The Result of Advertising Channels in Impulsive, Balanced and Considered Journeys

ERASMUS UNIVERSITY ROTTERDAM
Erasmus School of Economics
Bachelor Thesis [Business Analytics and Quantitative Marketing]

Linde Bakens
Student ID number: 468605

Supervisor: K. Gruber
Second assessor: F.J.L. van Maasakkers

30th of June 2022

Abstract

This paper uses eight models to research what important advertising channels are. A logit model and three heuristics are used, namely first-, last- and linear-click. Those four models are used as a benchmark for comparison with Markov models. The Markov models range from first- to higher-order. The paper distinguishes the types of customer journeys: impulsive, balanced and considered. The distinction is made in two ways: using the length of the customer journey and the number of channels in the journey. Two channels are the most important across the whole dataset and for the types of journeys. Impulsive journeys have the same most important channel as the entire dataset. In contrast, balanced and considered journeys also have another important channel.

Contents

1 Introduction **1**

2 Literature Review **2**

3 Data **5**

4 Methodology **8**

4.1 Heuristics 8

4.2 Logit Model 8

4.3 Markov graphs 9

4.3.1 Markov property 10

4.3.2 Removal effect 11

4.4 ROC curves 11

4.5 Types of Customer Journeys 12

4.5.1 Length of Journeys 13

4.5.2 Number of Channels 13

5 Results **14**

5.1 Full dataset 14

5.1.1 Predictive Accuracy 14

5.1.2 Robustness 16

5.1.3 Attribution Results 16

5.1.4 Transition Matrix and Probabilities 18

5.2 Length of Journeys 19

5.2.1 Predictive Accuracy 19

5.2.2 Robustness 20

5.2.3 Attribution Results 20

5.3 Number of Channels 21

5.3.1 Predictive Accuracy 21

5.3.2 Robustness 22

5.3.3 Attribution Results 22

6 Conclusion **23**

7 Discussion **25**

References **27**

A Appendix **31**

A.1 Table Appendix 31

A.2 Figure Appendix 38

A.3 Code Appendix 44

1 Introduction

There are various ways how companies try to convince potential customers to buy their items. Think of a billboard on the streets, an advertisement in a newspaper or a mail from a company. All three are examples of advertising which can be explained as providing information to persuade, remind or motivate consumers (Ratliff & Rubinfeld, 2010). Online advertising is on the rise, whereas offline advertising used to be the most popular way of advertising. Online advertising expenses increased over the past years in the United States. In 1998 the expenses were $1.8 billion, they grew to $20 billion in 2007. Thus, the expenses multiplied by ten within nine years. The growth of the internet is one of the reasons why online advertising is growing so fast (Ratliff & Rubinfeld, 2010). Besides, online advertising is cheaper than offline advertising (Goldfarb, 2014). Therefore, companies may decide to switch partly or entirely to online advertising. The most significant advantage of the internet is that targeting on the individual level is possible and that there is direct communication and feedback (Barbu, Ponea, & Bogdănoiu, 2019). Also, the reach of the internet is one of the advantages since it can reach customers on both a national and global level (Hanekom & Scriven, 2002).

There are many ways how online advertising occurs, for example, via e-mail marketing or as an advertisement on social media. Those two are examples of advertising channels which promote something to potential customers (*Advertising channel meaning, importance, factors amp; example*, n.d.). A customer follows a journey consisting of one or more channels before making a conversion decision: a sign-up or a purchase. Hence, the customer journeys come across one or multiple advertising channels, which the individual uses to choose whether to make a purchase or not.

This paper uses a dataset which contains information on 10,000 customer journeys (*Markov model for online multi-channel attribution [R package channelattribution version 2.0.5]*, 2022). For each customer journey, information is available on the channels' sequence and the conversion decision. From the dataset, it is evident that there is variation in the journey lengths and the number of channels in the journeys across observations. The shortest journey has a length of one, and the longest journey consists of 89 clicks. The number of channels varies from one to eight.

Because of the variation, it is quite probable that the journeys are not all the same. Therefore, the journeys are split into three groups: impulsive, balanced and considered. The distinction between the three types of journeys will be based either on the length of the journey or on the number of channels in the journey. When the length of journeys makes the differentiation between groups, impulsive journeys will be the shortest ones, balanced

journeys are of a medium length and considered journeys are the longest. Second, the number of channels can determine the kind of the journey. Impulsive journeys contain the least number of channels, and considered journeys contain the most. The group in the middle consists of balanced journeys.

It will be interesting to see whether the subgroups behave differently from the complete dataset. Therefore, results for both the whole dataset and each group are produced. The results come from eight models. First, several heuristics will be applied. Two single-touch attribution models, first- and last-click, and one multiple-touch attribution model, linear-click, are discussed. The three heuristics and a logit model are used as benchmark models. These models are compared with some more difficult models, namely first- and higher-order Markov graphs. All eight models will be used to conclude the importance of the channels.

Two channels seem the most important since they have the highest removal effect and attribution results. The full dataset finds one channel the most important across all models. All impulsive journeys and the balanced journeys based on the number of channels find the same results. Different results arise when considered journeys are based on the number of channels or for all considered journeys. These journeys find that the most crucial channel in all previously mentioned results is only the most important using last-click for those channels. For first-click, linear-click and the fourth-order Markov model, the other channel is the most important.

This paper continues with a discussion of the existing literature, which discusses the various channels, the models and the split between the three groups. After that, the dataset, the journeys' length, and the number of different channels within the journeys are discussed. The methodology discusses three heuristics, Markov graphs and the logit model. This section also discusses the Receiver Operating Characteristic (ROC) curve and the two corresponding measures. The split of the groups using the length and the number of channels is also explained in the methodology. Then, the results are presented for the whole dataset and subgroups. The predictive accuracy and robustness will be discussed before heading to the attribution results of the specific channels. Furthermore, the conclusion summarizes the results of the paper. The final section discusses the limitations of this research and provides suggestions for further research.

2 Literature Review

As discussed before, numerous channels exist for online advertising. Previous studies have mentioned several online advertising channels. An overview and description of those channels are shown in table A1. Table A2 shows which papers discuss which types of online

advertising in their paper. Anderl, Becker, Von Wangenheim, and Schumann (2016) discuss eleven different channels, namely type-in, search engine advertising (SEA), search engine optimization (SEO), price comparison, display, newsletter, retargeting, social media, affiliate, referrer and other. A distinction between firm-initiated and customer-initiated channels can be made (Wiesel, Pauwels, & Arts, 2011). Firm-initiated channels are the channels where the advertiser determines when and where the advertising takes place. The customer-initiated channels result from customers' actions, such as typing in a search word in a search engine. Anderl, Becker, et al. (2016) found that the attribution results for customer-initiated channels were higher compared to firm-initiated channels across four datasets. Hence, those channels are expected to be more effective. The channels type-in, SEA, SEO and price comparison are customer-initiated channels. Thus, they should be more effective. The following four are firm-initiated, and for the resulting three, we cannot say to which group they belong. Mei, Hua, Yang, and Li (2007) introduce another type of online advertising: video advertising. Partner website is also a type of online advertising (Anderl, Schumann, & Kunz, 2016).

The effectiveness of channels is interesting to research such that advertising can be done most efficiently. First- and last-click attribution are two examples of applied heuristics. First-click gives all the credit for the conversion to the advertisement that is clicked on first (H. Li, Kannan, Viswanathan, & Pani, 2016). First-touch assigns much credit to the search channel. This is logical since a customer who directly searches for a product is more likely to be interested in the product. For last-click, the last advertisement is essential. Thus, the difference depends on which advertisement gets the credit for the conversion. Those single-click attribution approaches assign all the value to one channel and no value to assisting channels (Anderson & Cheng, 2017). The multi-touch attribution model used in this paper is linear-touch which assigns the same credit to all the channels that occur in the customer journey (Ji, Wang, & Zhang, 2016).

Anderl, Becker, et al. (2016) introduce a new attribution framework to evaluate the effectiveness and interplay of channels. They propose to use a graph-based Markovian framework to research customer journeys. Those models have been used in the marketing field, and the number of papers on this topic increased (Harary & Lipstein, 1962). Research about brand loyalty and buying habits used the Markovian framework (Harary & Lipstein, 1962; Styan & Smith Jr, 1964). The Markov chains can show dependencies between different channels in a customer journey. First-order Markov models state that the present depends only on the previous channel (Anderl, Becker, et al., 2016). Furthermore, Archak, Mirrokni, and Muthukrishnan (2010) propose a higher-order model where the present depends on more channels in the past. Anderl, Becker, et al. (2016) also used two logit models in their paper. They used the heuristics and the logit models as benchmark models to compare against

the proposed Markovian framework. In their paper higher-order Markov models outperform first-click, last-click and the simple logit model. A logit model that includes order effects performs the same as higher-order Markov models.

As discussed, the types of journeys are based on the length of journeys or on the number of channels present in the journeys. Hence, the length and the number of channels need to be known to distinguish between the journeys. A distinction between customer journeys can be made, and three types of journeys emerge (Wolny & Charoensuksai, 2014). The first type is an *impulsive journey* where customers spend not much time searching for information on the product. The customers make an impulsive or emotional choice when deciding whether to buy the product or not. Their opinion can be influenced easily by their mood, seeing a product on display, previous experience or friends' opinion, for example. *Balanced journeys* are the second type of customer journeys. Customers exhibit an extended search for information and evaluation. Cognitive evaluation is used to decide on making the purchase or not. Traditional and digital media can trigger those journeys. What can characterize those journeys is that the customers often do research using different sources across channels. Lastly, *considered journeys* are the most elaborated journeys before deciding on the purchase. Those journeys have a large pre-shopping stage where customers look for information from several sources such as news, product reviews, blogs and friends. This information is crucial when someone wants to make a purchase.

Previous research states that it is challenging to determine the length of a customer journey (J. Li, Abbasi, Cheema, & Abraham, 2020). An example of a problem they mention is that only online purchases are recorded. Hence, offline purchases are not taken into account. Albrecht (2002) explains that cookies can recognize users when they return to a website. However, this is only possible if the user agrees with the cookies. When this is not the case, this delivers a problem for determining the customer journey of that individual. Furthermore, it is difficult to determine the length when a user uses several devices because the customer journey cannot be measured across devices. Hence, the length of customer journeys is not always easy to measure. Thus this paper contributes to existing research since the length of the journeys is known in the dataset. The research on the journey length and attribution is related to customer heterogeneity which is a common topic in marketing. Therefore, it is interesting to research.

Furthermore, research about the interaction between specific channels already exists (Anderl, Becker, et al., 2016). Lemon and Verhoef (2016) state that the rise of new channels, for example, mobile channels, adds complexity to the journeys. The rise of new channels can imply that there are more channels in the journey or that those new channels replace old channels. The rise in the number of available channels makes it easier to get the message

to a big group. However, it becomes more challenging to capture attention to the message (Berte & Gysels, 2007). Hence, there is enough research about the number of channels for advertising or the interaction of channels. However, it is refreshing to research the number of channels in customer journeys, which this paper does.

As discussed in the previous paragraph, the number of channels increased over time. This increase resulted in multichannel marketing, defined as the "design, deployment, coordination, and evaluation of channels to enhance customer value through effective customer acquisition, retention, and development" (Neslin, Grewal, et al., 2006). Consumers interact with firms through online and offline media channels due to the rise of the internet (Cui et al., 2021). This led to omnichannel marketing, which emphasizes a unified customer experience. Cui et al. (2021) define omnichannel as "the synergistic management of all customer touchpoints and channels both internal and external to the firm to ensure that the customer experience across channels as well as firm-side marketing activity, including marketing-mix and marketing communication (owned, paid, and earned), is optimized for both firms and their customers".

However, optimization is difficult, leading to the omnichannel problem. Cui et al. (2021) mention three challenges in their paper. The first one are data challenges. Information is needed on the interaction with the customer for the whole customer journey. However, the data might not always be accessible. Second, marketing attribution challenges arise. The effect of channels needs to be known and thus the result of the spending on marketing. However, this information is not always available, so the second challenge arises. The final challenge is the customer privacy challenge, where the balance between obtaining data and the infringement of customers' privacy must be found.

3 Data

The dataset contains 10,000 observations, and each observation has four variables (*Markov model for online multi-channel attribution [R package channelattribution version 2.0.5]*, 2022). The first variable 'path' indicates the customer journey and shows the channels in the journey in order. The states have been made anonymous and are represented as greek letters. This is depicted in equation 1.

$$C = \{eta, iota, alpha, beta, theta, lambda, kappa, zeta, epsilon, gamma, delta, mi\} \quad (1)$$

The other three variables will be explained briefly, and the descriptive statistics of those

variables can be found in table 1. The variable 'total_conversions' explains how many conversions occurred following that journey. The mean of this variable is 1.978, implying that, on average, 1.978 conversions occur per journey. The maximum is 4,197. In the same way, the variable 'total_null' shows the total amount of times the journey resulted in the null state, which can imply no transaction or a transaction that eventually ended in the null state. The minimum is 2, the mean 6.86 and the maximum 14,413. The journeys that did not end up in a conversion are thus the difference between total_null and total_conversions. The last variable in the dataset is 'total_conversion_value', which indicates the total monetary value. It is possible to assign different values to different conversions since some are worth more than others. This variable will be able to show the total value driven by online advertising across all conversions. Here the mean is 7.48, and the maximum is 12,452.

Table 1: Descriptive Statistics

Variable	min	mean	max
Total conversions	0	1.978	4,197
Total conversion value	0	7.480	12,452
Total null	2	6.860	14,413

Some channels occur more often in customer journeys than others. In table 2 you can see how often the channels occur in total over all the journeys. The table shows that eta is the channel that occurs the most with 35.34% followed by iota, beta and alpha with 20.48%, 14.42% and 12.70%, respectively. The channels delta and mi are not used often, with only thirteen and two occurrences corresponding to 0.02% and 0.00%. Therefore, these channels are removed for the rest of the paper. Thus, there are fifteen observations removed, and 19,985 are still present.

Table 2: Occurrences channels

Channel	Eta	Iota	Alpha	Beta	Theta	Lambda	Kappa	Zeta	Epsilon	Gamma	Delta	Mi
Number	26,915	15,593	9,674	10,982	4,527	4,453	832	1,187	1,561	411	13	2
%	35.34%	20.48%	12.70%	14.42%	5.94%	5.85%	1.09%	1.56%	2.05%	0.54%	0.02%	0.00%

Anderl, Becker, et al. (2016) find a high share of paths of length one in their datasets. In table 3 the length of the journeys is shown along with the number of times that this length occurs. In this dataset, 8.03% are one-click journeys. Journeys with a length of three are the most common, followed by journeys with two-clicks and four-clicks with 15.75%, 15.17% and 12.69% respectively. Paths that exceed ten clicks are shown in table 4 in tens. Most paths

are at most ten clicks; this is 87.48% of the total observations. The longest path consists of 89 clicks.

Table 3: Length of the paths with maximum length of 10

Length	1	2	3	4	5	6	7	8	9	10
Number	802	1,515	1,573	1,267	1,060	785	629	510	344	250
%	8.03%	15.17%	15.75%	12.69%	10.62%	7.86%	6.30%	5.11%	3.45%	2.50%
Cum. %	8.03%	23.20%	38.96%	51.65%	62.26%	70.13%	76.42%	81.53%	84.98%	87.48%

Table 4: Length of the paths in tens

Length	1-10	11-20	21-30	31-40	41-50	51-60	61-70	71-80	81-90
Number	8,735	999	173	42	17	10	3	5	1
%	87.48%	10.01%	1.73%	0.42%	0.17%	0.10%	0.03%	0.05%	0.01%
Cum. %	87.48%	97.49%	99.22%	99.64%	99.81%	99.91%	99.94%	99.99%	100%

The number of different channels is shown in table 5. Most journeys consist of two different channels, with 36.16% of all journeys. When there are two channels, there are two different channels, but a particular channel can occur several times in the journey. Journeys with three, one and four channels are the most common after the journeys with two channels with 29.33%, 15.60% and 12.89%, respectively. There are no journeys with more than eight channels. The average number of channels in our dataset is 2.60.

Table 5: Number of different channels within journey

Channels	1	2	3	4	5	6	7	8
Number	1,558	3,611	2,929	1,287	431	125	41	3
%	15.60%	36.16%	29.33%	12.89%	4.32%	1.25%	0.41%	0.03%
Cum. %	15.60%	51.77%	81.10%	93.99%	98.31%	99.56%	99.97%	100%

Most journeys have a length of three or two in this dataset. It is reasonable that the length is relatively short because of the use of multiple electronic devices and the existence of cookies on the internet. First, people use several electronic devices to browse the internet. The customer journey only accounts for one electronic device, so the journeys are likely to be short. Also, part of the journey may not be tracked because users did not agree with the cookies or have installed an ad-blocker. Anderl, Becker, et al. (2016) used four different datasets, and the average journey length was between 1.38 and 2.46 for those datasets. Hence, short journeys occur in other datasets as well.

In this dataset, journeys contain most often two or three different channels. On average, European marketers use seven channels (Teradata Corporation, 2013). More than 50% of Dutch households used more than one channel to decide whether to buy white goods, financial products or something in the travelling industry (Van der Veen & van Ossenbruggen, 2015). Thus, we can say that the number of different channels can vary between fields and datasets.

4 Methodology

4.1 Heuristics

As discussed in section 2, two common applied heuristics in previous research are first- and last-click attribution. Most web analytics packages use last-click as a default setting (*Digital Marketing Encyclopedia*, n.d.). The web analytics system can tell where a customer comes from, i.e. from which channel. However, customers mostly visit several channels before deciding whether they want to make a purchase. Hence, deciding which channel to give all or part of the credit for this purchase is essential. The *last-click* attribution gives all credit for the purchase to the last visited channel before a purchase occurs. The *first-click* attribution assigns the credit to the channel visited at the start of the customer journey. So, those two heuristics assign all the credit to a single channel in the customer journey. We call those single-touch attribution models.

As seen in table 5, 1,558 of the 9,985 journeys consist of only one channel. Hence, in 84.40% of the customer journeys, at least two different channels occur. Therefore, another heuristic might be appropriate in this dataset for assigning credit to more than one channel. Those heuristics are called multiple-touch attribution models. For comparison, it would be useful to have a multiple-touch attribution model closely related to first- and last-touch attribution models. Therefore, the *linear-click* attribution is considered. This model gives an equal amount of credit to all the channels in the customer journey (*Marketing attribution models*, 2020). The three heuristics used in this paper are graphically depicted in figure A1.

4.2 Logit Model

Shao and Li (2011) proposed a simple logistic regression where the dependent variable y_i is binary and corresponds to a conversion, 1, or no conversion, 0, of customer journey i. The probability of conversion is given in equation 2 where the explanatory variables are represented by $NC_{i,k} = \sum_{j=1}^{J_i} \#C(v_{i,j}) = C_k$ which is the number of occurrences of channel or state k in journey i.

$$P(y_i) = \frac{\exp(\beta_0 + \sum_{k=1}^{K} \beta_k NS_{i,k})}{1 + \exp(\beta_0 + \sum_{k=1}^{K} \beta_k NS_{i,k})} \tag{2}$$

The odds are stated in equation 3:

$$\frac{P(y_i)}{1 - P(y_i)} = \exp(\beta_0 + \sum_{k=1}^{K} \beta_k NS_{i,k}) \tag{3}$$

Equation 4 shows the logistic regression. The parameters β_k can be estimated by maximum likelihood (ML). The proposed model has two advantages since the model can be interpreted easily and obtain stable and reproducible estimation results.

$$\ln(\frac{P(y_i = 1)}{1 - P(y_i = 1)}) = (\beta_0 + \sum_{k=1}^{K} \beta_k NS_{i,k}) \tag{4}$$

4.3 Markov graphs

Besides the three heuristics and the logit model, Markov graphs will be discussed. Anderl, Becker, et al. (2016) modify an approach that is applied in the context of search engine advertising which results in a graph-based Markovian framework (Archak et al., 2010). It will reflect the journey of channels an individual takes. This journey may end in a transaction, also called conversion, or not. The journeys are depicted in directed Markov graphs. Figure A2 shows a simple example. The graph shows how an individual goes from one channel to another following a directed edge and finally ends up in a transaction or not (Mauldin, Urbanski, & Urbański, 2003). There is a probability of 0.75 that the individual goes to C1 when he or she is currently in start. The probability of going to C2 is 0.25.

The Markov graph M = <C, W> consist of a set of states C and a transition matrix W. For this graph, we first define a set of states, shown in equation 5.

$$C = \{C_1, ..., C_n\} \tag{5}$$

Furthermore, three special states are introduced: the START state, a CONVERSION state, and an absorbing NULL state (Anderl, Becker, et al., 2016). The START state is the start of the customer journey and is always followed by a channel. The CONVERSION state is reached when a transaction is successful. The NULL state means no transaction has occurred at the end of the customer journey. Hence the complete set of states is shown in equation 6.

$$C = \{START, C_1, ..., C_n, CONVERSION, NULL\} \tag{6}$$

Edges connect the states and show the probability of going from one state to another. Those probabilities, which are the edge weights of transition matrix W, are depicted in equation 7. The probability that an individual goes from channel i to channel j is w_{ij}. If, for example, $w_{STARTC_1} = 0.75$, there is a 75% chance that an individual goes to state one if he or she is currently at the START state. A direct edge from CONVERSION to NULL exists such that $w_{CONVERSION,NULL} = 1$. Every journey ends in the NULL state, even when a transaction occurs. All those weights come together in a transition matrix. An example is depicted in figure 1 where $p_{i,j}$ represents the probability of going from state i in period t - 1 to state j in period t (Craig & Sendi, 2002). So $p_{1,2}$ shows the probability of going from state 1 to state 2.

$$w_{ij} = P(X_t = c_j | X_{t-1} = c_i), 0 \leq w_{ij} \leq 1, \sum_{j=1}^{N} w_{ij} = 1 \forall i \tag{7}$$

All those weights come together in a transition matrix. An example is depicted in figure 1 where $p_{i,j}$ represents the probability of going from state i in period t - 1 to state j in period t (Craig & Sendi, 2002). So $p_{1,2}$ shows the probability of going from state 1 to state 2.

Figure 1: Transition Matrix

	Period t		
	1	2	3
1	$p_{1,1}$	$p_{1,2}$	$p_{1,3}$
2	$p_{2,1}$	$p_{2,2}$	$p_{2,3}$
3	$p_{3,1}$	$p_{3,2}$	$p_{3,3}$

Period t - 1

4.3.1 Markov property

For Markov graphs, the Markov property holds (Gudivada, Rao, & Raghavan, 2015). This property means that the current state is only influenced by the previous state or a small group of previous states. A first-order Markov graph means that the current state only depends on the previous state. The Markov property for the first-order Markov graph is stated in equation 8 (Rocca, 2021). A graph is of second-order when the previous two states are important. The same reasoning holds for all other higher-order states. The Markov property can also be written more generally for all higher-order models where the present depends on the last k observations. Equation 9 shows the generalization of the Markov property (Anderl, Becker, et al., 2016).

$$
\begin{aligned}
&P(X_t = c_t | X_{t-1} = c_{t-1}, X_{t-2} = c_{n-2}, ...) \\
&= P(X_t = c_t = X_{t-1} = c_{n-1})
\end{aligned}
\tag{8}
$$

$$
\begin{aligned}
&P(X_t = c_t | X_{t-1} = c_{t-1}, X_{t-2} = c_{t-2}, ..., X_1 = c_1) \\
&= P(X_t = c_t | X_{t-1} = c_{t-1}, X_{t-2} = c_{t-2}, ..., X_{t-k} = c_{t-k})
\end{aligned}
\tag{9}
$$

4.3.2 Removal effect

This paper uses the removal effect to determine channel effectiveness by calculating the conversion attribution per channel. Anderl, Becker, et al. (2016) state that the removal effect helps measuring the contribution of advertising channels. For the removal effect, consider what happens to the probability of conversion when a specific state s_i is removed. All the edges towards s_i that are removed are then shifted to the NULL state. The removal effect shows the importance of a channel. It is calculated using two measures (Archak et al., 2010). The first measure is Eventual Conversion(s_i) which indicates the probability of reaching conversion when you are currently in state s_i. Visit(s_i) shows the probability that you ever reach state s_i when you start from the START state. Removal Effect(s_i) is the product of those two measures.

4.4 ROC curves

The Receiver Operating Characteristic (ROC) curve evaluates the predictive accuracy of the models. This curve shows the performance of a classification model for all possible thresholds. The model plots the True Positive Rate (TPR) on the y-axis and the False Positive Rate (FPR) on the x-axis (Hoo, Candlish, & Teare, 2017). Those two parameters are calculated using equations 10 and 11. TPR, called sensitivity or recall, is the percentage of True Positive (TP) classifications relative to the sum of TP and False Negative (FN) classifications. Figure A3 shows the Positive and Negative classifications. In the case of customer journeys, it means the percentage of accurately classified conversions over the total number of true conversions. TPR takes on a value between zero and one, where one means that all conversions are correctly classified. Additionally, FPR is the percentage of False Positive (FP) classifications relative to the sum of FP and True Negative (TN) classifications. FPR is equal to 1 - specificity, where the latter is the percentage of accurately classified non-conversions over the total number of true non-conversions. Specificity also obtains a value

between zero and one. When specificity equals one or FPR equals zero, all non-conversions are classified as non-conversions.

$$TPR = \frac{TP}{TP + FN} \tag{10}$$

$$FPR = \frac{FP}{FP + TN} \tag{11}$$

An example of a ROC curve is depicted in figure A4. When the ROC curve is a 45-degree line, the dashed line, the number of False Positives equals the number of True Positives. The line then produces the same results as for random guessing. The better the accuracy, the more the line deviates from the 45-degree line towards the upper left corner. Hence, the blue line shows better accuracy than the dashed line. The upper left corner (0,1) implies no false negatives and no false positives. This is the best possible outcome and corresponds to high accuracy.

For comparison of models, it is helpful to have a single scalar value corresponding to the ROC curve. The paper will use two measures: the Area Under the ROC Curve (AUC) and the Top-Decile Lift (TDL). The first one is precisely what the name says. TDL is the ratio of the 10% of customer journeys predicted to be most likely to end up in a conversion that actually end up in a conversion over the baseline conversion rate (Neslin, Gupta, Kamakura, Lu, & Mason, 2006).

4.5 Types of Customer Journeys

The dataset in this paper consists of 9,985 customer journeys. In this section, those customer journeys are divided into three types of customer journeys: impulsive, balanced and considered. The length of the channels and the number of channels are the two ways to distinguish between those three types of journeys. The length of the journeys can be found in tables 3 and 4. In this dataset, the length varies from one click to 89 clicks. The number of different channels within a journey differs from one to eight, as seen in table 5. Some specific channels are appropriate for certain types of channels. An obvious channel for the considered journey is 'price comparison'. This channel is used when someone wants to know the prices and features of products. That is in line with the description 'large pre-shopping stage where customers look for information' given in section 2. However, in our dataset, the channels have been made anonymous, so it will be difficult to know which channels are actually in the dataset.

As explained in section 2, impulsive journeys result from a quick search. Hence, it could be argued that a quick search implies that the journeys are short or that only a few channels

are encountered. Thus, the length of the journey is short, or only a few different channels occur in the journey. Considered journeys are the most elaborated journeys, which imply the longest length of journeys or the most number of channels. The journeys that lie between the impulsive and considered journeys are balanced journeys, and hence their length and number of channels are a bit more mediocre. Thus, the length goes from small to long from impulsive to balanced to considered journeys. The same order is valid for the number of channels where impulsive journeys correspond to the least amount of channels.

