Using Nutch in Baa (Alecso Open Source Search Engine)

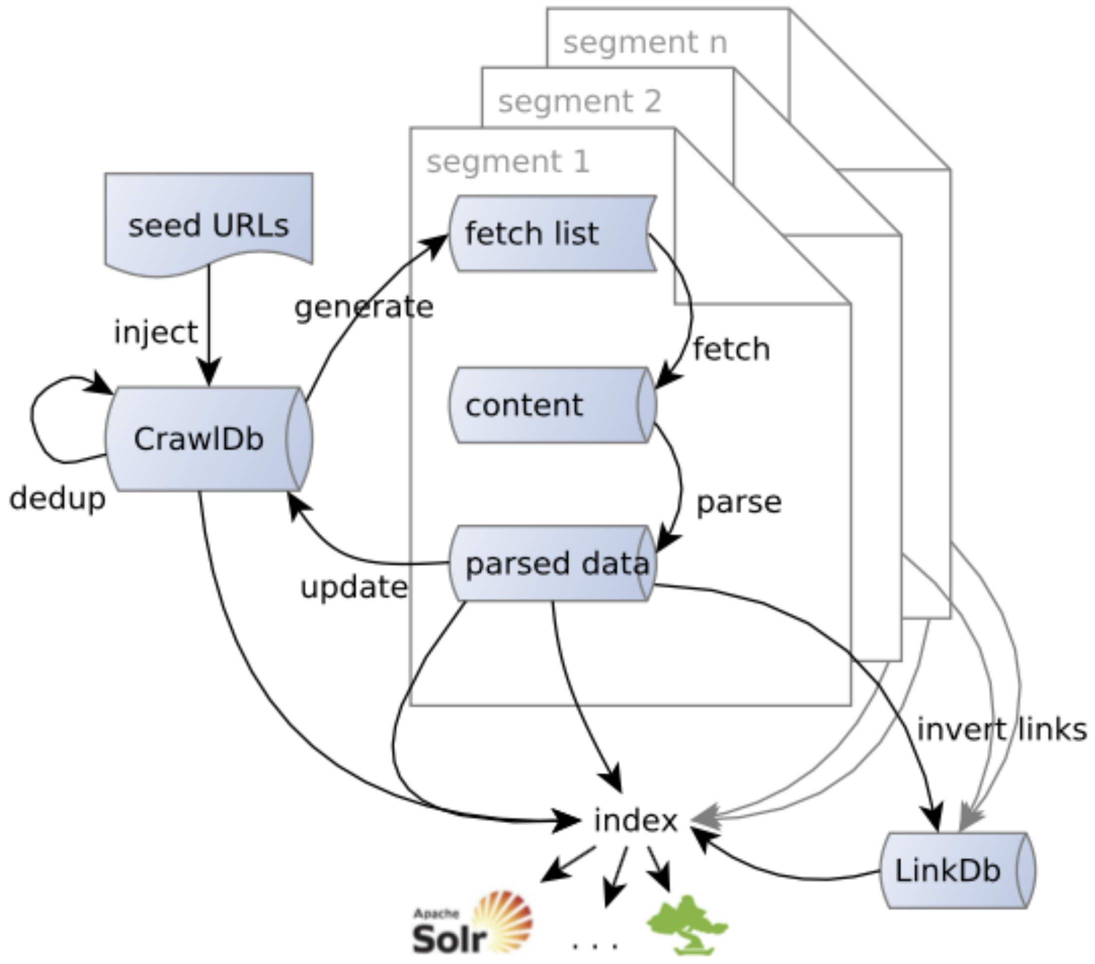
إعداد: د. عبدالسلام النويصري

contents

“ an extensible and schalable web crawler based on Hadoop”

- Runs on top of Hadoop
- Customizable
  - Pluggable protocols
  - URL filter
  - Parsing TIKA
  - -Indexing back end
- Mostly used to feed search engines

