Patient Similarity

A textual recommender system and other text mining approaches for clinical data

---

Philipp Hummel

June 14, 2017

Idea

- Doctors write down most important information about the patient in physician letters
- Doctors often face unusual patients; therapy decision is unclear
- In those cases: retrieve letters of similar patients from database to support clinical decision making

Sample Letter

Data Clean-Up

Two types of duplicates: Exact copies and Follow-Ups

Two approaches for finding them: String Matching and Bag of Words

Bag of Words Example:

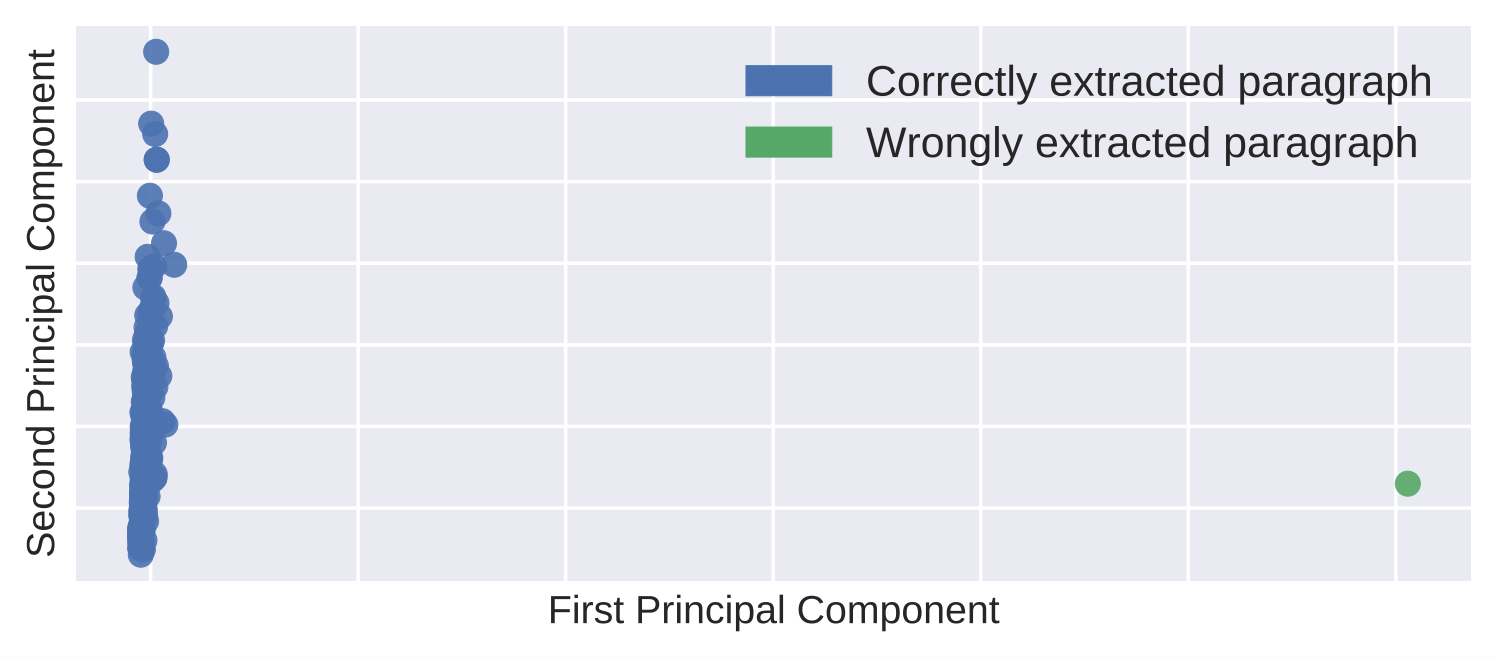
$d_1 =$ "The patients with disease A."
$d_2 =$ "The patients with disease B."

$$\mathbf{v}_{d_1} = \begin{matrix} 1 \\ 1 \\ 1 \\ 0 \end{matrix} \quad \mathbf{v}_{d_2} = \begin{matrix} 1 \\ 1 \\ 0 \\ 1 \end{matrix} \quad \begin{matrix} \text{patient} \\ \text{disease} \\ \text{A} \\ \text{B} \end{matrix}$$

