

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 2](#_Toc149860673)

[About the company and the product 2](#_Toc149860674)

[Background 2](#_Toc149860675)

[Contribution 3](#_Toc149860676)

[Objectives 3](#_Toc149860677)

[Data 3](#_Toc149860678)

[Framework 3](#_Toc149860679)

[II. Sales prediction 5](#_Toc149860680)

[III. 1 Relevance 5](#_Toc149860681)

[III. 2 Pre-processing and feature engineering 5](#_Toc149860683)

[Correlations between features 7](#_Toc149860684)

[III. 3 Sales prediction with Time Series modelling 11](#_Toc149860685)

[SARIMA 13](#_Toc149860686)

[Conclusion 14](#_Toc149860687)

[Considerations: 15](#_Toc149860688)

[III. 4 Sales prediction with Classification modelling 16](#_Toc149860689)

[III. Churn prediction 17](#_Toc149860690)

[III. 1 Data collection & Description 17](#_Toc149860691)

[III. 2 Pre-processing, Visualization & Dependencies 17](#_Toc149860692)

[Target variable 17](#_Toc149860693)

[Handling Missing Values 19](#_Toc149860694)

[Outliers 20](#_Toc149860695)

[Correlations between features 22](#_Toc149860696)

[Correlations with target variable 22](#_Toc149860697)

[Distributions 23](#_Toc149860698)

[Encoding 24](#_Toc149860699)

[Scaling 25](#_Toc149860700)

[Final preprocessing parameters 25](#_Toc149860701)

[III. 3 Churn prediction modelling 26](#_Toc149860719)

[Classification of the problem 26](#_Toc149860720)

[Model choice and optimization 27](#_Toc149860721)

[Alternative target ds\_terminated 28](#_Toc149860722)

[Model comparison on ‘terminated’ target 29](#_Toc149860723)

[Interpretations with SHAP & Feature reduction 31](#_Toc149860724)

[Predicting Probabilities 32](#_Toc149860725)

[Conclusion 34](#_Toc149860726)

[Bibliography 36](#_Toc149860727)

[Appendices 36](#_Toc149860728)

[Figures & tables 36](#_Toc149860729)

[Code 39](#_Toc149860730)

# 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. Johnathan 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 2 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.

Withing this sub-project another 2 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 Fig.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 (see chapter xy) and classification models (see chapter xy).

## 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 Fig.2). Ein Bild, das Text, Diagramm, Reihe, Zahl enthält.

Automatisch generierte Beschreibung

Figure 22: 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.

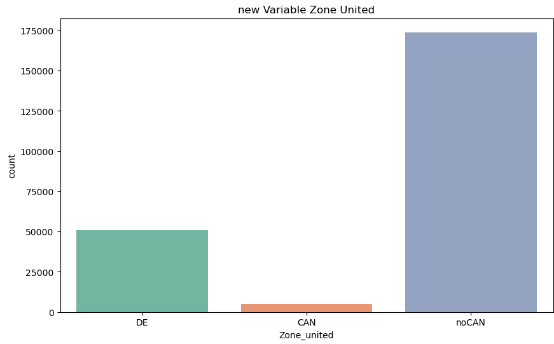
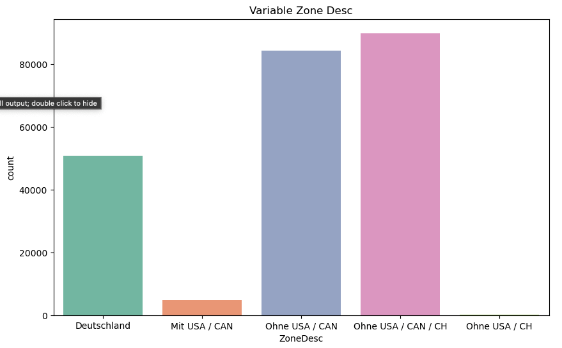


Figure 3: Count of unique Contract IDs by Zone: New Zone model vs. Old Zone model

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.

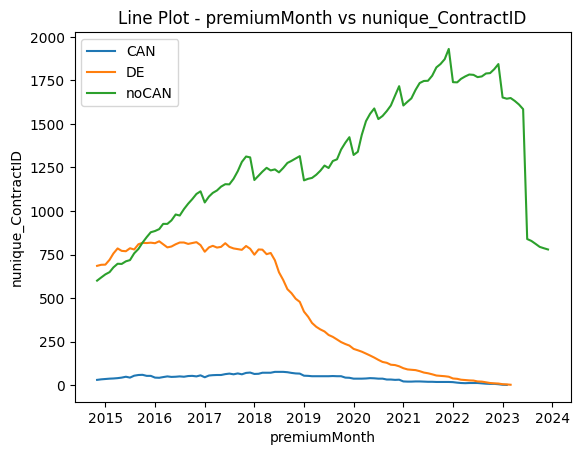


Figure 4: Count of unique Contract IDs by Zone: New Zone model vs. Old Zone model

