CHARITY MATCHING

Dhanashri Tidke: 9884218205
Jeffy Merin Jacob: 6385224167
Seher Khan: 9273030117

ABSTRACT

DonorsChoose.org is an organization that has funded over 1.1 million classroom requests through the support of about 3 million donors. Out of the 3 million donors, a majority of them were making their first ever donation to a public school. We believe that if even a fraction of the donors are motivated to make another donation, it would have a really huge impact on the number of requests that are fulfilled [0] With this motivation in mind, we have aimed to build a recommendation system that would recommend the right projects of interest to donors. We build two recommender systems based on the following approaches: Content based filtering and Collaborative filtering. We conclude by performing a comparative study of two approaches.

INTRODUCTION

DonorsChoose.org was founded in 2000 by a Bronx history teacher. It has managed to raise $685 million for America's classrooms. There were teachers from three-quarters of the public schools in the U.S who had come to DonorsChoose.org to make requests of the need of their students, making DonorsChoose.org the leading platform for supporting public education. But the sad part is teachers still spend more than a billion dollars of their own money to fund classroom materials. To make it possible for students to receive what they need to learn, the team at DonorsChoose.org want to be able to connect donors with projects that inspire them the most.

1. PROBLEM STATEMENT

Our aim with this project is to build a recommender system that recommends right projects to donors so that the donation amounts received by DonorsChoose.org is maximized. For instance, if a donor named Alice donated to one or more projects, we will be able to recommend more projects to her based on her donation history and the donation history of other donors who are similar to her.

We apply two approaches to the design of recommender system: one common approach is **Content-based filtering.** Content-based filtering methods are based on a description of the item and a profile of the user's preferences [1]. In our case, we consider the description of the project and the profile of the donor.

The other approach that has wide use is **Collaborative filtering** [2]. Collaborative filtering methods are based on collecting and analyzing a large amount of information on users' behaviors, activities or preferences and predicting what users will like based on their similarity to other users. Collaborative filtering is based on the assumption that people who donated to same projects in the past will donate to similar projects in the future.

2. THE DATASET

The dataset from Kaggle has a total of 6 csv files namely: Donations.csv, Donors.csv, Projects.csv, Resources.csv, Schools.csv and Teachers.csv.

TABLES (SIZE)	DESCRIPTION (MAIN ATTRIBUTES)
Donations (4.69m x 7)	Date, donor, amount and benefiting project of each donation
Donors (2.12m x 5)	Location of donor and whether or not donor is a teacher
Projects (1.11m x 18)	Teacher, School, Date posted and expiration date, Further details (Project Type, Title, Essay, Short Description, Need Statement, Category, Grade Level, Cost, Project Posted Date, Project Current Status (Expired/Fully Funded/Live), Project Fully Funded Date
Resources (7.21m x 5)	Name, quantity, price, vendor of resource to be bought for a particular project
Schools (73.0k x 9)	Name, type of area % of students on free lunch, location of the school
Teachers (403k x 3)	Date when a particular teacher first posted a project on donorschoose.org

The key csv files that are used by the two approaches are only three namely: Donations.csv, Donors.csv, Projects.csv.

"Projects"

Project ID	School ID	Teacher ID	Teacher Project Posted Sequence	Project Type	Project Title	Project Essay	Project Short Description	Project Need Statement	Project Subject Category Tree	Project Subject Subcategory Tree	Project Grade Level Category	Project Resource Category	Project Cost	Project Posted Date	Project Expiration Date	Project Current Status	Project Fully Funded Date
7685f0265	e180c7424cb	4ee5200e89d9	25	Teacher-Led	Stand Up to Bullying: Together We Can!	Did you know that 1-7 students in grades K-12 ...	Did you know that 1-7 students in grades K-12 ...	My students need 25 copies of "Bullying in Sch...	Applied Learning	Character Education, Early Development	Grades PreK-2	Technology	361.80	2013-01-01	2013-05-30	Fully Funded	2013-01-11
						Help us have a	Help us have a fun,	My students need a	Applied	Early							

"Donors"

