Information Retrieval Systems
Part 1 - Retrieval Evaluation

Contents

Task 1: Process of creating an Inverted Index

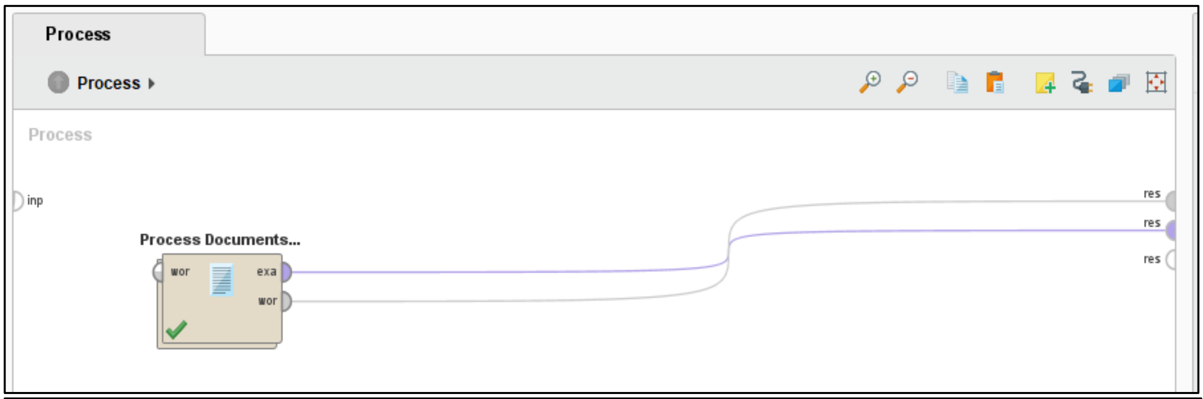
Inverted index is a method for efficiently storing information. It is an important data structure on which modern IR systems are built (Massie 2021).
Advantages:

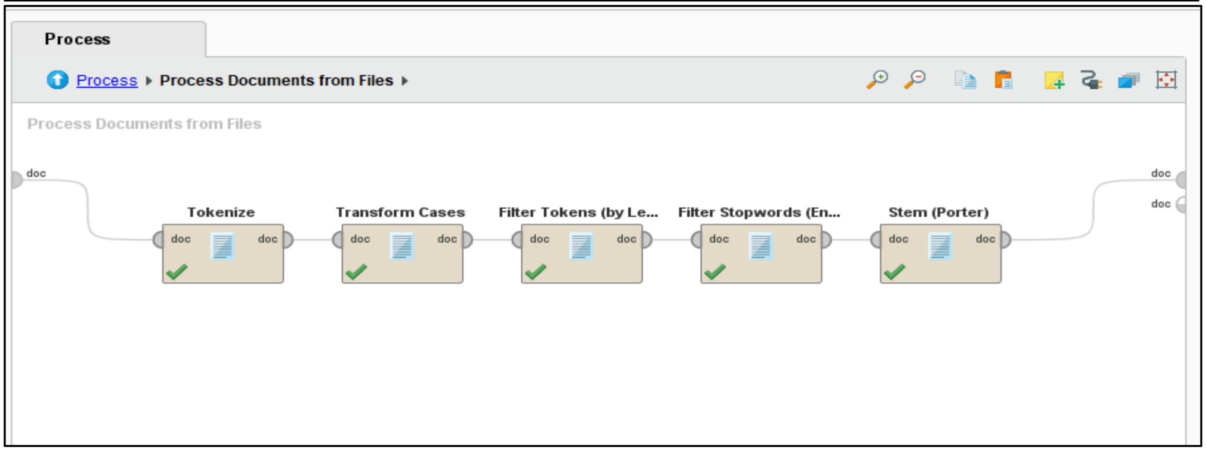
- It allows fast and full text searches
- Optimizing the speed of a query, allowing the search processes to perform faster and better.
- It is easy to develop
- This is considered as the most commonly used and popularly used data structure in document retrieval systems.
- Used on a large scale, for example search engines (Kopanev, 2019).

The key tasks to create an inverted index:

1. Collect the documents to be indexed
2. Tokenizing
3. Normalization
4. Stop word removal
5. Stemming
6. Dictionary and postings

Please find below steps from Rapid Miner process





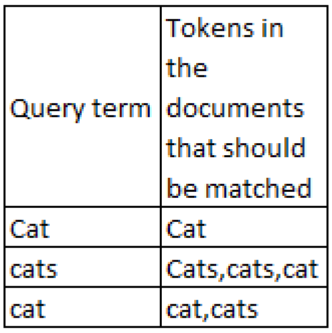
Process explained using IRD documents given in the assignment specification.

1. **Collect the documents:** This step involves bringing together all the documents that required in the process of creating an inverted index.

2. **Tokenizing:** This is the process of breaking down the document in small size tokens (words) and the same time eliminating any punctuations.
   For example, in the IRD documents doc1, doc, doc3, doc4
   Doc 1 The cat sat on the mat next to the cat.
   Doc 2 The black dogs sat on the black mat.
   Doc 3 The dog was on the mat.
   Doc 4 the black cat sat with the dogs.

   Here "The" is given document id (DocID) 1 because it appears in document 1. Similarly, all tokens/terms in document 1 are assigned DocID 1, all terms in document 2 are assigned DocID 2, all terms in document 3 are assigned DocID 3 and all terms in document 4 are assigned DocID 4.
   The only punctuation/s in these documents is a full stop, so that will be eliminated in creating tokens which are as below and each token is assigned a document id which is the document number in which it appears.

3. **Normalisation**: This step involves matching of the tokens of the query to the tokens in the token list of the documents.
   Here for example,



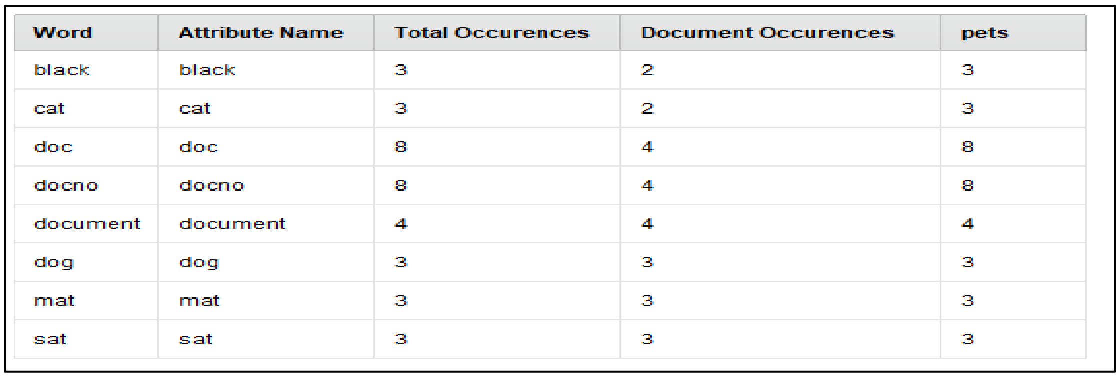
Query term	Tokens in the documents that should be matched
Cat	Cat
cats	Cats,cats,cat
cat	cat,cats

   The other form of normalization is case folding, i.e. changing all the letters to lower case. This will allow all instances of "Cat" at the beginning of a sentence to match with a query of "cat".

4. **Stop word removal:** There are some very common words that have little or no value in matching the documents with query. These words are called stop words. They may be eliminated from the document token list. Here in our set of documents these are "on", "the","to","was","with".

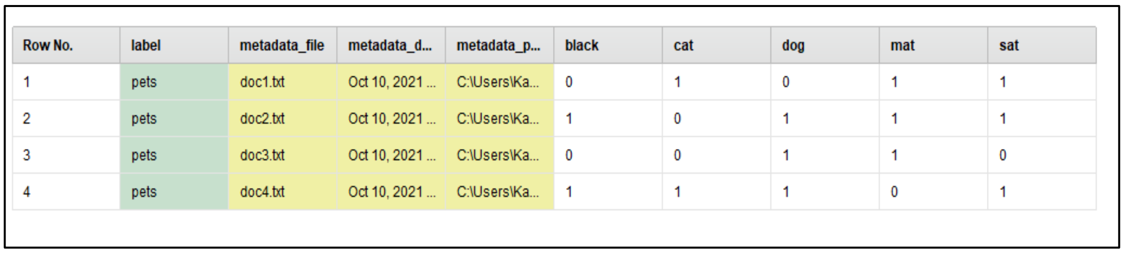
5. **Stemming:** "Stemming is the process of transforming a word into root form by cutting the ending of the word". Sometimes, different forms a single word can be used in documents. This technique involves different forms of a root to match. For example, in our set of documents "dog", "dogs" are different forms of the same word. With stemming the word "dogs" is matched to "dog" and the leaf 's' is taken out to form a common base "dog".

6. **Dictionary, documents and posting list:**
   The result of all the terms that follow after the above steps are successfully completed forms the dictionary(see figure below) of the inverted index or it can also be called the vocabulary where the term look up takes place on top of the posting list. The posting list is where the indexed data is stored. Each term is assigned its own posting. It is used to identify a particular document in which the term occurs (Bhat 2020).



Word	Attribute Name	Total Occurences	Document Occurences	pets
black	black	3	2	3
cat	cat	3	2	3
doc	doc	8	4	8
docno	docno	8	4	8
document	document	4	4	4
dog	dog	3	3	3
mat	mat	3	3	3
sat	sat	3	3	3

The column, "attribute name" is nothing but the term that remains after pre-processing and is used to represent the document collection. The number of standard attributes is the number of unique terms and forms the dictionary. The "total occurrences" column is the number of times the attribute appears in the collection. The column "document occurrences" is nothing but the number of documents the attribute appears in from the collection. For example, we can say from the above table that the term "black" appears three times (total occurrences) in two documents (document occurrences). Similarly, the term "mat" appears three times in three documents.
Below is the term document representation(example set as we call it in rapid miner),we can get the information about which documents the terms appear in with binary term occurrences, where '1' represents presence of the term in the document and '0' represents absence.

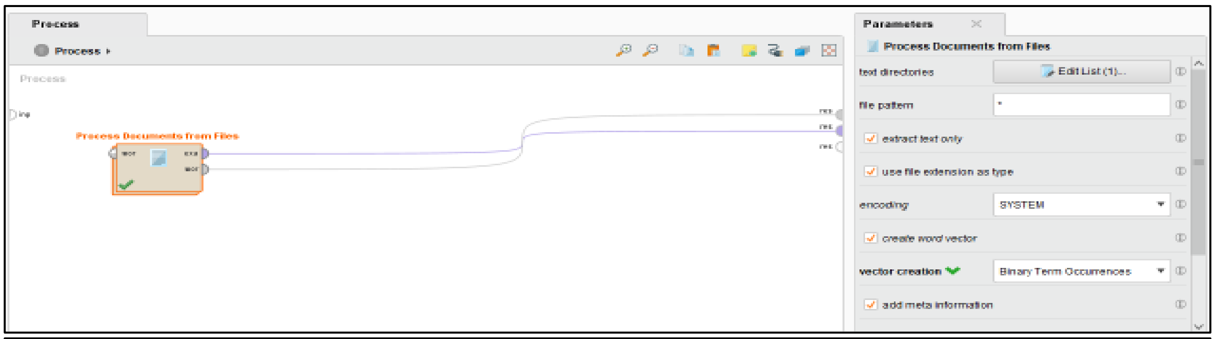


Row No.	label	metadata_file	metadata_d...	metadata_p...	black	cat	dog	mat	sat
1	pets	doc1.txt	Oct 10, 2021 ...	C:\Users\Ka...	0	1	0	1	1
2	pets	doc2.txt	Oct 10, 2021 ...	C:\Users\Ka...	1	0	1	1	1
3	pets	doc3.txt	Oct 10, 2021 ...	C:\Users\Ka...	0	0	1	1	0
4	pets	doc4.txt	Oct 10, 2021 ...	C:\Users\Ka...	1	1	1	0	1

For example, the term “black” has a ‘1’ against row 2 and row 4. This means that the term “black” appears in document 2 and document 4 as we can see the corresponding metadata_file column has the document numbers respectively .‘0’ in row 1 and 3 for the term “black” means that it does absent from document 1 and document 3. Similarly, “cat” appears in document 1 and document 4 (Massie 2021).

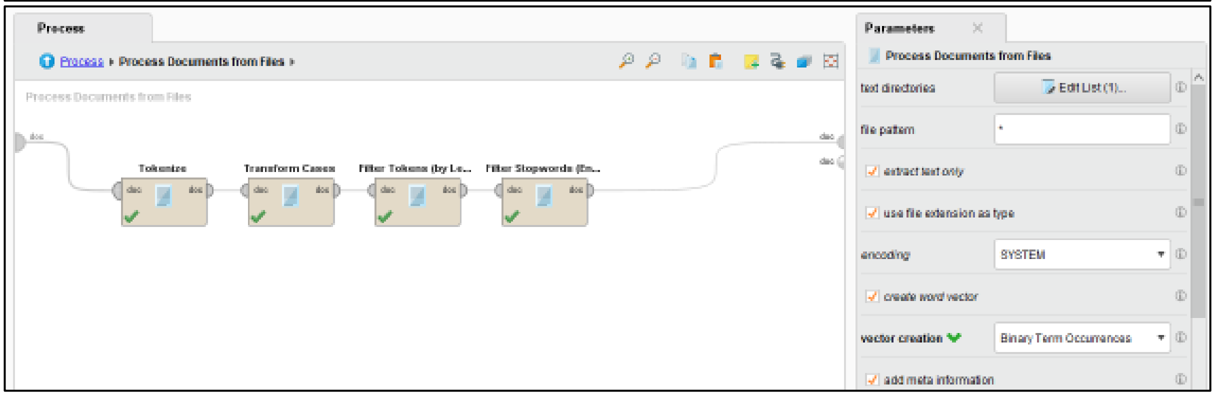
Task 2A: The inverted index with and without Stemming

