VILNIUS UNIVERSITY
FACULTY OF MATHEMATICS AND INFORMATICS
MODELLING AND DATA ANALYSIS
MASTER'S STUDY PROGRAMME

Gross Merchandise Value Prediction for a Buyer in e-commerce

Master's thesis

Author: Monika Žilinskienė
VU email address: monika.zilinskiene@mif.stud.vu.lt
Supervisor: (Assoc. Prof. Dr. Jurgita Markevičiūtė)

Vilnius

2022

Abstract

In order to make informed marketing decisions on customers' acquisition, retention strategies and amount worth to spend on them, business aim to know Gross Merchandise Value customer will create in a given future. In this thesis, besides already researched methods, the application of two- stage model using gradient boosted decision tree and two-stage feed forward neural network, where first stage is classification model to predict if customer will churn in a given future period and second stage is predicting GMV for those whom churn prediction is negative. Further, we propose a multi-output feed forward neural network for combined results: classification for churn and regression for GMV. Finally, models were created on real life customer-to-customer marketplace dataset and evaluated in between using the mean absolute error (MAE) and root-mean-squared-error (RMSE) metrics.

Keywords: GMV, CLTV, e-commerce, XGBoost, FFNN, multi-output FFNN.

Santrauka

Norint atlikti informuotus, duomenimis paremtus sprendimus apie investicijų į vartotojų pritraukimo, išlaikymo strategijas, verslai turi žinoti Bendrąją Pirkėjo Vertę duotam ateities laikui. Šiame darbe, be jau prieš tai išbandytų ir aprašytų metodų, pritaikome dviejų-stadijų modelį naudodami sprendimų medį bei tiesioginio sklidimo neuroninį tinklą, kur pirma stadija yra klasifikacijos modelis nustatyti ar vartotojas bus aktyvus ateityje, antroje stadijoje regresijos modelis prognozuoja Bendrąją Pirkėjo Vertę tų vartotojų, kurių prognozė pirmoje stadijoje yra teigiama (vartotojas bus aktyvus). Toliau yra siūlomas kelių-rezultatų tiesioginio sklidimo neuroninis tinklas norint gauti du rezultatus: aktyvumą ir Bendrąją Pirkėjo Vertę. Galiausiai modeliai yra pritaikomi realiems duomenims ir palyginami tarpusavyje naudojant suvidurkintą absoliučią paklaidą bei šaknį iš vidutinės kvadratinės paklaidos.

Raktažodžiai: Bendroji Pirkėjo Vertė, BPV, vartotojo gyvenimo vertė, e-prekyba, sprendimų medis, neuroninis tinklas.

Acronyms

AUC - Area Under ROC (receiver operating characteristic) curve
BG/NBD - Beta-geometric/Negative Binomial Distribution
BTYD - "buy-till-you-die" models family
C2C - Customer-to-Customer business model
CLTV - Customers' Lifetime Value
CLTV - Customers' Lifetime Value
DL - Deep Learning
DNN - Deep Neural Network
Deep-MLP - Deep Multilayer Perceptrons
FAQ - Frequently Asked Questions
FFNN - Feedforward Neural Networks
GBM - Gradient Boosting Machine
GGG - Gamma-Gamma-Gamma
GMV - Gross Merchandise Value
HB - Hierarchical Bayes
MAE - Mean Absolute Error
MBG/NBD - Modified Beta-geometric/Negative Binomial Distribution
MCMC - Markov-Chain-MonteCarlo
MLE - Maximum Likelihood Estimation
MLP - Multilayer Perceptron
MSE - Mean Squared Error
NBD - Negative Binomial Distribution
NLP - Natural Language processing
NRMSE - Normalized Rooted Mean Squared Error
RF - Random Forest
RFM - Recency, Frequency, Monetary
RMSE - Root-Mean-Squared-Error
RNN - Recurrent Neural Network
SMOTE - Synthetic Minority Oversampling
XGBoost - Extreme Gradient Boosting
ZILN - Zero-Inflated Lognormal distribution

Contents

1 Introduction

To stay in line with the financial planning, businesses need to predict the inflow of revenue in the future. The longer the time horizon could be predicted, the better. While knowing the future incomes is essential to the business and is no news to any of them, more and more companies want to know not overall but each customer's future value. Such predictions are called customer lifetime value and are calculated by multiplying profit margin, and customers spendings over a selected period. In customer-to-customer (C2C) business, such spending is called gross merchandise value (GMV).

Usually, GMV is the main and only volatile component of lifetime value calculation. With its accurate prediction, business owners will be able to make informed decisions on how much is worth to spend into the specific user and still stay profitable. Such information is critical when deciding on customer relationship management, or even acquisitions of the new members as customers could be clustered by the forecasted GMV and different promotional offers, deals or personalized messages could be allocated by that.

As GMV is the main part of customers' lifetime value calculation, the reviewed literature is mainly focused on the latter metric. Traditional and still much in use models for customer lifetime value prediction are probabilistic (also called "buy-till-you-die" (BTYD)) models which are developed on only transaction information of the customer, that is why with the growing quantity of data available (especially in e-commerce business), machine learning techniques are becoming more used in the field. In our case and any other non-contractual market, lifetime value is highly skewed (many customers are one-time buyers with zero further spend); hence researchers proposed two-stage models [21] [5] where using random forest technique customers were classified whether they will churn and, if not - model for prediction of lifetime value is applied. The newest papers introduce deep learning methodology for a lifetime value forecasting problem [22], [20], [3] but it is not yet a highly used technique in the field.

The aim of the thesis is to adapt a model for buyers' gross merchandise value for 360 days forecast. As a baseline, the probabilistic model (BG/NBD with Gamma-Gamma submodel) is presented as it is still highly used in the field and is a golden standard for such problems. Since two-stage models produced valid results in previous researchers, in thesis new approaches with such techniques are proposed: two-stage model using gradient boosted decision tree and two-stage feed forward neural network. Further, we propose a multi-output feed forward neural network for combined results: classification for churn and regression for GMV. To be able to compare models in-between one-stage, one-output feed forward neural network and gradient boosted decision tree were developed as well. Models were evaluated in between using the mean absolute error (MAE) and root-mean-squared-error (RMSE) metrics.

The dataset from the customer-to-customer e-commerce platform will be used. While predicting lifetime value for two-sided platforms, it can be calculated on either the seller or the buyer, but as the GMV is based on transactions that are made by both sides (buyer and seller), calculations should not be done for both. The decision was made to predict GMV on the buyers as it is more similar to the literature review. The dataset contains information about buyers' demographics, purchase history, and overall activity in the platform during the feature period. For every user feature period could be different as it was artificially created by adding random numbers from 1 to 360 to the customer's first purchase date. This was done in order to have a diverse dataset and be able to use models for business needs instantly. The aim of the model is to predict GMV spending in the target period, which is equal to 360 days after the feature period ends.

In this thesis, related work in the field will be reviewed, then the broader presentation of the proposed methodology will be explained. The further dataset will be presented and used to demonstrate proposed models. The latest sections will conclude the results of the thesis.

2 Literature review

Prediction of a specific customer's gross merchandise value (GMV) is a complicated problem and methods of solving it vary from statistical models to deep learning algorithms as there is no single established practice. Moreover as we are interested in the non-contractual market where customers are not linked to the platform through subscription and it is unknown if the customer is alive or already churned, the task becomes even more problematic. As GMV is a leading variable of customers' lifetime value (CLTV) calculation, the reviewed literature is mainly focused on the latter metric. Further in this section, various researches and methodologies will be reviewed - probabilistic models, decision trees and neural networks. Probabilistic models are still highly used in a field as it requires only a few variables and small computational resources. As e-commerce nowadays have way more than transactional data about customers, the decision trees became more used in a field as it could include additional explanatory variables and forecast CLTV of visitors with even zero transactions. Nowadays neural networks approaches proved its applicability in practice in various competitions thus it is only natural that researchers try to apply it to CLTV prediction problems as well.

2.1 Probabilistic models

The standard approach - regression - is problematic and inadequate method in order to model CLTV: the regression-type models are ad-hoc, there is no theoretical story why some of explanatory variables should impact CLTV and are designed to predict behavior in the next period which is irrelevant for CLTV as businesses are interested in further periods than only next one [10].

The golden standard for CLTV prediction in non-contractual markets (such as e-commerce) are probabilistic, called "buy-till-you-die" (BTYD) models family. Some of the BTYD models can be efficiently estimated via means of maximum likelihood estimation (MLE), such as:

- Negative Binominal Distribution [9] - NBD;
- Pareto/NBD [19] - Pareto/NBD;
- Beta-geometric/NBD [11] - BG/NBD;
- Modified Beta-geometric/NBD [2] - MBG/NBD;

Distribution functions are provided in Appendix. NBD model assumes heterogenous purchasing process but does not take possibility of customer defection into account. The Pareto/NBD model solves this problem with added dropout process and until this day is considered to be a golden standard of probabilistic models family for non-contractual market. BG/NBD model adjusts Pareto/NBD assumptions in order to speed up computation. Concerning inconsistency between customer's with many and without any purchases, MBG/NBD was introduced with assigned probability of dropout after every purchase.

Besides maximum likelihood models, others have parameter estimation from Markov-Chain-MonteCarlo (MCMC) simulation. Such an approach helps with the behavioral assumptions behind MLE models. MCMC models include:

- Hierarchical Bayes Pareto/NBD [15]- Pareto/NBD (HB);
- Abe's variant of Pareto/NBD (with and without incorporated covariates) [1] - Pareto/NBD (Abe);
- Pareto/Gamma-Gamma-Gamma [18] - Pareto/GGG

Pareto/NBD (HB) has the same assumptions as the classical Pareto/NBD model but estimates parameters with MCMC simulation. Pareto/NBD (Ape) takes more advantage of MCMC and relaxes the independences between purchase and dropout process, plus it is capable of incorporating customer covariates. Finally, Pareto/GGG allows a varying degree of regularity within the transaction timings.

The models described above forecast the expected number of transactions in the future period but not CLTV (same for GMV). That is why the Gamma-Gamma spending submodel was introduced in [12] and further clarified in [13]. The model assumes that a customer's transaction value varies randomly around the average order value, which differs across customers but stays the same for a given specific customer. Also, the distribution of average transaction values across customers is independent of the transaction process.

