Master Thesis

Topic:

Unsupervised learning in decision making

Author:	Domagoj Fizulic
	Felix Gutmann
Student number:	125604
	125584
Program:	M.S. Data Science
E-Mail:	domagoj.fizulic@barcelonagse.eu
	felix.gutmann@barcelonagse.eu

I Table of Contents

II List of Figures

III List of Tables

IV List of Listings

V List of mathematical symbols

Symbol	Meaning
a_t	Action at time t
$Q(a)_t$	Value function at time t
ϵ	Probability of exploration in epsilon greedy
α	Learning rate
τ	Softmax parameter
$H(X)$	Entropy of a discrete random variable X
$\mathbb{R}_0^+$	Positve real numbers including zero

VI List of abbreviations

Abbreviations	Description

1 Introduction and conceptual approach

Learning is a complex procedure. The learning procedure can affected due to social conditions...

2 Relevant Literature

3 Theoretical Backround and experiments

In this section we provide ...

3.1 Experiment design

Before analysing real data we run some simmulation experiments to indentify adequat approaches to model our data. To generate data artificially we follow a reinforcement learning and multi arm bandit approach where we let an agent learn the distribution of a simulated data set. The procedure is illustrated in figure 1. To keep things simple we draw a set of rewards from a normal distribution. We let several agents with different parameter setting process the reward data. From this procedure we obtain for each agent a time series of choices.

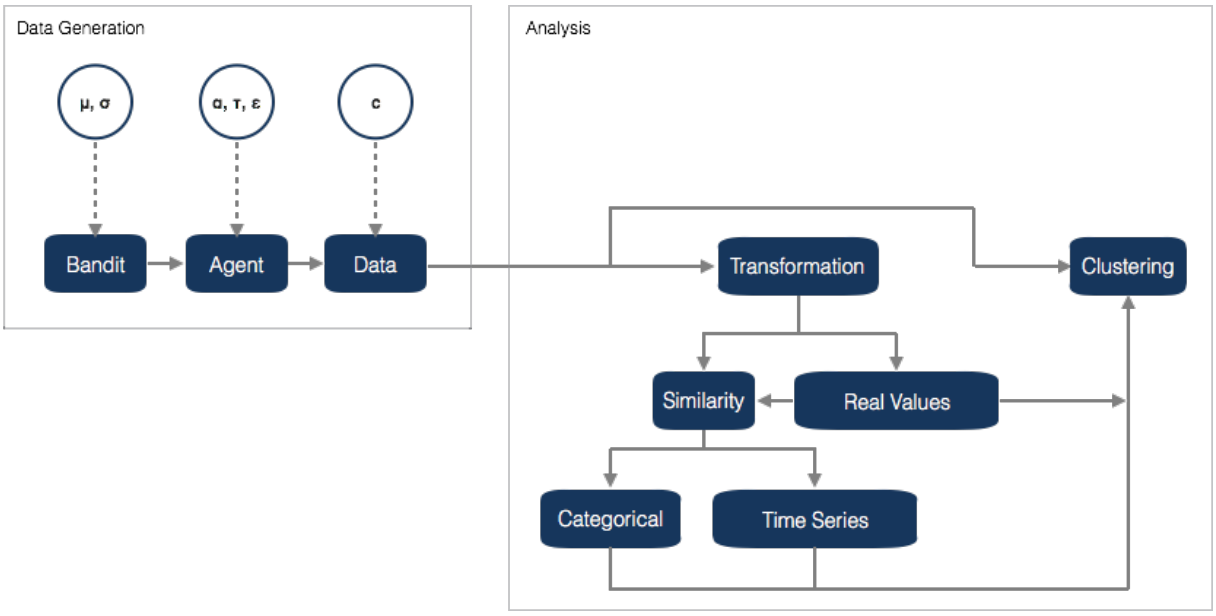


Figure 1: Flowchart experiment desgin

Concerning our data we find two main challanges. First, our data have a categorical nature. Furhtermore, the learning process also imposes a time series dependence on the data. We address this issue in the following ways. The first approach is to map the series of choices to a real valued series. For each time step we can compute *Shannon's Entropy* based on the empirical probability of the choices. Let X be a discrete random variable with probability p, then the entropy is defined as:

$$H(X) := -\sum_{i=1}^{N} p_i \log_2 p_i \tag{1}$$

Entropy gives measure on how random a random variable behaves. Therefore, transforming choices to sequentiel entropy is to discriminate individuals by the randomness