Rapid Miner process without Stemming



Parameters
Process Documents from Files

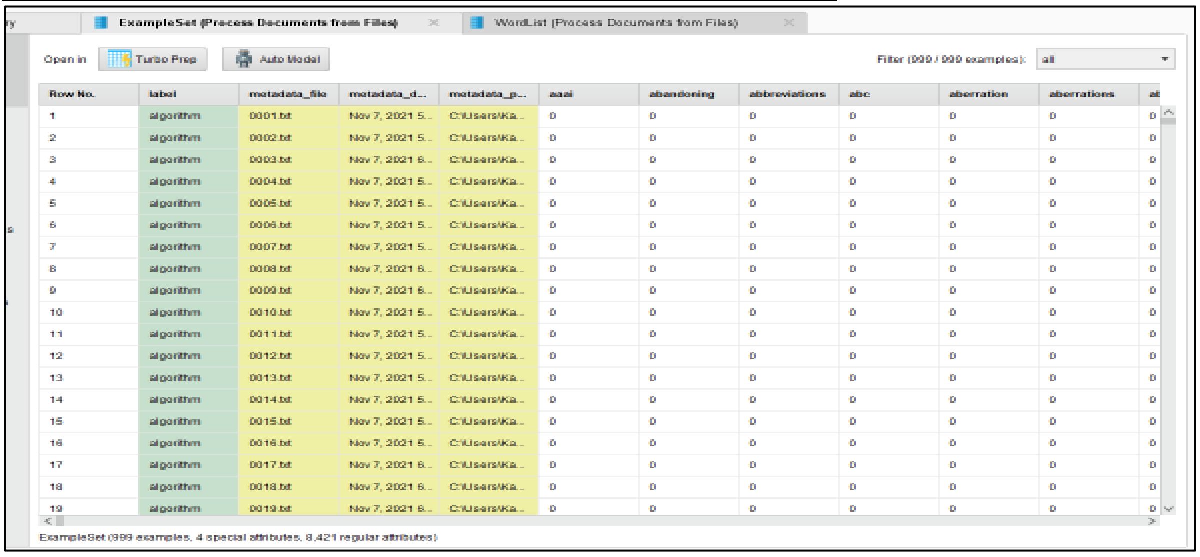
text directories	Edit List (1)...
file pattern	*
✓ extract text only	
✓ use file extension as type	
encoding	SYSTEM
✓ create word vector	
vector creation	Binary Term Occurrences
✓ add meta information	



Parameters
Process Documents from Files

text directories	Edit List (1)...
file pattern	*
✓ extract text only	
✓ use file extension as type	
encoding	SYSTEM
✓ create word vector	
vector creation	Binary Term Occurrences
✓ add meta information	

Example set (term document representation) without stemming



ExampleSet (Process Documents from Files) | WordList (Process Documents from Files)

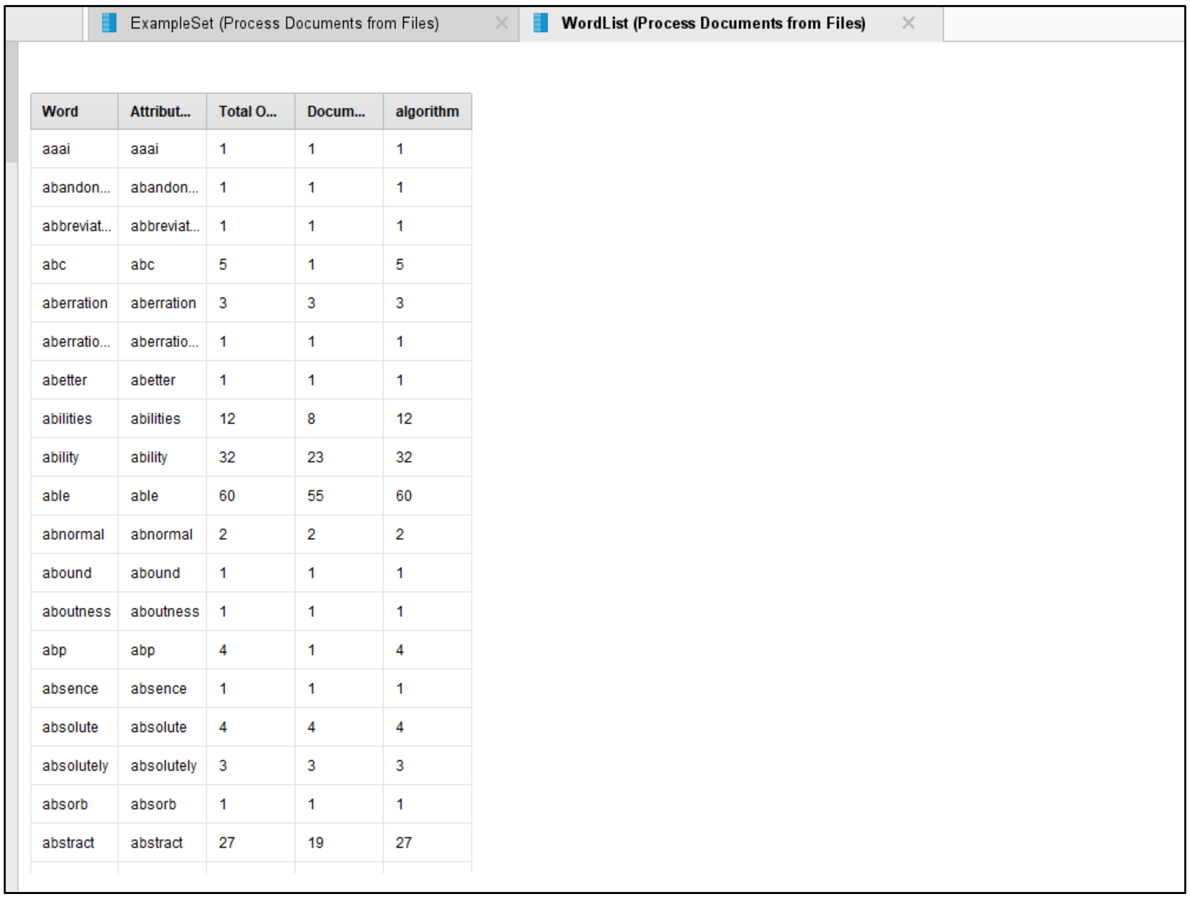
Open in: Turbo Prep | Auto Model

Filter (999 / 999 examples): all

Row No.	label	metadata_file	metadata_d...	metadata_p...	aaai	abandoning	abbreviations	abc	aberration	aberrations	ab
1	algorithm	0001.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
2	algorithm	0002.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
3	algorithm	0003.txt	Nov 7, 2021 6...	C:\Users\Ka...	0	0	0	0	0	0	0
4	algorithm	0004.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
5	algorithm	0005.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
6	algorithm	0006.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
7	algorithm	0007.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
8	algorithm	0008.txt	Nov 7, 2021 6...	C:\Users\Ka...	0	0	0	0	0	0	0
9	algorithm	0009.txt	Nov 7, 2021 6...	C:\Users\Ka...	0	0	0	0	0	0	0
10	algorithm	0010.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
11	algorithm	0011.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
12	algorithm	0012.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
13	algorithm	0013.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
14	algorithm	0014.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
15	algorithm	0015.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
16	algorithm	0016.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
17	algorithm	0017.txt	Nov 7, 2021 6...	C:\Users\Ka...	0	0	0	0	0	0	0
18	algorithm	0018.txt	Nov 7, 2021 6...	C:\Users\Ka...	0	0	0	0	0	0	0
19	algorithm	0019.txt	Nov 7, 2021 6...	C:\Users\Ka...	0	0	0	0	0	0	0

ExampleSet (999 examples, 4 special attributes, 8,421 regular attributes)

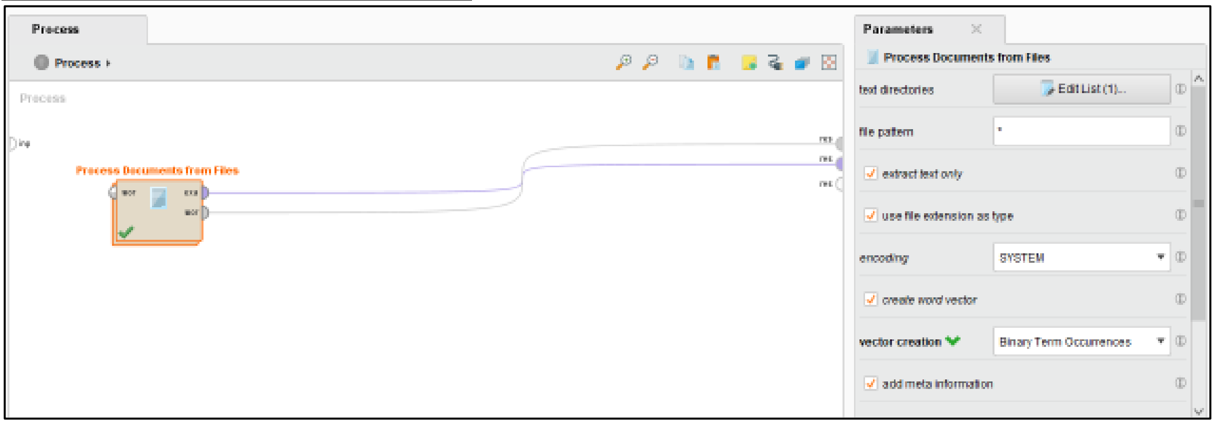
Wordlist (dictionary) in Rapid Miner



ExampleSet (Process Documents from Files) | WordList (Process Documents from Files)

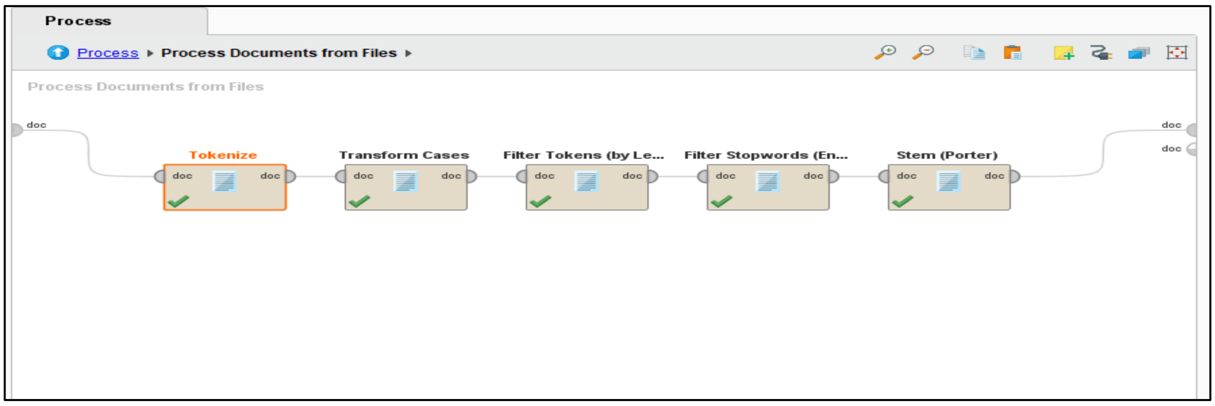
Word	Attribut...	Total O...	Docum...	algorithm
aaai	aaai	1	1	1
abandon...	abandon...	1	1	1
abbreviat...	abbreviat...	1	1	1
abc	abc	5	1	5
aberration	aberration	3	3	3
aberratio...	aberratio...	1	1	1
abetter	abetter	1	1	1
abilities	abilities	12	8	12
ability	ability	32	23	32
able	able	60	55	60
abnormal	abnormal	2	2	2
abound	abound	1	1	1
aboutness	aboutness	1	1	1
abp	abp	4	1	4
absence	absence	1	1	1
absolute	absolute	4	4	4
absolutely	absolutely	3	3	3
absorb	absorb	1	1	1
abstract	abstract	27	19	27

Rapid Miner process with Stemming

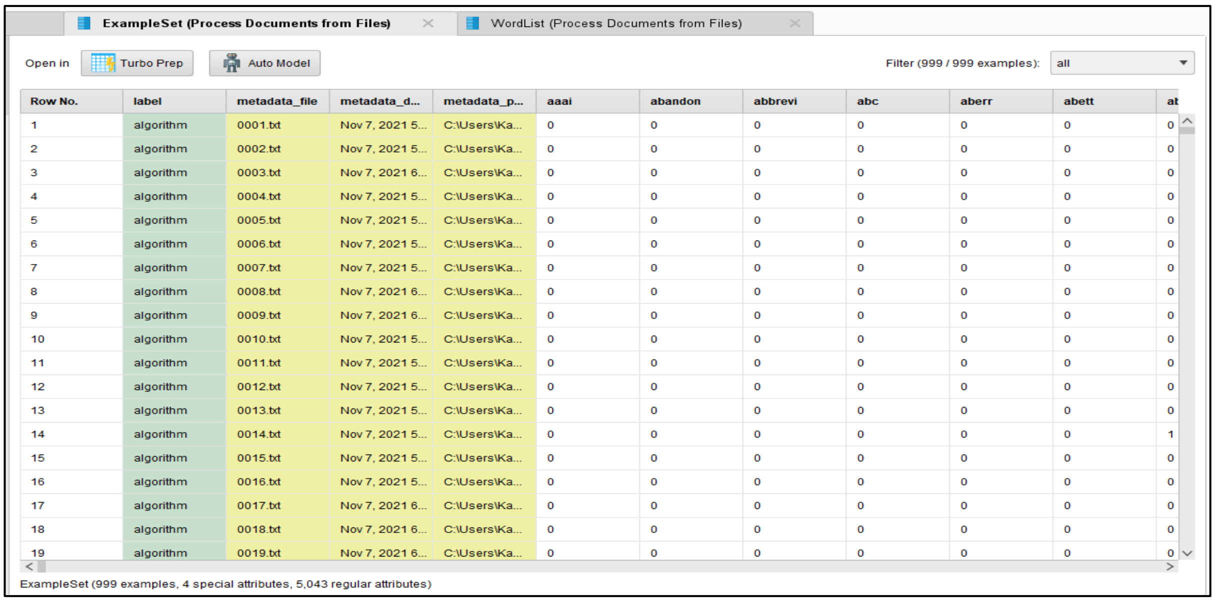


Process Documents from Files

text directories: Edit List (1)...
file pattern: *
✓ extract text only
✓ use file extension as type
encoding: SYSTEM
✓ create word vector
vector creation: Binary Term Occurrences
✓ add meta information



