Fair Pricing in the Telecommunications Sector

By

Ana Beatriz Simões Pereira Querido

Master Thesis in Modelling, Data Analysis
and Decision Support Systems

Supervised by:

Professor João Gama

Nuno Paiva

Faculdade de Economia

Universidade do Porto

2022

Abstract

Pricing in the Telecommunications sector is characterized by a competitive market where aggressive acquisition campaigns happen on a daily basis. Therefore, it is important for Telco companies to provide the best offer possible to their customers. However, this often results in better offers for some customers which are perceived as unfair. Introducing fairness mechanisms in the bundle offer process will settle the balance between company's profit and overall customer satisfaction. Taking this into account and given a large dataset of the customers past offers and current status, this work aims at providing a fairness analysis and determining offers that can be fair for the customer. To reach this goal, the Xtreme Gradient Boost ML algorithm is used after carefully cleaning and filtering the data. Besides the existing company offers, some artificial offers are created and compared using the considered fairness metrics. Results using this approach show that, on average, clients improve their fairness position in 32.4%. Using Artificial offers reflect, on average, a slight improvement (35.3%).

Keywords: Fairness, Fair Pricing, Price Fairness, Fairness Metrics, Machine Learning

Resumo

Os preços dos tarifários no setor de Telecomunicações são caracterizados por um mercado competitivo em que campanhas agressivas de aquisição acontecem diariamente. Portanto, é importante que as empresas forneçam a melhor oferta possível aos seus clientes. No entanto, isso pode resultar em melhores ofertas para alguns clientes, que são consideradas como injustas. A introdução de mecanismos de justiça no processo de oferta de pacotes estabelecerá o equilíbrio entre o lucro da empresa e a satisfação geral do cliente. Tendo isso em consideração e dado um grande conjunto de dados das ofertas anteriores e do estado atual dos clientes, este trabalho visa fornecer uma análise de justiça e determinar as ofertas que podem ser justas para o cliente. Para atingir esse objetivo, o algoritmo Xtreme Gradient Boost ML é usado após limpar e filtrar cuidadosamente os dados. Para além das ofertas existentes na empresa, algumas ofertas artificiais são criadas e comparadas utilizando as métricas de justiça consideradas. Os resultados com esta abordagem mostram que, em média, os clientes melhoram sua posição de justiça em 32,4%. A utilização de ofertas Artificiais refletem, em média, uma ligeira melhoria (35,3%).

Palavras-chave: Justiça, Preço Justo, Machine Learning, Justiça nos preços

Contents

Abstract ii

Resumo iii

1 Introduction **1**

1.1 Context 1

1.2 Motivation 2

1.3 Objectives 3

1.4 Document Outline 3

1.5 Development Tools 4

2 Literature Review **5**

2.1 Fairness 5

2.1.1 Business Relevancy 5

2.1.2 Fairness Formulation 6

2.1.3 Sources of Bias 8

2.1.4 Unfairness Mitigation Methods 10

2.2 Fairness in Pricing 11

2.2.1 Price Fairness Perceptions 11

2.2.2 Algorithmic Pricing 13

2.2.3 Ethicality in Algorithmic Pricing . . . 13
2.2.4 Legal Protection against Algorithmic Pricing . . . 14
2.3 Pricing in Telecommunications . . . 15
2.3.1 Product and Service Bundles . . . 15
2.3.2 Bundling Benefits . . . 16

3 Data Understanding **17**
3.1 Introduction to the Dataset . . . 17
3.2 Dataset Characteristics . . . 17

4 Data Preparation **19**
4.1 Uninformative Variables . . . 19
4.2 Missing Values . . . 21
4.3 Datetime Conversion . . . 22
4.4 Label Encoder and One Hot Encoder . . . 23
4.4.1 Label Encoding . . . 24
4.4.2 One-Hot Encoding . . . 24
4.4.3 Technique Complementarity . . . 24
4.5 Standardization . . . 25
4.6 Creation of ID Variable . . . 25
4.7 Cleaned Dataset . . . 27

5 Fairness Process **28**
5.1 Fairness Position . . . 28
5.1.1 Fairness Criteria . . . 30
5.1.2 Similarity Measurement . . . 30
5.2 Fairness Impact . . . 35

6 Modeling **36**

6.1 Model Selection 37

6.1.1 Pre-evaluation 38

6.1.2 Extreme Gradient Boosting Tree 40

6.2 Model Validation 46

6.2.1 Validation Method 46

6.2.2 Validation Results 49

6.3 Model Evaluation 50

6.4 Work Flow 51

7 Results **55**

7.1 Fairness Analysis 55

7.2 Adhesion Analysis 56

7.3 Bias Analysis 58

8 Final Remarks **61**

8.1 Contributions of this thesis 61

8.2 Future Work 62

A Appendix **63**

A.1 Relevant Catalog Variables Description 63

Bibliography **65**

List of Figures

2.1 Factors that influence a customers perception of fairness (Izaret, 2022) 12

4.1 Number of unique values per variable 20

5.1 Dimensionality Reduction Methods 32

6.1 Evolution of XGBoost from Decision Trees (Morde, 2019) 44

6.2 Growing Window technique 47

6.3 Sliding Window technique 48

6.4 Evaluation of XGBoost algorithm with 1st split of data 49

6.5 Evaluation of XGBoost algorithm with 10th split of data 50

6.6 Work Flow 52

7.1 Histograms of Fairness Impact values 55

7.2 Summary statistics of Fairness Impact values 56

7.3 Adhesion Scores Frequency 57

7.4 Histograms of the offered bundle's price 57

7.5 Pie charts of the type of price difference between current bundle and offered bundle 58

7.6 Pie charts of types of clients regarding their fairness impact 58

7.7 Pie charts of types of clients regarding their responses 59

7.8 Pie charts of types of bundles regarding their clients' fairness impact 59

7.9 Pie charts of types of bundles regarding their clients' response 60

List of Tables

1.1 Software description 4

4.1 Number of catalog/non-catalog variables with and without information 21

4.2 Label Encoding 24

4.3 One-Hot Encoding 25

4.4 Variables used to distinguish packages 26

6.1 Evaluation of 10 splits of the dataset 49

6.2 Client A's source pack and corresponding fair pack 51

6.3 Recommendation pack and corresponding fair pack for client A 6.2 51

6.4 Artificial offer, showing 2 different recommendation packs and their corresponding fair packs for Client A 53

A.1 Variables description 63

Chapter 1

Introduction

1.1 Context

Nowadays, Artificial Intelligence (AI) systems, which take up such a big part in our daily lives, are also frequently used to make really important decisions, in areas such as law, medicine, childhood welfare, employment matching, flight routing, advertising placement, and more (Allen & West, 2018). However, in recent years, there have been multiple real-world cases of decision-making models that have led to discriminatory results: Amazon's recruiting tool that showed bias against women (Dastin, 2018), Staples website that displayed different prices to people depending on distance from a rival brick-and-mortar store (Valentino-DeVries, Singer-Vine, & Soltani, 2012) and an algorithm used by courts to assist in bail decisions that was biased against black defendants (Angwin, Larson, Mattu, & Kirchner, 2016).

This has raised serious Fairness concerns within the research community, with a great number of publications attempting to design a framework that guides practitioners through the process of defining a Fairness measure and assessing its impact when properly implemented. Indeed, Fairness is not a general concept and it is not easy to identify what problems might be present in a Machine Learning (ML) application nor to decide on the appropriate application, which calls for the need to address Fairness concerns on a case by case basis.

One of the ways Fairness can be put into practice is in the context of algorithmic pricing. In this computational pricing strategy, prices are continuously adjusted to a set of varying market conditions and regard customer-specific variables. At first glance, this brings great benefits for the companies, from efficiently managing the inventory and properly reacting to demand and competition, to inducing the highest price each customer is willing to pay and maximizing profit. However, customer experience has become a strategic imperative and personalized pricing is often perceived as generally unfair, which can trigger customer distrust and translate into great losses for the company in the long run. Therefore, pricing policies, if adequate, can maximize the revenue of lifetime value but should also incorporate concerns of Fairness.

This work was developed in the context of a curricular internship done at a telecommunication company based in Portugal, in collaboration with the company's Data Science team. The company currently has a system that recommends new offers for their clients ("Next Best Offer"). This system uses large bulks of data from the client's history, for example, the previous offers they had and what types of services they prefer, and uses a Machine Learning algorithm to determine which offer is best. This algorithm essentially takes into account the trade-off between the probability of the client to accept the offer and the profit margin that the company can have.

1.2 Motivation

Dynamic and personalized pricing strategies have become a generalized practice in the Telecommunications sector. These are really competitive markets where aggressive acquisition campaigns from traditional competitors and new entrants happen on a daily basis, so a bad experience can quickly lead the costumer to switching to the many other options available. In order to differentiate their offers and secure market share, Telecom & Media, have adopted a recommendation system that considers information on a client's past purchases and consumption trends to form an idea about their budget and interests. This allows them to make sophisticated personalized offers with a higher likelihood of acceptation and, consequently, increase profit. Previous works about AI implemented in the Telecom industry to retain client have had successful results (Wu, Yau, Ong, & Chong, 2021) (Ebrah & Elnasir, 2019).

These offers revolve around bundles, that is, packages containing a variety of telecommunication services, such as television, landline, fixed internet, mobile internet and mobile voice cards, which are put together according to what the consumer is likely to value. Because of the different client profiles, the number of ways a bundle can be configured becomes quite unlimited, within the company's policies and/or strategies, and so prices end up being adjusted almost on a case-to-case basis.

Given that the bundle offers to clients are very customizable, this can naturally result in situations perceived as unfair by the customer, if, for example, different clients are offered a bundle with the exact same configuration but at different prices. Or if they receive slightly different offers and one of these offers comes at a lower price but is actually richer in composition, for instance with a bonus or coupon that the client can take advantage of.

> "What is a fair price? That has become a vitally important question in today's era of transparency. Customers have the means and motivation to share and compare prices directly, and companies have the means and motivation to personalize their prices so they are in line with the measurable value they provide to individual customers." - (Izaret, 2022)

With this being said, there is a lot of room for improvement towards creating the balance between customer satisfaction and the company's overall profit.

1.3 Objectives

No matter the relevancy of personalized and dynamic offerings for the telecommunication industry, pricing should still be addressed as a way to create confidence and satisfaction among customers. This said, the objective of this project is to devise an analytical strategy to include "Fairness" in Pricing through product recommendations, conceptualizing a context-specific definition of Fairness, exploring different mitigation techniques and machine learning algorithms, and modelling different scenarios of profit maximization versus Fairness.

To sum up, objectives are:

- Develop a concept for Fairness that aligns with the personalized telecommunication offers;
- Extract meaningful variables from the company's client dataset that provide Fairness criteria
- Develop a ML learning model that can determine the Fairness of a given offer and/or determine the most fair offer
- Understand the trade-off between Fairness and profit maximization

1.4 Document Outline

This dissertation follows the CRISP-DM methodology (Wirth & Hipp, 2000). This framework consists of the following stages: Data Understanding, Data Preparation, Fairness Processes, Modelling, Evaluation and Deployment.

Chapter 2 reviews literature about Fairness, Fairness in Pricing and Pricing in Telecommunications. Chapter 3, Data Understanding, provides a detailed description of the data that has been used for the problem in hand. Chapter 4, Data Preparation, describing the processes of data extraction, description and exploration as well as describing the following pre-processing methods: data cleaning, data transformation and feature selection. Chapter 5, Fairness Process, describes the criteria and metrics that are used for fairness evaluation. Chapter 6 corresponds to the Modelling section, where the Machine Learning algorithms developed are presented as well as the framework used to validate the model predictions. Afterwards, Chapter 7 describes the several experiments performed and the evaluation metrics computed to select the best bundle offer, as well as a fairness comparison with some artificial made offers. To end, Chapter 8 presents the conclusions of this study, the main limitations and some suggestions for future work.

1.5 Development Tools

For this project, and as a intern in the Data Science team, a variety of programs and tools have been used to access and work with data. The following table lists the different software, along with a description of their utility.

Table 1.1: Software description

Tool	Description
DBeaver	Structured Query Language (SQL) editor that establishes the connection to the company's remote database.
mRemoteNG	Remote connections manager used for establishing the first connection to a remote machine that runs any coding scripts.
VSCode	Code editor, where an extension for Python is used. On startup, it automatically connects to the remote machine.
Git	System that controls script versions, tracking any coding changes and allowing to revert to specific versions if necessary. It also provides code review tools, making collaboration with the supervised easier.
Python	Programming language used for coding and working with the different libraries that contain the ML models, data structure and data processing

Along with Python, several libraries were used to process and structure the data.

- **Pandas** - a library that provides fast data analysis and structure
- **Scipy/Numpy** - tools for scientific computing and mathematical operations
- **Matplotlib** - for plotting the data, providing data visualization

The libraries and frameworks mentioned above made it possible to implement the complete pipeline of the data processing used for this thesis, allowing to focus on the problem solving.

Chapter 2

Literature Review

2.1 Fairness

Fairness has been debated among philosophers and psychologists even before its introduction in computer science and yet, there is no universally accepted definition. Different personal outlooks and cultural backgrounds inevitably lend a preference to different ways of looking at it, making it difficult to come up with a single concept that is acceptable to everyone in a situation (Mehrabi, Morstatter, Saxena, Lerman, & Galstyan, 2019). In other words, the concept ultimately boils down to individual perception.

It is only in the context of discrimination that the ideal of Fairness becomes somewhat specific, describing a concept that can be concerned both with withheld important life opportunities or resources and with socially salient qualities used to diminish a particular identity. These qualities refer to what is known as "sensitive data" and include "racial or ethnic origin, political opinions, religious or philosophical beliefs, or trade union membership" and "(…) genetic data, data concerning health, data concerning a natural person's sex life or sexual orientation" (Goodman & Flaxman, 2017). Thus, in very broad terms, unfair discrimination can consist of acts, practices, or policies that impose a relative disadvantage on people based on their membership in a salient social group.

2.1.1 Business Relevancy

The social impact of discriminatory practices and the moral responsibility behind it have led to increasing research on algorithmic Fairness, but there are also good business reasons to consider including Fairness awareness and mitigation Data Science Practice in the industry.

A first reason has to do with the obligation to comply with the European Union's new General Data Protection Regulation (GDPR), which represents EU's normative framework for the

collection, storage, and use of personal information. As summarized in Goodman and Flaxman (2017), the new GDPR introduces two new sets of regulations: the right to nondiscrimination and the right to explanation. Regarding nondiscrimination, Paragraph 71 explicitly requires data controllers to "implement appropriate technical and organizational measures" that "prevents, inter alia, discriminatory effects" on the basis of processing sensitive data. Regarding explanation, Articles 13-15 specify that data subjects have the right to access information collected about them, and that data processors have the obligation to inform data subjects when and why data is collected and processed. In other words, these rights not only place restrictions on automated individual decision making that processes personal data but also greatly strengthen individuals' actual ability to pursue action against companies that fail to comply with the GDPR.

A second reason has to do with customer trust and how it impacts the acceptance and successful adoption of AI-driven services and products. Business executives today want to win customers' trust as growth comes from customers trusting the business, and customers trust businesses which have verifiable, explainable, trustworthy systems. As a result, there has been growing interest in creating frameworks that regulate AI based services or products as a means to ensure their trustworthiness. Some examples include papers like Toreini et al. (2019) and Fjeld, Achten, Hilligoss, Nagy, and Srikumar (2020). The first one relates concepts of trust, drawn from the social sciences literature and trust frameworks, with (classes of) machine learning technologies, aiming to identify trust-enabling machine learning technologies. It also discusses how trust can be enhanced in the various stages of an AI-based system's life space, naming it the Chain of Trust. The second one analyzes the contents of thirty-six "principles" documents in order to identify the foundational requirements for AI that is ethical and respectful of human rights. Comparing the results of both papers, it can be concluded that a common set of principles for AI seems to include "Fairness", "Explainability", "Auditability" and "Safety". Therefore, trustworthy ML practices translate not only into better products for customers but also into a competitive advantage for organizations.

2.1.2 Fairness Formulation

In Machine Learning, Fairness is commonly formalized through mathematical notions that describe some criterion that should be met in order for the algorithm to be considered "fair". However, because there is no universally accepted concept of Fairness and laws for computation are also quite abstract, multiple mathematical formulations have been proposed by the research community in the last years (Verma & Rubin, 2018).

Several papers have attempted to summarize these different algorithmic definitions, but the most complete taxonomy of Fairness seems to be the one presented in Verma and Rubin (2018). For this work, the author analyzed publications in major conferences, journals and reports that dated back to 2012 and was able to distinguish twenty different definitions, grouping them into two main families: "group Fairness" (or "statistical measures") and "individual Fairness" (or "similarity-based measures"). There are other subdivisions, but for simplicity purposes, they will

not be covered here.

At a high level, group notions of Fairness partition data instances into protected and unprotected groups, typically via some sensitive attribute, and then ask that some statistical property be approximately equalized across groups. Some well-known measures include:

- **statistical parity** (or "demographic parity"): it requires that protected and unprotected groups have equal probability of being assigned to the positive class;
- **conditional statistical parity**: it requires that protected and unprotected groups have equal probability of being assigned to the positive class, given a set of legitimate factors
- **equalized odds**: it requires that protected and unprotected groups have equal True Positive Rate (probability of correctly being assigned to the positive class) and equal False Positive Rate (probability of incorrectly being assigned to the positive class);
- **equal opportunity**: it requires that protected and unprotected groups have equal True Positive Rate;
- **treatment equality**: it requires that protected and unprotected groups have an equal ratio of False Negatives and False Positives;
- **test Fairness**: it requires that protected and unprotected groups have equal TPR, for any predicted probability score.