Three types of journeys are distinguished, resulting in three groups of 33.33%. Impulsive journeys are the 33.33% shortest journeys, balanced journeys range from 33.33% to 66.67% and considered journeys are the journeys starting from 66.67%. The same holds for the number of channels. When a split is not that clear, the percentage closest to 33.33% is chosen. This will become clear in section 4.5.1 using the actual data.

4.5.1 Length of Journeys

For the length of paths, tables 3 and 4 show how often each length occurs, and the percentage is shown. The impulsive journey ranges from 0% to 33.33%. The cumulative percentage of journeys with length one and two together corresponds to 23.20% and for journeys with a length of one till three to 38.96%. It is evident that 33.33% lies in the middle of those two percentages. The split will be made by looking at which number is closest to 33.33%, which in this case is 38.96%. Hence, the impulsive journeys consist of journeys with lengths of one to three. The balanced journeys range from a length of four to a length of six, ranging from 38.96% to 70.13%. Then, the considered channels are all journeys that exceed a length of six. To conclude, impulsive channels are until a length of three, balanced channels have a length of four to six and considered channels have a length of at least seven. This can also be seen in table 6.

4.5.2 Number of Channels

The types of journeys will also be distinguished by looking at the number of different channels in the journeys, which can be found in table 5. The cumulative percentage for journeys with only one channel is 15.60% and 51.77% for journeys with two different channels. The split for impulsive channels has to be made between those two channels. The number that is closest to 33.33% is 15.60%. Hence, impulsive journeys consist of one channel. However, there is only a tiny difference between the distance from 15.60% to 33.33% and from 33.33% to 51.77%. The same holds for the balanced journeys since 51.77% and 81.10% are both not close to 66.67%. The latter is closest to 66.67%, and therefore balanced journeys range

from two to three different channels. Considered journeys are thus journeys that have at least four different channels. Thus, impulsive journeys are until 15.60%, balanced journeys range from 15.60% to 81.10%, and considered journeys are from 81.10%. It is important to note that these groups are not all close to 33.33%, which is vital to keep in mind when concluding. However, these groups are the most equal, and therefore these groups are used in this paper. To conclude, impulsive journeys consist of one channel, balanced journeys of two to three different channels and considered journeys have at least four different channels. The splits are shown in table 6.

Table 6: Types of Journeys

	Impulsive	Balanced	Considered
Length	1 - 3	4 - 6	7 - 89
Number of channels	1	2 - 3	4 - 8

5 Results

In this section, the results are presented both for the whole dataset as well as for the three different types of customer journeys. First, the predictive accuracy is evaluated by looking at the ROC curves, AUC and TDL. Also, the robustness of the removal effects will be evaluated by looking at the average standard deviation as a percentage of the average removal effect. Thereafter, the attribution results are presented and compared across models. The transition matrix for the first-order Markov model is given for the complete dataset and this matrix is used to illustrate the first-order directed Markov graph.

5.1 Full dataset

This section will show the results for the complete dataset where the channels delta and mi are removed. The dataset consists of 9,985 observations. In section 5.2 the length of the customer journeys is used to distinguish between impulsive, balanced and considered journeys. The split is made using the number of channels in section 5.3.

5.1.1 Predictive Accuracy

The ROC curves for the logit model and the first- to fourth-order Markov models are depicted in figure A5 to evaluate the predictive accuracy. Anderl, Becker, et al. (2016) found that the accuracy improved for higher-order Markov models, which also holds for this dataset. The first-order Markov model lies most closely to the diagonal line, and the fourth-order

Markov model is furthest away. This implies that the accuracy improves when the order of the Markov model increases. Thus, when there is more information on a higher number of previous channels, it is easier to estimate the following channel. However, the ROC curves are nearly the same as the 45-degree line, which is somewhat concerning as the latter indicates random guessing. The ROC curve of the logit model lies further from the diagonal line, which implies higher predictive accuracy.

Table A3 shows the results for the logit model for the dataset containing 9,985 observations. Only eta, iota and the intercept are significant, the other channels are not significant. The significance of the intercept is remarkably higher than for the two channels. Furthermore, the value for the intercept is high, and the values for the channels are somewhat similar and close to zero. Hence, it might be the case that the logit model does not explain the effects of the channels well since the intercept heavily influences the model.

To compare models, AUC and TDL are measured which are single scalar values. The values are stated in table 7. AUC is expected to be not much larger than 0.5000 for the Markov models because the ROC curves are so close to the diagonal line. The table shows that this is the case for all four Markov models. The AUC increases when the order of the Markov model increases. Therefore, the fourth-order Markov model has the best predictive accuracy when AUC is used. This is in line with the conclusions drawn from the ROC curves. The AUC for the logit model is 0.654, which is remarkably higher than the AUC of the Markov models. It seems that the logit model is performing well. However, it is pretty plausible that this only happens because of the highly significant intercept and that the model is not working well. Anderl, Becker, et al. (2016) also mentioned an AUC for the heuristic models. However, this seems to be incorrect, and they probably tried to calculate something close to AUC to compare it to the Markov models. Therefore, AUC is only calculated for the Markov models and the logit model in this paper.

Since AUC cannot be calculated for our heuristic models, it can be helpful to use another measure. The TDL can be measured for all eight models and logit has the highest. However, this is again caused by the vague results of the model. The last-click heuristic has the highest predictive accuracy when logit is ignored. This is surprising since we might have expected the Markov models to work best.

Table 7: Predictive Accuracy

Measure	First-click	Last-click	Linear-click	Logit	Markov 1	Markov 2	Markov 3	Markov 4
AUC	-	-	-	**0.654**	0.509	0.517	0.528	0.543
TDL	1.600	2.108	1.790	**4.649**	1.707	1.735	1.714	1.727

bold = highest AUC

5.1.2 Robustness

The predictive accuracy is essential, but it is also important to look at the robustness since it prevents inconsistency (Little, 2004). The robustness of the removal effect will be evaluated. In table A4 the removal effects for the number of conversions are shown. For all four Markov models, the removal effect of alpha is the largest, which means that the probability of ending with a conversion drops the largest when the channel alpha is removed. Channels with the largest removal effects after alpha are iota, eta, beta and theta.

Furthermore, the average standard deviation of the removal effects is calculated and can be found in A5. This table will be used to calculate the average standard deviation as a percentage of the average removal effect, for which the results can be found in table 8. Simpler models are generally more robust, as seen from the table.

Table 8: Average standard deviation as % of average removal effect

Markov 1	Markov 2	Markov 3	Markov 4
0.90%	0.94%	0.95%	0.94%

bold = lowest %

To conclude, the results of section 5.1.1 and this section will be combined to decide which models to include in the section 5.1.3. Of course, all three heuristics are used since they are industry standards. The choice between Markov models depends on the predictive accuracy and robustness. The fourth-order Markov model has the highest AUC. However, when the TDL is used, the second-order Markov model has the highest predictive accuracy out of the four Markov models. The first-order Markov model is the best, only with a slight difference, when the robustness of the removal effects is considered. Only the fourth-order Markov model could be considered, but for the complete dataset all four Markov models are considered. When the types of journeys are researched, a choice between Markov models is made.

5.1.3 Attribution Results

The total number of conversions in the dataset is 19,772. Table A6 shows which channel gets the credit for the conversion using several different methods. This is depicted graphically in figure A6. For ease of interpretation, table 9 shows this number as a percentage of the total number of conversions. Some columns do not seem to add up to 100%, but this occurs because of rounding to percentages. In reality, they add up to 100%, so this is not a thing to worry about.

First, the heuristics will be discussed. First-click shows that alpha occurs most often as the first channel with the full credit for 31.90% of the conversions. This channel is followed by iota, eta and beta with 23.27%, 16.00% and 14.31%, respectively, such that these channels are also crucial at the start of the customer journey. Alpha also gets the most credit out of all channels when using last-click with 42.72%. The channels eta and iota are the second and third most important channels for the last-click attribution model. Linear-click assigns value to all channels that occur in the customer journey. In this heuristic, alpha is still given the most credit, with 38.31% of all the total conversions. Alpha is followed by iota, eta, beta, lambda and theta with 19.49%, 17.90%, 10.53%, 5.23% and 5.17%, respectively. The values of linear-click seem somewhat the average of first- and last-click, which makes sense since the value is distributed equally among all channels. To conclude, heuristics show that alpha, iota and eta are the three most important channels to reach a conversion.

Furthermore, the first- and higher Markov models give clear results. For all four models, the order of the channels is the same. The most important channel is alpha with 27.35% to 29.36%. Thereafter, iota, eta and beta are the most important for conversion. The percentages differ slightly across the Markov models, but the differences are minor. The biggest difference is for alpha with 2.01%.

In table 2 we see that the channel occurrences for alpha are as fourth following eta, iota and beta. Table 9 shows that alpha gets the most credit out of all channels for the conversions. Hence, alpha seems to be a important channel since the channel does not occur the most often but comes first when the credit of a conversion is assigned. In section 5.1.2, we saw that the removal effect was the largest for alpha, which is in line with the conclusions from this section.

Table 9: Attribution results in comparison to heuristic models (in %) of number conversions

Channel	First-click	Last-click	Linear-click	Markov 1	Markov 2	Markov 3	Markov 4
Eta	16.00%	21.08%	17.90%	17.07%	17.35%	17.14%	17.27%
Iota	23.27%	16.95%	19.49%	19.26%	19.35%	19.33%	18.77%
Alpha	**31.90%**	**42.72%**	**38.31%**	**27.35%**	**28.94%**	**29.31%**	**29.36%**
Beta	14.31%	5.00%	10.53%	12.35%	12.05%	12.22%	11.71%
Theta	8.12%	3.30%	5.17%	10.32%	9.03%	9.47%	9.78%
Lambda	4.55%	6.09%	5.23%	6.34%	5.83%	5.94%	5.97%
Kappa	0.37%	1.16%	0.70%	1.42%	1.37%	1.10%	0.98%
Zeta	0.14%	0.54%	0.69%	2.06%	1.95%	1.78%	1.97%
Epsilon	0.50%	2.69%	1.38%	3.00%	3.34%	2.79%	3.24%
Gamma	0.83%	0.47%	0.61%	0.83%	0.78%	0.92%	0.95%

bold = highest % for each model

5.1.4 Transition Matrix and Probabilities

The Markov chain consists of thirteen states, of which there are ten channels and three special states. The transition matrix of the first-order Markov model is depicted in table 10 and some conclusions can be drawn from this matrix. First, when starting in the start state, it is most likely that the individual goes to alpha with a probability of 32.70%. This might mean that alpha is a channel that often occurs at the beginning of a short customer journey, for example, direct type-in. This is in line with the conclusion of the importance of alpha in first-click. Other channels that often follow start are iota, eta and beta. Conversion occurs most often directly after alpha, but also eta and lambda are often followed by a conversion. Null occurs after a conversion which shows that the direct edge exists. The absorbing null state is also evident from the transition matrix.

Table 10: Transition probabilities first-order Markov

Channel	Eta	Iota	Alpha	Beta	Theta	Lambda	Gamma	Epsilon	Kappa	Zeta	Conversion	Null
Start	0.161	0.231	0.327	0.137	0.084	0.043	0.008	0.005	0.003	0.001	-	-
Eta	-	0.089	0.132	0.129	0.046	0.023	0.002	0.031	0.010	0.010	0.120	0.408
Iota	0.102	-	0.183	0.121	0.094	0.063	0.007	0.027	0.013	0.036	0.078	0.278
Alpha	0.041	0.109	-	0.035	0.062	0.037	0.003	0.020	0.005	0.009	0.149	0.529
Beta	0.392	0.181	0.127	-	0.070	0.029	0.004	0.015	0.005	0.015	0.037	0.126
Theta	0.079	0.229	0.361	0.084	-	0.048	0.005	0.032	0.015	0.012	0.032	0.102
Lambda	0.046	0.132	0.170	0.048	0.085	-	0.006	0.038	0.037	0.019	0.098	0.319
Gamma	0.088	0.168	0.227	0.082	0.080	0.066	-	0.014	-	0.011	0.057	0.206
Epsilon	0.087	0.108	0.190	0.050	0.058	0.064	0.007	-	0.011	0.027	0.095	0.303
Kappa	0.096	0.136	0.111	0.049	0.076	0.105	0.005	0.024	-	0.019	0.093	0.287
Zeta	0.081	0.292	0.194	0.095	0.068	0.083	0.002	0.048	0.012	-	0.030	0.096
Conversion	-	-	-	-	-	-	-	-	-	-	-	1.000
Null	-	-	-	-	-	-	-	-	-	-	-	1.000

Figure A7 shows the first-order directed Markov graph. In this figure, the states can be found with the probabilities on the edges as well. Both the incoming and outgoing edges sum up to one. The direct edge from conversion to null is also depicted in the figure. Furthermore, null is an absorbing state. The transition matrix from table 10 could not have been used since the rows have to sum up to one. That is not the case since the numbers are rounded. Therefore, the table is adjusted a tiny bit so that the rows sum up to one. Some rows summed up to 0.999; in that case, 0.001 was added to the channel with the largest probability. When the rows added up to 1.001 (1.002), 0.001 (0.002) was subtracted from the largest probability. The probabilities used for conducting the first-order directed Markov graph are depicted in table A7.

5.2 Length of Journeys

In this section, a split is made using the length of customer journeys where three groups arise. The length of impulsive journeys is between one and three, balanced journeys have a length of four to six and considered journeys are journeys that have a length of at least seven. Those journeys occur 3,890, 3,112 and 2,983 times, respectively. Using the length to split the data into three groups seems a good idea since the groups look similar in size.

The heuristics will be used in this section since they can conclude something about the importance of channels at the beginning or the end of the customer journey. This is interesting in researching the three different types of customer journeys. When looking at the predictive accuracy and robustness in sections 5.2.1 and 5.2.2, the best Markov model is chosen to achieve attributing results. For impulsive journeys, the length is at most three, implying that the fourth-order Markov model is the same as the third-order model.

5.2.1 Predictive Accuracy

The ROC curves for all three types of customer journeys are shown in figure A8. The Markov graphs lie close to the diagonal again, but there is a difference between the types of journeys. The ROC curves lie closest to the diagonal for impulsive journeys and are further away for balanced and considered journeys. The ROC for the logit model shows that it seems to have high predictive accuracy for impulsive journeys. It looks more similar to the Markov models for balanced and considered journeys.

In table 11 the AUC and TDL are shown. The logit model has the highest AUC for all three types of customer journeys. The value for impulsive is especially high, whereas the AUC for balanced and considered journeys is only a bit higher than for the Markov models. Table A8 shows the logit results and this shows that the channels are (highly) significant for impulsive journeys and are not significant for the other two types of journeys. This can show why impulsive journeys have a higher AUC. It is again concerning that the intercept is large. Hence, the logit model is probably unable to explain the effects of the channels. For all three types of journeys, the fourth-order Markov model has the highest AUC out of the Markov models and thus the highest predictive accuracy. The third-order Markov model for impulsive journeys obtains the same value, which is logical since those two models are equal. Again, logit has the highest TDL, which is suspected to result from the large intercept. After logit, first-click has the highest value for impulsive journeys, and last-click has the highest TDL for balanced and considered journeys. Hence, the heuristics seem to outperform the Markov models, considering the TDL.

Table 11: Predictive Accuracy

Measure	Journey	First-click	Last-click	Linear-click	Logit	Markov 1	Markov 2	Markov 3	Markov 4
AUC	Impulsive	-	-	-	**0.759**	0.511	0.520	0.525	0.525
	Balanced	-	-	-	**0.589**	0.522	0.532	0.548	0.570
	Considered	-	-	-	**0.536**	0.506	0.522	0.549	0.580
TDL	Impulsive	1.284	0.585	0.961	**5.218**	1.093	1.102	1.108	1.108
	Balanced	0.700	1.332	0.929	**8.721**	1.187	1.159	1.202	1.190
	Considered	0.821	2.862	2.136	**19.281**	1.663	1.715	1.801	1.881

bold = highest AUC for each type of journey

5.2.2 Robustness

The removal effects for all four Markov models for the three types of customer journeys can be found in table A9. The removal effect for alpha is the highest for impulsive and balanced journeys. For impulsive journeys, eta is the second largest, followed by iota. For balanced journeys, those channels are switched. The largest removal effect for considered journeys is for iota followed by alpha and beta. The results from table 12 are important to test the robustness of the removal effects. Models with the lowest percentage are preferred. However, note that the percentages differ only a tiny bit.

Table 12: Average standard deviation as % of average removal effect

Impulsive Journeys				Balanced Journeys				Considered Journeys			
M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4
1.14%	1.16%	1.16%	1.16%	**0.95%**	1.01%	1.01%	0.98%	**0.64%**	0.66%	0.68%	0.69%

bold = lowest % per for each type of journey

For the next section, deciding which models to use is important. All three heuristics will be used because they have the highest TDL, excluding the logit model. Furthermore, these models can interpret the importance of channels at the beginning or the end of customer journeys. AUC implies that the fourth-order models give the highest predictive accuracy. The first-order Markov models are the most robust, but this model will not be used because of the small difference. Thus, the following section uses three heuristics and the fourth-order Markov model.

5.2.3 Attribution Results

The number of conversions can be found in table A10. However, it is more interesting to look at the attribution results. The results for all three types of journeys can be found in table 13. For impulsive journeys, alpha is the most important, followed by eta, iota and beta. All four models show that alpha and iota are the first and second most essential channels in

balanced journeys. After those two channels, the order differs between the models with beta, theta, eta and lambda occurring. For considered journeys, there are quite some differences between the four models. First-click, linear-click and the Markov model find iota the most important, whereas last-click finds alpha the most important. For first-click, alpha is the fifth most crucial channel. Section 5.1.3 concluded that alpha was the most important for the complete dataset. Hence, we see that the considered journeys have a difference for the three models where iota is the most important. Thus it can be concluded that alpha is a vital channel at the end of a considered journey.

Table 13: Attribution Results (in %) of Number of Conversions

	Impulsive Journeys				Balanced Journeys				Considered Journeys			
Channel	First	Last	Linear	M4	First	Last	Linear	M4	First	Last	Linear	M4
Eta	21.15%	24.75%	22.69%	23.10%	7.00%	13.32%	9.30%	11.91%	9.24%	18.73%	12.05%	13.87%
Iota	17.86%	16.18%	17.04%	17.85%	27.71%	16.88%	20.95%	19.35%	**39.64%**	20.60%	**27.88%**	**20.32%**
Alpha	**36.10%**	**43.24%**	**39.93%**	**34.40%**	**34.43%**	**49.10%**	**43.46%**	**32.43%**	8.20%	**28.62%**	21.36%	18.81%
Beta	12.84%	5.85%	9.61%	11.08%	13.74%	3.20%	9.57%	11.45%	22.02%	4.44%	16.49%	14.07%
Theta	6.09%	1.74%	3.73%	6.19%	11.23%	4.41%	7.09%	10.45%	11.55%	8.36%	8.16%	12.68%
Lambda	4.27%	5.30%	4.72%	4.34%	3.85%	5.93%	4.98%	5.56%	7.15%	10.00%	8.02%	7.53%
Kappa	0.34%	0.63%	0.47%	0.68%	0.38%	1.86%	1.04%	1.84%	0.52%	2.28%	1.10%	1.76%
Zeta	0.04%	0.36%	0.23%	0.45%	0.16%	0.79%	1.08%	2.25%	0.52%	0.90%	2.05%	4.38%
Epsilon	0.46%	1.37%	0.89%	1.30%	0.69%	4.19%	2.00%	3.79%	0.34%	5.90%	2.41%	5.44%
Gamma	0.85%	0.58%	0.69%	0.62%	0.81%	0.32%	0.50%	0.98%	0.82%	0.19%	0.48%	1.14%

bold = highest % for each model for each type of customer journey

5.3 Number of Channels

The dataset is also split into three groups using the number of different channels occurring in customer journeys. The journey that uses the least amount of channels is the impulsive journey followed by balanced and considered journeys with a range of one, two to three and at least four channels, respectively. Those journeys occur 1,558, 6,540 and 1,887 times in the same order as mentioned in the previous sentence. This split is way less equal than the split made using the length of the journeys. However, this is the most equal split possible so these groups will be used. The unequal groups have to be kept in mind when drawing a conclusion.

5.3.1 Predictive Accuracy

The ROC curves are shown in figure A9 and the values for predictive accuracy can be found in table 14. Logit again obtains the highest AUC for impulsive and balanced journeys. However, it is interesting that the fourth-order Markov model for considered journeys obtains the highest predictive accuracy. The logit results in table A11 show that only eta has a low significance for balanced journeys. Also, the intercept is high again and is highly significant. Therefore, the comparison between Markov models is more important, and the fourth-order

model has the highest AUC for all three types of customer journeys. When excluding logit, the fourth-order Markov model has the highest TDL for impulsive journeys. For balanced and considered journeys, this is last-click and first-click, respectively.

Table 14: Predictive Accuracy

Measure	Journey	First-click	Last-click	Linear-click	Logit	Markov 1	Markov 2	Markov 3	Markov 4
AUC	Impulsive	-	-	-	**0.691**	0.505	0.508	0.509	0.518
	Balanced	-	-	-	**0.708**	0.522	0.532	0.548	0.570
	Considered	-	-	-	0.534	0.506	0.522	0.549	**0.602**
TDL	Impulsive	1.260	1.260	1.260	**7.578**	1.267	1.238	1.270	1.313
	Balanced	1.300	2.388	1.702	**5.036**	1.688	1.712	1.736	1.729
	Considered	3.860	1.316	2.299	**8.270**	1.685	1.751	1.749	1.788

bold = highest AUC for each type of journey

5.3.2 Robustness

The removal effects for all three types of customer journeys are stated in table A12. Alpha has the highest removal effect when looking at impulsive journeys, followed by eta and iota. Both balanced and considered journeys have the highest removal effect for iota, and the second-largest removal effect is for alpha. Table 15 shows again that the first-order Markov model is the most robust.

Table 15: Average standard deviation as % of average removal effect

Impulsive Journeys				Balanced Journeys				Considered Journeys			
M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4
1.64%	1.66%	1.67%	1.65%	**0.79%**	0.81%	0.83%	0.82%	**0.54%**	0.57%	0.58%	0.60%

bold = lowest % for each type of journey

The attribution results will be evaluated using the three heuristics and the fourth-order Markov model again. Using AUC, the fourth-order models show the highest predictive accuracy. When looking at the TDL, the predictive accuracy differs among the types of journeys. The fourth-order Markov model, last-click and first-click obtain the highest values for impulsive, balanced and considered journeys, respectively. The first-order Markov model is the most robust, but since the differences are minor, this model will not be used in the next section.

5.3.3 Attribution Results

Table A13 shows the number of conversions, and the attribution results can be found in table 16. A prominent result is that the three heuristics deliver the same results for impulsive journeys. This is logical since those journeys consist only of one channel, so the first click is

also the last. The results for the impulsive journeys are the same as when the split of the length of the journey was used. The order from most important to less important channels for all four models is alpha, eta, iota and beta. For balanced journeys, the results are somewhat different. Using first-click, the importance of alpha goes from the most important channel to the fifth most important channel. Alpha remains the most important for last-click. For linear-click and the Markov model, alpha was the most crucial channel when the split was made using the length of the customer journeys, but using the split on the number of channels iota is the most important. Thus, alpha becomes more important at the end of the balanced journeys when the latter split is made. Also, iota becomes more important at the beginning of the journey. For the considered channels, the results are almost entirely similar to using the length of the journey to make the split. Only the second and third most essential channels using last-click switched. Hence, the split using the number of channels changes the results for balanced journeys compared to the split used in section 5.2. This subset grew from 3,112 to 6,540 observations, and this increase could be the reason for the differences in results.

Table 16: Attribution Results (in %) of Number of Conversions

	Impulsive Journeys				Balanced Journeys				Considered Journeys			
Channel	First	Last	Linear	M4	First	Last	Linear	M4	First	Last	Linear	M4
Eta	19.25%	19.25%	19.25%	20.04%	13.00%	23.88%	17.02%	17.29%	10.33%	18.58%	13.67%	14.21%
Iota	14.00%	14.00%	14.00%	14.59%	**31.98%**	21.45%	**25.76%**	**21.42%**	**38.60%**	13.16%	**22.99%**	**17.88%**
Alpha	**56.46%**	**56.46%**	**56.46%**	**56.14%**	6.10%	**28.61%**	19.38%	20.39%	4.85%	**25.94%**	17.59%	16.55%
Beta	5.36%	5.36%	5.36%	4.85%	23.75%	4.80%	15.86%	15.16%	24.06%	3.65%	16.79%	14.74%
Theta	0.03%	0.03%	0.03%	0.00%	17.38%	6.21%	10.72%	12.72%	13.16%	9.57%	10.07%	12.85%
Lambda	4.33%	4.33%	4.33%	3.97%	4.52%	7.13%	5.69%	5.41%	6.17%	12.22%	8.69%	8.03%
Kappa	0.07%	0.07%	0.07%	0.18%	0.70%	1.95%	1.24%	1.52%	0.69%	4.22%	2.01%	2.84%
Zeta	0.00%	0.00%	0.00%	0.00%	0.22%	1.15%	1.12%	1.74%	0.57%	0.94%	2.90%	4.77%
Epsilon	0.12%	0.12%	0.12%	0.05%	0.99%	4.23%	2.37%	3.26%	0.50%	11.34%	4.43%	6.78%
Gamma	0.38%	0.38%	0.38%	0.18%	1.36%	0.59%	0.85%	1.10%	1.07%	0.38%	0.86%	1.34%

bold = highest % for each model for each type of customer journey

6 Conclusion

This paper discussed the importance of advertising channels using three heuristics, a logit model and first- to fourth-order Markov models. The channels delta and mi were removed at the beginning of the research since they only occurred fifteen times out of a total of 76,150 occurrences. The results are split up into three subsections. The first part of the results uses the whole dataset, excluding delta and mi, to generate results. Thereafter, the journeys are split into three types. First, impulsive journeys have a length that ranges from one to three and the journey has only one type of channel. Balanced journeys have a length from four to six and contain two or three different channels. The considered journeys exceed this so that the length is at least seven, and there are, at the minimum, four different channels in the

journey.

Each subsection starts with evaluating the predictive accuracy and the robustness of the removal effects. The predictive accuracy is measured using both AUC and TDL. The logit model has the highest AUC and TDL for all three sections. This does not hold for the fourth-order Markov model for considered journeys when those journeys are based on the number of channels. However, the results for the logit model seem strange. Only a few channels are significant, and those have really small and similar values. Furthermore, the intercept is highly significant and is large compared to the channels. Therefore, it is expected that the high measures for the predictive accuracy result because the intercept distorts the results and makes it seem that logit is performing well.

When considering the predictive accuracy for the other models, AUC shows that the fourth-order Markov model has the highest predictive performance out of the four Markov models for all three sections. It is important to note that the third- and fourth-order models are the same for impulsive journeys when they are defined using the length of the journeys. TDL has contradicting results for the predictive accuracy. For the entire dataset, the predictive accuracy is highest for last-click. When the types of journeys are split using the length of the journeys, the highest value corresponds to first-click for the impulsive journeys and last-click for the balanced and considered journeys. TDL is the highest for the fourth-order Markov model for impulsive journeys, last-click for balanced journeys and first-click for considered journeys when the distinction is made based on the number of different channels. The first-order Markov model is the most robust all sections. However, the differences are minimal and therefore this model will not be used for the attribution results of the different types of journeys.

