Using tree-based ensemble methods to improve the B2B customer acquisition process in the fashion industry

Daniel José Canelas Filipe

Master's Dissertation

Supervisor: Prof. José Luís Moura Borges

U. PORTO

FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Mestrado Integrado em Engenheria e Gestão Industrial

2020-06-29

Abstract

In the present competitive business environment, some of the most critical business decisions are related to customer acquisition. At HUUB, a start-up that offers an all-in-one supply chain management platform for brands in the fashion industry, this process is handled by the Sales Team. The role of this team is to contact fashion brands that may be interested in partnering up with HUUB. In the Sales and Marketing world, such brands are often called leads. The main problem with the customer acquisition approach is that the decision of which brands to contact relies heavily on empirical knowledge, resulting in the pursuit of leads that end up not bearing fruit. The primary goal of this project was to suggest a new methodology to help the Sales Team decide which brands have the most potential to become part of HUUB's ecosystem. The proposed methodology is based on the creation of a lead score, a numeric value attributed to each lead that translates its proneness of being converted into a client (the higher the score, the higher the likeliness of conversion). To achieve this, the company provided a dataset containing all the information about past deals. Since no lead scoring methodology was implemented at HUUB, a new numeric lead score scale was developed based on the outcome of the previous deals depicted in the dataset. With all the past deals scored according to this new scale, all the data was then used to train machine learning regression algorithms to make the lead score predictions.

One of the most prominent issues with the data provided was its large amount of missing values. Hence, three different missing value imputation techniques were analyzed - *k*-NN, MICE and Mean/Mode. Throughout this process, a novel methodology to evaluate the accuracy of imputation of the *k*-NN algorithm was developed, to allow tuning the *k* parameter in a more effective manner. The missing values issue proved to be especially concerning on one of the features in the dataset - *revenue*. This feature is a numeric variable that states the revenue yielded by a deal, and given it was an important input for the lead scoring model, its imputation was treated as a separate regression problem.

Given the two regression problems presented (revenue and lead score predictions) had the same missing data complications, a generalized automated machine learning tool capable of combining the best imputation methods with the best regression algorithms was developed, entitled Regression Models with Imputation Tool (RMIT). In what concerns the regression algorithms used by the tool, this project focused on applying tree-based ensemble methods (Bagging and Boosting) due to their ability to combine several models to produce the best results. To that extent, Random Forest, Adaptive Boosting and Gradient Boosting were the selected algorithms, as well as the *k*-NN, that acted as a single model baseline comparison.

Results showed that predicting revenue missing data using the RMIT prior to the final lead scoring prediction produced the lowest error values. The suggested lead scoring methodology is expected to result in more leads converted to clients, hence increasing HUUB's revenue.

Resumo

No mundo empresarial atual, algumas das decisões de negócio mais críticas estão relacionadas com a aquisição de clientes. Na HUUB, uma start-up que oferece uma plataforma de gerenciamento da cadeia de abastecimento para marcas da indústria da moda, este processo é tratado pela Equipa de Vendas. O objetivo desta equipa é entrar em contacto com marcas de moda que possam estar interessadas em fazer parceria com a HUUB. No mundo de Vendas e Marketing, estas marcas são frequentemente chamadas de *leads*. O principal problema com a abordagem de aquisição de clientes é que a decisão de quais marcas contactar depende muito do conhecimento empírico, resultando na procura de *leads* que se revelam infrutíferas. O objetivo principal deste projeto foi sugerir uma nova metodologia para ajudar a Equipa de Vendas a decidir quais as marcas com maior potencial para se juntarem à HUUB. A metodologia proposta foi baseada na criação de *lead score*, um valor numérico atribuído a cada *lead* que traduz a sua propensão de ser convertido num cliente (quanto maior o score, maior a probabilidade de conversão). Para esse efeito, a empresa forneceu um *dataset* contendo todas as informações sobre deals anteriores. Como nenhuma metodologia de *lead score* estava implementada na HUUB, uma nova escala foi desenvolvida com base nos resultados dos *deals* anteriores descritos no *dataset*. Com todos os deals anteriores pontuados de acordo com esta nova escala, os dados foram usados para treinar algoritmos de regressão de *machine learning* para fazer previsões de pontuação de *leads*.

Uma das questões mais complexas do dataset apresentado foi a grande quantidade de *missing values*. Assim, foram analisadas três técnicas diferentes de imputação dos mesmos - *k*-NN, MICE e Média/ Moda. Ao longo deste processo, foi desenvolvida uma nova metodologia para avaliar a *accuracy* da imputação usando o algoritmo *k*-NN, de modo a permitir o ajuste do parâmetro *k* de maneira mais eficaz. A problemática dos *missing values* provou ser especialmente preocupante numa das *features* do *dataset* - a *revenue*. Esta *feature* numérica indica a receita gerada por um determinado *deal* e, dado ser um *input* importante para o modelo de previsão *lead score*, a sua imputação foi tratada como um problema de regressão separado.

Dado que os dois problemas de regressão apresentados (previsões de *revenue* e *lead score*) tiveram as mesmas complicações de *missing values*, foi desenvolvida uma ferramenta de *machine learning* generalizada capaz de combinar os melhores métodos de imputação com os melhores algoritmos de regressão, intitulada *Regression Models with Imputation Tool* (RMIT). No que diz respeito aos algoritmos de regressão usados pela ferramenta, este projeto concentrou-se na aplicação de métodos *ensemble* baseados em árvores de decisão (*Bagging* e *Boosting*), devido à sua capacidade de combinar vários modelos para produzir os melhores resultados. Deste modo, *Random Forest*, *Adaptive Boosting* e *Gradient Boosting* foram os algoritmos selecionados, assim como o *k*-NN, que atuou como modelo comparativo, por se tratar de um modelo simples.

Os resultados mostraram que a previsão de *missing values* da *revenue* usando o RMIT antes da previsão final de *lead score* produziu os menores valores de erro. Espera-se que metodologia de *lead score* sugerida se traduza num aumento do número de *leads* convertidos em clientes, aumentando assim a receita da HUUB.

Acknowledgments

It is not often that one has the opportunity to publicly express gratitude towards the people that impact our lives. Given the importance of this Chapter, I will proceed its writing in Portuguese, since I believe it conveys the message in the most meaningful way.

Ao Professor José Luís Borges pelo apoio prestado ao longo de todo o projeto. A prontidão das suas respostas no esclarecimento de dúvidas, bem como as palavras de positivismo e encorajamento dirigidas, em muito contribuíram para a realização desta dissertação.

À HUUB, pela oportunidade que me foi dada, e por me dar a conhecer o mundo da Inteligência Artificial. Dirijo uma especial palavra de apreço à Cristina, que ao longo dos últimos meses se mostrou incansável, acompanhando de perto todo o meu percurso. O seu contributo foi essencial para desbloquear todas as complicações que surgiram ao longo do desenvolvimento do projeto. Deixo, também, o meu agradecimento ao Tony por estar sempre disponível para responder às minhas dúvidas e por me mostrar que existem pessoas no mundo que levam a questão da privacidade *online* muito a sério.

A todos os amigos e colegas que me acompanharam ao longo deste percurso académico.

À Liliana e à Mariana, por me mostrarem que há pessoas para as quais a definição de amizade parece ser pequena. Nesta jornada alimentada a cafeína e a quantidades de açúcar suficientes para levar qualquer pessoa a um estado pré-diabético, não podia ter escolhido melhores companheiras. Não sei o que o futuro nos reserva, mas se somos à prova de pandemia, somos à prova de tudo. Vocês fazem com que toda esta caminhada seja muito mais que um diploma.

Aos meus pais, por nunca duvidarem de mim e sempre me apoiarem ao longo de toda a minha vida. Sem vocês, nada disto seria possível, e serei eternamente grato por me terem dado esta oportunidade. Agradeço também ao meu irmão, que para além de companheiro de vida, partilhou comigo grande parte deste percurso académico e fez com que tudo ficasse mais fácil.

A todos, o meu muito obrigado.

"Blessed is he who expects nothing, for he shall never be disappointed. "

Alexander Pope

Contents

Acronyms and Symbols

AI	Artificial Intelligence
B2B	Business-to-Business
B2C	Business-to-Consumer
ML	Machine Learning
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
RMSE	Root Mean Square Error

List of Figures

List of Tables

Chapter 1

Introduction

Customer acquisition is one of the main concerns of marketing teams in modern organizations. During this phase, companies gather information about potential clients and try to use it to target the ones that are more susceptible to accept a business offer. In order to make this process time and cost-effective, organizations have been using multiple lead scoring methodologies. Lead scoring can be defined as the general procedure applied by organizations in prioritizing which customer leads to target (Nygard, 2019). Lead scoring procedures involve attributing a score (quantitative or qualitative) to potential customers, based on how interesting they are. After potential clients are scored and ranked, companies can then focus their efforts on targeting customers with higher lead scores. It is particularly important to have an accurate lead score methodology implemented in organizations when there is a vast amount of potential clients to analyze.

Traditionally, this score is calculated relying on human knowledge and experience. Nowadays, with the development of new technologies and with the quick rise of machine learning, companies are starting to use data to more accurately predict which customers are more receptive to a given marketing campaign. Having an automated approach to lead scoring can not only help to save time but also to reduce variance and bias introduced by the human judgment used in the traditional approach.

This project addresses a novel methodology for lead scoring to be implemented by HUUB, a tech-based Portuguese start-up which works in the areas of logistics and supply chain management for brands in the fashion industry.

1.1 Company overview

HUUB, founded in 2015 by Luis Roque, Tiago Craveiro, Pedro Santos and Tiago Paiva, is a Portuguese startup whose purpose is to simplify the supply chain management for brands in the fashion industry, from the contact with the suppliers to dealing with the final customer. The services provided by the company include production follow up, storage, stock management and capillary distribution for retailers/final customers. HUUB operates both in wholesale and ecommerce, involving a myriad of stakeholders such as brands, suppliers, carriers and customers. These stakeholders can have an overview of the entire supply chain through the company's full-scale logistics service and an all-in-one management online platform entitled SPOKE.

Furthermore, the company collects, stores and analyses operational information, elaborating reports that allow customers to make supported business decisions. HUUB is also a data-driven organization, since it converts data into business insights, thus helping companies to boost their growth.

The company revenue stream encompasses two distinct flows. On the one hand, HUUB defines a standardized price per product transacted which guarantees the proper functioning of its logistics operations. On the other hand, the company also sets a custom transportation price, to be set individually with each brand, at the beginning of every season.

Despite being a recent company, HUUB has been able to expand its business internationally and operates in three warehouses, two in Maia (Portugal) and one in the Netherlands. This geographical spread allows the company to reduce transportation costs, once it can be closer to the final customer, and also makes it more attractive for brands with an international market. HUUB is currently planning its series A round of investing to gather funds from venture capitalists, which will allow the organization to grow even further.

1.2 Project Description

The company's fast growth over the last years has been accompanied by an increased amount of work to be performed by its Marketing & Sales department. Customer acquisition is one of the focuses of this department, and since one of the primary goals of the company is to scale its business, the amount of companies to contact has been growing. Such growth led to an increased necessity for a strategy to target customers who are more prone to accept a business offer. Currently, HUUB contacts brands based on the knowledge and experience of the sales team employees, who decide whether or not a brand may be receptive to make a deal. However, this process introduces a lot of variability and bias since it relies on human judgment to make decisions. Hence the necessity of developing a new and automated way to target potential customers, not only to standardize the process, making it less susceptible to human error, but also to increase the percentage of successful deals made.

Throughout the journey from potential to converted client (where the brand chooses to work with HUUB), data is generated and stored. However, this data is currently not being used for any purpose, and its exploration may result in valuable insights for HUUB. The database stores not only features that contain characteristics about the brand, such as its country of origin, country of production or the sales channels it operates in, but also about the stages of the customer acquisition process in which negotiations stopped progressing. One of the main features HUUB is interested in knowing is the potential revenue a given deal may bring to the company, since that may be a decisive factor when pursuing a possible customer. However, such information is usually unknown until the brand reaches more advanced stages in the negotiation.

The main goal of the project is to use the information available to implement a new methodology that will allow HUUB to filter its potential customers according to the characteristics they exhibit. This will be achieved through the implementation of a machine-learning powered lead scoring model which will take information about a brand and output a numeric value that translates its proneness to reach a certain stage in the customer acquisition process. The higher the lead score value, the more likely a brand is to reach more advanced stages in the process, and ultimately be converted to a client. In addition, given that HUUB puts a special emphasis on knowing the expected revenue yielded from a potential partnership with a brand and once it is one of the inputs of the lead scoring model with the larger amount of missing values, a predictive model was also developed in order to get more accurate predictions of this variable. The ultimate goal of having good revenue predictions is to improve the results of the lead scoring model.

1.3 Thesis outline

The current chapter describes the project and its goals, and gives a brief overview about the company and its area of operation. It is key to have a clear view on the company foundations and what benefits may come with the implementation of this project.

The following chapter will focus on reviewing the existing literature on the topics addressed throughout the dissertation, while providing the theoretical background on the relevant subjects for this work.

Chapter 3 starts by detailing HUUB's value proposition, so one can better understand the relevance of the project. The current customer acquisition process is then thoroughly explained.

Chapter 4 characterizes in detail the dataset provided, as well as the data preprocessing steps necessary to make data suitable to be used by the proposed machine learning model, which is then described.

Chapter 5 depicts the results obtained from the application of the proposed model, showing its performance in two distinct scenarios.

The sixth and final chapter, "Conclusions and Future Work", contains a summary and a reflection on the findings of this thesis, as well as future improvements that would be complementary to the study developed over this dissertation.

Chapter 2

Theoretical Background

2.1 Lead generation and lead scoring

Organizations are moving towards more analytical and senior-management focused aspects of selling as the market moves from a goods-dominant logic to a service-dominant one(Terho et al., 2015). Customer relationship management systems and marketing automation software have become popular tools for companies with sales and marketing teams (Duncan and Elkan, 2015). These systems are capable of storing a large amount of historical sales data, thus providing great potential for machine learning algorithms to improve the sales process. The sales process can be represented as a sales funnel which represents the stages that must successfully be completed by any sales organization in order to close a sale, ultimately bringing revenue to the company (Söhnchen and Albers, 2010). According to Duncan and Elkan (2015), a typical sales funnel has five different stages: awareness, lead generation, transformation into a marketing-qualified lead (MQL), conversion of a MQL into a sales-qualified lead (SQL) and the final stage where the deal is closed. To separate leads into appropriate "buckets" most companies develop some sort of system for lead ranking or sorting, often called lead scoring. Lead scoring is a method of ranking leads which assigns a numerical score to a potential client, and then pushes the lead through to the appropriate next step based on the score (Stevens, 2011). Traditionally, in order to move across the pipeline previously mentioned, decisions rely heavily on humans who use their knowledge and experience to qualify leads. However, a salesperson with a rich pipeline of qualified potential clients has to make decisions on a daily, or even hourly, basis as to where to focus their time when it comes to closing deals to hit their monthly or quarterly quota (Antonio, 2018). Thus, speed is a critical element in this prospecting stage.

With AI, an algorithm can compile historical information about a client, along with social media postings, and the salesperson's customer interaction history to rank the opportunities or leads in the pipeline according to their chances of closing successfully (Antonio, 2018). Resorting to AI, predictive algorithms can be developed to engage in lead scoring. AI systems can analyze previous prospect data and determine which potential clients have the highest probability of converting to effective clients. Algorithms that use e-commerce sales data usually yield better results, since there is a larger amount of data available from transactions in this sales channel (Syam and Sharma, 2018).

Kim et al. (2005) used a genetic algorithm, called the evolutionary local selection algorithm (ELSA), to do feature selection to identify the feature subset that maximizes classification accuracy. This machine learning approach outperformed both the traditional methods of feature selection (done by principal components analysis) and classification (using logistic regression). Another study by the Harley-Davidson company Power (2017) presents the case of a Harley-Davidson dealership in New York, which through machine learning algorithms, was able to go from one qualified lead per day to 40. This algorithm used data generated through text and visual campaign variables as well as customer variables to predict which online campaigns, implemented through different digital channels (SMS text, email, search, display, social media, etc.), were most likely to convert different customer segments. Within three months of implementing this machine learning based lead generation and qualification program, the dealership's qualified leads had increased 2930%.

AI's main contribution to lead generation is the ability to target customers individually, with highly personalized, tailored advertising and marketing. After the leads are generated, qualified and the optimized contact strategy is determined, AI can uncover patterns in data and provide information about when and how prospects should be contacted, thus placing more leads in the funnel and increasing productivity (Syam and Sharma, 2018).

