Bayesian Inference in the Multinomial Probit Model: A case study

Oleksandr Dyklevych, 82/02/21
Abstract

Binary, multinomial and multivariate response models play an important role in the variety of scientific fields. The most commonly encountered response models are Multinomial Logit model (MNL) and Multinomial Probit model (MNP). Being easier to deal with, MNL suffers from an independence of irrelevant alternatives assumption (IIA). MNP model does not have IIA issue, but imposes its own difficulties. A direct estimation of the coefficients in case of MNP model is computationally burdensome. Bayesian approach allows to avoid complex calculations of the maximum likelihood function for MNP model, but it should be applied with caution because of the identification problem. Response models are often applied for prediction of the customers behavior and preferences. The paper is also concerned with using MNP model for prediction of customers retention behavior and lifetime value on the individual level. A Bayesian approach is used to estimate the parameter of interest and models’ adequacy is tested. The results are discussed, leaving some space for future work.
# Contents

1 Introduction 6  
   1.1 Motivation of the model selection . . . . . . . . . . 7  
   1.2 Outline . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8  

2 Theoretical concepts 9  
   2.1 Setting up the model . . . . . . . . . . . . . . . . . . . . . . 9  
   2.2 Bayesian framework . . . . . . . . . . . . . . . . . . . . . . 13  
      2.2.1 Binary case . . . . . . . . . . . . . . . . . . . . . . . 13  
      2.2.2 Multinomial case . . . . . . . . . . . . . . . . . . . . . 14  

3 Model estimation and inference: binary case 16  
   3.1 Target population . . . . . . . . . . . . . . . . . . . . . . . 16  
   3.2 Dataset description . . . . . . . . . . . . . . . . . . . . . . 18  
   3.3 Model estimation results . . . . . . . . . . . . . . . . . . . . 28  

4 Model estimation and inference: multinomial case 36  
   4.1 Target population . . . . . . . . . . . . . . . . . . . . . . . 36  
   4.2 Dataset description . . . . . . . . . . . . . . . . . . . . . . 37  
   4.3 Model estimation results . . . . . . . . . . . . . . . . . . . . 40  
   4.4 Applying results for CLV calculation . . . . . . . . . . . . . 53  

5 Conclusion 57  
   5.1 Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 57  
   5.2 Extensions, applications, and future work . . . . . . . . . . . 58  

A Posterior distribution of beta, binary case 60  

B Posterior distribution of beta, Sigma, multinomial case 62  

C Poisson-Binomial distribution 65  

D Source code documentation 68
List of Tables

<table>
<thead>
<tr>
<th>No.</th>
<th>Table Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Customers related covariates</td>
<td>19</td>
</tr>
<tr>
<td>3.2</td>
<td>Product related covariates</td>
<td>24</td>
</tr>
<tr>
<td>3.3</td>
<td>The correlation matrix for variables $LAM$, $LF$, $LT$, and $LI$</td>
<td>24</td>
</tr>
<tr>
<td>3.4</td>
<td>The summary table of covariates, model 1</td>
<td>29</td>
</tr>
<tr>
<td>3.5</td>
<td>Posterior results for model 1</td>
<td>30</td>
</tr>
<tr>
<td>3.6</td>
<td>Misclassification table for model 1</td>
<td>34</td>
</tr>
<tr>
<td>4.1</td>
<td>The distribution of the outcomes for model 2</td>
<td>37</td>
</tr>
<tr>
<td>4.2</td>
<td>The summary table of covariates, model 2</td>
<td>40</td>
</tr>
<tr>
<td>4.3</td>
<td>Posterior results for model 2, $\beta$</td>
<td>42</td>
</tr>
<tr>
<td>4.4</td>
<td>Comparison of observed and estimated frequencies, model 2</td>
<td>52</td>
</tr>
<tr>
<td>4.5</td>
<td>Posterior predictive probabilities example, model 2</td>
<td>54</td>
</tr>
</tbody>
</table>
List of Figures

3.1 Imaginary time-lines for customers, model 1. 18
3.2 Contingency table and mosaic plot for variable GEN, model 1. 20
3.3 Contingency table and mosaic plot for (grouped) variable AGE, model 1. 21
3.4 Contingency table and mosaic plot for variable ZIP, model 1. 22
3.5 Contingency table and mosaic plot for variable DP, model 1. 22
3.6 Contingency table and mosaic plot for variable DW, model 1. 23
3.7 Contingency table and mosaic plot for variable PU, model 1. 23
3.8 Contingency table and mosaic plot for (grouped) variable LAM, model 1. 25
3.9 Contingency table and mosaic plot for variable LF, model 1. 26
3.10 Contingency table and mosaic plot for variable ZF, model 1. 27
3.11 Contingency table and mosaic plot for variable DS, model 1. 28
3.12 Posterior densities for $\beta$, model 1. 31
3.13 Sampled output for $\beta$, model 1. 32
3.14 Autocorrelation plots of sampled output for $\beta$, model 1. 33
4.1 Contingency table and mosaic plot for variable GEN, model 2. 38
4.2 Contingency tables for variables AGE and ZIP, model 2. 38
4.3 Contingency tables for variables DP and DW, model 2. 39
4.4 Contingency tables for variables PU and LAM, model 2. 39
4.5 Contingency tables for variables ZF and DS, model 2. 39
4.6 Contingency tables for variable LF, model 2. 40
4.7 Posterior densities for $\tilde{\beta}$, model 2, (part 1). 43
4.8 Posterior densities for $\tilde{\beta}$, model 2, (part 2). 44
4.9 Posterior densities for $\tilde{\beta}$, model 2, (part 3). 45
4.10 Sampled output for $\tilde{\beta}$, model 2, (part 1). 46
4.11 Sampled output for $\tilde{\beta}$, model 2, (part 2). 47
4.12 Sampled output for $\tilde{\beta}$, model 2, (part 3). 48
4.13 Autocorrelation plots of output for $\tilde{\beta}$, model 2, (part 1). 49
4.14 Autocorrelation plots of output for $\tilde{\beta}$, model 2, (part 2) . . . 50
4.15 Autocorrelation plots of output for $\tilde{\beta}$, model 2, (part 3) . . . 51
4.16 Dependence of CLV on LAM . . . . . . . . . . . . . . . . . . . . . . . . . 55
# Listings

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>D.1</td>
<td>Source C-code for MNPBayes.dll ............... 69</td>
</tr>
<tr>
<td>D.2</td>
<td>Source R-code for bp.bayes function ............ 71</td>
</tr>
<tr>
<td>D.3</td>
<td>Source R-code for mnp.bayes function ............ 72</td>
</tr>
<tr>
<td>D.4</td>
<td>Source R-code for data.reshape function ........ 74</td>
</tr>
<tr>
<td>D.5</td>
<td>Source R-code for bp.bayes.predict function ........ 76</td>
</tr>
<tr>
<td>D.6</td>
<td>Source R-code for mnp.bayes.predict function ........ 77</td>
</tr>
<tr>
<td>D.7</td>
<td>Source R-code for tab.data function ............ 78</td>
</tr>
<tr>
<td>D.8</td>
<td>Source C-code for poissbin.dll .................. 79</td>
</tr>
<tr>
<td>D.9</td>
<td>Source R-code for poiss.binom.pmf function ....... 80</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

Multinomial Probit and Multinomial Logit (hereinafter, MNP and MNL respectively) represent extensions to a simple linear regression model. They both target the similar set of problems — prediction of the individual probabilities of a certain outcome. Most of the readers must be familiar with the simplest special cases of MNL and MNP, namely, binary logit model, which is often called logistic regression, and binary probit model. A variable of interest in such binary models can take two possible values, or, in other words, we observe two possible outcomes for a dependent variable. By making an assumption that the outcomes are influenced by the set of regressors, the probabilities of the outcomes are estimated by means of maximum likelihood estimation.

When there are more than two possible outcomes observed for a response variable, the situation is modeled by means of MNL or MNP. As usually, the outcomes are unordered, i.e. the order of the outcomes does not matter. For example, consider the following problem: there are four options to get from one city to another — by car, by bus, by train, or by plane. Definitely, different people prefer different types of transportation, and the choice is influenced by many factors: price, travel time, comfort, etc. We can arrange the outcomes by 24 different ways, but it does not affect the estimation process anyway. In many experiments, however, the order of the outcomes does matter. To deal with ordered outcomes, one shall use the other extensions of the binary outcome models, namely, Ordered Logit and Ordered Probit regressions.

Multinomial models shall not be mixed with multivariate models. Multivariate regression applies to the cases when one wants to estimate several binary models jointly, assuming that their outcomes are correlated.

As it was said, both MNP and MNL target the same set of problems. Nonetheless, there are several fundamental differences between them. MNL
Chapter 1. Introduction

relies on the so-called *independence of the irrelevant alternatives* assumption (hereinafter, IIA), which states that relative probability of each outcome does not depend on the presence (or absence) of the other alternatives. In many cases such assumption sounds unrealistic. That said, the formulation of the MNP assumes the correlation between outcomes. Apart from MNP, there are another classes of the choice models, which were developed to solve the IIA problem. Some of the most renowned such models are Nested Logit and Mixed Logit.

On the other hand, MNP imposes its own limitations. The estimation of the model coefficient is computationally burdensome, when the number of the outcomes for the variable of interest is greater than 3. In this case, the expression for the maximum likelihood function has no closed form, and cannot be solved directly. The Bayesian approach allows to estimate the coefficients of the model and avoid the complex computation of the maximum likelihood. But the application of the Bayesian technique is restricted by another issue, namely, *multiplicative redundancy*. A more detailed explanation of the problems and the ways to deal with them are given in chapter 2.

1.1 Motivation of the model selection

The main goal of the paper is to explore implementation, performance and adequacy of the Bayesian approach in the real-world application by modeling the retention behavior of the customers.

In a service-based environment, establishing long-term relationships with customers plays vital role in maximizing company’s profit. A quantitative measure of the efficiency of such relationships is expressed by means of a *customer lifetime value*, or CLV. A hindsight on customers’ database reveals an influence of the individual factors on customers’ loyalty, and, therefore, allows to manipulate and forecast it for new and existing customers.

An important part of CLV is *retention rate*. The choice of modeling approach for CLV and retention behavior depends on the business environment and the target population\(^1\). A variety of models are proposed for estimation and forecasting of the retention behavior. One can find a comprehensive overview of CLV modeling in Gupta et al. (2006).

The data set under the study comes from continuous and non-contractual business environment. It means that customers are allowed to make purchases when they want, and they are not bound by any subscription agreement. The target population includes new customers only, i.e. those, who

\(^1\)A short, but nice description of types of relationship with customers can be found in Fader and Hardie (2009).
Chapter 1. Introduction

made a purchase for the first time\(^2\).

Referring to Gupta et al. (2006), a probit model is found to be suitable for the retention rate analysis and inference. It is also pointed by Verhoef (2004), that a univariate binomial probit model is a good starting point for the problem of the estimation of the retention behavior. Univariate probit model can be extended to a multinomial target, and serves as a base to the other approaches in modeling CLV — survival models, Pareto/NBD models etc. (see, for example, Fader and Hardie (2009)).

The selection of the Bayesian approach to the model estimation is driven by several reasons. Purely statistical advantages can be found in Albert and Chib (1993). A personal motivation of the author in studying Bayesian statistics and its application to a real-life problems is also a cause.

Overall business benefits from the model are also at the stake. Revealing the factors, which affect the retention rate, will lead to the more efficient allocation of the marketing expenses, more robust budgeting processes and customer segmentation; therefore, it shall help to maximize company’s profit.

1.2 Outline

This thesis will be structured as follows:

In chapter 2, the notation and theoretical basis will be introduced, that will be then used throughout other chapters. An overview of the model and sampling techniques will be given.

In chapter 3 and chapter 4, the theory will be applied to the real-world data. A data description will be given, together with graphical representation. The chapters will introduce different approaches to the target definition and models will be formalized, estimated and tested.

In chapter 5, the results will be summarized and possible extensions, applications and future work will be discussed.

Finally, Appendix C will present a supplementary theory about Poisson-Binomial distribution and in Appendix D source code for the models estimation will be listed.

\(^2\)A more detailed description of the dataset and the business environment is given in chapter 3 and chapter 4.
Chapter 2

Theoretical concepts

2.1 Setting up the model

We start with an introduction of a general form of the multinomial probit (hereinafter referred to as MNP) model. The following notation for the MNP model based on the Chib et al. (1998). The form, in which the MNP model is introduced, will be useful afterwards, when we’ll move to its application on practice. Throughout the notation we assume that scalars are written in italics, vectors are written in boldface, and matrices are written in BOLDFACE.

Let $y_1^*, y_2^*, \ldots, y_n^* : \forall i = 1, n, y_i^* \in (1, \ldots, p)$ be a set of observable unordered outcomes for some population. Each outcome is associated with a set of covariates $(v_{ij}, w_i), j = 1, p$, where $v_{ij}$ is a covariate vector that varies across both subjects from the population and outcomes; and $w_i$ of size $q \times 1$ contains characteristics of subject $i$. Let $z_i^* = (z_{i1}^*, \ldots, z_{ip}^*)'$ be a $p \times 1$ vector of continuous unobserved latent variables, such that

$$y_i^* = j, \text{ if max}(z_i^*) = z_{ij}^* \quad (2.1)$$

and every latent variable $z_{ij}^*$ is an utility for an alternative $j$

$$z_{ij}^* = v_{ij}' \delta + w_i' \gamma_j + \xi_{ij} \quad (2.2)$$

In the real world one does not observe $z_{ij}^*$, only $y_i$ and covariates are known to the researcher. Writing (2.2) simultaneously for all $j = 1, p$ we obtain

$$\begin{align*}
z_{i1}^* &= v_{i1}' \delta + w_i' \gamma_1 + \xi_{i1} \\
z_{i2}^* &= v_{i2}' \delta + w_i' \gamma_2 + \xi_{i2} \\
&\vdots \\
z_{ip}^* &= v_{ip}' \delta + w_i' \gamma_p + \xi_{ip}
\end{align*} \quad (2.3)$$
By denoting

\[
V_i = \begin{pmatrix} v_{i1}' & 0' & \cdots & 0' \\
v_{i2}' & 0' & \cdots & 0' \\
\vdots & \vdots & \ddots & \vdots \\
v_{ip}' & 0' & \cdots & w_i' \end{pmatrix}
\]

\[
W_i = \begin{pmatrix} w_i' & 0' & \cdots & 0' \\
0' & w_i' & \cdots & 0' \\
\vdots & \vdots & \ddots & \vdots \\
0' & 0' & \cdots & w_i' \end{pmatrix}
\]

\[
Y = \begin{pmatrix} \gamma_1 \\
\gamma_2 \\
\vdots \\
\gamma_p \end{pmatrix}
\]

we obtain

\[
z_i^* = V_i \delta + W_i Y + \xi_i
\]

Note, that \(V_i\) has the same number of rows as \(W_i\). The notation can be simplified further by introducing:

\[
R_i = (V_i, W_i) = \begin{pmatrix} v_{i1}' & w_i' & 0' & \cdots & 0' \\
v_{i2}' & 0' & w_i' & \cdots & 0' \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
v_{ip}' & 0' & 0' & \cdots & w_i' \end{pmatrix}
\]

\[
\beta^* = \begin{pmatrix} \delta \\
Y \end{pmatrix}
\]

Summarizing (2.1 - 2.6) and, assuming that \(\xi_{ij}\) are distributed according to \(p\)-dimensional normal distribution with \(0\) mean vector and the covariance matrix \(\Xi\), i.e. \(\xi_i = (\xi_{i1}, \xi_{i2}, \ldots, \xi_{ip})'_{p \times 1} \sim N_p(0, \Xi)\), the MNP model is formalized as

\[
y_i^* = j, \text{ if } \max (z_i^*) = z_{ij}^*, \quad \forall j = 1, p
\]

\[
z_i^* = R_i \beta^* + \xi_i
\]

The definition of MNP, as it given in (2.7), lacks of unique identification of stated preference. Suppose, one adds a positive constant \(a\) to the every equation in (2.3). It does not introduce any changes to (2.1) in terms that
stated preference remains unaffected, thus, value of $a$ cannot be estimated. This problem is known as the additive redundancy.

