

Table of Contents

1. [Information retrieval system](#_heading=h.30j0zll)
2. [What is text analysis, and why is it important?](#_heading=h.1fob9te)
3. [What is an Elasticsearch Analyzer?](#_heading=h.3znysh7)
4. [Character Filter:](#_heading=h.2et92p0)
5. [Tokenizer:](#_heading=h.tyjcwt)
6. [Token Filter:](#_heading=h.3dy6vkm)
7. [Built-in analyzer in Elasticsearch:](#_heading=h.1t3h5sf)
8. [Standard Analyzer:](#_heading=h.4d34og8)
9. [Whitespace Analyzer:](#_heading=h.2s8eyo1)
10. [Language Analyzers:](#_heading=h.17dp8vu)
11. [Simple Analyzer:](#_heading=h.3rdcrjn)
12. [Keyword Analyzer:](#_heading=h.26in1rg)
13. [Stop Analyzer:](#_heading=h.lnxbz9)
14. [Pattern Analyzer:](#_heading=h.35nkun2)
15. [Fingerprint Analyzer:](#_heading=h.1ksv4uv)
16. [What I have done:](#_heading=h.44sinio)
17. [Two Pipelines/Custom Analyzers (Part A):](#_heading=h.z337ya)
18. Evaluation of Three[IR systems (Part B):](#_heading=h.3j2qqm3)
19. Evaluation in Information Retrieval Briefly Explained
20. Precision and Recall
21. Truth Table
22. Average Precision
23. Mean Average Precision
24. Precision-Recall Curves
25. Practical Example
26. P@5 and R@5 for all three systems
27. Discounted Cumulative Gain
28. Our Results Of Average Precision ,MAP & Precision-Recall Curves

# Information retrieval system

Computing information retrieval (IR) systems make it easy to find information that will meet user-specific information needs. The majority of modern and well equipped IR systems, especially during the online search, use keyword searches to find pages in an ultimate index that includes phrases that find and match the words in the query of the user. Have Most readers recognize keyword searches as a standard pattern for performing general IR. And for good reason: It often performs admirably in many settings. Despite that, owning one for the average person is still out of the reach it highlights keyword phrase matching shortcomings for retrieving information.

There are two well-known (and recognized) issues with text-based IR.

1. How to handle so-called "stop words" in queries, and

2. How to classify the importance of documents matching complex queries. Stop words are words that are usually ignored for search and related ranking purposes.

In English examples of stop words include “a”, “in”, “an”, “the”, “and” and so on. A simple one but effective way to deal the problems regarding stop words is ignoring them while at indexing time and query time. This method is applicable for common terms such as the terms which are the articles listed above, but it cannot completely solve the terms that look common but they do not This method solves for the most common terms, such as the articles listed above, but it do not easily grouped together with unrelated classified articles.

This problem is more difficult to solve, which is basically a search compatibility issue.

Many innovative solutions to the search problem are offered. Some have proven to be quite effective and have gained widespread acceptance.

In particular, Hans Peter Lohan is credited with coming up with the idea of ​​“stop words”, although not the actual term "stop word".

Frequency / Reverse Document Frequency Statistics Term, or TD-IDF for short, is a method that deals with the terms in a corpus which are frequently occurring and tries to emphasize them properly which are most important terms.

General-purpose search engines typically process queries using TF-IDF to assign less weight to documents that match terms found at higher frequencies in the corpus. Whether. Thus, the \squared terms are weighted based on how this statistic determines their \squared relative to the corpus.

# What is text analysis, and why is it important?

Elasticsearch's text analysis makes large-scale data collection usable and searchable. Text analysis is used by search engines to link your query to hundreds of online sites that may be relevant to your needs.

Text search in Elasticsearch is commonly used to set up search engines, record event data and metrics, display text responses, and log analytics. Each of these may require a logical set of unstructured text fields to generate the final data used.

Since the popularity of micro-services, Elasticsearch has become an outgoing tool for log storage. It can be used as a central area for storing logs from different functions, allowing a complete system to be studied simultaneously.

You can also integrate Elasticsearch with other tools that already have analytics and metrics, eliminating the need for further analysis of the results of your Elasticsearch query. Coralogix Elasticsearch provides a flexible API for digesting data.

# What is an Elasticsearch Analyzer?

Elasticsearch Analyzer consists of three parts: a character filter, a tokenizer, and a token filter. All three can convert a text field into a searchable format. Single words, emails, or program logs can be used as text values.

