NOVA IMS Information Management School

MDSAA

Master Program in

Data Science and Advanced Analytics

Customer Churn Prediction in Auto Insurance

Predicting policy cancelation for new clients

Ricardo de Sá Nunes

Internship Report

presented as a partial requirement for obtaining the Master's degree in Data Science and Advanced Analytics

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

CUSTOMER CHURN PREDICTION IN AUTO INSURANCE

by

Ricardo de Sá nunes

Internship report presented as a partial requirement for obtaining the Master's degree in Data Science and Advanced Analytics, with specialization in Data Science

Supervisor: Prof. Mauro Castelli

July 2023

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism, any form of undue use of information, or falsification of results along the process leading to its elaboration. I further declare that I have fully acknowledged the Rules of Conduct and Code of Honor from the NOVA Information Management School.

Ricardo de Sá Nunes

Lisboa, July 2023

ACKNOWLEDGMENTS

Firstly, I must thank Prof. Mauro Castelli for accepting to supervise this project, enabling a successful conclusion of this master's program.

I would like to express my deep gratitude to Raquel Albuquerque, who provided guidance throughout this project and whose willingness to assist others is inspiring. I thank you for all your support and patience. The crucial insights and exemplary contributions led to higher achievements.

To Telmo Vieira, I am very grateful for sharing his geodesic knowledge that served as guidance through a new field that had been unexplored. The unquestionable effort and availability to help are truly regarded.

To Diana Montrond, who constantly showed support and shared the inner workings of the insurance market, I am forever thankful.

To my supporting family, that encouraged me to push through this step in my academic life.

Finally, I would like to thank all the colleagues who contributed in various forms, engaged in discussions of different approaches, and shared their knowledge.

ABSTRACT

Customer churn presents a challenge for any company, including insurers. Knowing which customers are more likely to cancel their insurance policy, enables the company to proactively take action to prevent such churn. To better describe the customers' behavior and environment, new features can be created from external data. This project used data spanned from January 2019 to March 2023 from the auto branch of an insurance company. Given the availability of geospatial data, two new variables were added that help to portray the customer's exposure to insurance intermediaries. Different feature selections and techniques to impute missing values were tested to build the probability model. After conducting a literature review on churn, four types of models were considered: random forests, neural networks, LightGBM, and XGBoost. To improve results, an ensemble was constructed using a Generalized Linear Model (GLM), and isotonic regression was applied to one of the models to calibrate the predictions. The main goal is to achieve a well-calibrated model whose probability predictions are expected to have the same percentage of confidence. To compare the models obtained, RMSE and LogLoss were used to measure the loss, while Expected Calibration Error (ECE) and reliability diagrams helped to assess the calibration.

KEYWORDS

Customer Churn; Insurance; Geospatial variables; Ensemble; GLM; Model Calibration;

INDEX

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

DMLC	Distributed (Deep) Machine Learning Community
ECE	Expected Calibration Error
FR	Feature Removal
GLM	Generalized Linear Model
K-NN	K-Nearest Neighbors
LGBM	LightGBM
LNG	Liquified natural gas
MSE	Mean Squared Error
NN	Neural network
PAV	Pool-adjacent Violators
RF	Random Forest
RMSE	Root Mean Squared Error
WGS84	World Geodetic System of 1984
XGB	XGBoost, an Extreme Gradient Boost model

1. INTRODUCTION

This report is the outcome of a project developed for an insurance company in Portugal to build a model predicting customer churn for the auto branch. Due to sensitive information present in the data, the focus will be on the models' performance without presenting variable values or distributions. For that reason, the performed exploratory data analysis cannot be included in this report.

The purpose of insurance is to protect against uncertainties and unpredicted risks. To achieve such a purpose, the principle of risk pooling is applied, where a group of people contributes to a common fund through premiums. Given this contribution, the entity responsible for managing the fund assumes the agreed-upon risks and compensates for losses covered in the insurance policy. The advantage of this principle is spreading the risk for both the members and the entity. Once a loss occurs, the person will not have to assume a substantial cost amount because it is covered in the insurance policy, nor will the entity sustain the financial burden just from that person's contributions since it uses other members' premiums to compensate for the costs.

Customer churn is a phenomenon where customers stop using a service/product by switching to a competitor or ending the usage altogether. This phenomenon presents a challenge for insurance companies as it can put at risk their client portfolio profitability. Therefore, there is a need to identify the customers more likely to churn and decide what actions to take to mitigate. Not all churns can be prevented, but they can be downsized.

This project aimed to predict the probability of a new customer churning in the first 30 days of their contract. It has been shown that capturing new customers can be multiple times more costly than retaining the current ones (Ng & Liu, 2000). Hence, given the significant investment in attracting new customers, preserving them to obtain revenue and maintain the client's portfolio sustainability is crucial. Although retaining the current customers is essential, the scope of this project only included the new ones because these two types of customers have different behaviors. Thus, they must be analyzed separately to draw correct conclusions and build appropriate predictive models.

The interest in predicting the churn probability of a new customer lies in identifying which clients will most likely cancel their policy and accordingly perform actions to prevent it. Such actions consider not only the churn probability but also other characteristics and information of the customer to create the most accurate portrait. Most of these actions rely on direct contact with the customer, which can be performed by text message, email, or call. In these contacts, it is usually offered a better proposal than initially to retain the customer.

Considering this project intended to determine a probability, the best model must have calibrated values, i.e., a predicted churn probability of a customer must reflect the confidence degree of the result. For example, if the predicted probability of churn is 70%, it is expected to be correct 70% of the time. On the one hand, if it is correct more often, then the model is pessimistic. On the other hand, being correct less often means the model is optimistic. These two last situations are not ideal because the proportion of correct instances does not match the predicted value.

Moreover, different metrics must be considered because the final output is a probability, not a class membership. However, the approach used in this project is similar to binary classification. Also, the most accurate model in a binary classification problem can have very uncalibrated probabilities, being

counterproductive and useless for the ambition of this project. Therefore, predicting a probability or class membership can lead to different models.

The beginning of this project started with extracting the data necessary to build the models. The data used is from the auto insurance branch of the company, and the period selected was from January 2019 until March 2023. The models were trained and validated on data from the first four years (i.e., 2019 to 2022) using 85% of observations of this period for training and the remaining 15% for validation. After fine-tuning parameters and comparing models, the first three months of 2023 were used to evaluate how the final selected model will perform on new unseen data. This evaluation will clarify how the probabilities' error will be distributed.

This report is organized into multiple chapters. Chapter 2 presents the literature review to find examples of work previously developed on this subject, giving more relevance to the models regularly used for churn prediction and the selection of variables that improve performance. Chapter 3 explains the methodology used throughout this project, including feature selection and engineering, missing values imputation, outlier detection, and the datasets outputted from such transformations. Additionally, the results are shown and discussed in Chapter 4. Finally, the conclusions are drawn in Chapter 5 while also referring to limitations and future work recommendations.

An interesting development was engineering two new features that enlightened customer behavior and environment. The first is the distance to the closest insurance intermediary, and the second is the number of intermediaries in a square of 5 km centered in the client's postal code. The construction of such variables led to the application of techniques from other areas, such as Geodesy. Furthermore, another relevant development was using a Generalized Linear Model (GLM), which is widely used by actuaries for pricing insurance products, as the ensemble of three XGBoost algorithms.

By the end of this project, it was expected to obtain a model that predicts calibrated churn probabilities. If the results are quality ensured, they could be used by other teams in the company to decide what measures to enact, preventing customers from canceling their insurance policies.

2. LITERATURE REVIEW

The main concerns while researching literature regarding customer churn are (1) variable selection, which data should be considered to predict churn accurately, and (2) model selection, finding what models yield the best performance in customer churn problems, ideally in insurance data since it is similar to the problem at hands.

Risselada et al. (2010) studied the staying power of a churn model in insurance data, which is the predictive performance of a model over various periods after the estimation period. Logit models and decision trees were used for this study, both with and without bagging. The authors concluded that the staying power is low for all methods, given that the model parameters for logistic regression and the significant variables for prediction differ between periods. Therefore, the recommendation is to regularly retrain the models to maintain a high performance.

Günther et al. (2011) analyzed churn for a client considering information from all insurance branches (car, home, and health). The logistic regression fitted to the data incorporates time-dynamic explanatory variables and interactions, an improvement from Risselada et al. (2010). To enforce the assumed linear relationships between the explanatory variables and the logit, their approach uses generalized additive models (GAM) as an intermediate modeling step to allow the identification of complex non-linear relationships between the explanatory and the response variables.

Huigevoort (2015) analyzed the customer churn for a health insurance company. In their research, relevant variables were identified from other literature and by consulting experts in the insurance field. The variables were split into socio-demographic, customer/company interaction, and product-related variables. From the literature, the selection of variables analyzed previously conducted research in insurance and other areas such as telecommunications, banking, newspaper market, and multimedia. To predict churn, four models were considered: logistic regression, decision tree, neural networks, and support vector machine. In the end, logistic regression and neural networks were the most well-performing models. One interesting approach from this author was creating customer profiles using K-means clustering and using this information for churn prediction.

Spiteri and Azzopardi (2018) aimed to predict churn in motor insurance. It also includes extensive research on prediction techniques and variable selection of customer churn. The authors list logistic regression, decision trees, and neural networks as the most popular implemented models.

Additionally, the subsequent literature is more recent and presents the performance of gradient-boosting models, which have not yet been considered.

Gregory (2018), using an XGBoost, won the challenge sponsored and managed by the 11th ACM International Conference on Web Search and Data Mining (WSDM 2018). Such winning showcases the high performance of an XGBoost model in predicting customer churn for a music streaming service.

Senthan et al. (2021) investigated customer churn prediction in the telecommunication industry. They compared the performance of a decision tree, logistic regression, support vector machine, neural networks, random forest, XGBoost, and AdaBoost. Eventually, the highest accuracy was achieved by the XGBoost.

Muthupriya et al. (2022) also worked with customer churn prediction in the telecommunication industry. Once again, XGBoost was the best-performing model compared to logistic regression, random forest, Naïve Bayes, and support vector machine.

To finalize, the following literature items detail more specific considerations to have while selecting the variables for the model:

- Van den Poel and Lariviere (2004) investigated predictors of churn in financial services. Their findings suggest that demographic characteristics, changes in the environment, and active relationships with customers highly impact customer retention;

- de la Llave et al. (2018) explored how the physical proximity of customers to insurance offices and the interaction between nearby customers can be used as spatial factors that impact churn probability. The authors showed churn probability decreases for customers in the surroundings of an insurance company's offices, whereas high lapse risk was identified for customers close to the competitors' offices. Moreover, a customer's churn probability increases if nearby customers churn, demonstrating spatial correlation. Thus, geographical information can be contemplated as an explicative factor of customer churn behavior;

- Scriney et al. (2020) showed, for healthcare insurance, the importance of having a complete customer history to generate retention, which is one of the key parameters for calculating customer lifetime value.

3. METHODOLOGY

3.1. BUSINESS UNDERSTANDING

The aim of this project is to create a probability model for churn of new clients. As such, it will be considered churn when a new client does not pay the premium in the first 30 days of the contract start. The amount paid is according to the payment instalments selected by the client, this means clients who choose instalment payment will only have to pay the established value and not the full premium.

Once a client contracts an insurance, it must pay the premium in the following days. In this case, the company requires the payment to be completed by the policy start date. If a client does not perform the payment within this time range, it will not be automatically considered a canceled policy because it is applied a grace period. The grace period is a 25 days timeframe counting from the start of the insurance policy and extending to the next 25 days. During this period, the policy will not be canceled, and the client has the chance to pay the premium without any repercussions. Once this period ends, if no payment was made, the policy is canceled with effect from the start date. Therefore, a canceled policy has two associated dates: the cancelation date, when the policy was canceled; and the cancelation effect date, since when the policy cancelation has effect. Therefore, to determine if a client churned in the first 30 days it is considered the cancelation effect date.

