First Order Markov Models for Web Server Log Data

1. Markov Chains

Consider a system that can be in any of a finite number of states, and assume that it moves from state to state according to some prescribed probability law. The system, for example, could be the visitors to a web page and the states the web page categories visited (i.e., frontpage, news, sports...etc). Observing visitors over a long period of time would allow one to find the probability of visiting a particular web page given that the visitor was just at another page. For example, the probability that the visitor goes to frontpage given that s/he was in the sports page.

Let X_i denote the state of the system at time point i, and let the possible states be denoted by $S_1, S_2, ..., S_m$ for a finite integer m. We are interested not in the elapsed time between transitions from one state to another, but only in the states and the probabilities of going from one state to another, that is the transition probabilities.

We assume that

$$P(X_i = s_k \mid X_{i-1} = S_j) = p_{jk}$$

where p_{jk} is the transition probability from s_j to S_k; and this probability is independent of i. So the transition probabilities do not depend on the time points; they only depend on the states. The event $(X_i = S_k \mid X_{i-1} s_j)$ is assumed to be independent of the past history of the process. Such a process is called a Markov chain with stationary transition probabilities. The transition probabilities can conveniently be displayed in a matrix:

$$P = \begin{bmatrix} p_{11} & p_{12} & .... & p_{1m} \\ P_{21} & p_{22} & .... & p_{2m} \\ .. & ... & .... & .... \\ .. & ... & .... & .... \\ .. & ... & .... & .... \\ p_{m1} & p_{m2} & ...... & p_{mm} \end{bmatrix}$$

I am illustrating this with an example using the msnbc data set. If P is regular (i.e.P^n has all positive entries for some power of P, then the chain has a stationary or equilibrium distribution that gives the probabilities of its being in the respective states after many transitions have evolved. Suppose that limit exists and denote it by $\pi = (\pi_1, ..., \pi_m)$. Then it must satisfy the

$$\pi = \pi P$$

.

The matrix that we will see below is not regular because it has what is called an absorbing state, i.e, a state for which the row has a one and all zeros. Once the system is in that state, it can not leave it. But we assume that we can go from any other state to the absorbing state. A question of interest in this case, is the expected number of steps to absorption.

2. The Data

The original data comes from Internet Information Server (IIS) logs for msnbc.com and news-related portions of msn.com for the entire day of September 28, 1999 (Pacific Standard Time) (Heckerman, UCI KDD Archive). The reader will understand the description that follows better if s/he opens the msnbc.com web Site before reading. Keep in mind, though, that since 1999 the structure of the Site has changed slightly.

The representation of the processed server-log data is fairly abstract: (a) the server-log files have been converted into a set of sequences, and one sequence for each user session, (b) each sequence is represented as an ordered list of discrete symbols (numbers), and (c) each symbol represents one of 17 categories of web pages requested by the user. The 17 categories correspond to sets of Uniform Resource Locators (URLs) on the site. Each sequence in the dataset corresponds to page views of a user during that twenty-four hours period. Since there are 989818 users, there are 989818 sequences. Each symbol in a sequence corresponds to a user's request for a page. Requests are not recorded at the finest level of detail–that is, not at the level of URL, but rather, they are recorded at the levels of page category (as determined by a site administrator). The categories (and their corresponding symbols) are 1="frontpage", 2="news", 3="tech", 4="local", 5="opinion", 6="on-air", 7="misc", 8="weather", 9="health", 10="living", 11="business", 12="sports", 13="summary", 14="bbs (bulletin board service)", 15="travel", 16="msn-news", and 17="msn-sports". The number of URLs per category ranges from 10 to 5000. Page requests served via a caching mechanism were not recorded in the server logs, and hence, are not present in the data (Heckerman, UCI KDD Archive). As an example, we write below the sequences for the first three user sessions in the data set (one line per user): (to see the big data set go to

`http://kdd.ics.uci.edu/databases/msnbc/msnbc.html`

but remember that this data set is already a processed version of the original web logs.

```
User session 1: frontpage, frontpage
User session 2:  news
User session 3:  tech,news,news,local,news,news,news,tech,tech
```

or, symbolically,

```
User session 1:  1, 1
User session  2:  2
User session 3:  3, 2, 2, 4, 2, 2, 2, 3, 3
```

The above is saying that the first user session entered the msnbc.com web site via the frontpage, and hit two links in the frontpage. User session 3, however, started in the tech page, moved to the news page and hit two links there, then to local, then to news again and hit two links there, then to tech again, and hit two links there. The reader should visit the msnbc.com web site to understand that the user can move from any page category to another due to the frame structure of that web Site. This aspect makes this web Site different from Sites used in other papers.

