Similarity Analysis of Web Crawled Data

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering

in

Computer Science and Engineering

Submitted By

Parul Garg

(Roll No. 801332018)

Under the supervision of

Mr. Vinay Arora

Assistant Professor

(CSED)

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

THAPAR UNIVERSITY

PATIALA – 147004

July 2015

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, "*Similarity Analysis of Web Crawled Data*", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Computer Science and Engineering* submitted in Computer Science and Engineering Department of Thapar University, Patiala; is an authentic record of my own work carried out under the supervision of *Mr. Vinay Arora* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University. I also certify that in future the guide will not be responsible for any kind of issue related to my thesis work (e.g. originality of concept, plagiarism etc.).

(Parul Garg)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge. In future, candidate is self responsible for any kind of issue related to his thesis work (e.g. originality of concept, plagiarism, content etc.). I believe that the Computer Science & Engineering Department has checked for originality of content through Turnitin.

(Vinay Arora)
Assistant Professor
Thapar University
Patiala

Countersigned by

(Dr. Deepak Garg)
Head of Department
Computer Science and Engineering Department
Thapar University
Patiala

(Dr. S. S. Bhatia)
Dean(Academic Affairs)
Thapar University
Patiala

Abstract

With the advancements in the internet, searching the web is much significant. To retrieve the web pages automatically, web crawler is used. Web crawler feeds on a seed URL and visits all the subsequent URLs to gather information. The processed information is stored in the documents with a file known as JSON documents. Since the number of pages retrieved by Web crawler is in millions, there is a need to find the association between web pages and this can be done with the use of an efficient data mining technique called association rule mining. In this thesis the frequent items are found using Apriori algorithm and association rules are formed using these frequent items. We have used a crawler that crawls a recipe site and proposed a technique to find out the similarity from the set of data related to the recipe items. Then from the structured data of JSON file, association rules are predicted. These association rules are of much significance and can be used to obtain data and solve queries in many desktop as well as web applications.

ACKNOWLEDGMENT

First of all, I am extremely thankful to my respective guide ***Dr. Deepak Garg,*** Associate professor, CSED, Thapar University for his valuable guidance, advice, motivation, encouragement, moral support, sincere effort and positive attitude with which he solved my queries and provide delightful ambiance for learning, exploring and making this thesis possible. He always set high goals for me and help me to find the right direction to achieve those goals. It has been a great experience to work under his guidance.

I am also heartily thankful to ***Mr. Vinay Arora***, Assistant Professor, CSED, Thapar University for motivation and guidance that was very helpful to achieve my goals.

I would like to thank my family members and my friends who are dearest and precious to me for their love, encouragement, blessings and support in all respects. Most importantly, none of this would have been possible without the love and patience of my family. To my brother and friends for showing me right direction. They are constant source of love, concern, support and strength for me all these years.

Finally I would like to thank management of Thapar University for proving a great platform for learning, not just for academics but also for sports and many other creative things.

Parul Garg

801332018

ME (CSE)

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

The World Wide Web has changed very fast from the very first days of its inception. At the same time, a number of new technologies have improved computers and networks, website maintenance and handling costs and the computer hardware prices have dropped. These changes popularized the Internet at a mass level. In addition, a surprising number of development tools, applications and frameworks have developed that allows a flexible development of a website without any effort to learn coding. These tools and frameworks provide all the crucial facilities for publishing and editing content so that ordinary users need not required to be the computer experts. As a result of these facilities a large number of blogs, forums, and web sites are a part of the world wide web, raising the quantity of the content generated by normal users. The concept of Web2.0 gives the plus value to the active involvement of users and communities on the internet.

In order to meet the latest design standards popularized by Web 2.0, most of the web sites such as social networking web sites, websites of journals, newspapers and e-commerce needed to modify their systems. The data generated by the users can be very useful as a major and unique cause of information. The content which is in the form of unstructured data, is used for, recovering and extracting meaningful information using the expertise techniques.

This work delves into the use of such techniques, concentrated on the statistics of user-generated data in the e-commerce background.

1.1 Search Engines

As the amount of data available on internet is growing exponentially, the need for extraction of useful information is also growing but before one can extract information from data one needs to find out where is the data located. Web search engines are the means to achieve this. According to Wikipedia "A *web search engine* is a software

system that is designed to search for information on the World Wide Web." Some of the popular search engines include Google, Yahoo, Bing, Baidu and MSN Search are shown in Figure 1.1.

Figure 1.1 Some of the popular Search Engines

When search engines were initially launched, the only textual data available was but today it is a combination of text, images, videos, etc.

A search engine operates in the following order:

- Web crawling
- Indexing
- Matching

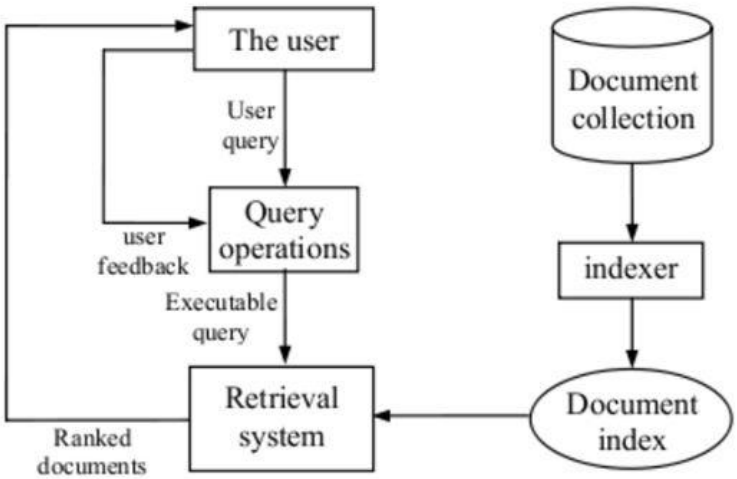

Figure 1.2 Search Operation in a search engine [9]

1.1.1 Web Crawling

Matthew Gray's World Wide Web Wanderer (1993) was one of the first efforts to automate the discovery of web pages. The role of the web crawler is to extract the data from the Web pages and place the downloaded web resources into a local repository.

1.1.2 Indexing

Once the web pages are downloaded in the local repository (Fig. 1), the next important step followed by the search engine is indexing of the data or contents which have been downloaded by the web crawler. The steps followed in this include:

- **Stop words removal**
- **Stemming** : For example, playing, plays, and play may all be stemmed to play so that a search for play will match all its variants.

  An example index (usually called an **inverted index**) will look something like this where the number corresponds to a web page that contains the text:

  play> 3,6

  game > 2,3, 5

  indoor > 1, 3

  outdoor > 4

  So a query for *play* would result in pages 3 and 6. A query for *game and indoor* would give page 3 as result as the only page that contains both search terms game and indoor is 3. Other possibilities of improvements are by adding the advanced features like *OR and NOT*. Further multiple weights are added corresponding to each term in search engine depending upon various factors that determines how relevant the term is to its host web page.

1.1.3 Matching

Matching is the process of relating the search query with the indexed pages which are available to the user on Search Engine Result Page (SREP). Based on user' query the

Search Engine scans the entire list generated by indexing the web pages and all the relevant results which fulfill the requested pages are displayed to the user. The user's request is fulfilled only through the crawlers which crawl through the websites to get the relevant search results.

Whenever a user feed some input query, some results are definitely received in the form of web links, for example, Figure 2 shows a search engine result page generated by Google for key word "example keywords". The efficiency of the results achieved i.e. the relevance of the results determine the actual usefulness of the search engine.

Since millions of web pages may have the word or group of words containing the required web pages it is very difficult to judge which page should be displayed at first position. To do this most search engines use different ranking methods and based on the rank the web pages position is determined by the search engine.

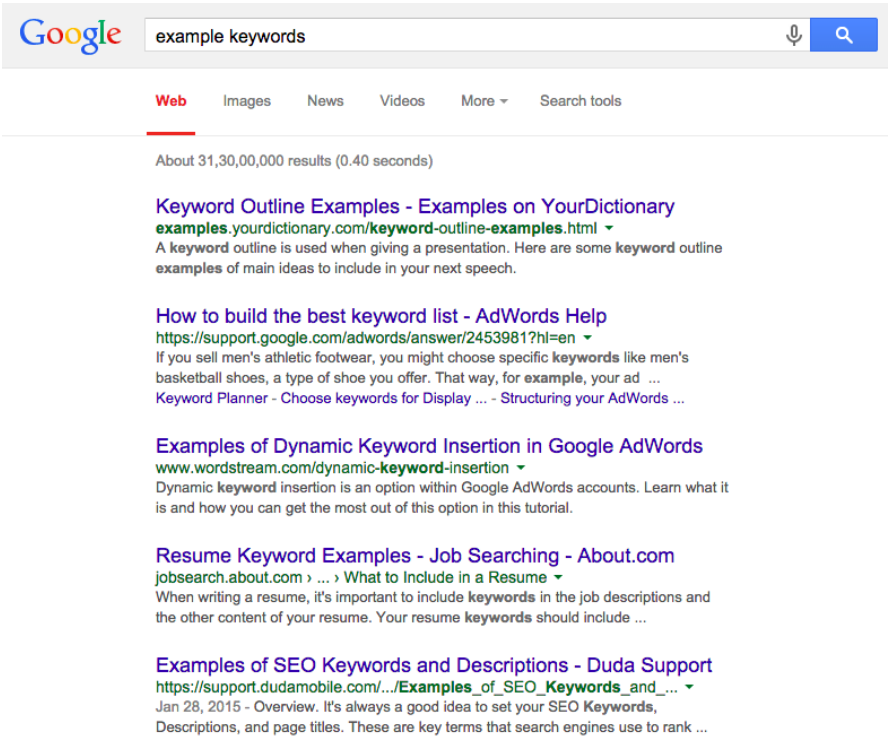

Figure 1.3 Search engine result page generated by Google for key word "example keywords"

1.2 Role of Web Crawlers

A Web crawler is an Internet bot which systematically browses the World Wide Web, typically for the purpose of Web indexing. The web crawler sets out from the search engine's base computer system looking for websites to index.

1.2.1 Basic Crawling Terminology

Seed Page: To crawl means to move through the entire Web by recursively following links from a starting URL or a set of starting URLs. The initial state is the URL set where crawler starts its first search. This set of starting URL is known as “Seed Page” and one of the most influential factor in the process of crawling is the choice of a good seed.

Frontier (Processing Queue): The seed extracts the links and keep on adding un-visited URLS called Frontiers and enhance the list of URLs. Crawler Scheduler picks a URL from the frontier which can be implemented by using advanced data structures.

Parser: After fetching the pages, parsing is the stage leading to extraction of information which feed and guide the future path of the crawler. The parser’s role is to parse the fetched web page to extract list of new URLs from it and return the new un-visited URLs to the Frontier.

