Generate the Decision Tree

August 10, 2020

Based on the outputs of the decision tree, decision trees can be categorized into two types: classification tree and regression tree. The output variable of a classification tree is categorical (or discrete). On the other hand, the output variable of a regression tree is numerical (or continuous).

In this document, we start with an overview of the decision tree generation algorithm, which uses the same recursive approach as the other tree creation in Data structure. We then discuss how to select the best attribute to be a node to split the tree, which is one of the key issues. Since we have two types of decision tree, we will also discuss the attribute selection. As for the classification tree, we introduce the information entropy approach with an example. As for the regression tree, we select the attribute to split the space into two parts with the objective of minimizing the *residual sum of squares* (RSS). Since the optimal partition problem is NP-complete, we focus on the recursive binary splitting. After splitting, each area looks like a high-dimensional rectangle and thus, we call it *box*. After introducing the attribute selection, we will discuss an essential extra step for the decision tree, i.e. pruning. Finally, we end with this document by showing the advantage and disadvantage of the decision tree model as well as comparing with linear regression model (another traditional machine learning model) so that the reader may choose the proper machine learning model for their own dataset.

1 Algorithm Overview

The basic idea behind the algorithm is to partition the current dataset into two or more smaller datasets based on values of a selected attribute. The algorithm will partition the dataset recursively until all labels in the current dataset are the same or no attribute is available for further partition. The overall algorithm is shown in Algorithm 1.

Line 1-3 is the base case of the algorithm where we create the leaf node and assign the proper label based on the majority of label value. Line 4-8 is the recursive case where we will select one best attribute 1) to be a root of the new tree and 2) to further partition the dataset to multiple smaller datasets. The overall algorithm is similar to build a tree or visit all nodes in the tree through

Algorithm 1 TreeGeneration(D, A)
/* Input:
* D: available data point set; Initially, D refers to the whole dataset
* A: available attribute set; Initially, A contains all attributes.
* Output:
* A decision tree
*/
1: if set A is empty or all data points in D have the same label or D has less than 5 data points left for regression tree
2: Leaf = Yes
3: Label = most common label for all data points in D
4: else
5: Leaf = No
6: a = bestAttribute(D, A) // Find the best attribute within A for dataset D
7: for all possible value v_i for attribute a
8: $Subtree_i$ = TreeGeneration($\{d|d \in D$ with the attribute value $v_i\}$, A\{a})
9: endif

computer programs. The major task for decision tree generation is how to select the best attribute in Line 6.

2 Best Attribute Selection

What is the best attribute to separate the dataset? Our common sense tells us that the best attribute should 1) make every derived subset more "purity" than the current dataset; or 2) be "closest" to the prediction results (leaves) compared to other attributes.

In this section, we will describe how to measure "purity" or "close to leaves" in a mathematical way and can be computed through algorithms. Since the decision tree can be either classification or regression and the attribute can also have either discrete or continuous values, we will discuss classification tree with discrete attribute value and regression tree with continuous attribute value in detail. We will also provide the hint on how to select the attributes for the other two cases.

2.1 Attribute Selection for Classification Tree

As for the classification tree with discrete attribute values, the attribute selection has many different ways. Here, we introduce a possible tool based on the information theory to select the best attribute.

2.1.1 Entropy

In this subsection, we will provide the mathematical formula of entropy. The entropy of a random variable is the average level of "information", "uncertainty"

