**Birch Clustering Implementation Logic**

The BIRCH clustering algorithm consists of two main phases as shown here:

**BIRCH CLUSTERING ALGORITHM**

• **Phase 1: Build the CF Tree.** Load the data into memory by building a *cluster-feature*

*tree* (CF tree, defined below). Optionally, condense this initial CF tree into a smaller

CF.

• **Phase 2: Global Clustering.** Apply an existing clustering algorithm on the leaves

of the CF tree. Optionally, refine these clusters.

We are focusing on Phase 1 for the project to publish a package in CRAN.

**CLUSTER FEATURE**

A CF is a set of three summary statistics that represent a set of data points in a single cluster.

These statistics are as follows:

• *Count*. How many data values in the cluster.

• *Linear Sum*. Sum the individual coordinates. This is a measure of the location of the

cluster.

• *Squared Sum*. Sum the squared coordinates. This is a measure of the spread of the

Cluster.

For example, consider Clusters 1 and 2 in Figure 21.1. Cluster 1 contains data

values (1, 1), (2, 1), and (1, 2), whereas Cluster 2 contains data values (3, 2), (4, 1),

and (4, 2). CF1, the CF for Cluster 1, consists of the following:

CF1 = {3*,* (1 + 2 + 1*,* 1 + 1 + 2)*,* (12 + 22 + 12*,* 12 + 12 + 22)}

= {3*,* (4*,* 4)*,* (6*,* 6)}

And for Cluster 2, the CF is

CF2 = {3*,* (3 + 4 + 4*,* 2 + 1 + 2)*,* (32 + 42 + 42*,* 22 + 12 + 22)}

= {3*,* (11*,* 5)*,* (41*,* 9)}

CF1 and CF2 represent the data in Clusters 1 and 2.

One mechanism of the BIRCH algorithm calls for the merging of clusters under

certain conditions. The *Additivity Theorem* states that the CFs for two clusters may

be merged simply by adding the items in their respective CF trees. Thus, if we needed

to merge Clusters 1 and 2, the resulting CF would be

CF12 = {3 + 3*,* (4 + 11*,* 4 + 5)*,* (6 + 41*,* 6 + 9)} = {6*,* (15*,* 9)*,* (47*,* 15)}

**CLUSTER FEATURE TREE PARAMETERS**

**•** Branching Factor *B*. *B* determines the maximum children allowed for a non-leaf

node.

• Threshold *T*. *T* is an upper limit to the radius of a cluster in a leaf node.

• Number of Entries in a Leaf Node *L*.

**PHASE 1: BUILDING THE CF TREE**

**1.** For each given record, BIRCH compares the location of that record with the location

of each CF in the root node, using either the linear sum or the mean of the

CF. BIRCH passes the incoming record to the root node CF closest to the incoming

record.

**2.**The record then descends down to the non-leaf child nodes of the root node CF

selected in step 1. BIRCH compares the location of the record with the location of

each non-leaf CF. BIRCH passes the incoming record to the non-leaf node CF closest

to the incoming record.

**3.** The record then descends down to the leaf child nodes of the non-leaf node CF

selected in step 2. BIRCH compares the location of the record with the location

of each leaf. BIRCH tentatively passes the incoming record to the leaf closest to the

incoming record.

**4.** Perform one of (a) or (b):

**a.** If the radius (defined below) of the chosen leaf including the new record does

not exceed the Threshold *T*, then the incoming record is assigned to that leaf.

The leaf and all of its parent CFs are updated to account for the new data

point.

**b.** If the radius of the chosen leaf including the newrecord does exceed the Threshold

*T*, then a new leaf is formed, consisting of the incoming record only. The parent

CFs are updated to account for the new data point.

**PHASE 2: CLUSTERING THE SUB-CLUSTERS**

Once the CF tree is built, any existing clustering algorithm may be applied to the

sub-clusters (the CF leaf nodes), to combine these sub-clusters into clusters.

**IMPLEMENTATION EXAMPLE OF BIRCH CLUSTERING, PHASE 1:**

**BUILDING THE CF TREE**

Let us examine in detail the workings of the BIRCH clustering algorithm as applied

to the following one-dimensional toy data set.4

*x*1 = 0*.*5 *x*2 = 0*.*25 *x*3 = 0 *x*4 = 0*.*65 *x*5 = 1 *x*6 = 1*.*4 *x*7 = 1*.*1

Let us define our CF tree parameters as follows:

• Threshold *T*=0.15; no leaf may exceed 0.15 in radius.

• Number of entries in a leaf node *L*=2.

• Branching factor *B*=2; maximum number of child nodes for each non-leaf

node.

*The first data value x*1 = 0*.*5 *is entered*. The root node is initialized with the

CF values of the first data value. A new leaf *Leaf*1 is created, and BIRCH assigns

the first record *x*1 to *Leaf*1. Because it contains only one record, the radius of

*Leaf*1 is zero, and thus less than *T*=0.15. The CF tree after one record is shown in

Figure 21.3.