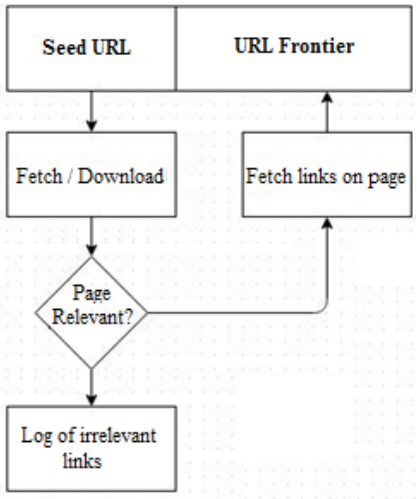

Figure 1.4 Architecture of Web Crawler [36]

1.3 Data Mining

Data mining or knowledge discovery is the computer-assisted process of extracting the meaning of data by analyzing enormous sets of data. Data mining aims at digging the similarities between searching for valuable information in a large database and mining a mountain for a vein of valuable ore. Steps followed in Data mining are:

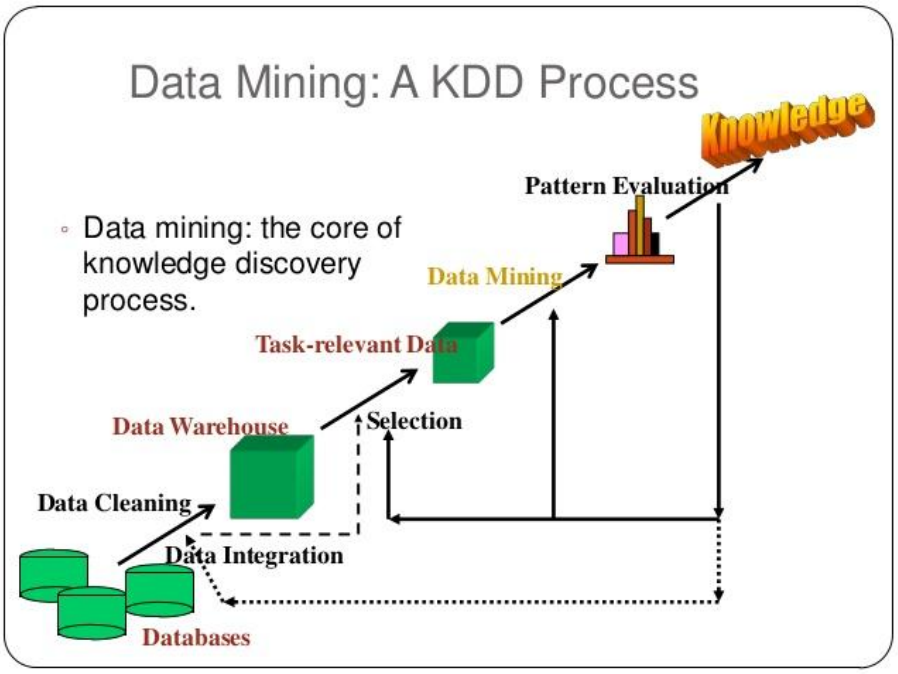

Figure 1.5 Steps followed in Data Mining

Data mining can answer questions that cannot be addressed through simple query and reporting techniques.

1.4 Association Rules

Association rules are if/then statements that help uncover relationships between seemingly unrelated data in a relational database or other information repository. An example of an association rule would be "If a customer buys a dozen eggs, he is 80% likely to also purchase milk."

It is classically defined as: Suppose I as a set of items i1, i2, . . . , in. Suppose T = t1, t2, . . . ,tn as a transaction set. Every transaction t in T occupies a unique id and carries a subset of items in I. An association rule implies the following: $X \Rightarrow Y$, where $X \subset I$, $Y \subset I$, and $X \cap Y = \emptyset$.

Examples of association mining applications are, market basket analysis, medical diagnosis and research, web site navigation analysis and homeland security.

Measures of Association Mining

Support count: The support count of an itemset *X*, denoted by *X.count*, in a data set *T* is the number of transactions in *T* that contain *X*. Assume *T* has *n* transactions. Then

$$support = \frac{(X \cup Y).count}{n}$$

$$confidence = \frac{(X \cup Y).count}{X.count}$$

1.5 Structure of the thesis

Below is the summary of the rest of the chapters of the thesis:

Chapter 2: Literature Survey. This chapter introduces all the related work and survey of the Web crawlers and its approaches used for crawling.

Chapter 3: Problem Statement. In this chapter the gaps in the field of Web crawlers are described.

Chapter 4: Proposed Work and Experimental Analysis. Proposed methodology with proper example is explained in details in this chapter.

Chapter 5: Conclusion and Future Scope. The whole work presented in thesis is summarized in this chapter and it also contains the directions for the future research of the work.

Chapter 2
Literature Survey

Most likely, the largest scale analysis of Web page updates was carried out by Fetterly et al. [1]. System proposed by [1] crawled 151 million web pages one time a week, regularly for 11 weeks, and observed the alteration across the web pages.

The basic architecture of effective parallel crawler was given by J. Cho and H. Molina [8]. It was very crucial to use the crawling since the size of the web continuously increasing. The author proposed the architecture of the given system and then checked the problems related to the crawler which were parallel. Based on this understanding, author proposed design structure using a lot of data pages which were collected from the Internet.

E. Adar et al [10] provided description of algorithms, analysis, and structure for giving the evolution of content of web .Analysis which has been proposed provides dinner sight into the change of content from time to time. S. Sharma et al. [13] had given the design structure for a parallel crawler which included various crawling processes; called C-process. Each C-process will provide the important work that only a single crawler will do. It downloaded pages from the WWW, stored the pages in the local server, then extracted URLs and follows their links. The C-proc's executed these tasks on the same local network or at geographically different locations.

A.G. Kwang Leng et al. [14] proposed an algorithm based on the standard Breadth-First Search technique to create a web crawler named PyBot. Web Pages were downloaded and structured in Excel CSV format and ranking was made with the most popular web pages at the top of the list. S. Zheng [15] given a focused crawler analysis model, which was based on the combination of both the Genetic Algorithm and Ant Colony Algorithm

and known as the Genetic Algorithm-Ant Algorithm extended to achieve higher recall rate.

Lili Yan et al. [16] put forth Genetic Pagerank Algorithms used in calculation finding the optimum solutions. Andoena Balla et al. [17] proposed a technique for detection of the web crawlers in real time using decision trees and machine learning techniques. B. Saket and F. Behrang [18] proposed a method to precisely determine the quality of links, which were only accessible through an intermediary link, otherwise, not detectable. The author applied AntNet routing algorithm based on genetic algorithms (GA).

Anbukodi. S and Muthu M.K [19] presented a methodology, which used mobile agents to crawl the web pages that reduced the network load by decreasing the quantity of data transmitted over the provided network. K. S. Kim et al. [20] presented a dynamic web-data crawling methodologies, which included the sensitive examination of the web site changes and retrieval of web pages from the target websites dynamically.

Breadth first is suitable for shallower parts of a deeper tree and its performance was not so well where there were so many branches in a game tree, such as in the chess game, where a path approaches the similar objective with the similar length of the path [22] [23]. Andy yoo et al. [24] presented a technique using distributed Breadth First Search using Poisson random graphs to achieve high scalability. Y. Qin and D. Xu [28] presented an algorithm based on PageRank and also the page belief recommendation to aim the subjective needs of the users.

T. Chong [29] presented a page ranking algorithm by joining classified tree along with static PageRank algorithm enabling the classified tree to be modified according to the number of users reducing the problem of outdated pages, and increasing the effectiveness of the search. J. Kleinberg [30] presented the page ranking algorithm which was dynamic in nature. S. Qiao [31] modified the page rank algorithm by adding similarity measure derived from vector space model known as SimRank used for scoring the web pages.

Solutions to a given problem exist in a search algorithm but the methodology is to get the fittest solution in a specified time [32]. [33] proved that the genetic algorithm was suitable when there were no time constraints in searching a bulky database and it is also much efficient in getting the multimedia results. In contrast to traditional methods, genetic algorithms always worked on a whole population in place of starting from a single point [34].

In "A Novel Web Crawler Algorithm on Query based Approach with Increases Efficiency", a query based crawling approach using a filter has been proposed by the authors. The updated web pages were redirected by the filter and crawler downloaded all the altered web pages after the last visit [36]. In some cases, web pages of the similar topics do not direct to each other, therefore to get to the next related one it is necessary to go through several off-topic pages. Bergmark et al. [37] proposed a technique known as tunneling to crawl the bad pages to reach to good one. Two projects, named context-graph-based crawler and Cora's focused crawler, perform the tunneling. [38].

Zhang and Lu [39] used semi-supervised learning along with Q-learning to select the URL which was most relevant and based on scores from the unvisited list. These scores were calculated on the basis of the Q values of the unlabelled URLs and the fuzzy class memberships.

Chandramouli et al. and his companions [40] sequenced the URL queue associated within formation fetched from the Web logs. To produce the URL ordering researchers has used the popularity information from web logs. Bazarganigilan et al. [42] proposed a focused crawler technique that uses similarity function to give relevancy in the pages. They used genetic programming technique to find the best relation for estimation of the similarity test in the pages.

Huang et al. [43] proposed an intelligent focused crawler method that evaluated the relevance of pages to a particular domain with domain ontology and the hyperlinks to web pages in the domain. Yang and Hsu [44] proposed an ontology-support web focused-crawler (Onto Crawler) in which the keywords entered by the user were not able to

clarify the query requirements of the users. So, their technique used the domain ontology to give comparison and verification of the keywords to become the precise.

Pahal et al. [45] endeavored to refine the existing focused crawling by applying the approach with its context and context material, for fetching the web documents. Their focused crawler begins by crawling accustomed set of URLs. The URLs which were downloaded, were rated according to relevance measured. Relevance measure function which endeavored to trace the content of a Web page and its existent, already possessed context material to earn a total relevance score. The score provided was dependent on the level of interrelationship.

CORE[46], a focused crawler architecture through which the documents were retrieved based on their ontological relevancy score. Seed of words that were taken from the user initiated it. These seeds were used by CORE to fetch their URL from the Yahoo, Google and MSN search engines. These seeds URL were ranked in three classes: High, Medium, and Low. The documents were downloaded by the Crawler manager in accordance with URL priority queue and accumulated it in the document repertory. From documents in the Document Repository new URLs were extracted and the URLs with surrounding text which had a fixed number of letters leading and succeeding the hyperlink, the heading or sub-heading under which the hyperlink emerges were fetched. For each context link, the relevance score was calculated by the extracted link that was passed to ontology based relevant score. Then, to look ahead, the relevancy score, which was calculated from the adaptive rules to improve the crawling process.

Luong and his companions[47] fetched documents and information in a biological area by applying search engines and digital libraries. This had automatically filtered the documents that were truly compatible to the biological area of interest. Queries for each concept were automatically generated in the hierarchy. These queries were applied to a course of digital libraries and Web search engines. Researchers used SVM classification to separate out fetched documents that resemble the query nicely but which were less applicable to the area of amphibian morphology ontology. A set of documents applicable to amphibian morphology were formed. Their system begins by a manual ontology that was raised by mining the information fetched from the crawled documents.

For Association rule mining, an improved algorithm named Apriori was presented by Agrawal et al. in 1994 [48], which was more flexible. A new pruning technique and a different candidate generation method were employed. There were two processes in Apriori to search out whole major itemsets from the database. Initially a candidate itemset was generated and then the database was searched to analyze the definite support count of the matched itemsets. The support count was calculated in the first scanning and the major itemsets were developed by shortening the itemsets fallen below threshold that was predefined. Apriori was an effective algorithm for mining several itemsets for Boolean association rules[49].

