IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

A BRIEF STUDY ON LTR AND USAGE OF ELASTICSEARCH FOR SEARCH RESULT RANKING

A comprehensive research study on Learning to Rank algorithms and the implementation of a data-driven search result ranking system using Elasticsearch platform

[1]A V Praveen Krishna, [2]Nithin Kalavagunta, [3]Sai Ganesh Katakam, [4]Sai Manoj Tekumalla
[1]Assoc. Professor, [2]Student, [3]Student, [4]Student
[1]Department of CSE,
[1]Koneru Lakshmaiah Education Foundation, Vaddeswaram, AP, India

Abstract : In this era of internet, information is a vital asset that forms the basis for almost every trade and constructs any meaningful interaction among various stakeholders. Tools like web browsers provide reliable sources of information to users who can find anything they want with minimal effort. A typical browser or information store uses a search engine to allow users to search for the information they need by entering relevant keywords, following which the engine generally returns a set of elements, each representing the information associated with a record in the server. Hence it is highly important to optimize the mechanism using which these engines return search results and the order in which the results are arranged, to provide satisfying user experience.

Different search result ranking techniques are used by search engines, distinguished by the ranking algorithm each one uses and its suitability to the data maintained by the engine. Learning to rank (LTR) is a class of such algorithms, implementing which involves the process of training a ranking machine or model using various approaches to rank the search results. This research paper includes the findings acquired as an outcome of a brief study that was conducted on LTR algorithmic approaches for search result ranking and an elaborate account of the Elasticsearch platform, which is a cloud-based search engine configuration tool. This tool was used to develop a search engine with a customised ranking mechanism for the project associated with this research work to display the understanding achieved during the study.

***Keywords* - Web Browsers, Search Engines, Ranking**

I. INTRODUCTION

Ranking is one of the primary areas that define the efficiency of a searching tool by analysing the effectiveness of a result return for a user query in the engine. This process is essentially composed of various techniques that define an order for the search results to be displayed based on the query entered by a user, considering a set of factors on the basis of which this prioritization needs to occur. An appropriate Ranking Methodology is a fundamental requirement for developing a search engine or any other search-oriented system for it lays a format for results to reach the users. Hence, selecting the most popular and desired ranking methodology is extremely important to promote the usage of the tool, failing which users would pursue more convenient options to fulfil their search requirements [2].

Data Driven Ranking is a widely used methodology for ranking search results in various search tools, which relies on the principle of establishing a ranking technique that takes into consideration the data elements associated with search results. The unadulterated motive behind using this technique is to obtain results that are more popular with users to make the engine more efficient in terms of the time taken by users to reach the result of their choice in case of a query. This objective leads to the exploration of numerous mechanisms, procedures, and algorithms required to operate on the search results to create a suitable ranking framework for the system. Amongst a host of techniques available in usage, one that caters to the need of the circumstance better than the rest is chosen to be applied.[1]

II. LITERATURE SURVEY

Extensive research and study were required to obtain an unambiguous understanding of the subject matter that would prove to be effective in devising an approach for the task of ranking. Dealing with heavily enhanced branches of Computer Science and inheriting several key features of the concepts in these areas, the project domain necessitates an elaborate study of not only the core tenets of the fields but also the resources available to work on their practical facets. This study developed an approach to deriving a solution to the challenge in question by providing a plethora of perspectives to proceed with the project. From conceptual guides to research papers, various resources have been used to conduct the study as diversity in material would offer a deeper insight.

The foundational step to conducting the research is identifying the sphere this body of work firmly belongs to, for technically affluent sources available in it to be pursued. This was where the project presented an initial challenge as it needed implements and knowledge encompassing multiple fields, which prevented it from being identified by a unique discipline. However, the core idea of the statement allowed a deduction of its relevance to the concepts of Machine Learning. Fundamental principles of this area were thoroughly studied and the vital components to be instilled in the project were identified at a rudimentary level. The transformative perception that the major real-time application of this project could revolve around this field, cemented the further objectives to be fulfilled, as a devised research plan displayed certainty in progressing the project [1].

