Purchase Modeling using Clickstream Data and Markov Chains

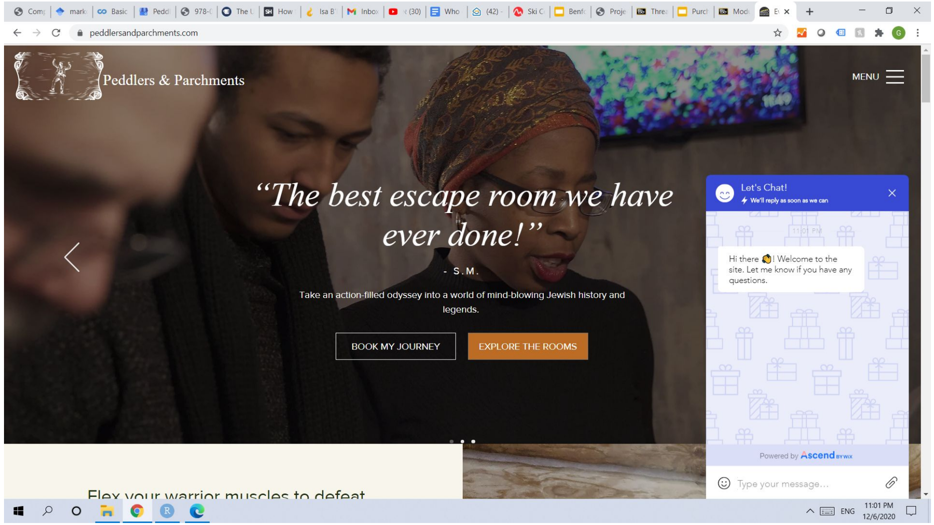
Gamliel Beyderman

Founder, Chief Data Scientist @ Peddlers&Parchments

Science is about solving real problems

One of the key questions for a business owner selling products online is, **How to recognize the web site browsers who will convert to customers?**

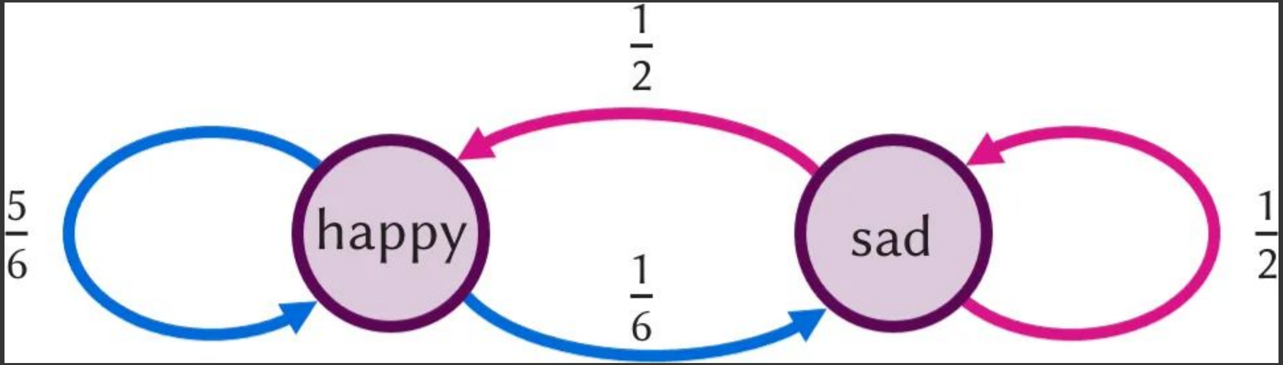
As the owner of an escape room in Brooklyn, I am no exception. In what will follow I will try to show how to use a technique we extensively discussed to try to answer this very question!