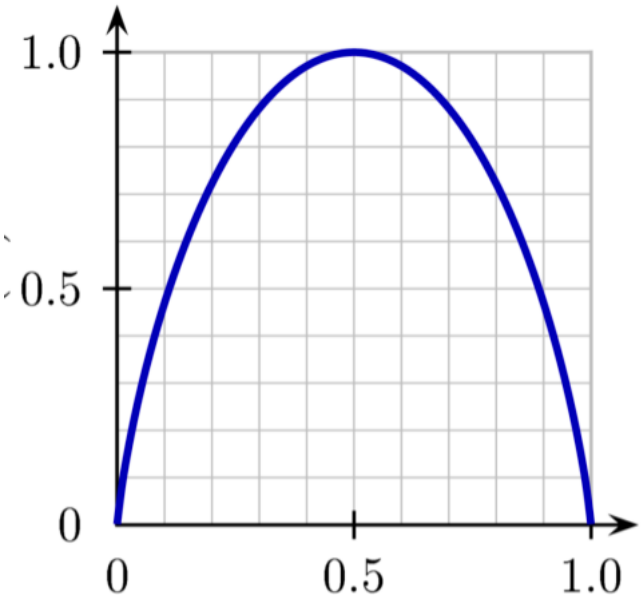

Figure 1: Entropy Function with probability p as x axis and $H(X)$ as y axis

or "surprise" inherent in the variable's possible outcomes. The higher the entropy, the less confident we are in the outcome. Consider a biased coin as an example with probability p of landing on heads and probability $1-p$ of landing on tails. The maximum surprise is for $p = 0.5$, when there is no reason to expect one outcome over another. On the other hand, the minimum surprise is if $p = 0$ or $p = 1$, when the event is known.

More formally, let X denote the event. In the above example, event X refers to the landing outcome of a biased coin. The entropy for event X in the above example is

$$H(X) = -p \cdot \log_2 p - (1-p) \cdot \log_2(1-p). \tag{1}$$

Based on the definition, the maximum entropy for the biased coin is 1 when $p = 0.5$ and the minimum entropy is 0 when $p = 0$ or $p = 1$. The entropy function graph for two outcomes is shown in Fig.1.

In general, if the event has n outcomes ($X = \{x_1, x_2, ..., x_i, ..., x_n\}$) with the corresponding probability $(p_1, p_2, ..., p_i, ..., p_n)$, the total entropy for event X is

$$H(X) = -\sum_{i=1}^{n} p_i \cdot \log_2 p_i. \tag{2}$$

The conditional entropy (or equivocation) quantifiers the amount of information needed to describe the outcome of a random variable Y given that the value of another random variable X is known. The entropy of Y conditioned on

X is written as $H(Y|X)$. Let $X = \{x_1, x_2, ..., x_i, ..., x_n\}$ with the corresponding probability $(p_1, p_2, ..., p_i, ..., p_n)$, $Y = \{y_1, y_2, ..., y_j, ..., y_m\}$ and the joint probability $P(X = x_i, Y = y_j) = p_{ij}$ for any $1 \leq i \leq n$ and $1 \leq j \leq m$. The conditional entropy can be formally defined as

$$H(Y|X) = \sum_{i=1}^{n} P(X = x_i) H(Y|X = x_i) = -\sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij} \log_2 \frac{p_{ij}}{p_i}. \quad (3)$$

where $H(Y|X = x_i)$ can be calculated through Eq. (2). In detail,

$$H(Y|X = x_i) = -\sum_{j=1}^{m} \frac{p_{ij}}{p_i} \log_2 \frac{p_{ij}}{p_i} \quad (4)$$

2.1.2 Entropy based Attribute Selection

When we look at the base case of Algorithm 1, the entropy will be 0 when all labels are the same. In order to fast move to the leaf node, we should choose the attribute which tremendously reduce the entropy of the label at the sub-dataset. Based on the this greedy consideration, our entropy based attribute selection algorithm can be described in Algorithm 2.