2.2 Data preprocessing

Data preprocessing is a data mining technique used to transform raw data in a useful and efficient format, which can be used by machine learning algorithms. Real-world data is often incomplete and inconsistent and is likely to contain many errors, hence the need to preprocess it. Data preprocessing includes several steps, from data cleaning to feature normalization and outlier analysis, depending on the type of features in the dataset in study. The following Section focuses on describing in detail an important step in data preprocessing - handling missing values.

2.2.1 Dealing with Missing Data

Missing data is a common problem in practical data analysis. To tackle this, one has to identify missing values in a data set and treat them in such a way that the minimum amount of information is lost. Missing values treatment is to be applied to data before it can be used as input to a machine learning algorithm. This is an important step since missing values in data can reduce the power of the model and can result in wrong inferences, thus leading to wrong predictions and classifications (Kapil, 2018).

There are three main ways in which data can be missing from a data set: missing completely at random (MCAR), missing at random (MAR) and not missing at random (NMAR). Data missing completely at random (MCAR) is independent from any variable observed in the data set. This means that the probability of being missing is the same for all cases (Jonsson and Wohlin, 2004).

An example of data MCAR would be a weighing scale running out of batteries. In such case, data would be missing simply due to bad luck. Missing at random (MAR) means that the missing data may depend on variables observed in the data set, but not on the missing values themselves (Jonsson and Wohlin, 2004). Turning back to the weighing scale analogy, this would happen if the amount of missing data produced when the scale was put on a hard surface differed significantly from when it was laid on a softer one. The third and final type of missing data occurs when data is not missing at random (NMAR). In this situation, missing data depends on the missing values themselves, and not on any other observed variable (Jonsson and Wohlin, 2004). MNAR means that the probability of being missing varies for reasons unknown to the researcher (van Buuren, 2018). For example, the weighing scale mechanism may wear out over time, producing more missing data as time progresses.

Generally, there are two broad missing data management methodologies. The simplest methodology includes techniques that omit the missing data. Such techniques, reduce the size of the data set, hence reducing computational requirements, but can also affect the models' accuracy (Cheliotis et al., 2019). The alternative to this method is to replace the missing values with an estimate. In data science terminology, this process of estimating missing values is often called *imputation*.

2.2.2 Mean, median and mode imputation

A quick approach to missing values is to replace them with the mean, median or mode. In mean imputation the mean value of each non-missing variable is used to fill in missing values for all observations. Median imputation follows the same rule, but uses the median instead of the mean to fill the missing values. These methods are used when the variables at hand are numeric. When the variables to impute are categorical, the most frequent value (mode) can be used to replace missing data. Mean, median and mode imputations are simple, but they underestimate variance and ignore the relationship with other variables (Zhang, 2016).

2.2.3 k-Nearest Neighbours (k-NN) Imputation

In the k-NN imputation method, missing values are imputed using the *k* nearest neighbours, as the name suggests. This method takes the *k* closest data points to the observation with missing data and imputes it based on the the non-missing values in the neighbours. The nearest, more similar neighbours are found by minimising a distance function.

Strike et al. (2001) and Troyanskaya et al. (2001) recommend the use of the Euclidean distance as the distance function, defined as:

$$E(a,b) = \sqrt{\sum_{i \in D} (x_{ai} - x_{bi})^2} \tag{2.1}$$

where:

$E(a,b)$, is the distance between the two cases *a* and *b*
x_{ai} , is the value of the attribute for case *a*
x_{bi} , is the value of the attribute for case *b*, and
D , is the set of attributes with non-missing values in both cases

As an example, consider the dataset presented in Table 2.1, from which we want to estimate the distance between the Hoodie and the Jacket. One can see that the attributes for which both the Hoodie and the Jacket have values are *Height* and *Percentage of cotton*, thus defining *D*. Since they are not a part of *D*, *Width*, *Length* and *Weight* do not contribute to the distance calculation. This implies that whether a neighbour has values for attributes outside *D* or not, this does not affect its similarity to the case being imputed.

Table 2.1: Example of an incomplete data set

Item	**Width (cm)**	**Length (cm)**	**Height (cm)**	**Weight (g)**	**% cotton**
T-Shirt	10	10	2	100	50
Hoodie	20	-	5	400	70
Jacket	-	-	7	-	60
Polo Shirt	10	-	-	-	-

Hence, using Equation 2.1, we find that

$$E(Hoodie, Jacket) = \sqrt{(5-7)^2 + (70-60)^2} \approx 10.2 \tag{2.2}$$

The previous example shows how the distance is calculated between two instances. The procedure is repeated for every instance in the dataset, but ultimately only the *k* nearest data points are considered. Finally, once the value of *k* is set, a replacement value to substitute the missing attribute is estimated. This estimation depends on the type of missing value: for discrete data, the most frequent value presented in the neighbours is used as the imputed value, whereas for continuous data the mean is typically used (Monard, 2002).

An important parameter to tune in this method is the value of *k*. Duda and Hart (1973) suggest the use of $\sqrt{N}$, where *N* corresponds to the number of instances in the dataset (in the example presented in Table 2.1, $N = 4$). Jonsson and Wohlin (2004) point out the importance of selecting the correct value of *k* when imputing data, once as *k* approaches *N*, the method converges to ordinary mean imputation.

2.2.4 Multivariate Imputation by Chained Equations (MICE)

In large data sets it is common for missing values to occur in several variables. Multiple Imputation by Chained Equations (MICE) is a practical approach to generate imputations based on a set of imputation models, one for each variable with missing values. Initially, all missing values are filled in by using values derived solely from the non-missing values available (this initial imputation can be done using the mean, for example) (Azur et al., 2011). Consider the first variable with missing values, x_1. The instances with missing values for x_1 are set back to missing and then imputed based on a regression performed with all the other variables, $x_2,...,x_k$. In other words, x_1 is the dependent variable in a regression model and all the other variables are the independent variables.

These regression models operate under the same assumptions one would make when performing linear, logistic, or Poison regression models outside the context of imputing missing data (Azur et al., 2011). The process is repeated for all other variables with missing values, following the aforementioned approach. This is called a cycle. At the end of each cycle, all of the missing values have been replaced with predictions from regressions that reflect the relationships observed in the data. In order to stabilize the results, the procedure is usually repeated for several cycles. The number of cycles to be performed can be specified by the researcher. Raghunathan et al. (2000) suggest using 10 cycles as the optimal number of cycles. Royston and White (2011) state that only if variables with missing values to be imputed are highly correlated are more than 10 cycles needed for convergence. However, this parameter is dependent on the type of problem at hand and should be adapted to it. The idea is that by the end of the cycles, the distribution of the parameters governing the imputations (i.e. the coefficients in the regression models) should have converged in the sense of becoming stable. This will, for example, avoid dependence on the order in which the variables are imputed.

Assessing the convergence of the parameters can be done by comparing the regression models at subsequent cycles. The final imputed dataset refers to the last cycle performed. According to (Madley-Dowd et al., 2019), Multiple Imputation reduces bias even when the proportion of missing values is large.

2.3 Tree-based ensemble methods: Random Forest, Adaptive Boosting and Gradient Boosting

In recent years, ensemble based algorithms have been gaining popularity among practitioners when solving prediction and classification problems. These type of algorithms consist of multiple base models (such as decision trees or neural networks), and each base model provides an alternative solution to the problem. The final model results from the combination of these multiple solutions, usually by weighted/unweighted voting or averaging (Zhang and Haghani, 2015). This idea is often used in our daily lives. Oftentimes our decision-making process is guided by the opinion of various sources, then resulting in a decision that is the weighted combination of all the ideas/opinions gathered. Tree-based ensembles are regarded as one of the best off-the-shelf procedures for both classification and regression tasks (Dawer et al., 2017). The two most successful algorithms on this topic are Random Forest, Adaptive Boosting and Gradient Boosting decision trees. Both methods use a single decision trees as its base. However, whereas the Random Forest method is derived based on the idea of Bagging, detailed in Section 2.3.3.1, Adaptive and Gradient Boosting use the Boosting technique, as explained in Section 2.3.3.2.

2.3.1 Decision Trees: Basic Concepts

Decision trees are widely used in the data science community due to their ability to handle complex problems by providing an understandable representation, easy to visualize and interpret (Amor et al., 2006). This diagrammatic representation looks like an inverted tree with the root at the top and branches spreading underneath, leading to different nodes (Figure 2.1). A decision tree classifies data items by posing a series of questions about the features associated with the items. Each question is contained in a node, and every internal node points to one child node for each possible answer to its question (Kingsford and Salzberg, 2008).

Decision trees are built top-down from a root node and involve partitioning the data into subsets that contain instances with similar values (homogeneous) (Yang, 2019). As the tree grows it can either stop at a leaf node or pass through an internal node. An internal node is a node with outgoing edges, connected to a top parent and to a bottom child, and represents one of the possible choices available at that point in the tree structure. Finally, the lead nodes contain the class labels, and are the final result of a combination of decisions or events (Song and Lu, 2015). Segments of the trees that connect the nodes are designated as branches, and show the flow from question to answer.

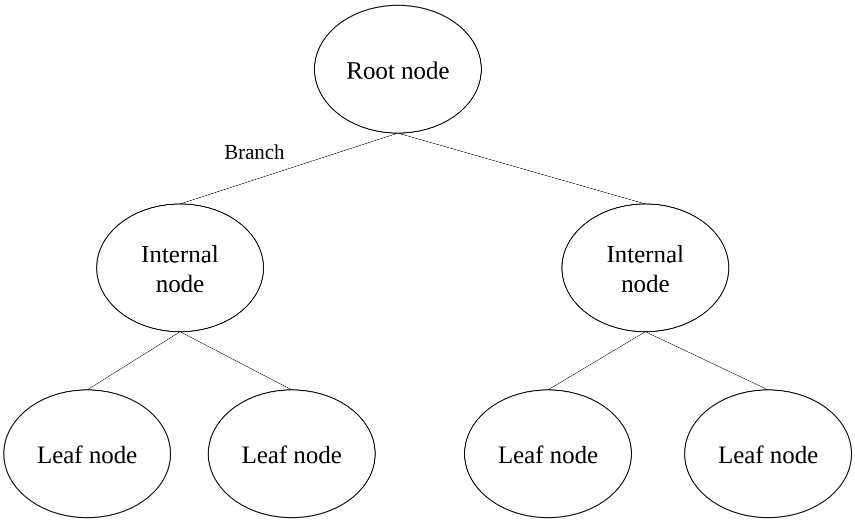


Figure 2.1: Decision Tree Representation

Decision trees can be applied to both regression and classification problems. When the target variable takes continuous values, the tree is called a regression tree; when the variable to predict is categorical, the tree generated is a classification tree.

In order to fully understand how a decision tree works, one must get familiarized with the concepts of nodes and branches, as well as the main steps involved in the algorithm (splitting, stopping and pruning).

2.3.2 Splitting

One of the most crucial steps when building a decision tree is determining which nodes to place at the root and at different levels of the tree as internal nodes. Decisions which generate a simple, compact tree with few nodes are the desired outcome. At each node, the goal is to query the data in such a way that the data reaching subsequent nodes is as ‘pure’ as possible. The node impurity is a measure of the homogeneity of the labels at the node. This concept can be more easily understood taking into consideration the metallic ball analogy. Let a metallic ball represent a dataset and its atoms single instances of that dataset. If all the atoms of the metallic ball are gold, the ball is considered to be purely gold, indicating the highest level of purity (and consequently, an impurity of zero). Similarly, if all the examples in the dataset are of the same class, then the set’s purity is highest. If 1/3 of the atoms are gold, 1/3 are silver, and are 1/3 iron - one would say its purity is lowest. Similarly, if the examples are split evenly between all of the classes, then the set’s purity is lowest.

Distinct metrics can be used for selecting the best split, based on the degree of impurity of the child nodes. One of the most commonly used in classification problems is the Gini Index. If a data set W contains examples from n classes, the Gini Index (W) is computed using the following expression where p_j is the probability of class j:

$$\text{Gini Index (W)} = 1 - \sum_{j=1}^{n} p_j^2$$

After splitting W into subsets, the Gini Index of the split data is defined as the weighted sum of the Gini indices of the sets. The attribute providing the smallest weighted sum is chosen to split the node (Brown and Myles, 2009).

For regression trees, since the target variable is not a class but a continuous value, an impurity metric suitable for continuous variables is required. Reduction in variance is an algorithm that uses the standard formula of variance to select the best split. The split that shows the lower variance is selected as the criteria to split the population (Rokach and Maimon, 2014).

The variance can be computed using the following formula:

$$\text{Variance} = \frac{\sum (x - \bar{x})}{N}$$

Where:

x , is the actual value
$\bar{x}$, is the mean of the values in the node, and
N, is the number of values in the node

2.3.3 Stopping Criteria and Pruning

One of the main issues in learning decision and regression trees is the problem of overfitting the training examples. The two most common approaches to tackle this problem can be either stopping tree growth before all the data is perfectly partitioned, or allow the tree to overfit the data and later prune it. All decision trees need a stopping criteria or it would be possible to grow a tree in which each case occupied its own node. The resulting tree would be computationally expensive, difficult to interpret and would probably not work very well with new data. According to Rokach and Maimon (2014), the most common rules for stopping tree growth are the following:

- All instances in the training set belong to a single value of y.

- The maximum tree depth (length of the longest path from a root to a leaf) has been reached.

- The number of cases in the terminal node is less than the minimum number of cases for parent nodes.

- If the nodes were split, the number of cases in one or more child nodes would be less than the minimum number of cases for child nodes.

- The best splitting criterion is not greater than a certain threshold.

There are some situations where the stopping criteria does not work as expected. Alternatively, it is possible to let the model grow a large tree at first, and then pruning it to optimal size by removing nodes which provide less relevant information (Hastie et al., 2009). Such methods typically use statistical measures to remove the least reliable branches, generally resulting in faster classification and an improvement in the ability of the tree to correctly classify independent test data (Patil et al., 2010).

2.3.3.1 Bagging: Random Forest

Building a single decision tree provides a simple model of the world, but it is often too simple or too specific. With the rapid evolution of data mining techniques, it has become clear that multiple models working together are better than one model doing it all. A Random Forest is a method where a large set of unstable but independent decision trees are aggregated using a majority vote to produce a more accurate classification than a single model.

Random Forests have become a widely used tool due to their high accuracy and ability to handle many features with small samples. The main concepts behind this algorithm are Bagging and Random Selection of Features. Bagging (or bootstrap aggregation), generates training sets of n samples by drawing samples with replacement (the same instance can be chosen more than once) from the original training set. A specified number of such training sets are generated and the response of models trained on these training sets are then averaged to arrive at the bagging response. The concept of random selection of features used in Random Forests can be considered as a form of attribute bagging where classifiers are generated on a random subspace of the original space and then combined (Tin Kam Ho, 1998). This approach is expected to increase independence between the classifiers and the combination of such classifiers can potentially reduce the variance of the integrated classifier and increase generalization accuracy. According to Castelli et al. (2019), given N training samples, M variables and B decision trees in the forest, each one of the B trees is constructed as follows:

1. Choose a training set for a tree by selecting n times with replacement from all N available training cases (bagging). Use the rest of the cases to estimate the error of the tree, by predicting their classes - Out of Bag (OOB) error.

2. For each node of the tree, randomly select m variables (m should be much less than M) on which to base the decision at that node. Calculate the best split based on these m variables in the training set;

3. Each tree is fully grown and not pruned (as in the construction of a regular decision tree).

As described in step 1, the data instances which are not used to train the model are later used to test it. According to Fratello and Tagliaferri (2019), when building the bootstrap dataset for each decision tree, each observation in the original dataset has a probability of $\frac{1}{e} \approx 0.3679$ of not appearing in a given bootstrapped dataset. Moreover, this probability tends to increase as n increases, with n being the number of observations in the original dataset. This means that each decision tree is trained on a dataset which, on average, contains roughly two thirds of the observations in the original dataset, whereas the remaining are replicated observations. The remaining one third of samples belonging to the original dataset, different for each tree, are used to estimate the generalization performance of the tree.

The generalization estimates of all trees of the ensemble are aggregated by averaging the OOB error estimate of the ensemble (Fratello and Tagliaferri, 2019).