Two channels seem to be the most important for obtaining a conversion: alpha and iota. The whole dataset shows that the removal effect and the attribution results are the highest for alpha. Impulsive journeys for both possible ways of splitting the journeys also conclude that alpha has the highest removal effect and attribution results. When the split is made based on the length of the journeys, the removal effect is highest for alpha for balanced journeys and highest for iota for considered journeys. The attribution results for balanced journeys are also highest for alpha. However, the attribution results for considered journeys are a bit contradicting. Iota is the highest for first- and linear-click and for the fourth-order Markov model. For last-click, alpha is the highest. This might imply that alpha becomes more important at the end of considered journeys with a length of at least seven. Hence, alpha could be important for this journey to let the customer decide whether or not a transaction will be made. The types of journeys are also based on the number of channels. Then, the removal effect is highest for iota for balanced and considered journeys. For both journeys,

iota obtains the highest attribution results using first- and linear-click and the fourth-order Markov model, and alpha obtains the highest when using last-click.

Last-click obtains several remarkable results. First, it obtains the highest TDL for the whole dataset, balanced and considered journeys based on their length and for balanced journeys based on the number of channels in the journey. AUC shows that the Markov models have values close to 0.500, implying that the data is close to random. However, the last-click attribution model overestimates any data dependencies. Also, the attribution results for last-click are higher than for the other models. For the entire dataset, alpha gets the highest attribution results for all models. However, using last-click, this value is 42.72%, whereas first-click has a value of 31.90% and the Markov models have a value between 27.35% and 29.36%. The attribution results of the types of journeys show that the last-click has the highest value again, or it shows a higher attribution result for a different channel, alpha, than for first-click, linear-click and the fourth-order Markov model. These results show that the last-click attribution model overestimates the effectiveness of channels.

From previous research, the overestimating of last-click is a common known result. Ji and Wang (2017) found that last-click overestimates the contribution of search advertisements, such as SEO and SEA, and ignores the advertisements before the last click. This implies that the model assigns too much value to the last channel. A quote from research about last-click is that it "seems to partly overestimate both statistically and economically insignificant channels and underestimate efficient ones" (Georgopoulos, 2017). Furthermore, last-touch incentives an increase in ad exposures (Berman, 2018). This increase results in a too high level of exposure, and the method thus results in overexposure of advertising. In turn, this results in lower profits for the advertising company.

Overall, alpha seems to be the most important channel. This is especially true for short journeys or journeys with a small number of channels occurring. It also holds when looking at last-click for long journeys or journeys with many channels in the journey. For balanced and considered channels, iota is also an important channel. The removal effects are often the largest for iota for those two journeys. For those journeys, the attribution results for first- and linear-click and the fourth-order Markov model are the highest for iota, where alpha has the highest attribution result for first-click.

7 Discussion

This paper used eight models to determine the importance of advertising channels in a dataset consisting of 10,000 observations (*Markov model for online multi-channel attribution [R package channelattribution version 2.0.5]*, 2022). The models used are three heuristics,

a logit model and first- and higher-order Markov graphs. Thus, this paper contributes to existing research by introducing a dataset combined with eight different models.

Furthermore, the distinction between types of customer journeys is new and expands existing research. Wolny and Charoensuksai (2014) explained the difference between three types of customer journeys. The first type is impulsive journeys which correspond to short journeys of length one to three or journeys with only one channel. Then, balanced journeys are a bit longer and contain more different channels in the journey. The length is between four and six, and the journeys contain two or three different channels. The last type of journeys are considered journeys, and those are the longest ones or contain the most different channels. They have a length of at least seven or contain at least four different channels.

The research on different types of customer journeys is refreshing. Hence, for further research, it would be interesting to use these three types of customer journeys on other datasets. A dataset that contains real information instead of anonymous channels could contribute to this paper. Hopefully, more evident results can be drawn from that research. Using another dataset would also be good since this dataset has some strange results. The values for AUC indicate that the dataset is close to random, and the results for logit do not look good either. When using another dataset, this could give better results.

Additionally, this paper used three types of journeys since those types are known from existing research. However, it can be interesting to look at different types of journeys as well. An example is splitting the data into ten or even more groups where the first group contains the shortest journeys or the least number of channels. This increases for the following groups. It is also refreshing to look at another way to split the data rather than at the length or the number of channels. Maybe it is possible to combine those two criteria or come up with another kind of criteria.

Another limitation in this research is that the split using the number of channels seems unequal. Impulsive, balanced and considered journeys contain 1,559, 6,543 and 1,898 observations, respectively. It would be good to try to deal with this. One option could be to look at several different splits and compare the results. Another possibility is to make three groups consisting of 33.33% each exactly. In this dataset, that would mean that impulsive journeys consist of all journeys that contain one channel and 17.74% of the 36.13% of journeys with two channels. This can, of course, also be done by making the split using the length of the journeys. However, the split already seems pretty equal in this paper, so it is unnecessary. It can, of course, be the case that the splits are more or less equal in other datasets. Hence, this is important to keep in mind when researching further.

References

Abhishek, V., Fader, P., & Hosanagar, K. (2012). Media exposure through the funnel: A model of multi-stage attribution. *Available at SSRN 2158421*.

Advertising channel meaning, importance, factors amp; example. (n.d.). Retrieved from `https://www.mbaskool.com/business-concepts/marketing-and-strategy-terms/10819-advertising-channel.html`

Albrecht, L. J. (2002). Online marketing: The use of cookies and remedies for internet users. *Suffolk UL Rev.*, *36*, 421.

Anderl, E., Becker, I., Von Wangenheim, F., & Schumann, J. H. (2016). Mapping the customer journey: Lessons learned from graph-based online attribution modeling. *International Journal of Research in Marketing*, *33*(3), 457–474.

Anderl, E., Schumann, J. H., & Kunz, W. (2016). Helping firms reduce complexity in multichannel online data: A new taxonomy-based approach for customer journeys. *Journal of Retailing*, *92*(2), 185–203.

Anderson, C. K., & Cheng, M. (2017). Multi-click attribution in sponsored search advertising: An empirical study in hospitality industry. *Cornell Hospitality Quarterly*, *58*(3), 253–262.

Archak, N., Mirrokni, V. S., & Muthukrishnan, S. (2010). Mining advertiser-specific user behavior using adfactors. In *Proceedings of the 19th international conference on world wide web* (pp. 31–40).

Barbu, C. M., Ponea, Ş., & Bogdănoiu, C.-L. (2019). Offline advertising versus online advertising. *Theoretical and Practical Research in Economic Fields*, *10*(2 (20)), 118–131.

Berman, R. (2018). Beyond the last touch: Attribution in online advertising. *Marketing Science*, *37*(5), 771–792.

Berte, K., & Gysels, J. (2007). Problemen met reclame-inkomsten in de printmedia in een digitale omgeving.

Breuer, R., Brettel, M., & Engelen, A. (2011). Incorporating long-term effects in determining the effectiveness of different types of online advertising. *Marketing Letters*, *22*(4), 327–340.

Craig, B. A., & Sendi, P. P. (2002). Estimation of the transition matrix of a discrete-time markov chain. *Health economics*, *11*(1), 33–42.

Cui, T. H., Ghose, A., Halaburda, H., Iyengar, R., Pauwels, K., Sriram, S., ... Venkataraman, S. (2021). Informational challenges in omnichannel marketing: remedies and future research. *Journal of Marketing*, *85*(1), 103–120.

Digital marketing encyclopedia. (n.d.). Retrieved from `https://willmarlow.com/resources-2/digital-marketing-encyclopedia/`

Georgopoulos, A. (2017). Business analytics report.

Goldfarb, A. (2014). What is different about online advertising? *Review of Industrial Organization*, *44*(2), 115–129.

Gudivada, V. N., Rao, D., & Raghavan, V. V. (2015). Big data driven natural language processing research and applications. In *Handbook of statistics* (Vol. 33, pp. 203–238). Elsevier.

Hanekom, J., & Scriven, C. (2002). Traditional and online advertising: an explanation of current and future trends.

Harary, F., & Lipstein, B. (1962). The dynamics of brand loyalty: A markovian approach. *Operations Research*, *10*(1), 19–40.

Hoo, Z. H., Candlish, J., & Teare, D. (2017). *What is an roc curve?* (Vol. 34) (No. 6). BMJ Publishing Group Ltd and the British Association for Accident ....

Ji, W., & Wang, X. (2017). Additional multi-touch attribution for online advertising. In *Thirty-first aaai conference on artificial intelligence.*

Ji, W., Wang, X., & Zhang, D. (2016). A probabilistic multi-touch attribution model for online advertising. In *Proceedings of the 25th acm international on conference on information and knowledge management* (pp. 1373–1382).

Klapdor, S. (2013). *Effectiveness of online marketing campaigns: An investigation into online multichannel and search engine advertising.* Springer Science & Business Media.

Lemon, K. N., & Verhoef, P. C. (2016). Understanding customer experience throughout the customer journey. *Journal of marketing*, *80*(6), 69–96.

Lewis, R. A., & Nguyen, D. T. (2014). A samsung ad for the ipad? display advertising's competitive spillovers to search. *Display Advertising's Competitive Spillovers to Search (January 2, 2014).*

Li, H., & Kannan, P. (2014). Attributing conversions in a multichannel online marketing environment: An empirical model and a field experiment. *Journal of Marketing Research*, *51*(1), 40–56.

Li, H., Kannan, P., Viswanathan, S., & Pani, A. (2016). Attribution strategies and return on keyword investment in paid search advertising. *Marketing Science*, *35*(6), 831–848.

Li, J., Abbasi, A., Cheema, A., & Abraham, L. B. (2020). Path to purpose? how online customer journeys differ for hedonic versus utilitarian purchases. *Journal of Marketing*, *84*(4), 127–146.

Little, J. D. (2004). Comments on "models and managers: the concept of a decision calculus". *Management Science*, *50*(12_supplement), 1854–1860.

Marketing attribution models. (2020, Feb). Retrieved from https://odysseyattribution.co/academy/marketing-attribution-models/#:~:text=touch\%20attribution\%20models-,Linear,the\%20position\%20of\%20the\%20click.

Markov model for online multi-channel attribution [r package channelattribution version 2.0.5]. (2022, Feb). Comprehensive R Archive Network (CRAN). Retrieved from https://CRAN.R-project.org/package=ChannelAttribution

Mauldin, R. D., Urbanski, M., & Urbański, M. (2003). *Graph directed markov systems: geometry and dynamics of limit sets* (Vol. 148). Cambridge University Press.

Mei, T., Hua, X.-S., Yang, L., & Li, S. (2007). Videosense: towards effective online video advertising. In *Proceedings of the 15th acm international conference on multimedia* (pp. 1075–1084).

Neslin, S. A., Grewal, D., Leghorn, R., Shankar, V., Teerling, M. L., Thomas, J. S., & Verhoef, P. C. (2006). Challenges and opportunities in multichannel customer management. *Journal of service research*, *9*(2), 95–112.

Neslin, S. A., Gupta, S., Kamakura, W., Lu, J., & Mason, C. H. (2006). Defection detection: Measuring and understanding the predictive accuracy of customer churn models. *Journal of marketing research*, *43*(2), 204–211.

Nottorf, F. (2014). Modeling the clickstream across multiple online advertising channels using a binary logit with bayesian mixture of normals. *Electronic Commerce Research and Applications*, *13*(1), 45–55.

Papadimitriou, P., Garcia-Molina, H., Krishnamurthy, P., Lewis, R. A., & Reiley, D. H. (2011). Display advertising impact: Search lift and social influence. In *Proceedings of the 17th acm sigkdd international conference on knowledge discovery and data mining* (pp. 1019–1027).

Ratliff, J. D., & Rubinfeld, D. L. (2010). Online advertising: Defining relevant markets. *Journal of Competition Law and Economics*, *6*(3), 653–686.

Rocca, J. (2021, Mar). *Introduction to markov chains.* Retrieved from https://towardsdatascience.com/brief-introduction-to-markov-chains-2c8cab9c98ab

Shao, X., & Li, L. (2011). Data-driven multi-touch attribution models. In *Proceedings of the 17th acm sigkdd international conference on knowledge discovery and data mining* (pp. 258–264).

Styan, G. P., & Smith Jr, H. (1964). Markov chains applied to marketing. *Journal of Marketing Research*, *1*(1), 50–55.

Teradata Corporation. (2013). *Data-driven: How marketers profit from technology in a multi-channel world.* Retrieved from http://www.ecircle.com/fileadmin/

files/pdfs/04\Resource\Centre/4.4.\Studien/UK/Teradata-eCircle\Data
-DrivenMarketing-Survey-2013\UK.pdf]

Van der Veen, G., & van Ossenbruggen, R. (2015). Mapping out the customer's journey: Customer search strategy as a basis for channel management. *Journal of Marketing Channels*, *22*(3), 202–213.

Wiesel, T., Pauwels, K., & Arts, J. (2011). Practice prize paper—marketing's profit impact: Quantifying online and off-line funnel progression. *Marketing Science*, *30*(4), 604–611.

Wolny, J., & Charoensuksai, N. (2014). Mapping customer journeys in multichannel decision-making. *Journal of Direct, Data and Digital Marketing Practice*, *15*(4), 317–326.

Xu, L., Duan, J. A., & Whinston, A. (2014). Path to purchase: A mutually exciting point process model for online advertising and conversion. *Management Science*, *60*(6), 1392–1412.

A Appendix

A.1 Table Appendix

Table A1: Description Online Advertising Channels

Channel	Description
Type-in	Users access the advertiser's website by either using a URL or by locating a bookmark, favourite or shortcut.
SEO	Users type in a keyword in a search engine and they obtain organic search results ranked by search algorithm.
SEA	Users type in a keyword in a search engine and they obtain sponsored search results.
Price Comparison	Price comparison websites show the features and prices of products so that users can use this website for comparison. The website directs users to the advertiser's website.
Display	Entails embedding a graphical object with the advertising message into a website. The timing and exposures of display banners are determined by the firm.
Newsletter or E-mail	Sending marketing messages toward potential customers using e-mail, also known as e-mail marketing.
Retargeting	Is a subclass of display advertising that is personalized towards the user based on his or her browsing history. It aims to re-engage users who have visited a website but did not complete a purchase.
Social Media	Comprises a set of advertising platforms belonging to the field of social media.
Affiliate	A business (e.g., retailer) rewards the affiliate (e.g., a product review website) for referring a user toward the business's website.
Referrer	Covers all traffic that is forwarded by external content websites, for example by including a text link.
Video	Video-oriented websites such as Youtube and Google Videos insert advertisements at the beginning of the end of a video.
Partner Website	Traffic coming from a virtual showroom run by an offline partner retailer.
Other	All other types that do not fit into one of the categories above.

Table A2: Research Online Advertising Channels

Research	Channels
Mei et al. (2007)	Video
Shao and Li (2011)	SEO, SEA, Display, Social Media, Video
Breuer, Brettel, and Engelen (2011)	Price Comparison, Display, Newsletter
Papadimitriou, Garcia-Molina, Krishnamurthy, Lewis, and Reiley (2011); Abhishek, Fader, and Hosanagar (2012); Nottorf (2014)	SEA, Display
Klapdor (2013)	SEO, SEA, Display, Newsletter, affiliate, referrer
H. Li and Kannan (2014)	Type-in, SEO, SEA, Display, Newsletter, Referrer
Xu, Duan, and Whinston (2014)	SEA, Display, Other
Lewis and Nguyen (2014)	SEO, SEA, Display
Anderl, Becker, et al. (2016)	Type-in, SEO, SEA, Price Comparison, Display, Newsletter, Retargeting, Social Media, Affiliate, Referrer, Other
Anderl, Schumann, and Kunz (2016)	Type-in, SEO, SEA, Price Comparison, Display, Newsletter, Retargeting, Affiliate, Partner Website, Other

Table A3: Logit Model

Channel	B	Exp(B)	ME
Eta	0.019*	1.019	0.003*
Iota	0.010*	1.010	0.002*
Alpha	-0.003	0.997	-0.001
Beta	0.009	1.009	0.002
Theta	-0.000	1.000	-0.000
Lambda	0.010	1.010	0.002
Gamma	0.027	1.027	0.005
Epsilon	0.024	1.025	0.004
Kappa	0.020	1.022	0.004
Zeta	0.006	1.006	0.001
Intercept	-1.266***	0.282	

* $p < 0.05$
** $p < 0.01$
*** $p < 0.001$

Table A4: Removal Effect Conversions

Channel	Markov 1	Markov 2	Markov 3	Markov 4
Eta	0.354	0.327	0.326	0.329
Iota	0.399	0.365	0.367	0.357
Alpha	**0.566**	**0.546**	**0.557**	**0.559**
Beta	0.256	0.227	0.232	0.223
Theta	0.214	0.170	0.180	0.186
Lambda	0.131	0.110	0.113	0.114
Kappa	0.029	0.026	0.021	0.019
Zeta	0.043	0.037	0.034	0.038
Epsilon	0.062	0.063	0.053	0.062
Gamma	0.017	0.015	0.018	0.018
Average	0.207	0.189	0.190	0.190

bold = highest removal effect for each model

Table A5: Average standard deviation Removal effect

Markov 1	Markov 2	Markov 3	Markov 4
0.186	0.177	0.181	0.179

Table A6: Number of Conversions

Channel	First-click	Last-click	Linear-click	Markov 1	Markov 2	Markov 3	Markov 4
Eta	3,164	4,167	3,538.796	3,376.024	3,430.428	3,389.486	3,415.292
Iota	4,600	3,352	3,853.249	3,808.293	3,826.061	3,820.956	3,711.660
Alpha	**6,308**	**8,447**	**7,574.102**	**5,407.690**	**5,722.204**	**5,795.932**	**5,805.055**
Beta	2,830	988	2,082.800	2,442.332	2,383.448	2,415.342	2,314.495
Theta	1,605	653	1,022.101	2,040.311	1,785.173	1,872.669	1,933.450
Lambda	900	1,205	1,033.849	1,253.581	1,153.125	1,174.310	1,180.769
Kappa	74	230	137.964	280.975	270.188	217.959	192.875
Zeta	27	107	136.010	406.333	385.983	351.403	390.454
Epsilon	99	531	272.087	592.209	660.997	551.570	639.779
Gamma	165	92	121.041	164.262	154.393	182.374	188.170

bold = highest number of conversion for each model

Table A7: Transition probabilities first-order Markov

Channel	Eta	Iota	Alpha	Beta	Theta	Lambda	Gamma	Epsilon	Kappa	Zeta	Conversion	Null
Start	0.161	0.231	0.327	0.137	0.084	0.043	0.008	0.005	0.003	0.001	-	-
Eta	-	0.089	0.132	0.129	0.046	0.023	0.002	0.031	0.010	0.010	0.120	0.408
Iota	0.102	-	0.183	0.121	0.094	0.063	0.007	0.027	0.013	0.036	0.078	**0.276**
Alpha	0.041	0.109	-	0.035	0.062	0.037	0.003	0.020	0.005	0.009	0.149	**0.530**
Beta	**0.391**	0.181	0.127	-	0.070	0.029	0.004	0.015	0.005	0.015	0.037	0.126
Theta	0.079	0.229	**0.362**	0.084	-	0.048	0.005	0.032	0.015	0.012	0.032	0.102
Lambda	0.046	0.132	0.170	0.048	0.085	-	0.006	0.038	0.037	0.019	0.098	**0.321**
Gamma	0.088	0.168	**0.228**	0.082	0.080	0.066	-	0.014	-	0.011	0.057	0.206
Epsilon	0.087	0.108	0.190	0.050	0.058	0.064	0.007	-	0.011	0.027	0.095	0.303
Kappa	0.096	0.136	0.111	0.049	0.076	0.105	0.005	0.024	-	0.019	0.093	**0.286**
Zeta	0.081	**0.291**	0.194	0.095	0.068	0.083	0.002	0.048	0.012	-	0.030	0.096
Conversion	-	-	-	-	-	-	-	-	-	-	-	1.000
Null	-	-	-	-	-	-	-	-	-	-	-	1.000

bold = edited number compared to table 12

Table A8: Logit Model

	Impulsive Journeys			Balanced Journeys			Considered Journeys		
Channel	B	Exp(B)	ME	B	Exp(B)	ME	B	Exp(B)	ME
Eta	0.070**	1.073	0.012**	0.023	1.023	0.004	0.007	1.007	0.001
Iota	0.003	1.003	0.001	0.005	1.005	0.001	0.005	1.005	0.001
Alpha	0.040**	1.041	0.007**	-0.031	0.969	-0.006	-0.002	0.998	-0.000
Beta	0.075**	1.078	0.013**	0.005	1.005	0.001	0.001	1.001	0.000
Theta	-0.001	0.999	-0.000	-0.020	0.980	-0.004	-0.002	0.998	-0.000
Lambda	0.153***	1.165	0.027***	-0.006	0.994	-0.001	0.001	1.001	0.000
Gamma	0.049	1.050	0.009	0.019	1.019	0.003	0.027	1.028	0.005
Epsilon	0.094	1.099	0.016	0.003	1.003	0.001	0.004	1.004	0.001
Kappa	0.121	1.129	0.021	0.061	1.063	0.011	-0.020	0.980	-0.004
Zeta	0.141	1.152	0.025	0.021	1.022	0.004	-0.003	0.997	-0.000
Intercept	-1.355***	0.258		-1.193***	0.303		-1.168***	0.311	

Type of journey is based on the Length of Journeys
* $p < 0.05$
** $p < 0.01$
*** $p < 0.001$

Table A9: Removal Effect

	Impulsive Journeys				Balanced Journeys				Considered Journeys			
Channel	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4
Eta	0.292	0.310	0.315	0.315	0.245	0.227	0.238	0.247	0.560	0.535	0.501	0.483
Iota	0.246	0.242	0.244	0.244	0.419	0.389	0.384	0.401	**0.745**	**0.747**	**0.708**	**0.707**
Alpha	**0.461**	**0.466**	**0.470**	**0.470**	**0.630**	**0.648**	**0.661**	**0.672**	0.651	0.643	0.633	0.654
Beta	0.147	0.149	0.151	0.151	0.241	0.213	0.222	0.237	0.579	0.553	0.499	0.506
Theta	0.090	0.084	0.085	0.085	0.238	0.218	0.193	0.216	0.500	0.475	0.452	0.441
Lambda	0.062	0.062	0.059	0.059	0.135	0.115	0.122	0.115	0.372	0.307	0.272	0.262
Kappa	0.010	0.009	0.010	0.010	0.035	0.029	0.034	0.038	0.089	0.086	0.076	0.061
Zeta	0.006	0.005	0.006	0.006	0.034	0.033	0.038	0.047	0.181	0.160	0.144	0.152
Epsilon	0.017	0.020	0.018	0.018	0.068	0.074	0.075	0.079	0.194	0.200	0.190	0.196
Gamma	0.013	0.011	0.008	0.008	0.017	0.012	0.013	0.020	0.042	0.046	0.039	0.040
Average	0.134	0.135	0.137	0.137	0.206	0.196	0.198	0.207	0.391	0.375	0.351	0.348

bold = highest removal effect for each model
Type of journey based on the Length of Journeys

Table A10: Number of Conversions

	Impulsive Journeys				Balanced Journeys				Considered Journeys			
Channel	First	Last	Linear	M4	First	Last	Linear	M4	First	Last	Linear	M4
Eta	2,570	3,007	2,756.667	2,806.499	346	658	459.167	588.027	248	502	322.962	371.800
Iota	2,170	1,966	2,070.667	2,168.837	1,369	834	1,035.333	956.105	**1,061**	552	**747.249**	**544.468**
Alpha	**4,387**	**5,254**	**4,852.000**	**4,180.227**	**1,701**	**2,426**	**2,149.717**	**1,602.486**	220	**767**	572.385	504.113
Beta	1,560	711	1,167.833	1,346.175	679	158	473.067	565.583	591	119	441.900	377.092
Theta	740	211	453.333	751.811	555	218	350.050	516.207	310	224	218.718	339.714
Lambda	519	644	573.333	527.449	190	293	245.600	274.936	191	268	214.916	201.777
Kappa	41	77	57.167	82.660	19	92	51.400	90.897	14	61	29.397	47.302
Zeta	5	44	28.167	55.107	8	39	52.883	111.097	14	24	54.960	117.428
Epsilon	56	166	108.333	157.447	34	207	99.067	187.406	9	158	64.687	145.875
Gamma	103	71	83.500	74.788	40	16	24.717	48.254	22	5	12.825	30.432

bold = highest removal effect for each model
Type of journey is based on the Length of Journeys

Table A11: Logit Model

	Impulsive Journeys			Balanced Journeys			Considered Journeys		
Channel	B	Exp(B)	ME	B	Exp(B)	ME	B	Exp(B)	ME
Eta	0.016	1.016	0.003	0.029*	1.030	0.005*	-0.002	0.998	-0.000
Iota	0.003	1.003	0.001	0.008	1.008	0.001	0.008	1.008	0.001
Alpha	-0.003	0.997	-0.000	0.001	1.001	0.000	0.003	1.003	0.001
Beta	0.002	1.002	0.000	0.008	1.008	0.001	0.003	1.003	0.001
Theta	0.074	1.077	0.013	-0.012	0.988	-0.002	-0.009	0.991	-0.002
Lambda	0.092	1.097	0.016	0.007	1.007	0.001	0.002	1.002	0.000
Gamma	-0.047	0.954	-0.01	0.067	1.069	0.012	-0.049	0.952	-0.009
Epsilon	0.098	1.103	0.017	0.007	1.007	0.001	0.009	1.010	0.002
Kappa	0.044	1.045	0.008	0.045	1.046	0.008	-0.023	0.978	-0.004
Intercept	-1.282***	0.278		-1.255***	0.285		-1.149***	0.317	

Type of journey is based on the Number of Channels
* $p < 0.05$
** $p < 0.01$
*** $p < 0.001$

Table A12: Removal Effect

	Impulsive Journeys				Balanced Journeys				Considered Journeys			
Channel	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4
Eta	0.180	0.195	0.194	0.200	0.434	0.417	0.419	0.408	0.636	0.638	0.643	0.634
Iota	0.141	0.138	0.141	0.146	**0.570**	**0.534**	**0.533**	**0.506**	**0.750**	**0.777**	**0.780**	**0.797**
Alpha	**0.564**	**0.566**	**0.569**	**0.561**	0.466	0.471	0.484	0.481	0.660	0.683	0.713	0.738
Beta	0.063	0.055	0.049	0.048	0.390	0.364	0.361	0.358	0.646	0.634	0.640	0.658
Theta	0.001	0.000	0.000	0.000	0.341	0.315	0.304	0.300	0.554	0.561	0.586	0.573
Lambda	0.046	0.041	0.040	0.040	0.172	0.149	0.134	0.128	0.439	0.411	0.375	0.358
Kappa	0.000	0.001	0.001	0.002	0.042	0.040	0.034	0.036	0.142	0.126	0.125	0.127
Zeta	0.000	0.000	0.000	0.000	0.051	0.045	0.041	0.041	0.238	0.227	0.213	0.213
Epsilon	0.002	0.002	0.001	0.000	0.079	0.081	0.082	0.077	0.303	0.312	0.307	0.303
Gamma	0.003	0.002	0.004	0.002	0.026	0.021	0.022	0.026	0.081	0.066	0.077	0.060
Average	0.111	0.111	0.111	0.111	0.257	0.244	0.241	0.236	0.445	0.444	0.446	0.446

bold = highest removal effect for each model
Type of journey based on the Number of Channels