Algorithm 2 bestAttribute(D, A)
/* Input:
* D: available data point set passed from Line 6 in Algorithm 1.
* A: available attribute set passed from Line 6 in Algorithm 1.
* Output:
* Attribute a
*/
1: Calculate the entropy $H(Y)$ for the label column in D according to Eq. (2)
2: for each available attribute $A^{(j)}$ in set A
3: Calculate the conditional entropy $H(Y|A^{(j)} = x_i)$ according to Eq. (4)
4: Calculate the conditional entropy $H(Y|A^{(j)})$ according to Eq. (3)
5: Calculate the information gain $I(Y|A^{(j)}) = H(Y) - H(Y|A^{(j)})$
6: endfor
7: return attribute $a = \max_{A^{(j)} \in A}\{I(Y|A^{(j)})\}$

2.1.3 Classification Tree Generation Example

Table 1 provides a movie dataset example for us to build a decision tree. In this table, we have $|D| = 9$ data points (Movies 1 to 9). Each data point has $|A| = 4$ attributes (i.e. Type, Length, Director, and Famous Actors) and one label as outcome (Liked?). We treat each attribute and label outcome as one event, respectively. Therefore, we totally have five events (event T, L, I, F, and O) in this example.

Table 1: Movie Dataset

Movie	Type (T)	Length (L)	Director (I)	Famous Actors (F)	Liked? (O)
1	Comedy	Short	Adamson	No	Yes
2	Animated	Short	Lassester	No	No
3	Drama	Medium	Adamson	No	Yes
4	Animated	Long	Lassester	Yes	No
5	Comedy	Long	Lassester	Yes	No
6	Drama	Medium	Singer	Yes	Yes
7	Animated	Short	Singer	No	Yes
8	Comedy	Long	Adamson	Yes	Yes
9	Drama	Medium	Lassester	No	Yes

We will process Algorithm 1 for the movie dataset and use Algorithm 2 to select the best attribute.

<Round 1>

Since the labels in the whole are not the same (6 Yes and 3 No at the last column) and we have 4 available attributes, we will enter the "else" part in Algorithm 1 to call Algorithm 2 to select an attribute act as a root of the tree.

1) Calculate the entropy for the last column as follows.

When we look at the label (i.e. last column), we have 6 out of 9 with "yes" and 3 out of 9 with "no". Thus, we have $P(O = "yes") = \frac{6}{9} = \frac{2}{3}$ and $P(O = "no") = \frac{3}{9} = \frac{1}{3}$. The entropy for the last column can be calculated as $H(O) = -\frac{2}{3}\log_2\frac{2}{3} - \frac{1}{3}\log_2\frac{1}{3} = 0.918$.

2) Calculate the information gain for each attribute.

Since we have four available attributes in this round, we have to calculate information gain for each attribute. Here, we go through the calculation for the first attribute (i.e. Type) and the reader can work with the other three attributes.

As for the type attribute, we have three values associate with type (i.e. Comedy, Animated, and Drama).

We can calculate $P(T = "Comedy") = \frac{3}{9} = \frac{1}{3}$, $P(T = "Animated") = \frac{3}{9} = \frac{1}{3}$, and $P(T = "Drama") = \frac{3}{9} = \frac{1}{3}$.

We then calculate $P(O = "yes", T = "Comedy") = \frac{2}{9}$, which means we have 2 out of 9 rows n the table with both "Comedy" for "Type" and "Yes" for "Liked?". Similarly, We obtain $P(O = "no", T = "Comedy") = \frac{1}{9}$.

Now we can obtain based on Eq. (4) shown as follows.

$H(O|T = "Comedy") = -\frac{P(O="yes",T="Comedy")}{P(T="Comedy")}\log_2\frac{P(O="yes",T="Comedy")}{P(T="Comedy")} - \frac{P(O="no",T="Comedy")}{P(T="Comedy")}\log_2\frac{P(O="no",T="Comedy")}{P(T="Comedy")} = -\frac{2/9}{1/3}\log_2\frac{2/9}{1/3} - \frac{1/9}{1/3}\log_2\frac{1/9}{1/3} = 0.918$.

