BACHELOR'S THESIS / ENGINEERING 2024

Advanced Analytics AI Applications in Digital Experience

Creating a churn solution for a multinational company

Martina Tarrés Castellanos

upf. Universitat Pompeu Fabra Barcelona

Grau en Enginyeria Matemàtica en Ciència de Dades
Universitat Pompeu Fabra

Advanced Analytics AI Applications in Digital Experience

Creating a churn solution for a multinational company

TREBALL DE FI DE GRAU DE

Martina Tarrés Castellanos

Supervisor: Miguel Àngel Cordobés Aranda
Co-Supervisor: Anna Tormey

Barcelona, July 2024

A la gent que estimo.

Acknowledgement

Firstly, I would like to express my deep gratitude to Dynatrace for making this project possible. I am sincerely thankful to the Digital Experience team for providing me with the opportunity to work on a real-world problem and for their invaluable feedback and support throughout.

Special thanks go to my work colleagues. It has been a pleasure to share so many moments with them, learning from each one of them. I am truly grateful for their assistance during times when I felt lost, unsure about my path in life. They helped me realize that life is short, time flies by, and it's important to enjoy every stage of it. Their support during these months of personal and professional growth means a lot to me, and they will always have a special place in my heart.

Additionally, I would like to extend my deepest appreciation to my thesis tutor, Miguel Ángel. His guidance has been fundamental throughout every step of this project. I am incredibly grateful for his calmness and for giving me the freedom to build my own thesis.

Finally, I want to say thanks to my family for the unconditional love and support they gave me.

Abstract

This study explores the development and implementation of a churn prediction solution for Dynatrace, a software monitoring platform. By using historical customer data as well as relevant features, this project aims to build an effective predictive model capable of identifying customers at risk of churn. This solution also includes a component of Explainability AI, where we analyze and interpret the model's predictions to provide insights into the factors contributing to customer churn. For a better understanding of the data, an interactive dashboard was also built, allowing team members to explore the model's predictions and gain actionable insights.

Resumen

Este proyecto explora el desarrollo e implementación de una solución de predicción de *churn* para Dynatrace, una plataforma de software. Utilizando datos históricos de clientes y otras características relevantes, este proyecto tiene como objetivo construir un modelo predictivo eficaz, capaz de identificar a los clientes en riesgo de *churn*. Esta solución también incluye un componente de *Explainability AI*, donde analizamos e interpretamos las predicciones del modelo para proporcionar información sobre los clientes. Para una mejor comprensión de los datos, también se construirá un *Dashboard* interactivo, que permitirá a los miembros del equipo explorar las predicciones del modelo.

Resum

Aquest projecte explora el desenvolupament i implementació d'una solució de predicció de churn per a Dynatrace, una plataforma de *Software*. Utilitzant dades històriques de clients i altres característiques rellevants, aquest projecte té com a objectiu construir un model predictiu eficaç, capaç d'identificar els clients en risc de churn.

Aquesta solució també inclou un component d'*Explainability AI*, on analitzem i interpretem les prediccions del model per proporcionar informació sobre els clients. Per a una millor comprensió de les dades, també es construirà un *Dashboard* interactiu, que permetrà als membres de l'equip explorar les prediccions del model.

Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

Nowadays, businesses face increasing challenges in retaining customers. Customer churn, or the loss of customers, presents significant implications for revenue and business sustainability. To mitigate the impact of churn, predictive modeling techniques offer powerful tools to forecast which customers are likely to churn, enabling proactive retention strategies.

This thesis explores the development and implementation of a churn prediction solution using machine learning techniques for Dynatrace, a software monitoring platform. By using historical customer data and relevant features, this study aims to build an effective predictive model capable of identifying customers at risk of churn. Additionally, this solution incorporates Explainability AI to analyze and interpret the model's predictions, providing insights into the factors contributing to customer churn. To enhance usability, an interactive dashboard will be developed, allowing team members to visualize the predictions and understand the underlying reasons for churn.

1.1 Background

1.1.1 What is Dynatrace

Dynatrace is a software monitoring platform that has several functionalities (see Figure 1). It is mainly composed by two key concepts: OneAgent and Davis ®. OneAgent is a single agent that automatically discovers, instruments, and gathers monitoring data of IT environments [1]. On the other hand, Davis ® is Dynatrace's own Artificial Intelligence that analyses the dependencies in complex IT environments and provides information about the root cause of problems and their business impact.

Figure 1: Dynatrace Funcionalities

1.1.2 Company departments involved in this project

This project focuses on one specific solution offered by Dynatrace: Digital Experience Monitoring (DEM). This solution aims to monitor and optimize the digital experience of the users. Within a solution, there are various capabilities corresponding to different products developed by Dynatrace. Digital Experience Monitoring primarily consists of Real User Monitoring (for both Web and Mobile applications), Session Replay, and Synthetics Monitoring. [2]

The key concepts of each capability are the following:

- Real User Monitoring (RUM): it focuses on tracking and analyzing the behavior and performance of real users interacting with web and mobile applications.

- Session Replay: it allows playback of user sessions, providing insights into user interactions and behaviors on the application.

- Synthetics Monitoring: it involves simulating user interactions (such as website visits or transactions) to proactively monitor application performance and detect issues before real users are affected.

1.2 Objectives

The main objective of this project is to develop and implement an effective churn prediction solution within the Dynatrace software monitoring platform. Therefore, the objectives of this work are:

1. Familiarize with Machine Learning for Churn Prediction.
2. Analyse Dynatrace database to preprocess the appropriate datasets.
3. Analyse the data using visualization techniques.
4. Implement and evaluate different machine learning and deep learning algorithms to identify the most effective model for predicting customer churn.
5. Assess and compare the performance metrics of the developed churn prediction models.
6. Visualize feature importance to understand key drivers of customer churn.
7. Create a Dashboard Interface that has the most accurate ML model with the intention to share the results with the team.

Chapter 2

INTRODUCTION TO ADVANCED ANALYTICS

In this chapter, we will explore three main categories of models used in advanced analytics: Machine Learning (ML) models, Deep Learning (DL) models, and Statistical models.

2.1 Statistic Models

In addition to ML and DL models, we plan to incorporate Statistical models. Common statistical models are [3]:

- **Linear Regression:** This method models the relationship between a dependent variable and one or more independent variables by fitting a linear equation to observed data.
- **Logistic Regression:** Used for binary classification problems, this model estimates the probability of an outcome based on predictor variables.
- **Survival Analysis:** Techniques used to analyze and predict the time until an event of interest.

2.2 Traditional Machine Learning

ML models are algorithms trained on some data to identify patterns or correlations, enabling them to generate predictions and making it possible to classify new data. We can find two types: Supervised Learning (SL) and Unsupervised Learning. The main difference between these two categories is on the training data used. SL models are trained using labeled samples to make predictions, whereas Unsupervised Learning models work with unlabeled data to uncover patterns or clusters in the data.

In our study, we have a customer database with various attributes, and our objective is to predict whether a customer will churn or not, along with estimating the probability of churn. This will be based on historical data of previous churners and current customers. Therefore, we will focus on developing Supervised Learning models.

For finding patterns in our data, we will develop a classification algorithm, where different classifiers assign test data into specific categories. We have explored different techniques to identify which model performs better:

- Extreme Gradient Boosting (XGBOOST)
- Random Forest (RF) Classifier
- Light GBM
- Gradient Boosting

2.3 Deep Learning

To gain a more comprehensive understanding of the project, we will also apply Deep Learning models, which is a subset of Machine Learning. It uses neural networks with multiple layers to model complex patterns [4].
Key types of DL models include:

- **Fully connected neural network (FC):** These are composed of interconnected nodes or neurones and are used for tasks such as image and speech recognition.
- **Convolutional Neural Networks (CNNs):** These are designed for processing structured grid data like images, using convolutional layers to learn spatial hierarchies of features.
- **Recurrent Neural Networks (RNNs):** These are used for sequential data. They are particularly good at understanding the context of a sentence or phrase, and they can be used to generate text or translate languages.

To summarize, we will compare the results produced by all models and choose the best performing one [1].

[1] See Appendix for the explanation of the algorithms.

Chapter 3

IMPLEMENTATION AND DESIGN

3.1 MLOPS Framework

In this section, we describe the whole end-to-end process focused on developing a Churn Prediction Model that is usually approached in a real world company. The process followed for the project is:

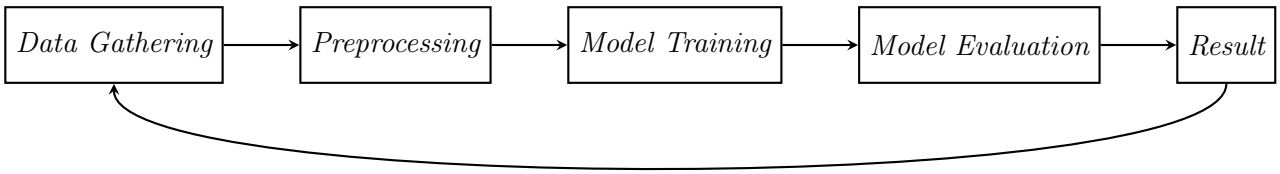


Figure 2: MLOPS Framework Outline

1. **Data Gatheting:** Dynatrace provided the whole database of the company. One of the first steps was to identify all relevant data and create the necessary queries to gather the main information.

2. **Preprocessing:** After collecting the data, it was cleaned, formatted, and transformed to a form that could be used for the ML model. This involved tasks like addressing missing values and removing duplicate entries to ensure our dataset was well-prepped and ready for the analysis, as well as creating some additional columns.

3. **Model Selection and Training:** Next, different types of ML algorithms were applied to find the better algorithm that fits to our data. This step involved exploring various algorithms, tuning hyperparameters, and evaluating model performance to identify the best approach for our churn prediction task.

4. **Model Evaluation:** Following the training of various models, we conducted an evaluation to assess their performance. We examined metrics such as accuracy, precision, recall, F1 score, Root Mean Squared Error. Additionally, we employed Shapley Additive Explanations (SHAP) values to enhance our understanding of the ML algorithms' interpretability.

5. **Result:** The final step is to create a Dashboard interface the visualize the results. This interface will provide insights into which customers are at risk of churning as well as some statistics about their customer's behaviours.

3.2 The Data

3.2.1 Data Preprocessing

One of the main objectives of the study establish an automated process for detecting churn customers. For this purpose, we have developed a Python script that interfaces with Dynatrace's database in Snowflake. This script will also be integrated with a Business Intelligence (BI) tool, Power BI in our case. Figure 3 provides an overview of the system architecture.

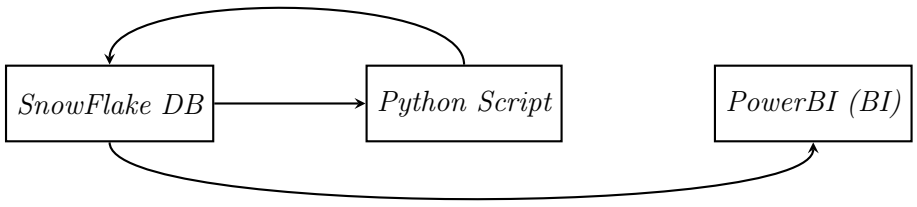


Figure 3: Framework Outline

As a multinational corporation, Dynatrace's data model is complex, composed by many tables and data sources. So after identifying relevant information, the first step was to create a relational model (Figure 4).

