Erasmus University Rotterdam
Erasmus School of Economics
Master Thesis Data Science & Marketing Analytics

# Introducing Rule-Based methods for conversion prediction and attribution in digital advertising.

Thijmen van der Put (433871)

| | |
|---|---|
| Supervisor: | Akyüz, M.H. |
| Second assessor: | Almeida Camacho, N.M. |
| Final version: | October 27, 2023 |

**Abstract**

In the digital age, businesses continuously work on optimizing their (digital) marketing strategy. Potential customers can online be interacted with in various manners, in an attempt by businesses to generate conversions. To do this efficiently, advertisers require meaningful insights on what advertisements and their features are associated with these conversions, which can be distilled from internet users' browser behaviour. In an attempt to credit advertisement for their contribution towards a conversion, the literature has introduced various multi-touch attribution (MTA) models. Whereas the first of these were relatively simple and straightforward to interpret, considerably more complex MTA models have been introduced in the form of Deep Learning based models. These so-called black boxes are infamous for their lack of transparency, as their internal mechanisms are not revealed, making these models not intrinsically interpretable. This thesis introduces rule-based methods to the current literature on MTA, a branch of models which is considered as one of the most intrinsically interpretable. Rule-based methods produce decision rules in order to classify data instance. The interpretability of these decision rules stems from their if-then structure, which resembles natural language. Various rule-based methods are introduced for conversion prediction. Additionally, the produced rules will be utilized to extract meaningful insights on advertisements and features associated with (non-)conversions. These models are compared to one-another and to traditional models in terms of predictive performance and interpretability. It follows that these models allow to easily induce what features are associated with (non-)converting journeys, while their predictive performance is better than that of the Bagged Logistic Regression and comparable to that of the Random Forest and Gradient Boosted trees.

# Contents

# 1 Introduction

The popularity and spending on online advertising is increasing rapidly: global spending on digital advertisments is expected to grow from $232 billion in 2017 to $427 billion in 2022 (McNair, 2018). Specifically for the Netherlands, spending on digital advertising increased by 13% in 2022, resulting in 3.5 billion euros being spent (Deloitte and VIA, 2022). Consequently, digital advertising has become a central point in businesses' advertising models (Raman et al., 2012). Businesses have numerous ways through which they can interact with customers and prospects, i.e., display advertisements, e-mails, social-media and search engines. These moments in which an interaction takes place between a business and a member of the audience, is typically referred to as a *touchpoint* or *impression*.

A major advantage of digital advertising as opposed to traditional advertising, is that the advertisements can be tailored specifically to the characteristics and preferences of individuals. Moreover, digital advertising allows businesses to track the behavior of individuals across the digital space, such as whether a digital advertisement was clicked or how long an individual views an advertisement (Chaffey and Ellis-Chadwick, 2019). In particular, businesses track the sequences of touchpoints individuals come across and whether these sequences result in a desirable action: a conversion (i.e., sale or subscription). These sequences of touchpoints, which either result in a conversion or not, are referred to as *customer journeys*.

Analyzing these customer journeys is of high importance to businesses, as it may provide insights into which touchpoints or characteristics of touchpoints were influential in driving eventual conversions. Specifically, measuring the influence of touchpoints and subsequently allocating credit to the touchpoints utilized by a firm, is referred to as the Multi-Touch Attribution (MTA) problem (Ren et al., 2018). It is of high importance to businesses to allocate credits of conversions in the right manner, for the particular reason that an adequate and reliable attribution methodology determines the influence of advertisements on conversions, giving insights to businesses on which advertisements they should focus with the aim to maximize conversion and minimize costs (Dalessandro et al., 2012). Originally, businesses applied heuristic-based methods to tackle the MTA problem. One of these heuristic-based methods is the Last-Touch Atribution (LTA) method, which credits a conversion fully to the last touchpoint in a customer journey. Although these heuristic-based methodologies are straightforward to implement, they may lead to businesses drawing the wrong conclusions as they are not data-driven.

For that reason, various data-driven and machine-learning based MTA models have been proposed in the academic literature (Abhishek et al., 2012); (de Haan et al., 2016); (Anderl et al., 2016); (Kannan and Li, 2017). Among the first to introduce a data-driven MTA model, were Shao and Li (2011) who utilized a Bagged Logistic regression (BLR) and also proposed a

simple probabilistic approach. Dalessandro et al. (2012) extended on the probabilistic approach, and have shown that it is equivalent to the Shapley Value solution from Game Theory (Shapley, 1953). This approach allows for the distribution of credit among players in a coalition game, in line with their contribution. Deep learning approaches, which utilize neural networks, have also been leveraged in the MTA context (Ren et al., 2018); (Du et al., 2019); (Yao et al., 2021). Particularly, Du et al. (2019) and Yao et al. (2021) first use a deep learning approach to estimate converting customer journeys and subsequently apply Shapley Value to perform credit distribution among the touchpoints.

Although these deep learning approaches and data-driven methodologies have shown the be successful in conversion prediction and subsequent credit allocation, most of these models are complex and not simple to interpret. Specifically, the deep learning methods typically consist of multiple layers and complex mechanisms, making these models intrinsically not interpretable; it's no coincidence these type of models are typically referred to as black boxes.

Notably, Dalessandro et al. (2012) have formulated properties of an adequate attribution system. In particular, these authors argue that an attribution model should be accepted by all parties, based on statistical merit and intuitive understanding of the model. By this statement, black boxes are not considered as adequate models.

An intrinsically interpretable type of models, are decision rules. These decision rules have a common if-else structure. As an illustrative example: **If** test score is 5.5, **then** student passed exam. As these decision rules resemble natural language, they are considered one of the most inherently interpretable models, provided that the number of conditions in each individual rule and the total number of rules are limited. Arguably, this type of models could be advantageous in the conversion prediction and attribution context, because these models could help marketers or businesses in general in easily comprehending what features are associated with (non)-conversions. Importantly, these decision rules generally come in one of two forms: A decision list or a decision set. A decision list enforces an hierarchical structure on the rules, while the rules in a decision set can be considered in any order. In recent years, rule-based methods have been of large interest in the literature and various models have been introduced which approximate the predictive performance of black-boxes (Lakkaraju et al., 2016); (Lumadjeng et al., 2023); (Proença and van Leeuwen, 2020); (Friedman and Popescu, 2008).

Although these models are great in terms of interpretability and their predictive performance approximates that of black boxes in some contexts, these methods have not been leveraged yet in the field of conversion prediction and credit attribution. Leveraging rule-based methods in the MTA context can help businesses to easily infer which features are associated with (non-)converting customer journeys, as the produced decision rules by these rule-based methods are supposed to be easily interpretable. Thus, the produced decision rules should be able to

provide businesses with insights to enhance the touchpoints the audience comes across during their customer journeys. Especially for practitioners with no or limited knowledge on machine learning, these intuitive decision rules generated by the rule-based methods may be a welcome addition in the MTA context.

Therefore, this thesis aims to fill this gap within the literature by introducing these methods in this field. In order to accomplish this, this thesis aims to answer the following research questions: *I): How do rule-based methods leveraged in the conversion prediction and attribution context differ from traditional methods in terms of predictive performance?*, *II) How do rule-based methods compare to the traditional methods, in terms of interpreting touchpoint or feature importance?*

The rest of this thesis is structured in the following manner: Section two provides an overview on the relevant literature. The Criteo set which will be utilized for this thesis is discussed in section three. Section four provides an overview of traditional methods, rule based methods and evaluation procedure. Section five presents the results. Finally, conclusions and final remarks regarding this thesis are provided.

# 2 Literature Review

## 2.1 Digital Marketing

Ever since the inception of the internet, the marketing landscape has transformed drastically. The rise of the internet has subsequently resulted in the development of new communication media for reaching and engaging with audiences, known as digital media (Chaffey and Ellis-Chadwick, 2019). Specifically, digital media concern media which transmit digitized information to engage with audiences, typically through a screen and/or speaker (Feldman, 1996). Unsurprisingly, firms swiftly adapted these new digital media, beside traditional media, within their marketing strategies (Feldman, 1996). Moreover, constant innovations ever since the nineties within the digital landscape (e.g., E-commerce, search-engines and smartphones) have resulted in a dynamic playing field in which businesses are required to stay aware of new developments and assess which developments are most relevant for potentially gaining an advantage by introducing them (Chaffey and Ellis-Chadwick, 2019). Such innovations include new ways to communicate with customers and prospects, which are of great importance to marketers and businesses' marketing strategies.

Both digital media and digital innovations are an important aspect of digital marketing (Chaffey and Ellis-Chadwick, 2019). The definition of marketing has shifted from referring to the practise of marketing products using digital channels and media, to an umbrella term which describes the procedure of employing digital technologies to acquire and retain customers, determine customers' preferences, promote brands and increase sales. (Kannan and Li, 2017) These innovations followed one another in rapid succession and the growth in digital marketing has been primarily due to these rapid advancements (Madhu and Deepak, 2018). Nowadays, digital marketing involves many different and dynamic types of interaction possibilities with the audience. Chaffey and Ellis-Chadwick (2019) describe that in current times, it's essential for businesses to utilize the 'Five Ds' of interactions within the digital marketing landscape:

- Digital devices: the possibility to interact with the audience through different types of devices (e.g., smartphones, tablets, TVs, etc.).
- Digital platforms: A large portion of the interactions on these devices take place through browsers and applications of the major platforms or online services (e.g., Google's Android, Apple's IOS, Twitter, Facebook).
- Digital media: Various digital communication channels exist through which businesses can interact with their audiences.
- Digital data: Digital marketing allows businesses to collect data about the interactions with their audiences, their audiences' profiles and their preferences.

- Digital technology: The technologies utilized for developing new methods within the digital marketing scene.

The fourth point above, digital data, describes how businesses are able to collect data about the interactions with their audiences, among other types of data. Concretely, businesses can track who the business interacted with, when this interaction took place and through what channel the interaction was facilitated (Chaffey and Ellis-Chadwick, 2019). These moments of interaction between audiences and businesses, are typically referred to as touchpoints and numerous digital marketing channels and platforms are utilized to facilitate these interactions (Lemon and Verhoef, 2016). Some of the most common types of digital marketing channels are: Organic- and paid-search, referrals, affiliates, display banners and e-mail (de Haan et al., 2016); (Li and Kannan, 2014). The different digital marketing channels are often categorized in the literature, based on common characteristics such as: The degree of content integration (de Haan et al., 2016) and initiator of the interaction (de Haan et al., 2016); (Anderl et al., 2016). Interaction through a channel is either initiated by the firm or a customer or prospect. The former type of channels, are often called firm initiated channels (FICs), while the later are referred to as customer initiated channels (CIGs) (Anderl et al., 2016).

An individual can encounter numerous different channels at each touchpoint. A sequence of such touchpoints, may eventually result in a conversion (i.e., sale or subscription). Such a sequence of touchpoints is usually called a customer journey in the academic literature (Lemon and Verhoef, 2016). As previously mentioned, digital marketing has allowed businesses to collect data about the interactions with their audience. Consequently, this allows businesses to analyse the customer journeys that their audience experience (Chaffey and Ellis-Chadwick, 2019).

## 2.2 Multi-touch Attribution

In table 1, an overview is provided of the different types of MTA models that will be discussed in this section and whether they have certain properties. These properties concern whether these models are data-driven, are interpretable and whether they incorporate the sequential nature of customer journeys. These properties will also be discussed in the following sections. A type of MTA models has a certain property, if a tick is shown in that property's column.

MTA modelling is the practice of assigning credit to the employed digital advertisement channels for driving a consumer to perform a desirable action, such as taking a subscription or making a purchase, which is typically referred to as a conversion in the literature (Shao and Li, 2011). Moreover, MTA modelling can also be used to determine the efficiency and effectiveness of digital channels (Chaffey and Ellis-Chadwick, 2019). Businesses can subsequently use the results of the attribution modelling to allocate the digital marketing budget to the different digital channels accordingly.

Table 1: An overview of the types of MTA models in the literature.

| Type | Paper(s) | Data-driven | Interpretability | Sequential nature |
|---|---|---|---|---|
| Heuristics | - | | ✓ | |
| Shapley value approximations | Shao and Li (2011) & Dalessandro et al. (2012) | ✓ | ✓ | |
| Generalized linear models | Shao and Li (2011) | ✓ | ✓ | |
| Markov models | Abhishek et al. (2012) & Anderl et al. (2016) | ✓ | ✓ | ✓ |
| LSTM-RNN based models | Ren et al. (2018), Du et al. (2019), Yang et al. (2020) & Yao et al. (2021). | ✓ | | ✓ |
| Rule-based methods | This study | ✓ | ✓ | ✓ |

Traditionally, the task of crediting marketing channels for conversion was done by means of applying simple heuristic-based methods. One such heuristic-based method, is the last-touch attribution (LTA) method, which attributes the conversion entirely to the last touchpoint in the customer journey. Similarly, first-touch attribution (FTA), assigns the credit of a conversion in a customer journey to the first touchpoint in the sequence (Ren et al., 2018). Unmistakably, these two attribution methods completely ignore the other touchpoints. Research has indicated that such approaches can lead to incorrect conclusions, as these approaches completely disregard the effect of the other touchpoints and possible interaction effects between touchpoints (Shao and Li, 2011); (Abhishek et al., 2012).