Han et al. [50,51] designed a tree structure pattern mining algorithm named as FP-Tree algorithm (Frequent Pattern Tree). In the FP-Tree algorithm, FP-Tree building process preceeded the building of frequent patterns from the FP-Tree with a procedure called FP-growth. C. Hidber [52] proposed Continuous Association Rule Mining Algorithm (CARMA), an algorithm to compute huge itemsets on the web. The algorithm required maximum of two scans, In Phase-I, the algorithm in continuity built a lattice of all the potentially huge itemsets and Phase-II removed all the small itemsets, i.e. itemsets with maxSupport under the threshold specified by the last user.

Chapter 3
Gap Analysis and Problem Statement

3.1 Gap Analysis

From the analysis of the previous work done in the field of web crawling and web data mining, following gaps in this research area have been identified:

1. Web Crawler based on many languages is visible but not in python.
2. The output of the crawled data in form of graphs has not been visualized.
3. The resultant similarity charts on crawled data itself gives more relevant information along with the occurrence of similarity.

3.2 Problem Statement

As the size of data available on the World Wide Web is growing, searching the information is becoming more difficult task. As the data is increasing, the task of web crawlers, the programs responsible for the automatic retrieval of the web pages, is also increasing. The information processed by the Web crawlers is stored in the format of JSON (JavaScript Object Notation) documents. Since the number of pages retrieved by Web crawler is in millions, there is a need to find the association between web pages and this can be done with the use of an efficient data mining technique called association rule mining. In this thesis the frequent items are found using Apriori algorithm and association rules are formed using these frequent items. We have used a crawler that crawls a recipe site and proposed a technique to find out the similarity from the set of data related to the recipe items. Next association rules are predicted from the structured data of JSON file.

3.2.1 Objective

The objective here is:

1. To extract the information from web pages using a crawler.

2. To find the similarity using web mining.

3. To visualize the data.

4. To find similarity between the data sets.

Chapter 4

Implementation and Experimental Setup

This chapter starts by describing the steps used in developing the project, before explaining the system architecture, design and implementation details are presented for each of the major system components.

4.1 Proposed Technique

First we choose a seed URL because a Web crawler initiates with a list of URLs to visit, known as the seeds. So our first step is to provide seed URL to a crawler so that it can approach all of these URLs, it identifies every single of the hyperlinks in the webpage and put them in the list and then crawl the added link in the list.

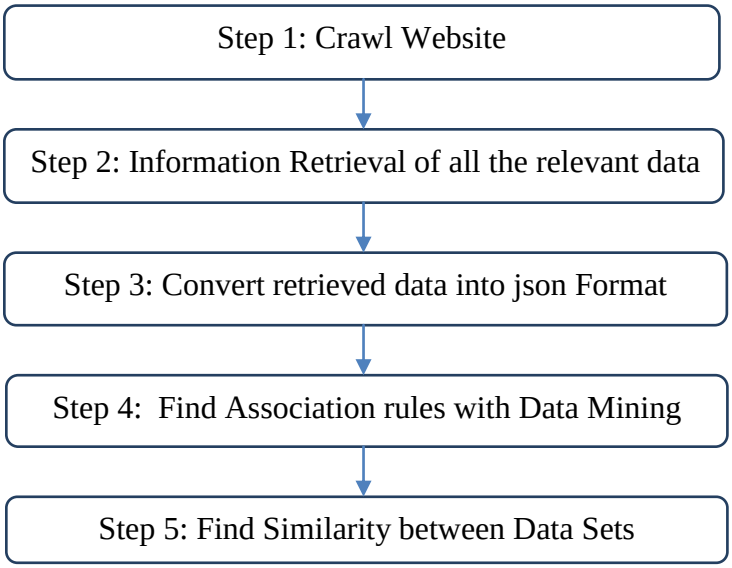

Figure 4.1 Steps involved in proposed technique

4.2 System Architecture and Implementation

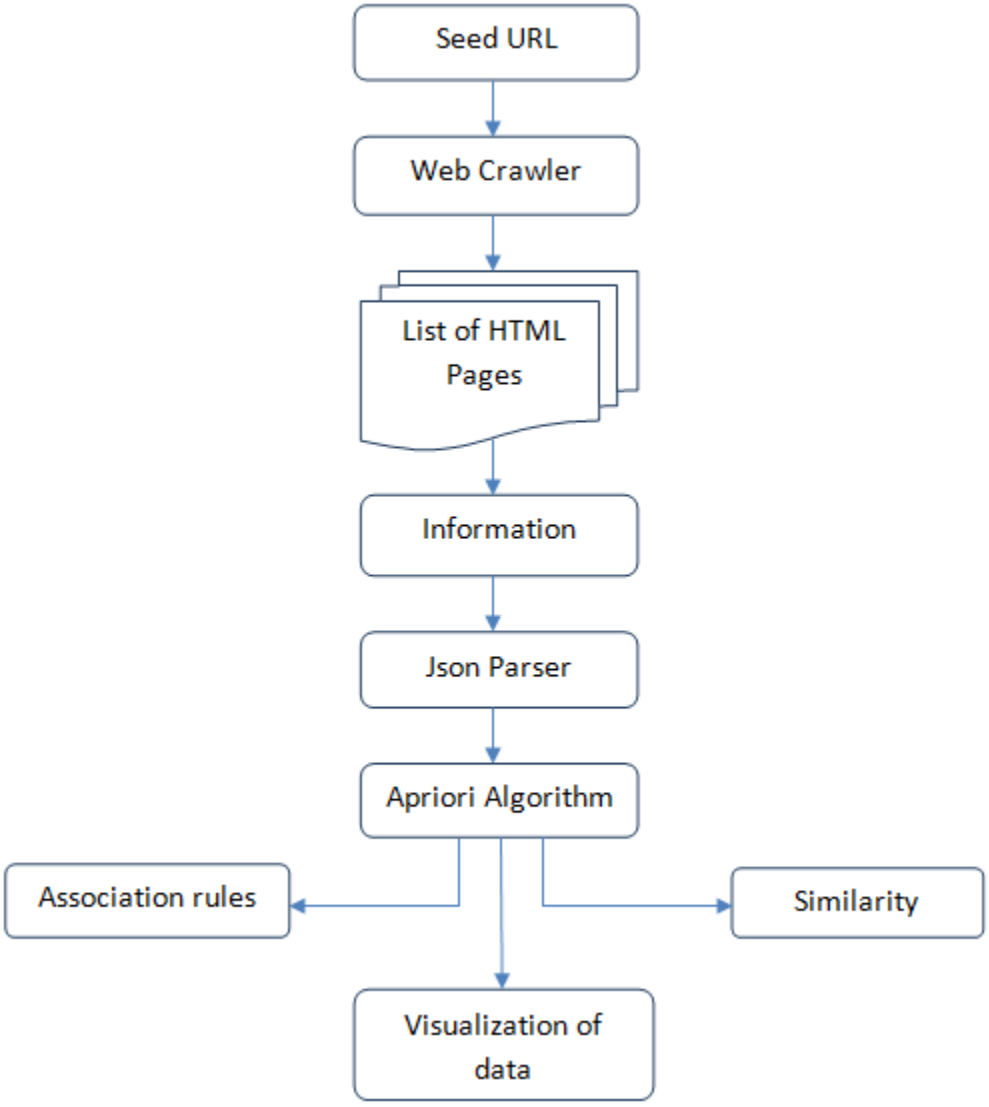

Figure 4.2 Architecture of Proposed Technique

When crawler completed the crawling it gives us the list of URLs then we start information retrieval by hitting one by one to list of URLs. Data is in html form and also noisy. Then we remove noise from data with help of regular expression. For interoperability and ease of usage we convert data in json type. Now data is json data which is simple to process. We can easily apply data mining to json data. With the help of data mining we can discover frequent pattern and association rules.

First a seed URL is selected. Then the list of URLs of recipes present is fetched. The desired information is fetched from html document and is stored in JSON document.

From the information in JSON document, we derive association rules, data visualization and similarity of data.

4.2.1 Seed URL

Seed URL is the starting URL which is fed to the crawler. As we have crawled the recipe websites, we have considered “www.sanjeevkapoor.com” as our seed URL.

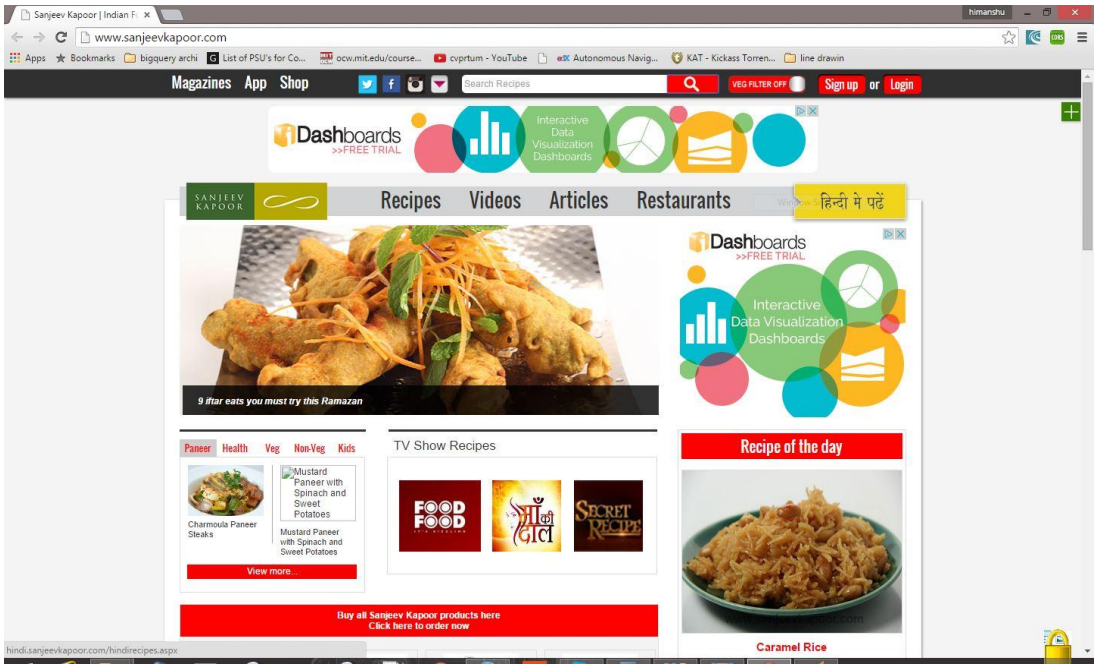

Figure 4.3 Seed URL

4.2.2 Recipe URL

Every document has three views: document view, HTML view and DOM. The HTML view of one of the recipes is shown in figure

After selecting the seed URL, the pages of this URL are visited and a list of URLs of recipes is retrieved by hitting upon by Python and stored in a text file. For performing this activity various open source crawlers can be used. Here we have used Scrapy.

