

Sales Forecast and Churn Prediction for the International Health Insurance Company

FINAL REPORT

Course: Data Science Continuous Mar23

Jonathan Leipold | Christian Hirning | Rumiya Al-Meri

09.11.2023

Table of contents

[I. Introduction 3](#_Toc150706828)

[About the company and the product 3](#_Toc150706829)

[Background 3](#_Toc150706830)

[Contribution 3](#_Toc150706831)

[Objectives 4](#_Toc150706832)

[Data Framework 4](#_Toc150706833)

[II. Sales prediction 6](#_Toc150706834)

[III. 1 Relevance 6](#_Toc150706835)

[III. 2 Pre-processing and feature engineering 6](#_Toc150706836)

[Correlations between features 8](#_Toc150706837)

[III. 3 Sales prediction with Time Series modelling 11](#_Toc150706838)

[SARIMA 13](#_Toc150706839)

[Conclusion 14](#_Toc150706840)

[Considerations 15](#_Toc150706841)

[III. 4 Sales prediction with Classification modelling 16](#_Toc150706842)

[Classification of the problem 16](#_Toc150706843)

[Model choice and optimization 16](#_Toc150706844)

[Interpretation of results 17](#_Toc150706845)

[III. Churn prediction 19](#_Toc150706846)

[III. 1 Data collection & Description 19](#_Toc150706847)

[III. 2 Pre-processing, Visualization & Dependencies 19](#_Toc150706848)

[Target variable 19](#_Toc150706849)

[Handling Missing Values 21](#_Toc150706850)

[Outliers 22](#_Toc150706851)

[Correlations between features 24](#_Toc150706852)

[Correlations with target variable 24](#_Toc150706853)

[Distributions 25](#_Toc150706854)

[Encoding 26](#_Toc150706855)

[Scaling 27](#_Toc150706856)

[Final preprocessing parameters 27](#_Toc150706857)

[III. 3 Churn prediction modelling 28](#_Toc150706858)

[Classification of the problem 28](#_Toc150706859)

[Model choice and optimization 29](#_Toc150706860)

[Alternative target ds\_terminated 29](#_Toc150706861)

[Model comparison on ‘terminated’ target 31](#_Toc150706862)

[Interpretations with SHAP & Feature reduction 34](#_Toc150706863)

[Predicting Probabilities 35](#_Toc150706864)

[Conclusion 37](#_Toc150706865)

[III. 4 CHALLENGES & CONCLUSION 39](#_Toc150706866)

[Bibliography 40](#_Toc150706867)

[Appendices 41](#_Toc150706868)

[Figures & tables 41](#_Toc150706869)

[Code 45](#_Toc150706870)

# Introduction

## About the company and the product

[BDAE](https://www.bdae.com/en/bdae-group/about-bdae-group) is a specialist for international health insurance products. It is an insurance broker in co-working with the big health insurance companies. BDAE has its’ own sales & product development as well as claims handling service. Main target group are expatriates with the longer international assignments (from at least 1 year).

The company offers own health insurance products. There are several products depending on multiple factors like the insurance time period, home country and the area where the insured country of stay is situated. For this project it was initially decided on considering only one product (insurance type) to reduce the complexity. The product price is a premium amount which is paid by a client on a monthly, quarterly or a yearly basis.

## Background

The sales forecast is inevitable for the company. Based on it BDAE takes decisions on new products development and clients’ acquisition to ensure a continuous growth and avoid existential threat. Quick access to forecast by product allows quick decision-making and resources optimization which in turn reduce costs and increase overall profit. Churn predictions are not made yet and would be a novelty in the company’s data analysis. In general, there is no implementation of python or ML models yet in the business.

All data is collected in the ERP-system (SAP) based on a SQL-database. The forecast of future sales is still partly based on manual estimations. The sums and ratios of historical sales data are used combined with manual estimations of summed sales amounts to predict future sales. Data is grouped by (insurance-)product / category, time and some other variables in Power Pivot and Excel. Churn predictions are not yet implemented into the forecast procedure by the company. The goal of this project from a technical point of view is to support the manual estimations and decisions with the reliable ML-models.

From a scientific point of view there are several challenges to overcome by finding and purifying useful data as well as choosing the best ML-model for sales forecast. It is described later in details in the chapter IV. Since there is no information about future behaviour such as the number of contracts is unknown as well as the profile of future clients, it is challenging to find the right and reliable ML-model to predict future sales.

## Contribution

In this project group Johnathan Leipold is a representative of the BDAE company, an industry expert, the data owner and the initiator of a current project. He has a mathematical background and little programming and Deep Learning knowledge from a student job and private courses but no experience in ML projects. Jonathan was consulting on the data during the project, set up the initial data base and mostly all pre-processing steps as well as new feature collection from the ERP-System and feature engineering in SQL for churn prediction. Christian Hirning and Rumiya Al-Meri have no experience in the insurance industry but rather in statistics and Christian as well a deeper knowledge in programming. Raphael Kassel (DataScientest) contributed as a project tutor.

Jonathan regularly consulted with a company IT expert with basic statistics and ML skills about useful modelling options, and with another colleague about the quality and meaning of the feature content. No data scientists or ML experts were involved.

## Objectives

The **main objective** was to create **the best performing model for sales predictions**, in particularly prediction of premium amounts per month. Due to the big variety of product characteristics, only transactions concerning one main product type were considered to build a prototype.

**Initially two main goals were defined:**

1. Find the best model for forecasting / predicting the premium amount
2. Find out how premium adjustments impact the value of premium amount

During the project, the project group faced the problem of a limited number of features which are known for the future. Therefore, it was decided on project extension with the further objective, namely **churn predictions**. The contracts’ data for all products was taken and enriched by additional, information from the ERP-System.

Within this sub-project another two goals were defined:

1. Identify main features that have an impact on customers’ termination behaviour
2. Find active contracts that are more likely to get terminated by the customer

## Data Framework

The data is owned by BDAE Group and not available to the public. Each group member signed the confidentiality agreement with BDAE. Personal information like name and address were not collected, others like ContractID and ProductName were replaced with pseudonymised values in SQL.

All data comes from the ERP-System SAP Business One. It was collected and joined on a Microsoft SQL Server as views. Some features were created by calculations in SQL.

In the end 4 views were exported as csv files for import and modelling:

1. SalesData-Example.csv  
   Sales Data in form of transactions, in total about 230 000 transactions. Each transaction belongs to a specific contract and time period and represents incoming and outgoing cash flow. This data only includes transactions belonging to one main product and the period of 2014-2023 YTD. The Sales Data dataset includes as well 36 variables (e.g., birthday, policy StartDate, premium Amount, Contract Id and Fee Rate, Zone Desc).

1. premium\_adjustments\_example.csv  
   This data contains information about adjustments of premium amounts of the main product, in total 58 lines. Each line belongs to a specific adjustment date, ZoneModel and product group. It is merged with the sales data later via time and product code.

For the second sub-project 2 more datasets were created:

1. BDAE\_DataMining\_Policies.csv  
     
   This data contains information belonging to one specific contract. Some data comes directly from the contract information, other variables were calculated as sums, ratios etc. of the transaction lines used above as SalesData. It includes all contracts created after 01.01.2017. In total about 20 000 lines. The file was updated with more recent data as well as corrections and additional features from SQL in another file BDAE\_DataMining\_Policies\_v2.csv
2. BDAE\_DataMining\_Products.csv

Describes special characteristics of products like category, max. duration, etc. They can be merged with the contracts’ information via the unique product code. In total about 300 lines.

The project is split in two sub-projects: sales (premium amount) prediction and churn prediction and will be presented in this part by sub-project for a better understandability.

# Sales prediction

## Relevance

The development of the premium amount over the years 2014 - 2023 is most relevant for the sales prediction. Therefore, the sum of premium amounts per month was taken as a target variable. Monthly grouping was chosen from the business perspective as BDAE forecast is done by month.

During the project it turned out that the zone model has changed in 2018. Germany as a separate zone was now included into one of the new zones (see Figure 1). The replacement of old zone model with the new one explains the drop in 2019

Ein Bild, das Text, Diagramm, Reihe, Schrift enthält.

Automatisch generierte Beschreibung

Figure 1: Zone models by sum of premium amounts and number of unique contract IDs

Because to this change, the old zone model and the new zone model were considered and analysed separately. This presented a greater challenge. After analysing the data, no relevant features could be generated to predict the future development of the premium amount. This limited our ability to select additional features for modelling at a later stage. Despite the limitations, we analysed the premium amount using time series models and classification models.

## Pre-processing and feature engineering

After initial exploratory analysis of the data, missing values were deleted or replaced, and the variables were converted to the correct data type.

Looking at the premium amounts over time by month, there is a dip in 2019 (see Figure 2).

Ein Bild, das Text, Diagramm, Reihe, Zahl enthält.

Automatisch generierte Beschreibung

Figure 2: Sum of premium amounts and number of unique contract IDs over time by month

As mentioned earlier, a change in the zone model was identified. A new variable Zone\_United was created to include this zone conversion (see Figure 3). Moreover, optional parameters for ZoneModel are included in the preprocessing function.

