Using Markov Chain Monte Carlo methods to analyze customer churn in the telecommunications industry

Abhishek Kumar
Enroll. No.: 20312001
Supervised by: Professor Chaman Kumar

Objectives

The objectives of this project are:

- To learn about Markov Chain Monte Carlo methods
- To understand the concept of customer churn in the telecommunications industry
- To analyze customer churn using Markov Chain Monte Carlo methods
- To draw conclusions and make recommendations based on the analysis

Markov Chain Monte Carlo (MCMC) Method

The **Markov Chain Monte Carlo (MCMC) method** is a popular statistical technique used to sample from complex probability distributions that are difficult to compute or intractable to solve analytically.

In general, the MCMC method involves generating a sequence of random samples from the target distribution by constructing a **Markov chain** that has the target distribution as its stationary distribution.

Each sample in the sequence is generated by moving from the current state of the Markov chain to a new state based on a transition probability function that satisfies the **detailed balance condition**.

The Monte Carlo Approximation

The Monte Carlo approximation is a stochastic integral approximation method that is widely used in finance, engineering, and other fields.

The mathematical form of the Monte Carlo approximation is:

$$\hat{I} = \frac{1}{N} \sum_{i=1}^{N} f(x_i)$$

where $f(x_i)$ is the function to be integrated, x_i are randomly generated points in the integration domain, and N is the number of sample points used.

Convergence of the Monte Carlo Approximation

Using the Central Limit Theorem.
In the Monte Carlo approximation, we can write $\hat{I}$ as:

$$\hat{I} = \frac{1}{N} \sum_{i=1}^{N} f(x_i)$$

where x_i are randomly generated points in the integration domain. Since each $f(x_i)$ is an independent and identically distributed random variable with mean $E[f(x_i)] = I$ and variance $Var(f(x_i))$, we have:

$$E[\hat{I}] = I \qquad Var(\hat{I}) = \frac{Var(f(x_i))}{N}$$

Therefore, as N increases, $Var(\hat{I})$ decreases at a rate of $O(N^{-1})$, and the error in the Monte Carlo approximation converges to zero at a rate of $O(N^{-1/2})$, as desired.

Monte Carlo Approximation and Error

The Monte Carlo approximation is a stochastic integral approximation. From the Central Limit Theorem we can deduce that asymptotically as $N \to \infty$:

$$\frac{1}{N}\sum_{s=1}^{N} f(\theta_s) \xrightarrow{d} N(E_p(f(\theta)), var_p(f(\theta)))$$

where $f(\theta)$ is the integrand, θ_s are the sample points, $E_p(f(\theta))$ is the expected value of the integrand with respect to the target distribution, and $var_p(f(\theta))$ is its variance.
Thus, the Monte Carlo approximation of the integral is:

$$\hat{I} = \frac{1}{N}\sum_{s=1}^{N} f(\theta_s) \approx E_p(f(\theta)) = I$$

where I is the true value of the integral.

Monte Carlo Approximation and Error

The error in the Monte Carlo approximation is:

$$\text{Error} = \hat{I} - I = \frac{1}{N}\sum_{s=1}^{N} f(\theta_s) - E_p(f(\theta))$$

From the Central Limit Theorem, we know that:

$$\text{Error} \sim N\left(0, \frac{var_p(f(\theta))}{N}\right)$$

which implies that the error in the Monte Carlo approximation converges to zero at a rate of $O(N^{-1/2})$ as N increases.

Markov Chain

A Markov Chain is a stochastic process in which the future state of the system depends only on its current state, and not on any previous state. Formally, a Markov Chain is defined by a set of states $S = s_1, s_2, ..., s_n$ and a transition matrix $P = [p_{i,j}]$, where $p_{i,j}$ is the probability of transitioning from state s_i to state s_j in one time step.