# Character Filter:

A character filter will test each character in its original text value. It has the ability to add, delete or change characters in a string. These edits can be useful if you need to change the alphabet in different alphabet languages.

Analysts do not need character filters. You will also want to use multiple text analytics, which are allowed. Elasticsearch uses all possible character filters in the layout you choose.

# Tokenizer:

A token is a text unit used in search. A tokenizer will divide a continuous series of text into tokens. Tokenizers note the order and location of each word in the text, as well as the start and end character offsets and the type of token.

Position monitoring is useful for word proximity search, and is used to highlight character offsets. Token types identify the type of token data (alphanumeric, numeric, etc.).

Elasticsearch includes lots of tokenizers by default. There are several ways to divide a sentence into tokens, incomplete words, keywords and patterns. See the Elasticsearch website for a comprehensive list.

Elasticsearch analysts should use a tokenizer. Each analyst can have only one tokenizer.

# Token Filter:

The token filter will take a series of tokens from the output of tokenizer. It will then somehow change the tokens. Such as, all the characters in a token can be minimized by token filter, remove tokens in settings, or introduce new tokens based on existing patterns or tokens. A comprehensive set of built-in token filters can be found on Elasticsearch's website.

Analysts do not need token filters. There can be no token filters or multiple token filters with different utilities.

# Built-in analyzer in Elasticsearch:

Elasticsearch includes several built-in analyzers that can be used in any index without any additional configuration:

# Standard Analyzer:

The standard analyzer separates the text into word-defined terms as defined by the Unicode text segmentation algorithm. Most of the punctuations are removed, keywords are lowercase, and removing stop words is supported.

# Whitespace Analyzer:

When it detects a white space character, White Space Analyzer separates the text into terms. It does not use lowercase letters.

# Language Analyzers:

Elasticsearch has specialized analysts in several languages, including English and French.

# Simple Analyzer:8

When it matches a character that is not a letter, the basic analyst separates the text into terms. All terms are lowercased.

# Keyword Analyzer:

A keyword analyzer is a "noop" analyst that receives any text and returns the same text as a single sentence.

# Stop Analyzer:

Stop Analyzer is similar to Simple Analyzer, but it also allows you to remove stop words.

# Pattern Analyzer:

The pattern analyzer breaks down the text into words using regular expressions. It accepts lower case and stop words.

# Fingerprint Analyzer:

Fingerprint Analyzer is a specialized analyzer that develops fingerprints that can be used to detect duplicates.

# What I have done:

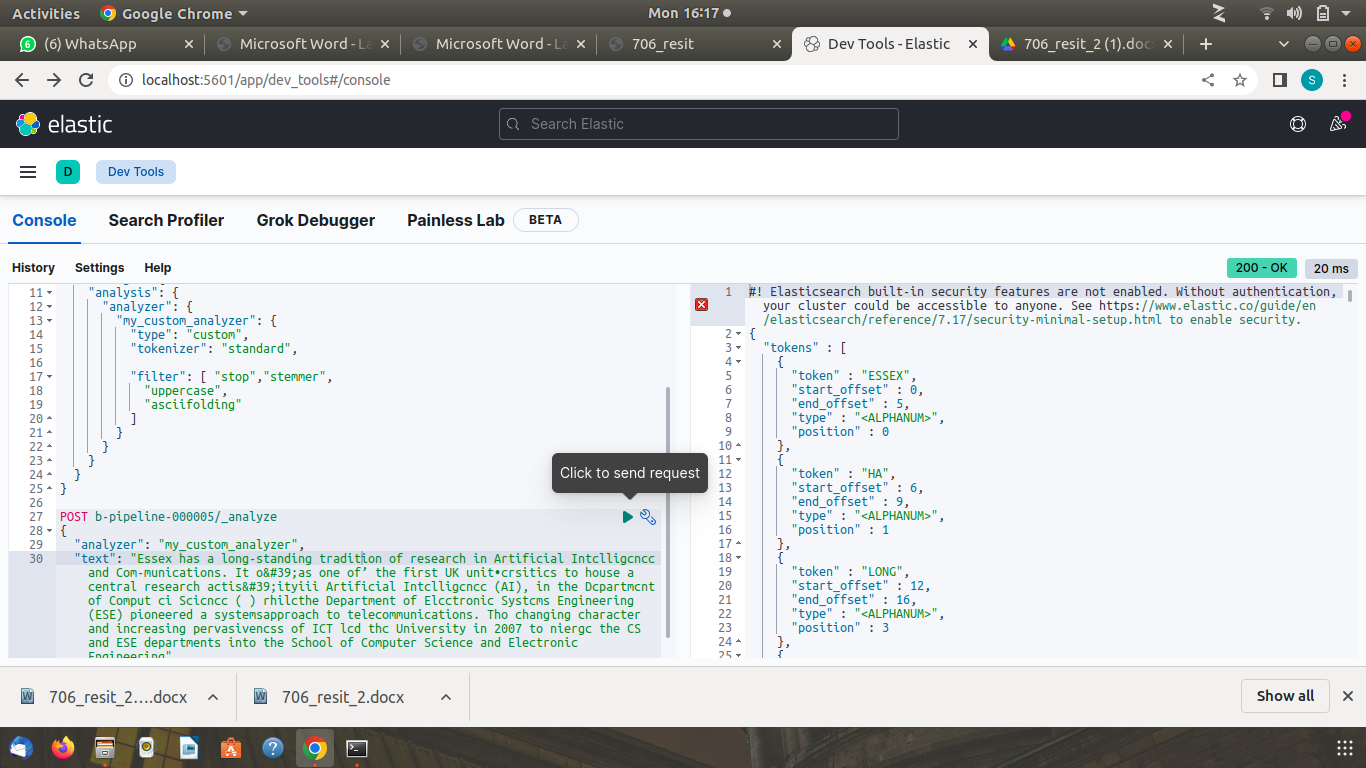
In accordance with the requirements, I created two preparation pipelines (analysers) for the university's web search. Elasticsearch is used to power the search engine. The specifics of two pipelines are shown below