3.2 Reinforcement Learning backround and multi arm bandits

Reinforcement Learning (RIL) is a branch of *Machine Learning* try model how an artificial agents interact with its environment and learns from the process over time.
In particular an agents is confronted with the task of choosing sequently from a set of choices. In comparison to *supervised learning*, where an agent is learning based on set of examples an agent in RIL doesn't have any knowledge about the system apriori. Therefore, it has to learn the nature of the system by sequentially interacting with its environment and keeping tack of the obtained information. Since the agent doesn't have any apriori information about the system it has to explore new possible action and so has to deviate from the optimal action. Furthermore, it has to keep track of value of each action he did so far. So the main task of the agent is to balance exploration and explotation. There are to basic approaches to model this trade-off; An *"Epsilon-Greedy"* selection method and Soft *"Softmax"* selection method. Before explaining both cocepts we introduce the value function for a given action a. Therefore, let $Q_t(a)$ be the value function of action defined as:

$$Q_t(a) := \frac{R_1 + R_2 + \cdots + R_{K_\alpha}}{K_\alpha} \tag{2}$$

The value function is the average over rewards.

Considering now epsilon greedy action selection mehtod: The rule in general is to select the next action as the current current highest value function. However, to model exploration we introduce a random element to deviate from that greedy strategy. Following that the next action

$$a_{t+1} = \begin{cases} \text{random action} & \text{, with probability } \epsilon \\ \arg\max_i Q_t(i) & \text{, with probability } 1-\epsilon \end{cases} \tag{3}$$

where $\epsilon \in [0, 1]$ is a parameter controling the random behaviour of the agent.

In the softmax action selection method compute for each action a probability (also called *Boltzmann Distribution).* The probability for action a is computed by:

$$P(a_t|X) = \frac{e^{\frac{Q_t(a)}{\tau}}}{\sum_i^K e^{\frac{Q_t(i)}{\tau}}} \tag{4}$$

In each iteration the next action of the agent is drawn with probability p_a:

$$a_{t+1} \sim p_a \tag{5}$$

After selecting an action the agent is updating its believe of the chosen action. Formally the update rule is defined

$$Q(a)_{k+1} = Q(a)_k + \alpha \left[R(a)_k - Q(a)_k \right], \tag{6}$$

where α is a is the non negativ *learning rate* defining how much the current action is affecting the believes.

3.3 Unsupervised Learning

We consider several unsupervised clustering techniques. A technical description for all of the is provided in

Algorithm	Input	Datatype
Spectral Clustering	Similarity Matrix	-
Affinity Propagation	Similarity Matrix	-
K-Means Clustering	Data Matrix	-
Ward Clustering	Data Matrix	-
PCA + Ward Clustering	Data Matrix	-

Table 1: Overview clustering algorithms

3.4 Simmulation results

The following table shows a snippet of our simmulation results.

Specification	μ	σ	CL Size	SD	Decision	α	τ	N	ALG	TRNS	MI	NMI	AMI	CS	HS	VMS
1	-	-	-	-	-	-	-	-	-	-	-	-	-	-		

Table 2: Overview Outcome

4 Data Analysis

4.1 Data Description

4.2 Results

5 Conclusion

6 List of Literature

Appendix

Metrics and Similarities

This part of the appendix formally defines metrics and similarities and dissimilarities (proximity) used in this paper. We first define some basic general concepts followed by a description of the applied distance and similarity concepts.

Distances vs. Similarities

Let $\boldsymbol{X}$ be a dataset and let $\boldsymbol{x_i}, \boldsymbol{x_j}$ be two datapoints, such that $\boldsymbol{x_i}, \boldsymbol{x_j} \in \boldsymbol{X}$.