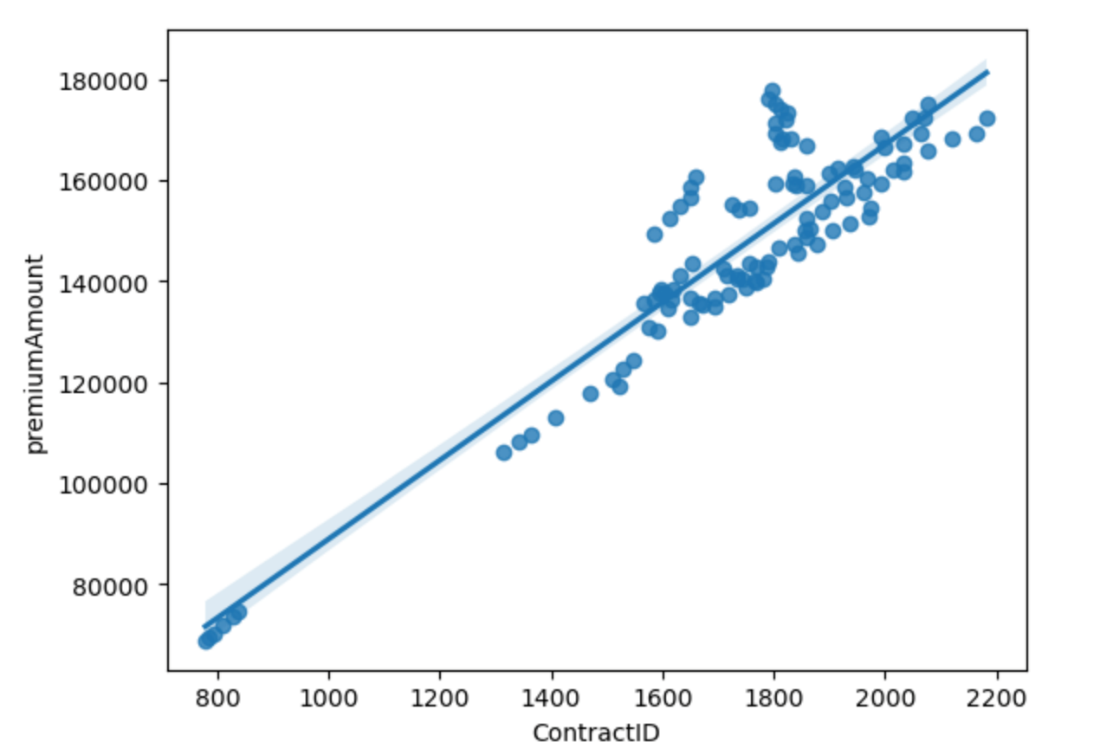
Dummies were created for the categorical variables and other variables such as the number of days since the last adjustment. It turned out that these variables are not suitable for the Time Series models.

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 1). Also, a positive correlation of the premium amount and the last adjustment of the premium amount.



The distribution of the variables used for a feature engineering is shown below.



Figure 5: Count of unique Contract IDs by Zone: New Zone model vs. Old Zone model

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.

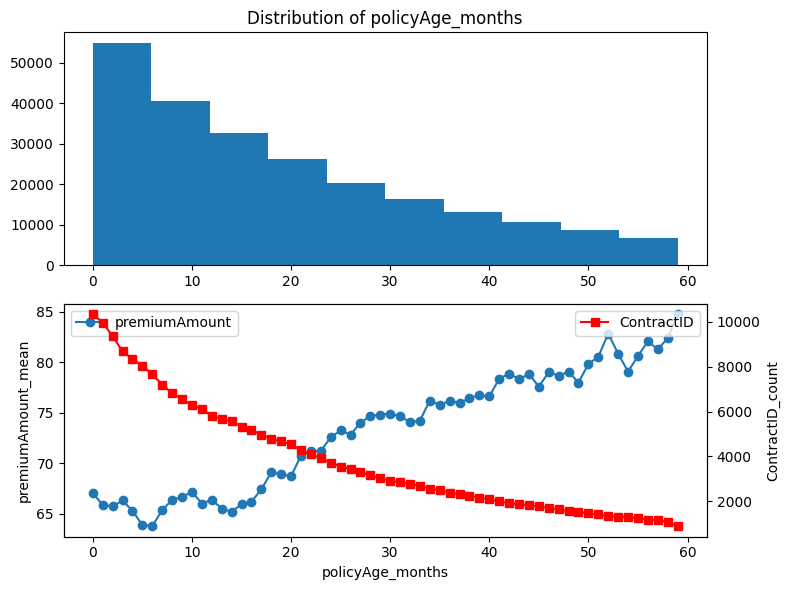


Figure 6: Count of unique Contract IDs by Zone: New Zone model vs. Old Zone model

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 7).

