0/10 Questions Answered
9 questions with unsaved changes

Midterm 2

STUDENT NAME

Search students by name or email... ▾

Q1 Exam Instructions

0 Points

This exam is open book/note/internet, but you are expected to do the work on your own, without the help of other people (remote or local.) You are expected to complete the test in 50 minutes; it will cut you off after 60, but the extra 10 is for uploading handwritten answers; you are expected to complete your work in 50 minutes. Watch your time, and do not spend too long on any question. The times given are estimates - they are not enforced, but if you spend longer than the given time, you probably should go on and come back later.

You may ask questions in the office hours WebEx room during the hours it is staffed (hours sent via email): https://purdue.webex.com/purdue/j.php?MTID=m7433bffc48eafa61dcb8257f071b1790

Clarifications will be posted at: https://docs.google.com/document/d/1vouE22dytR87dADsZu-IgjEPVCHPymui8UG1M4lpvgg/edit?usp=sharing

Q1.1 Purdue Honor Code

0 Points

As a Boilermaker pursuing academic excellence, I pledge to be honest and true in all that I do. Accountable together - We are

Purdue.

○ I agree with the Purdue Honor Code.

○ I do not agree with the Purdue Honor Code.

Save Answer

Q1.2 Answer upload method

0 Points

◉ I will enter my answers in the text boxes and/or upload responses for each question. You must click "submit answer" before time expires. You can go back and change submitted answers (and resubmit) up until the time limit is reached, or you submit and view the completed exam.

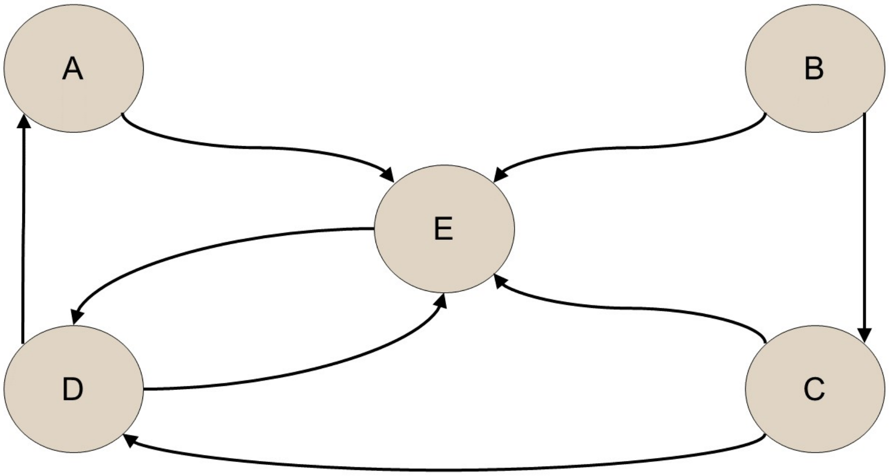
○ I will upload my entire answer set to Gradescope as a single PDF under "Midterm 2 (Paper only submission)"; answers submitted here will not be viewed or graded.

Save Answer ***Unsaved Changes**

Q2 Graph-structure Based Ranking (8 minutes)

10 Points

Given the following linked document collection:



1. Give the initial matrix **B** representing this graph (the graph you would start with to calculate PageRank.)

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1/2 & 1/2 \\ 1/2 & 0 & 0 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

2. There is a formula involving the **B** matrix that holds once we know the PageRank for each page. Show this formula (you don't actually need to calculate PageRank).

$p = B^T p$ or $r = \alpha B^T r$ (where r is the eigenvector and α is the eigenvalue of the matrix.) Note that even though these formulas give a different value for the pagerank vector, the order is the same with either p or r (and order is all we really worry about.) Since we can find the eigenvector efficiently, this is how pagerank is computed.

3. The naive version of PageRank gives a poor result if we have a page with no outgoing links (note that the diagram above does not have this situation.) Name or briefly describe this problem,

and show the adjustment we make to the **B** matrix (NOT the original diagram) to correct for this. (If you already made this adjustment in your answer to part 1, you can skip this part.)

Rank sink: The page with no outgoing links will end up with pagerank 1, everything else 0. If we imagine this as a random walk, if we ever reach that page, we will stay there. Since the random walk model is the amount of time spent on each page as $t \to \infty$, once we reach that page, the fraction of time ends up (in the limit) as 1.

To solve this, we add ``weak'' links from every node to every other node, $B' = \alpha H + (1 - \alpha)B$. Note that we want to do this from all nodes, not just the one with no outgoing links, to ensure all nodes are treated the same.

4. The basic version of PageRank gives a PageRank of 0 to a document with no incoming links. With HITS, if a page has no incoming links, does this mean the Hub and Authority scores must be 0? Explain your answer.