Example set (term frequency document) with stemming

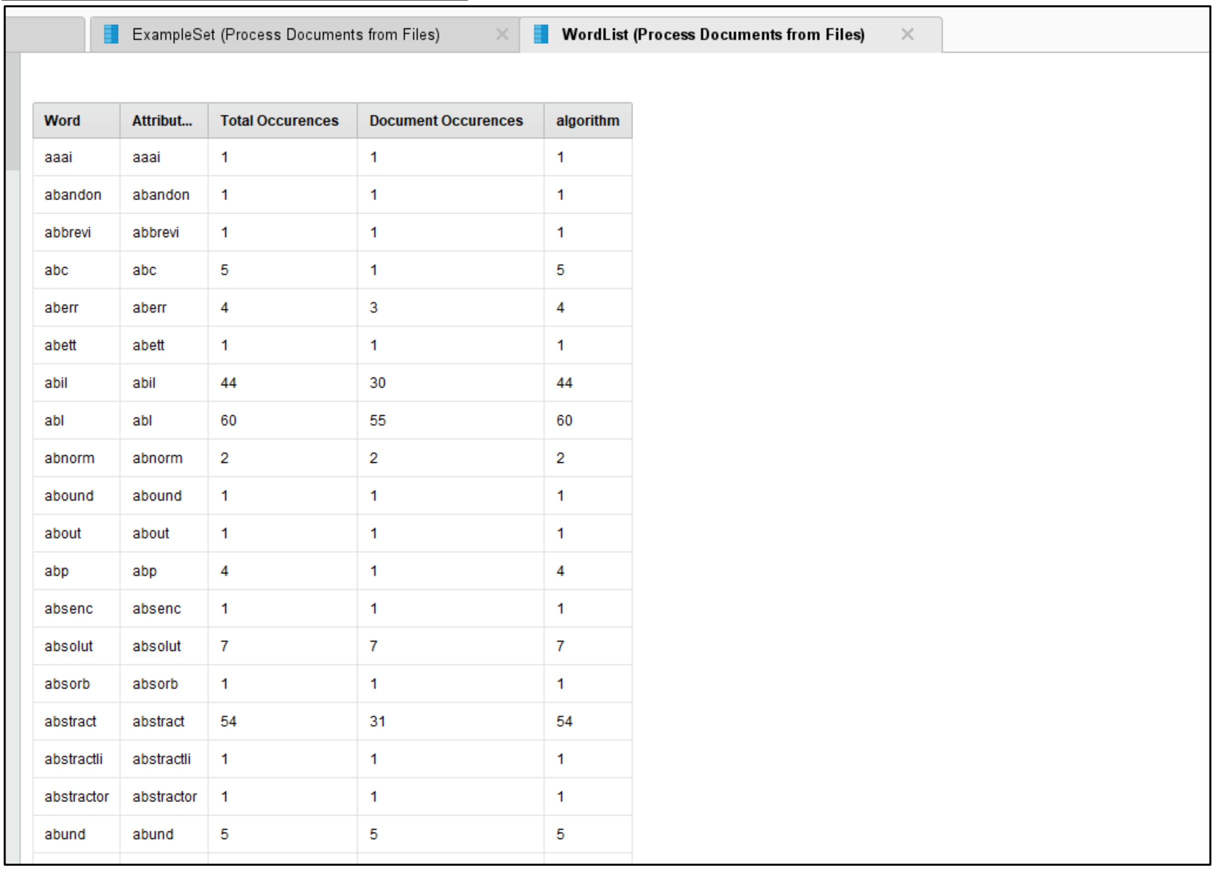

ExampleSet (Process Documents from Files)
WordList (Process Documents from Files)
Open in
Turbo Prep
Auto Model

Filter (999 / 999 examples): all

Row No.	label	metadata_file	metadata_d...	metadata_p...	aaai	abandon	abbrevi	abc	aberr	abett	ab
1	algorithm	0001.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
2	algorithm	0002.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
3	algorithm	0003.txt	Nov 7, 2021 6...	C:\Users\Ka...	0	0	0	0	0	0	0
4	algorithm	0004.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
5	algorithm	0005.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
6	algorithm	0006.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
7	algorithm	0007.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
8	algorithm	0008.txt	Nov 7, 2021 6...	C:\Users\Ka...	0	0	0	0	0	0	0
9	algorithm	0009.txt	Nov 7, 2021 6...	C:\Users\Ka...	0	0	0	0	0	0	0
10	algorithm	0010.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
11	algorithm	0011.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
12	algorithm	0012.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
13	algorithm	0013.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
14	algorithm	0014.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	1
15	algorithm	0015.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
16	algorithm	0016.txt	Nov 7, 2021 5...	C:\Users\Ka...	0	0	0	0	0	0	0
17	algorithm	0017.txt	Nov 7, 2021 6...	C:\Users\Ka...	0	0	0	0	0	0	0
18	algorithm	0018.txt	Nov 7, 2021 6...	C:\Users\Ka...	0	0	0	0	0	0	0
19	algorithm	0019.txt	Nov 7, 2021 6...	C:\Users\Ka...	0	0	0	0	0	0	0

ExampleSet (999 examples, 4 special attributes, 5,043 regular attributes)

Wordlist (dictionary) in Rapid Miner



ExampleSet (Process Documents from Files) | WordList (Process Documents from Files)

Word	Attribut...	Total Occurences	Document Occurences	algorithm
aaai	aaai	1	1	1
abandon	abandon	1	1	1
abbrevi	abbrevi	1	1	1
abc	abc	5	1	5
aberr	aberr	4	3	4
abett	abett	1	1	1
abil	abil	44	30	44
abl	abl	60	55	60
abnorm	abnorm	2	2	2
abound	abound	1	1	1
about	about	1	1	1
abp	abp	4	1	4
absenc	absenc	1	1	1
absolut	absolut	7	7	7
absorb	absorb	1	1	1
abstract	abstract	54	31	54
abstractli	abstractli	1	1	1
abstractor	abstractor	1	1	1
abund	abund	5	5	5

In this task we compare the inverted index with and without stemming using some measures as shown in the table below.

Measure	Without Stemming	With Stemming
Number of documents	999	999
Total number of index terms extracted from documents	92866	92866
Number of posting lists	8421	5043
Total number of postings	65142	59601
Average number of postings per posting list	65142/8421 =7.735	59601/5403 = 11.818
Retrieval time	19	17
Storage space	More	Less

- The number of documents and the total number of index terms are same i.e. 999, 8848 respectively in the process with and without stemming.
- Number of posting lists without stemming is more compared to the process with stemming, because stemming matches the terms to a root and reduces the multiple forms of that term appearing in the inverted index, hence reduced number of posting lists.
- Total number of postings will also be more in the process without stemming compared to process with stemming as a result of multiple forms of the same term being extracted.

- The storage space in a process without stemming is comparatively more because the inverted index has more terms, i.e. similar forms of terms are also extracted.
- The retrieval efficiency is better in the process with stemming compared to that without stemming. The retrieval is more accurate because the relevance to documents is more precise.
- As we can see from the table above that the size of dictionary is bigger without stemming when compared to with stemming, this also impacts the retrieval time because the process has to run through a bigger dictionary size(no stemming)to find relevance it takes more time.

<u>With Stemming</u>

Storage space – when considering the storage capacity on a disk, the process occupies less space. Here, the stemming process already eliminates duplicates or repeated words. So greatly reduces the required disk space.

Retrieval efficiency – In case of the retrieval efficiency, indexing with stemming process leads to the

- Retrieval process in an efficient manner
- Searching the required term from index did not needed any additional resources
- Compression over the index file is possible in an efficient way

Since the indexed words or terms are processed with the stemming, the searching processes do not expect any additional support.

<u>Without stemming</u>

Storage space – when considering the storage capacity on a disk, the process occupies more disk space for the following reasons such as

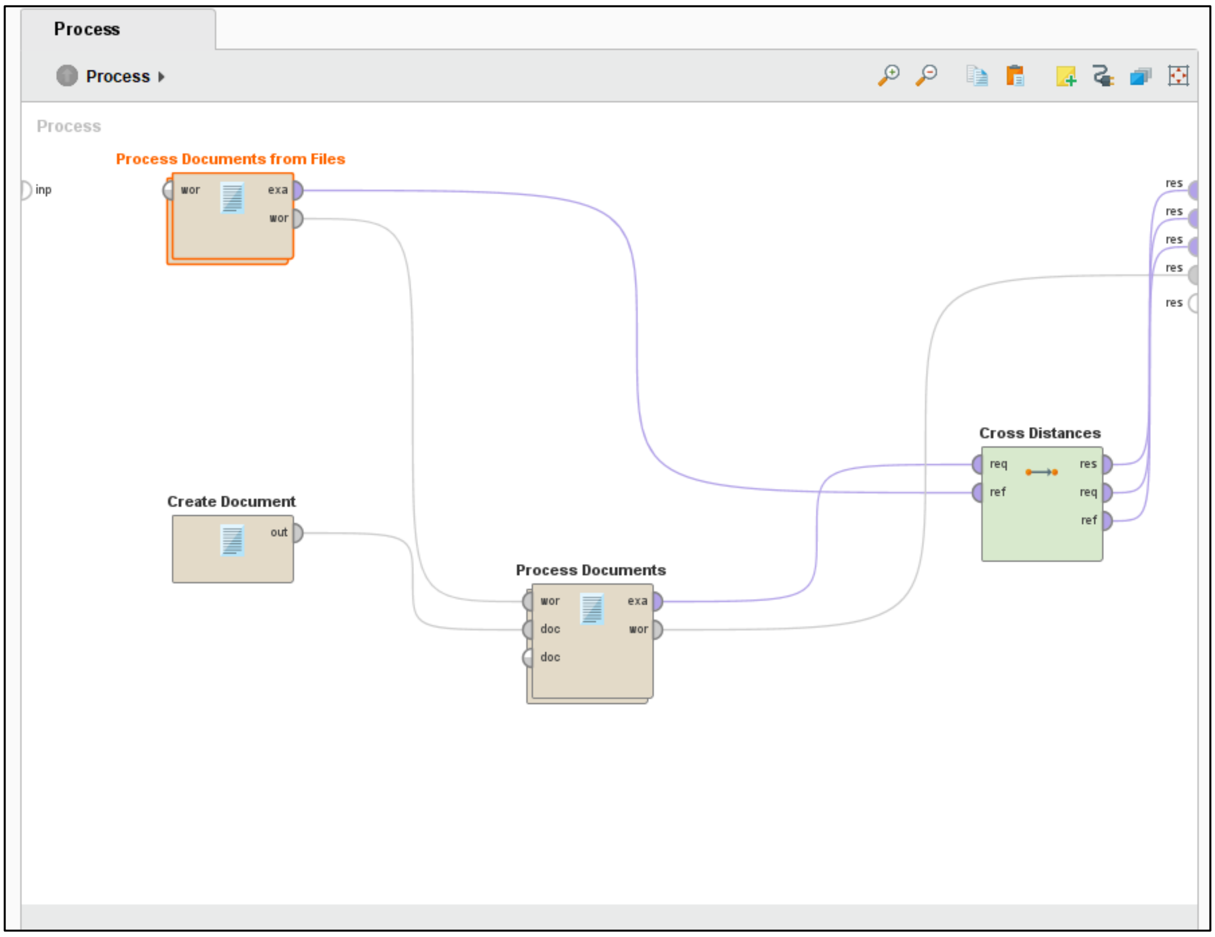
- Requires additional temporary disk space while performing operations
- Requires permanent memory space, higher than the stemmed inverted index over the disk in order to save the indexing processes.

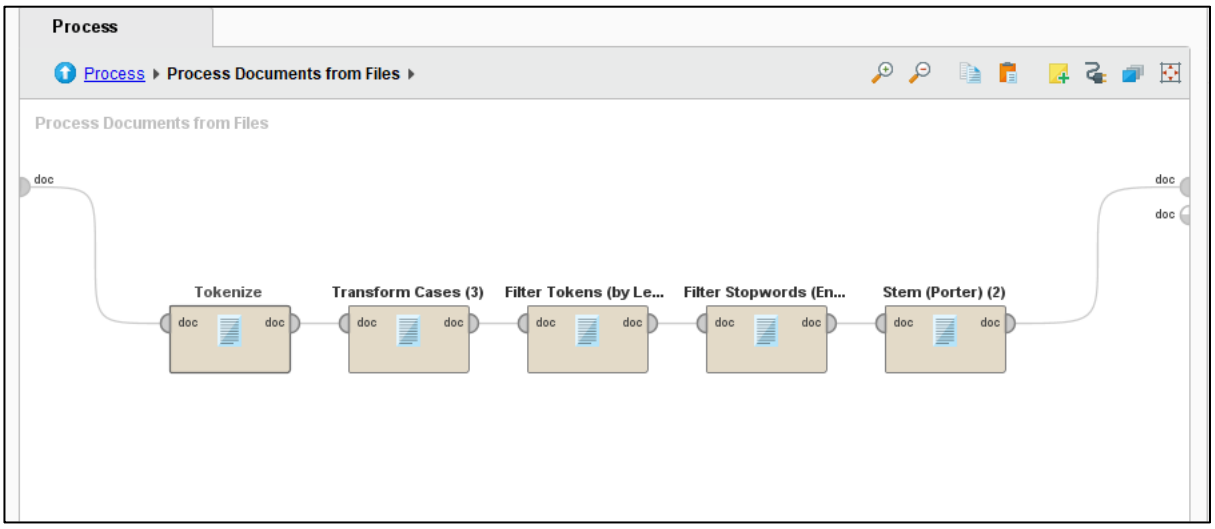
Retrieval efficiency – In case of the retrieval efficiency, indexing without any stemming means

