Data Science Fundamentals

(CSE 519 - Fall 2018)

FINAL PROJECT REPORT

Stony Brook
University

Dataset

We have used Citation Network Dataset [Citation-network V1, Citation-network V2, DBLP-Citation-network V3] which is extracted from DBLP, ACM, MAG (Microsoft Academic Graph). Each paper in the dataset is associated with abstract, authors, year, venue, id of references of paper, and title.

Data Cleaning

The main challenge we faced during data cleaning was to combine the data from multiple sources into one csv file. To solve this problem, we wrote a parser (which extracted the data from text files into a dataframe) to extract the fields into a dataframe depending on the format of the dataset file. Finally, all the data present in different dataframes was concatenated into one dataframe and we were able to get the data in csv file. Since we were not planning to use the publication venue field in the data, it was dropped. Also, there were many values (in the Abstract and Authors columns) which were not present in the data set, so we imputed those values with N/A. Moreover the datatype of Year column was float, so we converted data type of that column to integer.

Generation of Paper Table

After the basic cleaning was done, we planned on incorporating various new fields in the table which are discussed below:

1. Citation Count
    a. By iterating through the id of references of the paper in the whole dataset, we came up with the citation count of each paper and citation count was represented as a separate field named ‘citation_count’ in the dataset. We had taken the dictionary to count the number of references of each paper and if we got id of some paper in the reference list of other papers, we incremented the citation count of this particular paper.
2. Paper domain
    a. To find domain of each paper in the dataset was very challenging. To solve this task, firstly we came up with a list of common domains in which most of research papers got published. Some of these domains were Biomedical research, Chemistry, Biology, Economics, Computer Science, Business etc. And, to fetch domain for each paper, we used a Scumpy tool based on Named Entity Recognition(NER). The field ‘paper_title’ was sent to this tool and score was calculated after comparing each of the domains with the paper title. The domain with the maximum score was assigned to that particular paper.

Hence, after data cleaning, the following fields were present in the dataframe (which contains information about each paper in the dataset):

a. Abstract

b. Authors
c. Year
d. Index id
e. Paper title
f. Citation count
g. Domain

Generation of Author Table

To find top researchers from various domains, we have generated an author table and calculated our own metric with the help of various factors. The following fields were included in the author table:

1. Citation count: Total citation count for each author was calculated from the citation count present in the paper table which was described in the previous section. For each author in our dataset, we scanned the above generated table (paper information) to get all the papers published by the author and summing the citation count of each paper. Finally, the sum of all these citations give the total citation count for this particular author.
2. Number of papers published (Paper Count): Value of total number of papers published for each author was calculated from the paper table. We calculated number of papers of the authors by creating a dictionary for each author and incremented the value by considering ‘author name’ as the key in the dictionary.
3. H Index: Number of papers(h) with a citation number >= h indicates the h-index. It was computed for each author with the help of citation count calculated above.
4. I-10 Index: It refers to the number of paper with 10 or more citations. For every paper published by the author, we looked up the citation count for that paper in our paper table, and incremented the value of count if the citation value of that paper is more than 10. Finally, the value of count denotes our I-10 index for the author.
5. G-Index: It can be defined as the number of articles of the author for which the citation count is more than average number of citation for the author.
6. Average citations: It indicates average citations for each author and was calculated by dividing total number of citations divided by total number of papers.
7. Domain: Domain of each author was calculated based on the domain of the papers published by them.
8. Domain Score: Domain score is assigned corresponding to each author. It was calculated based on the number of papers published by a particular author in its domain and the total number of papers published by all the authors in that domain. Domain Score signifies about the popularity of the author in his/her domain. More domain score signifies more popularity of the author in his/her domain.

