SOCIOECONOMIC ANALYTICS

Application of the Beta Distribution model to the Customer Churn Rate

Henrique Pinto dos Santos Zaidan [1 2]

[1] Postgraduate Program in Biometrics and Applied Statistics, Federal Rural University of Pernambuco, Recife, Brazil.
[2] Data Scientist at Desktop, Sumaré, São Paulo, Brazil
Email: henrique.zaidan@desktop.tec.br

How to cite this paper: Zaidan, H. P. S. (2023). Application of the Beta Distribution Model to the Customer Churn Rate. *Socioeconomic Analytics,* 1(1), 78-91. https://doi.org/10.51359/2965-4661.2023.259280

RESEARCH ARTICLE

Socioeconomic Analytics
https://periodicos.ufpe.br/revistas/SECAN/
ISSN Online: 2965-4661

Submitted on: 28.07.2023.
Accepted on: 07.08.2023.
Published on: 07.08.2023.

Abstract

The Beta Distribution model has been applied in many different research environments due to the flexibility of its two parameters. In this research, I fit this probabilistic model for modeling a recurring problem confronted by many businesses called the Customer Churn Rate. It represents the proportion of customers who cancel their subscriptions after a given time. I use data from a Brazilian media service company to develop the modeling. The parameters are estimated by the maximum likelihood estimation (MLE) technique. Finally, I perform the MLE technique by considering two programming languages: Ox and R.

Keywords

Beta Distribution; Costumer Churn Rate; Maximum Likelihood Estimation; Ox; R.

1. Introduction

In a subscription business model, an adverse effect for any subscription company is the departure of a customer. Thus, the development of metrics to analyze this phenomenon has received massive attention from companies. In this context, the concept of the customer churn rate is relevant to be discussed and analyzed. According to Hwang et al. (2004), the churn rate describes the percentage of subscribers who abandon a relationship with a service provider after a given time. It is computed as

$$C_r = \frac{nc_t}{N_t} \tag{1}$$

where C_r is the churn rate, nc_t is the number of churned customers in period t, and N_t is the total number of subscribers at the beginning of period t.

Customer churn can be split into two main groups, voluntary and non-voluntary (Hadden et al., 2007). When the customer is withdrawn by the company for reasons such as abuse of service, lack of payments, fraud, or similar, we have non-voluntary churn. On the other hand, if the customer decides to terminate their relationship with the service provider, we have voluntary churn. This one also divides into two types, incidental churn, and deliberative churn. Incidental churn refers to churn due to changes in circumstances that prevent the customer from continuing as a customer. Deliberative churn is when the consumer changes their service provider, choosing a competing company (Axelsson and Nostam, 2017). Figure 1 systematizes types of customer churn.

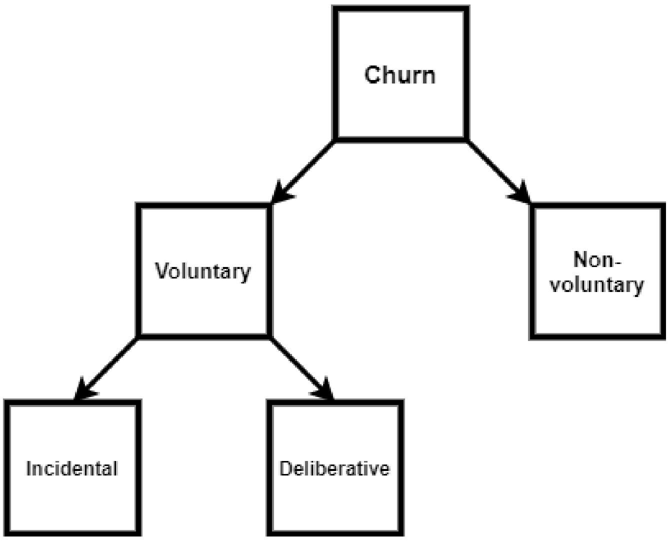

Figure 1. Types of Customer Churn.

Research into the churn rate has been developed under many different techniques. Decision trees, support vector machines, random forests, artificial neural networks, Markov chains, naive Bayes, and logistic regression were applied in the prediction of customer churning or customer classification (Hoppner et al., 2020; Yeon and Sehun, 2005; Burez and Van den Poel, 2007; Vafeiadis et al., 2015). However, it does not exist studies focused on the fit of the churn rate distribution. This study fills this gap through the utilization of the beta distribution model. Generally speaking, the beta distribution is a continuous probability distribution having two parameters. It applies to the modeling of random phenomena whose set of possible values is bounded into the interval (0, 1), such as percentages and proportions. Within this framework, the churn rate whose domain belongs to this interval can be modeled by the beta distribution.
In this paper, I use the maximum likelihood estimation (MLE) technique (Casella and Berger, 2002) for estimating the parameters by using customer churn rate data collected from a Brazilian media service company. Moreover, I provide standard errors and confidence intervals for each parameter. The implementation of the MLE technique is developed in two programming languages, Ox and R. Thus, the computational procedures for each programming language are discussed and compared.
The rest of this paper is organized as follows. The next section presents the features of the beta distribution model and some applications in different research environments. Section 3 describes the maximum likelihood estimation technique for the beta distribution model. Section 4 presents the data set, the results generated from the MLE for each programming language, and the computational implementation. Finally, I conclude the study and show some further extensions in section 5.