### 2.2.1 Data-driven MTA Models

Due to the limitations of the heuristics-based attribution methods, data-driven attribution methods have been developed in the literature. Dalessandro et al. (2012) propose properties of an adequate attribution system, which are the following:

- Fairness: An adequate method ought to give credit to individual channels in line with the channel's capacity to affect the likelihood of a conversion
- Data driven: An adequate method ought to learn the attribution from the data of the advertising campaign of interest.
- Interpretability: An adequate method ought to be accepted by all parties with material interest in play, based on the statistical merit as well as intuitive comprehension of the model.

Shao and Li (2011) are among the first to propose a data driven approach. Their first model is a BLR model which is used to classify conversions of customer journeys. Shao and Li (2011) opt for the BLR over a regular Logistic Regression, because the concept of bagging in combination with a logistc regression will result in a more stable model in terms of variability than a regular Logistic Regression. Moreover, the bagged model will retain the ease of interpretability, which is

in line with the third property of adequate attribution models put forward by Dalessandro et al. (2012). Additionally, Shao and Li (2011) propose a probabilistic model. First, the model learns the aggregate distribution of the channels with regard to the conversions in the data. Then, the contribution of each individual touchpoint on an individual customer journey level is determined and a second-order interaction term is included to account for interaction effects between any two channels. As a result, for a particular journey, the attribution of a given touchpoint is the contribution of the touchpoint itself, as well as any interaction effects between the given touchpoint and any other touchpoints (Shao and Li, 2011). Shao and Li (2011) state that higher-order interactions could also be included. However, the number of data instances with the same higher-order interactions drops sharply and consequently, these higher-order interactions are left out (Shao and Li, 2011).

Dalessandro et al. (2012) extend on the probabilistic model of Shao and Li (2011), by including these higher-order interactions and describe that the model is equivalent to the Shapley Value solution from cooperative game theory (Shapley, 1953). Within cooperative game theory, the Shapley value is the concept of fairly distributing profits among the players participating in a coalition (Shapley, 1953). Intuitively, this can be processed by iteratively excluding a single player from the coalition and assigning the decrease in value as a result of the exclusion to the excluded player. A major advantage of the Shapley value solution in the MTA context, is its ease of interpretation due to its straightforward computations (Dalessandro et al., 2012). In addition, the attribution by the Shapley value solution is fair: If a channel doesn't contribute, its attributed credit will be zero. Moreover, two identical channels will be assigned equivalent credit (Dalessandro et al., 2012). Nevertheless, determining credit attribution by means of Shapley values becomes computationally intensive as the number of channels $k$ increase, as $2^k$ Shapley values need to be determined. Moreover, the Shapley Value solution only considers the journeys which result in a conversion for the attribution task (Dalessandro et al., 2012). Nonetheless, attribution solutions offered by some firms in the marketing analytics industry, such as Google and Nielsen, use a variant of the Shapley value solution for MTA as described above (Kannan et al., 2016).

Importantly, the previous discussed MTA models disregard the sequential nature of customer journeys. Nevertheless, other MTA models have been suggested which do incorporate this aspect. Abhishek et al. (2012) have introduced a Hidden Markov model in the MTA context. The authors aimed to incorporate the notion of the conversion funnel, where the audience goes through the states of being disengaged, becoming aware of the product or service, considering the product or service and finally a conversion. In the proposed model, an advertisement can cause an increased probability of conversion, as well as an increased probability of the audience transferring to another state in the consumer funnel. As a result, Abhishek et al. (2012) were able to infer

whether different advertisements affect the audience differently, dependent on the current state in the conversion funnel the audience is. Thus, this model incorporates the sequential aspect of customer journeys. The attribution scheme for this model is based on the incremental effect of an advertisement on the probability of conversion.

Anderl et al. (2016) have also employed a Markov model in their attempt to integrate the sequential nature of customer journeys by means of first- and higher-order Markov Chains. The first-order Markov assumption states that the information captured at time $t$, is fully explained by the feature at time $t-1$, implying information before $t-1$ does not matter (Keilson, 1979). As this assumption arguably does not hold in the context of MTA (Chierichetti et al., 2012), Anderl et al. (2016) have also introduced higher-order Markov chains, although they limited the number of orders to four due to the amount of parameters increasing exponentially with the number of orders (Anderl et al., 2016). Alike to the model put forward by Abhishek et al. (2012), this model determines the probability of the audience moving from one state to the other, the states being different marketing channels or the absorption state. The absorption state signifies the end-result of a customer journey, being a conversion or not (Anderl et al., 2016). Attribution with this model for marketing channel $i$ is determined over the change in probability of reaching the conversion state, when channel $i$ is removed from the transition possibilities; the so called removal effect (Anderl et al., 2016).

### 2.2.2 Deep Learning MTA Models

Another branch of machine learning algorithms which have been intensively studied in the stream of literature on data-driven MTA models, are deep learning models. The corner stone of deep learning models, are neural networks (James et al., 2021). The name of neural networks and their structure have been inspired by the human brain, as these models imitate the manner biological neurons signal to one another (Goodfellow et al., 2016). The term neural networks doesn't refer to a single model, but instead is an umbrella term which encompasses a large class of models.

In the following part, the relatively simple and widely utilized single layered feed-forward neural network is discussed to provide a main notion of how neural networks generally work. The features $X$ are first fed to an input layer, where the features $X$ make up the units in this input layer. Then, each of the inputs in the input layer are directed to the hidden units, which derive the features $A$ from the input features $X$ through non-linear transformations, referred to as activation functions, and linear combinations of these input features $X$. The resulting derived features $A$ are typically denoted as activations (Hastie et al., 2009). These activations $A$ are then leveraged in the output layer for modelling the output variable $y$. The output layer allows for the inclusion of an output function for a final transformation of the derived features. The

previous description is applicable to a simple feed forward neural network with a single hidden layer, whereas in reality these type of models have more than one hidden-layer and many hidden units in each layer (Hastie et al., 2009). Feed-forward neural networks have shown to perform well in areas such as computer vision and speech recognition (Hastie et al., 2009).

A type of neural networks which specifically have been leveraged in the MTA context, are Recurrent Neural Networks (RNN). The structure of the RNN is designed in such a manner to accommodate and take advantage of the sequential nature of the input data (Hastie et al., 2009). Intuitively, the design of the RNN suits data on customer journeys well, as customer journeys are sequences of touchpoints (Ren et al., 2018). Contrary to the feed-forward neural networks, which takes as input the features $X$, a RNN takes as input a sequence $S$ consisting of vectors representing each element in the sequence, $S = (s_1, s_2, ..., s_L)$. Each of these elements $s_l$, can consist of $p$ components, which in the MTA context would represent the features associated with each touchpoint: $s_l = (s_{l1}, s_{l2}, ..., s_{lp})$. The RNN processes a sequence one element at a time, updating the activations $A_l$ with each element in the sequence by considering the previous activation $A_{l-1}$ and the current element $s_l$ of the sequence. Subsequently, each $A_l$ is fed to the output layer and generates a prediction $O_l$ for $y$; $O_L$ being the last one and used as the final prediction in case of a binary classification problem (James et al., 2021). An extension of the RNN architecture, utilizes the long short-term memory (LSTM) architecture. This ensures that when activation $A_l$ is computed, it receives input from activations both further- and closer back in time; the LSTM-RNN (James et al., 2021). Various LSTM-RNN based methods have been introduced in the literature to tackle the MTA problem (Ren et al., 2018); (Du et al., 2019); (Kumar et al., 2020); (Yao et al., 2021).

Neural networks tend to be referred to as black boxes. Black boxes are referred to as models which can be represented by the input and output, yet how these models exactly combine variables and make predictions, i.e., the model's internal workings, cannot be comprehended by humans (Molnar, 2018). Specifically for the neural networks discussed above, these models perform numerous non-linear transformations on features and various linear combinations of these features are built. As a consequence, it is impossible for humans to track the mapping of input data to prediction (Molnar, 2018). Nonetheless, it is possible to approximate the internal workings of a black box by means of post-hoc interpretation methods. One such method is the Shapley Value mentioned earlier in this section, which determines the contribution of each player in a coalition game (Shapley, 1953). Du et al. (2019), Yang et al. (2020) and Yao et al. (2021) formalize LSTM-RNN based models, which post-hoc utilize Shapley Values to determine the credit allocation across the marketing channels and other features included in their studies. Alternatively, Ren et al. (2018) leverage an attention mechanism in their model to determine the credit attribution to the various channels in their data. Subsequently, they evaluate the

attribution of their own LSTM-RNN based models and other MTA models by means of budget allocation experiments, in which customer journeys are replayed based on an available timestamp feature and a limited budget is allocated to the channels, based on the attribution results of the models utilized in their study.

## 2.3 Interpretability in Machine Learning

### 2.3.1 Relevance & Scope of Interpretability

In the previous section, the properties of an adequate attribution system put forward by Dalessandro et al. (2012) was introduced. One of these criteria for an adequate attribution system, is interpretability. Specifically, an adequate attribution approach ought to be accepted by all parties involved, based on statistical merit as well as intuitive comprehension.

Nonetheless, why would parties involved not just have confidence in a methodology, if it performs well in terms of predictive performance? The issue is that reporting a performance metric, such as predictive accuracy, is insufficient for various real-world problems (Doshi-Velez and Kim, 2017). In particular in the MTA context, is it sufficient to know what customer journeys eventually convert? As the aim of MTA is to uncover what marketing channels and other features drive conversions, quantifying a model's predictive performance is not sufficient and explanations regarding the predicted class label of the customer journeys are arguably required.

Interpretability has been an important topic in general in the literature on machine learning and artificial intelligence since 2016 (Kim et al., 2016); (Miller, 2017). It is no coincidence that in the same year, the European Union adapted the General Data Protection Regulation (GDPR), a binding legislative act with regard to the collection, storage and usage of personal information (Mita, 2021). Specifically, Article 22 restricts *'automated individual decision-making, including profiling'*, and states a *'right to explanation'* (EU, 2016). As a consequence, interpretable, i.e., explainable, AI and machine learning became a requirement by law in certain areas.

Nevertheless, quantifying interpretability, or explainability for that matter, in light of machine learning and AI has proven to be difficult. There is little consensus on a definition of interpretability in machine learning and on how to evaluate it (Doshi-Velez and Kim, 2017). Mita (2021) argues that various of the definitions of interpretability in machine learning and AI suggested in the literature are too limited, and refers to the definition put forward by (Arrieta et al., 2020) as a more coherent and comprehensive definition: *'Given an audience, an explainable AI is one that produces details or reasons to make its functioning clear or easy to understand'.*

Interpretable machine learning methodologies can be categorized in line with a number of

criteria. Two of such criteria by means of which interpretable methodologies can be categorized are the following (Molnar, 2018):

1. Intrinsically interpretable methods and post-hoc interpretation methods.

    - Intrinsically interpretable methods are models which are considered to be interpretable by themselves due to their relative simple structure, such as decision-trees and sparse linear models. Contrary to this, post-hoc interpretation methods are utilized to interpret a model, after the model has been trained. An example of a post-hoc interpretation method is the Shapley Value (Shapley, 1953).

2. Local- and global interpretation methods.

    - Local interpretation methods refer to methods which attempt to elucidate an individual prediction, whereas global interpretation methods are utilized to infer how the model makes predictions or how parts of the model affect the made predictions (Molnar, 2018).

### 2.3.2 Rule-Based Methods

One such class of intrinsically interpretable models are rule-based methods, which output a set of decision rules. Decision rules are if-then statements, which can contain one or more conditions and a consequent resulting from satisfying said conditions. Decision rules follow a common structure: **If** a condition is true, **then** a certain prediction is made. Because decision rules follow a structure which semantically resembles natural language, decision rules are one of the most interpretable models, given that the length of the conditions are short and there are not too many rules (Molnar, 2018).

The decision rules produced by a model are typically combined in one of two fashions. The decision rules are either combined into a decision list or into a decision set, which differentiate in terms of their structure. Decision lists introduce an hierarchy in the decision rules, by means of an if-then-else structure. The rules in decision lists are considered in order, beginning at the top rule and ending with the last one. Once a rule of which the conditions are applicable to a data instance is encountered, that rule's consequent is used to make a prediction for the data instance (Molnar, 2018). The structure of a decision list means that additional rules further down in the hierarchy can only cover an increasingly smaller section of the feature space. Consequently, rules further down the list are progressively less interpretable, as these only cover increasingly narrower situations. Accordingly, a growing number of conditions must be comprehended before a rule is applicable, resulting in these decision lists being less interpretable (Lakkaraju et al., 2016). On the contrary, decision sets do not introduce such a hierarchical structure. The rules

in a decision set can be considered in any order. The predicted class of instances covered by multiple rules, can for example be decided by majority vote. Moreover, if an instance is not covered by a single rule, this rule can be assigned a default[1] class. Due to not having the if-then-else structure, the rules in the decision set do no have to cover a narrower feature space, as with the decision lists. Consequently, the decision-rules in a decision set have to be an accurate predictor individually. Therefore, decision sets are typically more interpretable than decision lists, as humans can consider the rules in a decision set one at a time to understand them (Molnar, 2018).