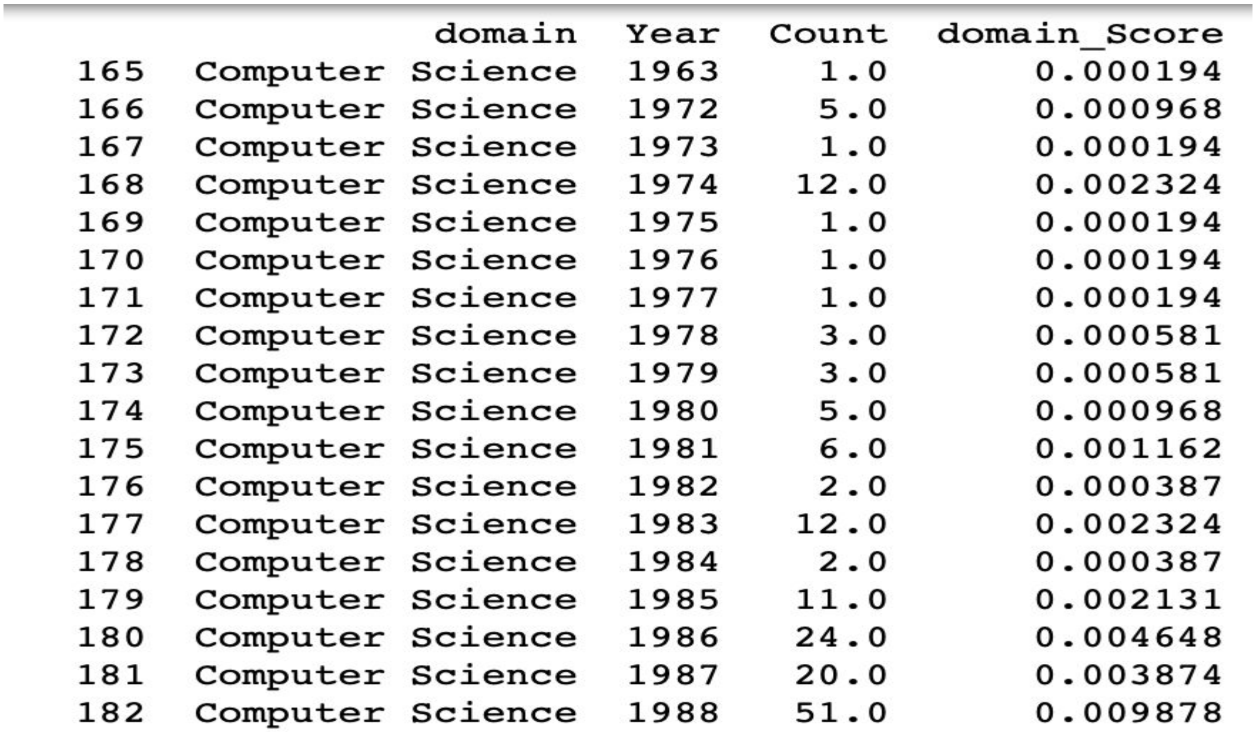
Author Metric (SVS Index) : We had given score to each author based on some parameters using a formula:

```
df_n["SVS - index"] = 0.3*df_n["average_citations"] + 0.3*df_n["paper_count"] +
0.5*df_n["citation_count"]  + 0.25*df_n["i10_index"] + 0.25*df_n["g_index"] +
0.25*df_n["domain_Score"]
```

We had given some weights to each important factor and evaluated the author's level score metric(which we called as SVS-index). The important parameters to be taken into consideration were citation_count, paper_count, average citations, g_index, i10_index and domain score. We had fine-tuned our parameters to change their weights by applying more complex models (lgm) on our data set and checked the importance of each feature given by our training model.

Generation of Domain Table

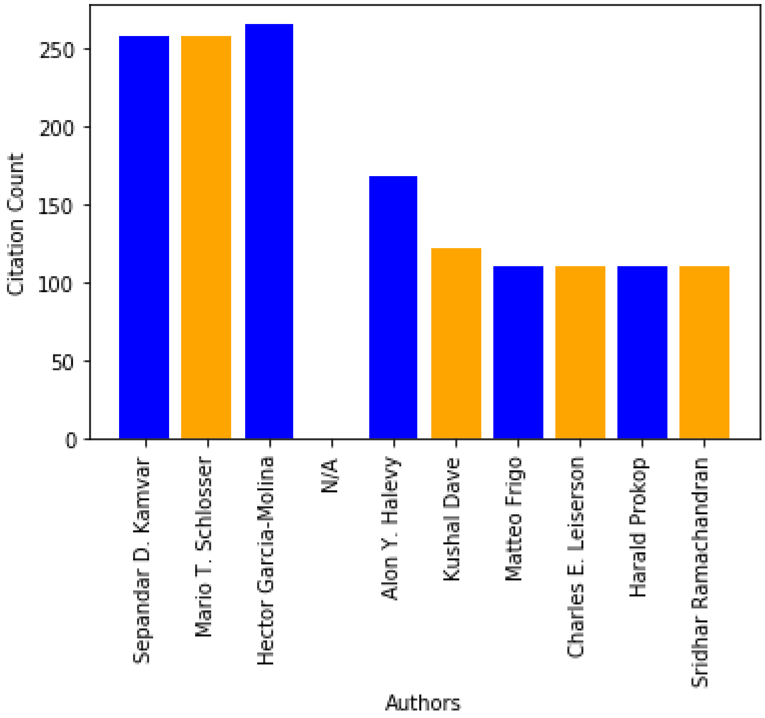
We had created another table called the domain table to find the popularity of the domain with time. The domain table consisted of 4 columns : Domain Name, Year, Papers Count and Domain Score. We ran group by query on our paper table to count number of papers published in each domain in a particular year. Domain Score for each domain in a particular year was calculated as papers published in that domain in a year divided by the total number of papers in that domain. For example, to calculate the domain score of "Art" domain in year 1980, we calculated the art domain papers in year 1980 and divided it by the total number of papers published in art domain. The domain table snapshot is attached below:



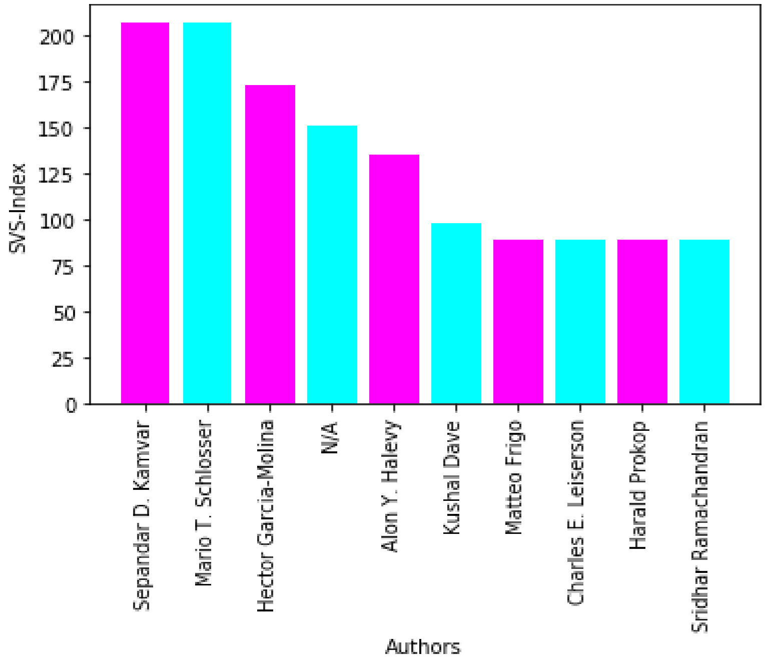
```
               domain  Year  Count  domain_Score
165  Computer Science  1963    1.0      0.000194
166  Computer Science  1972    5.0      0.000968
167  Computer Science  1973    1.0      0.000194
168  Computer Science  1974   12.0      0.002324
169  Computer Science  1975    1.0      0.000194
170  Computer Science  1976    1.0      0.000194
171  Computer Science  1977    1.0      0.000194
172  Computer Science  1978    3.0      0.000581
173  Computer Science  1979    3.0      0.000581
174  Computer Science  1980    5.0      0.000968
175  Computer Science  1981    6.0      0.001162
176  Computer Science  1982    2.0      0.000387
177  Computer Science  1983   12.0      0.002324
178  Computer Science  1984    2.0      0.000387
179  Computer Science  1985   11.0      0.002131
180  Computer Science  1986   24.0      0.004648
181  Computer Science  1987   20.0      0.003874
182  Computer Science  1988   51.0      0.009878
```

Graph Plots

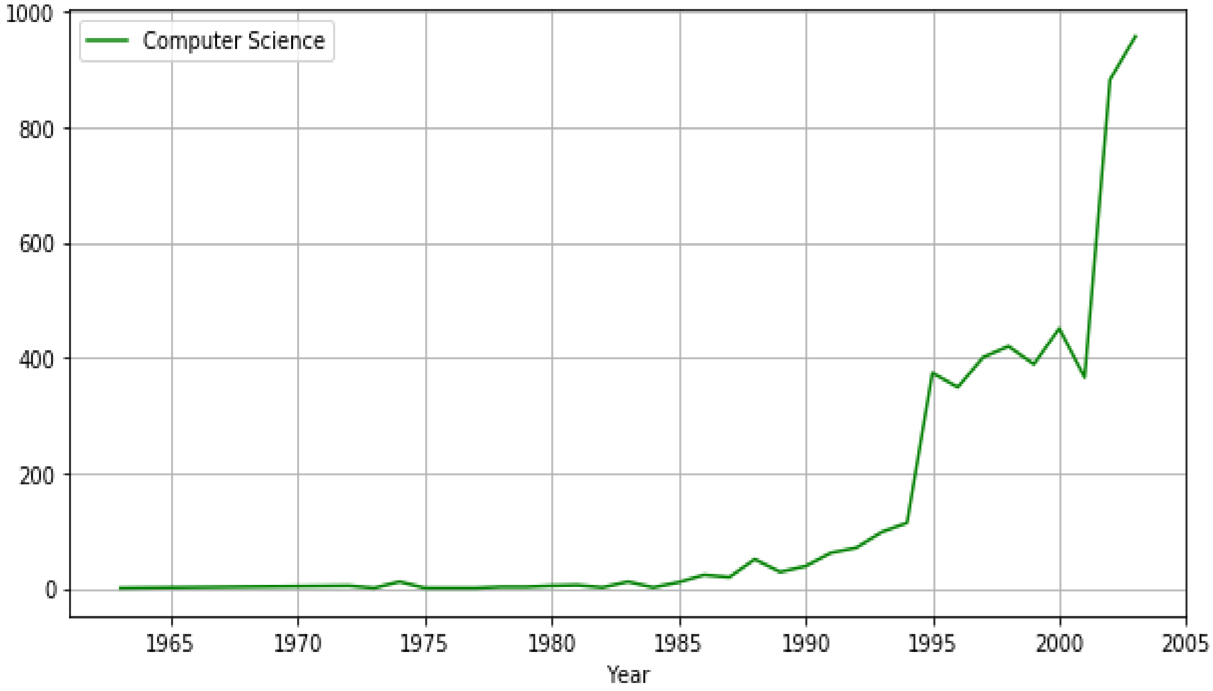
Plot 1: Plot of citation count of top 10 authors



