

Web User Profiling

Anjana Ashokkumar

Tennessee Technological university

Cookeville Tennessee USA

aashokkum42@tntech.edu

Abstract

In this report the problem of web user profiling is explained. In the search engines we use in our day to day lives, browsing is an important part. We find various resources through the web. But this is made possible by search through keywords. To improve this search we analyse large amounts of information about the web user through web searches. Web models are created for the users so as to improve the efficiency of searches, since using these models the most likely searches for a particular user could be found.

Objectives

The aim of this project is to develop a system that helps us develop an intelligent web browser. The project will focus on the use of decision tree learning to create models of web users. You will be provided with decision tree learning tools and will collect data from web searches. You will then experiment with creating web user models and use these models in order to improve the efficiency of web searches performed by the same or new users. The learning objectives of the project are to:

- Learn the basics of information retrieval and machine learning
- Gain experience with recent software applications in these areas, and
- Gain better understanding of fundamental AI concepts such as knowledge representation and search.

Problem Description

The entire project is divided into three phases, namely the data collection, feature extraction and machine learning. At the data collection stage, 100 web documents are collected and labelled according to the user preference as like and dislike. Then they are represented as feature vectors. In the machine learning stage various algorithms are applied and a user model is created. This model is created to improve the efficiency of web search by the same user or it helps in finding out the interests, or the likes and dislikes of the new user, using the records or data analysis which was done for previous user models.

Phase – 1

This is the data collection phase. Here 100 web documents are collected manually. (Documents obtained from the Web Crawler were considered as examples, or instances for using the Web Crawler) These documents are then concatenated into a single text file. Three attributes, the name of the document, the content in the document and then the attribute that shows the like and dislike of the user i.e. the document class. They can also be done with the help of a Web Crawler. Various parameters could be changed according to the search and the web documents could be obtained. The BFS and DFS search algorithms are used to find the web documents.

ANSWERS TO THE QUESTIONS FROM THE GIVEN PDF.

1)The Web Crawler is a computer program that searches the search engines with the help of certain keywords, or URL. It crawls through the web to find pages and these pages could be saved. Here the web crawler used is the WebSPHINX. Various parameters such as the different kind of search, or the page size, the number of threads, search time, timeout and the destination for the search to take place such as the web, server and subtree could be changed according to the result that needs to be obtained.

Here the search used was the breadth first search and the depth of the breadth first search was 50. The URL used here was <http://www.google.com>. As a result of this using the breadth first search the web documents were obtained and stored in the system.

2) The **Web** is a collection of web sites or web pages stored in the server connected through the internet. The **Tree** is a hierarchical data structure. It consists of nodes connected by edges. It has parent node and root nodes. A **Directed Acyclic Graph** is the one in which the nodes are connected using arrows pointed towards a particular direction and the edges connected to the nodes do not form a cycle. A **Directed Graph** is one where the nodes are connected using arrows pointed towards a direction but, however they could form a cycle. A **Graph** is the representation of connected values in multi-dimensional space. Each object in a graph is called a node and they are connected using the edges.

3)The nodes are represented by the document symbol which indicated the web documents that have been crawled through i.e. the pages that have been visited.

4)The edges are represented by arrows, that is the hyperlinks to connect to the web documents from the other web document. Hence this helps in visiting a series of web pages or documents and they could be extracted.

5)The two search algorithms used by the web crawler WebSPHINX are

- Depth First Search and
- Breadth First Search

The BFS and DFS are used since BFS traverses through the root node to all the connected nodes from the root and keeps on traversing until it finds an objective, then the process

terminates, whereas the DFS traverses through the entire graph visiting all the nodes, until it finds a reason for the process to be terminated.

6) In most of the cases the web crawler would not get into a loop. And here the Breadth First Search and Depth First Search are used. In these algorithms, the visited nodes get marked. Hence the WebSPHINX does not get into a loop.

7) The search into the subtree the web and server widens the chance of visiting more number of nodes. They expand into the wider parts of the web respectively.

8) Using multi threads in a web crawler helps you to find all the web pages of a particular web document. They get linked to all the web pages of a website. And there are acyclic.

9) When the page size and timeout limits are reached, the crawling process is terminated. Since no more web pages of that size could be found.