The usual way of dealing with the additive redundancy is to subtract $p^{th}$ equation in (2.3) from the others, so we have

$$\begin{align*}
    z_{i1}^* - z_{ip}^* &= (v'_{i1} - v'_{ip}) \delta + w'_i (\gamma_1 - \gamma_p) + (\xi_{i1} - \xi_{ip}) \\
    z_{i2}^* - z_{ip}^* &= (v'_{i2} - v'_{ip}) \delta + w'_i (\gamma_2 - \gamma_p) + (\xi_{i2} - \xi_{ip}) \\
    &\vdots \\
    z_{i(p-1)}^* - z_{ip}^* &= (v'_{i(p-1)} - v'_{ip}) \delta + w'_i (\gamma_{p-1} - \gamma_p) + (\xi_{i(p-1)} - \xi_{ip})
\end{align*}$$

By introducing $z_{ij} = z_{ij}^* - z_{ip}^*$ and $\varepsilon_{ij} = \xi_{ij} - \xi_{ip}, \forall j = 1, p-1$ into (2.8) and following the same logic as in (2.4 - 2.6), we re-parameterize the model in (2.7) as next

$$X_i = (A_i, B_i)$$

$$A_i = \begin{pmatrix}
    v'_{i1} - v'_{ip} \\
    v'_{i2} - v'_{ip} \\
    \vdots \\
    v'_{i(p-1)} - v'_{ip}
\end{pmatrix}$$

$$B_i = \begin{pmatrix}
    w'_i & 0' & \cdots & 0' \\
    0' & w'_i & \cdots & 0' \\
    \vdots & \vdots & \ddots & \vdots \\
    0' & 0' & \cdots & w'_i
\end{pmatrix}$$

(2.9)

and, finally

$$y_i = \begin{cases}
    j, \text{ if } \max(z_i) = z_{ij} > 0 \\
    0, \text{ if } \max(z_i) < 0
\end{cases}, j = 1, p-1$$

$$z_i = X_i \beta + \varepsilon_i$$

$$\varepsilon_i \sim \mathcal{N}_{p-1}(0, \Sigma)$$

(2.10)

The covariance matrix $\Sigma$ of new error terms satisfies

$$\Sigma = [I_{p-1}, -1_{p-1}] \Xi [I_{p-1}, -1_{p-1}]'$$

(2.11)
Chapter 2. Theoretical concepts

Let us denote

\[ P_{ij} = Pr \left[ y_i = j \mid X_i, \beta, \Sigma \right] \]
\[ = Pr \left[ (z_{ij} > z_{ic}, \forall c \neq j) \cap (z_{ij} > 0) \right] \]
\[ = Pr \left[ (x'_{ij}\beta + \varepsilon_{ij} > x'_{ic}\beta + \varepsilon_{ic}, \forall c \neq j) \cap (x'_{ij}\beta + \varepsilon_{ij} > 0) \right] \]
\[ = Pr \left[ (\varepsilon_{ij} - \varepsilon_{ic} > (x'_{ic} - x'_{ij}) \beta, \forall c \neq j) \cap (\varepsilon_{ij} > -x'_{ij}\beta) \right] \]
\[ = \int_{E_j} f_{\varepsilon_i}(0, \Sigma) \, d\varepsilon_i, \forall j = 1, p-1 \]

where

\[ E_j = (\varepsilon_{ij} > -x'_{ij}\beta) \cap (\varepsilon_{ij} - \varepsilon_{ic} > (x'_{ic} - x'_{ij}) \beta) \]

and

\[ f_{\varepsilon_i}(0, \Sigma) = \frac{1}{(2\pi)^{\frac{p-1}{2}} \sqrt{\left| \Sigma \right|}} e^{-\frac{1}{2} \varepsilon_i^T \Sigma^{-1} \varepsilon_i} \]

Now, the likelihood function for individual \( i \) becomes

\[ l_i(\beta, \Sigma) = \sum_{j=0}^{p-1} \left[ Pr \left[ y_i = j \right] I (y_i = j) \right] \]
\[ = \prod_{j=0}^{p-1} \left[ Pr \left[ y_i = j \right] \right]^{d_{ij}} \]

where \( d_{ij} = 1 \), if \( y_i = j \), and \( d_{ij} = 0 \) otherwise. The likelihood function for the full sample is

\[ L(\beta, \Sigma) = \prod_{i=1}^{n} \prod_{j=0}^{p-1} \left[ Pr \left[ y_i = j \right] \right]^{d_{ij}} \]

From the definition of \( E_j \) in (2.13) we conclude, that the additive redundancy problem has vanished. However, the model in (2.10) still lacks of the identification. If both sides of the every equation in (2.8) are multiplied by some positive constant \( c \), this will change data generation process while leaving probabilities \( P_{ij} \) in (2.12) unchanged. Thus, \( L(\beta, \Sigma) = L(c\beta, c^2\Sigma) \) and we face multiplicative redundancy.

A standard solution to multiplicative redundancy is to normalize one of the diagonal elements \( \sigma_{jj} \) of \( \Sigma \) to 1 (the general rule is to normalize \( \sigma_{11} \)). One can find more extensive explanation of both additive redundancy and multiplicative redundancy (or, commonly saying, the identification problem).
Chapter 2. Theoretical concepts


The expression for probabilities in (2.12) captures practical obstacles of applications of MNP. There is no closed form expression for the high dimensional integral, when \( p \geq 4 \). And that is where the Bayesian approach comes in very handy. While some academical works were dedicated to the simulation estimation of the maximum likelihood in (2.16), Bayesian methods in inference about MNP were proven to be more efficient as they allowed to completely avoid burdensome computation of maximum likelihood. One can find references to comparison of both methods in Nobile (1998).

2.2 Bayesian framework

As one may note, the multiplicative redundancy imposes certain difficulties to the estimation of the MNP model with Bayesian inference in its general form. The reason for that is simple enough: it is difficult to define a prior distribution on the set of covariance matrices such that the first diagonal element is 1.

2.2.1 Binary case

Let us first consider a special case of the MNP model, when the target variable takes only two values, i.e. \( p = 2 \). The special case above is nothing but binary probit model, which takes the form:

\[
y_i = \begin{cases} 
1, & \text{if } z_i \geq 0 \\
0, & \text{if } z_i < 0
\end{cases}
\]

\[
z_i = x'_i \beta + \varepsilon_i \tag{2.17}
\]

\[
\varepsilon_i \sim N(0,1)
\]

Note, that in case of binary probit we drop out indexing by \( j \), since there is only one latent variable \( z_i \) that represents utility function for every individual \( i = 1, \ldots, n \). The binary probit model is estimable by the frequentist technique as well. The latter approach, however, is not of the main interest for the paper, but we can use it to compare the frequentist results with the Bayesian-estimated model, especially in the case of the non-informative prior distribution for \( \beta \).

To estimate parameters of binary probit model we follow Albert and Chib (1993). Let us briefly depict the sampling scheme for the estimation. Note, that we assume non-informative prior for \( \beta \):
Chapter 2. Theoretical concepts

1. given $X, y$ and $\beta$, sample $z_i, \forall i = 1, n$ according to
   
   \[ z_i|X, y, \beta \sim N^+(x'_i\beta, 1), \text{ if } y_i = 1 \]
   \[ z_i|X, y, \beta \sim N^-(x'_i\beta, 1), \text{ if } y_i = 0 \]

   where $N^+$ denotes normal distribution, truncated by 0 from left; and $N^-$ — truncated by 0 from right;

2. given $X$ and $z$ from step 1, sample $\beta$ according to (see (A.2))
   
   \[ \beta|X, z \sim N_q\left( (X'X)^{-1} (X'z), (X'X)^{-1} \right) \]

3. repeat steps 1 and 2 sufficient number of times to obtain samples from posteriors for inference.

The first step of the above sampling scheme is called data augmentation. The sampling algorithm from truncated univariate normal distribution is implemented according to Robert (1995). An initial value of $\beta_0$ for step 1 can be taken either as ordinary least square estimate or as maximum likelihood estimate. Once getting a vector of latent variables sampled in step 1, we use it to sample vector of coefficients $\beta$. The sampling procedure from multivariate normal distribution is described in Gelman et al. (2003). The model estimation procedures are implemented in R and C code (see Appendix D for details).

2.2.2 Multinomial case

As it was said in the beginning of the section 2.2, the application of the Bayesian framework to MNP estimation in its general form is complicated by the identification problem. We will follow McCulloch and Rossi (1994) and Rossi et al. (2005) for inference about MNP in its general form. The Gibbs sampler, proposed by the authors, iterates over the non-identified set of model parameters. It imposes certain restriction to the sampling scheme, namely, we have to use proper prior for model parameters $\beta$ and $\Sigma$ to ensure the convergence of the Gibbs sampler. Note, that with this approach $\beta$ and $\Sigma$ are not identified, but $\tilde{\beta} = \beta/\sqrt{\sigma_{jj}}$ and $\tilde{\Sigma} = \Sigma/\sigma_{jj}$ are.

The sampling scheme for MNP model estimation is given below:

1. given $X, y, \Sigma$ and $\beta$, sample $z_{ij}, \forall i = 1, n, j = 1, p - 1$ according to
   
   \[ z_{ij}|X, y, z_{i\cdot j}, \beta, \Sigma \sim N^{+\max(z_{i\cdot j}, 0)}(m_{ij}, \tau_{jj}^2), \text{ if } y_i = j \]
   \[ z_{ij}|X, y, z_{i\cdot j}, \beta, \Sigma \sim N^{-\max(z_{i\cdot j}, 0)}(m_{ij}, \tau_{jj}^2), \text{ if } y_i \neq j \]
Chapter 2. Theoretical concepts

where \( \mathbb{N}^+_{\text{max}}(z_{1,j}, 0) \) denotes univariate normal distribution, truncated by \( \max (z_{1,j}, 0) \) from left; and \( \mathbb{N}^-_{\text{max}}(z_{1,j}, 0) \) — truncated by \( \max (z_{1,j}, 0) \) from right, and \( z_{1,j} \) denotes \( z_i \) vector without \( z_{ij} \) element. \( m_{ij} \) and \( \tau_{jj}^2 \) represent conditional mean and variance of normal distribution, calculated according to:

\[
\tau_{jj}^2 = 1 / \sigma_{jj}^2 \\
m_{ij} = x_{ij}' \beta - (\sigma_{jj}^2 \gamma_{j,-j})' (z_{1,-j} - X_{1,-j} \beta)
\]

where \( \sigma_{ij} \) is a \((i,j)\) element of \( \Sigma^{-1} \), \( \gamma_{j,-j} \) is a \( j \)th row of \( \Sigma^{-1} \) without \( j \)th element, and \( X_{1,-j} \) is \( X_i \) with \( j \)th row deleted;

2. given \( X, \Sigma \) and \( z \) from step 1, sample \( \beta \) according to (see (B.3)):

\[
\beta | X, z, \Sigma \sim N (\mu_1, A_1)
\]

where

\[
A_1 = (A_0^{-1} + X^o \cdot X^o)^{-1} \\
\mu_1 = A_1 (A^{-1}_0 \mu_0 + X^o \cdot z^o) \\
X^o = C'X, z^o = C'z
\]

and, \( \mu_0 \) and \( A_0 \) are the prior mean and covariance matrix of \( \beta \);

3. given \( X, \beta \) and \( z \) from steps 1 and 2, sample \( \Sigma \) according to:

\[
\Sigma^{-1} | X, z, \beta \sim \mathcal{W} (\nu + n, (V_0 + S)^{-1})
\]

where

\[
S = \sum_{i=1}^{n} \varepsilon_i \varepsilon_i'
\]

\[
\varepsilon_i = z_i - X_i \beta
\]

and, \( \mathcal{W} \) denotes Wishart distribution, \( \nu \) and \( V_0 \) are the parameters of conjugate (Wishart) prior distribution.

4. repeat steps 1, 2, and 3 sufficient number of times to obtain samples from posteriors for inference.

As in binary case, the first step of the above sampling scheme is already familiar data augmentation. Since \( z_{ij} \) are sampled from truncated univariate normal distribution, we can use the same algorithm as in binary case. Initial values are taken as \( 0 \) vector for \( z_i \), and as prior modes for \( \Sigma \) and \( \beta \). The model estimation procedures are implemented in \( R \) and \( C \) code (see Appendix D for details).
Chapter 3

Model estimation and inference: binary case

3.1 Target population

Before proceeding to the practical application of the theoretical concepts, we shall describe the business environment, discuss the dataset and give the target definition.

The dataset for the analysis originates from a fast-growing part of the financial market — an unsecured individual-based micro-loan lending business. As usually, the product offered by companies, which operate on that market, is characterized by: a) small loan sizes; b) short loan terms; c) simplified application procedures and identity verifications; d) the least amount of time to gain approval; e) high credit risks; and, as a result, f) high interest rates.

The target population for study is defined as all the new customers\(^1\), “acquired” over a particular period of time\(^2\) by a privately held company\(^3\), that offers a micro-loan product.

First, we consider the simplest form of the MNP model — a model with binary outcome. Let’s call it model 1. We determine the target variable for the model in the following way. Let us call the fact of taking any successive loan by a customer as “customer \(i\) utilizes the \(n\)th loan” and denote the target variable with traditional \(y\). Let the event \(E\) be defined as “customer \(i\) utilizes the 2nd loan 365 days after the 1st one” and assign binary outcomes for \(y\) on

---

\(^1\)A customer is considered to be a new for the company if he/she is granted a loan for the first time.

\(^2\)The explicit specification of time period depends on the target definition as we shall see below.

\(^3\)The name of the company is kept in secret due to the confidentiality issues.
Chapter 3. Model estimation and inference: binary case

the individual level as follows:

\[ y_i = \begin{cases} 
1, & \text{if } E \text{ occurs} \\
0, & \text{otherwise} 
\end{cases} \tag{3.1} \]

To have completely specified population with the target variable defined in (3.1), we must refer to a period in the past that allows us to track down the event \( E \) for every customer in the population. For model 1 we just incorporate the time frame by specifying the target population as “all the new customers who took the first loan in year 2010”. On the one hand, the definition allows to track down the target variable for every new customer who meets the definition; and, on the other hand, it leaves the space for the model validation. After the estimation of model parameters we take “all the new customers who took the first loan in January, February or March of year 2011” as the population for the checking goodness-of-fit.

It ought to be remarked that we deal with a product which has the loan repayment term as one of the characteristics. It turns out that with the current definition of the event \( E \), a customer, who utilizes a micro-loan with longer repayment term, might be at a disadvantage to a customer opted for a loan with shorter repayment term.

To better understand the situation, let us illustrate it. Suppose a customer A got a loan with 30 days repayment period, and we start monitoring him/her to detect the target variable at the very same moment he/she was granted the loan. After 365 days we stop monitoring and assign a proper value to the target variable according to the occurrence of event \( E \). At the same time, suppose that a customer B took a 60 days loan, and a picture for him is somewhat different, as it is shown in Figure 3.1.
Despite an attractive simplicity, \textbf{model 1} does not capture an integral feature of the micro-loan business. As usually, a company is reluctant to grant a consequent loan to the late payers. The described characteristic is not covered by the target variable, neither it can be captured by covariates\footnote{Since we do not know beforehand if a customer pays back in time, as we do not know for sure if a customer pays back the loan at all.}. The extension of \textbf{model 1} to a multinomial case can solve the problem, but we will first concentrate on the estimation of the \textbf{model 1} anyway. In some sense the model may imply the answer to the question about late payers, because, for such customers we will definitely observe $y_i = 0$ as the outcome.

### 3.2 Dataset description

The dataset, obtained according to the model definition in (3.1), counts 7768 observations, out of which 5130 resulted in $y_i = 1$ as outcome for the target variable.

