Recommender System

(5th - Semester SOE Project)

Project Supervisor

Dr. Abhishek Vaish

Project Members

Ankit Kumar (IRM 2016002)
Avinash Yadav (ITM 2016004)
Neeraj Mishra (IIM 2016003)
Chirag Meel (ISM 2016006)

Abstract

This report presents the idea of a system for movie recommendation using content-based, collaborative approaches on static datasets with extension over real time.

Recommender systems improve access to relevant products and information by making personalized suggestions based on previous examples of a users likes and dislikes. Most existing recommender systems use collaborative filtering methods that base recommendations on other users preferences. We describe a recommender system using a hybrid approach using the positive aspects of both approaches to provide a wider array of recommendation. Initial results demonstrate that this hybrid approach can bring forth more accurate recommendations than existing system.

Introduction and Motivation

Recommender System as the name suggest are used to predict some item based on certain characteristics of user as well as items. The recommendations usually carried out by predicting the ratings a user will give to particular item which is still not purchased or reviewed by the user. To predict the ratings we design different engines which is fed the Data of user, other users as well as items whose predictions are to be made. There are traditionally following types of models used for recommendations:

- Simple Recommender.
- Content based Recommender.
- Collaborative Filtering based Recommender.
- Hybrid Recommender.

There are various more variations / enhancements of these tradition models and various algorithms / techniques and Frameworks such as:

- Pre-Filtering for collaboration based.
- Item-to-Item Collaborative Filtering.
- Lambda Architecture.
- Artificial Neural Networks.
- Cosine Correlation.
- Apache Spark.

- ## Simple Recommenders

  Simple Recommenders offer generalized recommendations to every user, based on movie's popularity and/or genre and many times several other attributes like year of release, gross collection, suitable age group, current geographic location of the user. The basic idea behind this system is that movies that are more popular and have proved to be better in the past will get higher preferences.

- ## Content based Recommenders

  Content-based recommenders suggest similar items based on a particular item already consumed or reviewed by the user. This system uses item metadata, such as genre, director, description, actors, etc. for movies, to make these recommendations. The recommendation system came into existence due to the sole instinct that a person usually have a finite preferences which doesn't change very quickly. So, if a person likes one movies then he/she is most likely to like movies similar to that.

- ## Collaborative Filtering based Recommender

  Collaborative filtering engines try to predict the rating or preference that a user would give an item-based on past ratings and preferences of other users. Actually we can say it is based on popular saying which states “You should learn from others experience”. It recommends based on the ratings of other users which have same likings and preferences. First it finds similar users in terms of ratings given to movies which they both have rated. Then based on the pattern of other similar users on movies which this user has not seen, recommends to the user.

- ## Hybrid Recommender

  These Recommendation systems combine both the methods i.e, Content Based and Collaborative Filtering to eliminate their individual shortcomings.

Problem Definition

Recommendations for movies over MovieLens dataset on collaborative based filtering and content based filtering utilizing latest methods and techniques as well as implementing the recommender engines on real time data effectively and efficiently.

Literature Review

In this report the new state of the art technologies in the field of recommender system are reviewed along with the effective and efficient industry grade algorithms and softwares. The intent of this report is to get the knowledge about the current trends in recommender systems which forms the basis of almost all the ecommerce business. The main aim of this report is to get the reader a deep knowledge about the ongoing techniques in recommender systems and what lies beyond. Currently the user and item data is represented as a 2 dimensional matrix. Then various methods are used to predict the expected ratings. These algorithms are generally time consuming and are run after a fixed amount of time frequently offline on the dataset. While with certain new techniques we can do all this in near real time which is currently the topic to be explored. Not to mention the data we are talking here about is much bigger so machine learning is the basic of implementations.

Benefits and Challenges

According to some report the E-commerce business around the globe will reach around Fifty two billion dollars (37,49,46,00,00,000.00 Indian Rupee) in upcoming years and for such a huge amount at sake we need to have it running efficiently and effectively. Not only in E-commerce but in different fields recommendations systems are useful, in some it has become a necessity. Some of the major fields are : Books Management System for a Library, Movies and serials recommendation on MovieLens, Amazon and netflix. The major benefit of working in this area is that we are making the web intelligence where it can predict one what he likes or would want to explore. The one of the biggest challenges in this field is lack of required data set available for research. Lack of real time dynamic data and very very less knowledge about cross-domain recommendation which is the future.

Software Requirements

- Apache Spark

  Apache Spark is an open source distributed computing framework which can be used by researchers and students for their general purpose projects and research. Spark provides the users with many features, fault tolerance and data parallelism being some of features. Its latest release provides the data sets API for the smooth functioning. Generally, Spark requires two main components to work with, which are : a cluster manager and a distributed storage system.

