

Question 1

Interpolated means that for each recall value from 0.0, 0.1, 0.2 ... to 1.0 (11 values) find the maximum precision at Recall Precision table where the recall value is greater than or equal to recall level.

For instance for recall level of 0.2, we need to find the maximum precision where recall is greater than or equal to 0.2 in original recall precision table.

Part A

For Query 1

Total number of relevant documents for query 1 is 15.

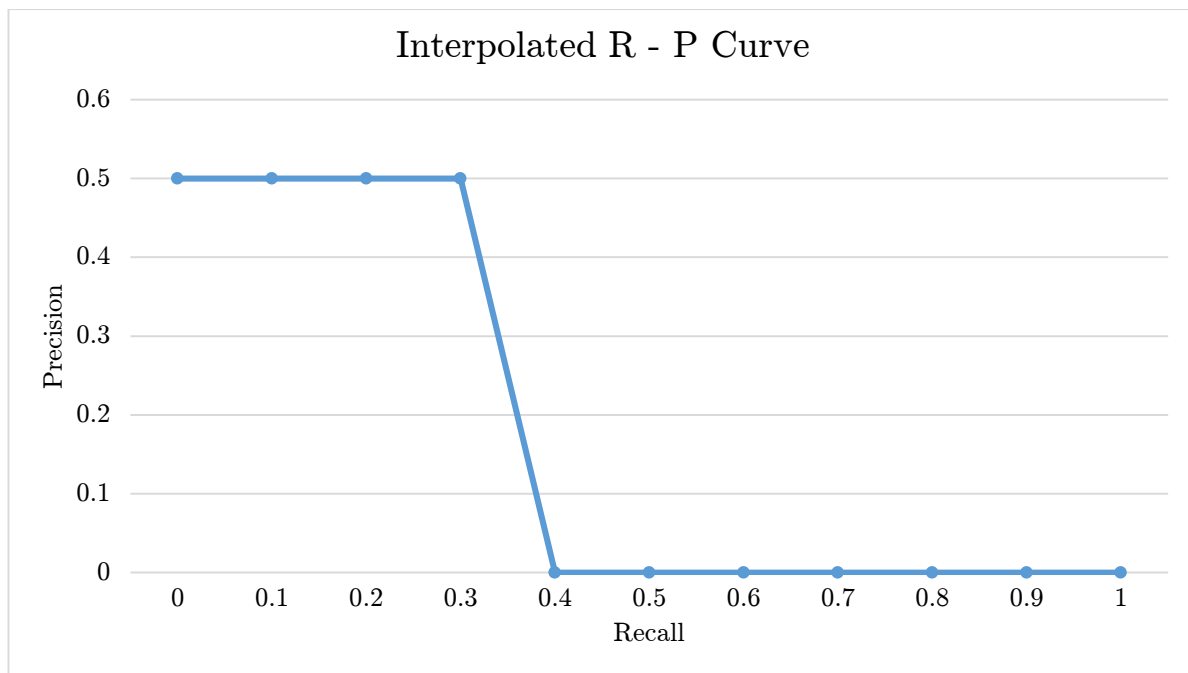
Recall – Precision Table :

Rank	1	2	3	4	5	6	7	8	9	10
Relevance	0	1	0	1	0	1	0	0	0	0
Precision	0	1/2	1/3	2/4	2/5	3/6	3/7	3/8	3/9	3/10
Recall	0	1/15	1/15	2/15	2/15	3/15	3/15	3/15	3/15	3/15

Interpolated Recall – Precision Table :

Precision	1/2	2/4	3/6	3/6	0	0	0	0	0	0	0
Recall	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0

Interpolated Recall – Precision Graph :



For Query 2

Total number of relevant documents for query 2 is 4.