Project ID	Donation ID	Donor ID	Donation Included Optional Donation	Donation Amount	Donor Cart Sequence	Donation Received Date
000009891526c0ade7180f8423792063	6887291208586662212085 29ee3fc18e	1f4b5b6e68445c6c4a0509b3aca93f38	No	178.37	11	2016-08-23 13:15:57
000009891526c0ade7180f8423792063	dcf1071da3aa3561f91ac689d1f73dee	4aaab6d244bf3599682239ed5591af8a	Yes	25.00	2	2016-06-06

"Donations"

Project ID	Donation ID	Donor ID	Donation Included Optional Donation	Donation Amount	Donor Cart Sequence	Donation Received Date
000009891526c0ade7180f8423792063	6887291208586662212085 29ee3fc18e	1f4b5b6e68445c6c4a0509b3aca93f38	No	178.37	11	2016-08-23 13:15:57
000009891526c0ade7180f8423792063	dcf1071da3aa3561f91ac689d1f73dee	4aaab6d244bf3599682239ed5591af8a	Yes	25.00	2	2016-06-06

Each table in the dataset has over a million rows and contains information such as the ID of the donor, the city the donor is from or the description of the project in need of donations. We will see in the following section how these features are used to make recommendations to donors.

2.1 DATA PREPROCESSING

We start by doing some exploratory data analysis on the data to see if the data is balanced, to handle null values and remove duplicates. The biggest challenge in preprocessing the data was the presence of duplicates rows in the Projects, Donations and Donors table. We had been performing manipulations on the table assuming that the ID columns in each of the tables were unique, it turned out not to be the case. We had to therefore handle duplicate records to ensure that our results were consistent. We also saw that in many cases multiple donations had been made to the same project by the same donor. For such instances we replaced the multiple donations with a single donation of the summed amount.

Once we have a clean dataset to work with, we perform an outer join on the 3 tables and choose only features that are required by the two approaches for recommender systems.
After data preprocessing we had two final datasets. The first one was the **Projects Table**. This consisted of **1,110,015** records with six features: Project ID, Project Title, Project Type, Project Resources Category, Project Subject Category Tree and Project Subject Subcategory Tree.
Due to the challenge we faced in handling large data (as described in the Challenges section), we decided to direct our recommendations to donors of a particular demographic region, in our case, donors who are based in Oakland. As a result our second table, the **Donations by Donors** table had **15,924** records with seven features: Donor ID, Project ID, Project Title, Project Type, Project Resources Category, Project Subject Category Tree and Project Subject Subcategory Tree.

Below is a tabulation of all the columns we use in the Donations by Donor table and what their definitions with examples:

Donor ID	Unique ID of a donor
Project ID	Unique ID of a project
Donation Amount	Total amount donated for a project
Donor City	The donor's city
Project Title	The title of the project
Project Type	Project types like Teacher-led
Project Resources Category	Resource category like Technology, Supplies and Books.
Project Subject Category Tree	Subject categories like Applied Learning, Special needs.
Project Subject Subcategory Tree	Subject Sub Categories like ESL, Literacy

3. CHALLENGES

Some of the challenges we faced with the dataset were ones we did not see coming. As mentioned in the previous section, we faced a few challenges in preprocessing the data. One would usually assume that the ID column uniquely identifies the rows in a table. We came to know that this wasn't the case with our dataset, when merging of the tables gave us inconsistent results. So, we were faced with the task of identifying rows that were not entirely duplicates i.e. one of their columns had a different value. These were rows that were not dropped when we performed drop duplicates in Python. So, we had to manually find and delete them.