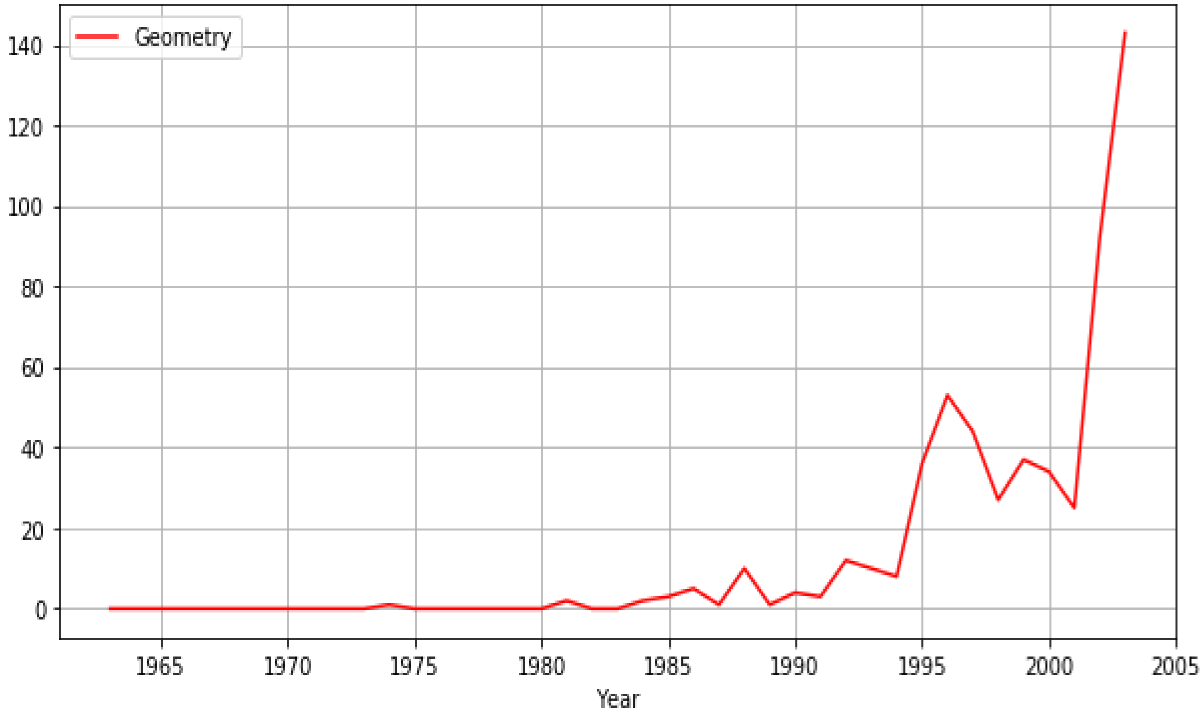
Plot 2: Plot of SVS-Index of top 10 authors



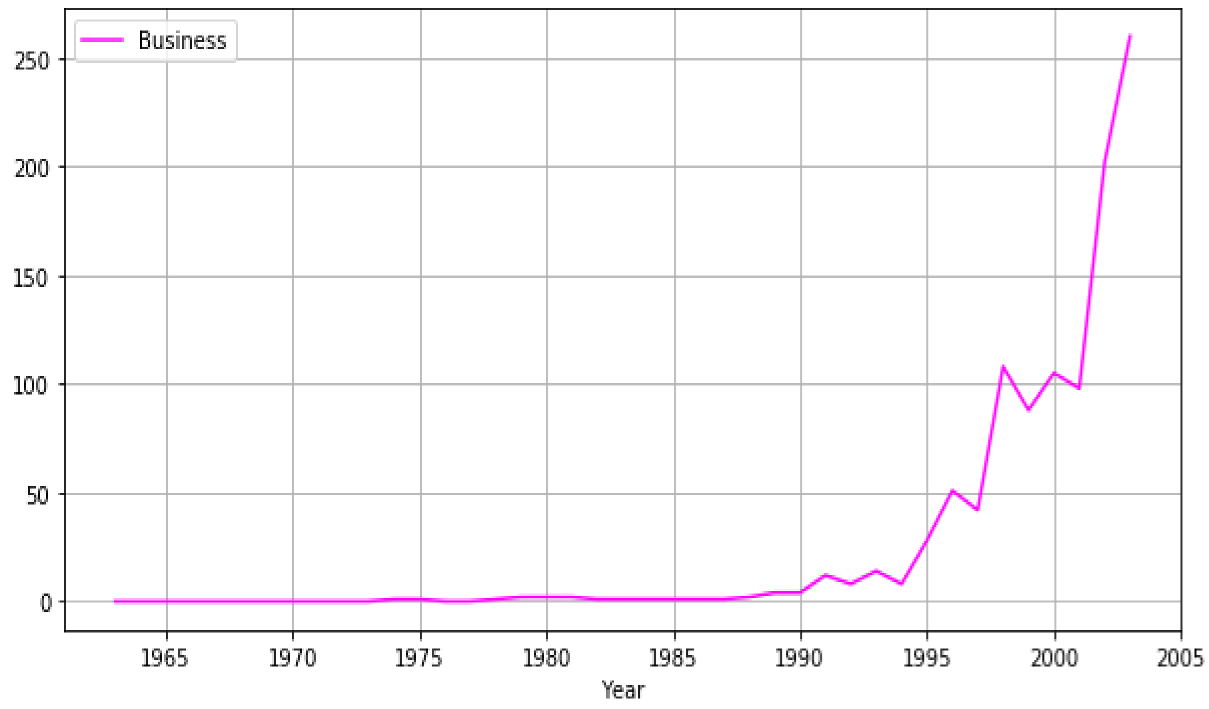
Plot 3: Plot of year wise count of Computer Science papers