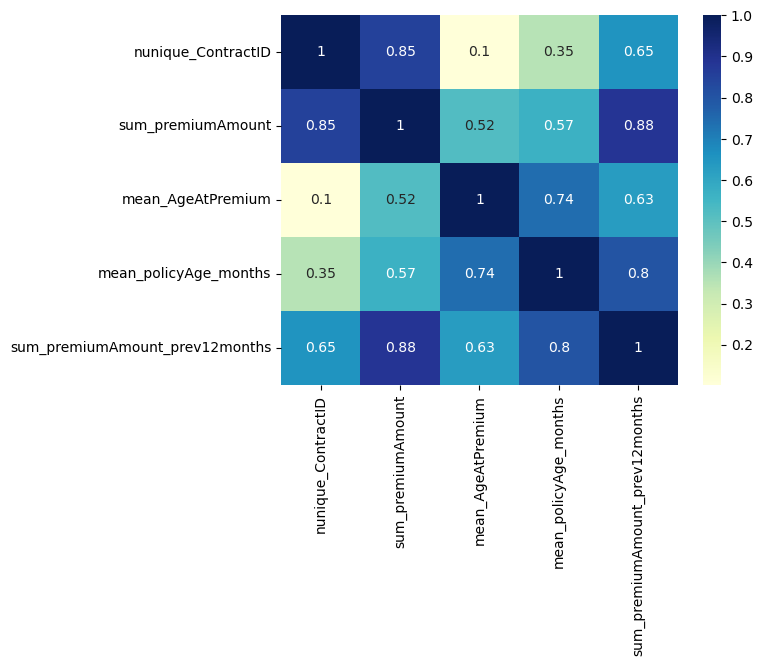


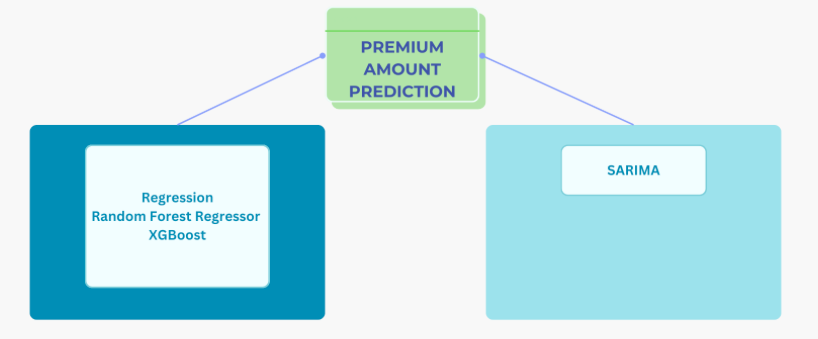
Figure 7: Correlation between policy Age, Age at premium and lag premium amount over the last twelwe months.

Obviously premium amount is mostly correlated with the rolling premium amount over the last twelwe 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). The Jupyter Notebook by Rob Mulla https://www.kaggle.com/code/robikscube/tutorial-time-series-forecasting-with-xgboost was used as a basis and adapted. On the other hand, the premium amount was divided into classes and predicted with classification models.

Finally, we had a dataset grouped by month with lag features over the last twelve months. In addition, temporal factors such as the year, quarter were also included.

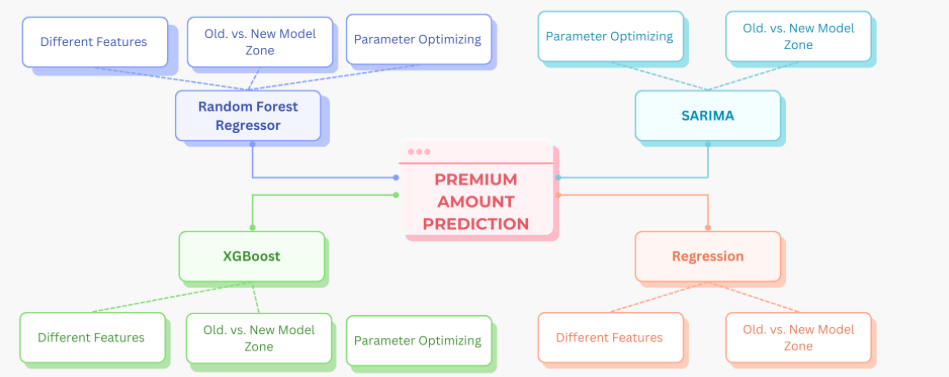
## Sales prediction with Time Series modelling



Four different models were calculated for the two zone models. The R^2 and RSME were used as metrics. The models were optimized using Time Series Split and Grid Search CV. Furthermore, a different number of lags was tested.