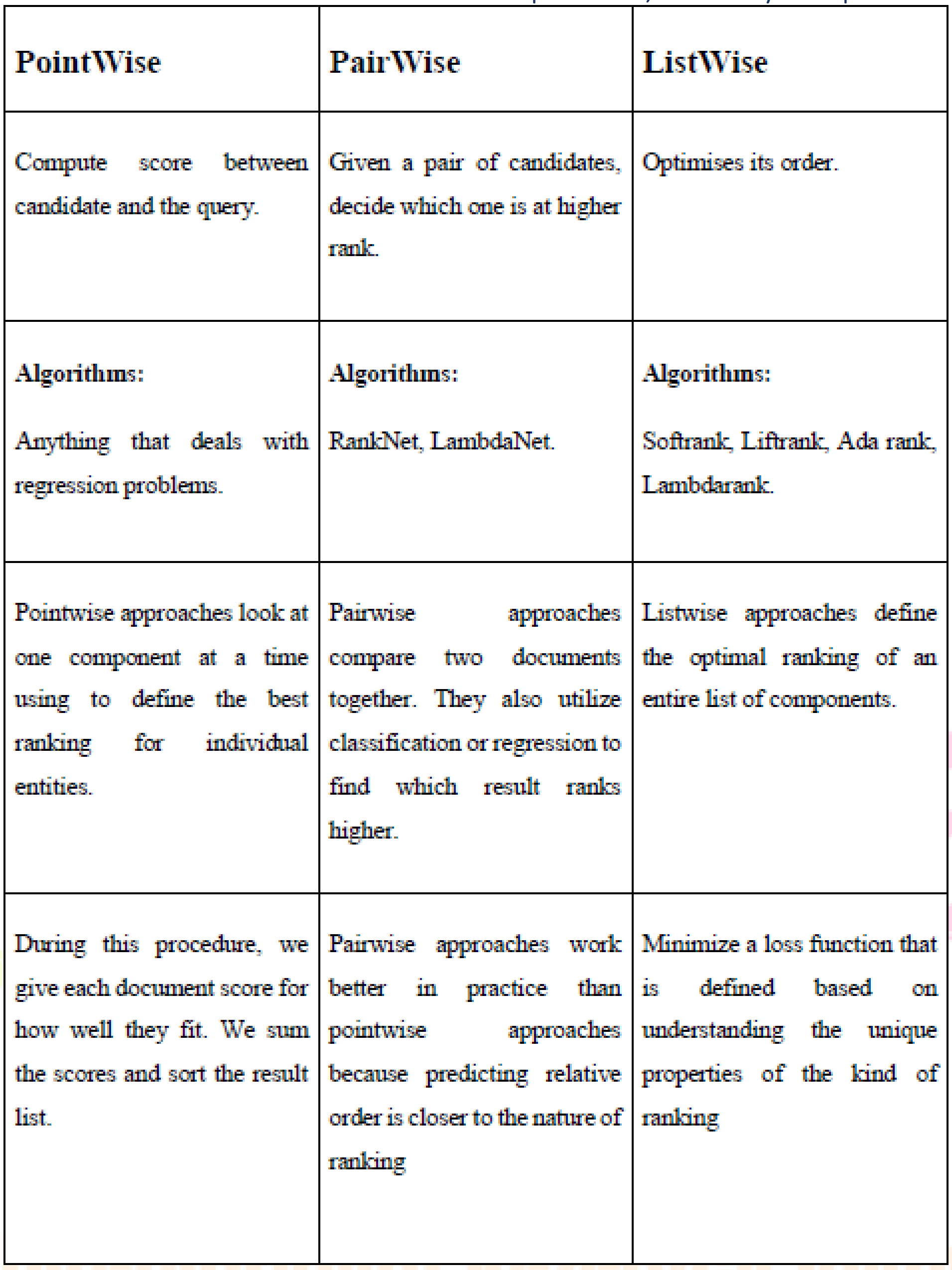
The ability in using Python was conceived to assume a significant role in furthering our work as the language is heavily used in most of the applications developed using this technology. Various libraries in Python were keenly studied for their features and capabilities, as their contribution to the design and development of the solution can greatly impact its effectiveness. Taking the next step in this process of finding the potential tools that would be utilized in practically implementing the solution, we proceeded to explore Elasticsearch, which is a search-engine oriented software service that enables developers to create engines to meet their requirements in terms of formatting the results, ranking patterns, user interface, and data integration [3]. It is a powerful tool comprising numerous services that redefine the search experience for users and enrich the process of development. Along with Elasticsearch, exist the widely used Kibana and Logstash, forming the ELK Stack. Kibana is a data visualization tool that has the capacity to handle enormous datasets and operate on them to visualize the information appropriately for better comprehension and analysis. Logstash is a data processing pipeline that accepts data from multiple sources, transforms the data, and sends it to a stash. Further application of ELK Stack is discussed in the following sections.

Various research papers have been referred to obtain established and accurate information regarding the project prior to the commencement of implementation as part of the research. This study welcomed many possible approaches with a host of ideas to incorporate into the implementation. Much of the study revolved around the mechanisms adopted by search engines to effectively rank the search results and the algorithms available to define the ranking processes. Of all the options researched and the feasibility of their implementation analysed, pursuing the features of Learning to Rank (LTR) algorithms was marked as an ideal approach to this project, as it is a vastly acknowledged method in affairs involving ranking decisions. This class of algorithms is used to ingrain a fixed mechanism of ranking in the search tool based on which the search results take into consideration specific characteristics of the components to be returned and finalize the order of their display by evaluating the factors with respect to the particular requirement posed. Factors promoted by statistics, popularity, and usability are generally claimed to bring an efficient return to a search result [2]. It makes use of Supervised Machine Learning to train the system to rank search results for a query.

These algorithms are incorporated into the searching tools and are brought into effect through errorless deployment following which results are ranked in accordance with the conditions specified in their corresponding algorithm. Dynamic implementation of such algorithms provides the feasibility for developers to modify the criteria of ranking the search results, thereby enabling the users to browse the contents in a flexible environment. It is through such measures that search systems achieve success in terms of user experience, which is the most important ambition any entity invests in the development of an engine. A few of the basic requirements of using LTR algorithms are obtaining training data, analysing the data, and defining functions to be assigned to the sections of data in order for the system to be trained.

