National Research University Higher School of Economics Faculty of Computer Science
Programme 'Master of Data Science'

MASTER'S THESIS

Customer churn prediction

Student **Sharipov Ainur Aivarovich**

Supervisor **Chankin Andrey Andreevich**

Moscow, 2024

ABSRACT

In the competitive banking sector, retaining customers is vital due to the high costs of acquiring new ones. This thesis develops a machine learning algorithm to predict customer churn using six months of transaction data. The project is part of the Data Fusion 2024 competition by VTB.

The study employs survival analysis and various machine learning methods to address challenges such as data censoring and feature engineering. Key processes include data preprocessing, feature generation, and model optimization. Developing relevant features, particularly tracking changes in transaction flow rates over time, is vital for boosting methods to be effective. The created model, comprised of XGBoost and CatBoost ensemble, is a superior open-source solution to the competition.

The research demonstrates that accurate churn prediction is achievable through careful data handling and advanced modeling techniques. This work offers valuable insights for businesses aiming to improve customer retention strategies.

The paper contains 51 pages, 5 chapters, 17 figures, 16 tables and 42 sources.

Keywords: customer churn, machine learning, survival analysis, data preprocessing, feature engineering, XGBoost, CatBoost, ensemble.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

1.1. Background

In today's increasingly competitive world, understanding customer behavior and customer retention are becoming key factors for a successful business. One important tool in achieving this goal is analyzing customer data and predicting user churn.

Customer data analytics is the process of extracting, cleaning, exploring and visualizing data about a company's customers in order to identify trends and patterns in their behavior. This analysis allows companies to understand what makes their customers happy, what annoys them and, most importantly, to predict the likelihood that a customer may leave for a competitor, which is known as user churn.

Churn prediction is the process of developing machine learning models that predict how many and which customers are likely to leave a company in a given period of time. Warning of potential churn gives companies the opportunity to take steps to retain customers, which can significantly reduce losses and increase overall profits.

Telecommunication operators, banks, insurance companies and other organizations are engaged in predicting and managing customer churn. In a highly competitive environment, predicting customer churn in order to retain customers is becoming one of the most important areas in modern business.

1.2. Problem statement

For the banking sector, predicting customer churn [1] is a serious problem and has a huge impact on banks' profits. The loss of customers equals the loss of the bank's future income plus the loss of the initial investment made to acquire these customers. In addition, the loss of customers leads not only to lost opportunities due to lower sales, but also to an increased need to attract new customers, which is five to six times more expensive than retaining customers [2].

Thus, a customer retention scheme can target high-risk customers who want to abandon their services and switch to another competitor. In order to minimize the costs of the banking sector on customer retention marketing scheme [1], accurate and preliminary identification of such customers is of great importance.

1.3. Thesis objective

The goal of this paper is to presents the best possible solution for Data Fusion 2024 competition hosted by VTB at ods.ai [3].

It is necessary to solve the problem of predicting the churn rate of bank customers. Namely, using transaction data for 6 months, we need to build an algorithm that predicts the probability of customer churn in the next 6 months. The peculiarity of the problem is that as part of the training data for training, the participants are given not only the label corresponding to the fact that the client will "churn", but also the time until his last transaction.

Predicting customer churn is an extremely common task found in many companies in many different industries. Despite its widespread use, churn tasks have a large number of pitfalls and peculiarities that are often talked about but almost never shown in practice. On the other hand, there are many useful insights, analytical practices, and entire research areas in churn tasks that few people outside of practicing expert teams know about.

In this competition, participants can approach the problem in a large number of ways. In particular, the availability of information such as time to the last transaction within the training data will allow participants to take advantage of Time-to-Event (Survival) approaches in machine learning.

CHAPTER 2: LITERATURE REVIEW

2.1. Introduction

The topic of customer churn has garnered extensive research interest, leading to a substantial body of literature. There are numerous academic papers and industry studies that focus on various aspects of churn prediction, ranging from methodological advancements to practical applications.

Churn prediction and survival analysis both deal with time-to-event data, focusing on customer departure as the event of interest. Techniques from survival analysis, such as the Kaplan-Meier estimator [4] and Cox proportional hazards model [5], are commonly used to predict and understand churn behavior. Additionally, survival analysis concepts like censoring and hazard functions are directly applicable to analyzing customer retention and the risk of churn over time [6].

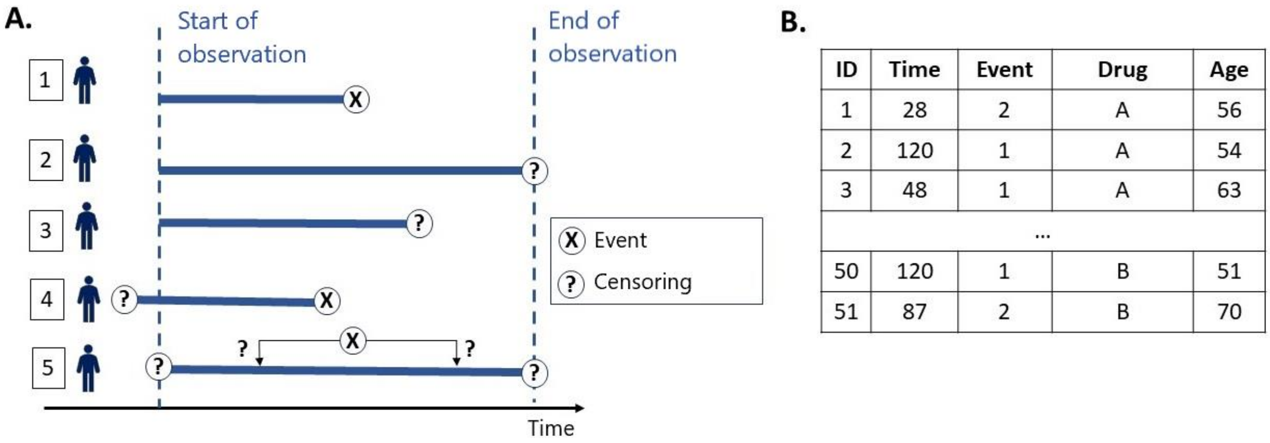
2.2. Survival analysis

Survival analysis is a branch of statistics that deals with analyzing the time until an event of interest occurs, often referred to as “time-to-event” analysis. This could involve studying the time until death, client's withdrawal from the company's services, time until failure of a mechanical system, time until relapse of a disease, and many other scenarios.

It would seem that we are considering a continuous positive measure - time, and, accordingly, we can use standard regression methods to analyze such values. But survival analysis data has one feature, and it is called “censoring” [7]. In reality, we do not always know the exact event times for all subjects in the study group. This could be a consequence of either the limited observation period or the absence of evidence brought on by other irrelevant events.

2.2.1. Data censoring

Censoring in survival analysis refers to the situation where the event of interest (such as death, failure, or churn) has not been observed for some subjects within the study period. This results in incomplete data for these subjects, and understanding the types and implications of censoring is crucial for accurate survival analysis. Below are the key types and concepts associated with censoring.



ID	Time	Event	Drug	Age
1	28	2	A	56
2	120	1	A	54
3	48	1	A	63
...				
50	120	1	B	51
51	87	2	B	70

Figure 1. A. Types of censoring of survival data; B. Example of a dataset for survival analysis

Types of censoring

1. *Right-censoring (subjects 2 and 3 in Figure 1 [8])*

   – Definition: This occurs when the observation period ends before the event happens, or the subject leaves the study for reasons unrelated to the event.

   – Example: A clinical trial ends before a patient dies, or a customer cancels a subscription before churning.

2. *Left-censoring (subject 4 in Figure 1 [8])*

   – Definition: This occurs when the event has already happened before the subject is included in the study, but the exact time of the event is unknown.

   – Example: A study on the onset of a disease where some participants have already contracted the disease before the study begins.

3. *Interval-censoring (subject 5 in Figure 1 [8])*

   - Definition: This happens when the exact time of the event is not known, but it is known to occur within a specific time interval.
   - Example: A regular health check-up that only identifies the occurrence of a disease within the intervals between check-ups.

Implications of censoring

1. *Incomplete data*

Censoring results in incomplete data for some subjects, which complicates the analysis [10]. For example, knowing that a customer did not churn during the observation period, but not knowing when they might churn in the future.

2. *Statistical methods*

Survival analysis uses statistical techniques that account for censored data. Methods like Kaplan-Meier estimators and Cox proportional hazards models are designed to handle censored data effectively [9].

3. *Bias*

Proper handling of censored data is critical to avoid bias. Ignoring censoring can lead to inaccurate estimates of survival probabilities and hazard rates [10].

4. *Random censoring assumption*

Often, it is assumed that censoring is random, meaning the reason for censoring is independent of the survival time. This assumption is crucial for the validity of many survival analysis methods.

2.2.2. Survival and hazard function

One of the basic methods of Time-to-Event (TTE) analysis was developed by Edward Kaplan and Paul Meyer, students of the famous statistician John Tukey, and consisted in estimating the survival function $S(t)$, which characterizes the probability of observing an event later than a certain time:

$$S(t) = P(T > t) \tag{1}$$

The survival function monotonically decreases with t. When $t = 0$, the initial value is 1, indicating that 100% of the observed subjects live at the start of the observation; in other words, none of the events of interest have happened [6].

In contrast, the cumulative death distribution function $F(t)$, which represents the probability that the event of interest occurs earlier than t, is defined as

$$F(t) = 1 - S(t), \tag{2}$$

and the death density function can be obtained as

$$f\ (t) = \frac{\partial}{\partial t}F(t) \text{ for continuous cases, and} \tag{3}$$

$$f\ (t) = \frac{[F\ (t + \Delta t) - F\ (t)]}{\Delta t}, \text{ where } \Delta t \text{ denotes a small time interval, for discrete cases.} \tag{4}$$

Figure 2 [6] shows the relationships between these functions.

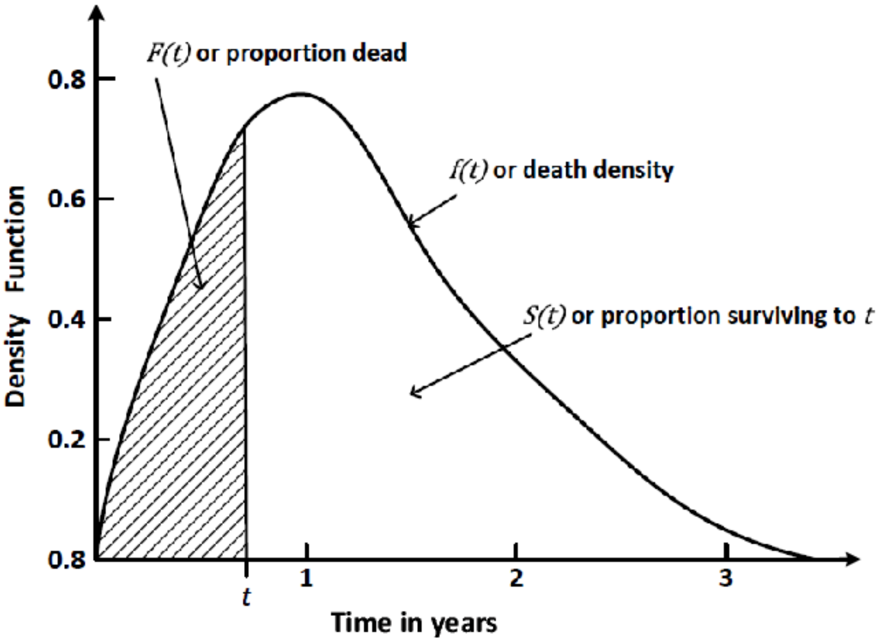


Figure 2. Relationship among different entities $f(t)$, $F(t)$ and $S(t)$

The hazard function $h(t)$, also known as the force of mortality, the instantaneous death rate, or the conditional failure rate, is another often used function in survival analysis [6].

$$h(t) = \lim_{\Delta t \to 0} \frac{P(t \le T < t + \Delta t \mid T \ge t)}{\Delta t} = \frac{f(t)}{S(t)} \tag{5}$$

The broad methods used in survival analysis can be categorized into non-parametric, semi-parametric, and parametric methods. An overview of these approaches is following.

