International Journal of Applied Engineering Research
ISSN 0973-4562 Volume 10, Number 11 (2015) pp. 28749-28758

http://www.ripublication.com

An Overview of Apache Mahout

Niveditha M
+91 8754853976
niveditha093@gmail.com
Deeksha Vimmadisetti
+91 9742999509
vdeeksha23@gmail.com
Aarthi R
Assistant Professor, +91 9442330313
aarthi.r4@gmail.com
Dept. of Computer Science and Engineering,
Amrita Vishwa Vidyapeetham, Amritanagar Post
Ettimadai, Coimbatore - 641112, Tamil Nadu,India

Abstract

We live in an age where there is abundant amount of data coming from every device an application. Managing and getting useful information from this large amount of data, popularly called as Big Data, has become a very tedious task. Usually, data mining algorithms are used to process bulk amounts of data. But no data mining algorithm is efficient in analyzing and finding common patterns from very large data sets in a very short amount of time.So we adopt the technique of divide and conquer where a task is broken down into smaller segments and each segment id processed on a different machine.
One of the most efficient data mining framework is Mahout which manages big data when coupled with Hadoop architecture.This paper will give an overview of Apache Mahout. We discuss the Mahout architecture, features and the most important machine learning algorithms and techniques implemented by Mahout libraries.

Keywords: Mahout', 'Recommendation', 'clustering', 'classification', 'Similarity'

Introduction

Due to the drastic increase in various applications like the social media platforms, application servers, sensors, traditional databases , network analysis ,internet of things and web analysis, there has been a tremendous growth in the amount of data that has

to be processed, which is popularly called as the big data. There is a need to process, manage and store this data in an efficient way to obtain results in real time. Big Data has become thebuzzword in every organization[1].

Mahout can manage big data by clubbing with Hadoop platform. A framework that can be used to store big data in a distributed environment is a called as Hadoop. It has been developed by Apache. It can process data across clusters of computers using simple models of programming.

Apache Mahout is a project that has been developed by the Apache Software foundation in order to implement many machine learning algorithms and techniques. The main focus of the Mahout project is to transform big data into useful information. Data science tools have been provided by Mahout to find meaningful patterns in big data sets.

Currently Mahout focuses on three main areas of machine learning:

1. Recommendation
2. Clustering
3. Classification

Features of Mahout

- The machine learning algorithms implemented by Mahout, are written on top of Hadoop. Thus it can scaled efficiently on the cloud
- Data mining techniques can be performed on large sets of data, with ease using Mahout
- Large data sets can be analyzed effectively in a short amount of time
- Most of the similarity and correlation algorithms are in-built in Mahout
- It has functions that provide a platform for evolutionary programming

Mahout Architecture

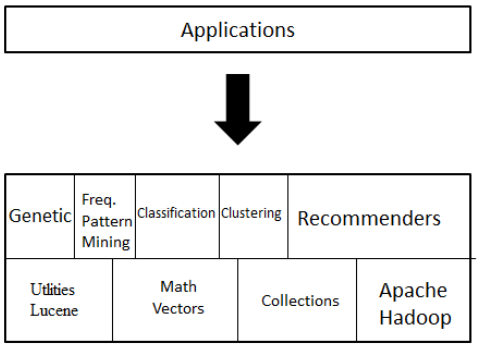

Figure 1: Mahout Architecture

Recommendations

A customized recommender system can be built using the components provided by mahout and by selecting the desired algorithm. Mahout is implemented using Collaborative filtering.

Collaborative filtering filters the data and provides recommendations to the users based on historical data. It can provide recommendations based on the following categories:

- User-Based: Items are recommended based on similar users
- Item- Based: Recommendations are calculated based on the similarity between items

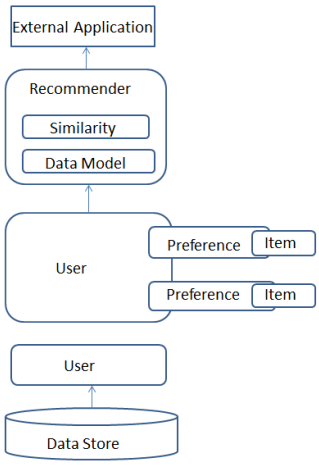

Figure 2: Recommender Architecture

Mahout recommender architecture consists of 4 major components:

1. Physical storage or Database
2. Data Model
3. Recommender
4. External Application

Step 1: The raw input or the historical data is mapped into Data Model that is compliant with Mahout.

Step 2: The recommender components like similarity measure, neighborhood etc. are tuned.

The recommender consists of many components that will be discussed in the following sections.