The comparative analysis of various probabilistic lifetime value models in online shopping was done by Pavel Jasek, et al. [14] where models described above were included into the comparison together with Status Quo model where lifetime value was assumed as zero (so was GMV) if user had not make purchase for more than a year or if user is active in the last year the assumption was made that he will not churn during the forecast period and will purchase the next week with the same value as his average weekly purchase in the last year. For model evaluation dataset from several e-commerce shops was analyzed and results revealed that the models do not perform stable per different shops so there is no one perfect model for every case. Yet Pareto/NBD showed mostly stable results per various datasets while BG/NBD had highest average accuracy while still having a solid relative standard deviation. Also the Status Quo model overestimated profit a lot, all probabilistic models gave more realistic forecasts.

Similar experiment to predict CLTV from store loyalty program scanner data showed that in comparison of NBD, Pareto/NBD and BG/NBD models the latter two outperformed first as it does not consider customer's inactivity [4]. BG/NBD performed satisfactorily in weekly purchases frequency prediction while Pareto/NBD underestimated it.

Traditional models of CLTV are developed on transactional information of the customer, frequently summarized as RFM characteristics where variables are:

- recency (time of most recent purchase)
- frequency (count of past purchases)
- monetary (average purchase amount)

The greatest advantages of probabilistic models are that they could be made with a few explanatory variables (RFM) and considerably small computational resources. However, they are usually based on specific assumptions on distributions and could lead to low accuracy predictions when the assumptions are not satisfied. Also, nowadays e-commerce businesses have way more than transactional data about a customer - including demographic and activity on the platform data such as screen views, clicks, sessions, checked products, search history. Further, machine learning methods which do not have such assumptions and could incorporate additional features will be presented.

2.2 Decision trees and two-stage approach

As machine learning methods can include way more explanatory variables into the model they perform significantly better than traditional methodologies. More importantly, probabilistic models cannot predict new customers (without any transactional data yet) purchase power thus even if it is a great start for predicting GMV, machine learning methods have way more advantage.

In Vanderveld et al. [21] research, two-stage Random Forest (RF) models were built with features such as engagement (email, app opens, downloads), user experience (available nearby deals, refunds, raised customer service tickets and solving experience, average shipping time), purchase history, demographics in order to predict CLTV for three time windows: short, medium, long. Researchers note that different features might have different effects on different users: purchase history was found to be the most important variable for the active buyers but email engagement score was the main feature for one-time buyers. Because of it, researchers manually divided users into six cohorts according to their buying history. Various algorithms were tested and evaluated by root-mean-squared error (RMSE) and Spearman correlations between actual and predicted values. RF performed the best thus the two-stage RF model was built:

1. Binary classification for whether or not user will make a purchase in a given time window
2. Predict the value of purchase of users who were predicted to buy in first step

The two-stage approach was introduced due to the distribution of CLTV which is highly left-skewed as a relatively big part of customers are one-time buyers. At some point similar to the probabilistic models as Pareto/NBD also do two separate steps in order to predict churn rate first and value of purchases later yet there purchase values follows an independent Gamma/Gamma distribution and it relies on a shaky assumption that value of order is independent of purchase frequency which is not always true as user who made one high value purchase might not come back to purchasing for longer time than frequent purchases who spend less every time.

Two-stage RF were used as baseline models for CLTV prediction in Chamberlain et al. [5] publication thus here authors introduced readers to neural embeddings - a technique which originally is used in Natural Language processing (NLP) as representation of words instead of one-of-k (one-hot) and Deep Neural Network (DNN) for CLTV prediction problem. Embedding captures the meaning of the input by placing frequently, in the same context appearing objects close to each other in the embedding space. The closeness is determined by cosine similarity. Authors learn embeddings using customer product views and add it to the RF model as additional features. Also they train additional hybrid models where RF is replaced by DNN (in specific deep feed-forward neural network) and embeddings of Handcrafter features are used. The experiments to compare the models were simplified to binary classification (churn or first stage) in order to have more interpretable predictions and metrics of performance. Results demonstrate that RF with additional embedding features obtained significant AUC uplift in comparison of baseline RF. DNN also improved baseline model performance but the expenses needed to train such models in live systems exceed the improvement.

In this section, experiments on CLTV prediction using decision trees and two-stage approach were described as well as the reader was introduced to neural networks application for the problem. The latest reviewed research showed the potential of deep neural networks but remarked high computational requirements as well. In the next section Deep Learning (DL) methods will be explored further.

2.3 Neural Networks

In the last few years, deep learning methods have become more and more used in business. Mostly because it could analyze more complex data types - visual imagery for example. As for CLTV prediction we have tabular data thus it is not so common to use neural networks for such problems. Either way, latter explorations in CLTV prediction topics are focused on deep learning applications.

In the last section Chamberlain et al. [5] research was presented where it was not the case and even though feed-forward neural networks improved the results, it was not deployed to the business due to higher computational requirements. Wang et al. [22] accentuate new customers' problems - it is not possible to predict their LTV with probabilistic model approach as frequency and recency are identical in the dataset. Therefore a deep feed-forward neural network is used. Even if the model type is same as in [5], here authors propose a mixture loss derived from zero-inflated lognormal (ZILN) distribution. The reasoning is that a significant part of customers are one-timers as well as the LTV for returning customers is volatile, with highly skewed distribution. While Mean Squared Error (MSE) loss is popular in regression modeling, in the CLTV case it is not a great fit due to the specifics of data which MSE ignores. Moreover MSE is highly sensitive to outliers. Proposed model was evaluated on a couple of public datasets and according to Spearman's Correlation metric DNN model with ZILN loss performs better than the same model with MSE loss by almost 24-25% on two different datasets.

Another non-contractual marketplace example is the free-to-play game industry where it is crucial to identify high-value users as those 20% of users make 80% of the revenue. Sifa et al. [20] emphasize the imbalance of LTV and suggest to use synthetic minority oversampling (SMOTE) [6] technique which creates a new minority class point in between of other minority examples. For the experiment, authors take a free-to-play games dataset and predict LTV for 360 days from 7 days of user data and adapt such models: Random Forests, Linear Regression, Decision Trees and Deep Multilayer Perceptrons (Deep-MLP) with and without SMOTE augmentation. According to Normalized Rooted Mean Squared Error (NRMSE) which was chosen to quantify the fit, the Deep-MLP model with SMOTE augmentation works best for the given dataset and the SMOTE improves high-value users prediction by six percent.

In all researches discussed in this chapter authors took the data as aggregated on customer's level. Bauer and Jannach [3] suggest to consider it as a time series forecasting problem (value customer generate per one time point) as then sequential structure of data is preserved. The method is based on encoder-decoder sequence-to-sequence recurrent neural networks (RNNs) with augmented temporal convolutions combined with gradient boosting machines (GBMs). To enrich the model, embeddings are created from customer's purchases. Authors compare RMSE from Autoregressive Moving Average (ARMA), BG/NBD, Markov-chain, RF, GBM, Sequence-to-sequence RNN (S2S), Stacking of our GBM and Sequence-To-Sequence RNN (GMB-S2S) models on two different e-commerce business datasets. The order of models in sequence before reflects the performance of them from worst to best. The sequence-based model leads to the highest results and stacking it with the GBM model further improves accuracy (1% smaller RMSE and 10% smaller MAE). Embedding alone improves the model by relatively 17%.

3 Methodology

The aim of the thesis is to adapt a model for customer's GMV prediction for 360 days. In the last chapter the methods used in business in order to predict customer's lifetime value were reviewed. Same methodologies could be applied to GMV prediction as it is the main, volatile variable of CLTV of each customer:

$$CLTV_i = GMV_i \cdot ProfitMargin$$

In this chapter models which will be adapted to the selected dataset is reviewed. The metric for model evaluation is discussed as well. The models are:

1. BG/NBD model with Gamma-Gamma submodel - probabilistic models are still a popular approach in business, thus it will work as a baseline model in the experimental part of this thesis

2. Gradient Boosted Decision Tree:

   2.1 One-stage approach - regression model to predict GMV

   2.2 Two-stage approach - first classification model is run to predict customer's churn, regression model for customer's GMV is only run when churn prediction is negative. For customers who were predicted to churn, GMV prediction is set as zero

3. Deep Neural network:

   3.1 Feed forward neural network

   3.2 Two-stage approach - feed forward neural network is built to classify customers churn, then same methodology is applied as regression model when the previous predictions was negative (customer will not churn). For customers who were predicted to churn, GMV prediction is set as zero

   3.3 Multi output - feed forward neural network with single input but two outputs - one for customer's probability to churn, another for their GMV prediction

In the reviewed literature we had examples of BG/NBD and Gamma-Gamma, Feed Forward Neural Network application to the real life dataset hence according to our knowledge the other methods are not before adapted to CLTV prediction problem.

3.1 BG/NBD with Gamma-Gamma submodel

The "buy-till-you-die" (BTYD) family models are statistical models created to capture customer's behavior in non-contractual market businesses. The models design two processes - one is to model the rate at which customer churns, another predicts the rate of customer's purchase frequency [11]. The first such model was introduced by Schmittlein in 1987 [19] and was called Pareto/NBD as it models the dropout purchase as Pareto Type II distribution and frequency of purchases as negative binomial distribution. In order to speed up computation the BG/NBD model was introduced [12] which is based on the almost same assumptions as Pareto/NBD and shows similar results. The last model was chosen to work within the experimental part of this thesis hence it will be reviewed more broadly in this chapter.

In order to predict a customer's GMV for the upcoming year it is not enough to model the number of purchases for the target period. Consequently, Gamma-Gamma model to model average GMV per purchase for every customer was designed [13]. The model will be reviewed in the next chapter as well.

3.1.1 BG/NBD

Beta-Geometric (BG) model assumes that customer churn action occurs instantly after the purchase is made. While Pareto/NBD model is same in most aspects, it assumes that customers could churn at any moment and it does not depend on actual purchase time. This chapter is based on [11] paper where BG/NBD method was introduced.

