

**R.V. COLLEGE OF ENGINEERING  
BANGALORE – 560059**

(Autonomous Institution Affiliated to VTU, Belgaum)



**IMPLEMENTATION OF MACHINE LEARNING  
ALGORITHMS FOR EFFICIENT CLASSIFICATION  
OF UNSTRUCTURED DATA FROM ONLINE COURSE  
WEB PAGES**

**PROJECT REPORT**

**Submitted by**

**PADMAVATHI KRISHNAMURTHY**

**1RV10CS062**

**MEGHANA H**

**1RV10CS052**

**CHINTALAPATI MALLIKA**

**1RV10CS050**

**Under the Guidance of  
Shanta Rangaswamy  
Assistant Professor,  
Department of Computer Science & Engineering,  
R.V. College of Engineering, Bangalore**

*In partial fulfillment for the award of degree*

*of*

***Bachelor of Engineering***

***in***

**COMPUTER SCIENCE AND ENGINEERING**

**2013-2014**

**R.V. COLLEGE OF ENGINEERING, BANGALORE - 560059**  
**(Autonomous Institution Affiliated to VTU, Belgaum)**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**



**CERTIFICATE**

Certified that the project work entitled “*Implementation of machine learning algorithms for efficient classification of unstructured data from online course web pages*” has been carried out by **Padmavathi Krishnamurthy (1RV10CS062)**, **Meghana H (1RV10CS052)** and **Mallika C (1RV10CS050)** a bonafide students of **R.V. College of Engineering, Bangalore** in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the **Visvesvaraya Technological University, Belgaum** during the year 2013-2014. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report deposited in the departmental library. The project report has been approved as it satisfies the academic requirement in respect of project work prescribed for the said degree.

**Shanta Rangaswamy**  
Assistant Professor  
Department of CSE,  
R.V.C.E., Bangalore –59

**Dr. G. Shobha**  
Head of Department,  
Department of CSE,  
R.V.C.E., Bangalore –59

**Dr. B. S. Satyanarayana**  
Principal,  
R.V.C.E.,  
Bangalore –59

**External Viva**

**Name of the Examiners**

**Signature with Date**

1. \_\_\_\_\_

\_\_\_\_\_

2. \_\_\_\_\_

\_\_\_\_\_

**R.V. COLLEGE OF ENGINEERING, BANGALORE - 560059**  
**(Autonomous Institution Affiliated to VTU)**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**DECLARATION**

We, **Padmavathi Krishnamurthy, Mallika C, Meghana H**, students of Eighth Semester B.E., in the Department of Computer Science and Engineering, R.V. College of Engineering, Bangalore declare that the project entitled “*Implementation of machine learning algorithms for efficient classification of unstructured data from online course web pages*” has been carried out by us and submitted in partial fulfillment of the course requirements for the award of degree in **Bachelor of Engineering in Computer Science and Engineering** during the academic year **2013 -2014**. We do declare that this work is not carried out by any other students for the award of degree in any other branch.

**Place: Bangalore**  
**Date:**

**Signature**

- 1. Padmavathi K**
- 2. Meghana H**
- 3. Mallika C**

## **ACKNOWLEDGEMENT**

Any achievement, be it scholastic or otherwise does not depend solely on the individual efforts but on the guidance, encouragement and cooperation of intellectuals, elders and friends. A number of personalities, in their own capacities have helped me in carrying out this project work. I would like to take this opportunity to thank them all.

First and foremost I would like to thank **Dr. B. S. Satyanarayana**, Principal, R.V.C.E, Bengaluru, for his moral support towards completing my project work.

I would like to thank **Dr.G.Shobha**, Head of Department, Computer Science & Engineering, R.V.C.E, Bengaluru, for her valuable suggestions and expert advice.

I deeply express my sincere gratitude to my guide Mrs. **Shanta Rangaswamy**, Asst. Prof, Department of CSE, R.V.C.E, Bengaluru, for her able guidance, regular source of encouragement and assistance throughout this project

I thank my Parents, and all the Faculty members of Department of Computer Science & Engineering for their constant support and encouragement.

Last, but not the least, I would like to thank my peers and friends who provided me with valuable suggestions to improve my project.

## **Abstract**

Innovation in the field of green technology by analyzing and developing efficient algorithms to reduce CPU utilization is the need of the hour. Managing the vast amount of online information is an important step towards this need. In the field of machine learning, binary classification algorithms are one such technique to process data. This data can have multiple sources and representations which influence the choice of classification technique. Data from web pages is unstructured in nature, and therefore requires many different processes and stages in classification. The aim of this project is two-fold: first, to compare important binary classification algorithms' performance in classifying samples of a large dataset from dynamic sources in the domain of online course websites, to identify valid online courses and secondly to implement the most precise algorithm in building an offline repository of all these courses for a user to access through an interactive front end.

The project involves multiple techniques such as crawling, URL filtering, metadata analysis, text extraction and processing while incorporating novel features such as social tagging from social network websites such as Delicious. A sample dataset covering all corner cases and representative examples is used for training a model classifier, which is then trained and tested using various methods like bootstrapping and cross validation. Finally the efficiency of the model is compared across five algorithms : Naive Bayes, K Nearest Neighbours, Decision Trees, Logistic Regression and Random Forest on datasets of increasing size and the most efficient one is used for classifying all the data collected from crawling. The relevant courses are put into an offline database made available to the users by an interactive and fresh user interface.

The parameter used to evaluate the algorithms was the accuracy of predictions on the test set, namely precision. It was observed that Random Forest algorithm outperforms Naive Bayes, K Nearest Neighbours, Decision Tree and Logistic Regression on this parameter. The accuracy observed upon evaluation of the test set for this classifier is 92.7%. Over 30,000 URLs were crawled and pre-processed using this classifier to create a repository of over 10,000 valid online courses.

# **Table of Contents**

<b>Acknowledgement</b>	<b>i</b>
<b>Abstract</b>	<b>ii</b>
<b>Table of Contents</b>	<b>iii</b>
<b>List of abbreviations</b>	<b>vii</b>
<b>List of Tables</b>	<b>viii</b>
<b>List of Figures</b>	<b>x</b>
<b>Chapter 1</b>	
<b>1    Introduction</b>	<b>1</b>
1.1 State of art developments	2
1.2 Motivation	13
1.3 Problem Statement	13
1.4 Objective	14
1.5 Scope	14
1.6 Methodology	14
1.7 Organization of the Report	15
1.8 Summary	16
<b>Chapter 2</b>	
<b>2    Software Requirements Specification of classification of unstructured data from online course web pages</b>	<b>17</b>
2.1 Overall Description	17
2.1.1 Product Perspective	17
2.1.2 Product Functions	18
2.1.3 User Characteristics	18
2.1.4 Constraints	18
2.1.5 Assumptions and Dependencies	19
2.2 Specific Requirements	19
2.2.1 Functionality Requirements	19
2.2.2 Performance Requirements	19
2.2.3 Supportability	20
2.2.4 Software Requirements	20
2.2.5 Hardware Requirements	20
2.2.6 Design Constraints	21
	<b>iii</b>

2.2.7	Interfaces	21
2.3	Other Non-functional Requirements	22
2.3.1	Performance Requirements	22
2.3.2	Safety and Security Requirements	22
2.3.3	Software Quality Attributes	23
2.4	Summary	24
<b>Chapter 3</b>		
<b>3</b>	<b>High level design of classification of unstructured data from online course web pages</b>	<b>25</b>
3.1	Design constraints	25
3.1.1	General constraints	25
3.1.2	Development methods	26
3.2.	Architectural strategies	26
3.2.1	Programming language	26
3.2.2	Future plans	27
3.2.3	Error detection and recovery	27
3.2.4	Data Storage management	27
3.2.5	Communication mechanism	27
3.3	System architecture	28
3.4	Data Flow Diagrams	28
3.4.1	DFD-level 0	28
3.4.2	DFD-level 1	29
3.4.3	DFD-level 2	30
3.5	Summary	30
<b>Chapter 4</b>		
<b>4</b>	<b>Detail design of classification of unstructured data from online course web pages</b>	<b>32</b>
4.1	Structure chart	32
4.2	Function description of modules	33
4.2.1	Crawler module	34
4.2.2	Featurizer module	35
4.2.3	Classifier module	37
4.2.4	Database module	38
4.2.5	User interface module	40
4.3	Algorithms	41
4.3.1	C4.5 algorithm	41
4.3.2	Logistic Regression algorithm	42
4.3.3	Naïve bayes algorithm	44
4.3.4	K nearest neighbour algorithm	45

4.3.5	Random Forest algorithm	45
4.3	Summary	46
<b>Chapter 5</b>		
<b>5</b>	<b>Implementation of classification of unstructured data from online course web pages</b>	<b>47</b>
5.1	Programming language selection	47
5.2	Platform selection	47
5.3	Code conventions	47
5.3.1	Naming conventions	48
5.3.2	File organizations	48
5.3.3	Properties declaration	49
5.3.4	Class declarations	49
5.3.5	Comments	49
5.4	Difficulties Encountered and Strategies Used to Tackle	50
5.4.1	Crawling and Classification Logic	50
5.4.2	Storage and Display Logic	51
5.5	Summary	51
<b>Chapter 6</b>		
<b>6</b>	<b>Software testing of classification of unstructured data from online course web pages</b>	<b>52</b>
6.1	Testing environment	52
6.2	Unit Testing of Main Modules	53
6.2.1	Unit Testing of Crawler Module	53
6.2.2	Unit Testing of Featurizer Module	54
6.2.3	Unit Testing of Classifier Module	55
6.2.4	Unit Testing of Database Module	56
6.2.5	Unit Testing of User Interface Module	57
6.3	Integration Testing Of Modules	57
6.4	System testing	59
6.5	Summary	60
<b>Chapter 7</b>		
<b>7</b>	<b>Experimental results and analysis of classification of unstructured data from online course web pages</b>	<b>61</b>
7.1	Evaluation metrics	61
7.2	Experimental dataset	62
7.3	Performance Analysis	62
7.3.1	C4.5 Algorithm Evaluation	63



7.3.2	Logistic Regression Algorithm Evaluation	65
7.3.3	Naive Bayes Algorithm Evaluation	68
7.3.4	K Nearest Neighbours Algorithm Evaluation	70
7.3.5	Random Forest Evaluation	73
7.4	Inference from the result	75
7.5	Summary	77
<b>Chapter 8</b>		
<b>8</b>	<b>Conclusion</b>	78
8.1	Limitation	78
8.2	Future enhancements	79
8.3	Summary	79
<b>References</b>		80
<b>Appendices</b>		85
Appendix A		85
Appendix B		96

## **List of abbreviations**

AI	:	Artificial Intelligence
ANFIS	:	Adaptive Neuro Fuzzy Inference System
FoCUS	:	Crawler Under Supervision
DOM	:	Document Object Module
NE	:	Named Entities
HMM	:	Hidden Markov Model
ME	:	Maximum Entropy
TBL	:	Transformation Based error-driven Learning
SVM	:	Support Vector Machine
JDBC	:	Java Database Connection
UDP	:	User Datagram Protocol
TCP	:	Transmission Control Protocol
OS	:	Operating System
HCL	:	Hardware Compatibility List
CPU	:	Central Processing Unit
MIPS	:	Million Instructions Per Second
GUI	:	Graphical User Interface
CSV	:	Common Separated Value
DBF	:	Database File
SQL	:	Structured Query Language
SDLC	:	Software Development Life Cycle
DFD	:	Data Flow Diagram
CLD	:	Context Level Diagram
API	:	Application Programming Interface
REST	:	Representational state transfer

## **List of Tables**

<b>Table No.</b>	<b>Title</b>	<b>Page No.</b>
<b>Table 6.1:</b>	Unit Test Case 1: Checks if Basic crawler works	53
<b>Table 6.2:</b>	Unit Test Case 2: Checks if multiple crawler works	54
<b>Table 6.3:</b>	Unit Test Case 3: Check if featurizer works	55
<b>Table 6.4:</b>	Unit Test Case 4: Check if featurizer works	55
<b>Table 6.5:</b>	Unit Test Case 5: Check if classifier works	56
<b>Table 6.6:</b>	Unit Test Case 6: Checks if Database module works	56
<b>Table 6.7:</b>	Unit Test Case 7: Checks if User Interface module works	57
<b>Table 6.8:</b>	Integrated test case 1: Test case to see if search option in the user interface works	58
<b>Table 6.9:</b>	Integrated test case 2: Test case to see if update option in the user interface works	58
<b>Table 6.10:</b>	Integrated test case 3: Test case to see if bookmark option in the user interface works	59
<b>Table 6.11:</b>	System Testing	60
<b>Table 7.1:</b>	Detailed accuracy by class on training set for C4.5 algorithm	63
<b>Table 7.2:</b>	Confusion matrix of training set for C4.5 algorithm	63
<b>Table 7.3:</b>	Detailed accuracy by class on test set for C4.5 algorithm	64
<b>Table 7.4:</b>	Confusion matrix of test set for C4.5 algorithm	64
<b>Table 7.5:</b>	Detailed accuracy by class on training set for Logistic Regression algorithm	65
<b>Table 7.6:</b>	Confusion matrix of training set for C4.5 for Logistic Regression algorithm	66
<b>Table 7.7:</b>	Detailed accuracy by class on test set for Logistic Regression algorithm	66
<b>Table 7.8:</b>	Confusion matrix of training test set for Logistic Regression algorithm	66

<b>Table 7.9:</b>	Detailed accuracy by class on training set for Naïve Bayes algorithm	68
<b>Table 7.10:</b>	Confusion matrix of training set for Naïve Bayes algorithm	68
<b>Table 7.11:</b>	Detailed accuracy by class on test set for Naïve Bayes algorithm	68
<b>Table 7.12:</b>	Confusion matrix of test set for Naïve Bayes algorithm	69
<b>Table 7.13:</b>	Detailed accuracy by class on training set for K-Nearest Neighbor algorithm	70
<b>Table 7.14:</b>	Confusion matrix of training set for k-Nearest Neighbor algorithm	71
<b>Table 7.15:</b>	Detailed accuracy by class on test set for K-Nearest Neighbor algorithm	71
<b>Table 7.16:</b>	Confusion matrix of test set for k-Nearest Neighbor algorithm	71
<b>Table 7.17:</b>	Detailed accuracy by class on training set Random Forest algorithm	73
<b>Table 7.18:</b>	Confusion matrix of training set for Random Forest algorithm	73
<b>Table 7.19:</b>	Detailed accuracy by class on test set for Random Forest algorithm	73
<b>Table 7.20:</b>	Confusion matrix of test set for Random Forest algorithm	74
<b>Table 7.21:</b>	Accuracies of the algorithms	76

## **List of Figures**

<b>Figure No.</b>	<b>Title</b>	<b>Page No.</b>
<b>Figure 1.1:</b>	General architecture of implementation of machine learning algorithms for efficient classification of unstructured data from online course web pages	2
<b>Figure 3.1:</b>	DFD-level 0 for online course webpage classification system	29
<b>Figure 3.2:</b>	DFD-level 1 for online course webpage classification system	29
<b>Figure 3.3:</b>	DFD-level 2 for online course webpage classification system	30
<b>Figure 4.2:</b>	Structure chart of implementation of machine learning algorithms for efficient classification of unstructured data from online course web pages	33
<b>Figure 4.3:</b>	Flow chart for crawler	35
<b>Figure 4.4:</b>	Flow chart for featurizer	36
<b>Figure 4.5:</b>	Flow chart for classifier	38
<b>Figure 4.6:</b>	Flowchart for database module	39
<b>Figure 4.7:</b>	Flow chart for user interface module	40
<b>Figure 4.8:</b>	The logistic function graph	44
<b>Figure 7.1:</b>	ROC curve of C4.5 classification algorithm for the class value “yes”	64
<b>Figure 7.2:</b>	ROC curve of C4.5 classification algorithm for the class value “no”	65
<b>Figure 7.3:</b>	ROC curve of Logistic Regression classification algorithm for the class value “no”	67
<b>Figure 7.4:</b>	ROC curve of Logistic Regression classification algorithm for the class value “yes”	67
<b>Figure 7.5:</b>	ROC curve of Naive Bayes classification algorithm for the class value ‘no’	69
<b>Figure 7.6:</b>	ROC curve of Naive Bayes classification algorithm for the class value ‘yes’	70
<b>Figure 7.7:</b>	ROC curve of K Nearest classification algorithm for the class value no’	72
<b>Figure 7.8:</b>	ROC curve of K Nearest classification algorithm for the class value ‘no’	72
<b>Figure 7.9:</b>	ROC curve of Random Forest classification algorithm for the class value ‘no’	74
<b>Figure 7.10:</b>	ROC curve of Random Forest classification algorithm for the class value ‘yes’	75



# Chapter 1

## Introduction

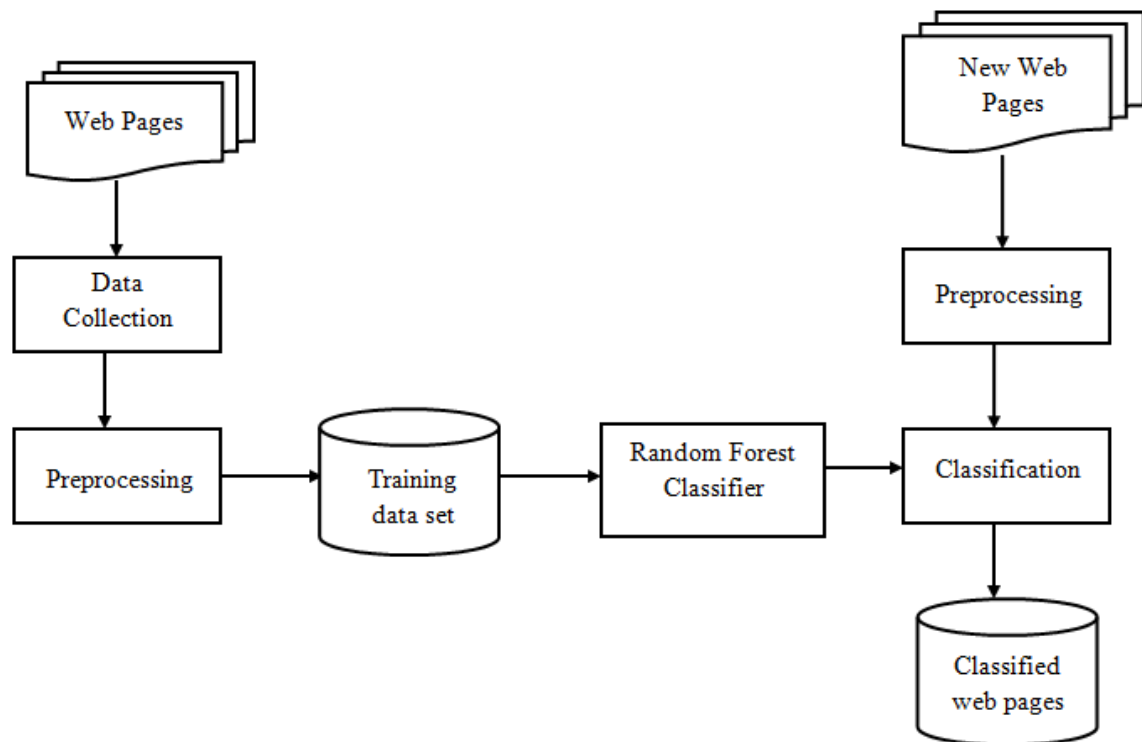
The project is a specific area of research in the domain of machine learning which deals with evaluation and comparison of various algorithms such as decision trees, naïve Bayesian network, K nearest neighbours, logistic regression and random forest ensemble for binary classification of data [1]. The type of data dealt with is extracted from web pages and is hence unstructured in nature. The project involves research into the process of automatically identifying valid online course web pages after web crawling, pre-processing and filtering of this data to extract features from the page source and its contents [2]. Previous efforts in this direction deal with smaller datasets of varying sources and use dissimilar metrics for comparison of the success of the various classification algorithms mentioned above [3]. This project seeks to successfully evaluate all of them across a single large dataset and a uniform metric, namely the precision of the algorithm which can be measured in terms of the number of correctly classified instances in the evaluation test set, as well as the area under the receiver operating characteristic curve for various threshold values of the positive example rate of classification of data [4].

Classification ultimately labels whether the page represents a valid online course and aims to create an offline repository of all such course pages. This enables a user to find and select the online training courses according to interest. An offline interface exists between the user and the system where the user enters his field of interest. The results obtained are a list of all the online courses available pertaining to that field. Thus, the aim is to help users across the world to select the courses of their interest.

The software has the following major architectural units which guide the working of the whole system: A fast multi-threaded java crawler to crawl web pages, a text parsing and regular expression based parser for extraction of features, efficient machine learning framework for classification of web pages using various machine learning algorithms, visualization tools for comparison of the machine learning algorithm results and finally a front end which allows the user to view any online course related information.

Figure 1.1 depicts the general architecture. The online course web pages are crawled from which data is collected for pre-processing. The pre-processing involves

mainly crawling, featurizing and classification, each of which will be dealt in detail in chapter 4. After the pre-processing a training set is created to train the classifier model. The trained model is then given test dataset as input which was extracted from new web pages [5]. The model then classifies the web pages into relevant online course web pages and stores the data in the repository.



**Figure 1.1: General architecture of implementation of machine learning algorithms for efficient classification of unstructured data from online course web pages**

## 1.1 State of art developments

The basis for all research in this project is found in [6], where authors describe one of the most novel and unique techniques used in webpage classification, as it is based on natural evolutionary concepts. The motivation of this paper is to perform feature selection based on the selectivity seen in fireflies. It describes a meta heuristic algorithm developed to classify web pages that is based on the evolutionary and biological behaviour of fireflies. This seeks to find the best 'n' features out of the hundreds that can be used for web classification. They observed improvements in speed and time by using this algorithm to reduce features. However, the process was long and cumbersome, involving a large amount of computational resources and therefore future work in this direction requires the assessment of the use of this algorithm and the various



optimizations which can be applied on it. It is merely a way of feature selection to reduce the dimensionality of algorithms used.

In order to carry out any such work, it is necessary to understand the applications of data mining tools like Weka [7] by applying K means clustering to find clusters from huge data sets and find the attributes that govern optimization of search engines. The motivation of this paper is to compare and evaluate various tools and visualizations used in classification in order to select the best one for use in this project. It is important for search engine to maintain a high quality websites. This will improve the optimization. Then, a database in which following attributes -length of title, keywords in title, domain length and number of back links and Top rank website is created. After applying K means using Weka, the result window shows the centroid of each cluster as well as statistics on the number and percentage of instances assigned to different clusters. The future work in this direction involves various manipulations of existing visualization algorithms in order to better understand results, and this was one of the main research gaps in this paper that the authors have used only one algorithm, therefore there was need for comparative study.