A model which outputs a decision set, is the repeated incremental pruning to produce error reduction (RIPPER) algorithm by Cohen (1995). The RIPPER algorithm is a variant of sequential covering, which applies an intuitive simple idea: find a good rule which applies to some data points, remove these data points and then repeat the task until no more points are left or a stop condition is met (Molnar, 2018). Although the RIPPER algorithm has shown to perform great in terms of interpretability, the rather simplistic model is no match for more complex machine learning techniques leveraged nowadays.

On the other hand, the Classy algorithm (Proença and van Leeuwen, 2020) produces a probabilistic rule list, where each rule's consequent consists of a categorical distribution of the classes. Classy utilizes the minimum description length principle to ensure a balancing between accuracy and interpretability. Moreover, this methodology successfully avoids overfitting and the requisite of parameter tuning.

Lumadjeng et al. (2023) have utilized linear programming to construct rules by means of column generation and introduce two methods which both output a decision set: the rule generating (RUG) and rule extraction (RUX) algorithm. Whereas the RUG algorithm generates rules through fitting a decision tree, the RUX algorithm extracts rules from an existing tree-ensemble (e.g., Random Forest or Gradient Boosted trees). The decision sets formulated by these models consist of rules along with weights representing the importance of the rules, hence the prediction of an instance is given by a weighted combination of the rules applicable to that instance. Similarly to the RUX algorithm, the RuleFit algorithm put forward by Friedman and Popescu (2008) also obtains rules from an existing tree ensemble. The RuleFit algorithm learns a sparse linear model, which includes both the rules obtained from the tree ensemble and the original features. As the number of rules extracted from ensemble methods can be rather large, the model leverages lasso to shrink the coefficients of rules and features which do not add sufficient explanatory power to zero. This results in only having to consider a subset of the original features and extracted rules, making the model more comprehensible (Friedman and Popescu, 2008).

[1] Also applicable to rule lists

These rule-based methods specified above will be leveraged in the MTA context to classify customer journeys and subsequently determine the most relevant features with regard to converting and non-converting customer journeys from the produced rules. These methods will be elaborated on more in depth in the methodology section.

# 3 Data

## 3.1 Criteo Dataset

Criteo is a firm in online advertising which focuses specifically on display advertising. Criteo's product intents to provide their client's customers with personally tailored display advertisements, based on these customers' online browsing preferences and behavior. Criteo AI Lab is a team within Criteo which focuses on pioneering innovations in computational advertising. Criteo AI Lab has published a dataset for attribution modelling in a real-time advertisement auction setting (Diemert et al., 2017) which has been utilized regularly in the MTA context (Ren et al., 2018); (Kumar et al., 2020); (Yao et al., 2021).

The dataset contains 30 days of Criteo live traffic data. Each individual line in the dataset corresponds to a single impression of a display advertisement. In total, the dataset contains 16.5 million impressions from 6.1 million unique users. For each individual impression, the dataset contains information on: The relative timestamp of the impression, whether the advertisement was clicked, an unique user identifier, a conversion identifier if applicable, the timestamp of the conversion if applicable, the number of clicks and the time since the last click. Additionally, the dataset contains nine categorical variables associated with contextual features of the advertisment, publisher and user (Diemert et al., 2017). These variables have been anonymized and consequently their exact meaning is unknown. Finally, the dataset also includes variables on the price paid by Criteo for the advertisement and the order cost, in case a conversion is attributed to Criteo. The relevant feature for this thesis are the click, campaign and the nine contextual categorical features. Relevant information on these features is displayed in table 2. From the individual touchpoints, customer journeys can be constructed. The dataset consists of 6.5 million customer journeys of which 806 thousand convert. Figure 1 displays the total number of journeys and converting journeys with regard to the number of touchpoints along these journeys.

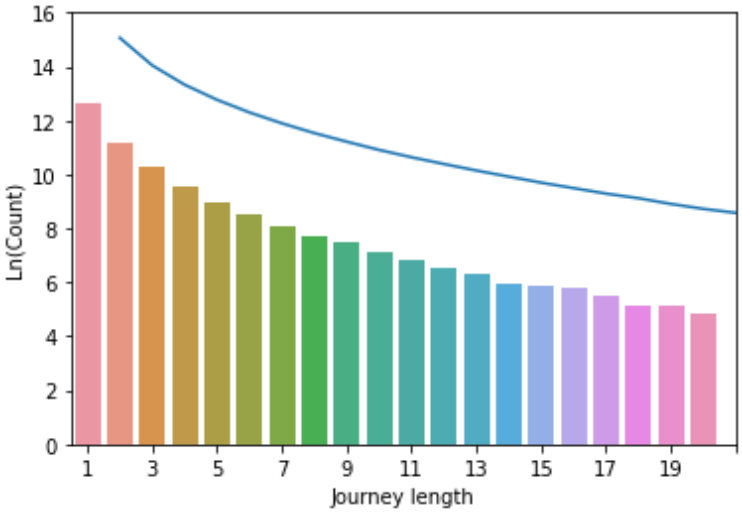
## 3.2 Data preprocessing

To transform the touchpoints into sequences, the user- and conversion identifiers are utilized. With these features, the customer journeys within the dataset are identified and sorted by the timestamp variable. If an user-id is related to two distinct conversions, the sequence will be

Table 2: Number of categories of the Campaign ID and Cat[1-9] variables.

| **Variable** | **# Options** |
|---|---|
| Click | 2 |
| Campaign | 675 |
| Category 1 | 9 |
| Category 2 | 70 |
| Category 3 | 1829 |
| Category 4 | 21 |
| Category 5 | 51 |
| Category 6 | 30 |
| Category 7 | 57,196 |
| Category 8 | 11 |
| Category 9 | 30 |

Figure 1: Total number of sequences (line) and converting sequences (bars) with regard to the number of touchpoints.




split on the conversion timestamp to ensure the construction of separate sequences for all (non-)converting customer journeys. As customer journeys with more than 20 touchpoints represent less than 0.6% of all the sequences, these customer journeys are omitted; which is also in line with Ren et al. (2018) and Kumar et al. (2020). Additionally, the sequences consisting of just a single touchpoint are removed as well. For training and evaluation of the models, a subset of one million touchpoints is extracted from which the customer journeys are built. This results in 125,862 sequences of which 8,247 convert (6.5%). Descriptive statistics with regard to the length of converting and non-converting customer journeys of both the original dataset and preprocessed dataset, is displayed down below in table 3.

As specified, the variables presented in table 2 will be employed to train the models; which results in 11 variables per touchpoint. To restrain the dimensionality, three separate datasets

Table 3: Descriptive statistic of the length of converting and non-converting customer journeys.

| | Min | 25% Quantile | Median | Mean | 75% Quantile | Max |
|---|---|---|---|---|---|---|
| *Original dataset* | | | | | | |
| Non-converting journeys | 1 | 1 | 1 | 2.578 | 3 | 880 |
| Converting journeys | 1 | 1 | 1 | 1.838 | 2 | 164 |
| *Preprocessed dataset* | | | | | | |
| Non-converting journeys | 2 | 2 | 2 | 2.545 | 3 | 20 |
| Converting journeys | 2 | 2 | 2 | 2.707 | 3 | 20 |

are extracted, consisting of the features from the first 3-, 5-, and 7 touchpoints of each sequence respectively. Hence, the three datasets consist of 33, 55 and 77 variables respectively. The set which contains these three datasets will be referred to as $D$, such that: $D = \{d_1, d_2, d_3\}$, where $d_1$, $d_2$ and $d_3$ are the datasets which hold the features associated with the first 3-, 5- and 7 touchpoints respectively. In the further sections, an underscore plus a number will be added to the features to indicate which touchpoint the feature is associated with (e.g., Cat4_2 is the Cat4 feature which is associated with the second touchpoint). Moreover, to ensure each sequence comprises of the features of the first-n touchpoints for each respective dataset, padding is used. Padding sets the features associated with the touchpoints that are past the end of a customer journey to zero.

As can be viewed in table 2, some of the variables hold numerous options resulting in high cardinality. Consequently, performing one-hot encoding, a regular method to encode qualitative variables, on these variables would result in an exploding dimensionality. To avoid high dimensionality, Campos et al. (2016) have proposed to transform categorical features in numeric ones by applying the inverse document frequency (IDF) originating from text mining. This method has previously been utilized in the context of mobile marketing conversion prediction by Pereira et al. (2019) and Pereira et al. (2021). This method transforms a level $t$ of a categorical feature to a numeric value in the following manner:

$$IDF(t) = \ln(\frac{N}{n_t}) \tag{1}$$

Where $N$ is the total number of data instances and $n_t$ is the frequency of the categorical level $t$ in the data. Consequently, categorical levels with a high frequency are transformed to a numeric values relatively close to zero, while levels with a lower frequency tend to the maximum value of $\ln(N)$. In addition, the new numeric values are accumulated, beginning at the lowest and ending at the highest IDF-transformed values, such that the new numeric values can be more easily be mapped back to the original levels in the categorical features. A toy example displaying the IDF transformation and the accumulation is presented in table 4. The features in the datasets

in $D$ will be transformed in line with the accumulative IDF-transformation.

Table 4: Toy example of the accumulative IDF-transformation (N = 40) with a categorical feature consisting of four levels

| Level | $n_t$ | $IDF$ value | Accumulative $IDF$ value |
|---|---|---|---|
| A | 20 | 0.693 | 0.693 |
| B | 12 | 1.204 | 1.897 |
| C | 5 | 2.079 | 3.976 |
| D | 3 | 2.590 | 6.566 |

# 4 Methodology

## 4.1 Bagged Logistic Regression

The first benchmark for this thesis that will be discussed, is the BLR put forward by Shao and Li (2011). The Logistic Regression is a widely used method for binary classification tasks (James et al., 2021). The Logistic Regression is combined with the notion of bagging, which results in less variability in the estimation procedure of the coefficients. The BLR is supposed to produce a fairly easily interpretable model with stable and reproducible results (Shao and Li, 2011). Whereas Shao and Li (2011) use data which represents the number of times a customer visits channel $k$ on their customer journey, the three datasets as described in the data section will be used. A Logistic Regression applies the logistic function, which results in the output values being between 0 and 1, which can be represented as follows:

$$logit(y_c) = \alpha + \sum_{j=1}^{J} \beta_j x_j \tag{2}$$

Subsequently, the probability of a customer journey converting given the features $X$ associated with the touchpoints, can be estimated in the following manner:

$$\hat{P}(Y = 1|X) = \Lambda(\hat{\alpha} + \sum_{j=1}^{J} \hat{\beta}_j x_j) \tag{3}$$

Where $\Lambda$ is the logistic function, $\Lambda(x) = \frac{1}{(1+e^{-x})}$.

Then, fitting a BLR involves the following two steps:

1. Generate a bootstrapped subset from the training set, containing a pre-specified proportion of the data instances. Additionally, sample a proportion of the features. In line with the recommendations by Shao and Li (2011), both the proportions are set to 0.5. Subsequently fit a Logistic Regression to the bootstrapped subset and sampled features.

2. Repeat step 1 M-times, after which the final coefficients are determined by taking the average over the estimated coefficients in the M iterations.

As mentioned, the BLR results in lower variability of the estimated coefficients than the regular Logistic Regression. As the attribution with these models is based on the estimated coefficients of each channel, lower deviations of the coefficients is preferred in attribution modelling to ensure fair conclusion with regard to credited conversions (Shao and Li, 2011).

## 4.2 Tree-ensembles

### 4.2.1 Random Forest

A Random Forest is an ensemble method which combines the output of multiple independently grown decision trees and makes predictions through majority voting among these trees (James et al., 2021). Decision trees start with a single root node and eventually branch out into terminal nodes. At each non-terminal node, a split is made based on a feature test condition. The feature $x_j$ and cut-off point $s$ which result in the highest impurity reduction is selected at each split:

$$\Delta I = I_z - w_{left(z)} I_{left(z)} - w_{right(z)} I_{right(z)} \tag{4}$$

Where $I_z$ represent the impurity at node $z$, $w_{left(z)} I_{left(z)}$ the weighted impurity of the resulting left child node and $w_{right(z)} I_{right(z)}$ the weighted impurity of the right child node. Consequently, the split results in two nodes comprising of samples which are most homogeneous with regard to the class of interest (Hastie et al., 2009). The impurity measure utilized to this end, is the Gini index:

$$G = \sum_{k=1}^{K} \hat{p}_{zk}(1 - \hat{p}_{zk}) \tag{5}$$

Where $K$ is the number of classes and $\hat{p}_{zk}$ is the proportion of class $k$ in node $z$. The predicted class of an observation with a decision tree, is the most occurring class in the terminal node the observation ends up in (Hastie et al., 2009). Subsequently, a Random Forest is built by growing a number of trees on bootstrapped training samples. Importantly, a Random Forest differentiates from the notion of bagging, by only considering a random sample of $p$ features at each split. This process decreases the correlation among the grown decision trees, thereby making the average of the resulting tree less variable and more reliable (James et al., 2021).