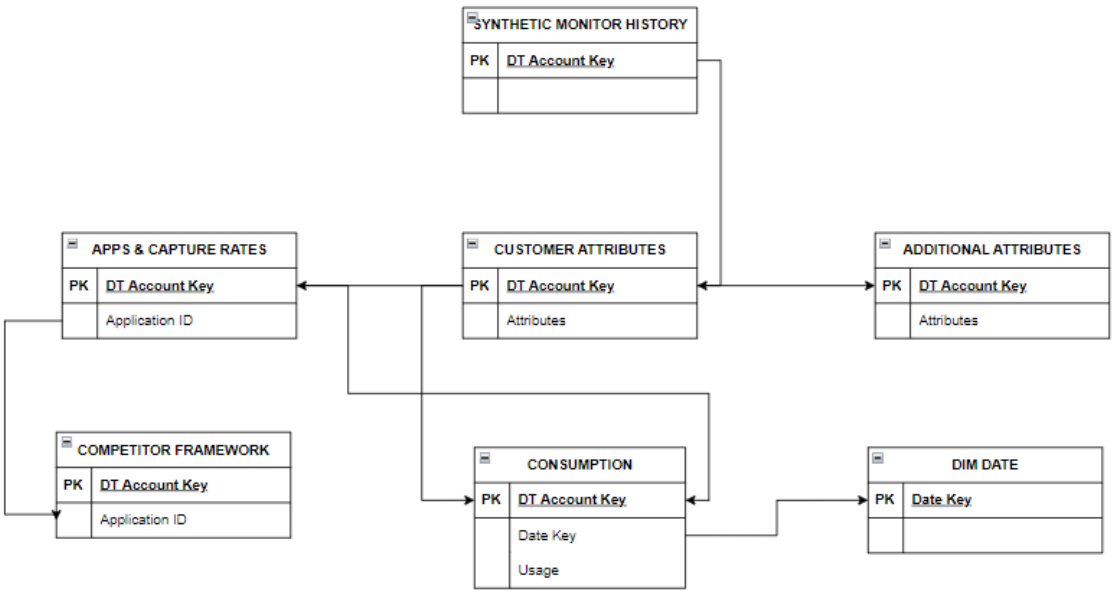


Figure 4: Simplified Version of the Relational Model

The dataset includes usage and features of Dynatrace DEM customers over the last three and a half years. In total we have a record of 4942 clients and 24 different attributes.

Each client has the following attributes:

- **DT Account Key:** unique identifier associated to a Dynatrace Account.
- **SFDC Account ID:** unique identifier of a client associated to a SalesForce account. Each SFDC Account ID can be associated to several DT Accounts.
- **Date:** Date of the usage, in format YYYY-MM
- **Deployment Type:** Type of Dynatrace product deployment (e.g., SaaS, Managed)
- **ARR Band:** annual Recurring Revenue on an SFDC account level (e.g., 1M+, 500K-999K, 250K-499K, 100K-249K, 50K-99K, 20K-49K, Under 10K)

- **Geolocation:** account geography e.g., NORAM, EMEA, LATAM
- **Country:** account country e.g., Spain, Italy
- **Vertical:** industry associated with an SFDC account such as "Automotive"
- **DEM Units:** total DEM Units consumed by a customer on a monthly basis. We have five different types of DEM Units, corresponding to each capability (e.g., DEM Units - Web, DEM Units - Mobile, DEM Units - Session Replay, DEM Units - Synthetics, DEM Units - Total)
- **Applications:** number of billed apps. We can distinguish Web Apps, Mobile Apps and Session Replay Apps.
- **Sessions:** number of billed sessions. We can distinguish Web Sessions, Mobile Sessions and Session Replay Sessions.
- **Cost Control Percentage:** percentage of user action and user session captured by Dynatrace.
- **Synthetic Tests:** tests executed from various locations around the world at regular intervals to check the availability and performance of the monitored services, simulating user interactions.
- **HTTP Checks:** check that focus on validating the response of an HTTP requests to ensure that web services are functioning correctly.
- **Competitor Framework:** Number of competitor frameworks that a customer is using.

3.2.2 Data Cleaning

To do the Exploratory Data Analysis (EDA) and the modelling part of the project, a data cleansing was applied to our initial data. The modifications done in our data included:

1. Delete missing values of the data, specificly those SFDC Accounts that are null.

2. Replace with a 0 null values of usage columns such as apps, sessions and tests.

3. As each SFDC Account can be associated to multiple DT Accounts, DT Account Key column was replace by a Concatenation of all DT Accounts associated to the same SFDC Account using a |.

4. A Month-over-Month (MoM) column was added for each capability.

5. Flagged Has Capability column was also added in order to register which capability was used by each customer.

6. In addition, a retention month column was added, counting consecutive months of usage for each customer.

7. In order to know how many account churned each month, a Is Lost Account column was added. Lost accounts are only going to be based on usage, not contract information. An account is considered to be lost if it satisfies any of the above conditions:

   (a) **Flagged with no usage:** The account has zero usage flagged during the current month.

   (b) **Previous Month's Usage with Current Month's Inactivity:** The account had usage in the previous month but has no activity in the current month.

   (c) **The last register of data is not the current date:** The last recorded data for the account does not correspond to the current date being analyzed.

8. Moreover, an encoding was used to convert categorical columns into numerical data. This process was used to improve the precision of the model as well as interpret the model. The encoding used was One-Hot-Encoder.

3.3 Exploratory Data Analysis

To have an overview and a better understanding of the distribution of the data, Python and PowerBI were used. PowerBi is a BI tool that will help us visualize and analyze the data as well as create interactive dashboards that will help us to provide usefull insights.

In the first figure, we can observe the churn rates for different products, represented by the percentage of churners vs. non-churners. It is quite evident that Session Replay, (represented in blue), presents the highest churn rate among all the other capabilities. Additionally, we can also observed that over the past three years, the churn rate has remained relatively stable, showing neither a significant increase nor decrease.The average churn rate of all products is around 5%, with DEM being the lowest in 2% and as we commented, Session Replay having the highest one, around 8%.

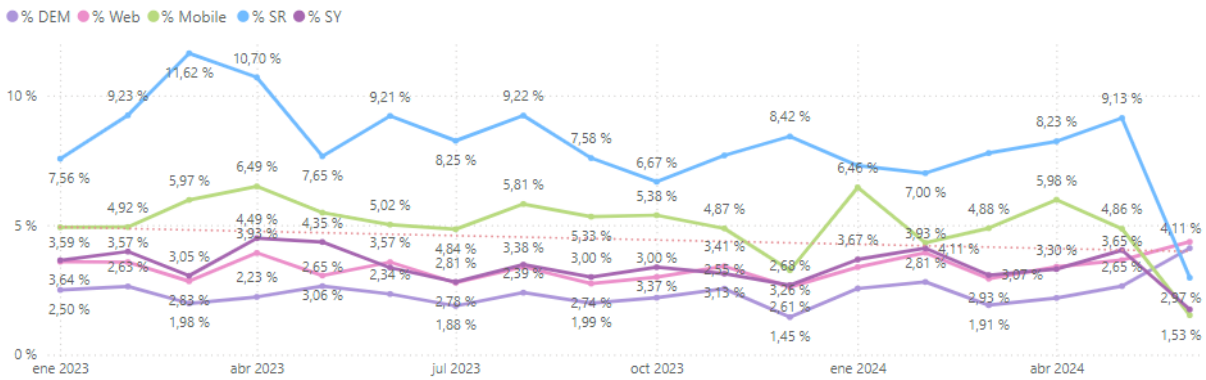


Figure 5: Lost vs all customers (%)

In figure 6 we see the geographic distribution of overall DEM customers. Darker shades of blue indicate a higher number of accounts, while lighter shades indicate the opposite. It is evident that the United States, followed by Brazil, France and Canada, are the countries with the highest number of accounts. Now, let us investigate whether these countries with more accounts also experience higher churn rates.

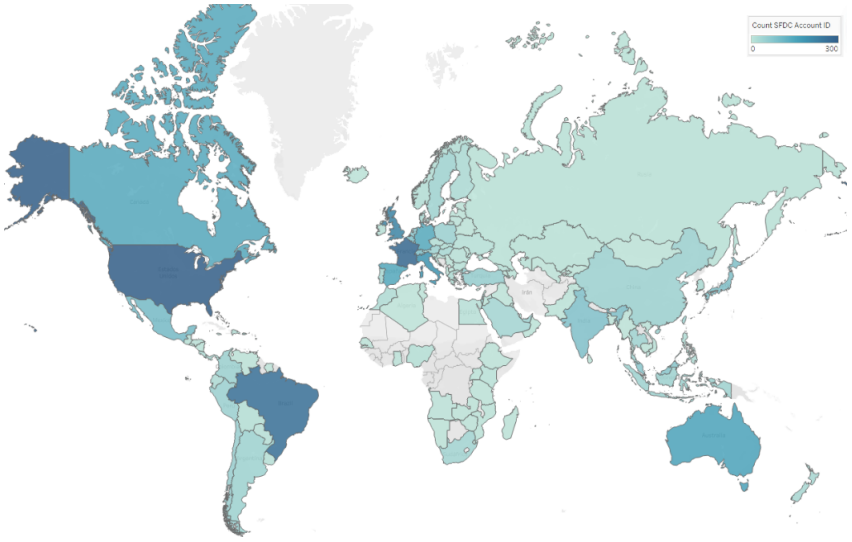


Figure 6: Geographic distribution of DEM customers

Here we present the distribution of lost accounts for each capability based on their geographical location.

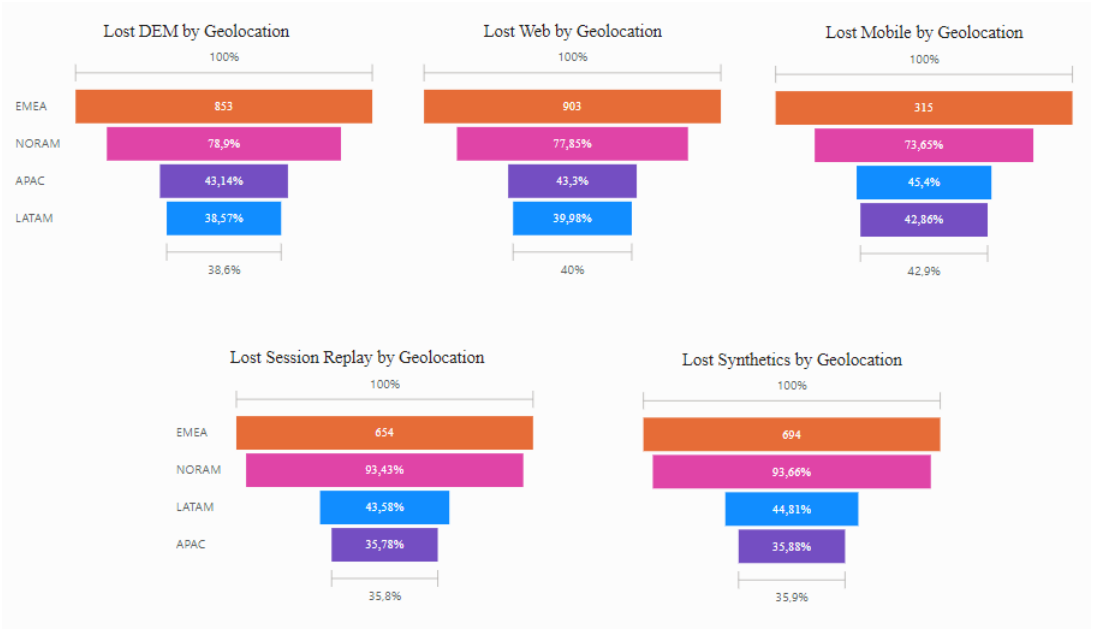


Figure 7: Lost Accounts by Geolocation

EMEA (Europe, Middle East, and Africa), followed by NORAM (North America), account for the majority of zones with lost accounts.

Let us delve deeper and analyze the distribution by countries. Across all capabilities, United States represents almost 50% of the overall lost accounts. This information

can be contextualized, considering that the United States had the highest number of accounts, as evident in the previous figure 6. Brazil and France represents both 10% of lost customers.

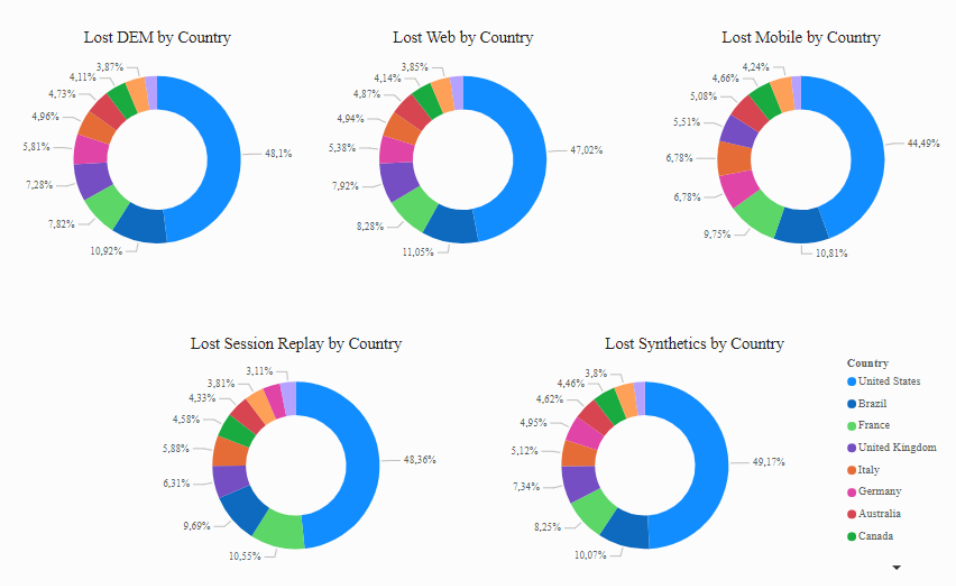


Figure 8: Lost Accounts by Country

To conclude, let us analyze the industry of the lost accounts. Both *Software* and *Banking* represents a 20% each, followed by *Retail* and *Consulting and Technology* representing around 10% each.

