Ramesh Parthasarathy

Capstone Project

Definition

- **Project Overview:**

DonorsChoose.org is an organization, having a user friendly website, act an intermediary between the teachers who need sponsorship for their classroom projects and the donors who want to contribute to the classroom projects. They are the leading platform for supporting public education.

With a view to maximize the sponsorship of open projects, they want to launch a periodical email campaign to the donors. For an effective email campaign, potential donors should be matched with the projects.

They have launched a competition with kaggle to achieve this objective of matching donors and projects. However I am limiting the scope of the assignment to identify related projects.

I plan to use ***Sklearn's TfidfVectorizer*** & ***linear_kernel*** for document classification using "project short description" and "project essay" fields in the project dataset. The goal of this classification is to identify the document similarity between a document of a project with documents of other projects using TF(Term Frequency)-IDF (Inverse Document Frequency) methodology.

The following factors need to be considered.

1) There are millions of projects each having their own free text document ("project short description" and "project essay" fields in the project dataset).
2) Identify top N similar projects for a given project, based document similarity, subject to an user defined similarity score (e.g. show me all projects where the document similarity >=0.7)

3) Document similarity of a project is computed with respect to other related projects document using ***Sklearn*** library.

- ***Problem Statement***:

  The data is available in a relational database format (i.e. in a normalized form)

  - Donations: Stores donation records with reference of projects and donors.
  - Donors: Stores all unique donors.
  - Projects: Stores all unique projects.
  - Resources: Stores all resource requirements.
  - Schools: Stores all unique schools.
  - Teachers: Stores all unique teachers.

  Current status: With available data, it is now possible in the user interface to broadly filter all projects for different criteria such as city, county, state, subject such as Math & Science, target age group, resource requirements, amount needed etc, Users can also search for specific key words to get the list of projects containing the search terms.

  My inputs: Project short description & project essay fields play a pivotal role in presenting the primary objective of the project & appealing to the donor. I don’t think any donor will be willing to donate to a project without going thru the project essay.

My Proposal: Essentially, the problem on hand is to identify a given project with other related projects in terms of their "project short description" & "project essay" fields.

I plan to apply the following *strategy* in solving this problem.

1. Extract fields "project short description" & "project essay" into a document list.
2. Using sklearn's TfidfVectorizer instance, apply fit & transform the above document list into a tf-idf matrix representation the documents.
3. Compare one document with other document using tf-idf matrix by applying linear kernel & deriving the cosine similarity.
4. Pick the top N projects after sorting the data in descending order.
5. Aggregate the results into another dataset (related projects dataset) and persisting the results to a datastore.
6. It is now possible to derive the related projects after applying the minimum cosine similarity score & and the original projects dataset using project id key column.

- **Metrics:**

Here we will comparing text fields between projects, document similarity will become the basis for comparison. This is usually expressed in terms of cosine similarity whose value will have a range of 0 to 1. It is my plan, for a given project, to derive cosine similarity top n projects (where n is a variable). This will form a separate dataset (related projects) which can be joined with projects dataset to derive the cosine similarity metrics. There will one-to-many relationship between project and related projects datasets.

Document Similarity concept

$$d = (x_1, x_2, x_3, \ldots, x_n)$$

is a document vector in an n-dimensional vector space

Length of d is given by (extension of Pythagoras's theorem)

$$|d|^2 = x_1^2 + x_2^2 + x_3^2 + \ldots + x_n^2$$

$$|d| = (x_1^2 + x_2^2 + x_3^2 + \ldots + x_n^2)^{1/2}$$

If d_1 and d_2 are document vectors, Inner product (or dot product) is given by

$$d_1 . d_2 = x_{11}x_{21} + x_{12}x_{22} + x_{13}x_{23} + \ldots + x_{1n}x_{2n}$$

Cosine angle between the docs d_1 and d_2 determines doc similarity

$$cos(\theta) = \frac{d_1 . d_2}{|d_1||d_2|}$$

$\text{if } cos(\theta) = 1;$ documents exactly the same

$\text{if } cos(\theta) = 0;$ documents totally different.

Analysis

- Data Exploration:

  I am going to focus on Projects dataset for the purposes of this submission.

Here are the columns of the dataset.

```
['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',
'cost', 'Posted Date', 'Posted Year', 'Posted Month’]
```

Column `'Project ID’` uniquely identifies a row.