The set of covariates for the model is determined by the operating environment and availability of the information about the customers, their characteristics and the history of purchases. For the sake of study, the set of covariates can be divided into two subsets: 1) covariates, related to the customers; and, 2) covariates, related to the product. The set of customers’ related covariates is given in Table 3.1.
### Chapter 3. Model estimation and inference: binary case

Table 3.1: Customers related covariates

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GEN</td>
<td>Dummy variable for gender; takes value 0 for females, and 1 for males</td>
</tr>
<tr>
<td>AGE</td>
<td>Age in years, calculated at the moment of taking a loan</td>
</tr>
<tr>
<td>ZIP</td>
<td>Denotes, if a customer belongs to one of the nine area code groups, formed according to the first digit of the area post code</td>
</tr>
<tr>
<td>DH</td>
<td>Hour of the day, when the loan was taken, according to 24h format</td>
</tr>
<tr>
<td>DP</td>
<td>Period of the day, when the loan was taken; takes values</td>
</tr>
</tbody>
</table>
|          | \[
|          | \begin{cases}
|          | 0, & \text{if } DH \geq 0 \text{ and } < 6 \\
|          | 1, & \text{if } DH \geq 6 \text{ and } < 12 \\
|          | 2, & \text{if } DH \geq 12 \text{ and } < 18 \\
|          | 3, & \text{otherwise} \\
|          | \end{cases}
|          | \] |
| DW       | Day of the week, when the loan was taken; takes value 1 for Sunday, 2 for Monday, and so on. In the model introduced through dummies. |
| PU       | Dummy variable; takes value 1 if a new customer had unsuccessful applications for loans before taking a loan, and 0 otherwise |

To illustrate the dataset for model 1, for the every variable we will present a contingency table and a mosaic plot as a graphical representation. We start with variable GEN (Figure 3.2).
Chapter 3. Model estimation and inference: binary case

<table>
<thead>
<tr>
<th></th>
<th>$y_i = 0$</th>
<th>$y_i = 1$</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Females</td>
<td>1188</td>
<td>2576</td>
<td>3764</td>
</tr>
<tr>
<td>Males</td>
<td>1450</td>
<td>2554</td>
<td>4004</td>
</tr>
<tr>
<td>Total</td>
<td>2638</td>
<td>5130</td>
<td>7768</td>
</tr>
</tbody>
</table>

Figure 3.2: Contingency table and mosaic plot for variable GEN, model 1.

For simplicity of representation (both tabular and graphical) of variable AGE, we group all the observation into the six disjoint sets according to the next rule:

$$
\begin{align*}
\text{\textless} 30, & \quad \text{if } AGE \leq 30 \\
31 - 40, & \quad \text{if } AGE \in (30; 40] \\
41 - 50, & \quad \text{if } AGE \in (40; 50] \\
51 - 60, & \quad \text{if } AGE \in (50; 60] \\
61 - 70, & \quad \text{if } AGE \in (60; 70] \\
\text{\textgreater} 70, & \quad \text{if } AGE > 70
\end{align*}
$$

The contingency table and mosaic plot for grouped observations for variable AGE are represented in Figure 3.3.

---

5The observations are grouped only for the sake of representation. The variable AGE is included in the model as a quantitative variable.
Chapter 3. Model estimation and inference: binary case

<table>
<thead>
<tr>
<th></th>
<th>( y_i = 0 )</th>
<th>( y_i = 1 )</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \leq 30 )</td>
<td>951</td>
<td>1666</td>
<td>2617</td>
</tr>
<tr>
<td>31-40</td>
<td>739</td>
<td>1381</td>
<td>2120</td>
</tr>
<tr>
<td>41-50</td>
<td>526</td>
<td>1166</td>
<td>1692</td>
</tr>
<tr>
<td>51-60</td>
<td>276</td>
<td>656</td>
<td>932</td>
</tr>
<tr>
<td>61-70</td>
<td>118</td>
<td>218</td>
<td>336</td>
</tr>
<tr>
<td>( &gt; 70 )</td>
<td>28</td>
<td>43</td>
<td>71</td>
</tr>
<tr>
<td>Total</td>
<td>2638</td>
<td>5130</td>
<td>7768</td>
</tr>
</tbody>
</table>

Figure 3.3: Contingency table and mosaic plot for (grouped) variable \( AGE \), model 1.

The pattern for variable \( AGE \) is clearly visible: the number of the observations decreases as age increases, while the quantum of the outcomes with \( y_i = 1 \) follows parabolic curve. In this case, as often happens, variable \( AGE \) has to be included into the model together with its squared value to capture the dependence of the target variable on the covariate.

The contingency table and mosaic plot for \( ZIP \) are represented in Figure 3.4.
Chapter 3. Model estimation and inference: binary case

<table>
<thead>
<tr>
<th>ZIP</th>
<th>$y_i = 0$</th>
<th>$y_i = 1$</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>674</td>
<td>1268</td>
<td>1942</td>
</tr>
<tr>
<td>2</td>
<td>324</td>
<td>632</td>
<td>956</td>
</tr>
<tr>
<td>3</td>
<td>154</td>
<td>352</td>
<td>506</td>
</tr>
<tr>
<td>4</td>
<td>335</td>
<td>633</td>
<td>968</td>
</tr>
<tr>
<td>5</td>
<td>265</td>
<td>529</td>
<td>794</td>
</tr>
<tr>
<td>6</td>
<td>225</td>
<td>473</td>
<td>698</td>
</tr>
<tr>
<td>7</td>
<td>330</td>
<td>633</td>
<td>963</td>
</tr>
<tr>
<td>8</td>
<td>200</td>
<td>371</td>
<td>571</td>
</tr>
<tr>
<td>9</td>
<td>130</td>
<td>240</td>
<td>370</td>
</tr>
<tr>
<td>Total</td>
<td>2638</td>
<td>5130</td>
<td>7768</td>
</tr>
</tbody>
</table>

Figure 3.4: Contingency table and mosaic plot for variable ZIP, model 1.

The contingency tables and mosaic plots of the time-related variables $DP$ and $DW$ are given in Figure 3.5 and Figure 3.6 respectively.

<table>
<thead>
<tr>
<th>$DP$</th>
<th>$y_i = 0$</th>
<th>$y_i = 1$</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>84</td>
<td>138</td>
<td>222</td>
</tr>
<tr>
<td>1</td>
<td>877</td>
<td>1762</td>
<td>2639</td>
</tr>
<tr>
<td>2</td>
<td>1202</td>
<td>2247</td>
<td>3449</td>
</tr>
<tr>
<td>3</td>
<td>475</td>
<td>983</td>
<td>1458</td>
</tr>
<tr>
<td>Total</td>
<td>2638</td>
<td>5130</td>
<td>7768</td>
</tr>
</tbody>
</table>

Figure 3.5: Contingency table and mosaic plot for variable $DP$, model 1.

22
Chapter 3. Model estimation and inference: binary case

\[ y_i = 0 \quad y_i = 1 \quad \text{Total} \]

<table>
<thead>
<tr>
<th></th>
<th>( y_i = 0 )</th>
<th>( y_i = 1 )</th>
<th>\text{Total}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mon</td>
<td>480</td>
<td>988</td>
<td>1468</td>
</tr>
<tr>
<td>Tue</td>
<td>447</td>
<td>866</td>
<td>1313</td>
</tr>
<tr>
<td>Wed</td>
<td>453</td>
<td>823</td>
<td>1276</td>
</tr>
<tr>
<td>Thu</td>
<td>456</td>
<td>911</td>
<td>1367</td>
</tr>
<tr>
<td>Fri</td>
<td>399</td>
<td>722</td>
<td>1121</td>
</tr>
<tr>
<td>Sat</td>
<td>171</td>
<td>379</td>
<td>550</td>
</tr>
<tr>
<td>Sun</td>
<td>232</td>
<td>441</td>
<td>673</td>
</tr>
<tr>
<td>Total</td>
<td>2638</td>
<td>5130</td>
<td>7768</td>
</tr>
</tbody>
</table>

Figure 3.6: Contingency table and mosaic plot for variable \(DW, \text{model 1} \).

The next variable to be discussed is \(PU\). Its tabular and graphical representation is given in Figure 3.7.

\[ y_i = 0 \quad y_i = 1 \]

<table>
<thead>
<tr>
<th></th>
<th>( y_i = 0 )</th>
<th>( y_i = 1 )</th>
<th>\text{Total}</th>
</tr>
</thead>
<tbody>
<tr>
<td>(PU = 0)</td>
<td>2043</td>
<td>4091</td>
<td>6134</td>
</tr>
<tr>
<td>(PU = 1)</td>
<td>595</td>
<td>1039</td>
<td>1634</td>
</tr>
<tr>
<td>Total</td>
<td>2638</td>
<td>5130</td>
<td>7768</td>
</tr>
</tbody>
</table>

Figure 3.7: Contingency table and mosaic plot for variable \(PU, \text{model 1} \).

The available characteristics of the product, which can be used as covariates, are given in Table 3.2.
Chapter 3. Model estimation and inference: binary case

Table 3.2: Product related covariates

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LAM</td>
<td>Loan amount, notional currency units</td>
</tr>
<tr>
<td>LF</td>
<td>Loan fee, %</td>
</tr>
<tr>
<td>ZF</td>
<td>Dummy variable; takes value 1 if a new customer was “acquired” through “zero fee” campaign, and 0 otherwise</td>
</tr>
<tr>
<td>LT</td>
<td>Loan term, days</td>
</tr>
<tr>
<td>LI</td>
<td>Number of loan installments</td>
</tr>
<tr>
<td>DS</td>
<td>Dummy variable; takes value 1 if a new customer took a loan after a down-sell offer, and 0 otherwise</td>
</tr>
</tbody>
</table>

The set of covariates, given in Table 3.2 is redundant. As it often the case in the micro-loan business, such product characteristics as amount, term, fee and number of installments are highly correlated with each other. The statement is supported by the correlation matrix for variables LAM, LF, LT, and LI, that is presented in Table 3.3:

Table 3.3: The correlation matrix for variables LAM, LF, LT, and LI.

<table>
<thead>
<tr>
<th></th>
<th>LAM</th>
<th>LF</th>
<th>LT</th>
<th>LI</th>
</tr>
</thead>
<tbody>
<tr>
<td>LAM</td>
<td>1.00</td>
<td>0.62</td>
<td>0.91</td>
<td>0.87</td>
</tr>
<tr>
<td>LF</td>
<td>0.62</td>
<td>1.00</td>
<td>0.60</td>
<td>0.46</td>
</tr>
<tr>
<td>LT</td>
<td>0.91</td>
<td>0.60</td>
<td>1.00</td>
<td>0.96</td>
</tr>
<tr>
<td>LI</td>
<td>0.87</td>
<td>0.46</td>
<td>0.96</td>
<td>1.00</td>
</tr>
</tbody>
</table>

As a result, there is no need to include all of the four variables above in the model. We can leave the most representative ones, which are defined to be LAM and LF.

For variable LAM, as in case with AGE, we group all the observations into seven non-overlapping sets, according to the rule below\(^6\) (all numbers

\(^6\)In the dataset under study, loans with amount greater than 600 units are not available for new customers.
are given in notional currency units):

\[
\begin{align*}
50, & \quad \text{if } LAM \leq 50 \\
100, & \quad \text{if } LAM \in (50; 100] \\
200, & \quad \text{if } LAM \in (100; 200] \\
300, & \quad \text{if } LAM \in (200; 300] \\
400, & \quad \text{if } LAM \in (300; 400] \\
500, & \quad \text{if } LAM \in (400; 500] \\
600, & \quad \text{if } LAM \in (500; 600]
\end{align*}
\]

We follow the same way as with customers’ related covariates, and present the contingency table and mosaic plot for the grouped values of variable \( LAM \).

\[
\begin{array}{ccc}
& y_i = 0 & y_i = 1 & \text{Total} \\
50 & 304 & 744 & 1048 \\
100 & 761 & 1635 & 2396 \\
200 & 362 & 968 & 1330 \\
300 & 683 & 1076 & 1759 \\
400 & 68 & 110 & 178 \\
500 & 119 & 131 & 250 \\
600 & 341 & 466 & 807 \\
\text{Total} & 2638 & 5130 & 7768
\end{array}
\]

Figure 3.8: Contingency table and mosaic plot for (grouped) variable \( LAM \), model 1.

As we can see from Figure 3.8, the number of observations and the number of outcomes with \( y_i = 1 \) both go down as the loan amount goes up.

Variable \( LF \) depends on amount range. As it set up by the company,

\[7\]The observations are grouped only for the sake of representation. The variable \( LAM \) is included in the model as a quantitative variable.
there are five major cut-off points for $LF$:

$$\begin{align*}
0\%, & \quad \text{if } LF = 0 \\
15\%, & \quad \text{if } LF \in (0; 0.15] \\
25\%, & \quad \text{if } LF \in (0.15; 0.25] \\
27.5\%, & \quad \text{if } LF \in (0.25; 0.275] \\
30\%, & \quad \text{if } LF \in (0.275; 0.30]
\end{align*}$$

The contingency table and mosaic plot for the grouped values of variable $LF$ are represented in Figure 3.9.

<table>
<thead>
<tr>
<th></th>
<th>$y_i = 0$</th>
<th>$y_i = 1$</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>0%</td>
<td>556</td>
<td>1211</td>
<td>1767</td>
</tr>
<tr>
<td>15%</td>
<td>362</td>
<td>968</td>
<td>1330</td>
</tr>
<tr>
<td>25%</td>
<td>1058</td>
<td>2048</td>
<td>3106</td>
</tr>
<tr>
<td>27.5%</td>
<td>362</td>
<td>491</td>
<td>853</td>
</tr>
<tr>
<td>30%</td>
<td>300</td>
<td>412</td>
<td>712</td>
</tr>
<tr>
<td>Total</td>
<td>2638</td>
<td>5130</td>
<td>7768</td>
</tr>
</tbody>
</table>

Figure 3.9: Contingency table and mosaic plot for variable $LF$, model 1.

The pattern for $LF$ resembles the one for $LAM$, because, as we mentioned before, loan fee depends on loan amount.

One of the reasons to include variable $ZF$ can be found in Gupta et al. (2006). The authors referred to the fact, that new customers who were acquired through a discount offer, showed lower retention rate\(^8\), than those, who bought the product with a regular price. The behavior of the new customers in case of variable $ZF$ is depicted in Figure 3.10.

---

\(^8\)Retention rate in the current situation is computed as number of customers who made a consequent purchase after using a discount offer.
There is one more dummy variable to depict: \textit{DS}. In many business environments a \textit{down-sell} is called an event when for some reasons a customer decides to back down from the purchase and a company offers him/her the product with cheaper price in order to keep the customers and push the sales up.

In the analyzed case the down-sell offer consists in as follows. It is a conventional approach, when a new customer applies for a loan, he/she becomes a subject for credit scoring procedures, which have different implementations across the lending companies. According to the total amount of scored points, a customer is placed in one of the risk categories, which, in turn, have upper limits for the loan amount that can be granted to the applicant. Thus, when the scoring procedure results in a lower amount than it was requested, the customer receives a down-sell proposal. Such approach definitely boosts up the sales, but its effect on preserving the customers is rather negative, as we can see in Figure 3.11. The percentage of “positive” outcomes for the target variable is noticeably smaller in case when the loan was a result of a down-sell offer.
### Chapter 3. Model estimation and inference: binary case

#### 3.3 Model estimation results

Before proceeding to the model estimation, let us summarize all the described covariates and results of their $\chi^2$ tests for contingency tables. One can find an overview of the $\chi^2$ test in, for example, Bolboača et al. (2011). Under the null hypothesis for the $\chi^2$ we assume independence between a covariate and the target variable. The summary is given in Table 3.4.

![Contingency Table and Mosaic Plot](image)

**Figure 3.11:** Contingency table and mosaic plot for variable $DS$, **model 1**.

<table>
<thead>
<tr>
<th></th>
<th>$y_i = 0$</th>
<th>$y_i = 1$</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>$DS = 0$</td>
<td>2688</td>
<td>4252</td>
<td>6340</td>
</tr>
<tr>
<td>$DS = 1$</td>
<td>550</td>
<td>878</td>
<td>1428</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>2638</td>
<td>5130</td>
<td>7768</td>
</tr>
</tbody>
</table>
As we can see, the test does support the null hypothesis about independence between a covariate and the target variable for the ZIP, DP, and DW independent variables. We should not be rather beforehand our conclusion, because the $\chi^2$ test checks the bivariate independence without considering the influence of the other covariates.

