Maharishi University of Management
Masters in Computer Science

CS 522 Big Data
October 2015

Non-Personalized Car Recommender Using Apriori Algorithm for Frequent Item Set

Hiep Nguyen
Toan Quach
Mark Pit

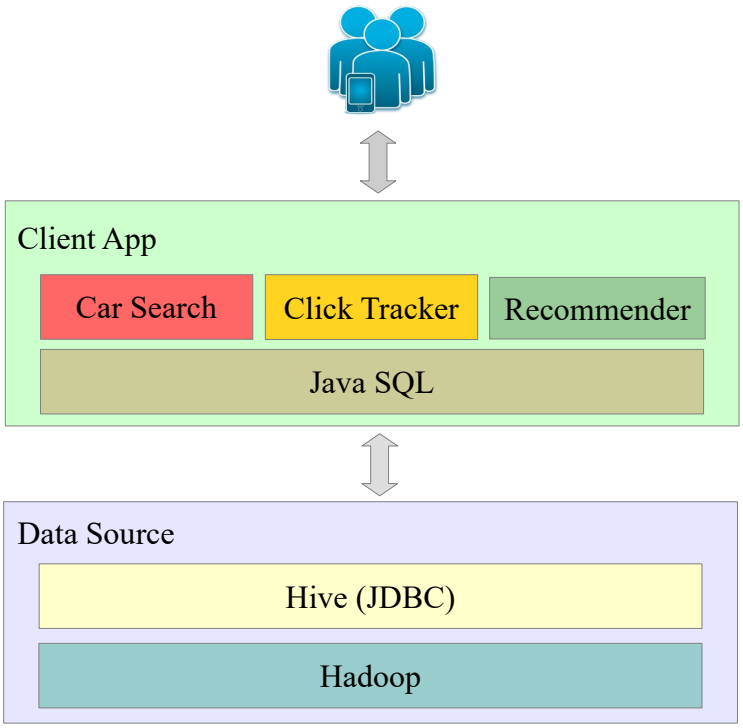
Contents:

I. **Overview**

The domain is non-personalized, which means it gets data from other users to form a recommendation. This is done through the detection and storage of user clicks and searches and transforming it into a set of frequently accessed items using a data mining algorithm called Apriori.

Datasets are scraped or crawled from data sources and pre-processed before importing to the HDFS using Hive. Data is then retrieved by the application through JDBC.

II. **Architecture**



III. **Apriori Algorithm**

Apriori is an algorithm for frequent item set mining and association rule learning over transactional databases. It proceeds by identifying the frequent individual items in the database and extending them to larger and larger item sets as long as those item sets appear sufficiently often in the database. The frequent item sets determined by Apriori can be used to determine association rules which highlight general trends in the database.

Example:

Assume a large set of car ID's taken from the database including the target user's clicks and search history:

Itemsets
{1,2,3,4}
{1,2,4}
{1,2}
{2,3,4}
{2,3}
{3,4}
{2,4}

We will use Apriori to determine the frequent item sets of this database. To do so, we will say that an item set is frequent if it appears in at least 3 transactions of the database: the value 3 is the *support threshold*.

The first step of Apriori is to count up the number of occurrences, called the support, of each member item separately, by scanning the database a first time. We obtain the following result

Item	Support
{1}	3
{2}	6
{3}	4
{4}	5

All the itemsets of size 1 have a support of at least 3, so they are all frequent.

Source: *wikipedia.org/wiki/Apriori_algorithm*

IV. **Recommendation Criteria**

1. Users need to login to view recommendations

2. New users will initially have no recommendations

3. If users have search and click history:

    3.1 If user car searches does not have similar data with other users, the recommendation is based on his own search and click history sorted according to *frequency*.

    3.2 If user car searches have *similar* data with other users:

        3.2.1 Retrieve all *associated* data and use *Apriori* to order based on the frequent *Car ID* given a *minimum support.*

V. **Data Sources and Technologies**

Data Sources:

- Edmunds Car Ratings API (edmunds.mashery.com)
- Edmunds Website (edmunds.com)
- User Search and Click Hits

Technologies:

- Spring MVC
- Hadoop Hive
- PHP WebCrawler and Data Extractor
- Facebook Login API
- jQuery, CSS