Netflix Recommendation System - Group 12

Nisarg Vinchhi - nvinchh
Rohit Mandge - rnmandge
Vaibhav Rajadhyaksha - vrajadh

Abstract

The recommendation systems of late is becoming a norm in this data driven world. Popular entertainment websites like Netflix, Hulu, Spotify etc are more drawn towards learning the user interests and recommending them with the similar titles which will helps boost their market. The application of machine learning techniques to make sense of unstructured data has transformed the paradigm of the movie recommendation and rating prediction to a much reliable, accurate model. There are certain challenges in building such systems given the individual tendency of each user in rating the movies. However, transforming data into a normalized form and observing the pattern of user ratings over similar movies will help us predict the rating of a movie. This paper presents one such prediction and recommendation system with grouping the similar movies and users over various similarity metric as a main agenda and building a prediction system to achieve lesser error than the Netflix's own prediction system. For grouping similar entities, we have used Pearson correlation coefficient as the similarity metric as well the Alternating Least Squares method which uses the latent factors to identify similarity. For rating prediction system, we learned the linear regression model over the normalized rating data for each user/movie pair available and predicted the ratings of unseen data.

Introduction

Recommendation systems try to recommend items (movies, music, webpages, products, etc) to interested potential customers, based on the information available. A successful recommendation system can significantly improve the revenue of e-commerce companies or facilitate the interaction of users in online communities. Recommendation systems are generally constructed using two different methods, content based filtering and collaborative filtering. Content based filtering uses the features of users or items. In some cases, it is difficult to extract features from the items which makes the recommendation task difficult. Collaborative Filtering on the other hand just uses the ratings of items given by users to predict ratings of new user-item pairs. The main idea behind collaborative filtering is that two users probably continue choosing similar products if they have already chosen similar ones. We will consider Collaborative Filtering for building recommender systems in this project.

Collaborative filtering tries to identify relationships and interdependencies among the users and the movies that they are rating. The only required information is the past habits of the users like the ratings they have given or the times they have watched a particular movie. Explicit feedback where users rate movies directly are most convenient as they are a direct indication of the amount of interest the user has in a particular movie. However, there are certain situations in which explicit feedback is not enough. For e.g.: initially a user watched a movie and gave it an average rating. However, he watched it again and liked it and went on to watch it many more times. Based on the rating which we have, we feel the user didn't like the movie much but that is not the case. This is

when implicit feedback comes into the picture. Implicit feedback indirectly reflects the opinions of users by observing their past behavior.
Collaborative filtering can be achieved by implementing various algorithms such as regression, clustering, matrix factorization, latent class models and Bayesian models. In this project we have used the Pearson Correlation Coefficient in the explicit feedback model and Alternating Least Squares latent class model in the implicit case for finding similarity between entities.

System Requirements

- Software Requirements:
    - The data structure that we plan to use. For e.g. Nested dictionaries in Python.
    - Frameworks used for parallel and distributed computing is Apache Spark, Hadoop or YARN.
- Hardware Requirements:
    - Memory (RAM) Requirements
    - Storage (ROM) Requirements
    - Processor Requirements. For e.g. Core i7 or i5 (64-bit Architecture)

Data

The Netflix Prize dataset consists of around 200 million records. Each movie name is mapped to a movie ID. For each movie ID a separate record is available which is of the form “userid, rating, time”. Userid is an integer and rating defined as an integral scale from 1 to 5. Date is represented as a timestamp.

Related Work

- Other approaches or techniques used in the past for recommending the movies to the user?
- What are the other frameworks that can be used to process the data in parallel and a distributed way?
- What is our approach and how does it differ from other approaches?

Analysis

- What do we want from the dataset?
- How are we going to establish the relationships between different attributes?
- The techniques used in the data analysis?
- Steps of the data analysis
    - Pre-processing
        - Data Cleaning
    - Algorithms to be used for identifying relationships between users and ratings
        - Collaborative Filtering techniques
        - Similarity matrix method
    - Predicting rating to the users
        - Predictions based on the previous ratings given by the user
        - Predictions based on the ratings given to the movie by similar users

Expected Results

- Expected answers from the algorithms applied.
- Tables showing the input data and expected recommendations

Experimental Results

- After applying algorithms, what is the actual result that we get?
- Tables showing the given input and the actual recommendations
- Comparison between expected results and experimental results
- What is the accuracy, precision and error rate of the predicted results?
- How did YARN or Spark managed the distributed computing?
- Pros and Cons of the framework that we used?
- How did it affect the parallel computing of the netflix dataset?

Conclusion

- Does the project predict the desired results?
- Are we successful in processing such a huge amount of data efficiently?
- Is the existing approach scalable?
- How has the used framework helped manage the data?
- What were the trade offs of using the framework that we used for processing the data?

Future Scope

- How can the accuracy, performance and robustness of the existing system be improved?
- How can we make the system more cost effective?

References

- Collaborative Filtering for Implicit Feedback Datasets, Yifan Hu,Yehuda Koren and Chris Volinsky
- Fast als-based matrix factorization for explicit and implicit feedback datasets, Istvn Pilszy, Dvid Zibriczky and Domonkos Tikk
- Large-Scale Parallel Collaborative Filtering for the Netflix Prize, Yunhong Zhou, Dennis Wilkinson, Robert Schreiber and Rong Pan