Authority measures the weight a page gets from hubs that point to it, so with no incoming links, authority is 0. The hub score is acquired from the authority of pages that are pointed TO, so even without incoming links, a page that points to high authority pages can have a high hub score.

Please select file(s) Select file(s)

Save Answer *Unsaved Changes

Q3 Relevance Feedback (9 minutes)

7 Points

The query `information retrieval` returns documents A, B, C, D:

A: information retrieval systems

B: relevance feedback in information retrieval

C: information retrieval systems by rocchio
D: retrieval of lost crab pots
The user feels documents B and C are relevant, and documents A and D are not relevant, and they would like to find more relevant documents.

1. Use relevance feedback and the Rocchio formula to develop a new weighted query. You can specify the query as term:weight, e.g., `exam:1 solutions:0.8`. Assume words of one or two letters are stop words. Use 0.5 for both α and β in the Rocchio formula.

Using the basic Rocchio formula, our query becomes the entire vector space, with the weight of each term being the weight in the original query, plus $\alpha \frac{1}{|R|} \sum_{d_i \in Relevant} term\ weight\ in\ d_i$, minus $\beta \frac{1}{|NR|} \sum_{d_i \in NonRelevant} term\ weight\ in\ d_i$.
Assuming weights of 1 for all terms in the query and in the documents, this gives weights of

crab = 0 - 0.5*0.5*1 = -0.25
feedback = 0 + 0.5*0.5*1 = 0.25
information = 1 + 0.5*0.5*(1+1) - 0.5*0.5*1 = 1.25
lost = (same as crab)
pots = (same as crab)
relevance = (same as feedback)
retrieval = 1 + 0.5*0.5*(1+1) - 0.5*0.5*(1+1) = 1
rocchio = (same as feedback)
systems = 0 + 0.5*0.5*1 - 0.5*0.5*1=0

This gives a query of:
crab:-0.25 feedback:0.25 information:1.25 lost:-0.25 pots:-0.25 relevance:0.25 retrieval:1`rocchio:0.25

Now you see why this was expected to take 9 minutes. In practice, we generally only use the few highest and lowest weighted (and assuming we know TF and IDF, we can estimate that without even calculating everything.)

So it looks like no more crab (it was good, though.) But certainly likely to concentrate on relevance feedback papers. Document B:, by the way, is the title of a seminal paper by J. Rocchio.

2. Briefly describe how you might accomplish the same result without showing the user the first list and having them determine which are and which are not relevant.

Blind relevance feedback (also called pseudo-relevance feedback) assumes that our retrieval model is already doing a somewhat reasonable job, and putting the more relevant documents at the top of the list. We treat the top few documents as relevant, and a few near the end of the list as non-relevant, and then apply relevance feedback assuming those are what the user would have chosen. We only need to show the results of the _second_ query to the user.

There were a lot of other reasonable ideas, but most failed to account for _relevance to the current query before showing results to the user_, particularly not downgrading documents that are irrelevant.

I didn't consider ideas reasonable that made use of text categorization or classification techniques and failed to specify what would be used for training data and how it could be obtained.

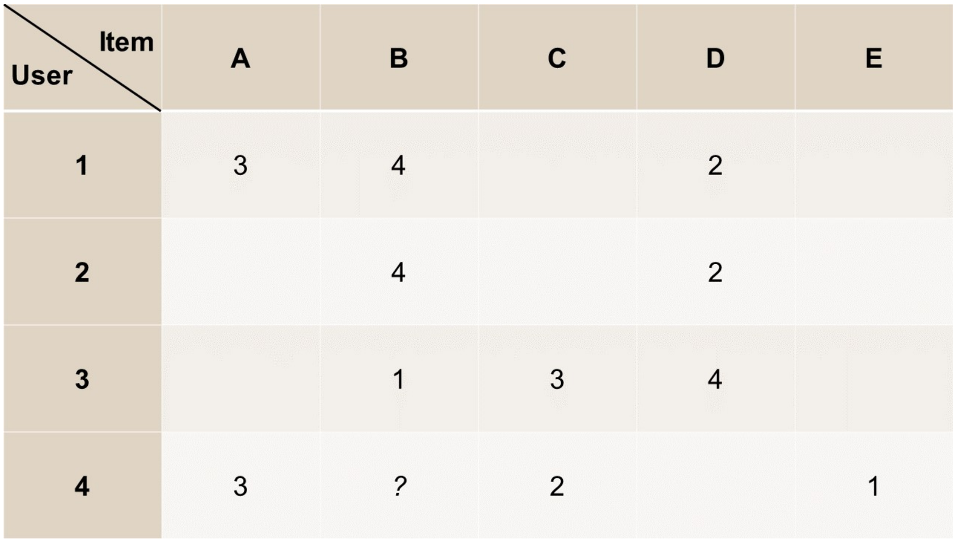
Please select file(s) | Select file(s)

Save Answer | *Unsaved Changes

Q4 Collaborative Filtering (6 minutes)

5 Points

Given the following user ratings on a set of items:



User \ Item	A	B	C	D	E
1	3	4		2	
2		4		2	
3		1	3	4	
4	3	?	2		1

Estimate a rating that user 4 would give to item B (the *?* in the table). Note that there are several ways you can do this, you need to choose a method and show your work.