Similarly, we can calculate the entropy $H(O|T = "Animated")$ as

$H(O|T = "Animated") = -\frac{P(O="yes",T="Animated)}{P(T="Animated")}\log_2\frac{P(O="yes",T="Animated")}{P(T="Animated")} -$

$\frac{P(O=\text{"}no\text{"},T=\text{"}Animated\text{"})}{P(T=\text{"}Animated\text{"})}\log_2\frac{P(O=\text{"}no\text{"},T=\text{"}Animated\text{"})}{P(T=\text{"}Animated\text{"})} = -\frac{1/9}{1/3}\log_2\frac{1/9}{1/3} - \frac{2/9}{1/3}\log_2\frac{2/9}{1/3} = 0.918$

and the entropy $H(O|T=\text{"}Drama\text{"})$ as

$H(O|T=\text{"}Drama\text{"}) = -\frac{P(O=\text{"}yes\text{"},T=\text{"}Drama\text{"})}{P(T=\text{"}Drama\text{"})}\log_2\frac{P(O=\text{"}yes\text{"},T=\text{"}Drama\text{"})}{P(T=\text{"}Drama\text{"})} - \frac{P(O=\text{"}no\text{"},T=\text{"}Drama\text{"})}{P(T=\text{"}Drama\text{"})}\log_2\frac{P(O=\text{"}no\text{"},T=\text{"}Drama\text{"})}{P(T=\text{"}Drama\text{"})} = -\frac{1/3}{1/3}\log_2\frac{1/3}{1/3} - \frac{0}{1/3}\log_2\frac{0}{1/3} = 0$. Note that $\lim_{z\to 0} z\log_2 z = 0$.

So far, we finish Line 3 in Algorithm 2 for the Type attribute. We continue to calculate the conditional entropy $H(O|T)$ at Line 4 in Algorithm 2 based on Eq. (3) .

$H(O|T) = P(T=\text{"}Comedy\text{"})H(O|T=\text{"}Comedy\text{"}) + P(T=\text{"}Animated\text{"})H(O|T=\text{"}Animated\text{"}) + P(T=\text{"}Drama\text{"})H(O|T=\text{"}Drama\text{"}) = \frac{1}{3}\times 0.918 + \frac{1}{3}\times 0.918 + \frac{1}{3}\times 0 = 0.612$.

Finally, we obtain the information gain at Line 5 in Algorithm 2 for the first attribute (i.e. Type) as $I(O|T) = H(O) - H(O|T) = 0.918 - 0.612 = 0.306$.

The reader can go through the other three attributes and you can check your answer here: $I(O|L) = 0.306$, $I(O|I) = 0.557$, and $I(O|F) = 0.073$.

From the information gain, we will select the third attribute (i.e. Director) as the best attribute and return back to Algorithm 1.

3) Build the tree and separate the dataset.

The tree is shown in Fig. 2 and the dataset is separated into three small datasets shown in Tables 2, 3, and 4, corresponding to three directors, respectively.

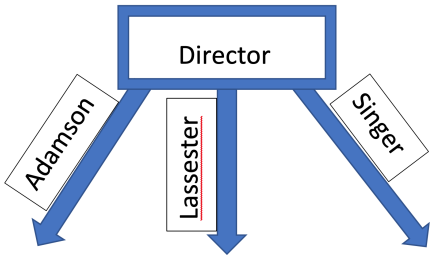

Figure 2: Tree Generated after Round 1

Table 2: Smaller Datasets for Director = "Adamson"

Movie	Type (T)	Length (L)	Famous Actors (F)	Liked? (O)
1	Comedy	Short	No	Yes
3	Drama	Medium	No	Yes
8	Comedy	Long	Yes	Yes

Table 3: Smaller Datasets for Director = "Lassester"

Movie	Type (T)	Length (L)	Famous Actors (F)	Liked? (O)
2	Animated	Short	No	No
4	Animated	Long	Yes	No
5	Comedy	Long	Yes	No
9	Drama	Medium	No	Yes

Table 4: Smaller Datasets for Director = "Singer"

Movie	Type (T)	Length (L)	Famous Actors (F)	Liked? (O)
6	Drama	Medium	Yes	Yes
7	Animated	Short	No	Yes

<Round 2>

Since the Algorithm 1 does not satisfy the base case condition at the first round, it will recursively call the second round to build three subtrees using the remaining attributes (i.e. Type, Length, and Famous Actors). Each subtree picks up one derived dataset shown in Tables on one dataset shown in Tables 2, 3, and 4.

