Package ‘ChannelAttribution’

April 6, 2020

Type Package
Title Markov Model for the Online Multi-Channel Attribution Problem
Version 1.17
Date 2020-04-03
Author Davide Altomare, David Loris
Maintainer Davide Altomare <davide.altomare@gmail.com>

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 the 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 (>= 2)

URL http://www.slideshare.net/adavide1982/markov-model-for-the-multichannel-attribution-problem
http://www.lunametrics.com/blog/2016/06/30/marketing-channel-attribution-markov-models-r/
http://analyzecore.com/2016/08/03/attribution-model-r-part-1/

LinkingTo Rcpp, RcppArmadillo
Imports Rcpp

NeedsCompilation yes
Repository CRAN
Date/Publication 2020-04-06 12:42:14 UTC

R topics documented:

ChannelAttribution-package ........................................ 2
auto_markov_model .................................................. 3
choose_order ......................................................... 4
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 the 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: 1.17
Date: 2020-04-03
License: GPL (>= 2)

Package contains functions for channel attribution in web marketing.

Author(s)

Davide Altomare, David Loris

Maintainer Davide Altomare <davide.altomare@gmail.com>
**auto_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**

```r
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=Inf, nfold=10, seed=0, conv_par=0.05, rate_step_sim=1.5,
  verbose=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 is displayed.
- **nsim_start** 
  minimum number of simulations used in computation.
- **max_step** 
  maximum number of steps for a single simulated path.
- **out_more** 
  if TRUE returns the transition probabilities between channels and removal effects.
- **sep** 
  separator between the channels.
- **ncore** 
  number of threads used in computation. Default is number of CPUs available.
- **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.

**References**


choose_order

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  show additional information about process convergence.

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 (<davide.altomare@gmail.com>.

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=Inf, roc_npt=100, plot=TRUE)
choose_order

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. Default is number of CPUs available.
roc_npt number of points used for approximating roc and auc.
plot if TRUE, a plot with penalized auc with respect to order is displayed.

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

Author(s)
Davide Altomare (<davide.altomare@gmail.com>).

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="auc",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 three heuristic models (first-touch, last-touch, linear) from customer journey data.

Usage
heuristic_models(Data, var_path, var_conv, var_value=NULL, sep=">")

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.

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 (<davide.altomare@gmail.com>).
Examples

```r
## 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)
```

markov_model

Markov model for the online attribution problem.

Description

Estimate a k-order Markov model from customer journey data.

Usage

```r
markov_model(Data, var_path, var_conv, var_value=NULL, var_null=NULL, 
order=1, nsim=NULL, max_step=NULL, out_more=FALSE, sep=">", 
seed=NULL)
```

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.
- **var_null**: column name containing total paths that do not lead to conversions.
- **order**: Markov Model order.
- **nsim**: total simulations from transition matrix.
- **max_step**: maximum number of steps for a single simulated path.
- **out_more**: if TRUE returns the transition probabilities between channels and removal effects.
- **sep**: separator between the channels.
- **seed**: random seed. Giving to this parameter the same value over different runs guarantee that results will not vary.
**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 (<davide.altomare@gmail.com>.

**Examples**

```r
## Not run:
library(ChannelAttribution)
data(PathData)

markov_model(Data, "path", "total_conversions")
markov_model(Data, "path", "total_conversions", var_value="total_conversion_value")
markov_model(Data,"path","total_conversions", var_value="total_conversion_value", var_null="total_null")
markov_model(Data, "path", "total_conversions", var_value="total_conversion_value", var_null="total_null", out_more=TRUE)

## End(Not run)
```

---

**Description**

This function is a multiprocessing version of `markov_model` function.

**Usage**

```r
markov_model_mp(Data, var_path, var_conv, var_value=NULL, var_null=NULL, order=1, nsim_start=1e5, max_step=NULL, out_more=FALSE, sep=">", ncore=Inf, nfold=10, seed=0, conv_par=0.05, rate_step_sim=1.5, verbose=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.
**markov_model_mp**

- **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.
- **out_more**: if TRUE returns the transition probabilities between channels and removal effects.
- **sep**: separator between the channels.
- **ncore**: number of threads used in computation. Default is number of CPUs available.
- **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**: show additional information about process convergence.

**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 (<davide.altomare@gmail.com>).

**Examples**

```r
## Not run:
library(ChannelAttribution)
data(PathData)
markov_model_mp(Data,"path","total_conversions", var_null="total_null")

## 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)
```

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 are considered.

Value

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

Author(s)

Davide Altomare (<davide.altomare@gmail.com>).

Examples

```r
## 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)
```
transition_matrix

## End(Not run)
Index

*Topic **channel attribution**
  ChannelAttribution-package, 2

*Topic **channel marketing**
  ChannelAttribution-package, 2

*Topic **choose markov graph order**
  choose_order, 4

*Topic **choose markov model order**
  choose_order, 4

*Topic **customer journey dataset**
  Data, 6

*Topic **customer journey**
  ChannelAttribution-package, 2

*Topic **customer path data**
  Data, 6

*Topic **dataset**
  Data, 6

*Topic **first touch**
  heuristic_models, 6

*Topic **last touch**
  heuristic_models, 6

*Topic **linear touch**
  heuristic_models, 6

*Topic **marketing attribution**
  ChannelAttribution-package, 2

*Topic **markov graph**
  auto_markov_model, 3
  markov_model, 7
  markov_model_mp, 8
  transition_matrix, 10

*Topic **markov model**
  auto_markov_model, 3
  markov_model, 7
  markov_model_mp, 8
  transition_matrix, 10

*Topic **multi channel funnel**
  ChannelAttribution-package, 2

*Topic **multi channel marketing**
  ChannelAttribution-package, 2

*Topic **online attribution**

  ChannelAttribution-package, 2

*Topic **web marketing**
  ChannelAttribution-package, 2

*Topic **web statistics**
  ChannelAttribution-package, 2
  choose_order, 4
  Data, 6

  heuristic_models, 6

  markov_model, 7
  markov_model_mp, 8
  transition_matrix, 10