We will first try to estimate the model coefficients for the full set of covariates, enlisted in Table 3.1. To estimate the coefficients of model 1 we will follow the algorithm for binary case, that was previously described in section 2.2. The sampling procedures are implemented in R (in connection with C) programming language according to the given references (the code listings are given in Appendix D). We run Gibbs sampler chain of 5000 iterations with “burn-in” period of 1000 iterations for the given data. The initial value of $\beta_0$ was taken as OLS estimate, i.e. $\beta_0 = (X'X)^{-1} (X'y)$, and a uniform non-informative prior was assumed for $\beta$.

The result of the model 1 estimation with the full set of covariates is not presented due to the fact that it is quite lengthy. But, it indeed supports the conclusion drawn from the $\chi^2$ test about the independence between the target variable and the ZIP, DP, and DW covariates. The estimated by means of MLE coefficients for these covariates are all statistically insignificant, and in Bayesian approach 90% higher posterior density intervals all cover 0 for them.

Therefore, it makes sense to exclude the irrelevant covariates and, fi-
Chapter 3. Model estimation and inference: binary case

Finally, model 1 will include the following covariates: \( \text{GEN}, \text{AGE}, \text{AGE}^2 \) (the squared value of \( \text{AGE} \)), \( \text{PU}, \text{LAM}, \text{LF}, \text{ZF} \), and \( \text{DS} \); and we proceed to the model estimation.

We also produce MLE estimate for comparison, as we may expect the results to be close for the large sample and non-informative prior. The intercept term was included in estimation. The initial value, posterior moments, 90% highest posterior density (HPD) intervals of \( \beta \), the MLE estimate of model 1 with corresponding 90% confidence intervals are given in Table 3.5.

Table 3.5: Posterior results for model 1.

<table>
<thead>
<tr>
<th>Variable</th>
<th>( \hat{\beta}_{\text{post}} )</th>
<th>( SD_{\beta_{\text{post}}} )</th>
<th>90% HPDI</th>
<th>( \hat{\beta}_{\text{MLE}} )</th>
<th>90% CI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>0.3261</td>
<td>0.1668</td>
<td>(0.0415; 0.5825)</td>
<td>0.3252</td>
<td>(0.0535; 0.5972)</td>
</tr>
<tr>
<td>( \text{GEN} )</td>
<td>-0.1152</td>
<td>0.0296</td>
<td>(-0.1613; -0.0642)</td>
<td>-0.1152</td>
<td>(-0.1641; -0.0663)</td>
</tr>
<tr>
<td>( \text{AGE} )</td>
<td>0.0301</td>
<td>0.0071</td>
<td>(0.0183; 0.0415)</td>
<td>0.0302</td>
<td>(0.0187; 0.0418)</td>
</tr>
<tr>
<td>( \text{AGE}^2 )</td>
<td>-0.0003</td>
<td>0.00008</td>
<td>(-0.0004; -0.0002)</td>
<td>-0.0003</td>
<td>(-0.0004; -0.0002)</td>
</tr>
<tr>
<td>( \text{PU} )</td>
<td>-0.0615</td>
<td>0.0368</td>
<td>(-0.1239; -0.0040)</td>
<td>-0.0620</td>
<td>(-0.1215; -0.0023)</td>
</tr>
<tr>
<td>( \text{LAM} )</td>
<td>-0.00009</td>
<td>0.00001</td>
<td>(-0.0001; -0.00007)</td>
<td>-0.00009</td>
<td>(-0.0001; -0.00007)</td>
</tr>
<tr>
<td>( \text{LF} )</td>
<td>-1.1390</td>
<td>0.4121</td>
<td>(-1.8395; -0.4028)</td>
<td>-1.1430</td>
<td>(-1.8283; -0.4597)</td>
</tr>
<tr>
<td>( \text{ZF} )</td>
<td>-0.3402</td>
<td>0.0950</td>
<td>(-0.4963; -0.1870)</td>
<td>-0.3396</td>
<td>(-0.4962; -0.1835)</td>
</tr>
<tr>
<td>( \text{DS} )</td>
<td>-0.1594</td>
<td>0.0390</td>
<td>(-0.2259; -0.0977)</td>
<td>-0.1591</td>
<td>(-0.2249; -0.0933)</td>
</tr>
</tbody>
</table>

As we can see, the 90% HPD intervals do not cover 0 for all the coefficients, which also supports our inference about significance of the selected covariates.

From the frequentist point of view, all the coefficients, but the intercept term, in model 1 are statistically significant in the MLE estimate, which was actually confirmed by HPD intervals in Bayesian estimate. The results from MLE and the Bayesian approach are very close to each other, since we used non-informative prior for \( \beta \).

Posterior results for the components of \( \beta \) are depicted below. Posterior densities are represented in Figure 3.12, posterior draws are given in Figure 3.13, and their autocorrelations functions are given in Figure 3.14.
Chapter 3. Model estimation and inference: binary case

Figure 3.12: Posterior densities for $\beta$, model 1.
Figure 3.13: Sampled output for $\beta$, model 1.
Chapter 3. Model estimation and inference: binary case

Figure 3.14: Autocorrelation plots of sampled output for $\beta$, model 1.
Chapter 3. Model estimation and inference: binary case

A visual inspection of the output reveals no issues with the model convergence. What is more interesting to the researcher is an inference about the individual probabilities of the observed outcome for the out-of-sample data. As it was mentioned in section 3.1, we can check how reasonable are the individual posterior probabilities from the estimated model. The dataset for checking consists of 1,428 observations, out of which 917 resulted in \( y_i = 1 \) as an outcome for the target variable.

Having a set of new covariates and the posterior sample of parameters, we can obtain \( P_{post}(y_i^{new} = 1|y) \) for the every individual in the new data set. First, we obtain vector-valued latent variables on individual level, then transform them into probabilities by noting that in binary case

\[
P_{post}(y_i^{new} = 1|y) = \int_{\beta} P(y_i^{new} = 1|\beta) P(\beta|y) d\beta = E_{\beta}[P(y_i^{new} = 1|\beta)] = E_{\beta}[P(x_i^{new}\beta > 0)] = E_{\beta}[\Phi(x_i^{new}\beta)]
\]

where \( \Phi(\cdot) \) denotes the standard Gaussian cumulative distribution function. One must pay attention that we predict probability of the outcome \( y_i = 1 \), but not the outcome itself. However, by taking \( y_i = 1 \) if \( P_{post}(y_i = 1) > 0.5 \) as rule of thumb we can compare the predicted outcomes with the observed ones and check goodness-of-fit of the model by testing how it predicts the out-of-sample cases. Misclassification table, according to the rule of thumb above, is presented by Table 3.6.

Table 3.6: Misclassification table for model 1.

<table>
<thead>
<tr>
<th></th>
<th>( y_i^{obs} = 0 )</th>
<th>( y_i^{obs} = 1 )</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>( y_i^{pred} = 0 )</td>
<td>80</td>
<td>123</td>
<td>203</td>
</tr>
<tr>
<td>( y_i^{pred} = 1 )</td>
<td>431</td>
<td>794</td>
<td>1225</td>
</tr>
<tr>
<td>Total</td>
<td>511</td>
<td>917</td>
<td>1428</td>
</tr>
</tbody>
</table>

As we can see from the Table 3.6 the model classified “positive” outcomes quite well. However, it classified incorrectly quite a lot of outcomes, which

\(^9\)In R programming environment one can use coda package to analyze the model convergence.
Chapter 3. Model estimation and inference: binary case

in a matter of fact resulted in $y_i = 0$. In total, approximately one third of all the observation are misclassified. The latter points us out the “weak” definition of model 1, as we mentioned in the end of section 3.1; and calls for model 1 extension to a multinomial case.

The calculation of the individual probabilities is not a final destination of our inference about the retention rate. The outcome $y_i = 1$ on the individual level is nothing but a Bernoulli trial with given probability of success. Statistically speaking, we have $B_1, \ldots, B_i, \ldots, B_N$ independent Bernoulli trials, each with probability of success $P(y_i = 1)$. Then, the sum $S_N = B_1 + \cdots + B_N$ is said to be a Poisson-Binomial random variable.

The description of the Poisson-Binomial distribution (hereinafter referred to as PBD) and methods for computing its probability mass function can be found in Appendix C.

The mean value of $S_N$ can be obtained directly from the properties of the sum of independent random variables. If we denote $P(y_i = 1) = p_i$, then, by using the fact that for the every individual $E(p_i) = p_i$, we compute $E(S_N) = E(B_1 + \cdots + E(B_N))$ as $\sum_{i=1}^{N} p_i$. Following the same logic, we compute variance of $S_N$ as $\sum_{i=1}^{N} p_i(1 - p_i)$.

After “attaching” posterior predictive probabilities to the every individual from the new dataset, and using theoretical probability mass function of PBD, we can make an inference about the total number of new customers, who are likely to take the consequent loan 365 days after the first one. For the dataset, which we used for the model checking, the expected number results in 811 (individuals), which is quite close to the real-observed value (917).

A sample of $\beta$ from the posterior distribution might be useful in the future, if we want to estimate model 1 over a new data set. We can use empirical posterior mean and covariance matrix of $\beta$ from the previously estimated model as priors for the new one and see how the new data changes the posterior distribution.
Chapter 4

Model estimation and inference: multinomial case

4.1 Target population

To set up the target population and the variable of interest for the multinomial case, we proceed as we discussed in section 3.1: we will extend model 1 so it captures the late paying backs and non-paying backs. In fact, we only have to re-define the target variable and distinguish outcomes which resulted in late- or non-paying back. The later two outcomes result in a prohibition for consequent loans, so the customers who pay back their loans late, as well as defaulting debtors, may not take consequent loans.

As before, we call the fact of taking any successive loan by a customer as “customer $i$ utilizes the $n^{th}$ loan” and denote the target variable with traditional $y$. Let the event $E'$ be defined as “customer $i$ pays back the $1^{st}$ loan within 89 days after the due date and utilizes the $2^{nd}$ loan 365 days after the $1^{st}$ one”. We define the event $E''$ as “customer $i$ pays back the $1^{st}$ loan from 90 to 365 days after the due date” (the late paying back case). And, finally, the event $E'''$ is defined as “customer $i$ does not pay back the $1^{st}$ loan within 365 days after the due date” (non-paying back case). We assign outcomes for $y$.

---

1As a rule, many companies have a limit for days late, and a customer may not take consequent loan if he/she exceeds this limit. For the dataset under study such limit equals to 89 days.
Chapter 4. Model estimation and inference: multinomial case

on the individual level as follows:

\[ y_i = \begin{cases} 
1, & \text{if } E' \text{ occurs} \\
2, & \text{if } E'' \text{ occurs} \\
3, & \text{if } E''' \text{ occurs} \\
4, & \text{otherwise} 
\end{cases} \quad (4.1) \]

We call the multinomial model, specified in (4.1) as model 2. According to the definition of the \( E' \), \( E'' \), and \( E''' \) events, we can refer to the same target population as in model 1.

4.2 Dataset description

As before, the dataset, obtained according to the model definition in (4.1), counts 7 768 observations. The distribution of the outcomes is given in Table 4.1.

Table 4.1: The distribution of the outcomes for model 2.

<table>
<thead>
<tr>
<th></th>
<th>( y_i = 1 )</th>
<th>( y_i = 2 )</th>
<th>( y_i = 3 )</th>
<th>( y_i = 4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>freq.</td>
<td>5128</td>
<td>268</td>
<td>494</td>
<td>1878</td>
</tr>
</tbody>
</table>

We can illustrate the dataset for model 2 in a familiar way by means of contingency tables and mosaic plots, like for variable \( \text{GEN} \) (Figure 4.1).
Chapter 4. Model estimation and inference: multinomial case

\[ y_i = 1 \quad y_i = 2 \quad y_i = 3 \quad y_i = 4 \]

Total Females 2574 110 217 863 3764
Males 2554 158 277 1015 4004
Total 5128 268 494 1878 7768

Figure 4.1: Contingency table and mosaic plot for variable GEN, model 2.

While mosaic plot does allow to visually inspect the influence of the covariate on the target variable, for the sake of saving readers time, we will represent only contingency tables for the rest of the independent variables.

\[
\begin{array}{cccccc}
AGE & y_i = 1 & y_i = 2 & y_i = 3 & y_i = 4 & \text{Total} \\
\leq 30 & 1664 & 74 & 160 & 719 & 2617 \\
31-40 & 1381 & 77 & 149 & 513 & 2120 \\
41-50 & 1166 & 63 & 100 & 363 & 1692 \\
51-60 & 656 & 34 & 54 & 188 & 932 \\
61-70 & 218 & 14 & 24 & 80 & 336 \\
> 70 & 43 & 6 & 7 & 15 & 71 \\
\text{Total} & 5128 & 268 & 494 & 1878 & 7768 \\
\end{array}
\]

Figure 4.2: Contingency tables for variables AGE and ZIP, model 2.
Chapter 4. Model estimation and inference: multinomial case

<table>
<thead>
<tr>
<th></th>
<th>y₁ = 1</th>
<th>y₂ = 2</th>
<th>y₃ = 3</th>
<th>y₄ = 4</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>DP</td>
<td>y₀ = 0</td>
<td>138</td>
<td>5</td>
<td>15</td>
<td>64</td>
</tr>
<tr>
<td>DP</td>
<td>y₀ = 1</td>
<td>1761</td>
<td>90</td>
<td>163</td>
<td>625</td>
</tr>
<tr>
<td>DP</td>
<td>y₀ = 2</td>
<td>2246</td>
<td>131</td>
<td>225</td>
<td>847</td>
</tr>
<tr>
<td>DP</td>
<td>y₀ = 3</td>
<td>983</td>
<td>42</td>
<td>91</td>
<td>342</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>5128</td>
<td>268</td>
<td>494</td>
<td>1878</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>y₁ = 1</th>
<th>y₂ = 2</th>
<th>y₃ = 3</th>
<th>y₄ = 4</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>DW</td>
<td>y₀ = 0</td>
<td>988</td>
<td>50</td>
<td>94</td>
<td>136</td>
</tr>
<tr>
<td>DW</td>
<td>y₀ = 1</td>
<td>1385</td>
<td>65</td>
<td>110</td>
<td>114</td>
</tr>
<tr>
<td>DW</td>
<td>y₀ = 2</td>
<td>823</td>
<td>38</td>
<td>101</td>
<td>114</td>
</tr>
<tr>
<td>DW</td>
<td>y₀ = 3</td>
<td>911</td>
<td>42</td>
<td>76</td>
<td>338</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>5128</td>
<td>268</td>
<td>494</td>
<td>1878</td>
</tr>
</tbody>
</table>

Figure 4.3: Contingency tables for variables $DP$ and $DW$, model 2.

<table>
<thead>
<tr>
<th></th>
<th>y₁ = 1</th>
<th>y₂ = 2</th>
<th>y₃ = 3</th>
<th>y₄ = 4</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>PU</td>
<td>y₀ = 0</td>
<td>4089</td>
<td>160</td>
<td>348</td>
<td>1517</td>
</tr>
<tr>
<td>PU</td>
<td>y₀ = 1</td>
<td>1019</td>
<td>88</td>
<td>146</td>
<td>361</td>
</tr>
<tr>
<td>PU</td>
<td>y₀ = 2</td>
<td>1075</td>
<td>21</td>
<td>22</td>
<td>76</td>
</tr>
<tr>
<td>PU</td>
<td>y₀ = 3</td>
<td>465</td>
<td>42</td>
<td>95</td>
<td>195</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>5128</td>
<td>268</td>
<td>494</td>
<td>2638</td>
</tr>
</tbody>
</table>

Figure 4.4: Contingency tables for variables $PU$ and $LAM$, model 2.