Phase – 2

In this phase the feature extraction takes place. Here the text document containing the concatenated text is converted into ARFF format so that it could be loaded into Weka Explorer. The text file is changed into ARFF format using the headers

```
@relation text_corpus_1_string
@attribute document_name string
@attribute document_content string
@attribute document_class {+,-}
```

```
@data
```

and then the data contains the name of the document separated by a comma and the content is enclosed into the quotation marks and at last the like and dislike label is mentioned. Here “like” and “dislike” is represented using “yes” and “no” respectively. Then this file is saved using the extension .arff. Hence the text file is now converted into arff file format. It consists of three attributes namely the document name, the document content and the document class. Now the ARFF file is loaded into Weka Explorer for pre-processing.

Weka is a software which consists of all the machine learning algorithm tools, it helps in data pre-processing, analysis, regression, classification and output visualisation.

After loading the file into the Weka explorer and after following a few steps the “numeric”, “Boolean” and “TFIDF” formats of the same file could be obtained. Hence we got our datasets in three different formats. Later anyone of these files could be used as the training dataset.

Below are the images of the datasets obtained as a result of the pre-processing. The three different dataset formats. Numeric, Boolean and TFIDF.

The table above is the “count” or “numeric” table. This table represents values, which show how many times a word is occurring in a particular document.

Viewer

Relation: technology-weka.filters.unsupervised.attribute.StringToWordVector-R1,2-W1000-prune-rate-1.0-I-N0-stemmerweka.core.stemmers.NullStemmer-stopwords-handlerweka.core.stopwords.Null-M1-to...

No.	1: document_class Nominal	2: 0 Numeric	3: 1 Numeric	4: 10 Numeric	5: 1957 Numeric	6: 1964 Numeric	7: 1970s Numeric	8: 1973 Numeric	9: 1977 Numeric	10: 1980 Numeric	11: 1980s Numeric	12: 1983 Numeric	13: 1990s Numeric	14: 1994 Numeric	15: 1997 Numeric	16: 1998 Numeric	17: 2 Numeric	18: 2007 Numeric	19: Num
1	y	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	y	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	y	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	y	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
5	y	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
6	y	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
7	n	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
8	n	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
9	y	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
10	y	4.60517	3.912023	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
11	n	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
12	n	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
13	y	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
14	y	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
15	n	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
16	y	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
17	y	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.60517	0.0
18	y	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
19	y	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
20	n	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
21	n	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
22	y	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
23	n	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
24	y	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
25	n	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
26	y	0.0	0.0	0.0	0.0	0.0	0.0	4.60517	4.60517	0.0	0.0	0.0	0.0	0.0	3.912023	0.0	0.0	0.0	0.0
27	y	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
28	y	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
29	y	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.6

This dataset indicates the TF-IDF format. That is Term Frequency- Inverse Document Frequency. This dataset shows values for words which are more relevant in a document. This could be found using the number of occurrences of a word in a particular document.