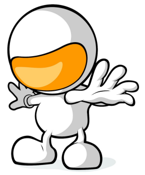
Nutch Workflow



Nutch Workflow

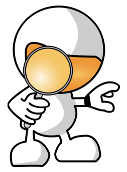
Typical workflow is a sequence of batch operations

- Inject: Populate crawlDB from seed list
- Generate: Selects URLs to fetch
- Fetch: Fetched URLs from fetchlist
- Parse: Parse content from fetched URLs
- UpdateDB: Update the crawlDB
- InvertLinks: Builds the linkDB
- Index: Optional step to index in SOLR, Elasticsearch, etc



Broad vs. Focused Crawling

- Broad Crawling :
  - Unlimited crawl frontier
  - Limited by bandwidth and politeness factors
  - Useful for creating an index of the open web
  - Can achieve high recall
  - Not useful for domain discovery as crawled content may include a lot of irrelevant material
- Focused Crawling :
  - Limit crawl frontier by calculating relevance of URL
  - Low resource consumption as compared to the above
  - Can achieve high precision
  - Useful for domain discovery as it prioritizes based on content relevance



Domain Discovery

A “Domain”, here, is defined as an area of interest for a user.

Domain Discovery is the act of exploring a domain of which a user has limited prior knowledge.

Domain discovery process may include :

- Using a focused crawler
- User providing some prior knowledge in the form of text, questions or reference websites

Focused Crawling with Nutch

Previously available tools :

- URL filter plugins
    - Filter based on regular expressions
    - Whitelist/blacklist hosts
- Filter based on content mimetype
- Scoring links (OPIC scoring)
- Breadth first or Depth first crawl

Limitations :

- Follows the link structure
- Does not capture content relevance to a domain

Focused Crawling with Nutch

To capture content relevance to a domain, two new tools have been introduced.

- Cosine Similarity scoring filter
- Naive Bayes parse filter

Nutch JIRA issues :
https://issues.apache.org/jira/browse/NUTCH-2039
https://issues.apache.org/jira/browse/NUTCH-2038

Cosine Similarity

Cosine similarity is a measure of similarity between two vectors of an inner product space that measures the cosine of the angle between them [1].

Similarity = cos(Θ) = A . B / |A| . |B|, where A and B are the vectors.