The time frame of 30 days was chosen considering the grace period. Time frames lesser than the grace period would not have reliable data of churned customers because both the cancelation date and cancelation effect date would still be empty, giving a false idea that the customer has not churned when they could have.

Customer churn is a problem for most companies, and numerous studies have been made in this field, which made it easier to find guides and considerations on how to develop this project. Additionally, the literature shed light on relevant variables that should be selected. These insights complement the business knowledge shared by insurance experts in the company, who have a more profound experience from a business perspective and a client perspective.

Table 1 - Selected variables and recommendation origin

Driver and policy holder	Reference
Policy holder age	Experts
Driver age	Literature and experts
Years of driving experience	Literature and experts
Policy holder is the driver	Experts
Latitude	Literature
Longitude	Literature
Has auto insurance in another company	Literature and experts
Has house insurance in the company	Literature and experts
Has health insurance in the company	Literature and experts
Has personal accidents insurance in the company	Experts
Has other insurances in the company	Literature and experts
Distance to closest insurance intermediary	Experts
Number of insurance intermediaries in the area	Experts

Vehicle	Reference
Vehicle year	Literature and experts
Vehicle value	Literature and experts
Engine power	Literature and experts
Fuel	Literature and experts
Policy	**Reference**
Discount campaign	Experts
Days between insurance contract and start	Experts
Sales channel	Experts
Instalment	Experts
Premium	Literature and experts
Default discount	Experts
Applied discount	Literature and experts
No claims discount	Literature and experts
Coverage level	Literature
Collision coverage	Experts
Theft coverage	Experts
Windscreen coverage	Experts
Historical behavior	**Reference**
Claims over last three years	Literature and experts
Years with insurance	Experts
Previous insurance provider	Experts
Number of previous insurance providers	Experts
Number of previous direct insurance providers	Experts
First insurance policy in the company	Literature and experts
Number of online quotes	Experts
Number of offline quotes	Experts
Number of total quotes	Literature and experts
Number of issued policies over last three years	Experts
Number of canceled policies over last three years	Experts
Policy balance over last three years	Experts

Table 1 summarizes the selected variables from both literature and experts. The literature used considers not only variables for auto insurance (Spiteri & Azzopardi, 2018) but also health insurance (Huigevoort, 2015). In fact, for the latter, Huigevoort took inspiration from customer churn variables applied in other industries such as telecommunications, banking, the newspaper market, and multimedia. Another study considers the customer churn associated with a client and not a policy, taking into account all policy types they have, namely car, home, and health (Günther et al., 2011). For these last two references, some adaptions were made to suit the variables of auto insurance. The experts consulted specialize in different fields such as data science, data analysis, actuary, and underwriting.

3.2. DATA PREPARATION

Before feeding the data to models, some adjustments are required to prevent errors and attempt to improve the results of such models. After all, the models are only as good as the data fed into them.

The adjustments ensure high-quality data by performing some restrictions and checks. Moreover, new features were created to better reflect customer behavior and environment.

With the data quality ensured and new features created, the data was split into train, validation, and holdout datasets. The train dataset comprises 85% of data ranging from 2019 to 2022 and is used to train the models, as shown in Figure 1. The remaining 15% of data from the same period corresponds to the validation dataset. It evaluates the models obtained from the train dataset and the parameter fine-tuning. Finally, the holdout dataset has data from the first three months of 2023 and will be used to assess how the chosen final model will perform on unseen recent data. To avoid any data leak, the holdout dataset is not considered in the model training nor in the model fine-tuning. As said, it is only used once the final model is selected.

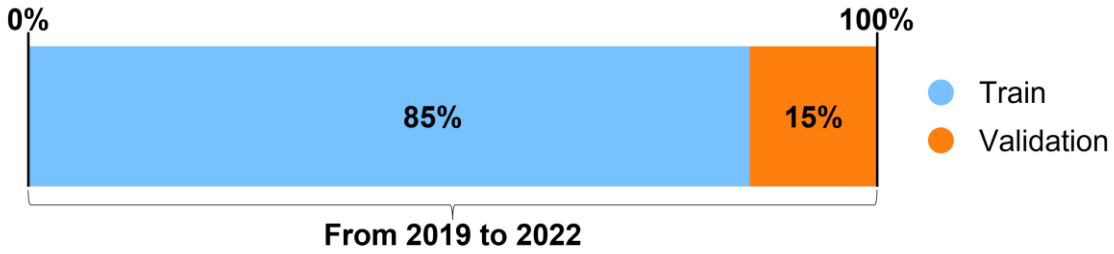

Figure 1 - Split of data from 2019 to 2022 into train and validation datasets

Instead of splitting the train and validation datasets considering the percentages above, another approach would be using the data from 2019 until September 2022 to train and the last three months of 2022 as validation. This approach was not applied because it would prevent the model of training on the most recent data of 2022, which could lead to worse results. The percentages approach may not validate the most recent data but trains on it, being more resourceful for this project's scope.

After this, missing data had to be filled in, so the models would avoid any errors. The correlation between variables was analyzed, which allowed identifying some highly correlated features in the data. Lastly, a feature selection was performed taking into account the correlations determined to reduce redundancy in the data.

It is worth mentioning that an outlier detection task was performed. From the distribution analysis of the variables, a few values could be considered outliers. However, such values can incorporate valuable information since it describes a different customer behavior. Therefore, a conservative approach was taken in this project to avoid discarding relevant customer data that is not considered standard.

3.2.1. Data quality

To ensure high-quality data, some restrictions and checks were applied. Firstly, policies canceled with the motives of cloning, policy replacement, and error were excluded. This exclusion was enforced because these motives for cancelation resulted in the creation of a new policy. Therefore, the clients do not indeed churn, they simply have changes in their policy parameters that, for established processes reasons, the old policy had to be canceled and a new one created with the corrected parameters.

The second exclusion applied to the data is not considering motorcycles for this project. Cars and motorcycles have different tariffs, different customer pools, and behavior. Given such variations, training a model with these two types of customers would probably not generalize and perform well. Also, it is relevant to notice the car portfolio is much larger than the motorcycle. Thus, the priority was to focus on the largest group of customers.

Thirdly, it was checked if the customers flagged as churned had their policy canceled within the first 30 days of the contract to ensure the premise of this project. Other checks enforced are regarding the cancelation dates or churned customers. One is confirming that the cancelation date is always more recent than the cancelation effect date, given that we apply a grace period in the company. The other check is verifying that the policy end date is always more recent than the cancelation effect date, since it should not be possible to cancel a policy that is no longer in effect. Also, it was validated that there were no premiums below 1€, and if there were, those policies were excluded since premiums with such low values must be an error.

Finally, the fuel types were simplified. The company follows the fuel types defined by Eurotax, a tool to appraise cars and provide their full specifications and technical data (Eurotax, 2023). This tool is quite descriptive of the fuel and motor type. For example, hybrids, mild hybrids, and plug-ins have different values, and it is also specified if they use diesel or gasoline. Liquified natural gas (LNG) can be 100% LNG or bifuel, i.e., it has both gasoline and LNG. This results in many irrelevant categories for the scope of this project. Hence, the values were simplified into three categories: diesel, gasoline, and electric.

3.2.2. Feature engineering

In an attempt to better describe the customer behavior and environment and having the work of de la Llave et al. (2018) as inspiration, two variables were created: (1) distance in meters to the closest insurance intermediary; (2) number of insurance intermediaries available in a square with sides of 5 km and centered in the client. It is expected for customers who live in areas with various close insurance intermediaries to be better informed of insurance pricing, making it easier for them to change insurance companies, leading to customer churn.

The development of these new variables was possible using postal codes. Since a postal code can range from a single street to a much larger area, it was necessary to characterize it with one single point. The best solution is using postal code centroids, which show the approximate center of the polygon that the postal code delimits. Once a centroid is determined, it is possible to assign a pair of latitude and longitude to the postal code.

Although using the centroid of a postal code can lead to substantial error in some cases, mainly where a postal code covers a large land area, this method provided valuable insights for researchers, as evidenced by its successful implementation in various other fields. One of those fields is transportation and urban planning, where, for example, the goal was to study how urban commute is impacted by income (Bogomolov et al., 2021). Another use is in health research, specifically in the study of intervention strategies for epidemiological analysis, wherein an emerging health threat, there is a need to identify which areas will most likely be impacted (Ajayakumar, 2023).

To achieve the goal, the process involved the clients' postal code, a dataset containing the addresses of all insurance intermediaries operating in Portugal, and another dataset associating each postal code to the corresponding pair of latitude and longitude coordinates for the centroid. With these datasets, every client and insurance intermediary was assigned a pair of latitude and longitude according to the centroid of their respective postal code.

For the first variable, there was a need to compare the distance to all insurance intermediaries for each client and determine the minimum value. This implies creating a distance matrix. The solution was converting the geodetic coordinates to cartesian. For this conversion was used the set of formulas from equations A.6 to A.8 presented in Appendix A, with the values defined by the World Geodetic System of 1984 (WGS84) for the semi-major axis and for the inverse flattening, $6\,378\,137\,m$ and 298.257223563, respectively. Appendix A provides explanations for the two reference systems used in this project (geodetic and cartesian coordinates), how to convert from geodetic to cartesian, and how to determine the distance between coordinates for both reference systems.

Cartesian coordinates calculate the distance between points using the Euclidean distance. Initially, creating the distance matrix was attempted, but it was impossible to compute due to a lack of available memory. The solution found was applying the k-d tree algorithm, which is a more efficient calculation since it does not require determining the distance between every client and all insurance intermediaries. This efficiency solved the lack of available memory for the distance matrix. Appendix B presents an overview of the k-d tree algorithm, explaining how the tree is constructed and how the algorithm finds the nearest neighbor between the tree and a given point.

There are more accurate approaches determining the distance between two geodetic points. One of them is Vincenty's algorithm, which considers the Earth's surface as an ellipsoid and determines the distance between two points over that surface. On the contrary, Euclidean distance determines the length of the line segment between two points (Britannica, 2023). Therefore, using the Euclidean distance instead of Vincenty's algorithm has an associated error that becomes greater the further away the two points are. The distances are relatively small for the problem at hand, and such errors should be minimal. To check the largest error possible in the dataset, it was considered the furthest Euclidean distance between a client and the closest insurance intermediary. Using the latitude and longitude from these two points and applying Vincenty's algorithm, it was determined that the error was below the centimeter level. Therefore, using the Euclidean distance does not create issues with data quality and is a reliable value.

Moreover, Vincenty's algorithm is anticipated to converge and perform well. Not only is it still widely used (Karney, 2013), but given the scope of this report, it is not expected to determine the distance of nearly antipodal points since only Portugal is being considered. Nonetheless, this algorithm could not be used in the k-d tree, requiring the computation of a full distance matrix, which would probably lead to the problem of insufficient available memory.

For the second variable, the variation must be defined given the point's latitude and longitude to create the square. Given that the square side is small, let us consider a geocentric system with a spheric approximation of the Earth's surface. Let us start to define the variation in latitude, denoted as $\Delta\phi$, which can be in degrees or radians. This variation occurs over a longitude value that cuts through the Earth and creates a circle, known as a meridian, represented in Figure 2.

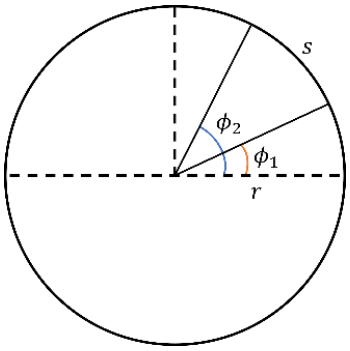

Figure 2 - Latitude variation $\Delta\phi$, over a meridian, is the difference between ϕ_2 and ϕ_1, and s is the arc length of this variation. The Equator is represented by the diameter in a dashed line and r is the meridian radius.

