WEB MINING THROUGH CUSTOMIZED MARKOV ALGORITHM

Mr.Sourabh Mayria
[ME(SE), INDIA]

Mr.Mahesh Malviya
Assi. Prof &Head CS Dept, JIT, Borawan,India

(1)ABSTRACT

The problem of predicting a user's behavior on a web site has gained Importance due to the very increases required web network. It also compulsory for market and research of social behavior. Although many approaches are comes and many will should come, however they all are important at their different different view.

Even I (Author, himself), suggests a physical approach to measure web traffic, results in paper titled "The Web Page Prediction by TRP Machine" , however I want to proposed an algorithm for it, which is not physically but logically measures the web traffic at all.

Results as "Customized Page Rank algorithm" from simple Page Rank algorithm. Because simple Buddy Prima algorithm gives results after complicated calculations, instead it the novel approach creates less calculations and more clear and precise results as we want.

The "Customized Buddy Prima algorithm" is based on the Markov Model with its chain in Forward reasoning [Research Paper on Forward Reasoning in Hierarchical representation , by me, (Author himself)], with the help of Apriori algorithm, Division algorithm, Association rule, page rank algorithm, Eigen vector calculations, modified Adjacency matrix, and many more calculations as tools.

The suggested method is very proper as the result point of view, as it gives more proper results as shown by comparison of both two approaches as traditional one and the modified as well.

Keywords Web Page Prediction, Buddy Prima algorithm, Customized Buddy Prima algorithm, Markov Model, Apriori algorithm, Division algorithm, Association rule, page rank algorithm, Eigen vector calculations, modified Adjacency matrix.

(2)INTRODUCTION

The problem of predicting a user's behavior on a web-site has gained importance due to the rapid growth of the world-wide-web and the need to personalize and influence a user's browsing experience. Markov models and their variations have been found well suited for addressing this problem. In recent years, the problem of modeling and predicting a user's surfing behavior on a web-site has attracted a lot of research interest as it can be used to improve the web cache performance, recommend related pages , improve search engines , understand and influence buying patterns , and personalize the browsing experience. Modeling user web navigation data is a challenging task that is continuing to gain importance as the size of the web and its user-base increases. Data characterizing web navigation can be collected from the server or client- based log files, enabling the reconstruction of user navigation sessions.

Basically the web mining is the task related to the database to collect the information related to the web page access in the form of web session for the future analysis. The web mining enables the prediction of page access and the analysis of user navigation behavior.

Here our task is related to the web usage mining which basically Consist task related to the use of web where the access of the web will considered and the navigation pattern and the prediction operation will performed in the mining of this kind we will use the database in the form of the web log files and we will generate the results on the basis of the database given.
Markov models have been used for studying and understanding stochastic processes, and were shown to be well-suited for modeling and predicting a user's browsing behavior on a web-site. In general, the input for these problems is the sequence of web-pages that were accessed by a user and the goal is to build Markov models that can be used to model and predict the web-page that the user will most likely access next. In many applications, first-order Markov models are not very accurate in predicting the user's browsing behavior, since these models do not look far into the past to correctly discriminate the different observed patterns. As a result, higher-order models are often used. Unfortunately, these higher-order models have a number of limitations associated with high state-space complexity, reduced coverage, and sometimes even worse prediction accuracy. One method proposed to overcome the problem is the clustering and cloning to duplicate the state corresponding to page that require a longer history to understand the choice of link that users made. Initially when the web log is not available means the web site is newly launched the prediction or the navigation decision will mad on the page rank our page rank strategy will also used to resolve the ambiguity of the model.

Our model will use the basic strategy for the preparing the model is the page rank , and variable length markov model, the problem of ambiguity in the markov model will solve on the basis of the page rank and the page rank will also used in the initial stage when the web log file is not available.

(3)Markov Model

As discussed in the introduction, techniques derived from Markov models have been extensively used for predicting the action a user will take next given the sequence of actions he or she has already performed. For this type of problems, Markov models are represented by three parameters < A; S; T >, where A is the set of all possible actions that can be performed by the user; S is the set of all possible states for which the Markov model is built; and T is a |S|*|A| Transition Probability Matrix (TPM), where each entry t_{ij} corresponds to the probability of performing the action j when the process is in state i . The state-space of the Markov model depends on the number of previous actions used in predicting the next action. The simplest Markov model predicts the next action by only looking at the last action performed by the user. In this model, also known as the first-order Markov model, each action that can be performed by a user corresponds to a state in the model. A somewhat more complicated model computes the predictions by looking at the last two actions performed by the user. This is called the second-order Markov model, and its states correspond to all possible pairs of actions that can be performed in sequence. This approach is generalized to the Kth -order Markov model, which computes the predictions by looking at the last K actions performed by the user, leading to a state-space that contains all possible sequences of K actions. For example, consider the problem of predicting the next page accessed by a user on a web site. The input data for building Markov models consists of web-sessions, where each session consists of the sequence of the pages accessed by the user during his/her visit to the site. In this problem, the actions for the Markov model correspond to the different pages in the web site, and the states correspond to all consecutive pages of length K that were observed in the different sessions. In the case of first-order models, the states will correspond to single pages, in the case of second-order models, the states will correspond to all pairs of consecutive pages, and so on.