Lesser the angle => higher the similarity

[1] https://en.wikipedia.org/wiki/Cosine_similarity

Cosine Similarity Scoring in Nutch

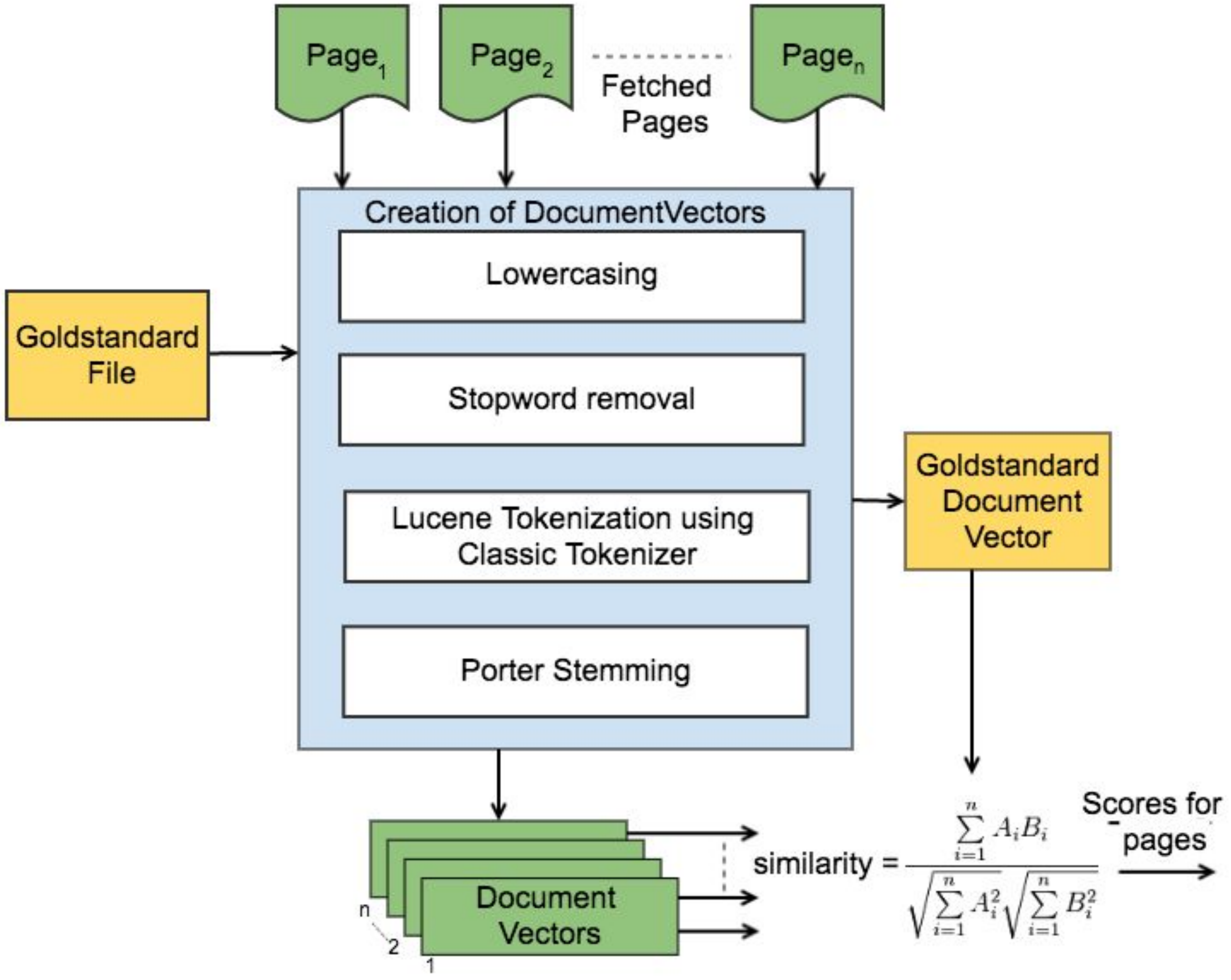
- Implemented as a Scoring filter
- Computed by measuring the angle between two Document Vectors.

Document Vector :

A term frequency vector containing all the terms occurring on a fetched page.