This circle has a radius equal to the Earth's with $6\,371\,km$, according to the WGS84. Knowing the perimeter of a circle is $2\pi r$, we can apply cross-multiplication to determine the variation in latitude using radians:

$$\frac{s}{2\pi} = \frac{5}{2\pi \times 6\,371} \Leftrightarrow s = \frac{5}{6\,371} \tag{3.2.1}$$

Changing it to degrees, we have:

$$\Delta\phi = \frac{s \times 180}{\pi} = \frac{5 \times 180}{6\,371\pi} deg \tag{3.2.2}$$

A similar logic is applied to determine the variation in longitude, denoted as $\Delta\lambda$. However, the longitude variation is measured in a latitude that is not on the equator. Therefore the circle created by cutting through Earth on that latitude originates a parallel whose radius decreases with the absolute latitude value. This circle radius can be determined by the cosine of latitude, as shown in Figure 3.

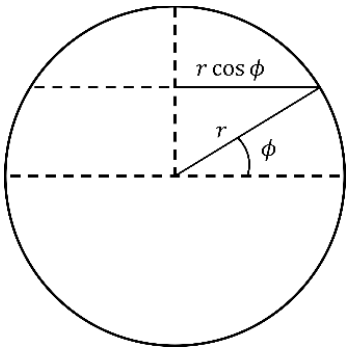

Figure 3 - Equator and parallel of latitude ϕ represented over a meridian. Given a latitude and Earth's radius r, it is possible to determine the radius of a parallel by $r\cos\phi$

Hence, the radius is given by $6\,371 \times cos\,\phi$. Following the same construction used in $\Delta\phi$, we have $\Delta\lambda = \frac{5}{6\,371 \times cos\,\phi}$ in radians. Converting to degrees, we have:

$$\Delta\lambda = \frac{5 \times 180}{6\,371\pi \times cos\,\phi} deg \quad (3.2.3)$$

With these formulas, we have the variation in latitude and longitude, translating into a 5 km change over the Earth's surface. Thus, the four sides of the square centered in the client's latitude and longitude with a side of 5 km are given by:

$$top\ side = \phi + \frac{\Delta\phi}{2} \quad (3.2.4)$$

$$bottom\ side = \phi - \frac{\Delta\phi}{2} \quad (3.2.5)$$

$$left\ side = \lambda - \frac{\Delta\lambda}{2} \quad (3.2.6)$$

$$right\ side = \lambda + \frac{\Delta\lambda}{2} \quad (3.2.7)$$

Another option was to consider a circle centered in the client's postal code with a radius of 5 km. This approach would require calculating the distance between the client and every insurance intermediary, becoming computationally expensive and possibly having the same memory issue the first variable initially had. The formulas above allow for a more efficient count of the number of insurance intermediaries. This is possible since it only requires filtering latitudes and longitudes instead of performing distance calculations.

3.2.3. Missing values

The absence of values must be dealt with before using data to train models. Some missing values represent an absence of what those variables measure. This is the case for variables such as difference in days between the contract date and policy start date, claims in the last three years, years with insurance, number of previous insurance providers, online/offline quotes, issued/canceled policies in the last three years, flags for different insurance covers, and others, where a missing value represents zero.

For the variable of no claims discount, when the value is empty, the default value was applied, i.e., no discount applied.

In the case of the categorical variable of the previous insurance provider, it is assumed that the risk unit was not covered by another company, which means this is its first insurance. So, a new category was created to accommodate this situation.

There were two approaches for variables constructed from other ones according to the variable at hand. One was handling the missing values in the independent variables and, after this, constructing the dependent variables. This was the case for variables such as number of total quotes, which depends on the number of online and offline quotes, and the policy balance, which is the difference between issued and canceled policies in the last three years. The other approach was using the dependent variables' median value. For the two variables created in the feature engineering, latitude and longitude values were necessary. Without such, there was no way to construct the values. Therefore, when the values were empty, it was replaced by the variable median.

Lastly, the missing values of remaining variables that do not fall under the abovementioned situations must be handled. There were two approaches for these cases. The first was filling the missing values with the median and the mode for numerical and categorical variables, respectively. The second approach was using the K-Nearest Neighbors (K-NN) algorithm. For this, the categorical variables had to be encoded to numerical values. Subsequently, the algorithm was trained using different values for k, specifically 1, 7, and 50. For this application of K-NN, the weight of each neighbor was proportional to its distance to the point being imputed. To select the best value for k, the distributions for the different k values were compared to the distribution with non-imputed values. Since there were few missing values, the distributions did not show differences. It was also considered that using k equal to 50 is more computationally expensive, taking a long time to compute. On the other hand, using the value 1 for k can induce an error since it only considers the closest neighbor instead of the neighborhood. For these reasons, the selected value was 7.

3.2.4. Correlation

Once the numerical variables are clean, it must be checked if they are repeating the same information. Such check can be performed by analyzing the correlation between the variables. In this case, it was used the standard Pearson correlation.

Considering a high correlation with values above 0.6 or below -0.6, some correlated variables were found by evaluating the correlation matrix. The policyholder age, driver's age, and years of driving experience have high values above or equal to 0.83 among all of them. The vehicle value and production year are highly correlated, and the vehicle value is also correlated to the flag of theft coverage. The other highly correlated variables are: the number of previous insurance providers and number of previous direct insurance providers; flag of home insurance and flag of other insurance policies in the company; total quotes and offline quotes; number of issued policies and number of canceled polices, both considering the last three years; flag of collision coverage and flag of theft coverage.

3.2.5. Feature selection

After the exploratory data analysis of the dataset, especially from the correlation matrix, it was verified that some variables might be repeating the same information. Hence, some of those variables could be removed. To select which highly correlated variables to remove, the correlation values with the remaining variables were inspected. This selection prioritized the ones that have a low correlation with other variables. Hence it selected those that do not repeat the same information given by other variables. Also, it was considered business knowledge to avoid removing variables that are valuable to understand the model behavior and have returned favorable results in previous situations. Considering these two aspects, the policyholder age, years of driving experience, vehicle value, theft coverage, number of previous insurance providers, flag of home insurance, offline quotes, and issued policies in the last three years were the chosen variables to remove.

Nonetheless, it will also be considered the dataset without removing these variables because in some previous projects, the simplest of the datasets, those with minimal alterations made by data scientists or analysts, yield the best results. By considering both the clean dataset, without correlated variables, and the original, with correlated variables, no data value will be lost. Hence, the chance of finding a better-suited model for this project increases.

3.3. MODELING

3.3.1. Datasets

After the data preparation, four datasets are considered. One with missing values imputed with the median and mode, and a second dataset using the K-NN algorithm with k equal to 7. The remaining two datasets have the missing values imputed the same way as the first two. The difference is that the correlated variables were removed in these last datasets. Listing these datasets, we have:

1. Missing values imputed using median and mode, with correlated variables;
2. Missing values imputed using median and mode, without correlated variables;
3. Missing values imputed using 7-NN algorithm, with correlated variables;
4. Missing values imputed using 7-NN algorithm, without correlated variables.

The training of models with different datasets is possible because the computing power is located in an auto-machine learning platform, removing the limitation from the personal computer. The platform allows to select various models and train them on distinct datasets. However, any required data alterations must be performed on a personal computer since the platform does not support such functionality.

3.3.2. Models

Customer churn is a widely studied issue from academic and business perspectives. In Chapter 2, multiple literature sources of similar customer churn problems were considered to decide which models to use. Given the abundance of research on this topic, it was possible to narrow down the selection of models to use. The models retrieving the best results usually are random forests (Spiteri & Azzopardi, 2018), neural networks (Agrawal et al., 2018; Almeida et al., 2022), and gradient boosting algorithms, more specifically LightGBM and XGBoost (Gregory, 2018; Muthupriya et al., 2022; Senthan et al., 2021). It is worth noting in the literature referenced, different models are used to compare results. The selection was made taking into account the models that repeatedly yield the best predictions compared to the rest. Hence, time was optimized to not consider models that do not perform well on customer churn.

3.3.2.1. Random Forests

Random Forests (RF) are an ensemble of independent decision trees that combine their predictions to output a final single result. Each decision tree in the forest is built using a sample with replacement from the training data, i.e., a bootstrap sample. Moreover, at each node of the trees, a subset of the original features is used to split (Cutler et al., 2007). To reach a single final result, instead of averaging class predictions, the implementation used in this project combines the multiple classifier decision trees by averaging their probabilistic predictions (Pedregosa et al., 2011). The random components for the training data and feature selection aim to decrease the variance of the estimation since individual decision trees usually have high variance and tend to overfit (Breiman, 2001; Louppe, 2014).

3.3.2.2. Neural networks

Neural networks (NN) are bio-inspired models aiming to reproduce a simplified version of the human brain. NNs are comprised of neuron layers containing an input layer, one or more hidden layers, and

an output layer. Data moves from the input neurons to the hidden layers, ending in the output neurons, creating a feedforward network. Except in the input neurons, each layer receives the output from the previous neuron layer as input. The connection between neurons has associated weights and bias. It also introduces non-linearity through an activation function, allowing the model to generalize accurately. During the train of a NN, the error is measured, i.e., the difference between the target value and the output value, and the weights are then readjusted according to the error. Successively repeating this process will increasingly produce an output similar to the target (Haykin, 2009).

3.3.2.3. Gradient Boosting algorithms

Gradient boosting combines an ensemble of weak prediction models to create a robust predictive model. In this project, only decision trees will be used as weak predictors. The idea behind gradient boosting is to iteratively train new weak predictors (Hastie et al., 2009). For each iteration, the weight of data points is updated according to the residual, i.e., the difference between predicted and actual target values. More significant residuals have a higher weight to ensure the next iteration improves misclassified predictions. The overall prediction accuracy is improved by focusing on the mistakes made by the previous predictors. To produce the final prediction, a weighted majority vote is used to combine the predictions from all the trees constructed (Hastie et al., 2009).

XGBoost is a gradient boosting algorithm developed by the Distributed (Deep) Machine Learning Community (DMLC) group to deliver a scalable end-to-end tree boosting algorithm. XGBoost implements regularization to prevent overfitting by helping to smooth the final learned weights. It also has a sparsity-aware algorithm to efficiently handle sparse data, such as missing values or categorical encoded variables, allowing parallel tree learning. Another relevant implementation is the weighted quantile sketch, which handles weighted data when finding the best candidate for a split while constructing the trees (Chen & Guestrin, 2016).

LightGBM is a gradient boosting algorithm developed by Microsoft, and other contributors, aiming to efficiently train large datasets. This algorithm has many of the advantages of other gradient boosting tree algorithms, such as regularization and parallel training. However, some characteristics set it apart from XGBoost. One difference is that it constructs the tree leaf-wise, selecting the leaf yielding the best results, instead of the usual level-wise manner creating balanced trees. Another difference is that it selects a subset of data points, prioritizing the ones that contribute the most for computing the gradient, denominated as Gradient-based One-Side Sampling. The final significant difference is Exclusive Feature Bundling, which reduces dimensionality by bundling features that rarely take nonzero values simultaneously (Ke et al., 2017).

3.3.3. Ensemble

Stacking is an ensemble machine learning technique that predicts a final value by combining the predictions from multiple models. This is based on the idea that different models learn some part of the problem, so adding multiple models will allow learning a more significant part of the problem (Ramos et al., 2020).

During the training and validation phase, many models were evaluated. In an attempt to improve the results, a stacking ensemble can be executed. Given that this project's scope is predicting the probability of a binary event, the target variable does not have a normal distribution. Instead, it has a

Bernoulli. Considering this distribution, the ensemble used a Generalized Linear Model (GLM) to combine the predictions from the multiple models.

It is worth noting that the ensemble was trained on validation data since training with the same data the models were trained on would induce bias.

3.3.3.1. Generalized Linear Model

The combination of multiple models in a stacking ensemble can be achieved through another model. In this project, it was used a Generalized Linear Model (GLM), which is an extension of linear regression. GLM allows different distributions for the target variables instead of the normal distribution imposed by linear regression.

To define a GLM is required:

1. Distribution of the target variable
2. Link function, g, which establishes a relation between the target variable and predicted values:

$$g(\mu) = X\beta \tag{3.3.1}$$

where μ is the mean of the target variable, and $X\beta$ is the linear predictor.

