Homework Appendix

Precision and Recall

Permutation	$P_{0.25}$	$P_{0.5}$	$P_{0.75}$	$P_{1.00}$	P_{mean1}	P_{mean2}	P_{norm}	R_{norm}
Default	0.4672	0.3240	0.1829	0.1099	0.3247	0.3150	0.7220	0.9301
Raw TF	0.3233	0.2088	0.0913	0.0475	0.2078	0.2277	0.6303	0.9041
Boolean TF	0.2171	0.1303	0.0686	0.0157	0.1387	0.1538	0.5231	0.8831
Dice similarity	0.4059	0.2977	0.1591	0.0817	0.2874	0.2813	0.6955	0.9282
Unstemmed	0.3931	0.1964	0.0777	0.0457	0.2224	0.1702	0.5962	0.8249
No stopwords	0.4755	0.3120	0.1755	0.0960	0.3210	0.3137	0.7127	0.9341
Equal weight	0.4882	0.3101	0.1865	0.0995	0.3283	0.3188	0.7238	0.9314
Weight 1114	0.3929	0.2705	0.1352	0.0695	0.2662	0.2711	0.6878	0.9264

Instructions

1. Run

```
python3 vector1.py
```

2. Choose option 2, you will get the default result.

Codes Modification for different permutation

1. **Raw TF**

   In *__init__*, comment the line *self.set_TFIDF_weight()*

2. **Boolean TF**

   In *init_doc_vectors* and *init_qry_vectors*, set *curr_doc_vector[word] = 1*

3. **Dice sim**

   In *set_retreived_set*, change *calc_cosine_sim_a* to *calc_dice_sim*

4. **Unstemmed tokens**

   In *__init__*, set *datafile = DataFile(DIR, HOME, FALSE)*

5. **No stop words**

   In *init_corp_freq*, comment the block about initializing the stop list

6. **Equal weight**

   In constants.py, set the all class vars in Weight class be 1.

7. **Weight 1114**

   In constants.py, set the all class vars in Weight class be 1, 1, 1, 4.

Retrieved documents for Queries 6, 9, 22

Instructions

1. Run the script.
2. Choose option 1 and then choose 1, input the query number and the number of matching documents (e.g 20)

For Query 6

Top 20 Retrieved Docs

Similarity Doc# Author Title
* 0.22833907 1543 Howard # Computer Formulation of th 0.19840111 438 Gorn # Mechanical Pragmatics: A T * 0.19098175 2078 Eastman # Representations for Space 0.15623915 1009 Weinberg # Solution of Combinatorial * 0.15415474 2828 Clark # Hierarchical Geometric Mod 0.15322432 2389 Eastman # Preliminary Report on a Sy 0.14836301 3035 Wetherbe # A Strategic Planning Metho 0.14406417 1186 Lynch # Recursive Solution of a Cl 0.13959341 242 Wilson # Notes on Geometric Weighte 0.13499323 2671 Stone # A Note on a Combinatorial 0.13047427 356 Ingerman # INTEREST (Algorithm 45) 0.12994097 2417 Ehrlich # Four Combinatorial Algorit 0.12863395 2187 Amarel # Computer Science: A Concep 0.12849939 740 Wright # INTEREST (Algorithm 45) 0.12702240 705 Blakely # Combinatorial Of M Things 0.12652130 704 Collins # Combinatorial of M Things 0.12626056 402 Ingerman # Dynamic Declarations 0.12553762 2753 Pfefferkorn # A Heuristic Problem Solvin 0.12540542 2230 Bracchi # A Language for Treating Ge 0.12483566 1256 King # Dynamic Variable Formattin 0.12133480 1247 Brown # An Operating Environment f

Top 10 Docs overlap terms

Term	Weight - Query 6	Weight - Query 1543	Docfreq
motion	6.074056	12.1325	13

Term	Weight - Query 6	Weight - Query 438	Docfreq
motion	6.074056	7.279534	13

Term	Weight - Query 6	Weight - Query 2078	Docfreq
robot	3.037028	12.1325	3
plan	3.037028	9.871068	35

Term	Weight - Query 6	Weight - Query 1009	Docfreq
combinatori	3.037028	8.234138	29

There are some other statistics that I did not put here. You can use the IR engine easily get these numbers.

Similar Documents to Doc 239, 1236, 2740

Instructions

1. Run the script.
2. Choose the option 1 and the choose 3, input the document number and the number of matching documents and you will get the result.

For Doc 239

Top 20 retrived Docs

Similarity Doc# Author Title 1.00000000 239 Verhoeff # Inefficiency of the Use of 0.56824468 1032 Belzer # Theoretical Considerations 0.21183787 2965 Hanani # An Optimal Evaluation of B 0.20792934 3168 Laird # Comment on "An Optimal Eva 0.19697361 2160 Wong # Canonical Structure in Att 0.18742635 3169 Gudes # Note on "An Optimal Evalua 0.15677082 3012 Lucas # The Use of an Interactive 0.15372891 3134 Motzkin # The Use of Normal Multipli 0.14877571 1207 Dodd # Remarks on Simulation of B 0.14872281 1329 Mano # Simulation of Boolean Func 0.14395829 1457 Salton # Data Manipulation and Prog 0.14043229 891 Whitley # Everyman's Information Ret 0.12851049 651 Grems # A Survey of Languages and 0.12492080 2278 Tan # On Foster's Information St 0.12107966 1699 Rubinoff # Experimental Evaluation of 0.11670860 635 Baker # A Note on Multiplying Bool 0.11602621 275 Sams # Dynamic Storage Allocation 0.11445816 634 Salton # Manipulation of Trees in I 0.11269534 2345 Ashenhurst # Curriculum Recommendations 0.11044405 2516 Salasin # Hierarchical Storage in In 0.10782485 2140 Mullin # Retrieval-Update Speed Tra

