# Package 'ChannelAttribution'

May 7, 2026

**Type** Package

**Title** Markov Model for Online Multi-Channel Attribution

**Version** 2.2.4

**Date** 2025-11-19

**Maintainer** Davide Altomare <info@channelattribution.io>

**Description** Advertisers use a variety of online marketing channels to reach consumers and they want to know the degree each channel contributes to their marketing success. This is called online multi-channel attribution problem. This package contains a probabilistic algorithm for the attribution problem. The model uses a k-order Markov representation to identify structural correlations in the customer journey data. The package also contains three heuristic algorithms (first-touch, last-touch and linear-touch approach) for the same problem. The algorithms are implemented in C++.

**License** GPL-3 | file LICENSE

**URL** https://channelattribution.io

**LinkingTo** Rcpp, RcppArmadillo

**Imports** Rcpp

**Suggests** curl, jsonlite

**NeedsCompilation** yes

**Author** Davide Altomare [cre, aut],
David Loris [aut]

**Repository** CRAN

**Date/Publication** 2025-11-19 13:20:22 UTC

## Contents

---

`ChannelAttribution-package`

*Markov Model for Online Multi-Channel Attribution*

---

## Description

Advertisers use a variety of online marketing channels to reach consumers and they want to know the degree each channel contributes to their marketing success. This is called online multi-channel attribution problem. In many cases, advertisers approach this problem through some simple heuristics methods that do not take into account any customer interactions and often tend to underestimate the importance of small channels in marketing contribution. This package provides a function that approaches the attribution problem in a probabilistic way. It uses a k-order Markov representation to identify structural correlations in the customer journey data. This would allow advertisers to give a more reliable assessment of the marketing contribution of each channel. The approach basically follows the one presented in Eva Anderl, Ingo Becker, Florian v. Wangenheim, Jan H. Schumann (2014). Differently for them, we solved the estimation process using stochastic simulations. In this way it is also possible to take into account conversion values and their variability in the computation of the channel importance. The package also contains a function that estimates three heuristic models (first-touch, last-touch and linear-touch approach) for the same problem.

## Details

| | |
|---|---|
| Package: | ChannelAttribution |
| Type: | Package |
| Version: | 2.2.4 |
| Date: | 2025-11-19 |
| License: | GPL (>= 2) |

Package contains functions for channel attribution in web marketing.

## Author(s)

Davide Altomare, David Loris

Maintainer Davide Altomare <info@channelattribution.io>

## References

ChannelAttribution Official Website: https://channelattribution.io

Eva Anderl, Ingo Becker, Florian v. Wangenheim, Jan H. Schumann: Mapping the Customer Journey, 2014, doi:10.2139/ssrn.2343077

`auto_markov_model` *Automatic Markov Model.*

## Description

Estimate a Markov model from customer journey data after automatically choosing a suitable order. It requires paths that do not lead to conversion as input.

## Usage

```
auto_markov_model(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, flg_pro=TRUE)
```

## Arguments

| | |
|---|---|
| `Data` | data.frame containing customer journeys data. |
| `var_path` | column name containing paths. |
| `var_conv` | column name containing total conversions. |
| `var_null` | column name containing total paths that do not lead to conversions. |
| `var_value` | column name containing total conversion value. |
| `max_order` | maximum Markov Model order considered. |
| `roc_npt` | number of points used for approximating roc and auc. |
| `plot` | if TRUE, a plot with penalized auc with respect to order will be displayed. |
| `nsim_start` | minimum number of simulations used in computation. |
| `max_step` | maximum number of steps for a single simulated path. if NULL, it is the maximum number of steps found into Data. |
| `out_more` | if TRUE, transition probabilities between channels and removal effects will be shown. |
| `sep` | separator between the channels. |
| `ncore` | number of threads used in computation. |
| `nfold` | how many repetitions are used to verify if convergence is reached at each iteration. |
| `seed` | random seed. Giving this parameter the same value over different runs guarantees that results will not vary. |
| `conv_par` | convergence parameter for the algorithm. The estimation process ends when the percentage of variation of the results over different repetitions is less than convergence parameter. |
| `rate_step_sim` | number of simulations used at each iteration is equal to the number of simulations used at previous iteration multiplied by rate_step_sim. |
| `verbose` | if TRUE, additional information about process convergence will be shown. |
| `flg_pro` | if TRUE, ChannelAttribution Pro banner is printed. |

## Value

An object of class data.frame with the estimated number of conversions and the estimated conversion value attributed to each channel.

## Author(s)

Davide Altomare (<info@channelattribution.io>).