A distance function assign for pairs a points a non negative real number as distance. $d : \boldsymbol{X} \times \boldsymbol{X} \mapsto \mathbb{R}_0^+$. Formally if the following properties are additionally staisfied the distance is also metric.

1. $d(\boldsymbol{x_i}, \boldsymbol{x_j}) \geq 0$
2. $d(\boldsymbol{x_i}, \boldsymbol{x_i}) \geq 0$
3. $d(\boldsymbol{x_i}, \boldsymbol{x_j}) = d(\boldsymbol{x_j}, \boldsymbol{x_i})$
4. $d(\boldsymbol{x_i}, \boldsymbol{x_j}) \leq d(\boldsymbol{x_i}, \boldsymbol{x_j}) + d(\boldsymbol{x_j}, \boldsymbol{x_i})$

A distance can be seen as a measure for disimilarity of two points. Besides distance some algorithms operate on a *similarity* matrix. Formally a similarity is a function $S : \boldsymbol{X} \times \boldsymbol{X} \mapsto [0, 1]$. Also for similarity we can define the following properties:

1. $0 \leq S(\boldsymbol{x_i}, \boldsymbol{x_j}) \leq 1, \text{for } i \neq j$
2. $S(\boldsymbol{x_i}, \boldsymbol{x_j}) = 1$
3. $S(\boldsymbol{x_i}, \boldsymbol{x_j}) = S(\boldsymbol{x_j}, \boldsymbol{x_i})$

Once we have computed distance or a similarity we can compute for two data points we can use this information to transform it to a similarity or the distance vice versa:

$$S(\boldsymbol{x_i}, \boldsymbol{x_j}) = \frac{1}{1 + d(\boldsymbol{x_i}, \boldsymbol{x_j})} \quad \Leftrightarrow \quad d(\boldsymbol{x_i}, \boldsymbol{x_j}) = \frac{1}{S(\boldsymbol{x_i}, \boldsymbol{x_j})} - 1 \tag{7}$$

.0.1 Similarity measures for timeseries data

We use three different similarity measures for time series. Empirical research suggest that simple euclidian distance for time series performs quite well and is hard to beat. Hence, the first distance measure for time series is simply eucldian distance between two time series. They might be converted to similarity based on equation 7. Let $\boldsymbol{x}, \boldsymbol{y}$ be two time series over N-periods. Then the distance

$$d_E(\boldsymbol{x}, \boldsymbol{y}) := \sqrt{\left(\sum_{i=1}^{N} (x_i - y_i)^2\right)} \tag{8}$$

.0.2 Similarity measures for categorical data

Reffering to equation 8 we define a simple measure for the categorical aspect of the data. Note that [SOURCE] defines the simplest measure for categorical data. Howevere, since the probability that to people behave exaclty the same in our context is arguably zero we modify this concept slightly. So we relax that and define the overlap similarity just as the count of overlapping instances. This serves as a benchmark similarity for categorical data.

In [Boriah 2008] we find a rich class of further categorical measures. To keep it conceise we considered two of them. The first one is *Eskin* similarity measures.

$$d_O(\boldsymbol{x}, \boldsymbol{y}) := \begin{cases} 1 & ,\text{if } \boldsymbol{x} = \boldsymbol{y} \\ \frac{n_k^2}{n_k^2+2} & \text{otherwise} \end{cases} \tag{9}$$

Furthermore, we consider lins similarity

$$d_{lin}(\boldsymbol{x}, \boldsymbol{y}) := \begin{cases} 2\log \hat{p}_k(X_k) & ,\text{if } \boldsymbol{x} = \boldsymbol{y} \\ 2\log\left(\hat{p}_k(x_k) + \hat{p}_k(y_k)\right) & \text{otherwise} \end{cases} \tag{10}$$

Clustering Evaluation

In this section we formally derive and explain the applied clustering metrics. Evaluating clustering performance has some issues. The algorithm is producing labels. Nevertheless we generated the "true" labels the might not be comparable. A simple example we might consider the following situation. Let y denote the labels of data the data and y' the corresponding prediction such that $y, y' \in \{0, 1\}$. In a small example let our data points be like $y = (1, 1, 0, 0)$ and the corresponding prediction $y' = (0, 0, 1, 1)$. Obviously the clustering worked perfectly, however comparing "labels" would produce an accruacy of zero.