Table A13: Number of Conversions

	Impulsive Journeys				Balanced Journeys				Considered Journeys			
Channel	First	Last	Linear	M4	First	Last	Linear	M4	First	Last	Linear	M4
Eta	1,958	1,958	1,958	2,037.756	1,042	1,914	1,363.772	1,385.923	164	295	217.024	225.622
Iota	1,424	1,424	1,424	1,488.700	**2,563**	1,719	**2,064.233**	**1,716.457**	**613**	209	**365.016**	**283.891**
Alpha	**5,742**	**5,742**	**5,742**	**5,709.474**	489	**2,293**	1,552.774	1,633.824	77	**412**	279.329	262.878
Beta	545	545	545	493.006	1,903	385	1,271.106	1,214.857	382	58	266.694	234.117
Theta	3	3	3	0	1,393	498	859.151	1,019.147	209	152	159.950	204.014
Lambda	440	440	440	403.795	362	571	455.909	433.463	98	194	137.940	127.565
Kappa	7	7	7	18.781	56	156	99.042	121.776	11	67	31.922	45.154
Zeta	0	0	0	0	18	92	89.881	139.172	9	15	46.129	75.704
Epsilon	12	12	12	4.695	79	339	189.752	260.948	8	180	70.335	107.744
Gamma	39	39	39	18.781	109	47	68.379	88.432	17	6	13.662	21.310

bold = highest removal effect for each model
Type of journey is based on the Number of Channels

A.2 Figure Appendix

Figure A1: Last-, first- and linear-click

Figure A2: Example simple Markov graph

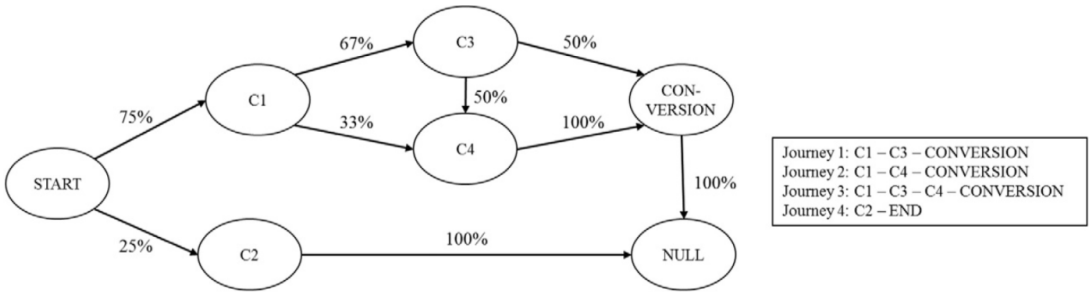


Figure A3: Positives and Negatives

		Predicted class	
		P	*N*
Actual Class	*P*	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

Figure A4: Example ROC curve

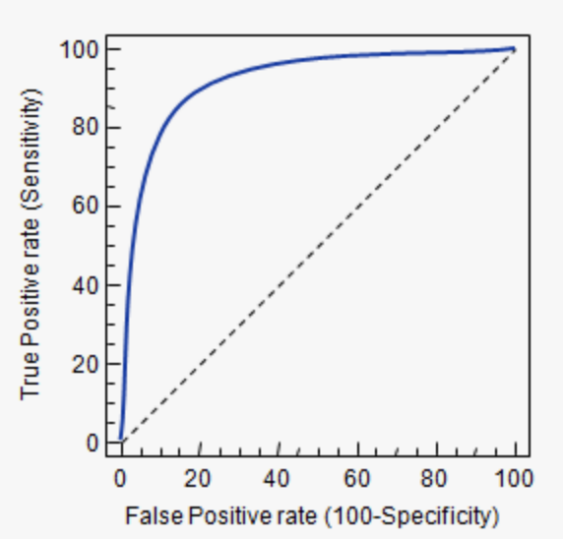


Figure A5: ROC curves

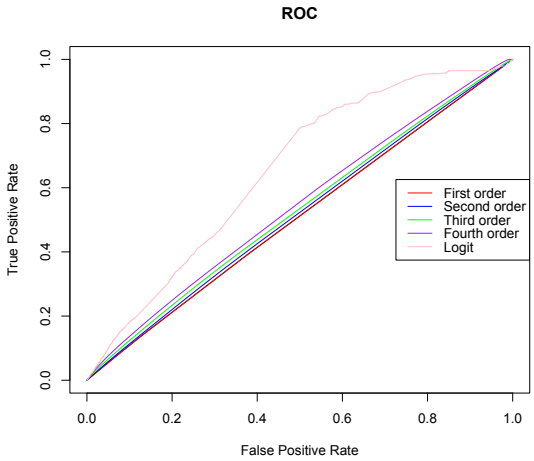


Figure A6: Number of conversions

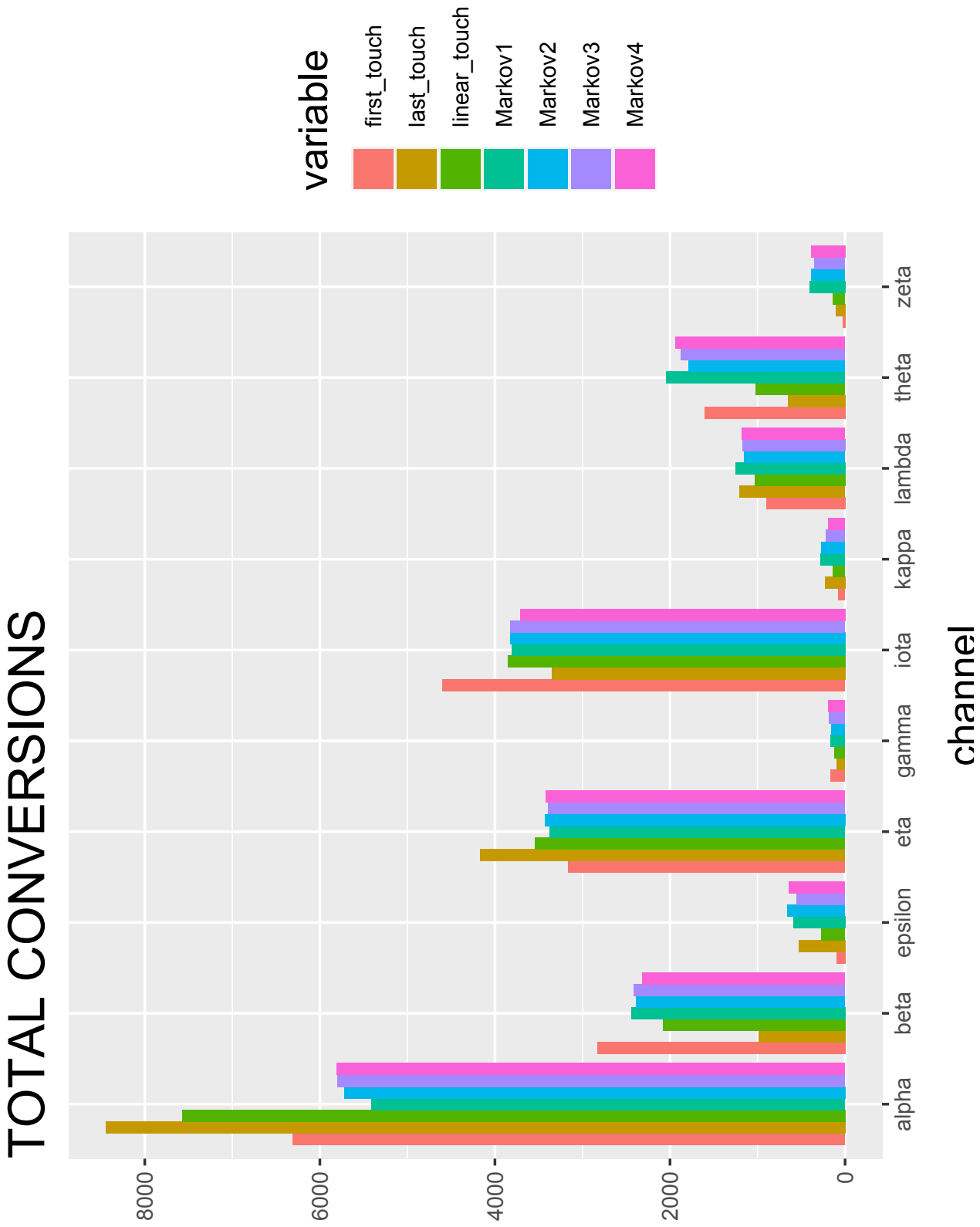


Figure A7: First-order Markov Graph

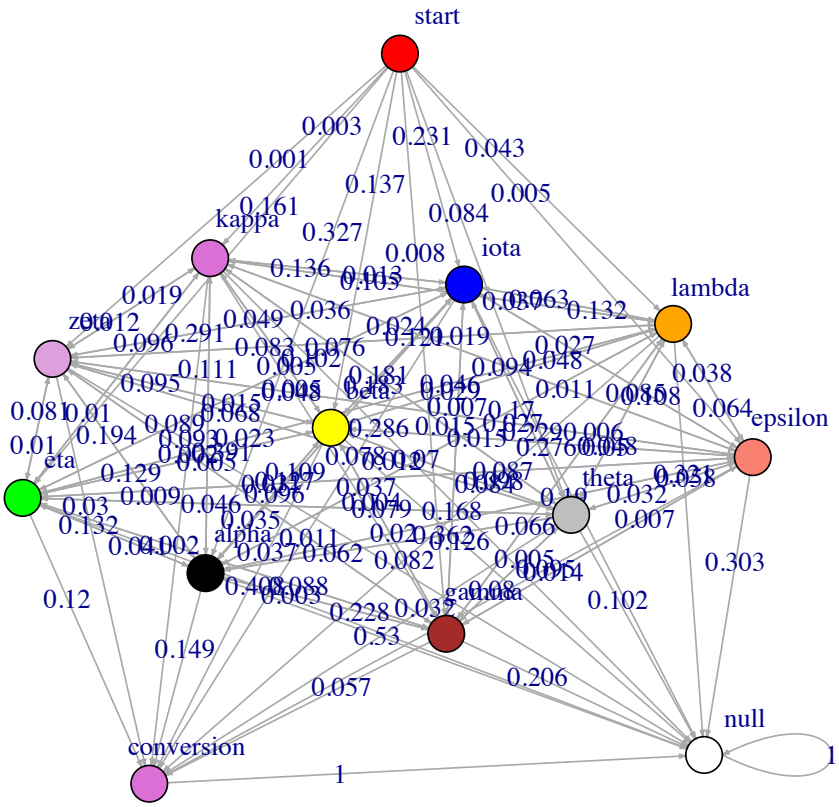
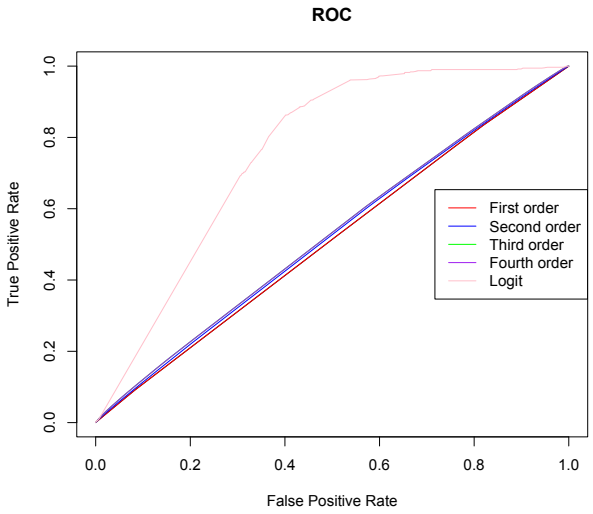
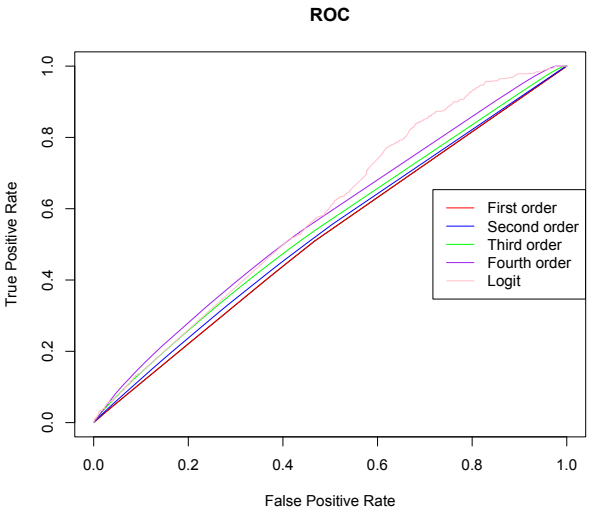


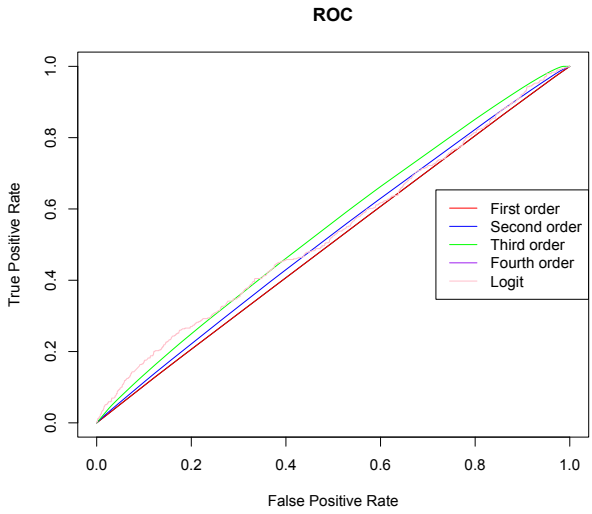
Figure A8: ROC curves - based on the Length of the Journeys



(a) Impulsive Journeys

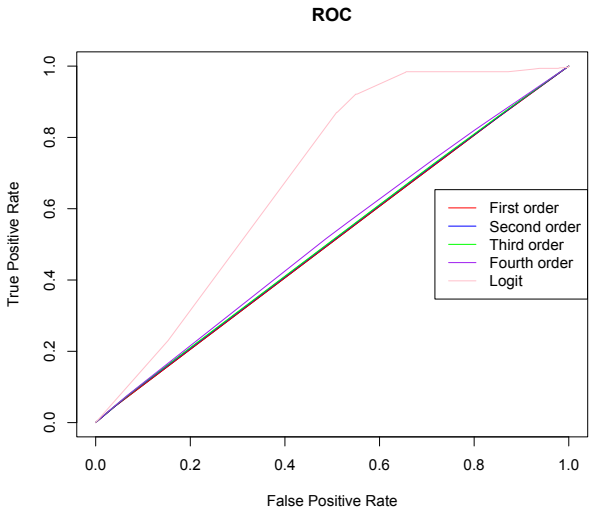


(b) Balanced Journeys

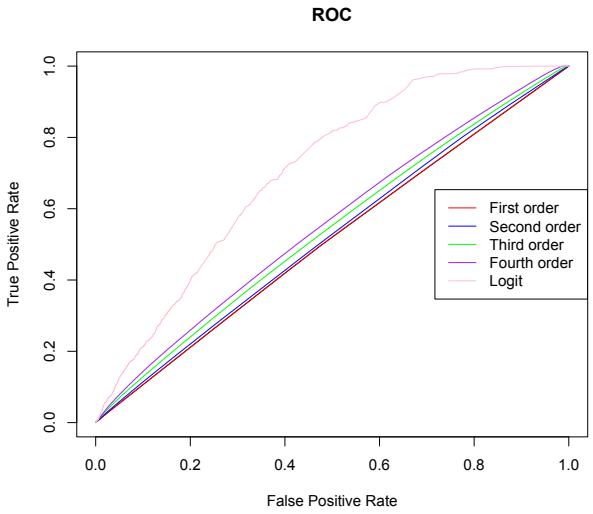


(c) Considered Journeys

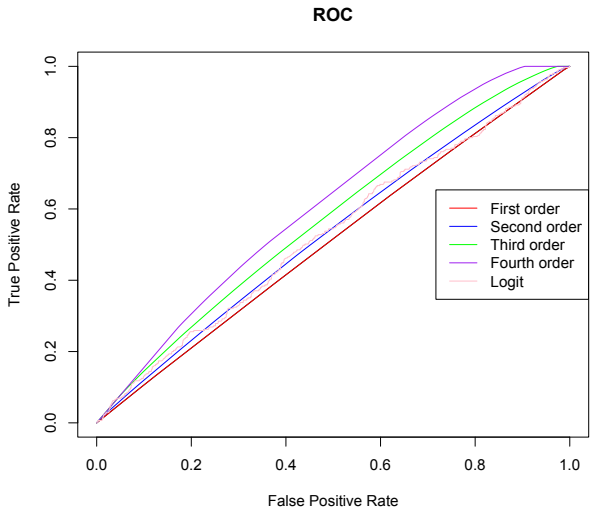
Figure A9: ROC curves - based on the Number of Channels in Journeys



(a) Impulsive Journeys



(b) Balanced Journeys



(c) Considered Journeys

A.3 Code Appendix

```
# CHANNEL ATTRIBUTION PACKAGE

.v=packageVersion("ChannelAttribution")

.onAttach = function(libname, pkgname) {

 packageStartupMessage(paste0("ChannelAttribution ",.v))
 packageStartupMessage("Looking for attribution at path level? Try
     ChannelAttributionPro! Visit https://channelattribution.io for more
     information.")

}

# HEURISTIC_MODELS
heuristic_models=function(Data, var_path, var_conv, var_value=NULL, sep=">"){

 if(!("data.frame"%in%class(Data)|"data.table"%in%class(Data))){
  print("Data must be a data.frame or a data.table")
 }

 if(is.character(var_path)){
  if(!var_path%in%names(Data)){
   print("var_path must be a column of Data")
  }
 }else{
  print("var_path must be a string")
 }
 if(is.character(var_conv)){
  if(!var_conv%in%names(Data)){
   print("var_conv must be a column of Data")
  }
 }else{
  print("var_conv must be a string")
 }

 if(!is.null(var_value)){
  if(!var_value%in%names(Data)){
   print("var_value must be a column of Data")
  }
 }