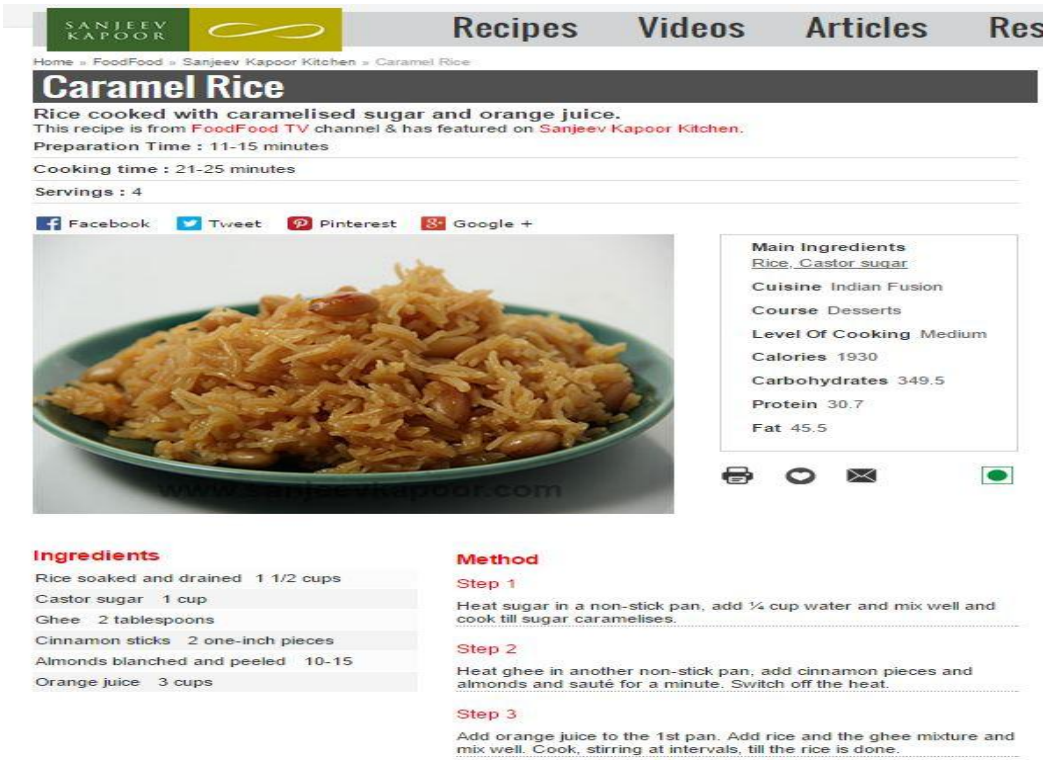

Figure 4.4 HTML view of Recipe page

4.2.3 Web crawler

Scrapy, a tool in python is used in this work. It is an open source as well as a collaborative framework which extracts the data needed from the websites in a fast, simple, and flexible way.

For crawling web sites, Scrapy is an application framework and importing structured data that can be useful for a long range of applications, such as, data mining, information processing or archival.

A spider has been developed to make list of recipes URL from seed URL .Spiders are the classes in which it is defined that how a particular site (or a set of sites) will be scraped, along with the method to operate the crawl (i.e. go after the links) and how to fetch structured data from the pages pointed by the links (i.e. scraping items). In short, Spiders are spot where the characteristic behavior for crawling and parsing pages for certain site (or a group of sites) is defined.

```python
from scrapy.selector import HtmlXPathSelector
from scrapy.spider import BaseSpider
from scrapy.http import Request

DOMAIN = 'sanjeevkapoor.com'
URL = 'http://%s' % DOMAIN
#sanjeevkapoor.com/Recipe/
class MySpider(BaseSpider):
    name = DOMAIN
    allowed_domains = [DOMAIN]
    start_urls = [
        URL
    ]

    def parse(self, response):
        hxs = HtmlXPathSelector(response)
        for url in hxs.select('//a/@href').extract():
            if not url.startswith('http://'):
                url= URL + url
            #print url

        if url.startswith('http://sanjeevkapoor.com/Recipe/'):
                print url
            yield Request(url, callback=self.parse)
```

Figure 4.5 Python spider code with Scrapy

In above code domain used is sanjeevkapoor.com. Above spider returns list of URL that starts with “http://sanjeevkapoor.com/Recipe”.

```
2015-06-02 01:11:42+0530 [sanjeevkapoor.com] DEBUG: Crawled (500) <GET http://sanjeevkapoor.com
/recipesearch.aspx?Search=pachadi&course=&cusine=&recipetitle=> (referer: http://www.sanjeevkap
oor.com/All-recipesearch.aspx?Search=pachadi&course=&cusine=)
2015-06-02 01:11:42+0530 [sanjeevkapoor.com] DEBUG: Ignoring response <500 http://sanjeevkapoor
.com/recipesearch.aspx?Search=pachadi&course=&cusine=&recipetitle=>: HTTP status code is not ha
ndled or not allowed
2015-06-02 01:11:42+0530 [sanjeevkapoor.com] INFO: Closing spider (finished)
2015-06-02 01:11:42+0530 [sanjeevkapoor.com] INFO: Dumping Scrapy stats:
        {'downloader/exception_count': 88,
         'downloader/exception_type_count/twisted.internet.error.DNSLookupError': 6,
         'downloader/exception_type_count/twisted.internet.error.TimeoutError': 81,
         'downloader/exception_type_count/twisted.web._newclient.ResponseNeverReceived': 1,
         'downloader/request_bytes': 8391620,
         'downloader/request_count': 19216,
         'downloader/request_method_count/GET': 19216,
         'downloader/response_bytes': 1577263343,
         'downloader/response_count': 19128,
         'downloader/response_status_count/200': 17506,
         'downloader/response_status_count/301': 9,
         'downloader/response_status_count/302': 15,
         'downloader/response_status_count/404': 200,
         'downloader/response_status_count/500': 1398,
         'dupefilter/filtered': 2042049,
         'finish_reason': 'finished',
         'finish_time': datetime.datetime(2015, 6, 1, 19, 41, 42, 680000),
         'log_count/DEBUG': 20399,
         'log_count/ERROR': 26,
         'log_count/INFO': 493,
         'offsite/domains': 48,
         'offsite/filtered': 462230,
         'request_depth_max': 57,
         'response_received_count': 18172,
         'scheduler/dequeued': 19216,
         'scheduler/dequeued/memory': 19216,
         'scheduler/enqueued': 19216,
         'scheduler/enqueued/memory': 19216,
         'spider_exceptions/AttributeError': 3,
         'spider_exceptions/UnicodeEncodeError': 21,
         'start_time': datetime.datetime(2015, 6, 1, 11, 34, 55, 393000)}
2015-06-02 01:11:42+0530 [sanjeevkapoor.com] INFO: Spider closed (finished)
```

Figure 4.6 Running scrapy on seed URL (sanjeevkapoor.com)

Our crawler took about 8 hours to crawl the seed URL(sanjeevkapoor.com). It crawled around 19000 pages.

Table 4.1 Overview of pages crawled

Number of pages	Status code	Code description
17506	200	OK
9	301	Moved Permanently
15	302	Found
200	404	Not Found
1398	500	Internal Server Error

4.2.4 List of URL's

Scrapy will return the list of recipe URLs which may or may not be unique. We have to choose only unique recipes URL. Spider has returned about 20,000 URLs of recipe but they are not unique. We have written a program to find unique URLs from that list. The unique recipies obtained are 1949 out of 20,000 recipe URLs given by scrapy.

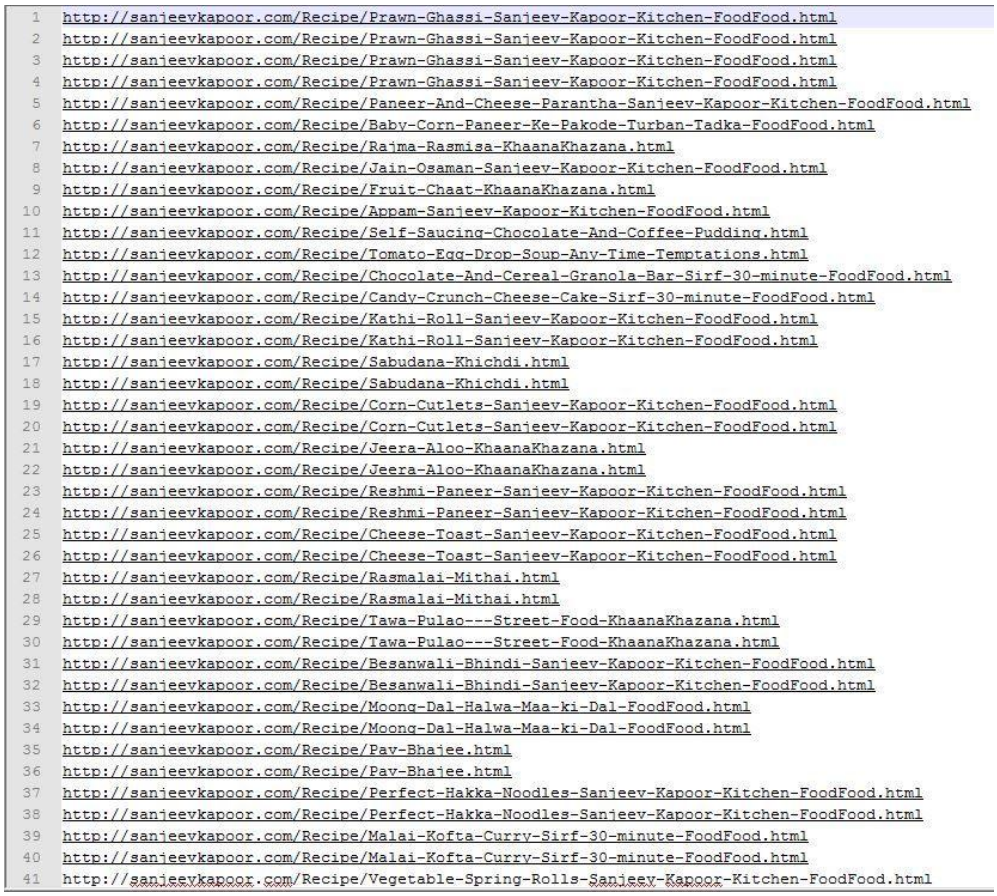

Figure 4.7 Non unique URLs returned by spider

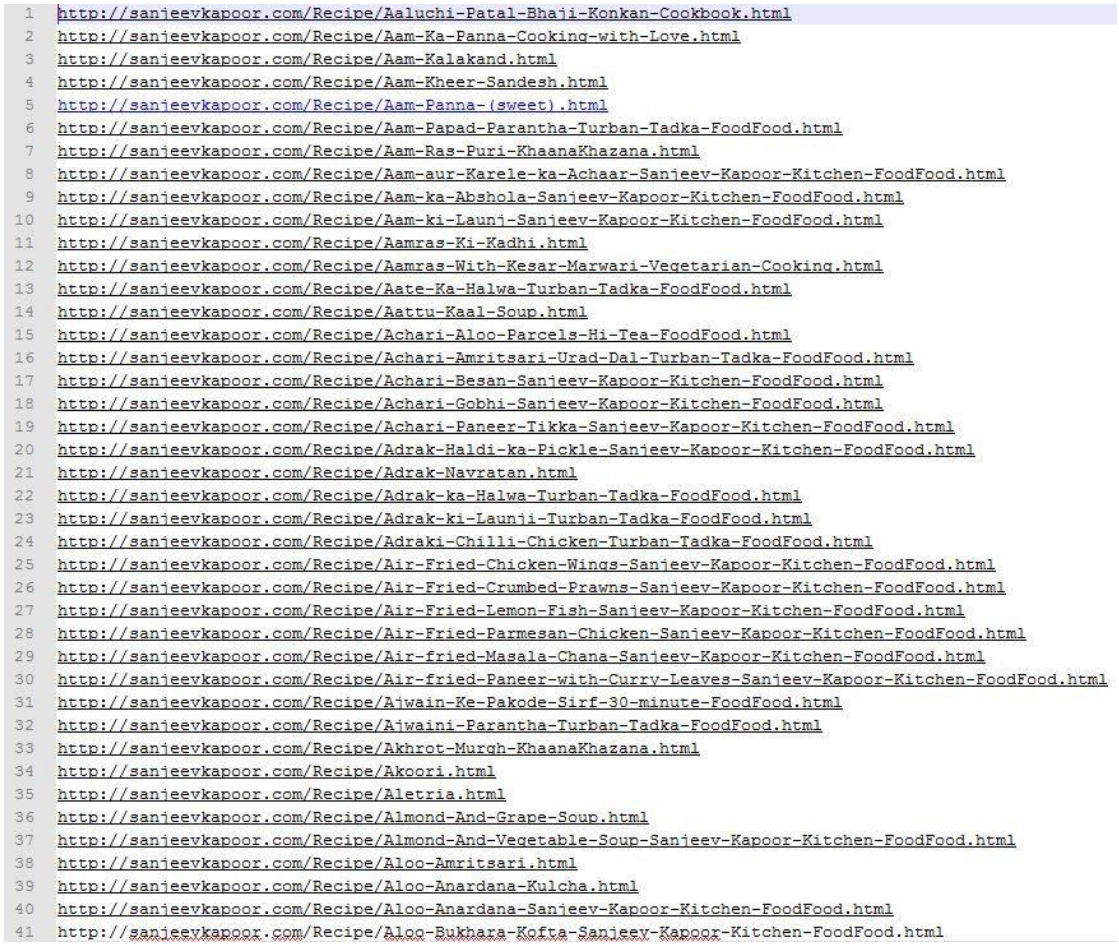

Figure 4.8 Processed URL (sorted and unique)