### 4.2.2 Gradient Boosting

Akin to Random Forest, Gradient Boosting involves[2] growing a multitude of decision trees. Except with Gradient Boosting, the trees are grown sequentially and learn from the mistake's made

[2]Other so called base-learners can be leveraged with the Gradient Boosting algorithm to build an ensemble

by their predecessors (Hastie et al., 2009). As such, the model involves minimizing a loss function which represents the difference between predicted and actual classes $L(f) = \sum_{i=1}^{N} L(y_i, f(x))$, which for binary classification is the binary cross-entropy (i.e., log-loss) (Hastie et al., 2009):

$$L = -\frac{1}{N}\sum_{i=1}^{N} y_i(1 - \log(\hat{p}_i)) + (1 - y_i)log(1 - \hat{p}_i) \tag{6}$$

The Gradient Boosting algorithm involves creating an ensemble $F_M$, which consists of $M$ trees. The algorithm is initiated with $F_0$ by finding a constant $\gamma$ which minimizes:

$$F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{N} L(y_i, \gamma) \tag{7}$$

In order to solve the minimalization of the loss function, the gradient descent is applied, which involves examining the derivative of the loss function and finding the steepest decent (Friedman, 2001). At each iteration $m$ the pseudo residuals $r_{im}$, also referred to as negative gradient, are determined on the loss function of the previous model $F_{m-1}$:

$$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}} \tag{8}$$

For each individual $r_{im}$ of data instance $i$, the negative gradient is the steepest decline in function space (Friedman, 2001). After computing these values, a decision tree $t_m$ is fitted onto $r_{im}$. Finally, the multiplier $c_m$ is computed and adopted to update the model:

$$c_m = \arg\min_{c} \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + c\ t_m(x_i)), \forall j = 1, ..., J \tag{9}$$

$$F_m(x) = F_{m-1} + \zeta c_m t_m(x) \tag{10}$$

Where in equation (10), $\zeta$ is the learning rate which controls the contribution of each tree. This iterative process is continued until a pre-specified number of iterations $M$ are completed. The resulting Gradient Boosted trees consists of $M$ trees.

## 4.3 Rule-based Methods

### 4.3.1 RIPPER

The RIPPER (repeated incremental pruning to produce error reduction) is an inductive rule-learning approach proposed by Cohen (1995). Cohen (1995) extends on the methodology put forward by Fürnkranz and Widmer (1994) who introduced the incremental reduced error pruning (IREP) algorithm. Although the IREP algorihm is shown to be efficient by Cohen (1995), the error rates produced by this algorithm are regularly higher than the error rates of well known CART algorithm by Breiman et al. (1984). Furthermore, the RIPPER algorithm has been shown to be able to efficiently process large datasets and datasets which contain an imbalanced class distribution (Cohen, 1995).

As the task of predicting conversions of customer journey involves a binary-classification problem, the RIPPER algorithm searches for rules which classify data instances as being converted customer journeys. The non-converting customer journeys are the majority class within the data and as a consequence, non-conversions are set as the default class (Cohen, 1995). If none of the produced rules by the algorithm are applicable to a data instance, these instances are covered by the default rule. The predicted class in the default rule, is the default class (Cohen, 1995). Including such a default rule with a default class ensures the decision set covers the entire feature space, resulting in the decision set being exhaustive (Molnar, 2018).

As stated, the primary goal of the RIPPER algorithm is to reduce the error rates, while also maintaining a relative simplistic set of rules and accordingly restrain the model from overfitting. To achieve this, the procedure of training the RIPPER algorithm consists of two steps: Rule construction and rule optimization (Cohen, 1995).

Upon initialization of the RIPPER algorithm, the training set is split in a grow set, used for growing the rules, and a prune set for subsequent pruning of the grown rules. Rules are constructed based on the sequential covering algorithm. Specifically, an individual rule $r$ is generated by greedily adding the best condition $\alpha^*$ to the conjunction of conditions $S$. To determine the best predicate $\alpha^*$, every possible condition within the feature space is assessed by utilizing FOIL's information gain criterion:

$$FOIL(S_0, S_1) = L(\log_2 \frac{p_1}{p_1 + n_1} - \log_2 \frac{p_0}{p_0 + s_0}) \tag{11}$$

Where $p_i$ and $n_i$ represent the number of positive (conversion) and negative (non-conversion) data instances covered by the conditions in $S_i$, $i = 0, 1$. In addition, the set $S_0$ indicates the current set of conditions, while $S_1$ includes the additional new predicate $\alpha^* \notin S_0$. After iteratively adding the best condition $\alpha^*$, the final set of conditions $S^*$ is gradually pruned by evaluating the cover-ratio:

$$v(S^*) = \frac{p + (N - n)}{P + N} \tag{12}$$

Where $p$ and $n$ denote the number of converting and non-converting instances respectively covered by $S^*$, while $P$ and $N$ represent the total number of converting and non-converting instances respectively. The model determines whether a predicate should be included within the final set of predicates $S^*$, beginning at the final predicate $\alpha_C$, by assessing whether the cover-ratio is higher without the predicate:

$$v(S^* \setminus \alpha_C) \geq v(S) \tag{13}$$

If the cover-ratio is higher without the predicate $\alpha_C$, the predicate is removed. Each individual condition within the set of conditions $S^*$ of the rule is considered one-by-one. Once construction

of a rule is finalized, the data instances covered by this rule are omitted and the algorithm continues constructing new rules, until one of the following terminating conditions is met:

1. The current set of rules has an error rate which exceeds 50%
2. There are no more converting customer journeys left in the training set
3. The description length of the set of rules is more than $x$ bits larger than the smallest description length obtained so far.

Then, the rule optimization phase of the RIPPER algorithm commences. The rule optimization phase involves considering two alternative variants of each individual rule. The first variant is constructed anew from an empty set of conditions and thereafter pruned, while the other variant is constructed by growing the original rule through adding conditions and subsequent pruning of this revised version of the original rule. Out of these three constructed rules, the original plus the two variants, the rule with the smallest description length is selected as the final rule (Cohen, 1995).

### 4.3.2 Classy

An alternative type of rule-based learners, are the probabilistic rule lists. Similarly to a regular rule, a probabilistic rule $r_k$ exists of a set of predicates $S_k$. Nevertheless, whereas with the RIPPER algorithm the consequent of a rule was typically the minority class label, the Classy algorithm provides each rule with a categorical distribution $\boldsymbol{\theta}(S)$ over both the converting- and non-converting class label (Proença and van Leeuwen, 2020). In the case of binary classification, the categorical distribution comprises of probabilities of both the positive and negative class:

$$\boldsymbol{\theta}(S) = (\theta_p, \theta_n), \theta_p + \theta_n = 1, \theta_p, \theta_n > 0 \tag{14}$$

Hence, a probabilistic rule contains a set of predicates and a consequent in the form of the categorical distribution. The global structure of the probabilistic rule list is an ordered set of rules, each rule consisting of a set of predicates and a categorical distribution. The rule list is concluded with the default rule, which typically assigns the data instances unaffected by the previous rules to the majority class (Proença and van Leeuwen, 2020). A probabilistic rule list could for instance look the following:

$$r_1 : \boldsymbol{IF}\ \theta_1\ and\ \theta_2\ \boldsymbol{THEN}\ Pr(conversion) = 0.8,\ Pr(non-conversion) = 0.2$$

$$r_2 : \boldsymbol{IF}\ \theta_3\ and\ \theta_4\ \boldsymbol{THEN}\ Pr(conversion) = 0.6,\ Pr(non-conversion) = 0.4$$

$$r_3 : \boldsymbol{IF}\ \theta_5\ and\ \theta_6\ \boldsymbol{THEN}\ Pr(conversion) = 0.3,\ Pr(non-conversion) = 0.7$$

$$r_{\emptyset} : \textbf{\textit{ELSE}}\ Pr(non-conversion) = 1.0$$

A data instance is classified by going through the list top-down and assigning the class with the highest probability level of the first rule applicable to the data instance. If none of the rules in the rule list are applicable to an observation, the model enforces the default rule onto this observation (Proença and van Leeuwen, 2020). With such a probabilistic rule list, the coverage of the set of conditions $Cov(S_k)$ of a rule $r_k$ is defined as the number of data instances which satisfy the set of conditions $S_k$. Considering the nested if-else structure of the rule list, the usage of the set of predicates $U(S_k)$ of a rule $r_k$ is equal to the coverage of this rule subtracted by the number of instances covered by the preceding rules (Lakkaraju et al., 2016).

One such model which outputs a probabilistic rule list, is the Classy algorithm introduced by Proença and van Leeuwen (2020). The Classy algorithm constructs a rule list based upon the Minimum Description Length (MDL) concept. Let $R$ be the complete space of possible rule lists, then the Classy algorithm is designed to find the optimal rule list $R^* \in R$ which minimizes:

$$L(D, R) = L(Y|X, R) + L(R) \tag{15}$$

Where $L(R)$ is the encoded length in bits of the rule list and $L(Y|X, R)$ is the encoded length of the class labels, given the avalaible data instances $X$ and the probabilistic rule list $R$. In line with the law of parsimony associated with the MDL principle, the encoding of a more complex model, as opposed to encoding a simpler model, should result in larger code lengths, and hence a model which can be more concisely encoded is preferred (Grünwald, 2007). The rule list model is encoded by means of the universal code for integers (Rissanen, 1983) and the uniform code (Grünwald, 2007). For encoding the data, the prequential plug-in code is used, due to its asymptotic optimality without requiring prior information on the class probabilities. This approach provides the smoothed maximum likelihood estimators of the categorical distributions of the consequents (Proença and van Leeuwen, 2020):

$$\hat{\theta}_k = \frac{U(S, k) + \epsilon}{U(S) + |Y|\epsilon} \tag{16}$$

Where $U(S, k)$ denotes the usage of the set of conditions $S$ while considering instances with class label $k$. Additionally, $\epsilon$ is a small pseudo count which is added to each class-specific usage to prevent zero-probabilities occuring for any class.

The Classy algorithm itself is initialized with a rule list containing just the default rule. The algorithm then iteratively adds rules from a candidate set which results in the largest normalized compression gain of the data. The normalized compression gain is defined as follows:

$$\delta L(Y|X, R \oplus r_p) = \frac{\Delta L(Y|X, R \oplus r_p)}{U(S)} \tag{17}$$

The numerator in equation (17) denotes the absolute compression gain which is normalized by means of the coverage of the set of predicates of the new rule. Due to normalizing over the number of instances covered by the rule, normalized gain will prefer rules that cover less observations, yet provides higher accuracy in comparison to absolute gain. Once a rule is added, the compressed data by this rule is removed and a next best rule is seeked. This procedure continues, until no more rules are present in the candidate set which would result in a positive normalized compression gain if added to the probabilistic rule list (Proença and van Leeuwen, 2020).

#### 4.3.3 RuleFit

The RuleFit algorithm proposed by Friedman and Popescu (2008) learns a sparse linear model, which includes extracted decision rules from tree-based ensembles, such as the Random Forest and Gradient Boosted trees. Extracting rules from such tree-ensembles can be accomplished by traversing the generated trees from the root-node to each leaf-node, which results in if-else statements being extracted from the trees (Friedman and Popescu, 2008). Let $X_m$ represent the features exploited in tree $t_m$. Moreover, $Z_j$ is the set consisting of all possible values for feature $x_j$ and $z_{jq}$ is a specified subset of $Z_j$, which contains $q$ of these values for feature $j$. Then, each rule $r_k$ can be denoted as:

$$r_k(x) = \prod_{j \in X_m} I(x_j \in z_{jq}) \tag{18}$$

Where $I(\cdot)$ is the indicator function which equals 1 if the value of input variable $x_j$ is present in the subset of values $z_{jq}$. For numerical features, which the pre-processed datasets as described in section 3 primarily consist of, the subsets of values $z_{jq}$ are transformed into intervals.

As previously mentioned, tree-ensembles such as the Random Forest and Gradient Boosting can be used to produce the decision-rules. Such a tree-ensemble can be expressed by $\{f(x)\}_{m=1}^{M}$, where $M$ is the number of decision trees and each individual tree's output is denoted as $f_m(x)$. The set of decision rules extracted from the ensemble can be expressed as $r_k(x)_{k=1}^{K}$ and the total number of rules extracted can be denoted as:

$$K = \sum_{m=1}^{M} 2(l_m - 1) \tag{19}$$

Where $l_m$ is the number of terminal nodes of tree $m$. The resulting sparse linear model which only considers the produced rules from the tree-ensemble, is denoted as follows:

$$F(x) = \hat{\alpha}_0 + \sum_{k=1}^{K} \hat{\alpha}_k r_k(x) \tag{20}$$

Where $\hat{\alpha}_0$ is an estimated constant and $\hat{\boldsymbol{\alpha}}$ are the estimated weights of the rules. As a consequence of RuleFit utilizing lasso, an additional constraint is introduced in the loss function. This

additional constraints results in some of the weights in $\hat{\boldsymbol{\alpha}}$ to receive a zero estimate (Molnar, 2018).

$$\{\hat{\alpha}\}_0^K = \underset{\{\hat{\alpha_k}\}_0^K}{\operatorname{argmin}} \sum_{i=1}^{n} L(y_i, \hat{\alpha}_0 + \sum_{k=1}^{K} \hat{\alpha}_k r_k(x)) + \lambda \sum_{k=1}^{K} |\alpha_k| \tag{21}$$

Where $\lambda$ is the parameter which controls the magnitude of the shrinkage of the weights.

