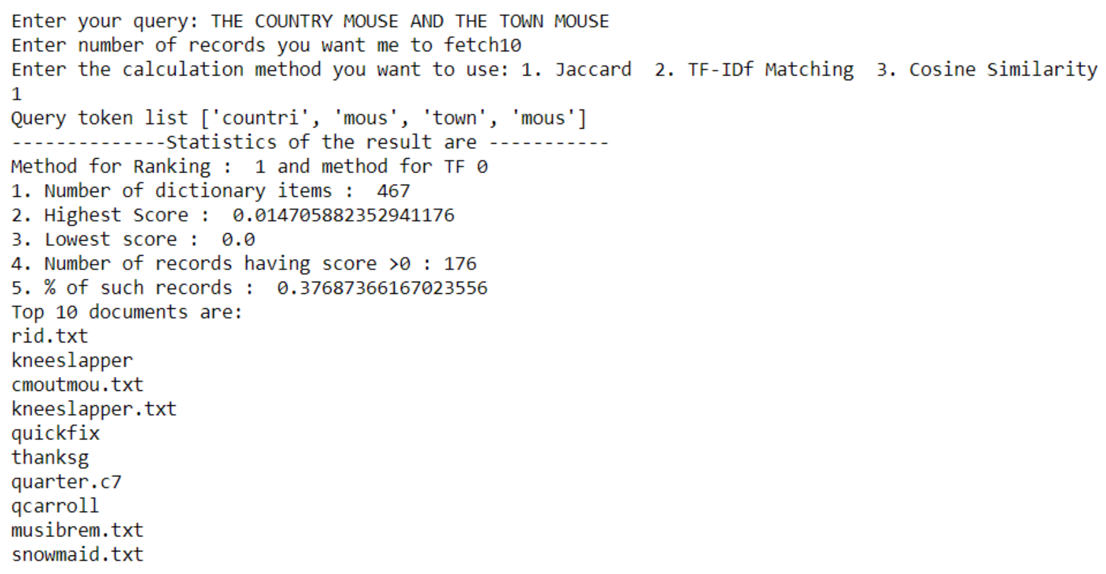
IR Assignment 2

Question 1: Analysis and Output

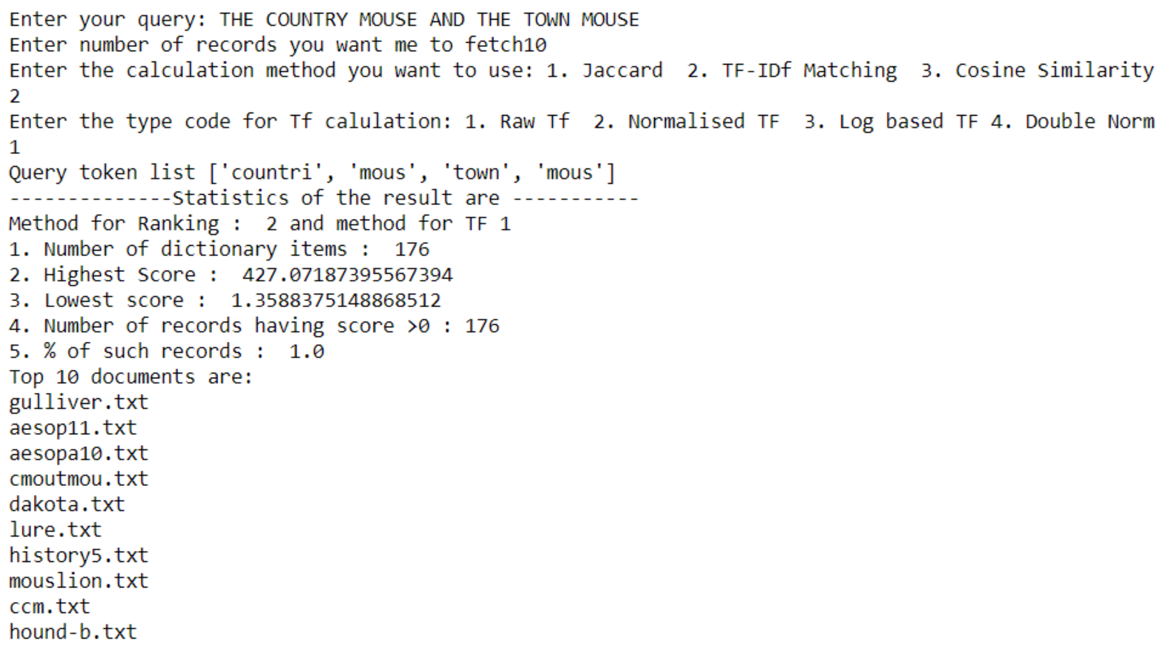
Jaccard Based Tool



```
Enter your query: THE COUNTRY MOUSE AND THE TOWN MOUSE
Enter number of records you want me to fetch10
Enter the calculation method you want to use: 1. Jaccard  2. TF-IDf Matching  3. Cosine Similarity
1
Query token list ['countri', 'mous', 'town', 'mous']
--------------Statistics of the result are -----------
Method for Ranking :  1 and method for TF 0
1. Number of dictionary items :  467
2. Highest Score :  0.014705882352941176
3. Lowest score :  0.0
4. Number of records having score >0 : 176
5. % of such records :  0.37687366167023556
Top 10 documents are:
rid.txt
kneeslapper
cmoutmou.txt
kneeslapper.txt
quickfix
thanksg
quarter.c7
qcarroll
musibrem.txt
snowmaid.txt
```