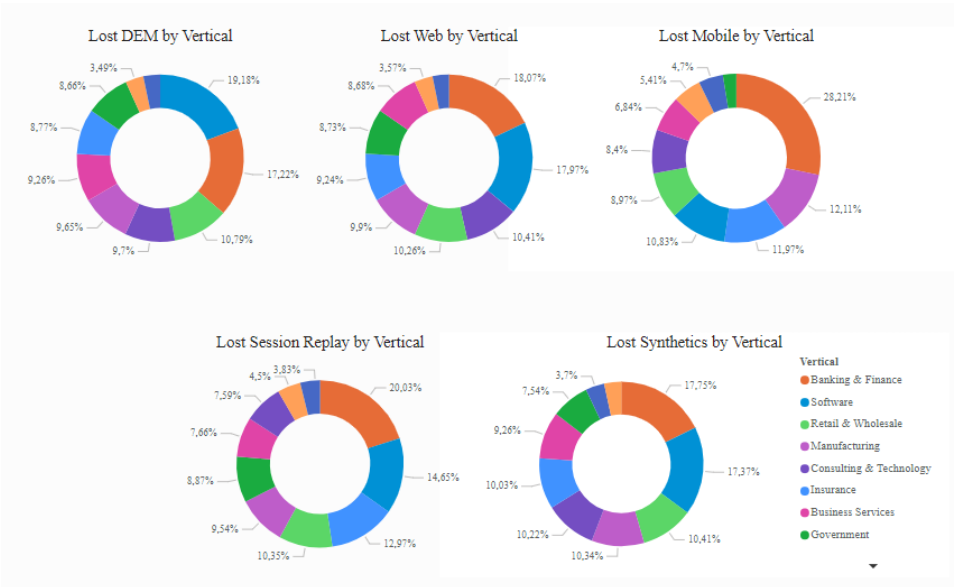


Figure 9: Lost Accounts by Vertical

Chapter 4

MODELING

After having had our data explored, certain characteristics of our data have been observed, so let us now move on the modeling and prediction part of the study.

This approach followed isn this thesis moves from statistical techniques to more complex as DL. We will start with simpler models and gradually move to more complex ones. The models we will implement include a traditional machine learning model, a statistical non-parametric model for survival analysis, and a deep learning model with embedding. We will analyze the performance of each model, and the best-performing one will be used for prediction and featured in the dashboard.

As we have mentioned at the beginning, Digital Experience solution is composed of five different capabilities. Each one of these capabilities is affected by different churn signals. Hence, we will need to implement five different models, one for each product. In this section, we will focus on explaining the model for DEM in detail. The results of the other models are summarized in the Chapter 5

4.1 Model 1 - Statistical non-parametric with survival analysis

In this section, we shift our focus to survival analysis, a statistical approach well-suited for predicting time-to-event data. Unlike traditional machine learning models that primarily predict binary outcomes, survival analysis allows us to model the time until a particular event occurs, in this case, customer churn. This method provides a deeper understanding of not only whether a customer will churn, but also when this churn is likely to happen [5].

4.1.1 Data Preparation

For our survival analysis model, we will prepare the data to include the duration of customer engagement and the event indicator (whether the customer has churned or not). Hence, the key variables for this model are:

- **SFDC Account ID:** unique client identifier.
- **Duration:** consecutive retention months of usage.
- **Event Indicator:** A binary variable indicating whether the event (churn) has occurred (1) or not (0).
- **Predictor Variables:** Various features that could potentially influence the time to churn, such as usage, customer demographics and other attributes.

These evaluations will help us understand how well our survival models predict the timing of churn.

4.1.2 Kaplan-Meier Survival Analysis Results

The Kaplan-Meier survival curve in Figure 10 illustrates the estimated survival probabilities over a period of 42 months for our customer cohort. By analyzing this

curve, we can identify periods of heightened churn risk and strategize accordingly to enhance customer retention efforts.

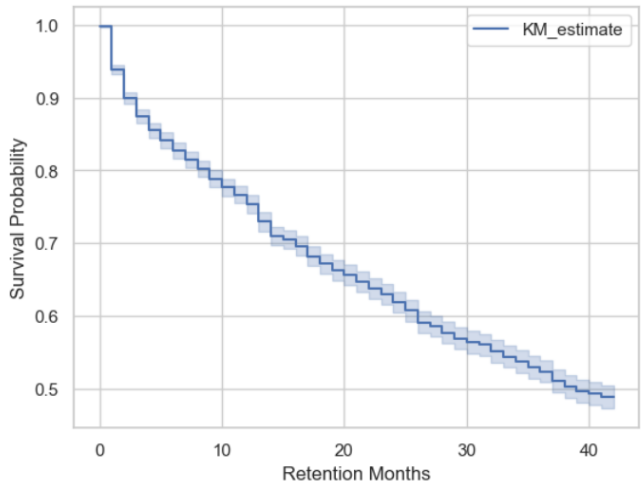


Figure 10: Kaplan-Meier survival curve

There is a noticeable decline in the survival probability within the first 10 months. This indicates a significant portion of customers churn during the early stages of their engagement. Specifically, by Month 10, the survival probability has dropped to approximately 0.75, meaning around 25% of customers have churned. Beyond 30 months, the survival probability stabilizes somewhat, but there is still a continued gradual decline. By Month 40, the survival probability is around 0.50, indicating that about half of the customers are still retained after 40 months.

4.1.3 Implementation Results

Following a similar approach to our first model, the data was split into training and testing data using a split size of 0.8 for training and 0.2 for testing. We implemented three different survival analysis models:

- Random Survival Forest
- Gradient Boosting Survival
- Survival Tree

The implementation of these models and their evaluation through the Kaplan-Meier curve and Concordance Index provides a comprehensive understanding of customer churn dynamics.

Model	Score
Random Survival Forest	0.915
Gradient Boosting Survival	0.984
Survival Tree	0.879

Table 1: Scores for different survival models

When we use a survival model, the output consists of the predicted probability for each customer to survive at each retention month. Thanks to this, we are able to construct the survival curve for each customer.

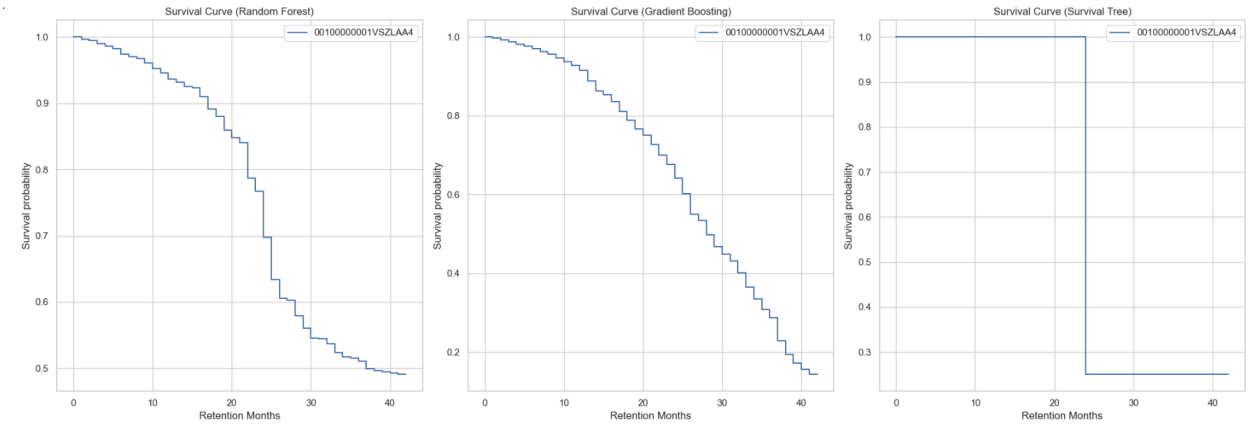
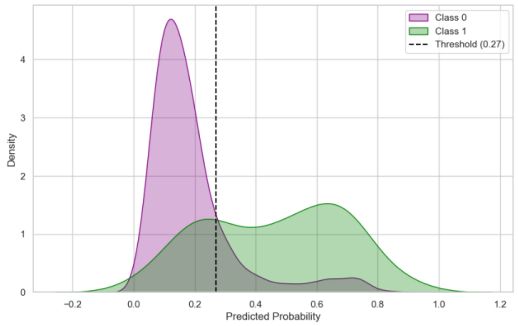


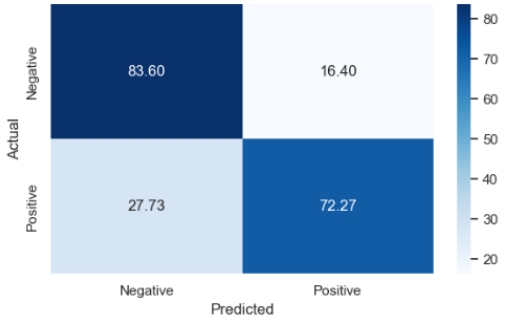
Figure 11: Kaplan-Meier Curve for a given customer

The survival curves for customer ID: 00100000001VSZLAA4 from three models reveal different patterns. The Random Forest model shows a gradual decline in survival probability over 40 months, eventually dropping to around 50%. Similarly, the Gradient Boosting model indicates a steady decrease in survival probability, also reaching approximately 50% at the end of the 40-month period, which coincides with the Kaplan-Meier curve of the overall dataset.

Following the same approach as before, we will use the last ended month (March 2024) to test the performance of the model. Let us look at the evaluation metrics:

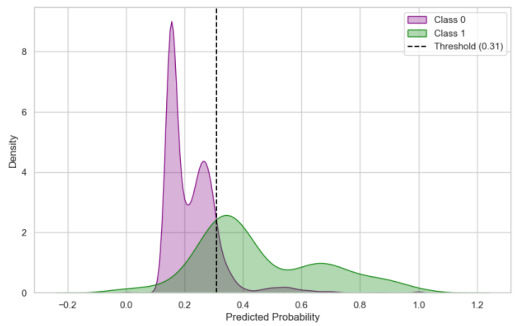


(a) Density Chart

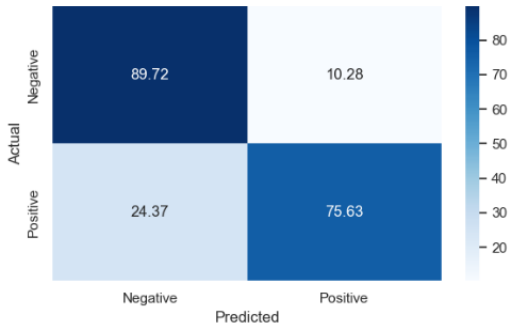


(b) Confussion Matrix

Figure 12: Evaluation of Random Survival Forest



(a) Density Chart



(b) Confussion Matrix

Figure 13: Evaluation of Gradient Boosting Survival

When comparing the Random Survival Forest and Gradient Boosting Survival models, the last model demonstrates superior performance. The density charts shown in Figure 12 indicate that the Gradient Boosting Survival model achieves better separation between classes, with less overlap around the decision threshold. This visual distinction is supported by the confusion matrices (see Figure 13) and performance metrics summarized in Table 2: *Performance Metrics for Random Survival Forest and Gradient Boosting Survival on Testing Set.* According to the table, Gradient Boosting Survival achieves higher precision (0.2174 vs. 0.1426), recall (0.7563 vs. 0.7227), accuracy (0.8921 vs. 0.8319), and F1 score (0.3377 vs. 0.2382). These metrics indicate that the Gradient Boosting Survival model is more effective at correctly identifying true positives and minimizing false positives, resulting in overall more accurate and reliable predictions.