To use markov models, some authors have considered the last category visited the "end state" but we attach an additional state at the end of each sequence, the "exit" state, with symbol 18. Once in it, the user can not return to any of the other states unless it starts another sequence. For the user sessions above then, the sequences are:

```
User session 1:  frontpage, frontpage, exit
User session 2:  news, exit
User session 3:  tech,news,news,local,news,news,news,tech,tech, exit
```

or, symbolically,

```
User session 1:  1, 1, 18
User session 2:  2, 18
User session 3:  3, 2, 2, 4, 2, 2, 2, 3, 3, 18
```

It is also interesting to see the first order Markov transition matrix derived from the data. As you know, the frequency of transition from one page to another in the data is the maximum likelihood estimate of the transition probabilities. It helps to see the frequencies with which user sessions move from the page they are at to the next page. This can be seen on the following table. Notice how there is a positive probability of moving from one state to the same state.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	44.5	7.8	2.5	2.9	0.9	3.3	3.0	0.9	0.3	2.0	2.9	4.4	0.2	3.8	0.6	0.1	0.5	19.3
2	10.1	49.1	2.4	2.5	0.7	2.1	1.4	1.5	0.5	1.7	0.8	2.2	0.1	1.9	1.3	0.0	0.2	21.6
3	8.0	3.8	34.7	1.7	0.7	1.4	0.4	0.6	2.3	1.2	1.4	2.0	0.4	1.3	1.3	0.0	0.3	38.5
4	5.6	2.9	0.9	58.2	0.4	1.3	5.6	1.0	1.9	0.7	0.6	0.8	0.6	1.1	0.2	0.0	0.1	18.0
5	3.8	1.8	0.6	0.5	78.6	1.5	0.2	0.6	0.9	0.4	0.5	0.4	0.1	0.4	1.2	0.2	0.0	8.3
6	4.4	2.6	1.4	1.7	0.7	34.7	8.9	1.1	1.8	1.6	0.7	1.0	0.7	0.8	2.8	0.1	0.0	35.1
7	8.3	1.1	0.3	10.8	0.1	9.6	58.1	0.3	2.4	0.7	0.1	0.3	2.6	0.9	0.3	0.0	0.0	4.1
8	1.4	1.9	0.3	0.8	0.2	0.7	0.6	74.9	0.7	0.2	0.2	0.3	0.4	0.5	0.1	0.0	0.0	16.6
9	5.1	2.1	2.7	5.1	1.4	1.9	4.9	2.4	41.0	0.6	1.0	2.1	2.6	0.3	0.0	0.0	0.0	26.8
10	9.1	5.5	2.1	1.4	0.6	3.5	1.5	0.6	0.9	51.5	1.2	1.7	0.1	0.9	1.7	0.0	0.2	17.5
11	19.6	4.1	1.9	2.4	1.6	3.1	0.7	0.7	1.3	1.7	31.5	1.5	0.3	1.8	1.5	0.0	3.0	23.2
12	10.3	3.2	2.1	1.3	0.4	1.1	0.6	0.5	2.1	0.9	0.7	49.0	0.4	2.0	0.5	0.0	0.1	24.8
13	1.3	0.1	0.3	0.8	0.0	0.7	3.9	1.1	1.7	0.0	0.1	0.3	55.1	8.4	0.0	0.0	0.0	26.0
14	6.7	1.2	0.6	0.8	0.1	0.6	0.8	0.4	0.4	0.2	0.3	0.7	2.5	63.8	0.2	0.1	0.1	20.4
15	6.2	8.8	4.3	1.8	4.4	11.6	3.6	0.8	0.4	5.0	3.8	2.0	0.1	2.4	23.8	0.1	0.3	20.8
16	2.7	0.9	0.1	0.2	0.6	1.1	0.1	0.1	0.1	0.2	0.1	0.2	0.0	1.6	0.2	87.7	0.0	4.1
17	21.8	4.6	2.2	2.0	1.1	2.2	3.2	0.8	0.4	1.7	9.5	1.6	0.1	1.5	0.9	0.0	26.3	20.0
18	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0