2.2.3. Non-parametric methods

Non-parametric survival analysis methods are used to analyze and interpret survival data without assuming a specific statistical distribution for the survival times. These methods are particularly useful when the underlying survival distribution is unknown or does not follow a common parametric form.

Kaplan-Meier (KM) estimator

One of the basic methods of survival analysis was developed by Edward Kaplan and Paul Meyer, students of the famous statistician John Tukey, and involved estimating the survival function $S(t)$ [4]. This approach proposed to divide the total observation time into intervals proportional to the intensity of the recorded events, and to calculate the probability of survival in each of them. For any value of t, the estimated $S(t)$ is the product of surviving each interval up to and including time t.

The estimated $S(t)$ from the Kaplan-Meier approach can be shown as a stepwise function of time on the X-axis. Such a plot visualizes the cohort's survival experience and can estimate the median or quartiles of survival time.

As illustrated in Figure 3 [8], the survival rate is higher on the experimental therapy compared to the standard of care. Signs (+) reflect censoring, gray area indicates confidence interval, median survival is shown in black dashed line. The table contains information on the number of patients remaining in the study at a given time period.

Key features

- Provides estimates of survival probabilities at different time points (see Figure 3).
- Can compare survival curves between groups using the log-rank test*.

*The log-rank test is a non-parametric hypothesis test used to compare the survival distributions of two or more groups. It tests the null hypothesis that there is no difference in survival between the groups.

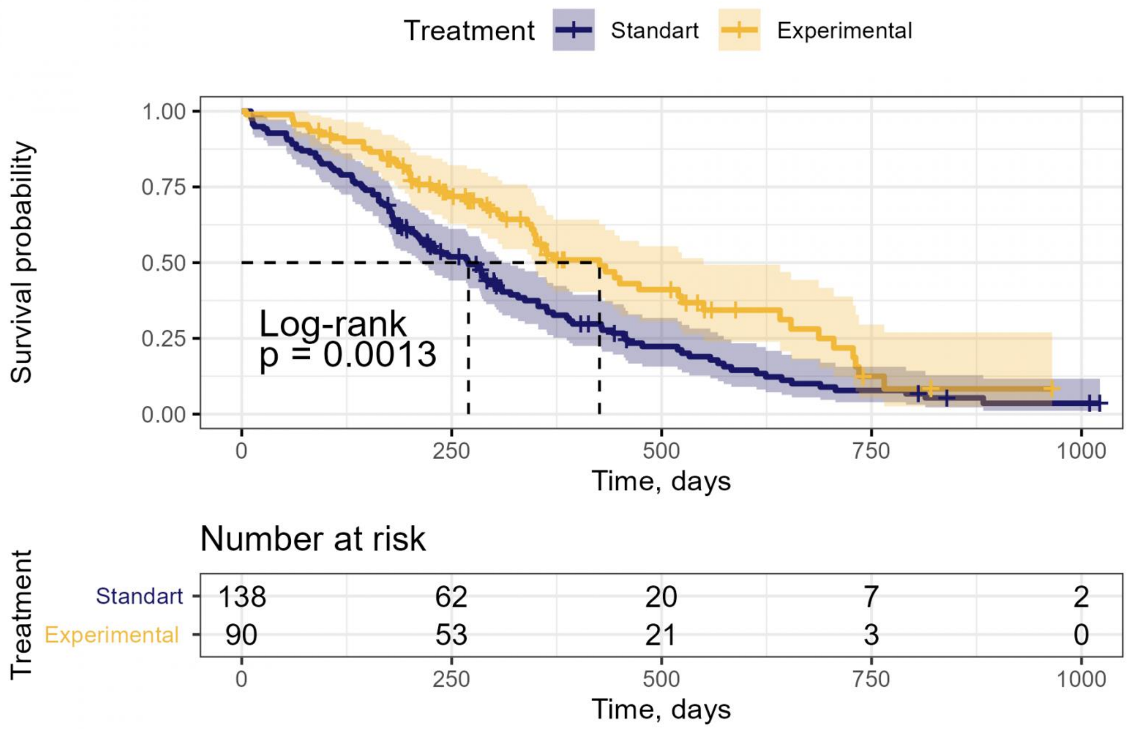


Figure 3. Example of Kaplan-Meier curve stratified by treatment group

Nelson-Aalen (NA) estimator

The Nelson-Aalen estimator [11] is another non-parametric method used to estimate the cumulative hazard function $h(t)$. The cumulative hazard function provides an estimate of the total amount of hazard experienced up to a certain time point. $S(t)$ can then be estimated using the estimate of $H(t)$. Estimates of $S(t)$ obtained using this method will always be bigger than the KM estimate, although the discrepancy between the two methods will be minimal in large samples.

Key features

- Suitable for estimating the cumulative hazard function.
- Can be used in conjunction with the Kaplan-Meier estimator to derive the survival function.

Life-Table (LT) estimator

The life table estimator is similar to the Kaplan-Meier method, with the exception that intervals are calculated using calendar time rather than observable events [5]. Since life table approaches are based on calendar intervals rather than individual events/censoring times, they use the average risk set size per interval to estimate $S(t)$ and must assume uniform censoring over the calendar time interval. As a result, the life table estimate is less precise than the Kaplan-Meier estimator, although the results will be comparable in large samples.

Key features

- Estimates $S(t)$ like Kaplan-Meier using calendar time.
- Assumes uniform censoring over the calendar time interval.
- Less accurate than Kaplan-Meier or Nelson-Aalen.

2.2.4. Semi-parametric methods

The Cox proportional model [5] is the most popular multivariable method for assessing survival data. It is essentially a time-to-event regression model that describes the relationship between event incidence (represented by the hazard function) and a set of covariates. Cox regression is classified as a semi-parametric method because the distribution of the outcome is unknown, even though it is based on a parametric regression model.

Key features

- A regression model commonly used to explore the relationship between survival time and one or more predictor variables.
- Assumes that the hazard ratio between different levels of the covariates is constant over time (proportional hazards assumption).
- The model does not assume a specific baseline hazard function, making it flexible and widely applicable.

Several helpful versions of the basic Cox model, such as the penalized Cox models [12], the CoxBoost algorithm [13], and the Time-Dependent Cox (TD-Cox) model [14], have also been proposed in the literature.

2.2.5. Parametric methods

When the time until the event of interest has a specific distribution that can be described in terms of certain parameters, parametric approaches are more effective and precise for estimate. Estimating the time to the event of interest is reasonably simple with parametric models, but it gets difficult, if not impossible, with the Cox model [15]. Linear regression is a popular parametric survival method, with the Tobit model [16], Buckley-James regression model [17], and penalized regression [18] being the most widely used linear models for survival analysis. Other parametric models, such as Accelerated Failure Time (AFT), which estimates survival time in terms of variables [14], are also commonly employed.

2.2.6. Summary

Table 1 [6] shows both the advantages and disadvantages of non-parametric, semi-parametric, and parametric methods based on theoretical and experimental analysis and lists the specific methods included.

Type	Advantages	Disadvantages	Specific methods
Non-parametric	More efficient when no suitable theoretical distributions known.	Difficult to interpret; yields inaccurate estimates.	Kaplan-Meier Nelson-Aalen Life-Table
Semi-parametric	The knowledge of the underlying distribution of survival times is not required.	The distribution of the outcome is unknown; not easy to interpret.	Cox model Regularized Cox CoxBoost Time-Dependent Cox
Parametric	Easy to interpret, more efficient and accurate when the survival times follow a particular distribution.	When the distribution assumption is violated, it may be inconsistent and can give sub-optimal results.	Tobit Buckley-James Penalized regression Accelerated Failure Time

Table 1. Summary of various statistical methods for survival analysis

2.3. Machine learning approaches

Machine learning techniques have been successful in various practical domains due to their ability to model nonlinear relationships and make accurate predictions. However, the fundamental problem for machine learning approaches in survival analysis is dealing effectively with censored information and the time estimation of the model. Machine learning is effective when there are many instances in a reasonably dimensional feature space, but this is not the case for some survival analysis issues [19]. This section gives a thorough examination of frequently employed machine learning algorithms in survival analysis.

2.3.1. Tree-based methods

Survival trees

These methods extend traditional decision trees to handle survival data. The key modification is the splitting criterion, which is adapted to maximize differences in survival times between nodes. Examples include CART for survival analysis and conditional inference trees [20].

Bagging Survival Trees (BST)

An ensemble method that builds multiple survival trees using bootstrap samples and averages their predictions [21]. BSTs are robust to overfitting and can handle high-dimensional data effectively.

Random Survival Forests (RSF)

Random Survival Forests [22] use a framework similar to bagging, but instead of using all attributes at a node, it selects a random subset of residual attributes based on the splitting criterion. Randomization reduces correlation among trees, leading to better prediction performance.

Gradient Boosting Machines (GBM)

This approach sequentially builds survival trees, where each new tree corrects the errors of the previous ensemble [23]. GBMs are flexible and often yield high predictive accuracy.

2.3.2. Bayesian methods

Naïve Bayes (NB)

Naïve Bayes, a widely used probabilistic method in machine learning, is a basic yet very successful prediction algorithm. Bellazzi and Zupan developed a naïve Bayesian classifier for clinical medicine [24], while Fard et al. used Bayesian methods with an AFT model to predict survival data for future time points [25]. However, a limitation of the Naïve Bayesian method is that it assumes independence between features, which may not always hold true in survival analysis problems.

Bayesian Networks (BN)

Bayesian Networks [26] can visually represent relationships between variables, making it easy to interpret. They can acquire knowledge from data and have been used in various models such as survival data analysis and Cox proportional hazards models.

2.3.3. Neural networks

DeepSurv

A deep learning extension of the Cox proportional hazards model, where a neural network is used to model the complex relationships between covariates and the hazard function [27].

Recurrent Neural Networks (RNNs)

Suitable for handling time-dependent covariates and capturing temporal dependencies in survival data. Most prominent models are WTTE-RNN [28] and RNN-SURV [29].

Autoencoders

Used for dimensionality reduction, these neural networks learn compact representations of high-dimensional survival data, which can then be used for further survival modeling. One of examples is SAVAE [30].

2.3.4. Kernel-based methods

Support Vector Machines (SVM)

These methods adapt SVM to handle censored data by modifying the loss function to account for partial information.

Gaussian processes for survival analysis

This non-parametric method [31] models survival functions as distributions over possible functions, allowing for flexible and probabilistic modeling of survival data.

2.3.5. Other methods

K-Nearest Neighbors (KNN)

Adapts KNN to censored data by considering the survival times of the nearest neighbors and weighting them appropriately.

Penalized regression models

Methods like Lasso and Ridge regression are used to handle high-dimensional covariates by adding regularization terms to the Cox model, improving model stability and interpretability.