4.2.5 Information Scraper

Information scraper is a technique of extracting information from websites. The second is to identify and extract wanted information by creating cite specific wrappers, often called web scraping. Initially we have a list of URL. From this list html documents are retrieved by hitting with python. The raw html data is extracted from these html documents. Further the data is cleaned with the help of regular expressions. The obtained result is stored in JSON documents.

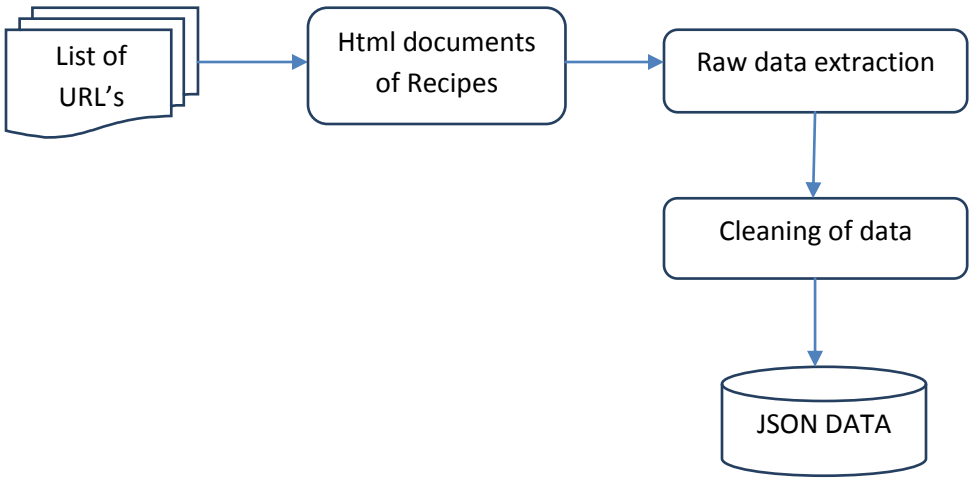

Figure 4.9 Process of information scrapper

```
<div class='recipeleftmain'>
    <div class='recipeleft'>
        <h1 id='headingh1' itemprop='name'>Caramel Rice</h1>
        <span itemprop='author' style='display:none;'>Saneev Kapoor</span>
        <div class='sponcerdiv'>
            <iframe scrolling='no'  id='external-frame' height='1'  src='../Sponcer.aspx?course=0'   frameborder='0'  ></iframe>
        </div>
        <div class='recipeshortdesc'>
            <p class='shrtdesc'>
                <span itemprop='description'>Rice cooked with caramelised sugar and orange juice.</span>
            </p>
            <p>
                <span> This recipe is from
                    <a href='../FoodFood'>  FoodFood TV  </a> channel & has featured on
                    <a href='../FoodFood-Shows/Sanjeev-Kapoor-Kitchen'> Sanjeev Kapoor Kitchen</a>.
                </span>
            </p>
            <h4 >
                <b>Preparation Time :</b>
                <meta itemprop='prepTime' content='PT15M'>11-15 minutes
                </h4>
                <h4>
                    <b>Cooking time :</b>
                    <meta itemprop='cookTime' content='PT25M'>21-25 minutes
                    </h4>
                    <h4>
                        <b>Servings :</b>
                        <span itemprop='recipeYield'>4</span>
                    </h4>
                </div>
                <div id='sharediv'>
                    <span class='st_facebook' displayText='Facebook'></span>
                    <span class='st_twitter' displayText='Tweet'></span>
                    <span class='st_pinterest' displayText='Pinterest'></span>
                    <span class='st_googleplus' displayText='Google +'></span>
                    <span class='st_whatsapp' displayText='WhatsApp'></span>
                </div>
```

Figure 4.10 Raw data containing information (prep time, cooking time, serving etc.)

To extract the clean data from raw data we have used regular expression in python. We have also used Beautiful Soup. Beautiful Soup is a library in Python for extracting data from HTML and XML files. It is compatible with every parser and provides typical methods of navigation, search, and modification of the parse trees. It usually saves a lot of time for programmers.

```
<html>
    <head>
        <title>
    The Dormouse 's story </title>
    </head>
    <body>
        <p class = "title">
            <b>The Dormouse 's story </b>
        </p>
        <p class = "story">
Once upon a time there were three little sisters;
and their names were
            <a class = "sister" href = "http://example.com/elsie" id = "link1">Elsie </a>,
            <a class = "sister" href = "http://example.com/lacie" id = "link2">Lacie </a>and
            <a class = "sister" href = "http://example.com/tillie" id = "link2">Tillie </a>;and they lived at the bottom of a well.
        </p>
        <p class = "story">... </p>
    </body>
</html>
```

Figure 4.11 HTML document

```
from bs4 import BeautifulSoup
soup = BeautifulSoup(html_doc, 'html.parser')

soup.title
# <title>The Dormouse's story</title>

soup.title.name
# u'title'

soup.title.string
# u'The Dormouse's story'

soup.title.parent.name
# u'head'

soup.p
# <p class="title"><b>The Dormouse's story</b></p>

soup.p['class']
# u'title'

soup.a
# <a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>

soup.find_all('a')
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>,
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</a>,
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie</a>]

soup.find(id="link3")
# <a class="sister" href="http://example.com/tillie" id="link3">Tillie</a>
```

Figure 4.12 Extracting data from HTML doc using beautiful soup

4.2.6 JSON parser

With the help of beautiful soup the data has been extracted, but this data is neither portable nor easy to access. So it is converted into json format.

4.2.6.1 JSON Advantages:

- **Simplicity**

  With respect to SGML, XML is relatively simpler, but JSON is much easier than XML. JSON has a much lesser grammar and applicable more precisely onto the data structures used in contemporary programming languages.

- **Extensibility**

  The reason for JSON being not extensible is that it does not require to be. It is not a document mark-up language, therefore, it is not mandatory to assign new tags or attributes to characterize data in it.

- **Interoperability**

  JSON has the same interoperability potential as XML. Openness JSON is at least as open as XML, perhaps more so because it is not in the centre of corporate/political standardization struggles.

Among the various modules of python, there is a JSON (Javascript Object Notation) module. To convert the data into json documents the JSON module of the python is used which is included in the python library.

```
import json

json.dumps(['foo', {'bar': ('baz', None, 1.0, 2)}])
'["foo", {"bar": ["baz", null, 1.0, 2]}]'

print(json.dumps("\"foo\bar"))
"\"foo\bar"

print(json.dumps(u'\u1234'))
"\u1234"

print(json.dumps('\\'))
"\\"

print(json.dumps({"c": 0, "b": 0, "a": 0}, sort_keys=True))
{"a": 0, "b": 0, "c": 0}

j = json.loads('{"one" : "1", "two" : "2", "three" : "3"}')
print j['two']
"2"


```

Figure 4.13 Handling json in python


```
"Jeera Aloo": {
    "Extra": {
        "Cooking time ": " 21-25 minutes",
        "Cuisine": "Punjabi",
        "Preparation Time ": " 11-15 minutes",
        "Course": "Main Course-Veg",
        "Carbohydrates": "6.4",
        "Fibers": "Potassium- 988",
        "Calories": "793",
        "Fat": "45.4",
        "Level Of Cooking": "Easy",
        "Servings ": " 4 ",
        "Protein": "90.4",
        "Main Ingredients": "Potatoes, Cumin seeds"
    },
    "Name": "Jeera Aloo",
    "Ingredients": {
        "Oil ": "4 tablespoons",
        "Fresh coriander leaves ": "2 tablespoons",
        "Dry mango powder (amchur) ": "1/2 teaspoon",
        "Salt ": " to taste",
        "Coriander seeds crushed": "1 tablespoon",
        "Roasted cumin powder ": "1 teaspoon",
        "Cumin seeds ": "1 teaspoon",
        "Red chilli powder ": "1 teaspoon",
        "Potatoes 1 inch pieces, boiled": "4 large"
    }
},
```


Figure 4.14 Json data of recipe

4.2.7 Data mining on json data

Before data mining we have to clean and pre-process data. The first and most important step in data mining is cleaning of data. The Preprocessed JSON data has been shown in figure 4.16.

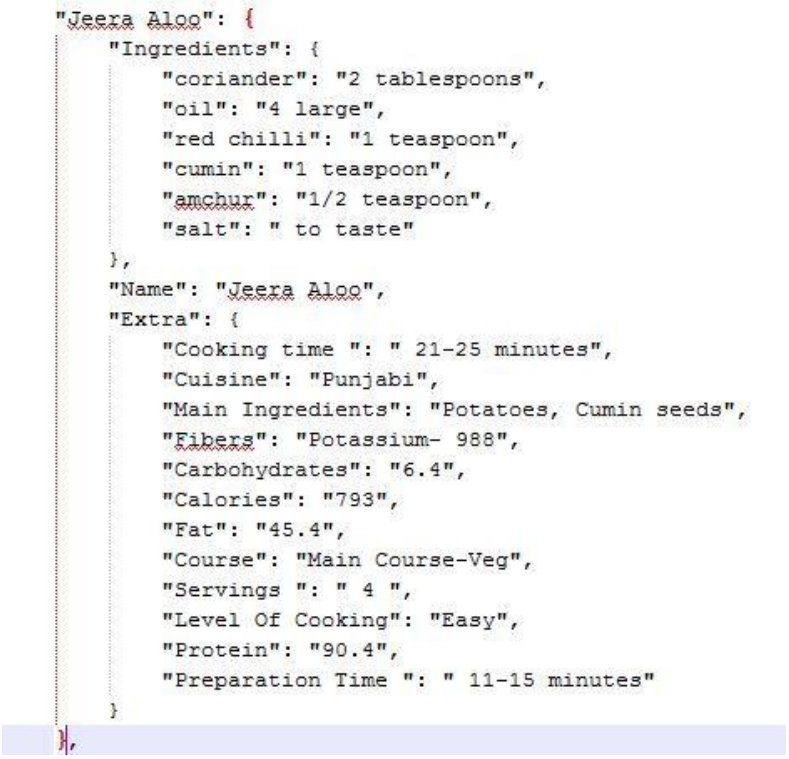

Figure 4.15 Pre-process json data

4.2.7.1 Association rules

Association rule is among the most widely used concepts of data mining. The aim of an association rule mining algorithm is to discover associations between data items. It is classically defined as: Consider I as a set of items i1, i2, . . . , in. Suppose T = t1, t2, . . . ,tn be a set of transactions. Every transaction t in T has its own unique id and consists of a subset of items in I. An association rule implies the following: $X \Rightarrow Y$, where $X \subset I$, $Y \subset I$, and $X \cap Y = \emptyset$.

