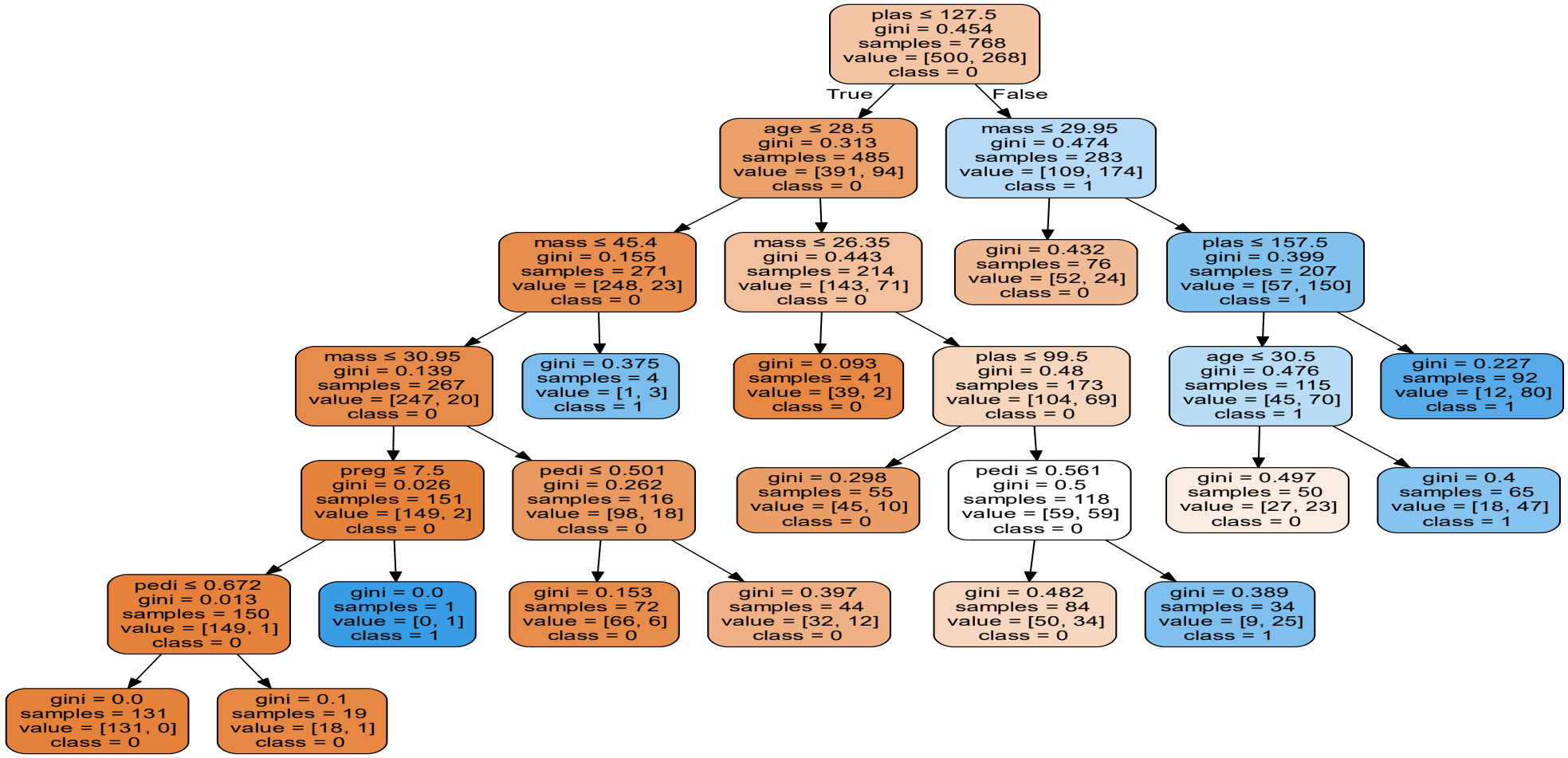
Tree for Diabetes max_depth=3



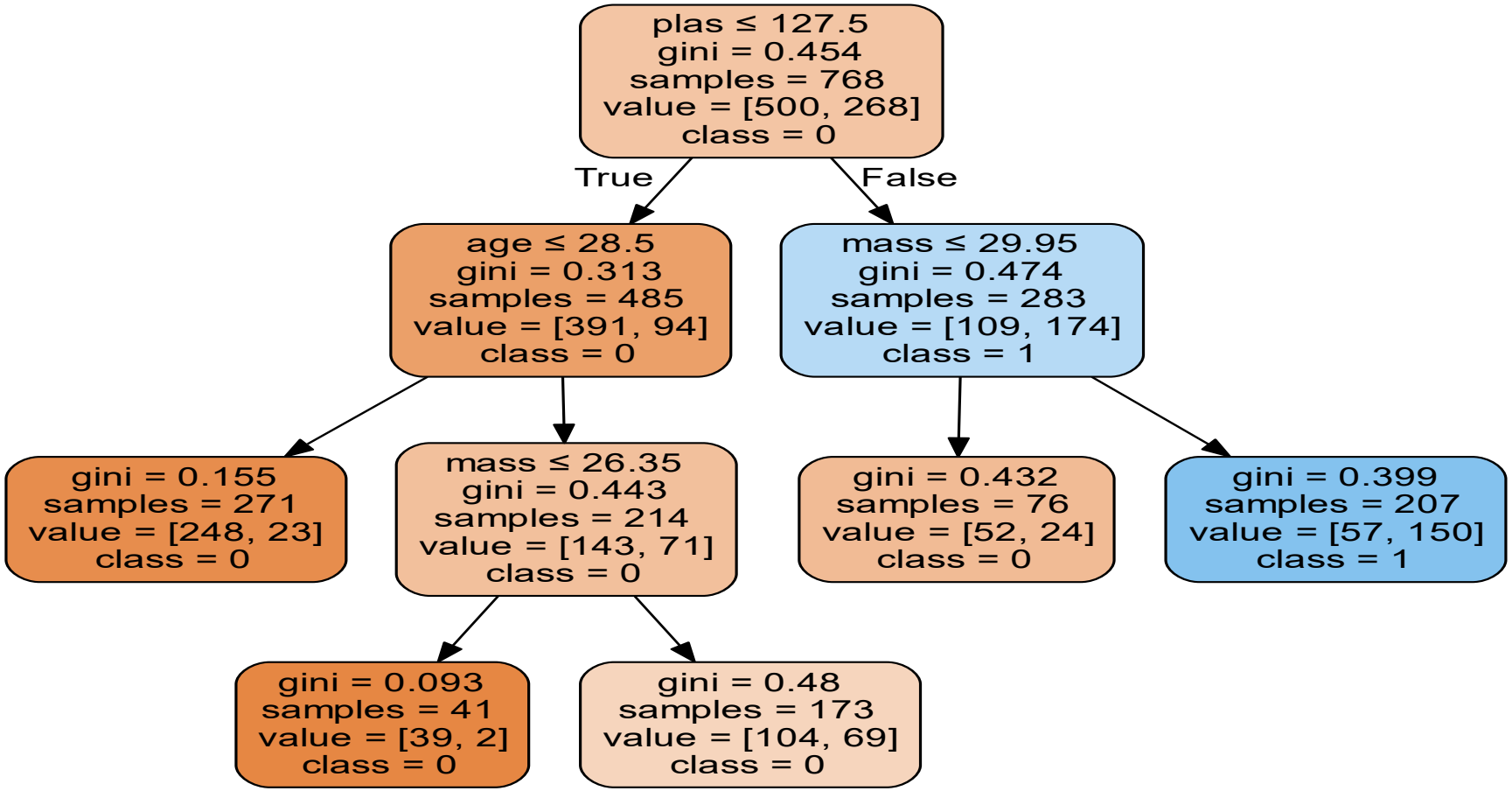
Tree for Diabetes min_samples_leaf=40



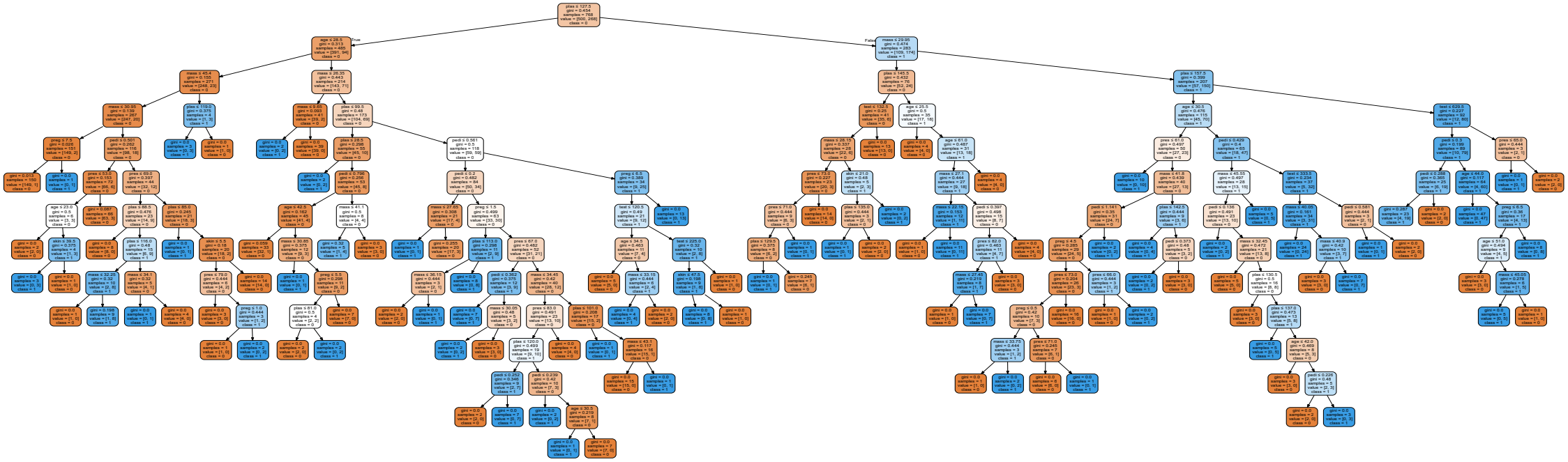
Tree for Diabetes min_samples_split=100



Tree for Diabetes max_leaf_nodes=5



Tree for Diabetes with min_impurity_decrease=.001



Pruning a tree

- To grow a tree too large can cause overfitting. This means that the model follows more the noise than the signal.

For any tree T and any $\alpha \geq 0$ (α is called the complexity parameter), a measure of merit of the tree T (or cost-complexity measure) is given by:

$R_\alpha(T) = Resub(T) + \alpha|T|$

where Resub(T) is the estimated by resubstitution of the misclassification error rate using T, and |T| is the number of terminal nodes in T. When $\alpha=0$, the largest possible tree is obtained and when $\alpha=\propto$ a tree with a single node is produced. Thus, as α increases the tree is pruned more and more.

The optimal tree T_α is the smallest tree that minimizes $R_\alpha(T)$.

DecisionTreesClassifier from scikit learn does not have a pruning option

Error estimation

Breiman, et al (1984) recommended the use of 10-fold cross validation to estimate the miss classification error. Other methods are recommended only for small samples.