2.3.6. Summary

The section focuses on the adaptation of machine learning algorithms to survival analysis. They surpass statistical methods in managing complicated and high-dimensional data, capturing non-linear relationships, and making robust predictions in the presence of censored observations. The survey [6] demonstrates the extensive application and efficiency of these strategies in a variety of survival analysis settings.

CHAPTER 3: DATA STRUCTURE AND TRANSFORMATION

All the data analysis and model training sources are available at GitHub [42].

3.1. Data and target variable structure

The hosts of the competition provided the data. The primary takeaways from its collection and preprocessing are:

- The data includes *client information*, *report dates*, and *transaction flow*.
- The information was gathered between *October 20, 2021*, and *March 20, 2023*, a duration of one year and five months.
- All clients with more than 90 days (window) between any adjacent transaction were removed from the database. This was done to *assure consistency*: *nobody must* unexpectedly 'resurrect' (*renew transactions*) *after the predetermined window*.

3.1.1. Dealing with churn

To understand the structure of the task, let's describe the initial process for dealing with churn.

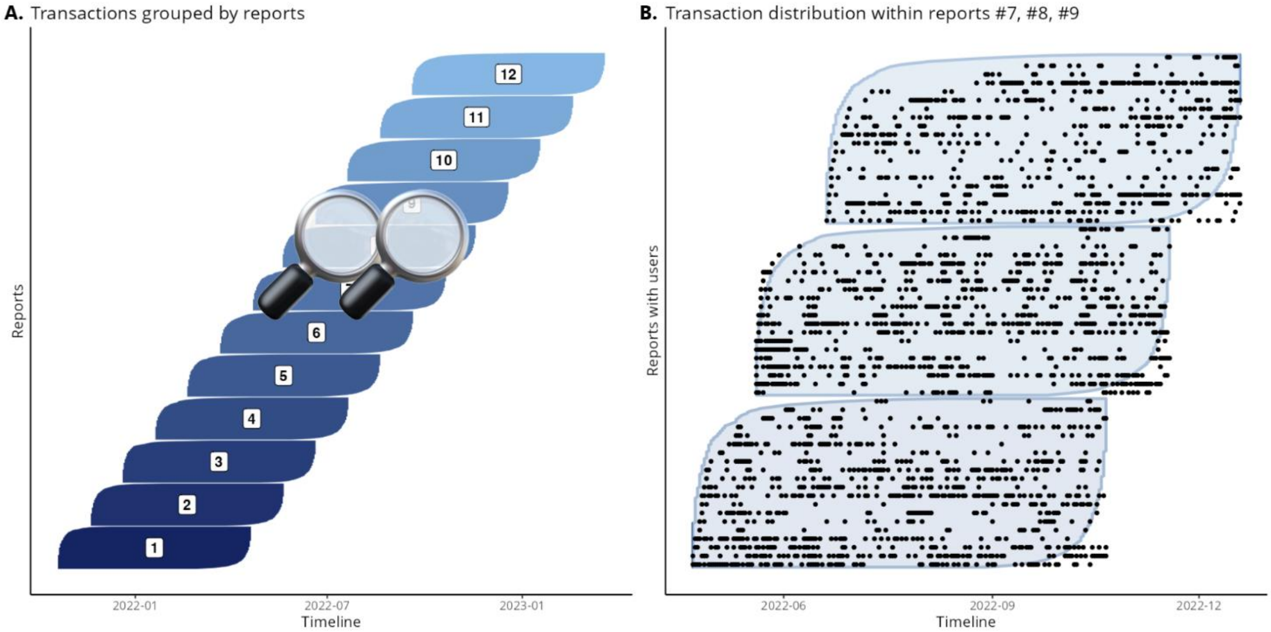


Figure 4. A. Transactions grouped by reports; B. Transaction distribution within reports #7, #8 and #9

1. Every month, on the last day of the month, a “report” is collected, as shown in Figure 4. Not all clients of the bank are included in it.
2. During the report, the information on the clients that got into it is recorded. This includes “starting a timer” for each of the customers (see Figure 5).
3. If in the period from 60 to 90 days after the report a business rule is triggered, the main essence of which is the absence of transactions, the client is considered to be “caught in the churn”. Because the presence of transactions after the report is uncertain, any transaction data obtained after the report is censored for survival analysis and the last transaction appears within a 90 days window before the report date, as illustrated in Figure 5.

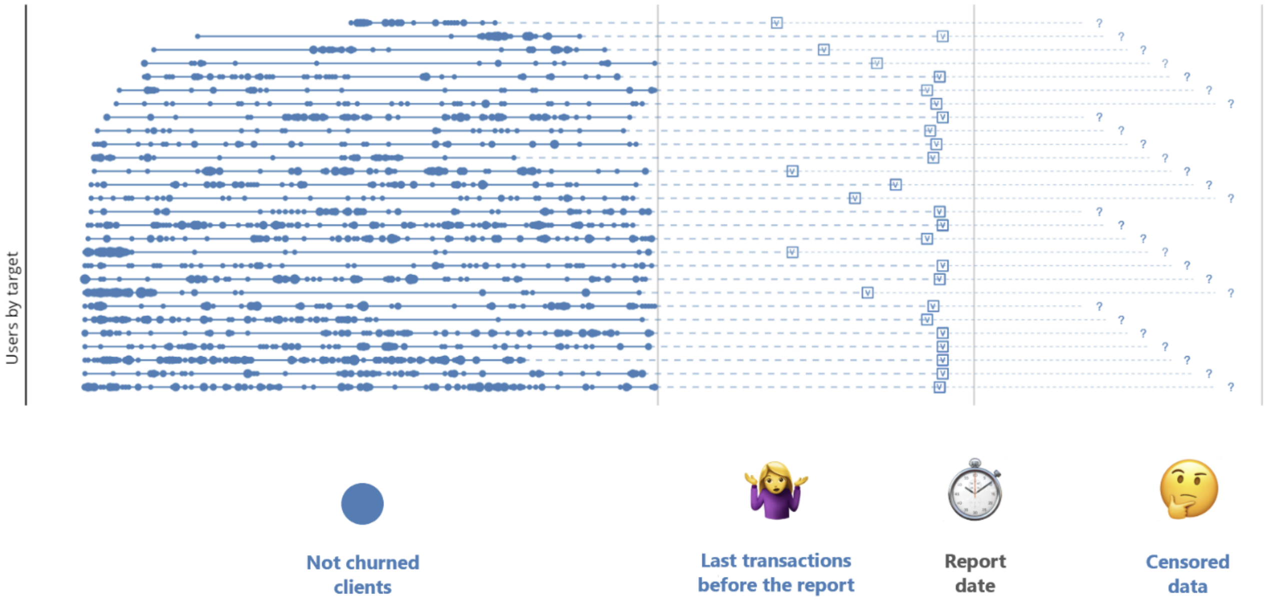


Figure 5. Transactions and censored data on a timeline

3.1.2. Clinical and documented churn

The above-described procedure was used to generate the initial churn labels for this competition. As is obvious, this definition requires that *the client's most recent transaction occurred prior to the report date*. In this scenario, we can consider the following two definitions of churn (see Figure 6):

“Documented” churn occurs when *matured tags activate a business rule* (60-90 days after the report).

"Clinical" churn is an event in which a *customer "stopped the pulse" of transactions until the last transaction* (prior to the report). After 60-90 days of no transactions, this customer will be classified as a *"Documented" churn*.

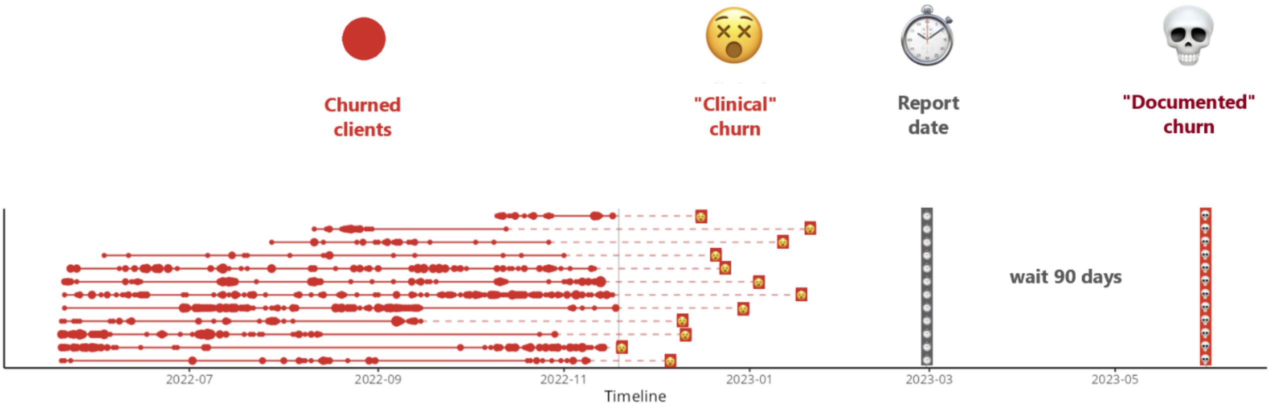


Figure 6. "Clinical" and "Documented" churn on a timeline

To make churn prediction more proactive, *the challenge of the competition is to predict "Clinical" churn 90 days in advance*.

3.1.3. Summary

The data for each client in the competition is organized as follows:

Top level

1. Select a report with a specified date. The data includes 12 consecutive reports (1–12), as illustrated in Figure 4.

2. Clients who appear in that report are picked. These are non-overlapping client sets; each customer in the competition is assigned to a single report.

3. Within the competition, with a few assumptions, the reports can be considered as individual repeated fixed-length studies (Time-to-Event / Survival context), cohorts within a product analysis (Buy Till You Die context), or another variable (in a broad DS/ML context).

Client level

1. Transactions for the nine months leading up to the report date are collected for a client.
2. Out of these 9 months of transactions, the first 6 months are provided to the participants for model training, as shown in Figure 4. The last 3 months are used to construct the target variable.
3. The target variable is constructed as follows: the date of the last transaction in the 3 months preceding the report date (expected date of *"Clinical" churn*), and a label on whether this transaction was the last transaction (confirmation of *"Documented" churn*).
4. Thus, 6 months of transaction history is used to train the models, and 6 to construct the target variable.

3.2. Datasets

Several datasets and artifacts are available to participants:

3.2.1. Tabular client data

1. Basic information about all *96,000* clients in tabular .csv format: clients.csv (2.5 MB).
2. Training data on the *target variable* and *last transaction time* for *64,000* clients: train.csv (745 KB).
3. Reporting information where clients are grouped by time: report_dates.csv (288 KB).

3.2.2. Client transaction data

Client transactions for all *96,000* clients (approximately *13 million*) in tabular .csv format: transactions.csv.zip (197 MB).

3.2.3. Fields

1. clients.csv – basic information about clients:
   - user_id – bank client ID,
   - report – number of one of 12 reports this client was included in,
   - employee_count_nm – information about client's employer: number of employees in the company,
   - bankemplstatus – information about whether the client is an employee of the bank,
   - customer_age – obfuscated age of the client (one of 4 groups 0...3).
2. train.csv – training data with target variable about clients:
   - user_id – bank client ID,
   - target – client's churn label (target class of churn event is 1),
   - time – in how many days the last transaction of the client will occur (can be used as survival time in Time-to-Event analysis).
3. report_dates.csv – information about report dates:
   - report – report sequence number,
   - report_dt – report date.
4. transactions.csv.zip – archive with transactions of bank clients:
   - user_id – bank client ID,
   - mcc_code – transaction Merchant Category Codes (MCC),
   - currency_rk – currency of transaction (1 – RUB, 2 – EUR, 3 – USD, 0 – any other currency),
   - transaction_dttm – date and time of transaction execution,
   - transaction_amt – amount in transaction currency.