The Forecast presented a special challenge. A recursive approach was chosen. The premium amount of the following month is estimated and transferred back to the fitted model.

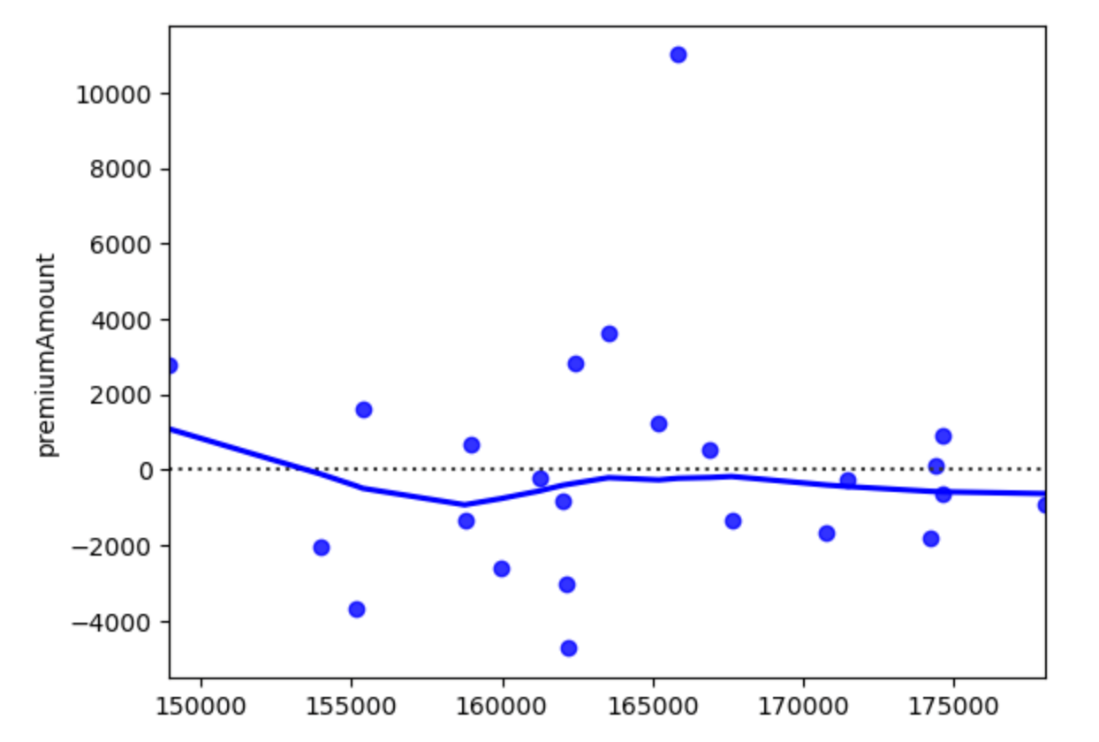
Advanced Models were not used.

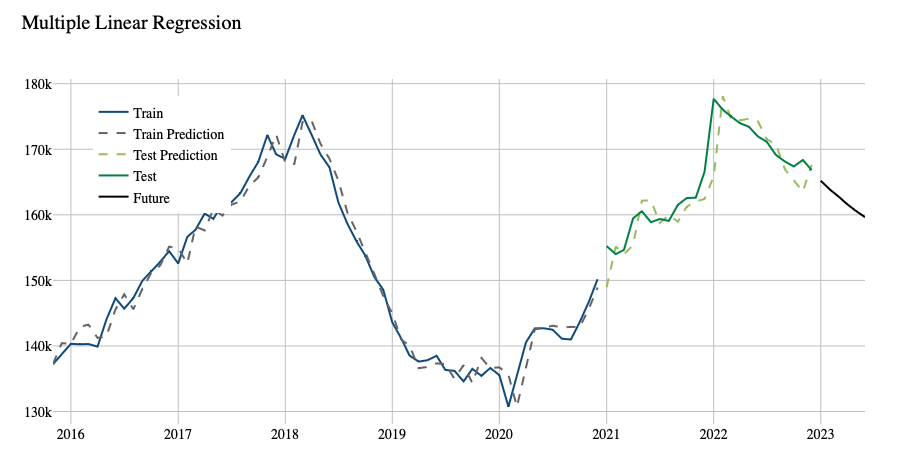


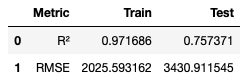
The interpretation of results is difficult. The two zone models produced very different results. The Random Forest Regressor and XGBoost models were clearly better for the strongly varying courses in the old zone model. For the linear course of the new zone model "noCAN" the regression models showed better results. Random Forest Regressor and XGBoost do not work for the new model Zone.