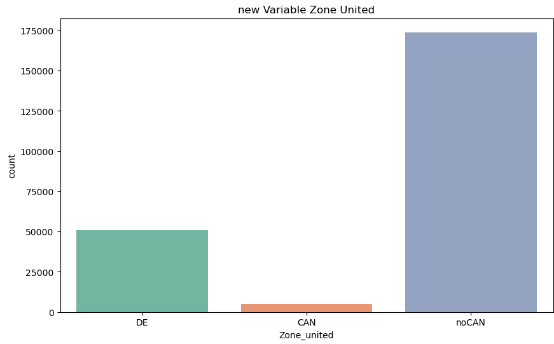
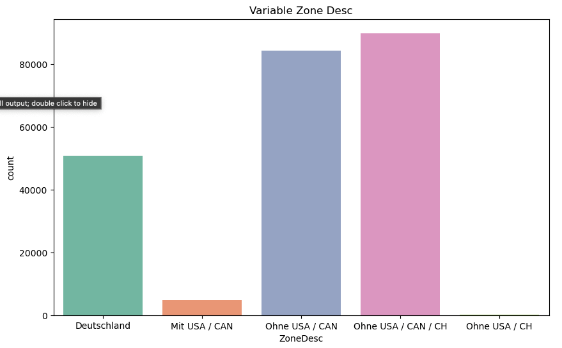


Figure 3: Count of unique Contract IDs by Zone: New Zone model vs. Old Zone model

This adjustment allowed to identify two different trajectories for the two zone models (see Figure 4). Based on this finding, the further procedure was adjusted.

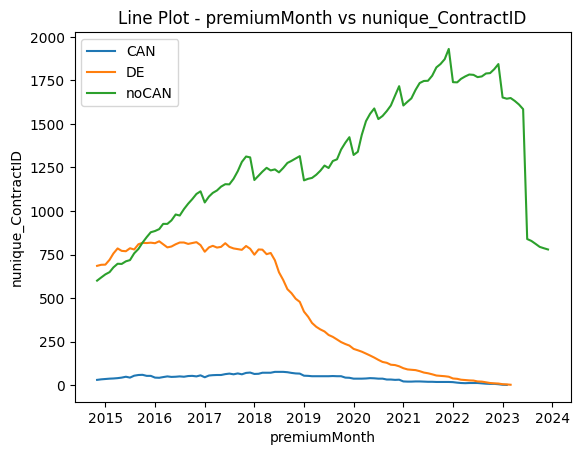


Figure 4: Sum of premium amounts and number of unique contract IDs over time by month

Dummies were created for the categorical variables and other variables such as the number of days since the last adjustment. It turned out that these variables are not suitable for the Time Series models.

For a classification model some features were created for a modelling. Due to a limited number of variables which can be grouped by month to fit to the target variable, the following three features were created: mean of the Age at premium and mean of the policy age in months. Later, the rolling mean of the last 12 months for a sum of premium amount was added. From the business perspective, the first two features do not make much sense. The rolling mean as will be shown later has the maximum relevance in this case.

### Correlations between features

A strong correlation was found between the number of contracts and the premium amount (see Figure 5). Also, a positive correlation of the premium amount and the last adjustment of the premium amount.

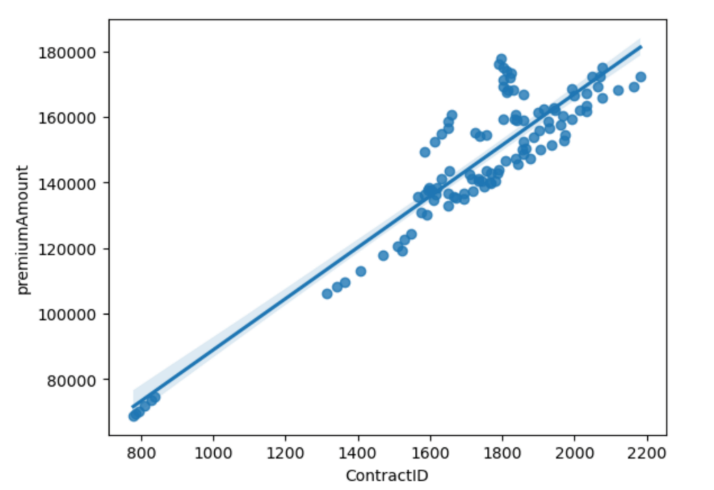


Figure 5: Correlation between the number of contracts and the premium amount.

The distribution of the variables used for a feature engineering is shown below (Figure 6).



Figure 6: Distribution of AgeatPremium and correlation with the number of contracts and the premium amount.

Most clients are between 25 and 58 years old. No obvious correlation between Age at premium and premium amount but still it can be seen that the premium amount increases with the age after 25 years.

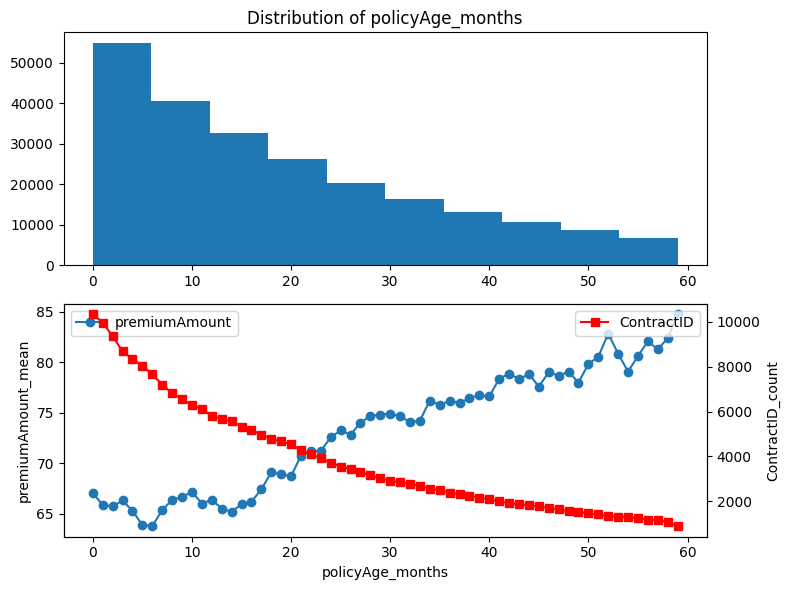


Figure 7; Distribution of policyAge\_months and correlation with ContractID and premiumAmount

The maximum contract duration is 5 years. The number of Contract ID is decreasing with the policy age increase. The premium amount, on the contrary, is growing which is obvious.

Both presented variables are taken as features for a classification model. Correlation between the created features and the target variable is shown below (see Figure 8).

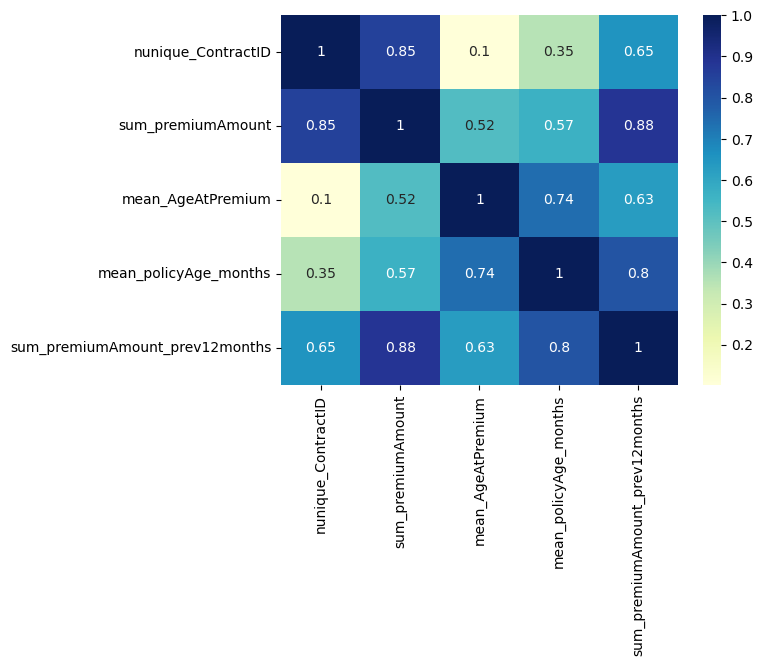


Figure 8: Correlation between policy Age, Age at premium and lag premium amount over the last twelve months.

Obviously premium amount is mostly correlated with the rolling premium amount over the last twelve months and the number of unique contract IDs. As was already mentioned, the policy age in months and the age at premium do not correlate much with the target variable. Still, they were kept for a modelling due to the lack of further relevant features.

It was decided to model the prediction of the premium amount in two different ways. On the one hand by classical time series models like SARIMA, regression models and ML models (Random Forest Regressor, XGBoost). On the other hand, the premium amount was divided into classes and predicted with classification models.