2.3.3.2 Boosting Algorithms

The idea of boosting came from the realization that it was possible to use a weak learner and modify it to become a more robust learner. A weak hypothesis or weak learner is defined as one whose performance is only slightly better than random chance. Hypothesis boosting was the idea of filtering observations, leaving the observations the weak learner can handle and focusing on developing new weak learns to handle the remaining more difficult observations (James et al., 2014). Similarly to Bagging, Boosting also uses multiple decision trees. However, the trees are grown sequentially, using information from previously grown trees to improve the model. In Gradient boosting, Shallow trees (trees with relatively few splits) are used as the weak learners (Lantz, 2013).

Figure 2.2 shows a schematic representation of the difference between single trees and Bagging and Boosting procedures.

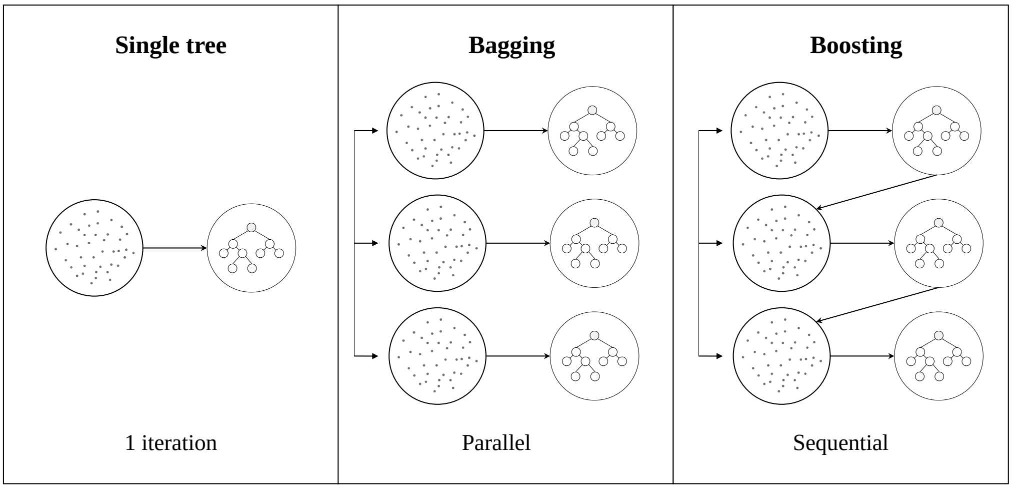


Figure 2.2: Difference between a single tree, bagging and boosting procedures

The first successful application of a boosting algorithm was Adaptive Boosting (AdaBoost). AdaBoost uses decision trees with a single split as weak learners and works by weighting the observations, putting more weight on difficult to classify instances and less on those already handled well (Brownlee, 2016). The manipulation of parameters is adaptive and based on the actual performance in the current iteration: both the weights for re-weighting the data as well as the weights for the final aggregation are re-computed iteratively. Predictions are made by majority vote of the weak learners' predictions, weighted by their individual accuracy (Mayr et al., 2014).

Besides Adaptive Boosting, one other popular Boosting algorithm, developed by Friedman (2000), is Gradient Boosting. This algorithm focuses on minimizing some function of the residuals (typically the Error Sum of Squares (SSE) or the Mean Squared Error (MSE)). In machine learning terminology, this function is called the loss function and is defined as the difference between the predicted and the observed value. The boosting algorithm outlines the approach of sequentially fitting trees to the residuals from the previous tree. This specific approach is how gradient boosting minimizes the mean squared error loss function (for SSE loss, the gradient is nothing more than the residual error).

The way an optimal solution is found is by using gradient descent (hence the name gradient boosting). The main idea of gradient descent is to manipulate parameters iteratively to minimize the loss function. An important parameter in gradient descent is the size of the steps which is controlled by the learning rate. If the learning rate is too small, the algorithm will take many iterations (steps) to find the minimum, thus hindering computational time. On the other hand, if the learning rate is too high it may not detect the minimum thus not giving the desired result (Lantz, 2013).

Chapter 3

Problem Context

Before diving deeper into the problem at hand, a brief overview of HUUB's value proposition is presented, in order to better frame the project. The customer acquisition process is then thoroughly analyzed (AS-IS situation), going into detail on the brand's profiling process, as well as the tools currently implemented to help manage the entire sales pipeline. After the identification of the main improvements to be made to further optimize the ongoing process, a description of the TO-BE situation is given.

3.1 HUUB's value proposition

HUUB manages all the supply chain activities for companies in the fashion industry, from its suppliers to its customers. The company was born to fill the needs of brands in this industry, that not only lack the ability to manage all the supply chain activities, but also struggle to keep up with the market's ongoing digital transformation. This problem is especially prominent in small and medium sized companies, since they are often not able to set-up their own in-house operations due to financial constraints. The services provided by HUUB include production follow up, storage, stock management and distribution for retailers and final customers. Besides handling the physical flow of the products commercialized by the brands, HUUB integrates information related to all stages of the supply chain in the its custom web-based platform, SPOKE. This platform allows brands to access the status of their ongoing operations at any time, in an intuitive and user friendly manner. Some of its features include order tracking, stock management, order history registrations and accounting management.

One other aspect that sets HUUB apart from its competitors is the fact that it offers a set price per item transacted prior to the beginning of each season, allowing brands to have a better overview of their cost structure. This pricing model benefits both the client and HUUB, since the two parties are aligned and benefit from the growth of one another. For instance, consider a warehousing operation. Traditional warehousing firms charge a fixed price per item (per day) to hold inventory, meaning that they make more revenue the longer they hold a brand's inventory. Since HUUB's price per item is independent from the number of days the item stays in the warehouse, the shorter the number of days a product is held there, the more the brand grows and the least HUUB spends in warehousing costs.

In addition, to assure that brands working with HUUB experience continuous growth, the Account Management team provides constant business insights regarding future strategical decisions, in order to facilitate the decision-making process.

3.2 Customer Acquisition Process

The customer acquisition stage is vital in any organization, since it is the one that focuses on the process of bringing new customers to the business. The main goal of this process is to gain information about potential customers, measure their potential value, and allocate resources to acquire those with greater long-term value (Arnold et al., 2011). One should notice that the acquisition process only refers to the act of getting a customer to join, meaning that the analysis of acquired customers that may give up on a deal falls out of the scope of this stage.

HUUB's Sales Team is the one responsible for managing the entire sales funnel. This process is initiated with the generation of leads. Leads are clients that show interest in a company's product or service, and can be viewed as potential customers. In HUUB's case, these clients are brands in the fashion industry. The Lead Generation stage is about finding customers. Interesting prospects are then contacted and move on to the Lead Qualifying stage, in case they engage with the Sales team after that contact. If the brand shows interest after this initial contact and meets certain criteria, it moves on to the Lead Conversion stage, where further details are discussed, objectives are aligned and negotiations occur. Given that all criteria is met in these stages, clients move on the Onboarding (or Integration) phase, where they go through the integration process with HUUB. This entire flow can be better seen as a funnel (or inverted pyramid), that starts off wide at the top, capturing as many leads as possible, and continues on to become narrower at the end – signalling deals that are in the prospect's final decision-making stage (Figure 3.1).

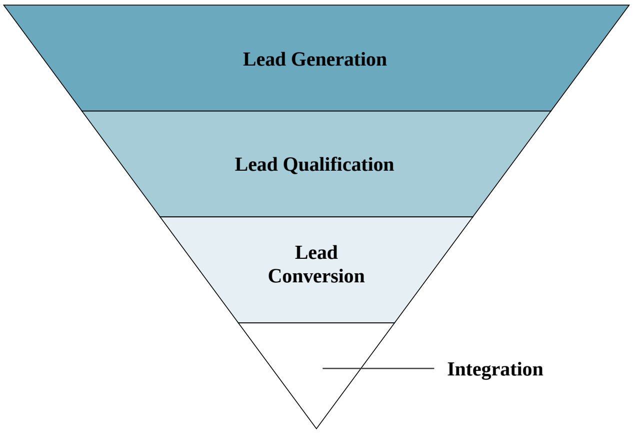


Figure 3.1: HUUB Sales funnel

3.2.1 Managing the Customer Acquisition Process

The four different stages mentioned in section 3.2 are managed by the Sales Team through Hubspot, an online Sales platform that offers tools for customer relationship management and lead generation. The platform displays the different stages of a given funnel in the form of a pipeline, a visual way of tracking multiple potential customers as they progress through different stages in the process. Currently, there are three distinct types of pipelines, the first for the Lead Generation and Qualifying stages, the second for the Lead Conversion Stage and the third for the Integration Stage. The following sections will go into further detail on the Lead Generation and Qualifying Pipeline as well as the Lead Conversion Pipeline, since they are the only ones that refer to the customer acquisition process. The Integration (or Onboarding) Pipeline depicts the entire process of familiarizing a brand with HUUB's service, and contains only brands that are already converted customers. Hence, this Pipeline will not be approached since it is not part of the Customer Acquisition process.

3.2.1.1 Lead Generation and Qualification Pipeline

The Lead Generation and Qualification (LGQ) Pipeline, as the name suggests, encompasses both the generation of leads and its qualification. In Hubspot, this pipeline comprises six different stages: three that represent main processes - Profiling, Campaign and Qualifying - and three others that act as the result of the Qualifying stage, namely the Qualified, Out and Postponed Stages (Figure 3.2).

Profiling Stage

In order to get good leads, it is vital to do a thorough research about potential customers. The process of identifying and describing the profiles of ideal customers, segmented based on different variables is called Customer Profiling. This identification process is underpinned by the characteristics HUUB values the most in a fashion brand.

HUUB currently profiles a brand according to its positioning in the market, size, sales channels, product specifications and target customers. In what concerns market positioning, HUUB distinguishes brands based on its segment - low cost, value, premium or luxury. As for company size, the most relevant indicators to collect are the sales volume (number of items sold in a given season). This last characteristic is difficult to obtain, since it is often not available to the general public.

In any logistics operation, knowing the location of the multiple stakeholders in the value chain is key for efficient planning. Hence, the three fundamental fields to look at are the brand's country of origin, its country of production (country from where they supply their products), as well as the locations of the main target markets. Moreover, since HUUB handles both B2C and B2B processes, knowing which sales channels a given brand operates in is also a valuable information.

In what concerns product related traits, the type of product commercialized by a brand (clothing, swimwear, underwear, footwear,...) is registered. In addition, the predominant age group of the target customer of the brand (usually divided in broad categories such as adult and kids) is also assessed.

With the quick rise of social media platforms as a way of showcasing a company's products, there are some insights that can be obtained from the analysis of these platforms. In the fashion industry, brands usually market their products on Instagram, and parameters such as the number of followers and the number of influential people that represent a brand online are carefully analyzed.

Campaign Stage

After researching about potential prospects, HUUB sends them automated or customized campaigns via e-mail, or contacts them by phone. This stage requires careful planning, since it is many times the first contact point between a potential client and HUUB.

Qualifying Stage

When a brand reacts positively to a given campaign (by answering the email or phone call), it goes through to the Qualifying stage. Qualifying refers to the stage where HUUB gets to know some details regarding a brand's business, introduces them to their value proposition and evaluates the potential of a partnership between both parties. As a result of this stage, a brand may enter one of three different stages: (i) the Postponed Stage, if negotiations are postponed, (ii) the Out Stage, if interest is lost by one of the parties, or the (iii) Qualified Stage, if both parties are aligned and the deal goes through to the Conversion Pipeline. One should notice that qualifying a brand is a two-way evaluation, where both parties assess the suitability of one another in the partnership. Thus, a lead may be disqualified not only if it loses interest in working with HUUB, but also if HUUB finds that working with that brand does not yield any benefits.

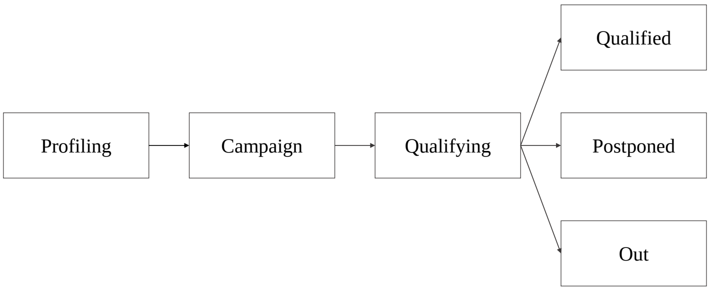


Figure 3.2: Lead Generation and Qualification Pipeline stages

Although every lead can fit in the presented pipeline, the Sales Team felt the need to separate them according to how the first contact was established. There are currently three ways a brand can get in contact with HUUB: (i) willingly, after a partner referral or web research, (ii) as a result of HUUB directly contacting them, or (iii) in events, namely fashion fairs. These three approaches originate three different pipelines, one from the brands that contact HUUB, referred to as the *Organic Pipeline*, one for leads contacted in fairs (*Fairs Pipeline*) and another for the leads that are generated inside HUUB, as a result of research done by the Sales Team, originating the *Inside Sales Pipeline* (Figure 3.3).

It must be noted that this differentiation causes the deletion of some stages inside the Organic Pipeline, since there is no need for Profiling and Campaigning. The same rationale could be applied to the Fairs Pipeline. However, oftentimes HUUB still goes through the Profiling and Campaign Stages for brands that are part of a given fair, as a preliminary step that occurs prior to the event. Thus, it still makes sense to include these two stages in the Fairs Pipeline.

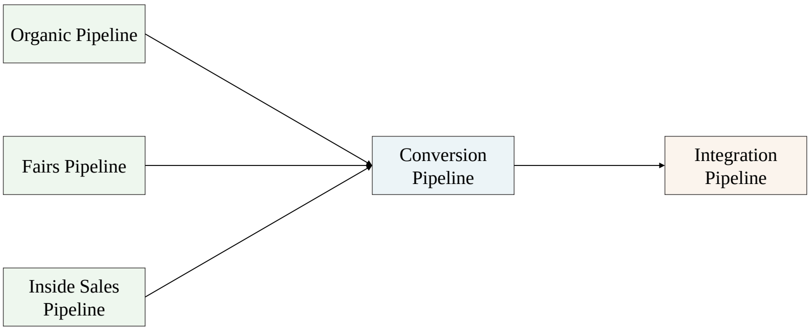


Figure 3.3: Division into Organic, Fairs and Inside Sales Pipelines

3.2.1.2 Conversion Pipeline

After going through the LGQ Pipeline, a brand enters the Conversion Pipeline. This pipeline contains seven different stages and, similarly to the LGQ Pipeline, four stages refer to main processes (namely Qualified, Demo, Proposal and Negotiation) and the remaining three act as a result of the Negotiation stage - Lost, Postponed and Won (Figure 3.4).

The first stage of this pipeline is the Qualified stage, where deals that come from the LGQ Pipeline stay until they advance to the Demo stage. This Qualified stage is, in reality, intertwined with the Qualified Stage presented in the previous pipeline.

Demo Stage

At this stage, HUUB goes into further detail about its value proposition, and dives deeper into its internal and external structure, explaining how the multiple teams operate and providing more information about the company's stakeholders. SPOKE, HUUB's online logistic platform, as the only tangible product ready to present at such an early stage, is also demoed to the customer.

Pre-Proposal and Proposal Stages

The Pre-Proposal stage occurs when HUUB's Sales Team is preparing the business proposal, that is tailored to each potential customer. After outlining all the details of the proposal and sending it to the customer, it moves on to the Proposal stage, and remains there until an answer is given.

Negotiation Stage

The Negotiation stage is the last one before the brand is considered acquired. After the brand responds to the offer presented in the Proposal Stage, both parties exchange multiple contacts so they can come to terms and decide whether or not the process should advance. From these discussions, similarly to the Qualifying stage described in section 3.2.1.1, there are three possible outcomes: the brand drops out of the process, thus going to the Lost Stage; it postpones its decision (Postponed Stage), or it advances to the Integration process (Won Stage). If a brand decides to go through with the settlement, it advances to the Integration Pipeline, where it is integrated in HUUB's ecosystem.

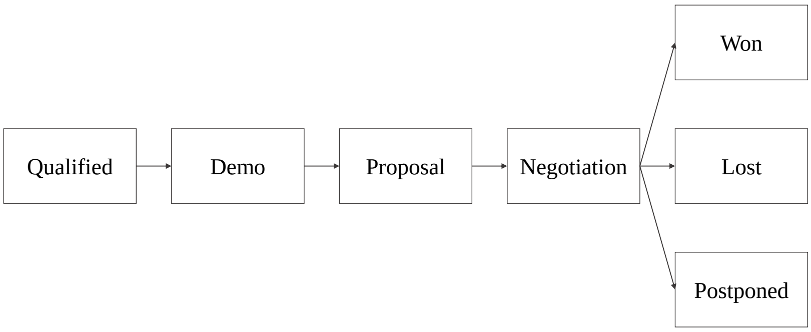


Figure 3.4: Conversion Pipeline stages

3.3 The problem

Traditionally, all the leads generated are fed to its assigned pipeline and move across the sales funnel without any prioritization. This methodology works well when the number of leads to manage is small. However, as this number rises, some differentiation must be made between the different leads so the company can focus on the ones that have more chances of being converted further down the pipelines. HUUB currently relies on human knowledge and experience to make this differentiation. This method is not optimal, since it introduces unnecessary variance and bias and is also not efficient. Figure 3.5 shows that the proportion of leads in the Lead Generation and Qualification Pipelines vastly exceeds the number of leads in the Conversion or Integration Pipelines, which indicates that the method in use can be improved.