Splitting the time series also produced very different results, making interpretation even more difficult.

Example for the Old Model Zone and Multiple Linear Regression:







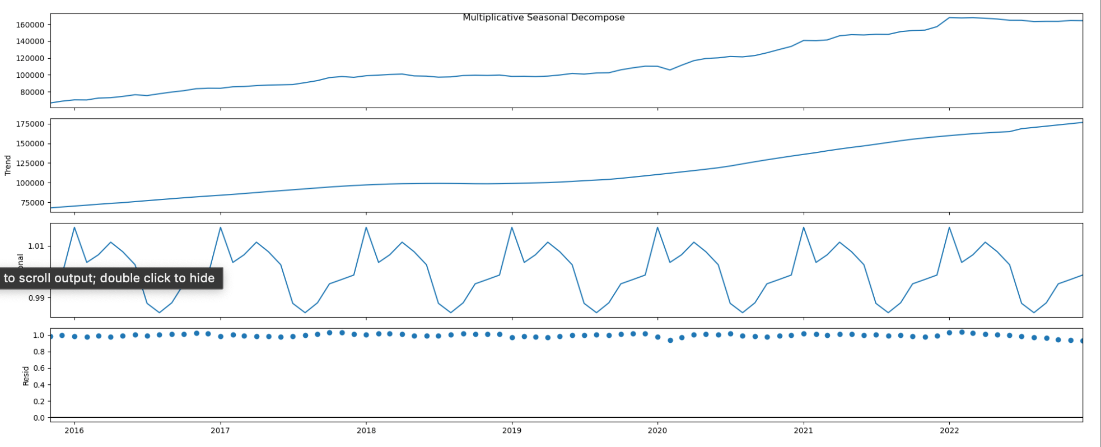
Overview over the different ML-Models with Lag=12 months,

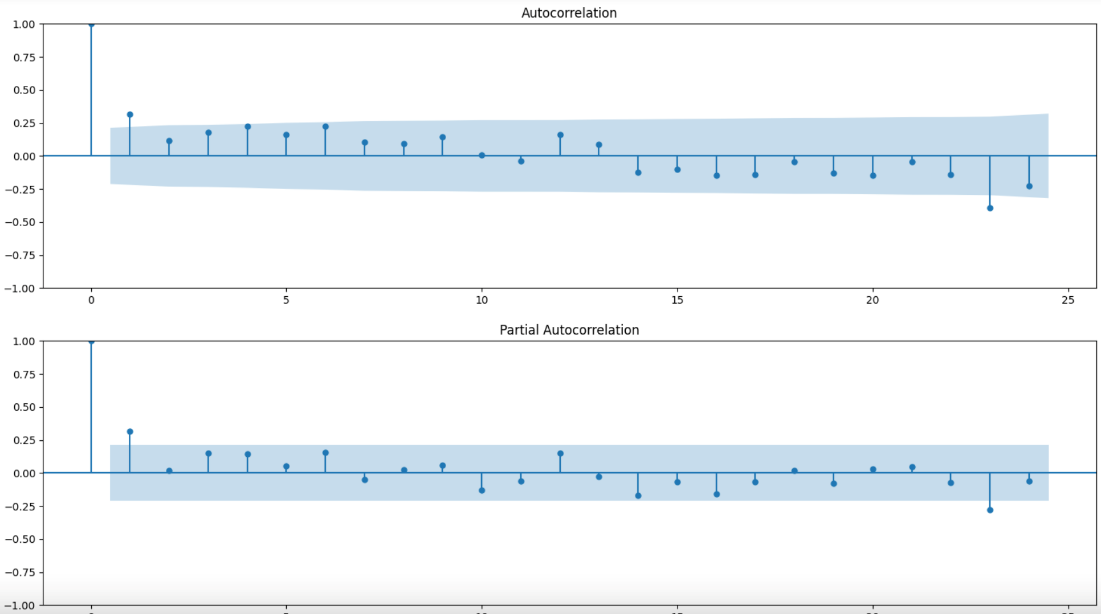
Train Data from 2015-2020 and Test Data 2021-2023