Another challenge we faced that we don't see commonly in other movie recommendation systems is that our dataset did not have a column that described the rating for a project by a donor. Rating place a crucial role in recommendation systems design by Collaborative filtering. Our best bet was to create a rating of our own that would best describe a donor's liking towards a project. To calculate this, we group all projects donor has donated to and get the total amount the donor has donated to all projects. We then divide the donation amount donor has donated to a particular
project by the sum of his total donations. This way we had calculated the rating of a project by a donor. We also tried to use the latent factor method for collaborative filtering, particularly ALS. The latent factor method assumes that both donors and projects live in low dimensional space describing their property. We recommend projects based on its proximity to the donors in latent space. If R is a matrix consisting of donor ID's as rows and Project Type ID's as columns, where,

$$R\,[\text{donor}_i, \text{project}_j] = \frac{\textit{Amount donated by donor } i \textit{ to project } j}{\textit{Amount donated by donor } i \textit{ to all projects}}$$

The final R' = U * V approached a zero matrix.

Working with large size of data was also challenging. Loading the initial tables with millions of rows of raw data albeit slow, worked. However, pivoting the data to create the utility matrix and then saving the large resulting file was much slower.
For comparison, consider the raw data file *Projects.csv* of size 2.6 GB. This file contains 1.11 million records, each with 18 features (including essay). Loading this file into python takes less than 50 seconds. On the other hand, when we created a utility matrix of donors only from Los Angeles (using 46,717 rows and 2 features - Donor ID and Project ID), the resulting utility matrix file (2.1 GB) took upto 10 minutes to save, and would take indefinitely long to load (often resulting in a MemoryError). We attributed this issue due to a large number of columns in the utility matrix (in tens of thousands). To tackle this problem we worked with donors from the city of Oakland which had a moderate amount of data.

Another problem that occurred during utility matrix construction was that the utility matrix of our chosen subset (donors from Oakland) was 99.98 % sparse and only a handful of overlaps existed between donors in terms of projects. As a result the matrix was not suitable for

collaborative filtering. We overcame this issue by working with a "Project Type ID" which is defined in section 5.1.1.

4. APPROACH I - CONTENT-BASED FILTERING

The first approach is Content based collaborative filtering. Here we strive to find projects that are similar to the project(s) that the donor has already donated to. We calculate the similarity between projects based on metadata (Project Title', 'Project Type', 'Project Resource Category', 'Project Subject Category Tree' and 'Project Subject Subcategory Tree) and/or text features extracted from project titles and descriptions.

4.1 PROCESS TEXT DATA

We will use a very popular technique, TF-IDF, to extract information from project title and descriptions. TF-IDF converts unstructured text into a vector structure, where each word is represented by a position in the vector, and the value measures how relevant a given word is for a project title/description. It is used to compute similarity between projects based on project titles and descriptions.

4.2 BUILD DONOR PROFILE

To build a donor's profile, we take all the projects the donor has donated to and average them. The average is weighted by the event strength based on donation times and amount. In other words, the project the donor has donated more money will have a higher strength in the final donor profile.

4.3 BUILD PROJECT PROFILE

To build a project profile, we take the corresponding entry of the project id in the TF-IDF matrix that we build during the preprocessing step. This represents the relevance of the project with respect to various domains i.e. project related terms that TF-IDF computes. For instance, a project with music related terms will have a profile with words like musical instruments, piano, etc. having a higher score than words like books, seeds, etc.

4.4 BUILD THE CONTENT-BASED RECOMMENDER

The Content Based Recommender recommends the top 10 (by default, can be modified as a parameter to our algorithm) projects to a donor by computing the cosine similarity between donor profile and all project profiles, getting the top 10 similar projects and sorting them by similarity. By default, the recommendations include the projects that the donor has already donated to. However, an extra parameter can be provided to ignore the those projects.

5. APPROACH II – COLLABORATIVE FILTERING

Collaborative filtering is a method of making automatic filtering about the interests of a user by collecting preferences from many users (collaborating). The underlying assumption of the collaborative filtering approach is that if a person A has the same opinion as a person B on an

issue, A is more likely to have B's opinion on a different issue than that of a randomly chosen person.

5.1 LOCALITY SENSITIVE HASHING

The general problem in recommendation systems can be cast as, given an $\{item_i, user_j\}$ determine the most likely rating. A common framework for generating such predictions is the neighborhood-based approach. Some of the methods commonly used for neighborhood-based computation are: KNN, K-means, k-d Trees and Locality Sensitive Hashing. In the subsequent sections, we explore the use of Locality Sensitive Hashing for the dataset.