BG/NBD model is based on such assumptions [11]:

1. If a customer is active, the number of purchases follows a Poisson process with purchase rate λ. In other words, assumption is that time between transactions is distributed exponential with the purchase rate λ;

$$f(t_j|t_{j-1};\lambda) = \lambda e^{\lambda(t_j - t_{j-1})}.$$

2. Heterogeneity in λ follows a gamma distribution with probability density function (pdf) [11]

$$f(\lambda|r,\alpha) = \frac{\alpha^r \lambda^{r-1} e^{-\lambda\alpha}}{\Gamma(r)}, \lambda > 0. \tag{1}$$

3. Probability of customer churning after every purchase is equal to p.Then the point of customer's churn is distributed per transactions according to shifted geometric distribution with probability mass function (pmf) [11]

$$P(\text{inactive immediately after } j_{th} \text{ transaction}) = p(1-p)^{j-1}, j = 1, 2, 3, \ldots$$

4. Heterogeneity in p follows beta distribution with pdf [11]:

$$f(p|a,b) = \frac{p^{a-1}(1-p)^{b-1}}{B(a,b)}, 0 \leq p \leq 1, \tag{2}$$

where $B(a,b)$ is the beta function and could be expressed as gamma function: $B(a,b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$

5. Both λ and p vary independently across customers

For BG/NBD model (applicable to Pareto/NBD as well) the only data points of customer needed are frequency and recency:

- x - number of repeat purchases since the first purchase (frequency)
- t_x - time between first and last customer's order in weeks (recency)
- T - time period to be considered since the first purchase

Model development at individual level

Derivation of the Likelihood Function In consideration of having customer with x as number of purchases made at time stamps $t_1, t_2, t_3, ..., t_x$ in period $(0, T]$ and the before discussed assumptions we get that individual-level likelihood function is [11]

$$L(\lambda, p|X = x, T) = (1-p)^x \lambda^x e^{-\lambda T} + \delta_{x>0} p(1-p)^{x-1} \lambda^x e^{-\lambda t_x}, \tag{3}$$

where $\delta_{x>0} = 1$ if $x > 0$, 0 otherwise.

Derivation of $P(X(t) = x)$ Let's say $X(t)$ stands for the number of purchases made by a customer in a selected time period of length t and T_x is random variable standing for the time of x_{th} purchase. The fundamental dependence between inter-event times and number of these events is$X(t) \geq x \iff T_x \leq t$. Having a third assumption in mind, the expression for $P(X(t) = x)$ [11]:

$$\begin{aligned} P(X(t) = x) = {} & P(\text{active afteer } x_{th} \text{ purchase} \cdot P(T_x < t \text{and} T_{x+1} > t) \\ & + \delta_{x>0} \cdot P(\text{becomes inactive after } x_{th} \text{ purchase} \cdot P(T_x \leq t). \end{aligned}$$

Assuming that time slots between purchases follows exponential distribution, we have that "$P(T_x < t\text{and}T_{x+1} > t)$ is simply the Poisson probability that $(X(t) = x)$, and $P(T_x \leq t)$ is the Erlang-x" [11] cumulative distribution function. Hence,

$$P(X(t) = x|\lambda, p) = (1-p)^x \frac{(\lambda t)^x e^{-\lambda t}}{x!} + \delta_{x>0} p(1-p)^{x-1} \Big[1 - e^{-\lambda t} \sum_{j=0}^{x-1} \frac{(\lambda t)^j}{j!} \Big]. \tag{4}$$

Derivation of $E[X(t)]$ As per first assumption, number of transactions follows a Poisson process, thus if customer is active at time point t, the$E[X(t)] = t$. It gets trickier when a buyer is not active anymore at time point τ, then $P(\tau > t) = e^{\lambda p t}$. In that case the probability density function of churn is $g(\tau|\lambda, p) = \lambda p e^{-\lambda p \tau}$. From here, the expected number of purchases per period t is [11]

$$E(X(t)|\lambda, p) = \lambda t \cdot P(\tau > t) + \int_0^t \lambda \tau g(\tau|\lambda, p) d\tau = \frac{1}{p} - \frac{1}{p} e^{-\lambda p t}. \tag{5}$$

Model development for Randomly-Chosen Individual Formulas given in previous section are conditional on transaction rate and churn probability p. Given (3) and distributions of λ (1) and p (2), the likelihood function for a randomly-chosen buyer is [11]:

$$L(r, \alpha, a, b|X = x, t_x, T) = \frac{B(a, b+x)}{B(a,b)} \frac{\Gamma(r+x)\alpha^r}{\Gamma(r)(\alpha+T)^{r+x}} + \delta_{x>0} \frac{B(a+1, b+x-1)}{B(a,b)} \frac{\Gamma(r+x)\alpha^r}{\Gamma(r)(r+t_{x)}^{r+x}}. \tag{6}$$

The model parameters (r, α, a, b) can be estimated via the maximum likelihood method as following: the N - sample of customers. Every customer i had $X_i = x_i$ purchases in selected period $(0, T_i]$. The last purchase has been bought at timestamp t_{x_i}. Then sample log-likelihood function (5) [11] can be maximized using standard numerical optimization routines

$$LL(r, \alpha, a, b) = \sum_{i=1}^{N} \ln[L(r, \alpha, a, b|X_i = x_i, t_{x_i}, T_i]. \tag{7}$$

With expectation of (4) the probability of having x transactions in time period of length t is [11]

$$P(X(t) = x|r, \alpha, a, b) = \frac{B(a, b+x)}{B(a,b)} \frac{\Gamma(r+x)}{\Gamma(r)x!} \Big(\frac{\alpha}{\alpha+t}\Big)^r \Big(\frac{t}{\alpha+t}\Big)^x + \delta_{x>0} \frac{B(a+1, b+x-1}{B(a,b)} \Big[1 - \Big(\frac{\alpha}{\alpha+t}\Big)^r \Big\{ \sum_{j=0}^{x-1} \frac{\Gamma(r+j)}{\Gamma(r)j!} \Big(\frac{t}{\alpha+t}\Big)^j \Big\}\Big]. \tag{8}$$

And with expectation of (5) gives us the that expected number of purchases during the time period of length t is [11]

$$E(X(t)|r, \alpha, a, b) = \frac{a+b-1}{a-1} \Big[1 - \Big(\frac{\alpha}{\alpha+t}\Big)^r {}_2F_1\Big(r, b; a+b-1; \frac{t}{\alpha+t}\Big)\Big]. \tag{9}$$

Here ${}_2F_1(\cdot)$ is Gaussian hypergeometric function (see Appendix). The requirement of a single evaluation of Gaussian hypergeometric function is only used after the likelihood function has been maximized.

Finally, for BG/NBD model to be used, the following expression is used to model expected number of transactions for a future period of length t for a specific customer with past behavior $(X = x, t_x, T)$ [11]:

$$E(Y(t)|X = x, t_x, T, r, \alpha, a, b) = \frac{\frac{a+b+x-1}{a-1} \Big[1 - \Big(\frac{\alpha+T}{\alpha+T+t}\Big)^{r+x} {}_2F_1(r+x, b+x; a+b+x-1' \frac{t}{\alpha+T+t}\Big]}{1 + \delta_{x>0} \frac{a}{b+x-1} \Big(\frac{\alpha+T}{\alpha+t_x}\Big)^{r+x}}. \tag{10}$$

As before ${}_2F_1(\cdot)$ is Gaussian hypergeometric function (see Appendix).

3.1.2 Gamma-Gamma model

In order to calculate customers' GMV for 360 days, it is not enough to predict the count of purchases. The average purchase value is needed as well. For it the gamma-gamma "spend" model is used, presented by Fader et al. [12]. Following chapter is based on [13] paper.

The assumptions for gamma-gamma "spend" model are:

- The value of specific customers' transactions varies randomly around that individual customers' average transaction value
- Average transaction value is same for a given customer but varies across customers

- Distribution of average purchase value per different customers is independent of the transaction process

The needed data to apply gamma-gamma model for specific buyer is:

- x - number of repeat purchases since the first purchase (frequency)
- $\overline{z}$ - average value of transactions (monetary value)

$$\overline{z} = \sum_{i=1}^{x} z_i/x,$$

z_i here stands for the value of each observed purchase with $i = 1, 2, \ldots, x$.

$\overline{z}$ is inexact estimation of unobserved mean transaction value ζ . The aim is to conclude $E(Z|\overline{z}, x)$. First step is given x transactions, derive the distribution of $\overline{z}$.

Model development Due to right side skewness of spend data, z_i is distributed with gamma distribution [8] with shape parameter p and scale parameter ν. Specifically:

1. Assume $z_i \sim gamma(p, \nu)$ with $E(Z_i|p, \nu) = \zeta = p/\nu$
   1.1 due to convolution properties of gamma, it follows that sum of spend per x transactions is distributed $gamma(px, \nu)$.
   1.2 due to scaling property of gamma, it follows that $z_i \sim gamma(px, \nu x)$
2. Assume $\nu \sim gamma(q, \gamma)$

From these we get the gamma-gamma (GG) model of spend per purchase.

Deriving $f(\overline{z}|x)$ Given the assumptions above, the distribution of $\overline{z}$ given x is [13]

$$\begin{aligned} f(\overline{z}|p,q,\gamma;x) &= \int_0^{\infty} \frac{(\nu x)^{px}\overline{z}^{px-1}e^{\nu x\overline{z}}}{\Gamma(px)} \frac{\gamma^q \nu^{q-1} e^{-\gamma\nu}}{\Gamma(q)} d\nu \\ &= \frac{\overline{z}^{px-1}x^{px}\gamma^q}{\Gamma(px)\Gamma(q)} \int_0^{\infty} \nu^{px+q-1} e^{-(\gamma+x\overline{z})\nu} d\nu \\ &= \frac{\Gamma(px+p)}{\Gamma(px)\Gamma(q)} \frac{\overline{z}^{px-1}x^{px}\gamma^q}{(\gamma+x\underline{z})^{px+q}} \quad (11) \\ &= \frac{1}{\overline{z}B(px,q)} \Big(\frac{\gamma}{\gamma+x\overline{z}}\Big)^q \Big(\frac{x\overline{z}}{\gamma+x\overline{z}}\Big)^{px}. \quad (12) \end{aligned}$$

Deriving $f(\zeta)$ As mentioned before, ζ stands for the unobserved average purchase value and equals to p/ν. Due to second assumption (see Model Development paragraph), the unobserved average purchase value is viewed as random variable Z with realization ζ.