```

```
  if(length(sep)>1){stop("Separator must have length 1")}

  if(is.null(var_value)){var_value="0"}

  res=.Call("heuristic_models_cpp", Data, var_path, var_conv, var_value, sep)

  return(as.data.frame(res))

}

# CHOOSE_ORDER
choose_order=function(Data, var_path, var_conv, var_null, max_order=10, sep="
    ", ncore=1, roc_npt=100, plot=TRUE){

  if(!("data.frame"%in%class(Data)|"data.table"%in%class(Data))){
   print("Data must be a data.frame or a data.table")
  }

  if(is.character(var_path)){
   if(!var_path%in%names(Data)){
    print("var_path must be a column of Data")
   }
  }else{
   print("var_path must be a string")
  }

  if(is.character(var_conv)){
   if(!var_conv%in%names(Data)){
    print("var_conv must be a column of Data")
   }
  }else{
   print("var_conv must be a string")
  }

  if(!is.null(var_null)){
   if(!var_null%in%names(Data)){
    print("var_null must be a column of Data")
   }
  }

  if(length(sep)>1){stop("sep must have length 1")}
  if(ncore<1){stop("ncore must be >= 1")}
  if(roc_npt<10){stop("roc_npt must be >= 10")}
```

```
  if(!plot%in%c(0,1)){stop("plot must be FALSE or TRUE")}

  res=.Call("choose_order_cpp", Data, var_path, var_conv, var_null, max_order,
      sep, ncore, roc_npt)

 ck=res$auc$order[res$auc$order!=0]
  res$auc$order=res$auc$order[ck]
  res$auc$auc=res$auc$auc[ck]
  res$auc$pauc=res$auc$pauc[ck]

  best_order=res$auc$order[res$auc$pauc==max(res$auc$pauc)]

  if(best_order==max_order){
   print(paste0("Suggested order not found. Try increasing max_order."))
  }else{
   print(paste0("Suggested order: ", res$auc$order[res$auc$pauc==max(res$auc$
       pauc)]))
  }

  if(plot=="TRUE"){
   plot(res$auc$order,res$auc$pauc,type="l",xlab="order",ylab="penalized auc",
       main="PENALIZED AUC")
  }

  res[['suggested_order']]=best_order

  return(res)

}

# MARKOV_MODEL
markov_model=function(Data, var_path, var_conv, var_value=NULL, var_null=NULL,
     order=1, nsim_start=1e5, max_step=NULL, out_more=FALSE, sep=">", ncore=1,
     nfold=10, seed=0, conv_par=0.05, rate_step_sim=1.5, verbose=TRUE){


  if(!("data.frame"%in%class(Data)|"data.table"%in%class(Data))){
   print("Data must be a data.frame or a data.table")
  }

  if(is.character(var_path)){
   if(!var_path%in%names(Data)){
    print("var_path must be a column of Data")
```

```
  }
 }else{
  print("var_path must be a string")
 }

 if(is.character(var_conv)){
  if(!var_conv%in%names(Data)){
   print("var_conv must be a column of Data")
  }
 }else{
  print("var_conv must be a string")
 }

 if(!is.null(var_value)){
  if(!var_value%in%names(Data)){
   print("var_value must be a column of Data")
  }
 }

 if(!is.null(var_null)){
  if(!var_null%in%names(Data)){
   print("var_null must be a column of Data")
  }
 }

 if(order<1){stop("order must be >= 1")}
 if(nsim_start<1){stop("nsim_start must be >= 1")}
 if(!is.null(max_step)){if(max_step<1){stop("max_step must be >= 1")}}
 if(!out_more%in%c(0,1)){stop("out_more must be FALSE or TRUE")}
 if(length(sep)>1){stop("sep must have length 1")}
 if(ncore<1){stop("ncore must be >= 1")}
 if(nfold<1){stop("nfold must be >= 1")}
 if(seed<0){stop("seed must be >= 0")}
 if(conv_par<0){stop("conv_par must be > 0")}
 if(rate_step_sim<0){stop("rate_step_sim must be > 0")}
 if(!verbose%in%c(0,1)){stop("verbose must be FALSE or TRUE")}

 if(nrow(Data[which(Data[var_conv]!=0),])==0){stop("Data must have at least
     one converting path")}

 if(is.null(var_value)){var_value="0"}
 if(is.null(var_null)){var_null="0"}
 if(is.null(max_step)){max_step=0}
```

```
  if(!is.null(seed)){set.seed(seed)}

  res=.Call("markov_model_cpp", Data, var_path, var_conv, var_value, var_null,
      order, nsim_start, max_step, out_more, sep, ncore, nfold, seed, conv_par,
       rate_step_sim,verbose)

  if(out_more==FALSE){
   return(as.data.frame(res))
  }else{
   return(list(result=as.data.frame(res$result),transition_matrix=as.data.frame
       (res$transition_matrix),removal_effects=as.data.frame(res$removal_
       effects)))
  }

}

# TRANSITION_MATRIX
transition_matrix=function(Data, var_path, var_conv, var_null, order=1, sep="
    ", flg_equal=TRUE){

  if(!("data.frame"%in%class(Data)|"data.table"%in%class(Data))){
   print("Data must be a data.frame or a data.table")
  }

  if(is.character(var_path)){
   if(!var_path%in%names(Data)){
    print("var_path must be a column of Data")
   }
  }else{
   print("var_path must be a string")
  }
  if(is.character(var_conv)){
   if(!var_conv%in%names(Data)){
    print("var_conv must be a column of Data")
   }
  }else{
   print("var_conv must be a string")
  }

  if(!is.null(var_null)){
   if(!var_null%in%names(Data)){
    print("var_null must be a column of Data")
   }
```

```
200   }
201
202   if(order<1){stop("order must be >= 1")}
203   if(length(sep)>1){stop("sep must have length 1")}
204   if(!flg_equal%in%c(0,1)){stop("flg_equal must be FALSE or TRUE")}
205
206   if(is.null(var_null)){var_null="0"}
207
208   res=.Call("transition_matrix_cpp", Data, var_path, var_conv, var_null, order,
          sep, flg_equal)
209
210   return(list(channels=data.frame(id=1:length(res$channels),channel_name=res$
          channels),transition_matrix=as.data.frame(res$transition_matrix)))
211    }
212
213 # AUTO_MARKOV_MODEL
214 auto_markov_model=function(Data, var_path, var_conv, var_null, var_value=NULL,
       max_order=10, roc_npt=100, plot=FALSE, nsim_start=1e5, max_step=NULL, out
      _more=FALSE, sep=">", ncore=1, nfold=10, seed=0, conv_par=0.05, rate_step_
      sim=1.5, verbose=TRUE){
215
216   if(!("data.frame"%in%class(Data)|"data.table"%in%class(Data))){
217    print("Data must be a data.frame or a data.table")
218   }
219
220   if(is.character(var_path)){
221    if(!var_path%in%names(Data)){
222     print("var_path must be a column of Data")
223    }
224   }else{
225    print("var_path must be a string")
226   }
227   if(is.character(var_conv)){
228    if(!var_conv%in%names(Data)){
229     print("var_conv must be a column of Data")
230    }
231   }else{
232    print("var_conv must be a string")
233   }
234
235   if(!is.null(var_value)){
236    if(!var_value%in%names(Data)){
237     print("var_value must be a column of Data")
```

```
238     }
239   }
240
241   if(!is.null(var_null)){
242     if(!var_null%in%names(Data)){
243       print("var_null must be a column of Data")
244     }
245   }
246
247   if(max_order<1){stop("max_order must be >= 1")}
248   if(roc_npt<10){stop("roc_npt must be >= 10")}
249   if(!plot%in%c(0,1)){stop("plot must be FALSE or TRUE")}
250   if(nsim_start<1){stop("nsim_start must be >= 1")}
251   if(!is.null(max_step)){if(max_step<1){stop("max_step must be >= 1")}}
252   if(!out_more%in%c(0,1)){stop("out_more must be FALSE or TRUE")}
253   if(length(sep)>1){stop("sep must have length 1")}
254   if(ncore<1){stop("ncore must be >= 1")}
255   if(nfold<1){stop("nfold must be >= 1")}
256   if(seed<0){stop("seed must be >= 0")}
257   if(conv_par<0){stop("conv_par must be > 0")}
258   if(rate_step_sim<0){stop("rate_step_sim must be > 0")}
259   if(!verbose%in%c(0,1)){stop("verbose must be FALSE or TRUE")}
260
261   order=choose_order(Data, var_path, var_conv, var_null, max_order=max_order,
          sep=sep, ncore=ncore, roc_npt=roc_npt, plot=plot)
262   order=order[['suggested_order']]
263
264   res=markov_model(Data, var_path, var_conv, var_value=var_value, var_null=var_
          null, order=order, nsim_start=nsim_start, max_step=max_step, out_more=out
          _more, sep=sep, ncore=ncore, nfold=nfold, seed=seed, conv_par=conv_par,
          rate_step_sim=rate_step_sim, verbose=verbose)
265
266   if(out_more==FALSE){
267     return(as.data.frame(res))
268   }else{
269     return(list(result=as.data.frame(res$result),transition_matrix=as.data.frame
          (res$transition_matrix),removal_effects=as.data.frame(res$removal_
          effects)))
270   }
271
272 }
273
274 # REMOVE JOURNEYS WITH DELTA AND MI
```

```
# Delta present in observations: 659, 1702, 1889, 2392, 3020, 5045, 5889, 5933, 6404, 6504, 6977, 7232, 7496
# Mi present in observations: 2094, 3033
Data2 <- Data[-c(659, 1702, 1889, 2392, 3020, 5045, 5889, 5933, 6404, 6504, 6977, 7232, 7496, 2094, 3033), ]

# HEURISTICS
Heuristics2 <- heuristic_models(Data2, var_path = 'path', var_conv = 'total_conversions', var_value='total_conversion_value')

# MARKOV 1 TO 4
Markov1_2 <- markov_model(Data2, var_path = "path", var_conv = "total_conversions",
var_value="total_conversion_value", var_null="total_null", order = 1)

Markov2_2 <- markov_model(Data2, var_path = "path", var_conv = "total_conversions",
var_value="total_conversion_value", var_null="total_null", order = 2)

Markov3_2 <- markov_model(Data2, var_path = "path", var_conv = "total_conversions",
var_value="total_conversion_value", var_null="total_null", order = 3)

Markov4_2 <- markov_model(Data2, var_path = "path", var_conv = "total_conversions",
var_value="total_conversion_value", var_null="total_null", order = 4)

# LOGIT
# Imported an excel file with the number of occurrences of each channel in each observation -> file is called Channel_occurences
Channels_occurences_2 <- Channels_occurences[-c(659, 1702, 1889, 2392, 3020, 5045, 5889, 5933, 6404, 6504, 6977, 7232, 7496, 2094, 3033), ]
logit2 <- glm(formula = cbind(Data2$total_conversions, Data2$total_null) ~ (Channels_occurences_2$eta + Channels_occurences_2$iota + Channels_occurences_2$alpha + Channels_occurences_2$beta + Channels_occurences_2$theta + Channels_occurences_2$lambda + Channels_occurences_2$kappa + Channels_occurences_2$zeta + Channels_occurences_2$epsilon + Channels_occurences_2$gamma), family = binomial("logit"), data = Data2)
summary(logit2)
exp(logit2$coefficients)
marg2 <- logitmfx(formula = cbind(Data2$total_conversions, Data2$total_null) ~ (Channels_occurences_2$eta + Channels_occurences_2$iota + Channels_occurences_2$alpha + Channels_occurences_2$beta + Channels_occurences_2$
```

```
    theta + Channels_occurences_2$lambda + Channels_occurences_2$kappa +
    Channels_occurences_2$zeta + Channels_occurences_2$epsilon + Channels_
    occurences_2$gamma), data = Data2)
302
303 ## PLOT THE NUMBER OF CONVERSIONS
304 # Rename the columns
305 colnames(Heuristics2) <- c('channel', 'first_touch_conversions', 'first_touch_
    value', 'last_touch_conversions', 'last_touch_value', 'linear_touch_
    conversions', 'linear_touch_value' )
306 colnames(Markov1_2) <- c('channel', 'conversion1', 'value1')
307 colnames(Markov2_2) <- c('channel', 'conversion2', 'value2')
308 colnames(Markov3_2) <- c('channel', 'conversion3', 'value3')
309 colnames(Markov4_2) <- c('channel', 'conversion4', 'value4')
310
311 # Merge Heuristics and Markov Models
312 H_M1_2 <- merge(Heuristics2, Markov1_2, by="channel")
313 H_M1_M2_2 <- merge(H_M1_2, Markov2_2, by="channel")
314 H_M1_M2_M3_2 <- merge(H_M1_M2_2, Markov3_2, by = "channel")
315 H_M_2 <- merge(H_M1_M2_M3_2, Markov4_2, by = "channel")
316
317 # Conversion
318 # Select conversion columns
319 Conversion_H_M_2 <- H_M_2[,colnames(H_M_2)%in%c('channel', 'first_touch_
    conversions', 'last_touch_conversions', 'linear_touch_conversions', '
    conversion1', 'conversion2', 'conversion3', 'conversion4')]
320
321 # Rename columns
322 colnames(Conversion_H_M_2) <- c('channel', 'first_touch', 'last_touch', '
    linear_touch', 'Markov1', 'Markov2', 'Markov3', 'Markov4')
323
324 # Transforms the dataset into a data frame that ggplot2 can use to graph the
    outcomes
325 Conversion_H_M_2 <- melt(Conversion_H_M_2, id='channel')
326
327 # Plot the total conversions
328 ggplot(Conversion_H_M_2 , aes(channel, value, fill = variable)) +
329   geom_bar(stat='identity', position='dodge') +
330   ggtitle('TOTAL CONVERSIONS') +
331   theme(axis.title.x = element_text(vjust = -2)) +
332   theme(axis.title.y = element_text(vjust = +2)) +
333   theme(title = element_text(size = 16)) +
334   theme(plot.title=element_text(size = 20)) +
335   ylab("")
```

```
336
337 # ROC CURVES
338 res2=choose_order(Data2, var_path="path", var_conv="total_conversions", var_
    null="total_null")
339 ROC_logit <- roc(test_logit$bin_conv, predicted_log)
340
341 fpr1_2 = res2[["roc"]][["1"]][["fpr"]]
342 trp1_2 = res2[["roc"]][["1"]][["tpr"]]
343 fpr2_2 = res2[["roc"]][["2"]][["fpr"]]
344 trp2_2 = res2[["roc"]][["2"]][["tpr"]]
345 fpr3_2 = res2[["roc"]][["3"]][["fpr"]]
346 trp3_2 = res2[["roc"]][["3"]][["tpr"]]
347 fpr4_2 = res2[["roc"]][["4"]][["fpr"]]
348 trp4_2 = res2[["roc"]][["4"]][["tpr"]]
349 trp_logit = ROC_logit[["sensitivities"]]
350 spec_logit <- ROC_logit[["specificities"]]
351 frp_logit = c(1) - spec_logit
352
353 plot(fpr1_2,trp1_2,type="l",xlab="False Positive Rate",ylab="True Positive
    Rate",main="ROC")
354 lines(fpr1_2,trp1_2,col="red")
355 lines(fpr2_2,trp2_2,col="blue")
356 lines(fpr3_2,trp3_2,col="green")
357 lines(fpr4_2,trp4_2,col="purple")
358 lines(frp_logit, trp_logit, col="pink")
359
360 legend("right", legend=c("First order","Second order", "Third order","Fourth
    order", "Logit"),
361        col=c("red","blue", "green","purple", "pink"),lty=1)
362
363 # REMOVAL EFFECT + TRANSITION MATRIX
364 # Calculate Removal effect + transition matrix/probability
365 m1_2 = markov_model(Data2, "path", "total_conversions", var_value="total_
    conversion_value", var_null="total_null", out_more=TRUE, order = 1)
366 m2_2 = markov_model(Data2, "path", "total_conversions", var_value="total_
    conversion_value", var_null="total_null", out_more=TRUE, order = 2)
367 m3_2 = markov_model(Data2, "path", "total_conversions", var_value="total_
    conversion_value", var_null="total_null", out_more=TRUE, order = 3)
368 m4_2 = markov_model(Data2, "path", "total_conversions", var_value="total_
    conversion_value", var_null="total_null", out_more=TRUE, order = 4)
369
370 # Calculate standard deviation of removal effect
```

```
removal_effects_m1_2 <- m1_2[["removal_effects"]][["removal_effects_conversion"]]
sum(removal_effects_m1_2)/10
sd(removal_effects_m1_2)
sd(removal_effects_m1_2)/(sum(removal_effects_m1_2)/10)

removal_effects_m2_2 <- m2_2[["removal_effects"]][["removal_effects_conversion"]]
sum(removal_effects_m2_2)/10
sd(removal_effects_m2_2)
sd(removal_effects_m2_2)/(sum(removal_effects_m2_2)/10)

removal_effects_m3_2 <- m3_2[["removal_effects"]][["removal_effects_conversion"]]
sum(removal_effects_m3_2)/10
sd(removal_effects_m3_2)
sd(removal_effects_m3_2)/(sum(removal_effects_m3_2)/10)

removal_effects_m4_2 <- m4_2[["removal_effects"]][["removal_effects_conversion"]]
sum(removal_effects_m4_2)/10
sd(removal_effects_m4_2)
sd(removal_effects_m4_2)/(sum(removal_effects_m4_2)/10)

# MARKOV GRAPH FIRST ORDER
# Create Markov Chain first-order
statesNames=c("start","eta", "iota", "alpha", "beta", "theta", "lambda", "gamma", "epsilon", "kappa", "zeta", "conversion", " null")
MarkovChain <-new("markovchain", transitionMatrix=matrix(c(
0, 0.161, 0.231, 0.327, 0.137, 0.084, 0.043, 0.008, 0.005, 0.003,  0.001, 0, 0,
0, 0, 0.089, 0.132, 0.129, 0.046, 0.023, 0.002, 0.031, 0.010, 0.010, 0.120, 0.408,
0, 0.102, 0, 0.183, 0.121, 0.094, 0.063, 0.007, 0.027, 0.013, 0.036, 0.078, 0.276,
0, 0.041, 0.109, 0, 0.035, 0.062, 0.037, 0.003, 0.020, 0.005, 0.009, 0.149, 0.530,
0, 0.391, 0.181, 0.127, 0, 0.070, 0.029, 0.004, 0.015, 0.005, 0.015, 0.037, 0.126,
0, 0.079, 0.229, 0.362, 0.084, 0, 0.048, 0.005, 0.032, 0.015, 0.012, 0.032, 0.102,
0, 0.046, 0.132, 0.170, 0.048, 0.085, 0, 0.006, 0.038, 0.037, 0.019, 0.098, 0.321,
```

```
0, 0.088, 0.168, 0.228, 0.082, 0.080, 0.066, 0, 0.014, 0, 0.011, 0.057, 0.206,
0, 0.087, 0.108, 0.190, 0.050, 0.058, 0.064, 0.007, 0, 0.011, 0.027, 0.095,
    0.303,
0, 0.096, 0.136, 0.111, 0.049, 0.076, 0.105, 0.005, 0.024, 0, 0.019, 0.093,
    0.286,
0, 0.081, 0.291, 0.194, 0.095, 0.068, 0.083, 0.002, 0.048, 0.012, 0, 0.030,
    0.096,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1), byrow=TRUE, nrow=13, dimnames=list(
    statesNames,statesNames)))

graph <- as(MarkovChain, "igraph")
V(graph)["start"]$color<-"red"
V(graph)["eta"]$color<-"green"
V(graph)["iota"]$color<-"blue"
V(graph)["alpha"]$color<-"black"
V(graph)["beta"]$color<-"yellow"
V(graph)["theta"]$color<-"gray"
V(graph)["lambda"]$color<-"orange"
V(graph)["gamma"]$color<-"brown"
V(graph)["epsilon"]$color<-"salmon"
V(graph)["kappa"]$color<-"orchid"
V(graph)["zeta"]$color<-"plum"
V(graph)["conversion"]$color<-"orchid"

plot(graph, vertex.color=V(graph)$color, vertex.size=10, vertex.label.cex=1,
        vertex.label.dist=2, edge.arrow.size=0.2, edge.label=E(graph)$prob)

# Attribution vs heuristics in %
Heuristics2$first_touch_conversions/sum(Heuristics2$first_touch_conversions)*
    100
Heuristics2$last_touch_conversions/sum(Heuristics2$last_touch_conversions)*100
Heuristics2$linear_touch_conversions/sum(Heuristics2$linear_touch_conversions)
    *100
Markov1_2$conversion1/sum(Markov1_2$conversion1)*100
Markov2_2$conversion2/sum(Markov2_2$conversion2)*100
Markov3_2$conversion3/sum(Markov3_2$conversion3)*100
Markov4_2$conversion4/sum(Markov4_2$conversion4)*100

chisq_first_conversion2 <- chisq.test(Heuristics2$first_touch_conversions/sum(
    Heuristics2$first_touch_conversions)*100)
chisq_last_conversion2 <- chisq.test(Heuristics2$last_touch_conversions/sum(
    Heuristics2$last_touch_conversions)*100)
```

```
chisq_linear_conversion2 <- chisq.test(Heuristics2$linear_touch_conversions/
    sum(Heuristics2$linear_touch_conversions)*100)
chisq_markov1_conversion2 <- chisq.test(Markov1_2$conversion1/sum(Markov1_2$
    conversion1)*100)
chisq_markov2_conversion2 <- chisq.test(Markov2_2$conversion2/sum(Markov2_2$
    conversion2)*100)
chisq_markov3_conversion2 <- chisq.test(Markov3_2$conversion3/sum(Markov3_2$
    conversion3)*100)
chisq_markov4_conversion2 <- chisq.test(Markov4_2$conversion4/sum(Markov4_2$
    conversion4)*100)

# PREDICTIVE ACCURACY
# Top Decile Lift
Expected_first_conversion2 = chisq_first_conversion2[["expected"]]
Observed_first_conversion2 = chisq_first_conversion2[["observed"]]
TDL_first_conversion2 <- TopDecileLift(Expected_first_conversion2, Observed_
    first_conversion2)

Expected_last_conversion2 = chisq_last_conversion2[["expected"]]
Observed_last_conversion2 = chisq_last_conversion2[["observed"]]
TDL_last_conversion2 <- TopDecileLift(Expected_last_conversion2, Observed_last
    _conversion2)

Expected_linear_conversion2 = chisq_linear_conversion2[["expected"]]
Observed_linear_conversion2 = chisq_linear_conversion2[["observed"]]
TDL_linear_conversion2 <- TopDecileLift(Expected_linear_conversion2, Observed_
    linear_conversion2)

Expected_markov1_conversion2 = chisq_markov1_conversion2[["expected"]]
Observed_markov1_conversion2 = chisq_markov1_conversion2[["observed"]]
TDL_markov1_conversion2 <- TopDecileLift(Expected_markov1_conversion2,
    Observed_markov1_conversion2)

Expected_markov2_conversion2 = chisq_markov2_conversion2[["expected"]]
Observed_markov2_conversion2 = chisq_markov2_conversion2[["observed"]]
TDL_markov2_conversion2 <- TopDecileLift(Expected_markov2_conversion2,
    Observed_markov2_conversion2)

Expected_markov3_conversion2 = chisq_markov3_conversion2[["expected"]]
Observed_markov3_conversion2 = chisq_markov3_conversion2[["observed"]]
TDL_markov3_conversion2 <- TopDecileLift(Expected_markov3_conversion2,
    Observed_markov3_conversion2)

```

```
Expected_markov4_conversion2 = chisq_markov4_conversion2[["expected"]]
Observed_markov4_conversion2 = chisq_markov4_conversion2[["observed"]]
TDL_markov4_conversion2 <- TopDecileLift(Expected_markov4_conversion2,
    Observed_markov4_conversion2)

Expected_logit = sum(logit2$effects)/9985
Observed_logit = logit2$effects
TDL_logit <- TopDecileLift(Expected_logit, Observed_logit)

# AUC
# Markov models -> values found [U+FFFD]res2[U+FFFD]
res2=choose_order(Data2, var_path="path", var_conv="total_conversions",
                      var_null="total_null")
# Preferred order is 5

# Logit
Data2$conv_and_null <- Data2$total_conversions + Data2$total_null
Data2_logit <- Data2 %>% slice(rep(1:n(), Data2$conv_and_null))
Data2_logit <- Data2_logit %>%
group_by(path) %>%
mutate(row_number = 1:n())
Data2_logit$bin_conv <- ifelse(Data2_logit$row_number <= Data2_logit$total_
    conversions, 1, 0)
Data2_logit$bin_null <- ifelse(Data2_logit$bin_conv == 0, 1, 0)
Channels_occurences_logit <- Channels_occurences_2 %>% slice(rep(1:n(), Data2$
    conv_and_null))

logit_nonagg <- glm(formula = cbind(Data2_logit$bin_conv, Data2_logit$bin_null
    ) ~ (Channels_occurences_logit$eta + Channels_occurences_logit$iota +
    Channels_occurences_logit$alpha + Channels_occurences_logit$beta +
    Channels_occurences_logit$theta + Channels_occurences_logit$lambda +
    Channels_occurences_logit$kappa + Channels_occurences_logit$zeta +
    Channels_occurences_logit$epsilon + Channels_occurences_logit$gamma),
    family = binomial("logit"), data = Data2_logit)

# Use 80% of dataset as training set and remaining 20% as testing set
sample_logit <- sample(c(TRUE, FALSE), nrow(Data2_logit), replace=TRUE, prob=c
    (0.8,0.2))
train_logit <- Data2_logit[sample_logit, ]
test_logit <- Data2_logit[!sample_logit, ]
Channels_occurences_nonagg <- Channels_occurences_logit[!sample_logit, ]

# Fit logistic regression model (on train)
```

```
model_logit_nonagg2 <-  glm(formula = cbind(test_logit$bin_conv, test_logit$
    bin_null) ~ (Channels_occurences_nonagg$eta + Channels_occurences_nonagg$
    iota + Channels_occurences_nonagg$alpha + Channels_occurences_nonagg$beta
    + Channels_occurences_nonagg$theta + Channels_occurences_nonagg$lambda +
    Channels_occurences_nonagg$kappa + Channels_occurences_nonagg$zeta +
    Channels_occurences_nonagg$epsilon + Channels_occurences_nonagg$gamma),
    family = binomial("logit"), data = train_logit)

# Calculate probability of default for each individual in test dataset
predicted_log <- predict(model_logit_nonagg2, test_logit, type="response")
auc(test_logit$bin_conv, predicted_log)

# TYPES OF JOURNEYS - SPLIT MADE BY LENGTH JOURNEYS
# Make subsamples
Data2$length <- sapply(strsplit(as.character(Data2$path),">"),FUN=function(x){
    length(x[x!="Null"])})
Length_impulsive_2 <- subset(Data2, length < 4)
Length_balanced_2 <- subset(Data2, length >3 & length < 7)
Length_considered_2 <- subset(Data2, length > 6 & length < 90)

# Heuristics
Heuristics_length_impulsive_2 <- heuristic_models(Length_impulsive_2, var_path
    = 'path', var_conv = 'total_conversions', var_value='total_conversion_
    value')
Heuristics_length_balanced_2 <- heuristic_models(Length_balanced_2, var_path =
    'path', var_conv = 'total_conversions', var_value='total_conversion_value
    ')
Heuristics_length_considered_2 <- heuristic_models(Length_considered_2, var_
    path = 'path', var_conv = 'total_conversions', var_value='total_conversion
    _value')

Heuristics_length_impulsive_2$first_touch_conversions/sum(Heuristics_length_
    impulsive_2$first_touch_conversions)*100
Heuristics_length_impulsive_2$last_touch_conversions/sum(Heuristics_length_
    impulsive_2$last_touch_conversions)*100
Heuristics_length_impulsive_2$linear_touch_conversions/sum(Heuristics_length_
    impulsive_2$linear_touch_conversions)*100

Heuristics_length_balanced_2$first_touch_conversions/sum(Heuristics_length_
    balanced_2$first_touch_conversions)*100
Heuristics_length_balanced_2$last_touch_conversions/sum(Heuristics_length_
    balanced_2$last_touch_conversions)*100
```

```
Heuristics_length_balanced_2$linear_touch_conversions/sum(Heuristics_length_balanced_2$linear_touch_conversions)*100

Heuristics_length_considered_2$first_touch_conversions/sum(Heuristics_length_considered_2$first_touch_conversions)*100
Heuristics_length_considered_2$last_touch_conversions/sum(Heuristics_length_considered_2$last_touch_conversions)*100
Heuristics_length_considered_2$linear_touch_conversions/sum(Heuristics_length_considered_2$linear_touch_conversions)*100

# Markov 1
Markov1_length_impulsive_2 <- markov_model(Length_impulsive_2, var_path = "path", var_conv = "total_conversions", var_value="total_conversion_value", var_null="total_null", order = 1)
Markov1_length_balanced_2 <- markov_model(Length_balanced_2, var_path = "path", var_conv = "total_conversions", var_value="total_conversion_value", var_null="total_null", order = 1)
Markov1_length_considered_2 <- markov_model(Length_considered_2, var_path = "path", var_conv = "total_conversions", var_value="total_conversion_value", var_null="total_null", order = 1)

# Markov 2
Markov2_length_impulsive_2 <- markov_model(Length_impulsive_2, var_path = "path", var_conv = "total_conversions", var_value="total_conversion_value", var_null="total_null", order = 2)
Markov2_length_balanced_2 <- markov_model(Length_balanced_2, var_path = "path", var_conv = "total_conversions", var_value="total_conversion_value", var_null="total_null", order = 2)
Markov2_length_considered_2 <- markov_model(Length_considered_2, var_path = "path", var_conv = "total_conversions", var_value="total_conversion_value", var_null="total_null", order = 2)
Markov2_length_balanced_2$total_conversion/sum(Markov2_length_balanced_2$total_conversion)*100

# Markov 3
Markov3_length_impulsive_2 <- markov_model(Length_impulsive_2, var_path = "path", var_conv = "total_conversions", var_value="total_conversion_value", var_null="total_null", order = 3)
Markov3_length_balanced_2 <- markov_model(Length_balanced_2, var_path = "path", var_conv = "total_conversions", var_value="total_conversion_value", var_null="total_null", order = 3)
Markov3_length_considered_2 <- markov_model(Length_considered_2, var_path = "path", var_conv = "total_conversions", var_value="total_conversion_value",
```

```
    var_null="total_null", order = 3)

# Markov 4
Markov4_length_impulsive_2 <- markov_model(Length_impulsive_2, var_path = "path", var_conv = "total_conversions", var_value="total_conversion_value", var_null="total_null", order = 4)
Markov4_length_balanced_2 <- markov_model(Length_balanced_2, var_path = "path", var_conv = "total_conversions", var_value="total_conversion_value", var_null="total_null", order = 4)
Markov4_length_considered_2 <- markov_model(Length_considered_2, var_path = "path", var_conv = "total_conversions", var_value="total_conversion_value", var_null="total_null", order = 4)

Markov4_length_impulsive_2$total_conversion/sum(Markov4_length_impulsive_2$total_conversion)*100
Markov4_length_balanced_2$total_conversion/sum(Markov4_length_balanced_2$total_conversion)*100
Markov4_length_considered_2$total_conversion/sum(Markov4_length_considered_2$total_conversion)*100

Expected_logit_length_impulsive = sum(logit2_length_impulsive$effects)/3890
exp_length_impulsive = sum(predicted_log_length_impulsive)/10902
Observed_logit_length_impulsive = logit2_length_impulsive$effects
TDL_logit_length_impulsive <- TopDecileLift(exp_length_impulsive, Observed_logit_length_impulsive)

Expected_logit_length_balanced = sum(logit2_length_balanced$coefficients)/11
exp_length_balanced = sum(predicted_log_length_balanced)/
Observed_logit_length_balanced = logit2_length_balanced$effects
TDL_logit_length_balanced <- TopDecileLift(Expected_logit_length_balanced, Observed_logit)

Expected_logit_length_considered = sum(logit2_length_considered$coefficients)/11
Observed_logit_length_considered = logit2_length_considered$coefficients
TDL_logit_length_considered <- TopDecileLift(Expected_logit_length_considered, Observed_logit)

# Logit
# Imported three excel files with the number of occurrences of each channel in each observation -> files are called Channel_occurences_length_impulsive, Channel_occurences_length_balanced, Channel_occurences_length_considered
```

```
logit2_length_impulsive <- glm(formula = cbind(Length_impulsive_2$total_conversions, Length_impulsive_2$total_null) ~ (Channels_occurences_length_impulsive$eta + Channels_occurences_length_impulsive$iota + Channels_occurences_length_impulsive$alpha + Channels_occurences_length_impulsive$beta + Channels_occurences_length_impulsive$theta + Channels_occurences_length_impulsive$lambda + Channels_occurences_length_impulsive$kappa + Channels_occurences_length_impulsive$zeta + Channels_occurences_length_impulsive$epsilon + Channels_occurences_length_impulsive$gamma), family = binomial("logit"), data = Length_impulsive_2)
summary(logit2_length_impulsive)
exp(logit2_length_impulsive$coefficients)
marg2_length_impulsive <- logitmfx(formula = cbind(Length_impulsive_2$total_conversions, Length_impulsive_2$total_null) ~ (Channels_occurences_length_impulsive$eta + Channels_occurences_length_impulsive$iota + Channels_occurences_length_impulsive$alpha + Channels_occurences_length_impulsive$beta + Channels_occurences_length_impulsive$theta + Channels_occurences_length_impulsive$lambda + Channels_occurences_length_impulsive$kappa + Channels_occurences_length_impulsive$zeta + Channels_occurences_length_impulsive$epsilon + Channels_occurences_length_impulsive$gamma), data = Length_impulsive_2)

logit2_length_balanced <- glm(formula = cbind(Length_balanced_2$total_conversions, Length_balanced_2$total_null) ~ (Channels_occurences_length_balanced$eta + Channels_occurences_length_balanced$iota + Channels_occurences_length_balanced$alpha + Channels_occurences_length_balanced$beta + Channels_occurences_length_balanced$theta + Channels_occurences_length_balanced$lambda + Channels_occurences_length_balanced$kappa + Channels_occurences_length_balanced$zeta + Channels_occurences_length_balanced$epsilon + Channels_occurences_length_balanced$gamma), family = binomial("logit"), data = Length_balanced_2)
summary(logit2_length_balanced)
exp(logit2_length_balanced$coefficients)
marg2_length_balanced <- logitmfx(formula = cbind(Length_balanced_2$total_conversions, Length_balanced_2$total_null) ~ (Channels_occurences_length_balanced$eta + Channels_occurences_length_balanced$iota + Channels_occurences_length_balanced$alpha + Channels_occurences_length_balanced$beta + Channels_occurences_length_balanced$theta + Channels_occurences_length_balanced$lambda + Channels_occurences_length_balanced$kappa + Channels_occurences_length_balanced$zeta + Channels_occurences_length_balanced$epsilon + Channels_occurences_length_balanced$gamma), data = Length_balanced_2)