2. Beta Distribution Model

A random variable *x* is said to have a beta distribution if its density is given by:

$$f(x) = \begin{cases} \frac{1}{\beta(a,b)} x^{a-1}(1-x)^{b-1} \text{ if } x \in (0,1) \\ 0 \qquad \text{otherwise} \end{cases} \tag{2}$$

$$\beta(a,b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)},$$

where the beta function is denoted $\beta(a, b)$ and $\Gamma(.)$ is the gamma function. The shape parameters of the distribution are $a(>0)$ and $b(>0)$ (Krishnamoorthy, 2016).
The density of the beta distribution may have different shapes since its parameters can take a wide range of values, as shown in Figure 2. This distribution is:

(i) uniform if $a = b = 1$,
(ii) *U*-shape for $a < 1$ and $b < 1$
(iii) symmetrical for $a = b$
(iv) *J*-shape for $a > 1$ and $b < 1$
(v) reverse *J*-shape for $a < 1$ and $b > 1$
(vi) mount shape if $a > 1$ and $b > 1$

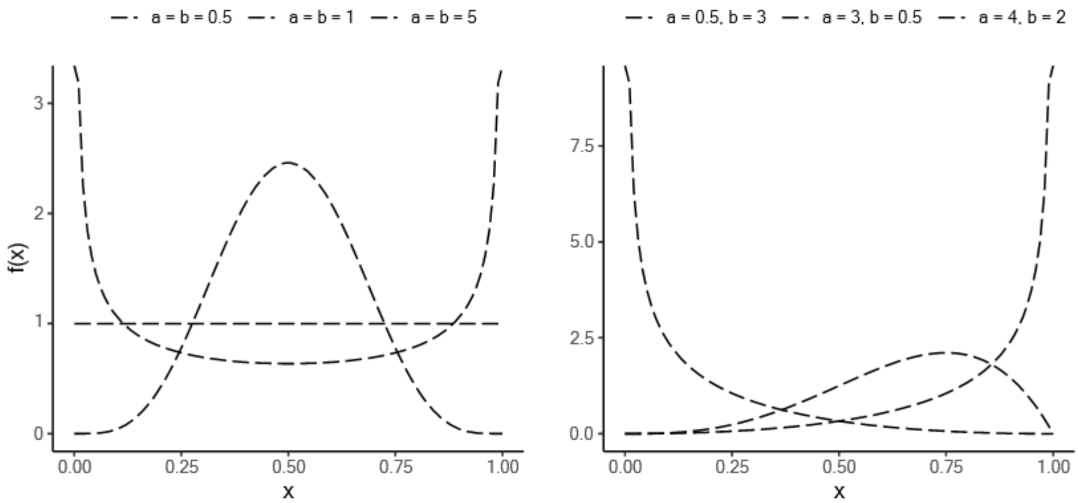

Figure 2. Probability Density Function of $\beta(a, b)$.

The expected value and variance are defined, respectively:

$$\mathbb{E}[X] = \frac{a}{a+b} \tag{3}$$

$$\text{var}[X] = \frac{ab}{(a+b)^2(a+b+1)} \tag{4}$$

Detailed information, special cases, generating pseudo-random numbers, and distributions derived from the beta distribution can be seen in Gupta and Nadarajah (2004) and Hung et al. (2009).

2.1. Review of previous works

Several works have employed the beta distribution to model many different types of phenomena whose random variable assumes values over the interval (0,1). Ferrari and Cribari-Neto (2004) developed a regression model when the dependent variable follows the beta distribution model. In finance, this distribution has been applied to measure the probability of payment or default in the credit granting decision (Bierman-Jr and Hausman, 1970). In management, the expected value of the four-parameter beta distribution is used to specify the project duration by considering three possible scenarios related to concluding of activities (Hajdu and Bokor, 2014; Golenko-Ginzburg, 1988; Malcolm et al., 1959).
Oguamanam et al. (1995) utilized the kurtosis of the beta distribution to predict the gear teeth condition. Lallemant and Kiremidjian (2015) applied the beta distribution for modeling the conditional probability of damage given ground-motion intensity using data from the Haiti 2010 earthquake. Other research applications for this probability model were directed in the meteorology field (Chia and Hutchinson, 1991; Sulaiman et al., 1999) and hydrological analysis (Bhunya et al., 2004; Jung et al., 2019; Seo and Baek, 2004). Finally, an application of the beta distribution for survival analysis was proposed by Fader and Hardie (2007). These authors fitted the shifted-beta-geometric distribution to provide a survivor function related to customer retention.