## Sales prediction with Time Series modelling

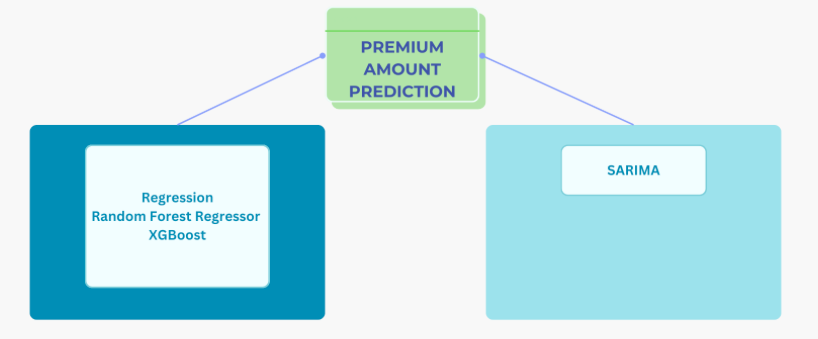


Table 1: Premium Amount prediction modelling with time series

Models and Metrics:

Four models (Multiple Linear Regression, Random Forest Regressor, XGBoost and SARIMA) were generated, one for each of the two zones. Model performance was assessed using R^2 and RSME as primary metrics, measuring accuracy and predictive quality. In the context of regression analysis, particular attention was given to examining residuals, providing insights into the goodness-of-fit and the presence of any systematic patterns or errors in the models.

Optimization with Time Series Split and Grid Search CV:

The models underwent optimization through the application of the Time Series Split method. This approach is particularly crucial for time series data as it ensures consideration of the temporal sequence during data splitting. Grid Search Cross-Validation was employed to identify the most suitable hyperparameters for the models. This involved systematically exploring the optimal parameters within a predefined parameter space.

Variation in Lags:

The number of lags, representing the count of past time points used as predictors, was systematically varied. This implies that different models considered diverse historical periods to enhance prediction accuracy. The variation in lag counts facilitated an examination of the impact of temporal dependencies within the data and allowed for determining the optimal number of past values to include.

In summary, these procedures indicate a meticulous approach to time series modeling. R^2 and RSME were chosen as robust evaluation metrics, and the models were fine-tuned through Time Series Split and Grid Search CV. The exploration of different lag counts aimed to adapt the models to the specific temporal patterns inherent in the data.

Recursive Approach for Premium Amount Forecast:

Given the unique challenges in forecasting, a recursive approach was adopted. This method involves estimating the premium amount for the next month and then incorporating this prediction back into the fitted model for subsequent forecasting iterations.

Table 2: Detailed Overview of the Applied Models to Premium Amount Prediction:\*\*

The table provides a comprehensive breakdown of the specific models used for predicting premium amounts. Each model's characteristics, such as parameters, lag considerations, and performance metrics, are detailed to offer a transparent view of the modeling process.

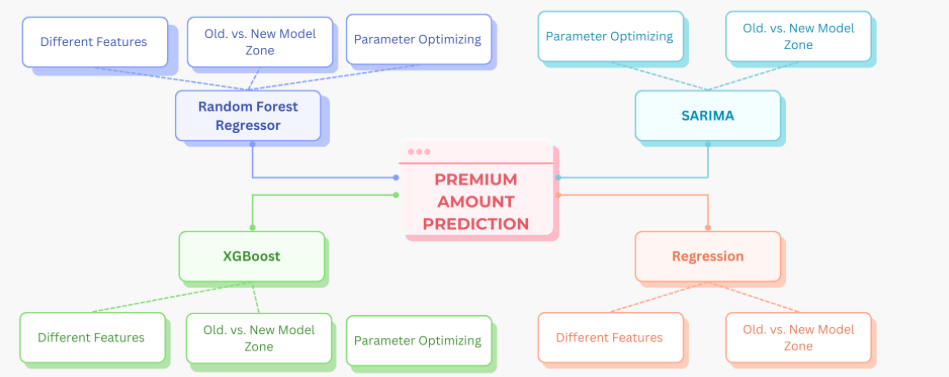


Table 2: Detailed overview of the applied models to a premium amount prediction

Interpretation of Results:

The interpretation of results poses challenges, particularly due to the substantial differences observed between the two zone models. The distinct characteristics of the old and new zone models have led to varying outcomes in the predictive performance of the applied models.

Divergent Model Performances:

Notably, the Random Forest Regressor and XGBoost models demonstrated superior performance for the strongly fluctuating patterns in the old zone model. These models excelled in capturing the complexities and variations present in this particular zone. In contrast, for the linear trends observed in the new zone model labeled "noCAN," the regression models exhibited better results. However, it's worth noting that Random Forest Regressor and XGBoost models did not yield satisfactory performance for the new zone model, indicating a sensitivity to the specific characteristics of the data.

Impact of Time Series Split:

The decision to split the time series introduced additional complexities, leading to diverse outcomes in model performance. The variations observed after splitting the time series further complicate the interpretation of results.

Example for the Old Model Zone and Multiple Linear Regression is shown below (see Figure 9, Figure 10).

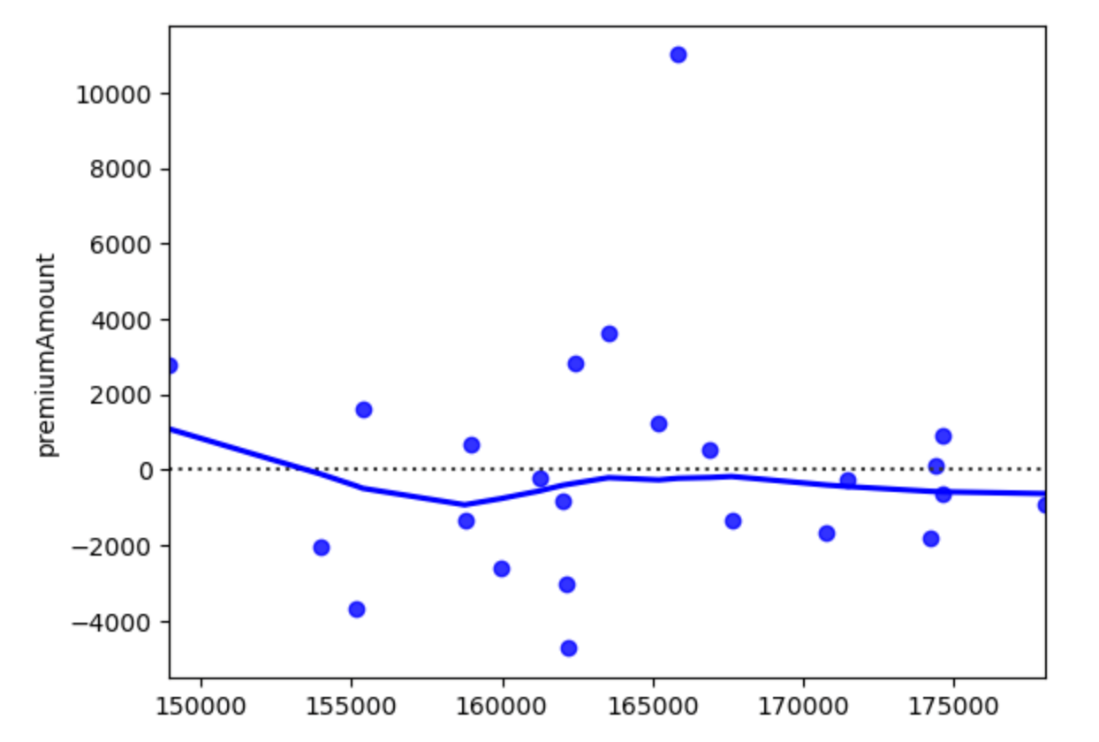


Figure 9: Premium amount for the Old Model Zone

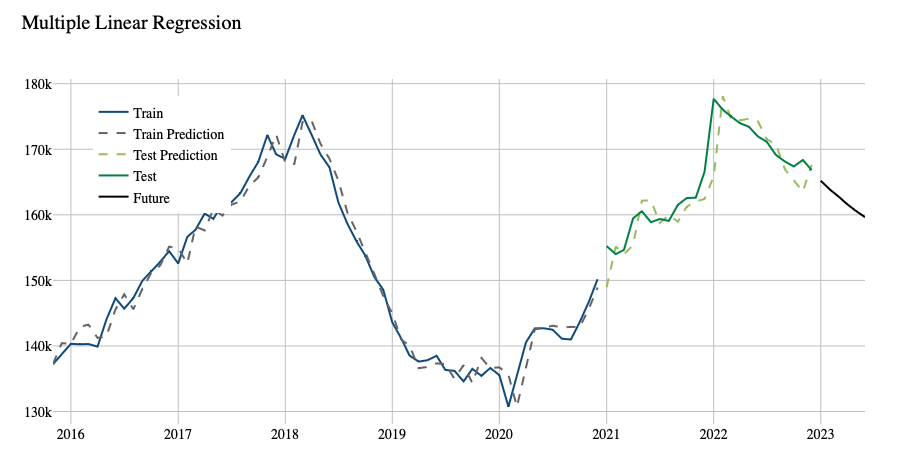


Figure 10: Multiple linear regression for the Old Model Zone

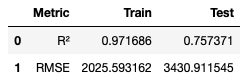


Figure 11: Metrics of the multiple linear regression for the train and test sets

The overview over the different ML-Models with Lag=12 months, Train Data from 2015-2020 and Test Data 2021-2023 and its results interpretation is collected below (see Table 3: Overview over the different ML-Models with Lag=12 months, Train Data from 2015-2020 and Test Data 2021-2023.Table 3).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Zone | Model | R^2 | RSME | Interpretation |
| Old Model Zone | Multiple Linear Regression | Train: 0.97  Test: 0.75 | Train: 2025  Test:3430 |  |
|  | Random Forest Regressor | Train: 0.99  Test: 0.78 | Train: 1089  Test:3250 | Best R^2 on Test Data |
|  | XGBoost | Train: 0.99  Test: 0.62 | Train: 85  Test:4241 | Overfit on Train data |
| New Model Zone | Multiple Linear Regression | Train: 0.99  Test: 0.77 | Train: 1299  Test:4542 |  |
|  | Random Forest Regressor | Train: 0.99  Test: -7 | Train: 611  Test:27285 | Model does not fit for the data |
|  | XGBoost | Train: 0.99  Test: -6 | Train: 75  Test:25330 | Overfit on Train data |

Table 3: Overview over the different ML-Models with Lag=12 months, Train Data from 2015-2020 and Test Data 2021-2023.