There a several clustering metrics, which respect such a situation. We consider a bunch of information based metrics. Most of the measures use some sort of entropy. Therefore, we stick to the notation of [SOURCE].

Hence we introduce the contigency table.

N_{11}**:** Number of pairs in the same cluster

	V_1	V_2	...	V_c	Σ
U_1	$n_{1,1}$	$n_{1,2}$	...		a_1
U_2	$n_{2,1}$	$\ddots$			a_1
$\vdots$	$n_{1,1}$	...			a_1
U_R	$n_{1,1}$	...			a_1
	b_1	b_2	...	b_c	N

Table 3: Contigency Table

N_{00}**:** Number of pairs that are in different clusters in both v and u

N_{01}**:** Number of pairs that are in the same cluster in both u but different in v

N_{10}**:** Number of pairs that are in the same cluster in both v but different in u

$$RI(u,v) = \frac{N_{00} + N_{11}}{\binom{N}{2}} \tag{11}$$

$$ARI(u,v) = \frac{2\left(N_{00}N_{11} - N_{01}N_{10}\right)}{\left(N_{00} + N_{01}\right)\left(N_{01} + N_{11}\right) + \left(N_{00} + N_{10}\right)\left(N_{10} + N_{11}\right)} \tag{12}$$

$$H(u) = -\sum_{i=1}^{R} \frac{a_i}{N} \log \frac{a_i}{N} \tag{13}$$

$$H(v) = -\sum_{i=1}^{C} \frac{b_i}{N} \log \frac{a_i}{N} \tag{14}$$

$$H(u,v) = -\sum_{i=1}^{R}\sum_{j=1}^{C} \frac{n_{i,j}}{N} \log \frac{n_{i,j}}{N} \tag{15}$$

$$H(u|v) = -\sum_{i=1}^{R}\sum_{j=1}^{C} \frac{n_{i,j}}{N} \log \frac{n_{i,j}/N}{b_j/N} \tag{16}$$

$$H(v|u) = -\sum_{i=1}^{C}\sum_{j=1}^{R} \frac{n_{i,j}}{N} \log \frac{n_{i,j}/N}{b_j/N} \tag{17}$$

$$I(u,v) = \sum_{i=1}^{R}\sum_{j=1}^{C} \frac{n_{i,j}}{N} \log \frac{n_{i,j}/N}{a_i b_j/N} \tag{18}$$

$$\tag{19}$$

Normalized Info Score:

This is one example of a normalized version

$$NMI_{max}(u,v) = \frac{I(u,v)}{\max\left(H(u), H(v)\right)} \tag{20}$$

$$\begin{aligned} AMI_{max}(u,v) &= \frac{NMI_{max}(u,v) - \mathbb{E}\left[NMI_{max}(u,v)\right]}{1 - \mathbb{E}\left[NMI_{max}(u,v)\right]} \\ &= \frac{I(u,v) - \mathbb{E}\left[I(u,v)\right]}{\max\left(H(u), H(v)\right) - \mathbb{E}\left[I(u,v)\right]} \end{aligned} \tag{21}$$

$$\mathbb{E}\left[I(u,v)\right] = \sum_{i=1}^{R}\sum_{j=1}^{C} \sum_{n_{i,j}=\max(a_i+b_j-N,0)}^{\min(a,b)} \frac{n_{ij}}{N} \log\left(\frac{N n_{ij}}{a_i b_j}\right) \frac{a_i! b_j! (N-a_i)! (N-b_j)!}{N! n_{ij}! (a_i - n_{ij})! (b_j - n_{ij})! (N - a_i - b_j + n_{ij}} \tag{22}$$

Homogeneity:

$$h = \begin{cases} 1 & \text{,if } H(u,v) = 0 \\ 1 - \frac{H(u|v)}{H(u)} & \text{otherwise} \end{cases} \tag{23}$$