3. Parameter Estimation

The method applied to estimate the parameters is the maximum likelihood estimation. As seen in Owen (2008), this provides good performance and is commonly used for fitting the two-parameter beta distribution.
The basic idea of the MLE technique is on assuming a statistical model parametrized by a fixed and unknown θ (parameter or vector of parameters) and Θ (parameter space), the likelihood L(θ) is the probability of the observed data x considered as a function of θ (Pawitan, 2001). This method captures all the information in the data yielding a point estimate for θ. The properties of its estimator ($\hat{\theta}$) are summarized as follows:

- $\hat{\theta}$ is an asymptotically unbiased estimator of θ;
- $\hat{\theta}$ is consistent for θ;
- $\hat{\theta}$ is the asymptotically efficient estimator of θ;
- When the sample size is large, $\hat{\theta} \xrightarrow{d} N\left(\theta, K^{-1}(\theta)\right)$. $K(\theta)$ is the Fisher information matrix.

The relevant assumptions for maximum likelihood inference can be found in Gauss and Cribari-Neto (2014).

3.1. MLE for the Beta Distribution

Let $x_1, x_2, \ldots, x_n$ be n independent random variables from the beta distribution with parameters a and b, then the likelihood function associated is:

$$\boldsymbol{L(a,b|x)} = \left(\frac{\boldsymbol{\Gamma(a)\Gamma(b)}}{\boldsymbol{\Gamma(a+b)}}\right)^{\boldsymbol{n}} \times \prod_{\boldsymbol{i=1}}^{\boldsymbol{n}} (\boldsymbol{x_i})^{\boldsymbol{a-1}} (\boldsymbol{1-x_i})^{\boldsymbol{b-1}} \quad (5)$$

Through Eq.(5), the log-likelihood function is:

$$\log L(a, b|x) = n\log(\Gamma(a+b)) - n\log(\Gamma(a)) - n\log(\Gamma(b)) + (a-1)\sum_{i=1}^{n}\log(x_i) + (b-1)\sum_{i=1}^{n}\log(1-x_i) \quad (6)$$

Taking derivates of Eq.(6) for each parameter, the score functions are:

$$S_1(a, b) = \frac{\partial}{\partial a}\log L(a, b|x) = n\psi(a+b) - n\psi(a) + \sum_{i=1}^{n}\log(x_i) \quad (7)$$

$$S_2(a, b) = \frac{\partial}{\partial b}\log L(a, b|x) = n\psi(a+b) - n\psi(b) + \sum_{i=1}^{n}\log(1-x_i) \quad (8)$$

where $\psi(.)$ is the digamma function that is defined as the logarithmic derivative of the gamma function:

$$\psi(h) = \frac{\partial}{\partial h}\log(\Gamma(h)) \quad (9)$$

To find $\hat{a}$ and $\hat{b}$, the MLE estimators, the score functions are set equal to zero. Hence, the solution is $S_1(a, b) = 0$ and $S_2(a, b) = 0$.
Taking further derivates of Eqs. (7) and (8), given the score functions, are twice differentiable to a and b, the observed Fisher information matrix $K(a, b)$ is defined by

$$K(a, b) = \begin{bmatrix} n\psi_1(a) - n\psi_1(a+b) & -n\psi_1(a+b) \\ -n\psi_1(a+b) & n\psi_1(b) - n\psi_1(a+b) \end{bmatrix} \quad (10)$$

where $\psi_1(.)$ is the trigamma function that is defined as the second logarithmic derivative of the gamma function:

$$\psi_1(h) = \frac{\partial}{\partial h}\psi(h) \quad (11)$$

Finally, since there is no closed-form solution to find $\hat{a}$ and $\hat{b}$, it is necessary to betake numerical optimization. I apply a Quasi-Newton method called BFGS (and its variant called L-BFGS-B), an algorithm that aims to find the local extrema of functions through the estimation of the inverse of the Hessian matrix (Nocedal and Wright, 2006). An advantage of the BFGS method is that it allows users to maximize likelihoods without having to specify a score function (Cribari-Neto and Zarkos, 2003).

4. Empirical Churn Rate Data

The customer churn data used in this study come from a Brazilian subscription company focused on digital services. The database constitutes a monthly time series, started in January 2019, and the last observation is in August 2020. As a result, we have twenty data. The range is fairly compacted; the minimum value is 0.075, while the maximum is 0.1190. Table 1 shows the summary statistics of the set of observations while Figure 3 presents data visualization under three perspectives (histogram, boxplot, and time series plot).

Table 1. Descriptive Statistics for the Customer Churn Rate.

Mean	**Median**	**Standard Deviation**	**Coefficient of Variation**	**Skewness**	**Kurtosis**
0.1026	0.1015	0.0110	0.1069	-0.4183	-0.1320

The result of the coefficient indicates a low variability of the data set, and the values of the sample mean, and median are almost the same. The sample skewness and the sample kurtosis are close to the normal distribution.