### SARIMA

The Seasonal Autoregressive Integrated Moving Average (SARIMA) method is employed to forecast premium trends. The model was built using log transformation and autocorrelation plots and optimized using parameter optimization.

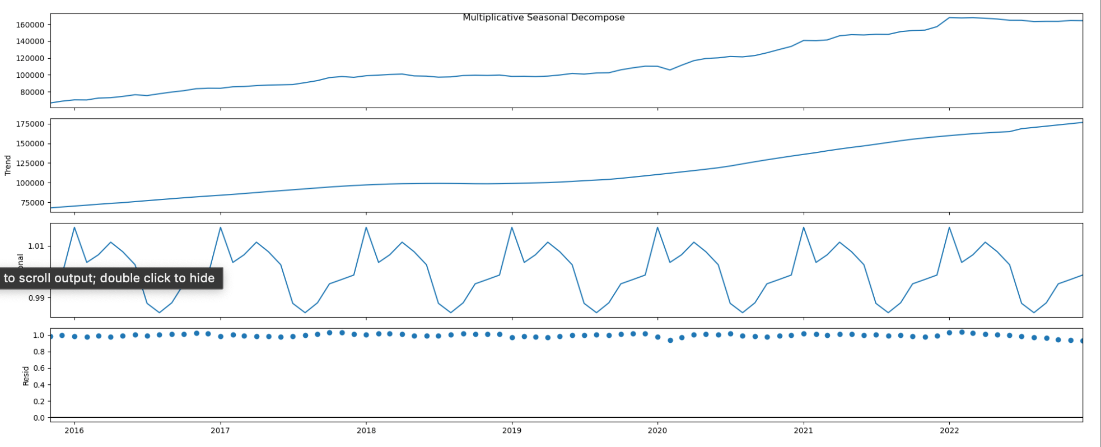


Figure 12: Modelling: Multiplicative Seasonal Decompose

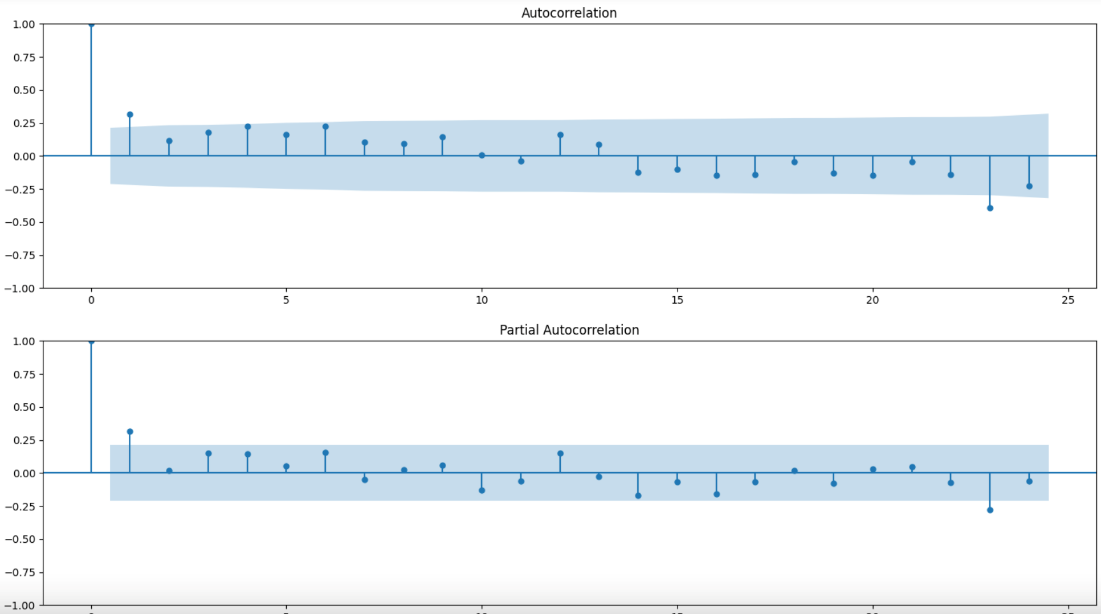


Figure 13: Sarima with Autocorrelation and Partial Autocorrelation

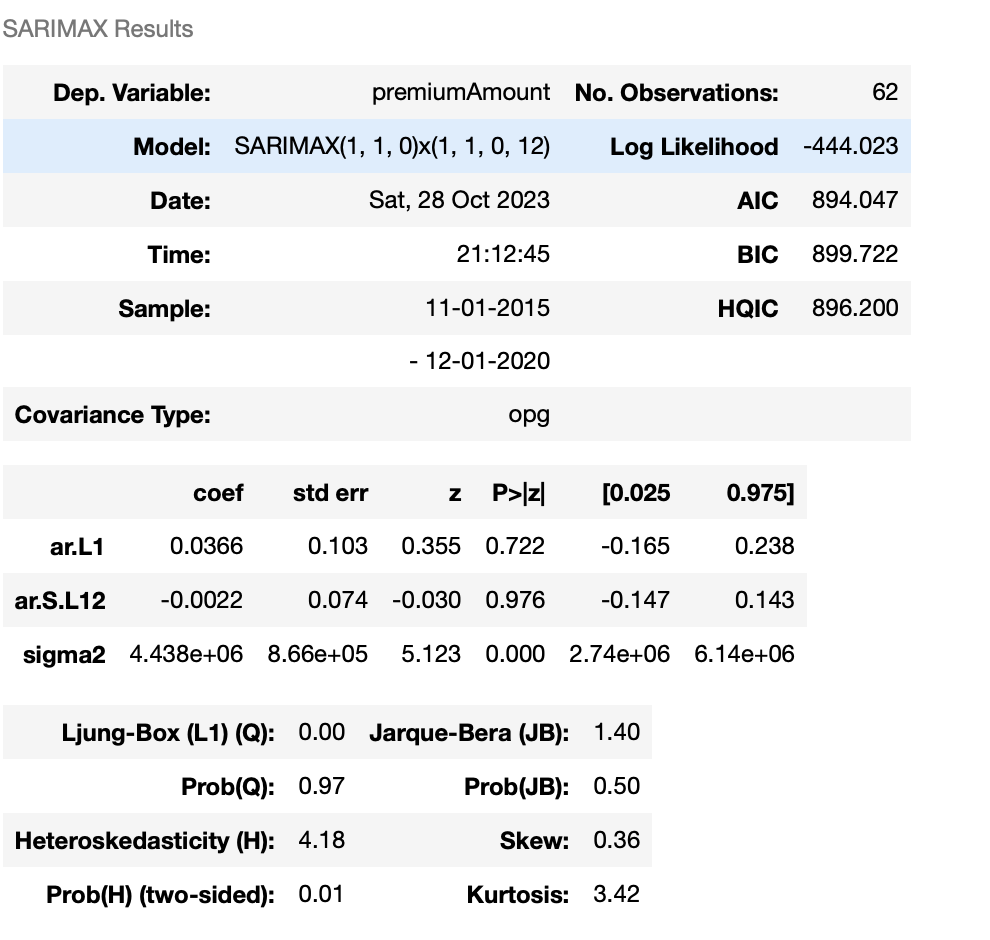


Table 4: SARIMAX results

### Conclusion

**Old Model Zone:**

In the analysis of the Old Model Zone, it was observed that the Random Forest model outperformed the other models. This conclusion is based on the evaluation metrics R^2 (coefficient of determination) and RMSE (Root Mean Square Error). The R^2 value indicates that the Random Forest model explains a significant proportion of the variance in the data, making it a robust choice for prediction in this zone. The RMSE, which quantifies the average prediction error, was also lower for the Random Forest model, further demonstrating its accuracy.

**New Model Zone:**

In contrast, for the New Model Zone, only the Multiple Linear Regression model was deemed suitable for the task. The R^2 value for this model, although not as high as in the Old Model Zone, indicates a reasonable level of explained variance. However, it is important to note that the SARIMA model, even with parameter optimization, did not produce results on a par with the Machine Learning (ML) models. The limitations of the SARIMA model may be due to the inhomogeneity of the data in this zone, which makes time series modelling less effective. The higher RMSE for SARIMA suggests that it struggled to capture the underlying patterns in the data from this particular zone.

### Considerations

When evaluating the SARIMA model, it's important to be cautious and recognise its limitations. The inhomogeneity of the data and the unique characteristics in the New Model Zone make it difficult for traditional time series models such as SARIMA to perform optimally. It's possible that further data pre-processing or alternative time series models may be required for more accurate predictions in this zone.