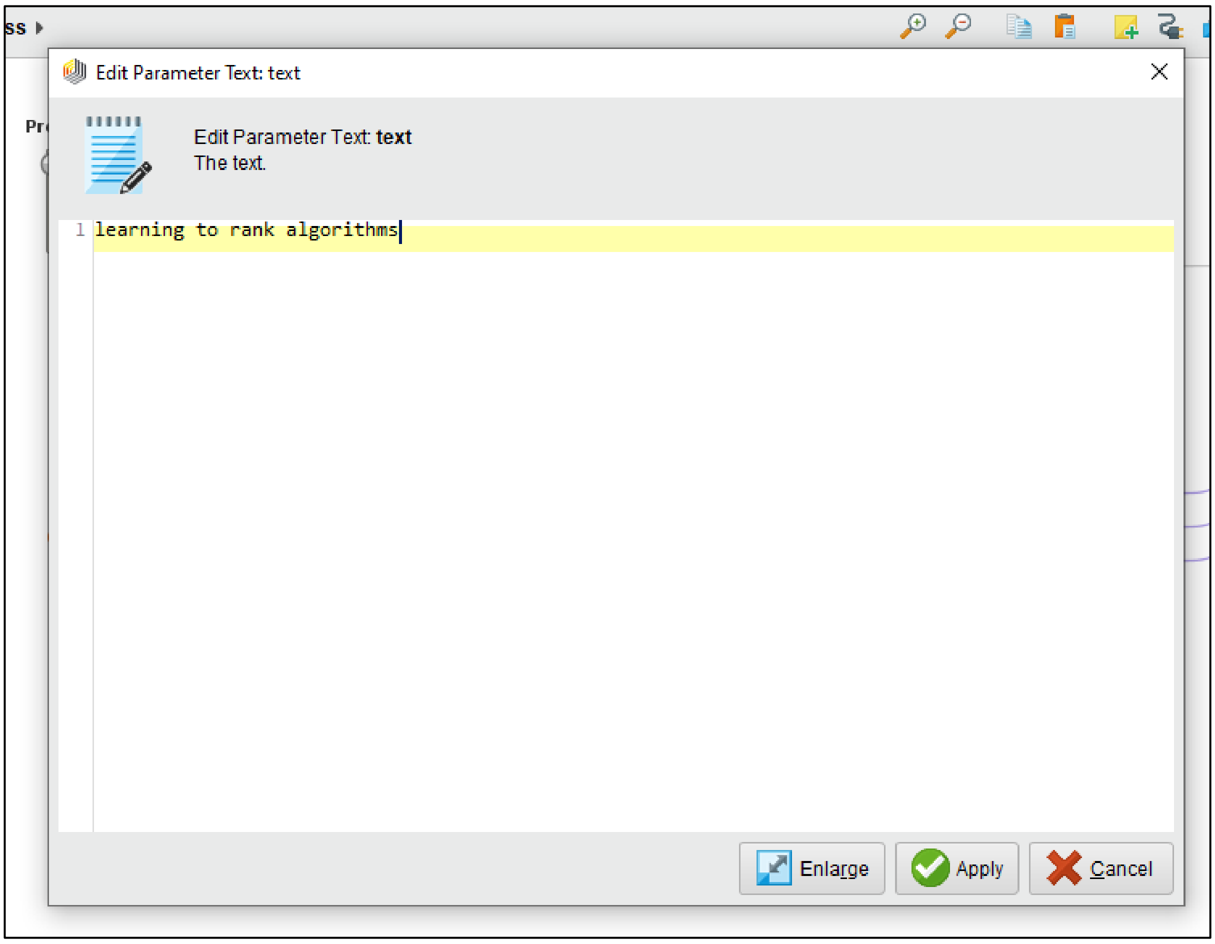
- Less unexpected delay
- While searching for a specific term, it requires some additional support or resources
- Compression operation is difficult and takes more time

Task 2B: Comparison about retrieval effectiveness with and without Stemming

Query1 “Learning to rank algorithms” with stemming

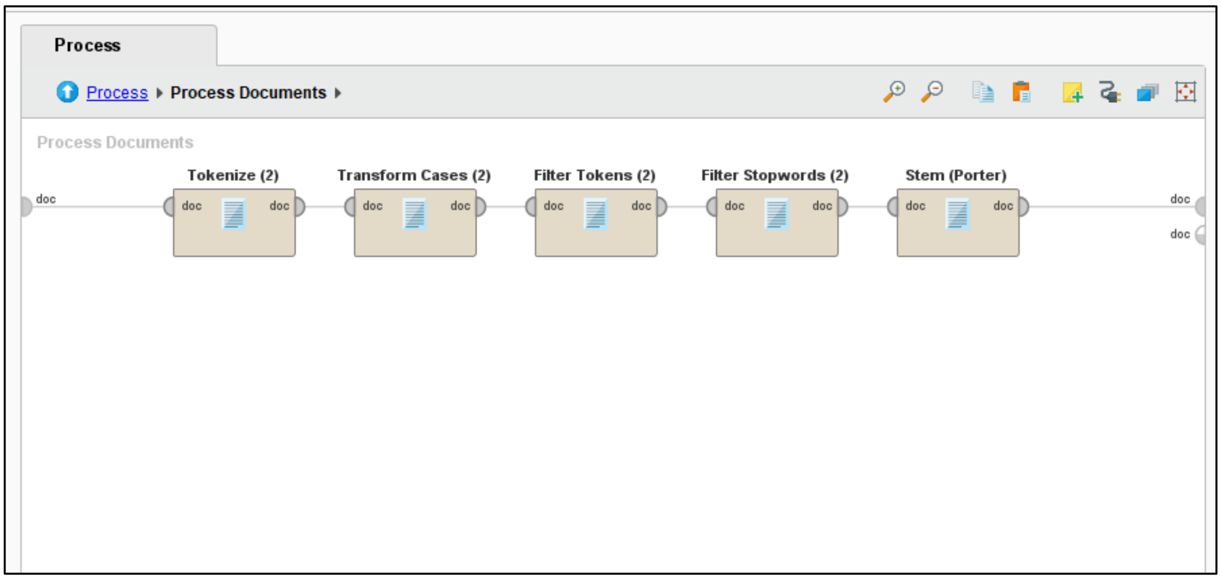


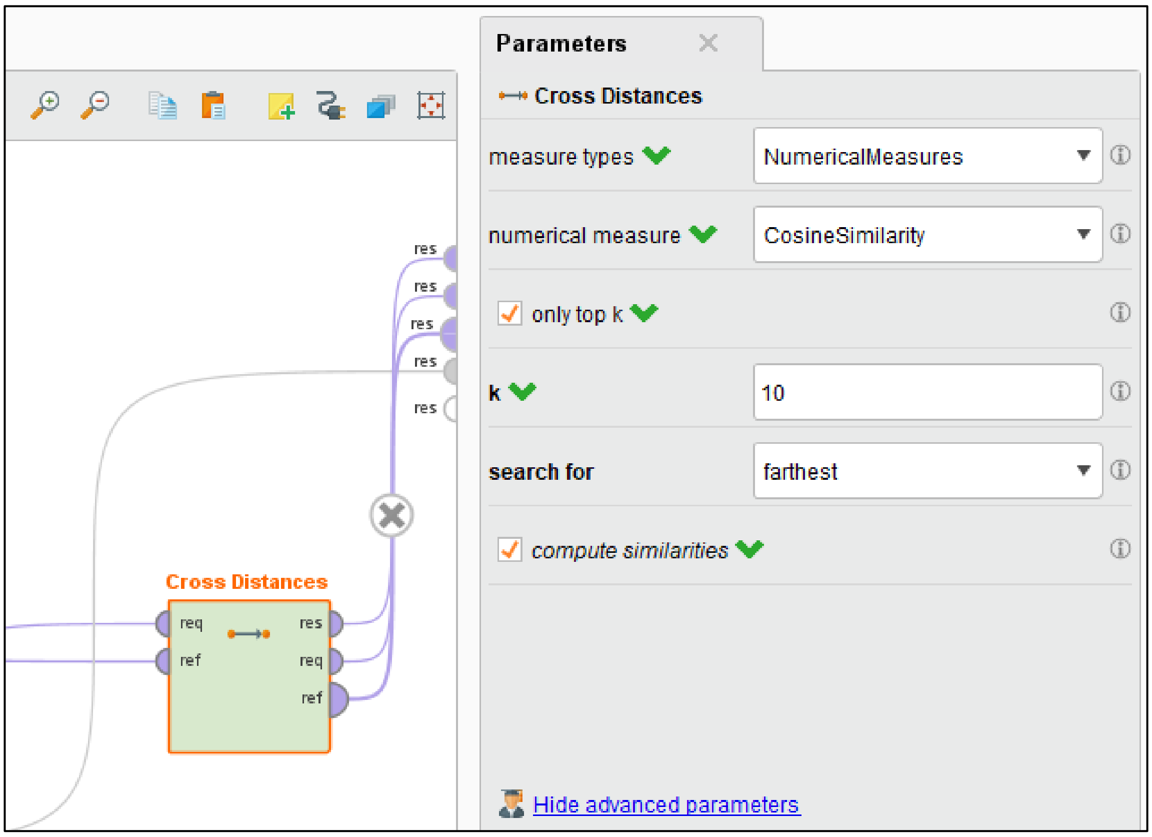




Edit Parameter Text: **text**
The text.

```
learning to rank algorithms
```



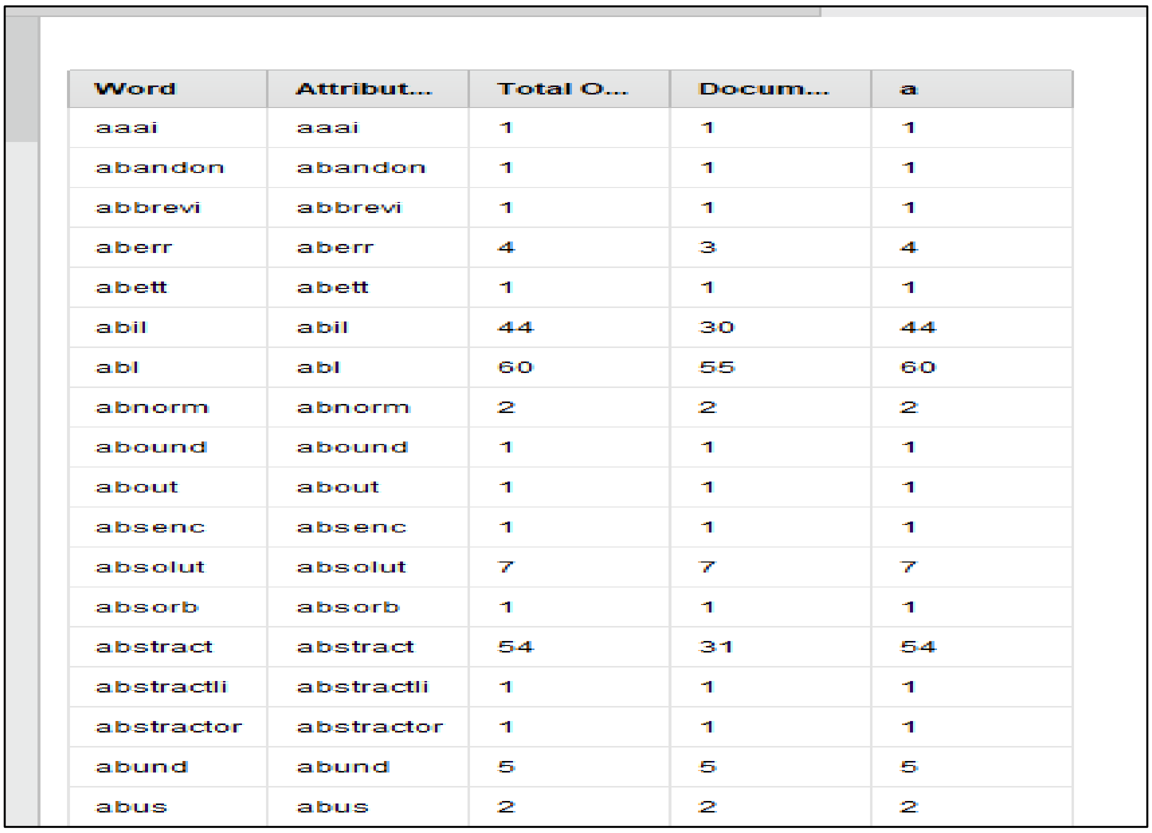


Parameters

Cross Distances

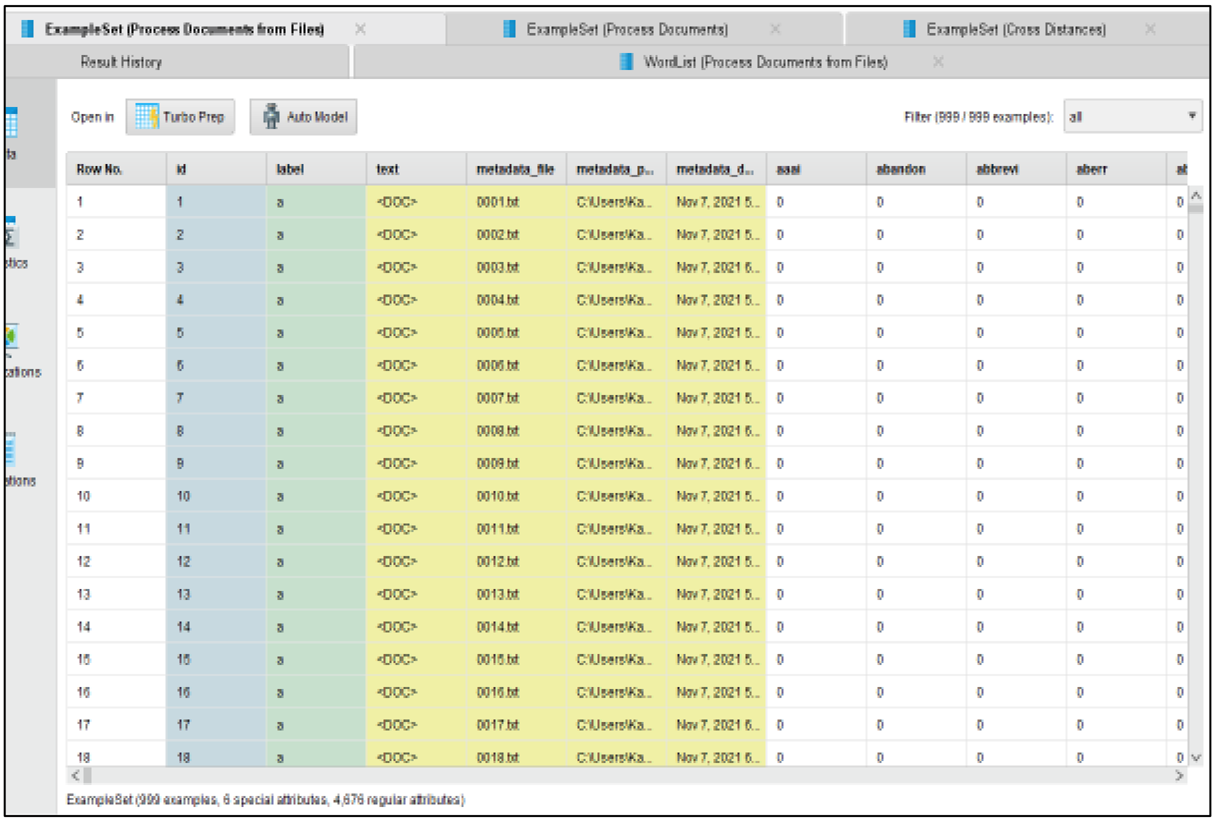
measure types	NumericalMeasures
numerical measure	CosineSimilarity
✓ only top k	
k	10
search for	farthest
✓ compute similarities	

Hide advanced parameters



Word	Attribut...	Total O...	Docum...	a
aaai	aaai	1	1	1
abandon	abandon	1	1	1
abbrevi	abbrevi	1	1	1
aberr	aberr	4	3	4
abett	abett	1	1	1
abil	abil	44	30	44
abl	abl	60	55	60
abnorm	abnorm	2	2	2
abound	abound	1	1	1
about	about	1	1	1
absenc	absenc	1	1	1
absolut	absolut	7	7	7
absorb	absorb	1	1	1
abstract	abstract	54	31	54
abstractli	abstractli	1	1	1
abstractor	abstractor	1	1	1
abund	abund	5	5	5
abus	abus	2	2	2

The above figure is the result set of the attribute list. For example, the term/attribute "abil" appears in 30 documents, and in a total of 44 times.