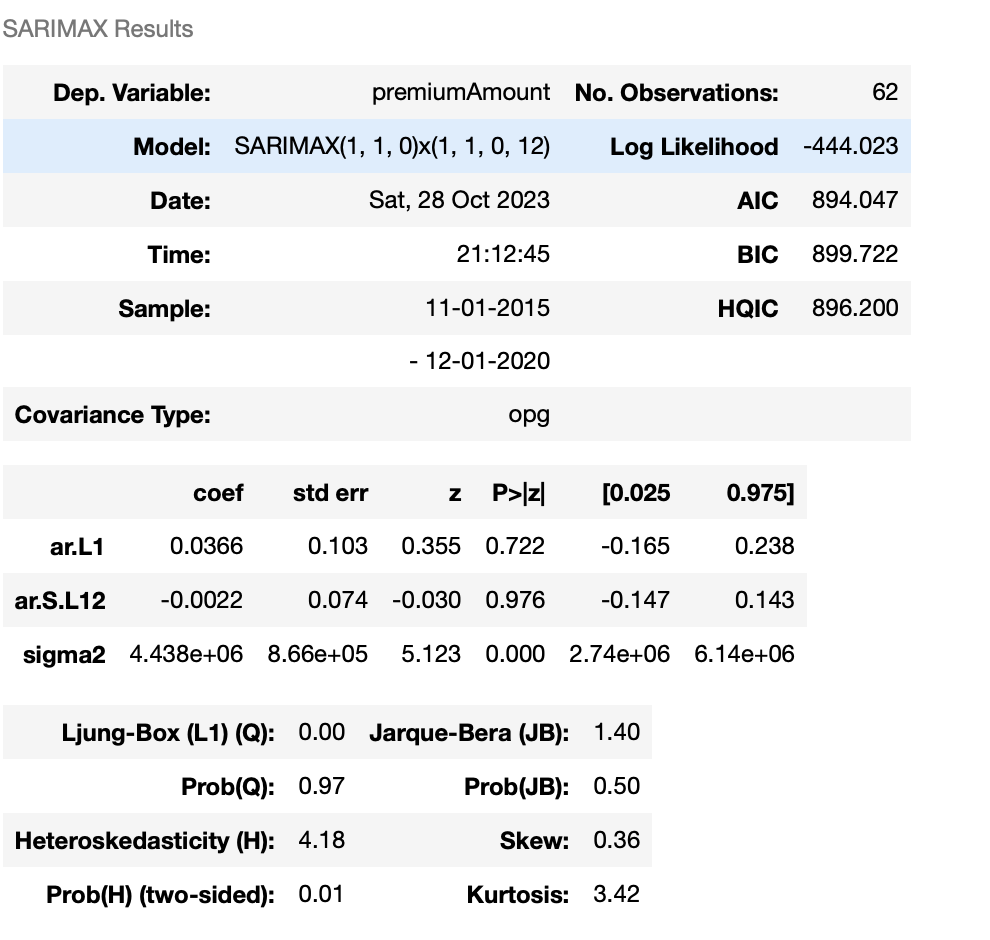
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Zone | Model | R^2 | RSME | Interpretation |
| Old Model Zone | Multiple Linear Regression | Train: 0.97  Test: 0.75 | Train: 2025  Test:3430 |  |
|  | Random Forest Regressor | Train: 0.99  Test: 0.78 | Train: 1089  Test:3250 | Best R^2 on Test Data |
|  | 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 |

### 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.







### 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 portion 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, while not as high as in the Old Model Zone, indicates a reasonable level of explained variance. However, it's crucial to note that the SARIMA model, even with parameter optimization, did not yield results on par with the Machine Learning (ML) models. The SARIMA model's limitations may stem from the inhomogeneity of the data in this zone, which makes time series modeling less effective. The higher RMSE for SARIMA suggests that it struggled to capture the underlying patterns in this particular zone's data.

### Considerations:

When evaluating the SARIMA model, it's important to exercise caution and recognize the limitations. The data's inhomogeneity and unique characteristics in the New Model Zone make it challenging for traditional time series models like SARIMA to perform optimally. It's possible that further data preprocessing 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 analyzed. 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 suitable 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 data's nature and the chosen evaluation metrics is crucial for model selection and interpretation of results.

## Sales prediction with Classification modelling

# 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 4: 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. This steps can be comprehended in Figure 5 - Figure 7.[[2]](#footnote-3)

To get an overview about the collected data, column descriptions can be found in Table 2: Churn prediction column descriptions.

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:

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

Automatisch generierte Beschreibung

Figure 8: 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:

A graph with blue and orange bars

Description automatically generated

Figure 8.1: MainProduct distribution + target ratio

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

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

Automatisch generierte Beschreibung

Figure 9: 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:

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

Automatisch generierte Beschreibung

Figure 10: 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:

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

Automatisch generierte Beschreibung

Figure 11: 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 beed 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 weresplit 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:

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

Automatisch generierte Beschreibung

Figure 12: Top 10 initial outlier columns

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

Automatisch generierte Beschreibung

Figure 13: Top 10 initial outlier columns after preprocessing

### Correlations between features

High correlation was foundespecially between:

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 14: 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):

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

Automatisch generierte Beschreibung

Figure 15: 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’:

Ein Bild, das Diagramm, Reihe, Screenshot enthält.

