Metasearch Engine

A search engine combining the results of 5 other search engines

Group No: 3

Himanshu Aggarwal (MT17015)

Ojasvi Aggarwal (MT17033)

Prerna Kalla (MT17042)

Urvashi Choudhary (MT 17062)

Aim

- **Query 5 search engines simultaneously.**
- **Rank the retrieved results using our ranking model.**
- **Annotate the results of our ranking model**
- **Use various metrics to evaluate our meta-search engine’s performance.**
- **Return best possible diverse results to user**

PHASE -1

- Fetch data from search engines.
- Develop a user interface.
- Get results simultaneously
- Ranking using
  - Tf-idf
  - Cosine similarity
  - WoT Score

SEARCH ENGINES USED

DuckDuckGo

FETCHING RESULTS

- Search engines used are
  - Baidu
  - Bing
  - DuckDuckGo
  - Google
  - Yahoo
- Simultaneously fetched results using a **multithreaded environment.**
- Number of results obtained: **top 10 from each.**

User Interface

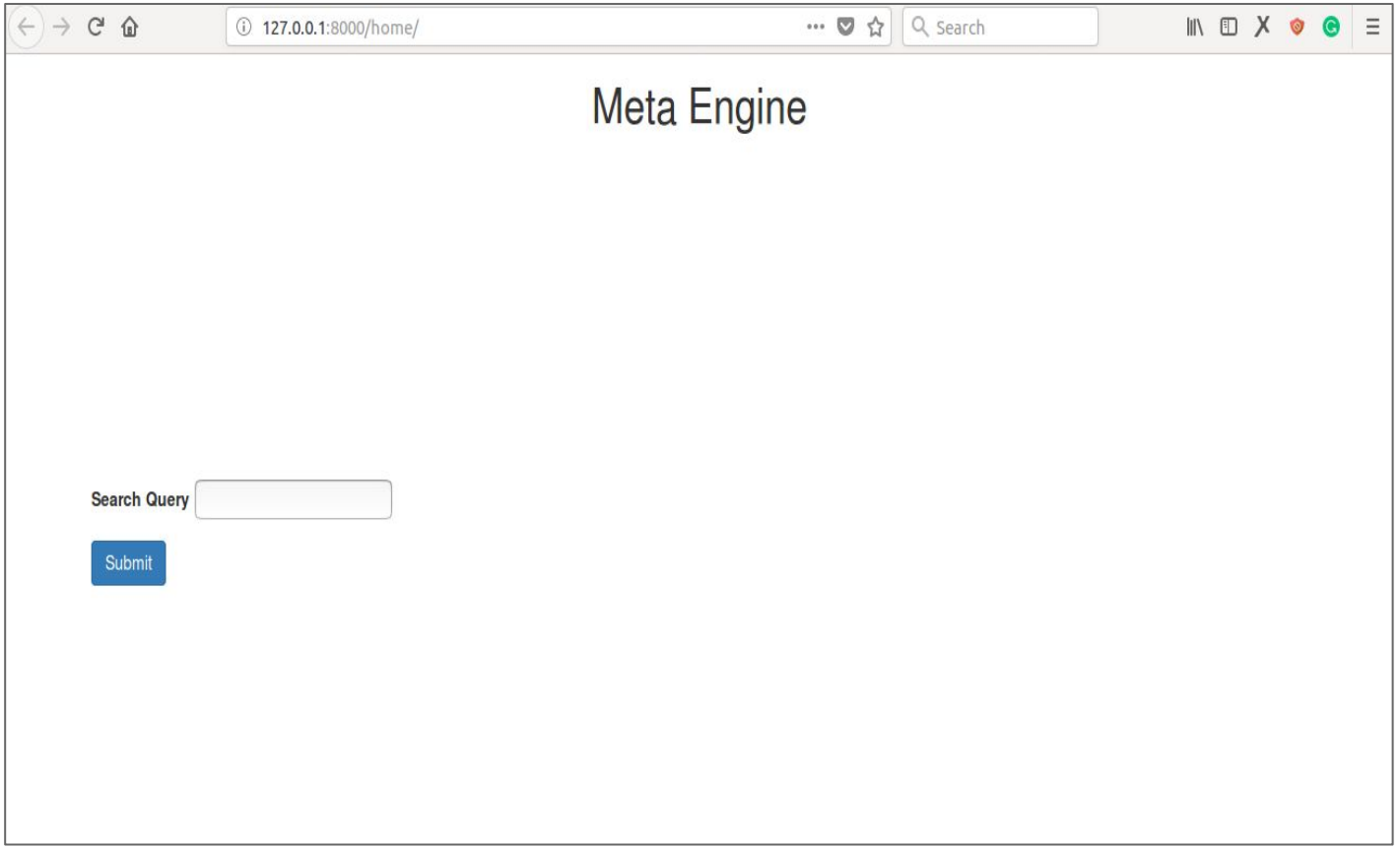
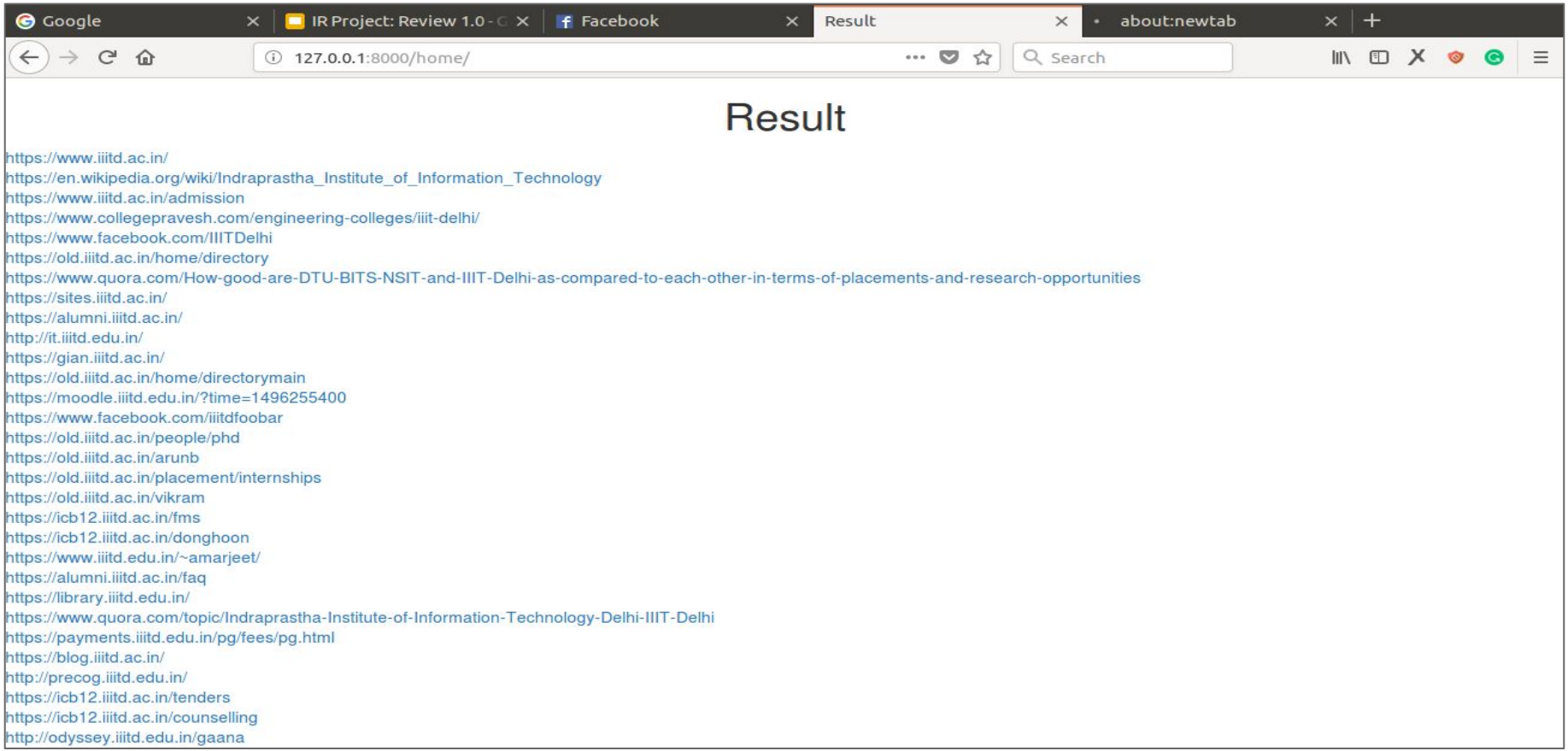


Fig (a) :Meta search engine UI



Figure(b): Results for the query: “ IIIT Delhi”

Ranking of results

TF- IDF

- Parsed all links.
- Computed tf-idf of query terms and documents.
- Using Tf-idf documents were ranked.