First, we only need to look at users 1 and 3, since we have no overlapping rates (and thus no means to calculate similarity) with user 2. Which user is closer depends on how you calculate - user 1 and 4 rank A the same (which makes the close), but user 1 ranks it average and user 4 ranks it above average. Likewise, user 3 ranks C as 3, user 4 ranks it as 2. Either such discussion is acceptable.

Alternatively, if we use vector space similarity it works out that with only one item in common the similarity is always 1. Using pearson Correlation Coefficient, the similarity for user 1 is $\frac{0}{0}$, for user 2 is $\frac{1/3}{1/3}$. This shows a weakness of both these similarity metrics.

If we treat similarity as 1 for both users, we get 4's average (2) + the difference from average of users 1 and 3 $\left(0 + \frac{1/8}{2}\right)$ for 2.0625.

This is one possible calculation. The expectation is some discussion of user similiarity, take into account that similarity with 2 is undetermined, take into account 4's average ratings, and have some level of mathematical rigor in determining an average rating.

Please select file(s) Select file(s)

Save Answer *Unsaved Changes

Q5 Text Categorization (8 minutes)

5 Points

Given the following three documents categorized as "Information Retrieval":
A: information retrieval systems
B: relevance feedback in information retrieval
C: information retrieval systems by rocchio

and these two categorized as "Fishing":
D: retrieval of lost crab pots
E: catching crab bait

1. Show how you would categorize the document
   U: systems for catching crab
   using k-nearest neighbor, and state which category your approach would give. You will need to make a few assumptions (e.g., the value for k), state what these assumptions are. You do not need to calculate similarity values, given the assumptions you make you should be able to state which are the nearest neighbors of U without actually calculating specific values. Treat words of three or fewer letters as stop words.

For each of calculation, I will use k=1 and TF*IDF (using raw term frequence and N/DF for document frequency, only because I'm dealing with small documents.) Catching has IDF 5/1, Crab is 5/2, and systems is 5/2. TF is 1 for everything. Since E has the most overlap with U, and that term has highest IDF among terms in U, the numerator would be the largest. Because bait also has high IDF, the denominator could be large - but the only other alternatives would be C (which has an even larger denominator because of more terms and "rocchio", and A which has only a single low IDF word in the numerator. So it is clear that the nearest neighbor is E, giving category "Fishing".

2. (More challenging, 1 point): If these documents are part of a large corpus, it is possible that the categorization of U as either "Fishing" or "Information Retrieval" could change depending on what is in those other documents, even if those are not categorized as either F or IR. Explain briefly (1-3 sentences) how this could happen.

Even if we assume that U gets classified as either IR or Fishing, which of these it is classified as could from what was calculated in 1 based on the weighting of terms. In 1 I calculated IDF based on just the given documents. But if all of the other documents in the corpus used the words "catching" and "crab", then the IDF of systems might be high enough to outweigh their being two matching terms in E, and A would be the nearest neighbor.

Note that some people made assumptions in 1 where this couldn't happen - a reasonable explanation of why it couldn't change (not just simply stating it couldn't) would also be good for credit here.

Please select file(s) | Select file(s)

Save Answer ***Unsaved Changes**

Q6 Duplicate Detection (5 minutes)

3 Points

Suppose I was trying to sell two products, and I created a web page that listed first Product A and it's description, the Product B and it's description. To try and get more visibility in search, I put the page at two web sites (say, creating two different eBay listings). Unfortunately, the search engine eliminates duplicates, and only one shows up.

I try to defeat this by just switching the order of the products; on the second web page I put Product B and it's description first, then Product A and it's description. The descriptions of each are unchanged.

Would a shingle-based approach likely find these to be duplicates, or not? Explain your reasoning. You may need to specify additional information, such as the specific similarity measure, or if you assume doing this on sketches or the full set of shingles, to justify your reasoning. You should be able to do this in 2-3 sentences.

The shingles formed within the products will be unchanged; the only difference will be the shingles spanning the border (e.g., "Buy now! Product B" vs. the swapped order giving ""Buy now! Product A"). While word order matters, it only matters within the size of the single. If we assume size four shingles, at most three shingles would contain words from both products - so the size of the set intersection would be three less than the size of the document. Unless this was a very short document, this would suggest that there is high overlap and the documents are likely duplicates.