In summary, the choice of model should be tailored to the specific characteristics of the zone being analysed. While the Random Forest model excels in the Old Model Zone, the New Model Zone presents a different challenge, and the Multiple Linear Regression model was the most appropriate choice. The SARIMA model, although widely used for time series data, struggled to match the performance of the ML models in this context. Careful consideration of the nature of the data and the chosen evaluation metrics is crucial to model selection and interpretation of the results.

## Sales prediction with Classification modelling

## Classification of the problem

This is a classification problem, trying to predict the class of monthly premium amount summarised over all ClientIDs. It is a supervised learning model that generates rules from the training database. For this purpose, the available dataset is divided into the training and test datasets.

After looking at the sum of the premium amount frequencies (see Figure 1Figure 14), the new target variable bin\_freq was created by dividing the target into 4 bins depending on the target amount: XS, S, M, L. Mainly the target variable takes the value between 130 701 and 177 682.

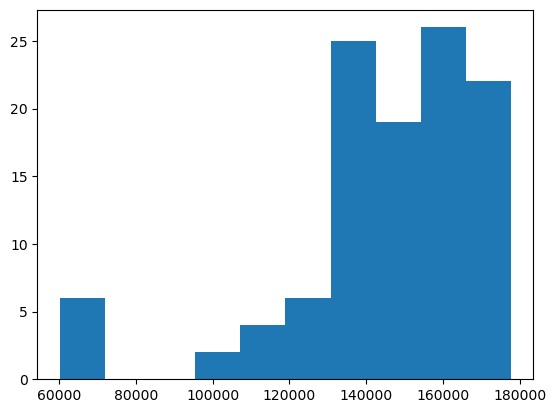


Figure 14: Frequencies distribution of the target value: sum of a premium amount.

Accuracy was chosen for a model evaluation as it is applicable to both models (KNN and SVC).

## Model choice and optimization

Two models were applied to a defined problem: K-Nearest Neighbour (KNN) and Support Vector Machine (SVM). KNN was chosen for its simplicity and often good performance on complex tasks. SVM was added to the analysis to obtain a model for comparison and to take advantage of support vectors, which provide a basis for comparison on a test sample rather than the entire training set. Other classification models were used extensively for time series modelling and churn prediction. Later, the main business obstacle associated with predicting classes in this case will be described.

In SVM modelling, the StandardScaler was used in the pre-processing step to ensure a normal distribution of the variables, independent of their original distributions.

## Interpretation of results

For the KNN-model three metrics were used: Chebyshev, Manhattan and Minkowski. The KNN model with the Manhattan metric performs the best. The confusion matrix is shown below with the k=3 (see Table 5:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class predicted** | **L** | **M** | **S** | **XS** |
| **Class real** |  |  |  |  |
| **XS** | 0 | 0 | 0 | 3 |
| **S** | 0 | 0 | 4 | 0 |
| **M** | 0 | 5 | 0 | 0 |
| **L** | 9 | 0 | 0 | 0 |

Table 5: Confusion matrix – KNN model with k=3 and metric Manhattan

To optimise performance, the modelling was run for different k-s and metrics. The accuracy scores were stored in the list for three different models (see Figure 15).

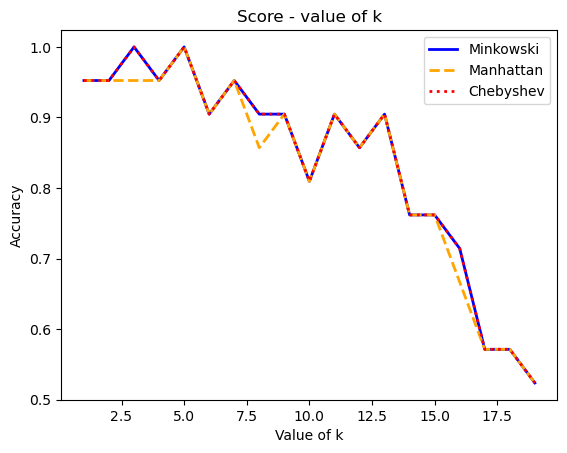


Figure 15: Score: Value of k vs. Accuracy

The highest accuracy is achieved for all metrics with the k=5 as well as for the metrics Minkowski and Chebyshev with the k=3.

Both models KNN and SVM show almost the same good results according to the accuracy score (see Table 6):

|  |  |
| --- | --- |
| **Model / Metric** | **Accuracy** |
| **KNN / Chebyshev metric** | 0.9523809523809523 |
| **KNN / Manhattan metric** | 1.0 |
| **KNN / Minkowski metric** | 0.9523809523809523 |
| **SVM** | 0.9523809523809523 |

Table 6: Accuracy score for different models

The best performing model is KNN with the Manhattan metric. However, it is important to mention the limitations of both models from an economic point of view:

* The number of features was limited and the features are not so well correlated with the target variable, except for the rolling mean of the sum\_premiumAmount of the previous 12 months.
* If the business is doing well and the sum\_premiumAmount increases each month, then the predicted class will always be L (Large) in the future. The model would need to be adjusted periodically.
* In real life, the business objective is to predict the exact amount, not just the class or range of amounts. With the range it is then difficult to make business plans including cost planning.

Therefore, the use of both models is not sufficient from a business point of view and it is recommended to consider the time series for the sales volume and churn prediction models.

# Churn prediction

## Data collection & Description

Data was collected directly from the contracts in SAP. This includes features like nationality, age, start- & end dates, as well as the termination date, which is filled, if the customer decided to end the contract. See Figure 43: Target & Feature variables in the ERP System. [[1]](#footnote-2)

This information is used for the target variable and is set to 1 if it is filled in. It’s important to note that this is not always a sign of dissatisfaction but can be caused by various reasons, such as returning to the home country. That’s why the termination reasons (Beendigungsgrund) will be considered later to select only contracts that were terminated for specific reasons as an alternative target variable ***ds\_terminated***.

To add more information about past customer behaviour, information from the transaction rows were added. They contain information about premiums paid as well as customer invoices claimed and are grouped and linked in SQL by their contractID. These steps can be comprehended in Figure 44 - Figure 46.[[2]](#footnote-3)

To get an overview about the collected data, column descriptions can be found in Error: Reference source not found.

In the analysis and preprocessing, particular attention was given to columns such as "mean\_payoutDays" and the ratios of paid out vs. claimed invoices, which were anticipated to be major factors of customer satisfaction. Unfortunately, this data includes **a high number of missing values**. One of many reasons that resulted in substantial preprocessing work which is described below.

## Pre-processing, Visualization & Dependencies

### Target variable

The target variable has been defined as 1 if the contract contains a terminationDate, meaning that the customer wanted to terminate the contract before a possible endDate.

Looking only at terminated contracts, the ratio of terminated contracts is extremely high (see Figure 16):

Ein Bild, das Text, Kreis, Screenshot, Diagramm enthält.

Automatisch generierte Beschreibung

Figure 16: Target distribution

After internal feedback, it became clear that there could be a number of reasons for this, and that a cancelled contract wasn't always the result of disappointed customers. E.g., some products don’t have a maximum duration and therefore tend to have a higher customer termination ratio as you can see in the following figure (see Figure 18):

A graph with blue and orange bars

Description automatically generated

Figure 17: MainProduct distribution + target ratio

That is why an alternative target variable was created after looking at the specific termination reasons and their distribution (see Figure 18):

Ein Bild, das Text, Screenshot, Farbigkeit, Schrift enthält.

Automatisch generierte Beschreibung

Figure 18: Termination reasons and their distribution

The variable ‘ds\_terminated’ is created within the pre-processing function and is set only set to 1 for specific termination reasons (default: 10014 - 10016). It can be selected as an alternative target within the train-test-split function.

Choosing this variable as a target will, depending on the reasons chosen, most often result in highly unbalanced data, as can be seen below for the default values (see Figure 19):

Ein Bild, das Kreis, Diagramm, Screenshot, Text enthält.

Automatisch generierte Beschreibung

Figure 19: Distribution of alternative target variable

### Handling Missing Values

In particular, there were a large number of missing values for variables containing information on the last year (see Figure 20):

Ein Bild, das Text, Schrift, Reihe, Screenshot enthält.

Automatisch generierte Beschreibung

Figure 20: Top NULL-Columns

This makes sense, as contracts that have ended more than a year ago will not contain any data here. That is why more features have been created in SQL to link active and closed contracts:

1. A reference date was created to compare the last active year of active and terminated contracts.
2. Sums and averages over the last active year were created.

LastYear columns have been removed from the Python preprocessing steps and replaced with lastActivYear columns.

In the preprocessing function other NULL values were depending on their content and dtype either

* replaced by mean (e.g. mean\_payoutDays)
* replaced by 0 (e.g. sum\_payout)
* Replaced by certain string value to avoid errors (e.g. ‘XX’ for countries and ‘None’ for terminationReasons)
* column was dropped (e.g. product\_code)

In some cases, company internal feedback was needed to clarify whether the number of missing values made sense and how to handle them meaningfully.

### Outliers

Due to the large number of features the columns were split by their data type for outlier detection. If outliers were detected, the preprocessing function was adjusted accordingly.