Web of trust

- Find reputation and confidence score for each link
- Add it to the td-idf score to obtain net score
- Return results to user on the basis of decreasing net score

Ranking in Baseline Model

- **Fetch top 10 search results for each query from Google, Yahoo, BIng, duckduckgo and Baidu.**
- **Our Baseline model assigns a net score to each retrieved result**

  Net score = Tf_idf Score + WoT score

Web of trust : WOT is an online reputation and Internet safety service. It provides reputation and confidence score for a website.

Weblinks are returned to user in order of decreasing net score.

PHASE -2

- Create better models than our baseline model
- Perform ranking of 50 results using our models.
- Using metrics to evaluate our search engine.
- Use caching to obtain faster results.

Covered till now...

- **API retrieval**
- **Initial Ranking model**
- **Ranked result retrieval**
- **Annotated initial results**
- **Cashing**

Now onwards...

- **Well defined query domain**
- **Rank Aggregation**

- **Two new models**
  - **BORDA Ranking Model**
  - **Markov Ranking Model**

- **Evaluation Matrices**
  - **MAP**
  - **NDGC**
  - **Kendall Distance**

- **Some results...**

Query Domains

To test our ranking model, we chose queries over the following 5 domains -

- **Delhi Metro**
- **IIIT Delhi**
- **Aadhar Card**
- **Elections**
- **GST**

Each domain has 10 queries.

Hence each model is queried 50 times over 5 domains.

DELHI METRO
1.Delhi metro fare
2. Delhi metro map
3. Airport line metro timing
4. Timing of last metro
5. Lost and found in dmrc
6. Helpline number of Delhi Metro
7. Smart card online recharge
8. Is metro closed today
9. DMRC recruitment
10. Accidents in delhi metro

IIIT DELHI
1.How to get admission in IIIT- Delhi
2. Route to IIIT-D
3. IIIT Delhi cutoff
4. Research opportunities at IIITD
5. Courses available at IIITD
6. IIITD faculty
7. Placements at IIITD
8. Nearest metro station to IIITD
9. Ranking of IIIT Delhi
10. Campus life at IIIT Delhi

AADHAAR CARD
1. How is Aadhaar different from any other identity issued by the government?
2. Get aadhaar card
3. Update aadhaar details
4. Link contact with aadhaar
5. Suspended aadhaar
6. Required documents for aadhaar
7. Download aadhaar card
8. Linking aadhaar and pan
9. Secure fingerprint aadhaar
10. aadhaar centres

ELECTIONS
1. Election schedule
2. Voting centres
3. Apply voting card
4. Operating evm
5. Election day a holiday
6. Result day election
7. Rigged evms
8. Update voter card
9. Political parties
10. Required documents for voter card
11. Election Commission

GST
1. Benefits of gst
2. Drawbacks of gst
3. What is gst
4. Gst calculation rules
5. Categories of gst
6. Items unaffected from gst
7. How to register for gst
8. Relaxation in gst
9. Penalty/consequences in case of violation of gst
10. Remission under gst