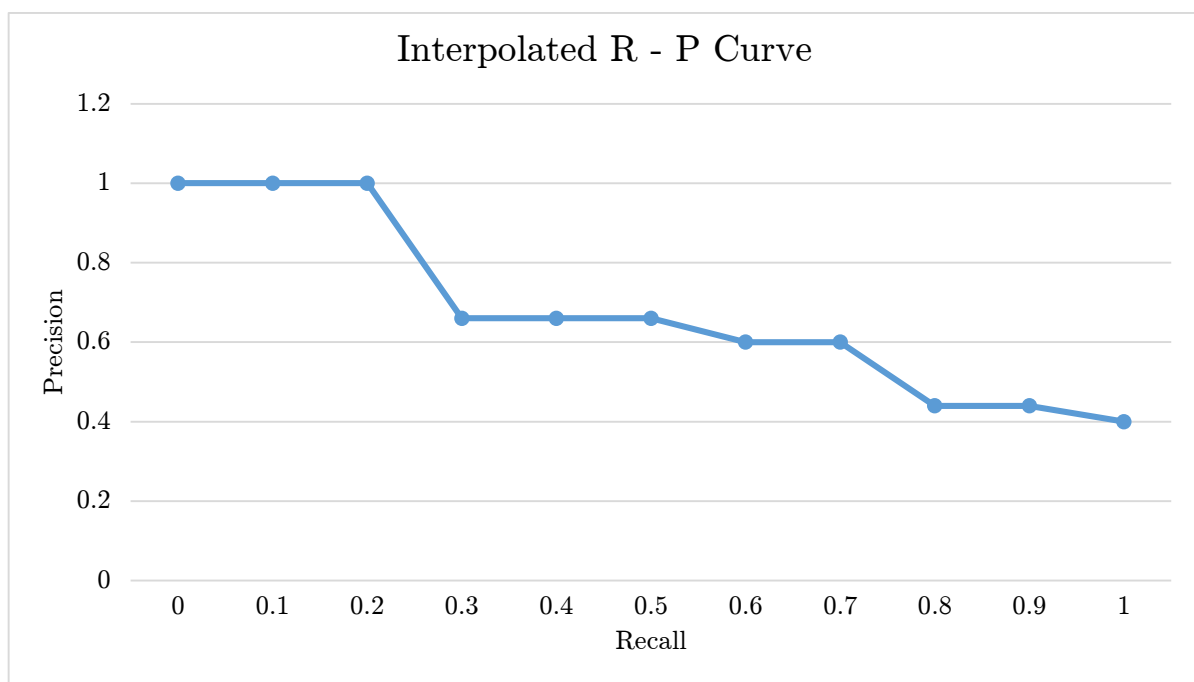
Recall – Precision Table :

Rank	1	2	3	4	5	6	7	8	9	10
Relevance	1	0	1	0	1	0	0	0	1	0
Precision	1	1/2	2/3	2/4	3/5	3/6	3/7	3/8	4/9	4/10
Recall	1/4	1/4	2/4	2/4	3/4	3/4	3/4	3/4	4/4	4/4

Interpolated Recall – Precision Table :

Precision	1	1	1	2/3	2/3	2/3	3/5	3/5	4/9	4/9	4/10
Recall	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0

Interpolated Recall – Precision Graph :

**For Query 1 And 2 Avarage**

Interpolated Recall – Precision Table For Average :

Precision	3/4	3/4	3/4	7/12	2/6	2/6	3/10	3/10	4/18	4/18	4/10
Recall	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0

Part B

R-Precision is the precision after R documents retrieved where R is the number of relevant documents.

For query 1 there are 15 relevant documents but we retrieved only 3 of them. To calculate actual R-Precision we need retrieve all of the relevant documents. However, for 10 document collection we have 3 documents retrieved, so the R-Precision would be **3/10**.

For query 2 there are 4 relevant documents. The precision at 4th point is **2/4** which is R-Precision.

Part C

MAP is equal to average of the average precision values for set of queries. For relevant documents that are not retrieved in query 1, we can use precision value of 0 in calculation.

For query 1 the average precision is : $(1/2 + 2/4 + 3/6) / 15 = 0.10$

For query 2 the average precision is : $(1 + 2/3 + 3/5 + 4/9) / 4 = 0.67$

MAP : $(0.10 + 0.67) / 2 = \mathbf{0.385}$

Question 2

This D matrix can be partitioned using either document based partitioning or term based partitioning. In document based approach, as an example 6 documents can be divided into 2 – 2- 2 document partition and each 2 document can be hosted on a different machine. For each of these machines, local index is built and queries are sent to every 3 machine. Local results are then combined for final result.

