Bayesian Computing

Ashutosh Singh

June 2021

1 Introduction

Vector quantization or clustering is a very important topic in Machine Learning. Unsupervised clustering techniques are use to group people with similar symptoms, group similar queries on a forum or in the context of this report grouping users based on their behaviour on a website.

Goal is to utilize Intrinsic Dimension based methodology HIDALGO to find clusters of users. To this end we chose to use the intRinsic R-package which provides very good set of of APIs and methods to run HIDALGO and analyze its results

2 Dataset

The dataset consists of events of each user along with timestamp of the event. Dataset sample

event_date	event_timestamp	event_name	user_id
20210425	1.619378e+15	login	APP-055rKogJet...
20210425	1.619378e+15	user_engagement	APP-055rKog...

Table 1: A sample of original dataset. Complete dataset has 2699 observations like the two shown above across 4 variables

2.0.1 Events

Events are the manifestation of user's browsing behaviour on the website. There a total of 25 unique events.

2.0.2 Users

Users of the website. There are a total of 136 users in the dataset. Out of which 108 have unique behaviour.

3 Segmentation with Hidalgo

3.1 Transforming the dataset

This structure of dataset is not suitable for clustering users. Performing the following transformation steps -

1. Represent each user as a 25-D vector
2. Each dimension in the vector represents a unique event.
3. The value of the event-index for user vector is the number of times user triggered this event
4. For all the events a user doesn't trigger that user never visits the value is 0(zero)

After following the above steps the dataset looks like -

	login	user_engagement	screen_view	hamburger_menu_clicked	menu_item_clicked	video_play	video_closed
	APP-055rKogJ...	1	2	1	1	0	0
	APP-0Tx2BgVd...	1	5	1	1	3	2
	APP-0iglq3QDI...	1	2	1	1	0	1

Figure 1: Transformed dataset. 136x25 matrix. 136 users and 25 unique events. Here only 6 events are shown.

3.2 Running Hidalgo

Running the Hidalgo algorithm from intRinsic R-package on the transformed data following results are observed.