According to the author, the main problem with statistical definitions of Fairness is that they ignore all variables except for the sensitive attribute and so are open to unFairness on structured subgroups that were not explicitly designated as protected. And asking for statistical Fairness across every possible division of the data can quickly lead to overfitting issues. Individual definitions of Fairness attempt to remedy this problem by asking for a constraint that binds at the individual level:

- **Fairness through unawareness**: it requires that no sensitive attributes are explicitly used in the decision-making process;
- **Fairness through awareness**: it requires that any two individuals who are similar with respect to a similarity metric receive similar classification;
- **counterfactual Fairness**: it requires that the output remains the same when the sensitive attribute is flipped to its counterfactual value

Even though these measures do not marginalize over insensitive attributes, they still seem to require strong assumptions about the data. In light of this, recent research (Kearns, Neel, Roth, & Wu, 2018) has focused on a third family of Fairness definitions: "subgroup Fairness",

which represents a middle ground between group Fairness and individual Fairness, asking for a statistical constraint across a large collection of subgroups.

As demonstrated, various mathematical formulations of Fairness have been proposed, each one arguably reasonable to be implemented. However, it has been formally proven that they are often competing and cannot hold at the same time (Chouldechova, 2017; Kleinberg, Mullainathan, & Raghavan, 2016), leading to researchers attempting to quantify the trade-offs between them.

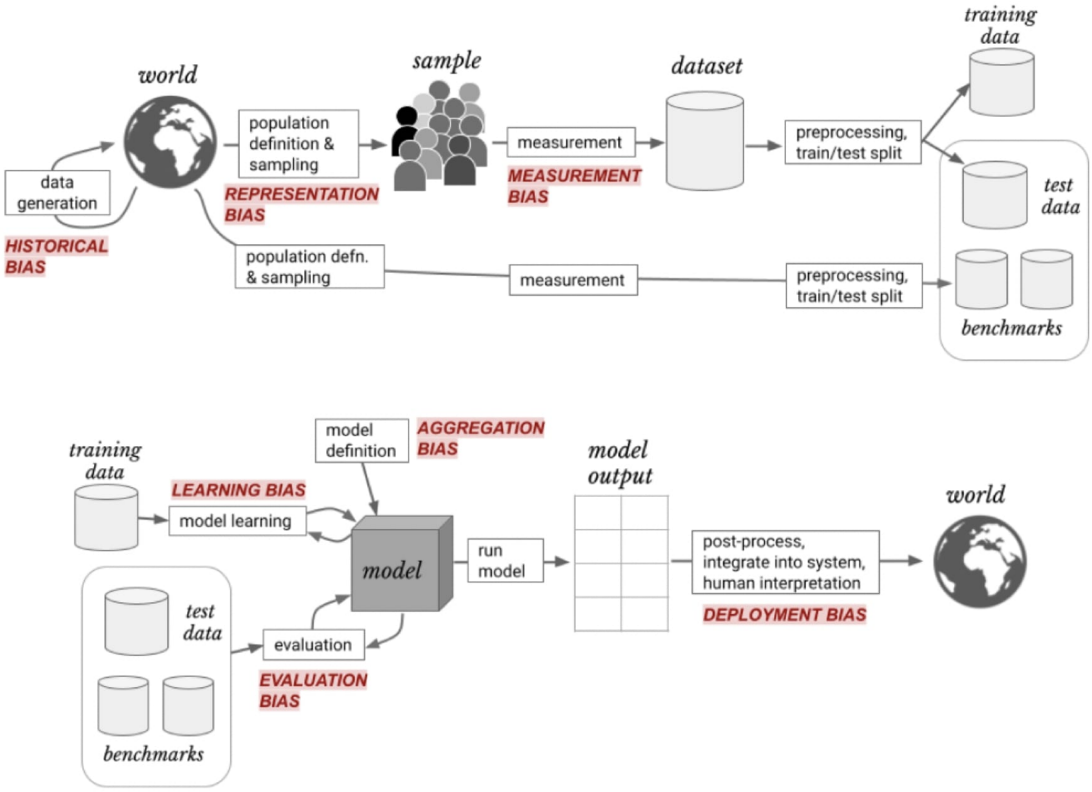
Meanwhile, it has been argued that, as a context-dependent ideal, Fairness should be instead defined according to the societal domain in which the decision-making model will be deployed (Srivastava, Heidari, & Krause, 2019). Otherwise, it may not promote improvement for sensitive groups in the given context.

2.1.3 Sources of Bias

Typically, ML algorithms operate by learning models from historical data and generalizing them to unseen data. As a result, problems with the data generation or model development process can lead to unintended consequences. A key challenge in Fairness-aware ML practices consists in identifying exactly what problems might be present, i.e., in understanding how bias is being introduced.

In Mehrabi et al. (2019), the author walks through the entire machine learning pipeline step-by-step, framing each potential source of bias and encouraging application-appropriate solutions as opposed to solutions that make implicit assumptions about the data and what it means to be fair. Some of the most well-known sources are:

- **Historical Bias**: corresponds to the already existing bias and socio-technical issues in the world;
- **Representation Bias**: occurs while defining and sampling the population, when the user population under-represents the target population and subsequently fails to generalize well;
- **Measurement Bias**: occurs while choosing and measuring features and labels to use, when the chosen set of features and labels leave out important factors or introduce group or input-dependent noise that leads to differential performance;
- **Aggregation Bias**: occurs during model construction, when false assumptions are made about a subgroup based on observing other different subgroups
- **Evaluation Bias**: occurs during model iteration and evaluation, when the target population do not equally represent the various parts of the user population, or when the performance metrics are not appropriate for the way in which the model will be used



(Suresh & Guttag, 2021)

- **Deployment Bias**: occurs after model deployment, when a system is used or interpreted in inappropriate ways
- **Population Bias**: occurs when statistics, demographics, representatives, and user characteristics are different in the population represented in the dataset from the original target population

As it can be seen, both the data and the ML pipeline involve a series of factors and choices, any of which can lead to unintended consequences. However, as the author points out, the very same outcomes that ML algorithms produce continually affect future data that will be collected for subsequent training rounds, creating a "feedback loop phenomenon" that intertwines the different bias definitions and makes it difficult to identify which source should be addressed.

2.1.4 Unfairness Mitigation Methods

Machine learning pipelines contain three possible points of intervention to mitigate unwanted bias: the collected data, the training data, the learning procedure, and the output predictions, with four corresponding classes of bias mitigation algorithms: **data collection**, **pre-processing**, **in-processing**, and **post-processing**.

An example of a **data collection** mitigation strategy is described in Gebru et al. (2020), where the author proposes a standardized process for documenting datasets, which, according to him, "fundamentally influence a model's behavior". By registering information such as motivation, composition, collection process and recommended uses, he hopes that researchers and practitioners are able to select more appropriate datasets while increasing transparency and accountability within the machine learning community. Additionally, in I. Chen, Johansson, and Sontag (2018), the author argues that even though much research has been devoted to constraining models and perturbing training data as a means to improve Fairness, these types of methods often harm predictive accuracy, as opposed to data collection.

Authors like Calmon, Wei, Ramamurthy, and Varshney (2017), however, choose to focus on **data pre-processing**, deeming it as the most flexible step in the ML pipeline due to its independence from the modeling algorithm and its possible integration with data release and publishing mechanisms. Kamiran and Calders (2012) additionally argue that the quality of the learned models depends critically on the quality of the training data, by going through various cases in which the input data are discriminatory. They also list existing preprocessing approaches, demonstrating how they all revolve around some sort of data transformation: suppressing the sensitive attribute, massaging the dataset by changing class labels, and reweighing or resampling the data to remove discrimination without relabeling instances.

Regarding **in-processing** methods, these involve some sort of modification of the learning algorithm during the training stage. In Kamishima, Akaho, Asoh, and Sakuma (2012), for example, the authors develop a technique that restricts the classifier's behavior through a regularization

parameter that controls the trade-off between predictive accuracy and degree of Fairness, where Fairness is deemed as the absence of prejudice (the statistical dependence between sensitive features and other information). A different modification is presented in Beutel, Chen, Zhao, and Chi (2017), where the authors remove information about the sensitive attribute and then use an adversarial training procedure that prevents the model from accurately predicting the sensitive attribute while trying to predict the target class, the idea being to discuss the effect of limited training data.

Finally, an example of **post-processing** techniques can be found in Hardt, Price, and Srebro (2016), which presents a framework that optimally adjusts any learned predictor through a simple post-processing step that minimizes the loss in utility, arguing that changing the original training process is often much more complex.

2.2 Fairness in Pricing

One of the ways Fairness concerns can be put into practice is in the context of pricing decisions. Several cases of illegal price discrimination have been documented (Alesina, Lotti, & Mistrulli, 2013; Angwin et al., 2016; Charles, Hurst, & Stephens, 2008; Pandey & Caliskan, 2020). However, as previously explained, unfair ML practices do not necessarily entail illegal discrimination, ultimately boiling down to individual perception.

2.2.1 Price Fairness Perceptions

While price changes are an expected part of everyday reality, the rationale for a price change might evoke negative price Fairness perceptions in customers. Even though a wide array of theoretical explanations has been developed, the majority of price Fairness research has been grounded in Equity theory's theoretical framework (J. Adams, 1965), which states that individuals judge the Fairness of their treatment based on how others like them are treated. Specifically, individuals compare the ratio between their own investment and its respective return with the ratio of the investment and return of another party enlisted in the exchange. Any significant difference between both ratios leads to perceptions of unFairness.

This theory links well with the highly cited Xia, Monroe, and Cox (2004), where the authors define perceived price Fairness as customers' perceptions and their related emotions about how fair, acceptable, and reasonable the difference is between two prices: the seller's price and the customer's reference price. This said, there are three possible outcomes: equality (the observed price is equal to the reference), disadvantaged inequality (the observed price is higher than the reference), or advantaged inequality (the actual price is lower than the reference).

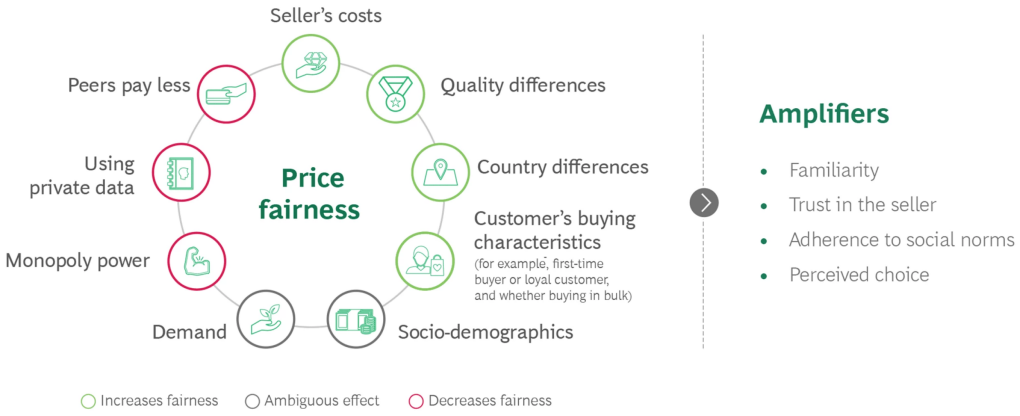
The reference price can either refer to a price the customer encountered on a past purchase or to the price another customer paid for the same product or service. Since customers have

difficulty in recalling the prices they paid before, several authors have stated that social comparisons play a larger role than comparisons with past transactions experiences in judging Fairness (Austin, McGinn, & Susmilch, 1980; Major, 1994; Major & Testa, 1989; Wood, 1989). However, this might change in the future with an increasing empowerment of customers through information accessibility and resulting price transparency.

Other factors have been concluded to influence price Fairness perceptions, namely the reasons behind the price change (Kahneman, Knetsch, & Thaler, 1986) and the general knowledge and beliefs about the seller's pricing practices (Xia et al., 2004), but these are not as salient to the customer as social comparisons and comparisons of current and past transactions.

Additionally, it has been concluded that consumers accept and positively assess price increases over time in terms of Fairness, and that people with higher personal income appear to be more desensitized to price differences and generally perceive price differences as more fair (Malc, Mumel, & Pisnik, 2016).

Exhibit 1 - Various Factors Influence a Consumer's Perceptions of Fairness



Source: Bruce Henderson Institute (BHI) survey.

Note: BHI surveyed 13,000 individuals in eight countries: Brazil, China, France, Germany, India, Japan, the US, and Sweden.

Figure 2.1: Factors that influence a customers perception of fairness (Izaret, 2022)

To sum up, as it can be seen on Figure 2.1, the consumer's fairness perception can be influenced by several factors, which were previously described above. It is noticeable that other factors can also amplify the fairness perception, for example, if the customer is familiar with or has trust in the seller.

2.2.2 Algorithmic Pricing

The increase in the recognition of the price Fairness concept can be attributed to the appearance of algorithmic pricing techniques, where price changes are a lot more frequent.

Simply put, algorithmic pricing is a pricing strategy that builds on computer algorithms to automatically generate dynamic and customer-specific prices. These algorithms process data about markets and actors, accounting for numerous factors such as competitors' prices, market demand, inventory constraints and even consumer behavior and characteristics to determine price in relation to highest achievable profit (Cohen, 2018; Fisher, Gallino, & Li, 2018; Keskin & Zeevi, 2014; Zhang, 2011).

In other words, there are two dimensions to algorithmic pricing: dynamism and personalization. The first one describes the practice of continually adjusting prices to a set of conditions, allowing firms to manage and control inventory efficiently and to react to market demand and competition in real-time (N. Chen & Gallego, 2019; Q. Chen, Jasin, & Duenyas, 2016). The second one introduces a strategy "whereby firms charge different prices to different consumers based on their willingness to pay" (Choudhary, Ghose, Mukhopadhyay, & Rajan, 2005), which firms are able to estimate thanks to digital tracking via 'cookies' and 'digital breadcrumbs'. Indeed, online behavior creates a long data trace consisting of individual purchasing preferences and habits that allow firms to build fine-grained customer profiles, often revealing their income or health status (Bitran & Caldentey, 2003; Ezrachi & Stucke, 2016).

2.2.3 Ethicality in Algorithmic Pricing

By using algorithmic pricing approaches, companies often try to engage in price discrimination tactics, charging different prices for identical items. There are three classic degrees of price discrimination (Pigou, 1932):

- **First-degree price discrimination** (or personalized pricing): It can be described as charging individual prices for identical products or services due to individual traits. These individual prices are expected to represent the customer's maximum willingness to pay, that is, their reservation price.
- **Second-degree price discrimination** (or menu pricing): It refers to pricing schemes in which the price does not depend on characteristics of the customer but on the quantity bought. Different prices are attached to specific quantities of a good or service, so it is said that customers "self-select": they choose a different price by choosing a different quantity.
- **Third-degree price discrimination** (or group pricing): It consists in charging different prices to different groups of customers, meaning that customers are not individually recognized.

Depending on what tactic is used, firms' excitement can be met by consumer concerns (Borgesius & Poort, 2017). Regarding second-degree price discrimination, as long as all customers have the same access to better prices and it is only the individual choices for quantity that lead to different prices, this type of discrimination is perceived as fair by consumers. Additionally, it is generally accepted that "a man's rewards in exchange with others should be proportional to his investments" (Homans, 1961). Third-degree price discrimination is also usually uncontroversial, since it is often implemented as a "voluntary" gesture: many companies offer special discounts to senior citizens, students, children, i.e., groups with a lower willingness to pay. However, there are some cases where these practices lead to over-identification of the customers, pushing towards perfect or first-degree price discrimination. On the other hand, many people perceive first-degree price discrimination as unfair or manipulative, even if it translates into benefits for the customer. According to a US survey, 78% of the respondents did not want "discounts (…) tailored for you based on (…) what you did on other websites you have visited", condemning the underlying invasion of digital privacy (Turow, King, Hoofnagle, Bleakley, & Hennessy, 2009).

Thus, it can be seen that unFairness judgement due to personal concerns could be changed to being perceived as fair if it is socially justifiable, which happens in second and third degree price discrimination. On the other hand, in personalized pricing there is no social component that could modify the impact of the personal component on perceived price Fairness.

It should also be noted that, since prices can fluctuate for generic reasons, it might be difficult to bring personalized pricing to the customers' attention. Additionally, there are some tactics that frame personalized prices as explicit personal offerings, thus mitigating unFairness perception and even raising the chances that consumers accept the prices offered (Barone & Roy, 2010).

2.2.4 Legal Protection against Algorithmic Pricing

Since personalized pricing is met with much skepticism, a valid question is if existing law can be of help. As previously seen in section 2.1.1, according to the GDPR, data protection law applies if personal data are processed. Typical examples of personal data are names, personal email addresses and identification numbers, i.e., "any information relating to an identified or identifiable natural person ('data subject')", as stated in Article 4. This means that cookies containing data sufficient to identify users qualify as personal data as well.