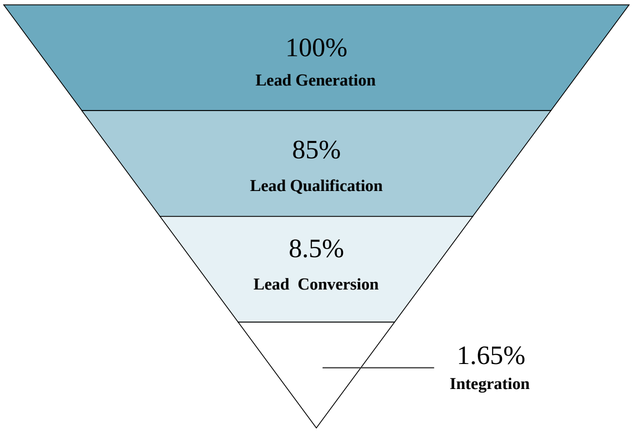


Figure 3.5: HUUB sales funnel

An alternative strategy is to attribute a lead score to each potential customer. This score aims to rank leads by assigning them a numerical score that translates its proneness to move on the stages further down the pipeline. Leads with higher lead scores are prioritized while leads with lower values are dealt with later in time. Lead scores are computed considering the information gathered in the Profiling stage and comparing it with a set of characteristics predefined by the company.

The goal of this project is to build a model that helps the Sales Team to be more selective when choosing which leads to pursue, in order to increase the ratio between the number of leads converted to the number of leads generated. This will be achieved by adding an extra stage between the Prolifing and Campaign stages, implemented "under the hood", thus not impacting the current pipeline structure (Figure 3.6). In this stage, data collected in the Profiling stage will be fed to a machine learning algorithm that will output a lead score result. The Sales Team can then use the results of the model to guide their decision-making process. In order to train the model, historical data containing all the leads' information was provided by HUUB.

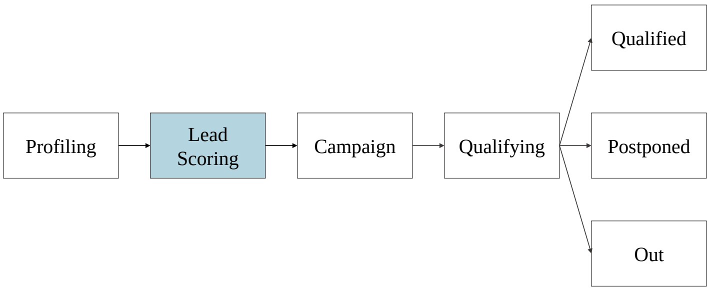


Figure 3.6: Updated Lead Generation and Qualification Pipeline Stages

3.3.1 Computing the expected lead revenue

Besides calculating the lead score, HUUB is also interested in knowing in advance the expected revenue that will be generated from a given lead, once it is seen as an important feature to be included in the lead scoring model. This is often difficult to estimate in the Profiling stage, and most of the times can only be accurately determined once a lead reaches the Conversion Pipeline. However, there is a formula used by the Sales Team that allows the calculation of the expected revenue when the number of items sold in a given season and the distribution of the sales channel (percentage of ecommerce vs percentage of wholesale businesses) are known variables.

$$\text{Revenue} = N \times P \times ((\frac{N_{ecommerce}}{N} \times f_{ecommerce}) + (\frac{N_{wholesale}}{N} \times f_{wholesale})) \tag{3.1}$$

Where:

N , is the number of items sold per season
$N_{ecommerce}$, is the number of items sold through e-commerce
$N_{wholesale}$, is the number of items sold through wholesale
$f_{ecommerce}$, is a factor pre-determined by the Sales Team, equal to 3, and
$f_{wholesale}$, is a factor pre-determined by the Sales Team, equal to 2

Although it is possible to estimate the revenue based on this formula, oftentimes the Sales Team does not have access to all the information necessary to do so (namely N, $N_{ecommerce}$ or $N_{wholesale}$). Having an accurate revenue estimate not only helps to better evaluate the quality of a lead, but also acts as a crucial input for the predictive lead scoring model.

3.3.2 Calculating the lead score

With the rise of machine learning techniques, lead scores can be estimated taking into account the data gathered by HUUB in the Profiling Stage, and factoring in the outcomes of past deals (the stage in which they stopped progressing). This method is called predictive lead scoring.

Since HUUB does not have a lead scoring methodology implemented in the company, the deals depicted in the provided dataset do not have an associated lead score. Once the model needs this score as a target variable to train on, the first step will be to calculate the initial lead scores, based on the data provided. After computing these lead scores, the predictive lead score model will be developed.

Chapter 4

Methodology

In this chapter, the methodology used to approach the lead score prediction problem is defined. After an initial description of the dataset, the preprocessing procedures applied to the data, namely feature transformation, data encoding and missing value imputation, are described. A new tool capable of handling different imputation methods and a selection of tree-based ensemble methods is then presented. This tool will be used not only to predict the lead score of a potential customer, but also the expected revenue it may bring for HUUB.

4.1 Data Preprocessing

4.1.1 Dataset characterization

The dataset provided by the company contains information about all the deals that occurred since may 2016. Before diving deeper into the data preprocessing procedures, it is crucial to understand the dataset in use. The dataset contains 5343 instances and 14 features. Out of these features, 1 is numeric, 12 are categorical, and 1 has both types of data, and will be referred to as an *hybrid* feature.

The dataset provided encompasses the following features:

Revenue Revenue that may be generated by HUUB as the result of partnering up with a given fashion brand. This value is calculated using Equation 3.1.

Country of Origin, Country of Production and Main Markets The Country of Origin refers to the country where the brand comes from, while the Country of Production is related to the country where its products are manufactured. Main markets is another location-related variable and has information about the target markets of a certain brand. Table 4.1 shows some of the values presented in the raw data that concern these features.

Table 4.1: Possible values for features *Country Origin*, *Country Production* and *Main Markets*

country_origin
United Kingdom/USA
france
ALEMANHA
Portugal
Suiça
USA GEORGIA

country_production
California
Vietnam, China
PT/world
Europe(France, Portugal)
Bali
UK, USA, CN, JP, KR, CA

main_markets
Iberian Peninsula; Europe
Finland; Europe
USA&Canada;Portugal; Iberian Peninsula
Italy
Asia
Middle East

Brand type Type of clothing the brand produces in regards to its customer age group (in case the brand sells clothing items) or according to the type of products sold. There are four possible categories the values may fit in: Men, Women, Kids, Home and Other. The Home category is present since HUUB works with brands that commercialize homewear such as towels and bed linen. Table 4.2 shows some examples of raw data for this feature.

Table 4.2: Possible values for feature *Brand Type*

brand_type
Men; Women
Kids; Men; Women
Home

Product Category States the product category of the products sold by the brand. It can be either Apparel, Footwear, Underwear, Homewear, Accessories or Other.

Brand Segment HUUB segments brands according to the pricing of its products. A brand can be included in one of the following categories: low cost, value, premium or luxury products. The majority of brands HUUB works with belongs to the premium segment (Figure A.1 of Appendix A).

Instagram followers Number of Instagram followers of a brand. This is the only feature in the dataset that can be represented categorically, namely as a range (e.g. 5k-20k), or expressed in its numeric format, as the exact number of followers (e.g. 12 500).

Sales channels and E-commerce platform Sales channels a brand works with. In case it works with B2B clients it is said to be in the Wholesale business. Besides wholesale, brands may also work with B2C channels, namely through e-commerce platforms. The name of these platforms is specified in the *E-commerce platform* feature (the name of some of these platforms is shown in Table 4.3).

Table 4.3: Possible values for feature *E-commerce platform*

ecommerce_platform
Shopify
Magento
WooCommerce
Prestashop
(...)

Pipespike Pipespike is an external company that searches and collects data regarding potential interesting leads, thus being a potential source of information in the Profiling stage. This boolean feature states whether a given deal was sourced from Pipespike or not.

Source pipeline, Current pipeline and Stage Section 3.2.1 discusses the multiple pipelines involved in a lead's journey, as well as all the stages it goes through (denoted in the *Stage* feature). Features *Source Pipeline* and *Current Pipeline*, as the names suggest, indicate in which pipeline a lead came from and where its progression ended, respectively.

Existing Business Boolean feature that indicates if a brand is new and launching in the market for the first time, or if it is an already established brand.

An important distinction to make regarding the categorical features presented has to do with whether an instance may be assigned to multiple classes. These type of features are called *multi-label* features while the ones that take solely one class are called *single-label* features. For instance, the feature *Country-Production* may take "Portugal, Spain and Bangladesh" as values, thus being considered a *multi-label* feature; feature *E-commerce Platform*, on the other hand, can only take one class (i.e. "Shopify" or "Magento"), hence the *single-label* designation. Table 4.4 indicates which features are categorical and numeric, as well as the feature subset they belong to.

Table 4.4: Distinction between feature type and subset

Type	**Subset**	**Feature**
Numeric	Continuous	Revenue
Categorical	Single-label	E-commerce Platform, Source Pipeline, Current Pipeline, Stage
	Binary	Existing Business, Pipespike
	Multi-label	Country of production, Country of Origin, Main Markets, Brand Type, Product Category, Sales Channels
Hybrid		Instagram Followers

4.1.2 Feature transformation

A large portion of time in any machine learning project is spent preprocessing data. One of the data preprocessing steps is feature transformation. In simple terms, feature transformation can be defined as a function that transforms features from one representation to another.

All features in the dataset required some preprocessing, some more laborious than others. This treatment included analysing typographical errors and different types of separators for values in *multi-label* features (e.g. comma, semi-colons, forward slash). Two of the most complex features to analyze were *Country Origin* and *Country Production*, since the countries presented in the dataset contained not only typographical errors but also different ways of referring to the same country (i.e. USA/United States/US/United States of America/Washington). The complete list of problems detected in these 2 features is shown in Appendix B.

Features *Instagram followers* and *E-commerce platform* required a different type of feature transformation, hence the need of presenting a separate analysis.

Instagram followers

This feature contained both categorical and numeric data. Categorical data was represented as an interval, with five distinct bins: <5k, 5k-20k, 20k-50k, 50k-100k and >100k; numeric values depicted the exact number of followers of a given brand. This presented an issue since a feature cannot have mixed types. The analysis of Figure 4.1 shows the proportion of values represented as intervals is similar to the proportion of values depicted as numerical values. Hence, since there is no prevalence of one type of variable, it is not possible to easily decide whether to transform the numeric values into categorical or vice-versa. The transformation from categorical to numeric could be made by fetching the exact number of Instagram followers from a deal's Instagram page. However, such process is not possible since one is interested in knowing the number of followers at the time the deal occurred and not in the present moment. Hence, transforming numeric values into categorical ones is, in this case, the most reasonable procedure, and allows information to be represented in the most accurate manner.

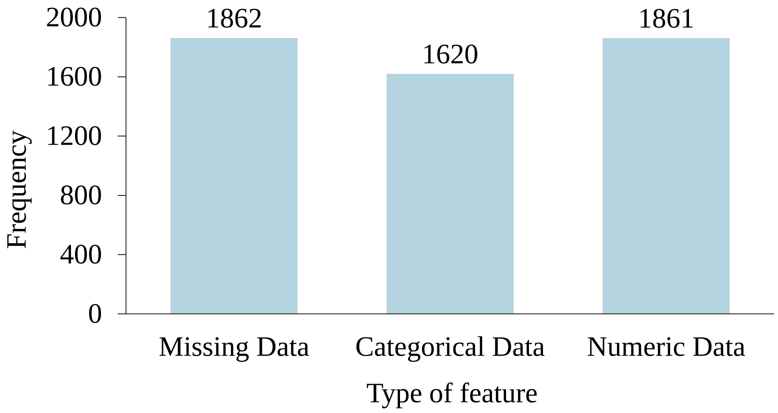


Figure 4.1: Instagram followers distribution per type of feature

E-commerce Platform

This feature suffered a transformation since it contained non-relevant information. Currently, HUUB works with two different e-commerce platforms: WooCommerce and Shopify. Hence, making the distinction between every e-commerce platforms is not as helpful as simply stating whether the brand works with the same E-commerce platforms HUUB does. Consequently, the feature *E-commerce platform* was converted into a Boolean feature (1, if the brand works with the same E-commerce platform as HUUB; 0, if contrary).

4.1.3 Data Encoding

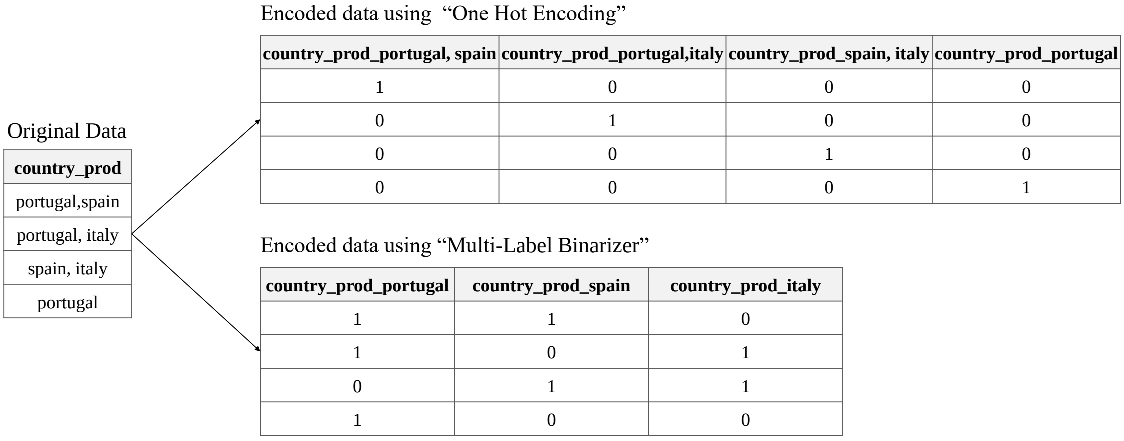
Categorical data are usually more challenging to work with with than numerical data. In particular, many machine learning algorithms require their input to be numerical and therefore categorical features must be transformed before being used with these algorithms.

There are two types of categorical variables, *nominal* and *ordinal*. An ordinal variable has an intrinsic ordering to its categories whereas a nominal variable does not. In this dataset, features *Brand Segment* and *Instagram followers* were the only two that presented an intrinsic order in its categories, and were then encoded using Ordinal Encoding. This procedure involves converting each string value to a whole number, while taking into consideration the meaning of each value attributed. Ordinal Encoding for feature *Brand Segment* is demonstrated in Table 4.5. Features *Existing business*, *E-commerce Platform* and *Pipespike* did not require any encoding, since they are Boolean and can be interpreted as 1 (True) or 0 (False).

Table 4.5: Ordinal encoding for feature Brand Segment

Original Value	Encoded Value
Low Cost	1
Value	2
Premium	3
Luxury	4

The remaining variables were nominal and multi-label which narrowed the scope of possible encoders that may be used. In such cases, One Hot Encoding is usually the strategy adopted. This encoding system creates new binary columns that indicate the presence of each possible value in the original data. However, this method may lead to a quick "explosion" in the number of columns, especially in high cardinality features. This is critical in features like *Country Production* and *Country Origin*, given the amount of possible combinations involving countries. The encoding system used was similar to One Hot Encoding but instead of considering the possible combinations of values, it transformed each individual label (category) into a column (Figure 4.2). Hence, for each instance, it is possible to have more than one column with a positive value, which leads to a decrease in the overall number of columns created. Due to the nature of this encoding methodology, it is often referred to as *Multi-label Binarizer*.