The boxplot shows that the set of observations does not present extreme values. The first quantile is 0.0950, and the third is 0.1115. The dashed line of the time series plot is the sample mean where nine observations are above the mean. Finally, the histogram exhibits that most of the data is concentrated between 0.09 and 0.11.

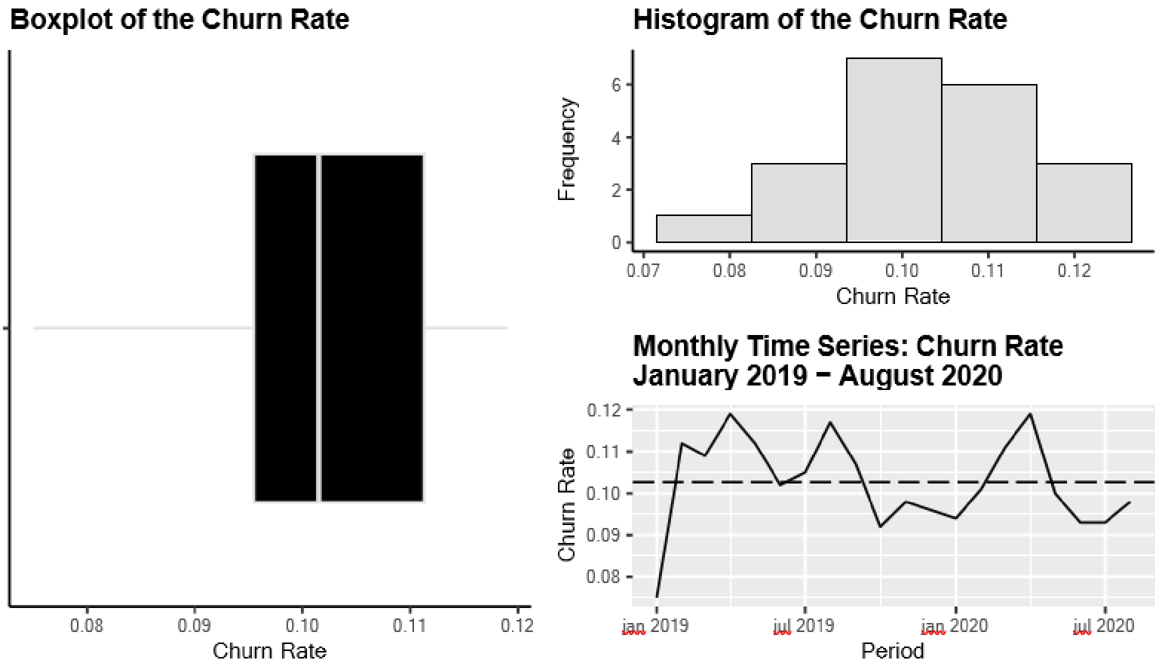

Figure 3. Data Visualization of the Churn Rate.

4.1. Analysis of the MLE technique

In the execution of the MLE technique for both programming languages, I use the same starting values ($\hat{a} = \hat{b} = 5$), the BFGS method, and the analytical gradient for the score functions and Hessian matrix. In R, convergence problems happened by using the BFGS method. Consequently, it was replaced by the L-BFGS-B, whose lower bounds were equal to 0.01. Moreover, in Ox, the numerical gradient for the Hessian matrix did not provide accurate values for the standard errors.

Table 2 shows the results of the estimation. Both programming languages generated the same estimates and standard errors.

Table 2. Results of the Parameter Estimates for the Beta Distribution.

Measures	$\hat{a}$	$\hat{b}$
R – Estimate	79.2383	692.7116

R – Standard Error	25.0051	219.2103
Ox – Estimate	79.2383	692.7116
Ox – Standard Error	25.0051	219.2103

Based on the values of the estimates of $\hat{\boldsymbol{a}}$ and $\hat{\boldsymbol{b}}$, the value of the maximized log-likelihood is 62.031 and the observed Fisher information matrix evaluated in the estimates is

$$\boldsymbol{K}(\hat{\boldsymbol{a}}, \hat{\boldsymbol{b}}) = \begin{bmatrix} \mathbf{0.22808} & \mathbf{-0.025925} \\ \mathbf{-0.025925} & \mathbf{0.0029677} \end{bmatrix}$$

The estimates of the mean, variance, and standard deviation of the beta distribution are 0.1026469, 0.0001191676, and 0.01091639, respectively. Furthermore, Table 3 shows the confidence intervals for each parameter considering a 95% confidence level.

Table 3. Confidence Interval for each Parameter - 95% Confidence Level.

Parameter	Lower endpoint	Upper endpoint
a	38.1086	120.3681
b	332.1428	1053.281

Figure 4 portrays the estimate of the probability density function based on the values of the parameter estimates yielded by the MLE technique. Since both estimates are superior to 1, then the density function curve is in the mount shape.