## Examples

```
## Not run:

library(ChannelAttribution)

data(PathData)

auto_markov_model(Data, "path", "total_conversions", "total_null")


## End(Not run)
```

---

# choose_order *Choose order for Markov model.*

---

## Description

Find the minimum Markov Model order that gives a good representation of customers' behaviour for data considered. It requires paths that do not lead to conversion as input. Minimum order is found maximizing a penalized area under ROC curve.

## Usage

```
choose_order(Data, var_path, var_conv, var_null, max_order=10, sep=">",
             ncore=1, roc_npt=100, plot=TRUE, flg_pro=TRUE)
```

## Arguments

| | |
|---|---|
| Data | data.frame containing customer journeys. |
| var_path | column name of Data containing paths. |
| var_conv | column name of Data containing total conversions. |
| var_null | column name of Data containing total paths that do not lead to conversion. |
| max_order | maximum Markov Model order considered. |
| sep | separator between channels. |
| ncore | number of threads used in computation. |

| | |
|---|---|
| roc_npt | number of points used for approximating roc and auc. |
| plot | if TRUE, a plot with penalized auc with respect to order will be displayed. |
| flg_pro | if TRUE, ChannelAttribution Pro banner is printed. |

## Value

An object of class List with the estimated roc, auc and penalized auc.

## Author(s)

Davide Altomare (<info@channelattribution.io>).

## Examples

```
## Not run:

library(ChannelAttribution)

data(PathData)

res=choose_order(Data, var_path="path", var_conv="total_conversions",
                 var_null="total_null")

#plot auc and penalized auc

plot(res$auc$order,res$auc$auc,type="l",xlab="order",ylab="pauc",main="AUC")
lines(res$auc$order,res$auc$pauc,col="red")
legend("right", legend=c("auc","penalized auc"),
       col=c("black","red"),lty=1)


## End(Not run)
```

---

# Data *Customer journeys data.*

---

## Description

Example dataset.

## Usage

```
data(PathData)
```

## Format

Data is a data.frame with 10.000 rows and 4 columns: "path" containing customer paths, "total_conversions" containing total number of conversions, "total_conversion_value" containing total conversion value and "total_null" containing total number of paths that do not lead to conversion.

heuristic_models *Heuristic models for the online attribution problem.*

## Description

Estimate theree heuristic models (first-touch, last-touch and linear) from customer journey data.

## Usage

```
heuristic_models(Data, var_path, var_conv, var_value=NULL, sep=">", flg_pro=TRUE)
```

## Arguments

| | |
|---|---|
| Data | data.frame containing paths and conversions. |
| var_path | column name containing paths. |
| var_conv | column name containing total conversions. |
| var_value | column name containing total conversion value. |
| sep | separator between the channels. |
| flg_pro | if TRUE, ChannelAttribution Pro banner is printed. |

## Value

An object of class data.frame with the estimated number of conversions and the estimated conversion value attributed to each channel for each model.

## Author(s)

Davide Altomare (<info@channelattribution.io>).

## Examples

```
## Not run:

library(ChannelAttribution)

data(PathData)

heuristic_models(Data,"path","total_conversions")
heuristic_models(Data,"path","total_conversions",var_value="total_conversion_value")


## End(Not run)
```

`install_pro` *Install ChannelAttributionPro binary tailored to your OS/arch/R*

## Description

Interactively installs the **ChannelAttributionPro** package by contacting the ChannelAttribution build service, which returns a platform-appropriate binary (or source) package URL for your current operating system, CPU architecture, and R version. At the prompt you can enter:

- a *token* (recommended), or
- a work/university *email* to request a token (the function sends the request and exits).

## Usage

```
install_pro()
```

## Details

When called, the function asks you to enter your ChannelAttributionPro *token*. If you don't have one, you can enter your work/university email; the function will request a token for that address and return invisibly, so you can re-run it once the token arrives.

Processing steps:

1. Detect your platform (`os`, `os_vers`, `arch`) and R minor version.
2. Send a form-encoded request to the ChannelAttribution builder endpoint to obtain a package URL.
3. Install the returned package via `install.packages` with `repos = NULL`.
4. On exit, send a small status message to a monitoring endpoint with minimal text (no local files).

**Dependencies:** This function requires **curl** (for HTTPS) and **jsonlite** (for JSON parsing). If either is missing, the function aborts with a friendly message.

**Networking and privacy:** The function performs outbound HTTPS requests to `https://app.channelattribution.io`, sending only the minimal parameters needed to resolve the correct binary and (on exit) a small status string for diagnostics. No local files or datasets are uploaded.

**Interactive prompt:** Input is collected via base R `readline()` (no masking). In some environments the input may be visible.

