Yelp Dataset Challenge

Jivitesh Poojary, Prathamesh Jangam, Shreyas Rewagad, Vighnesh Nayak

Github: https://github.iu.edu/srewagad/srewagad-jpoojary-vnayak-pjangam-information-retrieval

Department of Information and Library Science

School of Informatics, Computing, and Engineering
December 04, 2017

Project Contributions

Jivitesh Poojary	Code - Task 1 - CBR - Data Extraction, CF - Pearson & Cosine similarity, Presentation
Prathamesh Jangam	Code - Lexical Centrality, Presentation
Shreyas Rewagad	Code - Task 1 - Entire workflow - CBR & CF, CF - Sentiment Analysis and Model Evaluation, Github repo, Documentation.
Vighnesh Nayak	Code - continuous Lex Rank, IDF modified cosine.

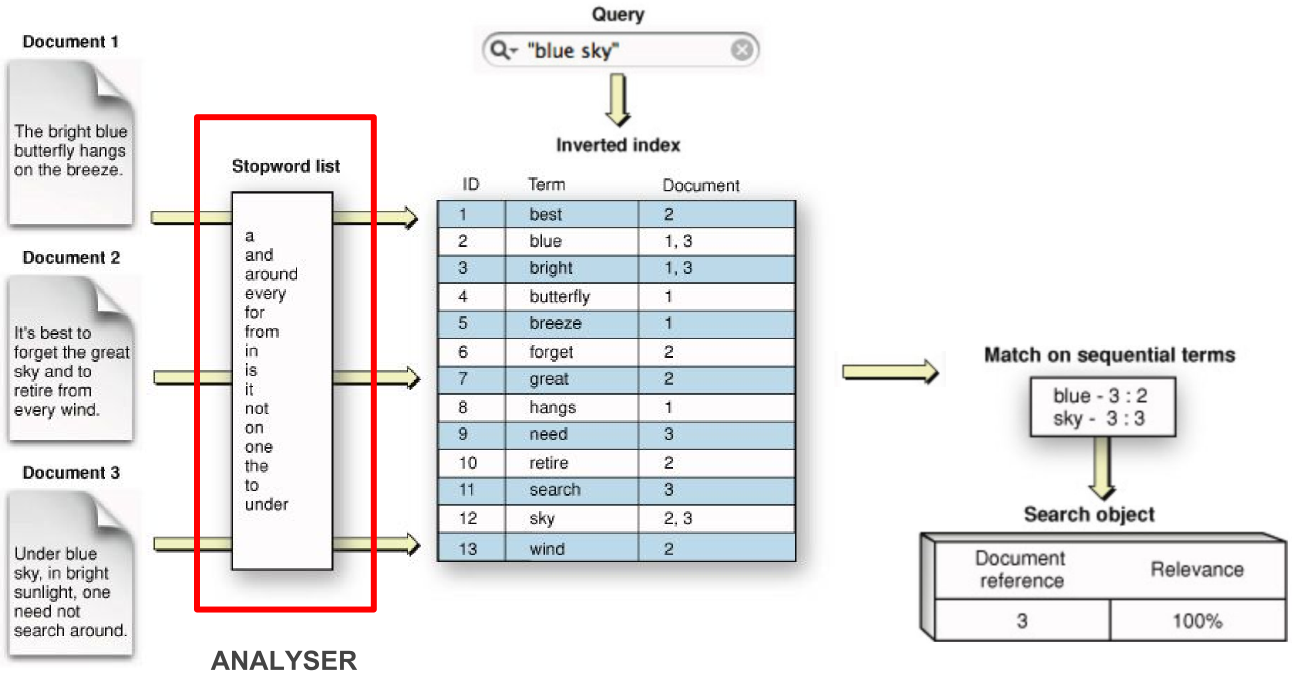
Project Objective

- **Task 1: Recommend Businesses to User**
  - Content based recommendation using Reviews
  - User-User Memory-based Collaborative filtering using sentiment analysis scores form reviews
  - Item-Item Memory-based Collaborative filtering using sentiment analysis scores form reviews

- **Task 2: Generation of Business Description form User reviews**
  - Lex-Rank based review summarization.

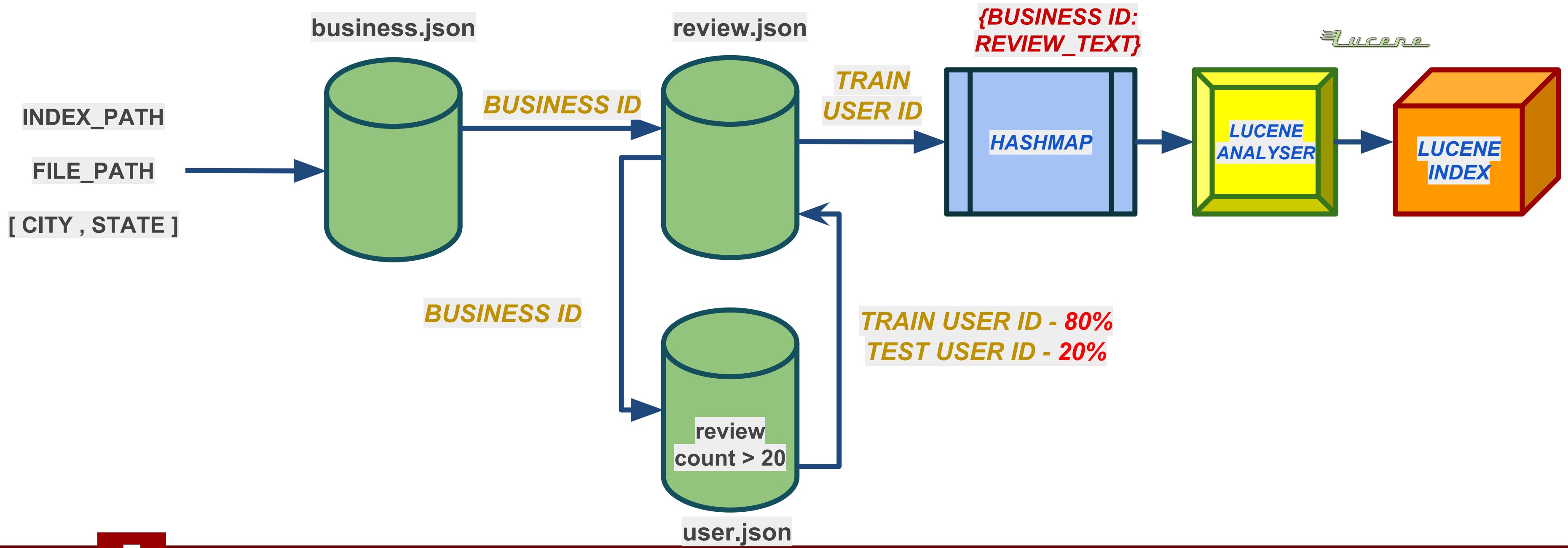
Task 1

Task 1: Content based recommendation (CBR) using Reviews and Tips



https://developer.apple.com/library/content/documentation/UserExperience/Conceptual/SearchKitConcepts/searchKit_basics/searchKit_basics.html

Task 1: CBR - Indexing Workflow



Task 1: CBR - Similarity

$$\mathrm{tfidf}(t,d,D) = \mathrm{tf}(t,d)\cdot \mathrm{idf}(t,D)$$

$$\mathrm{tf}(t,d) = 0.5 + 0.5\cdot \frac{f_{t,d}}{\max\{f_{t',d} : t' \in d\}}$$

$$\mathrm{idf}(t,D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

TF IDF

$$\mathrm{score}(D,Q) = \sum_{i=1}^{n} \mathrm{IDF}(q_i)\cdot \frac{f(q_i,D)\cdot(k_1+1)}{f(q_i,D) + k_1\cdot\left(1 - b + b\cdot \frac{|D|}{\mathrm{avgdl}}\right)},$$

$$\mathrm{IDF}(q_i) = \log \frac{N - n(q_i) + 0.5}{n(q_i) + 0.5},$$

BM25

$$P_\beta(w \mid \hat\theta) = (1-\beta)\,\frac{c(w,D)}{|D|} + \beta\, P(w \mid C)$$