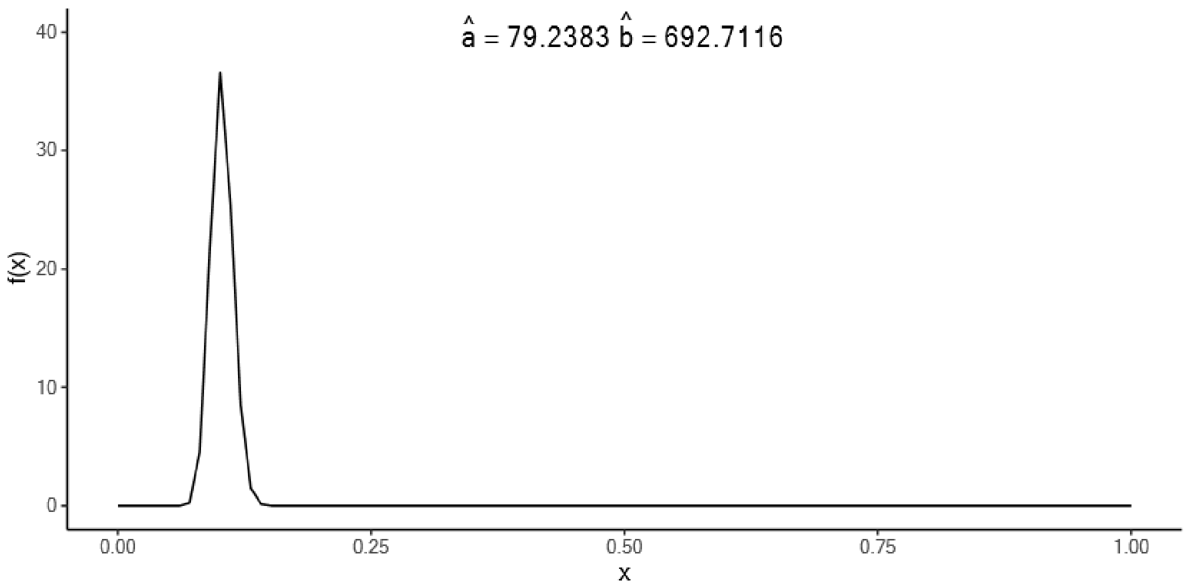

Figure 4. Estimated Probability Density Function of the Beta Distribution.

The estimated probability density function specifies that the probability of the churn rate falling within a particular range of values above 0.13 or below 0.07 tends to zero, reflecting the features of the data set.

4.1.1. *Application in Ox*

Ox is a matrix programming language developed by Jurgen Doornik, having a similar syntax to C, and holds a maximization package (called `maximize`) useful to maximize functions of many parameters (Podivinsky, 1999). The function that performs BFGS is called `MaxBFGS` which must be written within the `main function`.

`MaxBFGS` holds five arguments: 1 - the function that will be optimized, 2 - the matrix with starting values, 3 - the final function value, 4 - 0 (standard argument related to the Hessian matrix), 5 - TRUE or FALSE, depending on the type of gradient (analytical - FALSE and numerical - TRUE).

The structure of the maximization of the log-likelihood function divides into three steps:

(1) Import the libraries and packages.
(2) Write the log-likelihood function and the matrices of partial derivatives (first order).
(3) Call the `MaxBFGS` function.

Step 1: Libraries and Packages required.

```
#include <oxstd.h>    //standard library header
#include <oxprob.oxh> //probability library
#import <maximize>    //optimization package
```

Step 2: Write the log-likelihood function and the matrices of partial derivates in Ox syntax.

```
fllbeta(const vP, const adFunc, const avScore, const amHess)
{
    decl a = vP[0]; //first parameter
    decl b = vP[1]; //second parameter
    decl nobs = rows(churn); //number of observations
    //churn is the set of observations (vector)

    //log likelihood function
    adFunc[0] = nobs * loggamma(a+b) - nobs *loggamma(a) - nobs * loggamma(b)
    + (a-1) * sumc(log(churn)) + (b-1) * sumc(log(1 - churn));

    //score function
    if(avScore){
                (avScore[0])[0] = nobs*polygamma(a + b, 0) - nobs*polygamma(a,0)
                + sumc(log(churn));
                (avScore[0])[1] = nobs*polygamma(a + b, 0) - nobs*polygamma(b,0)
                + sumc(log(1.0-churn));
    }

    return 1; // 1 indicates success
}
```

`fllbeta` is the function that will be maximized by the `MaxBFGS` function. This function must contain four elements:

(1) `Vp` is a 2 x 1 matrix of parameter values at which the function is to be evaluated;
(2) `adFunc` is the log-likelihood function of the beta distribution - Eq.(6);
(3) `avScore` contains the analytical first derivatives of the log-likelihood function - Eq.(7) and Eq.(8);
(4) `amHess` always 0 for `MaxBFGS`, as it does not need the Hessi