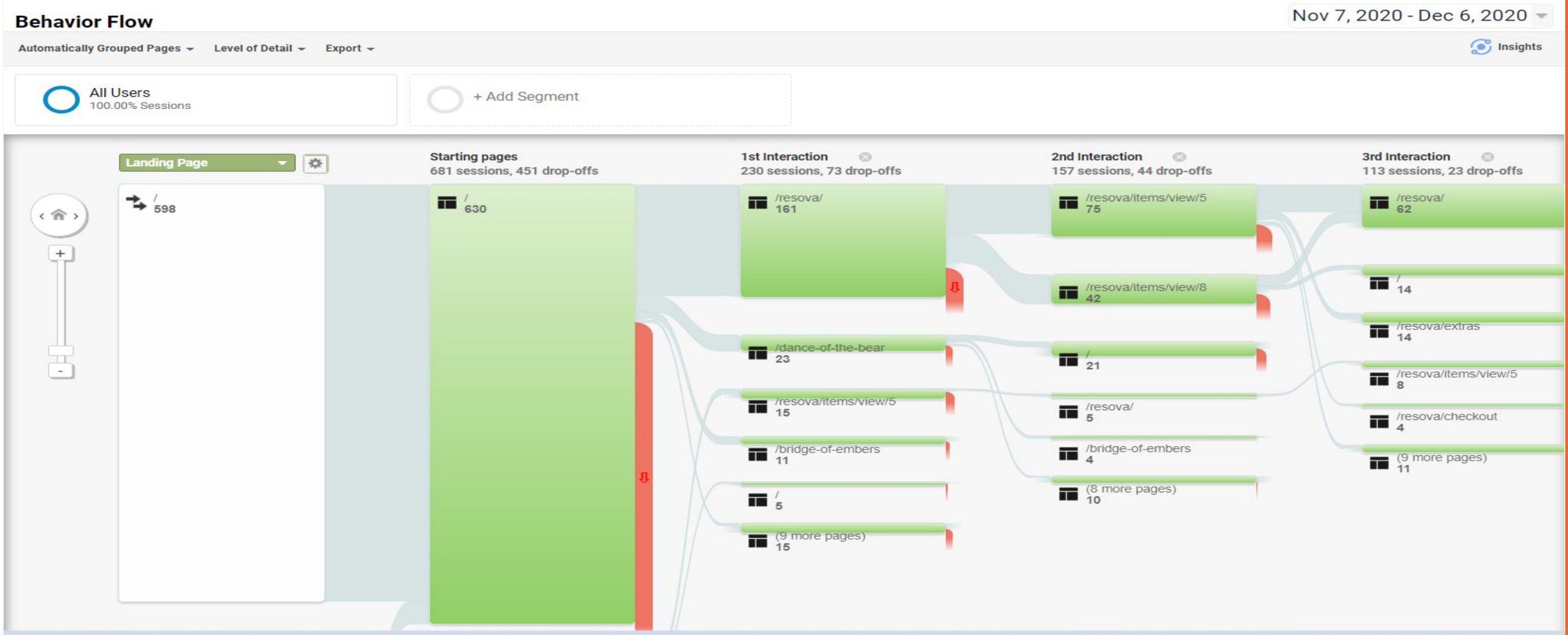
Imagine having a very nice day and feeling happy, and for some unknown reason, only possessing the emotional capacity to either be happy or sad.

Can you predict what your feelings will be tomorrow?

We know that our feelings tomorrow will be dependent on our feelings today; a function where the input is today's mood and only today's mood. Or in other words, a **Markov chain!**



Back to business! Usually, a web site user considering a product might either add the product to the shopping cart, view detailed product pages, scroll further down the main page. The probability for a transition to either of the possible next states depends on the mode (browsing, buying...) the user is currently in. This mode can be identified when considering the recent k states (pages) of a user rather than only the last state. **Higher-order** Markov chains are hence more promising when analyzing clickstream data.



The Markov property specifies that the probability of a state depends only on the probability of the **previous** state.

We can build more **"memory"** into our states by using a higher order Markov model.

An **nth order** Markov model:

$$P(x_i \mid x_{i-1}, x_{i-2}, \ldots, x_1) = P(x_i \mid x_{i-1}, \ldots, x_{i-n})$$

More Formally:

Raftery (1985) proposed a model for higher-order Markov chains that can be estimated with one additional parameter for each order k. His model is based on the idea that the distribution of state probabilities **X** can be approximated as weighted sum of the last k transition probabilities. **Q** is a m × m transition probability matrix and **λ_i** denotes the weight for each lag **i** in the model. :