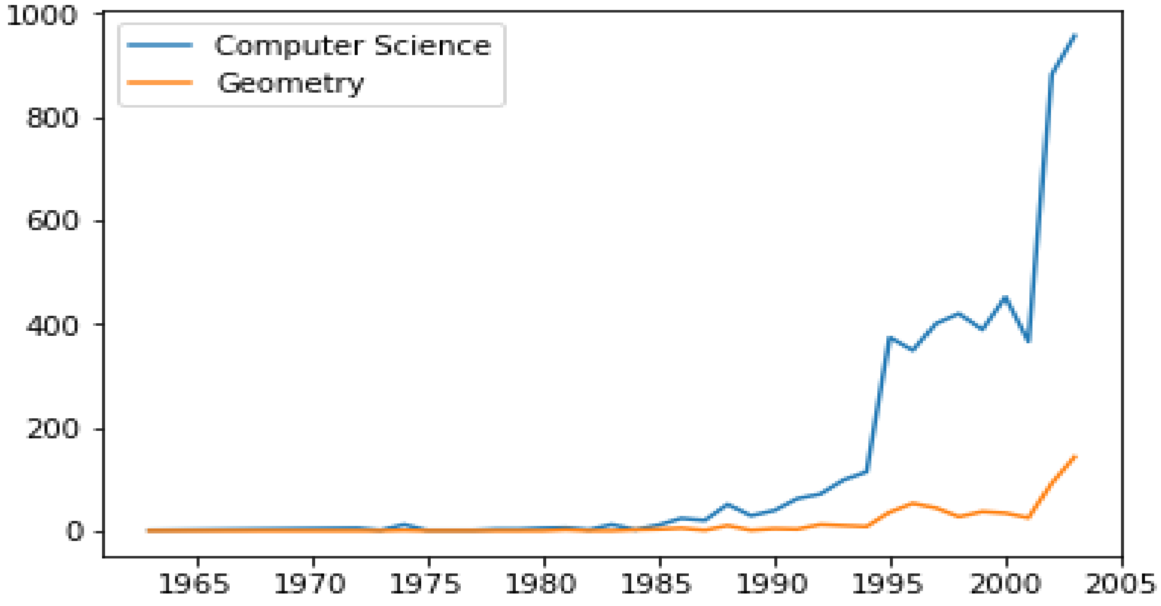
Plot 4: Plot of year wise count of Geometry papers



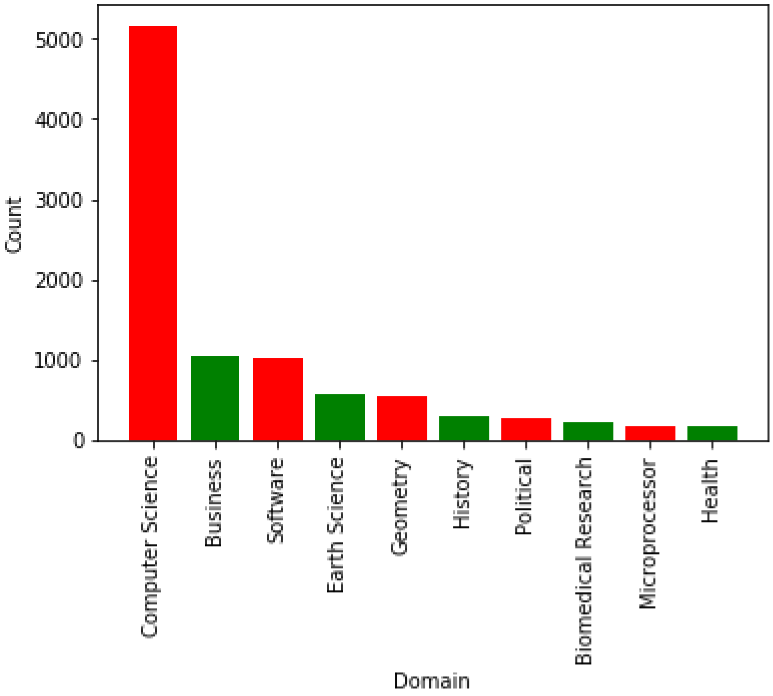
Plot 5: Plot of year wise count of Business papers



Plot 6: Comparison plot of year wise count of Computer Science and Geometry papers



Plot 7: Domain wise count comparison of papers



Baseline Model

We have used Random Forest Regressor as our baseline model. We have selected Random Forest over Linear Regression since there were a lot of categorical fields like abstract, paper title, authors name, domain which needs to be converted to numerical fields using label encoder. But, Linear Regression don’t give correct results in such cases.