However, the Learning to Rank algorithm class is not limited to a single mode of working but is a collection of multiple approaches, each having its own specifications, functionalities, advantages, and demerits [4]. Each of these methods garners applications that suit their properties and are considered optimal depending on the nature of the body of work that requires their implementation. For that reason, it is important to study and analyse the 3 major approaches to Learning to Rank, mentioned as follows:

- Pointwise Approach
- Pairwise Approach
- Listwise Approach



PointWise	PairWise	ListWise
Compute score between candidate and the query.	Given a pair of candidates, decide which one is at higher rank.	Optimises its order.
Algorithms: Anything that deals with regression problems.	**Algorithms:** RankNet, LambdaNet.	**Algorithms:** Softrank, Liftrank, Ada rank, Lambdarank.
Pointwise approaches look at one component at a time using to define the best ranking for individual entities.	Pairwise approaches compare two documents together. They also utilize classification or regression to find which result ranks higher.	Listwise approaches define the optimal ranking of an entire list of components.
During this procedure, we give each document score for how well they fit. We sum the scores and sort the result list.	Pairwise approaches work better in practice than pointwise approaches because predicting relative order is closer to the nature of ranking	Minimize a loss function that is defined based on understanding the unique properties of the kind of ranking

Fig. 1: Analysis of the learning obtained through research on various ranking approaches

2.1 Pointwise Approach

Pointwise approach is a single-component concentrated approach that uses classification or regression to determine the function value of the component or point. Classification involves grouping the components based on properties while regression implies assigning similar functional value to similar components to offer equivalent priority. The function value of each point is independent of the others.

2.2 Pairwise Approach

Pairwise approach involves the comparison of components in pairs using classification or regression. This method contrasts a pair of documents with a standard known as ground truth in order to arrange them to match the ranking. Minimizing incorrect ranking is the major objective of this approach.

2.3 Listwise Approach

Listwise approach determines the ranking order for an entire list of documents in reference to a ground truth list that is already created. This format is mainly focused on arranging a set of components in the desired order to optimize the process of ranking. They make use of probability models to minimize errors in ranking.

While LTR is a class of algorithms that is considered a standard for an algorithm used for ranking search results to abide by, many renditions of the principle have been designed by various entities with unique individual properties possessed by each of the mechanisms. Microsoft developed LTR algorithms such as RankNet, LambdaRank, and LambdaMart which are widely usedin the industry for projects that are driven by objectives suiting the capabilities of these algorithms. Some studies consider that LTR is not an algorithm itself but is a ranking or classification technique that makes use of Support Vector Machine (SVM). With diverse crucial applications in the industry, LTR algorithmsplay a key role in shaping effective solutions [4].

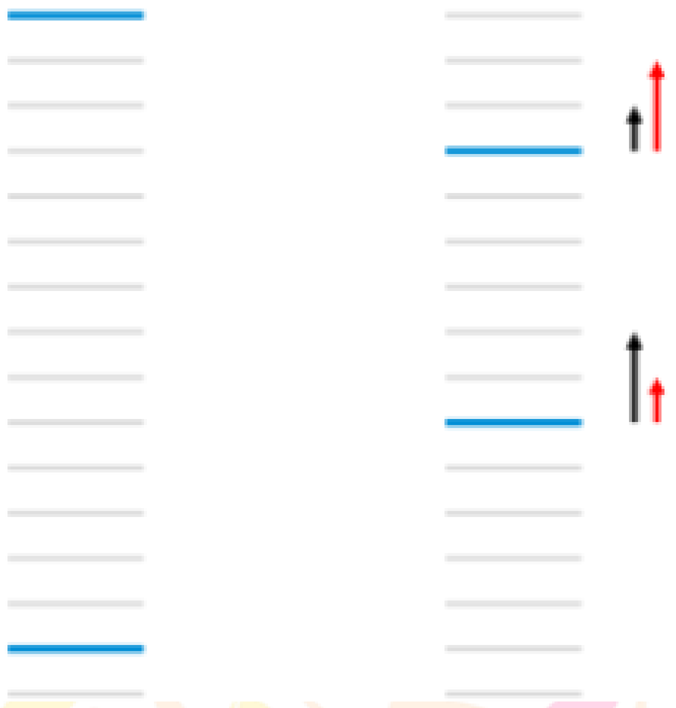


Fig. 2: An illustration of the mechanism of the mentioned ranking algorithm approaches – Pointwise, Pairwise, Listwise (left to right)

III. METHODOLOGY

The initial steps of working on the project objectives were understanding the problem statement thoroughly and identifying the necessary requirements or research areas to obtain knowledge regarding the worklet. After recognizing the sources of learning, we procured information concerning the project and the domain it predominantly belongs to. This procedure was followed by finding potential approaches to the principle task by enhancing the perspective acquired during the initial stages regarding the project, after exploring the field of the subject matter of the worklet domain. In the process of considering various ideas with the aim of reaching an ideal approach, the necessary tools and components required to be used to operate on resources related to the worklet are identified, and requisite arrangements to establish those tools in the local system and a cloud deployment were made.

Continuing this process, we obtained a huge dataset containing search results from a popular app store service hosting many applications with an array of properties acting as factors that enable us to rank them with respect to certain requirements. After acquiring this dataset, it has been extensively analysed with the aim of understanding the data and becoming familiar with its nature in order to easily identify existing patterns and make informative deductions. Using Python Libraries and other data visualization and analytics tools effectively is crucial to obtain the best outcomes in this step. The results extracted through this were studied in detail for inferring any peculiar pieces of information that could aid in formulating an approach for this project.

Once the analysis process was finished, the next immediate action was to use the Elasticsearch platform to create a cloud deployment and generate a search engine based on the preferences we decided to include in developing the engine. Using the multitude of tools available, an interactive search engine was developed with the option to tune the relevance score of each result accordingly for the experience to be more user-centric. An accurately functioning UI was also designed on the platform which can be modified according to our requirements.