DV = {“robots”:51, “autonomous” : 12, “artificial” : 23, …. }

Cosine Similarity Scoring - Architecture

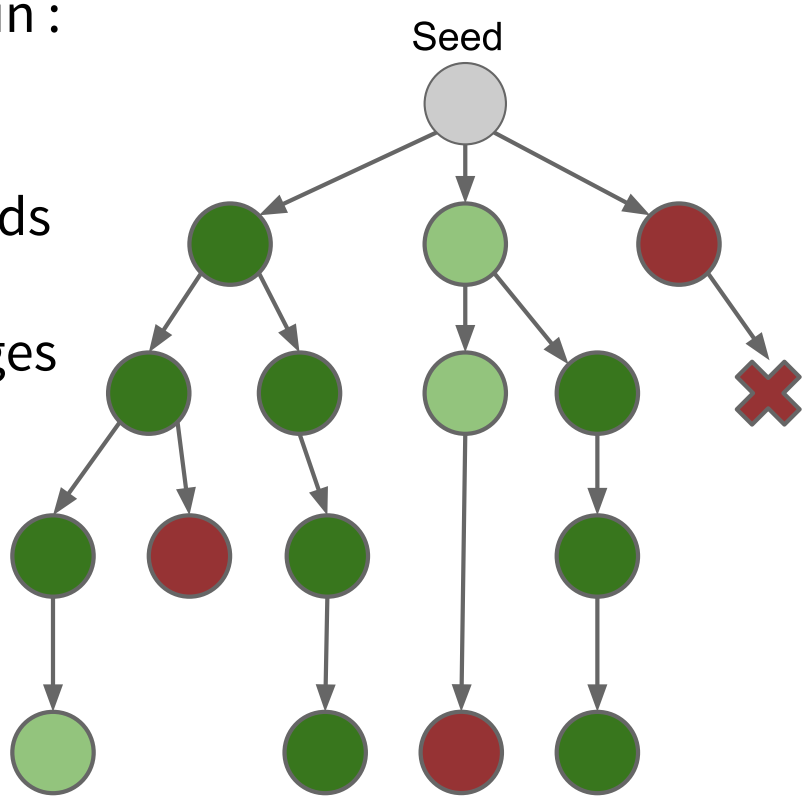


$$\text{similarity} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

Cosine Similarity Scoring - Working

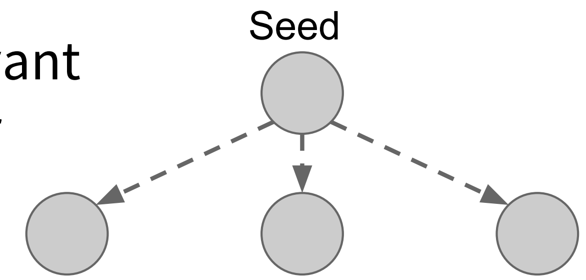
Features of the similarity scoring plugin :

- Scores a page based on content relevance
- Leverages a simplistic bag-of-words approach
- Outlinks from relevant parent pages are considered relevant



Iteration 1

- Start with an initial seed
- Seed is considered to be relevant
- User provides keyword list for cosine similarity



All children given same priority as parent in the crawl frontier

Policy : Fetch top 4 urls in frontier

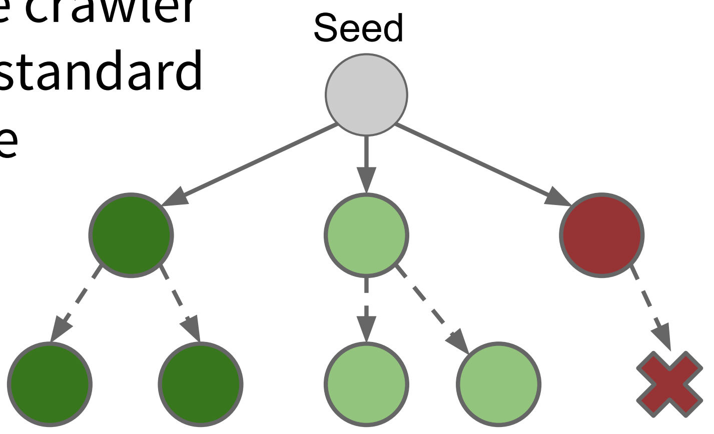
- - - - -▶ Unfetched (in the crawl frontier)

——▶ Fetched

🟢🟢🔴 Decreasing order of relevance

Iteration 2

- Children are fetched by the crawler
- Similarity against the goldstandard is computed and scores are assigned.



Policy : Fetch top 4 urls in frontier

- - - - -▶ Unfetched (in the crawl frontier)