Table 1: First Order Markov transition matrix (Percent)

Question 1 Notice that in the transition matrix, there is a positive probability of moving from one state to the same state. What in the way the data have been processed explains why ? Would summarizing the data differently result in 0 probability of moving from one state to the same state?

We can see in the transition matrix that users tend to move from a link in a page category to another link in the same page category more often than any other move. It is also worth mentioning that regardless of the page category in which a user finds her or himself, the next most preferred page category is page 1, the front page.

Attached, you will find the Perl programs used to first add the 18 at the end, and then to generate the first order markov model transition probability matrix given above from the msnbc data set.

3. Simulating to determine the lengh of a visitor's visit, i.e., total number of clicks

We can simulate states for visitors with the above transition probabilities to see how long it takes a person to reach the exit state. With the simulated lengths, we can determine the distribution of length implied by the model and compare it to the length from a random sample of users from the data. This way, we can see whether the markov model assumed is reasonable. If the distribution of length in the data is very different from the distribution of length in the simulated data, we can doubt that Markov is a good model. And viceversa otherwise. There are many other things that we could check, but in the present activity, the length variable will be the only one compared.

To simulate, we need to make some assumption about the probability model for entering the system, i.e,, the probability model generating the initial state. That is, are we assuming that visitors are equally likely to enter through the frontpage as through the sport page or any of the other pages? Or should we set up a more realistic model?

4. Activity

In this activity, you are going to simulate the visits of 1000 users, i.e., for each user, you are going to generate a path of page clicks until they exit. The assumption made are (1) that a uniform probability model represents the gate through which the visitor enters, or the initial state. We are also assuming (2) that only the last state matters (i.e., we will use the above estimated transition matrix), or a first order markov model for transitions from one page to another. This implies that we are assuming that there is no back button actions, or long memory of any kind.

4.1. Part 1 of the activity

Start R and once in it, type the following command

```
>source("http://www.stat.ucla.edu/˜jsanchez/teaching/stat258/reading/simulatedlog.R")
```

This runs the R program attached at the end of this activity, which you should have studied before starting the activity to understand how the simulation is done. You will see that the program simulates visits and will compute the length of a visit for each of 1000 visitors, assuming that visitors entered the web site with equal probability through any of the pages.

Open the file

```
 simulated_logo.txt
```

with a text editor, to see the sequences of visits generated for the 1000 simulated visitors. Some of them are long, some short.

A file called

```
 simulated_length.txt
```

is also generated that will contain the length implied by the simulated sequences for each user and the cumulative length, two variables. Read this file and extract the length variable only. Look at a plot of the simulated length and describe it.

```
> simlengthdata= read.table("simulated_length.txt")
>simlength=simlengthdata$V1
>n=length(simlength)
> hist(simlength,main="histogram of simulated data",xlim=c(0,100))
```

To see whether this data is a good fit to the real length data obtained from the msnbc data file, we can start by doing a qq-plot. But first we sample 1000 observations from the msnbc data set, which we will read again.

```
> msnbc=read.table("http://www.stat.ucla.edu/~jsanchez/teaching/stat258/
reading/msnbclength.txt",header=T)
> msnbclength=msnbc$length
> msnbclength1000=sample(msnbclength,1000)
> hist(msnbclength1000,main="histogram of msnbc length data",xlim=c(0,100))
> qqplot(simlength,msnbclength1000,main="qq plot length data and sim")
> abline(0,1)
```