Original Data

country_prod
portugal,spain
portugal, italy
spain, italy
portugal

Encoded data using "One Hot Encoding"

country_prod_portugal, spain	country_prod_portugal,italy	country_prod_spain, italy	country_prod_portugal
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

Encoded data using "Multi-Label Binarizer"

country_prod_portugal	country_prod_spain	country_prod_italy
1	1	0
1	0	1
0	1	1
1	0	0

Figure 4.2: Difference between One Hot Encoding and Multi-label Binarizer

4.1.4 Handling Missing Data

One of the main characteristics of this dataset is the large amount of missing values. From all the values of the dataset, 30% are missing, and only 200 out of the 5343 instances (3.7%) contained no missing values (a visualization of the dataset and its missing values is provided in Appendix C). Figure 4.3 shows the missing data distribution per feature.

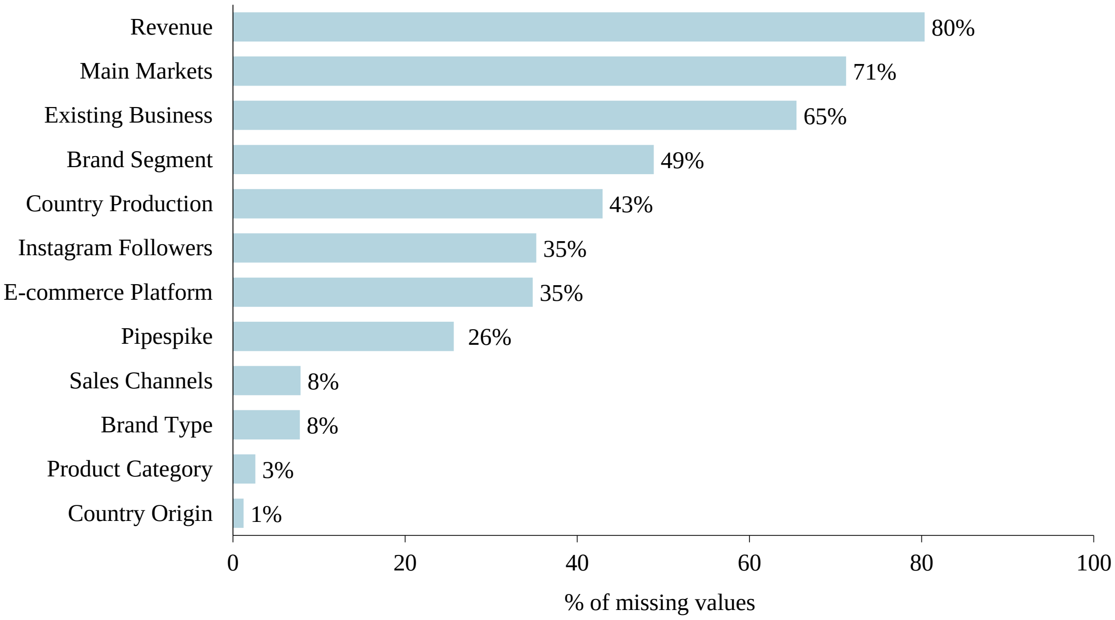


Figure 4.3: Percentage of missing values per feature

Although there were features with a significant percentage of missing values, no feature was dropped from the analysis since it was believed that they could have an impact on the final model, even if a large portion of it was obtained via imputation techniques.

Among the most popular imputation techniques are k-NN and MICE, vastly used due to their ability to handle both continuous and categorical features. When using the k-NN imputation methodology, the most important parameter to tune is the value of k. However, it is not feasible to test multiple datasets imputed with different values of k in the final model, since it is computationally expensive. To tackle this issue, a methodology that allows the estimation of the imputation accuracy for different k values was developed, so the best value could be found in a more efficient manner. This methodology can is generic, and can be used with datasets with both features types (numeric and categorical).

4.1.4.1 Calculating k-NN imputation accuracy

The methodology used to determine the accuracy of the k-NN imputation method comprised the following steps:

1. The dataset was divided into a training and a test set (80/20 split).
2. The test set was duplicated in order to create a synthetic test set.
3. For each instance in the synthetic test set, one feature without missing values was set to missing. This feature was randomly selected, but conditioned with weights based on the frequency of missing values of the features, so that a feature with more missing values is more likely to be picked. This ensures the selected feature is one that has a higher rate of missing values, thus making the accuracy calculation more meaningful.
4. The synthetic test set was merged with the training set so the model could train with the entire dataset.
5. The synthetic test set, with the imputed (predicted) missing values, was detached from the training set.
6. The synthetic test set was compared with the real test set and the accuracy was computed.

When calculating the accuracy, the model compares the test and predicted values (equation 4.1). However, since the number of features of the multiple types (single-label, multi-label and binary) may be differ, the accuracy should be weighted to account for these differences. Consequently, the equation used to compute the accuracy was weighted according to the number of features of each type (Equation 4.2).

$$\text{Accuracy (ACC)} = \frac{\text{Number of correctly classified instances}}{\text{Number of instances classified}} \tag{4.1}$$

$$\text{Weighted ACC} = ACC_{ML} \times \frac{N_{ML}}{N} + ACC_{SL} \times \frac{N_{SL}}{N} + ACC_{binary} \times \frac{N_{binary}}{N} \tag{4.2}$$

Where:

N , is the total number of features
N_{ML} , is the number of features that contain multi-label (ML) values
N_{SL} , is the number of features that contain single-label (SL) values, and
N_{binary}, is the number of features that contain binary values

To evaluate the performance of the imputation method for numeric features, a separate analysis had to be conducted, since it is not possible to use classification metrics for numeric variables. The metric used to evaluate the performance of the imputation in numeric features was the Mean Absolute Percentage Error (MAPE), given by the following expression:

$$\text{MAPE} = \left(\frac{1}{n}\right) \sum_{i=1}^{n} \left| \frac{y_i - x_i}{y_i} \right| \times 100\% \tag{4.3}$$

Where:

y_i, is the actual value
x_i, is the predicted value, and
n , is the number of fitted points

In case the dataset encompasses both categorical and numeric variables, one must choose what metric should prevail in the decision for the best k value. The rationale applied had into consideration the number of categorical features when compared to the number of numeric ones. When the proportion of categorical features exceeds the proportion of numeric features, the weighted accuracy is used to make the comparison between models; when the opposite occurs, this comparison is made based on MAPE. For this specific dataset, since the number of categorical features vastly exceeds the number of numeric ones (1 out of 14), this comparison was made based on the weighted accuracy.

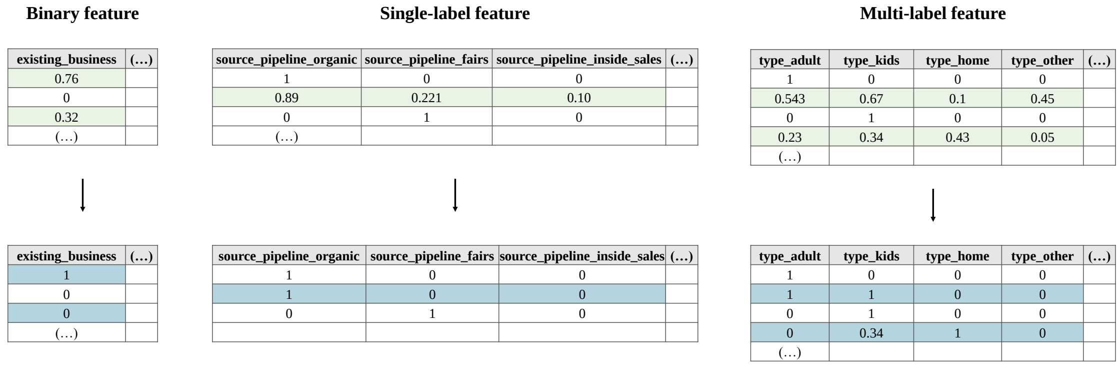
4.1.4.2 Transforming the imputation results

One should note that both imputation techniques (*k*-NN and MICE), since they worked with data in its encoded state, generate decimal values between 0 and 1. Since the variables are encoded using a Multi-Label binary encoder, the output of the imputation should also be binary (0 or 1). Thus, a transformation following the imputation process was applied in order to convert the values from decimal to integer. This transformation process required setting a threshold above which values would be converted to 1. The chosen threshold was 0.5, since it is is the middle point between 0 and 1.

This transformation can occur in three different ways, depending on the type of variable:

- **Binary**: values above the threshold were set to 1.
- **Single-label**: the maximum value was set to 1.
- **Multi-label**: all the values above the threshold were set to 1; in case there were no values above the threshold, the maximum value was set to 1.

An example of each type of transformation is shown in Figure 4.4.



Binary feature

existing_business	(…)
0.76	
0	
0.32	
(…)	

existing_business	(…)
1	
0	
0	
(…)	

Single-label feature

source_pipeline_organic	source_pipeline_fairs	source_pipeline_inside_sales	(…)
1	0	0	
0.89	0.221	0.10	
0	1	0	
(…)			

source_pipeline_organic	source_pipeline_fairs	source_pipeline_inside_sales	(…)
1	0	0	
1	0	0	
0	1	0	

Multi-label feature

type_adult	type_kids	type_home	type_other	(…)
1	0	0	0	
0.543	0.67	0.1	0.45	
0	1	0	0	
0.23	0.34	0.43	0.05	
(…)				

type_adult	type_kids	type_home	type_other	(…)
1	0	0	0	
1	1	0	0	
0	1	0	0	
0	0.34	1	0	
(…)				

Figure 4.4: Transformation from decimal to integer values

4.2 Lead Scoring Prediction

4.2.1 Computing the Lead Score

In order to train the lead scoring model, a target feature is necessary so the machine learning algorithm can learn. Since scoring leads is a process that was not yet implemented (neither manually nor automatically), the initial lead score had to be computed based exclusively on the data available. Features *Current Pipeline* and *Stage*, indicate in which step of the customer acquisition process a lead stopped progressing, hence presenting the information needed to make a simple lead score. The ultimate goal of a lead score is to translate into a number how promising a given lead is. The association between how far a lead gets in the customer acquisition process with how promising it is can be made since the more time a lead spends in the process, engaging with HUUB Sales Team, the more interesting it becomes.

The first step in creating the lead score consisted in reducing the number of stages in the process. Figure 4.5 shows a diagram of the whole lead conversion process, as well as the percentage of deals that stopped progressing in each stage. Since any given deal must fall under one of these stages, the sum of all frequencies is equal to 100%. Stages "Qualified", "Demo", "Pre-proposal", "Proposal" and "Negotiation" presented low frequency values, and are not very differentiated from a business view-point. Consequently, these five stages were merged, resulting in one macro stage called solely "Negotiation". All the other stages were considered to be relevant distinct stages, and were translated into different lead scores.

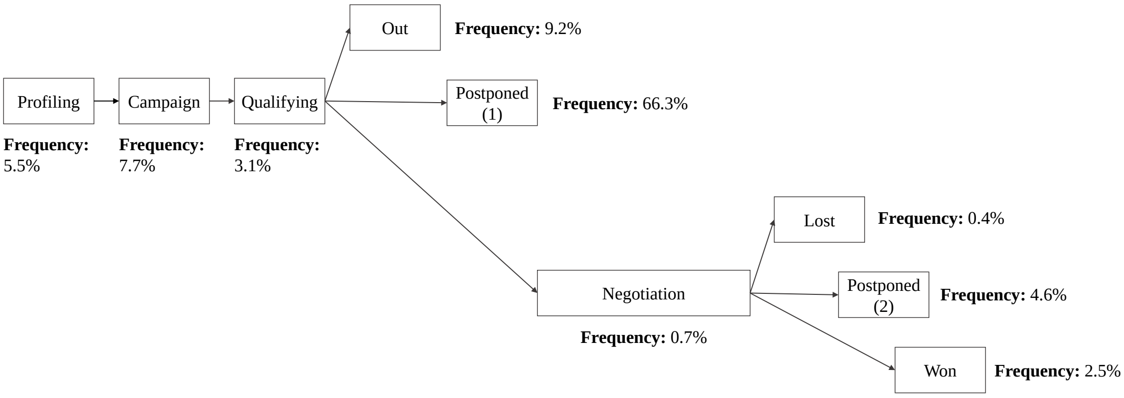


Figure 4.5: Lead conversion process and frequency of leads in each stage

In the end of this process, there were 9 stages to consider. For each stage, a value was attributed, based not only on the order it appears in the lead conversion process but also in its business importance (Figure 4.6). It must be noted that since features *Current Pipeline* and *Stage* were used to compute the lead score, they were not included in the lead scoring predicting model.

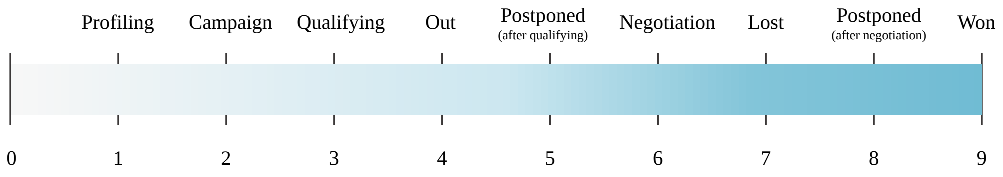


Figure 4.6: Lead Score Scale

Figure 4.7 shows the distribution of the lead score, after applying the rationale aforementioned to every deal in the dataset.

The reason why the lead scoring prediction was treated as a regression problem had to do with the need to preserve the order between the values of the lead score. If the problem were to be treated as classification, the model would not be able to perceive the differences between lead scores, since it would treat all values as separate categories. For instance, in a classification problem, a lead score of 8 has the same intrinsic meaning as a lead score of 9, whereas in a regression problem a lead score of 8 is recognized to be closer to 9 than to 3.

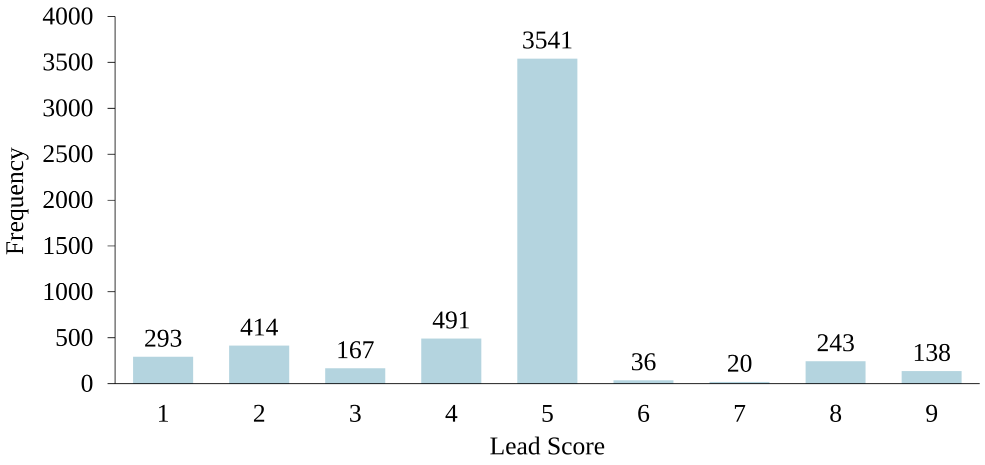


Figure 4.7: Lead Score Distribution

4.2.2 Regression Models with Imputation Tool (RMIT)

The ultimate goal of this project is to predict the lead score of a potential customer. However, once the revenue feature is the one with the highest missing data rate and given its importance for the lead scoring model, one other goal is to get more accurate predictions for this feature. Given this twofold objective, an automated machine learning tool that tests different combinations of imputation methods and machine learning algorithms was developed, meant to be used with regression problems. The main advantage of this tool is that it can be used to predict different target variables, hence allowing the prediction of the revenue as well as the lead score. Given the purpose of the tool, it will henceforward be designated as Regression Models with Imputation Tool (RMIT) (Figure 4.8).

RMIT requires data to be already preprocessed and encoded. The dependent variable must also be specified in the beginning, so it knows what to predict. The missing values in the dataset are then filled using *k*-NN, MICE and Mean/Mode imputations, thus generating three different datasets. Following the Missing Data Imputation Step is Model Selection. In this step, the algorithms chosen to make the predictions were based on two different ensemble methods: bagging and boosting. These ensemble methods, as discussed in sections 2.3.3.1 and 2.3.3.2, include Random Forest, Adaptive Boosting and Gradient Boosting algorithms, which are among the most powerful and popular algorithms used in the machine learning field, recognized for their success on improving the prediction accuracy over single models. In addition to these algorithms, *k*-NN was also considered, so it could act as the single-model baseline comparison against the bagging and boosting algorithms. In the end of the training period, the best performing combination of imputation method with regression model is selected, which is then used to make the predictions.