For Doc 1236

1.00000000 1236 Salton # The SMART Automatic Docume 0.34944316 634 Salton # Manipulation of Trees in I 0.34553934 1699 Rubinoff # Experimental Evaluation of 0.33416031 2711 Salton # A Vector Space Model for A 0.30452400 2307 Salton # Dynamic Document Processin 0.29303631 1457 Salton # Data Manipulation and Prog 0.27830236 2575 Van # The Best-Match Problem in 0.23533013 1681 Rubinoff # Easy English,a Language fo 0.23453286 2990 Yu 0.22254445 1032 Belzer # Theoretical Considerations 0.21233885 3012 Lucas # The Use of an Interactive 0.20300069 3134 Motzkin # The Use of Normal Multipli 0.17807016 2293 Jones # Comment on Average Binary 0.17490636 891 Whitley # Everyman's Information Ret 0.17457752 2140 Mullin # Retrieval-Update Speed Tra 0.16954592 1536 Lesk # Dynamic Computation of Der 0.16173103 1935 Arora # Randomized Binary Search T 0.16037117 1271 Davis # Secondary Key Retrieval Us 0.15557737 651 Grems # A Survey of Languages and 0.15518805 1680 Engvold # A General-Purpose Display 0.15388545 1194 Rubinoff # Establishment of the ACM R

For Doc 2740

1.00000000 2740 Lauesen # A Large Semaphore Based Op 0.31307158 1749 Dijkstra # The Structure of the "THE" 0.28751575 2379 Liskov # The Design of the Venus Op 0.27688655 2597 Hoare # Monitors: An Operating Sys 0.27047562 3043 Hansen # Distributed Processes: A C 0.26307507 2700 Lipton # Reduction: A Method of Pro 0.25283441 2618 Lamport # A New Solution of Dijkstra 0.24834129 2376 Habermann # Synchronization of Communi 0.24333602 2866 Howard # Proving Monitors 0.23233273 2500 Frailey # A Practical Approach to Ma 0.22766265 2228 Holt # Comments on Prevention of 0.22471783 2920 Devillers # Game Interpretation of the 0.22210889 3128 Reed # Synchronization with Event 0.22205545 2280 Parnas # Comment on Deadlock Preven 0.21372259 1611 Klein # Scheduling Project Network 0.21126016 2482 Howard # Mixed Solutions for the De 0.19615873 2080 Hansen # The Nucleus of a Multiprog 0.19343545 2320 Hansen # Structured Multiprogrammin 0.18902975 2777 Parnas # On a Solution to the Cigar 0.18136038 2738 Parnas # Use of the Concept of Tran 0.17771760 2378 Gaines # An Operating System Based

Extension - SVD, Dimensionality Reducing

In *SVDExtension.py*, I implement the SVD mechansim. The idea is to store all the weights for documents in a big matrix. We can decompose the matrix by *numpy*.

We calculate the sum of squares of all the singulars in the sigma matrix. We call this A. We start from the biggest singular and iteratively calculate the sum of squares of the singulars, when we find the cumulative sum is equal to/more than $0.9A$, we stop and record the index.

In our calculation, we reduce the dimensionality from 3204 to 1050 (for doc vectors), 33 to 19 (for qry vectors). We use the new matrix to calcute the cosine similarity for the Query 6. Following is the result.

0.24667780 859 French # Computer Planned Collates 0.19341146 705 Blakely # Combinatorial Of M Things 0.18372842 2389 Eastman # Preliminary Report on a Sy 0.16794188 402 Ingerman # Dynamic Declarations * 0.15822489 1543 Howard # Computer Formulation of th 0.15726717 2671 Stone # A Note on a Combinatorial 0.15722312 279 Sams # The Case for Dynamic stora 0.15516841 3035 Wetherbe # A Strategic Planning Metho 0.15340266 1009 Weinberg # Solution of

Combinatorial * 0.14863034 2828 Clark # Hierarchical Geometric Mod * 0.14705843 2078 Eastman # Representations for Space 0.14124590 530 Siler # A Computer Method for Radi 0.13621427 1186 Lynch # Recursive Solution of a Cl 0.13612721 276 Holt # Program Organization and R 0.13474524 275 Sams # Dynamic Storage Allocation 0.13363468 1247 Brown # An Operating Environment f 0.13335833 888 Wolfson # Algorithm 160 Combinatoria 0.13286647 2187 Amarel # Computer Science: A Concep 0.13233898 704 Collins # Combinatorial of M Things 0.13010605 1517 Naylor # Methods for Analyzing Data 0.12681724 2753 Pfefferkorn # A Heuristic Problem Solvin

If you want run the SVDExtension, you should uncomment the block about init the SVD in the __*init*__ method in vector1.py and set the similarity calculation as SVDExtension calculation in set_retrieved_set.