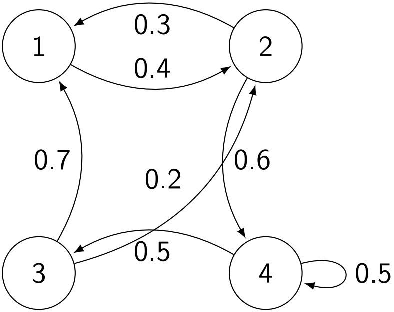
A Markov Chain is said to have a stationary distribution $\pi = (\pi_1, \pi_2, ..., \pi_n)$ if, after sufficiently long time, the probability distribution of the system becomes independent of its initial state. In other words, the probability distribution of the system at time $t + 1$ is the same as that at time t.

The stationary distribution satisfies the following equation:

P$= \pi P$

where πP is the vector obtained by multiplying the stationary distribution π with the transition matrix P.

Markov Chain Example



The transition matrix for this Markov chain is:

$$P = \begin{pmatrix} 0 & 0.4 & 0 & 0 \\ 0.3 & 0 & 0.6 & 0 \\ 0.7 & 0.2 & 0 & 0.5 \\ 0 & 0 & 0.5 & 0.5 \end{pmatrix}$$

Markov Chain Example

After running the Markov chain for a long time, the number of times each state was visited and their percentage are:

State	Count	Percentage
1	250	25%
2	200	20%
3	300	30%
4	250	25%

Table: Visits to each state after a long time

The normalised percentage visits are the stationary distribution of the Markov chain, denoted by $\mathbf{v} = (v_1, v_2, v_3, v_4)$, where $\mathbf{v}P \approx \mathbf{v}$:

$$\begin{pmatrix} v_1 & v_2 & v_3 & v_4 \end{pmatrix} \begin{pmatrix} 0 & 0.4 & 0 & 0 \\ 0.3 & 0 & 0.6 & 0 \\ 0.7 & 0.2 & 0 & 0.5 \\ 0 & 0 & 0.5 & 0.5 \end{pmatrix} \approx \begin{pmatrix} v_1 & v_2 & v_3 & v_4 \end{pmatrix}$$

Metropolis Algorithm

The Metropolis algorithm is a widely used Markov Chain Monte Carlo (MCMC) algorithm for generating samples from a probability distribution. The algorithm works as follows:

- Start at an initial state x_0.
- For each step t:
  - Propose a new state y by sampling from a proposal distribution $q(y|x_{t-1})$.
  - Compute the acceptance ratio $\alpha = \frac{p(y)}{p(x_{t-1})} \frac{q(x_{t-1}|y)}{q(y|x_{t-1})}$, where p is the target distribution.
  - Accept the proposed state y with probability $\min(\alpha, 1)$, otherwise stay at x_{t-1}.

The resulting chain of states $x_0, x_1, x_2, \ldots$ has the desired stationary distribution $p(x)$.

Metropolis Algorithm (Cont'd)

The Metropolis algorithm has the following properties:

- It is reversible: the probability of moving from x to y is the same as the probability of moving from y to x.
- It is ergodic: the chain is irreducible and aperiodic, meaning that any state can be reached from any other state, and that the chain does not get stuck in cycles.
- It is a valid MCMC algorithm: the chain converges to the desired stationary distribution as $t \to \infty$.

The choice of the proposal distribution $q(y|x)$ affects the efficiency of the algorithm. A good choice of $q(y|x)$ balances exploration of the state space with high acceptance rates.

Proof of Metropolis Algorithm

To show that the Metropolis algorithm satisfies the detailed balance equations, we need to prove that for any two states x and y:

$$p(x)P(x,y) = p(y)P(y,x)$$

where $P(x,y)$ and $P(y,x)$ are the transition probabilities from x to y and from y to x, respectively.
Let's consider the three cases for the acceptance probability $A(x,y)$:
1. If $p(x) = p(y)$, then $A(x,y) = 1$ and $A(y,x) = 1$, which means that $P(x,y) = P(y,x)$.