Depending on the columns content:

1. The column was dropped as new features make it redundant (e.g. lastYear-columns after adding lastActivYear-columns)
2. The column was replaced by another column to avoid high correlations (e.g. sum\_payout\_lastActivYear by payout\_ratio\_lastActivYear)
3. Outliers were dropped (e.g. sum\_claimed-columns)

This way the top 10 numerous outlier columns were thus pre-processed (see Figure 21, Figure 22):

Ein Bild, das Text, Zahl, Schrift, parallel enthält.

Automatisch generierte Beschreibung

Figure 21: Top 10 initial outlier columns

Ein Bild, das Text, Zahl, parallel, Schrift enthält.

Automatisch generierte Beschreibung

Figure 22: Top 10 initial outlier columns after preprocessing

### Correlations between features

High correlation was found especially between (see Figure 23):

1. Dates around policy\_startDate
2. Product specific characteristics
3. Claimed & payout sums

Ein Bild, das Text, Screenshot, Diagramm, parallel enthält.

Automatisch generierte Beschreibung

Figure 23: Top feature correlations

This was put into the pre-processing function in the way that:

1. Apply- & SignDate were dropped. Instead, date difference between Apply- & startDate was considered.
2. Product columns can be selected to be dropped when product information is merged.
3. Redundant claim columns have been dropped: e.g., ‘sum\_retained’ (= sum\_claimed – payout\_sum) &

Instead of absolute values, ratios of payout to claimed amount are calculated to avoid high correlation of payout- & claimed amount:

This way the amount of highly correlated columns may be reduced.

### Correlations with target variable

To compare all features with the target variable, some modifications had to be made to the df:

* Encode string/ categorical columns
* Convert datetime columns to int by keeping only the year

In the end, the most important correlating features are those that give direct information about the current status of the contract (active / ended) (see Figure 24):

Ein Bild, das Text, Screenshot, Schrift, Reihe enthält.

Automatisch generierte Beschreibung

Figure 24: Top correlations with target variable

In addition, correlations of the initial and pre-processed df were compared. Further variations of correlation comparison can be found in the notebook.

### Distributions

The high correlations with the target variable can also be visualised by looking at the distribution of these values with hue=’terminated’ (see Figure 25):

Ein Bild, das Diagramm, Reihe, Screenshot enthält.

Automatisch generierte Beschreibung

Figure 25: effEndDate distribution

There are many contracts with an endDate of 2100 - an internal date for “infinit” - that have many products. In order to be able to reduce these "outliers", 2 more parameters have been added to the preprocessing function to optionally cut the effEndDate to a specific date.

This allows a closer look at the distribution of the main columns (see Figure 26):

Ein Bild, das Diagramm, Reihe, Screenshot, Design enthält.

Automatisch generierte Beschreibung

Figure 26: Distribution of effEndDate after cutting max to 31-12-2023 (default)

It can clearly be seen that the effEndDate has a high influence on the target variable, as (almost) all terminated contracts have an effEndDate in the past. This correlation is much lower when the alternative target variable is chosen (see target distribution).

Another interesting distribution is the MainProductCode (see Figure 27):

Ein Bild, das Diagramm, Screenshot enthält.

Automatisch generierte Beschreibung

Figure 27: MainProduct distribution

While the most common MainProducts have a balanced distribution of the target variable, there are common MainProducts with a positive (G014) or negative (G004) impact on the termination status. Looking only at ended contracts can change the ratio completely. Example: G014; overall, there are 19% terminations, within ended contracts it’s 70%. This shows in particular once again the main impact of the effEndDate, telling if a contract is still active or not.

### Encoding

To create numerical data for the use of ML Models the data can be encoded with the help of predefined functions.

2 Encoder were chosen:

1. CountFrequency:

This encoder has been chosen to reduce the feature amount. To avoid data leakage, this encoder will be used in the process / after train-test-split. To re-transform the encoded data, an encoder is returned by the create\_train\_test function.

The disadvantage of this encoder is, that it can give a false impression because nominal data is interpreted as ordinal data. That is why another semi-manual encoding has been created:

1. Dummy Encoding

This encoder can also be used prior to generating train and test data, avoiding data leakage and maintaining nominal data. [[3]](#footnote-4)

The disadvantage is the large number of features to be created. Some additional things have been done to reduce this:

* Values of categorical columns with only a small number of rows are dropped (e.g. MainProducts with less than 5 contracts).
* Countries and nationalities are grouped by their regions using a REST countries API within the country\_to\_region\_mapping function.

### Scaling

Due to large differences in features distributions, the data can be optionally scaled.

**MinMaxScaler** was chosen as the scaler as it is an easy to use and understand scaling function that maintains the ratios and should work well in most cases. Since the distributions above showed enough examples without Gaussian distribution (e.g. policy\_effEndDate, MainProductCode) the use of the standardization is omitted.[[4]](#footnote-5)

### Final preprocessing parameters

Finally, some pre-processing parameters have been defined. Depending on the values of these parameters the predefined functions are switched in/out and filled with input parameters:

* year\_only: bool, default=False
  + Set to True to keep only the year of all datetime features and convert them to int. Otherwise year and month are separated and kept.
* Drop\_cols: list of strings, default = []
  + Try to drop inserted strings. If col name does not exist or has already been dropped during regular preprocessing process, a corresponding message is printed.
* claim\_ratios: bool, default=True
  + If set to True, some claim-related columns will be dropped or replaced and cleaned up to minimise correlated columns. Specifically, retained columns are dropped, payout amount columns are cleaned and replaced with a ratio of the claimed amount.
* cut\_effEnd: bool, default=False
  + Set to True to cut policy\_effEndDate values at a specific cut\_date to tighten distribution.
* cut\_date: datetime, default = ‘2030-12-31’
  + Optional, if cut\_effEnd == True. All policy\_effEndDate values > cut\_date will be replaced by this value.
* add\_products: bool
  + Set to True to add product characteristics to the contract df based on the product\_code.
* product\_drop\_cols: list of strings, default = []
  + Optional, if add\_products == True, product columns can be selected to be dropped before merging.
* save\_csv: bool
  + Set to True to save the preprocessed df as a csv file in the 'preprocessed’ subfolder.
* filename: string, default = ‘contracts\_preprocessed'
  + Optional, if save\_csv == True. Filename can be set to the name of the selected parameter options.

These parameters are given as preprocessing options in streamlit to create different transformations of the initial df and use them for modelling.

## Churn prediction modelling

### Classification of the problem

Churn prediction is a binary classification problem. Supervised learning methods were chosen to make predictions. Initially, the main goal was to predict (the probability) whether a currently active contract will be cancelled by the customer in the near future. From a business perspective, this would mean identifying contracts at risk of cancellation in order to take timely action and minimise termination rates.

Over time, it became apparent that it was difficult to define such a target variable with a time component for the historical data. Instead, it was decided to use the existing variable to predict whether a contract would be terminated by the customer based on the features. Combined with highlighting the key global and individual features that lead to a positive prediction using Shap, this can still provide insights into identifying active contracts with a higher risk of termination.

The results of the modelling can be visualised using a confusion matrix (see Table 7: Confusion Matrix descriptionTable 7):

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **PREDICTED** | |
| **Contract Terminated (1)** | **Contract Non-terminated (0)** |
| **TRUE** |  | TRUE POSITIVE (TP) | FALSE NEGATIVE (FN) |
| **Contract Terminated (1)** | Model correctly predicts that the contract is terminated by the customer. | The model predicts that the contract is still active or has ended naturally but it is terminated. |
|  | FALSE POSITIVE (FP) | TRUE NEGATIVE (TN) |
| **Contract Non-terminated (0)** | The model predicts that the contract has been cancelled, but it is still active or has ended naturally. | The model correctly predicts that the contract is still active or has ended naturally. |

Table 7: Confusion Matrix description

An FP error means that the model incorrectly predicts that a contract will be terminated by the customer. This would result in a “false warning”. On the other hand, an FN error would result in missing a terminated contract and therefore a missed opportunity to counteract the termination. Therefore, the primary choice is to **minimise the FN rate** (type 2 error) in order to maximise the number of terminations detected.

So, while a good Precision score within the positive class (1) is important to avoid wasting capacity on controlling too many false alarms in practice, the Recall of the positive class should be ranked even slightly higher to avoid undiscovered terminations.

To keep track of both – precision and recall - the main metric chosen is the **f1-score**. An alternative metric could be the f2-score to put more weight to recall.[[5]](#footnote-6)

### Model choice and optimization

Initially DecisionTree was chosen as the first option in order to start with an easily interpretable model. When SHAP was discovered as an interpreter, it was replaced by the XGBClassifier algorithm to maximise performance. SupportVectorClassifier was chosen as an addition approach with solid results but too long execution times. RandomForestClassifier was found to be a good alternative but caused the kernel to hang several times when used with Shap. So the decision was made to use **XGBClassifier** for most of the further steps as a good, fast and interpretable model.

Without setting any parameters, it gave solid results right from the start on the ‘terminated’ target (see Figure 28).

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Figure 28: First results of xgboost on 'terminated' and 'ds\_terminated'

### Alternative target ds\_terminated

For the alternative target variable, the first results were terrible. The F1-score on the test data was 0. Adjustments of the preprocessing parameters only resulted in only insignificant changes. The main reason seems to be the high imbalance of the data between class 0 and 1, especially within the test data, when splitting the data by date.