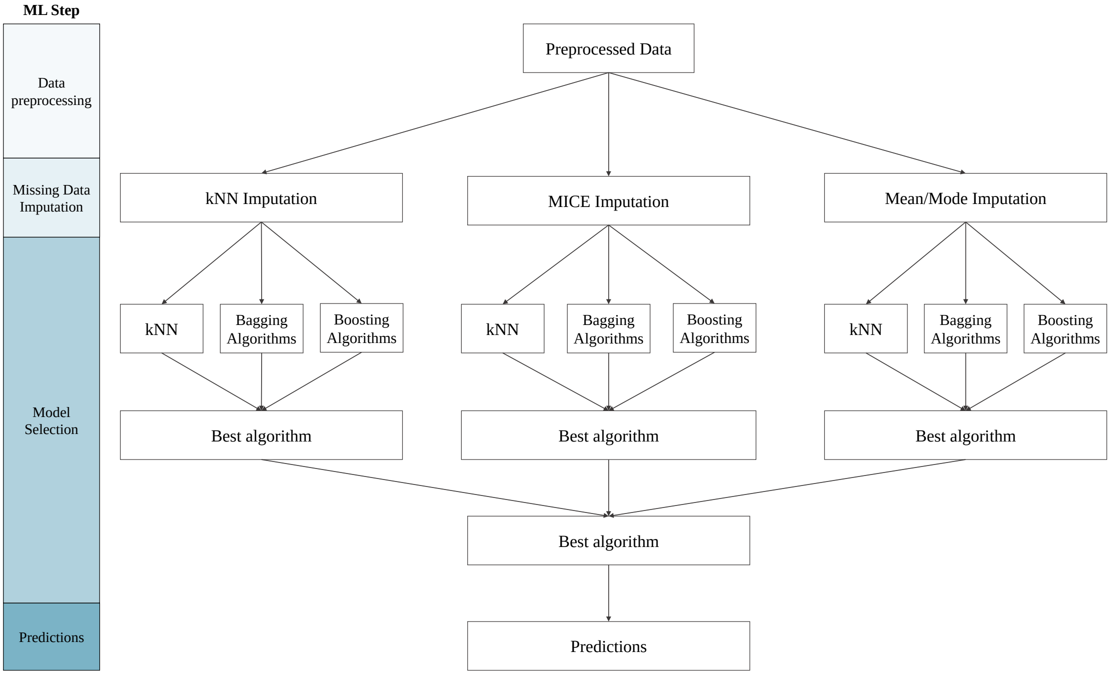


Figure 4.8: Regression Models with Imputation Tool (RMIT) framework

4.2.3 Scenario Definition

In order to assess the impact of using the revenue results predicted by RMIT on the lead score, one must have a term of comparison. Hence, two different scenarios were designed (Figure 4.9).

- **Scenario A**: the revenue is not predicted using RMIT and is fed to the lead scoring model as a feature with missing values. Thus, this feature is treated like any other feature, and its missing values are imputed using *k*-NN, MICE and Mean imputation.
- **Scenario B**: the revenue is predicted using RMIT and then fed into the lead scoring model.

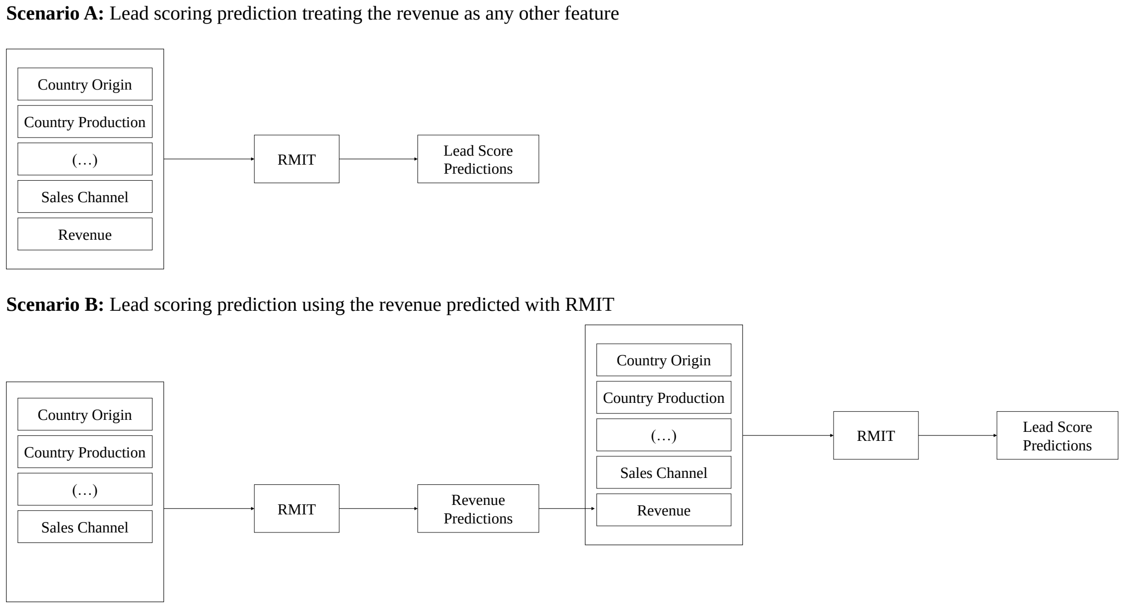


Figure 4.9: Schematic of the Scenario design

4.2.3.1 Revenue prediction using RMIT

The difference between both scenarios is that scenario B uses the RMIT to make the revenue prediction prior to the lead score prediction. In this step, the features considered to train the model were the ones outlined in section 4.1.1, with the exception of *Current Pipeline* and *Stage* since they reflect the final result of the lead conversion process.

In any machine learning problem, it is important to analyse the feature in which one wants to gain a deeper understanding. In this case, the target feature is the revenue. Since HUUB works with a wide variety of brands with various sizes, their revenues are also very distinct. In the dataset provided, the revenue varies from 3000€ to 3.5M€, with an average value of 54 530€ and a median of 22 450€. The analysis of the distribution of revenue, presented in Figure 4.10, shows that the revenue distribution is severely right-skewed. However, although infrequent, extreme values are a possibility and should be accounted for in the model.

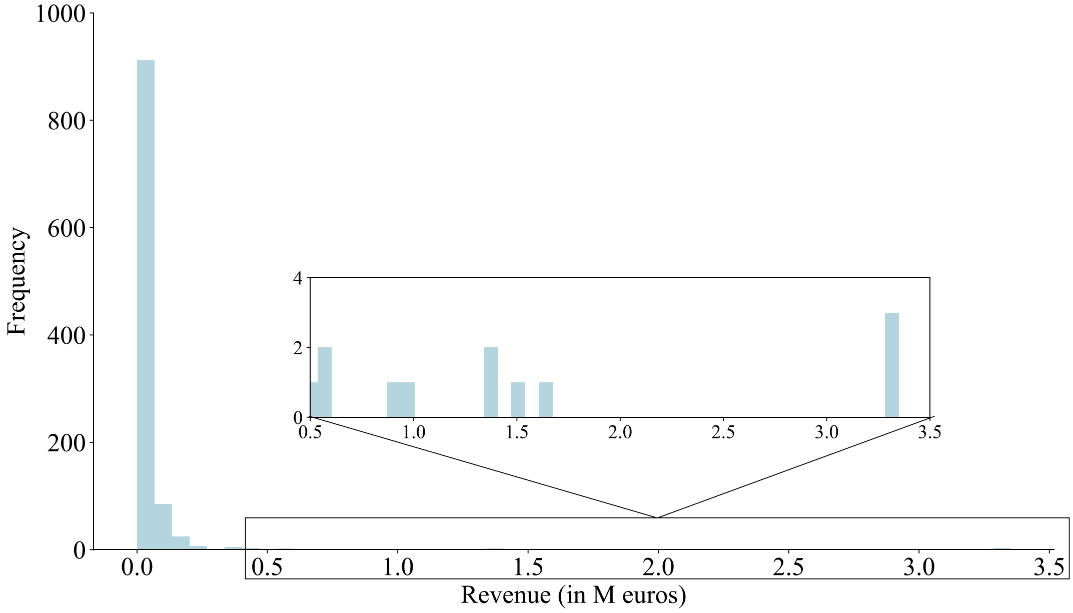


Figure 4.10: Distribution of the target feature

4.2.4 Model Evaluation and Parameter tuning

The evaluation procedure is essential to ensure that the models depicted in RMIT are reliable. The most common method is to split the data into a training and a test set, train the model using the training set and assessing its performance on the test set. This method is not very reliable as the accuracy obtained for one test set can be very different to the accuracy obtained for a different test set.

Hence, the method chosen to evaluate model performance was *k*-fold cross validation. Cross validation is often used since it is easy to understand, easy to implement, and results in estimates that have lower biases than other methods (Brownlee, 2018). This procedure randomly splits the data into *k* groups, then using one of the groups as the test set and the remaining (*k*-1) as the training set. This process is repeated until each unique group as been used as the test set. In this project, the value of *k* was set to 10.

Since both problems addressed, the revenue prediction and the lead score prediction, are regression problems, the metrics used to assess the performance of the model were the Mean Absolute Error (MAE) and the Root Mean Squared Error (RMSE). The RMSE was used to evaluate the results of the revenue prediction model, since the revenue distribution is right-skewed. Given that this metric squares errors before they are averaged, it gives more weight to larger errors. This is specially important in the business context, once larger errors (i.e. failing to predict a brand that may bring a larger revenue) are more relevant than smaller ones. To assess the performance of the lead scoring model, the MAE was used.

These metrics are computed using the following expressions, where y_i is the actual value, x_i is the predicted value, and n the number of instances:

$$RMSE = \sqrt{\left(\frac{1}{n}\right)\sum_{i=1}^{n}(y_i - x_i)^2} \tag{4.4}$$

$$MAE = \left(\frac{1}{n}\right)\sum_{i=1}^{n}|y_i - x_i| \tag{4.5}$$

Hyperparameter tuning

In order to apply these algorithms correctly, their hyperparameters must be tuned. At the beginning of each run, an $n \times h$ matrix of n combinations of h hyperparameters was generated, so the algorithm could test multiple settings. As for the hyperparameters used in each model, Boosting algorithms were allowed to vary its number of trees, maximum depth and learning rate. For the Random Forest algorithm, the number of trees and the maximum depth were the hyperparameters varied. As for the k-NN algorithm, the k value was tuned. A brief description of these hyperparameteres is provided in Table 4.6.

Table 4.6: Hyper-parameter description

Hyperparameter	**Description**
k	Number of nearest neighbours
Number of trees	Number of trees built to produce the ensemble
Maximum depth	Maximum allowed tree size
Learning rate	Measure of how fast the error is corrected from each tree to the next

Chapter 5

Results

In this Chapter, the results obtained from applying the algorithms and methodology defined in the previous Chapter are shown. To that extent, two distinct scenarios are defined in order to assess the impact of using the RMIT to calculate the revenue prior to the lead scoring prediction.

5.1 Programming Tools

To develop this model, the programming language used was Python. Python stands out as the main programming language used in the machine learning field due to its simple syntax and readability. Moreover, it has a vast library of ready-to-go packages, allowing developers to concentrate their efforts on problem-solving and achieving project goals, rather than in coding issues.

As for the packages used in the imputation phase, *fancyimpute* was applied since it comprised all the techniques used in the model (kNN, MICE and Mean/Mode Imputation). In the prediction stage, three different packages were used: *sklearn* for Random Forest and Adaptive Boosting (AdaBoost) implementations and XGBoost and CatBoost for Gradient Boosting. XGBoost stands for eXtreme Gradient Boosting and is currently the most powerful gradient boosting algorithm available. CatBoost is another gradient boosting algorithm whose main advantage, besides being faster than XGBoost, is its ability to handle categorical features out-of-the box. The way CatBoost is able to work with categorical features is based on a special type of encoding, which takes into account the target variable (target-encoding). In order to work with CatBoost, after the imputation procedures, features had to be transformed back to the original format (without encoding).

All experiments were computed using an Intel® Core™ i7 2,4GHz processor with 8GB of RAM.

5.2 Scenario Analysis

In this section, the results of the scenarios defined on section 4.2.3 are presented. To keep the length of this chapter under reasonable limits, all the optimum hyperparameter values used in the machine learning algorithms in both scenarios are shown in Appendix D, as well as the computational times needed to run these algorithms.

5.2.1 Scenario A: Lead scoring prediction treating the revenue as any other feature

In this scenario, the revenue feature is treated like a missing value. The first step of the RMIT, specified in section is 4.2.2, is the imputation of missing values. When applying the *k*-NN imputation method, the most important parameter to tune is the value of *k*. The comparison between the accuracy of imputation for different *k* values was done using the methodology described in section 4.1.4.1. Since the dataset contains both categorical and numeric features with missing values, the imputation procedure outputs two different metrics: the weighted accuracy and the MAPE. Since this dataset contains far more categorical feature than numeric ones, although the MAPE results are generated as an output of the imputation process, as detailed in the methodology, the weighted accuracy analysis should prevail over the MAPE. The value of *k* that yielded the best weighted accuracy was 36, as shown in Figure 5.1.

The three imputation methods described in the methodology (*k*-NN, MICE and Mean/Mode) were applied to the dataset, generating three distinct datasets.

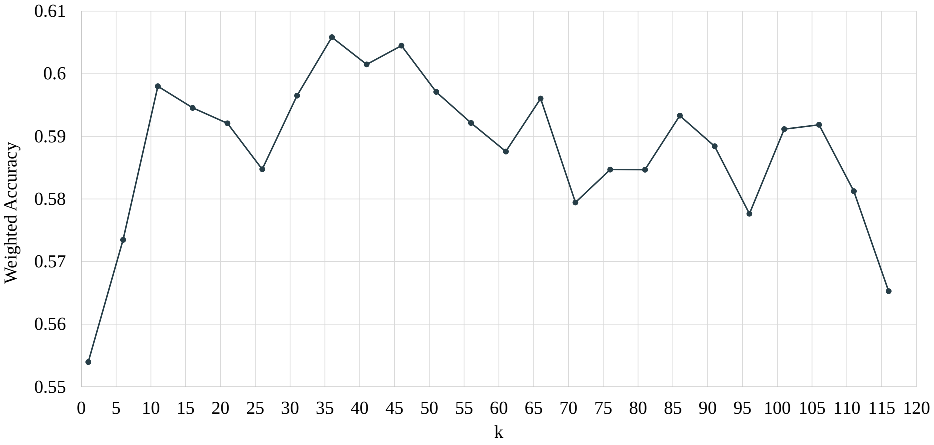


Figure 5.1: Weighted accuracy variation with *k*

The three different datasets were then used as input for the lead scoring model. Figure 5.2 shows the results of all the combinations of these datasets with the regression models used in the RMIT.

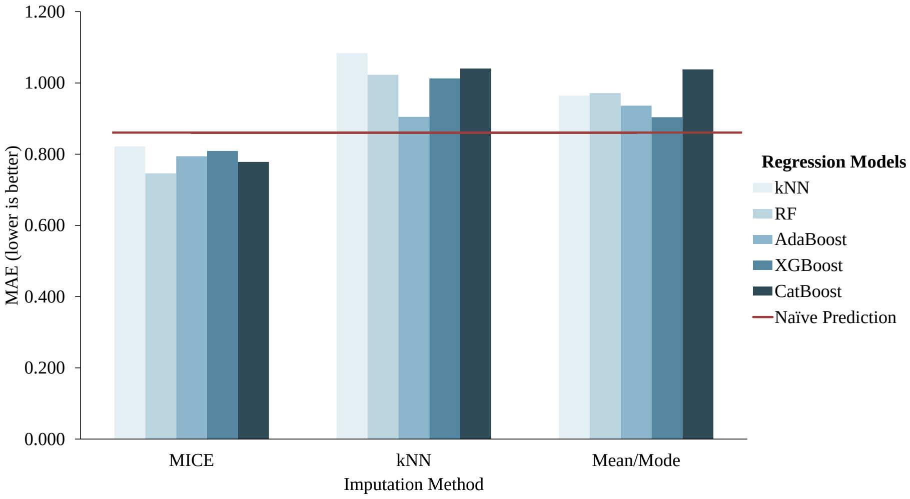


Figure 5.2: MAE results for Scenario A for the different combinations of imputation methods and regression models

Figure 5.2 shows that the imputation method that generates the lowest values of MAE is MICE. This imputation technique, followed by the implementation of the Random Forest algorithm, yields the best results, with a MAE of 0.746. This means that when predicting a potential customer lead score, the model fails, on average, by 0.746. If one were to naïvely predict the lead score, by taking into consideration its most frequent value (lead score = 5, as shown in Figure 4.7), the MAE of the prediction would be 0.860. Hence, this configuration results in an improvement of 15.28% when compared to the naïve prediction.