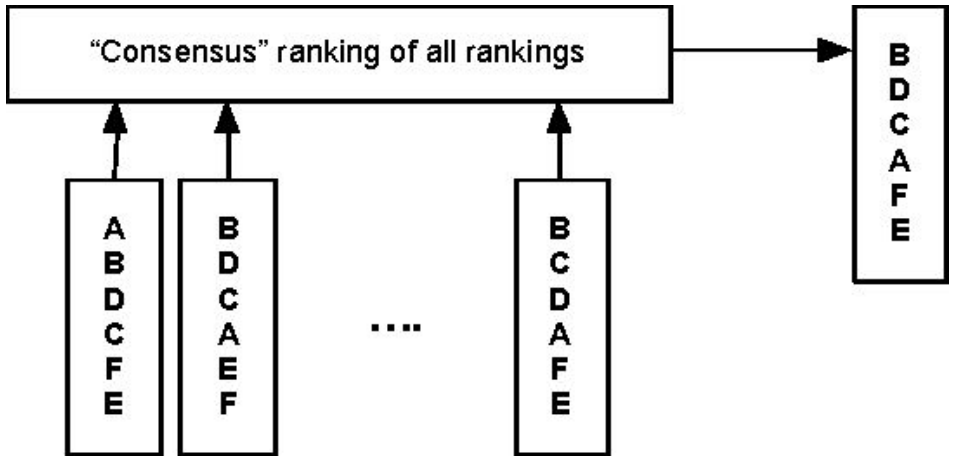
Key properties of Rank Aggregation

- **Non Dictatorship**
  - **No single competent to be preferred over others**
- **Pareto efficiency**
  - **If everyone competent prefers A to B, needs to be reflected in final result**
- **Independence of irrelevant alternatives**
  - **Irrelevant intermediate results from competents should be ignored**

Rank Aggregation

Purpose

- Input to multiple search engines
- Retrieve ranked results from these engines
- Taking into account, the preferences from these search engines



Heuristics for Rank Aggregation

- **BORDA Model**
  - **Weighted Indegree**
- **Markov Chain Model**
  - **Stationary Probabilities**

BORDA Ranking Model

- **Consensus based ranking model**
- **It tends to elect broadly rather than majority**
- **Single-winner election methods**
- **Voters rank candidates in order of preference**
- **Each candidate is allotted a ballot or a number of points**
- **Ballot determines overall ordering of candidates**

BORDA (our implementation)

- **Fetched results from google, bing, baidu, yahoo, ddg**
- **Results are retrieved in ranked order**
- **Results ordered in dictionary, along with ballot score**
- **Score of a link:**
  - **If returned as result, accumulated rank from that api**
  - **If not returned as result, rank considered as negligible**
- **Accumulated total ranks of all results**
- **Return sorted results on basis of ballot score**

BORDA (Tie Breaker)

- **Tie Case**

	Google	Bing	Baidu	DDG	Yahoo	Total
Link1	2	1	3	2	4	12
Link2	3	1	1	2	5	12

- **Tie breaker policy:**
  - **Compute tf-idf score**
  - **Use these scores for tie breaking**

BORDA (Example)

	Engine 1	Engine 2	Engine 3	Engine 4	Ballot Score	Aggregate Ballot Score
Link 1	1	2	1	2	6	1.5
Link 2	2	Not a result	2	3	7	2.33
Link 3	Not a result	1	3	1	5	1.67

Winner of voting is : Link 1

Markov Ranking Model

- **States correspond to ‘*n*’ candidates to be ranked.**
- **Transition probabilities depend on the given partial lists**
    - **Partial lists: Ranked results from all the search engines are partial lists.**

Intuition Behind Markov Model

- **For transition probabilities (i,j) pair wise ranking is needed.**
- **The ranking of partial list is used to infer the comparison outcome of (i,j)**
- **IS capable of handling uneven comparison**
    - ***If P is towards the bottom of 70% of pages and ranked as first in 30% of pages, markov chain model does not give importance to page P.***
- **Borda's underlying principle "*More wins is better*", Markov model says "*More wins against good candidates is better*".**
- **It is considerably fast.**

Markov Chain (Our implementation)

- **Steps for markov chain**
- **Get ranked results from all the search engines.**
- **Using the ranked results generate a transition matrix.**
- **Make this transition matrix ergodic , taking $\alpha = 0.15$**
- **Transition probabilities were calculated.**
- **The final steady probabilities define a natural ordering of elements and is known as Markov Chain ordering.**

KEMENY OPTIMAL AGGREGATION

Perfectly ordered List :

Noisy versions of perfect list, List 1 :

Noisy versions of perfect list, List 2 :

- **Kemeny optimal aggregation is the one that is maximally likely to produce the list that is closest to the perfect list.**
- **It has the property of removing noise from various different ranking schemes.**
- **Satisfies condorcet property and consistency in social choice.**
- **A condorcet winner is the one which wins all pairwise tournament. If a condorcet winner exists, then this model gives it the top rank.**
- **It is an NP hard problem**