#### 4.3.4 Rule Generation & Rule Extraction

Lumadjeng et al. (2023) introduce two rather similar rule-based classification methods, the Rule Generation (RUG) and Rule Extraction (RUX) algorithms. Both methods utilize linear programming which enables to incorporate additional constraints and the use of the column generation procedure, resulting in these methods being scalable to large datasets (Lumadjeng et al., 2023). To this end, a vector-valued mapping for the classes is introduced. In the case of binary classification, this mapping function and the prediction vector for data instance $i$ can be denoted as

$$\mathbf{y_i}(\mathbf{w}) = \begin{cases} (1,-1) & \text{if } y_i = \text{conversion} \\ (-1,1) & \text{if } y_i = \text{no conversion} \end{cases}, \hat{\mathbf{y}}_\mathbf{i}(\mathbf{w}) = \sum_{k \in K} \alpha_{ik} \mathbf{R_k}(\mathbf{x_i}) w_k \tag{22}$$

Where $\hat{\mathbf{y}}(\mathbf{w})$ is the predicted class for instance $i$, $\mathbf{w}$ is the vector of nonnegative rule weights associated with the rules. In addition, $\alpha_{ik}$ equals 1 if instance $i$ is covered by rule $k \in K$ and 0 otherwise. Consequently, the vector $R_k(x_i)$ is only allocated to input $x_i$ if instance $i$ is covered by rule $k$. Note that the eventual prediction by the model is formulated as a weighted combinations of the rule predictions and is given by the largest element in $\hat{\mathbf{y}}_\mathbf{i}(\mathbf{w})$. In order to assess the misclassification rate of the model, the hinge-loss is utilized, which for binary classification problems can be denoted as:

$$L(\hat{\mathbf{y}}_\mathbf{i}(\mathbf{w}), \mathbf{y_i}) = \max\{1 - \frac{1}{2}\hat{\mathbf{y}}_\mathbf{i}(\mathbf{w})\mathbf{y_i}, 0\} \tag{23}$$

If the hinge loss of all samples $i \in I$ are aggregated, the total hinge loss or classification error becomes:

$$\sum_{i \in I} \max\{1 - \frac{1}{2}\hat{\mathbf{y}}_\mathbf{i}(\mathbf{w})\mathbf{y_i}, 0\} \tag{24}$$

The primary aim of the algorithms is to minimize the misclassification rate by employing the following linear program:

$$\begin{aligned} \text{minimize} \quad & \lambda \sum_{k \in K} c_k w_k + \sum_{i \in I} v_i \\ \text{subject to} \quad & \sum_{k \in K} \hat{\alpha}_{ik} w_k + v_i \geq 1, \quad i \in I \\ & v_i,\ w_k \geq 0, \qquad\qquad i \in I,\ k \in K \end{aligned} \tag{25}$$

Where $\hat{\alpha}_{ik} = \frac{1}{2}(\alpha_{ik}\mathbf{R_k}(\mathbf{x_i})\mathbf{y_i})$ measures the classification accuracy of rule $k$ for instance $i$, given that instance $i$ is covered by rule $k$, and $v_i,\ i \in I$, is an auxiliary variable such that $v_i \geq L(\hat{\mathbf{y}}_\mathbf{i}(\mathbf{w}), \mathbf{y_i})$. This auxiliary variable $v_i$ imposes the objective of minimizing the error-rate by evaluating the hinge-loss described in equation (23). It can be deduced that if an instance $i$ is classified correctly by the model, then $v_i = 0$. Nevertheless, a positive value of $v_i$ does not by definition imply an incorrectly classified instance $i$, it rather signifies the strength of an incorrect classification of instance $i$. Additionally, $c_k \geq 0,\ k \in K$, are cost coefficients which penalizes rules with relatively many conditions. Evidently, the objective function in equation (25) demonstrates a trade-off between accuracy and interpretability (Lumadjeng et al., 2023). Finally, as the objective function consists of two summation terms in different units, $\lambda$ is utilized for scaling.

After an initial set of rules $R_0$ is obtained, the initial set of rules is extended in an iterative fashion by means of the column generation procedure. Column generation is an efficient procedure for solving linear programs when the number of features is too large to consider all at once. The notion of column generation is to first consider a subset of the original set of features and iteratively consider additional features. The linear program with the limited number of features, is referred to as the restricted master problem. Once this problem is solved, the dual problem's optimal solution is obtained. With the dual solution, a pricing problem can be defined and solved to determine which features should be included in the next iteration (Lumadjeng et al., 2023).

With the RUG algorithm, the columns in the linear program in equation (25) correspond to the rules. At each iteration $t$, the dual problem of equation (25), which considers the current set of rules $R_t$, is solved. This dual problem can be denoted as:

$$\begin{aligned} \text{maximize} \quad & \sum_{i \in I} \beta_i \\ \text{subject to} \quad & \sum_{i \in I} \hat{\alpha}_{ik}\beta_i \leq \lambda c_k, \quad k \in R_t \\ & 0 \leq \beta_i \leq 1, \qquad i \in I \end{aligned} \tag{26}$$

Where $\beta_i,\ i \in I$ are the dual variables associated with the primal problem in equation (25). Once the solution $\beta^t$ to his problem is derived, the objective function in equation (25) can be improved by finding at least one rule $k^*$ which satisfies:

$$\overline{c}_{k^*} = \lambda c_{k^*} - \sum_{i \in I} \hat{\alpha}_{ik}\beta_i^t < 0 \tag{27}$$

Where $\overline{c}_{k^*}$ is the reduced cost of rule $k^*$. During each iteration, rules are added which satisfy equation (27). This process continues, until no more rules satisfy equation (27) and improve the objective function in equation (25) (Lumadjeng et al., 2023). Importantly, whereas the RUG

and RUX algorithms have an equivalent linear program structure, the algorithms differentiate in the approach of obtaining rules. The RUX algorithm extracts these rules from an existing ensemble method, while the RUG algorithm generates the rules itself by fitting a decision tree (Lumadjeng et al., 2023). The ensemble methods leveraged for the RUX algorithm in this thesis, will be a Random Forest and Gradient Boosting model.

The predictive performance of the RUG algorithm has been shown to be on par with the Random Forest algorithm, while the RUG algorithm is intrinsically interpretable, whereas the Random Forest algorithm is typically considered a black-box (Lumadjeng et al., 2023).

### 4.4 Model training procedure

As previously mentioned in the data section, three datasets were constructed: $D = \{d_1, d_2, d_3\}$, where $d_1$ consists of the first three touchpoints of each customer journey and $d_2$ and $d_3$ of the first five and seven touchpoints respectively. Each dataset is split in a training- and test set, the former containing 80% of the data instances and the latter 20% of the data instances. The train-test split is done in a stratified fashion to ensure both the training- and test set hold the same ratio of converting to non-converting customer journeys.

First of all, every model except for the BLR and the Classy algorithm hold hyperparameters which require tuning. The optimal set of parameters for each model on each individual dataset in $D$ is obtained by searching over a grid and stratified 10-fold cross-validation on the training sets obtained from each of the three datasets. The parameters and the grids over which is searched for each model are specified down below.

- **Random forest:**

  1. Max. features: Number of random features to consider at every split.
     Grid = $\{5, 6, 7, 8, 9, 10\}$
  2. N-estimators: Number of trees the forest contains.
     Grid = $\{300, 400, 500, 600, 700\}$
  3. Max. depth: Number of levels in each tree (i.e., number of splits).
     Grid = $\{4, 5, 6, 7, 8, 9, 10\}$

- **Gradient boosted trees:**

  1. Learning rate: Controls the contribution of each tree to the model.
     Grid = $\{0.01, 0.02, 0.03, 0.04, 0.05\}$
  2. N-estimators: Number of trees the forest contains.
     Grid = $\{300, 400, 500, 600, 700\}$

3. Max. depth: Number of levels in each tree (i.e., number of splits).
   Grid = $\{4, 5, 6, 7, 8, 9, 10\}$

- **RIPPER:**

1. k: Number of optimization iterations.
   Grid = $\{1, 2, 3\}$
2. Prune-size: Proportion of the training set utilized for pruning.
   Grid = $\{0.2, 0.03, 0.4, 0.5\}$
3. dl-allowance: Description length limit at which construction phase is terminated.
   Grid = $\{40, 50, 60, 70, 80, 90\}$

- **RUG**

1. Max. depth: Number of levels in each tree (i.e., number of splits).
   Grid = $\{4, 5, 6, 7, 8, 9, 10\}$
2. Pen_par: Penalty parameter $\lambda$.
   Grid = $\{0.001, 0.01, 0.1, 0.5\}$

The RUX and RuleFit will utilize the trained Random Forest and Gradient Boosted trees, with optimal hyperparemeters, to extract their rules. Thus, two distinct RUX models (RUX-RF & RUX-GB) and two distinct RuleFit models (RuF-RF & RuF-GB) will be trained on each of the three training sets obtained from the three datasets in $D$. Consequently, on each training set ten different models will be trained. Finally, to limit the number of rules of the RUX and RUG models, only the rules with an associated weight which is equal to or above 0.05 will be considered.

### 4.5 Performance measures

To assess the predictive performance of the applied methods, proper evaluation metrics are required. An important concept of classification performance is the confusion matrix, which is the basis for various evaluation measures. The confusion matrix, which holds four categories with regard to correctly or incorrectly predicted class labels, is displayed down below in table 5.

Table 5: Confusion matrix

| | Actual positive class | Actual negative class |
|---|---|---|
| Predicted positive class | True Positive | False positive |
| Predicted negative class | False negative | True negative |

In the context of this thesis, the positive class refers to converting journeys while the negative class refers to non-converting journeys. Accordingly, a true positive (TP) data instance denotes a customer journey which is correctly predicted to convert, while a false positive (FP) data instance denotes a customer journey which is falsely predicted to convert. Similarly, a true negative (TN) instance is correctly classified as a non-converting sequence and a false negative (FN) instance is incorrectly classified as a non-converting sequence, as it in fact holds the converting (positive) class label. From these four categories, the accuracy of a method can be determined. The accuracy of a method denotes the proportion of data instances which are assigned a class label which coincides with their actual class label:

$$Accuracy \ = \ \frac{TP + TN}{TP + TN + FP + FN} \tag{28}$$

However, within the MTA Context, the datasets leveraged typically exhibit class imbalance; i.e., non-converting customer journeys are over-represented while the converting journeys are under-represented. As previously specified in the data section, the preprocessed dataset has a conversion rate of 6.5%. As a result, solely stating the accuracy in case of a class imbalance may provide a distorted view of the performance of the model. To illustrate this, imagine a lazy-classifier which predicts the majority class for every data instance in the preprocessed dataset. Consequently, this classifier will still obtain an accuracy of 93.5% by only correctly classifying the majority class and in essence completely disregarding the minority class. Therefore, it is beneficial to consider other metrics which do not favor the majority class. Two of such metrics are precision and recall. Precision represents the fraction of instances correctly classified as the positive class over all the instances classified as positive by the classifier:

$$Precision \ = \ \frac{TP}{TP + FP} \tag{29}$$

In addition, recall denotes the fraction of positive classified instances over all the instances which have the positive class label:

$$Recall \ = \ \frac{TP}{TP + FN} \tag{30}$$

These two metrics can subsequently be combined to form the F1-score, which is the weighted harmonic mean of precision and recall and is denoted as follows:

$$F1 - score \ = \ \frac{(1 - \alpha) * precision * recall}{(\alpha * precision) + recall}, \ \alpha > 0 \tag{31}$$

Where $\alpha$ controls whether recall or precision is weighted more heavily. In this thesis $\alpha = 1$, which results in precision and recall having an equal weight (Japkowicz, 2013). Both the accuracy and F1-score will be reported for evaluating the predictive performance of the trained models.

## 4.6 Rule & Feature importance

To grasp the influence of different features on the predictions made by the various models, the feature and/or rule importance is determined. Because different types of machine learning models are leveraged in this thesis, different approaches are required for determining the most influential features and/or rules for the various model types.

### 4.6.1 Feature importance Bagged Logistic Regression

For the BLR, the learned coefficients from the input features can be utilized to represent the feature importance. The coefficients represent the average change in log-odds per one unit increase of the input features (Siegel and Wagner, 2022).

### 4.6.2 Feature importance tree-ensembles

To determine the feature importance for the Random Forest and Gradient Boosted trees, the Gini importance is leveraged. The Gini importance is defined as the normalized total reduction of Gini index yielded by a feature. First, the reduction in gini at each node $z$ in every tree is determined, which can be referred to as node importance and is denoted as:

$$ni_z = G_z - w_{left(z)}G_{left(z)} - w_{right(z)}C_{right(z)} \tag{32}$$

Where $G_z$ represents the Gini index of node $z$, $w_{left(z)}G_{left(z)}$ is the weighted Gini index of the left child node and $w_{right(z)}C_{right(z)}$ is the weighted Gini index of the right child node. For an individual decision tree, the Gini importance of feature $j$ is then computed by the weighted fractions of node importances in which feature $j$ was used:

$$g_j = \frac{\sum_{j \in J} I_{zj}\ ni_z}{\sum_{j \in J} ni_z} \tag{33}$$

Where $Z$ represents the set consisting of all the nodes and $I_{jz}$ is an indicator function which equals 1 if feature $j$ is utilized to split at node $z$ and otherwise 0. Finally, the Gini importance of feature $j$ in a tree ensemble is then obtained by normalizing $g_j$ and averaging over all the trees. The Gini importance values of the features within a single model accumulate to one, allowing for straight-forward examination of the most influential features within a model.