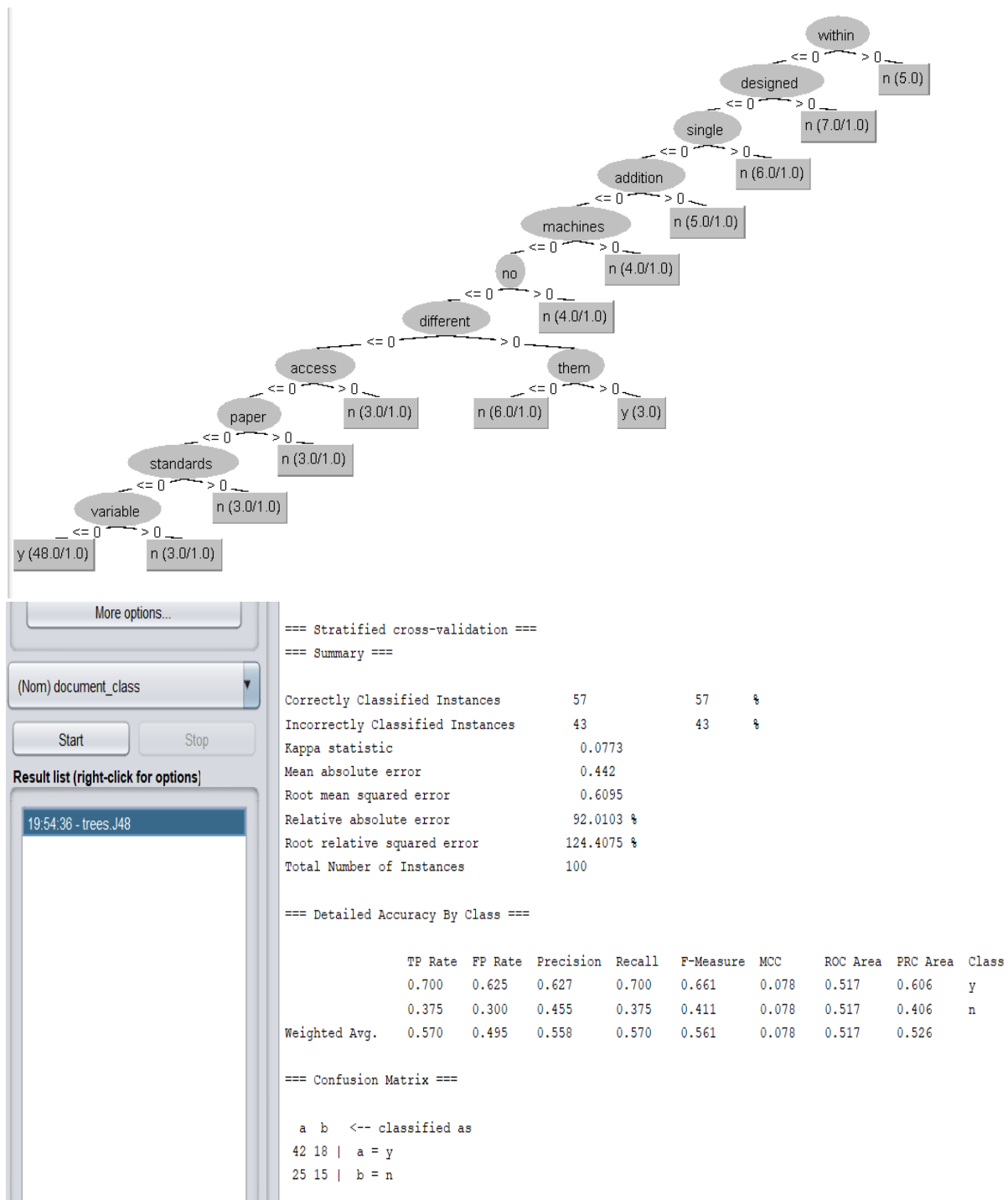
So now we have the relevant datasets built for the machine learning step.

Phase – 3

This phase is the machine learning phase. Here the dataset is run using a decision tree algorithm and the documents which the user dislike can be discarded to make it easy.

The Decision tree algorithm and the K-Nearest Neighbour algorithm are used to run the dataset and the accuracy results are obtained. The results are obtained in the form of confusion matrices as well.

The result obtained from the J48 is shown below.



Hence these snapshots show the results of the decision tree.

ANSWERS TO QUESTIONS IN THE PDF

1)The most important terms are variable, designed, addition and machines.

2)**Precision** is given by the ratio of the positively obtained observations to the correct and incorrect results of the observations.

Recall is given by the ratio of the positively obtained observations to the total positives obtained from the observation.

3)The initial state is the word “different”

4)The state transitions can be represented using 0 and 1. They help us represent the path of the nodes of the tree. When an edge connects one node and the other it is represented using 1 or if there is no path then it is represented using 0.

5)The final state is the word “within” from the graph

6)We use informed search algorithm here since this helps in searching for a particular word according to the user's preference

7)The evaluation function here is 57% since it is the correctly classified instance.

8)Here tree pruning means that sections of tree are removed which are not required for the user.

Here now the dataset is run using the K-Nearest Neighbour and the result is shown below.