* First pipeline deals with simple text and must be capable of tokenization, case folding, stopword removal, and stemming.
* The second pipeline employs HTML and necessitates content removal, tokenization, case folding, and stopwords removal plc marking.

I used two sample phrases out from assignment file, each one for every question, to evaluate my system. I ran those through my pipelines. The images and code for the Elasticsearch outputs of my pipelines are provided below.

The two analyzers are 000005 and 000003, as illustrated in the photos below.

**Screenshot Plain Text Analyzer:**

****

**Code Plain Text Analyzer:**

In my pipeline, I utilised a conventional tokenizer using uppercase filters, stemmer, and stop to meet the criteria.

**PUT b-pipeline-000005**

**{**

**"settings": {**

**"analysis": {**

**"analyzer": {**

**"my\_custom\_analyzer": {**

**"type": "custom",**

**"tokenizer": "standard",**

**"filter": [ "stop","stemmer",**

**"uppercase",**

**"asciifolding"**

**]**

**}**

**}**

**}**

**}**

**}**

Then i have used the following code to test analyzer

**POST b-pipeline-000005/\_analyze**

**{**

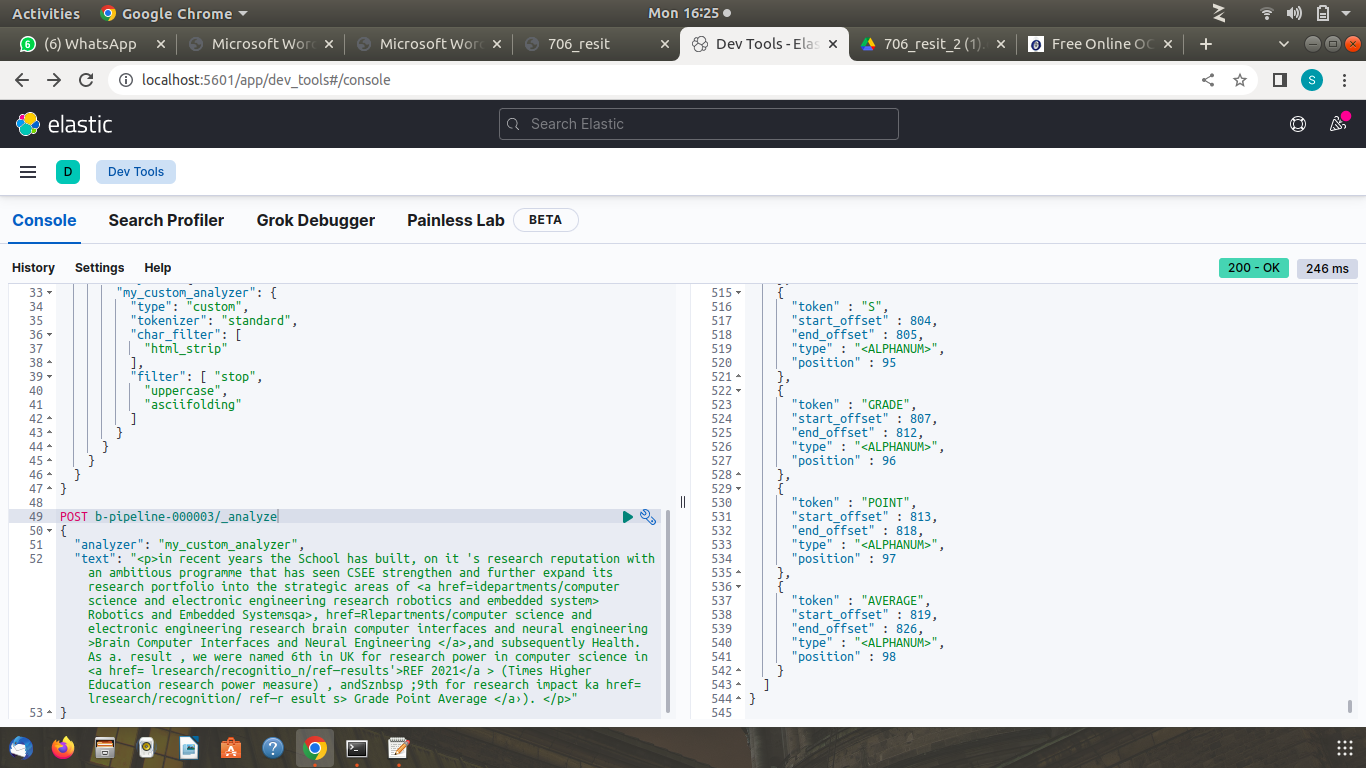
**"analyzer": "my\_custom\_analyzer",**

**"text": “Essex has a long-standing tradition of research in Artificial Intelligence…..)”**

**}**

The output can be viewed on Screenshot.

**Screenshot HTML Text Analyzer:**

****

**Code HTML Text Analyzer:**

In my pipeline, I utilized a basic tokenizer with uppercase, html strip, and stop filters to meet the criteria.