Paragraph Extraction

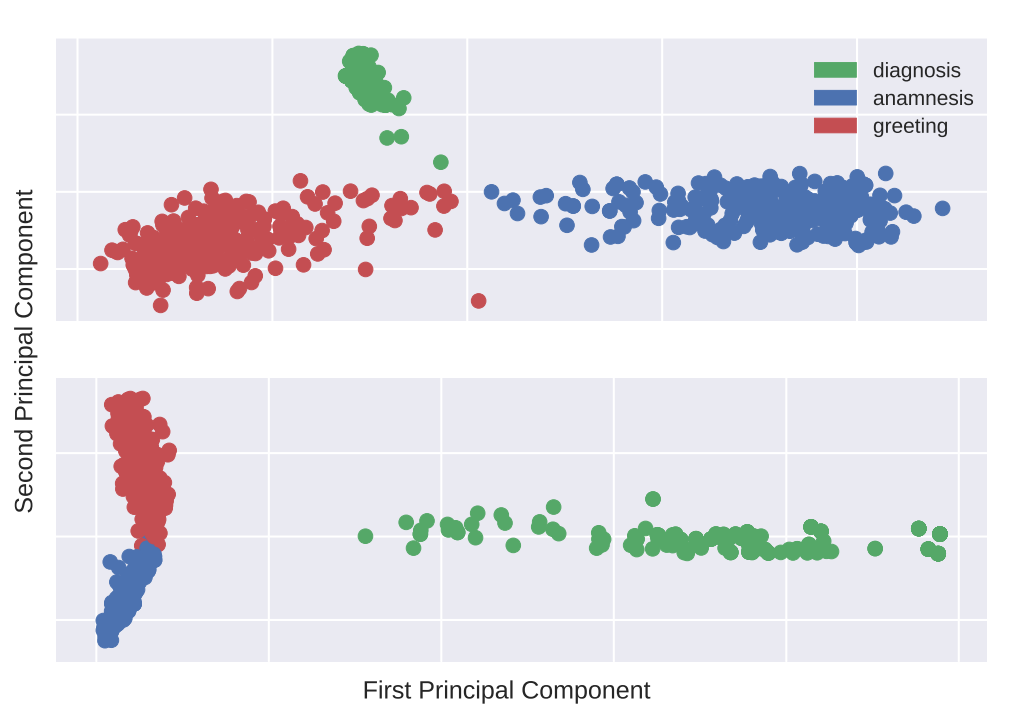
Rule-based extraction of several paragraphs

Semi-automatic outlier detection



Paragraph Classification

- Embed all paragraphs into different vector space models like bag of words, tf-idf, paragraph vector
- use these embeddings and hand-made labels as inputs to a classifier
- see if classifier can learn to identify different kinds of paragraphs



Recommender System

- Find similar letters based on distance of corresponding vectors in the embedding space
- How to find which embedding works best? Supervised Information!
- Our Data: Grouping of letters into groups of similars
- Our Metric: How often are the similars ranked by algorithm into the top N
- tf-idf turns out to be best method; paragraph vector not much worse