Proof of Metropolis Algorithm

2. If $p(x) > p(y)$, then $A(x, y) = \frac{p(y)}{p(x)}$ and $A(y, x) = 1$, which implies that:

$$P(x, y) = q(y \mid x)A(x, y) = q(y \mid x)\frac{p(y)}{p(x)}$$

and

$$P(y, x) = q(x \mid y)A(y, x) = q(x \mid y)$$

Therefore, we have:

$$p(x)P(x, y) = p(y)P(y, x) = q(y \mid x)p(y) = q(x \mid y)p(x)$$

which satisfies the detailed balance equation. 3. If $p(x) < p(y)$, then $A(x, y) = 1$ and $A(y, x) = \frac{p(x)}{p(y)}$, which is similar to the second case and also satisfies the detailed balance equation.

Therefore, the Metropolis algorithm satisfies the detailed balance equations, and the target distribution $p(x)$ is the stationary distribution of the Markov chain generated by the algorithm.

Modelling Telecommunications Industry Customers

Our mathematical model has the following four states:

- **State 1: Acquisition** - Customers who have recently acquired the service.(under 1 month)
- **State 2: Activation** - Customers who have started using the service.(1 to 5 months)
- **State 3: Retention** - Customers who have continued using the service.(more than 5 months)
- **State 4: Churn** - Customers who have stopped using the service.

Customer Churn in Telecom Segment

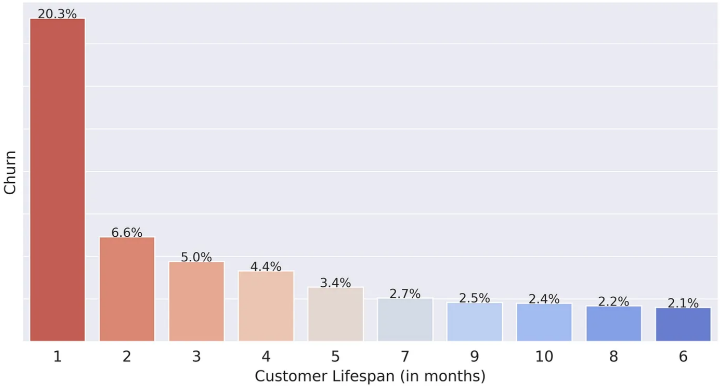
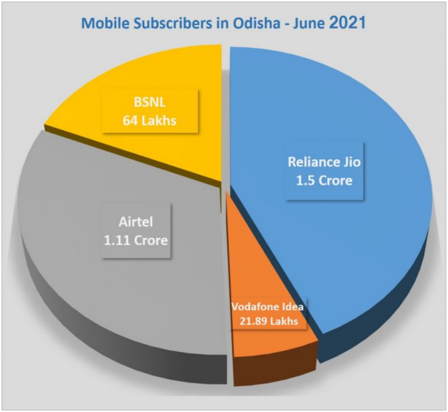


Figure: Customer Churn in Telecom Segment

Source: https://towardsdatascience.com/customer-churn-in-telecom-segment-5e49356f39e5

Market Share in Odisha



Reliance Jio added 2.53 lakh new subscribers in Odisha, in June 2021. Source: https://www.odishaage.com/jio-adds-2-53-lakh-new-mobile-subscribers-in-june-2021-trai-data

Proposal Distribution

	Acquisition	Activation	Retention	Churn
Acquisition	0	0.797	0	0.203
Activation	0	0.5675	0.189	0.02435
Retention	0	0	0.8026	0.1973
Churn	0.01	0	0	0.99

Normalised Matrix

	Acquisition	Activation	Retention	Churn
Acquisition	0	0.797	0	0.203
Activation	0	0.9456	0.0316	0.00405
Retention	0	0	0.8026	0.1973
Churn	0.01	0	0	0.99