<table>
<thead>
<tr>
<th></th>
<th>y₁ = 1</th>
<th>y₂ = 2</th>
<th>y₃ = 3</th>
<th>y₄ = 4</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZF</td>
<td>y₀ = 0</td>
<td>3917</td>
<td>250</td>
<td>452</td>
<td>1382</td>
</tr>
<tr>
<td>ZF</td>
<td>y₀ = 1</td>
<td>1211</td>
<td>18</td>
<td>42</td>
<td>496</td>
</tr>
<tr>
<td>ZF</td>
<td>y₀ = 2</td>
<td>1211</td>
<td>18</td>
<td>42</td>
<td>496</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>5128</td>
<td>268</td>
<td>494</td>
<td>1878</td>
</tr>
</tbody>
</table>

Figure 4.5: Contingency tables for variables $ZF$ and $DS$, model 2.
Chapter 4. Model estimation and inference: multinomial case

\[
y_i = 1 
\begin{array}{cccc}
\text{Total} & \text{0.5\%} & 1211 & 18 & 42 & 496 & 1767 \\
\text{15\%} & 908 & 28 & 45 & 289 & 1330 \\
\text{25\%} & 2047 & 117 & 251 & 691 & 3106 \\
\text{27.5\%} & 491 & 59 & 73 & 230 & 853 \\
\text{30\%} & 411 & 46 & 83 & 172 & 712 \\
\end{array}
\]

Figure 4.6: Contingency tables for variable \( LF \), model 2.

4.3 Model estimation results

As before, we summarize all the described covariates and results of their \( \chi^2 \) tests for contingency tables. The summary is given in Table 4.2.

Table 4.2: The summary table of covariates, model 2.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Figure</th>
<th>( X^2 )</th>
<th>( \nu (\text{df}) )</th>
<th>( \chi^2 (5%) )</th>
<th>( p )-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>GEN</td>
<td>4.1</td>
<td>20.87</td>
<td>3</td>
<td>7.81</td>
<td>0.0001</td>
</tr>
<tr>
<td>AGE</td>
<td>4.2</td>
<td>44.95</td>
<td>15</td>
<td>24.99</td>
<td>0.0001</td>
</tr>
<tr>
<td>ZIP</td>
<td>4.2</td>
<td>37.02</td>
<td>24</td>
<td>36.41</td>
<td>0.0436</td>
</tr>
<tr>
<td>DP</td>
<td>4.3</td>
<td>8.22</td>
<td>9</td>
<td>16.21</td>
<td>0.5121</td>
</tr>
<tr>
<td>DW</td>
<td>4.3</td>
<td>18.52</td>
<td>18</td>
<td>28.87</td>
<td>0.4215</td>
</tr>
<tr>
<td>PU</td>
<td>4.4</td>
<td>49.62</td>
<td>3</td>
<td>7.81</td>
<td>0.0000</td>
</tr>
<tr>
<td>LAM</td>
<td>4.4</td>
<td>374.75</td>
<td>18</td>
<td>28.87</td>
<td>0.0000</td>
</tr>
<tr>
<td>LF</td>
<td>4.6</td>
<td>251.02</td>
<td>12</td>
<td>21.02</td>
<td>0.0000</td>
</tr>
<tr>
<td>ZF</td>
<td>4.5</td>
<td>112.78</td>
<td>3</td>
<td>7.81</td>
<td>0.0000</td>
</tr>
<tr>
<td>DS</td>
<td>4.5</td>
<td>60.53</td>
<td>3</td>
<td>7.81</td>
<td>0.0000</td>
</tr>
</tbody>
</table>

The \( \chi^2 \) test again points on independence between the target variable and the time-related covariates \( DP \) and \( DW \), and “weak” significance of \( ZIP \), therefore, we exclude them from the list of covariates for the model estimation, and model 2 includes the same set of covariates as model 1, and we proceed to the model estimation.
Chapter 4. Model estimation and inference: multinomial case

The sampling procedures are implemented in R and C programming languages with references and code listings given in Appendix D. As referred to McCulloch and Rossi (1994), Rossi et al. (2005), and Imai and van Dyk (2005), the current sampling scheme shows very slow convergence, so, we use longer chain and “burn-in” period. To ensure the convergence we will generate a chain with length set to 50 000 iterations with 100 000 “burn-in” period, i.e. the total chain length is 150 000. Finally, we will apply “thinning” to the chain output when reporting sample properties, i.e. we will report each 10th observation. Note, as it was pointed out in subsection 2.2.2, the current approach requires proper prior specifications for the parameters of interest, but still, we are able to use highly diffuse proper priors to express our uncertainty about initial location and distribution of estimated parameters. We use the following priors:

\[
\pi(\beta) \sim \mathcal{N}(0, 200) \\
\pi(\Sigma^{-1}) \sim \mathcal{W}(p + 2, (p + 2)I_{p-1})
\]

and corresponding initial values for Gibbs sampler steps 1, 2, and 3: \(z_0 = 0, \beta_0 = 0, \Sigma_0 = I_{p-1}\). It should be noted that we sample from non-identified parameters set, and we should not report posteriors for \(\beta\) and \(\Sigma\), but for \(\tilde{\beta} = \beta/\sqrt{\sigma_{jj}}\) and \(\tilde{\Sigma} = \Sigma/\sigma_{jj}\).

In our case, we have 4 alternatives and only individual characteristics which implies that in (2.9) - (2.10) we have the following:

\[
X_i = B_i = \begin{pmatrix}
w'_i & 0' & \cdots & 0' \\
0' & w'_i & \cdots & 0' \\
\vdots & \vdots & \ddots & \vdots \\
0' & 0' & \cdots & w'_i
\end{pmatrix}
\]

\[
y_i = \begin{cases}
j, & \text{if } \max(z_i) = z_{ij} > 0 \\
0, & \text{if } \max(z_i) < 0
\end{cases}, \quad j = 1, 3
\]

\[
z_i = X_i\beta + \varepsilon_i \\
\varepsilon_i \sim \mathcal{N}(0, \Sigma)
\]

The posterior moments, 90% highest posterior density (HPD) intervals of \(\tilde{\beta}\) are given in Table 4.3. For convenience, the individual characteristics are additionally labeled by the number of equation in (2.9) to identify their place in the vector of the coefficients. The values are based on “thinned” posterior sample output.

\footnote{Imai and van Dyk (2005) mention 60 000 to 80 000 iterations required to achieve convergence for this algorithm.}
### Table 4.3: Posterior results for model 2, $\tilde{\beta}$.

<table>
<thead>
<tr>
<th>Variable</th>
<th>$\tilde{\beta}_{post}$</th>
<th>SD $\tilde{\beta}_{post}$</th>
<th>90% HPDI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept$_1$</td>
<td>0.1231</td>
<td>0.1224</td>
<td>(-0.0784; 0.3308)</td>
</tr>
<tr>
<td>Intercept$_2$</td>
<td>-0.0350</td>
<td>0.0694</td>
<td>(-0.1431; 0.0807)</td>
</tr>
<tr>
<td>Intercept$_3$</td>
<td>0.0272</td>
<td>0.0521</td>
<td>(-0.0609; 0.1098)</td>
</tr>
<tr>
<td>GEN$_1$</td>
<td>-0.0892</td>
<td>0.0290</td>
<td>(-0.1378; -0.0439)</td>
</tr>
<tr>
<td>GEN$_2$</td>
<td>-0.0229</td>
<td>0.0146</td>
<td>(-0.0482; -0.00009)</td>
</tr>
<tr>
<td>GEN$_3$</td>
<td>-0.0266</td>
<td>0.0127</td>
<td>(-0.0468; -0.0049)</td>
</tr>
<tr>
<td>AGE$_1$</td>
<td>0.0316</td>
<td>0.0059</td>
<td>(0.0223; 0.0417)</td>
</tr>
<tr>
<td>AGE$_2$</td>
<td>0.0128</td>
<td>0.0032</td>
<td>(0.0078; 0.0182)</td>
</tr>
<tr>
<td>AGE$_3$</td>
<td>0.0112</td>
<td>0.0026</td>
<td>(0.0068; 0.0151)</td>
</tr>
<tr>
<td>AGE$^2$_1</td>
<td>-0.0003</td>
<td>0.00007</td>
<td>(-0.0004; -0.0002)</td>
</tr>
<tr>
<td>AGE$^2$_2</td>
<td>-0.0001</td>
<td>0.00004</td>
<td>(-0.0002; -0.00006)</td>
</tr>
<tr>
<td>AGE$^2$_3</td>
<td>-0.0001</td>
<td>0.00003</td>
<td>(-0.0002; -0.00005)</td>
</tr>
<tr>
<td>PU$_1$</td>
<td>-0.0069</td>
<td>0.0338</td>
<td>(-0.0611; 0.0508)</td>
</tr>
<tr>
<td>PU$_2$</td>
<td>0.0382</td>
<td>0.0173</td>
<td>(0.0123; 0.0692)</td>
</tr>
<tr>
<td>PU$_3$</td>
<td>0.0266</td>
<td>0.0144</td>
<td>(0.0036; 0.0511)</td>
</tr>
<tr>
<td>LAM$_1$</td>
<td>-0.00007</td>
<td>0.00001</td>
<td>(-0.00009; 0.00005)</td>
</tr>
<tr>
<td>LAM$_2$</td>
<td>-0.000007</td>
<td>0.000006</td>
<td>(-0.00002; 0.00003)</td>
</tr>
<tr>
<td>LAM$_3$</td>
<td>-0.000008</td>
<td>0.000005</td>
<td>(-0.00002; -0.000006)</td>
</tr>
<tr>
<td>LF$_1$</td>
<td>-0.2481</td>
<td>0.2055</td>
<td>(-0.5715; 0.0997)</td>
</tr>
<tr>
<td>LF$_2$</td>
<td>0.0262</td>
<td>0.1160</td>
<td>(-0.1633; 0.2130)</td>
</tr>
<tr>
<td>LF$_3$</td>
<td>0.0371</td>
<td>0.0927</td>
<td>(-0.1210; 0.1828)</td>
</tr>
<tr>
<td>ZF$_1$</td>
<td>-0.1757</td>
<td>0.0566</td>
<td>(-0.2680; -0.0824)</td>
</tr>
<tr>
<td>ZF$_2$</td>
<td>-0.0838</td>
<td>0.0316</td>
<td>(-0.1357; -0.0316)</td>
</tr>
<tr>
<td>ZF$_3$</td>
<td>-0.0672</td>
<td>0.0255</td>
<td>(-0.1073; -0.0247)</td>
</tr>
<tr>
<td>DS$_1$</td>
<td>-0.0950</td>
<td>0.0372</td>
<td>(-0.1578; -0.0356)</td>
</tr>
<tr>
<td>DS$_2$</td>
<td>0.0102</td>
<td>0.0199</td>
<td>(-0.0212; 0.0436)</td>
</tr>
<tr>
<td>DS$_3$</td>
<td>0.0255</td>
<td>0.0162</td>
<td>(-0.0019; 0.0512)</td>
</tr>
</tbody>
</table>

As we can see, the 90% HPD intervals do cover 0 for some of the coefficients of $\tilde{\beta}$, for instance, $LF$ and the intercept term, thus their inclusion in
the model is questionable.

Posterior results for the components of $\tilde{\beta}$ are depicted below. Posterior densities are represented in Figure 4.7, Figure 4.8, and Figure 4.9, posterior draws are given in Figure 4.10, Figure 4.11, and Figure 4.12, and their autocorrelations functions are given in Figure 4.13, Figure 4.14, and Figure 4.15. All the figures are based on “thinned” posterior sample output.

Figure 4.7: Posterior densities for $\tilde{\beta}$, model 2, (part 1).
Figure 4.8: Posterior densities for $\tilde{\beta}$, model 2, (part 2).
Figure 4.9: Posterior densities for $\tilde{\beta}$, **model 2**, (part 3).
Figure 4.10: Sampled output for $\tilde{\beta}$, model 2, (part 1).
Figure 4.11: Sampled output for $\tilde{\beta}$, model 2, (part 2).
Figure 4.12: Sampled output for $\tilde{\beta}$, model 2, (part 3).
Figure 4.13: Autocorrelation plots of output for $\tilde{\beta}$, model 2, (part 1).
Figure 4.14: Autocorrelation plots of output for \( \tilde{\beta} \), model 2, (part 2).
Figure 4.15: Autocorrelation plots of output for $\tilde{\beta}$, model 2, (part 3).
Note, that visualization of the distribution of the components of $\tilde{\Sigma}$ is not a trivial task. It would involve 3-dimensional plots or some more elaborated approaches, such as proposed by Tokuda et al. (n.d.). We omit visualization of variance-covariance matrix in the paper.

Since each unobserved latent variable is sampled conditional on the others, it causes autocorrelation in $\tilde{\beta}$. It is expected result due to the specific nature of the Gibbs sampler in use. The “thinning” does help to mitigate autocorrelation problem. A visual inspections of the posterior sample output reveals higher than in binary case autocorrelation and not very good “mixing” for some coefficients. In general, there are no serious issues with the convergence of the sampler and we find it acceptable for further inference.

Having a set of new covariates and the posterior sample of parameters, we can obtain $P_{\text{post}}(y_{i}^{\text{new}} = j | y)$ for the every individual in the new data set:

\[
P_{\text{post}}(y_{i}^{\text{new}} = j | y) = \int P(y_{i}^{\text{new}} = j | \beta, \Sigma, y) P(\beta, \Sigma | y) d(\beta, \Sigma)
\] (4.2)

First, having a sample from joint posterior distribution for $\beta$ and $\Sigma$ we can sample latent variables for every new individual, and then compute preferred choice as defined in (2.10). Thus, we obtain a sample of possible outcomes out of which we can get the probability of each outcome, and using theoretical probability mass function of $PBD$, we can make an inference about the total expected number of new customers, who are likely to “fall” in one of the categories defined in (4.1).

We can use the same data set as for model 1, but with outcomes defined according to model 2 specifications (4.1). Recall, that the dataset consists of 1 428 observations. After calculating the posterior probabilities of each outcome for individuals from the new data set, and obtaining the expected values of individuals per outcome, we get the following result:

<table>
<thead>
<tr>
<th></th>
<th>$y_i = 1$</th>
<th>$y_i = 2$</th>
<th>$y_i = 3$</th>
<th>$y_i = 4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>observed freq.</td>
<td>916</td>
<td>45</td>
<td>108</td>
<td>359</td>
</tr>
<tr>
<td>expected freq.</td>
<td>827</td>
<td>98</td>
<td>153</td>
<td>350</td>
</tr>
</tbody>
</table>

As we can see, expected values are quite close to the real observed frequencies, which means good model performance in terms of estimation.
4.4 Applying results for CLV calculation

The calculation of the individual probabilities for each outcome is not a final destination of our inference about the retention rate in the multinomial case. In a fact, from model 2 we can obtain all the required components for calculation of the CLV. We gave some references to the CLV concept in section 1.1. In general, CLV is defined as the net present value of the future expected earnings of customers, Gupta et al. (2006):

$$\text{CLV}^j = \sum_{t=0}^{T^j} \left( p^j_t - c^j_t \right) r^j_t \frac{1}{(1+i)^t} - AC^j$$

(4.3)

where

- $p^j_t$ = price, paid by a customer $j$ at time $t$,
- $c^j_t$ = cost of service for the customer $j$ at time $t$,
- $r^j_t$ = retention rate for the customer $j$ at time $t$, or probability, that the customer “alive” at time $t$,
- $i$ = discount rate or capital cost for the company,
- $AC^j$ = acquisition cost for the customer $j$, or price, paid by the company to “acquire” the customer $j$,
- $T^j$ = time horizon for estimating CLV for the customer $j$.

The definition of model 2 allows us to estimate a lower bound of expected CLV of new customers for one year horizon\(^3\). In this case we can set $i = 0$ in (4.3) because omitting the discount factor does not affect the estimation of CLV significantly.