For $\zeta = h(\nu)$, we have [13]

$$f_\zeta(\zeta) = \Big|\frac{d}{d\zeta}h^{-1}(\zeta)\Big| f_\nu(h^{-1}(\zeta)). \tag{13}$$

Due to $\zeta = p/\nu \Rightarrow \nu = p/\zeta$, $d\nu/d\zeta = -p/\zeta^2$. Therefore [13],

$$f(\zeta|p,q,\gamma) = \frac{p}{\zeta^2}\frac{\gamma^q\left(\frac{p}{\zeta}\right)^{q-1}e^{-\gamma\frac{p}{\zeta}}}{\Gamma(q)} = \frac{(p\gamma)^q\zeta^{-q-1}e^{-\frac{p\gamma}{\zeta}}}{\Gamma(q)}. \tag{14}$$

Deriving $E(Z|\overline{z}, x)$ Finally we come to the individual customer's average purchase value, denoted as $E(Z|\overline{z}, x)$. Due to Bayes' theorem (see Appendix),

$$\begin{aligned} g(\nu|p,q,\gamma;\overline{z},x) &= \frac{f(\overline{z}|p,\nu;x)g(\nu|q,\gamma)}{f(\overline{z}|p,q,\gamma;x)} \\ &= \frac{(\nu x)^{px}\overline{z}^{px-1}e^{-\nu x\overline{z}}}{\Gamma(px)}\frac{\gamma^q\nu^{q-1}e^{-\gamma\nu}}{\Gamma(q)}\Big/\frac{\Gamma(px+q)}{\Gamma(px)\Gamma(q)}\frac{\overline{z}^{px-1}x^{px}\gamma^q}{(\gamma+x\overline{z})^{px+q}} \\ &= \frac{(\gamma+x\overline{z})^{px+q}\nu^{px+q-1}e^{-\nu(\gamma+x\overline{z})}}{\Gamma(px+q)}. \end{aligned} \tag{15}$$

In other words, the posterior distribution of ν is gamma with shape parameter $px + q$ and scale parameter $\gamma + x\overline{z}$ [13]. It follows that

$$\begin{aligned} E(Z|p,q,\gamma;\overline{z},x) &= \frac{p(\gamma+x\overline{z})}{px+q-1} \\ &= \Big(\frac{q-1}{px+q-1}\Big)\frac{p\gamma}{q-1} + \Big(\frac{px}{px+q-1}\Big)\overline{z}. \end{aligned} \tag{16}$$

The conditional expectation of Z is a weighted average of the population mean $E(Z)$ and observed average purchase value $\overline{z}$. Hence with growing number of transactions x, weight on $E(Z)$ decreases while weight placed on customer's observed average value increases. Meaning that for customers without repeated purchases the weight is 1 and the unobserved transactions mean value is equal to $E(Z) = \frac{p\gamma}{q-1}$.

3.2 Gradient boosted Decision Tree

Next model adapted for the selected dataset in the thesis is Extreme Gradient Boosting (XGBoost). XGBoost is a decision-tree-based implementation of gradient boosting framework introduced by Chen et al. [7]. Since its introduction, XGBoost started to dominate applied Machine Learning competitions especially when a given dataset is structured or tabular. In the reviewed literature researchers used the RF method for CLTV forecasting hence XGBoost application for the problem is, as far as we know, a novelty in a field. In this section, readers will be introduced to XGBoost and how it differs from the RF model.

Boosting was created to get strong learners by converting a set of weak learners. In classification tasks, the strong learner has a low (close to 0) error rate while weak learner is somewhat lesser than 0.5. Gradient Boosting is a boosting algorithm but instead of adding new models to correct the errors, new models are created to predict errors of previous models. All models added together make a final prediction. In order to minimize the loss when adding new models, a gradient descent algorithm is used, explaining the name of the algorithm. XGBoost might be called as regularized gradient boosting as it incorporates a regularized model in order to control overfitting which results in higher accuracy.

3.2.1 XGBoost

The algorithm and advantages The novelty of XGBoost is a Newton boosting optimization approach (also called second-order gradient boosting or Hessian boosting). The generic XGBoost algorithm is given further referring to [16].

Given data set $\mathcal{D}$, loss function L, a base learner $\mathcal{L}_\Phi$, number of iterations M and the learning rate η:

1. initialize $\hat{f}^{(0)}(x) = \hat{f}_0(x) = \hat{\theta}_0 = \arg\min\limits_{\theta} \sum_{i=1}^{n} L(y_i, \theta)$;

2. For $m = 1, 2, \dots, M$ do:

$$\begin{aligned}
\hat{g}_m(x_i) &= \Big[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\Big]_{f(x)=\hat{f}^{(m-1)}(x)};\\
\hat{h}_m(x_i) &= \Big[\frac{\partial^2 L(y_i, f(x_i))}{\partial f(x_i)^2}\Big]_{f(x)=\hat{f}^{(m-1)}(x)};\\
\hat{\phi}_m &= \arg\min_{\phi\in\Phi} \sum_{i=1}^{n} \frac{1}{2}\hat{h}_m(x_i)\Big[\Big(-\frac{\hat{g}_m(x_i}{\hat{h}_m(x_i)}\Big) - \phi(x_i)\Big]^2;\\
\hat{f}_m(x) &= \eta\hat{\phi}_m(x);\\
\hat{f}^(m)(x) &= \hat{f}^{m-1}(x) + \hat{f}_m(x);
\end{aligned}$$

3. end

Output: $\hat{f}(x) \equiv \hat{f}^{(M)}(x) = \sum_{m=0}^{M} \hat{f}_m(x)$

XGBoost is very well known due to computational power - “system runs more than ten times faster than existing popular solutions on a single machine and scales to billions of examples in distributed or memory-limited settings” [7]. Such scalability is reached through several optimizations [7]:

- Approximate Algorithm for split finding algorithm;
- Column block for parallel learning - data is stored in in-memory units and is sorted in it, thus sorting only needs to be done once before training;
- Weighted Quantile Sketch algorithm for approximate tree learning;
- Sparsity-aware algorithm - XGBoost visits only default direction (non-missing entries) in each node;
- Cache-aware access - 2^{16} examples per block is chose to prevent cache miss during split finding and ensure parallelization;
- Out-of-core computation - if main memory is full, data is divided into multiple blocks which are stored on the disk. The blocks are compressed by columns and decompressed on the fly by an independent thread while disk reading;
- Regularized Learning Objective. Most algorithms do not include regularization term into the objective function and have training loss function part only. The regularization term helps to smooth the final learnt weights. As a result it penalizes the complexity of the model and to avoid overfitting.

The above described features might be found in other algorithms individually, but the combination of the techniques is specific for XGBoost and ensures to provide not only an effective solution in resource aspect but also a winning method in many machine learning challenges.

Hyperparameters In order to maximize power of the XGBoost model, the hyperparameters are tuned specifically for a given dataset. Further only parameters which were finetuned (others were left with the default values) for XGBoost models in practical part are reviewed [23]. Chapter is written according to the official XGBoost package page guidelines.

Booster parameters:

- η (default = 0.3, alias: *learning rate*) - step size shrinkage used to prevent overfitting, values are $[0, 1]$;

- *max depth* (default = 6) - maximum depth of tree used to control overfitting. The higher the value, the higher possibility of overfitting there is. Deep trees also require a lot of computational power. Range is $[0, \infty]$;
- *colsample bytree* (default = 1) - percentage of features when constructing each tree. Subsampling is done once for every tree construction;
- γ (default = 0, alias: *min split loss*) - specifies a minimum loss reduction required in order to make a node split (the nodes are split only if it results in positive reduction in the loss function). The larger the gamma is, the more conservative the algorithm will be. Range is $[0, \infty]$;
- α (default = 0, alias: *reg alpha*) - L1 regularization term on weights (analogous to Lasso regression (Appendix)). Used in case of high dimensionality (to fasten the algorithm). The higher the value, the more conservative the model will be;
- λ (default = 1, alias: *reg lambda*) - L2 regularization term on weights (analogous to Ridge regression (Appendix)); Used for XGBoost regularization on leaf weight and is smoother than L1 regularization. The higher the value, the more conservative the model will be;
- *n estimators* - number of gradient boosted trees to build. Equivalent to number of boosting round;

Learning Task parameters:

- *objective* (default = reg:squarederror) - defines the loss function to be minimized;
- *eval metric* (default for regression = *rmse*, for classification = *error*, for ranking = *mean average precision*) - metric used for data validation;

3.2.2 XGBoost versus Random Forest

From Literature Review, readers were introduced to search for methods to CLTV prediction where Random Forest (RF) algorithm was used. In this thesis we propose using the XGBoost algorithm instead.

RF differs from XGBoost mostly due to technique of learning - even though it is also an ensemble learning method, the RF gives final prediction by averaging the predictions of individual trees (in case of regression task) or pick out the class selected by most trees of the forest (in case of classification task). As already mentioned before, XGBoost learns iterative by creating new models for prior models residual explanation. Due to this XGBoost tends to show higher accuracy results in practice.

Moreover, in case of having similar samples per tree in RF, we are facing the overfitting problem. As XGBoost has a so-called "similarity score" and prunes the tree before modeling, this challenge is overcome.

Furthermore, due to boosting technique XGBoost is a good option when working with unbalanced dataset while there is a high chance that RF would make predictions with some randomness. When predicting customers' future spend, the businesses are mostly interested in high-value customers as it is crucial to recognize them and give special treatment in case their possibility to churn is growing. As seen from Literature Review (chapter X), in non-contractual market significant part of customers do not even have a repeatable purchase and usually small fraction of customers makes the bigger part of revenue (Pareto principle states that relation is 20/80 [17]) hence it only makes sense to implement XGBoost tree model for GMV prediction due to it advantage when predicting anomalies.