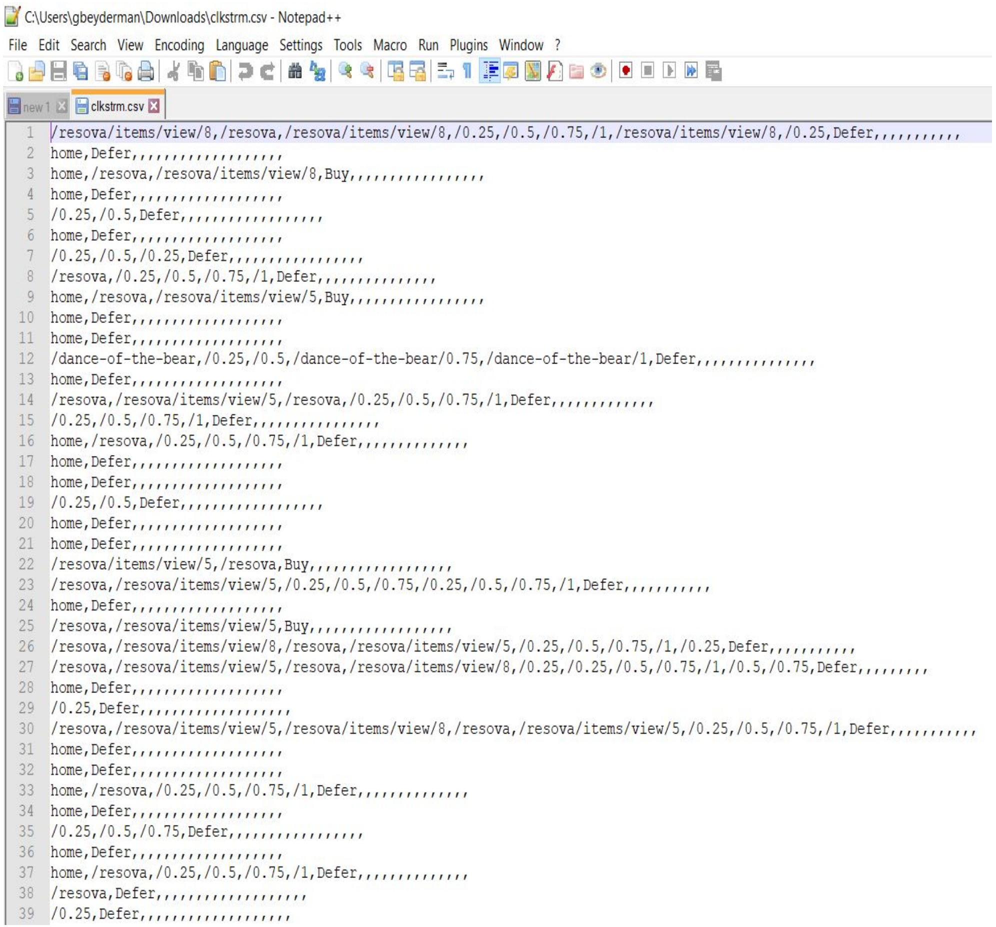
$$X^{(n+k+1)} = \sum_{i=1}^{k} \lambda_i \mathbf{Q} X^{(n+k+1-i)}$$

$$\sum_{i=1}^{k} \lambda_i = 1, \qquad \lambda_i \geq 0 \quad \forall i.$$

Solve Optimization Problem:

We can estimate Q_i by observing the transition probability from n – i to n. State probabilities are estimated from the sequence $X_{(n)}$. We are able to derive the following optimization problem to estimate the lag parameters λ. The optimization can be solved as either a linear problem or as a quadratic problem :

$$\min_{\lambda} \left\{ \left\| \sum_{i=1}^{k} \lambda_i \hat{\mathbf{Q}}_i \hat{X} - \hat{X} \right\| \right\}$$