In term based approach, as an example 6 terms can be divided into 2 – 2 – 2 term partition. So we would need 3 machines to host these partitions. Every machine would be responsible for 2 term. Unlike document based approach, queries would not be sent to every machine, since a machine is responsible for subset of terms. This would result in fewer disk seeks and transfer operations. On the other hand, each transfer operation involves more data. In other words, load might fall to single machine and it can become a bottleneck.

Most of the time document based partitioning would result in better balance and workload due to following reasons.

- Since the work is distributed to several machines instead of single one, load will be distributed and this will reduce to chance of creating bottlenecks. This will also provide better throughput.
- When new documents are added, in term based partitioning we might need to reorganize indexes on every machine, since a machine might keep information about any document. However, in document based approach, only single machine would need to create or reorganize its index (assuming no new terms are added).
- When single machine is offline, in term based approach, we might not answer the query at all. However, in document based approach, we can still show some results to the user.

Due to above points, most of the time using document based partitioning would be better. However, if we know that all terms will be searched almost equally and load would not be a problem, and if we want to reduce network costs and disk I/O access, term based approach can be applied for distribution.

Question 3

Part A

Without Skipping

Both of the posting lists are sorted according to document number. Since they are sorted, when we found the location of an element in term-b list, we do not need to repeat these comparisons for rest of the elements in term-b list. In other words, single scan of term-a posting list would be enough.

For <12, 3>, we need to make **5 comparisons** with <1, 5> <4, 1> <9, 3> <10, 4> <12, 4>

For <40, 2>, we need to make **7 comparisons** with <17, 4> <18, 3> <22, 2> <24, 4> <33, 4> <38, 5> <43, 5>

For <66, 1>, we need to make **3 comparisons** with <55, 3> <64, 2> <68, 4>

In total we need **15 comparisons**.

With Skipping Introduced

Skipping with chunk size 4 can be applied. Total of 5 chunks for term a will be created.

Chunk 1 : <1, 5> <4, 1> <9, 3> <10, 4>	Descriptor : <10, 4>
Chunk 2 : <12, 4> <17, 4> <18, 3> <22, 2>	Descriptor : <22, 2>
Chunk 3 : <24, 4> <33, 4> <38, 5> <43, 5>	Descriptor : <43, 5>
Chunk 4 : <55, 3> <64, 2> <68, 4> <72, 5>	Descriptor : <72, 5>
Chunk 5 : <75, 1> <88, 2>	Descriptor : <88, 2>

Merge Part for <12, 3> - 3 Comparisons

- Compare it with descriptor of Chunk 1. Go to Chunk 2.
- Compare it with descriptor of Chunk 2. Chunk found.
- Compare it with first element of Chunk 2. Position is found.

Merge Part for <40, 2> - 5 Comparisons

- Compare it with descriptor of Chunk 2. Go to Chunk 3.
- Compare it with descriptor of Chunk 3. Chunk found.
- Compare it with 1st, 2nd and 3rd elements of Chunk 3. Position is found.

Merge Part for <66, 1> - 5 Comparisons

- Compare it with descriptor of Chunk 3. Go to Chunk 4.
- Compare it with descriptor of Chunk 4. Chunk found.
- Compare it with 1st, 2nd and 3rd elements of Chunk 4. Position is found.

When skipping with chunk size of 4 is applied, total of **13 comparisons** is needed.

Part B

Posting list of term-a ordered by frequency of term t in document d

<1, 5> <38, 5> <43, 5> <72, 5> <10, 4> <12, 4> <17, 4> <24, 4> <33, 4> <68, 4> <9, 3> <18, 3> <55, 3> <22, 2> <64, 2> <88, 2> <4, 1> <75, 1>

Posting list of term-a by frequency information in prefix form

<5 : 4 : 1, 38, 43, 72> <4 : 6 : 10, 12, 17, 24, 33, 68> <3 : 3 : 9, 18, 55>
<2 : 3 : 22, 64, 88> <1 : 2 : 4, 75>

The second form does not store the frequencies repeatedly and it would save some memory. This might be crucial when the posting list is long there are many terms. Thus, I would prefer the second approach.