3.2.3 Our approach

Further in this thesis, XGBoost algorithm is applied on selected real life dataset. Two XGBoost models will be presented:

1. XGBoost for regression - model predicts every customer's GMV for upcoming 360 days.
2. 2-stage approach
   - 2.1 XGBoost for classification - model predicts if customer will churn or be active (make a purchase) in upcoming 360 days. When prediction is positive (customer will churn), the GMV (360 days) for that customer is set as zero, while if prediction is that customer won't churn, the 2.2. step is applied
   - 2.2 XGBoost for regression - model predicts customers' GMV for upcoming 360 days (only customers who were predicted to stay alive in 2.1. step are modeled)

Expectation is that XGBoost models will outperform baseline model and that 2-stage approach will outperform one-stage XGBoost regression.

The hyperparameters were optimized by using the random search (RandomizedSearchCV) algorithm. In the Table 1 you will find optimized parameters for all three models (1, 2.1, 2.2). Other parameters were left as default. The explanation of parameters could be found in subparagraph Hyperparameters.

3.3 Deep Neural Networks

Finally, the last methodology to be adapted to GMV prediction is Deep Neural Networks (DNN), also called Feedforward Neural Networks (FFNN) as starting with input layer data flows one-way (only forward) between the layers and to the output layer and do not form a cycle. As

Parameter	1-stage regression	2-stage classification	2-stage regression
objective	reg:squarederror	binary:logistic	reg:squarederror
eval_metric	RMSE	error	RMSE
n_estimator	50	80	50
gamma	1	1	0
max_depth	5	6	10
learning_rate	0.5	0.2	0.1
colsample_bytree	0.8	0.6	0.8
lambda	10	100	10
alpha	1	1	1

Table 1: XGBoost parameters

far as the search was done, DNN is not a highly used algorithm for customer's GMV/CLTV prediction - could be due to high computational resources requirement or that DNN models are described as black box in behalf of interpretability. Either way, in recent years, artificial neural networks are becoming more common in many businesses and as it shows encouraging results in various fields it seems only fair to try to implement it in order to forecast buyers' GMV for upcoming 360 days.

3.3.1 Neural Network algorithm

In simple words, a neural network is a graph of connected perceptrons. Let's say we have a binary classification task. If the selected model is a single-perceptron neural network - it is becoming a simple logistic regression model. Thus we are interested in multilayer perceptron (MLP) neural networks. MLP consists of at least three layers - an input layer, a hidden layer and an output layer. The layers contain a certain number of nodes which are connected to the following layer nodes with specific weight which shows the importance of a particular node. Perceptron works as follows: inputs are multiplied by their weights, the multiplied values are added with the bias and a selected activation function is applied. Bias is needed to shift calculation up or down, while activation function maps the output to the required values (for example between zero and one). Same as in Gradient Boosted Decision Trees, the aim of the algorithm is to minimize the loss by recalculating weight for wrong predictions by using gradient descent.

3.3.2 Parameters of the model

There are a lot of moving parts in order to build a good enough neural network for a specific dataset. Further the parameters of neural network will be introduced:

- *Number of layers* - the simpler the problem, the smaller amount of layers is needed. Besides hidden layers which working scheme was discussed before there are regularization layers as well:
  - *Batch normalization* - normalizes the values passed to it;
  - *Dropout layer* - randomly drops a certain number of neurons in a layer. The share of neurons to drop is set by the network creator;
- *Number of neurons in each hidden layer* - the number could differ per different layers. Number of neurons grows together with the complexity of the problem;
- *An activation function* - different activation functions could be applied to different layers. It defines the output of the node meaning that in case of binary classification problem for output layer *sigmoid* (17) activation function is used, thus output would be a probability between zero and one. Other highly popular activation functions which were tried in order to be an empower model are: *ReLu* (rectified linear unit) (18), *Leaky ReLu* (19), *Tanh* (Hyperbolic tangent) (20);

$$\sigma(x) = \frac{1}{1 - e^{-x}} \tag{17}$$

$$\begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases} \tag{18}$$

$$\begin{cases} 0.01x & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases} \tag{19}$$

$$tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}. \tag{20}$$

- *Optimizer* - method used to change the attributes of a neural network in order to minimize the loss. For example machine learning models use Gradient Descent as the optimization algorithm. For neural networks the most popular and considered as the best optimization algorithm is Adaptive Moment Estimation (Adam) algorithm 21

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t \\ v_t &= \beta_2 vt - 1 + (1-\beta_2) g_t^2. \end{aligned} \tag{21}$$

Here $\beta_1, \beta_2 \in [0,1)$ control the exponential decay of the moving averages of past and past squared gradients m_t, v_t. m_t, v_t are the estimates of mean and uncentered variance of the gradients;

  - *Learning rate* - is the parameter given for the optimizer which determines the size of the steps for the movement towards minimum of loss function. High learning rate makes models faster but the probability to find the optimal weights is smaller than with lower learning rate;

- *Batch size* - the number of training data subsample for the input. In the case of a huge dataset, this parameter saves computational resources by dividing learning into smaller batches. The consequence of small batches is that the variance of validation has higher variance;

- *Epochs* - number of times the dataset is passed through a neural network. It is important to select enough epochs so the model could learn enough and avoid underfitting as well as too many epochs leads to overfitting thus the model does not give good results on the unseen data;

3.3.3 Multi Output

As a contribution to the GMV prediction field, a multi output neural network was built as, as far as it is known, such approach was never used on CLTV prediction problems. The idea to adapt multi output algorithms arose from the already used in a field 2-stage approach with machine learning methods where the first classification model is built to forecast whether a specific customer will churn during the target period (360 days in our case). The second stage is a regression to forecast GMV but only users who were predicted to remain alive in the first stage are used in this model. Multi output approach combines the two - multiple outputs are generated in one neural network with the same single input [24]. The expectation is that while learning about both variables - churn and GMV, the model would combine the learnings and improve GMV prediction with that.

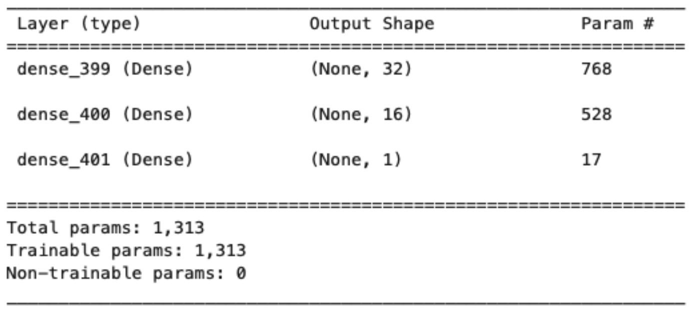
3.3.4 Our approach

Further in this thesis, FFNN algorithm is applied on selected real life dataset. Three models will be presented:

1. FFNN as a regression task - model predicts every customer's GMV for upcoming 360 days;
2. 2-stage approach:
   - 2.1 FFNN for classification - model predicts if customer will churn or be active (make a purchase) in upcoming 360 days. When prediction is positive (customer will churn), the GMV (360 days) for that customer is set as zero, while if prediction is that customer won't churn, the 2.2. step is applied;
   - 2.2 FFNN for regression - model predicts customers' GMV for upcoming 360 days (only customers who were predicted to stay alive in 2.1. step are modeled);
3. Multi Output - one input neural network delivers two outputs - whether the customer will stay alive in the selected future period and what the predicted GMV for that future period is. Model is a feedforward network as well;

As NN is considered to be highly advanced machine learning method, we expect that FFNN will outperform baseline and XGBoost approaches. Also as churn prediction should help to forecast GMV more accurately, we expect that both 2-stage approach and multi output will outperform one-stage FFNN regression.

Series of experimentation was done and parameters were optimized with the random search (RandomizedSearchCV) algorithm. During the experimentation the batch normalization layers were also included but as there was no improvement on selected evaluation metric it was removed from the final construction of the models. Final models constructions are shown in Figures 1, 2, 3, 4, 5, parameters in Table 2.



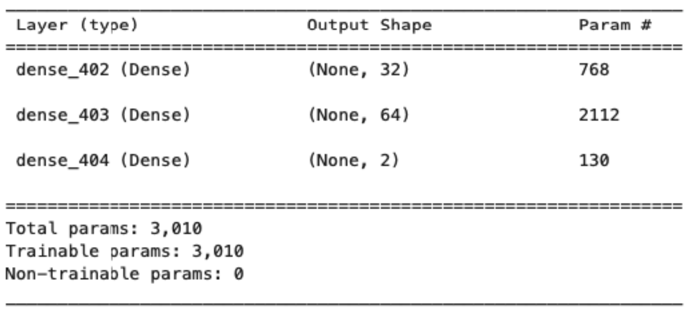
```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_399 (Dense)           (None, 32)                768

 dense_400 (Dense)           (None, 16)                528

 dense_401 (Dense)           (None, 1)                 17

=================================================================
Total params: 1,313
Trainable params: 1,313
Non-trainable params: 0
_________________________________________________________________
```

Figure 1: FFNN regression model (1.) construction



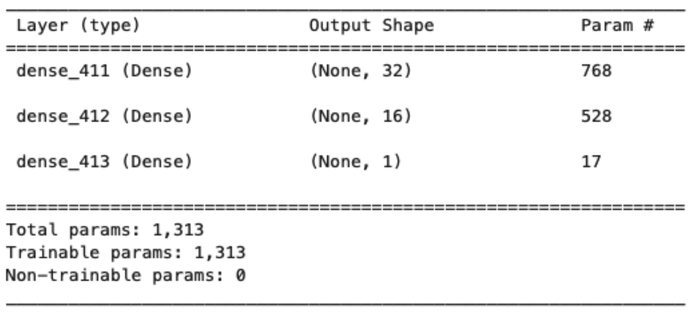
```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_402 (Dense)           (None, 32)                768

 dense_403 (Dense)           (None, 64)                2112

 dense_404 (Dense)           (None, 2)                 130

=================================================================
Total params: 3,010
Trainable params: 3,010
Non-trainable params: 0
_________________________________________________________________
```