The link function serves as a monotonic differentiable function between the linear predictor and the mean of the target variable. Moreover, it specifies that the transformation of the mean is linearly related to the variables in X (De Jong & Heller, 2008).

In stacking ensemble, using a GLM can yield as good results as other meta-models (Kwon et al., 2019).

3.3.4. Calibration

A stacking ensemble requires more computational power and more time to make predictions. Therefore, adding probability calibration to the best model was another considered approach to yield better predictions. Calibration is a postprocessing method applied to a trained model to improve the estimated probability or error distribution (Bella et al., 2009). This is achieved by taking the trained model outputs, which will be the probabilities, using them as input data for the calibrator, and training it according to the actual target values. The two calibration methods considered for this project are Platt scaling and isotonic regression (Niculescu-Mizil & Caruana, 2005).

Platt scaling is a parametric approach applied when the reliability diagram appears to have a sigmoid distribution, as shown in Figure 4. In simpler terms, the distribution has an “S” shape. The other method is isotonic regression a non-parametric approach that creates a stepwise constant function. This method is not restricted to the distribution of the probabilities. However, it is not applicable for small datasets as it tends to overfit.

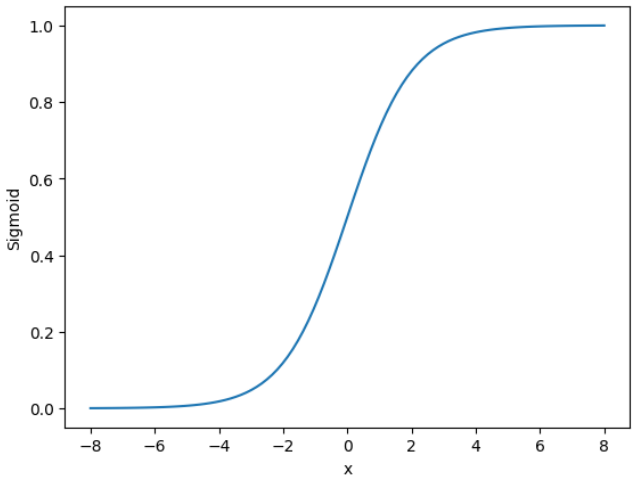

Figure 4 – Example of the shape of the sigmoid function

Regardless of the chosen method, it cannot be fitted with the same data used to train the model we want to calibrate. Doing so would introduce unwanted bias. Therefore, an independent calibration set is required to obtain improved posterior probabilities (Niculescu-Mizil & Caruana, 2005). To fulfill this objective, the test dataset will be used to fit the calibration method.

3.3.5. Isotonic regression

One of the calibration methods is isotonic regression, a nonparametric method that aims to minimize the squared loss between prediction probability and actual label (Menon et al., 2012; Zadrozny & Elkan, 2002). This method is more general, having only one restriction: the fit function must be monotonically nondecreasing (isotonic) (Zadrozny & Elkan, 2002). Isotonic regression can be seen as (Guo et al., 2017; Niculescu-Mizil & Caruana, 2005):

$$argmin_f = \sum_i \big(y_i - f(p_i)\big)^2 \tag{3.3.2}$$

Subject to $f(p_i) \leq f\big(p_j\big)$ when $p_i \leq p_j$, where y_i and p_i are actual value and predicted probability of observation i, respectively.

To find the fit function that minimizes loss, the pool-adjacent violators (PAV) algorithm is used. The algorithm does what its name suggests: inspects the points, and if it finds a point that violates the restriction, it pools that value with its adjacent points, forming a block (de Leeuw et al., 2009).

In simple terms, PAV works as follows. Consider the pairs (p_i, y_i). Without loss of generality, assume we organize the predicted probabilities in ascending order, i.e., $p_1 \leq p_2 \leq \cdots \leq p_n$, where n is the cardinality of the set of points (Brummer & Preez, 2013; Menon et al., 2012). After this, start with y_1 on the left and move to the right until finding a point i such that $y_i \geq y_{i+1}$. Replace these two by their average and back average to the left as needed to ensure monotonicity. Continue this process to the right until reaching y_n (Guntuboyina & Sen, 2017). This construction creates a stepwise constant solution.

One disadvantage of isotonic regression is its sensitivity to outliers. Therefore, it performs well with large amounts of data to prevent overfitting (Guntuboyina & Sen, 2017; Niculescu-Mizil & Caruana, 2005).

3.4. EVALUATION

During the models training, ensemble implementation, and tackle calibration, metrics must be used to access the performance.

For binary classification, many of the most common metrics, such as accuracy, recall, precision, F1-score, and others, require the model output to be a 0 or 1. However, this project aims to predict how likely it is for a customer to churn rather than binarily predict if they will churn. By not defining a threshold upon which the final binary classification is decided, it becomes impossible to measure the model performance using the typical metrics.

To evaluate the models in this project, a metric cannot be defined by a threshold. Instead, it must make a comparison between the data we have, the target variable, and the model outputs, predicted probabilities. Therefore, the chosen metrics are Mean Squared Error (MSE) and LogLoss (Murphy, 2012).

When performing a predictive model for classification, we often want not only the predicted label but also the probability of the respective label. Some models reach well-calibrated probabilities, and others cannot. For the latter, there are calibration methods to improve the prediction's probabilities.

For the models and the ensemble, MSE and LogLoss will be used as metrics. The reliability diagrams will supply a visual representation for the calibration, while the Expected Calibration Error (ECE) provides a metric to compare more accurately.

3.4.1. Mean Squared Error

Mean Squared Error (MSE) allows to assess the inaccuracy of probability estimation properly (Zhang et al, 2006). More explicitly, it measures the average loss of the data at hand by squaring the difference between the actual and the predicted values. Therefore, small MSE values represent less inaccurate models. MSE is determined by the following formula:

$$MSE = \frac{1}{n}\sum_{i}(y_i - p_i)^2 \tag{3.4.1}$$

where y_i and p_i are actual value and predicted probability of observation i, respectively, and n is the total number of observations.

Since the models output values between 0 and 1, the MSE can retrieve minimal values since it squares the difference between already small numbers. This can jeopardize the comparison between models. Therefore, the Root Mean Squared Error (RMSE) can also be considered because it is the root of the MSE and yields more significant values that are easier to compare.

3.4.2. LogLoss

LogLoss measures the inaccuracy of predicted probabilities. It increases as the predicted probability diverges from the actual label (Li et al., 2020). Therefore, just like the MSE, smaller values represent a minor loss. For binary classification, the LogLoss s formula is (Bishop, 2007):

$$LogLoss = -\frac{1}{n}\sum_{i} y_i \, log(p_i) + (1 - y_i) \, log(1 - p_i) \tag{3.4.2}$$

where y_i and p_i are actual value and predicted probability of observation i, respectively, and n is the total number of observations.

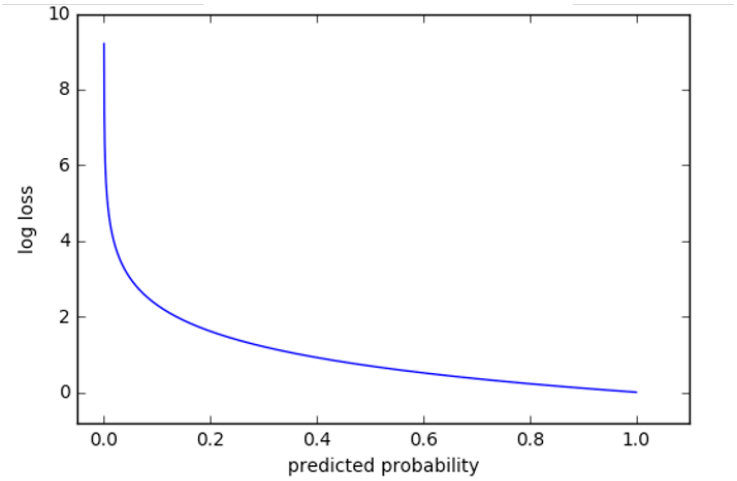

Figure 5 - LogLoss distribution for a positive observation. Adapted from ML Glossary (2023)

This metric is not linear when determining the loss since it uses the log. The figure above plots the range of possible loss values for a positive observation, i.e., the actual label equals one. As the predicted probability gets closer to one, the LogLoss decreases slowly. On the other hand, while decreasing the predicted probability, the loss rapidly increases. Therefore, LogLoss penalizes heavily confident predictions that are wrong.

In machine learning LogLoss and cross-entropy are the same for binary classification, despite being slightly different in other contexts.

3.4.3. Reliability diagram

Model calibration can be visualized with reliability diagrams (DeGroot & Fienberg, 1982). The plot is created by binning the predicted values. Usually, the prediction space is discretized into ten equally spaced bins. The first bin will have predicted values between 0 and 0.1, the second bin will include values between 0.1 and 0.2, and so on until reaching the tenth bin with values between 0.9 and 1. Once the predicted values are discretized, the diagram is created by plotting the mean value predicted value against the actual fraction of positive cases (Niculescu-Mizil & Caruana, 2005). In the scope of this project, positive cases are the churned customers. The reliability diagram should plot the identity function in a well-calibrated model. Any miscalibration will be represented by deviations from the perfect diagonal line (Guo et al., 2017). An example of a perfect calibration and a miscalibration is shown in the figure below.

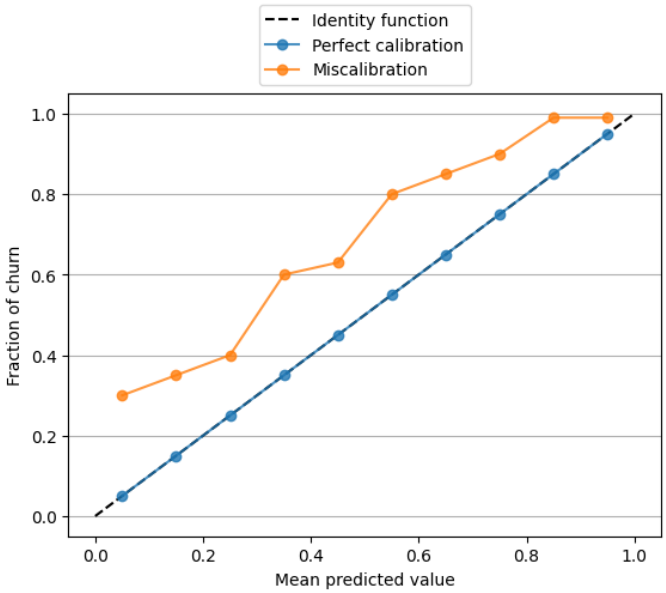

Figure 6 – Example of a perfect calibrated and miscalibrated models that coincides with the identity function

3.4.4. Expected Calibration Error

While reliability diagrams are helpful to visualize the calibration level of models easily, it is necessary to define a metric to evaluate the calibration. One such metric is the Expected Calibration Error (ECE), which measures the weighted average absolute difference between a bin's mean predicted value and actual fraction of positive cases.

$$ECE = \sum_{m} \frac{|B_m|}{n} |\hat{y}_m - \hat{p}_m|, \tag{3.4.3}$$

where m is the number of bins, B_m is the number of observations in the m-th bin, n is the total number of observations, $\hat{y}_m$ is the fraction of positive cases in the m-th bin, and $\hat{p}_m$ is the mean predicted value in the m-th bin.

4. RESULTS AND DISCUSSION

4.1. MANUAL FINE-TUNING

Considering there are four datasets presented in Chapter 3 of Methodology, their names will be simplified in the following way:

1. Original: dataset with missing values imputed using median and mode, and with correlated variables;
2. Original with FR (Feature Removal): dataset with missing values imputed using median and mode, without correlated variables;
3. 7-NN: dataset with missing values imputed using 7-NN algorithm, and with correlated variables;
4. 7-NN with FR: dataset with missing values imputed using 7-NN algorithm, without correlated variables.