### 4.6.3 Feature importance rule-based methods

The various rule-based methods require different approaches of determining the importance of rules, which can be formulated based on the form of the decision rules these models output.

First, as discussed in both the literature review and methodology section, the RIPPER model by Cohen (1995) produces a decision set. This decision sets consists of rules which attempt to

filter out the minority class, i.e., the converting customer journeys, from the training data. A default rule is included to ensure the entire feature space is covered and the decision set is exhaustive (Cohen, 1995). To illustrate the rule-importance of the RIPPER algorithm, the five rules with the relative highest coverage on the test set will be displayed. The coverage on the test set is utilized, as it is arguably interesting to consider which rules had the greatest impact in making the predictions by the model.

On the other hand, the Classy model requires a different approach. The Classy model generates a decision list, which is a set of decision rules with a hierarchical structure. This has the result that *only* the first rule in the list which is applicable to a data instance, affects the prediction made for the data instance. Consequently, for the Classy model it is not appropriate to utilize the coverage, but instead the usage should be leveraged for inferring rule importance. As noted in the methodology section, the usage $U(S_k)$ of rule $r_k$ is the coverage of the rule $Cov(S_k)$, minus the coverages of the preceding rules (Lakkaraju et al., 2016). For the Classy model, the five rules with the relative highest usage on the test set will be displayed to demonstrate the most important rules with regard to the predictions made by the Classy model.

The RuleFit, RUG and RUX models output a decision set. However, these models also return weights or coefficients, specifying the importance of the produced rules. Friedman and Popescu (2008) have proposed an importance measure for decision rules with corresponding coefficients or weights signifying the importance of the rules. The rule importance measure proposed by Friedman and Popescu (2008) for these type of decision sets, is computed in the following manner:

$$Imp \ = \ |\beta_k| \ * \sqrt{rel.Cov(S_k)(1 - rel.Cov(S_k))} \tag{34}$$

Where $\beta_k$ is the coefficient associated to rule $r_k$ for the RuleFit models. For the RUX and RUG models, $\beta_k$ will be replaced by $w_k$ to denote the weights associated with each rule $r_k$ in these models. Additionally, $rel.Cov(S_k)$ is the relative coverage of rule $r_k$, which is denoted as:

$$rel.Cov(S_k) = \frac{1}{n} \sum_{i=1}^{n} I_i(r_k) \tag{35}$$

Where $I_i(r_k)$ equals 1 if rule $r_k$ is applicable to data instance $i$. As with the previous models, the rule importance for these models will be determined over the test set. Consequently, $n$ is the number of instances in the test set. An overview of how the rule-importance for each of the models is determined, is presented in table 6.

The importance of the rules will only be determined over dataset $d_2$, which consists of the features associated with the first 5 touchpoints of each customer journey. The reason for this, is to keep the number of tables limited. Finally, each of the relative rule importance measures are

Table 6: Overview of the rule-importance measure for each of the rule-based models.

| Model | Form of the generated rules | Rule importance measure |
|---|---|---|
| Ripper | Decision set | Relative coverage, $\frac{Cov(S_k)}{n}$ |
| Classy | Decision list | Relative usage, $\frac{U(S_k)}{n}$ |
| RUG | Decision set + weights | $Imp = \lvert w_k \rvert * \sqrt{rel.Cov(S_k)(1 - rel.Cov(S_k))}$ |
| RUX | Decision set + weights | $Imp = \lvert w_k \rvert * \sqrt{rel.Cov(S_k)(1 - rel.Cov(S_k))}$ |
| RuleFit | Decision set + coefficients | $Imp = \lvert \beta_k \rvert * \sqrt{rel.Cov(S_k)(1 - rel.Cov(S_k))}$ |

normalized to allow for easier comparison among the five displayed rules of each of the rule-based methods.

### 4.7 Quantifying interpretability of rule-based methods

The rule-based methods discussed in the methodology section are supposed to be intrinsically interpretable, given the number of rules and the number of conditions per rule are limited. In order to quantify the interpretability of the rule-based methods, the following metrics will be utilized:

- **Number of rules**: The number of rules in the rule list or rule set.
- **Average rule length**: The average number of conditions per rule.
- **Average number of rules per sample**: The average number of rules utilized to classify a data instance.

For both the RIPPER and Classy models, the default rule will be disregarded in the computation of these metrics. The number of rules and the average rule length are considered to be global interpretability metrics, while the average number of rules per sample is viewed as a local interpretability metric. These metrics will be determined for the rule-based methods that are trained on the training set obtained from the $d_2$ dataset.

## 5 Results

In this section, the results will be discussed. First the performances of each of the models on each distinct dataset will be evaluated, after which the feature importance of the BLR and tree-ensembles will be discussed. Then, the rule-importance of the rule-based methods will be considered. Finally, the interpretability metrics for the rule-based methods will be discussed.

## 5.1 Predictive Performances

Table 7, 8 and 9 respectively display the performance of each of the models on the test set of the data consisting of the features originating from the first three, five and seven touchpoints. The BLR appears to be the least model in terms of predictive performance on each test set. This may be due to the monotonicity implied by the BLR between the predictors and the outcome variable, which seems to not be the adequate manner of modelling the relationship between the features and outcome variable (Molnar, 2018).

Conversely, the Gradient Boosted trees emerge to be the best performing model in both metrics, although this status is shared for the test set consisting of the first seven touchpoints. In general, the tree-ensembles and rule-based methods are generally not extremely far apart in terms of score for both metrics. However, despite the fact that the accuracy of these models is generally high, the F-score for these models is considerably lower, signifying that the models are rather average at correctly classifying converting customer journeys.

Nevertheless, considering only the tree-ensembles and rule-based methods, the probabilistic rule list produced by the Classy model did the least in both metrics and has so consistently for all three test sets, although it matches the accuracy of the RUX-RF model for $d_2$ and both RUX models for $d_3$. Notably, both the RUX models produce worse than the models from which they extract their rules, the Random Forest and Gradient Boosted trees; especially the F1-score appears to be lower. Even so, the RuleFit models appear to be more comparable with the models they extract their rules from. Although the Random Forest and RuF-RF barely differentiate in terms of accuracy, the F1-score of the RuF-RF is consistently higher. The RuF-GB does better than the Random Forest and the RuF-RF regarding the F1-score for all three datasets. These three models are however more or less equivalent in terms of accuracy.

Moreover, the RUG algorithm also outperforms the RUX models in both metrics. Whilst RUG is outperformed by both RuF models concerning the F1-score and mostly accuracy, it equals the accuracy of the RuF-GB for $d_1$. It only matches the accuracy of the Random Forest for $d_1$ and other than that performs less than the Random Forest. Finally, the RIPPER model consistently does less well in terms of accuracy than the RUG, while it is better in terms of F1-score. RIPPER's accuracy is for $d_1$ in between that of the RUX models, although its accuracy value is better for the latter two datasets. RIPPER's accuracy is generally below that of the other rule-based models and tree-ensembles, besides Classy. However, RIPPER's F1-score is consistently higher than that of the the RUG and RUX models, though it is generally below that of the other tree-ensembles and RuleFit models.

Table 7: Accuracy- and F1-scores of the models on the test set obtained from $d_1$

| | BLR | RF | GB | RIPPER | Classy | RUG | RUX-RF | RUX-GB | RuF-RF | RuF-GB |
|---|---|---|---|---|---|---|---|---|---|---|
| Accuracy | 0.746 | 0.953 | **0.955** | 0.948 | 0.945 | 0.953 | 0.947 | 0.951 | 0.954 | 0.953 |
| F1-score | 0.340 | 0.541 | **0.560** | 0.539 | 0.499 | 0.528 | 0.512 | 0.520 | 0.542 | 0.553 |

Table 8: Accuracy- and F1-scores of the models on the test set obtained from $d_2$

| | BLR | RF | GB | RIPPER | Classy | RUG | RUX-RF | RUX-GB | RuF-RF | RuF-GB |
|---|---|---|---|---|---|---|---|---|---|---|
| Accuracy | 0.742 | 0.954 | **0.955** | 0.950 | 0.945 | 0.952 | 0.945 | 0.948 | 0.953 | 0.954 |
| F1-score | 0.337 | 0.547 | **0.562** | 0.542 | 0.501 | 0.529 | 0.520 | 0.523 | 0.552 | 0.560 |

Table 9: Accuracy- and F1-scores of the models on the test set obtained from $d_3$

| | BLR | RF | GB | RIPPER | Classy | RUG | RUX-RF | RUX-GB | RuF-RF | RuF-GB |
|---|---|---|---|---|---|---|---|---|---|---|
| Accuracy | 0.744 | **0.954** | **0.954** | 0.950 | 0.945 | 0.952 | 0.945 | 0.945 | **0.954** | **0.954** |
| F1-score | 0.338 | 0.538 | **0.562** | 0.542 | 0.501 | 0.535 | 0.515 | 0.521 | 0.553 | **0.562** |

## 5.2 Rule & Feature importance results

### 5.2.1 Bagged Logistic Regression coefficients

Table 10 shows the seven features with the largest absolute coefficient and the intercepts of each of the three BLRs. As mentioned in the data section, the numbers after the underscore in each of the feature names, denotes the touchpoint the feature is associated with. Notably, this top seven includes the same seven features for each of the BLRs and the order of these features is also the same across all three BLRs. Moreover, none of these variables have a negative coefficient. The coefficients represent the average increase (decrease) in the log-odds of converting, per one unit increase (decrease) of the input features. Apparently, the three Click features are associated

Table 10: Seven features with the highest absolute coefficients and the intercepts of the three Bagged Logistic Regressions.

| | $d_1$ | $d_2$ | $d_3$ |
|---|---|---|---|
| Click_1 | 3.6135 | 3.9450 | 3.5451 |
| Click_2 | 3.5257 | 3.6506 | 3.7504 |
| Click_3 | 0.7319 | 0.7450 | 0.7835 |
| Cat4_2 | 0.1423 | 0.1372 | 0.1364 |
| Cat4_3 | 0.0912 | 0.0918 | 0.0985 |
| Cat4_1 | 0.0815 | 0.0834 | 0.0864 |
| Cat1_3 | 0.0426 | 0.0416 | 0.0421 |
| Intercept | -5.3722 | -5.7558 | -5.8054 |

with the highest average change in log-odds per one unit change in these features. Accordingly, customer journeys along which the shown advertisements are clicked, are associated with a higher log-odds of conversion. However, this positive effect on the log-odds of clicked advertisements seems to drop considerable after the second touchpoint, as the coefficient of the 'Click_3' feature is considerable lower than that of the 'Click_1' and 'Click_2' features.

Besides the Click features, the Cat4 feature of the first, second and third touchpoints and the Cat1 feature of the third touchpoint are all also positively associated with the log-odds of converting across all three models, although the coefficients of these models appear to be substantially lower than those of the Click variables. All coefficients associated with the other variables not displayed in table 10 consist of absolute values converging even closer to zero.

### 5.2.2 Gini importance tree-ensembles

Table 11 and 12 display the five variables with the highest Gini importance values for each of the Random Forests and Gradient Boosted trees respectively. As previously mentioned in the methodology section, the Gini importance values associated with the features in a model accumulate to one. As can be viewed in both table 11 and 12, the accumulated Gini importance values of these five features alone is above 0.5 for both the Random Forests and Gradient Boosted trees trained on each of the datasets, signifying these five features alone have a great impact on the made predictions.

Remarkably, each of the Random Forests include the same features in the top five regarding the Gini importance, although the order differentiates across the models. Similar to the ranked absolute coefficients of the BLR in table 10, table 11 includes the Click and Cat4 features associated with the first two touchpoints. However, table 11 does not include the Click and Cat4 variables associated with the third touchpoint. Moreover, the Random Forests include the Cat1_2 variable consistently in fifth place in terms of Gini importance, while this variable is not included in table 10 regarding the highest absolute coefficients of the BLRs.

Table 11: Gini importance of the five most influential features of the three Random Forests.

| $d_1$ | | $d_2$ | | $d_3$ | |
|---|---|---|---|---|---|
| Click_1 | 0.193 | Click_2 | 0.161 | Click_1 | 0.163 |
| Cat4_2 | 0.188 | Click_1 | 0.155 | Click_2 | 0.162 |
| Click_2 | 0.171 | Cat4_2 | 0.150 | Cat4_2 | 0.130 |
| Cat4_1 | 0.088 | Cat4_1 | 0.079 | Cat4_1 | 0.081 |
| Cat1_2 | 0.081 | Cat1_2 | 0.077 | Cat1_2 | 0.060 |

Each of the Gradient Boosted trees include the same variables in terms of Gini importance, as can be viewed in table 12. However, whereas the Random Forests include the Cat4_1 variable,

table 12 does not hold this feature and instead incorporates the Cat1_1 feature. In general, the top-3 in terms of Gini importance for both the Random Forests and Gradient Boosted trees incorporate the same three variables: The Cat4_2, Click_1 and Click_2 features. Finally, the latter two positions of the top five is either occupied by the Cat1_1 and Cat4_1 features (Random Forests), or by the Cat1_1 and Cat1_2 features (Gradient Boosted trees).