The aim of this tool is to provide an interface for the users and service providers to exercise a data-driven search engine model that allows itself to be re-configured in accordance with the relevance of the attributes of data supporting the search operation. This step culminated the application part of this study, as we could arrive at a simple and modern approach to identifying a model search engine that implements ranking in the desired fashion, bypassing the complexity of configuring an engine on the basis of a selected ranking algorithm, while providing the flexibility of continuous modification and deployment in real-time. During this study, the focus was also directed towards enhancing our knowledge and ability in using LTR Algorithms by extending the conducted research study and comprehending the classification of the algorithms to analyse which type of them is suited for what kind of implementation need. After this process, as an optional extension to our work, the most suitable algorithm is to be identified for a search application and the requisite arrangements can be made accordingly for its implementation. Selection of a ranking algorithm to be used for a project is necessitated by the idea to develop a Machine Learning model as a solution for the problem statement. Hence, an appropriate algorithm must be identified to build the model that gives optimal results with maximum possible efficiency.

For the aforementioned process of configuring an Elasticsearch-oriented engine, the model needs training data from real-time app stores, which were obtained using an Automated Web Scraper that was written in Python with the Selenium framework. This scraping application acquires data from an app store and stores them in a local file as assigned to it. Using this data, a machine learning model can also be subsequently trained to analyse important factors in ranking search results based on user requirements and deliver effective outputs when deployed in realtime. With this, an alternate approach to the project can be achieved.

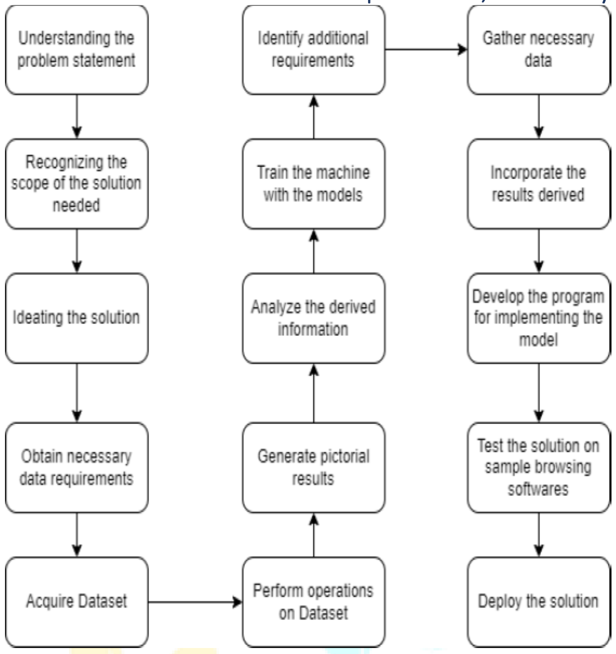

Understanding the problem statement
Recognizing the scope of the solution needed
Ideating the solution
Obtain necessary data requirements
Acquire Dataset
Perform operations on Dataset
Generate pictorial results
Analyze the derived information
Train the machine with the models
Identify additional requirements
Gather necessary data
Incorporate the results derived
Develop the program for implementing the model
Test the solution on sample browsing softwares
Deploy the solution

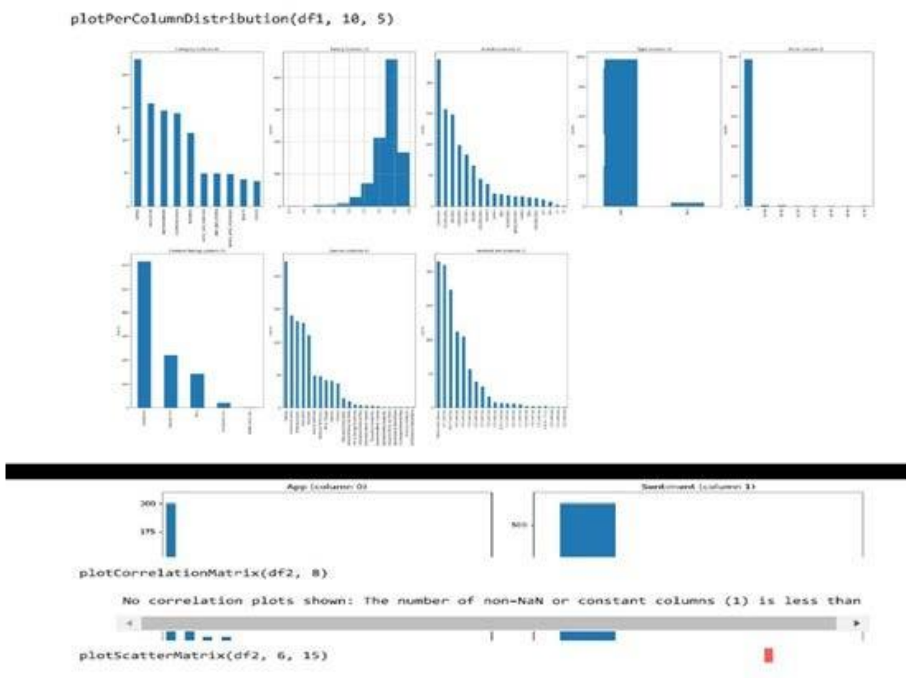
Fig. 3: A flow chart describing the workflow of the project and a solution approach of building an ML model

IV. EXPERIMENTAL RESULTS

Data are the fundamental requirement of any solution that is expected to deliver a result based on pre-configured information and instantaneous input. It is hence highly important for any development team to incorporate proper data analysis measures in the project life cycle in order to obtain adequate information regarding the mechanism of the system or user experiences. This data can be fetched from any relevant sources that bear familiarity with the body of work to a considerable extent, as this information is what provides the initial insights to the team. Only the inferences obtained through analysing this data coupled with knowledge of the subject matter lead the process to the subsequent stage of ideating a suitable solution. For this procedure to forbid any errors in the flow, a dataset that contains fields that offer extensive information about the objects and represent their particular characteristics either quantitatively or qualitatively should be utilised. The initial dataset for this project was obtained through multiple methods after a thorough scrutiny of what properties of data should be available to work with after extraction. Sources on the internet that are specialized in offering purpose-specific datasets that provide high analysis scope were identified and as a result, a large dataset of app store search results was extracted. This information was loaded into a spreadsheet with a huge number of rows present in it, which enhances the range of analytical procedures that could be carried out on it. The dataset essentially consisted of the search results returned by the search engine available on an application store with fields like App Name, Category, Price, Rating, Size, Installs, and Current Version among others [1].