The association rules for the recipes are as follows:

Total 1959 Recipes

Coriander ==> Salt #SUP: 555 #CONF: 0.96522
Cumin ==> Salt #SUP: 457 #CONF: 0.96211
Garlic ==> Oil #SUP: 378 #CONF: 0.82713
Garlic ==> Salt #SUP: 439 #CONF: 0.96061
Ginger ==> Salt #SUP: 444 #CONF: 0.93277
Green chilli ==> Salt #SUP: 537 #CONF: 0.96583
Onion ==> Oil #SUP: 542 #CONF: 0.80296
Red Chilli ==> Oil #SUP: 590 #CONF: 0.80054
Oil ==> Salt #SUP: 988 #CONF: 0.91228
Onion ==> Salt #SUP: 641 #CONF: 0.94963
Red Chilli ==> Salt #SUP: 708 #CONF: 0.96065
Turmeric ==> Salt #SUP: 427 #CONF: 0.97267
Coriander Oil ==> Salt #SUP: 440 #CONF: 0.96916
Green chilli, Oil ==> Salt #SUP: 403 #CONF: 0.96643
Onion, Salt ==> Oil #SUP: 515 #CONF: 0.80343
Oil, Onion ==> Salt #SUP: 515 #CONF: 0.95018
Red Chilli, Salt ==> Oil #SUP: 568 #CONF: 0.80226
Oil, Red Chilli ==> Salt #SUP: 568 #CONF: 0.96271

4.2.8 Visualization of data

The processed data needs to be stored in a human readable easy format. Pictorial representation is the best way to represent data in a easily readable form. We have shown the results in form of d3js graph in Figure 4.17.With help of d3.js graph we visualize the data.

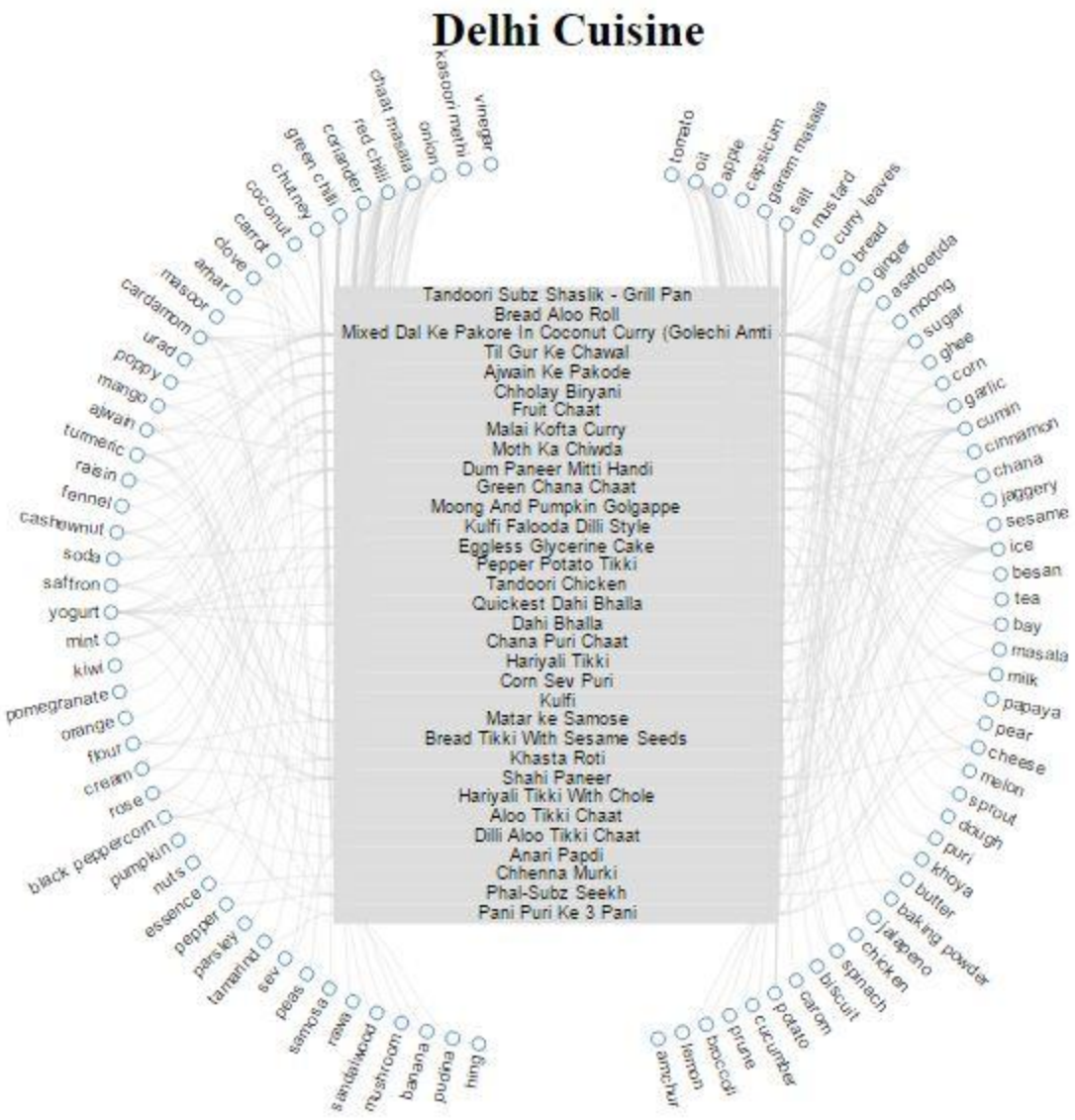

Figure 4.16 Graph depicting the ingredients and recipes of Delhi Cuisine

In figure 4.17 the ingredients and the recipes of Delhi Cuisine has been shown in a mesh. It is an overall view showing all ingredients and recipes but no link between them.

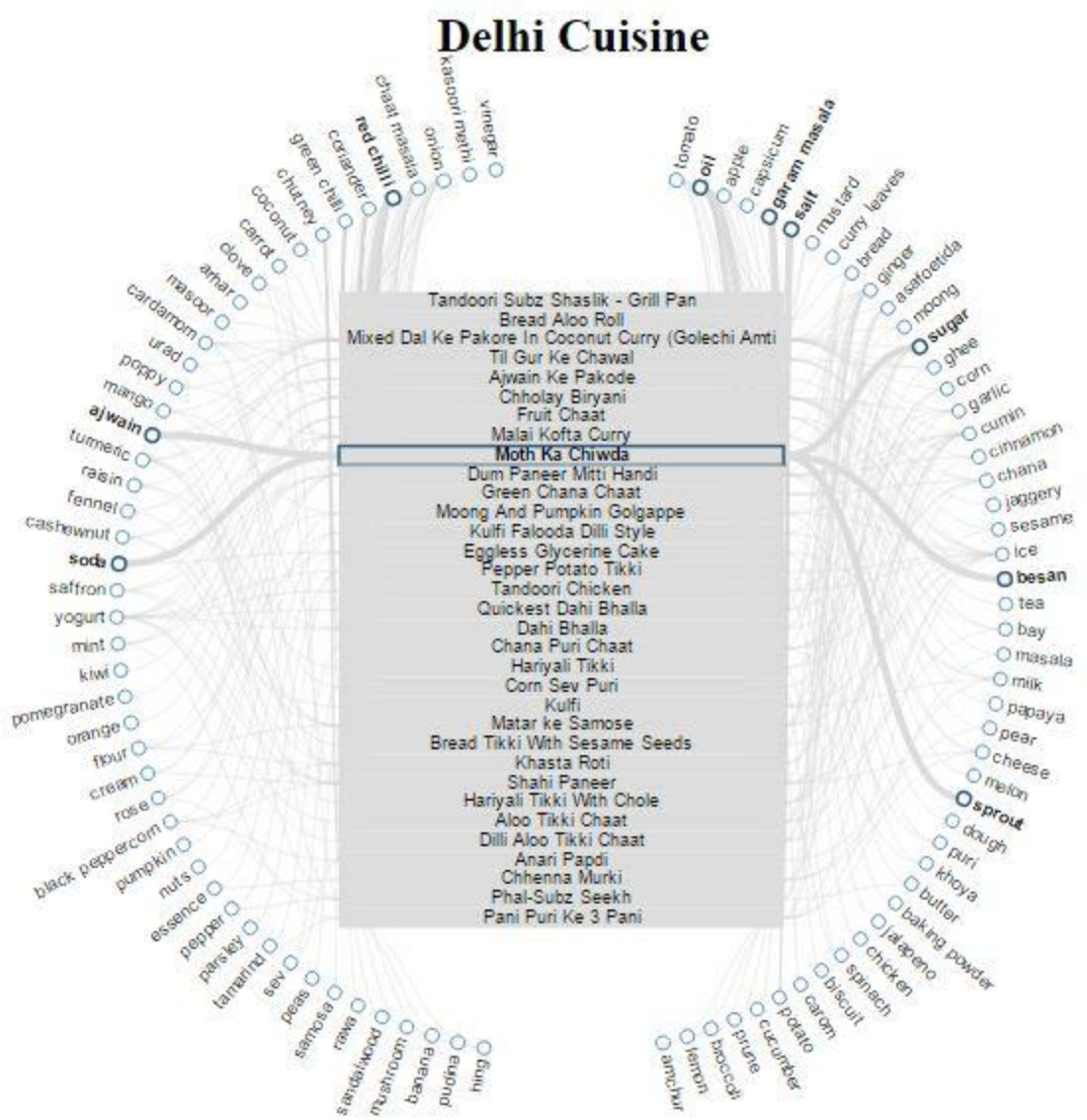

Figure 4.17 Ingredients in a particular recipe

The figure 4.18 shows the link between a particular recipe and the ingredients. Here we chose Moth ka Chiwda recipe, corresponding to which ajwain, sugar, sprouts, oil, salt, besan and garam masala.