Step 3: Call the MaxBFGS function.

```
main()
{
    decl vp, dfunc, ir, vep;
    vp = <5; 5>; //starting values
    MaxControl(-1,1); //number of interations
    ir = MaxBFGS(fllbeta, &vp, &dfunc, 0, FALSE); //BFGS method
    //Results obtained:
    println("\nCONVERGENCE: ", MaxConvergenceMsg(ir));
    println("\nMaximized log-likelihood: ", "%7.3f", double(dfunc));
    println("\nML estimates: ", "%13.4f", vp' );
}
```

The outputs generated from `MaxBFGS` are the type of convergence (I found strong convergence), the maximum value of the log-likelihood function, and the estimates.
Through a call called `MaxControl(-1, 1)`, it is possible to see the result of the value of the log-likelihood for each interaction (Figure 5). The program implemented 38th interactions. After the 20th interaction, the log-likelihood value reached 62.

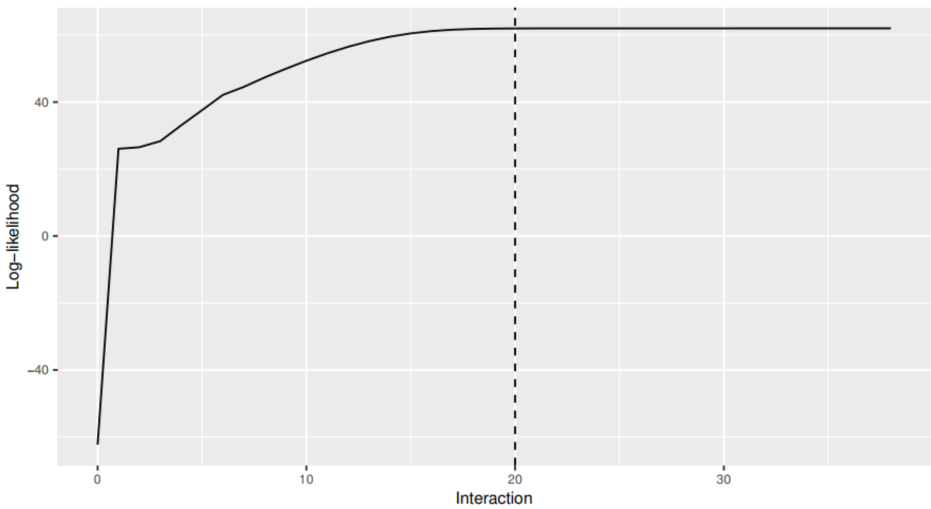

Figure 5. The Results of the Log-likelihood Value for Each Interaction.

Finally, the code below explains to process to implement the analytical gradient to find Fisher's information matrix and standard errors for each parameter.

```
//standard errors obtained from Fisher’ s information matrix
decl nobs = rows(churn);
decl K = zeros(2,2), Kinv;
//K is the Fisher’ s information matrix
//Kinv is the inverse matrix of K
K[0][0] = double(nobs*polygamma(vp[0], 1) - nobs*polygamma(vp[0]+vp[1], 1));
K[0][1] = K[1][0] = double(-nobs*polygamma(vp[0]+vp[1], 1));
K[1][1] = double(nobs*polygamma(vp[1], 1) - nobs*polygamma(vp[0]+vp[1], 1));
println("Fisher’ s information matrix = ", K);
Kinv = invert(K);
println("\nInverse Fisher’ s information matrix = ", Kinv); print("\n");
println("Standard error of \hat{a}: ", "%8.4f", sqrt(Kinv[0][0]));
println("Standard error of \hat{b}: ", "%8.4f", sqrt(Kinv[1][1]));
```

In a nutshell, the process is to set a matrix and fill it with the elements from the observed information matrix (the negative of the second derivative). After, this matrix is inverted, and take the root square of elements from the main diagonal from the inverted matrix.

4.1.2. Application in R

R is a language and environment for statistical computing and graphics, like the S language (Venables and Ripley (2002)). It contains several functions and libraries related to numerical methods, among which the `optim` function brings the BFGS and L-BFGS-B methods, among others.

Estimating likelihood functions entails a two-step process. First, one declares the log-likelihood function and the score functions. Then one optimizes the log-likelihood function via `optim` by selecting a numerical method.

Step 1: Declaring the beta log-likelihood function and the score functions.

```
#Log-likelihood function
logLikBeta <- function(theta){
    a = theta[1];    #first parameter
    b = theta[2];    #second paramenter

    loglik = sum(dbeta(y, a, b, log=TRUE))
    #y is the set of observations (vector)
    return(loglik)}

#Score functions (scorefn) - first-order partial derivatives
scorefn<- function(theta){
    a = theta[1];    #first parameter
    b = theta[2];    #second paramenter

    cbind(n*digamma(a+b) - n*digamma(a) + sum(log(y)),
        n*digamma(a+b) - n*digamma(b) + sum(log(1-y)))
    #n is the number of observations
}
```

`loglik` represents the Eq.(6) and `scorefn` combines the score functions (Eq.(7) and Eq.(8)) in a vector. The term `logLikBeta` is the name of the log-likelihood function; this name will be used `optim`. theta is the 2 x 1 matrix of parameter values similar to `vP` from `fllbeta` of the Ox language.