*The second data value x*2 = 0*.*25 *is entered*. BIRCH tentatively passes *x*2 =

0*.*25 to *Leaf*1. The radius of *Leaf*1 is now*R* = 0*.*126 *< T* = 0*.*15, so *x*2 is assigned to

*Leaf*1. The summary statistics for CF1 are then updated as shown in Figure 21.4.

*The third data value x*3 = 0 *is entered*. BIRCH tentatively passes *x*3 = 0 to

*Leaf*1. However, the radius of *Leaf*1 now increases to *R* = 0*.*205 *> T* = 0*.*15.

Threshold value *T* = 0*.*15 is exceeded, so *x*3 is *not* assigned to *Leaf*1. Instead, a new

leaf is initialized, called *Leaf*2, containing *x*3 only.

*The fourth data value x*4 = 0*.*65 *is entered*. BIRCH compares *x*4 to the locations

of CF1 and CF2. The location is measured by *x* = LS∕*n*. We have *x*CF1 = 0*.*75∕2 =

0*.*375 and *x*CF2 = 0∕1 = 0. The data point *x*4 = 0*.*65 is thus closer to CF1 than to CF2.

BIRCH tentatively passes *x*4 to CF1. The radius of CF1 now increases to *R* = 0*.*166 *>*

*T* = 0*.*15. The Threshold value *T* = 0*.*15 is exceeded, so *x*4 is not assigned to CF1.

Instead, we would like to initialize a new leaf. However, *L*=2 means that we cannot

have three leafs in a leaf node. We must therefore split the root node into (i) *Node*1,

which has as its children *Leaf*1 and *Leaf*2, and (ii) *Node*2, whose only leaf *Leaf*3 contains only *x*4.

Note that the summary statistics for the parent CFs equal the sum of their children CFs.

*The fifth data value x*5 = 1 *is entered*. BIRCH compares *x*5 = 1 with the location

of CF12 and CF3. We have *x*CF12 = 0*.*75∕3 = 0*.*25 and *x*CF4 = 0*.*65∕1 = 0*.*65.

The data point *x*5 = 1 is thus closer to CF3 than to CF12. BIRCH passes *x*5 to CF3. The radius of CF3 now increases to *R* = 0*.*175 *> T* = 0*.*15, so *x*5 cannot be assigned to CF3. Instead, a new leaf in leaf node *Leaf*4 is initialized, with CF, CF4, containing *x*5 only.

*The sixth data value x*6 = 1*.*4 *is entered*. At the root node, BIRCH compares

*x*6 = 1*.*4 with the location of CF12 and CF34. We have *x*CF12 = 0*.*75∕3 = 0*.*25 and

*x*CF34 = 1*.*65∕2 = 0*.*825. The data point *x*6 = 1*.*4 is thus closer to CF34, and BIRCH

passes *x*6 to CF34. The record descends to *Node* 2, and BIRCH compares *x*6 = 1*.*4

with the location of CF3 and CF4. We have *x*CF3 = 0*.*65 and *x*CF4 = 1. The data point

*x*6 = 1*.*4 is thus closer to CF4 than to CF3. BIRCH tentatively passes *x*6 to CF4. The

radius of CF4 now increases to *R* = 0*.*2 *> T* = 0*.*15. The Threshold value *T* = 0*.*15

is exceeded, so *x*6 is not assigned to CF4. But the branching factor *B*=2 means that

we may have at most two leaf nodes branching off of any non-leaf node. Therefore,

we will need a new set of non-leaf nodes, *Node*2*.*1 and *Node*2*.*2, branching off from

*Node*2. *Node*2*.*1 contains CF3 and CF4, while *Node*2*.*2 contains the desired new CF5

and the new leaf node *Leaf 5* as its only child, containing only the information from

*x*6. *Finally*, *the last data value x*7 = 1*.*1 *is entered*. In the root node, BIRCH

compares *x*7 = 1*.*1 with the location of CF12 and CF345. We have *x*CF12 = 0*.*25 and

*x*CF345 = 1*.*02, so that *x*7 = 1*.*1 is closer to CF345, and BIRCH passes *x*7 to CF345.

The record then descends down to *Node* 2. The comparison at this node has *x*7 = 1*.*1

closer to CF34 than to CF5. The record then descends down to *Node* 2.1. Here,

*x*7 = 1*.*1 closer to CF4 than to CF3. BIRCH tentatively passes *x*7 to CF4, and to

*Leaf* 4. The radius of *Leaf* 4 becomes *R* = 0*.*05, which does not exceed the radius

threshold value of *T* = 0*.*15. Therefore, BIRCH assigns *x*7 to *Leaf* 4. The numerical

summaries in all of its parents are updated and we obtain the final form of the CF tree.