Columns `'Project Essay'`, `'Project Short Description’` & `'Project Current Status’` are the fields of interest.

- Dataset Visualization:

Projects dataset

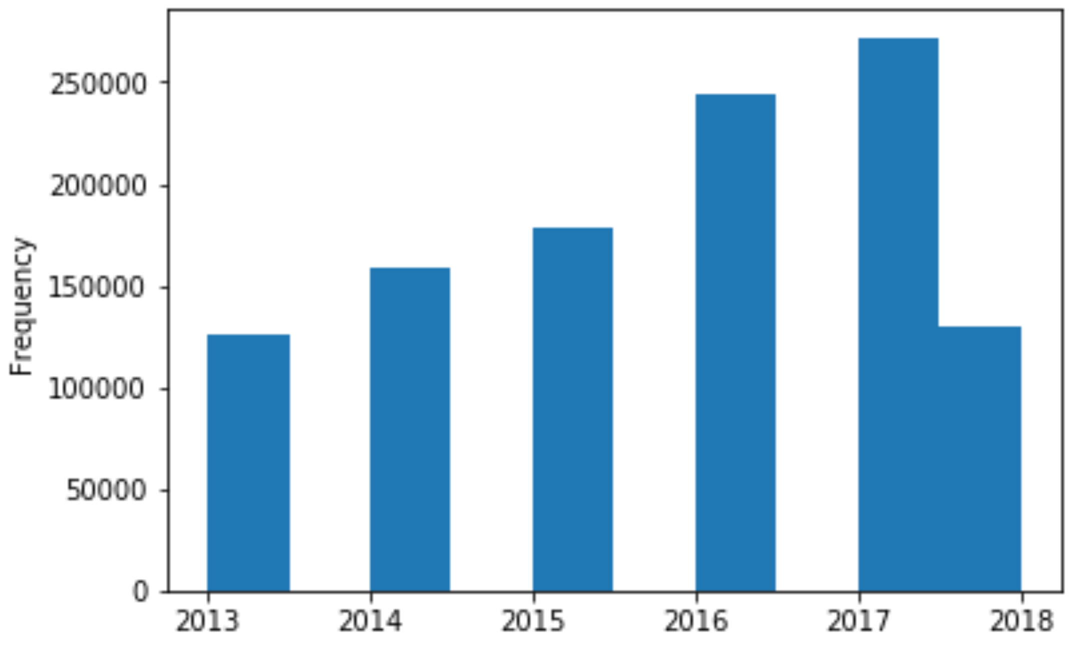
1. Total record count : 1,110,017
2. Unique values of 'Project Current Status’ field:

   ['Fully Funded' 'Expired' ‘Live']
3. Record counts by project current status

```
                        Project ID
Project Current Status
Expired                     241402
Fully Funded                826764
Live                         41851
```

4. Years for the projects current status.

   Note: Project Year is a derived field and its details are covered "Methodology" section.