- Lambda Architecture

  Lambda architecture is a big data-processing architecture designed to handle massive quantities of data by taking advantage of both batch and stream processing methods. This approach to architecture attempts to balance latency, throughput, and fault tolerance by using batch processing to provide comprehensive and accurate views of batch data, while simultaneously using real-time stream processing to provide views of online data. The two view outputs may be joined before presentation. The rise of lambda architecture is correlated with the growth of big data, real-time analytics, and the drive to mitigate the latencies of map-reduce

- Redis

  Spark-Redis provides access to all of Redis' data structures - String, Hash, List, Set and Sorted Set - from Spark as RDDs. It also supports reading/writing Dataframes and Spark SQL syntax.The library can be used both with Redis stand-alone as well as clustered databases. When used with Redis cluster, Spark-Redis is aware of its partitioning scheme and adjusts in response to resharding and node failure events.

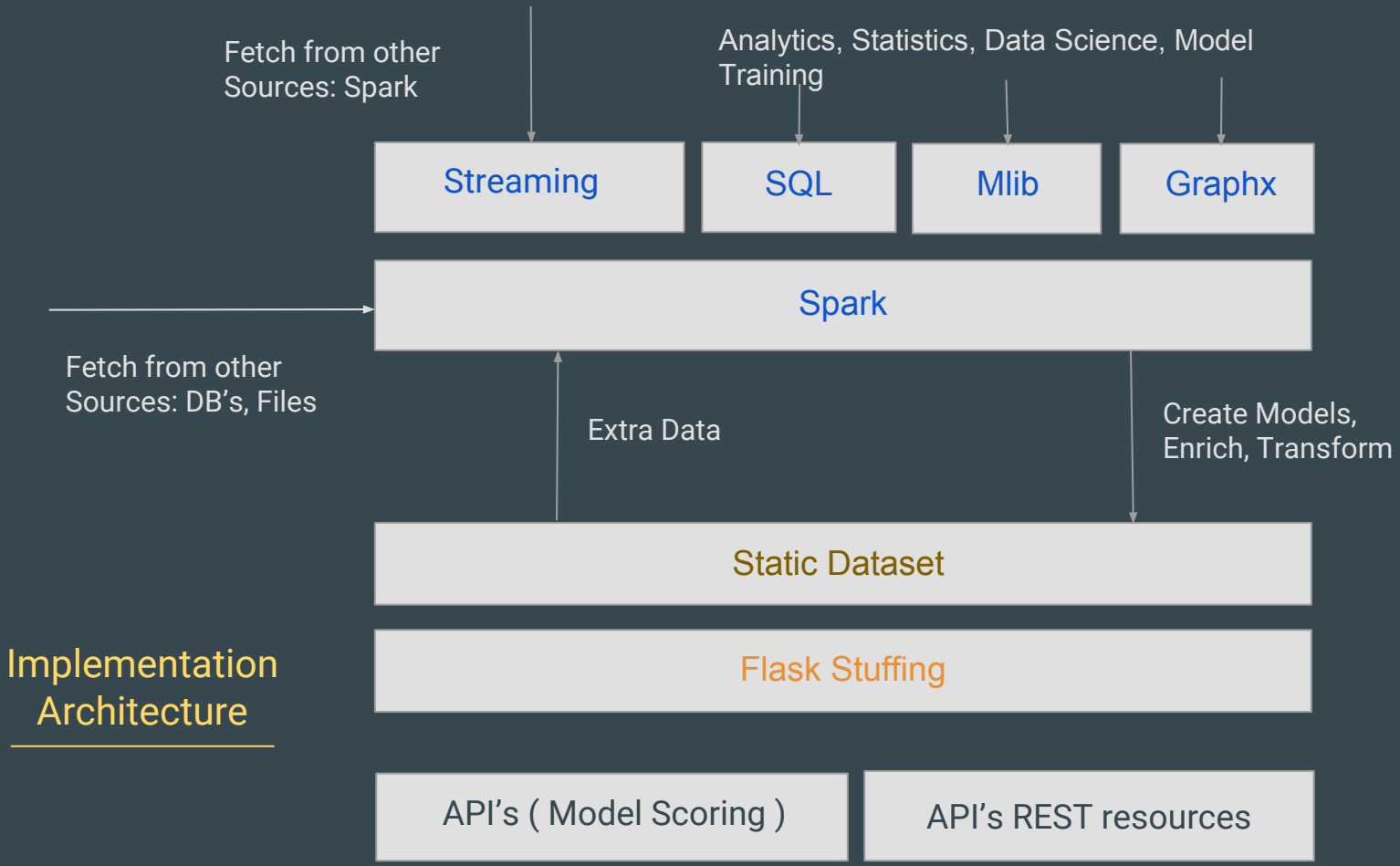
Proposed methodology

To construct the content-based movie recommender system, we consider the user profile consisting of movie information (such as story line and genre), which user rated in a specified time. The movie information is gathered from IMDb. We incorporate the profile of individuals into the interests extraction module and discover user interests. Each interest indicate a group of similar movies that are selected by the user in the past.