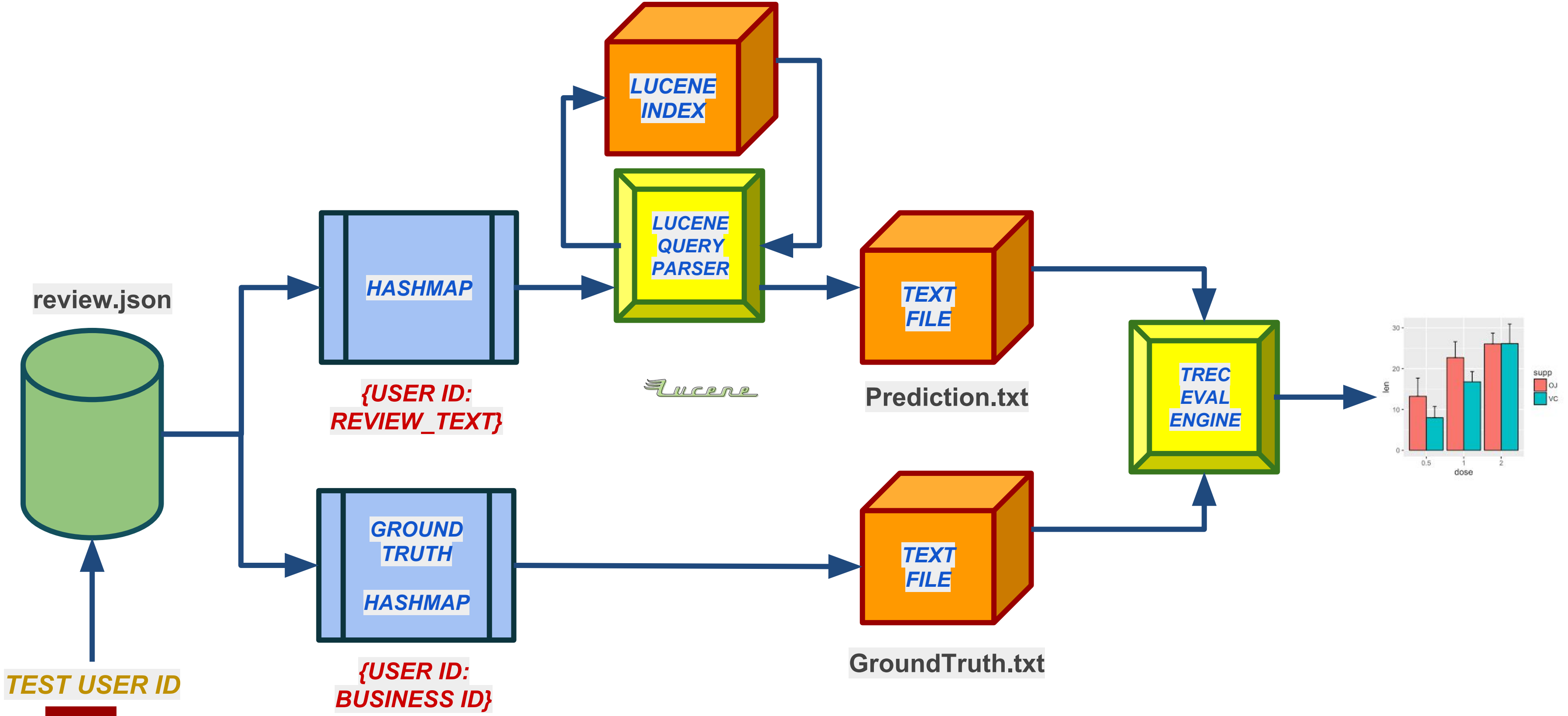
JELINEK-MERCER SMOOTHING

$$P_\mu(w \mid \hat\theta) = \frac{c(w,D) + \mu\, P(w \mid C)}{|D| + \mu} = \frac{|D|}{|D|+\mu}\cdot\frac{c(w,D)}{|D|} + \frac{\mu}{\mu + |D|}\cdot P(w \mid C)$$

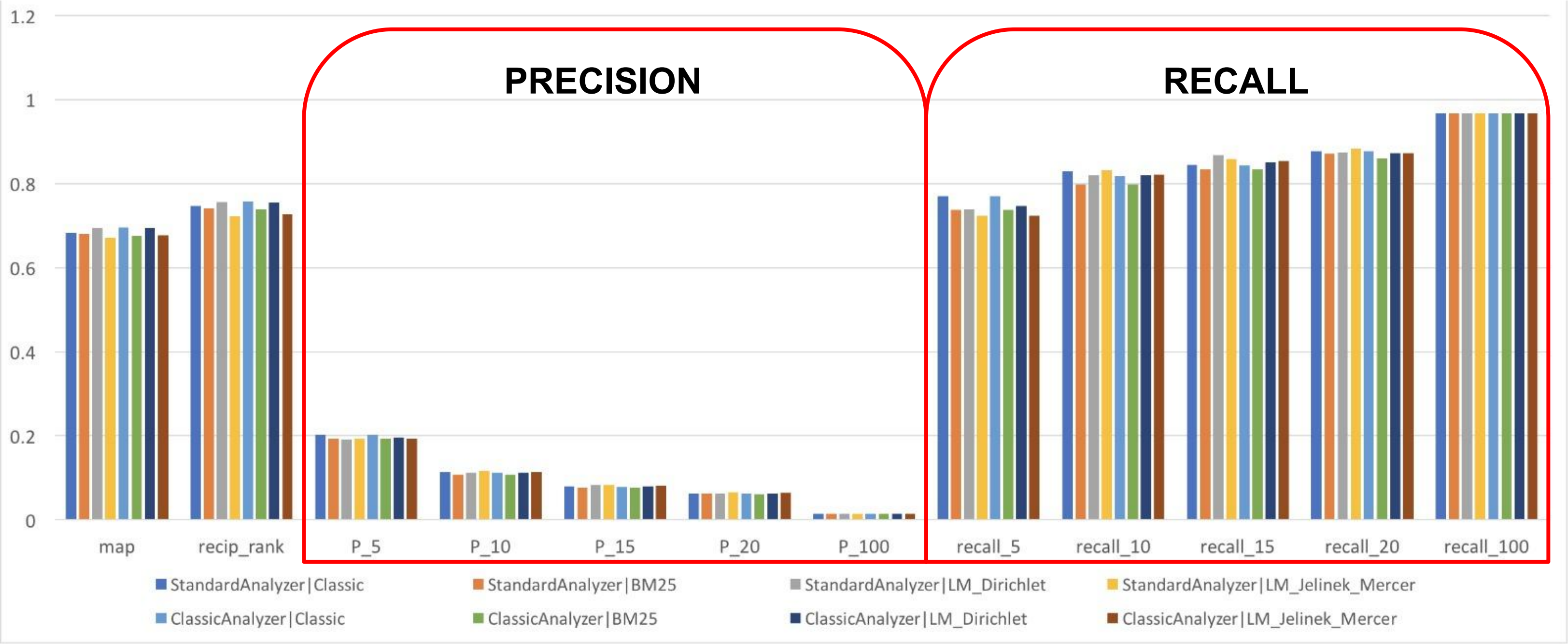
DIRICHLET PRIOR SMOOTHING

https://en.wikipedia.org/wiki/Tf%E2%80%93idf
https://en.wikipedia.org/wiki/Okapi_BM25
http://mlwiki.org/index.php/Smoothing_for_Language_Models