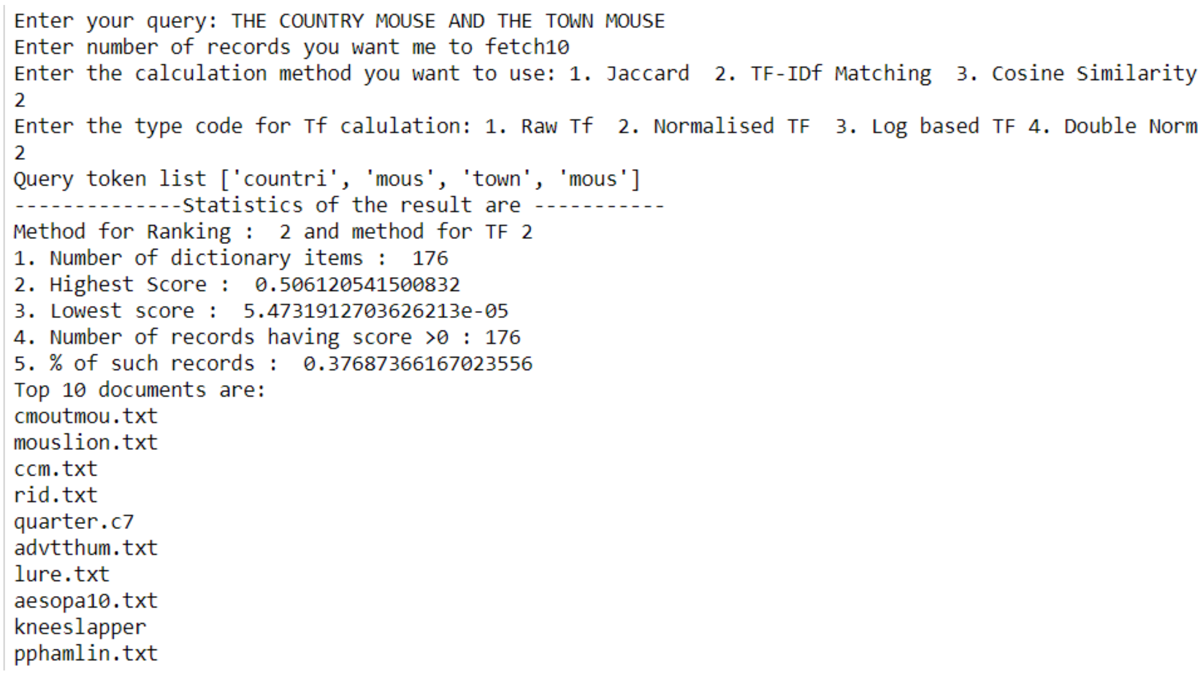
TF-IDF Matching Score

Raw TF score



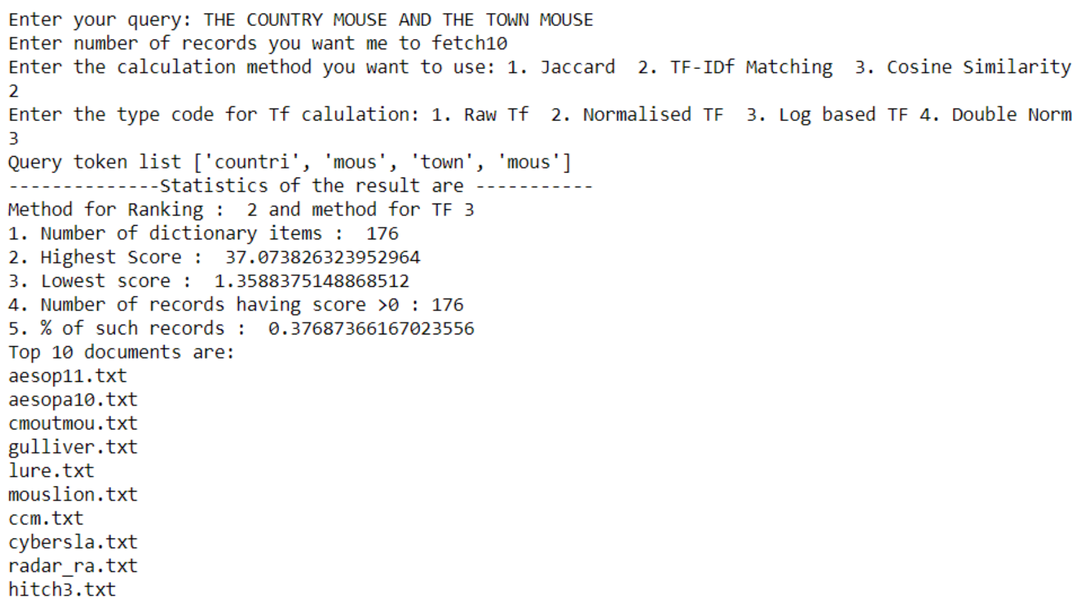
```
Enter your query: THE COUNTRY MOUSE AND THE TOWN MOUSE
Enter number of records you want me to fetch10
Enter the calculation method you want to use: 1. Jaccard  2. TF-IDf Matching  3. Cosine Similarity
2
Enter the type code for Tf calulation: 1. Raw Tf  2. Normalised TF  3. Log based TF 4. Double Norm
1
Query token list ['countri', 'mous', 'town', 'mous']
--------------Statistics of the result are -----------
Method for Ranking :  2 and method for TF 1
1. Number of dictionary items :  176
2. Highest Score :  427.07187395567394
3. Lowest score :  1.3588375148868512
4. Number of records having score >0 : 176
5. % of such records :  1.0
Top 10 documents are:
gulliver.txt
aesop11.txt
aesopa10.txt
cmoutmou.txt
dakota.txt
lure.txt
history5.txt
mouslion.txt
ccm.txt
hound-b.txt
```

