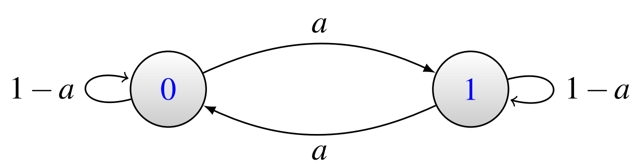
Markov Chains as Market Predictors

Introduction

Markov chains are powerful tools to model dependent events. We can model Markov chains by simply drawing a graph to illustrate those dependent events, which give us information about the behavior of these events in the long run. In this project, we focused on analyzing the market transition probabilities for three different market types: bull, stagnant, and bear. Using Markov chains, we deduce how long would it take to transition from bear to bull, the probability of transitioning form bull to stagnant without visiting bear, the stationary distribution vector, maximum stream of events, and the convergent stream of events using the law of large numbers.

1 Markov Chain

A Markov chain is simple scheme to transition from one state to another in the state space:



In this case, our state space S consists of states $\{X_0, X_1\}$ and the transition probabilities between states are:

$$\mathbb{P}[X_0 = 0 | X_1 = 0] = 1 - a$$
$$\mathbb{P}[X_0 = 1 | X_1 = 0] = a$$
$$\mathbb{P}[X_1 = 0 | X_0 = 1] = a$$
$$\mathbb{P}[X_1 = 1 | X_0 = 1] = 1 - a$$

As illustrated above, transitions from one node always add up to 1, and the probability of next state is only dependent on the previous state, so the Markov Process itself is memoryless. In addition, this Markov chain is irreducible and aperiodic, which implies that a unique invariant distribution for this Markov chain exists, implying that at the end of the process, this Markov chain will have converging values. In addition, it is possible to find the time it takes to transition from one state to another as well as finding the probability of entering a state without entering another state. In addition, using a Monte Carlo process approach for more complex Markov Chains will enable us to simulate large sample of data sets to give us the most likely stream of events. The transition probabilities in finite Markov chains produce a matrix $P \in \mathbb{R}^{|S|x|S|}$ and the rows of the matrix contain transition probabilities from one state to other states in order. In this case, our probability matrix would look as follows:

$$\begin{bmatrix} 1-a & a \\ a & 1-a \end{bmatrix}$$

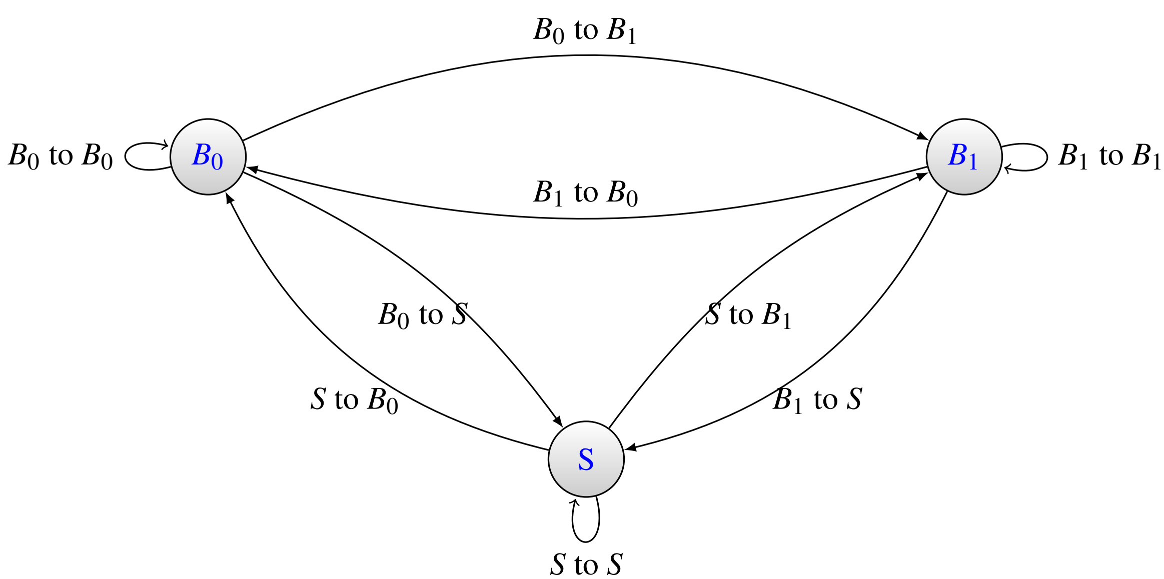
In a probability matrix the following is always true:

$$\sum_{j=1}^{|S|} P(i,j) = 1, \forall i \in \mathbb{N}$$

In addition, given an initial distribution π_0, we can find the final distribution after n steps with the following equation:

$$\pi = \pi_0 P^n$$

The state space of our Markov chain is $S = \{\text{Bull}, \text{Stagnant}, \text{Bear}\}$, and it is also irreducible, which will result in a graph as follows (B_0 represents Bull, B_1 represents Bear, and S represents Stagnant):



2 Stationary Distribution Vector

Given a finite irreducible aperiodic Markov chain (in our case it is), it is guaranteed to have a stationary distribution after any number of steps, we set the equation as:

$$\pi = \pi P$$

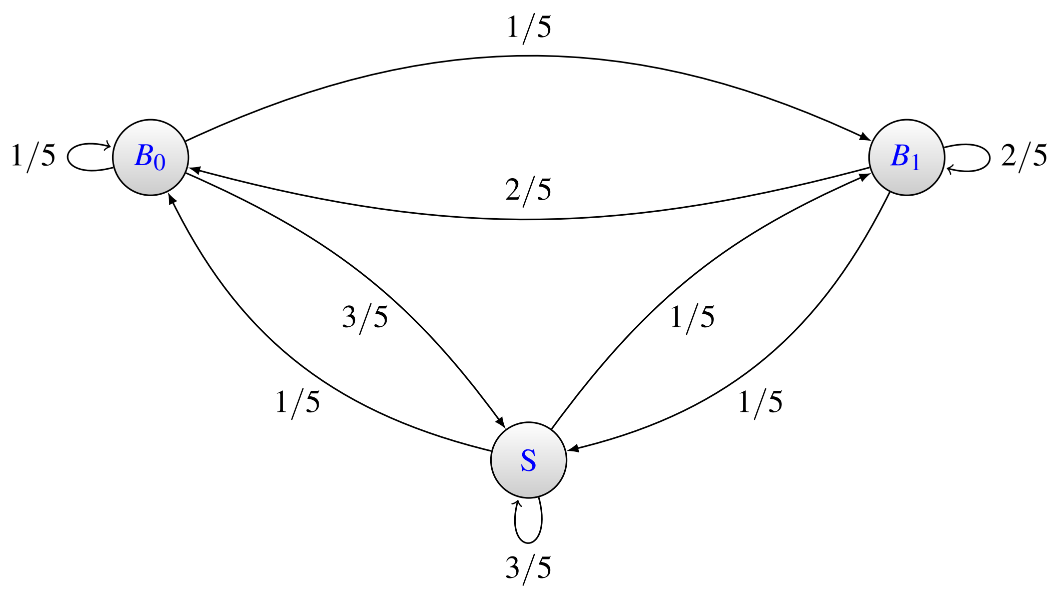
Where π is the initial distribution vector and P is the probability transition matrix. We can find the stationary distribution vector in two different ways, eigendecomposition and a brute force algorithm to solve for the value of π. In our project, we took the first approach as it was easier to implement. With eigendecomposition, we'll set the determinant of the matrix $\det(P - \lambda\mathbb{I}) = 0$, then solve for the eigenvalues. Since our Markov chain is guaranteed to have a stationary distribution, which also guarantees that one of the eigenvalues is 1 and the rest are smaller than 1 (if they exist). Having this information, the initial distribution vector corresponds to the eigenvector of the matrix P. Let $A = P - \mathbb{I}$, then $\pi(P - \lambda\mathbb{I}) = \pi A$, let π be the tranposed vector $[\pi_{B_0}, \pi_S, \pi_{B_1}]$, then $\pi A = \vec{0}$, which corresponds to finding the nullspace of A^T. The stationary distribution vector gives us the final probabilities of being in each state, which is an important feature. The method in the class MarkovProcess takes no input and calculates the following process with the help of numpy.