As for Table 2, we can put the label "yes" since all labels in this table are the same. Similarly, we put label "yes" for Table 4. Those two tables reach the base case in Algorithm 1.

Now we process Table 3. Since the labels are not the same and three attributes are available, Algorithm 1 will go to the "else" part to further partition this dataset. Make sure the calculation focuses on this small dataset instead of the original dataset due to partition.

1) Calculate the entropy for the last column as follows.

When we look at the label (i.e. last column), we have 1 out of 4 with "yes" and 3 out of 4 with "no". Thus, we have $P(O = "yes") = \frac{1}{4}$ and $P(O = "no") = \frac{3}{4}$. The entropy for the last column can be calculated as $H(O) = -\frac{1}{4}\log_2\frac{1}{4} - \frac{3}{4}\log_2\frac{3}{4} = 0.811$.

2) Calculate the information gain for each attribute.

Since we have three available attributes in this round, we have to calculate information gain for each attribute. Here, we go through the calculation for the second attribute (i.e. Length) as an example and the reader can work with the other two attributes.

We can calculate $P(L = "Short") = \frac{1}{4}$, $P(L = "Long") = \frac{2}{4} = \frac{1}{2}$, and $P(L = "Medium") = \frac{1}{4}$.

We then calculate $P(O = "yes", L = "Short") = 0$ and $P(O = "no", L = "Short") = \frac{1}{4}$.

Now we can obtain based on Eq. (4) shown as follows.

$H(O|L = "Short") = -\frac{P(O="yes",L="Short")}{P(L="Short")}\log_2\frac{P(O="yes",L="Short")}{P(L="Short")} - \frac{P(O="no",T="Short")}{P(T="Short")}\log_2\frac{P(O="no",T="Short")}{P(T="Short")} = -\frac{0}{1/4}\log_2\frac{0}{1/4} - \frac{1/4}{1/4}\log_2\frac{1/4}{1/4} = 0.$

Similarly, we can calculate the entropy $H(O|L = \text{"}Long\text{"}) = 0$ and the entropy $H(O|L = \text{"}Medium\text{"}) = 0$.

Since all entropies are 0, we obtain the conditional entropy $H(O|L) = 0$ at Line 4 in Algorithm 2 based on Eq. (3) .

Finally, the information gain at Line 5 in Algorithm 2 for the second attribute (i.e. Length) is $I(O|L) = H(O) - H(O|L) = 0.811 - 0 = 0.811$.

The reader can go through the other two attributes and you can check your answer here: $I(O|T) = 0.811$ and $I(O|F) = 0.311$.

From the information gain, we can select either the first attribute (i.e. Type) or the second attribute (i.e. Length) as the best attribute and return back to Algorithm 1. Here, we suppose Algorithm 2 picks "Length".

3) Build the tree and separate the dataset.

The tree is shown in Fig. 3 and the dataset is separated into three small datasets shown in Tables 5, 6, and 7, corresponding to three lengths, respectively.

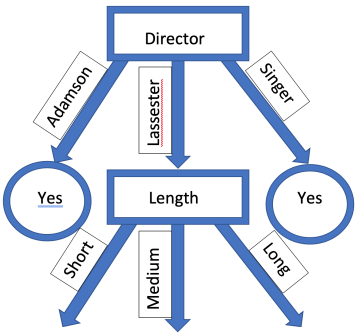

Figure 3: Tree Generated after Round 2

Table 5: Smaller Datasets for Length = "Short"

Movie	Type (T)	Famous Actors (F)	Liked? (O)
2	Animated	No	No

Table 6: Smaller Datasets for Length = "Medium"

Movie	Type (T)	Famous Actors (F)	Liked? (O)
9	Drama	No	Yes

<Round 3>

Since one branch does not satisfy the base condition at Round 2, we need one more round. All three tables (Table 5, 6, and 7) only contain one or two data