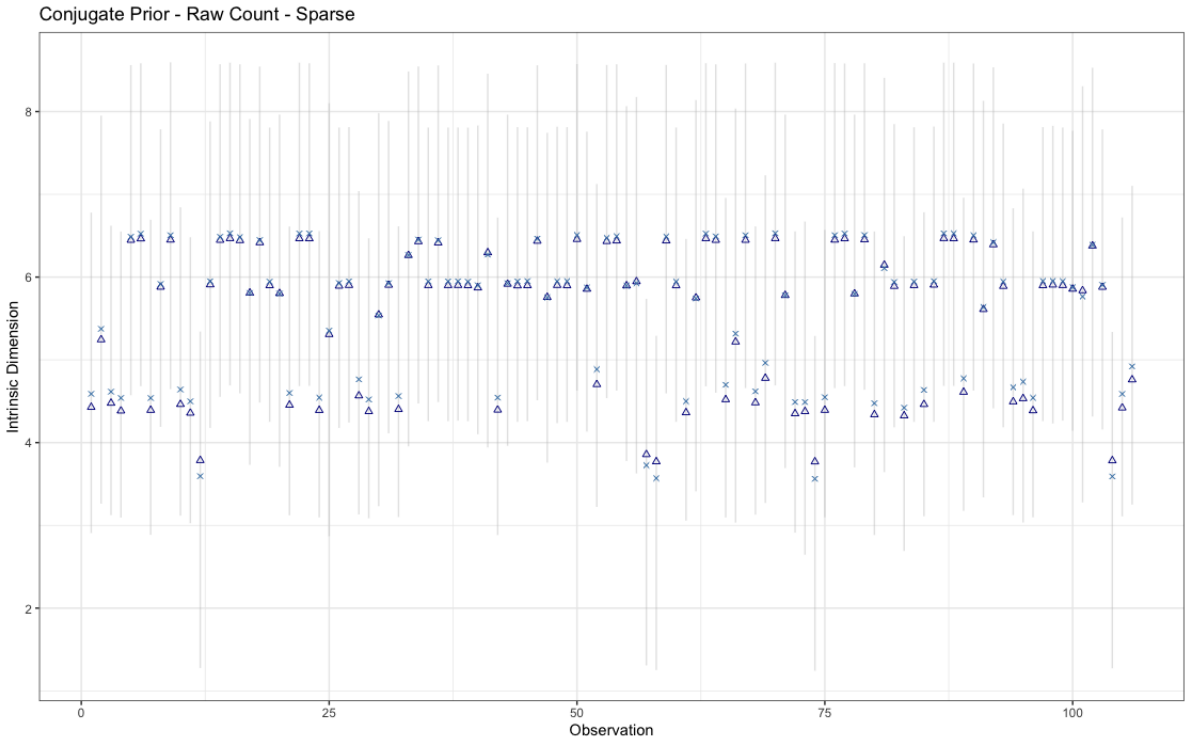

Figure 2: Plot of users(X-axis) and ID(Y-axis) with a Conjugate Prior. Burn-in=2500 and nSim=5000. Triangles are median values and crosses are mean values of ID. The grey vertical lines represent a 90% credible interval

3.3 Inducing state dependence

The clusters look good and the results from above mentioned transformation and Hidalgo are quite impressive.

But there are certain problems with the above approach. It doesn't take user's previous state into account when considering his score for an event.

3.3.1 Checking for Markov Property

The above raw-scores just measure how many times a user triggered the event.

So with this in mind I tried to look for Markov Property. These are the steps I followed:

1. Take the entire event_name column from the original dataset i.e. a column vector of size 2699.

2. Used "verifyMarkovProperty" method of the "markovchain" package to see the results

3. Results: Chi - square statistic is: 1571.218
   Degrees of freedom are: 224
   And corresponding p-value is: 0

4. As we can see the p-value is really small. Hence we can reject H_0 that Markov Property exists

This approach seemed to be the best approach. But after getting these results, I realized the mistake that was made. Each user accesses the markov chain independently. So taking the entire event column is not the right way

3.3.2 Making a transition matrix

The entire event column should not be used to verify Markov Property.

To create a transition matrix for transitions between events the following steps were taken:

1. Use chain of events visited by each user to make a transition count matrix
2. The chain ends with each user and is restarted for the next user.
3. Code iterates for each user from the 2nd event he triggered till the last event.
4. Divide each element in row by the sum of all elements in that row to get transition probabilities.

3.3.3 Making a Markov Chain

Now that the transition matrix is ready I use the markovchain package to create our Markov chain.

And this time I got a favourable result, the Markov Chain is Irreducible and Aperiodic and hence there is unique limiting distribution.

I store this transition matrix and limiting distribution and put them to use in the subsequent sections.

3.3.4 Transforming the data

With the transition matrix I obtained I modified the Data transformation steps mentioned in Section 3.1 to induce state dependence.

Now To represent each user as a 25-D vector, I follow the steps mentioned below -

1. Get the sequence of events for the user.
2. Give the first event of each user a score of 1
3. Iterate over the sequence of steps for the user from 2nd event and use transition_matrix[i-1, i] as the score for i-th event.

4. Every event the user doesn't visit he gets a 0(zero) score.

This method improves upon the previous data transformation process by inducing a state dependence. For example a user triggering event-2 after event-1 would be an exact copy of user triggering event-1 after event-2; if we go with the transformation steps in section 3.1.

But here this is not true. We get different vectors for bot of these hypothetical users