Step 2: Call the L-BFGS-B method from the `optim` function.

```
r2<- optim(c(5,5), logLikBeta, method="L-BFGS-B",
    lower=c(0.01, 0.01),
    control=list(fnscale=-1), hessian = F, gr = scorefn)
```

`r2` is the object that stores all information associated with the maximization process, including the estimates, the maximum value of the log-likelihood function, and other information. `c(5,5)` is the vector with starting values, `lower` specifies the lower limits for each parameter, this bound is connected with the L-BFGS-B method. Finally, we mention that the `optim` function is based on the minimization. Thus, to perform the maximization is necessary to add the following argument `control=list(fnscale=-1)`.
`hessian` indicates `T` or `F`. If it is `T`, then it returns a numerical Hessian matrix.
`gr` indicates the score functions. If it is `NULL`, then a finite-difference approximation will be used.
The output produced by `r2` is seen below. It consists of five elements.

```
r2$par   #Estimates of the parameters

r2$value #Final value of the log-likelihood function

r2$counts #Vector that reports the number of calls to
          #the log-likelihood function and the gradient

r2$convergence #Value of 0 indicates normal convergence

r2$message #This shows warnings of any problems
           #that occurred during optimization
```

Finally, the code to find the standard errors through the analytical gradient is described below. It follows the same steps as in the Ox language.

```
#Step 1 - Fisher’ s Information Matrix
FM<-matrix(1:4,nrow = 2, ncol = 2)
FM[1,1] = (n * trigamma(est[1])) - (n * trigamma(est[1]+est[2]))
FM[2,2] = (n * trigamma(est[2])) - (n * trigamma(est[1]+est[2]))
FM[1,2] = FM[2,1] = ((-1) * n * trigamma(est[1]+est[2]))
#Fisher Matrix
FM
#Step 2 - Inverse of the observed Fisher information
IFM<- solve(FM)
IFM
#Step 3 - Compute the standard errors
se<-sqrt(diag(IFM))
###Standard error output
se
```

5. Concluding Remarks

This study presented the customer churn rate from a different perspective considering the beta distribution model. I perform its estimation via MLE in two programming languages, R and Ox. Both languages provide the same results.
The study can be extended in several ways. One can develop confidence intervals under the bootstrap technique (Efron and Tibshirani, 1986) and compare it with the classical approach. Further, the bootstrap technique also can be used for bias correction, as seen in Gauss and Cribari- Neto (2014). Finally, we could compare the beta distribution model with other probability models whose domain lies between 0 and 1 to see which model fits better via goodness of fit tests.

Conflict of Interest Declaration

The author has no conflict of interest to declare and there is no financial interest to report.

References

1. Axelsson, R. Nostam, A. (2017). Identify churn a study in how transaction data can be used to identify churn for merchants. Master’s thesis, KTH, School of Industrial Engineering and Management (ITM).
2. Bhunya, P., Mishra, S., Ojha, C., Berndtsson, R. (2004). Parameter estimation of beta distribution for unit hydrograph derivation. Journal of Hydrologic Engineering, 9(4), 325–332. https://doi.org/10.1061/(ASCE)1084-0699(2004)9:4(325)
3. Bierman-Jr, H., Hausman, W. H. (1970). The credit granting decision. Management Science, 16(8), B519–B532.
4. Burez, J., Van den Poel, D. (2007). CRM at a pay-TV company: Using analytical models to reduce customer attrition by targeted marketing for subscription services. Expert Systems with Applications, 32(2), 277–288. https://doi.org/10.1016/j.eswa.2005.11.037