Metrics	Random Survival Forest	Gradient Boosting Survival
Precision	0.1426	0.2174
Recall	0.7227	0.7563
Accuracy	0.8319	0.8921
F1 Score	0.2382	0.3377

Table 2: Performance Metrics on Testing Set

4.2 Model 2 - Traditional Machine Learning

For the first model, a DataFrame for each capability was created, containing a sliding window of three months and the Month over Month growth metric. This sliding window approach enables us to capture historical patterns and trends in customer behavior over time, providing insights into potential churn risk factors [6].

Customer ID	Date	Usage	Attributes	Lost Account
1	ene-24	50	...	False
1	feb-24	75	...	False
1	mar-24	70	...	False
1	abr-24	40	...	False
1	may-24	0	...	True

Table 3: Customer Usage Data

Customer ID	Predicting Month	MoM (1-2)	MoM (2-3)	MoM (1-3)	Attributes	Target
1	abr-24	50%	-20%	-6.67%	...	False
1	may-24	-6.67%	-42.86%	-46.67%	...	True

Table 4: Sliding window DataFrame transformed

In Table 3, we can see the data before the transformation while in Table 4 we have the new DataFrame, containing the sliding window with three months of historical customer's data.

We will use *Pycaret*, a python open-source library that aims to simplify the process of evaluating and deploying machine learning models.

By looking at Table 5 we will start by applying Random Forest Classifier, Extreme Gradient Boosting and LightGBM, as the overall Accuracy and Recall is higher. By looking at the evaluation metrics of each model we will finally decide which one performs best.

Model	Accuracy	Recall	Precision	F1
rf	0.9905	0.4262	0.9602	0.5890
lightgbm	0.9904	0.4252	0.9492	0.5861
xgboost	0.9902	0.4271	0.9145	0.5812
et	0.9877	0.3539	0.7487	0.4798
gbc	0.9851	0.4551	0.5501	0.4962
dt	0.9780	0.4725	0.3587	0.4070
ada	0.9679	0.5284	0.2604	0.3468
knn	0.8386	0.3529	0.0361	0.0655

Table 5: Performance comparison of different classifiers for DEM

The dataset was split into training and testing data using a split size of 0.8 for training and 0.2 for testing. To evaluate the performance of the models, the last month of the training data was excluded and reserved for testing. This approach allows us to assess how well the models perform on unseen data.

In order to get a graphical representation of our model, we will plot a density chart, so we can observed the distribution of our data. This is particularly useful for understanding the probability of different outcomes within a dataset and helps in identifying patterns, skewness, and the presence of multiple modes (peaks) in our data. By comparing the density plots of both classes, we can gain insights into how well our model distinguishes between customers who are likely to churn and those who are not.

The threshold in a binary classification model is the probability cutoff that determines the class label assigned to a given prediction. For instance, in a churn prediction model, if the predicted probability of a customer churning is greater than the threshold, the customer is classified as a "churner"; otherwise, they are classified as a "non-churner". Adjusting this threshold can significantly impact the model's performance metrics, such as precision, recall, and accuracy. Choosing an appropriate threshold is crucial because it directly influences the balance between false positives (incorrectly predicting churn when the customer does not churn) and false negatives (failing to predict churn when the customer does churn). As we can observed in Figure 14, the he selected threshold that maximizes the model's performance is 0.4.

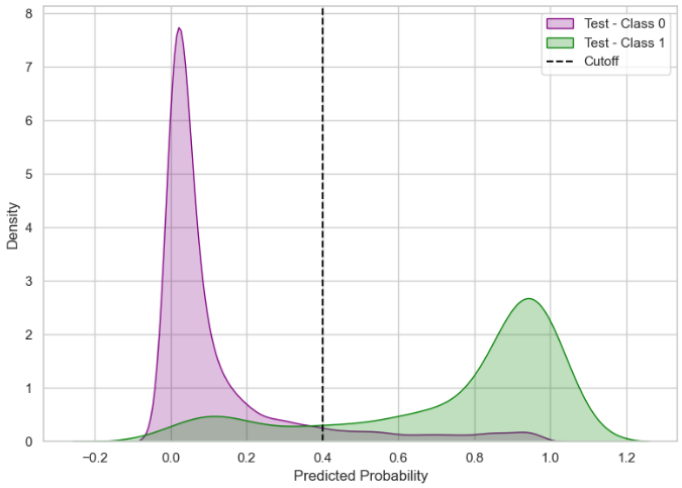


Figure 14: Density chart for both classes

In the context of churn prediction, recall (also known as sensitivity or true positive rate) measures the proportion of actual churners that are correctly identified by the model. It is defined as:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

Recall is particularly important in churn prediction for several reasons. Identifying potential churners is critical for targeted retention strategies. High recall ensures that most of the customers who are at risk of churning are identified, allowing for timely intervention. The cost associated with losing a customer is typically higher than the cost of retaining one. Therefore, it is more crucial to minimize false negatives (missed churn predictions) even if it means tolerating a higher number of false positives.

Metrics	Training Set	Testing Set
Precision	0.2013	0.1668
Recall	0.9907	0.9130
Accuracy	0.9140	0.9057
F1 Score	0.3347	0.2781

Table 6: Performance metrics for the training and testing sets

To rigorously test the accuracy of our model, we will use the most recent complete month of data (March 24) as our test set. This dataset, representing real-world

scenarios, will serve as a robust benchmark for assessing our model's performance. After applying our trained model to this test set, we will compare the predicted outcomes with the actual values. The confusion matrix (see Figure 15) provides a detailed breakdown of the model's classification performance by summarizing the counts of true positive (TP), false positive (FP), true negative (TN), and false negative (FN) predictions. As we can see in Figure 6, the model demonstrates a recall of 0.9130, capturing a substantial portion of true churners. However, the precision is lower at 0.1668, indicating a higher rate of false positives.

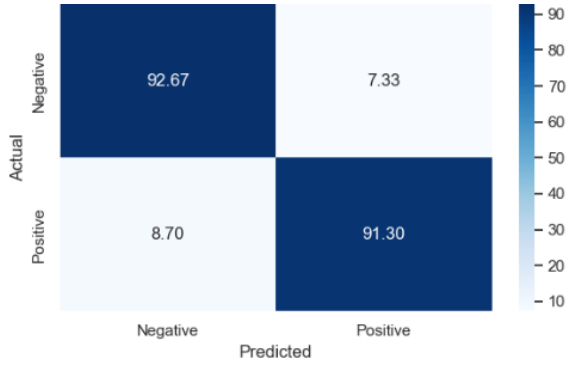


Figure 15: Test Confusion Matrix (Percentage)

4.2.1 Explainability and Interpretability of ML

Now that our model has demonstrated solid performance metrics, let us deep dive into to gain a clearer understanding of its decision-making process. To achieve this, we will use Explainability AI, particularly the Shapley Additive Explanations (SHAP) technique. [7]

Let us first take a look at which variables have the most influence when using our model (see Figure 16):

The variable that affects the most is usage. MoM (2,3) represents the Month-over-Month change between Month 2 and Month 3. This is logical since it reflects the usage pattern immediately preceding the determination of account status. Hence, usage metrics (both for web and mobile) are the most critical features, emphasizing the importance of user engagement patterns. Retention history also play and important role, likely capturing user loyalty. Finally, demographic and categorical

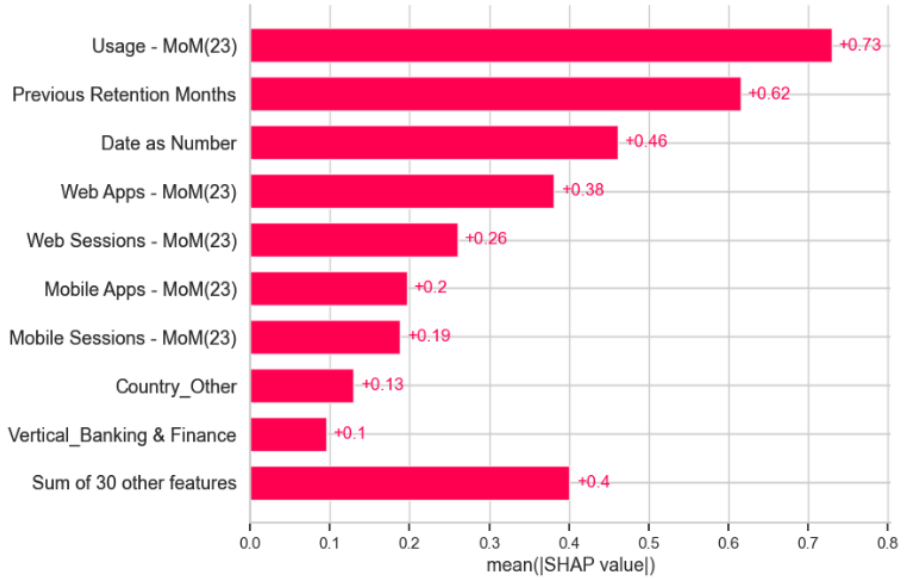
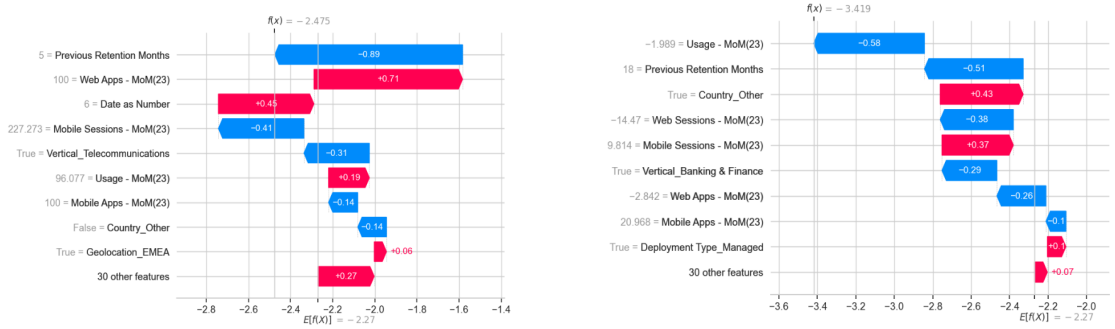


Figure 16: Shap bar plot

attributes like country and industry vertical have less impact but still contribute meaningful information.

The waterfall plots that can be seen in Figure 17 are used to illustrate the contribution of various features to the prediction of whether a customer will churn or not.



(a) Churned customer (b) Non Churned customer

Figure 17: Waterfall plot for both classes

The features that contributed most to the churn prediction include high values for *Web Apps MoM(2,3)* and low values for *Previous Retention Months*. These indicate higher activity in usage and a shorter retention period, both of which contribute to the likelihood of churn. This explanation coincides with the Survival Analysis

that we previously did, which highlights the early months as a critical period for customer retention. (see Subsection 4.1.3). On the other hand, the features that contributed most to the non-churn prediction include high values for *Usage MoM (2,3)* and *Previous Retention Months*. This indicates higher usage over time and longer retention, both of which reduce the likelihood of churn.

4.3 Model 3 - Deep Learning: Embedding

In this final chapter of our modeling journey, let us now move on the deep learning methodology, particularly to embedding techniques. Embeddings are representations of categorical variables in a continuous vector space, learned through the model's training process. These representations encode essential information about the categorical variables, allowing the model to better understand and make predictions based on them. In other words, it translates human-readable text into machine-readable and searchable vectors. [8]

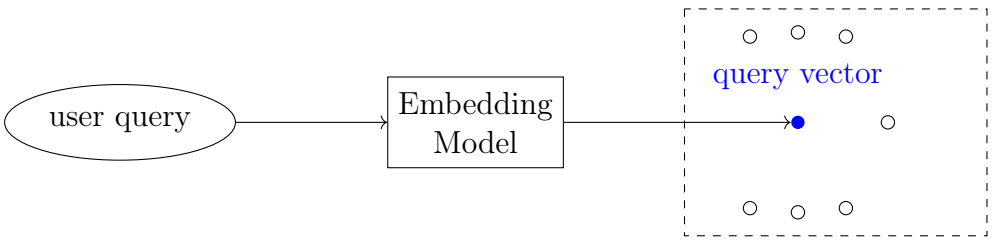


Figure 18: Framework of Embedding Models

4.3.1 Data Preparation

Before delving into the deep learning model, we need to prepare the data appropriately. This involves:

1. **Feature Compilation:** We will start by compiling relevant features from our dataset into a structured textual format. For doing that, we will create a function that concatenates these features into a cohesive textual representation.