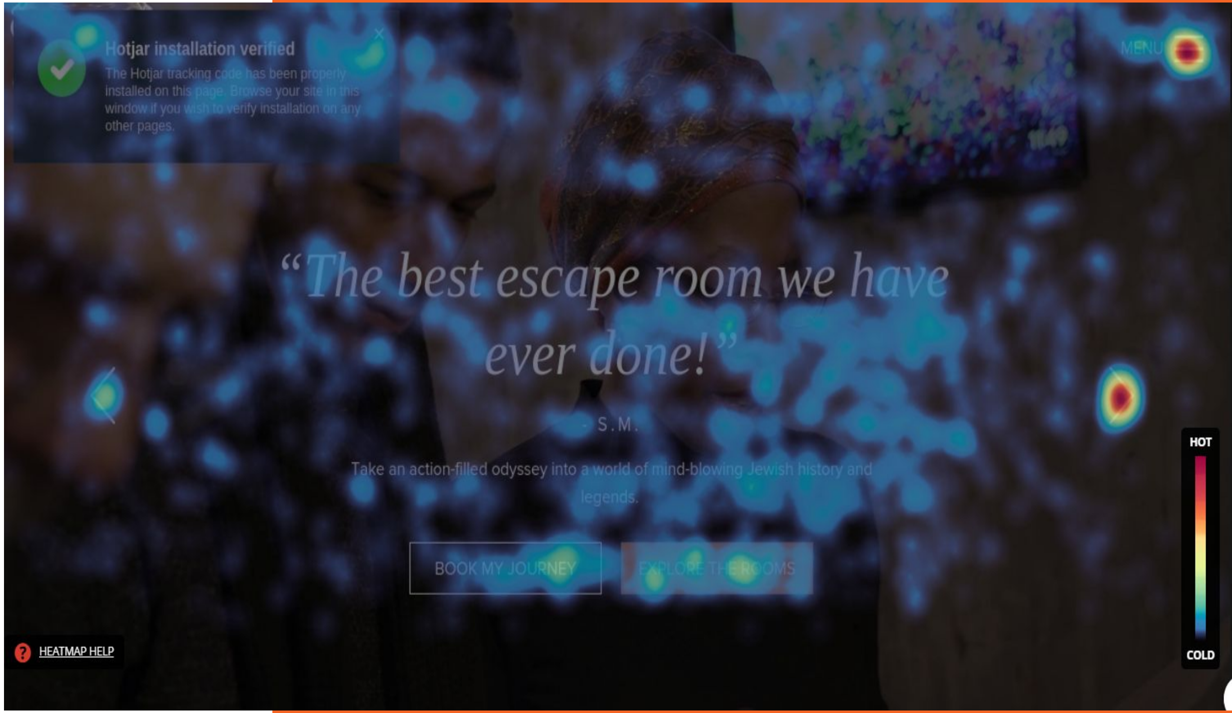
C:\Users\gbeyderman\Downloads\clkstrm.csv - Notepad++

```
/resova/items/view/8,/resova,/resova/items/view/8,/0.25,/0.5,/0.75,/1,/resova/items/view/8,/0.25,Defer,,,,,,,,,,
home,Defer,,,,,,,,,,,,,,,,,,
home,/resova,/resova/items/view/8,Buy,,,,,,,,,,,,,,,,
home,Defer,,,,,,,,,,,,,,,,,,
/0.25,/0.5,Defer,,,,,,,,,,,,,,,,,
home,Defer,,,,,,,,,,,,,,,,,,
/0.25,/0.5,/0.25,Defer,,,,,,,,,,,,,,,,
/resova,/0.25,/0.5,/0.75,/1,Defer,,,,,,,,,,,,,,
home,/resova,/resova/items/view/5,Buy,,,,,,,,,,,,,,,,
home,Defer,,,,,,,,,,,,,,,,,,
home,Defer,,,,,,,,,,,,,,,,,,
/dance-of-the-bear,/0.25,/0.5,/dance-of-the-bear/0.75,/dance-of-the-bear/1,Defer,,,,,,,,,,,,,,
home,Defer,,,,,,,,,,,,,,,,,,
/resova,/resova/items/view/5,/resova,/0.25,/0.5,/0.75,/1,Defer,,,,,,,,,,,,
/0.25,/0.5,/0.75,/1,Defer,,,,,,,,,,,,,,,
home,/resova,/0.25,/0.5,/0.75,/1,Defer,,,,,,,,,,,,,
home,Defer,,,,,,,,,,,,,,,,,,
home,Defer,,,,,,,,,,,,,,,,,,
/0.25,/0.5,Defer,,,,,,,,,,,,,,,,,
home,Defer,,,,,,,,,,,,,,,,,,
home,Defer,,,,,,,,,,,,,,,,,,
/resova/items/view/5,/resova,Buy,,,,,,,,,,,,,,,,,
/resova,/resova/items/view/5,/0.25,/0.5,/0.75,/0.25,/0.5,/0.75,/1,Defer,,,,,,,,,,
home,Defer,,,,,,,,,,,,,,,,,,
/resova,/resova/items/view/5,Buy,,,,,,,,,,,,,,,,
/resova,/resova/items/view/8,/resova,/resova/items/view/5,/0.25,/0.5,/0.75,/1,/0.25,Defer,,,,,,,,,,
/resova,/resova/items/view/5,/resova,/resova/items/view/8,/0.25,/0.25,/0.5,/0.75,/1,/0.5,/0.75,Defer,,,,,,,,
home,Defer,,,,,,,,,,,,,,,,,,
/0.25,Defer,,,,,,,,,,,,,,,,,,
/resova,/resova/items/view/5,/resova/items/view/8,/resova,/resova/items/view/5,/0.25,/0.5,/0.75,/1,Defer,,,,,,,,,,
home,Defer,,,,,,,,,,,,,,,,,,
home,Defer,,,,,,,,,,,,,,,,,,
home,/resova,/0.25,/0.5,/0.75,/1,Defer,,,,,,,,,,,,,
home,Defer,,,,,,,,,,,,,,,,,,
/0.25,/0.5,/0.75,Defer,,,,,,,,,,,,,,,,
home,Defer,,,,,,,,,,,,,,,,,,
home,/resova,/0.25,/0.5,/0.75,/1,Defer,,,,,,,,,,,,,
/resova,Defer,,,,,,,,,,,,,,,,,,
/0.25,Defer,,,,,,,,,,,,,,,,,,
```