The models used to train with the datasets above are RF, NN, LGBM, and XGB. Three manual fine-tuning iterations were attempted to improve the results of all models. The iterations were evaluated using MSE and LogLoss, giving more importance to the latter since it penalizes the occurrences furthest from the actual target value. After these iterations, a grid search was performed for the selected model. This model was selected considering its performance when compared to the remaining models. For the manual fine-tuning and the grid search, the metrics were calculated for both the train and validation datasets to verify the model's degree of overfitting.

For the first run of the models, first iteration, RF has 450 estimators with Gini as criterion, NN is composed of a single layer of 64 neurons, and both LGBM and XGB have early stopping to determine the best number of trees.

Table 2 - Iteration 1 train results for all models and datasets

Dataset	MSE				LogLoss			
	RF	NN	LGBM	XGB	RF	NN	LGBM	XGB
Original	0.0768	0.0805	0.0755	0.0730	0.2774	0.2840	0.2678	0.2596
Original with FR	0.0789	0.0810	0.0758	0.0772	0.2813	0.2859	0.2680	0.2746
7-NN	0.0642	0.0820	0.0673	0.0695	0.2197	0.2896	0.2345	0.2429
7-NN with FR	0.0652	0.0827	0.0658	0.0602	0.2226	0.2923	0.2293	0.2103

Table 3 - Iteration 1 validation results for all models and datasets

Dataset	MSE				LogLoss			
	RF	NN	LGBM	XGB	RF	NN	LGBM	XGB
Original	0.0805	0.0802	0.0791	0.0789	0.2852	0.2835	0.2797	0.2793
Original with FR	0.0809	0.0806	0.0795	0.0792	0.2862	0.2855	0.2812	0.2802
7-NN	0.0799	0.0820	0.0771	0.0768	0.2825	0.2895	0.2723	0.2717
7-NN with FR	0.0801	0.0827	0.0775	0.0773	0.2832	0.2922	0.2742	0.2741

Giving more importance to LogLoss, the XGB model of the 7-NN dataset presents the best result for validation, although it might be overfitting given the difference with the train data. From the model with low LogLoss, the XGM model for the Original dataset with FR performs better in avoiding overfitting. In general, the Original datasets present lower overfitting than the 7-NN datasets.

The step of fine-tuning was applied to all models because the parameters used in the first run were simple, and there is a need to investigate if different values could return improved results. Hence, for the second iteration, RF considers only balanced trees to reduce overfitting. NN has an additional hidden layer, now having two hidden layers, the first with 128 neurons and the second with 64. LGBM applies a tree dropout to prevent overfitting, and XGB has a decreased learning rate.

Table 4 - Iteration 2 train results for all models and datasets

Dataset	MSE				LogLoss			
	RF	NN	LGBM	XGB	RF	NN	LGBM	XGB
Original	0.0772	0.0803	0.0820	0.0716	0.2767	0.2833	0.2915	0.2577
Original with FR	0.0787	0.0806	0.0822	0.0734	0.2807	0.2845	0.2910	0.2630
7-NN	0.0642	0.0816	0.0815	0.0678	0.2491	0.2880	0.2888	0.2368
7-NN with FR	0.0652	0.0823	0.0811	0.0572	0.2530	0.2904	0.2875	0.2008

Table 5 - Iteration 2 validation results for all models and datasets

Dataset	MSE				LogLoss			
	RF	NN	LGBM	XGB	RF	NN	LGBM	XGB
Original	0.0811	0.0802	0.0819	0.0788	0.2880	0.2835	0.2905	0.2788
Original with FR	0.0813	0.0804	0.0819	0.0792	0.2888	0.2846	0.2902	0.2802
7-NN	0.0799	0.0819	0.0819	0.0768	0.2876	0.2888	0.2908	0.2715
7-NN with FR	0.0813	0.0824	0.0816	0.0774	0.2883	0.2909	0.2894	0.2738

The main goal of the second iteration was to improve results and decrease overfitting. Both were partially achieved. The overfit of RF was generally decreased across all datasets, NN performance had a slight increase, LGBM performance also decreased although improving the overfitting, and XGB had

a slight increase in performance. Hence, with the overfitting problem at an acceptable level, the primary intention for the next iteration is to improve performance overall.

The final manual parameter fine-tuning, the third iteration, changed the criterion from Gini to Entropy in RF. In NN, a third hidden layer was added with 32 neurons. LGBM applied L2 regularization term on weights, and XGB increased the maximum depth of each tree to 10.

Table 6 - Iteration 3 train results for all models and datasets

Dataset	MSE				LogLoss			
	RF	NN	LGBM	XGB	RF	NN	LGBM	XGB
Original	0.0708	0.0800	0.0790	0.0666	0.2539	0.2823	0.2799	0.2442
Original with FR	0.0755	0.0804	0.0798	0.0700	0.2667	0.2840	0.2819	0.2573
7-NN	0.0496	0.0803	0.0594	0.0477	0.1764	0.2831	0.2078	0.1723
7-NN with FR	0.0622	0.0811	0.0709	0.0467	0.2124	0.2860	0.2466	0.1689

Table 7 - Iteration 3 validation results for all models and datasets

Dataset	MSE				LogLoss			
	RF	NN	LGBM	XGB	RF	NN	LGBM	XGB
Original	0.0800	0.0800	0.0792	0.0789	0.2834	0.2831	0.2803	0.2789
Original with FR	0.0805	0.0804	0.0794	0.0793	0.2850	0.2849	0.2812	0.2805
7-NN	0.0793	0.0808	0.0770	0.0771	0.2802	0.2849	0.2722	0.2724
7-NN with FR	0.0799	0.0816	0.0775	0.0777	0.2818	0.2880	0.2743	0.2747

Once again, by checking the LogLoss, it is possible to notice that RF performance improved, but the overfitting also increased. For NN, the performance is slightly better, and the overfitting is still relatively small. Observing the values for LGBM, the performance improved, despite increased overfitting in the 7-NN datasets. Finally, XGB's performance was slightly worse, and the overfitting increased.

From these three iterations, it was possible to obtain information about the model's performance and which ones are expected to yield the best results. Two conclusions were drawn: 7-NN tends to overfit, and Gradient Boosting algorithms tend to perform better. Therefore, the XGB model with the original dataset is considered the best.

It is worth noting that the MSE metric was not disregarded. Since it is calculated using percentage values between 0 and 1, the squared difference outputs minimal values that do not allow for a more straightforward model differentiation. One way to solve this is to consider the RMSE instead of the MSE. Applying the root will yield higher values because the differences are between 0 and 1, allowing us to draw better conclusions. Appendix C is presented the six tables above with RMSE instead of RMSE. In the end, this new metric will also draw the same conclusion given in the previous paragraph.

4.2. GRID SEARCH

The previous sub-chapter concluded that the best model was the XGB with the Original dataset. Improving the results of such model through manual fine-tuning would be time-consuming for the low number of parameter combinations trained during that time. Therefore, a grid search was performed to find the parameters that yield the best results for the metric defined. The grid search will return the top three parameter combinations to evaluate which is the best and infer how much their results differ. Another advantage of considering multiple fine-tuned models is the possibility of constructing an ensemble from them to improve the results further.

The values for the grid search were selected by considering the values used in the manual fine-tuning and the limits of each parameter. The grid search was performed with the following values:

- Number of boosting stages: 2500, 7500, 12500
- Maximum tree depth: 5, 7, 8, 10
- Learning rate: 0.02, 0.05
- Subsample ratio of columns for each tree: 0.3, 0.5

With this grid search, the top model used 12500 boosting stages, a tree depth of 8, learning rate of 0.02, and a column subsample of 0.5. The second model also considered 12500 boosting stages, a tree depth of 8, learning rate of 0.02 and a column subsample of 0.3. Lastly, the third model had 2500 boosting stages, a tree depth of 7, 0.05 as learning rate, and column subsample of 0.3.

Table 8 - Results from the top three model obtained from the grid search considering both the train and validation data

Dataset	Boosting stages	Tree depth	Learning rate	Column subsample	Train		Validation	
					RMSE	LogLoss	RMSE	LogLoss
1st Model	12500	8	0.02	0.5	0.2678	0.2590	0.2806	0.2786
2nd Model	12500	8	0.02	0.3	0.2675	0.2577	0.2808	0.2788
3rd Model	2500	7	0.05	0.3	0.2710	0.2624	0.2809	0.2788

4.3. STACKING ENSEMBLE

Using the three models, it was constructed an ensemble with a GLM. Given the scope of this project is predicted the probability of an event happening, or not, the distribution of the target variable is a Bernoulli, having the following link function associated, g, known as logit:

$$g(\mu) = log\frac{\mu}{1-\mu} \tag{4.3.1}$$

where μ is the mean of the target variable.

With the link function defined, the GLM can be fitted to the predictions from the three models of the grid search considering the target values. This data cannot be fit on the train data as it would result in unwanted bias. Hence, the GLM must be trained on the validation data, which was not used to train

the three models that feed the GLM. From such, we obtain 0.2782 of LogLoss and 0.2804 of RMSE, both on the validation data. This represents a slight increase when compared to the three models.

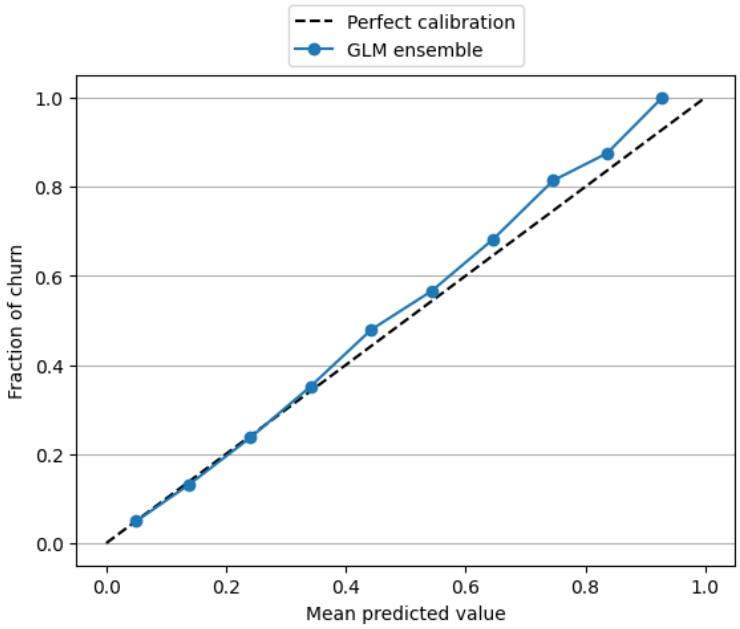

Figure 7 - Reliability diagram for the GLM ensemble. The dashed line represents the perfect calibration where the mean predicted value is equal to the fraction of churn

From the reliability diagram of the GLM ensemble in Figure 7, the model is very close to perfect calibration. Moreover, the ECE value for the ensemble is 0.003289, which is relatively low. This means the model is not heavily uncalibrated.

4.4. CALIBRATION

The model considered for calibration is the 3rd model obtained in the grid search. This was the chosen model because, although it does not have the best values, its performance is very close to the best result and shows a lower overfit when compared to the remaining models. Balancing the criteria of performance and overfitting, the 3rd model is considered the most appropriate choice.

To evaluate how well calibrated the model is, the reliability diagram must be constructed to check how far it is from the perfect calibration.

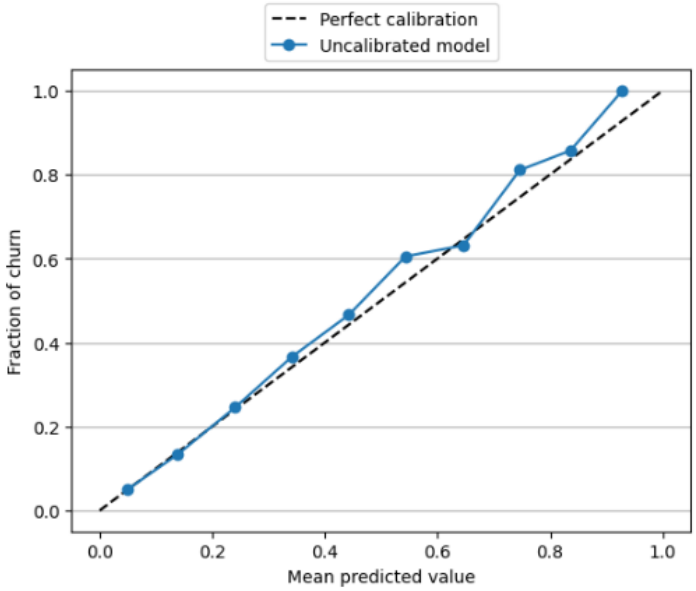

Figure 8 - Reliability diagram for the uncalibrated model. The dashed line represents the perfect calibration where the mean predicted value is equal to the fraction of churn

Figure 8 shows that the model is not poorly calibrated, but some percentages have room for improvement, especially above 0.5. The ECE was determined at 0.003072, which is already relatively low. Therefore, the bins that appear miss-calibrated must have a low number of observations, having a low impact on the weighted average.