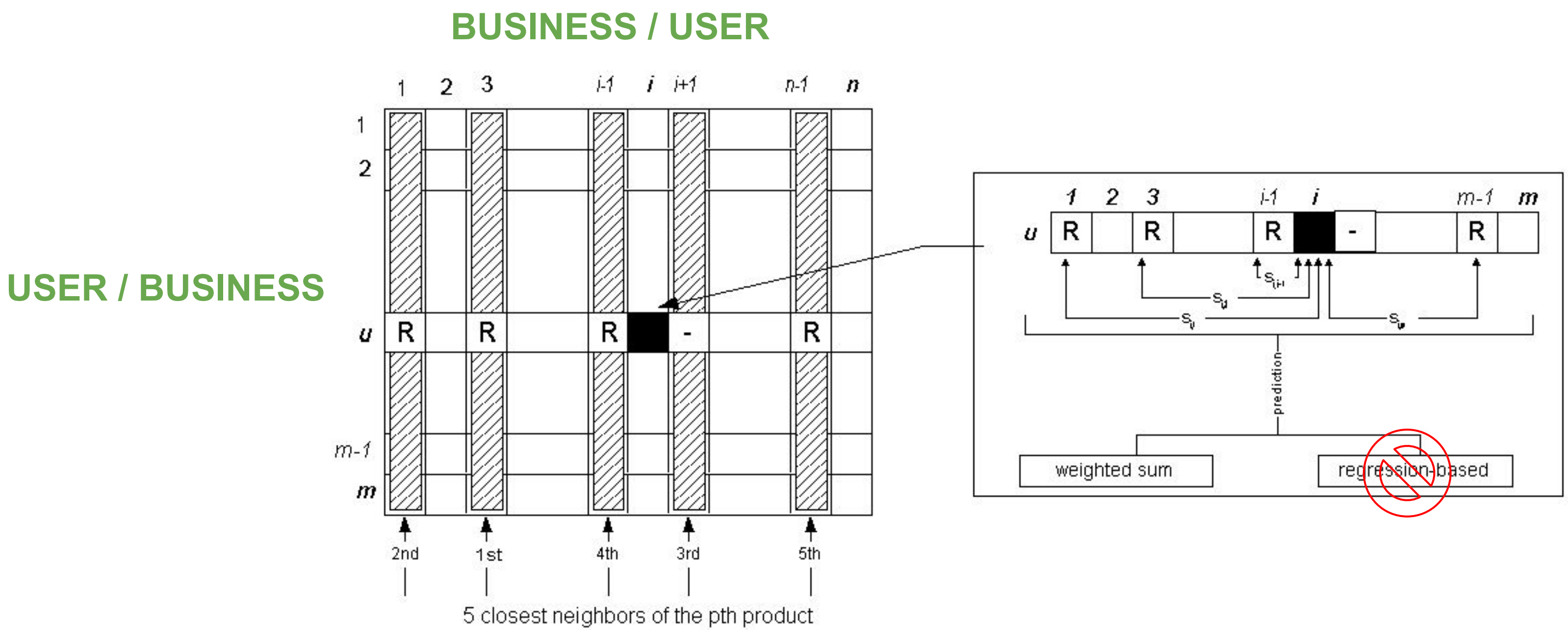
Task 1: CBR - Prediction Workflow



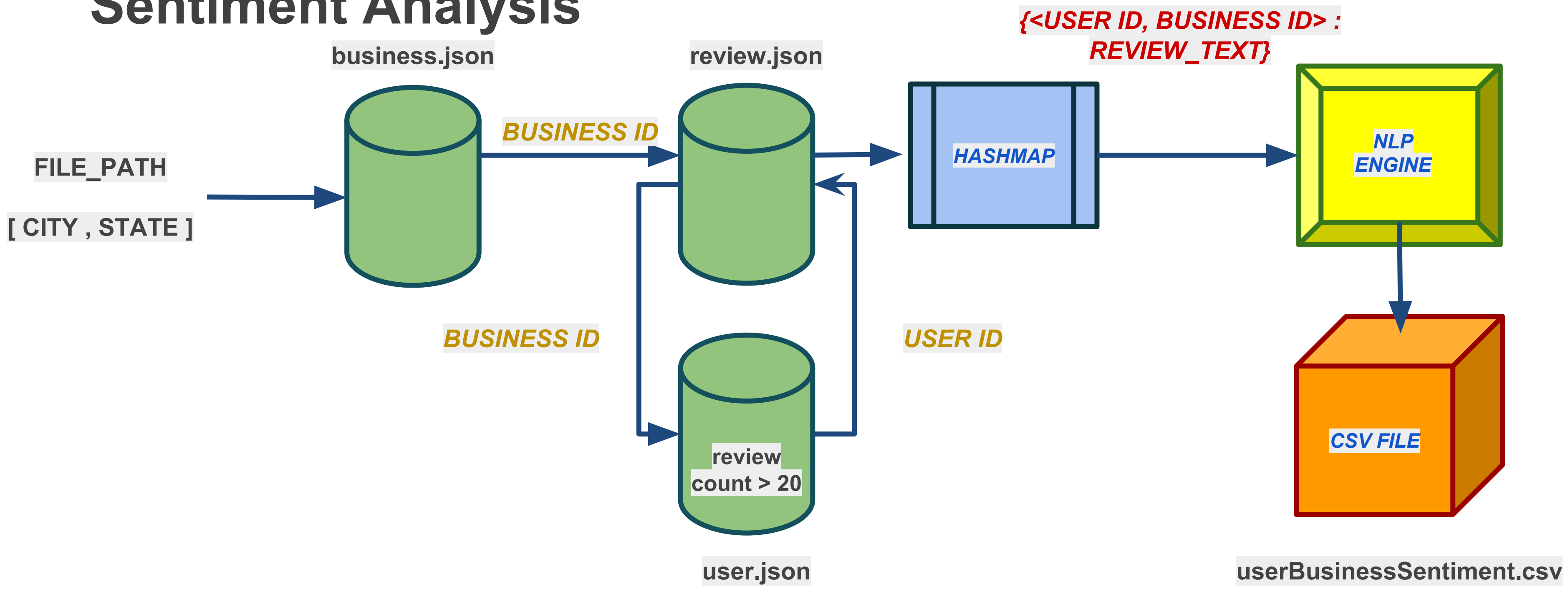
Result - Content based Recommendation



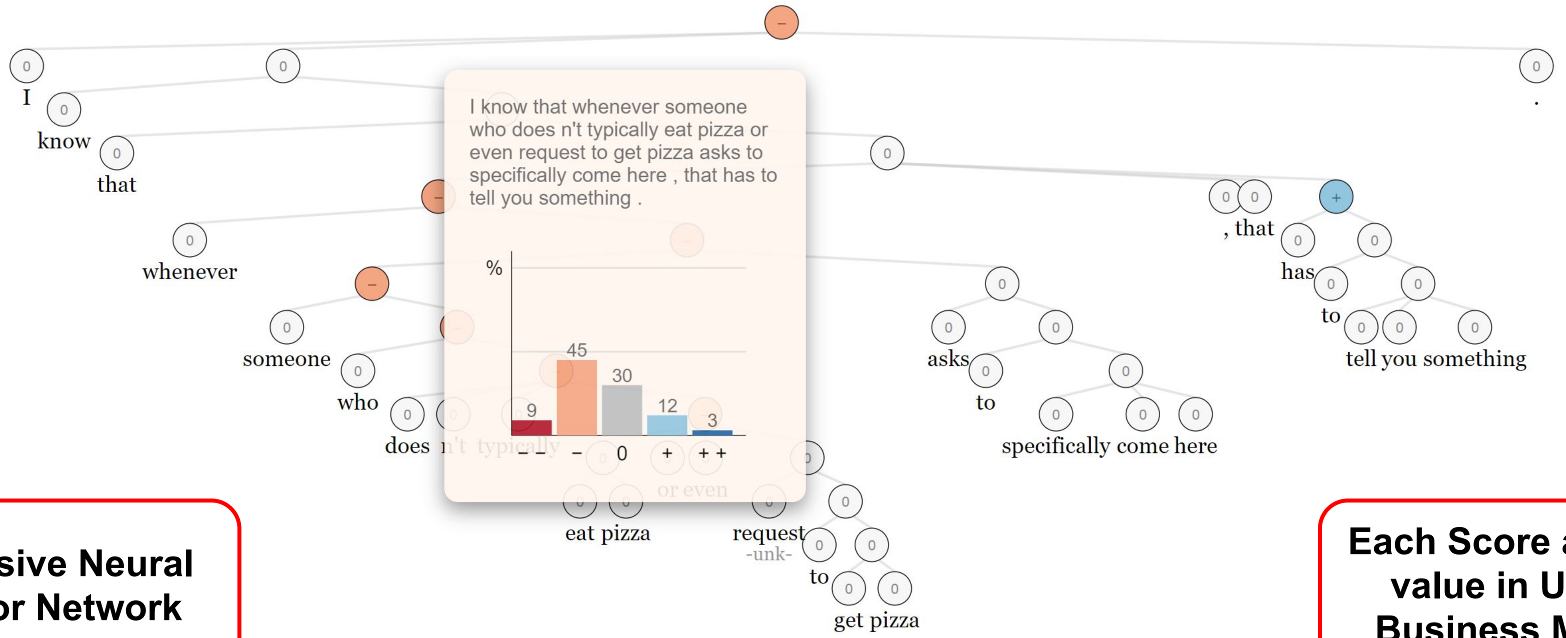
Task 1: Collaborative Filtering (CF)