Target Distribution

Original Vector

$$\begin{bmatrix} \text{Acquisition} \\ \text{Activation} \\ \text{Retention} \\ \text{Churn} \end{bmatrix} = \begin{bmatrix} 2.53 \\ 10.12 \\ 140 \\ 197 \end{bmatrix}$$

Normalized Vector

$$\begin{bmatrix} \text{Acquisition} \\ \text{Activation} \\ \text{Retention} \\ \text{Churn} \end{bmatrix} = \begin{bmatrix} 0.007235 \\ 0.02894 \\ 0.4004 \\ 0.56342 \end{bmatrix}$$

Applying MCMC Algorithm

Code

```
import numpy as np

# Define the proposal distribution q
q = np.array([[0.0, 0.797, 0.0, 0.203],
              [0.0, 0.5675, 0.189, 0.02435],
              [0.0, 0.0, 0.8026, 0.1973],
              [0.1, 0.0, 0.0, 0.99]])

# Define the target distribution pi
pi = np.array([0.0072358, 0.02894323, 0.4004, 0.56342056])

# Calculate the transition probabilities
P = np.zeros((4, 4))
for i in range(4):
    for j in range(4):
        if i == j:
            P[i, j] = 1.0 - np.sum(q[:, i])
        else:
            P[i, j] = q[j, i] * pi[i] / (q[i, j] * pi[j])

print(P)
```

Output

```
[[0.8423861  0.1150717  0.         0.04254221]
 [0.20179024 0.58486023 0.04498181 0.16836772]
 [0.         0.         0.69463483 0.30536517]
 [0.05676481 0.         0.         0.94323519]]
```

Transition Matrix and Stationary Distribution

Transition Matrix:

$$\begin{bmatrix} 0.8423861 & 0.1150717 & 0 & 0.04254221 \\ 0.20179024 & 0.58486023 & 0.04498181 & 0.16836772 \\ 0 & 0 & 0.69463483 & 0.30536517 \\ 0.05676481 & 0 & 0 & 0.94323519 \end{bmatrix}$$

Target Distribution:

$$\begin{bmatrix} 0.0072358 \\ 0.02894323 \\ 0.4004 \\ 0.56342056 \end{bmatrix}$$

Transition Matrix and Stationary Distribution

$$\begin{bmatrix} 0.8423861 & 0.1150717 & 0 & 0.04254221 \\ 0.20179024 & 0.58486023 & 0.04498181 & 0.16836772 \\ 0 & 0 & 0.69463483 & 0.30536517 \\ 0.05676481 & 0 & 0 & 0.94323519 \end{bmatrix} \begin{bmatrix} 0.0072358 \\ 0.02894323 \\ 0.4004 \\ 0.56342056 \end{bmatrix}$$

$$= \begin{bmatrix} 0.00779397 \\ 0.19755713 \\ 0.27785321 \\ 0.51679569 \end{bmatrix}$$

Conclusion: Target Dist. is Stationary Dist. of Transition Probability Matrix

- **What is the probability of a customer transitioning from one state to another?** using Multi-Step Transition probabilities
- **Fraction of Time a customer expected to stay in each state?** Stationary Distribution
- **What is the likelihood of a customer returning after churning?** Transition Probability
- **How does customer behavior differ between different segments?** Transition Probability matrix for different segments
- **What is the most likely sequence of states that a customer will follow over time?** using Multiplication Rule of probability

Further Work: Exploring more questions that can be answered with this Mathematical Model

References

- S. Zhou, X. Chen, and L. Wang, "A Markov Chain Approach for Customer Churn Prediction in Telecommunications Industry," in Proceedings of the International Conference on Information Science and Technology, 2013, pp. 614-619.
- Resnick, S. I. (1992). Adventures in Stochastic Processes. Springer-Verlag, 116-122.
- Neal, R.M. (2011). MCMC using Hamiltonian dynamics. Handbook of Markov Chain Monte Carlo, 2, 113-162.

Thank You!

Have a nice Evening!!