BirchR: An R Package for BIRCH Clustering

Ankit Anand, Akshay Kurapaty, Rohit Kata, Hemanth Devavarapu, Matthew A. Lanham

Purdue University Krannert School of Management

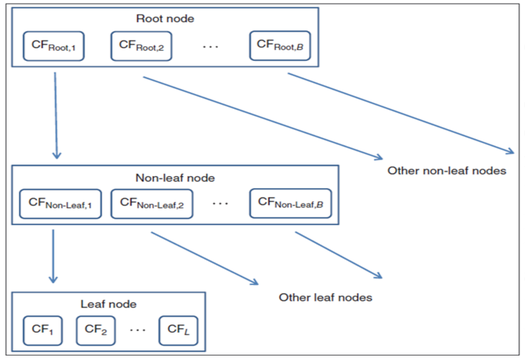
anand57@purdue.edu; akurapat@purdue.edu; kata@purdue.edu; hdevavar@purdue.edu; lanhamm@purdue.edu

Abstract

All the machine learning models are majorly classified into supervised or unsupervised learning. Clustering, which falls under the unsupervised learning umbrella is a computationally expensive operation, as most algorithms require multiple data scans. Clustering gigabytes of data will eventually become the norm. In light of these events, there is a need to develop optimal algorithms, that process the data faster and provide more accurate clusters. We developed a **BirchR** algorithm that solves this problem by creating fewer representative datapoints that are significant to making a cluster decision. Since R is open source and a core language for machine learning, we developed an R package of our algorithm that can be shared with the open source analytics community. To date there is no current active R package available on CRAN that performs Birch Clustering.

Introduction

Our **BirchR** package handles very large data sets with a time complexity and space efficiency that is superior to other algorithms. Previous clustering algorithms performed less effectively over very large databases and did not adequately consider the case wherein a data-set was too large to fit in main memory.



BIRCH is local, in that each clustering decision is made without scanning all data points and currently existing clusters. It makes full use of available memory to derive the finest possible sub-clusters while minimizing I/O costs. It is also an incremental method that does not require the whole data set in advance.

Thus, this package provides a way to perform clustering over large data sets without having to worry too much about memory and computational constraints.

Research question:

Can we develop an more efficient implementation of the Birch clustering algorithm and create an R package to be shared with the CRAN R community?

Literature Review

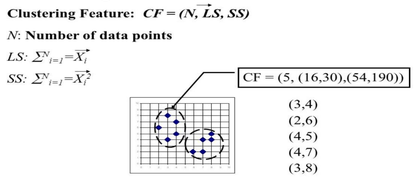
From Data Mining and Predictive Analytics Textbook by Daniel T. Larose and Chantal D. Larose, we obtained deeper understanding of the Birch Clustering, which led us to create an algorithm to tackle the problem in manageable chunks. From the R packages book by Hadley Wickham, we obtained detailed steps and pre requisites to publish an R package in CRAN. The **sci-kit learn** library in Python has an existing BIRCH clustering function that can be accessed using **sklearn.cluster.Birch**. We contrasted the results obtained with the python package to ascertain our results. To date there is no existing implementation of Birch on CRAN.

Methodology

CF-Tree: CF stands for Clustering Feature; each clustering feature helps to create representative data point based on the radius condition.

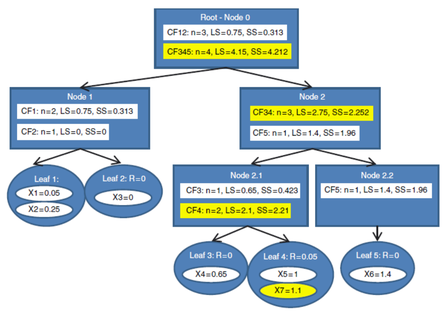
How does the CF reduces data points?

Below is a graph that illustrates the functionality of clustering features, circular dots on the graph indicate clustering features, where as other dots indicate data points.



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 new record 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.

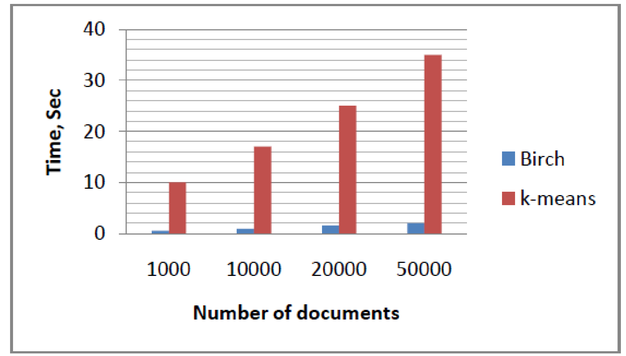


PHASE 2: CLUSTERING THE SUB-CLUSTERS

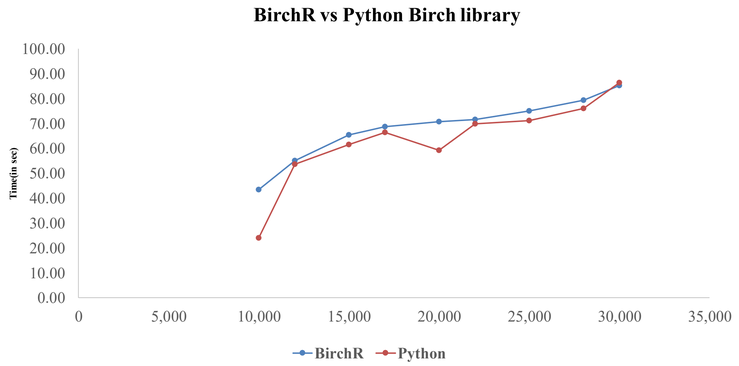
Once the CF tree is built, any existing clustering algorithm may be applied to the sub-clusters (the CF leaf nodes), to combine these sub-clusters into clusters.

Results

On contrasting our algorithm with k-means, the results are clearly visible. The processing times between the two are vastly different.



The performance, though slightly slower than the functionality in python BIRCH library is comparable to it in speed and accuracy. We aim to further improve the performance in future versions.



Conclusions

By publishing the **BirchR** package in CRAN, we are able to provide the functionality of BIRCH clustering to a wider audience. The functionalities provided by our package closely matches BIRCH clustering function in Python in speed and accuracy. Additionally, we have provided the user with a choice between obtaining cluster features or have an option to choose cluster via either **kmeans** or **hclust** (hierarchical clustering) to obtain clusters as the output after obtaining the cluster features.

Acknowledgements

We thank Professor Matthew Lanham for guidance on this project.