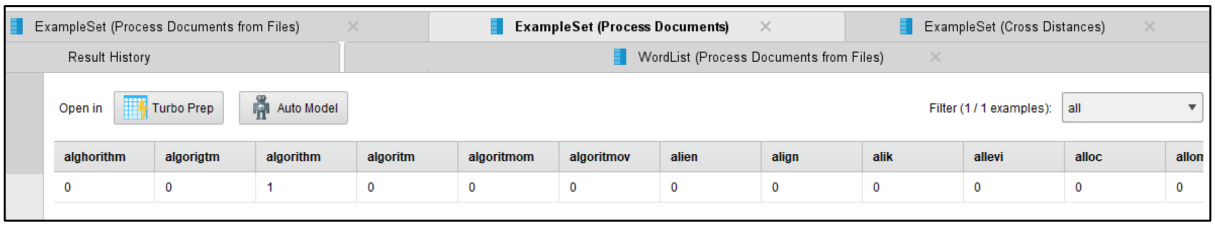

ExampleSet (Process Documents from Files)
ExampleSet (Process Documents)
ExampleSet (Cross Distances)
Result History
WordList (Process Documents from Files)
Open in
Turbo Prep
Auto Model
Filter (999 / 999 examples):
all

Row No.	id	label	text	metadata_file	metadata_p...	metadata_d...	aaai	abandon	abbrevi	aberr	ab
1	1	a	<DOC>	0001.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
2	2	a	<DOC>	0002.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
3	3	a	<DOC>	0003.txt	C:\Users\Ka...	Nov 7, 2021 6...	0	0	0	0	0
4	4	a	<DOC>	0004.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
5	5	a	<DOC>	0005.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
6	6	a	<DOC>	0006.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
7	7	a	<DOC>	0007.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
8	8	a	<DOC>	0008.txt	C:\Users\Ka...	Nov 7, 2021 6...	0	0	0	0	0
9	9	a	<DOC>	0009.txt	C:\Users\Ka...	Nov 7, 2021 6...	0	0	0	0	0
10	10	a	<DOC>	0010.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
11	11	a	<DOC>	0011.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
12	12	a	<DOC>	0012.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
13	13	a	<DOC>	0013.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
14	14	a	<DOC>	0014.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
15	15	a	<DOC>	0015.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
16	16	a	<DOC>	0016.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
17	17	a	<DOC>	0017.txt	C:\Users\Ka...	Nov 7, 2021 6...	0	0	0	0	0
18	18	a	<DOC>	0018.txt	C:\Users\Ka...	Nov 7, 2021 6...	0	0	0	0	0

ExampleSet (999 examples, 6 special attributes, 4,676 regular attributes)

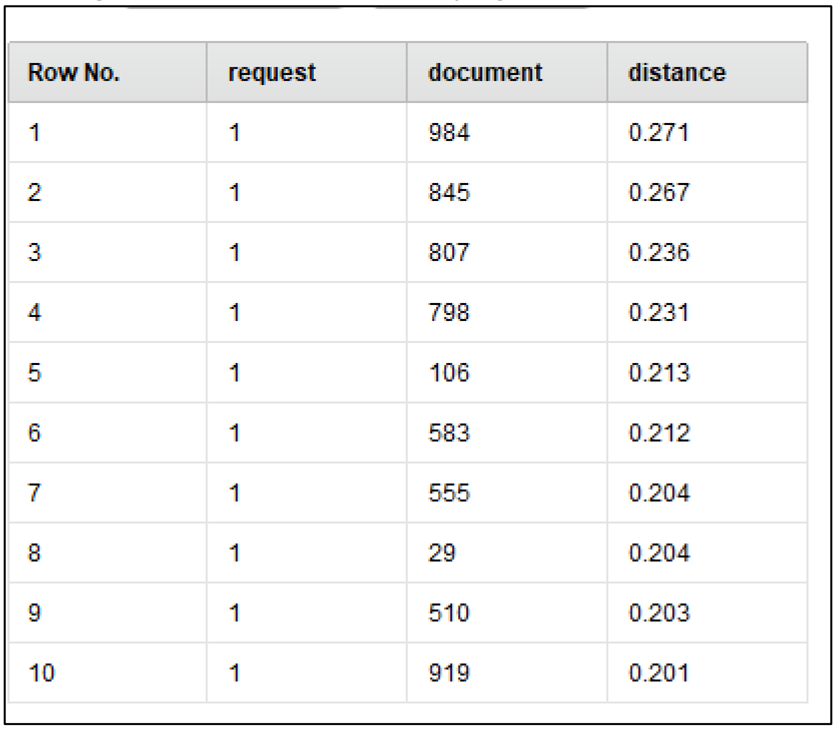
The term document matrix above shows there are 999 examples, with 4676 standard attributes. For the term "abil" if we look closely binary term 1 occurs 30 times as it appears in 30 documents. The execution time with stemming was around 25 seconds.

Below is the query term document matrix, with binary occurrences 1,0. As we can see there is a 1 underneath the term/attribute "algorithm". Similarly, the binary term will be 1 for all the terms matched in the query i.e. "learn" and "rank".


ExampleSet (Process Documents from Files)
ExampleSet (Process Documents)
ExampleSet (Cross Distances)
Result History
WordList (Process Documents from Files)
Open in
Turbo Prep
Auto Model
Filter (1 / 1 examples):
all

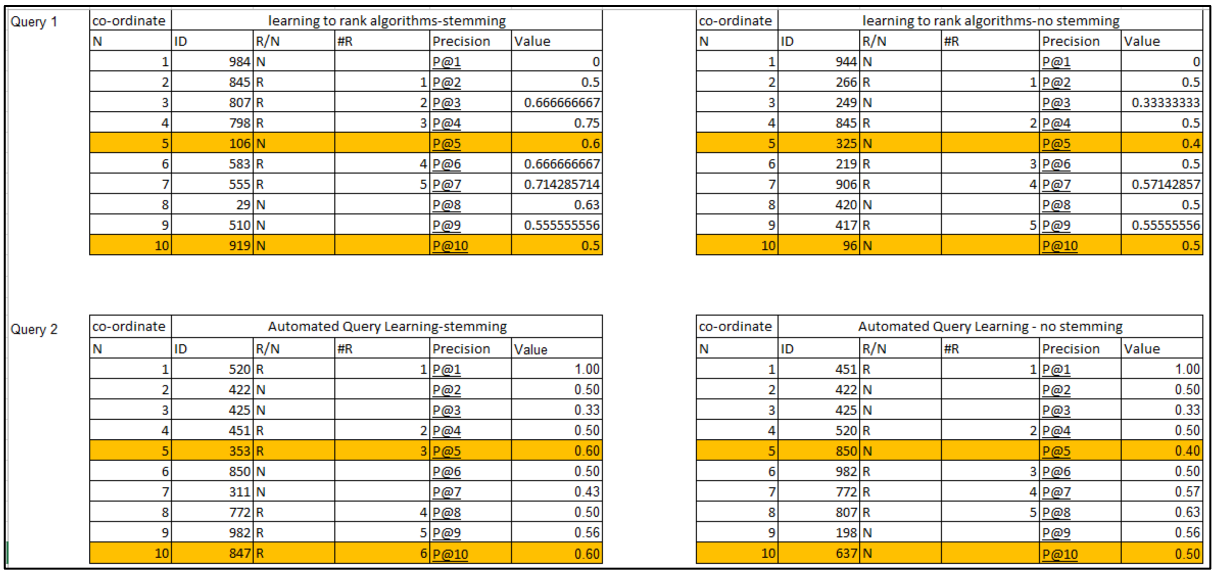
alghorithm	algorigtm	algorithm	algoritm	algoritmom	algoritmov	alien	align	alik	allevi	alloc	allon
0	0	1	0	0	0	0	0	0	0	0	0

The table below is extracted from Rapid miner results and shows the top ten documents selected to match the query with the distances between them. Using this and comparing the query terms and referring to the document we make a judgement on whether the document is relevant or not.



Row No.	request	document	distance
1	1	984	0.271
2	1	845	0.267
3	1	807	0.236
4	1	798	0.231
5	1	106	0.213
6	1	583	0.212
7	1	555	0.204
8	1	29	0.204
9	1	510	0.203
10	1	919	0.201

The process is repeated for both queries with and without stemming and we obtain the following tables below.



Query 1

co-ordinate	learning to rank algorithms-stemming				
N	ID	R/N	#R	Precision	Value
1	984	N		P@1	0
2	845	R	1	P@2	0.5
3	807	R	2	P@3	0.666666667
4	798	R	3	P@4	0.75
5	106	N		P@5	0.6
6	583	R	4	P@6	0.666666667
7	555	R	5	P@7	0.714285714
8	29	N		P@8	0.63
9	510	N		P@9	0.555555556
10	919	N		P@10	0.5

co-ordinate	learning to rank algorithms-no stemming				
N	ID	R/N	#R	Precision	Value
1	944	N		P@1	0
2	266	R	1	P@2	0.5
3	249	N		P@3	0.33333333
4	845	R	2	P@4	0.5
5	325	N		P@5	0.4
6	219	R	3	P@6	0.5
7	906	R	4	P@7	0.57142857
8	420	N		P@8	0.5
9	417	R	5	P@9	0.55555556
10	96	N		P@10	0.5

Query 2

co-ordinate	Automated Query Learning-stemming				
N	ID	R/N	#R	Precision	Value
1	520	R	1	P@1	1.00
2	422	N		P@2	0.50
3	425	N		P@3	0.33
4	451	R	2	P@4	0.50
5	353	R	3	P@5	0.60
6	850	N		P@6	0.50
7	311	N		P@7	0.43
8	772	R	4	P@8	0.50
9	982	R	5	P@9	0.56
10	847	R	6	P@10	0.60

co-ordinate	Automated Query Learning - no stemming				
N	ID	R/N	#R	Precision	Value
1	451	R	1	P@1	1.00
2	422	N		P@2	0.50
3	425	N		P@3	0.33
4	520	R	2	P@4	0.50
5	850	N		P@5	0.40
6	982	R	3	P@6	0.50
7	772	R	4	P@7	0.57
8	807	R	5	P@8	0.63
9	198	N		P@9	0.56
10	637	N		P@10	0.50

Please refer to appendix for calculation attached as a separate document.

A ranking is given to each document that is relevant and then the precision is calculated. As per the specification precision is calculated at 5 (P@5) and at 10 (P@10) for both queries with and without stemming.

Precision is the total number of retrieved documents that are relevant/ total number of documents retrieved (Springer 2011).

<u>Query 1: learning to rank algorithms</u>

Comparison of the Precision results:

Query 1	With stemming	Without stemming
P @5	0.6	0.4
P@10	0.5	0.5

With stemming, P@5 is 0.6 and P@10 is 0.5.We can see that the precision value is good when calculated at 5 but decreases when calculated at 10. This could possibly be due to Overstemming i.e. removing part of the stem taking it as a suffix by mistake. This type of error causes words that are not related also to be combined (Flores and Moreira 2016). The total number of relevant documents here with stemming is five. The higher the precision, the better the system.
Without stemming, P@5 is 0.4 and P@10 is 0.5. We can see that precision gets better with more number of documents. The total number of relevant documents without stemming is also five.

It is difficult to say from the above analysis for this query which is a better system. However, we may say that for this query stemming or no stemming, does not have a major impact on the results.

<u>Query 2: Automated Query Learning</u>

Comparison of the Precision results:

Query 2	With stemming	Without stemming
P @5	0.6	0.4
P@10	0.6	0.5

With stemming, P@5 is 0.6 and P@10 is also 0.6.We can see that the precision value is good when calculated at both the points. Whereas, we can see from the table above that, with stemming the value of P@5 is better than value of P@10. The total number of relevant documents here with stemming is six.

Without stemming, P@5 is 0.4 and P@10 is 0.5. We can see that precision gets better with more number of documents. The total number of relevant documents without stemming is five.

From the above analysis of precision values for this query that applying stemming gives a better precision and also returns more number of relevant documents to when stemming is not applied. Hence, we may choose to say that stemming works better for this particular query.

Stemming is combining of different forms of a word into a single representation, *i.e.,* the stem. For example, the terms *presentation, presenting,* and *presented* could all be stemmed to *present.* The stem does not have to be a valid word, but it needs to capture the meaning of the words (Flores and Moreira 2016).

Stemming is widely used in IR with the aim of increasing recall (i.e*.,* the number of relevant documents retrieved in response to a user query) (Baeza-Yates & Ribeiro-Neto, 2011). Another benefit is the reduction of the size of the index files, because a stem can represent many different words, resulting in fewer distinct index entries.

<u>Conclusion:</u>
Stemming will enhance recall but at the cost of decreased precision.

Task 3A Precision and Recall with Weighting Functions

Please refer to appendix for calculation attached as a separate document.

Task 3B Topic Retrieval and Weighting Function Retrieval

Topic retrieval (Boolean queries) are good for experts who know what they need and are specific in their understanding. It is also good for applications as they can have thousands of results. But, Boolean queries are not good for majority of the users because they cannot write such queries or it is too much work and also do not want to go through thousands of search results, which is true in case of web search.

TO: Term occurrence measures whether a term occurs in a document.

TF: Number of occurrences of a term t in document d. Assigning a weight equal to the number of occurrences of the term t in document d is called term frequency tf.