A **clickstream** is a sequence of click events for exactly one web session. The clickstreams of different sessions typically differ in type and number of click events. Each click event is of type character. The clickstreams for a particular session can then be modeled as a vector, whereas a collection of clickstreams can be modeled as a list in R. The package **clickstream** provides an S3 class for storing lists of vectors of click events.

Simplified States for 7 days:

/home
25% page scroll
50% page scroll
75% page scroll
100% page scroll
2 Product Detail pages
2 Product Checkout pages
Purchase (Buy)
Leave without Buying (Defer)



7 Day Snapshot

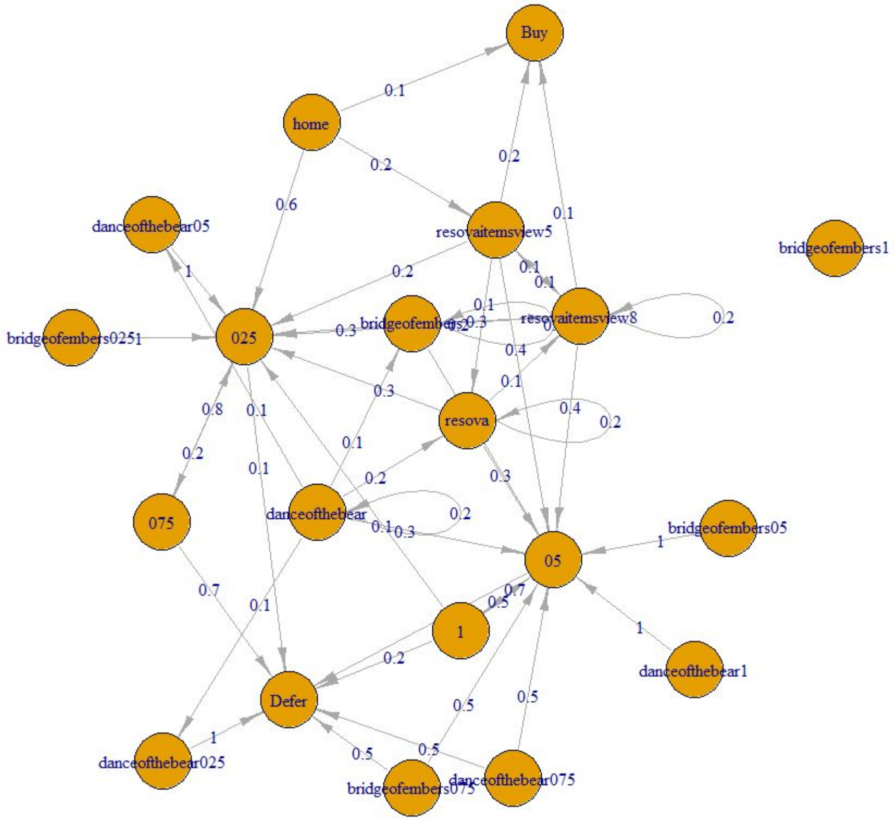
```
> clickstream::summary(cls)
Observations: 187

Click Frequencies:
              025                05               075                 1     bridgeofembers bridgeofembers025  bridgeofembers05
              138               114                96                74                  7                 1                 1
bridgeofembers075   bridgeofembers1               Buy   danceofthebear danceofthebear025   danceofthebear05 danceofthebear075
                2                 1                17                12                  1                 2                 2
  danceofthebear1             Defer              home            resova   resovaitemsview5   resovaitemsview8
                4               170                79                88                 36                29
```