Advanced Model

To get more accurate results we have used lightlgbm model. The set of features used in the model are abstract, paper title, domain, author, year, citation count, domain score. We have used root mean square error as an error metric and value came out to be 0.408.

Validation

We used various criterias to validate Author and Paper metric. Below are the methods used:

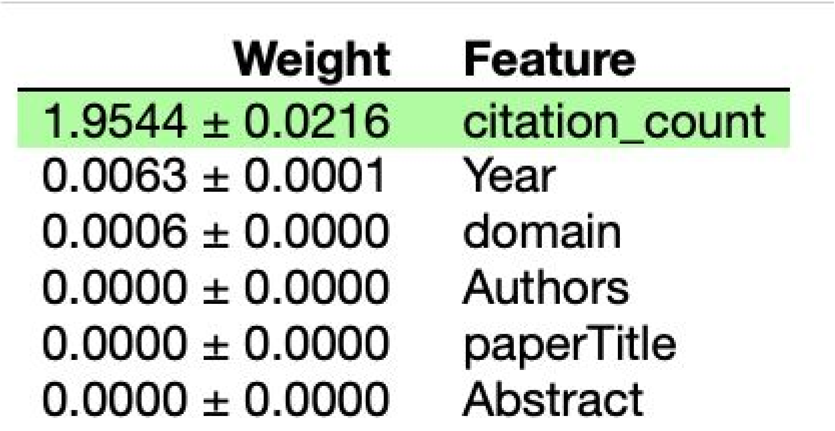
a. H-Index: Number of papers(h) with a citation number >= h.

b. G-Index: Given a set of articles ranked in decreasing order of the number of citations that they have received, the G-Index is the largest number such that the top g articles received at least g^2 citations.
c. I10 Index: Number of publications with at least 10 citations.
d. Google Scholar Metric: It allows authors to view journal rankings and ratings by various h-indexes.

For each of the above methods, we have obtained the ranking of the research paper. Once obtained, the ranking is compared with the one obtained from our model. Hence, this helped us in validating our model.

Features Importance analysis after applying Advanced Model

We tried on the dataset for 1000 rows and below are the features importance results which we obtained:



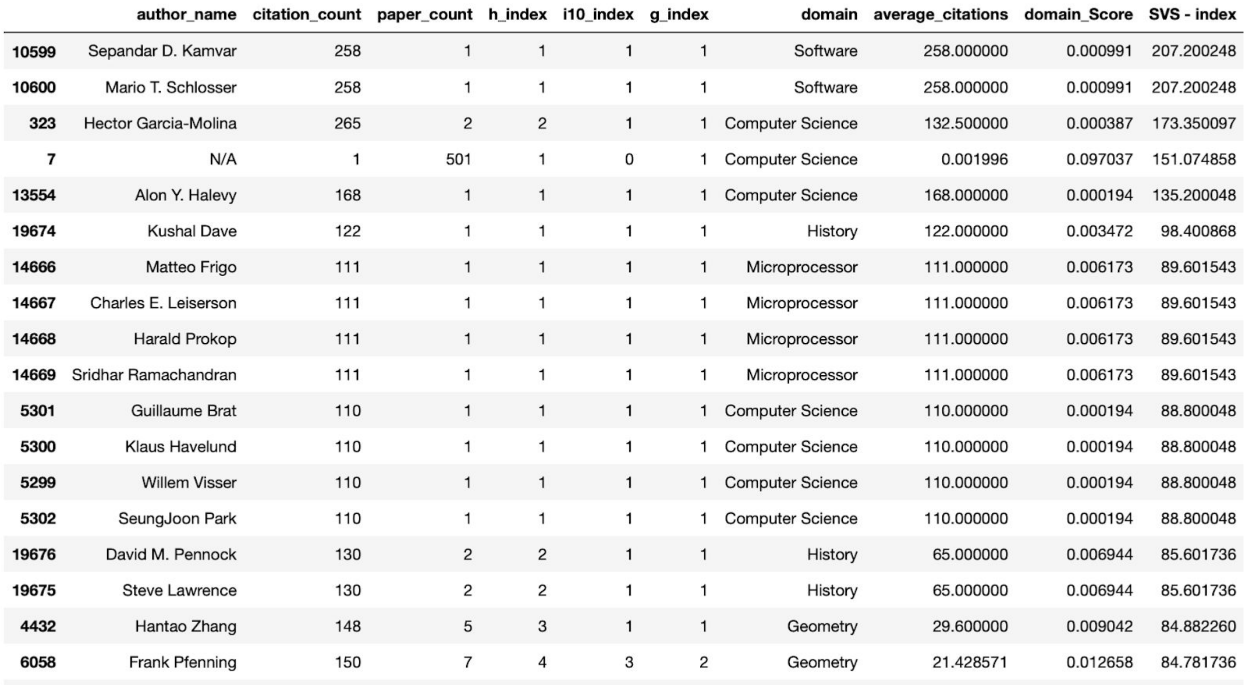
Weight	Feature
1.9544 ± 0.0216	citation_count
0.0063 ± 0.0001	Year
0.0006 ± 0.0000	domain
0.0000 ± 0.0000	Authors
0.0000 ± 0.0000	paperTitle
0.0000 ± 0.0000	Abstract