Figure 2: FFNN classification model (2.1.) construction



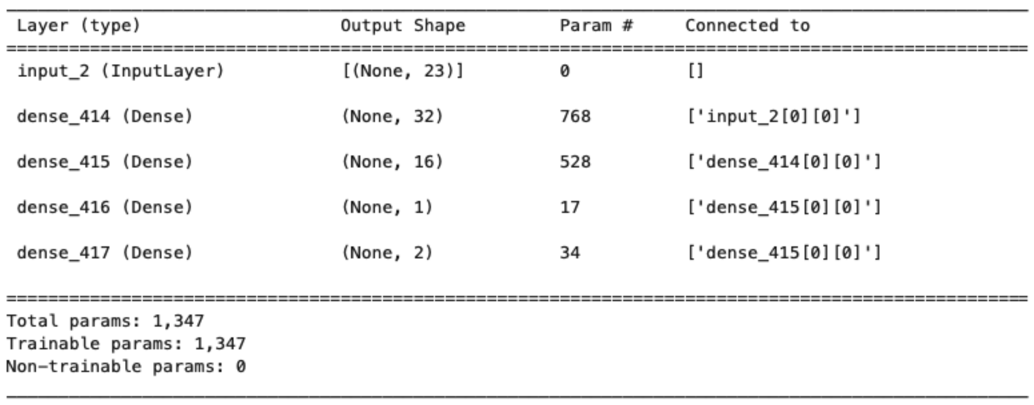
```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_411 (Dense)           (None, 32)                768

 dense_412 (Dense)           (None, 16)                528

 dense_413 (Dense)           (None, 1)                 17

=================================================================
Total params: 1,313
Trainable params: 1,313
Non-trainable params: 0
_________________________________________________________________
```

Figure 3: FFNN regression model (2.2.) construction



```
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_2 (InputLayer)           [(None, 23)]         0           []

 dense_414 (Dense)              (None, 32)           768         ['input_2[0][0]']

 dense_415 (Dense)              (None, 16)           528         ['dense_414[0][0]']

 dense_416 (Dense)              (None, 1)            17          ['dense_415[0][0]']

 dense_417 (Dense)              (None, 2)            34          ['dense_415[0][0]']

==================================================================================================
Total params: 1,347
Trainable params: 1,347
Non-trainable params: 0
__________________________________________________________________________________________________
```

Figure 4: FFNN multi output (3.) construction (1)

Parameter	**1. FFNN**	**2.1. FFNN classification**	**2.2 FFNN regression**	**3. Multi Output**
epochs	100	100	80	150
loss	mse	sparse_categorical_crossentropy	mse	mse and sparse_categorical_crossentropy
optimizer	Adam	Adam	Adam	Adam
learning rate	0.01	0.001	0.01	0.01
batch size	128	256	128	64
activation function	*ReLu for hidden layers*	*Tanh for hidden layers, Sigmoid for output layer*	*ReLu for hidden layers*	*ReLu for hidden layers, Sigmoid for output layer for classification task*

Table 2: FFNN parameters

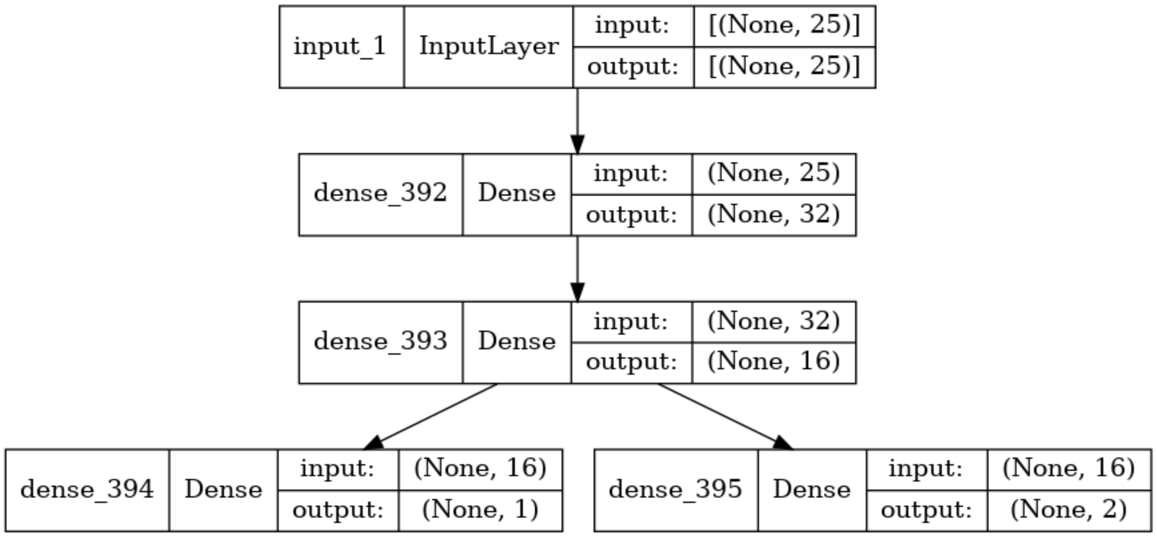


Figure 5: FFNN multi output (3.) construction (2)

3.4 Evaluation metrics - RMSE and MAE

In order to evaluate the models performance on selected dataset and compare them, two metrics were chosen:

- root-mean-square-error (RMSE) - the residuals are squared, then averaged over the sample and finally the square root of it is taken;

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

- Mean Absolute Error (MAE) gives an accuracy for regression results, it is an average of the absolute errors;

$$MAE = \frac{\sum_{i=1}^{n} |\hat{y}_i - y_i|}{n}$$

The smaller MAE and RMSE values are, the more accurate the model is. For MAE all residuals are weighted equally as score is linear, meanwhile RMSE gives bigger weight to the large errors meaning that it is not robust to the outliers. When having these two metrics together we can diagnose the variation in the errors - if RMSE is to MAE all errors are of the same magnitude while the bigger the difference is (RMSE is always higher than MAE), the greater the variance in the individual residuals are.

4 Empirical application

4.1 Dataset

For practical experiments, a dataset from a two-sided C2C plattform is taken. An user of the platform could be both - a seller and a buyer thus it is important to clarify that further in this work we concentrate on buyer activity and purchases meaning that gross merchandise value (GMV) in our case is an amount of EUR bought by the user through the platform.

The algorithm to create a dataset (Figure 6):

1. Filter only those users who had their first purchase from 2018-01-05 to 2018-12-31
2. The date of first purchase is marked $feature_{start}$.
3. For every user random integer r is generated, where $r \in (0, 360]$
4. Then feature period end is calculated: $feature_{end} = feature_{end} + r$
5. The aim is to predict GMV in target period, where $target_{start} = feature_{end} + 1$ and $target_{end} = target_{start} + 360$.
6. All independent variables were aggregated from customer's activity during feature period - $[feature_{start}; feature_{end}]$
7. The dependent variable is GMV generated in target period - $[target_{start}; target_{end}]$

The reasoning behind making the dataset where we have customer's with various sizes of activity information (from 1 to 360 days) is to be able to adapt the best working model to the business right away. As the feature period is different for every user - could be from 1 to 360 days of lifetime - we expect higher accuracy for users with more known information. This hypothesis will be evaluated in the Residuals Analysis paragraph.

Dataset was splitted to three parts: train, validation and test with such ratio: 70/20/10.

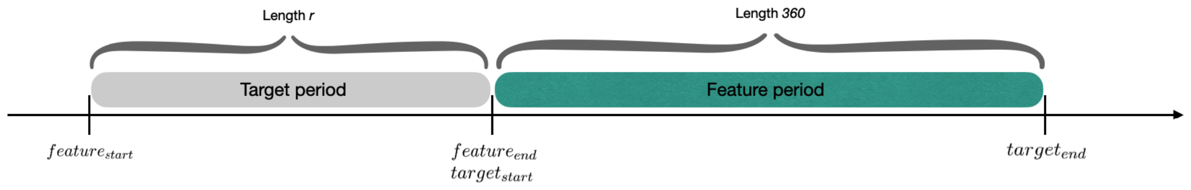


Figure 6: Dataset construction

Parameter	**Value**
Count	1,220,050
Mean	161.4882
Standard deviation	260.5789
Min	0.000000
25%	0.000000
50%	55.2500
75%	203.500
Max	1795.00

Table 3: GMV 360 description

4.2 Dependent variable

Dependent variable of the model is gross merchandise value spent by a buyer in 360 days (or feature period)(GMV 360). Users whose GMV 360 was higher than 99th percentile (1795 EUR) of the sample were removed from the dataset.

The dependent variable is left skewed as 29.85% of users do not have any purchases made during the target period. Description of the dependent variable is given in Table 3, distribution - in Figure 7.

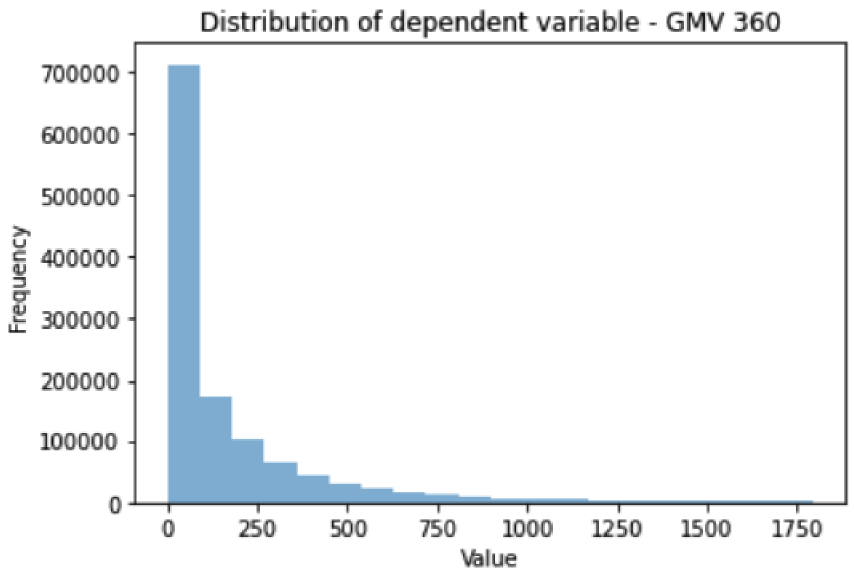


Figure 7: Distribution of dependent variable

4.3 Independent variables

Every independent variable with the explanation is given in the Table 4. The columns are made from aggregated information from the feature period $[feature_{start}; feature_{end}]$. The categorical variables were one-hot encoded while for XGBoost and FFNN models numerical variables were standardized. In order to avoid multicollinearity problem, Variance Inflation Factor (VIF) measure was calculated and with a threshold of 10 it was made sure that independent numerical variables won't be correlated in between.