Task 1: CF - Content Extraction Workflow using Sentiment Analysis



Task 1: CF - Sentiment Analysis



Negative Score

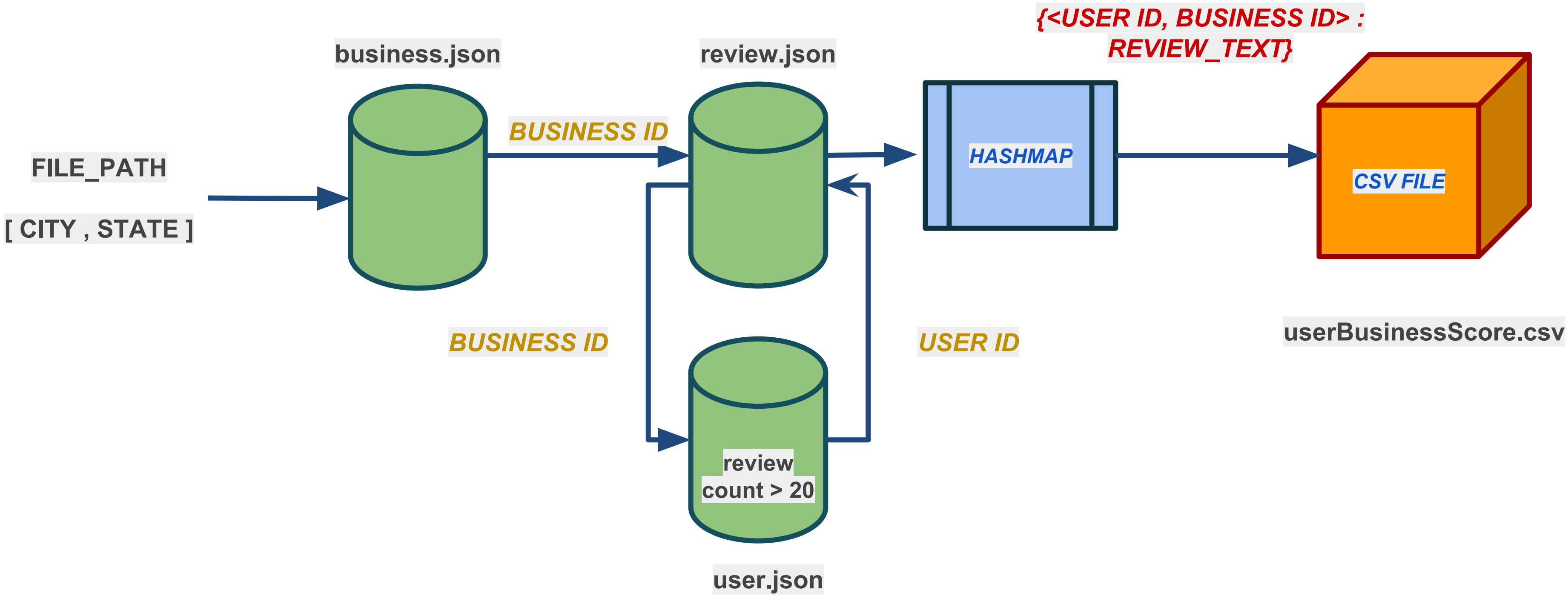
1

Recursive Neural Tensor Network

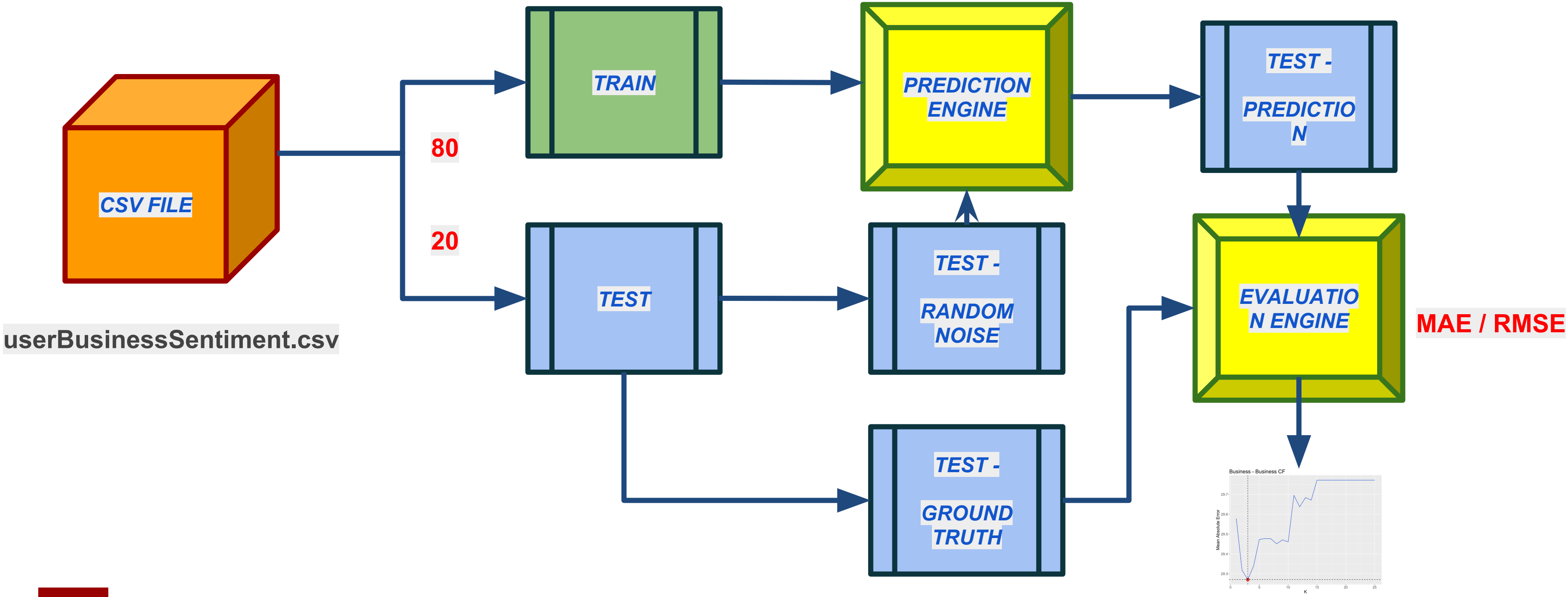
Each Score acts as value in User - Business Matrix

https://stanfordnlp.github.io/CoreNLP/simple.html
https://nlp.stanford.edu/~socherr/EMNLP2013_RNTN.pdf