2. **Text Transformation:** Using the compiled text function, we apply it to each row of our dataset to generate a list of sentences. Each sentence represents a data instance with its features compiled into a textual format.

3. **Embedding Model Initialization:** The Sentence Transformer model is now initialized to encode the textual sentences into fixed-length vector representations, also known as embeddings.

4. **Embedding Generation:** With the initialized model, we encode the generated sentences into embeddings. This process involves converting each textual sentence into a numerical representation in a continuous vector space.

The resulting embeddings are stored in a DataFrame (see Figure 7), where each row represents the embedding for a corresponding sentence.

	0	1	2	3	4	...	382	383
0	0.123	0.456	0.789	0.321	0.654	0.987	0.345	0.678
1	0.234	0.567	0.890	0.432	0.765	0.098	0.456	0.789
2	0.345	0.678	0.901	0.543	0.876	0.209	0.567	0.890
3	0.456	0.789	0.012	0.654	0.987	0.320	0.678	0.901
4	0.567	0.890	0.123	0.765	0.098	0.431	0.789	0.012
...	0.678	0.901	0.234	0.876	0.209	0.542	0.890	0.123

Table 7: Embeddings table

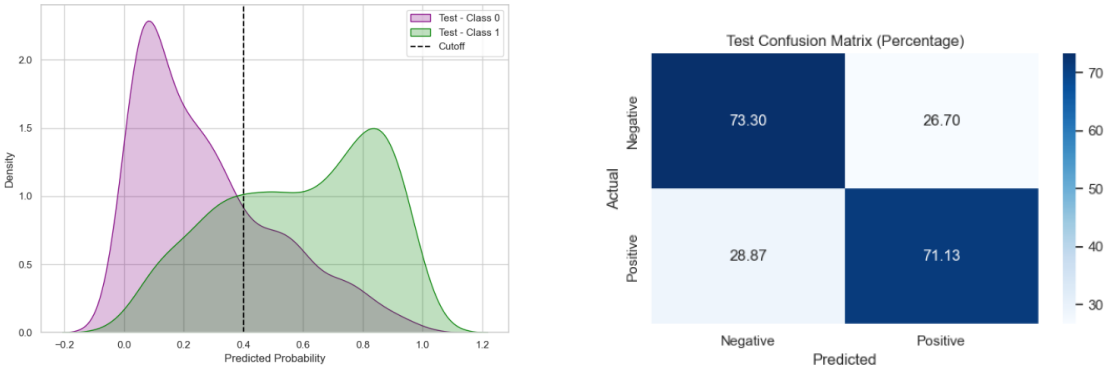
4.3.2 Implementation and Results

Once the data is prepared and embeddings are generated, we proceed with the implementation of the embedding model and evaluate its performance.

Following the same methodology as before, the data was split into training and testing with a 80% and 20% respectively. As we have seen in Section 4.2, among all the models LGBM was the one that performed better, hence, we will again apply LGBM with our embedings.

Figure 19 illustrates the density chart and confusion matrix after applying the embedding model. The density chart shows that the model can separate the data accurately with a threshold of 0.4. Table 8 summarizes the performance metrics for both the training and testing sets. Although the model demonstrates excellent performance on the training set, the testing set results reveal a recall of 0.7378, a precision of 0.6667, and an accuracy of 0.7004.

These results indicate that while the model performs well on the training data, there is a noticeable drop in performance on the testing data.



(a) Density Chart (b) Confussion Matrix

Figure 19: Evaluation of LGBM using Embedding model

Metrics	Training Set	Testing Set
Precision	0.9994	0.6667
Recall	0.9907	0.7113
Accuracy	0.9140	0.7241
F1 Score	0.8895	0.7004

Table 8: Performance metrics for the training and testing sets

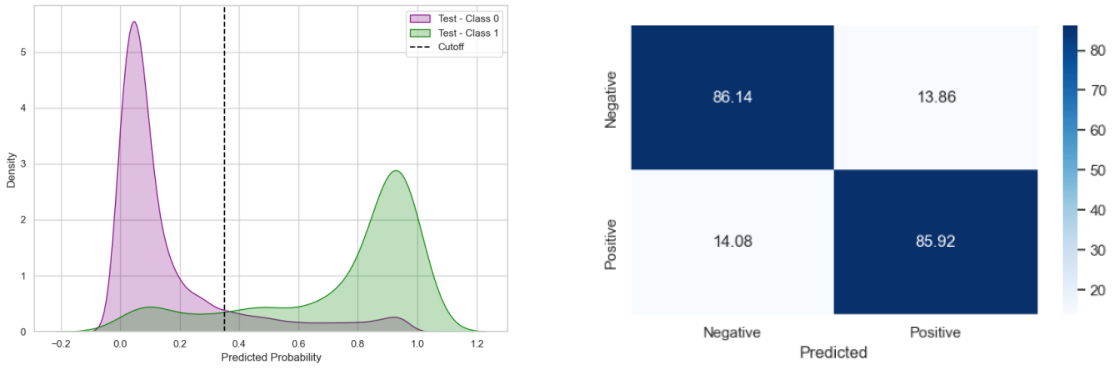
Chapter 5

RESULTS

In this chapter we present the results and findings form the four different remaining capabilities.

5.1 Model 1 - Traditional Machine Learning

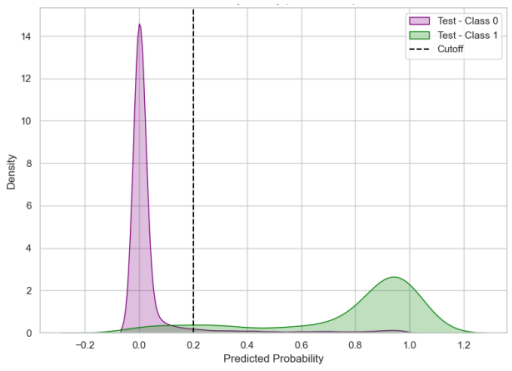
Figure 20, 21, 22, 23, are the results of implementing a traditional machine learning model, LGBM. As we mention the goal is to implement a model able to identify correctly the two classes: Churn and No-Churn. Hence, we want have this two classes as separated as possible. This is correctly achieved using traditional ML.



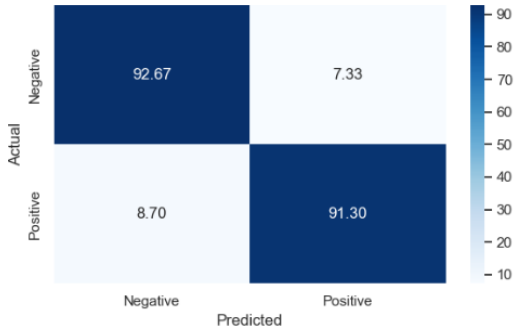
(a) Density Chart Web

(b) Confusion Matrix Web

Figure 20: Evaluation of LGBM Web

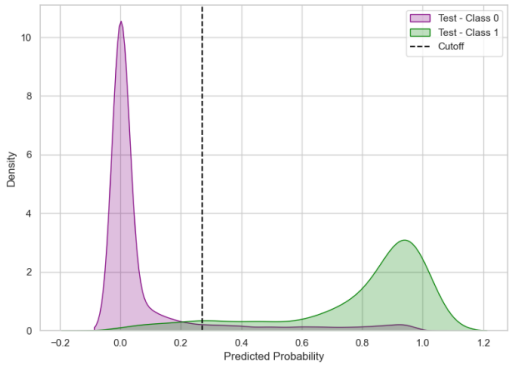


(a) Density Chart Web

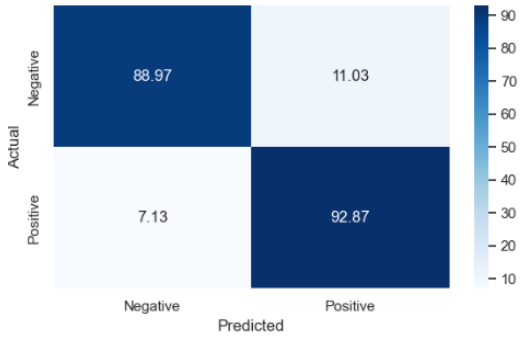


(b) Confusion Matrix Web

Figure 21: Evaluation of LGBM Mobile

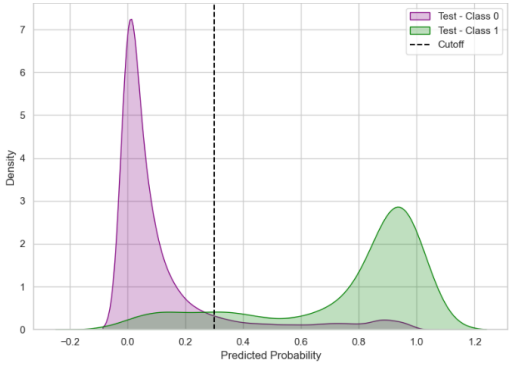


(a) Density Chart Session Replay

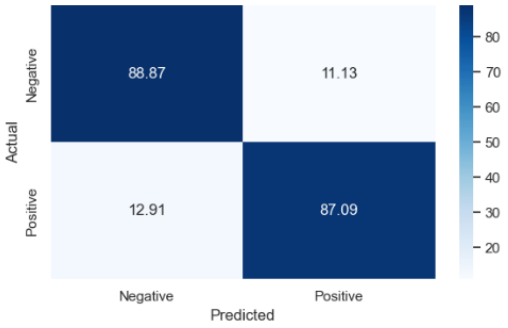


(b) Confusion Matrix Session Replay

Figure 22: Evaluation of LGBM Session Replay



(a) Density Chart Synthetics



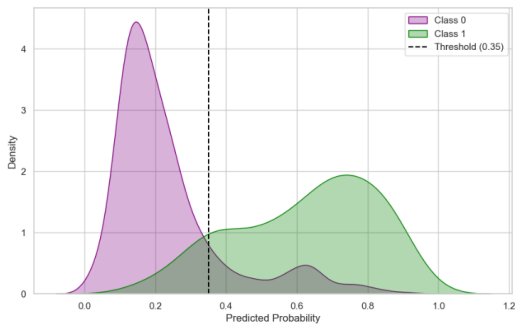
(b) Confusion Matrix Synthetics

Figure 23: Evaluation of LGBM for Synthetics

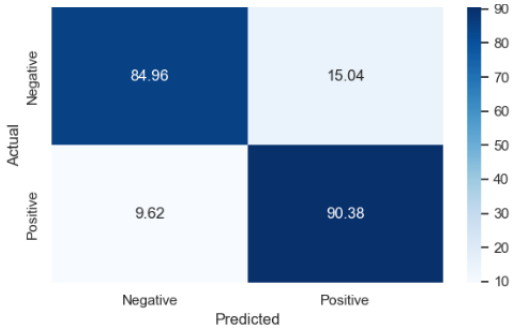
5.2 Model 2 - Statistic with Survival Analysis

Figures 24, 25, 26, 27 show results of the survival model analysis. For Web and Mobile capabilities, the models perform reasonably well, with a recall around 0.9 and an accuracy of 0.85.

However, for Session Replay and Synthetics, the performance metrics are less satisfactory. The density chart does not separate the classes well, leading to poor results in the confusion matrix. This indicates that the model struggles to distinguish between different classes in these particular categories, affecting overall performance.