**PUT b-pipeline-000003**

**{**

**"settings": {**

**"analysis": {**

**"analyzer": {**

**"my\_custom\_analyzer": {**

**"type": "custom",**

**"tokenizer": "standard",**

**"char\_filter": [**

**"html\_strip"**

**],**

**"filter": [ "stop",**

**"uppercase",**

**"asciifolding"**

**]**

**}**

**}**

**}**

**}**

**}**

Then i have used the following code to test analyzer

**POST b-pipeline-000003/\_analyze**

**{**

**"analyzer": "my\_custom\_analyzer",**

**"text": " <p>in recent years the School has built, on it 's research reputation with an ambitious programme…."**

**}**

The output can be viewed on Screenshot.

**Part B Of Assignment:**

Now, based on the information provided, let's examine three information retrieval systems. I have 20 Documents (A-T). The relevance judgments of all documents are given in Table and may be viewed in the Table below, which also shows the rank of a document of all three IR systems. Lets compute various metrics for each of the three IR systems.

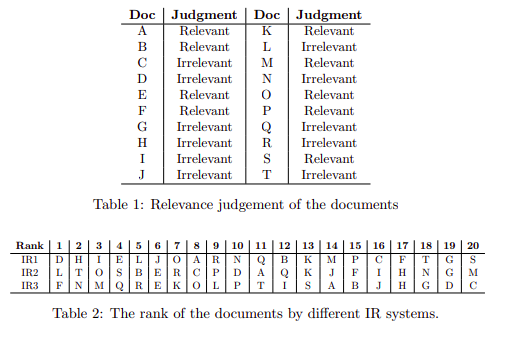


Table No 1

Before we start calculating, let's first learn information retrieval techniques and how to calculate and understand recall, accuracy, rank, precision & recall curve, precision and relevance with examples, and Then we will briefly address the instructor's questions.

**Assessment in information retrieval defined briefly**

For information searching, a typical way to evaluate information retrieval systems use the idea of relevant and irrelevant content.

A user looks for information. System outputs can be characterized as relevant or irrelevant depending on their ability to meet a certain information need.