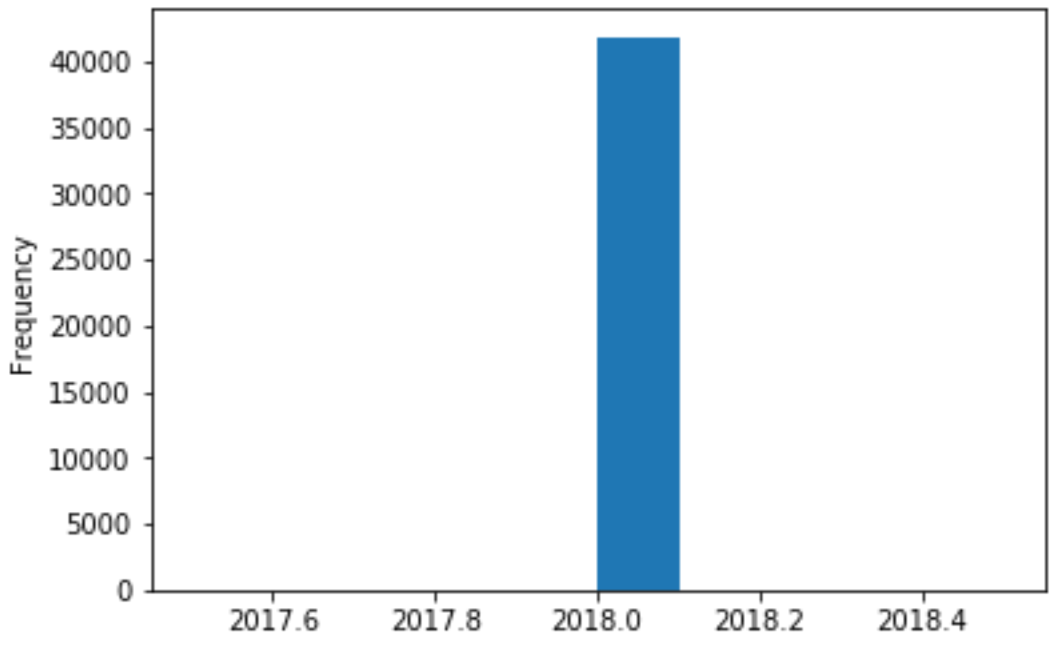


By Studying the above, it is clear that we will be interested in projects which have project current status as Live as they are ones which matter. All other projects are either fully funded or expired.

So I have created another active projects dataset from projects dataset where the projects current status is "Live"

5. Verify the years contained in this active projects dataset.

   ```
   active_projects_df["Posted Year"].plot(kind='hist')
   ```



It is clear that by selecting projects with "project current status" as "Live", we find that these projects have 2018 as the project year which is current.

A snapshot of the dataset is shown below showing a few select columns. Since I had difficulty in showing sample data in usual column format due to limited page width , I transposed the table to show in row format.

School ID	e180c7424cb9c68cb49f141b092a988f	08b20f1e2125103ed7aa17e8d76c71d4	1287f5128b1f36bf8434e5705a7cc04d	900fec9cd7a3188acbc90586a09584ef	3b200e7fe3e6dde3c169c02e5fb5ae86
Teacher ID	4ee5200e89d9e2998ec8baad8a3c5968	cca2d1d277fb4adb50147b49cdc3b156	6c5bd0d4f20547a001628aefd71de89e	8ed6f8181d092a8f4c008b18d18e54ad	893173d62775f8be7c30bf4220ad0c33
Project Type	Teacher-Led	Teacher-Led	Teacher-Led	Teacher-Led	Teacher-Led
Project Title	Stand Up to Bullying: Together We Can!	Learning in Color!	Help Second Grade ESL Students Develop Languag...	Help Bilingual Students Strengthen Reading Com...	Help Us Make Each Minute Count!
Project Essay	Did you know that 1-7 students in grades K-12 ...	Help us have a fun, interactive listening cent...	Visiting or moving to a new place can be very ...	Students at our school are still working hard ...	"Idle hands" were something that Issac Watts s...
Project Short Description	Did you know that 1-7 students in grades K-12 ...	Help us have a fun, interactive listening cent...	Visiting or moving to a new place can be very ...	Students at our school are still working hard ...	"Idle hands" were something that Issac Watts s...
Project Current Status	Fully Funded	Expired	Fully Funded	Fully Funded	Fully Funded

- Algorithms and Techniques

(Reference https://appliedmachinelearning.blog/2017/02/12/sentiment-analysis-using-tf-idf-weighting-pythonscikit-learn/)

Conventionally, histogram of words are the features for the text classification problems. In general, we first build the vocabulary of the corpus and then we generate word count vector from each document which is nothing but frequency of words present in the vocabulary. Most of them will be zero as a single document won't contain all the words in the vocabulary.

There are limitations in this conventional approach of extracting features as listed below:

a) Frequently occurring words present in all documents of corpus will be treated equally like other distinguishing words in the document.

b) Stop words will be present in the vocabulary if not processed properly.

c) Rare words or key words which can be distinguishing will not get special weight.

tf-idf weighting factor which eliminates these limitations.

***TF (Term Frequency)*:** It increases the weight of the terms (words) that occur more frequently in the document.

$$tf_{ij} = \frac{f_{ij}}{m_i} \; when \; f_{ij} > 0$$

Where

tf_{ij} is the term frequency j in a document I

$m_i = max(f_{ij})$ i.e., m_i is the maximum frequency of any term in document i.