Let us assume that some new customer wants to get a loan of 200 notional currency units with 30 days term and 15% fee. The individual characteristics of this new applicant is known to us at the moment of application, and we can use them to get the probabilities of the each outcome according to the model definition. We are primarily interested in outcomes $y = 1$, which denotes that the new customer pays back current loan within allowed days past due period and utilizes next loan within a year, and $y = 3$, which denotes that the new customer fails to pay the loan back within a year, so the outcome actually denotes credit losses. Therefore, these two outcomes are the main factors affecting the margin\(^4\). We want to ensure, that the acquisition cost will not exceed the expected margin, so one-year expected CLV is non-negative.

\(^3\)Lower bound is assumed due to the fact that new customers have the worst retention rate, and the highest credit losses.

\(^4\)In the business environment under study expected margin is defined as expected earnings minus expected losses due to non-paying back.
Chapter 4. Model estimation and inference: multinomial case

We substitute individual characteristics to the estimated model, and obtain posterior predictive probabilities as following:

<table>
<thead>
<tr>
<th>( P_{\text{post}}(y_i = 1) )</th>
<th>( P_{\text{post}}(y_i = 2) )</th>
<th>( P_{\text{post}}(y_i = 3) )</th>
<th>( P_{\text{post}}(y_i = 4) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.6690</td>
<td>0.0418</td>
<td>0.1030</td>
<td>0.1862</td>
</tr>
</tbody>
</table>

We are missing only one thing for calculation of CLV — the amount and the cost of the next loan. They are required for margin calculation. They can be predicted quantities as well, but for simplicity, we can take them as average values over existing customer database. Suppose, they are equal to 270 notional currency units and 22% correspondingly.

Now, expected CLV (lower bound, without acquisition cost) for the new individual for one year horizon becomes (in notional currency units):

\[
\text{CLV} = 200 \cdot (0.15 - 0.1030) + 270 \cdot 0.6690 \cdot (0.22 - 0.1030) \approx 30.53
\]

and, we clearly see, that in order to guarantee the profitability, the acquisition cost must not exceed this value.

This has very important practical meaning, since we can apply the similar calculation to every new customer and guarantee the overall profitability of the portfolio.

We can also analyze sensibility of the probabilities of outcomes (and, as a result, sensibility of CLV) to the magnitude of the individual components of the model. For instance, in the model under analysis we may be particularly interested how the loan amount (variable \( \text{LAM} \)) and loan fee (variable \( \text{LF} \)) affect the CLV. It is worth noting that the 90% HPD interval covers 0 for the corresponding coefficients of \( \tilde{\beta} \) for variable \( \text{LF} \), therefore, it is more likely that this variable will not have any influence on the individual probabilities of outcomes, so we will analyze the influence of \( \text{LAM} \) only.

The sensibility inference can be conducted as following: we take a new individual, and while keeping all the other characteristics constant, we gradually increase loan amount and after each increase we save it as a new observation. Then, we can use the obtained observations as an input to the predictive routine, and then we can analyze and visualize the changes for both CLV and probabilities of outcomes.
Let us continue the previous example. We take the same new customer, and keep all the individual characteristics the same, but we change loan amount starting from 50 notional currency units and gradually increase it by 10 units until it reaches 600. Following this way we obtain 56 observations which we use together with the posterior sample to predict the probabilities and substitute them into the \( CLV \) calculation. The dependency of \( CLV \) on the loan amount is depicted in Figure 4.16 below.

As we can see, when loan amount increases, the probability of taking next loan \( P(y = 1) \) goes down, and the probability of non-paying back the loan \( P(y = 3) \) goes up. It totally corresponds to business logic. At some moment, when loan amount is greater than 400 units, the probability of default exceeds loan fee, and the margin decays significantly and eventually
becomes negative. The result can be used for adjusting pricing policy on the individual level to ensure the profitability of the whole portfolio.
Chapter 5

Conclusion

5.1 Summary

In the current paper we presented an application of multinomial probit model to an important economic quantity, namely, customers’ retention rate, which, in turn, is a part of the more general conception of customer lifetime value. We gave an overview of the CLV conception, its mathematical representation and motivated the model selection.

We presented a general specification of the multinomial probit model, and described the identification problem that was specific for MNP. The sampling procedures, necessary for the estimation, were referenced and described.

The dataset for the practical application originated from an unsecured individual-based micro-loan lending business. We gave a description of the business environment, and discussed several approaches to the target definition.

We defined the models, starting from the special case of MNP — binary probit model. We motivated inclusion (or not inclusion) of certain covariates into it, and estimated models’ coefficients with a Bayesian approach. The Bayesian approach has several attractive features which allow to perform a more robust post-estimation inference. The individual posterior predictive probabilities for binary probit model led us to Poisson-Binomial distribution. Somewhat new approach was proposed and implemented for the calculation of pmf for PBD.

After the estimation of the binary probit model, we computed the posterior probabilities for the out-of-sample data. We found out that due to the “weak” definition, binary probit model had “bad” predictive performance for the cases where we observed $y_i = 0$ as an actual outcome. We must note here, that it all depends on the rule of thumb, and we predict probabilities of
the outcomes, but not the outcomes themselves. We compared an expected number of new customers, who were likely to take a consequent loan, with the real one. We found them to be very close to each other.

Then we extended the model to a multinomial case. When estimating the model we saw that the proposed sampling algorithm resulted in high autocorrelation of the output posterior sample, but it was expected and controlled by “thinning” of the output. We compared the expected number of new customers, who were likely to “fall” in one of the possible outcomes, to the observed quantities, and again, we found them to be very close to each other. The $MNP$ model estimation allowed us to get the necessary inputs for $CLV$ prediction as we demonstrated with an example. The results have important practical implications in terms of maximizing the profitability of the customers on the individual level.

The estimation and auxiliary procedures were implemented in the R software package with connection to C-code. It allowed us to speed up the execution time for both binary and multinomial cases.

### 5.2 Extensions, applications, and future work

The benefits for the business from the inference about the individual retention rates are quite apparent. Customers with higher retention rate bring more value to company’s profit. Therefore, knowing what affects the retention rate on the individual level, a company may allocate marketing budgets more efficiently and attract customers with higher probabilities of taking a consequent loan.

We are able to highlight some issues in the model 1 and model 2. Those are the ways of the target variable was defined. Indeed, a quantum of new customers do take a consequent loan more than 365 days after the first one. The root of the problem lies in the fact that we deal with a non-contractual business environment, i.e. a customer is not obliged to report if he/she wants to end up the relationship with the company. That part, of course, cannot significantly affect the retention rate. Moreover, due to the fact, that in the vast majority of the companies (not just on the micro-loans market) the budgeting period is a year (or 365 days), the given definition sounds reasonable, if one is going to apply the models for budgeting purposes.

The presented model cannot be considered as an ultimate one. The set of covariates, which were found to be relevant for this particular case, might not fit the datasets which are present across the different companies on the market. On the other hand, one may have an access to an extent set of covariates and explore their influence on the subject of interest.
Chapter 5. Conclusion

The sampling algorithm which we used to estimate the model parameters is quite easily implemented in practice, but, it has issues with convergence and it requires longer sampling chains, therefore it takes more time. Another issue is that we sample from non-identified set of parameters. There are many works, addressed to the issues above, and even to the alternative approaches to the model definition. Most noticeable of them are: McCulloch, Polson, et al. (2000) (algorithm for sampling from identified set of parameters), Imai and van Dyk (2005) (sampling schemes for fast convergence), and Johndrow et al. (2013) (new alternative to MNP).

MNP can be further extended to so-called multinomial multi-period probit model, where we could observe every individual over particular periods of time. MNMP model can be extremely useful as an application for the CLV calculation, since the mathematical expression of CLV assumes the time indexing.
Appendix A

Posterior distribution of $\beta$, binary case

We derive the posterior distribution of $\beta$ for binary case. The posterior distribution for $\beta$ is proportional to sample likelihood times prior:

$$p(\beta | z, X) \propto p(z | \beta, X) \pi(\beta) \quad (A.1)$$

$$\propto \exp \left[ -\frac{1}{2} (z - \beta'X)'(z - \beta'X) \right] \pi(\beta)$$

If prior distribution of $\beta$ is non-informative, i.e. $\pi(\beta) = 1$, then posterior density is an exponential of a quadratic form in $\beta$, thus, it is normal. By putting $\hat{\beta} = (X'X)^{-1}(X'z)$ and $V_\beta = (X'X)^{-1}$, we have:

$$p(\beta | z, X) \propto \exp \left[ -\frac{1}{2} (\beta - \hat{\beta}')V_\beta^{-1}(\beta - \hat{\beta}) \right] \sim N(\hat{\beta}, V_\beta)$$

We may express our prior beliefs about $\beta$ by assigning a proper prior distribution to the vector of coefficients. Sample likelihood is normally distributed, so, in order to get the posterior density of normal form, the conjugate prior distribution should also be normal. Let the conjugate prior
Appendix A. Posterior distribution of beta, binary case

distribution for $\beta$ is given by

$$\pi(\beta) \sim N(\mu_0, A_0)$$

Then, the posterior distribution for $\beta$ becomes

$$p(\beta|z, X) \propto p(z|\beta, X)\pi(\beta)$$

$$(A.3) \propto \exp\left[-\frac{1}{2}\left((z - X\beta)'(z - X\beta) + (\beta - \mu_0)' A_0^{-1}(\beta - \mu_0)\right)\right]$$

$$(A.4) \propto \exp\left[-\frac{1}{2}\left((\beta - \mu_1)' A_1^{-1}(\beta - \mu_1)\right)\right]$$

$\sim N(\mu_1, A_1)$

where

$$A_1 = (A_0^{-1} + X'X)^{-1}$$

$$\mu_1 = A_1 (A_0^{-1} \mu_0 + X'z)$$
Appendix B

Posterior distribution of $\beta$, $\Sigma$, multinomial case

The joint posterior distribution of $\beta$ and $\Sigma$ is proportional to sample likelihood times prior. Recall, that sample likelihood is given by (2.16), so that:

$$L(\beta, \Sigma) \propto |\Sigma|^{-\frac{n}{2}} e^{\frac{1}{2} \sum_{i=1}^{n} (z_i - X_i \beta)' \Sigma^{-1} (z_i - X_i \beta)}$$  \quad (B.1)

Thus, joint posterior distribution for of $\beta$ and $\Sigma$ becomes:

$$p(\beta, \Sigma | z, X) = p(\beta | \Sigma, z, X)p(\Sigma | z, X) \propto L(\beta, \Sigma)\pi(\beta|\Sigma)\pi(\Sigma)$$  \quad (B.2)

Before deriving posteriors for $p(\beta | \Sigma, z, X)$, let us transform the original model definition as following. Let

$$\Sigma^{-1} = C'C$$

This implies

$$\Sigma = (C'C)^{-1} = C^{-1}(C')^{-1}$$

By using properties of linear transformation of MNP distribution, we can multiply both left and right sides of $z_i$, defined in (2.10), by $C'$ from the right. This yields:

$$C'z_i = C'X_i \beta + C'\varepsilon_i$$
Appendix B. Posterior distribution of beta, Sigma, multinomial case

We have \( \varepsilon_i \sim N_{p-1}(0, \Sigma) \). Thus

\[
C'\varepsilon_i = \varepsilon_i^o \sim N_{p-1}(0, CC^{-1}(C')^{-1}C') \\
\sim N_{p-1}(0, I)
\]

When \( \Sigma \) is given, \( C' \) is known, and we can employ the same technique as in (A.3) to get:

\[
p(\beta|X, z, \Sigma) \sim N(\mu_1, A_1) \tag{B.3}
\]

where

\[
A_1 = (A_0^{-1} + X^oX^o)^{-1} \\
\mu_1 = A_1(A_0^{-1}\mu_0 + X^o z^o) \\
X^o = C'X, z^o = C'z
\]

To derive posterior distribution of \( \Sigma \) for the third step of Gibbs sampler, we note that having \( z_i \) and \( \beta \) we “observe” \( \varepsilon_i \):

\[
\varepsilon_i = z_i - X_i\beta \\
\sim N_{p-1}(0, \Sigma)
\]

Sample likelihood for \( \varepsilon_i \) is proportional to:

\[
p(\varepsilon_i|X, z, \beta) \propto |\Sigma|^{-n/2} \exp \left[ -\frac{1}{2} \sum_{i=1}^{n} \varepsilon_i^t \Sigma^{-1} \varepsilon_i \right]
\]

Using the fact that \( \varepsilon_i^t \Sigma^{-1} \varepsilon_i \) is a scalar, and the properties of the trace of the matrix \( tr(\cdot) \) operator, we can rewrite the likelihood as:

\[
p(\varepsilon_i|X, z, \beta) \propto |\Sigma|^{-n/2} \exp \left[ -\frac{1}{2} tr(\Sigma^{-1} S) \right]
\]

which is a kernel of inverse-Wishart distribution.

Thus, the kernel distribution of likelihood of \( \Sigma^{-1} \) is Wishart, and conjugate prior distribution for \( \Sigma^{-1} \) is Wishart. Sampling scheme for drawing from Wishart \( \mathcal{W}(\nu, S^{-1}) \) distribution is given in McCulloch and Rossi (1994). \( \Sigma^{-1} \) is partitioned as:

\[
\Sigma^{-1} = L\tilde{V}L'
\]
Appendix B. Posterior distribution of beta, Sigma, multinomial case

where

\[ \mathbf{L} : \mathbf{S} = \mathbf{LL}' \]
\[ \tilde{\mathbf{V}} \sim \mathcal{W}(\nu, \mathbf{I}) \]
\[ \tilde{\mathbf{V}} = \mathbf{T}\mathbf{T}' \]

and, \( \mathbf{T} \) is a lower triangular matrix obtained by drawing the square root of \( \chi^2 \) for diagonal elements, \( t_{ii} \sim \sqrt{\chi^2_{\nu}} \), and \( \mathcal{N}(0,1) \) draws for off-diagonal elements. Thus

\[ \mathbf{C} = \mathbf{LT} \]
\[ \Sigma^{-1} = \mathbf{CC}' \]
Appendix C

Poisson-Binomial distribution

We assume that the readers are familiar with the concept of a Bernoulli trial with the given probability of success. From the general statistical theory we know, that the sum of $N$ independent and identical Bernoulli trials follows Binomial distribution. The PBD arises, when we deal with the sum of $N$ independent but not identical Bernoulli trials. The application areas and properties of the PBD are well described in Chen and Liu (1997).

The main difficulty in using of the PBD on practice is concerned with the computation of its theoretical probability mass function. A variety of methods were proposed either for direct or for approximate and simulation methods for the computation of pmf for PBD. A nice overview of such methods is given in Hong (2011).

We propose the direct method of the computation of pmf for a Poisson-Binomially distributed random variable $S_N$ with given vector of success probabilities $p$ of size $N \times 1$. Mathematically, the pmf can be expressed as

$$P(S_N = s) = \sum_{A \in F_s} \left( \prod_{i \in A} p_i \prod_{j \in A^c} (1 - p_j) \right)$$  \hspace{1cm} (C.1)

where $F_s$ is the set of all subsets of $s$ integers that can be selected from $\{1, 2, \ldots, N\}$, and $A^c = \{1, 2, \ldots, N\} \setminus A$.