I think it's a very nice property

3.3.5 Running Hidalgo with new data

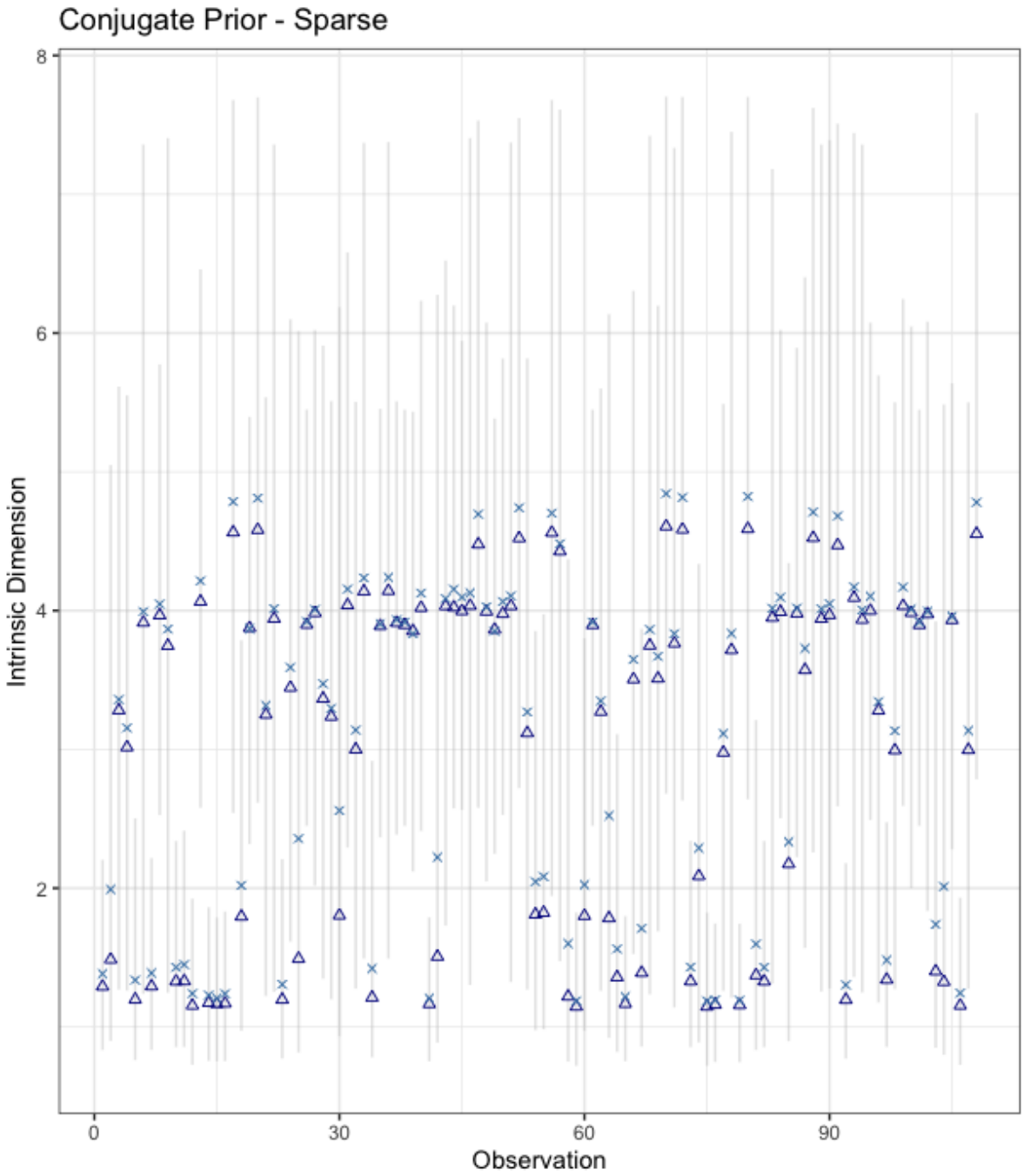

Figure 3: Plot of users(X-axis) and ID(Y-axis) with a Conjugate Prior. Burn-in=2500 and nSim=5000. Triangles are median values and crosses are mean values of ID. The grey vertical lines represent a 90% credible interval

Even without a manual check we can see that the credible intervals are more disconnected for observations in different ID levels.

3.3.6 One final trial

The above data transformation brings the previous state into play when giving the score for current state; but it still leaves the unvisited events with a zero score.

Since we already have a stationary distribution I decided to replace 0 for an event in a user vector with its limiting distribution probability.

And ran this modified data through Hidalgo. The previous data matrix was sparse but new modified matrix is dense hence the title of the chart.