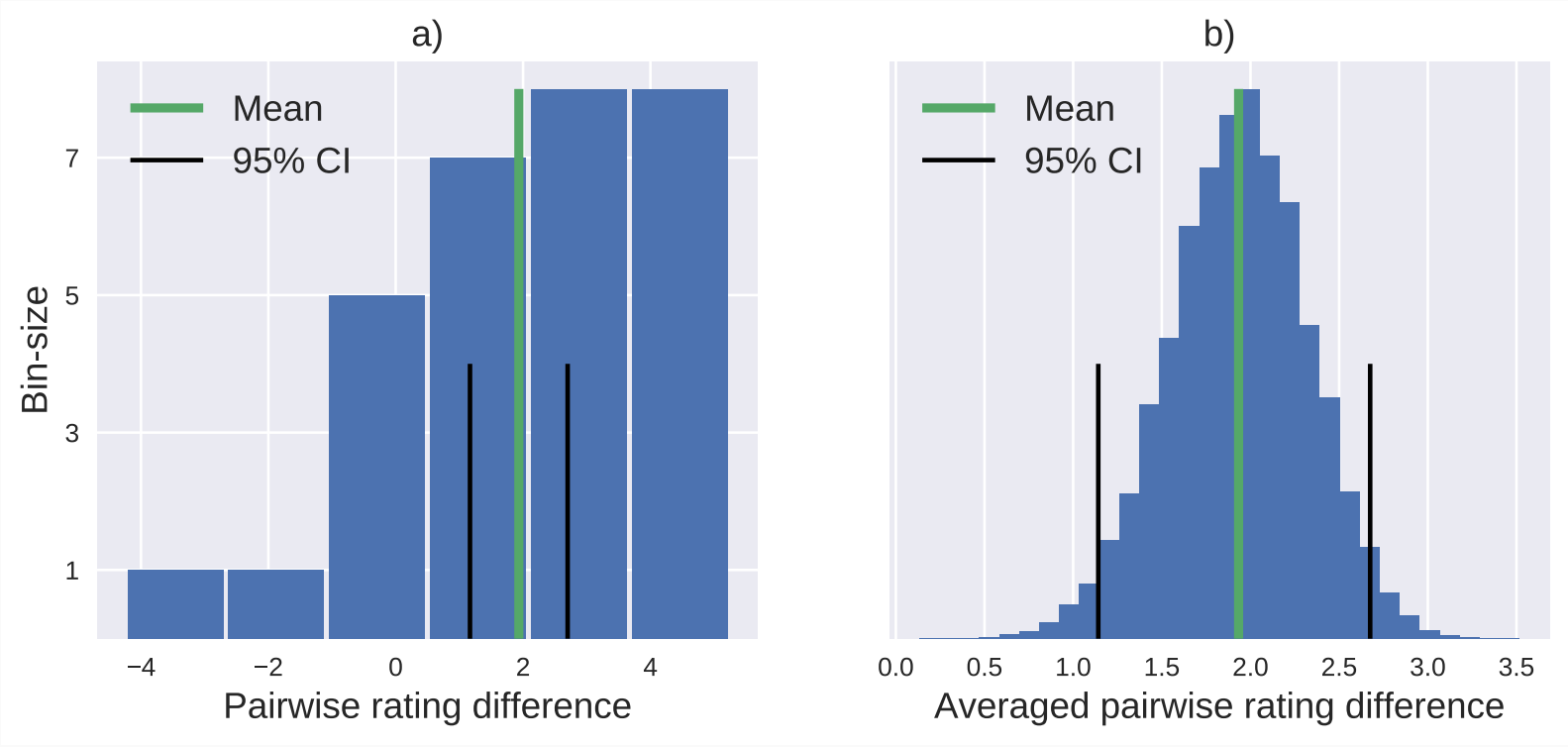
Experiment

Results – Inter Rater Agreement

- We measure correlation with spearman rank correlation
- Often Expert Ratings are more consistent than student ratings
- This is similar in our case
- We therefore discard student data for further analysis