3.2.4. Summary

Figure 7 presents a diagram showing the structure of the datasets and how they are interconnected.

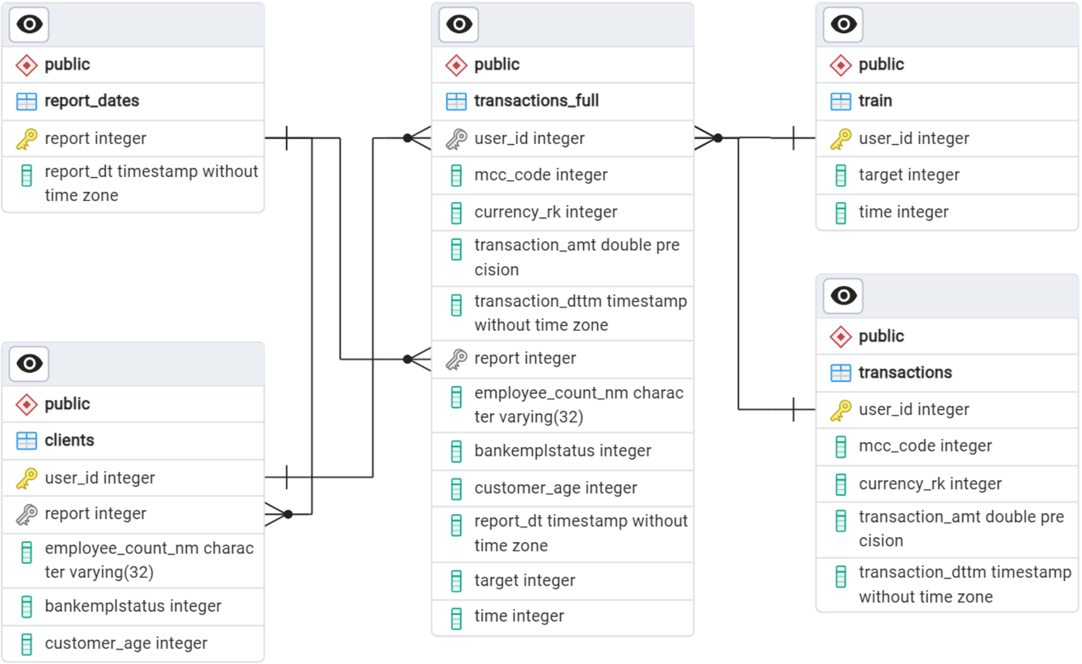


Figure 7. The entity relationship diagram of data

3.3. Exploratory data analysis

Exploratory Data Analysis (EDA) was carried out, which involved gathering and analyzing basic statistics for the available features and the target variable. This section provides a comprehensive overview of drawn conclusions.

3.3.1. Target variable

The target variable, which indicates if a client has churned (1 – positive class) or not (0 – negative class), is expected to be imbalanced with a ratio of positive to negative classes greater than ten. Figure 8 depicts the class disparity and churn rate over a 90 day period. The number of customers who churned remains fairly consistent, except for a few fluctuations at the beginning and end of the period. In contrast, the number of customers who did not churn exponentially grows over time, leading to a class imbalance.

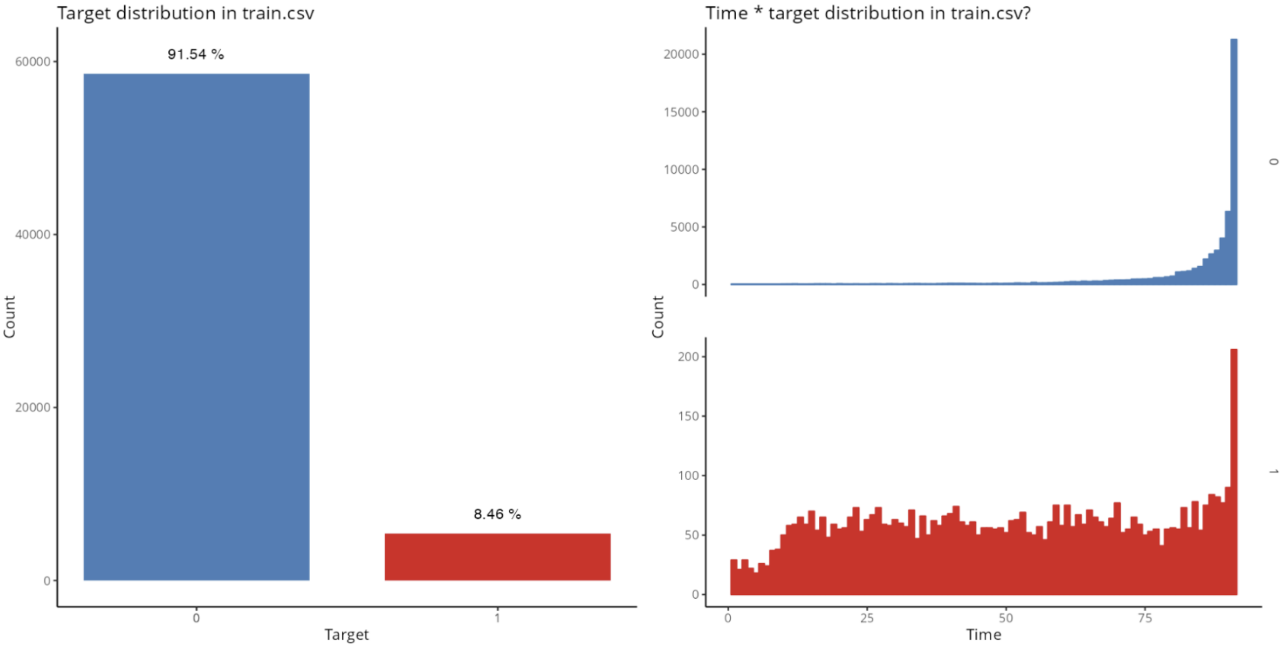


Figure 8. Target distribution in train dataset

When dealing with a highly imbalanced dataset, it is considerably more essential to examine the metric's applicability. Instead of relying on metrics such as Accuracy or Recall which are influenced by arbitrary thresholds, it is better to choose metrics like ROC-AUC (Area Under the Receiver Operating Characteristic Curve).

The Population Stability Index (PSI) is a tool used to compare the distributions of users who have churned and those who have not. With a PSI of 0.05%, which is below the threshold of 0.1%, we can conclude that there has been little to no significant change in the churn variable between 2022 and 2023. Therefore, it is reasonable to consider the reports as consistent Time-To-Event studies with a fixed duration.

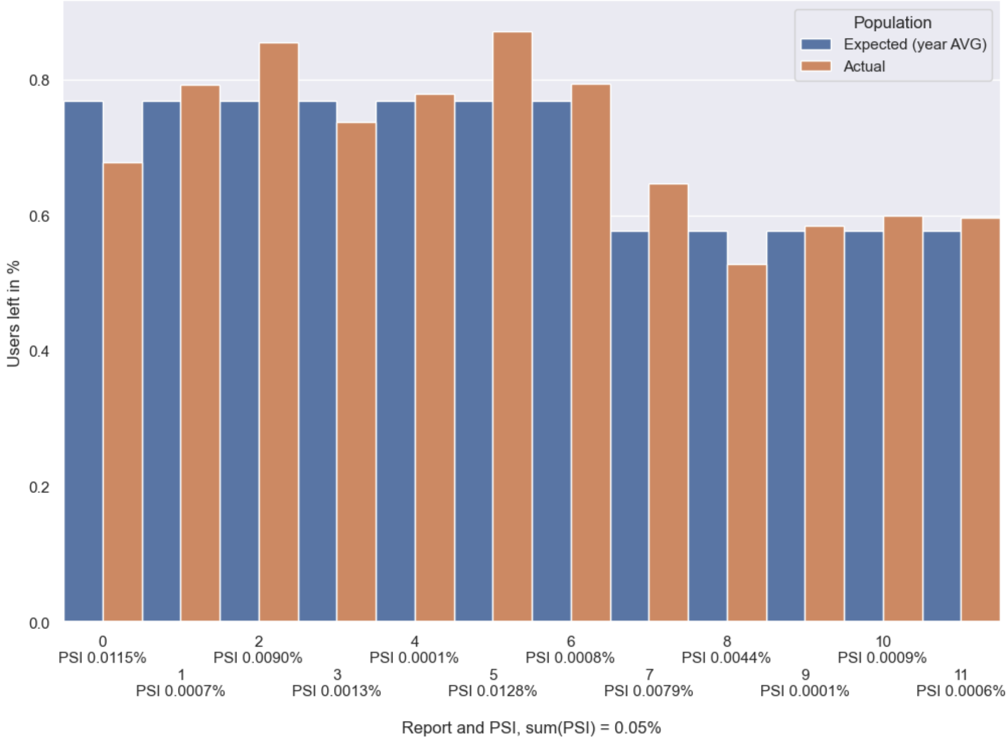


Figure 9. Target distribution and PSI across reports

3.3.2. Concordance Index

The Concordance Index (CI) or (Harrel's) C-index is a generalization of the area under the ROC curve that can take into account censored data. It is calculated as the proportion of concordant pairs divided by the total number of possible evaluation pairs. Let's look at some examples to better understand what this definition entails.

We are a bank with 5 customers and we are trying to predict who will leave the bank first. Alex left after 1 year, Bron after 2 years, Cate after 3 years, Dave after 4 years, and Eric after 5 years.

Assume that the algorithm predicted: *Alex will churn after 2 years, Bron after 4 years, Cate after 7 years, Dave after 8 years and Eric after 10 years*, as illustrated in Table 2.

Name	**Alex**	**Bron**	**Cate**	**Dave**	**Eric**
Churn times (T)	1	2	3	4	5
Predictions (X)	2	4	7	8	10

Table 2. Example with a Concordance Index of 1

The concordance index is equal to its maximum value 1 (even though none of the individual predictions are correct), because all customer pairs are concordant – for any pair $X_1 < X_2$, it's true that $T_1 < T_2$. This means that larger values of X imply larger values of T. Therefore, rather than the actual predictions, the concordance index is concerned with the sequence in which they were made.

This is very different from other evaluation metrics like mean squared error or mean absolute error. Actually, for any strictly rising function of churn times, the concordance index will remain equal to one.

Assume another example: *Alex will churn after 5 years, Bron after 2 years, Cate after 1 year, Dave after 10 years and Eric after 7 years*, as depicted in Table 3.

Name	**Alex**	**Bron**	**Cate**	**Dave**	**Eric**
Churn times (T)	1	2	3	4	5
Predictions (X)	5	2	1	10	7

Table 3. Example with a Concordance Index of 0.6

There are $\binom{5}{2} = \frac{5!}{2! \times (5-2)!} = 10$ pairs. 4 of them are discordant:

1. (Alex, Bron): Alex is expected to leave after Bron, while Bron left after Alex.
2. (Alex, Cate): Alex is expected to leave after Cate, while Cate left after Alex.
3. (Bron, Cate): Bron is expected to leave after Cate, while Cate left after Bron.
4. (Dave, Eric): Dave is expected to leave after Eric, while Eric left after Dave.

The concordance index is $6/10 = 0.6$.

Suppose *Cate and Dave will both churn in 3 years*, as shown in Table 4. In this instance, they will be considered half concordant pairs.