Completeness:

$$c = \begin{cases} 1 & \text{,if } H(v,u) = 0 \\ 1 - \frac{H(v|u)}{H(v)} & \text{otherwise} \end{cases} \tag{24}$$

V Measure Score:

$$V_\beta = \frac{(1+\beta)hc}{\beta h + c} \tag{25}$$

.1 Algorithms

Principal Components and Multidimensional Scaling

Spectral Clustering

For the spectral clustering algorithm we formally introduce some graph notation. If not stated otherwise the following derivation follows [SOURCE]. In the following we consider a weighted and simple undirected graph.

$$G = \{V, E\} \tag{26}$$

$$V = \{v_1, \ldots, v_n\} \tag{27}$$

$$E = \{e_1, \ldots, e_n\} \tag{28}$$

Furthermore let the graph has a weighted and symetric ($|V| \times |V|$) adjacency matrix, such that:

$$\boldsymbol{W} = \begin{cases} w_{i,j} & \text{,if } v_i v_j \in E \\ 0 & \text{otherwise} \end{cases} \tag{29}$$

The *degree* of a node is defined as the sum of edge weights of connected nodes. Formmally we denote the degree of node i as:

$$d_i := \sum_{j=1}^{n} w_{ij} = \sum_{i=1}^{n} w_{ij} \tag{30}$$

Using the last expression we define matrix $\boldsymbol{D}$ as the diagonal matrix of the degress

$$\boldsymbol{D} := diag(\boldsymbol{d}) \tag{31}$$

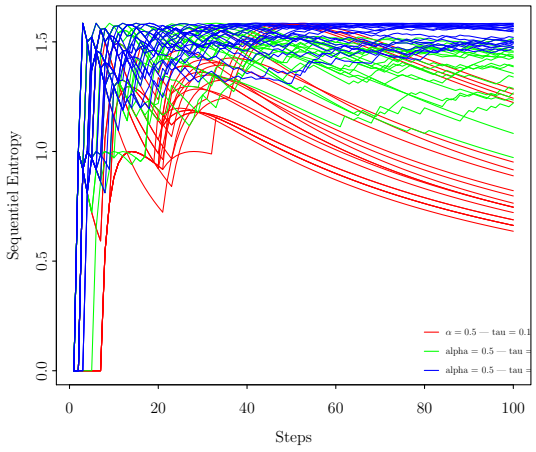
The algorithm works on the *Laplacian* matrix defined by:

$$\boldsymbol{L} := \boldsymbol{D} - \boldsymbol{W} \tag{32}$$

Former versions of the algorithm are applied on the graph laplcian. However, there were proposed newer versions using the so called *normalized laplacian*. Since also the python version is using this package we will focus on this version of the algorithhm. Following that the normalized graph laplacian is defined as:

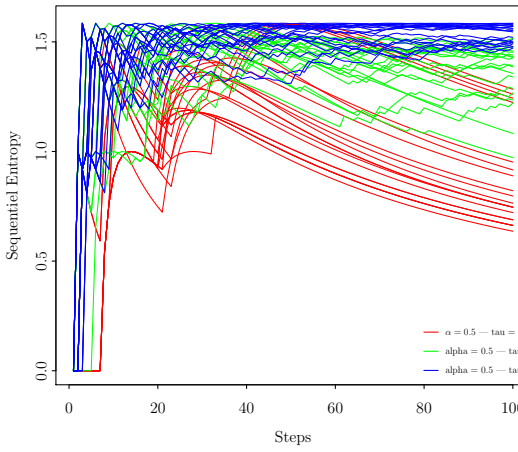
$$\boldsymbol{L}_{norm} := \boldsymbol{D}^{1/2} \boldsymbol{L} \boldsymbol{D}^{1/2} = \boldsymbol{I} - \boldsymbol{D}^{1/2} \boldsymbol{W} \boldsymbol{D}^{1/2} \tag{33}$$

$$p(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$$



(a) ACF

$$p(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$$



(b) PACF

Figure 2: ACF and PACF of MCSI