Graphs can be very misleading. It would be better to have some more specific measures of the discrepancy or similarity between the simulated and the msnbc length data. Let's do some additional computations.

```
>library(fBasics)     # Load this package.
>skewness(simlength)
>skewness(msnbclength1000)
>kurtosis(simlength)
>kurtosis(msnbclength1000)
```

The kurtosis is based on the size of the distribution's tails. The normal distribution has kurtosis 0. Distributions with large tails are called leptokurtic and are indicated by kurtosis coefficient larger than 0. [1]. Distributions with small tails have negative kurtosis coefficient. The skewness coefficient measures whether a distribution has one tail longer than the other.[2] The normal distribution has skewness coefficient 0. The farther the coefficient is from 0, the higher the skewness.

Supposedly, a first order markov model should generate length data that fits well the inverse gaussian distribution. So we do a mle and check the goodness of fit to see.

```
library(stats4)
fn=function(mu,lambda)
{
 -(   (n/2) *log(lambda)-(n/2)*log(2*pi) -(3/2)*sum(log(simlength))-((lambda)/2)*
 sum((((simlength/mu)-1)**2)/simlength))
 }
est=mle(minuslogl=fn,start=list(mu=0.1,lambda=0.1))
summary(est)
```

To see graphically how the length model implied by the transition matrix fits the msnbc length data, we can overimpose the model derived from the transition matrix over the histogram of the msnbc data set.

```
library(dse1)
library(SuppDists)
siminvnormal=rinvGauss(1000,5.459,46.206)
h=hist(msnbclength1000,breaks=15)
xhist=c(min(h$breaks),h$breaks)
yhist=c(0,h$density,0)
xfit=seq(min(siminvnormal),max(siminvnormal),length=40)
yfit=dinvGauss(xfit,5.459,46.206)
plot(xhist,yhist,type="s",ylim=c(0,max(yhist,yfit)),main="Inv Gauss pdf and msnbc lenght")
lines(xfit,yfit,col="red")


library(stats4)
fn=function(mu,lambda)
{
 -(   (n/2) *log(lambda)-(n/2)*log(2*pi) -(3/2)*sum(log(msnbclength1000))-((lambda)/2)*
 sum((((msnbclength1000/mu)-1)**2)/msnbclength1000))
 }
est=mle(minuslogl=fn,start=list(mu=0.1,lambda=0.1))
summary(est)
```

[1] the Kurtosis coefficient is computed as $\frac{\sum(x-\mu)^4}{n\sigma^4} - 3$

[2] The skewness coefficient is calculated by $\frac{\sum(x-\mu)^3}{n\sigma^3}$

Question 1: Based on all the analysis done above, what is your impression about the markov model. Do you think that it models well the browsing behavior of the msnbc length data? Why? Why not? Support your answers with the analysis done.

4.2. Part II of the activity

The assumption that all pages are equally likely to be gates of entry seems a little unrealistic in view of the following table, which shows clearly that most people enter via page 1, 2, 3, 4, 6, 8,9, 12,13,14. For this reason, we should try to correct the program to allow for this fact by sampling with different probabilities from the different gates instead of with uniform probabilities. We will use

Table 2: default

Page	Proportion entering via this page
1	28.02
2	7.79
3	6.72
4	5.15
5	0.42
6	16.47
7	0.35
8	7.25
9	6.59
10	1.29
11	1.44
12	5.73
13	6.36
14	5.41
15	0.74
16	0.04
17	0.15

We could just do as in Baldi et al. and use these probabilities to respecify the transition probability matrix. These could be the Probabilities of returning to those pages after exiting the system.

Question 2.- Rewrite the markov matrix to account for this slight change. Also, in the program used, determine where we would have to change the program to account for this.

References

[1] Cadez, I., Heckerman, D., Meek, C., Smyth, P., White, S. (2003). Model-based clustering and visualization of navigation patterns on a Web site. Accepted for publication on Journal of Data Mining and Knowledge Discovery, 7(4).

[2] Hansen, M.H. and Sen, R.(2003). Predicting Web User's next access based on log data. Journal of Computational and Graphical Statistics, Vol 12, No. 1, March, p. 143-155.