4.3.1 Correlation with dependent variable

Correlation between numerical independent and dependent variables is given in a Table 5 (ordered by importance).

Variable 'Days since first purchase' is very low correlated with a GMV 360 (-0.014) but this variable indicated how many days of customer's activity information there is hence it is included as there is need for model to understand that customer's known history differs and when forecasting the target period it should be taken into the account. Without knowing how much the customer's history is known, the model would assume that a customer with let's say 5 purchases would have higher future GMV than a customer with 2 purchases. Meanwhile it could be that a customer with 2 purchases was observed for a month while the first customer was observed for ten months.

Also 'Average Order GMV' has low correlation with the dependent variable (-0.010). According to the theory of CLTV predictions and importance of average order value (monetary from RFM data set) to it, the decision was made to leave this variable.

Regarding categorical variables (differences mentioned further are all statistically significant with 95% confidence):

- **Gender**: GMV (360 days) generated by females is on average 8 EUR higher (4.9% difference) than by mens;

- **Language**: Customers with FR language generate 3x more GMV (on average 171 EUR versus 53 EUR). French speakers take 91% of all customers;

- **Registration** type: Email identified users spend on average 12 EUR more than customers identified with a Google account and 16 EUR more than customers with Facebook identification;

- **First visit platforms** also have significant differences on GMV. On average, customers who first visit the platform through mobile web generate 192 EUR of GMV in a year, while web(desktop or other) 171 EUR, android 155 EUR, iphone 149 EUR;

information type	**Column name**	**Explanation**
Demographic and registration information	Gender	Gender of the customer: M / F / other (categorical variable)
	Language	Language of the customer: FR / other (categorical variable)
	Age	Age of user by years
	Registration type	Type to identify self and register to the e-commerce page: Unknown / Email / Google / Facebook (categorical variable)
	First visit platform	Type of platform customer first visited: ndroid / iphone / other / mobile web / web (desktop or other) (categorical variable)
Transactional information	Days since first purchase	Number of days between $[feature_{start}; feature_{end}]$
	Purchase count	Number of purchases made by the customer
	GMV generated	GMV generated by the customer
	Days first to last purchase	Days between first and last purchase
	Average order GMV	Average value of the order
Activity information	Personalized size	Categorical variable (yes/no). In the chosen e-commerce there is a setting to add your size preferences.
	Visited FAQ	Categorical variable (yes/no). Shows if customer visited FAQ page
	Is seller	Categorical variable (yes/no). Selected marketplace is C2C business thus the buyer could also be a seller.
	Session Frequency Ratio	Number of days when user visited e-commerce page divided by total days of feature period ('days since first purchase')
	Item Views Ratio	Number of viewed items in e-commerce page per feature period divided by total days of feature period ('days since first purchase')
	Item Wants Ratio	Number of showed interest on buying items in e-commerce page per feature period divided by total days of feature period ('days since first purchase')
	Count Ticket Ratio	Number of customer support tickets customer was involved in divided by total days of feature period ('days since first purchase')

Table 4: Independent variables

Independent variable	**Correlation**
Days since first purchase	-0.014
Average Order GMV	-0.010
Count Ticket Ratio	0.055
Item Wants Ratio	0.297
Days first to last purchase	0.298
Item views Ratio	0.338
Session Frequency Ratio	0.372
Purchase count	0.464
GMV generated	0.489

Table 5: Correlation between the independent variable and GMV 360

- Customers who have **personalized size**, generate on average 26 EUR more than users without personalization;
- Customers who have **visited the FAQ** page (81% of the sample), generate 2x more GMV;
- Buyers who are **sellers** in the platform too, buy for about 60 EUR more;

4.4 Results

The adapted models for the real life dataset were reviewed in the methodology chapter. Here the names of them as a reminder together with a numerical notation which would be used later in this chapter:

1. BG/NBD (Gamma-Gamma)
2. XGBoost
   - 2.1 One-stage approach
   - 2.2 Two-stage approach
3. Feed forward neural network (FFNN):
   - 3.1 One-stage approach
   - 3.2 Two-stage approach
   - 3.3 Multi output

Model	**RMSE**	**MAE**
1.BG/NBD (Gamma-Gamma)	315.99	150.88
2.1. XGBoost - one stage approach	198.88	119.88
2.2. XGBoost - two stage approach	200.81	124.14
3.1. FFNN - one stage approach	197.54	118.39
3.2. FFNN - two stage approach	202.23	121.78
3.3. FFNN - multi output	198.15	120.27

Table 6: Experiment results

		Predicted label	
		Not churn	Churn
True label	Not churn	77132 (90.38%)	8202 (9.62%)
	Churn	15829 (43.81%)	20301 (56.19%)

Table 7: Confusion Matrix of (2.1.) model

Models were evaluated in between with MAE and RMSE metrics. Results are given in the Table 6.

Both XGBoost (2.1 and 2.2) and FFNN (3.1, 3.2, 3.3) approaches have improved the baseline (1.) model as their RMSE is lower by 35%, while MAE is lower by 25%. Nevertheless the average model errors are still huge - the best performing model - FFNN (3.1.) have average absolute error (MAE) of 118 EUR (while average customers' GMV 360 is 161 EUR) - thus the model is not ready to be used in real life application yet.

Two-stage approach (2.2 and 3.2) did not improve the predictions as expected - models were less accurate than the one-stage regression approach in both XGBoost and FFNN classes. Even though the difference is small. The reasons behind it might be not a good enough classification model (to predict customers' churn). The confusion matrix of the XGBoost classification model is given in the Table 7 where we can see that from truly churned users the model identifies only 56.19% of them. Hence final accuracy of the model is 80.22% while zero rate classifier (when model always classifies the larger (not churned) class) would be 70.15% thus even though our model works better, the improvement is not huge. During the experimentation, an additional model was created where churn prediction was included as an extra variable for the regression problem but importance of that feature was low showing that the churn prediction model was not developed well enough. It seems that churn prediction should be considered as a separate topic and it needs more knowledge and experiments.

Though the multi output FFNN model (3.3) has lower RMSE than two-stage approach (3.2), it did not outperform simple FFNN (3.1) thus the hypothesis that simultaneous (in the same neural network) learning for two output variables would give higher accuracy of GMV 360 prediction is rejected.

In a Figure 8, SHAP value (see Appendix) of features of FFNN (3.1) model is shown. The features are given in order of importance from top to bottom while the location on the X-axis shows whether the effect of the feature is positive or negative to the dependent variable. The color indicates whether that feature is high (red) or low (blue) for that observation. GMV 360 spend during the feature period is the most important variable in the model, the higher it is, the higher the target period GMV is. Same interpretation applies to the user's age, the item view, and item want ratios. Days since first purchase - variable gives model an information on how many days of information we have about customer hence the business logic says (and correlations given in Table 5 confirms) that it should not affect GMV 360 spend thus we see that it is second most important feature and the effect on target period GMV is negative.

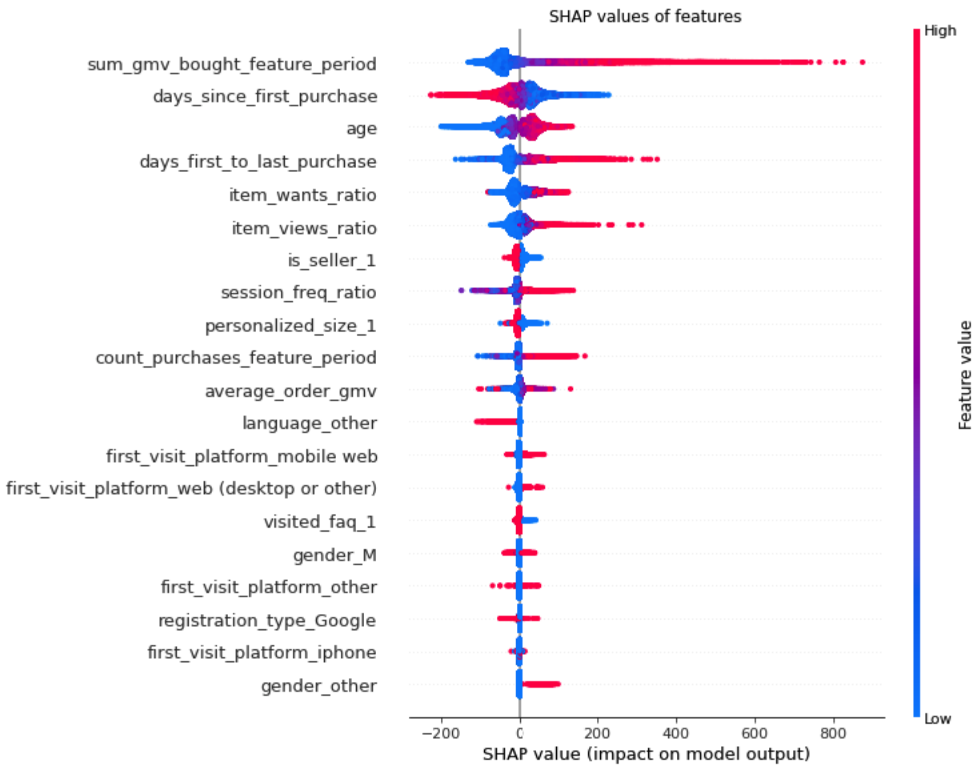


Figure 8: Feature importance by SHAP value

4.4.1 Residual analysis

In order to improve the model in the future, the residuals of FFNN (3.1.) model were analyzed. Due to the algorithm with which the dataset was created, we expect higher accuracy for users with longer feature period (where 'days since first purchase' is higher). In the Figure 9, average real and predicted values are shown per different length of periods (12 months). The red line shows average absolute error in the particular segment. As expected, error drops within more information about the user behavior is collected. This information implies that for the next stages there could be different models for different length of feature period tried out.