Various approaches have been tried over the years, like the efforts of [8] which introduce a shift from keyword-based representation to other perspective on representation of document's focus in form of key-concepts. This paper motivated a study of algorithms in the direction of words, concepts and intent based analysis, which requires a large amount of domain knowledge. The method is based on disambiguating the word senses using PageRank algorithm. The principle of this proposed approach is to do a two-pass ranking. So it is more accurate approach than keyword approach and its space efficient. However, there is scope for further research into graph processing algorithms which optimize storage and representation concepts without causing wastage of computational resources. There is also the research area of limited nodes in a graph, which needs to be explored to suit expanding dictionaries and vocabularies.

It is also necessary to have a quality and quantity metric such as in [9], where the author explains the way to create the website quality prediction models and nine quantitative web measures from different categories of Pixel Awards website are computed by the Web Metrics Analyzer tool. This helps to understand the various metrics used in predicting validity of a page the methodology employs the quantitative web page attributes (number of links, words etc.) to compare the goodness of the web pages and to

construct a quality prediction model utilizing Subtractive and FCM clustering model for predicting the class of website as good or bad. The methodology has 5 sections - Empirical Data Collection, web metrics analyzer, data processing, model building and data result analysis. One of the major results of this study is that ANFIS clustering techniques can be used to predict the quality of the website. This provides substantial improvement and the basic outline and structure followed in the methodology has been emulated in this project with the future improvement of dynamically completing the same, which this paper listed as a limitation.

A specific example of explicitly identifying webpage content can be seen in [10], where the authors describe the method to detect malicious web pages and to identify the specific threat types. This is a common application of web page classification and provides scope for common techniques and approaches used to be studied. The study suggests that KNN is better than SVM with 95.4% accuracy than SVM which has 92.3% accuracy. However, the method of comparison here uses a limited data set of only 4 types of web pages which is small in size and statically put together. This approach is extended to include dynamic crawling and classification, and compare multiple algorithms, one of each type in order to provide a better picture of the domain overview of webpage classification and machine learning algorithms.

A distinction needs to be made between the identification explained above and the general working of a search engine. This is seen in [11], the author focuses on estimating semantic similarity measures as an alternative to normal search engines, and the motivation of this paper is to study search hits of various patterns. The method is proposed via exploiting the page counts of two biomedical concepts returned by Google web search engine. The similarity scores of different patterns are evaluated, by adapting support vector machines for classification, to leverage the robustness of semantic similarity measures. The gaps in this paper include use of a single algorithm and hence the paper was studied with a view of understanding semantic search which can be used as an improvement in further iterations of the current system and provide multilingual support.

One of the applications of machine learning in webpage navigation is seen in [12] which deals with the application of Neural Networks for the Classification of Chinese web pages to develop a Web Information Navigation System. The motivation in studying

this paper is to understand how neural networks can be used in multilingual classification, which can be further explored as a research area. The quantum neural network has three layers: input layer, hidden layer, and output layer. The keywords of Web document are chosen as inputs of quantum neural network classifier. The quantum neural network classifying result and the manual work classifying result are compared, and there are 867 web documents rightly classified into correct subject class by quantum neural network classifier. The test result shows that the average system performance of the quantum neural network classifier is about 86.7%. The scope for future work includes use of a larger dataset with different types of samples which can be improved in precision over the current output.

Another application specific to forums is seen in [13] where the author presents Forum Crawler Under Supervision (FoCUS), a supervised web-scale forum crawler. The motivation of FoCUS is to crawl relevant content, i.e., user posts, from forums with minimal overhead. Forums exist in many different layouts or styles and are powered by a variety of forum software packages, but they always have implicit navigation paths to lead users from entry pages to thread pages and these can be exploited for efficient crawling. Forum threads contain information content that is the target of forum crawlers. This paper also shows how to learn accurate and effective regular expression patterns of implicit navigation paths from automatically created training sets using aggregated results from weak page type classifiers. Robust page type classifiers can be trained from as few as five annotated forums and applied to a large set of unseen forums. This can be applied in multiple domains with numerous applications and can also be extended to improve the precision of classifiers by including more data in the automatically created training sets.

Advances in dynamic HTML and webpage scripting requires advanced parsing such as in [14] which describes a system which uses vision and DOM based methods to apply page segmentation in obtaining academic data. This methodology consists of an algorithm which combines vision-based and DOM based segmentation methods, Bayesian network classification and post-processing. It can improve initial classification results by repairing wrong results and adding unclear results, this improves the result greatly: the precision is improved from 0.90 to 0.96, the recall is improved from 0.89 to 0.98, and the F1 is improved from 0.90 to 0.97. There is scope for further improvement since this system relies heavily on post processing which involves manual effort. Also,

the system is offline in nature because it relies on a static dataset of conference pages, which is a major research gap.

Along with HTML content parsing, it is required to extract intent related data as well. The work in [15] presents an algorithm is presented to extract keywords, which can be used to classify pages based on intent. The algorithm generates and extracts keywords from a poetry book. A matrix is formed in which words are number of columns and distiches are number of rows. The elements of matrix are filled by zero and one. The results indicate that “Love”, “Heart”, and “Eyes” are the most important keywords of the selected book with frequency 5%, 14%, and 9% respectively. This is able to tell us about the nature of content on the page and will be useful in large scale classification which involves text mining instead of conventional algorithms. Sparse representation of matrix and efficient processing of high dimensionality matrices need to be explored.

Other methods involving structured data include schemas as shown in [16] where the author explores the use of formal source code structure for classifying a large collection of the web content. The motivation was to focus on use of schemas collection Schema.org to classify web pages and categorize them unambiguously. Future work included implementation of this concept in larger domains and using multiple keywords.

In the absence of structured data, it is required to have structural units. An effort in [17] incorporated named entities as feature for web page classification. Named Entities (NEs) are phrases that contain names of persons, organizations, locations, numeric expressions including time, date, money, percent expressions and names of other miscellaneous things. The motivation to research this topic is that it gives better results for narrow domains. Some relevant classification algorithms include HMMs, Maximum Entropy (ME), Transformation Based error-driven Learning (TBL), SVMs and Conditional Random Fields (CRF). The technique used is information gain (IG) feature selection to reduce the dimensionality. Future work in this domain involves distinction between Unicode characters and extended character sets so that this can be applied in websites which do not have plain text.

Structural units are easier to categorize with link based information that can simplify parsing within a domain. For example, in [18] the author presents a web page classification algorithm, Link Information Categorization (LIC) based on the K nearest

neighbor method, it combines information on the website features, to implement the Web page link to information classification. It was seen LIC is more suitable for Web information classification, especially for professional websites, it has better classification results than other traditional classification algorithms. Future work involves implementing LIC in systems which have completely unlabelled data.

A complete project based on multiple previous efforts can be seen in [19] the author focuses on categorizing product pages on the Web depending on their information. Naive Bayes and the complement naive Bayes classifier are used. The experiments showed that the product pages can be classified most correctly depending on only the nouns of the titles of the product pages. It was found that the complement naive Bayes classifier outperformed the naive Bayes classifier and the future work involved using pictorial product catalogues for classification as well, by combining image processing related concepts.

The above work can be further refined by experimenting with the algorithm and mathematical parameters. An example is [20] where the author uses Naive Bayes Expectation Maximization (EM) algorithm classification method (using hierarchical clustering EM framework) that trains the Naive Bayes classifier iteratively. It is used to classify the massive unlabeled data iteratively until all labelled data (pages) has been classified completely. The proposed M-EM algorithm can produce high classification accuracy. With the increase in number of labelled pages, F I-score ( F I-score is an important indicator to assess the performance of classification result) value in both algorithms also became larger. However, the algorithm performance would be stable or even decline when the labelled data increased to a certain number. Therefore future work involves finding the inflection point in terms of size of dataset for which this algorithm is favourable.

Classification can also be done using a focused crawler [21], which is a web crawler that attempts to download only web pages that are relevant to a predefined topic or set of topics. In order to determine a web page is about a particular topic, focused crawlers use classification techniques. In this study the motivation is the classification of links instead of downloaded web pages to determine relevancy. Naïve Bayes classifier is combined for classification of URLs with a simple URL scoring optimization to improve the system performance. Focused crawlers use page scoring and link scoring techniques.

The results demonstrate that the link scoring method based on the Naïve Bayes (NB) classifier increases accuracy of the system considerably. Based on these results, the modified NB link classifier to incrementally update its training set during crawling will improve the system performance further.

The best features for web page classification problem, accuracy and run time performance of the classifiers can be improved by using Genetic Algorithm (GA) [22]. GA is important because it apparently attempts to decrease the feature space and determines the best features of the given set of web pages. It is found that when features selected by their genetic algorithm are used and a KNN classifier is employed, the accuracy improves up to 96%. However, since this is time consuming and computationally intense, future work aims to make it more efficient.

Malicious web pages can be detected using a two-stage classification model [23] which uses something called static and run-time features which are defined and used efficiently in each stage based on their values. It happens in 2 classification stages. The operation flow basically starts with a list of URLs which are fed into the static feature extractor which extracts the properties of the web page. This is then fed into the first stage of classification to estimate the maliciousness or basically categorize the web pages into potential malicious and benign one. Research gap includes work on both static or runtime features for classifying.

The unstructured data from web pages can be transformed into structured relations using a relation predictor to filter out irrelevant pages so that speed of the overall information extraction process is increased substantially. SVM [24] approach is used for this purpose. The evaluation pages on a sentence level, where each sentence is transformed into a token representation of shallow text features. The concept of prediction + extraction speeds up the process of extraction by a factor of 2 and research gaps include text extraction to remove advertisements, spam and other unwanted data.

One of the application of Neural Networks is the Classification of Chinese web pages to develop a Web Information Navigation System [25]. The quantum neural network has three layers: input layer, hidden layer, and output layer. The keywords of Web document are chosen as inputs of quantum neural network classifier. The test result shows that the average system performance of our quantum neural network classifier is

about 86.7%. Future work consists of six main parts: information collecting part, web document indexing part, web document classifying part, hyperlink depositing part, information inquiring part, and hyperlink updating part.

Another web page classification algorithm which can be used in the E-government system combines the Support Vector Machine (SVM) and the Unsupervised Clustering (UC) methods, called Combined UC and SVM algorithm (CUCS) [26]. The CUCS improves the speed and accuracy of the web page classification through pruning the training set. The accuracy rate, CUCS is much higher than UC, slightly higher than SVM. At the point of the time cost, UC is shortest, SVM is longest, and CUCS is in between, far less than SVM because the pruned training set reduces the training time complexity. This algorithm is implemented in E-government system theoretically and future work includes practical implementation.

The analogies of students' behaviour and internet usage pattern related to academic issues can be surveyed. Based on the classification scheme [27], out of 7000 visited websites by students during one year's data collection period, 36% of websites are classified as ACademic (AC) and remaining (64%) are classified as Non-ACademic websites (NAC). In the category of NAC visited websites, results presented that, the most visited websites belongs to Entertainment (43%) and the next most favourite websites were mentioned Social Networks (37%). The survey results of BTECH students, declared that department of computer science having majority of users and department of mathematic and civil having minimum number of users among all departments.

The three techniques generally used in classification are Naïve Bayes (NB), Decision Trees (DT) and Neural Networks (NN). NB [28] models are popular in machine learning applications, due to their simplicity in allowing each attribute to contribute towards the final decision equally and independently from the other attributes. This enhanced NB classifier has 3 stages - Crawler, Trainer and NB classifier with a common Indexer. The results showed that the enhanced NB classifier was comfortably in the lead by over 7% in both accuracy and F-Measure value. The NB classifier achieved the highest accuracy (97.89%), precision (99.20%), recall (98.61%) and F-Measure (98.90%) values, however, the DT classifier achieved the fastest execution time. However, the NB classifier achieves higher accuracy, precision and most importantly, overall F-measure value.

Social tags from social networking sites like Facebook, Twitter, MySpace can be used in classification of web pages. Social tagging [29] is a process in which many users add metadata to a shared content. Through the past few years, the popularity of social tagging has grown on the web and this motivates further research. In this paper the use of social tags for web page classification: adding new web pages to an existing web directory, has been investigated. The future work is based on applying different automatic approaches of using social tags for extending web directories with new URLs.

Another highly efficient classifier called Random Forest (RF) [30] classifier that can classify web pages efficiently according to their corresponding class without using other feature selection methods is generally used for large data sets. Random Forest classifier grows many classification trees. Each tree is trained on a bootstrapped sample of training data and at each node the algorithm only search across a random subset of the variables to determine a split. To classify an input vector in random forests, the vector is submitted as an input to each of the trees in the forest. Each tree gives classification, and it is said that the tree votes for that class. In the classification, the forest chooses the class having the most votes. The experiments have shown that the proposed approach is suitable for the multi-category web page classification.

A web-crawled academic video search engine, named as LeeDeo [31] that can search, crawl, archive, index, and browse academic videos from the Web, provides an alternative for other popular academic search engines like CiteSeer and Google Scholar. They built a proof-of-concept system and demonstrated how their classification technique worked. There were multiple gaps in this paper which provide scope for future research and work, such as extension of techniques used to multiple forums and across various languages, domains and providers. A focus on images and video grabs instead of complete videos is also suggested.

Another algorithm for web page classification called CUCS (Combined UC and SVM) [32] is used when the training set is large. The motivation is that it provides an automatic web page classification method to help users locate the required information on the internet in a convenient way. CUCS combines the advantages of SVM (Support Vector Machine) and UC (Unsupervised Clustering), achieving high precision and fast speed. In precision, CUCS is far higher than UC and a little better than SVM. As to time



efficiency, CUCS costs more time than UC, which form the research gap of this paper. The future work is to improve the time efficiency of CUCS algorithm.

The web pages can be classified automatically using Rough Set Theory [33]. The motivation provided is to overcome the difficulty in mining high dimensional data because of the curse of dimensionality. The definition of a set of attributes consists of one conditional attribute which is evaluated on the basis of the others. The rules for classification are extracted iteratively by choosing different attributes as the conditional so that we can find which ones are superfluous and eliminate them. Experimental results using a dataset from [www.sohu.com](http://www.sohu.com) show that the precision and recall of this method is much higher. The way this is achieved is by noting the decision weight age given to different conditional variables in the VSM (Vector Space Model). The research gap is that this classification effect is not very good for the web page classifications in which class partitions are not obvious.

The identification of web pages, by making use of Intensive Explosive Devices or IEDs [34] to detect the malicious use by terrorists and extremists is the prime need of the hour by various security organizations. The motivation is that it provides an improvised explosive device web pages representing a significant source of knowledge. Here the approach used is to make a lexicon of 100 Arab words related to IEDs and then crawl specifically those websites which are mentioned in known sources and intelligence agencies as potential targets for terrorists. In this way Focused Crawling is used, an approach to select and narrow the search space. Three types of features were used- stylistic features deal with words, vocabulary, richness, frequency etc, while structural features deal with HTML tag elements like number of paragraphs. The topical features used include bag of words model etc. and this forms an extended feature set. Classification accuracy exceeded 88%. This paper provides only a framework for genre classification of IED related web pages, which poses as the research gap. The future work is to research and implement this framework.

A graph based approach [35], for classification, that scores over other approaches and that does not depend on labelled data is gaining popularity. The motivation is that it discusses about semi-supervised learning, which has been an important research oriented

topic in recent years, because it is difficult and expensive to obtain labelled data for classification purposes. It assigns weights according to similarity to unlabelled web pages and uses similarity measures to construct a K Nearest Neighbours graph. The problem then simplifies to that of label propagation between graph nodes. The noisy links like advertisements are removed and Transducer SVMs or TSVMs are used to compare the results. The research gap is that there still exists noisy links which reduces the efficiency of this approach. Hence future work is to effectively remove noise from the link information and optimize the calculation algorithm of semi-supervised learning, so as to investigate its new applications in the future.

The use of ontology [36], or a domain based representation of knowledge, in order to provide mechanism to enable machine reasoning has increased during the last few years. The motivation is the use of ontology for document classification which expresses terminology information contained in web documents. This approach seeks to exploit the fact that typically web documents have the following characteristics - First, they are structured in a web site with the web site as a unit, and in other words, a web site normally consists of many web texts with the same or a similar subject. Second, each web document exists as a part of other web sites. Finally, individuals or organizations that have specific purposes manage typical web sites. The advantages of an ontology model has higher accuracy which has been proved later in the paper and it is also mentioned that it needs no training data unlike existing methods. It is compared with an existing Bayesian classifier and it is found that this model outperforms. The research gap is that it is required to develop more efficient and accurate ontological expressions and to document classification methods. Future work would be to conduct further studies on how to improve the efficiency of an information search using the document classification technique suggested in this paper and how to automatically determine the meaning of concepts and relations from Web documents.

In dynamic and hierarchical classification system, the hierarchical structure[37] is exploited making use of categories of web pages. The motivation of this paper is the automatic classification of web pages to overcome the difficulty of retrieving information from the internet. First the web pages are arranged in a format that supports classification. They are then classified into categories using a "bag-of-words" approach. The categories

are then hierarchically arranged to form a tree where common features are propagated from parent to child node and at the same time each node can be reached through a unique path because it has its own set of unique features. It has been tested on two aspects of their system viz unique path traversal and dynamic updation where they find 84% and 78% accuracy respectively. The research gap is that this paper provides just a starting point for classification of web pages from the internet but the internet is highly unstructured and thus future work would be to convert the whole internet to a well-structured system based on some classification standard.

All the efforts above have dealt with isolated instances and examples of algorithms used in specific domains for webpage classification. Many research gaps remain such as integration of multiple approaches, dynamic updation, improvements in algorithms accuracy and precision as well as scalability for larger and more complex datasets. Each of these problems has to be dealt with and researched upon in order to improve this domain and use the results in productive and feasible final applications.

## 1.2 Motivation

All work till date in the field of web page classification has dealt with a single algorithm on a limited dataset and a specifically selected metric for measurement. There has been no attempt to compare multiple algorithms across large complex datasets and use multiple measurement metrics. There is also a need to examine a domain of widely changing web pages such as those of online courses which can change as per date and location. This project is motivated to solve the above issue. There is also no system in place which provides access to information about these courses offline unless it has been manually composed. Such efforts are tedious and prone to error and obsolescence. Therefore it is strongly required to provide multiple facets of information to query over 30,000 courses from over 60 providers worldwide and create an offline repository that can be accessed easily. The application developed is motivated to be dynamic in updation and also provides the first of its kind, an offline automatically sourced and accurate standalone application to provide online course information.

## 1.3 Problem Statement

The problem statement is to compare and analyse the various machine learning algorithms used for binary classification of unstructured data collected from web pages of online course providers across the internet. There are over 35,000 online courses from world's leading universities. According to a recent survey, more than 7 million students are enrolled in them. Manual updation and collection of information about these courses is tedious and static in nature. Automatic analysis of web pages in order to extract information about these courses is the need of the hour. A system is to be developed which can crawl the web, filter unnecessary pages, extract content, gather social tags, pre-process links and metadata, compute features and then classify whether each page visited is a relevant online course or not. After refining the system and evaluating the results of multiple algorithms, the project aims to use the best one to classify a large dataset and store offline information about relevant course pages. This offline information must then be presented to the users with a fresh and intuitive interface and the feature to update dynamically.

## 1.4 Objectives

The main objective of the project is to be able to collect information on web pages relevant to training courses according to various factors such as cost, curriculum, provider, university, instructor, description, webpage link and certification. The first way to achieve this is to fulfill the aim of successfully and quickly crawling multiple domains while filtering out advertisements, spam and unwanted pages. Preprocessing is required in terms of analyzing text contents as well as page metadata enriched features, date and time, social tagging and other parsing and text extraction based features. The next objective is to create an innovative, efficient and fast way to classify courses and training web pages according to user need and relevance. This could be done by evaluating various binary classification algorithms such as decision trees, logistic regression, naïve Bayes network, K nearest neighbors and random forest ensemble. Once done, this must be available in the form of a fresh and interactive offline standalone application which users can query according to multiple parameters. Automatic updating of content from the retrieved web pages according to the relevance for storage and querying for long term use is ensured.

## 1.5 Scope

The scope of the project is that the end users would have an offline system which would allow them to search for the available online courses according to the area of interest. For initial setup as well as subsequent updates over a defined time frame like a month, the system requires accessing of the internet to perform crawling and indexing of web pages. Once this data is collected, efficient classification algorithms will be applied offline to segregate the pages and fill them into a database for the user to access in future.

## 1.6 Methodology

The methodology of the project involves firstly, gathering of data from multiple sources to create a dataset large enough for training a model classifier. The next step is analysis and preprocessing of data collected, selection and creation of features with innovative improvements. After this, training of model and testing using various methods like bootstrapping and cross validation must be done. The penultimate step is comparison using multiple parameters such as F measure and ROC curves. Finally, the development a complete product with the best model and an interactive front end for user and business applications must be carried out.

## 1.7 Organization of the report

This section of report briefly describes the organization of each topic in the report. The report contains 8 chapters altogether. The following paragraphs describe the brief summary of each chapter respectively.

Chapter 2 is Software Requirement Specifications. It gives an overall description of the functionalities available in the system along with the specific requirements to achieve each of these functionalities.

Chapter 3 is High Level Design. It focuses on High level Design of the system. It gives an abstract view of the architecture of the basic building blocks of the system. It explains how the data flows between various layers of system.

Chapter 4 is Detailed Design, which focuses on the structure chart of this project and and flowchart diagrams. It explains the key modules involved in the project.

Chapter 5 is Implementation, which explains the programming language, development environment and code conventions followed during the project. This chapter puts light on the various difficulties encountered in the project and how they were overcome.

Chapter 6 is Testing, which explains the test environment and briefly explains the test cases which were executed during unit testing, functional testing and integration testing.

Chapter 7 is Experimental Analysis and Inference of results where the results of the project are listed and inferences are made about the efficiency of the editor.

Chapter 8 is conclusion, which gives the outcome of the project work carried out and also brings out the limitations of the project and future enhancements.

## **1.8 Summary**

This chapter has provided an introduction to the various domains covered in this project as well as relevant information regarding the same. It has provided an extensive literature survey about the background of the field in which research is conducted, as well as introduced various objectives, aims and overview of the entire project, while setting the tone for the rest of the detailed phases conducted through the project. The remaining chapters of the report deal with further stages in the conceptualization, design and development of the idea explained herewith.

## Chapter 2

### **Software Requirements and Specifications of classification of unstructured data from online course web pages**

A Software Requirements Specification (SRS) is a comprehensive description of the intended purpose and environment for software under development. The SRS fully describes what the software will do and how it will be expected to perform. An SRS minimizes the time and effort required by developers to achieve desired goals and also minimizes the development cost. A good SRS defines how an application will interact with system hardware, other programs and human users in a wide variety of real-world situations. Parameters such as operating speed, response time, availability, portability, maintainability, footprint, security and speed of recovery from adverse events are evaluated [41].