Question 4

Cranfield approach is a system oriented approach to information retrieval evaluation which uses test collections. These test collections include set of documents, queries and relevant documents from query. These tests are reusable and their purpose is to evaluate information retrieval systems. In Cranfield approach purpose is to create a laboratory like controlled environment to test IR systems.

Question 5

The purpose in pooling method is to collect unbiased sample of the relevant documents in a large test collection. This is done by gathering top p results from different IR systems under test for each topic and combining these results into a single list. (p is the pool depth) Assumption here is that results list for different IR systems are diverse. Therefore, it will bring relevant documents into pool. However, according to some researchers, pooling can create biased samples in some degree.

The particular problem with pooling is that it is impossible to be sure about whether all relevant documents are located. Also as the new IR systems added the limited pool size might lead to biased samples.

First problem is when there are documents with similar idea which are retrieved by two IR system help each other to outweigh the importance of other document retrieved by third IR system. Second problem might occur when there are many answers to queries and due to size of pool only limited number of them can be added. However, these are investigated by Zobel [1] and he stated that assuming pool size is deep enough, the pooling usually leads to reliable results.

In pooling approach documents are that are judged are split into two groups as relevant or not relevant. However, there are also unjudged documents which are not in the pool. Researchers choose to ignore these unjudged documents [2] or they classified unjudged documents as not relevant [3]. However, Buckley and Voorhees [4] believed that this may cause problems. The problem according to them is the size of the pool relative to size of collections was reducing when the test collections grew. As a result, new evaluation measure is considered by them to use when large number of unjudged documents exist. This method is called **bpref**.

$$BPref = \frac{1}{R} \sum_r \left(1 - \frac{|n \text{ ranked higher than } r|}{\min(R, N)} \right),$$

Where R is the number of documents judged relevant for particular topic, N is the number of documents judged and not relevant, r is the number of retrieved documents that are relevant and n is a member of the first R retrieved irrelevant documents.

Question 6

Part A

In traditional IR systems, we assume there is a finite amount of data. This is not the case in streaming systems. In traditional IR systems, we can store and analyze data in multiple steps. This is not the case in streaming systems because,

- i. Data objects arrive continuously,
- ii. There is no control over the order of data,
- iii. Stream size is potentially unbounded, so we cannot keep all of it even on secondary disks.
- iv. Unlike traditional systems, when data is processed it is discarded,

Although there are differences in the way data generated and stored in traditional systems and streaming systems, the knowledge discovery techniques still need to be applied on these streaming data.

Part B

In streams unlike regular datasets there is no fixed amount of data that we can run queries on. The data continuously arrives in streaming fashion and usually most up to date objects in data is relevant. In other words, up to date objects are more important in streaming systems. Also we need a fixed data to run operations on it. This is why time windows are needed. We can keep data limited via time window and run clustering or any other operation on this data.

Time windows are useful, because it keeps the most recent data and limits the streaming data.

Part C

In Silva's paper, by **abstraction** he means the summary of streaming data. He uses this concept in step called data abstraction just before actual clustering. Space and memory is limited and data size is usually large in streaming systems. Instead of actually storing the objects in stream, Silva suggests that we can use **data abstraction** step to summarize the data stream with the help of particular data structures for dealing with space and memory constraints. For instance, prototype arrays, data grids, vectors can be used in this step while keeping the meaning of original objects without actually storing the whole object.

Part D

Some stream clustering algorithms use prototype array. The purpose of this array is to summarize the data partition using medoids or centroids. (Prototypes). In this algorithm.

- i. Data stream is divided into chunks of size m .
- ii. Each chunk of m objects is summarized into $2k$ representative objects by using variants of k -medoids algorithm
- iii. Above steps (compression of data) are repeated until there are m number of prototypes in array.
- iv. Then this m prototypes are further decompressed into $2k$ prototypes and this process is used for whole stream.