Kemenization Heuristic Approach

- **Kenemization can be applied on the results of any rank aggregation algorithm to get better results.**
- **Let π1, π2, π3,... be partial lists (ranked results from search engines)**
- **Let μ = Results of some rank aggregation algorithm (Final result from any ranking model).**
- **Let π = The optimal list that will be created.**
- **Pick an element from μ and place it in the bottom of the list.**
- **Bubble it upwards, and a swap should happen or should not, depends upon the partial lists.**
- **The winner bubbles up to the top.**

Evaluation Schemes

- **Measure of performance of a system**
- **Evaluation measures used:**
  - **Mean Average Precision (Binary relevance)**
  - **Normalised Discounted Cumulative Gain (Ranked relevance)**
  - **Kendall Distance (Pairwise relevance)**

Evaluation Schemes

- **Annotators**
  - **Himanshu**
  - **Ojasvi**
  - **Prerna**
  - **Urvashi**

- **Tie cases:**
  - **2 annotators are in favour**
  - **2 annotators are in against**

- **Tie Breaker : Discussion by annotators**

Mean Average Precision

MAP:

- **Computes precision values for a retrieval system**
- **Precision values imply how early the relevant results are returned**

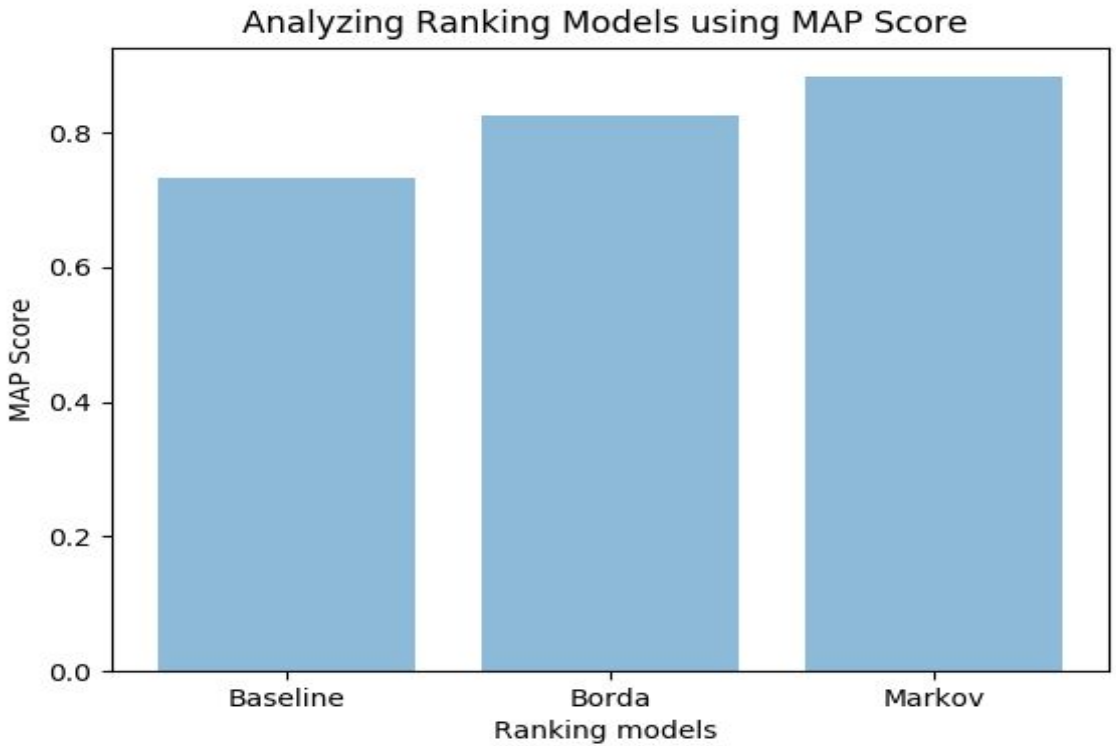
Steps:

- **Computes precision for all results of a query**
- **Computes average precision for a single query**
- **Computes average precision for a complete query domain**

Mean Average Precision Scores

Model	Initial Naive Model	Borda Model	Markov Chain Model
MAP Score	0.732575869237	0.824848875661	0.882938879441

MAP Scores



MAP score increases as ranking model improves.