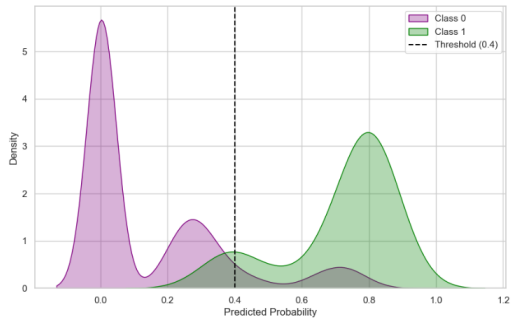


(a) Density Chart Web

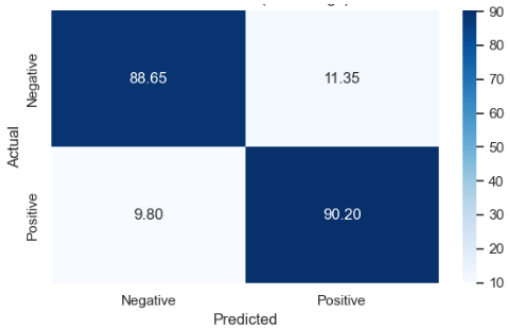


(b) Confusion Matrix Web

Figure 24: Evaluation of Survival Random Forest for Web

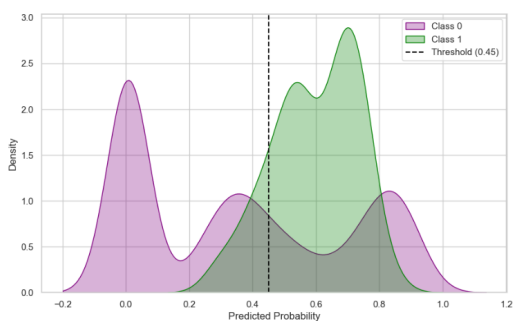


(a) Density Chart Mobile

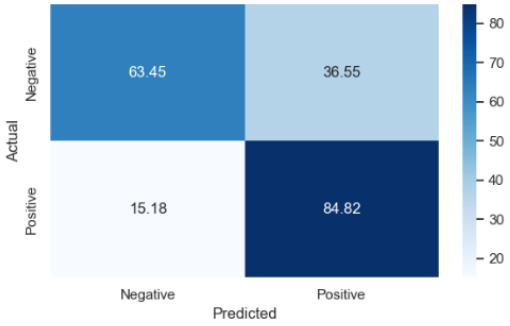


(b) Confusion Matrix Mobile

Figure 25: Evaluation of Survival Random Forest for Mobile

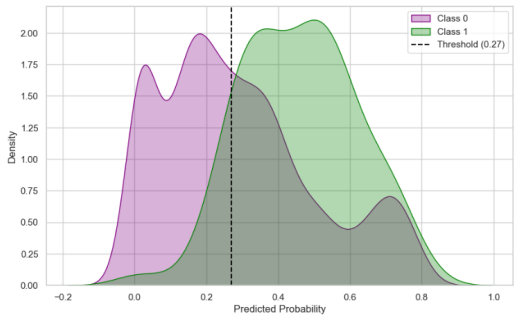


(a) Density Chart Session Replay

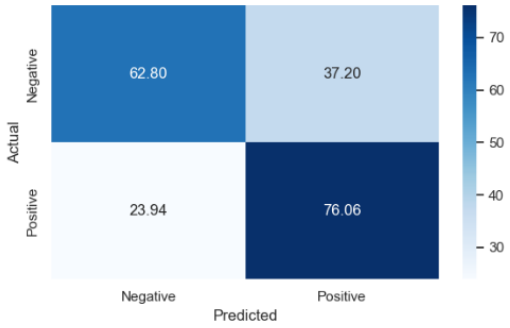


(b) Confusion Matrix Session Replay

Figure 26: Evaluation of Survival Random Forest for Session Replay



(a) Density Chart Synthetics

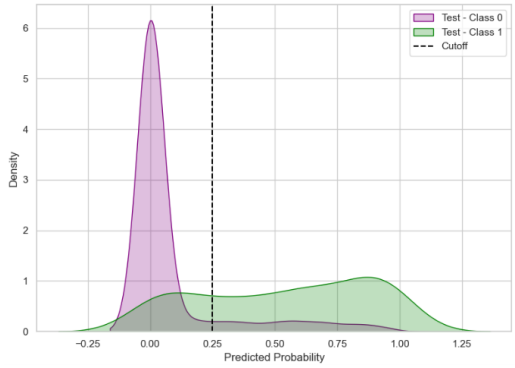


(b) Confusion Matrix Synthetics

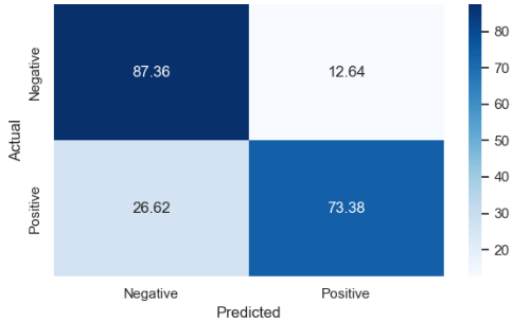
Figure 27: Evaluation of Survival Random Forest for Synthetics

5.3 Model 3 - Deep Learning with Embedding

Finally, figures 28, 29, 30, 41, show the results of our latest model. The performance across different capabilities varies significantly. For Web and Mobile, the model performs quite well, especially for class 0, achieving a recall around 0.85. However, for Session Replay and Synthetics, the results are less satisfactory, with class 1 not being predicted as accurately. This is evident from the density chart, which shows poor class separation. Overall, these results are not superior to our initial model based on traditional machine learning techniques, which will be featured in the final dashboard.

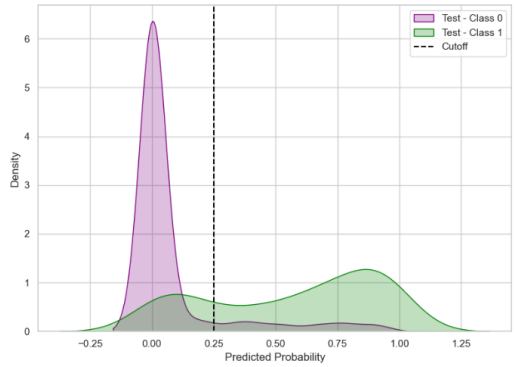


(a) Density Chart Web

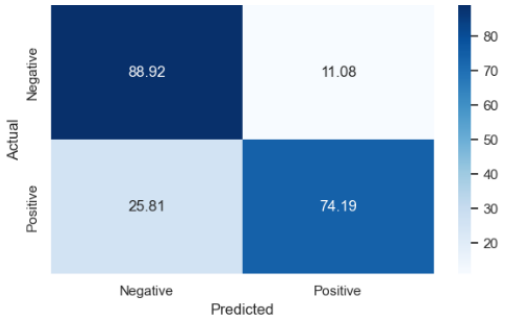


(b) Confusion Matrix Web

Figure 28: Evaluation of Embedding Model Web

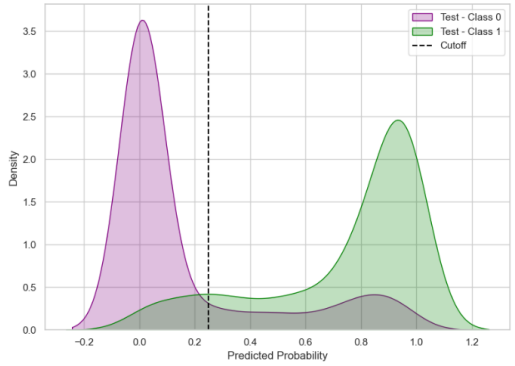


(a) Density Chart Web

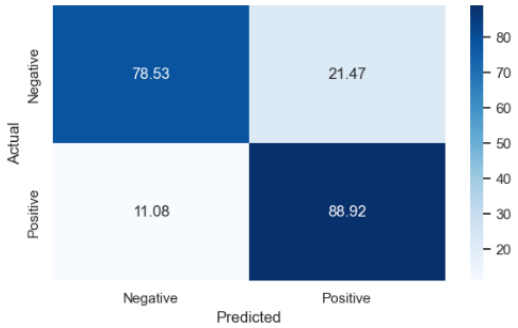


(b) Confusion Matrix Web

Figure 29: Evaluation of Embedding Model Mobile

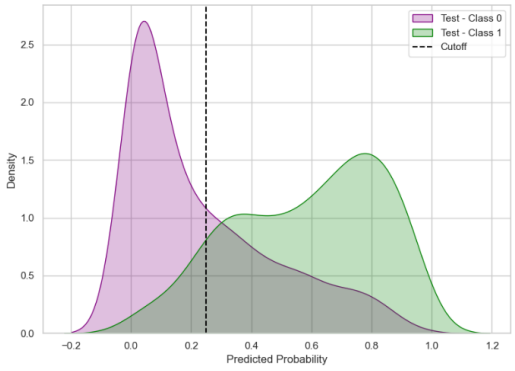


(a) Density Chart Web

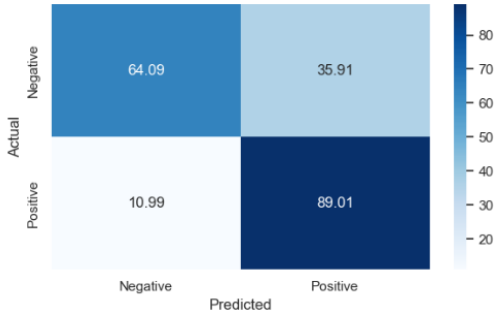


(b) Confusion Matrix Web

Figure 30: Evaluation of Embedding Model Session Replay



(a) Density Chart Web



(b) Confusion Matrix Web

Figure 31: Evaluation of Embedding Model Synthetics

In this study, we have explored three approaches to predicting customer churn: survival analysis, traditional machine learning and deep learning with embedding.

The survival analysis model performed well for Web and Mobile capabilities, but struggled with Session Replay and Synthetics, indicating its suitability for specific data types where the time-to-event aspect is crucial. The traditional machine learning model (LGBM) demonstrated robust performance across all categories, effectively separating churn and no-churn classes, making it a reliable choice. On the other hand, the deep learning model showed promise for Web and Mobile capabilities, but under-performed for Session Replay and Synthetics, suggesting the need for further tuning or additional data.

Overall, the LGBM model outperformed the other approaches in accuracy and class separation, though each method offers unique insights.

Chapter 6

DASHBOARD INTERFACE

In order to show the results to the team members, an interactive Dashboard was build. The GitHub repository with the Report can be found in this link: https://github.com/Tinsooon/TFG

Requirements:
To have Power BI Desktop or any compatible software to open the dashboard.

Disclaimer:
For data privacy reasons, this is a simplified version, so no company names or sensitive data appear.

The dashboard is structured in four different pages with interactive chart and filters. On the right side of each page, five different buttons are placed so the user can select the different capabilities available.

6.1 Lost Accounts

In this first section, some basic information about lost accounts can be found. The main line chart represents the number of churned accounts each month, along with a trend line showing whether it is increasing or decreasing. There is also a tool tip showing what percentage represents out of the total. In the table above, information

about each client can be found, including the Sales Force link, last month usage, Account Executive email, Retention Month, and more.