***IDF* (Inverse document frequency):** It diminishes the weight of the terms that occur in all the documents of corpus and similarly increases the weight of the terms that occur in rare documents across the corpus. Basically, the rare keywords get special treatment and stop words/non-distinguishing words get punished. It is defined as:

$$idf_j = log_2(\frac{n}{n_j}) + 1 \; n_j > 0$$

Here,

n documents in the corpus

n_j is the number of documents in which term j occurs.

tf is a intra-document factor which depends on individual document and idf is a per corpus factor which is constant for a corpus. Finally, tf–idf is calculated as:

$$t_{ij} = tf_{ij} \cdot idf_j = (\frac{f_{ij}}{m_i}) \cdot (log_2(\frac{n}{n_j}) + 1) \;\; n_j > 0$$

Applying the concept.

- ✓ Using sklearn’s TfidfVectorizer instance, apply fit & transform the above document list into a tf-idf matrix representation the documents.

- ✓ Pass an individual document's tf-idf matrix and overall tf-idf (above step) to sklearn's linear_kernel function.

  According to sklearn's documentation "*The function* ***linear_kernel*** *computes the linear kernel, that is, a special case of* ***polynomial_kernel*** *with degree=1 and coef0=0 (homogeneous). If x and y are column vectors, their linear kernel is:*

  $$k(x, y) = x^\top y\text{''}$$

  Essentially this is a SVM (Support Vector Machine) with Linear Kernel for ***text processing***.

- ✓ The result will a *dot product* of the individual matrix and overall matrix which is representative of the cosine similarity of the specific document with the rest of the documents.
- ✓ Sort the above matrix in the descending order and pick up the top N related records

- Benchmarks

  The following are my proposed benchmarks

  - The program should be able to compute the cosine similarity of a given project with top N other projects.

    The cosine similarity score is the measure for determining how far a given project is related to other projects. This entire effort is to identify this relationship between the projects.

- The process should be efficient and should be completed in a reasonable amount of time. (Less than 30 minutes for this given volume)

  Performance of the algorithm is a major benchmark for machine learning. It is usually a trade off between thee accuracy and speed. It should reasonably perform well.

Methodology:

- Data Preprocessing:

  I am going to primarily focus on projects dataset. It has project id, school id, teacher id as key fields. In addition to other fields available in this dataset, I am interested in fields “project short description” & “project essay”. This are the fields which will be utilized for computation of cosine similarity. Ideally, any project definition should have both the fields correctly populated with appropriate contents. I’ll use pandas data frame and I’ll be populating any null value in these fields with blank string.

  ```
  # clean up missing values
  projects_df['Project Essay'].fillna("", inplace = True)
  projects_df['Project Short Description'].fillna("", inplace = True)
  # Create additional fields for the purposes of querying.
  projects_df['Posted Date'] = pd.to_datetime(projects_df['Project
  Posted Date'])
  projects_df['Posted Year'] = projects_df['Posted Date'].dt.year
  ```

- Implementation

This is a high level sequence of processing

1. After preprocessing, filter project data frame for “project current status”= “Live”
2. Extract fields 'Project Essay', 'Project Short Description’ from the dataset into a document list.
3. Create an instance of TfidfVectorizer with suitable parameters and invoke fit_transform passing the above list as a parameter to obtain tf-idf matrix of the documents.
4. Iterate thru the projects and for each project, apply linear kernel on the project tf-idf matrix with overall tf-idf matrix to derive cosine similarity. Sort by descending oder and pick only top N related projects.
5. Partition the dataset and processes step 4 in parallel depending on the CPU availability in the machine. (I assumed a default of 2 CPU) and return the dataset.
6. Apply the minimum cosine similarity threshold on the related projects dataset and persist this dataset.
7. It is possible now to join the project dataset with related dataset to identify the related projects.

Improvements scope:

- For TfidfVectorizer (https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html) there are numerous parameters and I’ll discuss only those I would think are important for this project.
    1. lowercase: convert everything to lowercase.

2. analyzer: I have chosen "word" as the basis.
3. stop words: "English" will eliminate all stop words from the vocabulary.
4. max_df: ignore terms more than the given threshold. 0.9 is chosen value.
5. sublinear_idf: Whether to treat the term frequency as linear or apply log function to the term frequency.

For all other parameters, I chose to go with the default value.