#### **2.1 Overall Description**

In this project, a system is considered which provides the user offline access to the various online courses available. For initial setup as well as subsequent updates over a defined time frame like a month, the system requires to access the Internet to perform crawling and indexing of web pages. Once this data is collected, efficient classification algorithms will be applied offline to segregate the pages and fill them into a database for the user to access in future.

##### **2.1.1 Product Perspective**

This project is aimed at providing the service of web page classification by making use of machine learning algorithms and improving the experience by using a very high quality of pre-processed data used for training. The technique proposed in this project is through improved random forest algorithm and compare it with other classification algorithms which have hitherto not been used with this particular type of data. This novel approach to classification also seeks to combine multiple features such as social tagging in multiple websites as well as spatial and temporal relevance of web pages. Users are provided access only to relevant courses using an interactive, fresh and intuitive front end that enhances user experience and is also portable enough to be used on mobile platforms.

### 2.1.2 Product Functions

The project aims at providing users easy offline access to relevant online courses available. Thus, the product performs the following functions:

- Crawls the web and identifies relevant online course web pages
- Provides the option for a user to find courses by subject, duration and university
- Enables a user to update the application over a long term span so that the information remains relevant
- Uses highly optimized machine learning algorithms to ensure the most accurate results without wastage of computational resources
- Allow code portability across multiple platforms.

### 2.1.3 User Characteristics

A user can be any person who is new to the service. The user might be very much familiar with the available functionalities or a complete novice. Hence the GUI of the application should be built in such a way that the user takes no time to get used to its interface. The target audience is everyone who wishes to obtain information about online courses right from students to universities and corporate organizations wishing to provide training to employees.

### 2.1.4 Constraints

This software has been designed keeping in mind the basic factors of user convenience and improved functionality together. However, there are various aspects that constrain the functioning of the software:

- Crawling the entire web is extremely time consuming and does not justify the cost of resources spent due to relevance of results. Hence the user must note that since there is no manual creation of results, the application can only strive for highest levels of accuracy but cannot guarantee the absence of any discrepancies.
- Slow internet connections or outdated versions of browsers might not be able to handle the website. Since the functionality of the software depends on the functioning of the server, this would be an additional constraint.
- User understanding and familiarity is another constraint upon which the usefulness of the software depends, something that may vary across systems.



## **2.1.5 Assumptions and Dependencies**

For the service to function as expected, certain prerequisite conditions must exist on which the application can run. The presence of these conditions results in the most optimal performance. Thus, the following subsection discusses few assumptions and dependencies that help the application to run effectively.

The web service is assumed to be updated at frequent intervals such that all the information provided is most recent. This will enable the database to be more reliable and useful. Thus, a dependency exists on the regular updates of records for proper user information and proper automation.

## **2.2 Specific Requirements**

This section contains the various requirements specific to each parameter of the entire project. The following section briefly describes the different project requirements essential for its proper working.

### **2.2.1 Functionality Requirements**

The functional requirements for the product are as follows:

- Enable users to initially set up the application easily.
- Enable a user to easily access instructions related to the application.
- Enable a user to specify the parameters for selection of new courses.
- Enable the system to periodically update by re-crawling and classification of data obtained thereof.

### **2.2.2 Performance Requirements**

The performance of the project mainly depends on the available internet bandwidth which goes hand-in-hand with the computer hardware requirements. The various performance requirements are detailed further in this chapter with respect to both the developers' and users' point of view.

### **2.2.3 Supportability**

The web service will be supported on various operating systems as it is a web based system. It only requires a web browser to access its contents since web browsers provide UI to modify the database and use its functionalities. Therefore, supportability or maintainability of the system being built, including coding standards, naming conventions, class libraries, maintenance access and maintenance utilities can be enhanced by implementing it like the real world application projects.

### **2.2.4 Software requirements**

Software requirements deal with defining software resource requirements and prerequisites that need to be installed on a computer to provide optimal functioning of an application. These requirements or prerequisites are generally not included in the software installation package and need to be installed separately before the software is installed.

- Java1.4 or higher
- Java Swing-front end
- JDBC-Database connectivity
- UDP-User Datagram Protocol
- TCP-Transmission Control Protocol
- Networking-Socket programming
- MySQL server on Windows/ Linux/ Mac OS X
- Windows 98 or higher-Operating System

### **2.2.5 Hardware requirements**

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware, A hardware requirements list is often accompanied by a Hardware Compatibility List (HCL), especially in case of operating systems. An HCL lists tested, compatible, and sometimes incompatible hardware devices for a particular operating system or application. The following sub-sections discuss the various aspects of hardware requirements.

All computer operating systems are designed for a particular computer architecture. Most software applications are limited to particular operating systems running on particular architectures. Although architecture-independent operating systems and applications exist, most need to be recompiled to run on a new architecture.

The power of the Central Processing Unit (CPU) is a fundamental system requirement for any software. Most software running on x86 architecture define processing power as the model and the clock speed of the CPU. Many other features of a CPU that influence its speed and power, like bus speed, cache, and MIPS (Million Instructions Per Second) are often ignored. This definition of power is often erroneous, as AMD Athlon and Intel Pentium CPUs at similar clock speed often have different throughput speeds. Hence the following are a recommended minimum hardware configuration:

- 10GB HDD
- 0.125 GB RAM
- Pentium P4 Processor 3.3GHz

### **2.2.6 Design Constraints**

Due to the evolving technologies being used and with the growing demand for better technologies the conventions being used in programming languages is also changing. To support better GUI and User experience the designers have favoured the use of Java Swing front end which also provides good database connectivity. Hence, even though Java is becoming a ubiquitous part of almost all machines today, this design constraint mandates its presence for correct and complete functioning of the application.

### **2.2.7 Interfaces**

The interface must be as user friendly as possible. Even a new user should easily understand the complete functionalities provided by the social networking service. This can only be achieved by providing a user-friendly GUI. The user will be provided with a HELP page with information on how to navigate the functions of the product. A possible improvement in subsequent versions would be the addition of a help video or a similar tutorial.

## 2.3 Other Non-functional Requirements

The other non-functional requirements include performance requirements that focus to accommodate user requirements in terms of usability. It also includes safety and security requirements along with software quality attributes.

### 2.3.1 Performance Requirements

There are a basic set of requirements that state the minimum performance expected from the system from the user's point of view to ensure a smooth and glitch-free operation of the system. This set of requirements is listed below:

- No user shall experience difficulty in understanding the rules and functioning of the application
- No user shall experience any difficulty in navigating through the various features of the application
- No user shall find it cumbersome to sort and use different search parameters for online courses
- No user will be subjected to a lot of menus to access all the relevant information.

### 2.3.2 Safety and Security Requirements

Safety and security of the system are crucial attributes that seek to preserve the integrity of the system. The system must be protected against all failures and attempts to unauthorized attack and access of information. The following requirements detail the same:

- Since the system seeks to function offline, it is recommended that client machines are installed with strong firewalls and antivirus software to ensure that the software can function smoothly. In addition to this, there should be adequate privacy and firewall authentication each time the application attempts to connect to the internet and update itself, but must request user permission for the same at all times.
- The database will be stored on the client machine hence reducing the possibility of server related downloading attacks

- To ensure higher security standards, it is recommended that the web interface is accessed using secure protocols such as the https protocol for secure data transmission between the client and the server machines.

### 2.3.3 Software Quality Attributes

The additional requirements detailed below are related to the quality of the software and list out the desired characteristic attributes for the software to have. These are not technically specified but encouraged to be adhered to in development. They are:

- Adaptability** - Since the software is dynamic in nature, provision must be made to include newer courses based on the need of the hour. Since certain courses are location-based or occupation based, the software must be able to adapt to these with minor modifications
- Availability** - The software must be running and able to deliver required services to the user at any given point in time.
- Correctness** - Since the information generated as output of the classification algorithm is meant to eliminate manual labour, the application must strive to be of highest level of accuracy when classifying web pages as relevant to user needs.
- Flexibility** - The types of users will vary in levels of experience, expertise, familiarity with the interface and multiple other factors. The product must be flexible enough to cater to this diversity and the user needs as well.
- Interoperability** - The product must be universally available on all platforms and operating systems.
- Maintainability** - The product must require minimal maintenance which should be easy to accomplish by means of simple code and detailed documentation.
- Portability** - The services offered by the product must be independent of location, platform and other location or system based parameters.
- Reliability** - The risk, likelihood and severity of potential failure of the product must be minimized. If failure occurs, the product should show adequate recovery from the failure state.
- Reusability** - The product must be developed using methods in accordance with software reuse like improving on a pre-existing version without having to start from scratch.

- j) **Robustness** - The product must be able to function under different sets of constraints like high server load, incorrect input data and other potentially dangerous scenarios, with minimal chance of failure.
- k) **Testability** - The product must be such that it's testing is simple, orderly and easy to document and improve upon. Unit testing, component testing, sub-system and system integration testing, as well as end-user and release testing must be done and the reports of the same must be documented.
- l) **Usability** - The product must be easy to use and must not cause the end user any trouble in navigational or understanding its various functionalities and interfaces.

## 2.4 Summary

The second chapter of the report has detailed the software requirements specification of the project in terms of hardware, software, functional, non functional, security, quality and other desirable attributes of software. The entire product has been developed according to these specifications which keep the user need and developer convenience and quality uppermost at all times.

## Chapter 3

### High Level Design of classification of unstructured data from online course web pages

Software sometimes can be a complex entity. Its development usually follows what is known as Software Development Life Cycle (SDLC). The second stage in the SDLC is the Design Stage. The objective of the design stage is to produce the overall design of the software. The design stage involves two sub-stages namely:

1. High-Level Design
2. Detailed Design

In the High-Level Design, the Technical Architect of the project will study the proposed applications functional and non-functional (qualitative) requirements and design overall solution architecture of the application, which can handle those needs. High Level Design precisely discusses an overall view of how the system should work and the top-level components that will form the system that work to achieve the proposed solution. It should have very little detail on implementation, i.e. no explicit class definitions, and in some cases not even details such as database type (relational or object) and programming language and platform [44].

The high level design consists of the detail structuring of the preprocessing phase. It consists of a crawler that finds and retrieves web pages, a trainer that analyses a list of relevant (training pages) and irrelevant links and compute probabilities about the feature-category pairs found and an indexer which is responsible for identifying and extracting all suitable features from each web page.

### 3.1 Design Considerations

This section addresses the issues that need to be discussed or resolved before attempting to devise a complete design solution.

#### 3.1.1 General Constraints

The initial stage of the project i.e. is the pre-processing stage involves crawling which can be time consuming in order to crawl for relevant online course web pages.

The application can only strive for highest levels of accuracy however cannot guarantee the absence of variations. This software has been designed in accordance to the basic factors of user convenience and improved functionality together. The constraints are:

- Slow internet connections or outdated versions of browsers due to which the software may not deliver effectively.
- The functioning of the server, upon on which the functionality of the software depends.
- User understanding and familiarity upon which the usefulness of the software depends, which varies across systems.

### **3.1.2 Development Methods**

The development method used in this software design is modular/functional development method [44].The approach is categorized into various phases. The input-output data that flows from one module to another shows the dependency. The phases are: crawling, extraction (feature set), application of classification algorithms (including the modified algorithm), comparison of results, design of front end. Data flow diagrams have been used in the modular design of the system.

## **3.2 Architectural Strategies**

This section describes the design decisions and strategies that affect the overall organization of the system and its higher-level structures. These strategies will provide insight into the key abstractions and mechanisms used in the system architecture.

### **3.2.1 Programming Language**

The programming language plays a major role in the efficiency as well as the future development of the project. This project is developed in Java. Since Java is system independent and portable, it is the most suitable language to be used for this project. The front end is built using java swing. Swing is the primary Java GUI widget toolkit, an API for providing a Graphical User Interface (GUI) for Java programs. Swing is a platform-independent, Model-View-Controller GUI framework for Java, which follows a single-threaded programming model. Additionally, this framework provides a layer of



abstraction between the code structure and graphic presentation of a Swing-based GUI. It also provides good database connectivity.

### **3.2.2 Future Plans**

Upon the completion of the project, the following are to be implemented

- To build a crawler for AJAX based web pages
- To build a mobile application
- Improve data set as per increasing number of online course web pages.
- Research into more efficient classification algorithms.

### **3.2.3 Error Detection and Recovery**

The software should be built in such a way that it should avoid errors and prevent failures from occurring. The correct set of rules for classification are used to ensure appropriate set of features(existent in all web pages) are extracted for classification in the classifier stage. This ensures and improves the quality of the data and renders good results for the machine learning algorithms used. The training of the model and testing needs have been performed appropriately using bootstrapping and cross validation methods. Necessary precautions are taken to ensure faulty data is avoided in the database such that errors can be prevented. Care should be taken to ensure that the database is valid and up to date.

### **3.2.4 Data Storage Management**

The data is stored in CSV files. The data retrieval is done with the help of CsvJdbc driver. CsvJdbc is a read-only JDBC driver that uses Comma Separated Value (CSV) files or DBF files as database tables. It is ideal for writing data import programs or analyzing log files. The driver enables the access of a directory or a ZIP file containing CSV or DBF files as if it were a database containing tables. CsvJdbc accepts only SQL SELECT queries from a single table

### **3.2.5 Communication Mechanism**

The project involves communication between the user and the system through an interface. The interface is responsible for querying the repository for required online

courses from the user end. There is also a communication established between the repository and the crawler and repository and the classifier.

### 3.3 System Architecture

The software has the following major phases which guide the working of the whole system:

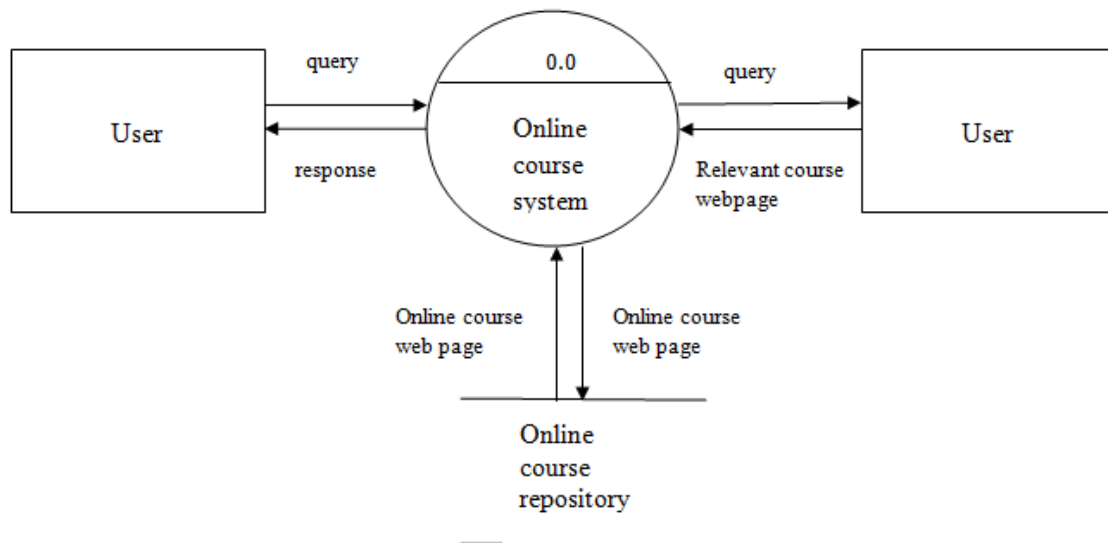
- Crawler(fast multi-threaded java crawler)
- Extraction of features.
- Classification of web pages using various machine learning algorithms.
- Comparison of the machine learning algorithm results.
- Designing the front end.

### 3.4 Data Flow Diagrams

A Data Flow Diagram (DFD) is a graphical representation of the "flow" of data through an information system. Data Flow models are used to show how data flows through a sequence of processing steps. The data is transformed at each step before moving on to the next stage. During the live migration the data flow occurs between the two physical machines. These processing steps or transformations are program functions when data flow diagrams are used to document a software design. The DFD for the performance analysis of live migration of the virtual machines can be decomposed into two levels such as level-0 and level-1 [44].

#### 3.4.1 Data Flow Diagram – Level 0

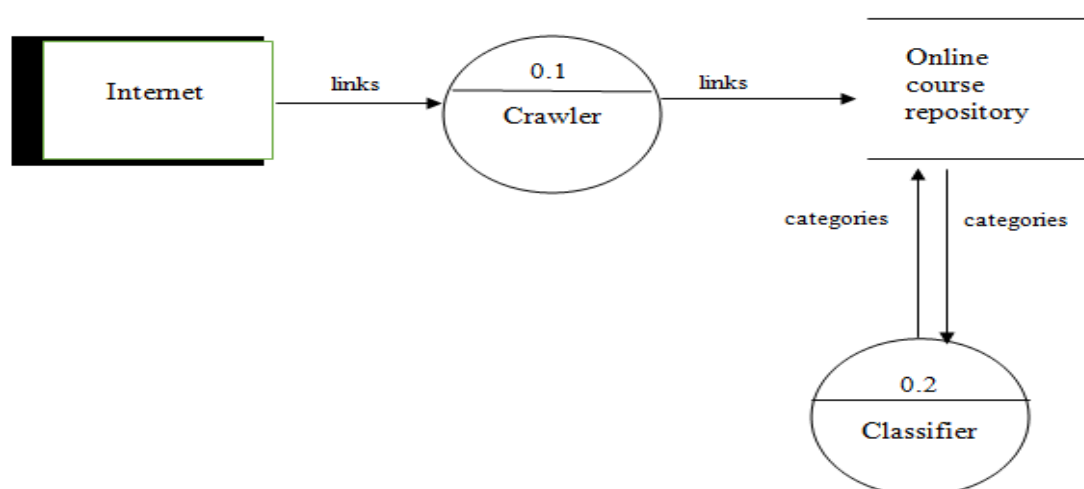
Level 0 is the initial level Data Flow Diagram and it is generally called as the Context Level Diagram (CLD). It is a common practice for a designer to draw a context-level DFD first which shows the interaction between the system and outside entities. This context level DFD is then "exploded" to show more detail of the system being modelled. Figure 3.1 below shows the Level 0 Data Flow Diagram. It shows the interaction between the user the online course repository. The client passes query requests in the front end. The results for the requests are searched in the repository and results are displayed back on the front end.



**Figure 3.1: Data Flow Diagram level 0 for online course web page classification system**

### 3.4.2 Data Flow Diagram – Level 1

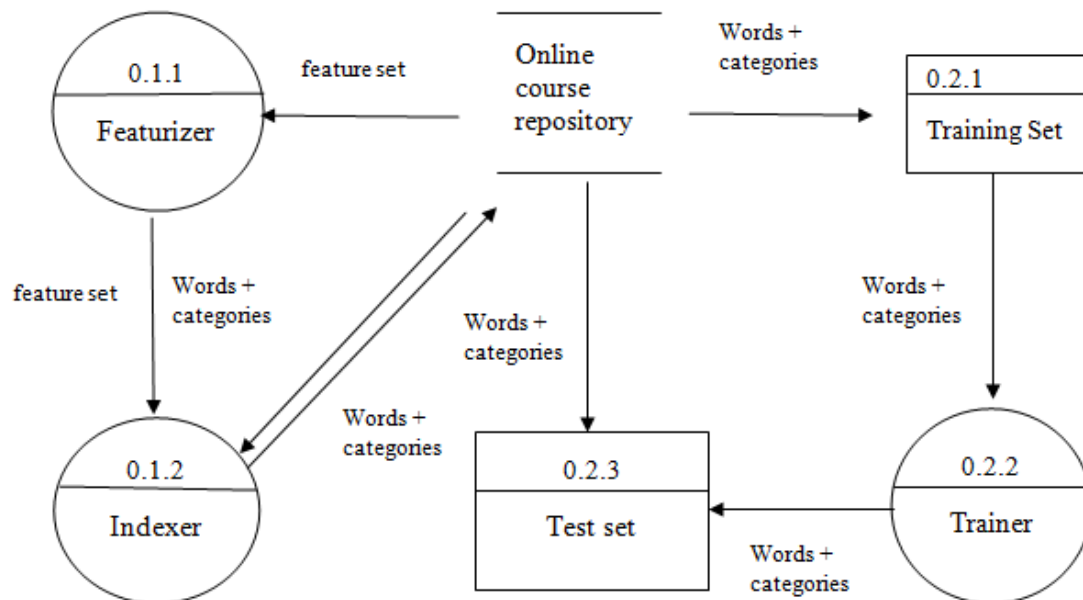
The Level 1 Data Flow Diagram gives more information than the level 0 Data Flow Diagram. The Figure 3.2 below shows the Level 1 Data Flow Diagram. It shows the back end mechanism. This mechanism does the job of retrieving the data and having the data to be classified. It involves a crawler which essentially crawls the urls of online course web pages. The fetched data is stored in a repository in a structured format upon which the classifier does the job of classifying the web pages as online course web pages.



**Figure 3.2: Data Flow Diagram level 1 for online course webpage classification system**

### 3.4.3 Data Flow Diagram – Level 2

The Figure 3.3 below shows Level 2 data flow diagram. It represents more detailed flow of data between the processes. It elaborates the procedure of fetching the data which is in the unstructured format. After the crawler crawls the urls, the featurizer extracts the feature set from the web pages and consolidates the data in the structured format. This structured data is then put in the repository for the classifier to work upon. The classifier is trained with a trained data set and then used to classify the test data set.



**Figure 3.3: Data Flow Diagram level 2 for online course webpage classification system**

## 3.6 Summary

This chapter details the phases required by architects to understand the overall flow and functioning of the system as a whole, as well as a collection of individual modules which have been elaborated with various features and descriptions of data flow within the system. The chapter has also detailed various phases of the development and

working of the project right from preprocessing and crawling to classification and the user interaction. This flow of data, architecture and other design related constraints and specifications constitute the high level design of the project in completion.