**Precision and recall:**

Precision and recall are two variables that are widely employed in the evaluation of information retrieval.

Accuracy is the fraction of relevant documents among those recovered.

**Precision = #(relevant items retrieved) / #(total retrieved items)**

The percentage of relevant documents that are recovered is referred to as recall.

**Recall = #(relevant items retrieved) / #(total relevant items in the collection)**

* **THE TRUTH TABLE**

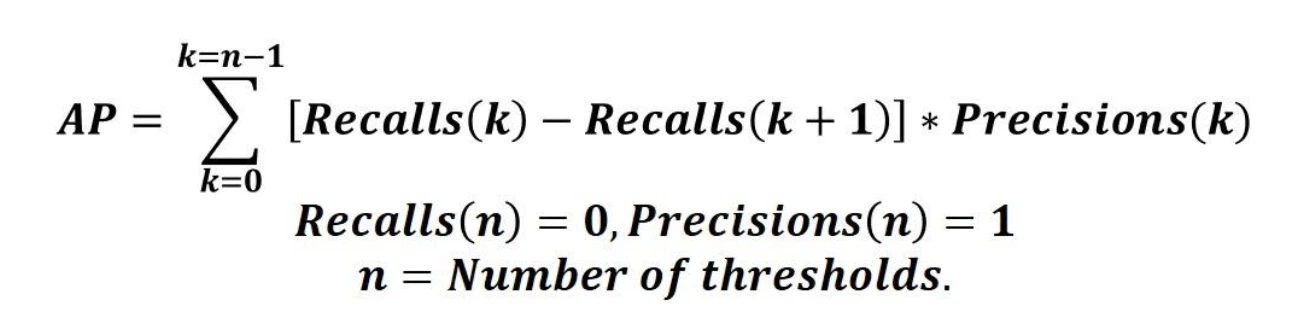
| What the system thinks | Relevant | Non Relevant |
| --- | --- | --- |
| Retrieved | True Positives | False Positives |
| Not Retrieved | False Negatives | True Negatives |

* SYSTEM Retrieved true positives (TP) false positives (FP)
* Not retrieved false negatives (FN) true negatives (TN)

**Average Precision**

For a question, average precision (AvP) provides a single statistical measure of performance across recall levels. It is the average of the accuracy values obtained after retrieving each relevant document from the result lists.

**Formal Average Precision formula:**



As a result, the following is a general average precision calculation algorithm. Get your model's predictions, define thresholds, and create Precision-Recall curves (for example, in a classification situation, you can compute micro or macro precision/recall).

Run over all precision/recall pairs in a loop; Calculate the difference (weight) between the current and next recall values. Steps 2-4 should be repeated for the following pairs. total score obtained; And check the average accuracy value.

Also, if you have a multi-class situation, you can calculate the average accuracy score for each class separately. You'll get a better grasp of the algorithm's performance when you draw PR curves for each category and learn whether your model is effective at recognizing specific class items,start by determining the weight then multiply the weight by the appropriate values and the last step is to summarize the obtained values. The result is shown in the last section.

**Mean Average Precision**

Mean accuracy (MAP) is the average of AVP values for a given set of questions. This is the best indicator of system quality.

**Precision-Recall Curves**

A precision-recall curve illustrates the trade-off between precision and recall at various thresholds.

**Practical Example:**

A customer wants to know if eating fruit may help him improve his immune system (information needed). A user looks for fruits that help the immune system (query).

Assume the system produces five outcomes.

From the first to the final result, they were categorized as non-relevant, relevant, irrelevant, irrelevant, relevant, relevant as per the user's desire for information.

Assume that the indexed collection has 8 items that are pertinent to this information query and the set is 00101

Precision = (3 / 5) = 0.6

Recall = (3 / 8) = 0.375

For each of the five outcomes, we may compute the accuracy and recall value.

**Results** **0 0 1 0 1**

**Recall** 1/8 2/8 2/8 3/8 3/8

**Precision** 1/1 2/2 2/3 3/4 3/5

The precision-Recall curve is constructed using those recall and precision parameters.

Using the precision values for relevant documents, we could now calculate the average precision (AvP).

**AvP = (1/1 + 2/2 + 3/4) / 3 = 0.91**

Assume the user conducted a second search for a different information requirement, and the system reported an overall accuracy of 0.82.