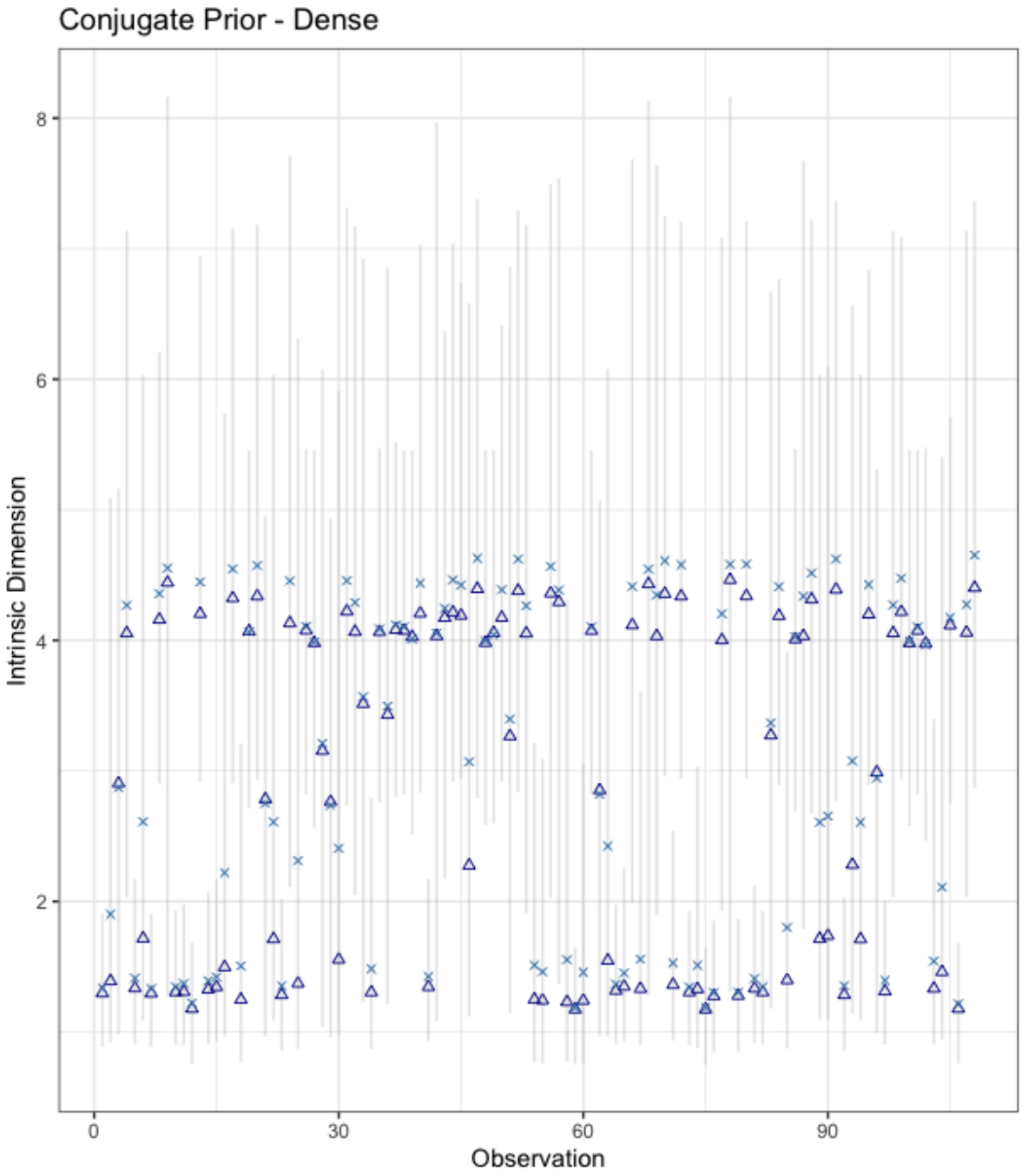

Figure 4: Plot of users(X-axis) and ID(Y-axis) with a Conjugate Prior. Burn-in=2500 and nSim=5000. Triangles are median values and crosses are mean values of ID. The grey vertical lines represent a 90% credible interval. These results are for Dense Matrix we just obtained

4 Analysis of Results

Now we get to the part where we make comparisions on all Hidalgo outputs.

1. Raw-Count: Using this term to represent the first run with results in section 3.2
2. Sparse-Transition: For the run where use transition probabilities as scores. Section 3.3.1 - 3.3.5
3. Dense-Transition: For the third last trial . Section 3.3.6

4.1 Visual observations

As we can see from Figure-2, Figure-3 and Figure 4 for Raw-Count, Sparse-Transition and Dense-Transition, the credible interval lines(the vertical grey lines) become more visibly separate from each other as we go from Raw-Count to Dense-Transition

This implies that model is becoming more confident in segregating the users.

4.2 Comparing Optimal Clusters

I will now only compare Dense-Transition and Sparse-Transition.

I use Hidalgo_coclustering_matrix method from R-package. The results also confirm what we saw visually.

I most cases both these iterations give same result. The results for level 3 are exactly same for both methods.