In Borgesius and Poort (2017), the author argues that most personalized pricing entail the processing of personal data and therefore must comply with the GDPR, that is, the data processing must happen fairly, lawfully, and transparently. Using pricing practices that discriminate against protected attributes (i.e., race, colour, sex, sexual orientation, age, physical or mental disability, marital status, etc.) might be, consequently, considered illegal.

2.3 Pricing in Telecommunications

Fueled by technological progress and an unprecedented amount of (personal) data, algorithmic pricing has quickly spread into multiple industries - electricity and gasoline markets (Faruqui & Sergici, 2010), sports and entertainment (Bouchet, Troilo, & Walkup, 2016), airlines (van Ryzin & McGill, 2000) - and is now considered a key driver of business success.

The Telecommunication industry too has seen a digital transformation of its businesses. This is a really competitive market where aggressive acquisition campaigns happen on a daily basis, following the shift in the behavior of the Consumer, who now constantly interacts through digital channels and expects highly dynamic and personalized offers, looking for the best possible deals in terms of diversity, customization and price. Telecom & Media companies have no option other than to leverage the user data within their networks to create good personalized user experiences.

2.3.1 Product and Service Bundles

Bundles in the telco market consist in the combination of several services such as mobile connectivity and broadband allocation, usually granted for a monthly payment. These services are often marketed together with other multimedia and entertainment services such as streaming TV, music, and storage service subscriptions. Stremersch and Tellis (2002) identify two categories of bundling:

- **price bundling**: the sale of two or more separate products in a package at a discount without any integration of the products;
- **product bundling**: the integration and sale of two or more separate products in a discounted package with added value to some consumers.

Another way of classifying bundling strategies is how the different products are offered (W. Adams & Yellen, 1976):

- **pure bundling**: the products are sold only bundled;
- **mixed bundling**: the products can be sold bundled or separately.

In Chan-Olmsted and Guo (2011), it is postulated that telco firms are expected to practice the more strategic "product bundling", adding value from the integration of broadband Internet with IPTV video and IP phone services. Also, to adopt a "mixed product" bundling strategy, which offers more opportunities for differentiation in such a competitive marketplace.

2.3.2 Bundling Benefits

Bundling is one of the most effective marketing tools in e-commerce. Consumers receive a more diverse and customized set of offers, at discount rates and lower shipping costs (Bai et al., 2019). However, customers are not the only ones benefiting from product and service bundling. Considering that customers have asymmetric valuations of separate products, by selling these products together in a package, companies are able to reduce uncertainty and consumer valuation variance, thus achieving higher profit (Rautio, Anttila, & Tuominen, 2007). Additionally, sellers benefit from higher revenue with shared fixed operational costs involved with the packaging and shipping processes, while exposing consumers to new items that they may have not considered in isolation (Garfinkel, Gopal, Tripathi, & Yin, 2006). Finally, bundling also changes competitive structure and reduces the threat of focused specialists.

Chapter 3

Data Understanding

3.1 Introduction to the Dataset

This work processed data used for the company's product recommendation system – Next Best Offer (NBO). Gathered from several different sources in the database system – "portfolio", "catalog" and "actions" -, these data were ultimately compiled into a table containing a semantic representation of past actions taken toward customers in the context of marketing campaigns.

An action corresponds to any type of communication with the client, but in this work only commercial offers were considered. When making an offer, the company needs to consider several pieces of information: the historical actions that were previously made to the client ("actions"), the products the client currently owns ("source portfolio"), the features of each of these products ("catalog"), the client's aggregated mobile usage and other service details such as loyalty period, direct debit or contact preferences. Based on that, clients are recommended a new bundle that either unlocks more features or expands the features already available ("destiny portfolio"), and the result of the sale opportunity is registered.

Possible products constituting a customer portfolio are the following: base products (television, landline, fixed internet), integrated mobile cards (voice and internet), and premium TV channels/packs subscriptions. Depending on what gets included, customers can have packages containing 1, 2, 3, 4 or 5 different products. In this particular dataset, only packages containing at least 2 products can be observed.

3.2 Dataset Characteristics

The original dataset provided by the Telecom company consisted of 1419377 observations (rows) across 335 variables (columns), where each row corresponded to a specific action (offer)

toward a client and columns represented different information regarding that action. However, some variables were immediately excluded for their overall irrelevancy in the context of this problem: the execution's ID, the action's ID, associated tags, user of creation, among others. Additionally, several duplicate rows were removed and the dataset was filtered so that it only included the most recently executed actions and this work would focus on an updated set of products and offers. These adjustments resulted in a first workable version of the dataset that included 1123841 observations and 318 variables.

Due to the big amount of variables in the data, a table with the variables description is available on Appendix A.1. This table presents, in the same order as in the dataset, the most relevant variables in regard to their meaning and data type.

As it can be seen, besides the client identification, there are four main groups of variables, representing different types of information and different data types.

- **Catalog variables**: represent features of four products: television ("tv"), fixed internet ("if"), mobile voice cards ("vm") and mobile internet cards ("im"), where television and landline are assumed to be already included in the clients' package, even if not explicitly represented in the dataset. Most of these are considered as categorical as they typify packages, the only numerical variables being those that regard revenue.
  - *Catalog tech variables*: are all binary and regard different accessibility technology, from coaxial and fiber optic cable to satellite and Router 4G technology.
- **VM variables**: specifically analyse mobile voice cards in greater detail, aggregating communication data from/towards a different telecommunication company.
- **SRS variables**: store historical data about the client's service, including technical problems, number of contact attempts, positive/negative responses, number of payments and so on.
- **Base variables**: either represent financial information about the client, such as discounts, current and previous loyalty period or invoice value, or indicate which forms of contact the client has authorized: letter, email, SMS, etc.

Chapter 4

Data Preparation

Upon close inspection of the values assumed by these variables, it was immediately understood that the dataset would need some encoding procedures. It was also noticed that the number of variables itself could yield computational struggles in the long-run, therefore calling for data reduction.

In order to work with a more accurate and concise dataset that also complies with specific ML techniques, several data preprocessing tasks were conducted on the features described in Chapter 3, from data reduction to data cleaning and data transformation procedures.

4.1 Uninformative Variables

A good basis for data reduction starts with filtering out columns with very few unique values. These are referred to as "near-zero variance predictors" and are likely to contain little valuable predictive information as well as to cause errors or unexpected results (Kuhn & Johnson, 2019).

This said, it was observed that a total of 165 variables contained useful information while 33 only showed 1 unique value. Not all of these 33 variables were removed, though, as 6 of them were non-catalog features. This is because a major step of this work, which will be explained later on, consisted in comparing bundles, where uninformative catalog-related variables do not create differences between bundles and so become irrelevant. Consequently, the remaining 27 uninformative features were excluded from the dataset.

The following countplot on Figure 4.1 represents the number of unique values each variable assumes across the entire spectrum of observations.

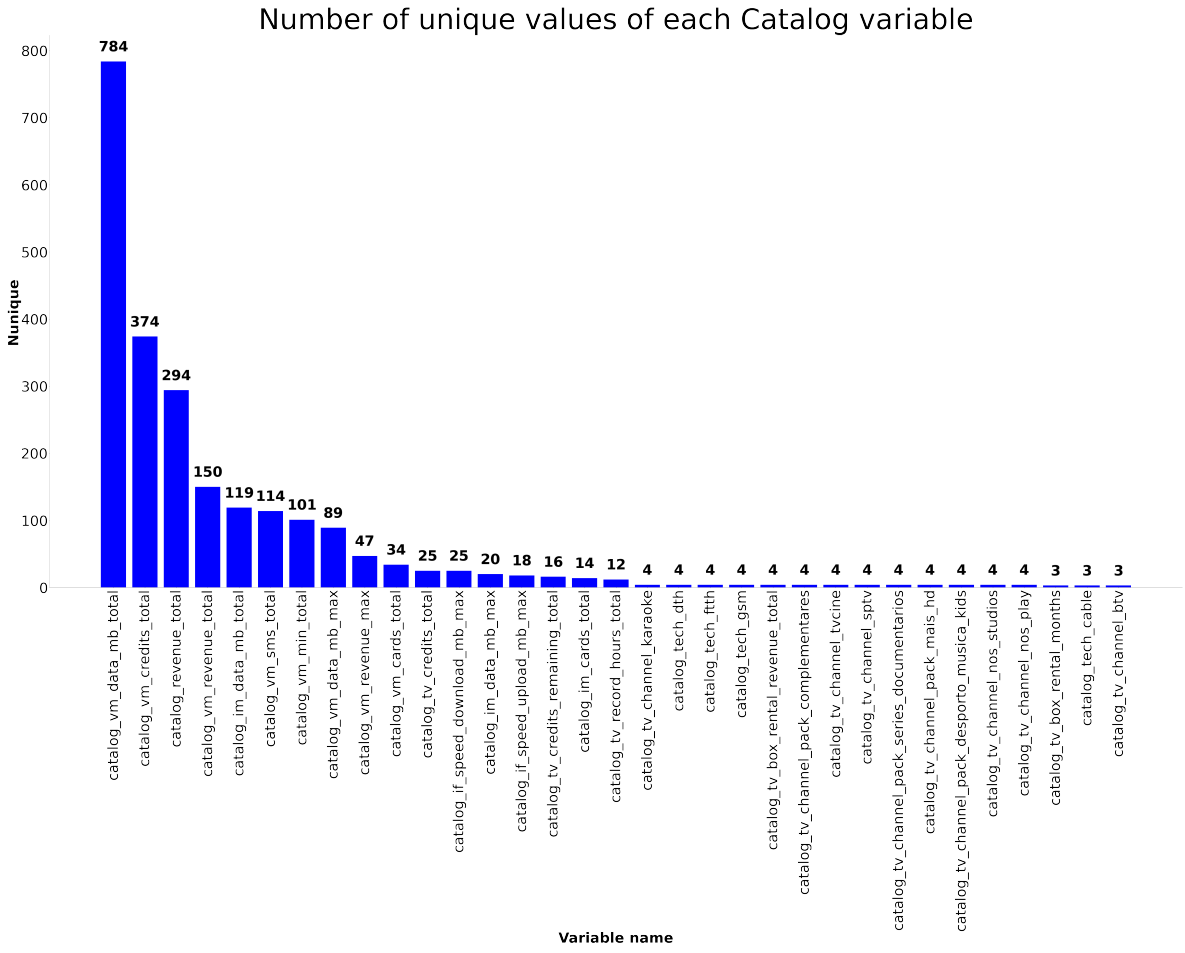


Figure 4.1: Number of unique values per variable

	Informative	Uninformative
Catalog	33	27
Not Catalog	132	6

Table 4.1: Number of catalog/non-catalog variables with and without information

It can be seen that the majority of the variables assume very few unique values. Specifically, a total of 107 variables present up to two unique values. However, these should not constitute automatic candidates to be dropped from the dataset. For example, categorical columns, by design, are not expected to have a high number of unique values, and in this particular dataset a lot of variables are specifically binary. This said, 63 of these 107 variables corresponded to zero-variance predictors, i.e., variables with only one singular value, which are truly redundant and should be excluded from the dataset.

4.2 Missing Values

Dealing with missing values is another important task, in the context of data cleaning. Most of the machine learning models will produce an error when given NaN values so these are either dropped or replaced during data cleaning. Even though the latter is usually preferred as to not lose important information, it requires minimum understanding of the sources of missing data. For example, if a value is missing because it simply does not exist, then it does not make sense to guess what it might be. On the other hand, if a value is missing because it was not recorded, then it should be estimated based on the other values of the same column. This approach is referred to as Imputation.

In the case of Catalog variables, clients are not expected to own all available products, so it becomes difficult to tell whether a specific feature is actually included in a client's package, hampering the imputation task. Additionally, if the percentage of missing values is not significant, excluding them will not be problematic as it will not translate into a great loss of information. This said, for these particular variables, missing values had been previously encoded as "999999" or "9999999", as confirmed by the company personnel, and only corresponded to 2.7% of the total observations. Thus, they were removed.

As for the remaining variables, they contained missing values in approximately 97% of the observations, calling for a much less risky solution that would not discard this much valuable information: imputation. Specifically, two different imputation strategies were implemented, depending on the data type. The first one was made considering the median, since it is a more robust measure than the mean (Mislevy, Little, & Rubin, 1991), and was only applied to numeric data. Missing values in categorical features, in turn, were replaced by the most frequent values

within each column. A common disadvantage associated with both strategies is that they do not factor the correlations between features, besides not being very accurate at times. On the other hand, they are significantly simpler and faster to execute than other more computationally expensive strategies like Imputation based on k-NN or based on Deep Learning.

4.3 Datetime Conversion

Date/time is another rich source of information that needs some type of feature engineering process that converts it to a numeric format. The most common numerical conversion methods include:

(1) Unix Timestamp: Unix time is a single signed number that increments every second as it represents the number of seconds that have passed since 00:00:00 UTC, 1 January 1970, an arbitrary date called "Unix Epoch". Therefore, it can describe a specific point in time as the number of seconds between that specific date and the "Unix Epoch".

(2) KSP date format: This format uses the year and adds the month/day within the year as a fraction:

$$KSPdate = year + \frac{ndays - 0.5}{365 + isleapyear}, \tag{4.1}$$

where year is the year of the date in question; isleapyear takes the value of 1 if year is/was a leap year and of 0 otherwise; and ndays serves as a day counter that calculates the number of days between the date in question and the beginning of the year, so ndays would be "1" for January 1st and "365" for December 31st (366 in a leap year). It can be extended to include the time as well.

(3) Dividing into several features: Instead of a single variable displaying the whole date, new variables could be created to replace it, each representing one of the following: year, month, day, hour, minute, second.

(4) Constructing a new feature: Here, the new variable is not a perfect replacement of the original date variable, although it describes similar information that may be more/less relevant to the modeling problem. For example, the dataset could use "age" instead of "date of birth", "time to delivery" instead of "date order created" and "date order delivered".

In this case, the dataset at hand only included one date-time variable ("timestamp"), which followed a year, month, day, hour, minute, second format (for example, '2018-01-04 00:00:00'). Out of the four methods described, the Unix Timestamp seemed to be the least disadvantageous, considering the cons of the other three: (2) does not preserve intervals between days perfectly

due to leap years; (3) can make the model unnecessarily complex if too many new dimensions are added; and (4) might lead to important information loss. This said, the Unix Timestamp's disadvantage of not being as intuitive was not considered as critical, and so, this method was used to transform "timestamp" into a numeric format.

4.4 Label Encoder and One Hot Encoder

Finally, there is Data encoding, which means mapping all the possible values in a **categorical** feature to a corresponding **numeric** value and it frequently comes into play due to ML algorithms not being able to process categorical data. There are a few models, mostly Tree-based, that do a better job in handling categorical variables, however, this data transformation task is still wildly performed on the basis that the majority of ML algorithms perform better when the data is represented as a number instead of categorical.

This said, there are many ways to convert categorical values into numerical values, each approach with its own trade-offs and impact on the feature set. In this work, specifically, two well-known categorical encoding techniques were implemented, **Label Encoder** and **One Hot Encoder**, to deal with the following variables:

1. **base_credit_classification**, which describes the client credit classification, that is, the reputation of the client in terms of their ability to pay and if there is any risk of debt according to the client's financial status data.
2. **base_status_collection**, where the last notice message that was sent to the client is registered. This can regard delayed payments, contract renewals, contract termination, and others.
3. **base_bad_payer_status**, which indicates whether a specific client has ever failed to pay to the company and to what extent.
4. **base_fiscal_number**, which simply represents the client's fiscal number.

So, even though all these variables are categorical, while **base_fiscal_number** takes numerical values, variables 1,2 and 3 are defined as string features. In terms of uniqueness, there are also some significant differences: variables 1 and 3 show little variety, with 13 and 4 distinct values, respectively; variable 2 contains a total of 51 different labels; and variable 4 takes on 544629 unique fiscal numbers.

In order to better understand the transformation this data went through, the two encoding techniques are briefly described in the following subsections.

4.4.1 Label Encoding

In this approach, each label is assigned a unique integer between 0 and N, where N corresponds to the total number of unique values in a categorical feature, so it does not affect the dimensionality of the dataset. Which label is assigned the value 0, 1, 2, and so on, i.e. the ordering rule, may change for different versions of scikit-learn, but for this specific work, the conversion was done in sort order (for numerical categorical variables) and alphabetic order (for string variables).

Considering, as an example, some dataset with a column containing fruit names, the following table shows how this feature would be transformed after applying label encoding to it.

Original encoding	Label Encoding
Apple	0
Cherry	2
Banana	1
Kiwi	3

Table 4.2: Label Encoding

This may lead to the generation of priority issues in the training of data sets. A label with a high value may be considered to have high priority than a label having a lower value.

It should be pointed out that even though there is no relation between various fruits, when looking at the numbers, one might think that 'Apple' has higher precedence over 'Kiwi'. Similarly, there might be priority issues in the training of datasets for ml algorithms. This is Label Encoding's main disadvantage.

4.4.2 One-Hot Encoding

This ordering issue is addressed in the One-Hot Encoding approach, where each unique label is converted into a new column with binary encoding (0 or 1) to denote whether a particular row belongs to this category. Consequently, however, there might be a significant increase in the dimensionality of the dataset, which is not a problem for the previous approach.