The screenshot shows the Weka GUI with the 'Cross-validation' method selected and 10 folds. The results panel on the right displays the following information:

IB1 instance-based classifier
using 1 nearest neighbour(s) for classification

Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances	43	43	%
Incorrectly Classified Instances	57	57	%
Kappa statistic	0.0138		
Mean absolute error	0.5685		
Root mean squared error	0.7468		
Relative absolute error	118.3258 %		
Root relative squared error	152.4407 %		
Total Number of Instances	100		

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.117	0.100	0.636	0.117	0.197	0.026	0.508	0.604	y
	0.900	0.883	0.404	0.900	0.558	0.026	0.508	0.404	n
Weighted Avg.	0.430	0.413	0.544	0.430	0.342	0.026	0.508	0.524	

=== Confusion Matrix ===

```

a b  <-- classified as
7 53 | a = y
4 36 | b = n

```

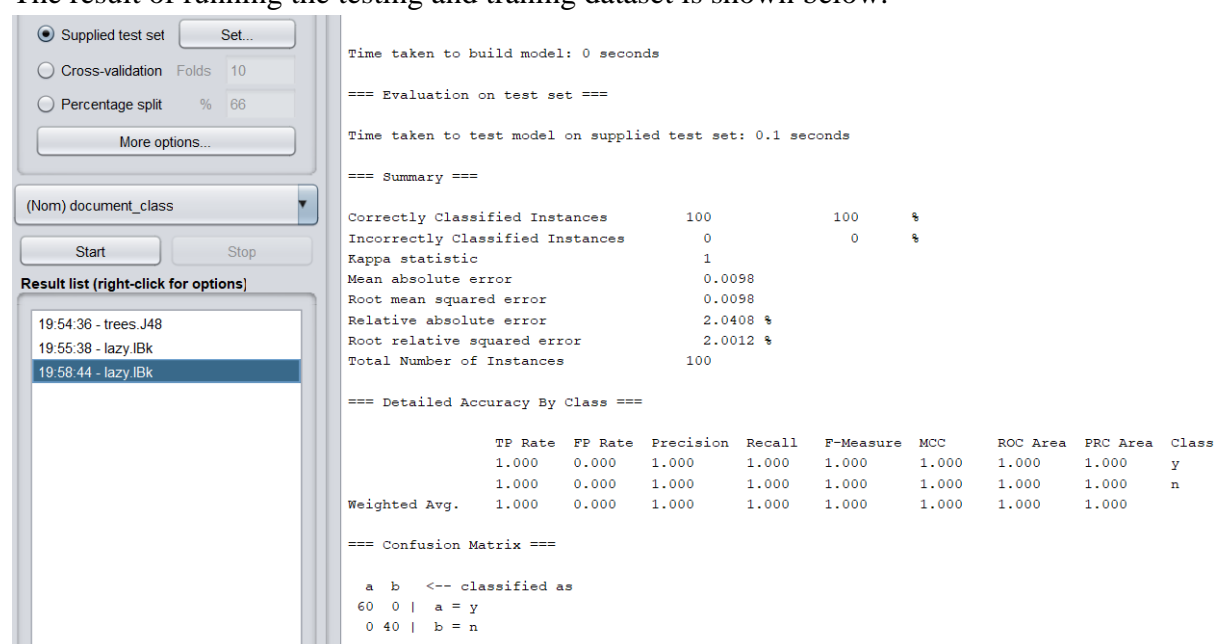
This is the result of the K-Nearest neighbour algorithm.

Answer to the Question

The results obtained from the decision tree is better. Since the decision tree traverses through the entire tree whereas in KNN it searches using the distance between the data.

The training dataset is taken and it is used as the testing set as well, with only a few changes. This dataset could be split into 66% training set and 34% testing set. The data showing the likeliness are retained and the rest of the data could be discarded for easy searching

The result of running the testing and training dataset is shown below.



The screenshot displays a machine learning software interface. On the left, there are settings for the test set (Supplied test set, Cross-validation, Percentage split) and a dropdown menu for the document class. The main area shows the evaluation results for a decision tree model. The results include a summary of performance metrics, a detailed accuracy by class table, and a confusion matrix.

Time taken to build model: 0 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.1 seconds

=== Summary ===

Correctly Classified Instances	100	100	%
Incorrectly Classified Instances	0	0	%
Kappa statistic	1		
Mean absolute error	0.0098		
Root mean squared error	0.0098		
Relative absolute error	2.0408	%	
Root relative squared error	2.0012	%	
Total Number of Instances	100		

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	y
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	n
Weighted Avg.	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	

=== Confusion Matrix ===

```
a b <-- classified as
60 0 | a = y
0 40 | b = n
```

Now the user models are created using the numeric dataset. Five user models are created based on the likes and dislikes of the user. And a test dataset is created. These models and test dataset are built using the original processed numeric ARFF dataset.