```

```
logit2_length_considered <- glm(formula = cbind(Length_considered_2$total_
    conversions, Length_considered_2$total_null) ~ (Channels_occurences_length
    _considered$eta + Channels_occurences_length_considered$iota + Channels_
    occurences_length_considered$alpha + Channels_occurences_length_considered
    $beta + Channels_occurences_length_considered$theta + Channels_occurences_
    length_considered$lambda + Channels_occurences_length_considered$kappa +
    Channels_occurences_length_considered$zeta + Channels_occurences_length_
    considered$epsilon + Channels_occurences_length_considered$gamma), family
    = binomial("logit"), data = Length_considered_2)
summary(logit2_length_considered)
exp(logit2_length_considered$coefficients)
marg2_length_considered <- logitmfx(formula = cbind(Length_considered_2$total_
    conversions, Length_considered_2$total_null) ~ (Channels_occurences_length
    _considered$eta + Channels_occurences_length_considered$iota + Channels_
    occurences_length_considered$alpha + Channels_occurences_length_considered
    $beta + Channels_occurences_length_considered$theta + Channels_occurences_
    length_considered$lambda + Channels_occurences_length_considered$kappa +
    Channels_occurences_length_considered$zeta + Channels_occurences_length_
    considered$epsilon + Channels_occurences_length_considered$gamma), data =
    Length_considered_2)

# ROC
# Impulsive
res_length_impulsive_2=choose_order(Length_impulsive_2, var_path="path", var_
    conv="total_conversions", var_null="total_null")
ROC_logit_length_impulsive <- roc(test_logit_length_impulsive$bin_conv,
    predicted_log_length_impulsive)

fpr1_length_impulsive_2 = res_length_impulsive_2[["roc"]][["1"]][["fpr"]]
trp1_length_impulsive_2 = res_length_impulsive_2[["roc"]][["1"]][["tpr"]]
fpr2_length_impulsive_2 = res_length_impulsive_2[["roc"]][["2"]][["fpr"]]
trp2_length_impulsive_2 = res_length_impulsive_2[["roc"]][["2"]][["tpr"]]
fpr3_length_impulsive_2 = res_length_impulsive_2[["roc"]][["3"]][["fpr"]]
trp3_length_impulsive_2 = res_length_impulsive_2[["roc"]][["3"]][["tpr"]]
fpr4_length_impulsive_2 = res_length_impulsive_2[["roc"]][["4"]][["fpr"]]
trp4_length_impulsive_2 = res_length_impulsive_2[["roc"]][["4"]][["tpr"]]
trp_logit_length_impulsive = ROC_logit_length_impulsive[["sensitivities"]]
spec_logit_length_impulsive <- ROC_logit_length_impulsive[["specificities"]]
frp_logit_length_impulsive = c(1) - spec_logit_length_impulsive

plot(fpr1_length_impulsive_2,trp1_length_impulsive_2,type="l",xlab="False
    Positive Rate",ylab="True Positive Rate",main="ROC")
lines(fpr1_length_impulsive_2,trp1_length_impulsive_2,col="red")
```

```
lines(fpr2_length_impulsive_2,trp2_length_impulsive_2,col="blue")
lines(fpr3_length_impulsive_2,trp3_length_impulsive_2,col="green")
lines(fpr4_length_impulsive_2,trp4_length_impulsive_2,col="purple")
lines(frp_logit_length_impulsive, trp_logit_length_impulsive, col="pink")

legend("right", legend=c("First order","Second order", "Third order","Fourth order", "Logit"),
       col=c("red","blue", "green","purple", "pink"),lty=1)

# Balanced
res_length_balanced_2=choose_order(Length_balanced_2, var_path="path", var_conv="total_conversions", var_null="total_null")
ROC_logit_length_balanced <- roc(test_logit_length_balanced$bin_conv, predicted_log_length_balanced)

fpr1_length_balanced_2 = res_length_balanced_2[["roc"]][["1"]][["fpr"]]
trp1_length_balanced_2 = res_length_balanced_2[["roc"]][["1"]][["tpr"]]
fpr2_length_balanced_2 = res_length_balanced_2[["roc"]][["2"]][["fpr"]]
trp2_length_balanced_2 = res_length_balanced_2[["roc"]][["2"]][["tpr"]]
fpr3_length_balanced_2 = res_length_balanced_2[["roc"]][["3"]][["fpr"]]
trp3_length_balanced_2 = res_length_balanced_2[["roc"]][["3"]][["tpr"]]
fpr4_length_balanced_2 = res_length_balanced_2[["roc"]][["4"]][["fpr"]]
trp4_length_balanced_2 = res_length_balanced_2[["roc"]][["4"]][["tpr"]]
trp_logit_length_balanced = ROC_logit_length_balanced[["sensitivities"]]
spec_logit_length_balanced <- ROC_logit_length_balanced[["specificities"]]
frp_logit_length_balanced = c(1) - spec_logit_length_balanced

plot(fpr1_length_balanced_2,trp1_length_balanced_2,type="l",xlab="False Positive Rate",ylab="True Positive Rate",main="ROC")
lines(fpr1_length_balanced_2,trp1_length_balanced_2,col="red")
lines(fpr2_length_balanced_2,trp2_length_balanced_2,col="blue")
lines(fpr3_length_balanced_2,trp3_length_balanced_2,col="green")
lines(fpr4_length_balanced_2,trp4_length_balanced_2,col="purple")
lines(frp_logit_length_balanced, trp_logit_length_balanced, col="pink")

legend("right", legend=c("First order","Second order", "Third order","Fourth order", "Logit"),
       col=c("red","blue", "green","purple", "pink"),lty=1)

# Considered
res_length_considered_2=choose_order(Length_considered_2, var_path="path", var_conv="total_conversions", var_null="total_null")
```

```
643 ROC_logit_length_considered <- roc(test_logit_length_considered$bin_conv,
        predicted_log_length_considered)
644
645 fpr1_length_considered_2 = res_length_considered_2[["roc"]][["1"]][["fpr"]]
646 trp1_length_considered_2 = res_length_considered_2[["roc"]][["1"]][["tpr"]]
647 fpr2_length_considered_2 = res_length_considered_2[["roc"]][["2"]][["fpr"]]
648 trp2_length_considered_2 = res_length_considered_2[["roc"]][["2"]][["tpr"]]
649 fpr3_length_considered_2 = res_length_considered_2[["roc"]][["3"]][["fpr"]]
650 trp3_length_considered_2 = res_length_considered_2[["roc"]][["3"]][["tpr"]]
651 fpr4_length_considered_2 = res_length_considered_2[["roc"]][["4"]][["fpr"]]
652 trp4_length_considered_2 = res_length_considered_2[["roc"]][["4"]][["tpr"]]
653 trp_logit_length_considered = ROC_logit_length_considered[["sensitivities"]]
654 spec_logit_length_considered <- ROC_logit_length_considered[["specificities"]]
655 frp_logit_length_considered = c(1) - spec_logit_length_considered
656
657 plot(fpr1_length_considered_2,trp1_length_considered_2,type="l",xlab="False
        Positive Rate",ylab="True Positive Rate",main="ROC")
658 lines(fpr1_length_considered_2,trp1_length_considered_2,col="red")
659 lines(fpr2_length_considered_2,trp2_length_considered_2,col="blue")
660 lines(fpr3_length_considered_2,trp3_length_considered_2,col="green")
661 lines(frp_logit_length_considered, trp_logit_length_considered, col="pink")
662
663 legend("right", legend=c("First order","Second order", "Third order","Fourth
        order", "Logit"),
664            col=c("red","blue", "green","purple", "pink"),lty=1)
665
666 # AUC
667 # Markov models -> values found in[U+FFFD]reslength_impulsive[U+FFFD][U+FFFD]res
        length_balanced[U+FFFD][U+FFFD]reslength_considered[U+FFFD]
668
669 # Logit - Impulsive
670 Length_impulsive_2$conv_and_null <- Length_impulsive_2$total_conversions +
        Length_impulsive_2$total_null
671 Length_impulsive_2logit <- Length_impulsive_2 %>% slice(rep(1:n(), Length_
        impulsive_2$conv_and_null))
672 Length_impulsive_2logit <- Length_impulsive_2logit %>%
673 group_by(path) %>%
674 mutate(row_number = 1:n())
675 Length_impulsive_2logit$bin_conv <- ifelse(Length_impulsive_2logit$row_number
        <= Length_impulsive_2logit$total_conversions, 1, 0)
676 Length_impulsive_2logit$bin_null <- ifelse(Length_impulsive_2logit$bin_conv ==
        0, 1, 0)
```

```r
Channels_occurences_length_impulsive_logit <- Channels_occurences_length_impulsive %>% slice(rep(1:n(), Length_impulsive_2$conv_and_null))

logit_nonagg_length_impulsive <- glm(formula = cbind(Length_impulsive_2logit$bin_conv, Length_impulsive_2logit$bin_null) ~ (Channels_occurences_length_impulsive_logit$eta + Channels_occurences_length_impulsive_logit$iota + Channels_occurences_length_impulsive_logit$alpha + Channels_occurences_length_impulsive_logit$beta + Channels_occurences_length_impulsive_logit$theta + Channels_occurences_length_impulsive_logit$lambda + Channels_occurences_length_impulsive_logit$kappa + Channels_occurences_length_impulsive_logit$zeta + Channels_occurences_length_impulsive_logit$epsilon + Channels_occurences_length_impulsive_logit$gamma), family = binomial("logit"), data = Length_impulsive_2logit)

# Use 80% of dataset as training set and remaining 20% as testing set
sample_logit_length_impulsive <- sample(c(TRUE, FALSE), nrow(Length_impulsive_2logit), replace=TRUE, prob=c(0.8,0.2))
train_logit_length_impulsive <- Length_impulsive_2logit[sample_logit_length_impulsive, ]
test_logit_length_impulsive <- Length_impulsive_2logit[!sample_logit_length_impulsive, ]
Channels_occurences_nonagg_length_impulsive <- Channels_occurences_length_impulsive_logit[!sample_logit_length_impulsive, ]

# Fit logistic regression model (on train)
model_logit_nonagg2_length_impulsive <-  glm(formula = cbind(test_logit_length_impulsive$bin_conv, test_logit_length_impulsive$bin_null) ~ (Channels_occurences_nonagg_length_impulsive$eta + Channels_occurences_nonagg_length_impulsive$iota + Channels_occurences_nonagg_length_impulsive$alpha + Channels_occurences_nonagg_length_impulsive$beta + Channels_occurences_nonagg_length_impulsive$theta + Channels_occurences_nonagg_length_impulsive$lambda + Channels_occurences_nonagg_length_impulsive$kappa + Channels_occurences_nonagg_length_impulsive$zeta + Channels_occurences_nonagg_length_impulsive$epsilon + Channels_occurences_nonagg_length_impulsive$gamma), family = binomial("logit"), data = train_logit_length_impulsive)

# Calculate probability of default for each individual in test dataset
predicted_log_length_impulsive <- predict(model_logit_nonagg2_length_impulsive, test_logit_length_impulsive, type="response")
auc(test_logit_length_impulsive$bin_conv, predicted_log_length_impulsive)

# Logit - Balanced
```

```
695 Length_balanced_2$conv_and_null <- Length_balanced_2$total_conversions +
    Length_balanced_2$total_null
696 Length_balanced_2logit <- Length_balanced_2 %>% slice(rep(1:n(), Length_
    balanced_2$conv_and_null))
697 Length_balanced_2logit <- Length_balanced_2logit %>%
698 group_by(path) %>%
699 mutate(row_number = 1:n())
700 Length_balanced_2logit$bin_conv <- ifelse(Length_balanced_2logit$row_number <=
     Length_balanced_2logit$total_conversions, 1, 0)
701 Length_balanced_2logit$bin_null <- ifelse(Length_balanced_2logit$bin_conv ==
    0, 1, 0)
702 Channels_occurences_length_balanced_logit <- Channels_occurences_length_
    balanced %>% slice(rep(1:n(), Length_balanced_2$conv_and_null))
703
704 logit_nonagg_length_balanced <- glm(formula = cbind(Length_balanced_2logit$bin
    _conv, Length_balanced_2logit$bin_null) ~ (Channels_occurences_length_
    balanced_logit$eta + Channels_occurences_length_balanced_logit$iota +
    Channels_occurences_length_balanced_logit$alpha + Channels_occurences_
    length_balanced_logit$beta + Channels_occurences_length_balanced_logit$
    theta + Channels_occurences_length_balanced_logit$lambda + Channels_
    occurences_length_balanced_logit$kappa + Channels_occurences_length_
    balanced_logit$zeta + Channels_occurences_length_balanced_logit$epsilon +
    Channels_occurences_length_balanced_logit$gamma), family = binomial("logit
    "), data = Length_balanced_2logit)
705
706 # Use 80% of dataset as training set and remaining 20% as testing set
707 sample_logit_length_balanced <- sample(c(TRUE, FALSE), nrow(Length_balanced_2
    logit), replace=TRUE, prob=c(0.8,0.2))
708 train_logit_length_balanced <- Length_balanced_2logit[sample_logit_length_
    balanced, ]
709 test_logit_length_balanced <- Length_balanced_2logit[!sample_logit_length_
    balanced, ]
710 Channels_occurences_nonagg_length_balanced <- Channels_occurences_length_
    balanced_logit[!sample_logit_length_balanced, ]
711
712 # Fit logistic regression model (on train)
713 model_logit_nonagg2_length_balanced <-  glm(formula = cbind(test_logit_length_
    balanced$bin_conv, test_logit_length_balanced$bin_null) ~ (Channels_
    occurences_nonagg_length_balanced$eta + Channels_occurences_nonagg_length_
    balanced$iota + Channels_occurences_nonagg_length_balanced$alpha +
    Channels_occurences_nonagg_length_balanced$beta + Channels_occurences_
    nonagg_length_balanced$theta + Channels_occurences_nonagg_length_balanced$
    lambda + Channels_occurences_nonagg_length_balanced$kappa + Channels_
```

```
    occurences_nonagg_length_balanced$zeta + Channels_occurences_nonagg_length
    _balanced$epsilon + Channels_occurences_nonagg_length_balanced$gamma),
    family = binomial("logit"), data = train_logit_length_balanced)

# Calculate probability of default for each individual in test dataset
predicted_log_length_balanced <- predict(model_logit_nonagg2_length_balanced,
    test_logit_length_balanced, type="response")
auc(test_logit_length_balanced$bin_conv, predicted_log_length_balanced)

# Logit - Considered
Length_considered_2$conv_and_null <- Length_considered_2$total_conversions +
    Length_considered_2$total_null
Length_considered_2logit <- Length_considered_2 %>% slice(rep(1:n(), Length_
    considered_2$conv_and_null))
Length_considered_2logit <- Length_considered_2logit %>%
group_by(path) %>%
mutate(row_number = 1:n())
Length_considered_2logit$bin_conv <- ifelse(Length_considered_2logit$row_
    number <= Length_considered_2logit$total_conversions, 1, 0)
Length_considered_2logit$bin_null <- ifelse(Length_considered_2logit$bin_conv
    == 0, 1, 0)
Channels_occurences_length_considered_logit <- Channels_occurences_length_
    considered %>% slice(rep(1:n(), Length_considered_2$conv_and_null))

logit_nonagg_length_considered <- glm(formula = cbind(Length_considered_2logit
    $bin_conv, Length_considered_2logit$bin_null) ~ (Channels_occurences_
    length_considered_logit$eta + Channels_occurences_length_considered_logit$
    iota + Channels_occurences_length_considered_logit$alpha + Channels_
    occurences_length_considered_logit$beta + Channels_occurences_length_
    considered_logit$theta + Channels_occurences_length_considered_logit$
    lambda + Channels_occurences_length_considered_logit$kappa + Channels_
    occurences_length_considered_logit$zeta + Channels_occurences_length_
    considered_logit$epsilon + Channels_occurences_length_considered_logit$
    gamma), family = binomial("logit"), data = Length_considered_2logit)

# Use 80% of dataset as training set and remaining 20% as testing set
sample_logit_length_considered <- sample(c(TRUE, FALSE), nrow(Length_
    considered_2logit), replace=TRUE, prob=c(0.8,0.2))
train_logit_length_considered <- Length_considered_2logit[sample_logit_length_
    considered, ]
test_logit_length_considered <- Length_considered_2logit[!sample_logit_length_
    considered, ]
```

```
Channels_occurences_nonagg_length_considered <- Channels_occurences_length_considered_logit[!sample_logit_length_considered, ]

# Fit logistic regression model (on train)
model_logit_nonagg2_length_considered <-  glm(formula = cbind(test_logit_length_considered$bin_conv, test_logit_length_considered$bin_null) ~ (Channels_occurences_nonagg_length_considered$eta + Channels_occurences_nonagg_length_considered$iota + Channels_occurences_nonagg_length_considered$alpha + Channels_occurences_nonagg_length_considered$beta + Channels_occurences_nonagg_length_considered$theta + Channels_occurences_nonagg_length_considered$lambda + Channels_occurences_nonagg_length_considered$kappa + Channels_occurences_nonagg_length_considered$zeta + Channels_occurences_nonagg_length_considered$epsilon + Channels_occurences_nonagg_length_considered$gamma), family = binomial("logit"), data = train_logit_length_considered)

# Calculate probability of default for each individual in test dataset
predicted_log_length_considered <- predict(model_logit_nonagg2_length_considered, test_logit_length_considered, type="response")
auc(test_logit_length_considered$bin_conv, predicted_log_length_considered)

# TOP DECILE LIFT
# Impulsive
chisq_first_conversion_length_impulsive_2 <- chisq.test(Heuristics_length_impulsive_2$first_touch_conversions/sum(Heuristics_length_impulsive_2$first_touch_conversions)*100)
chisq_last_conversion_length_impulsive_2 <- chisq.test(Heuristics_length_impulsive_2$last_touch_conversions/sum(Heuristics_length_impulsive_2$last_touch_conversions)*100)
chisq_linear_conversion_length_impulsive_2 <- chisq.test(Heuristics_length_impulsive_2$linear_touch_conversions/sum(Heuristics_length_impulsive_2$linear_touch_conversions)*100)
chisq_markov1_conversion_length_impulsive_2 <-
chisq.test(Markov1_length_impulsive_2$total_conversion/sum(Markov1_length_impulsive_2$total_conversion)*100)
chisq_markov2_conversion_length_impulsive_2 <-
chisq.test(Markov2_length_impulsive_2$total_conversion/sum(Markov2_length_impulsive_2$total_conversion)*100)
chisq_markov3_conversion_length_impulsive_2 <-
chisq.test(Markov3_length_impulsive_2$total_conversion/sum(Markov3_length_impulsive_2$total_conversion)*100)
chisq_markov4_conversion_length_impulsive_2 <- chisq.test(Markov4_length_impulsive_2$total_conversion/sum(Markov4_length_impulsive_2$total_
```

```
    conversion)*100)
756
757 Expected_first_conversion_length_impulsive_2 = chisq_first_conversion_length_
    impulsive_2[["expected"]]
758 Observed_first_conversion_length_impulsive_2 = chisq_first_conversion_length_
    impulsive_2[["observed"]]
759 TDL_first_conversion_length_impulsive_2 <- TopDecileLift(Expected_first_
    conversion_length_impulsive_2, Observed_first_conversion_length_impulsive_
    2)
760
761 Expected_last_conversion_length_impulsive_2 = chisq_last_conversion_length_
    impulsive_2[["expected"]]
762 Observed_last_conversion_length_impulsive_2 = chisq_last_conversion_length_
    impulsive_2[["observed"]]
763 TDL_last_conversion_length_impulsive_2 <- TopDecileLift(Expected_last_
    conversion_length_impulsive_2, Observed_last_conversion_length_impulsive_
    2)
764
765 Expected_linear_conversion_length_impulsive_2 = chisq_linear_conversion_length
    _impulsive_2[["expected"]]
766 Observed_linear_conversion_length_impulsive_2 = chisq_linear_conversion_length
    _impulsive_2[["observed"]]
767 TDL_linear_conversion_length_impulsive_2 <- TopDecileLift(Expected_linear_
    conversion_length_impulsive_2, Observed_linear_conversion_length_impulsive
    _2)
768
769 Expected_logit_length_impulsive = sum(logit2_length_impulsive$effects)/3890
770 Observed_logit_length_impulsive = logit2_length_impulsive$effects
771 TDL_logit_length_impulsive <- TopDecileLift(Expected_logit_length_impulsive,
    Observed_logit_length_impulsive)
772
773 Expected_markov1_conversion_length_impulsive_2 = chisq_markov1_conversion_
    length_impulsive_2[["expected"]]
774 Observed_markov1_conversion_length_impulsive_2 = chisq_markov1_conversion_
    length_impulsive_2[["observed"]]
775 TDL_markov1_conversion_length_impulsive_2 <- TopDecileLift(Expected_markov1_
    conversion_length_impulsive_2, Observed_markov1_conversion_length_
    impulsive_2)
776 Expected_markov2_conversion_length_impulsive_2 = chisq_markov2_conversion_
    length_impulsive_2[["expected"]]
777 Observed_markov2_conversion_length_impulsive_2 = chisq_markov2_conversion_
    length_impulsive_2[["observed"]]
```

```
TDL_markov2_conversion_length_impulsive_2 <- TopDecileLift(Expected_markov2_conversion_length_impulsive_2, Observed_markov2_conversion_length_impulsive_2)

Expected_markov3_conversion_length_impulsive_2 = chisq_markov3_conversion_length_impulsive_2[["expected"]]
Observed_markov3_conversion_length_impulsive_2 = chisq_markov3_conversion_length_impulsive_2[["observed"]]
TDL_markov3_conversion_length_impulsive_2 <- TopDecileLift(Expected_markov3_conversion_length_impulsive_2, Observed_markov3_conversion_length_impulsive_2)

Expected_markov4_conversion_length_impulsive_2 = chisq_markov4_conversion_length_impulsive_2[["expected"]]
Observed_markov4_conversion_length_impulsive_2 = chisq_markov4_conversion_length_impulsive_2[["observed"]]
TDL_markov4_conversion_length_impulsive_2 <- TopDecileLift(Expected_markov4_conversion_length_impulsive_2, Observed_markov4_conversion_length_impulsive_2)

# Balanced
chisq_first_conversion_length_balanced_2 <- chisq.test(Heuristics_length_balanced_2$first_touch_conversions/sum(Heuristics_length_balanced_2$first_touch_conversions)*100)
chisq_last_conversion_length_balanced_2 <- chisq.test(Heuristics_length_balanced_2$last_touch_conversions/sum(Heuristics_length_balanced_2$last_touch_conversions)*100)
chisq_linear_conversion_length_balanced_2 <- chisq.test(Heuristics_length_balanced_2$linear_touch_conversions/sum(Heuristics_length_balanced_2$linear_touch_conversions)*100)
chisq_markov1_conversion_length_balanced_2 <- chisq.test(Markov1_length_balanced_2$total_conversion/sum(Markov1_length_balanced_2$total_conversion)*100)
chisq_markov2_conversion_length_balanced_2 <- chisq.test(Markov2_length_balanced_2$total_conversion/sum(Markov2_length_balanced_2$total_conversion)*100)
chisq_markov3_conversion_length_balanced_2 <- chisq.test(Markov3_length_balanced_2$total_conversion/sum(Markov3_length_balanced_2$total_conversion)*100)
chisq_markov4_conversion_length_balanced_2 <- chisq.test(Markov4_length_balanced_2$total_conversion/sum(Markov4_length_balanced_2$total_conversion)*100)

```

```
797 Expected_first_conversion_length_balanced_2 = chisq_first_conversion_length_
        balanced_2[["expected"]]
798 Observed_first_conversion_length_balanced_2 = chisq_first_conversion_length_
        balanced_2[["observed"]]
799 TDL_first_conversion_length_balanced_2 <- TopDecileLift(Expected_first_
        conversion_length_balanced_2, Observed_first_conversion_length_balanced_2)
800
801 Expected_last_conversion_length_balanced_2 = chisq_last_conversion_length_
        balanced_2[["expected"]]
802 Observed_last_conversion_length_balanced_2 = chisq_last_conversion_length_
        balanced_2[["observed"]]
803 TDL_last_conversion_length_balanced_2 <- TopDecileLift(Expected_last_
        conversion_length_balanced_2, Observed_last_conversion_length_balanced_2)
804
805 Expected_linear_conversion_length_balanced_2 = chisq_linear_conversion_length_
        balanced_2[["expected"]]
806 Observed_linear_conversion_length_balanced_2 = chisq_linear_conversion_length_
        balanced_2[["observed"]]
807 TDL_linear_conversion_length_balanced_2 <- TopDecileLift(Expected_linear_
        conversion_length_balanced_2, Observed_linear_conversion_length_balanced_
        2)
808
809 Expected_logit_length_balanced = sum(logit2_length_balanced$effects)/3112
810 Observed_logit_length_balanced = logit2_length_balanced$effects
811 TDL_logit_length_balanced <- TopDecileLift(Expected_logit_length_balanced,
        Observed_logit_length_balanced)
812
813 Expected_markov1_conversion_length_balanced_2 = chisq_markov1_conversion_
        length_balanced_2[["expected"]]
814 Observed_markov1_conversion_length_balanced_2 = chisq_markov1_conversion_
        length_balanced_2[["observed"]]
815 TDL_markov1_conversion_length_balanced_2 <- TopDecileLift(Expected_markov1_
        conversion_length_balanced_2, Observed_markov1_conversion_length_balanced_
        2)
816
817 Expected_markov2_conversion_length_balanced_2 = chisq_markov2_conversion_
        length_balanced_2[["expected"]]
818 Observed_markov2_conversion_length_balanced_2 = chisq_markov2_conversion_
        length_balanced_2[["observed"]]
819 TDL_markov2_conversion_length_balanced_2 <- TopDecileLift(Expected_markov2_
        conversion_length_balanced_2, Observed_markov2_conversion_length_balanced_
        2)
820
```

```
821 Expected_markov3_conversion_length_balanced_2 = chisq_markov3_conversion_
    length_balanced_2[["expected"]]
822 Observed_markov3_conversion_length_balanced_2 = chisq_markov3_conversion_
    length_balanced_2[["observed"]]
823 TDL_markov3_conversion_length_balanced_2 <- TopDecileLift(Expected_markov3_
    conversion_length_balanced_2, Observed_markov3_conversion_length_balanced_
    2)
824
825 Expected_markov4_conversion_length_balanced_2 = chisq_markov4_conversion_
    length_balanced_2[["expected"]]
826 Observed_markov4_conversion_length_balanced_2 = chisq_markov4_conversion_
    length_balanced_2[["observed"]]
827 TDL_markov4_conversion_length_balanced_2 <- TopDecileLift(Expected_markov4_
    conversion_length_balanced_2, Observed_markov4_conversion_length_balanced_
    2)
828
829 # Considered
830 chisq_first_conversion_length_considered_2 <- chisq.test(Heuristics_length_
    considered_2$first_touch_conversions/sum(Heuristics_length_considered_2$
    first_touch_conversions)*100)
831 chisq_last_conversion_length_considered_2 <- chisq.test(Heuristics_length_
    considered_2$last_touch_conversions/sum(Heuristics_length_considered_2$
    last_touch_conversions)*100)
832 chisq_linear_conversion_length_considered_2 <- chisq.test(Heuristics_length_
    considered_2$linear_touch_conversions/sum(Heuristics_length_considered_2$
    linear_touch_conversions)*100)
833 chisq_markov1_conversion_length_considered_2 <- chisq.test(Markov1_length_
    considered_2$total_conversion/sum(Markov1_length_considered_2$total_
    conversion)*100)
834 chisq_markov2_conversion_length_considered_2 <- chisq.test(Markov2_length_
    considered_2$total_conversion/sum(Markov2_length_considered_2$total_
    conversion)*100)
835 chisq_markov3_conversion_length_considered_2 <- chisq.test(Markov3_length_
    considered_2$total_conversion/sum(Markov3_length_considered_2$total_
    conversion)*100)
836 chisq_markov4_conversion_length_considered_2 <- chisq.test(Markov4_length_
    considered_2$total_conversion/sum(Markov4_length_considered_2$total_
    conversion)*100)
837
838 Expected_first_conversion_length_considered_2 = chisq_first_conversion_length_
    considered_2[["expected"]]
839 Observed_first_conversion_length_considered_2 = chisq_first_conversion_length_
    considered_2[["observed"]]
```

```
TDL_first_conversion_length_considered_2 <- TopDecileLift(Expected_first_conversion_length_considered_2, Observed_first_conversion_length_considered_2)

Expected_last_conversion_length_considered_2 = chisq_last_conversion_length_considered_2[["expected"]]
Observed_last_conversion_length_considered_2 = chisq_last_conversion_length_considered_2[["observed"]]
TDL_last_conversion_length_considered_2 <- TopDecileLift(Expected_last_conversion_length_considered_2, Observed_last_conversion_length_considered_2)

Expected_linear_conversion_length_considered_2 = chisq_linear_conversion_length_considered_2[["expected"]]
Observed_linear_conversion_length_considered_2 = chisq_linear_conversion_length_considered_2[["observed"]]
TDL_linear_conversion_length_considered_2 <- TopDecileLift(Expected_linear_conversion_length_considered_2, Observed_linear_conversion_length_considered_2)

Expected_logit_length_considered = sum(logit2_length_considered$effects)/2983
Observed_logit_length_considered = logit2_length_considered$effects
TDL_logit_length_considered <- TopDecileLift(Expected_logit_length_considered, Observed_logit_length_considered)

Expected_markov1_conversion_length_considered_2 = chisq_markov1_conversion_length_considered_2[["expected"]]
Observed_markov1_conversion_length_considered_2 = chisq_markov1_conversion_length_considered_2[["observed"]]
TDL_markov1_conversion_length_considered_2 <- TopDecileLift(Expected_markov1_conversion_length_considered_2, Observed_markov1_conversion_length_considered_2)

Expected_markov2_conversion_length_considered_2 = chisq_markov2_conversion_length_considered_2[["expected"]]
Observed_markov2_conversion_length_considered_2 = chisq_markov2_conversion_length_considered_2[["observed"]]
TDL_markov2_conversion_length_considered_2 <- TopDecileLift(Expected_markov2_conversion_length_considered_2, Observed_markov2_conversion_length_considered_2)

Expected_markov3_conversion_length_considered_2 = chisq_markov3_conversion_length_considered_2[["expected"]]
```

```
Observed_markov3_conversion_length_considered_2 = chisq_markov3_conversion_length_considered_2[["observed"]]
TDL_markov3_conversion_length_considered_2 <- TopDecileLift(Expected_markov3_conversion_length_considered_2, Observed_markov3_conversion_length_considered_2)

Expected_markov4_conversion_length_considered_2 = chisq_markov4_conversion_length_considered_2[["expected"]]
Observed_markov4_conversion_length_considered_2 = chisq_markov4_conversion_length_considered_2[["observed"]]
TDL_markov4_conversion_length_considered_2 <- TopDecileLift(Expected_markov4_conversion_length_considered_2, Observed_markov4_conversion_length_considered_2)

# REMOVAL EFFECT + TRANSITION MATRIX
# Impulsive
m1_length_impulsive_2 = markov_model(Length_impulsive_2, "path", "total_conversions", var_value="total_conversion_value", var_null="total_null", out_more=TRUE, order = 1)
m2_length_impulsive_2 = markov_model(Length_impulsive_2, "path", "total_conversions", var_value="total_conversion_value", var_null="total_null", out_more=TRUE, order = 2)
m3_length_impulsive_2 = markov_model(Length_impulsive_2, "path", "total_conversions", var_value="total_conversion_value", var_null="total_null", out_more=TRUE, order = 3)
m4_length_impulsive_2 = markov_model(Length_impulsive_2, "path", "total_conversions", var_value="total_conversion_value", var_null="total_null", out_more=TRUE, order = 4)

# Balanced
m1_length_balanced_2 = markov_model(Length_balanced_2, "path", "total_conversions", var_value="total_conversion_value", var_null="total_null", out_more=TRUE, order = 1)
m2_length_balanced_2 = markov_model(Length_balanced_2, "path", "total_conversions", var_value="total_conversion_value", var_null="total_null", out_more=TRUE, order = 2)
m3_length_balanced_2 = markov_model(Length_balanced_2, "path", "total_conversions", var_value="total_conversion_value", var_null="total_null", out_more=TRUE, order = 3)
m4_length_balanced_2 = markov_model(Length_balanced_2, "path", "total_conversions", var_value="total_conversion_value", var_null="total_null", out_more=TRUE, order = 4)