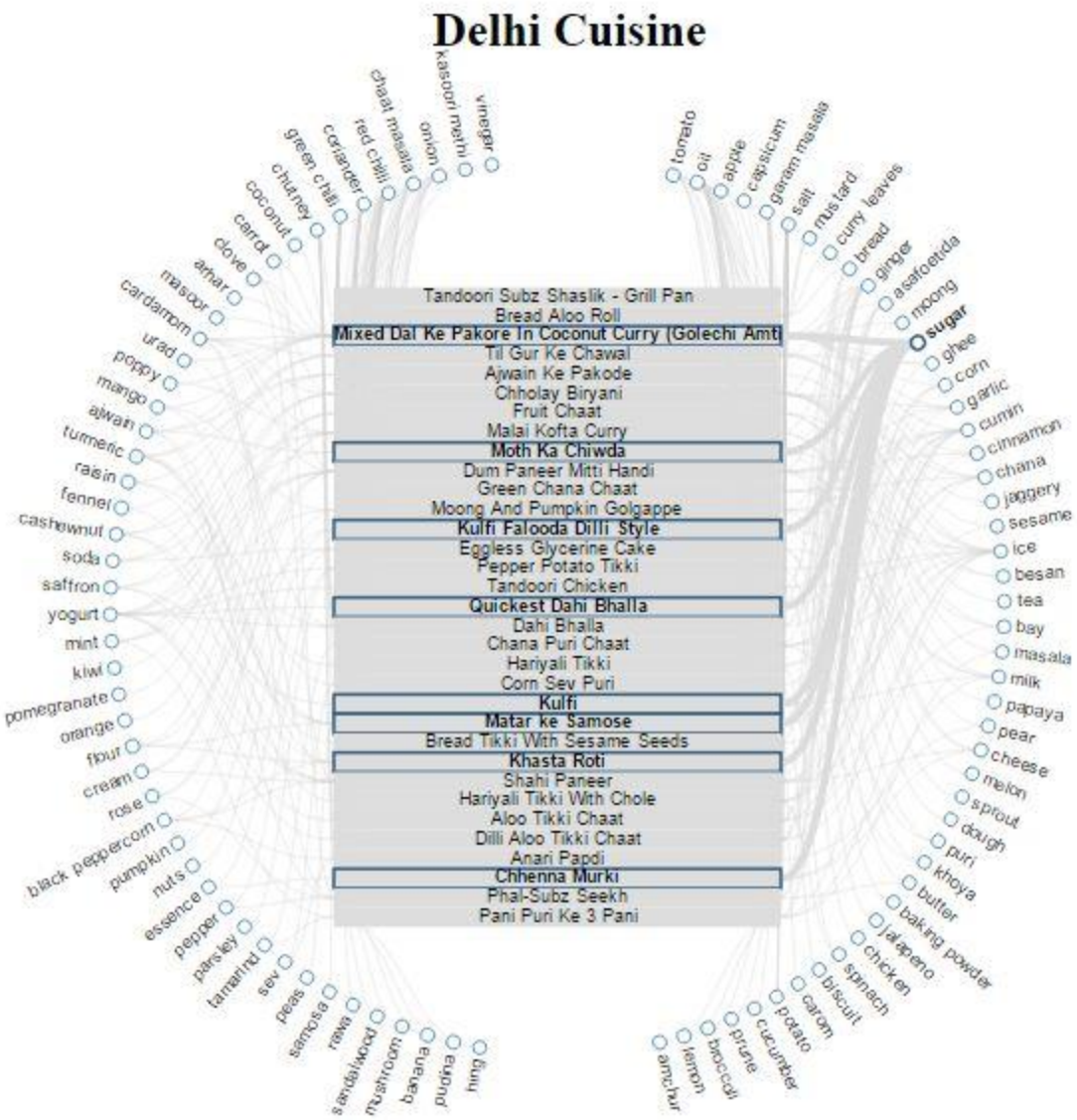

Figure 4.18 Relation of different recipes with particular ingredient.

The figure 4.19 shows the link between a particular ingredient and the recipes. Here we chose Sugar ingredient, corresponding to which Moth ka chidwa, Kulfi faluda, Kulfi and many more recipes are highlighted.

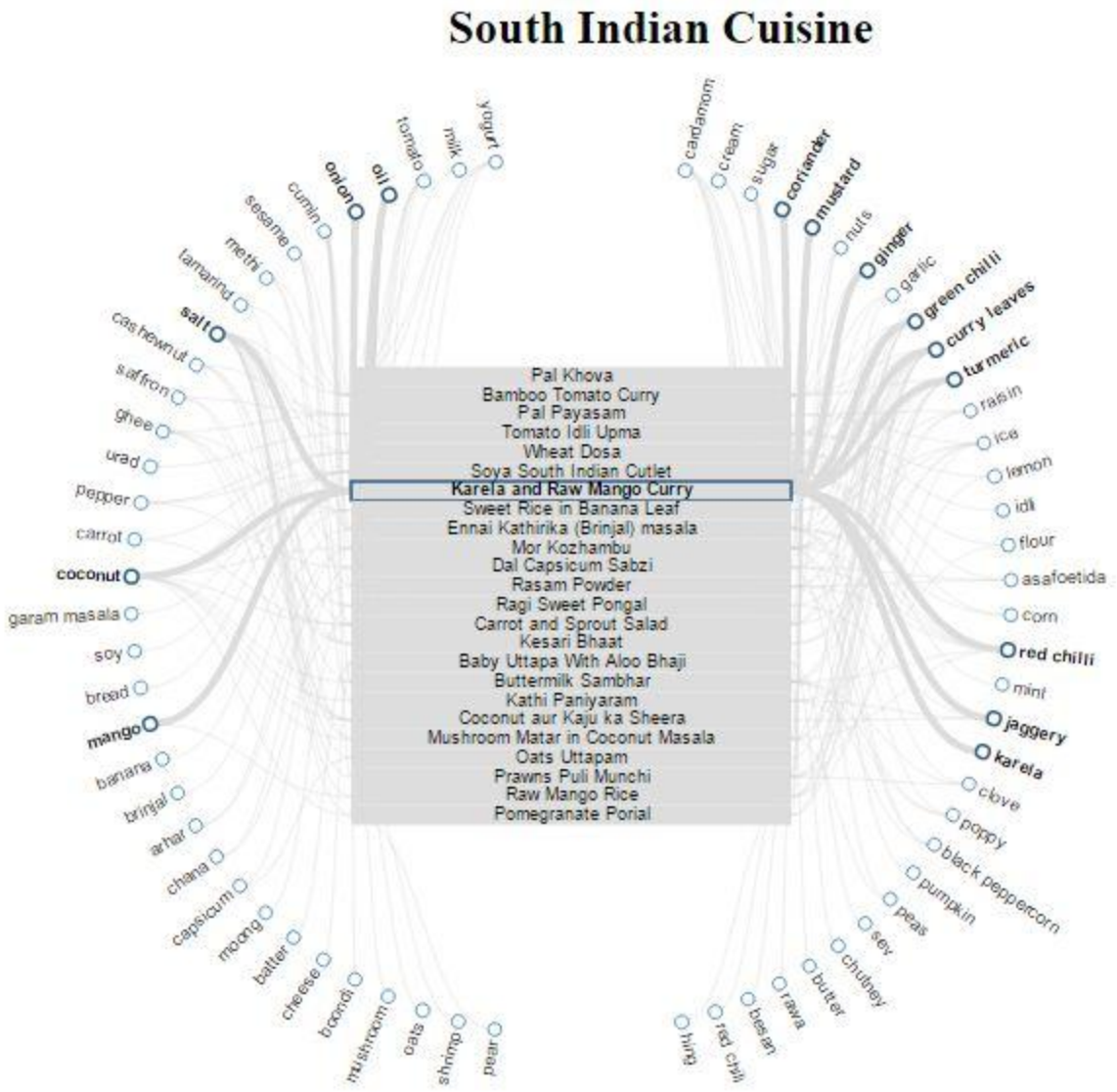

Figure 4.19 An example of South Indian Cuisine

4.2.9 Similarity between data sets

Choice of similarity or dissimilarity measures will depend on what kind of data one is handling and what exactly similarity/dissimilarity measures need to signify that is what the application that one is dealing with is. One can use various distance measures e.g. the conventional measures like Eucledian distance, Manhattan/city-block distance, Chebychev distance, Minkowsky distance. Besides, what we use often in image

processing are: Hausdorff Distance (the two data vectors will be considered close if every point of each set is close to some point of the other set), Bhattacharyya Distance (measure of similarity of two probability distributions), Bhattacharyya coefficient (measure of relative closeness of the two vectors) and Mahalanobis Distance (a specific case of Bhattacharyya Distance).

We have used SequenceMatcher to find similarity between recipes. SequenceMatcher is a type of flexible class used for correlating a combination of sequences of any kind, as far as the elements in the sequence are hashable. The basic algorithm precedes an algorithm published by Ratcliff and Obershelp named as "gestalt pattern matching". The motive is to discover the longest adjoining matching subsequence that carries no "junk" elements. The identical method is then applied iteratively to the segments of the sequences to the right as well as left of the matching subsequence.

296 Recipes are found with 80% or more similarity when SequenceMatcher is applied to our json data. Some examples are given below

- (Bhajanee-Thalipeeth , Bhee-Aloo-Tikki)
- (Bhajanee-Thalipeeth , Sichuan-Pakoda)
- (Oats-with-dried-fruits , Sweet-Rice-in-Banana-Leaf)
- (Pepper-Chicken , Mushroom-And-Spinach-Calzone)
- (Rasgulla-Phirni , Kesari-Phirni-with-Nutty-Caramel-Discs)

4.3 Experimental setup

The following tool has been used to perform this study:

4.3.1 PYCharm IDE

In the world of coding, Pycharm is an integrated development environment (IDE). It has a bottom workspace and a flexible plug-in system for adapting the environment. PyCharm’s intelligent code editor gives a first-class hold for Python, JavaScript, AngularJS, CoffeeScript, TypeScript, CSS, trendy template languages and much more. It also offers the advantage of language-aware completion of code, fault detection, and code

repairs. PyCharm gives a great framework-specific support for latest frameworks of web development such as Django, Flask, Google App Engine, Pyramid, and web2py. PyCharm runs well on Windows, Mac OS or Linux with a unique license key. It has a smooth workspace with customizable color layouts and key-bindings, with the availability of VIM emulation. Including Python, Pycharm supports JavaScript, CoffeeScript, HTML/CSS, Cython, Node.js, and other languages.

4.3.2 Python

Python is an extensively used general-purpose and high-level programming language. Its design ideology enlarges the code readability, and due to its syntax programmers can express the concepts in fewer lines of code as compared to other traditional programming languages. It provides constructs required to facilitate fluent programs of both small and large number of lines of code. Python is supportive to multiple programming paradigms, along with object-oriented, imperative and functional programming. One of its features is a dynamic kind of system and automatic memory management. Python has a huge and comprehensive standard library.

Similar in terms of other dynamic languages, Python is usually used as a scripting language, however, also used in a large range of non-scripting applications. With the third-party tools, like Py2exe or Pyinstaller, code of Python can be integrated into standalone executable programs. The interpreters of Python are available for almost all of the operating systems. CPython, which is the reference development of Python, is free of cost and is an open source tool. It has a development model which is entirely community based. CPython is handled by the non-profit Python Software Foundation.

4.3.3 D3js

D3.js(Data-Driven Documents) is a library based on Javascript for managing the documents related to data. D3 helps in making the data presentable using HTML, SVG, and CSS. D3's focus on web standards provides all the features of modern browsers, assembling effective visualization parts and a data-driven method to DOM

manipulation.It also allows binding random data to a Document Object Model (DOM), followed by implementing the data-driven changes to the document. For example, D3 can be used to build an HTML table from a given array of numbers.

D3 is not a monolithic platform that provides every possible feature, but, solves the crux of the problem which is effective management of documents related to data which provides extraordinary flexibility. It is compatible with web standards such as HTML, SVG, and CSS. D3 is extremely fast with minimum overhead, supporting huge datasets. D3's functional layout permits reuse of code through a diverse collection of extensions and plugins.