## Chapter 4

### Detailed Design of classification of unstructured data from online course web pages

Detailed Design is a phase where in the internal logic of each of the modules specified in high-level design is specified. In this phase further details and algorithmic design of each of the modules is specified. Other low-level components and sub-components are also described as well. Each subsection of this section will refer to or contain a detailed description of system software component. This chapter also discusses about the control flow in the software with much more details about software modules by clarifying the details about each function with functionality, purpose, input, and output. This chapter presents the following:

- Structure Chart for the project
- Functional Description of units
- Flowchart for units

#### 4.1 Structure Chart

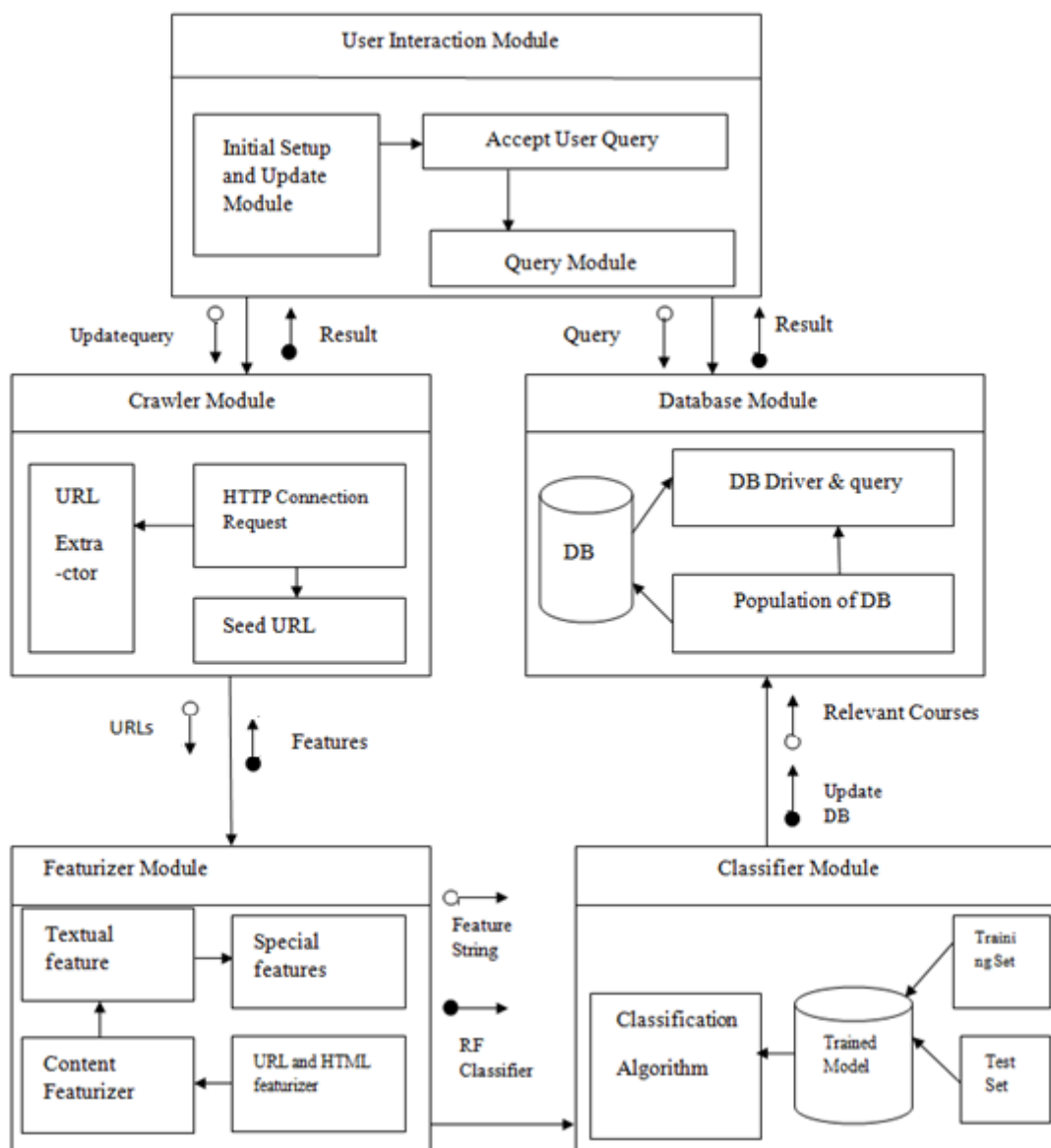
A Structure Chart shows the control flow among the units in a system. The Structure Chart explains the identified units and the interaction between the units. It also explains the sub-modules. It explains the input for each unit and output generated.

There are 5 main modules to this project:

1. Crawler
2. Featurizer
3. Classifier
4. Database
5. User Interface

The sequence of flow and control of each module/sub-module is shown in figure 4.1.

This is a structured chart that represents all the modules and the data that flows in them.



**Figure 4.1: Structure chart of classification of unstructured data from online course web pages**

## 4.2 Functional Description of Modules

This section contains a detailed description of software components, low-level components and other sub components of the project.

### 4.2.1 Crawler module

This section contains some information on the functionality and some software component attributes of the Crawler Module. Figure 4.2 is the flow chart for the crawler module. The crawler is first initialized with seed urls. After the initialization http request

is sent to download the page. The feature module is then initialized if the the web page is of online course.

- **Purpose:** The purpose of this module is to crawl the online courses related web urls.
- **Functionality:** The functionality of this module is to use a multi-threaded crawler to crawl the web for urls upto the depth of 5 to ensure all the related web pages are covered.
- **Input:** Intialize the crawler with seed urls.
- **Output:** The output is the list of urls. These urls are fed into the featurizer as an input.
- **Flowchart:** The flowchart for the given module is given as follows in figure 4.2. This is the flowchart for the Crawler module. This flowchart indicates how the crawler performs the function of scanning the web and fetching pages. This is done within a particular domain beginning from an initial seed variable specified and up to the configured depth of crawling.

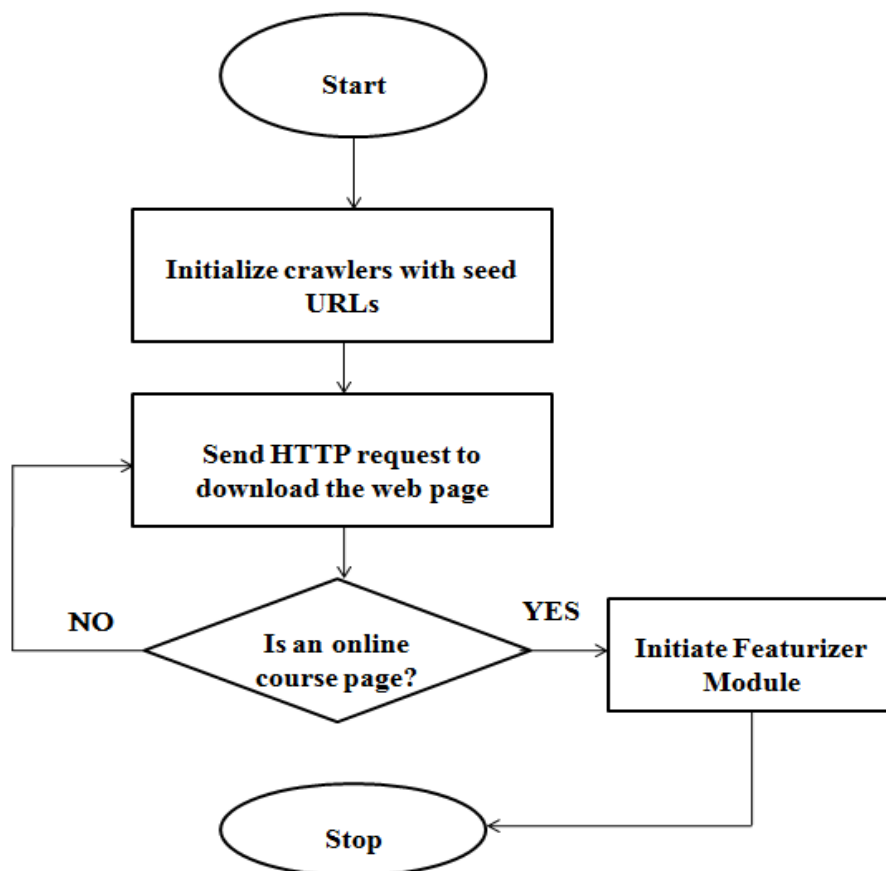


Figure 4.2: Flow chart for crawler



### 4.2.2 Featurizer module

This section contains some information on the functionality and some software component attributes of the Featurizer Module. Figure 4.3 is the flow chart for the featurizer module.

- **Purpose:** The purpose of this module is to extract features from the web pages crawled urls of the online course webpages.
- **Functionality:** The functionality of this module is to extract features such as the title of the webpage, meta-description, course name, keywords, social tags, provider, price, duration, certificate, instructor, description and level.
- **Input:** The urls crawled by the crawler.
- **Output:** The output will be the feature string that essentially contains url features,content features,social features,HTML features. This module structures the data from the web pages.
- **Flowchart:** The flowchart for the given module is given as follows in figure 4.3.

First step of the module is to obtain the list of urls crawled. The features are then extracted such as urls features, content features, social features, HTML features. The set of features extracted are title of the web page, meta-description, course name, keywords, social tags, provider, price, duration, certificate, instructor, description and level. These features are extracted based on the type of domain as well as the specific pages within the domain. Various scripting language and regular expression based methods are used to extract the required features and they are concatenated into a single comma separated string so that they can directly be given to the classifier and further modules in the system.

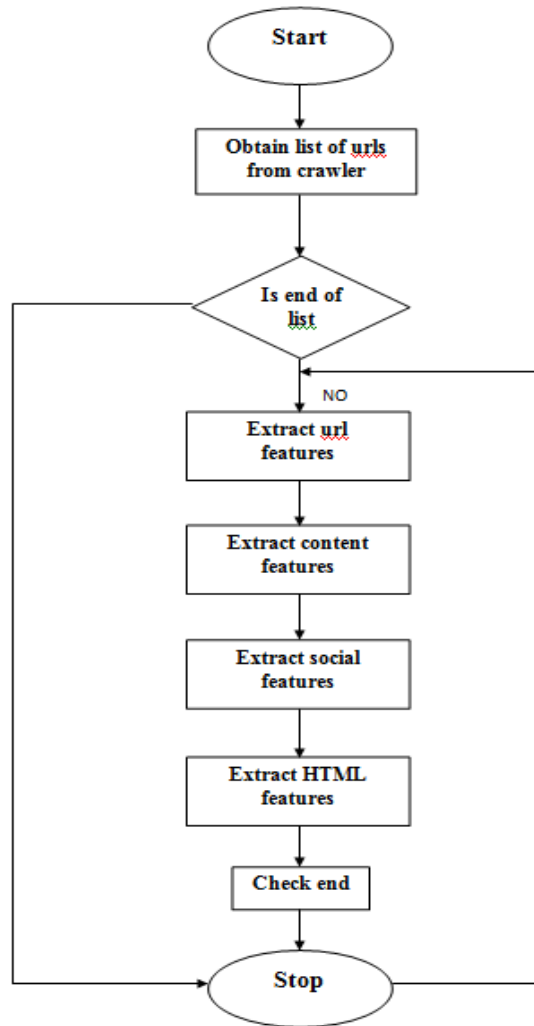


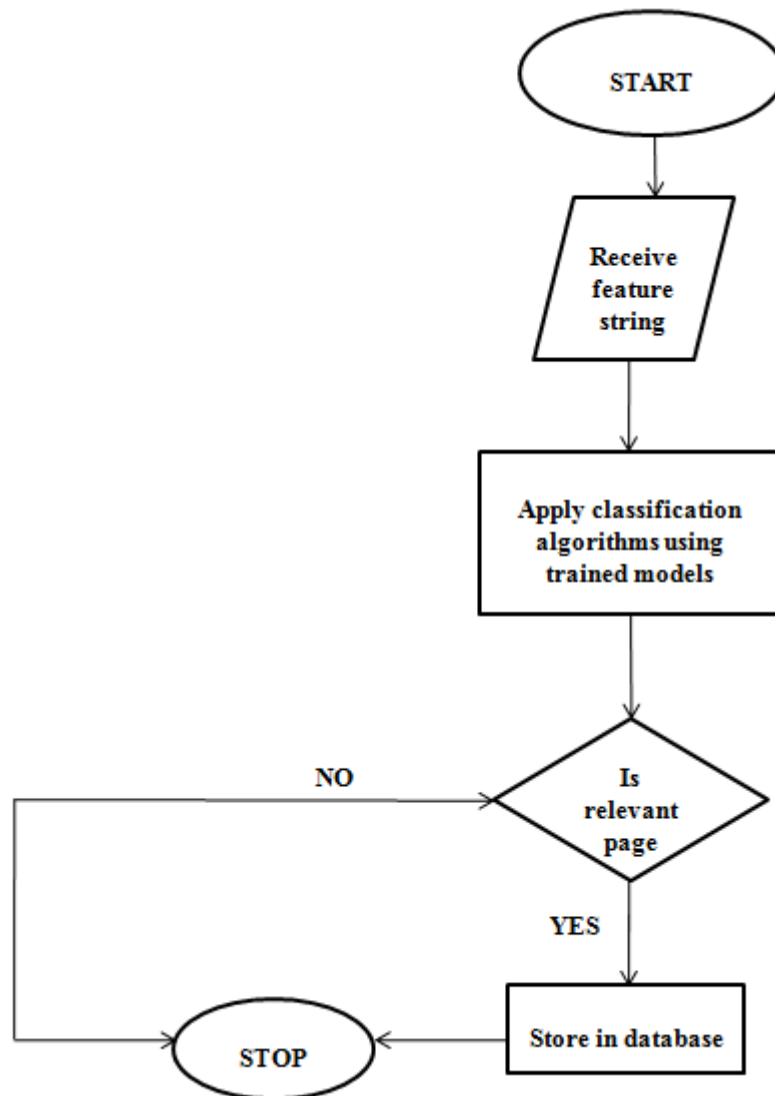
Figure 4.3: Flow chart for featurizer

### 4.2.3 Classifier module

This section contains some information on the functionality and some software component attributes of the classifier Module

- **Purpose:** The purpose of this module is to classify the featurized web pages as relevant or irrelevant pages.
- **Functionality:** A model is trained initially with a trained dataset. It is bootstrapped with a trained set. The trained model is then applied on the the test dataset to classify the rest of the web pages. Classification algorithms are applied using the trained model for classification.
- **Input:** The feature string which was the output of the featurizer module.

- **Output:** Classified data indicating the relevance for each of the web page that was crawled. This data is then stored in the database.
- **Flowchart:** The flowchart for the given module is given as follows in figure 4.4.



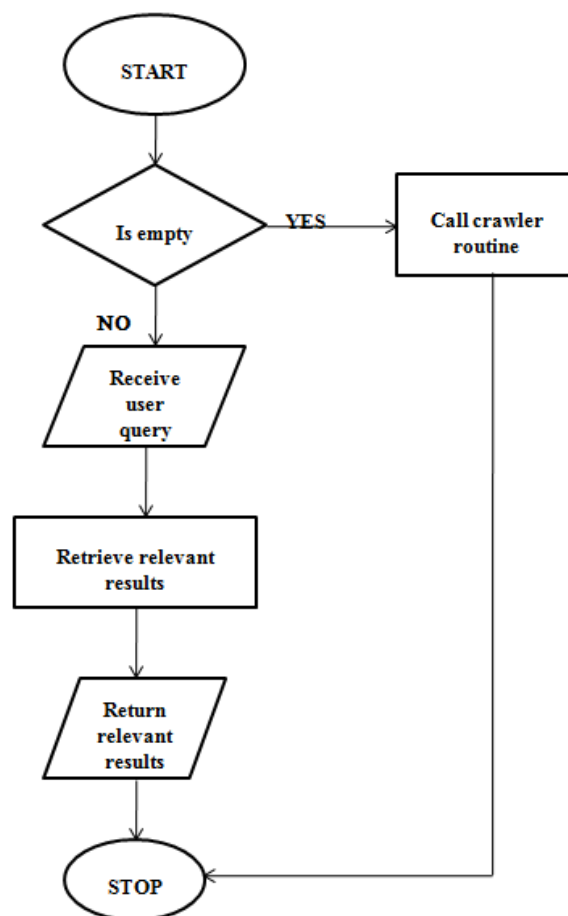
**Figure 4.4: Flow chart for classifier**

The first step of the module is to receive feature string obtained from the featurizer. Classification algorithms are then applied upon the received feature string using trained models. The classifier then classifies the web page into relevant online course web page. The relevant online course pages are then stored in the database.

#### 4.2.4 Database module

This section contains some information on the functionality and some software component attributes of the database Module

- **Purpose:** The purpose of this module is store the classified structured web page data.
- **Functionality:** The functionality of this module is to retrieve relevant results for user query and return the results to the user.
- **Input:** The user query. The user query is made in the form of an SQL statement by the module.
- **Output:** The output will be the results for the query made.
- **Flowchart:** The flowchart for the given module is given as follows in figure 4.5.



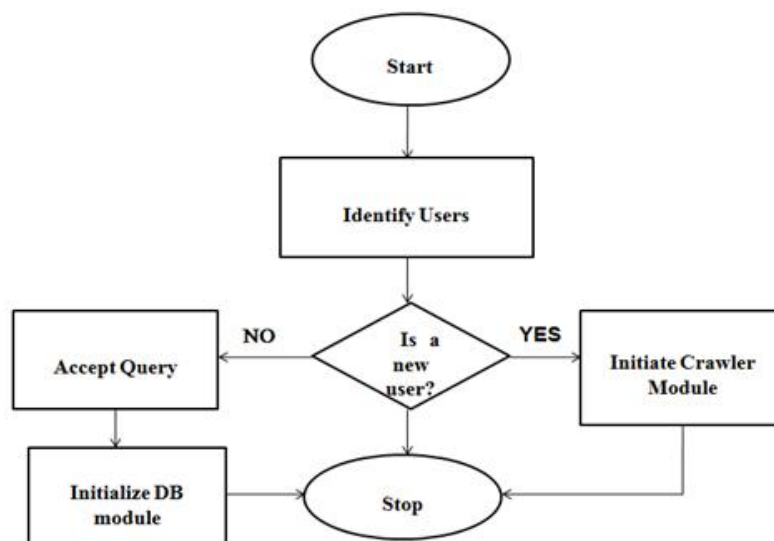
**Figure 4.5: Flow chart for database module**

The first step begins with received user query. Relevant results i.e the relevant online course web pages are returned that satisfies the user query. If the database is empty crawler routine is called.

### 4.2.5 User Interface module

This section contains some information on the functionality and some software component attributes of the user interface module

- **Purpose:** The purpose of this module is to design a user friendly front end.
- **Functionality:** The functionality of this module to accept the query requests from the user and retrieve the results from the database. The main functionality is to display the information in a consolidated and a structured manner. It also contains an update module. The update module delivers the data about the new courses introduced in the recent past. It calls the crawler routine to crawl the web for new course and delivers an upto date information to the users.
- **Input:** The user query.
- **Output:** Results for the query made by the user.
- **Flowchart:** The flowchart for the given module is given as follows in figure 4.6.



**Figure 4.6: Flow chart for user interface module**

## 4.3 Algorithms

Five classification algorithms were chosen for comparing the accuracy in labelling the online course web pages as relevant courses. The following are the algorithms compared which are discussed in detail.

1. C4.5
2. Logistic Regression
3. Naive bayes
4. K Nearest Neighbor
5. Random Forest

### 4.3.1 C4.5 Algorithm

C4.5 is an algorithm used to generate a decision tree. C4.5 builds decision trees from a set of training data. The training data is a set  $S = s_1, s_2, s_3, \dots$  of already classified samples. Each sample  $S_i$  consists of a p-dimensional vector  $(x_{1,i}, x_{2,i}, \dots, x_{p,i})$ , where the  $x_j$  represent attributes or features of the sample, as well as the class in which  $S_i$  falls. At each node of the tree, C4.5 chooses the attribute of the data that most effectively splits its set of samples into subsets enriched in one class or the other. The splitting criterion is the normalized information gain (difference in entropy). The attribute with the highest normalized information gain is chosen to make the decision. The C4.5 algorithm then recurses on the smaller sublists [44].

This algorithm has a few base cases.

- All the samples in the list belong to the same class. When this happens, it simply creates a leaf node for the decision tree saying to choose that class.
- None of the features provide any information gain. In this case, C4.5 creates a decision node higher up the tree using the expected value of the class.
- Instance of previously-unseen class encountered. Again, C4.5 creates a decision node higher up the tree using the expected value.

The general algorithm for building decision trees is:

1. Check for base cases for each attribute a Find the normalized information gain ratio from splitting on  $a$

2. Let  $a\_best$  be the attribute with the highest normalized information gain
3. Create a decision *node* that splits on  $a\_best$
4. Recurse on the sublists obtained by splitting on  $a\_best$ , and add those nodes as children of *node*.

### 4.3.2 Logistic Regression algorithm

Logistic regression or logit regression is a type of probabilistic statistical classification model. It is also used to predict a binary response from a binary predictor, used for predicting the outcome of a categorical dependent variable (i.e., a class label) based on one or more predictor variables (features). That is, it is used in estimating empirical values of the parameters in a qualitative response model. The probabilities describing the possible outcomes of a single trial are modeled, as a function of the explanatory (predictor) variables, using a logistic function. Frequently (and subsequently in this article) "logistic regression" is used to refer specifically to the problem in which the dependent variable is binary—that is, the number of available categories is two—and problems with more than two categories are referred to as multinomial logistic regression or, if the multiple categories are ordered, as ordered logistic regression.

Logistic regression measures the relationship between a categorical dependent variable and one or more independent variables, which are usually (but not necessarily) continuous, by using probability scores as the predicted values of the dependent variable. As such it treats the same set of problems as does probit regression using similar techniques [45].

An explanation of logistic regression begins with an explanation of the logistic function, which always takes on values between zero and one as shown in equation (1):

$$F(t) = \frac{e^t}{e^t + 1} = \frac{e^t}{e^{-t} + 1} \quad (1)$$

viewing  $t$  as a linear function of an explanatory variable  $x$  (or of a linear combination of explanatory variables), the logistic function can be written as in equation (2) given below:

$$F(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}} \quad (2)$$

This will be interpreted as the probability of the dependent variable equalling a "success" or "case" rather than a failure or non-case. We also define the inverse of the logistic function, the logit as in equation (3):

$$g(x) = \ln \frac{F(x)}{1-F(x)} = \beta_0 + \beta_1 x \quad (3)$$

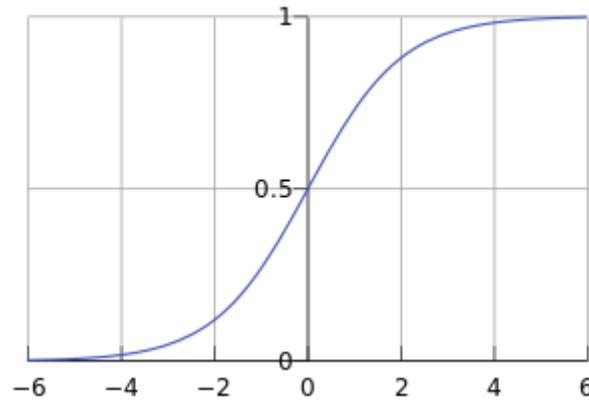
and equivalently in equation 4:

$$\frac{F(x)}{1-F(x)} = e^{\beta_0 + \beta_1 x} \quad (4)$$

A graph of the logistic function  $F(x)$  is shown in Figure 4.6. The input is the value of  $\beta_0 + \beta_1 x$  and the output is  $F(x)$ . The logistic function is useful because it can take an input with any value from negative infinity to positive infinity, whereas the output  $F(x)$  is confined to values between 0 and 1 and hence is interpretable as a probability. In the above equations,  $g(x)$  refers to the logit function of some given linear combination  $x$  of the predictors,  $\ln$  denotes the natural logarithm,  $F(x)$  is the probability that the dependent variable equals a case,  $\beta_0$  is the intercept from the linear regression equation (the value of the criterion when the predictor is equal to zero),  $\beta_1 x$  is the regression coefficient multiplied by some value of the predictor, and base  $e$  denotes the exponential function.