A. Data Model

The recommendation input is encapsulated by an abstraction called as the Data Model. It is an interface that is used to draw the user preferences and has methods which can be used to map data to mahout compliant format. It can import data from

databases, external files and can be fed in directly through the java code. Irrespective of the data source, the Data Model has a common implementation.

It consists of a basic object called as the preference that represents three values (User ID, Item ID and Preference Value). All these preferences are stored in User Preference Array or Item Preference Array. These arrays can be implemented in two ways:

- Generic User Preference Array or Generic Item Preference Array:

This stores all the user ratings or preferences in the form of numerical values that can be either negative or positive integers.

- Boolean User Preference Array or Boolean Item Preference Array:

This does not store any user preferences. It considers a default value as preference for all the items in the data set.

Data Model consists of three major components:

- Generic Data Model:

This data model can be used when the data has to be constructed in-memory rathen than importing from any external database or file. It is the simplest Data Model. Preferences can be directly accepted as inputs and it creates a preference array.

Data Model model = new Generic Data Model (preferences)

- File Data Model:

This Data Model reads the input content from an extrenal file and stores the preferences generated into the Generic Data Moodel that is in memory. Most of the file types are supported but usually, comma separated value files are preferred. Each line should contain an User ID, Item ID and preference value.

Data Model model = new File Data Model (new File("input.csv");

- JDBC Data Model:

Implementations of this model can be used to access the preference values. It also used the My SQLJDBC Data Model to get the input from the database. It is assumed by this model that all the preferences are stored in a table called taste_preferences with columns user_id, item_id and preference.

JDBC Data Model model = new My SQLJDBC Data Model (data Source, "prefs_table", "user_column", "item_column", "pref_value_column")

B. User Similarity and Item Similarity

User Similarity is used to calculate the similarity between two different users. Similarly, Item Similarity is used to calculate the similarity between two different items. These are the most important components of the user-based and item-based recommenders.

There are different similarity measures that can be used to calculate the similarity between users and items.

The following algorithms and methods that are present in the User Similarity and Item Similarity classes can be used to find the similarity between two entities:[3]

- Generic User Similarity
- Generic Item Similarity
- Euclidean Distance Similarity
- Log Likelihood Similarity

- Pearson Correlation Similarity
- Spearman Correlation Similarity
- Tanimoto Coeficient Similarity
- Cosine Similarity

Using Pearson Correlation, similarity can be calculated using:

User Similarity similarity = new Pearson Correlation Similarity(model);

C. User Neighborhood

User Neighborhood is used to define a group of users that are most similar to one particular user. Two methods are used in order to determine the neighborhood:

- Fixed – size neighborhood:

In this method, the number of neighbors for a particular user is decided in the beginning, say N. The first N users who have the highest similarity are considered to be neighbors.The algorithm used is Nearest N User Neighborhood For 10 similar users, User Neighborhood neighborhood = new Nearest N User Neighborhood (100, similarity, model)

- Threshold – based neighborhood:

A particular threshold value for similarity is decided and all the users who have a similarity higher than the threshold value. With respect to the one chosen user, are considered to be neighbors.

User Neighborhood neighborhood = new Threshold User Neighborhood (0.7, similarity, model)

D. Recommender Interface

Once we have a data model, similarity between users or items and user neighborhood, a recommender can be used to produce an estimate of relevance for new items.

There are many recommendation algorithms that can be implemented. In this paper, we discuss the algorithms relate to User-based collaborative filtering and Item-based collaborative filtering.

- User-based Recommender

Mahout supports many different similarity and neighborhood measures. These can be used to generate recommendations of items to different users based on their similarity with each other.

Recommender recommender = new Generic User Based Recommender (model, neighborhood, similarity);

- Item-based Recommender

These require only the item similarity metric. Here recommendations are provided to an user based on the ratings he has given to an item and the ratings that have been given to that item by all other users.

Recommender recommender = new Generic Item Based Recommender (model, similarity);

Clustering

Clustering or Cluster Analysis is an unsupervised technique which works on grouping data based on similarity. Any cluster must have a greater intra cluster similarity than inter clusters. As human beings we associate what we hear, smell, see and taste with the ones already in memory. Like the taste of honey reminds us more of sugar than salt. We group together the things that taste like sugar. Without even knowing what sweet tastes like, we know that all the sugary things in the world are similar and hence in the same group. We also know how different they are from the things of other group. Unconsciously, we group together tastes into such clusters [4].

Samples within a class have high similarity in comparison to one another but are very dissimilar to samples in other classes. These groups or classes are called clusters[5].The number of clusters, size, and shape are not in general known in anticipation, and each of these parameters must be determined by either the user or the clustering algorithm [6].

Clustering in Mahout helps in understanding the internal structure of data, segmenting the data so that a particular audience group is separated for better analysis, preprocessing data for other Artificial Intelligence techniques and further leads to Knowledge Discovery in data (KDD). Clustering can be done in different approaches such as Top-down approach, Bottom up approach, fixed number of centers.