Let us give an illustrative example. Suppose, that $N = 3$, i.e. $p = \{p_1, p_2, p_3\}$. Then, $S_N$ can take one of the values from 0 to 3. According to (C.1), we have:

- $P(S_N = 0) = (1 - p_1)(1 - p_2)(1 - p_3)$
- $P(S_N = 1) = p_1(1 - p_2)(1 - p_3) + p_2(1 - p_1)(1 - p_3) + p_3(1 - p_1)(1 - p_2)$
- $P(S_N = 2) = p_1p_2(1 - p_3) + p_1p_3(1 - p_2) + p_2p_3(1 - p_1)$
- $P(S_N = 3) = p_1p_2p_3$
Appendix C. Poisson-Binomial distribution

As one may notice, the direct enumeration of all the probabilities even for relatively small values of $N$ is impractical. But, as we will show, there is a workaround, which makes the computation considerably easier. Let us continue to work with the example above, when $N = 3$; and consider the following expression:

$$\prod_{i=1}^{3} \left( x + \frac{1 - p_i}{p_i} \right)$$  \hspace{1cm} (C.2)

Equation (C.2) is a representation of a polynomial of order 3, which roots are $\frac{p_i - 1}{p_i}$, $\forall i = 1, 3$. Another representation of the polynomial in (C.2) in the general form is given as:

$$\sum_{k=0}^{3} A_k x^k$$  \hspace{1cm} (C.3)

To obtain the coefficients $A_k$ in (C.3) we will proceed by consequent multiplying out the expression in (C.2) and collecting the similar terms:

$$\left( x + \frac{1 - p_1}{p_1} \right) \left( x + \frac{1 - p_2}{p_2} \right) \left( x + \frac{1 - p_3}{p_3} \right)$$

$$= \left( x^2 + \frac{p_2(1-p_1) + p_1(1-p_2)}{p_1p_2} x + \frac{(1-p_1)(1-p_2)}{p_1p_2} \right) \left( x + \frac{1 - p_3}{p_3} \right)$$

$$= x^3 + \frac{p_2p_3(1-p_1)}{p_1p_2p_3} x^2 + \frac{p_2(1-p_1)(1-p_3) + p_1(1-p_2)(1-p_3) + p_3(1-p_1)(1-p_2)}{p_1p_2p_3} x$$

$$+ \frac{(1-p_1)(1-p_2)(1-p_3)}{p_1p_2p_3}$$

Now, if we multiply (C.2) and, consequently (C.3) by $p_1p_2p_3$, we can recognize, that after the multiplication, the coefficients $A_k$ become “familiar” probabilities from the example above, i.e.

$$p_1p_2p_3 \prod_{i=1}^{3} \left( x + \frac{1 - p_i}{p_i} \right)$$  \hspace{1cm} (C.4)

$$= P(S_N = 3)x^3 + P(S_N = 2)x^2 + P(S_N = 1)x^1 + P(S_N = 0)x^0$$

The generalization of (C.4) to an arbitrary $N > 1$ is straightforward.

$$p_1p_2 \cdots p_N \prod_{i=1}^{N} \left( x + \frac{1 - p_i}{p_i} \right)$$  \hspace{1cm} (C.5)

$$= P(S_N = N)x^N + P(S_N = N-1)x^{N-1} + \cdots + P(S_N = 0)x^0$$

66
Appendix C. Poisson-Binomial distribution

The algorithm for \((C.5)\) can be easily implemented on practice, because we know the roots of the polynomial, since they’re defined by the vector of probabilities \(p\). The implementation of the algorithm in C programming language is given in Listing D.8 along with Monte Carlo simulation method (for comparison). The computation of pmf leads us directly to cumulative distribution function of PBD, which is defined in a usual for discrete random variables way as:

\[
P(S_N \leq s) = \sum_{k=0}^{s} \left[ \sum_{A \in F_k} \left( \prod_{i \in A} p_j \prod_{j \in A^c} (1 - p_j) \right) \right]
\]  
\((C.6)\)

One may object to \((C.5)\) by arguing, that it requires \(p_i > 0, \forall i = 1, N\). However, it can be shown, that this restriction imposes no problem in computation of pmf of PBD with the proposed algorithm. As a matter of fact, when some of \(p_i = 0\), the whole vector of probabilities \(p\) can be partitioned into two disjoint sets: 1) \(s = \{ l : p_l > 0 \}\), and 2) \(s' = \{ m : p_m = 0 \}\). Let \(s\) contain \(L\) elements, and \(s'\) contain \(M\) elements, such that \(L + M = N\). Then we have:

\[
P(S_N = s) = P \left( \sum_{k=1}^{N} B_k = s \right)
= P \left( \sum_{k \in s} B_k + \sum_{k \in s'} B_k = s \right)
= P \left( \sum_{k \in s} B_k = s \right)
= P(S_L = s)
\]

and \(P(S_L = s)\) indeed can be computed with \((C.5)\). By noting, that in this case \(S_N\) cannot exceed \(L\), i.e. \(P(S_N > L) = 0\), we complete the computation of pmf.

Another issue one may face when dealing with PBD on practice is “tiny” size of the probabilities in pmf when \(N\) is considerably large (for instance, if \(N \geq 10000\)). The software environment has to be capable to handle such small numbers, which is not always the case. To overcome the situation we can “magnify” the probabilities in pmf by multiplying them with some positive integer power of 10 bearing in mind that we deal with “magnified” probabilities.
Appendix D

Source code documentation

All the computational and sampling procedures are implemented in R and C programming languages. R is an interpreted programming language, designed for statistical computations and graphics, and is freely available under the GNU General Public License.

R is computationally efficient when working with huge datasets, because most of the operations are vectorized, that allows to avoid loop control flows in many problems. But sometimes, loops are inevitable, as it often the case in sampling procedures. Despite R does support the loop control flows, they may considerably slow down the computations, especially when we cycle through very long lists of values. Such behavior of the loops is typical for most of the interpret programming languages.

As opposed to the interpretable code paradigm, programming languages, which produce pre-compiled executables, are much more efficient with loops. One of such languages is C and, fortunately for us, R provides different interfaces to call the code, written and compiled in C from within the R programming environment. In the current Appendix we will concern with .Call() interface, when implementing connection between R and C. The code in C has to be compiled into static link library in order to be referred through .Call() interface. In the following, we assume that all the static link libraries are compiled under Windows®¹ operational system.

The vast majority of the literature that explain all the details about calling other languages from the R programming environment, including a comprehensive manual “Writing R Extensions”, can be freely found in the Internet. In the following, we assume that the readers are familiar with basic conceptions of R and how it interacts with other programming languages. It is required, that R software package is properly installed and configured. For

¹Windows is a registered trademark of Microsoft Corporation in the United States and other countries.
Appendix D. Source code documentation

C-code compilation it is recommended to use an IDE that supports MinGW compiler. The author uses cross-platform and free Code::Blocks IDE. The compiled libraries has to be put into the R working directory\textsuperscript{2}. The every link library is presented in 32- and 64-bit version.

Let us introduce a naming convention, which we will follow, when giving names to function and variables in the code listings. The naming convention is simple enough and consists in: 1) use lowercase\_separated\_by\_underscores for variable names, and 2) use lowercase\_separated\_by\_dots for function names in R and lowercase\_separated\_by\_underscores in C. The meaning of the every input variable is commented either in the corresponding listing or in the text of the appendix. An illustrative example of usage together with output is given for the every R-function in the appendix.

Listing D.1: Source C-code for MNPBayes.dll

\begin{verbatim}
#include <Rmath.h>
#include <Rinternals.h>
#include <R.h>
#include <math.h>

double rnormtrunc(int td, double mu, double sigma, double tp){
  double p, x, u, res;
  if(td == 1){
    p = (tp - mu) / sigma;
  } else {
    p = (-tp + mu) / sigma;
  }
  if(p <= 0){
    while(1 == 1){
      x = rnorm(0.0, 1.0);
      if(x >= p) break;
    }
  } else {
    while(1 == 1){
      x = rexp(1 / p) + p;
      u = runif(0.0, 1.0);
      if(u <= exp(-pow((x - p), 2) / 2)) break;
    }
  }
  if(td == 1){
    res = sigma * x + mu;
  } else {
    res = -sigma * x + mu;
  }
  return res;
}

SEXP sample_z_bp(SEXP td, SEXP me, SEXP sig, SEXP tp){
  int n = length(td), i, *y;
  if(n != length(me) || n != length(sig) || n != length(tp)){
    Rprintf("Error! Inputs are of incorrect length!\n");
    return R_NilValue;
  }
  for(i = 0; i < n; i++) y[i] = i;
  for(i = n - 1; i > 0; i--)
    if(rand() / RAND_MAX <= p) swap(y[i], y[rand() % i]);
  SEXP res = VectorSym(RF_allocVector(STRSXM, n), FALSE);
  int j = 0;
  for(i = 0; i < n; i++)
    if(y[i] == i) res[j++] = td;
  return res;
}
\end{verbatim}

\textsuperscript{2}One can obtain the address of the working directory from within R environment by executing \texttt{getwd()} function.
Appendix D. Source code documentation

```c
double *mu, *zs, *a, *sigma;
td = PROTECT(coerceVector(td, INTSXP));
me = PROTECT(coerceVector(me, REALSXP));
sig = PROTECT(coerceVector(sig, REALSXP));
tp = PROTECT(coerceVector(tp, REALSXP));
y = INTEGER(td);
u = REAL(me);
a = REAL(tp);
sigma = REAL(sig);
SEXP z; PROTECT(z = allocVector(REALSXP, n));
zs = REAL(z);
GetRNGstate();
for(i = 0; i < n; i++){
    zs[i] = rnormtrunc(y[i], mu[i], sigma[i], a[i]);
    // R_CheckUserInterrupt();
}
PutRNGstate();
UNPROTECT(5);
return z;
}

SEXP sample_z_mnp(SEXP iz, SEXP o, SEXP me, SEXP sig, SEXP alt){
    int n = length(o), i, j, k, p = length(alt) - 1, *y, ind, jind, td;
    if(n * p != length(me) || n * p != length(iz) || length(sig) != p * p){
        Rprintf("Error! Inputs are of incorrect length!\n");
        return R_NilValue;
    }
    me = PROTECT(coerceVector(me, REALSXP));
    sig = PROTECT(coerceVector(sig, REALSXP));
    iz = PROTECT(coerceVector(iz, REALSXP));
    o = PROTECT(coerceVector(o, INTSXP));
    double *mu, *sigma, *zs, cond_mu, cond_sigma, tp;
    sigma = REAL(sig);
    mu = REAL(me);
    y = INTEGER(o);
    SEXP z = PROTECT(duplicate(iz));
    zs = REAL(z);
    GetRNGstate();
    for(i = 0; i < n; i++){
        ind = p * i;
        for(j = 0; j < p; j++){
            td = (y[i] == (j + 1)) ? 1 : 0;
            tp = 0.0;
            for(k = 0; k < p; k++){
                if(k != j){tp = (zs[ind + k] > tp) ? zs[ind + k] : tp;}
            }
            jind = j * p;
            cond_sigma = 1. / sigma[jind + j];
            cond_mu = 0.0;
            for(k = 0; k < p; k++){
                if(k != j){
                    cond_mu += -cond_sigma * sigma[jind + k] * (zs[ind + k] - mu[ind + k]);
                }
            }
            cond_mu += mu[j + ind];
            cond_sigma = sqrt(cond_sigma);
            zs[ind + j] = rnormtrunc(td, cond_mu, cond_sigma, tp);
        }
        // R_CheckUserInterrupt();
    }
    PutRNGstate();
}
```

70
Listing D.1 implements interfaces `sample_z_bp()` and `sample_z_mnp()` for data augmentation sampling for binary and multinomial probit models correspondingly, and a helper function `rnormtrunc()` for sampling from one-side truncated univariate normal distribution with given truncation direction, mean value, standard deviation and truncation point. The interfaces do a simple check for proper dimensionality of the inputs. Both interface functions in library `MNPBayes.dll` return a vector of random variables of corresponding length. They serve as a part to `R`-function for Bayesian estimation of binary and multinomial probit models, which are presented in Listing D.2 and Listing D.3.

Listing D.2: Source R-code for `bp.bayes` function

```r
bp.bayes <- function(
  data_set, #underlying dataset, data.frame
  reg_formula, #formula for the regression, i.e. y~AGE+GEN+...
  b0_as_MLE = FALSE, #how to compute an initial value for b0, boolean scalar
  n = 5000, #number of iterations in Gibbs sampling, integer scalar
  burnin = 1000, #burn-in size, integer scalar
  set_z = TRUE, #to set or not to set seed, boolean scalar
  seed = 7402, #if yes, seed value for RNG, scalar
  thinning = 1, #put >1 to thin posterior sample output, integer scalar
  mu0 = NULL, #prior mean for beta, numeric vector
  A0 = NULL, #prior covariance matrix for beta, numeric matrix
)
{
  fit_mle <- glm(reg_formula, data=data_set, family=binomial(link="probit"), x=TRUE, y=TRUE)
  if(!b0_as_MLE){
    fit <- lm(reg_formula, data=data_set, x=TRUE, y=TRUE)
  } else {
    fit <- fit_mle
  }
  b0 <- coef(fit)
  k <- length(b0)
  x <- fit$x
  xx <- crossprod(x)
  if(is.null(A0)){
    ch_var <- chol(xx)
  } else {
    ch_var <- chol(xx + solve(A0))
    IAO <- solve(A0)
    PA <- (xx + IAO)
    IAO_mu0 <- crossprod(t(IAO), mu0)
  }
  mu <- crossprod(t(x), b0)
  sig <- rep(1, length(mu))
  a <- rep(0, length(mu))
  y <- fit$y
  if(!is.loaded("sample_z_bp")){
    if(.Platform$r_arch == "i386") {dyn.load("MNPBayes.dll")}
    else {dyn.load("MNPBayes64.dll")}
  }
}
```
Appendix D. Source code documentation

z <- numeric(length(y))
b_post <- matrix(0, nrow=n, ncol=k, dimnames=list(NULL, names(b0)))
if(set_s){set.seed(seed)}
b <- b0
for(i in 1:(n + burnin)){
  # step 1
  if(is.null(mu0)){
    b <- solve(xx, crossprod(x, z))
  } else {
    b <- solve(PA, crossprod(x, z) + IA0_mu0)
  }
b <- b + backsolve(ch_var, rnorm(k))
  if(i > burnin){
    b_post[i - burnin,] <- b
  }
  mu <- crossprod(t(x), b)
  # step 2
  z <- .Call("sample_z_bp", as.integer(y), as.double(mu), as.double(sig), as.double(a))
}
if(.Platform$r_arch == "i386"){dyn.unload("MNPBayes.dll")
} else {dyn.unload("MNPBayes64.dll")
}
b_post_thin <- NULL
if(thinning > 1){
  b_post_thin <- b_post[seq(thinning, n, thinning),]
}
if(thinning > 1){
  b_post <- b_post[seq(thinning, n, thinning),]
}
return(list(b_post = b_post, b_post_thin = b_post_thin, b0 = b0, fit_mle = fit_mle))

Listing D.3: Source R-code for \texttt{mnp.bayes} function

\begin{verbatim}
mnp.bayes <- function(
data_set, n = 50000, burnin = 10000, set_s = TRUE, seed = 7402, thinning = 1, mu0 = NULL, A0 = NULL, nu = NULL, V0 = NULL, b0 = NULL, S0 = NULL)
{
x_vars <- data_set$x_vars
y <- data_set$y
p <- data_set$p
X <- matrix(unlist(data_set$X, use.names = FALSE), ncol = length(x_vars), 
dimnames=list(NULL, x_vars))
#initial settings
l <- length(p) - 1
k <- ncol(X)
z <- numeric(nrow(X))
if(is.null(b0)) {b0 <- numeric(k)}
if(is.null(S0)) {S0 <- diag(l)}
C <- chol(solve(S0))
IA0 <- solve(A0)
\end{verbatim}
Appendix D. Source code documentation

```r
IA0_mu0 <- crossprod(t(IA0), mu0)
ly <- length(y)
b_post <- matrix(0, nrow = n, ncol = k, dimnames = list(NULL, names(x_vars)))
S_post <- matrix(0, nrow = n, ncol = l * l)
if(!is.loaded("sample_z_mnp")){
  if(.Platform$r_arch == "i386"){dyn.load("MNPBayes.dll")
  } else {dyn.load("MNPBayes64.dll")}
}
if(set.seed){set.seed(seed)}
b <- b0
S <- S0
m <- ncol(S)
df <- (2 * (nu + ly) - m + 1) - ((nu + ly) - m + 1):(nu + ly)
# Gibbs sampler
for(i in 1:(n + burnin)){
  # step 1
  S_inv <- crossprod(C)
  mu <- X %*% b
  z <- .Call("sample_z_mnp", as.double(z), as.integer(y), as.double(mu), as.double(S_inv),
              as.integer(p))
  # step 2
  Circ <- matrix(cbind(z, X), nrow = l)
  Circ <- C %*% Circ
  Circ <- matrix(Circ, nrow = nrow(X))
  z_circle <- Circ[, 1]
  X_circle <- Circ[, -1, drop = FALSE]
  XX <- crossprod(X_circle)
  ch_var <- chol(XX + IA0)
  b <- solve((XX + IA0), crossprod(X_circle, z_circle) + IA0_mu0)
  b <- b + backsolve(ch_var, rnorm(k))
  # step 3
  eps <- matrix((z - X %*% b), nrow = l)
  S <- crossprod(t(eps))
  V <- chol2inv(chol(V0 + S))
  T <- diag(sqrt(rchisq(c(rep(1, m)), df)))[lower.tri(T)] <- rnorm((m * (m + 1) / 2 - m))
  U <- chol(V)
  C <- t(T) %*% U
  if(i > burnin){
    CI <- backsolve(C, diag(m))
    b_post[i - burnin,] <- b
    S_post[i - burnin,] <- crossprod(t(CI))
  }
}
if(.Platform$r_arch == "i386"){dyn.unload("MNPBayes.dll")
} else {dyn.unload("MNPBayes64.dll")}
b_post_thin <- NULL
S_post_thin <- NULL
if(thinning > 1){
  b_post_thin <- b_post[seq(thinning, n, thinning), , drop = FALSE]
  S_post_thin <- S_post[seq(thinning, n, thinning), , drop = FALSE]
}
return(list(b_post = b_post, S_post = S_post, b_post_thin = b_post_thin, S_post_thin = S_post_thin, b0 = b0, S0 = S0))
```

The `bp.bayes` function returns a posterior sample for the vector of parameters $\beta$ for the further inference, an initial value of $\beta_0$ according to the value of the input variable `b0`, an MLE, and a MLE estimate of the model for
Appendix D. Source code documentation

the comparison.