The formula for  $F(x)$  illustrates that the probability of the dependent variable equaling a case is equal to the value of the logistic function of the linear regression expression. This is important in that it shows that the value of the linear regression expression can vary from negative to positive infinity and yet, after transformation, the resulting expression for the probability  $F(x)$  ranges between 0 and 1. The equation for  $g(x)$  illustrates that the logit (i.e., log-odds or natural logarithm of the odds) is equivalent to the linear regression expression. Likewise, the next equation illustrates that the odds of the dependent variable equaling a case is equivalent to the exponential function of the linear regression expression. This illustrates how the logit serves as a link function between the probability and the linear regression expression. Given that the logit ranges between minus infinity and infinity, it provides an adequate criterion upon which to conduct linear regression and the logit is easily converted back into the odds.





**Figure 4.6.** The logistic function, with  $\beta_0 + \beta_1 x$  horizontal axis and  $F(x)$  on the vertical axis

### 4.3.3 Naive Bayes algorithm

A naive Bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumptions. A more descriptive term for the underlying probability model would be "independent feature model" [46].

The Bayes Naive classifier selects the most likely classification  $V_{nb}$  given the attribute values  $a_1, a_2, a_3, \dots, a_n$ .

This results in:

$$V_{nb} = \operatorname{argmax}_{v_j \in V} P(v_j) \prod P(a_i | v_j) \quad (5)$$

Where  $P(a_i | v_j)$  is estimated by:

$$P(a_i | v_j) = \frac{nc + mp}{n + m} \quad (6)$$

where:

$n$  = the number of training examples for which  $v = v_j$

$n_c$  = number of examples for which  $v = v_j$  and  $a = a_i$

$p$  = a priori estimate for  $P(a_i | v_j)$

$m$  = the equivalent sample size

### 4.3.4 K Nearest Neighbor algorithm

k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. The k-NN algorithm is among the simplest of all machine learning algorithms.

KNN is a method for classifying objects based on closest training examples in the feature space. An object is classified by a majority vote of its neighbors. K is always a positive integer. The neighbors are taken from a set of objects for which the correct classification is known. It is usual to use the Euclidean distance, though other distance measures such as the Manhattan distance could in principle be used instead [47]. The algorithm on how to compute the K-nearest neighbors is as follows:

1. Determine the parameter K = number of nearest neighbors before hand. This value is all up to you.
2. Calculate the distance between the query-instance and all the training samples. You can use any distance algorithm.
3. Sort the distances for all the training samples and determine the nearest neighbor based on the K-th minimum distance.
4. Since this is supervised learning, get all the Categories of your training data for the sorted value which fall under K.
5. Use the majority of nearest neighbors as the prediction value.

### 4.3.5 Random Forest algorithm

Random forests are an ensemble learning method for classification (and regression) that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes output by individual trees.

The training algorithm for random forests applies the general technique of bootstrap aggregating, or bagging, to tree learners. Given a training set  $X = 1, \dots, x_n$  with responses  $Y = y_1$  through  $y_n$ , bagging repeatedly selects a bootstrap sample of the training set and fits trees to these samples:

For  $b = 1$  through  $B$ :

1. Sample, with replacement,  $n$  training examples from  $X, Y$ ; call these  $X_b, Y_b$ .
2. Train a decision or regression tree  $f_b$  on  $X_b, Y_b$ .

After training, predictions for unseen samples  $x'$  can be made by averaging the predictions from all the individual regression trees on  $x'$  as shown in equation (7):

$$\hat{f} = \frac{1}{B} \sum_{b=1}^B \hat{f}_b(x') \quad (7)$$

or by taking the majority vote in the case of decision trees.

In the above algorithm,  $B$  is a free parameter. Typically, a few hundred to several thousand trees are used, depending on the size and nature of the training set. Increasing the number of trees tends to decrease the variance of the model, without increasing the bias. As a result, the training and test error tend to level off after some number of trees has been fit. An optimal number of trees  $B$  can be found using cross-validation, or by observing the out-of-bag error: the mean prediction error on each training sample  $x_i$ , using only the trees that did not have  $x_i$  in their bootstrap sample [48].

## 4.4 Summary

This chapter gives the detail layout of the architecture. It discusses all the modules in detail and explains its working. The detail explanation includes the purpose, functionality, input, output and flow chart of each module. Also all the algorithms used for comparison are explained in this chapter. The comparison and results of the algorithms are discussed in later chapters.

## Chapter 5

### Implementation of classification of unstructured data from online course web pages

Implementation of any software is always preceded by important decisions regarding selection of the platform, the language used, etc. these decisions are often influenced by several factors such as the real environment in which the system works the speed that is required, the security concerns, other implementation specific details etc.

#### 5.1 Programming Language Selection

The programming language plays a major role in the efficiency as well as the future development of the project. As such, Java has been chosen as the programming language to be used.

#### 5.2 Platform Selection

The platform selected for the project is Windows 7, as it is a compatible OS for running Eclipse, NetBeans, Weka and other tools. Eclipse is selected for performing web crawling and machine learning related code with high level of dependency on java libraries. NetBeans is used to develop a fresh and intuitive user interface which can process and present the data generated after running the Eclipse code. Weka is used to visualize the graphical evaluation and tabulated results of the various machine learning algorithms.

#### 5.3 Code Conventions

Coding conventions are a set of guidelines for a specific programming language that recommend programming style, practices and methods for each aspect of a piece program written in this language. These conventions usually cover file organization, indentation, comments, declarations, statements, white space, naming conventions, programming practices, programming principles, programming rules of thumb, architectural best practices, etc. These are guidelines for software structural quality. Software programmers are highly recommended to follow these guidelines to help improve the readability of their source code and make software maintenance easier. Coding conventions are only applicable to the human maintainers and peer reviewers of a software project. Conventions may be formalized in a documented set of rules that an

entire team or company follows, or may be as informal as the habitual coding practices of an individual. Coding conventions are not enforced by compilers.

As a result, not following some or all of the rules has no impact on the executable programs created from the source code [52].

### **5.3.1 Naming Conventions**

All class, function and variable names have been chosen keeping in mind that they must reflect the purpose of that class/function/variable. These have been followed with the exception of local counters which are used and discarded almost immediately. For example, the counter of a FOR-loop [53].

### **5.3.2 File Organization**

Efficient file organization for the Classification System may be done as follows:

- Use a separate folder for each project
- Write header comments

The above points are further explained as follows:

#### **Managing Folders**

The Current Folder Browser provides a few features to make managing separate folders easier. The tree views multiple projects from a root directory. Having a top-down hierarchical view makes it easy to move files between project directories. The address bar is used for quickly switching back and forth between project directories. This allows keeping only one of these folders on the Classifier search path at the same time. If there is a nested folder structure of useful functions that needs to be accessed (for example a hierarchical tools directory), “Add with Subfolders” from the context menu can be used to quickly add a whole directory tree to the Classifier search path.

#### **Write Header Comments**

Having comment lines in files enables the functions, scripts, and classes to participate in functions like MainControlCenter and CrawlerController. When a directory is supplied to the help function it reads out a list of functions in that directory. [52]

### 5.3.3 Properties Declarations

The Classification System makes use of a limited number of operators and properties. The maximum scope of the allowed properties are similar to those found in Delicious REST connection API which requires an account to assign social tags to crawled web pages while featurizing them.

### 5.3.4 Class Declarations

The Classification System is completely object-oriented. Thus every block of code resides in a class. Standard Class Naming conventions are used. The Classification System classes use the following words to describe different parts of a class definition and related concepts.

- Class definition - Description of what is common to every instance of a class.
- Properties - Data storage for class instances
- Methods - Special functions that implement operations that are usually performed only on instances of the class
- Events - Messages that are defined by classes and broadcast by class instances when some specific action occurs
- Objects - Instances of classes, which contain actual data values stored in the objects' properties
- Subclasses - Classes that are derived from other classes and that inherit the methods, properties, and events from those classes (subclasses facilitate the reuse of code defined in the superclass from which they are derived).
- Superclasses - Classes that are used as a basis for the creation of more specifically defined classes (i.e., subclasses).
- Packages - Folders that define a scope for class and function naming [52].

### 5.3.5 Comments

Comment lines begin with the character `'//'`, and anything after a `'//'` character is ignored by the interpreter. The `//` character itself only tells the interpreter to ignore the remainder of the same line. In the Classification system, commented areas are printed in green by default, so they should be easy to identify. There are two useful keyboard shortcuts for adding and removing chunks of comments. Select the code to be commented

or uncommented, and then press Ctrl-K + Ctrl-C to place one `//` symbol at the beginning of each line and Ctrl-K + Ctrl -U to do the opposite.

Comments for blocks of code are started by a `/*` and are delimited by a `*/`. In WPF, the symbol `<!--` is used to begin a comment block while `-->` is used to end it. The `'''` comment character is used just before major functions to indicate the role of the specific function. Unlike the `'''` comment character, it is grey by default.

### **Common uses**

Comments are useful for explaining what function a certain piece of code performs especially if the code relies on implicit or subtle assumptions or otherwise perform subtle actions. For example,

```
// Calculate average velocity, assuming acceleration is constant
// and a frictionless environment.
//force = mass * acceleration [52].
```

## **5.4 Difficulties Encountered and Strategies Used to Tackle**

This project had several key challenges, which had to be addressed and resolved. The most pertinent few have been listed here.

### **5.4.1 Crawling and Classification Logic**

The task of crawling the entire web page and extracting content and HTML related features from each URL obtained was extremely time consuming and difficult. Each web page has its own structure of embedding data and most web pages are not crawler or search-engine friendly in terms of metadata and indexing.

To solve this problem, domain specific parsing was used to narrow down the scope of pages along with filters given to the crawler so as not to crawl stylesheets, blogs, downloadable content or other non MIME data which could not be processed.

While classifying this data, the challenge encountered was the presence of ‘outliers’ or rogue characters that appear in data due to incorrect webpage design. These characters could abruptly terminate a string, cause exceptions and obfuscate data. In order to normalize and clean the data, these characters have to be eliminated at an early stage. Hence the feature string was made to be of uniform and consistent format for all links crawled in order to solve this problem and classify efficiently.

### **5.4.2 Storage and Display Logic**

Since the system aims to display a large amount of information, it must be able to store this information efficiently. It must be ensured that data is saved after the Bookmark button is pressed. Also, it must be ensured that all data is once again displayed when the system is re-opened.

To solve this problem, a CSV file is used to store data which is appended with new bookmarks each time. Another CSV file is used to store all the data used for the system, as the output of classification. This way data can be uniformly stored and queried using simple SQL queries.

## **5.5 Summary**

This chapter has presented a concise explanation of the various objected oriented concepts and theoretical fundamentals used in implementing the various functions and features envisioned in this project. The use of various standards and conventions in variable naming, use of modularity, object oriented programming and storage or display logic have been detailed to describe the efficiency of the project in complying with standard industry practices in coding,



## Chapter 6

### Software testing of classification of unstructured data from online course web pages

A primary purpose of testing is to detect software failures so that defects may be discovered and corrected. Testing cannot establish that a product functions properly under all conditions but can only establish that it does not function properly under specific conditions. The scope of software testing often includes examination of code as well as execution of that code in various environments and conditions as well as examining the aspects of code: does it do what it is supposed to do and do what it needs to do. In the current culture of software development, a testing organization may be separate from the development team. There are various roles for testing team members. Information derived from software testing may be used to correct the process by which software is developed.

#### 6.1 Testing Environment

Testing was done on a DELL laptop, connected to a power supply (not on battery power), with the following specifications:

##### Hardware Specifications:

- Intel ® Core ™ i5 CPU
- 4.00 GB RAM
- 32 bit OS
- 500GB HDD

##### Software Specifications:

- Windows 7 OS
- Netbeans IDE

## 6.2 Unit Testing of Main Modules

In computer programming, unit testing is a method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine if they are fit for use. Intuitively, one can view a unit as the smallest testable part of an application. In procedural programming a unit could be an entire module but is more commonly an individual function or procedure. In object-oriented programming a unit is often an entire interface, such as a class, but could be an individual method. Unit tests are created by programmers or occasionally by white box testers during the development process.

### 6.2.1 Unit Testing of Crawler Module

The Crawler unit is tested and test cases are tabulated in the tables. Two tests were performed to check whether basic crawler and multiple crawler were working. Basic crawler filters images and other non required information from a web page. It filters out non relevant web pages like search, account, login, rss etc. It checks whether the given url should be crawled be based on a crawler logic. Multiple crawler works with multiple crawlers crawling simultaneously .Both tests were found to have been successful. Tables 6.1 and 6.2 reference these two tests.

**Table 6.1: Unit Test Case 1: Checks if Basic crawler works**

<b>Sl.no of the test case</b>	Unit test case-1
<b>Name of the test</b>	Test to check if basic crawler works
<b>Featured being tested</b>	Crawler module
<b>Description</b>	Basic crawler filters out irrelevant information from a web page
<b>Sample input</b>	URL
<b>Expected output</b>	.The relevant url
<b>Actual output</b>	Same as above
<b>Remarks</b>	Working

The unit test case 1 is to check whether basic crawler filters images and other non required information from a web page correctly. It checks whether the filtering of irrelevant web pages is happening. This ensures that the crawler is able to bring out only

pages which can be featurized correctly in the required feature string format for the classification process to take place.

**Table 6.2: Unit Test Case 2: Checks if multiple crawler works**

<b>Sl.no of the test case</b>	Unit Test case-2
<b>Name of the test</b>	Test to check if multiple crawler works
<b>Featured being tested</b>	Crawler module
<b>Description</b>	This class decides which URLs should be crawled and handles the downloaded page
<b>Sample input</b>	A web URL.
<b>Expected output</b>	relevant web page
<b>Actual output</b>	Same as above
<b>Remarks</b>	Working

### 6.2.2 Unit Test of Featurizer module

The featurizer unit is tested and the test case is tabulated in the following table 6.3. The test case checks whether the feature string is extracted correctly. The features involved in the feature string are URL, the title of the webpage, meta-description, course name, keywords, social tags, provider, price, duration, certificate, instructor, description and level.

The above mentioned test case fails because the featurizer encounters a page of unrecognized file type which may be a multimedia file, document or stylesheet. Therefore modifications are made to the crawler in order to filter out such pages and include only pure HTML pages and remove unnecessary ones. The test is repeated after revision of code below in Table 6.4.

**Table 6.3: Unit Test Case 3: Check if featurizer works**

<b>Sl.no of the test case</b>	Unit Test case-3
<b>Name of the test</b>	Test to check if featurizer works
<b>Featured being tested</b>	Featurizer module
<b>Description</b>	The featurizer has to return the feature string containing all the HTML page features, social features, URL features, content features.
<b>Sample input</b>	A web URL.
<b>Expected output</b>	The feature string with url, title, description, course name, keywords, social tags, provider, price, duration, certificate, instructor, description, instructor
<b>Actual output</b>	The featurizer returns a java invalid file type exception and does not print the required feature string
<b>Remarks</b>	Not Working

**Table 6.4: Unit Test Case 4: Check if featurizer works**

<b>Sl.no of the test case</b>	Unit Test case-4
<b>Name of the test</b>	Test to check if featurizer works
<b>Featured being tested</b>	Featurizer module
<b>Description</b>	The featurizer returns the feature string containing all the HTML page features, social features, URL features, content features.
<b>Sample input</b>	A web URL.
<b>Expected output</b>	The feature string with url, title, description, course name, keywords, social tags, provider, price, duration, certificate, instructor, description, instructor
<b>Actual output</b>	Same as above
<b>Remarks</b>	Working

### 6.2.3 Unit Test of Classifier module

The classifier unit is tested and the test case is tabulated in the following table 6.5. The test case for this module checks if the datasets given as input to the classifier is compatible for the classifier to run.

**Table 6.5: Unit Test Case 5: Check if classifier works**

<b>Sl.no of the test case</b>	Unit test case-5
<b>Name of the test</b>	Test to check if classifier works properly
<b>Featured being tested</b>	Classifier module
<b>Description</b>	To test whether the the classifier model is trained with appropriate trained dataset. To compare the results of various classification algorithms.
<b>Sample input</b>	A trained dataset.
<b>Expected output</b>	Correctly classified output with good and efficient results
<b>Actual output</b>	Same as above
<b>Remarks</b>	Working

### 6.2.4 Unit Test of Database module

The database unit is tested and the test case is tabulated in the following table 6.6. The test case for this module checks if relevant results are returned as per the user query. The database here uses a Comma Separated Value (CSV) file as input and allows the user to perform Structured Query Language (SQL) queries on the data in this file and treat it as a conventional database. Commands are issued from the java code interface using an Application Programming Interface (API) similar to Java Data Base Connectivity (JDBC). This open source driver is called CSVJDBC and is linked to the front end by the java code that accepts user query requirements to display results fetched from the data source.

**Table 6.6: Unit Test Case 6: Checks if Database module works**

<b>Sl.no of the test case</b>	Unit test case-6
<b>Name of the test</b>	Test to check if database module works properly. A csvjdbc driver is used to read the data.
<b>Featured being tested</b>	Database module
<b>Description</b>	CsvJdbc is a read-only JDBC driver that uses Comma Separated Value (CSV) files or DBF files as database tables.
<b>Sample input</b>	A user query
<b>Expected output</b>	Return of relevant results
<b>Actual output</b>	Same as above
<b>Remarks</b>	Working

### 6.2.5 Unit Test of User Interface module

The user interface unit is tested and the test case is recorded in the following table 6.7. The test case for this module checks if the front end works properly with proper loading. The front end involves options such as search, update and bookmark. The search option enables the user to search for online course by the course name, course provider, instructor, price, level etc. The update option delivers the updated information about the latest online courses available. The bookmark option lets the user to bookmark the courses for later reference.

**Table 6.7: Unit Test Case 6: Checks if User Interface module works**

<b>Sl.no of the test case</b>	Unit test case-6
<b>Name of the test</b>	Tests to check if user interface module works with all the functionality such as search, update and bookmark.
<b>Featured being tested</b>	User interface module, Database module
<b>Description</b>	To check the functioning of search, update and bookmark function.
<b>Sample input</b>	A user input.
<b>Expected output</b>	Appropriate display of the results.
<b>Actual output</b>	Same as above
<b>Remarks</b>	Working

### 6.3 Integration Testing Of Modules

In integrated testing, the modules are joined to form sub-systems which each perform a specific function. The performance of the product is determined by its holistic performance. 3 integration tests performed are at the front end for search, update and bookmarks options. table 6.8, 6.9 and 6.10 illustrate the test cases.

**Table 6.8: Integrated test case 1: Test case to see if search option in the user interface works**

<b>Sl.no of the test case</b>	Integrated test case-1
<b>Name of the test</b>	Tests to check if the search function works fine.
<b>Featured being tested</b>	Search option.
<b>Description</b>	There are 6 types of searches that can be made by the user. The user can search by course, provider, level, instructor, price and certification.
<b>Sample input</b>	A user input according to the required search option.
<b>Expected output</b>	Appropriate display of the results. The display includes the course, provider instructor, level, price, certification and a brief description about the course
<b>Actual output</b>	Same as above
<b>Remarks</b>	Working

**Table 6.9: Integrated test case 2: Test case to see if update option in the user interface works**

<b>Sl.no of the test case</b>	Integrated test case-2
<b>Name of the test</b>	Tests to check if the update function works fine.
<b>Featured being tested</b>	User interface module, crawler module, featurizer module & classifier module
<b>Description</b>	The update option is to bring the all the new courses introduced in the recent past to the user in a consolidated manner. Upon the selection of the update option the crawler is set off in action to crawl for all the new course webpage urls. Once crawled these pages are featurized which is then given to the trained classifier to classify the webpage.
<b>Sample input</b>	Selection of the update option by the user.
<b>Expected output</b>	Appropriate display of the results. The display includes the course, provider instructor, level, price, certification and a brief description about the course.
<b>Actual output</b>	Same as above
<b>Remarks</b>	Working

**Table 6.10: Integrated test case 3: Test case to see if bookmark option in the user interface works**

<b>Sl.no of the test case</b>	Integrated test case-3
<b>Name of the test</b>	Tests to check if the bookmark function works fine.
<b>Featured being tested</b>	bookmark option in the user interface module.
<b>Description</b>	The bookmark option lets the user bookmark any course for future reference
<b>Sample input</b>	Selecting the bookmark option for a course by the user.
<b>Expected output</b>	The bookmarked courses get stored in the database and the list of selected course gets displayed once the user chooses to see them. Further the user can view the details of the bookmarked courses individually
<b>Actual output</b>	Same as above
<b>Remarks</b>	Working

## 6.4 System Testing

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic. The entire course portal is tested from start to finish, including testing of all conditions and permutations queries. The test was found to be successful. The Test Case is tabulated in table 6.11.



**Table 6.11: System testing**

<b>S.no of the test case</b>	System testing
<b>Name of the test</b>	Holistic system test
<b>Featured being tested</b>	User Interface module
<b>Description</b>	The search, update and bookmark option will be tested. Also the system testing checks the display of the results in the appropriate pattern.
<b>Sample input</b>	Selecting search, update bookmark, show all bookmark and clear all bookmarks options.
<b>Expected output</b>	If there is match to the user query in the search option the results are displayed, if there is no match found then a message is displayed intimating about the absence of the course the user is looking for. If the update option is chosen by the user the appropriate results are displayed. If the bookmark option is displayed the selected courses by the user gets stored in the database for future reference. The clear all bookmarks option clears the list of previously bookmarked courses.
<b>Actual output</b>	Same as above
<b>Remarks</b>	Working

The system test is used to ensure the compliance of all modules of the system. It tests to see if the various modules work in synchronization with each other to provide the expected results. If the test is a success, it indicates the application is ready for use.

## 6.5 Summary

This chapter has detailed, listed and evaluated the testing process of various components in an integrated manner to show successful working of the system under multiple constraints and in various stages of working, as well from the user's point of view to fulfil the characteristics mentioned in the requirements specification.

## Chapter 7

### Experimental Analysis and Results of classification of unstructured data from online course web pages

This chapter lists the results of the project and the inferences to be made from the testing results. The evaluation metrics have been listed and the results have been accordingly quantified. A real dataset was taken into consideration and the algorithms run on this set. The results are specific to this dataset but give an indication as to the general performance trend of the various classification algorithms. One algorithm each has been chosen from various domains namely decision-tree based, mathematical, probabilistic, lazy learner and ensemble methods.