This broad range of specifics regarding every application creates immense possibilities for analysis and a more flexible approach for ranking. The dataset was curated according to the results derived when certain keywords were entered in the engine. This enhancement was made with the aim of obtaining the most popular search results for a set of most used keywords. By developing an understanding based on this dataset, the process was to become simplified while paving the way for training the system with better results. Various data visualization & analytics techniques were used to obtain information from the data regarding many applicable statistics that aid in finding patterns, deriving conclusions, and making predictions. Python-based data analysis was rigorously conducted to achieve a holistic image of the data. Valuable inferences were recorded and evaluated to ideate a competent solution further. The insights obtained through this analysis contributed to the process of evaluating multiple approaches in the further steps.

An interface is necessary to provide a platform for users to interact with the search engine and obtain optimal results for their search queries. It needs to possess the basic elements of a responsive web page for users to interact with them and engage in the transaction. Elasticsearch platform was then utilized for making progress on this front by refining the process with a multitude of tools made available through it. It enables the developers to work on cloud deployment or stand-alone version of the platform. Both modes were utilized for the worklet to the best extent as the features offered by the tools are of immense applicability to the project. As the first step in adapting to this method of creating search engines, a cloud deployment was created with a selected service provider. Then a new engine was created in the deployment with a set of chosen specifics that enhance the outlook and experience of the search activity for users [3].



```
plotPerColumnDistribution(df1, 10, 5)
```

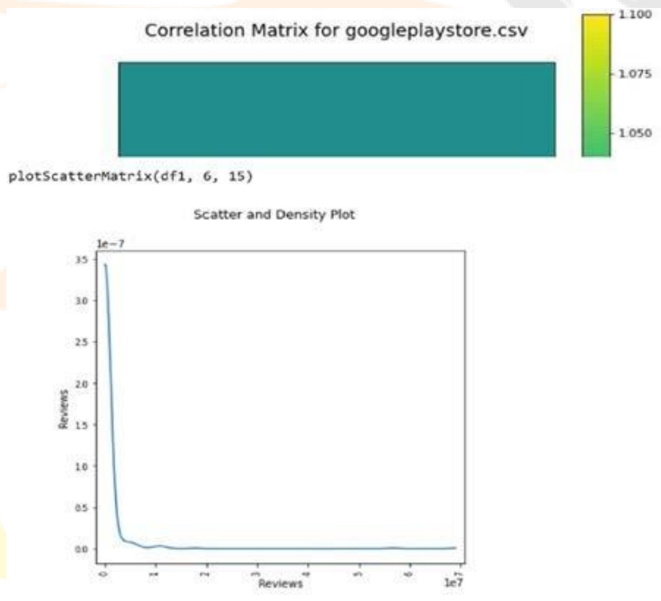
```
plotCorrelationMatrix(df2, 8)

    No correlation plots shown: The number of non-NaN or constant columns (1) is less than
```

```
plotScatterMatrix(df2, 6, 15)
```

Fig. 4: Bar Graphs obtained through data analysis with X-Axis containing different applications and Y-Axis displaying various key properties of characterizing them

As shown in figures 3.1 and 3.2, techniques like correlation matrix were used to study the relation between multiple attributes of the data. This helped us understand the popularity of certain applications and the search queries that lead to them.



```
plotScatterMatrix(df1, 6, 15)
```

Fig. 5: Correlation Matrix and Scatter & Density Plot obtained during analysis of reviews received by applications

To create a search engine, the dataset of results is to be loaded prior to any other design process for which the dataset obtained through the means mentioned earlier is utilized. This step can take place in multiple ways as the platform accepts datasets in various file formats and insertion modes. For the engine created as part of the project, preparation of the dataset commenced with a Web Scraping software application built using Python and Selenium framework [5]. An app store website was selected to perform scraping and the application was executed to derive search result data from the website. The data was curated to include search results related to the most used keywords so as to develop the solution model with diverse data. Finally, the dataset was uploaded as a .json format file with thousands of rows of information regarding the applications [2]. Once the dataset was uploaded, the schema of the dataset was altered in accordance with the requirement of the worklet, with the types of fields and values to be stored also configured in the same step. After defining the schema for the data attributes, we worked on modifying the relevance factors for various properties included in the schema. By adjusting these factors accordingly, the relevance score for search results is impacted. An account of the characteristics to be emphasized, as signified by the results, was made and the functions for those were given appropriate values. The process of relevance tuning eliminates results that appear in the primary portions of the search results without actual relevance to the search key, by effectively monitoring the relevance scoring.