NDGC Evaluation Scheme

- **It is designed for situations of non-binary notions of relevance**
- **The premise is that highly relevant documents appearing lower in a search result list should be penalized**

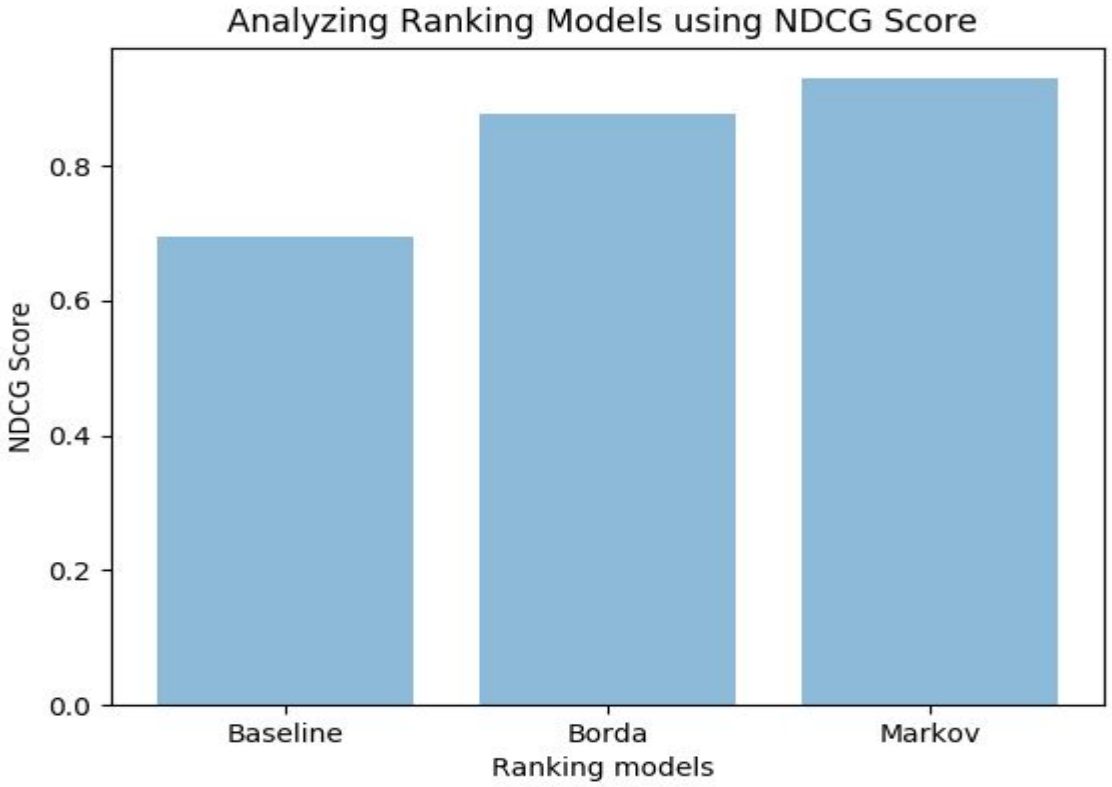
Steps -

- **Find the discounted cumulative gain for the documents**
- **Sort the list in order of decreasing relevance and find the ideal discounted cumulative gain**
- **DIvide the DGC by IDGC to get normalized discounted cumulative gain.**

NDGC Scores

Model	Initial Naive Model	Borda Model	Markov Chain Model
Normalized discounted cumulative gain score	**0.0.694085**	0.875721	**0.929309**

NDGC Scores



NDCG score increases as ranking model improves.