Extending the previous example, this next table now shows how the fruit feature would be transformed after applying one-hot encoding to it.

4.4.3 Technique Complementarity

There are various reasons for combining these two techniques. First off, variables 2 and 4 contain a large number of distinct values, which would raise dimensionality issues if transformed

Original Encoding	One-Hot Encoding			
	Apple	**Cherry**	**Banana**	**Kiwi**
Apple	1	0	0	0
Cherry	0	1	0	0
Banana	0	0	1	0
Kiwi	0	0	0	1

Table 4.3: One-Hot Encoding

by one-hot encoding. Secondly, although label encoding gets around this disadvantage, it can misinterpret numeric values as having some sort of hierarchy/order in them. The limitations almost seem complementary to one another, which is why if both approaches are used, one can successfully encode all categorical features without adding too many columns to the dataset nor introducing false order to the data.

With this in mind, for this work, label encoding was first run on variables 2 and 4, adding a total of 17 new columns to the dataset, followed by one-hot encoding on variables 1 and 3, which produced an output of non-ordinal values.

4.5 Standardization

Standardization is another major step of the data preprocessing stage where the values are centered around the mean with a unit standard deviation. It is usually performed when features are measured in different scales or have large differences between their ranges as this can prove to be a significant obstacle for many Machine Learning models and methods that are sensitive to the magnitude of variables. For example, in distance-based models, if one of the features has a broad range of values, it will dominate the entire distance metric. On the other hand, tree-based algorithms do not require standardization prior to being fitted as a split on a feature is never influenced by other features.

This said, this specific work included multiple features spanning varying degrees of magnitude, calling for a standardization technique.

4.6 Creation of ID Variable

An important piece of information that, for process reasons, was missing in the original dataset was the identification number for the packages involved in each offer, which had to be artificially added. This variable was computed at the end of data cleaning and according to each

unique combination of the remaining product features, including price, represented by "catalog" variables. In total, 71545 unique packages were identified, based on the following 33 features.

Table 4.4: Variables used to distinguish packages

Package composition
catalog_revenue_total
catalog_tech_dth
catalog_tech_cable
catalog_tech_ftth
catalog_tech_gsm
catalog_vm_data_mb_total
catalog_vm_credits_total
catalog_vm_revenue_total
catalog_vm_sms_total
catalog_vm_data_mb_max
catalog_vm_min_total
catalog_vm_revenue_max
catalog_vm_cards_total
catalog_im_data_mb_total
catalog_im_data_mb_max
catalog_im_cards_total
catalog_if_speed_download_mb_max
catalog_if_speed_upload_mb_max
catalog_tv_credits_remaining_total
catalog_tv_record_hours_total
catalog_tv_box_rental_months
catalog_tv_box_rental_revenue_total
catalog_tv_channel_pack_series_documentarios
catalog_tv_channel_sptv
catalog_tv_channel_nos_play
catalog_tv_channel_karaoke
catalog_tv_channel_pack_complementares
catalog_tv_channel_pack_mais_hd
catalog_tv_channel_tvcine
catalog_tv_channel_nos_studios
catalog_tv_channel_btv
catalog_tv_channel_pack_desporto_musica_kids

4.7 Cleaned Dataset

The resulting dataset consisted only of 1384972 rows and 71 columns, the main difference consisting of fewer product features and the additional source and destination package IDs.

Chapter 5

Fairness Process

As previously stated, Fairness is quite a subjective ideal that requires context-appropriate approaches. In a product recommendation setting characterized by very different product configurations and individual prices, it might seem intuitive to analyse "Fairness" from a "client versus client" perspective, under which a certain package would be "fair" if no other package had been built with the same features and sold to another client at a lower price. Similarly, a client would also not feel exploited as long as other clients did not pay a similar price for a bundle with more features.

However, with the telecommunication sector being a very competitive market with aggressive individual campaigns, it seemed more appropriate to comply with this personalization strategy and refrain from focusing exclusively on comparisons between clients. This was accomplished by incorporating an intrapersonal perspective into the concept of Fairness: "then versus now", which compares a client's current situation in terms of fairness to the situation they would be in on the premise that a specific offer is accepted. Several previous works have described the differences between these fairness levels (Räz, 2021) (Binns, 2020).

In other words, for this work, the concept of Fairness was built on the combination of two levels of comparison: **interpersonal** and **intrapersonal**. On the interpersonal level, the client's "**Fairness Position**" is calculated, and on the intrapersonal level, an offer's "**Fairness Impact**" is estimated.

5.1 Fairness Position

As briefly explained, for this work, it was important to measure a client's situation in terms of Fairness, i.e., how "fair" a client's bundle was. This notion is referred as "fairness position" and results from a comparison between clients and their corresponding bundles. Different kinds

of bundles can be considered depending on what the position is meant to represent, resulting in two types of "fairness positions" for each client:

- **current fairness position**: represents the degree of fairness of a bundle that a client currently owns ("source" bundle), i.e., corresponds to the the fairness situation a client is currently in by owning a specific bundle;
- **hypothetical fairness position**: represents the degree of fairness of an offer that a client receives ("destiny" bundle), i.e., corresponds to the fairness situation a client will be in by accepting a specific offer.

This said, a client's fairness position is calculated as the price difference between two bundles: either the "source" or the "destiny" bundle of the client whose fairness position we want to determine and a bundle that constitutes a fairness benchmark for that "source"/"destiny" bundle, known as the "fair" bundle (see equations 5.1 and 5.2).

$$FP_{current} = price_{fair} - price_{source} \tag{5.1}$$

$$FP_{hypothetical} = price_{fair} - price_{dest} \tag{5.2}$$

Considering a scenario where:

- a client's current bundle costs 40€;
- they are offered a different bundle for 42€;
- the fairness benchmark for the current and offered bundles is at 37€ and 32€, respectively;

The values for Fairness Position would be:

$$FP_{hypothetical} = 32 - 42 = -10 \tag{5.3}$$

$$FP_{current} = 37 - 40 = -3 \tag{5.4}$$

Thus, it can be concluded that both current and offered bundles represent an undesirable situation for the client as they could have been paying less for a very similar, if not equal, bundle.

5.1.1 Fairness Criteria

In order to elect a bundle as the fairness benchmark for each "source"/"destiny" bundle, a fairness criteria was used based on the following conditions:

1. **ownership**: the "fair" bundle must correspond to an already existing bundle that is owned by at least one client;
2. **life cycle similarity**: a bundle's life cycle corresponds to the time period that goes from the earliest start date to the latest end date of any client's loyalty period involving that specific bundle. This said, the life cycle of the fair bundle must either contain or be contained in the life cycle of the corresponding "source"/"destiny" bundle, with a tolerance period of 6 months, to account for the the Telecom market's dynamic: if no time constraint was to be made, it would allow for a bundle to be compared with a much older one that was no longer relevant to the more recent market conditions.
3. **highest composition similarity**: as previously explained, bundles are characterized by the set of "catalog" variables, with few exceptions. Therefore, similarity between bundles was estimated by measuring distance between these features. The fair bundle must have the lowest distance from the corresponding "source"/"destiny" bundle, among the set of eligible bundles that have previously satisfied condition 1.
4. **lowest price**: The fair bundle must have the lowest price, among the set of eligible bundles that have previously satisfied both condition 1 and condition 2.

5.1.2 Similarity Measurement

As mentioned, one of the central challenges in this work was to measure similarity between bundles. This was approached from a distance angle and was accomplished in three steps:

1. defining which set of variables characterize a bundle;
2. transforming those variables into a smaller set of numerical features with a dimensionality reduction technique: Multiple Correspondence Analysis (MCA);
3. measuring distance between new features with an appropriate distance metric: Euclidean distance.

Bundle Characterization

Regarding Step 1, it has been previously explained that "catalog" variables are features that describe bundles, representing, for example, the inclusion of a specific TV channel or the number

of mobile internet (IM) cards. However, not all "catalog" features should move into the next step: if the bundle's price was considered as part of a bundle's composition and contributed to the distance computation (in Step 3), this could result in two differently priced bundles having equal composition but a distance value greater than zero from each other. This said, all "catalog" features but those that regard **revenue** were selected in Step 1.

Multiple Correspondence Analysis

Considering this distance approach, where bundles with identical composition should present zero distance between them, it was important to think of distance metrics when formulating Step 2. Even though there are various distance functions for categorical variables, for reasons to be explained later, the Euclidean distance was preferred.

This metric, however, can only be applied to numerical data. At the same time, the high number of variables in the dataset could mean more noise in the data and lead to a worse overall performance of ML algorithms later on.

This said, it was deemed appropriate to use a Dimensionality Reduction method such as Multiple Correspondence Analysis (MCA).

Dimensionality reduction generally refers to the process of reducing the number of attributes in a dataset while retaining as much of the variation in the original dataset as possible. However, other definitions can be brought up depending on the approach used. The following scheme shows the existing grouping of different types of dimensionality reduction methods.

As it can be seen, there are multiple ways of reducing dimension. Some filter out the less important features, while others combine all variables into a subset of new ones that contains most of the original information. Additionally, these features can be combined through a linear or non-linear process.

Besides the reasons previously mentioned, and considering the extensive list of existing dimensionality reduction methods, there were other factors taken into account when selecting the MCA approach.

For starters, even though Forward selection and Backward elimination do filter out variables, they are only employed when building regression models, in the context of supervised learning. So using these techniques here would not really make sense.

Secondly, when combining features, while a nonlinear approach seems to be more powerful and successful, especially for complicated, high-dimensional datasets, linear methods have their own set of advantages that can turn out to be more appropriate depending on the context.

In this particular situation, issues such as computational efficiency, easy of use and forward mapping were considered of higher importance, reinforcing the preference for the MCA approach:

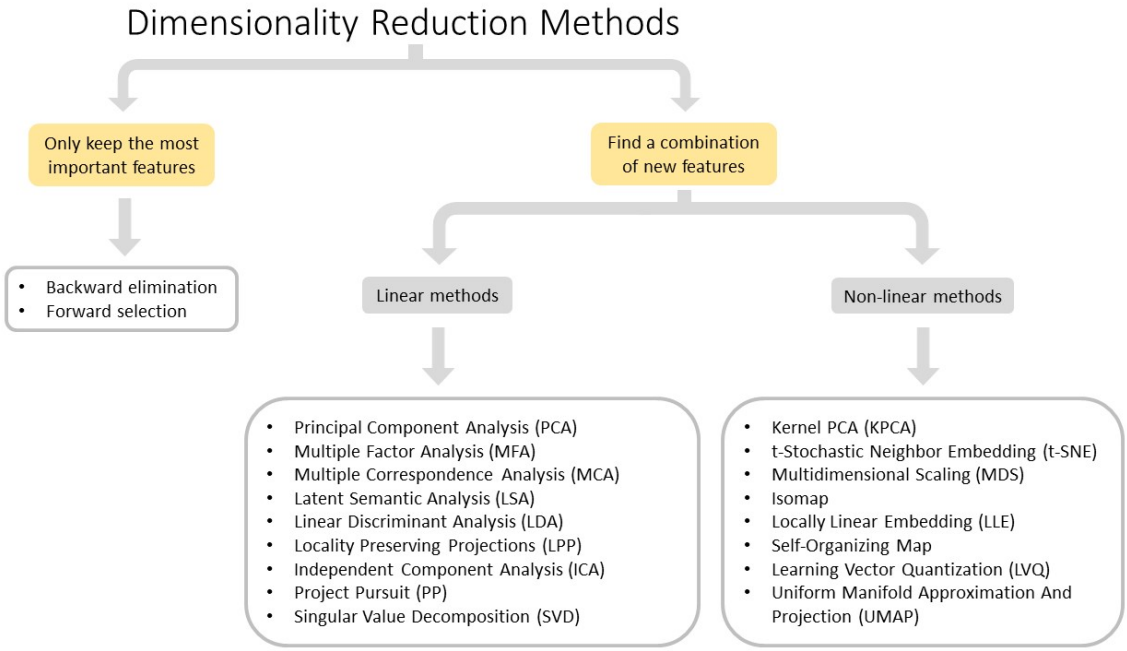


Figure 5.1: Dimensionality Reduction Methods

- Computational efficiency: MCA is more time/memory efficient than more complicated nonlinear techniques;
- Easy of use: MCA only supports adjustments in the number of dimensions, while nonlinear techniques can require many hyperparameters;
- Forward mapping: It is possible to apply the same MCA transformation to out-of-sample data that wasn't part of the training set. This, however, is not the case for most nonlinear methods, and even if it is, it requires additional runtime, learning, and/or hyperparameter tuning.

Additionally, the MCA approach derives from the old, trusted, and widely known standard that is PCA, so more implementations have been widely available.

Multiple Correspondence Analysis

Multiple Correspondence Analysis (MCA) is a data analysis technique that allows to study the pattern of relationship of several categorical variables (Greenacre, 2007). In simple terms, it can be understood as a generalization of Correspondence Analysis (CA) to the case where there are more than two variables. Alternatively, it can also be seen as the counterpart of Principal Component Analysis (PCA) when the variables to be analyzed are categorical instead of quantitative.

MCA can be performed by applying the CA algorithm using one of the two approaches: (A) based on an indicator matrix - called *complete disjunctive table* (CDT) or (B) based on a *Burt table*.

(A) Complete Disjunctive Table based approach

According to Abdi and Valentin (2007), after the dataset is completely represented as categorical variables, the indicator CDT matrix is constituted by individuals $\times$ variables matrix where the rows represent individuals and the columns are dummy variables denoted by 0 or 1 to identify the categories of the nominal variables. Through the indicator matrix analysis, it is possible to obtain the associations between variables by calculating the chi-square distance among different categories of the variables and among the individuals. This way, the CDT matrix allows the direct representation of individuals as points in geometric space. Similarly to PCA, the number of axis to be retained depends on the eigenvalues and the first axis is the most important dimension, the second axis the second most important, and so on, in terms of amount of variance (inertia) calculated. The number of axis to be retained depends on the eigenvalues.

Theoretical development:
Let $\mathbf{X}$ be the $I \times J$ indicator matrix, with K being the number of nominal variables where each variable has J_k levels, such as $\sum_{k=1}^{K} J_k = J$. All the elements of $\mathbf{X}$ matrix are equal to 0 or 1. Assuming that N denotes the number of 1's and considering the matrix $\mathbf{Z} = \mathbf{X}/N$, the vectors **r** and **c** are constituted by the row and column totals, respectively. The factor scores are obtained from the following decomposition

$$\mathbf{D}_r^{-\frac{1}{2}}(\mathbf{Z\text{-}rc}^T)\mathbf{D}_c^{-\frac{1}{2}} = \mathbf{P}\Delta\mathbf{Q}^T,$$

where $\mathbf{D}_r = diag\{\mathbf{r}\}$ and $\mathbf{D}_c = diag\{\mathbf{c}\}$ are diagonal matrices, Δ is the diagonal matrix of the singular values, and Δ^2 is the matrix of the eigenvalues. The row and columns factor scores are given respectively by

$$\mathbf{F} = \mathbf{D}_r^{-\frac{1}{2}}\mathbf{P}\Delta; \quad \mathbf{G} = \mathbf{D}_c^{-\frac{1}{2}}\mathbf{Q}\Delta.$$

The previous expressions give the coordinates of observations and variables (respectively) in the factor space. The squared distance from the rows or columns to the correspondent barycenters are given by matrices

$$\mathbf{d}_r = diag\{\mathbf{F}\mathbf{F}^T\}; \quad \mathbf{d}_c = diag\{\mathbf{G}\mathbf{G}^T\}.$$

The squared cosines between row i, or column j, and factor l can be given respectively by

$$o_{i,l} = \frac{f_{i,l}^2}{d_{r,i}^2}; \quad o_{j,l} = \frac{f_{j,l}^2}{d_{c,j}^2}$$

with $d_{r,i}^2$ and $d_{c,j}^2$ denoting the i-th element of $\mathbf{d}_r$ and the j-th element of $\mathbf{d}_c$, respectively. It is important to analyse the squared cosines since they give some information concerning

the location of the most important factors for a given observation or variable. It should be noticed that if for a given category, the cosines are low on the axes of interest, then any interpretation of the position of the category is hazardous. The analysis of the squared cosines allows to avoid misinterpretations of the plots that are due to projection effects.

The contributions of row i and column j to factor l are obtained respectively as

$$t_{i,l} = \frac{f_{i,l}^2}{\lambda_l}; \quad t_{j,l} = \frac{g_{j,l}^2}{\lambda_l},$$

with λ_l being the l-th eigenvalue. Contributions give information in locating the most important observations or variables for a given factor; they are helpful for interpreting the plots. The categories that most influenced the calculation of the axes are those with the higher contributions.

Since categorical variables are coded as binary columns with the constraint that only one of the columns takes the value 1, variance is artificially inflated, leading to severe underestimation of the percentage of variance explained by the first dimension. In order to obtain a better estimate of the variance extracted by each eigenvalue, Abdi and Valentin (2007) and Greenacre (1993) suggested some formulas to correct the eigenvalues values.

(B) Burt table based approach

The Burt table is the symmetric matrix of all two-way cross tabulations between the nominal variables, and has an analogy to the covariance matrix of continuous variables (Greenacre, 2007). In a sense, Burt table is a more natural generalization of simple CA, where individuals can be easily added as supplementary points to the graphic display.