#### 7.1 Evaluation Metrics

Classifier performance depends greatly on the characteristics of the data to be classified. There is no single classifier that works best on all given problems, a phenomenon that may be explained by the no-free-lunch theorem. Various empirical tests have been performed to compare classifier performance and to find the characteristics of data that determine classifier performance. Determining a suitable classifier for a given problem is however still more an art than a science.

The measures **precision, recall and f-measure** are popular metrics used to evaluate the quality of a classification system. **Receiver Operating Characteristic (ROC)** curves have been used to evaluate the tradeoffs between true- and false-positive rates of classification algorithms.

**True positives** are the number of items correctly labelled as belonging to the positive class while **false positives** are items incorrectly labelled as belonging to the class. These measures can be expressed as fractions of the entire data set, known as TP rate and FP rate.

Precision (also called positive predictive value) is the fraction of retrieved instances that are relevant, while recall (also known as sensitivity) is the fraction of

relevant instances that are retrieved. Both precision and recall are therefore based on an understanding and measure of relevance. In classification, the precision for a class is the number of true positives (i.e. the number of items correctly labelled as belonging to the positive class) divided by the total number of elements labelled as belonging to the positive class (i.e. the sum of true positives and false positives, which are items incorrectly labelled as belonging to the class) [54].

Recall in this context is defined as the number of true positives divided by the total number of elements that actually belong to the positive class (i.e. the sum of true positives and false negatives, which are items which were not labelled as belonging to the positive class but should have been) [54].

The F-measure is the harmonic mean of precision and recall, giving an overall expression of performance of an algorithm. A Receiver Operating Characteristic (ROC), [55] or simply ROC curve, is a graphical plot which illustrates the performance of a binary classifier system as its discrimination threshold is varied. It is created by plotting the fraction of true positives out of the total actual positives (TPR = True Positive Rate) vs. the fraction of false positives out of the total actual negatives (FPR = False Positive Rate), at various threshold settings.

## 7.2 Experimental Dataset

The initial dataset used for bootstrapping consisted of 350 positive and negative samples taken from open course websites, labelled manually and designed to cover all corner case data values that could appear while classifying crawled web pages. This dataset was used to evaluate various classification algorithms and compare their performance with the random forest classifier written for this project. 66% of this data was used as training set while the remaining was used as a test set. The data was randomly segregated into these two sets in order to evaluate the performance of various algorithms.

## 7.3 Performance Analysis

The data obtained for various algorithms is shown below for the various metrics listed above. The ROC curves and measures for the bootstrap dataset have been

mentioned, as these are accurately labelled data and will provide a clear idea of the performance of these algorithms.

### 7.3.1 C4.5 Algorithm Evaluation

The C4.5 Algorithm is based on decision tree approach to classification. The algorithm was evaluated on training and test sets as shown below. Table 7.1 and 7.3 contain the detailed accuracy by class for the algorithm on each of these datasets, while table 7.2 and 7.4 describe the confusion matrix for the class labels on the same.

#### Evaluation on training dataset

Correctly classified instances	161	88.4615%
Incorrectly classified instances	21	11.5385%

**Table 7.1: Detailed accuracy by class on training set for C4.5 algorithm**

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC area	Class
Weighted avg.	0.805	0.57	0.912	0.805	0.855	0.902	no
	0.943	0.195	0.868	0.943	0.904	0.867	yes
	0.885	0.137	0.887	0.885	0.883	0.882	

**Table 7.2: Confusion matrix of training set for C4.5 algorithm**

a	b	
62	15	a = no
6	99	b = yes

#### Re-evaluation on test dataset

Correctly classified instances	100	72.9927%
Incorrectly classified instances	37	27.0073%

**Table 7.3: Detailed accuracy by class on test set for C4.5 algorithm**

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC area	Class
	0.469	0.41	0.909	0.469	0.619	0.911	no
	0.959	0.531	0.673	0.959	0.791	0.911	yes
Weighted avg.	0.73	0.302	0.783	0.73	0.71	0.911	

**Table 7.4: Confusion matrix of test set for C4.5 algorithm**

a	b	
30	34	a = no
3	70	b = yes

Evaluation of bootstrap data on training and test data is mentioned above respectively. Accuracy is 72.9927%.

Figure 7.1 depicts the area under the curve for the instances classified as relevant online courses by C4.5 classification algorithm. The area under the curve was observed to be **0.9114**. This is reasonably high but holds scope for improvement.

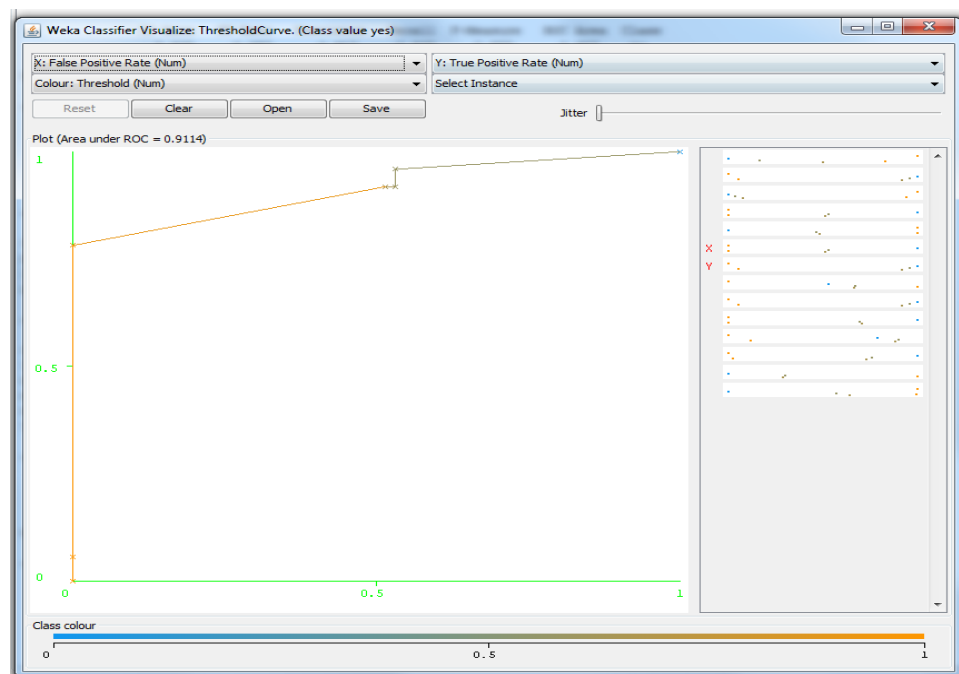
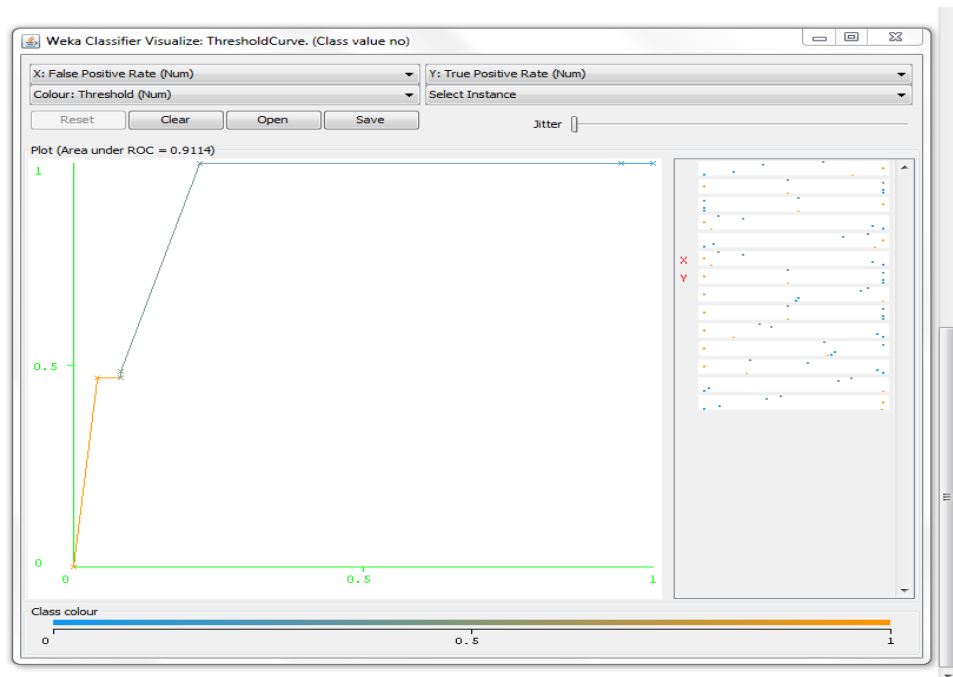
**Figure 7.1: ROC curve of C4.5 classification algorithm for the class value 'yes'**

Figure 7.2 depicts the area under the curve for the instances classified as non-relevant online courses by C4.5 classification algorithm. The area under the curve was observed to be 0.9114. This is reasonably high but holds scope for improvement.



**Figure 7.2: ROC curve of C4.5 classification algorithm for the class value ‘no’**

### 7.3.2 Logistic Regression Algorithm Evaluation

The Logistic Regression algorithm is based on mathematical approach to classification. The algorithm was evaluated on training and test sets as shown below. Table 7.5 and 7.7 contain the detailed accuracy by class for the algorithm on each of these datasets, while table 7.6 and 7.8 describe the confusion matrix for the class labels on the same.

#### Evaluation on training dataset

Correctly classified instances	181	99.4505%
Incorrectly classified instances	1	0.5495%

**Table 7.5: Detailed accuracy by class on training set for Logistic Regression algorithm**

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC area	Class
Weighted avg.	0.987	0.0	1	0.987	0.855	0.902	no
	1	0.013	0.991	1	0.904	0.867	yes
	0.995	0.007	0.995	0.995	0.883	0.882	

**Table 7.6: Confusion matrix of training set for Logistic Regression algorithm**

a	B	
76	1	a = no
0	105	b = yes

Re-evaluation on test dataset

Correctly classified instances	104	75.9124%
Incorrectly classified instances	33	24.0876%

**Table 7.7: Detailed accuracy by class on test set for Logistic Regression algorithm:**

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC area	Class
Weighted avg.	0.484	0.0	1	0.484	0.653	0.963	no
	0.1	0.516	0.689	1	0.816	0.963	yes
	0.759	0.275	0.834	0.759	0.739	0.963	

**Table 7.8: Confusion matrix of test set for Logistic Regression algorithm**

a	B	
31	33	a = no
0	73	b = yes

Evaluation of bootstrap data on training and test data is mentioned above respectively. Accuracy is 75.9124%.

Figure 7.3 depicts the area under the curve for the instances classified as non-relevant online courses by Logistic Regression algorithm. The area under the curve was observed to be 0.9631. It is slightly higher than the area given by C4.5 algorithm i.e 0.9114 thereby performing slightly better.



Weka Classifier Visualize: ThresholdCurve. (Class value yes)

X: False Positive Rate (Num) Y: True Positive Rate (Num)

Colour: Threshold (Num) Select Instance

Reset Clear Open Save

Jitter

Plot (Area under ROC = 0.9631)

1 0.5 0

0 0.5 1

Class colour

0.000021 0.5 1

**Figure 7.4: ROC curve of Logistic Regression classification algorithm for the class value ‘yes’**



### 7.3.3 Naive Bayes Algorithm Evaluation

The Naive Bayes algorithm is based on probabilistic approach to classification. The algorithm was evaluated on training and test sets as shown below. Table 7.9 and 7.11 contain the detailed accuracy by class for the algorithm on each of these datasets, while table 7.10 and 7.12 describe the confusion matrix for the class labels on the same.

#### Evaluation on training dataset

Correctly classified instances	172	94.5055%
Incorrectly classified instances	10	5.5495%

**Table 7.9: Detailed accuracy by class on training set for Naive Bayes algorithm**

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC area	Class
Weighted avg.	1	0.095	0.855	1	0.939	0.99	no
	0.905	0	1	0.905	0.95	0.983	yes
	0.945	0.04	0.951	0.945	0.945	0.986	

**Table 7.10: Confusion matrix of training set for Naive Bayes algorithm**

a	B	
77	0	a = no
10	95	b = yes

#### Re-evaluation on test dataset

Correctly classified instances	104	88.3212%
Incorrectly classified instances	33	11.6788%

**Table 7.11: Detailed accuracy by class on test set for Naive Bayes algorithm**

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC area	Class
Weighted avg.	1	0.219	0.8	1	0.889	0.957	no
	0.781	0	1	0.781	0.877	0.957	yes
	0.883	0.102	0.907	0.883	0.883	0.957	

**Table 7.12: Confusion matrix of test set for Naive Bayes algorithm**

a	B	
64	0	a = no
16	57	b = yes

Evaluation of bootstrap data on training and test data is mentioned above respectively. Accuracy is 88.3212%.

Figure 7.5 depicts the area under the curve for the instances classified as non-relevant online courses by Naive Bayes classification algorithm. The area under the curve was observed to be 0.9571.

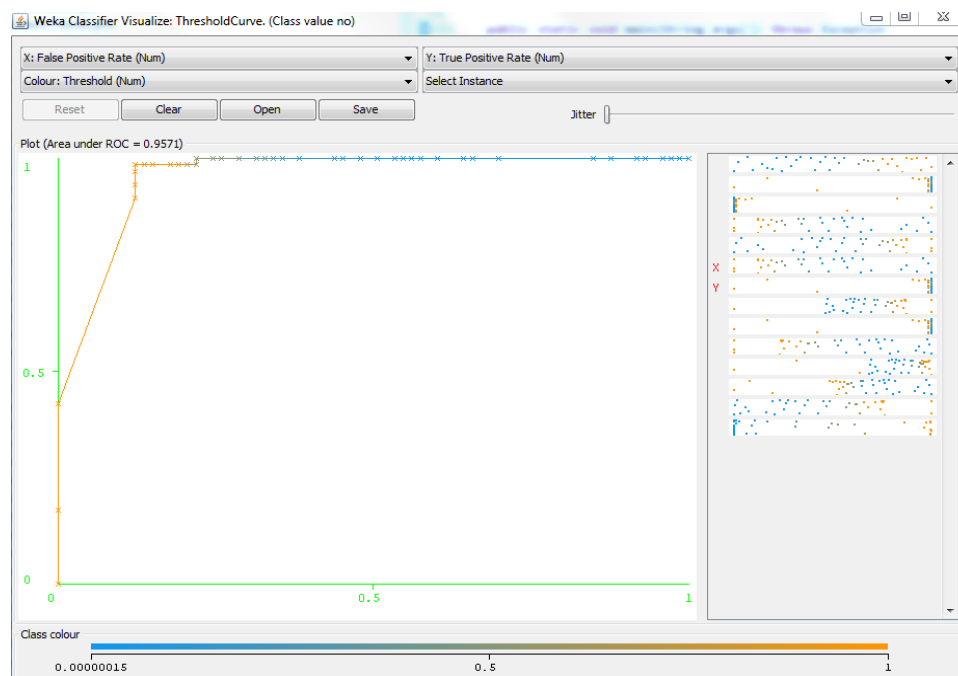
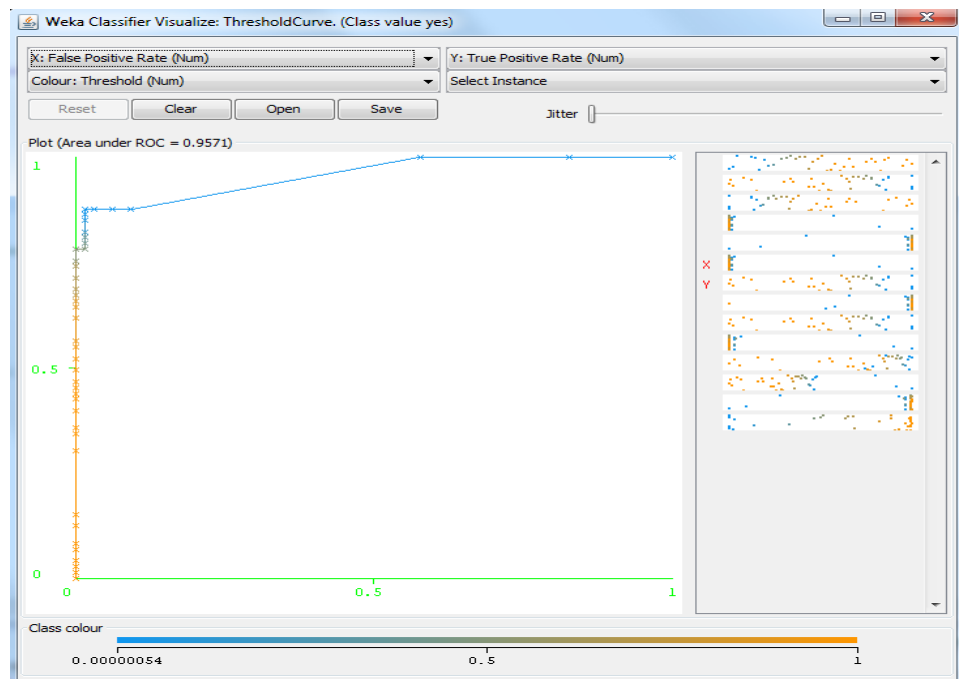
**Figure 7.5: ROC curve of Naive Bayes classification algorithm for the class value ‘no’**

Figure 7.6 depicts the area under the curve for the instances classified as relevant online courses by Naive Bayes classification algorithm. The area under the curve was observed to be 0.9571. It can be observed that the area is slightly higher for Naive Bayes as compared to C4.5.



**Figure 7.6: ROC curve of Naive Bayes classification algorithm for the class value ‘yes’**

### 7.3.4 K Nearest Neighbours Algorithm Evaluation

The K Nearest Neighbours algorithm is based on lazy learning approach to classification. The algorithm was evaluated on training and test sets as shown below. Table 7.13 and 7.15 contain the detailed accuracy by class for the algorithm on each of these datasets, while table 7.14 and 7.16 describe the confusion matrix for the class labels on the same.

#### Evaluation on training dataset

Correctly classified instances	124	90.5109%
Incorrectly classified instances	13	9.4891%

**Table 7.13: Detailed accuracy by class on training set for K Nearest Neighbours algorithm**

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC area	Class
Weighted avg.	0.984	0.164	0.84	0.984	0.906	0.939	no
	0.836	0.016	0.984	0.836	0.904	0.939	yes
	0.905	0.085	0.917	0.905	0.905	0.939	

**Table 7.14: Confusion matrix of training set for K Nearest Neighbours algorithm**

a	B	
63	1	a = no
12	61	b = yes

Re-evaluation on test dataset

Correctly classified instances	121	88.3212%
Incorrectly classified instances	16	11.6788%

**Table 7.15: Detailed accuracy by class on test set for K Nearest Neighbours algorithm**

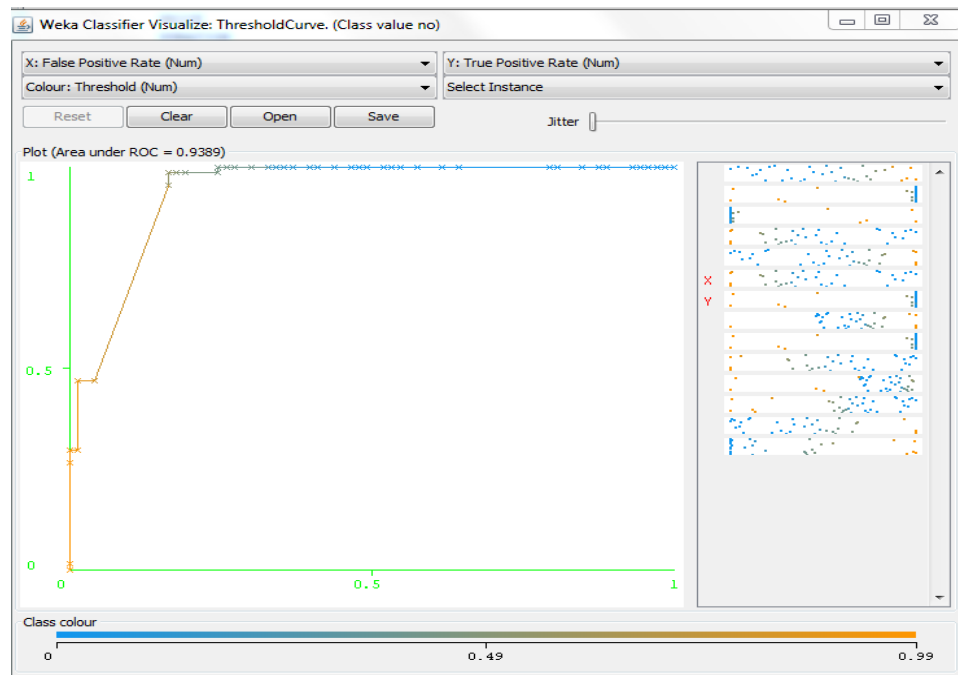
	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC area	Class
Weighted avg.	1	0.219	0.8	1	0.889	0.957	no
	0.781	0	1	0.781	0.877	0.957	yes
	0.883	0.102	0.907	0.883	0.883	0.957	

**Table 7.16: Confusion matrix of test set for K Nearest Neighbours algorithm**

a	B	
64	0	a = no
16	57	b = yes

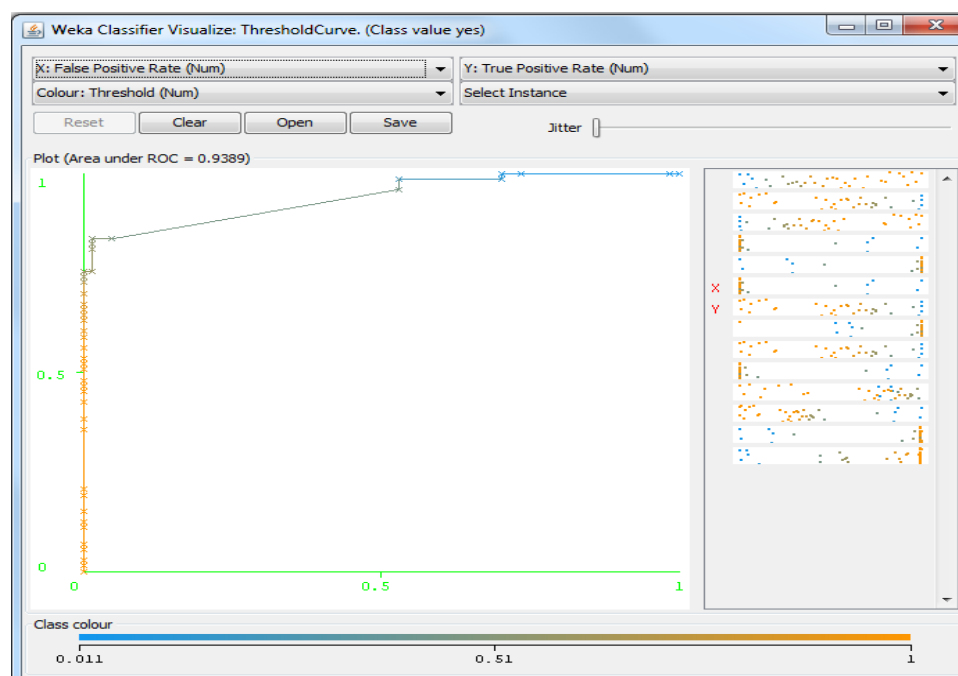
Evaluation of bootstrap data on training and test data is mentioned above respectively. Accuracy is 88.3212%.