Table 12: Gini importance of the five most influential features of the three Gradient Boosted trees.

| $d_1$ | | $d_2$ | | $d_3$ | |
|---|---|---|---|---|---|
| Cat4_2 | 0.370 | Cat4_2 | 0.317 | Cat4_2 | 0.255 |
| Click_1 | 0.179 | Click_2 | 0.163 | Click_1 | 0.134 |
| Click_2 | 0.166 | Click_1 | 0.128 | Click_2 | 0.119 |
| Cat1_1 | 0.068 | Cat1_2 | 0.069 | Cat1_2 | 0.066 |
| Cat1_2 | 0.057 | Cat1_1 | 0.044 | Cat1_1 | 0.038 |

#### 5.2.3 Rule-based methods' rule importance

In table 13, the five most important rules with regard to the relative coverage produced by the RIPPER algorithm are displayed. Note that RIPPER produces a decision set, which means these rules can be considered in any order and a data instance can leverage more than one rule, unlike with decision lists. Moreover, the produced rules by the RIPPER algorithm only seek to classify the minority class, i.e., converting customer journeys.

As can be viewed in table 13, each of the rules in the top five regarding relative coverage include the condition that the advertisement associated with the first touchpoint is clicked. As a clicked advertisement likely demonstrates interest in what is being advertised, it's not surprising that clicked advertisements are associated with converting customer journeys. Besides that, each of the rules also include at least one or more of the Cat variables. Specifically, these rules leverage the Cat1_1, Cat1_2, Cat2_2, Cat3_3 and Cat4_3 features by either requiring a specific level of these features, requiring these features to be above or below a specific threshold or by requiring these features to fall within a specific interval. Note that, in line with the accumulative $IDF$ transformation applied to the categorical features, values relatively close to zero are associated with high-frequency levels of the original categorical variables while higher values are associated with less frequent levels.

In table 14, the five most important rules in terms of relative usage of the rule list constructed by Classy can be viewed. The numbers between parentheses after the ranks denote the position of the rule in the rule list. The consequent comprises of the class distribution associated with each rule, wherein the class with the highest probability is the predicted class by that rule. The

Table 13: Most important rules produced by the RIPPER algorithm regarding the relative coverage based on the test set obtained from $d_2$

| Rank | Rule | Rel. Coverage |
|---|---|---|
| 1 | Click_1 = 1 & Cat1_2 = 4.97 | 0.487 |
| 2 | Click_1 = 1 & Cat1_1 = 4.97 | 0.471 |
| 3 | Click_1 = 1 & Cat1_2 = 7.36 & Cat1_1 = 2.78 & Cat3_3 $\leq$ 57.25 | 0.018 |
| 4 | Click_1 = 1 & Cat1_2 = 7.36 & Cat4_3 $\geq$ 0.13 & Cat2_2 $\in$ [2.25, 8.22] | 0.017 |
| 5 | Click_1 = 1 & Cat1_2 = 2.80 & Cat4_3 $\geq$ 0.13 & Cat1_1 = 0.72 | 0.007 |

advantage of this lies within the fact that not just the predictions are provided, but the class distributions also provide a sense of (un)certainty about the prediction being made (Proença and van Leeuwen, 2020).

Table 14: Most important rules produced by the Classy algorithm regarding the relative usage based on the test set obtained from $d_2$

| Rank | Rule | Consequent | Rel. Usage |
|---|---|---|---|
| 1 (72) | Click_1 = 1 | $P(0) = 1; P(1) = 0$ | 0.946 |
| 2 (63) | Cat1_1 $\in$ [2.78, 9.84) & Click_2 = 1 & Cat2_1 $\geq$ 1272.93 & Cat1_1 $\in$ [0.64, 11.85) | $P(0) = 0.756$; $P(1) = 0.244$ | 0.015 |
| 3 (67) | Cat1_1 $\in$ [2.80, 9.84) & Click_2 = 1 & Cat2_1 $\geq$ 1272.93 & Cat8_2 $<$ 13.33 & Cat6_2 $<$ 13.70 | $P(0) = 0.790$; $P(1) = 0.210$ | 0.013 |
| 4 (62) | Cat1_2 $\in$ [2.780, 9.84) & Click_1 = 0 & Campaign_1 $\in$ [101.19, 270.08] & Cat7_1 $<$ 9.53 & Cat5_2 $<$ 10.91 | $P(0) = 0.732$; $P(1) = 0.268$ | 0.013 |
| 5 (44) | Click_1 = 1 & Cat1_1 $\in$ [2.78, 9.84] & Cat6_1 $\in$ [4.62, 13.70] & Cat9_1 $<$ 6.24 | $P(0) = 0.587$; $P(1) = 0.413$ | 0.013 |

Alike to RIPPER, the rules include specific values, thresholds or intervals for the Campaign and Cat variables, indicting which combinations of values or ranges of these features are associated with the consequent of the rule.

Remarkably, the rule with the highest usage, associates Click_1 = 1 with a non-conversion, although a clicked advertisement would intuitively be associated with a conversion. This is likely due to the position of the rule, as this is the $72th$ rule in the rule list. Several rules preceding this rule also include a condition specifying one of the Click variables being equal to one, in combination with conditions including various Cat and Campaign features and a consequent favoring the converting class. Consequently, the converting journeys have already been filtered out and what remains are non-converting journeys along which the first advertisement was apparently clicked.

Generally, these five most important rules generated by the Classy algorithm all have a consequent with a higher probability of the non-converting class, with rules ranked two to five

holding non-zero probabilities for the converting class, signifying a higher uncertainty for these four rules.

Table 15 displays the most important rules generated by the RUG model with regard to the importance measure, which combines the relative coverage and the coefficient associated with the rule. The consequent represents the class to which the associated weight of the rule is attributed to. As stated in the methodology section, both the predictions made by the RUG and RUX models are based on the weighted combinations of the rule predictions (Lumadjeng et al., 2023). The first rule in table 15 is by far the most important, whereas the second to fifth rule are considerably less important than the first rule. In particular, the rules generated by the RUG model appear to have more conditions than the rules in the decision set produced by the RIPPER algorithm. This discrepancy in the number of conditions is due to the cross-validated tree depth of the decision tree from which the RUG generates its rules. As with the previous model, specific values or ranges of the Campaign and Cat variables are included. As these ranges and values can be mapped back to their original categorical levels, it is possible to deduce what combination of categorical levels of these features is associated with which class label and corresponding weight.

Table 15: Most important rules produced by the RUG algorithm regarding the importance measure based on the test set obtained from $d_2$

| Rank | Rule | Class | Rel. *Imp* |
|---|---|---|---|
| 1 | Click_1 = 1 & Cat4_1 $\leq$ 18.76 & Cat9_3 $>$ 24.43 & Cat4_3 $>$ 1.44 & Cat4_2 $\leq$ 1.44 | Y = 1 | 0.511 |
| 2 | Cat9_2 $\leq$ 78.10 & Cat9_5 $\leq$ 58.11 & Cat7_2 $\leq$ 129.99 & Cat5_4 $\leq$ 2.01 & Cat2_5 $>$ 3.58 | Y = 0 | 0.147 |
| 3 | Cat2_1 $\leq$ 12.21 & Cat7_2 $>$ 41.10 & Cat6_2 $>$ 53.19 & Cat1_2 $>$ 1.76 & Cat1_1 $>$ 6.17 | Y = 0 | 0.136 |
| 4 | Cat7_1 $>$ 31.86 & Cat5_5 $>$ 6.17 & Cat7_4 $\leq$ 26.34 Cat4_1 $>$ 1.44 & Cat1_2 $>$ 3.88 | Y = 0 | 0.105 |
| 5 | Cat6_3 $>$ 31.86 & Cat5_5 $\leq$ 30.89 & Cat7_4 $>$ 6406.56 & Cat1_1 $\leq$ 8.60 & Campaign_2 $>$ 8.07 | Y = 0 | 0.101 |

The most five most important rules in terms of the importance measure of the RUXRF and RUXGB models are displayed respectively in table 16 and 17. Notably, the five rules of the RUXRF model consists of more conditions than the five rules of the RUXGB model. The five rules of the RUXGB model also contain less conditions than the rules of the RUG model and seem to have a number of conditions per rule which is comparable to that of the five rules of the RIPPER model in table 13.

Finally, five most important rules in terms of the importance measure of the RuF-RF and RuF-GB models are presented in table 18 and 19 respectively. The five most important rules produced by both these models appear to have substantially less conditions than the RUX-RF and RUX-GB models, which is noteworthy as these methods leverage the same tree-ensembles to extract rules from. Moreover, the displayed rules produced by the RuleFit models in general contain the least conditions among all models, the RuF-RF rules having at most four conditions

Table 16: Most important rules produced by the RUX-RF algorithm regarding the importance measure based on the test set obtained from $d_2$

| Rank | Rule | Class | Rel. *Imp* |
|---|---|---|---|
| 1 | Cat1_1 > 11.35 & Cat7_2 & Cat8_1 > 3.53 & Cat9_2 ≤ 11.64 & Cat3_2 ≤ 4.93 & Cat4_2 ≤ 1.44 & Click_1 = 1 | Y = 1 | 0.787 |
| 2 | Click_1 = 1 Campaign_2 ∈ (1189.17, 1952,94] & Cat9_2 > 6.35 & Cat9_1 > 18.96 & Cat2_1 > 8.60 & Cat6_2 ≤ 1.77 & Cat1_2 ≤ 1.76 | Y = 1 | 0.066 |
| 3 | Click_1 = 1 & Cat3_2 > 11908.26 & Click_1 = 1 & Cat1_1 ≤ 1.76 & Cat5_2 ≤ 1.44 | Y = 1 | 0.061 |
| 4 | Cat1_1 ≤ 6.17 & Cat3_7 > 4.20 & Cat3_3 ∈ (38.51, 2442.11] & Cat5_2 ≤ 2.01 & Cat2_1 ≤ 8.60 & Campaign_3 > 1.76 | Y = 0 | 0.048 |
| 5 | Cat9_2 ≤ 104.40 & Campaing_2 ≤ 1196.00 & Cat4_1 ≤ 1161.51 & Cat4_3 > 1148.08 & Cat7_1 > 4.20 & Cat2_1 > 3.58 | Y = 0 | 0.038 |

Table 17: Most important rules produced by the RUX-GB algorithm regarding the importance measure based on the test set obtained from $d_2$

| Rank | Rule | Class | Rel. *Imp* |
|---|---|---|---|
| 1 | Click_1 = 0 | Y = 0 | 0.784 |
| 2 | Cat4_2 > 1.44 & Cat2_2 > 3.88 & Cat5_2 > 1.44 & Click_2 = 1 | Y = 1 | 0.067 |
| 3 | Cat4_2 > 1.44 & Cat6_2 ≤ 3070.42 & Cat7_2 ≤ 4.67 | Y = 0 | 0.061 |
| 4 | Cat7_2 ≤ 4.76 & Cat1_2 > 3.88 & Cat6_2 > 1.44 & Click_2 = 1 | Y = 1 | 0.047 |
| 5 | Cat4_2 > 4.67 & Cat3_2 ∈ (4.67, 8.91] | Y = 0 | 0.041 |

and the RuF-GB rules even consisting of at most two conditions.

Table 18: Most important rules produced by the RuF-RF algorithm regarding the importance measure based on the test set obtained from $d_2$

| Rank | Rule | Effect on conversion | Rel. *Imp* |
|---|---|---|---|
| 1 | Click_2 = 0 | Negative | 0.439 |
| 2 | Click_1 = 0 & Click_4 = 0 | Negative | 0.210 |
| 3 | Click_2 = 0 & Click_3 = 0 | Negative | 0.152 |
| 4 | Click_1 = 0 | Negative | 0.107 |
| 5 | Click_1 = 1 & Cat4_1 ≤ 1.44 & Cat4_2 ≤ 1.44 & Cat4_3 ≤ 1.44 | Negative | 0.092 |

Table 19: Most important rules produced by the RuF-GB algorithm regarding the importance measure based on the test set obtained from $d_2$

| Rank | Rule | Effect on conversion | Rel. *Imp* |
|---|---|---|---|
| 1 | Click_1 = 0 | Negative | 0.290 |
| 2 | Click_2 = 0 | Negative | 0.238 |
| 3 | Click_1 = 0 & Cat9_2 ≤ 92.87 | Negative | 0.190 |
| 4 | Click_2 = 0 & Cat9_2 ≤ 92.87 | Negative | 0.170 |
| 5 | Cat1_2 ≤ 8.60 & Cat3_3 ≤ 11474.93 | Negative | 0.112 |

In general, the above rules can help to infer what combinations of features linked with specific touchpoints are associated with either converting or non-converting customer journeys. For instance, although a clicked advertisement would intuitively be associated with an increase likelihood of conversion, the rules above display occasions of clicked advertisments with particular Campaign and Cat values or intervals for which this is not the case.