3 Hitting Time

Given any finite Markov chain, it is also possible to calculate the time it takes to transition from one state to another, with the following formula if the desired state is $\mathbb{E}$:

$$\beta(i) = 0, \text{if i} \in \mathbb{E}$$
$$\beta(i) = 1 + \sum_{i \in S} P(i, j)\beta(j)$$

Consider the following Markov chain and transitioning from B_0 to S:



For this Markov chain, we can write the first step equations (FSE) to calculate hitting time:

$$\beta(B_0) = 1 + 1/5\beta(B_0) + 1/5\beta(B_1) + 3/5\beta(S)$$
$$\beta(B_1) = 1 + 2/5\beta(B_0) + 2/5\beta(B_1) + 1/5\beta(S)$$
$$\beta(S) = 0$$

Solving this linear system of equations gives us:

$$\beta(B_0) = 2$$

This implies that it takes 2 time steps to transition from B_0 to S. The our method in MarkovProcess class, takes two inputs start state and end state, then applies this formula.

4 Probability of A before B

It is also possible to find the probability of transitioning from one state to another without ever visiting a specific state. Assume that you want to transition to a state in A without ever visiting a state in B:

$$\alpha(i) = \sum_j P(i,j)\alpha(j)$$

$$\alpha(i) = 1 \forall i \in A$$

$$\alpha(i) = 0 \forall i \in B$$

Consider the previous Markov chain again and consider transitioning from Bull to Stagnant without ever visiting Bear, we can set the first step equations as follows:

$$\alpha(B_0) = 1/5\alpha(B_0) + 1/5\alpha(B_1) + 3/5\alpha(S)$$

$$\alpha(B_1) = 0$$

$$\alpha(S) = 1$$

Solving the above set of linear equations, we get:

$$\alpha(B_0) = 3/4$$

The method in the class MarkovProcess, takes a start state and end state as input and finds the state to avoid and applies the above procedure to find the probability in question.

5 LLN simulator

The law of large numbers states that for a set of independent and identically distributed random variables, the probability of the average of their sum minus their mean being bounded by an integer $\varepsilon > 0$ tends to 1 as $n \to \infty$. Let X be the sum of iid random variables X_i and let X_i have variance σ^2:

$$\mathbb{P}[|\frac{1}{n}\sum_{i=1}^{n} X_i - \mathbb{E}[\frac{1}{n}\sum_{i=1}^{n} X_i]|] < \varepsilon = 1, n \to \infty$$

We can prove this by the Chebyshev equality as follows:

$$\mathbb{P}[|\frac{1}{n}\sum_{i=1}^{n} X_i - \mathbb{E}[\frac{1}{n}\sum_{i=1}^{n} X_i]| \geq \varepsilon] \leq \frac{Var(\frac{1}{n}\sum_{i=1}^{n} X_i)}{\varepsilon^2}$$

$$\frac{Var(\frac{1}{n}\sum_{i=1}^{n} X_i)}{\varepsilon^2} = \frac{\frac{1}{n^2}\sum_{i=1}^{n} Var(X_i)}{\varepsilon^2}$$

$$= \frac{n\sigma^2}{n^2\varepsilon^2}$$

As n goes to infinity, the last expression tends to 0. Since we are interested in the complement of this event, the complement goes to 1 as desired and proves that as the sample size goes to infinity, the average of the iid random variables approach to their mean. With the LLN simulator method, we can use this fact to find the probability of transitioning from one state to other by calling the random forecast method by input $\cdot$ 100 times and counting the number of times that the desired state is the end state in our random stream of events. The method we defined does this by creating an empty list "forecast list" and generating a random stream the input time times 100, then for each list in the forecast list, it looks at the last element and if it matches the last element, it increases the count by 1, then it returns the percentage $\frac{\text{count}}{100} \cdot 100$, which is guaranteed to be accurate by the law of large numbers.

6 Random Forecast and Max Stream

The MarkovProcess class includes three other methods, namely max stream and random forecast. Random forecast takes an integer input and takes a random walk along the Markov chain scheme and returns the stream with its probability. The max stream method takes an integer input and simulates the process with random forecast for input $\cdot$ 100 times, and stores the probabilities in a dictionary and returns the stream with the maximum likelihood in $O(n^2)$ time complexity.

Contributors

Atahan Ozdemir

Tan Sarp Gerzile