The \texttt{mnp.bayes} function returns a posterior sample for the vector of parameters \( \beta \) and a posterior sample for the variance-covariance matrix \( \Sigma \) for the further inference.

We introduce also an auxiliary \texttt{R}-function \texttt{data.reshape} (see Listing D.4), which helps to transform the dataset according to the \textit{MNP} model specifications, and to introduce dummies in the dataset. The function returns augmented data, and, in case of binary model, a formula for the regression to produce a \textit{MLE} estimate of the model.

Listing D.4: Source \texttt{R}-code for \texttt{data.reshape} function

```r
data.reshape <- function(data_set, y_var = "y", x_vars, d_vars = NULL, has_intercept = TRUE, is_mnp = FALSE, a_data_set = NULL, a_vars = NULL, p = NULL) {
  dummify <- function(dsm, x, d, l) {
    if(!is.null(d) && !identical(d, character(0))){
      for(i in 1:length(d)){
        x <- x[x != d[i]]
        d_val <- table(dsm[, d[i]])
        n <- as.numeric(names(d_val))
        r <- (length(d_val) - 1)
        if(r > 1){
          v <- as.vector(d_val)[c(1:r)]
          dm <- matrix(rep(n[c(2:(r + 1))], 1), byrow = TRUE, ncol = r, dimnames = list(NULL, paste(d[i], c(1:r), sep = "")))
          dm[,] <- (dsm[, d[i]] == dm[,])
          dsm <- cbind(dsm, dm)
        }
        x <- c(x, paste(d[i], c(1:r), sep = ""))
      }
    } return(list(dsm = dsm, x = x, d = d))
  }
  m <- matrix(unlist(data_set[c(y_var, x_vars)]), use.names = FALSE), ncol = length(c(y_var, x_vars)), dimnames = list(NULL, c(y_var, x_vars)))
  l <- length(m[, y_var])
  if(is.null(p)) {p <- sort(unique(unlist(data_set[y_var], use.names = FALSE)))}
  n <- length(p)
  if(is.null(d_vars)){
    d <- dsm[, x_vars]
    x_vars <- d$x
    dsm <- dsm
  }
}
```

74
Appendix D. Source code documentation

```r
m <- m[, c(y_var, x_vars), drop = FALSE]
if(has_intercept){
  o <- matrix(rep(1, l), byrow = FALSE, ncol = 1, dimnames = list(NULL,"(Intercept)"))
  m <- cbind(o, m)
  x_vars <- c("(Intercept)", x_vars)
}
if(!is.null(a_vars)) {
  na <- length(a_vars)
  a <- matrix(unlist(a_data_set[a_vars], use.names = FALSE), ncol = na,
    dimnames = list(NULL, a_vars))
  if(nrow(a) != n) {return(list(r_data = as.data.frame(m)))}
  b <- matrix(rep("-a[n]", n), ncol = na, byrow = TRUE)
  a <- a + b[-n, , drop = FALSE]
  if(!is.null(d_vars)){
    dum <- dummify(dsm = a, x = a_vars, d = d_vars, l = n - 1)
    a <- dum$dsm
    a_vars <- dum$x
  }
  a <- a[, a_vars, drop = FALSE]
  a <- rep(1, l) %x% a
  colnames(a) <- a_vars
}
if(!is_mnp) {
  if(!is.null(a_vars)) {
    m <- cbind(a, m)
    x_vars <- c(a_vars, x_vars)
  }
  if(has_intercept){
    f <- paste(y_var, "~", paste(x_vars[x_vars != "(Intercept)"], collapse = "+"))
  } else {
    f <- paste(y_var, "~", paste(c(0, x_vars), collapse = "+"))
  }
  return(list(r_data = as.data.frame(m), reg_formula = f))
} else {
  y <- m[, y_var]
  o <- m
  m <- m[, colnames(m) %in% y_var, drop = FALSE] %x% diag(n - 1)
  colnames(m) <- x_vars <- paste(rep(x_vars, each = n - 1), rep("\_", n - 1), c(1:(n-1)), sep = "")
  if(!is.null(a_vars)) {
    m <- cbind(a, m)
    x_vars <- c(a_vars, x_vars)
  }
  return(list(y = y, r_data = as.data.frame(o), X = as.data.frame(m), p = p, x_vars = x_vars))
}

An example of usage of `bp.bayes` function and its output are given below
(under assumption, that the original data set was previously assigned to
variable `gd`):

```r
> x <- c("GEN", "AGE", "AGE2", "PU", "LAM", "LF", "ZF", "DS")
> rd <- data reshape(data_set = gd$data_set, x_vars = x)
> ds <- rd$r_data
> regf <- rd$reg_formula
> z <- bp.bayes(data_set = ds, reg_formula = regf)
> post_m <- apply(z$mean_post, 2, mean)
```
Appendix D. Source code documentation

> post_m
  (output omitted)

An example of usage of mnp.bayes function and its output are given below (under assumption, that the original data set was previously assigned to variable gd_m):

> x <- c("GEN", "AGE", "AGE2", "PU", "LAM", "LF", "ZF", "DS")
> rd_m <- data.reshape(data_set = gd_m$data_set, x_vars = x, is_mnp = TRUE, has_intercept = TRUE)
> k <- length(rd_m$x_vars)
> mu0 <- rep(0, k)
> A0 <- 200 * diag(k)
> a <- length(rd_m$p) - 1
> nu <- a + 3
> V0 <- nu * diag(a)
> z_m <- mnp.bayes(rd_m, mu0 = mu0, A0 = A0, V0 = V0, nu = nu, n = 50000, burnin = 100000, thinning = 10)
> b_tilde_post <- z_m$b_post_thin / sqrt(z_m$S_post_thin[, 1])
> colnames(b_tilde_post) <- rd_m$x_vars
> S_tilde_post <- z_m$S_post_thin / z_m$S_post_thin[, 1]
> post_mm <- apply(b_tilde_post, 2, mean)
> post_mm
  (output omitted)

The R-function bp.bayes.predict accepts a sample from posterior distribution of $\beta$, a new dataset and the regression formula and produces a sample of posterior predictive probabilities for the every individual in the dataset (see Listing D.5). The size of the sample is as such as one for $\beta$.

Listing D.5: Source R-code for bp.bayes.predict function

```r
bp.bayes.predict <- function(
  b_post,       # a sample from posterior distribution of beta
  reg_formula,  # formula for regression, i.e. y ~AGE+GEN+...  
  data_set,     # a new dataset, data.frame
)
{
  new_fit <- lm(reg_formula, data = data_set, x = TRUE, y = TRUE)
  new_x <- new_fit$x
  predict-utils <- matrix(numeric(length(new_fit$y) * length(b_post[, 1])), ncol = length(b_post[, 1]))
  for(j in 1:length(b_post[, 1])){
    predict-utils[, j] <- crossprod(t(new_x, b_post[j,]))
  }
  predict_prob <- apply(predict-utils, 2, pnorm)
  return(predict_prob)
}
```

76
Appendix D. Source code documentation

An example of usage of `bp.bayes.predict` function and its output are given below (under assumption, that the data set was previously assigned to variable `new_gd`, and the values of variables `regf`, `x`, `z` and `d` were taken from the previous example):

```r
> new_rd <- data.reshape(data_set=new_gd$data_set, x_vars=x)
> new_ds <- new_rd$r_data
> pred_p <- bp.bayes.predict(data_set=new_ds, reg_formula=regf, z$b_post)
> pred_pm <- apply(pred_p, 1, mean)
> pred_pm
(output omitted)
```

The R-function `mnp.bayes.predict` accepts a sample from posterior distribution of $\tilde{\beta}$, $\tilde{\Sigma}$, a new dataset and the regression formula and produces a sample of posterior predictive probabilities for the every individual in the dataset (see Listing D.6).

Listing D.6: Source R-code for `mnp.bayes.predict` function

```r
def mnp.bayes.predict <- function(vol, b_post = NULL, S_post = NULL, data_set = NULL, set_s = TRUE, seed = 7402)
{
  require(mvtnorm)
  draw.eps <- function(Sigma)
    n <- sqrt(length(Sigma))
    S <- matrix(Sigma, ncol = n)
    eps <- rmvnorm(1, sigma = S)
    return(eps)
  draw.Xb <- function(beta, new_X)
    return(new_X %*% beta)
  get.y.post <- function(z)
    1 <- length(z) + 1
    if(all(z < 0))
      return(1)
    else
      return(which.max(z))
  }
  if(set_s){set.seed(seed)}
  k <- nrow(b_post)
  x_vars <- data_set$x_vars
  y <- data_set$y
  p <- data_set$p
  l <- length(p) - 1
  n <- nrow(b_post)
  X <- matrix(unlist(data_set$x, use.names = FALSE), ncol = length(x_vars),
              dimnames = list(NULL, x_vars))
```

77
Appendix D. Source code documentation

n_obs <- nrow(X) / l
ind <- c(1:l)
predict_prob <- matrix(numeric((l + 1) * n_obs), nrow = n_obs)
for(i in 1:n_obs){
  eps <- apply(S_post, 1, draw.eps)
  Xb <- apply(b_post, 1, draw.Xb, new_X = X[ind, , drop = FALSE])
  z_post <- Xb + eps
  y_post <- apply(z_post, 2, get.y.post)
  predict_prob[i, ] <- as.vector(table(factor(y_post, levels = p))) / n
  ind <- ind + l
}
return(predict_prob)

To sample from multivariate normal distribution with given variance-
covariance matrix we use R-package mvtnorm.

An example of usage of mnp.bayes.predict function and its output
are given below (under assumption, that the data set was previously as-
signed to variable new_gd_m, and the values of variables b_tilde_post and
S_tilde_post were taken from the previous example):

> new_rd_m <- data.reshape(data_set = new_gd_m$data_set,
x_vars = x, is_mnp = TRUE, has_intercept = TRUE)
> pred_p_m <- mnp.bayes.predict(data_set = new_rd_m,
b_post = b_tilde_post, S_post = S_tilde_post)
> pred_p_m
(output omitted)

Listing D.7 represents implementation of R-function tab.data, which
produces contingency tables and performs \( \chi^2 \) test for independence between
variables. The tab.data function returns the contingency table and the
results of \( \chi^2 \) test with critical and \( p \)-values.

Listing D.7: Source R-code for tab.data function

```r
tab.data <- function(
data_set #underlying dataset, data.frame
  ,y_var = "y" #name of dependent variable, scalar
  ,x_var = "X" #name of independent variable, scalar
  ,dim_names = c("y", "X") #vector of names for table, characters
  ,alpha = 0.05 #the significance level, scalar
)
{
y_data <- as.vector(unlist(data_set[[toString(y_var)]], use.names = FALSE))
x_data <- as.vector(unlist(data_set[[toString(x_var)]], use.names = FALSE))
dim_names <- as.character(dim_names)
tb <- t(table(y_data, x_data, dnn = dim_names))
r_tot <- Total <- apply(tb, 1, sum)
c_tot <- Total <- apply(cbind(tb, Total))
cdata <- table(rbind(c_tot, Total))
names(dimnames(cdata)) <- names(dimnames(tb))
tot <- c_to[1:length(c_tot)]
c_tot <- c_to[1:length(c_tot)-1]
```

78
Appendix D. Source code documentation

```r
ttb <- (r_tot %*% t(c_tot)) / tot
nu <- (length(r_tot) - 1) * (length(c_tot) - 1)
if(nu == 1){
  x_2 <- sum((abs(tb - ttb)-0.5)^2 / ttb)
} else {
  x_2 <- sum((tb - ttb)^2 / ttb)
}
cv <- qchisq(1-alpha, df = nu)
pv <- 1 - pchisq(x_2, df = nu)
test_t <- matrix(c(x_2, cv, nu, pv, alpha), byrow = TRUE, nrow = 1)
dimnames(test_t) <- list("Test:", c("X2", "chi2", "df", "p-value", "alpha"))
return(list(ctb = ctb, test_t = test_t))
```

An example of usage of `tab.data` function and its output is given below (under assumption, that the data set was previously assigned to the variable `ds`):

```r
> dn <- c("Target", "Gender")
> td <- tab.data(data_set=ds, dim_names=dn, x_var="GEN")
> td$ctb
   Target
  Gender  0  1 Total
    0  1188 2576 3764
    1  1450 2554 4004
  Total 2638 5130 7768
> td$test_t
                 X2   chi2    df  p-value    alpha
Test:  18.52928 3.841459  1 1.673141e-05  0.05
```

Listing D.8: Source C-code for `poissbin.dll`

```c
#include <Rmath.h>
#include <Rinternals.h>
#include <R.h>

SEXP d_sumbern(SEXP p, SEXP fact){
  int n = length(p), l = 2, i, j, m = 0, f = INTEGER(fact)[0];
  double *pv, *tpmf;
  long double pm[n + 1], prod = 1.0;
  SEXP pmf; PROTECT(pmf = allocVector(REALSXP, n + 1));
  pv = REAL(p); tpmf = REAL(pmf);
  for(i = 0; i < n; i++){
    if(pv[i] > 0) m += 1;
  }
  long double k[m], X[m + 1];
  m = 0;
  for(i = 0; i < n + 1; i++){
    X[m] = 0;
    if(i < n){
      if(pv[i] > 0){
        prod *= pv[i];
        k[m] = - (1 - pv[i]) / pv[i];
        m += 1;
      }
    }
  }
  // Further code...

  // Return...
}
```

79
Listing D.8 implements two approaches for calculation probability mass function for Poisson-Binomial distribution — direct (through the polynomial coefficients) and by Monte Carlo simulation. The compiled library poissbin.dll requires a “wrap-up” R-function, which is presented in Listing D.9.

Listing D.9: Source R-code for poiss.binom.pmf function
Appendix D. Source code documentation

The `poiss.binom.pmf` function returns `pmf` of *PBD*. As an input for the function `poiss.binom.pmf` we can use posterior predictive probabilities from the function `bp.bayes.predict`. An example of usage of `poiss.binom.pmf` function and its output is given below

```r
> p <- c(0.5, 0.6, 0.7)
> poiss.binom.pmf(p = p, type = "d")

P(Z=0) P(Z=1) P(Z=2) P(Z=3)
P  0.06  0.29  0.44  0.21
```
Bibliography


Bibliography