- 79 visitors (42%, home) left the site without proceeding to scroll even 25% down the page (025 had 138 visits). We lost almost 30% of further visits between 75% scroll and the bottom (96 to 74).
- Very few chose to learn more about the 2 escape room products (bridge of embers, dance of the bear)
- resova (the booking page) registered 88 hits (47% of all the 187 observations)
- Individual conversion pages (resovaitemview5 and resovaitemview8) registered 36 and 29 visits
- 17 Buy visits were actually the visitors that started the Checkout process.
- 170 chose not to Buy - hence we call them 'Defer'

Fitting a Markov Chain to the clickstream data in R produces a Transitional Probabilities Matrix:

```
Lag:  2
lambda:  0.9999999
                          025          05        075          1 Buy Defer bridgeofembers bridgeofembers025 bridgeofembers05
025                0.033613445 0.067961165 0.17073171 0.09090909   0     0      0.2857143                 1                0
05                 0.033613445 0.029126214 0.08536585 0.50000000   0     0      0.2857143                 0                1
075                0.781512605 0.019417476 0.00000000 0.04545455   0     0      0.0000000                 0                0
1                  0.016806723 0.699029126 0.00000000 0.00000000   0     0      0.0000000                 0                0
Buy                0.000000000 0.000000000 0.00000000 0.04545455   0     0      0.0000000                 0                0
Defer              0.092436975 0.126213592 0.65853659 0.22727273   0     0      0.0000000                 0                0
bridgeofembers     0.000000000 0.009708738 0.00000000 0.00000000   0     0      0.1428571                 0                0
bridgeofembers025  0.000000000 0.000000000 0.00000000 0.00000000   0     0      0.0000000                 0                0
bridgeofembers05   0.000000000 0.000000000 0.00000000 0.00000000   0     0      0.0000000                 0                0
bridgeofembers075  0.016806723 0.000000000 0.00000000 0.00000000   0     0      0.0000000                 0                0
bridgeofembers1    0.000000000 0.009708738 0.00000000 0.00000000   0     0      0.0000000                 0                0
danceofthebear     0.000000000 0.000000000 0.00000000 0.00000000   0     0      0.0000000                 0                0
danceofthebear025  0.000000000 0.000000000 0.00000000 0.00000000   0     0      0.0000000                 0                0
danceofthebear05   0.008403361 0.000000000 0.00000000 0.00000000   0     0      0.0000000                 0                0
danceofthebear075  0.016806723 0.000000000 0.00000000 0.00000000   0     0      0.0000000                 0                0
danceofthebear1    0.000000000 0.038834951 0.00000000 0.00000000   0     0      0.0000000                 0                0
home               0.000000000 0.000000000 0.01219512 0.00000000   0     0      0.0000000                 0                0
resova             0.000000000 0.000000000 0.04878049 0.04545455   0     0      0.0000000                 0                0
resovaitemsview5   0.000000000 0.000000000 0.00000000 0.04545455   0     0      0.0000000                 0                0
resovaitemsview8   0.000000000 0.000000000 0.02439024 0.00000000   0     0      0.2857143                 0                0
                   bridgeofembers075 bridgeofembers1 danceofthebear danceofthebear025 danceofthebear05 danceofthebear075
025                              0.0               0            0.0                 0                1               0.0
05                               0.5               0            0.3                 0                0               0.5
075                              0.0               0            0.0                 0                0               0.0
1                                0.0               0            0.0                 0                0               0.0
Buy                              0.0               0            0.0                 0                0               0.0
Defer                            0.5               0            0.0                 1                0               0.5
```