As a result of the MCA procedure, the selected "catalog" features were replaced by 10 new variables of numerical input, which are typically referred to as Principal Components. In the subsequent dataset, they were named "dim" and were ordered by the amount of variance each explained in the original dataset.

Euclidean Distance

As previously stated, given the numerical output of the MCA approach, i.e., the numerical nature of the "dim" variables, the distance metric used was the Euclidean distance. It can be explained as the length of a segment connecting two points and is used for numerical data. In n-dimensional data, it is given by Tabak (2014):

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2},$$

with **x** and **y** representing different bundles, **i** representing a "dim" variable and **n** representing the total number of "dim" variables in this problem.

5.2 Fairness Impact

Apart from evaluating a client's either actual or potential fairness situation, it was also fundamental to measure fairness on the intrapersonal level. This lead to the conceptualization of the term "fairness impact", which, as the name implies, quantifies the impact of an offer in terms of Fairness. It is given by:

$$FI = \frac{FP_{hypothetical} - FP_{current}}{|FP_{current}|} * 100 \tag{5.5}$$

- When FI > 0: in terms of price, the "destiny" bundle is closer to its fairness benchmark than the "source" bundle is to its corresponding "fair" bundle. In other words, the client's fairness situation will improve if the "destiny" bundle is accepted.
- When FI < 0: in terms of price, the "source" bundle is closer to its fairness benchmark than the "destiny" bundle is to its corresponding "fair" bundle. So, the client's fairness situation will deteriorate if the "destiny" bundle is accepted.
- When FI = 0: the offer does not affect the client's fairness situation.

Expanding on the example given in section 5.1, where $FP_{current} = -3.0$ and $FP_{hypothetical} = -10$, the resulting Fairness Impact would be calculated as such:

$$FI = \frac{-10 + 3}{|3|} * 100 = -233 \tag{5.6}$$

This means that the client will potentially get an unfairer bundle, considering the fairness benchmarks for both "source" and "destiny" bundles. Specifically, the client's fairness situation will worsen in 233% if they accept the offer.

Chapter 6

Modeling

Within Machine Learning, there are two types of approaches: **supervised learning** and **unsupervised learning**. A supervised problem is defined by the use of labeled datasets to try to predict outcomes for a specific target, i.e., it must be possible to define a target and there must be prior knowledge about it. On the other hand, in unsupervised learning the datasets are unlabeled and instead of predicting the value of the target variable, the goal is to discover hidden patterns in data. There is no need for human intervention, thus "unsupervised".

There are two main subclasses of supervised learning: **classification** and **regression**, distinguished by the type of target. Whereas regression involves a numeric target, classification involves a categorical (often binary) target. Some common types of classification algorithms include linear classifiers, support vector machines, decision trees and random forest. Linear regression, logistic regression and polynomial regression are instead popular in regression.

Unsupervised learning is used for three main tasks: **clustering**, where individuals in a population are grouped together by their similarity; **association rule discovery**, which attempts to find associations between entities based on transactions involving them; **data reduction**, where a large set of data is replaced with a smaller one that contains much of the important information in the larger set.

A crucial part in the early stages of any data mining process is to decide whether the problem at hand will be phrased as supervised or unsupervised, and to properly define the target variable, if supervised.

As it has already been explained, this work intended to evaluate the fairness situation among clients as well to check whether the company's recommendation system promoted honest products. However, a much more computationally complex challenge and the actual basis for the formulation of this work as a Machine Learning problem was to understand whether any other offers could have had been made that would improve the overall fairness situation, either **quantitatively** (as in more clients see their position improve) or **qualitatively** (as in the average im-

provement is much greater). This said, no matter how optimistically fair these new offers are, if they have a low acceptance probability, there will be little interest for the company to use them.

Considering this, the data mining problem in hand was to classify new ”artificially created” offers as either ”accepted” or ”rejected”, based on past marketing actions whose success was already known, thus formulating a **supervised classification task**.

In order to better understand the modeling procedure, this section will describe the algorithm developed, the framework used to validate that algorithm's solution and the development of the predictions dataset.

This section focuses on explaining both datasets in greater detail, as well as the modeling procedure that was conducted along the way.

6.1 Model Selection

Besides tasks like data cleaning, data reduction and feature engineering, described in Chapter 4, another crucial step of the machine learning process is **model selection**. This technique consists of selecting one among the many candidate models for a predictive problem after the individual models have been evaluated on the defined criteria.

There are many types of requirements one can think of when performing model selection, starting off with the most obvious one: **model performance**. All models have some predictive error. It can be a result of the statistical noise in the data, the incompleteness of the data sample, or even the model's own limitations. Because of this, there is never a ”perfect” model, so the aim is instead to find a model that appropriately fits the data and yields “good enough” results. Another concern is the entire **data preparation** that some models require in order to best expose the structure of the problem to the algorithms: data filtering, data transformation, feature selection, feature engineering, and more. This turns model selection into a process of selecting among model development pipelines, rather than simply models. Finally, a third concern has to do with the **time that the model takes to train**: given the constant need for modeling adjustments, it is preferable to use a sufficiently skillful model that takes little time to train than to use a model that shows great performance but requires much more training time.

In this work, both **model performance** and **training time** concerns were taken into account, while the **data preparation requirement** was ultimately regarded as model independent, rather than a case-by-case adjustment. The reason behind this lies in the conscious decision to previously clean and transform data in such a way that any model could read it, avoiding going back and forth on data preparation tasks as well as assuring a more appropriate exposure of the problem to the algorithms when comparing their predictive performance.

There are two main classes of techniques to estimate predictive performance and select a model:

- **Probabilistic Measures:** the performance is measured on the training dataset, i.e., via in-sample error. Additionally, the model's complexity is considered before selecting a model.
- **Resampling Methods:** the performance is measured on out-of-sample data, i.e., via out-of-sample error.

When using a **probabilistic measure**, the performance is typically penalized as to counterbalance the fact that the training error is optimistically biased. This is usually accomplished by using algorithm-specific methods that penalize the score based on the complexity of the model. As model complexity penalty is more easily tractable in simpler linear models, probabilistic measures are more appropriate for models like linear regression or logistic regression.

Most of the time, however, probabilistic measures are not available, so **resampling methods** are used instead. The most basic resampling method is the **Hold-Out technique**, which consists of splitting the training dataset into a sub train and a validation sets, fitting a model on the sub train set, and evaluating it on the validation set.

This process may be repeated several times as the basis for a more complex resampling method, such as the **Cross-validation** or **Bootstrap** techniques, but this will naturally require more run time, as it will be later explained.

Considering the large dataset containing bundle offers, and after a very patient K-fold Cross-validation experiment, it was clear that such elaborate methods would be too much time consuming for this stage of the project. Thus, in order to get a simple overview of how well different algorithms performed, the **Hold-Out method** was preferred.

6.1.1 Pre-evaluation

For this work, six different models were considered, based on their wide use in the context of classification, specifically binary, problems. Below is a simple description of their respective working functionality:

- **Logistic Regression**: Logistic Regression is an extension of the Linear Regression model for classification problems and is used to predict the probability of the two possible outcomes by fitting data to a logit function.

- **K-Nearest Neighbors Classifier**: By assuming that similar things exist in close proximity, this algorithm uses a distance function to measure the similarity between observations and find a set of k neighbours for a new observation. The latter is then assigned to the class that the majority of its neighbours belong to.

- **Naive Bayes classifier**: It is a classification technique based on the Bayes Theorem with the simple assumption that the predictors are independent, that is, the presence of a particular

feature is unrelated to the presence of another. It is known to be good for large datasets due to its low computation complexity.

- **Decision Tree Classifier**: A decision tree consecutively partitions the data into subsets, based on the possible values for the most significant attributes ("if" conditions). That way, various branches of variable length are formed and data points are classified according to the conditions of the splits, i.e., to the values of the features.

- **Random Forest Classifier**: Random Forest is a collective of uncorrelated decision trees that performs classification by having each individual tree classify a new observation and then choosing the class with the most votes over all the trees in the forest. It uses bagging and feature randomness when building the individual trees.

- **XGB Classifier**: XGBoost also operates on decision trees, by progressively adding more and more "if" conditions to the decision tree to build a stronger model. In other words, new models are sequentially added to correct the errors made by existing models and build a stronger predictor.

As previously explained, these models were compared in terms of training time and predictive accuracy, which was measured with the **Hold-out technique**. The values observed for both criteria and for each model are registered in the following table:

Model	**Time (min:sec)**	**Accuracy** (%)
LR	00:38	54.16
KNN	00:02	57.86
NB	00:06	52.92
DT	01:17	55.27
RF	06:14	63.92
XGB	05:28	63.79

The first thing to notice is how significantly different the six algorithms turned out to be regarding the time they took to make predictions, especially between both ensemble algorithms (RF & XGB) and the rest: KNN became the fastest model, with a 2 second execution time, while Random Forest showed the worst results, taking more than 6 minutes to run.

In terms of accuracy, there is also a significant difference between ensemble algorithms and base learners, with Naive Bayes having the worst predictive performance (52%) and Random Forest correctly classifying almost 64% of new observations.

This said, in spite of Random Forest's lead in accuracy, the model's runtime requirement could become too much of a hurdle later on, when dealing with larger datasets. On the other hand, **XGBoost** algorithm achieved an almost identical accuracy score in a shorter period of time, which seemed like a good compromise.

At the same time, KNN showed the best performance among basic learners', which was a very decent accuracy score when considering its execution time, so it was also a valid candidate. However, this model is known for struggling when the number of inputs is very large, whereas XGBoost can handle big datasets, which was another important factor. Given this, **XGB** was the chosen algorithm to use in this work and it is described in more detail in section 6.1.2.

6.1.2 Extreme Gradient Boosting Tree

In order to fully understand the Extreme Gradient Boosting Tree (XGBoost) algorithm, one must first grasp the machine learning concepts and algorithms that the model builds upon: decision trees, ensemble learning and gradient boosting.

Decision Tree

The Decision Tree (Quinlan, 1986) is a non-parametric supervised learning method widely used for both classification and regression problems. As the name goes, it uses a tree-like model of decisions, so it has the advantage of visually and explicitly representing the decision making process in a very simple way.

A decision tree is typically drawn upside down with its root at the top. It has several decision nodes, each containing a condition/test regarding a specific input attribute. On each node, the tree splits into branches that correspond to different possible values for that node's attribute. When a branch does not split anymore, it is said that it has reached a leaf, i.e., a decision. In other words, each path from the root to the leaf corresponds to a classification rule, i.e., a sequence of rules that can be used to classify the data.

This growing mechanism is known as the top-down recursive divide-and-conquer strategy and essentially involves three types of decisions:

(i) which features to test for splitting purposes;

(ii) how to test the selected features, i.e., what conditions to use;

(iii) when to stop growing the tree.

Starting with decisions (i) and (ii), they both come down to which criterion should be used for splitting. The most well-known criteria include:

- **Information Gain**: The information gain criterion is based on the decrease in entropy after a dataset is split on an attribute. Entropy is used to estimate the randomness or

difficulty to predict the target variable. The attribute that is chosen to split a trip is the one with the highest decrease in entropy associated, i.e., the one with the highest information gain.

- **Information Gain Ratio**: As the Information Gain criterion, this method chooses the attribute that has the highest Information Gain ratio associated.
- The **Gini Index** measures quantitatively the degree of impurity. This index states that if a population is pure, two objects selected at random must belong to the same class (Gini Index equal to zero). The attribute that is chosen to split a tree is the one that provides the smallest Gini Index, i.e., that provides the largest reduction in impurity.

Decision (iii), however, presents a bigger challenge for the Decision Tree algorithm. One that has to do with the fact that if a decision tree is too small, it might not capture important structural information about the sample space, but if it has grown too much, it can lead to overfitting. Overfitting occurs when the model has become too attuned to the data on which it was trained and loses its ability to generalize results, consequently performing much better on the training set than on the test set.

This problem of finding the optimal tree size is generally solved by using a pruning technique, which cuts sections of the tree that are redundant to classify instances. There are two types of pruning processes:

- **Pre-pruning**: Consists in stopping non-significant branches from being generated, going through each decision node and determining whether the corresponding sub-tree adds enough extra value to the model, based on a pre-specified threshold. If it does, the tree's growing process is resumed. Otherwise, the decision node is turned into a leaf node, prematurely pruning the sub-tree. This method is able to avoid generating overly complex sub-trees so it has the advantage of being faster and more efficient. However, it is also possible that a subsequent partition of the avoided sub-tree was extremely valuable to the model.
- **Post-pruning**: Consists in growing a tree to its full extent and then cutting out sub-trees considered as bad predictors, in a bottom-up fashion. This technique is used when the decision tree has very large or infinite depth and shows overfitting of the model.

Ensemble Methods

With the basic elements of decision trees covered, it should finally be added that, despite the model's popularity, a single tree is usually not strong enough to be used in practice. This leads us to **ensemble models**.

Ensemble methods are a machine learning technique that trains several base models using the same learning algorithm and combines them in order to produce a stronger predictive model (Breiman, 1996; Dietterich, 2000; Freund, Schapire, & Abe, 1999). In the context of the decision tree algorithm, this means that multiple decision trees are combined to achieve better predictive performance compared to the one obtained with a single decision tree.

There are two major ways of combining models, i.e., two very well-known ensemble methods, **Bagging** and **Boosting**. They work the following way:

1. Generate additional data in the training stage by random sampling with replacement from the empirical distribution. This means that some observations might be repeated in each new training set.
2. Learn a classifier for each new sampled set, building N basic learners if N new training sets have been produced.
3. Combine all classifiers for class prediction on test data, either through a weighted average of the N learners' predictions or through a weighted majority vote.

This said, the fundamental difference between the two methods has to do with how they perform steps 2 and 3, specifically.

- Regarding step 2: In the case of **Bagging**, each learner is built independently, i.e., the training stage is parallel. However, **Boosting** builds new learners in a sequential way: at first, each observation is assigned equal attention, but then, each new model will focus more on any prediction error (difference between actual value and the predicted value) caused by the previous model to become a stronger learner.
- Regarding step 3: While in **Bagging** each model receives an equal weight, in **Boosting**, models with better performance have their weight increased.

Additionally, the different operating methods end up resulting in different risks for the dataset that data scientists should be aware of: both methods are good at reducing variance and providing higher stability but only Boosting tries to reduce bias. On the other hand, Bagging may solve the over-fitting problem, while Boosting can increase it.

Gradient Boosting

Considering the principle behind the **Boosting** technique (of constructing each new model in a way that corrects the previous model's prediction errors), the main difference between the available boosting algorithms comes down to **how** they identify and rectify weak learners' errors. This can be exemplified by introducing both **Gradient Boosting** and **AdaBoost**.

In the **AdaBoost** Algorithm, short for Adaptive Boosting, each data point is initially assigned an equal **weight**. However, after computing the first learner's errors on the training set, the algorithm lowers the weights for observations that were well classified and increases the weight for observations that were incorrectly classified. Then, it trains the second learner on the re-weighted training set, computes its errors and adjusts weights in preparation for the upcoming learner's training phase. This procedure is continued until either all errors are minimized or the entire dataset is predicted correctly.

Gradient Boosting, on the other hand, uses a loss function that captures the performance of each learner with the goal of minimizing it. The direction and magnitude to which the loss can be minimized is provided by the function's negative gradient, so, in each iteration, Gradient Boosting adds a learner that follows the gradient (i.e., that can be maximally correlated with the negative gradient), reducing the residual loss.

In other words, instead of adjusting weights of data points, Gradient Boosting constructs new base learners using a gradient descent procedure that minimizes the loss. Training stops once loss reaches an acceptable level or no longer improves on an external validation dataset. Alternatively, a fixed number of base learners can be set.

In general, it is up to the researcher to choose the most appropriate loss function among the many standard loss functions already supported, with the additional possibility of defining their own. This makes the Grading Boosting algorithm highly customizable to any particular data-driven task and easy to experiment with.

Xtreme Gradient Boosting

Now that the concepts of decision tree, ensemble learning and gradient boosting have been explained, one can dive into the algorithm of **Extreme Gradient Boosting** (XGBoost).

XGBoost was developed as a research project at the University of Washington and was first published by Chen and Guestrin (2016)**?** as a novel gradient boosting algorithm. It is an extension to gradient boosted machines (GBM) and specially designed to be **scalable** and **highly accurate**, pushing the limits of computing power for boosted tree algorithms, being built largely for energizing machine learning model **performance** and **computational speed**. Also, as a tree-based algorithm, it can be used when there is a mixture of categorical and numeric features or just numeric features as well.

With XGBoost, trees are built in **parallel preprocessing**, instead of sequentially like GBDT, making it faster. It follows a level-wise **tree pruning** strategy, scanning across gradient values and using these partial sums to evaluate the quality of splits at every possible split in the training set.

This algorithm also includes a variety of regularization techniques that **reduce overfitting**, by setting the algorithm's hyperparameters. Additionally, it can **handle missing values** on its

own: during the training process, the model learns whether missing values should be in the right or left node of the tree.