Questions

Question 1: You must identify top 100 ranked researchers from multiple disciplines based on your ranking metric and see how it stacks up against their respective h-index.

Solution:
We sorted our author table in the decreasing order based on SVS-Index. We ran the below queries to get the top 100 researchers from multiple disciplines based on SVS index:
df = df.sort_values(by=’SVS-index’, ascending=False),
df.head(100)

Below is the screenshot of the results obtained:



	author_name	citation_count	paper_count	h_index	i10_index	g_index	domain	average_citations	domain_Score	SVS - index
10599	Sepandar D. Kamvar	258	1	1	1	1	Software	258.000000	0.000991	207.200248
10600	Mario T. Schlosser	258	1	1	1	1	Software	258.000000	0.000991	207.200248
323	Hector Garcia-Molina	265	2	2	1	1	Computer Science	132.500000	0.000387	173.350097
7	N/A	1	501	1	0	1	Computer Science	0.001996	0.097037	151.074858
13554	Alon Y. Halevy	168	1	1	1	1	Computer Science	168.000000	0.000194	135.200048
19674	Kushal Dave	122	1	1	1	1	History	122.000000	0.003472	98.400868
14666	Matteo Frigo	111	1	1	1	1	Microprocessor	111.000000	0.006173	89.601543
14667	Charles E. Leiserson	111	1	1	1	1	Microprocessor	111.000000	0.006173	89.601543
14668	Harald Prokop	111	1	1	1	1	Microprocessor	111.000000	0.006173	89.601543
14669	Sridhar Ramachandran	111	1	1	1	1	Microprocessor	111.000000	0.006173	89.601543
5301	Guillaume Brat	110	1	1	1	1	Computer Science	110.000000	0.000194	88.800048
5300	Klaus Havelund	110	1	1	1	1	Computer Science	110.000000	0.000194	88.800048
5299	Willem Visser	110	1	1	1	1	Computer Science	110.000000	0.000194	88.800048
5302	SeungJoon Park	110	1	1	1	1	Computer Science	110.000000	0.000194	88.800048
19676	David M. Pennock	130	2	2	1	1	History	65.000000	0.006944	85.601736
19675	Steve Lawrence	130	2	2	1	1	History	65.000000	0.006944	85.601736
4432	Hantao Zhang	148	5	3	1	1	Geometry	29.600000	0.009042	84.882260
6058	Frank Pfenning	150	7	4	3	2	Geometry	21.428571	0.012658	84.781736

Question 2: Using a citation network graph, devise a “reach” function which can identify the degree of a paper’s influence, which can either be localised in a domain or have a more global inter-disciplinary effect.

Solution:
To precisely find out the reach of a paper, we added 1 more column of domain_score in our paper table. Domain score for each paper is calculated by looking at the domain score from the domain table by matching the paper’s domain and the paper publishing year. Now to calculate the reach score of each paper, we have used the formula :

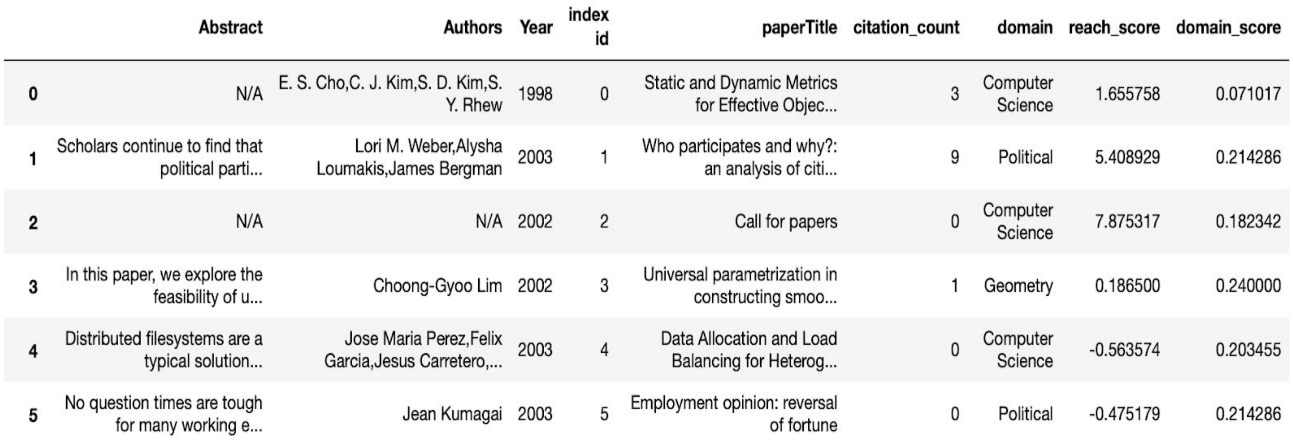
Old metric which we were using before:
reach_score = 0.3*curr_sum + 0.5*citations - 3*domain_score