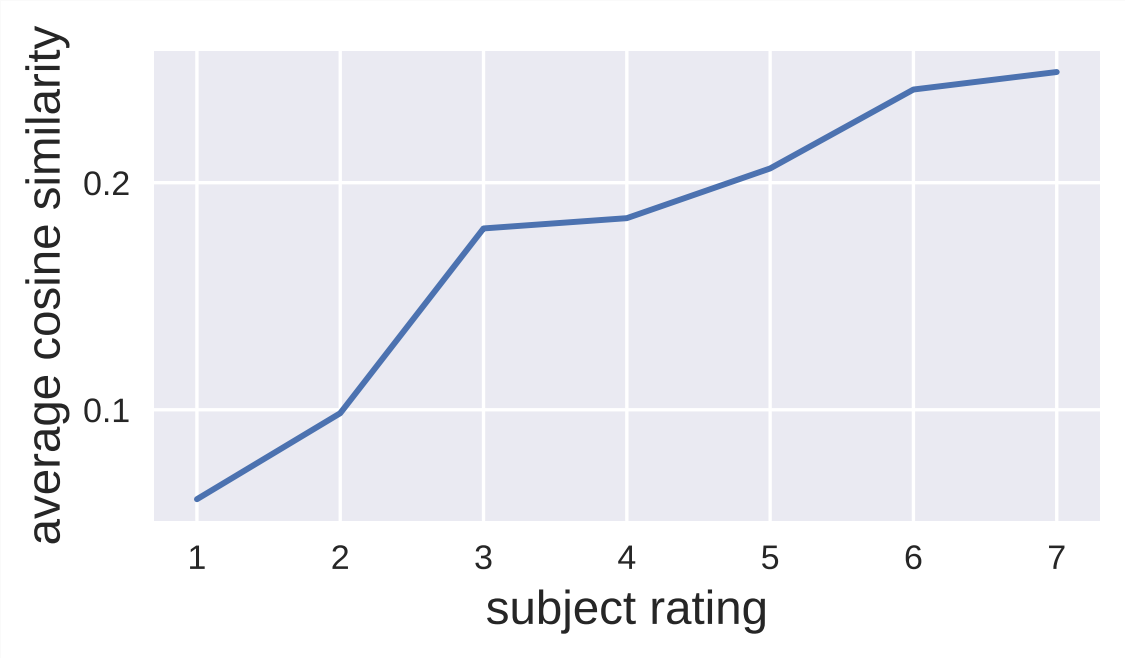
Results – Best vs. Random Letter

How much better than chance is our recommendation?



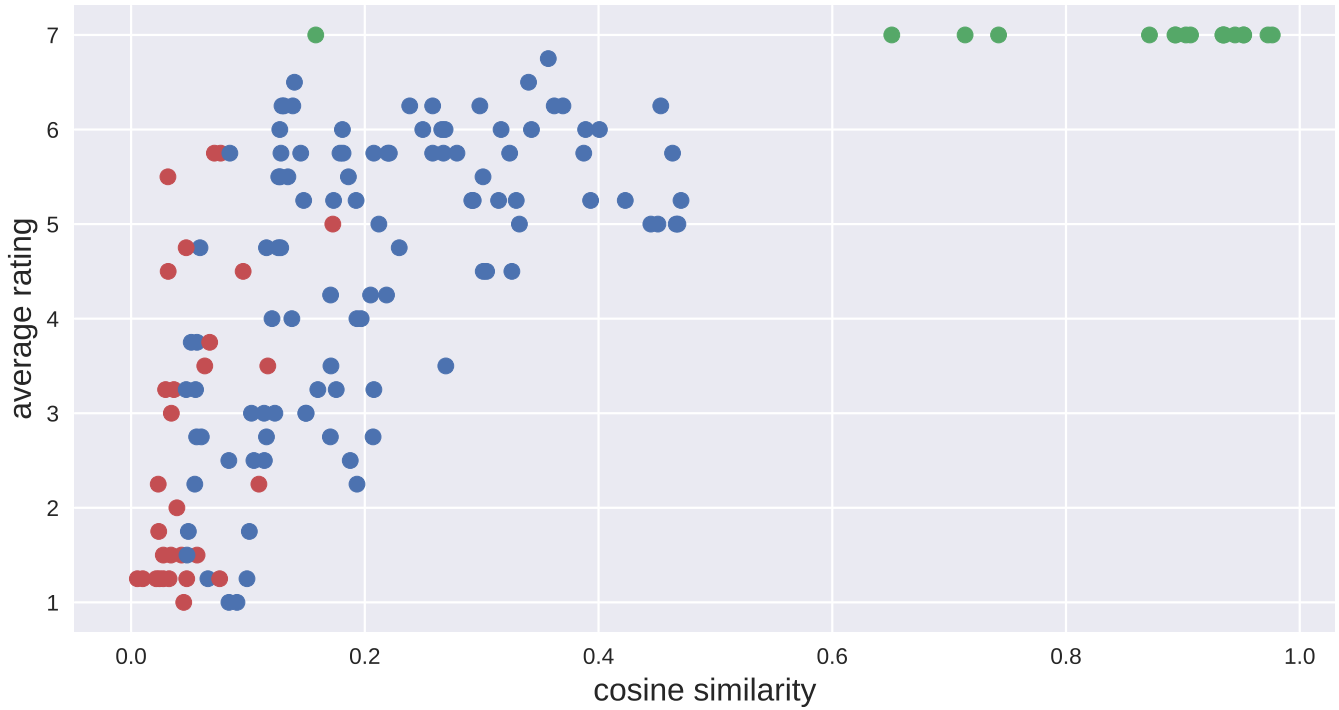
Results – Cosine Similarity and Subject Ratings

The recommender system produces a ranking of the letters, but can we use absolute distance (or similarity) information as well?