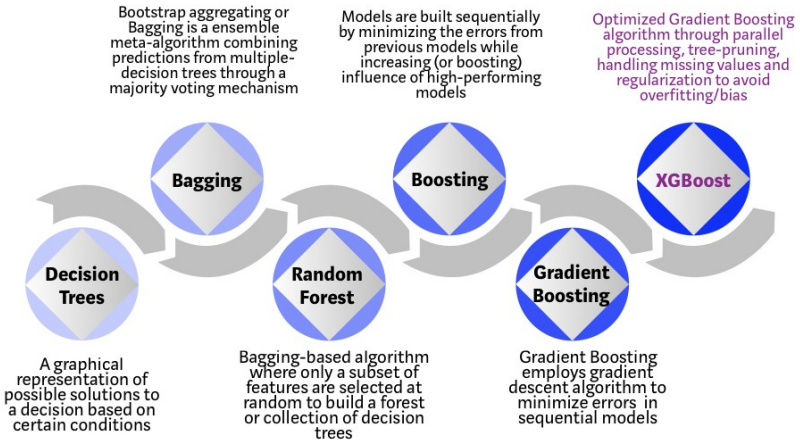


Figure 6.1: Evolution of XGBoost from Decision Trees (Morde, 2019)

As it can be seen on Figure 6.1, this model results from the evolution of other boosting algorithm, taking all of the knowledge from them and implementing several improvements, which were described before. Consequently, XGBoost presents several advantages:

1. it can be used to solve regression, classification, ranking and user-defined prediction problems;
2. it is feasible to train on large datasets, ideally greater than 1000 training samples according to (**?**);
3. it can take advantage of modern multicore computers by executing parallel tasks.

For a more mathematical explanation, here are XGBoost's main Features and Splitting algorithms, according to Chen and Guestrin (2016)**?**:

XGBoost Features:

- **Regularized Learning Objective**:

  For a data set $\mathbf{x}_i, i = 1, \ldots, n$, with $\mathbf{x}_i \in \mathbf{R}^m$ representing m features, the tree ensemble model to predict a target variable y_i is given by

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i). \tag{6.1}$$

In order to train the model to get the best fit between the data $\mathbf{x}_i$ and the output y_i it is necessary to minimize the regularized learning objective function defined by

$$L = \sum_{i=1}^{n} l(\hat{y}_i, y_i) + \sum_{k=1}^{K} \Omega(f_k). \tag{6.2}$$

The training loss in 6.2 measures how predictive the model is with respect to the training data and the common choice for l is the quadratic loss function. The second term corresponds to the regularization term which controls the complexity of the model, helping to smooth the final learnt weights to avoid over-fitting. It penalizes more complex models through both LASSO (L_1 norm) and Ridge(L_2 norm) regularization methods. In practice the regularized learning objective will tend to select a *simple* and *predictive* model; this balance between them is referred as bias-variance tradeoff in machine learning.

- **Gradient tree Boosting**:
  The tree ensemble model cannot be optimized using traditional methods in Euclidean space, it is trained in an additive manner instead. A formula that is used in practice for evaluatiating the split candidates can be found in Chen and Guestrin (2016)**?**.

- **Shrinkage and Column Subsampling**:
  Apart from the regularized objective, two additional techniques are used to prevent over-fitting. One of them is shrinkage introduced by Friedman (2002) **?**; it will scale newly added weights by a factor after each step of tree boosting, thus reduces the influence of each individual tree and leaves space for future trees to improve the model. The second technique is used in Random Forest and consists in the column (representing the feature) sub-sampling. This procedure will prevent over-fitting even more than the traditional row sub-sampling and the column sub-samples usage speeds up computations of the parallel algorithm.

Spliting Algorithms:

- **Exact Greedy Algorithm**:
  The main problem in tree learning is to find the best split. This algorithm called *exact greedy algorithm* enumerates over all the possible splits on all the features. Since it is computationally demanding to enumerate all the possible splits for continuous features, in order to do so on a efficient way the algorithm must first sort the data according to feature values and visit the data in sorted order to evaluate the split candidates.

- **Approximate Algorithm**:
  It is impossible to efficiently enumerate over all the possible splitting points greedily when the data does not fit entirely into memory. To overcome this problem is considered an

Approximate Algorithm which starts by proposing candidate splitting points according to percentiles of feature distribution. Afterwards the algorithm maps the continuous features into buckets split by these candidate points, aggregates the statistics and finds the best solution among proposals based on the aggregated statistics.

- **Weighted Quantile Sketch**:
  The candidate split points purposal is one important step in the approximate algorithm. XGBoost employs the distributed weighted quantile sketch algorithm to effectively find the optimal split points among weighted datasets.

- **Sparsity-aware Split Finding**:
  In many real-world problems, it is quite common for the input x to be sparse. There are several possible reasons for sparsity, among them can be highlighted: 1) the presence of missing values in the data; 2) frequent zero entries in the statistics and 3) artifacts of feature engineering such as one-hot encoding. XGBoost handles all sparsity patterns in a unified way.

6.2 Model Validation

This section covers the first step in the modeling process: training the machine learning algorithm to predict labels from a set of features and validating the trained model's solution on holdout data. Before continuing, however, two key points should be highlighted:

- The set of features that were used to both train and validate the XGBoost algorithm consisted of the variables described in Chapter 3 with some exceptions;
- The target variable represented customer acceptance;

6.2.1 Validation Method

In section 6.1.1, the predictive performance of five different models was compared using the quickest sampling procedure available for chronologically ordered datasets: **Holdout Evaluation**. However, in order to further confirm the suitability of the **XGBoost** algorithm to the problem, a more robust validation method - **Prequential Evaluation** - was carried out: It is appropriate for very large datasets and is able to provide a picture of performance over time, which, once again, would be a central perspective given the "timestamp" variable.

In this approach, the model goes through multiple rounds of training/testing in a chronological order, starting off with train and test datasets built with the oldest observations and progressively updating both datasets with newer data at the end of each round. This update step is done so that the model is always trained on older instances and tested on newer ones.

There are two ways of updating the datasets, each corresponding to a different category of the Prequential Evaluation method: Growing Window and Sliding Window.

Growing Window

In this approach, the test dataset merges with the whole train dataset at the end of each iteration, which means that the train dataset is constantly increasing in size as it keeps incorporating the test observations. This can be seen in Figure 6.2, where *i* represents the number of the iteration/round and *t* represents time.

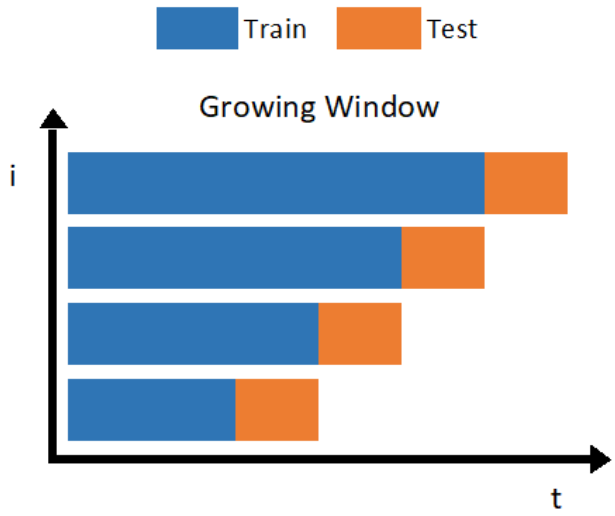


Figure 6.2: Growing Window technique

Sliding Window

In this approach, the test dataset merges with a pre-defined percentage of the train dataset at the end of each iteration, which means that the train set keeps incorporating the observations from the test set while losing a fixed number of older training observations and so, its size does not change. This is also graphically represented in the figure 6.3.

Approach Used

In conclusion, the main difference between these two techniques is that, for training, the **sliding window** uses the same amount of observations each time, while the **growing window** uses more and more observations each time.

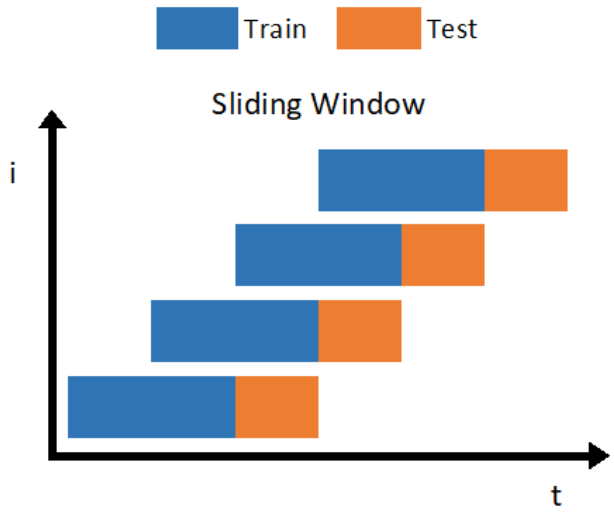


Figure 6.3: Sliding Window technique

This said, in this work, the **growing window technique** was performed through 10 different splits of the dataset and for an individual test set size of 10000 observations, resulting in an initial train set size of 19988 observations, as computed by scikit's TimeSeriesSplit library. It worked the following way:

- Use the first 19988 offers to estimate an XGBoost model;
- Use the model to predict client acceptance for the following 10000 offers, i.e. offers 19989 through 29989;
- Use all observations including the predicted value for offers 19989 through 29989 (i.e. offers 1 through 29989) to train the XGBoost model;
- Use the model to predict client acceptance for the following 10000 offers, i.e. offers 29990 through 39990;
- Use all of the observations including the predicted values to train the XGBoost model;
- Use the model to predict client acceptance for the following 10000 offers, i.e. offers 39991 through 49991;
- and so on...

In the last iteration, the model should be tested on the 10000 most recent observations and trained on the rest of the dataset. However, due to the total number of observations, there are actually 4 extra observations that need to get incorporated into this last test dataset, making a total of 10004 final test data points.

6.2.2 Validation Results

As explained in section 6.2.1, the dataset was split into 10 parts and using the Growing Window technique, each part was used for train and validation of the resulting XGBoost model, leading to different results in the following evaluation metrics: Accuracy, Precision, Recall and F1 Score. This can be seen on table 6.1.

Split	1	2	3	4	5	6	7	8	9	10
Accuracy	81.78	77.28	77.66	76.00	77.74	74.24	78.75	73.56	67.32	70.33
Precision	0.57	0.67	0.66	0.51	0.42	0.54	0.64	0.56	0.52	0.50
Recall	0.20	0.12	0.24	0.03	0.08	0.05	0.01	0.05	0.03	0.22
F1 Score	0.30	0.21	0.36	0.05	0.14	0.09	0.03	0.09	0.06	0.31

Table 6.1: Evaluation of 10 splits of the dataset

It can be seen that **accuracy** is showing different variations across the splits, either increasing or decreasing when compared to the previous one, but generally following a decreasing trend. This is due to the fact that with the **Growing Window** technique the test set used is always different with every iteration, which can lead to very different observations being contained in a specific split, such as outliers. Making balanced splits according to the data ranges could help improving these results. In terms of **precision**, there is also a slight decrease, but it stays around 0.50 to 0.65 range. It can be seen that split iterations 4 to 9 provide the worst results when taking into account all metrics shown. There might also be noise in the data or some information that wasn't properly labeled.

Another validation was performed in terms of Confusion Matrix, Distribution of Predictions and ROC Curve. The following plots on Figure 6.4 and Figure 6.5 show the evaluation using the 1st split and 10th split of data, respectively.

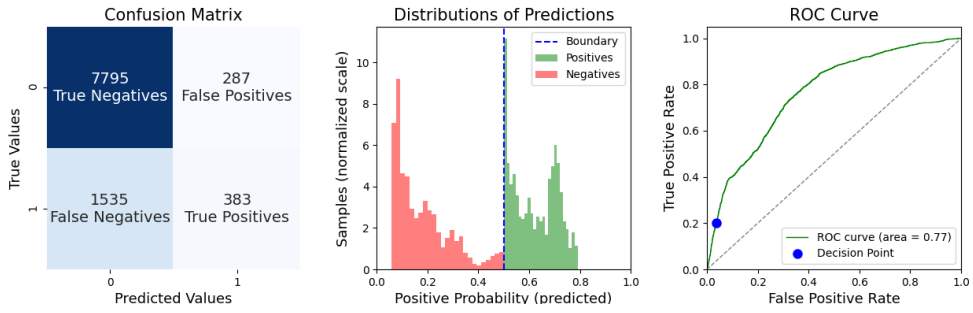


Figure 6.4: Evaluation of XGBoost algorithm with 1st split of data

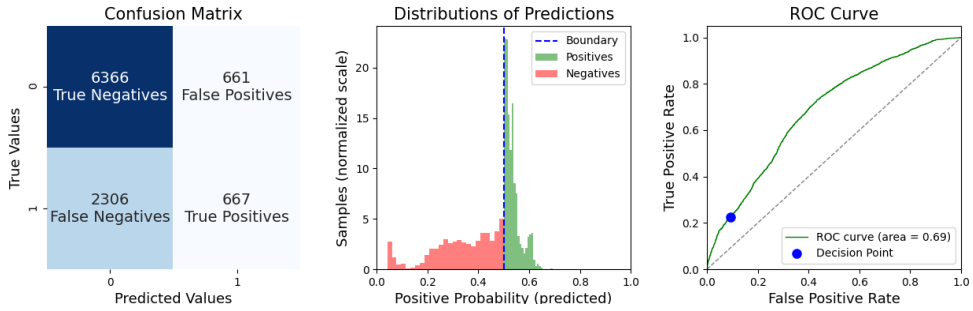


Figure 6.5: Evaluation of XGBoost algorithm with 10th split of data

When comparing the first split of data (first iteration) with the tenth split, we can see that the first split performs rather well with a balanced amount of true and falses in the Confusion Matrix, with balanced distribution of predictions, tending more towards either fully negative (0.0-0.1) or positive (0.5) probability predictions and a ROC of 0.77 which indicates a good characteristic value. However, when analysing the tenth split of the data, the number of false predictions increase and the predictions are mostly positive, meaning that the model is predicting more positive responses from the clients than it is supposed to be.

6.3 Model Evaluation

While the first challenge (evaluating the current fairness situation) was carried out on a dataset similar to the one exported from the company's database, the second challenge (evaluating the fairness of new offers) was done on a different dataset, the reason being precisely the creation of artificial offers.

An artificial offer consists in the combination of a "source" bundle already included in the training dataset with a "recommendation" bundle different from the one included in the real offer, or, to put it simply, it consists in offering a client a bundle that they have not been offered before, thus creating an offer that never existed.

This said, the idea was to create **multiple** artificial offers for each (client, source bundle) pairing, meaning that, while this Artificial Dataset does not structurally change from the one used for training (i.e, the same columns are used), "recommendation" associated columns now describe different bundles and (client, source bundle) pairings are repeated throughout the entire dataset.

In order to select the different "recommendation" bundles for a specific (client, source bundle) pairing, three conditions have to be met:

1. the "recommendation" bundles must correspond to bundles that were, at some point,

offered by the company, allowing for artificial yet realistic offers.

2. the ”recommendation” bundles must show a richer composition than the ”source” bundle’s. This way, the company can contact the customer and suggest adding some new feature to what their current bundle already includes.

3. the ”recommendation” bundles must be in the top 10 most relevant bundles among all bundles that meet the first criterion; this relevance is measured in volume of clients and must be >= 1. In other words, new offers must be grounded on the most popular bundles to then be hopefully aligned with successful campaigns and assure a higher acceptance probability.

This is illustrated in tables 6.2, 6.3 and 6.4:

- Client A currently has bundle 1, whose fairness benchmark corresponds to bundle 17;
- According to company’s history, client A was, at some point, offered bundle 12, whose fairness benchmark corresponds to bundle 41
- Only bundles 34 and 18 show a richer composition than bundle 1’s, as imposed by criterion 2;
- The two bundles belong to 17 and 5 other clients, respectively, meeting the third criterion.