Visualizing the Transition Matrix



Goodness of Fit

fitMarkovChain() computes the **log-likelihood** of a 'MarkovChain' object based on the m×m transition frequency matrices F_i :

$$LL = \sum_{i=1}^{k} \lambda_i \mathbf{F}_i \log\left(\frac{\mathbf{F}_i}{1_s \mathbf{F}_i}\right)$$

Slightly higher LL for order= 2, suggests a better model. Yet, AIC there is higher - it should be lower if order=2 is truly a better fit!

```
> mc <- clickstream::fitMarkovChain(clickstreamList = cls, order = 1, control = list(optimizer = "quadratic"))
> clickstream::summary(mc)
First-Order Markov Chain with 20 states.
 The Markov Chain has absorbing states.

Observations: 874
LogLikelihood: -607.8864
AIC: 1257.773
BIC: 1358.008
> mc <- clickstream::fitMarkovChain(clickstreamList = cls, order = 2, control = list(optimizer = "quadratic"))
Warning message:
In clickstream::fitMarkovChain(clickstreamList = cls, order = 2,  :
  Some click streams are shorter than 2.
> clickstream::summary(mc)
Higher-Order Markov Chain (order=2) with 20 states.
 The Markov Chain has absorbing states.

Observations: 874
LogLikelihood: -595.9958
AIC: 1275.992
BIC: 1476.461
```

Using the Model For Prediction

We are interested in those clicks just before a final decision (buy or defer). Each clickstream hence has an **absorbing state** which is either "Buy" or "Defer". If we know the probability B that our clickstreams will be absorbed in any of the possible absorbing states, we can use this information to more accurately predict the next click.

$$X^{(n)} = B \sum_{i=1}^{k} \lambda_i Q_i X^{(n-i)}$$

Using the Model For Prediction

A ‘Pattern’ object is a (part) of a clickstream described by a sequence of clicks and optionally a probability of occurrence and a vector of absorbing probabilities. To predict the next click of a given ‘Pattern’ object based on a given MarkovChain-object, we use the predict() function as follows:

If a user starts with the clickstream 025 and 05, the user will click on 075 next with 78.15% probability:

```
> pattern <- new("Pattern", sequence = c("025","05"))
> resultPattern <- clickstream::predict(mc, startPattern = pattern, dist = 1
> resultPattern
Sequence: 075
Probability: 0.7815126
Absorbing Probabilities:
  None
1  NaN
```

If a user starts with the clickstream “resova” - the booking page, we can predict the next two clicks: resovaitemsview5 and 05.

```
> pattern <- new("Pattern", sequence = c("resova"))
> resultPattern <- clickstream::predict(mc, startPattern = pattern, dist = 2)
> resultPattern
Sequence: resovaitemsview5 05
```

If a user starts with the clickstream resova and resovaitemview5, and has a probability of 20% to convert we can predict the next click: **The user has 36.74% probability to go back to the middle of the home page (50% scroll) which will lower his probability to buy down to 5.23%:**

```
> pattern <- new("Pattern", sequence = c("resova","resovaitemsview5"),
+                absorbingProbabilities = data.frame(Buy = 0.2, Defer = 0.8))
> resultPattern <- clickstream::predict(mc, startPattern = pattern, dist = 1)
> resultPattern
Sequence: 05
Probability: 0.3677409
Absorbing Probabilities:
        Buy     Defer
1 0.0522832 0.9477168
```

Reference

Scholz. “R Package clickstream: Analyzing Clickstream Data with Markov Chains.” (2016). Journal of Statistical Software. October 2016, Volume 74, Issue 4.

Thank You For Watching and Taking the Class with Me! Much blessing to you and yours!