Results – Cosine Similarity and Subject Ratings – More Details

Plot all pairs as data points. Subject rating vs. Cosine Similarity

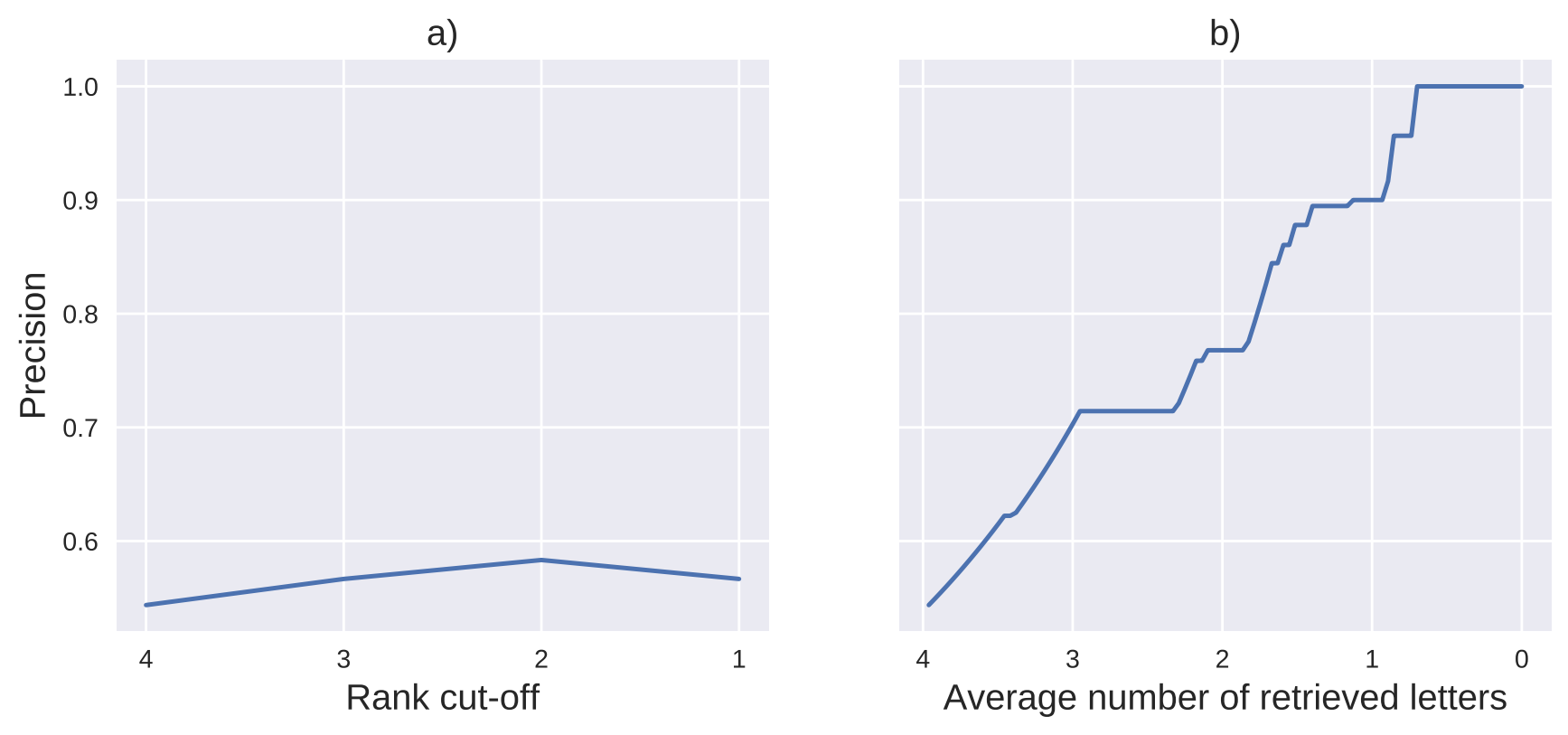


Results – Precision and Recall

- Precision $P(\textit{relevant}|\textit{retrieved})$ is easy to compute
- Every letter with rating > 5 is relevant
- We cannot know recall $P(\textit{retrieved}|\textit{relevant})$, because we do not know all relevant letters for a given information need
- Recall is not so important for our setting. We do care only whether 5 good letters are retrieved. Not whether all relevant letters are retrieved.
- We can still look at the correlation between ranking of algorithm and ranking of subjects.

Results – Precision

How good is precision in different recommending scenarios?



