Table

Description automatically generated with medium confidence

P1:

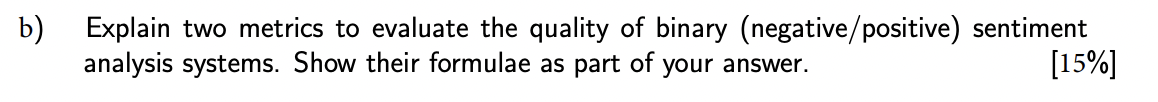
Overall precision System 1 = TP/TP+FP = 5/10; Overall precision System2 = 7/10.

P2:

Precision at cutoff point 3, System 1 = 2/3; Precision at cutoff point 3, System 2 = 1/3. Precision at cutoff point 5, System 1 = 4/5; Precision at cutoff point 5, System 2 = 3/5.

P3:

The difference between overall precision and precision at cutoff points is that the former involves the entire list of documents judged by the IR systems. The latter only considers the top x documents, and therefore is better at evaluating whether the rank of the most relevant documents is correct. In this case, System 2 is considered better according to overall precision, but if one looks at the top-ranked answers (both top 3 and top 5), one sees that System 1 is actually better at ranking relevant documents correctly.



Comparing the system output to human classifications and compute metrics like (any 2 answers, 10% each):

Accuracy = number of correctly classified segments / number of segments, where correctly classified segments are those where the system agrees with the human decision (i.e., both positive or both negative)  
Precision Positive = number of segments correctly classified as positive / number of segments classified as positive (idem for Precision Negative)

Recall Positive = number of segments correctly classified as positive / number of positive segments (idem for Recall Negative)  
F-measure Positive = (2 \* Precision Positive \* Recall Positive) / Precision Positive + Recall Positive (idem for F-measure Negative)

Text, letter

Description automatically generated

[P1](5%)

Document 1: dataset corrupt corrupt data hash  
Document 2: data system transfer corrupt data file trash

Document 3: politician corrupt developing country

[P2](10%) A stoplist is a list of words (‘stop-words’) that should be ignored when documents are indexed. When the key information of documents is needed to be extracted or similarity of some documents is needed to be compared, stoplist is helpful to ignore the words which are helpless to show the difference between documents because they are too common and used everywhere from the collection.

Capitalisation is the process that can be used to normal all words to lower (or upper) case so that same words which include letters with different case (upper and lower) but usually have similar means can be thought as the same, for example, Corrupted = corrupted.

Lemmatisation is the process to change words to their basic and common form from different forms (tense, singular), e.g. variants corruption, corrupted, corrupts, etc. are thought as the same as corrupt.

[P3](5%) Stoplist: {your, is, are, in, will, to, most, many, not, !, .}

Text

Description automatically generated

Boolean search is a method to retrieve and post a binary result about whether the document is relevant or not. The method usually involves combinations between logical operators (such as AND, OR, NOT, etc.) and terms. The advantage is that the method allows formulations and provides precise results, which may also benefit from specialised manual indexation of documents. However, if the result is abundant, satisfying result will not be shown on the top because of lack of ranking. Also, the Boolean search need precise input (formulation with proper term and operation), or the result will be abundant or not that precise.

Ranked search is a method to retrieve and compute the level of similarity between queries or documents and will return a ranked list which includes information of weight and frequency of words. The advantages of the method involves that users do not need to input precise formulations, so the results are not precise but query formulation does not require particular expertise knowledge. And, the ranking of results especially results on the top of the list is helpful, e.g. in web search.

Text

Description automatically generated

P1: An inverted index is a data structure that shows each word of documents with frequency of occurrence of word in particular document. An inverted index helps to make a comparation for words between documents and query. Also, it is easy for users to pick one word from query, look for whether the document needed is in inverted index adhere to the particular word.A more sophisticated version of an inverted index might also record position information for occurrences, to allow for efficient retrieval of phrases.