Clustering is all about organizing items from a given collection into groups of similar items. Clustering a collection involves three things [7]:

- An algorithm
- A rule to detect similarity and dissimilarity.
- A stopping condition

Clustering algorithms in Mahout are K-means, Fuzzy K-means, LDA (probabilistic clustering technique), Hierarchical and Canopy. All these algorithms takes input data in the form of vectors, the process of converting input data to vectors is called Vectorization.

In Mahout, vectors are implemented using three different classes which are optmized to address the need of different scenarios. Abstract Vector[8] class has all the different subclasses that implements different scenarios. The following are the classes and when it could be used,

- DenseVector
- RandomAccessSparseVector
- SequentialAccessSparseVector [4, 8].

In reality, the names and implementations are not as important as the results the mahout algorithms produce. The steps involved in clustering data using Mahout are:

Step 1: Preprocess the data available into the format that can be used by Mahout Algorithms like text input must be converted to numeric form vectors.

Step 2: After Preprocessing the vectors are fed to the clustering algorithms in the Hadoop server.

Step 3: Evaluate the vectors for result.

Step 4: End if the desired clusters are achieved else Iterate [9].

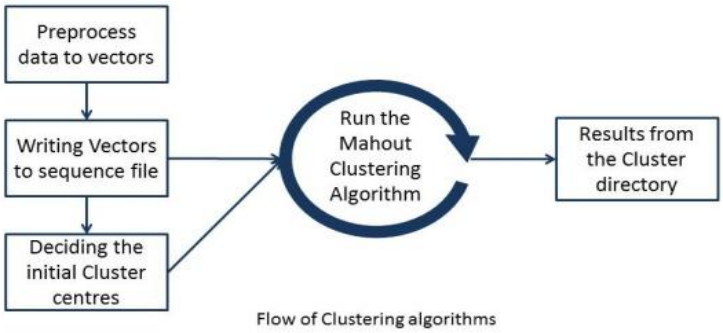

Figure 3: Clustering Algorithm Flow

Mahout Clustering algorithms help in finding nearby points in n-dimensional space, where each vector represents a point in this space, and each element of a vector corresponds to a dimension in this space. A notion of similarity and dissimilarity is based on a set of functions that measures similarity between new elements coming in and the elements already in the group. A stopping condition is mandatory so as to ensure that no more clustering can be done in the existing groups.

Finding a function that quantifies the similarity between any two data points as a number is the major issue to be handled in clustering. Similarities is measured by two ways, they are

1. Similarity by Correlation
2. Similarity by Distance

Similarity by Correlation:

Correlation can also be used as a measure to cluster data. Generally Correlation constants are used in case of Recommendations in Mahout.

Similarity by Distance:

Different measures of distance are the most convenient methods to quantize similarity as a numeric value. Distance measures are useful in cluster analysis and Mahout Classes that implement the various distance measures are [10],

- Chebyshev Distance Measure
- Cosine Distance Measure
- Euclidean Distance Measure
- Mahalanobis Distance Measure
- Manhattan Distance Measure
- Minkowski Distance Measure
- Squared Euclidean Distance Measure
- Tanimoto Distance Measure
- Weighted Euclidean Distance Measure
- Weighted Distance Measure
- Weighted Manhattan Distance Measure

Tuning a clustering problem involves choosing proper features and clustering algorithm so that the cluster quality is good. Evaluating clusters can be done on the basis ofinter and intra-cluster distances.

Classification

Classification is a form of decision making that gives discrete answers to questions. The advantage of Mahout over other approaches becomes evident as the number of training examples gets extremely large (more than 10 million training examples).In any non-scalable system, the time and memory requirements for training does not increase linearly which leads the performance of the system to slow down. Mahout being more scalable and could simultaneously process the data with optimum memory requirements makes it more advantageous for huge dataset. As the amount of electronically stored data increases, the expense of data acquisition decreases drastically. The large amount of data for training helps in increasing the accuracy for the classifier.

For building any classification system involves two phases,

1. Creating a classification model by learning algorithm.
2. Using the trained model to assign data to existing groups.

First phase of creating a classifier involves selecting a proper training set, deciding output categories, what learning algorithm to be used and what features to be selected as the input key parameters.

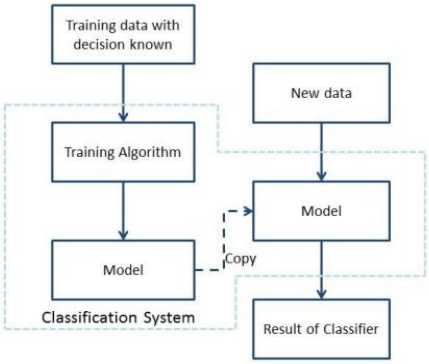

Figure 4: Classifier System

Classification algorithms are a form of supervised learning as the desired target category is already known. The general steps in building a classification project happens in three stages.