5.1.1 Construction of the Utility Matrix

The utility matrix used for LSH was of donors belonging to Oakland. To avoid using the matrix of Donor IDs by Project IDs with sparsity of 99.91% and a lack of overlap between donors in terms of projects, we defined a feature called "Project Type ID". This was constructed by concatenating the columns Project Type, Project Resources Category, Project Subject Category Tree and Project Subject Subcategory Tree, and turning the resulting feature into a categorical variable with unique integer values.

Our utility matrix then had 1781 rows (i.e. unique Donor IDs) and 1303 columns (i.e. unique Project Type IDs). The matrix was filled with 1 where the donor on the row had made at least one donation to the project on the column, and 0 where the donor had made no such donation.

This matrix was 99.91% sparse.

5.1.2 Compute donor signatures

We partitioned the utility matrix into 10 chunks and computed the minhash signatures of donors in each partition. We used 20 hash functions to permute the columns (since columns represented Project Type IDs). The hash functions were of the form,

$$h(x, i) = (5x + 13i) \bmod n\_projs \qquad i \in \{0, 1, 2, ..., 19\} \text{ and } n\_projs = 1303$$

As a result, the computed signature of each donor was of length 20.

5.1.3 Find similar donors

Next we repartitioned our data and proceeded in the following way. For each donor, we split the signature into 4 bands of 5 rows each. A pair of donors is considered a candidate pair if it has the same rows for at least one of the bands. We then calculate the Jaccard Similarity for the candidate pairs. If the Jaccard similarity is >= 0.5, the two pairs will be considered similar.

We had maintained a dictionary called "projs_of_donors_in_chunks". This had donors as keys and lists of projects a donor had donated to as values. This data structure made it easier to calculate Jaccard Similarity of the candidate pairs using the original donors (as a set of projects) rather than their signatures. Consequently this saved us from adjusting the number of rows and bands to find the right threshold to reduce false positives and false negatives. Using original donors for calculating Jaccard meant that our false positive and false negative rates were both zero.

5.1.4 Recommend projects

Our next task was to recommend projects to a given donor, say donor A. To do this we got the list of similar donors of A and then identified the set of projects these donors had contributed to. A was then recommended projects from this set to which he or she had not already contributed to. The recommendations were ranked by frequency of donation by similar donors.

6. APPLICATIONS

6.1 Application of Content-Based Approach in Research Paper Recommendation System for a Digital Library

Content based technique is known for its suitability in domains where the number of items considered are more than the number of users. One of the major problems library users encounter when using the library is finding favorite digital objects from a large collection of available digital objects. A solution to this problem is to build a system that users can use to locate quickly items of interest in a digital library containing a large collection of items. To determine how relevant or similar a research paper is to the user's query or a user's profile of interest, TF-IDF (Term Frequency Inverse Document Frequency) and cosine similarity are used. The user's query and research paper are represented as vectors of weighted using something known as the Keyword based vector space model. The weights, in this case, represent degree of association between a research paper and user's query. Integrating recommendation feature in digital libraries will help library users find more relevant research papers adhere to their needs.

6.2 Recommendation System for Netflix

We know the most popular use of a recommendation system is in the area of entertainment. Recommending similar movies when a user views or rates another movie they've watched is a system that every big digital entertainment company has employed. Recommendation systems are a big part of Netflix. They aim at providing useful suggestions of products to online users to increase their revenue from users who use their websites. It is based on the assumption that people usually select new products based on what they've heard to be good or maybe their friend had recommended it. It may also due to comparison of similar products or feedback from other users.To provide the best suggestions that suit the client's needs a recommendation system is used to automatically handle this task. This in turn, offers personalized content to users based on their past behavior and it makes sure customers keep coming back to the website [5]

7. MODEL EVALUATION

7.1 Content Based Filtering

For our recommendation system, we chose to work with **Top-N accuracy metrics**, which evaluates the accuracy of the top recommendations provided to a user, comparing to the items the user has actually interacted.