Figure 7.7 depicts the area under the curve for the instances classified as non-relevant online courses by K Nearest neighbour classification algorithm. The area under the curve was observed to be 0.9389. The area is slightly lower as compared to the area of Naive Bayes.



**Figure 7.7: ROC curve of K Nearest neighbour classification algorithm for the class value 'no'**

Figure 7.8 depicts the area under the curve for the instances classified as relevant online courses by K Nearest neighbour classification algorithm. The area under the curve was observed to be 0.9389. The area is slightly lower as compared to the area of Naive Bayes.



**Figure 7.8: ROC curve of K Nearest classification algorithm for the class value 'yes'**

### 7.3.5 Random Forest Algorithm Evaluation

The Random Forest algorithm is based on ensemble approach to classification. The algorithm was evaluated on training and test sets as shown below. Table 7.17 and 7.19 contain the detailed accuracy by class for the algorithm on each of these datasets, while table 7.18 and 7.20 describe the confusion matrix for the class labels on the same.

#### Evaluation on training dataset

Correctly classified instances	182	100%
Incorrectly classified instances	0	0%

**Table 7.17: Detailed accuracy by class on training set for Random Forest algorithm**

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC area	Class
Weighted avg.	1	0	1	1	1	1	no
	1	0	1	1	1	1	yes
	1	0	1	1	1	1	

**Table 7.18: Confusion matrix of training set for Random Forest algorithm**

a	B	
77	0	a = no
0	105	b = yes

#### Re-evaluation on test dataset

Correctly classified instances	127	92.7007%
Incorrectly classified instances	10	7.2993%

**Table 7.19: Detailed accuracy by class on test set for Random Forest algorithm:**

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC area	Class
Weighted avg.	0.984	0.123	0.875	1	0.984	0.926	no
	0.877	0.016	0.985	0.781	0.877	0.928	yes
	0.927	0.066	0.933	0.883	0.927	0.927	

**Table 7.20: Confusion matrix of test set for Random Forest algorithm**

a	B	
63	1	a = no
9	64	b = yes

Evaluation of bootstrap data on training and test data respectively. Accuracy is 92.7007%.

Figure 7.9 depicts the area under the curve for the instances classified as non-relevant online courses by Random Forest classification algorithm. The area under the curve was observed to be 0.9605. This value is recorded as the highest value as compared to the area value observed in other algorithms.

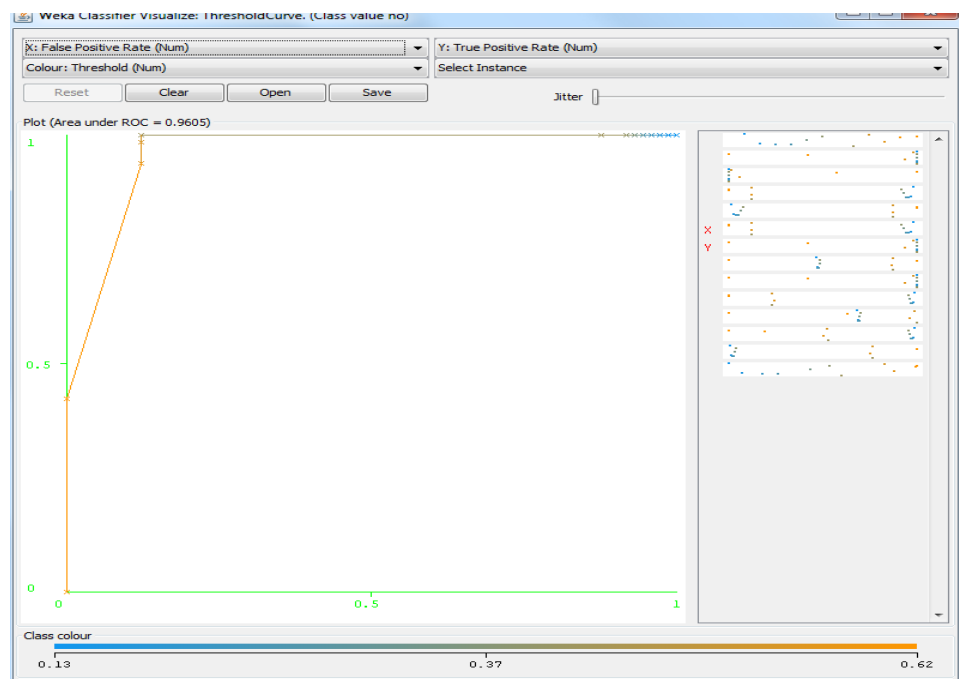
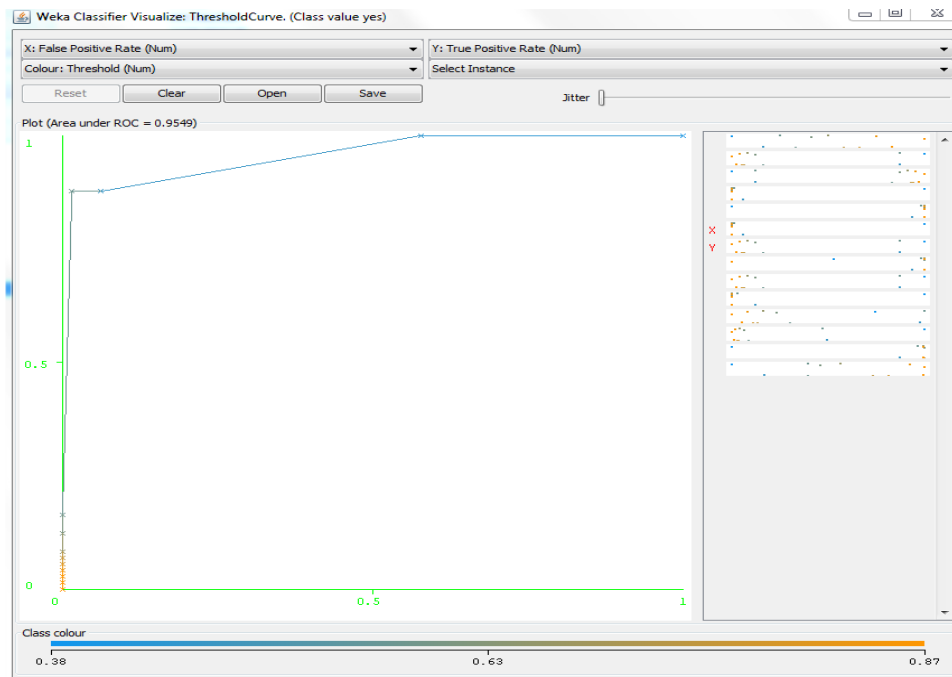
**Figure 7.9: ROC curve of Random Forest classification algorithm for the class value ‘no’**

Figure 7.10 depicts the area under the curve for the instances classified as non-relevant online courses by Random Forest classification algorithm. The area under the curve was observed to be 0.9549. This value is recorded as the highest value as compared to the area value observed in other algorithms.



**Figure 7.10: ROC curve of Random Forest classification algorithm for the class value ‘yes’**

## 7.4 Inference from the Result

The experimental analysis presented above has been classified under various heads such as precision, recall, F Measure and others as defined earlier. Under each of these parameters, it can be seen that the decision-tree based C 4.5 algorithm is the weakest of all. The probabilistic Naive Bayes and the lazy learner K Nearest Neighbour have identical results for the dataset used and the mathematical Logistic Regression is better than both of them. However, the ensemble based Random Forest algorithm has outperformed all the other algorithms.

It is also noticed that for all the algorithms mentioned above, the training set precision is sufficiently high but the test set precision is lesser, with the gap narrowing in respective order between training and test-set precision.

In terms of time and computational resources, all the algorithms had nearly identical performance and since the focus of this research is on the precision of the algorithms, this factor is ignored during the differentiation and comparison of the various algorithms.

Various methods such as random splitting of train and test data, cross-validation and others were considered however they did not substantially order the relative performance of the algorithms or the type of predictor used. This shows us that the



advantage that any one algorithm has over another is not merely for a particular type of test-set, but across all forms of testing and hence it would be safe to conclude a relative order of superiority as proposed. The standard split of 66% training data and 33% test data was adopted for presentation of results.

It is also seen that for the ROC curves, the area under the curve is sufficiently high for almost all algorithms for higher threshold values whereas for lower threshold values, there is a marked difference. This is a clear indication of the behaviour of the algorithm as the dataset size increases, because there will be a natural increase in the number of true positives and corresponding false positives. These curves therefore help us to predict the scalability of the algorithms and the resulting compromise on precision.

It is clear from the experimental analysis that the Random Forest algorithm outperforms all others not only in training set evaluation metrics, but more importantly in test set evaluation metrics. It can be inferred that this algorithm is more capable of dealing with hitherto unseen data type samples and the areas under ROC curves show that for Random Forest algorithm, varying of threshold intensity does not lower the performance significantly as is done in other algorithms and hence this algorithm works better with large datasets as well. Table 7.1 gives the comparison of all the five algorithms.

**Table 7.21: Accuracies of the algorithms**

Classification Algorithm	Accuracy (%)
C4.5	72.9
Logistic Regression	75.9
Naive Bayes	88.3
K Nearest Neighbour	88.3
Random Forest	92.7

Hence Random Forest Classification algorithm has been used in all further stages of development for prediction of class labels for a large data corpus.

## 7.5 Summary

The seventh chapter describes in detail the experimental results and its analysis of all the five classification algorithms – C4.5, Naïve Byes, K Nearest Neighbor, Logistic Regression and Random Forest. The evaluation metrics like confusion matrix, ROC curves and accuracies of the respective algorithms have been shown. The best of the algorithm has been chosen for the developmental purposes.

## Chapter 8

### Conclusion

In the vast and nebulous research done to classify unstructured data from web pages, there was hitherto no single effort to compare multiple algorithms for a representative, multi-domain dynamic and text-rich webpage dataset. This project has fulfilled this need by examining Naive Bayes, K Nearest Neighbours, Decision Trees, Logistic Regression and Random Forest. They have been compared upon various parameters and the results have been used in deciding on a suitable algorithm for use in the project. Random Forest algorithm was chosen as it achieved 92.7% precision and outperformed all other classification algorithms. The project has used novel methods of text extraction which require least processing resources in terms of feature subsetting and extraction of data from provider-specific pages without unnecessary inflation of data size. Hitherto unused features such as social tagging from websites like Delicious have also been included to improve the quality and relevance of data. The web data of over 30,000 web pages crawled and classified using Random Forest algorithm by the system. The offline repository covers over 10,000 courses and over 60 providers all over the world.

This system is presented in an innovative application that can work offline and allow the user to bookmark and view course information in a crisp and concise manner with a fresh and intuitive front end interface. It will provide a hand reference to students on the move as well as academic institutions. It can even update based on the latest courses and display information which is dynamically generated based on training set data.

### 8.1 Limitations

The main limitation of this is that there is no way to assure 100% accuracy of courses labelled by the classifier. This is because the very aim of this project is to eliminate manual effort in collection and management of data. However due to inconsistent HTML code and unconventional webpage design, errors might creep in to the code of the pages crawled which the classifier has not estimated or encountered earlier. These could include common HTML design errors or larger dynamic script related bugs. The system will have to be retrained and reset for adding new providers.

## 8.2 Future Enhancements

The system has scope for multiple enhancements that depend on advancements in web and data mining technologies. The future enhancements that address the advancements are listed and explained below:

- Super-fast multithreaded crawlers or even a server resource with indexing capacity can be used to fetch and store a large number of web pages quickly. This will make it possible to re-train the system easily whenever required.
- Use of Twitter, Facebook and other improved feeds if available offline will help to enhance the quality of data in social tagging.
- The system can be expanded to cover other languages using Natural Language Processing concepts for different scripts and providers.
- Inclusion of multimedia in the system can enable users to view video courses offline using efficient resource utilization to fetch and classify this data

The system has been designed keeping in mind the potential to convert it to a fully functional mobile application. This can be taken further and made available to users across multiple mobile platforms such as Android, Windows Phone and iOS so that it is accessible to multiple users and will help those in remote areas who do not have fast internet connection to efficiently plan and schedule the courses they wish to take.

## 8.3 Summary

The eighth chapter describes how the project work has fulfilled the research gap. It has mentioned about new features such as social tagging that were used and main front end display features. The limitations of the project work were discussed that mainly holds inconsistent HTML code and inconsistent web page design responsible for inefficient crawling and classification. The last part of the chapter discussed the scope of the project that addresses the future enhancements of the project.

## References

- [1] Bhardwaj, Aanshi, and Veenu Mangat, “A novel approach for content extraction from web pages”, *Recent Advances in Engineering and Computational Sciences (RAECS)*, 2014, pp. 1-4.
- [2] Fayyad U M, “Data mining and knowledge discovery: Making sense out of data”, *IEEE Intelligent Systems* , vol.11,(5), 2007, pp: 20-25
- [3] “Machine learning: neural networks, genetic algorithms, and fuzzy systems”, John Wiley & Sons Inc., 2007
- [4] Blumberg, Robert and Shaku Atre, “The problem with unstructured data”, *DM REVIEW* 13, 2008, pp: 42-49
- [5] “The Theory and practice of online learning”, Athabasca University Press, 2004
- [6] Esra Saraç and Selma Ayşe Özel, “Web Page Classification Using Firefly Optimization”, *IEEE International Symposium on Innovations in Intelligent Systems and Applications*, 2013, pp: 1-5.
- [7] MD. Ezaz Ahmed and Preeti Bansal, “Clustering Technique on Search Engine Dataset using Data Mining Tool”, *Third International Conference on Advanced Computing & Communication Technologies*, 2013, pp: 86 – 89.
- [8] M’arius Sajgal’ık, Michal Barla, M’aria Bielikov’a, “From ambiguous words to key-concept extraction”, *24th International Workshop on Database and Expert Systems Applications*, 2013, pp: 63-67.
- [9] R Malhotra and A. Sharma, “A Neuro-Fuzzy Classifier for Website Quality Prediction”, *International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2013, pp: 1274 – 1279.
- [10] Tao Yue, Jianhua Sun and Hao Chen, “Fine Grained Mining and Classification of malicious web pages”, *Fourth International Conference on Digital Manufacturing and Automation (ICDMA)*, 2013, pp: 616 – 619.
- [11] Sheau-Ling Hsieh, Wen-Yung Chang, Chi-Huang Chen and Yung-Ching Weng, “Semantic Similarity Measures in the Biomedical Domain by Leveraging a Web Search

- Engine”, *IEEE Journal of Biomedical and Health Informatics*, Volume: 17, Issue: 4, 2013, pp: 853 – 861.
- [12] Qingjie Meng and Changqing Gong, “Research of Web Information Classifying Based on Neural Network”, *6th International Conference on Information Management, Innovation Management and Industrial Engineering*, Volume: 1, 2013, pp: 625 – 628.
- [13] Jingtian Jiang, Xinying Song, Nenghai Yu and Chin-Yew Lin, “FoCUS: Learning to Crawl Web Forums”, *IEEE Transactions on Knowledge and Data Engineering*, Volume 25, 2013, pp : 1293-1306.
- [14] Peng Wang, Mingqi Zhou, Yue You and Xiang Zhang, “A New Vision-Based Method for Extracting Academic Information from Conference Web Pages”, *IEEE 24th International Conference on Tools with Artificial Intelligence*, Volume: 1, 2012, pp: 976 – 981.
- [15] A Pakgohar and M Khalili, “A Probabilistic Relational Model for Keyword Extraction”, *International Conference on Statistics in Science, Business, and Engineering (ICSSBE)*, 2012, pp: 1-5.
- [16] J Krutil, M Kudelka and V Snasel, ”Web Page Classification based on Schema.org Collection”, *Fourth International Conference on Computational Aspects of Social Networks (CASoN)*, 2012, pp : 356 – 360.
- [17] S Samarawickrama, L Jayaratne, ”Effect of Named Entities in Web Page Classification”, *Fourth International Conference on Computational Intelligence, Modelling and Simulation (CIMSIM)*, 2012, pp: 38 – 42.
- [18] Zhaohui Xu, Fuliang Yan, Jie Qin and Haifeng Zhu, “A Web Page Classification Algorithm Based On Link Information”, *Tenth International Symposium on Distributed Computing and Applications to Business, Engineering and Science* , 2011, pp: 82 – 86.
- [19] N Sato, K Komiya, K Fujimoto, Y Kotani, ”Categorization of Product Pages Depending on Information on the Web”, *Eighth International Joint Conference on Computer Science and Software Engineering (JCSSE)*, 2011, pp: 393 – 398.

- [20] Wang Zhixing and Chen Shaohong, “Web Page Classification based on Semi-supervised Naive Bayesian EM Algorithm”, *IEEE 3rd International Conference on Communication Software and Networks (ICCSN)*, 2011, pp: 242 – 245.
- [21] D Taylan, M Poyraz, S Akyokus, M C Ganiz, “Intelligent Focused Crawler: Learning which Links to Crawl”, *International Symposium on Innovations in Intelligent Systems and Applications (INISTA)*, 2011, pp: 504 – 508.
- [22] S A Ozel, “A Genetic Algorithm Based Optimal Feature Selection for Web Page Classification”, *International Symposium on Innovations in Intelligent Systems and Applications (INISTA)*, 2011, pp: 282 – 286.
- [23] Van Lam Le, I Welch, Xiaoying Gao and P Komisarczuk, “Two-Stage Classification Model to Detect Malicious Web Pages”, *International Conference on Advanced Information Networking and Applications*, 2011, pp: 113 – 120.
- [24] C Boden, T Hafele and A Loser, “Classification Algorithms for Relation Prediction”, *IEEE 27th International Conference on Data Engineering Workshops (ICDEW)*, 2011, pp: 46 – 52.
- [25] Qingjie Meng and Changqing Gong, “Web Information Classifying and Navigation based on Neural Network”, *Second International Conference on Signal Processing Systems (ICSPS)*, Volume: 2, 2010, pp: 431-433.
- [26] Boyi Xu, Jing Wang and Hongming Cai, “A Web Page Classification Algorithm and Its Application in E-government System”, *Seventh International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2010)*, Volume: 4, 2010, pp: 1767 – 1771.
- [27] R J Oskouei, “Identifying Students’ Behaviors Related to Internet Usage Patterns”, *International Conference on Technology for Education (T4E)*, 2010, pp: 232-233.
- [28] Daniela Xhemali, Christopher J. Hinde and Roger G Stone, “Naïve Bayes vs. Decision Trees vs. Neural Networks in the Classification of Training Web Pages”, *IJCSI International Journal of Computer Science Issues*, Vol. 4, No. 1, 2009

- 
- [29] S Aliakbary, H Abolhassani, H Rahmani and B Nobakht, "Web Page Classification Using Social Tags", *International Conference on Computational Science and Engineering*, Volume: 4, 2009, pp: 588 – 593.
- [30] Win Thanda Aung and Khin Hay Mar Saw Hla, "Random Forest Classifier for Multi-category Classification of Web Pages", *IEEE Asia-Pacific Services Computing Conference*, 2009, pp: 372 – 376.
- [31] Dongwon Lee, Hung-sik Kim, Eun Kyung Kim, Su Yan, Johnny Chen and Jeongkyu Lee, "LeeDeo: Web-Crawled Academic Video Search Engine", *Tenth IEEE International Symposium on Multimedia*, 2008, pp: 497 – 502.
- [32] Jing Wang, Hongming Cai, Boyi Xu and Lihong Jiang, "CUCS: A Web Page Classification Algorithm for Large Training Set", *IFIP International Conference on Network and Parallel Computing*, 2008, pp: 440 – 445.
- [33] Shiqun Yin, Fang Wang, Zhong Xie and Yuhui Qiu, "Study on Web-page Classification Algorithm Based on Rough Set Theory", *International Symposiums on Information Processing*, 2008, pp: 202 – 206.
- [34] Hsinchun Chen, "IEDs in the Dark Web: Genre Classification of Improvised Explosive Device Web Pages", *IEEE International Conference on Intelligence and Security Informatics*, 2008, pp: 94 – 97.
- [35] Rong Liu, ianzhong Zhou and Ming Liu, "A Graph-based Semi-supervised Learning Algorithm for Web Page Classification", *Sixth International Conference on Intelligent Systems Design and Applications*, Volume: 2, 2006, pp: 856 – 860.
- [36] Mu-Hee Song, Soo-Yeon Lim, Dong-Jin Kang and Sang-Jo Lee, "Automatic Classification of Web Pages based on the Concept of Domain Ontology", *12th Asia-Pacific Software Engineering Conference*, 2005
- [37] Xiaogang Peng and B Choi, "Automated Web Page Classification in a Dynamic and Hierarchical Way", *IEEE International Conference on Data Mining*, 2002, pp: 386 – 393.
- [38] "Introduction to Data Mining", Pearson Education, 2007
- [39] "Data Mining – Concepts and Techniques", Elsevier, 2011



- 
- [40] “Massively Open: How Massive Open Online Courses Changed The World”, CreateSpace Independent Publishing Platform, 2013
- [41] “A Concise Introduction to Software Engineering”, Springer – Verlog London Limited, 2008
- [44] Jiawei Han, “Data mining techniques”, *Proceedings of the 2003 ACM SIGMOD international conference on Management of data, ACM*, 2003, pp: 545
- [45] “Applied logistic regression”, John Wiley & Sons, 2004
- [46] Panda, Mrutyunjaya, and Manas Ranjan Patra, “Network intrusion detection using naive bayes”, *International Journal of Computer Science and Network Security*, 7.12, 2007, pp: 258-263
- [47] Cunningham, Pdraig, and Sarah Jane Delany, “k-Nearest neighbour classifiers”, *Mult Classif Syst*, 2007
- [48] Leo Breiman, “Random Forests Machine Learning”, vol.45, 2001
- [49] “Semantics and verification of data flow in UML 2.0 activities”, *Electronic Notes in Theoretical Computer Science – Elsevier*, 2005
- [50] “A corpus of Late Modern English texts” International Computer Archive of Modern and Medieval English – Elsevier, 2005
- [51] “Class and subclass declarations”, *Software pioneers - Springer Berlin Heidelberg*, 2002
- [52] “C++: The Complete Reference”, Tata McGraw-Hill Education, 2008
- [53] “Object Oriented Programming with C++”, Tata McGraw-Hill Education, 2008
- [54] “A systematic analysis of performance measures for classification tasks”, *Information Processing & Management - Elsevier*, 2009
- [55] “An Introduction to ROC Analysis”, *Pattern recognition- Elsevier*, 2006