Task 1: CF - Content Extraction Workflow



Task 1: CF - Prediction Workflow



Task 1: CF - Similarity

Here we use both the Cosine and Pearson similarity metric for measuring the distance between the vectors and finding the top K Users / Businesses most similar to the User / Business under consideration.

$$w(u,v) = cos(\boldsymbol{u},\boldsymbol{v}) = \frac{\boldsymbol{u}\cdot\boldsymbol{v}}{||\boldsymbol{u}|| * ||\boldsymbol{v}||} = \frac{\sum_{i\in I} r_{ui}r_{vi}}{\sqrt{\sum_{i\in I} r_{ui}^2}\sqrt{\sum_{i\in I} r_{vi}^2}}$$

COSINE

$$w(u,v) = \frac{Cov(u,v)}{\sigma_u\sigma_v} = \frac{\sum_{i\in I}(r_{ui}-\bar{r}_u)(r_{vi}-\bar{r}_v)}{\sqrt{\sum_{i\in I}(r_{ui}-\bar{r}_u)^2}\sqrt{\sum_{i\in I}(r_{vi}-\bar{r}_v)^2}}$$

PEARSON

SIMILARITY MEASURE

Task 1: CF - Prediction and Evaluation

Our Objective is to iterate over different values of K and find the optimal value which minimizes the MAE and RMSE

USER

$$P_{ai} = \frac{\sum_{u \in U^k} \mathbb{1}(r_{ui}) \cdot w_{au}}{\sum_{u \in U^k} w_{au}}$$

$$\mathbb{1}(r_{ui}) = \begin{cases} 1 & \text{if } u \text{ has listened to } i \\ 0 & \text{if } u \text{ has not listened to } i \end{cases}$$

ITEM

$$P_{ai} = \frac{\sum_{n \in N} r_{un} w_{in}}{\sum_{n \in N} |w_{in}|}$$

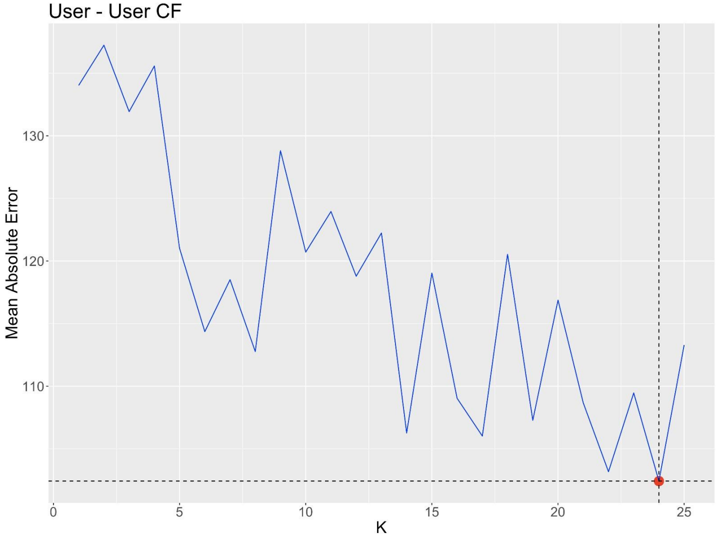
PREDICTION

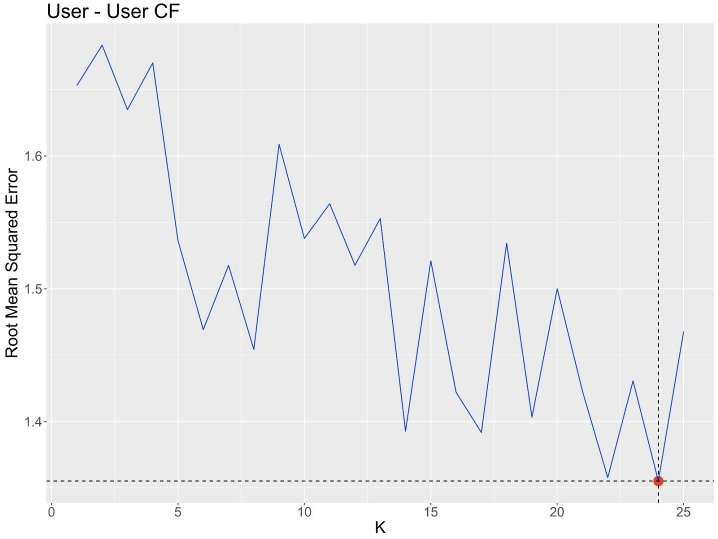
$$MAE = \sqrt{\frac{1}{|\mathcal{T}|} \sum_{(u,i) \in \mathcal{T}} |\hat{r}_{ui} - r_{ui}|}$$

$$RMSE = \sqrt{\frac{1}{|\mathcal{T}|} \sum_{(u,i) \in \mathcal{T}} (\hat{r}_{ui} - r_{ui})^2}$$

ERROR CALCULATION

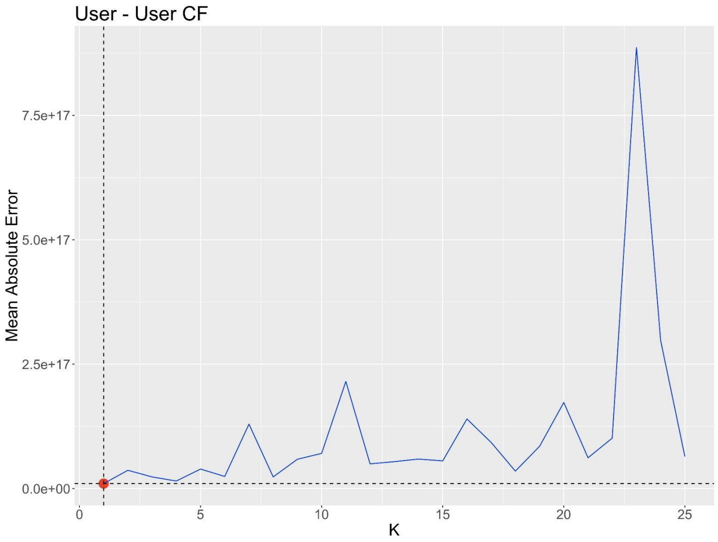
Task 1: User-User Memory-based CF using Cosine Similarity

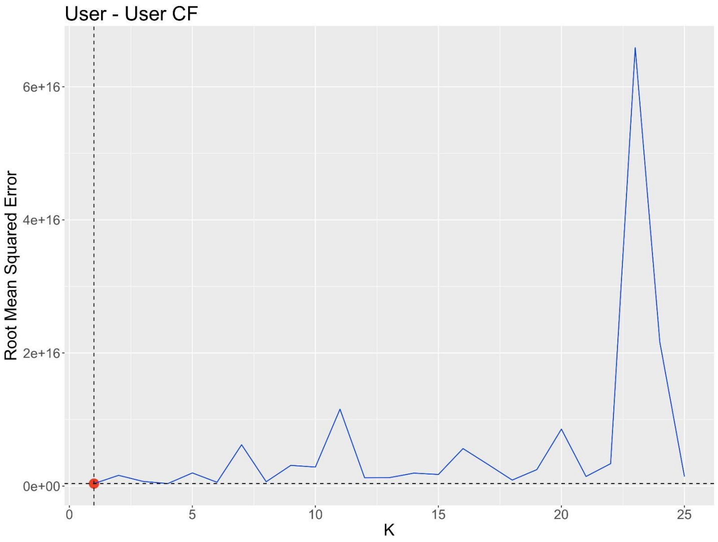




From the plots we can conclude that at K = 24 we obtain the least error

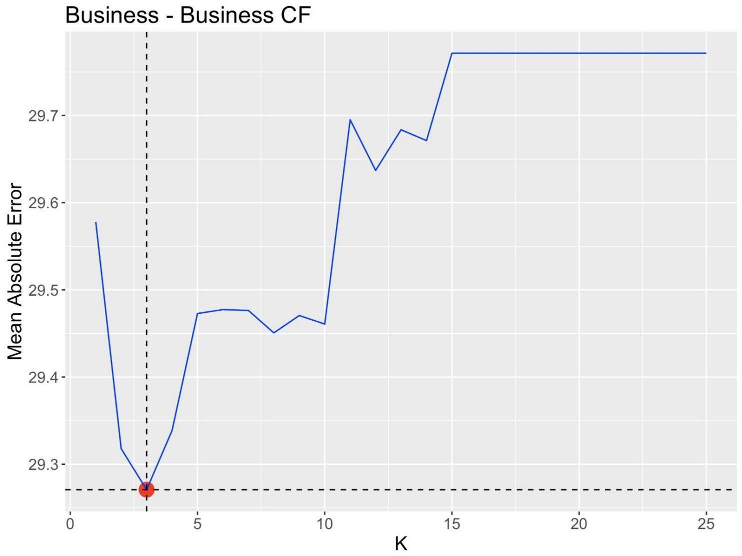
Task 1: User-User Memory-based CF using Pearson Similarity