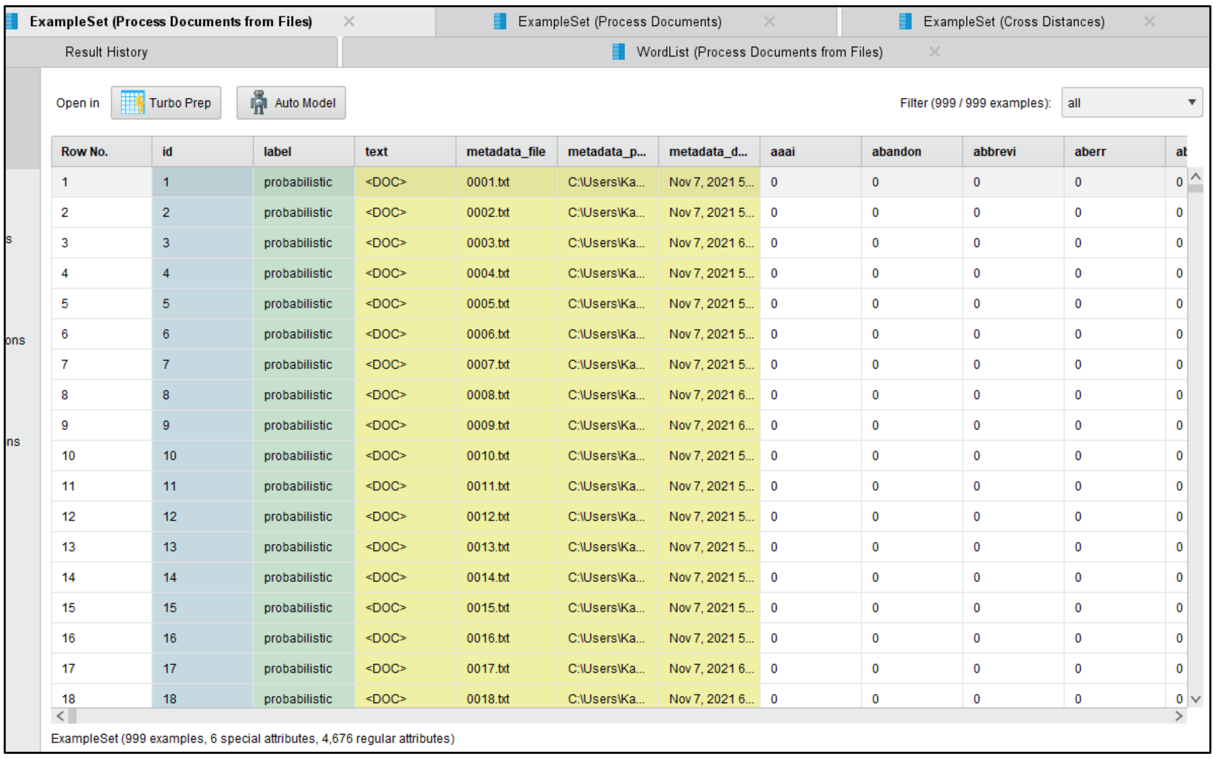
TF-IDF: *document frequency* df*t*, defined to be the number of documents in the collection that contain a term *t*. Denoting as usual the total number of documents in a collection by *N*, we define the *inverse document frequency* (idf) of a term *t* as follows:

idf*t* = log N/ df*t*

Thus the idf of a rare term is high, whereas the idf of a frequent term is likely to be low (Nayak and Raghavan 2009).

<u>Query 1</u> Below are the results obtained using Rapid miner process.

Term frequency document



ExampleSet (Process Documents from Files) | ExampleSet (Process Documents) | ExampleSet (Cross Distances)

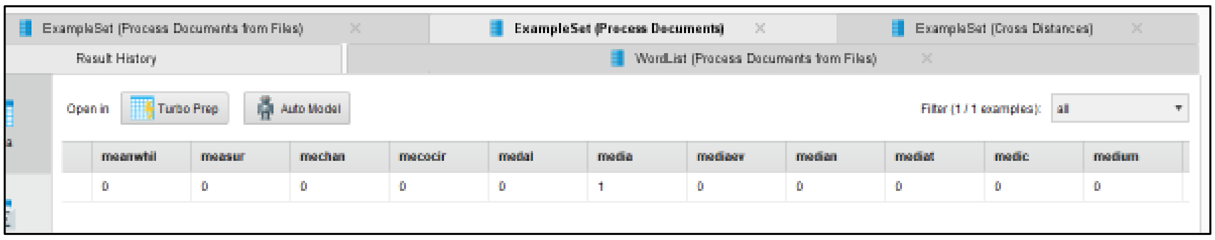
Result History | WordList (Process Documents from Files)

Open in Turbo Prep | Auto Model | Filter (999 / 999 examples): all

Row No.	id	label	text	metadata_file	metadata_p...	metadata_d...	aaai	abandon	abbrevi	aberr	at
1	1	probabilistic	<DOC>	0001.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
2	2	probabilistic	<DOC>	0002.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
3	3	probabilistic	<DOC>	0003.txt	C:\Users\Ka...	Nov 7, 2021 6...	0	0	0	0	0
4	4	probabilistic	<DOC>	0004.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
5	5	probabilistic	<DOC>	0005.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
6	6	probabilistic	<DOC>	0006.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
7	7	probabilistic	<DOC>	0007.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
8	8	probabilistic	<DOC>	0008.txt	C:\Users\Ka...	Nov 7, 2021 6...	0	0	0	0	0
9	9	probabilistic	<DOC>	0009.txt	C:\Users\Ka...	Nov 7, 2021 6...	0	0	0	0	0
10	10	probabilistic	<DOC>	0010.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
11	11	probabilistic	<DOC>	0011.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
12	12	probabilistic	<DOC>	0012.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
13	13	probabilistic	<DOC>	0013.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
14	14	probabilistic	<DOC>	0014.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
15	15	probabilistic	<DOC>	0015.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
16	16	probabilistic	<DOC>	0016.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
17	17	probabilistic	<DOC>	0017.txt	C:\Users\Ka...	Nov 7, 2021 6...	0	0	0	0	0
18	18	probabilistic	<DOC>	0018.txt	C:\Users\Ka...	Nov 7, 2021 6...	0	0	0	0	0

ExampleSet (999 examples, 6 special attributes, 4,676 regular attributes)

Query term frequency document



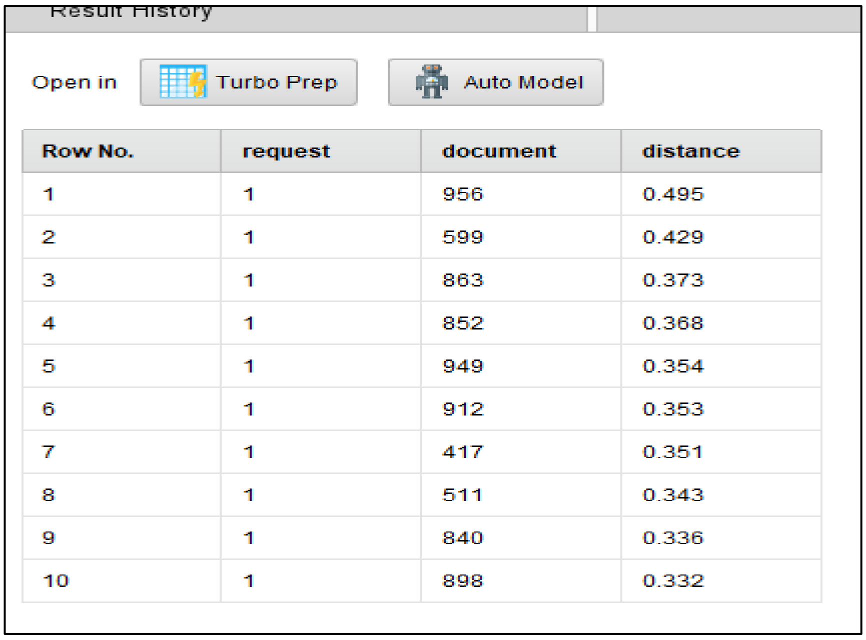
ExampleSet (Process Documents from Files) | ExampleSet (Process Documents) | ExampleSet (Cross Distances)

Result History | WordList (Process Documents from Files)

Open in Turbo Prep | Auto Model | Filter (1 / 1 examples): all

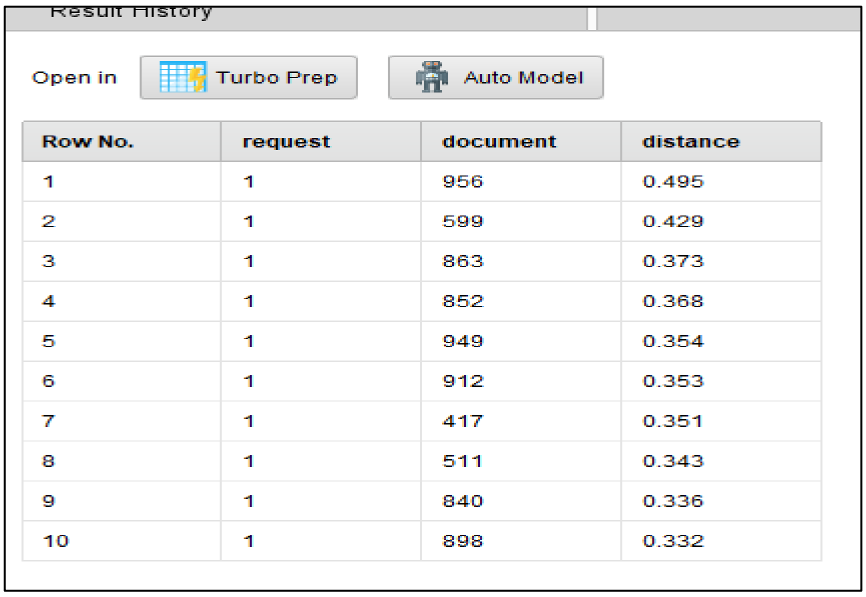
meanwhil	measur	mechan	mecocir	medal	media	mediaev	median	mediat	medic	medium
0	0	0	0	0	1	0	0	0	0	0

TO



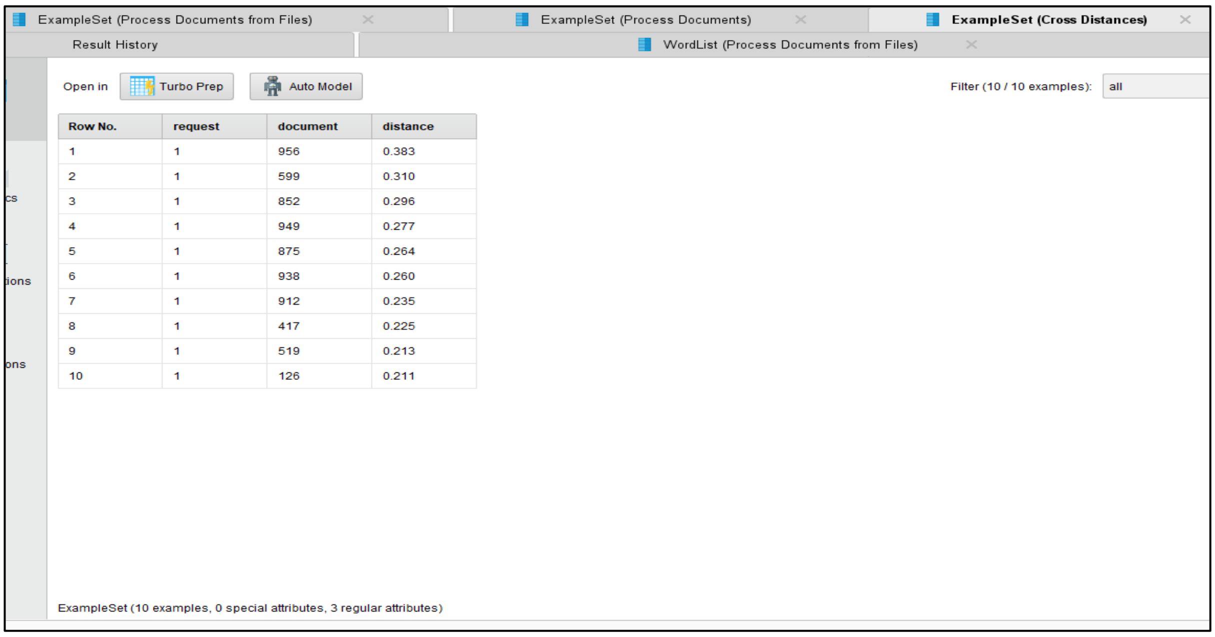
Row No.	request	document	distance
1	1	956	0.495
2	1	599	0.429
3	1	863	0.373
4	1	852	0.368
5	1	949	0.354
6	1	912	0.353
7	1	417	0.351
8	1	511	0.343
9	1	840	0.336
10	1	898	0.332

TF



Row No.	request	document	distance
1	1	956	0.495
2	1	599	0.429
3	1	863	0.373
4	1	852	0.368
5	1	949	0.354
6	1	912	0.353
7	1	417	0.351
8	1	511	0.343
9	1	840	0.336
10	1	898	0.332

TF-IDF



ExampleSet (Process Documents from Files) | ExampleSet (Process Documents) | **ExampleSet (Cross Distances)**

Result History | WordList (Process Documents from Files)

Open in | Turbo Prep | Auto Model | Filter (10 / 10 examples): all

Row No.	request	document	distance
1	1	956	0.383
2	1	599	0.310
3	1	852	0.296
4	1	949	0.277
5	1	875	0.264
6	1	938	0.260
7	1	912	0.235
8	1	417	0.225
9	1	519	0.213
10	1	126	0.211

ExampleSet (10 examples, 0 special attributes, 3 regular attributes)

Comparison table:

Query 1	TO	TF	TF-IDF
Retreival time	20 seconds	20 seconds	21 seconds
No. of relevant documents	4	4	5
Area on the graph	less compared to tf-idf	less compared to tf-idf	more compared to to,tf

Precision and Recall calculations and result analysis

TO

Q1-tot rel	Q2-tot rel
6	4

TO	probabilistic models for social			Recall	Precision	Recall	Precision
N	ID	R/N	#R			0.1	1
1	956	R	1	0.17	1.00	0.2	1
2	599	R	2	0.33	1.00	0.3	1
3	863	N				0.4	1
4	852	R	3	0.50	0.75	0.5	0.75
5	949	N				0.6	0.75
6	912	N				0.7	0.4
7	417	N				0.8	0
8	511	N				0.9	0
9	840	N				1	0
10	898	R	4	0.67	0.40		

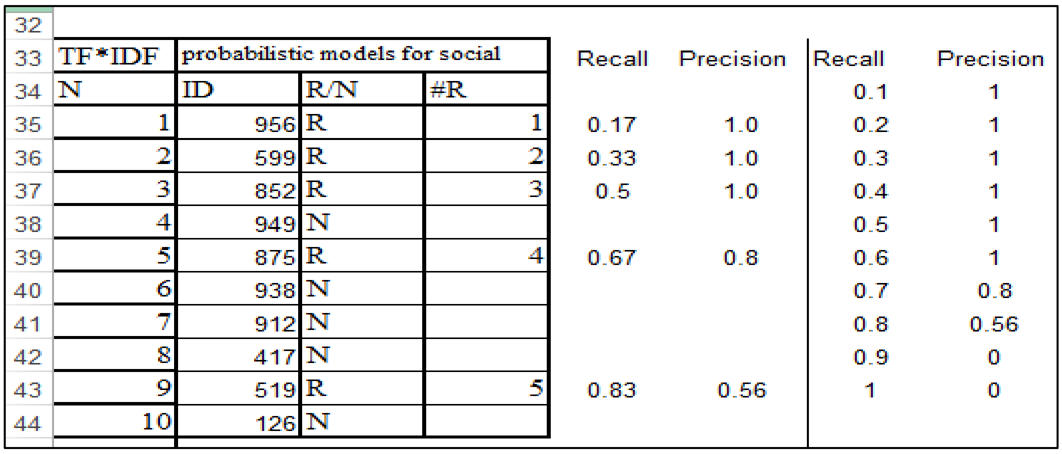
In the table for term occurrence which records the presence of a term, above we see that the number of relevant documents returned are four and the number of non-relevant documents are 6, which is more. Also, we can observe from this result, the order in which the relevant documents are returned. The top two documents retrieved are relevant, then the third relevant document at number 4, so on and so forth. A good measure returns the documents in highest order of their relevance. We can see that the 4th relevant document is appearing at the very bottom i.e. after a few non relevant documents.

