Homework 6: Markov Chains

UC Irvine CS177: Applications of Probability in Computer Science

Due on December 5, 2019 at 11:59pm

You may use computer software like Python to compute the probabilities requested in the following problems, as long as you show your work by specifying the commands you use.

Question 1: (30 points)

In a particular region of the Pokémon world, there are just three types of Pokémon: *Dark*, *Psychic*, and *Fighting*. Kecleon, the Color Change Pokémon, has the ability to change its type to blend into its surroundings. Its goal is to become the best type. It seeks to accomplish this goal by choosing a type of Pokémon to battle uniformly at random; it may choose a Pokémon of its own type. If it wins the battle, it will keep its old type. If it loses, it will take on the type of the Pokémon that defeated it. It continues this process indefinitely.

For all questions assume that *Dark* Pokémon always defeat *Psychic* Pokémon, *Psychic* Pokémon always defeat *Fighting* Pokémon, and *Fighting* Pokémon always defeat *Dark* Pokémon. If two Pokémon of the same type battle, each has an equal chance of winning.

a) *Suppose that Kecleon starts as the* Fighting *type. After two battles, what is the probability distribution of Kecleon's type? (Your answer should be three numbers that sum to one.)*

b) *After many (approaching infinity) battles, what is the steady-state probability distribution of Kecleon's type? (Your answer should be three numbers that sum to one.)*

A nearby graveyard is haunted! *Ghost*-type Pokemon invade the population. Assume that *Ghost* Pokémon always lose to *Dark* Pokémon, and always win against *Psychic* Pokémon and *Fighting* Pokémon.

c) *Suppose that Kecleon starts as the* Fighting *type when the* Ghost *Pokémon arrive. After two battles, what is the probability distribution of Kecleon's type? (Your answer should be four numbers that sum to one.)*

d) *After many (approaching infinity) battles, what is the steady-state probability distribution of Kecleon's type? (Your answer should be four numbers that sum to one.)*

Question 2: (30 points)

Consider a standard chessboard with an 8×8 grid of possible locations. We define a Markov chain by randomly moving a single chess piece on this board. The initial location X_0 is sampled uniformly among the $8^2 = 64$ squares. At time t, the piece then chooses X_{t+1} by sampling uniformly from the set of legal moves given its current location X_t. For a description of legal chess moves, see: `http://en.wikipedia.org/wiki/Rules_of_chess#Basic_moves`.

a) Suppose the chess piece is a king, which can move to any of the 8 adjacent squares. Is the Markov chain irreducible? Is the Markov chain aperiodic? Justify your answers.

b) Suppose the chess piece is a bishop. Is the Markov chain irreducible? Is the Markov chain aperiodic? Justify your answers.

c) Suppose the chess piece is a knight. Is the Markov chain irreducible? Is the Markov chain aperiodic? Justify your answers.

Question 3: (40 points)

When you type keywords into a search engine, it displays pages containing related terms, but how should the output be ordered? To address the weaknesses of ranking algorithms based solely on word counts, we explore the famous *pagerank* algorithm, which formed the basis for at least early versions of Google's search engine. Think of the whole internet as a directed graph where each node is a website, and there is a directed edge between node i and j if and only if website i hyperlinks to website j. Intuitively, the pagerank algorithm seeks a ranking for which: i) If a website is linked to by many other websites, then it's an important website; ii) If a website has only a few links, but those links comes from authoritative sites, then it's also important; iii) If a website links to a very large number of other websites, then the "importance" it transfers to each individual site is small. The pagerank algorithm uses Markov chains to allow the information provided by a link to implicitly flow both directions.

To illustrate pagerank, imagine a "random surfer" on the internet that starts at some webpage, and sequentially visits other webpages by following hyperlinks. As illustrated in Figure 1, the surfer chooses between the outgoing links from each page with equal probability. We can then define the "importance" of webpage i as the long-term frequency with which this random surfer visits webpage i. If a node has k outgoing edges, then the fraction of time a visit to this node is followed by each linked neighbor is only $\frac{1}{k}$. Denoting the state transition matrix by T, if the initial location of the surfer is uniform over the m nodes so $\pi_0 = [\frac{1}{m}, \frac{1}{m}, \ldots, \frac{1}{m}]$, the probability of viewing each webpage after n time steps is then $\pi_0 T^n$. If n is large and there are paths between all pairs of nodes, the state probabilities will converge to a stationary distribution $\pi = \pi T$. Sorting these probabilities gives the pagerank.

a) Create the state transition matrix T for the small network of figure 1. What is the equilbrium distribution of this Markov chain? Which webpage has the highest pagerank?

Of course, the structure of real-world networks is more complex than Figure 1. A naive random surfer could get stuck in a "dead end" page (an absorbing state) or some locally connected subset of the full web. Thus at each step, with probability α the surfer "teleports"

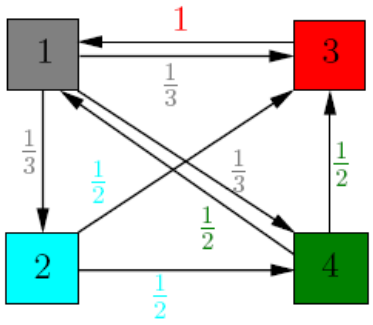

Figure 1: A small network of directed links between four webpages, and corresponding state transition probabilities. (Image courtesy of Mathematics Explorers Club, Cornell University.)

to an arbitrary website at random; each of the m websites has probability $\frac{1}{m}$ of being chosen. With probability $1-\alpha$, the surfer follows a hyperlink as above. The overall state transition matrix G of this new Markov chain is then

$$G = (1-\alpha)T + \alpha B, \qquad B = \frac{1}{m}\begin{pmatrix} 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{pmatrix}. \tag{1}$$

The matrix G is also known as the "Google matrix". In our experiments, we set $\alpha = 0.15$.

We now explore a larger network of $m = 9664$ websites. We provide the link structure of those sites in the matrix `L`, where $L(i,j) = 1$ if there is a link from website i to website j, and $L(i,j) = 0$ otherwise. The `name` variable stores the names of each website.

b) Write code that creates the state transition matrix T, and Google matrix G, for the provided data. If website i has no outgoing links, then set $T(i,i) = 1$. You should double-check that for both T and G, the sum of the transition probabilities for each state equals 1.

c) From a uniform initial state distribution π_0, apply the Google matrix G for $n = 100$ time steps to compute $\pi_1, \pi_2, \ldots, \pi_{100}$. At each iteration t also compute the magnitude of the absolute change in state probabilities:

$$\epsilon(t) = \sum_{i=1}^{m} |\pi_{t,i} - \pi_{t-1,i}|$$

Plot $\epsilon(t)$ versus t. Does π_t appear to converge to a limit? If so, which 25 webpages have the highest pagerank? What is the steady-state probability of visiting these top 25 sites? Discuss your results, keeping in mind that this dataset was collected in 2002.

d) Consider a modified pagerank algorithm where instead of following "outgoing links", the random surfer follows "incoming links". If the surfer is at website i, it chooses uniformly among the sites that link to website i. This algorithm is equivalent to reversing the direction of all links in the original data, and then applying the standard pagerank algorithm. Report the top 25 webpages, and their steady-state probabilities, for this reversed pagerank algorithm. Discuss differences from part (c). Which ranking seems more sensible?