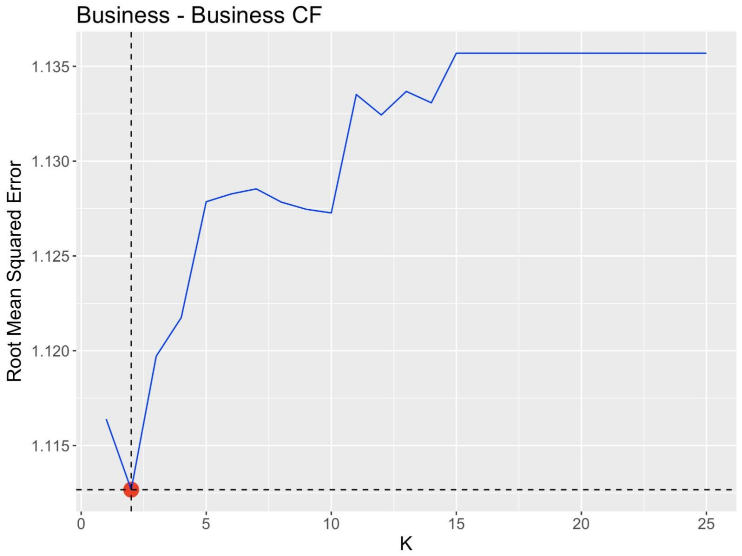




From the plots we can conclude that at K = 1 we obtain the least error

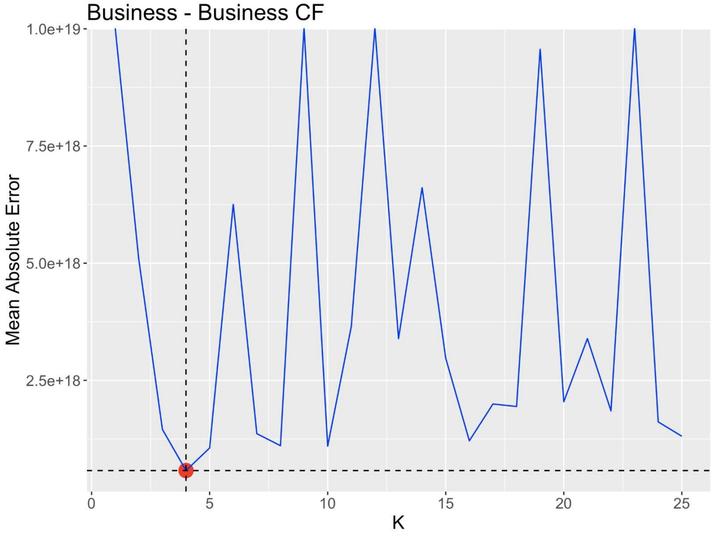
Task 1: Item-Item Memory-based CF using Cosine Similarity

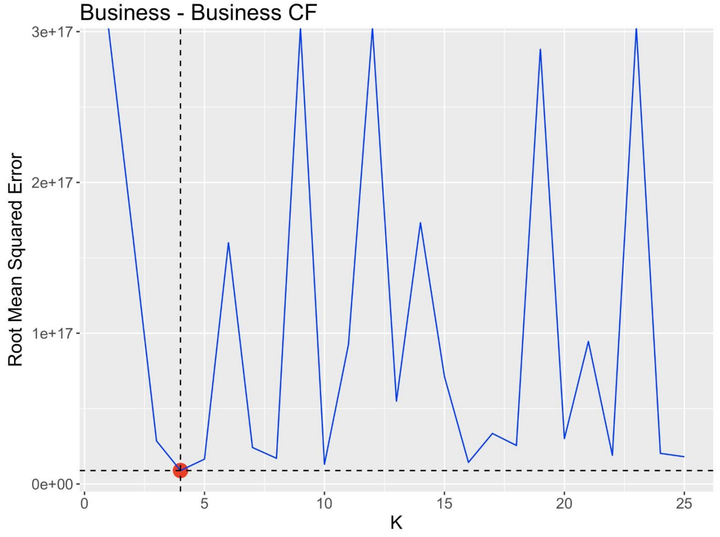




MAE is least when K = 3 and RMSE is least when K = 2

Task 1: Item-Item Memory-based CF using Pearson Similarity





From the plots we can conclude that at K = 4 we obtain the least error

Model Comparison

Content Based Recommendation

- **Advantages:**
  - The inverted index makes the query processing and evaluation very fast
- **Disadvantages:**
  - It is not very easy to explain
  - Suffers from cold start problem as the index entries as we do not have any information about the user.

Collaborative Filtering

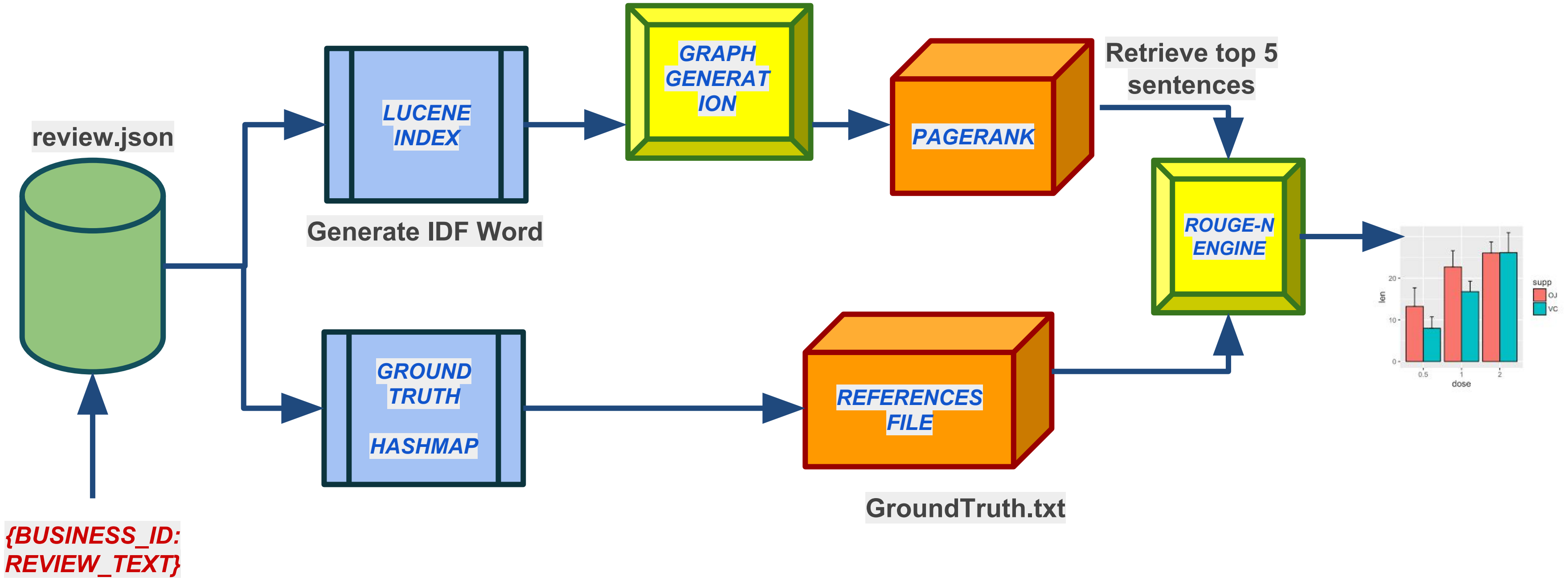
- **Advantages:**
  - The results are easy to explain.
  - Addresses cold start problem.
- **Disadvantages:**
  - Suffer from data sparseness and has a long tail with a good proportion of unique User-Business combinations
  - Because it is and online learning algorithm there are some scalability issues