P2:

Table

Description automatically generated

Text

Description automatically generated

Text, letter

Description automatically generated

Table

Description automatically generated with medium confidence

Text

Description automatically generated

[P1](10%)

If an experiment is processed by an information retriever, assuming that: A is a collection of documents that retrieved for a particular query; B is a collection of relevant documents with the query; AND is the logical operator to show the common elements between two groups. Therefore, precision can be defined as (A AND B)/A and recall can be defined as (A AND B)/B.

[P2](5%)

Graph A is not possible. In the graph, when precision is 1, recall is one, which means the collection of documents retrieved is the same as that relevant with the query. So, the curve would though both (1,1) and (0,0). However, it is not possible for the curve to stay away from this point of (0,0) which is shown in the graph.

Graphical user interface, text, application, email

Description automatically generated

Term manipulations can include any of the following (answer only need 3): capital- isation, stemming, stop-word removal, indexing multi-terms, normalisation. Three exemplar definitions:

[P1](6%) A stoplist is a list of words (‘stop-words’) that are ignored when docu- ments are indexed, and likewise discounted from queries. These are words that are so widespread in the document collection that they are of v.little use for discriminating between documents that are/are not relevant to a query. Their exclusion eliminates a large number of term occurrences that would need to be recorded, thereby reducing the size of indexes and saving computational effort during both indexing and retrieval.

[P1](8%) Stemming refers to the process of reducing words that are morphological variants to their common root or stem, e.g. so that variants computer, computes, computed, computing, etc. are reduced to a stem such as compute. For IR, stemming is applied to documents before indexing and to queries. The effect of this is that when a query contains a term such as computing, retrieval can potentially return documents on the basis of their containing any of the morphological variants of the same root. This is the intended key benefit of using stemming. Stemming will also produce some reduction in the size of document indexes.

[P1](6%) Capitalisation refers to the process of normalising the case of words so that a single case is used, for example, all words are lowercased (e.g. SHELL = shell). This procedure makes indexing and retrieval more efficient by decreasing the number of terms that have to be represented. It also makes the term weighting more reliable, as higher frequency counts will be observed when putting together different variants of the term.

Text

Description automatically generated

[P1](15%) TF.IDF assigns weights to terms by taking into account two elements: the frequency of that term in a particular document (TF) and the proportion of documents in the corpus in which it occurs. The IDF for a term w is computed as follows, where D is the set of documents in the document collection and df w is the document frequency of w (the count of documents in which w appears):

Text

Description automatically generated with medium confidence

The TF.IDF value for a given term (w) in a given document (d) is:

Text

Description automatically generated

[P2](15%)

In this example, using TF.IDF would set the weight of three of the query terms to 0:

buy, sea, shell, since they appear in all (two) documents in the collection and thus

log|D| = log2 = log(1) = 0. The only query term that receives a weight different dfw 2

from 0 is lovely, which only happens in Document 1, and therefore this document is ranked first. TF.IDF has a positive effect in this example and in general, since it disregards terms that are frequent across the whole collection of indexed documents.

Graphical user interface, text

Description automatically generated

[P1](10%) The first step is to do a subjectivity analysis. This has to do with detecting whether the text (word, phrase, sentence, document) contains opinions, emotions, sentiment, or simply facts. Only subjective sentences can be present sentiment analysis, which is related to the actual polarity of the text (words, phrases, sentences, documents): the difference between positive, negative, etc.

[P2](10%) In this example, only the first sentence is subjective and therefore the first sentence can be analysed.



[P1](10%) A simple rule-based subjectivity classifier can be built: if a sentence or document is subjective and it has at least 2 words that belong to an emotion words lexicon; a sentence/document is objective otherwise.

[P2](10%) A Naive Bayes classifier can be built with features that count emotion words from a lexicon, as well as other features, such as adjectives, negation words, etc. Training data with instances with binary labels (subjective vs objetive) would be required.

Text

Description automatically generated

Text, letter

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text, letter

Description automatically generated