TF

TF	probabilistic models for social			Recall	Precision	Recall	Precision
N	ID	R/N	#R			0.1	1
1	956	R	1	0.17	1.00	0.2	1
2	599	R	2	0.33	1.00	0.3	1
3	863	N				0.4	1
4	852	R	3	0.50	0.75	0.5	0.75
5	949	N				0.6	0.75
6	912	N				0.7	0.4
7	417	N				0.8	0
8	511	N				0.9	0
9	840	N				1	0
10	898	R	4	0.67	0.4		

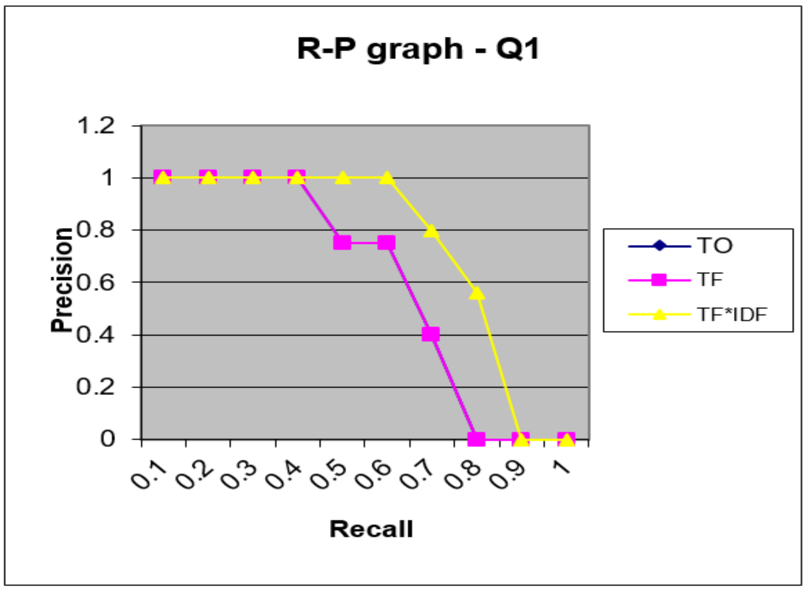
In the table for term frequency measure above we see that the number of relevant documents returned are four and the number of non-relevant documents are six, which is more. Also, we can observe from this result, the order in which the relevant documents are returned. The top two documents retrieved are relevant, then the third relevant document at number 4, so on and so forth. A good measure returns the documents in highest order of their relevance. For tf, we can see that the 4th relevant document is appearing at the very bottom i.e. after a few non relevant documents.

TF-IDF



32								
33	TF*IDF	probabilistic models for social			Recall	Precision	Recall	Precision
34	N	ID	R/N	#R			0.1	1
35	1	956	R	1	0.17	1.0	0.2	1
36	2	599	R	2	0.33	1.0	0.3	1
37	3	852	R	3	0.5	1.0	0.4	1
38	4	949	N				0.5	1
39	5	875	R	4	0.67	0.8	0.6	1
40	6	938	N				0.7	0.8
41	7	912	N				0.8	0.56
42	8	417	N				0.9	0
43	9	519	R	5	0.83	0.56	1	0
44	10	126	N					

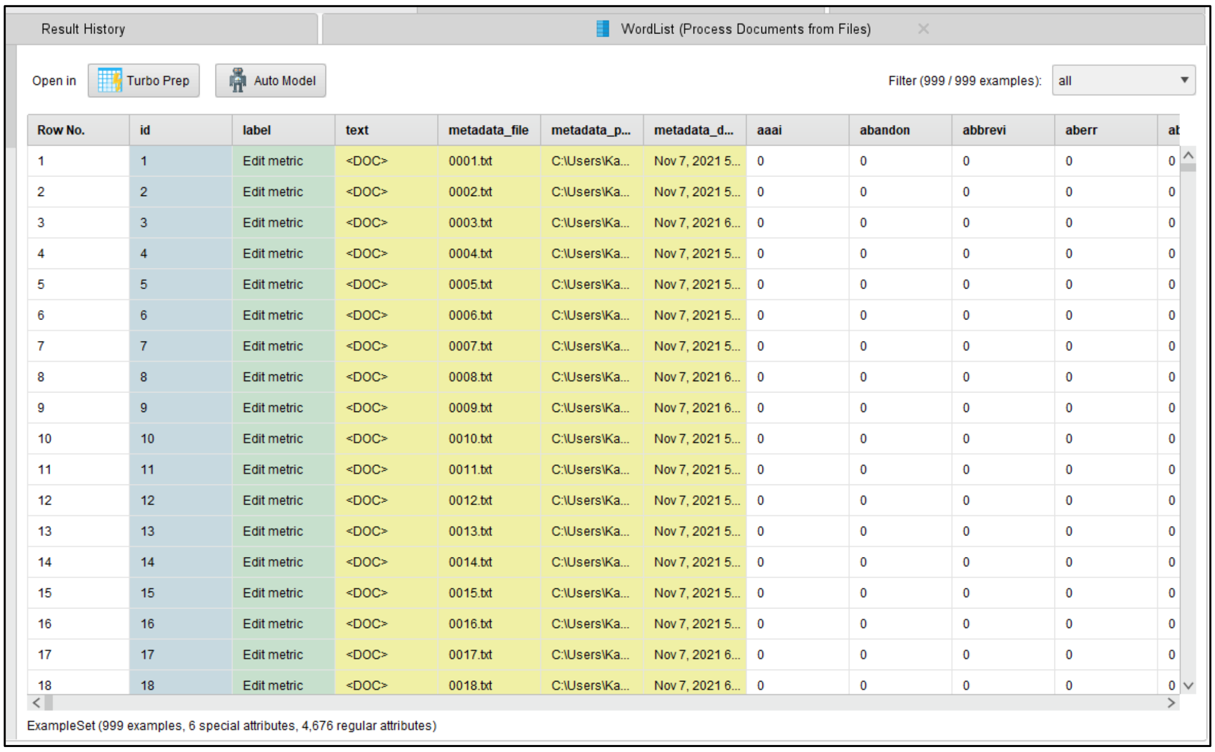
In the table for TF-IDF measure above we see that the number of relevant documents returned are five and the number of non-relevant documents are also five. Also, we can observe from this result, the order in which the relevant documents are returned. The top three documents retrieved are relevant, then the fourth relevant document is at number 5, then the fifth and the last relevant document is at the 9th position. A good measure returns the documents in highest order of their relevance. However, here we can still say that the relevant documents are retrieved in a good order.



Here, when we look at the precision and recall curves for the three measures to, tf, tf-idf for query 1, we can see that the area covered by the measure tf-idf is more than the area captured by to and tf respectively. Hence, for this query we may conclude that tf-idf clearly out shines and so works better.

Query 2 Below are the results obtained using Rapid miner process.

Term Frequency document



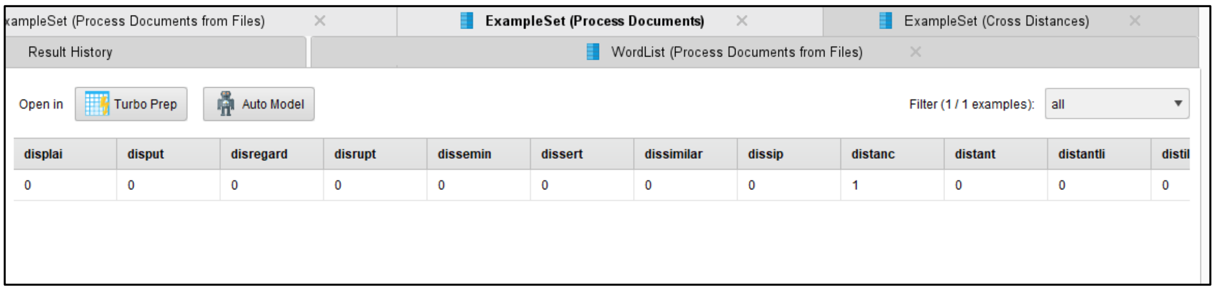
Result History | WordList (Process Documents from Files)

Open in | Turbo Prep | Auto Model | Filter (999 / 999 examples): all

Row No.	id	label	text	metadata_file	metadata_p...	metadata_d...	aaai	abandon	abbrevi	aberr	ab
1	1	Edit metric	<DOC>	0001.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
2	2	Edit metric	<DOC>	0002.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
3	3	Edit metric	<DOC>	0003.txt	C:\Users\Ka...	Nov 7, 2021 6...	0	0	0	0	0
4	4	Edit metric	<DOC>	0004.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
5	5	Edit metric	<DOC>	0005.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
6	6	Edit metric	<DOC>	0006.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
7	7	Edit metric	<DOC>	0007.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
8	8	Edit metric	<DOC>	0008.txt	C:\Users\Ka...	Nov 7, 2021 6...	0	0	0	0	0
9	9	Edit metric	<DOC>	0009.txt	C:\Users\Ka...	Nov 7, 2021 6...	0	0	0	0	0
10	10	Edit metric	<DOC>	0010.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
11	11	Edit metric	<DOC>	0011.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
12	12	Edit metric	<DOC>	0012.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
13	13	Edit metric	<DOC>	0013.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
14	14	Edit metric	<DOC>	0014.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
15	15	Edit metric	<DOC>	0015.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
16	16	Edit metric	<DOC>	0016.txt	C:\Users\Ka...	Nov 7, 2021 5...	0	0	0	0	0
17	17	Edit metric	<DOC>	0017.txt	C:\Users\Ka...	Nov 7, 2021 6...	0	0	0	0	0
18	18	Edit metric	<DOC>	0018.txt	C:\Users\Ka...	Nov 7, 2021 6...	0	0	0	0	0

ExampleSet (999 examples, 6 special attributes, 4,676 regular attributes)

Query Term Frequency Document



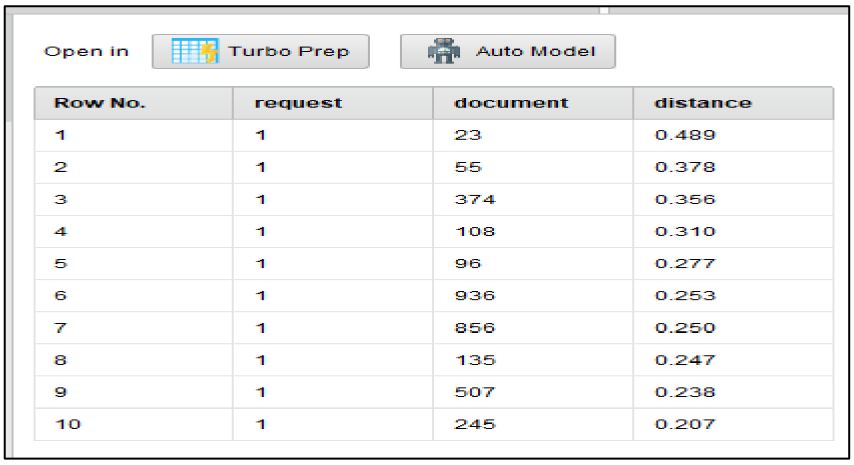
xampleSet (Process Documents from Files) | ExampleSet (Process Documents) | ExampleSet (Cross Distances)

Result History | WordList (Process Documents from Files)

Open in | Turbo Prep | Auto Model | Filter (1 / 1 examples): all

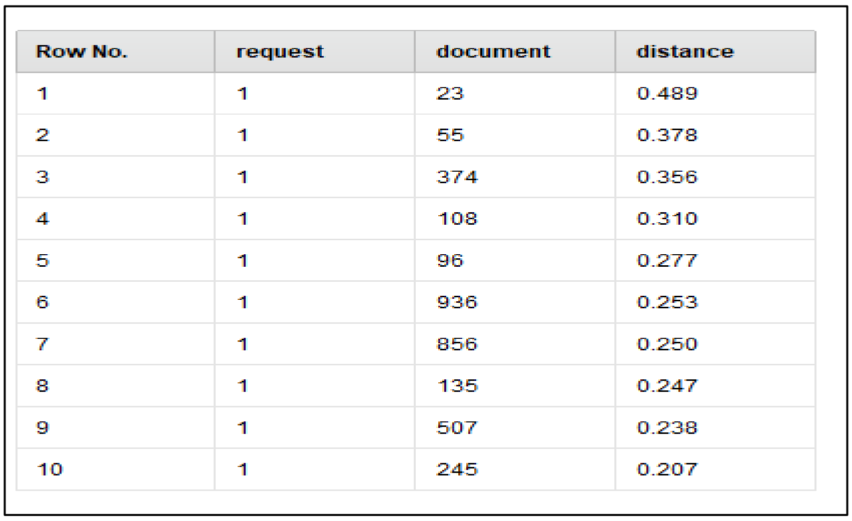
displai	disput	disregard	disrupt	dissemin	dissert	dissimilar	dissip	distanc	distant	distantli	distil
0	0	0	0	0	0	0	0	1	0	0	0

