MATH 728D: Machine Learning Lab #13: Markov Methods

John Burkardt

December 10, 2018

Can we predict long-term behavior in a system, knowing only one-step probabilities?

In this lab, we look at models in which there are a discrete set of possible states, and a discrete sequence of times. At each new time step, a system which was in state j will transition to state i with a probability $A(i,j)$. We would like to use this model to simulate the behavior of the system, to characterize the frequency with which the system occupies each state, to estimate an associated probability density function, or the expected value of a function which depends on the state.

1 Stock Market Mood

Stock markets can be simply categorized as *bull*, *bear* or *stagnant*, depending on whether stock prices are mostly trending up, down, or seem to be mainly stable. Label these three states as 1, 2, and 3. Suppose the matrix A describes in $A(i,j)$ the probability that, if on one day the stock market is in state j, that on the next day it will be in state i.
A = [0.850, 0.150, 0.250; ... 0.075, 0.750, 0.150; ... 0.075, 0.100, 0.600];

$$A = \begin{pmatrix} 0.85 & 0.15 & 0.25 \\ 0.08 & 0.75 & 0.15 \\ 0.07 & 0.10 & 0.60 \end{pmatrix}$$

Exercise 1:

1. create the matrix `A`;
2. create the matrix `A_cum` from `cumsum(A)`;
3. create a vector `s` of length 100, set `s(1)=1`, and set the next 99 values by simulation of the market:

```
for k = 2 : 100
  r = rand ( );
  j = s(k-1);
  for i = 1 : 3
    if ( r < A_cum(i,j) )
      s(k) = i;
      break
    end
end
```

4. Make a line plot of the 100 values of `s` and notice that some states seem more likely than others;
5. repeat the simulation to get 1000 values of `s`, and store the relative frequency of each state in a vector `x`; For instance, `x(1)=sum(s==1)/1000`;

6. Create a 3×1 column vector `y` with random entries. Overwrite `y` by `A*y` 100 times. Then normalize `y` by dividing by `sum(y)`;
7. Request the eigenvectors and eigenvalues of `A` by `[X,L]=eig(A);`.
8. The largest eigenvalue of A should be 1. Let `j` be the column of `L` in which this value occurs;
9. Set the vector `z` to the corresponding jth-column of `X`, normalized to have sum 1;
10. Print `x`, `y`, and `z` and comment;

2 Trapping a Fly

From Barber, Bayesian Reasoning, Example 5.1
You live in a house with three rooms, labelled 1, 2, 3. There is a door between rooms 1 and 2 and another between rooms 2 and 3. One cannot directly pass between rooms 1 and 3 in one time-step. An annoying fly is buzzing from one room to another and there is some smelly cheese in room 1 which seems to attract the fly more. Using x_i to indicate which room the fly is in at time i, the movement of the fly can be described by a transition matrix:

$$A = \begin{pmatrix} 0.7 & 0.5 & 0.0 \\ 0.3 & 0.3 & 0.5 \\ 0.0 & 0.2 & 0.5 \end{pmatrix}$$

so that $x_{i+1} = A * x_i$.

Exercise 2:

1. If the fly is in room 3 on time step 1, simulate the motion of the fly over the next 49 time steps, and display a line plot of this behavior; (Set x to 3. Pick a random value, and use the cumulative column sums in column x to decide where to move next)
2. If the fly is in room 3 on time step 1, what is the probability that the fly will be in room 1 on time step 5? (Let p=[0;0;1]. Compute next step probabilities by $p = A * p$. Repeat.)
3. What is the stationary (very-long-term) probability that the fly will be in room 1, 2 or 3; that is, what is the limiting probability distribution p, such that $A * p = p$? (Use power method or eig() command)

3 From Peru to New Guinea in the game of RISK

RISK is a war game played on a global map divided into 42 regions. A player's armies can attack any neighboring region, and move into that region if successful. Our interest is in studying the consequences of the way the regions are connected.

Exercise 3:

1. Use `csvread('risk_data.csv')` to load the RISK adjacency matrix `A` where `A(i,j)=1` if regions `i` and `j` are adjacent;
2. Use the command `spy(A)` to get a picture of the nonzero structure of the adjacency matrix;
3. We can think of `A(i,j)` as the number of ways to reach region `i` from region `j` in 1 step. It turns out that `A^2(i,j)` is the number of ways to reach `i` from `j` in exactly 2 steps, and so on. Peru is region 11, and New Guinea is region 40; what is the minimum number of steps required to move from Peru to New Guinea?
4. You can refer to the map *risk_map.png* to confirm your computation;
5. Now suppose that every region starts out with 1 barrel of oil. Every year, each region with n neighbors sends $\frac{1}{n}$ barrel of oil to each neighbor, and receives similar tribute from its neighbors. What happens to the oil as time passes?

6. Create matrix `B`, a copy of `A` in which each column has been divided by the number of 1's in that column.
7. Initialize the 42×1 vector `x` to 1's. Now repeatedly replace `x` by `B*x`; check the result after 10 steps; then again after 100 steps; what do you observe? Look at the map and try to explain the pattern.

4 Hyperlinks and Page Rank

The Page Rank algorithm for ranking web pages works by considering the World Wide Web to be a graph, with each web page a node, and each hyperlink from page j to page i represented by a value of 1 in the directed adjacency matrix $A(i, j)$.

We normalize each column of A so that it sums to 1. (And if a column has no entries at all, we can insert a 1 on the diagonal.) This guarantees that the dominant eigenvalue of A is 1. Now we seek a solution to $A * x = x$. We can assert that web page i has relative importance or rank equal to $x(i)$.

For small systems, we can easily compute x with eigenvalue software. For larger systems, (in particular, the World Wide Web), a different approach is necessary.

We will study a small system and show that both approaches get approximately the same result.

Exercise 4:

1. Create the adjacency matrix `A` by using `csvread()` to read the file *hyperlink_data.csv*;
2. You can see a map of the hyperlinks in *hyperlink_map.png*;
3. For each column of `A`, compute the column sum, and divide each entry in the column by that value.
4. Get the eigenvectors and eigenvalues of A from MATLAB; Set `y` to be the eigenvector corresponding to the dominant eigenvalue (which is 1). Normalize `y` to have sum 1;
5. Now we will try a second estimate, using "surfing";
6. Set `A_cum` as the output from `cumsum(A)`;
7. Initialize the 16×1 vector `z` to 0;
8. Initialize $j = randi([1, 16]$ and set `z(j)=z(j)+1`;
9. Now repeat the following n=999 times:
   (a) Set `r=rand()`;
   (b) find the first row `i` such that `r` $\leq$ `A_cum(i,j)`;
   (c) Set `j = i` and `z(j) = z(j) + 1`;
10. Normalize `z=z/sum(z)`;
11. Print `y` and `z` and compare;