In this evaluation method, for each donor, for each project the donor has donated to in the test set, we obtain a sample of 1000 other projects the donor has never donated to.

The content based recommender then produces a ranked list of recommended projects, from a set composed one donated project and the 100 non-donated projects.
We then compute the Top-N accuracy metrics for this donor and donated project from the recommendations ranked list and aggregate the global Top-N accuracy metrics.

We use Top-3, Top-5 and Top-10 accuracy metrics to evaluate the model in terms of **Recall**. Recall is the rate of the donated projects that are ranked among the Top-N recommended projects when mixed with a set of non-relevant projects.

Following is the model evaluation metric for our content based system.

Top-3 Recall : 0.989
Top-5 Recall : 0.995
Top-10 Recall : 0.998

7.2 Collaborative Filtering

Since our utility matrix was 99.91 % we were unable to make a test set and calculate precision and recall. Instead we came up with our own metric (which ranges from 0 to 1) to evaluate the similarity of recommended projects to actual projects using LSH.

$$\text{metric} = \sum_{i=1}^{D} \frac{\sum_{j=1}^{|P*_i|} sim(p'_j, P_i)}{|P*_i|}$$

Where D is the total number of donors,
P_i is the set of actual projects of the ith donor,
P^*_i is the set of recommended projects to the ith donor and $p'_j \in P^*_i$,
and $sim(p'_j, P_i)$ is defined as explained below.

$sim(p'_j, P_i)$ = average number of attributes of p_j' that exactly match at least one project in P_i (from among Project Type, Project Resource Category, Project Subject Category, Project Subject Subcategory Tree).
E.g.
Let P_i be the following,

	Project ID	Project Title	Project Type	Project Resource Category	Project Subject Category Tree	Project Subject Subcategory Tree
0	614fb2b1104496e090b01b7811c4b0ba	Kindergarteners Need "Level C" Library Books!	Teacher-Led	Books	Literacy & Language	Literacy
1	322f5421a84e6907ab0fa46b84bcd203	Kindergarteners Need Level B Library Books	Teacher-Led	Books	Literacy & Language	Literacy
2	9931ff1527205fa99ecba36272192289	Kindergarteners Need Writing Journals Too!	Teacher-Led	Supplies	Literacy & Language	Literacy, Literature & Writing
3	e57f981ca6fd98c6ffa2a4974c95f5b8	We P-L-C Ourselves Achieving!	Professional Development	Books	Math & Science	Mathematics

and let the following be p_j' from P^*_i

2	b27658f14381ace089843f554ed367d4	Help Oakland Students Connect Reading to the R...	Teacher-Led	Books	Math & Science, Literacy & Language	Health & Life Science, Literacy

Then the $sim(p_j', P_i)$ = ¼ = 0.25 since only the value of "Project Resource Category Tree" (i.e Books) appears at least once the corresponding column in P_i. Note that this measure understates

similarity, e.g. p_i' has "Math & Science, Literacy & Language" in its "Project Subject Category Tree" and at least one project in P_A project has either "Math & Science" or "Literacy & Language" in the same column; but this does not add to the similarity measure.

Our value of this metric comes out to be 0.39 which while low can be thought as a lower bound to the similarity (and usefulness) of recommended projects to actual projects of all donors.
Also note that for each donor i, and each recommended project p'_j we found that $sim(p'_j,P_i) \geq 0.25$. This means that even in the worst case, each recommended project matches some actual project exactly on at least one attribute.

8. BIBLIOGRAPHY

[0] https://www.donorschoose.org/about
[1] *Aggarwal, Charu C. (2016). Recommender Systems: The Textbook. Springer*
[2] John S. Breese; David Heckerman & Carl Kadie (1998). Empirical analysis of predictive algorithms for collaborative filtering. In Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence (UAI'98).
[3] Neighborhood based methods for Collaborative filtering, Ramesh Dommeti
[4] Application of Content- Based Approach in Research Paper Recommendation System for a Digital Library, Simon Philip, P.B. Shola, Abari Ovye John
[5] Recommendation System for Netflix, Leidy Esperanza MOLINA FERNÁNDEZ