Applying the isotonic regression leads to a perfect calibration with ECE of 0, visible in Figure 9, where the calibrated model (orange line) overlaps the perfect calibration (dashed line). This means the isotonic regression is overfitting, which is not due to a lack of available data, given that the validation dataset is above the recommended minimum value of 10 000 observations. The reason lies in the already well-calibrated model that any calibration technique will lead to modified values that do not generalize well.

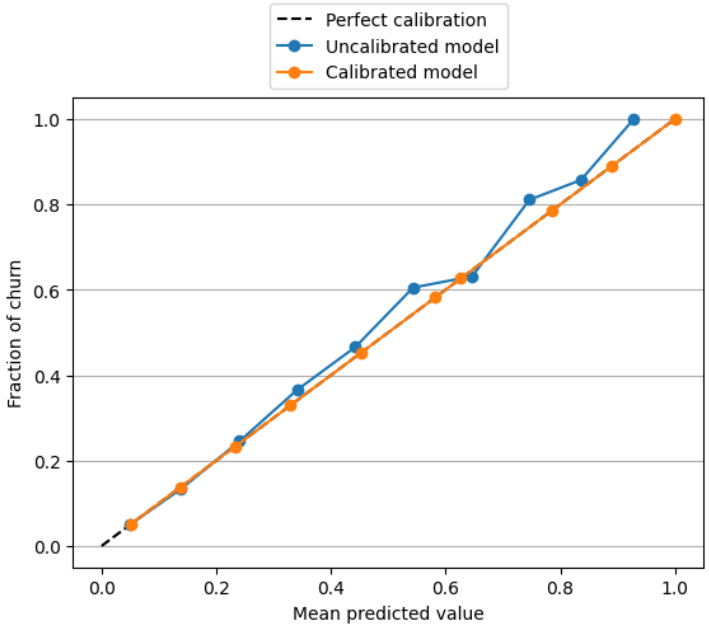

Figure 9 - Reliability diagram for the uncalibrated and calibrated model. The dashed line represents the perfect calibration where the mean predicted value is equal to the fraction of churn

4.5. Model selection

The final model selection is between the 3rd model obtained in the grid search and the GLM ensemble. Both of these models are well calibrated, and their reliability diagrams are similar, as displayed in Figure 10. Considering the ECE, GLM ensemble presents a more excellent value, although by a minimal difference.

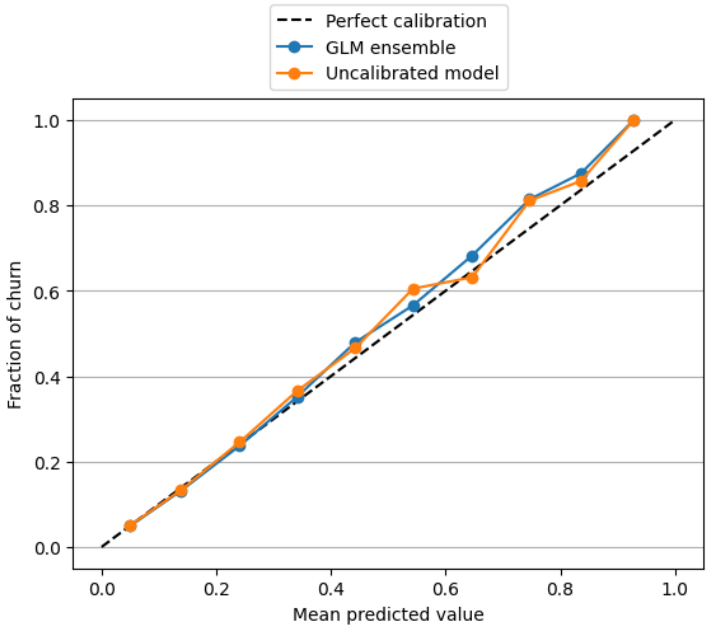

Figure 10 – Overlapping the reliability diagrams of the uncalibrated 3rd model obtained in the grid search and the GLM ensemble. The dashed line represents the perfect calibration where the mean predicted value is equal to the fraction of churn

Given the small difference of ECE and better performance results for LogLoss and RMSE, the GLM ensemble is considered the final model. To evaluate its performance on unseen data, the model will predict on the holdout data from the first three months of 2023. For this data, the model has a LogLoss of 0.2952 and a RMSE of 0.2901. The increase in these metrics indicates overfitting to the data previous to 2023. The figure below shows the reliability diagram of the GLM ensemble applied to the holdout data. For probabilities below 0.6, it is well calibrated. However, for probabilities between 0.6 and 0.9 is overestimating the churn probability, and for probabilities above 0.9 is underestimating.

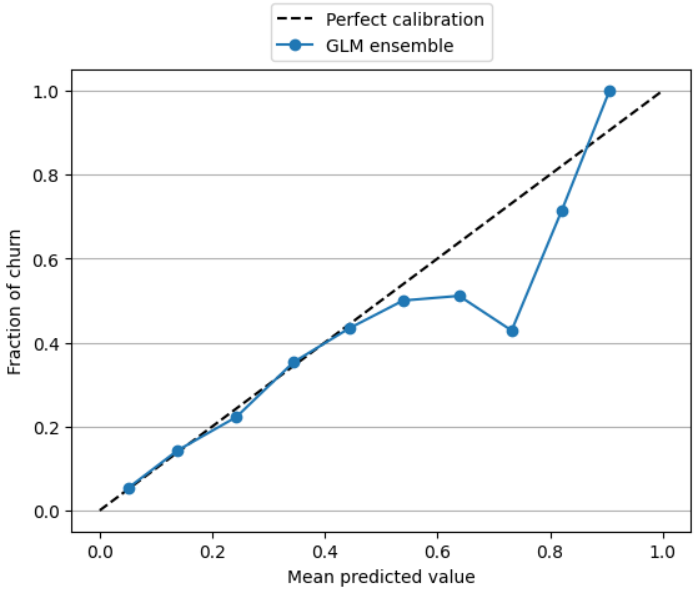

Figure 11 - Reliability diagram for the GLM ensemble. The dashed line represents the perfect calibration where the mean predicted value is equal to the fraction of churn

It is also worth noting the feature importance for the GLM ensemble, which was determined through permutation importance. The top five features are, in decreasing order: premium, days between insurance contract and start, number of previous insurance providers, instalment, and no claims discount. The two variables created during the feature engineering step are included in the top 20.

5. CONCLUSIONS AND FUTURE WORK

This report shows the work developed to create a churn probability model for the auto branch of an insurance company. Below are the drawn conclusions from this project and the improvements that can be made in future works to achieve better results.

5.1. CONCLUSIONS

Reaching this chapter, reflecting on the steps that led here is worthwhile. Two variables were created during feature engineering to describe customer behavior. These two variables ended up in a very favorable position regarding the feature importance of the final selected model. To handle missing values, two techniques were applied. The first was using the median and mode, which yielded great results. The second was using 7-NN that created models overfitted to the train data. This means those models could not generalize, so the datasets with this technique could not be considered for the final selected model. A correlation analysis was performed, and highly correlated variables were identified. From this, a feature removal was applied that performed worse compared to the datasets with all features. After considering the models referenced in the literature, four were chosen. The gradient boosting algorithms, LightGBM and XGBoost, performed the best. From multiple XGBoost algorithms, an ensemble was created using a GLM as the final model to combine the predictions from each algorithm. A calibration technique known as isotonic regression was tested to improve the results. From the reliability diagram and the ECE, it was concluded that the calibration was overfitting. Following the same logic applied to the 7-NN datasets, the calibrated model was not considered for the final selection.

The final model presented a good calibration for the validation data. However, there is room for improvement, especially for the holdout data. More importantly, it must be considered real data was used in this project, not a clean and artificial one. As such, two customers with very similar data can have opposite decisions about churning the company or remaining.

The reliability diagram obtained from the holdout data shows that the model is well-calibrated for smaller probabilities. For higher probabilities, the calibration could be better. More specifically, the model overestimates between probability values of 0.6 and 0.9. Above 0.9, it underestimates. From a business standpoint, it is more critical to avoid underestimation because engaging in action with a customer highly probable to churn, while in fact does not intend to churn, just adds the cost of such action. On the other hand, not engaging with a customer who intends to churn, and an action would revert that decision, represents a more significant loss in revenue for the company. For this reason, the reliability diagram is considered satisfactory since it only underestimates values above 0.9, and the underestimation is not significant.

Another positive aspect is the relevant importance of the variables created during feature engineering. As said previously, those two features (distance to closest insurance intermediary and number of intermediaries in a square of 5 km centered in the client's postal code) were created to describe the customers' surroundings and draw conclusion of their behavior. It is interesting to determine if customers with many intermediaries in the neighborhood are more likely to churn or, in the opposite direction, if customers in rural areas tend to remain with the same intermediary, making it more difficult to convert them to the company.

Customer churn is a concern to be dealt with in any company whose main product or service is highly dependent on individual consumers. The model obtained from this project was a step in addressing this issue since it presents satisfactory results. By using the predicted probability of churn and considering other customer information, the company could make a more informed decision regarding what actions to take to reduce policy cancelation, i.e., customer churn in an insurance company.

5.2. Future Work

The delivered solution to this project has space for improvement, given that the performance metrics and calibration results can improve. This improvement is highly dependable on the data used to build the models and how that data is handled from extraction, passing through all the transformations, and finally being fed to train the models.

Since feature-engineered features are relevant to the final model, more features describing the customer's environment and behavior could be added from the initial variable selection. Therefore, more external data could be considered. For example, income aspects of the customer's residential area or even tracking discount campaigns from the competitor companies during the contract period. Still regarding the data preparation, other approaches for the missing values could be used. Also, a less conservative outlier detection could be performed and exclude the observations considered outliers.

Regarding the models, two main improvements could be made. One is considering more models such as CatBoost, because it is another gradient boosting algorithm, or even Support Vector Machine, which is used in various literature referenced in Chapter 2. The other change is in the fine-tuning of the models. In this report, the fine-tuning presented is quite limited, and the chosen models have a variety of parameters to be adjusted. Studying many parameters' combinations is time-consuming but can yield better results.

Another recommendation is regularly retraining the model to access the most recent customer data. The reliability diagram of the holdout data showed a slight difference in calibration compared to the older validation data. Such difference can increase in the future. Therefore, regularly retraining the model will ensure the predictions align with the current customer behavior, avoiding errors due to a data shift.

The final recommendation is in a long-term timeframe. Since the predicted probability will be used to decide what actions to take, given more time, new data will be available concerning the customer churn for each action. Once a considerable amount of data is collected, it will be possible to create a model to predict the best action for a new customer. This includes if no action is recommended at all.

BIBLIOGRAPHY

Agrawal, S., Das, A., Gaikwad, A., & Dhage, S. (2018). Customer Churn Prediction Modelling Based on Behavioural Patterns Analysis using Deep Learning. *2018 International Conference on Smart Computing and Electronic Enterprise (ICSCEE)*, 1-6.