Normalised TF Score



```
Enter your query: THE COUNTRY MOUSE AND THE TOWN MOUSE
Enter number of records you want me to fetch10
Enter the calculation method you want to use: 1. Jaccard  2. TF-IDf Matching  3. Cosine Similarity
2
Enter the type code for Tf calulation: 1. Raw Tf  2. Normalised TF  3. Log based TF 4. Double Norm
2
Query token list ['countri', 'mous', 'town', 'mous']
--------------Statistics of the result are -----------
Method for Ranking :  2 and method for TF 2
1. Number of dictionary items :  176
2. Highest Score :  0.506120541500832
3. Lowest score :  5.4731912703626213e-05
4. Number of records having score >0 : 176
5. % of such records :  0.37687366167023556
Top 10 documents are:
cmoutmou.txt
mouslion.txt
ccm.txt
rid.txt
quarter.c7
advtthum.txt
lure.txt
aesopa10.txt
kneeslapper
pphamlin.txt
```

Log based TF score



```
Enter your query: THE COUNTRY MOUSE AND THE TOWN MOUSE
Enter number of records you want me to fetch10
Enter the calculation method you want to use: 1. Jaccard  2. TF-IDf Matching  3. Cosine Similarity
2
Enter the type code for Tf calulation: 1. Raw Tf  2. Normalised TF  3. Log based TF 4. Double Norm
3
Query token list ['countri', 'mous', 'town', 'mous']
--------------Statistics of the result are -----------
Method for Ranking :  2 and method for TF 3
1. Number of dictionary items :  176
2. Highest Score :  37.073826323952964
3. Lowest score :  1.3588375148868512
4. Number of records having score >0 : 176
5. % of such records :  0.37687366167023556
Top 10 documents are:
aesop11.txt
aesopa10.txt
cmoutmou.txt
gulliver.txt
lure.txt
mouslion.txt
ccm.txt
cybersla.txt
radar_ra.txt
hitch3.txt
```

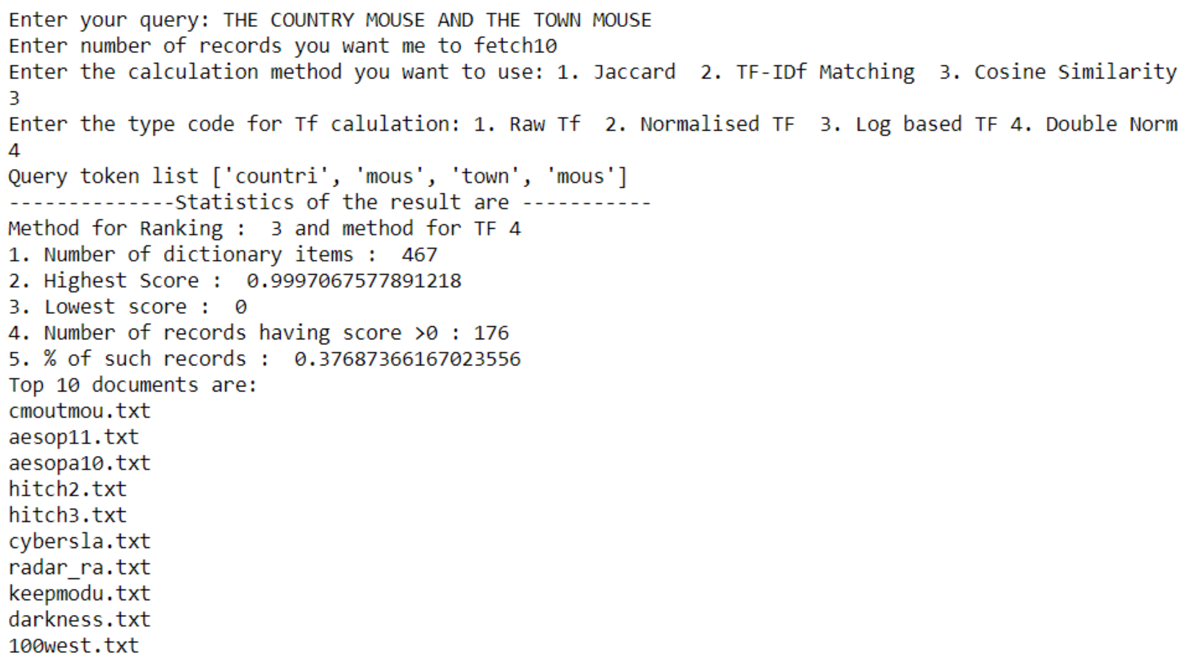
Double Norm TF score

```
Enter your query: THE COUNTRY MOUSE AND THE TOWN MOUSE
Enter number of records you want me to fetch10
Enter the calculation method you want to use: 1. Jaccard  2. TF-IDf Matching  3. Cosine Similarity
2
Enter the type code for Tf calulation: 1. Raw Tf  2. Normalised TF  3. Log based TF 4. Double Norm
4
Query token list ['countri', 'mous', 'town', 'mous']
--------------Statistics of the result are -----------
Method for Ranking :  2 and method for TF 4
1. Number of dictionary items :  176
2. Highest Score :  7.434716331108538
3. Lowest score :  0.6820218944451246
4. Number of records having score >0 : 176
5. % of such records :  0.37687366167023556
Top 10 documents are:
cmoutmou.txt
ccm.txt
mouslion.txt
aesopa10.txt
aesop11.txt
gulliver.txt
100west.txt
darkness.txt
keepmodu.txt
radar_ra.txt
```