Improvement
Currently, we have also included age of the paper as a parameter to evaluate the reach score of the paper. To include age factor into the score, we have added a new parameter i.e. 2*citations/age_of_paper into the reach score. To calculate age of the paper, we have subtracted publishing year of the paper from 2018.

New metric:
reach_score = 0.5*curr_sum + 0.8*citations - domain_score + 2 * citations/age_of_paper

where curr_sum is the author’s popularity(SVS-Index), citations is the number of citations of the paper and domain_score is the domain_score of the paper. In case we have multiple authors of a paper, we had taken the average of SVS-Index of the authors. Also, we had subtracted the domain score to consider the reach score of a paper since if the domain was popular then that means the influence of the paper was not solely due to its quality, but it was because due to its domain popularity.

Attached is the snapshot of the paper table after evaluated the reach-score of each paper :



	Abstract	Authors	Year	index id	paperTitle	citation_count	domain	reach_score	domain_score
0	N/A	E. S. Cho,C. J. Kim,S. D. Kim,S. Y. Rhew	1998	0	Static and Dynamic Metrics for Effective Objec...	3	Computer Science	1.655758	0.071017
1	Scholars continue to find that political parti...	Lori M. Weber,Alysha Loumakis,James Bergman	2003	1	Who participates and why?: an analysis of citi...	9	Political	5.408929	0.214286
2	N/A	N/A	2002	2	Call for papers	0	Computer Science	7.875317	0.182342
3	In this paper, we explore the feasibility of u...	Choong-Gyoo Lim	2002	3	Universal parametrization in constructing smoo...	1	Geometry	0.186500	0.240000
4	Distributed filesystems are a typical solution...	Jose Maria Perez,Felix Garcia,Jesus Carretero,...	2003	4	Data Allocation and Load Balancing for Heterog...	0	Computer Science	-0.563574	0.203455
5	No question times are tough for many working e...	Jean Kumagai	2003	5	Employment opinion: reversal of fortune	0	Political	-0.475179	0.214286

Question 3: Identify examples of researchers with substantial differences between your metric and h-index. Develop an evaluation to decide who is better in these cases, and use it to improve your metric.

Solution: We have identified some authors who have substantial differences between our metric and h-index. For example, authors Mike Schuster and Dongbin Xiu have the same index but we had given higher metric to Mike Schuster since he had more number of citations than Dongbin Xiu for the same number of papers count. Another example is I. Foster and Kang G. Shin. I. Foster and Kang G. Shin both have the same h-index but we had given more SVS-index to I. Foster since he had more number of citations count than Kang G. Shin.

Question 4: Do a time analysis of paper citation count against your metric. Is there any relevance to the topic of the paper against the time it was published. Maybe a paper got too popular due to its time of publishing coinciding with a recent technology interest which the said paper covered. Ensure your metric incorporates that to smooth out papers of high h-index which got higher citation counts because of factors other than solely the quality of material presented.

Solution: This was important to include the relevance to the topic of the paper against the time it was published. To do this, we have incorporated a domain table which keeps track of number of papers published in a particular domain in a given year. The formula which we have used to determine our metric is:

reach_score = 0.5*curr_sum + 0.8*citations - domain_score + 2 * citations/age_of_paper

where curr_sum is the author's popularity(SVS-Index), citations is the number of citations of the paper and domain_score is the domain_score of the paper. It was possible that the influence of a paper was not due to the quality of material presented in the paper but it can also be happen due to the popularity of the paper domain. So we have subtracted the domain influence from the paper metric to smooth out our metric.

Question 5: Can you identify papers on Arxiv which should become popular or important?

Solution: We tested our model on first 100 rows of arxivData.json downloaded from https://www.kaggle.com/neelshah18/arxivdataset/version/2. Our metric gave the below 5 papers as our top 5 rows according to our metric:

1. Describing Videos by Exploiting Temporal Structure
2. "Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models"
3. Adversarial Feature Learning
4. Building Machines That Learn and Think Like People
5. The Mythos of Model Interpretability

References

1. https://aminer.org/citation
2. https://nlp.stanford.edu/software/CRF-NER.shtml
3. https://scholar.google.com/intl/en/scholar/metrics.html
4. https://aminer.org/open-academic-graph
5. http://guides.library.cornell.edu/impact
6. https://en.wikipedia.org/wiki/Google_Scholar
7. https://matplotlib.org/users/pyplot_tutorial.html