### 5.3 Interpretability metrics for rule-based methods

In table 20, the interpretability metrics of the rule-based methods can be viewed. The RIPPER model produces the least number of rules, while the number of rules produced by the Classy model is also relatively limited compared to the other models. The average number of rules per sample of these two models is below one, because the produced rules by these models do not cover all the data instances and the default rule of these models is not taken into account.

Whereas the number of rules of the RUG model is also relatively limited, the number of rules of the RUX and RuleFit models is considerable higher. Especially the number of rules produced by the RUX-RF is extensive, while the average rule length of these rules is also the largest. On the other hand, the RUX-GB produced considerably fewer and shorter rules than the RUX-RF.

Notably, the average number of rules per sample for the RuleFit models are substantially higher than that of the other models, while the total number of rules generated by these models is also relatively high, especially for the RuF-GB. Remarkably, the average rule length of the RuF-GB is the lowest.

Table 20: Interpretability measures for the rule-based methods trained on the training set obtained from the $d_2$ dataset

| | Number of rules | Avg. rule length | Avg. rules per sample |
|---|---|---|---|
| RIPPER | 30 | 4.83 | 0.08 |
| Classy | 73 | 4.88 | 0.32 |
| RUG | 91 | 4.26 | 1.75 |
| RUX-RF | 2713 | 8.52 | 1.92 |
| RUX-GB | 156 | 4.77 | 2.06 |
| RuF-RF | 193 | 4.00 | 28.92 |
| RuF-GB | 502 | 3.24 | 70.78 |

In general, it appears RIPPER, Classy and RUG strike the best balance in term of these three metrics, although the average number of rules per sample of the former two is low because their decision rules are not exhaustive. Moreover, the RUX-RF is rather hard to interpret in global terms, due to its high number of rules and average rule length. Finally, both the RuleFit models seem to be mediocre in terms of local intepretability, as their average number of rules

per sample is excessive, especially for the RuF-GB model.

# 6 Conclusion

This thesis attempts to fill a gap in the current literature on conversion and attribution modelling by introducing rule-based methods to the MTA context. Intrinsically interpretable models, such as rule-based methods, and interpretability in general are relevant topics in the literature currently. Specifically, Dalessandro et al. (2012) have suggested interpretability should be viewed as an important characteristic of methodologies in the MTA. Intuitively, models which are straightforward to interpret should allow marketers and practitioners, who do not necessarily have knowledge on machine learning, to easily infer what features drive (non-)converting customer journeys.

To this end, seven rule-based methods have been leveraged for conversion prediction and attribution on the Criteo attribution dataset. These methods have been compared to a BLR, which was introduced as one of the earliest data-driven MTA models by Shao and Li (2011), and a Random Forest and Gradient Boosted trees. The latter two models, which fall in the category of tree ensembles, are considered black-boxes and hence not inherently interpretable, although post-hoc methods such as the Shapley Value (Shapley, 1953) can be applied to these models in order to credit touchpoints or features for conversions along customer journeys.

In terms of predictive performance, the BLR performed the worse. Whereas Shao and Li (2011) used data which consists of the counts a channel was visited along a customer journey, the input data of the Criteo dataset possesses a high level of cardinality in various features. To facilitate the data, the accumulative $IDF$ transformation was applied to reduce the cardinality. As the BLR assumes monotonicity between the input features and the output variable. This assumption is likely not suitable for the $IDF$ transformed features. The rule-based methods were in terms of predictive performance more comparable to the ensemble-methods, although the Classy algorithm scored considerable worse than the tree-ensembles.

Regarding interpretation of feature importance, the coefficients of the BLR were examined to this end. Although these are straightforward to interpret, the interpretation of the coefficients come with the ceteris paribus clause; all else remaining equal. This clause puts a limitation on the interpretation, as this clause needs to be kept in mind. The tree ensembles requires a post-hoc method to inspect the importance of variables. Therefore, Gini importance is utilized, which outputs an aggregated overview of the most important features. Although the Gini importance is low in computational cost, the resulting overview does not provide information on whether features are positively or negatively related to the output variable. For the rule-based methods, the five most important rules regarding importance measures fitting for each of the models were

displayed. Although this only allows for the comprehension of a part of the model, the displayed rules and their consequences are generally easy to understand, even though the consequences can differ among the various rule-based methods applied.

The output of the Classy algorithm differentiates from the other rule-based methods in the sense that the former generates a decision list, whereas the latter models output a decision set. Displaying the five most important rules of the Classy algorithm is less interpretable, as one would need to also consider the rules preceding these rules, resulting in decision rules which are less intuitive than those produced by the other rule-based methods. However, the consequent associated with each rule includes a class distribution, providing a(n) (un)certainty of the prediction being made. In general, comparing the probabilistic rule list produced by Classy to the rule sets of the other rule-based methods is not straightforward.

The rule-based methods utilized in this thesis generally provide businesses with decision rules which are straightforward to interpret and provide information with regard to what features are associated with (non-)converting customer journeys. As a result, these methods can provide businesses with insights on what combinations of feature values associated with the touchpoints should be utilized in an attempt to increase conversions. Although the rule-based methods do not provide an exact credit attribution among the features or touchpoints, the rule-based methods can instead be utilized to perform an initial analysis on what combinations of feature values are associated with (non-)converting customer journeys.

By considering the features associated with each of the 3-, 5- and 7 touchpoints, an attempt was made at incorporating the sequential nature of the data. To ensure each of the entries in the individual datasets have the same number of features, padding was used. As most of the customer journeys consist of no more than three touchpoints, the features associated with the touchpoints beyond the third touchpoint may have been reduced to noise. This notion is also visible in the rule- and feature importance of the models, as these primarily consist of features associated with the first three touchpoints. Although the sequential nature of the customer journeys was included to some extent, other models used in the MTA context (e.g., LSTM-RNN based models) are able to integrate the sequential nature in its entirety.

In future research, considering other manners to incorporate the sequential nature of customer journeys may be beneficial. Moreover, most features of the Criteo dataset used in this thesis, have a high cardinality. It may be interesting to consider a dataset which consists of a smaller number of touchpoints, a smaller number of features associated with each touchpoint or features with lower cardinality, so that a data transformation method, such as the accumulative $IDF$, is not a necessity. Finally, methods to alleviate the class imbalance could be utilized in future research, although such techniques could either result in parts of the datasets being ignored or assumptions being made which might not be valid for the conversion prediction and

attribution context.

# Bibliography


V. Abhishek, P. Fader, and K. Hosanagar. The long road to online conversion: A model of multi-channel attribution. *SSRN Electronic Journal*, 2012.

E. Anderl, I. Becker, F. von Wangenheim, and J. H. Schumann. Mapping the customer journey: Lessons learned from graph-based online attribution modeling. *International Journal of Research in Marketing*, 33(3), sep 2016.

A. B. Arrieta, N. Díaz-Rodríguez, J. D. Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, and F. Herrera. Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58, jun 2020.

L. Breiman, J. Friedman, C. J. Stone, and R. Olshen. *Classification and Regression Trees*. Chapman and Hall/CRC, 1984.

G. O. Campos, A. Zimek, J. Sander, R. J. G. B. Campello, B. Micenková, E. Schubert, I. Assent, and M. E. Houle. On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. *Data Mining and Knowledge Discovery*, 30(4), jan 2016.

D. Chaffey and F. Ellis-Chadwick. *Digital Marketing: Strategy, implementation and practice 7th edition*. Pearson, 2019.

F. Chierichetti, R. Kumar, P. Raghavan, and T. Sarlos. Are web users really markovian? In *Proceedings of the 21st international conference on World Wide Web*, apr 2012.

W. W. Cohen. Fast effective rule induction. In *Machine Learning Proceedings, 1995*. 1995.

B. Dalessandro, C. Perlich, O. Stitelman, and F. Provost. Causally motivated attribution for online advertising. In *Proceedings of the Sixth International Workshop on Data Mining for Online Advertising and Internet Economy*, aug 2012.

E. de Haan, T. Wiesel, and K. Pauwels. The effectiveness of different forms of online advertising for purchase conversion in a multiple-channel attribution framework. *International Journal of Research in Marketing*, 33(3), sep 2016.

Deloitte and VIA. Digital advertising spend study, 2022. URL https://view.deloitte.nl/TMT-AdSpendStudy.html.

E. Diemert, J. Meynet, P. Galland, and D. Lefortier. Attribution modeling increases efficiency of bidding in display advertising. 2017. URL https://arxiv.org/pdf/1707.06409.pdf.

F. Doshi-Velez and B. Kim. Towards a rigorous science of interpretable machine learning. 2017.

R. Du, Y. Zhong, H. Nair, B. Cui, and R. Shou. Causally driven incremental multi touch attribution using a recurrent neural network. 2019. URL https://arxiv.org/pdf/1902.00215.pdf.

EU. Gdpr. *Official Journal of the European Union*, 2016.

T. Feldman. *An introduction to Digital Media*. Routledge, 1996.

J. H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5), oct 2001.

J. H. Friedman and B. E. Popescu. Predictive learning via rule ensembles. *The Annals of Applied Statistics*, 2(3), sep 2008. doi: 10.1214/07-aoas148. URL https://doi.org/10.1214%2F07-aoas148.

J. Fürnkranz and G. Widmer. Incremental reduced error pruning. In *Machine Learning Proceedings 1994*. 1994.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

P. D. Grünwald. *The Minimum Description Length Principle*. 2007.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2009.

G. James, D. Witten, T. Hastie, and R. Tibshirani. *An Introduction to Statistical Learning*. Springer, 2 edition, 2021.

N. Japkowicz. *Assessment Metrics for Imbalanced Learning*, chapter 8, pages 187–206. Wiley, jun 2013.

P. Kannan, W. Reinartz, and P. C. Verhoef. The path to purchase and attribution modeling: Introduction to special section. *International Journal of Research in Marketing*, 33(3), sep 2016.

P. K. Kannan and H. Li. Digital marketing: A framework, review and research agenda. *International Journal of Research in Marketing*, 34(1), mar 2017.

J. Keilson. *Markov Chain Models - Rarity And Exponentiality*. 1979.

B. Kim, R. Khanna, and O. Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. *Advances in Neural Information Processing Systems*, 29, 2016.

S. Kumar, G. Gupta, R. Prasad, A. Chatterjee, L. Vig, and G. Shroff. Camta: Causal attention model for multi-touch attribution. Dec 2020. URL https://arxiv.org/pdf/2012.11403.pdf.

H. Lakkaraju, S. H. Bach, and J. Leskovec. Interpretable decision sets. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, aug 2016.

K. N. Lemon and P. C. Verhoef. Understanding customer experience throughout the customer journey. *Journal of Marketing*, 80(6), nov 2016.

H. Li and P. K. Kannan. Attributing conversions in a multichannel online marketing environment: An empirical model and a field experiment. *Journal of Marketing Research*, 51(1), feb 2014.

A. C. Lumadjeng, T. Röber, M. H. Akyüz, and İlker Birbil. Rule generation for classification: Scalability, interpretability, and fairness. 2023.

B. Madhu and V. Deepak. A critical review of digital marketing. *International Journal of Management, IT Engineering*, 8(10), oct 2018. URL https://papers.ssrn.com/sol3/Delivery.cfm?abstractid=3545505.

C. McNair. Global advertisement spending forecast, may 2018.

T. Miller. Explanation in artificial intelligence: Insights from the social sciences. 2017.

G. Mita. Toward interpretable machine learning, with applications to large-scale industrial systems data. 2021. URL https://theses.hal.science/tel-03467524/document.

C. Molnar. *Interpretable machine learning: A guide for making black box models explainable.* 2018.

P. J. Pereira, P. Pinto, R. Mendes, P. Cortez, and A. Moreau. Using neuroevolution for predicting mobile marketing conversion. *Progress in Artificial Intelligence*, Aug 2019.

P. J. Pereira, P. Cortez, and R. Mendes. Multi-objective grammatical evolution of decision trees for mobile marketing user conversion prediction. *Expert Systems with Applications*, 168, apr 2021.

H. M. Proença and M. van Leeuwen. Interpretable multiclass classification by MDL-based rule lists. *Information Sciences*, 512, feb 2020.

K. Raman, M. K. Mantrala, S. Sridhar, and Y. E. Tang. Optimal resource allocation with time-varying marketing effectiveness, margins and costs. *Journal of Interactive Marketing*, 26, feb 2012.

K. Ren, Y. Fang, W. Zhang, S. Liu, J. Li, Y. Zhang, Y. Yu, and J. Wang. Learning multi-touch conversion attribution with dual-attention mechanisms for online advertising. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, oct 2018.

J. Rissanen. A universal prior for integers and estimation by minimum description length. *The Annals of Statistics*, 11(2), jun 1983.

X. Shao and L. Li. Data-driven multi-touch attribution models. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, aug 2011.

L. S. Shapley. A value for n-person games. *Contributions to the Theory of Games volume II*, 1953.

A. F. Siegel and M. R. Wagner. Multiple regression. In *Practical Business Statistics*, chapter 12, pages 371–431. 2022.

D. Yang, K. Dyer, and S. Wang. Interpretable deep learning model for online multi-touch attribution. 2020.

D. Yao, C. Gong, L. Zhang, S. Chen, and J. Bi. Causalmta: Eliminating the user confounding bias for causal multi-touch attribution. Dec 2021. URL https://arxiv.org/pdf/2201.00689.pdf.