DSCI 6007

Distributed and Scalable Data Engineering

Project: Collaborative Filtering Using the Netflix Data

Desire Matouba

December 15th 2021

Dr. Vahid Behzadan

Table of Contents

Abstract

This project is about designing, building, and customizing a recommendation model using Netflix data. A recommendation model may be defined as an algorithm whose goal is to provide the most important and accurate information about users experience of a particular product or service. Multinational companies such as Netflix have a deep interest to get the most up to date data about user interactions, the average time spent on Netflix (which may be on a monthly, weekly, daily basis). Since customers around the world use Netflix, it is not surprising that data about customer experience may be segmented by country, regions or states within countries, gender, age, and other demographic variables. Netflix as a company has worked very hard to reach the point where they are today: the goal is to continue to learn about customer experience, expectations, likes, preferences to better serve them and customize the content of films, movies, and other programs available on Netflix.

Introduction

This project revolves around collaborative filtering, which is a specific type of Recommender System. Recommender System is a as is a complex information processing tool that used to predict which a product or service that users would probably like. With that precious information, the goal is to suggest additional products and services to users what on what they already like. Collaborative filtering is based on the user past behavior and not on potential future additional data. An analogy can be made with YouTube videos. Once YouTube identifies some topics or areas of interests based on the videos users tend to see on the website, the company automatically suggest videos that may be appealing to customers based on their browsing history.

In this report we use two methods to predict movie rating based on the models we built: Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE). RMSE is the square root of the mean of the square of all of the error in a particular set of data. RMSE is an excellent general-purpose error metric for numerical predictions because it accounts for both positive and negative differences in the same manner by squaring these differences. MAE is the mean of the absolute values of the individual prediction errors on over all instances in the test set. Each prediction error can be seen as the difference between the actual value and the predicted value for an instance.

Steps to Connect to Jupyter Notebook

We are using Spark from AWS to work on this project. Instead of using the more known Jupyter Notebook that can be opened with Anaconda, we use AWS to connect to the Jupyter Notebook. The first step is to create an EMR Cluster by specifying the Spark option. This will enable to use Spark to later connect to Jupyter Notebook. It takes about 10 minutes for the cluster to be ready to be used. We also create a key-pair that is linked to the EMR Cluster. The key-pair is converted from a .pem file to a .ppk file by using PuTTygen. It is worth mentioning that both the .pem and .ppk files must have the same name for the PuTTY connection to work. Once is cluster created is completed, we add the security nodes 22 and 8888 to the master instance located in AWS. These two nodes are mandatory to connect Spark to Jupyter Notebook. we open Putty to connect the information from the EMR Cluster. We use the form "hadoop@EMR Cluster public DNS" to enable the Putty connection. After cluster is created, and we connect the cluster to Putty as a way to launch Spark from Putty. With Putty, we can easily connect to Jupyter Notebook and work from there. The text files provided as part of the project are uploaded into a bucket in the AWS interface.

Once we are in PuTTy, we run some codes to install Spark, Jupyter, and some other needed libraries. These installations are made into the root user. Once the installation is over, we exit the root user and go back to the standard user. At that point, we run the code "which jupyter" to check the location of the Jupyter installed in the root user. Once the location of Jupyter is confirmed, we run some additional codes that allows to identify both the location and the node 8888 that was added into the master instance. At that point, we are ready to launch Spark. The code to launch spark provides a URL that is copied and pasted into the web browser. The first part of the URL is deleted and is replaced by the ERM Cluster Public DNS Name. When we run the URL we are finally connected to Jupyter Notebook through Spark.

Going Through the Jupyter Notebook

There are 3 problems the project attempts to solve.

Problem 1: Collaborative Filtering Approach

The first problem involves getting all the needed libraries (matplotlib, Pandas, NumPy, Spark, and others), reading the files of interests that were uploaded into the AWS bucket (train, test, and movie). Next, we transform these three datasets into Pandas DataFrame and we do some data cleaning (such as removing NaN and zeros) to make sure that the data is ready to use. Furthermore, we do data caching of these three datasets. Caching is a technique of storing frequently used data/information in memory. To finalize the first problem, we calculate the RMSE of movie rating predictions on the training data of 3 ranks that we customize: 4, 8, and 12. We find out that Rank 8 is the best among the three models because it has the lowest RMSE of about 0.857. Finally, we also calculate the RMSE of movie rating predictions on the test data and it is roughly 0.871.

Problem 2: Analyzing the Netflix Data

First Question: how many distinct users present in the test set.

We find out that there are 27,555 distinct items (movies in this case) and 1,701 (distinct users) respectively.

Second and Third Question: user similarities measured by overlapping items and item similarities measured by overlapping users, and which one is the best.

For these questions we create 2 DataFrames: one which provides the average user rating of five movies (we get an average of 2.44), and another which provides the average rating of one movie by five different users (we get an average of 3.98).

It would seem like the movie (item of interest) rated by the user is the more objective measure. The reason is that different users will provide different ratings. At the end, the movie rating is the average of all the users, in this case 5 users. Although one is free to disagree with the average movie ratings, this rating does give an idea of how the users view the movie. On the other hand, the user that rate a movie (item of interest) may be very bias depending on personal taste, likes, and preference which are all highly subjective matters.

Fourth Question: Whether normalized ratings are used
For this question, we decide to use normalize ratings so that all the ratings can be between 0 and 1. For each user rating, a particular rating is subtracted from the mean and divided by the standard deviation. It makes a more fruitful comparison.

Problem 3: Collaborative Filtering Implementation

We choose the SVD approach for this problem because it shows the decomposition of key matrices U, S, and Transpose. We create a function to predict movie recommendation and we obtain a mean absolute error for the training data of 0.65, which is not too bad. After that we build a model for the test data and we obtain and by using the SVD, we obtain an RMSE of 0.16 and a MAE of 0.022. This is excellent because the MSE of the test data must be better than that of the training data. This confirms that the model built to predict movie recommendation significantly helps to reduce the two types of errors.

Conclusion

This project was a great opportunity to use collaborative filtering as a means to predict user movie ratings using the Netflix data. A comprehensive process of the motivation of the project, as well as the steps to connect from AWS to Jupyter Notebook, and the questions of the problems were addressed in this report. We find out that the application of the movie recommendation rating was successful in considerably reducing the RMSE and MAE of the test data while using the SVD method. Since we are most concerned about the test data, the fact that the training data RMSE was higher is not a major concern. It was a great opportunity to analyze Netflix data using Spark.

References

1. ScienceDirect RMSE
https://www.sciencedirect.com/topics/engineering/root-mean-squared-error

2. Springer Link MAE
https://link.springer.com/referenceworkentry/10.1007%2F978-0-387-30164-8_525