Automatisch generierte Beschreibung

Figure 16: 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:

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

Automatisch generierte Beschreibung

Figure 17: 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:

Ein Bild, das Diagramm, Screenshot enthält.

Automatisch generierte Beschreibung

Figure 18: 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 isinterpreted 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 is 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 ases. 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 where 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 client 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 unsing a confusion matrix:

Table 1: Confusion Matrix description

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **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. |

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. That is why the primary choice is to **minimize the FN rate** (type 2 error) in order to maximize the number of detected terminations.

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 emphasis on 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 selected as addition approach with solid results but too long execution times. RandomForestClassifier was found to be a good alternative but has led to a hang up of the kernel multiple times when using it with Shap. So the decision was to go with **XGBClassifier** for most of the further steps as a good, fast and interpretable model.

Without setting any parameters, it gave solid results from the start on the ‘terminated’ target.

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

Automatisch generierte Beschreibung

Figure 19: 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 insignificant changes. The main reason seems to be the high imbalance of the data between class 0 and 1. Especially inside the test data, when splitting data by date.

It was tried to handle 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 both did not result in a higher F1 score.

Instead, a custom function was written to rebalance the df by an input factor:Ein Bild, das Text, Screenshot, Reihe, Zahl enthält.

Automatisch generierte Beschreibung

Figure 20: Resampling ds\_terminated

Looping over different resampling factors resulted in an increase of F1 from 0(!) to at least 0.34 on the test set:

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

Automatisch generierte Beschreibung

Figure 21: Finding best Resampling ratio to increase F1

Using resampled data for parameter optimaisation with GridSearch and CV resulted in further improvements up to an F1 score of 0.5 on the test data.

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

Automatisch generierte Beschreibung

Figure 22: Best XGBClassifier on resampled data

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.

### Model comparison on ‘terminated’ target

The first modelling approach was using a simple DecisionTreeClassifier. The results were surprisingly good and could be even increased by finding the best max\_depth parameter:

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

Automatisch generierte Beschreibung

Figure 19: Best DecisionTreeClassifier

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

Automatisch generierte Beschreibung

Figure 20: Main Features of DecisionTreeClassifier

Main influencing factors are – as expected – the contracts endDates. Dropping them from data and focus on other features led to poorer results.

To create a more realistic test set, train- & test data got separated not randomly but by policy\_startDate to use more recent contracts as test data. For further comparisons DecisionTree got replaced by XGBoost and supplemented by SupportVector and RandomForest.

Preprocessing variations and hyperparameter tuning using GridSearch and CV got tried to improve modelling scores. Resulting in mainly only insignificant improvements. That’s why finally the results of the best GridSearch model got compared with results for model with default parameters and the better model was chosen. This was surprisingly mainly the default model.

As evaluation metrics not only F1-score on train- & test data was chosen, but as well the execution time for the model to fit & predict.

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

Automatisch generierte Beschreibung

Figure 21: Model comparison

As you can see in the figure above the models all seem to be overfitted to the train data and perform similar on the test data with F1 Scores between 0.65 and 0.67. SVC has a massively longer execution time, which made it hard to use GridSearch on this model.

### Interpretations with SHAP & Feature reduction

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

To break down 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 evaluates the model again with this data only. The results with XGBoost and only the top 5 features are almost as good as for using all features.

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

Automatisch generierte Beschreibung

Figure 22: Shap results for xgb with top 5 features only

As you can see - besides effEnd\_Year - the policy age is one main feature. Since train- & test- data got split by startDate, the values of this feature are distributed totally different for train- & test-data. The SHAP values show that the impact on model output of policyAge is different for train- & test data. That should be one reason why the test score is much worse than the train score.

Waterfall plots of SHAP were created for single contracts to check most important features of individual choices. If there would have been more time, investigations on main impacts of falsely predicted contracts could have been done.

### Predicting Probabilities

To reach the additional goal of predicting top n active contracts with the highest probability of termination, probabilities got calculated instead of classifications. Therefore, XGBoostClassifier with top 10 main features (*xgb\_top10*) was chosen to look at some more features but on the other hand keep the results easier interpretable.

Evaluation of the predictions were made using lift curve and cumulative gain curve to compare true values vs. predicted termination probabilities.

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

Automatisch generierte Beschreibung

Figure 23: Lift Curve & Cumulative Gain of xgb\_top10

Especially on the test data the model performs very good on predicting class 1. If we trust the Cumulative Gain Curve, using only the top 40% of contracts based on their termination probability will be enough to detect all terminating contracts.

From a business perspective this would mean, that we can reduce the number of contracts to look at, to find active customers that are more likely to terminate their contract. If it is possible to understand individual factors for a high termination probability, it might be possible to take actions to keep the customer.