Name	**Alex**	**Bron**	**Cate**	**Dave**	**Eric**
Churn times (T)	1	2	3	4	5
Predictions (X)	1	2	3	3	5

Table 4. Example with a Concordance Index of 0.95

Cate and Dave are supposed to leave at the same time, while Dave left after Cate.

The concordance index is $(9 + 0.5)/10 = 0.95$.

The concordance index can handle situations where the event being studied has only occurred for some of the observations by the end of the study, known as right *censoring*. This information tells us that if the event does occur, it will happen at a time equal to or greater than the specified time.

Suppose that *Alex churned after 3 years, Bron hasn't churned after 2 years, Cate churned after 3 years, Dave hasn't churned after 6 years* and *Eric churned after 1 year*.

Name	**Alex**	**Bron**	**Cate**	**Dave**	**Eric**
Churn times (T)	1	2	3	4	5
Predictions (X)	3	2	3	6	1
Event observed	Yes	No	Yes	No	Yes

Table 5. Example with a Concordance Index of 0.42

It is important to note that Bron has not left after 2 years, indicating that if he does leave, it will be after at least 2 years. Similarly, Dave may leave after at least 6 years. As a result, we cannot assign a score to the pairs (Bron, Cate), (Bron, Dave), (Bron, Eric) and (Dave, Eric) because we have no idea who churned first (see Table 5).

Thus, we only have 6 potential pairs (see Figure 10). 2 of them are discordant:

1. (Alex, Bron): Alex is expected to leave after Bron, while Bron left after Alex.
2. (Alex, Eric): Alex is expected to leave after Eric, while Eric left after Alex.
3. (Cate, Eric): Eric is expected to leave after Cate, while Cate left after Eric.

One of them is half-concordant:

1. (Alex, Cate): Eric is expected to leave after Cate, while Cate left after Eric.

The concordance index is $(0 + 1/2 + 1 + 0 + 1 + 0)/6 = 2.5/6 \sim 0.42$.

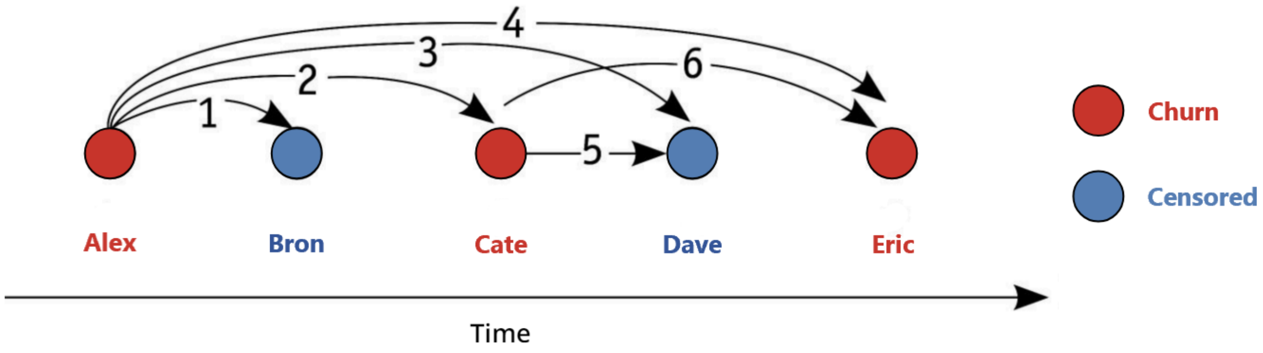


Figure 10. Concordance Index with censored data

3.4. Data Preprocessing

3.4.1. Handling NULLs

Only *employee_count_nm* categorical feature in *clients* dataset, representing the number of employees a client’s employer has, contains *NULL* values. Given that a ranking representation is implied by the feature's nature, we encoded it in ascending order according to Table 6.

Value	NULL	НЕТ	ДО 10	ОТ 11 ДО 30	ОТ 11 ДО 50	ОТ 31 ДО 50	ОТ 51 ДО 100
Code	0		1	2	3	4	5

Value	ОТ 101 ДО 500	БОЛЕЕ 500	ОТ 501 ДО 1000	БОЛЕЕ 1001
Code	6	7	8	9

Table 6. Category mapping of employee count

3.4.2. Transaction datetime

The first transaction datetime is 09:00:00 on October 20, 2021.

The last transaction datetime is 20:59:58 on March 20, 2023.

Table 7 contains features extracted from every transaction during preprocessing.

Feature	Date	Day of Week	Year	Month	Hour
Codename	date	weekday	year	month	hour
Scalar type	datetime.date	integer			

Feature	Saturday or Sunday	January or February	November or December	Season of Year	Time of Day
Codename	is_weekend	begin_of_year	end_of_year	season_of_year	time_of_day
Scalar type	integer			string	
Description	One of				
	0 (No) or 1 (Yes)			winter, spring, summer, fall	morning, day, evening, night

Table 7. Transaction features

Figure 11 illustrates the spread of transactions over time, revealing a normal distribution pattern.

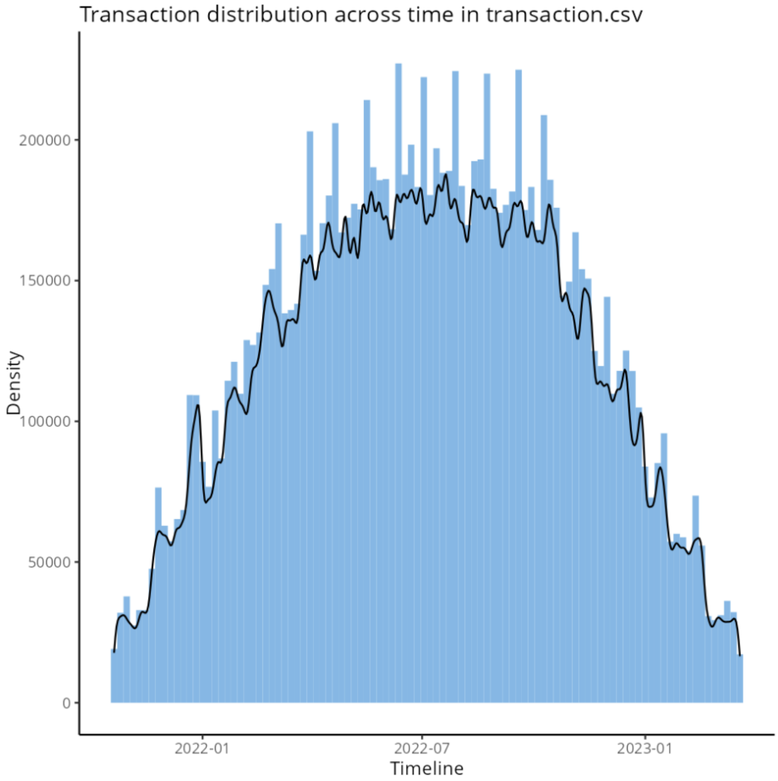


Figure 11. Transaction distribution across time

3.4.3. Normalizing transaction amounts

The majority of transactions are made in Russian Rubles, accounting for 97.1% of all transactions. Transactions in US Dollars and Euros each make up only 0.0001%, while other currencies account for nearly 2.9% of all transactions.

All transaction amounts are converted to RUB using the official exchange rate supplied by the Central Bank of the Russian Federation [32].

Class 0 is represented by the Chinese Yuan (CYN) rate, because:

1. According to Forbes [33], it is the third most popular currency in Russia in 2022–2023. The USD and EUR rank first and second, respectively.
2. Rambler [34] reports that in February 2023, it became the most traded currency on Moscow Exchange.

3. The CYN rate, which is approximately 10 RUB, has been stable over the observed time period (except from a brief interval in the spring of 2022).

Furthermore, inflation is also taken into consideration. We want to see how prices have changed each month since December 2021 using data from Rosstat [35]. We are starting with October as our starting point with no inflation. Monthly and cumulative inflation rates are shown in Table 8.

Year	**Month**	**Monthly Inflation %**	**Cumulative Inflation %**
2023	3	0,37	15,84
2023	2	0,46	15,41
2023	1	0,84	14,88
2022	12	0,78	13,92
2022	11	0,37	13,04
2022	10	0,18	12,63
2022	9	0,05	12,42
2022	8	-0,52	12,37
2022	7	-0,39	12,95
2022	6	-0,35	13,40
2022	5	0,12	13,79
2022	4	1,56	13,66
2022	3	7,61	11,91
2022	2	1,17	4,00
2022	1	0,99	2,80
2021	12	0,82	1,79
2021	11	0,96	0,96
2021	10	0,00	0,00

Table 8. Inflation in Russia from October 2021 to March 2023

Table 9 show additional normalized features added to transaction dataset after normalization. Note that normalized transactions with zero sums are not included.

Feature	Normalized Transaction Amount	Positive Normalized Transaction Amount (Deposits)	Negative Normalized Transaction Amount (Expenses)
Codename	normal _transaction_amt	plus_normal _transaction_amt	minus_normal _transaction_amt
Scalar type	float		

Table 9. Normalized transaction amount features

3.5. Feature generation: aggregated transaction features

3.5.1. Basic features

Table 10 contains summarized transaction data grouped by user for the initial transaction amount. The features for normalized, positive normalized, and negative normalized transaction amounts are generated in analogous manner.

Feature	Number	Sum	Maximal Amount	Median	Minimal Amount
Codename	amt_count__agg_user	amt_sum__agg_user	amt_max__agg_user	amt_median__agg_user	amt_min__agg_user
Scalar type	integer	float			

Feature	Standard Deviation	Variance	Fisher-Pearson Coefficient of Skewness
Codename	amt_std__agg_user	amt_var__agg_user	amt_skew__agg_user
Scalar type	float		

Table 10. Transaction amount features grouped by user

Additional essential features developed in this stage are listed in Table 11.

Feature	Unique MCC Count	Unique Currency Count	Minimal Date	Maximal Date	Count Unique Weekday
Codename	uniq_mcc__agg_user	uniq_currency__agg_user	min_date__agg_user	max_date__agg_user	count_weekday__agg_user
Scalar type	integer	integer	datetime.date	datetime.date	integer
Description	–				Number of unique days of the week when the client made transactions.

Table 11. Additional transaction amount features grouped by user

3.5.2. MCC grouped features

Find the 51 Merchant category codes that have at least 20000 transactions each (51 such codes), and use them as distinct features for categorization. Divide the MCCs into 8 groups and analyze the normalized transaction amounts (positive, negative and combined) within each group for various users. Some examples of the resulting features can be found in Table 12.

Feature	Number of Transactions in a given MCC group	Median of Transactions in a given MCC group	Sum of Transactions in a given MCC group
Codename	count___minus_normal_transaction_amt___cat_mcc_code_0	median___minus_normal_transaction_amt___cat_mcc_code_0	sum___minus_normal_transaction_amt___cat_mcc_code_0
Scalar type	integer	float	

Table 12. Transaction amount features grouped by user and MCC

3.5.3. Date related features

Table 13 includes important date-related features, excluding *max_date* and *min_date* which are already listed in Table 11.

	Days Between				
Feature	Report Date And Last Transaction Date	Report Date And First Transaction Date	First Transaction Date And Last Transaction Date	Last Transaction Date And 01.01.2010	First Transaction Date And 01.01.2010
Codename	delta_report_last_date	delta_report_first_date	delta_first_last_date	days_max_date	days_min_date
Scalar type	integer				

Table 13. Date related features grouped by user

3.5.4. Active and non-active periods

The next set of features examines client activity by studying the flow of transactions. A transaction is considered part of an active period if it occurred within the last 24 hours of the previous transaction. Active and non-active period statistics are described in Table 14.