Numerical Query with Double Norm TF score

```
Enter your query: i am only 17
Enter number of records you want me to fetch10
Enter the calculation method you want to use: 1. Jaccard  2. TF-IDf Matching  3. Cosine Similarity
2
Enter the type code for Tf calulation: 1. Raw Tf  2. Normalised TF  3. Log based TF 4. Double Norm
4
Query token list ['seventeen']
--------------Statistics of the result are -----------
Method for Ranking :  2 and method for TF 4
1. Number of dictionary items :  56
2. Highest Score :  1.4846842968536234
3. Lowest score :  1.062737964343792
4. Number of records having score >0 : 56
5. % of such records :  0.11991434689507495
Top 10 documents are:
imonly17.txt
poem-2.txt
luf
fic2
gatherng.txt
withdraw.cyb
keepmodu.txt
17.lws
enc
assorted.txt
```

Cosine Similarity with Double Norm TF score



```
Enter your query: THE COUNTRY MOUSE AND THE TOWN MOUSE
Enter number of records you want me to fetch10
Enter the calculation method you want to use: 1. Jaccard  2. TF-IDf Matching  3. Cosine Similarity
3
Enter the type code for Tf calulation: 1. Raw Tf  2. Normalised TF  3. Log based TF 4. Double Norm
4
Query token list ['countri', 'mous', 'town', 'mous']
--------------Statistics of the result are -----------
Method for Ranking :  3 and method for TF 4
1. Number of dictionary items :  467
2. Highest Score :  0.9997067577891218
3. Lowest score :  0
4. Number of records having score >0 : 176
5. % of such records :  0.37687366167023556
Top 10 documents are:
cmoutmou.txt
aesop11.txt
aesopa10.txt
hitch2.txt
hitch3.txt
cybersla.txt
radar_ra.txt
keepmodu.txt
darkness.txt
100west.txt
```

Analysis:

Query used in all the above examples was actually taken as ‘title‘of file named ‘cmoutmou.txt’. When ranking was done on the base of Jaccard score, this file was on the 3rd place even though the query is title only.

During TF-IDF matching score, 4 variants of TF score are used and the results were amazing. While raw TF score and log based TF score ranked this article on 4th and 3rd place but normalised and Double Norm based TF score outperformed the others by placing the text file on the 1st place which is good.

Cosine Similarity using Double Norm was applied and again the results were same. Text file was on the 1st place of top 10 retried relevant documents.