Page	Rank
1.0	0.218
2.0	0.145
3.0	0.165
4.0	0.14
5.0	0.165
6.0	0.105
7.0	0.156

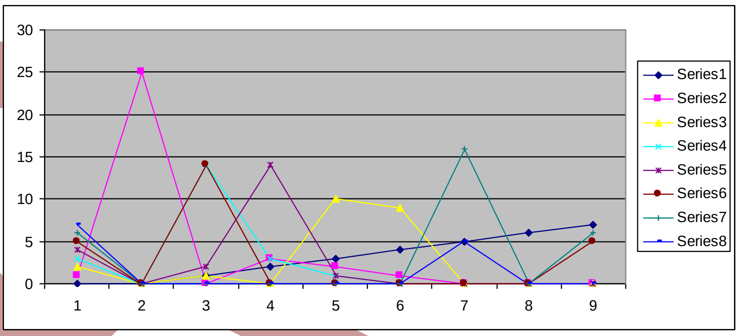
Once the states of the Markov model have been identified, the transition probability matrix can then be computed. There are many ways in which the TPM can be built.. For example consider the web-session P3; P5; P2; P1; P4shown in Figure 1. If we are using first-order Markov model then each state is made up of a single page, so the first page P3 corresponds to the state s3. Since page p5 follows the state s3 the entry t35 in the TPM will be updated. Similarly, the next state will be s5 and the entry t52 will be updated in the TPM. In the case of higher-order model each state will be made up of more than one actions, so for a second-order model the first state for the web-session WS2 consists of pages P3, P5 and since the page P2 follows the state P3 P5 in the web-session the TPM entry corresponding to the state P3, P5 and page P2 will be updated. Once the transition probability matrix is built making prediction for web sessions is straight forward. For example, consider a user that has accessed pages P1; P5; P4. If we want to predict the page that will be accessed by the user next, using a first-order model, we will first identify the state s4 that is associated with page P4 and look up the TPM to find the page pi that has the highest probability and predict it. In the case of our example the prediction would be page P5, P6 and suppose P7.

The random web log for the data as:

```
1 2 3 3 4 1 5 6 7
1 5 6 7 6
1 3 2 4 2 4
1 3 2 1 5 7 6
1 5 6 7 5 6
1 3 2 4 2 4
1 5 6 7 5 6
1 3 2 4 2 4
1 4 2 3 1 5 6 5 7
1 5 6 7 6
1 3 2 4 2 4
1 3 2 1 5 7 6
1 5 6 7 5 6
1 3 2 4 2 4
1 5 6 7 5 6
1 3 2 4 2 4
1 4 2 3 1 5 6 5 7
1 5 6 7 6
1 3 2 4 2 4
1 3 2 1 5 7 6
1 5 6 7 5 6
1 3
1 3
1 3
1 3
```

The analysis of that related page values gives:

That can show as:



This is the **Limitations of Markov model**

In many applications, first-order Markov models are not successful in predicting the next action to be taken by the user. This is because these models do not look far into the past to correctly discriminate the different behavioral modes of the markov model to handle the situation again we will use the page rank parameter to correctly predict the web page access.

The traditional approach says that,

(4)Formula to find the Page Rank

It may look daunting to non-mathematicians, but the Page Rank algorithm is in fact elegantly simple and is calculated as follows:

$$PR(A) = (1-d) + d\,(PR(B)/C(B) + \ldots + PR(T_n)/C(T_n))$$

Where PR(A) is the Page Rank of a page A

PR(B) is the Page Rank of a page B

C(B) is the number of outgoing links from the page B

d is a damping factor in the range $0 < d < 1$, usually set to 0.85.

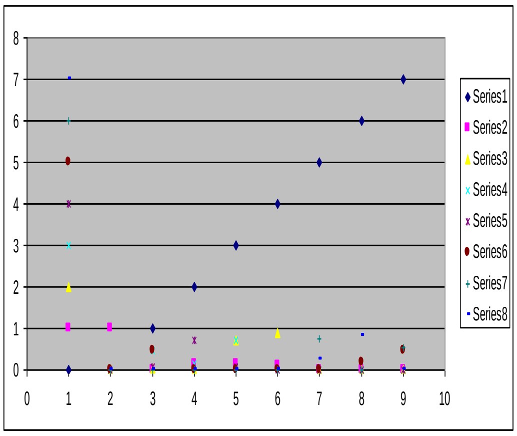
The page rank of a web page is therefore calculated as a sum of the page ranks of all pages linking to it i.e. incoming link, divided by the number of links on each of those pages, i.e. outgoing links.