After finalizing the relevance configuration for the dataset, the Search UI was generated. While incorporating the elements to be constituted in the search engine application, the necessary components related to operations on the results such as sorting and filtering were also included. With all the aspects of the UI configured, the engine was generated, and a working deliverable was issued. Bearing the majority of the required functional aspects, the search engine provides a varied set of essential features and enhances our search experience. After multiple trial runs of the deployment, some vital revisions were made in the search engine to enable users to have a flexible search facility. The search engine was tested using various results, and its ability to curate search results according to the provided query by taking into consideration the functional values of attributes defined during configuration process was primarily verified. Special emphasis was given to verifying whether the engine is capable of ranking the actual app documents ahead of its cloned versions.

Subsequently, the study was continued by organizing research on the LTR Algorithms, various approaches to using LTR, several real-time renditions of those approaches, and how to choose the suitable algorithm for further real-time applications of this worklet. To document the study as part of the project and to display the findings of the study, an interactive web page was created. This React-build web page includes a major portion of the knowledge acquired through our research, designed in the format of front-end for an application. This paves way for the body of work to be accomplished in continuation to this development.

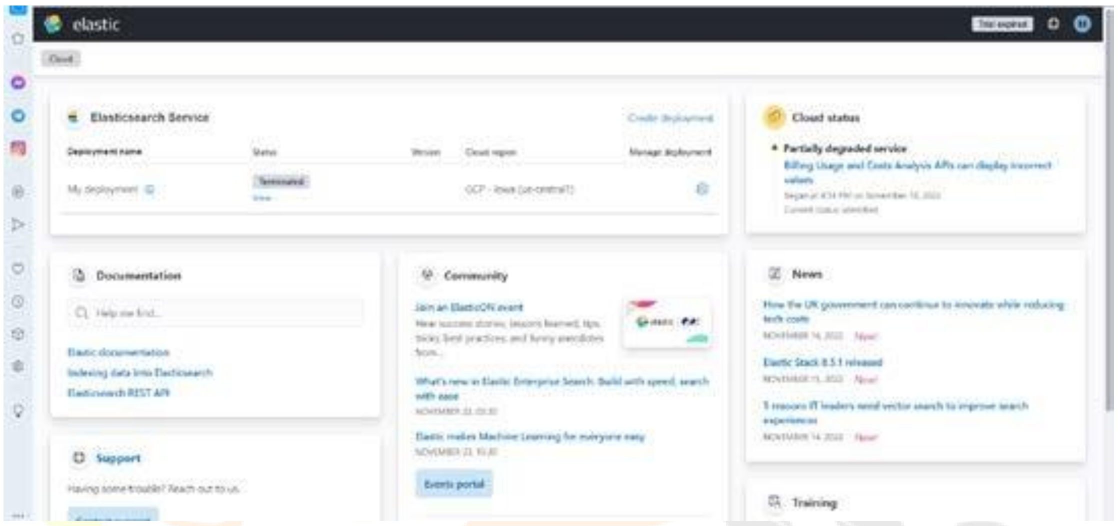
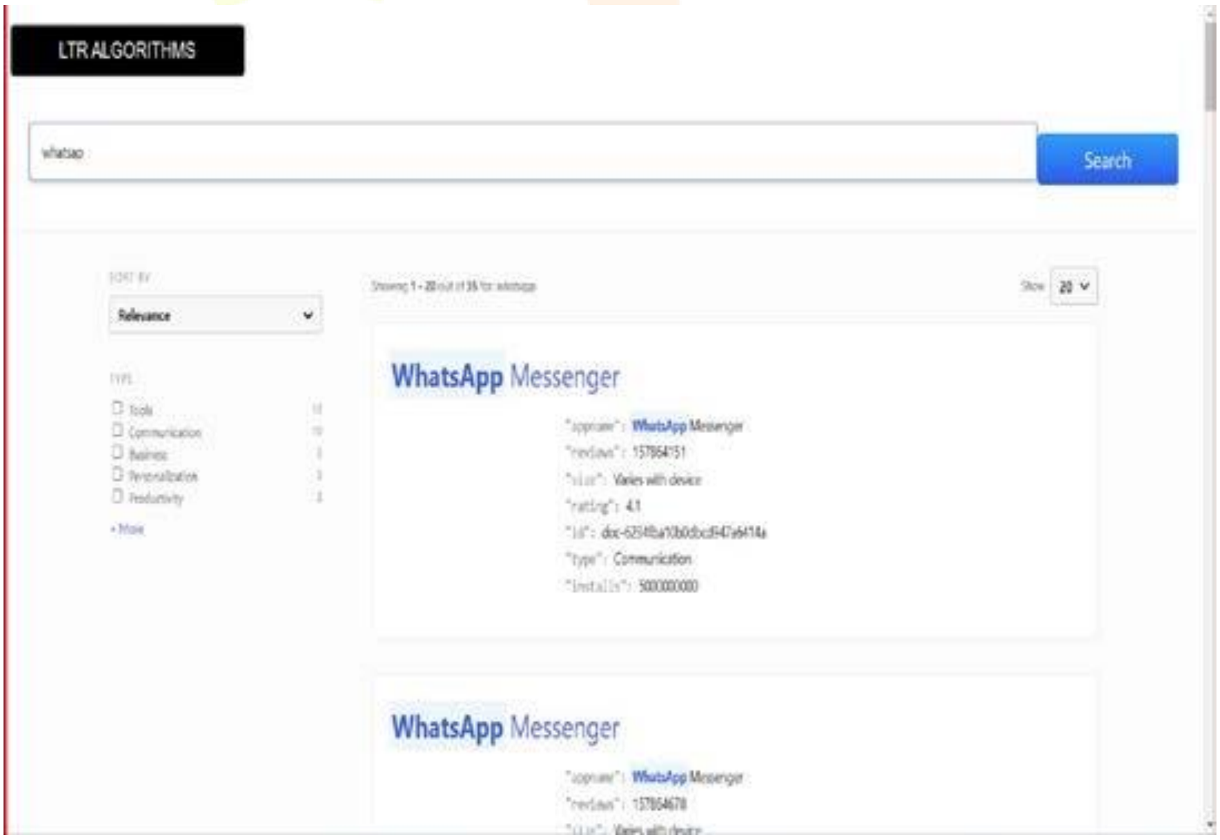