5. Casella, G., Berger, R. (2002). Statistical Inference. Vol. 2, Duxbury Pacific Grove, California.
6. Chia, E., Hutchinson, M. (1991). The beta distribution as a probability model for daily cloud duration. Agricultural and Forest Meteorology, 56(3), 195–208. https://doi.org/10.1016/0168-1923(91)90091-4
7. Cribari-Neto, F., Zarkos, S.G. (2003). Econometric and statistical computing using Ox. ComputationalEconomics, 21(3), 277–295. https://doi.org/10.1023/A:1023902027800
8. Efron, B., Tibshirani, R. (1986). Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy. Statistical Science, 1(1), 54–75. https://doi.org/10.1214/ss/ 1177013815
9. Fader, P., Hardie, B. (2007). How to project customer retention. Journal of Interactive Marketing, 21(1), 76–90. https://doi.org/10.1002/dir.20074
10. Ferrari, S., Cribari-Neto, F. (2004). Beta regression for modelling rates and proportions. Journal of Applied Statistics, 31(7), 799–815. https://doi.org/10.1080/0266476042000214501
11. Gauss, C., Cribari-Neto, F. (2014). An Introduction to Bartlett Correction and Bias Reduction. Vol. 1, Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-55255-7
12. Golenko-Ginzburg, D. (1988). On the distribution of activity time in pert. The Journal of the Operational Research Society, 39(8), 767–771. https://doi.org/10.1057/jors.1988.132
13. Gupta, A., Nadarajah, S. (2004). Handbook of Beta Distribution and Its Applications. CRC Press, Boca Raton. https://doi.org/10.1201/9781482276596
14. Hadden, J., Tiwari, A., Roy, R., Ruta, D. (2007). Computer assisted customer churn management: State-of-the-art and future trends. Computers & Operations Research, 34(10), 2902–2917. https://doi.org/10.1016/j.cor.2005.11.007
15. Hajdu, M., Bokor, O. (2014). The effects of different activity distributions on project duration in pert networks. Procedia - Social and Behavioral Sciences, 119, 766–775. https://doi.org/10.1016/j.sbspro.2014.03.086
16. Hoppner, S., Stripling, E., Baesens, B., vanden Broucke, S., Verdonck, T. (2020). Profit driven decision trees for churn prediction. European Journal of Operational Research, 284(3), 920 – 933. https://doi.org/10.1016/j.ejor.2018.11.072
17. Hung, Y., Balakrishnan, N., Lin, Y. (2009). Evaluation of beta generation algorithms. Communications in Statistics - Simulation and Computation, 38(4), 750–770. https://doi.org/10.1080/03610910802645347
18. H. Hwang, T. Jung, and E. Suh. An LTV model and customer segmentation based on customer value: a case study on the wireless telecommunication industry. Expert Systems with Applications, 26(2), 181–188. https://doi.org/10.1016/S0957-4174(03)00133-7
19. Jung, K., Shin, J., Park, D. (2019). A new approach for river network classification based on the beta distribution of tributary junction angles. Journal of Hydrology, 572, 66–74. https://doi.org/10.1016/j.jhydrol.2019.02.041

20. Krishnamoorthy, K. (2016). Handbook of statistical distributions with applications. CRC Press, New York. https://doi.org/10.1201/b19191

21. Lallemant, D., Kiremidjian, A. (2015). A beta distribution model for characterizing earthquake damage state distribution. Earthquake Spectra, 31(3), 1337–1352. https://doi.org/10.1193/012413EQS013M

22. Malcolm, D.G., Roseboom, J., Clark, C. E., Fazar, W. (1959). Application of a technique for researchand development program evaluation. *Operations Research*, 7(5), 646–669. https://doi.org/10.1287/opre.7.5.646

23. Nocedal, J., Wright, S. (2006). Numerical Optimization. Vol. 1. Springer New York. https://doi.org/10.1007/978-0-387-40065-5

24. Oguamanam, D., Martin, H., Huissoon, J. (1995). On the application of the beta distribution to geardamage analysis. Applied Acoustics, 45(3), 247–261.

25. Owen, C. (2008). Parameter estimation for the beta distribution. Master's thesis, Statistics Department.

26. Pawitan, P. In all likelihood: statistical modelling and inference using likelihood. Vol. 1, OxfordUniversity Press, Oxford.

27. Podivinsky, J. M. (1999). Ox 2.10: beast of burden or object of desire? Journal of Economic Surveys, 13(4), 491–502. https://doi.org/10.1111/1467-6419.00094

28. Seo, I.W., Baek, K.O. (2004). Estimation of the longitudinal dispersion coefficient using the velocity profile in natural streams. Journal of Hydraulic Engineering, 130(3), 227–236. https://doi.org/10.1061/(ASCE)0733-9429(2004)130:3(227)

29. Sulaiman, M.Y., Hlaing Oo, W.M., Wahab, M.A., Zakaria, A. (1999). Application of beta distribution model to Malaysian sunshine data. Renewable Energy, 18(4), 573–579. https://doi.org/10.1016/S0960-1481(99)00002-6

30. Vafeiadis, T., Diamantaras, K., Sarigiannidis, G., Chatzisavvas, K. (2015). A comparison of machine learning techniques for customer churn prediction. Simulation Modelling Practice and Theory, 55, 1–9. https://doi.org/10.1016/j.simpat.2015.03.003

31. Venables, W., Ripley, B. (2002). Modern Applied Statistics with S. Vol. 1, 4. Ed., Springer New York. https://doi.org/10.1007/978-0-387-21706-2

32. Hur, Y., Lim, S. (2005). Customer churning prediction using support vector machines in online auto insurance service. In: Advances in Neural Networks – ISNN 2005, pages 928–933. Springer Berlin Heidelberg. https://doi.org/10.1007/11427445_149