**Intelligent Information Retrieval**   
**CSC 575**

**Assignment 3  
Due: Thursday, February 2**5

**Rong Zhuang**

1. **Query Expansion Using Automatic Global Analysis**

For this problem you will use the document-term matrix given in problem 3 of Assignment 2. It is provided here as an [**Excel spreadsheet**](http://facweb.cs.depaul.edu/mobasher/classes/csc575/assignments/HW3.xlsx).

* 1. Create a term-by-term association matrix based corresponding to this document-term matrix (see Lecture: "[**Query Operations; Relevance Feedback; Personalization**](http://facweb.cs.depaul.edu/mobasher/classes/csc575/Lectures/Relevance-Feedback.pptx)" -- **Slide 6**-- on the [**Class Material**](http://facweb.cs.depaul.edu/mobasher/classes/csc575/lecture.html) page). Hint: You can facilitate this computation by using matrix operations in Excel (see:[**Using Microsoft Excel 2007 to Perform Matrix Operations**](http://facweb.cs.depaul.edu/mobasher/classes/csc575/assignments/MatrixOperations-Excel2007.pdf)). An intermediate term-term association matrix can be created by multiplying the transpose of the above document-term matrix (which is a term-document matrix) by the document-term matrix, itself. This will result in a symmetric term-term matrix where each entry corresponds to the dot product of the corresponding terms. The final association matrix can be computed by normalizing the entries as described in the lecture material. For your convenience, the transpose of the above matrix is already provided in the [**Excel document**](http://facweb.cs.depaul.edu/mobasher/classes/csc575/assignments/HW3.xlsx).

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** |
| **A** | 47 | 40 | 25 | 21 | 32 | 8 | 32 | 38 |
| **B** | 40 | 143 | 27 | 41 | 34 | 43 | 116 | 64 |
| **C** | 25 | 27 | 54 | 33 | 25 | 16 | 28 | 31 |
| **D** | 21 | 41 | 33 | 43 | 27 | 0 | 28 | 35 |
| **E** | 32 | 34 | 25 | 27 | 45 | 0 | 24 | 41 |
| **F** | 8 | 43 | 16 | 0 | 0 | 77 | 60 | 30 |
| **G** | 32 | 116 | 28 | 28 | 24 | 60 | 112 | 56 |
| **H** | 38 | 64 | 31 | 35 | 41 | 30 | 56 | 61 |
|  | **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** |
| **A** | 1 | 0.2667 | 0.3289 | 0.3043 | 0.5333 | 0.069 | 0.252 | 0.5429 |
| **B** |  | 1 | 0.1588 | 0.2828 | 0.2208 | 0.2429 | 0.8345 | 0.4571 |
| **C** |  |  | 1 | 0.5156 | 0.3378 | 0.1391 | 0.2029 | 0.369 |
| **D** |  |  |  | 1 | 0.4426 | 0 | 0.2205 | 0.5072 |
| **E** |  |  |  |  | 1 | 0 | 0.1805 | 0.6308 |
| **F** |  |  |  |  |  | 1 | 0.4651 | 0.2778 |
| **G** |  |  |  |  |  |  | 1 | 0.4786 |
| **H** |  |  |  |  |  |  |  | 1 |

*\*Refer to Similarities sheet in Matrix3.xlsx.*

* 1. Consider a query containing the terms A, C, F. What would be the new query after expansion based on global analysis with n=1 (i.e., for each query term, the top 1 associated term is added to the original query)? Explain your answer.

|  |  |
| --- | --- |
| A | Max(A) = c(AH) = 0.5429 |
| C | Max(C) = c(CD) = 0.5156 |
| F | Max(F) = c(FG) = 0.4651 |

H, D, G are the candidates. Since n = 1, just take the highest, add ‘H‘ to the new query.