Therefore, SHAP was used again to create waterfall plots of individual contracts, to show main impacts on the prediction. Looking only at active contracts of the test set, the effEnd\_Year remains most important feature. Most of the contracts with highest termination probability have an effective End Date in 2023, meaning that the contract will end soon or did just end. Because of the large number of terminations within ended contracts it explains the big impact. See example below.

A graph of numbers and a graph

Description automatically generated with medium confidence

Figure 24: SHAP waterfall plot of xgb\_top10 - individual example

To be clear about the assumption a look at the effEnd\_Year distribution of train & test data helps. Obviously, it’s enough to predict the top 37% with 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 25: effEnd\_Year distribution of train & test set

So, it becomes clear that the model relies to much on the effective end date. After dropping all columns related to effEndDate other features like start year and paid premiums become more important but cannot predict terminations good anymore, so that the F1-Score on the test set drops to 0.36. Looking at SHAP values at least some interesting discoveries could be made, like the fact that a high total sum of paid premium has a negative impact on termination probability (presumably because they are longstanding customers), while a high premium amount in the last activ year has a positive impact (presumably because the premium sometimes got too high for the customer).

A graph of different colored lines

Description automatically generated with medium confidence

Figure 26: SHAP values for XGBClassifier without effectiveEndDate

Finally, an attempt was made to improve the F1 scores by adjusting the threshold of the classification. But here, too, no improvements could be achieved.

### Conclusion

The results of our models are kind of solid. Especially XGBoostClassifier predicts fast, reliable, and easy to interpret results using SHAP. With an F1-Score of ~0.66 on the most recent contracts (X\_test) and a higher recall then precision, the majority of terminated contracts are getting detected, but a large amount of False Positive predictions is included as well. Most of improvement approaches like GridSearch & CV, preprocessing variations, threshold adjustments & different train-test data creation could not improve the scores significantly. However, it must also be said that the number of possible constellations was too large to test all the variants.

Adding probabilities has made it possible to take a closer look at active contracts with high termination probability. But still the effective End Date remained most impactful and therefore no further patterns of termination behaviour could be discovered. Therefore, a deeper analysis of errors on the test set could have been helpful.

Creating an alternative target variable that specifies the target to specific termination reasons led to too imbalanced data to achieve good results. By resampling and hyperparameter tuning they could at least be improved from a F1-Score of 0 to ~0.5 on the test set.

To conclude, the results are interesting, but from a business perspective not extremely valuable, since we could not create a target variable to predict probability of future termination, but only of current termination status. As well our model relied too much on the effective End date and could not predict reliable based on other patterns.

Therefore 3 main further steps could help improving possible knowledge:

1. A way to create data to predict future termination probabilities based on current & past data.
2. Building pipelines including self-defined functions, to go faster trough 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 not be forgotten that it is difficult to recognise such patterns, as these are always human decisions and a large part of the actual motives will not be reflected in the available data.

1. CHALLENGES & CONCLUSION

The main challenge was the fact that the project group was working with real data from a system that real people work with (& make mistakes from time to time). Additionally, the ERP System had been changed during the period of collected data, making some older data less reliable. Therefore, the validity of the data could never be relied upon 100%. Jonathan, in particular, therefore 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, it was a lack of known information about the features development in the future. For the churn part, only a target value in the sense of “currently terminating” could be predicted instead of “will be terminated”.

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 divide the tasks, especially in the first part of a project: data exploration and visualisation. Furthermore, theoretical background was missing on how task delegation works best in such projects. It is highly recommended that a *data science project management course* will be added as a mandatory module at the beginning to provide the in-depth skills in this area. In terms of structure, the group would have been benefited personally by a deeper familiarisation with the project and an assessment of the objectives on the part of DataScientest, as well as a regular check on progress and a request for interim reports at the required time to keep the project group on track.

Furthermore, the large Jupyter notebook files became a challenge, especially later in the project. Due to the lack of knowledge and structure of the notebooks, the kernel usually had to be restarted and everything had to be executed to continue working on the modelling, which often led to the kernel hanging and consumed a lot of time. This could be at least partially optimised in the Churn project towards the end. E.g., by incorporating variables & conditions and saving & loading variables with the help of joblib.

So in the end, scientific and business-relevant findings were limited by the described factors and only came about to a limited extent. Nevertheless, interesting results and findings were obtained, which can be read in particular in the conclusions of the sub-projects.

# Bibliography

A lot of articles, websites as well as 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://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 4: Target & Feature variables in the ERP System

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

Automatisch generierte Beschreibung

Figure 5: Claim & Premium variables in the ERP System

↓

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

Automatisch generierte Beschreibung

Figure 6: Feature Engineering of premium- & claims-data

↓

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

Automatisch generierte Beschreibung

Figure 7: Example of premium- & claims-data

Table 2: Churn 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 |

### 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> [↑](#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)