Results – Recall proxy

We can look at correlation between ranking of subjects and ranking of algorithm with spearman rank correlation

- Algo – Expert Agreement: 0.39
- Inter Expert Agreement: 0.72

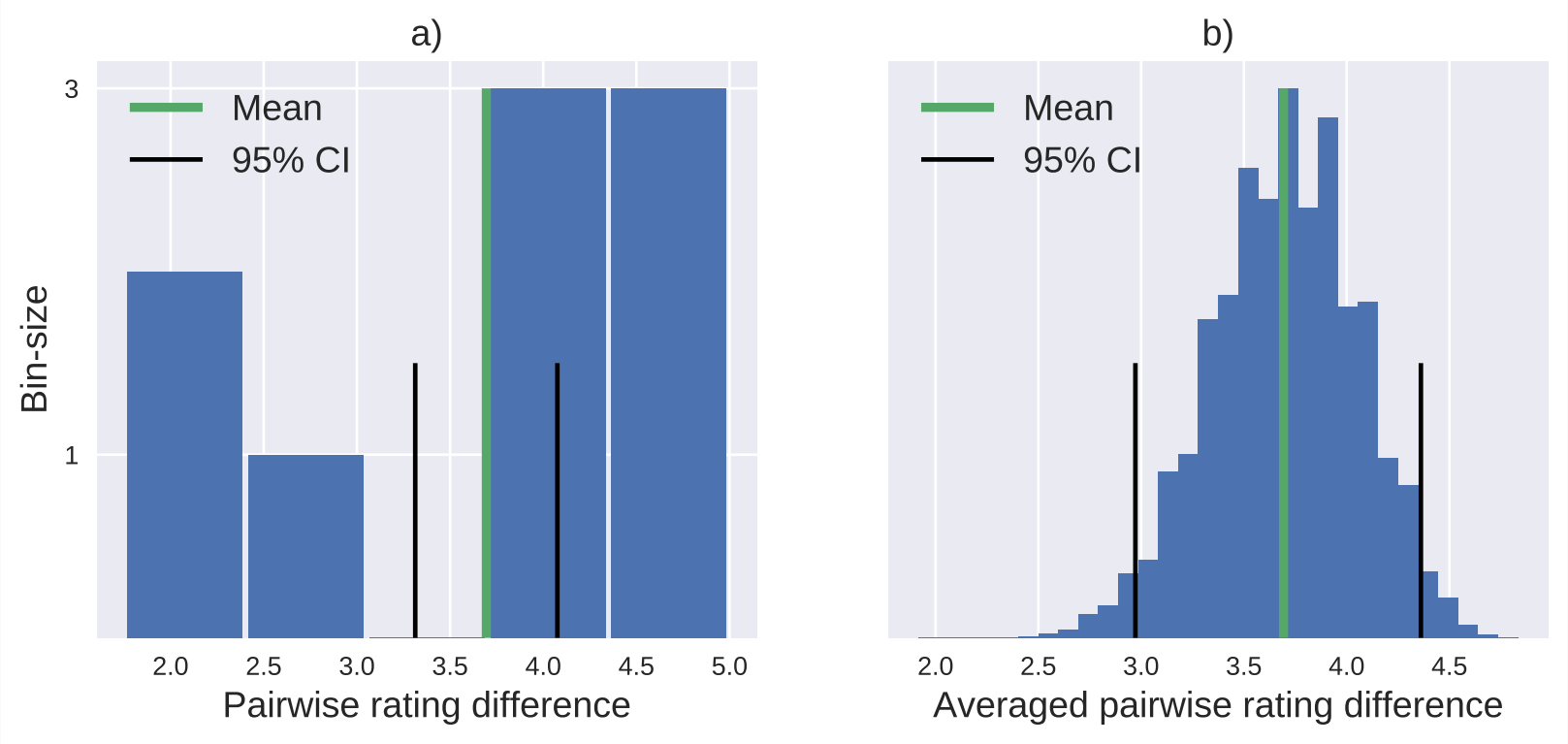
Considerations of Usability

Two reasons why results are probably better in reality than it seems here:

- subjects might adjust their ratings, when all presented examples are bad. Therefore somewhat similar letters might be rated more similar than they really are.
- in bigger database we expect more letters with high cosine similarity

Possibilities to improve

Can we use the embeddings of two different embedding methods to obtain better results? Paragraph vector seems most promising.