```
modeltree = tree.DecisionTreeClassifier(max_depth=3)
modeltree = modeltree.fit(X,y)
from sklearn.model_selection import cross_val_score
scores = cross_val_score(modeltree, X, y, cv=10)
#Hallando la precision media y un intervalo de confianza
print("CV Accuracy: %0.3f (+/- %0.3f)" % (scores.mean(), scores.std() * 2))
CV Accuracy: 0.732 (+/- 0.078)
modeltree4 = tree.DecisionTreeClassifier(min_impurity_decrease=.001)
modeltree4 = modeltree4.fit(X,y)
scores = cross_val_score(modeltree4, X, y, cv=10)
#Hallando la precision media y un intervalo de confianza
print("CV Accuracy: %0.3f (+/- %0.3f)" % (scores.mean(), scores.std() * 2))
CV Accuracy: 0.719 (+/- 0.128)
```

Handling of missing values in decision trees

In the CART algorithm the procedure to handle the missing values is as follows:

The best partition of a node based on a predictor, say x_k is found only with the available data. If when a observation either from the training or the test sample need to be classified and there is not a corresponding value of the variable x_k then a surrogate partition is used. If the observed value of the variable in this partition is also missing then a second surrogate partition is used and so on.

This is somehow similar when in a linear model a missing value of a predictor variable is replaced by the predicted value with the regression with other predictor that is most correlated wit it.

Other decision trees algorithm have a more complicated treatment of missing values.

Advantages and disadvantages of tree classification

Advantages:

- It can be applied to any type of predictors: continuous and categorical.
- The results are very easy to understand and to be interpreted.
- There is not problem to work with missing values
- It performs automatically feature selection
- It is invariant to transformations of the predictor variables.
- It is robust to the presence of outliers.
- Is a non-prametric classifier, this means that it does not require assumptions.
- Fast computation.

Disadvantages:

- The feature selection process is biased towards the feature with more different values.
- It is not easy to establish the optimal tree.
- The prediction surface is not smooth, since it is a set of hyperplanes.
- It requires a large amount of data to assure that there is a significative number of instances in each terminal node.
- There is not a mathematical model.
- It does no take into account the interactions that can exists between the predictor variables.