Feature	Number of Active Periods	Transactions occurred on how many days?	Median of Non-Active Period in Days	Max of Non-Active Period in Days	Total Number of Days Between First and Last Transactions
Codename	active _period	unique_ active_days	median_non _active_period	max_non _active_period	all_period
Scalar type	integer				

Feature	unique_ active_days Divided by Number of Transactions	unique_ active_days Divided by Number of Transactions	unique_ active_days Divided by all_period	unique_ active_days Divided by all_period
Codename	prc_non _active _period	freq_active _transact _count	freq_active _day	prc_non _active _period _by_days
Scalar type	float			

Table 14. Active and non-active periods grouped by user

3.5.5. Activity period change rate

Divide transactions into intervals of 30 days and find ratios of sums among the current period (T), the previous period (T-1) and the period before that (T-2) – 3 comparisons (T vs T-1, T vs T-2 and T-1 vs T-2) for deposits and expenses separately, so 6 features in total. This method allows for a more detailed examination of transaction patterns and trends over time. Table 15 contains the full definition of all features generated in this step.

Feature	Sum of Positive Transactions 30 days Before the Last One	Sum of Positive Transactions Between 30 and 60 days Before the Last One	Sum of Positive Transactions Between 60 and 90 days Before the Last One
Codename	cur_period _sum_plus	pre_period _sum_plus	pre_pre_period _sum_plus
Scalar type	float		

Feature	cur_period _sum_plus Divided by pre_period _sum_plus	cur_period _sum_plus Divided by pre_period _sum_plus	cur_period _sum_plus Divided by pre_period _sum_plus
Codename	plus_speed_cur	plus_speed_pre	plus_speed _cur_prepre
Scalar type	float		

Table 15. Activity period change rate grouped by user

3.6. Feature generation: Catboost for time prediction

Stratified 5-fold cross validation (CV) was used to train a CatBoost regression model for predicting the timing and quantile values of the last transaction. The model underwent 2000 iterations with overfitting detection enabled, and the mean absolute error was used as the evaluation metric. Feature importance is illustrated in Figure 12.

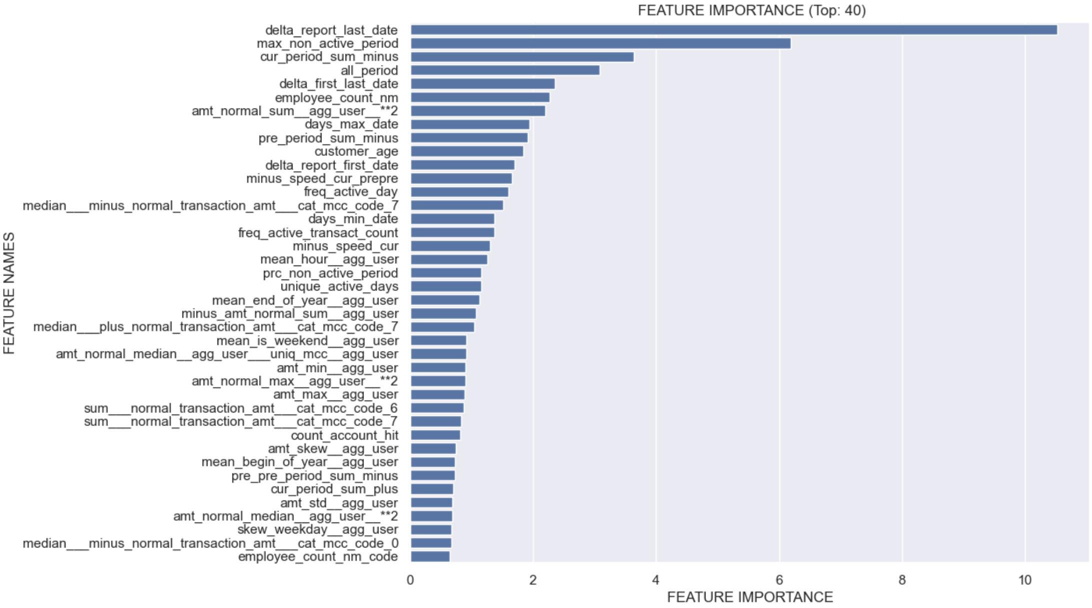


Figure 12. Feature importance sorted by descending significance

3.7. Summary

This section discusses the difficulties and steps involved in getting data ready for predicting customer churn.

The section begins by explaining the data structure and target variable, differentiating between clinical and documented churn. The datasets include client information and transaction records, totaling around 13 million transactions. The Exploratory Data Analysis focuses on understanding the properties of the target variable and datasets, and selecting the appropriate metric for modeling – the Concordance Index.

Feature generation involves grouping transactions by user and, optionally, by MCC, to create features by calculating aggregated values. Analyzing periods of activity and inactivity can help identify behavioral patterns important for predicting churn. The CatBoost regression model is utilized to predict the time of the last transaction and its quantile levels. At the end, we have 166 features: 156 numerical, 3 categorical and 7 last transaction time predictions.

Thorough preprocessing steps, like data normalization and feature engineering, are crucial for enhancing a model's performance. By meticulously preparing and creating features, this stage lays the groundwork for developing a dependable and accurate customer churn prediction model that can effectively tackle the challenges of churn prediction.

CHAPTER 4: TRAINING MACHINE LEARNING MODELS

4.1. Introduction

In this chapter, we focus on the process of training machine learning models to predict customer churn probabilities. This classification task aims to identify customers likely to churn, thus enabling proactive retention strategies. Seven classifier models were employed: *RandomForest*, *EasyEnsemble*, *ExtraTrees*, *GradientBoosting*, *AdaBoost*, *XGBoost*, and *CatBoost*. Each model was trained on a subset of the data with parameters detailed in Table 16.

Model	**Parameters**		**Mean CV ROC-AUC**	**Test CI**	**Run Time MM:SS**
	Name	**Value**			
RandomForest	n_estimators n_jobs class_weight	100 -1 {0: 0.54, 1: 5.94}	0.7408	0.7357	1:14
EasyEnsemble	n_estimators n_jobs	100 -1	0.7553	0.7577	02:42
ExtraTrees	n_estimators n_jobs class_weight	100 -1 {0: 0.54, 1: 5.94}	0.741	0.759	00:22
GradientBoosting	n_estimators class_weight learning_rate	100 {0: 0.54, 1: 5.94} 0.02	0.7473	0.7577	42:15
AdaBoost	n_estimators class_weight learning_rate	100 {0: 0.54, 1: 5.94} 0.02	0.7305	0.7417	11:03
XGBoost	n_estimators learning_rate objective verbosity device	100 0.02 binary:logistic 0 cuda	0.7586	0.7676	0:10
CatBoost	n_estimators learning_rate eval_metric verbose early_stopping_rounds iterations	100 0.02 AUC False 500 1000	0.7674	0.7709	03:05

Table 16. Model parameters and metrics

4.2. Model training and evaluation

To handle class imbalance and minimize overfitting, we calculated the mean ROC-AUC metric using stratified 5-fold cross-validation for each model. This approach ensures a balanced evaluation by considering both the true positive rate and false positive rate. Following this, we calculated the concordance index on an unseen testing subset to further assess model performance in predicting the probabilities of churn.

4.3. Classifier models

RandomForest [36]: Constructs multiple decision trees and outputs the mode of the classes for classification tasks. It is robust to overfitting due to its ensemble nature.

ExtraTrees [36]: Similar to *RandomForest*, but with more randomness injected into the decision trees, potentially enhancing generalization.

EasyEnsemble [37]: An ensemble method designed to handle class imbalance by creating balanced subsets through random undersampling and training multiple s.

GradientBoosting [36]: Builds an ensemble of trees in a sequential manner, where each tree attempts to correct the errors of its predecessor. This model is known for its high accuracy and robustness.

AdaBoost [36]: An adaptive boosting algorithm that combines multiple weak classifiers to form a strong classifier, adjusting weights based on classification errors.

XGBoost [38]: An efficient and scalable gradient boosting framework known for its speed and performance, especially when run on a GPU.

CatBoost [39]: A gradient boosting library that natively supports categorical features and provides robust performance without extensive hyperparameter tuning.

4.4. Model performance

After extensive testing, *XGBoost* and *CatBoost* demonstrated the highest potential (see Table 16). They achieved the highest validation scores and exhibited optimal runtime efficiency. *XGBoost*, in particular, benefited from Nvidia CUDA GPU training, significantly enhancing its training speed. However, it is important to note that *CatBoost* does not support the ROC-AUC metric when trained on a GPU.

4.5. Survival curve

Figure 13 illustrates global and report-wise Kaplan-Meier survival curves.

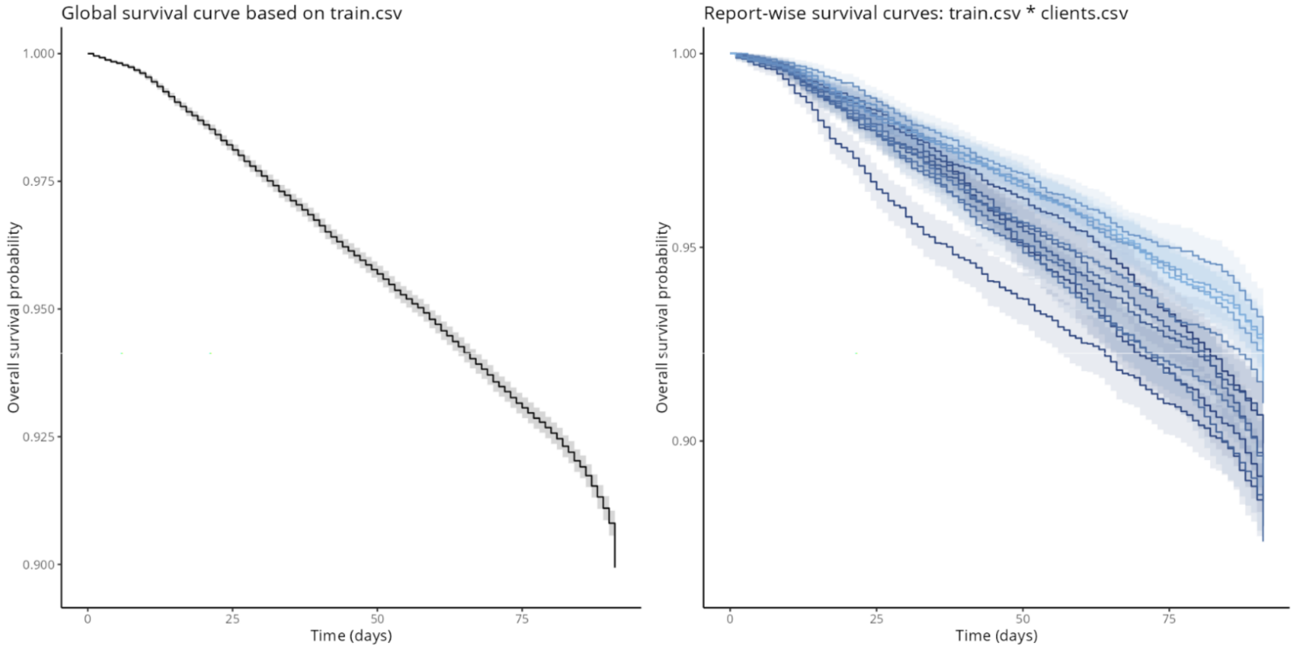


Figure 13. Global and report-wise survival curves

4.6. Removing outliers improves the score

Removing 296 clients who made over 20 transactions per day and 93 clients who made transactions with 11 different merchant category codes in a day improves the performance of the model. There are 364 clients who belong to both outlier groups – if we drop them from the train dataset, 63636 users are left.