5.2.2 Scenario B: Lead scoring prediction using the revenue predicted with RMIT

This scenario comprises two different steps. In the first step, RMIT is used to predict the revenue of a given deal. These predictions are then used as input for the lead scoring model.

Predicting the revenue using RMIT

In this stage, the model trains with only 20% of the dataset, which corresponds to the fraction that has no missing values in the revenue feature. There is the need to create three distinct datasets, one for each imputation method. For the *k*-NN imputation, the best value of *k*, according to the weighted accuracy, was 21 (Figure E.1 in Appendix E shows the variation of the weighted accuracy with *k*).

In this case, the evaluation metric used to assess the performance of the algorithms was not the MAE but the RMSE. The need to use a different evaluation metric is related to the skewness of the target variable. The range of revenues varies from 3000€ to 3.5M€, and from a business stand-point, if the model fails by a large amount, it should be penalized. If the model underpredicts the revenue of a large deal, it may result in HUUB not pursuing that lead, hence resulting in the loss of a big potential client, which is undesirable. In the same way, overpredicting the revenue of a given deal is also negative, since it results in the pursuit of leads that are seen as more profitable than they really are. By using the RMSE, since the error is squared (unlike the MAE), larger differences are further emphasized.

The results of using the three imputation methods combined with the algorithms selected to make the revenue prediction are shown in Figure 5.3.

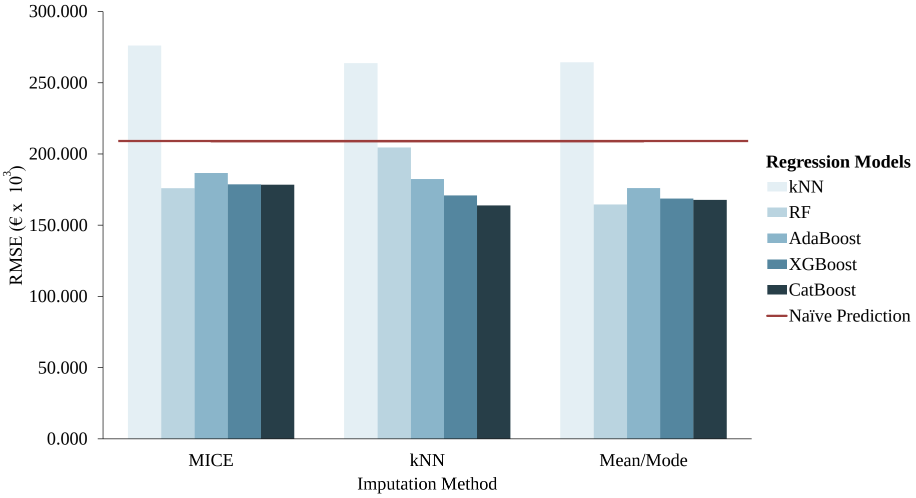


Figure 5.3: Revenue prediction RMSE for the different combinations of imputation methods and regression models

In order to assess if the results obtained are acceptable, one should establish a baseline comparison value for the RMSE. A naïve prediction assumes that every deal yields the same revenue. In this case, the mean revenue (54 530€) was used as the predicted value for every deals, which results in a RMSE of 208 934€.

From the analysis of the RMSE shown in Figure 5.3 it can be concluded that *k*-NN is the model that performs the worst, being worse than the naïve prediction for every imputation technique. Moreover, the application of both bagging and boosting algorithms shows better results than the naïve prediction. Out of these regression models, the one that yields the best results is Gradient Boosting (CatBoost), when preceded by a *k*-NN imputation technique, yielding a RMSE of 163 859€. When compared to the naïve prediction, this represents an improvement of 27.5%. Given the revenue distribution presented in Figure 4.10, it was expected that the error values would be high, since the range of revenues is large. Since deals that generate extremely large amounts of revenue are scarce in the dataset provided (only 2.7% yield revenues above 200k €), the model does not have sufficient information to make accurate estimations for these deals, which hinders its performance.

Lead Score Prediction

Since the revenue feature has no missing values (as they were predicted in the previous stage), the missing value imputation step was applied only to the categorical features. For the k-NN imputation, Figure 5.4 shows the weighted accuracy variation with k (the maximum weighted accuracy occurs when $k = 61$) .

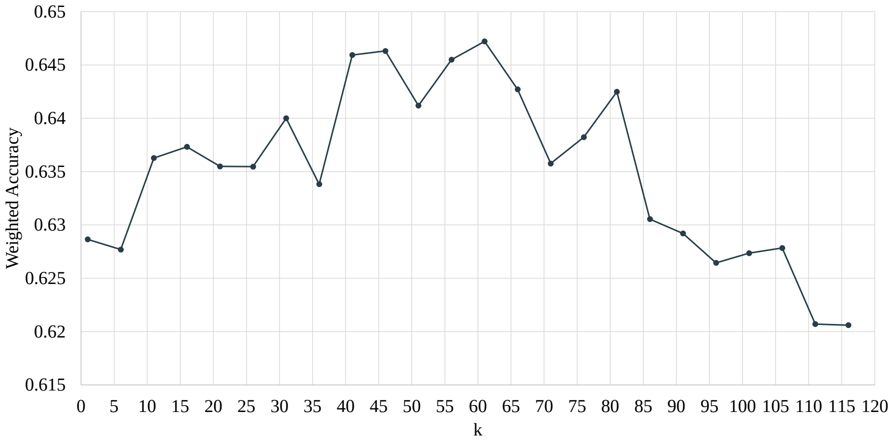


Figure 5.4: Weighted accuracy variation with k

The combination of the three imputation techniques with the bagging and boosting models, produces the results presented in Figure 5.5.

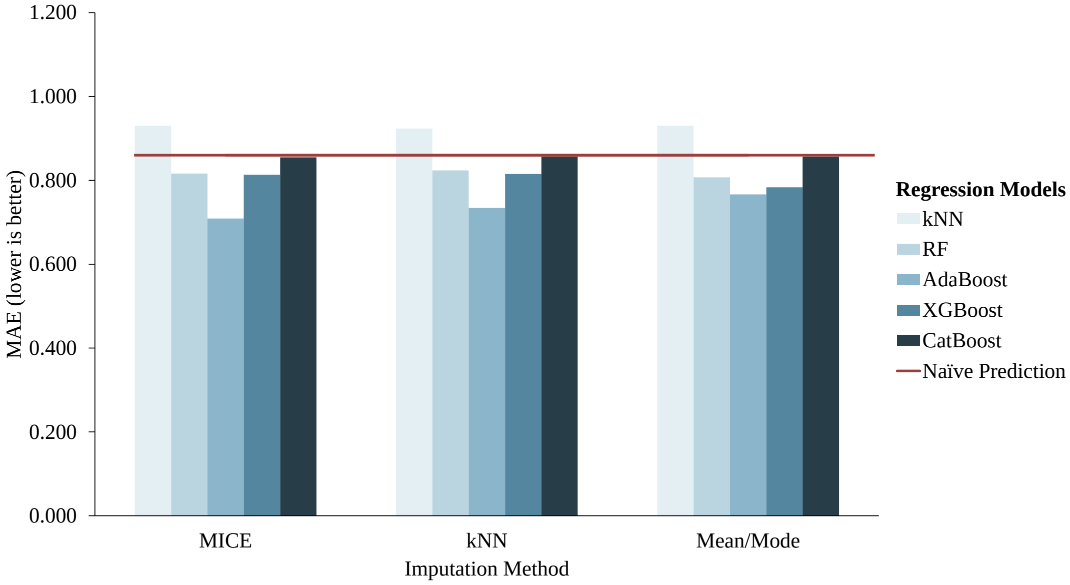


Figure 5.5: MAE results for Scenario B for the different combinations of imputation methods and regression models

In this scenario, using a combination of MICE imputation method with an Adaptive Boosting algorithm produces the smallest MAE (0.708). When compared to scenario A, this represents a 5.4% improvement, meaning that it is preferable over simply treating the revenue as a missing value (scenario A). Comparing this result to the naïve prediction, mentioned in section 5.2.1, an improvement of 21.47% is obtained.

Chapter 6

Conclusions and Future Work

The developed work focused on the analysis of the customer acquisition process at HUUB. This process is handled by the Sales Team, whose role is to research and contact fashion brands that may be interested in taking advantage of HUUB's services. The main problem with the current approach is that the decision of which brands to contact relies heavily on empirical knowledge, often resulting in the pursuit of leads that end up not bearing fruit. The primary goal of this project was to suggest a new method to help the Sales Team decide which brands have the most potential to become part of HUUB's ecosystem. The chosen methodology was based on the creation of a lead score, a numeric value attributed to each lead that translates its proneness to being converted into a client. This lead scoring system was built taking into account historical data, and relied on using multiple machine learning techniques to get a model that can predict the lead score of future potential clients.

The project started by framing the problem at hand, namely characterizing the current customer acquisition process, with all its different stages. The dataset provided had information about the stage in the negotiation process where a certain deal stopped progressing. Once there was no lead scoring implementation being used at HUUB prior to this project, the proposed lead score scale results from the transformation of the final stage of a given deal into a numeric value, on a scale from 1 to 9.

After having a clear understanding of the problem, a more in-depth analysis of the features contained in the dataset was carried out. The main particularity of the data was that out of the 14 features it contained, only 1 was numeric. When it comes to preprocessing procedures, one of the biggest challenges, besides having to clean and transform unstandardized features, was handling the large amount of missing values. Consequently, the first part of the project was spent trying to find solutions to tackle this issue, namely through the use of imputation techniques such as *k*-NN, MICE and Mean/Mode. Throughout this process, a novel methodology to evaluate the accuracy of imputation in the *k*-NN algorithm was developed, to allow tuning the *k* parameter in a more effective manner.

Once the data preprocessing step finished, the following step was to test multiple machine learning models to make the lead scoring prediction. This project focused on using tree-based ensemble methods (bagging and boosting) due to their ability to combine several models to produce the best results. To that extent, Random Forest, Adaptive Boosting and Gradient Boosting were tested in the context of the problem, as well as the kNN algorithm, which acted as a single model baseline comparison. In order to take advantage of the best imputation methods together with the best regression algorithms, a generalized automated machine learning tool capable of combining both steps was developed - Regression Models with Imputation Tool (RMIT).

After the preprocessing step, the RMIT was used to make the lead score predictions. One important feature of the dataset and a valuable input to the lead scoring model is the revenue a given deal brings to HUUB. This feature, besides being the only numeric one, was also the one that presented the highest missing value rate. Thus, it was believed that using the RMIT to fill these missing values prior to the lead scoring prediction phase would result in more accurate lead scores. Two scenarios were designed to test this premise: scenario A, where the lead score was predicted using RMIT and revenue was treated as a normal variable, and scenario B where the revenue was predicted using RMIT, prior to the lead scoring prediction stage. The results of this analysis showed that the lead scoring model that took the revenue computed using RMIT (scenario B) produced the best lead scoring results, showing a 5.4% improvement in MAE when compared to scenario A. When compared to the naive prediction (predicting the same lead score for every deal, based on the most frequent value), 21.47% improvement was achieved in the MAE.

The presented methodology brought up multiple advantages. Firstly, this project allowed HUUB to gain a deeper understanding on data that was not being used. The data cleaning process carried out was automated, hence removing the need to manually transform data to a suitable format when future deals are introduced in the database. Secondly, the missing data problematic was thoroughly analyzed. This issue is inherent to the problem at hand, since the amount of information HUUB can gather in regards to a certain deal depends on the responsiveness of the brand, and falls out of the Sales Team responsibility. However, it still deserved a careful analysis, since such a high rate of missing values (30%) influences the outcome of the machine learning algorithms tested. Finally, the main contribution of this project was the development of a novel lead scoring methodology, that will help the Sales Team make more informed decisions when contacting potential customers. Moreover, the lead scoring problem and the dataset particularities (namely the high rate of missing data on the revenue feature) incited the development of a new tool (RMIT) that the company can from now on use to approach other regression problems.

Despite the advantages brought up by the presented methodology, there is still some work to be done so this project can yield maximum benefits. The next step is to deploy the lead scoring model. In order to do so, an interface has to be developed to link the lead scoring model with the third-party software the Sales Team currently works with and where the information collected about new potential customers is stored. After this linkage is established, a brand's lead score will be automatically computed right after the introduction of the characteristics of a given deal in the system.

Another improvement that should be made is related to the quality of the data gathered. The dataset provided required a lot of effort to be put on preprocessing procedures, in particular in the data cleaning stage. Hence, a special attention should be put on the format of the data that is introduced in the Sales Team platform, to avoid it having a myriad of representations for the same value. More specifically, the introduced data should follow the same format data is currently represented, after the data cleaning procedures. In addition, more effort should be put into gathering more features when tracing the profile of fashion brands since new information can help the algorithms to better understand the relationship between the characteristics of a brand and its lead score.

The implementation of this lead scoring methodology is expected to help HUUB's Sales Team prioritize which leads to pursue and improve the ratio of leads converted to client. However, it is important to stress out that this tool should be seen as an enhancer of human judgement, not as a replacement. The ultimate goal of acquiring new customers does not change; what changes are the tools used to achieve that goal.

Bibliography

Amor, N. B., Benferhat, S., and Elouedi, Z. (2006). Qualitative classification with possibilistic decision trees. In Bouchon-Meunier, B., Coletti, G., and Yager, R. R., editors, *Modern Information Processing*, pages 159 – 169. Elsevier Science, Amsterdam.

Antonio, V. (2018). How ai is changing sales. *Harvard Business Review*.

Arnold, T., Fang, E., and Palmatier, R. (2011). The effects of customer acquisition and retention orientations on a firm's radical and incremental innovation performance. *Journal of the Academy of Marketing Science*, 39:234–251.

Azur, M., Stuart, E., Frangakis, C., and Leaf, P. (2011). Multiple imputation by chained equations: What is it and how does it work? *International journal of methods in psychiatric research*, 20:40–9.

Brown, S. and Myles, A. (2009). 3.17 - decision tree modeling in classification. In Brown, S. D., Tauler, R., and Walczak, B., editors, *Comprehensive Chemometrics*, pages 541 – 569. Elsevier, Oxford.

Brownlee, J. (2016). *XGBoost With Python: Gradient Boosted Trees with XGBoost and scikit-learn*. Machine Learning Mastery.

Brownlee, J. (2018). *Statistical Methods for Machine Learning*.

Castelli, M., Vanneschi, L., and Álvaro Rubio Largo (2019). Supervised learning: Classification. In Ranganathan, S., Gribskov, M., Nakai, K., and Schönbach, C., editors, *Encyclopedia of Bioinformatics and Computational Biology*, pages 342 – 349. Academic Press, Oxford.

Cheliotis, M., Gkerekos, C., Lazakis, I., and Theotokatos, G. (2019). A novel data condition and performance hybrid imputation method for energy efficient operations of marine systems. *Ocean Engineering*, 188:106220.

Dawer, G., Guo, Y., and Barbu, A. (2017). Generating compact tree ensembles via annealing.

Duda, R. O. and Hart, P. E. (1973). Pattern classification and scene analysis john wiley and sons.

Duncan, B. and Elkan, C. (2015). Probabilistic modeling of a sales funnel to prioritize leads. pages 1751–1758.

Fratello, M. and Tagliaferri, R. (2019). Decision trees and random forests. In Ranganathan, S., Gribskov, M., Nakai, K., and Schönbach, C., editors, *Encyclopedia of Bioinformatics and Computational Biology*, pages 374 – 383. Academic Press, Oxford.

Friedman, J. (2000). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer series in statistics. Springer.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2014). *An Introduction to Statistical Learning: With Applications in R*. Springer Publishing Company, Incorporated.

Jonsson, P. and Wohlin, C. (2004). *An evaluation of k-nearest neighbour imputation using Likert data*.

Kapil, A. (2018). Methods of missing value treatment and their effect on the accuracy of classification models.

Kim, Y., Street, N., Russell, G., and Menczer, F. (2005). Customer targeting: A neural network approach guided by genetic algorithms. *Management Science*, 51:264–276.

Kingsford, C. and Salzberg, S. (2008). What are decision trees? *Nature biotechnology*, 26:1011–3.

Lantz, B. (2013). *Machine Learning with R*. Packt Publishing.

Madley-Dowd, P., Hughes, R., Tilling, K., and Heron, J. (2019). The proportion of missing data should not be used to guide decisions on multiple imputation. *Journal of Clinical Epidemiology*, 110:63 – 73.