For both information requirements, we can calculate the system's Mean Average Precision (MAP) as follows:

**MAP = (0.91 + 0.82) / 2 = 0.865**

**Precision/Recall @ Rank Example:**

| Rank  1  2  3  4  5  6  7  8  9  10 | Doc  d12  d123  d4  d57  d157  d222  d24  d26  d77  d90 |
| --- | --- |

Blue documents are relevant

P@n: P@3=0.33, P@5=0.2, P@8=0.25

R@n: R@3=0.33, R@5=0.33, R@8=0.66

**Now let's do the first task and calculate the P@5 and R@5 for all three systems:**

As the details have given we have three Ir systems and 20 documents as shown in table no 1:

**P@5 and R@5 Of IR System 1:**

In IR1, from out a total of 20 documents, 8 are applicable and 12 are just not, hence the P@5 and R@5 are as follows.

P = 8/(8 + 12) = 0.40

R = 8/(8 + 20) = 0.28

P@n: P@5=0.40

R@n: R@5=0.28

**P@5 and R@5 Of IR System 2￼:**

In IR2 out of total 20 documents 9 are relevant and 11 are not relevant so the P@5 and R@5 are given below

P = 9/(9 + 11) = 0.45

R = 9/(9 + 20) = 0.31

P@n: P@5=0.45

R@n: R@5=0.31

**P@5 and R@5 Of IR System 3￼:**

In IR3 out of total 20 documents 9 are relevant and 11 are not relevant so the P@5 and R@5 are given below

P = 9/(9 + 11) = 0.45

R = 9/(9 + 20) = 0.31

P@n: P@5=0.45

R@n: R@5=0.31

**Now let's compute the DCG for all systems**

Discounted profit after deduction

Discounted gross profit is the measure used to calculate the grading criteria (DCG). It is commonly used in information retrieval problems such as measuring the performance of search engine algorithms by ranking articles presented based on relevance to a search query.

Let’s consider that a search engine that outputs 20 documents named in alphabets as mentioned below the relevance scale (0 and 1) where:

0 : not relevant

1 : relevant

The documents have relevance scores:

**A, B, E, F, K, M, O, P, S, = 1 (Relevant)**

**C, D ,G, H, I, J, K, L, N, Q, R, T= 0 (Not Relevant)**

The Cumulative Gain is the sum of these relevance scores and can be calculated as:

For IR1 The CG is 0+ 0+ 0+ 1+ 0 +0 +1 +1 +0 + 0+ 0+ 1+ 0+ 1+ 1+ 0+ 1+ 0+ 0+ 1= 8

For IR2 The CG is 0+ 0+ 1+ 1+ 0 +0 +1 +1 +0 + 0+ 0+ 1+ 0+ 1+ 1+ 0+ 1+ 0+ 0+ 1= 9

For IR3 The CG is 1+ 0+ 0+ 1+ 0 +0 +1 +1 +0 + 0+ 0+ 1+ 0+ 1+ 1+ 0+ 1+ 0+ 0+ 1= 9

The following formula can be used to determine the discounted cumulative gain:

**Results**

**4.** Let's draw the precision recall curves for all systems and select one system for a scholar search that requires 80% of recall. The methods of Evaluation steps are mentioned above after following them the results are obtained and mentioned below.

**Information Retrieval System 1 SET (1110100111010010110):**

* **PRECISION:**

[0.0, 0.0, 0.0, 0.25, 0.2, 0.16666666666666666, 0.2857142857142857, 0.375, 0.3333333333333333, 0.3, 0.2727272727272727, 0.3333333333333333, 0.3076923076923077, 0.35714285714285715, 0.4, 0.375, 0.4117647058823529, 0.3888888888888889, 0.3684210526315789, 0.4]

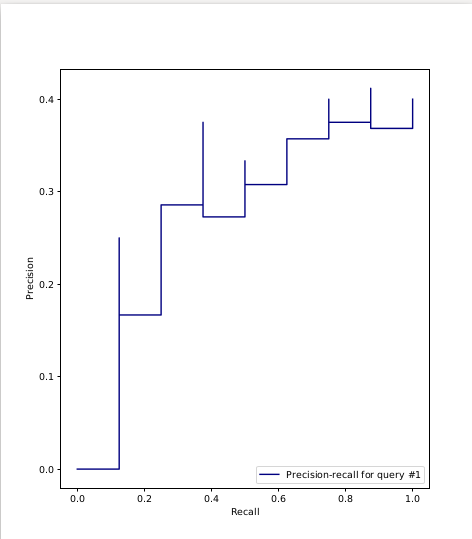
* **RECALL:**

[0.0, 0.0, 0.0, 0.125, 0.125, 0.125, 0.25, 0.375, 0.375, 0.375, 0.375, 0.5, 0.5, 0.625, 0.75, 0.75, 0.875, 0.875, 0.875, 1.0]