Attempts were made to deal with the imbalance using RandomOverSampler() with GridSearch within a pipeline with the XGBClassifier, as well as a GridSearch on the 'scale\_pos\_weight' parameter of the XGBClassifier, but neither resulted in a higher F1 score.

Instead, a custom function was written to reweight the df by an input factor (see Figure 29):Ein Bild, das Text, Screenshot, Reihe, Zahl enthält.

Automatisch generierte Beschreibung

Figure 29: Resampling ds\_terminated

Looping over different resampling factors increased F1 from 0(!) to at least 0.34 on the test set (Figure 30):

Ein Bild, das Text, Reihe, Diagramm, Screenshot enthält.

Automatisch generierte Beschreibung

Figure 30: Finding best Resampling ratio to increase F1

Using resampled data for parameter optimisation with GridSearch and CV resulted in further improvements up to an F1 score of 0.5 on the test data (see Figure 31).

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Figure 31: Best XGBClassifier on resampled data

The results look good, but should be treated with caution, as not only the training data but also the test data were resampled and, due to the reduced imbalance, e.g. a random selection would also have led to higher results. One consequence of the resampling was, for example, that the base score of the test data in SHAP increased, which means that class 1 is generally assigned a higher probability (see Figure 32, Figure 33).

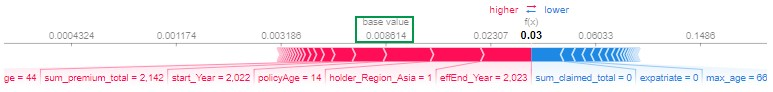


Figure 32: SHAP example of test data before resampling

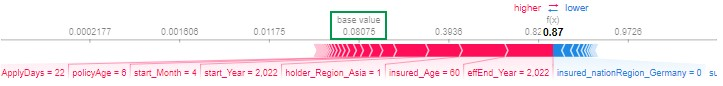


Figure 33: SHAP example of test data after resampling

### Model comparison on ‘terminated’ target

The first modelling approach was to use a simple DecisionTreeClassifier. The results were surprisingly good and could be improved by finding the best max\_depth parameter (see Figure 34, Figure 35):

Ein Bild, das Text, Reihe, Diagramm, Zahl enthält.

Automatisch generierte Beschreibung

Figure 34: Best DecisionTreeClassifier

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Figure 35: Main Features of DecisionTreeClassifier

The main influencing factors are, as expected, contract end dates. Removing them from the data and focusing on other features led to poorer results.

To create a more realistic test set, the training and test data were separated by policy\_startDate rather than randomly to use more recent contracts as test data (see Figure 36). For further comparisons, DecisionTree was replaced by XGBoost and SupportVector and RandomForest were added.

A graph of a number of years

Description automatically generated with medium confidence

Figure 36: Train & test data split by start date

Preprocessing variations and hyperparameter tuning using GridSearch and CV were attempted to improve modelling results. This resulted in mostly insignificant improvements. That's why the results of the best GridSearch model were compared with the results of the default parameter model, and the better model was selected. Surprisingly, this was mainly the default model (see Figure 37).

The evaluation metrics chosen were not only the F1 score on training and test data, but also the execution time for the model to fit and predict.

Ein Bild, das Text, Screenshot, parallel, Schrift enthält.

Automatisch generierte Beschreibung

Figure 37: Model comparison

As seen in the figure above, the models all appear to overfit the train data and perform similarly on the test data with F1 scores between 0.65 and 0.67. SVC has a massively longer run time which made it difficult to use GridSearch on this model.

### Interpretations with SHAP & Feature reduction

SHAP was used to explain the poorer performance on the test data, as well as to find the most important features and individual stopping reasons. As described above, there were problems when trying to use SHAP with SVC and RandomForest. That's why the focus is on XGBoost.

To decompose the results, a function was defined that first uses SHAP to pick out the most important features, reduces the training and test data to these features, and then trains and scores the model again using only these data. The results using XGBoost and only the top 5 features are almost as good as using all features (see Figure 38).

Ein Bild, das Text, Screenshot, Diagramm, Reihe enthält.

Automatisch generierte Beschreibung

Figure 38: Shap results for xgb with top 5 features only

As one can see, along with effEnd\_Year, the policy age is a major feature. As the train and the test data have been split by startDate, the values of this feature are distributed quite differently for the train and test data. The SHAP values show that the effect of the policyAge on the model output is different for the train and test data. This may be one reason why the test score is much worse than the train score.

Waterfall plots of SHAP were made for individual contracts to check the key features of individual decisions. If there had been more time, the main effects of incorrectly predicted contracts could have been investigated.

### Predicting Probabilities

To achieve the additional goal of predicting the top n active contracts with the highest probability of cancellation, probabilities were calculated instead of classifications. Therefore, the XGBoostClassifier with the top 10 main features (xgb\_top10) was chosen in order to consider more features, but also to keep the results more interpretable.

The evaluation of the predictions was done using the lift curve and the cumulative gain curve to compare the true values with the predicted termination probabilities (see Figure 39).

Ein Bild, das Text, Reihe, Diagramm, Screenshot enthält.

Automatisch generierte Beschreibung

Figure 39: Lift Curve & Cumulative Gain of xgb\_top10

In particular, on the test data, the model performs very well in predicting Class 1. If we trust the Cumulative Gain Curve, using only the top 40% of contracts based on their churn probability will be sufficient to detect all churning contracts.

From a business perspective, this would mean that we can reduce the number of contracts to look at to find active customers who are more likely to churn. If it is possible to understand the individual factors for a high likelihood of churn, it may be possible to take an action to retain the customer.

Therefore, SHAP was used again to create waterfall plots of individual contracts to show the main impact on the prediction. Looking only at active contracts in the test set, effEnd\_Year remains the most important feature. Most of the contracts with the highest probability of termination have an effEnd\_Year of 2023, meaning that the contract is about to end or has just ended. Due to the large number of terminations within terminated contracts, this explains the large impact. See example below (see Figure 40).

A graph of numbers and a graph

Description automatically generated with medium confidence

Figure 40: SHAP waterfall plot of xgb\_top10 - individual example

To be clear about the assumption, it helps to look at the distribution of effEnd\_Year of the training & test data (see Figure 41). Obviously, it's enough to predict the top 37% with the lowest effEnd\_Year as 'terminated' to get all 17% of terminated contracts within the test set.

A graph of different sizes and colors

Description automatically generated with medium confidence

Figure 41: effEnd\_Year distribution of train & test set

It becomes clear that the model relies too much on the effective end date. After dropping all columns related to effEndDate, other features such as start year and paid premiums become more important but cannot predict terminations well anymore, so the F1 score on the test set drops to 0.36 (see Figure 42). Looking at the SHAP values, we can at least make some interesting discoveries, such as the fact that a high total amount of premiums paid has a negative effect on the cancellation probability (presumably because they are long-standing customers), while a high amount of premiums in the last active year has a positive effect (presumably because the premium sometimes became too high for the customer).

A graph of different colored lines

Description automatically generated with medium confidence

Figure 42: SHAP values for XGBClassifier without effectiveEndDate

Finally, an attempt was made to improve the F1 scores by adjusting the classification threshold. But again, no improvement could be achieved.

### Conclusion

The results of the models are quite solid. In particular, XGBoostClassifier predicts fast, reliable and easy to interpret results using SHAP. With an F1 score of ~0.66 on the latest contracts (X\_test) and a higher recall than precision, the majority of cancelled contracts are detected, but a large number of false positive predictions are also included. Most of the improvement approaches such as GridSearch & CV, preprocessing variations, threshold adjustments & different train test data generation could not improve the results significantly. However, it must also be said that the number of possible constellations was too large to test all variants.

The addition of probabilities made it possible to take a closer look at active contracts with a high cancellation probability. However, the effective end date remained the most influential and therefore no further patterns of termination behaviour could be discovered. A more in-depth analysis of the errors in the test set would therefore have been helpful.

The creation of an alternative target variable that specified the target in terms of specific termination reasons resulted in data that was too unbalanced to achieve good results. Resampling and hyperparameter tuning could at least improve them from an F1 score of 0 to ~0.5 on the test set.

In conclusion, the results are interesting but not very valuable from a business perspective as we could not create a target variable to predict the probability of future termination, only the current termination status. Also, the model relied too heavily on the actual termination date and could not reliably predict based on other patterns.

Therefore 3 main further steps could help to improve possible knowledge:

1. A way to generate data to predict future termination probabilities based on current & past data.
2. Build pipelines with custom functions to iterate faster through many preprocessing variations.
3. Deeper analysis of false classifications on the test set using SHAP to find patterns for decisions.

However, even if this is successful, it should be remembered that it is difficult to identify such patterns, as these are always human decisions and a large proportion of the actual motives will not be reflected in the available data.

## CHALLENGES & CONCLUSION

The main challenge was that the project group was working with real data from a system that real people work with (& make mistakes from time to time). In addition, the ERP system had changed during the period of data collection, making some older data less reliable. Therefore, the validity of the data could never be relied upon 100%. Jonathan, in particular, spent most of his time explaining and preparing the data in both SQL and Python. This time was particularly lacking later in the modelling part.