```

```
# Considered
m1_length_considered_2 = markov_model(Length_considered_2, "path", "total_
    conversions", var_value="total_conversion_value", var_null="total_null",
    out_more=TRUE, order = 1)
m2_length_considered_2 = markov_model(Length_considered_2, "path", "total_
    conversions", var_value="total_conversion_value", var_null="total_null",
    out_more=TRUE, order = 2)
m3_length_considered_2 = markov_model(Length_considered_2, "path", "total_
    conversions", var_value="total_conversion_value", var_null="total_null",
    out_more=TRUE, order = 3)
m4_length_considered_2 = markov_model(Length_considered_2, "path", "total_
    conversions", var_value="total_conversion_value", var_null="total_null",
    out_more=TRUE, order = 4)

# STANDARD DEVIATION REMOVAL EFFECT
# Impulsive
removal_effects_m1_length_impulsive_2 <- m1_length_impulsive_2[["removal_
    effects"]][["removal_effects_conversion"]]
sum(removal_effects_m1_length_impulsive_2)/10
sd(removal_effects_m1_length_impulsive_2)
sd(removal_effects_m1_length_impulsive_2)/(sum(removal_effects_m1_length_
    impulsive_2)/10)

removal_effects_m2_length_impulsive_2 <- m2_length_impulsive_2[["removal_
    effects"]][["removal_effects_conversion"]]
sum(removal_effects_m2_length_impulsive_2)/10
sd(removal_effects_m2_length_impulsive_2)
sd(removal_effects_m2_length_impulsive_2)/(sum(removal_effects_m2_length_
    impulsive_2)/10)

removal_effects_m3_length_impulsive_2 <- m3_length_impulsive_2[["removal_
    effects"]][["removal_effects_conversion"]]
sum(removal_effects_m3_length_impulsive_2)/10
sd(removal_effects_m3_length_impulsive_2)
sd(removal_effects_m3_length_impulsive_2)/(sum(removal_effects_m3_length_
    impulsive_2)/10)

removal_effects_m4_length_impulsive_2 <- m4_length_impulsive_2[["removal_
    effects"]][["removal_effects_conversion"]]
sum(removal_effects_m4_length_impulsive_2)/10
sd(removal_effects_m4_length_impulsive_2)
sd(removal_effects_m4_length_impulsive_2)/(sum(removal_effects_m4_length_
    impulsive_2)/10)
```

```
910
911 # Balanced
912 removal_effects_m1_length_balanced_2 <- m1_length_balanced_2[["removal_effects
    "]][["removal_effects_conversion"]]
913 sum(removal_effects_m1_length_balanced_2)/10
914 sd(removal_effects_m1_length_balanced_2)
915 sd(removal_effects_m1_length_balanced_2)/(sum(removal_effects_m1_length_
    balanced_2)/10)
916
917 removal_effects_m2_length_balanced_2 <- m2_length_balanced_2[["removal_effects
    "]][["removal_effects_conversion"]]
918 sum(removal_effects_m2_length_balanced_2)/10
919 sd(removal_effects_m2_length_balanced_2)
920 sd(removal_effects_m2_length_balanced_2)/(sum(removal_effects_m2_length_
    balanced_2)/10)
921
922 removal_effects_m3_length_balanced_2 <- m3_length_balanced_2[["removal_effects
    "]][["removal_effects_conversion"]]
923 sum(removal_effects_m3_length_balanced_2)/10
924 sd(removal_effects_m3_length_balanced_2)
925 sd(removal_effects_m3_length_balanced_2)/(sum(removal_effects_m3_length_
    balanced_2)/10)
926
927 removal_effects_m4_length_balanced_2 <- m4_length_balanced_2[["removal_effects
    "]][["removal_effects_conversion"]]
928 sum(removal_effects_m4_length_balanced_2)/10
929 sd(removal_effects_m4_length_balanced_2)
930 sd(removal_effects_m4_length_balanced_2)/(sum(removal_effects_m4_length_
    balanced_2)/10)
931
932 # Considered
933 removal_effects_m1_length_considered_2 <- m1_length_considered_2[["removal_
    effects"]][["removal_effects_conversion"]]
934 sum(removal_effects_m1_length_considered_2)/10
935 sd(removal_effects_m1_length_considered_2)
936 sd(removal_effects_m1_length_considered_2)/(sum(removal_effects_m1_length_
    considered_2)/10)
937
938 removal_effects_m2_length_considered_2 <- m2_length_considered_2[["removal_
    effects"]][["removal_effects_conversion"]]
939 sum(removal_effects_m2_length_considered_2)/10
940 sd(removal_effects_m2_length_considered_2)
```

```
sd(removal_effects_m2_length_considered_2)/(sum(removal_effects_m2_length_
    considered_2)/10)

removal_effects_m3_length_considered_2 <- m3_length_considered_2[["removal_
    effects"]][["removal_effects_conversion"]]
sum(removal_effects_m3_length_considered_2)/10
sd(removal_effects_m3_length_considered_2)
sd(removal_effects_m3_length_considered_2)/(sum(removal_effects_m3_length_
    considered_2)/10)

removal_effects_m4_length_considered_2 <- m4_length_considered_2[["removal_
    effects"]][["removal_effects_conversion"]]
sum(removal_effects_m4_length_considered_2)/10
sd(removal_effects_m4_length_considered_2)
sd(removal_effects_m4_length_considered_2)/(sum(removal_effects_m4_length_
    considered_2)/10)

# TYPES OF JOURNEYS - SPLIT MADE BY NUMBER CHANNELS
# Imported excel file where the number of different channels is counted (excel
    file is called [U+FFFD]Numberchannels[U+FFFD] the column in the excel file is
    called [U+FFFD]Number_channels[U+FFFD])
# Remove journeys with delta and mi
# Delta present in observations: 659, 1702, 1889, 2392, 3020, 5045, 5889,
    5933, 6404, 6504, 6977, 7232, 7496
# Mi present in observations: 2094, 3033
Numberchannels2 <- Numberchannels[-c(659, 1702, 1889, 2392, 3020, 5045, 5889,
    5933, 6404, 6504, 6977, 7232, 7496, 2094, 3033), ]

Data2$numberofchannels <- c(Numberchannels2$Number_channels)
Channels_impulsive_2 <- subset(Data2, numberofchannels < 2)
Channels_balanced_2 <- subset(Data2, numberofchannels > 1 & numberofchannels <
    4)
Channels_considered_2 <- subset(Data2, numberofchannels > 3 & numberofchannels
    < 9)

# Heuristics
Heuristics_channels_impulsive_2 <- heuristic_models(Channels_impulsive_2, var_
    path = 'path', var_conv = 'total_conversions', var_value='total_conversion
    _value')
Heuristics_channels_balanced_2 <- heuristic_models(Channels_balanced_2, var_
    path = 'path', var_conv = 'total_conversions', var_value='total_conversion
    _value')
```

```
968 Heuristics_channels_considered_2 <- heuristic_models(Channels_considered_2,
        var_path = 'path', var_conv = 'total_conversions', var_value='total_
        conversion_value')
969
970 Heuristics_channels_impulsive_2$first_touch_conversions/sum(Heuristics_
        channels_impulsive_2$first_touch_conversions)*100
971 Heuristics_channels_impulsive_2$last_touch_conversions/sum(Heuristics_channels
        _impulsive_2$last_touch_conversions)*100
972 Heuristics_channels_impulsive_2$linear_touch_conversions/sum(Heuristics_
        channels_impulsive_2$linear_touch_conversions)*100
973
974 Heuristics_channels_balanced_2$first_touch_conversions/sum(Heuristics_channels
        _balanced_2$first_touch_conversions)*100
975 Heuristics_channels_balanced_2$last_touch_conversions/sum(Heuristics_channels_
        balanced_2$last_touch_conversions)*100
976 Heuristics_channels_balanced_2$linear_touch_conversions/sum(Heuristics_
        channels_balanced_2$linear_touch_conversions)*100
977
978 Heuristics_channels_considered_2$first_touch_conversions/sum(Heuristics_
        channels_considered_2$first_touch_conversions)*100
979 Heuristics_channels_considered_2$last_touch_conversions/sum(Heuristics_
        channels_considered_2$last_touch_conversions)*100
980 Heuristics_channels_considered_2$linear_touch_conversions/sum(Heuristics_
        channels_considered_2$linear_touch_conversions)*100
981
982 # Markov 1
983 Markov1_channels_impulsive_2 <- markov_model(Channels_impulsive_2, var_path =
        "path", var_conv = "total_conversions", var_value="total_conversion_value"
        , var_null="total_null", order = 1)
984 Markov1_channels_balanced_2 <- markov_model(Channels_balanced_2, var_path = "
        path", var_conv = "total_conversions", var_value="total_conversion_value",
         var_null="total_null", order = 1)
985 Markov1_channels_considered_2 <- markov_model(Channels_considered_2, var_path
        = "path", var_conv = "total_conversions", var_value="total_conversion_
        value", var_null="total_null", order = 1)
986
987 # Markov 2
988 Markov2_channels_impulsive_2 <- markov_model(Channels_impulsive_2, var_path =
        "path", var_conv = "total_conversions", var_value="total_conversion_value"
        , var_null="total_null", order = 2)
989 Markov2_channels_balanced_2 <- markov_model(Channels_balanced_2, var_path = "
        path", var_conv = "total_conversions", var_value="total_conversion_value",
         var_null="total_null", order = 2)
```

```
990 Markov2_channels_considered_2 <- markov_model(Channels_considered_2, var_path
    = "path", var_conv = "total_conversions", var_value="total_conversion_
    value", var_null="total_null", order = 2)
991
992 # Markov 3
993 Markov3_channels_impulsive_2 <- markov_model(Channels_impulsive_2, var_path =
    "path", var_conv = "total_conversions", var_value="total_conversion_value"
    , var_null="total_null", order = 3)
994 Markov3_channels_balanced_2 <- markov_model(Channels_balanced_2, var_path = "
    path", var_conv = "total_conversions", var_value="total_conversion_value",
     var_null="total_null", order = 3)
995 Markov3_channels_considered_2 <- markov_model(Channels_considered_2, var_path
    = "path", var_conv = "total_conversions", var_value="total_conversion_
    value", var_null="total_null", order = 3)
996
997 # Markov 4
998 Markov4_channels_impulsive_2 <- markov_model(Channels_impulsive_2, var_path =
    "path", var_conv = "total_conversions", var_value="total_conversion_value"
    , var_null="total_null", order = 4)
999 Markov4_channels_balanced_2 <- markov_model(Channels_balanced_2, var_path = "
    path", var_conv = "total_conversions", var_value="total_conversion_value",
     var_null="total_null", order = 4)
1000 Markov4_channels_considered_2 <- markov_model(Channels_considered_2, var_path
    = "path", var_conv = "total_conversions", var_value="total_conversion_
    value", var_null="total_null", order = 4)
1001
1002 Markov4_channels_impulsive_2$total_conversion/sum(Markov4_channels_impulsive_2
    $total_conversion)*100
1003 Markov4_channels_balanced_2$total_conversion/sum(Markov4_channels_balanced_2$
    total_conversion)*100
1004 Markov4_channels_considered_2$total_conversion/sum(Markov4_channels_considered
    _2$total_conversion)*100
1005
1006 # Logit
1007 # Imported three excel files with the number of occurrences of each channel in
     each observation -> files are called Channel_occurences_channels_
    impulsive, Channel_occurences_channels_balanced, Channel_occurences_
    channels_considered
1008 logit2_channels_impulsive <- glm(formula = cbind(Channels_impulsive_2$total_
    conversions, Channels_impulsive_2$total_null) ~ (Channels_occurences_
    channels_impulsive$eta + Channels_occurences_channels_impulsive$iota +
    Channels_occurences_channels_impulsive$alpha + Channels_occurences_
    channels_impulsive$beta + Channels_occurences_channels_impulsive$theta +
```

```
    Channels_occurences_channels_impulsive$lambda + Channels_occurences_
    channels_impulsive$kappa + Channels_occurences_channels_impulsive$zeta +
    Channels_occurences_channels_impulsive$epsilon + Channels_occurences_
    channels_impulsive$gamma), family = binomial("logit"), data = Channels_
    impulsive_2)
summary(logit2_channels_impulsive)
exp(logit2_channels_impulsive$coefficients)
marg2_channels_impulsive <- logitmfx(formula = cbind(Channels_impulsive_2$
    total_conversions, Channels_impulsive_2$total_null) ~ (Channels_occurences
    _channels_impulsive$eta + Channels_occurences_channels_impulsive$iota +
    Channels_occurences_channels_impulsive$alpha + Channels_occurences_
    channels_impulsive$beta + Channels_occurences_channels_impulsive$theta +
    Channels_occurences_channels_impulsive$lambda + Channels_occurences_
    channels_impulsive$kappa + Channels_occurences_channels_impulsive$epsilon
    + Channels_occurences_channels_impulsive$gamma), data = Channels_balanced_
    2)

logit2_channels_balanced <- glm(formula = cbind(Channels_balanced_2$total_
    conversions, Channels_balanced_2$total_null) ~ (Channels_occurences_
    channels_balanced$eta + Channels_occurences_channels_balanced$iota +
    Channels_occurences_channels_balanced$alpha + Channels_occurences_channels
    _balanced$beta + Channels_occurences_channels_balanced$theta + Channels_
    occurences_channels_balanced$lambda + Channels_occurences_channels_
    balanced$kappa + Channels_occurences_channels_balanced$zeta + Channels_
    occurences_channels_balanced$epsilon + Channels_occurences_channels_
    balanced$gamma), family = binomial("logit"), data = Channels_balanced_2)
summary(logit2_channels_balanced)
exp(logit2_channels_balanced$coefficients)
marg2_channels_balanced <- logitmfx(formula = cbind(Channels_balanced_2$total_
    conversions, Channels_balanced_2$total_null) ~ (Channels_occurences_
    channels_balanced$eta + Channels_occurences_channels_balanced$iota +
    Channels_occurences_channels_balanced$alpha + Channels_occurences_channels
    _balanced$beta + Channels_occurences_channels_balanced$theta + Channels_
    occurences_channels_balanced$lambda + Channels_occurences_channels_
    balanced$kappa + Channels_occurences_channels_balanced$zeta +
Channels_occurences_channels_balanced$epsilon + Channels_occurences_channels_
    balanced$gamma), data = Channels_balanced_2)

logit2_channels_considered <- glm(formula = cbind(Channels_considered_2$total_
    conversions, Channels_considered_2$total_null) ~ (Channels_occurences_
    channels_considered$eta + Channels_occurences_channels_considered$iota +
    Channels_occurences_channels_considered$alpha + Channels_occurences_
    channels_considered$beta + Channels_occurences_channels_considered$theta +
```

```
    Channels_occurences_channels_considered$lambda + Channels_occurences_
    channels_considered$kappa + Channels_occurences_channels_considered$zeta +
     Channels_occurences_channels_considered$epsilon + Channels_occurences_
    channels_considered$gamma), family = binomial("logit"), data = Channels_
    considered_2)
summary(logit2_channels_considered)
exp(logit2_channels_considered$coefficients)
marg2_channels_considered <- logitmfx(formula = cbind(Channels_considered_2$
    total_conversions, Channels_considered_2$total_null) ~ (Channels_
    occurences_channels_considered$eta + Channels_occurences_channels_
    considered$iota + Channels_occurences_channels_considered$alpha + Channels
    _occurences_channels_considered$beta + Channels_occurences_channels_
    considered$theta + Channels_occurences_channels_considered$lambda +
    Channels_occurences_channels_considered$kappa +
Channels_occurences_channels_considered$zeta + Channels_occurences_channels_
    considered$epsilon + Channels_occurences_channels_considered$gamma), data
    = Channels_considered_2)

# ROC
# Impulsive
res_channels_impulsive_2=choose_order(Channels_impulsive_2, var_path="path",
    var_conv="total_conversions", var_null="total_null")
ROC_logit_channels_impulsive <- roc(test_logit_channels_impulsive$bin_conv,
    predicted_log_channels_impulsive)

fpr1_channels_impulsive_2 = res_channels_impulsive_2[["roc"]][["1"]][["fpr"]]
trp1_channels_impulsive_2 = res_channels_impulsive_2[["roc"]][["1"]][["tpr"]]
fpr2_channels_impulsive_2 = res_channels_impulsive_2[["roc"]][["2"]][["fpr"]]
trp2_channels_impulsive_2 = res_channels_impulsive_2[["roc"]][["2"]][["tpr"]]
fpr3_channels_impulsive_2 = res_channels_impulsive_2[["roc"]][["3"]][["fpr"]]
trp3_channels_impulsive_2 = res_channels_impulsive_2[["roc"]][["3"]][["tpr"]]
fpr4_channels_impulsive_2 = res_channels_impulsive_2[["roc"]][["4"]][["fpr"]]
trp4_channels_impulsive_2 = res_channels_impulsive_2[["roc"]][["4"]][["tpr"]]
trp_logit_channels_impulsive = ROC_logit_channels_impulsive[["sensitivities"]]
spec_logit_channels_impulsive <- ROC_logit_channels_impulsive[["specificities"
    ]]
frp_logit_channels_impulsive = c(1) - spec_logit_channels_impulsive

plot(fpr1_channels_impulsive_2,trp1_channels_impulsive_2,type="l",xlab="False
    Positive Rate",ylab="True Positive Rate",main="ROC")
lines(fpr1_channels_impulsive_2,trp1_channels_impulsive_2,col="red")
lines(fpr2_channels_impulsive_2,trp2_channels_impulsive_2,col="blue")
lines(fpr3_channels_impulsive_2,trp3_channels_impulsive_2,col="green")
```

```
lines(fpr4_channels_impulsive_2,trp4_channels_impulsive_2,col="purple")
lines(frp_logit_channels_impulsive, trp_logit_channels_impulsive, col="pink")

legend("right", legend=c("First order","Second order", "Third order","Fourth order", "Logit"),
       col=c("red","blue", "green","purple", "pink"),lty=1)

# Balanced
res_channels_balanced_2=choose_order(Channels_balanced_2, var_path="path", var_conv="total_conversions", var_null="total_null")
ROC_logit_channels_balanced <- roc(test_logit_channels_balanced$bin_conv, predicted_log_channels_balanced)

fpr1_channels_balanced_2 = res_channels_balanced_2[["roc"]][["1"]][["fpr"]]
trp1_channels_balanced_2 = res_channels_balanced_2[["roc"]][["1"]][["tpr"]]
fpr2_channels_balanced_2 = res_channels_balanced_2[["roc"]][["2"]][["fpr"]]
trp2_channels_balanced_2 = res_channels_balanced_2[["roc"]][["2"]][["tpr"]]
fpr3_channels_balanced_2 = res_channels_balanced_2[["roc"]][["3"]][["fpr"]]
trp3_channels_balanced_2 = res_channels_balanced_2[["roc"]][["3"]][["tpr"]]
fpr4_channels_balanced_2 = res_channels_balanced_2[["roc"]][["4"]][["fpr"]]
trp4_channels_balanced_2 = res_channels_balanced_2[["roc"]][["4"]][["tpr"]]
trp_logit_channels_balanced = ROC_logit_channels_balanced[["sensitivities"]]
spec_logit_channels_balanced <- ROC_logit_channels_balanced[["specificities"]]
frp_logit_channels_balanced = c(1) - spec_logit_channels_balanced

plot(fpr1_channels_balanced_2,trp1_channels_balanced_2,type="l",xlab="False Positive Rate",ylab="True Positive Rate",main="ROC")
lines(fpr1_channels_balanced_2,trp1_channels_balanced_2,col="red")
lines(fpr2_channels_balanced_2,trp2_channels_balanced_2,col="blue")
lines(fpr3_channels_balanced_2,trp3_channels_balanced_2,col="green")
lines(fpr4_channels_balanced_2,trp4_channels_balanced_2,col="purple")
lines(frp_logit_channels_balanced, trp_logit_channels_balanced, col="pink")

legend("right", legend=c("First order","Second order", "Third order","Fourth order", "Logit"),
       col=c("red","blue", "green","purple", "pink"),lty=1)

# Considered
res_channels_considered_2=choose_order(Channels_considered_2, var_path="path", var_conv="total_conversions", var_null="total_null")
ROC_logit_channels_considered <- roc(test_logit_channels_considered$bin_conv, predicted_log_channels_considered)