[3] Heckerman, D. The UCI KDD Archive (`http://kdd.ics.uci.edu`) Irvine, CA: University of California, Department of Information and Computer Science. The URL for the data used in this paper is `http://kdd.ics.uci.edu/databases/msnbc/msnbc.data.html`

[4] Huberman, B.A., Pirolli, P.L.T., Pitkow, J.E., Lukose, R.M. (1998). Strong Regularities in World Wide Web Surfing. Science, Vol. 280, 3 April.

Perl program to append the 18 (exit) state at the end of the data before we calculate the transition matrix for first order markov model.

```
#This program reads the msnbc data after downloading it from its site  and appends the
# number 18 (exit) at the end of each line.

#!/usr/bin/perl -w
open(INFILE,'msnbc');
open(NEW,'>>appended');

$WordCount = 0;

while(<INFILE>) {
$TheLine = $_;    #Save the line's contents
chomp($TheLine);  #Get rid of the line break

 $TheLine .= 18;  #Append the exit code 18 to TheLine

@TheLine = $TheLine;

printf NEW "@TheLine\n";
}
```

After this program,the data looks like (first few lines)

```
1 1 18
2 18
3 2 2 4 2 2 2 3 3 18
5 18
1 18
6 18
1 1 18
6 18
6 7 7 7 6 6 8 8 8 8 18
```

This Perl program that follows, calculates the transition matrix for the first order markov model for the msnbc data.

```
#!/usr/bin/perl/

######
# filename: transition.pl
# Goal: Compute the transitioin matrix by reading the data file
# data file: each row represents one users.
# Transition matrix : 18 states
######

### Parameters setting ###
#$fin1 = "test_appended.txt";
#$fout1 = "test_tran_matrix_count.txt";
#$fout2 = "test_tran_matrix_prob.txt";
#$fout3 = "test_length.txt";


$fin1 = "appended";
$fout1 = "tran_matrix_count.txt";
$fout2 = "tran_matrix_prob.txt";
$fout3 = "length.txt";
$state = 18;

##
open(FIN1, $fin1) || die "Cannot open the file $fin1\n";
open(FOUT1, '>'.$fout1) || die "Cannot create the file $fout1\n";
open(FOUT2, '>'.$fout2) || die "Cannot create the file $fout2\n";
open(FOUT3, '>'.$fout3) || die "Cannot create the file $fout3\n";

#$t = join " ", 4,5;
#print "$t\n";

#### Define the hash table for transition matrix
$i =1;
$j =1;
while ($i < ($state+1)) {
$tm{"$i"} = 0;
while ($j < ($state+1)) {
$key = join " ", $i, $j;
$tm{"$key"} = 0;
$j++;
}
$j = 1;
$i++;
}

## to count the time for each transition
$count = 0;

while (<FIN1>) {
chomp;
@temp = split(/\s+/,$_);
$number = $#temp;
$k1 = 0;
while ($k1  < $number) {
$tm{"$temp[$k1]"}++;
$key = join " ", $temp[$k1], $temp[$k1+1];
#$tm{"$key"} = $tm{"$key"} +1;
$tm{"$key"}++;
$k1++;
$count++;
}
print FOUT3 "$k1 $count\n";    # print the length
#last;
}

### print the transition prob and count
$ii =1;
$jj =1;
# last state is an absoring state
$absor = join " ", $state, $state;
##$tm{"$absor"} = $count;
while ($ii < ($state+1)) {
while ($jj < ($state+1)) {
$key = join " ", $ii, $jj;
if ($tm{"$ii"} >0) {
$prob = $tm{"$key"}/$tm{"$ii"};
} elsif ($key =~ $absor) { ## abosring state ie state 18
$prob =1;
} else {
$prob =0;
}
print FOUT1 "$tm{\"$key\"} ";
#print FOUT1 " ";
printf FOUT2 "%.3f", "$prob";
print FOUT2 " ";
$jj++;
}
print FOUT1 "\n";
print FOUT2 "\n";
$jj = 1;
$ii++;
}
print FOUT1 "$count\n";
#print FOUT2 "$count\n";
```