4.7. Hyperparameter optimization

To further enhance the performance of *XGBoost* and *CatBoost*, we utilized the *Optuna* [40] framework in Python for hyperparameter optimization. *Optuna's* sophisticated search algorithm efficiently navigated the parameter space, identifying multiple optimal parameters. These parameters were then saved for subsequent stacking. Overall, 12 XGBoost and 16 CatBoost models were saved.

4.8. Model stacking and ensemble

The best combination of models was chosen using preliminary weight optimization with *Optuna*. In the final ensemble, we combined 4 XGBoost and 2 CatBoost models. The stacking approach leverages the strengths of individual models, creating a robust ensemble that performs better than any single model.

We used *Optuna* for weighted averaging to optimize the model weights within the ensemble. This process ensured that the final predictions were a weighted combination of the individual models' predictions, enhancing overall performance.

4.9. Final feature selection

The final step involved performing feature selection on the ensemble model. By identifying and retaining the most significant features, we reduced model complexity and improved interpretability, potentially enhancing performance.

The resulting concordance index is 0.79, which is equal to the area under the ROC curve shown in Figure 14.

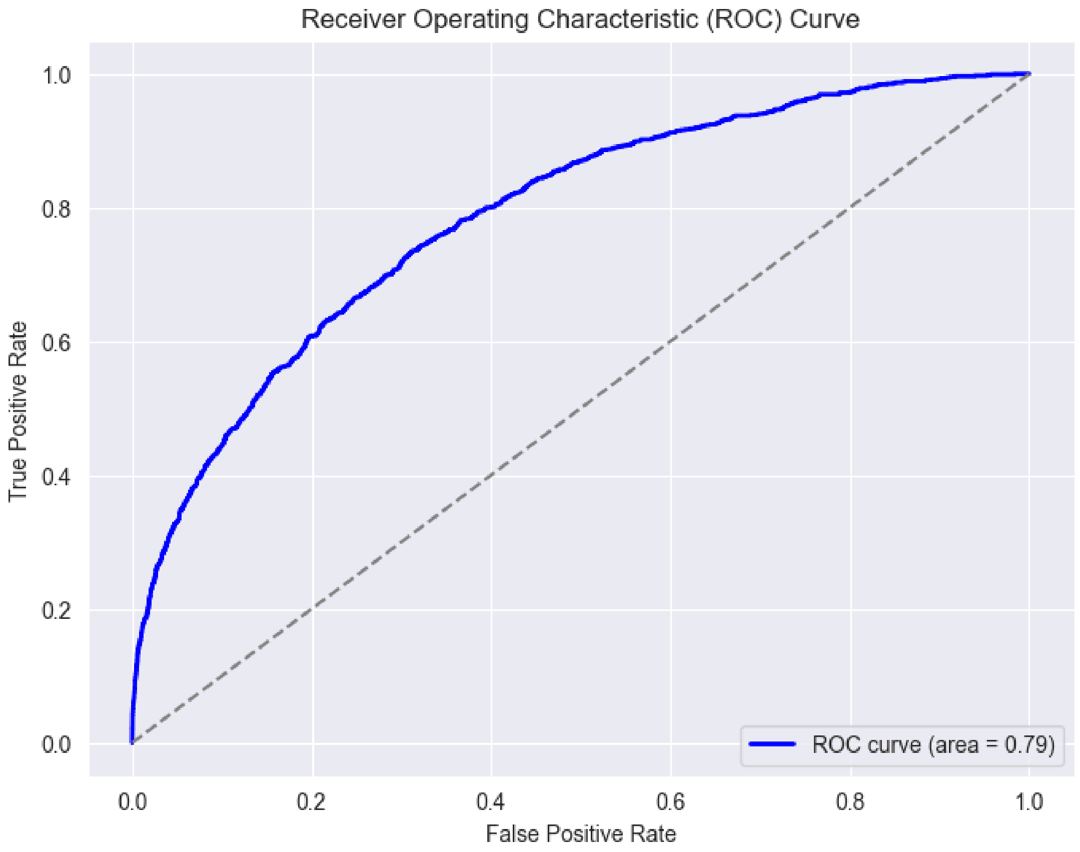


Figure 14. Area under the ROC curve

4.10. Shapley additive explanations

Shapley additive explanations (SHAP) [41] values give the contribution of a model feature to a prediction. Let’s have a look at some examples.

Our model predicts with a high probability of 78% that user 538675 will churn from the bank, while the client actually left. According to Figure 15, the main contributors are the time predictions, the number of deposits made (74 account hits), the median normalized transaction amount in MCC group 7 and others.

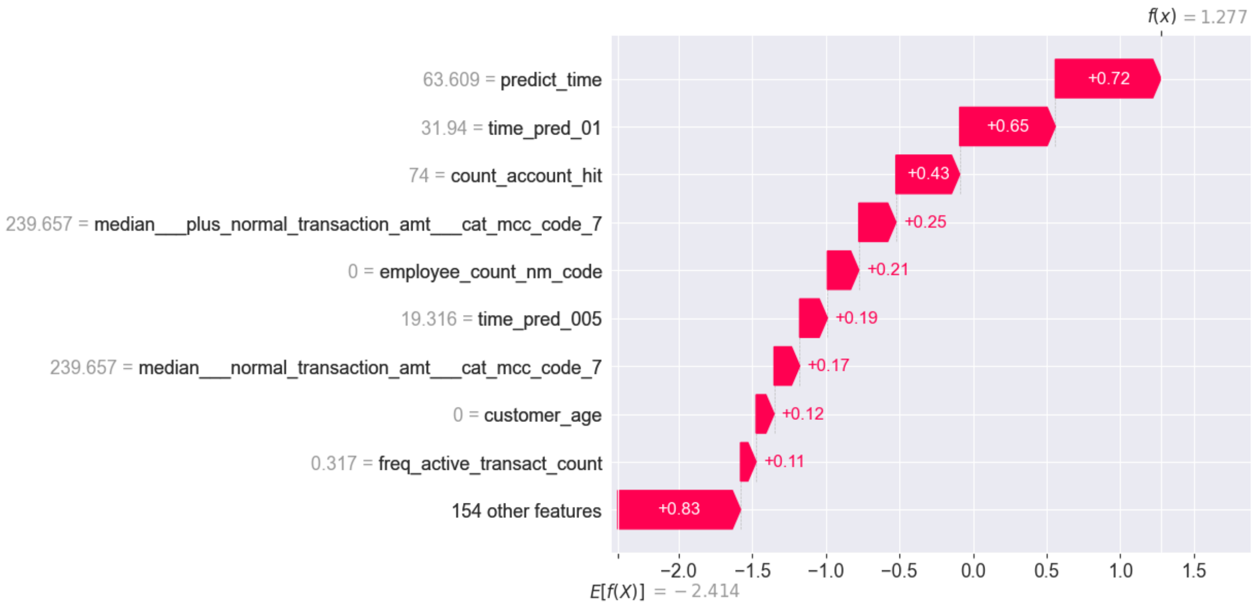


Figure 15. Feature contributions for user 538675 who churned

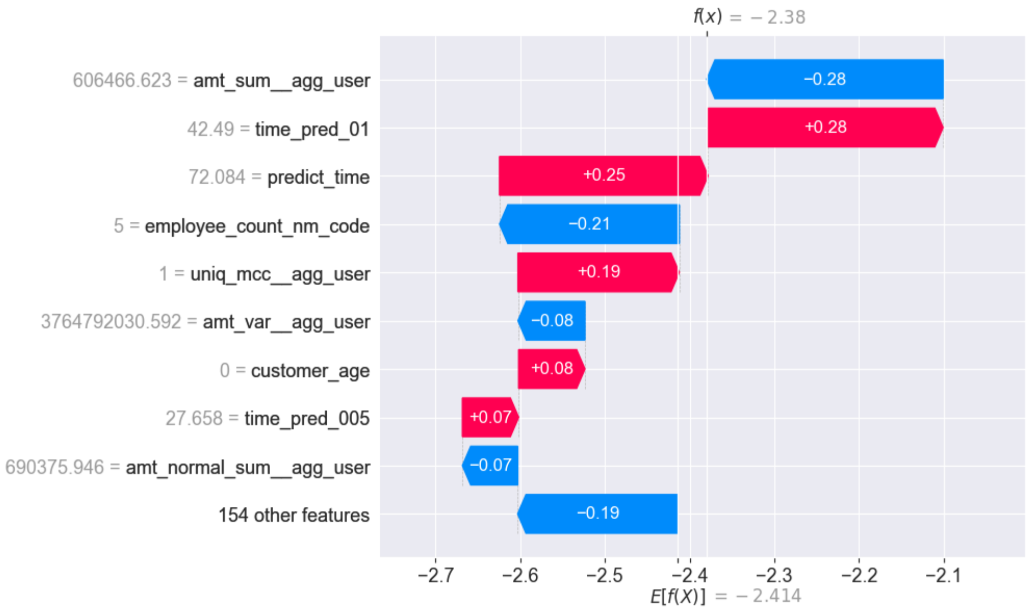


Figure 16. Feature contributions for user 112038 who stayed

At the same time, user 112038 did not leave the bank, as predicted by the model with 92% chance. Figure 16 shows the contribution of the time predictions, unique MCC groups (the client only made transactions in one category) and customer age is negated by the transaction amount sum (more than 600000 in rubles) and unbiased variance, the number of employees in the client's company, and other factors.

Figure 17 depicts feature contributions for all test users, offering an alternative to permutation feature importance. There is a significant difference between the two importance measures: permutation feature importance is based on the decrease in model performance, whereas SHAP is based on magnitude of feature attributions. The main contributors are time predictions, employee count, customer age group, percent of non-active periods in days and number of unique days with transactions.

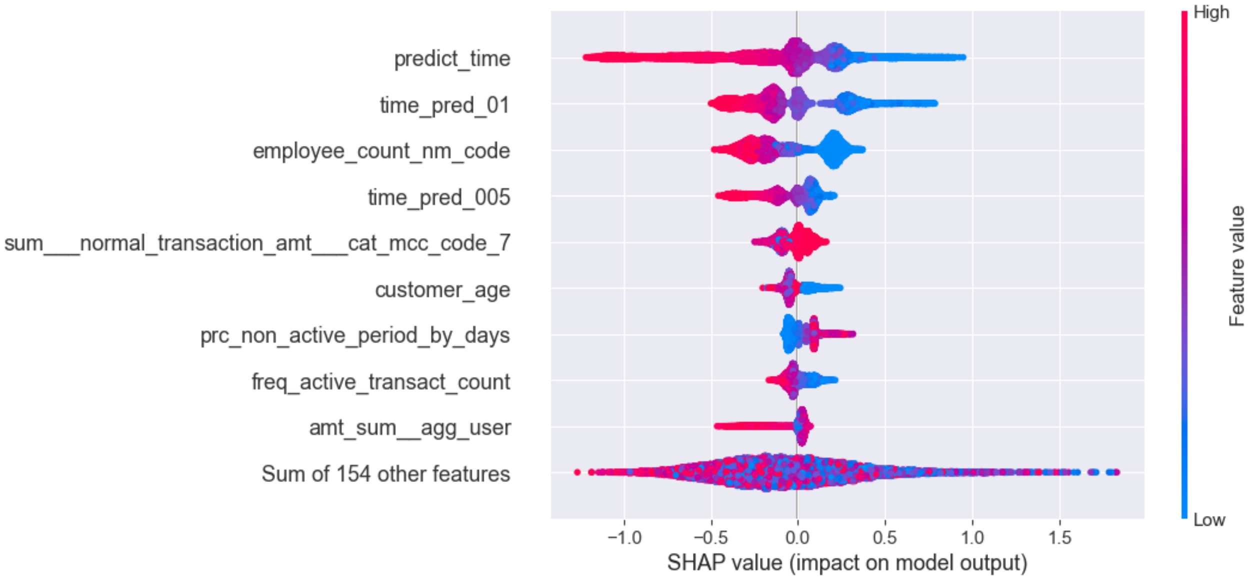


Figure 17. Overall feature contributions

4.11. Summary

This chapter outlined the comprehensive process of training machine learning models for predicting customer churn. By meticulously evaluating and optimizing various classifiers, we identified *XGBoost* and *CatBoost* as the best performers. The development of a stacked ensemble model, through careful hyperparameter tuning and model stacking, resulted in a robust predictive model. The final model ranks in the top 3 on public leaderboard and is the best open-source solution [3].