```

```
fpr1_channels_considered_2 = res_channels_considered_2[["roc"]][["1"]][["fpr"]]
trp1_channels_considered_2 = res_channels_considered_2[["roc"]][["1"]][["tpr"]]
fpr2_channels_considered_2 = res_channels_considered_2[["roc"]][["2"]][["fpr"]]
trp2_channels_considered_2 = res_channels_considered_2[["roc"]][["2"]][["tpr"]]
fpr3_channels_considered_2 = res_channels_considered_2[["roc"]][["3"]][["fpr"]]
trp3_channels_considered_2 = res_channels_considered_2[["roc"]][["3"]][["tpr"]]
fpr4_channels_considered_2 = res_channels_considered_2[["roc"]][["4"]][["fpr"]]
trp4_channels_considered_2 = res_channels_considered_2[["roc"]][["4"]][["tpr"]]
trp_logit_channels_considered = ROC_logit_channels_considered[["sensitivities"]]
spec_logit_channels_considered <- ROC_logit_channels_considered[["specificities"]]
frp_logit_channels_considered = c(1) - spec_logit_channels_considered

plot(fpr1_channels_considered_2,trp1_channels_considered_2,type="l",xlab="False Positive Rate",ylab="True Positive Rate",main="ROC")
lines(fpr1_channels_considered_2,trp1_channels_considered_2,col="red")
lines(fpr2_channels_considered_2,trp2_channels_considered_2,col="blue")
lines(fpr3_channels_considered_2,trp3_channels_considered_2,col="green")
lines(fpr4_channels_considered_2,trp4_channels_considered_2,col="purple")
lines(frp_logit_channels_considered, trp_logit_channels_considered, col="pink")

legend("right", legend=c("First order","Second order", "Third order","Fourth order", "Logit"),
           col=c("red","blue", "green","purple", "pink"),lty=1)


# AUC
# Markov models -> values found in[U+FFFD]reschannels_impulsive_2[U+FFFD][U+FFFD]reschannels_balanced_2[U+FFFD][U+FFFD]reschannels_considered_2[U+FFFD]

# Logit - Impulsive
Channels_impulsive_2$conv_and_null <- Channels_impulsive_2$total_conversions + Channels_impulsive_2$total_null
```

```
Channels_impulsive_2logit <- Channels_impulsive_2 %>% slice(rep(1:n(),
    Channels_impulsive_2$conv_and_null))
Channels_impulsive_2logit <- Channels_impulsive_2logit %>%
group_by(path) %>%
mutate(row_number = 1:n())
Channels_impulsive_2logit$bin_conv <- ifelse(Channels_impulsive_2logit$row_
    number <= Channels_impulsive_2logit$total_conversions, 1, 0)
Channels_impulsive_2logit$bin_null <- ifelse(Channels_impulsive_2logit$bin_
    conv == 0, 1, 0)
Channels_occurences_channels_impulsive_logit <- Channels_occurences_channels_
    impulsive %>% slice(rep(1:n(), Channels_impulsive_2$conv_and_null))

logit_nonagg_channel_impulsive <- glm(formula = cbind(Channels_impulsive_2
    logit$bin_conv, Channels_impulsive_2logit$bin_null) ~ (Channels_occurences
    _channels_impulsive_logit$eta + Channels_occurences_channels_impulsive_
    logit$iota + Channels_occurences_channels_impulsive_logit$alpha + Channels
    _occurences_channels_impulsive_logit$beta + Channels_occurences_channels_
    impulsive_logit$theta + Channels_occurences_channels_impulsive_logit$
    lambda + Channels_occurences_channels_impulsive_logit$kappa + Channels_
    occurences_channels_impulsive_logit$zeta + Channels_occurences_channels_
    impulsive_logit$epsilon + Channels_occurences_channels_impulsive_logit$
    gamma), family = binomial("logit"), data = Channels_impulsive_2logit)

# Use 80% of dataset as training set and remaining 20% as testing set
sample_logit_channels_impulsive <- sample(c(TRUE, FALSE), nrow(Channels_
    impulsive_2logit), replace=TRUE, prob=c(0.8,0.2))
train_logit_channels_impulsive <- Channels_impulsive_2logit[sample_logit_
    channels_impulsive, ]
test_logit_channels_impulsive <- Channels_impulsive_2logit[!sample_logit_
    channels_impulsive, ]
Channels_occurences_nonagg_channels_impulsive <- Channels_occurences_channels_
    impulsive_logit[!sample_logit_channels_impulsive, ]

# Fit logistic regression model (on train)
model_logit_nonagg2_channels_impulsive <-  glm(formula = cbind(test_logit_
    channels_impulsive$bin_conv, test_logit_channels_impulsive$bin_null) ~ (
    Channels_occurences_nonagg_channels_impulsive$eta + Channels_occurences_
    nonagg_channels_impulsive$iota + Channels_occurences_nonagg_channels_
    impulsive$alpha + Channels_occurences_nonagg_channels_impulsive$beta +
    Channels_occurences_nonagg_channels_impulsive$theta + Channels_occurences_
    nonagg_channels_impulsive$lambda + Channels_occurences_nonagg_channels_
    impulsive$kappa + Channels_occurences_nonagg_channels_impulsive$zeta +
    Channels_occurences_nonagg_channels_impulsive$epsilon + Channels_
```

```
    occurences_nonagg_channels_impulsive$gamma), family = binomial("logit"),
    data = train_logit_channels_impulsive)
1128
1129 # Calculate probability of default for each individual in test dataset
1130 predicted_log_channels_impulsive <- predict(model_logit_nonagg2_channels_
    impulsive, test_logit_channels_impulsive, type="response")
1131 auc(test_logit_channels_impulsive$bin_conv, predicted_log_channels_impulsive)
1132
1133 # Logit - Balanced
1134 Channels_balanced_2$conv_and_null <- Channels_balanced_2$total_conversions +
    Channels_balanced_2$total_null
1135 Channels_balanced_2logit <- Channels_balanced_2 %>% slice(rep(1:n(), Channels_
    balanced_2$conv_and_null))
1136 Channels_balanced_2logit <- Channels_balanced_2logit %>%
1137 group_by(path) %>%
1138 mutate(row_number = 1:n())
1139 Channels_balanced_2logit$bin_conv <- ifelse(Channels_balanced_2logit$row_
    number <= Channels_balanced_2logit$total_conversions, 1, 0)
1140 Channels_balanced_2logit$bin_null <- ifelse(Channels_balanced_2logit$bin_conv
    == 0, 1, 0)
1141 Channels_occurences_channels_balanced_logit <- Channels_occurences_channels_
    balanced %>% slice(rep(1:n(), Channels_balanced_2$conv_and_null))
1142
1143 logit_nonagg_channel_balanced <- glm(formula = cbind(Channels_balanced_2logit$
    bin_conv, Channels_balanced_2logit$bin_null) ~ (Channels_occurences_
    channels_balanced_logit$eta + Channels_occurences_channels_balanced_logit$
    iota + Channels_occurences_channels_balanced_logit$alpha + Channels_
    occurences_channels_balanced_logit$beta + Channels_occurences_channels_
    balanced_logit$theta + Channels_occurences_channels_balanced_logit$lambda
    + Channels_occurences_channels_balanced_logit$kappa + Channels_occurences_
    channels_balanced_logit$zeta + Channels_occurences_channels_balanced_logit
    $epsilon + Channels_occurences_channels_balanced_logit$gamma), family =
    binomial("logit"), data = Channels_balanced_2logit)
1144
1145 # Use 80% of dataset as training set and remaining 20% as testing set
1146 sample_logit_channels_balanced <- sample(c(TRUE, FALSE), nrow(Channels_
    balanced_2logit), replace=TRUE, prob=c(0.8,0.2))
1147 train_logit_channels_balanced <- Channels_balanced_2logit[sample_logit_
    channels_balanced, ]
1148 test_logit_channels_balanced <- Channels_balanced_2logit[!sample_logit_
    channels_balanced, ]
1149 Channels_occurences_nonagg_channels_balanced <- Channels_occurences_channels_
    balanced_logit[!sample_logit_channels_balanced, ]
```

```
1150
1151 # Fit logistic regression model (on train)
1152 model_logit_nonagg2_channels_balanced <-  glm(formula = cbind(test_logit_
     channels_balanced$bin_conv, test_logit_channels_balanced$bin_null) ~ (
     Channels_occurences_nonagg_channels_balanced$eta + Channels_occurences_
     nonagg_channels_balanced$iota + Channels_occurences_nonagg_channels_
     balanced$alpha + Channels_occurences_nonagg_channels_balanced$beta +
     Channels_occurences_nonagg_channels_balanced$theta + Channels_occurences_
     nonagg_channels_balanced$lambda + Channels_occurences_nonagg_channels_
     balanced$kappa + Channels_occurences_nonagg_channels_balanced$zeta +
     Channels_occurences_nonagg_channels_balanced$epsilon + Channels_occurences
     _nonagg_channels_balanced$gamma), family = binomial("logit"), data = train
     _logit_channels_balanced)
1153
1154 # Calculate probability of default for each individual in test dataset
1155 predicted_log_channels_balanced <- predict(model_logit_nonagg2_channels_
     balanced, test_logit_channels_balanced, type="response")
1156 auc(test_logit_channels_balanced$bin_conv, predicted_log_channels_balanced)
1157
1158 # Logit - Considered
1159 Channels_considered_2$conv_and_null <- Channels_considered_2$total_conversions
      + Channels_considered_2$total_null
1160 Channels_considered_2logit <- Channels_considered_2 %>% slice(rep(1:n(),
     Channels_considered_2$conv_and_null))
1161 Channels_considered_2logit <- Channels_considered_2logit %>%
1162 group_by(path) %>%
1163 mutate(row_number = 1:n())
1164 Channels_considered_2logit$bin_conv <- ifelse(Channels_considered_2logit$row_
     number <= Channels_considered_2logit$total_conversions, 1, 0)
1165 Channels_considered_2logit$bin_null <- ifelse(Channels_considered_2logit$bin_
     conv == 0, 1, 0)
1166 Channels_occurences_channels_considered_logit <- Channels_occurences_channels_
     considered %>% slice(rep(1:n(), Channels_considered_2$conv_and_null))
1167
1168 logit_nonagg_channel_considered <- glm(formula = cbind(Channels_considered_2
     logit$bin_conv, Channels_considered_2logit$bin_null) ~ (Channels_
     occurences_channels_considered_logit$eta + Channels_occurences_channels_
     considered_logit$iota + Channels_occurences_channels_considered_logit$
     alpha + Channels_occurences_channels_considered_logit$beta + Channels_
     occurences_channels_considered_logit$theta + Channels_occurences_channels_
     considered_logit$lambda + Channels_occurences_channels_considered_logit$
     kappa + Channels_occurences_channels_considered_logit$zeta + Channels_
     occurences_channels_considered_logit$epsilon + Channels_occurences_
```

```
    channels_considered_logit$gamma), family = binomial("logit"), data =
    Channels_considered_2logit)
1169
1170 # Use 80% of dataset as training set and remaining 20% as testing set
1171 sample_logit_channels_considered <- sample(c(TRUE, FALSE), nrow(Channels_
    considered_2logit), replace=TRUE, prob=c(0.8,0.2))
1172 train_logit_channels_considered <- Channels_considered_2logit[sample_logit_
    channels_considered, ]
1173 test_logit_channels_considered <- Channels_considered_2logit[!sample_logit_
    channels_considered, ]
1174 Channels_occurences_nonagg_channels_considered <- Channels_occurences_channels
    _considered_logit[!sample_logit_channels_considered, ]
1175
1176 # Fit logistic regression model (on train)
1177 model_logit_nonagg2_channels_considered <-  glm(formula = cbind(test_logit_
    channels_considered$bin_conv, test_logit_channels_considered$bin_null) ~ (
    Channels_occurences_nonagg_channels_considered$eta + Channels_occurences_
    nonagg_channels_considered$iota + Channels_occurences_nonagg_channels_
    considered$alpha + Channels_occurences_nonagg_channels_considered$beta +
    Channels_occurences_nonagg_channels_considered$theta + Channels_occurences
    _nonagg_channels_considered$lambda + Channels_occurences_nonagg_channels_
    considered$kappa + Channels_occurences_nonagg_channels_considered$zeta +
    Channels_occurences_nonagg_channels_considered$epsilon + Channels_
    occurences_nonagg_channels_considered$gamma), family = binomial("logit"),
    data = train_logit_channels_considered)
1178
1179 # Calculate probability of default for each individual in test dataset
1180 predicted_log_channels_considered <- predict(model_logit_nonagg2_channels_
    considered, test_logit_channels_considered, type="response")
1181 auc(test_logit_channels_considered$bin_conv, predicted_log_channels_considered
    )
1182
1183 # TOP DECILE LIFT
1184 # Impulsive
1185 chisq_first_conversion_channels_impulsive_2 <- chisq.test(Heuristics_channels_
    impulsive_2$first_touch_conversions/sum(Heuristics_channels_impulsive_2$
    first_touch_conversions)*100)
1186 chisq_last_conversion_channels_impulsive_2 <- chisq.test(Heuristics_channels_
    impulsive_2$last_touch_conversions/sum(Heuristics_channels_impulsive_2$
    last_touch_conversions)*100)
1187 chisq_linear_conversion_channels_impulsive_2 <- chisq.test(Heuristics_channels
    _impulsive_2$linear_touch_conversions/sum(Heuristics_channels_impulsive_2$
    linear_touch_conversions)*100)
```

```
chisq_markov1_conversion_channels_impulsive_2 <- chisq.test(Markov1_channels_impulsive_2$total_conversion/sum(Markov1_channels_impulsive_2$total_conversion)*100)
chisq_markov2_conversion_channels_impulsive_2 <- chisq.test(Markov2_channels_impulsive_2$total_conversion/sum(Markov2_channels_impulsive_2$total_conversion)*100)
chisq_markov3_conversion_channels_impulsive_2 <- chisq.test(Markov3_channels_impulsive_2$total_conversion/sum(Markov3_channels_impulsive_2$total_conversion)*100)
chisq_markov4_conversion_channels_impulsive_2 <- chisq.test(Markov4_channels_impulsive_2$total_conversion/sum(Markov4_channels_impulsive_2$total_conversion)*100)

Expected_first_conversion_channels_impulsive_2 = chisq_first_conversion_channels_impulsive_2[["expected"]]
Observed_first_conversion_channels_impulsive_2 = chisq_first_conversion_channels_impulsive_2[["observed"]]
TDL_first_conversion_channels_impulsive_2 <- TopDecileLift(Expected_first_conversion_channels_impulsive_2, Observed_first_conversion_channels_impulsive_2)

Expected_last_conversion_channels_impulsive_2 = chisq_last_conversion_channels_impulsive_2[["expected"]]
Observed_last_conversion_channels_impulsive_2 = chisq_last_conversion_channels_impulsive_2[["observed"]]
TDL_last_conversion_channels_impulsive_2 <- TopDecileLift(Expected_last_conversion_channels_impulsive_2, Observed_last_conversion_channels_impulsive_2)

Expected_linear_conversion_channels_impulsive_2 = chisq_linear_conversion_channels_impulsive_2[["expected"]]
Observed_linear_conversion_channels_impulsive_2 = chisq_linear_conversion_channels_impulsive_2[["observed"]]
TDL_linear_conversion_channels_impulsive_2 <- TopDecileLift(Expected_linear_conversion_channels_impulsive_2, Observed_linear_conversion_channels_impulsive_2)

Expected_logit_channels_impulsive = sum(logit2_channels_impulsive$effects)/1558
Observed_logit_channels_impulsive = logit2_channels_impulsive$effects
TDL_logit_channels_impulsive <- TopDecileLift(Expected_logit_channels_impulsive, Observed_logit_channels_impulsive)

```

```
Expected_markov1_conversion_channels_impulsive_2 = chisq_markov1_conversion_channels_impulsive_2[["expected"]]
Observed_markov1_conversion_channels_impulsive_2 = chisq_markov1_conversion_channels_impulsive_2[["observed"]]
TDL_markov1_conversion_channels_impulsive_2 <- TopDecileLift(Expected_markov1_conversion_channels_impulsive_2, Observed_markov1_conversion_channels_impulsive_2)

Expected_markov2_conversion_channels_impulsive_2 = chisq_markov2_conversion_channels_impulsive_2[["expected"]]
Observed_markov2_conversion_channels_impulsive_2 = chisq_markov2_conversion_channels_impulsive_2[["observed"]]
TDL_markov2_conversion_channels_impulsive_2 <- TopDecileLift(Expected_markov2_conversion_channels_impulsive_2, Observed_markov2_conversion_channels_impulsive_2)

Expected_markov3_conversion_channels_impulsive_2 = chisq_markov3_conversion_channels_impulsive_2[["expected"]]
Observed_markov3_conversion_channels_impulsive_2 = chisq_markov3_conversion_channels_impulsive_2[["observed"]]
TDL_markov3_conversion_channels_impulsive_2 <- TopDecileLift(Expected_markov3_conversion_channels_impulsive_2, Observed_markov3_conversion_channels_impulsive_2)

Expected_markov4_conversion_channels_impulsive_2 = chisq_markov4_conversion_channels_impulsive_2[["expected"]]
Observed_markov4_conversion_channels_impulsive_2 = chisq_markov4_conversion_channels_impulsive_2[["observed"]]
TDL_markov4_conversion_channels_impulsive_2 <- TopDecileLift(Expected_markov4_conversion_channels_impulsive_2, Observed_markov4_conversion_channels_impulsive_2)

# Balanced
chisq_first_conversion_channels_balanced_2 <- chisq.test(Heuristics_channels_balanced_2$first_touch_conversions/sum(Heuristics_channels_balanced_2$first_touch_conversions)*100)
chisq_last_conversion_channels_balanced_2 <- chisq.test(Heuristics_channels_balanced_2$last_touch_conversions/sum(Heuristics_channels_balanced_2$last_touch_conversions)*100)
chisq_linear_conversion_channels_balanced_2 <- chisq.test(Heuristics_channels_balanced_2$linear_touch_conversions/sum(Heuristics_channels_balanced_2$linear_touch_conversions)*100)
```

```
1229 chisq_markov1_conversion_channels_balanced_2 <- chisq.test(Markov1_channels_
         balanced_2$total_conversion/sum(Markov1_channels_balanced_2$total_
         conversion)*100)
1230 chisq_markov2_conversion_channels_balanced_2 <- chisq.test(Markov2_channels_
         balanced_2$total_conversion/sum(Markov2_channels_balanced_2$total_
         conversion)*100)
1231 chisq_markov3_conversion_channels_balanced_2 <- chisq.test(Markov3_channels_
         balanced_2$total_conversion/sum(Markov3_channels_balanced_2$total_
         conversion)*100)
1232 chisq_markov4_conversion_channels_balanced_2 <- chisq.test(Markov4_channels_
         balanced_2$total_conversion/sum(Markov4_channels_balanced_2$total_
         conversion)*100)
1233
1234 Expected_first_conversion_channels_balanced_2 = chisq_first_conversion_
         channels_balanced_2[["expected"]]
1235 Observed_first_conversion_channels_balanced_2 = chisq_first_conversion_
         channels_balanced_2[["observed"]]
1236 TDL_first_conversion_channels_balanced_2 <- TopDecileLift(Expected_first_
         conversion_channels_balanced_2, Observed_first_conversion_channels_
         balanced_2)
1237
1238 Expected_last_conversion_channels_balanced_2 = chisq_last_conversion_channels_
         balanced_2[["expected"]]
1239 Observed_last_conversion_channels_balanced_2 = chisq_last_conversion_channels_
         balanced_2[["observed"]]
1240 TDL_last_conversion_channels_balanced_2 <- TopDecileLift(Expected_last_
         conversion_channels_balanced_2, Observed_last_conversion_channels_balanced
         _2)
1241
1242 Expected_linear_conversion_channels_balanced_2 = chisq_linear_conversion_
         channels_balanced_2[["expected"]]
1243 Observed_linear_conversion_channels_balanced_2 = chisq_linear_conversion_
         channels_balanced_2[["observed"]]
1244 TDL_linear_conversion_channels_balanced_2 <- TopDecileLift(Expected_linear_
         conversion_channels_balanced_2, Observed_linear_conversion_channels_
         balanced_2)
1245
1246 Expected_logit_channels_balanced = sum(logit2_channels_balanced$effects)/6540
1247 Observed_logit_channels_balanced = logit2_channels_balanced$effects
1248 TDL_logit_channels_balanced <- TopDecileLift(Expected_logit_channels_balanced,
          Observed_logit_channels_balanced)
1249
```

```
Expected_markov1_conversion_channels_balanced_2 = chisq_markov1_conversion_channels_balanced_2[["expected"]]
Observed_markov1_conversion_channels_balanced_2 = chisq_markov1_conversion_channels_balanced_2[["observed"]]
TDL_markov1_conversion_channels_balanced_2 <- TopDecileLift(Expected_markov1_conversion_channels_balanced_2, Observed_markov1_conversion_channels_balanced_2)

Expected_markov2_conversion_channels_balanced_2 = chisq_markov2_conversion_channels_balanced_2[["expected"]]
Observed_markov2_conversion_channels_balanced_2 = chisq_markov2_conversion_channels_balanced_2[["observed"]]
TDL_markov2_conversion_channels_balanced_2 <- TopDecileLift(Expected_markov2_conversion_channels_balanced_2, Observed_markov2_conversion_channels_balanced_2)

Expected_markov3_conversion_channels_balanced_2 = chisq_markov3_conversion_channels_balanced_2[["expected"]]
Observed_markov3_conversion_channels_balanced_2 = chisq_markov3_conversion_channels_balanced_2[["observed"]]
TDL_markov3_conversion_channels_balanced_2 <- TopDecileLift(Expected_markov3_conversion_channels_balanced_2, Observed_markov3_conversion_channels_balanced_2)

Expected_markov4_conversion_channels_balanced_2 = chisq_markov4_conversion_channels_balanced_2[["expected"]]
Observed_markov4_conversion_channels_balanced_2 = chisq_markov4_conversion_channels_balanced_2[["observed"]]
TDL_markov4_conversion_channels_balanced_2 <- TopDecileLift(Expected_markov4_conversion_channels_balanced_2, Observed_markov4_conversion_channels_balanced_2)

# Considered
chisq_first_conversion_channels_considered_2 <- chisq.test(Heuristics_channels_considered_2$first_touch_conversions/sum(Heuristics_channels_considered_2$first_touch_conversions)*100)
chisq_last_conversion_channels_considered_2 <- chisq.test(Heuristics_channels_considered_2$last_touch_conversions/sum(Heuristics_channels_considered_2$last_touch_conversions)*100)
chisq_linear_conversion_channels_considered_2 <- chisq.test(Heuristics_channels_considered_2$linear_touch_conversions/sum(Heuristics_channels_considered_2$linear_touch_conversions)*100)
```

```
chisq_markov1_conversion_channels_considered_2 <- chisq.test(Markov1_channels_considered_2$total_conversion/sum(Markov1_channels_considered_2$total_conversion)*100)
chisq_markov2_conversion_channels_considered_2 <- chisq.test(Markov2_channels_considered_2$total_conversion/sum(Markov2_channels_considered_2$total_conversion)*100)
chisq_markov3_conversion_channels_considered_2 <- chisq.test(Markov3_channels_considered_2$total_conversion/sum(Markov3_channels_considered_2$total_conversion)*100)
chisq_markov4_conversion_channels_considered_2 <- chisq.test(Markov4_channels_considered_2$total_conversion/sum(Markov4_channels_considered_2$total_conversion)*100)

Expected_first_conversion_channels_considered_2 = chisq_first_conversion_channels_considered_2[["expected"]]
Observed_first_conversion_channels_considered_2 = chisq_first_conversion_channels_considered_2[["observed"]]
TDL_first_conversion_channels_considered_2 <- TopDecileLift(Expected_first_conversion_channels_considered_2, Observed_first_conversion_channels_considered_2)

Expected_last_conversion_channels_considered_2 = chisq_last_conversion_channels_considered_2[["expected"]]
Observed_last_conversion_channels_considered_2 = chisq_last_conversion_channels_considered_2[["observed"]]
TDL_last_conversion_channels_considered_2 <- TopDecileLift(Expected_last_conversion_channels_considered_2, Observed_last_conversion_channels_considered_2)

Expected_linear_conversion_channels_considered_2 = chisq_linear_conversion_channels_considered_2[["expected"]]
Observed_linear_conversion_channels_considered_2 = chisq_linear_conversion_channels_considered_2[["observed"]]
TDL_linear_conversion_channels_considered_2 <- TopDecileLift(Expected_linear_conversion_channels_considered_2, Observed_linear_conversion_channels_considered_2)

Expected_logit_channels_considered = sum(logit2_channels_considered$effects)/1887
Observed_logit_channels_considered = logit2_channels_considered$effects
TDL_logit_channels_considered <- TopDecileLift(Expected_logit_channels_considered, Observed_logit_channels_considered)

```

```
Expected_markov1_conversion_channels_considered_2 = chisq_markov1_conversion_channels_considered_2[["expected"]]
Observed_markov1_conversion_channels_considered_2 = chisq_markov1_conversion_channels_considered_2[["observed"]]
TDL_markov1_conversion_channels_considered_2 <- TopDecileLift(Expected_markov1_conversion_channels_considered_2, Observed_markov1_conversion_channels_considered_2)

Expected_markov2_conversion_channels_considered_2 = chisq_markov2_conversion_channels_considered_2[["expected"]]
Observed_markov2_conversion_channels_considered_2 = chisq_markov2_conversion_channels_considered_2[["observed"]]
TDL_markov2_conversion_channels_considered_2 <- TopDecileLift(Expected_markov2_conversion_channels_considered_2, Observed_markov2_conversion_channels_considered_2)

Expected_markov3_conversion_channels_considered_2 = chisq_markov3_conversion_channels_considered_2[["expected"]]
Observed_markov3_conversion_channels_considered_2 = chisq_markov3_conversion_channels_considered_2[["observed"]]
TDL_markov3_conversion_channels_considered_2 <- TopDecileLift(Expected_markov3_conversion_channels_considered_2, Observed_markov3_conversion_channels_considered_2)

Expected_markov4_conversion_channels_considered_2 = chisq_markov4_conversion_channels_considered_2[["expected"]]
Observed_markov4_conversion_channels_considered_2 = chisq_markov4_conversion_channels_considered_2[["observed"]]
TDL_markov4_conversion_channels_considered_2 <- TopDecileLift(Expected_markov4_conversion_channels_considered_2, Observed_markov4_conversion_channels_considered_2)

# REMOVAL EFFECT + TRANSITION MATRIX
# Impulsive
m1_channels_impulsive_2 = markov_model(Channels_impulsive_2, "path", "total_conversions", var_value="total_conversion_value", var_null="total_null", out_more=TRUE, order = 1)
m2_channels_impulsive_2 = markov_model(Channels_impulsive_2, "path", "total_conversions", var_value="total_conversion_value", var_null="total_null", out_more=TRUE, order = 2)
m3_channels_impulsive_2 = markov_model(Channels_impulsive_2, "path", "total_conversions", var_value="total_conversion_value", var_null="total_null", out_more=TRUE, order = 3)
```

```
m4_channels_impulsive_2 = markov_model(Channels_impulsive_2, "path", "total_
    conversions", var_value="total_conversion_value", var_null="total_null",
    out_more=TRUE, order = 4)

# Balanced
m1_channels_balanced_2 = markov_model(Channels_balanced_2, "path", "total_
    conversions", var_value="total_conversion_value", var_null="total_null",
    out_more=TRUE, order = 1)
m2_channels_balanced_2 = markov_model(Channels_balanced_2, "path", "total_
    conversions", var_value="total_conversion_value", var_null="total_null",
    out_more=TRUE, order = 2)
m3_channels_balanced_2 = markov_model(Channels_balanced_2, "path", "total_
    conversions", var_value="total_conversion_value", var_null="total_null",
    out_more=TRUE, order = 3)
m4_channels_balanced_2 = markov_model(Channels_balanced_2, "path", "total_
    conversions", var_value="total_conversion_value", var_null="total_null",
    out_more=TRUE, order = 4)

# Considered
m1_channels_considered_2 = markov_model(Channels_considered_2, "path", "total_
    conversions", var_value="total_conversion_value", var_null="total_null",
    out_more=TRUE, order = 1)
m2_channels_considered_2 = markov_model(Channels_considered_2, "path", "total_
    conversions", var_value="total_conversion_value", var_null="total_null",
    out_more=TRUE, order = 2)
m3_channels_considered_2 = markov_model(Channels_considered_2, "path", "total_
    conversions", var_value="total_conversion_value", var_null="total_null",
    out_more=TRUE, order = 3)
m4_channels_considered_2 = markov_model(Channels_considered_2, "path", "total_
    conversions", var_value="total_conversion_value", var_null="total_null",
    out_more=TRUE, order = 4)

# STANDARD DEVIATION REMOVAL EFFECT
# Impulsive
removal_effects_m1_channels_impulsive_2 <- m1_channels_impulsive_2[["removal_
    effects"]][["removal_effects_conversion"]]
sum(removal_effects_m1_channels_impulsive_2)/9
sd(removal_effects_m1_channels_impulsive_2)
sd(removal_effects_m1_channels_impulsive_2)/(sum(removal_effects_m1_channels_
    impulsive_2)/9)

removal_effects_m2_channels_impulsive_2 <- m2_channels_impulsive_2[["removal_
    effects"]][["removal_effects_conversion"]]
```

```
sum(removal_effects_m2_channels_impulsive_2)/9
sd(removal_effects_m2_channels_impulsive_2)
sd(removal_effects_m2_channels_impulsive_2)/(sum(removal_effects_m2_channels_
    impulsive_2)/9)

removal_effects_m3_channels_impulsive_2 <- m3_channels_impulsive_2[["removal_
    effects"]][["removal_effects_conversion"]]
sum(removal_effects_m3_channels_impulsive_2)/9
sd(removal_effects_m3_channels_impulsive_2)
sd(removal_effects_m3_channels_impulsive_2)/(sum(removal_effects_m3_channels_
    impulsive_2)/9)

removal_effects_m4_channels_impulsive_2 <- m4_channels_impulsive_2[["removal_
    effects"]][["removal_effects_conversion"]]
sum(removal_effects_m4_channels_impulsive_2)/9
sd(removal_effects_m4_channels_impulsive_2)
sd(removal_effects_m4_channels_impulsive_2)/(sum(removal_effects_m4_channels_
    impulsive_2)/9)

# Balanced
removal_effects_m1_channels_balanced_2 <- m1_channels_balanced_2[["removal_
    effects"]][["removal_effects_conversion"]]
sum(removal_effects_m1_channels_balanced_2)/10
sd(removal_effects_m1_channels_balanced_2)
sd(removal_effects_m1_channels_balanced_2)/(sum(removal_effects_m1_channels_
    balanced_2)/10)

removal_effects_m2_channels_balanced_2 <- m2_channels_balanced_2[["removal_
    effects"]][["removal_effects_conversion"]]
sum(removal_effects_m2_channels_balanced_2)/10
sd(removal_effects_m2_channels_balanced_2)
sd(removal_effects_m2_channels_balanced_2)/(sum(removal_effects_m2_channels_
    balanced_2)/10)

removal_effects_m3_channels_balanced_2 <- m3_channels_balanced_2[["removal_
    effects"]][["removal_effects_conversion"]]
sum(removal_effects_m3_channels_balanced_2)/10
sd(removal_effects_m3_channels_balanced_2)
sd(removal_effects_m3_channels_balanced_2)/(sum(removal_effects_m3_channels_
    balanced_2)/10)

removal_effects_m4_channels_balanced_2 <- m4_channels_balanced_2[["removal_
    effects"]][["removal_effects_conversion"]]
```

```
sum(removal_effects_m4_channels_balanced_2)/10
sd(removal_effects_m4_channels_balanced_2)
sd(removal_effects_m4_channels_balanced_2)/(sum(removal_effects_m4_channels_
    balanced_2)/10)

# Considered
removal_effects_m1_channels_considered_2 <- m1_channels_considered_2[["removal
    _effects"]][["removal_effects_conversion"]]
sum(removal_effects_m1_channels_considered_2)/10
sd(removal_effects_m1_channels_considered_2)
sd(removal_effects_m1_channels_considered_2)/(sum(removal_effects_m1_channels_
    considered_2)/10)

removal_effects_m2_channels_considered_2 <- m2_channels_considered_2[["removal
    _effects"]][["removal_effects_conversion"]]
sum(removal_effects_m2_channels_considered_2)/10
sd(removal_effects_m2_channels_considered_2)
sd(removal_effects_m2_channels_considered_2)/(sum(removal_effects_m2_channels_
    considered_2)/10)

removal_effects_m3_channels_considered_2 <- m3_channels_considered_2[["removal
    _effects"]][["removal_effects_conversion"]]
sum(removal_effects_m3_channels_considered_2)/10
sd(removal_effects_m3_channels_considered_2)
sd(removal_effects_m3_channels_considered_2)/(sum(removal_effects_m3_channels_
    considered_2)/10)

removal_effects_m4_channels_considered_2 <- m4_channels_considered_2[["removal
    _effects"]][["removal_effects_conversion"]]
sum(removal_effects_m4_channels_considered_2)/10
sd(removal_effects_m4_channels_considered_2)
sd(removal_effects_m4_channels_considered_2)/(sum(removal_effects_m4_channels_
    considered_2)/10)
```