Task 2

Task 2 : Text Summarization using Lexrank

- Goal
  - To produce a summary of a business reviews.
- Significance
  - Help owners understand the key points affecting their business.
  - Save time by helping users get a compressed version of the reviews in 4-5 sentences.
  - Lexrank being a extractive summarization technique, easier than Natural Language Generation.

Task 2 : Algorithm Workflow



Task 2 : Inverse Document Frequency and Sentence Similarity

- Inverse Document Frequency of a Word

$$\mathrm{idf}_i = \log\left(\frac{N}{n_i}\right)$$

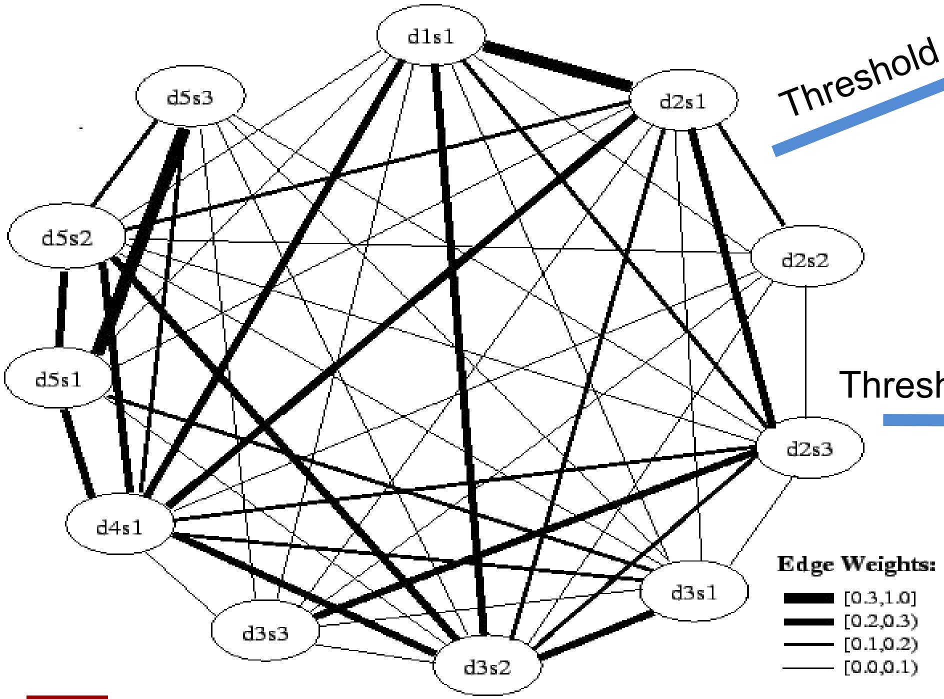
N = Total number of documents
N_i = Number of documents containing the word i.

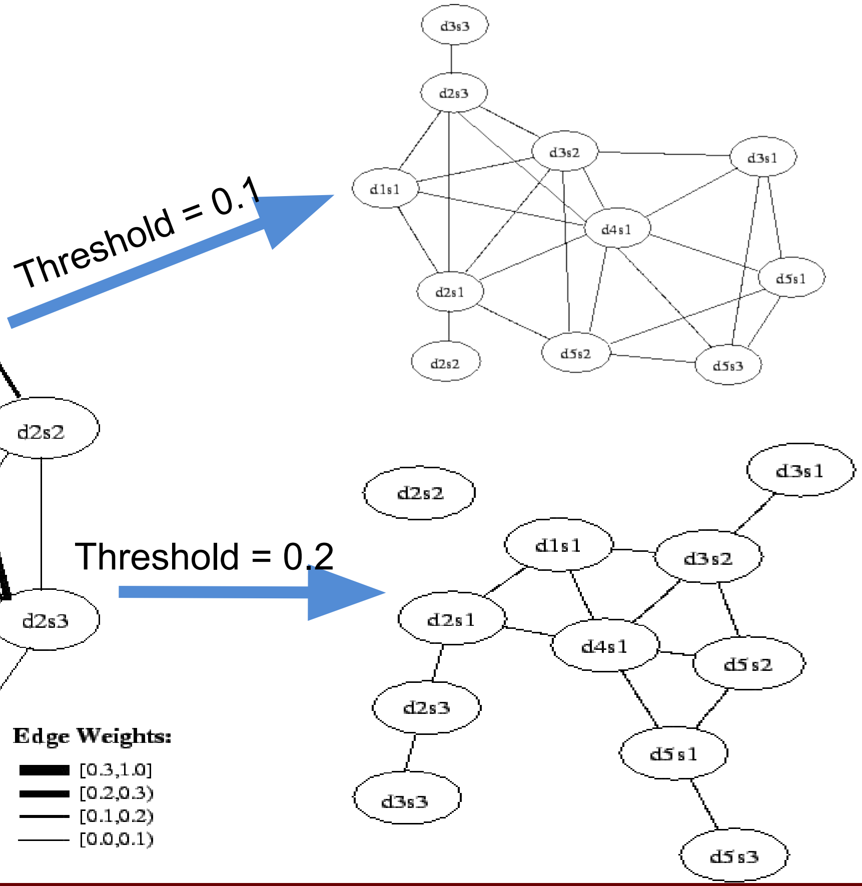
- Similarity Measure between two sentences using Bag of Words and Cosine Similarity.

$$\text{idf-modified-cosine}(x, y) = \frac{\sum_{w \in x,y} \mathrm{tf}_{w,x} \mathrm{tf}_{w,y} (\mathrm{idf}_w)^2}{\sqrt{\sum_{x_i \in x} (\mathrm{tf}_{x_i,x} \mathrm{idf}_{x_i})^2} \times \sqrt{\sum_{y_i \in y} (\mathrm{tf}_{y_i,y} \mathrm{idf}_{y_i})^2}}$$

$Tf_{w,s}$ = Number of times Word w occurs in the Sentence s

Task 2 : Graph Creation





Task 2 : Lexical Pagerank or Lexrank

- We know that PageRank is originally given by

$$p(u) = \frac{d}{N} + (1-d) \sum_{v \in adj[u]} \frac{p(v)}{deg(v)}$$

d = Damping factor
N = total number of nodes in the graph

Task 2 : Lexical Pagerank or Lexrank

- ➢ Graph is symmetric, unlike the original Pagerank
- ➢ Create a idf-modified-cosine similarity Matrix
- ➢ Set weight[i][j] = 1 if weight[i][j] > threshold
- ➢ Normalize using rowsums.

```
MInputAn array S of n sentences, cosine threshold t  output: An array L of LexRank
        scores
Array CosineMatrix[n][n];
Array Degree[n];
Array L[n];
for i ← 1 to n do
    for j ← 1 to n do
        CosineMatrix[i][j] = idf-modified-cosine(S[i],S[j]);
        if CosineMatrix[i][j] > t then
            CosineMatrix[i][j] = 1;
            Degree[i] + +;
        end
        else
            CosineMatrix[i][j] = 0;
        end
    end
end
for i ← 1 to n do
    for j ← 1 to n do
        CosineMatrix[i][j] = CosineMatrix[i][j]/Degree[i];
    end
end
L = PowerMethod(CosineMatrix,n,ε);
return L;
```

Algorithm 3: Computing LexRank scores.