Fig. 6.1: Portal displaying the services available on Elasticsearch



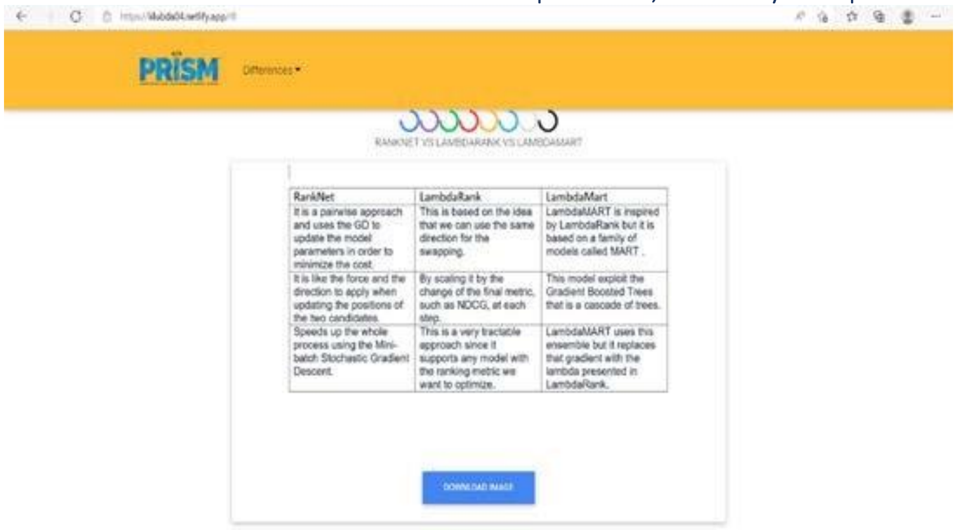
WhatsApp Messenger

"appname": WhatsApp Messenger
"reviews": 157864151
"size": Varies with device
"rating": 4.1
"id": doc-6254fba10b0dbcd947a6414a
"type": Communication
"installs": 5000000000

WhatsApp Messenger

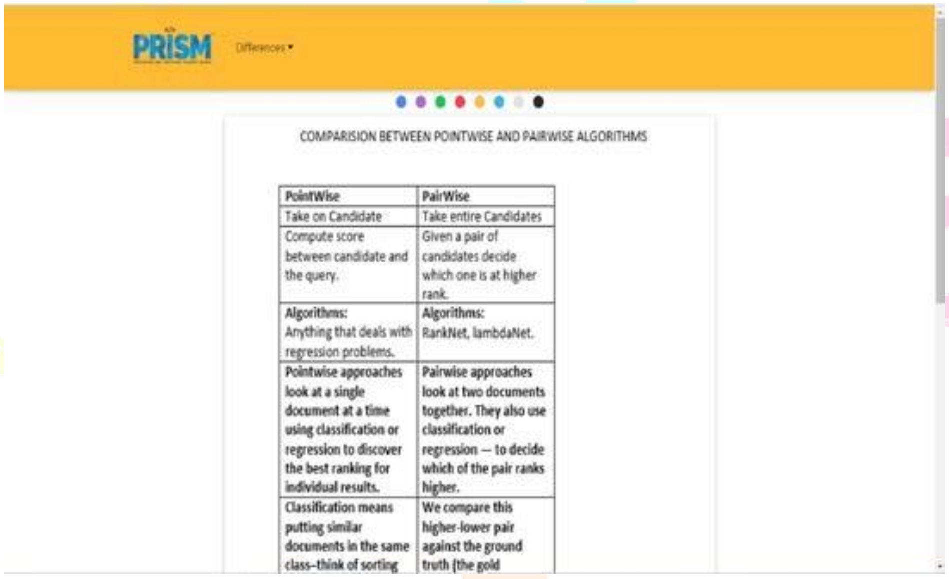
"appname": WhatsApp Messenger
"reviews": 157864678
"size": Varies with device

Fig. 6.2: Model Search Engine developed on Elasticsearch



RankNet	LambdaRank	LambdaMart
It is a pairwise approach and uses the GD to update the model parameters in order to minimize the cost.	This is based on the idea that we can use the same direction for the swapping.	LambdaMART is inspired by LambdaRank but it is based on a family of models called MART .
It is like the force and the direction to apply when updating the positions of the two candidates.	By scaling it by the change of the final metric, such as NDCG, at each step.	This model exploit the Gradient Boosted Trees that is a cascade of trees.
Speeds up the whole process using the Mini-batch Stochastic Gradient Descent.	This is a very tractable approach since it supports any model with the ranking metric we want to optimize.	LambdaMART uses this ensemble but it replaces that gradient with the lambda presented in LambdaRank.