*Stage 1:*Training the model, we define target variable, collect historical data, define predictor variables, select a learning algorithm and use the learning algorithm to train the model.

Stage 2: Evaluating the model, deals with running test data, adjusting the input (use different predictor variables, different algorithms, or both).

Stage 3: Using the model in production which estimates the unknown target value of the new input data and also retrains the model if required.

In reality, raw data must be collected and preprocessed so as to convert it into a form that can be used by the classifier. This process is very complex and is time consuming. After obtaining the data in the required form we select the target variable and feature on what basis the selection needs to be done, then encode them into vectors so that can be fed into the Mahout classifiers. The feature selected can be of four types:

- Continuous
- Categorical
- Word-like
- Text-like

Mahout classifier uses several algorithms based on the size of the data in the training set.[4]

Table 1: Comparison of Classification Algorithms

Size of training data	Algorithm	Execution Type and Characteristics
< 10 million	Stochastic gradient descent (SGD)	Sequential, Incremental. Efficientover the appropriate datarange
	Support vector machine (SVM)	Sequential, Efficient over the appropriate data range
	Random forests	Parallel, High overhead fortraining; costly but offers complex classifications
>10 million	Naive Bayes	Parallel, Strongly prefers text-like data; medium to high overhead for training
	Complementary naive Bayes	Parallel, Similar limitations to Naïve Bayes

Evaluation is an iterative process that occurs when we start building the classifier followed by tuning it for better performance. As the time goes on, if there is a lack in performance we can improvise it using the current evaluation techniques to come up with new solutions. Also Evaluation is an important step after production launches the classifier.

Mahout offers variety of performance metrics to evaluate classifiers. The major approaches are percent correct, confusion matrix, AUC, and log likelihood. Naïve Bayes and Complementary Naïve Bayes are efficiently evaluated using Percent correct and Confusion matrix. Whereas any of the methods works for SGD but AUC or Log likelihood provides more meaningful insights.[4]

After successful evaluation the classifier is integrated as a part of the larger system and is deployed to production for beneficial use.

Conclusion

Apache Mahout has significant capabilities for Clustering, Classification, Recommendation and Collaborative Filtering. Clustering does not give any unique solution but it depends on the feature based on which the clustering is done. Clustering algorithms still provide clusters even if there are no clusters in reality. When we apply clustering to the data, we hypothesize that clusters exist. So clustering results cannot be generalized but can be analyzed for the requirement.

Classification helps us decide whether a new input fed to the classifier matches any previously existing pattern or not and helps in detecting suspicious activity or fraud.Finally, just as a real mahout leverages the strength and capabilities of the elephant, so too can Apache Mahout help you leverage the strength and capabilities of the yellow elephant that is Apache Hadoop [9].

References

[1] Mahout-Introduction. Available: http://www.tutorialspoint.com/mahout/mahout_introduction.htm

[2] UserSimilarity(Mahout Core 0.7-cdh4.3.2 API). Available:http://archive.cloudera.com/cdh4/cdh/4/mahout-0.7-cdh4.3.2/mahout-core/org/apache/mahout/cf/taste/similarity/UserSimilarity.html

[3] Sean Owen, R.A., Ted Dunning and Ellen Friedman: 'Mahout in Action' (Manning Publications, 2010. 2010)

[4] Esteves, R.M., Rong, C., Pais, R.,'K-means clustering in the cloud - a Mahout test', Workshops of International Conference on Advanced Information Networking and Applications,2011

[5] Osei Bryson, K.M., 'Assessing Cluster Quality Using Multiple Measures - A Decision Tree Based Approach in The Next Wave in Computing', The Next Wave in Computing, Optimization, and Decision Technologies, 2005 29, (VI), pp. 371-384

[6] Mahout-Clustering. Available: http://www.tutorialspoint.com/mahout/mahout_clustering.htm

[7] AbstractVector (Mahout Math 0.5-cdh3u6 API). Available:http://archive.cloudera.com/cdh/unstable/mahout-0.5-cdh3u6/mahout-math/org/apache/mahout/math/AbstractVector.html

[8] Introducing Apache Mahout. Available: https://www.ibm.com/developerworks/library/j-mahout

[9] Uses of Interface org. apache. mahout. common. distance. Distance Measure (Mahout Core 0.7-cdh4.7.0 API). Available: http://archive.cloudera.com/cdh4/cdh/4/mahout/mahout-core/org/apache/mahout/common/distance/class-use/ DistanceMeasure.html