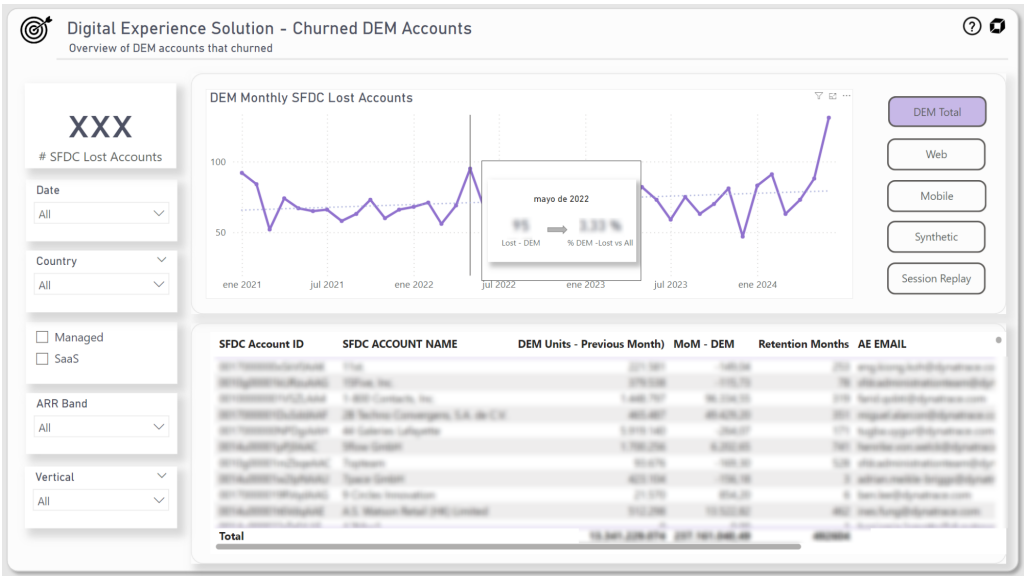


Figure 32: Enter Caption

6.2 Survival & Account Retention

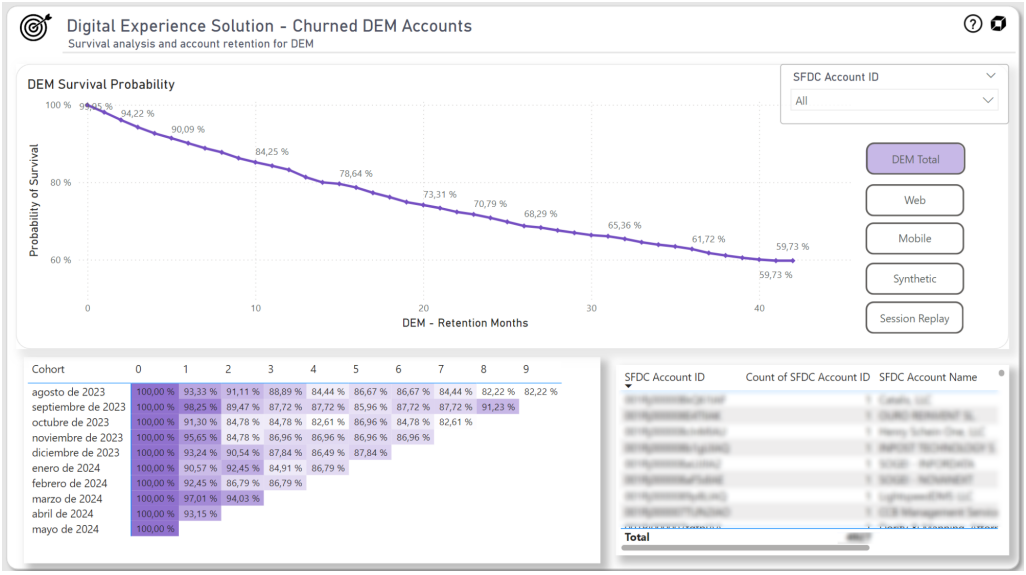


Figure 33: Enter Caption

This second page provides insights into survival analysis and account retention for DEM Accounts. The first chart displays the survival curve of all accounts, with

the possibility to select individual accounts using the provided filters. Below the main chart, a heatmap cohort analysis table illustrates account retention for different months. Each row represents a cohort starting from a specific month, and the columns show retention percentages. This matrix allows us to understand the retention patterns over time and identify trends that may indicate potential risks of churn.

6.3 Statistics

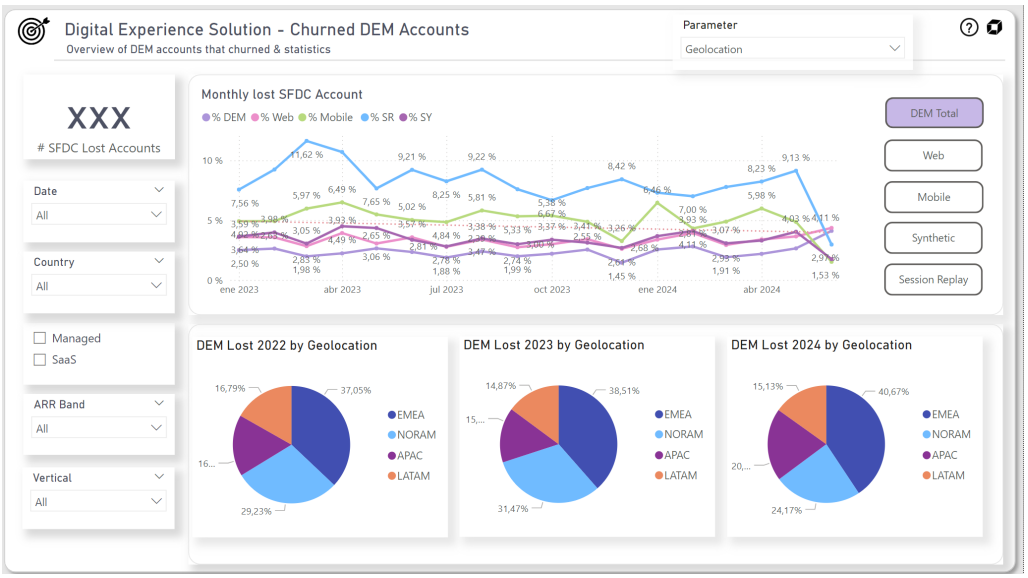


Figure 34: Enter Caption

This next page allows users to gain insights into where and when accounts are being lost and helps identify trends and patterns in customer churn. The main line chart displays the churn rate of each capability. This type of chart allows to see which product has the highest churn rate, enabling the identification of at-risk products.

The churn rate can be calculated using the following formula:

$$\text{Churn Rate} = \frac{\text{Number of Lost Customers}}{\text{Total Number of Customers at the Start of the Month}} \tag{6.1}$$

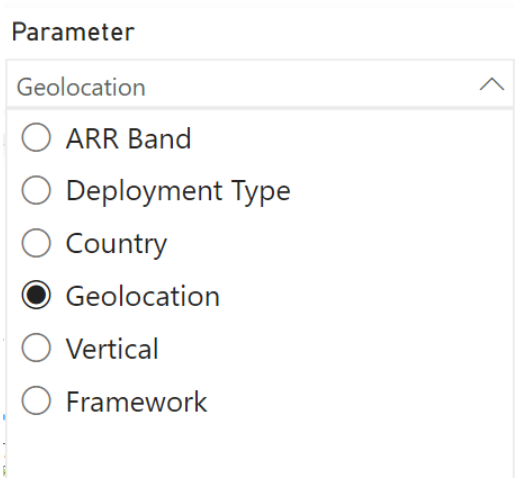


Figure 35: Your image caption

The three pie charts at the bottom show the distribution of lost accounts in 2022, 2023 and 2024 according to a field parameter. This parameter can be chosen using the drop down menu at the top. This menu has different parameters to compare with, such as Country, Vertical, ARR Band and many more. By adjusting these parameter, users can analyze the data from different perspectives.

6.4 Churn Prediction

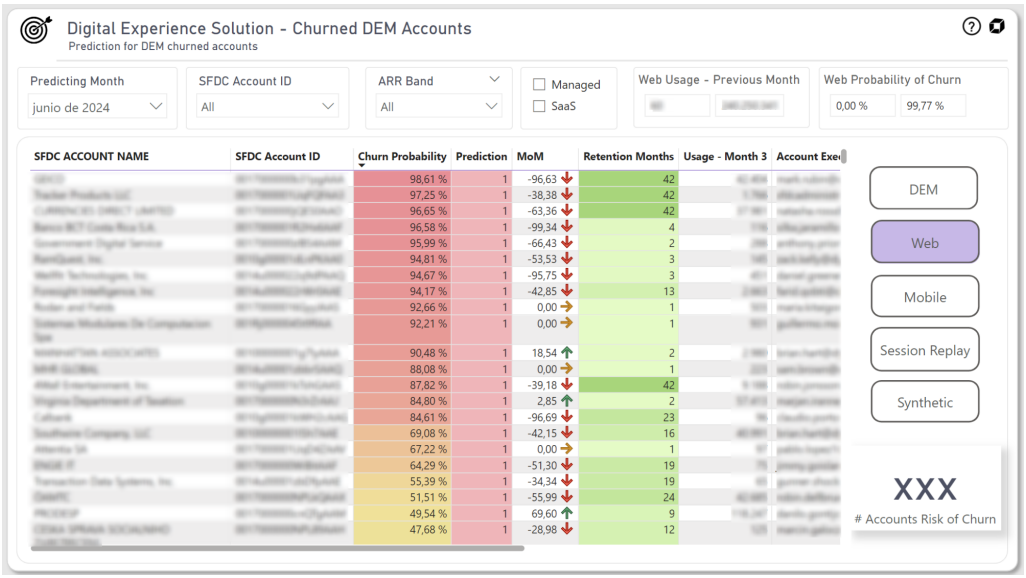


Figure 36: Enter Caption

This last page shows the results of the implemented models for predicting churn. There is a main table that lists information about each account including the churn probability, the prediction of the selected month and some other usage metrics such as retention months or Month-over-Month grow. The visual indicators and color coding enhance the user experience by making it easy to identify accounts at high risk of churn and track changes in usage patterns.

Chapter 7

CONCLUSIONS AND FUTURE WORK

In this project, we successfully developed a comprehensive approach to understanding and predicting customer churn within a digital experience solution and all its capabilities.

Looking at the future, there are several things that can be done to improve the project:

1. **Incorporate Additional Attributes:** By including more attributes in our dataset, we can improve the model's precision and ability to capture patterns in customer behavior. Attributes such as metrics, events, clicks could provide deeper insights.
2. **Extend Sliding Window:** Adjusting the sliding window approach to incorporate more months of historical data could improve the model's ability to detect longer trends and patterns.
3. **Longer Term Predictions:** Extending the prediction horizon to forecast, so instead of just month it could predict what would happen over the next three months, would provide a more strategic perspective on potential risks and allow for more timely and effective retention strategies.

4. **Implement Alert Signals**: Developing an alert system to notify product specialists or account executives when there is a significant drops in usage can enable proactive interventions. These alerts could be integrated with communication platforms such as Slack or a dedicated forum, and can be categorized into different levels of urgency (low, medium, very alarming) to prioritize responses.

We started developing some simple models and gradually we have improved its complexity. However, upon comparing all our models, we made a surprising discovery: in our case, complexity does not necessarily implicate a better performance. Instead, we found out that simplicity consistently outperforms complexity.

Bibliography

[1] Dynatrace. Monitoring environment. URL https://docs.dynatrace.com/docs/get-started/monitoring-environment. Retrieved February 2024.

[2] Nambakhsh, C. Digital experience monitoring explained! (February 13, 2024). URL https://watchthem.live/digital-experience-monitoring/. Retrieved February 2024.

[3] Statistical modeling. URL https://www.heavy.ai/technical-glossary/statistical-modeling. Retrieved April 2024.

[4] What is deep learning? URL https://cloud.google.com/discover/what-is-deep-learning. Retrieved June 2024.

[5] Adib, R. Understanding kaplan-meier estimator (survival analysis) (2019). URL https://towardsdatascience.com/understanding-kaplan-meier-estimator-68258e26a3e4. Retrieved March 2024.

[6] Thakral, K. Window sliding technique (18 Apr, 2024). URL https://www.geeksforgeeks.org/window-sliding-technique/. Retrieved March 2024.

[7] Lundberg, S. M. An introduction to explainable ai with shapley values. URL https://shap.readthedocs.io/en/latest/example_notebooks/overviews/An%20introduction%20to%20explainable%20AI%20with%20Shapley%20values.html. Retrieved June 16, 2024.

[8] Pinecone. Choosing an embedding model. URL `https://www.pinecone.io/learn/series/rag/embedding-models-rundown/`. Retrieved May, 2024.

[9] Towards Data Science. Performance metrics: Confusion matrix, precision, recall. URL `https://towardsdatascience.com`. Retrieved June 2024.

Appendix A

Appendix

A.1 Python notebook and Dashboard

The source code, notebooks, and Dashboard used can be found in the following GitHub Repository: https://github.com/Tinsooon/TFG

A.2 Evaluation Plots

A.2.1 ROC-Curve

A ROC (Receiver Operating Characteristic) curve is a graphical representation used to evaluate the performance of a binary classification model. It plots the True Positive Rate (TPR) against the False Positive Rate (FPR) at various threshold settings.

$$\text{TPR} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

$$\text{FPR} = \frac{\text{False Positives}}{\text{False Positives} + \text{True Negatives}}$$

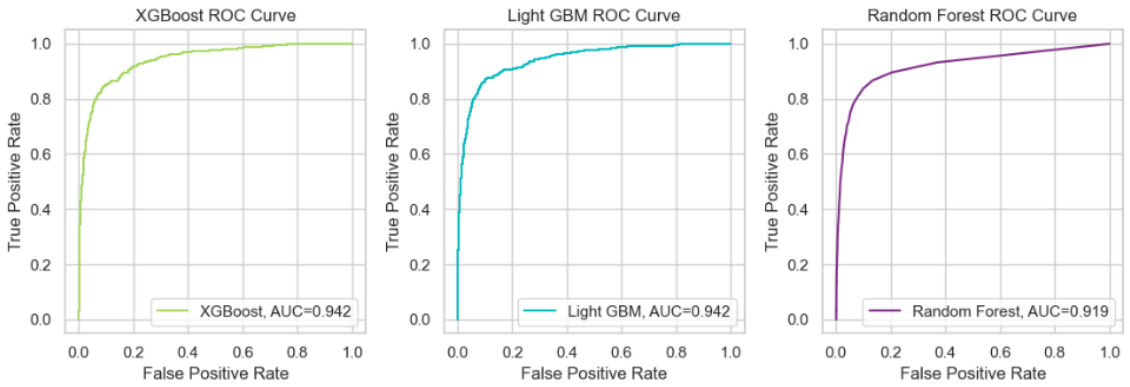


Figure 37: ROC- Curve for ML models

A.2.2 Precision-Recall Curve

The precision-recall (PR) curve is a graphical representation used in binary classification tasks to assess the performance of a model. It plots the trade-off between precision and recall for different thresholds of a classifier.