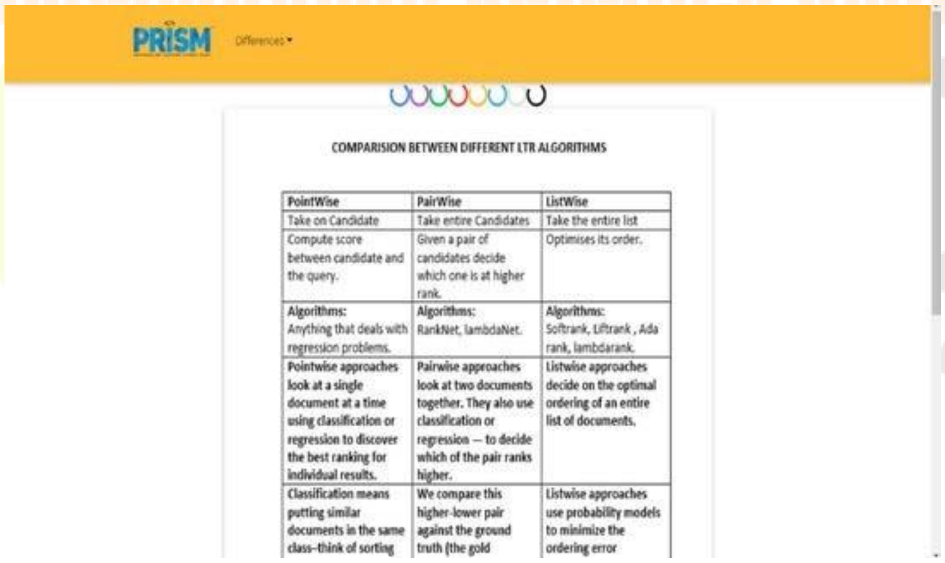
Fig. 6.3: An interface built using React.js displaying the comparison among Ranking algorithms



COMPARISION BETWEEN POINTWISE AND PAIRWISE ALGORITHMS

PointWise	PairWise
Take on Candidate	Take entire Candidates
Compute score between candidate and the query.	Given a pair of candidates decide which one is at higher rank.
Algorithms: Anything that deals with regression problems.	Algorithms: RankNet, lambdaNet.
Pointwise approaches look at a single document at a time using classification or regression to discover the best ranking for individual results.	Pairwise approaches look at two documents together. They also use classification or regression — to decide which of the pair ranks higher.
Classification means putting similar documents in the same class–think of sorting	We compare this higher-lower pair against the ground truth (the gold

Fig. 6.4: Findings of the research displayed on the user interface



COMPARISION BETWEEN DIFFERENT LTR ALGORITHMS

PointWise	PairWise	ListWise
Take on Candidate	Take entire Candidates	Take the entire list
Compute score between candidate and the query.	Given a pair of candidates decide which one is at higher rank.	Optimises its order.
Algorithms: Anything that deals with regression problems.	Algorithms: RankNet, lambdaNet.	Algorithms: Softrank, Listrank , Ada rank, lambdarank.
Pointwise approaches look at a single document at a time using classification or regression to discover the best ranking for individual results.	Pairwise approaches look at two documents together. They also use classification or regression — to decide which of the pair ranks higher.	Listwise approaches decide on the optimal ordering of an entire list of documents.
Classification means putting similar documents in the same class–think of sorting	We compare this higher-lower pair against the ground truth (the gold	Listwise approaches use probability models to minimize the ordering error

Fig. 6.5: Comparison of Ranking approaches

V. Acknowledgment

With the aforementioned research study and an attempt to implement the chosen solution strategy, the project achieved significant progress and gratified its objective. The process of selecting an approach to developing a solution included the extensive application of various tools and resources related to the domain of the project. The results derived from the utilization of such resources contributed to framing an insight into the type of solution necessary to be established for the worklet and the statistical inferences drawn from the experimentation with these tools aided the activity of selecting a suitable approach to implement the solution.

VI. FUTURE WORK

From the obtained training data, an LTR Model can be built to establish an efficient ranking framework for an independent search engine. Primarily, an algorithm needs to be selected for usage in the engine by referring to the study conducted and analysing the most suitable choice for the requirement. After the selection of the algorithm, the data should be integrated, and the model building process should commence. Once the model is built, it is deployed with the search engine and results can be witnessed with appropriate ranking, paving way for a real-time implementation of this project.

VII. References

[1] Gary Michael Taylor [2020]. Search Results: Predicting Ranking Algorithms With User Ratings and User-Driven Data. Walden University, February 2020.

[2] João Lages, Joao Paulo Carvalho [2020]. Relevance Ranking for Web Search. IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), July 2020.

[3] Nikita Kathare, O. Vinati Reddy, Dr. Vishalakshi Prabhu [2021]. A Comprehensive Study of Elasticsearch, International Journal of Science and Research (IJSR) – Vol. 6 Issue 1, June 2019.

[4] Sandeep Suri, Arushi Gupta, Kapil Sharma [2019]. Comparative Study of Ranking Algorithms. International Conference on Computing, Electronics & Communications Engineering (iCCECE). INSPEC No. 19276360, August 2019.

[5] Vidhi Singrodia, Anirban Mitra, Subrata Paul [2019]. A Review on Web Scraping and its Applications. International Conference on Computer Communication and Informatics (ICCCI). January 2019.

[6] Ioannis C. Drivas 1, Georgios A. Giannakopoulos and Daphne Kyriaki-Manessi , Damianos P. Sakas.[2020].Big Data Analytics for Search Engine Optimization. Year 2020

[7] Axel-Cyrille, Ngonga Ngomo, Michael Hoffmann, Ricardo Usbeck, Kunal Jha.Holistic and scalable ranking ofRDF data. IEEE International Conference on Big Data (BigData). Year 2017