Ajayakumar, J., Curtis, A., & Curtis, J. (2023). *The utility of Zip4 codes in spatial epidemiological analysis*. PloS one, 18(5), e0285552. https://doi.org/10.1371/journal.pone.0285552

Almeida, M., Mota, M., Souza, W., Nicolau, M., Luz, E., & Moreira, G. (2022). A Temporal Approach to Customer Churn Prediction: A Case Study for Financial Services. *Anais do XIX Encontro Nacional de Inteligência Artificial e Computacional*, (pp. 83-94). Porto Alegre: SBC.

Bella, A., Ferri, C., Hernandez-Orallo, J., & Ramírez-Quintana, M. (2009). Calibration of Machine Learning Models. Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques. 10.4018/978-1-60566-766-9.ch006.

Bentley, J. (1975). Multidimensional Binary Search Trees Used for Associative Searching. *Communications of the ACM*, 18, 509-517

Bishop, C. M. (2007). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer. 203-206, 209. ISBN: 0387310738

Bogomolov, Y., He, M., Khulbe, D., & Sobolevsky, S. (2021). *Impact of income on urban commute across major cities in US*. Procedia Computer Science. 193. 325-332. 10.1016/j.procs.2021.10.033

Breiman, L. (2001). Random Forests. Machine Learning. 45. 5-32.

Britannica, T. Editors of Encyclopaedia (2023, April 14). Euclidean distance. Encyclopedia Britannica. https://www.britannica.com/science/Euclidean-distance

Brummer, N., & Preez, J. D. (2013). The PAV algorithm optimizes binary proper scoring rules

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 785-794).

Cutler, D., Edwards, T., Beard, K., Cutler, A., Hess, K., Gibson, J., & Lawler, J. (2007). Random Forests for Classification in Ecology. *Ecology*. 88. 2783-2792.

DeGroot, M., & Fienberg, S. (1982). The comparison and evaluation of forecasters. Statistician, 32, 12–22.

De Jong, P., & Heller, G. (2008). *Generalized Linear Models for Insurance Data* (International Series on Actuarial Science). Cambridge: Cambridge University Press. 64-65. doi:10.1017/CBO9780511755408

de la Llave, M. Á., López, F. A., & Angulo, A. (2019). The impact of geographical factors on churn prediction: an application to an insurance company in Madrid's urban area. *Scandinavian Actuarial Journal*, *2019*(3), 188-203. DOI: 10.1080/03461238.2018.1531781

de Leeuw, J., Kurt, H., & Mair, P. (2009). *Isotone Optimization in R: Pool-Adjacent-Violators Algorithm (PAVA) and Active Set Methods*. Journal of Statistical Software. 32. 10.18637/jss.v032.i05.

Eurotax. (2023, July 12). *Eurotax* https://eurotax.pt/produto/eurotax/

Friedman, J., Bentley, J., & Finkel, R. (1977). *An Algorithm for Finding Best Matches in Logarithmic Expected Time*. ACM Transactions on Mathematical Software, 3, 209-226

Gregory, B. (2018). Predicting customer churn: Extreme gradient boosting with temporal data. *arXiv preprint arXiv:1802.03396*.

Günther, C., Tvete, I., Aas, K., Sandnes, G., & Borgan, Ø. (2011). Modelling and predicting customer churn from an insurance company. Scandinavian Actuarial Journal 2011. 1–14. 10.1080/03461238.2011.636502.

Guntuboyina, A., & Sen, B. (2017). *Nonparametric Shape-Restricted Regression*. Statistical Science. 33. 10.1214/18-STS665

Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. (2017). On Calibration of Modern Neural Networks

Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction* (Vol. 2, pp. 337-338, 353-357). New York: springer.

Haykin, S. S. (2009). *Neural networks and learning machines*. Upper Saddle River, NJ: Pearson Education. pp. 10-15, 18-20, 22

Huigevoort, C. (2015). Customer churn prediction for an insurance company. Master's thesis. Eindhoven University of Technology

Jekeli, C. (2006). *Geometric reference systems in geodesy*. Division of Geodesy and Geospatial Science, School of Earth Sciences, Ohio State University

Karney, C. (2013). *Algorithms for geodesics*. Journal of Geodesy. 87. 43-55. 10.1007/s00190-012-0578-z

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T. (2017). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. *NIPS*.

Kwon, H., Park, J., & Lee, Y. (2019). Stacking Ensemble Technique for Classifying Breast Cancer. *Healthcare informatics research*, *25*(4), 283–288. https://doi.org/10.4258/hir.2019.25.4.283

Li, X., Shen, X., Zhou, Y., Wang, X., & Li, T. Q. (2020). Classification of breast cancer histopathological images using interleaved DenseNet with SENet (IDSNet). *PloS one*, *15*(5), e0232127. https://doi.org/10.1371/journal.pone.0232127

Louppe, G. (2014). Understanding Random Forests: From Theory to Practice. PhD thesis. University of Liège

Menon, A. K., Jiang, X. J., Vembu, S., Elkan, C., & Ohno-Machado, L. (2012). Predicting accurate probabilities with a ranking loss. *Proceedings of the 29th International Conference on Machine Learning*. International Conference on Machine Learning, 2012, 703–710

Mercator, P. (2011, December 18). In *Wikipedia*. https://commons.wikimedia.org/wiki/File:Geodetic_coordinates.svg

ML Glossary (2023, June 30) https://ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html

Moore, A. (1991). Efficient Memory-based Learning for Robot Control. PhD thesis. University of Cambridge

Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. MIT press. 583.

Muthupriya, V., Narayanan, R., Nakeeb, S., & Abhishek, A. (2022). Customer churn analysis using XGBoosted decision trees. Indonesian Journal of Electrical Engineering and Computer Science. 25.

National Geodetic Survey (U.S.) (1986). *Geodetic Glossary* (NOAA technical publications). U.S. Department of Commerce, National Oceanic and Atmospheric Administration, National Ocean Service, Charting and Geodetic Services

Ng, K., & Liu, H. (2000). Customer Retention via Data Mining. *Artificial Intelligence Review*. 14 (6). pp. 569–590. https://doi.org/10.1023/A:1006676015154

Niculescu-Mizil, A., & Caruana, R. (2005). Predicting good probabilities with supervised learning. *ICML 2005 - Proceedings of the 22nd International Conference on Machine Learning*. 625-632

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., & Louppe, G. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research. 12. 2825-2830.

Ramos, D., Carneiro, D., & Novais, P. (2020). Using a Genetic Algorithm to optimize a stacking ensemble in data streaming scenarios. AI Communications. 33, 1, 27–40. https://doi.org/10.3233/AIC-200648

Risselada, H., Verhoef, P. C., & Bijmolt, T. H. (2010). Staying power of churn prediction models. *Journal of Interactive Marketing*, *24*(3), 198-208. https://doi.org/10.1016/j.intmar.2010.04.002

Scriney, M., Nie, D., & Roantree, M. (2020). Predicting customer churn for insurance data. In *International conference on big data analytics and knowledge discovery* (pp. 256-265). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-59065-9_21

Senthan, P., Rathnayaka, R.K., Banujan, K., & Kumara, B. (2021). Development of Churn Prediction Model using XGBoost - Telecommunication Industry in Sri Lanka. *2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*, 1-7.

Spiteri, M., & Azzopardi, G. (2018). Customer Churn Prediction for a Motor Insurance Company. *2018 Thirteenth International Conference on Digital Information Management (ICDIM)*, 173-178.

Van den Poel, D., & Lariviere, B. (2004). Customer attrition analysis for financial services using proportional hazard models. *European Journal of Operational Research*, *157*(1), 196-217. https://doi.org/10.1016/S0377-2217(03)00069-9

Vincenty, T. (1975). *Direct and Inverse Solutions of Geodesics on the Ellipsoid with Application of Nested Equations*. Survey Review. 23. 88-93. 10.1179/sre.1975.23.176.88

Zadrozny, B., & Elkan, C. (2002). *Transforming Classifier Scores into Accurate Multiclass Probability Estimates*. Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining

Zhang, K., Fan, W., Buckles, B.P., Yuan, X., & Xu, Z. (2006). Discovering Unrevealed Properties of Probability Estimation Trees: On Algorithm Selection and Performance Explanation. *Sixth International Conference on Data Mining (ICDM'06)*, 741-752.

APPENDIX A – COORDINATES SYSTEMS

GEODETIC AND CARTESIAN COORDINATES

The science of measuring and mapping the Earth's surface is Geodesy. It uses reference systems of coordinates to describe points on the Earth (Jekeli, 2006). One of these reference systems is the geodetic coordinates system, which considers the ellipsoid shape of the Earth. A clear advantage of this reference surface is the, although rigorous, mathematical simplicity, especially when comparing to the geoid. Geoid is the representation of the Earth at mean sea level, accounting with gravity variations and the planet's rotation.

The geodetic coordinates are (National Geodetic Survey (U.S.), 1986):

- Latitude (ϕ): angle between the Equator and the normal to the ellipsoid surface from a point of interest. It is measured in degrees ranging from -90° and 90°. The Equator is considered the origin, being itself at 0° latitude;

- Longitude (λ): angle between Greenwich meridian and the meridian passing through a point of interest. It is measured in degrees ranging from -180° and 180°. The Greenwich meridian is considered the origin, being itself at 0° longitude;

- Height (h): perpendicular distance between ellipsoid surface and a point of interest. For points above ellipsoid surface it has positive values, for point below surface it has negative values, for point on the surface it equals zero.

It is worth noting that the normal to the ellipsoid surface from a point of interest intersects the Earth's center of mass only when the geodetic latitude is 0°, 90° or -90°. This phenomenon occurs because the geodetic latitude's origin is the Equator. Figure 12 provides a visual representation of this event, where P is a point of interest, N is the normal to the ellipsoid surface, O is Earth's center of mass, A is a point on the Equator, and C is the intersection between N and the Equator.

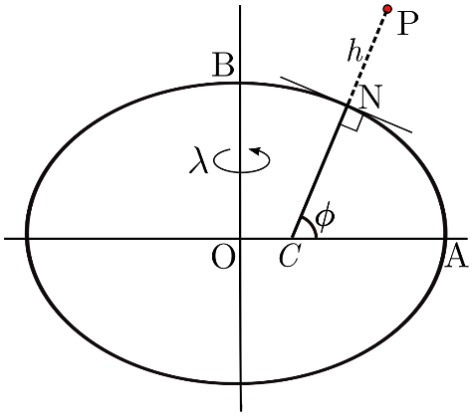

Figure 12 - Geodetic latitude. From Mercator (2011)

Another reference system is the Cartesian coordinates system. This system uses x, y, and z as coordinates to measure the distance between the Earth's center of mass and a point of interest, resulting in the geocentric distance, R:

$$R = \sqrt{x^2 + y^2 + z^2} \tag{A.1}$$

CONVERT GEODETIC AND CARTESIAN COORDINATES

To convert geodetic coordinates (ϕ, λ, h) to Cartesian coordinates (x, y, z), two essential parameters are required to define the ellipsoid: a size parameter and a shape parameter (Jekeli, 2006). The semi-major axis, a, of the elipse will be used as the size parameter. The shape parameter, f, describes the flattening of the ellipse and is determined by:

$$f = \frac{a - b}{a}, \tag{A.2}$$

where b is the semi-minor axis of the ellipse.

Therefore, b can be found by:

$$b = a - af \tag{A.3}$$

The following two values can be determined through a series of calculations outlined in Jekeli (2006). Since such calculations are beyond the scope of this report and to prevent unnecessary intricacy, only the results will be presented.