1. **Relevance Feedback Using Rocchio's Method**  
   Suppose that after receiving the results of a query**Q0 = "dog race"**, a user has provided relevance feedback by rating the following 3 document as **non-relevant**:

**DOC1: "greyhound race track betting"  
DOC2: "dog race betting"**   
**DOC3: "greyhound dog training"**

and the following 4documents as **relevant**:

**DOC4: "iditarod dog sled race"  
DOC5: "husky dog sled race malamute dog sled"  
DOC6: "betting alaska dog sled race"  
DOC7: "dog race alaska iditarod"**

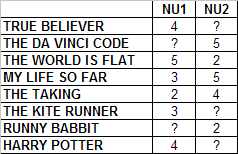
Assuming simple term frequency weights, use Rocchio’s relevance feedback method to compute a new query **Q1** (use a positive feedback factor of 1.0 and negative feedback factor of 0.5). Show Q1 as a vector over the above index terms with the corresponding weights generated by Rocchio. Explain any significant increase or decrease in term weights. Show your work.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **greyhound** | **race** | **track** | **betting** | **dog** | **training** | **iditarod** | **sled** | **husky** | **malamute** | **alaska** | **Rv** |
| **Q0** | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |  |
| **Doc1** | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | NR |
| **Doc2** | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | NR |
| **Doc3** | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | NR |
| **Doc4** | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | RE |
| **Doc5** | 0 | 1 | 0 | 0 | 2 | 0 | 0 | 2 | 1 | 1 | 0 | RE |
| **Doc6** | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | RE |
| **Doc7** | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | RE |
| **Q1** | -1 | 4 | -0.5 | 0 | 5 | -0.5 | 2 | 4 | 1 | 1 | 2 |  |
| **Q1** | 0 | 4 | 0 | 0 | 5 | 0 | 2 | 4 | 1 | 1 | 2 |  |

*\*Refer to Rocchio sheet in Matrix3.xlsx.*

* Q1 = Q0 + 1 \* SUM(Doc4:Doc7) – 0.5 \* SUM(Doc1:Do3), then normalize by converting negative values to zero, finally Q1 = (0,4,0,0,5,0,2,4,1,1,2)
* New query increased weigh for Iditarod, sled, husky, malamute and alaska. Especially for Iditarod, sled and alaska, they are significant enough in relevant documents.
* The old weight, race and dog are also significant increased, they appear in all relevant documents.

1. **Collaborative Filtering for Recommendation**  
     
   Suppose that an online bookseller has collaborative filtering recommender system. The bookseller has collected [**ratings information**](http://facweb.cs.depaul.edu/mobasher/classes/csc575/assignments/knn.xls) from 20 past users (U1-U20) on a selection of recent books. The ratings range from 1 = worst to 5 = best. Two new users (NU1 and NU2) who have recently visited the site and rated some of the books as follows ("?" represents missing ratings):



Using the *K-Nearest Neighbor* algorithm predict the ratings of these new users for each of the books they have not yet rated. Use the Pearson **correlation coefficient** as the similarity measure. For your convenience, this data is given in the Excel spreadsheet "[**knn.xls**](http://facweb.cs.depaul.edu/mobasher/classes/csc575/assignments/knn.xls)". **Note:** In Microsoft Excel, you can use the **CORREL** function to compute correlation.

* 1. First compute the correlations between the new users (**NU1** and **NU2**) and all other users (you can show these as added columns in original spreadsheet). Then for each new user give the predicted rating for each of the unrated items using ***K*=3** (i.e., 3 nearest neighbors). Use the [**weighted average function to compute the predictions**](http://facweb.cs.depaul.edu/mobasher/classes/csc575/assignments/knn-predictions.html) based on ratings of the nearest neighbors. Be sure to show the intermediate steps in your work (or provide a short explanation of how you computed the predictions).