Each of these user models are run along with the test dataset. And the accuracy results are obtained. Each user model has different instance rate. The results of the five users are shown below:

Correctly Classified Instances 64 64 %
Incorrectly Classified Instances 36 36 %
Kappa statistic 0.2757
Mean absolute error 0.3627
Root mean squared error 0.5942
Relative absolute error 72.9783 %
Root relative squared error 118.962 %
Total Number of Instances 100

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.679	0.404	0.655	0.679	0.667	0.276	0.637	0.615	y
	0.596	0.321	0.622	0.596	0.609	0.276	0.637	0.561	n
Weighted Avg.	0.640	0.365	0.639	0.640	0.639	0.276	0.637	0.589	

=== Confusion Matrix ===

a b <-- classified as
36 17 | a = y
19 28 | b = n

User 1

Correctly Classified Instances 58 58 %
Incorrectly Classified Instances 42 42 %
Kappa statistic 0.1549
Mean absolute error 0.4216
Root mean squared error 0.6418
Relative absolute error 84.8126 %
Root relative squared error 128.4912 %
Total Number of Instances 100

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.623	0.468	0.600	0.623	0.611	0.155	0.577	0.574	y
	0.532	0.377	0.556	0.532	0.543	0.155	0.577	0.516	n
Weighted Avg.	0.580	0.425	0.579	0.580	0.579	0.155	0.577	0.546	

=== Confusion Matrix ===

a b <-- classified as
33 20 | a = y
22 25 | b = n

User 2

=== Summary ===

Correctly Classified Instances	60	60	%
Incorrectly Classified Instances	40	40	%
Kappa statistic	0.1913		
Mean absolute error	0.402		
Root mean squared error	0.6263		
Relative absolute error	81.2525 %		
Root relative squared error	124.6407 %		
Total Number of Instances	100		

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.679	0.489	0.610	0.679	0.643	0.193	0.595	0.584	y
	0.511	0.321	0.585	0.511	0.545	0.193	0.595	0.529	n
Weighted Avg.	0.600	0.410	0.599	0.600	0.597	0.193	0.595	0.558	

=== Confusion Matrix ===

a b <-- classified as
36 17 | a = y
23 24 | b = n

User 3

Correctly Classified Instances	63	63	%
Incorrectly Classified Instances	37	37	%
Kappa statistic	0.2546		
Mean absolute error	0.3725		
Root mean squared error	0.6024		
Relative absolute error	75.0395 %		
Root relative squared error	120.4893 %		
Total Number of Instances	100		

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.679	0.426	0.643	0.679	0.661	0.255	0.627	0.607	y
	0.574	0.321	0.614	0.574	0.593	0.255	0.627	0.553	n
Weighted Avg.	0.630	0.376	0.629	0.630	0.629	0.255	0.627	0.581	

=== Confusion Matrix ===

a b <-- classified as
36 17 | a = y
20 27 | b = n

User 4

=== Summary ===

Correctly Classified Instances	63	63	%
Incorrectly Classified Instances	37	37	%
Kappa statistic	0.2546		
Mean absolute error	0.3725		
Root mean squared error	0.6024		
Relative absolute error	75.0395 %		
Root relative squared error	120.4893 %		
Total Number of Instances	100		

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.679	0.426	0.643	0.679	0.661	0.255	0.627	0.607	y
	0.574	0.321	0.614	0.574	0.593	0.255	0.627	0.553	n
Weighted Avg.	0.630	0.376	0.629	0.630	0.629	0.255	0.627	0.581	

=== Confusion Matrix ===

a b <-- classified as
36 17 | a = y
20 27 | b = n

User 5

Hence these results obtained from the models show better results and the users could be classified according to the user preferences, so that it becomes easy for the searches to take place.

Conclusion

Hence this project could be used to find out the user preference for the old user and also the new user using the old user preference. This enhances the searching techniques and makes it easy for the user. This could give us well-structured web, to search for the web documents.