——▶ Fetched

🟢🟢🔴 Decreasing order of relevance

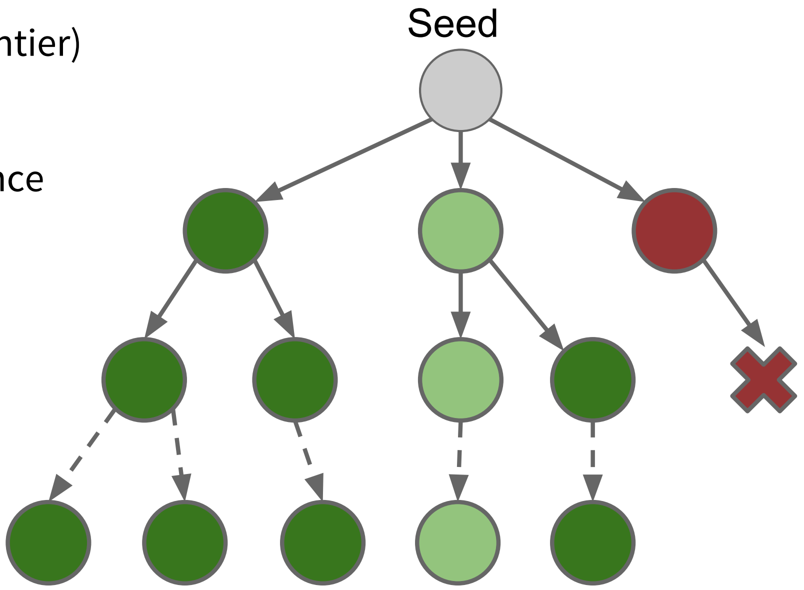
Iteration 3

Policy : Fetch top 4 urls in frontier

- - - - -► Unfetched (in the crawl frontier)

———► Fetched

🟩🟩🟥 Decreasing order of relevance



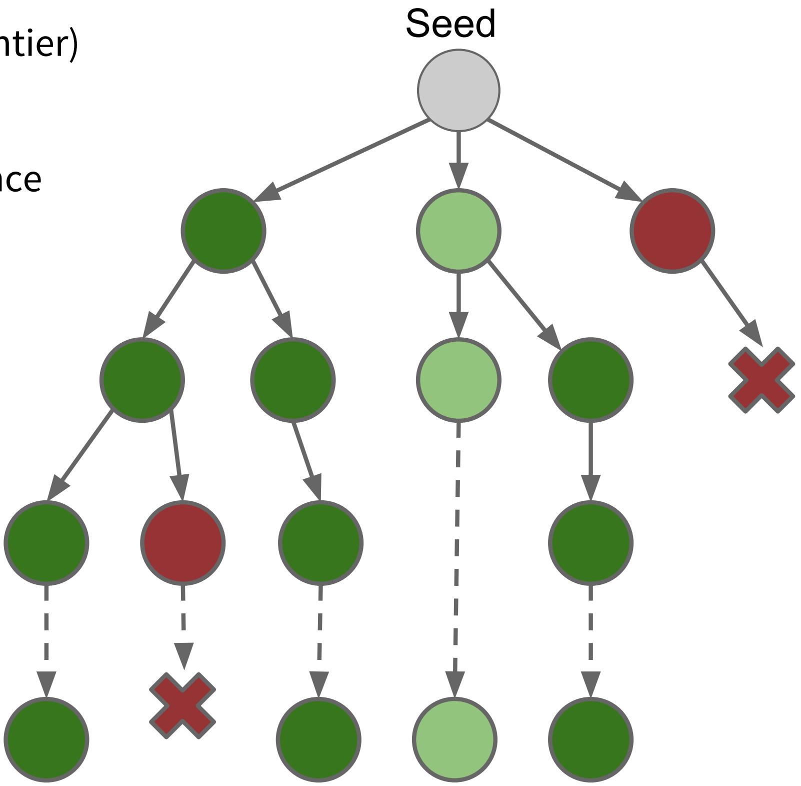
Iteration 4

Policy : Fetch top 4 urls in frontier

- - - - -► Unfetched (in the crawl frontier)