However these methods predict don't agree with each other in the following cases

	users	dense_cluster_id	sparse_cluster_id
6	APP-20w0TLO08iBLExZSrHQM0UEcVlVQdd2ca9be3c5...	1	2
22	APP-Amyo1Z6vLFyyoAZTefYZiH1cmUhv5707925ba21...	1	2
33	APP-FEaXrRlD7r9K7tDPZ6CeaqlcoNuTcde9c1e8c975e...	1	2
36	APP-HoDWrqe4H8flsGwSxsn8RV4rOcHMa933d17afd0...	1	2
42	APP-JqIG3xQRHwepGrT3QxZ2djpvBZJcfbae6aac18ff0...	2	1
46	APP-M2qSJaBDBSDykH9OP8qx8Sx2jn8Q23d33bb0cd...	1	2
51	APP-O11OrPxodT9q9byfvOneCbP8Nwu9fef906478b7...	1	2
71	APP-ZjV7KlbsiKtCn7eVstAzWlBWKvZo35ba6a383a028...	1	2
74	APP-cphtpaXEjuvmmrwp4qdllCZnVwjNf938185a17fa...	1	2
83	APP-m7TfjfUupg6OlWpOD0ODtSIKIY0Maafc587c262e...	1	2
85	APP-nv1arXszWYnQCzyuWZD4QjCPv6qE607e51212a...	1	2
89	APP-piuAutWXYN8HQuXtKnsDqJb17gZO55796b05fb4...	1	2
90	APP-puYRgstF4dLHyNgEgjit6k9hSnKue341ce16ea7d5...	1	2
93	APP-qR6fq7pnvTCLhIzRdXGLZMyzrx8592c8c413cf5a...	1	2
94	APP-qezk32hW0NWNpTj9ifQVUTG37xgjb805137e97...	1	2

Figure 5: Users for which we get different results from Dense-Transition and Sparse-Transition

4.3 Comparing Discriminatory Power

To support the visual observation that Dense-Transition is better at discriminating the clusters as visible from 90% credible intervals.

For this purpose we need to inspect the posterior co-clustering matrix.

A method that takes in each user which is classified under different clusters and uses co-clustering matrices from Dense-Transition and Sparse-Transition to get the posterior scores of user with other users in same cluster and users in other clusters. Compute mean of absolute value of diff between above two scores.

The method for which this mean is higher should indicate higher confidence.

Out of the 15 incorrectly classified users we get 8 users having these mean values higher for Dense-Transition than for Sparse-Transition

But this is not a very dependable method or accurate method. Since the overall scale of both runs i.e. Dense-Transition and Sparse-Transition might be different.

I leave finetuning this method for future work.

4.4 Manual analysis

To further check the performance of Dense-Transition and Sparse-Transition, I did a manual check on the cluster they have the most different results for - Cluster-2. Adding the results in cluster_2_users_dense.csv and cluster_2_users_sparse.csv

5 Future Work

5.1 Using different Distance metric

We can try using a different distance metric than Hidalgo uses right now. Instead of Euclidean Distance we can try using Cosine-Similarity and see the results. Instead of spatial-distance Cosine-Similarity focusses on orientation.

5.2 Induce a time dependence

In addition to having dependence on previous event we can also think towards adding a time dependence.

Since the time between two events is not scaled we will need to handle that as well. We can look into recently introduced time-embedding methods for that.

5.3 Sophisticated Discriminatory Power measures

Improve the method in section 4.3 or develop new methods to measure this.

5.4 Human annotation

I did a very basic experiment with cluster 2 as mentioned in section 4.4. But to get a more clear idea on algorithm's behaviour we need to do more human annotation

6 Results and Files

6.1 Code

1. pr.R - File with Dense-Transition and Sparse-Transition Methods code
2. raw_count.R - File with Raw-Count method code

6.2 Data

1. results-20210426-141638.csv - Original data file
2. user_rows.R - Data file with users as rows and their events in columns. Each row has different lengths.

6.3 Results

All the files related to this section are in ‘results‘ directory

1. cluster_2_users_dense.csv - Results of manual analysis of cluster 2
2. cluster_2_users_sparse.csv - - Results of manual analysis of cluster 2 Sparse-Transition
3. RawCount-ConjugatePrior-Sparse.png - Hidalgo results Raw-Count
4. DenseTransition-ConjugatePrior.png - Hidalgo results Dense-Transition
5. SparseTransition-ConjugatePrior.png - Hidalgo results Sparse-Transition