The first eccentricity, e, which is a measure of deviance from a circular shape with values between 0 and 1, can be determined by:

$$e = \frac{\sqrt{a^2 - b^2}}{a} \tag{A.4}$$

The remaining value to determine is the length of the normal to the ellipsoid surface from the point on the ellipsoid to the minor axis, known as N and represented in Figure 13, is given by:

$$N = \frac{a}{\sqrt{1 - e^2 \sin^2 \phi}} \tag{A.5}$$

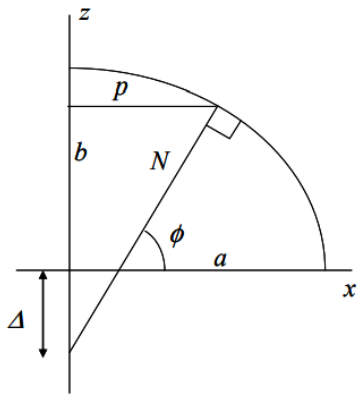

Figure 13 - Prime vertical radius of curvature. From Jekeli (2006)

Now we can determine the Cartesian coordinates:

$$x = (N + h)\cos\phi\cos\lambda \tag{A.6}$$

$$y = (N + h)\cos\phi\sin\lambda \tag{A.7}$$

$$z = (N(1 - e^2) + h)\sin\phi \tag{A.8}$$

DISTANCE BETWEEN COORDINATES

Vincenty's algorithm

In geodesy, there are usually two problems considered when using the distance between two points. The direct problem of locating a point given a distance and direction from another point, and the inverse problem of computing the distance between two points. For the scope of this report, only the inverse method will be used.

Vincenty's algorithm is based on previous works in trigonometry and presents a solution to the inverse problem by allowing to calculate the distance between points in geodetic coordinates through iterative methods. This algorithm considers not only the curvature of the Earth's surface but also, it's ellipsoidal shape (Karney, 2013; Vincenty, 1975). However, it does not take into account the height of the points as it assumes both are on the ellipsoid surface.

The implementation of the algorithm uses only the three basic trigonometric functions (sine, cosine, and tangent). It also uses nested equations, which reduce the computing storage and intermediate results retrieval, leading to a program with reduced length and execution time (Vincenty, 1975). This implementation is available for python and other programming languages.

One downside of Vincenty's algorithm is that the accuracy parameter in the nested equations does not converge for the desired accuracy degree for nearly antipodal points, i.e., points on opposite sides of the Earth (Vincenty, 1975). Therefore, in some cases the algorithm may fail to compute a distance. In the last decade, there has been development to solve this problem by reformulating it as a root-finding exercise (Karney, 2013).

Euclidean

The Euclidean distance determines the shortest distance between two points by measuring the length of the line segment connecting both points. This calculation can be applied in an n-dimensional space. Given the scope of this report, the 3-dimensional formula will be used.

Let $P_1 = (x_1, \mathrm{y}, z_1)$ and $P_2 = (x_2, y_2, z_2)$ be two points in cartesian coordinates. The Euclidean distance between P_1 and P_2 is:

$$d_{euclidean} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2} \tag{A.9}$$

The distance has the same unit as the coordinates.

APPENDIX B – K-D TREE ALGORITHM

The k-d tree is a binary search tree developed by Bentley (1975) as a data structure to organize finite points in a k-dimensional space. It is known to allow an efficient search for the nearest neighbor between a set of points and a new point. This nearest neighbor search has an expected complexity of $O(\log n)$, while the worst-case scenario is $O(n)$, where n is the cardinality of the set of points (Bentley, 1975). Nonetheless, in higher dimensions the search can become inefficient (Moore, 1991).

In k-d tree, each tree node represents a k-dimensional point where the non-leaf nodes generate a hyperplane to divide de dimensional space in two parts. Hence, two subtrees are created from the two parts separated by this hyperplane. For each depth of the tree, an axis of the k-dimensional space is associated, and the generated hyperplane is normal to that axis.

There are two constraints that must be defined to construct a k-d tree: a method to associate an axis to each depth, and how to select a point for each node. There are different forms to define these constraints. The next two paragraphs will present the original constraints created by Bentley (1975).

Regarding the first constraint, one method is to cycle through the axes. For example, in 3-dimensions we have Ox, Oy, and Oz axes. The root node will be x-aligned and the division from the first hyperplane will split this axis. The root's children will be y-aligned, and the root's grandchildren will be z-aligned. The root's great-grandchildren will be x-aligned once again, and so on.

For the second constraint, one way of selecting a point to the node is by determining the median of the points in the subtree. This association leads to a balanced tree, meaning each leaf node is the same distance from the root. The chosen selection method will impact the complexity of nearest neighbor search (Moore, 1991).

CONSTRUCTION OF A K-D TREE

The construction of a k-d tree is a recursive process of partitioning the points in the k-dimensional space. The first step is to select an axis. Then choose the median point along the axis and create the hyperplane to split the space. The remaining points will be separated according to their relative position to the hyperplane:

- Subset of points with axis value minor or equal to the hyperplane will create the left node;
- Subset of points with axis value greater than the hyperplane will create the right node.

This process is reclusively repeated (select a new axis, split points according to new hyperplane) until all points are represented by a node.

For example, consider the following list of 2-dimensional points: (15,17), (2,4), (7,7), (1,13), (3,9), (5,1), (6,8). The first associated axis will be Ox. Now we must determine the median from the x-values of the points. Hence the median will be 5, selecting the point (5,1) as root and defining the hyperplane as $x = 5$. For the remaining six points, we now must compare their x-values to the hyperplane. Since (2,4), (1,13), (3,9) have x-value minor or equal to 5, they will go to the left node. While (15,17), (7,7), (6,8) will go to the right node. We created the first depth with the root's children. Next, we must select a new axis, which will be Oy. Let's consider the left node. The median of the y-values is 9, meaning we select point (3,9) as node and generate hyperplane $y = 9$. Consequently, point (2,4) will go to a new

left node and point (1,13) will go to a new right node. These two last points are in a new depth, meaning they have another axis associated, in this case will be Ox, and the nodes from these points will create hyperplanes according to that axis despite having no other points to generate new nodes. The same logic is followed for the point on the right node in depth 1. Figure 3 display the 2-d tree generated from the example points and the plane splits from the hyperplanes.

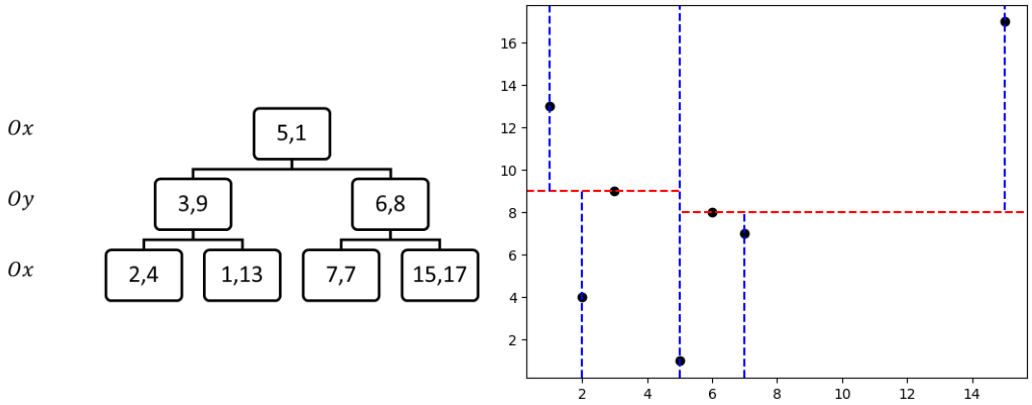

Figure 14 – 2-d tree for the example points and how the tree splits the xOy plane

NEAREST NEIGHBOR SEARCH

K-d trees allow to find which point in the tree is closest to a given point.

The first step is to go through the tree, following the hyperplanes' constraints, until a leaf node is reached. Then the distance between the leaf node and the given point is calculated using Euclidean distance and stored as the current minimum distance.

The second step guarantees the global minimum distance is found. This is achieved by going through the tree in the opposite direction, i.e., from down-up. Now, determine the distance between the given point and the node above the leaf. If it is a better distance, update the current minimum distance. After this, determine the distance between the given point and the hyperplane of the node (Friedman et al., 1977):

- If is better than current minimum distance, there is a possibility of existing better points on the other side of the hyperplane. To investigate this possibility, go through the subtree defined by the other hyperplane side like in the first step. Once reaching a leaf node start the second step from beginning;

- If is worse than current minimum distance, no better points exist on the other side of the hyperplane and continue going up the tree comparing distances to nodes.

Conceptually, the second step uses a hypersphere centered in the given point with radius equal to current minimum distance checking if it intersects the hyperplanes (Moore, 1991). This is implemented by measuring distance between the given point and the hyperplane and comparing it to the current minimum distance.

APPENDIX C - MANUAL FINE-TUNING ITERATIONS WITH RMSE

Below are presented the six tables, based on similar tables presented in Results, with RMSE instead of MSE.

Table 9 - Iteration 1 train results for all models and datasets

Dataset	RMSE				LogLoss			
	RF	NN	LGBM	XGB	RF	NN	LGBM	XGB
Original	0.2771	0.2837	0.2747	0.2702	0.2774	0.2840	0.2678	0.2596
Original with FR	0.2809	0.2846	0.2753	0.2778	0.2813	0.2859	0.2680	0.2746
7-NN	0.2534	0.2863	0.2595	0.2636	0.2197	0.2896	0.2345	0.2429
7-NN with FR	0.2553	0.2876	0.2566	0.2453	0.2226	0.2923	0.2293	0.2103

Table 10 - Iteration 1 validation results for all models and datasets

Dataset	RMSE				LogLoss			
	RF	NN	LGBM	XGB	RF	NN	LGBM	XGB
Original	0.2838	0.2832	0.2813	0.2809	0.2852	0.2835	0.2797	0.2793
Original with FR	0.2844	0.2839	0.2819	0.2814	0.2862	0.2855	0.2812	0.2802
7-NN	0.2827	0.2864	0.2776	0.2772	0.2825	0.2895	0.2723	0.2717
7-NN with FR	0.2830	0.2876	0.2783	0.2781	0.2832	0.2922	0.2742	0.2741

Table 11 - Iteration 2 train results for all models and datasets

Dataset	RMSE				LogLoss			
	RF	NN	LGBM	XGB	RF	NN	LGBM	XGB
Original	0.2779	0.2834	0.2863	0.2675	0.2767	0.2833	0.2915	0.2577
Original with FR	0.2805	0.2839	0.2867	0.2710	0.2807	0.2845	0.2910	0.2630
7-NN	0.2534	0.2857	0.2855	0.2604	0.2491	0.2880	0.2888	0.2368
7-NN with FR	0.2553	0.2868	0.2848	0.2391	0.2530	0.2904	0.2875	0.2008

Table 12 - Iteration 2 validation results for all models and datasets

Dataset	RMSE				LogLoss			
	RF	NN	LGBM	XGB	RF	NN	LGBM	XGB
Original	0.2847	0.2832	0.2861	0.2808	0.2880	0.2835	0.2905	0.2788
Original with FR	0.2851	0.2836	0.2861	0.2814	0.2888	0.2846	0.2902	0.2802
7-NN	0.2827	0.2862	0.2862	0.2772	0.2876	0.2888	0.2908	0.2715
7-NN with FR	0.2852	0.2871	0.2856	0.2782	0.2883	0.2909	0.2894	0.2738

Table 13 - Iteration 3 train results for all models and datasets

Dataset	RMSE				LogLoss			
	RF	NN	LGBM	XGB	RF	NN	LGBM	XGB
Original	0.2660	0.2829	0.2811	0.2581	0.2539	0.2823	0.2799	0.2442
Original with FR	0.2748	0.2836	0.2825	0.2646	0.2667	0.2840	0.2819	0.2573
7-NN	0.2227	0.2833	0.2438	0.2184	0.1764	0.2831	0.2078	0.1723
7-NN with FR	0.2494	0.2847	0.2662	0.2160	0.2124	0.2860	0.2466	0.1689

Table 14 - Iteration 3 validation results for all models and datasets

Dataset	RMSE				LogLoss			
	RF	NN	LGBM	XGB	RF	NN	LGBM	XGB
Original	0.2829	0.2829	0.2815	0.2809	0.2834	0.2831	0.2803	0.2792
Original with FR	0.2837	0.2836	0.2818	0.2816	0.2850	0.2849	0.2812	0.2805
7-NN	0.2816	0.2843	0.2774	0.2777	0.2802	0.2849	0.2722	0.2724
7-NN with FR	0.2826	0.2857	0.2784	0.2788	0.2818	0.2880	0.2743	0.2747