	Pack Source			Pack Neighbour Fair		
Client	**ID**	**Price**	**Dist**	**ID**	**Price**	**Delta Fairness**
A	1	40	0	17	37	3

Table 6.2: Client A’s source pack and corresponding fair pack

Pack Recommendation			Pack Recommendation Fair			
ID	**Price**	**Dist**	**ID**	**Price**	**Delta Fairness**	**Adhesion**
12	42	0.01	41	32	10	1 (True)

Table 6.3: Recommendation pack and corresponding fair pack for client A 6.2

6.4 Work Flow

Now that the whole ML pipeline has been explained in detail, a summarized version is presented in Figure 6.6, consisting in the following steps:

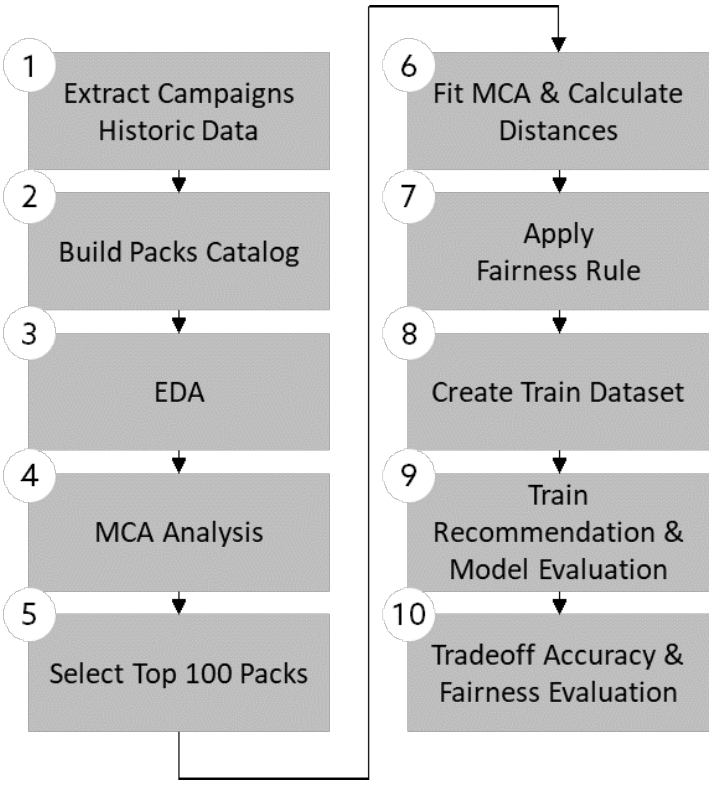


Figure 6.6: Work Flow

Pack Recommendation			Pack Recommendation Fair		
ID	**Price**	**Dist**	**ID**	**Price**	**Delta Fairness**
34	41	0.05	6	40	1
18	46	0.008	10	43	1

Table 6.4: Artificial offer, showing 2 different recommendation packs and their corresponding fair packs for Client A

1. **Extract Campaigns Historic Data**: information about existing customers and bundles and interesting features for the selected fair policy experiment (e.g. the timestamp of the offer) are extracted from the company's remote database and into a dataset;

2. **Build Package Catalog**: a "Catalog" dataset is created to store all unique bundles (bundles are distinguished by their set of features, including price and other revenues);

3. **EDA**: an Exploratory Data Analysis is performed to assess data quality, distributions and types;

4. **MCA**: given the large number of categorical variables typifying bundles, Multiple Correspondence Analysis is used as a dimensionality reduction technique and also to study the association between these variables;

5. **Select Top 100 Bundles**: the 100 most representative (source) bundles are identified based on the % of customer base covered; a new dataset is prepared, pairing each representative bundle with each other bundle of the entire "Catalog" (and corresponding characteristics);

6. **Fit MCA & Calculate Distances**: the MCA solution found on step number 4 is fitted to the new dataset that contains the bundle pairs; euclidean distance is calculated between the two bundles of each pair without taking revenue variables into consideration;

7. **Apply Fairness Rule**: bundle revenues on each pair of bundles are added to the dataset for Fairness analysis; for each key <customer, source bundle, destination bundle>, the corresponding nearest bundles are identified and compared according to a specific Fairness policy; for each bundle, a "fair" bundle is selected;

8. **Create Train Dataset**: the dataset for training is enriched with the MCA row coordinates and the "fair" bundles found for each observation;

9. **Train Recommendation & Evaluate Model**: models are trained, using different algorithms to understand if there is any prevalent technique with outstanding results;

10. **Create Score Dataset & Make Predictions**: this step consists of creating the test dataset - which contains artificial offers that the model will use to predict the adhesion of the client. An artificial offer consists on the combination of a "source" bundle already included in the

training dataset with a new "recommendation" bundle, while keeping the training columns and making sure that two conditions are met:

1. the "recommendation" bundle has to show a richer composition that the "source" bundle;
2. the "recommendation" bundle is select according to its relevance (measured in volume of clients);

11. **Tradeoff Accuracy & Fairness Evaluation**: with the resulting model, a new dataset is generated with different recommendation bundles; the different bundle options are scored based on a Fairness gap; a tradeoff analysis of accuracy/short term revenue and "Fairness pricing" is studied.

Chapter 7

Results

7.1 Fairness Analysis

1 - On average, do the clients improve their fairness positions? Do the artificial offers reflect some improvement/worsening in terms of fairness according to the real situation of the company?

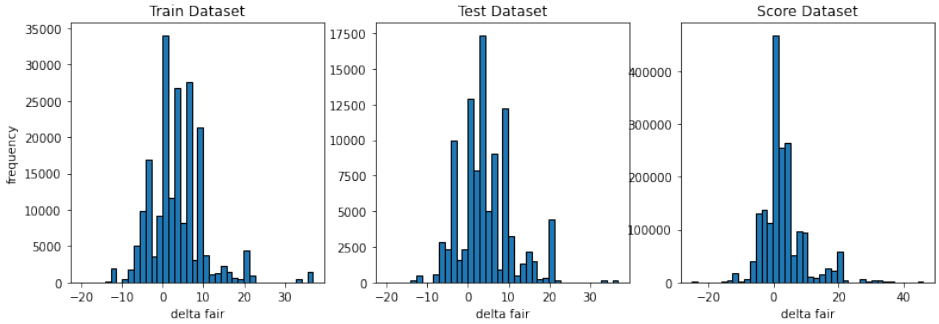


Figure 7.1: Histograms of Fairness Impact values

	Train	Test	Score
count	198876.000	99022.000	1843204.000
mean	0.324	0.891	0.353
std	2.782	2.915	2.663
min	-12.000	-12.000	-12.000
25%	0.000	0.000	0.000
50%	1.167	1.333	0.333
75%	1.500	1.600	1.250
max	15.000	15.000	15.000

Figure 7.2: Summary statistics of Fairness Impact values

Regarding the current company situation, on average, the clients improve their fairness position in 32.4%, while artificial offers reflect a fairness improvement of 35.3%. In other words, the FI is 9% greater among artificial offers than among real ones. In spite of this, however, the same minimum and maximum fluctuations of fairness are shown on both offer sets. Only on absolute terms does the artificial offers dataset stand out in terms of the best improvement of fairness for the client (46 u.m.).

2 - Does the impact on the client's fairness position vary significantly through time?

It should be noted that, given the dataset offers on which the model was validated, on average, the clients improve their fairness position. Since this dataset is identical to the training dataset with the exception that it excludes several initial observations, this might indicate that older offers represented better improvements for the client's fairness position.

7.2 Adhesion Analysis

1. Are the artificial offers well thought? Are they appealing, i.e., is there a lot of adhesion?

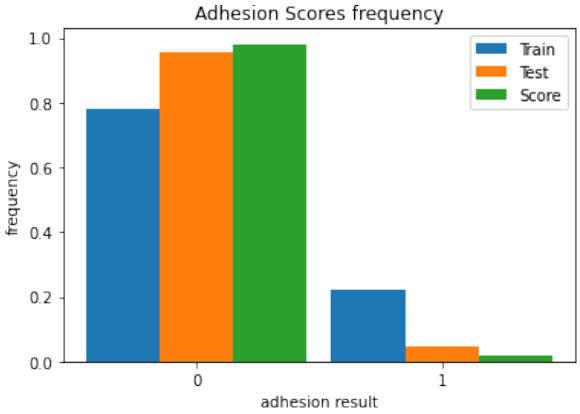


Figure 7.3: Adhesion Scores Frequency

Regarding the real company offers, in most situations, the client rejects the offer that is given to him, and this behavior is also predicted by the model for the artificial offers, in a more unbalanced fashion.

2- Is there any reason for this behavior in terms of average price?

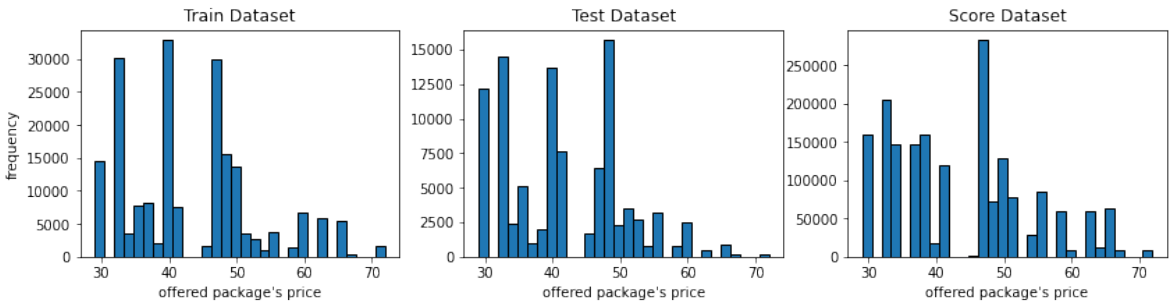


Figure 7.4: Histograms of the offered bundle's price

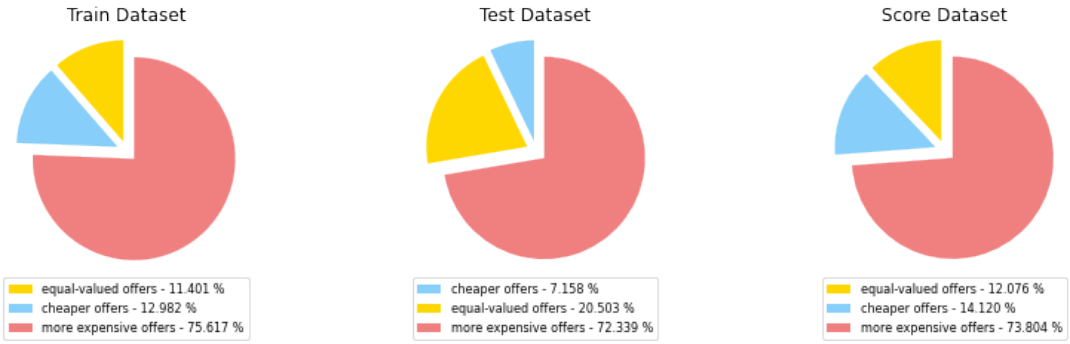


Figure 7.5: Pie charts of the type of price difference between current bundle and offered bundle

In absolute terms, there are no big differences in the price of offers between the ”score” dataset and the training dataset. However, if comparing the prices of the offers with the current price the client is paying, there is a bigger percentage of more expensive offers in the ”training” dataset than on the ”score” dataset, which does not explain the rejection of offers prevailing in the ”score” dataset.

7.3 Bias Analysis

1. Is there any client that always shows the same type of variation in his fairness index? Or one that results in the same outcome?

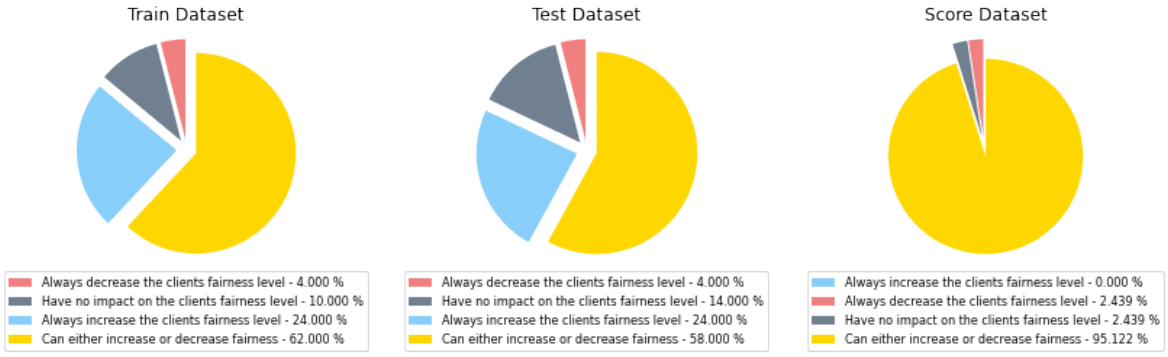


Figure 7.6: Pie charts of types of clients regarding their fairness impact

As expected, the results are a lot more similar between the training and test datasets, showing that a significant percentage of clients have their fairness position varying the same direction, even though the majority of clients get offers that can either improve or worsen their position. The absence of bias is, however, quite evident in artificially created offers.

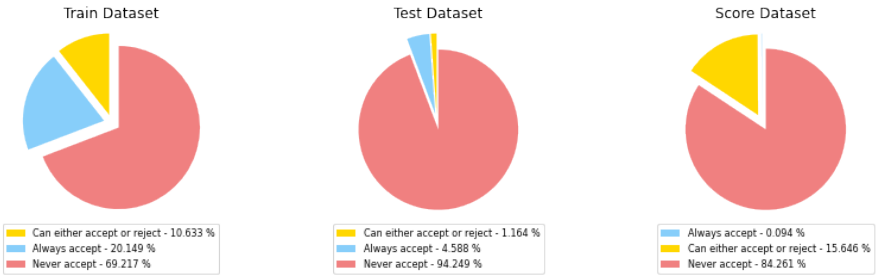


Figure 7.7: Pie charts of types of clients regarding their responses

The results of the adhesion scores distribution that were previously analyzed are very well aligned with what can be observed on these pie charts: in the score dataset there is a special predominance in the refusal of the offer by the client, and at the same time, in comparison with the training dataset, there is a higher percentage of clients that constantly decline the proposed offers, as well an almost zero percentage of clients that always accept the new offers.

2. Is there any bundle offer that always results in the same type of variation in the fairness index? Or that results in the same decision for the client?

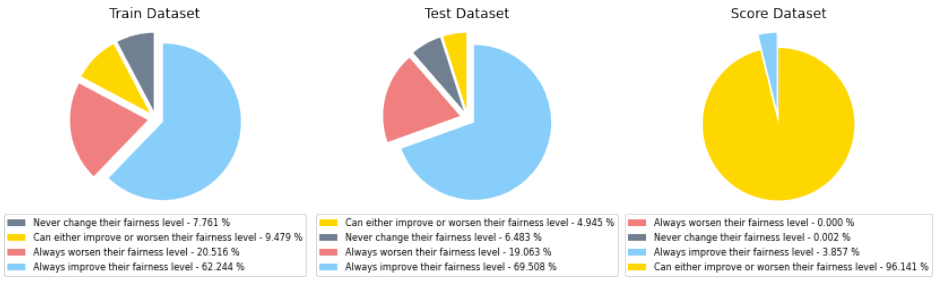


Figure 7.8: Pie charts of types of bundles regarding their clients' fairness impact

Curiously, while the previous analysis to the fairness position indicated slightly optimistic variations to the score dataset, according to these pie charts, in the training dataset there is a stronger presence of offers that constantly affect the fairness position of the client in a positive way. This leads to conclude that there is a possible presence of bias in the client's perception of the offered bundle.

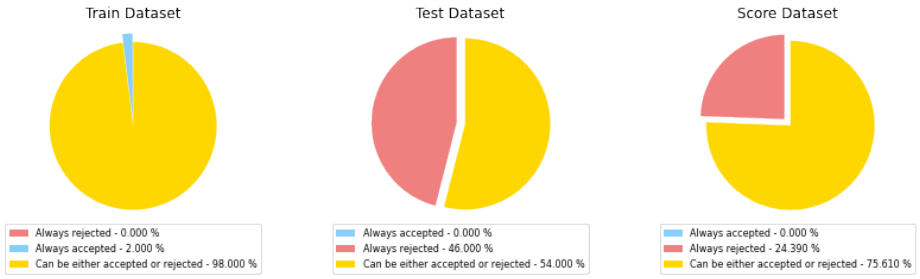


Figure 7.9: Pie charts of types of bundles regarding their clients' response

Generally speaking, in terms of the bundle offers characteristics, there is no evidence of bias in any of the datasets. The artificial bundle offers tend to be more easily declined, which confirms the analysis made prior to the adhesion prediction.

Chapter 8

Final Remarks

Over the past few years, Fairness has emerged as a matter of serious concern within Machine Learning. In particular, policymakers, regulators and advocates have expressed fears about the potentially amplification of social inequities due to discriminatory biases in ML practices. However, research on fair ML is often driven more by the availability of algorithmic methods than by real-world needs, restricting the applicability of that technical research (Holstein, Vaughan, Daumé III, Dudík, & Wallach, 2019).

8.1 Contributions of this thesis

This work presents a distance/similarity based approach to implement and assess Fairness in the context of the Telecommunication sector, responding to real concerns regarding the pricing of telecom and media bundles.

Two similar baseline methodologies were proposed, where Exploratory data analysis has already been successfully carried to assess data distributions and quality and feed downstream decisions. Similar bundles have also been identified through Jaccard Index computation and, more recently, a first MCA solution has been found. These results should now be subject to experimentation for improvement, before applying the remaining steps of the corresponding pipeline. This is because several questions have already been raised:

- MCA's preliminary results are indicating a low percentage of explained inertia; a possible solution might be to group categorical variables with sparse data into classes;
- On the other hand, it might be more appropriate to consider some of the categorical variables as numerical. This requires that MCA is replaced by Multiple Factor Analysis, which deals with mixed data type and might enhance a more compact representation;

- In both approaches, other distance/similarity metrics could be considered, discussing corresponding benefits and trade-offs. For example, Euclidean distance might not be the most adequate metric in high dimensional spaces (Keim, Hinneburg, & Aggarwal, 2001).

8.2 Future Work

Throughout this thesis work, several aspects could not be explored in a deeper level. Taking this into account, here are some of the most relevant to be considered in future work:

- The variety of bundles was restricted by the information on the dataset. Adding a new bundle generator independent from the dataset could help diversify the offers and achieve higher fairness results;
- The artificial offers that resulted in higher acceptance didn't have any real data to confirm the model results. To improve this, a study can be conducted comparing the acceptance rate calculated by the model with a real client acceptance data;
- MCA's preliminary results are indicating a low percentage of explained inertia; a possible solution might be to group categorical variables with sparse data into classes.