## Appendices

### Appendix A

#### Screen Shots of User Interface

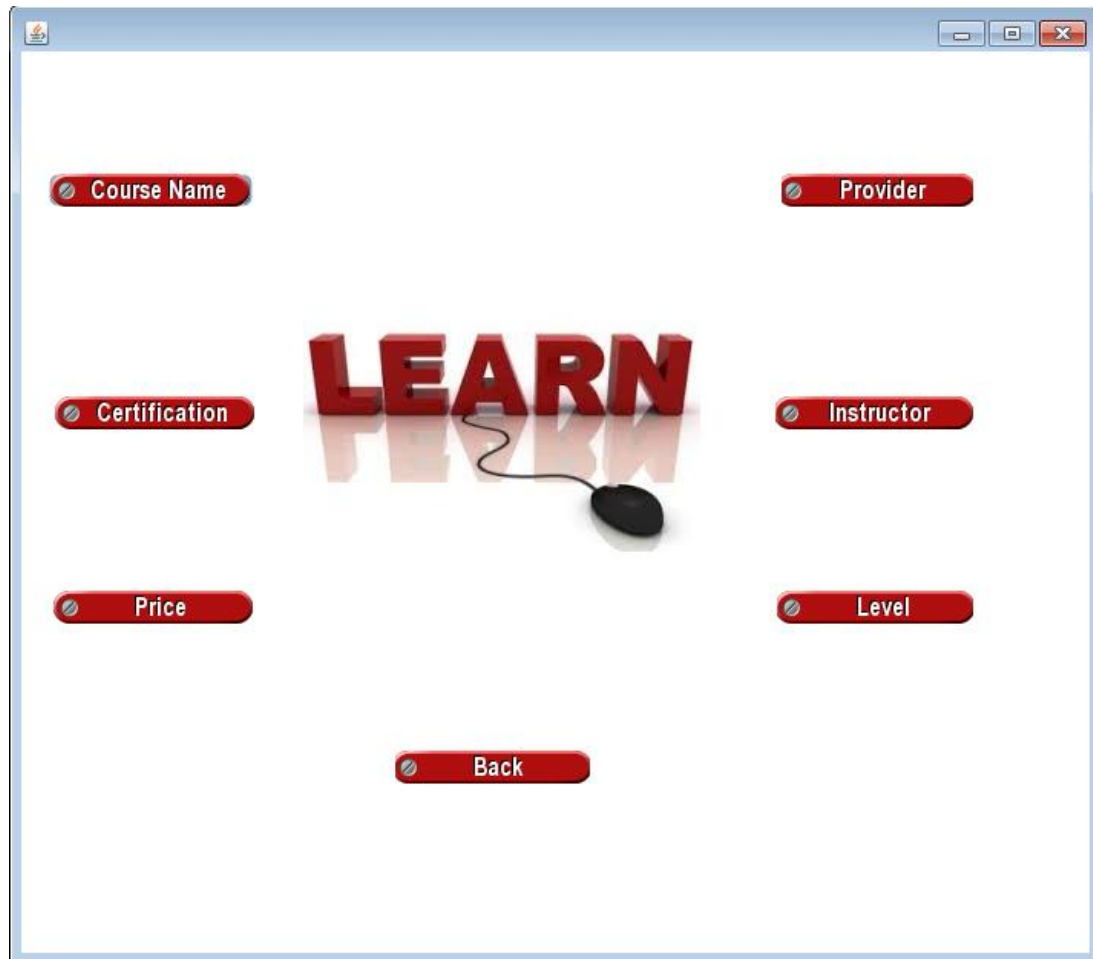
This appendix provides screenshots of various stages of the working of the complete system and the various user interface features. The system has the option for users to search for details about various online courses in terms of topic, provider, instructor, certification, price and level. The option to bookmark courses is also included and the entire system works offline with provision to update newest courses.

The application when started is shown below in Figure A.1 with the main menu showing various buttons – ‘Search’, ‘Update’, ‘Bookmarked Courses’ and ‘Clear All Bookmarks’.



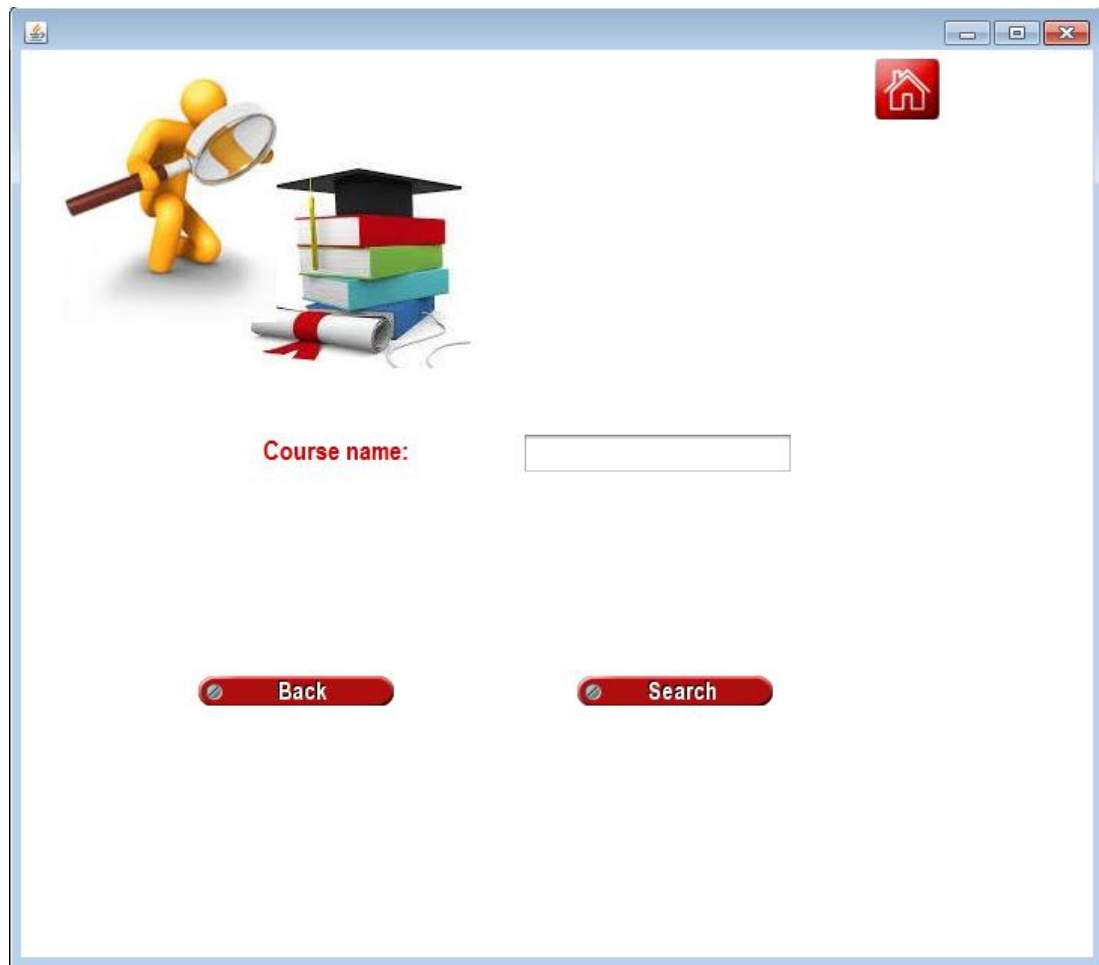
**Figure A.1: Main menu of the interface**

Figure A.2 shows the window of the application when ‘Search’ button is clicked on the main menu. This window has different functionalities like searching for courses based on course name, course provider, certification, instructor, price and level.



**Figure A.2: Search menu of the interface**

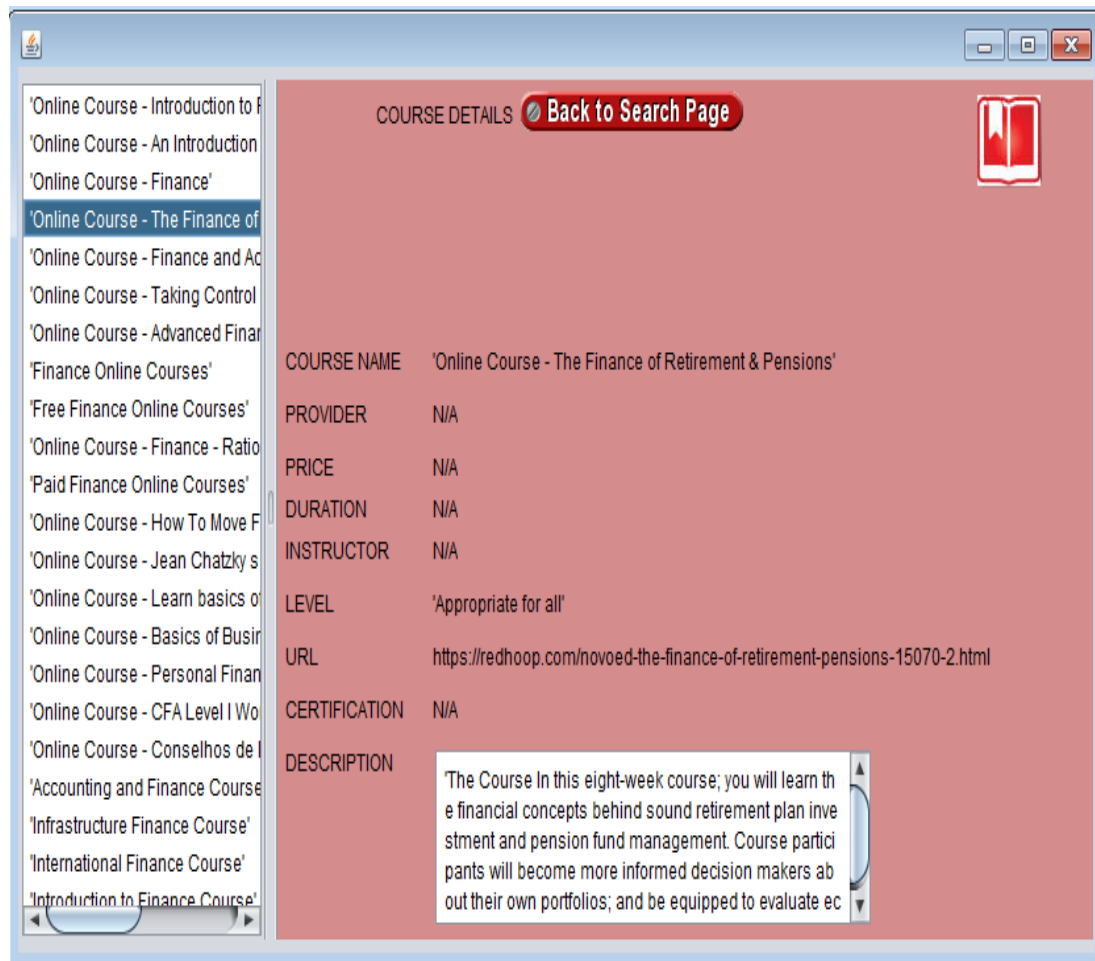
Figure A.3 shows the interface of the application for search by ‘Course name’ filter which provides a text field for the user to enter the name of the online course and a ‘Search’ and ‘Back’ button to find the courses and navigate back to the main menu respectively.



The image shows a web application window with a light blue border. In the top right corner, there are standard window control buttons (minimize, maximize, close) and a red home icon. The main content area features a 3D illustration of a yellow stick figure holding a magnifying glass over a stack of four books (red, green, blue, and white) with a black graduation cap on top. Below this illustration, the text 'Course name:' is displayed in red, followed by a white text input field. At the bottom of the window, there are two red buttons with white text: 'Back' on the left and 'Search' on the right.

**Figure A.3: Search by course**

Figure A.4 shows the interface of the application containing the search results after ‘Search’ button is clicked (based on any of the filters). The left pane shows the list of courses and when any of the courses is clicked, the description of the course is shown on the right pane of the window. It has ‘Back to Search Page’ button at the top of the window for navigation purpose and also a ‘Bookmark’ button on the top right corner used to bookmark the courses as a list of favourites.



**Figure A.4: Results**

Figure A.5 shows the interface for search by ‘Certificate’ filter, which searches for the courses that provide certification at the end of the course. It has two options – ‘Yes’ and ‘No’. It also has ‘Back’ and ‘Home’ buttons for navigation purposes.



**Figure A.5: Search by certificate**

Figure A.6 shows the interface for search by 'Price' filter. It has two options – 'Free' and 'Paid' which provides free courses and paid courses respectively. It also has 'Back' and 'Home' buttons for navigation purposes.



**Figure A.6: Search by price**

Figure A.7 shows the interface for search by ‘Provider name’ filter, which provides a textbox for the user to enter the name of the course provider as per their interests. It also has ‘Back’ and ‘Home’ buttons for navigation purposes.



**Figure A.7: Search by provider name**



Figure A.8 shows the interface for search by ‘Instructor’ filter, which allows the user to search for the list of courses taught by a specific instructor. It provides a text field for the user to enter the name of the instructor. It also has ‘Back’ and ‘Home’ buttons for navigation purposes.



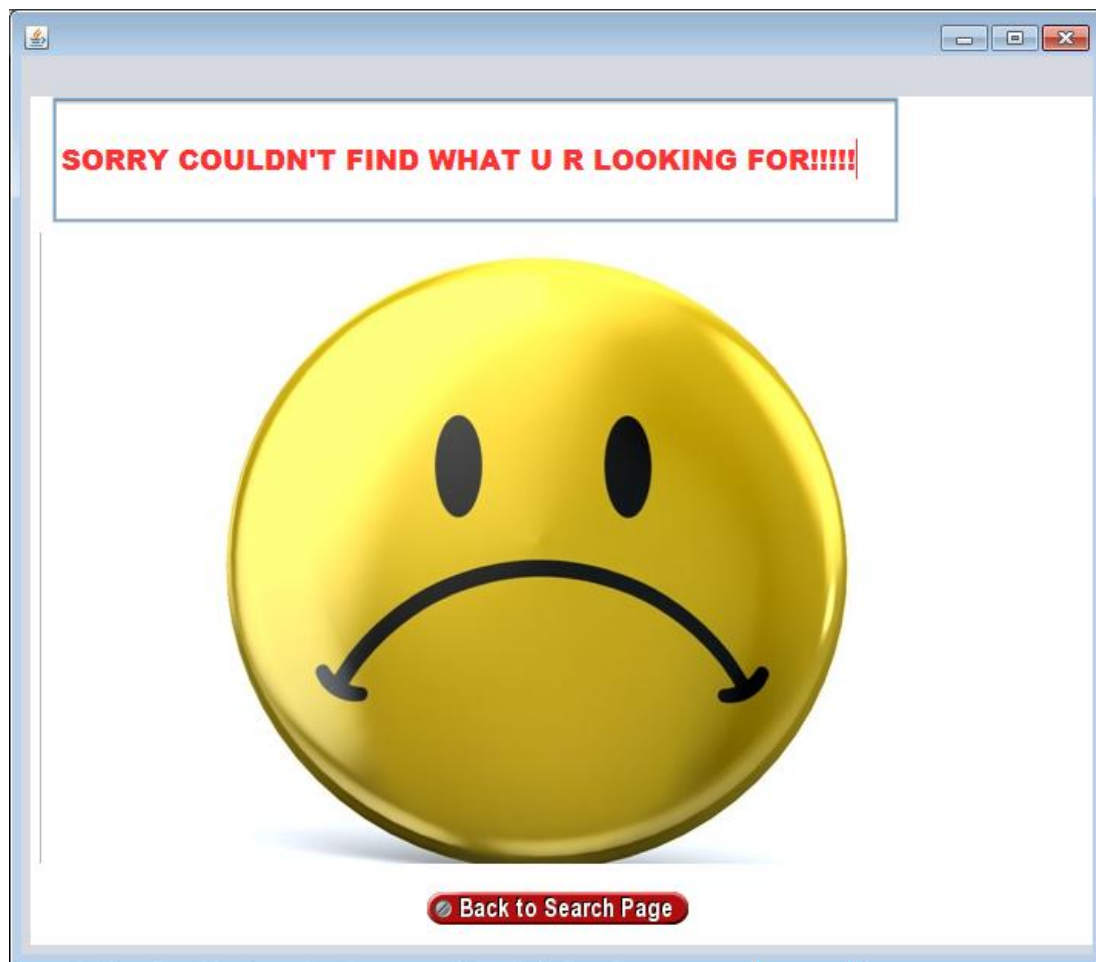
**Figure A.8: Search by instructor**

Figure A.9 shows the interface for search by ‘Level’ filter, which provides the user with options – ‘Beginner’, ‘Intermediate’ and ‘Advanced’ levels of courses. It also has ‘Back’ and ‘Home’ buttons for navigation purposes.



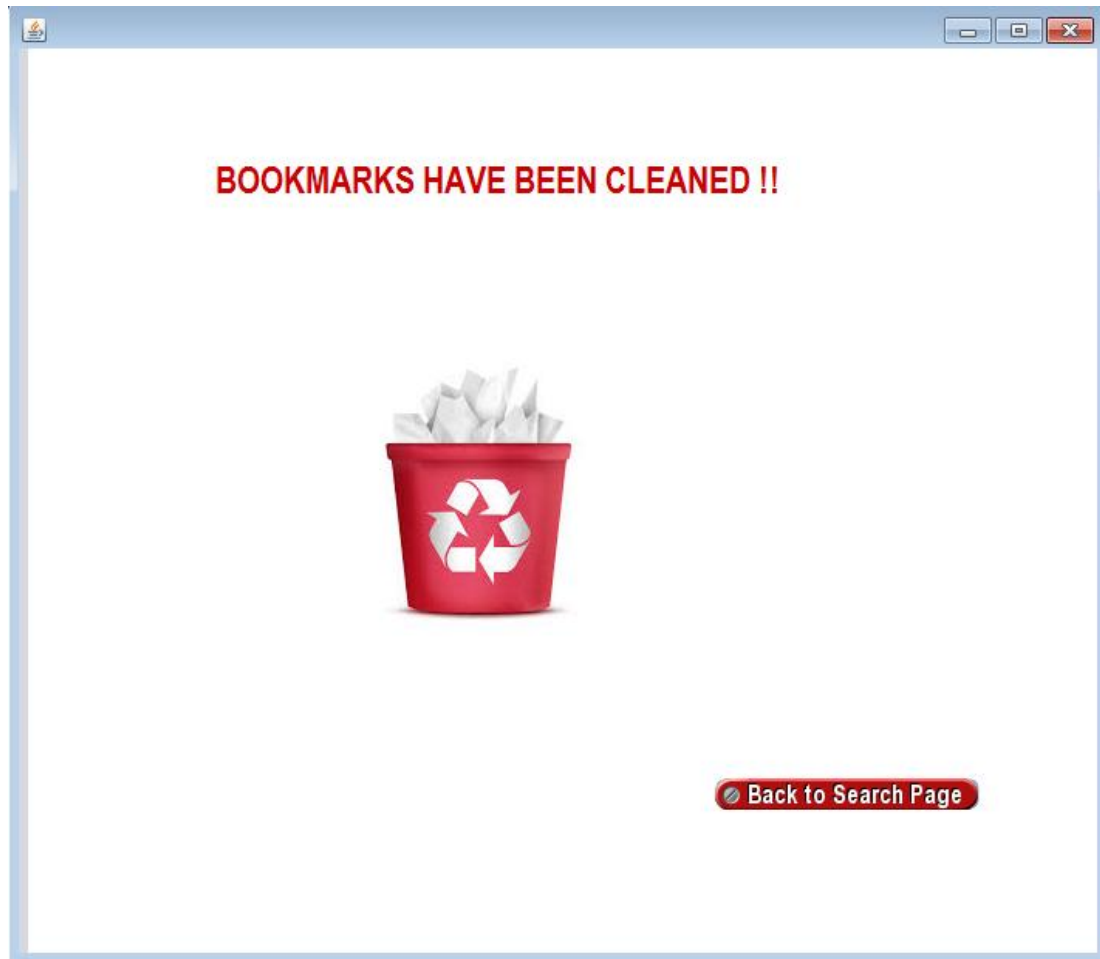
**Figure A.9: Search by level**

Figure A.10 shows the interface of the application when invalid searches are performed like usage of numbers/special characters in the text field of the search option or if the course searched for doesn't exist. It also has 'Back to Search page' button for navigation purpose.



**Figure A.10: Sorry page**

Figure A.11 shows the interface when ‘Clear All Bookmarks’ button from the main menu is clicked. It erases all the bookmarks saved previously. It also has ‘Back to Search page’ button for navigation purpose.



**Figure A.11: Clearing of bookmarks**

## Appendix B

### List of course providers covered in the application

This appendix lists the various providers covered for various online courses, gathered from crawling the web. The providers are listed in descending order of the number of courses offered by the provider. Each of these providers is gathered from various renowned sources and cover multiple countries, departments and lecture formats. These are listed below:

1. Massachusetts Institute of Technology
2. University of Michigan
3. Coursera
4. Udemy
5. Edx
6. Khan Academy
7. Open2Study
8. Udacity
9. Learnable
10. PluralSight
11. Treehouse
12. Tuts+
13. CreativeLive
14. CreatorUp
15. Lynda
16. SkillShare
17. TrainSimple
18. Craftsy
19. GeneralAssembly
20. CreativeBug
21. Columbia University
22. Harvard Hillsborough
23. Community College
24. IISc Bangalore
25. IIT Bombay
26. IIT Delhi

27. IIT Guwahati
28. IIT Kanpur
29. IIT Kharagpur
30. IIT Madras
31. IIT Roorkee
32. Lancaster University
33. London School of Business and Finance
34. McGill University
35. New Jersey Institute of Technology
36. North Carolina State University
37. OpenMichigan
38. Portland Community College
39. Princeton University Press
40. Stanford
41. Stony Brook University
42. The University of New South Wales
43. UC Berkley
44. UC San Diego
45. UCLA
46. University of California
47. University of Canterbury
48. University of Houston
49. University of Illinois
50. University of Michigan
51. University of Missouri Kansas City
52. University of Oregon
53. University of Sulaymayyah
54. University of Toronto
55. University of Washington
56. Virtual University
57. Yale University
58. University of Minnesota
59. University of Pennsylvania

60. University of Florida