## Value

Invisibly returns `NULL`. Progress and outcome messages are printed.

### Errors and messages

- *Invalid or expired token (HTTP 401)*: You will see a message inviting you to contact <info@channelattribution.io> or request a new token.
- *Network/SSL issues*: Reported as a concise `network_or_ssl_error: ...`.
- *Unexpected builder response*: The function prints diagnostic information (platform and R version) that you can forward to support.

### Author(s)

ChannelAttribution Team

### See Also

`install.packages`

### Examples

```
## Not run:
## Typical interactive use:
install_pro()

## If you don't have a token at prompt:
##   - enter your work/university email to request one
##   - check your inbox (including spam)
##   - re-run install_pro() with the token

## End(Not run)
```

---

## markov_model *Markov model for the online attribution problem.*

---

### Description

Estimate a k-order Markov model from customer journey data. Differently from markov_model, this function iterates estimation until convergence is reached and enables multiprocessing.

### Usage

```
markov_model(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, flg_pro=TRUE)
```

## Arguments

| | |
|---|---|
| Data | data.frame containing customer journeys data. |
| var_path | column name containing paths. |
| var_conv | column name containing total conversions. |
| var_value | column name containing total conversion value. |
| var_null | column name containing total paths that do not lead to conversions. |
| order | Markov Model order. |
| nsim_start | minimum number of simulations used in computation. |
| max_step | maximum number of steps for a single simulated path. if NULL, it is the maximum number of steps found into Data. |
| out_more | if TRUE, transition probabilities between channels and removal effects will be returned. |
| sep | separator between the channels. |
| ncore | number of threads used in computation. |
| nfold | how many repetitions are used to verify if convergence has been reached at each iteration. |
| seed | random seed. Giving this parameter the same value over different runs guarantees that results will not vary. |
| conv_par | convergence parameter for the algorithm. The estimation process ends when the percentage of variation of the results over different repetions is less than convergence parameter. |
| rate_step_sim | number of simulations used at each iteration is equal to the number of simulations used at previous iteration multiplied by rate_step_sim. |
| verbose | if TRUE, additional information about process convergence will be shown. |
| flg_pro | if TRUE, ChannelAttribution Pro banner is printed. |

## Value

An object of class data.frame with the estimated number of conversions and the estimated conversion value attributed to each channel.

## Author(s)

Davide Altomare (<info@channelattribution.io>).

## Examples

```
## Not run:

library(ChannelAttribution)

data(PathData)

#Estimate a Makov model using total conversions
```

```
markov_model(Data, var_path="path", "total_conversions")

#Estimate a Makov model using total conversions and revenues
markov_model(Data, "path", "total_conversions",
var_value="total_conversion_value")

#Estimate a Makov model using total conversions, revenues and paths that do not lead to conversions
markov_model(Data, "path", "total_conversions",
var_value="total_conversion_value", var_null="total_null")

#Estimate a Makov model returning transition matrix and removal effects
markov_model(Data, "path", "total_conversions",
var_value="total_conversion_value", var_null="total_null", out_more=TRUE)

#Estimate a Markov model using 4 threads
markov_model(Data, "path", "total_conversions",
var_value="total_conversion_value", ncore=4)


## End(Not run)
```

---

`transition_matrix` *Transition matrix.*

---

## Description

Estimate a k-order transition matrix from customer journey data.

## Usage

```
transition_matrix(Data, var_path, var_conv, var_null, order=1, sep=">",
                  flg_equal=TRUE, flg_pro=TRUE)
```

## Arguments

| | |
|---|---|
| `Data` | data.frame containing customer journeys data. |
| `var_path` | column name containing paths. |
| `var_conv` | column name containing total conversions. |
| `var_null` | column name containing paths that do not lead to conversions. |
| `order` | Markov Model order. |
| `sep` | separator between the channels. |
| `flg_equal` | if TRUE, transitions from a channel to itself will be considered. |
| `flg_pro` | if TRUE, ChannelAttribution Pro banner is printed. |

## Value

An object of `class List` containing a dataframe with channel names and a dataframe with the estimated transition matrix.

## Author(s)

Davide Altomare (<info@channelattribution.io>).

## Examples

```
## Not run:

library(ChannelAttribution)

data(PathData)

transition_matrix(Data, var_path="path", var_conv="total_conversions",
                  var_null="total_null", order=1, sep=">", flg_equal=TRUE)

transition_matrix(Data, var_path="path", var_conv="total_conversions",
                  var_null="total_null", order=3, sep=">", flg_equal=TRUE)


## End(Not run)
```

# Index