———► Fetched

●●● Decreasing order of relevance



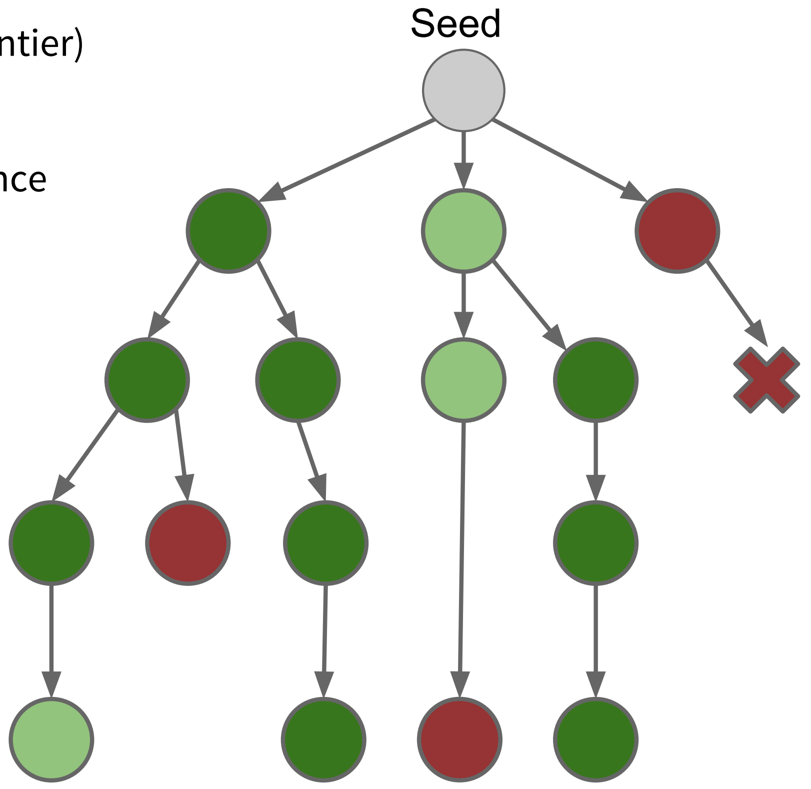
Iteration 5

Policy : Fetch top 4 urls in frontier

- - - - -► Unfetched (in the crawl frontier)

———► Fetched

● ● ● Decreasing order of relevance



Naive Bayes Classifier

Naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between the features [1].

