2013-08-09

Applying Data Mining Techniques Over Big Data

Al-Hashemi, Idrees

http://hdl.handle.net/2144/6381

Boston University
Applying Data Mining Techniques Over Big data
Idrees Al-Hashemi, Suresh Kalathur
Computer Science Department

Abstract
With rapid development of information technology, data flows in different variety of formats - sensors data, tweets, photos, raw data, and unstructured data. Statistics show that there were 800,000 Petabytes stored in the world in 2000. Today Internet is about 1.8 Zettabytes (Zettabytes is 10^21), and this number will reach 35 Zettabytes by 2020. With that, data management systems are not able to scale to this huge amount of raw, unstructured data, which is called today big data. In this present study, we show the basic concept and design of big data tools, algorithms and techniques. We compare the classical data mining algorithms with big data algorithms by using hadoop/MapReduce as the core implementation of big-data for scalable algorithms. We implemented K-means and A-priori algorithm by using Hadoop/MapReduce on 5 nodes cluster of hadoop. We also show their performance for Gigabytes of data. Finally, we explore NoSQL (Not Only SQL) databases for semi-structured, massively large-scale of data using MongoDB as an example. Then, we show the performance between HDFS (Hadoop Distributed File System) and MongoDB data stores for these two algorithms.

Introduction
Data Mining is about knowledge discovery and extracting patterns from datasets. There are many algorithms of data mining techniques, however in this study we focus on K-means algorithm and A-priori Algorithm.

K-means Algorithm: kmeans works by clustering the data into K clusters each feature, observation, are closely similar to each other and dissimilar from the features of the other clusters based on some metric distance. Example: let say we want to partition the size of the T-shirts as to {small, medium, large}; through this experiment, each person was asked to give their best fit size (numeric), height, weight, etc. Based of the above dataset design, we can partition the data by using Euclidean distance as below:

\[ D = \sqrt{(X - X_i)^2 + (Y - Y_i)^2 + ... + (D - D_i)} \]

Using Euclidean distance, assign the closest person's object feature to each other. The iteration will stop until there are no changes in cluster centeroid.

A-priori Algorithm: works by finding frequent itemset in transactional databases.
Let suppose I = {11,12,13...} is an itemset, D is a transactional database, where each transaction T is a nonempty itemset; such that T ⊆ I. A rule A =⇒ B holds with min_sup, and min_confidence that are calculate as below:

\[ \text{Min}_\text{Sup} (\{A \Rightarrow B\}) = P(A \cup B) = \text{Sup}_\text{Count}(A \cup B) \]

\[ \text{Min}_\text{Confidence} (\{A \Rightarrow B\}) = \frac{P(A \cap B)}{\text{Sup}_\text{Count}(A)} \]

Parallel K-means
Implementation and results
Parallel Apriori

A-priori Algorithm: works by finding frequent itemset in transactional databases.
Let suppose I = {11,12,13...} is an itemset, D is a transactional database, where each transaction T is a nonempty itemset; such that T ⊆ I. A rule A =⇒ B holds with min_sup, and min_confidence that are calculate as below:

\[ \text{Min}_\text{Sup} (\{A \Rightarrow B\}) = P(A \cup B) = \text{Sup}_\text{Count}(A \cup B) \]

\[ \text{Min}_\text{Confidence} (\{A \Rightarrow B\}) = \frac{P(A \cap B)}{\text{Sup}_\text{Count}(A)} \]

Hadoop/MapReduce

Hadoop: is an open source apache project that is used for distributed computing that helps hiding the complexity of the parallel computing.

MapReduce: is a programming paradigm that is used to hide the complexity of parallel programming, where developer only needs to write Map and Reduce functions.

Future works
1. Explore data visualization with big data
2. Design software package for data pre-processing with hadoop/MapReduce

References
[1]. www.confusedcoders.com