Page/path	1	2	3	4	5	6	7
start	25.0	0.0	0.0	0.0	0.0	0.0	0.0
1.0	0.0	1.0	14.0	2.0	14.0	0.0	0.0
2.0	3.0	0.0	3.0	14.0	0.0	0.0	0.0
3.0	2.0	10.0	1.0	1.0	0.0	0.0	0.0
4.0	1.0	9.0	0.0	0.0	0.0	0.0	0.0
5.0	0.0	0.0	0.0	0.0	0.0	16.0	5.0
6.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
7.0	0.0	0.0	0.0	0.0	5.0	6.0	0.0

The above discussion gives results as:

Page/P.	1	2	3	4	5	6	7
Start	1.0	0.0	0.0	0.0	0.0	0.0	0.0
1.0	0.0	0.0322	0.4516	0.0645	0.4516	0.0	0.0
2.0	0.15	0.0	0.15	0.7	0.0	0.0	0.0
3.0	0.1428	0.7142	0.714	0.0714	0.0	0.0	0.0
4.0	0.1	0.9	0.0	0.0	0.0	0.0	0.0
5.0	0.0	0.0	0.0	0.0	0.0	0.7619	0.2380
6.0	0.0	0.0	0.0	0.0	0.1818	0.0	0.8181
7.0	0.0	0.0	0.0	0.0	0.4545	0.5454	0.0

That gives as:



(5)New Approach says that

$$PR(A) = \frac{PR(B)}{2} + \frac{PR(C)}{1} + \frac{PR(D)}{3}.$$

In the general case, the Page Rank value for any page **u** can be expressed as:

$$PR(u) = \sum_{v \in B_u} \frac{PR(v)}{L(v)},$$

And damping factor says that,

The Page Rank theory holds that even an imaginary surfer who is randomly clicking on links will eventually stop clicking. The probability, at any step, that the person will continue is a damping factor ***d***. various studies has tested different damping factors, but it is generally assumed that the damping factor will be set around precious. That is,

$$PR(A) = 1 - d + d\left(\frac{PR(B)}{L(B)} + \frac{PR(C)}{L(C)} + \frac{PR(D)}{L(D)} + \cdots\right)$$

Or (*N* = the number of documents in collection)

$$PR(A) = \frac{1-d}{N} + d\left(\frac{PR(B)}{L(B)} + \frac{PR(C)}{L(C)} + \frac{PR(D)}{L(D)} + \cdots\right).$$

So, the equation is as follows:

$$PR(p_i) = \frac{1-d}{N} + d \sum_{p_j \in M(p_i)} \frac{PR(p_j)}{L(p_j)}$$

where $p_1, p_2, \ldots, p_N$ are the pages under consideration, $M(p_i)$ is the set of pages that link to p_i, $L(p_j)$ is the number of outbound links on page p_j, and N is the total number of pages.

Algorithm for finding the page rank

1. PR(A) is the rank for the page A, L(A) is the number of out link to page A. d is the dumping Factor
2. For each page of web site find the following

   PR (A) = (1-d)/n+SUM (PR (O)/L (O))

Algorithm for finding the dynamic order model for

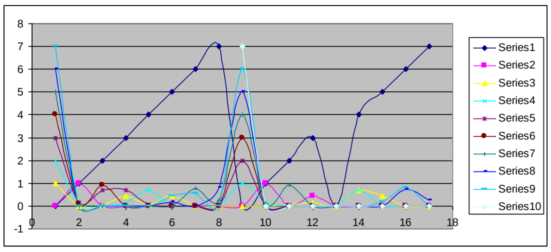
1. Wi- the Weight of state W,Wij is the Weight of link i to j,
   P is the transition probability.accuracy thrashold p=0.1.
   n number of states
2. Find the First Order transition Probability Matrix
3. Repeat until the accurate model
4. For each value of x=1 …. N
   For state x
   If |Pi,x,k-px,k|>0.1 then state x is not accurate make clone
   X and X'
   (To find the new link weight)
   For each in link y and y' and out link z,z'
   IF |P y,x,z-Py',x,z| < 0.1 then keep y, y' in same clone

```
        Update the weight
        W x z=W yxz+W y'xz;
5.  End.
```

That gives:

Pg/p.	1	2	3	3a	4	5	6	7
Strt	1.0	0.0	0.451	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.03	0.15	0.0	0.7	0.4516	0.0	0.0
1.0	0.15	0.0	0.0	0.0	0.7	0.0	0.0	0.0
2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4.0	0.1	0.9	0.0	0.0	0.0	0.0	0.0	0.0
5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.7	0.2
6.0	0.0	0.0	0.0	0.0	0.0	0.1818	0.8	0.0
7.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

(6)The comparison between two patterns gives the result as



So in conclusion we can say that there must be a considerable difference between both patterns, and the new approach gives some more proper results, as shown above.

References

(1) "Research paper on forward reasoning in hierarchical representation" by "Mr.Sourabh Mayria"(Author himself), IJERT, vol1, issue 6(Aug-2012)
(2) Jose Borges and Mark Levene, Generating Dynamic Higher-Order Markov model, University Collage London, 2002
(3) Jiawei Han, Michline Kamber, "Data mining concepts and techniques", Simon Fraser University
(4) www.Google.com