Appendix A

Appendix

This chapter is where long sets of more specific data regarding this thesis' work can be found. There are several datasets with a large number of variables, and for the sake of document organization they are made available here with their full description.

A.1 Relevant Catalog Variables Description

Table A.1: Variables description

Variable Name	**Description**
client_id	Client identification.
catalog_tech_cable_source	Whether the client has cable technology.
catalog_tech_copper_source	Whether the client has cable technology, specifically coaxial cable.
catalog_tech_dth_source	Whether the client has satellite technology ("Direct to home").
catalog_tech_ftth_source	Whether the client has cable technology, specifically fiber optic cable ("Fiber to home").
catalog_tech_gsm_source	Whether the client has Router 4G technology.
catalog_tv_box_rental_months_source	Rental period of the TV Box. It can either be 24 months (loyalty period) or 0 months (the customer does not have loyalty program).
catalog_tv_box_rental_offer_source	Whether the client is offered the TV box.
catalog_tv_box_rental_revenue_total_source	TV box revenue: 0€ if the customer has only one box, 6€ if the customer has a second box.
catalog_tv_record_hours_total_source	Recording storage capacity of the TV in hours.

Continue

Table A.1 – Variables description: continuation

Variable Name	Description
catalog_tv_channels_total_source	Sum of all TV channels across the client's contracts.
catalog_tv_channel_sptv_source	Whether the client has the channel "SportTV".
catalog_tv_credits_total_source	Credits available to be spent on TV (e.g. to acquire premium channels).
catalog_tv_credits_remaining_total_source	TV credits remaining.
catalog_if_speed_download_mb_max_source	Maximum of all top download speeds of fixed internet connections for the client.
catalog_if_speed_upload_mb_max_source	Maximum of all top upload speeds of fixed internet connections for the client.
catalog_vm_cards_total_source	Sum of voice mobile (VM) cards across all the client's contracts.
catalog_vm_credits_total_source	Sum of credits available for each individual VM card and across all the client's contracts.
catalog_vm_sms_total_source	Sum of SMS available for each individual VM card and across all the client's contracts.
catalog_vm_min_total_source	Sum of minutes available for each individual VM card and across all the client's contracts.
catalog_vm_data_mb_total_source	Sum of all data traffic for upload and download of VM cards.
catalog_vm_data_mb_max_source	Maximum data traffic for upload and download of VM cards.
catalog_vm_revenue_total_source	Sum of the revenue of all VM cards across all the client's contracts.
catalog_vm_revenue_max_source	Maximum revenue of all VM cards across all the client's contracts.
catalog_im_cards_total_source	Sum of mobile internet (IM) cards across all the client's contracts.
catalog_im_data_mb_total_source	Sum of all data traffic for upload and download of IM cards.
catalog_im_data_mb_max_source	Maximum data traffic for upload and download of IM cards.

Bibliography

Abdi, H., & Valentin, D. (2007). Multiple Correspondence Analysis. *Encyclopedia of Measurement and Statistics*.

Adams, J. (1965). Inequity In Social Exchange. In L. Berkowitz (Ed.), (Vol. 2, pp. 267 – 299). Academic Press. (ISSN: 0065-2601) doi: 10.1016/S0065-2601(08)60108-2

Adams, W., & Yellen, J. (1976). Commodity Bundling and the Burden of Monopoly. *The Quarterly Journal of Economics*, *90*(3), 475–498. doi: 10.2307/1886045

Alesina, A., Lotti, F., & Mistrulli, P. (2013). Do Women Pay More for Credit? Evidence From Italy. *Journal of the European Economic Association*, *11*(suppl_1), 45–66. doi: 10.1111/j.1542-4774.2012.01100.x

Allen, J., & West, D. (2018). *How artificial intelligence is transforming the world.* Retrieved 2021-01-16, from https://www.brookings.edu/research/how-artificial-intelligence-is-transforming-the-world/

Angwin, J., Larson, J., Mattu, S., & Kirchner, L. (2016). *Machine Bias.* Retrieved 2021-01-16, from https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing?token=KCQtlys2Jot94pBZSAGZgQHpS8pwmTyy

Austin, W., McGinn, N., & Susmilch, C. (1980). Internal standards revisited: Effects of social comparisons and expectancies on judgments of fairness and satisfaction. *Journal of Experimental Social Psychology*, *16*(5), 426 – 441. doi: 10.1016/0022-1031(80)90049-9

Bai, J., Zhou, C., Song, J., Qu, X., An, W., Li, Z., & Gao, J. (2019). Personalized Bundle List Recommendation. In *The World Wide Web Conference* (pp. 60–71). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3308558.3313568

Barone, M., & Roy, T. (2010). The effect of deal exclusivity on consumer response to targeted price promotions: A social identification perspective. *Journal of Consumer Psychology*, *20*(1), 78–89. doi: 10.1016/j.jcps.2009.10.002

Beutel, A., Chen, J., Zhao, Z., & Chi, E. (2017). Data Decisions and Theoretical Implications when Adversarially Learning Fair Representations. *arXiv:1707.00075*.

Binns, R. (2020). On the apparent conflict between individual and group fairness. *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. doi: 10.1145/3351095.3372864

Bitran, G., & Caldentey, R. (2003). An Overview of Pricing Models for Revenue Management. *Manufacturing & Service Operations Management*, *5*(3), 203–229. doi: 10.1287/msom.5.3.203.16031

Borgesius, F., & Poort, J. (2017). Online Price Discrimination and EU Data Privacy Law. *Journal*

of Consumer Policy, *40*(3), 347–366. doi: 10.1007/s10603-017-9354-z

Bouchet, A., Troilo, M., & Walkup, B. (2016). Dynamic pricing usage in sports for revenue management. *Managerial Finance*, *42*(9), 913–921. doi: 10.1108/MF-01-2016-0017

Calmon, F., Wei, D., Ramamurthy, K., & Varshney, K. (2017). Optimized Data Pre-Processing for Discrimination Prevention. *arXiv:1704.03354*.

Chan-Olmsted, S., & Guo, M. (2011). Strategic Bundling of Telecommunications Services: Triple-Play Strategies in The Cable TV and Telephone Industries. *Journal of Media Business Studies*, *8*(2), 63–81. doi: 10.1080/16522354.2011.11073523

Charles, K., Hurst, E., & Stephens, M. (2008). Rates for Vehicle Loans: Race and Loan Source. *American Economic Review*, *98*(2), 315–20. doi: 10.1257/aer.98.2.315

Chen, I., Johansson, F., & Sontag, D. (2018). Why Is My Classifier Discriminatory? *arXiv:1805.12002*. Retrieved from http://arxiv.org/abs/1805.12002

Chen, N., & Gallego, G. (2019). Welfare Analysis of Dynamic Pricing. *Management Science*, *65*(1), 139–151. doi: 10.1287/mnsc.2017.2943

Chen, Q., Jasin, S., & Duenyas, I. (2016). Real-Time Dynamic Pricing with Minimal and Flexible Price Adjustment. *Management Science*, *62*(8), 2437–2455. doi: 10.1287/mnsc.2015.2238

Choudhary, V., Ghose, A., Mukhopadhyay, T., & Rajan, U. (2005). Personalized Pricing and Quality Differentiation. *Management Science*, *51*(7), 1120–1130. doi: 10.1287/mnsc.1050.0383

Chouldechova, A. (2017). Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *arXiv:1703.00056*.

Cohen, M. C. (2018). Big Data and Service Operations. *Production and Operations Management*, *27*(9), 1709–1723. doi: https://doi.org/10.1111/poms.12832

Dastin, J. (2018). Amazon scraps secret AI recruiting tool that showed bias against women. *Reuters*. Retrieved 2021-01-16, from https://www.reuters.com/article/us-amazon-com-jobs-automation-insight-idUSKCN1MK08G

Ebrah, K., & Elnasir, S. (2019). Churn prediction using machine learning and recommendations plans for telecoms. *Journal of Computer and Communications*, *07*(11), 33–53. doi: 10.4236/jcc.2019.711003

Ezrachi, A., & Stucke, M. (2016). The Rise of Behavioural Discrimination. *SSRN Electronic Journal*. doi: 10.2139/ssrn.2830206

Faruqui, A., & Sergici, S. (2010). Household response to dynamic pricing of electricity: a survey of 15 experiments. *Journal of Regulatory Economics*, *38*(2), 193–225. doi: 10.1007/s11149-010-9127-y

Fisher, M., Gallino, S., & Li, J. (2018). Competition-Based Dynamic Pricing in Online Retailing: A Methodology Validated with Field Experiments. *Management Science*, *64*(6), 2496–2514. Retrieved from https://doi.org/10.1287/mnsc.2017.2753 (_eprint: https://doi.org/10.1287/mnsc.2017.2753) doi: 10.1287/mnsc.2017.2753

Fjeld, J., Achten, N., Hilligoss, H., Nagy, A., & Srikumar, M. (2020). Principled Artificial Intelligence: Mapping Consensus in Ethical and Rights-Based Approaches to Principles for AI. *SSRN Electronic Journal*. doi: 10.2139/ssrn.3518482

Garfinkel, R., Gopal, R., Tripathi, A., & Yin, F. (2006). Design of a shopbot and recommender system for bundle purchases. *Decision Support Systems*, *42*(3), 1974–1986. doi: 10.1016/

j.dss.2006.05.005

Gebru, T., Morgenstern, J., Vecchione, B., Vaughan, J., Wallach, H., Daumé III, H., & Crawford, K. (2020). Datasheets for Datasets. *arXiv:1803.09010*. (arXiv: 1803.09010)

Goodman, B., & Flaxman, S. (2017). European Union Regulations on Algorithmic Decision-Making and a "Right to Explanation". *AI Magazine*, *38*(3), 50–57. doi: 10.1609/aimag .v38i3.2741

Greenacre, M. (1993). *Correspondence analysis in practice*. London: Academic Press.

Greenacre, M. (2007). *Correspondence analysis in practice* (3rd ed.). Chapman and Hall CRC.

Hardt, M., Price, E., & Srebro, N. (2016). Equality of Opportunity in Supervised Learning. *arXiv:1610.02413*, 22.

Holstein, K., Vaughan, J., Daumé III, H., Dudík, M., & Wallach, H. (2019). Improving fairness in machine learning systems: What do industry practitioners need? *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. doi: 10.1145/3290605.3300830

Homans, G. (1961). *Social behavior: Its elementary forms*. Oxford, England: Harcourt, Brace.

Izaret, J.-M. (2022, May). *Solving the paradox of fair prices*. BCG Global. Retrieved from https://www.bcg.com/publications/2022/considering-pricing-variation-to-help-solve-the-paradox-of-fair-prices

Kahneman, D., Knetsch, J., & Thaler, R. (1986). Fairness as a Constraint on Profit Seeking: Entitlements in the Market. *The American Economic Review*, *76*(4), 728–741. Retrieved from http://www.jstor.org/stable/1806070

Kamiran, F., & Calders, T. (2012). Data preprocessing techniques for classification without discrimination. *Knowledge and Information Systems*, *33*(1), 1–33. Retrieved 2021-01-16, from http://link.springer.com/10.1007/s10115-011-0463-8 doi: 10.1007/s10115-011-0463-8

Kamishima, T., Akaho, S., Asoh, H., & Sakuma, J. (2012). Fairness-Aware Classifier with Prejudice Remover Regularizer. In D. Hutchison et al. (Eds.), *Machine Learning and Knowledge Discovery in Databases* (Vol. 7524, pp. 35–50). Berlin, Heidelberg: Springer Berlin Heidelberg.

Kearns, M., Neel, S., Roth, A., & Wu, Z. (2018). Preventing Fairness Gerrymandering: Auditing and Learning for Subgroup Fairness. *arXiv:1711.05144*.

Keim, D. A., Hinneburg, A., & Aggarwal, C. C. (2001). On the Surprising Behavior of Distance Metrics in High Dimensional Space. In G. Goos, J. Hartmanis, J. van Leeuwen, J. Van den Bussche, & V. Vianu (Eds.), *Database Theory — ICDT 2001* (Vol. 1973, pp. 420–434). Berlin, Heidelberg: Springer Berlin Heidelberg. doi: 10.1007/3-540-44503-X_27

Keskin, N., & Zeevi, A. (2014). Dynamic Pricing with an Unknown Demand Model: Asymptotically Optimal Semi-Myopic Policies. *Operations Research*, *62*(5), 1142–1167. doi: 10.1287/opre.2014.1294

Kleinberg, J., Mullainathan, S., & Raghavan, M. (2016). Inherent Trade-Offs in the Fair Determination of Risk Scores. *arXiv:1609.05807*.

Kuhn, M., & Johnson, K. (2019). *Feature Engineering and Selection: A Practical Approach for Predictive Models*. Chapman & Hall/CRC.

Major, B. (1994). From social inequality to personal entitlement: The role of social comparisons, legitimacy appraisals, and group membership. In *Advances in experimental social psychology, Vol. 26*. (pp. 293–355). San Diego, CA, US: Academic Press. doi: 10.1016/S0065-2601(08)

60156-2

Major, B., & Testa, M. (1989). Social comparison processes and judgments of entitlement and satisfaction. *Journal of Experimental Social Psychology*, *25*(2), 101–120. doi: 10.1016/0022-1031(89)90007-3

Malc, D., Mumel, D., & Pisnik, A. (2016). Exploring price fairness perceptions and their influence on consumer behavior. *Journal of Business Research*, *69*(9), 3693–3697. doi: 10.1016/j.jbusres.2016.03.031

Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., & Galstyan, A. (2019). A Survey on Bias and Fairness in Machine Learning. *arXiv:1908.09635*.

Mislevy, R. J., Little, R. J., & Rubin, D. B. (1991). Statistical analysis with missing data. *Journal of Educational Statistics*, *16*(2), 150. doi: 10.2307/1165119

Morde, V. (2019, April). *XGBoost Algorithm: Long May She Reign!* Retrieved 2021-09-07, from https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d

Pandey, A., & Caliskan, A. (2020). Iterative Effect-Size Bias in Ridehailing: Measuring Social Bias in Dynamic Pricing of 100 Million Rides. *arXiv:2006.04599*.

Pigou, A. (1932). *The Economics of Welfare*. London: Macmillan and Co.

Rautio, T., Anttila, M., & Tuominen, M. (2007). Bundling of information goods: a value driver for new mobile TV services. *International Journal of Revenue Management*, *1*, 45–64. doi: 10.1504/IJRM.2007.011193

Räz, T. (2021). Group fairness. *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*. doi: 10.1145/3442188.3445876

Srivastava, M., Heidari, H., & Krause, A. (2019). Mathematical Notions vs. Human Perception of Fairness: A Descriptive Approach to Fairness for Machine Learning. *arXiv:1902.04783*.

Stremersch, S., & Tellis, G. J. (2002). Strategic Bundling of Products and Prices: A New Synthesis for Marketing. *Journal of Marketing*, *66*(1), 55–72. (Publisher: SAGE Publications Inc) doi: 10.1509/jmkg.66.1.55.18455

Suresh, H., & Guttag, J. (2021). A framework for understanding sources of harm throughout the machine learning life cycle. *Equity and Access in Algorithms, Mechanisms, and Optimization*. doi: 10.1145/3465416.3483305

Tabak, J. (2014). *Geometry: The Language of Space and Form*. Infobase Publishing.

Toreini, E., Aitken, M., Coopamootoo, K., Elliott, K., Zelaya, C., & van Moorsel, A. (2019). The relationship between trust in AI and trustworthy machine learning technologies. *arXiv:1912.00782*.

Turow, J., King, J., Hoofnagle, C., Bleakley, A., & Hennessy, M. (2009). Americans Reject Tailored Advertising and Three Activities That Enable It. *SSRN Electronic Journal*. doi: 10.2139/ssrn.1478214

Valentino-DeVries, J., Singer-Vine, J., & Soltani, A. (2012). Websites Vary Prices, Deals Based on Users' Information. *Wall Street Journal*. Retrieved 2021-01-16, from https://online.wsj.com/article/SB10001424127887323777204578189391813881534.html

van Ryzin, G., & McGill, J. (2000). Revenue Management Without Forecasting or Optimization: An Adaptive Algorithm for Determining Airline Seat Protection Levels. *Management Science*, *46*(6), 760–775. doi: 10.1287/mnsc.46.6.760.11936

Verma, S., & Rubin, J. (2018). Fairness definitions explained. In *Proceedings of the International Workshop on Software Fairness* (pp. 1–7). Gothenburg Sweden: ACM. doi: 10.1145/3194770.3194776

Wood, J. (1989). Theory and research concerning social comparisons of personal attributes. *Psychological Bulletin*, *106*(2), 231–248. doi: 10.1037/0033-2909.106.2.231

Wu, S., Yau, W.-C., Ong, T.-S., & Chong, S.-C. (2021). Integrated churn prediction and customer segmentation framework for telco business. *IEEE Access*, *9*, 62118–62136. doi: 10.1109/access.2021.3073776

Xia, L., Monroe, K., & Cox, J. (2004). The Price is Unfair! A Conceptual Framework of Price Fairness Perceptions. *Journal of Marketing*, *68*(4), 1–15. doi: 10.1509/jmkg.68.4.1.42733

Zhang, J. (2011). The Perils of Behavior-Based Personalization. *Marketing Science*, *30*(1), 170–186. doi: 10.1287/mksc.1100.0607