|  |  |  |
| --- | --- | --- |
|  | **COR(NU1)** | **COR(NU2)** |
| **U1** | -0.866 | 0.945 |
| **U2** | -1.000 | 0.982 |
| **U3** | 0.029 | 0.945 |
| **U4** | -0.327 | -1.000 |
| **U5** | 0.866 | 0.756 |
| **U6** | 0.455 | 0.187 |
| **U7** | 0.157 | -0.246 |
| **U8** | -0.655 | 0.500 |
| **U9** | 0.849 | -0.786 |
| **U10** | -0.945 | 0.693 |
| **U11** | -0.218 | 1.000 |
| **U12** | 0.845 | 0.756 |
| **U13** | -0.500 | 0.500 |
| **U14** | -0.400 | -0.417 |
| **U15** | -0.982 | -1.000 |
| **U16** | 0.945 | -0.455 |
| **U17** | -0.512 | 0.693 |
| **U18** | 0.700 | -0.866 |
| **U19** | 0.500 | 0.866 |
| **U20** | -0.609 | 0.408 |

*\*Refer to Recommendation sheet in Matrix3.xlsx.*

The nearest 3 neighbors with NU1 is U16(0.945), U5(0.866) and U9(0.849).:

* r(NU1, I2) = (3\*0.945 + 4 \* 0.866) / (0.945 + 0.866) = 3.478
* r(NU1, I7) = (4\*0.945 + 1\*0.866) / (0.945 + 0.866) = 2.565

The nearest 3 neighbors of NU2 is U11(1.000), U2(0.982), U1(0.945). U1 and U3 has the same value, we take U1 for the sequential calculation.

* r(NU2, I1) = (2\*1+2\*0.945)/(1+0.945) = 2
* r(NU2, I6) = (1\*1 + 2\* 0.982) /(1 +0.982) = 1.495
* r(NU2, I8) = (2\*1 + 1 \*0.945) / (1+0.945) = 1.514
  1. Measure the **Mean Absolute Error** (MAE) on the predictions for NU1 and NU2. You can compute MAE by generating predictions for items already rated by the target user (e.g., for NU1 these are all items except "**The DaVinci Code**" and "**Runny Babbit**"). Then, for each of these items you can compute the absolute value of the difference between the predicted and the actual ratings. Finally, you can average these errors across all compared items to obtain the MAE.

Prediction matrix.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **I1** | **I2** | **I3** | **I4** | **I5** | **I6** | **I7** | **I8** |
| **NU1** | 2.000 |  | 2.964 | 1.473 | 1.000 | 2.000 |  | 5.000 |
| **NU2** |  | 4.510 | 2.000 | 3.514 | 3.000 |  | 1.000 |  |

Actual matrix.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **I1** | **I2** | **I3** | **I4** | **I5** | **I6** | **I7** | **I8** |
| **NU1** | 4 |  | 5 | 3 | 2 | 3 |  | 4 |
| **NU2** |  | 5 | 2 | 5 | 4 |  | 2 |  |

* MAE(NU1) = (|4-2| + |5-2.964| + |3-1.473| + |2-1| + |3-2| + |4-5|) / 6 = 1.427
* MAE(NU2) = (|5-4.510| + |2-2| + |5-3.514| + |4-3| + |2-1|) / 5 = 0.795
  1. Consider the following simple "popularity-based" recommendation algorithm: Given a user **U** and an item**I**, compute the predicted rating of **U** on **I** as the mean rating for**I**among all users who have rated **I**. Using this algorithm instead of KNN re-compute the MAE on the predictions for NU1 and NU2 (as in part b). Which one of the algorithms (KNN or this algorithm) perform better? Briefly explain what you think are the pros and cons of each of these two recommendation approaches.

Popularity Rating for item I = sum of all rating value / amount of users who has rated. In excel, each cell has the formula like this: SUM(B2:B21)/COUNTA(B2:B21).