Mayr, A., Binder, H., Gefeller, O., and Schmid, M. (2014). The evolution of boosting algorithms from machine learning to statistical modelling. *Methods of information in medicine*, 53.

Monard, M.-C. (2002). A study of k-nearest neighbour as an imputation method.

Nygard, R. J. (2019). Automating lead scoring with machine learning: An experimental study.

Patil, D., V.M, W., and J.A, G. (2010). Evaluation of decision tree pruning algorithms for complexity and classification accuracy. *International Journal of Computer Applications*, 11.

Power, B. (2017). How harley-davidson used artificial intelligence to increase new york sales leads by 2,930%. *Harvard Business Review*.

Raghunathan, T. E., Solenberger, P., and Hoewyk, J. V. (2000). Iveware: Imputation and variance estimation software: Installation instructions and user guide.

Rokach, L. and Maimon, O. (2014). *Data Mining With Decision Trees: Theory and Applications*. World Scientific Publishing Co., Inc., USA, 2nd edition.

Royston, P. and White, I. (2011). Multiple imputation by chained equation (mice): Implementation in stata. *Journal of Statistical Software*, 45.

Song, Y. Y. and Lu, Y. (2015). Decision tree methods: applications for classification and prediction. *Shanghai Arch Psychiatry*, 27(2):130–135.

Stevens, R. P. (2011). *Maximizing Lead Generation: The Complete Guide for B2B Marketers*. Que Publishing Company, 1st edition.

Strike, K., El-Emam, K., and Madhavji, N. (2001). Software cost estimation with incomplete data. *Software Engineering, IEEE Transactions on*, 27:890–908.

Syam, N. and Sharma, A. (2018). Waiting for a sales renaissance in the fourth industrial revolution: Machine learning and artificial intelligence in sales research and practice. *Industrial Marketing Management*, 69:135 – 146.

Söhnchen, F. and Albers, S. (2010). Pipeline management for the acquisition of industrial projects. *Industrial Marketing Management*, 39(8):1356 – 1364. Building, Implementing, and Managing Brand Equity in Business Markets.

Terho, H., Eggert, A., Haas, A., and Ulaga, W. (2015). How sales strategy translates into performance: The role of salesperson customer orientation and value-based selling. *Industrial Marketing Management*, 45:12 – 21.

Tin Kam Ho (1998). The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844.

Troyanskaya, O., Cantor, M., Sherlock, G., Hastie, T., Tibshirani, R., Botstein, D., and Altman, R. (2001). Missing value estimation methods for dna microarrays. *Bioinformatics*, 17:520–525.

van Buuren, S. (2018). *Flexible Imputation of Missing Data*. Chapman & Hall/CRC Interdisciplinary Statistics. CRC Press, Taylor & Francis Group.

Yang, X.-S. (2019). 6 - data mining techniques. In Yang, X.-S., editor, *Introduction to Algorithms for Data Mining and Machine Learning*, pages 109 – 128. Academic Press.

Zhang, Y. and Haghani, A. (2015). A gradient boosting method to improve travel time prediction. *Transportation Research Part C: Emerging Technologies*, 58:308 – 324. Big Data in Transportation and Traffic Engineering.

Zhang, Z. (2016). Missing data imputation: Focusing on single imputation. *Annals of translational medicine*, 4:9.

Appendix A

Brand Segment Distribution

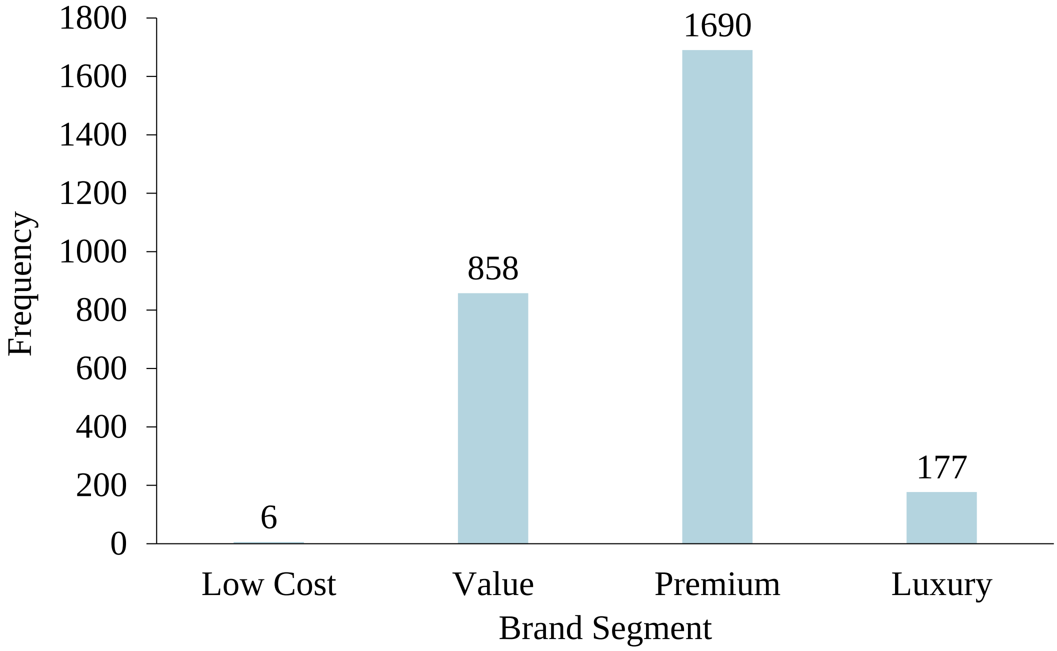
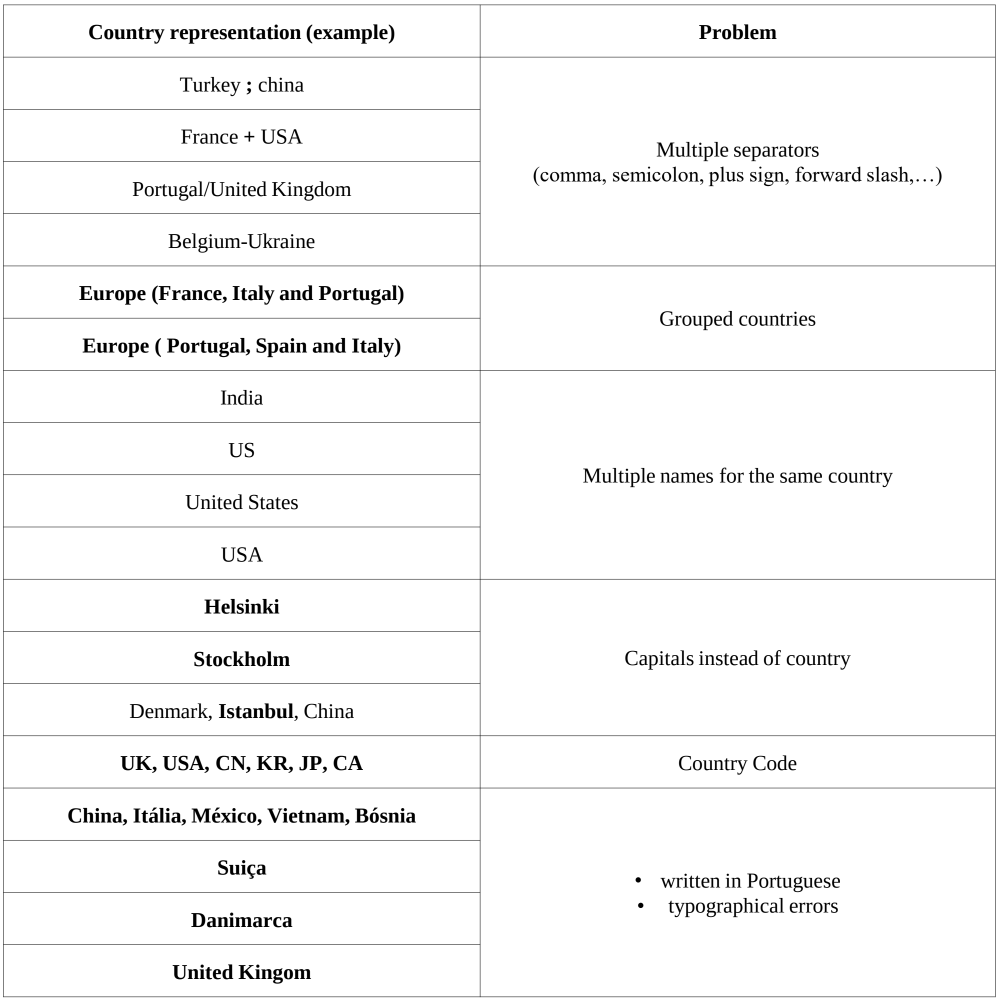


Figure A.1: Brand Segment frequency

Appendix B

Feature transformation



Country representation (example)	**Problem**
Turkey **;** china	Multiple separators (comma, semicolon, plus sign, forward slash,…)
France **+** USA	
Portugal/United Kingdom	
Belgium-Ukraine	
Europe (France, Italy and Portugal)	Grouped countries
Europe (Portugal, Spain and Italy)	
India	Multiple names for the same country
US	
United States	
USA	
Helsinki	Capitals instead of country
Stockholm	
Denmark, **Istanbul**, China	
UK, USA, CN, KR, JP, CA	Country Code
China, Itália, México, Vietnam, Bósnia	• written in Portuguese • typographical errors
Suiça	
Danimarca	
United Kingom	

Figure B.1: Problems detected in features containing country names

Appendix C

Missing Data - Dataset representation

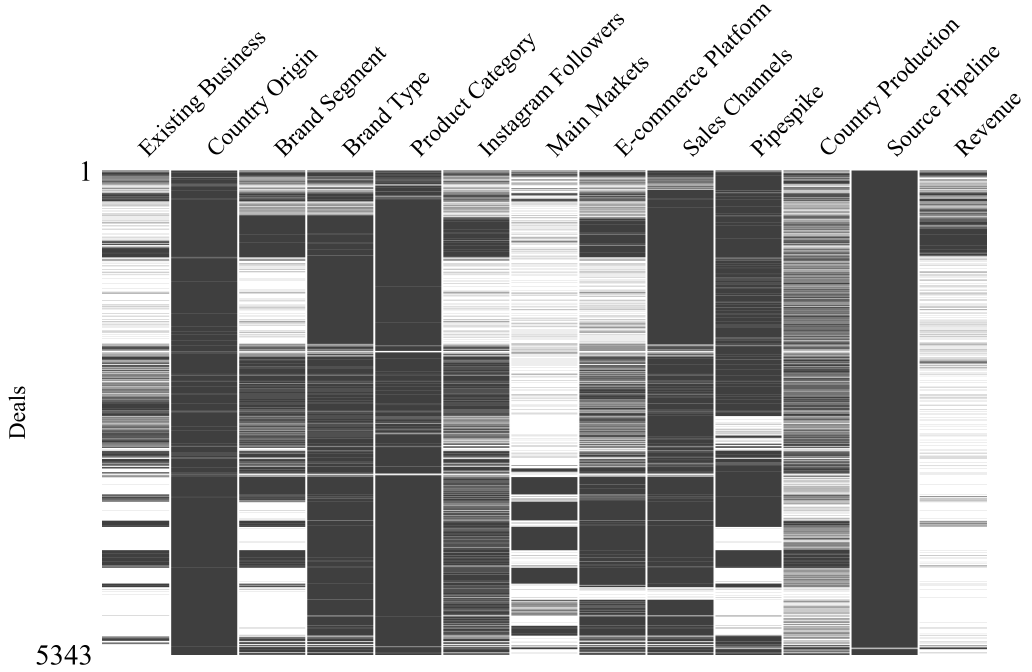


Figure C.1: Visual representation of the entire dataset (missing values are represented in white)

Appendix D

Results

Table D.1: Results for Lead Scoring Model - Scenario A

		Cross Validation Metric			Hyperparameters		Runtime
Imputation method	Algorithm	MAE	k	# trees	Maximum Depth	Learning Rate	(minutes)
MICE	kNN	0.822	12				13.94
	RF	0.746		200	None		26.40
	AdaBoost	0.794		500		0.1	101.49
	XGBoost	0.809		50	5	0.2	76.62
	CatBoost	0.778		1000	6	0.1	19.22
kNN	kNN	1.084	36				6.28
	RF	1.023		200	10		23.61
	AdaBoost	0.905		1000		0.1	122.25
	XGBoost	1.013		50	3	0.2	76.32
	CatBoost	1.041		50	3	0.05	19.22
Mean/Mode	kNN	0.964	11				14.08
	RF	0.972		100	10		20.57
	AdaBoost	0.936		500		0.1	83.11
	XGBoost	0.904		50	5	0.3	75.37
	CatBoost	1.038		50	3	0.05	18.07

Table D.2: Results for the Revenue Prediction Model - Scenario B

		Cross Validation Metric	Hyperparameters				Runtime
Imputation method	Algorithm	RMSE	k	# trees	Maximum Depth	Learning Rate	(minutes)
MICE	kNN	276102.524	59				1.17
	RF	175937.229		100	32		9.09
	AdaBoost	186644.516		1000		1	5.64
	XGBoost	178680.641		50	3	0.1	18.72
	CatBoost	178429.180		30	8	0.05	11.99
kNN	kNN	263739.473	49				1.20
	RF	204509.248		200	None		9.45
	AdaBoost	182362.462		50		1	5.04
	XGBoost	170898.480		50	3	0.2	19.21
	CatBoost	163859.390		50	6	0.1	12.59
Mean/Mode	kNN	264387.201	44				1.18
	RF	164546.045		300	32		9.45
	AdaBoost	176069.983		500		1	4.54
	XGBoost	168674.781		50	3	0.3	19.10
	CatBoost	167784.245		30	8	0.1	12.11

Table D.3: Results for the Lead Scoring Model - Scenario B

		Cross Validation Metric		Hyperparameters			Runtime
Imputation method	Algorithm	MAE	k	# trees	Maximum Depth	Learning Rate	(minutes)
MICE	kNN	0.930	49				6.26
	RF	0.816		100	10		22.63
	AdaBoost	0.709		1000		0.05	123.55
	XGBoost	0.813		50	3	0.2	80.65
	CatBoost	0.855		30	3	0.01	19.51
kNN	kNN	0.923	35				6.27
	RF	0.824		100	10		22.61
	AdaBoost	0.734		50		1	88.91
	XGBoost	0.815		50	3	0.2	81.65
	CatBoost	0.856		30	3	0.01	19.28
Mean/Mode	kNN	0.943	49				6.30
	RF	0.807		100	10		21.55
	AdaBoost	0.767		100		0.3	84.10
	XGBoost	0.783		50	4	0.1	81.14
	CatBoost	0.856		30	3	0.01	17.44

Appendix E

Revenue prediction: *k*-NN Imputation

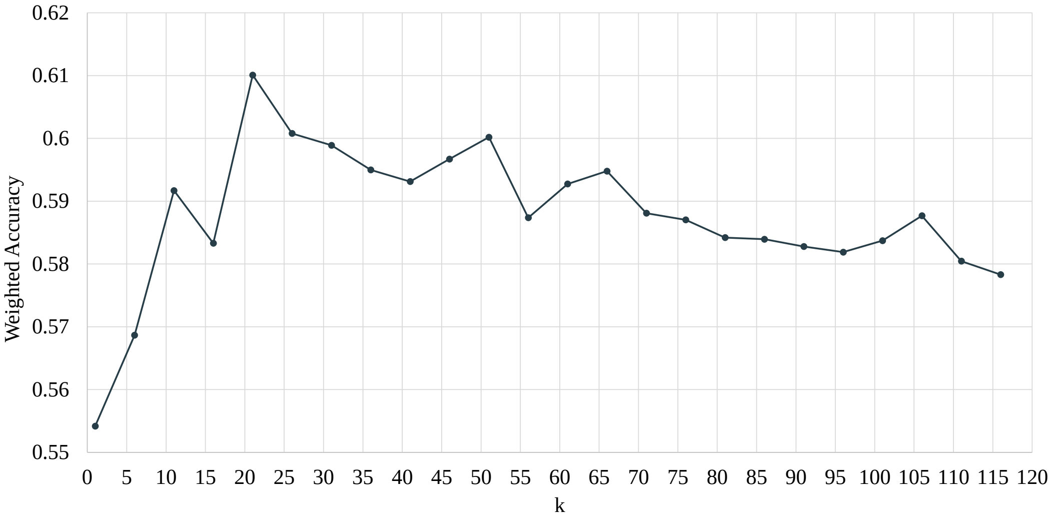


Figure E.1: Weighted accuracy variation with *k*