Please select file(s) Select file(s)

Save Answer *Unsaved Changes

Q7 Ad-hoc retrieval (5 minutes)

4 Points

Assume we are doing ad-hoc retrieval based only on some form of term similarity (e.g, TF*IDF). We find we have low recall, particularly for terms with many synonyms (e.g., a query for "car" does not return documents about autos or automobiles.

1. Propose a method for addressing this problem that does NOT involve making changes to the retrieval model or require re-indexing documents.

Query expansion using a thesaurus. Add terms to the query that are synonyms of terms already in the query.

2. Suggest a retrieval model that might inherently solve this

problem, and briefly explain how it solves the problem of synonyms.

Latent Semantic Indexing. Terms and documents are mapped into a _latent space_. Significant overlap in terms (even if not all are exactly matching) results in mapping two the same vector in the latent space. The query is then mapped into the same space, so the query terms are put together with other terms that are used with the same terms, even if those terms do not match.

Several people suggested clustering. While this may get appropriate documents, it is likely to give too many - and it isn't clear how you would rank them.

Each part can be answered in 1-3 sentences.

Please select file(s) Select file(s)

Save Answer *Unsaved Changes

Q8 Query Expansion (5 minutes)

4 Points

A user issued a query and observed the search results; but find they get few relevant results in the top 10 documents. What we can say about how the results would get better or worse, in terms of some rank-based metric (you choose and name the metric) if we modify the query by:

1. Adding a new term to the query that is found only the the relevant documents,

Since the user is already using the top 10 documents, let's look at precision@10. Right now it is low (few relevant documents). By adding terms from those documents should return more documents, but it is unknown if those will be relevant. For example, a query about "COVID-19" when we are interested in COVID-19 testing may result in adding "testing" from the relevant document s, but it may may contain many documents with the term "testing", but adding testing will also get a lot of documents that are non-COVID-19. It may also add terms like "respiratory" that could lead to irrelevant documents coming near the top, forcing relevant documents out of the top 10. As such, precision@10 could easily go down.

Where it is likely to help is if the query uses a term that is not in common use, e.g., "SARS-COV-2", so there are few relevant documents containing that term. If adding terms from relevant documents gets terms like "COVID-19" and "testing", we may get a number of additional relevant documents ranked near the top, and precision@10 would go up.

Either example alone would be a perfectly good answer to this question. Kudos to the few of you who discussed average precision, showing how this is impacted makes for a very interesting discussion. For those that didn't do so, I suggest you think about it, as figuring it out on your own will help you understand quite a bit about ranking.

2. Removing a term from the query that is found only in irrelevant documents.

Removing terms found in irrelevant documents is likely to improve precision@10, as it will cause those irrelevant documents to be ranked lower, allowing different documents to "float up"; hopefully some of those would be relevant, giving us more relevant documents in the top 10 (and those higher precision@10).

Please select file(s) Select file(s)

Save Answer ***Unsaved Changes**

Q9 Collaborative Filtering (4 minutes)

2 Points

How can we use collaborative filtering to improve ad-hoc retrieval in a personalized search setting, where we have a history of what a user has found relevant or not relevant in the past? Note that this isn't the typical product/dish recommendation problem, but a system where a user issues a specific query. Briefly propose an approach (a good answer could be as short as one or two sentences.)

We can use the history to identify documents the user has found relevant/non-relevant as ratings (+/-); this gives us a matrix of ratings (although a very simple 0/1, rather than the 5-6 point scale we often think of). We can then apply collaborative filtering techniques to score documents a user hasn't seen.

The question is where to apply this. I would use this as a component in ranking search results, much the way we use PageRank. Thus the ranking would be based on a similarity score that combines similarity to the query and the relevance prediction for thinks that show up as sufficiently similar.

While using collaborative filtering based on similar queries is a possibility, this misses the point of AD-HOC retrieval. From Webster's dictionary, ad-hoc is _formed or used for specific or immediate problems or needs_, so we don't want to assume we've seen similar queries before. Although if you are operating at Google-scale, this might work for some queries - but you need to have not only similar queries, but considerable overlap between users in similar queries. It isn't enough that you have 100 people who've searched for "deep-fried artichoke", you also need to have several other things you've searched for that some of those 100 people have searched for, so you can determine similarity between users. More likely that users have looked at a selection of similar documents, perhaps returned by different queries.

Another common answer was to look at just the user's own history - but this misses the point of COLLABORATIVE filtering.

Please select file(s) Select file(s)

Save Answer *Unsaved Changes

Save All Answers

Submit & View Submission ❯