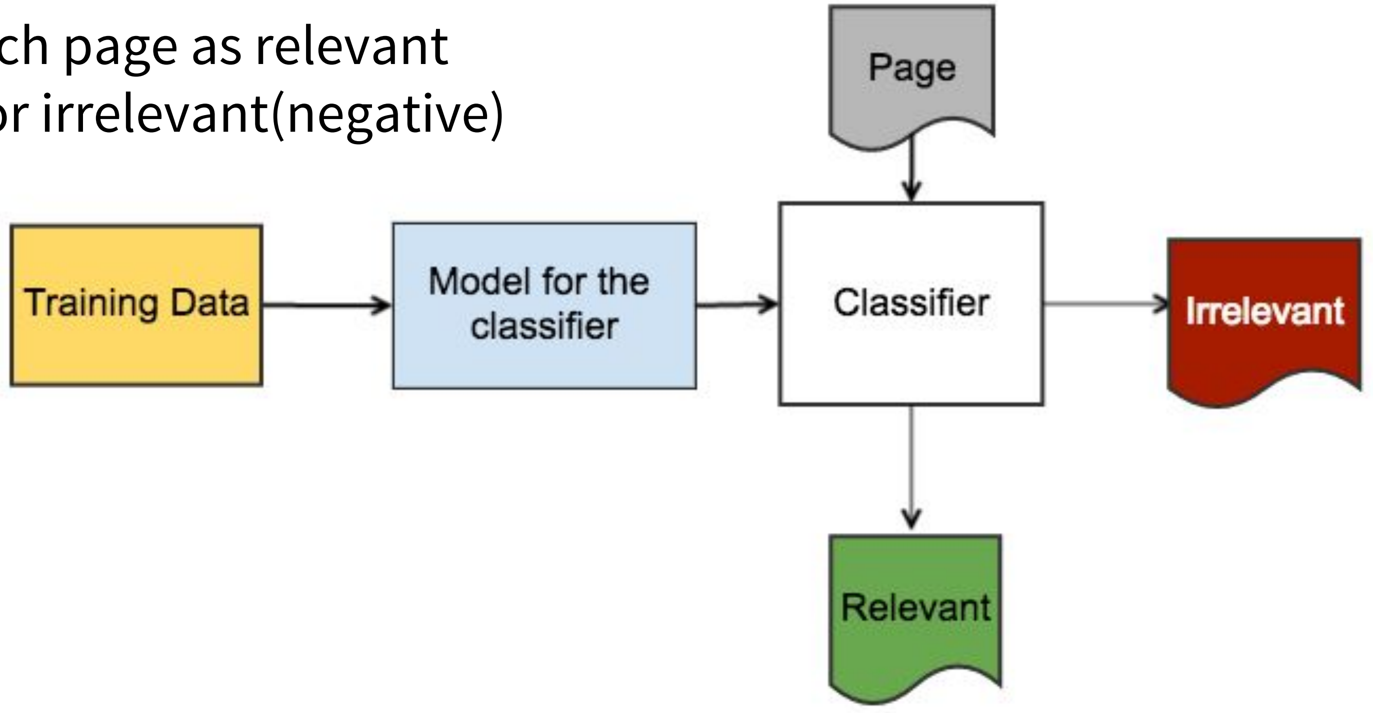
Naive Bayes in Nutch

- Implemented as a parse filter
- Classifies a fetched page relevant or irrelevant based on a user provided training dataset

[1] https://en.wikipedia.org/wiki/Naive_Bayes_classifier

Naive Bayes Classifier Working

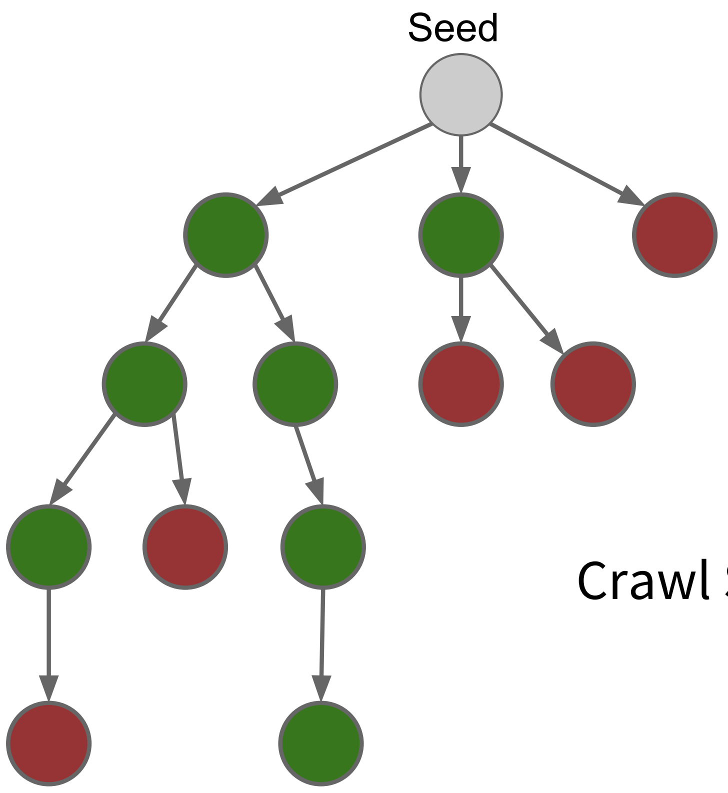
- User provides a set of labeled examples as training data
- Create a model based on given training data
- Classify each page as relevant (positive) or irrelevant(negative)



Naive Bayes Classifier Working

Features:

- All outlinks from an irrelevant (negative) page are discarded
- All outlinks from a relevant (positive) page are followed



Crawl Scenario

Rooms for research

- Check how to focus crawling on Arabic
- Check parsing different documents using Tika. Check for rooms of improvement to Arabic
- Explore current duplication detection techniques for Arabic text and suggest improvements.

Requirements

- To be able to carry research on crawling Arabic text, we need
  - Nutch installed on a dedicated server, preferably integrated with Solr
  - Access to the Internet with unlimited bandwidth
  - One research assistant with the following skills:
    - Java programmer
    - Shell scripting

Time frame

- Installing Nutch and starting crawling,
  - Two weeks, depends on the server availability
- Checking current focused crawling and looking for rooms of improvement
    - 6 months
- Reviewing current Arabic documents parsing
    - 6 months
- Checking the effectiveness of current duplication detection techniques for Arabic
    - 6 months