* **AVERAGE PRECISION:**

0.3516193977591036

* **MAP:** 0.3516193977591036
* **The Graph Of Precision Recall Curve**



**Information Retrieval System 2 SET (11000011010101011110):**

* **PRECISION:**

[0.0, 0.0, 0.3333333333333333, 0.5, 0.6, 0.6666666666666666, 0.5714285714285714, 0.5, 0.5555555555555556, 0.5, 0.5454545454545454, 0.5, 0.5384615384615384, 0.5, 0.5333333333333333, 0.5, 0.47058823529411764, 0.4444444444444444, 0.42105263157894735, 0.45]

* **RECALL:**

[0.0, 0.0, 0.1111111111111111, 0.2222222222222222, 0.3333333333333333, 0.4444444444444444, 0.4444444444444444, 0.4444444444444444, 0.5555555555555556, 0.5555555555555556, 0.6666666666666666, 0.6666666666666666, 0.7777777777777778, 0.7777777777777778, 0.8888888888888888, 0.8888888888888888, 0.8888888888888888, 0.8888888888888888, 0.8888888888888888, 1.0]

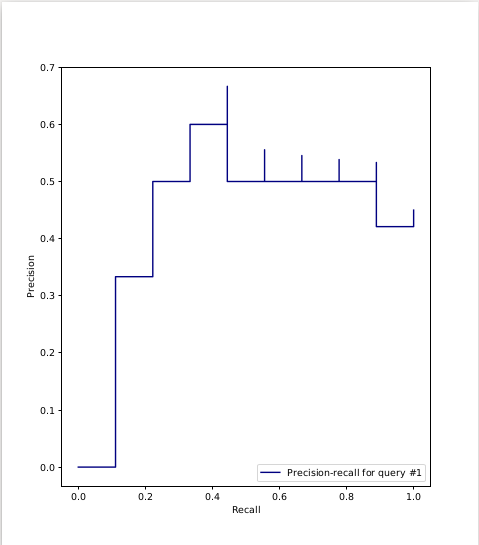
* **AVERAGE PRECISION:**

0.5247561080894414

* **MAP:**

0.5247561080894414

* **The Graph Of Precision Recall Curve**

****

**Information Retrieval System 3 SET (0101100010110001111):**

* **PRECISION**:

[1.0, 0.5, 0.6666666666666666, 0.5, 0.4, 0.5, 0.5714285714285714, 0.625, 0.5555555555555556, 0.6, 0.5454545454545454, 0.5, 0.5384615384615384, 0.5714285714285714, 0.6, 0.5625, 0.5294117647058824, 0.5, 0.47368421052631576, 0.45]

* **RECALL:**

[0.1111111111111111, 0.1111111111111111, 0.2222222222222222, 0.2222222222222222, 0.2222222222222222, 0.3333333333333333, 0.4444444444444444, 0.5555555555555556, 0.5555555555555556, 0.6666666666666666, 0.6666666666666666, 0.6666666666666666, 0.7777777777777778, 0.8888888888888888, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]

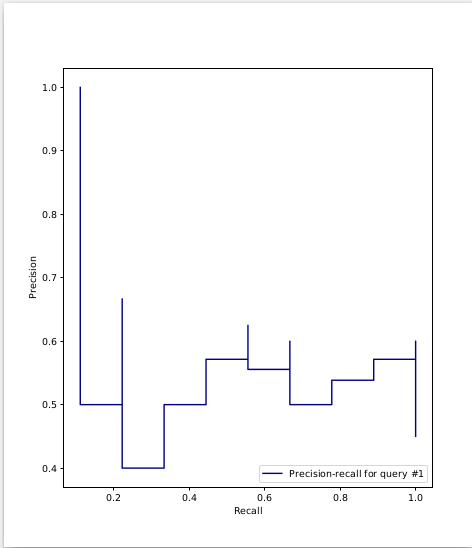
* **AVERAGE PRECISION:**

0.6303317053317053

* **MAP:**

0.6303317053317053

* **The Graph Of Precision Recall Curve**



### 