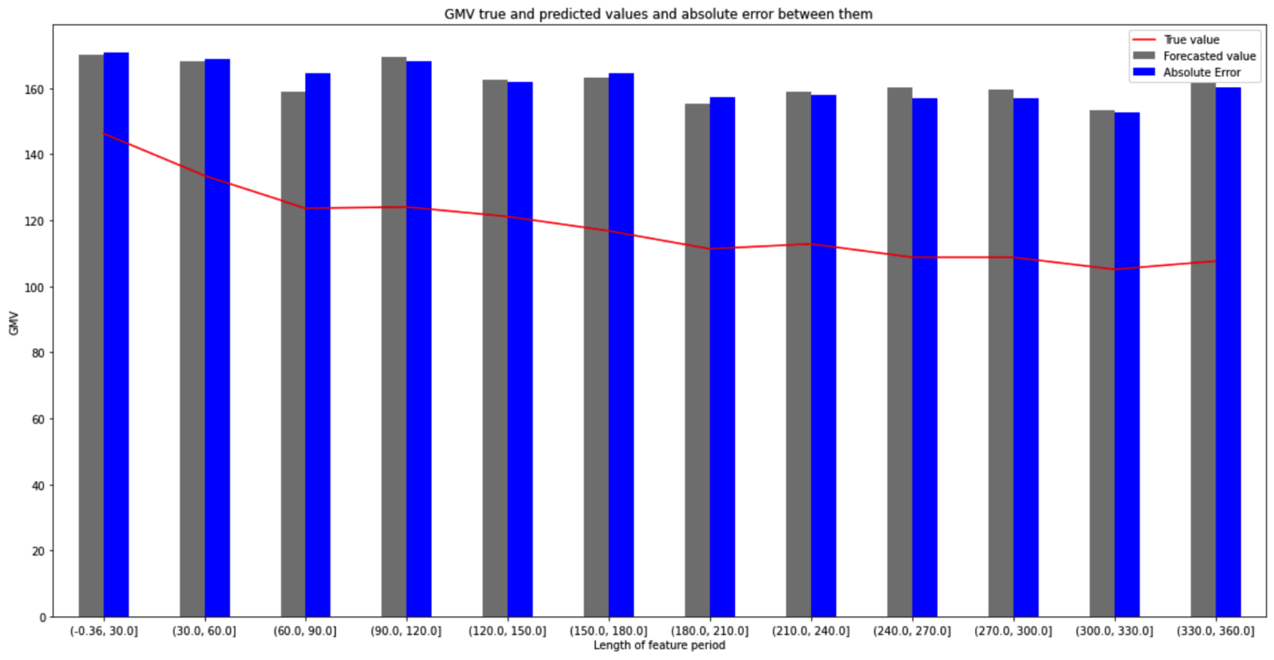


Figure 9: Predicted and true GMV 360 values per length of feature period

5 Conclusions

In this thesis, we adapted models to predict Gross Merchandise Value (GMV) for a buyer in e-commerce. As novelty we introduced XGBoost (regression and two-stage approach), FFNN (regression, two-stage approach, multi output) models which, as far as we know, were not used in a field before and applied these methods to the real life dataset. In conclusions:

- As expected, introduced XGBoost and FFNN method models outperformed baseline - BG/NBD (with Gamma-Gamma submodel) model;
- For both XGBoost and FFNN, one-stage regression model showed slightly higher accuracy than 2-stage or multi output approaches. This was not expected, in this way we identify that churn prediction should be addressed as a separate topic and only when higher accuracy is reached it should be adapted as a pre-model for GMV prediction;
- As expected, the highest importance variable is the GMV spend during the feature period. Besides it, model values activity data - such as impressions, showed interest about items;
- From demographic data, customer's age showed high importance as the older the buyer, the higher GMV is spend;
- The more days of information model has about the user, the better it works which identifies that different models should be adapted for very new or much experienced customers;

For further research, we advise to focus more in order to identify churned users before aiming to predict GMV. Also, seems that higher accuracy might be reached with having separate models for separate customer segments according to their activity and potential value. Embedding created on buyers purchases or even item impressions level as additional feature showed great promise for other researchers [5] [3] thus it could be a great added value for further experimentation. Finally, due to left-skewness of the GMV, the selected loss function for FFNN - MSE is not very appropriate hence loss function adjustment might improve the predictions.

References

[1] Abe, M. *“Counting your customers” one by one: A hierarchical Bayes extension to the Pareto/ NBD model.* Marketing Science, 2009, pp. 541-553.

[2] Batislam, Emine Persentili, et al. *Empirical validation and comparison of models for customer base analysis* International Journal of Research in Marketing, 2007, pp. 201-209.

[3] Bauer, Josef, and Dietmar Jannach. *Improved Customer Lifetime Value Prediction With Sequence-To-Sequence Learning and Feature-Based Models*, 2021.

[4] Castéran, Herbert, et al. *Modeling Customer Lifetime Value, Retention, and Churn* Handbook of Market Research, 2017, pp. 1-33.

[5] Chamberlain, Benjamin, et al. *Customer Lifetime Value Prediction Using Embeddings* 2017.

[6] Chavla, Nitesh, et al. *SMOTE: Synthetic Minority Over-sampling Technique* Journal of Artificial Intelligence Research, 2018, pp. 321-357

[7] Chen, Tianqi et al. *XGBoost: A Scalable Tree Boosting System*, 2016

[8] Colombo, RIchard and Weina Jiang *A stochastic RFM model* Journal of Interactive Marketing, 1999, pp. 2-12

[9] Ehrenberg, A.S.C.*The Pattern of Consumer Purchases* Journal of the Royal Statistical Society, 1959, pp. 26-41

[10] Fader, Peter S., et al.. *CLV: More than meets the eye* Marketing Research, 2006.

[11] Fader, Peter S., et al. *“Counting Your Customers” the Easy Way: An Alternative to the Pareto/NBD Model* Marketing Science, 2005, pp. 275-284

[12] Fader, Peter S., et al.*RFM and CLV: Using iso-value curves for customer base analysis* Journal of Marketing Research, 2005, pp. 415-430

[13] Fader, Peter S., and Bruce G.S. Hardie.*The Gamma-Gamma model of monetary value*, 2013

[14] Jasek, Pavel, et al. *Comparative Analysis of Selected Probabilistic Customer Lifetime Value Models in Online Shopping* Journal of Business Economics and Management, 2019, pp. 398-423

[15] Ma, S. H., J. L. Liu *The MCMC approach for solving the Pareto/NBD model and possible extensions. Proceedings of the 3rd International Conference on Natural Computation*, 2007, pp. 505-512

[16] Nielsen Didrik *Tree Boosting With XGBoost – Why Does XGBoost Win "Every" Machine Learning Competition*, 2017

[17] Pareto, Vilfredo, *Cours d'Économie Politique: Nouvelle édition* par G.-H. Bousquet et G. Busino, Librairie Droz, Geneva, 1964

[18] Platzer, M., T. Reutterer. *Ticking away the moments: Timing regularity helps to better predict customer activity*, Marketing Science, 2016, pp. 779-799

[19] Schmittlein, David C., et al.*Counting Your Customers: Who-Are They and What Will They Do Next?*, Management Science, 1987, pp. 1-24

[20] Sifa, Rafet, et al. *Customer Lifetime Value Prediction in Non-Contractual Freemium Settings: Chasing High-Value Users Using Deep Neural Networks and SMOTE*, 2018

[21] Vanderveld, Ali, et al. *An Engagement- Based Customer Lifetime Value System for E-commerce*, 2016

[22] Wang, Xiaojing, et al. *A Deep Probabilistic Model for Customer Lifetime Value Prediction*, 2019

[23] Official XGBoost documentation https://xgboost.readthedocs.io/en/latest/index.html

[24] Xu, et al. *A Survey on Multi-output Learning*, 2019

6 Appendix

Negative Binomial Distribution

$$f(x; r, P) =^{x-1} C_{r-1} \cdot P^r \cdot (1-P)^{x-r},$$

where x - total number of trials, r - number of success trials, P - probability of success on each trial, ${}^nC_r$ - combination of n items taken r at a time.

Pareto distribution

$$F(x) = 1 - (k/x)^\alpha,$$

where x - random variable, k - lower bound of the data, α - shape parameter.

Beta-Geometric distribution

$$P(X = k) = \frac{u\Pi_{i=1}^{k-1}(1 - u + (i-1)\theta)}{\Pi_{i=1}^{k}(1 + u\theta},$$

where $u = \alpha/(\alpha + \beta)$, $tetha = 1/(\alpha + \beta)$ and α, β - unknown non-negative parameter.

Beta-Geometric distribution

$$P(X = k) = \frac{u\Pi_{i=1}^{k-1}(1 - u + (i-1)\theta)}{\Pi_{i=1}^{k}(1 + u\theta},$$

where $u = \alpha/(\alpha + \beta)$, $tetha = 1/(\alpha + \beta)$ and α, β - unknown non-negative parameter.

Gamma function

$$\Gamma(\alpha) = \int_0^\infty x^{\alpha-1} e^{-x} dx,$$

where α is any positive real number.

Gaussian hypergeometric function

$${}_2F_1(a, b; c, z) = \frac{1}{B(b, c-b)} \int_0^1 t^{b-1}(1-t)^{c-b-1}(1-zt)^{-a} dt, c > b.$$

Bayes theorem

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}.$$

Lasso regression

$$\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{M}\Big(y_i - \sum_{j=0}^{p} w_j \times x_{ij}\Big)^2 + \lambda \sum_{j=0}^{p} |w_j|.$$

Ridge regression

$$\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{M}\Big(y_i - \sum_{j=0}^{p} w_j \times x_{ij}\Big)^2 + \lambda \sum_{j=0}^{p} w_j^2.$$

SHAP value

$$SHAP_{feature}(x) = \sum_{set: feature \in set} [|set| \times \binom{F}{|set|}]^{-1}[Predict_{set}(x) - Predict_{set/feature}(x)].$$