CHAPTER 5: CONCLUSION

This study successfully developed a robust machine learning algorithm to predict customer churn using transaction data from a six-month period, as part of the Data Fusion 2024 competition by VTB. By employing survival analysis and various machine learning techniques, the research addressed critical challenges such as data censoring and feature engineering.

Key processes included meticulous data preprocessing, innovative feature generation, and model optimization. Specifically, tracking changes in transaction flow rates over time proved vital for enhancing the predictive power of the models. The study demonstrated that accurate churn prediction is achievable through careful data handling and advanced modeling techniques.

The findings highlight that XGBoost and CatBoost were the top-performing models, with the final ensemble model delivering superior performance through careful hyperparameter tuning and model stacking. This approach not only improved predictive accuracy but also provided valuable insights for businesses aiming to implement effective customer retention strategies.

In conclusion, this research offers significant contributions to the field of customer churn prediction by demonstrating the efficacy of combining survival analysis with state-of-the-art machine learning methods. These insights can be leveraged by businesses to enhance their customer retention efforts, ultimately leading to better customer relationship management and reduced churn rates. Future research could further explore the integration of additional data sources and the application of emerging machine learning techniques to continue advancing this critical area of study.

REFERENCES

[1] Dalmia, H., Nikil, C.V.S.S. & Kumar, S. “Churning of Bank Customers Using Supervised Learning”. *Innovations in Electronics and Communication Engineering* (2020): 681-691. DOI: https://doi.org/10.1007/978-981-15-3172-9_64

[2] Keramati, A., Ghaneei, H. & Mirmohammadi, S.M. “Developing a prediction model for customer churn from electronic banking services using data mining”. Financ Innov 2, 10 (2016): 1-13. DOI: https://doi.org/10.1186/s40854-016-0029-6

[3] Ods.ai, “Data Fusion Contest 2024 - Task 2 ‘Churn’”. Website. URL: https://ods.ai/competitions/data-fusion2024-churn

[4] E. L. Kaplan & Paul Meier. “Nonparametric Estimation from Incomplete Observation”. Journal of the American Statistical Association, Vol. 53, No. 282 (1958): 457- 481. DOI: https://doi.org/10.1080/01621459.1958.10501452

[5] Cox R David. “Regression models and life tables”. Journal of the Royal Statistical Society 34, 2 (1972): 187–220. DOI: https://doi.org/10.1111/j.2517-6161.1972.tb00899.x

[6] Ping Wang, Yan Li & Chandan k. Reddy. 2019. “Machine Learning for Survival Analysis: A Survey”. ACM Comput. Surv. 51, 6, Article 110 (2019): 1-36. DOI: https://doi.org/10.1145/3214306

[7] John P Klein & Melvin L Moeschberger. “Survival analysis: techniques for censored and truncated data”. Springer Science & Business Media (2005): 63-90. DOI: https://doi.org/10.1007/b97377

[8] Habr, “Time is the relation of being to non-being. A few words about Time-to-event analysis”. Website. URL: https://habr.com/ru/articles/795191/

[9] Hazra A, Gogtay N & Indian J Dermatol. “Biostatistics Series Module 9: Survival Analysis” (2017): 251-257. DOI: https://doi.org/10.4103/ijd.ijd_201_17

[10] Lucijanic M & Petrovecki M. “Analysis of censored data”. Biochem Med (2012): 151-155. DOI: https://doi.org/10.11613/bm.2012.018

[11] Wayne Nelson. “Theory and applications of hazard plotting for censored failure data”. Technometrics 14, 4 (1972): 945–966. DOI: http://dx.doi.org/10.1080/00401706.2000.10485975

[12] Hao H. Zhang & Wenbin Lu. “Adaptive Lasso for Cox’s proportional hazards model”. Biometrika 94, 3 (2007): 691–703. DOI: http://dx.doi.org/10.1093/biomet/asm037

[13] Binder, H. & Schumacher, M. “Allowing for mandatory covariates in boosting estimation of sparse high-dimensional survival models”. BMC Bioinformatics 9, 14 (2008): 1-10. DOI: https://doi.org/10.1186/1471-2105-9-14

[14] David G. Kleinbaum & Mitchel Klein. “Survival Analysis: A Self-learning Text, Third Edition”. Springer Science & Business Media (2012): 1-524. DOI: https://doi.org/10.1007/978-1-4419-6646-9

[15] Paul D Allison. 2010. “Survival Analysis Using SAS: A Practical Guide. Second Edition”. Sas Institute (2010): 1-324. DOI: https://doi.org/10.1093/aje/kwr202

[16] James Tobin. “Estimation of relationships for limited dependent variables”. Econometrica: journal of the Econometric Society 26, 1 (1958): 24–36. DOI: https://doi.org/10.2307/1907382

[17] Jonathan Buckley & Ian James. “Linear regression with censored data”. Biometrika 66, 3 (1979): 429–436. DOI: https://doi.org/10.2307/2335161

[18] Minjung Kyung, Jeff Gill, Malay Ghosh & George Casella. “Penalized regression, standard errors, and Bayesian lassos”. Bayesian Analysis 5, 2 (2010): 369–411. DOI: http://dx.doi.org/10.1214/10-BA607

[19] Zupan B, Demsar J, Kattan MW, Beck JR & Bratko I. “Machine learning for survival analysis: a case study on recurrence of prostate cancer”. Artif Intell Med. (2000): 59-75. DOI: https://doi.org/10.1016/s0933-3657(00)00053-1

[20] Loh, Wei-Yin. “Classification and Regression Trees”. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery (2011): 14-23. DOI: http://dx.doi.org/10.1002/widm.8

[21] Hothorn, Torsten, Lausen, Berthold, Lausen, Berthold & Radespiel-Tröger, Martin. “Bagging survival trees” Statistics in medicine 23 (2004): 77-91. DOI: http://dx.doi.org/10.1002/sim.1593

[22] Hemant Ishwaran, Udaya B. Kogalur, Eugene H. Blackstone & Michael S. Lauer. “Random survival forests” Annals of Applied Statistics (2008): 841-860. DOI: https://doi.org/10.48550/arXiv.0811.1645

[23] Zhiyuan He, Danchen Lin, Thomas Lau & Mike Wu. “Gradient Boosting Machine: A Survey” Published online (2019): 1-9. DOI: https://doi.org/10.48550/arXiv.1908.06951

[24] Bellazzi R & Zupan B. “Predictive data mining in clinical medicine: current issues and guidelines”. International journal of medical informatics 77, 2 (2008), 81–97. DOI: https://doi.org/10.1016/j.ijmedinf.2006.11.006

[25] M. J. Fard, P. Wang, S. Chawla & C. K. Reddy. “A Bayesian Perspective on Early Stage Event Prediction in Longitudinal Data”. IEEE Transactions on Knowledge and Data Engineering (2016): 3126-3139. DOI: https://doi.org/10.1109/TKDE.2016.2608347

[26] Friedman, N., Geiger, D. & Goldszmidt, M. “Bayesian Network Classifiers”. Machine Learning 29 (1997): 131–163. https://doi.org/10.1023/A:1007465528199

[27] Jared L. Katzman, Uri Shaham, Alexander Cloninger, Jonathan Bates, Tingting Jiang & Yuval Kluger. “DeepSurv: personalized treatment recommender system

using a Cox proportional hazards deep neural network" BMC Medical Research Methodology (2018): 1-15. DOI: https://doi.org/10.48550/arXiv.1606.00931

[28] Martinsson, Egil. "WTTE-RNN: Weibull Time To Event Recurrent Neural Network". Master's Thesis (2017): 1-103. URL: https://publications.lib.chalmers.se/records/fulltext/253611/253611.pdf

[29] Giunchiglia, E. & Nemchenko, A., van der Schaar, M. (2018). "RNN-SURV: A Deep Recurrent Model for Survival Analysis". Artificial Neural Networks and Machine Learning – ICANN (2018): 23-32. DOI: https://doi.org/10.1007/978-3-030-01424-7_3

[30] Patricia A. Apellániz, Juan Parras & Santiago Zazo. "SAVAE: Leveraging the variational Bayes autoencoder for survival analysis". Machine Learning (2023): 1-14. DOI: https://doi.org/10.48550/arXiv.2312.14651

[31] Tamara Fernández, Nicolás Rivera & Yee Whye The. "Gaussian Processes for Survival Analysis". The Neural Information Processing Systems conference (2016): 1-13. DOI: https://doi.org/10.48550/arXiv.1611.00817

[32] Bank of Russia, "Dynamics of the official exchange rates". Website. URL: https://www.cbr.ru/eng/currency_base/dynamics/

[33] Forbes, "Demand for the yuan in Russian banks has increased significantly: who is buying it and how". Website. URL: https://www.forbes.ru/investicii/480428-spros-na-uan-v-rossijskih-bankah-vyros-v-razy-kto-i-kak-ego-pokupaet

[34] Rambler, "Yuan became the most traded currency in Russia". Website. URL: https://finance.rambler.ru/markets/50434944-yuan-stal-samoy-torguemoy-valyutoy-v-rossii/

[35] Rosstat (Federal State Statistics Service), "Prices, inlfation". Website. URL: https://rosstat.gov.ru/statistics/price

[36] Sklearn, “Ensemble-based methods for classification, regression and anomaly detection.”. Website. URL: https://scikit-learn.org/stable/api/sklearn.ensemble.html

[37] Imbalanced learn, “EasyEnsembleClassifier”. Website. URL: https://imbalanced-learn.org/stable/references/generated/imblearn.ensemble.EasyEnsembleClassifier.html

[38] Tianqi Chen & Carlos Guestrin, “XGBoost: A Scalable Tree Boosting System”. The 22nd ACM SIGKDD International Conference (2016): 1-13. DOI: http://dx.doi.org/10.1145/2939672.2939785

[39] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush & Andrey Gulin, “CatBoost: unbiased boosting with categorical features”. NeurIPS Conference (2018): 1-23. DOI: https://doi.org/10.48550/arXiv.1706.09516

[40] Optuna, “Optuna - A hyperparameter optimization framework”. Website. URL: https://optuna.org/

[41] SHAP, “SHAP Documentation”. Website. URL: https://shap.readthedocs.io/en/latest/

[42] GitHub, Open-source solution “MDS-Churn_Sharipov – Customer churn prediction, Data Fusion Contest 2024”. Website. URL: https://github.com/BeteLgis/MDS-Churn_Sharipov