TO



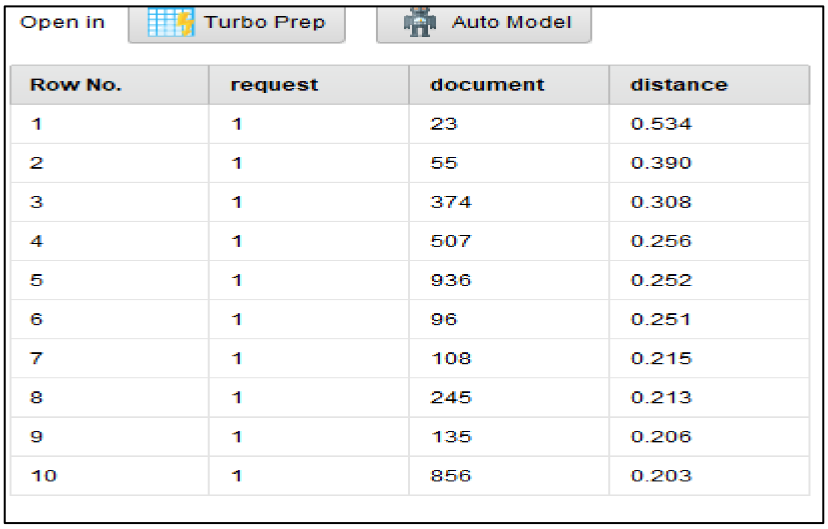
Row No.	request	document	distance
1	1	23	0.489
2	1	55	0.378
3	1	374	0.356
4	1	108	0.310
5	1	96	0.277
6	1	936	0.253
7	1	856	0.250
8	1	135	0.247
9	1	507	0.238
10	1	245	0.207

TF



Row No.	request	document	distance
1	1	23	0.489
2	1	55	0.378
3	1	374	0.356
4	1	108	0.310
5	1	96	0.277
6	1	936	0.253
7	1	856	0.250
8	1	135	0.247
9	1	507	0.238
10	1	245	0.207

TF-IDF



Row No.	request	document	distance
1	1	23	0.534
2	1	55	0.390
3	1	374	0.308
4	1	507	0.256
5	1	936	0.252
6	1	96	0.251
7	1	108	0.215
8	1	245	0.213
9	1	135	0.206
10	1	856	0.203

Comparison table:

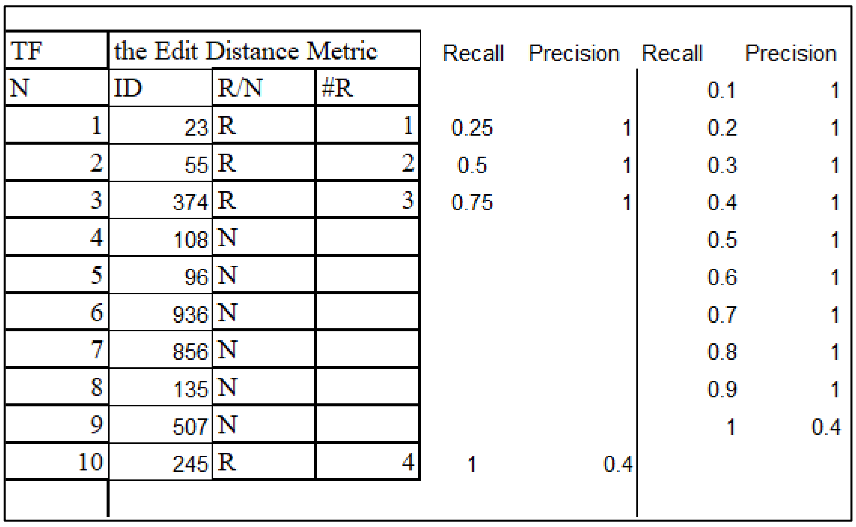
Query 2	TO	TF	TF-IDF
Retreival time(sec)	22	22	31
No. of relevant documents	4	4	4
Area on the graph	Approx. same	Approx. same	Approx. same

TO

TO	the Edit Distance Metric			Recall	Precision	Recall	Precision
N	ID	R/N	#R			0.1	1
1	23	R	1	0.25	1	0.2	1
2	55	R	2	0.5	1	0.3	1
3	374	R	3	0.75	1	0.4	1
4	108	N				0.5	1
5	96	N				0.6	1
6	936	N				0.7	1
7	856	N				0.8	1
8	135	N				0.9	1
9	507	N				1	0.4
10	245	R	4	1	0.4		

In the table for term occurrence which records the presence of a term, above we see that the number of relevant documents returned are four and the number of non-relevant documents are six, which is more in number. Also, we can observe from this result, the order in which the relevant documents are returned. The top three documents retrieved are relevant, then the fourth and last relevant document at number 10. A good measure returns the documents in highest order of their relevance. For to, we can see that the 4th relevant document is appearing at the very bottom i.e. after a few non relevant documents.

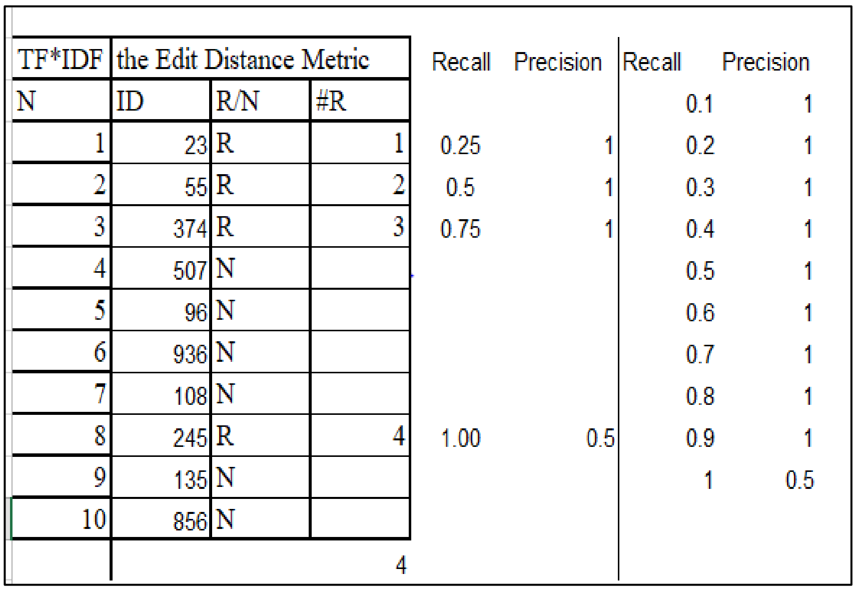
TF



TF	the Edit Distance Metric			Recall	Precision	Recall	Precision
N	ID	R/N	#R			0.1	1
1	23	R	1	0.25	1	0.2	1
2	55	R	2	0.5	1	0.3	1
3	374	R	3	0.75	1	0.4	1
4	108	N				0.5	1
5	96	N				0.6	1
6	936	N				0.7	1
7	856	N				0.8	1
8	135	N				0.9	1
9	507	N				1	0.4
10	245	R	4	1	0.4		

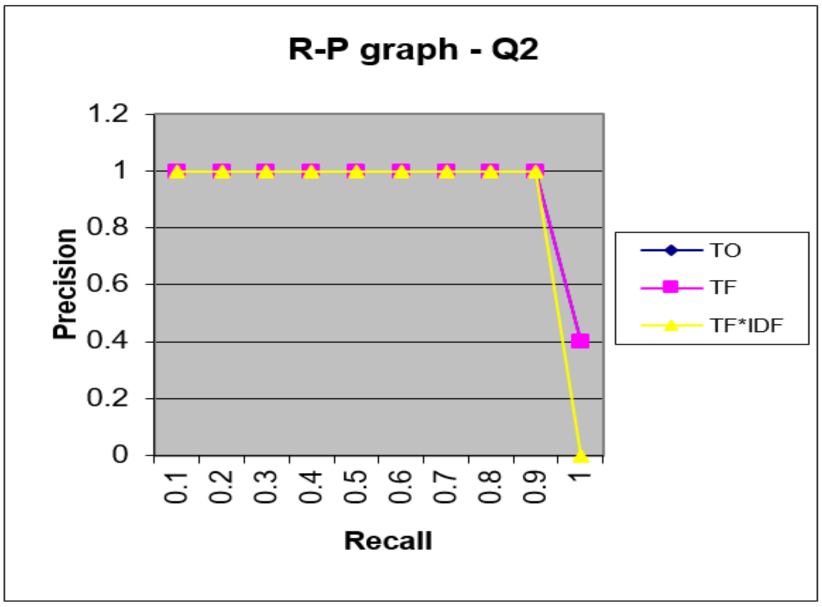
In the table for term frequency measure above we see that the number of relevant documents returned are four and the number of non-relevant documents are six, which is more in number. Also, we can observe from this result, the order in which the relevant documents are returned. The top three documents retrieved are relevant, then the fourth relevant document at number 10. A good measure returns the documents in highest order of their relevance. For tf, we can see that the 4th relevant document is appearing at the very bottom i.e. after all the non-relevant documents.

TF-IDF



TF*IDF	the Edit Distance Metric			Recall	Precision	Recall	Precision
N	ID	R/N	#R			0.1	1
1	23	R	1	0.25	1	0.2	1
2	55	R	2	0.5	1	0.3	1
3	374	R	3	0.75	1	0.4	1
4	507	N				0.5	1
5	96	N				0.6	1
6	936	N				0.7	1
7	108	N				0.8	1
8	245	R	4	1.00	0.5	0.9	1
9	135	N				1	0.5
10	856	N					
			4				

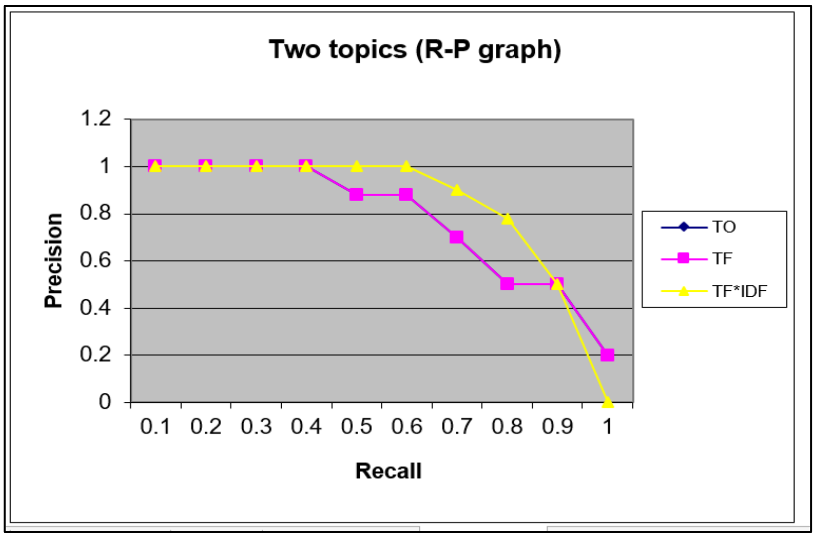
In the table for TF-IDF measure above we see that the number of relevant documents returned are four and the number of non-relevant documents are six, which is more in number. Also, we can observe from this result, the order in which the relevant documents are returned. The top three documents retrieved are relevant, then the fourth relevant document is at number 8, then the fifth or the next relevance may be at position 11. A good measure returns the documents in highest order of their relevance.



When we look at the precision and recall curves for the three measures to, tf, tf-idf for query 2, we can see that the area covered by the all the three measures is approximately the same. But when we see closely, to and tf cover a slightly larger area. Therefore, we can say that to and tf perform better for this query and tf-idf is not a very useful measure here as it gives approximately the same results as the other two.

Comparison of the two queries/topics

When an average of the measures is calculated and we plot our calculations for both queries onto a graph, the below curves are obtained.



As we see that the area captured by to, tf is comparatively less than the area captured by tf-idf. Another thing to observe in the graphs above is that tf-idf provides more consistent results. So, when we are looking at topic retrieval with the queries given in the assignment tf-idf is a better measure.

The result for the query terms after pre-processing for query 1 i.e. “probabilistic models for social media” obtained from Rapid miner is “probabilist model social media”. A reasonable change is observed here in the query. Whereas if we look at query 2, there is no much change in the terms after pre-processing, i.e. “the Edit Distance Metric”, is transformed to “edit distanc metric”.
We can say by conducting the above task that to and tf work better for simpler queries in terms of number of terms used in the query and the term rarity. Whereas, tf-idf is seen to give better results for more complex queries where there are more number of terms and term rarity is also high.

References

- Massie, Dr. Stewart., 2021. *Inverted Index & the Boolean Model*. [online lecture/tutorial/seminar]. Information Retrieval Systems. Robert Gordon University. School of Computing and Digital Media, 28/09/2021. Available from: https://eu.bbcollab.com/collab/ui/session/playback [Accessed 27/10/2021].
- Kopanev, Ihor., 2019. *A brief explanation of Inverted Index*. [online]. Medium. Available from: https://medium.com/@igorkopanev/a-brief-explanation-of-the-inverted-index-f082993f8605 [30/10/2021 date].
- Bhat, Ritesh., 2020. *Introduction to Inverted Indexes*. [online]. Dev Community. Available from: https://dev.to/im_bhatman/introduction-to-inverted-indexes-l04 [30/10/2021].
- Samir, A. & Lahbib, Z., 2018. *Stemming and Lemmatization for Information Retrieval Systems in Amazigh Language.* s.l., International Conference on Big Data, Cloud and Applications.
- Ting K.M. ,2011. *Precision and Recall*. In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-30164-8_652
- Flores, Felipe N. and Moreira, Viviane P. 2016. Assessing the impact of Stemming Accuracy on Information Retrieval – A multilingual perspective. *Information Processing and Management*, 52(2016), pp. 2-3.
- Baeza-Yates, R. A., & Ribeiro-Neto, B. (2011). Modern information retrieval (2nd ed.). Addison-Wesley Longman Publishing Co., Inc.
- Nayak, Pandu and Raghavan, Prabhakar. 2009. *Scoring, term weighting and the vector space model*. [online]. Cambridge: Cambridge University Press. Available from: https://web.stanford.edu/class/cs276/19handouts/lecture6-tfidf-1per.pdf [Accessed 11/11/2021].