Table 7: Smaller Datasets for Length = "Long"

Movie	Type (T)	Famous Actors (F)	Liked? (O)
4	Animated	Yes	No
5	Comedy	Yes	No

rows with the same label outcomes. We can finish our example without calculating more entropy. The final tree for the example is shown in Fig. .4

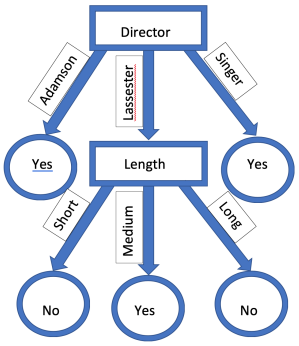

Figure 4: Final Decision Tree

2.1.4 Continuous Attribute Value

From the above discussion, we have found that entropy works fine for classification tree when the attribute values are discrete. However, many attributes such as salary and age may have continuous value. Here we provide the basic idea to deal with the continuous attribute value. We may apply the threshold to partition the continuous value into multiple groups. For example, we know that the age of persons is continuous value. However, we can partition the ages into multiple age groups: i.e. age group 1: [0, 10), age group 2: [10, 20), age group 3: [20, 30), and so on. By doing so, we get discrete age groups instead of continuous age value.

2.2 Attribute Selection for Regression Tree

When the label (or the outcome) has the continuous value, the decision tree becomes a regression tree. If the label value can be partitioned into multiple ranges and each range can be represented by assigning a number, then we can convert the regression tree to classification tree. Considering the salary dataset in which the amount of salary is the label. We may set \$50K as a threshold and convert the salary, which has continuous value, to the discrete label: label "no" means salary is less than \$50K and label "yes" means salary is no less than \$50K. The meaning associated with this conversion may be predict whether or not the

personal income can maintain a middle class live in certain area. However, if we want to learn more from the data, we may need to create the regression tree directly for the continuous label value. Without specific explanation, we deal with the continuous attribute value and the continuous label value in this subsection.

2.2.1 Residual Sum of Squares (RSS)

There are two major difference between Regression tree and classification tree. The first one is that we deal with the continuous label value and thus, we may not obtain the entropy to select the best attribute. The second one is that the attribute value is continuous and thus, we may not obtain multiple branches for one node. To solve those two difference, we introduce the recursive binary splitting.

Although we cannot use entropy to select the best attribute, we still want to derived the sub-dataset with labels more and more close to each other. Suppose we have n data points $(d_1, d_2, ..., d_i, ...d_n)$ and m attributes $(A^{(1)}, A^{(2)}, ..., A^{(j)}, ..., A^{(m)})$. Therefore, each data point d_i can be represented through a vector $d_i =< a_i^{(1)}, a_i^{(2)}, ..., a_i^{(j)}, ..., a_i^{(m)}, y_i >$ where $a_i^{(j)}$ represents the jth attribute value at the ith data point and y_i represents the label value at the ith data point. The original dataset D can be treated as a matrix with $n \times (m+1)$ entities. The tree generation equals to the matrix separation. If we select attribute $A^{(j)}$ to partition the matrix into two smaller matrices, we want find the value x_j such that

$$R_j = \min_{x_j} \sum_{a_i^{(j)} \le x_j} (y_i - \hat{y_1})^2 + \sum_{a_i^{(j)} > x_j} (y_i - \hat{y_2})^2, \tag{5}$$

where $\hat{y_1} = \frac{\sum_{a_i^{(j)} \le x_j} y_i}{\sum_{a_i^{(j)} \le x_j} 1}$ and $\hat{y_2} = \frac{\sum_{a_i^{(j)} > x_j} y_i}{\sum_{a_i^{(j)} > x_j} 1}$. Eq. (5) defines how the difference between the label value to the average value which is used as the prediction value. We can recursively partition the matrix into multiple smaller matrices to generate the tree.