In addition, a lack of knowledge about the capabilities of ML modelling at the start of the project led to vague objectives that were largely unachievable with the existing data. In the sales project, there was a lack of known information about the future development of the features. For the churn part, only a target value in the sense of "currently terminating" instead of "will terminate" could be predicted.

Another difficulty faced by the project group was the lack of skills related to working in a remote group on a data science project. The group underestimated the time needed to delegate tasks, especially in the first part of a project: data exploration and visualisation. There was also a lack of theoretical background on how best to delegate tasks in such projects. It is highly recommended that a data science project management course be added as a mandatory module at the beginning, to provide in-depth skills in this area. In terms of structure, the group would have personally benefited from a deeper understanding of the project and an assessment of the objectives on the part of DataScientest, as well as regular progress checks and requests for interim reports at the appropriate times to keep the project group on track.

In addition, the large Jupyter notebook files became a challenge, especially later in the project. Due to the lack of knowledge and structure of the notebooks, it was often necessary to restart the kernel and run everything in order to continue modelling, which often caused the kernel to hang and took a lot of time. This was at least partially optimised in the Churn project towards the end. E.g. by including variables & conditions and saving & loading variables using joblib.

In the end, therefore, the scientific and business-relevant findings were limited by the factors described above and could only be achieved to a limited extent. Nevertheless, interesting results and findings were obtained, which can be read in particular in the conclusions of the subprojects.

# Bibliography

Many articles, websites and YouTube channels were used. Here is a selection:

* <https://www.mckinsey.com/capabilities/operations/our-insights/ai-driven-operations-forecasting-in-data-light-environments>
* <https://thecleverprogrammer.com/2021/05/19/sales-prediction-with-machine-learning/>
* <https://towardsdatascience.com/5-machine-learning-techniques-for-sales-forecasting-598e4984b109>
* [https://medium.com/aiskunks/categorical-data-encoding-techniques-d6296697a40f#:~:text=It%20refers%20to%20the%20process,with%20text%20or%20categorical%20variables](https://medium.com/aiskunks/categorical-data-encoding-techniques-d6296697a40f" \l ":~:text=It refers to the process,with text or categorical variables)
* <https://towardsdatascience.com/time-series-forecasting-with-arima-sarima-and-sarimax-ee61099e78f6>
* <https://dev.to/balapriya/cross-validation-and-hyperparameter-search-in-scikit-learn-a-complete-guide-5ed8>
* <https://www.simplilearn.com/normalization-vs-standardization-article>
* <https://neptune.ai/blog/evaluation-metrics-binary-classification>
* <https://www.kaggle.com/code/robikscube/tutorial-time-series-forecasting-with-xgboost>

# Appendices

### Figures & tables

Ein Bild, das Text, Screenshot, Software, Webseite enthält.

Automatisch generierte Beschreibung

Figure 43: Target & Feature variables in the ERP System

Ein Bild, das Text, Software, Webseite, Computersymbol enthält.

Automatisch generierte Beschreibung

Figure 44: Claim & Premium variables in the ERP System

↓

Ein Bild, das Text, Screenshot, Schrift, Software enthält.

Automatisch generierte Beschreibung

Figure 45: Feature Engineering of premium- & claims-data

↓

Ein Bild, das Text, Screenshot, Zahl, Schrift enthält.

Automatisch generierte Beschreibung

Figure 46: Example of premium- & claims-data

|  |  |  |
| --- | --- | --- |
| **Column** | **Datatype** | **Description** |
| BirthDate | datetime64 | Customer birth date |
| Nation | Object | Nationality of Insured Person |
| AgeAtPremium | int64 | Age of Insured Person at PremiumMonth [Years] |
| PolicyAgeAtPremium | int64 | Age of Contract at PremiumMonth [Years] |
| policy\_StartDate | datetime64 | start of contract |
| policy\_EffEndDate | datetime64 | effective end of contract possibly infinite (NULL OR 31.12.2099) |
| premiumAmount | float64 | target-value: paid premiumAmount [€] (aggregated by Product,CmpPrivate, Deductible, Zone, Time) |
| FeeAmount | float64 | Fee that BDAE as a broker gets from the premium amount (from the insurance company) |
| feeRate | object | Fee in % |
| ContractID | object | contract number, personal customers ID |
| product\_code | object | code of insurance product |
| MainProductCode | object | code of main insurance product group |
| MainProductName | string | name of main insurance product group (pseudonymised) |
| Model | int64 | Zone Model |
| Zone | int64 | Zone of Country |
| Zone Desc | string | Zone description |
| product\_code | string | code of insurance product |
| MainProductCode | string | code of main insurance product group |
| MainProductName | string | name of main insurance product group (pseudonymised) |
| premium\_Country | string | Insured country |
| premium\_CountryName | string | Insured country |
| product\_group | int64 | Kind of product |
| product\_groupName | string | Description of product group |
| premiumMonth | datetime64 |  |
| Model\_Zone | string | The zone name and number |
| Nation\_region | string | Region for a nation of an insured person |
| premiumCountry\_region | string | Region for the insured country |
| policy\_startMonth | datetime64 | Start month of a contract |
| policyAge\_months | int64 | Number of months since contract was signed |
| pA\_lastDate | datetime64 | Last date of premium adjustments to the contract |
| pA\_currentPremium | float64 | Premium amount at the moment |
| pA\_lastPremium | float64 | Premium amount before adjustments |
| pA\_last\_Change | float64 | The difference after premium adjustment |
| pA\_new\_model | float64 | New model after the pA |
| time\_since\_last\_pA | float64 | Time since last pA in months |

Table 8: Sales prediction column descriptions

|  |  |  |
| --- | --- | --- |
| **Column** | **Datatype** | **Description** |
| ContractID | string | pseudonymised unique ContractID (index / primary Key) |
| policy\_startDate | date | start Date of contract (Filtered to >= 01.01.2017) |
| policy\_initialEndDate | date | initial end Date at beginning of contract |
| policy\_effEndDate | date | effective end Date of contract (due to earlier cancellation/…) |
| update\_Date | date | date of last data extraction from SQL database |
| RefDate | date | reference date to compare last year of active (-> update\_Date) and ended contracts (-> effEndDate) |
| activ | boolean | if contract is still activ or already ended |
| ApplyDate | date | date of contact application by the customer |
| SignDate | date | date of contract signing by the customer |
| paid\_until | date | date until premium is paid for this contract |
| terminationDate | date | date of contract termination by the customer |
| terminationReason | string / categorical | reason for contract termination |
| **terminated** | boolean | if contract got terminated by the customer (**target variable**) |
| product\_code | string | code of insurance product |
| MainProductCode | string | code of main insurance product group |
| MainProductName | string | name of main insurance product group (pseudonymised) |
| insured\_birthDate | date | birthday of insured person |
| insured\_Gender | string | gender of insured person |
| insured\_nationality | string | nationality of insured person |
| holder\_country | string | country of contract holder |
| expatriate | boolean | if insured person is expatriate |
| additional\_insurance | boolean | if insured person has an additional insurance |
| num\_claims\_total | integer | total number of claims (invoices) handed in by the customer |
| sum\_claimed\_total | float | total amount of money claimed by the customer |
| sum\_payout\_total | float | total amount of money paid out to the customer |
| sum\_retained\_total | float | part of claimed money that did not get paid out |
| payout\_ratio\_total | float | total ratio of paid out vs. claimed money (payout/claimed) |
| mean\_payoutDays | integer | mean waiting time in days between claim and payout of money |
| sum\_premium\_total | float | total amount of paid premiums by the customer |
| num\_claims\_lastYear | integer | same as above, but for last year before update\_Date |
| sum\_claimed\_lastYear | float |
| sum\_payout\_lastYear | float |
| sum\_retained\_lastYear | float |
| payout\_ratio\_lastYear | float |
| sum\_premium\_lastYear | float |
| mean\_payoutDays\_lastYear | integer |
| num\_claims\_lastActivYear | integer | same as above, but for last year before RefDate  (activ contracts --> last year / ended contracts --> last year before effEnd) |
| mean\_payoutDays\_lastActivYear | integer |
| sum\_payout\_lastActivYear | float |
| sum\_claimed\_lastActivYear | float |
| sum\_retained\_lastActivYear | float |
| sum\_premium\_lastActivYear | float |

Table 9: Churn prediction column descriptions

### Code

* See github repo: <https://github.com/JonathanPablo/DataScientest_Sales-Churn_Project>

1. Screenshots from the ERP-System and Date collection in SQL can be found in the appendix. [↑](#footnote-ref-2)
2. More information about the collection and preprocessing in SQL can be found in the sql-files in github. [↑](#footnote-ref-3)
3. See: [https://medium.com/aiskunks/categorical-data-encoding-techniques-d6296697a40f#:~:text=It%20refers%20to%20the%20process,with%20text%20or%20categorical%20variables](https://medium.com/aiskunks/categorical-data-encoding-techniques-d6296697a40f" \l ":~:text=It refers to the process,with text or categorical variables) [↑](#footnote-ref-4)
4. See: <https://www.simplilearn.com/normalization-vs-standardization-article> [↑](#footnote-ref-5)
5. See: <https://neptune.ai/blog/evaluation-metrics-binary-classification> [↑](#footnote-ref-6)