Chapter 5

Conclusion and Future scope

5.1 Conclusion

Internet, in present days, is among the most easily accessible sources for searching and approaching any type of data globally. The framework of the World Wide Web is a graphical framework, and from the associations given in a page other web pages can be retrieved. Web crawlers are the effective programs or software or tools that exercise the graphical framework of the Web to travel from one page to another. In this text, crawling process is briefly discussed and the association rules are obtained for the content present on the pages visited by the crawler. In a search engine, the crawler is the most significant module. The quality of the search of a search engine is directly affected by the quality and efficiency of a crawler.

The web crawler being a program visits the web pages in a way humans do, with the objective of validating, analyzing and visualizing the web pages. The association rules are presented in the form A ->B, which implies wherever there is an occurrence of A there is occurrence of B also.

In this thesis we have implemented a crawler for recipe website. It detects the entry URL first, here the entry URL is sanjeevkapoor.com and then processes the pages to find the subsequent URLs. It uses the sorted order to perform crawling. Then using Apriori algorithm, the association rules have been formed. These association rules help to give us the occurrence of two ingredients together.

5.2 Future Scope

In future:

1. The crawler could be extended to other sites like blog sites and forum sites.
2. The crawler can be used with the social networking sites.

Chapter 6

References

[1].Fetterly, D., M. Manasse, M. Najork, and J. Wiener. “A large-scale study of the evolution of Web pages”. WWW ‘03, Beizer B., “Software testing techniques”, Dreamtech Press, pp. 669-678, 2003.

[2].Kim, J. K., and S. H. Lee, “An empirical study of the change of Web pages” APWeb ‘05, pp. 632-642, 2005.

[3].Koehler, “W. Web page change and persistence: A four-year longitudinal study”. JASIST, 53(2), pp. 162-171, 2002

[4].Kwon, S. H., S. H. Lee, and S. J. Kim. “Effective criteria for Web page changes” , In Proceedings of APWeb ’06, pp. 837-842, 2006.

[5].Ntoulas, A., Cho, J., and Olston, C. “What’s new on the Web? The evolution of the Web from a search engine perspective”. WWW ’04, pp. 1-12, 2004.

[6].Olston, C. and Pandey, “Recrawl scheduling based on information longevity”, WWW ’08, pp. 437-446, 2008.

[7].Pitkow, J. and Pirolli, P., “Life, death, and lawfulness on the electronic frontier”, CHI ’97, pp. 383-390, 1997.

[8].Junghoo Cho and Hector Garcia-Molina “Parallel Crawlers”. Proceedings of the 11th international conference on World Wide Web WWW '02”,Honolulu, Hawaii, USA. ACM, 1-58113-449-5/02/0005,2002.

[9].RajashreeShettar, Dr. Shobha G, “Web Crawler On Client Machine”, Proceedings of the International MultiConference of Engineers and Computer Scientists 2008 Vol II IMECS, 2008,.

[10].Eytan Adar, Jaime Teevan, Susan T. Dumais and Jonathan L. Elsas “The Web Changes Everything: Understanding the Dynamics of Web Content”, ACM 2009.

[11].A. K. Sharma, J.P. Gupta and D. P. Agarwal “PARCAHYD: An Architecture of a Parallel Crawler based on Augmented Hypertext Documents”, International Journal of Advancements in Technology, pp. 270-283, October 2010.

[12].Ashutosh Dixit and Dr. A. K. Sharma, "A Mathematical Model for Crawler Revisit Frequency", IEEE 2nd International Advance Computing Conference, pp. 316-319, 2010.

[13].Shruti Sharma, A.K.Sharma and J.P.Gupta "A Novel Architecture of a Parallel Web Crawler", International Journal of Computer Applications (0975 – 8887) Volume 14– No.4, pp. 38-42, January 2011.

[14].Alex GohKwangLeng, Ravi Kumar P, Ashutosh Kumar Singh and Rajendra Kumar Dash "PyBot: An Algorithm for Web Crawling", IEEE 2011.

[15].Song Zheng, "Genetic and Ant Algorithms Based Focused Crawler Design", Second International Conference on Innovations in Bio-inspired Computing and Applications, pp. 374-378, 2011.

[16].Lili Yana, ZhanjiGuia, Wencai Dub and QingjuGuoa "An Improved PageRank Method based on Genetic Algorithm for Web Search", Procedia Engineering, pp. 2983-2987, Elsevier 2011.

[17].AndoenaBalla, Athena Stassopoulou and Marios D. Dikaiakos (2011), "Real-time Web Crawler Detection", 18th International Conference on Telecommunications, pp. 428-432, 2011.

[18].BahadorSaket and FarnazBehrang "A New Crawling Method Based on AntNet Genetic and Routing Algorithms", International Symposium on Computing, Communication, and Control, IACSIT Press, pp. 350-355, Singapore, 2011.

[19].Anbukodi.S and MuthuManickam.K "Reducing Web Crawler Overhead using Mobile Crawler", PROCEEDINGS OF ICETECT, pp. 926-932, 2011.

[20].K. S. Kim, K. Y. Kim, K. H. Lee, T. K. Kim, and W. S. Cho "Design and Implementation of Web Crawler Based on Dynamic Web Collection Cycle", pp. 562-566, IEEE 2012.

[21].Douglis, F., A. Feldmann, B. Krishnamurthy, and J. Mogul. Rate of change and other metrics: A live study of the World Wide Web. USENIX Symposium on Internet Technologies and Systems, 1997.

[22].Steven S. Skiena "The Algorithm design Manual" Second Edition, Springer Verlag London Limited, pp. 162,2008.

[23].Ben Coppin “Artificial Intelligence illuminated” Jones and Barlett Publishers, pp. 77, 2004.

[24].Andy Yoo,Edmond Chow, Keith Henderson, William McLendon, Bruce Hendrickson, ÄUmitCatalyÄurek “A Scalable Distributed Parallel Breadth-First Search Algorithm on BlueGene/L” ACM 2005.

[25].Alexander Shen “Algorithms and Programming: Problems and solutions” Second edition Springer pp. 135,2005.

[26].NarasinghDeo “Graph theory with applications to engineering and computer science” PHI, pp. 301,2004.

[27].Sergey Brin and Lawrence Page “Anatomy of a Large scale Hypertextual Web Search Engine” Proc. WWW conference 2004.

[28].Yongbin Qin and DaoyunXu “A Balanced Rank Algorithm Based on PageRank and Page Belief recommendation”.

[29].TIAN Chong “A Kind of Algorithm For Page Ranking Based on Classified Tree In Search Engine” Proc International Conference on Computer Application and System Modeling (ICCASM), 2010.

[30].J.Kleinberg “Authoritative sources in a hyperlinked environment”, Proc 9th ACM-SIAM Symposium on Discrete Algorithms, 1998.

[31].ShaojieQiao, Tianrui Li, Hong Li and Yan Zhu, Jing Peng, JiangtaoQiu “SimRank: A Page Rank Approach based on similarity measure” 2010 IEEE.

[32].S. N. Sivanandam, S. N. Deepa “Introduction to Genetic Algorithms” Springer, pp. 20,2008.

[33].S.N. Palod, DrS.K.Shrivastav,DrP.K.Purohit “Review of Genetic Algorithm based face recognition” International Journal of Engineering Science and Technology (IJEST) Vol. 3 No. 2, Feb 2011.

[34].Deep MalyaMukhopadhyay, Maricel O. Balitanas, AlisherovFarkhod A.,Seung-Hwan Jeon, and Debnath Bhattacharyya “Genetic Algorithm: A Tutorial Review”,International Journal of of Grid and Distributed Computing Vol.2, No.3, September, 2009.

[35].Mehdi Ravakhah, M. K. "Semantic Similarity BasedFocused Crawling" 'First International Conference on Computational Intelligence, Communication Systems and Networks', 2009.

[36].] S SVishwakarma, A Jain ,A K Sachan," A Novel Web Crawler Algorithm on Query based Approach with Increases Efficiency" International Journal of Computer Applications ,Volume 46– No.1,pp. 0975 – 8887, May 2012.

[37].Miladshokouhi, PiroozChubak, ZaynabRaeesy," Enhancing Focused Crawling with Genetic Algorithms," Information Technology: Coding and Computing, Volume 2, pp.503 – 508, 2005.

[38].C. Singh, Ramkala. Article: "Web Crawling Algorithms" pp. 161-165, ID-raictia-10 - 194.

[39].AH. Zhang and J. Lu, "SCTWC: An online semi-supervised clustering approach to topical web crawlers", in Applied Soft ComputingVol. 10, No. 2, pp. 490-495, 2010.

[40].A. Chandramouli, S. Gauch, and J. Eno, "A Cooperative Approach to Web Crawler URL Ordering", iIn Human Computer Systems Interaction, AISC 98, Part I, pp. 343–357, 2012.

[41].Xu and W. Zuo, "First-order Focused Crawling", pp. 1159-1160, 2007.

[42].M. Bazarganigilani, A. Syed and S. Burki, "Focused web crawling using decay concept and genetic programming", In International Journal of Data Mining & Knowledge Management Process (IJDKP) Vol.1, No.1, pp:1-12, 2011.

[43].W. Huang, L. Zhang, J. Zhang, M. Zhu, "Focused Crawling for Retrieving E-commerce Information Based on Learnable Ontology and Link Prediction" ieec, International Symposium on Information Engineering and Electronic Commerce, pp.574-579, 2009.

[44].S. Yang and C. Hsu, "An Ontology-Supported Web Focused-Crawler for Java Programs", Proc. of 2010 International Workshop on Mobile Systems, E-commerce, and Agent Technology, Jinhua, China, Jul. 5-6, pp. 266-271, , 2010.

[45].N. Pahal, N. Chauhan, and A.K. Sharma, "Context-Ontology Driven Focused Crawling of Web Documents", A.K. Wireless Communication and Sensor

Networks, 2007. WCSN apos:07. Third International Conference, pp.121-124, 13-15 Dec. 2007.

[46].M. Kumar and R. Vig, “Design of CORE: context ontology rule enhanced focused web crawler”, International Conference on Advances in Computing, Communication and Control (ICAC3) pp. 494-497, 2009.

[47].H. P. Luong, S. Gauch, and Q. Wang, “Ontology-Based Focused Crawling”, Information, Process, and Knowledge Management, 2009 (eKNOW '09) ,pp. 123-128, 1-7 Feb. 2009.

[48].Agrawal, R. and Srikant, R.,”Fast Algorithms for Mining Association rules”,Proc. 20th VLDB conference, Santiago, Chile, 1994.

[49].Luo Fang. “The Study on the Application of Data Mining based on Association Rules”, International Conference on Communication Systems and Network Technologies, pp. 477-480, 11-13 May 2012.

[50].Han, J. and Kamber, M., “Data Mining: Concepts and Techniques”, Morgan Kanufmann, San Francisco, CA USA, 2001.

[51].Han, J., Pei, J. and Yin, Y. “Mining frequent patterns without candidate generation”. ACM SIGMOD Intl. Conference on Management of Data, ACM Press, pp. 1-12, 2000.

[52].Christian Hidber. “Online Association rule mining”. SIGMOD ’99 Philadelphia PA. ACM 1-58113-084-8/99/05, 1999