Kendall Distance Evaluation Scheme

- **It is finds the distance between two lists.**
- **This distance is the number of swaps required for list A to be converted to list B.**
- **It basically gives the % of disagreement between the lists**

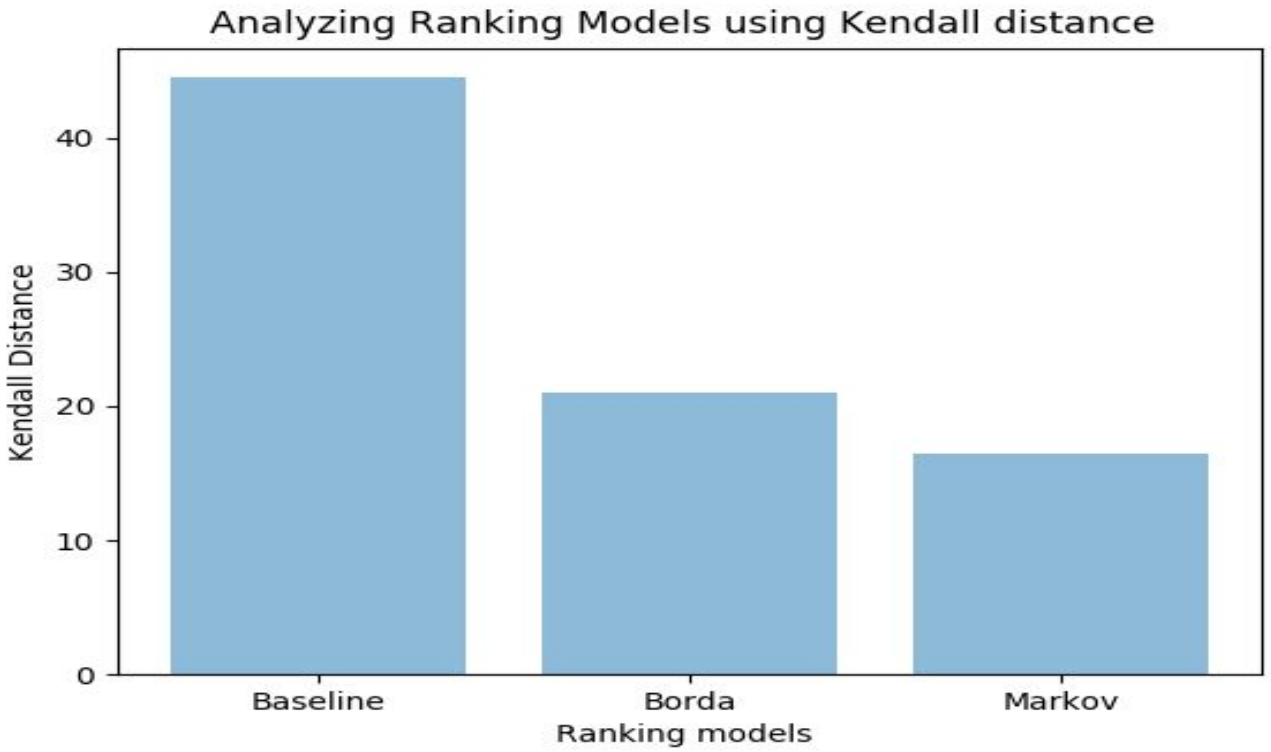
Steps -

- **We have pairwise annotations of 10 results for each query.**
- **Using this pair wise annotation and results of our model number of disagreements between the pairs were computed.**
- **The results were normalized by dividing it by 10C2.**
- **Average kendall distance was computed for all the queries.**

Kendall Distance Scores

Model	Initial Naive Model	Borda Model	Markov Chain Model
Kendall Distance (Percentage disagreement)	44.48%	20.97%	16.48%

Kendall Distance Scores



Kendall score decreases as ranking model improves.

Summary

	Naive Model	Borda Model	Markov Chain Model
Approach	Tf-idf + cosine similarity + WOT score	Ranking obtained from search engines + tf-idf	Ranking obtained from search engines + Kemenization
Map Score	**0.732575869237**	**0.824848875661**	**0.882938879441**
NDGC Score	**0.0.694085**	0.875721	**0.929309**
Kendall Distance (Percentage disagreement)	**44.48%**	20.97%	**16.48%**
Time	Slow	Relatively fast	Fastest

References

- **Dwork, C., Kumar, R., Naor, M., & Sivakumar, D. (2001, April). Rank aggregation methods for the web. In *Proceedings of the 10th international conference on World Wide Web* (pp. 613-622). ACM.**

- **https://people.orie.cornell.edu/dpw/talks/RankAggDec2012.pdf**

Thank you