Raw TF:

While the advantage is to the pretty simple metric but it hamper the results when a word appears much larger times in a document in comparison to others.

Normalised TF:

It mitigates the above problem and provide some sort of proportional importance all the words. But it depends upon the length of the document.

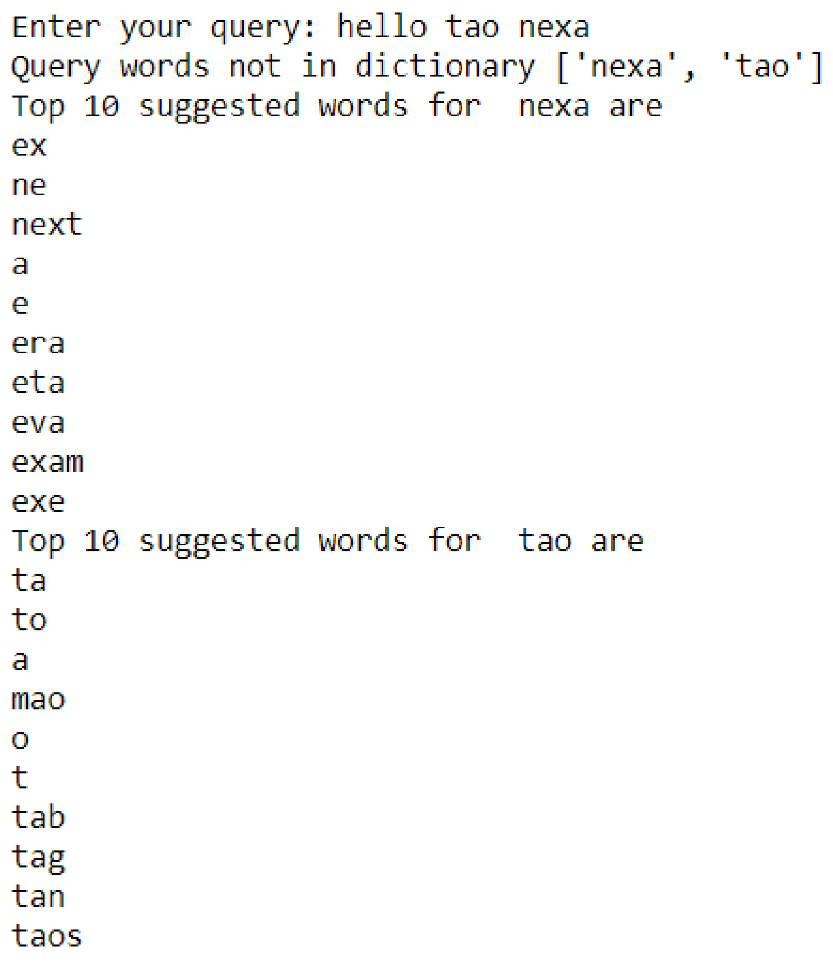
Log based TF:

It does smoothing and take care of the proportion of the words in the documents. Doc size is not considered and bias can be there, is one disadvantage

Double Norm TF score:

Advantage of this technique is to mitigate the anomaly of large TF in larger documents as they tend to repeat same words. But on the other hand, this method is unstable if the stop words list is modified. Also, it can give weightage to outlier of the data having large TF.

Question 2: Analysis and Output



```
Enter your query: hello tao nexa
Query words not in dictionary ['nexa', 'tao']
Top 10 suggested words for  nexa are
ex
ne
next
a
e
era
eta
eva
exam
exe
Top 10 suggested words for  tao are
ta
to
a
mao
o
t
tab
tag
tan
taos
```

Analysis:

Complexity for finding the edit distance of two string by the above method is O(m*n) where m, n are length of the strings. If closely notices, for ‘tao’ and ‘nexa’ the top 2 words already contain the some of the characters. Thus, edit distance cost is low and they are the preferred choice.