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **I1** | **I2** | **I3** | **I4** | **I5** | **I6** | **I7** | **I8** |
| **New User** | 2.750 | 3.667 | 2.333 | 2.500 | 3.000 | 2.545 | 2.417 | 3.563 |

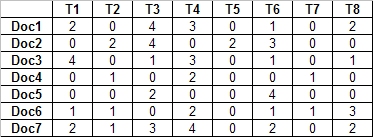
* MAE(NU1) = (|4-2.75| + |5-2.333| + |3-2.5| + |2-3| + |3-2.545| + |4-3.563|) / 6 = 1.051
* MAE(NU2) = (|5-3.667| + |2-2.333| + |5-2.5| + |4-3| + |2-2.417|) / 5 = 1.117

For NU1, MAE is smaller, which means popularity-based algorithm performs better than KNN. For NU2, MAE is bigger, which means KNN performs better.

Pros and cons:

* Since popularity-based algorithm utilize most of the users, the result is somehow more ‘average’, you can see the gap between NU1 and NU2 is little.
* Popularity-based algorithm is more accurate when the user has the similar taste with most of people.
* In contrast, KNN is more accurate when the user has the similar taste with only a few ‘neighbor’ people. The more ratings his/her neighbors submit, the more accurate prediction can be retrieved.

1. **Document Categorization**  
     
   Consider the following document-term matrix:



Assume that documents have been manually assigned to two pre-specified categories as follows:

**Cat1 = {Doc1, Doc2, Doc5}   Cat2 = {Doc3, Doc4, Doc6, Doc7}**

* 1. Using the **K-Nearest-Neighbor approach for document categorization with K = 3** (see notes on[**Document Categorization**](http://facweb.cs.depaul.edu/mobasher/classes/csc575/lectures/Categorization.pptx)),  determine how each of the**new documents** given below will be classified. Show your steps. **Note:**Use **non-normalized tf\*idf** weights and **cosine similarity** when computing similarities.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **T1** | **T2** | **T3** | **T4** | **T5** | **T6** | **T7** | **T8** |
| **Doc8** | 3 | 1 | 0 | 4 | 1 | 0 | 2 | 1 |
| **Doc9** | 0 | 0 | 3 | 0 | 1 | 5 | 0 | 1 |

|  |  |  |
| --- | --- | --- |
|  | **Doc8** | **Doc9** |
| **Doc1** | 0.435 | 0.414 |
| **Doc2** | 0.514 | 0.878 |
| **Doc3** | 0.580 | 0.181 |
| **Doc4** | 0.704 | 0.000 |
| **Doc5** | 0.000 | 0.703 |
| **Doc6** | 0.650 | 0.211 |
| **Doc7** | 0.510 | 0.405 |

*\*Refer to Categorization sheet in Matrix3.xlsx.*

* The top 3 similar with Doc8 is Doc4, Doc6 and Doc3. So, Doc8 has Cat2.
* The top 3 similar with Doc9 is Doc2, Doc5 and Doc1. So, Doc9 has Cat1.
  1. Repeat the classification problem above, but this time use the **Rocchio-Based** vector space model to determine how **Doc 8 and Doc 9**given above will be classified. As before, **use non-normalized tf\*idf weights and cosine similarity when computing similarities**.

Prototype vectors for Cat1 and Cat2:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **T1** | **T2** | **T3** | **T4** | **T5** | **T6** | **T7** | **T8** |
| **P(cat1)** | 2 | 2 | 10 | 3 | 2 | 8 | 0 | 2 |
| **P(cat2)** | 7 | 3 | 4 | 11 | 0 | 4 | 2 | 6 |

*\*Refer to Categorization sheet in Matrix3.xlsx.*

Similarities for Doc8 and Doc9:

|  |  |  |
| --- | --- | --- |
|  | **Cat1** | **Cat2** |
| **Doc8** | 0.275143 | 0.729728 |
| **Doc9** | 0.657959 | 0.313796 |

*\*Refer to Categorization sheet in Matrix3.xlsx.*

Large value is better, Doc8 should belong to Cat2, and Doc9 should belong to Cat1.