Based on the linear algebra, the matrix can span a vector space. The tree generation equals to the partition of a corresponding vector space (S).

2.2.2 RSS based Best Attribute Selection

Based on the definition of RSS, we select the best attribute for regression tree through Algorithm 3.

2.2.3 Discrete Attribute Value

When we deal with the discrete attribute value, we can assign numerical value for different discrete value. For example, if the attribute has two possibilities (say "yes" and "no"), we can assign 0 to "no" and 1 to "yes".

Algorithm 3 bestAttribute(D, A)
/* Input:
* D: available data point set passed from Line 6 in Algorithm 1.
* A: available attribute set passed from Line 6 in Algorithm 1.
* Output:
* Attribute a
*/
1: for each available attribute $A^{(j)}$ in set A
2: Calculate the best RSS R_j and correpsonding x_j according to Eq. (5)
3: endfor
4: return attribute $a = \min_{A^{(j)} \in A} R_j$ and its correpsonding x_j

3 Pruning

Through Algorithm 1, we can find that we try to generate the tree with homogeneous leaves or may use all attributes. This is very dangerous since it cause the overfitting. The overfitting refers to the situation that the model works perfect for the training dataset but may cause the problem when it is applied to the test datasets or other real world datasets. We then provide the pruning approach to improve the test performance. Note that if we only have one dataset, we are required to separate the whole dataset into training sets and test sets. The pruning is shown in Algorithm 4.

Algorithm 4 Pruning(T, D', D)
/* Input:
* T: the decision tree generated in Algorithm 1.
* D': the test dataset.
* D: the training dataset.
* Output:
* new tree T
*/
1: Record the error E by testing T on the dataset D'
1: for all internal node v in T (starting at the root)
2: Remove the subtree rooted at node v so that make node v as a leaf.
3: if T is the classification tree
4: Assign the label value to v as the most common label within training dataset D used to generate subtree rooted at v.
5: else
6: Assign the label value to v as the average label value iwithin training dataset D used to generate subtree rooted at v.
7: check the error E' for the new tree T' on the dataset D'
8: if $E' < E$, T=T' and E=E'
9: endfor
10: return T

4 Discussion on Decision Tree

As the conclusion, we will briefly compare the decision tree model with the linear model and discuss the advantage and disadvantage of the decision tree.

4.1 Comparison between Decision Tree and Linear Model

In general, the linear model make assumption that the model may satisfy the following formula:

$$f(X) = \alpha_0 + \sum_{j=1}^{J} X_j \alpha_j. \tag{6}$$

The decision tree make the assumption that the model should look like

$$f(X) = \sum_{m=1}^{M} c_m \cdot 1_{X \in S_m}, \tag{7}$$

where S_m represents the vector space corresponding to the matrix.

Which model is better? It depends. If the actual function is very similar to the linear model, Eq. (6) will show this relationship quite well. On the other hand, if the dataset is more complicated and shows nonlinear relationship, the decision tree model shown in Eq. (7) is a good option.

4.2 Advantage of Decision Tree

One major advantage of the decision tree is high explainable. It is very easy to describe the classification based on the tree architecture. Since the decision tree uses the graph representation, it is easy to understand the whole process.

Some researchers believe that the decision tree is much closer to the procedure when people make decision in their daily life compared to other models.

Another advantage of the decision tree is that it does not require to introduce any more slave variables which is often required by other models.

4.3 Disadvantage of Decision Tree

One major problem with trees is their high variance. Often a small change in the data can result in a very different series of splits, making interpretation somewhat precarious. The major reason for this instability is the hierarchical nature of the process: the effect of an error in the top split is propagated down to all of the splits below it. One can alleviate this to some degree by trying to use a more stable split criterion, but the inherent instability is not removed.

Another limitation of trees is the lack of smoothness of the prediction surface. In classification with 0/1 loss, this doesn't hurt much, since bias in estimation of the class probabilities has a limited effect. However, this can degrade performance in the regression tree, where we would normally expect the underlying function to be smooth.