Idf-Modified-Cosine → **Cosine Matrix**

$$
\begin{array}{c|ccc}
 & 1 & 2 & 3 \\
1 & 1.0 & 0.04 & 0.15 \\
2 & 0.04 & 1.0 & 0.07 \\
3 & 0.15 & 0.07 & 1.0
\end{array}
\rightarrow
\begin{array}{c|ccc}
 & 1 & 2 & 3 \\
1 & 1 & 0 & 1 \\
2 & 0 & 1 & 0 \\
3 & 1 & 0 & 1
\end{array}
\rightarrow
\begin{array}{c|ccc}
 & 1 & 2 & 3 \\
1 & 1/2 & 0 & 1/2 \\
2 & 0 & 1 & 0 \\
3 & 1/2 & 0 & 1/2
\end{array}
$$

Task 2 : Continuous Lexrank

- Uses Weighted Graph.
- Does not perform Binarization.
- Uses strength of similarity links.

$$p(u) = \frac{d}{N} + (1-d) \sum_{v \in adj[u]} \frac{\text{idf-modified-cosine}(u,v)}{\sum_{z \in adj[v]} \text{idf-modified-cosine}(z,v)} p(v)$$

Task 2 : Experiment

➢ Data

- Ten businesses with most number of reviews.
- All reviews of a particular business used as a Text file

➢ Indexing

- Consider each sentence as a Document.
- Index using Lucene.
- Get Inverse Document Frequency of each word in all the documents.
- Get Term Frequency Vector of each sentence/Document.

Task 2 : Experiment

➢ Graph Creation

- Created a Graph and Adjacency Matrix using JUNG, assigned Edge Weights.
- Calculate ranking score of each sentence using Lexical Pagerank or Continuous Lexrank.

➢ Results and Evaluation

- Sort the sentences in descending order of their scores.
- Retrieve top five results as review summarization.
- Used Rouge-N to evaluate results.
- Created Ground Truth by selecting important sentences from most useful reviews.

Task 2: Rouge-N Evaluation Metric

- N-Gram evaluation models
  - Unigram
  - Bigram

- Recall

$$\frac{number\_of\_overlapping\_words}{total\_words\_in\_reference\_summary}$$

- Precision

$$\frac{number\_of\_overlapping\_words}{total\_words\_in\_system\_summary}$$

Bi-Gram Model

```
the cat,
cat was,
was found,
found under,
under the,
the bed
```

Task 2: Evaluation Results using Unigram Model

	Continuous Lexrank			Lexrank with Threshold 0.05			Lexrank with Threshold 0.1		
Task Name	Avg_Recall	Avg_Precision	Avg_F-Score	Avg_Recall	Avg_Precision	Avg_F-Score	Avg_Recall	Avg_Precision	Avg_F-Score
1	0.16854	0.17045	0.16949	0.16854	0.15957	0.16393	0.19101	0.28814	0.22973
2	0.26829	0.352	0.3045	0.26829	0.34646	0.30241	0.2378	0.375	0.29104
3	0.27378	0.52486	0.35985	0.26225	0.4715	0.33704	0.27378	0.60897	0.37773
4	0.14524	0.38854	0.21144	0.14762	0.35632	0.20875	0.13571	0.42857	0.20615
5	0.29221	0.47872	0.3629	0.4513	0.49643	0.47279	0.01299	0.07843	0.02228
6	0.35079	0.29911	0.32289	0.35079	0.27801	0.31019	0.49738	0.4185	0.45455
7	0.32584	0.56863	0.41429	0.32584	0.56863	0.41429	0.30712	0.53947	0.39141
8	0.23497	0.27564	0.25369	0.39891	0.365	0.3812	0.07104	0.11607	0.08814
9	0.20879	0.21591	0.21229	0.03297	0.03797	0.03529	0.01099	0.02564	0.01538
10	0.61503	0.69409	0.65217	0.61731	0.69309	0.65301	0.52847	0.67836	0.59411

Task 2: Evaluation Results using Trigram Model

Task Name	System Name	Avg_Recall			Avg_Precision			Avg_F-Score		
		Continuous	Threshold 0.05	Threshold 0.1	Continuous	Threshold 0.05	Threshold 0.1	Continuous	Threshold 0.05	Threshold 0.1
	1CONT.TXT	0.15029	0.15029	0.16185	0.15385	0.14365	0.25	0.15205	0.14689	0.19649
	2CONT.TXT	0.22152	0.22152	0.20253	0.29661	0.29167	0.32653	0.25362	0.2518	0.25
	3CONT.TXT	0.25074	0.24779	0.00885	0.48571	0.4492	0.04545	0.33074	0.31939	0.01481
	4CONT.TXT	0.11622	0.11622	0.11622	0.32	0.28743	0.38095	0.17052	0.16552	0.17811
	5CONT.TXT	0.27152	0.41722	0	0.45055	0.45985	0	0.33884	0.4375	0
	6CONT.TXT	0.30978	0.30978	0.4837	0.26512	0.24569	0.40639	0.28571	0.27404	0.44169
	7CONT.TXT	0.29231	0.29231	0.27692	0.51701	0.51701	0.49315	0.37346	0.37346	0.35468
	8CONT.TXT	0.18539	0.37079	0	0.22	0.34021	0	0.20122	0.35484	0
	9COUNT.TXT	0.18644	0	0	0.19412	0	0	0.1902	0	0
	10COUNT.TXT	0.60739	0.61201	0.51732	0.68848	0.6901	0.66667	0.6454	0.64871	0.58257

Task 2:Example Results

While I had had a long day of travel and would have eaten just about anything by that time, I thoroughly enjoyed my genoa salami sandwich (though I wish they would have stuffed a little m
The addition of the fries and Cole slaw on top of the sandwich will make any of the offerings at Primanti a delicious and unique flavor of Pittsburgh.
I can see why people come here....the prices are great and the sandwiches are huge....after the bf saw Primanti Bros on one of the food networks, he made it a point to have a sandwich her
Honestly the burger or per Mantee sandwich it's self was lacking in flavor but put together quite well from the very nice bread to the well cook fries the slaw and I got the corn beef sty
The sandwiches come with meat of your choice, cheese, french fries, and coleslaw, and if you want, an egg.
Sandwich starts with thick slice of homemade bread, then the usual suspects of meats, cheeses, french fries are put on top, and then sweet & sour slaw is slung on top of that, and finally
My first prim bros experience was at this location fresh baked Cibrones bread with HotSausage Prov fries and slaw good lord Prim Bros is a staple in Pittsburgh and if you can't make it to
A nice combo of pastrami & corned beef, a layer of good cole slaw and topped off with some golden fries makes for one tasty sandwich.

The sandwich came out with fries on it!?!?
We had a great time, had great service, and loved the food.
I had an steak sandwich, greasy but delicious and with the fries and slaw are on the sandwich!
It was definitely great to try but I didn't enjoy the mess and the sandwich was a bit bland.
They are famous for large sandwiches with the fries IN the sandwich, not on the side.

Task 2: Limitations

- Whole sentences extracted.
- No named entity recognition/lemmatization used.
- Groundtruth formation could be better.
- No specific amount of summary generation w.r.t. length of original review content.
- Graph implementation could be made more efficient.

Future Work

- Task - 1
  - Ground truth creation needs to be handled in a better way.
  - Efficient way for parsing and lemmatization of the json input needs to be implemented.
  - Try Matrix decomposition approach to make the system more scalable.
- Task - 2
  - Abstractive summarization using deep neural nets.
  - Aiding lexRank with machine learning to incorporate feedback from Rouge.
  - Using NLP to get semantic similarity of sentences(and not statistical)