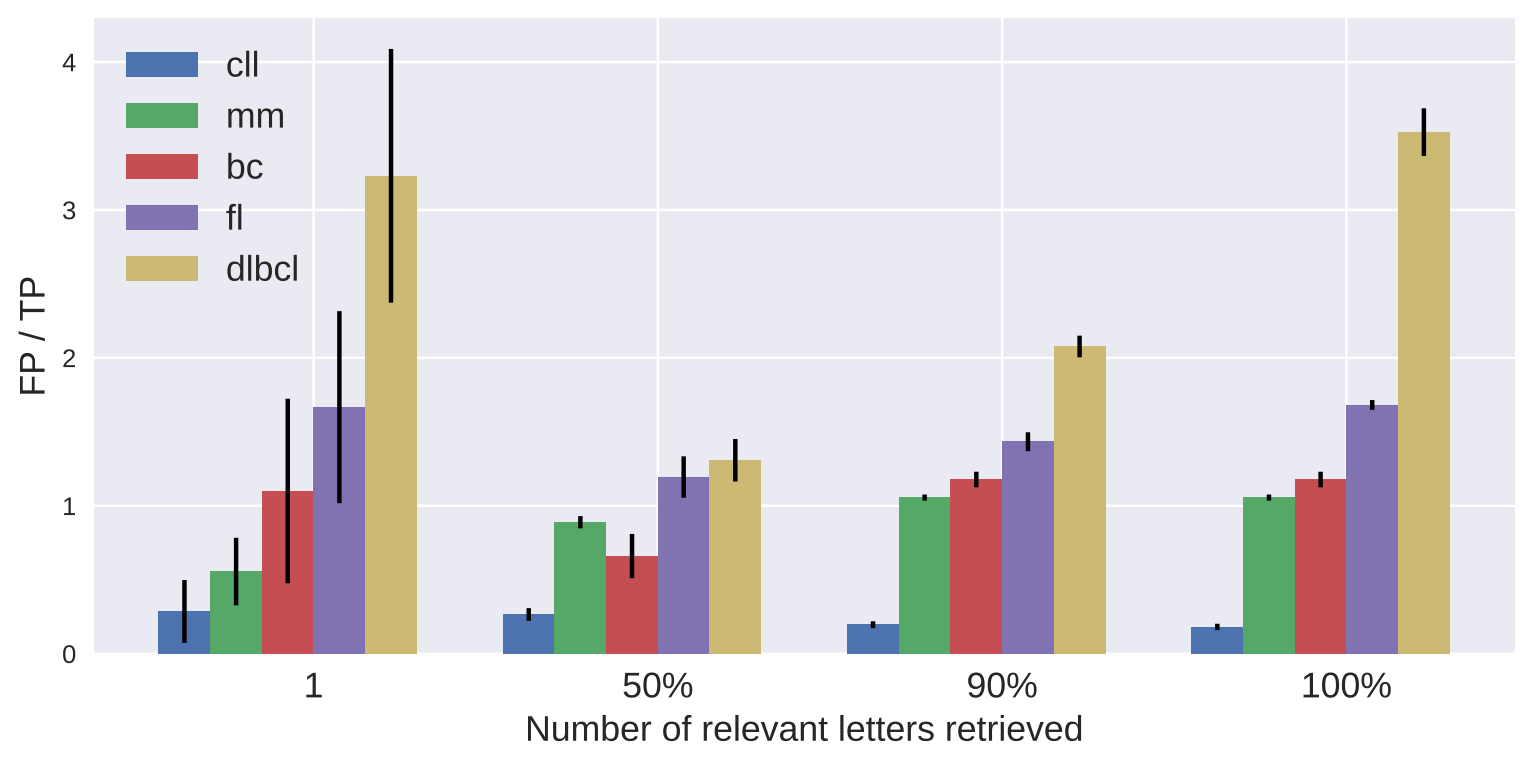


Additional Application – Intelligent Search

- Say we want to find all letters of patients with a specific feature.
- Traditionally: search database with string matching algorithms.
- Hardly doable in some cases
- Can we do better with embedding methods?
- We do experiment to find all letters of patients with a specific disease.

Intelligent Search – Results

For five diseases: How expensive (timewise) is it to find different fractions of the relevant letters?
We start with one letter and look for the next best match first with cosine similarity, then with classifier.



Possible application for text mining in law etc.?!

Questions?