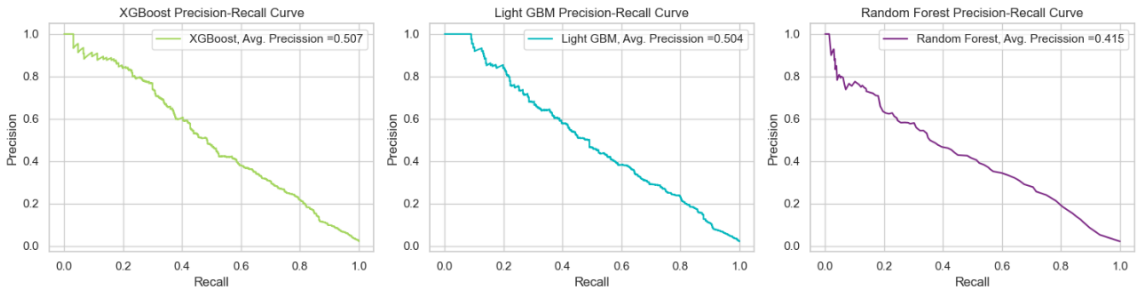


Figure 38: Precision-Recall for ML models

A.3 Evaluation Metrics

To analyze how the ML models perform with the data, several metrics and plots were utilized: [9]

A.3.1 Confusion Matrix

A confusion matrix evaluates the accuracy of a classification model. It displays the number of correct and incorrect predictions made by the model for each class. The

matrix is composed of four values: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).

A.3.2 Precision

Precision measures the proportion of positive predictions that are actually correct. It ranges from 0 to 1, and a higher value indicates better performance. Precision can be computed using the formula:

$$\text{Precision} = \frac{TP}{TP + FP}$$

A.3.3 Recall

Recall measures the proportion of actual positives that are correctly identified by the model. It also ranges from 0 to 1, and higher values are desirable. Recall is calculated as:

$$\text{Recall} = \frac{TP}{TP + FN}$$

A.3.4 Accuracy

Accuracy is the fraction of predictions that the model got correct [9]. It ranges from 0 to 1, and higher values indicate better accuracy:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

A.3.5 F1 Score

F1 score is the harmonic mean of precision and recall, providing a single metric to evaluate a model's performance [9]. A higher F1 score indicates better overall performance:

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

These metrics are essential tools for assessing and comparing the effectiveness of machine learning models in classification tasks.

A.4 Shap Values

The SHAP approach is a powerful strategy for explaining the influence of each feature on a machine learning model. It uses SHAP values and various SHAP plots to achieve this.

SHAP values quantify the contribution of each feature to the model's output. They accomplish this by measuring the average change in the model's prediction when a feature is included versus when it is excluded across all possible combinations of features. These values are computed by comparing the expected output with and without each feature, ensuring a comprehensive understanding of feature importance.

A.4.1 Waterfall Plot

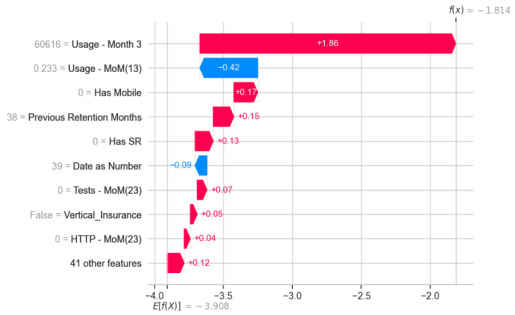
This graph illustrates both positive and negative contributions of features to the model prediction. Features are ordered from most to least influential, providing a clear hierarchy of feature importance.

A.4.2 Bar Plot

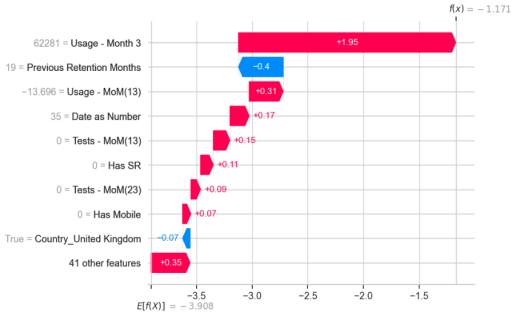
This plot displays the average or sum of absolute SHAP values for each feature in the model. It offers a relative ranking of feature importance, highlighting the most significant features at the top and less influential ones at the bottom.

A.4.3 Summary Plot

A graphical summary of the model's global feature importance. It shows how each feature contributes positively or negatively to the model's output, facilitating the identification of patterns and relationships between features.

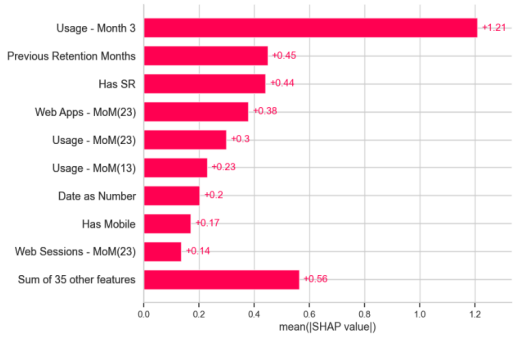


(a) Waterfall plot Synthetic (non-churn)

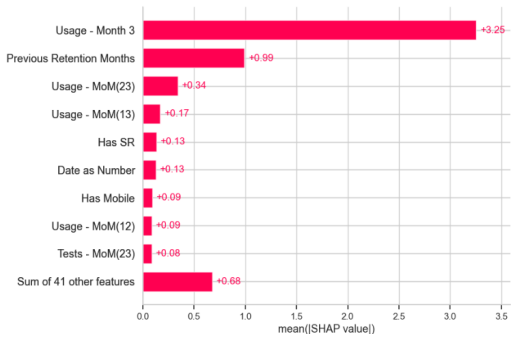


(b) Waterfall plot Synthetic (churn)

Figure 39: Waterfall plot

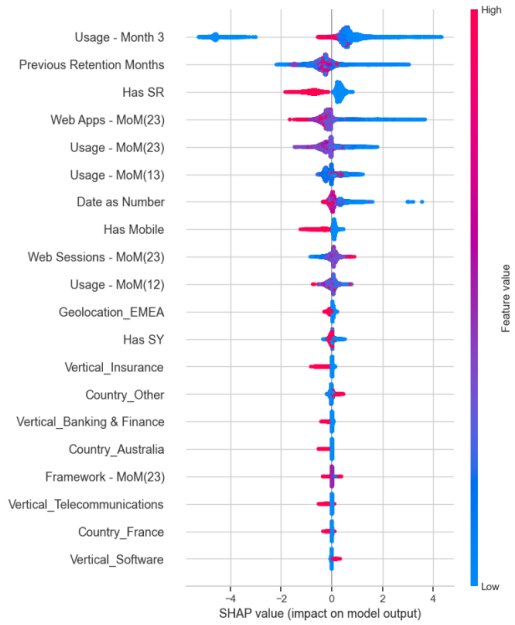


(a) Bar plot Web

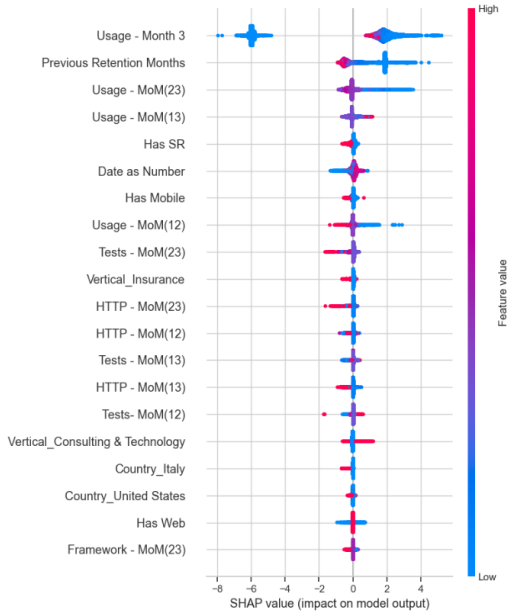


(b) Bar plot Synthetic

Figure 40: Waterfall plot



(a) Summary plot for Web



(b) Summary plot for Synthetic

Figure 41: Summary plot

A.5 Relational Model PowerBI

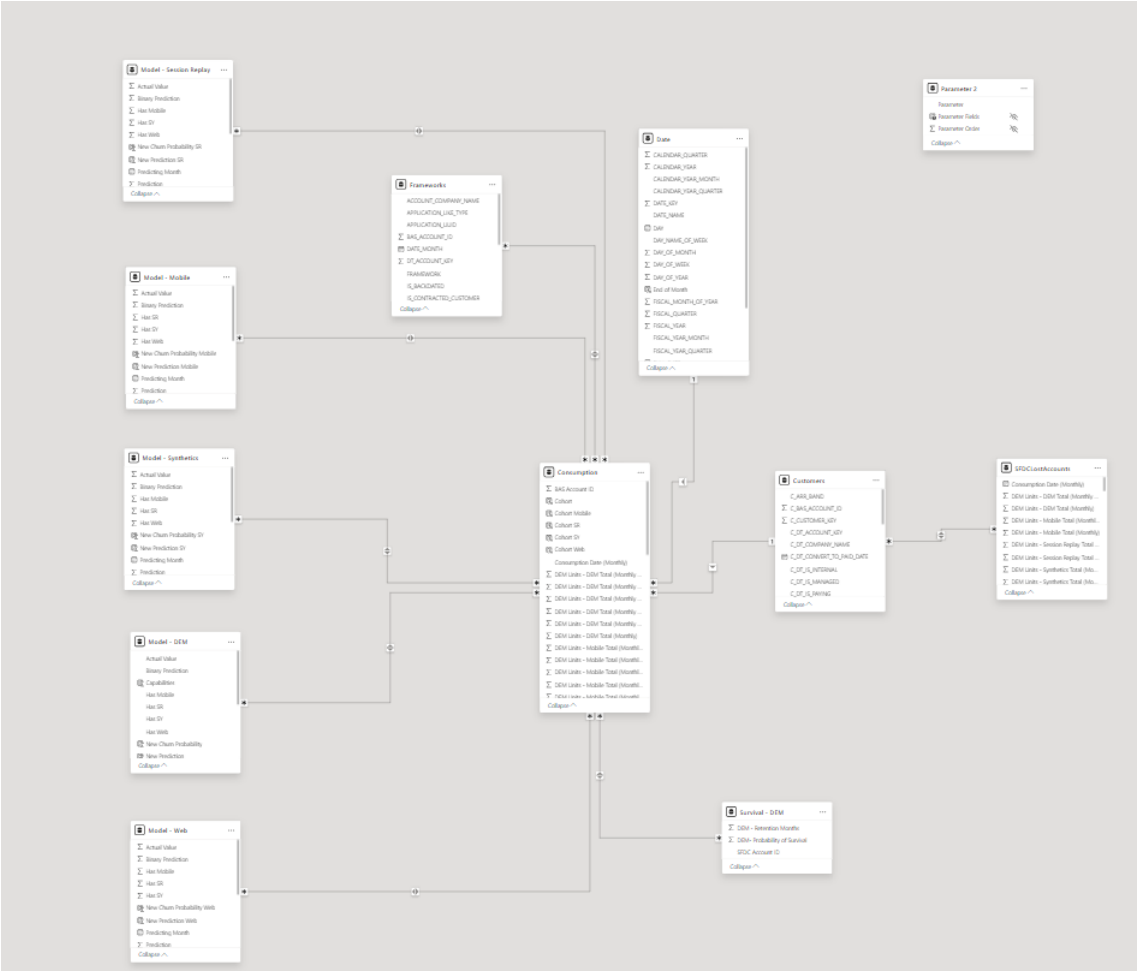


Figure 42: Relational Model