Part E

The main idea in clustering data streams is using CF vectors. To make use of CF vectors in k -means algorithm, variants of k -means that are suitable to data streams are proposed such as extended k -means and k -means++.

For CF vectors case, the k -means algorithm can be modified in one of the three ways below.

- i. Calculate the centroid for each CF vector and consider each centroid as object to be clustered.
- ii. Do same with first one but this time weight each object proportional to N so that CF vectors with more objects will have a higher influence on the centroid calculation process performed by k -means.
- iii. Apply clustering directly to CF vectors, as they keep enough statistics to calculate most of the required distances and equality metrics.

Question 8

Part A

If the values in dataset can be assigned to a label, this dataset is called nominal. In other words, if the dataset is categorically discrete then it is nominal. For instance, in a dataset male people can be coded as M and females can be coded as F. Another example can be following. In a patient tumor dataset, the tumors can be marked as B for benign or M for malignant

Part B


In D matrix, documents are marked as relevant or not relevant according to some term. So we can safely say that this data that can be categorized. Categorized data can be divided into two subsets, nominal and ordinal. Difference between them is ordinal data can be ordered whereas nominal cannot. For instance, age is ordinal, race is nominal. In our case, we label documents as relevant or not relevant which cannot be ordered so we can say that document set used in D matrix is nominal.

Part C

Nominal data is a categorical data and they are different in continuous numerical data in some ways. Since there is no Euclidean distance among nominal data, distance functions cannot be applied to nominal data set [5]. Therefore, we need to either convert our nominal data to numerical data or tweak our existing cluster algorithms for nominal data.

For instance, nominal data such as gender can be converted to numerical form as M = 1 and F = 0. Or grades of students can be converted to numerical data like A = 4.0 B = 3.0 and so on. For nominal dataset with few values, flag can be created for each attribute and they can be assigned to either 0 or 1. [6] For instance for colors,

ID	Color	...
371	red	
433	yellow	



ID	C_re	C_orange	C_yello	...
371	1	0	0	
433	0	0	1	

However, this could be problematic for nominal data with many attributes since it would require many bits to represent each attribute. For example Profession codes (7000 values). In such cases,

- i. ID like fields whose values are unique can be ignored,
- ii. For rest of them we can group them naturally. For example, for professions we can select most frequent ones and group the rest of them.

Another way to use nominal data in clustering algorithms is to discover or tweak existing algorithms to work with nominal data. For instance, for k-means algorithm to work with nominal data, it is extended and k-modes algorithm is created. In this algorithm, instead of using the means for average, simple matching dissimilarity measure for categorical objects is used. [7]

References

- [1] J. Zobel, "How reliable are the results of large-scale information retrieval experiments?," in Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 307–314, New York, NY, USA: ACM Press, 1998.
- [2] W. Hersch, C. Buckley, T. J. Leone, and D. Hickam, "OHSUMED: An interactive retrieval evaluation and new large test collection for research," in Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 192–201, New York, NY, USA: Springer-Verlag New York, Inc, 1994.
- [3] S. B. Tutter, C. L. A. Clarke, P. C. K. Yeung, and I. Soboroff, "Reliable information retrieval evaluation with incomplete and biased judgements," in Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 63–70, New York, NY, USA: ACM Press, 2007.
- [4] C. Buckley and E. M. Voorhees, "Retrieval evaluation with incomplete information," in Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 25–32, New York, NY, USA: ACM, 2004.
- [5] Wang and Piyang, "Clustering and classification techniques for nominal data application," City University of Hong Kong 2008 <http://hdl.handle.net/2031/5400>
- [6] Nguyen Hung Son, "Data cleaning and data preprocessing" <http://www.mimuw.edu.pl/~son/datamining/DM/4-preprocess.pdf>
- [7] Zengyou He, "Approximation Algorithms for K-modes Clustering," Harbin Institute of Technology <http://arxiv.org/ftp/cs/papers/0603/0603120.pdf>