- I initially tried with sublinear_tf as false. This essentially means TF will be treated as it is and it will carry a high weightage for repeated terms.
- Later tried with sublinear_tf as true. This means that TF will be applied a log function and it might carry as much weightage as linear term frequency.

My results are shown below.

You will notice that the cosine similarity decreases when we applied logarithmic term frequency. Since we are trying to present an

Related Projects with parameter sublinear_tf as False

Project ID	Related Project ID	Cosine Similarity
7de2c24972afb35b675c95341c9196a2	35cdda4e82a90df733e6017dba6a8dc2	0.99
7de2c24972afb35b675c95341c9196a2	28411c609459b01b4b58096f1299c155	0.73
7de2c24972afb35b675c95341c9196a2	75c1c726531c163f198d0447c4174966	0.72
7de2c24972afb35b675c95341c9196a2	f4ca78112adf488b2492d473521afb73	0.6
7de2c24972afb35b675c95341c9196a2	b202a1eb6828a6f2aab3a4109e571261	0.54

Related Projects with parameter sublinear_tf as True

	Project ID	Related Project ID	Cosine Similarity
675	7de2c24972afb35b675c95341c9196a2	35cdda4e82a90df733e6017dba6a8dc2	0.99
676	7de2c24972afb35b675c95341c9196a2	28411c609459b01b4b58096f1299c155	0.7
677	7de2c24972afb35b675c95341c9196a2	75c1c726531c163f198d0447c4174966	0.67
678	7de2c24972afb35b675c95341c9196a2	f4ca78112adf488b2492d473521afb73	0.56
679	7de2c24972afb35b675c95341c9196a2	b202a1eb6828a6f2aab3a4109e571261	0.49

optimistic view of the related projects, I would prefer to go with parameter sublinear_tf=False.

Results:

- Validation:

1. Model Evaluation and Validation

The assignment is to identify top N related projects for any given project. I have employed document similarity(TF-IDF) algorithm to identify the related projects using the words provided in 'Project Essay', 'Project Short Description' fields for each project using optimal parameters described in earlier section. Essentially, the model designed will compute the document similarity exactly the same way whether it is the current dataset or any future dataset.

I am simply utilizing sklearn's library for text classification and I have performed verification by validating the results to my satisfaction.

2. The final model and solution:

I have not made any dramatic changes to the model except to fine tune the parameters and I have identified the ideal parameter for this purpose. I am convinced that this model will consistently perform well.

Conclusion:

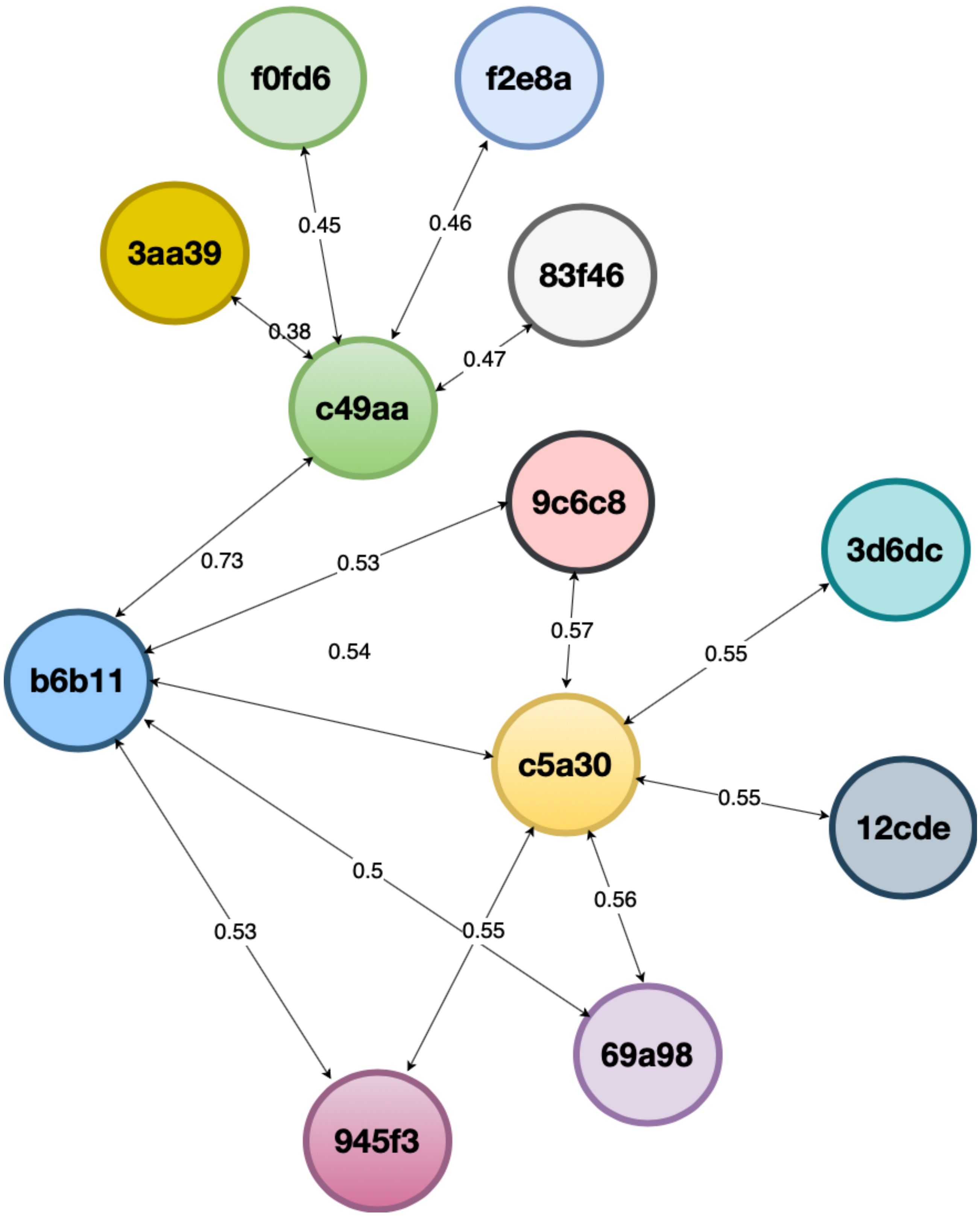
Visualization:

This visualization depicts how the projects are related to each other.

Legends:

1. Each node represents a project (used a 5 digit project id)
2. Each connector represents their cosine similarity score.
3. The number of related projects is assumed to be 5.

Only 3 interconnected projects are shown for simplicity.



Related Projects Visualization

Summary:

Usefulness of this deliverable: Prior to this project, the users could search for word(s) in the project and it will list all the projects containing the word. Considering that there are thousands of projects, users never had any way of picking up similar projects using search algorithm. Now it is possible to list related projects of a given project in terms of cosine similarity scores. This will enable donorschoose.org to present similar projects to the donors and allow the donor make informed choices about the donation amounts for different but similar projects. In their promotion emails, they could also list all the related open projects based on the profile of the donor.

Separate dataset: The final result set is an independent dataset but related to the projects dataset thru the project id (one-to many relationship). The related project id column is again references project id column in projects dataset. Essentially, there is no change in data structure of existing datasets and it is now possible to get similar projects by joining project dataset with related projects dataset.

Challenges:

✓ Sequential vs parallel processing: One of the challenges I faced was the processing time involved with the projects dataset. Initially, I was focussed on achieving the intended functionality first. After completing it, I found that sequential processing was taking very long time to complete. I did some research on alternative to sequential processing and found that python, indeed, supports parallel processing. So I implemented parallel processing with this project. It was a good learning experience implementing this feature.

✓ Process Timing: In order to optimize the processing time, I timed individual steps involved to identify the step which was taking a long time. I addressed those issues. After this fix, the average number of iterations went up from 2 iterations per second to 16 iterations per second.

Reflection:

✓ I have learnt a lot about document similarity, TF (Term Frequency), IDF (Inverse Document Frequency), matrix dot products, linear kernels etc. In addition to these machine learning features, I was thoroughly exposed about python programming features such as virtual environments, parallel processing, list comprehension etc. I am left with a sense of satisfaction with the deliverable I have produced.

Improvement scope:

✓ I am certainly a novice with machine machine learning & python programming. I am sure there is a large scope of improvement over what has been done in this project. I think it is worthwhile considering a derived feature (tf-idf metric) on "project short description" & "project essay" fields which can be used for predicting similar projects.

Quality:

Presentation: I tried to deliver this project to the best of my knowledge and satisfaction.