A. Content Based Recommender
B. Item-based Collaborative Filtering
C. Collaborative Filtering with User Pre-Filtering

Implementation Details

In the modern era, real-time data processing pipeline consists of 3 elements: event logging ,then streaming of data by a powerful engine and NoSQL storage system. The event logging helps to collect events in fast and secure way, Data streaming processes them and the results are stored into a JSON file using Apache Cassandra. We will use these three major tools such as Apache Kafka, Apache Spark and Apache Cassandra. We receive user's events (i.e. user's taste stream) in Kafka, process them in Spark and storing part will place in Cassandra.



Implementation Architecture

MLlib

MLlib is a library made by Apache which is fully compatible with every product of Apache. This library is used to implement machine learning algorithms like Linear Regression, Linear Clustering, Classification, Dimensionality Reduction, Alternating Least Square, Logistic Regression. It increases the performance over large data using parallelism and it is having utilities for linear algebra as well as random data can also be generated efficiently.

Redis Spark Connector

Spark-Redis provides access to all of Redis' data structures - String, Hash, List, Set and Sorted Set - from Spark as RDDs. It also supports reading/writing Dataframes and Spark SQL syntax.The library can be used both with Redis stand-alone as well as clustered databases. When used with Redis cluster, Spark-Redis is aware of its partitioning scheme and adjusts in response to resharding and node failure events.

TimeLine

Pre Mid-Sem

- Study of research papers, probable applications and study of the functioning of collaborative and content based filtering.
- Analysing Strategies for enhancing efficiency, Finding out viable datasets of static nature.

Post Mid-Sem

- Week 1 - 2 : Understanding the basic Framework
- Week 2 - 4: Coding the necessary segments for the system, testing, debugging, and running analysis

Expected Result

Top k lists of recommended items based on different models on static and real time dynamic data.

Conclusion

We will implement several classic algorithms to produce accurate real-time recommendations, including the well-known recommendation algorithms commonly used in industry such as the item based and user based collaborative filtering, content based. In addition, we will employ some mechanisms to make these algorithms more practical and accurate in the real-time recommendations using Apache Kakfa for data streaming and and Apache Storm for streamed data processing and Cassandra for storing result for future recommendation.

References

[1] Greg Linden, Brent Smith and Jeremy York Amazon.com Recommendations :Item-to-Item Collaborative Filtering:2003

[2] Jianming He and Wesley W. Chu A Social Network Based Recommender System (SNRS) :2010

[3] Gediminas Adomavicius and Alexander Tuzhilin Toward the Next Generation of Recommender Systems: A Survey of the state-of-the-Art and Possible Extensions:2005

[4] Thanisa Numnonda A real-time recommendation engine using lambda architecture:2017

[5] Nikos Manouselis, Hendrik Drachsler, R iina Vuorikari, Hans Hummel, Rob Koper Recommender Systems in Technology Enhanced Learning:2017

[6] Emanuel Lacic,Dominik Kowald and Elisabeth Lex Neighborhood troubles: On the Value of User Pre-Filtering To Speed Up and Enhance Recommendations:2018

[7] Yanxiang Huang, Bin Cui,Wenyu Zhang, Jie Jiang, Ying Xu tencentRec: Real-time Stream Recommendation in Practice:2015

[8] Raymond J. Mooney, Loriene Roy Content-Based Book Recommending Using Learning for Text Categorization:2015

[9] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl Analysis of Recommendation Algorithms for E-Commerce:2000

[10] Marko Balabanovic and Yoav Shoham Content-Based, Collaborative Recommendation:2015

[11] Chhavi Rana and Sanjay Kumar JainA study of the dynamic features of recommender systems:2012

[12] Ago Luberg, Tanel Tammet and Priit JrvSmart City: A Rule-based Tourist Recommendation System:2011

[13] Francesco Ricci, Lior Rokach, and Bracha Shapira Recommender Systems: Introduction and Challenges:2015

[14] Zi-Ke Zhang, Chuang Liu, Yi-Cheng Zhang and Tao Zhou Solving the cold-start problem in recommender systems with social tags:2010

Thank You